EACL 2021

The 16th Conference of the European Chapter of the Association for Computational Linguistics

Proceedings of the Conference

April 19 - 23, 2021

Platinum Sponsors

grammarly

FACEBOOK AI

Gold Sponsors

Bloomberg
Engineering

Bronze Sponsors

LegalForce

Babelscape

Supporter Sponsors

CELI
LANGUAGE TECHNOLOGY
A company of H-FARM

SAMSUNG

Diversity & Inclusion Champion Sponsors

Microsoft

Diversity & Inclusion Ally Sponsors

Adobe

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-954085-02-2

Message from the General Chair

Welcome to EACL 2021, the 16th conference of the European Chapter of the Association for Computational Linguistics! This year's conference is held from the 21st to the 23rd of April, 2021. While we were planning to hold the conference in Kyiv, due to the current COVID situation the conference is held entirely online. EACL 2021 is also an anchor conference to several workshops and tutorials, that are held on April 19th and 20th, also online.

This year's conference continues the successful growing trend of the community, and further requires a large organisational effort due to the COVID restrictions. We are learning how to organise and run conferences online, how to attend them and interact, and how to weave them into this strange suspension of our ordinary physical lives, that is our common current experience.

I would like to take the opportunity here to thank all the people involved, who have managed to pull through despite lockdowns, lack of child care, and the many other daily disruptions.

- Scientific programme chairs Jorg Tiedemann, from University of Helsinki and Reut Tsarfaty, from Bar Ilan University chaired a large scientific programme committee and introduced several innovative topics in the submissions.
- Workshop chairs Jonathan Berant, from Tel-Aviv University and Angeliki Lazaridou, from DeepMind selected the workshops, fourteen of which are affiliated to EACL 2021. Tutorial chairs Isabelle Augenstein, from University of Copenhagen and Ivan Habernal, from Technische Universitaet Darmstadt selected the tutorials. Demonstration chairs Dimitra Gkatzia, from Edinburgh Napier University and Djamé Seddah, University Paris la Sorbonne selected the system demonstrations. They have generated very interesting programmes, which add variety of topics and serve focussed subcommunities.
- The work of the younger members of our community have been the object of attention of our Student Research Workshop chairs Ionut-Teodor Sorodoc, from Pompeu Fabra University, Madhumita Sushil, from University of Antwerp and Ece Takmaz, from University of Amsterdam, and of their faculty advisor, Eneko Agirre, from the University of the Basque Country.
- Special thanks go to the publication chairs Valerio Basile, from the University of Turin and Tommaso Caselli, from the University of Groningen, who had to deal with our self-produced proceedings.
- Thank you also to our publicity chair Julie Weeds, from University of Sussex for making our conference known online, before and during the meeting.
- We belong, we know, to a scientific community of extreme demographic uniformity and we are striving to become more aware of issues of inclusivity and diversity. Thanks to our diversity and inclusion chair, Aline Villavicencio, University of Sheffield and Federal University of Rio Grande do Sul.
- When we decided to move to a virtual conference, we contacted a knowledgeable crowd of colleagues to form a Virtual Infrastructure Committee: Amirhossein Kazemnejad, Bruno Guillaume, Cyril Weerasooriya, Gisela Vallejo, Jan-Christoph Klie, Oles Dobosevych, Viktoria Kolomiets. The virtual organisation all happens thanks to them. Thanks especially to Jan-Christoph for sharing all the accumulated knowledge from past conferences and his senior advisor role for this one, and to Bonnie Webber, for sharing past experiences.

- We are very grateful to the local chairs from Grammarly and Ukrainian Catholic University, Viktoria Kolomiets, Dmytro Lider, Iryna Kotkalova, Oleksii Molchanovskyi, Oles Dobosevych. Thank you for offering to host the conference, manage the web site and be remarkably supportive and cooperative even when we had to decide to put off the opportunity to visit beautiful Kyiv.
- A large number of volunteers is being recruited as I write: thank you for your availability and enthusiasm. And thanks to the volunteer chair, Carolina Scarton, from the University of Sheffield, for hitting the ground running.
- We thank EACL 2021's sponsors for their very welcome contributions, which were obtained by the efforts of Raffaella Bernardi, our ACL sponsorship committee members for Europe. Their names and logos can be seen in the proceedings and on the conference web site.
- Thanks also to David Yarowsky and Priscilla Rasmussen from ACL for their help and advice.

Finally, and foremost, thank to all the authors and conference attendees that have made and will make this conference a success and source of inspiration.

EACL 2021 General Chair

Paola Merlo, University of Geneva, Switzerland

Message from the Program Chairs

Welcome to EACL 2021 — the 16th meeting of the European Chapter of the Association for Computational Linguistics. It has now been almost 4 years since EACL was last held, in Valencia, Spain, 2017, and it is the first time that the EACL conference will be held entirely virtually. This edition of the EACL conference comes at a challenging time for many in our community, due to consequences of the covid19 pandemic, but also at an exciting time for NLP researchers, seeing unprecedented growth and interest in the progress in our field, from both within and outside of our community. We are grateful for all the contributions and support that we have received, which allowed us to hold a successful and memorable event, despite having to cope with the challenges of covid and despite EACL being held and attended from remote.

EACL 2021 had received a record number of submissions compared to all past EACL events — exactly 1,400 submissions, an increase of 35

Organising a conference at this scale is a huge undertaking and the process is demanding, but exciting at the same time. We have been able to recruit a large number of reviewers with expertise that is necessary for making appropriate decisions in the many research areas that this conference covers, and we are beyond thankful for the tremendous support we got from the dedicated senior area chairs, area chairs and all reviewers involved in the selection process. Altogether, we have been fortunate to have been able to recruit 1691 reviewers, 149 area chairs and 34 senior area chairs, all professional experts in their fields.

We adopted the recent strategy of automatic COI detection and paper assignments to reviewers, according to their scholarly profiles and affiliations. This process is fairly new and has its own learning curve, but it comes with great advantages, in particular the ability to scale for the increasing number of submissions and reviewers in the *ACL conferences. At the same time, this process also demonstrated the importance of humans in the loop to make proper adjustments and (re)assignments of papers where the automatic decisions may be suboptimal. With the enormous help of the senior area chairs we could successfully run a detailed review process with at least three reviewers per paper, an author rebuttal period, and reviewer discussions. Thank you all for your efforts to ensure the scientific quality of the reviewing process and the resulting conference programme!

After the reviewing process, we could include a total of 326 excellent papers, referring to an acceptance rate of 24.7

The event will be organised in a similar fashion to other recent on-line conferences, emphasising pre-recorded talks with dedicated live question/answering sessions and interactive poster sessions in a virtual environment. Setting up the virtual event is yet another challenge, especially considering the various time zones around the world our keynotes, authors and participants come from. We opted for a morning session and a late-afternoon session according to the Central European calendar, to emphasise the European focus of the event . At the same time, in this EACL we introduce a certain novelty: all papers get assigned a slot at an interactive poster session that takes place at a time-slot that can reasonably be attended across all different time zones. We hope that this setup will provide the opportunity to truly immerse in the event, scientifically and socially, to increase both the impact of the different works and the opportunity of participants to network.

One of the important highlights in the conference is the lineup of renowned keynote speakers who we could attract to join EACL 2021. We are excited to have the following three speakers who have graciously accepted to provide lectures at the conference: Melanie Mitchell from the Santa Fe Institute, Fernanda Ferreira from the University of California, Davis and Marco Baroni from Facebook AI Research and the University of Trento. We are also delighted to announce a panel discussion on information accessibility and language technology in situations of emergency and ongoing crises, with international experts and representatives from the non-profit organization of Translators without Borders (Alp Öktem), the Masakhane NLP community, the University of Oxford (Scott Hale), the Bay Area NLP community (Robert Monarch) moderated by the language enthusiast and internet linguist Gretchen McCulloch.

Needless to say, an event like EACL would have not been possible without the efforts and contributions of a large number of people, to whom we are indebted:

- Our great 34 Senior area chairs, who meticulously managed the reviewing process in individual tracks, and led the discussion and selection process.
- and 149 area chairs, who carefully checked the papers, led reviewers' discussions, wrote meta-reviews and provided indispensable inputs for the selection process.
- Our 1691 reviewers, who wrote dedicated reviews and provided valuable feedback to the authors. Special thanks to reviewers who stepped in at the last minute to serve as emergency reviewers.
- Our Excellent Best Paper Committee for selecting the best EACL papers under a very tight schedule.
- The ACL Executive Review Committee. In particular, Amanda Stent, Arya McCarthy and Graham Neubig for making the COI detection and reviewer-paper assignment software available to us — these tools were instrumental in streamlining the paper assignment process. Special thanks for Graham Neubig and Trevor Cohn for technical advice in using these tools throughout the process.
- The 3343 authors who submitted their work to EACL 2021. While not being able to accept all submissions, it is their work that eventually makes up the exciting contributions and advances in our community.
- TACL editors-in-chief Ani Nenkova and Brian Roark, TACL Editorial Assistant Cindy Robinson, and CL Editor-in-Chief Hwee Tou Ng for coordinating the TACL and CL paper presentations with us.
- The Program co-Chairs of ACL 2020: Joel Tetreault, Natalie Schluter and Joyce Chai; and the Program co-Chairs of of EMNLP 2020: Trevor Cohn, Yulan He and Yang Liu, for sharing their experience and providing invaluable advice for the conference organization and the PC-chairing activities.
- Our Publication Chairs, Valerio Basile and Tommaso Caselli, for the efficient and streamlined production of the EACL conference proceedings.
- Our Publicity Chair, Julie Weeds and our Web Infrastructure Chair, Viktoria Kolomiets, for effectively and efficiently taking care of all event communication and PR aspects of the conference.
- Jarda Fikr from SlidesLive, for coordinating the presentations and recordings by the authors with the SlideLive team.
- Rich Gerber at SoftConf, for extremely quick responses on any email inquiry or emerging difficulties encountered with the START system.
- Our students, interns, postdocs, colleagues, and families. Sorry for not being available to you as much as we hoped to, especially in these crazy times of global pandemic. We promise to make up for it!
- Last but not least, we wish to express our deepest thanks to our General Chair Paola Merlo. She has been extremely professional and supportive from the start, providing us with solid advice while completely trusting us and providing flexibility and room to innovate. From the initial plan to have

EACL as a physical conference all the way to its realization as a virtual event, Paola has led and coordinated all efforts through the thick and thin of covid-related uncertainties, confidently leading to this successful event.

Our deepest gratitude to all of you. We hope you will enjoy this conference experience.

EACL 2021 Program Committee Co-Chairs

Reut Tsarfaty, Bar-Ilan University

Jörg Tiedemann, University of Helsinki

Organizing Committee

General Chair:

Paola Merlo, University of Geneva

Program Chairs:

Jorg Tiedemann, University of Helsinki
Reut Tsarfaty, Bar Ilan University

Tutorial Chairs:

Isabelle Augenstein, University of Copenhagen
Ivan Habernal, Technische Universitaet Darmstadt

Workshop Chairs:

Jonathan Berant, Tel-Aviv University
Angeliki Lazaridou, DeepMind

Publication Chairs:

Valerio Basile, University of Turin
Tommaso Caselli, University of Groningen

Student Research Workshop Chairs:

Ionut-Teodor Sorodoc, Pompeu Fabra University
Madhumita Sushil, University of Antwerp
Ece Takmaz, University of Amsterdam

Faculty Advisor to the Student Research Workshop:

Eneko Agirre, University of the Basque Country

Demonstration Chairs:

Dimitra Gkatzia, Edinburgh Napier University
Djamé Seddah, University Paris la Sorbonne

Diversity & Inclusion (D&I) Chair:

Aline Villavicencio, University of Sheffield and Federal University of Rio Grande do Sul

Publicity Chair:

Julie Weeds, University of Sussex

Virtual Infrastructure Committee:

Amirhossein Kazemnejad, Mila
Bruno Guillaume, LORIA, Inria NGE
Carolina Scarton, University of Sheffield
Cyril Weerasooriya, Rochester Institute of Technology
Gisela Vallejo, Independent researcher
Jan-Christoph Klie, UKP Lab, Technical University of Darmstadt
Oles Dobosevych, Ukrainian Catholic University

Viktoria Kolomiets, Grammarly

Local Chairs:

Viktoria Kolomiets, Grammarly
Dmytro Lider, Grammarly
Iryna Kotkalova, Grammarly
Oleksii Molchanovskyi, Ukrainian Catholic University
Oles Dobosevych, Ukrainian Catholic University

Program Committee

Program Chairs:

Jorg Tiedemann, University of Helsinki
Reut Tsarfaty, Bar Ilan University

Senior Area Chairs and Area Chairs:

Senior area chairs are in bold.

Computational Social Science and Social Media

Oren Tsur, Dirk Hovy, Sara Rosenthal, Dan Goldwasser, Brendan O'Connor, Svitlana Volkova, Djame Seddah

Discourse and Pragmatics

Bonnie Webber, Shafiq Joty, Yufang Hou, Sharid Loaiciga, Mohsen Mesgar

Dialogue and Interactive Systems

Matthew Purver, **Verena Rieser**, Layla El Asri, Casey Kennington, Pierre Lison, Luciana Benotti, Stefan Ultes, Ravi Shekhar, Malihe Alikhani, Svetlana Stoyanchev, Julian Hough, Arash Eshghi, Paweł Budzianowski, Christine Howes, Jason Williams, Nikola Mrkšić

Document analysis, Text Categorization and Topic Models

Udo Kruschwitz, Jochen Leidner, Andrew Yates, Anders Søgaard, Mark Stevenson, Iris Hendrickx, Jeff Dalton, Tony Russell-Rose

Generation and Summarization

Anya Belz, **Annie Louis**, Claire Gardent, Sebastian Gehrmann, Xiaojun Wan, Shashi Narayan, Manabu Okumura, Laura Perez-Beltrachini, Katja Markert, Angela Fan, Jackie Chi Kit Cheung, Fei Liu

Green and Sustainable NLP

Roy Schwartz, **Emma Strubell**, Jesse Dodge, Dallas Card, Angela Fan, Anna Rogers, Aleksandr Drozd

Information Retrieval, Search, Question Answering

Maarten de Rijke, **Mounia Lalmas**, Suzan Verberne, Aleksandr Chuklin, Gabriella Pasi, Julia Kiseleva, Julio Gonzalo, Azadeh Shakery, Theodora Tsikrika

Information Extraction and Text Mining

Antoine Doucet, Jing Jiang, Adam Jatowt, Jaap Kamps, Paolo Rosso, Efstathios Stamatatos, Kang Liu, Cornelia Caragea

Interpretability and Model Analysis in NLP

Arianna Bisazza, **Aurelie Herbelot**, Raffaella Bernardi, German Kruszewski, Dieuwke Hupkes, Alessandro Raganato, Lisa Beinborn

Language Resources and Evaluation

Barbara Plank, **Vera Demberg**, Asif Ekbal, Yvette Graham, Henning Wachsmuth, Ines Rehbein, Alexis Palmer, Ondřej Dušek, Manfred Stede, Taylor Berg-Kirkpatrick

Language Grounding to Vision, Robots, and other

Marie Sien Moens, Iacer Calixto, Douwe Kiela, Jean Oh

Linguistic Theories, Cognitive modeling and Psycholinguistics

Afra Alishahi, Roger Levy, Emily Prud'hommeaux, Antske Fokkens, Cassandra Jacobs

Machine Learning in NLP

Shay Cohen, **Andre F. Martins**, Matthias Galle, Andreas Vlachos, Karl Stratos, Xavier Carreras, Lei Yu, Julia Kreutzer, Philip John Gorinski, Reza Haffari, Carolin Lawrence, Edwin Simpson, Zita Marinho, Jasmijn Bastings

Machine translation

Martin Volk, **Mark Fishel**, Inguna Skadina, Marco Turchi, Dagmar Gromann, Andrei Popescu-Belis, Alex Fraser, Marcello Federico, Maja Popovic, Antonio Toral, Thierry Etchegoyhen

Multidisciplinary and COI

Marco Kuhlmann, Marco Guerini, Aurélie Névéol, Enrico Santus

Multilinguality

Roi Reichart, Omri Abend, Ivan Vulić , Sebastian Ruder, Goran Glavaš, Lea Frermann

NLP Applications for Crisis Managment and Emergency Situations

Robert Munro, Rada Mihalcea, Graham Neubig, Antonios Anastasopoulos, Ishan Jindal

Semantics: lexical

Sebastian Pado, **Marianna Apidianaki**, Gemma Boleda, Jose Camacho Collados, Emmanuele Chersoni, Anne Cocos, Tim Van de Cruys, Katrin Erk, Manaal Faruqui, Alexander Panchenko, Lonneke van der Plas, Vered Shwartz

Semantics: sentence level and other areas

James Henderson, Mike Lewis, Wai Lam, Nafise Sadat Moosavi, Daniel Khashabi, Michael

Roth, Adam Poliak, Swabha Swayamdipta

Sentiment Analysis and Argument Mining

Roman Klinger, **Viviana Patti**, Jeremy Barnes, Steffen Eger, Orphée De Clercq, Els Lefever, Farah Benamara, Tamar Solorio Solorio, Serena Villata, Svetlana Kiritchenko, Torsten Zesch

Phonology, Morphology, and Word Segmentation

Ryan Cotterell, Adina Williams, Johannes Bjerva, Ekaterina Vylomova, Edoardo Ponti, Christo Kirov

Speech

Mikko Kurrimo, Tanel Alumäe, Ebru Arisoy, Dhananjaya Gowda

Tagging, Chunking, Syntax, and Parsing

Joakim Nivre, **Carlos Gómez**, Miguel Ballesteros, Jonas Kuhn, Zeljko Agic, Jennifer Foster, Yue Zhang, Kenji Sagae

Outstanding Members of the PC:

Outstanding Area Chairs

Jennifer Foster, Ebru Arisoy, Tanel Alumäe, Dhananjaya Gowda, Jeremy Barnes, Svetlana Kiritchenko, Vered Shwartz, Katrin Erk, Omri Abend, Lea Frermann, Goran Glavaš, Ivan Vulić, Sebastian Ruder, Aurélie Névéol, Alex Fraser, Marcello Federico, Antonio Toral, Maja Popovic, Andrei Popescu-Belis, Dagmar Gromann, Marco Turchi, Carolin Lawrence, Julia Kreutzer, Jazmijn Bastings, Andreas Vlachos, Edwin Simpson, Ondrej Dusek, "Cassandra L. " Jacobs, Jing Jiang, Jaap Kamps, Adam Jatowt, Paolo Rosso, Kang Liu, Efstathios Stamatatos, Aleksandr Chuklin, Gabriella Pasi, Suzan Verberne, Laure Soulier, Olivier Sprangers, Iacer Calixto, Douwe Kiela, Jean Oh, Claire Gardent, Fei Liu, Sebastian Gehrmann, Angela Fan, Katja Merkert, Anders Søgaard, Andrew Yates, Yufang Hou, Shafiq Joty, Sharid Loáiciga, Mohsen Mesgar, Jason Williams, Luciana Benotti, Casey Kennington, Svetlana Stoyanchev, Djame Seddah.

Outstanding Reviewers

James Barry, Aditya Bhargava, Mathieu Dehouck, Miryam de Lhoneux, Timothy Dozat, Agnieszka Falenska, Nikita Kitaev, Giorgio Satta, David Vilares, Joachim Wagner, Johannes Daxenberger, Christopher Hidey, Udo Hahn, Caroline Brun, Andrew Moore, Esther van den Berg, Wenbo Wang, Mario Sänger, Lilja Øvrelid, Erik Velldal, Aditya Joshi, Chloé Clavel, Cynthia Van Hee, Daniel Dahlmeier, Shachar Mirkin, Forrest Sheng Bao, Patrick Paroubek, Gilles Jacobs, Thomas Haider, Anette Frank, Florian Mai, Yinfei Yang, Mathias Creutz, Hiroki Ouchi, Esther Seyffarth, Francis Ferraro, Pengxiang Cheng, James H. Martin, Ji-Ung Lee, Panupong Pasupat, Julian Michael, Aina Garí Soler, Delphine Bernhard, Eduard Hovy, Shoaib Jameel, Ingrid Falk, Guy Emerson, Leonardo Zilio, Diarmuid Ó Séaghdha, Enrico Santus, Olivier Ferret, Shiva Taslimipoor, Aditya Gupta, Daniil Sorokin, Pavankumar Reddy Muddireddy, Ellie Pavlick, Carlos Ramisch, Thomas Kober, Tristan Miller, Timothy Baldwin, Mikel Artetxe, Xilun Chen, Benjamin Heinzerling, Yova Kementchedjhieva, Shruti

Rijhwani, Mareike Hartmann, Edoardo Maria Ponti, Shuyan Zhou, Shuly Wintner, Haim Dubossarsky, Yulia Tsvetkov, Takashi Wada, Richart Sproat, Diana McCarthy, Telmo Pires, David Reitter, Saranya Venkatraman, Mathias Müller, Rico Sennrich, Mirjam Sepesy Maucec, Alberto Poncelas, Mattia A. Di Gangi, Raivis Skadiņš, Dusan Varis, Nikolay Bogoychev, Maarit Koponen, Jean-Yves Antoine, Marcel Bollmann, Bruno Martins, Naoki Otani, Agata Savary, Yves Scherrer, Miguel A. Alonso, Cristina Bosco, Tommaso Caselli, Joachim Wagner, Johnny Wei, Dian Yu, Marcos Zampieri, Heike Zinkmeister, Joel Tetreault, Emma Manning, Jakob Prange, Stefanie Dipper, Leila Arras, Matthijs Westera, John Hewitt, Sandro Pezzelle, Allyson Ettinger, Guy Emerson, Ludovic Tanguy, Pia Sommerauer, Thomas McCoy, Yonatan Belinkov, Noah Smith, Denis Emelin, Jindřich Libovický, Xiaochuang Han, Georgeta Bordea, Emanuela Boros, Yung-Chun Chang, John Chen, Hsin-Hsi Chen, Jennifer D'Souza, Xiang Dai, Herve Dejean, Luciano Del Corro, Gal Dias, Ismail El Maarouf, Ahmed El-Kishky, Rob Koeling, Bhushan Kotnis, Gal Lejeune, Fei Li, Xiang Li, Xiao Liu, Yuanliang Meng, Lidia Pivovarova, Julien Tourille, Guorui Zhou, Ozam Caglayan, Steven Bedrick, Sofia Serrano, Tim Dettmers, Louis Martin, Chris Quirk, Zhihan Zhang, Zhihan Zhang, Nadjet Bouayad-Agha, Deng Cai, Xutan Peng, Gerasimos Lampouras, Aman Madaan, Michael Elhadad, Ji Ma, Yao Fu, Mark Cieliebak, Jiawei Zhou, Jack Hessel, Timothy Miller, Laure Thompson, Mijail Kabadjov, Cyril Goutte, Luca Soldaini, Berfin Aktaş, Chloé Braud, Debopam Das, Junyi Jessy Li, Ana Marasović, Tatjana Scheffler, Noriki Nishida, Thomas Brox Røst, David DeVault, Kai Yu, Nurul Lubis, Sakriani Sakti, Shikib Mehri, Raghav Gupta, Andrea Kahn, Takenobu Tokunaga, David Traum, Zdeněk Kasner, Alborz Geramifard, Jan Alexandersson.

Table of Contents

Conference Program

Unsupervised Sentence-embeddings by Manifold Approximation and Projection
Subhradeep Kayal

Contrastive Multi-document Question Generation
Woon Sang Cho, Yizhe Zhang, Sudha Rao, Asli Celikyilmaz, Chenyan Xiong, Jianfeng Gao, Mengdi Wang and Bill Dolan

Disambiguatory Signals are Stronger in Word-initial Positions
Tiago Pimentel, Ryan Cotterell and Brian Roark

On the (In)Effectiveness of Images for Text Classification
Chunpeng Ma, Aili Shen, Hiyori Yoshikawa, Tomoya Iwakura, Daniel Beck and Timothy Baldwin

If you've got it, flaunt it: Making the most of fine-grained sentiment annotations
Jeremy Barnes, Lilja Øvrelid and Erik Velldal

Keep Learning: Self-supervised Meta-learning for Learning from Inference
Akhil Kedia and SAI CHETAN CHINTHAKINDI

ResPer: Computationally Modelling Resisting Strategies in Persuasive Conversations
Ritam Dutt, Sayan Sinha, Rishabh Joshi, Surya Shekhar Chakraborty, Meredith Riggs, Xinru Yan, Haogang Bao and Carolyn Rose

BERxiT: Early Exiting for BERT with Better Fine-Tuning and Extension to Regression
Ji Xin, Raphael Tang, Yaoliang Yu and Jimmy Lin

Telling BERT's Full Story: from Local Attention to Global Aggregation
Damian Pascual, Gino Brunner and Roger Wattenhofer

Effects of Pre- and Post-Processing on type-based Embeddings in Lexical Semantic Change Detection
Jens Kaiser, Sinan Kurtyigit, Serge Kotchourko and Dominik Schlechtweg

The Gutenberg Dialogue Dataset
Richard Csaky and Gábor Recski

On the Calibration and Uncertainty of Neural Learning to Rank Models for Conversational Search
Gustavo Penha and Claudia Hauff

No Day Set (continued)

No Day Set (continued)

No Day Set (continued)

MONAH: Multi-Modal Narratives for Humans to analyze conversations
Joshua Y. Kim, Kalina Yacef, Greyson Kim, Chunfeng Liu, Rafael Calvo and Silas Taylor

Does Typological Blinding Impede Cross-Lingual Sharing?
Johannes Bjerva and Isabelle Augenstein

AdapterFusion: Non-Destructive Task Composition for Transfer Learning
Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho and Iryna Gurevych

CHOLAN: A Modular Approach for Neural Entity Linking on Wikipedia and Wikidata
Manoj Prabhakar Kannan Ravi, Kuldeep Singh, Isaiah Onando Mulang', Saeedeh Shekarpour, Johannes Hoffart and Jens Lehmann

Grounding as a Collaborative Process
Luciana Benotti and Patrick Blackburn

Does She Wink or Does She Nod? A Challenging Benchmark for Evaluating Word Understanding of Language Models
Lutfi Kerem Senel and Hinrich Schütze

Joint Coreference Resolution and Character Linking for Multiparty Conversation
Jiaxin Bai, Hongming Zhang, Yangqiu Song and Kun Xu

Improving Factual Consistency Between a Response and Persona Facts
Mohsen Mesgar, Edwin Simpson and Iryna Gurevych

PolyLM: Learning about Polysemy through Language Modeling
Alan Ansell, Felipe Bravo-Marquez and Bernhard Pfahringer

Predicting Treatment Outcome from Patient Texts:The Case of Internet-Based Cognitive Behavioural Therapy
Evangelia Gogoulou, Magnus Boman, Fehmi Ben Abdesslem, Nils Hentati Isacsson, Viktor Kaldo and Magnus Sahlgren

Scalable Evaluation and Improvement of Document Set Expansion via Neural Positive-Unlabeled Learning
Alon Jacovi, Gang Niu, Yoav Goldberg and Masashi Sugiyama

The Role of Syntactic Planning in Compositional Image Captioning
Emanuele Bugliarello and Desmond Elliott

No Day Set (continued)

Is “hot pizza" Positive or Negative? Mining Target-aware Sentiment Lexicons
Jie Zhou, Yuanbin Wu, Changzhi Sun and Liang He

Quality Estimation without Human-labeled Data
Yi-Lin Tuan, Ahmed El-Kishky, Adithya Renduchintala, Vishrav Chaudhary, Francisco Guzmán and Lucia Specia

How Fast can BERT Learn Simple Natural Language Inference?
Yi-Chung Lin and Keh-Yih Su

GRIT: Generative Role-filler Transformers for Document-level Event Entity Extraction
Xinya Du, Alexander Rush and Claire Cardie

Cross-lingual Entity Alignment with Incidental Supervision
Muhao Chen, Weijia Shi, Ben Zhou and Dan Roth

Query Generation for Multimodal Documents
kyungho kim, Kyungjae Lee, Seung-won Hwang, Young-In Song and seungwook lee

End-to-End Argument Mining as Biaffine Dependency Parsing
Yuxiao Ye and Simone Teufel

FakeFlow: Fake News Detection by Modeling the Flow of Affective Information
Bilal Ghanem, Simone Paolo Ponzetto, Paolo Rosso and Francisco Rangel

CTC-based Compression for Direct Speech Translation
Marco Gaido, Mauro Cettolo, Matteo Negri and Marco Turchi

A Crowdsourced Open-Source Kazakh Speech Corpus and Initial Speech Recognition Baseline
Yerbolat Khassanov, Saida Mussakhojayeva, Almas Mirzakhmetov, Alen Adiyev, Mukhamet Nurpeiissov and Huseyin Atakan Varol

TDMSci: A Specialized Corpus for Scientific Literature Entity Tagging of Tasks Datasets and Metrics
Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin and Debasis Ganguly

Top-down Discourse Parsing via Sequence Labelling
Fajri Koto, Jey Han Lau and Timothy Baldwin

No Day Set (continued)

Does the Order of Training Samples Matter? Improving Neural Data-to-Text Generation with Curriculum Learning
Ernie Chang, Hui-Syuan Yeh and Vera Demberg

TrNews: Heterogeneous User-Interest Transfer Learning for News Recommendation
Guangneng Hu and Qiang Yang

Dialogue Act-based Breakdown Detection in Negotiation Dialogues
Atsuki Yamaguchi, Kosui Iwasa and Katsuhide Fujita

Neural Data-to-Text Generation with LM-based Text Augmentation
Ernie Chang, Xiaoyu Shen, Dawei Zhu, Vera Demberg and Hui Su

Self-Training Pre-Trained Language Models for Zero- and Few-Shot Multi-Dialectal Arabic Sequence Labeling
Muhammad Khalifa, Muhammad Abdul-Mageed and Khaled Shaalan

Multiple Tasks Integration: Tagging, Syntactic and Semantic Parsing as a Single Task
Timothée Bernard

Coordinate Constructions in English Enhanced Universal Dependencies: Analysis and Computational Modeling
Stefan Grünewald, Prisca Piccirilli and Annemarie Friedrich

Ellipsis Resolution as Question Answering: An Evaluation
Rahul Aralikatte, Matthew Lamm, Daniel Hardt and Anders Søgaard

Jointly Improving Language Understanding and Generation with Quality-Weighted Weak Supervision of Automatic Labeling
Ernie Chang, Vera Demberg and Alex Marin

Continuous Learning in Neural Machine Translation using Bilingual Dictionaries
Jan Niehues

Adv-OLM: Generating Textual Adversaries via OLM
Vijit Malik, Ashwani Bhat and Ashutosh Modi

Conversational Question Answering over Knowledge Graphs with Transformer and Graph Attention Networks
Endri Kacupaj, Joan Plepi, Kuldeep Singh, Harsh Thakkar, Jens Lehmann and Maria Maleshkova

No Day Set (continued)

DRAG: Director-Generator Language Modelling Framework for Non-Parallel Author Stylized Rewriting
Hrituraj Singh, Gaurav Verma, Aparna Garimella and Balaji Vasan Srinivasan

Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering
Gautier Izacard and Edouard Grave

Clinical Outcome Prediction from Admission Notes using Self-Supervised Knowledge Integration
Betty van Aken, Jens-Michalis Papaioannou, Manuel Mayrdorfer, Klemens Budde, Felix Gers and Alexander Loeser

Combining Deep Generative Models and Multi-lingual Pretraining for Semi-supervised Document Classification
Yi Zhu, Ehsan Shareghi, Yingzhen Li, Roi Reichart and Anna Korhonen

Multi-facet Universal Schema
Rohan Paul, Haw-Shiuan Chang and Andrew McCallum

Exploring Transitivity in Neural NLI Models through Veridicality
Hitomi Yanaka, Koji Mineshima and Kentaro Inui

A Neural Few-Shot Text Classification Reality Check
Thomas Dopierre, Christophe Gravier and Wilfried Logerais

Multilingual Machine Translation: Closing the Gap between Shared and Language-specific Encoder-Decoders
Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa and Mikel Artetxe

Clustering Word Embeddings with Self-Organizing Maps. Application on LaRoSeDa - A Large Romanian Sentiment Data Set
Anca Tache, Gaman Mihaela and Radu Tudor Ionescu

Elastic weight consolidation for better bias inoculation
James Thorne and Andreas Vlachos

Hierarchical Multi-head Attentive Network for Evidence-aware Fake News Detection
Nguyen Vo and Kyumin Lee

Identifying Named Entities as they are Typed
Ravneet Arora, Chen-Tse Tsai and Daniel Preotiuc-Pietro

No Day Set (continued)

SANDI: Story-and-Images Alignment
Sreyasi Nag Chowdhury, Simon Razniewski and Gerhard Weikum

Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets
Patrick Lewis, Pontus Stenetorp and Sebastian Riedel

El Volumen Louder Por Favor: Code-switching in Task-oriented Semantic Parsing
Arash Einolghozati, Abhinav Arora, Lorena Sainz-Maza Lecanda, Anuj Kumar and Sonal Gupta

Generating Syntactically Controlled Paraphrases without Using Annotated Parallel Pairs
Kuan-Hao Huang and Kai-Wei Chang

Data Augmentation for Hypernymy Detection
Thomas Kober, Julie Weeds, Lorenzo Bertolini and David Weir

Few-shot learning through contextual data augmentation
Farid Arthaud, Rachel Bawden and Alexandra Birch

Zero-shot Generalization in Dialog State Tracking through Generative Question Answering
Shuyang Li, Jin Cao, Mukund Sridhar, Henghui Zhu, Shang-Wen Li, Wael Hamza and Julian McAuley

Zero-shot Neural Passage Retrieval via Domain-targeted Synthetic Question Generation
Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall and Ryan McDonald

Discourse-Aware Unsupervised Summarization for Long Scientific Documents
Yue Dong, Andrei Mircea Romascanu and Jackie Chi Kit Cheung

MIDAS: A Dialog Act Annotation Scheme for Open Domain HumanMachine Spoken Conversations
Dian Yu and Zhou Yu

Analyzing the Forgetting Problem in Pretrain-Finetuning of Open-domain Dialogue Response Models
Tianxing He, Jun Liu, Kyunghyun Cho, Myle Ott, Bing Liu, James Glass and Fuchun Peng

Leveraging End-to-End ASR for Endangered Language Documentation: An Empirical Study on Yolóxochitl Mixtec
Jiatong Shi, Jonathan D. Amith, Rey Castillo García, Esteban Guadalupe Sierra, Kevin Duh and Shinji Watanabe

No Day Set (continued)

Mode Effects' Challenge to Authorship Attribution
Haining Wang, Allen Riddell and Patrick Juola

Generative Text Modeling through Short Run Inference
Bo Pang, Erik Nijkamp, Tian Han and Ying Nian Wu

Detecting Extraneous Content in Podcasts
Sravana Reddy, Yongze Yu, Aasish Pappu, Aswin Sivaraman, Rezvaneh Rezapour and Rosie Jones

Randomized Deep Structured Prediction for Discourse-Level Processing
Manuel Widmoser, Maria Pacheco, Jean Honorio and Dan Goldwasser

Automatic Data Acquisition for Event Coreference Resolution
Prafulla Kumar Choubey and Ruihong Huang

Joint Learning of Representations for Web-tables, Entities and Types using Graph Convolutional Network
Aniket Pramanick and Indrajit Bhattacharya

Multimodal Text Style Transfer for Outdoor Vision-and-Language Navigation
Wanrong Zhu, Xin Wang, Tsu-Jui Fu, An Yan, Pradyumna Narayana, Kazoo Sone, Sugato Basu and William Yang Wang

ECOL-R: Encouraging Copying in Novel Object Captioning with Reinforcement Learning
Yufei Wang, Ian Wood, Stephen Wan and Mark Johnson

Enriching Non-Autoregressive Transformer with Syntactic and Semantic Structures for Neural Machine Translation
Ye Liu, Yao Wan, Jianguo Zhang, Wenting Zhao and Philip Yu

NLQuAD: A Non-Factoid Long Question Answering Data Set
Amir Soleimani, Christof Monz and marcel worring

Debiasing Pre-trained Contextualised Embeddings
Masahiro Kaneko and Danushka Bollegala

Language Models for Lexical Inference in Context
Martin Schmitt and Hinrich Schütze

Few-Shot Semantic Parsing for New Predicates
Zhuang Li, Lizhen Qu, shuo huang and Gholamreza Haffari

Alternating Recurrent Dialog Model with Large-scale Pre-trained Language Models
Qingyang Wu, Yichi Zhang, Yu Li and Zhou Yu

On the Evaluation of Vision-and-Language Navigation Instructions
Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge and Eugene Ie

Cross-lingual Visual Pre-training for Multimodal Machine Translation
Ozan Caglayan, Menekse Kuyu, Mustafa Sercan Amac, Pranava Madhyastha, Erkut Erdem, Aykut Erdem and Lucia Specia

Memorization vs. Generalization : Quantifying Data Leakage in NLP Performance Evaluation
Aparna Elangovan, Jiayuan He and Karin Verspoor

An Expert Annotated Dataset for the Detection of Online Misogyny
Ella Guest, Bertie Vidgen, Alexandros Mittos, Nishanth Sastry, Gareth Tyson and Helen Margetts

WikiMatrix: Mining 135M Parallel Sentences in 1620 Language Pairs from Wikipedia
Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong and Francisco Guzmán

ChEMU-Ref: A Corpus for Modeling Anaphora Resolution in the Chemical Domain
Biaoyan Fang, Christian Druckenbrodt, Saber A Akhondi, Jiayuan He, Timothy Baldwin and Karin Verspoor

Syntactic Nuclei in Dependency Parsing – A Multilingual Exploration
Ali Basirat and Joakim Nivre

Searching for Search Errors in Neural Morphological Inflection
Martina Forster, Clara Meister and Ryan Cotterell

Quantifying Appropriateness of Summarization Data for Curriculum Learning
Ryuji Kano, Takumi Takahashi, Toru Nishino, Motoki Taniguchi, Tomoki Taniguchi and Tomoko Ohkuma

Evaluating language models for the retrieval and categorization of lexical collocations
Luis Espinosa Anke, Joan Codina-Filba and Leo Wanner

No Day Set (continued)

BART-TL: Weakly-Supervised Topic Label Generation
Cristian Popa and Traian Rebedea

Dynamic Graph Transformer for Implicit Tag Recognition
Yi-Ting Liou, Chung-Chi Chen, Hen-Hsen Huang and Hsin-Hsi Chen

Implicit Unlikelihood Training: Improving Neural Text Generation with Reinforcement Learning
Evgeny Lagutin, Daniil Gavrilov and Pavel Kalaidin

Civil Rephrases Of Toxic Texts With Self-Supervised Transformers
Léo Laugier, John Pavlopoulos, Jeffrey Sorensen and Lucas Dixon

Generating Weather Comments from Meteorological Simulations
Soichiro Murakami, Sora Tanaka, Masatsugu Hangyo, Hidetaka Kamigaito, Kotaro Funakoshi, Hiroya Takamura and Manabu Okumura

SICK-NL: A Dataset for Dutch Natural Language Inference
Gijs Wijnholds and Michael Moortgat

A phonetic model of non-native spoken word processing
Yevgen Matusevych, Herman Kamper, Thomas Schatz, Naomi Feldman and Sharon Goldwater

Bootstrapping Relation Extractors using Syntactic Search by Examples
Matan Eyal, Asaf Amrami, Hillel Taub-Tabib and Yoav Goldberg

Towards a Decomposable Metric for Explainable Evaluation of Text Generation from AMR
Juri Opitz and Anette Frank

The Source-Target Domain Mismatch Problem in Machine Translation
Jiajun Shen, Peng-Jen Chen, Matthew Le, Junxian He, Jiatao Gu, Myle Ott, Michael Auli and Marc'Aurelio Ranzato

Cross-Topic Rumor Detection using Topic-Mixtures
Xiaoying Ren, Jing Jiang, Ling Min Serena Khoo and Hai Leong Chieu

Understanding Pre-Editing for Black-Box Neural Machine Translation
Rei Miyata and Atsushi Fujita

No Day Set (continued)

RelWalk - A Latent Variable Model Approach to Knowledge Graph Embedding
Danushka Bollegala, Huda Hakami, Yuichi Yoshida and Ken-ichi Kawarabayashi

Few-shot Learning for Slot Tagging with Attentive Relational Network
Cennet Oguz and Ngoc Thang Vu

SpanEmo: Casting Multi-label Emotion Classification as Span-prediction
Hassan Alhuzali and Sophia Ananiadou

Exploiting Position and Contextual Word Embeddings for Keyphrase Extraction from Scientific Papers
Krutarth Patel and Cornelia Caragea

Benchmarking Machine Reading Comprehension: A Psychological Perspective
Saku Sugawara, Pontus Stenetorp and Akiko Aizawa

Multilingual Neural Machine Translation with Deep Encoder and Multiple Shallow Decoders
Xiang Kong, Adithya Renduchintala, James Cross, Yuqing Tang, Jiatao Gu and Xian Li

With Measured Words: Simple Sentence Selection for Black-Box Optimization of Sentence Compression Algorithms
Yotam Shichel, Meir Kalech and Oren Tsur

WiC-TSV: An Evaluation Benchmark for Target Sense Verification of Words in Context
Anna Breit, Artem Revenko, Kiamehr Rezaee, Mohammad Taher Pilehvar and Jose Camacho-Collados

Self-Supervised and Controlled Multi-Document Opinion Summarization
Hady Elsahar, Maximin Coavoux, Jos Rozen and Matthias Gallé

NewsMTSC: A Dataset for (Multi-)Target-dependent Sentiment Classification in Political News Articles
Felix Hamborg and Karsten Donnay

Cross-lingual Contextualized Topic Models with Zero-shot Learning
Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza and Elisabetta Fersini

Dependency parsing with structure preserving embeddings
Ákos Kádár, Lan Xiao, Mete Kemertas, Federico Fancellu, Allan Jepson and Afsaneh Fazly

No Day Set (continued)

Active Learning for Sequence Tagging with Deep Pre-trained Models and Bayesian Uncertainty Estimates
Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov and Alexander Panchenko

MultiHumES: Multilingual Humanitarian Dataset for Extractive Summarization
Jenny Paola Yela-Bello, Ewan Oglethorpe and Navid Rekabsaz

Learning From Revisions: Quality Assessment of Claims in Argumentation at Scale
Gabriella Skitalinskaya, Jonas Klaff and Henning Wachsmuth

Few Shot Dialogue State Tracking using Meta-learning
Saket Dingliwal, Shuyang Gao, Sanchit Agarwal, Chien-Wei Lin, Tagyoung Chung and Dilek Hakkani-Tur

BERT Prescriptions to Avoid Unwanted Headaches: A Comparison of Transformer Architectures for Adverse Drug Event Detection
Beatrice Portelli, Edoardo Lenzi, Emmanuele Chersoni, Giuseppe Serra and Enrico Santus

Semantic Parsing of Disfluent Speech
Priyanka Sen and Isabel Groves

Joint Energy-based Model Training for Better Calibrated Natural Language Understanding Models
Tianxing He, Bryan McCann, Caiming Xiong and Ehsan Hosseini-Asl

What Sounds “Right" to Me? Experiential Factors in the Perception of Political Ideology
Qinlan Shen and Carolyn Rose

Language Models as Knowledge Bases: On Entity Representations, Storage Capacity, and Paraphrased Queries
Benjamin Heinzerling and Kentaro Inui

Globalizing BERT-based Transformer Architectures for Long Document Summarization
quentin grail, Julien PEREZ and Eric Gaussier

Through the Looking Glass: Learning to Attribute Synthetic Text Generated by Language Models
Shaoor Munir, Brishna Batool, Zubair Shafiq, Padmini Srinivasan and Fareed Zaffar

We Need To Talk About Random Splits
Anders Søgaard, Sebastian Ebert, Jasmijn Bastings and Katja Filippova

No Day Set (continued)

No Day Set (continued)

Boosting Low-Resource Biomedical QA via Entity-Aware Masking Strategies
Gabriele Pergola, Elena Kochkina, Lin Gui, Maria Liakata and Yulan He

Attention-based Relational Graph Convolutional Network for Target-Oriented Opinion Words Extraction
Junfeng Jiang, An Wang and Akiko Aizawa

“Laughing at you or with you”: The Role of Sarcasm in Shaping the Disagreement Space
Debanjan Ghosh, Ritvik Shrivastava and Smaranda Muresan

Learning Relatedness between Types with Prototypes for Relation Extraction
Lisheng Fu and Ralph Grishman

I Beg to Differ: A study of constructive disagreement in online conversations
Christine De Kock and Andreas Vlachos

Acquiring a Formality-Informed Lexical Resource for Style Analysis
Elisabeth Eder, Ulrike Krieg-Holz and Udo Hahn

Probing into the Root: A Dataset for Reason Extraction of Structural Events from Financial Documents
Pei Chen, Kang Liu, Yubo Chen, Taifeng Wang and Jun Zhao

Language Modelling as a Multi-Task Problem
Lucas Weber, Jaap Jumelet, Elia Bruni and Dieuwke Hupkes

ChainCQG: Flow-Aware Conversational Question Generation
Jing Gu, Mostafa Mirshekari, Zhou Yu and Aaron Sisto

The Interplay of Task Success and Dialogue Quality: An in-depth Evaluation in Task-Oriented Visual Dialogues
Alberto Testoni and Raffaella Bernardi

"Are you kidding me?": Detecting Unpalatable Questions on Reddit
Sunyam Bagga, Andrew Piper and Derek Ruths

Neural-Driven Search-Based Paraphrase Generation
Betty Fabre, Tanguy Urvoy, Jonathan Chevelu and Damien Lolive

No Day Set (continued)

Word Alignment by Fine-tuning Embeddings on Parallel Corpora
Zi-Yi Dou and Graham Neubig

Paraphrases do not explain word analogies
Louis Fournier and Ewan Dunbar

An Empirical Study on the Generalization Power of Neural Representations Learned via Visual Guessing Games
Alessandro Suglia, Yonatan Bisk, Ioannis Konstas, Antonio Vergari, Emanuele Bastianelli, Andrea Vanzo and Oliver Lemon

A Unified Feature Representation for Lexical Connotations
Emily Allaway and Kathleen McKeown

FAST: Financial News and Tweet Based Time Aware Network for Stock Trading
Ramit Sawhney, Arnav Wadhwa, Shivam Agarwal and Rajiv Ratn Shah

Building Representative Corpora from Illiterate Communities: A Reviewof Challenges and Mitigation Strategies for Developing Countries
Stephanie Hirmer, Alycia Leonard, Josephine Tumwesige and Costanza Conforti

Process-Level Representation of Scientific Protocols with Interactive Annotation
Ronen Tamari, Fan Bai, Alan Ritter and Gabriel Stanovsky

Machine Translationese: Effects of Algorithmic Bias on Linguistic Complexity in Machine Translation
Eva Vanmassenhove, Dimitar Shterionov and Matthew Gwilliam

First Align, then Predict: Understanding the Cross-Lingual Ability of Multilingual BERT
Benjamin Muller, Yanai Elazar, Benoît Sagot and Djamé Seddah

Stereotype and Skew: Quantifying Gender Bias in Pre-trained and Fine-tuned Language Models
Daniel de Vassimon Manela, David Errington, Thomas Fisher, Boris van Breugel and Pasquale Minervini

On the evolution of syntactic information encoded by BERT's contextualized representations
Laura Pérez-Mayos, Roberto Carlini, Miguel Ballesteros and Leo Wanner

Identify, Align, and Integrate: Matching Knowledge Graphs to Commonsense Reasoning Tasks
Lisa Bauer and Mohit Bansal

No Day Set (continued)

Calculating the optimal step of arc-eager parsing for non-projective trees
Mark-Jan Nederhof

Subword Pooling Makes a Difference
Judit Ács, Ákos Kádár and Andras Kornai

Content-based Models of Quotation
Ansel MacLaughlin and David Smith

L2C: Describing Visual Differences Needs Semantic Understanding of Individuals
An Yan, Xin Wang, Tsu-Jui Fu and William Yang Wang

VoiSeR: A New Benchmark for Voice-Based Search Refinement
Simone Filice, Giuseppe Castellucci, Marcus Collins, Eugene Agichtein and Oleg Rokhlenko

Event-Driven News Stream Clustering using Entity-Aware Contextual Embeddings
Kailash Karthik Saravanakumar, Miguel Ballesteros, Muthu Kumar Chandrasekaran and Kathleen McKeown

Adversarial Learning of Poisson Factorisation Model for Gauging Brand Sentiment in User Reviews
Runcong Zhao, Lin Gui, Gabriele Pergola and Yulan He

Lexical Normalization for Code-switched Data and its Effect on POS Tagging
Rob van der Goot and Özlem Çetinoğlu

Structural Encoding and Pre-training Matter: Adapting BERT for Table-Based Fact Verification
Rui Dong and David Smith

A Study of Automatic Metrics for the Evaluation of Natural Language Explanations
Miruna-Adriana Clinciu, Arash Eshghi and Helen Hastie

Adversarial Stylometry in the Wild: Transferable Lexical Substitution Attacks on Author Profiling
Chris Emmery, Ákos Kádár and Grzegorz Chrupała

Cross-Cultural Similarity Features for Cross-Lingual Transfer Learning of Pragmatically Motivated Tasks
Jimin Sun, Hwijeen Ahn, Chan Young Park, Yulia Tsvetkov and David R. Mortensen

No Day Set (continued)

PHASE: Learning Emotional Phase-aware Representations for Suicide Ideation Detection on Social Media
Ramit Sawhney, Harshit Joshi, Lucie Flek and Rajiv Ratn Shah

Exploiting Definitions for Frame Identification
Tianyu Jiang and Ellen Riloff

ADePT: Auto-encoder based Differentially Private Text Transformation
Satyapriya Krishna, Rahul Gupta and Christophe Dupuy

Conceptual Grounding Constraints for Truly Robust Biomedical Name Representations
Pieter Fivez, Simon Suster and Walter Daelemans

Adaptive Mixed Component LDA for Low Resource Topic Modeling
Suzanna Sia and Kevin Duh

Evaluating Neural Model Robustness for Machine Comprehension
Winston Wu, Dustin Arendt and Svitlana Volkova

Hidden Biases in Unreliable News Detection Datasets
Xiang Zhou, Heba Elfardy, Christos Christodoulopoulos, Thomas Butler and Mohit Bansal

Annealing Knowledge Distillation
Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma and Ali Ghodsi

Unsupervised Extractive Summarization using Pointwise Mutual Information
Vishakh Padmakumar and He He

Context-aware Neural Machine Translation with Mini-batch Embedding
Makoto Morishita, Jun Suzuki, Tomoharu Iwata and Masaaki Nagata

Deep Subjecthood: Higher-Order Grammatical Features in Multilingual BERT
Isabel Papadimitriou, Ethan A. Chi, Richard Futrell and Kyle Mahowald

Streaming Models for Joint Speech Recognition and Translation
Orion Weller, Matthias Sperber, Christian Gollan and Joris Kluivers

No Day Set (continued)

DOCENT: Learning Self-Supervised Entity Representations from Large Document Collections
Yury Zemlyanskiy, Sudeep Gandhe, Ruining He, Bhargav Kanagal, Anirudh Ravula, Juraj Gottweis, Fei Sha and Ilya Eckstein

Scientific Discourse Tagging for Evidence Extraction
Xiangci Li, Gully Burns and Nanyun Peng

Incremental Beam Manipulation for Natural Language Generation
James Hargreaves, Andreas Vlachos and Guy Emerson

StructSum: Summarization via Structured Representations
Vidhisha Balachandran, Artidoro Pagnoni, Jay Yoon Lee, Dheeraj Rajagopal, Jaime Carbonell and Yulia Tsvetkov

Project-then-Transfer: Effective Two-stage Cross-lingual Transfer for Semantic Dependency Parsing
Hiroaki Ozaki, Gaku Morio, Terufumi Morishita and Toshinori Miyoshi

LSOIE: A Large-Scale Dataset for Supervised Open Information Extraction
Jacob Solawetz and Stefan Larson

Changing the Mind of Transformers for Topically-Controllable Language Generation
Haw-Shiuan Chang, Jiaming Yuan, Mohit Iyyer and Andrew McCallum

Unsupervised Abstractive Summarization of Bengali Text Documents
Radia Rayan Chowdhury, Mir Tafseer Nayeem, Tahsin Tasnim Mim, Md. Saifur Rahman Chowdhury and Taufiqul Jannat

From Toxicity in Online Comments to Incivility in American News: Proceed with Caution
Anushree Hede, Oshin Agarwal, Linda Lu, Diana C. Mutz and Ani Nenkova

On the Computational Modelling of Michif Verbal Morphology
Fineen Davis, Eddie Antonio Santos and Heather Souter

A Few Topical Tweets are Enough for Effective User Stance Detection
Younes Samih and Kareem Darwish

Do Syntax Trees Help Pre-trained Transformers Extract Information?
Devendra Sachan, Yuhao Zhang, Peng Qi and William L. Hamilton

No Day Set (continued)

Informative and Controllable Opinion Summarization
Reinald Kim Amplayo and Mirella Lapata

Coloring the Black Box: What Synesthesia Tells Us about Character Embeddings
Katharina Kann and Mauro M. Monsalve-Mercado

How Good (really) are Grammatical Error Correction Systems?
Alla Rozovskaya and Dan Roth

BERTective: Language Models and Contextual Information for Deception Detection
Tommaso Fornaciari, Federico Bianchi, Massimo Poesio and Dirk Hovy

Learning Coupled Policies for Simultaneous Machine Translation using Imitation Learning
Philip Arthur, Trevor Cohn and Gholamreza Haffari

Complementary Evidence Identification in Open-Domain Question Answering
Xiangyang Mou, Mo Yu, Shiyu Chang, Yufei Feng, Li Zhang and Hui Su

Entity-level Factual Consistency of Abstractive Text Summarization
Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown and Bing Xiang

On Hallucination and Predictive Uncertainty in Conditional Language Generation
Yijun Xiao and William Yang Wang

Fine-Grained Event Trigger Detection
Duong Le and Thien Huu Nguyen

Extremely Small BERT Models from Mixed-Vocabulary Training
Sanqiang Zhao, Raghav Gupta, Yang Song and Denny Zhou

Diverse Adversaries for Mitigating Bias in Training
Xudong Han, Timothy Baldwin and Trevor Cohn

'Just because you are right, doesn't mean I am wrong': Overcoming a bottleneck in development and evaluation of Open-Ended VQA tasks
Man Luo, Shailaja Keyur Sampat, Riley Tallman, Yankai Zeng, Manuha Vancha, Akarshan Sajja and Chitta Baral

No Day Set (continued)

Better Neural Machine Translation by Extracting Linguistic Information from BERT
Hassan S. Shavarani and Anoop Sarkar

CLiMP: A Benchmark for Chinese Language Model Evaluation
Beilei Xiang, Changbing Yang, Yu Li, Alex Warstadt and Katharina Kann

Measuring and Improving Faithfulness of Attention in Neural Machine Translation
Pooya Moradi, Nishant Kambhatla and Anoop Sarkar

Progressively Pretrained Dense Corpus Index for Open-Domain Question Answering
Wenhan Xiong, Hong Wang and William Yang Wang

Exploring the Limits of Few-Shot Link Prediction in Knowledge Graphs
Dora Jambor, Komal Teru, Joelle Pineau and William L. Hamilton

ProFormer: Towards On-Device LSH Projection Based Transformers
Chinnadhurai Sankar, Sujith Ravi and Zornitsa Kozareva

Joint Learning of Hyperbolic Label Embeddings for Hierarchical Multi-label Classification
Soumya Chatterjee, Ayush Maheshwari, Ganesh Ramakrishnan and Saketha Nath Jagaralpudi

Segmenting Subtitles for Correcting ASR Segmentation Errors
David Wan, Chris Kedzie, Faisal Ladhak, Elsbeth Turcan, Petra Galuscakova, Elena Zotkina, Zhengping Jiang, Peter Bell and Kathleen McKeown

Crisscrossed Captions: Extended Intramodal and Intermodal Semantic Similarity Judgments for MS-COCO
Zarana Parekh, Jason Baldridge, Daniel Cer, Austin Waters and Yinfei Yang

On-Device Text Representations Robust To Misspellings via Projections
Chinnadhurai Sankar, Sujith Ravi and Zornitsa Kozareva

ENPAR:Enhancing Entity and Entity Pair Representations for Joint Entity Relation Extraction
Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li and Junchi Yan

Text Augmentation in a Multi-Task View
Jason Wei, Chengyu Huang, Shiqi Xu and Soroush Vosoughi

No Day Set (continued)

Representations for Question Answering from Documents with Tables and Text
Vicky Zayats, Kristina Toutanova and Mari Ostendorf

PPT: Parsimonious Parser Transfer for Unsupervised Cross-Lingual Adaptation
Kemal Kurniawan, Lea Frermann, Philip Schulz and Trevor Cohn

Modelling Context Emotions using Multi-task Learning for Emotion Controlled Dialog Generation
Deeksha Varshney, Asif Ekbal and Pushpak Bhattacharyya

Gender and Racial Fairness in Depression Research using Social Media
Carlos Aguirre, Keith Harrigian and Mark Dredze

MTOP: A Comprehensive Multilingual Task-Oriented Semantic Parsing Benchmark
Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta and Yashar Mehdad

Adapting Event Extractors to Medical Data: Bridging the Covariate Shift
Aakanksha Naik, Jill Fain Lehman and Carolyn Rose

NoiseQA: Challenge Set Evaluation for User-Centric Question Answering
Abhilasha Ravichander, Siddharth Dalmia, Maria Ryskina, Florian Metze, Eduard Hovy and Alan W Black

Co-evolution of language and agents in referential games
Gautier Dagan, Dieuwke Hupkes and Elia Bruni

Modeling Context in Answer Sentence Selection Systems on a Latency Budget
Rujun Han, Luca Soldaini and Alessandro Moschitti

Syntax-BERT: Improving Pre-trained Transformers with Syntax Trees
Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu and Yunhai Tong

DISK-CSV: Distilling Interpretable Semantic Knowledge with a Class Semantic Vector
Housam Khalifa Bashier, Mi-Young Kim and Randy Goebel

Attention Can Reflect Syntactic Structure (If You Let It)
Vinit Ravishankar, Artur Kulmizev, Mostafa Abdou, Anders Søgaard and Joakim Nivre

Extractive Summarization Considering Discourse and Coreference Relations based on Heterogeneous Graph
Yin Jou Huang and Sadao Kurohashi

CDA: a Cost Efficient Content-based Multilingual Web Document Aligner
Thuy Vu and Alessandro Moschitti

Metric-Type Identification for Multi-Level Header Numerical Tables in Scientific Papers
Lya Hulliyyatus Suadaa, Hidetaka Kamigaito, Manabu Okumura and Hiroya Takamura

EmpathBERT: A BERT-based Framework for Demographic-aware Empathy Prediction
Bhanu Prakash Reddy Guda, Aparna Garimella and Niyati Chhaya

Are Neural Networks Extracting Linguistic Properties or Memorizing Training Data? An Observation with a Multilingual Probe for Predicting Tense
Bingzhi Li and Guillaume Wisniewski

Is Supervised Syntactic Parsing Beneficial for Language Understanding Tasks? An Empirical Investigation
Goran Glavaš and Ivan Vulić

Facilitating Terminology Translation with Target Lemma Annotations
Toms Bergmanis and Mārcis Pinnis

Enhancing Sequence-to-Sequence Neural Lemmatization with External Resources
Kirill Milintsevich and Kairit Sirts

Summarising Historical Text in Modern Languages
Xutan Peng, Yi Zheng, Chenghua Lin and Advaith Siddharthan

Challenges in Automated Debiasing for Toxic Language Detection
Xuhui Zhou, Maarten Sap, Swabha Swayamdipta, Yejin Choi and Noah Smith

Adaptive Fusion Techniques for Multimodal Data
Gaurav Sahu and Olga Vechtomova

Detecting Scenes in Fiction: A new Segmentation Task
Albin Zehe, Leonard Konle, Lea Katharina Dümpelmann, Evelyn Gius, Andreas Hotho, Fotis Jannidis, Lucas Kaufmann, Markus Krug, Frank Puppe, Nils Reiter, Annekea Schreiber and Nathalie Wiedmer

No Day Set (continued)

LESA: Linguistic Encapsulation and Semantic Amalgamation Based Generalised Claim Detection from Online Content
Shreya Gupta, Parantak Singh, Megha Sundriyal, Md. Shad Akhtar and Tanmoy Chakraborty

Interpretability for Morphological Inflection: from Character-level Predictions to Subword-level Rules
Tatyana Ruzsics, Olga Sozinova, Ximena Gutierrez-Vasques and Tanja Samardzic

Expanding, Retrieving and Infilling: Diversifying Cross-Domain Question Generation with Flexible Templates
Xiaojing Yu and Anxiao Jiang

Handling Out-Of-Vocabulary Problem in Hangeul Word Embeddings
Ohjoon Kwon, Dohyun Kim, Soo-Ryeon Lee, Junyoung Choi and SangKeun Lee

Exploiting Multimodal Reinforcement Learning for Simultaneous Machine Translation
Julia Ive, Andy Mingren Li, Yishu Miao, Ozan Caglayan, Pranava Madhyastha and Lucia Specia

STAR: Cross-modal [STA]tement [R]epresentation for selecting relevant mathematical premises
Deborah Ferreira and André Freitas

Do Multi-Hop Question Answering Systems Know How to Answer the Single-Hop Sub-Questions?
Yixuan Tang, Hwee Tou Ng and Anthony Tung

Multilingual LAMA: Investigating Knowledge in Multilingual Pretrained Language Models
Nora Kassner, Philipp Dufter and Hinrich Schütze

Variational Weakly Supervised Sentiment Analysis with Posterior Regularization
Ziqian Zeng and Yangqiu Song

Framing Word Sense Disambiguation as a Multi-Label Problem for Model-Agnostic Knowledge Integration
Simone Conia and Roberto Navigli

Graph-based Fake News Detection using a Summarization Technique
Gihwan Kim and Youngjoong Ko

Cognition-aware Cognate Detection
Diptesh Kanojia, Prashant Sharma, Sayali Ghodekar, Pushpak Bhattacharyya, Gholamreza Haffari and malhar kulkarni

No Day Set (continued)

A Simple Three-Step Approach for the Automatic Detection of Exaggerated Statements in Health Science News
Jasabanta Patro and Sabyasachee Baruah

Modeling Coreference Relations in Visual Dialog
Mingxiao Li and Marie-Francine Moens

Increasing Robustness to Spurious Correlations using Forgettable Examples
Yadollah Yaghoobzadeh, Soroush Mehri, Remi Tachet des Combes, T. J. Hazen and Alessandro Sordoni

On Robustness of Neural Semantic Parsers
shuo huang, Zhuang Li, Lizhen Qu and Lei Pan

Benchmarking a transformer-FREE model for ad-hoc retrieval
Tiago Almeida and Sérgio Matos

Reanalyzing the Most Probable Sentence Problem: A Case Study in Explicating the Role of Entropy in Algorithmic Complexity
Eric Corlett and Gerald Penn

Probing the Probing Paradigm: Does Probing Accuracy Entail Task Relevance?
Abhilasha Ravichander, Yonatan Belinkov and Eduard Hovy

One-class Text Classification with Multi-modal Deep Support Vector Data Description
Chenlong Hu, Yukun Feng, Hidetaka Kamigaito, Hiroya Takamura and Manabu Okumura

Unsupervised Word Polysemy Quantification with Multiresolution Grids of Contextual Embeddings
Christos Xypolopoulos, Antoine Tixier and Michalis Vazirgiannis

Mega-COV: A Billion-Scale Dataset of 100+ Languages for COVID-19
Muhammad Abdul-Mageed, AbdelRahim Elmadany, El Moatez Billah Nagoudi, Dinesh Pabbi, Kunal Verma and Rannie Lin

Disfluency Correction using Unsupervised and Semi-supervised Learning
Nikhil Saini, Drumil Trivedi, Shreya Khare, Tejas Dhamecha, Preethi Jyothi, Samarth Bharadwaj and Pushpak Bhattacharyya

Complex Question Answering on knowledge graphs using machine translation and multi-task learning
Saurabh Srivastava, Mayur Patidar, Sudip Chowdhury, Puneet Agarwal, Indrajit Bhattacharya and Gautam Shroff

Recipes for Adapting Pre-trained Monolingual and Multilingual Models to Machine Translation
Asa Cooper Stickland, Xian Li and Marjan Ghazvininejad

From characters to words: the turning point of BPE merges
Ximena Gutierrez-Vasques, Christian Bentz, Olga Sozinova and Tanja Samardzic

A Large-scale Evaluation of Neural Machine Transliteration for Indic Languages
Anoop Kunchukuttan, Siddharth Jain and Rahul Kejriwal

Communicative-Function-Based Sentence Classification for Construction of an Academic Formulaic Expression Database
Kenichi Iwatsuki and Akiko Aizawa

Regulatory Compliance through Doc2Doc Information Retrieval: A case study in EU/UK legislation where text similarity has limitations
Ilias Chalkidis, Manos Fergadiotis, Nikolaos Manginas, Eva Katakalou and Prodromos Malakasiotis

The Chinese Remainder Theorem for Compact, Task-Precise, Efficient and Secure Word Embeddings
Patricia Thaine and Gerald Penn

Don't Change Me! User-Controllable Selective Paraphrase Generation
Mohan Zhang, Luchen Tan, Zihang Fu, Kun Xiong, Jimmy Lin, Ming Li and Zhengkai Tu

Rethinking Coherence Modeling: Synthetic vs. Downstream Tasks
Tasnim Mohiuddin, Prathyusha Jwalapuram, Xiang Lin and Shafiq Joty

From the Stage to the Audience: Propaganda on Reddit
Oana Balalau and Roxana Horincar

Probing for idiomaticity in vector space models
Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart and Aline Villavicencio

Is the Understanding of Explicit Discourse Relations Required in Machine Reading Comprehension?
Yulong Wu, Viktor Schlegel and Riza Batista-Navarro

Why Is MBTI Personality Detection from Texts a Difficult Task?
Sanja Stajner and Seren Yenikent

No Day Set (continued)

Enconter: Entity Constrained Progressive Sequence Generation via Insertion-based Transformer
Lee Hsun Hsieh, Yang-Yin Lee and Ee-Peng Lim

Meta-Learning for Effective Multi-task and Multilingual Modelling
Ishan Tarunesh, Sushil Khyalia, vishwajeet kumar, Ganesh Ramakrishnan and Preethi Jyothi

"Killing Me" Is Not a Spoiler: Spoiler Detection Model using Graph Neural Networks with Dependency Relation-Aware Attention Mechanism
Buru Chang, Inggeol Lee, Hyunjae Kim and Jaewoo Kang

BERTese: Learning to Speak to BERT
Adi Haviv, Jonathan Berant and Amir Globerson

Lifelong Knowledge-Enriched Social Event Representation Learning
Prashanth Vijayaraghavan and Deb Roy

GLaRA: Graph-based Labeling Rule Augmentation for Weakly Supervised Named Entity Recognition
Xinyan Zhao, Haibo Ding and Zhe Feng

An End-to-end Model for Entity-level Relation Extraction using Multi-instance Learning
Markus Eberts and Adrian Ulges

WER-BERT: Automatic WER Estimation with BERT in a Balanced Ordinal Classification Paradigm
Akshay Krishna Sheshadri, Anvesh Rao Vijjini and Sukhdeep Kharbanda

Two Training Strategies for Improving Relation Extraction over Universal Graph
Qin Dai, Naoya Inoue, Ryo Takahashi and Kentaro Inui

Adaptation of Back-translation to Automatic Post-Editing for Synthetic Data Generation
WonKee Lee, Baikjin Jung, Jaehun Shin and Jong-Hyeok Lee

Removing Word-Level Spurious Alignment between Images and Pseudo-Captions in Unsupervised Image Captioning
Ukyo Honda, Yoshitaka Ushiku, Atsushi Hashimoto, Taro Watanabe and Yuji Matsumoto

Towards More Fine-grained and Reliable NLP Performance Prediction
Zihuiwen Ye, Pengfei Liu, Jinlan Fu and Graham Neubig

No Day Set (continued)

Metrical Tagging in the Wild: Building and Annotating Poetry Corpora with Rhythmic Features
Thomas Haider

Enhancing Aspect-level Sentiment Analysis with Word Dependencies
Yuanhe Tian, Guimin Chen and Yan Song

Unsupervised Sentence-embeddings by Manifold Approximation and Projection

Subhradeep Kayal
Prosus N.V.
Amsterdam, Netherlands.
deep.kayal@pm.me

Abstract

The concept of unsupervised *universal sentence encoders* has gained traction recently, wherein pre-trained models generate effective task-agnostic fixed-dimensional representations for phrases, sentences and paragraphs. Such methods are of varying complexity, from simple weighted-averages of word vectors to complex language-models based on bidirectional transformers. In this work we propose a novel technique to generate sentence-embeddings in an unsupervised fashion by projecting the sentences onto a fixed-dimensional manifold with the objective of preserving local neighbourhoods in the original space. To delineate such neighbourhoods we experiment with several set-distance metrics, including the recently proposed *Word Mover's distance*, while the fixed-dimensional projection is achieved by employing a scalable and efficient manifold approximation method rooted in topological data analysis. We test our approach, which we term *EMAP* or *Embeddings by Manifold Approximation and Projection*, on six publicly available text-classification datasets of varying size and complexity. Empirical results show that our method consistently performs similar to or better than several alternative state-of-the-art approaches.

1 Introduction

1.1 On sentence-embeddings

Dense vector representation of words, or *word-embeddings*, form the backbone of most modern NLP applications and can be constructed using context-free (Bengio et al., 2003; Mikolov et al., 2013; Pennington et al., 2014) or contextualized methods (Peters et al., 2018; Devlin et al., 2019).

Given that practical systems often benefit from having representations for sentences and documents, in addition to word-embeddings (Palangi et al., 2016; Yan et al., 2016), a simple trick is to use the weighted average over some or all of the embeddings of words in a sentence or document. Although *sentence-embeddings* constructed this way often lose information because of the disregard for word-order during averaging, they have been found to be surprisingly performant (Aldarmaki and Diab, 2018).

More sophisticated methods focus on jointly learning the embeddings of sentences and words using models similar to *Word2Vec* (Le and Mikolov, 2014; Chen, 2017), using encoder-decoder approaches that reconstruct the surrounding sentences of an encoded passage (Kiros et al., 2015), or training bi-directional LSTM models on large external datasets (Conneau et al., 2017). Meaningful sentence-embeddings have also been constructed by fine-tuning pre-trained bidirectional transformers (Devlin et al., 2019) using a Siamese architecture (Reimers and Gurevych, 2019).

In parallel to the approaches mentioned above, a stream of methods have emerged recently which exploit the inherent geometric properties of the structure of sentences, by treating them as sets or sequences of word-embeddings. For example, Arora et al. (2017) propose the construction of sentence-embeddings based on weighted word-embedding averages with the removal of the dominant singular vector, while Rücklé et al. (2018) produce sentence-embeddings by concatenating several power-means of word-embeddings corresponding to a sentence. Very recently, spectral decomposition techniques were used to create sentence-embeddings, which produced state-of-the-art results when used in concatenation with averaging (Kayal and Tsatsaronis, 2019; Almarwani et al., 2019).

Our work is most related to that of Wu et al. (2018) who use Random Features (Rahimi and Recht, 2008) to learn document embeddings which preserve the properties of an explicitly-defined kernel based on the Word Mover's Distance (Kusner

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1–11
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

et al., 2015). Where Wu et al. predefine the nature of the kernel, our proposed approach can learn the similarity-preserving manifold for a given set-distance metric, offering increased flexibility.

1.2 Motivation and contributions

A simple way to form sentence-embeddings is to compute the dimension-wise arithmetic mean of the embeddings of the words in a particular sentence. Even though this approach incurs information loss by disregarding the fact that sentences are sequences (or, at the very least, sets) of word vectors, it works well in practice. This already provides an indication that there is more information in the sentences to be exploited.

Kusner et al. (2015) aim to use more of the information available in a sentence by representing sentences as a weighted point cloud of embedded words. Rooted in transportation theory, their *Word Mover's distance (WMD)* is the minimum amount of distance that the embedded words of a sentence need to travel to reach the embedded words of another sentence. The approach achieves state-of-the-art results for sentence classification when combined with a k-NN classifier (Cover and Hart, 1967). Since their work, other distance metrics have been suggested (Singh et al., 2019; Wang et al., 2019), also motivated by how transportation problems are solved.

Considering that sentences are sets of word vectors, a large variety of methods exist in literature that can be used to calculate the distance between two sets, in addition to the ones based on transport theory. Thus, as a *first contribution*, we compare alternative metrics to measure distances between sentences. The metrics we suggest, namely the *Hausdorff distance* and the *Energy distance*, are intuitive to explain and reasonably fast to calculate. The choice of these particular distances are motivated by their differing origins and their general usefulness in the respective application domains.

Once calculated, these distances can be used in conjunction with k-nearest neighbours for classification tasks, and k-means for clustering tasks. However, these learning algorithms are rather simplistic and the state-of-the-art machine learning algorithms require a fixed-length feature representation as input to them. Moreover, having fixed-length representations for sentences (*sentence-embeddings*) also provides a large degree of flexibility for downstream tasks, as compared to having only relative distances between them. With this as motivation, the *second contribution* of this work is to produce sentence-embeddings that approximately preserve the topological properties of the original sentence space. We propose to do so using an efficient scalable manifold-learning algorithm termed *UMAP* (McInnes et al., 2018) from topological data analysis. Empirical results show that this process yields sentence-embeddings that deliver near state-of-the-art classification performance with a simple classifier.

2 Methodology

2.1 Calculating distances

In this work, we experiment with three different distance measures to determine the distance between sentences. The first measure (Energy distance) is motivated by a useful linkage criterion from hierarchical clustering (Rokach and Maimon, 2005), while the second one (Hausdorff distance) is an important metric from algebraic topology that has been successfully used in document indexing (Tsatsaronis et al., 2012). The final metric (Word Mover's distance) is a recent extension of an existing distance measure between distributions, that is particularly suited for use with word-embeddings (Kusner et al., 2015).

Prior to defining the distances that have been used in this work, we first proceed to outline the notations that we will be using to describe them.

2.1.1 Notations

Let $\mathscr{W} \in \mathbb{R}^{N \times d}$ denote a word-embedding matrix, such that the vocabulary corresponding to it consists of N words, and each word in it, $w_i \in \mathbb{R}^d$, is d-dimensional. This word-embedding matrix and its constituent words may come from pre-trained representations such as Word2Vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014), in which case $d = 300$.

Let $\mathscr{S}$ be a set of sentences and s, s' be two sentences from this set. Each such sentence can be viewed as a set of word-embeddings, $\{w\} \in s$. Additionally, let the length of a sentence, s, be denoted as $|s|$, and the cardinality of the set, $\mathscr{S}$, be denoted by $|\mathscr{S}|$.

Let $e(w_i, w_j)$ denote the distance between two word-embeddings, w_i, w_j. In the context of this paper, this distance is Euclidean:

$$e(w_i, w_j) = \|w_i - w_j\|_2 \qquad (1)$$

Finally, $D(s, s')$ denotes the distance between two sentences.

2.1.2 Energy distance

Energy distance is a statistical distance between probability distributions, based on the inter and intra-distribution variance, that satisfies all the criteria of being a metric (Székely and Rizzo, 2013).

Using the notations defined earlier, we write it as:

$$\begin{aligned} D(s, s') = &\frac{2}{|s||s'|} \sum_{w_i \in s} \sum_{w_j \in s'} e(w_i, w_j) \\ &- \frac{1}{|s|^2} \sum_{w_i \in s} \sum_{w_j \in s} e(w_i, w_j) \\ &- \frac{1}{|s'|^2} \sum_{w_i \in s'} \sum_{w_j \in s'} e(w_i, w_j) \end{aligned} \tag{2}$$

The original conception of the energy distance was inspired by gravitational potential energy of celestial objects. Looking closely at Equation 2, it can be quickly observed that it has two parts: the first term resembles the attraction or repulsion between two objects (or in our case, sentences), while the second and the third term indicate the self-coherence of the respective objects. As shown by Székely and Rizzo (2013), energy distance is scale equivariant, which would make it sensitive to contextual changes in sentences, and therefore make it useful in NLP applications.

2.1.3 Hausdorff distance

Given two subsets of a metric space, the *Hausdorff distance* is the maximum distance of the points in one subset to the nearest point in the other. A significant work has gone into making it fast to calculate (Atallah, 1983) so that it can be applied to real-world problems, such as shape-matching in computer vision (Dubuisson and Jain, 1994).

To calculate it, the distance between each point from one set and the closest point from the other set is determined first. Then, the Hausdorff distance is calculated as the maximal point-wise distance. Considering sentences $\{s, s'\}$ as subsets of word-embedding space, $\mathbb{R}^{d \times N}$, the *directed* Hausdorff distance can be given as:

$$h(s, s') = \max_{w_i \in s} \min_{w_j \in s'} e(w_i, w_j) \tag{3}$$

such that the symmetric Hausdorff distance is:

$$D(s, s') = \max\{h(s, s'), h(s', s)\} \tag{4}$$

2.1.4 Word Mover's distance

In addition to the representation of a sentence as a set of word-embeddings, a sentence s can also be represented as a N-dimensional normalized term-frequency vector, where n_i^s is the number of times word w_i occurs in sentence s normalized by the total number of words in s:

$$n_i^s = \frac{c_i^s}{\sum_{k=1}^{k=N} c_k^s} \tag{5}$$

where, c_i^s is the number of times word w_i appears in sentence s.

The goal of the *Word Mover's distance (WMD)* (Kusner et al., 2015) is to construct a sentence similarity metric based on the distances between the individual words within each sentence, given by Equation 1. In order to calculate the distance between two sentences, WMD introduces a transport matrix, $T \in \mathbb{R}^{N \times N}$, such that each element in it, T_{ij}, denotes how much of n_i^s should be transported to $n_j^{s'}$. Then, the WMD between two sentences is given as the solution of the following minimization problem:

$$\begin{gathered} D(s, s') = \min_{T \geq 0} \sum_{i,j=1}^{N} T_{ij} e(i, j) \\ \text{subject to, } \sum_{j=1}^{N} T_{ij} = n_i^s \text{ and } \sum_{i=1}^{N} T_{ij} = n_j^{s'} \end{gathered} \tag{6}$$

Thus, WMD between two sentences is defined as the minimum distance required to transport the words from one sentence to another.

2.2 Generating neighbourhood-preserving embeddings via non-linear manifold-learning

In this work, we propose to construct sentence-embeddings which preserve the neighbourhood around sentences delineated by the relative distances between them. We posit that preserving the local neighbourhoods will serve as a proxy for preserving the original topological properties.

In order to learn a topology-preserving fixed-dimensional manifold, we seek inspiration from methods in non-linear dimensionality-reduction (Lee and Verleysen, 2007) and topological data analysis literature (Carlsson, 2009). When broadly categorized, these techniques consist of methods, such as *Locally Linear Embedding* (Roweis and Saul, 2000), that preserve local distances between points, or those like *Stochastic Neighbour Embedding* (Hinton and Roweis, 2003; van der Maaten and Hinton, 2008) that preserve the conditional probabilities of points being neighbours. However, existing manifold-learning algorithms suffer from two shortcomings: they are computationally expensive and are often restricted in the number of output dimensions. In our work we use a method termed *Uniform Manifold Approximation and Projection (UMAP)* (McInnes et al., 2018), which is scalable and has no computational restrictions on the output embedding dimension.

The building block of UMAP is a particular type of a simplicial complex, known as the Vietoris-Rips complex. Recalling that a *k-simplex* is a k-dimensional polytope which is the convex hull of its $k + 1$ vertices, and a simplicial complex is a set of simplices of various orders, the Vietoris-Rips simplicial complex is a collection of 0 and 1-simplices. In essence, this is a means to building a simple neighbourhood graph by connecting the original data points.

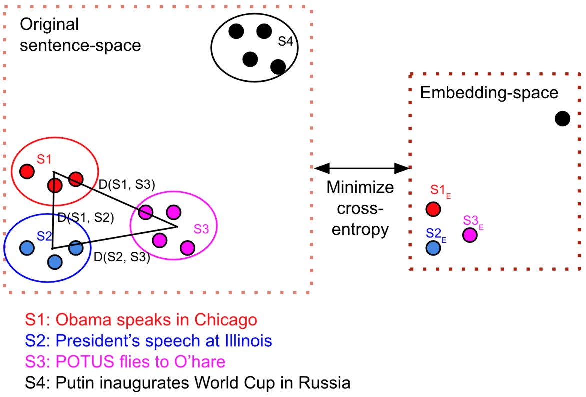


Figure 1: Figure showing a simple example of the embedding algorithm. On the left is the original sentence-space, approximated by the nearest neighbours graph formed by the Vietoris-Rips complex. Instead of points and edges, our simplicial complex has sets of points and edges between them, formed by one of the distance metrics mentioned in Section 2.1. In this example, four sentences, denoted by $S1$ through $S4$, form two simplices, with $S4$ being a *0*-simplex. The sentences are denoted by colored ellipses, while the high-dimensional embedding of each word in a sentence is depicted by a point having the same color as the parent sentence ellipse. The UMAP algorithm is then employed to find a similarity-preserving Euclidean embedding-space, shown on the right, by minimizing the cross-entropy between the two representations.

A key difference, in this work, to the original formulation is that an individual data sample (i.e., the vertex of a simplex) is not a d-dimensional point but a set of d-dimensional words that make up a sentence. By using any of the distance metrics defined in Section 2.1, it is possible to construct the simplicial complex that UMAP needs in order to build the topological representation of the original sentence space. An illustration can be found in Figure 1.

As per the formulation laid out for UMAP, the similarity between sentences s' and s is defined as:

$$v_{s'|s} = \exp \frac{-(D(s,s') - \rho_s)}{\sigma_s} \tag{7}$$

where σ_s is a normalisation factor selected based on an empirical heuristic (See Algorithm 3 in the work of McInnes et al. 2018), $D(s,s')$ is the distance between two sentences as outlined by Equation 2, 4 or 6, and ρ_s is the distance of s from its nearest neighbour. It is worth mentioning that for scalability, $v_{s'|s}$ is calculated only for predefined set of approximate nearest neighbours, which is a user-defined input parameter to the UMAP algorithm, using the efficient *nearest-neighbour descent* algorithm (Dong et al., 2011).

The similarity depicted in Equation 7 is asymmetric, and symmetrization is carried out by a fuzzy set union using the probabilistic t-conorm:

$$v_{ss'} = (v_{s'|s} + v_{s|s'}) - v_{s'|s} v_{s|s'} \tag{8}$$

As UMAP builds a Vietoris-Rips complex governed by Equation 7, it can take advantage of the nerve theorem (Borsuk, 1948), which makes this construction a homotope of the original topological space. In our case, this implies that we can build a simple nearest neighbours graph from a given corpus of sentences, which has certain guarantees of approximating the original topological space, as defined by the aforementioned distance metrics.

The next step is to define a similar nearest neighbours graph in a fixed low-dimensional Euclidean space. Let $s_E, s'_E \in \mathbb{R}^{d_E}$ be the corresponding d_E-dimensional sentence-embeddings. Then the low dimensional similarities are given by:

$$w_{ss'} = (1 + a||s_E - s'_E||_2{}^{b}))^{-1} \tag{9}$$

where, $||s_E - s'_E||$ is the Euclidean distance between the d_E-dimensional embeddings, and setting a, b are input-parameters, set to 1.929 and 0.791, respectively, as per the original implementation.

Algorithm 1: Constructing sentence-*E*mbeddings by *M*anifold *A*pproximation and *P*rojection: *EMAP*

```
Data: A pre-trained word-embeddings matrix, $\mathscr{W}$; a set of sentences, $\mathscr{S}$;
      desired dimension of the generated sentence-embeddings, $d_E$
Result: A set of sentence-embeddings, $\{s_E\} \in \mathscr{S}_E$
Calculate the distance matrix for the entire set of sentences, such that the distance
    between any two sentences is given by Equation 2, 4 or 6;
Using this distance matrix, calculate the nearest neighbour graph between all input
    sentences, given by Equations 7 and 8;
Calculate the initial guess for the low dimensional embeddings, $\mathscr{S}_E \in \mathbb{R}^{|\mathscr{S}| \times D_E}$,
    using the graph laplacian of the original nearest neighbour graph;
Until convergence, minimize the cross-entropy between the two representations
    (Equation 10) using stochastic gradient descent;
Return the set of $d_E$-dimensional sentence-embeddings, $\mathscr{S}_E$;
```

The final step of the process is to optimize the low dimensional representation to have as close a fuzzy topological representation as possible to the original space. UMAP proceeds to do so by minimizing the cross-entropy between the two representations:

$$C = \sum_{s \neq s'} v_{ss'} \log \frac{v_{ss'}}{w_{ss'}} + (1 - v_{ss'}) \log \frac{1 - v_{ss'}}{1 - w_{ss'}} \tag{10}$$

usually done via stochastic gradient descent.

A summary of the proposed process used to produce sentence-embeddings is provided in Algorithm 1, and pictorially presented in Figure 1.

3 Datasets and resources

3.1 Datasets

Six public datasets[1] have been used to empirically validate the method proposed in this paper. These datasets are of varying sizes, tasks and complexities, and have been used widely in existing literature, thereby making comparisons and reporting possible. Information about the datasets can be found in Table 1.

3.2 Resources

Pre-trained word-embedding corpus: We use the pre-trained set of word-embeddings provided by Mikolov et al (2013)[2].

Software implementations: We use a variety of software packages and custom-written programs perform our experiments, the starting point being the calculation of sentence-wise distances. We calculate the Hausdorff distance using a directed implementation provided in the *Scipy* python library[3], whereas the energy distance is calculated using *dcor*[4]. Lastly, the word mover's distance is calculated using implementation provided by Kusner et al. (2015)[5]. In order to produce the symmetric distance matrix for a dataset, we employ custom parallel implementation which distributes the calculations over all available logical cores in a machine.

To calculate the sentence-embeddings, the implementation of UMAP provided by McInnes et al (2018) is used[6]. Finally, the classification is done via linear kernel support vector machines from the *scikit-learn* library (Pedregosa et al., 2011)[7].

All of the code and datasets have been packaged and released[8] to rerun all of the experiments.

Compute infrastructure: All experiments were run on a *m4.2xlarge* machine on AWS-EC2[9], which has 8 virtual CPUs and 32GB of RAM.

4 Experiments

4.1 Competing methods

In order to check the usefulness of our proposed approach, we benchmark its performance in two different ways. The first, and most obvious, approach is to consider the performance of the k-NN

---

[1] `https://drive.google.com/open?id=1sGgAo2SBoYKhQQK_kilUp8KSToCI55jl`

[2] `https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit`

[3] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.directed_hausdorff.html`

[4] `https://dcor.readthedocs.io/en/latest/functions/dcor.energy_distance.html#dcor.energy_distance`

[5] `https://github.com/mkusner/wmd`

[6] `https://umap-learn.readthedocs.io/en/latest/api.html`

[7] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`

[8] `https://github.com/DeepK/distance-embed`

[9] `https://aws.amazon.com/ec2/`

Dataset	#classes	#train docs	#test docs	#avg tokens	Data-details
amazon	4	5600	2400	70	Reviews labeled by product
bbcsport	5	517	220	192	Articles labeld by sport
classic	4	4965	2128	62	Manuscripts labeled by publisher
ohsumed	10	3999	5153	104	Medical abstracts categorized by subject headings
reuters8	8	5485	2189	69	News article categorization
twitter	3	2176	932	8	Tweet sentiment analysis

Table 1: **Dataset information**: Metadata describing the datasets used in our experiments.

classifier as a baseline. This is motivated by the state-of-the-art k-NN based classification accuracy reported by Kusner et al. for the word mover's distance. Thus, our embeddings need to match or surpass the performance of a k-NN based approach, in order to be considered for practical use.

The second approach is to compare the classification accuracies of several state-of-the-art embedding-generation algorithms on our chosen datasets. These are:

dct (Almarwani et al., 2019): embeddings are generated by employing discrete cosine transform on a set of word vectors.

eigensent (Kayal and Tsatsaronis, 2019): sentence representations produced via higher-order dynamic mode decomposition (Le Clainche and Vega, 2017) on a sequence of word vectors.

wmovers (Wu et al., 2018): a competing method which can learn sentence representations from the word mover's distance based on kernel learning, termed in the original work as *word mover's embeddings*.

p-means (Rücklé et al., 2018): produces sentence-embeddings by concatenating several power-means of word-embeddings corresponding to a sentence.

doc2vec (Le and Mikolov, 2014): embeddings produced by jointly learning the representations of sentences, together with words, as a part of the word2vec procedure.

s-bert (Reimers and Gurevych, 2019): embeddings produced by fine-tuning a pre-trained BERT model using a Siamese architecture to classify two sentences as being similar or different.

Note that the results for *wmovers* and *doc2vec* are taken from Table 3 of Wu et al.'s work (2018), while all the other algorithms are explicitly tested.

4.2 Setup

Extensive experiments are performed to provide a holistic overview of our neighbourhood-preserving embedding algorithm, for various sets of input parameters. The steps involved are as follows:

Choose a dataset (one of the six mentioned in Section 3.1). For every word in every sentence in the train and test splits of the dataset, retrieve the corresponding word-embedding from the pre-trained embedding corpus (as stated in Section 3.2).

Calculate symmetric distance matrices corresponding to each of the chosen distance metrics, for all of the sets of word-embeddings from the train and test splits.

Apply the UMAP algorithm on the distance matrices to generate embeddings for all sentences in the train and the test splits.

Calculate embeddings for competing methods for the methods outlined in Section 4.1.

Embeddings are generated for various hyperparameter combinations for *EMAP* as well as all the compared approaches, as listed in Table 2.

Train a classifier on the produced embeddings to perform the dataset-specific task. In this work, we train a simple linear-kernel support vector machine (Cortes and Vapnik, 1995) for every competing method and every dataset tested. The classifier is trained on the train-split of a dataset and evaluated on the test-split. The only parameter tuned for the SVM is the L2 regularization strength, varied between 0.001 and 100. The overall test *accuracy* has been been reported as a measure of performance.

5 Results and Discussion

The results of all our experiments are in compiled in Tables 3 and 4. All statistical tests reported are z-tests, where we compute the right-tailed p-value and call a result significantly different if $p < 0.1$.

Performance of the distance metrics: From Table 3 it can be observed that the word mover's distance consistently performs better than the others experimented with in this paper. WMD calculates the total effort of aligning two sentences, which seems to capture more useful information compared to the hausdorff metric's worst-case effort of alignment. As for the energy distance, it calculates pairwise potentials amongst words within and between sentences, and may suffer if there are

Method	Parameter	Value(s) Tested
EMAP	*n_neighbors*	40
	embedding_dim	50, 100, 300, 1000
	min_dist	1.0, 1.5, 2.0
	spread	1.0, 2.5
	n_iters	1000
	distance	wmd, hausdorff, energy
kNN	*k*	1
	distance	wmd, hausdorff, energy
dct	*components*	1 through 6
eigensent	*components*	1 through 3
	time_lag	1, 2, 3, [1,2], [1,2,3], [1,2,3,4]
pmeans	*powers*	1, [1,2], [1,2,3], [1,2,3,4,5,6]
s-bert	*model*	bert-base-nli-mean-tokens

Table 2: **Hyperparameter values tested**. For *EMAP*, *n_neighbours* refers to the size of local neighborhood used for manifold approximation, *embedding_dim* is the fixed dimensionality of the generated sentence-embeddings, *min_dist* is the minimum distance apart that points are allowed to be in the low dimensional representation, *spread* determines the scale at which embedded points will be spread out, *n_iters* is the number of iterations that the UMAP algorithm is allowed to run, and finally, *distance* is one of the metrics proposed in Section 2.1. For the spectral decomposition based algorithms, *dct* and *eigensent*, *components* represents the number of components to keep in the resulting decomposition, while *time_lag* corresponds to the window-length in the dynamic mode decomposition process. For *pmeans*, *powers* represents the different powers which are used to generate the concatenated embeddings.

Distance	energydist		hausdorffdist		wmddist	
Method	**knn**	**EMAP**	**knn**	**EMAP**	**knn**	**EMAP**
amazon	**0.923**$*$	0.909	0.781	**0.844**$*$	0.918	**0.929**$*$
bbcsport	0.941	**0.942**	0.925	**0.941**	0.972	**0.987**
classic	0.912	**0.921**	0.943	**0.953**$*$	0.961	**0.978**$*$
ohsumed	0.456	**0.505**$*$	0.491	**0.603**$*$	0.551	**0.630**$*$
r8	0.942	**0.962**$*$	**0.863**$*$	0.837	0.951	**0.973**$*$
twitter	0.731	**0.749**	0.736	**0.741**	0.712	**0.722**

Table 3: **Comparison versus kNN**. Results shown here compare the classification accuracies of k-nearest neighbour to our proposed approach for various distance metrics. For every distance, **bold** indicates better accuracy, while $*$ indicates that the winning accuracy was statistically significant with respect to the compared value (,i.e., EMAP vs kNN for a given distance metric). It can be observed that our method almost always outperforms k-nearest neighbour-based classification.

Method	wmd-EMAP	dct	eigensent	wmovers	pmeans	doc2vec	s-bert
amazon	0.929	0.932	0.902$\vee$	**0.943**$\wedge$	*0.938*	0.912$\vee$	0.923
bbcsport	**0.986**	0.972	0.968	*0.982*	0.981	0.979	**0.986**
classic	**0.978**	0.964	0.947$\vee$	*0.971*	0.960	0.965	0.966
ohsumed	*0.630*	0.594$\vee$	0.574$\vee$	**0.645**$\wedge$	0.614$\vee$	0.598$\vee$	0.556$\vee$
r8	**0.973**	0.967	0.958$\vee$	*0.972*	0.969	0.949$\vee$	0.954$\vee$
twitter	*0.722*	0.644$\vee$	0.669$\vee$	**0.745**	0.636$\vee$	0.673$\vee$	0.673$\vee$

Table 4: **Comparison versus competing methods**. We compare *EMAP* based on word mover's distance to various state-of-the-art approaches. The best and second-best classification accuracies are highlighted in **bold** and *italics*. We perform statistical significance tests of our method (*wmd-EMAP*) against all other methods, for a given dataset, and denote the outcomes by $\vee$ when the compared method is worse and $\wedge$ when our method is worse, while the absence of a symbol indicates insignificant differences. In terms of absolute accuracy, we observe that our method achieves state-of-the-art results in 2 out of 6 datasets.

shared commonly-occurring words in both the sentences. However, given that energy and hausdorff distances are reasonably fast to calculate and perform respectably well, they might be worth using in applications with a large number of long sentences.

Comparison versus kNN: *EMAP* almost always outperforms k-nearest neighbours based classification, for all the tested distance metrics. The performance boost for WMD is between a relative percentage accuracy of 0.5% to 14%. This illustrates the efficiency of the proposed manifold-learning method.

Query Sentence	Best Match Sentence	Cosine Sim
I have spent thousands of dollar's On Meyers cookware everthing from KitchenAid Anolon Prestige Faberware & Circulan just to name a few Though Meyers does manufacture very high quality pots & pans and I would recommend them to anyone it's just sad that if you have any problem with them under warranty you have to go throught the chain of command that never gets you anywhere even if you want to speak with upper management about the rudeness of the customer service department Their customer service department employees are always very rude and snotty and they act like they are doing you a favor to even talk to you about their products	When I opened the box I noticed corrosion on the lid When I contacted Rival customer service via email they told me I had to purchase a new lid I called and spoke with a customer service representative and they told me that a lid was not covered under warranty When I explained that I just opened it and it was defective they told me to just return the product that there was nothing that they were going to do After being treated this way I will NOT be purchasing any more Rival products if they don't stand behind their product VERY VERY poor customer service	0.997
This movie will bring up your racial prejudices in ways that most movies just elude to It demonstrates how connected we all are as people and how seperated we are by only one thing our viewpoints The acting is superb and you get one cameo appearance after another which is a treat Of course the soundtrack is terrific The ending is intense to witness one situation after another coming to an unfortunate finish	I waited years for this movie to be released in the United States As far as I was concerned it wasn't about the acting as much as it was about the feeling the actors wanted to portray in which they profoundly accomplished I would recommend this movie to anyone who can reach that one step deeper into the minds of creativity and passion and appreciate the struggles of rising above and beyond the pain of broken dreams	0.998
We see a phrase a lot when we visit how to sites for writers World building By this we mean the setting the characters and everything else where our story will occur For me this often means maps memories and visits since I write about where I live But if you'd like to see exactly what world building means head down to your local library and grab SALEM'S LOT by Stephen King When Stephen King mania first gripped the English speaking world I missed it I saw the film of CARRIE and hated it Years later at a guard desk on a long shift scheduled so suddenly that I hadn't had a chance to visit the library I read what was in the desk instead THINNER If I were Stephen King I'd have put a pen name on that crap as well One of King's fans brought me around She recommended THE SHINING Of course I thought of that Kubrick/Nicholson travesty No no she said read the book It's much different Yes it is It's fantastic for its perceptiveness Next up PET SEMATARY which scared the crap out of me And that my friends is not easy ON WRITING I've gushed about that enough times The films STAND BY ME and THE APT PUPIL So in the end I appreciate King and forgive him for CARRIE and I think he's forgiven himself	in the possibility that Steve Berry could ever transcend his not so great debut The Amber Room Romanov Prophecy started in the right direction Third Secret was OK but I think he hit his *peak* right there	0.955

Table 5: **Examples of best-matching sentences**. From the *amazon* reviews dataset using *wmd-EMAP*.

Comparison versus state-of-the-art methods: Consulting Table 4, it seems that *wmovers*, *pmeans* and *s-bert* form the strongest baselines as compared to our method, *wmd-EMAP* (*EMAP* with word mover's distance). Considering the statistical significance of the differences in performance between *wmd-EMAP* and the others, it can be seen that it is almost always equivalent to or better than the other state-of-the-art approaches. In terms of absolute accuracy, it *wins* in 3 out of 6 evaluations, where it has the highest classification accuracy, and comes out second-best for the others. Compared to it's closest competitor, the *word mover's embedding* algorithm, the performance of *wmd-EMAP* is found to be on-par (or slightly better, by 0.8% in the case of the *classic* dataset) to slightly worse (3% relative p.p., in case of the *twitter* dataset). Interestingly, both of the distance-based embedding approaches, *wmd-EMAP* and *wmovers*, are found to perform better than the siamese-BERT based approach, *s-bert*.

Thus, the overall conclusion from our empirical studies is that *EMAP* performs favourably as compared to various state-of-the-art approaches.

Examples of similar sentences with EMAP: We provide motivating examples of similar sentences from the *amazon* dataset, as deemed by our approach, in Table 5. As can be seen, our method performs quite well in matching complex sentences with varying topics and sentiments to their closest pairs. The first example pair has the theme of a customer who is unhappy about poor customer service in the context of cookware warranty, while the second one is about positive reviews of deeply-moving movies. The third example, about book reviews, is particularly interesting: in the first example, a reviewer is talking about how she disliked the first Stephen King work which she was exposed to, but subsequently liked all the next ones, while in the matched sentence the reviewer talks about a similar sentiment change towards the works of another author, Steve Berry. Thus in the last example, the similarity between sentences is the change of sentiment, from negative to positive, towards the works of books of particular authors.

6 Conclusions

In this work, we propose a novel mechanism to construct unsupervised sentence-embeddings by preserving properties of local neighbourhoods in the original space, as delineated by set-distance metrics. This method, which we term, *EMAP* or *Embeddings by Manifold Approximation and Projection* leverages a method from topological data analysis can be used as a framework with any distance metric that can discriminate between sets, three of which we test in this paper. Using both quantitative empirical studies, where we compare with state-of-the-art approaches, and qualitative probing, where we retrieve similar sentences based on our generated embeddings, we illustrate the efficiency of our proposed approach to be on-par or exceeding in-use methods. This work demonstrates the successful application of topological data analysis in sentence embedding creation, and we leave the design of better distance metrics and manifold approximation algorithms, particularly targeted towards NLP, for future research.

References

Hanan Aldarmaki and Mona Diab. 2018. Evaluation of unsupervised compositional representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2666–2677, Santa Fe, New Mexico, USA.

Nada Almarwani, Hanan Aldarmaki, and Mona Diab. 2019. Efficient sentence embedding using discrete cosine transform. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3672–3678, Hong Kong, China.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*.

Mikhail J. Atallah. 1983. A linear time algorithm for the hausdorff distance between convex polygons. Technical report, Department of Computer Science, Purdue University.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Karol Borsuk. 1948. On the imbedding of systems of compacta in simplicial complexes. In *Fundamenta Mathematicae*, volume 35, pages 217–234.

Gunnar Carlsson. 2009. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308.

Minmin Chen. 2017. Efficient vector representation for documents through corruption. *5th International Conference on Learning Representations*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

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

Wei Dong, Charikar Moses, and Kai Li. 2011. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th International Conference on World Wide Web*, pages 577–586.

M. . Dubuisson and A. K. Jain. 1994. A modified hausdorff distance for object matching. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 566–568 vol.1.

Geoffrey E Hinton and Sam T. Roweis. 2003. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 857–864.

Subhradeep Kayal and George Tsatsaronis. 2019. EigenSent: Spectral sentence embeddings using higher-order dynamic mode decomposition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4536–4546, Florence, Italy.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 957–966.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196.

Soledad Le Clainche and José M. Vega. 2017. Higher order dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 16(2):882–925.

John A. Lee and Michel Verleysen. 2007. *Nonlinear Dimensionality Reduction*, 1st edition.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

L. McInnes, J. Healy, and J. Melville. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. 2018. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119.

H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Ali Rahimi and Benjamin Recht. 2008. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184.

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

Lior Rokach and Oded Maimon. 2005. Clustering methods. In *The Data Mining and Knowledge Discovery Handbook*, pages 321–352.

Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326.

Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. 2018. Concatenated p-mean word embeddings as universal cross-lingual sentence representations. *CoRR*, abs/1803.01400.

Sidak Pal Singh, Andreas Hug, Aymeric Dieuleveut, and Martin Jaggi. 2019. Context mover's distance & barycenters: Optimal transport of contexts for building representations. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*.

Gábor J. Székely and Maria L. Rizzo. 2013. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249 – 1272.

George Tsatsaronis, Iraklis Varlamis, and Kjetil Nørvåg. 2012. Semafor: Semantic document indexing using semantic forests. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, page 1692–1696, New York, NY, USA.

Zihao Wang, Datong Zhou, Yong Zhang, Hao Wu, and Chenglong Bao. 2019. Wasserstein-fisher-rao document distance. *CoRR*, abs/1904.10294.

Lingfei Wu, Ian En-Hsu Yen, Kun Xu, Fangli Xu, Avinash Balakrishnan, Pin-Yu Chen, Pradeep Ravikumar, and Michael J. Witbrock. 2018. Word mover's embedding: From Word2Vec to document embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4524–4534, Brussels, Belgium.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. 2016. DocChat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 516–525, Berlin, Germany.

Contrastive Multi-document Question Generation

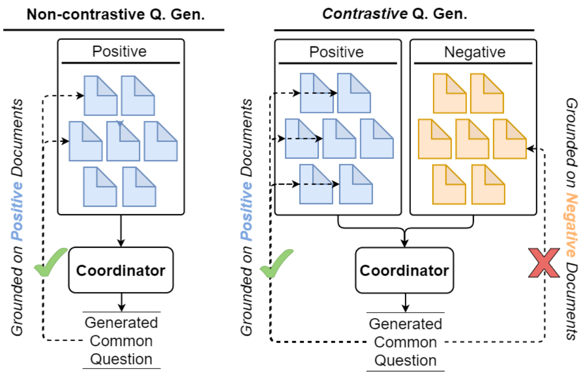
Woon Sang Cho⋆† **Yizhe Zhang**◦ **Sudha Rao**◦ **Asli Celikyilmaz**◦
Chenyan Xiong◦ **Jianfeng Gao**◦ **Mengdi Wang**⋆ **Bill Dolan**◦
⋆Princeton University, Princeton, NJ, USA
◦Microsoft Research, Redmond, WA, USA
⋆`{woonsang,mengdiw}@princeton.edu`
◦`{yizzhang,sudhra,aslicel,cxiong,jfgao,billdol}@microsoft.com`

Abstract

Multi-document question generation focuses on generating a question that covers the common aspect of multiple documents. Such a model is useful in generating clarifying options. However, a naive model trained only using the targeted ("positive") document set may generate too generic questions that cover a larger scope than delineated by the document set. To address this challenge, we introduce the contrastive learning strategy where given "positive" and "negative" sets of documents, we generate a question that is closely related to the "positive" set but is far away from the "negative" set. This setting allows generated questions to be more specific and related to the target document set. To generate such specific questions, we propose Multi-Source Coordinated Question Generator (MSCQG), a novel framework that includes a supervised learning (SL) stage and a reinforcement learning (RL) stage. In the SL stage, a single-document question generator is trained. In the RL stage, a coordinator model is trained to find optimal attention weights to align multiple single-document generators, by optimizing a reward designed to promote specificity of generated questions. We also develop an effective auxiliary objective, named Set-induced Contrastive Regularization (SCR) that improves the coordinator's contrastive learning during the RL stage. We show that our model significantly outperforms several strong baselines, as measured by automatic metrics and human evaluation. The source repository is publicly available at `www.github.com/woonsangcho/contrast_qgen`.

1 Introduction

User queries on web search engines can sometimes be vague. Search engines may resolve this ambiguity by suggesting clarification options back to the user in the form of questions (Braslavski et al., 2017; Aliannejadi et al., 2019; Zamani et al., 2020). However, asking the right clarification questions is a challenging information-seeking task, given a plethora of possible questions (Rao and Daumé III, 2018, 2019; Qi et al., 2020). One workaround is to take informational cues from the search engine results given the initial query. The clarification options are then generated from non-ranked and non-overlapping thematic partitions of the search engine results. The whole pipeline is akin to the pseudo-relevance feedback (Rocchio, 1971; Cao et al., 2008). This can significantly reduce the search space, and has the potential to generate correct clarification questions within the context (Cho



Positive set about:	*number of saturn's moons*
Negative set about:	*uranus how many moons*
Non-contrastive Q. Gen	*what is the largest moon*
Contrastive Q. Gen	*how many moons are there in saturn*

Figure 1: Non-contrastive and contrastive method for multidocument question generation. Left: non-contrastive modeling that takes input as a set of positive documents. However, model-generated questions from this method are rather generic and not specific to the input documents. Right: *contrastive* modeling, which considers both positive and negative document sets, and learns to generate questions that are more grounded on the positive document set.

---

† Work done when the author was an intern at Microsoft Research.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 12–30
April 19 - 23, 2021.

et al., 2019b).

This particular approach may involve three non-trivial phases: i) *retrieval*: gather the initial return documents by the search engine; ii) *partition*: partition the documents into semantically similar clusters in an unsupervised manner; iii) *multi-document question generation*: generate a clarification question by finding an "overlap" among documents in each cluster. In principle, the clarification questions should be **specific** to each cluster rather than generic and bland, otherwise it is counter to the objective of clarification (Radlinski and Craswell, 2017). In this work, we focus on developing a multi-document question generator to generate cluster-specific questions in the iii) step. Nevertheless, we believe our approach can be readily applied to multi-document text generation such as summarization (Liu and Lapata, 2019) and response generation (Zhang et al., 2020).

We address this challenge by leveraging contrastive learning. Given a set of positive documents $\mathfrak{D}^+$ and a set of negative documents $\mathfrak{D}^-$ (where $\mathfrak{D}^-$ is yet semantically close to $\mathfrak{D}^+$), we propose a new strategy to generate a question that is semantically relevant to $\mathfrak{D}^+$ and far away from $\mathfrak{D}^-$. Ideally, the model would use both $\mathfrak{D}^+$ and $\mathfrak{D}^-$ to identify distinguishing features between the two sets and constrain the generation to be *specific* to $\mathfrak{D}^+$. The similarity between the $\mathfrak{D}^+$ and $\mathfrak{D}^-$ makes the generation more challenging and forces the model to be as specific as possible in order to distinguish between the two sets. The comparison between the contrastive and non-contrastive multi-document question generation is illustrated in Figure 1.

This task is particularly challenging because i) there does not exist direct supervised *ground-truth* multi-document question given positive and negative sets of documents. ii) The whole procedure involves multiple aspects including language understanding, inter-document information aggregation, coordinative planning and language generation. In theory, the generator can be trained to maximize the chance that the generated question specifically retrieves the given document cluster, using RL. However, the space of possible sequence is prohibitively large which results in large variance in RL (Lewis et al., 2017). To effectively reduce the search space of RL, we employ a hybrid supervised learning (SL) and RL strategy. We also propose a novel reward-shaping auxiliary objective, Set-induced Contrastive Regularization (SCR) (Section 2), which heuristically drives the generation closer towards $\mathfrak{D}^+$, by *minimizing/maximizing* the KL divergence between the hypothesis distribution and distributions induced by $\mathfrak{D}^+/\mathfrak{D}^-$.

Our contributions are summarized below: i) We develop a novel Multi-Source Coordinated Question Generator (MSCQG) model that is trained using a hybrid hierarchical generation scheme. The document-specific generator is fine-tuned from GPT-2 and the inter-document coordinator is trained using reinforcement learning. ii) We introduce Set-induced Contrastive Regularization (SCR), an auxiliary regularizer that pushes MSCQG toward $\mathfrak{D}^+$ relative to $\mathfrak{D}^-$ while limiting the effect of $\mathfrak{D}^-$ in a principled manner. iii) Empirical results show that our model is able to generate more grounded and specific questions, significantly outperforming existing baseline models in automatic measures and human evaluation.

2 Method

Overview: The overview of our model is illustrated in Figure 2. The model consists of two major components: i) The *document-specific generator* generates a question from a single document, and is fine-tuned from OpenAI GPT-2. ii) The *inter-document coordinator* integrates multiple-document information from the *document-specific generator* instances. The coordinator is trained using reinforcement learning after fixing the document-specific generator.

During the RL training, at each generation time step, each (positive and negative) document will independently use the same generator trained from i) to predict the next token. The coordinator will learn to aggregate the probabilities to a *consensus probability* by maximizing a reward function. The reward function is designed to encourage the generated question to tie to the positive set and to be away from the negative set. The word newly generated from the *consensus probability* are concatenated to all documents as inputs for next time step.

Document-specific Generator: At the first pre-training stage, we load the publicly available GPT-2 (Radford et al., 2019) model as our underlying document-specific generator. The GPT-2 model leverages massive out-domain data and serves as a good initialization to generate grammatical and

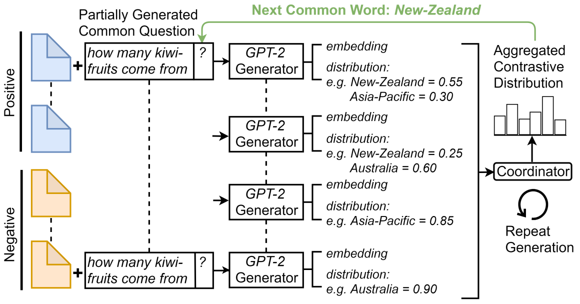


Figure 2: System overview. The example is an illustration using fictitious tokens for ease of understanding. Our MSCQG model learns to attend different weights and form a final aggregated distribution at each decoding time. The decision to enforce or penalize the negative set distributions to the aggregated distribution is controlled in a principled manner.

informative question. Then, we further fine-tune the language model on MS-MARCO (Nguyen et al., 2016) *selected* document as an input and the corresponding question as an output.

Ranking-based Rewards: Before moving to the RL training, we first describe calculating the reward signal based on retrieval statistics from a BERT-based ranker (Nogueira and Cho, 2019) ($Ranker$), a state-of-the-art model [*] in the MS-MARCO document retrieval task (Nguyen et al., 2016) , is trained to rank (document, question) pairs. This ranker assigns high scores for *true positive* document and question pairs. We assume the ranker delivers an accurate reward signal since it achieves good performance on the challenging MARCO retrieval task, which covers a vast range of general topics. Let $\tilde{q}$ be the generated question from the underlying generator block and coordinator with the positive and negative document sets ($\mathfrak{D}^+$ and $\mathfrak{D}^-$) as the input.

$$Ranker(d, \tilde{q}) = \text{score} \in (0, 1) \qquad (1)$$
$$\forall d \in \mathfrak{D}^+, \mathfrak{D}^-$$

We pair $\tilde{q}$ with each of the documents in the positive and negative set, and evaluate the question-document pairs through the ranker for answer-relevancy. Using the scores and their memberships in $\mathfrak{D}^+$ or $\mathfrak{D}^-$, we compute retrieval statistics, such as *Precision@10* and *mean-Average-Precision (mAP)* (Zhu, 2004) which are candidate non-differentiable rewards R.

Training an Inter-generator Coordinator via RL: Next, we train a coordinator system using policy gradient to optimize the reward described above. The separation between the generator and the coordinator aims to ease the RL training by significantly reducing the action space.

Note that the generator model is fixed during this stage. We find that using RL to train the entire generating pipeline yields large variance since the action space is large and the auto-regressive nature of the generation process further amplifies such variance. Therefore, we fix the underlying generator component and then on top of multiple instances of the underlying generator, we stack our coordinator model, which is trained using RL in isolation. Instead of training *both* token-level GPT-2 and document-level coordinator over multiple GPT-2 instances using RL, only the coordinator is trained using RL, which structure dramatically reduces variance.

The coordinator is a transformer-based (Vaswani et al., 2017) model to utilize its superior attention capabilities across input documents.Unlike Transformer Decoder (Liu et al., 2018), there is no causal mask. Instead, the coordinator model uses the hidden states updated every decoding time from the underlying fine-tuned GPT-2 (Radford et al., 2019) language model generators.

We add learned *cluster embedding* c_i to the input document hidden states h_i, similar to learned positional embedding (Devlin et al., 2019), to indicate whether the source document i is in $\mathfrak{D}^+$ or $\mathfrak{D}^-$.

$$x_i^0 = h_i + c_i \qquad (2)$$

The coordinator model consists of n recurrent transformers blocks (Vaswani et al., 2017), followed by three different feed-forward layers (FF_w, FF_v, and FF_z) to output w, v, and z.

$$\begin{aligned} x^k &= \text{Add-Norm}(u, \text{FF}_x(u)) && (3) \\ u &= \text{Add-Norm}(x^{k-1}, \text{MultiHead}(x^{k-1})) && (4) \\ &\quad \text{for } k = 1, \ldots, n \\ w &= \text{FF}_w(x^n) && (5) \\ v &= \text{FF}_v(x^n) && (6) \\ z &= \text{FF}_z(x^n) && (7) \end{aligned}$$

w and v are the 10-dimensional attention weights that sum to 1.0 among the positive documents $\mathfrak{D}^+$, and negative documents $\mathfrak{D}^-$.

z parametrizes η in how much the coordinator model penalizes, or sometimes reinforces,

---

[*] http://www.msmarco.org/leaders.aspx

weighted average of decoding distributions from the negative set $\mathfrak{D}^-$. η is a simple heuristic variation of $\tanh$ such that the image lies in $(-1, 0.5)$ for all real numbers $\mathbb{R}$. Thus, η is a damped penalization coefficient.

$$\eta(z) = -\frac{e^{2z} - 0.5}{e^{2z} + 1} \in (-1, 0.5) \quad \forall z \in \mathbb{R} \tag{8}$$

Given w, v, and z, we obtain the final question decoding distribution at test time t.

$$\pi_\theta^t = \frac{1}{C}\Big[\sum_{i \in \mathfrak{D}^+} w_{i,\theta}^t \pi_i^t - \eta(z_\theta^t) \cdot \sum_{i \in \mathfrak{D}^-} v_{i,\theta}^t \pi_i^t\Big]_+ \tag{9}$$

where θ is the coordinator's parameters, the subscript $+$ is a ReLU (Nair and Hinton, 2010) operator that selects non-negative weighted tokens, and C is the normalizing factor that converts it into a distribution. The concatenations of each input document in $\mathfrak{D}^+$ and $\mathfrak{D}^-$, EOS token, and partially decoded question word sequence are used to obtain new hidden states and next decoding distributions. The decoding process is repeated until the generation is complete.

Policy Gradient Loss: The policy gradient loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{PG}}(\theta) = -\mathbb{E}\Big[&(R(\tilde{q}|\mathfrak{D}^+, \mathfrak{D}^-) - R_{\text{baseline}}) \\ &\cdot \sum_t \log \pi_\theta^t(o_t|\tilde{q}_{<t}, G, \mathfrak{D}^+, \mathfrak{D}^-)\Big]\end{aligned} \tag{10}$$

With a complete generation $\tilde{q}$, a terminal retrieval statistics reward is computed from the *Ranker* scores and score memberships, noted as $R(\tilde{q}|\mathfrak{D}^+, \mathfrak{D}^-)$. This reward weights the sum of log-likelihoods of generating the observed words o_t given the generation so far $\tilde{q}_{<t}$, from the underlying generator G, and the two document sets. We use *oracle* questions as the policy gradient baseline for variance reduction in R_{baseline}. Results using a different policy gradient baseline are in Appendix.

Set-induced Contrastive Regularization: We further propose an auxiliary to provide richer signals when optimizing the coordinator model. The intuition is that we would like to encourage the coordinator model to generate questions *toward* the positive set $\mathfrak{D}^+$ relative to the negative set $\mathfrak{D}^-$. We name the regularizer as *Set-induced* Contrastive Regularization (SCR) because the decoding distributions from $\mathfrak{D}^+$ and $\mathfrak{D}^-$ guide the coordinator to learn to make contrasts between the two sets. Although the decoding distributions from $\mathfrak{D}^+$ and $\mathfrak{D}^-$ are not gold supervision signals, modifying distributional distance toward or away from them helps regulate specificity to $\mathfrak{D}^+$. The former idea can be formulated as *minimizing* the KL-divergence, evaluated at timestep t:

$$\begin{aligned}\min_\theta \mathcal{L}_{\text{KL},t}^{\text{pos}}(\theta) = \min_\theta \sum_{i \in \mathfrak{D}^+} \Big[&D_{\text{KL}}(\pi_\theta^t || \pi_i^t) \\ &+ D_{\text{KL}}(\pi_i^t || \pi_\theta^t)\Big]\end{aligned} \tag{11}$$

We minimize both the forward and the reverse KL divergence since the forward KL does not penalize high mass of π_θ where π_i does not. Likewise for the reverse KL. On the other hand, the latter idea can be formulated as *maximizing* the KL-divergence against the negative set, evaluated at time step t:

$$\begin{aligned}\max_\theta \mathcal{L}_{\text{KL},t}^{\text{neg}}(\theta) = \max_\theta \sum_{i \in \mathfrak{D}^-} \Big[&D_{\text{KL}}(\pi_\theta^t || \pi_i^t) \\ &+ D_{\text{KL}}(\pi_i^t || \pi_\theta^t)\Big]\end{aligned} \tag{12}$$

However, we need to *cap* the negative set penalty rather than naïvely maximizing it, more restrictively if the positive set and the negative sets are semantically close. Intuition is that if the KL divergence against the negative set is too large, then we do not penalize further. Therefore, we define our contrastive regularization function as follows:

$$\begin{aligned}\mathcal{L}_{\text{SCR}}(\theta) = \frac{1}{T}\sum_{t=1}^{T} \Big[&\mathcal{L}_{\text{KL},t}^{\text{pos}}(\theta) - \mathcal{L}_{\text{KL},t}^{\text{neg}}(\theta) \\ &\cdot \mathbb{1}_{\nu_t \cdot \mathcal{L}_{\text{KL},t}^{\text{neg}}(\theta) < \mathcal{L}_{\text{KL},t}^{\text{pos}}(\theta)}\Big]\end{aligned} \tag{13}$$

where T is the length of the completed generation, and ν_t is the similarity measure between positive and negative sets at decoding time t. Specifically,

$$\nu_t = \cos\text{sim}\left(\frac{1}{|\mathfrak{D}^+|}\sum_{i \in \mathfrak{D}^+} \pi_i^t, \frac{1}{|\mathfrak{D}^-|}\sum_{i \in \mathfrak{D}^-} \pi_i^t\right) \tag{14}$$

Negative Entropy Loss: We add negative entropy loss $\mathcal{L}_{\text{H}}$ across the attention weights w and v, averaged over T to encourage the model attend to all the documents rather than attend to a small subset of the documents and risk losing positive and

	Out-Sample IR				**Search-Engine Augmented IR**				
Model	mAP	RPrec	MRR (=MRR@10)	nDCG	mAP	RPrec	MRR	MRR@10	nDCG
Top-TFIDF @100	0.416	0.533	0.696	0.545	0.113	0.0588	0.0260	0.0050	0.181
Top-Frequent @100	0.680	0.742	0.921	0.779	0.171	0.129	0.0404	0.0119	0.204
MSQG (Cho et al. '19)	-	-	-	-	-	-	0.0704	**0.0441**	0.234
MSQG$_{GPT2}$	0.713	0.763	0.945	0.804	0.245	0.217	0.0714	0.0400	0.240
MSCQG$_{SCR}$	0.751	0.790	0.974	0.836	0.258	0.234	0.0745	0.0420	0.245
MSCQG$_{PG}$	0.753	0.791	0.978	0.838	0.256	0.232	0.0742	0.0421	0.244
MSCQG$_{PG+SCR}$	**0.767**	**0.803**	**0.981**	**0.849**	**0.265**	**0.242**	0.0748	0.0420	0.245
MSCQG$_{PG+SCR+H}$	0.765	0.800	0.976	0.847	0.262	0.239	**0.0759**	0.0434	**0.246**
Oracle Questions for $\mathfrak{D}^+$	0.759	0.797	0.976	0.842	0.292	0.273	0.0846	0.0495	0.256

Table 1: Retrieval performance. "Out-Sample IR" refers to the evaluation data sample that consists of 10+10 documents $\mathfrak{D}^+$ and $\mathfrak{D}^-$. "Search-Engine Augmented IR" refers to augmenting the out-sample into 100 documents in total through Lucene.

	BL-1	BL-2	BL-3	BL-4	METEOR	ROUGE_L	CIDEr	ST	EM	VE	GM
Oracle Question for $\mathfrak{D}^-$	0.449	0.291	0.177	0.100	0.215	0.428	1.076	0.547	0.766	0.617	0.697
Top-TFIDF @100	0.253	0.157	0.104	0.075	0.195	0.339	1.174	0.470	0.747	0.575	0.671
Top-Frequent @100	0.438	0.328	0.260	0.217	0.281	0.476	2.684	0.573	0.799	0.682	0.735
MSQG$_{GPT2}$	0.457	0.313	0.207	0.139	0.282	0.494	1.993	0.563	0.814	0.705	0.768
MSCQG$_{SCR}$	0.501	0.363	0.260	0.193	0.303	0.535	2.533	0.604	0.829	0.729	0.786
MSCQG$_{PG}$	0.562	0.418	0.310	0.234	0.304	0.565	2.702	0.630	0.844	0.734	0.798
MSCQG$_{PG+SCR}$	**0.589**	**0.449**	**0.339**	**0.262**	**0.323**	**0.591**	**2.994**	**0.647**	**0.858**	**0.759**	**0.815**
MSCQG$_{PG+SCR+H}$	0.573	0.436	0.330	0.255	0.321	0.583	2.946	0.641	0.851	0.752	0.808

Table 2: Comparison against the *oracle* MARCO questions for $\mathfrak{D}^+$. Since retrieval scores cannot give a complete picture of the generation, we aim to understand how close the generations are in terms of various metrics. The numbers show that our proposed model generates questions similar to the *oracle* MARCO questions. **Notations:** BL for BLEU; ST for Skip-Thought similarity; EM for Embedding Mean similarity; VE for Vector Extrema similarity; and GM for Greedy Matching.

negative set representational information.

$$\mathcal{L}_{\mathrm{H}}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\Big[\sum_{i\in\mathfrak{D}^+} w^t_{i,\theta}\log w^t_{i,\theta} + \sum_{i\in\mathfrak{D}^-} v^t_{i,\theta}\log v^t_{i,\theta}\Big] \tag{15}$$

We finally optimize for the following loss:

$$\mathcal{L}(\theta) = \lambda_1\mathcal{L}_{\mathrm{PG}}(\theta) + \lambda_2\mathcal{L}_{\mathrm{SCR}}(\theta) + \lambda_3\mathcal{L}_{\mathrm{H}}(\theta) \tag{16}$$

where $\lambda_{1,2,3}$ are the scaling hyper-parameters.

3 Experiments

Dataset: We use the MS-MARCO Q&A dataset (Nguyen et al., 2016) where for the Bing query q, we consider the top-10 retrieved documents as our positive set $\mathfrak{D}^+$. To get our negative set $\mathfrak{D}^-$, we use the Conversational Search[†] dataset, which contains additional annotations for the same MS-MARCO Bing queries, to find a query q' that is similar to q yet not a paraphrase and consider the top-10 documents retrieved for q' as our negative set $\mathfrak{D}^-$. In total, we gather 100K train/10K dev/10K eval data points. Details of the pre-processing, usage of additional annotations from the secondary dataset, and experimental configuration are in Appendix.

Automatic evaluation: The generated questions are evaluated through standard retrieval-based metrics: MRR and MRR10 (Voorhees, 1999; Radev et al., 2002), nDCG (Järvelin and Kekäläinen, 2002), precision, mAP. These metrics are computed from the 10 positive and 10 negative document sets (=: *Out-Sample IR*). In addition, as a standardized evaluation routine in the MS-MARCO Retrieval task, for each generated question, we use Lucene[‡] to retrieve the most relevant 100 MARCO documents via BM25 (Robertson and Zaragoza, 2009), and use the retrieved document set and a trained model to rank (document, generated question) pairs, thus compute the retrieval statistics (=: *Search-Engine Augmented IR*).

The generated questions are also evaluated in

[†] https://github.com/microsoft/MSMARCO-Conversational-Search

[‡] https://lucene.apache.org/

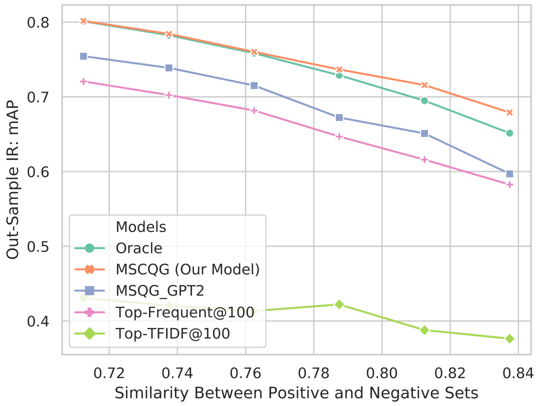


Figure 3: Out-Sample IR: mAP among $\mathfrak{D}^+$ and $\mathfrak{D}^-$

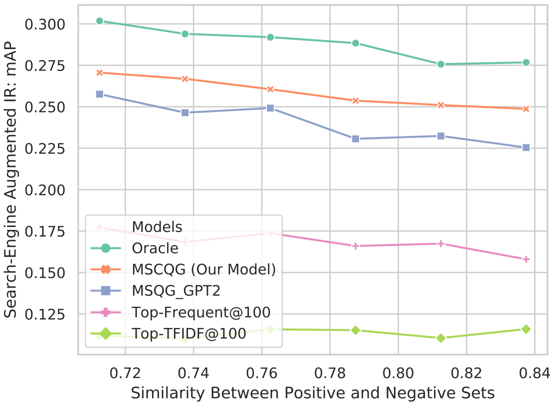


Figure 4: Search-Engine Augmented IR: mAP

Figure 3 shows that our model $\text{MSCQG}_{PG+SCR+H}$ outperforms the *oracle* questions by a small margin on the Out-Sample IR. In the larger retrieval evaluation using Lucene, it performs subpar against the *oracle* questions, but performs significantly better than all the considered baseline models, shown in Figure 4.

	Pair (M vs. B)			Pair (M vs. O)			Pair (B vs. O)		
Criteria	M	B	=	M	O	=	B	O	=
Ans.	**52.2**	17.5	30.3	39.2	19.8	**41.0**	32.3	**42.5**	25.2
Rel.	**53.3**	18.7	28.0	35.2	22.2	**42.7**	31.7	**44.2**	24.2
Flu.	**49.3**	22.3	28.3	**50.8**	24.7	24.5	**43.7**	32.7	23.7
Ovr.	**57.5**	21.3	21.2	**49.5**	27.0	23.5	38.3	**42.8***	18.8

	M			B			O		
Criteria	$d+$	$d-$	=	$d+$	$d-$	=	$d+$	$d-$	=
Ans.	**70.7**	10.2	19.2	**61.2**	14.0	24.8	**67.2**	13.7	19.2
Rel.	**72.2**	11.3	16.5	**62.2**	16.3	21.5	**70.3**	14.3	15.3
Ovr.	**72.0**	10.5	17.5	**63.2**	14.2	22.7	**69.0**	14.0	17.0

Table 3: Pairwise comparison and d^+/d^- comparison of human evaluation. M=MSCQG, B=MSQG_{GPT2}, O=*Oracle*. Preferences are expressed in percentage (%). Comparison results are statistically significant ($p < 0.01$) unless indicated *. Ans., Rel., Flu. and Ovr. denotes **Ans**werability, **Rel**evancy and **Flu**ency and **Over**all, respectively.

terms of BLEU (Papineni et al., 2002), ROUGE (Lin and Hovy, 2003), METEOR (Banerjee and Lavie, 2005), CIDEr (Vedantam et al., 2015), Greedy Matching (Rus and Lintean, 2012), Skip-Thought (Kiros et al., 2015), Embedding Average (Kenter et al., 2016) and Vector Extrema (Forgues et al., 2014) cosine similarities.

Human evaluation: We conduct human evaluation through Amazon Mechanical Turk where we evaluate questions generated by MSCQG, MSQG_{GPT2}, and the *oracle* question in four criteria: *fluency*, *relevancy*, *answerability*, and *overall*. First, we randomly select 600 (d, q_A, q_B) tuples where the d is any from $\mathfrak{D}^+$ and q_A, q_B from the three questions, and collect responses on which question is preferred over the other. Secondly, we evaluate 600 (d^+, d^-, q) tuples where given a question, d^+, d^- are randomly chosen from $\mathfrak{D}^+$ and $\mathfrak{D}^-$. This can determine questions' specificity to $\mathfrak{D}^+$ relative to $\mathfrak{D}^-$. Each sample is judged by 3 crowd-sourced workers who passed a rigorous spam-detection screening, totalling 3,600 samples to obtain reliable results. For details, see Appendix.

3.1 Baseline models

Multi-Source Question Generator: This model MSQG_{GPT2} is similar to MSQG in Cho et al. (2019b). It processes individual documents in parallel through the fine-tuned GPT-2 generator, rather than RNN-based Seq2Seq model in MSQG, and averages the decoding distributions at test time t.

$$\pi^t_{\text{MSQG}_{GPT2}} = \frac{1}{|pos|} \sum_i^{pos} \pi^t_i \tag{17}$$

Unlike MSQG in Cho et al. (2019b), no further heuristic modifications are made to the model.

Top-TFIDF@K: *Why do we not simply retrieve the top question implied by the 10 positive documents?* To this end, we design a retrieval baseline using the learned TF-IDF (Luhn, 1957; Jones, 1972; Salton and McGill, 1983) weights. This baseline re-evaluates the collection of retrieved questions from the corpus, gathered against each document in $\mathfrak{D}^+$ using TF-IDF, and retrieves the most relevant question. For design details, see Appendix.

Top-Frequent@K: Another retrieval model is to find an intersecting subset among all the 10 top-k question sets. For pseudo-code details, see Appendix.

3.2 Results and Analysis

Model comparison and ablation study: For simplicity, we abuse the term *oracle* by calling the ground-truth question that retrieves $\mathfrak{D}^+$ when constructing the dataset as the *oracle* question. However, these questions are not gold questions as they might not be the most relevant and specific questions to the given positive and negative sets.

Table 1 shows that our proposed model is effective at generating questions given multiple documents. In particular, it shows that policy gradient or set-induced contrastive regularization alone is effective in improving performance. The coordinator performs better when optimized for both policy gradient and regularization objectives.

The retrieval results for the questions that initially clustered $\mathfrak{D}^+$ sets are presented. Note that these are not *gold* questions because in most cases not all the retrieved documents in $\mathfrak{D}^+$ answer the questions. For clarity of our presentation, we abuse the term and name them as *oracle* questions. Search-engine augmented IR evaluation shows that our methods are upper-bounded by the *oracle* MARCO questions.

Entropy regularization improves the search-engine augmented IR scores, in particular, MRR. However, it is not crucial as supplemented by Table 2. For additional results, see Table 4 in Appendix.

Model performance v.s. similarities between $\mathfrak{D}^+$ and $\mathfrak{D}^-$: $\cos\mathrm{sim}(\mathfrak{D}^+, \mathfrak{D}^-)$ is approximated using the *oracle* questions that are available in the dataset. The similarity is computed by the cosine similarity of the two GEN-Encoder (Zhang et al., 2019) representations. Figures 3 and 4 show that our model generated questions are more grounded on $\mathfrak{D}^+$ than the baseline model generations. The more similar the two sets $\mathfrak{D}^+$ and $\mathfrak{D}^-$, the more difficult for the models, even humans, to distinguish which document is more relevant, if not answerable, given the generated question. The model outperforms the baseline model uniformly across different similarities between $\mathfrak{D}^+$ and $\mathfrak{D}^-$.

Role of $\mathfrak{D}^-$ by visualizing w, v, and z: Figures 5 and 6 show that our model MSCQG learns to gradually penalize $\mathfrak{D}^-$ as it sequentially generates words that are more grounded on $\mathfrak{D}^+$. Notice the roughly uniform weights across $\mathfrak{D}^+$ but increasing penalization weights across $\mathfrak{D}^-$, in decoding time.

η, which is controlled by the z, is learned to encourage, rather than discourage, certain words during decoding. The displayed $\mathfrak{D}^-$ weights v are multiplied by the dampening factor $-\eta(z)$ for interpretation purposes, thus it does not necessarily sum to 1, see equation 9. We observe that words that are not semantically distinguishing between $\mathfrak{D}^+$ and $\mathfrak{D}^-$, are encouraged by the coordinator to maintain readability. For example, the weights of the word *of* is mostly non-negative, whereas weights for other words are mostly negative. This indicates that the coordinator learns to selectively activate/suppress decoding of certain words by coordinating information from $\mathfrak{D}^+$ and $\mathfrak{D}^-$.

Human judgments: Table 3 shows that our model significantly outperforms the strong baseline in every aspect. Furthermore, we draw a more favorable conclusion toward our model-generated questions when compared against the *oracle* questions than from the automatic metrics, which are approximate yet reasonable metrics. The pairwise agreement between judges is 54% ± 1%. The Cohen's Kappa score is 0.19 ± 0.01. Note that this is a reasonable number given the *"same"* or ambiguous option in pairwise comparisons. $Ranker$ achieves a relatively high Pearson correlation of 0.6 with respect to human evaluation. For details, see Appendix.

4 Related Work

Multi-Source Encoder-Decoder: Ensemble set induction mechanism (Rokach, 2010) has been widely applied to neural machine translation (NMT) tasks (Bojar et al., 2014). Firat et al. (2016) introduced a new type of ensemble of NMT systems which take inputs as multiple sentences in different languages and output a translation into a single language. Each NMT system is trained on a mono-lingual source to target language translation dataset. Garmash and Monz (2016) further developed the multi-source encoder-decoder framework for multi-lingual NMT systems, by learning to assign uneven attention weights, called *expert combination weights*. To handle multi-source input, we take a similar multi-source encoder-decoder approach

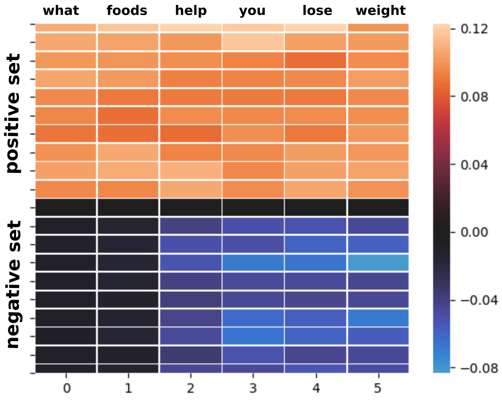


Figure 5: Example attention weights #1

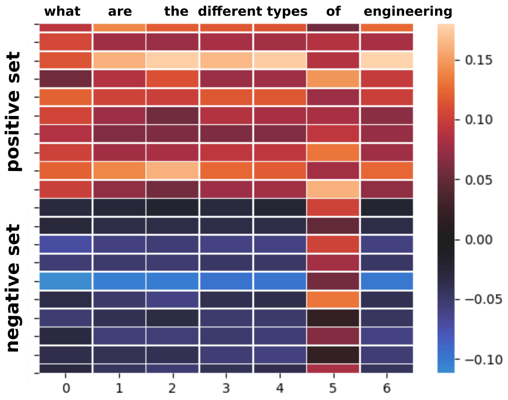


Figure 6: Example attention weights #2

Visualization of sequential attention weights. In the vertical axis, 0-9 indices indicate documents in $\mathfrak{D}^+$, and 10-19 in $\mathfrak{D}^-$. For explanation on $\mathfrak{D}^-$ weights v, see analysis below. Figure 5 shows that the model learns to push the sequential generation semantics more toward $\mathfrak{D}^+$ by gradually penalizing $\mathfrak{D}^-$. Figure 6 shows that frequent and semantically less distinguishing words such as *'of'* are encouraged even by $\mathfrak{D}^-$, which empirically aligns with our intuition for TF-IDF.

for our coordinator model. For such multi-lingual translation tasks, the target translation is available. However, in our task of generating multi-document questions, the target *does not exist* which makes it more challenging, thus we train via RL, rather than supervised learning.

Question Generation: Most prior work on question generation has been on single document i.e. given a document and an answer phrase in the document, generate a question that is answered by the answer phrase (Heilman, 2011; Rus et al., 2010). For a survey, see Pan et al. (2019). However, in our work, we aim to generate a multi-document question that is answerable by multiple input documents. Fan et al. (2018) propose a visual question generation model to generate natural questions about images using reinforcement learning where they use naturalness and human-like as reward signals. In our work, we use retrieval statistics, similar to Nogueira and Cho (2017), derived from a document-question ranker as the reward for training our coordinator model in isolation, rather than the entire generating pipeline.

Contrastive learning in NLP: Contrastive learning has been widely used in NLP (Smith and Eisner, 2005; Collobert et al., 2011; Bordes et al., 2013; Hjelm et al., 2019; Deng et al., 2020). Broadly, contrastive learning methods differentiate observed data from artificial negative examples. Gutmann and Hyvärinen (2010) leverages the Noise Contrastive Estimation (NCE) metric to differentiate the target sample from noise samples. Negative Sampling proposed by Mikolov et al. (2013) is a simplified variation of NCE loss. Recently, contrastive learning has also been employed in learning sentence representations (Clark et al., 2020). To our best knowledge, we are the first to leverage contrastive learning and establish set-induced penalization in the context of question generation.

5 Conclusion

We proposed a novel coordinator model that can generate questions that are more grounded on documents of interest. This coordinator model consists of transformer blocks, and is trained through reinforcement learning and an effective auxiliary: Set-induced Contrastive Regularization.

The rewards are derived from a publicly available state-of-the-art pre-trained ranker (Section 2) to compute retrieval statistics among $\mathfrak{D}^+$ and $\mathfrak{D}^-$. Our novel contrastive regularization induces generations to be more specific to $\mathfrak{D}^+$ than to $\mathfrak{D}^-$ while limiting the effect of $\mathfrak{D}^-$ in a principled manner by accounting for their semantic similarity.

We evaluate a generated question from each model by assessing how many of the input $\mathfrak{D}^+$ documents among a pool of relevant documents it can retrieve, based on the (document, question) ranker that is trained on the same wide-ranging domain. For a comprehensive automatic evaluation of the models, retrieval statistics are computed from a larger pool of relevant documents gathered via BM25. Experiment results show that our model significantly outperforms previous neural generation as well as strong retrieval baselines in both automatic and human metrics.

Given the promising comprehensive results of the proposed models and training approach, we can extend the framework with appropriate modifications and train via imitation learning algorithms, and this is left for future work.

Acknowledgements

We would like to thank Paul Bennett, Xiang Gao, Matthew Richardson, and Michel Galley at Microsoft Research for thoughtful feedback. We would also like to thank the anonymous reviewers for their time and providing helpful suggestions.

References

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 475–484, New York, NY, USA. Association for Computing Machinery.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Delphine Bernhard. 2010. Query expansion based on pseudo relevance feedback from definition clusters. In *Coling 2010: Posters*, pages 54–62, Beijing, China. Coling 2010 Organizing Committee.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Pavel Braslavski, Denis Savenkov, Eugene Agichtein, and Alina Dubatovka. 2017. What do you mean exactly? analyzing clarification questions in cqa. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 345–348, New York, NY, USA. Association for Computing Machinery.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2017. Ask the right questions: Active question reformulation with reinforcement learning.

Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. 2008. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, page 243–250, New York, NY, USA. Association for Computing Machinery.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana. Association for Computational Linguistics.

Woon Sang Cho, Pengchuan Zhang, Yizhe Zhang, Xiujun Li, Michel Galley, Chris Brockett, Mengdi Wang, and Jianfeng Gao. 2019a. Towards coherent and cohesive long-form text generation. In *Proceedings of the First Workshop on Narrative Understanding*, pages 1–11.

Woon Sang Cho, Yizhe Zhang, Sudha Rao, Chris Brockett, and Sungjin Lee. 2019b. Generating a common question from multiple documents using multi-source encoder-decoder models. In *The 3rd Workshop on Neural Generation and Translation*.

Eric Chu and Peter J. Liu. 2018. Unsupervised neural multi-document abstractive summarization. *CoRR*, abs/1810.05739.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(76):2493–2537.

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020. Residual energy-based models for text generation. In *International Conference on Learning Representations*.

Nina Dethlefs and Heriberto Cuayáhuitl. 2010. Hierarchical reinforcement learning for adaptive text generation. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 37–45. Association for Computational Linguistics.

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

Rui Dong and David Smith. 2018. Multi-input attention for unsupervised OCR correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2363–2372, Melbourne, Australia. Association for Computational Linguistics.

Marco Dorigo and Marco Colombetti. 1994. Robot shaping: Developing autonomous agents through learning. *Artificial intelligence*, 71(2):321–370.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Zhihao Fan, Zhongyu Wei, Siyuan Wang, Yang Liu, and Xuanjing Huang. 2018. A reinforcement learning framework for natural question generation using bi-discriminators. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1763–1774, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman-Vural, and Kyunghyun Cho. 2016. Zero-resource translation with multi-lingual neural machine translation. *CoRR*, abs/1606.04164.

Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. 2014. Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop*, volume 2.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational AI. *arXiv preprint arXiv:1809.08267*.

Ekaterina Garmash and Christof Monz. 2016. Ensemble learning for multi-source neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1409–1418, Osaka, Japan. The COLING 2016 Organizing Committee.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy. PMLR.

Vrindavan Harrison and Marilyn Walker. 2018. Neural generation of diverse questions using answer focus, contextual and linguistic features.

Michael Heilman. 2011. *Automatic factual question generation from text*. Ph.D. thesis, Carnegie Mellon University.

Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR 2019*. ICLR.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Srinivasan Janarthanam and Oliver Lemon. 2009. Learning lexical alignment policies for generating referring expressions for spoken dialogue systems. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pages 74–81, Athens, Greece. Association for Computational Linguistics.

Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E Turner, and Douglas Eck. 2017. Sequence tutor: Conservative fine-tuning of sequence generation models with kl-control. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1645–1654. JMLR. org.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

Tom Kenter, Alexey Borisov, and Maarten De Rijke. 2016. Siamese cbow: Optimizing word embeddings for sentence representations. *arXiv preprint arXiv:1606.04640*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *arXiv preprint arXiv:1506.06726*.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 889–898, Beijing, China. Association for Computational Linguistics.

Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. Adapting the neural encoder-decoder framework from single to multi-document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141, Brussels, Belgium. Association for Computational Linguistics.

Oliver Lemon. 2008. Adaptive natural language generation in dialogue using reinforcement learning. In *Proceedings of the 12th SEMdial Workshop on the Semantics and Pragmatics of Dialogues*, pages 149–156.

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125*.

Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. 2018a. NPRF: A neural pseudo relevance feedback framework for ad-hoc information retrieval. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4482–4491, Brussels, Belgium. Association for Computational Linguistics.

Ruizhi Li, Xiaofei Wang, Sri Harish Reddy Mallidi, Takaaki Hori, Shinji Watanabe, and Hynek Hermansky. 2018b. Multi-encoder multi-resolution framework for end-to-end speech recognition. *CoRR*, abs/1811.04897.

Jindřich Libovický and Jindřich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 196–202, Vancouver, Canada. Association for Computational Linguistics.

Jindřich Libovický, Jindřich Helcl, and David Mareček. 2018. Input combination strategies for multi-source transformer decoder. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 253–260, Belgium, Brussels. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 71–78, Stroudsburg, PA, USA.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *International Conference on Learning Representations*.

Yang Liu and Mirala Lapata. 2019. Hierarchical transformers for multidocument summarization. In *ACL*.

Yi Luan, Yangfeng Ji, Hannaneh Hajishirzi, and Boyang Li. 2016. Multiplicative representations for unsupervised semantic role induction. In *ACL*.

Hans Peter Luhn. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Donald Metzler and W. Bruce Croft. 2007. Latent concept expansion using markov random fields. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 311–318, New York, NY, USA. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Nasrin Mostafazadeh, Chris Brockett, Bill Dolan, Michel Galley, Jianfeng Gao, Georgios Spithourakis, and Lucy Vanderwende. 2017. Image-grounded conversations: Multimodal context for natural question and response generation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 462–472, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA. Omnipress.

Andrew Y Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016.*

Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2019. Multi-style generative reading comprehension. *CoRR*, abs/1901.02262.

Yuta Nishimura, Katsuhito Sudoh, Graham Neubig, and Satoshi Nakamura. 2018. Multi-source neural machine translation with missing data. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 92–99, Melbourne, Australia. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

Franz Josef Och and Hermann Ney. 2001. Statistical multi-source translation. In *In MT Summit 2001*, pages 253–258.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *CoRR*, abs/1905.08949.

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

Daraksha Parveen and Michael Strube. 2014. Multi-document summarization using bipartite graphs. In *TextGraphs-9: the workshop on Graph-based Methods for Natural Language Processing*.

Ramakanth Pasunuru and Mohit Bansal. 2017. Reinforced video captioning with entailment rewards. *arXiv preprint arXiv:1708.02300*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Peng Qi, Yuhao Zhang, and Christopher D Manning. 2020. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. *arXiv preprint arXiv:2004.14530*.

Dragomir Radev. 2000. A common theory of information fusion from multiple text sources step one: Cross-document structure. In *1st SIGdial Workshop on Discourse and Dialogue*, pages 78–83.

Dragomir R. Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating web-based question answering systems. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 117–126, New York, NY, USA. Association for Computing Machinery.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.

Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2737–2746, Melbourne, Australia. Association for Computational Linguistics.

Sudha Rao and Hal Daumé III. 2019. Answer-based Adversarial Training for Generating Clarification Questions. In *Proceedings of the 2019 Conference* *of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 143–155, Minneapolis, Minnesota. Association for Computational Linguistics.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.

Verena Rieser and Oliver Lemon. 2009. Natural language generation as planning under uncertainty for spoken dialogue systems. In *Empirical methods in natural language generation*, pages 105–120. Springer.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323.

Lior Rokach. 2010. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1):1–39.

A Rothe, Brenden Lake, and Todd Gureckis. 2016. Asking and evaluating natural language questions. In *Proceedings of the 38th Annual Conference of the Cognitive Science Society*.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. 1988. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1.

Vasile Rus and Mihai Lintean. 2012. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 157–162. Association for Computational Linguistics.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 251–257. Association for Computational Linguistics.

G. Salton. 1971. *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Gerard Salton and Michael J McGill. 1983. *Introduction to modern information retrieval*. mcgraw-hill.

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

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany. Association for Computational Linguistics.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 354–362, Ann Arbor, Michigan. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 1057–1063. MIT Press.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Ellen M. Voorhees. 2001. The trec question answering track. *Nat. Lang. Eng.*, 7(4):361–378.

EM Voorhees. 1999. Proceedings of the 8th text retrieval conference. *TREC-8 Question Answering Track Report*, pages 77–82.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Conference on Empirical Methods in Natural Language Processing*.

Yizhong Wang, Kai Liu, Jing Liu, Wei He, Yajuan Lyu, Hua Wu, Sujian Li, and Haifeng Wang. 2018. Multi-passage machine reading comprehension with cross-passage answer verification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1918–1927, Melbourne, Australia. Association for Computational Linguistics.

P. J. Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, pages 4–11, New York, NY, USA. ACM.

Ming Yan, Jiangnan Xia, Chen Wu, Bin Bi, Zhongzhou Zhao, Ji Zhang, Luo Si, Rui Wang, Wei Wang, and Haiqing Chen. 2018. A deep cascade model for multi-document reading comprehension. *CoRR*, abs/1811.11374.

Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *The Web Conference 2020 (formerly WWW conference)*.

Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, pages 403–410, New York, NY, USA. ACM.

Hongfei Zhang, Xia Song, Chenyan Xiong, Corby Rosset, Paul N. Bennett, Nick Craswell, and Saurabh Tiwary. 2019. Generic intent representation in web search. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 65–74, New York, NY, USA. ACM.

Jianmin Zhang, Jiwei Tan, and Xiaojun Wan. 2018. Adapting neural single-document summarization model for abstractive multi-document summarization: A pilot study. In *International Conference on Natural Language Generation.*

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*.

Mu Zhu. 2004. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2:30.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 30–34, San Diego, California. Association for Computational Linguistics.

Appendix

Appendix A. Data Pre-processing Details

Data Pre-processing: MS-MARCO Q&A data-set (Nguyen et al., 2016) contains 1,010,916 questions, in which each question is associated with top-10 documents. Each data point contains a question and its top-10 returned documents from the Bing search engine[§]. This question is not a target itself since not all the top-10 retrieved documents answer the question. However, it can give relative evaluation against a model-generated question based on the top-10 retrieved documents. We target a broader class of problems where only document groups are available but no such group-inducing or *oracle* questions.

In fact, among the top-10 retrieved documents, often one document is labeled '*selected*' by human annotators to indicate that the document answers the question (*true positive*), and left unknown or unlabeled for the rest of the documents, implying they may or may not answer the question (*true*

[§] `https://www.bing.com`

negative or *false negative*). This label information is used to train the underlying generator block of their MSQG model (Cho et al., 2019b). A single *selected* MS-MARCO (Nguyen et al., 2016) document is fed into a long short-term memory-based sequence-to-sequence model to output the corresponding question. An example of the input *selected* document is: *The House of Representatives shall be composed of Members chosen every second Year by the People of the several States.... Article I, Section 2, Clause 1*, and the corresponding question is: *how long is a term for a member of the house of representatives*. We chose this dataset since the question that retrieve the top-10 documents can shed light to relative performance of our model.

To find two 10-document sets $\mathfrak{D}^+$ and $\mathfrak{D}^-$ that are similar, we find a pair of questions that are semantically similar. However, computing pair-wise similarities among roughly 1 million questions is computationally intractable. Therefore, we leverage another dataset: MS-MARCO-Conversational Search[¶]: an artificially constructed public dataset that simulate user search sequences.

Each data point or session is an artificial sequence of similar questions grounded on true user behavior. Since many similar questions are grouped together, we can reduce the search space for finding pairs of similar questions. Then we take pairs of high semantic similarity (≥ 0.7) yet not a paraphrase (≤ 0.85 following their classification criteria) using GEN-Encoder (Zhang et al., 2019) which two associated 10-document sets do not have overlaps, primarily for prototype evaluation convenience. For deployment models, one may choose to allow overlaps between two sets for more challenging learning. From the two similar 10-document sets, either one is set to positive $\mathfrak{D}^+$ or negative $\mathfrak{D}^-$, yielding two data points for the our derived dataset.

These pre-processing steps yield 346,215 data points, each of which contains a pair of positive and negative questions, and positive and negative 10-document sets. Training MSCQG on the entire dataset requires processing about 7 million MARCO documents. This is computationally intensive and takes about two days on 8 Nvidia Tesla V100 GPU cards for a single epoch. Therefore, for building small research prototypes and benchmarks, we will also release a subset of the data, that consists of 100K/10K/10K training, development, and evaluation data points.

Appendix B. Data Example

***Oracle* question for $\mathfrak{D}^+$:**
number of saturn's moons

***Oracle* question for $\mathfrak{D}^-$:**
uranus how many moons

Positive Set $\mathfrak{D}^+$:

1. moons of saturn. there are 62 moons orbiting saturn. the moons of saturn vary not only in size but also in composition and shape. the largest of the moons of saturn is the aptly named titan, more than 5,000 km across and is bigger than mercury. there are 7 major moons of saturn and the rest are grouped based on the mythology from which it is taken.

2. iapetus with a diameter of 1,470 km, it is the 3rd largest moon of saturn. it was discovered by giovanni cassini in 1671. it has a distinct feature of having a bright and dark hemisphere. dione the 4th largest moon of saturn named after a vague character in greek mythology.

3. titan is the largest of saturn's moons and the first to be discovered. titan is the only moon in the solar system known to have a significant atmosphere. nitrogen and methane extend around the moon 10 times as far into space as earth's atmosphere, sometimes falling to the surface in the form of methane rain.

4. saturn has at least 150 moons and moonlets, 53 of which have formal names. titan, the largest, comprises more than 90% of the mass in orbit around saturn, including the rings. saturn's second-largest moon, rhea, may have a tenuous ring system of its own, along with a tenuous atmosphere.

5. their journeys around the ringed planet average from half an earth day to just over four earth years. saturn's moons formed early in the history of the solar system. one of the moons, titan, makes up 96 percent of the mass orbiting the planet. scientists think that the system may have originally housed

---

[¶] https://github.com/microsoft/MSMARCO-Conversational-Search *two such moons, but the second broke up, creating the debris that formed the rings and smaller, inner moons.*

6. saturn has a prominent ring system that consists of nine continuous main rings and three discontinuous arcs and that is composed mostly of ice particles with a smaller amount of rocky debris and dust. sixty-two moons are known to orbit saturn, of which fifty-three are officially named.

7. sixteen of the moons are tidally locked, with one face permanently turned toward saturn. the first moon was discovered in 1655. over the next 200 years, the other seven major satellites were spotted. by 1997, astronomers on earth had found 18 moons in orbit around the planet.

8. saturn is the sixth planet from the sun and the second-largest in the solar system, after jupiter. it is a gas giant with an average radius about nine times that of earth. although only one-eighth the average density of earth, with its larger volume saturn is just over 95 times more massive.

9. this temporary name usually consists of the year of discovery and a number indicating the order of discovery in that year. in the case of saturn's moons, these provisory names follow the format s/2005-s1, s/2005-s2 etc. the first s (before the slash) is for saturn. the second s (after the dash) is for satellite.

10. this does not include the hundreds of moonlets comprising the rings. titan, saturn's largest moon, and the second-largest in the solar system, is larger than the planet mercury, although less massive, and is the only moon in the solar system to have a substantial atmosphere.

Negative Set $\mathfrak{D}^-$:

11. uranus has 27 moons that we know of. five of the moons are large and the rest are much smaller. the five large moons are called miranda, ariel, umbriel, titania, and oberon. titania is the largest moon of uranus and it is covered with small craters, a few large craters, and very rough rocks. ariel is the brightest moon of uranus and has canyons and valleys as well as a lot of craters. umbriel is very dark.

12. uranus can't seem to catch a break these days. besides spinning on its side like the drunkard of the solar system and being the butt of everyone's jokes, new research suggests several of its tiny moons will collide in a million years. uranus can't seem to catch a break these days.

13. the gas giant uranus is the third largest planet in our solar system, has many moons, a ring system, and composed of gases and ices. universe today space and astronomy news login

14. the researchers used cressida's mass and orbit to determine its possible doom. since uranus' 27 moons are tightly packed together, the team posits that in a million years, cressida will likely have a deadly encounter with one of its neighboring moons, called desdemona. previous research and simulations suggest cupid and belinda will also probably smack into each other some time between 1,000 and 10 million years from now.

15. puck, at 162 km, is the largest of the inner moons of uranus and the only one imaged by voyager 2 in any detail while puck and mab are the two outermost inner satellites of uranus. all inner moons are dark objects.

16. uranus, which takes its name from the greek god of the sky, is a gas giant and the seventh planet from our sun. it is also the third largest planet in our solar system, ranking behind jupiter and saturn. like its fellow gas giants, it has many moons, a ring system, and is primarily composed of gases that are believed to surround a solid core.

17. in 1986, the voyager 2 spacecraft hit the jackpot while studying uranus and discovered 10 other moons, including desdemona and cressida. since then, hubble observations have helped bring that number up to 27 for now.

18. at an average distance of 3 billion km from the sun, it takes uranus roughly 84 years (or 30,687 days) to complete a single orbit of the sun. 1 the rotational period of the interior of uranus is 17 hours, 14 minutes. as with all giant planets, its upper atmosphere experiences strong winds in the direction of rotation.

19. uranus' size, mass and orbit: with a mean *radius of approximately 25,360 km, a volume of 6.833—10^13 km3, and a mass of 8.68 — 10^25 kg, uranus is approximately 4 times the sizes of earth and 63 times its volume.*

20. uranus has 27 known satellites, which are divided into the categories of larger moons, inner moons, and irregular moons (similar to other gas giants). the largest moons of uranus are, in order of size, miranda, ariel, umbriel, oberon and titania.

Appendix C. Retrieval Baselines

Top-TFIDF@K and **Top-Frequent@K**

The retrieval baselines are designed to give a relative insight into the performance between MSQG in Cho et al. (2019b) and our novel coordinator model. We use Lucene to retrieve *questions* instead of documents from a corpus composed of the 1,010,916 MS-MARCO questions. The retrieved questions from Top-TFIDF@K and Top-Frequent@K baselines are evaluated in the same manner as the generated ones.

For the intersection to be non-empty, k should be sufficiently large. However, even for $k = 1000$, there were no intersecting subset questions for almost all cases. Therefore, we relax the intersection among all 10 retrieved sets, into finding the most frequently occurring question among the 10 top-k retrieved sets. $k = 100$ was an appropriate value that is not too large to retrieve remotely relevant questions, and not too small to yield vastly different retrieval sets. If there are multiple questions with the same count, we randomly choose one.

Algorithm 1 Top-TFIDF@K

```
Input: $\mathfrak{D}^+$, Corpus $\mathbb{C}$
For each $d \in \mathfrak{D}^+$, retrieve top-K questions in $\mathbb{C}$;
Using all unique questions $\mathbb{Q}$, compute TF-IDF;
Let $\Psi$ be the TF-IDF transform operator;
$q^* = \arg\max_{q \in \mathbb{Q}} \sum_{d \in \mathfrak{D}^+} \cos\text{sim}\,(\Psi_q, \Psi_d)$;
Output: $q^*$
```

Algorithm 2 Top-Frequent@K

```
Input: $\mathfrak{D}^+$, Corpus $\mathbb{C}$
For each $d \in \mathfrak{D}^+$, retrieve top-K questions in $\mathbb{C}$;
Let $\mathbb{S}_d$ be the retrieved set for each $d$;
$q^* = \arg\max_{q \in \mathbb{Q}} \sum_{d \in \mathfrak{D}^+} \mathbb{1}_{q \in \mathbb{S}_d}$;
Output: $q^*$
```

Appendix D. Experiment Configurations

Document-specific GPT-2 Generator: From each document i, the generator yields its final layer hidden state $h_i \in \mathbb{R}^H$ ($H = 768$ is the hidden dimension) and a document-specific discrete output distribution $\pi_i \in \mathbb{R}^V$ ($V = 50257$ is the vocabulary dimension) from the learned language model head.

Coordinator: The input size is 20 with the dimensionality of the embeddings and hidden states as 768. The number of recurrent layers is 2, with 4 attention heads in each layer. The epsilon value used in the layer normalization is set to 1e$-$5. The number of cluster embeddings is 2 (positive or negative). The standard deviation of the truncated normal initializer for weight matrices is 0.02. $\lambda_1, \lambda_2, \lambda_3 = 1.0, 100.0, 0.1$. We performed coarse hyper-parameter search for equidistant values in log-scale for all $\lambda_1, \lambda_2, \lambda_3$, and use the best configuration. Maximum generation length is 20 tokens. We use the BERT (Devlin et al., 2019) version of Adam optimizer (Kingma and Ba, 2014) with weight decay of 0.01 and learning rate of 1e$-$5.

We trained the coordinator model by maximizing *Precision@10* with *oracle* questions as the policy gradient baseline. It is reasonable to weigh the documents unevenly because often times not all the top-10 retrieved documents from the Bing search engine share the same content. Thus, we leave to the model to learn the optimal attention weights among positive and negative sets that produce a more grounded question. Additional experiment results using a different baseline - self-critic (Rennie et al., 2017) - is shown in Table 4. This shows that our proposed model framework is effective even with any of the two policy gradient baselines. Conceptually, the coordinator model would generate a question that can better retrieve the documents from the positive set, aided by the negative set.

Appendix E. Human Evaluation Details

We performed two human evaluations: In the first experiment, we showed judges one randomly selected positive document, which is about 300 words long, followed by a pair of questions from the three sources. Judges were asked to evaluate which one of the two questions is preferred based on four criteria. For each pair of three sources, we evaluated 200 same random samples for each judge (or 600 samples for the 3 judges), totalling 1,800 samples.

In the second experiment, human annotators evaluated contrastive ability from 1,800 samples of one question, followed by two documents each from the positive and negative sets. Note that our model is trained to generate questions, accounting for the negative set.

The results were averaged across all samples and judges.

For computing the Pearson correlation between the ranker and human evaluation results, we map *Option A preferred* $\rightarrow 0$, *Same* $\rightarrow 0.5$, *Option B preferred* $\rightarrow 1$, accounting for the random assignments between A and B. This projection ensures that image of two metrics are the same (between 0 and 1). Then we compute the correlation value between two results.

	Out-Sample IR				**Search-Engine Augmented IR**				
Model	mAP	RPrec	MRR (=MRR@10)	nDCG	mAP	RPrec	MRR	MRR@10	nDCG
Top-TFIDF @100	0.416	0.533	0.696	0.545	0.113	0.0588	0.0260	0.0050	0.181
Top-Frequent @100	0.680	0.742	0.921	0.779	0.171	0.129	0.0404	0.0119	0.204
MSQG (Cho et al. '19)	-	-	-	-	-	-	0.0704	0.0441	0.234
MSQG$_{GPT2}$	0.713	0.763	0.945	0.804	0.245	0.217	0.0714	0.0400	0.240
MSCQG$_{SCR}$	0.751	0.790	0.974	0.836	0.258	0.234	0.0745	0.0420	0.245
MSCQG$^{\text{self-critic,null-neg}}_{PG+SCR+H}$	0.714	0.764	0.945	0.805	0.247	0.220	0.0724	0.0407	0.241
MSCQG$^{\text{self-critic}}_{PG}$	**0.762**	**0.798**	**0.982**	**0.845**	0.259	0.237	0.0746	0.0420	0.244
MSCQG$^{\text{self-critic}}_{PG+SCR}$	0.760	0.797	0.977	0.843	0.260	0.236	0.0744	0.0416	0.245
MSCQG$^{\text{self-critic}}_{PG+SCR+H}$	0.760	0.797	0.977	0.843	**0.262**	**0.238**	**0.0771**	**0.0444**	**0.247**
MSCQG$^{\text{orcl-critic,null-neg}}_{PG+SCR+H}$	0.717	0.766	0.950	0.808	0.246	0.220	0.0722	0.0404	0.241
MSCQG$^{\text{orcl-critic}}_{PG}$	0.753	0.791	0.978	0.838	0.256	0.232	0.0742	0.0421	0.244
MSCQG$^{\text{orcl-critic}}_{PG+SCR}$	**0.767**	**0.803**	**0.981**	**0.849**	**0.265**	**0.242**	0.0748	0.0420	0.245
MSCQG$^{\text{orcl-critic}}_{PG+SCR+H}$	0.765	0.800	0.976	0.847	0.262	0.239	**0.0759**	**0.0434**	**0.246**
Oracle Questions for $\mathfrak{D}^+$	0.759	0.797	0.976	0.842	0.292	0.273	0.0846	0.0495	0.256

Table 4: Additional retrieval performance using self-critic (Rennie et al., 2017) baseline in the policy gradient, applicable to datasets with no *oracle* questions. It shows that our framework is also effective using a different baseline. The superscript *null-neg* denotes models that do not use negative attentions when generating questions. This shows the importance of the negative set in promoting specificity in the generated question. It further corroborates that the non-uniform weighted-sum scheme among $\mathfrak{D}^+$ improves performance because not all documents in $\mathfrak{D}^+$ revolve around the same topic, and the model learns to address this nature of the dataset through unequal weights and generate a more representative question.

Disambiguatory Signals are Stronger in Word-initial Positions

Tiago Pimentel[ð] **Ryan Cotterell**[ð,ʕ] **Brian Roark**[ə]
[ð]University of Cambridge [ʕ]ETH Zürich [ə]Google
`tp472@cam.ac.uk, ryan.cotterell@inf.ethz.ch, roark@google.com`

Abstract

Psycholinguistic studies of human word processing and lexical access provide ample evidence of the preferred nature of word-initial versus word-final segments, e.g., in terms of attention paid by listeners (greater) or the likelihood of reduction by speakers (lower). This has led to the conjecture—as in Wedel et al. (2019b), but common elsewhere—that languages have evolved to provide more information earlier in words than later. Information-theoretic methods to establish such tendencies in lexicons have suffered from several methodological shortcomings that leave open the question of whether this high word-initial informativeness is actually a property of the lexicon or simply an artefact of the incremental nature of recognition. In this paper, we point out the confounds in existing methods for comparing the informativeness of segments early in the word versus later in the word, and present several new measures that avoid these confounds. When controlling for these confounds, we still find evidence across hundreds of languages that indeed there is a cross-linguistic tendency to front-load information in words.[1]

1 Introduction

The psycholinguistic study of human lexical access is largely concerned with the incremental processing of words—whereby, as individual sub-lexical units (e.g., phones) are perceived, listeners update their expectations of the word being spoken. One common tenet of such studies is that the disambiguatory signal contributed by units early in the word is stronger than that contributed later—i.e. **disambiguatory signals are front-loaded in words**. This intuition is derived from ample indirect evidence that the beginnings of words are more important for humans during word processing—including, e.g., evidence of increased attention to word beginnings (Nooteboom, 1981, *inter alia*) or evidence of increased levels of phonological reduction in word endings (van Son and Pols, 2003b).

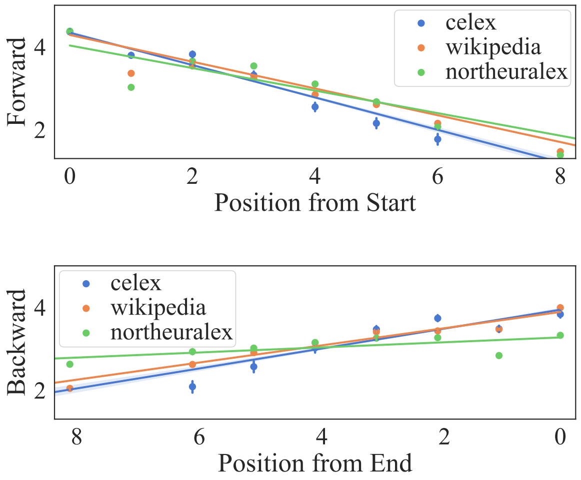


Figure 1: Forward and Backward Surprisals with LSTM model from Pimentel et al. (2020). The bottom plot has been flipped horizontally such that it visually corresponds to the normal string direction.

To analyse this front-loading effect, researchers have investigated the information provided by segments in words. van Son and Pols (2003a,b) showed that, in Dutch, a segment's position in a word is a very strong predictor of its conditional surprisal, with later segments being more predictable than earlier ones—a result which we show to arise directly from its definition in §3.3.1. Recently King and Wedel (2020) and Pimentel et al. (2020) confirmed the effect on many more languages.

Their analysis, however, presents an inherent confound between the amount of conditional information available to a model and the surprisal of the subsequent segment—see Fig. 1 for results illustrating this. Using the LSTM training recipes from Pimentel et al. (2020),[2] we calculated the conditional surprisal at each segment position within the words across all languages in three datasets.[3] The top-half of Fig. 1 shows that, indeed, positions

---

[1] Our code is available at https://github.com/tpimentelms/frontload-disambiguation.

[2] https://github.com/tpimentelms/phonotactic-complexity

[3] See §3 and §5 for specifics on training and data. Each segment corresponds to a single phone in CELEX and NorthEuraLex, and to a single grapheme in Wikipedia.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 31–41
April 19 - 23, 2021.

earlier in the string have higher surprisal than positions later in the string, supporting the thesis of higher informativity earlier in words. The bottom-half shows that modelling the strings right-to-left instead of left-to-right reverses the resulting effect.

This decouples conditional surprisal from the disambiguatory strength. To expose this decoupling, consider an artificial language where every word contains a copy of its first half, e.g., *foofoo*, *barbar*, *foobarfoobar*, etc. The first and second halves of these words have identical disambiguatory strength; they are the same so one could disambiguate the word as easily from its second half as from the first. In contrast, conditional surprisal would be nearly zero for the second halves of words because the second half is perfectly predictable from the first half.

In natural languages, measuring conditional entropy in a left-to-right fashion inherently forces a reduction of conditional entropy in later segments because of a language's phonotactic constraints. However, the disambiguatory strength of later segments is not inherently less than that of earlier segments. For instance, in a language like Turkish, which has vowel harmony, knowledge of any of the vowels in a word will provide information about the word's other vowels in a similar way. As such, knowledge of vowels towards the front of a word is as disambiguating as of vowels towards its end.

The contributions of this paper are threefold. First, we document and demonstrate the shortcomings of existing methods for measuring the informativeness of individual segments in context, including the confound with the amount of conditional information discussed above. Second, we introduce three surprisal-based measures that control for this confound and enable comparison of word-initial versus -final positions in this respect: unigram, position-specific and cloze surprisal (see §3). Finally, we find robust evidence across many languages of stronger disambiguatory signals in word initial than word-final positions. Out of a total of 151 languages analysed across three separate collections, 82 of them present a higher cloze surprisal in word beginnings than in endings—with similar patterns arising with the other two measures.

2 Background and Related Work

Psycholinguistic evidence. Lexical access has long been a topic of interest for psycholinguists, leading to many distinct models being proposed for this process (Morton, 1969; Marcus, 1981; Marslen-Wilson, 1987). Far earlier, though, Bagley (1900) had already demonstrated that earlier segments in words were more important for word recognition than later segments; specifically, they found that, when exposed to words with word-initial or word-final consonant deletions, listeners found the word-initial deletions more disruptive. Fay and Cutler (1977) showed mispronunciations are more likely in word endings, while Bruner and O'Dowd (1958) showed that recognizing written words with flipped initial characters was harder than with word final ones—demonstrating that the initial part of the word was more "useful" for readers. More recently, Wedel et al. (2019a) found evidence in support of Houlihan (1975), showing neutralizing rules tend to target word endings more significantly than beginnings in both suffixing and prefixing languages.

Nooteboom (1981) investigated the ease of recovering lexical items from either word beginnings or endings, finding that people had an easier time recovering words from their beginnings. For this, he examined words for which the first and second halves each completely identified them in a large Dutch dictionary—controlling for both segments' length and uniqueness. Later on, though, Nooteboom and van der Vlugt (1988) showed this difference vanishes when priming people with the length of the word—proposing the difference comes not from how informative segments were, but from the difficulty in time aligning later segments in mental lexicons. Connine et al. (1993) also found no difference in priming effects with non-words that differed from real words in either word initial or medial positions, suggesting initial positions have no special status in word recognition.

Psycholinguistic evidence is key to understanding how lexical access works in human language processing, and can help us understand why lexicons may evolve to provide more disambiguatory signals earlier in words.[4] Given the incremental nature of human lexical processing, however, such evidence cannot provide direct evidence of the nature of the lexicon uninfluenced by incrementality.

Computational evidence. To the best of our knowledge, van Son and Pols (2003b,a) were the first to use computational methods coupled with an

---

[4] Note that there are many possible reasons why the effects we demonstrate in this paper may arise, from the demands of lexical access to constraints on articulation. We provide no evidence for any of the possible explanations, evolutionary or otherwise, just methods for measuring the effect.

information theoretic definition of informativeness to investigate this question. They showed that segments in the beginning of words carry most of a word's information, as measured by their contextual surprisal using a plug-in tree structured probabilistic estimator. Although assessing a less-biased sample of words than Nooteboom (1981),[5] this study is also limited to a single language (Dutch), hence cannot assess whether this is a general phenomenon or specific to that language.

Further, van Son and Pols (2003a,b) use absolute word positions in their analysis. Word length correlates strongly with frequency, hence while early positions are present in all words, later positions only exist for a much smaller sample of typically lower frequency words. Thus this comparison amounts to asking if later positions in longer and infrequent words have lower surprisal than earlier positions in all (frequent or infrequent) words. We analyse this confounding factor in §6.

Wedel et al. (2019b) and King and Wedel (2020) applied a methodology similar to that of van Son and Pols (2003a) to show, for many diverse languages, that more frequent words contain less informative segments in word initial positions, while less frequent types carry more informative ones. They further showed that segments in later word positions were less informative (given the previous ones) than average in rarer words. While controlling for length, King and Wedel (2020) also compared words' forward and backward uniqueness points—nodes in a trie from which only one leaf node can be reached, i.e., where the word is uniquely identified—showing they happened earlier in forward strings.

While these studies provide evidence from more diverse sets of languages, they follow van Son and Pols (2003a) in studying closed lexicons.[6] As we show in §3.3.1, the use of probabilistic trie models on a closed lexicon yields a trivial effect of higher informativity at word initial positions. Furthermore, such studies cannot account for out-of-vocabulary words (e.g., nonce, proper name or otherwise unknown words) or derivational morphology, which are key parts of lexical recognition. Lexical access is also somewhat robust to segmental misordering (Toscano et al., 2013) and sounds later in a word help determine the perception of earlier ones (Gwilliams et al., 2018). In contrast, a trie over a closed lexicon is deterministic. Beyond this, Luce (1986) showed in a corpus study that the probability of a word type being uniquely identifiable before its last segment was only 41%—and 19% of types were identified only by the end of word, being proper prefixes of other words, such as *cat* and *cats*. They conclude that uniqueness point statistics may only be useful for long word analysis.

In Pimentel et al. (2020), we analysed several languages' phonotactic distributions, focusing on presenting a trade-off between phonotactic entropy and word length across languages. As a control experiment we analysed the correlation between a segment's surprisal and its word position across 106 languages. We did not control for word length and did not run per-language experiments, though—so we could have just been capturing the effect that later positions will mostly be present in languages with longer words (which, as we find, have lower information on average).[7]

While this last work avoids many of the issues raised earlier in this section, it fails to control the key confound mentioned earlier: it relies on left-to-right conditional probabilities to calculate surprisal. Thus segments early in the word have less conditional information and hence are generally of lower probability—a trivial effect that does not indicate a segment's disambiguatory signal strength.

3 Measures of Disambiguatory Strength

3.1 A Lexicon Generating Distribution

In this work, instead of the lexicon itself, we investigate the probability distribution from which it is sampled. The distribution is unobserved, but we can get glimpses of it via the sampled lexicon:

$$\left\{\mathbf{w}^{(n)}\right\}_{n=1}^{N} \sim p(\mathbf{w}) = \prod_{t=1}^{|\mathbf{w}|} p(w_t \mid \mathbf{w}_{<t}) \qquad (1)$$

The distribution $p(\mathbf{w})$ is defined over the entire space of possible phonological wordforms $\mathbf{w} \in \Sigma^*$, where Σ is a language-specific alphabet and the operator * indicates its Kleene closure.[8] This dis-

---

[5] Nooteboom (1981) looked at words completely identifiable by both their first and second halves in a large Dutch dictionary—this resulted in a study with only 14 words.

[6] The closed lexicon assumption is incorporated implicitly in the probabilistic trie models used by van Son and Pols (2003a,b) and King and Wedel (2020)—i.e. they assign zero probability to any form not in their training sets—and in the uniqueness point analysis of King and Wedel (2020).

[7] We note this issue only applies to the control experiment, and has no bearing on the key findings of that paper.

[8] We pad all strings with the end-of-word (EOW) symbol. For simplicity, we assume the alphabet includes EOW throughout the rest of the paper.

tribution should assign high probability to likely wordforms (attested or not) and low probability to unlikely ones. Using Chomsky and Halle's (1965) classic example from English, *brick* (attested) and *blick* (unattested) would have high probability, whereas **bnick* (unattested) would have a low probability.

3.2 Entropy and Conditional Entropy

Shannon's entropy is a measure of how much information a random variable contains. Consider a segment w_t at word position t, which is a value of the random variable W_t. The average information (surprisal) relayed per segment is:

$$\mathrm{H}(W_t) \equiv \sum_{w_t \in \Sigma} p(w_t) \log \frac{1}{p(w_t)} \tag{2}$$

A random variable is maximally entropic if it is a uniform distribution, in which case $\mathrm{H}(W_t) = \log(|\Sigma|)$. Conditional entropy measures how much information the knowledge of a variable conveys, given some previous knowledge. The average information transmitted per segment, given the previous ones in a word, is

$$\mathrm{H}(W_t \,|W_{<t}) \equiv \sum_{\mathbf{w}_{\leq t} \in \Sigma^*} p(\mathbf{w}_{\leq t}) \log \frac{1}{p(w_t \mid \mathbf{w}_{<t})} \tag{3}$$

where $\mathbf{w}_{\leq t} = \mathbf{w}_{<t} \circ w_t$. We note the conditional entropy is always smaller or equal to the entropy, i.e. $\mathrm{H}(W_t \mid W_{<t}) \leq \mathrm{H}(W_t)$.

3.3 Plug-in Estimators, Context Size, and Disambiguatory Strength

Our criticism of previous work investigating the disambiguatory strength of word-initial vs. word-final segments can be mainly divided in two parts: (i) the use of maximum likelihood plug-in estimators of the conditional entropy, by e.g. van Son and Pols (2003b); (ii) the use of left-to-right conditional entropy in itself, by all previous information-theoretic work in this vein.

3.3.1 A Critique of van Son and Pols (2003b)

We present a *reductio ad absurdum* which shows that van Son and Pols's (2003b) method will lead to the conclusion that word-initial segments are more informative even if all segments were equally entropic and sampled independently—a nonsensical finding. Accordingly, assume the probability distribution $p(w_t \mid \mathbf{w}_{<t})$, from which each segment in a word is sampled, was independent, e.g. define

$$\hat{p}(\mathbf{w}) = \prod_{t=1}^{|\mathbf{w}|} \hat{p}(w_t \mid \mathbf{w}_{<t}) = \prod_{t=1}^{|\mathbf{w}|} \hat{p}(w_t) \tag{4}$$

Assume now that a large, but finite, lexicon is sampled from it $\left\{\hat{\mathbf{w}}^{(n)}\right\}_{n=1}^{N} \sim \hat{p}(\mathbf{w})$. Further consider modelling this sampled lexicon with a probabilistic trie structure, similarly to what was done by van Son and Pols (2003a,b),[9] i.e.

$$q_{\text{trie}}(w_t \mid \mathbf{w}_{<t}) = \frac{\text{count}(w_t, w_{t-1}, \ldots, w_0)}{\text{count}(w_{t-1}, \ldots, w_0)} \tag{5}$$

where w_0 is the beginning-of-word symbol. Such a model uses all N words to approximate the distribution of the first segment—i.e. $\text{count}(w_0) = N$. Yet after $t-1$ segments, an exponentially smaller sample is used to capture the distribution—i.e. $\mathbb{E}[\text{count}(w_{t-1}, \ldots, w_0)] = N/|\Sigma|^{t-1}$. Using this model as a plug-in estimator of the entropy will lead to negatively biased estimates, where the error is approximately (Basharin, 1959):

$$\begin{aligned} \mathrm{H}(W_t \mid W_{t-1}) - \mathbb{E}\left[\hat{\mathrm{H}}\right] &\approx \frac{(|\Sigma| - 1) \log e}{\text{count}(w_{t-1}, \ldots, w_0)} \\ &\approx \frac{|\Sigma|^{t-1}(|\Sigma| - 1) \log e}{N} \end{aligned} \tag{6}$$

where $\hat{\mathrm{H}}$ is a plug-in estimate of the entropy. The error grows exponentially in t due to the $|\Sigma|^{t-1}$ factor. However, by assumption, $\mathrm{H}(W_t \mid W_{t-1})$ is constant—we have equally entropic and independent segments. Thus, the only way for this difference to increase is for the second term to decrease as a function of t. It follows that the estimated cross-entropies decrease as a function of t due to a methodological technicality. Indeed, in the extreme case, every position after a word's uniqueness point would be estimated to have zero entropy. Thus, van Son and Pols's (2003a) method only reveals a trivial effect.

3.3.2 Conditional Entropy and Context Size

As previously mentioned, the conditional entropy measures how much information the knowledge of a variable conveys, given some previous information, and it is always smaller or equal to the entropy. For this reason, relying on left-to-right conditional

---

[9] This is in fact a simplification of van Son and Pols's (2003a) model, which in practice uses Katz smoothing.

entropies to estimate the strength of disambiguatory signals yields straightforward results; the availability of larger conditioning contexts in a word's final segments will naturally reduce its conditional entropy. This will negatively skew the estimated informativeness of the later parts of a word.

$$\mathrm{H}(W_t) \geq \mathrm{H}(W_t \mid W_{t-1}) \geq \mathrm{H}(W_t \mid W_{<t}) \quad (7)$$

This effect can also be easily demonstrated by the symmetrical nature of mutual information (MI), where the MI is defined as:

$$\begin{aligned}\mathrm{MI}(W_t; W_{t-1}) &= \mathrm{H}(W_t) - \mathrm{H}(W_t \mid W_{t-1}) \\ &= \mathrm{H}(W_{t-1}) - \mathrm{H}(W_{t-1} \mid W_t) \\ &= \mathrm{MI}(W_{t-1}; W_t) \qquad (8)\end{aligned}$$

If we assume both segments had the same unconditional entropy, i.e. $\mathrm{H}(W_t) = \mathrm{H}(W_{t-1})$, then using left-to-right conditional entropies would suggest the later segment was less informative, while right-to-left conditioning would imply the opposite. Nonetheless, both their contextual and uncontextual disambiguatory strength would in fact be the same, if we estimated it with equal-sized contexts:

$$\begin{aligned}&\mathrm{H}(W_t) = \mathrm{H}(W_{t-1}) \Longrightarrow \qquad (9) \\ &\qquad \mathrm{H}(W_t \mid W_{t-1}) = \mathrm{H}(W_{t-1} \mid W_t)\end{aligned}$$

3.4 Cross-Entropy and Entropy

As mentioned above, the distribution $p(\mathbf{w})$ is not directly observable. We can, however, approximate it using character-level language models $p_\theta(\mathbf{w})$. We are interested in the entropy of variable W_t, as a proxy we measure its cross-entropy

$$\mathrm{H}_\theta(W_t) \equiv \sum_{w_t \in \Sigma} p(w_t) \underbrace{\log \frac{1}{p_\theta(w_t)}}_{\text{surprisal}} \quad (10)$$

where the surprisal is the information provided by a single segment instance w_t. The cross-entropy is an upper bound on the entropy, i.e. $\mathrm{H}(W_t) \leq \mathrm{H}_\theta(W_t)$, with their difference being the Kullback–Leibler (KL) divergence between both distributions. Since the KL-divergence is always positive, this upper-bound holds. Furthermore, the closer p_θ is to the true distribution p, the smaller the divergence is, and the tighter this bound. As such, the better our model is at estimating the true distribution, the better our estimates of the entropy will be.

Calculating eq. (10) still requires knowledge of the true p. We overcome this limitation by empirically estimating it on a held out part of the lexicon

$$\mathrm{H}_\theta(W_t) \approx \frac{1}{N} \sum_{n=1}^{N} \log \frac{1}{p_\theta(w_t^{(n)})} \quad (11)$$

3.5 Earlier vs. Later Word Entropy

For the remainder of this work, we will discuss information in terms of surprisal, since the entropy is its expected value. We analyse the distribution of disambiguatory information across word positions via three distinct measures—all of which control for the amount of conditioning per position:

- **Unigram Surprisal $\mathrm{H}_\theta(W_t)$:** the surprisal of individual segments.
- **Cloze Surprisal $\mathrm{H}_\theta(W_t \mid W_{\neq t})$:** surprisal of a segment given all others in the same word.
- **Position-Specific Surprisal** $\mathrm{H}_\theta(W_t \mid T = t, |W|)$: the surprisal of individual segments given their position in the wordform and the word's length.

The unigram surprisal captures the information provided by each segment when considering no context; while the cloze surprisal represents the information provided by a segment when one already knows the rest of the word. The position-specific surprisal represents a mid way between both, conditioning each segment only on its position and the word's length—being inspired by Nooteboom and van der Vlugt's (1988) experiments. These three measures of information control for the context size considered at each position, being thus better for an investigation of disambiguatory strength.

We used an unigram model (see §4) to estimate the unigram surprisal, and transformers (Vaswani et al., 2017) for cloze and position-specific surprisals. We also use the LSTM (Long-Short Term Memory, Hochreiter and Schmidhuber, 1997) model from Pimentel et al. (2020) for two other entropy measures which do not control for the amount of conditional information:

- **Forward Surprisal $\mathrm{H}_\theta(W_t \mid W_{<t})$:** the surprisal of a segment given the previous ones.
- **Backward Surprisal $\mathrm{H}_\theta(W_t \mid W_{>t})$:** the surprisal of a segment given the future ones.

We include the beginning- and end-of-word symbols in the forward and backward surprisal analysis, respectively, following previous work (Wedel et al., 2019b; Pimentel et al., 2020; King and Wedel, 2020). However, we ignore them in the unigram, position-specific and cloze surprisal analyses. Position-specific and cloze surprisal are given information about word length, hence these symbols are unambiguously predictable. We analyse the impact of these symbols in §6.

4 Character-Level Language Models

In this paper, we make use of character-level language models to model the probability distributions p_θ and approximate the relevant cross-entropies.

Unigram. This might be the simplest language model still in use in Natural Language Processing. We use its Laplace-smoothed variant

$$p_\theta(w_t) = \frac{\text{count}(w_t) + 1}{\sum_{c' \in \Sigma} \text{count}(c') + |\Sigma|} \tag{12}$$

LSTM. This architecture is the state-of-the-art for character-level language modelling (Melis et al., 2020). Given a sequence of segments $\mathbf{w} \in \Sigma^*$, we use one hot lookup embeddings to transform each of them into a vector $\mathbf{z}_t \in \mathbb{R}^d$. We then feed these vectors into a k-layer LSTM

$$\mathbf{h}_t = \text{LSTM}(\mathbf{z}_{t-1}, \mathbf{h}_{t-1}) \tag{13}$$

where $\mathbf{h} \in \mathbb{R}^d$, $\mathbf{h}_0$ is a vector with all zeros and w_0 is the beginning-of-word symbol. We then linearly transform these vectors before feeding them into a softmax non-linearity to obtain the distribution

$$p_\theta(w_t \mid \mathbf{w}_{<t}) = \text{softmax}(W\mathbf{h}_t + b) \tag{14}$$

in this equation, $W \in \mathbb{R}^{|\Sigma| \times d}$ is a weight matrix and $b \in \mathbb{R}^{|\Sigma|}$ a bias vector.

Backward LSTM. To get the backward surprisals we use models with the same architecture, but reverse all strings before feeding them to the models. As such, we get the similar equations

$$\mathbf{h}_t = \text{LSTM}(\mathbf{z}_{t+1}, \mathbf{h}_{t+1}) \tag{15}$$

$$p_\theta(w_t \mid \mathbf{w}_{>t}) = \text{softmax}(W\mathbf{h}_t + b) \tag{16}$$

Transformer. Transformers allow a segment to be conditioned on both future and previous symbols. Our implementation starts similar to the LSTM one, getting embedding vectors $\mathbf{z}_t$ for each segment in the string $\mathbf{w} \in \Sigma^*$, except that we replace segment w_t with a MASK symbol. We then feed these vectors through k multi-headed self-attention layers, as defined by Vaswani et al. (2017). Finally, the representations from the last layer are linearly transformed and fed into a softmax

$$p_\theta(w_t \mid \mathbf{w}_{\neq t}) = \text{softmax}(W\mathbf{h}_t + b) \tag{17}$$

Position-Specific Transformer. To get position-specific surprisal values, we again use a transformer architecture, but instead of replacing a single segment with a MASK symbol, we replace all of them. This is equivalent to conditioning each segment's distribution on its position and the word length—i.e., estimating $p_\theta(w_t \mid t, |\mathbf{w}|)$.

5 Data

In order to estimate redundancy and informativeness of segments we use three different datasets, each with its own pros and cons. We focus on types instead of tokens—i.e., the datasets consist of lexicons—for a few different reasons. First, it is easier to get reliable samples of types than tokens for a language, specially low-resource ones. Second, it is a well known result that token frequency correlates with both word length (Zipf, 1949) and phonotactic probability (Mahowald et al., 2018; Meylan and Griffiths, 2017), so that would be a strong confound in the results. Third, morphology is more easily modeled at the type level than at token level (Goldwater et al., 2011).[10]

CELEX (Baayen et al., 2015) allows us to experiment exclusively on monomorphemic words, but covers only three closely related languages. It contains both morphological and phonetic annotations for a large number of words in English, Dutch and German. We follow Dautriche et al. (2017) in using only words labeled as monomorphemic in our study, leaving us with 4,810 words in German, 6,206 words in English and 7,045 words in Dutch.

NorthEuraLex (Dellert et al., 2019) spans 107 languages from 21 language families in a unified IPA format. This database is composed of concept aligned word lists for these languages, containing 1016 concepts, each of them translated in most languages. However, most of these languages are from Eurasia, hence the collection lacks the typological diversity we would ideally like.

Wikipedia allows us to investigate a broader and more diverse set of languages, but has no phonetic information (only graphemes) and lexicons

---

[10] For each of the analysed datasets, we use 80% of the word types for training, with the rest being equally split between development and test sets; only test set surprisal and cross-entropies are used in our analysis.

extracted from it may be “contaminated” with foreign words. We fetch the Wikipedia for a set of 41 diverse languages,[11] and tokenise their text using language-specific tokenisers from spaCy (Honnibal and Montani, 2017). When a language-specific tokeniser was not available, we used a multilingual one. We then filtered all non-word tokens—by removing the ones with any symbol not in the language’s scripts—and kept only the 10,000 most frequent types in each language.

6 Experiments and Results

Forward Surprisal. We first replicate the results from van Son and Pols (2003a,b), Wedel et al. (2019b), and Pimentel et al. (2020), which show that surprisal decreases with as the words position advances. On average, forward surprisal, i.e. $\mathrm{H}_\theta(W_t \mid W_{<t})$, could decrease for two reasons: (i) words indeed front-load disambiguatory signals; or (ii) the trivial fact that conditioning reduces entropy. For each word, we first get the forward surprisal for each segment in it. We then group surprisal values in two groups: word initial (when they are in the first half of its word) and final (when in the second half), ignoring mid positions in words with uneven lengths; we average these initial and final surprisals per word, getting a single value of each per word. This way we compare earlier vs. later word positions while ignoring any length effect—words with all lengths will possess segments in both groups. For each analysed language, we then use permutation tests (permuting word initial and final surprisals) to evaluate if one group is statistically larger than the other—using 100,000 permutations. All but one language in three analysed datasets had significantly larger surprisal in word initial positions[12]—the exception being Abkhaz in NorthEuraLex. These results can be seen in Tab. 1 and in Fig. 2 (left).

Backward Surprisal. If the result for forward surprisal is largely due to the amount of conditional information, then reversing the strings should lead to a roughly opposite effect. With this in mind, for each language, we again bin surprisals in word initial vs. final position, but now we evaluate languages using backward surprisal, i.e., $\mathrm{H}_\theta(W_t \mid W_{>t})$.[13] When using backward surprisal, many of the analysed languages have significantly higher surprisals in word final positions (see Tab. 1 and the right graph in Fig. 2). However, 11 languages in the NorthEuraLex dataset still have higher word initial surprisals, suggesting that initial positions in these languages are indeed largely more informative than final ones.[14] There does seem to be a large effect of the amount of conditional information and also some lexical effect of front-loading disambiguatory signals, however it is difficult to determine if there are cross-linguistic tendencies with these measures.

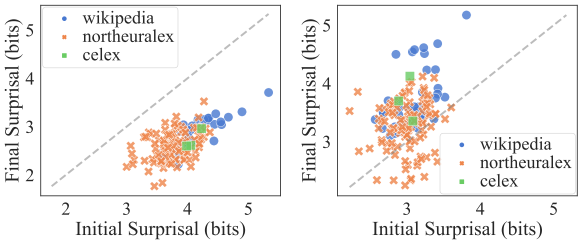


Figure 2: Word initial vs. final surprisals with: (left) Forward; (right) Backward.

Unigram Surprisal. To control for the conditioning aspect of the question: *do words front-load their disambiguatory signals?*, we can look at unigram surprisal $\mathrm{H}_\theta(W_t)$. This value tells us how uncommon the segments that appear in a certain position are, when analysed in isolation from the rest of the word—uncommon segments are more informative and provide stronger signal for disambiguation. In NorthEuraLex, 71 of the languages have significantly higher informativity in word beginnings than in endings—nonetheless, one language (Kildin Saami) has higher surprisals in word endings. In CELEX, Dutch and German have higher surprisals in initial positions, but English does not. And in Wikipedia, all languages but Hebrew and Bengali have higher surprisal in initial positions—with Bengali having higher surprisal in word endings. This experiment suggests that indeed most languages are biased towards providing stronger disambiguatory signals in word beginnings, even when we control for the

---

[11] These languages were: af, ak, ar, bg, bn, chr, de, el, en, es, et, eu, fa, fi, ga, gn, haw, he, hi, hu, id, is, it, kn, lt, mr, no, nv, pl, pt, ru, sn, sw, ta, te, th, tl, tr, tt, ur, zu.

[12] All statistical significance results in this work have been corrected for multiple tests with Benjamini and Hochberg (1995) corrections and use a confidence value of $p < 0.01$.

[13] We note that King and Wedel (2020) also used backward surprisal, although with a different objective in mind. In one of their experiments, they presented aggregate results of a comparison between the forward and backward surprisal.

[14] We also ran the same experiments with a probabilistic trie model like the ones used in van Son and Pols (2003b) and Wedel et al. (2019b), which showed an even stronger result reversal when using backward surprisal.

<table>
<tr><th></th><th></th><th colspan="5">Surprisal</th></tr>
<tr><th>Dataset</th><th># Languages</th><th>Forward</th><th>Backward</th><th>Unigram</th><th>Position-Specific</th><th>Cloze</th></tr>
<tr><td>CELEX</td><td>3</td><td>3 | 0</td><td>0 | 3</td><td>2 | 0</td><td>2 | 1</td><td>2 | 1</td></tr>
<tr><td>NorthEuraLex</td><td>107</td><td>106 | 0</td><td>11 | 31</td><td>71 | 1</td><td>24 | 4</td><td>45 | 1</td></tr>
<tr><td>Wikipedia</td><td>41</td><td>41 | 0</td><td>0 | 39</td><td>39 | 1</td><td>31 | 1</td><td>35 | 2</td></tr>
</table>

Table 1: Number of languages in the analysed datasets with significantly larger surprisals in initial | final positions.

amount of conditional information. Nonetheless, this is not a universal characteristic which all languages share and two analysed languages even had a statistically significant inverse effect.

Position-Specific Surprisal. While cloze surprisal makes explicit the non-redundant informativity a segment conveys, unigram surprisal analyses the same segments in isolation. Position-specific surprisal provides a midway analysis, incorporating the position as some previously-specified knowledge, but not conditioning on the other segments in the word. The position-specific surprisal is inspired by Nooteboom and van der Vlugt (1988) experiments, which prime individuals on word length and position. As can be seen in Tab. 1, position-specific surprisal again seems to favour initial positions over final, but only slightly. Interestingly, most languages present no significant difference and some the inverse effect (i.e. higher surprisal in final positions).

Position-specific Unigram models. To better understand the differences between the unigram and position-specific surprisal results, we trained position-specific unigram models—which count each segment's frequency per position—and then calculated their Kullback–Leibler (KL) divergence per position with the traditional unigram

$$\mathrm{KL}(p(w_t \mid t) \mid\mid p(w_t)) \qquad (18)$$
$$= \sum_{w_t \in \Sigma} p(w_t \mid t) \log \frac{p(w_t \mid t)}{p(w_t)}$$

We compare these KL divergences and find that, for all but four languages, the KL is largest in either the first or second segment positions.[15] This suggests that one of the reasons for higher unigram surprisal in initial positions is that the first two segments usually differ from the rest of the positions, potentially serving as markers for word segmentation.

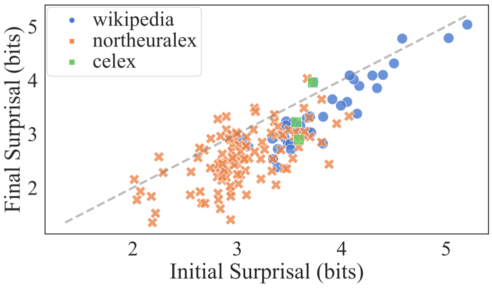


Figure 3: Word initial vs. final cloze surprisals.

Cloze Surprisal. When we condition a segment on all others in the same word, we measure how much uncertainty is left about that individual segment when considering everything else, or, in other words, how much information is passed only by that segment non-redundantly. Word initial surprisal is higher in most analysed languages (see Tab. 1). Nonetheless, two languages in Wikipedia, Thai and Bengali, have significantly higher surprisal in their final segments—while English in CELEX and Hungarian in NorthEuraLex also present this same inverse effect. Front-loading disambiguatory information, thus, is not established to be the linguistic universal it is believed to be, with only roughly half the analysed languages showing this property when we control for morphology (CELEX and NorthEuraLex). Fig. 3 plots the results for all languages analysed.

When we compare these results, we find an interesting pattern. Morphology seems to reduce non-redundant (cloze) information later in the words—while only half of the languages had significant surprisals in CELEX (which consists of monomorphemic words) and NorthEuraLex (base forms), most languages were significant in Wikipedia. Furthermore, English and Hungarian had significantly higher surprisals in word endings in CELEX and NorthEuraLex, while the opposite trend in Wikipedia—this is consistent with the fact that suffix morphemes are present in more types than word roots are, so morphology would make word endings less surprising.

---

[15] We use Laplacian smoothing in the position-specific unigrams and constrain the analysis to positions which appear in at least 75% of the analysed words in that language.

	EOW	Non-EOW
Forward	1.14	3.55
Backward	0.89	3.61
Unigram	2.75	4.90
Position-specific	0.00	4.36
Cloze	0.00	3.23

Table 2: Average surprisal (in bits) of EOW vs. non-EOW segments averaged over all datasets.

Length as a Confounding Effect. We evaluate the impact of length as a confounding effect on previous methodologies. As mentioned in §2, by directly analysing surprisal–position pairs (as opposed to binning word initial vs. final positions), previous work confounds position and word length—i.e., only long words will have later word positions. In this study, we analyse forward surprisal–length pairs; instead of pairing a segment's surprisal with its position, we pair it with its word length. We then get the slope formed by a linear regression between these pairs of values and test for its significance per language by using a permutation test, in which we shuffle surprisal–length values. On the three datasets, all languages have statistically significant negative slopes, meaning long words have smaller surprisals on average than shorter ones.[16] A caveat, though, is that now we are confounding position into our length analysis. Constraining our analysis only to the first two segments in each word, we still find the same effect—though now one language (Hebrew) in Wikipedia and seven in NorthEuraLex are not significant. We can thus conclude that longer words have smaller surprisal values than shorter ones, even when controlling for the same word positions. This implies that directly using surprisal–position pairs for such an analysis is not ideal.

The Effect of End of Word in Surprisal. The end-of-word (EOW) symbol is a special "segment" which symbolises the end of a string. It is necessary when modelling the probability distribution over strings $\mathbf{w} \in \Sigma^*$, to guarantee that the overall distribution sums to 1. Nonetheless, it is expected to behave in a different way from other segments. If a speaker wants to reduce their production effort, although changing from one phone to another may help, the most efficient way is usually just ending the string earlier. Furthermore, since all realisable strings must eventually end, it will be present in all words, making it a very frequent symbol—in fact, Tab. 2 shows its average surprisal is much lower than that of other segments. As such, it is only natural it should be analysed on its own, separately from other segments. Through the same logic, other segments should also be analysed separately from EOW—or else, lower word final surprisals may be due to this symbol alone. As such, we analyse the surprisal of LSTM "language models" without the EOW symbol here.[17]

	EOW			No EOW		
	Initial	Final	Diff (%)	Initial	Final	Diff (%)
Forward	3.85	2.65	31.1 %	3.83	3.00	21.6 %
Backward	3.02	3.40	-11.3 %	3.63	3.39	6.7 %
Unigram	-	-	-	4.85	4.40	9.3 %
Position	-	-	-	4.36	4.17	4.3 %
Cloze	-	-	-	3.26	2.81	13.9 %

Table 3: Average surprisal per segment in word initial and final positions with and without EOW symbols.

Unsurprisingly, Tab. 3 shows the difference between word initial and final positions is considerably reduced when we remove the EOW symbol from the forward surprisal analysis. Surprisingly, we see that when we remove the beginning-of-word from the backward surprisal analysis, instead of a larger word final surprisal, we get a larger word initial value—even though we are still conditioning the models right-to-left. This result further supports the hypothesis that the disambiguatory signals are on average stronger in word initial positions.

7 Conclusions

In this work, we analysed the distribution of disambiguatory information in word positions. We present an in-depth critique of previous work, showing several confounding effects in their analysis. We then proposed the use of three new methods which corrected for these biases—namely unigram, position-specific and cloze surprisal. These models controlled for the amount of conditional information across word positions, allowing for an unbiased analysis of the lexicon. Using these models we show that the lexicons of most languages indeed front-load their disambiguatory signals. This effect, though, is not universal and the difference in disambiguatory information between word initial and final positions is much lower than previously estimated—ranging from 4% to 14%, depending on the used metric, instead of 31%.

---

[16] King and Wedel (2020) indeed present a similar correlation in their Figure 2.

[17] To be more precise, we actually ignore the beginning-of-word symbol when estimating backward surprisal.

References

R. H. Baayen, R. Piepenbrock, and L. Gulikers. 2015. CELEX2 LDC96L14.

William Chandler Bagley. 1900. The apperception of the spoken sentence: A study in the psychology of language. *The American Journal of Psychology*, 12(1):80–130.

Georgij P. Basharin. 1959. On a statistical estimate for the entropy of a sequence of independent random variables. *Theory of Probability & Its Applications*, 4(3):333–336.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300.

Jerome S. Bruner and Donald O'Dowd. 1958. A note on the informativeness of parts of words. *Language and Speech*, 1(2):98–101.

Noam Chomsky and Morris Halle. 1965. Some controversial questions in phonological theory. *Journal of Linguistics*, 1(2):97–138.

Cynthia M. Connine, Dawn G. Blasko, and Debra Titone. 1993. Do the beginnings of spoken words have a special status in auditory word recognition? *Journal of Memory and Language*, 32(2):193–210.

Isabelle Dautriche, Kyle Mahowald, Edward Gibson, Anne Christophe, and Steven T. Piantadosi. 2017. Words cluster phonetically beyond phonotactic regularities. *Cognition*, 163:128–145.

Johannes Dellert, Thora Daneyko, Alla Münch, Alina Ladygina, Armin Buch, Natalie Clarius, Ilja Grigorjew, Mohamed Balabel, Hizniye Isabella Boga, Zalina Baysarova, et al. 2019. NorthEuraLex: A wide-coverage lexical database of Northern Eurasia. *Language Resources and Evaluation*, pages 1–29.

David Fay and Anne Cutler. 1977. Malapropisms and the structure of the mental lexicon. *Linguistic Inquiry*, 8(3):505–520.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2011. Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12(Jul):2335–2382.

Laura Gwilliams, Tal Linzen, David Poeppel, and Alec Marantz. 2018. In spoken word recognition, the future predicts the past. *Journal of Neuroscience*, 38(35):7585–7599.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Kathleen Houlihan. 1975. *The Role of Word Boundary in Phonological Processes*. Ph.D. thesis, University of Texas at Austin.

Adam King and Andrew Wedel. 2020. Greater early disambiguating information for less-probable words: The lexicon is shaped by incremental processing. *Open Mind*, pages 1–12.

Paul A. Luce. 1986. A computational analysis of uniqueness points in auditory word recognition. *Perception & Psychophysics*, 39(3):155–158.

Kyle Mahowald, Isabelle Dautriche, Edward Gibson, and Steven T. Piantadosi. 2018. Word forms are structured for efficient use. *Cognitive Science*, 42(8):3116–3134.

Stephen Michael Marcus. 1981. ERIS-context sensitive coding in speech perception. *Journal of Phonetics*, 9(2):197–220.

William D. Marslen-Wilson. 1987. Functional parallelism in spoken word-recognition. *Cognition*, 25(1-2):71–102.

Gábor Melis, Tomáš Kočiský, and Phil Blunsom. 2020. Mogrifier LSTM. In *International Conference on Learning Representations*.

Stephan C. Meylan and Thomas L. Griffiths. 2017. Word forms—not just their lengths—are optimized for efficient communication. *arXiv preprint arXiv:1703.01694*.

John Morton. 1969. Interaction of information in word recognition. *Psychological Review*, 76(2):165.

S. G. Nooteboom and M. J. van der Vlugt. 1988. A search for a word-beginning superiority effect. *The Journal of the Acoustical Society of America*, 84(6):2018–2032.

Sieb G. Nooteboom. 1981. Lexical retrieval from fragments of spoken words: Beginnings vs endings. *Journal of Phonetics*, 9(4):407–424.

Tiago Pimentel, Brian Roark, and Ryan Cotterell. 2020. Phonotactic complexity and its trade-offs. *Transactions of the Association for Computational Linguistics*, 8:1–18.

Rob J. J. H. van Son and Louis C. W. Pols. 2003a. Information structure and efficiency in speech production. In *Eighth European Conference on Speech Communication and Technology*.

Rob J. J. H. van Son and Louis C.W. Pols. 2003b. How efficient is speech? In *Proceedings of the Institute of Phonetic Sciences*, volume 25, pages 171–184.

Joseph C. Toscano, Nathaniel D. Anderson, and Bob McMurray. 2013. Reconsidering the role of temporal order in spoken word recognition. *Psychonomic Bulletin & Review*, 20(5):981–987.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Andrew Wedel, Adam Ussishkin, and Adam King. 2019a. Crosslinguistic evidence for a strong statistical universal: Phonological neutralization targets word-ends over beginnings. *Language*, 95(4):e428–e446.

Andrew Wedel, Adam Ussishkin, and Adam King. 2019b. Incremental word processing influences the evolution of phonotactic patterns. *Folia Linguistica*, 40(1):231–248.

George Kingsley Zipf. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley Press.

On the (In)Effectiveness of Images for Text Classification

Chunpeng Ma[1], **Aili Shen**[2], **Hiyori Yoshikawa**[1],
Tomoya Iwakura[1], **Daniel Beck**[2], **Timothy Baldwin**[2]
[1] Fujitsu Laboratories LTD., Japan
[2] The University of Melbourne, Australia
{ma.chunpeng, y.hiyori, iwakura.tomoya}@fujitsu.com
ailis@student.unimelb.edu.au, {beck.d, tbaldwin}@unimelb.edu.au

Abstract

Images are core components of multi-modal learning in natural language processing (NLP), and results have varied substantially as to whether images improve NLP tasks or not. One confounding effect has been that previous NLP research has generally focused on sophisticated tasks (in varying settings), generally applied to English only. We focus on text classification, in the context of assigning named entity classes to a given Wikipedia page, where images generally complement the text and the Wikipedia page can be in one of a number of different languages. Our experiments across a range of languages show that images complement NLP models (including BERT) trained without external pre-training, but when combined with BERT models pre-trained on large-scale external data, images contribute nothing.

1 Introduction

Combining data from multiple modalities (e.g., text, images, categorical metadata, or user interaction features) has become commonplace in artificial intelligence. In NLP, examples include multi-modal machine translation (MMT) (Elliott et al., 2016; Elliott, 2018), visual question answering (VQA) (Goyal et al., 2017; Johnson et al., 2017), visual commonsense reasoning (VCR) (Zellers et al., 2019; Geva et al., 2019), and multi-modal pre-training (Lu et al., 2019; Chen et al., 2019).

While tasks such as VQA and VCR are multi-modal in nature, there has been research on traditionally text-based tasks such as text classification (Shen et al., 2020; Huang, 2018) and word embedding learning (Bruni et al., 2014) which has demonstrated that the addition of images boosts performance. At the same time, however, there is evidence of images providing no additional information, e.g. Caglayan et al. (2019) show that MMT models learn to ignore visual content when trained on a parallel corpus of image captions (Elliott et al., 2016). These mixed findings raise the question of *when* visual context is actually useful in NLP.

In this work, we take a first step towards answering this question, in focusing on the task of text classification, which has traditionally been addressed using textual data only. We identify two gaps in the literature on multi-modal NLP: (1) no results for pre-trained language models (LMs); and (2) no results for languages other than English. The first is important in terms of updating the research relative to state-of-the-art approaches, while the second relates to the question of how "language-independent" systems actually are (Bender, 2011). We fill these gaps via a text classification task over Wikipedia articles (Sekine et al., 2019). Our main findings are: (1) while images do help in a traditional supervised learning setting, *their utility disappears almost completely* when combined with a pre-trained LM; and (2) this phenomenon *is not restricted to English*, and generalises across a variety of languages from different families.

2 Task Description

This research is couched in the context of a shared-task dataset released by the SHINRA project (Sekine et al., 2019), aimed at classifying Wikipedia pages into fine-grained entity classes.[1] We chose this benchmark as many Wikipedia documents contain images, and data is provided for a total of 29 typologically-diverse languages.[2] The task is not trivial as it involves classifying Wikipedia documents into a set of 219 classes, with the possibility of multiple labels for a given document.[3]

---

[1] `http://shinra-project.info/shinra2020ml/?lang=en`

[2] Data is also provided for Greek but we do not include it in our experiments because there was no officially preprocessed data available for this language.

[3] See: `http://ene-project.info/ene8/?lang=en`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 42–48
April 19 - 23, 2021.

hi	th	ar	da	bg	ro	he	tr	id	vi
30,546	59,790	73,053	86,237	89,016	92,001	96,433	111,591	115,642	116,279
hu	cs	no	ca	fi	uk	fa	sv	ko	nl
120,294	125,958	135,934	139,031	144,749	167,236	169,052	180,947	190,806	199,982
pt	pl	ru	es	zh	it	de	fr	en	
217,895	225,551	253,011	257,834	267,106	270,192	274,731	318,827	439,351	

Table 1: Statistics of annotated data for each language.

The number of annotated pages for each language in the SHINRA dataset is shown in Table 1 (sorted according to the number of pages). In addition to these annotated datasets — which form the basis of the experiments in this paper — there is a large amount of evaluation data for each language. In an evaluation campaign over these evaluation datasets, we achieved first place across 4 languages: English, Italian, Spanish and Catalan (Yoshikawa et al., 2020).

The SHINRA dataset contains only textual information from the original documents. In order to add images, we extract the image links from the English Wikipedia dump of June 2020[4] using the `zim` library.[5] The extracted images are then linked with image links in the source documents in the SHINRA dataset,[6] resulting in about 88% pages being augmented with images (noting that images are generally shared across Wikipedia pages for different languages other than English).

Out of the 30 languages in the original SHINRA dataset, we experiment primarily with Arabic ("ar"), English ("en"), Finnish ("fi"), Hindi ("hi"), and Mandarin Chinese ("zh"), selected to span five different language families and where the dataset size is relatively large. From the SHINRA data, we randomly sample 30k documents for each language, and construct a 80%/10%/10% fixed split for training/development/test in each language. We use a maximum of four images for each document.[7]

3 Baseline Experiments

Our first set of experiments is aimed at evaluating the empirical utility of images in the absence of pre-trained models. This is in line with previous work over similar text classification tasks (Shen et al., 2020; Huang, 2018).

Model and Features As our basic learner, we use a linear-kernel support vector machine (Cortes and Vapnik, 1995, SVM). For the textual inputs, we experiment with three representations: (1) a binary bag-of-words ("`BOW`"); (2) sent2vec ("`S2V`": Pagliardini et al. (2018)); and (3) `BERT` (Devlin et al., 2019). In this set of experiments, we train both `S2V` and `BERT` from scratch on the SHINRA training data only. We simply use the suggested configuration provided by developers, without any task-specific hyperparameter tuning. For `BERT`, we use the `[CLS]` token as the document representation. For each document, an image representation for each of the (up to) four images is generated. Specifically, following standard practice in the computer vision community, we firstly use the SIFT algorithm (Lowe, 1999) to extract hundreds of features, then use the K-means algorithm to cluster these features and generate frequency histograms, which are so-called visual bag-of-words (VBoW), and finally use an SVM to classify these histogram features. We also experiment with Faster R-CNN (Ren et al., 2015), pre-trained on Visual Genome (Krishna et al., 2017), following the settings of Anderson et al. (2018). We ensure the dimensionality of input features for the SVM and Faster R-CNN are the same (both are $1,024$), to remove this possible representational confound. Note that this is the externally pre-trained image model across all experiments, and that none of the text models in this first set of experiments involve pre-training on external resources (something we return to in Section 4).

Results and Analysis We report F_1 scores over the test set in Table 2. The main finding is that images improve performance in all settings, for all languages and both image representations. `S2V` and `BERT` both perform worse than the simple bag of words, because of the limited training data in each case. We would, of course, expect the models

---

[4] https://dumps.wikimedia.org/other/kiwix/zim/wikipedia/wikipedia_en_all_maxi_2020-06.zim

[5] https://github.com/openzim/libzim

[6] Because it is quite difficult to find correspondences between images and texts (Hessel et al., 2019), image links extracted are "document-level", instead of "sentence-level".

[7] When a document has less than 4 images, we pad the representation with blank images.

Text	Image	Language				
		ar	en	fi	hi	zh
`BOW`	—	65.1	72.3	72.1	67.1	74.7
`BOW`	SIFT+V	**68.2**	**74.1**	**73.2**	**69.0**	**76.0**
`BOW`	R-CNN	67.1	73.0	72.7	68.7	75.3
`S2V`	—	63.6	68.1	63.1	63.1	72.0
`S2V`	SIFT+V	66.0	70.2	66.3	66.9	72.9
`S2V`	R-CNN	65.4	69.0	65.0	65.2	72.3
`BERT`	—	59.1	65.3	51.9	60.5	68.6
`BERT`	SIFT+V	62.9	68.7	54.2	63.1	70.9
`BERT`	R-CNN	61.4	67.3	52.7	62.9	70.0

Table 2: F_1 score of the SVM models without external pre-training of the textual models, across the five languages. "SIFT+V" refers to the combination of SIFT and Visual Bag-of-Words features. "R-CNN" corresponds to features extracted from Faster R-CNN.

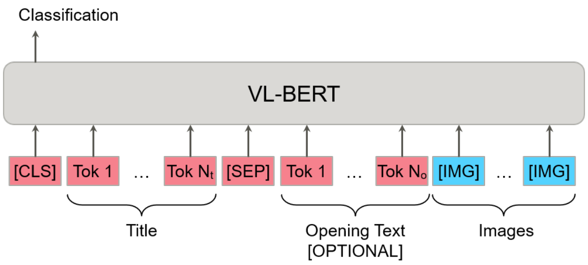


Figure 1: `VL-BERT` architecture applied to the SHINRA2020-ML task. The "opening text" segment are additional textual data obtained from the documents that are optional in our experimental setting.

to perform better with more extensive pre-training, as we return to explore in Section 4, but the focus here is on training of the textual models within the bounds of the training dataset.

Strikingly, the SIFT + Visual Bag-of-Words representation results in better performance than the pre-trained Faster R-CNN. A potential explanation is that Faster R-CNN is trained in a supervised way using Visual Genome (unlike the self-supervised setting of pre-trained BERT, for instance), over a set of labels that is not particularly well aligned with SHINRA (SHINRA includes many abstract classes such as RELIGION, NATIONALITY, and OFFENCE, whereas Visual Genome is focused on physical objects and attributes, and relations between objects; even among physical objects, SHINRA distinguishes between MEDICAL INSTITUTION, PUBLIC INSTITUTION, and RESEARCH INSTITUTE, most of which are represented simply as BUILDING in Visual Genome).

4 Adding a Pre-trained Textual Encoder

We next turn to a setting where we employ pre-trained textual models. This not only better reflects the state-of-the-art in text classification, but also allows us to investigate the effect of images under such conditions.

Model As the main backbone, we employ `VL-BERT` (Su et al., 2020), which uses a transformer to combine textual inputs and image embeddings within a BERT-style transformer, and has been shown to perform well on multimodal tasks. The visual embeddings are obtained from the combination of pre-trained Faster R-CNN and ResNet-101 (He et al., 2016), as illustrated in Figure 1. For the text modality, the input consists of two parts: the document title, and the opening text of the Wikipedia page in the form of the first 300 tokens. The token embeddings are obtained from a pre-trained BERT model, which is fine-tuned during training.[8] The full model is plugged into a one-layer feed-forward neural network (FFNN) with a 1,024d hidden layer, and training is performed by minimizing the cross-entropy over the SHINRA category labels. The model predict one label for each page. For the case of multi-label inputs, we choose one randomly as the "correct" label.

Results and Analysis Table 3 shows the performance of `VL-BERT` with different combinations of textual (document title = "T" and optionally the document body = "B") and image inputs, based on pre-trained BERT ("$\texttt{BERT}_{\text{pre}}$").

The first thing to notice is that the image-only model is well above the majority baseline, but well below the best multimodal model without an externally pre-trained text encoder from Table 2. This shows that images provide useful information for document classification, consistent with the earlier finding that images enhance the various text-only models. However, when combined with the externally pre-trained $\texttt{BERT}_{\text{pre}}$ (over either the title only, or the title + document body), the utility of images is marginal at best. That is, the large-scale pre-training of $\texttt{BERT}_{\text{pre}}$ both boosts overall performance, but much more surprisingly, removes any advantage from including images.

---

[8] We use `bert-large-uncased` for English, and `bert-base-multilingual-uncased` for the other languages, as obtained from `https://huggingface.co/transformers/pretrained_models.html`

Text	Image	ar	en	fi	hi	zh
—	✓	50.6	50.1	53.9	46.1	44.1
T	—	70.9	73.1	71.1	66.2	76.5
T	✓	70.8	73.2	71.2	66.7	76.7
T+B	—	**82.8**	88.7	87.7	**85.0**	**88.6**
T+B	✓	82.6	**88.8**	**88.0**	84.8	88.0
Best non pre-trained		68.2	74.1	73.2	69.0	76.0
Majority class		21.5	22.2	28.1	19.1	21.7

Table 3: F_1 scores for pre-trained `VL-BERT`. "T" = document title, and "T+B" = document title + body. We reproduce the best non-trained For comparison, we restate the result for the best non pre-trained model from Table 2, along with the majority class baseline.

Influence of the size of training data One hypothesis is that images are not useful due to the size of the training data (24k instances), and in lower-resource scenarios will improve performance. To test this, we perform additional experiments varying the training data size, ranging from 4k to 24k training instances, in steps of 4k.

Figure 2 plots the F_1 performance as the training set size increases. While we observe substantial improvements for the image-only approach (the bottom curve), the differences in the models with textual data are modest, and even in small-data settings, there is no real advantage in including images. We also separated the test data in terms of the number of images, and found no differences. See the Supplementary Material for details.

Results on the full SHINRA dataset In the previous experiments, we fixed the dataset size for all languages to control for training data volume. However, the SHINRA dataset includes many more documents for many of the languages. As a final experiment, we apply the `VL-BERT` models to the full dataset available for each language. The development and test data are also different in this configuration, so the results are not directly comparable with Tables 2 and 3.

In Table 4, we present results for $\texttt{BERT}_{\text{pre}}$, and mostly corroborate our earlier findings: while we do see improvements when including images in the case of the titles only, their utility decreases when we add the body of text for each document.

What caused the difference? Comparing the results from Sections 3 and 4, we see two main differences: the presence of external pre-training (`BERT` vs. $\texttt{BERT}_{\text{pre}}$), and the model architecture. To determine whether the model architecture is a cause of the performance difference, we train `VL-BERT` *from scratch*, using only text and images from the 24k training set used in Section 3.

Text	Image	ar	en	fi	hi	zh
—	✓	85.2	61.2	72.4	45.6	58.6
T	—	84.4	77.4	75.5	66.6	78.5
T	✓	86.9	79.1	75.8	67.3	80.7
T+B	—	94.7	90.3	91.7	85.8	89.8
T+B	✓	94.7	90.2	91.6	85.4	90.2

Table 4: Comparison of F_1 scores over the full SHINRA dataset for $\texttt{BERT}_{\text{pre}}$.

Text	Image	ar	en	fi	hi	zh
T+B	—	56.7	61.8	49.8	57.2	66.1
T+B	✓	58.6	63.2	52.4	60.1	68.7

Table 5: Comparison of F_1 scores for `VL-BERT` without external pre-training of BERT.

The results in Table 5 shows that even for `VL-BERT`, a neural-based model that is much more complex than the linear-kernel SVM, when $\texttt{BERT}_{\text{pre}}$ is not used, images provide a gain in performance. Hence, having an externally pre-trained text encoder is the predominant determinant of whether visual content has utility in NLP tasks.

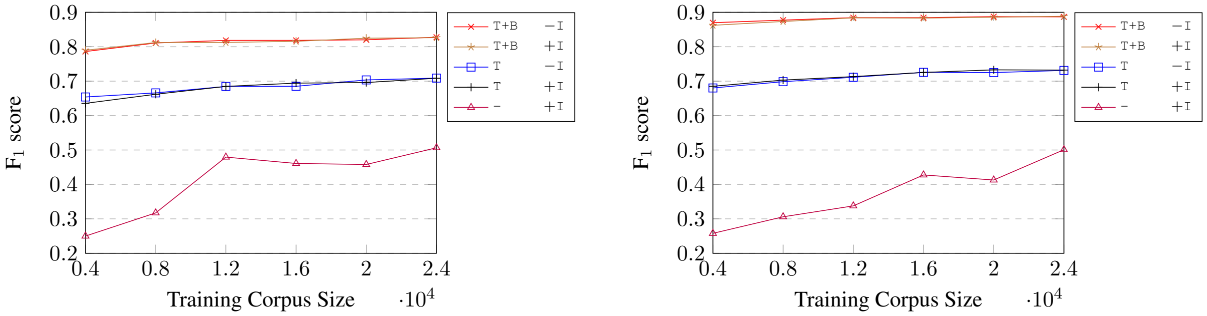
5 Discussion and Conclusion

We investigated the utility of images as a supplementary input for a text classification task, and found that although images have empirical utility in traditional supervised learning, when externally pre-trained language models are utilised, any advantage from the visual modality disappears. The results were remarkably consistent across different languages and different volumes of training data.

It is important to distinguish between "inherently multi-modal tasks" (e.g. VQA) and "potentially multi-modal tasks" (e.g. text classification) in drawing any broader conclusions about the (in)effectiveness of images. Here, a "potentially multi-modal task" in NLP means that the primary modality is text and the task is defined based on that single data modality, but there is potentially the option to include extra modalities such as images.

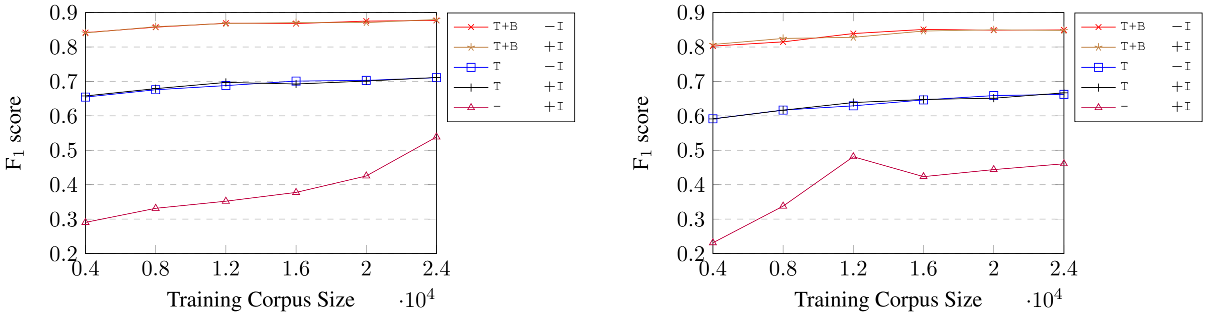
There remain a lot of open questions in more fully determining the (in)effectiveness of images for NLP tasks, even for text classification, such as:

- Due to the seeming redundancy between textual and visual representations of Wikipedia pages, is there any utility in multi-modal inputs for simple NLP tasks such as text clas-



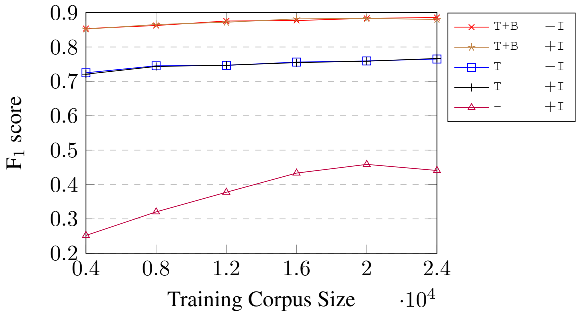
(a) Arabic

(b) English



(c) Finnish

(d) Hindi



(e) Chinese

Figure 2: Model performance with different sizes of training corpus, with and without images (±I) and with and without text (in the form of the title ["T"] and optionally document body ["B"]).

sification in the era of large-scale pre-trained language models such as BERT and GPT-3 (Brown et al., 2020)?

- What performances do humans achieve in the single-modal setting and multi-modal setting? Can we get some insights by comparing the (potentially) different performances between humans and computers?
- Apart from images, what other modalities and forms of input (e.g. audio) could be effective in building better NLP models?
- Although pre-trained image models (e.g. Faster R-CNN) contribute a lot for vision tasks (e.g. object detection) and multi-modal tasks (e.g. VQA), for *"pure"* NLP tasks (e.g. text classification), they appear to work no better than traditional image representation feature extractors (e.g. SIFT). Why?
- In our experiments, we use at most 4 images for each page. Could instance selection enhance image utility?
- We focused on the text classification task, in classifying Wikipedia pages into different entities. Are our observations NLP task-independent?

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Emily M. Bender. 2011. On achieving and evaluating language-independence in NLP. *Linguistic Issues in Language Technology*, 6.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Ozan Caglayan, Pranava Madhyastha, Lucia Specia, and Loïc Barrault. 2019. Probing the need for visual context in multimodal machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4159–4170.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

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

Desmond Elliott. 2018. Adversarial evaluation of multimodal machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2974–2978.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30k: Multilingual english-german image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Jack Hessel, Lillian Lee, and David Mimno. 2019. Unsupervised discovery of multimodal links in multi-image, multi-sentence documents. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2034–2045.

Jia-Bin Huang. 2018. Deep paper gestalt. *arXiv preprint arXiv:1812.08775*.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2989–2998.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

David G Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Satoshi Sekine, Akio Kobayashi, and Kouta Nakayama. 2019. SHINRA: Structuring wikipedia by collaborative contribution. In *Automated Knowledge Base Construction (AKBC)*.

Aili Shen, Bahar Salehi, Jianzhong Qi, and Timothy Baldwin. 2020. A general approach to multimodal document quality assessment. *Journal of Artificial Intelligence Research*, 68:607–632.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Hiyori Yoshikawa, Chunpeng Ma, Aili Shen, Qian Sun, Chenbang Huang, Guillaume Pelat, Akiva Miura, Daniel Beck, Timothy Baldwin, and Tomoya Iwakura. 2020. UOM-FJ at the NTCIR-15 SHINRA2020-ML task. In *Proceedings of the 15th NTCIR Conference on Evaluation of Information Access Technologies*, pages 201–207.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

If you've got it, flaunt it:
Making the most of fine-grained sentiment annotations

Jeremy Barnes, Lilja Øvrelid, Erik Velldal
University of Oslo
Department of Informatics
`{jeremycb,liljao,erikve}@ifi.uio.no`

Abstract

Fine-grained sentiment analysis attempts to extract sentiment holders, targets and polar expressions and resolve the relationship between them, but progress has been hampered by the difficulty of annotation. *Targeted* sentiment analysis, on the other hand, is a more narrow task, focusing on extracting sentiment targets and classifying their polarity.

In this paper, we explore whether incorporating holder and expression information can improve target extraction and classification and perform experiments on eight English datasets. We conclude that jointly predicting target and polarity BIO labels improves target extraction, and that augmenting the input text with gold expressions generally improves targeted polarity classification. This highlights the potential importance of annotating expressions for fine-grained sentiment datasets. At the same time, our results show that performance of current models for predicting polar expressions is poor, hampering the benefit of this information in practice.

1 Introduction

Sentiment analysis comes in many flavors, arguably the most complete of which is what is often called fine-grained sentiment analysis (Wiebe et al., 2005; Liu, 2015). This approach models the sentiment task as minimally extracting all opinion holders, targets, and expressions in a text and resolving the relationships between them. This complex task is further complicated by interactions between these elements, strong domain effects, and the subjective nature of sentiment. Take the annotated sentence in Figure 1 as an example. Knowing that the target "UMUC" is modified by the expression "5 stars" and not "don't believe" is important to correctly classifying the polarity. Additionally, the fact that this is a belief held by "some others" as apposed to the author of the sentence can help us determine the overall polarity expressed in the sentence.

Compared to document- or sentence-level sentiment analysis, where distant labelling schemes can be used to obtain annotated data, fine-grained annotation of sentiment does not occur naturally, which means that current machine learning models are often hampered by the small size of datasets. Furthermore, fine-grained annotation is demanding, leads to relatively small datasets, and has low inter-annotator agreement (Wiebe et al., 2005; Wang et al., 2017). This begs the question: **is it worth it to annotate full fine-grained sentiment**?

Targeted sentiment (Mitchell et al., 2013; Zhang et al., 2015) is a reduction of the fine-grained sentiment task which concentrates on extracting sentiment targets and classifying their polarity, effectively ignoring sentiment holders and expressions. The benefit of this setup is that it is faster to annotate and simpler to model. But **would targeted sentiment models benefit from knowing the sentiment holders and expressions**?

In this work, we attempt to determine whether holder and expression information is useful for extracting and then classifying sentiment targets. Specifically, we ask the following research questions:

> **RQ1:** Given the time and difficulty required to annotate opinion holders, expressions, and polarity, is this information useful to **extract sentiment targets**?
>
> (a) Does augmenting the input text with holders and expressions improve target extraction?
>
> (b) Do target extraction models benefit from predicting holders and expressions?

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 49–62
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

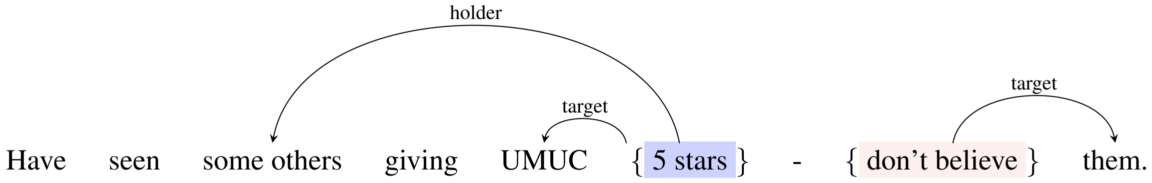


Figure 1: An opinion annotation from the Darmstadt Review Corpus.

(c) Do target extraction models benefit from predicting the polarity of targets and/or expressions?

RQ2: Can holder and expression information improve **polarity classification** on extracted targets?

(a) Does augmenting the input text with holders and expressions improve polarity classification?

(b) Do potential benefits of augmenting the input depend on how we model the target, *i.e.,* using the [CLS] embeddings, mean pooling the target embeddings, etc.?

(c) Can sentiment lexicons provide enough information on expressions to give improvements?

We conduct a series of experiments on eight English sentiment datasets (three with full fine-grained sentiment and five targeted) with state-of-the-art models based on fine-tuned BERT models. We show that (1) it is possible to improve target extraction by also trying to predict the polarity, and that (2) classification models benefit from having access to information about sentiment expressions. We also (3) release the code[1] to reproduce the experiments, as well as the scripts to download, preprocess, and collect the datasets into a compatible JSON format, with the hope that this allows future research on the same data.

2 Related work

Fine-grained approaches to sentiment analysis attempt to discover opinions from text, where each opinion is a tuple of *(opinion holder, opinion target, opinion expression, polarity, intensity)*. Annotation of datasets for this granularity requires creating in-depth annotation guidelines, training annotators, and generally leads to lower inter-annotator scores than other sentiment tasks, *e.g.,* document- or sentence-level classification, as deciding on the spans for multiple elements and their relationships is undeniably harder than choosing a single label for a full text. **Targeted sentiment**, on the other hand, generally concentrates only on target extraction and polarity classification. This has the benefit of allowing non-experts and crowd-sourcing to perform annotation, making it easier to collect larger datasets for machine learning. This simplified annotation can be crowd-sourced, leading to larger datasets for machine learning.

2.1 Datasets

The Multi-purpose Question Answering dataset (*MPQA*) (Wiebe et al., 2005) is the first dataset that annotated opinion holders, targets, expressions and their relationships. The news wire data leads to complex opinions and a generally difficult task for sentiment models. Normally, the full opinion extraction task is modelled as extraction of the individual elements (holders, targets, and expressions) and the subsequent resolution of the relationship between them.

The *Darmstadt Review Corpora* (Toprak et al., 2010) contain annotated opinions for consumer reviews of universities and services. The authors annotate holders, targets, expressions, polarity, modifiers, and intensity. They achieve between 0.5 and 0.8 agreement using the *agr* method (Wiebe et al., 2005), with higher disagreement on what they call "polar targets" – targets that have a polarity but no annotated sentiment expression – holders, and expressions.

The *Open* Domain Targeted dataset (Mitchell et al., 2013) makes use of crowd sourcing to annotate NEs from scraped tweets in English and Spanish (Etter et al., 2013) with their polarities. The authors use majority voting to assign the final labels for the NEs, discarding tweets without sentiment consensus on all NEs.

---

[1] `https://github.com/ltgoslo/finegrained_modelling`

The 2014 *SemEval* shared task (Pontiki et al., 2014) on aspect-based sentiment analysis include labeled data from restaurant and laptop reviews for two subtasks: 1) target extraction, which they call "aspect term extraction" and 2) classification of polarity with respect to targets ("aspect term polarity").

As most targeted datasets only contain a single target, or multiple targets with the same polarity, sentence-level classifiers are strong baselines. In order to mitigate this, Jiang et al. (2019) create a *Challenge* dataset which has both multiple targets and multiple polarities in each sentence. Similarly, Wang et al. (2017) also point out that most targeted sentiment methods perform poorly with multiple targets and propose *TDParse*, a corpus of UK election tweets with multiple targets per tweet.

2.2 Modelling

Katiyar and Cardie (2016) explore jointly extracting holders, targets, and expressions with LSTMs. They find that adding sentence-level and relation-level dependencies (IS-FROM or IS-ABOUT) improve extraction, but find that the LSTM models lag behind CRFs with rich features.

Regarding modelling the interaction between elements, there are several previous attempts to jointly learn to extract and classify targets, using factor graphs (Klinger and Cimiano, 2013), multi-task learning (He et al., 2019) or sequence tagging with collapsed tagsets representing both tasks (Li et al., 2019). In general, the benefits are small and have suggested that there is only a weak relationship between target extraction and polarity classification (Hu et al., 2019).

3 Data

One of the difficulties of working with fine-grained sentiment analysis is that there are only a few datasets (even in English) and they come in incompatible, competing data formats, *e.g.,* BRAT or various flavors of XML. With the goal of creating a simple unified format to work on fine-grained sentiment tasks, we take the eight datasets mentioned in Section 2 – *MPQA* (Wiebe et al., 2005), *Darmstadt Services and Universities* (Toprak et al., 2010), *TDParse* (Wang et al., 2017), *SemEval Restaurant and Laptop* (Pontiki et al., 2014), *Open* Domain Targeted Sentiment (Mitchell et al., 2013), and the *Challenge* dataset from Jiang et al. (2019) – and convert them to a standard *JSON* format. The datasets are sentence and word tokenized using NLTK (Loper and Bird, 2002), except for MPQA, DS. Service and DS. Uni, which already contain sentence and token spans. All polarity annotations are mapped to positive, negative, neutral, and conflict[2]. As such, each sentence contains a sentence id, the tokenized text, and a possibly empty set of opinions which contain a holder, target, expression, polarity, and intensity. We allow for empty holders and expressions in order generalize to the targeted corpora. Finally, we use 10 percent of the training data as development and another 10 percent for test for the corpora that do not contain a suggested train/dev/test split. For training and testing models, however, we convert the datasets to CoNLL format.

Table 1 presents an overview of the different datasets and highlights important differences between them. The fully fine-grained sentiment datasets (MPQA, DS. Services, and DS. Uni) tend to be larger but have fewer targets annotated, due to a larger number of sentences with no targets. However, the MPQA dataset contains much longer targets than the other datasets – an average of 6, but a maximum of 56 tokens. It also contains more opinion holders and expressions and these also tend to be longer, all of which marks MPQA as an outlier among the datasets. The distribution of polarity is also highly dependent on the dataset, with DS. Services being the most skewed and SemEval Laptop the least skewed. Finally, the challenge dataset is by far the largest with over 11,000 training targets. Additionally, Table 6 in Appendix A shows the percentage of unique targets per dataset, as well as the percentage of targets shared between the training set and the dev and test sets. Again MPQA has the largest number of unique targets and the least overlap.[3]

4 Experimental Setup

We split the task of targeted sentiment analysis into the **extraction of sentiment targets** and subsequent **polarity classification** of extracted targets, given their context. Figure 2 shows the two tasks and the eight models used in the experiments. As a base model, we take the tar-

---

[2] We discard conflict during evaluation because there are not enough examples to properly learn this class in most datasets

[3] We do not, however, consider partial overlap which may exaggerate the true uniqueness of targets.

		domain		sentences		holders			targets			expressions			polarity		
				#	avg.	#	avg.	max	#	avg.	max	#	avg.	max	+	neu	−
Fine-grained Sentiment	MPQA	newswire	train	4500	25	1306	2.6	27	1382	6.1	56	1656	2.4	14	675	271	658
			dev	1622	23	377	2.6	16	449	5.3	41	552	2.1	8	241	105	202
			test	1681	24	371	2.8	32	405	6.4	42	479	2.0	8	166	89	199
	DS. Services	service reviews	train	5913	16	18	1.2	2	2504	1.2	7	1273	1.2	10	1623	46	838
			dev	744	18	1	1.7	3	288	1.2	4	144	1.4	5	103	1	104
			test	748	17	2	1	1	328	1.2	5	168	1.4	6	241	7	80
	DS. Uni	university reviews	train	2253	20	65	1.2	2	1252	1.2	5	837	1.9	9	495	149	610
			dev	232	9	17	1.1	3	151	1.2	3	106	1.7	6	40	19	92
			test	318	20	12	1.3	4	198	1.2	6	139	2.0	5	77	18	103
Targeted Sentiment	TDParse	political tweets	train	2889	6.9	-	-	-	9088	1.2	7	-	-	-	1238	3931	3919
			dev	321	6.6	-	-	-	1040	1.2	5	-	-	-	128	454	458
			test	867	6.9	-	-	-	2746	1.2	6	-	-	-	378	1162	1206
	SemEval R.	restaurant reviews	train	2740	13	-	-	-	3293	1.4	19	-	-	-	1902	574	734
			dev	304	11.3	-	-	-	350	1.4	5	-	-	-	226	54	63
			test	800	9.6	-	-	-	1128	1.4	8	-	-	-	724	195	195
	SemEval L.	laptop reviews	train	2744	22.5	-	-	-	2049	1.5	6	-	-	-	870	402	747
			dev	304	21.1	-	-	-	244	1.6	5	-	-	-	99	44	96
			test	800	18.6	-	-	-	633	1.6	7	-	-	-	327	162	128
	Open	tweets	train	1903	12.8	-	-	-	2594	1.6	8	-	-	-	578	1801	215
			dev	211	12.3	-	-	-	291	1.6	6	-	-	-	46	220	25
			test	234	11.6	-	-	-	337	1.6	7	-	-	-	74	232	31
	Challenge	restaurant reviews	train	4297	8.8	-	-	-	11186	1.3	9	-	-	-	3380	5042	2764
			dev	500	8.9	-	-	-	1332	1.3	8	-	-	-	403	604	325
			test	500	8.9	-	-	-	1336	1.3	8	-	-	-	400	607	329

Table 1: Stastistics of the datasets, including number of sentences, as well as average, and max lengths (in tokens) for holder, target, and expression annotations. Additionally, we include the distribution of polarity – restricted to positive, neutral, and negative – in each dataset.

get extraction and classification models from Xu et al. (2019), which achieve state-of-the-art performance on the SemEval task. The approach first fine-tunes BERT (Devlin et al., 2019) on domain-specific unlabeled data as a domain-adaptation step. We use the datasets themselves to perform this step, except for the SemEval datasets. For these, we follow Rietzler et al. (2020) and instead use larger amounts of unlabeled data – 1,710,553 and 2,000,000 sentences for SemEval Laptop and Restaurant respectively – taken from Amazon Laptop reviews (He and McAuley, 2016) and the Yelp Dataset Challenge.[4] We further deviate from Xu et al. (2019) by not pretraining the models on the SQUAD question answering dataset and in-domain sentiment questions which they create, as this data is not publicly available. Finally, a linear prediction is added after the BERT model and the full model is updated on the sentiment task.

For **target extraction**, we use the contextualized BERT embeddings as input to a softmax layer and predict the sequence of tags. We compare three prediction strategies:

1. **TARG.:** The model predicts the labels $y \in \{B,I,O\}$ for the targets only.
2. **PRED.:** We additionally predict the labels for holders and expressions and predict $y \in \{B\text{-}holder, I\text{-}holder, B\text{-}target, I\text{-}target, B\text{-}expression, I\text{-}expression, O\}$.
3. **+POL.:** Finally, we add the polarity (*positive*, *negative*, *neutral*) to the annotation specific BIO-tag, which leads to an inventory of 19 labels for the full fine-grained setup and 7 for the targeted setup.

For **polarity classification**, we take as a baseline the classification architecture from Xu et al. (2019), which makes use of the two-sentence training procedure for BERT, by prepending the target before the sentence separation token, and then adding the full sentence after. We compare five strategies for producing the input to the softmax layer for predicting the sentiment of the target:

1. **[CLS]:** this model uses the [CLS] embedding from the final BERT layer.

---

[4] `https://www.yelp.com/dataset/challenge`

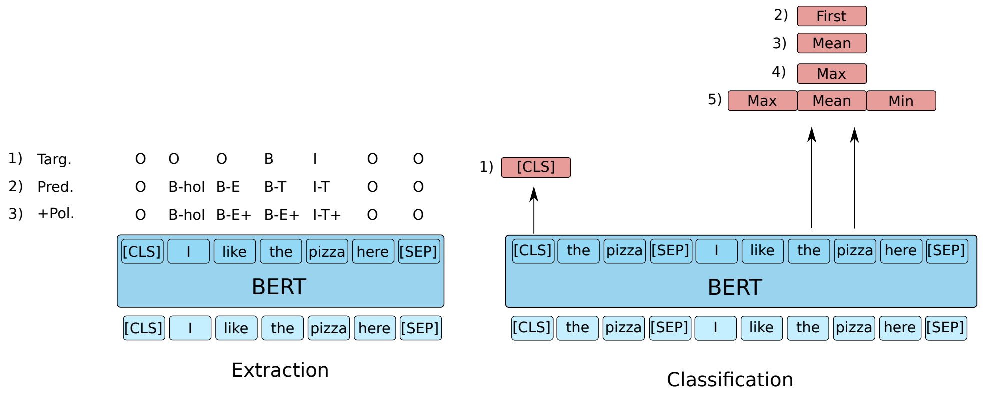


Figure 2: Our BERT-based *target extraction* and *classification models*, with the three strategies for extraction ((1) predict only targets, (2) predict holders, targets and expressions, and (3) predict the polarity of the targets and expressions as well) and five strategies for sentiment classification (passing to the softmax layer the contextualized embedding from (1) the [CLS] embedding, (2) the first token in the target (3) averaging all embeddings in the target phrase, (4) taking the max of the target embeddings, (5) concatenating the max, mean, and min).

2. **FIRST:** uses the contextualized BERT embedding from the first token of the target in context.
3. **MEAN:** instead takes the average of the BERT embeddings for the tokens in the target.
4. **MAX:** uses the max of the contextualized BERT embeddings for the tokens in the target.
5. **MAXMM:** takes the max, min, and mean pooled representations and passes the concatenation to the softmax layer, which has shown to perform well for sentiment tasks (Tang et al., 2014). However, this triples the size of the input representation to the softmax layer.

The TARG. and [CLS] models correspond to the models used in Xu et al. (2019) and serve as baselines. The extraction and classification models are fine-tuned for 50 epochs using Adam with an initial learning rate of 3e−5, with a linear warmup of 0.1 and all other hyperparameters are left at default BERT settings (further details in Appendix B). The best model on the development set is used for testing. Combined with the four input manipulations (Table 2), this leads to eleven extraction experiments – TARG. and PRED. on the original data which only has annotated targets are the same and for simplicity we only show the results from TARG.– and twenty classification experiments per dataset. In order to control for the effect of random initialization, we run each experiment 5 times on different random seeds and report the mean and standard deviation.

4.1 Training with gold annotations

Given that we are interested in knowing whether it is beneficial to include information about additional annotations (holder, expressions, polarity), we perform experiments where we systematically include these. We do so by adding special tags, *e.g.*,, [<E], into the input text surrounding the annotated spans, as shown in Table 2. The models then have access to this information both during training and at test time, albeit in an indirect way. For the first set of experiments, we perform controlled experiments under ideal conditions, *i.e.*, having gold annotations during testing. This allows us to isolate the effects of incorporating the additional annotations, without worrying about noisy predictions

4.2 Training with predicted expressions

It is equally important to know whether the models are able to use noisy predicted annotations. In order to test this, we train *expression prediction models* on the three full fine-grained sentiment corpora. We use the same BERT-based model and hyperparameters from the target extraction models above and train five models with different random seeds. Preliminary results suggested that these models had high precision, but low recall. Therefore, we take a simple ensemble of the five trained models, where for each token, we keep labels predicted by at least one of the expression models in order to increase recall.

We perform an additional set of experiments where we use sentiment lexicons and assume any word in these lexicons is a sentiment expres-

original	Money Magazine rated E-Trade highly .
+ holders	[<H] Money Magazine [H>] rated E-Trade highly
+ expressions	Money Magazine [<E] rated [E>] E-Trade [<E] highly [E>]
+ full	[<H] Money Magazine [H>] [<E] rated [E>] E-Trade [<E] highly [E>]

Table 2: We inform our models regarding annotations other than targets by inserting special tags into the input text before and after annotated holders and expressions .

sion. We use the **Hu and Liu** lexicon (Hu and Liu, 2004), the **SoCal** and **SoCal-Google** lexicons (Taboada et al., 2006) and the **NRC** emotion lexicon (Mohammad and Turney, 2013), which also contains sentiment annotations. The lexicons contain 6,789, 5,824, 2,142, and 5,474 entries, respectively. The MPQA and Darmstadt experiments show the effect of predicted vs. gold expressions, as well as domain transfer. The experiments on the targeted datasets, on the other hand, will show us whether it is possible to improve the targeted models with predicted expressions.

5 Results

In this section we describe the main results from the extraction and two classification experiments described in Section 4.

5.1 Target extraction

Table 3 shows the results for the extraction experiment, where token-level F_1 is measured only on targets. The models perform poorer than the state-of-the-art, as we did not finetune on the SQUAD question answering dataset and in-domain sentiment questions or perform extensive hyperparameter tuning. The average F_1 score depends highly on the dataset – MPQA is the most difficult dataset with 13.1 F_1 on the original data, while the Darmstadt Universities corpus is the easiest for target extraction with 84.6. Augmenting the input text with further annotations, but predicting only sentiment targets (Targ. in Table 3) hurts the model performance in all cases. Specifically, adding holder tags leads to an average drop of 1.3 percentage points (pp), expressions 1.2 and full 1.5. Attempting to additionally predict these annotations (Pred. in Table 3) leads to mixed results – the model leads to improvements on MPQA + exp. and Darmstadt Services + holders, no notable difference on MPQA + full and Darmstadt Universities + exp., and a loss on the rest.

Adding the polarity to the target BIO tags (original +Pol. in Table 3) leads to the most consistent improvements across experiments – an average of 0.5 pp – with the largest improvement of 1.5 pp on the TDParse dataset. This suggests a weak-to-moderate relationship between polarity and extraction, which contradicts previous conclusions (Hu et al., 2019). Finally, further adding the holder and expression tags (+Pol. in Table 3) tends to decrease performance.

5.2 Polarity classification with gold annotations

Table 4 shows the macro F_1 scores for the polarity classification task on the gold targets. The model performs better than the best reported results on Challenge (Jiang et al., 2019), and similar to previous results on the SemEval corpora. Regarding the choice of target representation, First is the strongest overall, with an average of 64.7 F_1 across the original eight datasets, followed by Max (64.6), Mean (64.4), MaxMM (64.2), and finally [Cls] (64.1). It is, however, unclear exactly which representation is the best, as it differs for each dataset. But we can conclude that [Cls] is in general the weakest model, while either First or Max provide good starting points.

Adding holder annotations to the input text delivers only small improvements on four of the fifteen experiments, and has losses on seven. The +exp. model, however, leads to significant improvements on 10 experiments. The outlier seems to be Darmstadt Services, which contains a large number of "polar targets" in the data, which do not have polar expressions. This may explain why including this information has less effect on this dataset. Finally, +full performs between the original input and +exp.

		MPQA	DS. Services	DS. Unis	Challenge	SemEval R.	SemEval L.	Open	TDParse
	Xu et al. (2019)	n/a	n/a	n/a	n/a	78.0	84.3	n/a	n/a
	BiLSTM-CRF	12.2 (1)	85.0 (1)	84.4 (1)	73.4 (1)	72.5 (1)	74.0 (1)	62.2 (1)	82.6
	original	14.1 (2)	85.9 (1)	**84.6 (0)**	75.8 (1)	51.9 (1)	71.3 (1)	62.0 (4)	81.7 (3)
TARG.	+ holders	11.9 (1)	84.3 (1)	83.6 (1)	-	-	-	-	-
	+ exp.	11.6 (1)	85.0 (0)	83.4 (0)	-	-	-	-	-
	+ full	10.5 (2)	84.8 (1)	83.8 (1)	-	-	-	-	-
PRED.	+ holders	12.1 (2)	**86.2 (0)**	**84.6 (0)**	-	-	-	-	-
	+ exp.	**14.9 (1)**	84.7 (1)	84.5 (1)	-	-	-	-	-
	+ full	13.0 (3)	85.5 (1)	84.3 (1)	-	-	-	-	-
+POL.	BiLSTM-CRF	13.9 (1)	85.2 (1)	83.7 (1)	73.6 (1)	**73.7 (1)**	**74.5 (1)**	62.3 (1)	81.8 (1)
	original	13.8 (1)	85.4 (1)	84.3 (1)	**76.9 (1)**	52.5 (1)	**71.6 (1)**	**62.9 (1)**	**83.2 (0)**
	+ holders	13.8 (2)	85.6 (1)	84.4 (1)	-	-	-	-	-
	+ exp.	13.5 (2)	85.4 (1)	84.3 (0)	-	-	-	-	-
	+ full	12.0 (1)	86.0 (1)	**84.6 (0)**	-	-	-	-	-

Table 3: Average token-level F1 scores for the target extraction task across five runs, (standard deviation in parenthesis). **Bold** numbers indicate the best model per dataset, while blue and pink highlighting indicates an improvement or loss in performance compared to the original data, respectively.

5.3 Polarity classification with predicted annotations

The expression models achieve modest F_1 scores when trained and tested on the same dataset – between 15.0 and 47.9 –, and poor scores when transferred to a different dataset – between 0.9 and 14.9 (further details shown in Table 7 in Appendix A). The lexicons often provide better cross-dataset F_1 than the expression models trained on another dataset, as they have relatively good precision on general sentiment terms.

Figure 3 shows a heatmap of improvements (blue) and losses (red) on the eight datasets (x-axis) when augmenting the input text with expression tags from the expression models and lexicons (y-axis). We compare the expression augmented results to the original results for each pooling technique and take the average of these improvements and losses. For a full table of all results, see Table 5 in Appendix A.

Augmenting the input text with predicted sentiment expressions leads to losses in 41 out of averaged 56 experiments shown in Figure 3 (or in 173 out of 280 experiments in Table 5). Curiously, the experiments that use an expression model trained on the same dataset as the classification task, *e.g.,* MPQA predicted expressions on the MPQA classification task, have the largest losses – the largest of which is MPQA (-2.78 on average). This seems to indicate that the mismatch between the train prediction, which are near perfect, and the rather poor test predictions is more problematic than cross-dataset predictions, which are similar on train and test.

The best expression prediction model is the one trained on MPQA, improving the performance on Darmstadt Universties, Open, and SemEval Restaurants. This is likely due to the fact that MPQA has the largest number of annotated expressions, and that the domain is more general, leading to expression predictions that generalize better. The expression models trained on Darmstadt Services leads to small benefits on two corpora and the expression model trained on Darmstadt Universities only leads to losses

The datasets that receive the most benefit from expression annotations are Darmstadt Universities (6/7 experiments) and the TDParse dataset (5/7). In both cases, the lexicon-based expression models provide more consistent benefits than the trained expression prediction models. The fact that the dataset that benefits most is the TDParse dataset suggests that expression information is most useful when there are multiple targets with multiple polarities.

There is no significant correlation between the performance of the expression prediction model and the performance on the classification task on the three fine-grained datasets. In fact, there is a small but insignificant negative correlation (-0.33

		MPQA	DS. Services	DS. Unis	Challenge	SemEval R.	SemEval L.	Open	TDParse
	Previous Results	n/a	n/a	n/a	70.3	80.1	78.3		
[CLS]	original	63.5 (2)	57.3 (1)	57.6 (4)	84.3 (0)	74.1 (2)	72.8 (1)	54.6 (1)	48.8 (1)
	+ holders	63.1 (2)	57.1 (1)	60.5 (0)	-	-	-	-	-
	+ exp.	64.0 (3)	56.4 (0)	62.9 (4)	-	-	-	-	-
	+ full	61.9 (2)	56.6 (1)	62.8 (2)	-	-	-	-	-
FIRST	original	64.3 (2)	57.8 (1)	58.7 (4)	**84.4 (1)**	75.6 (1)	74.3 (1)	55.6 (2)	46.6 (1)
	+ holders	63.4 (2)	57.7 (2)	60.5 (3)	-	-	-	-	-
	+ exp.	**64.8 (2)**	57.0 (1)	63.7 (2)	-	-	-	-	-
	+ full	64.0 (1)	55.2 (1)	**65.7 (4)**	-	-	-	-	-
MEAN	original	63.5 (2)	57.3 (1)	60.2 (4)	**84.4 (1)**	74.1 (2)	72.8 (1)	56.8 (3)	46.1 (1)
	+ holders	63.1 (2)	57.8 (1)	56.7 (5)	-	-	-	-	-
	+ exp.	64.3 (2)	56.2 (1)	64.1 (3)	-	-	-	-	-
	+ full	64.2 (2)	56.3 (1)	63.7 (2)	-	-	-	-	-
MAX	original	60.8 (4)	58.2 (1)	57.8 (3)	81.4 (1)	73.9 (2)	**74.5 (2)**	**61.4 (5)**	**49.0 (3)**
	+ holders	61.9 (4)	57.9 (1)	53.9 (1)	-	-	-	-	-
	+ exp.	64.3 (2)	57.4 (1)	61.5 (6)	-	-	-	-	-
	+ full	62.7 (3)	57.9 (1)	54.5 (2)	-	-	-	-	-
MAXMM	original	59.3 (2)	57.8 (1)	55.2 (3)	81.3 (1)	**77.2 (1)**	**74.5 (1)**	60.2 (5)	48.5 (5)
	+ holders	61.3 (1)	57.8 (1)	54.7 (3)	-	-	-	-	-
	+ exp.	64.1 (2)	**59.8 (3)**	54.0 (2)	-	-	-	-	-
	+ full	63.9 (1)	57.7 (1)	54.4 (4)	-	-	-	-	-

Table 4: Average macro F_1 scores for polarity classification across five runs (standard deviation in parenthesis) on gold targets, also adding information about holders and expressions. **Bold** indicates the best model per dataset, while blue and pink highlighting indicates an improvement or loss in performance compared to the original (targets only) data, respectively.

p=0.13, -0.16 p=0.48, -0.26 p=0.25 for macro Precision, Recall, or F_1 respectively, as measured by Pearson's correlation between the expression performances and the F_1 of the classification models augmented with these predicted expressions). It seems that the possible benefits depends more on the target dataset than the actual expression model used.

6 Conclusion

In this work we have explored the benefit of augmenting targeted sentiment models with holder and sentiment expressions. The experiments have shown that although augmenting text with holder and expression tags (**RQ1 a**) or simultaneously predicting them (**RQ1 b**) have no benefit for target extraction, predicting collapsed BIO + polarity tags consistently improves target extraction (**RQ1 c**). Furthermore, augmenting the input text with gold expressions generally improves targeted polarity classification (**RQ2 a**), although it is not clear which target representation strategy is best (**RQ2 b**). Furthermore, we have found benefits of including lexicon-based expressions for the more complex targeted datasets (**RQ2 c**).

The rather poor performance of the learned expression models and the difference between augmenting with gold or predicted expressions reveals the need to improve expression prediction approaches, both by creating larger corpora annotated with sentiment expressions, as well as performing further research on the modeling aspect. Any future work interested in modelling more complex sentiment phenomena should therefore be aware that we may first require more high-quality annotated data if we wish to do so with current state-of-the-art machine learning approaches.

Furthermore, we introduce a common format for eight standard English datasets in fine-grained sentiment analysis and release the scripts to download and preprocess them easily. We plan to include further datasets in our script in the future, as well as extending our work to other languages with available fine-grained corpora.

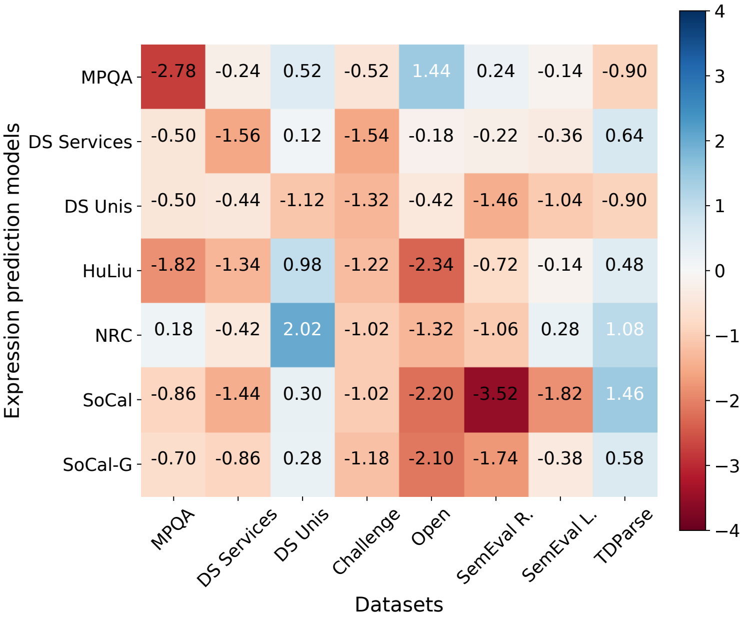


Figure 3: Heatmap of average improvements (blue) and losses (red) on the target classification tasks (x-axis) when augmenting the input text with predicted sentiment expressions from the expression prediction models (y-axis).

Acknowledgements

This work has been carried out as part of the SANT project (Sentiment Analysis for Norwegian Text), funded by the Research Council of Norway (grant number 270908). The computation for the experiments was provided by UNINETT Sigma2 - the National Infrastructure for High Performance Computing and Data Storage in Norway

References

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

David Etter, Francis Ferraro, Ryan Cotterell, Olivia Buzek, and Benjamin Van Durme. 2013. Nerit: Named Entity Recognition for Informal Text. Technical Report 11, Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, Maryland.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546, Florence, Italy. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 168–177, Seattle, USA.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6279–6284, Hong Kong, China. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 919–929, Berlin, Germany. Association for Computational Linguistics.

Roman Klinger and Philipp Cimiano. 2013. Bi-directional inter-dependencies of subjective expressions and targets and their value for a joint model. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 848–854, Sofia, Bulgaria. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019)*, pages 6714–6721, Honolulu, Hawaii. AAAI Press.

Bing Liu. 2015. *Sentiment analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, Cambridge, United Kingdom.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, pages 63–70.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA. Association for Computational Linguistics.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 29(3):436–465.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2020. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4933–4941, Marseille, France. European Language Resources Association.

Maite Taboada, Caroline Anthony, and Kimberly Voll. 2006. Methods for Creating Semantic Orientation Dictionaries. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Bo Wang, Maria Liakata, Arkaitz Zubiaga, and Rob Procter. 2017. TDParse: Multi-target-specific sentiment recognition on twitter. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 483–493, Valencia, Spain. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation (formerly Computers and the Humanities)*, 39(2/3):164–210.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621, Lisbon, Portugal. Association for Computational Linguistics.

Appendix

Appendix A Additional tables

			MPQA	DS. Services	DS. Unis	Challenge	Open	SemEval R.	SemEval L.	TDParse
	original	[CLS]	63.5 (2)	57.3 (1)	57.6 (4)	84.3 (0)	54.6 (1)	74.1 (2)	72.8 (1)	48.8 (1)
		FIRST	**64.3 (2)**	57.8 (1)	58.7 (4)	**84.4 (1)**	55.6 (2)	75.6 (1)	74.3 (1)	46.6 (1)
		MEAN	63.5 (2)	57.3 (1)	60.2 (4)	**84.4 (1)**	56.8 (3)	74.1 (2)	72.8 (1)	46.1 (1)
		MAX	60.8 (4)	**58.2 (1)**	57.8 (3)	81.4 (1)	61.4 (5)	73.9 (2)	74.5 (2)	49.0 (3)
		MAXMM	59.3 (2)	57.8 (1)	55.2 (3)	81.3 (1)	60.2 (5)	**77.2 (1)**	74.5 (1)	48.5 (5)
Predicted Expressions	MPQA	[CLS]	60.3 (2)	57.0 (1)	61.3 (5)	83.1 (1)	57.5 (4)	74.2 (2)	72.2 (1)	47.5 (2)
		FIRST	61.6 (2)	57.0 (2)	59.8 (3)	83.5 (1)	55.2 (3)	77.1 (1)	73.9 (2)	45.2 (1)
		MEAN	60.3 (2)	57.0 (1)	61.3 (5)	83.1 (1)	57.5 (4)	74.2 (2)	72.2 (1)	47.5 (2)
		MAX	59.1 (2)	58.1 (0)	57.0 (4)	82.3 (0)	**63.7 (1)**	75.0 (2)	74.7 (1)	48.5 (2)
		MAXMM	56.2 (5)	58.1 (1)	52.7 (2)	81.3 (1)	61.9 (3)	75.6 (2)	75.2 (1)	45.8 (4)
	DS. Services	[CLS]	63.6 (1)	56.3 (1)	60.6 (1)	82.4 (1)	53.4 (4)	72.1 (2)	72.2 (1)	49.4 (2)
		FIRST	61.3 (2)	54.5 (0)	59.4 (3)	82.6 (1)	56.2 (9)	76.3 (1)	74.8 (1)	45.4 (2)
		MEAN	64.1 (2)	56.0 (0)	60.1 (5)	82.4 (1)	56.1 (3)	73.0 (2)	72.4 (1)	49.8 (2)
		MAX	61.4 (1)	56.7 (1)	55.2 (2)	80.3 (1)	63.0 (2)	76.5 (1)	74.5 (1)	48.1 (3)
		MAXMM	58.5 (2)	57.1 (1)	54.8 (5)	80.4 (1)	59.0 (2)	75.9 (2)	73.2 (2)	49.5 (4)
	DS. Unis	[CLS]	63.1 (1)	57.0 (1)	60.3 (3)	82.8 (1)	54.7 (2)	73.6 (3)	72.8 (1)	48.1 (3)
		FIRST	64.1 (2)	56.9 (1)	58.2 (2)	82.6 (1)	55.2 (3)	70.8 (3)	72.9 (2)	44.4 (1)
		MEAN	62.3 (1)	57.0 (1)	59.7 (3)	82.8 (1)	54.7 (2)	73.6 (3)	72.8 (1)	48.1 (3)
		MAX	59.6 (4)	57.8 (1)	53.4 (1)	80.5 (1)	62.0 (1)	74.8 (2)	71.0 (1)	46.9 (2)
		MAXMM	59.8 (3)	57.5 (1)	52.3 (4)	80.5 (0)	59.9 (4)	74.8 (1)	74.2 (1)	47.0 (4)
Lexicon Expressions	HuLiu	[CLS]	60.3 (2)	56.2 (1)	60.8 (3)	82.8 (1)	54.0 (3)	73.6 (1)	73.2 (1)	**50.6 (1)**
		FIRST	61.2 (2)	55.0 (1)	61.2 (1)	82.3 (1)	46.5 (3)	74.8 (1)	74.2 (1)	43.8 (1)
		MEAN	60.3 (2)	56.2 (1)	60.8 (3)	82.8 (1)	54.0 (3)	73.6 (1)	73.2 (1)	**50.6 (1)**
		MAX	59.7 (3)	57.2 (1)	56.4 (2)	81.0 (1)	61.1 (2)	75.5 (2)	73.7 (3)	47.2 (3)
		MAXMM	60.8 (3)	57.1 (1)	55.2 (2)	80.8 (1)	61.3 (3)	73.8 (3)	73.9 (1)	49.2 (4)
	NRC	[CLS]	64.0 (2)	56.9 (1)	63.0 (2)	83.1 (1)	54.8 (3)	72.0 (1)	73.0 (1)	49.4 (1)
		FIRST	63.7 (2)	56.9 (1)	61.1 (2)	83.3 (1)	49.1 (5)	74.9 (3)	**74.9 (0)**	46.1 (2)
		MEAN	64.0 (2)	56.9 (1)	**63.0 (2)**	83.1 (1)	54.8 (3)	72.0 (1)	73.0 (1)	49.4 (1)
		MAX	61.1 (3)	58.0 (1)	55.6 (2)	80.4 (1)	62.0 (1)	75.3 (2)	74.6 (2)	49.7 (3)
		MAXMM	59.5 (3)	57.6 (1)	56.9 (4)	80.8 (1)	61.3 (2)	75.4 (2)	74.8 (1)	49.8 (4)
	SoCal	[CLS]	63.2 (2)	56.6 (1)	60.5 (4)	83.0 (0)	51.5 (4)	69.8 (1)	71.0 (1)	50.0 (1)
		FIRST	61.8 (2)	53.7 (2)	59.9 (4)	81.8 (0)	51.4 (6)	72.8 (2)	73.0 (1)	45.4 (2)
		MEAN	63.2 (2)	56.6 (1)	60.5 (4)	83.0 (0)	51.5 (4)	69.8 (1)	71.0 (1)	50.0 (1)
		MAX	59.2 (2)	57.8 (1)	54.5 (2)	79.3 (1)	62.3 (2)	71.5 (4)	71.8 (2)	49.5 (4)
		MAXMM	59.7 (2)	56.5 (2)	55.6 (1)	79.1 (1)	60.9 (3)	73.4 (3)	73.0 (1)	51.4 (2)
	SoCal-Google	[CLS]	62.6 (3)	56.5 (1)	60.0 (3)	83.0 (1)	53.2 (3)	71.5 (1)	72.9 (1)	50.2 (1)
		FIRST	62.1 (1)	56.2 (1)	60.8 (5)	82.5 (1)	49.7 (6)	74.9 (2)	74.0 (1)	46.2 (0)
		MEAN	62.6 (3)	56.5 (1)	60.0 (3)	83.0 (1)	53.2 (3)	71.5 (1)	72.9 (1)	50.2 (1)
		MAX	60.0 (3)	57.8 (0)	55.5 (5)	80.9 (1)	61.5 (3)	74.8 (2)	74.5 (2)	49.5 (4)
		MAXMM	60.6 (4)	57.1 (1)	54.6 (3)	80.5 (1)	60.5 (3)	73.5 (4)	72.7 (2)	45.8 (5)

Table 5: Macro F_1 scores for polarity classification of gold targets. **Bold** numbers indicate the best model per dataset, while blue and pink highlighting indicates an improvement or loss in performance compared to the original data (gold targets only), respectively.

	% Unique			% Overlap	
	train	dev	test	train-dev	train-test
MPQA	**85.7**	88.5	**89.2**	15	19
DS. Services	36.2	48.6	47.5	45.0	35.6
DS. Uni	35.2	52.9	45.0	**58.5**	47.6
TDParse	33	51.8	41.7	57.4	47.3
SemEval R.	36.3	59.8	49.4	56.4	33.8
SemEval L.	45.5	71.7	64.8	48.9	33.7
Open	85	**92.4**	87.1	23	24
Challenge	23.1	39.0	39.7	54.1	**52**

Table 6: Analysis of targets in the datasets. % Unique describes the number of targets that are found only in that split. % Overlap describes the percentage of dev/test targets that are found in the train set. We disregard partial matches, *e.g.,* "chinese food" and "food".

		MPQA	DS. Services	DS. Unis
trained	MPQA	15.0 (1.7)	1.0 (0.8)	2.2 (1.2)
	DS. Services	0.9 (0.3)	47.9 (7.3)	14.9 (1.2)
	DS. Unis	1.4 (0.6)	10.9 (1.5)	18.5 (1.5)
lexicons	HuLiu	4.7	17.9	16.0
	NRC	3.3	7.4	9.0
	SoCal	2.4	13.2	13.8
	SoCal Google	1.0	13.2	11.4

Table 7: Token-level macro F_1 scores for expression prediction models (trained) and lexicon expressions (lexicons) when tested on the three fine-grained datasets (x-axis). The trained model scores are the average and standard deviation across five runs with different random seeds. The lexicon models are deterministic and therefore only have a single score.

Appendix B Training details

GPU Infrastructure	1 NVIDIA P100, 16 GiB RAM
CPU Infrastructure	Intel Xeon-Gold 6126 2.6 GHz
Number of search trials	50
Domain training duration	2580 sec
Extraction fine-tuning duration	15381 sec
Classification fine-tuning duration	9080 sec
Model implementation	https://github.com/blinded/for/review

Hyperparameter	**Assignment**
number of epochs	50
max. sequence length	128
metric early stopping monitored	validation loss
batch size	32
sentiment dropout	0.3
learning rate optimiser	Bert Adam
fine-tuning learning rate	3e-5
learning rate warmup proportion	0.1
regularisation type	L2
regularisation value	0.01

Keep Learning: Self-supervised Meta-learning for Learning from Inference

Akhil Kedia
Samsung Research, Seoul, South Korea
`akhil.kedia@samsung.com`

Sai Chetan Chinthakindi
Samsung Research, Seoul, South Korea
`sai.chetan@samsung.com`

Abstract

A common approach in many machine learning algorithms involves self-supervised learning on large unlabeled data before fine-tuning on downstream tasks to further improve performance. A new approach for language modelling, called dynamic evaluation, further fine-tunes a trained model during inference using trivially-present ground-truth labels, giving a large improvement in performance. However, this approach does not easily extend to classification tasks, where ground-truth labels are absent during inference. We propose to solve this issue by utilizing self-training and back-propagating the loss from the model's own class-balanced predictions (pseudo-labels), adapting the Reptile algorithm from meta-learning, combined with an inductive bias towards pre-trained weights to improve generalization. Our method improves the performance of standard backbones such as BERT, Electra, and ResNet-50 on a wide variety of tasks, such as question answering on SQuAD and NewsQA, benchmark task SuperGLUE, conversation response selection on Ubuntu Dialog corpus v2.0, as well as image classification on MNIST and ImageNet without any changes to the underlying models. Our proposed method outperforms previous approaches, enables self-supervised fine-tuning during inference of any classifier model to better adapt to target domains, can be easily adapted to any model, and is also effective in online and transfer-learning settings.

1 Introduction

It is a common consensus that the performance of Machine Learning algorithms improves with increasing data. However, due to the difficulty of obtaining large quantities of labelled data, many models (particularly in Natural Language Processing domain) such as BERT (Devlin et al., 2019), GPT (Radford et al., 2018) and UniLM (Dong et al., 2019) rely on unsupervised pre-training on unlabelled data to learn useful features which are then fine-tuned for other downstream tasks. While this approach leads to large gains in performance, it leads to a mismatch between a network's pre-training and final fine-tuning. Some approaches such as pseudo-labelling (Lee, 2013) have proposed utilizing data-augmentation of unlabelled data with the model's own predictions to better pre-train a model.

While these methods are limited to the training phase, Krause et al. (2018) proposed to continue training a language modeling model (which is the task of predicting the next token in a sequence of tokens) during the evaluation stage, achieving significant improvements as the model learns to better adapt to the inference data, without any modifications to the model architecture or any access to training data. For language modeling, the ground truth labels are the next input token, which are trivially accessible to the model to facilitate this learning. However, this method does not easily generalize to standard classification tasks due to the unavailability of labels during inference. This is the setting which we further explore in this paper, in which we are provided with a classification model already trained on training data, but with no access to the training data, and the aim is to further improve the performance of the model by utilizing self-training on the inference data.

To solve the above problem, we propose a method to train any classifier model during inference, utilizing methods used in domain adaptation, noisy-label learning, and multi-task meta-learning. With ground truth labels being absent, we utilize the model's own predictions as the pseudo-labels for those samples and utilize Class Balanced Self Training (CBST) (Zou et al., 2018) to filter samples based on the model's confidence while retaining class balance. However, naive online learning or

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 63–77
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

re-training on the inference data is not optimal due to the noise in the labels biasing the network, as well as the small size of the inference set. We solve this issue by leveraging the Reptile Meta Learning Algorithm (Nichol et al., 2018) to improve generalization, supplemented with an explicit inductive bias towards the model's pre-trained weights.

Our experimental results and ablation studies show that our method improves the performance of standard backbones such as BERT, Electra (Clark et al., 2020) and ResNet (He et al., 2016) on a wide variety of tasks, such as question answering on SQuAD (Rajpurkar et al., 2018) and NewsQA (Trischler et al., 2017), benchmark task SuperGLUE (Wang et al., 2019), and conversation re-ranking on Ubuntu Dialog corpus v2.0 (Lowe et al., 2017) for NLP, as well as object classification on MNIST (Deng, 2012) and ImageNet (Deng et al., 2009) without any changes to the underlying models, while outperforming previous approaches. Our method can also be utilized for continual self-supervised fine-tuning of classifiers on target domains, as well as in transfer-learning settings, without any model-level modifications.

2 Proposed Method

Our proposed technique is the self-supervised training of a classifier model during inference, consisting of three parts – using confident predictions as pseudo-labels, utilizing the Reptile algorithm to improve generalization, and an explicit inductive bias to minimize the effect of noisy labels.

2.1 Class Balanced Pseudo-labels

We utilize our classifier's most likely predicted class during inference as hard ground truth labels (pseudo-labels). Hendrycks and Gimpel (2017) show that using a model's own softmaxed probability values, $max_k\{p(y=k|x)\}$, where k are the classes, x is the input, and y is the predicted class is a reasonable proxy for its expected accuracy. To filter out samples with low maximum probabilities, one can simply threshold the output with some fixed value p_t. As proposed by Zou et al. (2018), a separate threshold $p_t(k_{max})$ for each class, where k_{max} is the class with the maximum predicted probability, works better by reducing skewing in favour of easier classes.

In CBST $p_t(k)$ are automatically selected for each k such that a fixed fraction f of examples of each predicted-class are filtered out from the inference set, i.e.,

$$P^k = \{p(y=k|x) | \text{Argmax}_k p(y=k|x) = k\},$$

$$p_t(k) = \max(\{i \mid \frac{|p > i, p \in P^k|}{|P^k|} <= f\}),$$

$$X_t = \{x \mid max_k\{p(y=k|x)\} > p_t(k)\},$$

$$Y_t = \{k \mid max_k\{p(y=k|x)\} > p_t(k)\}$$

These thresholds $p_t(k)$ can be kept fixed based on the validation set, or can be a running estimate in an online setting. Unlike the original CBST, we do not further normalize the class probabilities with these thresholds, as that led to a drastic reduction in the accuracy of pseudo-label classification. X_t inputs with hard pseudo-labels Y_t are used as a training set to further fine-tune the model, using the Reptile Algorithm below. This approach is also unaffected by a lack of model calibration, as long as the model's accuracy on X_t is acceptably high.

2.2 Reptile Algorithm, but for Single Task

Naively using the confident inferred labels for fine-tuning the model is not optimal due to small size of the test set compared to the train set as well as label noise, lowering generalization, and reducing the gains that can be achieved using the pseudo-labels. Since aligned gradients between samples improve a model's generalization, as shown in Chatterjee (2020) and Fort et al. (2019), we leverage the Reptile Meta-Learning Algorithm to this end. The meta-gradient for the Reptile algorithm contains as a component the gradient for maximizing the inner product between different mini-batches from the same task, as we prove in Section 3.

Algorithm 1: REPTILE + l^2sp

```
Input: Batches $B = \{b_0, b_1, \ldots, b_n\}$
       $W = \theta_{0,0} \leftarrow$ Initial network params
Output: Final fine-tuned $\theta$
for $i \leftarrow 0$ to $\lfloor n/k \rfloor$ do
    for $j \leftarrow 0$ to $k - 1$ do
        $\nabla_{\text{inner}} \leftarrow$ grad from $\theta_{i,j}(b_{i*k+j})$
        $\nabla_{\text{LR}} \leftarrow LR_{\text{inner}} * \nabla_{\text{inner}}$
        $l^2sp \leftarrow decay * (\theta_{i,j} - W)$
        $\theta_{i,j+1} \leftarrow \theta_{i,j} - \nabla_{LR} - l^2sp$
    $\nabla_{\text{outer}} \leftarrow (\theta_{i,0} - \theta_{i,k})$
    $\theta_{i+1,0} \leftarrow \theta_{i,0} - LR_{\text{outer}} * \nabla_{\text{outer}}$
return $\theta_{\lfloor n/k \rfloor + 1, 0}$
```

The Reptile Algorithm is a batched First-Order MAML (FO-MAML) Algorithm, originally in-

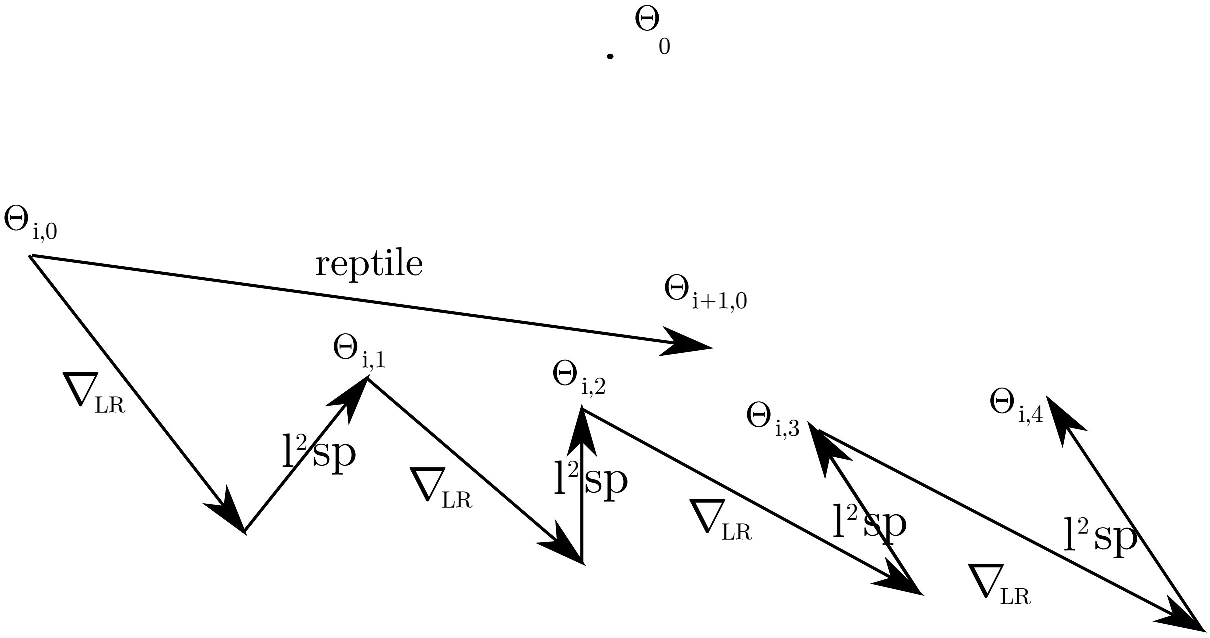


Figure 1: Overview of Reptile with l^2sp update for 4 inner steps.

tended for multi-task meta-learning. We use this algorithm in a single-task setting, as shown in Algorithm 1. The Reptile algorithm consists of $k > 1$ inner steps of standard SGD updates with learning rate LR_{inner}. The difference between original network weights $\theta_{i,0}$ and the final network weights $\theta_{i,k}$ is used as a meta-gradient for SGD for updating the network parameters with a learning rate LR_{outer}, where i is the outer step. The SGD optimizer can be replaced with any other, such as Adam.

The Reptile algorithm for this single task setting is First Order, requiring little extra compute compared to standard optimization, and can be plugged in to any model with ease. Some other multi-task algorithms with Experience-Replay, such as Riemer et al. (2018), may exhibit better learning but are computationally orders of magnitude more expensive and are hence infeasible for large datasets and models.

2.3 Explicit Inductive Bias

While all the models we use employ a weight decay towards 0 in their training phase, given the usually smaller size of the inference set, we regularize the model by biasing the network towards its pre-trained weights instead. For this, we use the l^2sp decay (Li et al., 2018), slowly decaying the model weights between updates towards the initial trained model weights. An example of the update steps involved for $k = 4$ is shown in Fig.1. We conjecture that this will also make the learning more stable to the noisy pseudo-labels.

Some recent works such as Goldblum et al. (2020) also show that standard l^2 weight decay towards 0 may not be ideal and recommend biasing weights towards some model-dependent non-zero norm value instead. l^2sp can be seen as a generalization of the same, while simultaneously taking advantage of the pre-training.

3 Theoretical Analysis

In this section, we provide a theoretical analysis of the meta update of $Reptile + l^2sp$. We generalize the Taylor expansion approach for Reptile as used in (Nichol et al., 2018) to accommodate l^2sp, and show how our approach maximizes the inner product of gradients between different mini-batches.

We consider one set of k inner updates. For $i \in [0, k]$, we define -

$$\begin{aligned}
\theta_i &= \text{network weights before } i^{th} \text{ step},\\
b_i &= \text{input batch for } i^{th} \text{ step},\\
L_i &= \text{loss function corresponding to } b_i,\\
W &= \text{pre-trained network weights for } l^2sp,\\
\beta &= l^2sp \text{ weight decay rate},\\
\alpha &= LR_{inner},\\
g_i &= L_i'(\theta_i), && \text{(gradient of } i^{th}batch)\\
\overline{g_i} &= L_i'(\theta_0), && \text{(gradient at initial point)}\\
H_i &= L_i''(\theta_i), && \text{(Hessian of } i^{th}batch)
\end{aligned}$$

$$\overline{H_i} = L_i''(\theta_0), \qquad \text{(Hessian at initial point)}$$

Then, our update rule is -

$$\theta_i = (1-\beta)\theta_{i-1} - \alpha g_{i-1} + \beta W \tag{1}$$

In the following analysis, to keeps the analysis tractable, we assume both α and β are small and comparable, and ignore terms involving $O(\alpha^2)$, $O(\beta^2)$ and $O(\alpha\beta)$. Using the first order Taylor expansion of g_i, we get -

$$g_i = \overline{g_i} + \overline{H_i}(\theta_i - \theta_0) \tag{2}$$

The following equations can be proved using simple induction on Eq (1) and (2) -

$$\theta_i = \theta_0 + i\beta(W - \theta_0) - \alpha\sum_{j=0}^{i-1}\overline{g_j}, \tag{3}$$

$$g_i = \overline{g_i} + i\beta\overline{H_i}(W - \theta_0) - \alpha\overline{H_i}\sum_{j=0}^{i-1}\overline{g_j}, \tag{4}$$

By summing up the displacements from all variable updates, the expectation of the meta-gradient from Reptile + l^2sp under mini-batch sampling is -

$$\mathbb{E}[-(\theta_k - \theta_0)] = \mathbb{E}[\alpha\sum_{i=0}^{k-1} g_i - \sum_{i=0}^{k-1}\beta(W - \theta_i)]$$

When expanding the terms above with Eq (3) and (4) and simplifying, we get -

$$\begin{aligned}\mathbb{E}[-(\theta_k - \theta_0)] &= c_1\mathbb{E}[\overline{g_i}] + c_2(\theta_0 - W)\\ &-c_3\mathbb{E}[\overline{H_j}\overline{g_i}] - c_4\mathbb{E}[\overline{H_j}(\theta_0 - W)],\end{aligned} \tag{5}$$

where each c_i is a positive constant, dependent on k, α and β.

The first term in R.H.S. of Eq (5) is the gradient which takes us to the minimum of the training problem. For the third term, note that -

$$\begin{aligned}\mathbb{E}[\overline{H_j}\overline{g_i}] = \mathbb{E}[\overline{H_i}\overline{g_j}] &= \frac{1}{2}\mathbb{E}[\overline{H_j}\overline{g_i} + \overline{H_i}\overline{g_j}]\\ &= \frac{1}{2}\mathbb{E}[\frac{\partial}{\partial\theta_0}(\overline{g_i}\cdot\overline{g_j})]\end{aligned}$$

Therefore the third term maximizes the dot product between the gradients of the batches for improved generalization, as in the original Reptile algorithm.

For the second and fourth terms, note that $(\theta_0 - W)$ is the direction of the gradient of the l^2sp, and hence can be interpreted similar to the first and third term, but with training gradients replaced by this l^2 gradient.

Hence, we have shown that the Reptile algorithm maximizing product of gradients for improving generalization holds true in our extension as well.

Corpus	**Task**	**|Train|**	**|Dev|**
BoolQ	QA	9427	3270
CB	NLI	250	57
COPA	QA	400	100
MultiRC	QA	5100	953
ReCoRD	QA	101K	10K
RTE	NLI	2500	278
WiC	WSD	6000	638

Table 1: Description of datasets in SuperGLUE.

Corpus	**Model**	**|Train|**	**|Test|**
MNIST	MLP	60K	10K
ImageNet	ResNet-50	1.2M	50K
SQuAD v2.0	Electra	130K	12K
Ubuntu Diag.	BERT	1M	18K
NewsQA	BERT-trans	97K	5.4K

Table 2: Description of NLP and image datasets. For SQuAD and ImageNet, column 4 refers to validation. Bert-trans is as described in Section 5.3.

4 Experimental Setup

4.1 Benchmark Datasets

SuperGLUE A popular NLP benchmark, which attempts to test various capabilities of language understanding. It itself consists of 8 datasets - Boolean Questions (Clark et al., 2019), Commitment Bank (De Marneffe et al., 2019), Choice of Possible Alternative (Gordon et al., 2012), Multi-Sentence Reading Comprehension (Khashabi et al., 2018), Reading Comprehension with Commonsense Reasoning (Zhang et al., 2018), Recognizing Textual Entailment (a combination of datasets from Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009; Poliak et al., 2018), Word-in-Context (Pilehvar and Camacho-Collados, 2019) and Winograd Schema Challenge (Levesque, 2011).

SQuAD v2.0 A popular span-style QA dataset, consisting of passages from Wikipedia, with questions and corresponding answer spans and unanswerable questions.

Ubuntu Dialog Corpus v2.0 A large-scale corpus of multi-turn conversations mined from Ubuntu IRC chat logs, and the task is to select the best response given a list of possible distractor responses.

NewsQA A span-style QA dataset, consisting of crowd-sourced questions and answers on CNN news articles, along with unanswerable questions.

Model	Params	Speed
Electra-large-cased	340M	8
BERT-large-cased	340M	8
BERT-base-uncased	110M	36
ResNet-50	23M	146
MLP (128H, 2L)	120K	~1M

Table 3: Models, number of network parameters, and training speeds in examples/second on a V100 GPU.

MNIST An image classification dataset of 28x28 scans of handwritten digits. While the dataset has long been *solved*, it nevertheless serves as a useful dataset to compare simpler architectures.

ImageNet A large-scale dataset for image classification, consisting of 1.2M training samples along with their corresponding class labels.

4.2 Models

BERT BERT is a transformer (Vaswani et al., 2017) model, and its derivative models are the backbone of most state-of-the-art models in NLP. We use the official implementation and pre-trained models of BERT-large-cased for SuperGLUE tasks, and BERT-base-uncased for Ubuntu Dialog Corpus, NewsQA, and for our ablation tests on SQuAD.

Electra Electra is a BERT-derived state-of-the-art model in many NLP tasks, with a discriminative pre-training task. We use the official pre-trained Electra-large model, and we implement our own classifier for SQuAD v2.0.

ResNet Residual blocks and their variants are the backbone of most image classification models today. We use Tensorflow Model Garden's implementation and pre-trained ResNet-50 for ImageNet.

MLP While models made of only simple Multi Layer Perceptrons have largely fallen out of favour, fully connected layers are often a part of larger architectures. We use an MLP with 2 Layers and 128 Hidden units as the model for MNIST.

4.3 Implementation Details

Fine-tuning on inference data is extremely quick as our method is first order, taking less than 15 minutes on a V100 for all datasets except ReCoRD and Ubuntu-Dialog, for which it takes a few hours. We use the Adam optimizer, and we disable our model's l^2 weight decay, if any. Batch-norm variables, if any, are also kept fixed.

Corpus	Metric	BERT	BERT + ours
BoolQ	Acc	76.4	**76.6** ± 0.01
CB	F1	88.1	**89.4** ± 0.01
	Acc	91.1	**92.9** ± 0.01
COPA	Acc	71.0	**72.0** ± 0.01
MultiRC	F1a	69.5	**70.0** ± 0.01
	EM	26.4	**26.8** ± 0.01
ReCoRD	F1	72.5	**73.0** ± 0.09
	EM	71.8	**72.4** ± 0.03
RTE	Acc	74.0	**75.1** ± 0.01
WiC	Acc	73.8	**74.3** ± 0.02

Table 4: Results on the validation set of SuperGLUE benchmark dataset, with Bert-large-cased model.

For each dataset, we train one model on the training set, followed by five runs on the pseudo-labeled thresholded inference set with varying seeds, and report the mean and standard deviation of the scores. As the test set for SuperGLUE and SQuAD are hidden, we provide results on the development set instead.

All default/official model hyper-parameters were used for each model/dataset, which can be found in their official source codes linked in the supplemental material, except we use $1e^{-5}$ as LR for Electra as we observed divergence with standard LR. We linearly decay LR except in the online case, where it is kept fixed. The hyper-parameters for Reptile and l^2sp are provided in the supplemental material. A reasonable set of hyper-params, that works across a range of datasets and models we tested, is 0.01 for LR_{outer}, 4 for $inner\_step$, and 0.1 for l^2sp, while LR_{inner} depends on the original model's LR. RTE, BoolQ, and WiC filter out f as 70% of data, while all other datasets filter 50%.

5 Results

5.1 Results on SuperGLUE benchmark

As shown in Table 4, our method consistently improves the performance on all the tasks in SuperGLUE, with very little extra compute, with upto 1.8 increase in accuracy. The gains tend to be larger on smaller datasets, but we observe significant improvement even with the largest task ReCoRD, with over 100K examples.

5.2 Results on other NLP datasets

Our method achieves gains of $0.68/0.72$ F1/EM on SQuAD v2.0 with BERT-base, as shown in Table 7. Even when using a state-of-the-art Electra model

Method	**F1**	**EM**
BERT	76.14	73.14
BERT + CBST (Zou et al., 2018)	76.22 ± 0.02	73.35 ± 0.02
BERT + Disagreement (Malach and Shalev-Shwartz, 2017)	76.23 ± 0.03	73.27 ± 0.06
BERT + Uncertainty Estimation (Zheng and Yang, 2020)	76.25 ± 0.05	73.29 ± 0.04
BERT + Mutual Mean-Teaching (Ge et al., 2020)	76.28 ± 0.04	73.23 ± 0.05
BERT + Co-teaching (Han et al., 2018)	76.28 ± 0.02	73.29 ± 0.04
BERT + Ours	**76.82** ± 0.04	**73.86** ± 0.04

Table 5: Comparison of our method to existing method, on SQuAD v2.0 corpus, using BERT-base-uncased.

Corpus	**Metric**	**Base**	**Base + ours**
MNIST	Acc	98.11	**98.38** ± 0.02
ImageNet	Acc	76.53	**76.69** ± 0.01
SQuAD v2.0	F1	90.13	**90.25** ± 0.01
	EM	87.44	**87.67** ± 0.01
Ubuntu Diag.	R10@1	76.79	**76.89** ± 0.01
NewsQA	F1	44.79	**49.36** ± 0.05
	EM	32.48	**38.71** ± 0.11
NewsQA (online)	F1	44.79	**47.49** ± 0.01
	EM	32.48	**34.11** ± 0.04

Table 6: Results on other NLP and Image datasets.

with SQuAD, we still observe consistent improvements in performance, as shown in Table 6. Even in the presence of large training-set sizes such as that of Ubuntu Dialog Corpus v2.0 with 1M training samples, we still observe consistent increase in performance with the BERT model.

5.3 Results in a Transfer Learning Setting

We also evaluate our approach in a transfer-learning setting on NewsQA, using a BERT-base-uncased model, which was pre-trained on SQuAD v2.0, by self-training on NewsQA train set, followed by evaluation on the test set. Our approach is especially effective in this setting, out-performing the original model by $4.57/6.23$ F1/EM respectively, as shown in Table 6. This experiment demonstrates that our approach is effective for unsupervised domain adaptation to a target domain even in the absence of source domain data.

5.4 Results on Image Classification

To demonstrate that our method also works in non-NLP domains, on ImageNet with ResNet-50, we report an increase in accuracy of 0.16. On MNIST dataset, the improvement in accuracy of our simple MLP model is 0.27.

5.5 Comparison with Existing Methods

We compare our method with several existing approaches for Self-Training, Zou et al. (2018), Malach and Shalev-Shwartz (2017), Malach and Shalev-Shwartz (2017), Ge et al. (2020) and Han et al. (2018), on SQuAD v2.0 dataset.

As shown in Table 5 our method greatly outperforms the existing approaches, giving 4 to 5 times the relative improvement compared to other methods, improving performance by $0.68/0.82$ F1/EM compared to $0.14/0.15$ F1/EM of the best performing existing approach.

5.6 Online Variant

Our approach can also be used effectively without any modifications in an online setting, where the model keeps learning continuously as inference data is fed to the model. We use a trained model to make predictions on the input inference data, and at the same time, we use the model's predictions to finetune the model. For this kind of learning, we use a constant learning rate, as the total size of inference data is unavailable. As a baseline, we use BERT + CBST (trained on SQuAD-v2.0 data) with a constant learning rate. BERT + CBST + Reptile + l^2sp (Online) clearly outperforms BERT + CBST (Online) by $0.38/0.37$ F1/EM as shown in Table 7.

We also compare the performance of our method when running in online mode for a long time on NewsQA dataset, as shown in Table 6. The performance improvement is not as large as with decreasing LR, but still results in significant performance improvements of $2.70/1.63$ F1/EM, respectively.

6 Ablation Studies

We conduct extensive ablation studies to test the effectiveness of all parts of our approach. We perform these ablations on SQuAD v2.0 with BERT-base model.

Method	F1	EM
BERT	76.14	73.14
BERT + CBST	76.22 ± 0.02	73.35 ± 0.02
BERT + CBST + l^2sp	76.24 ± 0.04	73.49 ± 0.04
BERT + CBST + Reptile	76.61 ± 0.02	73.60 ± 0.03
BERT + Reptile + l^2sp	76.47 ± 0.07	73.63 ± 0.08
BERT + CBST + Reptile + l^2sp	**76.82** ± 0.04	**73.86** ± 0.04
BERT + CBST (Online)	76.20 ± 0.01	73.28 ± 0.01
BERT + CBST + Reptile + l^2sp (Online)	**76.58** ± 0.02	**73.65** ± 0.01

Table 7: Ablation Study of our method on SQuAD v2.0 corpus, using the BERT-base-uncased model.

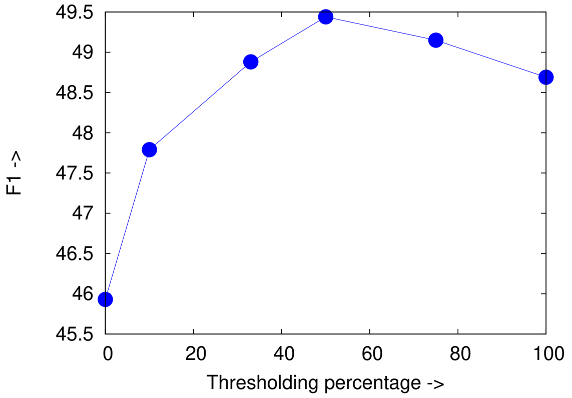


Figure 2: Ablation study of varying the thresholding percentage on NewsQA. The Y axis is F1 score, the X axis is the percentage of data left after thresholding.

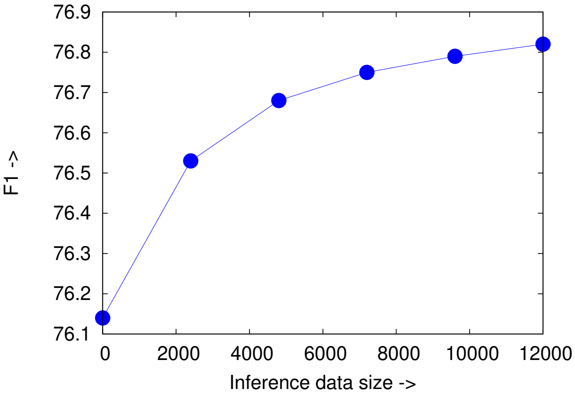


Figure 3: Effect of varying total size of Inference data on our method on SQuAD v2.0. The Y axis is F1 score, the X axis is the total amount of Inference data used.

6.1 Thresholding

In Table 7, we compare using CBST thresholding of model outputs to fine-tune the model vs. using all the data. Using CBST + Reptile + l^2sp increases scores by $0.35/0.23$ F1/EM respectively compared to using all the pseudo-labels with Reptile + l^2sp.

We further study the effect of the thresholding fraction f used to select the subset of confident data. We use the pre-trained Bert-base-uncased model, self-trained on the training set of NewsQA data with pseudo-labels, while varying f, and then evaluate on the dev set. As can be seen in Fig.2, the optimal value for thresholding is around 50%, decreasing slowly as more data (but with less confident labels) is used, and decreasing more sharply as the total filtered data used decreases.

6.2 Reptile Algorithm

Compared to using just CBST, using the Reptile Algorithm to finetune results in more performance gains of $0.58/0.37$ F1/EM, as we can see in Table 7. This effect persists irrespective of whether l^2sp or the model's default weight decay towards 0 is used. This demonstrates that the increased generalization from Reptile's meta-gradients is indeed effective in increasing model performance and robustness.

We also conduct an ablation study on the choice of number of inner steps k on the performance of our model. As shows in Table 8, the number of inner updates does not have a major impact on the results, but we advise it be kept less than or equal to 4 as higher inner steps reduce the number of outer updates (as the total number of epochs is kept constant).

6.3 Inductive Bias towards pre-trained weights

We can also see in Table 7 that l^2sp is indeed effective, and by simply biasing the model towards the pre-trained weights, we can achieve better results. This effect becomes more pronounced when the Reptile algorithm is used, with $0.21/0.26$ F1/EM improvement of CBST + Reptile + l^2sp compared to CBST + Reptile.

We also conduct an ablation study on the choice of this bias, by transfer learning on NewsQA

Num Updates	**F1**	**EM**
Baseline	76.24 ± 0.04	73.49 ± 0.04
2	76.79 ± 0.02	73.87 ± 0.02
4	76.82 ± 0.01	73.86 ± 0.02
6	76.80 ± 0.01	73.74 ± 0.02
8	76.71 ± 0.01	73.68 ± 0.03

Table 8: Ablation of choice of hyper-parameter number of inner steps k for our method CBST + Reptile + l^2sp on SQuAD with BERT-base.

l^2sp **decay**	**F1**	**EM**
Baseline	76.14	73.14
0	76.38 ± 0.00	69.87 ± 0.21
6e-4	76.54 ± 0.02	70.74 ± 0.15
2e-3	76.10 ± 0.01	73.74 ± 0.12

Table 9: Performance of BERT-base on SQuAD, after self-training on NewsQA with transfer learning with our method, for varying choices of hyper-parameter decay for l^2sp.

dataset using our method with a model trained on SQuAD, and measuring the performance on SQuAD thereafter. As shows in Table 9, l^2sp prevents the model from forgetting its performance on SQuAD. However, higher values prevent it from improving its performance on the original squad by minimizing learning on NewsQA.

6.4 Effect of Inference Data Size

In Figure 3, we vary the amount of inference data available for our model to learn from, by training a BERT-base model on varying sizes of pseudo-labelled SQuAD v2.0 dev set, while keeping f fixed at 50%. The largest increase occurs early on in the training. However, even on using the full dev set, the performance keeps improving, giving an improvement in F1 of 0.68.

7 Related Works

7.1 Pseudo-labeling

Lee (2013) proposed a simple and efficient method of semi-supervised learning for deep neural networks, in which the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously, using *pseudo-labels* created by selecting the classes which have the highest predicted probabilities as ground truth labels for unlabeled data. CBST (Zou et al., 2018) used different thresholds for pseudo-labels of different classes. Mutual Mean-teaching (Ge et al., 2020) used a moving average of two separate classifiers to refine pseudo-labels. Zheng and Yang (2020) used KL-divergence between two classifiers as a measure of classifier variance to filter incorrect pseudo-labels. Pseudo-labels and similar self-supervised techniques have grown increasingly popular, particularly when used in conjunction with extremely large unlabelled data, and was used by Noisy-Student (Xie et al., 2019) recently to achieve state-of-the-art performance on image classification.

7.2 Dynamic Evaluation

Adaptive language modelling has a long history, such as Kuhn (1988), and caching based models have resulted in improved performances over state-of-the-art, such as Merity et al. (2018). Krause et al. (2018) proposed to use dynamic evaluation adapted to recent history via a gradient descent based mechanism. However, their approach is limited to language modelling, where ground-truth labels are trivially available during inference, and does not generalize to standard classification setting.

Rahman et al. (2019) also used pseudo-labels during inference to learn, but differently from our paper, they primarily focus on a transductive zero-shot detection, and do not use our proposed meta-learning and inductive bias. Kim et al. (2019) also proposed to use pseudo-labels to learn during evaluation, but require changes to the model's training phase. Su et al. (2016) also used pseudo-labels on inference data to improve model performance, but their contributions are primarily focused on adapting Self-Training to unbalanced classes. Dynamic evaluation can be considered a form of Fast-weights (Ba et al., 2016), which unlike our approach, requires changes during the training phase.

7.3 Generic Methods for Noisy Labels

Loss correction methods such as Patrini et al. (2017) model the noise transition matrix. Other approaches try to directly correct the noisy labels, such as Veit et al. (2017), but require access to a clean set. Others directly modify the loss function to make it more stable to noisy labels, such as Generalized Cross Entropy (Zhang and Sabuncu, 2018). Other approaches, most related to our approach, refine the training strategy, such as Co-teaching (Han et al., 2018) or Mutual Mean-teaching, using two classifiers to select the data for each other.

7.4 Unsupervised Domain Adaptation

Unsupervised Domain Adaptation methods often use Adversarial Methods, such as Jiang et al. (2020), to distinguish between source and target domains. Distance based methods, such as Chen et al. (2019), aim to minimize the distribution discrepancy across different domains. Other methods such as Courty et al. (2017) rely on optimal transport between source and target domains. These methods often need access to source domain data, or modify the original model or training procedure.

7.5 Meta-Learning for Transfer Learning

Algorithms that rely on Fisher/Hessian matrices have been proposed to improve transfer learning, such as Kirkpatrick et al. (2016). Nichol et al. (2018) proposed using batched FO-MAML during training to learn better weight initialization values. Often these algorithms also use some form of Experience Replay, where saved/cached examples from previous tasks are replayed to prevent the model from forgetting. Riemer et al. (2018) proposed Meta-Experience Replay (MER), exploiting a trade-off between transfer and interference by enforcing gradient alignment across examples.

8 Conclusion

We propose a method for self-supervised learning for any classifier model during inference using the model's own predictions, adapting Reptile algorithm from meta-learning and an inductive bias for maintaining generalization while improving performance. We demonstrate the effectiveness of our method on a wide range of tasks, including SuperGLUE benchmark, question answering on SQuAD v2.0 and NewsQA, response selection on Ubuntu Dialog Corpus v2.0, and image classification on ImageNet and MNIST. Our approach consistently improves the performance of standard backbones such as BERT, Electra, and ResNet. Our method is effective for improving the performance of neural models without any changes to the underlying models, their training, or access to training data, requires minimum extra compute, and is also effective in online and transfer-learning settings.

Acknowledgments

We want to thank Mr. Wonho Ryu and Mr. Haejun Lee of Samsung Research, Seoul, Korea, for their guidance and leadership. We would also like to thank all the reviewers for their valuable feedback and suggestions.

References

Jimmy Ba, Geoffrey E. Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. 2016. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4331–4339.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST.

Satrajit Chatterjee. 2020. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chao Chen, Zhihong Chen, Boyuan Jiang, and Xinyu Jin. 2019. Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3296–3303. AAAI Press.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. 2017. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3730–3739.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, volume 3944 of *Lecture Notes in Computer Science*, pages 177–190. Springer.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23-2, pages 107–124.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society.

Li Deng. 2012. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.*, 29(6):141–142.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2185–2194. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 13042–13054.

Stanislav Fort, Pawel Krzysztof Nowak, and Srini Narayanan. 2019. Stiffness: A new perspective on generalization in neural networks. *CoRR*, abs/1901.09491.

Yixiao Ge, Dapeng Chen, and Hongsheng Li. 2020. Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL@ACL 2007 Workshop on Textual Entailment and Paraphrasing, Prague, Czech Republic, June 28-29, 2007*, pages 1–9. Association for Computational Linguistics.

Micah Goldblum, Jonas Geiping, Avi Schwarzschild, Michael Moeller, and Tom Goldstein. 2020. Truth or backpropaganda? an empirical investigation of deep learning theory. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Andrew S. Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. Semeval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Proceedings of the 6th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2012, Montréal, Canada, June 7-8, 2012*, pages 394–398. The Association for Computer Linguistics.

R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montr ́e al, Canada*, pages 8536–8546.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Xiang Jiang, Qicheng Lao, Stan Matwin, and Mohammad Havaei. 2020. Implicit class-conditioned domain alignment for unsupervised domain adaptation. *CoRR*, abs/2006.04996.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. 2019. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6091–6100. IEEE.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. 2018. Dynamic evaluation of neural sequence models. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2771–2780. PMLR.

Roland Kuhn. 1988. Speech recongnition and the frequency of recently used words: a modified markov model for natural language. In *Proceedings of the 12th International Conference on Computational Linguistics, COLING '88, Budapest, Hungary, August 22-27, 1988*, pages 348–350. John von Neumann Society for Computing Sciences, Budapest.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 2.

Hector J. Levesque. 2011. The winograd schema challenge. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Xuhong Li, Yves Grandvalet, and Franck Davoine. 2018. Explicit inductive bias for transfer learning with convolutional networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2830–2839. PMLR.

Ryan Thomas Lowe, Nissan Pow, Iulian Vlad Serban, Laurent Charlin, Chia-Wei Liu, and Joelle Pineau. 2017. Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue Discourse*, 8(1):31–65.

Eran Malach and Shai Shalev-Shwartz. 2017. Decoupling "when to update" from "how to update". In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 960–970.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2233–2241. IEEE Computer Society.

Mohammad Taher Pilehvar and José Camacho-Collados. 2019. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1267–1273. Association for Computational Linguistics.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 67–81. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language understanding paper.pdf*.

Shafin Rahman, Salman Khan, and Nick Barnes. 2019. Transductive learning for zero-shot object detection. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6081–6090. IEEE.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2018. Learning to learn without forgetting by maximizing transfer and minimizing interference. *CoRR*, abs/1810.11910.

Kyungmin Su, W. David Hairston, and Kay A. Robbins. 2016. Adaptive thresholding and reweighting to improve domain transfer learning for unbalanced data with applications to EEG imbalance. In *15th IEEE International Conference on Machine Learning and Applications, ICMLA 2016, Anaheim, CA, USA, December 18-20, 2016*, pages 320–325. IEEE Computer Society.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 191–200. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge J. Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6575–6583. IEEE Computer Society.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 3261–3275.

Qizhe Xie, Eduard H. Hovy, Minh-Thang Luong, and Quoc V. Le. 2019. Self-training with noisy student improves imagenet classification. *CoRR*, abs/1911.04252.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *CoRR*, abs/1810.12885.

Zhilu Zhang and Mert R. Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 8792–8802.

Zhedong Zheng and Yi Yang. 2020. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *CoRR*, abs / 2003.03773.

Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar, and Jinsong Wang. 2018. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 297–313. Springer.

A Statistical Significance of Results

For each dataset, we train one model on the training set, followed by five runs on the pseudo-labeled thresholded test set with varying seeds, and report the mean and standard deviation of the scores. Smaller datasets in SuperGLUE are known to have significant variation between multiple runs when fine-tuning BERT model, however, most of this variation comes from random initialization of the classification layer. In our experiments, as the model has already been fine-tuned on the train set, the only variation between runs is the order of input data. This results in an extremely small variation in score between different runs, much smaller than the performance gains observed, making the improvements statistically significant.

A.1 Significance tests

We provide below in Table 10 and Table 12 the P-values for one-sample T-test for the Table 6 and Table 7, with the null hypothesis that the scores of our results have the same mean as the baseline. Our results are significant at 99% confidence in all settings.

B Improved Generalization

The NewsQA results in Table 6 are scores on the test set, while the model was self-trained on the train set. The scores indicate that, the model does not over-fit while self-training as our approach significantly improves the scores on the test set.

Corpus	$p - value$
SQuAD v2.0 Electra (F1)	3e-5
Ubuntu Dialog v2.0	5e-5
NewsQA (F1)	1e-9
ImageNet	4e-7

Table 10: P-values for one-sample T-test with the null hypothesis for Table 6.

As a further test of improved generalization, we split the squad dev set in two equal halves, performed our self-training on one half, and evaluated on the other half. Scores in Table 11 show, self-training on one half improved generalization on the other.

Model/Approach	$F1$	EM
BERT	76.28	73.32
BERT+ours	76.40	73.45

Table 11: Results on one half of the squad-dev set.

C Links to Source code

For SuperGLUE, we use the Official Implementation for BERT available at `https://github.com/nyu-mll/jiant`, along with the default pre-trained models. For ablation tests on SQuAD, we used the official implementation and pre-trained models at `https://github.com/google-research/bert`. For Ubuntu Dialog, we used the same pre-trained models, but we implement our own classifier. For Electra, we used the pre-trained models from `https://github.com/google-research/electra`. For ResNet-50, we used Tensorflow Model Garden's official implementation as well as pre-trained model on ImageNet at `https://github.com/tensorflow/models/tree/r1.13.0/official/resnet`. For MNIST, we implemented our own MLP following `https://www.tensorflow.org/datasets/keras_example`. The Reptile+l^2sp Optimizer is trivial to implement in all of the above models following the pseudo-code from the main paper, by modifying the *Optimizer* class used for each of the models.

D Links to download data

SuperGLUE can be downloaded from `https://super.gluebenchmark.com/`. SQuAD v2.0 can be downloaded from `https://rajpurkar.github.io/SQuAD-explorer/`. Ubuntu Dialog Corpus can be generated using `https://github.com/rkadlec/ubuntu-ranking-dataset-creator`. ImageNet can be downloaded from `http://image-net.org`. MNIST can be downloaded from `http://yann.lecun.com/exdb/mnist/`. NewsQA can be downloaded from `https://www.microsoft.com/en-us/research/project/newsqa-dataset/`.

E Corresponding Validation set results for Test Set

Our only reported test scores are on MNIST, NewsQA, and Ubuntu Dialog Corpus. For MNIST, there is no official validation set. For Ubuntu Dialog Corpus, the validation score of our model is 76.44 Recall10@1, and for NewsQA, it is 49.44±0.04 F1 and 39.26±0.15 EM, respectively.

F Hyper-parameters of our approach

The hyper-parameter search bounds were chosen based on heuristic manual estimates, primarily considering the product of the LR_{inner} and LR_{outer}, compared to the model's native LR when the fraction of training steps left equals the ratio of the size of the training set to the size of the filtered inference set. Each set of hyper-parameters was run three times, and the hyper-parameter search was run in a grid. We list the hyper-parameters of our $Reptile + l^2sp$ approach in Table 13.

G Dataset descriptions

G.1 SuperGLUE

BoolQ Boolean Questions, a Question Answering (QA) dataset with short passages and yes/no questions, with data from Wikipedia and Google search engine queries.

CB Commitment Bank, consisting of passages with labels for *commitment* of speakers of clauses to said clause, framed as three-class NLI, with data from WSJ, British National Corpus and SwitchBoard. Evaluated with unweighted average F1 and accuracy.

COPA Choice of Possible Alternative, a dataset to classify the cause/effect of a given premise from two alternatives, with fully handcrafted data.

MultiRC Multi-Sentence Reading Comprehension, a QA dataset, with a list of multiple-choice possible answers for each question to a

Approaches (F1)	$p - value$
CBST+Reptile-vs-CBST	3e-7
CBST+Reptile+l2sp-vs-CBST+Reptile	1e-5
CBST+Reptile+l2sp(online)-vs-CBST(online)	2e-6
CBST+Reptile+l2sp-vs-Baseline(One-sample)	1e-6

Table 12: P-values for one-sample T-test with the null hypothesis for Table 7.

Corpus	LR_{outer}	LR_{inner}	$inner\_steps$	l^2sp
SQuAD v2.0 Bert	[**1e-2**, 3e-3]	1e-5	4	1e-2
SQuAD v2.0 Electra	3e-3	[**1e-5**, 1e-6]	4	[1e-2, **5e-2**]
Ubuntu Dialog v2.0	[1e-2, **3e-3**]	[**1e-5**, 1e-6]	4	[5e-2, **1e-1**]
NewsQA	1e-2	1e-5	4	[**2e-3**, 1e-2]
ImageNet	[3e-2, **1e-2**]	1e-4	4	[**1e-1**, 5e-1]
MNIST	1e-1	1e-4	4	1e-1
BoolQ	1e-2	1e-6	4	0.4
CB	1e-2	1e-6	4	0.4
COPA	1e-2	1e-6	2	0.4
MultiRC	1e-1	1e-4	2	0.4
ReCoRD	1e-2	1e-6	4	0.4
RTE	1e-2	1e-6	2	0.4
WiC	1e-2	1e-6	2	0.4

Table 13: Hyper-parameters for all Datasets. Best performing parameters are in **bold**.

paragraph. Evaluated with F1 over all answer-options($F1_a$), and exact match of each question's set of answers(EM).

ReCoRD Reading Comprehension with Commonsense Reasoning, a QA dataset consisting of articles and Cloze-style questions with a masked entity, scored on predicting the masked entity from the entities in the article, with data from CNN and Daily Mail. Scored with token-level F1 and EM.

RTE Recognizing Textual Entailment, as binary classification of *entailment* or *not entailment*, with data from Wikipedia and news.

WiC Word-in-Context, a word sense disambiguation (WSD) dataset, tasked with binary classification of sentence pairs based on the sense of a common polysemous word. Data is from WordNet and Wiktionary.

WSC Winograd Schema Challenge, a coreference resolution task on resolving pronouns to a list of noun phrases. As the models we tested only predicted the majority class, we omit this dataset.

G.2 SQuAD v2.0

The Stanford Question Answering Dataset v2.0 is a popular span-style QA dataset, consisting of passages from Wikipedia, labelled by annotators for questions on the passages and corresponding answer spans, along with unanswerable questions as well. This dataset is evaluated with F1 and EM scores of predicted answer spans.

G.3 Ubuntu Dialog Corpus v2.0

The Ubuntu Dialog Corpus is a large-scale corpus of multi-turn real human conversations mined from Ubuntu IRC chat logs, with only two participants per conversation. Each conversation is annotated with the next utterance (response) following the conversation, and the task is to select the best response given a list of possible distractor responses. The dataset is evaluated with Recall score of picking the correct response out of 10 possible responses, $Recall10@1$.

G.4 NewsQA

NewsQA is a span-style QA dataset, consisting of crowd-sourced questions on CNN news articles and their corresponding answer spans, along with unanswerable questions. This datasets is evaluated with F1 and EM scores of predicted answer spans.

G.5 MNIST

MNIST is a popular image classification dataset, consisting of normalized and anti-aliased 28x28 scans of handwritten numerical digits. While the dataset has long been *solved*, it nevertheless serves as a useful dataset to compare simpler architectures.

G.6 ImageNet

ImageNet is a large-scale dataset for image classification, consisting of 1.2M training samples along with their corresponding class labels. It is often the de-facto dataset when comparing Image Classification models.

H Expected validation performance

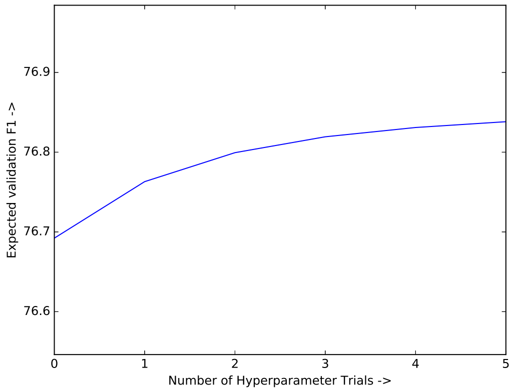


Figure 4: Expected Validation performance of our hyper-parameter searches, for SQuAD dataset with BERT-base model.

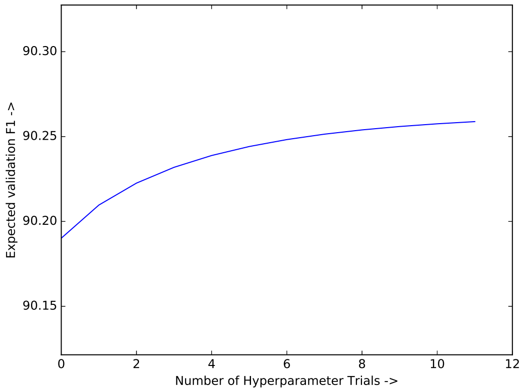


Figure 5: Expected Validation performance of our hyper-parameter searches, for SQuAD dataset with Electra-large model.

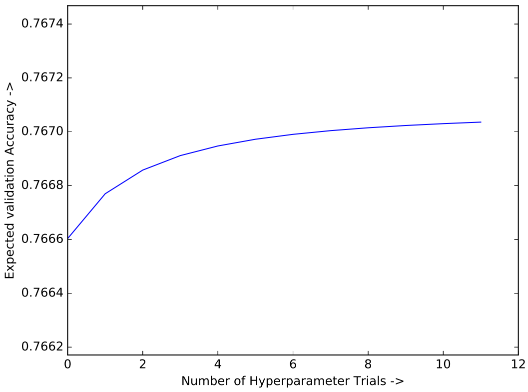


Figure 6: Expected Validation performance of our hyper-parameter searches, for ImageNet dataset with ResNet-50 model.

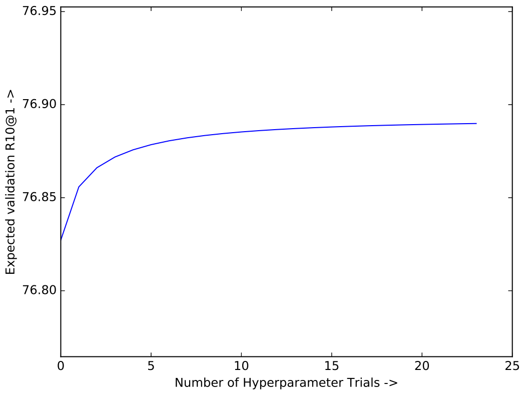


Figure 7: Expected Validation performance of our hyper-parameter searches, for Ubuntu dialog corpus with BERT-base model.

We provide the expected validation performance for all the datasets we ran hyper-parameter searches on, as described in (Dodge et al., 2019).

RESPER : Computationally Modelling Resisting Strategies in Persuasive Conversations.

Ritam Dutt[*,1], **Sayan Sinha**[*,2], **Rishabh Joshi**[1], **Surya Shekhar Chakraborty**[3],
Meredith Riggs[1], **Xinru Yan**[1], **Haogang Bao**[1], **Carolyn Penstein Rosé**[1]

[1]Carnegie Mellon University, [2]Indian Institute of Technology Kharagpur, [3]Zendrive Inc
{rdutt,rjoshi2,mriggs,xinruyan,haogangb,cprose}@cs.cmu.edu,
sayan.sinha@iitkgp.ac.in, suryaschak@gmail.com

Abstract

Modelling persuasion strategies as predictors of task outcome has several real-world applications and has received considerable attention from the computational linguistics community. However, previous research has failed to account for the resisting strategies employed by an individual to foil such persuasion attempts. Grounded in prior literature in cognitive and social psychology, we propose a generalised framework for identifying resisting strategies in persuasive conversations. We instantiate our framework on two distinct datasets comprising persuasion and negotiation conversations. We also leverage a hierarchical sequence-labelling neural architecture to infer the aforementioned resisting strategies automatically. Our experiments reveal the asymmetry of power roles in non-collaborative goal-directed conversations and the benefits accrued from incorporating resisting strategies on the final conversation outcome. We also investigate the role of different resisting strategies on the conversation outcome and glean insights that corroborate with past findings. We also make the code and the dataset of this work publicly available at `https://github.com/americast/resper`.

1 Introduction

Persuasion is pervasive in everyday human interactions. People are often exposed to scenarios that challenge their existing beliefs and opinions, such as medical advice, election campaigns, and advertisements (Knobloch-Westerwick and Meng, 2009; Bartels, 2006; Speck and Elliott, 1997). Of late, huge strides have been taken by the Computational Linguistics community to advance research in persuasion. Some seminal works include identifying persuasive strategies in text (Yang et al., 2019) and conversations (Wang et al., 2019), investigating the interplay of language and prior beliefs on successful persuasion attempts (Durmus and Cardie, 2018; Longpre et al., 2019), and generating persuasive dialogues (Munigala et al., 2018).

However, a relatively unexplored domain by the community is the investigation of resisting strategies employed to foil persuasion attempts. As succinctly observed by Miller (1965): “In our daily lives we are struck not by the ease of producing attitude change but by the rarity of it.” Several works in cognitive and social psychology (Fransen et al., 2015a; Zuwerink Jacks and Cameron, 2003) have put forward different resisting strategies and the motivations for the same. However, so far, there has not been any attempt to operationalise these strategies from a computational standpoint. We attempt to bridge this gap in our work.

We propose a generalised framework, grounded in cognitive psychology literature, for automatically identifying resisting strategies in persuasion oriented discussions. We instantiate our framework on two publicly available datasets comprising persuasion and negotiation conversations to create an annotated corpus of resisting strategies.

Furthermore, we design a hierarchical sequence modelling framework, that leverages the conversational context to identify resisting strategies automatically. Our model significantly outperforms several neural baselines, achieving a competitive macro-F1 score of 0.56 and 0.66 on the persuasion and negotiation dataset, respectively.

We refer to our model as RESPER, which is not only an acronym for **Res**isting **Per**suasion, but also a play on the word ESPer: a person with extrasensory abilities. The name is apt since we observe that incorporating such resisting strategies could provide additional insight on the outcome of the conversation. In fact, our experiments reveal that the resisting strategies are better predictors of conversation success for the persuasion dataset than the

---

[*] denotes equal contribution

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 78–90
April 19 - 23, 2021.

strategies employed by the persuader. We also observe that the buyer's strategies are more influential in negotiating the final price. Our findings highlight the asymmetric nature of power roles arising in non-collaborative dialogue scenarios and form motivation for this work.

2 Related Works

The use of persuasion strategies to change a person's view or achieve a desired outcome finds several real-world applications, such as in election campaigns (Knobloch-Westerwick and Meng, 2009; Bartels, 2006), advertisements (Speck and Elliott, 1997), and mediation (Cooley, 1993). Consequently, several seminal NLP research have focused on operationalising and automatically identifying persuasion strategies (Wang et al., 2019), propaganda techniques (Da San Martino et al., 2019), and negotiation tactics (Zhou et al., 2019), as well as the impact of such strategies on the outcome of a task (Yang et al., 2019; He et al., 2018; Joshi et al., 2021). However, there is still a dearth of research from a computational linguistic perspective investigating resisting strategies to foil persuasion.

Resisting strategies have been widely discussed in literature from various aspects such as marketing (Heath et al., 2017), cognitive psychology (Zuwerink Jacks and Cameron, 2003), and political communication (Fransen et al., 2015b) . Some notable works include the identification and motivation of commonly-used resisting strategies (Fransen et al., 2015a; Zuwerink Jacks and Cameron, 2003), the use of psychological metrics to predict resistance (San José, 2019; Ahluwalia, 2000), and the design of a framework to measure the impact of resistance (Tormala, 2008). However, these works have mostly relied on qualitative methods, unlike ours, which adopts a data-driven approach. We propose a generalised framework to characterise resisting strategies and employ state-of-the-art neural models to infer them automatically. Thus our work can be considered complementary to past research.

The closest semblance to our work in NLP literature ties in with argumentation, be it essays (Carlile et al., 2018), debates (Cano-Basave and He, 2016), or discussions on social media platforms (Al-Khatib et al., 2018; Zeng et al., 2020). Such works have revolved mostly on analysing argumentative strategies and their effect on others.

Recently, Al Khatib et al. (2020) demonstrated that incorporating the personality traits of the resistor was influential in determining their resistance to persuasion. Such an observation acknowledges the power vested in an individual to resist change to their existing beliefs. Our work exhibits significant departure from this because we explicitly characterise the resisting strategies employed by the user. Moreover, our work focuses on the general domain of non-collaborative task-oriented dialogues, where several non-factual resisting strategies are observed, making it distinctly different from argumentation (Galitsky et al., 2018). We assert that focusing on both parties is imperative to get a complete picture of persuasive conversations.

3 Framework

In this section, we describe the datasets, the resisting strategies employed, and the annotation framework to instantiate the strategies.

3.1 Dataset Employed

We choose persuasion-oriented conversations, rather than essays or advertisements (Yang et al., 2019), since we can observe how the participants respond to the persuasion attempts in real-time. To that end, we leverage two publicly available corpora on persuasion (Wang et al., 2019) and negotiation (He et al., 2018). We refer to these datasets as "Persuasion4Good" or P4G and "Craigslist Bargain" or CB hereafter.

P4G comprises conversational exchanges between two anonymous Amazon Mechanical Turk workers with designated roles of the persuader, ER and persuadee, EE. ER had to convince EE to donate a part of their task earnings to the charity *Save the Children*. We investigate the resisting strategies employed only by EE in response to the donation efforts. We emphasise that the conversational exchanges are not scripted, and the task is set up so that a part of EE's earnings is deducted if they agree to donate. Since there is a monetary loss at stake for EE, we expect them to resist.

CB consists of simulated conversations between a buyer (BU) and a seller (SE) over an online exchange platform. Both are given their respective target prices and employ resisting strategies to negotiate the offer.

We choose these datasets since they involve non-collaborative goal-oriented dialogues. As a result, we can definitively assess the impact of different resisting strategies on the goal.

Resisting Strategy	Persuasion (P4G)	Negotiation (CB)
Source Derogation	Attacks/doubts the organisation's credibility. *My money probably won't go to the right place*	Attacks the other party or questions the item. *Was it new denim, or were they someone's funky old worn out jeans?*
Counter Argument	Argues that the responsibility of donation is not on them or refutes a previous statement. *There are other people who are richer*	Provides a non-personal argument/factual response to refute a previous claim or to justify a new claim. *It may be old, but it runs great. Has lower mileage and a clean title.*
Personal Choice	Attempts to saves face by asserting their personal preference such as their choice of charity and their choice of donation. *I prefer to volunteer my time*	Provides a personal reason for disagreeing with the current situation or chooses to agree with the situation provided some specific condition is met. *I will take it for $300 if you throw in that printer too.*
Information Inquiry	Ask for factual information about the organisation for clarification or as an attempt to stall. *What percentage of the money goes to the children?*	Requests for clarification or asks additional information about the item or situation. *Can you still fit it in your pocket with the case on?*
Self Pity	Provides a self-centred reason for not being able/willing to donate at the moment. *I have my own children*	Provides a reason (meant to elicit sympathy) for disagreeing with the current terms. *$130 please I only have $130 in my budget this month.*
Hesitance	Attempts to stall the conversation by either stating they would donate later or is currently unsure about donating. *Yes, I might have to wait until my check arrives.*	Stalls for time and is hesitant to commit; specifically, they seek to further the conversation and provide a chance for the other party to make a better offer. *Ok, would you be willing to take $50 for it?*
Self-assertion	Explicitly refuses to donate without even providing a factual/personal reason *Not today*	Asserts a new claim or refutes a previous claim with an air of finality/ confidence. *That is way too little.*

Table 1: Framework describing the resisting strategies for persuasion (P4G) and negotiation (CB) datasets. We emphasise that Information Inquiry is not a resisting strategy for CB. Examples of each strategy are italicised.

Properties	**P4G**	**CB**
# of conversations	530	800
Max # of utterances/conversation	76	44
Avg # of utterances/conversation	36.34	11.94
Max # of tokens/utterance	90	93
Avg # of tokens/utterance	11.03	14.62
Vocabulary size	6137	5370

Table 2: Description for the Persuasion (P4G) (Wang et al., 2019) and Negotiation (CB) (He et al., 2018) datasets

3.2 Framework Description

In this subsection, we briefly describe the resisting strategies commonly referenced in social and cognitive psychology literature. This enables us to design a unified framework for the two datasets, built upon common underlying semantic themes. Fransen et al. (2015a) identified 4 major clusters of resisting strategies, namely **contesting** (Wright, 1975; Zuwerink Jacks and Cameron, 2003; Abelson and Miller, 1967), **empowerment** (Zuwerink Jacks and Cameron, 2003; Sherman and Gorkin, 1980), **biased processing** (Ahluwalia, 2000), and **avoidance** (Speck and Elliott, 1997). Each individual category can be subdivided into finer categories showcased in italics henceforth.

Contesting refers to attacking either the source of the message (*Source Derogation*) or its content (*Counter Argumentation*). A milder form of contesting involves seeking clarification or information termed *Information Inquiry*. Prior work has shown a positive association between working knowledge and one's ability to resist persuasion (Wood and Kallgren, 1988; Luttrell and Sawicki, 2020). Therefore, *Information Inquiry* can be interpreted as a form of resistance where the resistor seeks to satisfy their doubts because they are sceptical of the persuader's intents or messages. This is prominent in certain conversations in P4G where a sceptical EE questions the charity's legitimacy.

Empowerment strategies encompass reinforcing one's personal preference to refute a claim (*Attitude Bolstering*) (Sherman and Gorkin, 1980), attempting to arouse guilt in the opposing party (*Self Pity*) (Vangelisti et al., 1991; O'Keefe, 2002), stating one's wants outright (*Self Assertion*) (Zuwerink Jacks and Cameron, 2003), or seeking validation from like-minded people (*Social Validation*) (Fransen et al., 2015a). Overall, empowerment strategies drive the discussion towards the resistor's self as opposed to attacking the persuader.

Biased processing mitigates external persuasion by selectively processing information that conforms with one's opinion or beliefs (Fransen et al., 2015a). For simplicity, we subsume strategies that denote personal preference, namely *Attitude Bolstering* and *Biased Processing*, into a unified category *Personal Choice*. We refrain from incorporating *Self Assertion* into the *Personal Choice* category since it deals with bolstering one's confidence and not one's opinions or attitudes. The subtle difference is highlighted in Table 1.

Avoidance strategies distance the resistor from persuasion, either physically or mechanically, or refuse to engage in topics that induce cognitive dissonance (Fransen et al., 2015a). However, in the context of task-oriented conversations, wherein participants are expected to further a goal, avoidance often manifests as *Hesitance* to commit to the current situation.

We identify seven major resisting strategies across the datasets, namely *Source Derogation, Counter Argumentation, Information Inquiry, Personal Choice, Self Pity, Hesitance,* and *Self Assertion*. Since the datasets comprise two-party conversations between strangers, *Social Validation*, which requires garnering the support of others, was absent. We now describe how these resisting strategies were instantiated in the following section.

3.3 Instantiating the Resistance Framework

We emphasise that although the description and meaning of a strategy remain the same across the two datasets, their semantic interpretation depends on the context. For example, scepticism towards the charity in P4G and criticism of the product in CB are instances of *Source Derogation*. This is because ER represents the charity, whereas the seller is being accused of selling an inferior product. Likewise, we instantiate the predicates for the remaining six resisting strategies for the two datasets, with examples in Table 1.

We label the utterances of persuadee (EE) in P4G and the buyers (BU) and sellers (SE) in CB with at least one of the seven corresponding resisting strategies, or 'Not-A-Strategy' if none applies. The 'Not-A-Strategy' label includes greetings, off-task discussions, agreement, compliments, or other tokens of approval. We acknowledge that an utterance can have more than one resisting strategy embedded in it. For example, the utterance "The price is slightly high for used couches, would you come down to 240 if I also picked them up?", is an instance of both *Personal Choice* and *Counter Argumentation*.

We also note that ***Information-Inquiry* is not a resisting strategy for CB** since asking additional information/clarification is an expected behaviour before finalising a deal. We keep the label nevertheless to show comparison with P4G. We present the flowchart detailing the annotation framework in Figure 3 of Appendix.

3.4 Annotation Procedure and Validation

We describe the annotation procedure for both the CB and P4G dataset here and its subsequent validation. For CB, three authors independently annotated five random conversations adhering to the flowchart. If the conversations chosen were simple or had few labels, a new set of 5 conversations were taken up. This constitutes one round. After each round, the Fleiss Kappa score was computed, and the authors discussed to resolve the disagreements and revise the flowchart. Then began the next round on a new set of 5 random conversations. For CB, 5 rounds of revision were carried out over 24 conversations, until a high Fleiss kappa (0.790) (Fleiss, 1971) was obtained. Finally, the three authors independently went ahead and annotated approximately 250 distinct conversations, yielding a corpus of 800 CB conversations. Our annotation procedure requires a rigorous reliable refinement phase but a comparatively faster annotation phase by dividing the annotation between the authors. Thus the conversations annotated by each author were mutually exclusive. Similarly, for P4G dataset, four authors annotated 3 conversations per round, since a conversation in P4G was comparatively longer. 4 rounds of revision across 12 conversations was done to achieve the final kappa-score of 0.787. The four authors then went ahead and divided the task of annotating the 500 conversations amongst themselves. We show an annotated conversation snippet for the two datasets in Table 3.

3.5 Dataset Statistics

The P4G and CB datasets comprise 530 and 800 labelled conversations, respectively, spanning an average of 37 and 12 utterances per conversation.

Role	Text	Strategy
	Negotiation (CB)	
SE	I have a wonderful phone for you if you are interested.	No Strategy
BU	I am interested. Did you just buy it?	Info inquiry
SE	I bought it two weeks ago but it just wasn't what I needed anymore.	No Strategy
BU	Would you be willing to work with the price?	Hesitance
SE	Yes we can negotiate.	No Strategy
BU	If I come today would you accept $56 I can bring it now?	Per Choice
SE	How about 65 and I can deliver it to you now?	Per Choice
BU	Can you go $60 Kind of all I have right now ?	Self Pity
SE	Yes I can.	No Strategy
	Persuasion (P4G)	
ER	Hello, Save the Children looks like an interesting organisation.	-
EE	i would like to know more about it	Info Inquiry
..	..	..
EE	thanks i will definitely check it out	Hesitance
ER	They also promote children's rights and provide relief when needed.	-
EE	and where does the money go if i do donate ?	Info Inquiry
EE	Straight to the organisation?	Info Inquiry
ER	Yes, it goes straight to the organisation, where it can be used to help many children.	-
EE	because some organisations do not divide the money properly	Source Derogation
ER	This organisation has been checked by some groups, and they divide the money properly.	-
..	..	..
EE	I will certainly consider it	No Strategy

Table 3: Examples of annotation snippets for the Persuasion (P4G) and Negotiation (CB). The utterances of the EE and the SE are highlighted in cyan. Some strategies are shortened, like Info Inquiry, and Per Choice for Information Inquiry and Personal Choice.

The datasets cover two distinct persuasion scenarios and also illustrate the rights and obligations shown by the participants. For example, in P4G, EE comes into the interaction blind and is unaware of the donation attempt. We encounter several conversations where EE is willing to donate since it resonates with their beliefs, and no resisting strategies are observed. However, for CB, the participants received prior instructions to negotiate a deal, and hence resisting strategies were more prominent. We present the frequency distribution of the seven strategies in Table 4. We observe that the distributions of strategies are skewed for both the datasets and is more pronounced for P4G, where 'Not-A-Strategy' accounts for the lion's share. We also see that the buyer exhibits more resisting strategies than the seller highlighting the asymmetric role of the two participants.

Nevertheless, we reiterate that the resisting strategies we propose are applicable for both the domains. In the next section, we propose the framework to infer such strategies automatically.

Strategy	Persuasion (P4G)	Negotiation (CB)	
	EE	BU	SE
Source Derogation	2.16	7.61	0.44
Counter Argument	2.28	3.74	6.06
Personal Choice	2.52	9.43	8.49
Information Inquiry	7.19	18.27	0.38
Self Pity	1.58	4.66	0.34
Hesitance	1.76	15.78	9.14
Self-assertion	0.94	2.20	5.05
Not a strategy	81.56	38.30	70.09

Table 4: Proportion of resisting strategies (in %)for the Persuasion (P4G) and Negotiation (CB) dataset. The strategies are observed only for the persuadee (EE) in P4G and for both buyer (BU) and seller (SE) in CB.

4 Methodology

In this section, we describe the methodology adopted for inferring the resisting strategies in persuasion dialogues and how they can be leveraged to determine the dialogue's outcome.

4.1 Resisting Strategy prediction

We model the task of identifying resisting strategies as a sequence labelling task. We assign each utterance in the dialogues with a label representing either one of the seven resisting strategies or *Not-A-Strategy*.

Since the resisting strategies, by definition, occur in response to the persuasion attempts, our model architecture needs to be cognizant of the conversational history. To that end, we adopt a hierarchical neural network architecture, similar to Jiao et al. (2019), to infer the corresponding resisting strategy. The architecture leverages the previous conversational context in addition to the current contextualised utterance embedding. Our choice is motivated by the recent successes of hierarchical sequence labelling frameworks in achieving state-of-the-art performance on several dialogue-oriented tasks. Some myriad examples include emotion recognition (Majumder et al., 2019; Jiao et al., 2019), dialogue act classification (Chen et al., 2018; Raheja and Tetreault, 2019), face act prediction (Dutt et al., 2020), open domain chit-chat (Zhang et al., 2018; Kumar et al., 2020) and the like. We hereby adopt this as the foundation architecture for our work and refer to our instantiation of the architecture as RESPER.

Architecture of RESPER: An utterance u_j

We acknowledge that an utterance can have multiple labels. However, such utterances comprise only 1.2% and 3.85% of the P4G and the CB datasets, respectively. In such cases, the label is randomly selected.

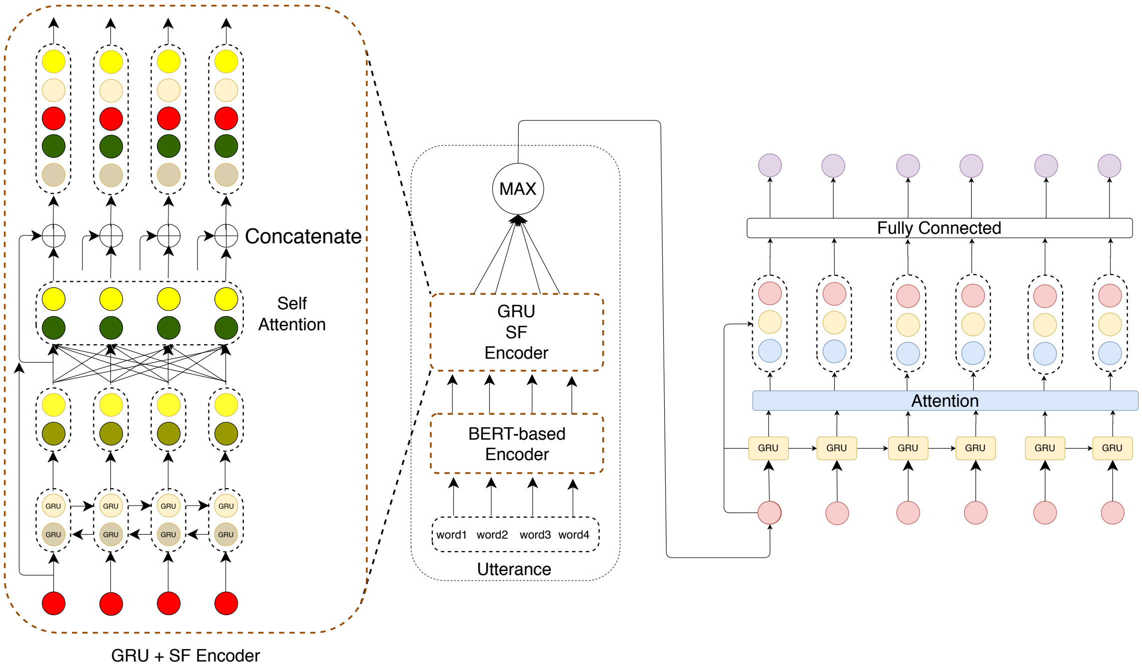


Figure 1: A diagram illustrating how ResPer works. The encoder shown on the left takes the BERT representations of a token as input and passes it through a BiGRU layer followed by Self Attention. The outputs from BERT, BiGRU and self-attention are then concatenated to form the output. Max-pooling over this output yields the corresponding utterance embedding. This utterance representation is passed through a uni-directional GRU followed by Masked-Self-Attention and fusion to yield the contextualised utterance embedding.

is composed of tokens $[w_0, w_1, ..., w_K]$ represented by their corresponding embeddings $[e(w_0), e(w_1), ..., e(w_K)]$. In ResPer, we obtain these using a pre-trained BERT model (Devlin et al., 2019). We pass these contextualised word representations through a bidirectional GRU to obtain the forward $\overrightarrow{h_k}$ and backward $\overleftarrow{h_k}$ hidden states of each word, before passing them into a Self-Attention layer. This gives us the corresponding attention outputs, $\overrightarrow{ah_k}$ and $\overleftarrow{ah_k}$ as described below.

$$\overrightarrow{h_k} = \text{GRU}\left(e\left(w_k\right), \overrightarrow{h_{k-1}}\right)$$
$$\overleftarrow{h_k} = \text{GRU}\left(e\left(w_k\right), \overleftarrow{h_{k+1}}\right)$$
$$\overrightarrow{ah_k} = SelfAttention(\overrightarrow{h_k})$$
$$\overleftarrow{ah_k} = SelfAttention(\overleftarrow{h_k})$$

Finally, we concatenate the contextualised word embedding with the GRU hidden states and Attention outputs in the *fusion layer* to obtain the final representation of the word $e_c(w_k)$. We represent the bias as b_w. Here, We perform max-pooling over the fused word embeddings to obtain the j^{th} utterance embedding, $e(u_j)$.

$$e_c(w_k) = \tanh(W_w[\overrightarrow{ah_k}; \overrightarrow{h_k}; e(w_k); \overleftarrow{h_k}; \overleftarrow{ah_k}] + b_w)$$
$$e(u_j) = \max(e_c(w_1), e_c(w_2), ...e_c(w_K))$$

We use a unidirectional GRU and Masked Self-Attention to encode conversational context, to ensure that the prediction for the j^{th} utterance is not influenced by future utterances. Similarly, we calculate the contextualized representation of an utterance $e_c(u_j)$ using the conversation context. We pass $e(u_j)$ through a uni-directional GRU that yields the forward hidden state $\overrightarrow{H_j}$. Masked Self-Attention over the previous hidden states, yields $\overrightarrow{AH_j}$. We fuse $e(u_j)$, $\overrightarrow{H_j}$ and $\overrightarrow{AH_j}$ before passing it through a linear layer with tanh activation to obtain $e_c(u_j)$.

We project the final contextualised utterance embedding $e_c(u_j)$ onto the state space of resisting strategies. We apply softmax to obtain a probability distribution over the strategies, with Negative Log-Likelihood (NLL) as the loss function to obtain the strategy loss.

4.2 Conversation Outcome prediction

We further investigate the impact of resisting strategies on the outcome of the conversation. We represent a strategy as a fixed dimensional embedding initialised at random. We subsequently encode a sequence of strategies by passing them through a unidirectional GRU to obtain a final representation for the sequence. We project the representation onto a binary vector which encodes for the conversation outcome. We apply softmax with NLL across all the conversations to obtain the outcome prediction loss.

5 Experiments

In this section, we describe the baselines and evaluation metrics. We present the experimental details of our model in Table 5.

5.1 Baselines

Resisting strategy prediction: We experiment with standard neural baselines for text classification, which have also been used in classifying persuasion strategies, namely CNN (Kim, 2014; Wang et al., 2019) and BiGRU (Yang et al., 2019). To ensure a fair comparison, we introduce pre-trained BERT-embeddings (Devlin et al., 2019) as input to the baselines, henceforth denoted as BERT-CNN and BERT-BiGRU. Furthermore, to inspect the impact of conversational history, we remove the conversational GRU from RESPER such that the utterance embedding $e(u_j)$ is directly used for prediction. We refer to this architecture as BERT-BiGRU-sf, since it employs self-attention(s) and fusion (f) on top of BERT-BiGRU. Finally, we experiment with the best performing HiGRU-sf model of Jiao et al. (2019) as another baseline.

Conversation success prediction: The notion of conversation success depends on the choice of dataset. For P4G, we consider the resisting strategies to be successful if the persuadee (EE) refused to donate to charity. For CB, we adopt the same notion of success as Zhou et al. (2019), namely when the seller (SE) can sell at a price greater than the median sale-to-list ratio r.

$$r = \frac{\text{sale price} - \text{buyer target price}}{\text{listed price} - \text{buyer target price}} \quad (1)$$

To observe the effect of conversation success, we experiment with strategies of both the parties involved. For P4G, we encode separately (i) the persuasion strategies of ER as identified by Wang et al. (2019), (ii) the resisting strategies employed by EE and (iii) both the persuasion and resisting strategies. Likewise, for CB, we encode the resisting strategies of only (i) the buyer (BU) (ii) the seller (SE) (iii) both. These experiments would enable us to investigate which party has a greater influence on conversation success.

Hyper-parameter	**Search space**	**Final Value**
learning-rate (lr)	1e-3 to 1e-5	1e-4
Batch-size	-	1 conversation
#Epochs	< 100	30.8, 22
lr-decay	-	0.5 every 20 epochs
d_{h1}	-	1024
d_{h2}	-	300

Table 5: Here we describe the search-space of all the hyper-parameters used in our experiments and describe the search space we used to find the hyper-parameters. d_{h1}, d_{h2} represents the hidden dimensions of the Utterance GRU and the Conversation GRU.

5.2 Evaluation metrics

We adopt the same evaluation procedure for both the resisting strategy and the conversation outcome prediction task across the datasets. In either case, we perform five-fold cross-validation due to paucity of annotated data. We report performance in terms of the weighted and macro F1-scores across the five folds. Our choice of the metric is motivated by the high label imbalance, as observed in Table 4.

6 Results

In this section, we answer the following :

Q1. How well does RESPER identify resisting strategies for Persuasion and Negotiation?

Q2. Are resisting strategies good predictors of conversation success? What insights can one glean from the results?

6.1 Predicting resisting strategies

We present the results for the automated identification of resisting strategies in Table 6. We observe that all the models achieve a comparatively lower performance on P4G, mainly due to the higher proportion of 'Not-a-Strategy' labels for the latter. We gauge the benefits of incorporating conversational context by the significant improvement of Macro F1 score by 0.036 and 0.011 for P4G and CB respectively. In fact, RESPER outperforms all the proposed baselines significantly.

Error Analysis: We present the confusion matrix for predicting resisting strategies using RESPER on the Persuasion (P4G) and Negotiation (CB)

Weighted F1 Scores are calculated by taking the average of the F1 scores for each label weighted by the number of true instances for each label.

We estimate the statistical significance using the paired bootstrapped test of Berg-Kirkpatrick et al. (2012), due to the small number of data (Dror et al., 2018).

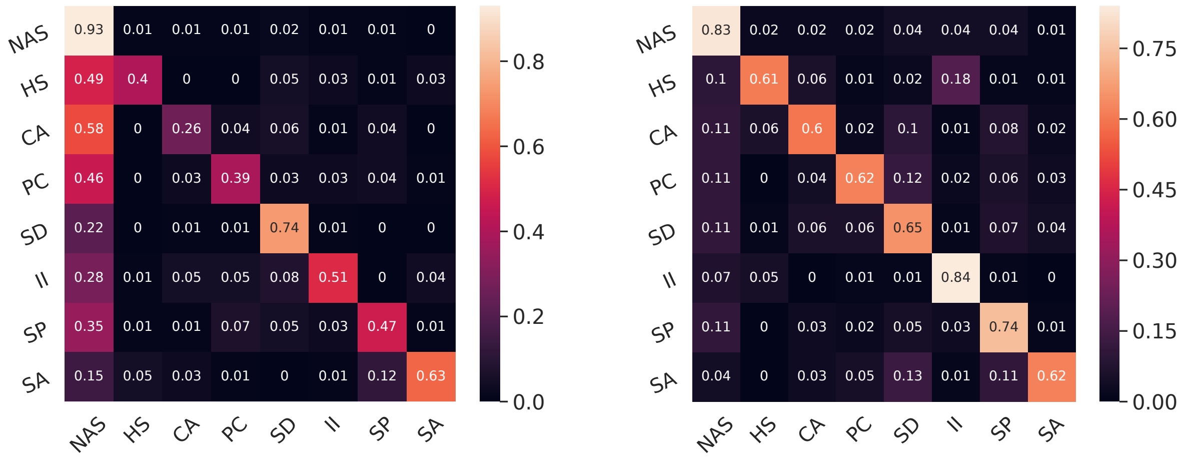


Figure 2: Confusion matrix for resisting strategies for the Persuasion (P4G) and Negotiation (CB) datasets on the left and right respectively. Each resisting strategy is represented as its initial (**S**elf **P**ity) as SP. True and Predicted Labels have been plotted on the X-axis and the Y-axis respectively.

Model	Persuasion (P4G)		Negotiation (CB)	
	M-F1	W-F1	M-F1	W-F1
CNN	0.261	0.757	0.560	0.706
BERT + CNN	0.508	0.819	0.651	0.751
HiGRU-sf	0.446	0.788	0.605	0.734
BERT + BiGRU	0.514	0.815	0.647	0.747
BERT + BiGRU-sf	0.522	0.814	0.649	0.750
RESPER	**0.558**	**0.828**	**0.662**	**0.767**

Table 6: Results of RESPER and other baselines on the resistance strategy prediction task on the Persuasion and CB dataset. The metrics used for evaluation are Macro F1 and Weighted F1 represented as M-F1 and W-F1 respectively. The best results are in bold.

Persuasion (P4G)			Negotiation (CB)		
User	Macro-F1	W-F1	User	Macro-F1	W-F1
ER	0.588	0.620	BU	**0.618**	**0.640**
EE	0.618	0.640	SE	0.462	0.508
Both	**0.646**	**0.671**	Both	0.605	0.626

Table 7: We observe the impact of incorporating sequence of strategies on conversation outcome prediction in terms of Macro-F1 and Weighted-F1 score. For P4G, we observe strategies of the persuader (ER), persuadee (EE) and both. For CB, we observe strategies of the buyer (BU), seller (SE) and both.

datasets in Figures 2(a) and 2(b) respectively. We observe that most classification errors occur when a resisting strategy is incorrectly inferred as 'Not-A-Strategy'. The effect is more prevalent for P4G since 'Not-A-Strategy' comprises 80% of all annotated labels. Other notable instances of misclassification for P4G occurs when *Self Assertion* is predicted as *Self Pity* since both strategies refer to one's self. These strategies occur so infrequently (see Table 4) that the models lack sufficient information to distinguish between the two categories. Likewise, for the CB corpus, *Hesitance* utterances which constitute a price request, are often posed as questions. This causes the model to predict the strategy as *Information Inquiry* instead. *Self Assertion* is often incorrectly marked as *Source Derogation* possibly because it often takes a firm stance, and is likely to disparage the other party in the process, thereby confusing the model.

6.2 Conversation Outcome Prediction

We observe how the sequence of strategies adopted by the two participants have a disproportionate impact on the final conversation outcome in Table 7. It is interesting to note that the resisting strategies for the persuadee have a greater effect on the conversation outcome (macro-F1 score of 0.62) than the persuasion strategies themselves (macro-F1 score of 0.59). Moreover, incorporating both the persuasion and resisting strategies boosts the prediction performance even further to 0.65.

We also observe an asymmetry in the roles of the buyer (BU) and the seller (SE) for the CB dataset. We observe that BU's strategies are significantly more effective in deciding the conversation outcome, probably because buyers demonstrate a higher number of resisting strategies. These experiments highlight the importance of incorporating resisting strategies to gain a complete picture.

6.3 Comparative Analysis of Strategies

Emboldened by the success of resisting strategies to infer the conversational outcome, we probe deeper to investigate the impact of individual strategies. We apply logistic regression with the frequency of strategies, of either participant, as the features while the outcome variable denotes conversation success. We observe the coefficients of the strategies to infer their correlation with conversation success and their corresponding p-values to determine whether the correlation was indeed statistically significant. Our procedure follows previous work in identifying influential persuasion strategies (Yang et al., 2019; Wang et al., 2019). We present the results of this analysis in Table 8.

	Persuasion (P4G)	Negotiation (CB)	
Strategy	EE	BU	SE
Not-A-Strategy	-0.008	0.287**	-0.138
Hesitance	0.344	0.328*	0.266
Counter Argument	-0.014	-0.256	0.429*
Personal Choice	0.153	0.126	0.164
Information Inquiry	0.180*	0.091	-0.704
Source Derogation	0.043	0.052	-0.455
Self Pity	0.103	0.081	-0.314
Self Assertion	0.843*	-0.576*	-0.040

Table 8: Coefficients of the different persuasion strategies corresponding to the persuadee, EE in Persuasion and the buyer, BU, and seller, SE in Negotiation. A value of * and ** means the strategy is signficant with p-value ≤ 0.05 and 0.01 respectively.

For P4G, all the resisting strategies for persuasion apart from *Counter-Argumentation* are positively correlated with a refusal to donate. The highest impact stems from *Self Assertion*. Previous research (Fransen et al., 2015a; Zuwerink Jacks and Cameron, 2003) has noticed that *Self Assertion* is prominent amongst individuals with high self-esteem. Such individuals are confident about their beliefs and less likely to conform. Similarly, a high positive coefficient for *Information Inquiry* can be attributed as follows. EE inquires information about the charity not only as a means to verify their legitimacy, but also to gain the knowledge they can exploit to their advantage. An innocuous question like ‘Where will my money go?’ would enable EE to assert that they are keener to help children in their own country instead, thereby resisting the donation attempt and saving face.

The CB scenario setup ensures that the coefficients of the strategies set for BU and SE would be anti-correlated, which holds for the Table 8. Like P4G, a high negative coefficient of *Self Assertion* signifies that SE’s price is disagreeable to BU - they would instead not buy. Moreover, the high coefficient of *Counter Argumentation* justifies that it is an effective tactic for both parties.

7 Conclusion

We present a generalised computational framework grounded in cognitive psychology to operationalise resisting strategies employed to counter persuasion. We identify seven distinct resisting strategies that we instantiate on two publicly available corpora comprising persuasion and negotiation conversations. We adopt a hierarchical sequence labelling architecture to infer the resisting strategies automatically and observe that our model achieves competitive performance for both datasets. Furthermore, we examine the interplay of resisting strategies in determining the final conversation outcome, which corroborates with previous findings. In the future, we would like to explore better models to encode the strategy information and apply our framework to improve personalised persuasion and negotiation dialogue systems. We would also like to study the influence of other confounding factors such as power dynamics on the outcomes of conversations featuring resisting strategies.

Acknowledgments

We thank the anonymous EACL reviewers for their insightful comments and constructive feedback. This research was funded in part by NSF Grants (IIS 1917668 and IIS 1822831) and Dow Chemical. The first author would also like to acknowledge his best friend, Ahana Sadhu, for her constant support and motivation, who unfortunately and untimely left us this year.

References

Robert P Abelson and James C Miller. 1967. Negative persuasion via personal insult. *Journal of Experimental Social Psychology*, 3(4):321–333.

Rohini Ahluwalia. 2000. Examination of psychological processes underlying resistance to persuasion. *Journal of Consumer Research*, 27(2):217–232.

Khalid Al Khatib, Michael Völske, Shahbaz Syed, Nikolay Kolyada, and Benno Stein. 2020. Exploiting personal characteristics of debaters for predicting persuasiveness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7067–7072, Online. Association for Computational Linguistics.

Khalid Al-Khatib, Henning Wachsmuth, Kevin Lang, Jakob Herpel, Matthias Hagen, and Benno Stein. 2018. Modeling deliberative argumentation strategies on Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2545–2555, Melbourne, Australia. Association for Computational Linguistics.

Larry M Bartels. 2006. Priming and persuasion in presidential campaigns. *Capturing campaign effects*, 1:78–114.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.

Amparo Elizabeth Cano-Basave and Yulan He. 2016. A study of the impact of persuasive argumentation in political debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1405–1413, San Diego, California. Association for Computational Linguistics.

Winston Carlile, Nishant Gurrapadi, Zixuan Ke, and Vincent Ng. 2018. Give me more feedback: Annotating argument persuasiveness and related attributes in student essays. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 621–631, Melbourne, Australia. Association for Computational Linguistics.

Zheqian Chen, Rongqin Yang, Zhou Zhao, Deng Cai, and Xiaofei He. 2018. Dialogue act recognition via crf-attentive structured network. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 225–234.

John W Cooley. 1993. A classical approach to mediation-part i: Classical rhetoric and the art of persuasion in mediation. *U. Dayton L. Rev.*, 19:83.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5636–5646, Hong Kong, China. Association for Computational Linguistics.

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

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Esin Durmus and Claire Cardie. 2018. Exploring the role of prior beliefs for argument persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1035–1045, New Orleans, Louisiana. Association for Computational Linguistics.

Ritam Dutt, Rishabh Joshi, and Carolyn Penstein Rose. 2020. Keeping up appearances: Computational modeling of face acts in persuasion oriented discussions. *arXiv preprint arXiv:2009.10815*.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Marieke L Fransen, Edith G Smit, and Peeter WJ Verlegh. 2015a. Strategies and motives for resistance to persuasion: an integrative framework. *Frontiers in psychology*, 6:1201.

Marieke L Fransen, Peeter WJ Verlegh, Amna Kirmani, and Edith G Smit. 2015b. A typology of consumer strategies for resisting advertising, and a review of mechanisms for countering them. *International Journal of Advertising*, 34(1):6–16.

Boris Galitsky, Dmitry Ilvovsky, and Dina Pisarevskaya. 2018. Argumentation in text: Discourse structure matters. *CICLing 2018*.

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2333–2343.

Teresa Heath, Robert Cluley, and Lisa O'Malley. 2017. Beating, ditching and hiding: consumers' everyday resistance to marketing. *Journal of Marketing Management*, 33(15-16):1281–1303.

Wenxiang Jiao, Haiqin Yang, Irwin King, and Michael R. Lyu. 2019. HiGRU: Hierarchical gated recurrent units for utterance-level emotion recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 397–406, Minneapolis, Minnesota. Association for Computational Linguistics.

Rishabh Joshi, Vidhisha Balachandran, Shikhar Vashishth, Alan Black, and Yulia Tsvetkov. 2021. Dialograph: Incorporating interpretable strategy-graph networks into negotiation dialogues. In *International Conference on Learning Representations*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Silvia Knobloch-Westerwick and Jingbo Meng. 2009. Looking the other way: Selective exposure to attitude-consistent and counterattitudinal political information. *Communication Research*, 36(3):426–448.

Gaurav Kumar, Rishabh Joshi, Jaspreet Singh, and Promod Yenigalla. 2020. AMUSED: A multi-stream vector representation method for use in natural dialogue. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 750–758, Marseille, France. European Language Resources Association.

Liane Longpre, Esin Durmus, and Claire Cardie. 2019. Persuasion of the undecided: Language vs. the listener. In *Proceedings of the 6th Workshop on Argument Mining*, pages 167–176, Florence, Italy. Association for Computational Linguistics.

Andrew Luttrell and Vanessa Sawicki. 2020. Attitude strength: Distinguishing predictors versus defining features. *Social and Personality Psychology Compass*, 14(8):e12555.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6818–6825.

Vitobha Munigala, Abhijit Mishra, Srikanth G Tamilselvam, Shreya Khare, Riddhiman Dasgupta, and Anush Sankaran. 2018. Persuaide! an adaptive persuasive text generation system for fashion domain. In *Companion Proceedings of the The Web Conference 2018*, pages 335–342.

Daniel J O'Keefe. 2002. Guilt as a mechanism of persuasion. *The persuasion handbook: Developments in theory and practice*, pages 329–344.

Vipul Raheja and Joel Tetreault. 2019. Dialogue Act Classification with Context-Aware Self-Attention. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3727–3733, Minneapolis, Minnesota. Association for Computational Linguistics.

Victor Tejedor San José. 2019. The role of humor and threat on predicting resistance and persuasion.

Steven J Sherman and Larry Gorkin. 1980. Attitude bolstering when behavior is inconsistent with central attitudes. *Journal of Experimental Social Psychology*, 16(4):388–403.

Paul Surgi Speck and Michael T Elliott. 1997. Predictors of advertising avoidance in print and broadcast media. *Journal of Advertising*, 26(3):61–76.

Zakary L Tormala. 2008. A new framework for resistance to persuasion: The resistance appraisals hypothesis. *Attitudes and attitude change*, pages 213–234.

Anita L Vangelisti, John A Daly, and Janine Rae Rudnick. 1991. Making people feel guilty in conversations: Techniques and correlates. *Human Communication Research*, 18(1):3–39.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5635–5649, Florence, Italy. Association for Computational Linguistics.

Wendy Wood and Carl A Kallgren. 1988. Communicator attributes and persuasion: Recipients' access to attitude-relevant information in memory. *Personality and Social Psychology Bulletin*, 14(1):172–182.

Peter Wright. 1975. Factors affecting cognitive resistance to advertising. *journal of Consumer Research*, 2(1):1–9.

Diyi Yang, Jiaao Chen, Zichao Yang, Dan Jurafsky, and Eduard Hovy. 2019. Let's make your request more persuasive: Modeling persuasive strategies via semi-supervised neural nets on crowdfunding platforms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3620–3630, Minneapolis, Minnesota. Association for Computational Linguistics.

Jichuan Zeng, Jing Li, Yulan He, Cuiyun Gao, Michael Lyu, and Irwin King. 2020. What changed your mind: The roles of dynamic topics and discourse in argumentation process. In *Proceedings of The Web Conference 2020*, WWW '20, page 1502–1513, New York, NY, USA. Association for Computing Machinery.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yiheng Zhou, He He, Alan W Black, and Yulia Tsvetkov. 2019. A dynamic strategy coach for effective negotiation. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 367–378, Stockholm, Sweden. Association for Computational Linguistics.

Julia Zuwerink Jacks and Kimberly A Cameron. 2003. Strategies for resisting persuasion. *Basic and applied social psychology*, 25(2):145–161.

Appendix

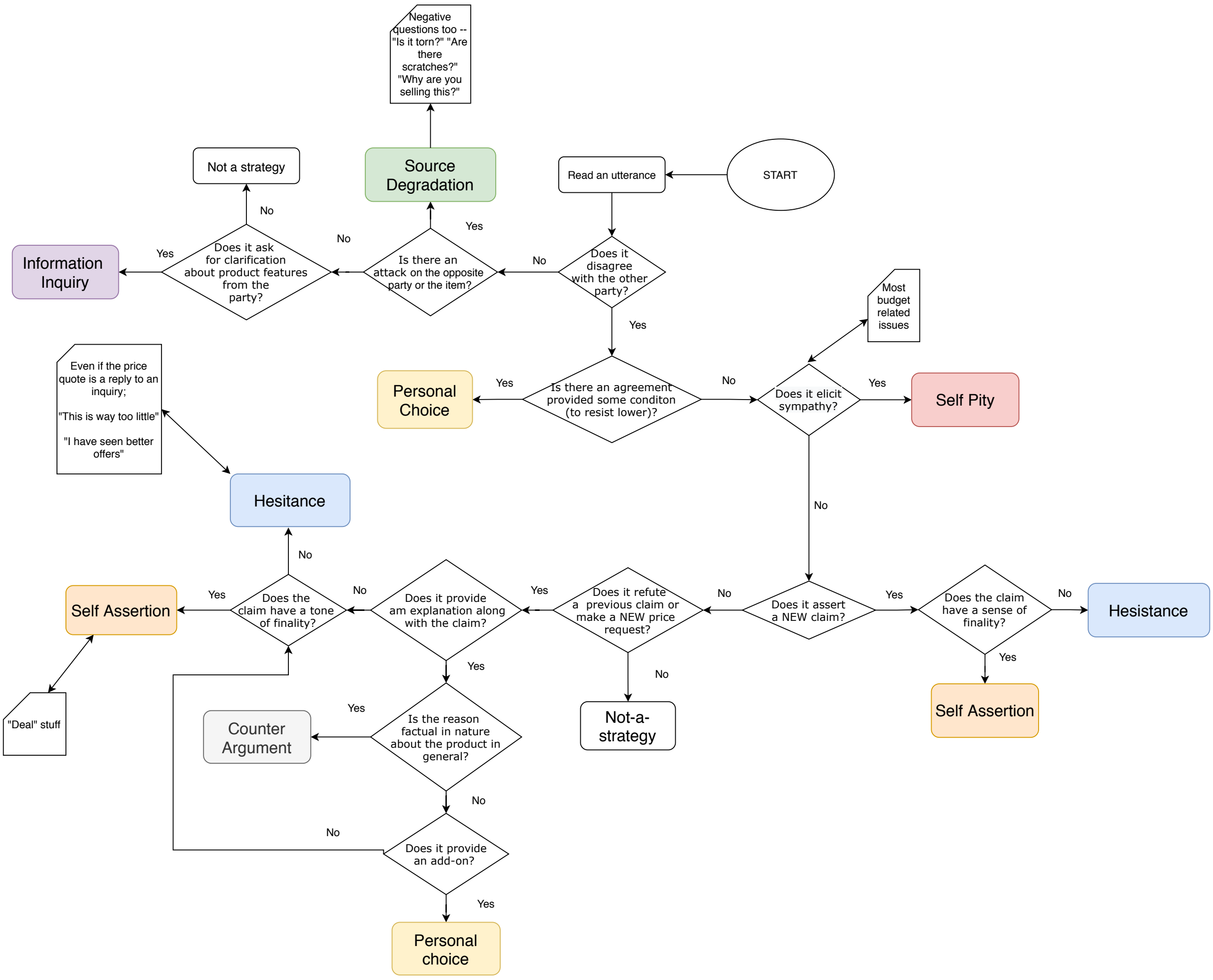


Figure 3: Flowchart for annotating CB

BERxiT: Early Exiting for BERT with Better Fine-Tuning and Extension to Regression

Ji Xin, Raphael Tang, Yaoliang Yu, and **Jimmy Lin**
David R. Cheriton School of Computer Science, University of Waterloo
Vector Institute for Artificial Intelligence
`{ji.xin,r33tang,yaoliang.yu,jimmylin}@uwaterloo.ca`

Abstract

The slow speed of BERT has motivated much research on accelerating its inference, and the early exiting idea has been proposed to make trade-offs between model quality and efficiency. This paper aims to address two weaknesses of previous work: (1) existing fine-tuning strategies for early exiting models fail to take full advantage of BERT; (2) methods to make exiting decisions are limited to classification tasks. We propose a more advanced fine-tuning strategy and a learning-to-exit module that extends early exiting to tasks other than classification. Experiments demonstrate improved early exiting for BERT, with better trade-offs obtained by the proposed fine-tuning strategy, successful application to regression tasks, and the possibility to combine it with other acceleration methods. Source code can be found at `https://github.com/castorini/berxit`.

1 Introduction

Large-scale pre-trained language models such as BERT (Devlin et al., 2019) have brought the natural language processing (NLP) community large performance gain but at the cost of heavy computational burden. While pre-trained models are available online and fine-tuning is typically done without a strict time budget, inference poses a much lower latency tolerance, and the slow inference speed of these models can impede easy deployment. It becomes even more difficult when inference has to be done on edge devices due to limited network capabilities or privacy concerns.

Early exiting (Schwartz et al., 2020; Xin et al., 2020a; Liu et al., 2020) has been proposed to accelerate the inference of BERT and models with similar architecture, i.e., those comprising multiple transformer layers (Vaswani et al., 2017) with a classifier at the top. Instead of using only one classifier, additional classifiers are attached to each transformer layer (see Figure 1), and the entire model is fine-tuned together. At inference time, the sample can perform early exiting through one of the intermediate classifiers.

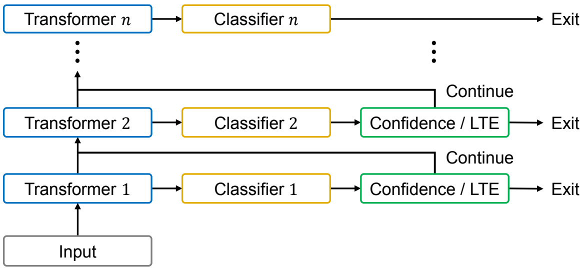


Figure 1: Multi-output structure of early exiting BERT.

While existing early exiting papers provide promising quality–efficiency trade-offs, improvements are necessary for two important components: fine-tuning strategies and exiting decision making. In these papers, fine-tuning strategies are relatively simple and fail to take full advantage of the pre-trained model's effectiveness; we propose a novel fine-tuning strategy, *Alternating*, for this multi-output model. Moreover, previous work makes exiting decisions based on the confidence of output probability distributions, and is hence only applicable to classification tasks; we extend it to other tasks by proposing the *learning-to-exit* idea. With carefully designed fine-tuning strategies and methods for making exiting decisions, the model can achieve better quality–efficiency trade-offs and can be extended to regression tasks.

We refer to our proposed ideas collectively as BERxiT (BERT+exit), and apply it to Muppets[1] including BERT, RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020); we also apply it on top of another BERT acceleration method, DistilBERT (Sanh et al., 2019). We conduct experiments

---

[1] BERT and his friends.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 91–104
April 19 - 23, 2021.

on datasets including classification and regression tasks, and show that our method can save up to 70% of inference time with minimal quality degradation.

Our contributions include the following: (1) an effective fine-tuning method *Alternating*; (2) the *learning-to-exit* idea that extends early exiting to tasks other than classification; (3) extensive experiments that show the effectiveness of our ideas and the successful combination of early exiting with other BERT acceleration methods; (4) additional experiments that provide insight into the inner mechanism of pre-trained models.

2 Related Work

BERT (Devlin et al., 2019) is a pre-trained multi-layer transformer (Vaswani et al., 2017) model. RoBERTa (Liu et al., 2019) and ALBERT (Lan et al., 2020) are variants of BERT with almost identical model architectures but different training methods and parameter sharing strategies. In our paper, we refer to these transformer-based models as Muppets, and apply our method on them.

In the general deep learning context, there are a number of well-explored methods to accelerate model inference. Pruning (Han et al., 2015; Fan et al., 2020; Gordon et al., 2020) removes unimportant parts of the neural model, from individual weights to layers and blocks. Quantization (Lin et al., 2016; Shen et al., 2020) reduces the number of bits needed to operate a neural model and to store its weights. Distillation (Hinton et al., 2015; Jiao et al., 2020) transfers knowledge from large teacher models to small student models. These methods typically require pre-training Muppets from scratch[2] and produce only one small model with a predetermined target size. Early exiting requires only fine-tuning and also produces a series of small models, from which the user can choose flexibly. It extends the idea of Adaptive Computation (Graves, 2016) for recurrent neural networks, and is also closely related to BranchyNet (Teerapittayanon et al., 2016), Multi-Scaled DenseNet (Huang et al., 2018), and Slimmable Network (Yu et al., 2019).

Early exiting for Muppets has been explored by RTJ[3] (Schwartz et al., 2020), DeeBERT (Xin et al., 2020a,b), and FastBERT (Liu et al., 2020). Despite their promising results, there is still room for improvement regarding the fine-tuning strategies of RTJ and DeeBERT. FastBERT, on the other hand, uses self-distillation (Zhang et al., 2019; Phuong and Lampert, 2019) for fine-tuning, which works well for small Muppets such as BERT$_{\text{BASE}}$. However, our preliminary experiments[4] show that self-distillation is unstable on larger Muppets such as BERT$_{\text{LARGE}}$, suggesting that future work is necessary for fully understanding and robustly applying self-distillation on Muppets.

All these three methods make early exiting decisions based on confidence (or its variants) of the predicted probability distribution, and are therefore limited to classification tasks. Runtime Neural Pruning (Lin et al., 2017), SkipNet (Wang et al., 2018b), and BlockDrop (Wu et al., 2018) use reinforcement learning (RL) to decide whether to execute a network module. Universal Transformer (Dehghani et al., 2019) and Depth-Adaptive Transformer (DAT, Elbayad et al., 2020) use learned decisions for early exiting in sequence-to-sequence tasks. Concurrently, PABEE (Zhou et al., 2020) proposes patience-based early exiting which is applicable to regression, but it relies on inter-layer prediction consistency and is therefore not very efficient for exiting at early layers. Inspired by them, we propose a method to extend early exiting for Muppets to regression tasks, using only layer-specific information as in classification. Moreover, our method requires neither RL nor complicated distribution fitting as in DAT, but uses a straightforward layer-wise certainty estimation, and achieves performance comparable with confidence-based early exiting on classification tasks.

3 Model Structure and Fine-Tuning

We start from a pre-trained Muppet model (the *backbone* model), attach additional classifiers to it, fine-tune the model, and use it for accelerated inference by early exiting.

Backbone model The backbone model is an n-layer pre-trained Muppet model. We denote the i^{th} layer hidden state corresponding to the `[CLS]` token as $\mathbf{h}_i$:

$$\mathbf{h}_i = f_i(x; \theta_1, \cdots, \theta_i), \tag{1}$$

where x is the input sequence, θ_i is the parameters of the i^{th} transformer layer, and f_i is the mapping from input to the i^{th} layer hidden state.

---

[2] In distillation, there is typically a general distillation that uses the large-scale pre-training corpus and is time consuming (Jiao et al., 2020).

[3] Short for *Right Tool for the Job*—the paper does not provide a concise name for the method.

[4] See Appendix A.

Classifiers In the original BERT paper (Devlin et al., 2019), the way to fine-tune is to attach a classifier to the *final* transformer layer, and then to jointly update both the backbone model and the classifier. The classifier is a one-layer fully-connected network. It takes as input the final layer hidden state $\mathbf{h}_n$ and outputs a prediction. Its output is a probability distribution over all classes for classification tasks and a scalar for regression[5] tasks.

To enable early exiting, we instead attach a classifier to *every* transformer layer, i.e., there are n classifiers in total. Each classifier can make its own prediction, and therefore the model can accelerate inference by exiting earlier.

Fine-tuning strategies We discuss how to fine-tune this multi-output network. The loss function for the i^{th} layer classifier is

$$L_i(x, y) = H(y, g_i(\mathbf{h}_i; w_i)), \tag{2}$$

where x and y are the input sequence and corresponding label, g_i the i^{th} layer's classifier, w_i the parameters of g_i, and H the task-specific loss function, e.g., cross-entropy for classification tasks and mean squared error (MSE) for regression tasks.

The most straightforward fine-tuning strategy is perhaps minimizing the sum of all classifiers' loss functions and jointly updating all parameters in the process. We refer to this strategy as *Joint*, and it is also used in RTJ (Schwartz et al., 2020):

$$\min_{\substack{\theta_1,\cdots,\theta_n \\ w_1,\cdots,w_n}} \sum_{i=1}^{n} L_i. \tag{3}$$

If we hope to preserve the best model quality for the final layer, the desired fine-tuning strategy is *Two-stage*, which is also used in DeeBERT (Xin et al., 2020a). The first stage is identical to vanilla BERT fine-tuning: updating the backbone model and only the final classifier. In the second stage, we freeze all parameters updated in the first stage, and fine-tune the remaining classifiers. Objectives in the two stages are as follows.

$$\text{Stage 1:} \quad \min_{\substack{\theta_1,\cdots,\theta_n \\ w_n}} \quad L_n \tag{4}$$

$$\text{Stage 2:} \quad \min_{w_1,\cdots,w_{n-1}} \quad \sum_{i=1}^{n-1} L_i \tag{5}$$

These two fine-tuning strategies are not ideal. Intuitively, in this multi-output network, loss functions of different classifiers interfere with each other in a *negative* way. Transformer layers have to provide hidden states for two competing purposes: immediate inference at the adjacent classifier and gradual feature extraction for future classifiers. Therefore, achieving a balance between the classifiers is critical. *Two-stage* produces final classifiers with optimal quality at the price of earlier layers, since most parameters are solely optimized for the final classifier. *Joint* treats all classifiers equally, and therefore its final classifier is less effective than that of *Two-stage*. To combine the advantages, we propose a novel fine-tuning strategy, *Alternating*. It alternates between two objectives (taken from Equation 3 and 4) for odd-numbered and even-numbered iterations.

$$\text{Odd:} \quad \min_{\substack{\theta_1,\cdots,\theta_n \\ w_n}} \quad L_n \tag{6}$$

$$\text{Even:} \quad \min_{\substack{\theta_1,\cdots,\theta_n \\ w_1,\cdots,w_n}} \quad \sum_{i=1}^{n} L_i \tag{7}$$

Combining objectives from *Joint* and *Two-stage*, *Alternating* has the potential to find the most preferable region in the parameter space: the intersection between optimal regions for different layers.

4 Exiting Decision Making

After the entire model (including the backbone and all classifiers) is fine-tuned, it can perform early exiting for an inference sample. In this section we discuss two methods to make exiting decisions.

4.1 Confidence Threshold

When the model is "certain" enough of its prediction at an intermediate layer, the forward inference can be terminated.

For classification tasks, a straightforward measurement of the prediction certainty is the maximum probability of the output prediction, which is referred to as *confidence* in previous work (Schwartz et al., 2020; Liu et al., 2020). Similarly, Xin et al. (2020a) use entropy as the metric, which is also closely related to confidence. Before inference starts, a confidence threshold is chosen. In forward propagation, the confidence of the output at each layer is compared with the threshold; if it is larger than the threshold at a certain layer, the sample exits and future layers are skipped.

---

[5] In this case, we still refer to this one-layer network as *classifier* for naming consistency.

4.2 Learning to Exit

While using a confidence or entropy threshold is straightforward and effective, it is exploiting the fact that the classifier's output is a probability distribution in classification tasks. This is generally not the case for other tasks such as regression. To address the gap, we propose *learning-to-exit* (LTE) as a substitute when the distribution is unavailable.

The i^{th} layer hidden state $\mathbf{h}_i$ is a vector in the embedding space. Intuitively, different regions of the embedding space have different certainty levels. For instance, in binary classification tasks, regions closer to the decision boundary have a lower certainty level, and this is explicitly expressed as a lower confidence of the output probability distribution. But even when certainty cannot be explicitly measured, we can still train an auxiliary LTE module to estimate such a metric.

Concretely, the LTE module is a simple one-layer fully-connected network. It takes as input the hidden state $\mathbf{h}_i$ and outputs the certainty level u_i of the sample at the i^{th} layer:

$$u_i = \sigma(\mathbf{c}^\top \mathbf{h}_i + b), \qquad (8)$$

where σ is the sigmoid function, $\mathbf{c}$ is the weight vector, and b is the bias term.

The loss function for the LTE module is a simple MSE between u_i and the "ground truth" certainty level at the i^{th} layer $\tilde{u}_i$:

$$J_i = ||u_i - \tilde{u}_i||_2^2. \qquad (9)$$

For classification, the ground truth certainty level is whether the classifier makes the correct prediction:

$$\tilde{u}_i = \mathbb{1}[\arg\max_j g_i^{(j)}(\mathbf{h}_i; w_i) = y], \qquad (10)$$

where g_i is the output probability distribution at the i^{th} layer and $g_i^{(j)}$ is its j^{th} entry. For regression, the ground truth certainty level is negatively related to the prediction's absolute error:

$$\tilde{u}_i = 1 - \tanh(|g_i(\mathbf{h}_i; w_i) - y|). \qquad (11)$$

To apply LTE, we initialize the LTE module together with classifiers and it is shared among all layers. We train the LTE module jointly with the rest of the model by *substituting* L_i in Equation 3–7 with $L_i + J_i$. At inference time, if the predicted certainty level is higher than the chosen threshold, the inference sample performs early exiting.

Dataset	Labels	Train / Dev / Test
RTE	2	2.5k / 0.3k / 3.0k
MRPC	2	3.7k / 0.4k / 1.7k
SST-2	2	67k / 0.9k / 1.8k
QNLI	2	105k / 5.5k / 5.5k
QQP	2	364k / 40k / 391k
MNLI	3	393k / 9.8k / 9.8k
STS-B	1	8.6k / 1.5k / 1.4k
SICK	1	4.4k / 4.9k / –

Table 1: Statistics of datasets.

5 Experiments

5.1 Setup

We conduct experiments on six classification datasets of the GLUE benchmark (Wang et al., 2018a); since there is only one regression dataset, STS-B (Cer et al., 2017), in GLUE, we additionally use another regression dataset, SICK (Marelli et al., 2014). Statistics of these datasets are listed in Table 1. Our implementation is adapted from the Huggingface Transformer Library (Wolf et al., 2020). We conduct searches on experiment settings such as the optimizer, learning rates, hidden state sizes, and dropout probabilities, and discover that it is best to keep original settings from the library. Random seeds are also unchanged from the library for fair comparisons.[6]

Most results in this paper use the dev split, since the large number of evaluations we need are forbidden by the GLUE evaluation server. The only exception is Table 2, where we report model quality–efficiency trade-offs on the test split.

5.2 Layer-wise Scores Comparison

We discuss three fine-tuning strategies in Section 3: *Joint* (also used in RTJ), *Two-stage* (also used in DeeBERT), and *Alternating* (proposed in this paper). In tables and figures, they are labeled respectively as JOINT, 2STG, and ALT. Figures 2 and 3 compare these three fine-tuning strategies by showing their *layer-wise score* curves: each point in the curve shows the output score at a certain exit layer, i.e., all samples are required to exit at this layer for evaluation. More specifically, we report *relative* scores, and the 100% baseline is the original score of the vanilla Muppet without early exiting, and

---

[6] Detailed experiment settings are in Appendix B.

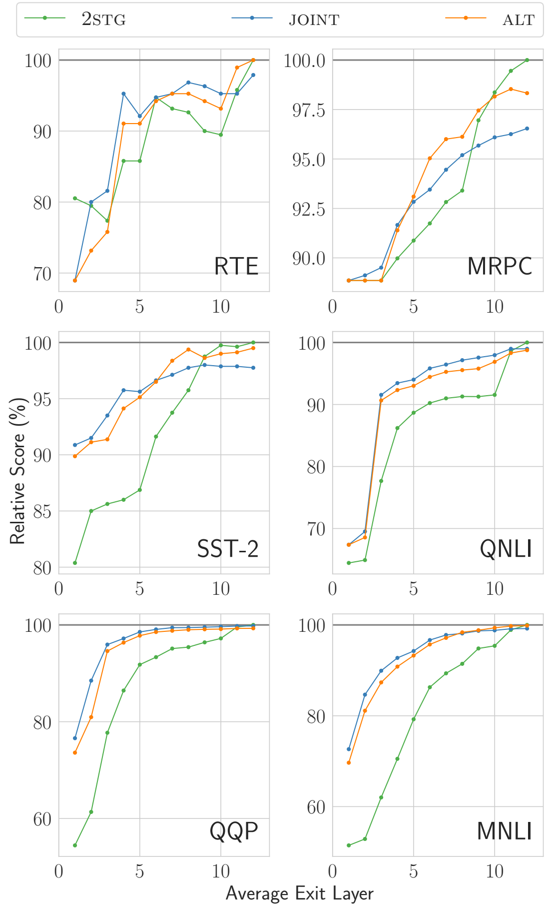


Figure 2: Layer-wise scores of different fine-tuning strategies for BERT$_{\text{BASE}}$.

this is also the score of the final layer of *Two-stage* because of parameter freezing in its second stage.

For BERT$_{\text{BASE}}$, we show plots for all six classification datasets, ordered by their training set sizes from smallest to largest. As we will see in later analyses, low-resource datasets show the most difference. Therefore, for RoBERTa$_{\text{BASE}}$ and ALBERT$_{\text{BASE}}$, we only show plots for RTE and MRPC (with training set size smaller than 6% of others) due to space limitations.[7]

We observe the following from the figures:

- *Two-stage* is unsatisfying. While it achieves the best score at the final layer, it comes at a large cost of other layers, especially for non-low-resource datasets.
- *Alternating* is better than *Joint* in later layers, and weaker in earlier layers. However, as we will see in the next section, when we evaluate quality–efficiency trade-offs of confidence-based early exiting, the weakness of *Alternating* in earlier layers is no longer substantial, while its advantage is preserved.
- The difference between *Joint* and *Alternating* is larger for low-resource datasets, where the training set is insufficient to fine-tune all layers well simultaneously.
- Interestingly, for ALBERT$_{\text{BASE}}$, *Alternating*'s relative scores are higher than 100% in the final layers. We speculate that this is because of the parameter sharing nature of ALBERT and the small sizes of the datasets: better supervision for intermediate layers also helps the final layer.

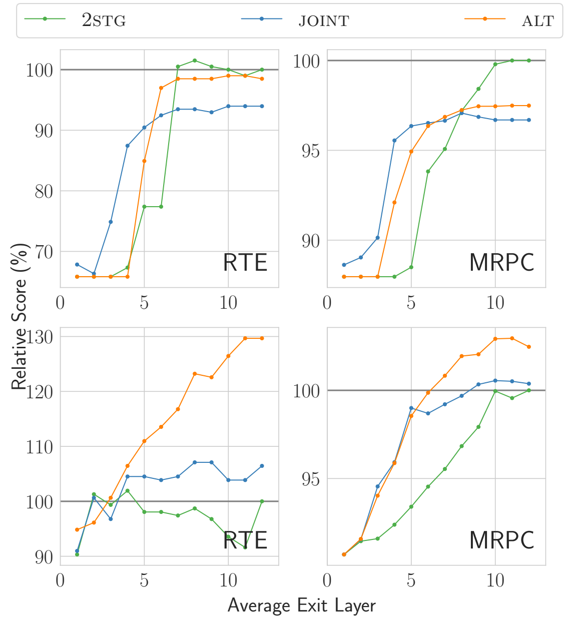


Figure 3: Layer-wise scores of different fine-tuning strategies for RoBERTa$_{\text{BASE}}$ (top two) and ALBERT$_{\text{BASE}}$ (bottom two).

5.3 Early Exiting Trade-offs Comparison

From the previous section, we see that *Two-stage* is visibly less preferable than the other two. Therefore in this section, we compare quality–efficiency trade-offs of *Joint* and *Alternating* when confidence threshold is used for making exiting decisions.

Specifically, we use *average exit layer* of all inference samples as the metric of efficiency for the following reasons: (1) it is linear w.r.t. the actual amount of computation; (2) according to our experiments, it is proportional to actual wall-clock runtime, and is also stable across different runs.[8]

---

[7] Results for other datasets are in Appendix C.

[8] Direct runtime measurement has the randomness caused

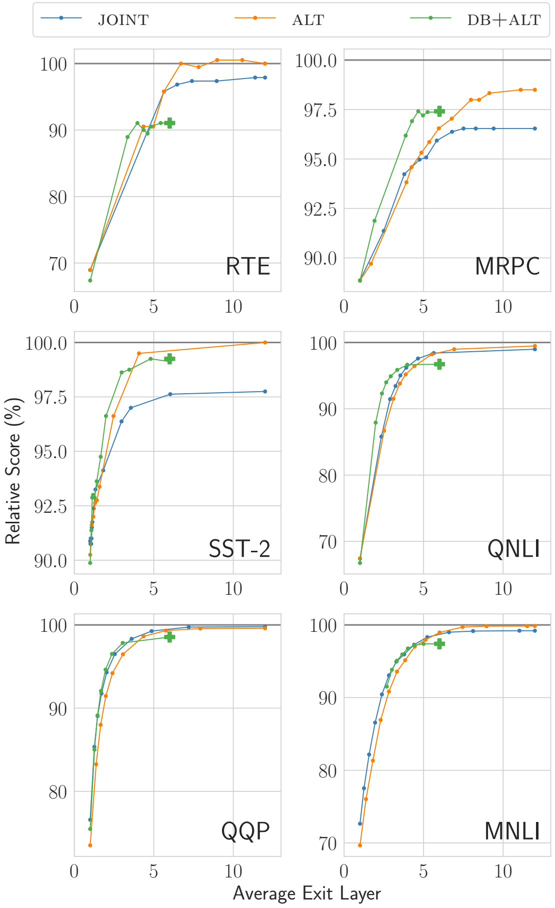


Figure 4: Quality–efficiency trade-offs using confidence for exiting decisions for $\text{BERT}_{\text{BASE}}$.

We visualize the trade-offs in Figures 4 and 5, and also show detailed numbers in Table 2 using results from the test set. Dots in the figures and ALT rows in the table are generated by varying the confidence threshold, and the thresholds are chosen to show trade-offs at different average exit layers. In addition to the comparison between *Joint* and *Alternating*, we add another strong baseline, DistilBERT (Sanh et al., 2019). We apply *Alternating* fine-tuning and early exiting on top of DistilBERT (labeled as DB+ALT), and the rightmost point of the curve is DistilBERT itself without early exiting (the green ✚). Observations from the table and figures are as follows:

- On the test set, early exiting with *Alternating* fine-tuning saves a large amount of inference computation, with only minimal quality degradation, compared with vanilla Muppets.
- Compared with *Joint*, *Alternating* inherits its benefits from the previous section: better trade-offs at higher scores (larger average exit layer). Additionally, its improvements are larger in smaller datasets.

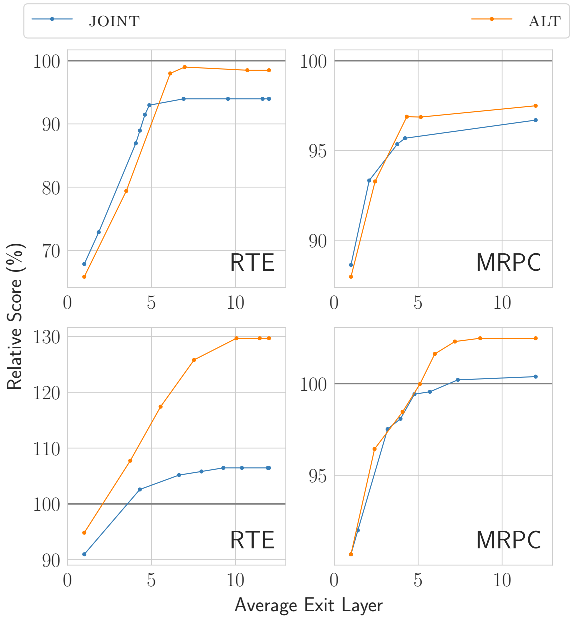


Figure 5: Quality–efficiency trade-offs using confidence for exiting decisions for $\text{RoBERTa}_{\text{BASE}}$ (top two) and $\text{ALBERT}_{\text{BASE}}$ (bottom two).

- *Alternating*'s weakness at more aggressive exiting (smaller average exit layer) is minimized. Take Figure 4 as an example, we report *the area of one curve above the other* as a numerical metric: JOINT over ALT and ALT over JOINT is respectively (0.4, 13.5) for RTE, (0.9, 8.8) for MRPC, and (0.2, 18.2) for SST-2. The advantage of *Alternating* indicates that later layers intrinsically contribute more to early exiting performance, partly because the final layer's score is the upper bound for all previous layers (ignoring randomness in training). This shows that the *Joint* fine-tuning strategy, which treats all layers equally, is not ideal.
- In most cases, *Alternating* outperforms DistilBERT, which requires distillation in pre-training and is therefore much more resource-demanding. It also further improves model efficiency on top of DistilBERT, indicating that early exiting is *cumulative* with other acceleration methods.

5.4 Learning to Exit Performance

To examine the effectiveness of LTE, we apply it on top of models fine-tuned with *Alternating*. We show the results in Figure 6 on four datasets.

---

by other processes on the same machine. Detailed discussions can be found in Appendix D.

	RTE		MRPC		SST-2		QNLI		QQP		MNLI-(m/mm)		STS-B	
	Score	Layer	Score	Layer	Score	Layer	Score	Layer	Score	Layer	Score	Layer	Score	Layer
BERT$_{\text{BASE}}$														
RAW	66.4	12	88.9	12	93.5	12	90.5	12	71.2	12	84.6/83.4	12	85.8	12
ALT	101%	−44%	99%	−30%	98%	−65%	99%	−42%	99%	−56%	99%/99%	−37%	95%	−50%
	99%	−54%	97%	−56%	96%	−79%	98%	−63%	97%	−75%	97%/97%	−57%	91%	−67%
	96%	−64%	94%	−74%	94%	−87%	95%	−71%	93%	−84%	93%/92%	−72%	85%	−75%
DB	86%	−50%	97%	−50%	98%	−50%	97%	−50%	98%	−50%	97%/97%	−50%	94%	−50%
DB+ALT	86%	−55%	97%	−60%	98%	−75%	97%	−72%	98%	−76%	96%/97%	−66%	93%	−66%
BERT$_{\text{LARGE}}$														
RAW	70.1	24	89.3	24	94.9	24	92.7	24	72.1	24	86.7/85.9	24	86.5	24
ALT	95%	−33%	99%	−32%	100%	−32%	97%	−62%	98%	−74%	99%/99%	−36%	97%	−39%
	94%	−46%	98%	−46%	99%	−61%	95%	−73%	96%	−82%	96%/97%	−57%	90%	−62%
	88%	−62%	94%	−71%	96%	−78%	91%	−83%	91%	−89%	90%/90%	−75%	76%	−80%

Table 2: Test set results comparing baselines (raw BERT/RoBERTa, from the original paper), DistilBERT, and early exiting with *Alternating* fine-tuning. Metric for model quality: score for RAW baselines and relative scores for others. Metric for model efficiency: used layers for RAW; relative saved layers for others (w.r.t. raw models).

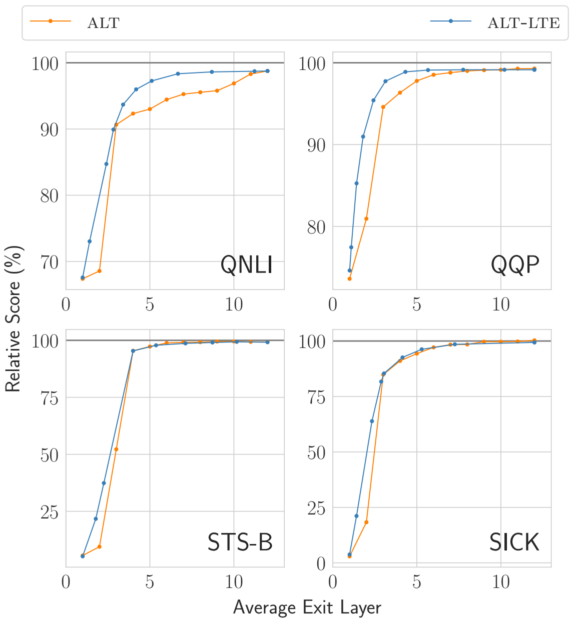


Figure 6: Comparing layer-wise score of *Alternating* with LTE-based early exiting on top of *Alternating*. BERT$_{\text{BASE}}$ is the backbone.

Method	Rel. Score	Speedup	Avg. Exit Layer
PABEE	99%	2.1	5.7
	98%	2.4	5.0
ALT-LTE	99%	2.2	5.4
	98%	2.6	4.6

Table 3: Comparing LTE with PABEE on STS-B.

We use the *layer-wise* score of *Alternating* as the baseline: if we want to save $x\%$ inference runtime, a straightforward way is to use the first $(100-x)\%$ layers for every sample, regardless of its difficulty. LTE is expected to dynamically allocate resources based on a sample's difficulty and therefore outperform this baseline. From the figures for QNLI and QQP, we observe that the blue curves are substantially above the orange curves, i.e., LTE provides better accuracy–efficiency trade-offs than the layer-wise baseline, achieving the same model quality with less computation. For regression tasks STS-B and SICK, the layer-wise baseline reaches its maximum score at relatively early layers, leaving little room for LTE to perform. Nevertheless, LTE still outperforms the baseline, especially in earlier layers (note that the y-axis is from 0 to 100%).

We also compare LTE with the concurrent patience-based baseline PABEE (Zhou et al., 2020) in Table 3, showing their speedups and average exit layers at the same relative scores. PABEE does not provide exact speedup numbers; therefore we estimate the values from their figures. We can see that *Alternating* fine-tuning plus LTE is marginally better than PABEE on regression tasks.

We further compare LTE-predicted certainty for each layer with layer-wise scores in Figure 7, where we observe large differences of predicted certainty both within and across layers. Also, predicted certainty is generally positively correlated with scores. This further demonstrates that the LTE module successfully captures certainty information based on the model's hidden state.

LTE extends confidence-based early exiting to

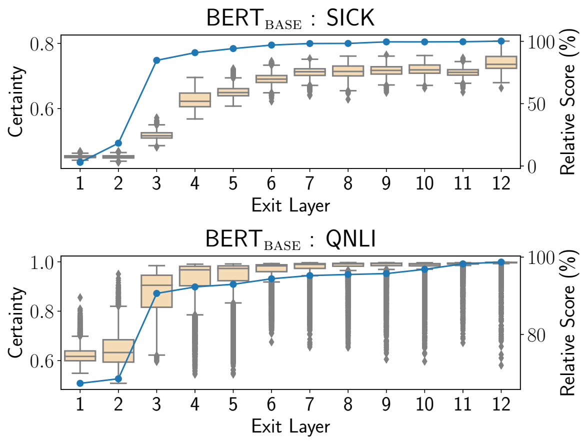


Figure 7: Comparison of each layer's score and learned certainty. Yellow box-plots: distribution of certainty at each layer; blue curve: relative score.

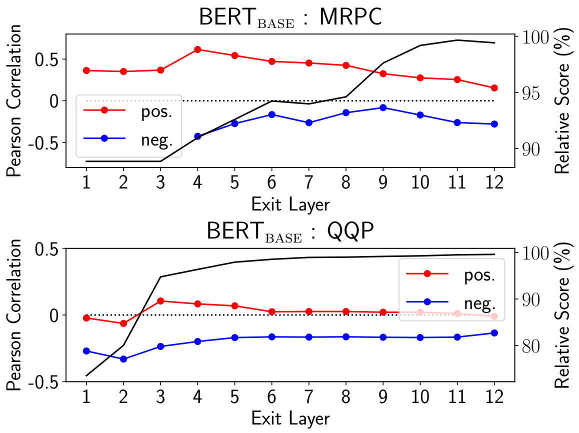


Figure 8: Comparison between BLEU–confidence correlation (red and blue) and layer-wise scores (black).

tasks other than classification. Furthermore, our LTE module is more straightforward and intuitive than DAT (Elbayad et al., 2020), yet achieves comparable results with classification tasks.

5.5 Prediction Confidence as a Probe

So far, we have only regarded prediction confidence as something produced by the black-box model and use it for making early exiting decisions. In this section, we show an example of how confidence is related to a human-interpretable feature, demonstrating its potential to reveal the inner mechanisms of Muppet models.

We choose two datasets, MRPC and QQP, where the task is to predict whether two input sequences are semantically equivalent. Intuitively, the BLEU score (Papineni et al., 2002) between the two sequences, which measures n-gram matching, may be related to the prediction. At each output layer, we first divide all dev set samples into two subsets by whether they are predicted as positive or negative; then, we calculate the BLEU-4 score for each sample, and calculate the Pearson correlation between BLEU scores and confidence in each subset; finally, we compare the correlation for both subsets in each layer, along with the layer-wise relative scores, in Figure 8.

We notice that the BLEU scores and predicted confidence show the strongest correlation in layers where the model quality starts to improve (layer 4–5 in MRPC[9] and 2–3 in QQP). After these layers, the correlation gradually weakens. It suggests that in the early layers, the model relies more on simple features such as n-gram matching for making semantic judgments: the higher the BLEU score is, the more certain it is for making positive predictions and the less certain it is for negative ones; however, with more layers, the model acquires the ability to look beyond the BLEU score, reducing its reliance on n-gram matching and achieving better performance. Therefore, with MRPC as an example, analyzing differences between layers 3 and 4 may reveal how the model detects n-gram matching, and analyzing differences between layers 4 and 6 may reveal advanced semantic features learned by the model.

6 Conclusion and Future Work

To improve early exiting for Muppets, we present BERxiT, including the *Alternating* fine-tuning strategy which outperforms methods from previous papers and the LTE idea which extends early exiting to a broader range of tasks. Experiments show the effectiveness of *Alternating* in providing better quality–efficiency trade-offs and the successful application of LTE to regression tasks. They also show that early exiting is cumulative with other acceleration methods such as DistilBERT and has the potential for model interpretation.

Future Work The fundamental question of early exiting for Muppets is how many transformer layers are sufficient for making good predictions. We draw inspiration from the *Limit* performance of Muppets: the score of *Limit* at the i^{th} layer is obtained by taking the first i transformer layers from the pre-trained Muppet model, attaching a classifier to the i^{th} layer, and fine-tuning this single-output

---

[9] In MRPC, the first three layers only make positive predictions due to the highly imbalanced training label distribution.

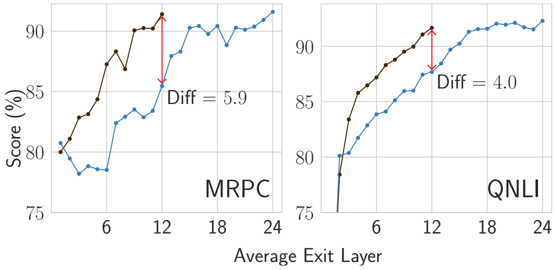


Figure 9: Comparison between LIMIT of BERT$_{\text{BASE}}$ (brown) and BERT$_{\text{LARGE}}$ (blue). Red arrow: difference between the two models when they both use 12 layers.

model. *Limit* estimates the upper bound for any fine-tuning methods by removing inter-classifier interference. We compare the *Limit* performance of BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$ in Figure 9, and notice that with the same number of layers (and identical fine-tuning strategy), BERT$_{\text{BASE}}$ almost always outperforms BERT$_{\text{LARGE}}$ by a large margin. This suggests that most transformer layers' potential to provide information for early exiting is limited by the single-output nature of pre-training. If we want to further improve early exiting Muppets for better trade-offs, adding more exiting paths in pre-training would be a promising direction.

Acknowledgements

We thank anonymous reviewers for their insightful suggestions. We also gratefully acknowledge funding support from the Natural Sciences and Engineering Research Council (NSERC) of Canada. Computational resources used in this work were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

References

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

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

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. 2020. Depth-adaptive transformer. In *International Conference on Learning Representations*.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*.

Mitchell A Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the effects of weight pruning on transfer learning. *arXiv preprint arXiv:2002.08307*.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. 2018. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. 2016. Fixed point quantization of deep convolutional networks. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2849–2858, New York, New York, USA. PMLR.

Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. 2017. Runtime neural pruning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

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

Mary Phuong and Christoph H. Lampert. 2019. Distillation-based training for multi-exit architectures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. 2020. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6640–6651, Online. Association for Computational Linguistics.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-BERT: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. 2016. BranchyNet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E. Gonzalez. 2018b. SkipNet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S. Davis, Kristen Grauman, and Rogerio Feris. 2018. BlockDrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020a. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online. Association for Computational Linguistics.

Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. 2020b. Early exiting BERT for efficient document ranking. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 83–88, Online. Association for Computational Linguistics.

Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. 2019. Slimmable neural networks. In *International Conference on Learning Representations*.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. 2019. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. BERT loses patience: Fast and robust inference with early exit. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18330–18341. Curran Associates, Inc.

A Negative Results for Self-Distillation

FastBERT (Liu et al., 2020) does not provide result for BERT$_{\text{LARGE}}$ or RoBERTa$_{\text{LARGE}}$. We show BERT$_{\text{LARGE}}$ and RoBERTa$_{\text{LARGE}}$ layer-wise scores for different fine-tuning strategies in Figure 10. SD in the legend stands for self-distillation. We can see that for *Two-stage* and *Alternating*, the patterns are similar to those of BERT$_{\text{BASE}}$: *Alternating* better in earlier layers while *Two-stage* better in later layers.

However, self-distillation's behavior is inconsistent between models and datasets. While it performs as expected for BERT$_{\text{LARGE}}$ in SST-2 and MNLI, and for RoBERTa$_{\text{LARGE}}$ in MRPC, self-distillation fails to improve after the first few layers for BERT$_{\text{LARGE}}$ in MRPC and for RoBERTa$_{\text{LARGE}}$ in SST-2 and MNLI, and most layers' quality is considerably worse than *Alternating*. We therefore consider self-distillation an unstable and premature fine-tuning strategy.

B Additional Experiment Setting

For pre-trained models, we use the following ones provided by the Huggingface Transformer Library (Wolf et al., 2020) as backbone models:

- BERT-BASE-UNCASED
- BERT-LARGE-UNCASED
- ROBERTA-BASE
- ROBERTA-LARGE
- ALBERT-BASE-V2
- DISTILBERT-BASE-UNCASED

For BERT, ALBERT, and DistilBERT, we fine-tune for 3 epochs; for RoBERTa, we fine-tune for 10 epochs; no early-stopping or checkpoint selection is performed.

Experiments are done on a single NVIDIA P100 GPU with CUDA 10.1. For inference, we use a batch size of 1 (since we need to perform early exiting based on each individual sample's difficulty). Inference runtime for the entire dev set for all models and datasets is shown in Table 4. RoBERTa has the same model structure as BERT, and therefore its runtime is also very close to that of BERT. Note that this is affected by competing processes, and may vary between different runs.

Numbers of parameters for BERT and ALBERT backbone models can be found in the paper by Lan et al. (2020). RoBERTa shares the same model structure with BERT and has the same number of parameters. Numbers of parameters for early-exiting-specific modules, such as additional classifiers and the LTE module, are on the order of thousands, and are therefore negligible compared with those of backbone models (millions).

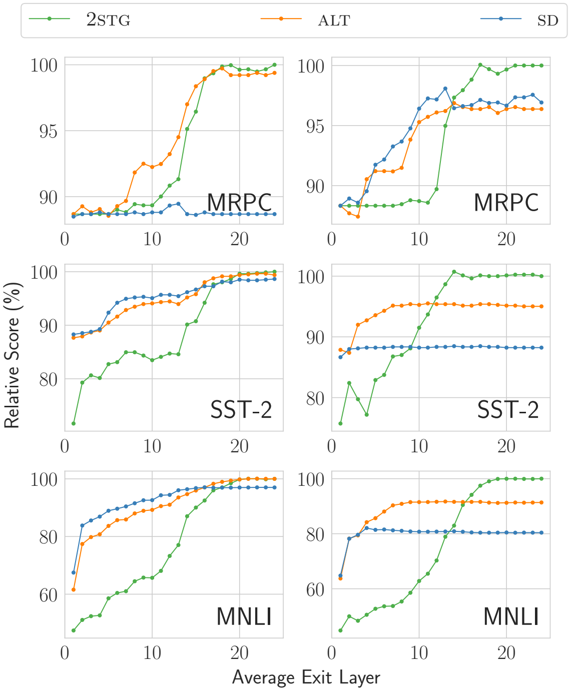


Figure 10: Layer-wise score for different fine-tuning strategies on BERT$_{\text{LARGE}}$ (left column) and RoBERTa$_{\text{LARGE}}$ (right column).

C Additional Experiment Results

In the main paper, we report results of RoBERTa$_{\text{BASE}}$ and ALBERT$_{\text{BASE}}$ only on the two smallest datasets. Results of the other datasets are provided in Figure 11. We can see that while *Two-stage* is visibly less preferable, *Joint* and *Alternating* are close to each other with larger dataset sizes, and this is the reason why we keep only the low-resource datasets in the main paper.

D Analyses of Efficiency Metric

In our experiments, we use *average exit layer* as the metric of efficiency for the following three reasons.

It is linear w.r.t. the amount of computation. Inference time computation in our model occurs in the following parts: the embedding layer, transformer layers, classifiers, and the LTE module (if used). If a layer is chosen, i.e., the exit layer is after

	RTE	MRPC	SST-2	QNLI	QQP	MNLI	STS-B	SICK
BERT-base	5.8	8.4	18.0	110.4	856.8	209.4	33.2	107.8
BERT-large	11.6	17.4	35.9	223.2	1952.8	400.3	61.3	209.8
ALBERT-base	6.5	9.4	18.5	114.6	864.8	204.8	31.6	104.2
DistilBERT	3.0	4.3	9.1	55.4	407.1	106.2	15.9	50.9

Table 4: Inference runtime in seconds for each model and dataset.

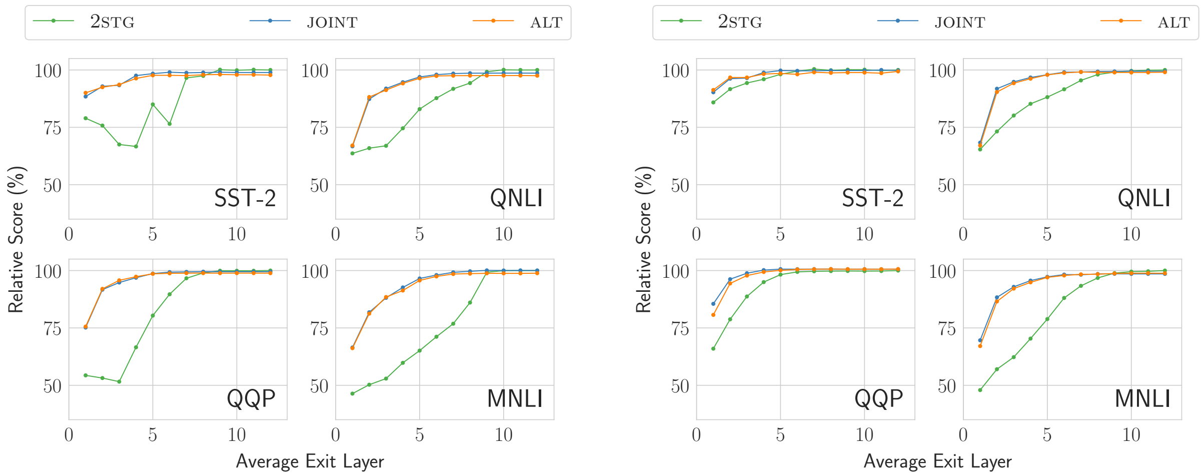


Figure 11: Additional layer-wise score for different fine-tuning strategies on RoBERTa$_{\text{BASE}}$ (left four) and ALBERT$_{\text{BASE}}$ (right four).

it, all components of the layer (transformer, classifier, LTE module) are used and incur computation cost. Additionally, embedding look-up (selecting a column in the matrix) is much faster than the above components (involving matrix-vector multiplication), and can therefore be neglected.

It is stable across different runs. With a fine-tuned model, an inference sample's exit layer only depends on the confidence (or LTE-predicted certainty) at each layer and the threshold. On the other hand, direct measurement of wall-clock runtime is frequently affected by competing processes and fluctuates between different runs.

The computation overhead of early exiting is negligible. With the above reasons, there is only one concern left for using average exit layer as the efficiency metric: how do additional layers in our model (including the additional classifier and possibly the LTE module) compare with transformer layers in the original BERT paper? We estimate FLOPS used in one sample's inference as follows. Since we will eventually end up with orders of magnitude differences, we use the *big-theta* asymptotic notation for estimation.

Most computation is incurred for matrix and vector multiplication. Using the naïve implementation, the cost for multiplying two vectors in $\mathbb{R}^d$ is $\Theta(d)$, and the cost for multiplying a matrix in $\mathbb{R}^{d_1 \times d_2}$ and a vector in $\mathbb{R}^{d_2}$ is $\Theta(d_1 d_2)$.

We denote d as the hidden state size of our model (768 for base models and 1024 for large models), c as the number of classes (less than 4 in our experiments), n as the sequence length (typically in the hundreds), and h as the number of heads in multi-head attention (12 for base and 16 for large).

The classifier is a one-layer fully-connected layer, mapping a vector in $\mathbb{R}^d$ to an output in $\mathbb{R}^c$, therefore the cost is $\Theta(cd)$. Similarly, the cost of the LTE module is $\Theta(2d)$, since its output is always a vector in $\mathbb{R}^2$.

The transformer layer mainly consists of multi-head self-attention, a fully-connected layer, and two layer normalization modules. Layer normalization is much faster than the other two and we therefore neglect it. The fully connected layer maps n vectors from $\mathbb{R}^d$ to $\mathbb{R}^d$, therefore the cost is $\Theta(nd^2)$. The multi-head attention[10] computes h individual uni-head attention. For each uni-head attention, the mapping from original query, key, and value

---

[10] Details can be found in the paper by Vaswani et al. (2017).

vectors ($\mathbb{R}^d$) to head-specific ones ($\mathbb{R}^{d/h}$) incurs a cost of $\Theta(nd^2/h)$; calculating the attention results incurs a cost of $\Theta(nd/h)$. Therefore the total cost here is $\Theta(nd^2 + nd) = \Theta(nd^2)$. Finally, results of each head are combined and one more matrix–vector multiplication is needed, incurring a cost of $\Theta(nd)$. The total cost of one transformer layer is therefore $\Theta(nd^2)$.

Comparing the above, the ratio of a transformer layer to a classifier/LTE module is

$$\frac{\Theta(nd^2)}{\Theta(cd + 2d)} = \Theta(nd). \tag{12}$$

Considering the value of n and d, the classifier and the LTE module are several orders of magnitude lighter than the transformer layer. Even with advanced algorithms and parallel hardware that may accelerate transformer layers, we can still safely come to the conclusion that the computation overhead of early exit is negligible.

Telling BERT's Full Story: from Local Attention to Global Aggregation

Damián Pascual, Gino Brunner, Roger Wattenhofer
Department of Electrical Engineering and Information Technology
ETH Zurich, Switzerland
{dpascual,brunnegi,wattenhofer}@ethz.ch

Abstract

We take a deep look into the behaviour of self-attention heads in the transformer architecture. In light of recent work discouraging the use of attention distributions for explaining a model's behaviour, we show that attention distributions can nevertheless provide insights into the local behaviour of attention heads. This way, we propose a distinction between local patterns revealed by attention and global patterns that refer back to the input, and analyze BERT from both angles. We use gradient attribution to analyze how the output of an attention head depends on the input tokens, effectively extending the local attention-based analysis to account for the mixing of information throughout the transformer layers. We find that there is a significant mismatch between attention and attribution distributions, caused by the mixing of context inside the model. We quantify this discrepancy and observe that interestingly, there are some patterns that persist across all layers despite the mixing.

1 Introduction

The inception of the transformer architecture has sparked significant progress across a wide range of language understanding tasks. Variants of transformers currently dominate the popular GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a) benchmarks and have even achieved super human performance on multiple tasks. The main innovations behind the transformer architecture are the stacking of self-attention layers into a multi-layer self-attention architecture, as well as an unsupervised pre-training phase that primes the model to be fine-tuned on a wide range of language tasks. Transformers and other self-attention-based models have been successfully adopted in other areas such as computer vision (Parmar et al., 2018), music processing (Huang et al., 2019) or protein research (Rao et al., 2019). Their extraordinary empirical success has led researchers to investigate transformers in order to better understand the source of this success, but also in an attempt to explain model decisions.

Much of the research around interpretability and explainability is focused on analyzing the self-attention operation (Clark et al., 2019). In multi-layer self-attention, every input computes an attention distribution over itself and all other inputs to produce ever more complex feature representations. In the case of language, a word in a sentence attends to itself and to all other words in order to compute an updated contextual representation of itself. It is tempting to directly rely on attention distributions to explain the model's predictions. The rationale is that if the attention distribution aligns with human intuition, we can conclude that the model learned robust features and obtained a deep understanding of language, in contrast to simply overfitting on spurious patterns. For example, if a transformer classifies an online comment as hate speech, but we find that the model mostly attended to neutral or even positive words, we would conclude that the model did not actually understand the text and that the correct prediction was either due to chance or to the exploitation of an underlying statistical bias in the data (Niven and Kao, 2019).

However, recent studies (Brunner et al., 2020; Pruthi et al., 2019) question the ability of attention maps to provide a faithful explanation of the inner workings of transformer models. In particular, when the explanations refer to the model input, attention maps do not account for the mixing of information throughout the model. Since self-attention mixes information among all input tokens, the hidden layers attend over mixtures of tokens. Therefore, attention maps may be useful to investigate the local behavior of attention heads but not to draw conclusions about how input tokens relate to each other.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 105–124
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

In this work we take a detailed look at the inner workings of BERT's attention heads, both by analyzing the self-attention distributions, and by using gradient attribution to account for the mixing of tokens throughout the model. We first show that self-attention distributions correlate strongly with Hidden Token Attribution (Brunner et al., 2020) (HTA) from hidden embedding to head output, this result validates HTA. We then present novel location based attention patterns, revealing that BERT, despite its bi-directional language modeling objective, attends to past embeddings in earlier layers, and to future ones in later layers. Next, we use HTA in order to extend the analysis to take the mixing of information into account, which allows to draw conclusions about the behaviour of an attention head with respect to the original input word. The patterns that emerge are different from the local attention-based patterns, giving us deeper insight into the operation of the model and emphasizing that local attention-based explanations are very different from global attribution-based explanations. Finally, we contrast attention and HTA distributions for individual examples. Our results further highlight the discrepancy between local attention patterns and global attribution patterns.

2 Related Work

The good performance of attention (Graves, 2013; Bahdanau et al., 2015) models in Natural Language Processing (NLP) arises from their ability to learn alignments between words. The transformer architecture (Vaswani et al., 2017) is a multi-layer multi-head self-attention architecture that is pre-trained in an unsupervised manner. The extraordinary performance of transformer models has accelerated progress in the field of NLP. Currently, there is a growing number of different transformer models that vary in size, pre-training objective and/or other architectural elements (Radford et al., 2018, 2019; Liu et al., 2019; Lan et al., 2020; Yang et al., 2019; Sanh et al., 2019; Kitaev et al., 2020; Raffel et al., 2019).

The success of transformers and the possibility of visualizing attention distributions (Vaswani et al., 2017), has motivated a line of research aiming to understand the inner workings of transformers and explain their decisions. Many of these studies have focused on BERT (Devlin et al., 2019a), a well-known transformer model, leading to a body of research grouped under the term BERTology (Rogers et al., 2020).

Aforementioned research builds on previous work on the interpretability of attention distributions in other models apart from transformers. In particular, Jain and Wallace (2019) examine the attention distributions of LSTM based encoder-decoder models and show a weak to moderate correlation between attention and dot-product gradient attribution. Furthermore, they show that adversarial attention distributions that do not change the model's decision can be constructed. In the same line, Serrano and Smith (2019) find, through zeroing out attention weights, that gradient attribution is a better predictor of feature importance with respect to the model's output than attention weights. Wiegreffe and Pinter (2019) find that although adversarial attention distributions can be easily obtained, they perform worse on a simple diagnostic task. All of these works raise concerns about the ability of attention distributions to explain the decisions of a model.

Despite existing concerns surrounding the interpretability of attention distributions, very few works have studied how this problem affects transformers. Pruthi et al. (2019) show that, just as in other attention models, it is possible to manipulate self-attention in transformers in order to generate different attention masks that cause only a small drop in performance. Brunner et al. (2020) find that attention distributions are not unique when the sequence length is larger than the head dimension and show that this can lead to the discovery of spurious patterns. Furthermore, they show that although it is possible to map hidden tokens back to their corresponding input tokens, there is a very large degree of information mixing inside the model, which raises questions about straightforward interpretations of attention maps. Recently, (Abnar and Zuidema, 2020) proposed a method to quantify information flow inside transformers. This method tracks the mixing of information due to attention but omits the effect of feed-forward networks.

Our work addresses this important issue by distinguishing between local and global aggregation patterns, where the former can be explained by attention distributions and the latter by attribution. We analyze BERT from both angles and quantify the mismatch between these interpretations. We show that attention correlates well with attribution locally but not globally and therefore attention maps are inadequate to draw conclusions that refer to the input of the model.

3 Background on Transformers

The original transformer architecture (Vaswani et al., 2017) is a sequence-to-sequence model consisting of an encoder and a decoder, both of which follow a multi-layer multi-head self-attention structure. Conversely, most of the pre-trained transformer models that can be fine-tuned on supervised language understanding tasks only consist of an encoder. Each transformer layer consists of a self-attention block and a non-linear feed forward block (MLP) with layer normalizations (Ba et al., 2016).

The input to a transformer layer is a sequence of embeddings $E^l = [\boldsymbol{e}^l_0, ..., \boldsymbol{e}^l_{d_s}] \in \mathbb{R}^{d_e \times d_s}$, where l denotes the layer index, d_e is the embedding dimension, and d_s is the sequence length. We refer to the sequence of non-contextual input word embeddings as E^0, and to the hidden contextual embeddings as E^l, where $l > 0$. Note that E^0 refers to the word embeddings after position and sequence embeddings have been added. A self-attention block consists of n_h separate attention heads. The attention heads independently perform the self-attention operation, and the results are then concatenated and projected back into the embedding space by a linear layer. The output of the attention block is then fed into the MLP.

The self-attention operation itself is implemented by projecting each input token $\boldsymbol{e}_i \in \mathbb{R}^{d_e}$ into a query vector $\boldsymbol{q}_i \in \mathbb{R}^{d_q}$, key vector $\boldsymbol{k}_i \in \mathbb{R}^{d_q}$ and value vector $\boldsymbol{v}_i \in \mathbb{R}^{d_v}$. We present the self-attention operation from the perspective of a single token $\boldsymbol{e}_i$ attending to all input tokens. For that, the key vectors $\boldsymbol{k}_i$ are aggregated into the key matrix $K = [\boldsymbol{k}_0, ..., \boldsymbol{k}_{d_s}] \in \mathbb{R}^{d_q \times d_s}$ and the value vectors $\boldsymbol{v}_i$ are aggregated into the value matrix $V = [\boldsymbol{v}_0, ..., \boldsymbol{v}_{d_s}] \in \mathbb{R}^{d_v \times d_s}$. The attention distribution $\boldsymbol{a}_i$ of token $\boldsymbol{e}_i$ over all input tokens is then computed as

$$\boldsymbol{a}_i = \text{softmax}\left(\frac{\boldsymbol{q}_i^T \cdot K}{\sqrt{d_q}}\right)$$

The attention vector $\boldsymbol{a}_i \in \mathbb{R}^{d_s}$ now contains an attention weight for each input token. $\boldsymbol{a}_i$ is then multiplied with the value matrix V to compute the output of the self-attention operation for a token i and a head h as

$$\boldsymbol{o}_{h,i} = V \cdot \boldsymbol{a}_i$$

The outputs of all heads $\{\boldsymbol{o}_{0,i}, ..., \boldsymbol{o}_{n_h,i}\} \in \mathbb{R}^{d_e}$ are then concatenated and fed through a linear layer to compute the output of the self-attention block for a single token. This linear layer can be thought of as an aggregation operation that projects the output of the independent heads back into embedding space. In practice, the attention distributions for all tokens are computed in parallel.

4 Extending Hidden Token Attribution

Hidden Token Attribution (Brunner et al., 2020) is a gradient-based attribution method that quantifies how much information from each input token is contained in a given hidden embedding. For each layer l, this method defines the relative contribution $c^l_{i,j}$ of an input token $\boldsymbol{e}^0_i$ to a hidden embedding $\boldsymbol{e}^l_j$ as:

$$c^l_{i,j} = \frac{||\nabla^l_{i,j}||_2}{\sum_{k=0}^{d_s} ||\nabla^l_{k,j}||_2} \quad \text{with} \quad \nabla^l_{i,j} = \frac{\partial \boldsymbol{e}^l_j}{\partial \boldsymbol{e}^0_i} \quad (1)$$

The contribution $c^l_{i,j}$ is normalized by the sum of the attribution values to all input tokens and hence, ranges between 0 and 1.

In this work, we apply Hidden Token Attribution to the individual attention heads of BERT. For a token $\boldsymbol{e}^l_j$ at layer l we back-propagate the gradients from the output $\boldsymbol{o}^l_{h,j}$ of each attention head h independently. This differs from the original method in that Hidden Token Attribution propagates the gradients from the layer output. In general, using Equation 1, we can compute the contribution between any two vectors in the model, as long as they are connected in the computation graph. We hence denote the contribution of any vector $\boldsymbol{x}$ to another vector $\boldsymbol{y}$ as $C(\boldsymbol{x}, \boldsymbol{y})$.

In particular, we calculate two different contributions to the head output:

Previous layer contribution: Contribution from the hidden embeddings at the input of the attention head to the output of the attention head: $C(\boldsymbol{e}^{l-1}_i, \boldsymbol{o}^l_{h,j})$

Input contribution: Contribution from tokens at the input of the transformer model to the output of an attention head h at layer l: $C(\boldsymbol{e}^0_i, \boldsymbol{o}^l_{h,j})$

Previous layer contribution allows us to study how attention heads operate locally and how HTA distributions compare to attention distributions. *Input contribution* enables us to extend the head attention patterns all the way back to the input, thereby controlling for the effect of information mixing.

5 Setup

For our experiments we use the non-finetuned, uncased BERT base model (Devlin et al., 2019b) as provided in the original repository.[1] Despite the recent explosion of new transformer variants, BERT remains the most popular model for research into the interpretability of transformer models. The reason for this is that most of the newer models are architecturally similar to BERT, and therefore, studies carried out on BERT either are likely to generalize to these models or can be repeated with relatively little effort.

We perform our experiments on 1800 examples from the development set of the MNLI matched (MNLIm) dataset. Brunner et al. (2020) show that when the sequence length d_s is larger than the head output dimension d_v, the attention distributions are not identifiable. Therefore, to guarantee that in our experiments we do not find spurious patterns that do not influence downstream parts of the model, we restrict the examples in our dataset to sequences of maximum length of 64 tokens, which is the head dimension of BERT. Thus, the examples in our dataset have sequence lengths ranging between 6 and 64 tokens, with a median length of 34 tokens. In total, this subset contains 63,456 tokens.

6 HTA: Local Validation

The ability of attention distributions to provide explanations has been the target of a number of studies (Wiegreffe and Pinter, 2019; Serrano and Smith, 2019; Pruthi et al., 2019). In particular, Jain and Wallace (2019) show that attention distributions do not explain the model output and do not correlate well with attribution methods. However, if we are exclusively interested in how attention heads behave locally, i.e., without considering their impact on the model's decisions, it is sound to examine attention distributions. The reason for this is that self-attention is the only operation performed in attention heads, and hence, attention distributions precisely represent the information flow within the heads. As a consequence, we can use attention distributions as a reference to validate whether HTA accurately quantifies how information mixes within transformers. To verify this, we compare attention distributions to previous layer contribution by computing the correlation between attention maps $\boldsymbol{a}_{h,i}$ and the contribution $C(\boldsymbol{e}_i^{l-1}, \boldsymbol{o}_{h,j}^{l})$ for each head.

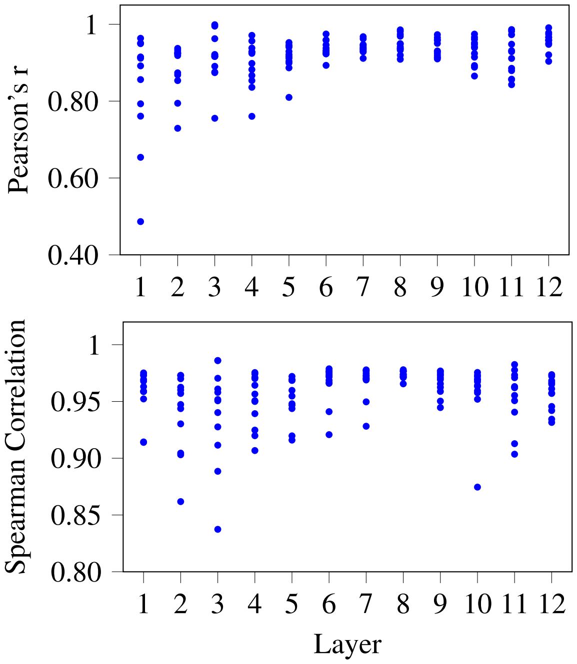


Figure 1: (Top) Pearson and (Bottom) Spearman correlation between attention and previous layer contribution.

A high correlation value would validate HTA as accurately representing the flow of information within transformers. To calculate the correlation, first, we extract the attention maps for all the heads of BERT for each of the tokens in the examples of our dataset. Then, we pair each attention map to the corresponding contribution. Note, that both attention maps and contributions are distributions that lie in the probability simplex, i.e., all the values are between 0 and 1 and their sum is 1. Next, we calculate Pearson's correlation coefficient for each attention-contribution pair and we aggregate the results into one value per head by computing the mean of the correlation values.

Figure 1 (Top) shows the mean correlation value per head. For all heads except for two, Pearson's correlation coefficient is larger than 0.7. Furthermore, 90% of the heads show a correlation between attention and Hidden Token Attribution of over 0.85. Similarly, we calculate Spearman's rank correlation coefficient r for each head. The results, displayed in Figure 1 (Bottom), show that only four heads have a Spearman's correlation smaller than 0.9, and that 75% of the heads have a correlation

---

[1] https://github.com/google-research/bert

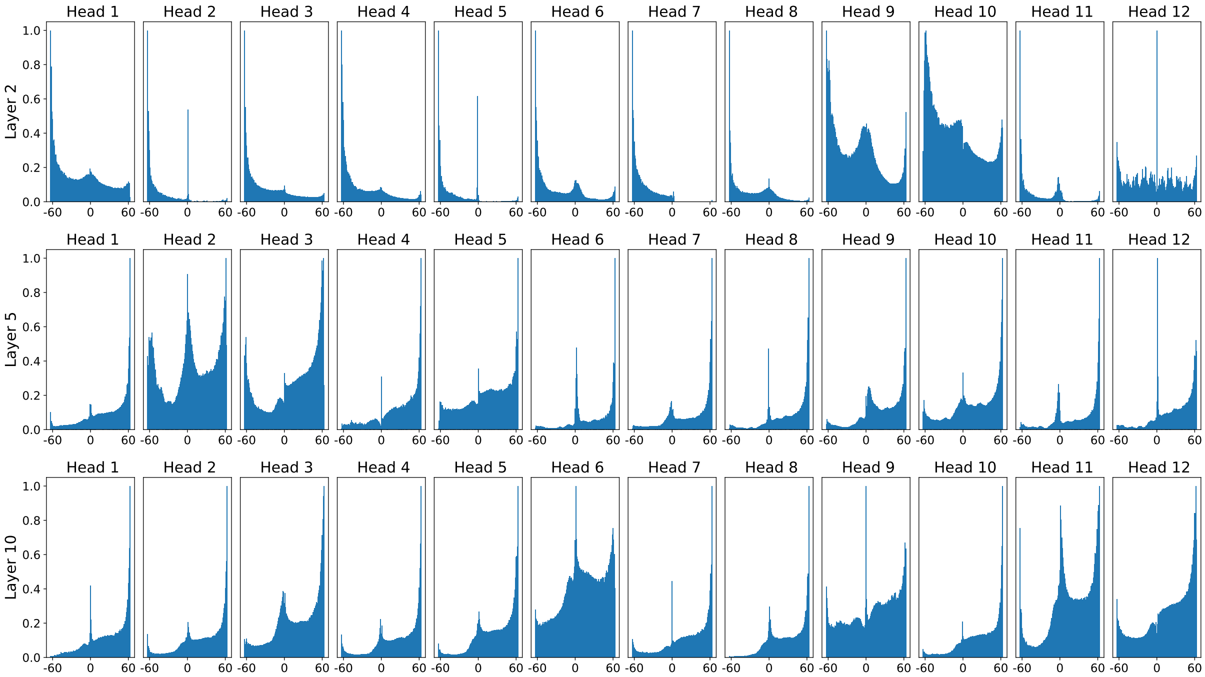


Figure 2: Attention histograms for layers 2, 5 and 10 of BERT. The horizontal axis represents the relative position of the attended tokens with respect to the attending token placed at position 0. Given that the maximum sequence length is 64 the horizontal axis ranges from -63 to 63. The vertical axis is normalized to the maximum value for a better visualization.

coefficient larger than 0.95. Note that in any case gradient attribution is a local first order approximation and thus introduces a small error that prevents perfect correlation.

These high correlation values empirically demonstrate that HTA does indeed represent the flow of information within attention heads with respect to the head inputs. Therefore, to study the inner workings of transformers beyond attention heads one can rely on Hidden Token Attribution and apply it at different points of the model. Now that we have validated HTA, we can investigate the behavior of the heads in more detail: examining the local patterns revealed by attention, the global patterns revealed by HTA, and the discrepancies between both.

7 Local Head Analysis

In this section we take a closer look into the local behaviour of attention heads. Here, local means that we analyze how the intermediate tokens fed into the heads are processed, as opposed to how the model input propagates. To this end, we study attention distributions, but rather than studying each individual example, we aggregate the attention distributions, thus obtaining a general picture of how each head behaves. In particular, we study how much attention is paid to tokens in each relative position with respect to the attending token.

For each head, we extract the attention maps for each token. Then, we define the position of the attending token in the sentence as the origin ($x = 0$), thereby generating a histogram where the horizontal axis represents the position of the neighbours and the vertical axis the amount of attention paid to a token. We sum the histograms of all tokens and then normalize the result. To normalize, we divide the value of attention at each position by the number of times that a token is at that relative position; given that the median length of the examples is 36, this normalization ensures that distant positions are not penalized for having fewer occurrences.

Figure 2 presents the histograms for the heads in layers 2, 5 and 10, the other layers can be found in Appendix A. From these histograms, a clear pattern is observable. In the first layers, heads tend to aggregate more information from past tokens than from future tokens. In fact, the attention of heads 2, 5 and 7 in Layer 2 to future tokens is negligible. However, this trend quickly reverses with increasing depth, and in later layers the aggregation of future hidden embeddings dominates for most heads. To illustrate this, we calculate the center of mass of the attention histograms per layer by

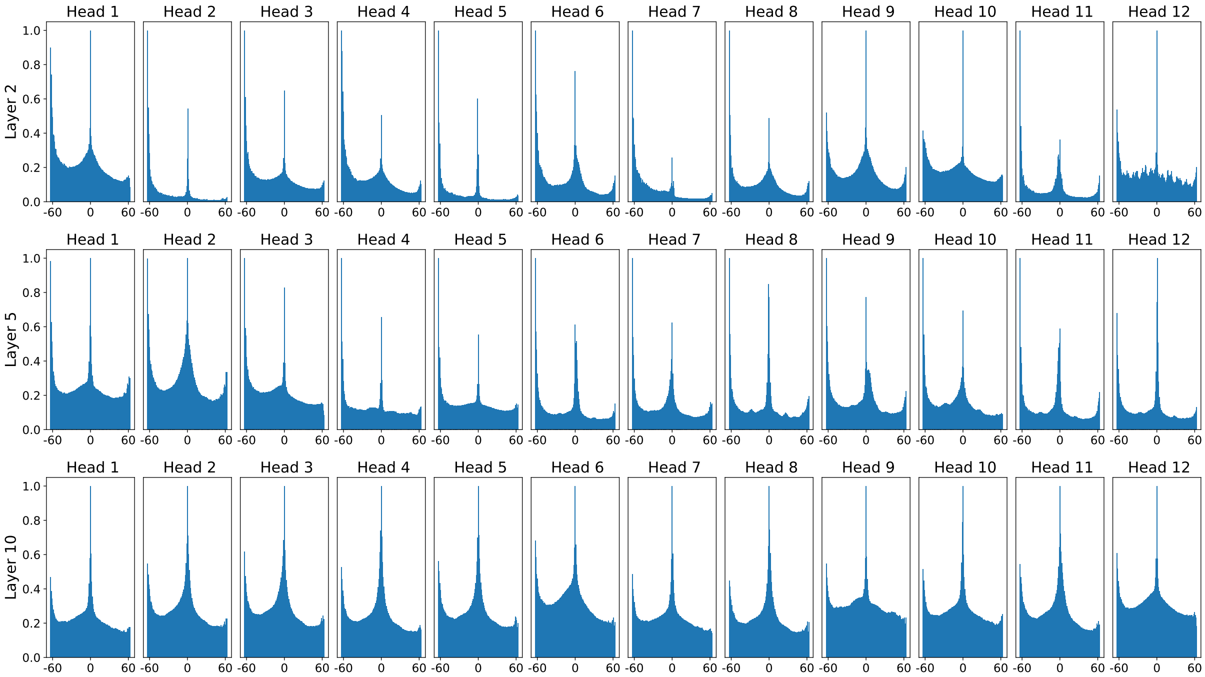


Figure 3: Input contribution histograms for layers 2, 5 and 10 of BERT. The horizontal axis represents the relative position of the attended tokens with respect to the attending token placed at position 0. The vertical axis is normalized to the maximum value for a better visualization.

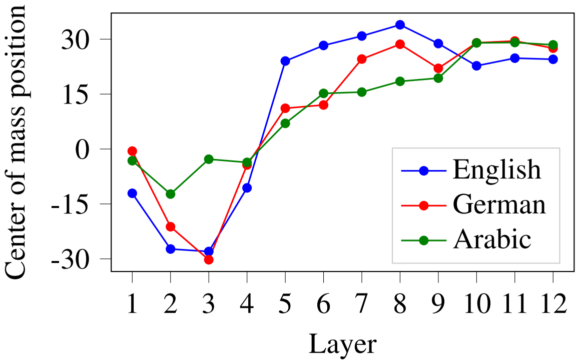


Figure 4: Mean center of mass for attention histograms per layer for English, German and Arabic.

averaging over each head. The results are shown in Figure 4. To understand whether this behavior is particular to English we perform the same calculations on the German (Deepset, 2019) and Arabic (Antoun et al., 2020) versions of BERT on 1800 examples of the XNLI dataset (Conneau et al., 2018); note that although Arabic is a right-to-left language, to process the examples the token order is reversed and input to BERT as left-to-right.

The figure shows that BERT follows the same trend regardless of the language, i.e., the models first attend to past and then to future hidden tokens. This suggests that despite its bidirectional training, BERT tends to handle language like humans, from left to right. This is also inline with the sequential nature of language, i.e., the past context needs to be known to understand the future context.

8 Global Head Analysis

Although attention maps are an effective tool to understand the local behavior of attention heads, drawing conclusions that refer to the input words can be misleading. Transformers are complex models that mix information from the entire input sequence at each layer. Recent work (Brunner et al., 2020; Pruthi et al., 2019) has raised concerns about the interpretability of attention maps as representative of global context aggregation. In this section, we look into the individual heads and study what we call global patterns, i.e., aggregation patterns that refer to the model's input.

To this end, we follow the same procedure we use in the previous section, but to generate input contribution $C(\boldsymbol{e}_i^0, \boldsymbol{o}_{h,j}^l)$ histograms instead of attention histograms. In Figure 3 we show the histograms for layers 2, 5 and 10, i.e., the same layers as in Figure 2. The histograms for the whole model can be found in Appendix B. Furthermore, we calculate the center of mass per layer and compare them to the attention centers of mass in Figure 5. The histograms and the centers of mass show that

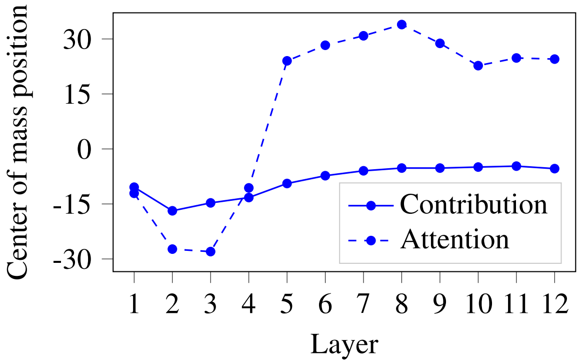


Figure 5: Mean center of mass for contribution vs. attention histograms per layer for English.

the global pattern of context aggregation is much more uniform than shown by the attention maps, especially after layer 4. This is intuitive: given that in the first layers the heads are attending mostly to the past context, on average, all the hidden tokens have a larger amount of past context. Therefore, when in later layers the attention shifts to the future hidden tokens, the past context already contained in these tokens balances the contribution, resulting in a uniform pattern of global context aggregation.

The difference in the patterns revealed by this global analysis and the local head analysis from the previous section shows a strong mismatch between attention distributions and global context aggregation in attention heads. In fact, local attention patterns can easily lead to spurious conclusions when used to interpret global context aggregation. Next, we study this difference quantitatively.

9 Local Attention vs. Global Attribution

To quantify the discrepancy between attention distributions and input contribution, i.e., local and global patterns of context aggregation, we calculate the correlation between attention maps and input contribution $C(\boldsymbol{e}_i^0, \boldsymbol{o}_{h,j}^l)$. We follow the same methodology as in Section 6 and report Pearson's and Spearman's correlation coefficient in Figure 6. In line with the mismatch between attention and contribution histograms (Figures 2 and 3), we observe how the correlation between attention and input contribution quickly decreases in deeper layers. Particularly, after only four layers Pearson's correlation coefficient for most heads is smaller than 0.5 and in the last four layers the median head correlation value is smaller than 0.25. Furthermore, Spearman's correlation follows a very similar trend, with the median head correlation value falling under 0.7 already at layer 3, and under 0.25 at the last layer.

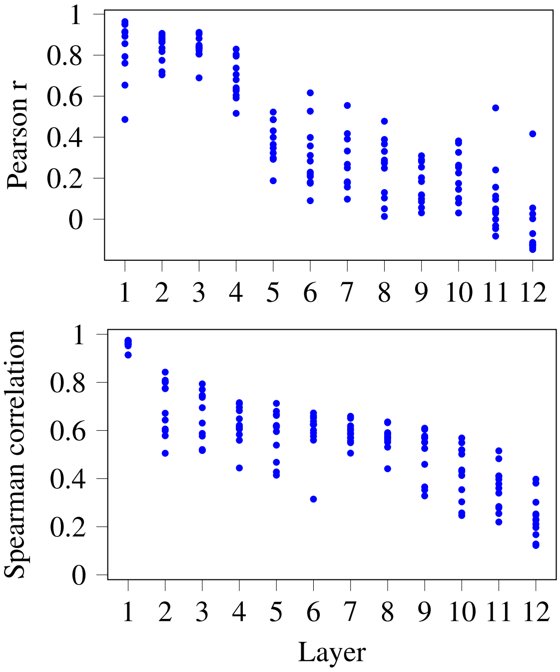


Figure 6: (Top) Pearson and (Bottom) Spearman correlation between attention and input contribution.

The results from this section point at the importance of information mixing: attention maps show how the heads behave locally, i.e., how they aggregate context, but not *what* context is in fact aggregated. Knowing how the heads behave locally can give us a better understanding of transformer models that could be leveraged to further improve the performance of these models (Wu et al., 2020). However, attention maps are misleading when drawing conclusions about what input words are being aggregated into the contextual embeddings.

9.1 Specific examples

The histograms studied in the previous sections give us a high level picture of what is happening inside the model. However, we averaged across examples with different sequence length and with different token types in different positions. To gain a more detailed understanding of the model's behaviour, we now look into specific input sequences randomly selected from our dataset.

Kovaleva et al. (2019) study attention maps generated by BERT for many different examples and divide the attention patterns into five types: vertical, diagonal, vertical-diagonal, block and heterogeneous. When looking at the attention maps, we

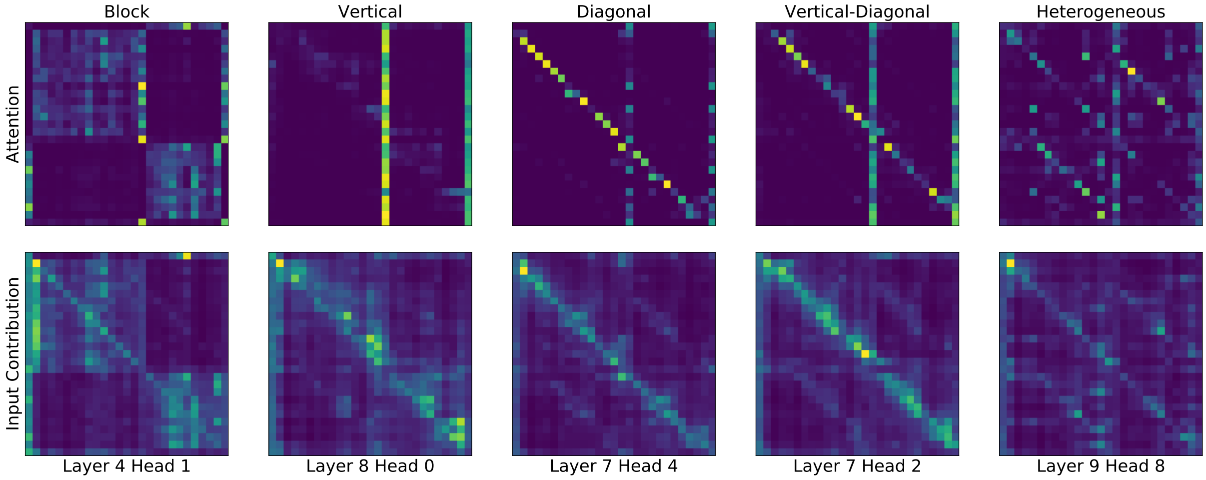


Figure 7: Comparison of the head patterns revealed by attention distributions (upper row) and input contribution (lower row).

observe the same five attention patterns. Nevertheless, to understand what input information these heads are actually aggregating, we need to look at the contribution from the input tokens.

In Figure 7, we compare the five patterns observed by Kovaleva et al. (2019) with the corresponding patterns revealed by Hidden Token Attribution with respect to the input, $C(\boldsymbol{e}_i^0, \boldsymbol{o}_{h,j}^l)$. A comparison for all heads is available in Appendix C. Remarkably, heads with the vertical pattern pay most attention to the SEP and CLS tokens. Nevertheless, the input contribution reveals that SEP tokens are used by the model to store general context, and by extracting information from the SEP token at intermediate layers, the model is in fact aggregating global context. Hence, with respect to the input, heads with vertical, diagonal and vertical-diagonal patterns have a similar behavior to heterogeneous heads. However, tokens around the diagonal tend to contribute the most given the prevalent aggregation of local context.

On the other hand, as shown in the first column of Figure 7, we observe that the block patterns prevail when we apply Hidden Token Attribution to the input. It is noteworthy that, while vertical and diagonal patterns fade away, the block pattern still remains visible. The fact that attending to tokens inside a block results in aggregation of context from within that block implies that up to that point, the context was mainly aggregated from within the blocks separated by SEP. We do not observe block patterns in the contribution maps for layers deeper than layer 4, which suggests that the first layers aggregate context within blocks and later layers aggregate context in a more global manner.

10 Conclusion

We provide justification for using HTA to study information flow within transformers. By studying attention distributions of BERT we uncover an interesting pattern: In earlier layers, attention heads attend mostly to earlier tokens, whereas this trend quickly reverses with increasing depth. This is surprising, since BERT is trained using bi-directional language modeling and it suggests that like humans, BERT understands language from left to right.

A problem with local attention patterns is that they do not reveal how the attention heads process the information contained in the input tokens. We thus use Hidden Token Attribution to compute per-head attribution distributions over the input words. Our results show that the mismatch between attention and attribution distributions increases with depth. This confirms the importance of accounting for information mixing when analyzing attention heads with respect to the input tokens. Finally, we show how five different attention head patterns differ from their token attribution equivalents. Our method and results are complementary to those in (Abnar and Zuidema, 2020), we believe a combination of attention flow and HTA may provide new interesting insights.

In this work, we aim to set a clear border that distinguishes between local and global aggregation in transformer models. This distinction is important when trying to interpret the behavior of these models, and we hope that it will help future studies in their analyses. Furthermore, our findings add new insights to the growing field of research on explaining transformers. This research, in turn, can help in guiding design decisions leading to further improvements of natural language processing architectures.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations (ICLR)*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Deepset. 2019. Open sourcing german bert. `https://deepset.ai/german-bert`. Accessed: 2020-05-18.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. 2019. Music transformer. In *International Conference on Learning Representations*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3543–3556.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4356–4365.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4658–4664. Association for Computational Linguistics.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2019. Learning to deceive with attention-based explanations. *CoRR*, abs/1909.07913.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. 2019. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, pages 9686–9698.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2931–2951.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. *CoRR*, abs/1908.04626.

Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. 2020. Lite transformer with long-short range attention. In *International Conference on Learning Representations*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *https://arxiv.org/abs/1906.08237*.

A Attention Histograms English

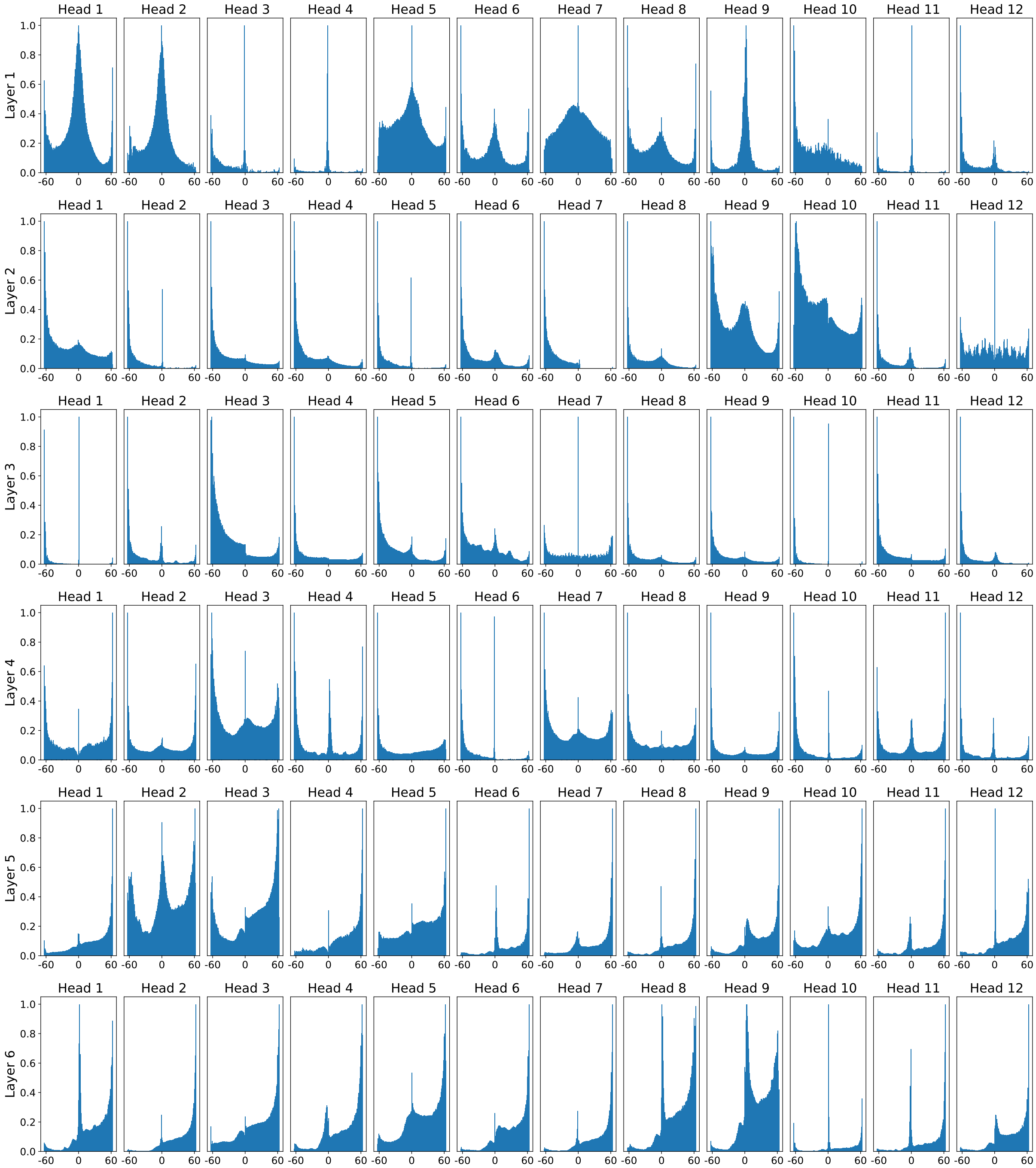


Figure 8: Attention histograms for layers 1 to 6

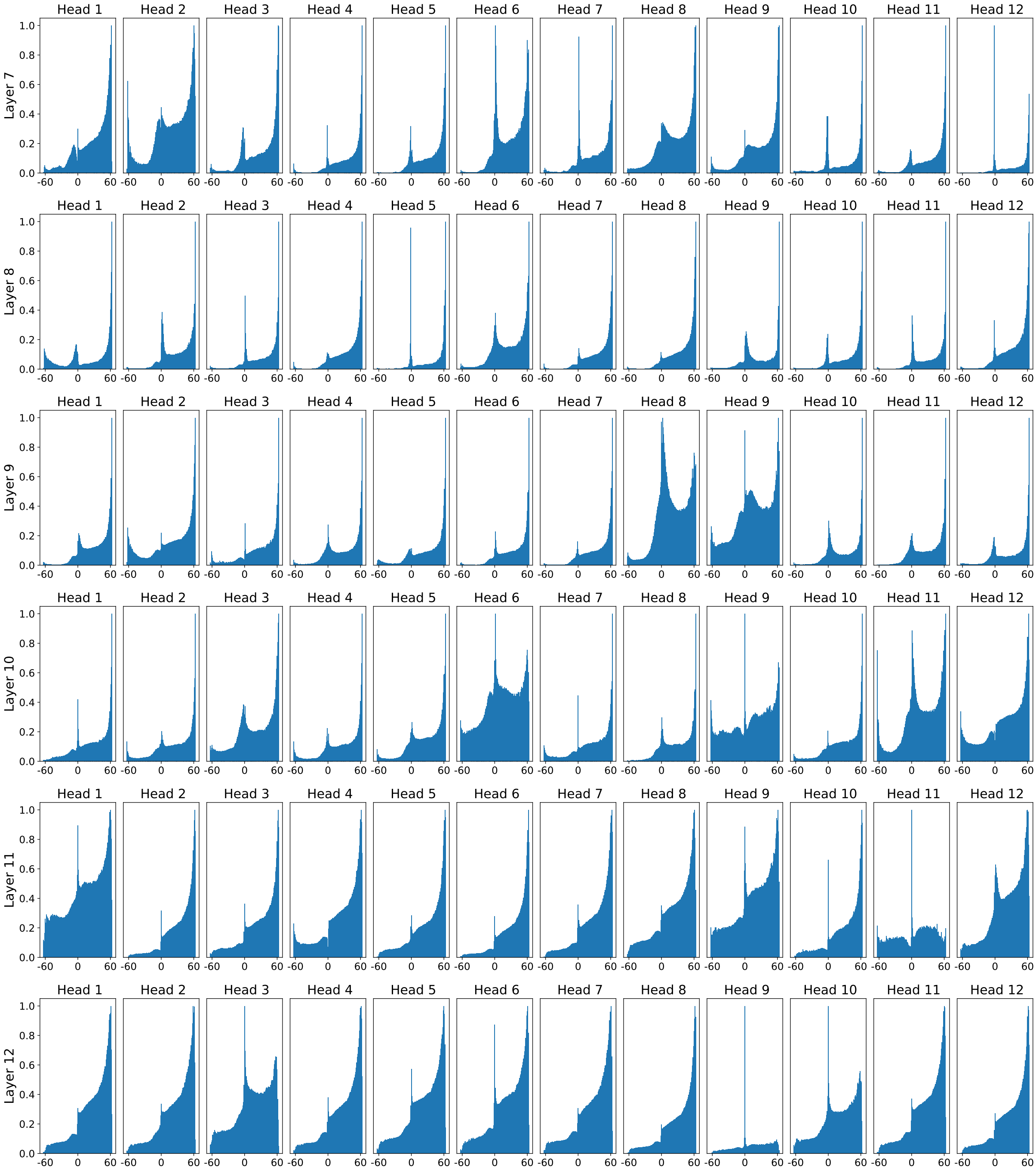


Figure 9: Attention histograms for layers 7 to 12

B Input Contribution Histograms

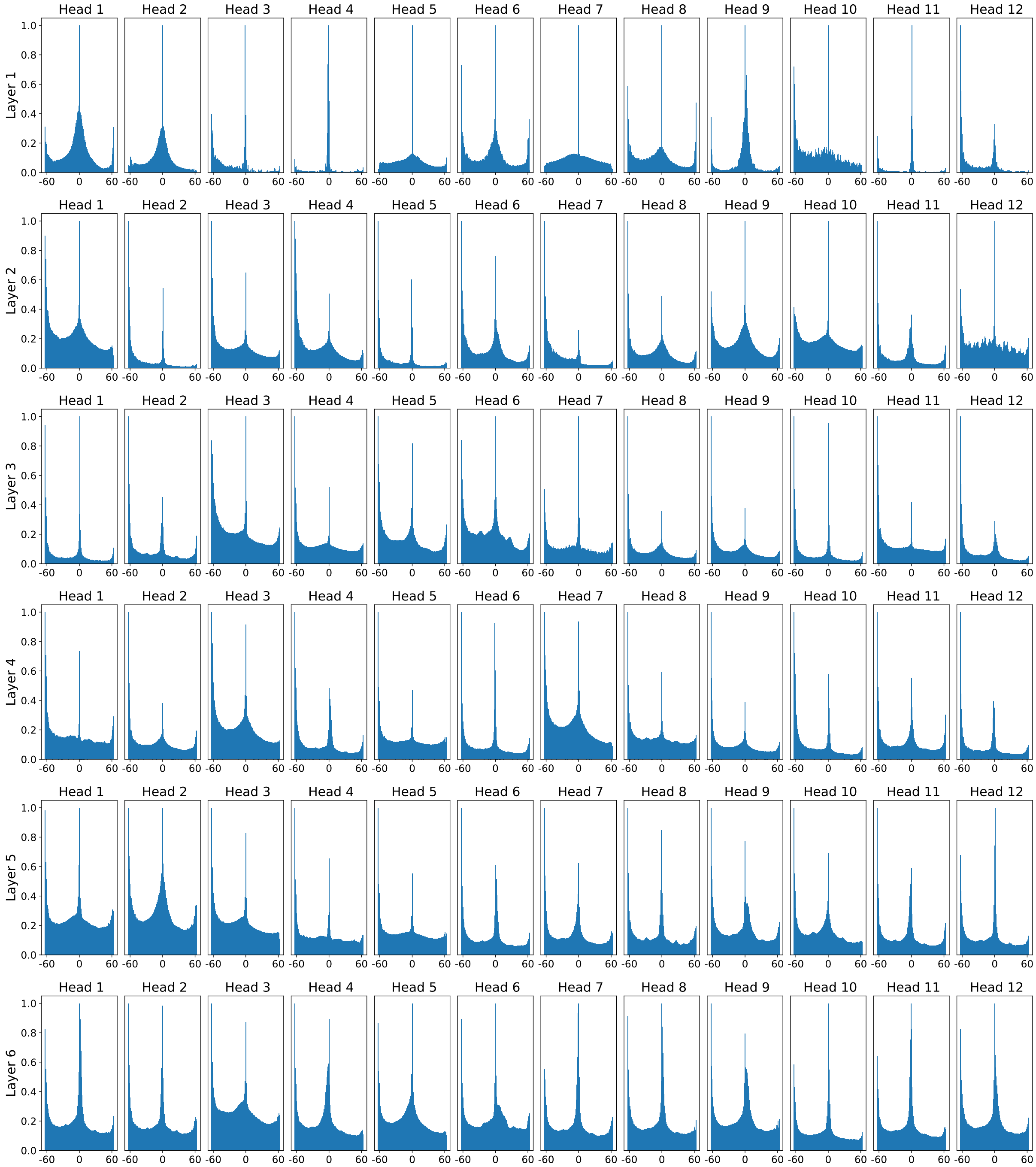


Figure 10: Input contribution histograms for layers 1 to 6

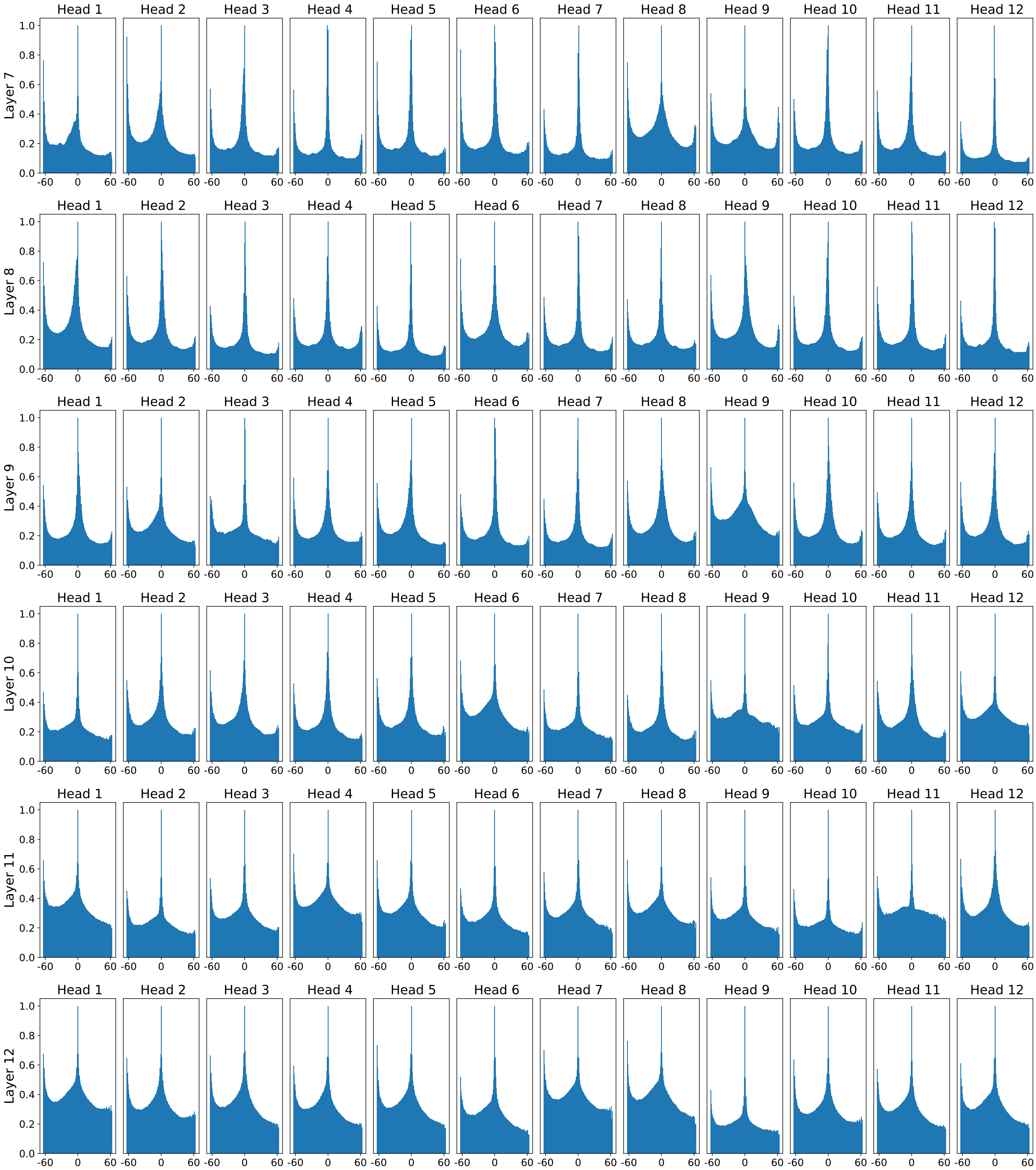


Figure 11: Input contribution histograms for layers 7 to 12

C Comparison of Local vs. Global Head Patterns

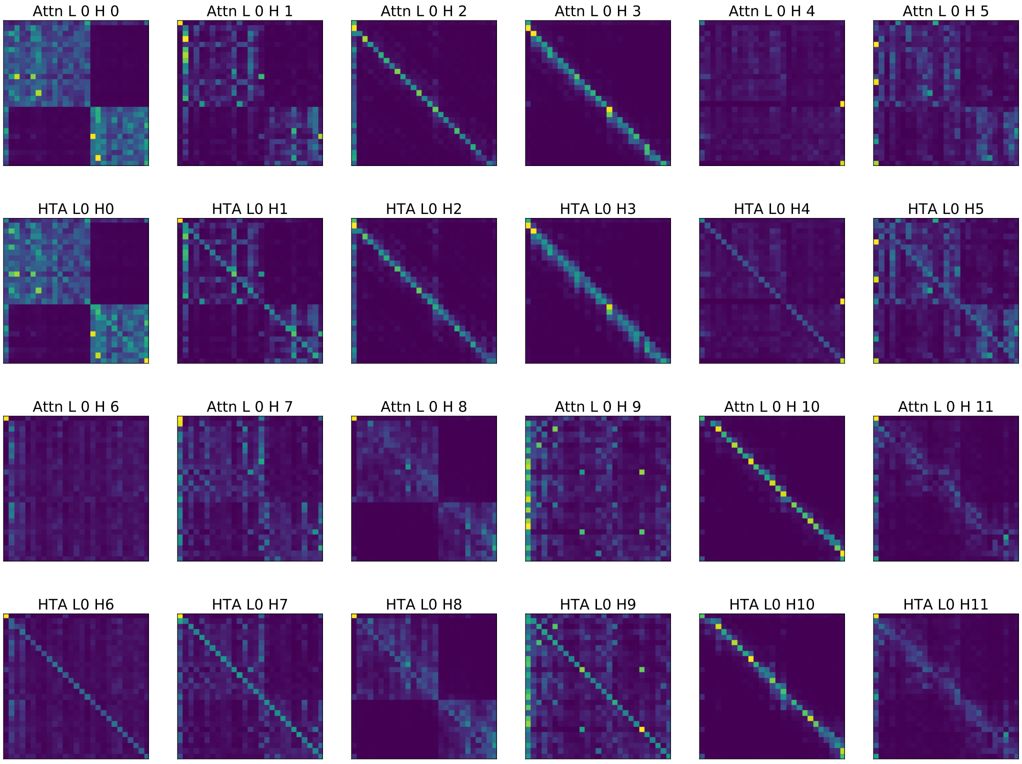


Figure 12: Layer 1: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

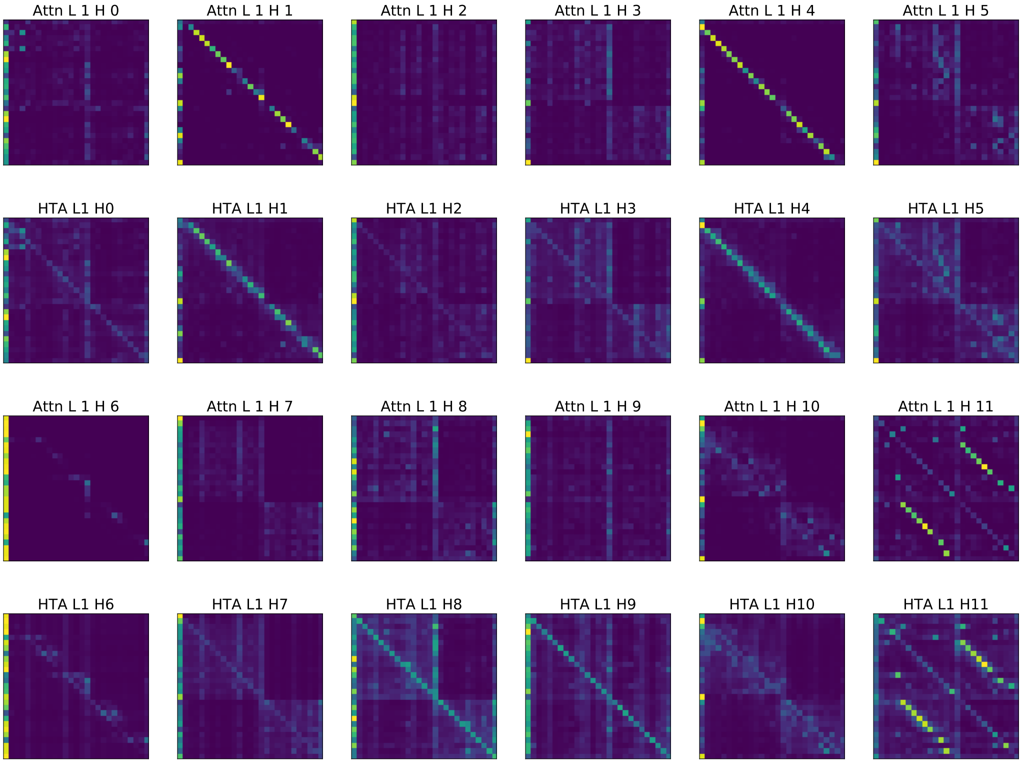


Figure 13: Layer 2: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

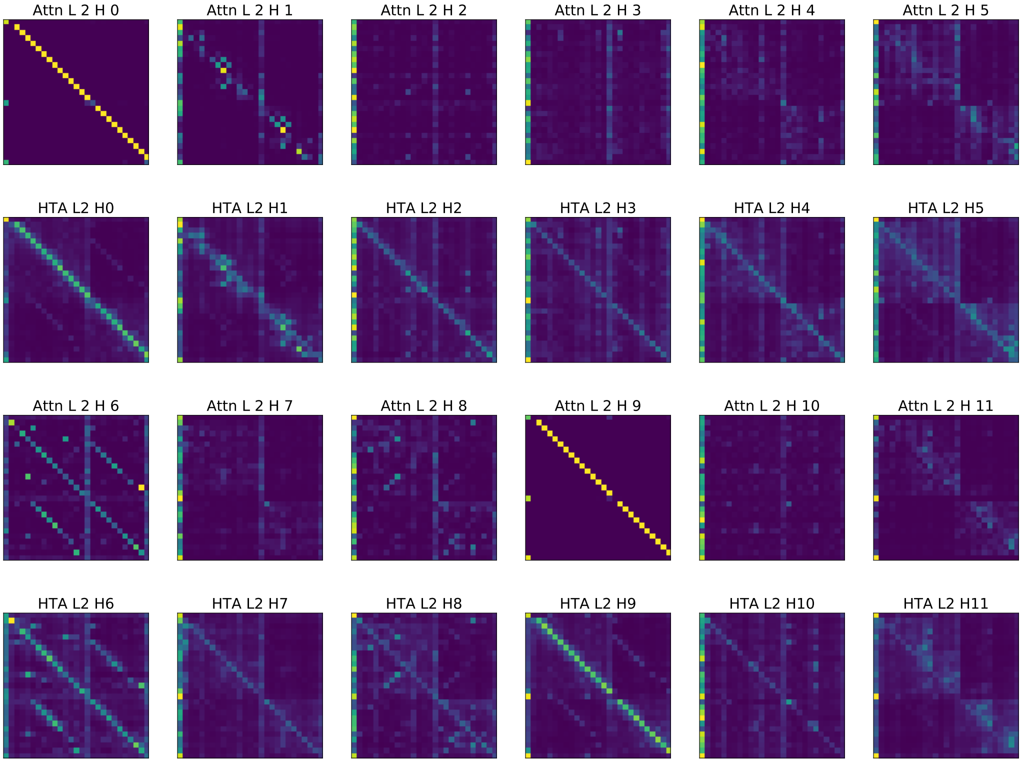


Figure 14: Layer 3: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

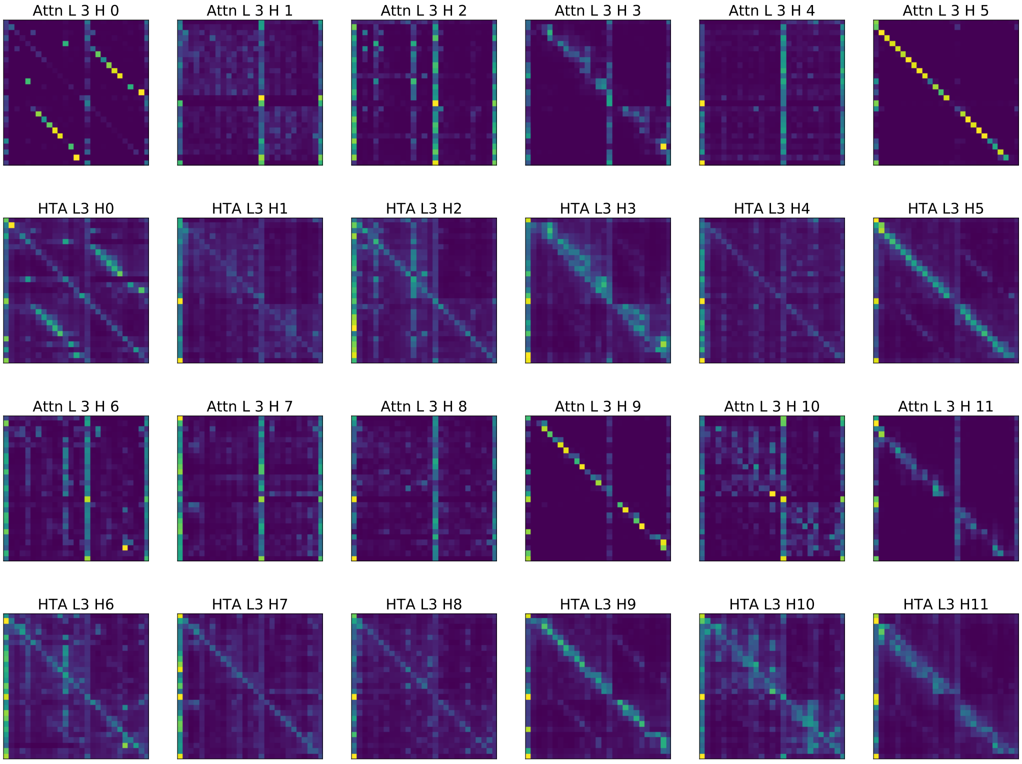


Figure 15: Layer 4: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

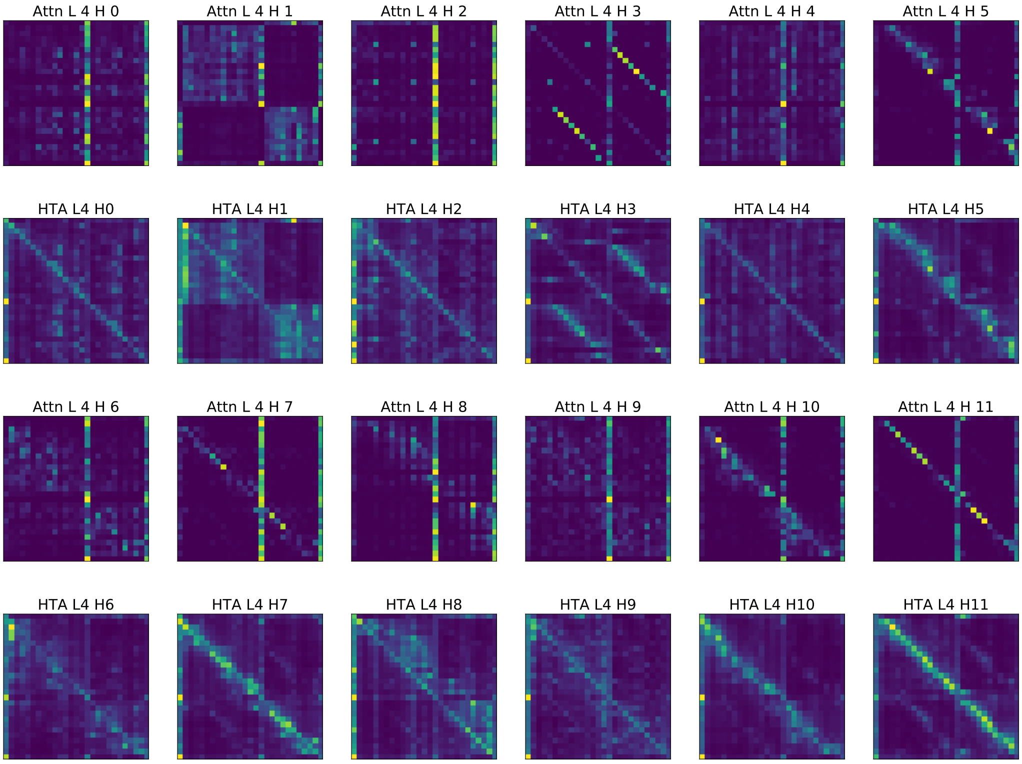


Figure 16: Layer 5: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

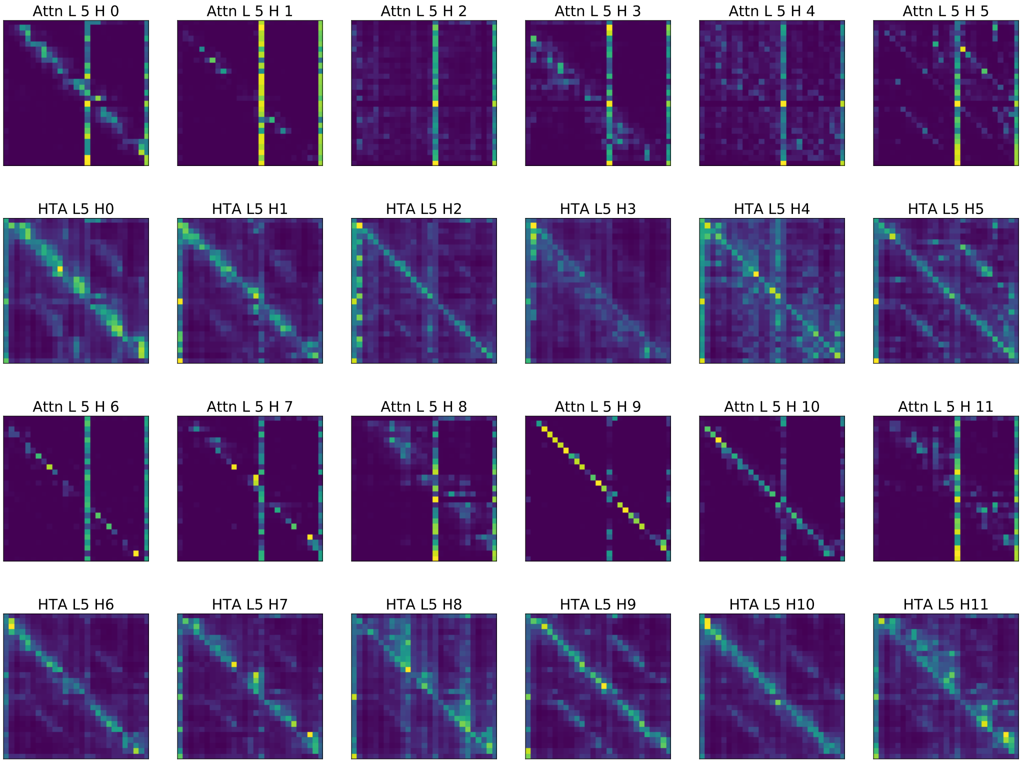


Figure 17: Layer 6: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

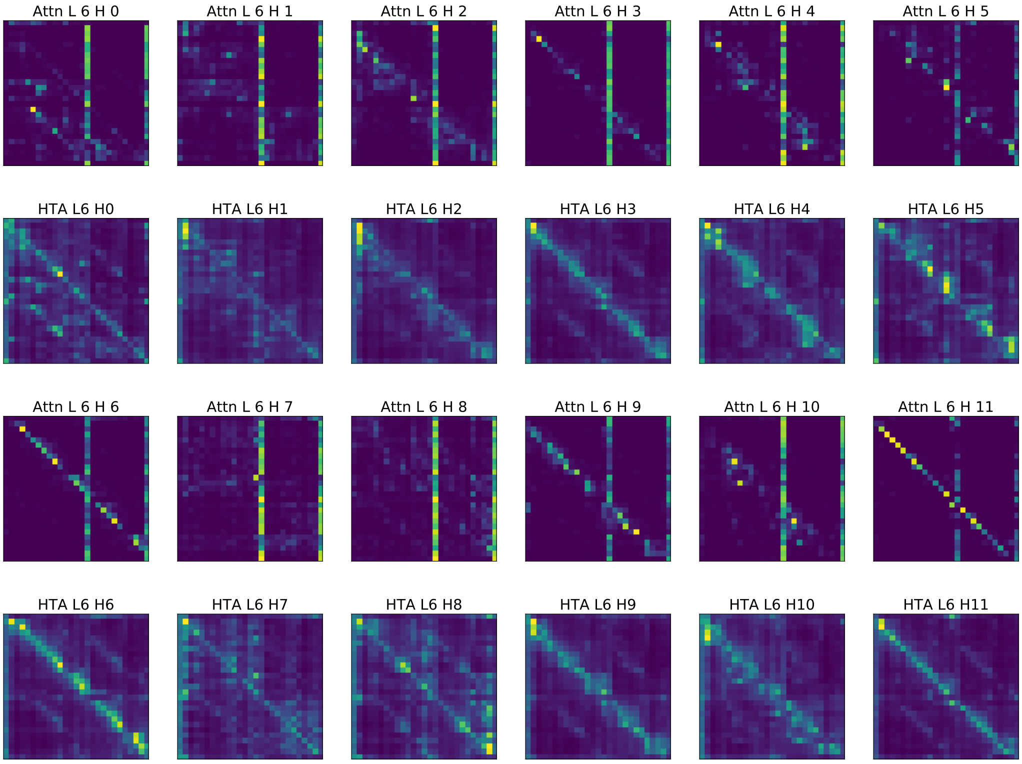


Figure 18: Layer 7: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

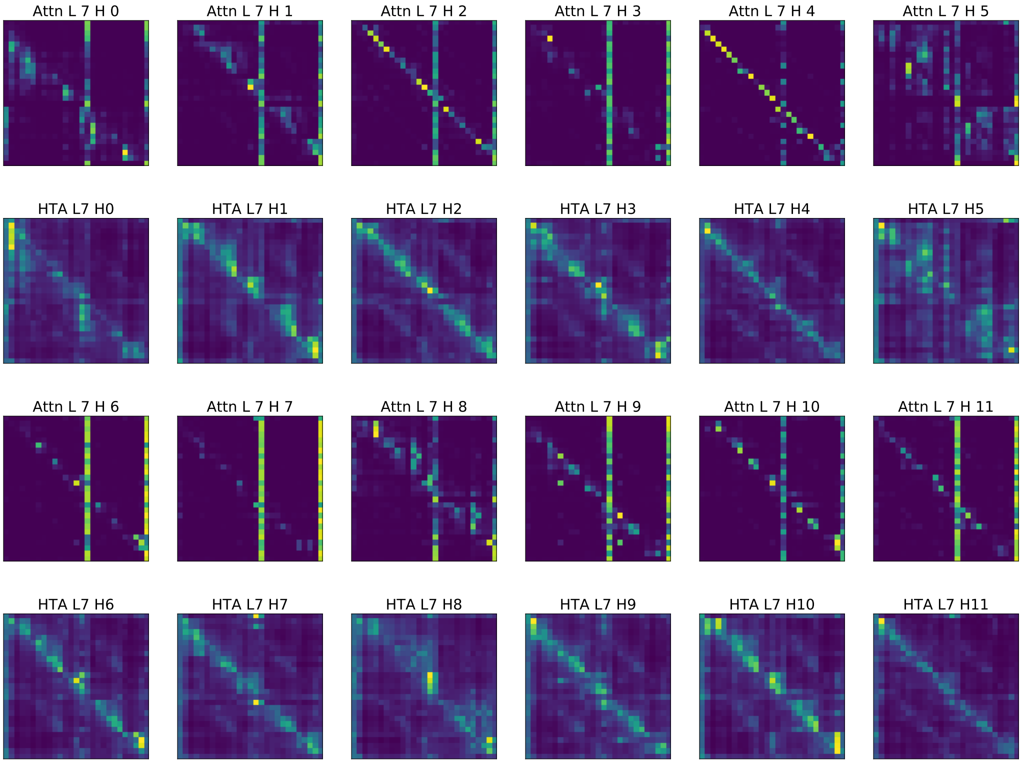


Figure 19: Layer 8: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

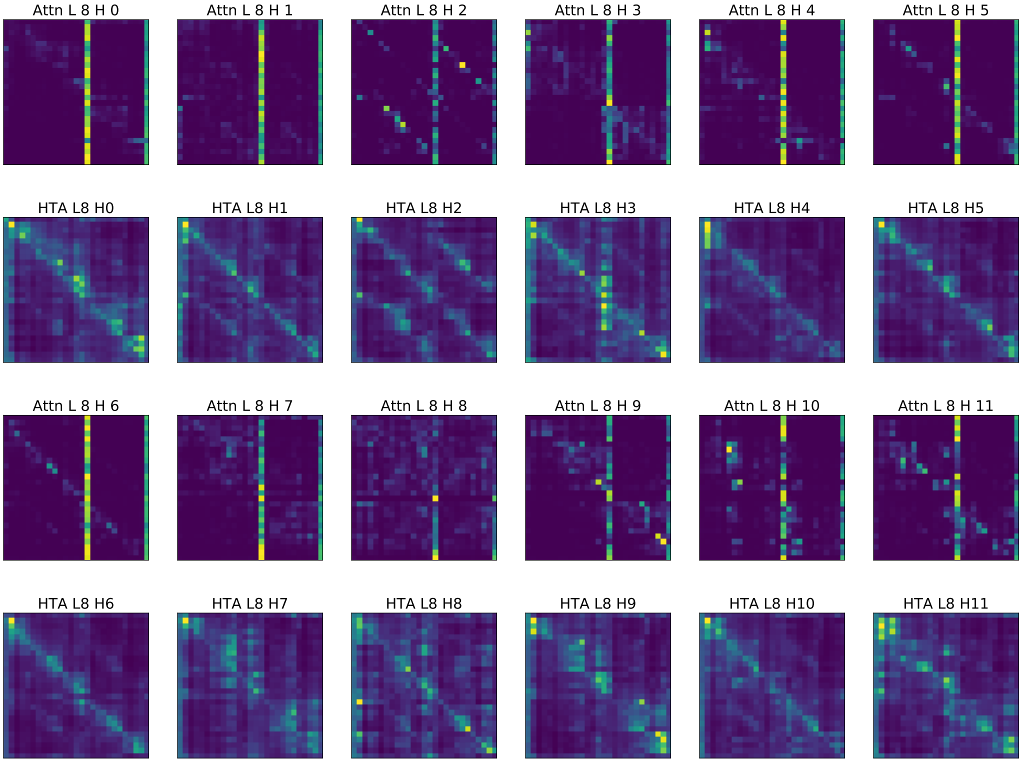


Figure 20: Layer 9: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

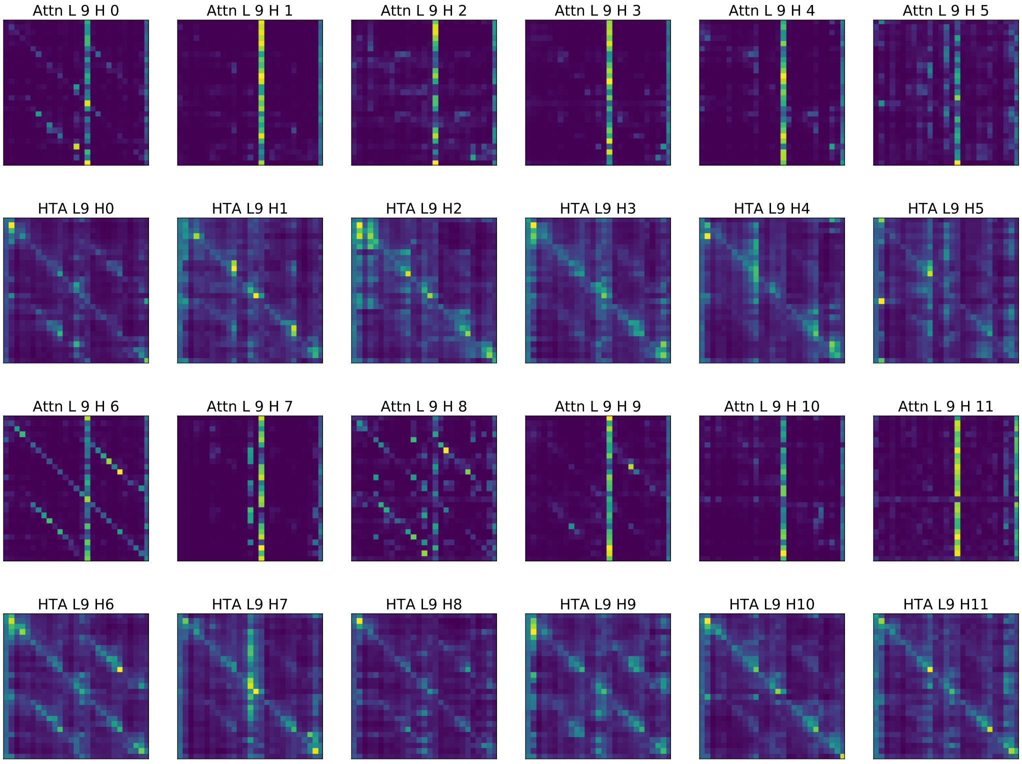


Figure 21: Layer 10: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

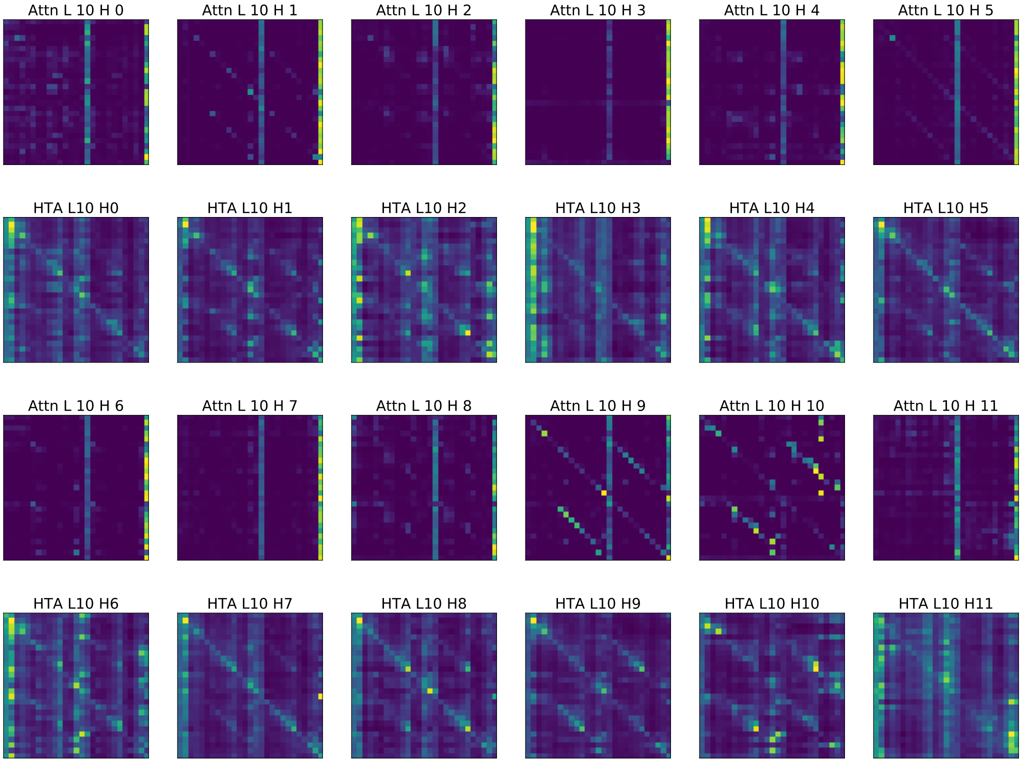


Figure 22: Layer 11: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

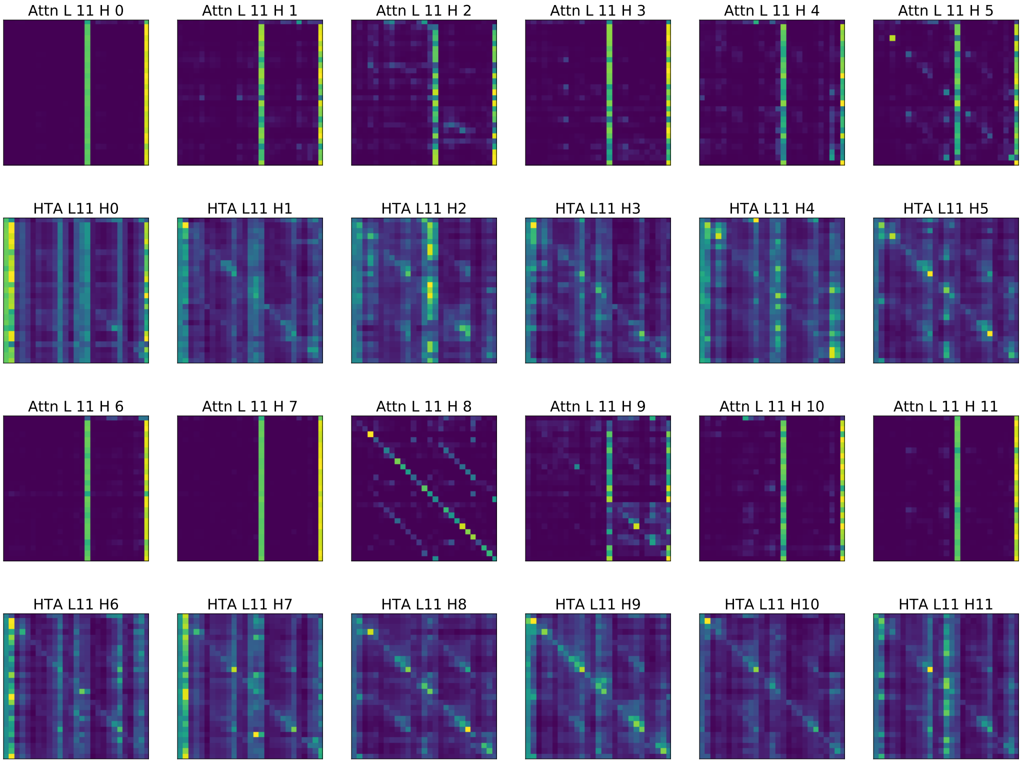


Figure 23: Layer 12: rows 1 and 3 represent attention maps, rows 2 and 4 show the corresponding contribution maps.

Effects of Pre- and Post-Processing on type-based Embeddings in Lexical Semantic Change Detection

Jens Kaiser[*], **Sinan Kurtyigit**[*], **Serge Kotchourko**[*], **Dominik Schlechtweg**
Institute for Natural Language Processing, University of Stuttgart
{jens.kaiser,sinan.kurtyigit,serge.kotchourko,schlecdk}@ims.uni-stuttgart.de

Abstract

Lexical semantic change detection is a new and innovative research field. The optimal fine-tuning of models including pre- and post-processing is largely unclear. We optimize existing models by (i) pre-training on large corpora and refining on diachronic target corpora tackling the notorious small data problem, and (ii) applying post-processing transformations that have been shown to improve performance on synchronic tasks. Our results provide a guide for the application and optimization of lexical semantic change detection models across various learning scenarios.

1 Introduction

In recent years Lexical Semantic Change Detection (LSCD), i.e. the detection of word meaning change over time, has seen considerable developments (Tahmasebi et al., 2018; Kutuzov et al., 2018; Hengchen et al., 2021). The recent publication of multi-lingual human-annotated evaluation data from SemEval-2020 Task 1 (Schlechtweg et al., 2020) makes it now possible to compare LSCD models in a variety of scenarios. The task shows a clear dominance of type-based embeddings, although these are strongly influenced by the size of training corpora. In order to mitigate this problem we propose pre-training models on large corpora and refine them on diachronic target corpora. We further improve the obtained embeddings with several post-processing transformations which have been shown to have positive effects on performance in semantic similarity and analogy tasks (Mu et al., 2017; Artetxe et al., 2018b; Raunak et al., 2019) as well as term extraction (Hätty et al., 2020). Extensive experiments are performed on the German and English LSCD datasets from SemEval-2020 Task 1. According to our findings, pre-training is advisable when the target corpora are small and should be done using diachronic data. We further show that pre-training on large corpora strongly interacts with vector dimensionality and propose a simple solution to avoid drastic performance drops. Post-processing often yields further improvements. However, it is hard to find a reliable parameter that performs well across the board. Our experiments suggest that it is possible to use simple pre- and post-processing techniques to improve the state-of-the-art in LSCD.

2 Related Work

As evident in Schlechtweg et al. (2020) the field of LSCD is currently dominated by Vector Space Models (VSMs), which can be divided into type-based (static) (Turney and Pantel, 2010) and token-based (contextualized) (Schütze, 1998) models. Prominent type-based models include low-dimensional embeddings such as Global Vectors (GloVe, Pennington et al., 2014) and Skip-Gram with Negative Sampling (SGNS, Mikolov et al., 2013a,b). However, as these models come with the deficiency that they aggregate all senses of a word into a single representation, token-based embeddings have been proposed (Peters et al., 2018; Devlin et al., 2019). According to Hu et al. (2019) these models can ideally capture complex characteristics of word use, and how they vary across linguistic contexts. The results of SemEval-2020 Task 1 (Schlechtweg et al., 2020), however, show that contrary to this, the token-based embedding models (Beck, 2020; Kutuzov and Giulianelli, 2020) are heavily outperformed by the type-based ones (Pražák et al., 2020; Asgari et al., 2020). The SGNS model was not only widely used, but also performed best among the participants in the task. This result was recently reproduced in the DIACR-

---

[*] Authors contributed equally, and their ordering was determined randomly.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 125–137
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Ita shared task (Basile et al., 2020; Laicher et al., 2020; Kaiser et al., 2020b). Its fast implementation and combination possibilities with different alignment types further solidify SGNS as the standard in LSCD (Schlechtweg et al., 2020, 2019a; Shoemark et al., 2019; Kutuzov et al., 2020). Hence, the embeddings used in this work are SGNS-based.

Further increases in performance of type-based VSMs can be achieved by various post-processing transformations. This has been shown for semantic similarity and analogy tasks (Mu et al., 2017; Artetxe et al., 2018b; Raunak et al., 2019) as well as term extraction (Hätty et al., 2020). It is still an open question whether these transformations improve performance in the special setting of LSCD where we typically have several corpora and vector spaces which have to be transformed simultaneously (Schlechtweg et al., 2020). An indication is given by Schlechtweg et al. (2019a) showing that for a simple LSCD model mean centering leads to consistent performance improvements on two German data sets. Whether this result is reproducible on further data sets, more complex models and further post-processing techniques has not been determined yet.

Post-processing methods operate on information already contained in a VSM, rather than adding additional information. Further semantic information can be introduced by pre-training vectors on a larger unspecific collection of text (Kutuzov and Kuzmenko, 2016) or by training a seperate matrix on such text and concatenating the two VSMs (Limsopatham and Collier, 2016). This is especially helpful for cases where only smaller specialized corpora are given. Combining the information from two models is also found in Kim et al. (2014), here it is used for alignment proposes. We operate similarly to Kim et al. but with the motivation of Limsopatham and Collier and Kutuzov and Kuzmenko, as we aim to enrich a VSM prior to the training process.

3 Data and Tasks

We train SGNS-based VSMs on various corpora and use a word similarity task and an LSCD task for evaluation. The two tasks share a common aspect: the vector representations of two words need to be compared with some metric (e.g. cosine similarity), and word pairs need to be ranked according to that metric. In the word similarity task, we have the vectors of two different words in the same vector space (w_i, w_j), while for LSCD we have the vectors of the same word but from different vector spaces representing different time periods (w_i^{t1}, w_i^{t2}).

Modern Data. We use two large modern English and German corpora, PUKWAC (Baroni et al., 2009) and SDEWAC (Faaß and Eckart, 2013) to validate the post-processing methods on the word similarity task and to create pre-trained embeddings for the LSCD task. PUKWAC and SDEWAC are web-crawled corpora from the *.uk* and *.de* domain respectively. Resulting in fairly large corpora, 2B tokens and 750M tokens (see Table 1). We evaluate vector representations created on the two corpora on a standard dataset of human similarity judgments, WordSim353 (Finkelstein et al., 2002), by measuring Spearman's rank correlation coefficient of the cosine similarity of vectors for target word pairs with human judgments.

Diachronic Data. We utilize the English and German datasets provided by SemEval-2020 Task 1 Subtask 2 (Schlechtweg et al., 2020). Each dataset contains two *target corpora* from different time periods, t_1 and t_2, as well as a list of target words. The corpora originate mostly from newspaper articles and books. Their biggest difference to PUKWAC and SDEWAC is their approximately 10 to 100 times smaller size, according to token counts (see to Table 1). The task is to rank the list of target words according to their word sense divergence, gradually from 0 (no change) to 1 (total change). The rank predictions are compared against gold data which is based on human judgments. Once again Spearman's rank correlation coefficient is used to measure performance on the task.

4 Models

Following the popular approach taken for type-based vector space models in LSCD, we combine three sub-systems: (i) creating semantic word representations, (ii) aligning them across corpora, and (iii) measuring differences between the aligned representations (Schlechtweg et al., 2019a; Dubossarsky et al., 2019; Shoemark et al., 2019). Alignment is needed as columns from different vector spaces may not correspond to the same coordinate axes, due to the stochastic nature of many low-dimensional word representations (Hamilton et al., 2016). Additionally, we aim to refine subsystem (i) by adding pre-trained semantic word representations and using post-processing methods

	DIACHRON				MODERN	
	GER$_{t1}$	GER$_{t2}$	ENG$_{t1}$	ENG$_{t2}$	SdeWaC	PukWaC
source	DTA	BZ+ND	CCOHA	CCOHA	web	web
time period	1800 – 1899	1946 – 1990	1810 – 1860	1960 – 2010	$\sim$2005 – 2005	$\sim$2005 – 2005
# of tokens	66.9M	67.2M	6.48M	6.62M	750M	1.92B
# of types	51.1K	59.1K	25.9K	37.5K	44.6K	51.9K
min word freq.	39	39	4	4	450	750

Table 1: Corpus statistics. GER$_{t1}$ and GER$_{t2}$ are sampled from DTA (Deutsches Textarchiv, 2017), BZ (Berliner Zeitung, 2018) and ND (Neues Deutschland, 2018). DTA contains texts from different genres, BZ and ND are collections of newspaper articles. Clean Corpus of Historical American English (CCOHA) (Davies, 2012; Alatrash et al., 2020) is a genre balanced collection of texts from a wide variety of time periods and the basis for ENG$_{t1}$ and ENG$_{t2}$.

to improve the quality of the created semantic word representations.[1]

We use SGNS (Mikolov et al., 2013a,b) to create type-based word representations in combination with three different alignment methods, Orthogonal Procrustes (OP), Vector initialization (VI), and Word Injection (WI). The three alignment methods combined with SGNS have been proven to be state-of-the-art, even when competing against token-based embeddings (Schlechtweg et al., 2020; Kaiser et al., 2020a; Basile et al., 2020). Cosine Distance (CD) is used to measure differences between word vectors.[2]

4.1 Alignment

Vector initialization (VI). In VI we first train SGNS on one corpus and then use the learned word and context vectors to initialize the model for training on the second corpus (Kim et al., 2014; Kaiser et al., 2020a). The motivation is that the vector of a word with similar contexts across both corpora will not deviate much from its initialized value. On the other hand, vectors of words with different contexts across both corpora, will be updated to accommodate the new semantic properties. Words which only appear in the second corpus are initialized on random vectors.

Orthogonal Procrustes (OP). SGNS is trained on each corpus separately, resulting in word matrices A and B. To align them, we follow Hamilton et al. (2016) and calculate an orthogonally-constrained matrix W^*:

$$W^* = \underset{W \in O(d)}{\arg\min} \|BW - A\|_F .$$

Prior to this alignment step both matrices are length-normalized and mean-centered (Artetxe et al., 2017; Schlechtweg et al., 2019a).

Word Injection (WI). The sentences of both corpora are shuffled into one joint corpus, but all occurrences of target words are substituted by the target word concatenated with a tag indicating the corpus it originated from (Ferrari et al., 2017; Schlechtweg et al., 2019a; Dubossarsky et al., 2019). This leads to the creation of two vectors for each target word in one vector space, while non-target words receive only one vector encoding information from both corpora.

No Alignment (NO). Comparing two vector spaces without aligning them results in poor performance on LSCD (Schlechtweg et al., 2019a). As VI shows, initializing the model with weights from the previous run, results in aligned vector spaces. We expand on this concept by initializing two models on the same pre-trained weights assuming that the resulting vector spaces are aligned to one another. The difference to VI is that instead of initializing model B with the weights from model A, the weights from a third pre-trained model C are used to initialize both models A and B.

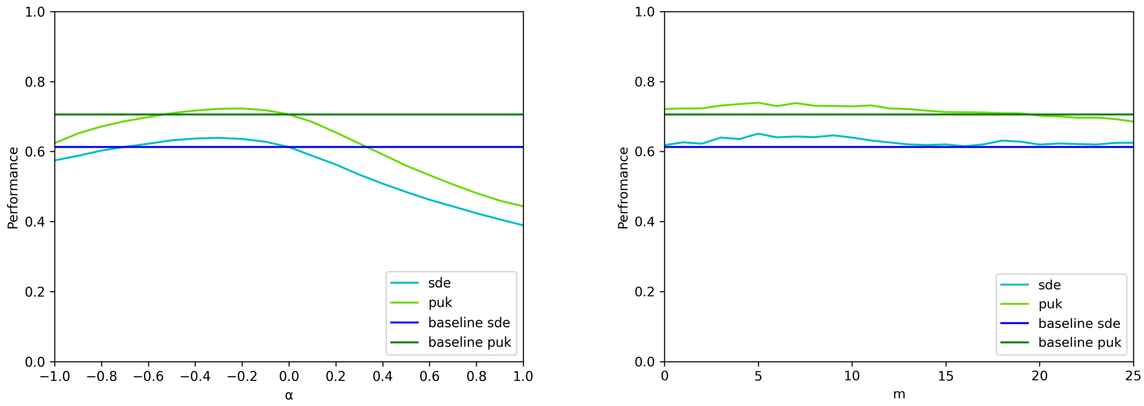
4.2 Pre-training

The corpora used in the context of LSCD are often small, as they are restricted by the length of time periods or availability of historical data. For example the English corpora of SemEval-2020 Task 1 only have 6.6M tokens each, compared to 1.9G of PukWaC. This reduced corpus size limits the amount of semantic information encoded into VSMs trained on the corpus. Pre-training addresses this problem by first training SGNS on a large, possibly external corpus, and then using these vectors

---

[1] Find a comprehensive overview of type-based LSCD models including semantic representations, alignments and measures in Schlechtweg et al. (2019a).

[2] We provide our code at: `https://github.com/Garrafao/LSCDetection`.



(a) (b)

Figure 1: Performance on modern data (wordsim353) **left**: SOT for $\alpha \in [-1, 1]$, **right:** MC+PCR across different amounts of PCs removed. Zero PCs removed indicates only mean centering. Baselines are performances without PP.

to initialize the model for training on the smaller diachronic target corpora. The idea is that the model first learns very broad and general semantic properties followed by the training on the target corpora, where corpus and time specific details are picked up, i.e., a form of refinement. This procedure is applicable to all alignment types.

We use PUKWAC and SDEWAC for pre-training, later referenced as MODERN. However, pre-training on modern corpora is only advisable if the assumption can be made that the meanings of words in the pre-training corpus roughly correspond to the meanings of words in the target corpora. It is unclear to which extent this assumption holds for our data. Hence, we also combine the two target corpora into a bigger corpus, referenced as DIACHRON, which is then used for pre-training.

4.3 Post-processing (PP)

Similarity Order Transformation (SOT). In 2nd order similarity, the similarity of two words is assessed in terms of how similar they are to a third word (Schütze and Pedersen, 1993; Artetxe et al., 2018b; Schlechtweg et al., 2019b). This can analogously be done for higher (3rd, 4th, etc.) orders. According to Artetxe et al. (2018b) these orders capture different aspects of language. Artetxe et al. propose a linear transformation deriving higher or lower orders of similarity from a given matrix X. For this, the product with the transpose matrix is split into its eigendecomposition $X^TX = Q\lambda Q^T$, so that λ is a positive diagonal matrix whose entries are the eigenvalues of X^TX and Q is an orthogonal matrix with their respective eigenvectors as columns. The linear transformation matrix is then defined as $W_\alpha = Q\lambda^\alpha$, where α is the parameter that adjusts the desired similarity order. Applying this to the original embeddings X yields the transformed embeddings $X' = XW^\alpha$.

Mean Centering (MC). The centroid of a matrix is the average vector over all vectors in a matrix: $\vec{c} = \frac{1}{|V|}\sum_i^V \vec{w_i}$. MC refers to subtracting $\vec{c}$ from each $\vec{w_i}$ in the matrix. MC alters all dimensions so that the mean of all columns is zero. Artetxe et al. provide the intuitive motivation for MC that it moves randomly similar vectors further apart and Mu and Viswanath (2018) consider mean centering as an operation making vectors "more isotropic", i.e., more uniformly distributed across the vector space. Mu and Viswanath indicate that isotropy of word vectors is positively correlated to performance.

Principal Component Removal (PCR). Given a n-dimensional matrix X, Principal Component Analysis (PCA, Pearson, 1901) returns n vectors where each vector describes a best fitting line for the data while being orthogonal to the first $n-1$ vectors. Thus, the first PC describes the greatest variance in the first direction, the second PC describes the second greatest variance in the second direction, and the nth PC describes the nth greatest variance in the nth direction. Mu and Viswanath (2018) use PCA to compute the top m PCs from a mean centered word embedding $\bar{M}$: $p_1, ..., p_m = \text{PCA}(\bar{M})$. Subsequently these PCs are used to project each vector $v \in M$ onto the subspace spanned by the PCs. This projection is then subtracted from the original mean centered word vector $\tilde{v}$ by $v' = \tilde{v} - \sum_{i=1}^{m}(p_i^{\intercal} v)p_i$, which results in nullifying the top m PCs in M. This is similar to the approach of Bullinaria and Levy (2012). Mu and Viswanath combine both MC and PCR into one PP transformation (MC+PCR).

As for MC Mu and Viswanath's main motivation for PCR is to make vectors more isotropic. They also demonstrate empirically that the top PCs encode word frequency and offer the removal of this noise from the matrix as an alternative explanation for observed performance improvements.

Stacking. VI and OP alignment result in two matrices, and hence, a proper way for applying PP to both of them is needed. The naïve way of simply post-processing both matrices separately (SEP) may violate the assumption that they are represented in the same space. Therefore, in a second approach, we apply PP to both matrices simultaneously by stacking them vertically beforehand (STA). Preliminary experiments showed that following the naïve way of PP (SEP) led to severe decrease in performance for SOT (but not for MC+PCR). Hence, applying SOT on two matrices separately is followed by an orthogonal post-alignment (SEP+PA).

5 Experiments

For the most part, we chose common model hyperparameter settings in order to keep our results comparable to previous research (Hamilton et al., 2016; Schlechtweg et al., 2019a; Kaiser et al., 2020a). We fine-tune for different alignment methods and datasets by varying dimensionality d, window size w and number of training epochs e.[3]

5.1 Validation

We validate the results reported by Artetxe et al. (2018a) and Mu and Viswanath (2018) on PUKWAC and SDEWAC. The performance peaks for negative α-values around -0.2 as well as the slight performance increase over the baseline for SOT are in line with the findings of Artetxe et al. (see Figure 1a). For MC+PCR we observe the greatest performance improvement when the number of removed PCs is around $m = \frac{d}{100}$ (see Figure 1b). This fits the rule of thumb as stated by Mu and Viswanath.

5.2 LSCD

5.2.1 Pre-training

We tune SGNS models for each alignment method with and without pre-training (baseline), see Table 2. Recall from Section 4.2 that we use the corpora MODERN and DIACHRON for pre-training. Table 2 lists the maximum and mean performances of the baseline and pre-training with different alignment methods, as well as the standard deviation (for a visual representation of the max values see Figure 2). The mean is calculated across different d, e and w, giving the expected performance in a realistic scenario where fine-tuning hyper-parameters is not possible (Schlechtweg et al., 2020; Basile et al., 2020). For German, the baseline max and mean scores could not be significantly improved by pre-training across alignments. For English, pre-training on DIACHRON results in better max and mean scores for OP and WI, with max improvements up to .10. Also, the overall best result is achieved with OP and pre-training on DIACHRON. The usage of MODERN does not improve on the maximum, while reducing the mean. The overall lower performance as well as the observed performance improvements compared to German, may be attributed to the roughly 10 times smaller target corpora. That is, pre-training is helpful on the smaller target corpora.

5.2.2 Post-processing

For every combination of alignment and pre-training method, the matrix with the highest performance across parameters is chosen as the baseline. SOT and MC+PCR are applied individually to these matrices within a wide parameter range (see Appendix B) for both stacking methods (STA and SEP/SEP+PA). Table 3 presents the mean optimal performance gains after PP, which is calculated by extracting the best performance after PP for every matrix, subtracting the baseline values and averaging the values per language. Averaging the respective parameter values yields the mean argmax. Figure 3a and 3d show the highest performances for every baseline matrix after SOT and MC+PCR respectively.

SOT. As we see in Figure 3a, SEP+PA and STA perform similarly. We find small mean performance gains across the board (.013 for GER+STA,

---

[3] For a detailed overview on SGNS parameters see Appendix B.

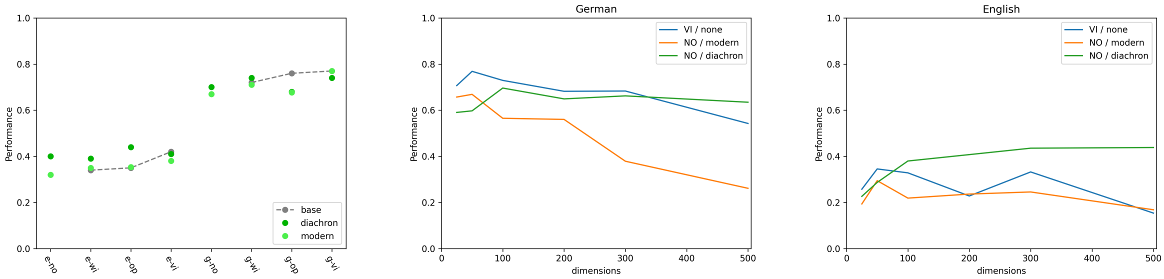


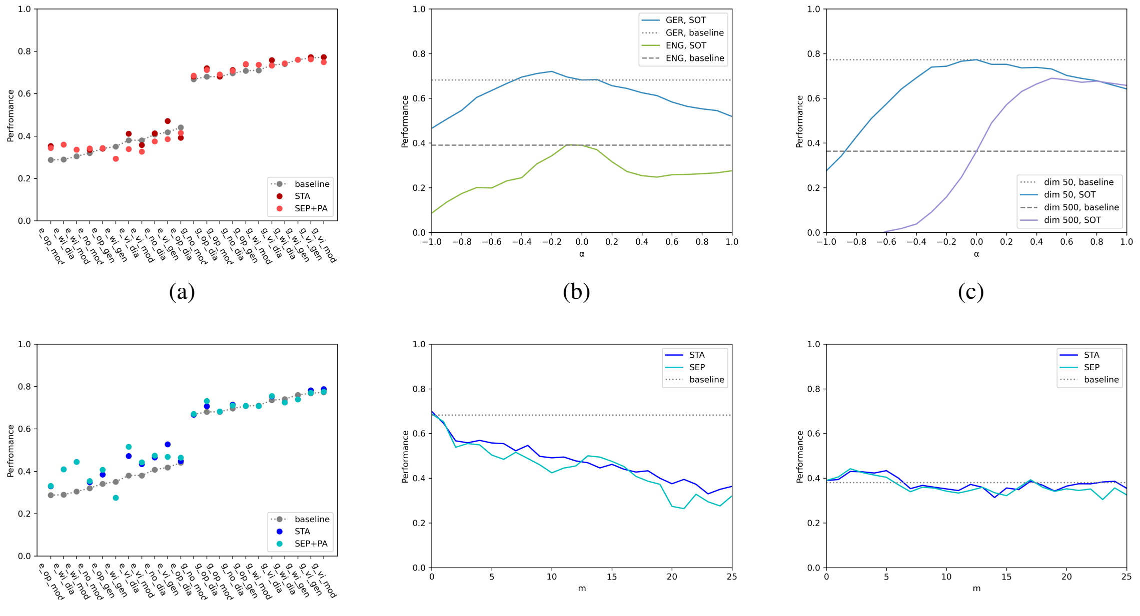
Figure 2: **Left**: max scores from Table 2, **middle** and **right**: Performance (Spearman's rho) of NO alignment method on LSCD task across different dimensionalities and pre-training corpora. VI without pre-training as comparable baseline.

	align.	baseline		DIACHRON		MODERN	
		max	*mean/std*	*max*	*mean/std*	*max*	*mean/std*
GER	VI	**.77**	.72 / .063*	.74	.61 / .067*	**.77**	.70 / .060*
	OP	**.72**	.69 / .022	.68	.59 / .049*	.68	.61 / .051*
	WI	**.76**	.70 / .033	.74	.69 / .037*	.71	.66 / .043*
	NO	-	- / -	.70	.58 / .081*	.67	.60 / .050*
ENG	VI	**.42**	.30 / .067	.41	.28 / .073	.38	.26 / .060
	OP	.34	.28 / .041	**.44**	.31 / .071	.35	.27 / .047
	WI	.35	.28 / .041	**.39**	.29 / .053	.35	.24 / .055
	NO	-	- / -	.40	.34 / .080	.32	.24 / .060

Table 2: max and mean performance on LCSD task (Spearman's rho) for all alignment methods. Note: mean values marked with (*) ignore results utilizing d <100 due to consistent performance drops at higher d.

.008 for GER+SEP+PA, .013 for ENG+STA), except for ENG+SEP+PA where a minuscule decrease (-.005) can be seen. Overall, STA outperforms SEP+PA slightly. We now further examine the effect of SOT+STA on individual matrices. In general, the data can approximately be described as a downward opening parabola (see Figure 3b), with different peaks for both languages and slight differences between alignment methods. Averaging the argmax for α shows us where these peaks are. The calculations yield a mean optimal α of 0 for GER+STA, and -0.2 for ENG+STA. For GER the peak performance always lies in the interval $[-0.2, 0.3]$. This changes to $[-0.4, 0.1]$ for ENG, except for one outlier, where the peak is at -0.8. Moving α away from this parameter range results in severe performance decreases. This behaviour can also be seen on the MODERN corpora (see Figure 1) and is in line with the findings of Artetxe et al. (2018b). In order to predict a high-performing parameter, independent from the underlying matrix, we calculate mean performance gains for fixed parameter values. The values are chosen according to the the above-described peak intervals for the respective languages. However, on average, using a fixed parameter results in slight performance losses, notwithstanding the α-value, and hence, finding a high-performing fixed parameter value was not possible. We observe similar findings for individual alignment methods and varying dimensionality. However, GER+VI alignment represents an interesting exception: With high dimensionality (> 300) base performance drops heavily (Kaiser et al., 2020a), and is then "repaired" by the PP, bringing it close to the baseline of the best performing dimension (see Figure 3c).

MC+PCR. As we see in Figure 3d, MC+PCR yields small improvements over the baselines for German. This is also reflected in the mean gain in Table 3. We find that no single value for m yields consistent improvements. However, we find that for m=0 (only MC) MC+PCR consistently improves the baseline slightly (see Figure 3e), while for higher m the performance decreases consistently. For English we see greater improvements, see Fig-



(a) (b) (c)

(d) (e) (f)

Figure 3: **Top:** SOT (3a, 3c, 3b), **Bottom:** MC+PCR (3d, 3e, 3f). Performance over high-scores (3a, 3d). Representative results after SOT+STA over german and english dataset (3b). Representative plot of "repair" effect after SOT+STA for GER+VI (3c). Representative result after MC+PCR over German and English dataset (3e, 3f). Note for 3a and 3d: where data points overlap only lighter colour visible; dashed line between baseline data points only a visual aid.

	PP + STA/SEP	argmax *mean/std*	gain *mean/std*
GER	SOT + STA	0.0/0.2	.013/.013
	SOT + SEP+PA	0.1/0.3	.008/.015
	MC+PCR + STA	1.2/1.6	.004/.042
	MC+PCR + SEP	0.7/1.1	.004/.043
ENG	SOT + STA	-0.2/0.2	.013/.041
	SOT + SEP+PA	-0.2/0.3	-.005/.043
	MC+PCR + STA	3.0/3.8	.049/.068
	MC+PCR + SEP	6.2/7.1	.058/.077

Table 3: Mean of best-performing parameters and mean performance gain compared to baseline on LSCD task. Parameter range for SOT [-1,1] and MC+PCR [0,25]

ure 3f, 3d and mean gain in Table 3. A range of parameters shows improvements with m=3 yielding the highest (.0175). This can also be seen in Figure 3f where several parameters yield improvements. We conclude that predicting a parameter for likely performance improvement is possible for English, but not for German. However, if this PP should be used, we recommend using a parameter space of $m \in [0, 5]$, as this parameter space is most likely to produce improvements on English, while not harming performance too much on German. This also roughly corresponds to the recommendation of Mu and Viswanath (2018), as they predict that the parameter should be chosen around $\frac{d}{100}$. Furthermore, we suggest using STA, as this does on average show better performance over SEP for the aforementioned parameter space. We see that the effects of SOT as well as MC+PCR are highly dependent on the underlying matrix.

6 Analysis

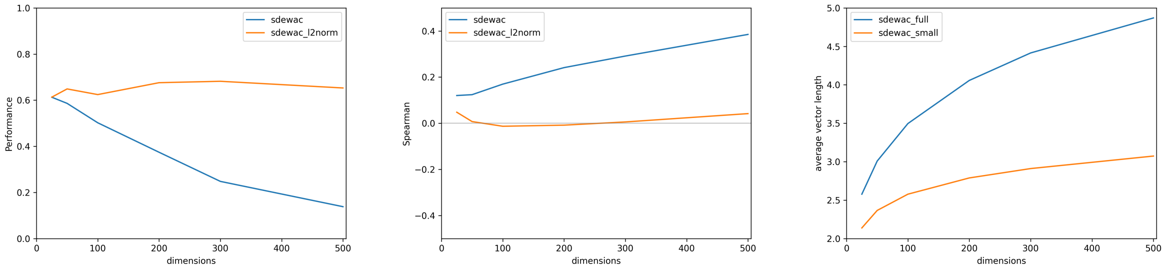
Test Statistics. The effects of pre-training and PP methods on word embeddings are not limited to performance differences in word similarity or LSCD tasks. We use two test statistics to further analyse vector spaces: (i) isotropy (Mu and Viswanath, 2018), i.e., uniformity of vector distribution and (ii) frequency bias (Dubossarsky et al., 2017; Kaiser et al., 2020a), i.e., correlation between cosine distance and frequency.[4]

6.1 Pre-training

On the German dataset it is noticeable that pre-training on DIACHRON often results in slight drop in performance at higher d. This behaviour is more pronounced, consistent and even visible on the En-

---

[4] We compute correlation based on frequency in the second target corpus, results were similar for the first target corpus.



(a) (b) (c)

Figure 4: Test-statistics for result analysis, **left**: Performance after pre-training on different corpora, **middle**: Correlation between CD and frequency, **right**: average vector length of the weights created on different-sized pre-training corpora.

glish dataset when pre-training on MODERN, see Figure 2.[5] Such a drop in performance after initializing on pre-trained vectors has already been observed by Kaiser et al. (2020a). The authors relate the drop to an increased frequency bias and reduce it by increasing e/w. It is noteworthy that the drop is much more pronounced for pre-training on MODERN compared to DIACHRON. This can be attributed to a difference in word vector lengths of the SGNS model used for initialization. We make the following observation: average word vector length increases with the amount of training word pairs. The difference more training data makes is amplified at higher d, see Figure 4c. By length-normalizing the word vectors between the initialization and training step, the drop in performance can be completely circumvented. Additionally, the frequency bias is reduced to 0, see Figure 4b.

For English, we expected a higher performance gain from pre-training when using MODERN because of the small data size. However, we observe no improvements over the baseline. Using length-normalized word vectors for initialization does result in slightly improved max and mean values for MODERN but these are still lower than max and mean values of DIACHRON.

6.2 SOT

SOT has a clear effect on isotropy, which has not been described in previous research. Isotropy shows the same behaviour across both languages and all models, and is best described as a vertically mirrored S-curve (see Figure 5a). Decreasing α increases isotropy close to 1, while increasing α decreases isotropy close to 0. The average correlation (Pearson) between α and isotropy over all matrices is -.89 for both languages. However, the performance correlates only slightly with isotropy (-.25, .35). Moreover, α correlates only weakly with frequency bias (.19, -.12, however with high variance). In order to explain the above-described "repair" effect we take a closer look at the three GER+VI models. Applying SOT brings large performance increases, as stated in Section 5.2.2. For all three models a considerably higher baseline frequency bias for d=500 is visible. SOT strongly reduces this bias for MODERN, and results in a huge performance gain (see Figure 5b).

6.3 MC+PCR

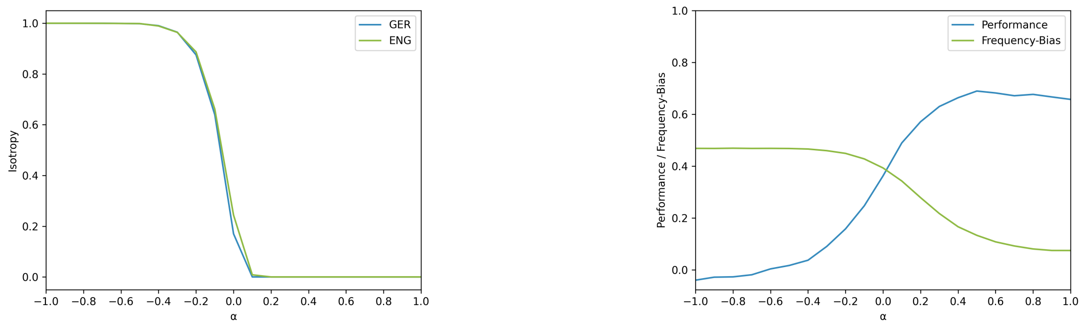
As Mu and Viswanath (2018)'s main motivation behind MC+PCR is to increase isotropy of a vector space as well as removal of word frequency noise through PCR, we examine how isotropy and frequency bias develop with m. While PCR has the predicted effect on frequency bias (GER: -.94, ENG: -0.6), PCR does in fact not increase isotropy, contrary to Mu and Viswanath's motivation of "rounding towards isotropy", but has a consistent reducing effect (GER -.75, ENG: -.7). Thus, we believe that rounding towards isotropy is not suitable for explaining performance. Furthermore, we observe that MC not only exhibits effects on isotropy, but also acts on frequency bias, thus Mu and Viswanath's PCR motivation can be extended to MC.

7 Conclusion

We tested the effects of pre-training and post-processing on a variety of LSCD models. We performed extensive experiments on a German and an

---

[5] Although not depicted, the other alignment techniques in combination with pre-training show very similar behaviour to NO.



(a)

(b)

Figure 5: Representative plot for the isotropy after SOT+STA (5a). Performance and frequency bias after SOT+STA for GER+VI+BIG (5b).

English LSCD dataset. According to our findings, pre-training is advisable when the target corpora are small and should be done using diachronic data. The size of the pre-training corpus is crucial, as a large number of training pairs leads to performance drops, which are probably caused by their effect on vector length. Length-normalization may be used on pre-trained vectors to counteract this effect.

Further performance improvements may be reached by post-processing. While SOT+STA yielded moderate improvements for both languages, MC+PCR showed larger improvements, but only on English. However, for neither we were able to find a reliable parameter that performed well across the board. Instead, we found that a well-performing parameter value is highly dependent on the underlying matrix. Both post-processing methods affect isotropy and frequency bias.

The methods we tested are particularly helpful when tuning data is available, as performance can be optimized and becomes more predictable. Hence, we recommend to obtain a small annotated sample of target words for the target corpora and to tune pre-training, model and post-processing parameters on the sample before performing predictions for semantic changes on unseen data. With the recent upsurge of digitized historical corpora and diachronic semantic annotation efforts (Tahmasebi and Risse, 2017; Schlechtweg et al., 2018, 2020; Basile et al., 2020; Rodina and Kutuzov, 2020) this may often be a likely and feasible scenario.

Acknowledgments

Dominik Schlechtweg was supported by the Konrad Adenauer Foundation and the CRETA center funded by the German Ministry for Education and Research (BMBF) during the conduct of this study. We thank the reviewers for their insightful feedback.

References

Reem Alatrash, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. 2020. CCOHA: Clean Corpus of Historical American English. In *Proceedings of the 12th International Conference on Language Resources and Evaluation*, Marseille, France. European Language Resources Association.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, Austin, Texas. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 451–462. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018a. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019.

Mikel Artetxe, Gorka Labaka, Iñigo Lopez-Gazpio, and Eneko Agirre. 2018b. Uncovering divergent linguistic information in word embeddings with lessons for intrinsic and extrinsic evaluation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 282–291, Brussels, Belgium. Association for Computational Linguistics.

Ehsaneddin Asgari, Christoph Ringlstetter, and Hinrich Schütze. 2020. EmbLexChange at SemEval-2020 Task 1: Unsupervised Embedding-based Detection of Lexical Semantic Changes. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43:209–226.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020. Overview of the EVALITA 2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Christin Beck. 2020. DiaSense at SemEval-2020 Task 1: Modeling sense change via pre-trained BERT embeddings. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Berliner Zeitung. Diachronic newspaper corpus published by Staatsbibliothek zu Berlin [online]. 2018.

John Bullinaria and Joseph Levy. 2012. Extracting semantic representations from word co-occurrence statistics: Stop-lists, stemming, and svd. *Behavior research methods*, 44:890–907.

Mark Davies. 2012. Expanding Horizons in Historical Linguistics with the 400-Million Word Corpus of Historical American English. *Corpora*, 7(2):121–157.

Deutsches Textarchiv. Grundlage für ein Referenzkorpus der neuhochdeutschen Sprache. Herausgegeben von der Berlin-Brandenburgischen Akademie der Wissenschaften [online]. 2017.

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

Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. 2019. Time-Out: Temporal referencing for robust modeling of lexical semantic change. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 457–470, Florence, Italy. Association for Computational Linguistics.

Haim Dubossarsky, Daphna Weinshall, and Eitan Grossman. 2017. Outta control: Laws of semantic change and inherent biases in word representation models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1147–1156, Copenhagen, Denmark.

Gertrud Faaß and Kerstin Eckart. 2013. SdeWaC – A corpus of parsable sentences from the web. In Iryna Gurevych, Chris Biemann, and Torsten Zesch, editors, *Language Processing and Knowledge in the Web*, volume 8105 of *Lecture Notes in Computer Science*, pages 61–68. Springer Berlin Heidelberg.

Alessio Ferrari, Beatrice Donati, and Stefania Gnesi. 2017. Detecting domain-specific ambiguities: An NLP approach based on wikipedia crawling and word embeddings. In *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference Workshops*, pages 393–399.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20:116–131.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Anna Hätty, Dominik Schlechtweg, Michael Dorna, and Sabine Schulte im Walde. 2020. Predicting degrees of technicality in automatic terminology extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Simon Hengchen, Nina Tahmasebi, Dominik Schlechtweg, and Haim Dubossarsky. 2021. Challenges for computational lexical semantic change. In Nina Tahmasebi, Lars Borin, Adam Jatowt, Yang Xu, and Simon Hengchen, editors, *Computational Approaches to Semantic Change*, volume Language Variation, chapter 11. Language Science Press, Berlin.

Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic sense modeling with deep contextualized word embeddings: An ecological view. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3899–3908, Florence, Italy. Association for Computational Linguistics.

Jens Kaiser, Dominik Schlechtweg, Sean Papay, and Sabine Schulte im Walde. 2020a. IMS at SemEval-2020 Task 1: How low can you go? Dimensionality in Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Jens Kaiser, Dominik Schlechtweg, and Sabine" Schulte im Walde. 2020b. OP-IMS @ DIACR-Ita: Back to the Roots: SGNS+OP+CD still rocks Semantic Change Detection. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal analysis of language through neural language models. In *LTCSS@ACL*, pages 61–65. Association for Computational Linguistics.

Andrey Kutuzov, Vadim Fomin, Vladislav Mikhailov, and Julia Rodina. 2020. Shiftry: Web service for diachronic analysis of russian news. In *Proceedings of the International Conference "Dialog"*.

Andrey Kutuzov and Mario Giulianelli. 2020. UiO-UvA at SemEval-2020 Task 1: Contextualised Embeddings for Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Andrey Kutuzov and Elizaveta Kuzmenko. 2016. Cross-lingual trends detection for named entities in news texts with dynamic neural embedding models. In *NewsIR@ECIR*, pages 27–32.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Severin Laicher, Gioia Baldissin, Enrique Castaneda, Dominik Schlechtweg, and Sabine Schulte im Walde. 2020. CL-IMS @ DIACR-Ita: Volente o Nolente: BERT does not outperform SGNS on Semantic Change Detection. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Nut Limsopatham and Nigel Collier. 2016. Modelling the combination of generic and target domain embeddings in a convolutional neural network for sentence classification. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 136–140, Berlin, Germany. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, Lake Tahoe, Nevada, USA.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. 2017. Representing sentences as low-rank subspaces. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 629–634, Vancouver, Canada. Association for Computational Linguistics.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

Neues Deutschland. Diachronic newspaper corpus published by Staatsbibliothek zu Berlin [online]. 2018.

Karl Pearson. 1901. *On Lines and Planes of Closest Fit to Systems of Points in Space*. University College.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Ondřej Pražák, Pavel Přibákň, Stephen Taylor, and Jakub Sido. 2020. UWB at SemEval-2020 Task 1: Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Vikas Raunak, Vivek Gupta, and Florian Metze. 2019. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Julia Rodina and Andrey Kutuzov. 2020. Rusemshift: a dataset of historical lexical semantic change in russian. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING 2020)*. Association for Computational Linguistics.

Dominik Schlechtweg, Anna Hätty, Marco del Tredici, and Sabine Schulte im Walde. 2019a. A Wind of Change: Detecting and evaluating lexical semantic change across times and domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy. Association for Computational Linguistics.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Dominik Schlechtweg, Cennet Oguz, and Sabine Schulte im Walde. 2019b. Second-order co-occurrence sensitivity of skip-gram with negative sampling. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 24–30, Florence, Italy. Association for Computational Linguistics.

Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. 2018. Diachronic Usage Relatedness (DURel): A framework for the annotation of lexical semantic change. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 169–174, New Orleans, Louisiana, USA.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Hinrich Schütze and Jan Pedersen. 1993. A vector model for syntagmatic and paradigmatic relatedness. In *Proc. of the 9th Annual Conference of the UW Centre for the New OED and Text Research*, pages 104–113, Oxford, England.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to Glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to diachronic conceptual change. *arXiv:1811.06278*.

Nina Tahmasebi and Thomas Risse. 2017. Finding individual word sense changes and their delay in appearance. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 741–749, Varna, Bulgaria.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Int. Res.*, 37(1):141–188.

A Corpus details

The corpora are lemmatized and contain no punctuation, further pre-processing on the corpora by us is limited to removing low-frequency words. All words with a frequency below the value listed in row *min word freq.* in Table 1 are removed from the corpora. This is done to reduce noise and unwanted artifacts.

B Parameter settings

SGNS. We use common hyper-parameter settings: initial learning rate of 0.025, number of negative samples k=5 and no sub-sampling. Vector dimensionality d, window size w and number of training epochs e are varied in order to fine-tune model and methods. This is important as alignment methods like VI are highly dependent on the choice of e and d (Kaiser et al., 2020a). The following values are used: $w \in \{5, 10\}$, $e \in \{5, 10, 20, 30\}$, $d \in \{25, 50, 100, 200, 300, 500\}$. Due to the immense amount of possible parameter combinations we only ran each setting once.

PP was performed on the high-scores of each language, where we differentiate between different combinations of alignment, pre-training as well as if the matrices were STA or SEP post-processed.

SOT. As stated in Section 4.3, SEP is used in combination with post-alignment. We apply SOT with α values ranging from -1 to 1 in 0.1 increments on every baseline matrix with $d \in \{25, 50, 100, 200, 300, 500\}$.

MC+PCR. MC+PCR is performed using a parameter space of [0, 25] in order to examine the performance development over a growing number of PCs removed. It is important to note that using the parameter 0 results in only applying MC.

The Gutenberg Dialogue Dataset

Richard Csaky
Budapest University of Technology
University of Oxford
`richard.csaky@psych.ox.ac.uk`

Gábor Recski
TU Wien
`gabor.recski@tuwien.ac.at`

Abstract

Large datasets are essential for neural modeling of many NLP tasks. Current publicly available open-domain dialogue datasets offer a trade-off between quality (e.g., DailyDialog (Li et al., 2017b)) and size (e.g., Opensubtitles (Tiedemann, 2012)). We narrow this gap by building a high-quality dataset of 14.8M utterances in English, and smaller datasets in German, Dutch, Spanish, Portuguese, Italian, and Hungarian. We extract and process dialogues from public-domain books made available by Project Gutenberg[1]. We describe our dialogue extraction pipeline, analyze the effects of the various heuristics used, and present an error analysis of extracted dialogues. Finally, we conduct experiments showing that better response quality can be achieved in zero-shot and finetuning settings by training on our data than on the larger but much noisier Opensubtitles dataset. Our open-source pipeline[2] can be extended to further languages with little additional effort. Researchers can also build their versions of existing datasets by adjusting various trade-off parameters.

1 Introduction

Current open-domain dialogue datasets offer trade-offs between quality and size. High-quality datasets are usually too small to represent the multitude of topics required for a conversational agent. Large datasets often lack good turn-segmentation and are generally noisy, models trained on such datasets generate low-quality or generic output. In Section 2 we analyze publicly available dialogue corpora and the trade-offs they offer. To address the need for large, high-quality datasets we build a corpus of 14.8M utterances in English using publicly available books from Project Gutenberg. We also build datasets for German, Dutch, Spanish, Portuguese, Italian, and Hungarian, with utterance counts in the 20k–200k range. We call this dataset ensemble the Gutenberg Dialogue Dataset. We wish to make it explicit that we are not aiming to create a gold dataset. Our goal is to create a dataset which offers a better size-quality trade-off than other dialogue corpora. The Gutenberg dataset is both larger than DailyDialog (Li et al., 2017b) and has better quality than Opensubtitles (Tiedemann, 2012), and we think it benefits researchers by filling a need in the landscape of dialogue corpora. The Gutenberg Dialogue Dataset and the code used to build it can be accessed through this repository: `https://github.com/ricsinaruto/gutenberg-dialog`. The repository also contains all trained models presented in this paper and all data and training scripts used to produce the results. We also built a web demo interface for interacting with the trained models[3].

In Section 3 we offer a detailed quantitative analysis of our heuristics to better understand their effects on data quality. Section 4 presents our error analysis of the English dataset both at the utterance and dialogue level. Using our MIT licensed pipeline, researchers can easily build various dataset versions by adjusting a small number of parameters that control multiple dimensions of the size-quality trade-off. In Section 5 we evaluate our dataset in a generative multi-turn and single-turn setting using the GPT-2 (Radford et al., 2019) and Transformer (Vaswani et al., 2017) architectures, respectively. For each of the 7 languages, we compare models trained on Gutenberg and Opensubtitles. For English, we also compare zero-shot and finetuning performance of Gutenberg and Opensubtitles on two smaller datasets. Potential improvements and future work is discussed in Section 6.

---

[1] `https://www.gutenberg.org/`

[2] `https://github.com/ricsinaruto/gutenberg-dialog`

[3] `https://ricsinaruto.github.io/chatbot.html`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 138–159
April 19 - 23, 2021.

Extension to additional languages is ongoing, we welcome all contributions from the community: our modular code requires only a limited amount of language-specific effort for each new language.

2 Background

Open-domain dialogue datasets vary in size, quality, and source, as demonstrated in Table 1. Generally, smaller datasets are constructed using controlled crowdsourcing environments, making their quality higher (e.g., PersonaChat (Zhang et al., 2018)). Crowdsourcing platforms like Amazon Mechanical Turk[4] are used to hire and instruct workers to carry out free-form conversations. Larger datasets can be built by automatic processing of dialogue-like text sources, such as Opensubtitles and Reddit[5] (Henderson et al., 2019)). Opensubtitles contains movie subtitles in multiple languages and Reddit is a discussion forum with millions of daily comments on various topics. Automatic extraction offers less quality control, and the data source heavily influences the genre of conversations. In Reddit data, everyday chit-chat is less common, comments in the same thread all discuss the same post. Two-party dialogues are rare as threads are almost always multi-speaker. Twitter[6] conversations have similar problems and they are also constrained by a character limit. Extracting conversations from Twitter and Reddit is straightforward as speaker segmentation is included and the thread chain can be used as dialogue history.

Books, especially fiction, have so far seen little use as a source of dialogue data. In DailyDialog (Li et al., 2017b), 90 000 high-quality utterances are extracted from online resources for English language learners, extraction steps are not detailed. The quality of these dialogues and the lack of a large book-based dataset motivates our work. Dialogues extracted from books, like movie subtitles, lack context, but their usefulness is evidenced by the Cornell Corpus (Danescu-Niculescu-Mizil and Lee, 2011) and DailyDialog. As argued by Danescu-Niculescu-Mizil and Lee (2011) and Fainberg et al. (2018), artificial dialogues in movies and books generally resemble natural conversations. Such dialogues are also called written dialogues as opposed to spoken corpora like the Switchboard corpus (Godfrey et al., 1992). Though our corpus contains written dialogues we also perform evaluation on Persona-Chat, which can be considered as a spoken dialogue corpus, and show Gutenberg's effectiveness in this setting as well.

Unfortunately, the Cornell Corpus is relatively small, while the Opensubtitles corpus suffers from the fact that the original dataset lacks both dialogue and turn segmentation: subtitle lines are treated as turns and dialogue history consists of the previous n lines, with little to no additional post-processing used to extract dialogues instead of using the raw data (Henderson et al. (2019) removes the shortest and longest utterances to improve quality). These issues lead to trained models outputting generic responses, e.g., to the input "yes i believe there are green teas black teas and scented teas. any others?" a model trained on Opensubtitles outputs "sure.". In addition, the types and ratio of errors in these datasets have not been explicitly analyzed. For the Gutenberg dataset, we build a multi-step extraction pipeline and analyze both the performance of each heuristic and the ratio of each error type in a sample of the final corpus. Unfortunately, most of the tools developed here are specific to the book domain, and use textual patterns which are not available in Opensubtitles. In order to increase the quality of Opensubtitles, subtitle-specific methods need to be developed, like taking into account the elapsed time between two subtitle lines.

The size of our corpus facilitates effective training of large Transformer-based models (Radford et al., 2019; Yang et al., 2019). Recently, pre-training and finetuning large language models on specific tasks (including dialogue modeling) has gained popularity (Wolf et al., 2018; Devlin et al., 2019). Transformer-based models and specifically GPT-2 have gained state-of-the-art status in the dialogue domain (Adiwardana et al., 2020; Roller et al., 2020; Zhang et al., 2019; Wolf et al., 2018). Through these models the community has gradually shifted from single-turn to multi-turn scenarios. Since we wish to demonstrate our dataset's quality on the dialogue-level, we conduct experiments primarily with GPT-2. We report some single-turn trainings using Transformer for comparison. We show Gutenberg's effectiveness for multi-turn pre-training in Section 5, comparing it to Opensubtitles pre-training, which is popular in the literature (Csaky and Recski, 2017; Krause et al., 2017; Xing and Fernández, 2018).

---

[4] `https://www.mturk.com/`

[5] `https://www.reddit.com/`

[6] `https://twitter.com/`

Dataset	Size	Source	Quality
DailyDialog (Li et al., 2017b)	90k	ESL websites	auto-extracted
Wizard-of-Wikipedia (Dinan et al., 2019)	100k	crowdsourcing	human-written
Document-grounded (Zhou et al., 2018)	100k	crowdsourcing	human-written
Persona-Chat (Zhang et al., 2018)	150k	crowdsourcing	human-written
Self-dialogue (Fainberg et al., 2018)	150k	crowdsourcing	human-written
Cornell Movie Corpus (Danescu-Niculescu-Mizil and Lee, 2011)	300k	movie scripts	auto-extracted
Self-feeding chatbot (Hancock et al., 2019)	500k	human-bot dialogues	partly human-written
Twitter corpus[7]	5M	Twitter posts/replies	auto-extracted
Opensubtitles (Henderson et al., 2019)	320M	movie subtitles	auto-extracted
Reddit (Henderson et al., 2019)	730M	Reddit threads	auto-extracted

Table 1: Comparison of open-domain dialogue datasets in English. *Size* is the rough number of utterances, *Source* describes where the data comes from, and *Quality* distinguishes between dataset collection techniques.

3 Extraction Pipeline

Most of Project Gutenberg's 60 000 online books are in English (47 300 books; 3 billion words). French, Finnish, and German, the next most common languages, contain 3000, 2000, 1750 books, and 194M, 74M, 82M words, respectively. Dutch, Spanish, Italian, Portuguese, and Chinese are all above 10M words, followed by a long tail of various languages. We used the Gutenberg python package[8] to download books and query their license, language, and author metadata. Further Gutenberg statistics can be found in Appendix A.2. This section describes heuristics and methods used to extract dialogues from books and remove noise. The main challenges are identifying changes between speakers within a dialogue and separating sets of utterances that do not belong to the same dialogue. To separate dialogues, changes in location, time, speaker, etc. would have to be identified directly, but we develop simple heuristics (e.g., distance between utterances) that can extract relatively high-quality conversations at scale. Tunable parameters of our system offer trade-offs between data quality and size. Using our open-source system researchers can build custom datasets that best suit their applications.

Our dialogue extraction pipeline includes three main steps: 1. Conversational and narrative text is separated. 2. Dialogic text is split into separate dialogues. 3. Dialogues are segmented into separate turns (utterances). In most books, conversational text is highlighted; e.g., placed between single/double quotation marks in English or started by an em-dash in Hungarian. Naturally, these delimiters have other uses as well, but such cases are rare (about 5% of utterances, see Section 4). We can only extract dialogues from books which clearly delimit both the start and end of conversations. In some languages/books, the start of an utterance is given, but the end is not, and narrative text can get mixed in (e.g., *Si vous arrivez avant nous, cria Luigi au messager, annoncez à la nourrice que nous vous suivons.* 'If you arrive before us, shouted Luigi to the messenger, tell the nurse that we are following you.'). This is why we could not build a French dataset, and have relatively smaller datasets in Dutch, Italian, Portuguese, and Hungarian. Figure 1 shows a sample dialogue highlighting our heuristics. In the following paragraphs, we offer a parameter-based description of our pipeline.

"Read what I have written," she gasped. "It may be utterly unintelligible."
For answer, Morton folded the sheet and placed it in an envelope.
"Address this, if you please," he said.
She obeyed his request, limply forcing herself to make the effort; and, as the pen once more fell from her fingers, she glanced up at him with a haggard piteousness in her eyes.
"Will you not read what I have written?" she asked again.
"I see no reason why I should," he answered.

Figure 1: A dialogue example. Utterances are in separate paragraphs, sometimes broken up by narrative text.

Pre-filtering After downloading books and separating them by language, all copyrighted works are removed. We also filter books containing unusual,

---

[7] `https://github.com/Marsan-Ma-zz/chat_corpus`

[8] `https://github.com/ageitgey/Gutenberg`

Method	**Parameter**	**Filtered**	**What**
Pre-filter	2 (KL-div)	2090 books (4.42%)	Old books and noise
Delimiter filter	150 delimiters / 10 000 words	20 500 books (43.3%)	Books with no dialogues
Long utterances	100 words	610 000 utterances (3.95%)	Non-conversational utterances
Post-filter	20% rare words	20 478 dialogues (0.8%)	Dialogues containing many rare words

Table 2: The various filtering steps for the English dataset.

mostly older, language: if the KL divergence between a book's word distribution and the total (all books) distribution is above a threshold (2), it is removed. The method is less accurate for short books with less than 20 000 words, these are not filtered. In the English dataset, 2090 books were removed (4.42%). By analyzing 100 filtered and 100 non-filtered books randomly, we found 8 false positives (books that should not have been removed), and 9 false negatives.

Delimiter filter Before dialogue extraction, books with less than 150 delimiters per 10 000 words are removed. We assume that under a certain threshold the probability of delimiters used for non-conversational purposes is increased. We empirically set this ratio by increasing it until the assumption starts failing. Since many books do not contain dialogues, almost half were removed (20 500) in the English pipeline. Sampling 100 filtered and 100 non-filtered books, we found 8 false positives (books that should not have been removed), and 22 false negatives. In a sample of the final dataset, less than 5% of utterances were non-conversational (cf. Section 4).

Dialogue gap If two dialogue segments highlighted by delimiters are far apart, i.e. there are >150 characters between them, they will not be considered part of the same dialogue. This heuristic, the dialogue gap, will always offer a false positive/negative trade-off since the amount of text between dialogues varies considerably. We tuned this trade-off by reasoning that shorter dialogues are less problematic than incoherent dialogues: our setting yields 3.5 times fewer false negatives, as shown in Section 4. Our turn segmentation heuristic will also always treat separate paragraphs as separate utterances. In a sample of the final dataset, this assumption fails for roughly 4% of utterance pairs (cf. Section 4).

Long utterances and rare words During dialogue extraction utterances with more than 100 words are removed to ensure that remaining utterances are truly conversational and to facilitate neural model training (Dai et al., 2019). As all other parameters in the pipeline, this is adjustable to the needs of the user or task. Finally, we remove dialogues with more than 20% rare words (not in the top 100 000), removing noise and facilitating neural model training. Dialogues are split randomly into train (90%), validation (5%), and test (5%) datasets, dialogues from the same book are placed in the same split.

	#U	$\lvert U\rvert$	#D	$\lvert D\rvert$
English	14 773 741	22.17	2 526 877	5.85
German	226 015	24.44	43 440	5.20
Dutch	129 471	24.26	23 541	5.50
Spanish	58 174	18.62	6 912	8.42
Italian	41 388	19.47	6 664	6.21
Hungarian	18 816	14.68	2 826	6.66
Portuguese	16 228	21.40	2 233	7.27

Table 3: Statistics of the final dialogue datasets. Columns are: language, number of utterances, average utterance length, number of dialogues, and average dialogue length.

Languages differ only in the dialogue extraction step. The modular pipeline can be easily extended to new languages by specifying conversational delimiters and a minimal implementation of dialogue and turn segmentation, generally adaptable from English. In practice, adapting the English pipeline to other languages ranged between 0-50 lines of python code. Optionally further analysis might be needed to check the output of the pipeline and refine the extracting process if needed. Delimiters and parameters for other languages were not analyzed as profoundly as for English, leaving room for improvements in future work. We aim to show that good dialogue datasets can be constructed with minimal effort, as a first step towards a high-quality multi-language dataset ensemble. In total, the four filtering steps removed about 12.5% of utterances from the English dataset, detailed in Table 2. Statistics of the final datasets in all 7 languages can be seen in Table 3. The standard deviation of dialogue length in English is 6.09, and there are 87 500 dialogues with at least 20 utterances. The average dialogue length can be linearly adjusted with the dialogue gap parameter.

4 Error Analysis

Utterance-level To assess the single-turn quality of the English dataset we manually analyzed 100 random utterance pairs with book context. 89 pairs did not contain any errors. Remaining utterance pairs contained 1 error type each, out of 2 major and 2 minor types, minor errors occurring in only 1 case each. The extracted text is not conversational in 5 utterance pairs, a consequence of the delimiter threshold and other sources of noise (Figure 2). Utterances of a single speaker were falsely treated as multiple turns in 4 cases, most often because of our assumption that paragraph breaks signal dialogue turns (Figure 3).

> And he was singing, too, as he went on with his task; sometimes–
> "Play on, minstrèl, play on, minstrèl, My lady is mine only girl;"

Figure 2: Non-dialogue text detected as an utterance.

> In his progress he passed the door of the dormitory of his victim—he paused a moment, and listened attentively. Then in a voice of deep anguish he said,—
> “She can sleep—she can sleep—no ghostly vision scares slumber from her eyes—while—”
> He shuddered, and passed a step or two on, then pausing again, he said,—
> “Oh, if she, the young and innocent...”

Figure 3: Two consecutive turns uttered by the same speaker.

Dialogue-level Errors in whole dialogues exhibit a much greater variety. Based on a manual analysis of 50 dialogues in the English dataset we identified 7 error categories (Figure 4). The following numbers are always out of the 50 analyzed dialogues. 16 dialogues contained 0 errors, 21 contained 1 error type, 11 contained 2 types, remaining dialogues containing 3. We detail the number of dialogues affected by each error type below. We note that this does not constitute a proper statistical analysis.

Utterances from the same conversation frequently end up in different dialogues (17 cases, example in Figure 5) because of the dialogue gap threshold. The inverse, a dialogue containing utterances from multiple conversations, occurred in 5 cases (Figure 6). While it is challenging to set this parameter, we consider this to be a reasonable trade-off: shorter dialogues mean less data, but incoherent dialogues with utterances from multiple conversations are bad data. In Section 6 we discuss possible further approaches to segmenting conversational text.

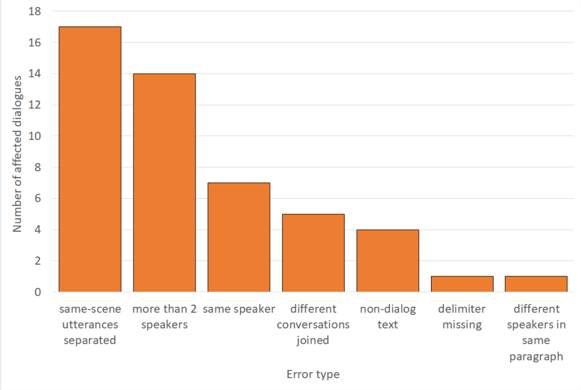


Figure 4: Number of dialogues affected by the various errors. In total 50 dialogues were analyzed. Some dialogues contained multiple types of errors and only 16 dialogues contained 0 errors.

> Richard curbed an impatient rejoinder, and said quietly, "William Durgin had an accomplice."
> Mr. Taggett flushed, as if Richard had read his secret thought. Durgin’s flight, if he really had fled, had suggested a fresh possibility to Mr. Taggett. What if Durgin were merely the pliant instrument of the cleverer man who was now using him as a shield? This reflection was precisely in Mr. Taggett’s line. In absconding Durgin had not only secured his own personal safety, but had exonerated his accomplice. It was a desperate step to take, but it was a skillful one.
> "He had an accomplice?" repeated Mr. Taggett, after a moment. "Who was it?

Figure 5: A single conversation cut up because of the long paragraph between the two utterances.

> “Carry pins, is it?” said Tom. “Ye can carry yer head level, me boy. So at it ye go, an’ ye’ll bate Rory fer me, so ye will.”
> “Well then,” cried Barney, “I will, if you give me first choice, and I’ll take Tom here.”
> “Hooray!” yelled Tom, “I’m wid ye.” So it was agreed, and in a few minutes the sides were chosen, little Ben Fallows falling to Rory as last choice.
> “We’ll give ye Ben,” said Tom, whose nerve was coming back to him. “We don’t want to hog on ye too much.”
> “Never you mind, Ben,” said Rory, as the little Englishman strutted to his place among Rory’s men. “You’ll earn your supper to-day with the best of them.”

Figure 6: First three and last two utterances are not part of the same conversation, but they were merged because of the dialogue gap threshold.

Books often contain dialogues between more than two speakers, our second most frequent source of error (14 dialogues). However, such conversations are still coherent and provide useful data for model training. In contrast, the same speaker uttering at least two consecutive turns breaks coherence in 7 dialogues. Tackling these issues would have to involve speaker identification (cf. Section 6). As in the utterance-level analysis, there were some dialogues (4) in which non-conversational text got mixed in. The remaining errors, *delimiter missing* and *different speakers in same paragraph* occurred in only 1 dialogue out of 50.

5 Experiments

5.1 Evaluation Metrics

Most automatic evaluation methods for dialogue models correlate poorly with human judgment (Liu et al., 2016), and recently proposed metrics that correlate better (Li et al., 2017a; Lowe et al., 2017; Tao et al., 2018) are harder to measure than perplexity or BLEU (Papineni et al., 2002). Human evaluation also has its shortcomings, like high variance, cost, and replication difficulty (Zhang et al., 2018; Tao et al., 2018). There does not seem to be any consensus on the best approach, as some researchers use only automatic metrics (Xing and Fernández, 2018; Xu et al., 2018b), others conduct human evaluation (Krause et al., 2017; Fang et al., 2018), and some use both (Shen et al., 2018; Xu et al., 2018a; Baheti et al., 2018; Ram et al., 2018).

We conduct an extensive automatic evaluation using our DIALOG-EVAL repository[9], which implements 17 metrics used frequently in the literature. These are described in detail by our previous study on metrics (Csáky et al., 2019). The metrics assess individual response quality, dialogue-level evaluation is left for future work[10]. In all tables that follow, metrics are listed in the following order: response length ($|U|$), i.e. average number of words in a response. Per-word and per-utterance unigram (H_w^u, H_u^u) and bigram (H_w^b, H_u^b) entropy, measuring the non-genericness of responses (Serban et al., 2017). Unigram and bigram-level KL divergence (D_{kl}^u, D_{kl}^b) between model and ground truth response sets (Csáky et al., 2019). Embedding metrics *average* (AVG), *extrema* (EXT), and *greedy* (GRE) measuring similarity between response and target embeddings (Liu et al., 2016). Coherence (COH), the cosine similarity between pairs of input and response (Xu et al., 2018b). Distinct-1 and distinct-2 (d1, d2) measuring the ratio of unique unigrams/bigrams in all responses (Li et al., 2016). The 4 BLEU metrics (b1, b2, b3, b4), measuring overlaps between respective n-grams (n=1,2,3,4) of response and target (Shen et al., 2018; Xu et al., 2018b). As discussed in Csáky et al. (2019), these metrics have been selected to provide a diverse evaluation measuring various aspects of response quality. Generally, we should assess response quality jointly as looking at individual metrics can be misleading.

5.2 Trainings

We conduct experiments with Transformer[11] and GPT2[12] models. The Transformer is trained on utterance pairs, and we use the base version of roughly 50M parameters (further training details are given in Appendix A.1). The vocabulary is set to the top 100 000 words for Gutenberg and Opensubtitles trainings, and 32 768 and 16 384, for PersonaChat and DailyDialog, respectively. The Transformer is trained for 21 epochs on Gutenberg and Opensubtitles, because of time and hardware constraints, but the validation loss was still decreasing. Training took about 80 hours on a single RTX 2080 Ti, with batch size set to the memory limit. We used the Adam optimizer (Kingma and Ba, 2014). For generating test outputs greedy decoding is used.

For the GPT2 trainings (117M pretrained version) we set the maximum number of previous utterances to be used as history to 3 (parameter details in Appendix A.1). The huggingface repository leverages GPT2 for dialogue modeling with an additional personality input and a random candidate classification loss (Wolf et al., 2018). We set the personality field to empty and use a single random candidate response from the training set for each example. We use the nucleus sampling implementation in the repository with default parameters to sample outputs (Holtzman et al., 2020). All GPT2 trainings are trained with a batch size of 2 and evaluated at the minimum of the validation loss. The English GPT2 Gutenberg training

---

[9] `https://github.com/ricsinaruto/dialog-eval`

[10] We believe that Gutenberg would perform especially well in dialogue-level metrics, since it contains high-quality extracted dialogues compared to the non-segmented noisy Opensubtitles utterances.

[11] `https://github.com/tensorflow/tensor2tensor`

[12] `https://github.com/huggingface/transfer-learning-conv-ai`

			$\|U\|$	H_w^u	H_w^b	H_u^u	H_u^b	D_{kl}^u	D_{kl}^b	AVG	EXT	GRE	COH	d1	d2	b1	b2	b3	b4
Transformer	**ZS**	G	**7.5**	**6.92**	**11.7**	**52**	**71**	**.90**	**1.72**	**.522**	**.509**	**.577**	**.579**	**.0251**	**.110**	**.098**	**.095**	**.091**	**.083**
		O	4.8	6.65	10.6	32	41	2.00	3.58	.461	.481	.533	.458	.0009	.002	.075	.068	.063	.056
	FT	G	8.7	**7.09**	**11.8**	62	87	**.51**	**1.03**	**.551**	**.535**	**.598**	**.580**	**.0292**	**.147**	**.140**	**.132**	**.126**	**.115**
		O	8.8	6.68	10.2	59	80	2.93	4.15	.486	.477	.560	.482	.0020	.005	.106	.117	.118	.110
		B	**9.9**	**7.11**	11.5	**71**	**94**	.88	1.60	.519	.514	.579	.525	.0132	.063	.127	.128	**.127**	**.117**
GPT2	**ZS**	G	**9.1**	**7.53**	**12.7**	**70**	**98**	**.30**	**.71**	**.538**	**.500**	**.564**	**.559**	.0333	.226	**.104**	**.109**	**.108**	**.101**
		O	5.7	7.19	12.3	42	56	.32	.81	.491	.484	.554	.532	**.0463**	**.249**	.082	.079	.076	.069
	FT	G	9.6	7.61	12.7	75	105	.12	**.33**	**.568**	**.540**	**.596**	**.573**	.0407	.259	**.151**	**.143**	**.139**	**.128**
		O	9.4	7.62	12.6	74	102	.14	.40	.561	.533	.589	**.574**	.0455	.264	.142	.136	.132	.122
		B	**10.0**	**7.76**	**12.8**	**80**	**109**	**.11**	.36	**.567**	.535	.589	**.576**	**.0486**	**.285**	.147	**.143**	**.141**	**.130**
RT			13.6	8.41	14.1	118	179	.03	.17	.496	.461	.523	.493	.0693	.414	.086	.117	.127	.122
GT			13.8	8.38	13.7	117	152	0	0	1	1	1	.572	.0587	.400	1	1	1	1

(a) DailyDialog test set

			$\|U\|$	H_w^u	H_w^b	H_u^u	H_u^b	D_{kl}^u	D_{kl}^b	AVG	EXT	GRE	COH	d1	d2	b1	b2	b3	b4
Transformer	**ZS**	G	**8.3**	**6.99**	**11.9**	**57.7**	**80**	**1.00**	**2.24**	**.493**	.540	**.545**	**.574**	**.0154**	**.077**	.091	.092	.091	.084
		O	6.6	6.70	11.5	45.2	67	2.00	2.85	.471	**.556**	.542	.476	.0004	.001	**.094**	**.098**	**.095**	**.088**
	FT	G	**11.0**	6.48	10.4	68.2	92	**1.28**	**2.15**	**.513**	**.575**	**.571**	**.593**	**.0104**	**.048**	**.165**	**.163**	**.164**	**.155**
		O	10.6	6.37	10.1	68.3	98	2.58	2.66	.431	**.575**	.532	.444	.0011	.002	.148	.151	.154	.146
		B	**11.1**	**6.88**	**11.0**	**76.5**	**110**	**1.28**	2.21	.508	.570	.562	.559	.0047	.018	**.164**	**.163**	**.165**	**.156**
GPT2	**ZS**	G	**9.5**	**7.62**	**13.1**	**72.7**	**101**	.56	1.15	**.510**	**.501**	**.531**	**.551**	.0206	.160	**.092**	**.104**	**.107**	**.101**
		O	6.0	7.35	12.6	44.9	60	**.44**	**1.11**	.478	.491	.519	.537	**.0294**	**.186**	.072	.074	.072	.066
	FT	G	**11.0**	7.45	**11.8**	**82.6**	**116**	.27	.64	**.536**	**.559**	**.558**	**.590**	.0182	.129	**.157**	**.159**	**.162**	**.153**
		O	10.5	7.41	11.6	78.1	108	.32	.71	.531	**.558**	.555	.583	.0205	.129	.153	.154	.155	.146
		B	10.3	**7.50**	**11.8**	77.9	108	**.25**	**.61**	.533	.554	.553	.587	**.0219**	**.136**	.151	.154	.155	.146
RT			11.6	8.51	14.0	98.6	148	.03	.14	.489	.499	.496	.488	.0495	.350	.099	.127	.136	.131
GT			11.5	8.46	13.4	97.3	124	0	0	1	1	1	.559	.0421	.337	1	1	1	1

(b) PersonaChat test set

Table 4: Metrics computed on the test set of DailyDialog and PersonaChat for Transformer and GPT2 trainings. Pre-trained models on Gutenberg (G) and Opensubtitles (O) are compared. B is a Transformer or GPT2 baseline trained only on the small datasets, evaluated at the validation loss minimum. RT refers to randomly selected responses from the DailyDialog or PersonaChat training set, and GT to the ground truth response set. Best results (with a 95% confidence interval) are highlighted separately for the zero-shot (ZS) and finetuned (FT) scenarios.

took about 20 days (7 epochs), on an RTX 2080 Ti, while on Opensubtitles the validation minimum was reached after a single epoch of training (about 2 days). Finetuning on DailyDialog and PersonaChat and trainings on other languages took generally less than 1 day, except the German trainings (2 days).

We evaluate Gutenberg and Opensubtitles pre-trained models in zero-shot and finetuning scenarios on DailyDialog and PersonaChat. The same amount of training data and train/test/dev ratio is used for both Gutenberg and Opensubtitles. Models are finetuned until the validation loss minimum is reached. Finetuning experiments are only done in English, due to the lack of additional datasets in other languages. For Transformer trainings, we remove overlapping utterance pairs between the official train and test sets from the DailyDialog training set. We observed that inflated results reported on DailyDialog (Csáky et al., 2019) are partly due to this overlap. For all datasets we use lowercase input text and NLTK[13] word tokenization as preprocessing. We use the official DailyDialog splits and we employ a random train/dev/test split of 80/10/10 for PersonaChat, which we make publicly available along all the datasets used in this paper[14].

Gutenberg pre-training performs better than Opensubtitles on DailyDialog across nearly all metrics in both zero-shot and finetuned settings (Table 4a). Gutenberg pre-training outperforms even the model trained only on DailyDialog on some metrics. All GPT2 models are pretrained as language models on web text. Thus it comes as no surprise that the additional pretraining on

---

[13] `https://www.nltk.org/`

[14] `https://github.com/ricsinaruto/gutenberg-dialog`

Input	**TRF**	**GPT2**	**GUT ZS**	**OPEN FT**	**GUT FT**
how are you doing today EOU awesome . just sitting here listening to some stones . how are you ? EOU i ’m good . just about to play some dd	what do you play ? i ’m a professional athlete .	what kind of music do you play ?	huh ! what do you think of that ?	i ’m just thinking about having a ball .	what are you watching ?
lol i scratch my nose when oreo shakes tickle it EOU you are funny . not so depressed about welfare and dad in ace rated now EOU ouch sorry your dad in jail mine passed before he could teach me to drive	i ’m so sorry . i ’m so sorry .	lol you should take him to the doctor . it is a good therapy	but you ’re not a boy any more .	i think he was . my mom was not there either	oh okay . he ’s a detective . my dad passed before he could train me
i used to drive my mother crazy i liked to smoke i am tee total now though EOU i just finish smoking . i take anxiety medication at night EOU that ’s not good having anxiety i mean . does the medication help ?	i do not know . i am very sorry .	it works for me but i have to get up early so i can smoke	you ’re perfectly right you ca n’t do much harm .	it does help . i am a teacher . i teach middle school kids	no but i can be on the watch with my friends . i ’m getting to see my dad
i ’ve one dog she ’s amazing . EOU i ’ve 2 dogs . i should take them walking instead of eating . EOU that would be a great idea .	i ’m so excited to go to the movies .	i think that ’s too bad for my health .	very well . it ’s a bargain .	what do you do for a living ?	i ’ve no opinion about it . i ’m very passionate about animals .
wonderful . i hail from germany . what about you ?	i ’m a small world . i ’m a waitress .	i ’m from chicago how about you ?	i ’m a british spy .	i am a student in the us .	i ’m from baltimore and i ’m also from florida

Table 5: Random test samples from PersonaChat. TRF is the base Transformer and GPT2 is the non-pretrained GPT2 model. GUT and OPEN refer to Gutenberg and Opensubtitles, respectively, and ZS and FT refer to zero-shot and finetuned settings, respectively. EOU means “End Of Utterance”.

Gutenberg does not lead to the same relative improvement as with the Transformer models, which are trained from scratch. Gutenberg pre-training achieves better results than Opensubtitles in all metrics after finetuning on PersonaChat (Table 4b). In the Transformer zero-shot scenario, Opensubtitles achieves better BLEU scores, however, zero-shot BLEU scores are generally much lower than randomly selected responses, questioning the validity of this comparison. Gutenberg pre-training outperforms the baseline PersonaChat training on some metrics after finetuning. Considering the domain mismatch between the older Gutenberg books and the modern chit-chat style datasets this is especially impressive. Since the metrics are all very similar it is also important to look at responses qualitatively. Table 5 presents 5 random test samples. More samples from both DailyDialog and PersonaChat can be found in Appendix A.3. It is clear that the Transformer and the zero-shot GPT2 scenario perform the worst, followed by the finetuned Opensubtitles training. This shows some anecdotal support for the effectiveness of pre-training on Gutenberg.

Table 6 compares Gutenberg and Opensubtitles trainings across all seven languages, using roughly the same amount of data. In absence of a third independent data source we create mixed test datasets for each language that include the same amount of data from Gutenberg and Opensubtitles, by limiting the larger of the two to the size of the smaller. Except for Hungarian, models trained on Gutenberg perform better on more metrics than Opensubtitles trainings. On some metrics, models perform worse than random responses from the training set. This is expected for entropy and distinct metrics, but we believe that BLEU scores would be higher after further training since overfitted models have been shown to perform better on these metrics (Csáky et al., 2019). This lack of stopping criteria also makes a fair comparison challenging. Example responses from all models are shown in Appendix A.3. To our knowledge, this is the first work to use non-English languages from the Opensubtitles dataset for dialogue modeling, and there are very few chatbot models in non-English languages in general.

6 Conclusion

We presented the Gutenberg Dialogue Dataset consisting of 14.8M utterances in English and smaller

		$\|U\|$	H_w^u	H_w^b	H_u^u	H_u^b	D_{kl}^u	D_{kl}^b	AVG	EXT	GRE	COH	d1	d2	b1	b2	b3	b4
EN	G	**8.8**	**7.77**	13.4	**69**	**105**	.331	.707	**.494**	.468	.518	**.529**	.0034	.037	.0806	**.0879**	**.0883**	**.0828**
	O	6.1	7.68	13.4	47	68	**.292**	**.689**	.472	**.475**	**.522**	.519	**.0048**	**.045**	**.0867**	.0855	.0810	.0739
	RT	14.3	9.21	16.4	135	223	.038	.148	.462	.443	.485	.462	.0139	.150	.0671	.0879	.0946	.0915
	GT	14.1	9.14	16.0	132	208	0	0	1	1	1	.526	.0089	.130	1	1	1	1
DE	G	**7.4**	7.98	13.9	**60**	**84**	**.194**	**.500**	**.536**	.581	**.581**	**.576**	**.0387**	**.241**	.0803	.0813	.079	.0734
	O	6.4	**8.12**	**14.3**	52	72	.269	.635	.524	.581	.579	.566	.0329	.236	**.0825**	**.0864**	**.083**	**.0769**
	RT	15.6	9.47	16.5	152	246	.106	.265	.519	.548	.560	.518	.0910	.453	.0723	.0946	.101	.0984
	GT	15.0	9.15	15.5	139	186	0	0	1	1	1	.583	.0610	.392	1	1	1	1
NL	G	**6.8**	7.81	13.8	**53**	**76**	**.214**	**.624**	.503	.526	.581	.541	**.0453**	**.282**	**.0858**	.0854	.083	.077
	O	5.8	7.79	**14.0**	45	64	.388	.922	.504	.524	.580	.543	.0382	.252	.0850	**.0869**	**.084**	.077
	RT	15.4	9.15	16.0	143	233	.155	.455	.513	.505	.566	.512	.0961	.487	.0855	.108	.115	.111
	GT	14.4	9.04	15.5	129	172	0	0	1	1	1	.558	.0659	.404	1	1	1	1
ES	G	**8.0**	7.16	12.1	**58**	**83**	.373	.744	**.452**	.471	**.524**	.473	.056	.242	**.0883**	**.0839**	**.0788**	**.0723**
	O	5.8	**7.76**	**13.4**	46	61	**.198**	**.621**	.438	.466	.516	**.507**	**.093**	**.397**	.0840	.0771	.0716	.0642
	RT	12.2	8.95	15.3	111	174	.127	.226	.429	.421	.495	.426	.180	.633	.0763	.0908	.0936	.0896
	GT	14.5	8.47	14.1	122	153	0	0	1	1	1	.490	.119	.502	1	1	1	1
IT	G	**6.9**	7.59	12.7	**51**	**69**	**.183**	**.331**	**.452**	.486	.544	**.490**	.131	.451	**.0732**	**.0746**	**.0708**	**.0658**
	O	4.9	**7.89**	**13.6**	39	49	.266	.987	.434	.485	.538	.473	**.155**	**.558**	.0676	.0638	.0604	.0551
	RT	12.7	9.24	15.5	119	182	.163	.280	.452	.452	.518	.453	.253	.755	.0668	.0801	.0827	.0797
	GT	14.6	8.64	14.0	123	138	0	0	1	1	1	.522	.182	.614	1	1	1	1
HU	G	4.59	7.62	**13.2**	34.3	38	**.176**	.530	**.410**	.452	.520	.447	**.120**	**.463**	.086	.075	.0677	.0609
	O	**5.56**	**7.73**	13.0	**42.1**	**44**	.278	.538	.401	.447	**.529**	.442	.111	.419	**.106**	**.100**	**.0937**	**.0848**
	RT	9.62	9.68	15.6	95.5	136	.195	.355	.393	.406	.487	.391	.305	.788	.075	.087	.0893	.0849
	GT	7.71	9.04	14.8	65.5	72	0	0	1	1	1	.440	.220	.658	1	1	1	1
PT	G	**8.4**	7.44	12.6	**63**	**88**	.189	**.495**	.455	.409	.552	.474	.184	.575	.0886	.0933	.093	**.087**
	O	6.3	7.62	**13.0**	49	61	.226	.671	.443	.407	.544	.488	**.210**	**.627**	.0816	.0812	.078	.072
	RT	14.5	9.16	15.2	134	207	.118	.415	.441	.368	.503	.441	.316	.821	.0784	.0971	.104	.100
	GT	17.1	9.02	14.8	156	235	0	0	1	1	1	.506	.249	.712	1	1	1	1

Table 6: Comparing Gutenberg and Opensubtitles GPT2 trainings across 7 languages on the union of the two test sets. The second column shows whether the model was trained on Gutenberg (G) or Opensubtitles (O). Randomly selected responses from the respective train set (RT) and groud truth (GT) performance is also given. Significantly better results between Gutenberg and Opensubtitles (95% confidence interval) are highlighted on each test set.

datasets in German, Dutch, Spanish, Italian, Hungarian, and Portuguese. We described heuristics used in our dialogue extraction pipeline and conducted a detailed error analysis to uncover the causes of errors and to assess data quality. In a pre-training comparison between Gutenberg and Opensubtitles we found that Gutenberg performs better on downstream datasets in both zero-shot and finetuning scenarios. We release the Gutenberg dataset as well as the open-source pipeline[15] with which researchers can build their own datasets. We also built a web demo interface to all models presented in the paper[16].

In future work, we wish to improve heuristics and dataset quality. A classifier could be trained to decide whether two consecutive utterances are part of the same dialogue (looking at non-conversational context). Positive and negative examples could be generated by a very low/high dialogue gap, or by manual annotation. Speaker-related errors could be addressed using speaker identification. We also hope to extend our dataset to more languages. This involves delimitation analysis, implementation of heuristics, and error analysis. We welcome contributions from the community, as our open-source modular pipeline minimizes the effort required for adding new languages.

Acknowledgments

We wish to thank Márton Makrai for inspiration and discussion about the idea of extracting dialogue from books. We wish to thank members of the SZTAKI HLT[17] group and all anonymous reviewers for their help and thoughtful feedback. Work partly supported by Project FIEK 16-1-2016-0007, financed by the FIEK 16 funding scheme of the Hungarian National Research, Development and Innovation Office (NKFIH). Recski was partly supported by BRISE-Vienna (UIA04-081), a European Union Urban Innovative Actions project.

---

[15] We also release all data, trained models, and training scripts to produces the results.

[16] `https://ricsinaruto.github.io/chatbot.html`

[17] `https://hlt.bme.hu/en/`

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3970–3980. Association for Computational Linguistics.

Richárd Csáky, Patrik Purgai, and Gábor Recski. 2019. Improving neural conversational models with entropy-based data filtering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5650–5669, Florence, Italy. Association for Computational Linguistics.

Richard Csaky and Gabor Recski. 2017. Deep learning based chatbot models. National Scientific Students' Associations Conference.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 76–87. Association for Computational Linguistics.

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

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Joachim Fainberg, Ben Krause, Mihai Dobre, Marco Damonte, Emmanuel Kahembwe, Daniel Duma, Bonnie Webber, and Federico Fancellu. 2018. Talking to myself: self-dialogues as data for conversational agents. *arXiv preprint arXiv:1809.06641*.

Hao Fang, Hao Cheng, Maarten Sap, Elizabeth Clark, Ari Holtzman, Yejin Choi, Noah A. Smith, and Mari Ostendorf. 2018. Sounding board: A user-centric and content-driven social chatbot. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 96–100. Association for Computational Linguistics.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society.

Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. 2019. Learning from dialogue after deployment: Feed yourself, chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684, Florence, Italy. Association for Computational Linguistics.

Matthew Henderson, Paweł Budzianowski, Iñigo Casanueva, Sam Coope, Daniela Gerz, Girish Kumar, Nikola Mrkšić, Georgios Spithourakis, Pei-Hao Su, Ivan Vulić, and Tsung-Hsien Wen. 2019. A repository of conversational datasets. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ben Krause, Marco Damonte, Mihai Dobre, Daniel Duma, Joachim Fainberg, Federico Fancellu, Emmanuel Kahembwe, Jianpeng Cheng, and Bonnie Webber. 2017. Edina: Building an open domain socialbot with self-dialogues. In *1st Proceedings of Alexa Prize (Alexa Prize 2017)*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT 2016*, pages 110–119. Association for Computational Linguistics.

Jiwei Li, Will Monroe, Tianlin Shi, Alan Ritter, and Dan Jurafsky. 2017a. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169. Association for Computational Linguistics.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017b. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Pro-*

ceedings of the The 8th International Joint Conference on Natural Language Processing, pages 986–995. AFNLP.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132. Association for Computational Linguistics.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an automatic turing test: Learning to evaluate dialogue responses. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1116–1126. Association for Computational Linguistics.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. `https://github.com/openai/gpt-2`.

Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. 2018. Conversational ai: The science behind the alexa prize. *arXiv preprint arXiv:1801.03604*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence.

Xiaoyu Shen, Hui Su, Wenjie Li, and Dietrich Klakow. 2018. Nexus network: Connecting the preceding and the following in dialogue generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4316–4327. Association for Computational Linguistics.

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. Ruber: An unsupervised method for automatic evaluation of open-domain dialog systems. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*. Association for the Advancement of Artificial Intelligence.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *LREC*, Istanbul, Turkey. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2018. Transfertransfo: A transfer learning approach for neural network based conversational agents. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*.

Yujie Xing and Raquel Fernández. 2018. Automatic evaluation of neural personality-based chatbots. In *Proceedings of The 11th International Natural Language Generation Conference*, pages 189–194. Association for Computational Linguistics.

Can Xu, Wei Wu, and Yu Wu. 2018a. Towards explainable and controllable open domain dialogue generation with dialogue acts. *arXiv preprint arXiv:1807.07255*.

Xinnuo Xu, Ondřej Dušek, Ioannis Konstas, and Verena Rieser. 2018b. Better conversations by modeling, filtering, and optimizing for coherence and diversity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3981–3991. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 2204–2213. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

A Appendix

A.1 Training hyperparameters

Name	Value
Hidden size	512
Number of hidden layers	6
Label smoothing	0.1
Filter size	2048
Number of attention heads	8
Layer dropout	0.1
Relu dropout	0
Attention dropout	0
Learning rate	0.2
Learning rate warmup steps	8000

Table 7: Transformer hyperparameters.

Name	Value
LM loss coefficient	2
Multiple-choice loss coefficient	1
Max. gradient norm	1
Gradient accumulation steps	8
Nucleus sampling p	0.9
Context size	1024
Embedding size	768
Number of attention heads	12
Number of layers	12
Vocabulary size	50262

Table 8: GPT2 hyperparameters.

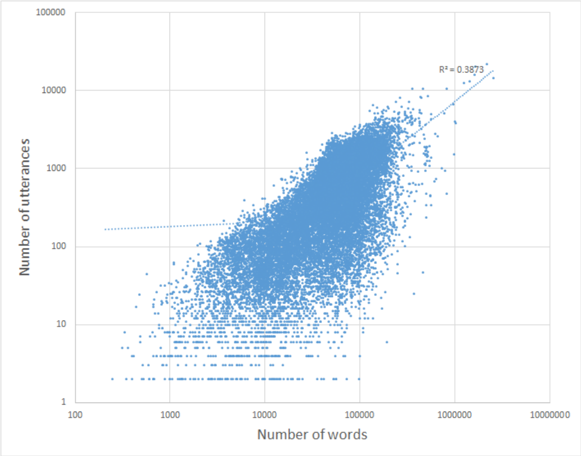


Figure 7: Number of extracted utterances with respect to number of words in each book on logarithmic scales (English Gutenberg dataset).

A.2 Gutenberg statistics

A.3 Example responses

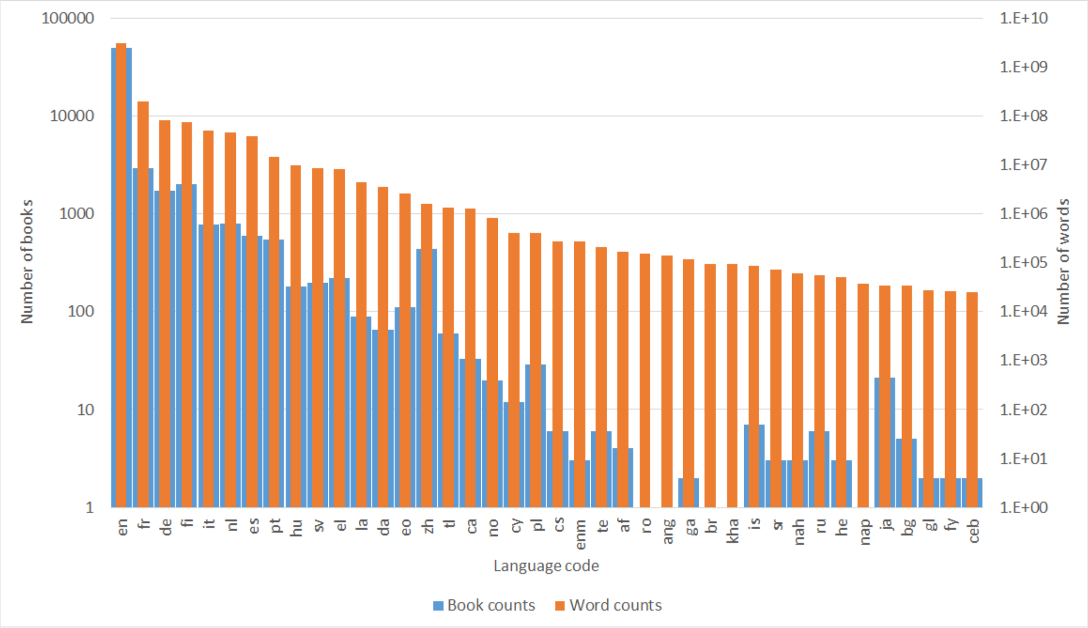


Figure 8: Number of books and words for top 40 languages in Project Gutenberg on logarithmic scales.

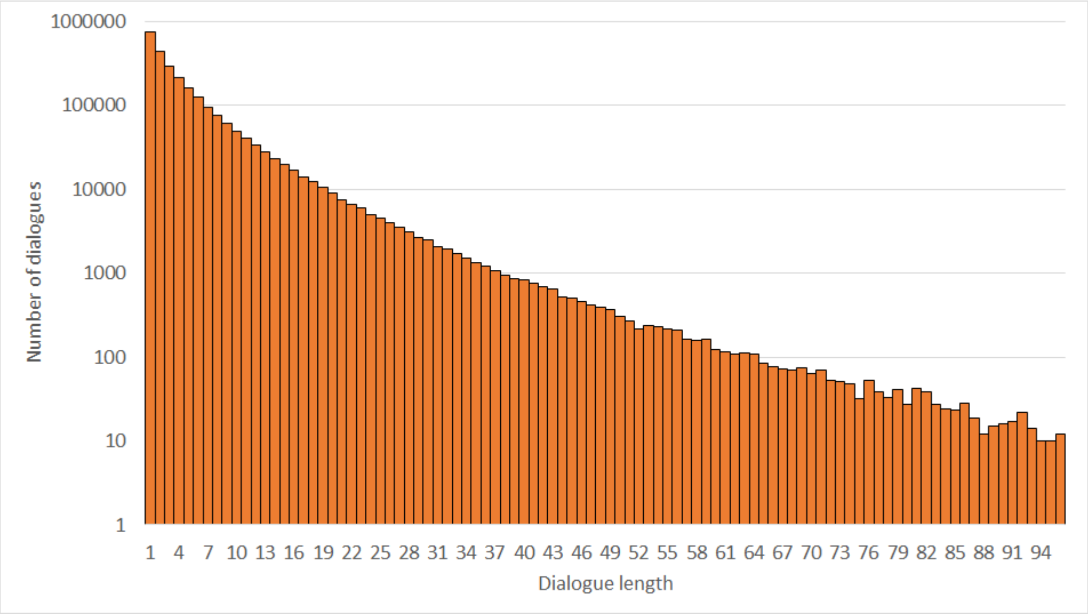


Figure 9: Distribution of dialogue lengths in the English Gutenberg dataset (truncated).

Input	**TRF**	**GPT2**	**GUT ZS**	**OPEN FT**	**GUT FT**
stay for dinner with us . EOU i 'm afraid i ca n't . i have a dinner party at seven tonight . you see i am going to england the day after tomorrow . EOU well i wo n't keep you then . it was very nice of you to come to see me .	you 're welcome .	thank you . i will .	i knew you would n't mind .	yes . i hope you feel better .	it was my pleasure .
sure . take a seat . . . what would you like to know ? EOU first i ve heard that your latest book is based on a true story . EOU that s correct . it s a murder mystery based on actual murders that book place in florida several years ago . the main characterthe police invesigatoris based on the man who investigated the case .	i see . i thought it was a great idea . i could do that .	oh i see . thanks a lot . i ll have to check it out .	well ?	i ve never been involved in an actual murder . what are your chances of catching the killer ?	so what happened to the man who investigated the case ?
well i do n't eat any meat but i do eat fish and eggs . EOU do you eat a lot of vegetables ? EOU oh yes and fruit i love fruit .	what kind of food do you like ?	do you like fruits and vegetables ?	and do you eat a good many things ?	do you like fruits ?	what do you do with your vegetables ?
that 's terrific . i ca n't wait to check it out . EOU hang on . are you sure you want to keep a nontraditional pet ? EOU why not ? it 's so cool .	i do n't know . i do n't know what to do .	the pets come with a 2 week guarantee .	i 've been in the market for two years .	i 'm really glad that you came .	then let 's go .
that 's my boy ! katherine curtis created this activity in 1920 . and in 1984 it was authorized as one activity in olympic games . EOU wow mom is more knowledgeable than dad . i must learn from you . EOU my little boy you should learn more from me . i 'm almost thirty years older than you .	i 'm not sure . i 'm a bit nervous .	dad i 'm not a child anymore . i 'm a grownup .	you 're a nice old lady mom .	okay . i will begin learning when i grow up .	what 's the matter with your boy ?
it will cost you 10 . EOU by the way what 's the time ? EOU you can see the time on the meter . i have radio time you know .	i know . i do n't know what to do .	that 's fine . i 'll see it tomorrow .	you had a good time in new york ?	you can watch the news .	i see .
hello . is bob there please ?	hello . this is mr . green .	yes he is . how 's your business going ?	yes my dear but is n't he coming ?	yes . he 's gone out to the market .	speaking .
it seldom rains this summer .	i think it 's a bit expensive .	why ?	you are always on the watch for rain	i am sure it does .	what is the weather like today ?
i think i may have seen it on the kitchen table .	i m afraid i have a bit of a small town .	it 's great . i 've enjoyed it .	i 'm sure you did	no i didn t . i didn t see it .	what did you see ?
thanks . and could i have his email just in case i ca n't get him by phone ? EOU sure . his cell phone is 09112223 33 . and his email is lower case t smiththat 's one word at c c w dot com dot t w . EOU thank you so much . does he read his emails daily ?	yes . he was in the middle of the city .	not every day . he just sends email to his friends .	sure . he has n't been here since the night of the murder .	yes he does . he reads a lot of letters per day .	yes he does . he 's a very good student . i can read and write and he keeps in touch with my friends .

Table 9: Random test samples from DailyDialog. TRF is the base Transformer and GPT2 is the non-pretrained GPT2 model. GUT and OPEN refer to Gutenberg and Opensubtitles, respectively, and ZS and FT refer to zeroshot and finetuned settings, respectively. EOU means “End Of Utterance”.

Input	**TRF**	**GPT2**	**GUT ZS**	**OPEN FT**	**GUT FT**
i do n't but that would be great EOU what do you do for a living ? EOU i work w french fries . i stink of them	that 's cool . i 'm a waitress .	do you have any pets ?	you mean to say that you do n't learn much ?	that sounds like a hard job . do you like to read ?	oh i see . i am a legal assistant .
great ! just came back from a walk with my bipolar dog . EOU that sounds nice i 've been taking care of a buddy horses this afternoon . EOU why ? wheres the hubby ?	i 'm not sure . i 'm not sure .	he has a dog i 've a dog too !	he 's in the stable .	he 's a shepherd and is very smart .	i 'm trying to get him to see me .
that is so sad i 'm sorry EOU i never see my dad now he moved to guam EOU i do not even know where guam is	i 'm a girl . i love to eat out .	where do you live now ?	i thought he was in the penitentiary	i do n't even know if i 've ever heard of it	i have always wanted to move there but he passed away
that 's very interesting . i workout for a living .	i do too . i am a waitress .	i 'm retired and working with animals .	you do eh ?	that 's cool . i like to eat breakfast .	oh my god i hate eating too much . i could n't do it .
my one child wanted to be like that too EOU it is good for the body EOU do you train yourself alone ?	i do . i am a waitress .	i do i try to at least once a month	yes .	no i do n't have a job .	yes i have a job in a factory
i am 35 and still single . EOU are you looking forward to turning 40 ? EOU no not at all . it seems dreadful .	i agree . i 'm very excited about that .	i 'm sure you are . what is your name ?	i am sorry to hear it .	what do you do for work ?	i have a dog named george . what do you do for a living ?
being from springfield missouri the midwest does not offer much EOU i can not wait to grow up EOU do n't rush it . . . believe me stay young	i am very tall so i do not have to go to the gym	i do n't want to be the boss of a nascar car	you know what i mean i ca n't let you go .	i will try it	i will be twenty in march
congrats ! i have two little ones and a wife back stateside and i miss them so much ! EOU i live with my best friend on a lake EOU sweet life . beats being in this desert although i got ta admit i love my long deployments	i 'm a bit of a guy . i love to swim .	i like to swim in the mountains	then why not stay at home ?	i am still in college but i am doing my masters	i 'm sorry to hear that
hi . how are you doing ? EOU well . . i got ta tell the truth to somebody i just have too EOU go ahead . i 'm almost seven feet tall by the way	i 'm a girl . i 'm tall .	that is tall ! you can ride horses and trucks .	oh i ai n't ! i 'm eight feet eight .	wow ! ! i 'm just like a sponge . . . i like to work with spongebob	wow that 's a huge size . i just bought a new new dress a few times .
hi i 'm jane and a part of my skin is blue i 'm 177 cm . EOU your skin is blue ? my names bill nice to meet you EOU yeah it was a dye i was using and it seeped into the skin .	i like to draw and paint .	oh wow that 's really pretty	then it was not a dye .	that is amazing ! what do you do for a living ?	what color was it ? i love the color red

Table 10: Random test samples from PersonaChat. TRF is the base Transformer and GPT2 is the non-pretrained GPT2 model. GUT and OPEN refer to Gutenberg and Opensubtitles, respectively, and ZS and FT refer to zeroshot and finetuned settings, respectively. EOU means "End Of Utterance".

Input	**Gutenberg**	**Opensubtitles**	**Target**
hogyan ? ön képes 200 forintot adni ? EOU nagyon . EOU eszerint jól áll a lapja ?	én .	nem vagyok a közelőtt .	kitűnően .
hát a fejét ne kössük be ? EOU mivel ? EOU vizes kendővel .	hogyan ?	felség , azt mondja a nő egy közelést .	fölösleges .
tessék kérem !	tessék , hogy miért nem ?	tudom . . .	egyedül van ?
veszünk ki más lakást . EOU meglövöldöz bennünket az öreg . EOU ne hidd azt !	meglövöldöz bennünket ,	ne légy nem akarják .	dehát miből éljünk ?
barátom , jób , az eördög megszökött . a miskolczi részvényeket fölapríthatod fidibusznak . EOU míriz olezáncz ! EOU barátom , jób ! szomorú hírt hozok : a török sorsjegyek elvesztették az értéküket !	szegény asszony !	mert őket , hogy megint mondtam .	míriz olezáncz !
ott vettem ezt a kést egy kis boltban , két utcával arrébb . EOU hat dollárba került . EOU a törvény tiltja rugóskések vásárlását .	ah !	ez egy kis kölykökre , hogy elváltoztatni az ügyvédő .	így van , megszegtem a törvényt .
honnan szerezte ? EOU tegnap este a tárgyalás után sétálni mentem . EOU a fiú lakása környékén .	hát nem szerezni , hogy nagyon szerezni !	én volt rá , mi történt ?	ott vettem ezt a kést egy kis boltban , két utcával arrébb .
oké . EOU esetleg megegyezhetnénk egy határidőben . EOU még egy tiszteletkör , és kész .	egy határ úr , hogy öreg asszonyt üve	és előtt a gyilkosságot .	most negyed 7 van .
ki mondja , hogy nem bűnös ? EOU egy . EOU rendben .	kicsoda ?	csak az ügyvéd volt .	tizenegy bűnös , egy nem bűnös .
én egyszerűen csak kérdéseket teszek fel . EOU ez most jutott eszembe . . . EOU ha én öltem volna meg az apámat , én is visszamentem volna a késért .	ugyan , ugyan !	a feleségét .	a srác azt hihette , hogy nem látták , amikor elrohant , és másnap reggelig nem fedezik fel

Table 11: Random test samples from the Hungarian Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means “End Of Utterance”.

Input	**Gutenberg**	**Opensubtitles**	**Target**
madame if i had the tact and wit of my brother charles i would say that the sun which so lately has risen must not set so soon upon its worshippers . but answer me one questionwhat is the meaning of the riddle with which your majesty has been entertaining us ? EOU may i answer with another question ? tell me sire have you ever seen the sun rise ? EOU i ? no your majesty . i confess that i never have .	well sire then i will tell you a secret .	the sun rises in the sky .	and you ladies and gentlemen ?
wo n't he confess how he became possessed of the diamonds ? EOU no not to me . he refuses to say a word in his own defence . EOU then i tell you what this quixotic young man is defending another person he is shielding the assassin .	you think he is innocent ?	the longer i wait he wo n't tell me anything .	if he is that shows him to be an accessory either before or after the fact but who is the person you
and who ought of right to avenge him ? EOU sir the knight that was in the red launde at the assembly that jousted with messire gawain and had the prize of the tournament . EOU did he better than messire gawain ?	he did he did indeed .	yes sir .	sir so did they adjudge him for that he was a longer time in the assembly .
why do n't you keep your mouth shut philippo ? it 's always talk talk with you . we are chosen because the captain can rely upon us .	but i have no right to speak to you and i 'm not going to be treated as if i were a child .	the captain is the captain .	he can rely upon anyone who knows that he will get his throat cut if he fails in his duty .
i 'd rather talk to you in my room if you please	so would i	but it 's not your fault .	all right
so what 's your take on that ? EOU i got nothing . EOU what am i gon na do ?	do n't worry . i 'll keep you a prisoner .	you do n't have to do anything .	i do n't know .
my face . EOU there was an argument . EOU i	what ?	i have no idea what happened .	i said some things i should n't have said and i
what do you do when several people attack you at once ? EOU get up EOU no matter how many i focus on just one	one for me !	do n't you want me to get into trouble ?	i was in a fight against five guys i went after one of them i did n't care what the other four did i
i do n't know . EOU come with me . EOU come on .	i ca n't	i do n't know .	you do n't need it come on .
they were all using me . EOU hanford everybody ! EOU but you did n't kill everybody .	you do n't know what it means to kill people .	you 're not .	you killed randy chase .

Table 12: Random test samples from the English Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means "End Of Utterance".

Input	**Gutenberg**	**Opensubtitles**	**Target**
pues debe de haber muchas . pablo penáguilas que las sabe todas , me ha dicho que son muchas y que la vida entera de un hombre no basta para una sola .	es que no hay que dar las solas . esos que la vida no tiene nada que hacer por una sola que se les ha de hablar con el cuerpo .	no le ha dicho a nadie .	ríete tú de eso . . . . ya me verás a mí . . .
¿ y por qué han de estar exentos de esa graciosa ley los solteros ricos ? ¿ por qué no han de cargar ellos también con su huérfano , como cada hijo de vecino ? EOU no me opongo . ¿ pero qué es esto ? . . . ¡ sangre ! EOU ¡ jesús ! . . . . si es la nela . mira cómo se ha puesto los pies .	¿ y no quiere usted que lo han robado ?	no .	ya se ve . . . . como tuvo que meterse entre las zarzas para coger a tu dichoso lili . nela , ven ac
¿ y roberto ?	no , no me llamo .	¿ y roberto ?	no sé , por ahí andará .
echaremos por estos prados adelante . ¿ a ver qué me has traído hoy ? EOU busca bien y encontrarás algo EOU ¡ ah , madre de dios ! chocolate crudo . . . ¡ y poco que me gusta el chocolate crudo ! . . . nueces . . . una cosa envuelta en un papel . . . ¿ qué es ? ¡ ah ! ¡ madre de dios ! , un dulce . . . . ¡ dios divino ! , ¡ pues a fe que me gusta poco el dulce ! ¡ qué rico está ! en mi casa no se ven nunca estas comidas ricas , pablo . nosotros no gastamos lujo en el comer . verdad que no lo gastamos tampoco en el vestir . total , no lo gastamos en nada .	¡ ah ! ¡ no lo sé !	¡ qué hizo eso !	¿ a dónde vamos hoy ?
quita allá , tonta , tú eres una alhaja . EOU que no señor . si no puedo trabajar . en cuanto cargo un peso pequeño , me caigo al suelo . si me pongo a hacer alguna cosa difícil en seguida me desmayo . EOU todo sea por dios . . . . vamos , que si cayeras tú en manos de personas que te supieran manejar , ya trabajarías bien .	¿ qué decís ?	¿ se encuentra en personas ?	no , señor ; si yo no sirvo más que de estorbo .
iluminó ciertos rincones oscuros del alma humana . EOU mostró al mundo las almas de aquellas personas que nacieron diferentes al resto . EOU para entender por qué estaba destinado a esto , tendríamos que retroceder en el tiempo a cualquier día de su infancia , en la ciudad de colorado springs .	¿ y qué ?	sólo una pelea , nunca había terminado .	no tuvo una infancia fácil , ya que sus padres eran diferentes .
algunos se fueron a sudamérica . EOU lo sé , señor . EOU ¿ por qué no se fue ?	¿ por qué ?	¿ por qué no ?	porque creo que este país no forma parte de los estados unidos .
ella es muy simpática . EOU es mi deber . EOU ella es muy simpática .	pues es muy poco que una enferma .	es mi deber .	¿ y bates ?
anoche vimos un estreno en el metropolitan . EOU ¡ eh , es el tercero ! EOU ¡ son tan buenos !	¿ son tan buenos ?	¡ no lo se !	ya sé que son buenos .
estás equivocada . EOU ¿ qué más da lo que piense ? EOU cuando llamas a alguien asesino , le suele molestar .	¿ y quién es el espectáculo ?	¿ y usted ?	no se lo diré a nadie .

Table 13: Random test samples from the Spanish Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means “End Of Utterance”.

Input	**Gutenberg**	**Opensubtitles**	**Target**
intendo di fare il mio dovere . . . denunziar tutto alla giustizia . . . la mia nomina a capo agente è combattuta dai miei rivali . . . si accorgeranno tutti sempre più che razza d'uomo è lucertolo , e che con lui non si combatte ! EOU lucertolo ! . . . EOU voi non agirete così ! . . . sarebbe una grande viltà	come ? . . .	ma è che hai fatto ?	ma denunziate pure denunziate pure la ragazza di piazza degli amieri . . . come voi la chiamate . .
tutti . . . chi . . . ? EOU il pittore . . . tu . . . la ragazza di via degli amieri . . . EOU la ragazza di via . . .	che cosa ?	davvero ?	eh , sì , mia cara . . . l'ho veduta un par di volte soltanto , mentre pedinavo te , e l'ho subito r
due avvenimenti straordinarii si sono compiuti la sera del 14 gennaio . . . l'assassinio nel vicolo della luna . . . e un altro avvenimento al quale non ho voluto accennare nella difesa per ragioni delicate . . .	volete dire ?	e'tutto , signore .	il secondo avvenimento è . . . la sparizione di una ragazza che abitava in piazza degli amieri .
ma la collana si ritroverà . . . ne sono certa - soggiungeva la principessa	che cos'ha ?	ecco .	anch'io basta guardare quella ragazza , per escludere ogni accusa come un'infamia . . .
e il babbo . . . e la mamma ?	sì , il babbo .	sì , ai babbo , ai babbo .	stanno meglio , e credo che tu li potrai salvare !
sono cose che non si fanno . EOU mi spiace , signore , ma non posso considerarla una seria obiezione . EOU l'ha sentito , lanyon ?	sì ,	sì .	è un'altra prova della sua eccentricità ?
tenete . EOU datele del cognac . EOU questo gli incoraggerà il cuore .	perdono .	non è mica una cosa .	siete una buon uomo .
qual ' è il suo nome ? EOU parla , stupido ! EOU mata hari .	e chi è quel che dice ?	franco , lasciami perdere .	questo è il suo nome , vero ?
di cosa vivrebbe il popolo dell'altai . . . se non avesse le pecore ? EOU sono un'insegnante e devo parlare col capo del soviet rurale . EOU che cosa insegnerai ?	di che cosa c ' è ?	ci penso io .	sola . . .
divertente ? EOU con signac ? EOU è necessario ?	lo spero .	si è una ragazza di peggioriose .	perchè no ?

Table 14: Random test samples from the Italian Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means "End Of Utterance".

Input	**Gutenberg**	**Opensubtitles**	**Target**
daarvoor is reeds gezorgd .	wat hebt gij dan te zeggen ?	kom binnen , kom binnen .	neen , o neen , naar het kot ga ik niet ! dood mij liever vr uwe voeten !
men kan niet weten , ’t is altijd goed , een speld bij zich te hebben . EOU wat wilt ge dan toch van nacht uitvoeren ? EOU zaken .	wat zegt ge ?	heb je ’ n gezin ?	apropos .
hij zegt , dat hij in elk geval binnenkomt , maar hij wou ’t liever met uw toestemming doen .	maar de jongeheer van ’t zelfde , h ?	waarom ben je hier ?	heb je gezegd , dat ik aan ’t werk was ?
wel , flipsen , wat scheelt je ? heb je je bezeerd ? EOU ja , burgemeester , ik heb me bezeerd , EOU zoo , hoe komt dat ?	komt daar nu op den grond !	zijn het niet ?	dat weet ik niet , burgemeester ,
leve jan verhelst ! leve mie - wan na !	leve jan verhelst !	wat bedoel je ?	leve sander ! leve sander ! hoera ! hoera !
een naald in ’ n naaldberg . EOU en onze compagnie ? EOU de besten voor ons , de rest naar b .	en de rest naar b . ?	gaan jullie naar binnen ?	jezus christus .
goed zo , meid . EOU dat is mijn molly . EOU gaat het goed met hem ?	wij zijn met hem ,	hij is er .	ja , maar hij wil gewoon niet slapen .
vijf man is ’ n doel . EOU eentje is zonde van de munitie . EOU hou ’t zand uit je wapen , zorg dat ’t blijft werken .	vijf man is ’ n doel .	je hebt haar vermoord .	tot zo , op ’t strand .
ik heet kovu . EOU ik heet kiara . EOU jij bent ’m .	zeg , hoeveel zijn d ’ r ?	ik heet kovu .	jij bent ’m .
je . . . EOU bent u gekomen om dat te zeggen ? EOU je moet naar huis .	waarom niet ?	de volgende keer niet .	we hebben bevel je terug te brengen .

Table 15: Random test samples from the Dutch Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means “End Of Utterance”.

Input	**Gutenberg**	**Opensubtitles**	**Target**
desculpai , minha boa senhora , rosinha é minha neta . EOU sim , snr . a d . thereza , é minha avó , de quem tantas vezes tenho fallado a v . exc . a e . . . EOU então porque não continuas ?	não , snr . a d . thereza , não foi nenhuma . não sei o que é que eu digo : eu conheço - o ao sr . seabra . . .	só um bom rapaz , não .	falla , falla , minha menina . não tenhas receio . queres pedir - me alguma cousa , não é assim ?
estiveste incommodada , minha filha ?	um pouco .	não , obrigado .	não , minha senhora . este cestinho , que aqui trago , é que foi a causa da minha demora .
não , minha senhora . este cestinho , que aqui trago , é que foi a causa da minha demora . EOU como é lindo não sabia julia , que tinhas a prenda de fazer cestos de juncos entrançados . EOU não fui eu que fiz este cestinho , minha mãi .	e tem razão , eu não posso dizer ao senhor simão , que está a dizer que esta senhora que não haja para aqui .	eu não estou apenas .	então quem foi ?
aonde vamos nós , rosa ? EOU em meio caminho , minha avó . EOU jesus senhor , valei - me , pois que as minhas pobres pernas já estão cançadas , e parece - me que não chego ao fim da jornada .	então , vamos lá !	o que é que eu não ?	encoste - se ao meu hombro , avósinha , que eu não estou cançada .
é muita honra para mim , minha querida senhora ; estou portanto ás vossas ordens .	e então não sabe ?	se quiser a senhora .	visto isso não vos recusareis a dizer - me se estaes satisfeita com a vossa neta ?
sr . hathaway . EOU onde está ferrante ? EOU como poderia saber ?	não .	pare de ser passá - lo para um pouco .	um passarinho me contou que ele deveria estar aqui .
a capela parece encantadora . EOU não quer entrar ? EOU posso ?	muito bem .	não .	faça favor .
você acha que eles estão apaixonados ? EOU não saberia dizer , gladys . EOU estou louca pra ver nickie ferrante .	oh ! não me entendo !	diz - me .	não sei qual foi a pergunta de 64 milhões de dólares , mas ele tinha a resposta .
não pense que foi tudo . . . EOU esperem aí . EOU fazem - me um favor ?	não , mas é preciso que não vou .	quem é um , não sei .	o quê ?
ele está fazendo uma liquidação . EOU temos que pensar sobre isto . EOU onde podemos te encontrar ?	eu sei ? . . .	o que que foi com isso ?	estarei na sala de coquetel ou , claro , na sala de jantar .

Table 16: Random test samples from the Portuguese Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means “End Of Utterance”.

Input	Gutenberg	Opensubtitles	Target
sieh , sieh , rico , die sonne , wie schön ! jetzt wird ' s sommer ; sieh , wie es glitzert auf dem see . es kann gar keinen schöneren see geben , als der ist	was ist denn das ?	unverletzlich	ja , ja , stineli , du solltest nur einmal den see sehen , den ich meine !
natürlich hab ich dich liebgehabt .	soll ich auch gleich an mein leben herkommen ?	du warst doch gar nicht wirklich ?	gehabt !
was , mögen sie in den barbierladen gehen und dort warten , bis sie an die reihe kommen ? sie können ihn ja zu sich kommen lassen . "	nein , nein , sie kommen nicht zu mir , " erwiderte perthes .	jetzt sofort .	wir demokraten sind nicht so vornehm , " erwidert der rechtsanwalt .
es gibt ärgere dinge als gefangenschaft EOU du galoppierst mit verhängtem zügel . EOU das tempo macht leben	so laß es mir selbst sagen .	und wir müssen uns anfangen	ich will mir das heut abend in mein tagebuch schreiben .
hallo , das boot ! EOU hallo , das ufer ! EOU setzt eure yolle aus und schickt die indianerin an 's land ,	jetzt nicht , was die indianerin an 's boot schickt ,	guten tag .	was giebt 's ?
du kannst dir innerlich was zerrissen haben . EOU du wirst dir eines tages noch den hals brechen ! EOU gott , birdy !	du kannst ihn ja nicht überlassen !	was denn ?	tu so etwas nie wieder !
sei still . EOU sollen wir ihr wasser nehmen , muad'dib ? EOU wir gaben kampflos auf .	ja , wir gaben kampflos ,	wir gaben die ganze zeit , muad 'ib .	wir ergaben uns eurer gnade .
wir können da nicht hinein . EOU wir haben keine wahl . EOU nichts überlebt in einem solchen sturm .	was ist denn das für ein sturm ?	so ist es nicht .	wenn sie nicht sterben wollen , werden sie uns nicht folgen , oder ?
gehst du mit ihm ? EOU tina , darf ich etwas sagen ? EOU tina ?	ja , tina , ich habe mit dir gesprochen .	ich hab 's ja gekämpft .	was macht ihr denn mit dem geld ?
und jetzt soll ich es über nacht da stehen lassen ? EOU holen sie es morgen wieder ab . EOU das geht schon .	das geht schon ,	ich bin ein richtiges ekel .	wenn irgendwas mit diesem auto passiert , dann werde ich böse , tina .

Table 17: Random test samples from the German Gutenberg test set (first 5 rows), and Opensubtitles test set (last 5 rows). Responses from the GPT2 model trained on Gutenberg and Opensubtitles are compared. EOU means "End Of Utterance".

On the Calibration and Uncertainty of Neural Learning to Rank Models for Conversational Search

Gustavo Penha
Delft University of Technology
Delft
Netherlands
`g.penha-1@tudelft.nl`

Claudia Hauff
Delft University of Technology
Delft
Netherlands
`c.hauff@tudelft.nl`

Abstract

According to the Probability Ranking Principle (PRP), ranking documents in decreasing order of their probability of relevance leads to an optimal document ranking for ad-hoc retrieval. The PRP holds when two conditions are met: **[C1]** the models are *well calibrated*, and, **[C2]** the probabilities of relevance are reported *with certainty*. We know however that deep neural networks (DNNs) are often not well calibrated and have several sources of uncertainty, and thus **[C1]** and **[C2]** might not be satisfied by neural rankers. Given the success of neural Learning to Rank (L2R) approaches—and here, especially BERT-based approaches—we first analyze under which circumstances deterministic neural rankers are calibrated for conversational search problems. Then, motivated by our findings we use two techniques to model the uncertainty of neural rankers leading to the proposed *stochastic rankers*, which output a predictive distribution of relevance as opposed to point estimates. Our experimental results on the ad-hoc retrieval task of conversation response ranking[1] reveal that (i) BERT-based rankers are not robustly calibrated and that stochastic BERT-based rankers yield better calibration; and (ii) uncertainty estimation is beneficial for both risk-aware neural ranking, i.e. taking into account the uncertainty when ranking documents, and for predicting unanswerable conversational contexts.

1 Introduction

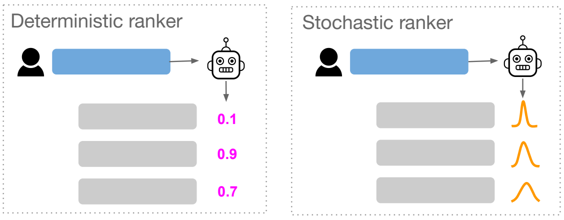


Figure 1: While deterministic neural rankers output a point estimate probability (magenta values) of relevance for a combination of query (blue bars) and document (grey bars), stochastic neural rankers output a predictive distribution (orange curves). The dispersion of the predictive distribution provides an estimation of the model uncertainty.

According to the Probability Ranking Principle (PRP) (Robertson, 1977), ranking documents in decreasing order of their probability of relevance leads to an optimal document ranking for ad-hoc retrieval[2]. Gordon and Lenk (1991) discussed that for the PRP to hold, ranking models must at least meet the following conditions: **[C1]** assign well calibrated probabilities of relevance, i.e. if we gather all documents for which the model predicts relevance with a probability of e.g. 30%, the amount of relevant documents should be 30%, and **[C2]** report certain predictions, i.e. only point estimates such as, for example, 80% probability of relevance.

DNNs have been shown to outperform classic Information Retrieval (IR) ranking models over the past few years in setups where considerable training data is available. It has been shown that DNNs are not well calibrated in the context of computer vision (Guo et al., 2017). If the same is true for neural L2R models for IR, e.g. transformer-based models for ranking (Nogueira and Cho, 2019), **[C1]** is not met. Additionally, there are a number of sources of uncertainty in the training process of neural networks (Gal, 2016) that make it unreasonable to assume that neural ranking models fulfill **[C2]**: *parameter uncertainty* (different combinations of weights that explain the data equally well), *structural uncertainty* (which neural architecture to

---

[1] The source code and data are available at `https://github.com/Guzpenha/transformer_rankers/tree/uncertainty_estimation`.

[2] Standard retrieval task where the user specifies his information need through a query which initiates a search by the system for documents that are likely relevant (Baeza-Yates et al., 1999).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 160–170
April 19 - 23, 2021.

use for neural ranking), and *aleatoric uncertainty* (noisy data). Given these sources of uncertainty, using point estimate predictions and ranking according to the PRP might not achieve the optimal ranking for retrieval. While the effectiveness benefits of risk-aware models (Wang, 2009; Wang and Zhu, 2009), which take into account the risk[3], i.e. the uncertainty of the document's prediction scores, have been shown for non-neural IR approaches, this has not yet been explored for neural L2R models.

In this paper we first analyze the calibration of neural rankers, specifically BERT-based rankers for IR tasks to evaluate how calibrated they are. Then, to model the uncertainty of BERT-based rankers, we propose *stochastic* neural ranking models (see Figure 1), by applying different techniques to model the uncertainty of DNNs, namely MC Dropout (Gal and Ghahramani, 2016) and Deep Ensembles (Lakshminarayanan et al., 2017) which are agnostic to the particular DNN.

In our experiments, we test models under *distributional shift*, i.e. the test data distribution is different from the training data, also referred to as out-of-distribution (OOD) examples (Lee et al., 2018). In real-world settings, there are often inputs that are shifted due to factors such as non-stationarity and sample bias. Additionally, this experimental setup provides a way of measuring whether the DNN *"knows what it knows"* (Ovadia et al., 2019), e.g. by outputting high uncertainty for OOD examples.

We find that BERT-based rankers are not robustly calibrated. Stochastic BERT-based rankers have 14% less calibration error on average than BERT-based rankers. Uncertainty estimation from stochastic BERT-based rankers is advantageous for downstream applications as shown by our experiments for risk-aware neural ranking (2% more effective on average relative to a model without risk-awareness) and for predicting unanswerable conversational contexts (improves classification by 33% on average of all conditions).

2 Related Work

Calibration and Uncertainty in IR

Even though to optimally rank documents according to the PRP (Robertson, 1977) requires the model to be calibrated (Gordon and Lenk, 1991) (**[C1]**), the calibration of ranking models has received little attention in IR. In contrast, in the machine learning community there have been a number of studies about calibration (Ovadia et al., 2019; Maddox et al., 2019), due to the larger decision making pipelines DNNs are often part of and their importance for model interpretability (Thiagarajan et al., 2020). For instance, in the automated medical domain it is important to provide a calibrated confidence measure besides the prediction of a disease diagnosis to provide clinicians with sufficient information (Jiang et al., 2012). Guo et al. (2017) have shown that DNNs are not well calibrated in the context of computer vision, motivating our study of the calibration of neural L2R models.

The second condition (**[C2]**) for optimal retrieval when ranking according to the PRP (Gordon and Lenk, 1991) is that models report predictions with certainty. While the (un)certainty has not been studied in neural L2R models, there are classic approaches in IR that model the uncertainty. Such approaches have been mostly inspired by economics theory, treating variance as a measure of uncertainty (Varian, 1999). Following such ideas, non-neural ranking models that take uncertainty into account (i.e. risk-aware models), and thus do not follow the PRP (Robertson, 1977), have been proposed (Zhu et al., 2009; Wang and Zhu, 2009), showing significant effectiveness improvements compared to the models that do not model uncertainty. Uncertainty estimation is a difficult task that has other applications in IR besides improving the ranking effectiveness: it can be employed to decide between asking clarifying questions and providing a potential answer in conversational search (Aliannejadi et al., 2019); to perform dynamic query reformulation (Lin et al., 2020) for queries where the intent is uncertain; and to predict questions with no correct answers (Feng et al., 2020).

Bayesian Neural Networks

Unlike standard algorithms to train neural networks, e.g. SGD, that fit point estimate weights given the observed data, Bayesian Neural Networks (BNNs) infer a distribution over the weights given the observed data. Denker et al. (1987) contains one of the earliest mentions of choosing probability over weights of a model. An advantage of the Bayesian treatment of neural networks (MacKay, 1992; Neal, 2012; Blundell et al., 2015) is that they are better at representing existing uncertainties in the training procedure. One limitation of BNNs is that they are computationally expensive compared to DNNs. This has lead to the development of techniques that scale well, and do not require mod-

---

[3] In this paper we use risk and uncertainty interchangeably.

ifications of the neural net architecture and training procedure. Gal and Ghahramani (2016) proposed a way to approximate Bayesian inference by relying on dropout (Srivastava et al., 2014). While dropout is a regularization technique that ignores units with probability p during every training iteration and is disabled at test time, `Dropout` (Gal and Ghahramani, 2016) employs dropout at both train and test time and generates a predictive distribution after a number of forward passes. Lakshminarayanan et al. (2017) proposed an alternative: they employ ensembles of models (`Ensemble`) to obtain a predictive distribution. Ovadia et al. (2019) showed that `Ensemble` are able to produce well calibrated uncertainty estimates that are robust to dataset shift.

Conversational Search

Conversational search is concerned with creating agents that fulfill an information need by means of a *mixed-initiative* conversation through natural language interaction. A popular approach to conversational search is its modeling as an ad-hoc retrieval task: given an ongoing conversation and a large corpus of historic conversations, retrieve the response that is best suited from the corpus (this is also known as conversation response ranking (Wu et al., 2017; Yang et al., 2018; Penha and Hauff, 2020; Gu et al., 2020; Lu et al., 2020)). This retrieval-based approach does not require task-specific knowledge provided by domain experts (Henderson et al., 2019), and it avoids the difficult task of dialogue generation, which often suffers from uninformative, generic responses (Li et al., 2016a) or responses that are incoherent given the dialogue context (Li et al., 2016b). One of the challenges of conversational search is identifying unanswerable questions (Feng et al., 2020), which can trigger for instance clarifying questions (Aliannejadi et al., 2019). Identifying unanswerable conversational contexts is one of the applications we employ uncertainty estimation for. Intuitively, if the system has high uncertainty in the available responses, there may be no correct response available. In this paper we focus on pointwise BERT for ranking—a competitive approach for the conversation response ranking task.

3 Method

In this section we introduce the methods used for answering the following research questions: **RQ1** *How calibrated are deterministic and stochastic BERT-based rankers?* **RQ2** *Are the uncertainty estimates from stochastic BERT-based rankers useful for risk-aware ranking?* **RQ3** *Are the uncertainty estimates obtained from stochastic BERT-based rankers useful for identifying unanswerable queries?* We first describe how to measure the calibration of neural rankers (**[C1]**), followed by our approach for modeling and ranking under uncertainty (**[C2]**), and then we describe how we evaluate their robustness to distributional shift.

3.1 Measuring Calibration

To evaluate the *calibration* of neural rankers (**RQ1**) we resort to the Empirical Calibration Error (ECE) (Naeini et al., 2015). ECE is an intuitive way of measuring to what extent the confidence scores from neural networks align with the true correctness likelihood. It measures the difference between the observed reliability curve (DeGroot and Fienberg, 1983) and the ideal one[4]. More formally, we sort the predictions of the model, divide them into c buckets $\{B_0, ..., B_c\}$, and take the weighted average between the average predicted probability of relevance $avg(B_i)$ and the fraction of relevant[5] documents $\frac{rel(B_i)}{|B_i|}$ in the bucket:

$$ECE = \sum_{i=0}^{c} \frac{|B_i|}{n} \left| avg(B_i) - \frac{rel(B_i)}{|B_i|} \right|,$$

where n is the total number of test examples.

3.2 Modeling Uncertainty

First we define the ranking problem we focus on, followed by the BERT-based ranker baseline model (`BERT`). Having set the foundations, we move to the methods we propose to answer **RQ2** and **RQ3**: a stochastic BERT-based ranker to *model uncertainty* (`S-BERT`) and a risk-aware BERT-based ranker to *take into account uncertainty provided by* `S-BERT` *when ranking* (`RA-BERT`).

3.2.1 Conversation Response Ranking

The task of conversation response ranking (Zhang et al., 2018; Gu et al., 2019; Tao et al., 2019; Henderson et al., 2019; Penha and Hauff, 2020; Yang et al., 2020) (also known as *next utterance selection*), concerns retrieving the best response given the dialogue context. We choose this specific task due to the large-scale training data available, suitable for the training of neural L2R models. Formally, let $\mathcal{D} = \{(\mathcal{U}_i, \mathcal{R}_i, \mathcal{Y}_i)\}_{i=1}^{N}$ be a data set con-

---

[4] See examples of reliability diagrams in Figure 2.

[5] We consider here binary relevance.

sisting of N triplets: dialogue context, response candidates and response relevance labels. The dialogue context $\mathcal{U}_i$ is composed of the previous utterances $\{u^1, u^2, ..., u^\tau\}$ at the turn τ of the dialogue. The candidate responses $\mathcal{R}_i = \{r^1, r^2, ..., r^k\}$ are either ground-truth responses or negative sampled candidates, indicated by the relevance labels $\mathcal{Y}_i = \{y^1, y^2, ..., y^k\}$[6]. The task is then to learn a ranking function $f(.)$ that is able to generate a ranked list for the set of candidate responses $\mathcal{R}_i$ based on their predicted relevance scores $f(\mathcal{U}_i, r)$.

3.2.2 Deterministic `BERT` Ranker

We use BERT for learning the function $f(\mathcal{U}_i, r)$, based on the representation of the `[CLS]` token. The input for BERT is the concatenation of the context $\mathcal{U}_i$ and the response r, separated by SEP tokens. This is the equivalent of early adaptations of BERT for ad-hoc retrieval (Yang et al., 2019) transported to conversation response ranking. Formally the input sentence to BERT is $concat(\mathcal{U}_i, r) = u^1 \mid [U] \mid u^2 \mid [T] \mid ... \mid u^\tau \mid [SEP] \mid r$, where $\mid$ indicates the concatenation operation. The utterances from the context $\mathcal{U}_i$ are concatenated with special separator tokens $[U]$ and $[T]$ indicating end of utterances and turns. The response r is concatenated with the context using BERT's standard sentence separator $[SEP]$. We fine-tune BERT on the target conversational corpus and make predictions as follows: $f(\mathcal{U}_i, r) = \sigma(FFN(BERT_{CLS}(concat(\mathcal{U}_i, r))))$, where $BERT_{CLS}$ is the pooling operation that extracts the representation of the `[CLS]` token from the last layer and FFN is a feed-forward network that outputs logits for two classes (relevant and non-relevant). We pass the logits through a softmax transformation σ that gives us a probability of relevance. We use the cross entropy loss for training. The learned function $f(\mathcal{U}_i, r)$ outputs a point estimate and we refer to it as `BERT`.

3.2.3 Stochastic `S-BERT` Ranker

In order to obtain a predictive distribution, $R_r = \{f(\mathcal{U}_i, r)^0, f(\mathcal{U}_i, r)^1, ..., f(\mathcal{U}_i, r)^n\}$, which allows us to extract uncertainty estimates, we rely on two techniques, namely `Ensemble` (Lakshminarayanan et al., 2017) and `Dropout` (Gal and Ghahramani, 2016). Both techniques scale well and do not require modifications on the architecture or training of BERT.

Using Deep Ensembles ($\texttt{S-BERT}^E$) We train M models using different random seeds without changing the training data, each with its own set of parameters $\{\theta_m\}_{m=1}^M$ and make predictions with each one of them to generate M predicted values: $R_r^E = \{f(\mathcal{U}_i, r)^0, f(\mathcal{U}_i, r)^1, ..., f(\mathcal{U}_i, r)^M\}$. The mean of the predicted values is used as the predicted probability of relevance: $\texttt{S-BERT}^E(\mathcal{U}_i, r) = E[R_r^E]$, and the variance $var[R_r^E]$ gives us a measure of the uncertainty in the prediction.

Using MC Dropout ($\texttt{S-BERT}^D$) We train a single model with parameters θ and employ dropout at test time and generate stochastic predictions of relevance by conducting T forward passes: $R_r^D = \{f(\mathcal{U}_i, r)^0, f(\mathcal{U}_i, r)^1, ..., f(\mathcal{U}_i, r)^T\}$. The mean of the predicted values is used as the predicted probability of relevance: $\texttt{S-BERT}^D(\mathcal{U}_i, r) = E[R_r^D]$, and the variance $var[R_r^D]$ gives us a measure of the uncertainty.

3.2.4 Risk-Aware `RA-BERT` Ranker

Given the predictive distribution R_r, obtained either by `Ensemble` or `Dropout`, we use the following function to rank responses with risk-awareness:

$$\begin{aligned} \texttt{RA-BERT}(\mathcal{U}_i, r) = E[R_r] - b * var[R_r] \\ -2b \sum_i^{n-1} cov[R_r, R_{r_i}], \end{aligned}$$

where $E[R_r]$ is the mean of the predictive distribution, and b is a hyperparameter that controls the aversion or predilection towards risk. Unlike (Zuccon et al., 2011), we are not combining different runs that encompass different model architectures. We instead take a Bayesian interpretation of the process of generating a predictive distribution from a single model architecture. We refer to the rankers as $\texttt{RA-BERT}^D$ and $\texttt{RA-BERT}^E$, when using $\texttt{S-BERT}^D$'s predictive distribution and $\texttt{S-BERT}^E$'s predictive distribution respectively.

3.3 Robustness to Distributional Shift

In order to evaluate whether we can trust the model's calibration and uncertainty estimates, similar to (Ovadia et al., 2019) we evaluate how robust the models are to different types of shift in the

---

[6] Typically, the number of candidates $k \ll K$, where K is the number of available responses and by design the number of ground-truth responses is usually one, the observed response in the conversational data. In our experiments k=10.

test data. We do so by training the model using one setting and applying it in a different setting. Specifically for all three research questions we test the models under two settings—cross domain and cross negative sampling—which we describe next.

3.3.1 Cross Domain

We train a model using the training set from one domain known as the source domain $\mathcal{D}_S$ and evaluate it on the test set of a different domain, known as the target domain $\mathcal{D}_T$. This is also known as the problem of domain generalization (Gulrajani and Lopez-Paz, 2020).

3.3.2 Cross Negative Sampling

Pointwise L2R models are trained on pairs of query and relevant document and pairs of query and non relevant document (Lucchese et al., 2017). Selecting the non-relevant documents requires a *negative sampling* (NS) strategy. For the cross-NS condition, we test models on negative documents that were sampled using a different NS strategy than during training, evaluating the generalization of the models on a shifted distribution of candidate documents. We use three NS strategies. In $\textbf{NS}_{\textbf{random}}$ we randomly select candidate responses from the list of all responses. For $\textbf{NS}_{\textbf{classic}}$ we retrieve candidate responses using the conversational context $\mathcal{U}_i$ as query to a conventional retrieval model and all the responses r as documents. In $\textbf{NS}_{\textbf{sentenceEmb}}$ we represent both $\mathcal{U}_i$ and all the responses with a sentence embedding technique and retrieve candidate responses using a similarity measure.

4 Experimental Setup

We consider three large-scale information-seeking conversation datasets[7] that allow the training of neural ranking models for conversation response ranking: **MSDialog** (Qu et al., 2018) contains 246K context-response pairs, built from 35.5K information seeking conversations from the Microsoft Answer community, a QA forum for several Microsoft products; **MANTiS** (Penha et al., 2019) contains 1.3 million context-response pairs built from conversations of 14 Stack Exchange sites, such as *askubuntu* and *travel*; $\textbf{UDC}_{\textbf{DSTC8}}$ (Kummerfeld et al., 2019) contains 184k context-response pairs of disentangled Ubuntu IRC dialogues.

4.1 Implementation Details

We fine-tune BERT (Devlin et al., 2019) (*bert-base-cased*) for conversation response ranking using the *huggingface-transformers* (Wolf et al., 2019). We follow recent research in IR that employed fine-tuned BERT for retrieval tasks (Nogueira and Cho, 2019; Yang et al., 2019), including conversation response ranking (Penha and Hauff, 2020; Vig and Ramea, 2019; Whang et al., 2019). When training BERT we employ a balanced number of relevant and non-relevant—sampled using BM25 (Robertson and Walker, 1994)—context and response pairs. The sentence embeddings we use for cross-NS is `sentenceBERT` (Reimers and Gurevych, 2019) and we employ dot product calculation from FAISS (Johnson et al., 2017). We consider each dataset as a different domain for cross-NS. We use the Adam optimizer (Kingma and Ba, 2014) with $lr = 5^{-6}$ and $\epsilon = 1^{-8}$, we train with a batch size of 6 and fine-tune the model for 1 epoch. This baseline BERT-based ranker setup yields comparable effectiveness with SOTA methods[8].

4.2 Evaluation

To evaluate the *effectiveness* of the neural rankers we resort to a standard evaluation metric in conversation response ranking (Yuan et al., 2019; Gu et al., 2020; Tao et al., 2019): recall at position K with n candidates[9]: $R_n@K$. To evaluate the *calibration* of the models, we resort to the Empirical Calibration Error (cf. §3.1, using $C = 10$). Throughout, we report the test set results for each dataset. To evaluate the *quality of the uncertainty estimation* we rely on two downstream tasks. The first is to improve conversation response ranking itself via Risk-Aware ranking (cf. §3.2.4). The second, which fits well with conversation response ranking, is to predict unanswerable conversational contexts. Formally the task is to predict whether there is a correct answer in the candidates list $\mathcal{R}$ or not. In our experiments, for half of the instances we remove the relevant response from the list, setting the label as *None Of The Above* (NOTA). The other half of the data has the label *Answerable* (ANSW) indicating that there is a suitable answer

---

[7] `MSDialog` is available at `https://ciir.cs.umass.edu/downloads/msdialog/`; `MANTiS` is available at `https://guzpenha.github.io/MANtIS/`; $\text{UDC}_{\text{DSTC8}}$ is available at `https://github.com/dstc8-track2/NOESIS-II`.

[8] We obtain 0.834 $R_{10}@1$ on $\text{UDC}_{\text{DSTC8}}$ with our baseline BERT model, c.f. Table 1, while `SA-BERT` (Gu et al., 2020) achieves 0.830. The best performing model of the DSTC8 (Kim et al., 2019) also employed a fine-tuned BERT

[9] For example $R_{10}@1$ indicates the number of relevant responses found at the first position when the model has to rank 10 candidate responses.

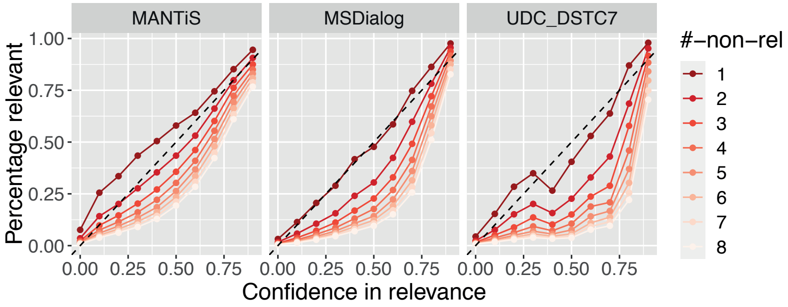


Figure 2: Calibration of `BERT` trained on a balanced set of relevant and non-relevant documents, and tested data with more non-relevant (`#-non-rel`) than relevant (1 per query) documents. A fully calibrated model is represented by the dotted diagonal: for every bucket of confidence in relevance, the % of relevant documents in that bucket is exactly the confidence. The calibration error is the difference between the curves and the diagonal line.

	cross-domain						cross-NS			
Test on →	`MANTiS`		`MSDialog`		`UDC`$_{\texttt{DSTC8}}$		`NS`$_{\texttt{random}}$		`NS`$_{\texttt{sentenceBERT}}$	
Train on ↓ (`NS`$_{\texttt{BM25}}$)	R$_{10}$@1	ECE	R$_{10}$@1	ECE	R$_{10}$@1	ECE	R$_{10}$@1	ECE	R$_{10}$@1	ECE
`MANTiS`	0.615	0.003	0.653	0.010	0.422	0.028	0.263	0.011	0.310	0.009
`MSDialog`	0.398	0.009	0.652	0.006	0.495	0.014	0.298	0.029	0.239	0.027
`UDC`$_{\texttt{DSTC8}}$	0.349	0.016	0.306	0.023	0.834	0.002	0.318	0.050	0.182	0.045

Table 1: Calibration (ECE, lower is better) and effectiveness (R$_{10}$@1, higher is better) of `BERT` for conversation response ranking in cross-domain, and cross-NS conditions. All models were trained using `NS`$_{\texttt{BM25}}$. ECE is calculated using a balanced number of relevant and non relevant documents. Underlined values indicate no distributional shift ($\mathcal{D}_{\mathcal{S}} = \mathcal{D}_{\mathcal{T}}$ and train `NS` = test `NS`).

	cross-domain						cross-NS			
Test on →	`MANTiS`		`MSDialog`		`UDC`$_{\texttt{DSTC8}}$		`NS`$_{\texttt{random}}$		`NS`$_{\texttt{sentenceBERT}}$	
Train on ↓ (`NS`$_{\texttt{BM25}}$)	`S-BERT`^{E}	`S-BERT`^{D}	`S-BERT`^{E}	`S-BERT`^{D}	`S-BERT`^{E}	`S-BERT`^{D}	`S-BERT`^{E}	`S-BERT`^{D}	`S-BERT`^{E}	`S-BERT`^{D}
`MANTiS`	-35.13%†	-56.14%†	-03.42%	-26.89%†	-04.94%	-00.83%	-31.35%	-18.65%†	-37.65%†	-02.79%
`MSDialog`	+25.05%	+08.27%	-43.11%	-11.54%	+22.77%	+05.85%	-15.91%	-10.58%	-17.17%	-12.93%
`UDC`$_{\texttt{DSTC8}}$	-54.95%†	-09.98%†	-25.78%†	-09.15%	+24.77%	-01.84%	-08.05%	-01.78%	-04.81%	-01.28%

Table 2: Relative decreases of ECE (lower is better) of `S-BERT`^{E} and `S-BERT`^{D} over `BERT`. Superscript † denote significant improvements (95% confidence interval) using Student's t-tests.

in the candidates list, for which we remove one of the negative samples instead. Similar to Feng et al. (2020), who proposed to use the outputs (logits) of a LSTM-based model in order to predict NOTA, we use the uncertainties as additional features to the classifier for NOTA prediction. The input space with the additional features is fed to a learning algorithm (Random Forest), and we evaluate it with a 5 fold cross-validation procedure using F1-Macro.

5 Results

5.1 Calibration of Neural Rankers (RQ1)

In order to answer our first research question about the calibration of neural rankers, let us first analyze `BERT` under standard settings (no distributional shift). Our results show that `BERT` is both effective and calibrated under no distributional shift conditions. In Table 1 we see that when the target data (*Test on* →) is the same as the source data (*Train on* ↓)—indicated by underlined values—we obtain the highest effectiveness (on average 0.70 R_{10}@1) and the lowest calibration error (on average 0.036 ECE). When plotting the calibration curves of the model in Figure 2, we observe the curves to be almost diagonal (i.e. having near perfect calibration) when there are an equal number of relevant and non-relevant candidates (`#-non-rel` $= 1$).

However, when we make the conditions more realistic[10] by having multiple non-relevant candidates for each conversational context, we observe in Figure 2 that the calibration errors start to in-

---

[10] In a production system, the retrieval stage would be executed over all candidate responses. As a consequence, the data is highly unbalanced, i.e. only a few relevant responses among potentially millions of non-relevant responses.

	cross-domain						cross-NS			
Test on →	MANTiS		MSDialog		UDC$_{DSTC8}$		NS$_{random}$		NS$_{sentenceBERT}$	
Train on ↓ (NS$_{BM25}$)	RA-BERT^E	RA-BERT^D	RA-BERT^E	RA-BERT^D	RA-BERT^E	RA-BERT^D	RA-B.^{E}	RA-B.^{D}	RA-B.^{E}	RA-B.^{D}
MANTiS	-0.14%	+0.16%†	+0.00%	+0.00%	+0.00%	+0.00%	+4.73%†	+4.58%†	+9.68%†	-2.68%
MSDialog	-2.74%	+0.39%	-1.05%	-0.66%	+5.08%†	-0.10%	-7.61%	+3.29%	-0.61%	+0.63%
UDC$_{DSTC8}$	+0.00%	+0.00%	+0.00%	+0.00%	+0.42%	-0.06%	+6.32%†	+3.83%†	+16.39%†	+17.18%†

Table 3: Relative improvements (higher is better) of $R_{10}@1$ of RA-BERT^E and RA-BERT^D over the mean of stochastic BERT predictions (S-BERT^E and S-BERT^D). Superscript † denote statistically significant improvements over the S-BERT ranker at 95% confidence interval using Student's t-tests.

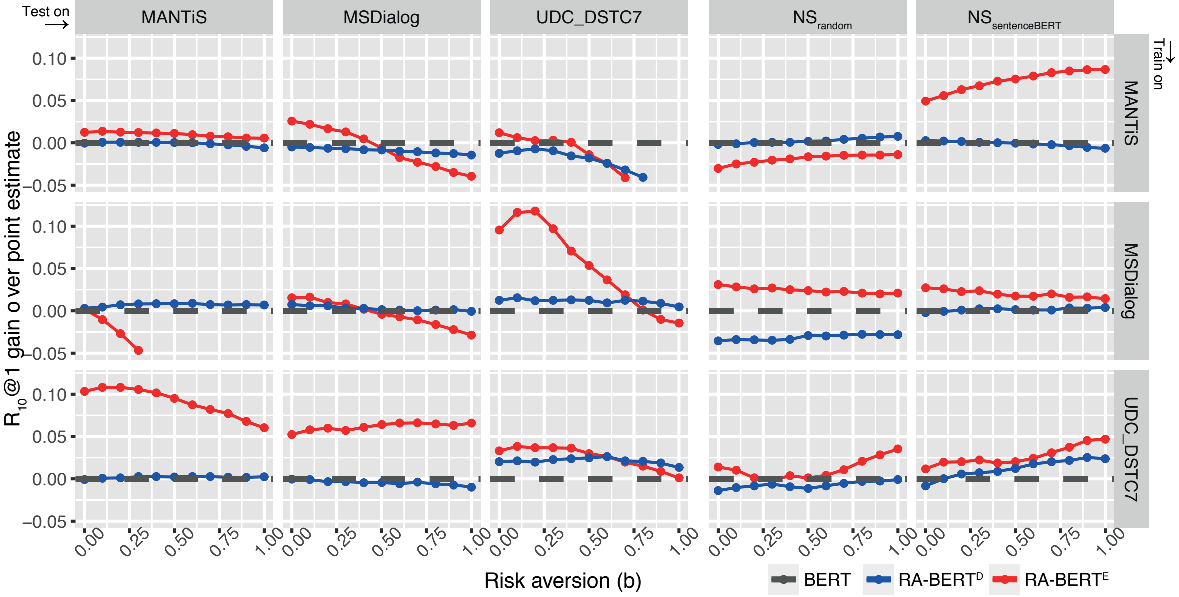


Figure 3: Gains of the Risk-Aware BERT-ranker for different values of risk aversion b.

crease, moving away from the diagonal. Additionally, when we challenge the model in cross-domain and cross-NS settings, the calibration error increases significantly as evident in Table 1. On average, the ECE is **4.6** times higher for cross-domain and **7.9** times higher for cross-NS. Thus **answering the first part of our first research question about the calibration of deterministic BERT-based rankers, indicating that they do not have robust calibrated predictions**, failing on the scenarios where there is a distributional shift.

In order to answer the remaining part of RQ1, on how calibrated *stochastic* BERT-based rankers are, let us consider Table 2. It displays the improvements (relative drop in ECE) over `BERT` in terms of calibration. S-BERT^E is on average **14%** better (has less calibration error) than `BERT`, while S-BERT^D is on average **10%** better than `BERT`, **answering our first research question: stochastic BERT-based rankers are better calibrated than deterministic BERT-based ranker**. We hypothesize that S-BERT^E leads to less ECE than S-BERT^D because it better captures the model uncertainty in the training procedure, since it combines different weights that explain equally well the prediction of relevance given the inputs. In the next section we focus on evaluating the effectiveness of such models that are better calibrated and also taking into account uncertainty when ranking.

5.2 Uncertainty Estimates for Risk-Aware Neural Ranking (RQ2)

In order to evaluate the quality of the uncertainty estimations, we first resort to using them as a measure of the risk through risk-aware neural ranking (RA-BERT^D and RA-BERT^E). Figure 3 displays the effectiveness in terms of $R_{10}@1$ gains over `BERT` for the different settings (cross-domain and cross-NS) when varying the risk aversion b.

We note that when $b = 0$, we are using the mean of the predictive distribution and disregard the risk, which is equivalent to S-BERT^D and S-BERT^E. The ensemble based average S-BERT^E is more effective than the baseline `BERT` for almost all com-

	cross-domain								
Test on →	`MANTiS`			`MSDialog`			UDC$_{\text{DSTC8}}$		
Train on ↓ (NS$_{\text{BM25}}$)	$E[R^D]$	+$var[R^E]$	+$var[R^D]$	$E[R^D]$	+$var[R^E]$	+$var[R^D]$	$E[R^D]$	+$var[R^E]$	+$var[R^D]$
`MANTiS`	0.635 (.02)	0.686 (.01)†	**0.792 (.02)**†	0.669 (.03)	0.731 (.04)	**0.855 (.02)**†	0.571 (.04)	0.590 (.08)†	**0.621 (.04)**†
`MSDialog`	0.561 (.02)	0.598 (.02)†	**0.633 (.02)**†	0.662 (.04)	**0.702 (.01)**†	0.699 (.06)†	0.596 (.04)	0.566 (.06)†	**0.655 (.06)**†
UDC$_{\text{DSTC8}}$	0.527 (.04)	0.665 (.02)†	**0.738 (.03)**†	0.523 (.05)	0.691 (.03)†	**0.757 (.04)**†	0.787 (.01)	**0.829 (.03)**†	0.807 (.01)†

Table 4: Results of the *cross-domain* condition for the NOTA prediction task, using a Random Forest classifier and different input spaces. The F1-Macro and standard deviation over the 5 folds of the cross validation are displayed. Superscript † denote statistically significant improvements over $E[R^D]$ at 95% confidence interval using Student's t-tests. Bold indicates the most effective approach.

	cross-NS					
Test on →	NS$_{\text{random}}$			NS$_{\text{sentenceBERT}}$		
Train on ↓ (NS$_{\text{BM25}}$)	$E[R^D]$	+$var[R^E]$	+$var[R^D]$	$E[R^D]$	+$var[R^E]$	+$var[R^D]$
`MANTiS`	0.557 (.01)	0.604 (.02)†	**0.698 (.02)**†	0.534 (.03)	0.587 (.02)†	**0.647 (.05)**†
`MSDialog`	0.505 (.02)	0.606 (.02)†	**0.702 (.05)**†	0.522 (.03)	0.611 (.07)†	**0.653 (.04)**†
UDC$_{\text{DSTC8}}$	0.565 (.03)	0.800 (.02)†	**0.942 (.04)**†	0.506 (.05)	0.755 (.05)†	**0.821 (.05)**†

Table 5: Results of the *cross-NS* condition for the NOTA prediction task.

binations and `S-BERT`^{D} is equivalent to the baseline. When using $b < 0$, we are ranking with risk predilection (the opposite of risk aversion), and in all conditions we found that the effectiveness was significantly worse than when $b = 0$ and thus $b < 0$ is not displayed in Figure 3.

When increasing the risk aversion ($b > 0$), we see that it has different effects depending on the combination of domain and NS. For instance, when training in `MSDialog` and applying on UDC$_{\text{DSTC8}}$, increasing the risk aversion improves effectiveness of `RA-BERT`^{E} until b reaches 0.25 and after that the effectiveness drops, meaning that too much risk aversion is not effective. In order to investigate whether ranking with risk aversion is more effective than using the predictive distribution mean, we select b based on the best value observed on the validation set. Table 3 displays the results of this experiment, showing the improvements of `RA-BERT`^{D} and `RA-BERT`^{E} over `S-BERT`^{D} and `S-BERT`^{E} respectively. The results show that in a few cases (8 out of 30) the best value of b is 0, for which risk-aversion is not the best option in the development set. We obtain effectiveness improvements primarily on the cross-NS condition (up to 17.2% improvement of $R_{10}@1$), which is the hardest condition (when the models are most ineffective, c.f. Table 1). **This answers our second research question, indicating that the uncertainties obtained from stochastic neural rankers are useful for risk-aware ranking, specially in the cross-NS setting where the baseline model is quite ineffective.** `RA-BERT`^{E} is on average 2% more effective than `S-BERT`^{E}, while `RA-BERT`^{D} is on average 1.7% more effective than `S-BERT`^{D}.

5.3 Uncertainty Estimates for NOTA prediction (RQ3)

Besides using the uncertainty estimation for risk-aware ranking, we also employ it for the NOTA (None of the Above) prediction task. We compare here different input spaces for the NOTA classifier. $E[R^D]$ stands for the input space that only uses the mean of the predictive distribution for the k candidate responses in $\mathcal{R}$ using `S-BERT`^{D}, +$var[R^E]$ uses both $E[R^D]$ and the uncertainties of `S-BERT`^{E} for the k candidates and +$var[R^D]$ uses both the scores $E[R^D]$ and the uncertainties of `S-BERT`^{D}. Our results show that the uncertainties from `S-BERT`^{D} and of `S-BERT`^{E} significantly improve the F1 for NOTA prediction for both cross-domain (Table 4, improvement of 24% on average when using `S-BERT`^{D}) and cross-NS settings (Table 5, improvement of 46% on average when using `S-BERT`^{D}). **We can thus answer our last research question: the uncertainty estimates from stochastic neural rankers do improve the effectiveness of the NOTA prediction task (by an average of 33% across all conditions considered).**

6 Conclusions

In this work we study the calibration and uncertainty estimation of neural rankers, specifically BERT-based rankers. We first show that the deterministic BERT-based ranker is not robustly calibrated for the task of conversation response ranking and we improve its calibration with two techniques to estimate uncertainty through *stochastic neural ranking*. We also show the benefits of estimating uncertainty using risk-aware neural ranking and for predicting unanswerable conversational contexts.

As future work, investigating the use of stochastic rankers in other settings is important, such as other neural L2R architectures, other search and retrieval tasks (Guo et al., 2019; Diaz et al., 2020; Lin et al., 2020), and the ensembling of neural rankers (Zuccon et al., 2011).

Acknowledgements

This research has been supported by NWO projects SearchX (639.022.722) and NWO Aspasia (015.013.027).

References

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 475–484.

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*, volume 463. ACM press New York.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1613–1622.

Morris H DeGroot and Stephen E Fienberg. 1983. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 32(1-2):12–22.

John S. Denker, Daniel B. Schwartz, Ben S. Wittner, Sara A. Solla, Richard E. Howard, Lawrence D. Jackel, and John J. Hopfield. 1987. Large automatic learning, rule extraction, and generalization. *Complex Systems*, 1.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, pages 4171–4186.

Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. 2020. Evaluating stochastic rankings with expected exposure.

Yulan Feng, Shikib Mehri, Maxine Eskenazi, and Tiancheng Zhao. 2020. "none of the above": Measure uncertainty in dialog response retrieval. In *ACL*, pages 2013–2020, Online. Association for Computational Linguistics.

Y Gal and Z Ghahramani. 2016. Dropout as a bayesian approximation. In *33rd International Conference on Machine Learning, ICML 2016*, volume 3, pages 1661–1680.

Yarin Gal. 2016. Uncertainty in deep learning. *University of Cambridge*, 1(3).

Michael D Gordon and Peter Lenk. 1991. A utility theoretic examination of the probability ranking principle in information retrieval. *Journal of the American Society for Information Science*, 42(10):703–714.

Jia-Chen Gu, Tianda Li, Quan Liu, Xiaodan Zhu, Zhen-Hua Ling, Zhiming Su, and Si Wei. 2020. Speaker-aware bert for multi-turn response selection in retrieval-based chatbots. *arXiv preprint arXiv:2004.03588*.

Jia-Chen Gu, Zhen-Hua Ling, and Quan Liu. 2019. Utterance-to-utterance interactive matching network for multi-turn response selection in retrieval-based chatbots. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:369–379.

Ishaan Gulrajani and David Lopez-Paz. 2020. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*.

Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2019. A deep look into neural ranking models for information retrieval. *Information Processing & Management*, page 102067.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Ivan Vulić, et al. 2019. Convert: Efficient and accurate conversational representations from transformers. *arXiv preprint arXiv:1911.03688*.

Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. 2012. Calibrating predictive model estimates to support personalized medicine. *Journal of the American Medical Informatics Association*, 19(2):263–274.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Seokhwan Kim, Michel Galley, Chulaka Gunasekara, Sungjin Lee, Adam Atkinson, Baolin Peng, Hannes Schulz, Jianfeng Gao, Jinchao Li, Mahmoud Adada, Minlie Huang, Luis Lastras, Jonathan K. Kummerfeld, Walter S. Lasecki, Chiori Hori, Anoop Cherian, Tim K. Marks, Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, and Raghav Gupta. 2019. The eighth dialog system technology challenge.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jonathan K. Kummerfeld, Sai R. Gouravajhala, Joseph J. Peper, Vignesh Athreya, Chulaka Gunasekara, Jatin Ganhotra, Siva Sankalp Patel, Lazaros C Polymenakos, and Walter Lasecki. 2019. A large-scale corpus for conversation disentanglement. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *NAACL*, pages 110–119.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016b. A persona-based neural conversation model. In *ACL*, pages 994–1003.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020. Query reformulation using query history for passage retrieval in conversational search.

Junyu Lu, Xiancong Ren, Yazhou Ren, Ao Liu, and Zenglin Xu. 2020. Improving contextual language models for response retrieval in multi-turn conversation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1805–1808.

Claudio Lucchese, Franco Maria Nardini, Raffaele Perego, and Salvatore Trani. 2017. The impact of negative samples on learning to rank. In *LEARNER@ ICTIR*.

David JC MacKay. 1992. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. 2019. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164.

Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence*, volume 2015, page 2901. NIH Public Access.

Radford M Neal. 2012. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13991–14002.

Gustavo Penha, Alexandru Balan, and Claudia Hauff. 2019. Introducing MANtIS: a novel Multi-Domain Information Seeking Dialogues Dataset. *arXiv preprint arXiv:1912.04639*.

Gustavo Penha and Claudia Hauff. 2020. Curriculum learning strategies for ir. In *European Conference on Information Retrieval*, pages 699–713. Springer.

Chen Qu, Liu Yang, W Bruce Croft, Johanne R Trippas, Yongfeng Zhang, and Minghui Qiu. 2018. Analyzing and characterizing user intent in information-seeking conversations. In *SIGIR*, pages 989–992.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983.

Stephen E Robertson. 1977. The probability ranking principle in ir. *Journal of documentation*.

Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*, pages 232–241. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Chongyang Tao, Wei Wu, Can Xu, Wenpeng Hu, Dongyan Zhao, and Rui Yan. 2019. Multi-representation fusion network for multi-turn response selection in retrieval-based chatbots. In *WSDM*, pages 267–275.

Jayaraman J Thiagarajan, Prasanna Sattigeri, Deepta Rajan, and Bindya Venkatesh. 2020. Calibrating healthcare ai: Towards reliable and interpretable deep predictive models. *arXiv preprint arXiv:2004.14480*.

Hal R Varian. 1999. Economics and search. In *ACM SIGIR Forum*, volume 33, pages 1–5. ACM New York, NY, USA.

Jesse Vig and Kalai Ramea. 2019. Comparison of transfer-learning approaches for response selection in multi-turn conversations. In *Workshop on DSTC7*.

Jun Wang. 2009. Mean-variance analysis: A new document ranking theory in information retrieval. In *European Conference on Information Retrieval*, pages 4–16. Springer.

Jun Wang and Jianhan Zhu. 2009. Portfolio theory of information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 115–122.

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and HeuiSeok Lim. 2019. Domain adaptive training bert for response selection. *arXiv preprint arXiv:1908.04812*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *ACL*, pages 496–505.

Liu Yang, Minghui Qiu, Chen Qu, Cen Chen, Jiafeng Guo, Yongfeng Zhang, W Bruce Croft, and Haiqing Chen. 2020. Iart: Intent-aware response ranking with transformers in information-seeking conversation systems. *arXiv preprint arXiv:2002.00571*.

Liu Yang, Minghui Qiu, Chen Qu, Jiafeng Guo, Yongfeng Zhang, W Bruce Croft, Jun Huang, and Haiqing Chen. 2018. Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. In *SIGIR*, pages 245–254.

Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple applications of bert for ad hoc document retrieval.

Chunyuan Yuan, Wei Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In *EMNLP*, pages 111–120.

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *ACL*, pages 3740–3752.

Jianhan Zhu, Jun Wang, Ingemar J Cox, and Michael J Taylor. 2009. Risky business: modeling and exploiting uncertainty in information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 99–106.

Guido Zuccon, Leif Azzopardi, and Keith van Rijsbergen. 2011. Back to the roots: mean-variance analysis of relevance estimations. In *European Conference on Information Retrieval*, pages 716–720. Springer.

Frequency-Guided Word Substitutions for Detecting Textual Adversarial Examples

Maximilian Mozes[1] **Pontus Stenetorp**[1] **Bennett Kleinberg**[2,1] **Lewis D. Griffin**[1]
[1]University College London
[2]Tilburg University
{m.mozes, p.stenetorp, l.griffin}@cs.ucl.ac.uk
bennett.kleinberg@tilburguniversity.edu

Abstract

Recent efforts have shown that neural text processing models are vulnerable to adversarial examples, but the nature of these examples is poorly understood. In this work, we show that adversarial attacks against CNN, LSTM and Transformer-based classification models perform word substitutions that are identifiable through frequency differences between replaced words and their corresponding substitutions. Based on these findings, we propose *frequency-guided word substitutions* (FGWS), a simple algorithm exploiting the frequency properties of adversarial word substitutions for the detection of adversarial examples. FGWS achieves strong performance by accurately detecting adversarial examples on the SST-2 and IMDb sentiment datasets, with F_1 detection scores of up to 91.4% against RoBERTa-based classification models. We compare our approach against a recently proposed perturbation discrimination framework and show that we outperform it by up to 13.0% F_1.

Attack	**Original or perturbed sequence**
None	A clever blend of fact and fiction
GENETIC	1.39 ⇠ 5.55 A **brainy** [*clever*] blend of fact and fiction
PWWS	1.61 ⇠ 5.55 0.00 ⇠ 3.81 A **cunning** [*clever*] **blending** [*blend*] of 0.00 ⇠ 4.39 fact and **fabrication** [*fiction*]

Figure 1: Corpus $\log_e$ frequencies of the replaced words (bold, italic, red) and their corresponding adversarial substitutions (bold, black) using the GENETIC (Alzantot et al., 2018) and PWWS (Ren et al., 2019) attacks on SST-2 (Socher et al., 2013).

1 Introduction

Artificial neural networks are vulnerable to adversarial examples—carefully crafted perturbations of input data that lead a learning model into making false predictions (Szegedy et al., 2014).

While initially discovered for computer vision tasks, natural language processing (NLP) models have also been shown to be oversensitive to adversarial input perturbations for a variety of tasks (Papernot et al., 2016; Jia and Liang, 2017; Belinkov and Bisk, 2018; Glockner et al., 2018; Iyyer et al., 2018). Here we focus on highly successful synonym substitution attacks (Alzantot et al., 2018; Ren et al., 2019; Zang et al., 2020), in which individual words are replaced with semantically similar ones. Existing defense methods against these attacks mainly focus on adversarial training (Jia and Liang, 2017; Ebrahimi et al., 2018; Ribeiro et al., 2018; Ren et al., 2019; Jin et al., 2019) and hence typically require a priori attack knowledge and models to be retrained from scratch to increase their robustness. Recent work by Zhou et al. (2019) instead proposes DISP (*learning to discriminate perturbations*), a perturbation discrimination framework that exploits pre-trained contextualized word representations to detect and correct word-level adversarial substitutions without having to retrain the attacked model. In this paper, we show that we can achieve an improved performance for the detection and correction of adversarial examples based on the finding that various word-level adversarial attacks have a tendency to replace input words with less frequent ones.[1] Figure 1 illustrates this tendency for two state-of-the-art attacks. We provide statistical evidence to support this observation and propose a rule-based and model-agnostic algorithm, *frequency-guided word substitutions* (FGWS), to detect adversarial sequences

---

[1] This frequency difference is expected for attacks that explicitly conduct symbol substitutions resulting in out-of-vocabulary (OOV) terms (Gao et al., 2018). We therefore study attacks that do not explicitly enforce a mapping to words that have lower frequencies.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 171–186
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

and recover model performances for perturbed test set sequences. FGWS effectively detects adversarial perturbations, achieving F_1 scores of up to 91.4% against RoBERTa-based models (Liu et al., 2019) on the IMDb sentiment dataset (Maas et al., 2011). Furthermore, our results show that FGWS outperforms DISP by up to 13.0% F_1 when differentiating between unperturbed and perturbed sequences, despite representing a conceptually simpler approach to this task.

2 Generating adversarial examples

In our experiments, we investigate two baseline attacks introduced by Ren et al. (2019) as well as two state-of-the-art attacks.

RANDOM. Our first baseline attack is a simple word substitution model that randomly selects words in an input sequence and replaces them with synonyms randomly sampled from a set of synonyms related to the specific word. We follow Ren et al. (2019) by using WORDNET (Fellbaum, 1998) to identify synonym substitutions for each selected word.

PRIORITIZED. Our second baseline builds upon RANDOM by selecting the replacement word from the synonym set that maximizes the change in prediction confidence for the true label of an input.

GENETIC. We additionally analyze an attack suggested by Alzantot et al. (2018), consisting of a population-based black-box mechanism based on genetic search that iteratively performs individual word-level perturbations to an input sequence to cause a misclassification.

PWWS. Lastly, we analyze the *probability weighted word saliency* (PWWS) algorithm (Ren et al., 2019). For each word in an input sequence, PWWS selects a set of synonym replacements from WORDNET and chooses the synonym yielding the highest difference in prediction confidence for the true class label after replacement. The algorithm furthermore computes the word saliency (Li et al., 2016a,b) for each input word and ranks word replacements based on these two indicators.

Datasets and models. We perform experiments on two binary sentiment classification datasets, the *Stanford Sentiment Treebank* (SST-2, Socher et al., 2013) and the IMDb reviews dataset (Maas et al., 2011), both of which are widely used in related work focusing on adversarial examples in NLP (Jia et al., 2019; Ren et al., 2019; Zhou et al., 2019). Dataset details can be found in Appendix A. Adhering to Zhou et al. (2019), we attack a pre-trained model based on the Transformer architecture (Vaswani et al., 2017). Zhou et al. (2019) use BERT (Devlin et al., 2019) in their experiments, but we found that RoBERTa (Liu et al., 2019) represents a stronger model for the specified tasks.

We additionally experiment with both a CNN (Kim, 2014) and an LSTM (Hochreiter and Schmidhuber, 1997) text classification model, both of which have been employed in existing work studying textual adversarial attacks (Alzantot et al., 2018; Lei et al., 2019; Jia et al., 2019; Tsai et al., 2019; Ren et al., 2019).

The fine-tuned RoBERTa model achieves 93.4% and 94.9% accuracy on the IMDb and SST-2 test sets, which is comparable to existing work (Beltagy et al., 2020; Liu et al., 2019). On the IMDb test set, the CNN achieves an accuracy of 86.0% and the LSTM achieves 83.1%. These performances are close to existing work using comparable settings (Zhang et al., 2019; Ren et al., 2019). On the SST-2 test set, the CNN achieves 84.0% and the LSTM 85.2% accuracy, which are also close to comparable experiments (Huang et al., 2019).

Following Ren et al. (2019), we apply all four attacks to a random subset of 2,000 sequences from the IMDb test set as well as the entire test set of SST-2 (1,821 samples). Implementation details for the models and attacks can be found in Appendix B. We report the after-attack accuracies[2] for the RoBERTa model in Table 2 and for the CNN/LSTM models in Table 3 (column Adv.). We observe that all four attacks cause notable decreases in model accuracy on the test sets, and that GENETIC and PWWS are more successful than the baseline attacks in most comparisons.

3 Analyzing frequencies of adversarial word substitutions

Next, we conduct an analysis of the word frequencies of individual words replaced by the attacks and their substitutions. We compute the $\log_e$ training set frequencies $\phi(x)$ of all words x that have been replaced by the respective attacks and all of their corresponding substitutions. Then, we conduct Bayesian hypothesis testing (Rouder et al., 2009) to statistically compare the two samples. This is achieved by computing the Bayes factor BF_{10}, rep-

---

[2] The after-attack accuracy represents the model accuracy on the test set after perturbing all correctly classified inputs. A lower after-attack accuracy indicates a stronger attack.

Dataset	Attack	Replaced		Subst.			non-OOV		
		μ_ϕ	σ_ϕ	μ_ϕ	σ_ϕ	d	μ_ϕ	σ_ϕ	d
IMDb	Random	7.6	2.5	3.4	2.8	**1.6**	4.4	2.4	1.3
	Prioritized	7.6	2.5	3.6	2.8	1.5	4.4	2.4	1.3
	Genetic	6.5	2.0	3.7	2.3	1.3	4.0	2.2	1.2
	PWWS	6.9	2.3	4.4	2.5	1.0	5.0	2.1	0.9
SST-2	Random	5.4	2.6	2.1	2.4	**1.4**	4.0	1.8	0.6
	Prioritized	5.4	2.6	2.1	2.4	1.3	4.0	1.8	0.6
	Genetic	4.4	1.9	1.9	2.2	1.2	3.6	1.6	0.4
	PWWS	4.8	2.1	2.9	2.2	0.9	4.0	1.5	0.4

Table 1: Mean $\log_e$ frequencies of replaced words and their substitutions. Values in bold denote largest effect sizes per dataset.

resenting the degree to which the data favor the alternative hypothesis over the null hypothesis. Here, the alternative hypothesis $\mathcal{H}_1$ states that *the frequencies of replaced words differ from the frequencies of the adversarial substitutions*. The null hypothesis $\mathcal{H}_0$ states that *there is no such difference*. The higher BF_{10}, the stronger the evidence in favor of the alternative hypothesis $\mathcal{H}_1$.[3] We additionally calculate Cohen's d effect sizes for all mean frequency comparisons.[4]

Table 1 shows the $\log_e$ frequencies (mean μ_ϕ and standard deviation σ_ϕ) and Cohen's d for the specified samples generated by the attacks against the RoBERTa model (the results for the CNN and LSTM models can be found in Appendix C). We report the mean frequencies of all adversarial substitutions (Subst.) and only those that occur in the training set (non-OOV), to demonstrate that the frequency differences are not solely caused by OOV substitutions. Across datasets and attacks, the substitutions are consistently less frequent than the words selected for replacement. We observe large Cohen's d effect sizes for the majority of comparisons, statistically supporting the observation of mean frequency differences between replaced words and their corresponding substitutions. We furthermore observe that $\mathrm{BF}_{10} > 10^{55}$ holds for all comparisons—both when considering all and only non-OOV substitutions (the BF_{10} scores can be found in Appendix D). This provides strong empirical evidence that $\mathcal{H}_1$ is more likely to be supported by the measured word frequencies (see Appendix E for additional illustrations).

4 Frequency-guided word substitutions

Based on the observation of consistent frequency differences between replaced words and adversarial substitutions, we argue that the effects of such substitutions can be mitigated through simple frequency-based transformations. To do this, we propose *frequency-guided word substitutions* (FGWS), a detection method that estimates whether a given input sequence is an adversarial example.[5] We denote a classification model by a function $f(X)$ that maps a sequence X to a C-dimensional vector representing the probabilities for predicting each of the C possible classes. We represent a sequence as $X = \{x_1, \ldots, x_n\}$, where x_i denotes the i-th word in the sequence. We furthermore introduce the notation $f^*(X) \in \{1, \ldots, C\}$ representing the class label predicted by f given input X. FGWS transforms a given sequence X into a sequence X' by replacing infrequent words with more frequent, semantically similar substitutions. We initially define the subset $X_E := \{x \in X \mid \phi(x) < \delta\}$ of words that are eligible for substitution, where $\delta \in \mathbb{R}_{>0}$ is a frequency threshold. FGWS then generates a sequence X' from X by replacing all eligible words with words that are semantically similar, but have higher occurrence frequencies in the model's training corpus. For each eligible word $x \in X_E$ we consider the set of replacement candidates $\mathcal{S}(x)$ and find a replacement x' by selecting $x' = \mathrm{argmax}_{w \in \mathcal{S}(x)}\ \phi(w)$. We then generate X' by replacing each eligible word x with x' if $\phi(x') > \phi(x)$. Given the prediction label $y = f^*(X)$ for X and a threshold $\gamma \in [0, 1]$, the sequence X is considered adversarial if $f(X)_y - f(X')_y > \gamma$, i.e., if the difference in prediction confidence on class y before and after transformation exceeds the threshold γ. The threshold allows control of the rate of false positives (i.e., unperturbed sequences that are erroneously identified as adversarial) flagged by our method.

4.1 Comparisons

DISP. We compare FGWS to the DISP framework (Zhou et al., 2019), which is, to the best of our knowledge, the best existing approach for the detection of word-level adversarial examples. DISP uses two independent BERT-based components, a perturbation discriminator and an embedding estimator for token recovery, to identify perturbed

---

[3] A Bayes factor $\mathrm{BF}_{10} > 100$ can be interpreted as "extreme" evidence for $\mathcal{H}_1$ (Wagenmakers et al., 2011).

[4] Cohen's d indicates the magnitude of the frequency differences of the two samples—larger effect sizes suggest a higher magnitude of the frequency difference. A value of $d = 0.8$ can be interpreted as a large effect, $d = 0.5$ is considered a moderate effect (Cohen, 1988).

[5] Code is available at `https://github.com/maximilianmozes/fgws`.

Dataset	Attack	Adv.	Restored acc.		TPR (FPR)		F_1	
			DISP	FGWS	DISP	FGWS	DISP	FGWS
IMDb	Random	87.3	89.2	**91.0**	63.6 (9.4)	**83.5** (9.3)	73.6	**86.6**
	Prioritized	41.5	81.0	**85.9**	87.8 (9.4)	**92.0** (9.3)	89.0	**91.4**
	Genetic	47.7	74.1	**80.6**	70.4 (9.4)	**81.5** (9.3)	78.3	**85.4**
	PWWS	41.0	68.7	**75.4**	66.2 (9.4)	**76.4** (9.3)	75.4	**82.3**
SST-2	Random	87.2	86.6	**90.0**	**66.2** (11.9)	61.3 (11.4)	**74.4**	71.0
	Prioritized	68.9	80.8	**84.8**	69.1 (11.9)	**74.7** (11.4)	76.3	**80.3**
	Genetic	40.8	60.1	**61.7**	**57.2** (11.9)	57.0 (11.4)	**67.7**	**67.7**
	PWWS	57.4	71.0	**78.2**	59.6 (11.9)	**65.6** (11.4)	69.6	**74.2**

Table 2: Adversarial example detection performances for DISP and FGWS when evaluated on attacks against RoBERTa. Adv. shows the model's classification accuracy on the perturbed sequences. Restored acc. denotes model accuracy on the adversarial sequences after transformation. Values in bold represent best scores per metric, dataset and attack.

Model/ Dataset	Attack	Adv.	Restored acc.		F_1	
			NWS	FGWS	NWS	FGWS
CNN/ IMDb	Random	73.0	79.5	**84.7**	75.2	**83.5**
	Prioritized	14.0	41.6	**78.9**	71.5	**89.3**
	Genetic	10.7	21.3	**68.5**	37.9	**83.5**
	PWWS	10.2	27.4	**70.2**	45.4	**83.9**
LSTM/ IMDb	Random	64.7	75.7	**80.9**	80.5	**83.9**
	Prioritized	3.2	32.0	**71.6**	62.4	**86.6**
	Genetic	1.2	10.9	**54.9**	34.3	**78.0**
	PWWS	1.6	17.3	**57.1**	41.7	**77.4**
CNN/ SST-2	Random	71.8	77.1	**78.4**	**71.4**	69.2
	Prioritized	50.3	60.1	**69.3**	54.8	**67.8**
	Genetic	19.6	34.9	**48.8**	49.9	**60.3**
	PWWS	28.1	47.4	**58.1**	55.1	**63.9**
LSTM/ SST-2	Random	73.4	79.3	**80.5**	**69.2**	62.2
	Prioritized	48.5	59.9	**74.0**	54.9	**67.3**
	Genetic	21.3	37.6	**61.1**	51.2	**62.8**
	PWWS	28.6	49.7	**67.2**	55.9	**63.4**

Table 3: Performance results of NWS and FGWS on attacks against the CNN and LSTM models. Values in bold indicate best performances per model-dataset-attack combination and metric.

tokens and to reconstruct the replaced ones.

NWS. For the CNN and LSTM models, we compare FGWS with the *naive word substitutions* (NWS) baseline. For a given input sequence, NWS selects all OOV words in that sequence and replaces each with a random choice from a set of semantically related words. We restrict NWS to allow only substitutions for which the replacement word occurs in the model's training vocabulary. NWS can be interpreted as a variant of FGWS that is not explicitly guided by word frequencies.

4.2 Experiments

We apply both methods to the adversarial examples crafted by the four attacks on the subsets of both the IMDb and SST-2 datasets as described in Section 2. To account for an imbalance between unperturbed and perturbed sequences, we repeatedly bootstrap a balanced set of unperturbed sequences for each set of perturbed sequences for 10,000 times and compute the average detection scores. For FGWS, we tune the frequency threshold δ for each model-dataset combination on the validation set. To do this, we utilize the Prioritized attack to craft adversarial examples from all sequences of the validation set[6] and compare FGWS detection performances with different values for δ. Specifically, we set δ equal to the $\log_e$ frequency representing the q^{th} percentile of all $\log_e$ frequencies observed by the words eligible for replacement in the training set, and experiment with $q \in \{0, 10, \ldots, 100\}$. We select γ so that not more than 10% of the unperturbed sequences in the validation set are labeled as adversarial.[7] For FGWS, we define the set of replacement candidates for each word $x \in X_E$ as the union of the word's K nearest neighbors in a pre-trained GloVe (Pennington et al., 2014) word embedding space and its synonyms in WordNet. We set K equal to the average number of WordNet synonyms for each word in the validation set (yielding $K = 6$ for IMDb and $K = 8$ for SST-2).

4.3 Results

We report the results comparing FGWS to DISP on attacks against RoBERTa in Table 2. Here, the true positive rate (TPR) represents the percentage of successful adversarial examples that were cor-

---

[6] We assume both baseline attacks as given to the defender, and prefer Prioritized over Random due to increased effectiveness and hence a larger sample size for parameter tuning.

[7] We provide additional results with varying false positive thresholds in Appendix F.

Unperturbed		a smart sweet and playful romantic comedy	*positive* (99.9%)
(A)	PWWS	a $\overset{0.00}{\textbf{impertinent}}$ [$\overset{5.69}{\textit{smart}}$] $\overset{1.79}{\textbf{odoriferous}}$ [$\overset{5.77}{\textit{sweet}}$] and playful romantic comedy	*negative* (56.3%)
(D)	DISP	$\overset{10.22}{\textbf{the}}$ [$\overset{9.99}{\textit{a}}$] $\overset{6.83}{\textbf{little}}$ [$\overset{0.00}{\textit{impertinent}}$] odoriferous and playful romantic comedy	*positive* (79.3%)
(D)	FGWS	a $\overset{5.69}{\textbf{smart}}$ [$\overset{0.00}{\textit{impertinent}}$] $\overset{5.77}{\textbf{sweet}}$ [$\overset{1.79}{\textit{odoriferous}}$] and playful romantic comedy	*positive* (99.9%)

Figure 2: The detection methods applied to an adversarial example from the PWWS attack against RoBERTa on SST-2. The words highlighted in bold, italic and red were selected for replacement by the attack (A) and the detection methods (D), the ones in bold and black denote the substitutions. The values above the words denote their $\log_e$ frequencies.

rectly identified as such, and the false positive rate (FPR) denotes the percentage of unperturbed sequences that were identified as adversarial. The column Adv. gives the classification accuracy on the perturbed sequences, and Restored acc. the model's accuracy on the adversarial sequences after transformation. We observe that FGWS best restores the model's classification accuracy across all comparisons, showing it to be effective in mitigating the effects of the individual attacks. Furthermore, FGWS outperforms DISP in terms of true positive rates and F_1 across the majority of experiments. These results show that, although contextualized word representations (DISP) serve as a competitive method to detect adversarial examples, relying solely on frequency-guided substitutions (FGWS) shows to be more effective. Figure 2 provides an example adversarial sequence generated with the PWWS attack and the two corresponding transformed sequences using DISP and FGWS (see Appendix G for additional examples).

The results of NWS and FGWS against the CNN and LSTM models are shown in Table 3. We observe that FGWS outperforms NWS across all comparisons in terms of restored model accuracy and in the majority of comparisons in terms of F_1. Moreover, the direct comparison between NWS and FGWS again underlines the importance of utilizing word frequencies as guidance for the word substitutions: while NWS is not guided by word frequency characteristics to perform the word replacements, we observe that FGWS outperforms NWS by a large margin in most comparisons, demonstrating the effectiveness of mapping infrequent words to their most frequent semantically similar counterparts to detect adversarial examples.

4.4 FGWS on unperturbed data

We furthermore investigate the effect of FGWS on model performance on unperturbed sequences after transformation. To do this, we transform the sampled test sets using FGWS and evaluate classification accuracies after sequence transformation. The differences in accuracy for the CNN, LSTM and RoBERTa models before and after transformation are 0.0%, $+1.0\%$ and -0.2% for IMDb and -1.8%, -2.9% and -1.8% for SST-2. This indicates that FGWS applied to unperturbed data has only small effects on classification accuracy, and in some cases even slightly increases prediction accuracy.

5 Limitations

It is worth mentioning that compared to FGWS, DISP represents a more general perturbation discrimination approach since it is trained to detect both character- and word-level adversarial perturbations, whereas FGWS solely focuses on word-level attacks.

Furthermore, it remains open whether FGWS would be effective against attacks for which the frequency difference is less evident. To investigate this, we conducted preliminary experiments by restricting the investigated attacks to only allow equifrequent substitutions. However, we observed that introducing this constraint has a substantial effect on attack performance, since the attacks are supplied with fewer candidate replacements. We will further investigate this in future work.

6 Conclusion

We have shown that the word frequency characteristics of adversarial word substitutions can be leveraged effectively to detect adversarial sequences for neural text classification. Our proposed approach outperforms existing detection methods despite representing a conceptually simpler approach to this task.

Acknowledgements

This research was supported by the Dawes Centre for Future Crime at University College London.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. Technical report, Google.

Jacob Cohen. 1988. *Statistical power analysis for the behavioral sciences*. Academic press.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. 2019. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4081–4091, Hong Kong, China. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4120–4133, Hong Kong, China. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Qi Lei, Lingfei Wu, Pin-Yu Chen, Alexandros G Dimakis, Inderjit S Dhillon, and Michael Witbrock. 2019. Discrete adversarial attacks and submodular optimization with applications to text classification. In *SysML 2019*.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016a. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association* *for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016b. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Nicolas Papernot, Patrick Drew McDaniel, Ananthram Swami, and Richard Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Proceedings - IEEE Military Communications Conference MILCOM, pages 49–54, United States. Institute of Electrical and Electronics Engineers Inc.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Jeffrey N Rouder, Paul L Speckman, Dongchu Sun, Richard D Morey, and Geoffrey Iverson. 2009. Bayesian t tests for accepting and rejecting the null hypothesis. *Psychonomic bulletin & review*, 16(2):225–237.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Yi-Ting Tsai, Min-Chu Yang, and Han-Yu Chen. 2019. Adversarial attack on sentiment classification. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 233–240, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Eric-Jan Wagenmakers, Ruud Wetzels, Denny Borsboom, and Han LJ Van Der Maas. 2011. Why psychologists must change the way they analyze their data: the case of psi: comment on bem (2011). *Journal of Personality and Social Psychology*, 100(3):426 – 432.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080, Online. Association for Computational Linguistics.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4903–4912, Hong Kong, China. Association for Computational Linguistics.

A Dataset statistics

The SST-2 dataset comes with a pre-defined split of 67,349 samples for training, 872 for validation and 1,821 for testing. The IMDb dataset consists of 50,000 positive and negative movie reviews with a pre-defined split of 25,000 training and 25,000 test samples. Since this dataset does not have a pre-defined validation set, we hold out 1,000 randomly selected training set samples for validation. We select a validation set of roughly the same size as for SST-2 for fair comparisons when tuning parameters for adversarial example detection. To the best of our knowledge, the compared work (Alzantot et al., 2018; Ren et al., 2019) does not validate model performance on held-out training data.

B Model and attack details

B.1 RoBERTa

We utilize a pre-trained RoBERTa (base) model (Liu et al., 2019) provided by the *Hugging Face Transformers* library (Wolf et al., 2019). We use maximum input sequence lengths of 256 and 128 after byte-pair encoding (Sennrich et al., 2016) for the IMDb and SST-2 datasets, respectively. The RoBERTa model consists of 125 million parameters.[8] The model was trained for 10 epochs with batch size 32 (SST-2) and 16 (IMDb) and a learning rate of $1 \cdot 10^{-5}$. We evaluated model performance after each epoch on the validation set and selected the best-performing checkpoints for testing.

B.2 CNN/LSTM

The CNN architecture consists of 3 convolutional layers with kernel sizes 2, 3 and 4 and 100 feature maps for each convolutional layer. The LSTM operates on a hidden state size of 128. Following Alzantot et al. (2018), we initialize the LSTM with pre-trained GLOVE (Pennington et al., 2014) word embeddings, and do the same for the CNN.

Both the LSTM and the CNN use *Dropout* (Srivastava et al., 2014) during training with a rate of 0.1 before applying the output layer. We trained both models for 20 epochs using the *Adam* optimizer (Kingma and Ba, 2014). We evaluated model performance after each epoch on the validation set and selected the best-performing checkpoints for testing. The CNN and LSTM models were trained with batch size 100 and a learning rate of $1 \cdot 10^{-3}$.

B.3 PWWS

Our implementation of PWWS is based on the code as provided by Ren et al. (2019) on GitHub.[9]

B.4 GENETIC

Note that we utilize a different language model for the `Perturb` subroutine as compared to the original implementation by Alzantot et al. (2018). While Alzantot et al. (2018) employ the Google 1 billion words language model (Chelba et al., 2013), we instead utilize the recently proposed GPT-2 language model (Radford et al., 2019) and compute the sequences' perplexity scores using the exponentialized language modelling loss (we employ the pre-trained `GPT2LMHeadModel` language model from Wolf et al. (2019)). We compute the perplexity scores for each perturbed sequence only around the respective replacement words by only considering a subsequence ranging from five words before to five words after an inserted replacement. The motivation for using a different language model as compared to the original implementation is due to computational efficiency, since we observed a notable decrease in attack runtime with our modification. This does not have an impact on attack performance, since our implementation of the GENETIC has an attack success rate of 98.6% against the LSTM on IMDb, whereas Alzantot et al. (2018) report an attack success rate of 97%.

For attacks against SST-2, we furthermore increase the δ threshold for the maximum distance between replaced words and substitutions to $\delta = 1.0$,

---

[8] `https://github.com/pytorch/fairseq/tree/master/examples/roberta`

[9] `https://github.com/JHL-HUST/PWWS`

Dataset	Model	Attack	Replaced		Subst.			non-OOV		
			μ_ϕ	σ_ϕ	μ_ϕ	σ_ϕ	d	μ_ϕ	σ_ϕ	d
IMDb	CNN	Random	7.6	2.5	3.5	2.8	**1.6**	4.4	2.4	1.3
		Prioritized	7.6	2.5	3.3	2.7	**1.6**	3.9	2.5	1.5
		Genetic	6.3	2.0	3.5	2.2	1.3	3.7	2.1	1.3
		PWWS	6.7	2.3	4.0	2.4	1.1	4.5	2.1	1.0
	LSTM	Random	7.6	2.5	3.5	2.8	1.6	4.4	2.4	1.3
		Prioritized	7.6	2.5	2.8	2.3	**2.0**	3.2	2.2	1.8
		Genetic	6.2	2.0	3.1	1.9	1.6	3.3	1.8	1.5
		PWWS	6.4	2.2	3.5	2.1	1.4	3.7	1.9	1.3
SST-2	CNN	Random	5.4	2.5	2.0	2.3	**1.4**	3.8	1.7	0.7
		Prioritized	5.4	2.5	2.4	2.1	1.3	3.5	1.7	0.8
		Genetic	4.3	1.8	2.2	1.9	1.2	3.2	1.4	0.6
		PWWS	4.8	2.1	2.8	2.1	1.0	3.8	1.5	0.6
	LSTM	Random	5.4	2.6	2.0	2.3	**1.4**	3.8	1.7	0.7
		Prioritized	5.4	2.5	2.3	2.1	1.3	3.4	1.6	0.9
		Genetic	4.3	1.7	2.0	1.9	1.3	3.1	1.4	0.8
		PWWS	4.8	2.0	2.7	2.1	1.0	3.7	1.4	0.6

Table 4: Mean $\log_e$ frequencies of replaced words and their corresponding substitutions by attack, model and dataset. The shown values are the mean μ_ϕ and standard deviation σ_ϕ of the $\log_e$ frequencies corresponding to each setting, and additionally the Cohen's d effect sizes for the substitutions. Values in bold denote largest effect sizes per dataset and model.

since we observed poor attack performances with $\delta = 0.5$ (which was used by Alzantot et al. (2018) and in our experiments on IMDb). All other parameters of the attack (e.g., the number of generations and population size) are directly adapted from Alzantot et al. (2018).

We restrict the words eligible for replacement by the Genetic attack to non-stopwords, in accordance to Alzantot et al. (2018). Since the attack computes nearest neighbors for a selected word from a pre-trained embedding space, we furthermore can only select words for which there exists an embedding representation in this pre-trained space. On the SST-2 test set, we found three input sequences consisting of only one word which we excluded from our evaluation, since the used GPT-2 language model implementation requires an input sequence consisting of more than one word.

B.5 Random, Prioritized, PWWS, Genetic

For the Genetic attack, we follow Alzantot et al. (2018) by limiting the maximum amount of word replacements to 20% of the input sequence length. We apply the same threshold to the Random and Prioritized attacks, but not to PWWS since we observed low replacement rates despite the attack's effectiveness. This is in agreement to the results reported in Ren et al. (2019).

C Frequency differences for CNN and LSTM models

The $\log_e$ frequencies for the four attacks against the CNN and LSTM models can be found in Table 4. In accordance to the experiments with RoBERTa (see Section 3 in the paper), we observe large Cohen's d effect sizes for the majority of the comparisons, which shows that the statistical frequency differences between replaced words and their substitutions are present for adversarial attacks against these two models as well.

D Bayes factors

The Bayes factors for the mean frequency comparisons between replaced words and their adversarial substitutions can be found in Table 5. We observe high values for BF_{10} across all comparisons, providing strong evidence for the hypothesis that the $\log_e$ frequency means between replaced words and their substitutions are different.

E Visualizations of frequency differences

Figure 3 illustrates the frequency differences for attacks against the RoBERTa model using his-

Dataset	Model	Attack	Subst.	non-OOV
IMDb	CNN	Random	$> 10^{10594}$	$> 10^{7004}$
		Prioritized	$> 10^{6549}$	$> 10^{5009}$
		Genetic	$> 10^{2581}$	$> 10^{2318}$
		PWWS	$> 10^{2182}$	$> 10^{1673}$
	LSTM	Random	$> 10^{9643}$	$> 10^{6338}$
		Prioritized	$> 10^{5949}$	$> 10^{4967}$
		Genetic	$> 10^{2550}$	$> 10^{2369}$
		PWWS	$> 10^{1666}$	$> 10^{1442}$
	RoBERTa	Random	$> 10^{12138}$	$> 10^{7948}$
		Prioritized	$> 10^{9014}$	$> 10^{6043}$
		Genetic	$> 10^{4215}$	$> 10^{3672}$
		PWWS	$> 10^{5182}$	$> 10^{3656}$
SST-2	CNN	Random	$> 10^{754}$	$> 10^{138}$
		Prioritized	$> 10^{573}$	$> 10^{222}$
		Genetic	$> 10^{388}$	$> 10^{104}$
		PWWS	$> 10^{397}$	$> 10^{131}$
	LSTM	Random	$> 10^{800}$	$> 10^{153}$
		Prioritized	$> 10^{648}$	$> 10^{264}$
		Genetic	$> 10^{522}$	$> 10^{148}$
		PWWS	$> 10^{456}$	$> 10^{144}$
	RoBERTa	Random	$> 10^{867}$	$> 10^{149}$
		Prioritized	$> 10^{779}$	$> 10^{130}$
		Genetic	$> 10^{584}$	$> 10^{55}$
		PWWS	$> 10^{600}$	$> 10^{125}$

Table 5: Bayes factors (BF_{10}) for the Bayesian hypothesis tests.

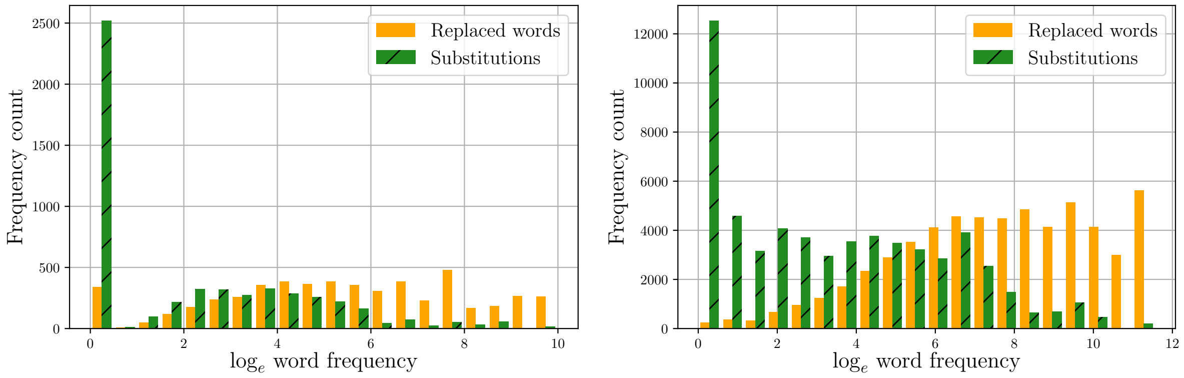
tograms. We observe that for the majority of the attacks, OOV substitutions occur most often among the perturbed sequences.

F Varying false positive thresholds

The rate of false positives predicted by a detection system is crucial for its practicability, and a limited amount of false positives is hence highly desirable. Figure 4 illustrates the true positive rates predicted by FGWS for all attacks against RoBERTa with different quasi-fixed false positive thresholds (as in Section 4.2 of the paper, δ was tuned on the validation set for each value of γ corresponding to the specific false positive threshold). As expected, we observe a trade-off between true and false positive rates for varying values of γ, such that lower false positive rates imply lower true positive rates. However, even for false positive rates of 1% and 5%, we observe that FGWS is able to detect between 33.6% and 90.0% of adversarial examples on IMDb and between 31.7% and 67.2% on SST-2. This indicates that FGWS has the potential to detect a useful fraction of adversarial examples without creating an excessive burden of false positives.

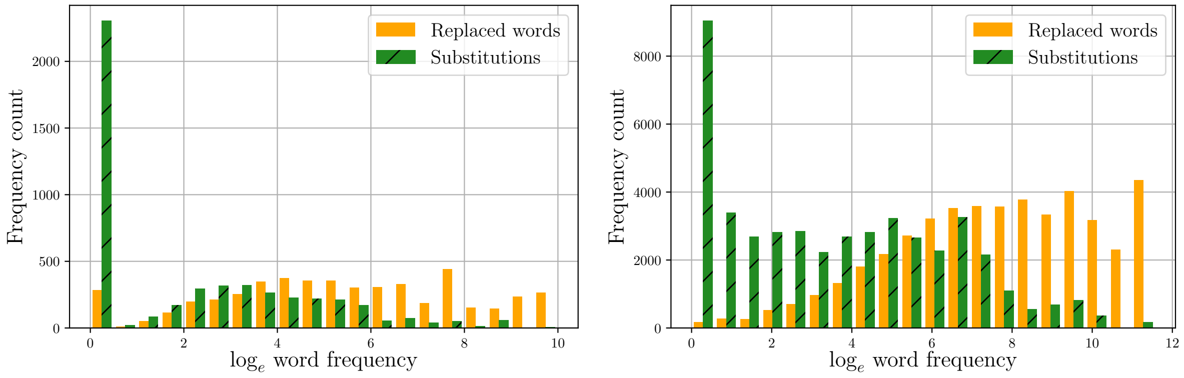
G Additional FGWS examples

Additional examples of FGWS can be found in Table 6 (true positives), Table 7 (false positives), Table 8 (true negatives) and Table 9 (false negatives).



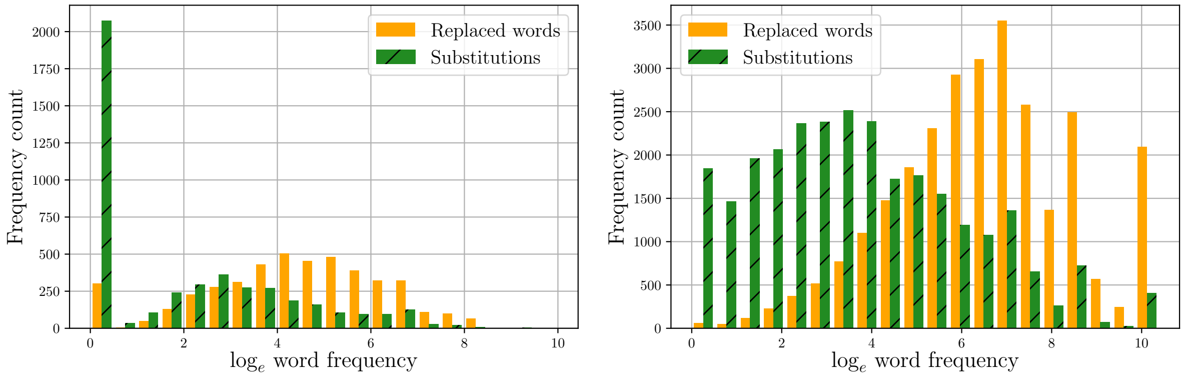
(a) Random on SST-2

(b) Random on IMDb



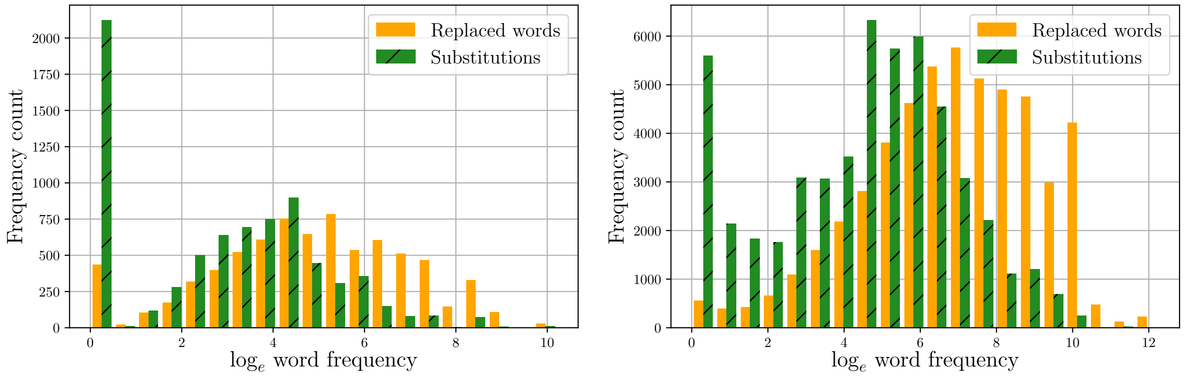
(c) Prioritized on SST-2

(d) Prioritized on IMDb



(e) Genetic on SST-2

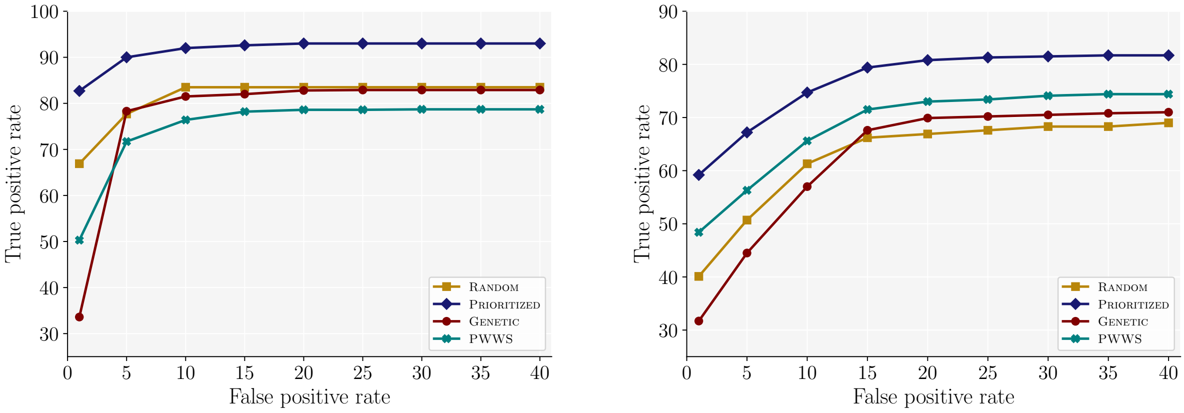
(f) Genetic on IMDb



(g) PWWS on SST-2

(h) PWWS on IMDb

Figure 3: Histograms showing the frequency distribution of words replaced by the attacks and their corresponding substitutions against the RoBERTa model. The x-axis represents the words' $\log_e$ frequency with respect to the model's training corpus, the y-axis denotes their respective frequencies among the perturbed test set sequences.



(a) IMDb

(b) SST-2

Figure 4: The trade-off between true and false positive rates on the test sets with all four attacks against RoBERTa on (a) IMDb and (b) SST-2. The true positive rates (y-axis) are computed when γ is set to allow for different quasi-fixed amounts of false positives (x-axis).

Model:RoBERTa on SST-2

Unperturbed	first good then bothersome	*negative* (74.5%)
GENETIC	first good then $\overset{0.00}{\textbf{galling}}$ $\overset{0.00}{[\textbf{\textit{bothersome}}]}$	*positive* (88.7%)
DISP	first good $\overset{8.96}{\textbf{that}}$ $\overset{5.31}{[\textit{then}]}$ galling	*positive* (84.8%)
FGWS	first good then $\overset{4.32}{\textbf{annoying}}$ $\overset{0.00}{[\textit{galling}]}$	*negative* (91.3%)

Model:RoBERTa on IMDb

Unperturbed	i am a huge rupert everett fan . i adore kathy bates so when i saw it available i decided to check it out . the synopsis didn t really tell you much . in parts it was silly touching and in others some parts were down right hysterical . any person that is a huge fan of a personality of any type will find some small identifying traits with the main character . of course there are many they won t but that is the point if you like any of the actors give it a watch but don t look for any thing too dramatic it s good fun . i might also mention you can see how darn tall rupert is . i mean i knew he was 6 4 but he seems even more in this film . he even seemed to stoop a bit due to the other characters height in this . he is tall i mean tall and for you rupert fans there is a bare chest scene ... wonderful	*positive* (99.2%)
PWWS	i am a huge rupert everett fan . i adore kathy bates so when i saw it available i decided to $\overset{6.54}{\textbf{stop}}$ $\overset{6.19}{[\textbf{\textit{check}}]}$ it out . the synopsis didn t really tell you much . in parts it was silly touching and in others some parts were down right hysterical . any person that is a huge fan of a personality of any type will find some small identifying traits with the main character . of course there are many they won t but that is the point if you like any of the actors give it a watch but don t look for any thing too dramatic it s $\overset{0.00}{\textbf{undecomposed}}$ $\overset{9.22}{[\textbf{\textit{good}}]}$ fun . i might also mention you can see how darn tall rupert is . i mean i knew he was 6 4 but he seems even more in this film . he even seemed to stoop a bit $\overset{0.00}{\textbf{imputable}}$ $\overset{6.31}{[\textbf{\textit{due}}]}$ to the other characters height in this . he is tall i mean tall and for you rupert fans there is a bare chest scene ... $\overset{4.45}{\textbf{tremendous}}$ $\overset{7.08}{[\textbf{\textit{wonderful}}]}$	*negative* (60.1%)
DISP	i am a huge rupert everett fan . i adore kathy bates so when i saw it available i decided to $\overset{9.27}{\textbf{out}}$ $\overset{6.54}{[\textit{stop}]}$ it out . the synopsis didn t really tell you much . in parts it was silly touching and in others some parts were down right hysterical . any person that is a huge fan of a personality of any type will find some small identifying traits with the main character . of course there are many they won $\overset{0.00}{\textbf{;}}$ $\overset{9.97}{[\textit{t}]}$ but that is the point if you like any of the actors give it a watch but don t look for any thing too dramatic it $\overset{0.00}{\textbf{’s}}$ $\overset{10.54}{[\textit{s}]}$ $\overset{9.50}{\textbf{so}}$ $\overset{0.00}{[\textbf{\textit{undecomposed}}]}$ fun . i $\overset{9.20}{\textbf{can}}$ $\overset{7.44}{[\textit{might}]}$ also mention you can see how darn tall rupert is . i mean i knew he was 6 4 but he seems even more in this film . he even seemed to stoop a bit imputable to the other characters height in this . he is tall i mean tall and for you rupert fans there is a bare chest scene ... $\overset{12.05}{\textbf{:}}$ $\overset{4.45}{[\textbf{\textit{tremendous}}]}$	*positive* (92.0%)
FGWS	i am a huge rupert everett fan . i adore kathy bates so when i saw it available i decided to stop it out . the synopsis didn t really tell you much . in parts it was silly touching and in others some parts were down right hysterical . any person that is a huge fan of a personality of any type will find some small $\overset{7.26}{\textbf{place}}$ $\overset{2.48}{[\textbf{\textit{identifying}}]}$ traits with the main character . of course there are many they won t but that is the point if you like any of the actors give it a watch but don t look for any thing too dramatic it s $\overset{9.22}{\textbf{good}}$ $\overset{0.00}{[\textbf{\textit{undecomposed}}]}$ fun . i might also mention you can see how darn tall rupert is . i mean i knew he was 6 4 but he seems even more in this film . he even seemed to $\overset{6.22}{\textbf{sit}}$ $\overset{2.40}{[\textbf{\textit{stoop}}]}$ a bit $\overset{6.31}{\textbf{due}}$ $\overset{0.00}{[\textbf{\textit{imputable}}]}$ to the other characters height in this . he is tall i mean tall and for you rupert fans there is a bare chest scene ... tremendous	*positive* (88.9%)

Table 6: Illustration of true positives generated with FGWS against RoBERTa on SST-2 (top) and IMDb (bottom). The substitutions caused the model to change the predicted label back to its ground-truth for the given adversarial examples.

Model: RoBERTa on SST-2

Unperturbed	imagine if you will a tony hawk skating video interspliced with footage from behind enemy lines and set to jersey shore techno	*negative* (83.6%)
DISP	imagine if you $\overset{5.97}{\textbf{get}}$ [$\overset{6.84}{\textit{will}}$] a tony hawk skating video $\overset{0.00}{\textbf{;}}$ [$\overset{0.00}{\textit{interspliced}}$] with footage from behind enemy lines and set to jersey shore techno	*negative* (87.1%)
FGWS	imagine if you will a $\overset{3.76}{\textbf{kevin}}$ [$\overset{1.10}{\textit{tony}}$] $\overset{3.93}{\textbf{pitch}}$ [$\overset{2.48}{\textit{hawk}}$] skating video interspliced with footage from behind enemy lines and set to $\overset{6.55}{\textbf{new}}$ [$\overset{2.08}{\textit{jersey}}$] $\overset{3.93}{\textbf{sea}}$ [$\overset{2.08}{\textit{shore}}$] $\overset{5.69}{\textbf{music}}$ [$\overset{1.61}{\textit{techno}}$]	*positive* (65.7%)

Model: RoBERTa on IMDb

Unperturbed	admittedly alex has become a little podgey but they are still for me the greatest rock trio ever . i wholeheartedly recommend this dvd to any fan . i was very disappointed that they canceled their planned recent munich gig logistics and regret not making an effort to see them elsewhere . the dvd is a small consolation the greatest incentive to acquire a proper dvd playback setup . naive perhaps but i still don t understand the significance of the tumble driers on stage i would be grateful for any clarification . cheers iain .	*positive* (99.4%)
DISP	admittedly alex has become a little podgey but they are still for me the greatest rock trio ever . i wholeheartedly recommend this dvd to any fan . i was very disappointed that they canceled their planned recent munich gig logistics and regret not making an effort to see them elsewhere . the dvd is a small consolation the greatest incentive to acquire a proper dvd playback setup . naive perhaps but i still don t understand the significance of the $\overset{9.77}{\textbf{one}}$ [$\overset{1.95}{\textit{tumble}}$] driers on stage i would be grateful for any clarification . cheers $\overset{10.73}{\textbf{that}}$ [$\overset{0.00}{\textit{iain}}$] .	*positive* (99.3%)
FGWS	admittedly alex has become a little podgey but they are still for me the greatest rock trio ever . i $\overset{4.55}{\textbf{disagree}}$ [$\overset{2.30}{\textit{wholeheartedly}}$] recommend this dvd to any fan . i was very disappointed that they canceled their planned recent $\overset{5.03}{\textbf{germany}}$ [$\overset{2.08}{\textit{munich}}$] gig $\overset{3.40}{\textbf{transport}}$ [$\overset{0.69}{\textit{logistics}}$] and regret not making an effort to see them elsewhere . the dvd is a small $\overset{5.69}{\textbf{win}}$ [$\overset{2.08}{\textit{consolation}}$] the greatest $\overset{5.53}{\textbf{opportunity}}$ [$\overset{1.61}{\textit{incentive}}$] to acquire a proper dvd $\overset{6.21}{\textbf{editing}}$ [$\overset{2.08}{\textit{playback}}$] setup . naive perhaps but i still don t understand the significance of the $\overset{6.19}{\textbf{fall}}$ [$\overset{1.95}{\textit{tumble}}$] $\overset{1.61}{\textbf{dryer}}$ [$\overset{0.00}{\textit{driers}}$] on stage i would be grateful for any $\overset{5.11}{\textbf{explanation}}$ [$\overset{1.10}{\textit{clarification}}$] . cheers iain .	*negative* (50.1%)

Table 7: Illustration of false positives generated with FGWS against RoBERTa on SST-2 (top) and IMDb (bottom). The substitutions caused the model to change the predicted label for the given unperturbed sequences.

Model: RoBERTa on SST-2

Unperturbed	it s a hoot and a half and a great way for the american people to see what a candidate is like when he s not giving the same 15 cent stump speech	*positive* (100.0%)
DISP	it $\overset{0.00}{\textbf{'s}}$ [$\overset{9.09}{\textit{s}}$] a hoot and a half and a great way for the american people to see what a candidate is like when he $\overset{0.00}{\textbf{'s}}$ [$\overset{9.09}{\textit{s}}$] not giving the same 15 $\overset{6.01}{\textbf{minutes}}$ [$\overset{0.00}{\textit{cent}}$] $\overset{10.22}{\textbf{the}}$ [$\overset{0.00}{\textit{stump}}$] speech	*positive* (100.0%)
FGWS	it s a hoot and a half and a great way for the american people to see what a $\overset{3.71}{\textbf{nomination}}$ [$\overset{1.95}{\textit{candidate}}$] is like when he s not giving the same 15 cent $\overset{2.48}{\textbf{stamp}}$ [$\overset{0.00}{\textit{stump}}$] $\overset{4.45}{\textbf{words}}$ [$\overset{0.00}{\textit{speech}}$]	*positive* (100.0%)

Model: RoBERTa on IMDb

Unperturbed	it was awful plain and simple . what was their message where was the movie going with this it has all the ingredients of a sub b grade movie . from plotless storyline the bad acting to the cheesey slow mo cinematography . i d sooner watch a movie i ve already seen like goodfellas a bronx tale even grease . there are no likeable characters . in the end you just want everyone to die already . save 2 hours of your life and skip this one .	*negative* (99.9%)
DISP	it was awful plain and simple . what was their message where was the movie going with this it has all the ingredients of a sub b grade movie . from plotless storyline the bad acting to the cheesey slow mo cinematography . i $\overset{8.94}{\textbf{would}}$ [$\overset{7.56}{\textit{d}}$] sooner watch a movie i $\overset{9.79}{\textbf{have}}$ [$\overset{8.17}{\textit{ve}}$] already seen like goodfellas a bronx tale $\overset{10.97}{\textbf{in}}$ [$\overset{8.97}{\textit{even}}$] grease . there are no likeable characters . in the end you just want everyone to die already . save 2 hours of your life and skip this one .	*negative* (99.9%)
FGWS	it was awful plain and simple . what was their message where was the movie going with this it has all the ingredients of a sub b grade movie . from $\overset{4.28}{\textbf{unwatchable}}$ [$\overset{1.39}{\textit{plotless}}$] storyline the bad acting to the $\overset{6.10}{\textbf{cheesy}}$ [$\overset{1.95}{\textit{cheesey}}$] slow mo cinematography . i d sooner watch a movie i ve already seen like goodfellas a bronx tale even grease . there are no likeable characters . in the end you just want everyone to die already . save 2 hours of your life and skip this one .	*negative* (99.9%)

Table 8: Illustration of true negatives generated with FGWS against RoBERTa on SST-2 (top) and IMDb (bottom). The substitutions did not cause the model to change the predicted label for the given unperturbed sequences.

Model: RoBERTa on SST-2

Unperturbed	the spark of special anime magic here is unmistakable and hard to resist	*positive* (100.0%)
PWWS	the spark of special anime $\overset{2.83}{\textbf{deception}}$ [$\overset{4.52}{\textit{magic}}$] here is unmistakable and $\overset{2.77}{\textbf{laborious}}$ [$\overset{6.15}{\textit{hard}}$] to $\overset{4.58}{\textbf{hold}}$ [$\overset{3.91}{\textit{resist}}$]	*negative* (84.4%)
DISP	the spark of special anime deception here is unmistakable and $\overset{4.88}{\textbf{able}}$ [$\overset{2.77}{\textit{laborious}}$] to hold	*positive* (99.9%)
FGWS	the spark of special anime deception here is $\overset{4.52}{\textbf{subtle}}$ [$\overset{2.48}{\textit{unmistakable}}$] and laborious to hold	*negative* (97.8%)

Model: RoBERTa on IMDb

Unperturbed	graduation day is a result of the success of friday the 13th . both of those films are about creative bloody murders rather than suspense . if you enjoy that type of film i d recommend graduation day . if not i wouldn t. there s nothing new here just the same old killings . even though i ve given the film a 4 out of 10 i will say that it s not a repulsive film . it is watchable if your curious about it just not creative .	*negative* (71.3%)
GENETIC	graduation day is a result of the success of friday the 13th . both of those films are about creative bloody murders rather than suspense . if you enjoy that type of film i d recommend graduation day . if not i wouldn t. there s nothing new here just the same $\overset{5.06}{\textbf{ancient}}$ [$\overset{8.03}{\textit{old}}$] killings . even though i ve given the film a 4 out of 10 i will say that it s not a repulsive film . it is watchable if your curious about it just not creative .	*positive* (53.5%)
DISP	graduation day is a result of the success of friday the 13th . both of those films are about creative bloody murders rather than suspense . if you enjoy that type of film i $\overset{8.94}{\textbf{would}}$ [$\overset{7.56}{\textit{d}}$] recommend graduation day . if not i $\overset{8.64}{\textbf{do}}$ [$\overset{6.48}{\textit{wouldn}}$] t. there $\overset{11.14}{\textbf{is}}$ [$\overset{10.54}{\textit{s}}$] nothing new here just the same ancient killings . even though i $\overset{9.79}{\textbf{have}}$ [$\overset{8.17}{\textit{ve}}$] given the film a 4 out of 10 i will say that it $\overset{0.00}{\textbf{'s}}$ [$\overset{10.54}{\textit{s}}$] not a $\overset{9.22}{\textbf{good}}$ [$\overset{3.69}{\textit{repulsive}}$] film . it is watchable if your curious about it $\overset{11.14}{\textbf{is}}$ [$\overset{9.32}{\textit{just}}$] not creative .	*negative* (99.5%)
FGWS	graduation day is a result of the success of friday the 13th . both of those films are about creative bloody murders rather than suspense . if you enjoy that type of film i d recommend graduation day . if not i wouldn t. there s nothing new here just the same ancient killings . even though i ve given the film a 4 out of 10 i will say that it s not a repulsive film . it is watchable if your curious about it just not creative .	*positive* (53.5%)

Table 9: Illustration of false negatives generated with FGWS against RoBERTa on SST-2 (top) and IMDb (bottom). The substitutions did not cause the model to change the predicted label back to its ground-truth for the given adversarial examples.

Maximal Multiverse Learning for Promoting Cross-Task Generalization of Fine-Tuned Language Models

Itzik Malkiel
Tel Aviv University
itzik.malkiel@gmail.com

Lior Wolf
Tel Aviv University
wolf@cs.tau.ac.il

Abstract

Language modeling with BERT consists of two phases of (i) unsupervised pre-training on unlabeled text, and (ii) fine-tuning for a specific supervised task. We present a method that leverages the second phase to its fullest, by applying an extensive number of parallel classifier heads, which are enforced to be orthogonal, while adaptively eliminating the weaker heads during training. We conduct an extensive inter- and intra-dataset evaluation, showing that our method improves the generalization ability of BERT, sometimes leading to a +9% gain in accuracy. These results highlight the importance of a proper fine-tuning procedure, especially for relatively smaller-sized datasets. Our code is attached as supplementary.

1 Introduction

Recently, there has been an increasing number of studies suggesting the use of general language models, for improving natural language processing tasks (Dai and Le, 2015; Peters et al., 2018; Radford et al., 2018; Howard and Ruder, 2018). Among the most promising techniques, the unsupervised pretraining approach (Dai and Le, 2015; Radford et al., 2018) has emerged as a very successful method, that achieves state-of-the-art results on many language tasks, including sentiment analysis (Socher et al., 2013), natural language inference (Williams et al., 2017) and similarity and paraphrasing tasks (Dolan and Brockett, 2005; Cer et al., 2017). This approach incorporates a two-phase training procedure. The first phase utilizes an unsupervised training of a general language model on a large corpus. The second phase applies supervision to fine-tune the model for a given task.

More recently, unsupervised pretraining models such as BERT (Devlin et al., 2018), XLNET (Yang et al., 2019) and RoBERTa (Liu et al., 2019), have achieved unprecedented performance. For example, in the GLUE benchmark (Wang et al., 2018), BERT (Devlin et al., 2018) was reported to achieve performance that exceeds human level on a few different datasets, such as QNLI (Rajpurkar et al., 2016), QQP (Chen et al., 2018) and MRPC (Dolan and Brockett, 2005). However, despite the great progress achieved by these task-specific and dataset-specific models, it is not yet clear how well they can generalize to different tasks, how well they generalize when evaluating the same task on different datasets, and how to improve this generalization ability.

In our work, we extend the multiverse method of (Littwin and Wolf, 2016), which was shown to improve transfer learning in the computer vision task of face recognition and on the CIFAR-100 small image recognition dataset. The multiverse loss generalizes the cross entropy loss, by simultaneously training multiple linear classification heads to perform the same task. In order to prevent multiple copies of the same classifier, in the multiverse scheme, each classifier is mutually orthogonal to the rest of classifiers. The number of multiverse heads used was limited, never more than seven and typically set to five.

We propose a novel fine-tuning procedure for enhancing the generalization ability of the recent unsupervised pretrained language models, by employing a large number of multiverse heads. The essence of our technique is as follows: given a pretrained language model and a downstream task with labeled data, we fine-tune the model using a maximal number of multiverse classifiers. The fine-tuning goal is to both minimize the task loss and an orthogonality loss applied to the classification heads. When enforcing orthogonality hinders the classifiers' performance, we detect and eliminate the less effective classification heads.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 187–199
April 19 - 23, 2021.

The technique, therefore, preserves a maximal set of classifiers, which is comprised of the best performing ones. By maintaining this maximal subset during training, our method leverages multiverse loss to its fullest. Hence, we name our method Maximal Multiverse Learning (MML).

Our contributions are as follows: (1) we present MML, a general training procedure to improve the transferability of neural models. (2) we apply MML on BERT and report its performance on various datasets. (3) we propose a set of cross dataset evaluations using common NLP benchmarks, demonstrating the effectiveness of MML, in comparison to regular BERT fine-tuning and to alternative regularization techniques.

2 Related work

Many recent breakthroughs in NLP employ unsupervised pretraining of language models. The different variants can be categorized into two main approaches: (1) feature-based models, such as (Peters et al., 2018) and (2) fine-tuning models, such as (Devlin et al., 2018; Liu et al., 2019; Yang et al., 2019). The former technique utilizes a language neural-based model as a feature extractor. The extracted features may be used for the training of another model, receiving the extracted features as input. The second approach utilizes a similar pre-trained model, but fine-tunes it in an end-to-end manner to specialize on a given task. During the fine-tuning phase, all of the parameters of the model are updated and a relatively small number of parameters are trained from scratch.

The usage of multiple classifiers can be found in few places in the literature. In GoogLeNet (Szegedy et al., 2015), the authors use multiple classifier heads in different places in the model architecture. The additional classifiers led to better propagation of the gradients during training. However, with the advent of better conditioning and normalization methods, as well as with the modern introduction of skip connections in architectures such as the ResNet (He et al., 2016), the practice of adding intermediate branches, for the sake of introducing an auxiliary loss at lower layers, was mostly abandoned.

The multiverse loss was shown to promote transfer learning and to lead to a low-dimensional representation in the penultimate layer (Littwin and Wolf, 2016). However, the current literature does not present any methodological way to select the number of multiverse heads and the idea was only applied for a handful of parallel classifiers. In MML, hundreds of multiverse heads are used, leading to a tradeoff between the classifier accuracy and the orthogonality constraint. MML balances the two terms by pruning, during training, the under-performing heads.

3 Method

Let $\mathscr{W} = \{w_i\}_{i=1}^{w}$ be the vocabulary of tokens in a given language. Let Y be the set of all possible sentences generated by $\mathscr{W}$, including the empty sentence. A language model $M : Y \times Y \to \mathbb{R}^d$ receives a pair of elements from Y and returns a vectors of d dimensions. Given a dataset with n training samples, $s_1 ... s_n \in Y \times Y$, each associated with a label $y_i \in [1...c]$, we denote the coding vector of each sample by $d_i := M(s_i)$. As a concrete example, for the BERT model, d_i is the latent embedding of the CLS token.

Common language models employ a classifier $C : \mathbb{R}^d \to \mathbb{R}^c$ that projects the coding vectors $d_i \in \mathbb{R}^d$ by a $d \times c$ matrix, $F_{d\times c} = [f_1, ..., f_c]$, $(f_i \in \mathbb{R}^d)$, and then adds a bias term $b \in \mathbb{R}^c$:

$$C(d_i) = d_i^T F_{d\times c} + b \tag{1}$$

The output of C is a logit vector, and pseudo-probabilities are obtained by applying softmax $p_i(y_i) = e^{C(d_i)_{y_i}} / \sum_{j=1}^{c} e^{C(d_i)_j}$.

Different from the single-classifier models, our model utilizes a multiverse classifier $\mathscr{C} : \mathbb{R}^d \to \mathbb{R}^{c\times m}$ defined as $\mathscr{C}(d_i) = (C_1(d_i), ..., C_m(d_i))$, where m is a multiplicity parameter, $\left\{C_j : \mathbb{R}^d \to \mathbb{R}^c\right\}_{j=1}^{m}$ are parallel classifiers, each with different weights, applying the same function as Eq. 1. :

$$C_j(d_i) = d_i^T F_{d\times c}^{j} + b_j \tag{2}$$

Additionally, we will define $B = \left\{\beta_j \in \{0,1\}\right\}_{j=1}^{m}$ as a set of binary scalars. Each classifier head C_j will be associated with a different binary scalar β_j, which is set during training. In our experiments, we set m to be equal to the coding vector size d, which entails a full rank of active multiverse classifiers at the beginning of the training.

The loss function is composed of two components, the task loss and the multiverse loss. Defining active multiverse classifiers as those that are associated with a value $\beta_j = 1$, the task loss optimizes the performance of all active multiverse

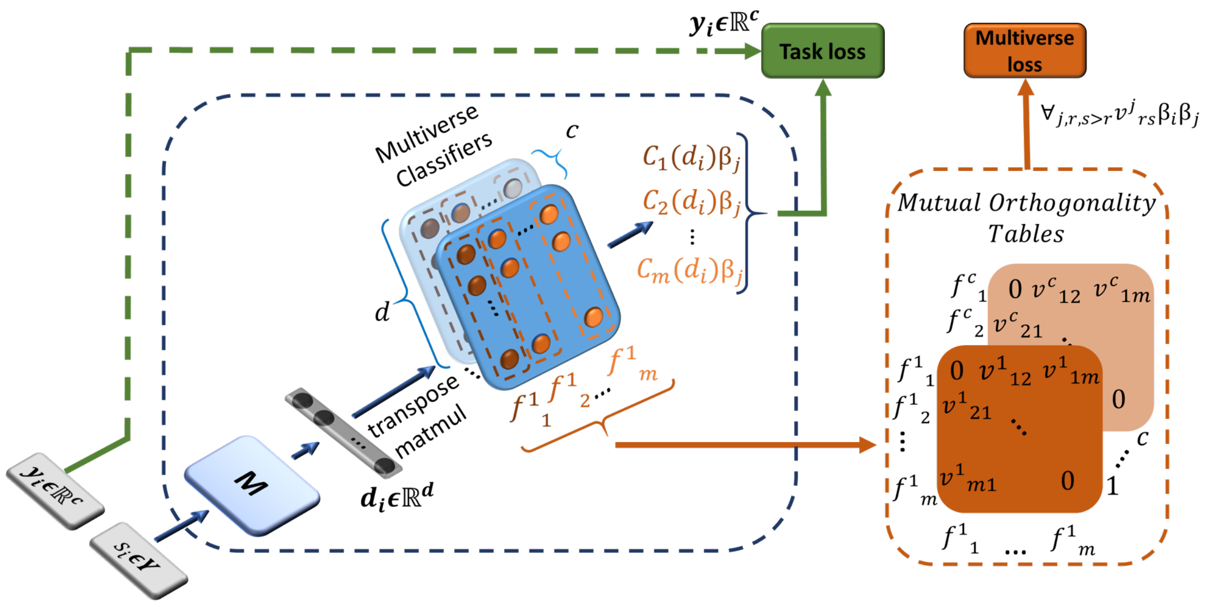


Figure 1: A schematic illustration of the MML model. The task loss comprises a loss-term for each multiverse classifier, using the given labels of the task at hand. The mutual orthogonality tables hold the absolute value of the dot product calculated between the weights of all classifiers, across the different classes. Since orthogonality of a classifier with itself is ignored, we set the diagonal to 0. Following multiverse loss definition and since orthogonality is symmetric, only half of each table value is passed to the multiverse loss.

Algorithm 1 MML training. The MeanShift function returns the center of the clusters. Params: $K = 1000$, $\gamma = 0.99$, $Threshold = 5$, $\alpha = 2 \cdot e^{-5}$, $\lambda = 0.005$.

```
$\forall 1 \le j \le m : \beta_j \leftarrow 1, a_j \leftarrow 0$
for $step = 1, 2, \ldots$ do
    Sample a minibatch $\{(s_i, y_i)\}_{i=1}^{t}$
    $MC_\theta \leftarrow \nabla_\theta \left[ \lambda \cdot \mathcal{L}_{mv}(\mathscr{C}, B) + \frac{1}{t}\Sigma_{i=1}^{t} \mathcal{L}_{task}\left(M, \mathscr{C}, B, \{(s_i, y_i)\}_{i=1}^{t}\right) \right]$
    for $1 \le j \le m$ do
        $a_j \leftarrow (1 - \gamma) \cdot a_j + \left(\gamma \cdot \frac{1}{t}\Sigma_{i=1}^{t} \mathcal{L}_{task}\left(M, \mathscr{C}, B, \{(s_i, y_i)\}_{i=1}^{t}\right)\right)$
    $\theta \leftarrow \theta + \text{Adam}(\theta, MC_\theta, \alpha)$
    if $step \% K = 0$ and $\sum_j \beta_j \ge Threshold$ then
        clusters $\leftarrow$ MeanShift$\left(\{a_j | \beta_j = 1\}_{j=1}^{m}\right)$
        if $|clusters| \ge 2$ then
            for $1 \le j \le m$ do
                $\beta_j \leftarrow 0$, if $a_j \notin min(clusters)$
```

classifiers, each independently, using the supervision obtained by the given labels. The multiverse loss soft-enforces orthogonality among the active classifiers. Its purpose is to regularize the model by encouraging M to produce coding vectors that are robust enough to be effective for a large number of orthogonal classifiers.

As mentioned earlier, each classifier C_j is associated with a binary value β_j, which controls the applicability of the classifier and is configured during training. Under the context of the loss function, setting β_j to 0 would eliminate the impact of the j^{th} classifier head C_j for both the task loss and multiverse loss.

For a multi-class classification task, we apply the following task loss:

$$\mathcal{L}_{task} = -\Sigma_{i=1}^{n} \Sigma_{j=1}^{m} \mathcal{L}_{cce}^{j,i} \beta_j \tag{3}$$

where n is the number of training samples, and $\mathcal{L}_{cce}^{j,i} = y_i log(C_j(M(s_i))_{y_i})$ is the cross entropy loss. For a binary classification task we set $\mathscr{C} : \mathbb{R}^d \to \mathbb{R}^{2 \times m}$ and use the same loss from Eq. 3. For a regression task, we replace $\mathcal{L}_{cce}^{j}$ with $\mathcal{L}_{L2}^{j,i} = \|y_i - C_j(M(s_i))\|_2^2$.

The second loss term enforces orthogonality between the set of classifiers, for each class separately, using the multiverse loss:

$$\mathcal{L}_{mv} = \Sigma_{j,r,s>r} \left| f_j^{r\top} f_j^{s} \beta_r \beta_s \right| \tag{4}$$

where f_j^r is the jth column of the weight matrix that corresponds to classifier C_r.

The total loss $\mathcal{L}_{total}$ is defined as: $\mathcal{L}_{total} = \mathcal{L}_{task} + \lambda \mathcal{L}_{mv}$. Similar to (Littwin and Wolf, 2016), we set $\lambda = 0.005$, throughout all of our experiments. The MML model is illustrated in Fig. 1.

The training algorithm begins with an initialization of the aggregated model $(M, \mathscr{C}, B)$. M may be initialized by any pre-trained general language model. The multiverse classifiers are randomly initialized from scratch, and all classifiers are initially activated, by setting $\beta_j = 1$ for $\forall \beta_j \in B$.

During training, we track the performance of each multiverse classifier separately. Every K steps, we search for a subset of the top-performing classifiers. When we find such a subset, we eliminate the less performing classifiers by setting their corresponding βs to 0.

In order to detect the top-performing classifiers, we calculate a moving average variable a_j for each multiverse classifier. Specifically, a_j holds the moving average of the task loss value $\mathcal{L}^j_{task}$ associated with classification head C_j. a_j is being updated for every training step, using the moving average momentum constant of 0.99.

Every K steps, we run the MeanShift algorithm (Comaniciu and Meer, 2002) on the set $\{a_j | \beta_j = 1\}$ to obtains the modes of the underlying distribution. MeanShift is a clustering algorithm that reveals the number of clusters in a given data, and retrieves the corresponding centroid for each detected cluster. In our case, MeanShift is applied on 1D data and by utilizing it, we identify the subset of top-performing multiverse heads as the cluster associated with the minimal centroid value. Next, we eliminate the rest of the multiverse heads, by setting their corresponding β to 0. This adaptive elimination is stopped, when we reach a minimal number of active heads, see Alg. 1.

At inference, we use the active multiverse heads to retrieve predictions. Specifically, given a sample s_i, we calculate the logits $\hat{y}$ as:

$$\hat{y} := \frac{\sum_{j=1}^{m} C_j(M(s_i)) \cdot \beta_j}{\sum_{j=1}^{m} \beta_j} \tag{5}$$

for classification tasks, we apply the softmax function on $\hat{y}$, and return its output. For regression tasks, we simply return $\hat{y}$.

4 Results

In this study, we evaluate MML, applied on a pretrained BERT (Devlin et al., 2018) model, using nine NLP datasets, while employing two different settings: (1) fine-tuning on different downstream tasks from the GLUE benchmark (Wang et al., 2018), and (2) cross dataset evaluations for different datasets of the same or similar tasks. For the first, we fine-tune MML on each dataset separately, and evaluate its performance on the development set and the test set of the same dataset. For the second, we evaluate our fine-tuned MML models on the train and development set of other datasets within the same task category. This allows us to study the generalization level of all models, across different datasets. Additionally, we perform an ablation study and report empirical results that showcase the efficacy of MML, compared to other multiverse schemes and to a BERT model with a higher dropout rate.

We adopt eight datasets from the GLUE benchmark (Wang et al., 2018) and one extra dataset supporting the task of Natural Language Inference (NLI). The datasets can be arranged to form the following three categories: (1) Inference tasks: this category incorporates four datasets of natural language inference: RTE, MNLI, SNLI and QNLI. RTE and QNLI are binary classification tasks, whereas MNLI and SNLI are multilabel classification tasks (which possess an additional "neutral" label). (2) Similarity and paraphrasing tasks: this category includes three datasets: MRPC, QQP, STS-B. MRPC and QQP are binary classifications tasks, while STS-B is a regression task with labels annotating the level of similarity between each sentence pair. (3) Misc. datasets: this category composed of two datasets that cannot be used for the cross dataset evaluation, due to the lack of commonality between their tasks: CoLA and SST-2. We refer the reader to the appendix for more details on the nine datasets.

4.1 Evaluation on GLUE datasets

We evaluated MML on the eight different datasets from the GLUE benchmark, and compared to BERT (Devlin et al., 2018). Each model was trained and evaluated on a single dataset. Development and test set performance are being reported for each model. In addition, we conduct an ablation analysis for our method, presenting the importance of our Maximal Multiverse Learning, which allows the training to adapt the number of multiverse classifiers to each dataset. The first ablation experiment disables the classifier elimination step during training and utilizes the same MML architecture with a fixed number of heads. The second ablation also disables the multiverse loss, leaving the training to optimize an ensemble of classifiers, without enforcing the or-

	Model	**MNLI**	**QQP**	**QNLI**	**SST-2**	**CoLA**	**STS-B**	**MRPC**	**RTE**	**Average**
		392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
development set	BERT	86.6/-	91.3	92.3	93.2	60.6	90.0	88.0	70.4	84.0
	MV-5	87.0/-	91.4	92.2	**94.0**	64.3	**91.1**	88.0	75.4	85.4
	MV-1024	86.2/-	91.4	92.2	93.2	57.9	90.6	**89.7**	**80.1**	85.2
	BERT-Drop	87.0/-	91.6	**93.0**	92.9	62.3	90.7	87.5	76.5	85.2
	BERT-Ensemble	86.6/-	91.4	92.3	93.2	60.9	90.2	88.0	70.4	84.1
	MML	**87.2/-**	**91.7**	**93.0**	**94.0**	**64.5**	**91.1**	89.0	**80.1**	**86.3**
test set	BERT	86.7/85.9	89.3	92.7	**94.9**	**60.5**	86.5	85.4	70.1	83.2
	MV-5	86.4 /85.9	88.9	92.2	94.1	56.9	87.4	86.3	70.6	82.8
	MV-1024	87.0/85.9	89.1	92.2	93.8	54.8	86.8	86.1	**74.2**	82.9
	BERT-Drop	86.6/86.0	89.3	**92.8**	94.1	56.1	87.9	85.3	74.1	83.2
	BERT-Ensemble	86.6/86.0	89.2	92.3	94.1	60.1	86.4	86.1	70.4	83.1
	MML	**87.0/86.0**	**89.4**	92.6	94.6	58.6	**88.1**	**86.7**	**74.2**	**83.8**
	MML #heads	19	14	23	979	31	45	913	1024	-

Table 1: Results on GLUE benchmarks (Wang et al., 2018). BERT results taken from (Devlin et al., 2018). Accuracy scores are reported for all datasets, except STS-B, for which Spearman Correlation is reported. For MNLI, accuracy scores are reported for both matched and mismatched test sets. The last row exhibits the number of active multiverse heads of the converged MML model. For example, for MRPC, our MML model used 913 active multiverse heads, while for MNLI, it maintained 19.

thogonality constraint.

The BERT model used is the BERT-Large model (Devlin et al., 2018). It contains 24 attention layers, each with 16 attention heads with a hidden layer of size $d = 1024$ dimensions. The model was pre-trained using sentence pairs, to both reconstruct masked words and to predict whether the pairs are consecutive. BERT's fine-tuning for downstream tasks employs supervision obtained by the given labels of each dataset.

MML utilizes the same pre-trained BERT-Large model and is initialized with $m = d = 1024$ active multiverse classifiers. During training, the model converges to a smaller number of multiverse classifiers. The number of active multiverse classifiers of each model is presented in last row of Tab. 1.

Tab. 1 presents the results for six models: (1) BERT (as a baseline), (2) MML, (3-4) two multiverse models utilizing a fixed number of multiverse classifiers, with 5 and 1024 classifiers, respectively, (5) BERT-Drop, a BERT model that was fine-tuned with a 30% dropout rate on the CLS embedding vector and which provides a baseline with additional regularization, (6) BERT-Ensemble, a BERT model with 1024 heads (similar to MV-1024 without the multiverse loss).

As can be seen in the table, compared to BERT, MML yields better results by a sizeable margin on the test set of five out of eight datasets. The largest gains were reported in the relatively smaller-sized dataests, such as RTE, MRPC and STS-B, for whom MML yields an absolute improvement of 4.1%, 1.3%, 1.6%, respectively. This can be attributed to the ability of MML to encourage a more robust learning. On the rest of the datasets, MML yields similar performance on the test. On the development set, MML outperforms BERT on all datasets, sometimes by a large margin. Specifically, for RTE, MML yields an improvement of almost 10%.

The ablation models MV-5 and MV-1024, utilize a fixed number of multiverse heads during the entire training. We have found that this hyper parameter can be crucial for the model's convergence, and when not initialized properly, may significantly reduce performance for the given task in hand. Specifically, for the CoLA dataset, MV-1024 and MV-5 yield a relative performance gap of more than 11%, in favor of MV-5, while in RTE, there is a gap of 6.2% in favor of MV-1024. When comparing both MV-5 and MV-1024 to MML, MML produces better or similar performance on the development set of all datasets. More specifically, on RTE and MRPC, MML yields similar performance as in MV-1024, and outperforms it on all the other six datasets. Compared to MV-5, MML yields better performance, by a sizeable margin, on four datasets out of eight, and produces similar performance on the rest.

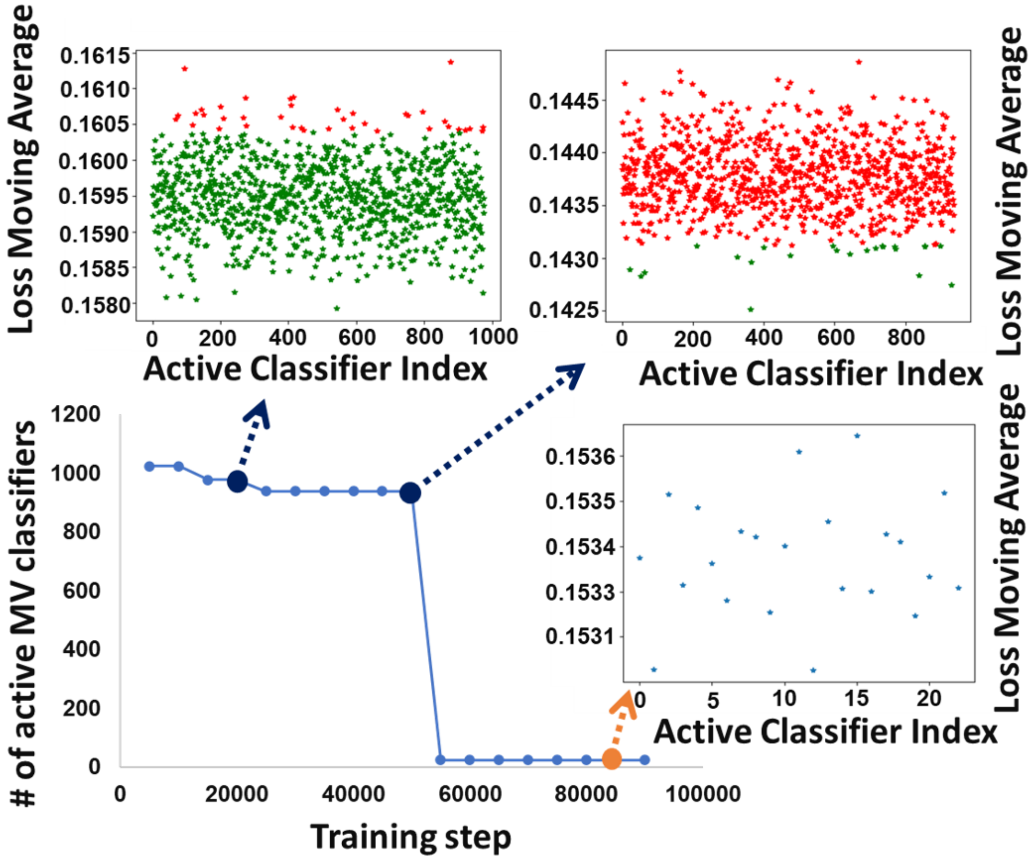


Figure 2: The number of active multiverse classifiers, per training step, for the MML model trained on QNLI. The MeanShift algorithm detects multiple clusters, for three times during training. The two upper plots present the selection of the top-performing heads (green stars), and the elimination of the heads showing lower-performance (red stars). The Y values are the moving average calculated on the multiverse heads' loss function. Our MML-QNLI model reaches local minima at step 85K, for which 23 heads were activated. The bottom right plot shows the moving average values of the activated 23 multiverse heads, by the same training step.

The BERT-Drop model is a BERT baseline trained with a dropout rate of 30% (instead of 10% as used in a regular BERT). Compared to BERT, BERT-Drop provides similar results on the test set, while showing some improvement on the development set. When comparing BERT-Drop to MML, MML results with a higher average performance on both the test set and development set, where in some dataests, such as RTE, MML yields an improvement of +3.6% on the development set. On the test set, MML surpasses BERT-Drop on seven out of eight datasets. Specifically, in some datasets, such as MRPC and CoLA, MML outperforms BERT-Drop in 1.4 and 2.5 accuracy points, respectively.

The BERT-Ensemble employs BERT with an ensemble of 1024 parallel classifiers. It optimizes all classifiers during training and infers predictions by calculating the mean over all classifiers' logits. As shown in the table, BERT-Ensemble yields similar performance to BERT.

Fig. 2 presents the amount of active multiverse heads, when applying MML on QNLI dataset. During the training of the MML-QNLI model, the MeanShift algorithm detected multiple clusters at three times[1], through the entire training. Each time, the model eliminated the lower-performance subsets, and kept the top-performing multiverse classifiers as the active set of classifiers. The model achieved best performance on the development set at training step 85K. At this step, the MML-QNLI model utilized 23 active multiverse heads. The plots in the figure present the cumulative loss of each multiverse head, sorted through the X axis according to the indices of the active heads. The red stars are asso-

---

[1] The elimination is being invoked every time the MeanShift algorithm detects multiple clusters. Specifically, for MML-QNLI experiment, multiple clusters appeared three times during the training process.

ciated with the classifiers heads that were eliminated, and the green stars are the heads that were selected as the top-performing subset.

4.2 Cross dataset evaluations

In the cross-task evaluations, we use the finetuned MML models from Sec. 4.1. For each model trained on a dataset from the two first categories above (NLI and similarity/paraphrasing), we evaluate the model on all datasets from the same category. Train and development set performance are reported to give a clear view on the cross-task generalization ability, and also to exhibit the level of overfitting, when evaluating on the same dataset that the model was trained on.

In order to conduct a clean comparison, we finetune BERT, MML, and all other methods with the same hyperparmeters, employing 10/30/100 epochs for the relative large/medium/small datasets, a batch size of 32, and a learning rate of 2e-5 (BERT obtains in all cases performance that at least matches the one published in (Devlin et al., 2018)). Our code can be found at `https://github.com/ItzikMalkiel/MML`.

In cross evaluation experiments, there is no training on the target training set, so both the target training set and validation sets can be used for evaluation. Overall, there are 36 cross-task experiments. In the first category, there are four datasets, so three cross-task experiments for each. Taking into account the two splits, this amounts to 24 experiments. Similarly, in the second category, where there are three datasets, there are 12 experiments. Fig. 3 is a Dolan Moré plot comparing the performance of BERT to the five finetuning variants explored (MML, MV-5, MV-1024, BERT-Drop, Bert-Ensable) for all 36 experiments. These plots show for each method the ratio of experiments for which it has a performance level that is at least as high a constant times the best result. As can be seen, regularization helps cross-task generality. However, none of the baseline methods is as effective as MML. In the supplementary appendix we provide the full data, and, for brevity, below we focus on comparing MML with BERT.

Inference tasks Since MNLI and SNLI are multiclass classification tasks with 3 classes, we collapse the labels "neutral" and "contradication" into one label ("non entailment"). This modification, applied only during inference, allows us to evaluate MNLI and SNLI models on RTE and QNLI, and vice versa.

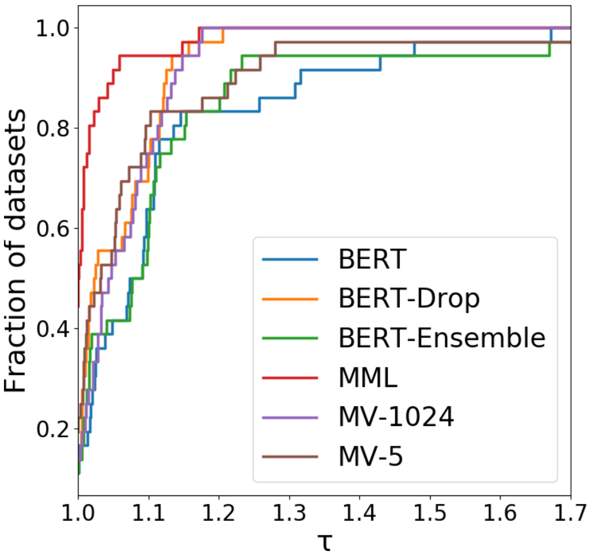


Figure 3: A Dolan-Moré profile, based on the results obtained across all cross-task experiments. The x-axis is the threshold τ. The y-axis depicts, for a given fine-tuning method, the ratio of datasets in which the obtained error is less than the threshold τ times the minimal error obtained by any of the six methods.

The results are reported in Tab. 2. As can be seen, MML exhibits a significantly improved transferability compared to BERT. Each row in the table represents an MML or BERT model trained on a single dataset associated by its name. All models are evaluated on all four datasets. In the last column, we report the relative average improvement obtained by MML, calculated by the performance ratio between MML and BERT across all three holdout datasets. For example, for RTE, our MML-RTE model yields 9.9% relative average performance on the train set of MNLI, QNLI and SNLI, and a 9.5% average improvement on the development set of these datasets.

Similarity and paraphrasing Since STS-B is a regression task benchmark, while MRPC and QQP address a binary classification task, to support cross dataset evaluations, we adapt STS-B to form a binary classification task. The adaptation is being done by collapsing the labels in the range 1-2 (4-5) to the value of 0 (1) and omitting all samples associated with label values between 2 and 4. The binary STS-B version has ~3.5K samples.

As can be seen in Tab. 3, MML yields better performance on the cross evaluations for the similarity and paraphrase datasets. Similar to Tab. 2, each row represent a single model trained on a single dataset. We evaluate all models on all three

Model	RTE	MNLI	QNLI	SNLI	MML Improvement
BERT-RTE	96.06/70.39	69.42/69.17	52.46/52.84	68.02/69.87	-
MML-RTE	99.39/**80.14**	79.24/78.42	50.86/51.30	80.85/82.53	+9.98%/+9.51%
BERT-MNLI	79.15/76.89	99.59/86.58	49.88/51.05	81.65/83.65	-
MML-MNLI	79.35/78.70	99.74/**87.18**	49.64/50.22	82.37/83.94	+0.21% /+0.35%
BERT-QNLI	53.37/48.73	59.76/59.89	99.99/**94.01**	59.33/60.03	-
MML-QNLI	53.41/53.79	64.93/63.85	95.75/92.86	62.13/63.58	+4.48%/+7.63%
BERT-SNLI	71.28/70.03	75.82/75.79	49.62/50.72	99.74/91.08	-
MML-SNLI	71.88/69.31	75.79/76.37	49.32/50.32	99.30/**91.38**	+0.07%/-0.36%

Table 2: Cross dataset evaluation for Language Inference tasks. Train/development accuracy are reported separately for each dataset. Each model (a row in the table) was trained on a single dataset denoted by its name, and was evaluated on the train/development sets of all four datasets. The last columns indicates the relative average improvement obtained by MML compared to BERT, and averaged across the three hold-out datasets (i.e. excluding the diagonal). SNLI is the only dataset for which MML does not improve cross dataset performance on average, perhaps since it is largest dataset with 570k samples. See supplementary appendix for farther comparison with MV-5, MV-1024 and BERT-Drop.

Model	QQP	MRPC	STS-B*	MML Improvement
BERT-QQP	99.73/91.57	66.90/68.85	88.34/90.12	-
MML-QQP	99.74/**91.68**	67.77/68.87	89.11/90.55	+1.08%/+0.25%
BERT-MRPC	65.28/65.18	99.37/87.25	82.53/88.58	-
MML-MRPC	68.37/68.15	99.23/**88.97**	86.12/91.32	+4.54%/+3.82%
BERT-STS-B*	73.13/73.11	75.59/75.49	100.0/95.49	-
MML-STS-B*	74.13/74.40	75.51/77.94	99.85/**96.70**	+0.63%/+2.50%

Table 3: Cross dataset evaluation for similarity and paraphrasing tasks. STS-B* is the modified version of STS-B that forms a binary classification dataset (instead of regression). STS-B* models were trained as binary classifiers on STS-B data. Accuracy scores are reported through all evaluations. The last column presents the relative cross dataset improvement obtained by MML, compared to BERT.

datasets, and report the average relative improvement obtained by MML calculated on the two holdout datasets. We have found MML to produce improved performance for all models, for example, MML-MRPC yields a ~+3.5% average improvement calculated on both train and development sets across STS-B and QQP.

5 Discussion

The results in both Tab. 2 and 3, reveal a significant gap in performance for all models when evaluated on holdout datasets, although the holdout datasets share the same or similar task each model was trained for. For example, both MML-MRPC and BERT-MRPC models yield a ~20% degradation in absolute accuracy on the QQP dataset. However, compared to BERT, our MML method produces significantly better performance on the cross evaluations. Specifically, when evaluated on QQP, MML-MRPC outperforms BERT-MRPC by a relative improvement of ~4.6%, for both development and train set.

We do not observe a direct link between the improvement obtained on the same dataset evaluation to that obtained in the cross dataset one. For example, our MML-QNLI model was able to outperform BERT-QNLI in the cross dataset evaluation, although, compared to our BERT-QNLI reproduction, MML-QNLI exhibits a somewhat degraded performance on QNLI's development set (see the third section in Tab.2) and test set (as published in (Devlin et al., 2018)). We attribute this to the ability of MML to encourage the model to produce more transferable coding vectors.

Computational overhead During training, the MML multiverse heads imply a maximal addition of 1024 operations of matrix multiplications and gradient derivations, each in the size of $1024 \times c$.

During the backward steps, the multiple gradients are averaged almost immediately, right before propagating them back through the last encoder layer of BERT. Interestingly, in many cases, the number of active heads shrinks in the early stages of the training. During inference, the additional calculations are solely the matrix multiplications, for which the average number of active heads, across our experiments, is 381. This average number of active heads translates to an increase of ~0.2% to the parameter count of BERT_Large.

6 Summary

We introduce the MML method for fine-tuning language models. MML utilizes a large set of parallel multiverse heads and eliminates the relatively weaker heads during training. We demonstrate the effectiveness of MML on nine common NLP datasets, by applying inter- and intra-datasets evaluation, where it is shown to outperform the originally introduced BERT fine-tuning procedure. The results shade light on the role of regularization in improving cross task generalization, and show the advantage of MML over alternative regularization methods.

Acknowledgment

This project has received funding from the European Research Council (ERC) under the European Union Horizon 2020 research and innovation programme (grant ERC CoG 725974). The contribution of the first author is part of a PhD thesis research conducted at Tel Aviv University.

References

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs.

Dorin Comaniciu and Peter Meer. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pages 603–619.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Etai Littwin and Lior Wolf. 2016. The multiverse loss for robust transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3957–3966.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *openai blog*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru

Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Supplementary Appendices[2]

A The datasets (more details)

We adopt eight datasets from the GLUE benchmark (Wang et al., 2018), and one extra dataset supporting the task of Natural Language Inference (NLI). The datasets can be arranged by categories as follows.

Inference tasks This category contains three datasets from the GLUE benchmark (Wang et al., 2018), along with an external dataset named SNLI (Bowman et al., 2015) that shares the same task. **RTE** The Recognizing Textual Entailment dataset (Bentivogli et al., 2009) is composed of sentence pairs associated with a binary classification task for entailment/non entailment relation between the sentences. **MNLI** Multi-Genre Natural Language Inference Corpus (Williams et al., 2017) is also an entailment dataset comprised of sentence pairs. The task is multiclass classification for predicting contradiction, neutral or entailment relation between the sentence pairs. **SNLI** The Stanford Natural Language Inference dataset (Bowman et al., 2015) is similar to MNLI, but contains data gathered from different sources. **QNLI** The Question-answering Natural Language Inference dataset (Rajpurkar et al., 2016) contains question-sentence pairs associated with the binary classification task for the entailment/non entailment relation between the question-answer pairs.

Similarity and paraphrasing Tasks This category contains three datasets. **MRPC** Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005) is a dataset of sentence pairs. The task is to determine whether a pair of sentences are semantically equivalent (binary classification). **QQP** Quora Question Pairs (Chen et al., 2018) is a dataset of question pairs taken from the Quora website. The goal is to determine whether a pair of questions are semantically equivalent (binary classification). **STS-B** Semantic Textual Similarity Benchmark (Cer et al., 2017) is a dataset composed of sentence pairs, each annotated with a score between 1 and 5, indicating the semantic similarity level of both sentences. The task is to predict these scores (regression).

Misc. datasets The two datasets in this category are not used for the cross dataset evaluation, due to the lack of commonality between their tasks. **CoLA** The Corpus of Linguistic Acceptability dataset (Warstadt et al., 2018) consists of examples of expert English sentence acceptability judgments. Each sample is annotated by whether it is a grammatically sentence of English (binary classification). **SST-2** The Stanford Sentiment Treebank (Socher et al., 2013) is a dataset composed of sentences assigned with a human annotations of their sentiment. The task is to determine whether the sentiment of each sentence is positive or negative (binary classification).

B Cross-task generalization results for all baselines

For the sake of completeness, we report cross dataset evaluations, for all six models described in the main text. Specifically, we compare MML, BERT, MV-5, MV-1024, BERT-Drop and BERT-Ensemble on cross dataset evaluations for inference datasets, and similarity and paraphrase datasets.

Tab. 4 presents the performance of the above six models evaluated on the cross dataset evaluation for the NLI category. Each row represents a different model, trained on a single dataset associated by its name. Each column corresponds to an evaluation on four datasets in this category. The last column presents the improvement obtained by each model, compared to BERT, and calculated across the three hold-out datasets. Accuracy is reported for all datasets on both the train and development sets, in the format train

---

[2] Put here for the reader's convenience.

accuracy/development accuracy. Note that no training is done on the training set in the case of cross-task evaluation. Therefore, the training set can be seen as an additional dataset to evaluate on.

As can be seen in Tab. 4, MML-RTE outperforms BERT-RTE on all datasets, and results with similar performance compared to MV-5-RTE, MV-1024-RTE and BERT-Drop-RTE. For the models trained with MNLI dataset, we found that the four models MML-MNLI, MV-5-MNLI, MV-1024-MNLI and BERT-Drop-MNLI, yield similar results, where all produce a slight improvement compared to BERT. For QNLI, MML yields a significant improvement compared to all models. Specifically, MML-QNLI yields +4.48%/7.63% relative improvement compared to BERT, averaged on the train and development set of the three hold-out dataests. BERT-Drop-QNLI yields significantly decreased performance of -1.93%/-2.89% compared to BERT, MV-5 yields improved performance of +1.0%/+2.73%, and MV-1024 results with a significantly decreased performance of -2.64% on the train sets, and somehow similar performance to BERT on the development set.

For SNLI, MML and BERT-Drop yield similar performance compared to BERT on the training sets, where MML shows slightly better performance. On the development set, both MML and Bert-Drop present slightly decreased performance. Mv-5 and Mv-1024 exhibit significantly decreased performance on both train and development sets.

In Tab.5, we present the performance of all six fine-tune variants, evaluated on the cross similarity and paraphrasing datasets. For QQP-based models, MML and MV-5 were able to improve the cross evaluation performance compared to BERT, measured on the two hold-out datasets. Both MML-QQP and MV-5-QQP yield more than one percentage of relative improvement on the train sets, and less than one percentage of improvement on the developments sets, where MV-5-QQP presents slightly better results. On the other hand, BERT-Drop-QQP and MV-1024-QQP yield significantly decreased performance compared to BERT, each results with 1.76%-3.22% of decreased performance on the train/development sets.

For MRPC, MML outperforms all models, showing a significantly improvement of +4.54%/3.82% on the train/development sets, when compared to BERT. The other models where able to improve BERT in +2.69% up to +3.89% on the train and +1.66% up to +2.95% on the development sets, which are significantly inferior to MML-MRPC. For STS-B, we see that MML, MV-5 and BERT-Drop were able to yield similar improvement compared to BERT, where MV-1024 yields decreased performance on the train sets, and similar performance on the development set.

All in all, MML gained the highest improvement, accumulated across all evaluations. Specifically, when averaging the cross evaluation results over the train sets, MML shows an average improvement of +3.0%, BERT-Drop shows average improvement of +1.64%, MV-5 yields +2.27% and MV-1024 yields +0.46%.

Model	**RTE**	**MNLI**	**QNLI**	**SNLI**	**Improvement compared to BERT**
BERT-RTE	96.06/70.39	69.42/69.17	52.46/52.84	68.02/69.87	-
MML-RTE	99.39/**80.14**	79.24/78.42	50.86/51.30	80.85/82.53	+9.98%/+9.51%
MV-1024-RTE	99.39/**80.14**	79.24/78.42	50.86/51.30	80.85/82.53	+9.98%/+9.51%
MV-5-RTE	100.0/75.45	74.50/73.73	58.22/58.72	75.48/77.15	+9.75% /+9.36%
BERT-D-RTE	100.0/76.53	79.31/78.29	52.55/52.80	79.53/81.10	+10.4% /+9.71%
BERT-Ensemble-RTE	96.06/70.39	70.31/70.02	51.25/51.89	69.54/69.97	-1.20%/+0.34%
BERT-MNLI	79.15/76.89	99.59/86.58	49.88/51.05	81.65/83.65	-
MML-MNLI	79.35/78.70	99.74/**87.18**	49.64/50.22	82.37/83.94	+0.21% /+0.35%
MV-1024-MNLI	78.91/78.70	98.71/86.24	49.93/50.48	82.33/83.60	+0.20% /+0.39%
MV-5-MNLI	78.83/77.61	98.11/87.00	49.91/50.46	82.51/83.95	+0.23% /+0.04%
BERT-D-MNLI	79.40/78.70	99.90/87.04	49.75/50.22	82.32/83.71	+0.29% /+0.26%
BERT-Ensemble-MNLI	78.68/75.58	99.59/86.58	50.21/52.21	82.51/82.35	+1.54%/+0.54%
BERT-QNLI	53.37/48.73	59.76/59.89	99.99/**94.01**	59.33/60.03	-
MML-QNLI	53.41/53.79	64.93/63.85	95.75/92.86	62.13/63.58	+4.48%/+7.63%
MV-1024-QNLI	51.04/51.26	60.28/58.6	99.49/92.24	58.68/58.78	-2.64%/+0.31%
MV-5-QNLI	51.12/50.90	61.58/60.36	97.95/92.18	61.81/61.86	+1.00%/+2.73%
BERT-D-QNLI	49.91/45.84	60.48/59.33	99.99/93.17	59.22/59.14	-1.93%/-2.89%
BERT-Ensemble-QNLI	54.52/49.84	58.84/59.25	98.95/93.01	60.35/61.87	-1.75%/-1.02%
BERT-SNLI	71.28/70.03	75.82/75.79	49.62/50.72	99.74/91.08	-
MML-SNLI	71.88/69.31	75.79/76.37	49.32/50.32	99.30/**91.38**	+0.07%/-0.26%
MV-1024-SNLI	68.87/64.98	73.41/74.09	48.89/49.33	94.54/91.16	-2.67% /-4.06%
MV-5-SNLI	71.24/67.87	75.38/75.00	49.42/50.35	95.63/91.18	-0.35% /-1.62%
BERT-D-SNLI	71.36/70.84	76.16/76.38	49.32/49.35	95.65/91.45	-1.35% /-1.24%
BERT-Ensemble-SNLI	72.52/71.89	74.75/74.85	50.54/51.02	99.57/91.2	-1.23%/-1.10%
MML impr.	+0.38%/+3.90%	+7.58%/+6.91%	-1.37%/-1.77%	+8.15%/+8.12%	
MV-1024 impr.	-2.68%/+0.11%	+3.94%/+2.99%	-1.47%/-2.25%	+6.19%/+5.32%	
MV-5 impr.	-1.55%/+0.76%	+3.26%/+2.11%	+3.54%/3.08%	+5.40%/-0.94%	
BERT-D impr.	-2.01%/-0.80%	+5.29%/+4.34%	-0.23%/-1.46%	+5.85%/+4.88%	
BERT-Ensemble impr.	-1.62%/-1.40%	+1.64%/+0.51%	-0.52%/-0.54%	-1.66%/-0.55%	

Table 4: Cross dataset evaluation for Language Inference tasks for all ablation models presented in the main text (BERT-D stands for BERT-Drop). Train/development accuracy are reported separately for each dataset. Each model (a row in the table) was trained on a single dataset denoted by its name, and was evaluated on the train/development sets of all four datasets. The last column indicates the relative average improvement obtained by each model compared to BERT, and averaged across the three hold-out datasets. The last four rows present the average column-wise improvement each technique yields, which is aggregated from the three models of the same technique. BERT models were reproduced with the same hyperparamters used for MML (all BERT reproductions result with similar or better performance compared to the one published in (Devlin et al., 2018)).

Model	**QQP**	**MRPC**	**STS-B***	**Improvement compared to BERT**
BERT-QQP	99.73/91.57	66.90/68.85	88.34/90.12	-
MML-QQP	99.74/91.68	67.77/68.87	89.11/90.55	+1.08%/+0.25%
MV-1024-QQP	98.78/91.42	65.59/67.40	88.20/90.01	-2.16%/-3.22%
MV-5-QQP	99.40/91.45	68.19/68.60	89.00/91.09	+1.58%/+0.63%
BERT-Drop-QQP	99.57/91.64	64.50/65.68	88.40/89.24	-1.76% /-2.80%
BERT-Ensemble-QQP	99.73/91.42	67.45/69.85	89.41/90.45	-1.51%/-1.18%
BERT-MRPC	65.28/65.18	99.37/87.25	82.53/88.58	-
MML-MRPC	68.37/68.15	99.23/88.97	86.12/91.32	+4.54%/+3.82%
MV-1024-MRPC	67.33/67.44	99.91/89.70	85.19/90.55	+3.18%/+2.76%
MV-5-MRPC	66.69/66.46	99.94/87.99	85.20/89.79	+2.69%/+1.66%
BERT-Drop-MRPC	67.65/67.66	99.97/87.5	85.97/90.45	+3.89% /+2.95%
BERT-Ensemble-MRPC	64.48/64.88	99.56/87.25	81.23/87.98	+1.58%/+1.27%
BERT-STS-B*	73.13/73.11	75.59/75.49	100.0/95.49	-
MML-STS-B*	74.13/74.40	75.51/77.94	99.85/96.70	+0.63%/+2.50%
MV-1024-STS-B*	73.96/74.25	75.59/77.20	99.85/96.48	-2.64%/+0.31%
MV-5-STS-B*	74.93/75.15	75.21/75.98	100.0/97.03	+0.97%/+1.71%
BERT-Drop-STS-B*	74.29/75.45	75.40/76.22	100.0/96.70	+0.66% /+2.08%
BERT-Ensemble-STS-B*	74.45/74.21	76.01/76.45	100.0/96.15	-1.27%/-1.08%
MML impr.	+3.05%/+3.16%	+0.59%/+1.63%	+2.61%/+1.78%	
MV-1024 impr.	+2.13%/+2.51%	+0.97%/+0.07%	+1.53%/+1.05%	
MV-5 impr.	+2.31%/+2.37%	+0.71%/+0.14%	+1.99%/+1.22%	
BERT-Drop impr.	+2.60%/+3.50	-1.91%/-1.81%	+2.11%/+0.56%	
BERT-Ensemble impr.	-0.31%/-1.02%	-1.42%/-1.28%	+0.21%/+0.19%	

Table 5: Cross dataset evaluation for similarity and paraphrase tasks. STS-B* is the modified version of STS-B that forms a binary classification dataset (instead of regression). STS-B* models were trained as binary classifiers on STS-B data. Accuracy scores are reported through all evaluations. The last column presents the relative cross dataset improvement obtained by each model compared to BERT, and averaged across the two hold-out datasets. The last four rows present the average column-wise improvement each technique yields, which is computed from the two models of the same technique.

Unification-based Reconstruction of Multi-hop Explanations for Science Questions

Marco Valentino[*†], **Mokanarangan Thayaparan**[*†], **André Freitas**[†‡]
Department of Computer Science, University of Manchester, United Kingdom[†]
Idiap Research Institute, Switzerland[‡]
`{marco.valentino,mokanarangan.thayaparan,andre.freitas}`
`@manchester.ac.uk`

Abstract

This paper presents a novel framework for reconstructing *multi-hop explanations* in science Question Answering (QA). While existing approaches for multi-hop reasoning build explanations considering each question in isolation, we propose a method to leverage *explanatory patterns* emerging in a corpus of scientific explanations. Specifically, the framework ranks a set of atomic facts by integrating lexical relevance with the notion of *unification power*, estimated analysing explanations for similar questions in the corpus.

An extensive evaluation is performed on the Worldtree corpus, integrating k-NN clustering and Information Retrieval (IR) techniques. We present the following conclusions: (1) The proposed method achieves results competitive with Transformers, yet being orders of magnitude faster, a feature that makes it scalable to large explanatory corpora (2) The unification-based mechanism has a key role in reducing semantic drift, contributing to the reconstruction of many hops explanations (6 or more facts) and the ranking of complex inference facts (+12.0 Mean Average Precision) (3) Crucially, the constructed explanations can support downstream QA models, improving the accuracy of BERT by up to 10% overall.

1 Introduction

Answering *multiple-choice science questions* has become an established benchmark for testing natural language understanding and complex reasoning in Question Answering (QA) (Khot et al., 2019; Clark et al., 2018; Mihaylov et al., 2018). In parallel with other NLP research areas, a crucial requirement emerging in recent years is *explainability* (Thayaparan et al., 2020; Miller, 2019; Biran and Cotton, 2017; Ribeiro et al., 2016). To boost automatic methods of inference, it is necessary not only to measure the performance on answer prediction, but also the ability of a QA system to provide explanations for the underlying reasoning process.

The need for explainability and a quantitative methodology for its evaluation have conducted to the creation of shared tasks on *explanation reconstruction* (Jansen and Ustalov, 2019) using corpora of explanations such as Worldtree (Jansen et al., 2018, 2016). Given a science question, explanation reconstruction consists in regenerating the gold explanation that supports the correct answer through the combination of a series of atomic facts. While most of the existing benchmarks for multi-hop QA require the composition of only 2 supporting sentences or paragraphs (e.g. QASC (Khot et al., 2019), HotpotQA (Yang et al., 2018)), the explanation reconstruction task requires the aggregation of an average of 6 facts (and as many as $\approx$20), making it particularly hard for multi-hop reasoning models. Moreover, the structure of the explanations affects the complexity of the reconstruction task. Explanations for science questions are typically composed of two main parts: a grounding part, containing knowledge about concrete concepts in the question, and a core scientific part, including general scientific statements and laws.

Consider the following question and answer pair from Worldtree (Jansen et al., 2018):

- q: what is an example of a **force** producing heat?
  a: two **sticks** getting warm when **rubbed together**.

An explanation that justifies a is composed using the following sentences from the corpus: *(f_1) a **stick** is a kind of object*; *(f_2) to **rub together** means to move against*; *(f_3) friction is a kind of **force***; *(f_4) friction occurs when two objects' surfaces move against each other*; *(f_5) friction causes the temperature of an object to increase*. The explanation

---

[*] `equal contribution`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 200–211
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

contains a set of concrete sentences that are conceptually connected with q and a (f_1,f_2 and f_3), along with a set of abstract facts that require multi-hop inference (f_4 and f_5).

Previous work has shown that constructing long explanations is challenging due to *semantic drift* – i.e. the tendency of composing out-of-context inference chains as the number of hops increases (Khashabi et al., 2019; Fried et al., 2015). While existing approaches build explanations considering each question in isolation (Khashabi et al., 2018; Khot et al., 2017), we hypothesise that semantic drift can be tackled by leveraging *explanatory patterns* emerging in clusters of similar questions.

In Science, a given statement is considered explanatory to the extent it performs *unification* (Friedman, 1974; Kitcher, 1981, 1989), that is showing how a set of initially disconnected phenomena are the expression of the same regularity. An example of unification is Newton's law of universal gravitation, which *unifies* the motion of planets and falling bodies on Earth showing that *all* bodies with mass obey the same law. Since the explanatory power of a given statement depends on the number of unified phenomena, highly explanatory facts tend to create *unification patterns* – i.e. similar phenomena require similar explanations. Coming back to our example, we hypothesise that the relevance of abstract statements requiring multi-hop inference, such as f_4 (*"friction occurs when two objects' surfaces move against each other"*), can be estimated by taking into account the unification power.

Following these observations, we present a framework that ranks atomic facts through the combination of two scoring functions:

- A *Relevance Score (RS)* that represents the lexical relevance of a given fact.
- A *Unification Score (US)* that models the explanatory power of a fact according to its frequency in explanations for similar questions.

An extensive evaluation is performed on the Worldtree corpus (Jansen et al., 2018; Jansen and Ustalov, 2019), adopting a combination of k-NN clustering and Information Retrieval (IR) techniques. We present the following conclusions:

1. Despite its simplicity, the proposed method achieves results competitive with Transformers (Das et al., 2019; Chia et al., 2019), yet being orders of magnitude faster, a feature that makes it scalable to large explanatory corpora.
2. We empirically demonstrate the key role of the unification-based mechanism in the reconstruction of many hops explanations (6 or more facts) and explanations requiring complex inference (+12.0 Mean Average Precision).
3. Crucially, the constructed explanations can support downstream question answering models, improving the accuracy of BERT (Devlin et al., 2019) by up to 10% overall.

To the best of our knowledge, we are the first to propose a method that leverages unification patterns for the reconstruction of multi-hop explanations, and empirically demonstrate their impact on semantic drift and downstream question answering.

2 Related Work

Explanations for Science Questions. Reconstructing explanations for science questions can be reduced to a multi-hop inference problem, where multiple pieces of evidence have to be aggregated to arrive at the final answer (Thayaparan et al., 2020; Khashabi et al., 2018; Khot et al., 2017; Jansen et al., 2017). Aggregation methods based on lexical overlaps and explicit constraints suffer from *semantic drift* (Khashabi et al., 2019; Fried et al., 2015) – i.e. the tendency of composing spurious inference chains leading to wrong conclusions.

One way to contain semantic drift is to leverage common explanatory patterns in explanation-centred corpora (Jansen et al., 2018). Transformers (Das et al., 2019; Chia et al., 2019) represent the state-of-the-art for explanation reconstruction in this setting (Jansen and Ustalov, 2019). However, these models require high computational resources that prevent their applicability to large corpora. On the other hand, approaches based on IR techniques are readily scalable. The approach described in this paper preserves the scalability of IR methods, obtaining, at the same time, performances competitive with Transformers. Thanks to this feature, the framework can be flexibly applied in combination with downstream question answering models.

Our findings are in line with previous work in different QA settings (Rajani et al., 2019; Yadav et al., 2019), which highlights the positive impact of explanations and supporting facts on the final answer prediction task.

In parallel with Science QA, the development of models for explanation generation is being explored in different NLP tasks, ranging from open domain question answering (Yang et al., 2018; Thayaparan et al., 2019), to textual entailment (Camburu et al., 2018) and natural language premise selection (Ferreira and Freitas, 2020b,a).

Scientific Explanation and AI. The field of Artificial Intelligence has been historically inspired by models of explanation in Philosophy of Science (Thagard and Litt, 2008). The deductive-nomological model proposed by Hempel (Hempel, 1965) constitutes the philosophical foundation for explainable models based on logical deduction, such as Expert Systems (Lacave and Diez, 2004; Wick and Thompson, 1992) and Explanation-based Learning (Mitchell et al., 1986). Similarly, the inherent relation between explanation and causality (Woodward, 2005; Salmon, 1984) has inspired computational models of causal inference (Pearl, 2009). The view of explanation as unification (Friedman, 1974; Kitcher, 1981, 1989) is closely related to Case-based reasoning (Kolodner, 2014; Sørmo et al., 2005; De Mantaras et al., 2005). In this context, analogical reasoning plays a key role in the process of reusing abstract patterns for explaining new phenomena (Thagard, 1992). Similarly to our approach, Case-based reasoning applies this insight to construct solutions for novel problems by retrieving, reusing and adapting explanations for known cases solved in the past.

3 Explanation Reconstruction as a Ranking Problem

A multiple-choice science question $Q = \{q, C\}$ is a tuple composed by a question q and a set of candidate answers $C = \{c_1, c_2, \ldots, c_n\}$. Given an hypothesis h_j defined as the concatenation of q with a candidate answer $c_j \in C$, the task of explanation reconstruction consists in selecting a set of atomic facts from a knowledge base $E_j = \{f_1, f_2, \ldots, f_n\}$ that support and justify h_j.

In this paper, we adopt a methodology that relies on the existence of a corpus of explanations. A corpus of explanations is composed of two distinct knowledge sources:

- A primary knowledge base, *Facts KB* (F_{kb}), defined as a collection of sentences $F_{kb} = \{f_1, f_2, \ldots, f_n\}$ encoding the general world knowledge necessary to answer and explain science questions. A fundamental and desirable characteristic of F_{kb} is *reusability* – i.e. each of its facts f_i can be potentially reused to compose explanations for multiple questions
- A secondary knowledge base, *Explanation KB* (E_{kb}), consisting of a set of tuples $E_{kb} = \{(h_1, E_1), (h_2, E_2), \ldots, (h_m, E_m)\}$, each of them connecting a true hypothesis h_j to its corresponding explanation $E_j = \{f_1, f_2, \ldots, f_k\} \subseteq F_{kb}$. An explanation $E_j \in E_{kb}$ is therefore a composition of facts belonging to F_{kb}.

In this setting, the explanation reconstruction task for an unseen hypothesis h_j can be modelled as a ranking problem (Jansen and Ustalov, 2019). Specifically, given an hypothesis h_j the algorithm to solve the task is divided into three macro steps:

1. Computing an explanatory score $s_i = e(h_j, f_i)$ for each fact $f_i \in F_{kb}$ with respect to h_j
2. Producing an ordered set $Rank(h_j) = \{f_1, \ldots, f_k, f_{k+1}, \ldots, f_n \mid s_k \geq s_{k+1}\} \subseteq F_{kb}$
3. Selecting the top k elements belonging to $Rank(h_j)$ and interpreting them as an explanation for h_j; $E_j = topK(Rank(h_j))$.

3.1 Modelling Explanatory Relevance

We present an approach for modelling $e(h_j, f_i)$ that is guided by the following research hypotheses:

- **RH1**: Scientific explanations are composed of a set of concrete facts connected to the question, and a set of abstract statements expressing general scientific laws and regularities.
- **RH2**: Concrete facts tend to share key concepts with the question and can therefore be effectively ranked by IR techniques based on lexical relevance.
- **RH3**: General scientific statements tend to be abstract and therefore difficult to rank by means of shared concepts. However, due to explanatory unification, core scientific facts tend to be frequently reused across similar questions. We hypothesise that the explanatory power of a fact f_i for a given hypothesis h_j is proportional to the number of times f_i explains similar hypotheses.

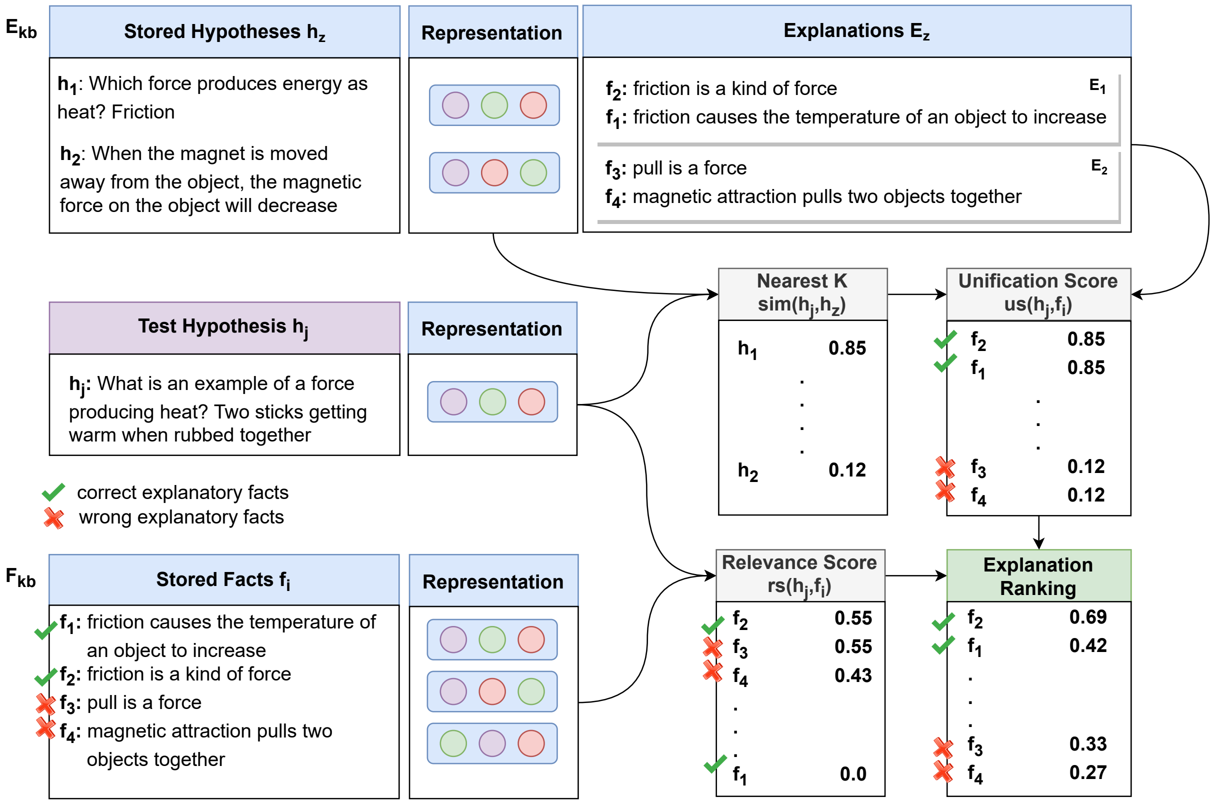


Figure 1: Overview of the Unification-based framework for explanation reconstruction.

To formalise these research hypotheses, we model the explanatory scoring function $e(h_j, f_i)$ as a combination of two components:

$$e(h_j, f_i) = \lambda_1 rs(h_j, f_i) + (1 - \lambda_1) us(h_j, f_i) \quad (1)$$

Here, $rs(h_j, f_i)$ represents a lexical Relevance Score (RS) assigned to $f_i \in F_{kb}$ with respect to h_j, while $us(h_j, f_i)$ represents the Unification Score (US) of f_i computed over E_{kb} as follows:

$$us(h_j, f_i) = \sum_{(h_z, E_z) \in kNN(h_j)} sim(h_j, h_z) in(f_i, E_z) \quad (2)$$

$$in(f_i, E_z) = \begin{cases} 1 & \text{if } f_i \in E_z \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

$kNN(h_j) = \{(h_1, E_1), \dots (h_k, E_k)\} \subseteq E_{kb}$ is the set of k-nearest neighbours of h_j belonging to E_{kb} retrieved according to a similarity function $sim(h_j, h_z)$. On the other hand, $in(f_i, E_z)$ verifies whether the fact f_i belongs to the explanation E_z for the hypothesis h_z.

In the formulation of Equation 2 we aim to capture two main aspects related to our research hypotheses:

1. The more a fact f_i is reused for explanations in E_{kb}, the higher its explanatory power and therefore its Unification Score;
2. The Unification Score of a fact f_i is proportional to the similarity between the hypotheses in E_{kb} that are explained by f_i and the unseen hypothesis (h_j) we want to explain.

Figure 1 shows a schematic representation of the Unification-based framework.

4 Empirical Evaluation

We carried out an empirical evaluation on the Worldtree corpus (Jansen et al., 2018), a subset of the ARC dataset (Clark et al., 2018) that includes explanations for science questions. The corpus provides an explanatory knowledge base (F_{kb} and E_{kb}) where each explanation in E_{kb} is represented as a set of lexically connected sentences describing how to arrive at the correct answer. The science questions in the Worldtree corpus are split into *training-set*, *dev-set*, and *test-set*. The gold explanations in the *training-set* are used to form the Explanation KB (E_{kb}), while the gold explanations in *dev* and *test* set are used for *evaluation purpose only*. The corpus groups the explanation sentences belonging to E_{kb} into three explanatory roles: *grounding*, *central* and *lexical glue*.

Consider the example in Figure 1. To support

Model	**Approach**	**Trained**	**MAP**	
			Test	Dev
Transformers				
Das et al. (2019)	BERT re-ranking with inference chains	Yes	**56.3**	**58.5**
Chia et al. (2019)	BERT re-ranking with gold IR scores	Yes	47.7	50.9
Banerjee (2019)	BERT iterative re-ranking	Yes	41.3	42.3
IR with re-ranking				
Chia et al. (2019)	Iterative BM25	**No**	45.8	49.7
One-step IR				
BM25	BM25 Relevance Score	**No**	43.0	46.1
TF-IDF	TF-IDF Relevance Score	**No**	39.4	42.8
Feature-based				
D'Souza et al.(2019)	Feature-rich SVM ranking + Rules	Yes	39.4	44.4
D'Souza et al. (2019)	Feature-rich SVM ranking	Yes	34.1	37.1
Unification-based Reconstruction				
RS + US (Best)	Joint Relevance and Unification Score	**No**	**50.8**	**54.5**
US (Best)	Unification Score	**No**	22.9	21.9

Table 1: Results on test and dev set and comparison with state-of-the-art approaches. The column **trained** indicates whether the model requires an explicit training session on the explanation reconstruction task.

q and c_j the system has to retrieve the scientific facts describing how friction occurs and produces heat across objects. The corpus classifies these facts (f_3, f_4) as *central*. *Grounding* explanations like *"stick is a kind of object"* (f_1) link question and answer to the central explanations. *Lexical glues* such as *"to rub; to rub together means to mover against"* (f_2) are used to fill lexical gaps between sentences. Additionally, the corpus divides the facts belonging to F_{kb} into three inference categories: *retrieval type*, *inference supporting type*, and *complex inference type*. Taxonomic knowledge and properties such as *"stick is a kind of object"* (f_1) and *"friction is a kind of force"* (f_5) are classified as *retrieval type*. Facts describing actions, affordances, and requirements such as *"friction occurs when two object's surfaces move against each other"* (f_3) are grouped under the *inference supporting types*. Knowledge about causality, description of processes and if-then conditions such as *"friction causes the temperature of an object to increase"* (f_4) is classified as *complex inference*.

We implement Relevance and Unification Score adopting TF-IDF/BM25 vectors and cosine similarity function (i.e. $1 - cos(\vec{x}, \vec{y})$). Specifically, The RS model uses TF-IDF/BM25 to compute the relevance function for each fact in F_{kb} (i.e. $rs(h_j, f_i)$ function in Equation 1) while the US model adopts TF-IDF/BM25 to assign similarity scores to the hypotheses in E_{kb} (i.e. $sim(h_j, h_z)$ function in Equation 2). For reproducibility, the code is available at the following url: `https://github.com/ai-systems/unification_reconstruction_explanations`. Additional details can be found in the supplementary material.

4.1 Explanation Reconstruction

In line with the shared task (Jansen and Ustalov, 2019), the performances of the models are evaluated via Mean Average Precision (MAP) of the explanation ranking produced for a given question q_j and its correct answer a_j.

Table 1 illustrates the score achieved by our best implementation compared to state-of-the-art approaches in the literature. Previous approaches are grouped into four categories: *Transformers*, *Information Retrieval with re-ranking*, *One-step Information Retrieval*, and *Feature-based models*.

Transformers. This class of approaches employs the gold explanations in the corpus to train a BERT language model (Devlin et al., 2019). The best-performing system (Das et al., 2019) adopts a multi-step retrieval strategy. In the first step, it returns the top K sentences ranked by a TF-IDF model. In the second step, BERT is used to re-rank the paths composed of all the facts that are within 1-hop from the first retrieved set. Similarly, other approaches adopt BERT to re-rank each fact

Model	MAP			
	All	Central	Grounding	Lexical Glue
RS TF-IDF	42.8	43.4	25.4	8.2
RS BM25	46.1	46.6	23.3	10.7
US TF-IDF	21.6	16.9	22.0	13.4
US BM25	21.9	18.1	16.7	15.0
RS TF-IDF + US TF-IDF	48.5	46.4	**32.7**	11.7
RS TF-IDF + US BM25	50.7	48.6	30.42	13.4
RS BM25 + US TF-IDF	51.9	48.2	31.7	14.8
RS BM25 + US BM25	**54.5**	**51.7**	27.3	**16.7**

(a) Explanatory roles.

Model	MAP		
	1+ Overlaps	1 Overlap	0 Overlaps
RS TF-IDF	57.2	33.6	7.1
RS BM25	62.2	37.1	7.1
US TF-IDF	17.37	18.0	12.5
US BM25	18.1	18.1	**13.1**
RS TF-IDF + US TF-IDF	60.2	38.4	9.0
RS TF-IDF + US BM25	62.5	39.5	9.6
RS BM25 + US TF-IDF	61.3	40.6	11.0
RS BM25 + US BM25	**64.8**	**41.9**	11.2

(b) Lexical overlaps with the hypothesis.

Model	MAP		
	Retrieval	Inference-supporting	Complex Inference
RS TF-IDF	33.5	34.7	21.8
RS BM25	36.0	36.1	24.8
US TF-IDF	17.6	12.8	19.5
US BM25	16.8	13.2	20.9
RS TF-IDF + US TF-IDF	38.3	33.2	30.2
RS TF-IDF + US BM25	40.0	35.6	33.3
RS BM25 + US TF-IDF	40.5	33.6	33.4
RS BM25 + US BM25	**40.6**	**38.3**	**36.8**

(c) Inference types.

Table 2: Detailed analysis of the performance (dev-set) by breaking down the gold explanatory facts according to their explanatory role (2.a), number of lexical overlaps with the question (2.b) and inference type (2.c).

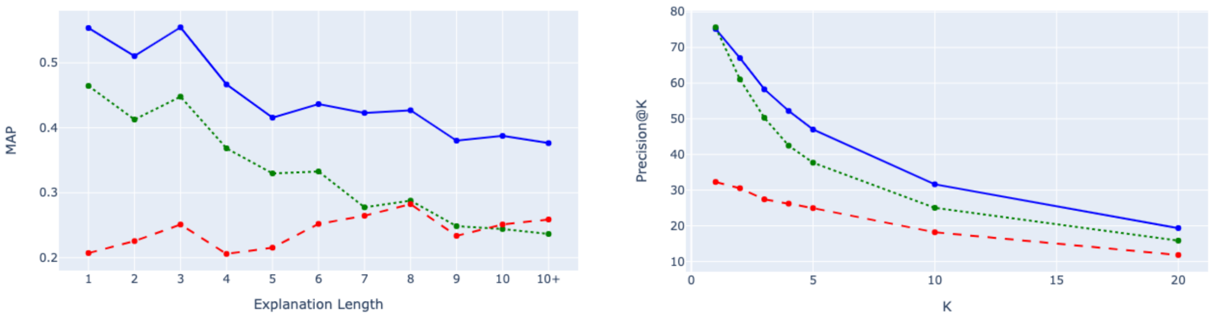
individually (Banerjee, 2019; Chia et al., 2019).

Although the best model achieves state-of-the-art results in explanation reconstruction, these approaches are computationally expensive, being limited by the application of a pre-filtering step to contain the space of candidate facts. Consequently, these systems do not scale with the size of the corpus. We estimated that the best performing model (Das et al., 2019) takes $\approx$ 10 hours to run on the whole test set (1240 questions) using 1 Tesla 16GB V100 GPU.

Comparatively, our model constructs explanations for all the questions in the test set in $\approx$ 30 seconds, without requiring the use of GPUs ($<$ 1 second per question). This feature makes the Unification-based Reconstruction suitable for large corpora and downstream question answering models (as shown in Section 4.4). Moreover, our approach does not require any explicit training session on the explanation regeneration task, with significantly reduced number of parameters to tune. Along with scalability, the proposed approach achieves nearly state-of-the-art results (50.8/54.5 MAP). Although we observe lower performance when compared to the best-performing approach (-5.5/-4.0 MAP), the joint RS + US model outperforms two BERT-based models (Chia et al., 2019; Banerjee, 2019) on both test and dev set by 3.1/3.6 and 9.5/12.2 MAP respectively.

Information Retrieval with re-ranking. Chia et al. (2019) describe a multi-step, iterative re-ranking model based on BM25. The first step consists in retrieving the explanation sentence that is most similar to the question adopting BM25 vectors. During the second step, the BM25 vector of the question is updated by aggregating it with the retrieved explanation sentence vector through a `max` operation. The first and second steps are repeated for K times. Although this approach uses scalable IR techniques, it relies on a multi-step retrieval strategy. Besides, the RS + US model outperforms this approach on both test and dev set by 5.0/4.8 MAP respectively.

One-step Information Retrieval. We compare the RS + US model with two IR baselines. The baselines adopt TF-IDF and BM25 to compute the Relevance Score only – i.e. the $us(q, c_j, f_i)$ term in Equation 1 is set to 0 for each fact $f_i \in F_{kb}$. In line with previous IR literature (Robertson et al., 2009), BM25 leads to better performance than TF-IDF. While these approaches share similar characteristics, the combined RS + US model outperforms both RS BM25 and RS TF-IDF on test and dev-set by 7.8/8.4 and 11.4/11.7 MAP. Moreover, the joint RS + US model improves the performance of the US model alone by 27.9/32.6 MAP. These results outline the complementary aspects of Relevance and Unification Score. We provide a detailed anal-



(a) MAP vs Explanation length.

(b) Precision@K.

Figure 2: Impact of the Unification Score on semantic drift (3.a) and precision (3.b). RS + US (Blue Straight), RS (Green Dotted), US (Red Dashed).

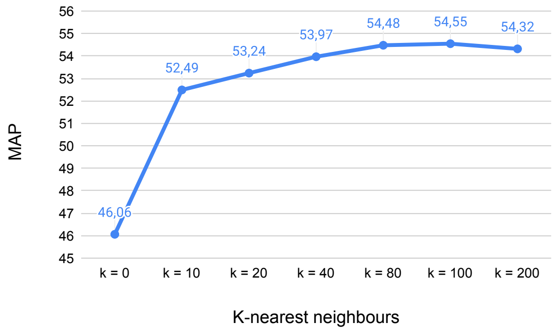


Figure 3: Impact of the k-NN clustering on the final MAP score. The value k represents the number of similar hypotheses considered for the Unification Score.

ysis by performing an ablation study on the dev-set (Section 4.2).

Feature-based models. D'Souza et al. (2019) propose an approach based on a learning-to-rank paradigm. The model extracts a set of features based on overlaps and coherence metrics between questions and explanation sentences. These features are then given in input to a SVM ranker module. While this approach scales to the whole corpus without requiring any pre-filtering step, it is significantly outperformed by the RS + US model on both test and dev set by 16.7/17.4 MAP respectively.

4.2 Explanation Analysis

We present an ablation study with the aim of understanding the contribution of each sub-component to the general performance of the joint RS + US model (see Table 1). To this end, a detailed evaluation on the development set of the Worldtree corpus is carried out, analysing the performance in reconstructing explanations of different types and complexity. We compare the joint model (RS + US) with each individual sub-component (RS and US alone). In addition, a set of qualitative examples are analysed to provide additional insights on the complementary aspects captured by Relevance and Unification Score.

Explanatory categories. Given a question q_j and its correct answer a_j, we classify a fact f_i belonging to the gold explanation E_j according to its explanatory role (*central, grounding, lexical glue*) and inference type (*retrieval, inference-supporting and complex inference*). In addition, three new categories are derived from the number of overlaps between f_i and the concatenation of q_j with a_j (h_j) computed by considering nouns, verbs, adjectives and adverbs (1+ overlaps, 1 overlap, 0 overlaps).

Table 2 reports the MAP score for each of the described categories. Overall, the best results are obtained by the BM25 implementation of the joint model (RS BM25 + US BM25) with a MAP score of 54.5. Specifically, RS BM25 + US BM25 achieves a significant improvement over both RS BM25 (+8.5 MAP) and US BM25 (+32.6 MAP) baselines. Regarding the explanatory roles (Table 2a), the joint TF-IDF implementation shows the best performance in the reconstruction of *grounding* explanations (32.7 MAP). On the other hand, a significant improvement over the RS baseline is obtained by RS BM25 + US BM25 on both *lexical* glues and *central* explanation sentences (+6.0 and +5.6 MAP over RS BM25).

Regarding the lexical overlaps categories (Table 2b), we observe a steady improvement for all the combined RS + US models over the respective RS baselines. Notably, the US models achieve the best performance on the 0 overlaps category, which includes the most challenging facts for the RS models. The improved ability to rank abstract

Question	Answer	Explanation Fact	Most Similar Hypotheses in E_{kb}	RS	RS + US
If you bounce a rubber ball on the floor it goes up and then comes down. What **causes** the ball to come down?	**gravity**	**gravity**; gravitational force **causes** objects that have mass; substances to be pulled down; to fall on a planet	(1) A ball is tossed up in the air and it comes back down. The ball comes back down because of - gravity (2) A student drops a ball. Which force causes the ball to fall to the ground? - gravity	#36	#2 (↑34)
Which **animals** would most likely be helped by flood in a coastal area?	alligators	as water increases in an environment, the population of aquatic **animals** will increase	(1) Where would animals and plants be most affected by a flood? - low areas (2) Which change would most likely increase the number of salamanders? - flood	#198	#57 (↑141)
What is an example of a force producing heat?	two sticks getting warm when rubbed together	friction causes the temperature of an object to increase	(1) Rubbing sandpaper on a piece of wood produces what two types of energy? - sound and heat (2) Which force produces energy as heat? - friction	#1472	#21 (↑1451)

Table 3: Impact of the Unification Score on the ranking of scientific facts with increasing complexity.

explanatory facts contributes to better performance for the joint models (RS + US) in the reconstruction of explanations that share few terms with question and answer (*1 Overlap* and *0 Overlaps* categories). This characteristic leads to an improvement of 4.8 and 4.1 MAP for the RS BM25 + US BM25 model over the RS BM25 baseline.

Similar results are achieved on the inference types categories (Table 2c). Crucially, the largest improvement is observed for *complex inference* sentences where RS BM25 + US BM25 outperforms RS BM25 by 12.0 MAP, confirming the decisive contribution of the Unification Score to the ranking of complex scientific facts.

Semantic drift. Science questions in the Worldtree corpus require an average of six facts in their explanations (Jansen et al., 2016). Long explanations typically include sentences that share few terms with question and answer, increasing the probability of semantic drift. Therefore, to test the impact of the Unification Score on the robustness of the model, we measure the performance in the reconstruction of many-hops explanations.

Figure 2a shows the change in MAP score for the RS + US, RS and US models (BM25) with increasing explanation length. The fast drop in performance for the Relevance Score reflects the complexity of the task. This drop occurs because the RS model is not able to rank abstract explanatory facts. Conversely, the US model exhibits increasing performance, with a trend that is inverse. Short explanations, indeed, tend to include question-specific facts with low explanatory power. On the other hand, the longer the explanation, the higher the number of core scientific facts. Therefore, the decrease in MAP observed for the RS model is compensated by the Unification Score, since core scientific facts tend to form unification patterns across similar questions. This results demonstrate that the Unification Score has a crucial role in alleviating the semantic drift for the joint model (RS + US), resulting in a larger improvement on many-hops explanations (6+ facts).

Similarly, Figure 2b illustrates the Precision@K. As shown in the graph, the drop in precision for the US model exhibits the slowest degradation. Similarly to what observed for many-hops explanations, the US score contributes to the robustness of the RS + US model, making it able to reconstruct more precise explanations. As discussed in section 4.4, this feature has a positive impact on question answering.

k-NN clustering. We investigate the impact of the k-NN clustering on the explanation reconstruction task. Figure 3 shows the MAP score obtained by the joint RS + US model (BM25) with different numbers k of nearest hypotheses considered for the Unification Score. The graph highlights the improvement in MAP achieved with increasing values of k. Specifically, we observe that the best MAP is obtained with $k = 100$. These results confirm that the explanatory power can be effectively estimated using clusters of similar hypotheses, and that the unification-based mechanism has a crucial role in improving the performance of the relevance model.

4.3 Qualitative analysis.

To provide additional insights on the complementary aspects of Unification and Relevance Score, we present a set of qualitative examples from the dev-set. Table 3 illustrates the ranking assigned by RS and RS + US models to scientific sentences of increasing complexity. The words in **bold** indicate lexical overlaps between question, answer and explanation sentence. In the first example, the sentence *"gravity; gravitational force causes objects* *that have mass; substances to be pulled down; to fall on a planet"* shares key terms with question and candidate answer and is therefore relatively easy to rank for the RS model (#36). Nevertheless, the RS + US model is able to improve the ranking by 34 positions (#2), as the gravitational law represents a scientific pattern with high explanatory unification, frequently reused across similar questions. The impact of the Unification Score is more evident when considering abstract explanatory facts. Coming back to our original example (i.e. *"What is an example of a force producing heat?"*), the fact *"friction causes the temperature of an object to increase"* has no significant overlaps with question and answer. Thus, the RS model ranks the gold explanation sentence in a low position (#1472). However, the Unification Score (US) is able to capture the explanatory power of the fact from similar hypotheses in E_{kb}, pushing the RS + US ranking up to position #21 (+1451).

4.4 Question Answering

To understand whether the constructed explanations can support question answering, we compare the performance of BERT for multiple-choice QA (Devlin et al., 2019) without explanations with the performance of BERT provided with the top K explanation sentences retrieved by RS and RS + US models (BM25). BERT without explanations operates on question and candidate answer only. On the other hand, BERT with explanation receives the following input: the question (q), a candidate answer (c_i) and the explanation for c_i (E_i). In this setting, the model is fine-tuned for binary classification ($bert_b$) to predict a set of probability scores $P = \{p_1,\ p_2,\ ...,\ p_n\}$ for each candidate answer in $C = \{c1,\ c_2,\ ...,\ c_n\}$:

$$bert_b(\texttt{[CLS]} \mid\mid q \mid\mid c_i \mid\mid \texttt{[SEP]} \mid\mid E_i) = p_i \quad (4)$$

The binary classifier operates on the final hidden state corresponding to the `[CLS]` token. To answer the question q, the model selects the candidate answer c_a such that $a = \text{argmax}_i\, p_i$.

Table 4 reports the accuracy with and without explanations on the Worldtree *test-set* for *easy* and *challenge* questions (Clark et al., 2018). Notably, a significant improvement in accuracy can be observed when BERT is provided with explanations retrieved by the reconstruction modules (+9.84% accuracy with RS BM25 + US BM25 model). The improvement is consistent on the *easy* split (+6.92%) and particularly significant for *challenge* questions (+15.69%). Overall, we observe a correlation between more precise explanations and accuracy in answer prediction, with BERT + RS being outperformed by BERT + RS + US for each value of K. The decrease in accuracy occurring with increasing values of K is coherent with the drop in precision for the models observed in Figure 2b. Moreover, steadier results adopting the RS + US model suggest a positive contribution from abstract explanatory facts. Additional investigation of this aspect will be a focus for future work.

Model	**Accuracy**		
	Easy	Challenge	Overall
BERT (no explanation)	48.54	26.28	41.78
BERT + RS (K = 3)	53.20	40.97	49.39
BERT + RS (K = 5)	54.36	38.14	49.31
BERT + RS (K = 10)	32.71	29.63	31.75
BERT + RS + US (K = 3)	**55.46**	**41.97**	**51.62**
BERT + RS + US (K = 5)	54.48	39.43	50.12
BERT + RS + US (K = 10)	48.66	37.37	45.14

Table 4: Performance of BERT on question answering (test-set) with and without the explanation reconstruction models.

5 Conclusion

This paper proposed a novel framework for multi-hop explanation reconstruction based on *explanatory unification*. An extensive evaluation on the Worldtree corpus led to the following conclusions: (1) The approach is competitive with state-of-the-art Transformers, yet being significantly faster and inherently scalable; (2) The unification-based mechanism supports the construction of complex and many hops explanations; (3) The constructed explanations improves the accuracy of BERT for question answering by up to 10% overall. As a future work, we plan to extend the framework adopting neural embeddings for sentence representation.

Acknowledgements

The authors would like to thank the anonymous reviewers for the constructive feedback. A special thanks to Deborah Ferreira for the helpful discussions, and to the members of the AI Systems lab from the University of Manchester. Additionally, we would like to thank the Computational Shared Facility of the University of Manchester for providing the infrastructure to run our experiments.

References

Pratyay Banerjee. 2019. Asu at textgraphs 2019 shared task: Explanation regeneration using language models and iterative re-ranking. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 78–84.

Or Biran and Courtenay Cotton. 2017. Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI)*, volume 8.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pages 9539–9549.

Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2019. Red dragon ai at textgraphs 2019 shared task: Language model assisted explanation generation. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 85–89.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Rajarshi Das, Ameya Godbole, Manzil Zaheer, Shehzaad Dhuliawala, and Andrew McCallum. 2019. Chains-of-reasoning at textgraphs 2019 shared task: Reasoning over chains of facts for explainable multi-hop inference. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 101–117.

Ramon Lopez De Mantaras, David McSherry, Derek Bridge, David Leake, Barry Smyth, Susan Craw, Boi Faltings, Mary Lou Maher, MICHAEL T COX, Kenneth Forbus, et al. 2005. Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review*, 20(3):215–240.

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

Jennifer D'Souza, Isaiah Onando Mulang, and Sören Auer. 2019. Team svmrank: Leveraging feature-rich support vector machines for ranking explanations to elementary science questions. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 90–100.

Deborah Ferreira and André Freitas. 2020a. Natural language premise selection: Finding supporting statements for mathematical text. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2175–2182.

Deborah Ferreira and André Freitas. 2020b. Premise selection in natural language mathematical texts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7365–7374.

Daniel Fried, Peter Jansen, Gustave Hahn-Powell, Mihai Surdeanu, and Peter Clark. 2015. Higher-order lexical semantic models for non-factoid answer reranking. *Transactions of the Association for Computational Linguistics*, 3:197–210.

Michael Friedman. 1974. Explanation and scientific understanding. *The Journal of Philosophy*, 71(1):5–19.

Carl G Hempel. 1965. Aspects of scientific explanation.

Peter Jansen, Niranjan Balasubramanian, Mihai Surdeanu, and Peter Clark. 2016. What's in an explanation? characterizing knowledge and inference requirements for elementary science exams. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2956–2965.

Peter Jansen, Rebecca Sharp, Mihai Surdeanu, and Peter Clark. 2017. Framing qa as building and ranking intersentence answer justifications. *Computational Linguistics*, 43(2):407–449.

Peter Jansen and Dmitry Ustalov. 2019. Textgraphs 2019 shared task on multi-hop inference for explanation regeneration. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 63–77.

Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison. 2018. Worldtree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Daniel Khashabi, Erfan Sadeqi Azer, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2019. On the capabilities and limitations of reasoning for natural language understanding. *arXiv preprint arXiv:1901.02522*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2018. Question answering as global reasoning over semantic abstractions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2019. Qasc: A dataset for question answering via sentence composition. *arXiv preprint arXiv:1910.11473*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 311–316.

Philip Kitcher. 1981. Explanatory unification. *Philosophy of science*, 48(4):507–531.

Philip Kitcher. 1989. Explanatory unification and the causal structure of the world.

Janet Kolodner. 2014. *Case-based reasoning*. Morgan Kaufmann.

Carmen Lacave and Francisco J Diez. 2004. A review of explanation methods for heuristic expert systems. *The Knowledge Engineering Review*, 19(2):133–146.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Tom M Mitchell, Richard M Keller, and Smadar T Kedar-Cabelli. 1986. Explanation-based generalization: A unifying view. *Machine learning*, 1(1):47–80.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Wesley C Salmon. 1984. *Scientific explanation and the causal structure of the world*. Princeton University Press.

Frode Sørmo, Jörg Cassens, and Agnar Aamodt. 2005. Explanation in case-based reasoning–perspectives and goals. *Artificial Intelligence Review*, 24(2):109–143.

Paul Thagard. 1992. Analogy, explanation, and education. *Journal of research in science teaching*, 29(6):537–544.

Paul Thagard and Abninder Litt. 2008. Models of scientific explanation. *The Cambridge handbook of computational psychology*, pages 549–564.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2020. A survey on explainability in machine reading comprehension. *arXiv preprint arXiv:2010.00389*.

Mokanarangan Thayaparan, Marco Valentino, Viktor Schlegel, and André Freitas. 2019. Identifying supporting facts for multi-hop question answering with document graph networks. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 42–51.

Michael R Wick and William B Thompson. 1992. Reconstructive expert system explanation. *Artificial Intelligence*, 54(1-2):33–70.

James Woodward. 2005. *Making things happen: A theory of causal explanation*. Oxford university press.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2019. Quick and (not so) dirty: Unsupervised selection of justification sentences for multi-hop question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2578–2589.

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

A Supplementary Material

A.1 Hyperparameters tuning

The hyperparameters of the model have been tuned manually. The criteria for the optimisation is the maximisation of the MAP score on the dev-set. Here, we report the values adopted for the experiments described in the paper.

The Unification-based Reconstruction adopts two hyperparameters. Specifically, λ_1 is the weight assigned to the relevance score in equation 1, while k is the number of similar hypotheses to consider for the calculation of the unification score (equation 2). The values adopted for these parameters are as follows:

1. $\lambda_1 = 0.83$ ($1 - \lambda_1 = 0.17$)
2. $k = 100$

A.2 BERT model

For question answering we adopt a BERT$_{BASE}$ model. The model is implemented using PyTorch (`https://pytorch.org/`) and fine-tuned using 4 Tesla 16GB V100 GPUs for 10 epochs in total with batch size 32 and seed 42. The hyperparameters adopted for BERT are as follows:

- gradient_accumulation_steps = 1
- learning_rate = 5e-5
- weight_decay = 0.0
- adam_epsilon = 1e-8
- warmup_steps = 0
- max_grad_norm = 1.0

A.3 Data and code

The experiments are carried out on the TextGraphs 2019 version (`https://github.com/umanlp/tg2019task`) of the Worldtree corpus. The full dataset can be downloaded at the following URL: `http://cognitiveai.org/dist/worldtree_corpus_textgraphs2019sharedtask_withgraphvis.zip`.

The code to reproduce the experiments described in the paper is available at the following URL: `https://github.com/ai-systems/unification_reconstruction_explanations`

Dictionary-based Debiasing of Pre-trained Word Embeddings

Masahiro Kaneko
Tokyo Metropolitan University
kaneko-masahiro@ed.tmu.ac.jp

Danushka Bollegala[*]
University of Liverpool, Amazon
danushka@liverpool.ac.uk

Abstract

Word embeddings trained on large corpora have shown to encode high levels of unfair discriminatory gender, racial, religious and ethnic biases. In contrast, human-written dictionaries describe the meanings of words in a concise, objective and an unbiased manner. We propose a method for debiasing pre-trained word embeddings using dictionaries, without requiring access to the original training resources or any knowledge regarding the word embedding algorithms used. Unlike prior work, our proposed method does not require the types of biases to be pre-defined in the form of word lists, and learns the constraints that must be satisfied by unbiased word embeddings automatically from dictionary definitions of the words. Specifically, we learn an encoder to generate a debiased version of an input word embedding such that it (a) retains the semantics of the pre-trained word embeddings, (b) agrees with the unbiased definition of the word according to the dictionary, and (c) remains orthogonal to the vector space spanned by any biased basis vectors in the pre-trained word embedding space. Experimental results on standard benchmark datasets show that the proposed method can accurately remove unfair biases encoded in pre-trained word embeddings, while preserving useful semantics.

1 Introduction

Despite pre-trained word embeddings are useful due to their low dimensionality, memory and compute efficiency, they have shown to encode not only the semantics of words but also unfair discriminatory biases such as gender, racial or religious biases (Bolukbasi et al., 2016; Zhao et al., 2018a; Rudinger et al., 2018; Zhao et al., 2018b; Elazar and Goldberg, 2018; Kaneko and Bollegala, 2019). On the other hand, human-written dictionaries act as an impartial, objective and unbiased source of word meaning. Although methods that learn word embeddings by purely using dictionaries have been proposed (Tissier et al., 2017), they have coverage and data sparseness related issues because precompiled dictionaries do not capture the meanings of neologisms or provide numerous contexts as in a corpus. Consequently, prior work has shown that word embeddings learnt from large text corpora to outperform those created from dictionaries in downstream NLP tasks (Alsuhaibani et al., 2019; Bollegala et al., 2016).

We must overcome several challenges when using dictionaries to debias pre-trained word embeddings. First, not all words in the embeddings will appear in the given dictionary. Dictionaries often have limited coverage and will not cover neologisms, orthographic variants of words etc. that are likely to appear in large corpora. A lexicalised debiasing method would generalise poorly to the words not in the dictionary. Second, it is not known apriori what biases are hidden inside a set of pre-trained word embedding vectors. Depending on the source of documents used for training the embeddings, different types of biases will be learnt and amplified by different word embedding learning algorithms to different degrees (Zhao et al., 2017).

Prior work on debiasing required that the biases to be pre-defined (Kaneko and Bollegala, 2019). For example, Hard-Debias (HD; Bolukbasi et al., 2016) and Gender Neutral Glove (GN-GloVe; Zhao et al., 2018b) require lists of *male* and *female* pronouns for defining the *gender* direction. However, gender bias is only one of the many biases that exist in pre-trained word embeddings. It is inconvenient to prepare lists of words covering all different types of biases we must remove from pre-trained word embeddings. Moreover, such pre-

---

[*] Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 212–223
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

compiled word lists are likely to be incomplete and inadequately cover some biases. Indeed, Gonen and Goldberg (2019) showed empirical evidence that such debiasing methods do not remove all discriminative biases from word embeddings. Unfair biases have adversely affected several NLP tasks such as machine translation (Vanmassenhove et al., 2018) and language generation (Sheng et al., 2019). Racial biases have also been shown to affect criminal prosecutions (Manzini et al., 2019) and career adverts (Lambrecht and Tucker, 2016). These findings show the difficulty of defining different biases using pre-compiled lists of words, which is a requirement in previously proposed debiasing methods for static word embeddings.

We propose a method that uses a dictionary as a source of bias-free definitions of words for debiasing pre-trained word embeddings[1]. Specifically, we learn an encoder that filters-out biases from the input embeddings. The debiased embeddings are required to simultaneously satisfy three criteria: (a) must preserve all non-discriminatory information in the pre-trained embeddings (*semantic preservation*), (b) must be similar to the dictionary definition of the words (*dictionary agreement*), and (c) must be orthogonal to the subspace spanned by the basis vectors in the pre-trained word embedding space that corresponds to discriminatory biases (*bias orthogonality*). We implement the semantic preservation and dictionary agreement using two decoders, whereas the bias orthogonality is enforced by a parameter-free projection. The debiasing encoder and the decoders are learnt end-to-end by a joint optimisation method. Our proposed method is agnostic to the details of the algorithms used to learn the input word embeddings. Moreover, unlike counterfactual data augmentation methods for debiasing (Zmigrod et al., 2019; Hall Maudslay et al., 2019), we do *not* require access to the original training resources used for learning the input word embeddings.

Our proposed method overcomes the above-described challenges as follows. First, instead of learning a lexicalised debiasing model, we operate on the word embedding space when learning the encoder. Therefore, we can use the words that are in the intersection of the vocabularies of the pre-trained word embeddings and the dictionary to learn the encoder, enabling us to generalise to the words not in the dictionary. Second, we do *not* require pre-compiled word lists specifying the biases. The dictionary acts as a clean, unbiased source of word meaning that can be considered as *positive* examples of debiased meanings. In contrast to the existing debiasing methods that require us to pre-define *what to remove*, the proposed method can be seen as using the dictionary as a guideline for *what to retain* during debiasing.

We evaluate the proposed method using four standard benchmark datasets for evaluating the biases in word embeddings: Word Embedding Association Test (WEAT; Caliskan et al., 2017), Word Association Test (WAT; Du et al., 2019), SemBias (Zhao et al., 2018b) and WinoBias (Zhao et al., 2018a). Our experimental results show that the proposed debiasing method accurately removes unfair biases from three widely used pre-trained embeddings: Word2Vec (Mikolov et al., 2013b), GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2017). Moreover, our evaluations on semantic similarity and word analogy benchmarks show that the proposed debiasing method preserves useful semantic information in word embeddings, while removing unfair biases.

2 Related Work

Dictionaries have been popularly used for learning word embeddings (Budanitsky and Hirst, 2006, 2001; Jiang and Conrath, 1997). Methods that use both dictionaries (or lexicons) and corpora to jointly learn word embeddings (Tissier et al., 2017; Alsuhaibani et al., 2019; Bollegala et al., 2016) or post-process (Glavaš and Vulić, 2018; Faruqui et al., 2015) have also been proposed. However, learning embeddings from dictionaries alone results in coverage and data sparseness issues (Bollegala et al., 2016) and does not guarantee bias-free embeddings (Lauscher and Glavas, 2019). To the best of our knowledge, we are the first to use dictionaries for debiasing pre-trained word embeddings.

Bolukbasi et al. (2016) proposed a post-processing approach that projects gender-neutral words into a subspace, which is orthogonal to the gender dimension defined by a list of gender-definitional words. They refer to words associated with gender (e.g., *she*, *actor*) as gender-definitional words, and the remainder gender-neutral. They proposed a *hard-debiasing* method where the gender direction is computed as the vector difference between the embeddings of the correspond-

---

[1] Code and debiased embeddings: `https://github.com/kanekomasahiro/dict-debias`

ing gender-definitional words, and a *soft-debiasing* method, which balances the objective of preserving the inner-products between the original word embeddings, while projecting the word embeddings into a subspace orthogonal to the gender definitional words. Both hard and soft debiasing methods ignore gender-definitional words during the subsequent debiasing process, and focus only on words that are *not* predicted as gender-definitional by the classifier. Therefore, if the classifier erroneously predicts a stereotypical word as a gender-definitional word, it would not get debiased.

Zhao et al. (2018b) modified the GloVe (Pennington et al., 2014) objective to learn gender-neutral word embeddings (GN-GloVe) from a given corpus. They maximise the squared ℓ_2 distance between gender-related sub-vectors, while simultaneously minimising the GloVe objective. Unlike, the above-mentioned methods, Kaneko and Bollegala (2019) proposed a post-processing method to preserve gender-related information with autoencoder (Kaneko and Bollegala, 2020), while removing discriminatory biases from stereotypical cases (GP-GloVe). However, all prior debiasing methods require us to pre-define the biases in the form of explicit word lists containing gender and stereotypical word associations. In contrast we use dictionaries as a source of bias-free semantic definitions of words and do not require pre-defining the biases to be removed. Although we focus on static word embeddings in this paper, unfair biases have been found in contextualised word embeddings as well (Zhao et al., 2019; Vig, 2019; Bordia and Bowman, 2019; May et al., 2019).

Adversarial learning methods (Xie et al., 2017; Elazar and Goldberg, 2018; Li et al., 2018) for debiasing first encode the inputs and then two classifiers are jointly trained – one predicting the target task (for which we must ensure high prediction accuracy) and the other protected attributes (that must not be easily predictable). However, Elazar and Goldberg (2018) showed that although it is possible to obtain chance-level development-set accuracy for the protected attributes during training, a post-hoc classifier trained on the encoded inputs can still manage to reach substantially high accuracies for the protected attributes. They conclude that adversarial learning alone does not guarantee invariant representations for the protected attributes. Ravfogel et al. (2020) found that iteratively projecting word embeddings to the null space of the gender direction to further improve the debiasing performance.

To evaluate biases, Caliskan et al. (2017) proposed the Word Embedding Association Test (WEAT) inspired by the Implicit Association Test (IAT; Greenwald et al., 1998). Ethayarajh et al. (2019) showed that WEAT to be systematically overestimating biases and proposed a correction. The ability to correctly answer gender-related word analogies (Zhao et al., 2018b) and resolve gender-related coreferences (Zhao et al., 2018a; Rudinger et al., 2018) have been used as extrinsic tasks for evaluating the bias in word embeddings. We describe these evaluation benchmarks later in § 4.3.

3 Dictionary-based Debiasing

Let us denote the n-dimensional pre-trained word embedding of a word w by $\boldsymbol{w} \in \mathbb{R}^n$ trained on some resource $\mathcal{C}$ such as a text corpus. Moreover, let us assume that we are given a dictionary $\mathcal{D}$ containing the definition, $s(w)$ of w. If the pre-trained embeddings distinguish among the different senses of w, then we can use the gloss for the corresponding sense of w in the dictionary as $s(w)$. However, the majority of word embedding learning methods do not produce sense-specific word embeddings. In this case, we can either use all glosses for w in $\mathcal{D}$ by concatenating or select the gloss for the dominant (most frequent) sense of w[2]. Without any loss of generality, in the remainder of this paper, we will use $s(w)$ to collectively denote a gloss selected by any one of the above-mentioned criteria with or without considering the word senses (in § 5.3, we evaluate the effect of using all vs. dominant gloss).

Next, we define the objective functions optimised by the proposed method for the purpose of learning unbiased word embeddings. Given, $\boldsymbol{w}$, we model the debiasing process as the task of learning an encoder, $E(\boldsymbol{w}; \boldsymbol{\theta}_e)$ that returns an $m (\leq n)$-dimensional debiased version of $\boldsymbol{w}$. In the case where we would like to preserve the dimensionality of the input embeddings, we can set $m = n$, or $m < n$ to further compress the debiased embeddings.

Because the pre-trained embeddings encode rich semantic information from a large text corpora, often far exceeding the meanings covered in the

---

[2] Prior work on debiasing static word embeddings do not use contextual information that is required for determining word senses. Therefore, for comparability reasons we do neither.

dictionary, we must preserve this semantic information as much as possible during the debiasing process. We refer to this constraint as *semantic preservation*. Semantic preservation is likely to lead to good performance in downstream NLP applications that use pre-trained word embeddings. For this purpose, we decode the encoded version of $\boldsymbol{w}$ using a decoder, D_c, parametrised by $\boldsymbol{\theta}_c$ and define J_c to be the reconstruction loss given by (1).

$$J_c(w) = ||\boldsymbol{w} - D_c(E(\boldsymbol{w}; \boldsymbol{\theta}_e); \boldsymbol{\theta}_c)||_2^2 \qquad (1)$$

Following our assumption that the dictionary definition, $s(w)$, of w is a concise and unbiased description of the meaning of w, we would like to ensure that the encoded version of $\boldsymbol{w}$ is similar to $s(w)$. We refer to this constraint as *dictionary agreement*. To formalise dictionary agreement empirically, we first represent $s(w)$ by a sentence embedding vector $\boldsymbol{s}(w) \in \mathbb{R}^n$. Different sentence embedding methods can be used for this purpose such as convolutional neural networks (Kim, 2014), recurrent neural networks (Peters et al., 2018) or transformers (Devlin et al., 2019). For the simplicity, we use the smoothed inverse frequency (SIF; Arora et al., 2017) for creating $\boldsymbol{s}(w)$ in this paper. SIF computes the embedding of a sentence as the weighted average of the pre-trained word embeddings of the words in the sentence, where the weights are computed as the inverse unigram probability. Next, the first principal component vector of the sentence embeddings are removed. The dimensionality of the sentence embeddings created using SIF is equal to that of the pre-trained word embeddings used. Therefore, in our case we have both $\boldsymbol{w}, \boldsymbol{s}(w) \in \mathbb{R}^n$.

We decode the debiased embedding $E(\boldsymbol{w}; \boldsymbol{\theta}_e)$ of w using a decoder D_d, parametrised by $\boldsymbol{\theta}_d$ and compute the squared ℓ_2 distance between it and $\boldsymbol{s}(w)$ to define an objective J_d given by (2).

$$J_d(w) = ||\boldsymbol{s}(w) - D_d(E(\boldsymbol{w}; \boldsymbol{\theta}_e); \boldsymbol{\theta}_d)||_2^2 \qquad (2)$$

Recalling that our goal is to remove unfair biases from pre-trained word embeddings and we assume dictionary definitions to be free of such biases, we define an objective function that explicitly models this requirement. We refer to this requirement as the *bias orthogonality* of the debiased embeddings. For this purpose, we first project the pre-trained word embedding $\boldsymbol{w}$ of a word w into a subspace that is orthogonal to the dictionary definition vector $\boldsymbol{s}(w)$. Let us denote this projection by $\phi(\boldsymbol{w}, \boldsymbol{s}(w)) \in \mathbb{R}^n$. We require that the debiased word embedding, $E(\boldsymbol{w}; \boldsymbol{\theta}_e)$, must be orthogonal to $\phi(\boldsymbol{w}, \boldsymbol{s}(w))$, and formalise this as the minimisation of the squared inner-product given in (3).

$$J_a(w) = \left(E(\phi(\boldsymbol{w}, \boldsymbol{s}(w)); \boldsymbol{\theta}_e)^\top E(\boldsymbol{w}; \boldsymbol{\theta}_e)\right)^2 \quad (3)$$

Note that because $\phi(\boldsymbol{w}, \boldsymbol{s}(w))$ lives in the space spanned by the original (prior to encoding) vector space, we must first encode it using E before considering the orthogonality requirement.

To derive $\phi(\boldsymbol{w}, \boldsymbol{s}(w))$, let us assume the n-dimensional basis vectors in the R^n vector space spanned by the pre-trained word embeddings to be $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n$. Moreover, without loss of generality, let the subspace spanned by the subset of the first $k(< n)$ basis vectors $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k$ to be $\mathcal{B} \subseteq \mathbb{R}^n$. The projection $\boldsymbol{v}_{\mathcal{B}}$ of a vector $\boldsymbol{v} \in \mathbb{R}^n$ onto $\mathcal{B}$ can be expressed using the basis vectors as in (4).

$$\boldsymbol{v}_{\mathcal{B}} = \sum_{j=1}^{k} (\boldsymbol{v}^\top \boldsymbol{b}_j)\boldsymbol{b}_j \qquad (4)$$

To show that $\boldsymbol{v} - \boldsymbol{v}_{\mathcal{B}}$ is orthogonal to $\boldsymbol{v}_{\mathcal{B}}$ for any $\boldsymbol{v} \in \mathcal{B}$, let us express $\boldsymbol{v} - \boldsymbol{v}_{\mathcal{B}}$ using the basis vectors as given in (5).

$$\begin{aligned}\boldsymbol{v} - \boldsymbol{v}_{\mathcal{B}} &= \sum_{i=1}^{n} (\boldsymbol{v}^\top \boldsymbol{b}_i)\boldsymbol{b}_i - \sum_{j=1}^{k} (\boldsymbol{v}^\top \boldsymbol{b}_j)\boldsymbol{b}_j \\ &= \sum_{i=k+1}^{n} (\boldsymbol{v}^\top \boldsymbol{b}_i)\boldsymbol{b}_i \qquad (5)\end{aligned}$$

We see that there are no basis vectors in common between the summations in (4) and (5). Therefore, ${\boldsymbol{v}_{\mathcal{B}}}^\top(\boldsymbol{v} - \boldsymbol{v}_{\mathcal{B}}) = 0$ for $\forall \boldsymbol{v} \in \mathcal{B}$.

Considering that $\boldsymbol{s}(w)$ defines a direction that does not contain any unfair biases, we can compute the vector rejection of $\boldsymbol{w}$ on $\boldsymbol{s}(w)$ following this result. Specifically, we subtract the projection of $\boldsymbol{w}$ along the unit vector defining the direction of $\boldsymbol{s}(w)$ to compute ϕ as in (6).

$$\phi(\boldsymbol{w}, \boldsymbol{s}(w)) = \boldsymbol{w} - \boldsymbol{w}^\top \boldsymbol{s}(w) \frac{\boldsymbol{s}(w)}{||\boldsymbol{s}(w)||} \qquad (6)$$

We consider the linearly-weighted sum of the above-defined three objective functions as the total objective function as given in (7).

$$J(w) = \alpha J_c(w) + \beta J_d(w) + \gamma J_a(w) \qquad (7)$$

Here, $\alpha, \beta, \gamma \geq 0$ are scalar coefficients satisfying $\alpha + \beta + \gamma = 1$. Later, in § 4 we experimentally determine the values of α, β and γ using a development dataset.

4 Experiments

4.1 Word Embeddings

In our experiments, we use the following publicly available pre-trained word embeddings: **Word2Vec**[3] (300-dimensional embeddings for ca. 3M words learned from Google News corpus (Mikolov et al., 2013a)), **GloVe**[4] (300-dimensional embeddings for ca. 2.1M words learned from the Common Crawl (Pennington et al., 2014)), and **fastText**[5] (300-dimensional embeddings for ca. 1M words learned from Wikipedia 2017, UMBC webbase corpus and statmt.org news (Bojanowski et al., 2017)).

As the dictionary definitions, we used the glosses in the WordNet (Fellbaum, 1998), which has been popularly used to learn word embeddings in prior work (Tissier et al., 2017; Bosc and Vincent, 2018; Washio et al., 2019). However, we note that our proposed method does not depend on any WordNet-specific features, thus in principle can be applied to any dictionary containing definition sentences. Words that do not appear in the vocabulary of the pre-trained embeddings are ignored when computing $\boldsymbol{s}(w)$ for the headwords w in the dictionary. Therefore, if all the words in a dictionary definition are ignored, then the we remove the corresponding headwords from training. Consequently, we are left with 54,528, 64,779 and 58,015 words respectively for **Word2Vec**, **GloVe** and **fastText** embeddings in the training dataset. We randomly sampled 1,000 words from this dataset and held-out as a development set for the purpose of tuning various hyperparameters in the proposed method.

E, D_c and D_d are implemented as single-layer feed forward neural networks with a hyperbolic tangent activation at the outputs. It is known that pre-training is effective when using autoencoders E and D_c for debiasing (Kaneko and Bollegala, 2019). Therefore, we randomly select 5000 words from each pre-trained word embedding set and pre-train the autoencoders on those words with a mini-batch of size 512. In pre-training, the model with the lowest loss according to (1) in the development set for pre-traininng is selected.

4.2 Hyperparameters

During optimisation, we used dropout (Srivastava et al., 2014) with probability 0.05 to $\boldsymbol{w}$ and $E(w)$. We used Adam (Kingma and Ba, 2015) with initial learning rate set to 0.0002 as the optimiser to find the parameters $\boldsymbol{\theta}_e, \boldsymbol{\theta}_c$, and $\boldsymbol{\theta}_d$ and a mini-batch size of 4. The optimal values of all hyperparameters are found by minimising the total loss over the development dataset following a Monte-Carlo search. We found these optimal hyperparameter values of $\alpha = 0.99998$, $\beta = 0.00001$ and $\gamma = 0.00001$. Note that the scale of different losses are different and the absolute values of hyperparameters do *not* indicate the significance of a component loss. For example, if we rescale all losses to the same range then we have $L_c = 0.005\alpha$, $L_d = 0.269\beta$ and $L_a = 21.1999\gamma$. Therefore, debiasing (L_d) and orthogonalisation (L_a) contributions are significant.

We utilized a GeForce GTX 1080 Ti. The debiasing is completed in less than an hour because our method is only a fine-tuning technique. The parameter size of our debiasing model is 270,900.

4.3 Evaluation Datasets

We use the following datasets to evaluate the degree of the biases in word embeddings.

WEAT: Word Embedding Association Test (WEAT; Caliskan et al., 2017), quantifies various biases (e.g. gender, race and age) using semantic similarities between word embeddings. It compares two same size sets of *target* words $\mathcal{X}$ and $\mathcal{Y}$ (e.g. European and African names), with two sets of *attribute* words $\mathcal{A}$ and $\mathcal{B}$ (e.g. *pleasant* vs. *unpleasant*). The bias score, $s(\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B})$, for each target is calculated as follows:

$$s(\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}) = \sum_{\boldsymbol{x} \in \mathcal{X}} k(\boldsymbol{x}, \mathcal{A}, \mathcal{B}) - \sum_{\boldsymbol{y} \in \mathcal{Y}} k(\boldsymbol{y}, \mathcal{A}, \mathcal{B}) \quad (8)$$

$$k(\boldsymbol{t}, \mathcal{A}, \mathcal{B}) = \text{mean}_{\boldsymbol{a} \in \mathcal{A}} f(\boldsymbol{t}, \boldsymbol{a}) - \text{mean}_{\boldsymbol{b} \in \mathcal{B}} f(\boldsymbol{t}, \boldsymbol{b}) \quad (9)$$

Here, f is the cosine similarity between the word embeddings. The one-sided p-value for the permutation test regarding $\mathcal{X}$ and $\mathcal{Y}$ is calculated as the probability of $s(\mathcal{X}_i, \mathcal{Y}_i, \mathcal{A}, \mathcal{B}) > s(\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B})$. The effect size is calculated as the normalised measure given by (10).

$$\frac{\text{mean}_{x \in \mathcal{X}} s(x, \mathcal{A}, \mathcal{B}) - \text{mean}_{y \in \mathcal{Y}} s(y, \mathcal{A}, \mathcal{B})}{\text{sd}_{t \in \mathcal{X} \cup \mathcal{Y}} s(t, \mathcal{A}, \mathcal{B})} \quad (10)$$

---

[3] https://code.google.com/archive/p/word2vec/

[4] https://github.com/stanfordnlp/GloVe

[5] https://fasttext.cc/docs/en/english-vectors.html

WAT: Word Association Test (WAT) is a method to measure gender bias over a large set of words (Du et al., 2019). It calculates the gender information vector for each word in a word association graph created with Small World of Words project (SWOWEN; Deyne et al., 2019) by propagating information related to masculine and feminine words $(w_m^i, w_f^i) \in \mathcal{L}$ using a random walk approach (Zhou et al., 2003). The gender information is represented as a 2-dimensional vector (b_m, b_f), where b_m and b_f denote respectively the masculine and feminine orientations of a word. The gender information vectors of masculine words, feminine words and other words are initialised respectively with vectors (1, 0), (0, 1) and (0, 0). The bias score of a word is defined as $\log(b_m/b_f)$. We evaluate the gender bias of word embeddings using the Pearson correlation coefficient between the bias score of each word and the score given by (11) computed as the averaged difference of cosine similarities between masculine and feminine words.

$$\frac{1}{|\mathcal{L}|}\sum_{i=1}^{|\mathcal{L}|}\left(f(w, w_m^i) - f(w, w_f^i)\right) \qquad (11)$$

SemBias: SemBias dataset (Zhao et al., 2018b) contains three types of word-pairs: (a) **Definition**, a gender-definition word pair (e.g. hero – heroine), (b) **Stereotype**, a gender-stereotype word pair (e.g., manager – secretary) and (c) **None**, two other word-pairs with similar meanings unrelated to gender (e.g., jazz – blues, pencil – pen). We use the cosine similarity between the $\overrightarrow{he} - \overrightarrow{she}$ gender directional vector and $\boldsymbol{a} - \boldsymbol{b}$ in above word pair (a, b) lists to measure gender bias. Zhao et al. (2018b) used a subset of 40 instances associated with 2 seed word-pairs, not used in the training split, to evaluate the generalisability of a debiasing method. For unbiased word embeddings, we expect high similarity scores in **Definition** category and low similarity scores in **Stereotype** and **None** categories.

WinoBias/OntoNotes: We use the WinoBias dataset (Zhao et al., 2018a) and OntoNotes (Weischedel et al., 2013) for coreference resolution to evaluate the effectiveness of our proposed debiasing method in a downstream task. WinoBias contains two types of sentences that require linking gendered pronouns to either male or female stereotypical occupations. In **Type 1**, co-reference decisions must be made using world knowledge about some given circumstances. However, in **Type 2**, these tests can be resolved using syntactic information and understanding of the pronoun. It involves two conditions: the pro-stereotyped (**pro**) condition links pronouns to occupations dominated by the gender of the pronoun, and the anti-stereotyped (**anti**) condition links pronouns to occupations not dominated by the gender of the pronoun. For a correctly debiased set of word embeddings, the difference between **pro** and **anti** is expected to be small. We use the model proposed by Lee et al. (2017) and implemented in AllenNLP (Gardner et al., 2017) as the coreference resolution method.

We used a bias comparing code[6] to evaluate **WEAT** dataset. Since the **WAT** code was not published, we contacted the authors to obtain the code and used it for evaluation. We used the evaluation code from GP-GloVe[7] to evaluate **SemBias** dataset. We used AllenNLP[8] to evaluate **WinoBias** and **OntoNotes** datasets. We used *evaluate_word_pairs* function and *evaluate_word_analogies* in gensim[9] to evaluate **word embedding benchmarks**.

5 Results

5.1 Overall Results

We initialise the word embeddings of the model by original (**Org**) and debiased (**Deb**) word embeddings and compare the coreference resolution accuracy using F1 as the evaluation measure.

In Table 1, we show the WEAT bias effects for cosine similarity and correlation on WAT dataset using the Pearson correlation coefficient. We see that the proposed method can significantly debias for various biases in all word embeddings in both WEAT and WAT. Especially in Word2Vec and fastText, almost all biases are debiased.

Table 2 shows the percentages where a word-pair is correctly classified as Definition, Stereotype or None. We see that our proposed method succesfully debiases word embeddings based on results on **Definition** and **Stereotype** in SemBias. In addition, we see that the SemBias-subset can be debiased for Word2Vec and fastText.

Table 3 shows the performance on WinoBias for **Type 1** and **Type 2** in **pro** and **anti** stereotypical

---

[6] `https://github.com/hljames/compare-embedding-bias`

[7] `https://github.com/kanekomasahiro/gp_debias`

[8] `https://github.com/allenai/allennlp`

[9] `https://github.com/RaRe-Technologies/gensim`

Embeddings	Word2Vec Org/Deb	GloVe Org/Deb	fastText Org/Deb
T1: flowers vs. insects	$1.46^\dagger/\mathbf{1.35}^\dagger$	$\mathbf{1.48}^\dagger/1.54^\dagger$	$1.29^\dagger/\mathbf{1.09}^\dagger$
T2: instruments vs. weapons	$1.56^\dagger/\mathbf{1.43}^\dagger$	$1.49^\dagger/\mathbf{1.41}^\dagger$	$1.56^\dagger/\mathbf{1.34}^\dagger$
T3: European vs. African American names	$0.46^\dagger/\mathbf{0.16}^\dagger$	$1.33^\dagger/\mathbf{1.04}^\dagger$	$0.79^\dagger/\mathbf{0.46}^\dagger$
T4: male vs. female	$1.91^\dagger/\mathbf{1.87}^\dagger$	$1.86^\dagger/\mathbf{1.85}^\dagger$	$1.65^\dagger/\mathbf{1.42}^\dagger$
T5: math vs. art	$0.85^\dagger/\mathbf{0.53}^\dagger$	$\mathbf{0.43}^\dagger/0.82^\dagger$	$1.14^\dagger/\mathbf{0.86}^\dagger$
T6: science vs. art	$1.18^\dagger/\mathbf{0.96}^\dagger$	$\mathbf{1.21}^\dagger/1.44^\dagger$	$1.16^\dagger/\mathbf{0.88}^\dagger$
T7: physical vs. mental conditions	0.90/**0.57**	1.03/**0.98**	0.83/**0.63**
T8: older vs. younger names	$-\mathbf{0.08}/-0.10$	$1.07^\dagger/\mathbf{0.92}^\dagger$	-0.32/$-\mathbf{0.13}$
T9: WAT	$0.48^\dagger/\mathbf{0.45}^\dagger$	$0.59^\dagger/\mathbf{0.58}^\dagger$	$0.54^\dagger/\mathbf{0.51}^\dagger$

Table 1: Rows T1-T8 show WEAT bias effects for the cosine similarity and row T9 shows the Pearson correlations on the WAT dataset with cosine similarity. † indicates bias effects that are insignificant at $\alpha < 0.01$.

Embeddings	Word2Vec Org/Deb	GloVe Org/Deb	fastText Org/Deb
definition	83.0/**83.9**	83.0/**83.4**	92.0/**93.2**
stereotype	13.4/**12.3**	12.0/**11.4**	5.5/**4.3**
none	**3.6**/3.9	**5.0**/5.2	**2.5/2.5**
sub-definition	50.0/**57.5**	**67.5/67.5**	82.5/**85.0**
sub-stereotype	40.0/**32.5**	**27.5/27.5**	12.5/**10.0**
sub-none	**10.0/10.0**	**5.0/5.0**	**5.0/5.0**

Table 2: Prediction accuracies for gender relational analogies on SemBias.

Embeddings	Word2Vec Org/Deb	GloVe Org/Deb	fastText Org/Deb
Type 1-pro	70.1/69.4	70.8/69.5	70.1/69.7
Type 1-anti	49.9/50.5	50.9/52.1	52.0/51.6
Avg	**60.0/60.0**	**60.9**/60.8	**61.1**/60.7
Diff	20.2/**18.9**	19.9/**17.4**	**18.1/18.1**
Type 2-pro	84.7/83.7	79.6/78.9	83.8/82.5
Type 2-anti	77.9/77.5	66.0/66.4	75.1/76.4
Avg	**81.3**/80.6	**72.8**/72.7	**79.5/79.5**
Diff	6.8/**6.2**	13.6/**12.5**	8.7/**6.1**
OntoNotes	62.6/**62.7**	62.5/**62.9**	63.3/**63.4**

Table 3: F1 on OntoNotes and WinoBias test set. WinoBias results have Type-1 and Type-2 in pro and anti stereotypical conditions. Average (Avg) and difference (Diff) of anti and pro stereotypical scores are shown.

conditions. In most settings, the diff is smaller for the debiased than the original word embeddings, which demonstrates the effectiveness of our proposed method. From the results for Avg, we see that debiasing is achieved with almost no loss in performance. In addition, the debiased scores on the OntoNotes are higher than the original scores for all word embeddings.

	GloVe	HD	GN-GloVe	GP-GloVe	Ours
T1	$0.89^\dagger$	$0.97^\dagger$	$1.10^\dagger$	$1.24^\dagger$	$\mathbf{0.74}^\dagger$
T2	$1.25^\dagger$	$1.23^\dagger$	$1.25^\dagger$	$1.31^\dagger$	$\mathbf{1.22}^\dagger$
T5	0.49	-0.40	**0.00**	0.21	0.35
T6	$1.22^\dagger$	**-0.11**	$1.13^\dagger$	$0.78^\dagger$	$1.05^\dagger$
T7	1.19	1.23	1.11	**1.01**	1.03

Table 4: WEAT bias effects for the cosine similarity on prior methods and proposed method. † indicates bias effects that are insignificant at $\alpha < 0.01$. T* are aligned with those in Table 1.

5.2 Comparison with Existing Methods

We compare the proposed method against the existing debiasing methods (Bolukbasi et al., 2016; Zhao et al., 2018b; Kaneko and Bollegala, 2019) mentioned in § 2 on WEAT, which contains different types of biases. We debias Glove[10], which is used in Zhao et al. (2018b). All word embeddings used in these experiments are the pre-trained word embeddings used in the existing debiasing methods. Words in evaluation sets T3, T4 and T8 are not covered by the input pre-trained embeddings and hence not considered in this evaluation. From Table 4 we see that only the proposed method debiases all biases accurately. T5 and T6 are the tests for gender bias; despite prior debiasing methods do well in those tasks, they are not able to address other types of biases. Notably, we see that the proposed method can debias more accurately compared to previous methods that use word lists for gender debiasing, such as Bolukbasi et al. (2016) in T5 and Zhao et al. (2018b) in T6.

5.3 Dominant Gloss vs All Glosses

In Table 5, we investigate the effect of using the dominant gloss (i.e. the gloss for the most frequent

---

[10] `https://github.com/uclanlp/gn_glove`

Embeddings	Word2Vec Dom/All	GloVe Dom/All	fastText Dom/All
definition	83.4/**83.9**	**83.9**/83.4	92.5/**93.2**
stereotype	12.7/**12.3**	11.8/**11.4**	4.8/**4.3**
none	**3.9**/**3.9**	**4.3**/5.2	2.7/**2.5**
sub-definition	55.0/**57.5**	**67.5**/**67.5**	77.5/**85.0**
sub-stereotype	35.0/**32.5**	**27.5**/**27.5**	12.5/**10.0**
sub-none	**10.0**/**10.0**	**5.0**/**5.0**	10.0/**5.0**

Table 5: Performance obtained when using only the dominant gloss (Dom) or all glosses (All) on SemBias.

Embeddings	Word2Vec Org/Deb	GloVe Org/Deb	fastText Org/Deb
WS	**62.4**/60.3	60.6/**68.9**	64.4/**67.0**
SIMLEX	44.7/**46.5**	39.5/**45.1**	44.2/**47.3**
RG	75.4/**77.9**	68.1/**74.1**	75.0/**79.6**
MTurk	63.1/**63.6**	62.7/**69.4**	67.2/**69.9**
RW	75.4/**77.9**	68.1/**74.1**	75.0/**79.6**
MEN	68.1/**69.4**	67.7/**76.7**	67.6/**71.8**
MSR	**73.6**/72.6	73.8/**75.1**	**83.9**/80.5
Google	**74.0**/73.7	76.8/**77.3**	**87.1**/85.7

Table 6: The Spearman correlation coefficients between human ratings and cosine similarity scores computed using word embeddings for the word pairs in semantic similarity benchmarks.

sense of the word) when creating $s(w)$ on SemBias benchmark as opposed to using all glosses (same as in Table 2). We see that debiasing using all glosses is more effective than using only the dominant gloss.

5.4 Word Embedding Benchmarks

It is important that a debiasing method removes only discriminatory biases and preserves semantic information in the original word embeddings. If the debiasing method removes more information than necessary from the original word embeddings, performance will drop when those debiased embeddings are used in NLP applications. Therefore, to evaluate the semantic information preserved after debiasing, we use semantic similarity and word analogy benchmarks as described next.

Semantic Similarity: The semantic similarity between two words is calculated as the cosine similarity between their word embeddings and compared against the human ratings using the Spearman correlation coefficient. The following datasets are used: Word Similarity 353 (**WS**; Finkelstein et al., 2001), **SimLex** (Hill et al., 2015), Rubenstein-Goodenough (**RG**; Rubenstein and Goodenough, 1965), **MTurk** (Halawi et al., 2012), rare words (**RW**; Luong et al., 2013) and **MEN** (Bruni et al., 2012).

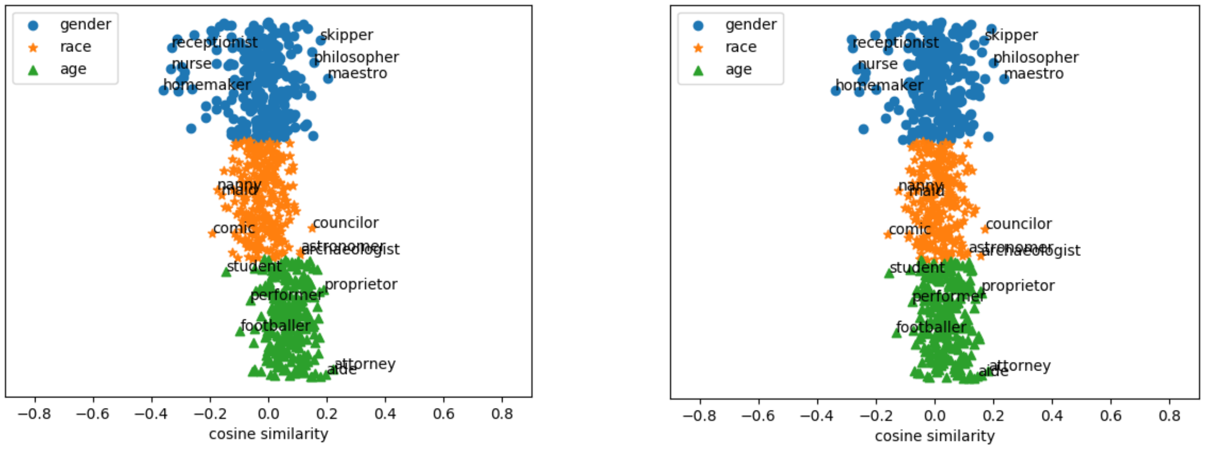
Word Analogy: In word analogy, we predict d that completes the proportional analogy "a is to b as c is to what?", for four words a, b, c and d. We use CosAdd (Levy and Goldberg, 2014), which determines d by maximising the cosine similarity between the two vectors ($\boldsymbol{b} - \boldsymbol{a} + \boldsymbol{c}$) and $\boldsymbol{d}$. Following Zhao et al. (2018b), we evaluate on **MSR** (Mikolov et al., 2013c) and **Google** analogy datasets (Mikolov et al., 2013a) as shown in Table 6.

From Table 6 we see that for all word embeddings, debiased using the proposed method accurately preserves the semantic information in the original embeddings. In fact, except for Word2Vec embeddings on WS dataset, we see that the accuracy of the embeddings have *improved* after the debiasing process, which is a desirable side-effect. We believe this is due to the fact that the information in the dictionary definitions is used during the debiasing process. Overall, our proposed method removes unfair biases, while retaining (and sometimes further improving) the semantic information contained in the original word embeddings.

We also see that for GloVe embeddings the performance has improved after debiasing whereas for Word2Vec and fastText embeddings the opposite is true. Similar drop in performance in word analogy tasks have been reported in prior work (Zhao et al., 2018b). Besides CosAdd there are multiple alternative methods proposed for solving analogies using pre-trained word embeddings such as CosMult, PairDiff and supervised operators (Bollegala et al., 2015, 2014; Hakami et al., 2018). Moreover, there have been concerns raised about the protocols used in prior work evaluating word embeddings on word analogy tasks and the correlation with downstream tasks (Schluter, 2018). Therefore, we defer further investigation in this behaviour to future work.

5.5 Visualising the Outcome of Debiasing

We analyse the effect of debiasing by calculating the cosine similarity between neutral occupational words and gender ($\overrightarrow{he} - \overrightarrow{she}$), race ($\overrightarrow{Caucasoid} - \overrightarrow{Negroid}$) and age ($\overrightarrow{elder} - \overrightarrow{youth}$) directions. The neutral occupational words list is based on Bolukbasi et al. (2016) and is listed in the Supplementary. Figure 1 shows the visualisation result for Word2Vec. We see that original Word2Vec shows some gender words are especially away from the



(a) Original Word2Vec

(b) Debiased Word2vec

Figure 1: Cosine similarity between neutral occupation words for vector directions on gender ($\overrightarrow{he} - \overrightarrow{she}$), race ($\overrightarrow{Caucasoid} - \overrightarrow{Negroid}$), and age ($\overrightarrow{elder} - \overrightarrow{youth}$) vectors.

origin (0.0). Moreover, age-related words have an overall bias towards "elder". Our debiased Word2Vec gathers vectors around the origin compared to the original Word2Vec for all gender, race and age vectors.

On the other hand, there are multiple words with high cosine similarity with the female gender after debiasing. We speculate that in rare cases their definition sentences contain biases. For example, in the WordNet the definitions for "homemaker" and "nurse" include gender-oriented words such as "a wife who manages a household while her husband earns the family income" and "a woman who is the custodian of children." It remains an interesting future challenge to remove biases from dictionaries when using for debiasing. Therefore, it is necessary to pay attention to biases included in the definition sentences when performing debiasing using dictionaries. Combining definitions from multiple dictionaries could potentially help to mitigate biases coming from a single dictionary. Another future research direction is to evaluate the proposed method for languages other than English using multilingual dictionaries.

6 Conclusion

We proposed a method to remove biases from pre-trained word embeddings using dictionaries, without requiring pre-defined word lists. The experimental results on a series of benchmark datasets show that the proposed method can remove unfair biases, while retaining useful semantic information encoded in pre-trained word embeddings.

References

Mohammed Alsuhaibani, Takanori Maehara, and Danushka Bollegala. 2019. Joint learning of hierarchical word embeddings from a corpus and a taxonomy. In *Proc. of the Automated Knowledge Base Construction Conference*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proc. of ICLR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Danushka Bollegala, Takanori Maehara, and Ken-ichi Kawarabayashi. 2015. Embedding semantic relationas into word representations. In *Proc. of IJCAI*, pages 1222 – 1228.

Danushka Bollegala, Takanori Maehara, Yuichi Yoshida, and Ken-ichi Kawarabayashi. 2014. Learning word representations from relational graphs. In *Proc. of 29th AAAI Conference on Aritificial Intelligence (AAAI 2015)*, pages 2146 – 2152.

Danushka Bollegala, Alsuhaibani Mohammed, Takanori Maehara, and Ken-ichi Kawarabayashi. 2016. Joint word representation learning using a corpus and a semantic lexicon. In *Proc. of AAAI*, pages 2690–2696.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NIPS*.

Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Student Research Workshop, pages 7–15, Minneapolis, Minnesota. Association for Computational Linguistics.

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *EMNLP*.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145. Association for Computational Linguistics.

A. Budanitsky and G. Hirst. 2006. Evaluating wordnet-based measures of semantic distance. *Computational Linguistics*, 32(1):13–47.

Alexander Budanitsky and Graeme Hirst. 2001. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *NAACL 2001*.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

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

Simon De Deyne, Danielle J. Navarro, Amy Perfors, Marc Brysbaert, and Gert Storms. 2019. The "small world of words" english word association norms for over 12,000 cue words. *Behavior Research Methods*, 51:987–1006.

Yupei Du, Yuanbin Wu, and Man Lan. 2019. Exploring human gender stereotypes with word association test. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6133–6143, Hong Kong, China. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial Removal of Demographic Attributes from Text Data. In *Proc. of EMNLP*.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1696–1705, Florence, Italy. Association for Computational Linguistics.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 406–414, New York, NY, USA. ACM.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Goran Glavaš and Ivan Vulić. 2018. Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–45. Association for Computational Linguistics.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 60–63, Florence, Italy. Association for Computational Linguistics.

Anthony G. Greenwald, Debbie E. McGhee, and Jordan L. K. Schwatz. 1998. Measuring individual differences in implicit cognition: The implicit association test. *Journal of Personality and Social Psychology*, 74(6):1464–1480.

Huda Hakami, Kohei Hayashi, and Danushka Bollegala. 2018. Why does PairDiff work? - a mathematical analysis of bilinear relational compositional operators for analogy detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2493–2504, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1406–1414, New York, NY, USA. ACM.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5266–5274, Hong Kong, China. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *10th Intl. Conf. Research on Computational Linguistics (ROCLING)*, pages 19 – 33.

Masahiro Kaneko and Danushka Bollegala. 2019. Gender-preserving debiasing for pre-trained word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2020. Autoencoding improves pre-trained word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1699–1713, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Anja Lambrecht and Catherine E. Tucker. 2016. Algorithmic bias? an empirical study into apparent gender-based discrimination in the display of stem career ads. *SSRN Electronic Journal*.

Anne Lauscher and Goran Glavas. 2019. Are we consistently biased? multidimensional analysis of biases in distributional word vectors. In **SEM@NAACL-HLT*.

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

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *CoNLL*.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30. Association for Computational Linguistics.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113. Association for Computational Linguistics.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 615–621, Minneapolis, Minnesota. Association for Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013a. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continous space word representations. In *NAACL-HLT*, pages 746 – 751.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Jeffery Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: global vectors for word representation. In *EMNLP*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null It Out: Guarding Protected Attributes by Iterative Nullspace Projection. In *Proc. of ACL*.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14. Association for Computational Linguistics.

Natalie Schluter. 2018. The word analogy testing caveat. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 242–246. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3405–3410, Hong Kong, China. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

Julien Tissier, Christopher Gravier, and Amaury Habrard. 2017. Dict2vec : Learning word embeddings using lexical dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Copenhagen, Denmark. Association for Computational Linguistics.

Eva Vanmassenhove, Christian Hardmeier, and Andy Way. 2018. Getting gender right in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3003–3008. Association for Computational Linguistics.

Jesse Vig. 2019. Visualizing Attention in Transformer-Based Language Representation Models.

Koki Washio, Satoshi Sekine, and Tsuneaki Kato. 2019. Bridging the defined and the defining: Exploiting implicit lexical semantic relations in definition modeling. In *EMNLP/IJCNLP*.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Proc. of NIPS*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634, Minneapolis, Minnesota. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2941–2951, Copenhagen, Denmark. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018a. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20. Association for Computational Linguistics.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018b. Learning Gender-Neutral Word Embeddings. In *Proc. of EMNLP*, pages 4847–4853.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2003. Learning with local and global consistency. In *NIPS*.

Ran Zmigrod, Sebastian J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual Data Augmentation for Mitigating Gender Stereotypes in Languages with Rich Morphology. In *Proc. of ACL*.

Belief-based Generation of Argumentative Claims

Milad Alshomary **Wei-Fan Chen** **Timon Gurcke** **Henning Wachsmuth**
Department of Computer Science
Paderborn University, Paderborn, Germany
`<first name>.<last name>@upb.de`

Abstract

When engaging in argumentative discourse, skilled human debaters tailor claims to the audience's beliefs to construct effective arguments. Recently, the field of computational argumentation witnessed extensive effort to address the automatic generation of arguments. However, existing approaches do not perform any audience-specific adaptation. In this work, we aim to bridge this gap by studying the task of *belief-based claim generation:* Given a controversial topic and a set of beliefs, generate an argumentative claim tailored to the beliefs. To tackle this task, we model the people's prior beliefs through their stances on controversial topics and extend state-of-the-art text generation models to generate claims conditioned on the beliefs. Our automatic evaluation confirms the ability of our approach to adapt claims to a set of given beliefs. In a manual study, we also evaluate the generated claims in terms of informativeness and their likelihood to be uttered by someone with a respective belief. Our results reveal the limitations of modeling users' beliefs based on their stances. Still, they demonstrate the potential of encoding beliefs into argumentative texts, laying the ground for future exploration of audience reach.

1 Introduction

According to van Eemeren and Houtlosser (1999), debaters engaging in argumentative discourse, aimed to resolve a disagreement, design their next argumentative move considering the topical potential, the audience demand, and appropriate presentational devices. Feinberg and Willer (2015) stress based on the moral foundation theory (Godden, 2010) how phrasing arguments to fit the audience's morals leads to a better agreement. For example, in a debate on former US president *Donald Trump*, potential topics could have been *immigration*, *health care plans*, *tax plans*, etc. However, knowledge about the audience being middle-class workers would have restricted the selection to *Trump's tax plans*. Appropriate usage of presentational devices may have then put a con argument as follows:

Example *"Donald Trump was a bad president. He did nothing but hurt the poor and middle class. His tax plan benefited only rich people who could afford it."*

There is a recent growth of interest in argument generation as a subfield of computational argumentation. Several tasks have been proposed, including claim negation (Bilu et al., 2015; Hidey and McKeown, 2019), counterargument generation (Hua et al., 2019), and conclusion generation (Alshomary et al., 2020). While some research considers argumentative strategies when delivering arguments (Wachsmuth et al., 2018; El Baff et al., 2019), no one has worked on adapting arguments to user beliefs yet. Our goal is to bridge this gap.

In this work, we propose to extend argument generation technologies with the ability to encode beliefs. This not only reflects the process by which humans reason, but it also allows controlling the output, to better reach the audience. In particular, we introduce the task of *belief-based claim generation:* Given a controversial topic and a representation of a user's beliefs, generate a claim relevant to the topic and matches the beliefs.

To approach this task, we first model user beliefs by their stances (pro or con) on a set of controversial topics, and then extend two state-of-the-art text generation approaches by conditioning their output on a specific set of beliefs. One approach builds on Li et al. (2016), equipping a sequence-to-sequence (Seq2Seq) model with a context vector representing the given stances. The other approach controls the output of a pre-trained argumentative language model (LM) using the algorithm of Dathathri et al. (2020) to ensure resembling the user's beliefs. We

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 224–233
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

study the given task empirically on the *debate.org* dataset of Durmus and Cardie (2018). The dataset contains users' arguments on various controversial topics and their stances towards the most popular topics on the website, called the *big issues*. For our purposes, we use these big issues as the controversial topics, and we model beliefs by the user's stances towards them.

In our automatic evaluation, we compare both models against their unconditioned correspondents (i.e., the same models without knowledge about a user). We assess the generated claims in terms of the similarity to the ground truth and the likelihood of carrying textual features that reflect users' stances on big issues. Our results suggest that using users' beliefs significantly increases the effectiveness of the Seq2Seq and LM in most cases. Moreover, a stance classifier trained on claims generated by the conditioned LM achieves the best-averaged accuracy across all big issues.

In a subsequent manual evaluation, we find that claims generated by the conditioned LM are more informative regarding the topic. In terms of predicting stance from generated claims, we analyze the limitations of our approach in detail, which lie in the belief encoding step. By avoiding these limitations, we find that the generated claims enable the annotators to predict correctly a stance on a given big issue in 45% of the cases (26% incorrectly). These results demonstrate the applicability of encoding a user's beliefs into argumentative texts, enabling future research on the effect of belief-based argumentative claims on audiences.

The contribution of this work is threefold[1]:

- A new task, *belief-based claim generation*.
- An approach to model and match users' beliefs in the generation of arguments.
- Empirical evidence of the applicability of encoding beliefs into argumentative texts.

2 Related Work

Early research on argument generation aimed to create argumentative texts starting from a symbolic representation (Zukerman et al., 2000; Grasso et al., 2000; Carenini and Moore, 2006). Conceptually, those approaches all had a similar architecture consisting of three main phases: text planning, sentence planning, and realization (Stede et al., 2018). While they included a user model to a certain extent and aimed to generate convincing arguments, they were still performed on a limited scale.

With the tremendous advances of NLP and machine learning since then, research has begun to address different tasks in the realm of argument generation, showing promising results. Hua et al. (2019) proposed a neural network-based framework for generating counter-arguments. Both Bilu et al. (2015) and Hidey and McKeown (2019) addressed the task of claim negation, using a rule-based and a neural approach respectively. Also, Sato et al. (2015) proposed an approach to argument generation based on sentence retrieval, in which, given a topic, a set of paragraphs covering different aspects is generated. However, these approaches are agnostic to the target audience.

Chen et al. (2018) modified the political bias of (often claim-like) news headlines using style transfer, accounting for general political sides (left and right) at least. Moreover, Wachsmuth et al. (2018) modeled rhetorical strategies in argument synthesis conceptually, but its computational realization (El Baff et al., 2019) considers the audience implicitly only, using a language model approach to select and arrange argumentative discourse units that are phrased in an argument.

In the field of conversational AI, researchers have utilized machine translation techniques to tackle the task of dialog generation (Ritter et al., 2011). Li et al. (2016) worked on augmenting sequence-to-sequence models by learning persona vectors from the given data. In a similar fashion, one of our approaches extends such a model by a context vector representing a user's belief. Here, however, we deal with argumentative text.

Progress in the field of text generation has been made due to the availability of large pre-trained language models (Devlin et al., 2018; Solaiman et al., 2019). While these models excel in generating coherent texts, ensuring a generated text possesses a certain property is not straightforward. Some research tackled this limitation, offering ways to better control the output (Keskar et al., 2019; Ziegler et al., 2019). One of the most flexible of such approaches is by Dathathri et al. (2020), which does not require fine-tuning for each controlling theme. Their algorithm conditions the output of a language model to contain certain properties defined by a discriminative classifier or a bag-of-words. One of our approaches makes use of this algorithm to condi-

---

[1] Code can be found under: http://www.github.com/webis-de/eacl21-belief-based-claim-generation

tion the output of an argumentative language model on a bag-of-words that represents a user's beliefs. A recent relevant work by Schiller et al. (2020) deals with the generation of aspect-controlled arguments. Similar to us, the authors utilize a pre-trained language model to generate arguments on a specific topic, with a controlled stance and aspect. Their focus is on topical aspects of arguments, though, and their approach based on Keskar et al. (2019) is limited to a predefined set of topics and aspects.

3 Task

Due to the importance of audience in argumentation when aiming for persuasiveness (van Eemeren and Houtlosser, 1999), and due to the fact that humans comply to certain morals that shape their beliefs and affect their reasoning (Godden, 2010; Feinberg and Willer, 2015), we introduce the audience's beliefs as a new dimension to the argument generation process in this work. For this, we propose a new task, belief-based claim generation:

> *Given a controversial topic and a representation of the audience's beliefs, generate a claim that is both relevant to the topic and matches the beliefs.*

We focus this task on generating *claims* rather than full arguments to keep it simple and because claims denote the main units from which arguments are built. As shown by Feinberg and Willer (2015), better agreement is achieved when arguments are framed with respect to audience's beliefs. Therefore, we argue that studying the mentioned task will enable argumentation technology, knowing its audience, to generate more convincing arguments, bridging the gap between disagreeing parties.

3.1 Data

To study the proposed task, a dataset is needed in which information about users revealing their beliefs as well as their arguments on various topics are given. Here, we build upon the dataset introduced by Durmus and Cardie (2018), which was collected from *debate.org*, an online platform where users can engage in debates over controversial topics and share their profiles. The dataset contains users' arguments as answers to topic questions and engagement in debates, along with various user information, including a user's self-specified stances (pro or con) on up to 48 predefined popular controversial topics, called *big issues*.

Dataset	**# Claims**	**# Topics**	**# Users**
Training set	41 288	22 241	5 189
Validation set	5 028	2 450	2 509
Test set	5 154	2 728	2 512
Full dataset	51 470	27 419	5 189

Table 1: Number of claims, topics, and users in each of the training, validation, and test set of the data used in this paper.

In our dataset, for the task at hand, we keep only users who have at least three arguments and stated their stance on at least one of the big issues. For those, we collected their arguments along with the topics and stances. In total, the dataset contains around 51k claims, on 27k topics from 5k users. We randomly split the dataset per topic into 10% test and 90% training. 10% of the latter are used as the validation set. Statistics are given in Table 1.

To develop approaches to the belief-based claim generation task, we need training data where claims can be identified as such. Since claim detection is not our focus, we preprocess all data using the claim detection approach of Chakrabarty et al. (2019). In particular, we score the likelihood of each sentence being a claim, and only keep the one with the highest score as the user's claim on the topic. To evaluate the model, we created a sample of 100 arguments, and two annotators decided whether the extracted sentence represents a claim on the given topic or not. In terms of full agreement, the model extracted claims correctly in 81% of the cases, the Cohen's κ inter-annotator agreement being 0.3. We note that this preprocessing step produces some noise in the data, mainly affecting the training of our Seq2Seq model below.

4 Approach

To study our research question, we propose and compare two approaches that build on top of known techniques for text generation. Both approaches rely on modeling users' beliefs via their stances on big issues. The first is an extension of the Seq2Seq model (Sutskever et al., 2014), where the user's stances are encoded as a context vector, while the second conditions the output of a pre-trained argumentative language model via a bag-of-words, constructed based on stances on big issues.

4.1 Seq2Seq-based Model

Given a topic, as a sequences of words $T = (w_1, w_2, ..., w_n)$, a user vector $\overrightarrow{U} \in \{0, 1\}^k$ with k being the number of big issues, and a claim as a sequence of words $C = (w_1, w_2, ..., w_m)$, first an LSTM-based encoder consumes the input topic and produces a hidden state $\overrightarrow{h}$, which is used to initialize the LSTM-based decoder. The user vector $\overrightarrow{U}$ is projected into a new embedding space via a feed forward network with a learned weight matrix W_U, producing a new vector, $\overrightarrow{V}$:

$$\overrightarrow{V} = \sigma(W_U \cdot \overrightarrow{U})$$

Following Li et al. (2016), our $\overrightarrow{V}$ is served as their speaker embedding in the model. The difference between the speaker model in Li et al. (2016) and this model is that the vector $\overrightarrow{V}$ is not explicitly predefined but rather learned from the data, while in our model it is already predefined as a binary vector representing the user's stances on big issues. By augmenting the Seq2Seq model with a context user vector, the model is supposed to capture the correlation between users' stances on big issue and the corresponding claims. Once the correlation is learned, the model can generate a claim utilizing not only the topic, but also the stances on big issues of the target user, which reflect the beliefs.

4.2 Conditioned Language Model

In this approach, we represent a user's stances on big issues as a bag-of-words. We then use the topic as a prompt for a pre-trained argumentative language model (LM) to synthesize a claim conditioned using the algorithm of Dathathri et al. (2020). The synthesis process is illustrated in Figure 1.

Argumentative Language Model Since we aim to generate *claims* in particular, a standard LM is not enough. To model argumentative language, we take a LM pre-trained on general language and fine-tune it on a large set of arguments (in our experiments, we use the corpus of Ajjour et al. (2019)). The result is an LM that is able to generate argumentative text.

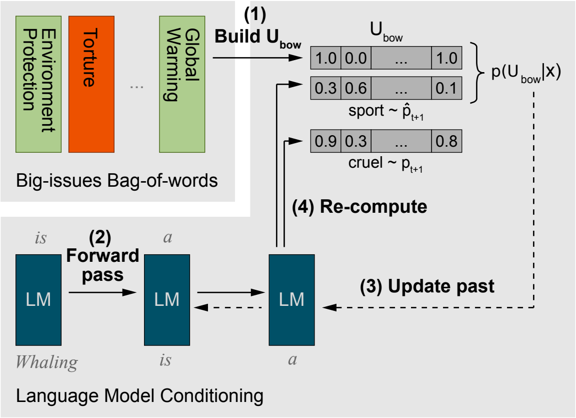


Figure 1: The synthesis process of the conditioned LM on the topic "Whaling", given a user who is pro *environmental protection* and *global warming* and con *torture*. Steps: (1) Building U_{bow}, based on stances (2) Forward pass through the LM to generate a token, *sport* (3) Updating the LM history H_t, based on $p(U_{bow}|x)$, and (4) Generating from the new history $\hat{H}_t$ a new token *cruel*.

Belief-based Bag-of-words Next, we build a bag-of-words that represents the beliefs of a user. We learn this from the user's stances on the big issues. For example, a user pro *abortion* would likely be pro *choice*. Hence, words such as *right* and *choice* are candidates to be included in their belief-based bag-of-words. To this end, we first build two bag-of-words representations for each big issue, one for the pro and for the con side. For a user, we then construct a belief-based bag-of-words based on their stances on big issues.

To build a representative pro and con bag-of-words for each big issue, we follow the topic signature approach of Lin and Hovy (2000). Given a big issue, we first collect from some corpus of arguments three sets: relevant pro arguments R_{pro}, relevant con arguments R_{con}, and a random set of non-relevant arguments $\hat{R}$. For each relevant set, we then compute a likelihood ratio for all its words with respect to $\hat{R}$ and keep only words with a score higher than a specific threshold τ, resulting in two sets of words, W_{pro} and W_{con}. Since a word may appear in both sets, we remove it from the set where it occurs fewer times. Finally, we sort words according to their likelihood ratio and keep in both W_{pro} and W_{con} the top k words, forming the final pro and con bag-of-words respectively.

Claim Generation Given a user (represented by stances on big issues) and a topic, we construct a belief-based bag-of-words (Step 1 in Figure 1):

$$U_{bow} = W_1 \cup W_2 \cup \ldots \cup W_n$$

where W_i is the pro bag-of-words if the stance is pro and the con bag-of-words otherwise. Then, we use the topic as a prompt and the user's bag-of-words U_{bow} to condition the generated claim (see Figure 1). In particular, given a transformer-based LM (Vaswani et al., 2017), a token x_{t+1} is generated at each time step as follows:

$$o_{t+1}, H_{t+1} = LM(x_t, H_t)$$

$$x_{t+1} \sim p_{t+1} = Softmax(W \cdot o_{t+1})$$

where H_t represents the history of the LM. Using the algorithm of Radford et al. (2019), called Plug and Play LM (PPLM), an update to the past, ΔH, is computed to control the generated claim, based on the sum of the log likelihood $p(U_{bow}|x)$ of all words in the belief-based bag-of-words. Then the new history, $\hat{H}_t = H_t + \Delta H_t$, is used as in the previous equations to draw a new distribution $\hat{p}_{t+1}$, of which a new token is sampled. To ensure fluency in the generated text, ΔH is further modified to ensure a high log-likelihood $p(x)$ with respect to the LM. More details on the algorithm can be found in the work of Radford et al. (2019).

In short, through fine-tuning an LM on argumentative text, we tune it to generate claims. Using the topic as a prompt, we ensure that the claim is on the topic. Finally, the PPLM represents beliefs, modeled as a bag-of-words U_{bow}, in the claim.

5 Automatic Evaluation

In this section, we evaluate whether utilizing user's beliefs as input, modeled as stances on big issues, leads to claims that better match the ground-truth claims and reveal the input stances on big issues.

5.1 Experimental Setup

On one hand, we compute the BLEU and METEOR scores of the generated claims with respect to the ground-truth claims. On the other hand, we compute the likelihood that the generated claims possess textual features that reflect the input user's beliefs. We do so by measuring the accuracy of predicting user's stances on big issues given the generated claims. We compute this accuracy for each of the 48 big issues individually and report the results for all of them. To this end, we carry out the following three steps for a given approach.

First, we generate claims for all given users and topics in the test dataset. Second, we keep only instances in which users have a stance (pro/con) on the tested big issue, and split the filtered dataset into training and test. Finally, we train a simple TF-IDF based linear classifier on the training set to predict the stance on the big issue given the text of the claim. The accuracy of the classifier on the test split then quantifies the likelihood of the generated claims possessing textual features that reflect the stance on the corresponding big issue.

Approach	**BLEU-1**	**BLEU-3**	**METEOR**
S2S-baseline	18.2%	0.44%	**16%**
S2S-model	***__18.4%__	***0.46%**	**16%**
LM-baseline	9.6%	**0.26%**	8%
LM-conditioned	***12.0%**	0.16%	***11%**

Table 2: BLEU and METEOR scores of the claims of each evaluated approach compared to the ground-truth claims. Values marked with * are significantly better than the respective baseline at $p < .05$ (student's t-test).

5.2 Implementation Details

In the following, we give implementation details of our approaches and the corresponding baselines:

Seq2seq-based Model Based on the OpenNMT framework (Klein et al., 2017), the encoder and decoder are each two-layer LSTMs of hidden size 512 with GloVe word embeddings of size 300. Users' stances on big issues are represented as a one-hot encoded vector, and then projected into 16 dimensions space through a one-layer dense neural network. We trained the model with the Adagrad optimizer (batch size 16) and refer to it as *S2S-model*.

Conditioned Language Model We constructed the pro/con relevant argument sets (R_{pro}, R_{con}) by querying the respective big issue from the API provided by Ajjour et al. (2019) and extracting pro/con arguments from the top 60 results. For the non-relevant argument set ($\hat{R}$), we used the same corpus (Ajjour et al., 2019) and randomly selected 100 arguments. We eliminated all words with a score under $\tau = 10$ and finally kept the top $k = 25$ words from each set (R_{pro}, R_{con}) to represent the bag-of-words.[2]

To model the argumentative language, we fine-tuned the GPT-2 model on the corpus of Ajjour et al. (2019), which contains around 400k arguments. The fine-tuning was performed using the transformers framework (Wolf et al., 2019). We used the topic as a prompt to trigger the generation process. However, since some topics are phrased as a question (e.g, "is abortion wrong?"), we extracted the noun phrase from the topic and used it as a prompt. For conditioning the generated claim,

---

[2] We refrained from tuning the parameters here since we do not have a ground truth.

Approach	Abortion	Death penalty	Gay Marriage	Drug legaliz.	Global warming	Environm. protection	Medical mariju.	Smok. ban	Minim. wage	Border fence	All 48 big issues
Ground-truth	0.49	0.59	0.55	0.55	0.55	0.55	0.50	0.53	0.48	0.62	0.52
S2S-baseline	0.49	0.48	**0.52**	0.45	0.51	0.51	0.57	**0.53**	0.53	0.46	0.50
S2S-model	0.55	**0.55**	0.45	0.45	0.51	**0.58**	0.57	**0.53**	0.49	**0.52**	0.51
LM-baseline	0.48	0.50	0.54	0.49	0.54	0.56	0.51	0.45	0.59	0.46	0.50
LM-conditioned	***0.58**	*0.53	0.45	**0.56**	***0.61**	**0.58**	**0.58**	**0.53**	**0.65**	0.50	**0.54**
# Training	1 610	1 532	2 098	1 538	1 960	2 196	2 096	1 370	1 580	1 092	-
# Test	350	366	196	316	156	86	138	294	172	280	-

Table 3: Accuracy of each classifier trained on claims generated by the evaluated approaches to predict the stance, on the 10 most frequent big issues as well as on average over all 48 big issues. Values marked with * are significantly better than corresponding baseline at $p < 0.05$ according to a one-tailed Student's t-test.

we used the PPLM implementation (Dathathri et al., 2020)[3]. We call this model the *LM-conditioned*.

Baselines To evaluate the gain of encoding user's beliefs, we compare our two approaches to the corresponding version without stances on big issues as an input. We refer to these baselines as *S2S-baseline* and *LM-baseline* respectively[4].

5.3 Results

Table 2 shows the results of our approaches and the baselines in terms of BLEU and METEOR.

For *S2S*, the BLEU scores of our approach are significantly better than the baseline. The *LM-conditioned* is significantly better than the baseline version in terms of BLEU-1 and METEOR. In general, the S2S-model has the highest scores across all measures. The reason may be that it was trained in a supervised manner on the given dataset, whereas the *LM-model* was only fine-tuned in an unsupervised way on a different argument corpus.

Regarding the encoding of user stances, Table 3 shows the accuracy of a linear classifier trained to predict the stance from the claims generated by each approach as well as from the ground-truth, on average and on the 10 most frequent big issues. A complete table with all big issues can be found in the appendix.

The best average accuracy across all the big issues is achieved by the *LM-model* (0.54). Compared to the corresponding baselines, the LM-model and the *S2S-model* generated claims that boosted the accuracy of the stance classifier on 33 (69%) and 21 (44%) of all big issues respectively. Overall, in 20 of the big issues, the best accuracy was achieved on the claims generated by the conditioned LM, compared to only nine big issues for the S2S-model. This indicates that the LM-conditioned can better encode a user's beliefs, modeled as stances on big issues, into generated claims.

6 Manual Evaluation

To obtain more insights into belief-based claim generation, we let users manually evaluate the output of the given approaches. Upon inspecting a sample of generated claims by our approaches, we noticed that the *LM-conditioned* produces more fluent and informative texts. Accordingly, we focused on the LM-conditioned and its baseline in the evaluation, where we conducted two user studies. The goal of the first was to assess the quality of the big-issue bag-of-words collected automatically, while the second targeted the output of the LM-model, its baseline, and a variant that utilizes a manually refined bag-of-words.

6.1 Automatic Collection of Bag-of-words

To keep the manual annotation effort manageable, we evaluated only the top-10 big issues. Two authors of this paper categorized each word in the pro/con bag-of-words of the corresponding big issue into five categories, *c1–c5*:

c1: Word irrelevant to the big issue.

c2: Relevant word, wrong stance.

c3: Relevant word, both stances possible.

c4: Relevant word, correct stance.

c5: Very relevant word, correct stance.

---

[3] step-size=0.15 and the repetition-penalty=1.2

[4] A baseline that uses the corresponding bag-of-words of the targeted topic to guide the generation wouldn't be valid, since we don't have information on the user's stance on this targeted topic.

	Overall			Relatedness Level 4			Relatedness Level 3			Relatedness Level-2		
Approach	**True**	**False**	**Undec.**	**True**	**False**	**Undec.**	**True**	**False**	**Undec.**	**True**	**False**	**Undec.**
LM-baseline	44%	34%	**22%**	50%	50%	**0%**	55%	31%	14%	**27%**	20%	**53%**
LM-conditioned	37%	32%	31%	35%	38%	27%	59%	41%	**0%**	13%	**13%**	74%
LM-cond. (manual)	**45%**	**26%**	28%	**50%**	**31%**	19%	**61%**	**28%**	11%	25%	18%	56%
Ground Truth	42%	30%	28%	38%	42%	19%	64%	27%	9%	27%	19%	54%

Table 4: Manual Evaluation: Percentage of cases for each approach where the majority of annotators predicted the stance of a generated claim on the given big issue correctly (true), incorrectly (false), or could not decide it (Undec.). The overall scores and those for each topic/big-issue relation level are listed.

	Irrelevant	Relevant			Very Relevant
Words	**c1**	**c2**	**c3**	**c4**	**c5**
Pro	14%	10%	36%	34%	6%
Con	36%	2%	34%	26%	2%

Table 5: Distribution of the pro/con bag-of-words, averaged across the top-10 big issues, over the five considered categories: c2 means wrong stance, c3 words that fit both stances, and c3 and c4 represent correct stance.

Examples can be found in the appendix. To compute inter-annotator agreement, three big issues were annotated by both annotators, resulting in Cohen's κ of 0.45, reflecting moderate agreement. Afterwards, only one annotator continued the annotations for the other big issues.

Table 5 shows the distribution of words over categories, averaged across the 10 big issues. For the pro bag-of-words, around 40% of the words are relevant and reflect the right stance, while 36% are relevant but could be used in arguments from both stances. For the con bag-of-words, however, the percentages are lower (28% and 34% respectively). A considerable proportion of words belong to categories c1 and c2, which creates noise that could confuse the conditioning process of the LM. Hence, we also consider a variant of the conditioned LM that uses only relevant words from c4 and c5.

6.2 Claim Generation

We evaluate the effectiveness in terms of whether a given generated claim reveals the stance of the given user on a specific big issue as well as how informative the claim is regarding the given topic.

Since not all topics are directly related to the big issues that can be revealed in the generated claims, we manually annotated the relatedness of the top frequent 200 topics in the test dataset to the most frequent 10 big issues, and created the evaluation sample accordingly. In particular, two authors of this paper scored the relatedness of each pair of topics and big issues on a scale from 1 to 4:

4: Topic and big issue are the same. Example: *"gay marriage should be legalized" and "gay marriage"*

3: A stance on the topic likely affects the stance on the big issue. Example: *"killing domestic abusers" and "death penalty"*

2: A stance on the topic may affect the stance on the big issue. Example: *"morality" and "abortion"*

1: Topic and big issue are not related. Example: *"do aliens exist?" and "abortion"*

The two annotators had a Cohen's κ agreement of 0.54. Around 97.4% of all pairs got score 1, 1.1% score 2, 0.8% score 3, and 0.7% score 4. The small percentage of cases that can be evaluated reflects a limitation in the designed evaluation study. However, it still allows us to evaluate the effectiveness of our approach for different levels of relatedness. Given the annotated pairs, we randomly selected 10 pairs from levels 2, 3, and 4 each. For each pair, we then collected all claims on the topic from the test set, where the author specifies a stance on the corresponding big issue. We randomly select 30 claims each, resulting in an evaluation sample of 90 instances.

We used the crowdsourcing platform *MTurk*[5] for evaluation. For each instance, we showed a topic, a claim, and the corresponding big issue to three annotators. The annotators had to perform two tasks: (1) to predict the stance of the user on the corresponding big issue from the text of the claim, and (2) to rate the claim's informativeness regarding the topic on a scale from 1 to 3.

Table 4 shows the percentage of cases in which the majority of annotators predicted the stance correctly (true), incorrectly (false), or could not decide

---

[5] A crowd sourcing platform: https://www.mturk.com/

Approach	Overall	Level 4	Level 3	Level 2
LM-baseline	1.8	**2.5**	1.9	1.4
LM-conditioned	**2.1**	2.3	**2.5**	**1.5**
LM-cond. (manual)	2.0	2.3	2.2	1.5
Ground Truth	2.0	1.9	1.8	2.2

Table 6: Manual Evaluation: Mean informativeness of the claims generated by each approach with regard to the topic (1–3, higher is better). The overall scores and those for each topic/big-issue relation level are listed.

about the stance (undec.) from the generated claim. Across the whole sample (Overall), the claims generated by *LM-conditioned (manual)*, the model conditioned on the refined bag-of-words, most often allowed to predict the stance correctly (45%). We thus attribute the low effectiveness of the *LM-model* to the noise generated by the automatic collection of big-issues' bag-of-words, especially seeing that the effectiveness gets better across all levels when eliminating this noise.

Analyzing each relatedness level individually yields more insights. For relatedness level 4, where the topic is the same as the big issue, the *LM-conditioned (manual)* generated claims where the majority of the cases with known stance were correct (63%). In level 3, we observe that both versions of our approach outperform the baseline in producing claims that express the correct stance on the corresponding big issue with percentages of 59% and 68% respectively. Finally, at relation level 2, which represents a weak relation between topics and big issues, predicting the stance seems to become hard, as indicated by high percentages of undecided cases. We believe that the weak relatedness made the annotators guess the stance in some cases, leading to unreliable annotations.

Table 6 shows the average score of all approaches regarding the informativeness of the generated claims. Here, both versions of our approach achieved better scores than the baseline, matching the ground-truth score. We believe that the low scores of the ground-truth claims stem from the noise generated in the claim detection step.

Error Analysis Table 7 shows some cases from our evaluation. In case (a), even though the model generated a claim focusing on *fundamental rights*, the majority of the annotators was undecided about the stance. Case (b) shows the effect of noise in the bag-of-words: the generated claim reflects a pro stance on abortion, while the ground-truth stance is con. This is avoided in the claim generated by *LM-conditioned (manual)*. Case (c) shows a working example of which our approach correctly generated a claim on *whaling* from an environmental perspective when conditioned as such. Case (d) is a level 2 example, indicating limitation in our evaluation, namely, the generated claim reveals a stance on abortion, but we asked about death penalty.

(a) *Topic:* is abortion ok
Big issue: (Con) Abortion *Level:* 4

Approach	Claim	Stance
LM-cond.	abortion rights groups argue that the right to abortion is a fundamental human right.	Undec.
LM-baseline	abortionists are not the only ones who are against abortion. There are many other people who are against abortion	Undec.

(b) *Topic:* abortion is right or wrong
Big issue: (Con) Abortion *Level:* 4

Approach	Claim	Stance
LM-cond.	abortion right or wrong is a personal choice. I believe that abortion should be legal because it's the woman"s body and she has every right to do what ever she wants.	Pro
LM-cond. (manual)	abortion right or wrong. I believe that abortion is murder and should be illegal in all cases except when the mother's life is at stake, rape, incest etc.	Con

(c) *Topic:* whaling
Big issue: (Pro) Environmental Protection *Level:* 3

Approach	Claim	Stance
LM-cond.	whaling is a very dangerous and cruel practice. It causes great harm to the environment, animals that are caught in it have their lives cut short by being eaten alive.	Pro
LM-baseline	whaling is a sport. It is a sport because it involves a lot of physical exertion and skill.	Undec.

(d) *Topic:* should murder be legalized?
Big issue: (Pro) Death Penalty *Level:* 2

Approach	Claim	Stance
LM-cond.	should murder be legalized? I think that abortion should not be legal. Abortion is killing a baby, and it's wrong to kill an innocent human being.	Undec.
LM-baseline	should murder be legalized? I think so. I think that it should be legalized because it is a good thing	Undec.

Table 7: A selection of claims generated by the different evaluated approaches for the different association levels between topic and big issue discussed in the text.

7 Conclusion

In this paper, we have proposed to equip argument generation technology with the ability to encode beliefs for two reasons: first, it reflects the human process of synthesizing arguments, and second, it gives more control on the generated arguments leading to a better reach of the audience. For this purpose, we have presented the task of belief-based claim generation. Concretely, we studied the research questions of how to model a user's beliefs as well as how to encode them when generating an argumentative text. We have modeled users' beliefs via their stances on big issues, and used them as an extra input in our approaches.

Our automatic evaluation has provided evidence of the applicability of encoding beliefs into argumentative texts. In manual studies, we found that limitations in the effectiveness of our approach stem from noise produced by the automatic collection of a bag-of-words. The findings of this paper lay the ground to investigate the role of beliefs in generating arguments that reach their audience.

We point out that ethical issues arise, when tuning arguments to affect specific people, such as attempts to manipulate them. While the task and settings considered here are rather too fundamental to already make these issues critical, future work should pay attention to them. Our goal is to develop systems that bring people together.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback. This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (SFB 901/3) under the project number 160364472.

References

Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Data acquisition for argument search: The args.me corpus. In *Proceedings of the 42nd Edition of the German Conference on Artificial Intelligence*, page 48?59.

Milad Alshomary, Shahbaz Syed, Martin Potthast, and Henning Wachsmuth. 2020. Target inference in argument conclusion generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4345, Online. Association for Computational Linguistics.

Yonatan Bilu, Daniel Hershcovich, and Noam Slonim. 2015. Automatic claim negation: Why, how and when. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 84–93, Denver, CO. Association for Computational Linguistics.

Giuseppe Carenini and Johanna D Moore. 2006. Generating and evaluating evaluative arguments. *Artificial Intelligence*, 170(11):925–952.

Tuhin Chakrabarty, Christopher Hidey, and Kathleen McKeown. 2019. Imho fine-tuning improves claim detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 558–563.

Wei-Fan Chen, Henning Wachsmuth, Khalid Al Khatib, and Benno Stein. 2018. Learning to flip the bias of news headlines. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 79–88. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Esin Durmus and Claire Cardie. 2018. Exploring the role of prior beliefs for argument persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1035–1045, New Orleans, Louisiana. Association for Computational Linguistics.

Frans H van Eemeren and Peter Houtlosser. 1999. Strategic manoeuvring in argumentative discourse. *Discourse studies*, 1(4):479–497.

Roxanne El Baff, Henning Wachsmuth, Khalid Al Khatib, Manfred Stede, and Benno Stein. 2019. Computational argumentation synthesis as a language modeling task. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 54–64, Tokyo, Japan. Association for Computational Linguistics.

Matthew Feinberg and Robb Willer. 2015. From gulf to bridge: When do moral arguments facilitate political influence? *Personality and Social Psychology Bulletin*, 41(12):1665–1681.

David M Godden. 2010. The importance of belief in argumentation: Belief, commitment and the effective resolution of a difference of opinion. *Synthese*, 172(3):397–414.

Floriana Grasso, Alison Cawsey, and Ray Jones. 2000. Dialectical argumentation to solve conflicts in advice giving: a case study in the promotion of healthy nutrition. *International Journal of Human-Computer Studies*, 53(6):1077–1115.

Christopher Hidey and Kathleen McKeown. 2019. Fixed that for you: Generating contrastive claims with semantic edits. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1756–1767.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019. Argument generation with retrieval, planning, and realization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2661–2672.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*.

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

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Misa Sato, Kohsuke Yanai, Toshinori Miyoshi, Toshihiko Yanase, Makoto Iwayama, Qinghua Sun, and Yoshiki Niwa. 2015. End-to-end argument generation system in debating. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 109–114.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. 2020. Aspect-controlled neural argument generation. *arXiv preprint arXiv:2005.00084*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

M. Stede, J. Schneider, and G. Hirst. 2018. *Argumentation Mining*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Henning Wachsmuth, Manfred Stede, Roxanne El Baff, Khalid Al Khatib, Maria Skeppstedt, and Benno Stein. 2018. Argumentation synthesis following rhetorical strategies. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3753–3765. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Ingrid Zukerman, Richard McConachy, and Sarah George. 2000. Using argumentation strategies in automated argument generation. In *INLG'2000 Proceedings of the First International Conference on Natural Language Generation*, pages 55–62, Mitzpe Ramon, Israel. Association for Computational Linguistics.

Non-Autoregressive Text Generation with Pre-trained Language Models

Yixuan Su$^\diamondsuit$ **Deng Cai**$^\heartsuit$ **Yan Wang**$^\spadesuit$ **David Vandyke**$^\clubsuit$
Simon Baker$^\diamondsuit$ **Piji Li**$^\spadesuit$ **Nigel Collier**$^\diamondsuit$
$^\diamondsuit$Language Technology Lab, University of Cambridge
$^\heartsuit$The Chinese University of Hong Kong
$^\spadesuit$Tencent AI Lab
$^\clubsuit$Apple
`{ys484,sb895,nhc30}@cam.ac.uk`
`thisisjcykcd@gmail.com, dvandyke@apple.com`
`{brandenwang,pijili}@tencent.com`

Abstract

Non-autoregressive generation (NAG) has recently attracted great attention due to its fast inference speed. However, the generation quality of existing NAG models still lags behind their autoregressive counterparts. In this work, we show that BERT can be employed as the backbone of a NAG model to greatly improve performance. Additionally, we devise mechanisms to alleviate the two common problems of vanilla NAG models: the inflexibility of prefixed output length and the conditional independence of individual token predictions. Lastly, to further increase the speed advantage of the proposed model, we propose a new decoding strategy, *ratio-first*, for applications where the output lengths can be approximately estimated beforehand. For a comprehensive evaluation, we test the proposed model on three text generation tasks, including text summarization, sentence compression and machine translation. Experimental results show that our model significantly outperforms existing non-autoregressive baselines and achieves competitive performance with many strong autoregressive models. In addition, we also conduct extensive analysis experiments to reveal the effect of each proposed component.[1]

1 Introduction

Autoregressive generation (AG) models achieve state-of-the-art performance on a wide range of text generation tasks, such as machine translation (Vaswani et al., 2017) and text summarization (Rush et al., 2015). Such models generate a token sequence in a left-to-right, token-by-token fashion. The prediction for the next token is conditioned on all previously generated tokens. This characteristic makes it impossible to parallelize the computational overhead for token predictions in different positions, which leads to a relatively high latency in inference. On the other hand, non-autoregressive generation (NAG) models (Gu et al., 2018) have emerged as a promising alternative due to their fast inference speed. NAG models omit the sequential dependencies within the output-side sequence and predict tokens in all positions simultaneously once the output length has been determined beforehand. While NAG models enjoy full parallelism and faster inference, the generation quality of NAG models often lags behind their autoregressive counterparts.

In this work, we explore the potential of large-scale pre-trained language models for improving the performance of non-autoregressive generation. Specifically, we utilize BERT (Devlin et al., 2019) as the backbone for NAG modelling and extend the architecture of BERT with a CRF output layer (Lafferty et al., 2001; Sun et al., 2019) for better capturing the output-side dependencies.

In addition, we analyze two significant limitations that NAG models currently suffer from: (1) the inflexibility of prefixed output length, and (2) the conditional independence of individual token predictions. Accordingly, we devise two solutions to these two problems.

First, prior NAG models require the output length to be determined before token generation, thus an extra module for output length prediction is always required. Nevertheless, the most likely length from the prediction module is not necessarily the best-suited one for the token generation model. To this end, previous works (Gu et al., 2018; Ma et al., 2019) usually rely on length-parallel decoding (LPD) (Wei et al., 2019) for performance enhancement; that is, generating and re-ranking the results from different output length candidates. In this work, we propose a simple and elegant decoding mechanism that lets the model determine the output length on-the-fly. Specifically, our model dynamically adjusts the output sequence length via

---

[1] All related code, data, and models can be found in https://github.com/yxuansu/NAG-BERT.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 234–243
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

emitting an `[eos]` token at any output position to indicate the ending of the generated sequence. Therefore, we can avoid the additional efforts of output length prediction and results re-ranking.

Second, most existing NAG models assume the token predictions in different positions are conditionally independent. As a consequence, they often tend to generate results that are ungrammatical with repetitions (Wang et al., 2019b). To alleviate this problem, we propose a context-aware learning objective which impels the model to output different tokens at adjacent positions, thereby reducing the possibility of repetitive generation.

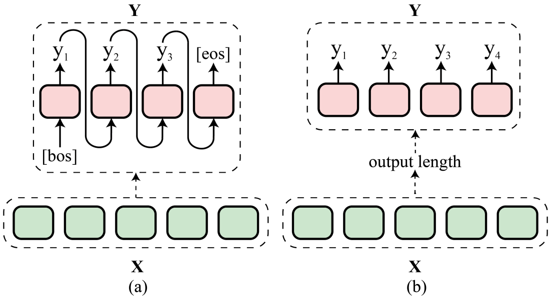
Furthermore, for tasks like text summarization, the output sequence (summary) is known to be shorter than the source sequence (article). In such cases, to further improve the model's inference efficiency, we introduce a new *ratio-first* decoding strategy. Specifically, instead of performing inference on all source-side hidden states, ratio-first generates the result only based on a subset of source hidden states. The subset size is jointly determined by the source length T and a predefined ratio α that is set based on our prior knowledge from the data statistics. In the experiments, we show that ratio-first can significantly improve the inference speed while maintaining the generation quality.

We evaluate the proposed model on three typical text generation tasks, including text summarization, sentence compression and machine translation. Experimental results show that our model significantly outperforms many strong non-autoregressive baselines, and even performs competitively with several strong autoregressive models. In addition, we conduct extensive analysis experiments to study the effect of individual proposed components.

In summary, our contributions are: (1) We propose a novel framework that utilizes BERT for text generation under the non-autoregressive generation paradigm; (2) We propose a decoding mechanism that allows the model to dynamically determine the output length, and a new context-aware learning objective that reduces errors stemming from the output-side conditional independence assumption; (3) We introduce a ratio-first decoding strategy that further improve the model's inference efficiency.

2 Background

Autoregressive generation (AG) models generate sequences based on a left-to-right factorization. As shown in Figure 1, given the source sequence $\mathbf{X}$, the target sequence $\mathbf{Y}$ with length T' is generated via a chain of conditional probabilities based on the left-to-right sequential dependencies as:



(a)

(b)

Figure 1: (a) Autoregressive; (b) Non-Autoregressive

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{T'} p(y_i|y_{<i}, \mathbf{X}), \qquad (1)$$

where $y_{<i}$ denotes the tokens before the i-th step. This property of autoregressive factorization makes the generation process hard to be parallelized as the result is generated token by token.

Unlike AG models, non-autoregressive (NAG) models generate sequences without modelling the output-side dependencies. As shown in Figure 1, given the prespecified output length T', the probability of the target sequence $\mathbf{Y}$ is then modelled as:

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{T'} p(y_i|\mathbf{X}, i, T'). \qquad (2)$$

With this conditional independence assumption, NAG models can fully parallelize their generation process, which significantly improves the inference speed. However, it has been shown that, the choice of the prespecified output length has a notable impact on the model's generation quality (Gu et al., 2018). In addition, the removal of output-side sequential dependency also causes the generation quality of NAG models to be inferior to their autoregressive counterparts (Wang et al., 2019b).

3 Proposed Model

In this section, we give a detailed explanation of the proposed model. First, we describe how to utilize BERT as a non-autoregressive generation model. Then we discuss the decoding mechanism which allows the model to determine the output length dynamically. Finally, we introduce the new ratio-first decoding strategy which further improves the model's decoding efficiency.

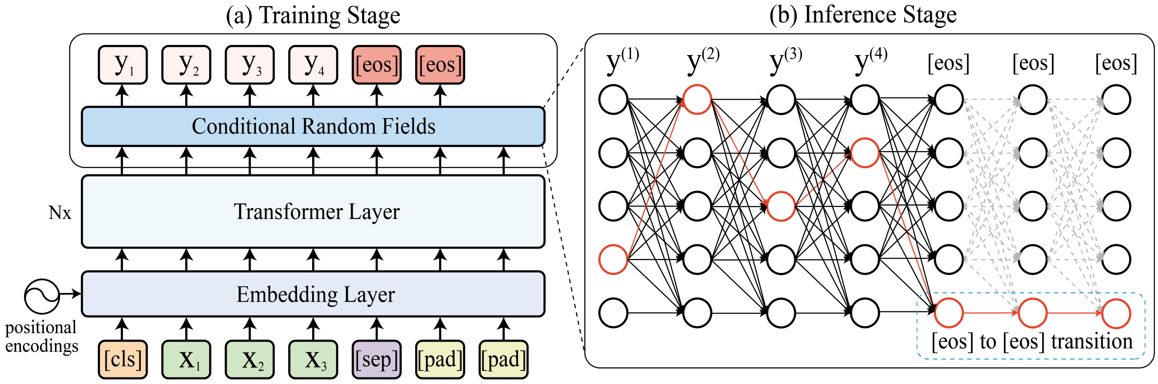


Figure 2: The overall illustration of the proposed model: During training, the model parameters are only updated on the positions of the target sequence. During inference, once the decoded trajectory (colored in red) gets into the `[eos]` state, it will only transit to the `[eos]` state in the remaining steps. The final result is obtained by removing the generated `[eos]` tokens from the entire decoded trajectory.

3.1 Model Architecture

The architecture of the proposed model is presented in Figure 2, in which the embedding layer and the stack of transformer layers are initialized with BERT (Devlin et al., 2019).

Input Representation Following the setup of BERT, we first append a `[cls]` and a `[sep]` token on both sides of the source sequence. Then we attach a number of `[pad]` tokens at the end of source sequence to make its length equal to the predefined maximum size (e.g., 256). Thus we can make sure the source length is longer than or equal to the output length. As a special case, for tasks like text summarization where the source is known to be longer than the target, we do not attach the `[pad]` tokens when constructing the input.

Transformer Layers Given the source sequence $\mathbf{X}$, it is processed by a stack of N transformer (Vaswani et al., 2017) layers. Formally, the Multi-Head Attention is defined as $\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$, where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ denotes the query, key and value respectively. The computation of the first transformer layer is then defined as:

$$\mathbf{V}^{(1)} = \text{MultiHead}(E(\mathbf{X}), E(\mathbf{X}), E(\mathbf{X})), \quad (3)$$

$$\mathbf{O}^{(1)} = \text{FFN}(\mathbf{V}^{(1)}), \quad (4)$$

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2, \quad (5)$$

where $E(\mathbf{X}) = TE(\mathbf{X}) + PE(\mathbf{X})$ in which $TE(\cdot)$ denotes the token embedding and $PE(\cdot)$ denotes the position embedding. For other layers:

$$\mathbf{V}^{(n)} = \text{MultiHead}(\mathbf{O}^{(n-1)}, \mathbf{O}^{(n-1)}, \mathbf{O}^{(n-1)}), \quad (6)$$

$$\mathbf{O}^{(n)} = \text{FFN}(\mathbf{V}^{(n)}), \quad (7)$$

where $n = 2, ..., N$ and N is the total number of transformer layers. The final sequence representation $\mathbf{H} \in \mathbb{R}^{T \times d_{\text{model}}}$ is the output states of BERT from the last layer, where T is the source sequence length and d_{model} is the model size.

CRF Layer Then, $\mathbf{H}$ is passed through a linear-chain CRF (Lafferty et al., 2001). Under the CRF framework, the likelihood of the target sequence $\mathbf{Y}$ with length T' is then modelled as:

$$P_{\text{CRF}}(\mathbf{Y}|\mathbf{X}) = \frac{e^{S(\mathbf{X},\mathbf{Y})}}{\sum_{\mathbf{Y}'} e^{S(\mathbf{X},\mathbf{Y}')}} = \frac{1}{Z(\mathbf{X})} \exp\Big(\sum_{i=1}^{T'} \Phi_{y_i}(h_i) + \sum_{i=2}^{T'} t(y_{i-1}, y_i)\Big), \quad (8)$$

where $Z(\mathbf{X})$ is the normalizing factor and $\Phi_{y_i}(h_i)$ denotes the label score of y_i at position i. In practice, Φ is parameterized by a neural network that maps the BERT output state h_i into the label (vocabulary) space. The $t(y_{i-1}, y_i) = \mathbf{T}_{y_{i-1}, y_i}$ denotes the transition score from label y_{i-1} to y_i where $\mathbf{T} \in \mathbb{R}^{|V| \times |V|}$ is the transition matrix.

Approximation In the context of text generation, the size of the label space (vocabulary size) $|V|$ is typically large, e.g., $32k$. Therefore, it is intractable to directly model the transition matrix $\mathbf{T}$ and the normalizing factor $Z(\mathbf{X})$. To this end, we adopt the techniques proposed by Sun et al. (2019) to approximate these two terms. Specifically, the full transition matrix is approximated by the product of two low-rank matrices $\mathbf{T} = \mathbf{E}_1 \mathbf{E}_2^T$, where $\mathbf{E}_1, \mathbf{E}_2 \in \mathbb{R}^{|V| \times d}$ and d is much smaller than $|V|$. To compute the normalizing factor $Z(\mathbf{X})$, at each time step, instead of searching through all possible paths, the number of candidates is heuristically truncated to a predefined beam size k. We refer readers to the original paper for further details.

3.2 Output Length Determination

In this section, we describe how to let the model determine the output sequence length by itself. Our basic idea is that we want the model to dynamically stop generation via emitting a special `[eos]` token. To achieve this, during training, we manually append **two** consecutive `[eos]` tokens to the end of the target sequence, as shown in the top left part of Figure 2. In this way, the model can learn a deterministic transition behaviour between two `[eos]` states, meaning that $t(\texttt{[eos]}, \texttt{[eos]}) = \max_{v \in \mathcal{V}} t(\texttt{[eos]}, v)$. This is because, during training, the model never sees a transition $(\texttt{[eos]}, v)$, where $v \neq \texttt{[eos]}$.

During inference, the result $\tilde{\mathbf{Y}}$ is acquired as $\tilde{\mathbf{Y}} = \arg\max_{\mathbf{Y}'} S(\mathbf{X}, \mathbf{Y}')$, where the CRF scoring function $S(\mathbf{X}, \mathbf{Y}')$ in Equation (8) can be decomposed as:

$$
\begin{aligned}
S(\mathbf{X}, \mathbf{Y}') &= \sum_{i=1}^{T} \Phi_{y_i'}(h_i) + \sum_{i=2}^{T} t(y_{i-1}', y_i') \\
&= \underbrace{\Phi_{y_1'}(h_1)}_{\text{initial state}} + \sum_{i=2}^{T} \underbrace{\{\overbrace{\Phi_{y_i'}(h_i)}^{\text{label score}} + \overbrace{t(y_{i-1}', y_i')}^{\text{transition score}}\}}_{\text{state transition}}.
\end{aligned} \tag{9}
$$

Once the decoded trajectory enters the `[eos]` state, the state transition term in $S(\mathbf{X}, \mathbf{Y}')$ will be dominated by the transition score term $t(\texttt{[eos]}, \texttt{[eos]})$. As a result, the model will keep transitioning to `[eos]` in the remaining steps. An example is provided in the right part of Figure 2, from which we can see that, at step 5, the decoded trajectory enters the `[eos]` state and remains at it in the rest of the generation process. In this way, our model can dynamically control the length of output sequence by entering the `[eos]` state during the generation process. After the entire generation process is completed, the final output sequence can be obtained by removing all generated `[eos]` tokens.

3.3 Ratio-First Decoding

We note that the outputs of BERT can be divided into two subsets. The first subset ranges from the beginning to the position where the first `[eos]` is emitted, and the second subset is the rest. For example, in Figure 2, the first subset are those corresponding to the output sequence "$y^{(1)}$ $y^{(2)}$ $y^{(3)}$ $y^{(4)}$ `[eos]`". As for the second part, we can see that it has little effect on the final output and removing it should not change the result. This indicates that it suffices to only consider the beginning part of BERT outputs for improving the inference speed. Especially, for tasks like summarization where the target is known to be shorter than the source sequence, we are safe to only use the first $[\alpha \cdot T]$ outputs of BERT to perform inference. Here T denotes the source length, $\alpha \in (0.0, 1.0)$ is set based on the data statistics and $[\cdot]$ is the integer rounding operation. Formally, given the source sequence $\mathbf{X}$, the ratio-first decoding is defined as

$$
\begin{aligned}
\tilde{\mathbf{Y}} &= \arg\max_{\mathbf{Y}'} \mathcal{F}(\mathbf{X}, \mathbf{Y}', \alpha), \\
&= \arg\max_{\mathbf{Y}'} \{ \sum_{i=1}^{[\alpha \cdot T]} \Phi_{y_i'}(h_i) + \sum_{i=2}^{[\alpha \cdot T]} t(y_{i-1}', y_i') \}.
\end{aligned} \tag{10}
$$

When $\alpha = 1.0$, ratio-first degenerates to the standard decoding strategy in CRF-based models.

It should be noted that, $[\alpha \cdot T]$ only constrains the maximum length of the generated result, and the actual output length (after removing the generated `[eos]` tokens) is still decided by the model itself. In the experiment section, we demonstrate that ratio-first can notably improve the inference speed whilst maintaining the generation quality.

4 Learning

Due to the conditional independence approximation on output tokens, NAG models often tend to generate repeated tokens (Wang et al., 2019b). One way to alleviate this problem is to introduce implicit dependencies on the output side. In this work, we propose to use the unlikelihood formulation of Welleck et al. (2020) in the context of NAG, where we define the set of negative candidate as the surrounding tokens within a predefined context window c. Formally, given the source sequence $\mathbf{X}$ and the target sequence $\mathbf{Y}$ with length T', the proposed context-aware objective is defined as:

$$
\begin{aligned}
\mathcal{L}_{\text{CA}}(\mathbf{Y}|\mathbf{X}) &= -\sum_{i=1}^{T'} \{\log p_\theta(y_i|h_i; \mathbf{X}) + l_{\text{CA}}(i)\}, \\
l_{\text{CA}}(i) &= \sum_{j=i-c, y_j \neq y_i}^{j=i+c} \log(1.0 - p_\theta(y_j|h_i; \mathbf{X})),
\end{aligned} \tag{11}
$$

where h_i is the model output state at position i. At position i, the proposed objective maximizes the probability of token y_i while minimizing the probabilities of the surrounding tokens. In this way, it discourages the model from generating repetitive tokens at different time steps.

The overall learning objective is then defined as

$$\begin{aligned} \mathcal{L}_{\text{CRF}} &= -\log P_{\text{CRF}}(\mathbf{Y}|\mathbf{X}), \\ \mathcal{L} &= \mathcal{L}_{\text{CRF}} + \lambda \cdot \mathcal{L}_{\text{CA}}, \end{aligned} \tag{12}$$

where λ controls the importance of different loss terms and $P_{\text{CRF}}(\mathbf{Y}|\mathbf{X})$ is described in Equation (8).

5 Related Work

Non-Autoregressive generation was first introduced by Gu et al. (2018) to reduce the inference latency in machine translation. Recent works in this area have investigated ways to mitigate the trade-off between the decoding speed and generation quality. Gu et al. (2018) utilized fertility as latent variables for better translation performance. Wang et al. (2019b) proposed two auxiliary objectives for better modelling the output states and solving the under-translation problem. To better model the intermediate alignments between source and target sides, Ma et al. (2019) proposed a model based on the generative flow framework. Ghazvininejad et al. (2019) proposed to use a masked language objective to train the NAG model. During inference, starting from a fully masked sequence, the output is generated in an iterative refinement manner. Recently, Sun et al. (2019) proposed to incorporate a conditional random field into the decoder of a NAG model for better modelling the output-side dependencies. Our work is different from prior works in two aspects: (1) we directly utilize a pre-trained language model (BERT) to perform non-autoregressive generation; (2) our model can dynamically generate the output sequence without the need of prespecified output length.

6 Experiments

We evaluate the proposed model on three typical text generation tasks: (1) text summarization; (2) sentence compression and (3) machine translation.

6.1 Experimental Setup

We implement the proposed model with PyTorch (Paszke et al., 2017). The BERT model we use is the Huggingface implementation (Wolf et al., 2019) (bert-base-uncased). To approximate the transition matrix in the CRF layer, we set the dimension d of matrices $\mathbf{E}_1$ and $\mathbf{E}_2$ as 32. For the normalizing factor $\mathbf{Z}(\mathbf{X})$, we set the predefined beam size k as 256. As for the overall learning objective, we set the window size c as 3 and λ as 1.0. In training, we use Adam optimizer (Kingma and Ba, 2015). To measure the relative speedup, we follow the standard setup which runs inference for each individual example separately. The model's inference speed is computed by averaging the results of test cases. For a fair comparison, we measure the inference speed of all models on the same platform.

6.2 Text Summarization

Text summarization aims to automatically generate a compact summary that retains the most important content of the original text document (Nenkova and McKeown, 2012). In this experiment, we use the Gigawords dataset (Rush et al., 2015) as our benchmark. For evaluation, standard metrics including ROUGE-1 (R-1), ROUGE-2 (R-2) and ROUGE-L (R-L) (Lin, 2004) are reported.

We compare our model with several representative and the latest NAG models, including NAG-NMT (Gu et al., 2018), NAR-REG (Wang et al., 2019b) and NAG-CRF (Sun et al., 2019). Following previous works, during training, we train a length predictor to predict the output length. During inference, for each NAG baseline, we adopt the length-parallel decoding strategy (LPD-k) (Wei et al., 2019), that is, generating k results using the top-k possible output length predictions from the length predictor. The results are then re-ranked by a transformer model to get the final ouput. In the experiment, we report the results of different NAG baselines using LPD-9 decoding. In addition, to better examine the effect of using BERT in NAG models, we add a **B**NAG-CRF baseline which adopts the same structure of the NAG-CRF model but using BERT as the encoder. We also compare our model with several strong autoregressive models, which are Luong-NMT (Luong et al., 2015), Pointer-Generator (See et al., 2017), DRGD (Li et al., 2017) and Concept Pointer (Wang et al., 2019a). To measure the relative inference speedup, we include transformer as a baseline model.

The results are shown in Table 1, from which we can see that, by using length-parallel decoding, the performance of all NAG baselines can be notably improved. However, such procedure significantly increases the inference latency. In contrast,

Models	R-1	R-2	R-L	Speedup
Autoregressive				
Luong-NMT	33.10	14.45	30.71	-
Pointer-Generator	35.98	15.99	33.33	-
DRGD	36.25	17.61	33.55	-
Concept Pointer	36.62	16.40	33.98	-
Transformer (b = 4)	35.74	16.97	33.43	1.00×
Non-Autoregressive				
NAG-NMT	27.20	8.96	25.58	**9.31**×
+LPD-9	29.76	10.03	28.04	5.28×
NAR-REG	28.56	9.79	26.83	8.64×
+LPD-9	31.23	11.14	29.55	4.74×
NAG-CRF	30.29	12.61	28.71	8.07×
+LPD-9	32.91	14.31	31.03	4.32×
BNAG-CRF	32.63	14.32	30.82	6.13×
+LPD-9	34.56	16.10	32.76	3.21×
Ours ($\alpha = 0.3$)	34.67	16.13	32.81	**9.31**×
Ours ($\alpha = 1.0$)	**35.05**	**16.48**	**33.28**	6.72×

Table 1: Results on Gigawords dataset, where b in the transformer baseline stands for beam search size.

our model can self-determine the output length without any re-ranking process. As shown in the results, our model outperforms the best NAG baseline (with LPD) and achieves performances that are comparable with several strong AG models.

Comparing the results of **B**NAG-CRF and NAG-CRF, we can see that incorporating BERT as encoder helps to improve the model performance. Nonetheless, our model still outperforms **B**NAG-CRF with LPD-9 decoding. This is because the dynamic length decoding mechanism allows our model to generate results with optimal length, leading to stronger model performances.

Finally, we analyze the proposed ratio-first decoding. From the results, we observe a moderate performance drop when using ratio-first ($\alpha = 0.3$). It comes from the fact that, for some input documents with length T, the reference summary is longer than $[\alpha \cdot T]$. In such cases, ratio-first fails to generate the complete reference summary, leading to the drop of performance. On the other hand, we can see that, ratio-first can notably improve the inference speedup. With $\alpha = 0.3$, our model achieves the highest inference speedup while still outperforms all compared NAG models.

6.3 Sentence Compression

Sentence compression aims at compressing a long sentence into a short one by deleting redundant words. In this experiment, we use the Google sentence compression dataset (Filippova and Altun, 2013) as our benchmark. For evaluation, we use the standard token-kept-F1 (F1) score. In addition, We also report the results of other standard metrics including ROUGE-1, ROUGE-2 and ROUGE-L.

Models	F1	R-1	R-2	R-L	Speedup
Autoregressive					
Bi-LSTM-Dep	82.3	81.5	74.1	81.3	-
Tagger	82.8	81.1	72.4	80.9	-
Tagger+ILP	79.0	76.1	64.6	75.8	-
HiSAN-Dep	82.7	82.1	74.9	81.9	-
HiSAN	83.2	82.9	75.8	82.7	-
Transformer (b = 4)	82.4	82.0	74.6	81.8	1.00×
Non-Autoregressive					
NAG-NMT	72.5	72.1	59.9	71.8	**10.71**×
+LPD-9	73.8	73.6	61.0	73.1	6.09×
NAG-REG	73.7	73.1	61.5	73.0	10.00×
+LPD-9	75.6	75.1	63.4	74.9	5.49×
NAG-CRF	75.1	74.4	66.8	74.2	9.41×
+LPD-9	77.3	76.5	69.0	76.3	5.04×
BNAG-CRF	77.1	76.2	68.9	76.0	7.21×
+LPD-9	79.3	78.5	71.7	78.2	3.91×
Ours ($\alpha = 0.7$)	79.5	79.0	72.1	78.7	10.00×
Ours ($\alpha = 1.0$)	**80.7**	**80.3**	**73.6**	**80.1**	8.42×

Table 2: Results on sentence compression task

We compare the proposed model with the same NAG baselines as in the previous experiment. We also compare our model with several strong autoregressive models, including Bi-LSTM-Dep (Filippova et al., 2015), Tagger and Tagger+ILP (Wang et al., 2017), HiSAN-Dep and HiSAN (Kamigaito et al., 2018). To measure the inference speedup, we include transformer as a baseline model.

The results are presented in Table 2, from which we see that our model outperforms the best reported NAG baseline (with LPD) in terms of both the generation quality and inference speed. Comparing with the strong autoregressive models, our model can achieve competitive performance with a over $8.42\times$ inference speed up. We also report the results of our model using the ratio-first decoding strategy. By setting α as 0.7, it achieves a $10.00\times$ inference speedup while still outperforming other compared NAG baselines.

6.4 Machine Translation

Machine translation aims at translating text from the source language to the target language. In this task, we use the IWSLT14 German-to-English (DE-EN) dataset as our benchmark. Following previous works, we use the sequence-level knowledge distillation (Gu et al., 2018) during training. For evaluation, we report results in BLEU scores (Papineni et al., 2002). In this experiment, we use the BERT model in German language.

We compare our model with a range of strong

Models	BLEU	Speedup(×)
Autoregressive		
LSTM-based	28.53	-
CNN-based	32.84	-
Transformer (b = 4)	33.31	1.00
Non-Autoregressive		
ENAG-E	24.13 (27.30)	15.08 (7.39)
ENAG-P	25.09 (28.60)	14.48 (7.24)
NAG-REG	23.89 (28.04)	16.45 (9.05)
NAG-NMT	23.04 (26.79)	13.92 (7.24)
NAG-CRF	26.39 (29.21)	11.74 (6.03)
BNAG-CRF	26.73 (29.67)	9.42 (5.01)
Ours ($\alpha = 0.8$)	29.71	13.92
Ours ($\alpha = 1.0$)	**30.45**	11.31

Table 3: Results on IWSLT14 De-En dataset. The numbers in () are results using length-parallel decoding.

BERT	CRF	R-1	R-2	R-L
✓	✓	35.05	16.48	33.28
×	✓	32.41	14.19	30.53
✓	×	32.16	11.33	30.34
×	×	27.02	8.81	25.25

Table 4: Ablation study on Gigawords dataset.

NAG models, including NAG-NMT (Gu et al., 2018), ENAG-E and ENAG-P (Guo et al., 2019), NAG-REG (Wang et al., 2019b), NAG-CRF (Sun et al., 2019) and **B**NAG-CRF. For each NAG baseline, we also report the results using LPD-9 decoding. In addition, we compare our model with several strong autoregressive models, including LSTM-based (Wu et al., 2016), CNN-based (Gehring et al., 2017) and transformer model.

The results are shown in Table 3, from which we see that our model outperforms the best NAG baseline (with LPD) in terms of both the generation quality and inference speedup. Additionally, we also report the results using the ratio-first decoding. By setting α as 0.8, the inference speedup can be further boosted to 13.92× while the generation quality is still higher than the best NAG baseline.

6.5 Further Analysis

In this section, we present further discussions and empirical analysis of the proposed model.

BERT & CRF To quantify the importance of each component (BERT & CRF) of our model, we evaluate the performance on Gigawords dataset by removing each component iteratively.

The results are shown in Table 4, from which we can see that by removing any of these components, the overall performance decreases. By removing BERT from the model, we observe notable drop across all metrics. This shows that the knowledge of BERT is an important factor of the model's strong performance. Comparing with results in Table 1, it still outperforms vanilla NAG-CRF and performs comparably with NAG-CRF using LPD decoding, which demonstrates the merit of the proposed dynamic length decoding mechanism. Another interesting finding is that, by only removing the CRF layer, the most notable drop is observed on the bigram-level metric (ROUGE-2). This shows that the bigram-level dependencies on the output side are mainly captured by the CRF module. In addition, by removing both BERT and CRF, all metrics further decrease. This confirms that each of these two components positively contributes to the model's overall performance.

Models	rep-1	rep-2	rep-3	rep-4	R-L
w/o CA	6.897	2.640	0.741	0.295	32.89
Ours	5.786	1.978	0.427	0.106	33.28
Transformer	4.329	1.348	0.267	0.089	33.43

Table 5: Evaluation results on n-gram repetitions.

Context-Aware Objective In this part, we study the effect of the context-aware objective. As described in Equation (11), it aims at alleviating the problem of repetitive generation. To give a quantitative analysis, we use the measurement of sentence-level repetition (Welleck et al., 2020) to compute the ratio of duplicate n-grams (rep-n) in the generated result. This metric is defined as

$$\text{rep-}n(\mathbf{Y}) = 100 \times (1.0 - \frac{|\text{unique } n\text{-grams}(\mathbf{Y})|}{|n\text{-grams}(\mathbf{Y})|}). \tag{13}$$

For each generated result, rep-n is 0.0 when it has no repeating n-grams. The final result is computed by averaging over the entire evaluation set.

We conduct experiments on Gigawords dataset to evaluate the n-gram repetitions ranging from uni-gram to 4-gram. The results are shown in Table 5, where w/o CA means the model is trained without using context-aware objective and R-L denotes the model's ROUGE-L score. Additionally, we also show the results from transformer model for a direct comparison. Comparing the two variants of our model, we see that training with context-aware objective leads to a 42% drop on rep-3 metric (0.427 vs 0.741) and a 64% drop on rep-4 metric (0.106 vs 0.295). The ROUGE-L results also indicate that

Models	Ours ($\alpha = 1.0$)	Length-Parallel Decoding		
		LPD-1	LPD-5	LPD-10
BLEU	**30.45**	27.15	29.62	30.37
Speedup(×)	11.31	11.84	8.92	6.01

Table 6: Results comparison on IWSLT14 dataset

the reduction in token repetition can effectively improve the model generation quality.

Dynamic Length Determination Next, we examine the importance of the model's ability to dynamically determine the length of the generated output. To this end, we train another model variant by removing the two `[eos]` tokens from the target sequence. In this way, the model is not able to self-determine the output length throughout the generation process. To perform inference, we use length-parallel decoding (LPD) with different number of length candidates. Formally, for each length candidate l, the model generates the result $\tilde{\mathbf{Y}}$ as

$$\tilde{\mathbf{Y}} = \arg\max_{\mathbf{Y}'}\{\sum_{i=1}^{l}\Phi_{y'_i}(h_i) + \sum_{i=2}^{l} t(y'_{i-1}, y'_i)\}. \tag{14}$$

The final result is acquired by re-ranking the generated results with a transformer model.

We conduct experiments on the IWSLT14 DE-EN dataset in which we try a different number of length candidates, including top-1, top-5 and top-10. The results are shown in Table 6, from which we can see, as the number of length candidates increases, the model performance increases as well. The reason is that a larger candidates set is more likely to contain the best-suited length for the generation model, leading to better performance. However, such decoding procedure inevitably increases the required computation overhead. We can see that, when setting k as 10, the inference speedup decreases from 11.84× to 6.01×. In contrast, our proposed model is able to determine the optimal output length by itself. Without any re-ranking process, it outperforms the model with LPD-10 decoding and achieves the inference speedup that is comparable with the model using LPD-1 decoding.

Ratio-First Decoding We are also interested in the effect of the ratio-first decoding strategy. To provide a quantitative analysis, we perform inference on the Gigawords dataset using ratio-first with different α. The experimental results with different α are presented in Figure 3. It can be observed that, when α reaches 0.3, the model approximately achieves its optimal performance. At the same time, a notable improvement can be observed in terms of the inference speedup ($6.72\times \rightarrow 9.31\times$).

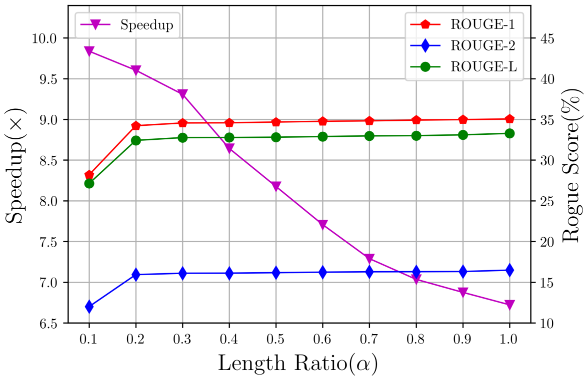


Figure 3: Experiment results on Gigawords dataset using ratio-first decoding with different α.

Figure 4: The distribution of target/source length ratio of the training and test set in Gigawords dataset.

Now we illustrate why the near optimal performance can be achieved when α reaches 0.3. In Figure 4, we present the distribution of the target/source length ratio of every data instance in the Gigawords dataset. We can see that, for most cases, the ratio between the target length T' and source length T is less than 0.3. Recall the definition of ratio-first decoding in Equation (10), the $[\alpha \cdot T]$ constrains the maximum length of the generated result. Therefore, once we have a prior knowledge on the data statistic, we can easily choose a proper α that both improves the inference speed whilst maintaining the generation quality. In this case, a proper α could be 0.3 which is demonstrated by the results in Figure 3 and 4. By setting different α, ratio-first provides us an explicit way to control the balance between the inference speed and the generation quality. This property of ratio-first is especially favorable in real-life scenarios where the inference speed is the highest concern.

7 Conclusion

In this work, we explored the potential of BERT in various text generation tasks under the NAG framework. To address problems from NAG models previously having a prefixed output length, we devised a decoding mechanism which enables the model to determine the output length dynamically. To reduce errors stemming from the assumption of conditional independence of output tokens, we proposed a context-aware objective as well as using a CRF decoding. Furthermore, to maximize the inference speed advantage of our model, we introduced a ratio-first decoding strategy. We evaluated our model on three benchmark datasets and the results show that our model significantly outperforms many strong NAG baselines and performs comparably to many strong AG models.

Acknowledgments

The authors wish to thank Jialu Xu, Guanlin Li, Xing Wang for their insightful discussions and support. Many thanks to our anonymous reviewers for their suggestions and comments.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 360–368.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1481–1491.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1243–1252.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6111–6120.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3723–3730.

Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. 2018. Higher-order syntactic attention network for longer sentence compression. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1716–1726.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289.

Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. 2017. Deep recurrent generative decoder for abstractive text summarization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2091–2100.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, page 10.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard H. Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4281–4291.

Ani Nenkova and Kathleen R. McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1073–1083.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhi-Hong Deng. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 3011–3020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Liangguo Wang, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song, and Lejian Liao. 2017. Can syntax help? improving an lstm-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1385–1393.

Wenbo Wang, Yang Gao, Heyan Huang, and Yuxiang Zhou. 2019a. Concept pointer network for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3074–3083.

Yiren Wang, Fei Tian, Di He, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2019b. Non-autoregressive machine translation with auxiliary regularization. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5377–5384.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1304–1312.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Multi-split Reversible Transformers Can Enhance Neural Machine Translation

Yuekai Zhao[1] **Shuchang Zhou**[2] **Zhihua Zhang**[3]
[1] Academy for Advanced Interdisciplinary Studies, Peking University
[2] Megvii Inc.
[3] School of Mathematical Sciences, Peking University
`yuekaizhao@pku.edu.cn`
`zsc@megvii.com`
`zhzhang@math.pku.edu.cn`

Abstract

Large-scale transformers have been shown the state-of-the-art on neural machine translation. However, training these increasingly wider and deeper models could be tremendously memory intensive. We reduce the memory burden by employing the idea of reversible networks that a layer's input can be reconstructed from its output. We design three types of multi-split based reversible transformers. We also devise a corresponding backpropagation algorithm, which does not need to store activations for most layers. Furthermore, we present two fine-tuning techniques: splits shuffle and self ensemble, to boost translation accuracy. Specifically, our best models surpass the vanilla transformer by at least 1.4 BLEU points in three datasets. Our large-scale reversible models achieve 30.0 BLEU in WMT'14 En-De and 43.5 BLEU in WMT'14 En-Fr, beating several very strong baselines with less than half of the training memory.

1 Introduction

Transformers (Vaswani et al., 2017) and their variants (So et al., 2019; Dehghani et al., 2019; Fonollosa et al., 2019; Liu et al., 2020; Zhu et al., 2020) significantly enhance the performance of neural machine translation (NMT). But this often requires a large size of the hidden layer (e.g., Raffel et al. (2019) used a dimension of 65K) or a deeper network by stacking more building blocks (e.g., Liu et al. (2020) used a 60-layer encoder). Training large networks could be extremely memory intensive and might even require model parallelization across multiple GPUs (Brown et al., 2020). As a result, reducing memory consumption is crucial to train wider and deeper networks efficiently.

Backpropagation (BP) is commonly used for training modern neural networks. BP needs to store layer activations to calculate the parameter gradients, which severely increases the memory burden. The idea of reversible networks can be a solution. During training, a reversible network layer's input can be reconstructed from its output. BP is run together with the reconstruction process, removing the need to store all layer activations except for the last layer. We extend the hidden dimension splitting approach by Gomez et al. (2017) and design three types of reversible transformers, namely, simple reversible transformers (**SIM-REV**), single dependent reversible transformers (**SD-REV**) and fully dependent reversible transformers (**FD-REV**). We also devise a corresponding BP algorithm for our reversible models, which is significantly memory efficient compared with the conventional BP.

Our reversible models rely on partitioning each layer's input and output into multiple equal splits. This multi-split feature inspires us to develop two fine-tuning techniques to further enhance translation accuracy. First, we randomly shuffle the output splits to encourage information sharing. Second, we train distinct translation models based on different output splits in the final decoder layer and run model ensemble during inference. These two techniques are applied after model convergence. Only a few epochs of fine-tuning are sufficient for boosting the translation performance. Also, both techniques do not break the reversibility of our proposed models.

We demonstrate that our reversible models can achieve similar or better performance than vanilla transformers do with less memory consumption. Specifically, by employing reversible training and the fine-tuning techniques, our best models can surpass vanilla transformers by 1.5 BLEU (IWSLT'14 De-En), 2.0 BLEU (WMT'19 En-Lt) and 1.4 BLEU (WMT'14 En-De). Our large-scale models also beat several very strong NMT models with less than half the training memory on WMT'14 En-De (30.0 BLEU) and WMT'14 En-Fr (43.5 BLEU).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 244–254
April 19 - 23, 2021.

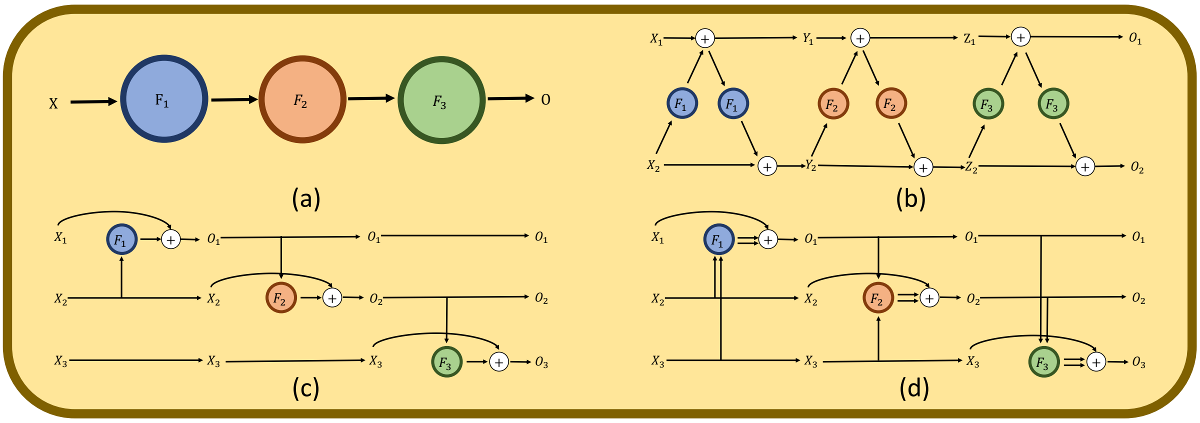


Figure 1: This figure is an illustration of different reversible architectures we explore. (a) shows a vanilla neural network (NN), where F, G and H are modules inside an NN layer. (b) shows **SIM-REV**, which resembles RevNets most. (c) shows a 3-split **SD-REV**. The i-th output split O_i depends only on the i-th input split X_i and O_{i-1}. (d) shows a 3-split **FD-REV**. Each output split O_i depends on all previous output splits $O_{<i}$ and all subsequent input splits $X_{\geq i}$.

2 Methodology

We introduce reversible transformers in this section. The definition and benefits of layer reversibility are given in Section 2.1. Section 2.2 shows three types of reversible architectures based on partitioning the layer input along the hidden/embedding dimension. Section 2.3 details the backpropagation algorithm we use. Finally, in Section 2.4, two techniques that can fit into the reversible training framework are introduced for further boosting model performance.

2.1 Reversible Architectures

A neural network layer is said to be reversible if its input can be reconstructed from its output. Usually, a network is trained in a forward-backward fashion. The activations in each layer are calculated in the forward process and stored for gradient computation in the backward process. The requirement for storing activations is memory intensive and often becomes a bottleneck for network training. However, if a network has reversible building blocks, we do not need to store the activations for most layers since they can be computed during the backward process.

A reversible layer can be designed in two ways. The first way is that this layer has an analytical inverse (Gomez et al., 2017; Jacobsen et al., 2018; Chang et al., 2018; MacKay et al., 2018). The second way is to compute the layer input via numerical methods, e.g., the fixed-point iteration (Behrmann et al., 2019). We focus on the first way following the dimension-splitting approach proposed by RevNets (Gomez et al., 2017). We give a brief review of RevNets. X is the network input, which is split into two halves X_1 and X_2. F and G are modules inside a layer (e.g., 3×3 convolutions). The forward process is as follows:

$$O_1 = X_1 + F(X_2); \;\; O_2 = X_2 + G(O_1). \quad (1)$$

X_1, X_2 can be reconstructed from O_1, O_2 by:

$$X_2 = O_2 - G(O_1); \;\; X_1 = O_1 - F(X_2). \quad (2)$$

2.2 Reversible Transformers

Transformers (Vaswani et al., 2017) achieved the state of the art performance in several tasks (Edunov et al., 2018; Brown et al., 2020). Despite its success, training transformers is memory intensive. We propose three reversible transformers inspired by RevNets (Gomez et al., 2017) to reduce the training memory consumption. X is the layer input. O is the layer output. X and O are partitioned into n equal splits along the hidden/embedding dimension. $X = \{X_1, X_2, \cdots, X_n\}$, $O = \{O_1, O_2, \cdots, O_n\}$. $F_1, F_2, \cdots, F_n$ are modules inside a layer (e.g. self-attention, fully connected layer).

Simple Reversible Transformer (SIM-REV) X is split into two halves X_1, X_2. For each module F_i inside a layer, the forward process resembles RevNets by changing F and G in Equation (1) into F_i. Part (b) of Figure 1 demonstrates the case of a layer with three modules:

$$\begin{aligned} Y_1 = X_1 + F_1(X_2); \;\; & Y_2 = X_2 + F_1(Y_1) \\ Z_1 = Y_1 + F_2(Y_2); \;\; & Z_2 = Y_2 + F_2(Z_1) \quad (3) \\ O_1 = Z_1 + F_3(Z_2); \;\; & O_2 = Z_2 + F_3(O_1) \end{aligned}$$

This is the simplest way to introduce reversibility into a transformer layer. But the computation complexity is doubled compared with vanilla transformers since each module function F_i is used twice.

Single Dependent Reversible Transformer (SD-REV) We propose another reversible architecture to reduce the computational complexity. The i-th output split O_i depends only on X_i and O_{i-1}. The forward process is as follows:

$$\begin{aligned} O_1 &= X_1 + F_1(X_2) \\ O_2 &= X_2 + F_2(O_1) \\ &\dots \\ O_n &= X_n + F_n(O_{n-1}) \end{aligned} \tag{4}$$

The reconstruction of X given O is also straightforward:

$$\begin{aligned} X_n &= O_n - F_n(O_{n-1}) \\ X_{n-1} &= O_{n-1} - F_{n-1}(O_{n-2}) \\ &\dots \\ X_1 &= O_1 - F_1(X_2) \end{aligned} \tag{5}$$

Part (c) of Figure 1 shows a 3-split example of **SD-REV**. With only half of **SIM-REV**'s computational complexity, experiments show that **SD-REV** can achieve similar or even better performance as **SIM-REV** does.

Fully Dependent Reversible Transformer (FD-REV) The **SD-REV** only encodes information in neighbour splits. The lack of interaction between distant splits may make the model less expressive. We force each output split O_i to depend on all previous output splits $O_{<i}$ and all subsequent input splits $X_{\geq i}$, while preserving the reversibility of the network layer. Despite the increased computational complexity, we hope the model to have a better generalization ability. A detailed description of the **FD-REV**'s forward process is as follows:

$$\begin{aligned} O_1 &= X_1 + \sum_{i=2}^{n} F_1(X_i) \\ &\dots \\ O_k &= X_k + \sum_{i=k+1}^{n} F_k(X_i) + \sum_{j=1}^{k-1} F_k(O_j) \\ &\dots \\ O_n &= X_n + \sum_{j=1}^{n-1} F_n(O_j) \end{aligned} \tag{6}$$

The reversibility of **FD-REV** is ensured by:

$$\begin{aligned} X_n &= O_n - \sum_{j=1}^{n-1} F_n(O_j) \\ &\dots \\ X_k &= O_k - \sum_{i=k+1}^{n} F_k(X_i) - \sum_{j=1}^{k-1} F_k(O_j) \\ &\dots \\ X_1 &= O_1 - \sum_{i=2}^{n} F_1(X_i) \end{aligned} \tag{7}$$

Part (d) of Figure 1 illustrates **FD-REV** in a 3-split case. Experiments in Section 3 shows that allowing interaction between distant splits is beneficial for translation performance.

Instantiation The building blocks of transformers are attention based modules and position-wise feed-forward layers:

$$\begin{aligned} &\text{Encoder : Self-Attn} \rightarrow \text{FFN} \\ &\text{Decoder : Self-Attn} \rightarrow \text{Cross-Attn} \rightarrow \text{FFN} \end{aligned}$$

For **SIM-REV**, each of the above modules are transformed into reversible modules, where S is an input/output split:

$$\begin{aligned} \text{Encoder: } F_1(S) &= \alpha(S + \text{Self-Attn}(S)), \\ F_2(S) &= \alpha(S + \text{FFN}(S)) \\ \text{Decoder: } F_1(S) &= \alpha(S + \text{Self-Attn}(S)), \\ F_2(S) &= \alpha(S + \text{Cross-Attn}(S)), \\ F_3(S) &= \alpha(S + \text{FFN}(S)) \end{aligned}$$

For an n-split **SD-REV** or **FD-REV** (actually it has an n-split encoder and an $(n+1)$-split decoder), we use multiple Self-Attn modules and a single Cross-Attn/FFN module within each layer:

$$\begin{aligned} \text{Encoder: } F_{k<n}(S) &= \alpha(S + \text{Self-Attn}_k(S)), \\ F_n(S) &= \alpha(S + \text{FFN}(S)) \\ \text{Decoder: } F_{k<n}(S) &= \alpha(S + \text{Self-Attn}_k(S)), \\ F_n(S) &= \alpha(S + \text{Cross-Attn}(S)), \\ F_{n+1}(S) &= \alpha(S + \text{FFN}(S)) \end{aligned}$$

Applying layer normalization (LN) to the layer output O is crucial to better convergence in training transformers. However, it requires extra storage to calculate the reverse of LN. We use the ReZero (Bachlechner et al., 2020) technique as a substitution of LN. Each layer has a distinct re-scaling

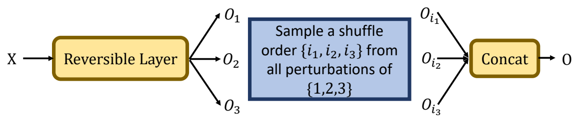


Figure 2: Illustration of shuffling splits

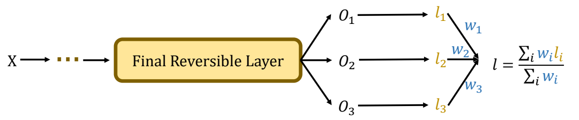


Figure 3: Illustration of self ensemble

weight α which is initialized to zero. α is trained together with other network parameters using Algorithm 1 in Section 2.3.

2.3 Backpropagation with Reconstructing Activations

In the backward pass, we are given the activations $O = \{O_1, \cdots, O_n\}$ and their total derivatives $dO = \{dO_1, \cdots, dO_n\}$. We wish to compute the inputs $X = \{X_1, \cdots, X_n\}$, their total derivatives $dX = \{dX_1, \cdots, dX_n\}$ and the derivatives of model parameters in $F_1, \cdots, F_n$. For **SIM-REV**, the backpropagation (BP) algorithm has no difference with that in Gomez et al. (2017). Such that our main focus is to derive the resulting BP algorithm for **SD-REV** and **FD-REV**.

The forward pass of **SD-REV** and **FD-REV** can be combined into a more general form:

$$O_k = X_k + G_k(X_{i>k}, O_{j<k}, \theta_k) \quad \text{where } k \in \{1, \cdots, n\}. \tag{8}$$

SD-REV corresponds to $G_k = F_k(O_{k-1})$ and **FD-REV** corresponds to $G_k = \sum_{i=k+1}^{n} F_k(X_i) + \sum_{j=1}^{k-1} F_k(O_j)$. Algorithm 1 defines the BP rule of this general form reversible network. Gradients for model parameters are computed in line 9 of Algorithm 1 as a side effect. A repeated apply of Algorithm 1 allows us to perform BP through a sequence of reversible layers, only requiring the activations and their derivatives of the top layer. In this way, the storage cost for activations can be small and independent of network depth.

2.4 Splits Shuffle and Self Ensemble

In this section, we propose two multi-split based fine-tuning methods that can enhance model performance, namely, splits shuffle and self ensemble.

Algorithm 1 BP Algorithm for Multi-Split Reversible Networks

```
Input:
    Layer output: $O = \{O_1, \cdots, O_n\}$;
    Derivatives of $O$: $dO = \{dO_1, \cdots, dO_n\}$;
    Modules: $G_1, \cdots, G_n$;
Output:
    Layer input: $X = \{X_1, \cdots, X_n\}$;
    Derivative of $X$: $dX = \{dX_1, \cdots, dX_n\}$;
 1: $X = \{\}; dX = \{\}$
 2: for k in $n$ to 1 do
 3:     $C = O_k; O = O \setminus \{O_k\}$
 4:     if $k == n$ then
 5:         $\text{grad}_k = dO_k$
 6:     else
 7:         $\text{grad}_k = dO_k + C.\text{grad}$
 8:     end if
 9:     $g_k = G_k(X, O, \theta_k)$; $g_k.\text{backward}^1(\text{grad}_k)$
10:     $X_k = C - g_k; X = X \cup \{X_k\}$
11: end for
12: $dX_1 = \text{grad}_1, dX = \{dX_1\}$
13: for k in 2 to $n$ do
14:     $dX_k = X_k.\text{grad} + \text{grad}_k$
15:     $dX = dX \cup \{dX_k\}$
16: end for
```

First, we train a reversible transformer till convergence. Then, several epochs of fine-tuning with one of these techniques can improve model accuracy.

Splits Shuffle A reversible transformer consists of several reversible layers. The inputs of a certain layer are the outputs of its preceding layer, which we denote as $O = \{O_1, \cdots, O_n\}$. Note that if the order of O_i is randomly shuffled, the whole network is still reversible as long as we keep a record of the shuffling order. This property inspires us to do the following fine-tuning technique:

- For each layer in the reversible network, sample $b \sim \text{Bernoulli}(p)$.
- If $b = 1$, uniformly sample a shuffle order $\{i_1, \cdots, i_n\}$ from all perturbations of $\{1, \cdots, n\}$.
- Next layer's input is $O = \{O_{i_1}, \cdots, O_{i_n}\}$.

Figure 2 shows the splits shuffle process. At inference time, we set p to 0. The idea behind splits shuffle is to apply dropout in the structure level. Splits shuffle provides a way to combine exponentially many network architectures efficiently. In

---

[1] Automatic differentiation routines, e.g. `tf.gradient`, `torch.autograd.backward`

order to let each structure perform well, each split is forced to become more expressive.

Self Ensemble Model ensemble is a commonly used method for boosting translation performance (Zhou et al., 2017; Wang et al., 2020b). Model ensemble usually requires multiple distinct models to output their probability distributions over the vocabulary. The ensemble process is both computational and memory intensive. Our multi-split model offers a new chance that we can view each split of the final output as an independent model. Our self ensemble technique works as follows:

- O_i is a split in the final layer output, y is the translation target, FC stands for a fully connected layer: $P_i = \text{Softmax}(\text{FC}(O_i)); l_i = \text{loss}(P_i, y)$
- Sample a weight $w_i \sim \text{uniform}(0, 1)$. Final loss $l = \sum_{i=1}^{n} w_i l_i / \sum_{i=1}^{n} w_i$.

Figure 3 illustrates the self ensemble method. At inference time, the final distribution is an average of P_i output by each split O_i. We manage to learn an ensemble of transformers in a single reversible transformer. The inference-time memory and computational consumption are largely reduced compared with conventional ensemble methods.

3 Experiments

3.1 Datasets, Architectures and Training

Datasets We experiment on four standard corpora to demonstrate reversible transformers' effectiveness: (1) IWSLT'14 German-English (De-En), which consists of 160K training sentence pairs. (2) WMT'19 English-Lithuanian (En-Lt), which consists of 800K training sentence pairs. (3) WMT'14 English-German (En-De), which consists of 4.5M training sentence pairs. (4) WMT'14 French-English (En-Fr), which consists of 36M training sentence pairs. Tokenization is done by Moses[2]. We employ BPE (Sennrich et al., 2016) to generate a shared vocabulary for each language pair. The BPE merge operation numbers are 10K (IWSLT'14 De-En), 20K (WMT'19 En-Lt), 32K (WMT'14 En-De), 40K (WMT'14 En-Fr). The evaluation metric is *BLEU* (Papineni et al., 2002). We use beam search for test datasets with a beam size of 8 and a length penalty of 0.7.

Architectures We experiment with all architectures proposed in Section 2.2. Multi-split based models have larger hidden dimensions than vanilla transformers do. We use a smaller embedding size than the hidden size by factorizing the word embedding matrix. N is the vocabulary size, d is the hidden size. The original word embedding matrix $E \in R^{N \times d}$ is factorized into a multiplication of two matrices of size $N \times l$ and $l \times d$, where $l \ll d$. We denote l as the embedding size. The embedding size for each language pair is 128 (IWSLT'14 De-En), 256 (WMT'19 En-Lt, WMT'14 En-De base models), 512 (WMT'14 En-De and WMT'14 En-Fr large models). For a specific language pair, we manage to ensure almost identical parameter sizes across different model architectures. One can refer to Appendix A for some details.

Training All models are trained on 8 RTX 2080Ti GPU cards with a mini-batch of 3584 tokens unless otherwise stated. We use the same learning rate scheduling strategy as (Vaswani et al., 2017) does with a warmup step of 4000. The learning rates are set to 5×10^{-4} (IWSLT'14 De-En), 7×10^{-4} (WMT'19 En-Lt, WMT'14 En-De base). The dropout probability and label smoothing factor are all set to 0.1. For training large models in Section 3.4, we increase the dropout probability to 0.3 and the learning rate to 1×10^{-3}. We also accumulate gradients for 16 batches.

3.2 Machine Translation Results

To make comparisons between various architectures, we carry experiments on all corpora except WMT'14 En-Fr. Results are summarized in Table 1. In general, the best reversible architecture can outperform the transformer baseline by 1.1 (IWSTL'14 De-En), 1.5 (WMT'19 En-Lt) and 0.8 (WMT'14 En-De) BLEU points.

All models we propose deliver similar or superior performance to the vanilla transformer. **SD-REV-2** (2 means the split number is 2) is almost as good as **SIM-REV** with only half the computational complexity. For **SD-REV**, translation performance increases as the split number becomes larger. A higher split number means more interaction between separate splits, which may benefit the translation quality. The good performance of **FD-REV** further indicates that interactions between splits should be encouraged. **FD-REV** translates best among different architectures. Increasing the split number is not necessary for **FD-REV**, since it

---

[2] https://github.com/moses-smt/mosesdecoder

Model	**IWSLT'14 De-En**		**WMT'19 En-Lt**		**WMT'14 En-De**	
	Parameters	**BLEU**	**Parameters**	**BLEU**	**Parameters**	**BLEU**
Transformer (Baseline)					61.0 M	27.3
Transformer (Our impl.)	20.3 M	33.7	38.3 M	20.1	62.9 M	27.4
SIM-REV (Ours)	20.3 M	33.8	38.3 M	20.6	62.9 M	27.4
SD-REV-2 (Ours)	20.7 M	33.7	38.0 M	20.5	51.9 M	27.4
SD-REV-3 (Ours)	20.6 M	34.0	38.0 M	20.9	51.6 M	27.8
SD-REV-4 (Ours)	20.2 M	34.2	38.0 M	21.1	51.9 M	28.0
FD-REV-2 (Ours)	20.7 M	**34.8**	38.0 M	21.0	51.9 M	**28.2**
FD-REV-3 (Ours)	20.6 M	34.3	38.0 M	21.0	51.6 M	27.7
FD-REV-4 (Ours)	20.2 M	34.1	38.0 M	**21.6**	51.9 M	27.7
Best + Shuffle Splits (Ours)	20.7 M	35.2	38.0 M	22.0	51.9 M	28.8
Best + Self Ensemble (Ours)	20.7 M	35.1	38.0 M	22.1	51.9 M	28.3

Table 1: Machine translation results on different test sets. **SD-REV-n** represents an n-split single dependent reversible transformer, as is the case with **FD-REV-n**. The best results are all achieved by **FD-REV**. We explore splits shuffle and self ensemble techniques with the best architecture for each language pair. Both of the techniques benefit translation performance. The bold numbers are the best BLEU scores without using the fine-tuning techniques. The numbers with an underline are the overall best BLEU scores for a certain language pair.

Model	**WMT'14 En-De**		**WMT'14 En-Fr**	
	Parameters	**BLEU**	**Parameters**	**BLEU**
Ott et al. (2018)	214.0 M	29.3	214.0 M	43.2
Wu et al. (2019)	213.0 M	29.7	213.0 M	43.2
So et al. (2019)	218.1 M	29.8	221.2 M	41.3
Kitaev et al. (2020)	204.0 M	29.1		
Lioutas and Guo (2020)	209.0 M	29.6	209.6 M	43.2
Best	188.5 M	29.3	193.0 M	42.8
Best + Shuffle Splits	188.5 M	30.0	193.0 M	43.5
Best + Self Ensemble	188.5 M	29.7	193.0 M	43.5

Table 2: Machine Translation Accuracy

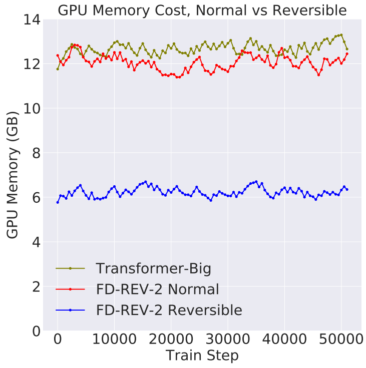


Figure 4: GPU Memory Consumption Comparison

already encodes information from different splits.

The remaining experiments are organized as follows: (1) In Section 3.3, we apply splits shuffle and self ensemble to the best models for each language pair. (2) In Section 3.4, we experiment with large model size for two large corpora, namely, WMT'14 En-De and WMT'14 En-Fr. We also try splits shuffle and self ensemble to validate their effectiveness.

3.3 Splits Shuffle and Self Ensemble

In this section, we focus on fine-tuning techniques to boost translation performance. After model convergence when training with Algorithm 1, we apply splits shuffle or self ensemble for fifteen epochs (IWSLT'14 De-En), five epochs (WMT'19 En-Lt) and one epoch (WMT'14 En-De). For the shuffle probability p, we use 0.3. Results are also summarized in Table 1.

Both fine-tuning techniques yield a performance gain over the original model. Splits shuffle is slightly better than self ensemble. Also, splits shuffle does not increase inference-time computational cost while self ensemble does. Several interesting phenomena are worth mentioning. First, the final validation perplexity decreases for splits shuffle and increases for self ensemble. Since both techniques are helpful, it is reasonable to think that splits shuffle indeed enhances model performance while self

Methods	**De-En**	**En-Lt**
ReZero (+)	34.8	21.6
ReZero (−)	32.8	19.9
Reversible Training (+)	34.8	21.6
Reversible Training (−)	34.8	21.6

Table 3: Ablation Study of ReZero and Reversible Training. **De-En** represents IWSLT'14 De-En. **En-Lt** represents WMT'19 En-Lt. + represents using a certain method. − stands for training without a certain method.

ensemble benefits more from the ensemble process. Second, a combination of splits shuffle and self ensemble fails to converge. The combination task may be too challenging for the model to learn even the model is already in a sub-optimal state. Third, we can use splits shuffle and self ensemble from the beginning. However, such complicated training objectives also bring no performance gain. Details can be found in Appendix B.

3.4 Performance and Memory Consumptions of Large Models

In this section, we investigate large-scale reversible transformers. Experiments focus on two aspects. First, whether reversible models' performance is comparable or even better than the non-reversible models? Second, how much GPU memory can be saved when using Algorithm 1 for backpropagation (BP). We choose **FD-REV-2** which performs best for WMT'14 En-De in Section 3.2. The hidden dimension is doubled, resulting in a similar parameter size with other large-scale models.

As shown in Table 2, **FD-REV-2** achieves comparable results in both datasets. The fine-tuning techniques in Section 2.4 offer a chance to enhance model performance further. We follow the settings for WMT'14 En-De in Section 3.3 and find out that large-scale models benefit more from splits shuffle and self ensemble. We can surpass various strong baselines by using splits shuffle for only one epoch of fine-tuning. Specifically, we achieve 30.0 BLEU points for WMT'14 En-De and 43.5 BLEU points for WMT'14 En-Fr.

We also compare the training memory consumption between three different settings: (1) Training Transformer-Big with conventional BP. (2) Training **FD-REV-2** with conventional BP. (3) Training **FD-REV-2** with Algorithm 1. WMT'14 En-De is the training dataset. We use a batch size of 2390 tokens and carry out one epoch of training. Figure 4 illustrates the memory consumptions of three training settings. Transformer-Big and **FD-REV-2** are similar to each other in GPU memory consumption. Reversible BP with Algorithm 1 removes the need to store activations for most layers, requiring about half of the GPU memory as conventional BP does.

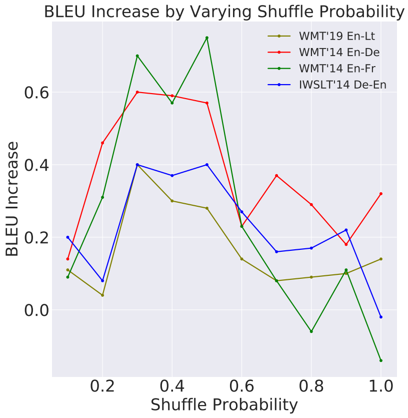


Figure 5: Effects of Shuffle Probability p

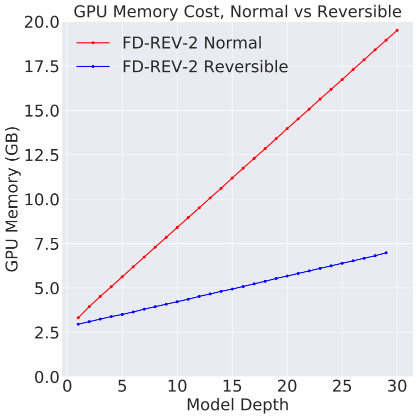


Figure 6: GPU Memory Consumption vs. Depth

4 Analysis

Splits Shuffle Probability We study the impact of splits shuffle probability p. For all language pairs, we use the best models as mentioned in Section 3.3 and Section 3.4. The results are summarized in Figure 5. We find out that a medium p value ($0.3 \leq p \leq 0.5$) yields the largest BLEU increase. A small p value is insufficient to enhance model generalization ability since the model has already been optimized for several epochs. Meanwhile, a very large p value makes the model highly unstable and hard to converge.

ReZero We study the impact of the ReZero technique. ReZero works as a substitution for layer normalization (LN). We can not apply LN to each layer's output O due to the requirement of reversibility. Instead, we can apply LN to each output split O_i. We compare using ReZero with using LN on O_i. Table 3 shows the results. Translation performance drops severely and the model converges more slowly. Therefore we choose to use ReZero throughout our experiments.

Reversible Training vs. Normal Training Reversible training saves GPU memory. However, reconstructing activations over many layers can introduce numerical errors. Inaccurate gradients may hurt model performance, so that it is important to compare the model performance between using reversible training and conventional backpropagation (BP). As shown in Table 3, reversible training does not hurt model performance. Also, since we update the model parameters for the same number of times, the convergence speed is almost identical between reversible training and conventional BP.

Memory Consumption of Deep Models Reversible transformers are more memory efficient when the model gets deeper. We validate this argument with a simple experiment. First, we get a mini-batch of 2390 tokens. Then, one step of parameter update is done by conventional BP or reversible training. We gradually increase the model depth and keep a record of the corresponding memory consumption in a single RTX TITAN GPU card. Results are shown in Figure 6. Deeper models mean more activations to store when using conventional BP, while reversible training only needs to store the extra model parameters. The memory consumption gap can be up to 12.6 GB when we use a 30-layers encoder and a 30-layers decoder.

Computational Overhead Roughly speaking, our proposed network is composed mostly of fully connected (FC) layers. For an FC layer with N connections, the forward and backward passes require approximately N and $2N$ add-multiply operations, respectively. As the reconstruction during backpropagation (BP) adds another N add-multiply operations, training with Algorithm 1 will be 33% slower. We compare the training speed of 4 structures, namely, **FD-REV-2-base**, **FD-REV-2-big**, **Transformer-base**, **Transformer-big**. We train on WMT'14 En-De for one epoch, using a batch size of 3584 tokens. Experiments are done on a single RTX Titan GPU card with 24 GB memory. The speed measurement is words per second (wps).

Structures	**use-rev**	**Speed (wps)**
FD-REV-2-base	×	10320
FD-REV-2-base	✓	7831
FD-REV-2-big	×	3735
FD-REV-2-big	✓	2788
Transformer-base	×	10081
Transformer-big	×	3879

Table 4: Training speed comparison between Transformers and our best reversible networks. The term **use-rev** being ✓ means training with Algorithm 1. Otherwise, we train the network with conventional backpropagation.

From Table 4, we can see that **FD-REV-2** trains almost as fast as vanilla Transformers when employing conventional BP. The apply of Algorithm 1 adds about 33% to 38% training time, which in turn saves about half the memory consumption.

5 Related Work

Reversible Networks The idea of reversible training without storing activations was first introduced by RevNets (Gomez et al., 2017). Jacobsen et al. (2018) attempted to use RevNets to learn representations without loss of information. Chang et al. (2018) associated well-posed ODEs with reversible networks. MacKay et al. (2018) extended RevNets to recurrent networks. Behrmann et al. (2019) showed that a simple normalization step could make standard ResNet architectures invertible. Generative flows are often combined with reversible networks. Kingma and Dhariwal (2018) used invertible convolutions to train a generative model. Later works (Huang et al., 2018; Tran et al., 2019; Ma et al., 2019) studied generative flows with reversible networks in discrete data.

Memory-Efficient Transformers Creating memory-efficient transformers has attracted immense interest in recent years. Most works focus on proposing an optimized version of the attention module. Liu et al. (2018); Child et al. (2019); Kitaev et al. (2020); Ainslie et al. (2020); Tay et al. (2020b) limited the attention span to local neighborhoods. Wang et al. (2020a); Tay et al. (2020a) employed low rank approximations for attention matrices. Roy et al. (2020) achieved sparse attention by k-mean clustering. Katharopoulos et al. (2020) used a kernel-based self-attention to reduce memory consumption. (Ho et al., 2019) operates on multi-dimensional tensors and applies multiple attentions, each along a single axis of the input tensor. Several works (Wu et al., 2019; Lioutas and Guo, 2020; Beltagy et al., 2020; Zaheer et al., 2020; Wu et al., 2020) incorporate convolution networks into transformers. Except for inventing new attention modules, weight sharing (Dehghani et al., 2019; Bai et al., 2019; Lan et al., 2020) is another practical approach to decreasing the memory burden. Reversible models are orthogonal to these approaches. A combination of reversible models and variants of transformers can further reduce memory consumption.

6 Conclusion

We have presented three types of multi-split based reversible transformers which outperform vanilla transformers. During backpropagation, activations for most layers need not be stored in memory because they can be reconstructed. Furthermore, we have proposed two fine-tuning techniques, namely, splits shuffle and self ensemble. Both techniques are easy to implement, and only a few fine-tuning epochs are sufficient for boosting translation performance. Our approach has beaten several strong baselines in two large datasets with fewer model parameters and much less training memory. Specifically, we have achieved 30.0 BLEU points in WMT'14 En-De and 43.5 BLEU points in WMT'14 En-Fr. Also, one can transform other network structures into their reversible versions by applying our methods. We would explore more computer vision or natural language processing tasks to widen reversible networks' applicability.

Acknowledgments

Yuekai Zhao and Zhihua Zhang have been supported by the Beijing Natural Science Foundation (Z190001), National Key Research and Development Project of China (No. 2018AAA0101004), and Beijing Academy of Artificial Intelligence (BAAI).

References

Joshua Ainslie, Santiago Ontañón, Chris Alberti, Philip Pham, Anirudh Ravula, and Sumit Sanghai. 2020. Etc: Encoding long and structured data in transformers. *CoRR*, abs/2004.08483.

Thomas Bachlechner, Huanru Henry Majumder, Bodhisattwa Prasad Mao, Garrison W. Cottrell, and Julian McAuley. 2020. Rezero is all you need: Fast convergence at large depth. In *arXiv*.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2019. Deep equilibrium models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. 2019. Invertible residual networks. volume 97 of *Proceedings of Machine Learning Research*, pages 573–582, Long Beach, California, USA. PMLR.

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

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

B. Chang, L. Meng, E. Haber, Lars Ruthotto, David Begert, and E. Holtham. 2018. Reversible architectures for arbitrarily deep residual neural networks. In *AAAI*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

José A. R. Fonollosa, Noe Casas, and Marta R. Costa-jussà. 2019. Joint source-target self attention with locality constraints. *arXiv preprint arXiv:1905.06596*.

Aidan Gomez, Mengye Ren, Raquel Urtasun, and Roger Grosse. 2017. The reversible residual network: Backpropagation without storing activations.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. 2019. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. 2018. Neural autoregressive flows. volume 80 of *Proceedings of Machine Learning Research*, pages 2078–2087, Stockholmsmässan, Stockholm Sweden. PMLR.

Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. 2018. i-revnet: Deep invertible networks.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10215–10224. Curran Associates, Inc.

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

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Vasileios Lioutas and Yuhong Guo. 2020. Time-aware large kernel convolutions. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Peter J. Liu, Mohammad Ahmad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences.

Xiaodong Liu, Kevin Duh, Liyuan Liu, and Jianfeng Gao. 2020. Very deep transformers for neural machine translation. *arXiv preprint arXiv:2008.07772*.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4282–4292, Hong Kong, China. Association for Computational Linguistics.

Matthew MacKay, Paul Vicol, Jimmy Ba, and Roger Grosse. 2018. Reversible recurrent neural networks. In *Neural Information Processing Systems (NeurIPS)*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Aurko Roy, Mohammad Taghi Saffar, David Grangier, and Ashish Vaswani. 2020. Efficient content-based sparse attention with routing transformers.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

David So, Quoc Le, and Chen Liang. 2019. The evolved transformer. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5877–5886.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2020a. Synthesizer: Rethinking self-attention in transformer models. *arXiv preprint arXiv:2005.00743*.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020b. Sparse sinkhorn attention. *arXiv preprint arXiv:2002.11296*.

Dustin Tran, Keyon Vafa, Kumar Agrawal, Laurent Dinh, and Ben Poole. 2019. Discrete flows: Invertible generative models of discrete data. In *Advances in Neural Information Processing Systems*, pages 14719–14728.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. 2020a. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Yiren Wang, Lijun Wu, Yingce Xia, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2020b. Transductive ensemble learning for neural machine translation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI*

2020, New York, NY, USA, February 7-12, 2020, pages 6291–6298. AAAI Press.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*.

Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. 2020. Lite transformer with long-short range attention. In *International Conference on Learning Representations*.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Long Zhou, Wenpeng Hu, Jiajun Zhang, and Chengqing Zong. 2017. Neural system combination for machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 378–384, Vancouver, Canada. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, W. Zhou, H. Li, and T. Liu. 2020. Incorporating bert into neural machine translation. *ArXiv*, abs/2002.06823.

A Model Configuration

Table 5 details the model hyper-parameters. As we use a factorized word embedding matrix, the embedding size l is smaller than the hidden dimension d. The hidden size d increases with the number of splits n to ensure a similar parameter size for a certain dataset. Another thing worth mentioning is that **SD-REV** and **FD-REV** have identical parameter sizes as long as they have the same number of splits n, embedding size l and hidden size d. Thus, we do not differentiate between **SD-REV** and **FD-REV** in Table 5.

B More Results on Splits Shuffle and Self Ensemble

We provide more experimental results on splits shuffle and self ensemble as shown in Figure 7 and Figure 8. Using any of the techniques from the beginning tend to hurt the model performance. Meanwhile, fine-tuning with splits shuffle or self ensemble after model convergence can bring some performance gain. A combination of splits shuffle and self ensemble fails to converge when limiting the number of fine-tuning epochs (15 epochs for IWSLT'14 De-En, 5 epochs for WMT'19 En-Lt, 1 epoch for WMT'14 En-De).

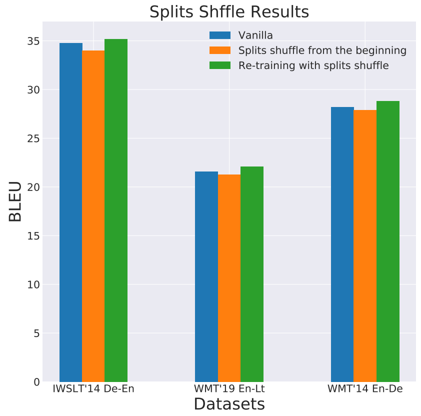


Figure 7: Splits shuffle results. No splits shuffle vs. Fine-tuning vs. Training from the beginning

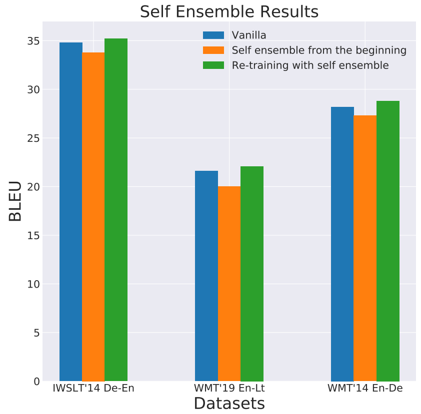


Figure 8: Self ensemble results. No self ensemble vs. Fine-tuning vs. Training from the beginning

Dataset	n	l	d
IWSLT'14 De-En	2	128	708
IWSLT'14 De-En	3	128	768
IWSLT'14 De-En	4	128	800
WMT'19 En-Lt	2	256	900
WMT'19 En-Lt	3	256	960
WMT'19 En-Lt	4	256	1040
WMT'14 En-De	2	256	1152
WMT'14 En-De	3	256	1200
WMT'14 En-De (Base)	4	256	1280
WMT'14 En-De (Big)	2	512	2304
WMT'14 En-Fr	2	512	2304

Table 5: Model configuration for **SD-REV** and **FD-REV**.n represents number of splits. l is the embedding size as mentioned in Section 3.1. d is the hidden size.

Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference

Timo Schick[1,2] **Hinrich Schütze**[1]
[1] Center for Information and Language Processing, LMU Munich, Germany
[2] Sulzer GmbH, Munich, Germany
`schickt@cis.lmu.de` `inquiries@cislmu.org`

Abstract

Some NLP tasks can be solved in a fully unsupervised fashion by providing a pretrained language model with "task descriptions" in natural language (e.g., Radford et al., 2019). While this approach underperforms its supervised counterpart, we show in this work that the two ideas can be combined: We introduce Pattern-Exploiting Training (PET), a semi-supervised training procedure that reformulates input examples as cloze-style phrases to help language models understand a given task. These phrases are then used to assign soft labels to a large set of unlabeled examples. Finally, standard supervised training is performed on the resulting training set. For several tasks and languages, PET outperforms supervised training and strong semi-supervised approaches in low-resource settings by a large margin.[1]

1 Introduction

Learning from examples is the predominant approach for many NLP tasks: A model is trained on a set of labeled examples from which it then generalizes to unseen data. Due to the vast number of languages, domains and tasks and the cost of annotating data, it is common in real-world uses of NLP to have only a small number of labeled examples, making *few-shot learning* a highly important research area. Unfortunately, applying standard supervised learning to small training sets often performs poorly; many problems are difficult to grasp from just looking at a few examples. For instance, assume we are given the following pieces of text:

- T_1: This was the best pizza I've ever had.
- T_2: You can get better sushi for half the price.
- T_3: Pizza was average. Not worth the price.

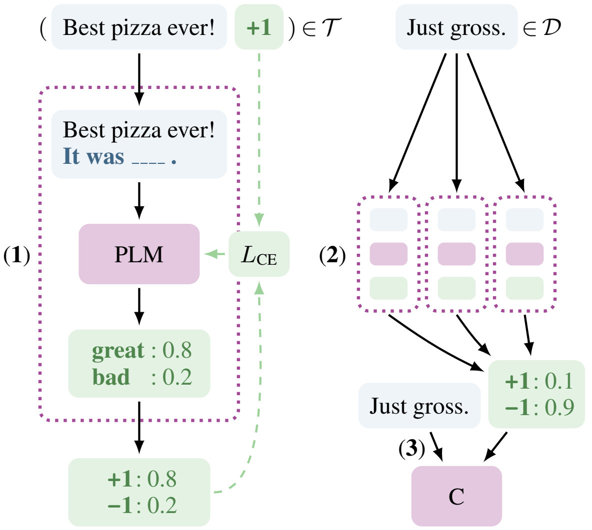


Figure 1: PET for sentiment classification. **(1)** A number of patterns encoding some form of task description are created to convert training examples to cloze questions; for each pattern, a pretrained language model is finetuned. **(2)** The ensemble of trained models annotates unlabeled data. **(3)** A classifier is trained on the resulting soft-labeled dataset.

Furthermore, imagine we are told that the labels of T_1 and T_2 are l and l', respectively, and we are asked to infer the correct label for T_3. Based only on these examples, this is impossible because plausible justifications can be found for both l and l'. However, if we know that the underlying task is to identify whether the text says anything about prices, we can easily assign l' to T_3. This illustrates that solving a task from only a few examples becomes much easier when we also have a *task description*, i.e., a textual explanation that helps us *understand* what the task is about.

With the rise of pretrained language models (PLMs) such as GPT (Radford et al., 2018), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), the idea of providing task descriptions has become feasible for neural architectures: We can

---

[1] Our implementation is publicly available at `https://github.com/timoschick/pet`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 255–269
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

simply append such descriptions in natural language to an input and let the PLM predict continuations that solve the task (Radford et al., 2019; Puri and Catanzaro, 2019). So far, this idea has mostly been considered in zero-shot scenarios where no training data is available at all.

In this work, we show that providing task descriptions can successfully be combined with standard supervised learning in few-shot settings: We introduce **P**attern-**E**xploiting **T**raining (PET), a semi-supervised training procedure that uses natural language patterns to reformulate input examples into cloze-style phrases. As illustrated in Figure 1, PET works in three steps: First, for each pattern a separate PLM is finetuned on a small training set $\mathcal{T}$. The ensemble of all models is then used to annotate a large unlabeled dataset $\mathcal{D}$ with soft labels. Finally, a standard classifier is trained on the soft-labeled dataset. We also devise iPET, an iterative variant of PET in which this process is repeated with increasing training set sizes.

On a diverse set of tasks in multiple languages, we show that given a small to medium number of labeled examples, PET and iPET substantially outperform unsupervised approaches, supervised training and strong semi-supervised baselines.

2 Related Work

Radford et al. (2019) provide hints in the form of natural language patterns for zero-shot learning of challenging tasks such as reading comprehension and question answering (QA). This idea has been applied to unsupervised text classification (Puri and Catanzaro, 2019), commonsense knowledge mining (Davison et al., 2019) and argumentative relation classification (Opitz, 2019). Srivastava et al. (2018) use task descriptions for zero-shot classification but require a semantic parser. For relation extraction, Bouraoui et al. (2020) automatically identify patterns that express given relations. McCann et al. (2018) rephrase several tasks as QA problems. Raffel et al. (2020) frame various problems as language modeling tasks, but their patterns only loosely resemble natural language and are unsuitable for few-shot learning.[2]

Another recent line of work uses cloze-style phrases to probe the knowledge that PLMs acquire during pretraining; this includes probing for factual and commonsense knowledge (Trinh and Le, 2018; Petroni et al., 2019; Wang et al., 2019; Sakaguchi et al., 2020), linguistic capabilities (Ettinger, 2020; Kassner and Schütze, 2020), understanding of rare words (Schick and Schütze, 2020), and ability to perform symbolic reasoning (Talmor et al., 2019). Jiang et al. (2020) consider the problem of finding the best pattern to express a given task.

Other approaches for few-shot learning in NLP include exploiting examples from related tasks (Yu et al., 2018; Gu et al., 2018; Dou et al., 2019; Qian and Yu, 2019; Yin et al., 2019) and using data augmentation (Xie et al., 2020; Chen et al., 2020); the latter commonly relies on back-translation (Sennrich et al., 2016), requiring large amounts of parallel data. Approaches using textual class descriptors typically assume that abundant examples are available for a subset of classes (e.g., Romera-Paredes and Torr, 2015; Veeranna et al., 2016; Ye et al., 2020). In contrast, our approach requires no additional labeled data and provides an intuitive interface to leverage task-specific human knowledge.

The idea behind iPET – training multiple generations of models on data labeled by previous generations – bears resemblance to self-training and bootstrapping approaches for word sense disambiguation (Yarowsky, 1995), relation extraction (Brin, 1999; Agichtein and Gravano, 2000; Batista et al., 2015), parsing (McClosky et al., 2006; Reichart and Rappoport, 2007; Huang and Harper, 2009), machine translation (Hoang et al., 2018), and sequence generation (He et al., 2020).

3 Pattern-Exploiting Training

Let M be a masked language model with vocabulary V and mask token $\_\_\_ \in V$, and let $\mathcal{L}$ be a set of labels for our target classification task A. We write an input for task A as a sequence of phrases $\mathbf{x} = (s_1, \dots, s_k)$ with $s_i \in V^*$; for example, $k = 2$ if A is textual inference (two input sentences). We define a *pattern* to be a function P that takes $\mathbf{x}$ as input and outputs a phrase or sentence $P(\mathbf{x}) \in V^*$ that contains exactly one mask token, i.e., its output can be viewed as a cloze question. Furthermore, we define a *verbalizer* as an injective function $v : \mathcal{L} \rightarrow V$ that maps each label to a word from M's vocabulary. We refer to (P, v) as a *pattern-verbalizer pair* (PVP).

Using a PVP (P, v) enables us to solve task A as follows: Given an input $\mathbf{x}$, we apply P to obtain an input representation $P(\mathbf{x})$, which is then processed

---

[2] For example, they convert inputs (a, b) for recognizing textual entailment (RTE) to "rte sentence1: a sentence2: b", and the PLM is asked to predict strings like "not_entailment".

by M to determine the label $y \in \mathcal{L}$ for which $v(y)$ is the most likely substitute for the mask. For example, consider the task of identifying whether two sentences a and b contradict each other (label y_0) or agree with each other (y_1). For this task, we may choose the pattern $P(a, b) =$ a? ____, b. combined with a verbalizer v that maps y_0 to "Yes" and y_1 to "No". Given an example input pair

$$\mathbf{x} = (\text{Mia likes pie, Mia hates pie}),$$

the task now changes from having to assign a label without inherent meaning to answering whether the most likely choice for the masked position in

$$P(\mathbf{x}) = \text{Mia likes pie? \_\_\_\_, Mia hates pie.}$$

is "Yes" or "No".

3.1 PVP Training and Inference

Let $\mathbf{p} = (P, v)$ be a PVP. We assume access to a small training set $\mathcal{T}$ and a (typically much larger) set of unlabeled examples $\mathcal{D}$. For each sequence $\mathbf{z} \in V^*$ that contains exactly one mask token and $w \in V$, we denote with $M(w \mid \mathbf{z})$ the unnormalized score that the language model assigns to w at the masked position. Given some input $\mathbf{x}$, we define the score for label $l \in \mathcal{L}$ as

$$s_{\mathbf{p}}(l \mid \mathbf{x}) = M(v(l) \mid P(\mathbf{x}))$$

and obtain a probability distribution over labels using softmax:

$$q_{\mathbf{p}}(l \mid \mathbf{x}) = \frac{e^{s_{\mathbf{p}}(l \mid \mathbf{x})}}{\sum_{l' \in \mathcal{L}} e^{s_{\mathbf{p}}(l' \mid \mathbf{x})}}$$

We use the cross-entropy between $q_{\mathbf{p}}(l \mid \mathbf{x})$ and the true (one-hot) distribution of training example $(\mathbf{x}, l)$ – summed over all $(\mathbf{x}, l) \in \mathcal{T}$ – as loss for finetuning M for $\mathbf{p}$.

3.2 Auxiliary Language Modeling

In our application scenario, only a few training examples are available and catastrophic forgetting can occur. As a PLM finetuned for some PVP is still a language model at its core, we address this by using language modeling as auxiliary task. With L_{CE} denoting cross-entropy loss and L_{MLM} language modeling loss, we compute the final loss as

$$L = (1 - \alpha) \cdot L_{\text{CE}} + \alpha \cdot L_{\text{MLM}}$$

This idea was recently applied by Chronopoulou et al. (2019) in a data-rich scenario. As L_{MLM} is typically much larger than L_{CE}, in preliminary experiments, we found a small value of $\alpha = 10^{-4}$ to consistently give good results, so we use it in all our experiments. To obtain sentences for language modeling, we use the unlabeled set $\mathcal{D}$. However, we do not train directly on each $\mathbf{x} \in \mathcal{D}$, but rather on $P(\mathbf{x})$, where we never ask the language model to predict anything for the masked slot.

3.3 Combining PVPs

A key challenge for our approach is that in the absence of a large development set, it is hard to identify which PVPs perform well. To address this, we use a strategy similar to knowledge distillation (Hinton et al., 2015). First, we define a set $\mathcal{P}$ of PVPs that intuitively make sense for a given task A. We then use these PVPs as follows:

(1) We finetune a separate language model $M_{\mathbf{p}}$ for each $\mathbf{p} \in \mathcal{P}$ as described in Section 3.1. As $\mathcal{T}$ is small, this finetuning is cheap even for a large number of PVPs.

(2) We use the ensemble $\mathcal{M} = \{M_{\mathbf{p}} \mid \mathbf{p} \in \mathcal{P}\}$ of finetuned models to annotate examples from $\mathcal{D}$. We first combine the unnormalized class scores for each example $\mathbf{x} \in \mathcal{D}$ as

$$s_{\mathcal{M}}(l \mid \mathbf{x}) = \frac{1}{Z} \sum_{\mathbf{p} \in \mathcal{P}} w(\mathbf{p}) \cdot s_{\mathbf{p}}(l \mid \mathbf{x})$$

where $Z = \sum_{\mathbf{p} \in \mathcal{P}} w(\mathbf{p})$ and the $w(\mathbf{p})$ are weighting terms for the PVPs. We experiment with two different realizations of this weighing term: either we simply set $w(\mathbf{p}) = 1$ for all $\mathbf{p}$ or we set $w(\mathbf{p})$ to be the accuracy obtained using $\mathbf{p}$ on the training set *before* training. We refer to these two variants as *uniform* and *weighted*. Jiang et al. (2020) use a similar idea in a zero-shot setting.

We transform the above scores into a probability distribution q using softmax. Following Hinton et al. (2015), we use a temperature of $T = 2$ to obtain a suitably soft distribution. All pairs $(\mathbf{x}, q)$ are collected in a (soft-labeled) training set $\mathcal{T}_C$.

(3) We finetune a PLM C with a standard sequence classification head on $\mathcal{T}_C$.

The finetuned model C then serves as our classifier for A. All steps described above are depicted in Figure 2; an example is shown in Figure 1.

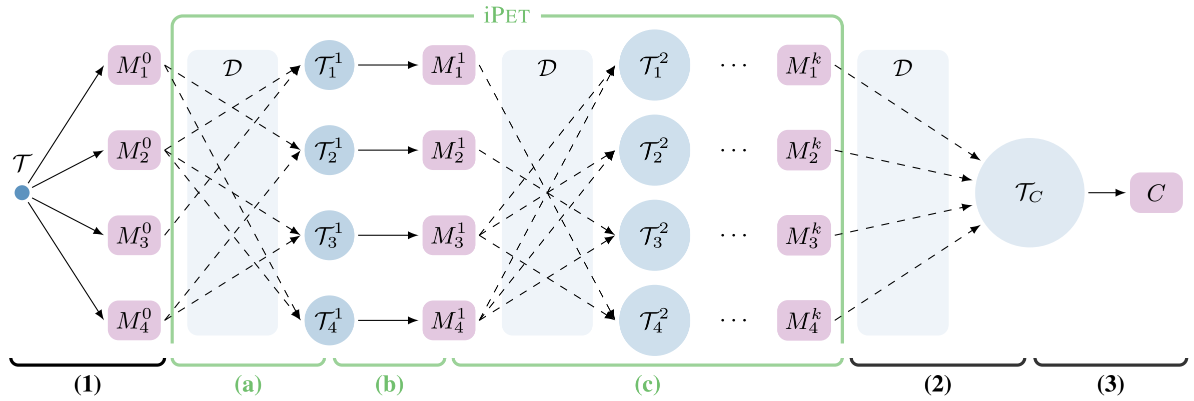


Figure 2: Schematic representation of PET (1-3) and iPET (a-c). **(1)** The initial training set is used to finetune an ensemble of PLMs. **(a)** For each model, a random subset of other models generates a new training set by labeling examples from $\mathcal{D}$. **(b)** A new set of PET models is trained using the larger, model-specific datasets. **(c)** The previous two steps are repeated k times, each time increasing the size of the generated training sets by a factor of d. **(2)** The final set of models is used to create a soft-labeled dataset $\mathcal{T}_C$. **(3)** A classifier C is trained on this dataset.

3.4 Iterative PET (iPET)

Distilling the knowledge of all individual models into a single classifier C means they cannot learn from each other. As some patterns perform (possibly much) worse than others, the training set $\mathcal{T}_C$ for our final model may therefore contain many mislabeled examples.

To compensate for this shortcoming, we devise iPET, an iterative variant of PET. The core idea of iPET is to train several *generations* of models on datasets of increasing size. To this end, we first enlarge the original dataset $\mathcal{T}$ by labeling selected examples from $\mathcal{D}$ using a random subset of trained PET models (Figure 2a). We then train a new generation of PET models on the enlarged dataset (b); this process is repeated several times (c).

More formally, let $\mathcal{M}^0 = \{M_1^0, \dots, M_n^0\}$ be the initial set of PET models finetuned on $\mathcal{T}$, where each M_i^0 is trained for some PVP $\mathbf{p}_i$. We train k generations of models $\mathcal{M}^1, \dots, \mathcal{M}^k$ where $\mathcal{M}^j = \{M_1^j, \dots, M_n^j\}$ and each M_i^j is trained for $\mathbf{p}_i$ on its own training set $\mathcal{T}_i^j$. In each iteration, we multiply the training set size by a fixed constant $d \in \mathbb{N}$ while maintaining the label ratio of the original dataset. That is, with $c_0(l)$ denoting the number of examples with label l in $\mathcal{T}$, each $\mathcal{T}_i^j$ contains $c_j(l) = d \cdot c_{j-1}(l)$ examples with label l. This is achieved by generating each $\mathcal{T}_i^j$ as follows:

1. We obtain $\mathcal{N} \subset \mathcal{M}^{j-1} \setminus \{M_i^{j-1}\}$ by randomly choosing $\lambda \cdot (n-1)$ models from the previous generation with $\lambda \in (0, 1]$ being a hyperparameter.
2. Using this subset, we create a labeled dataset
$$\mathcal{T}_\mathcal{N} = \{(\mathbf{x}, \arg\max_{l \in \mathcal{L}} s_\mathcal{N}(l \mid \mathbf{x})) \mid \mathbf{x} \in \mathcal{D}\}\,.$$
For each $l \in \mathcal{L}$, we obtain $\mathcal{T}_\mathcal{N}(l) \subset \mathcal{T}_\mathcal{N}$ by randomly choosing $c_j(l) - c_0(l)$ examples with label l from $\mathcal{T}_\mathcal{N}$. To avoid training future generations on mislabeled data, we prefer examples for which the ensemble of models is confident in its prediction. The underlying intuition is that even without calibration, examples for which labels are predicted with high confidence are typically more likely to be classified correctly (Guo et al., 2017). Therefore, when drawing from $\mathcal{T}_\mathcal{N}$, we set the probability of each $(\mathbf{x}, y)$ proportional to $s_\mathcal{N}(l \mid \mathbf{x})$.
3. We define $\mathcal{T}_i^j = \mathcal{T} \cup \bigcup_{l \in \mathcal{L}} \mathcal{T}_\mathcal{N}(l)$. As can easily be verified, this dataset contains $c_j(l)$ examples for each $l \in \mathcal{L}$.

After training k generations of PET models, we use $\mathcal{M}^k$ to create $\mathcal{T}_C$ and train C as in basic PET.

With minor adjustments, iPET can even be used in a zero-shot setting. To this end, we define $\mathcal{M}^0$ to be the set of *untrained* models and $c_1(l) = 10/|\mathcal{L}|$ for all $l \in \mathcal{L}$ so that $\mathcal{M}^1$ is trained on 10 examples evenly distributed across all labels. As $\mathcal{T}_\mathcal{N}$ may not contain enough examples for some label l, we create all $\mathcal{T}_\mathcal{N}(l)$ by sampling from the 100 examples $\mathbf{x} \in \mathcal{D}$ for which $s_\mathcal{N}(l \mid x)$ is the highest, even if $l \neq \arg\max_{l \in \mathcal{L}} s_\mathcal{N}(l \mid x)$. For each subsequent generation, we proceed exactly as in basic iPET.

4 Experiments

We evaluate PET on four English datasets: Yelp Reviews, AG's News, Yahoo Questions (Zhang et al., 2015) and MNLI (Williams et al., 2018). Additionally, we use x-stance (Vamvas and Sennrich, 2020) to investigate how well PET works for other languages. For all experiments on English, we use RoBERTa large (Liu et al., 2019) as language model; for x-stance, we use XLM-R (Conneau et al., 2020). We investigate the performance of PET and all baselines for different training set sizes; each model is trained three times using different seeds and average results are reported.

As we consider a few-shot setting, we assume no access to a large development set on which hyperparameters could be optimized. Our choice of hyperparameters is thus based on choices made in previous work and practical considerations. We use a learning rate of $1 \cdot 10^{-5}$, a batch size of 16 and a maximum sequence length of 256. Unless otherwise specified, we always use the weighted variant of PET with auxiliary language modeling. For iPET, we set $\lambda = 0.25$ and $d = 5$; that is, we select 25% of all models to label examples for the next generation and quintuple the number of training examples in each iteration. We train new generations until each model was trained on at least 1000 examples, i.e., we set $k = \lceil \log_d(1000/|\mathcal{T}|) \rceil$. As we always repeat training three times, the ensemble $\mathcal{M}$ (or $\mathcal{M}^0$) for n PVPs contains $3n$ models. Further hyperparameters and detailed explanations for all our choices are given in Appendix B.

4.1 Patterns

We now describe the patterns and verbalizers used for all tasks. We use two vertical bars (‖) to mark boundaries between text segments.[3]

Yelp For the Yelp Reviews Full Star dataset (Zhang et al., 2015), the task is to estimate the rating that a customer gave to a restaurant on a 1- to 5-star scale based on their review's text. We define the following patterns for an input text a:

$$P_1(a) = \text{It was \_\_\_\_ . } a \qquad P_2(a) = \text{Just \_\_\_\_ ! } \| \; a$$

$$P_3(a) = a\text{. All in all, it was \_\_\_\_ .}$$

$$P_4(a) = a \; \| \; \text{In summary, the restaurant is \_\_\_\_ .}$$

We define a single verbalizer v for all patterns as

$$v(1) = \text{terrible} \quad v(2) = \text{bad} \quad v(3) = \text{okay}$$
$$v(4) = \text{good} \quad v(5) = \text{great}$$

AG's News AG's News is a news classification dataset, where given a headline a and text body b, news have to be classified as belonging to one of the categories *World* (1), *Sports* (2), *Business* (3) or *Science/Tech* (4). For $\mathbf{x} = (a, b)$, we define the following patterns:

$$P_1(\mathbf{x}) = \text{\_\_\_\_: } a \; b \qquad P_2(\mathbf{x}) = a \text{ (\_\_\_\_) } b$$

$$P_3(\mathbf{x}) = \text{\_\_\_\_ -- } a \; b \qquad P_4(\mathbf{x}) = a \; b \text{ (\_\_\_\_)}$$

$$P_5(\mathbf{x}) = \text{\_\_\_\_ News: } a \; b$$

$$P_6(\mathbf{x}) = \text{[Category: \_\_\_\_] } a \; b$$

We use a verbalizer that maps 1–4 to "World", "Sports", "Business" and "Tech", respectively.

Yahoo Yahoo Questions (Zhang et al., 2015) is a text classification dataset. Given a question a and an answer b, one of ten possible categories has to be assigned. We use the same patterns as for AG's News, but we replace the word "News" in P_5 with the word "Question". We define a verbalizer that maps categories 1–10 to "Society", "Science", "Health", "Education", "Computer", "Sports", "Business", "Entertainment", "Relationship" and "Politics".

MNLI The MNLI dataset (Williams et al., 2018) consists of text pairs $\mathbf{x} = (a, b)$. The task is to find out whether a implies b (0), a and b contradict each other (1) or neither (2). We define

$$P_1(\mathbf{x}) = \text{“}a\text{”? } \| \text{ \_\_\_\_, “}b\text{”} \qquad P_2(\mathbf{x}) = a\text{? } \| \text{ \_\_\_\_, } b$$

and consider two different verbalizers v_1 and v_2:

$$v_1(0) = \text{Wrong} \quad v_1(1) = \text{Right} \quad v_1(2) = \text{Maybe}$$
$$v_2(0) = \text{No} \quad v_2(1) = \text{Yes} \quad v_2(2) = \text{Maybe}$$

Combining the two patterns with the two verbalizers results in a total of 4 PVPs.

X-Stance The x-stance dataset (Vamvas and Sennrich, 2020) is a multilingual stance detection dataset with German, French and Italian examples. Each example $\mathbf{x} = (a, b)$ consists of a question a concerning some political issue and a comment b; the task is to identify whether the writer of b

---

[3] The way different segments are handled depends on the model being used; they may e.g. be assigned different embeddings (Devlin et al., 2019) or separated by special tokens (Liu et al., 2019; Yang et al., 2019). For example, "a ‖ b" is given to BERT as the input "[CLS] a [SEP] b [SEP]".

Line	Examples	Method	Yelp	AG's	Yahoo	MNLI (m/mm)
1	$\lvert\mathcal{T}\rvert = 0$	unsupervised (avg)	33.8 ±9.6	69.5 ±7.2	44.0 ±9.1	39.1 ±4.3 / 39.8 ±5.1
2		unsupervised (max)	40.8 ±0.0	79.4 ±0.0	56.4 ±0.0	43.8 ±0.0 / 45.0 ±0.0
3		iPET	**56.7** ±0.2	**87.5** ±0.1	**70.7** ±0.1	**53.6** ±0.1 / **54.2** ±0.1
4	$\lvert\mathcal{T}\rvert = 10$	supervised	21.1 ±1.6	25.0 ±0.1	10.1 ±0.1	34.2 ±2.1 / 34.1 ±2.0
5		PET	52.9 ±0.1	87.5 ±0.0	63.8 ±0.2	41.8 ±0.1 / 41.5 ±0.2
6		iPET	**57.6** ±0.0	**89.3** ±0.1	**70.7** ±0.1	**43.2** ±0.0 / **45.7** ±0.1
7	$\lvert\mathcal{T}\rvert = 50$	supervised	44.8 ±2.7	82.1 ±2.5	52.5 ±3.1	45.6 ±1.8 / 47.6 ±2.4
8		PET	60.0 ±0.1	86.3 ±0.0	66.2 ±0.1	63.9 ±0.0 / 64.2 ±0.0
9		iPET	**60.7** ±0.1	**88.4** ±0.1	**69.7** ±0.0	**67.4** ±0.3 / **68.3** ±0.3
10	$\lvert\mathcal{T}\rvert = 100$	supervised	53.0 ±3.1	86.0 ±0.7	62.9 ±0.9	47.9 ±2.8 / 51.2 ±2.6
11		PET	61.9 ±0.0	88.3 ±0.1	69.2 ±0.0	74.7 ±0.3 / 75.9 ±0.4
12		iPET	**62.9** ±0.0	**89.6** ±0.1	**71.2** ±0.1	**78.4** ±0.7 / **78.6** ±0.5
13	$\lvert\mathcal{T}\rvert = 1000$	supervised	63.0 ±0.5	**86.9** ±0.4	70.5 ±0.3	73.1 ±0.2 / 74.8 ±0.3
14		PET	**64.8** ±0.1	**86.9** ±0.2	**72.7** ±0.0	**85.3** ±0.2 / **85.5** ±0.4

Table 1: Average accuracy and standard deviation for RoBERTa (large) on Yelp, AG's News, Yahoo and MNLI (m:matched/mm:mismatched) for five training set sizes $|\mathcal{T}|$.

supports the subject of the question (0) or not (1). We use two simple patterns

$$P_1(\mathbf{x}) = \text{“}a\text{”} \parallel \text{\_\_\_\_. “}b\text{”} \qquad P_2(\mathbf{x}) = a \parallel \text{\_\_\_\_. } b$$

and define an English verbalizer v_{En} mapping 0 to "Yes" and 1 to "No" as well as a French (German) verbalizer v_{Fr} (v_{De}), replacing "Yes" and "No" with "Oui" and "Non" ("Ja" and "Nein"). We do not define an Italian verbalizer because x-stance does not contain any Italian training examples.

Ex.	Method	Yelp	AG's	Yahoo	MNLI
$\lvert\mathcal{T}\rvert = 10$	UDA	27.3	72.6	36.7	34.7
	MixText	20.4	81.1	20.6	32.9
	PET	48.8	84.1	59.0	39.5
	iPET	**52.9**	**87.5**	**67.0**	**42.1**
$\lvert\mathcal{T}\rvert = 50$	UDA	46.6	83.0	60.2	40.8
	MixText	31.3	84.8	61.5	34.8
	PET	55.3	86.4	63.3	55.1
	iPET	**56.7**	**87.3**	**66.4**	**56.3**

Table 2: Comparison of PET with two state-of-the-art semi-supervised methods using RoBERTa (base)

4.2 Results

English Datasets Table 1 shows results for English text classification and language understanding tasks; we report mean accuracy and standard deviation for three training runs. Lines 1–2 (L1–L2) show unsupervised performance, i.e., individual PVPs without *any* training (similar to Radford et al., 2018; Puri and Catanzaro, 2019); we give both average results across all PVPs (avg) and results for the PVP that works best on the test set (max). The large difference between both rows highlights the importance of coping with the fact that without looking at the test set, we have no means of evaluating which PVPs perform well. Zero-shot iPET clearly outperforms the unsupervised baselines for all datasets (L3 vs L1); on AG's News, it even performs better than standard supervised training with 1000 examples (L3 vs L13). With just 10 training examples, standard supervised learning does not perform above chance (L4). In contrast, PET (L5) performs much better than the fully unsupervised baselines (L1–L2); training multiple generations using iPET (L6) gives consistent improvements. As we increase the training set size, the performance gains of PET and iPET become smaller, but for both 50 and 100 examples, PET continues to considerably outperform standard supervised training (L8 vs L7, L11 vs L10) with iPET (L9, L12) still giving consistent improvements. For $|\mathcal{T}| = 1000$, PET has no advantage on AG's but still improves accuracy for all other tasks (L14 vs L13).[4]

Comparison with SotA We compare PET to UDA (Xie et al., 2020) and MixText (Chen et al., 2020), two state-of-the-art methods for semi-supervised learning in NLP that rely on data augmentation. Whereas PET requires that a task can be expressed using patterns and that such patterns be found, UDA and MixText both use backtranslation (Sennrich et al., 2016) and thus require thousands of labeled examples for training a machine translation model. We use RoBERTa (base) for our comparison as MixText is specifically tailored towards

---

[4] One of the three supervised MNLI runs for $|\mathcal{T}| = 1000$ underfitted the training data and performed extremely poorly. This run is excluded in the reported score (73.1/74.8).

Examples	Method	De	Fr	It
$\lvert\mathcal{T}\rvert = 1000$	supervised	43.3	49.5	41.0
	PET	**66.4**	**68.7**	**64.7**
$\lvert\mathcal{T}\rvert = 2000$	supervised	57.4	62.1	52.8
	PET	**69.5**	**71.7**	**67.3**
$\lvert\mathcal{T}\rvert = 4000$	supervised	63.2	66.7	58.7
	PET	**71.7**	**74.0**	**69.5**
$\mathcal{T}_{\text{De}}$, $\mathcal{T}_{\text{Fr}}$	supervised	76.6	76.0	71.0
	PET	**77.9**	**79.0**	**73.6**
$\mathcal{T}_{\text{De}} + \mathcal{T}_{\text{Fr}}$	sup. (*)	76.8	76.7	70.2
	supervised	77.6	79.1	75.9
	PET	**78.8**	**80.6**	**77.2**

Table 3: Results on x-stance intra-target for XLM-R (base) trained on subsets of $\mathcal{T}_{\text{De}}$ and $\mathcal{T}_{\text{Fr}}$ and for joint training on all data ($\mathcal{T}_{\text{De}} + \mathcal{T}_{\text{Fr}}$). (*): Best results for mBERT reported in Vamvas and Sennrich (2020).

a 12-layer Transformer (Vaswani et al., 2017). Both Xie et al. (2020) and Chen et al. (2020) use large development sets to optimize the number of training steps. We instead try several values for both approaches directly on the test set and only report the *best* results obtained. Despite this, Table 2 shows that PET and iPET substantially outperform both methods across all tasks, clearly demonstrating the benefit of incorporating human knowledge in the form of PVPs.

X-Stance We evaluate PET on x-stance to investigate (i) whether it works for languages other than English and (ii) whether it also brings improvements when training sets have medium size. In contrast to Vamvas and Sennrich (2020), we do not perform any hyperparameter optimization on dev and use a shorter maximum sequence length (256 vs 512) to speed up training and evaluation.

To investigate whether PET brings benefits even when numerous examples are available, we consider training set sizes of 1000, 2000, and 4000; for each of these configurations, we separately finetune French and German models to allow for a more straightforward downsampling of the training data. Additionally, we train models on the entire French ($\lvert\mathcal{T}_{\text{Fr}}\rvert = 11\,790$) and German ($\lvert\mathcal{T}_{\text{De}}\rvert = 33\,850$) training sets. In this case we do not have any additional unlabeled data, so we simply set $\mathcal{D} = \mathcal{T}$. For the French models, we use v_{En} and v_{Fr} as verbalizers and for German v_{En} and v_{De} (Section 4.1). Finally, we also investigate the performance of a model trained jointly on French and German data ($\lvert\mathcal{T}_{\text{Fr}} + \mathcal{T}_{\text{De}}\rvert = 45\,640$) using v_{En}, v_{Fr} and v_{De}.

Method	Yelp	AG's	Yahoo	MNLI
min	39.6	82.1	50.2	36.4
max	52.4	85.0	63.6	40.2
PET (no distillation)	51.7	87.0	62.8	40.6
PET uniform	52.7	87.3	**63.8**	**42.0**
PET weighted	**52.9**	**87.5**	**63.8**	41.8

Table 4: Minimum (min) and maximum (max) accuracy of models based on individual PVPs as well as PET with and without knowledge distillation ($\lvert\mathcal{T}\rvert = 10$).

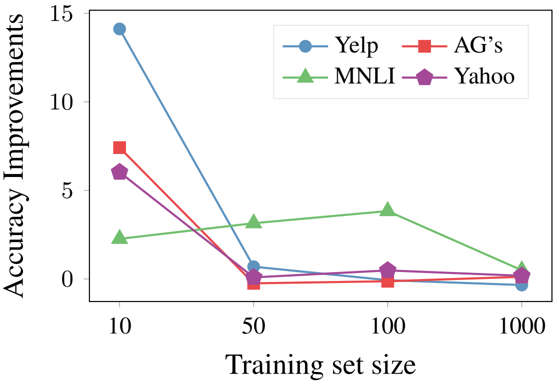


Figure 3: Accuracy improvements for PET due to adding L_{MLM} during training

Results are shown in Table 3; following Vamvas and Sennrich (2020), we report the macro-average of the F1 scores for labels 0 and 1, averaged over three runs. For Italian (column "It"), we report the average zero-shot cross-lingual performance of German and French models as there are no Italian training examples. Our results show that PET brings huge improvements across all languages even when training on much more than a thousand examples; it also considerably improves zero-shot cross-lingual performance.

5 Analysis

Combining PVPs We first investigate whether PET is able to cope with situations were some PVPs perform much worse than others. For $\lvert\mathcal{T}\rvert = 10$, Table 4 compares the performance of PET to that of the best and worst performing patterns after finetuning; we also include results obtained using the ensemble of PET models corresponding to individual PVPs without knowledge distillation. Even after finetuning, the gap between the best and worst pattern is large, especially for Yelp. However, PET is not only able to compensate for this, but even improves accuracies over using only the best-performing pattern across all tasks. Distillation brings consistent improvements over the ensemble; additionally, it significantly reduces the size of the

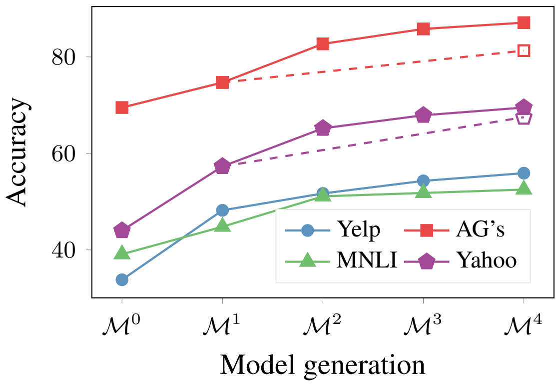


Figure 4: Average accuracy for each generation of models with iPET in a zero-shot setting. Accuracy on AG's News and Yahoo when skipping generation 2 and 3 is indicated through dashed lines.

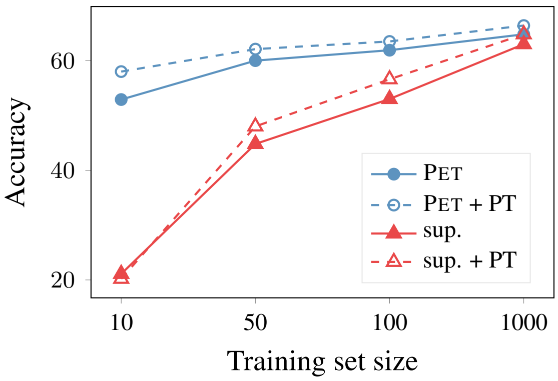


Figure 5: Accuracy of supervised learning (sup.) and PET both with and without pretraining (PT) on Yelp

final classifier. We find no clear difference between the uniform and weighted variants of PET.

Auxiliary Language Modeling We analyze the influence of the auxiliary language modeling task on PET's performance. Figure 3 shows performance improvements from adding the language modeling task for four training set sizes. We see that the auxiliary task is extremely valuable when training on just 10 examples. With more data, it becomes less important, sometimes even leading to worse performance. Only for MNLI, we find language modeling to consistently help.

Iterative PET To check whether iPET is able to improve models over multiple generations, Figure 4 shows the average performance of all generations of models in a zero-shot setting. Each additional iteration does indeed further improve the ensemble's performance. We did not investigate whether continuing this process for even more iterations gives further improvements.

Another natural question is whether similar results can be obtained with fewer iterations by increasing the training set size more aggressively. To answer this question, we skip generations 2 and 3 for AG's News and Yahoo and for both tasks directly let ensemble $\mathcal{M}^1$ annotate $10 \cdot 5^4$ examples for $\mathcal{M}^4$. As indicated in Figure 4 through dashed lines, this clearly leads to worse performance, highlighting the importance of only gradually increasing the training set size. We surmise that this is the case because annotating too many examples too early leads to a large percentage of mislabeled training examples.

In-Domain Pretraining Unlike our supervised baseline, PET makes use of the additional unlabeled dataset $\mathcal{D}$. Thus, at least some of PET's performance gains over the supervised baseline may arise from this additional in-domain data.

To test this hypothesis, we simply further pretrain RoBERTa on in-domain data, a common technique for improving text classification accuracy (e.g., Howard and Ruder, 2018; Sun et al., 2019). As language model pretraining is expensive in terms of GPU usage, we do so only for the Yelp dataset. Figure 5 shows results of supervised learning and PET both with and without this in-domain pretraining. While pretraining does indeed improve accuracy for supervised training, the supervised model still clearly performs worse than PET, showing that the success of our method is not simply due to the usage of additional unlabeled data. Interestingly, in-domain pretraining is also helpful for PET, indicating that PET leverages unlabeled data in a way that is clearly different from standard masked language model pretraining.

6 Conclusion

We have shown that providing task descriptions to pretrained language models can be combined with standard supervised training. Our proposed method, PET, consists of defining pairs of cloze question patterns and verbalizers that help leverage the knowledge contained within pretrained language models for downstream tasks. We finetune models for all pattern-verbalizer pairs and use them to create large annotated datasets on which standard classifiers can be trained. When the initial amount of training data is limited, PET gives large improvements over standard supervised training and strong semi-supervised approaches.

Acknowledgments

This work was funded by the European Research Council (ERC #740516). We would like to thank the anonymous reviewers for their helpful comments.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, DL '00, page 85–94, New York, NY, USA. Association for Computing Machinery.

David S. Batista, Bruno Martins, and Mário J. Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 499–504, Lisbon, Portugal. Association for Computational Linguistics.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from BERT. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Sergey Brin. 1999. Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases*, pages 172–183, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157, Online. Association for Computational Linguistics.

Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2089–2095, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

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

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Computing Research Repository*, arXiv:1503.02531.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Zhongqiang Huang and Mary Harper. 2009. Self-training PCFG grammars with latent annotations across languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 832–841, Singapore. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *Computing Research Repository*, arXiv:1806.08730.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

Juri Opitz. 2019. Argumentative relation classification as plausibility ranking. In *Preliminary proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 193–202, Erlangen, Germany. German Society for Computational Linguistics & Language Technology.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Raul Puri and Bryan Catanzaro. 2019. Zero-shot text classification with generative language models. *Computing Research Repository*, arXiv:1912.10165.

Kun Qian and Zhou Yu. 2019. Domain adaptive dialog generation via meta learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2639–2649, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report.

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

Roi Reichart and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 616–623, Prague, Czech Republic. Association for Computational Linguistics.

Bernardino Romera-Paredes and Philip Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *International Conference on Machine Learning*, pages 2152–2161.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Shashank Srivastava, Igor Labutov, and Tom Mitchell. 2018. Zero-shot learning of classifiers from natural language quantification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 306–316, Melbourne, Australia. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? In *Chinese Computational Linguistics*, pages 194–206, Cham. Springer International Publishing.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. oLMpics – on what language model pre-training captures. *Computing Research Repository*, arXiv:1912.13283.

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning. *Computing Research Repository*, arXiv:1806.02847.

Jannis Vamvas and Rico Sennrich. 2020. X-stance: A multilingual multi-target dataset for stance detection. *Computing Research Repository*, arXiv:2003.08385.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Sappadla Prateek Veeranna, Jinseok Nam, Eneldo Loza Mencıa, and Johannes Fürnkranz. 2016. Using semantic similarity for multi-label zero-shot classification of text documents. In *Proceeding of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges, Belgium: Elsevier*, pages 423–428.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? And why? A pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2020. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763. Curran Associates, Inc.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Zhiquan Ye, Yuxia Geng, Jiaoyan Chen, Jingmin Chen, Xiaoxiao Xu, SuHang Zheng, Feng Wang, Jun Zhang, and Huajun Chen. 2020. Zero-shot text classification via reinforced self-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3014–3024, Online. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1206–1215, New Orleans, Louisiana. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

A Implementation

Our implementation of PET and iPET is based on the Transformers library (Wolf et al., 2020) and PyTorch (Paszke et al., 2017).

B Training Details

Except for the in-domain pretraining experiment described in Section 5, all of our experiments were conducted using a single GPU with 11GB RAM (NVIDIA GeForce GTX 1080 Ti).

B.1 Hyperparameter Choices

Relevant training hyperparameters for both individual PET models and the final classifier C as well as our supervised baseline are listed in Table 5. All hyperparameters were selected based on the following considerations and experiments:

Batch size / maximum length Both batch size and maximum sequence length (or block size) are chosen so that one batch fits into 11GB of GPU memory. As Devlin et al. (2019) and Liu et al. (2019) use larger batch sizes of 16–32, we accumulate gradients for 4 steps to obtain an effective batch size of 16.

Learning rate We found a learning rate of $5\mathrm{e}{-5}$ (as used by Devlin et al. (2019)) to often result in unstable training for regular supervised learning with no accuracy improvements on the training set. We therefore use a lower learning rate of $1\mathrm{e}{-5}$, similar to Liu et al. (2019). Experiments with various learning rates can be found in Appendix D.

Training steps As the number of training epochs recommended by Liu et al. (2019) in a data-rich scenario is in the range 2–10, we perform supervised training for 250 training steps, corresponding to 4 epochs when training on 1000 examples. For individual PET models, we subdivide each batch into one labeled example from $\mathcal{T}$ to compute L_{CE} and three unlabeled examples from $\mathcal{D}$ to compute L_{MLM}. Accordingly, we multiply the number of total training steps by 4 (i.e., 1000), so that the number of times each labeled example is seen remains constant ($16 \cdot 250 = 4 \cdot 1000$). For the final PET classifier, we train for 5000 steps due to the increased training set size (depending on the task, the unlabeled set $\mathcal{D}$ contains at least 20 000 examples). Deviating from the above, we always perform training for 3 epochs on x-stance to match the setup of Vamvas and Sennrich (2020) more closely. The effect of varying the number of training steps is further investigated in Appendix D.

Temperature We choose a temperature of 2 when training the final classifier following Hinton et al. (2015).

Auxiliary language modeling To find a suitable value of α for combining language modeling loss and cross-entropy loss, we first observed that in the early stages of training, the former is a few orders of magnitude higher than the latter for all tasks considered. We thus selected a range $\{1\mathrm{e}{-3}, 1\mathrm{e}{-4}, 1\mathrm{e}{-5}\}$ of reasonable choices for α and performed preliminary experiments on Yelp with 100 training examples to find the best value among these candidates. To this end, we split the training examples into a training set and a dev set using both a 90/10 split and a 50/50 split and took the value of α that maximizes average dev set accuracy. We adopt this value for all other tasks and training set sizes without further optimization.

Models per ensemble As we always train three models per pattern, for both iPET and training the final classifier C, the ensemble $\mathcal{M}$ (or $\mathcal{M}^0$) for n PVPs contains $3n$ models. This ensures consistency as randomly choosing any of the three models for each PVP would result in high variance. In preliminary experiments, we found this to have only little impact on the final model's performance.

iPET dataset size For iPET, we quintuple the number of training examples after each iteration ($d = 5$) so that only a small number of generations is required to reach a sufficient amount of labeled data. We did not choose a higher value because we presume that this may cause training sets for early generations to contain a prohibitively large amount of mislabeled data.

iPET dataset creation We create training sets for the next generation in iPET using 25% of the models in the current generation ($\lambda = 0.25$) because we want the training sets for all models to be diverse while at the same time, a single model should not have too much influence.

Others For all other hyperparameters listed in Table 5, we took the default settings of the Transformers library (Wolf et al., 2020).

B.2 Number of parameters

As PET does not require any additional learnable parameters, the number of parameters for both PET and iPET is identical to the number of parameters in the underlying language model: 355M for RoBERTa (large) and 270M for XLM-R (base).

B.3 Average runtime

Training a single PET classifier for 250 steps on one GPU took approximately 30 minutes; training for 1000 steps with auxiliary language modeling took 60 minutes. Depending on the task, labeling examples from $\mathcal{D}$ took 15–30 minutes per model. Training the final classifier C for 5000 steps on the soft-labeled dataset $\mathcal{T}_C$ took 2 hours on average.

B.4 Comparison with SotA

For comparing PET to UDA (Xie et al., 2020) and MixText (Chen et al., 2020), we reduce the number of unlabeled examples by half to speed up the required backtranslation step. We use the backtranslation script provided by Chen et al. (2020) with their recommended hyperparameter values and use both Russian and German as intermediate languages.

For MixText, we use the original implementation[5] and the default set of hyperparameters. Specifically, each batch consists of 4 labeled and 8 unlabeled examples, we use layers 7, 9 and 12 for mixing, we set $T = 5$, $\alpha = 16$, and use a learning rate of $5 \cdot 10^{-6}$ for RoBERTa and $5 \cdot 10^{-4}$ for the final classification layer. We optimize the number of training steps for each task and dataset size in the range $\{1000, 2000, 3000, 4000, 5000\}$.

For UDA, we use a PyTorch-based reimplementation[6]. We use the same batch size as for MixText and the hyperparameter values recommended by Xie et al. (2020); we use an exponential schedule for training signal annealing and a learning rate of $2 \cdot 10^{-5}$. We optimize the number of training steps for each task and dataset size in the range $\{500, 1000, 1500, \ldots, 10000\}$.

B.5 In-Domain Pretraining

For in-domain pretraining experiments described in Section 5, we use the language model finetuning script of the Transformers library (Wolf et al., 2020); all hyperparameters are listed in the last column of Table 5. Pretraining was performed on a total of 3 NVIDIA GeForce GTX 1080 Ti GPUs.

C Dataset Details

For each task and number of examples t, we create the training set $\mathcal{T}$ by collecting the first $t/|\mathcal{L}|$ examples per label from the original training set, where $|\mathcal{L}|$ is the number of labels for the task. Similarly, we construct the set $\mathcal{D}$ of unlabeled examples by selecting 10 000 examples per label and removing all labels. For evaluation, we use the official test set for all tasks except MNLI, for which we report results on the dev set; this is due to the limit of 2 submissions per 14 hours for the official MNLI test set. An overview of the number of test examples and links to downloadable versions of all used datasets can be found in Table 6.

Preprocessing In some of the datasets used, newlines are indicated through the character sequence "\n". As the vocabularies of RoBERTa and XLM-R do not feature a newline, we replace this sequence with a single space. We do not perform any other preprocessing, except shortening all examples to the maximum sequence length of 256 tokens. This is done using the *longest first* strategy implemented in the Transformers library. For PET, all input sequences are truncated *before* applying patterns.

Evaluation metrics For Yelp, AG's News, Yahoo and MNLI, we use accuracy. For x-stance, we report macro-average of F1 scores using the evaluation script of Vamvas and Sennrich (2020).

D Hyperparameter Importance

To analyze the importance of hyperparameter choices for PET's performance gains over supervised learning, we look at the influence of both the learning rate (LR) and the number of training steps on their test set accuracies.

We try values of $\{1\text{e}{-5}, 2\text{e}{-5}, 5\text{e}{-5}\}$ for the learning rate and $\{50, 100, 250, 500, 1000\}$ for the number of training steps. As this results in 30 different configurations for just one task and training set size, we only perform this analysis on Yelp with 100 examples, for which results can be seen in Figure 6. For supervised learning, the configuration used throughout the paper (LR $= 1\text{e}{-5}$, 250 steps) turns out to perform best whereas for PET, training for fewer steps consistently performs even better. Importantly, PET clearly outperforms regular supervised training regardless of the chosen learning rate and number of training steps.

---

[5] `https://github.com/GT-SALT/MixText`

[6] `https://github.com/SanghunYun/UDA_pytorch`

Parameter	PET −LM	PET (En/Xs)	C (En/Xs)	sup. (En/Xs)	In-Dom. PT
`adam_epsilon`	1e-8	1e-8	1e-8	1e-8	1e-8
* `alpha`	–	1e-4	–	–	–
`block_size`	–	–	–	–	256
`gradient_accumulation_steps`	4	4	4	4	2
`learning_rate`	1e-5	1e-5	1e-5	1e-5	5e-5
`max_grad_norm`	1.0	1.0	1.0	1.0	1.0
`max_seq_length`	256	256	256	256	–
`max_steps`	250	1000 / –	5000 / –	250 / –	50000
`mlm_probability`	–	0.15	–	–	0.15
`num_train_epochs`	–	– / 3	– / 3	– / 3	–
`per_gpu_train_batch_size`	4	1	4	4	2
* `per_gpu_helper_batch_size`	–	3	–	–	–
* `temperature`	–	–	2.0	–	–
`weight_decay`	0.01	0.01	0.01	0.01	0.0

Table 5: Hyperparameters for training individual PET models without auxiliary language modeling (PET−LM) and with language modeling (PET), the final PET classifier (C), regular supervised training (sup.) and in-domain pretraining (In-Dom. PT). Whenever different values are used for the English datasets (En) and x-stance (Xs), both values are given separated by a slash. (*): PET-specific hyperparameters

Dataset	**Link**	**Test Examples**
AG's News	`http://goo.gl/JyCnZq`	7600
MNLI (m / mm)	`https://cims.nyu.edu/~sbowman/multinli/`	10000 / 10000
X-Stance (De / Fr / It)	`https://github.com/ZurichNLP/xstance`	3479 / 1284 / 1173
Yahoo! Answers	`http://goo.gl/JyCnZq`	60000
Yelp Review Full	`http://goo.gl/JyCnZq`	50000

Table 6: Download links and number of test examples for all datasets

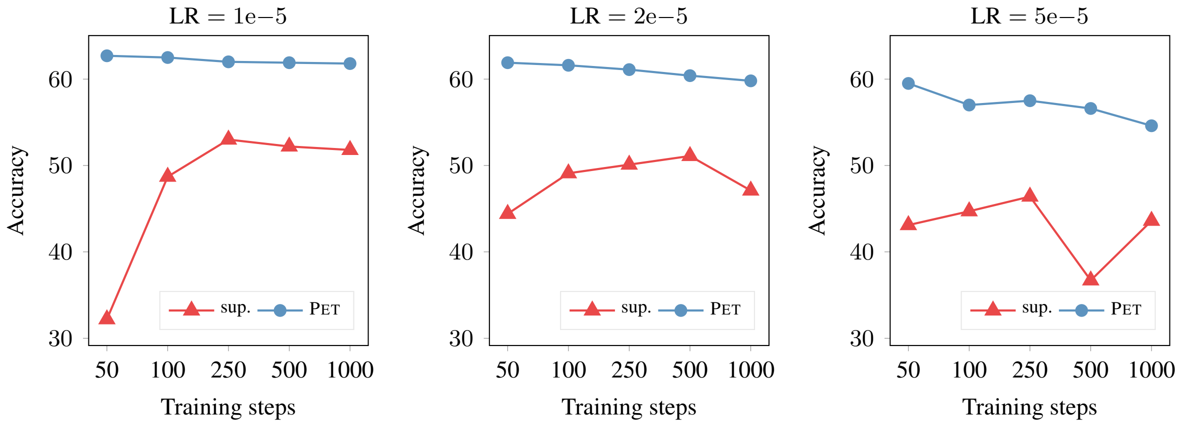


Figure 6: Performance of supervised learning and PET (weighted, without auxiliary language modeling) for various learning rates and training steps on Yelp with 100 training examples

E Automatic Verbalizer Search

Given a set of patterns $P_1, \ldots, P_n$, manually finding a verbalization $v(l)$ for each $l \in \mathcal{L}$ that represents the meaning of l well and corresponds to a single token in V can be difficult. We therefore devise *automatic verbalizer search* (AVS), a procedure that automatically finds suitable verbalizers given a training set $\mathcal{T}$ and a language model M.

Assuming we already have a PVP $\mathbf{p} = (P, v)$, we can easily check whether some token $t \in V$ is a good verbalization of $l \in \mathcal{L}$. To this end, we define $\mathbf{p}[l \leftarrow t] = (P, v')$, where v' is identical to v, except that $v'(l) = t$. Intuitively, if t represents l well, then $q_{\mathbf{p}[l \leftarrow t]}(l \mid \mathbf{x})$ (i.e., the probability M assigns to t given $P(\mathbf{x})$) should be high only for those examples $(\mathbf{x}, y) \in \mathcal{T}$ where $y = l$. We thus define the score of t for l given $\mathbf{p}$ as

$$s_l(t \mid \mathbf{p}) = \frac{1}{|\mathcal{T}_l|} \cdot \sum_{(\mathbf{x},y)\in\mathcal{T}_l} q_{\mathbf{p}[l\leftarrow t]}(l \mid \mathbf{x}) - \frac{1}{|\mathcal{T} \setminus \mathcal{T}_l|} \cdot \sum_{(\mathbf{x},y)\in\mathcal{T}\setminus\mathcal{T}_l} q_{\mathbf{p}[l\leftarrow t]}(l \mid \mathbf{x})$$

where $\mathcal{T}_l = \{(\mathbf{x}, y) \in \mathcal{T} : y = l\}$ is the set of all training examples with label l. While this allows us to easily compute the best verbalization for l as

$$\hat{t} = \arg\max_{t\in V} s_l(t \mid \mathbf{p}) \,,$$

it requires us to already know verbalizations $v(l')$ for all other labels l'.

AVS solves this problem as follows: We first assign random verbalizations to all labels and then repeatedly recompute the best verbalization for each label. As we do not want the resulting verbalizer to depend strongly on the initial random assignment, we simply consider multiple such assignments. Specifically, we define an initial probability distribution ρ_0 where for all $t \in V, l \in \mathcal{L}$, $\rho_0(t \mid l) = 1/|V|$ is the probability of choosing t as verbalization for l. For each $l \in \mathcal{L}$, we then sample k verbalizers $v_1, \ldots, v_k$ using ρ_0 to compute

$$s_l^k(t) = \frac{1}{n \cdot k} \sum_{i=1}^{n} \sum_{j=1}^{k} s_l(t \mid (P_i, v_j))$$

for all $t \in V$.[7] These scores enable us to define a probability distribution ρ_1 that more closely reflects a word's suitability as a verbalizer for a given label:

$$\rho_1(t \mid l) = \frac{1}{Z} \max(s_l^k(t), \epsilon)$$

where $Z = \sum_{t'\in V} \max(s_l^k(t'), \epsilon)$ and $\epsilon \geq 0$ ensures that ρ_1 is a proper probability distribution. We repeat this process to obtain a sequence of probability distributions $\rho_1, \ldots, \rho_{i_\text{max}}$. Finally, we choose the $m \in \mathbb{N}$ most likely tokens according to $\rho_{i_\text{max}}(t \mid l)$ as verbalizers for each l. During training and inference, we compute the unnormalized score $s_\mathbf{p}(y \mid \mathbf{x})$ for each label by averaging over its m verbalizers.

We analyze the performance of AVS for all tasks with $|\mathcal{T}| = 50$ training examples and set $k = 250$, $\epsilon = 10^{-3}$, $i_\text{max} = 5$ and $m = 10$.[8] To speed up the search, we additionally restrict our search space to tokens $t \in V$ that contain at least two alphabetic characters. Of these tokens, we only keep the 10 000 most frequent ones in $\mathcal{D}$.

Results are shown in Table 7. As can be seen, carefully handcrafted verbalizers perform much better than AVS; however, PET with AVS still considerably outperforms regular supervised training while eliminating the challenge of manually finding suitable verbalizers. Table 8 shows the most probable verbalizers found using AVS for the Yelp dataset. While most verbalizers for this dataset intuitively make sense, we found AVS to struggle with finding good verbalizers for three out of ten labels in the Yahoo dataset and for all MNLI labels.

	Yelp	**AG's**	**Yahoo**	**MNLI**
supervised	44.8	82.1	52.5	45.6
PET	**60.0**	**86.3**	**66.2**	**63.9**
PET + AVS	55.2	85.0	58.2	52.6

Table 7: Results for supervised learning, PET and PET with AVS (PET + AVS) after training on 50 examples

y	**Top Verbalizers**
1	worthless, BAD, useless, appalling
2	worse, slow, frustrating, annoying
3	edible, mixed, cute, tasty, Okay
4	marvelous, loved, love, divine, fab
5	golden, magical, marvelous, perfection

Table 8: Most probable verbalizers according to AVS for Yelp with 50 training examples

---

[7] Note that the score $s_l^k(t)$ jointly considers all patterns; in preliminary experiments, we found this to result in more robust verbalizers.

[8] We tried values of k and i_max in $\{250, 500, 1000\}$ and $\{5, 10, 20\}$, respectively, but found the resulting verbalizers to be almost identical.

CD^{2}CR: Co-reference Resolution Across Documents and Domains

James Ravenscroft[1,2,3] **Arie Cattan**[4] **Amanda Clare**[5] **Ido Dagan**[4] **Maria Liakata**[1,2]
[1]: University of Warwick, United Kingdom
[2]: Alan Turing Institute, United Kingdom
[3]: Filament AI, United Kingdom
[4]: Bar-Ilan University, Israel
[5]: Aberystwyth University, United Kingdom
{j.ravenscroft|m.liakata}@warwick.ac.uk
{arie.cattan|ido.k.dagan}@gmail.com
afc@aber.ac.uk

Abstract

Cross-document co-reference resolution (CDCR) is the task of identifying and linking mentions to entities and concepts across many text documents. Current state-of-the-art models for this task assume that all documents are of the same type (e.g. news articles) or fall under the same theme. However, it is also desirable to perform CDCR across different domains (type or theme). A particular use case we focus on in this paper is the resolution of entities mentioned across scientific work and newspaper articles that discuss them. Identifying the same entities and corresponding concepts in both scientific articles and news can help scientists understand how their work is represented in mainstream media. We propose a new task and English language dataset for cross-document cross-domain co-reference resolution (CD^{2}CR). The task aims to identify links between entities across heterogeneous document types. We show that in this cross-domain, cross-document setting, existing CDCR models do not perform well and we provide a baseline model that outperforms current state-of-the-art CDCR models on CD^{2}CR. Our data set, annotation tool and guidelines as well as our model for cross-document cross-domain co-reference are all supplied as open access open source resources.

1 Introduction

Cross-document co-reference resolution (CDCR) is the task of recognising when multiple documents mention and refer to the same real-world entity or concept. CDCR is a useful NLP process that has many downstream applications. For example, CDCR carried out on separate news articles that refer to the same politician can facilitate inter-document sentence alignment required for stance detection and natural language inference models. Furthermore, CDCR can improve information retrieval and multi-document summarisation by grouping documents based on the entities that are mentioned within them.

Recent CDCR work (Dutta and Weikum, 2015; Barhom et al., 2019; Cattan et al., 2020) has primarily focused on resolution of entity mentions across news articles. Despite differences in tone and political alignment, most news articles are relatively similar in terms of grammatical and lexical structure. Work based on modern transformer networks such as BERT (Devlin et al., 2019) and ElMo (Peters et al., 2018) have been pre-trained on large news corpora and are therefore well suited to news-based CDCR (Barhom et al., 2019).

However, there are cases where CDCR across documents from different domains (i.e. that differ much more significantly in style, vocabulary and structure) is useful. One such example is the task of resolving references to concepts across scientific papers and related news articles. This can help scientists understand how their work is being presented to the public by mainstream media or facilitate fact checking of journalists' work (Wadden et al., 2020). A chatbot or recommender that is able to resolve references to current affairs in both news articles and user input could be more effective at suggesting topics that interest the user. Finally, it may be helpful for e-commerce companies to know when product reviews gathered from third party websites refer to one of their own listings. The work we present here focuses on the first cross-document, cross-domain co-reference-resolution (CD^{2}CR) use case, namely co-reference resolution between news articles and scientific papers.

The objective of CD^{2}CR is to identify co-referring entities from documents belonging to different domains. In this case co-reference resolution is made more challenging by the differences in language use (lexical but also syntactic) across the dif-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 270–280
April 19 - 23, 2021.

ferent domains. Specifically, authors of scientific papers aim to communicate novel scientific work in an accurate and unambiguous way by using precise scientific terminology. Whilst scientific journalists also aim to accurately communicate novel scientific work, their work is primarily funded by newspaper sales and thus they also aim to captivate as large an audience as possible. Therefore journalists tend to use simplified vocabulary and structure, creative and unexpected writing style, slang, simile, metaphor and exaggeration to make their work accessible, informative and entertaining in order to maximise readership (Louis and Nenkova, 2013).

Success at the CD^{2}CR task in this setting is dependent on context sensitive understanding of how the accessible but imprecise writing of journalists maps on to precise terminology used in scientific writing. For example, a recent study has found that "convalescent plasma derived from donors who have recovered from COVID-19 can be used to treat patients sick with the disease" [1]. A news article[2] discussing this work says that "...blood from recovered Covid-19 patients in the hope that transfusions...[can help to treat severely ill patients]" . In this example the task is to link 'blood' to 'convalescent plasma' and 'recovered Covid-19 patients' to 'donors'. These cross-document, cross-domain co-reference chains can be used as contextual anchors for downstream analysis of the two document settings via tasks such as natural language inference, stance detection and frame analysis.

The contributions in this paper are the following:

- A novel task setting for CDCR that is more challenging than those that already exist due to linguistic variation between different domains and document types (we call this CD^{2}CR).
- An open source English language CD^{2}CR dataset with 7602 co-reference pair annotations over 528 documents and detailed 11 page annotation guidelines (section 3.1).
- A novel annotation tool to support ongoing data collection and annotation for CD^{2}CR including a novel sampling mechanism for calculating inter-annotator agreement (Section 3.4).
- A series of experiments on our dataset using different baseline models and an in-depth capability-based evaluation of the best-performing baseline (Section 5)

2 Related Work

2.1 Co-reference Resolution

Intra-document co-reference resolution is a well understood task with mature training data sets (Weischedel et al., 2013) and academic tasks (Recasens et al., 2010). The current state of the art model by Joshi et al. (2020) is based on Lee et al. (2017, 2018) and uses a modern BERT-based (Devlin et al., 2019) architecture. Comparatively, CDCR, which involves co-reference resolution across multiple documents, has received less attention in recent years (Bagga and Baldwin, 1998; Rao et al., 2010; Dutta and Weikum, 2015; Barhom et al., 2019). Cattan et al. (2020) jointly learns both entity and event co-reference tasks, achieving current state of the art performance for CDCR, and as such provides a strong baseline for experiments in CD^{2}CR. Both Cattan et al. (2020) and Barhom et al. (2019) models are trained and evaluated using the ECB+ corpus (Cybulska and Vossen, 2014) which contains news articles annotated with both entity and event mentions.

2.2 Entity Linking

Entity Linking (EL) focuses on alignment of mentions in documents to resources in an external knowledge resource (Ji et al., 2010) such as SNOMED CT[3] or DBPedia[4]. EL is challenging due to the large number of pairwise comparisons between document mentions and knowledge resource entities that may need to be carried out. Raiman and Raiman (2018) provide state of the art performance by building on Ling et al. (2015)'s work in which an entity type system is used to limit the number of required pairwise comparisons to related types. Yin et al. (2019) achieved comparable results using a graph-traversal method to similarly constrain the problem space to candidates within a similar graph neighbourhood. EL can be considered a narrow sub-task of CDCR since it cannot resolve novel and rare entities or pronouns (Shen et al., 2015). Moreover EL's dependency on expensive-to-maintain external knowledge graphs is also problematic when limited human expertise is available.

---

[1] DOI: 10.1101/2020.03.16.20036145

[2] `https://tinyurl.com/ycnq9xg7`

[3] `https://tinyurl.com/yy7g4ttz`

[4] `https://wiki.dbpedia.org/`

Given these limitations, EL is inappropriate within our task setting, hence our CDCR-based approach.

2.3 Semantic Specialisation

Like earlier static vector language models, contextual language models such as BERT (Devlin et al., 2019) and ElMo (Peters et al., 2018) use distributional knowledge (Harris, 1954) inherent in large text corpora to learn context-aware word embeddings that can be used for downstream NLP tasks. However, these models do not learn about formal lexical constraints, often conflating different types of semantic relatedness (Ponti et al., 2018; Lauscher et al., 2020). This is a weakness of all distributional language models that is particularly problematic in the context of CD^2CR for entity mentions that are related but not co-referent (e.g. “Mars” and “Jupiter”) as shown in section 5.

A number of solutions have been proposed for adding lexical knowledge to static word embeddings (Yu and Dredze, 2014; Wieting et al., 2015; Ponti et al., 2018) but contextual language models have received comparatively less attention. Lauscher et al (2020) propose adding a lexical relation classification step to BERT’s language model pre-training phase to allow the model to integrate both lexical and distributional knowledge. Their model, LIBERT, has been shown to facilitate statistically-significant performance boosts on a variety of downstream NLP tasks.

3 Dataset creation

Our dataset is composed of pairs of news articles and scientific papers gathered automatically (Section 3.1). Our annotation process begins by obtaining summaries of the news and science document pairs (extractive news summaries and scientific abstracts, respectively) (Section 3.2). Candidate co-reference pairs from each summary-abstract pair are identified and scored automatically. (Section 3.3). Candidate co-reference pairs are then presented to human annotators via a bespoke annotation interface for scoring (Section 3.4). Annotation quality is measured on an ongoing basis as new candidates are added to the system (Section 3.5).

3.1 Data Collection

We have developed a novel data set that allows us to train and evaluate a CD^2CR model. The corpus is approximately 50% the size of the ECB+ corpus (918 documents) (Cybulska and Vossen, 2014) and is split into training, development and test sets (statistics for each subset are provided in Table 1). Each pair of documents consists of a scientific paper and a newspaper article that discusses the scientific work. In order to detect pairs of documents, we follow the approach of (Ravenscroft et al., 2018), using approximate matching of author name and affiliation metadata, date of publishing and exact DOI matching where available to connect news articles to scientific publications.

Subset	Documents	Mentions	Clusters
Train	300	4,604	426
Dev	142	1,821	199
Test	86	1,177	101

Table 1: Total individual documents, mentions, co-reference clusters of each subset excluding singletons.

We built a web scraper that scans for new articles from the ‘Science’ and ‘Technology’ sections of 3 well-known online news outlets (BBC, The Guardian, New York Times) and press releases from Eurekalert, a widely popular scientific press release aggregator. Once a newspaper article and related scientific paper are detected, the full text from the news article and the scientific paper abstract and metadata are stored. Where available the full scientific paper content is also collected. We ran the scraper between April and June 2020 collecting news articles and scientific papers including preprints discussing a range of topics such as astronomy, computer science and biology (incl. coverage of COVID-19). New relevant content is downloaded and ingested into our annotation tool (see Section 3.4) on an ongoing basis as it becomes available.

3.2 Article Summarisation

Newspaper articles and scientific papers are long and often complex documents, usually spanning multiple pages, particularly the latter. Moreover the two document types differ significantly in length. Comparing documents of such uneven length is a difficult task for human annotators. We also assume that asking human annotators to read the documents in their entirety to identify co-references would be particularly hard with a very low chance for good inter-annotator agreement (IAA). We therefore decided to simplify the task by asking annotators to compare summaries of the newspaper article (5-10 sentences long) and the scientific paper (abstract).

For each document pair, we ask the annotators to identify co-referent mentions between the scientific paper abstract and a summary of the news article that is of similar length (e.g. 5-10 sentences). Scientific paper abstracts act as a natural summary of a scientific work and have been used as a strong baseline or even a gold-standard in scientific summarisation tasks (Liakata et al., 2013). Furthermore, abstracts are almost always available rather than behind paywalls like full text articles. For news summarisation, we used a state-of-the-art extractive model (Grenander et al., 2019) to extract sentences forming a summary of the original text. This model provides a summary de-biasing mechanism preventing it from focusing on specific parts of the full article, preserving the summary's informational authenticity as much as possible.

The difference in style between the two documents is preserved by both types of summary since abstracts are written in the same scientific style as full papers and the extractive summaries use verbatim excerpts of the original news articles.

3.3 Generation of pairs for annotation

To populate our annotation tool, we generate pairs of candidate cross-document mentions to be evaluated by the user. Candidate mentions are identified by using spaCy (Honnibal and Montani, 2017) for the recognition of noun phrases and named entities from each input document pair (abstract-news summary). For each pair of documents, pairs of all possible mention combinations are generated and stored for annotation.

In any given pair of documents, the majority of mention pairs (M_0, M_1) generated automatically in this way will not co-refer thus resulting in a vastly imbalanced dataset and also running the risk of demotivating annotators. To ensure that annotators are exposed to both positive and negative examples, we use a similarity score to rank examples based on how likely they are to co-refer. The first step in generating a similarity score s is to concatenate each abstract-news-summary pair together: "summary [SEP] abstract" into a pre-trained BERT$_{\text{large}}$ model. Then we take the mean of the word vectors that correspond to the mention spans within the documents and calculate the cosine similarity of these vectors. We find that this BERT-based similarity score performs well in practice. We also use it in combination with a thresholding policy as one of our baseline models in Section 4.

3.4 Annotation Tool & Interface

We developed an open source annotation tool[5] that allows humans to identify cross-document coreference between each pair of related documents. Whilst designing this tool, we made a number of decisions to simplify the task and provide clear instructions for the human annotators in order to encourage consistent annotation behaviour.

To maximise the quality and consistency of annotations in our corpus, we simplified the task as much as possible for the end user. Annotation tasks were framed as a single yes or no question: "Are **x** and **y** mentions of the same entity?". Mentions in context were shown in bold font whereas mentions already flagged as co-referent were shown in green. This enabled annotators to understand the implications for existing co-reference chains before responding (see Figure 3). Questions were generated and ranked via our task generation pipeline (see Section 3.3 above).

We added two additional features to our annotation interface to improve annotators' experience and to speed up the annotation process. Firstly, if the candidate pair is marked as co-referent, the user is allowed to add more mentions to the coreference cluster at once. Secondly, inspired by (Li et al., 2020), if the automatically shown mention pair is not co-referent, the user can select a different mention that is co-referent.

The upstream automated mention detection mechanism can sometimes introduce incomplete or erroneous mentions, leading to comparisons that don't make sense or that are particularly difficult. Therefore, annotators can also move or resize the mention spans they are annotating.

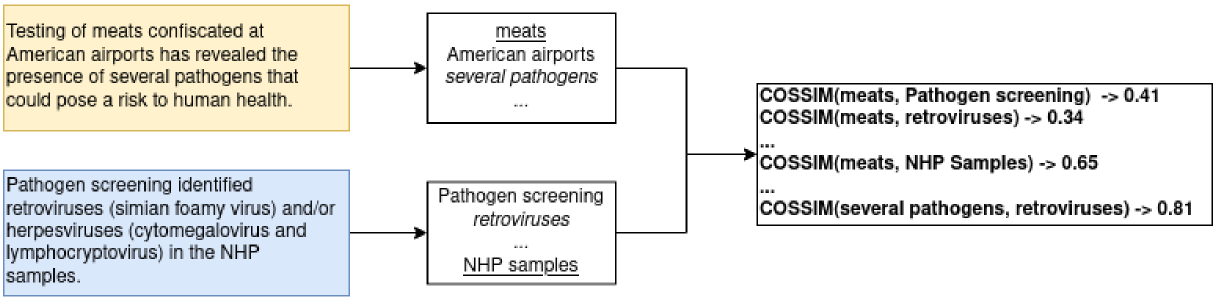
We use string offsets of mention span pairs to tokens to check that they do not overlap with each other in order to prevent the creation of duplicates. Figure 1 shows an illustrated example of the generation pipeline for mention pairs.

3.5 Annotation Protocol

We recruited three university-educated human annotators and provided them with detailed annotation guidelines for the resolution of yes/no questions on potentially co-referring entities in pairs from the ordered queue described above. By default each entity pair resolution is carried out once, allowing us to quickly expand our data set. However, we pseudo-randomly sample 5% of men-

---

[5] https://github.com/ravenscroftj/cdcrtool



Testing of meats confiscated at American airports has revealed the presence of several pathogens that could pose a risk to human health.

Pathogen screening identified retroviruses (simian foamy virus) and/or herpesviruses (cytomegalovirus and lymphocryptovirus) in the NHP samples.

Figure 1: Illustration of the generation process for pairs of potentially co-referring expressions, left boxes represent related news summary (top) and abstract (bottom), co-referent entity pairs in middle boxes shown with same formatting (underline,italic).

tion pairs in order to calculate inter-annotator-agreement (IAA) and make sure that data collected from the tool is consistent and suitable for modelling. New entity pairs for IAA are continually sampled as new document pairs and mention tuples are added to the corpus by the web scraper (Section 3.1). The annotation system puts mention pairs flagged for IAA first in the annotation queue. Thus, all annotators are required to complete IAA comparisons before moving on to novel mention pairs. This allows us to ensure that all annotators are well represented in the IAA exercise. To avoid annotators being faced with a huge backlog of IAA comparisons before being able to proceed with novel annotations, we also limited the number of comparisons for IAA required by each user to a maximum of 150 per week.

3.6 Task Difficulty and Annotator Agreement

We anticipated that annotation of the CD^{2}CR corpus would be difficult in nature due to its dependencies on context and lexical style. We invited users to provide feedback regularly to help us refine and clarify our guidelines and annotation tool in an iterative fashion. Users could alert us to examples they found challenging by flagging them as difficult in the tool. Qualitative analysis of the subset of 'difficult' cases showed that the resolution of mention pairs is often perceived by annotators as difficult when:

- Deep subject-matter-expertise is required to understand the mentions, e.g. is "jasmonic acid" the same as "regulator cis -(+)-12-oxophytodienoic acid".
- Mentions involve non-commutable set membership ambiguity e.g. "Diplodocidae" and "the dinosaurs"
- Mentions are context dependent e.g. "the struggling insect" and "the monarch butterfly".

This feedback prompted the introduction of highlighting for existing co-reference chains in the user interface (as described in section 3.4 above) to make it easier to tell when non-commutable set membership would likely introduce inconsistencies into the dataset. For mention pairs requiring subject-matter-expertise, annotators were encouraged to research the terms online. For context sensitive mention pairs, annotators were encouraged to read the full news article and full scientific paper in order to make a decision.

In our 11 page annotation guidelines document (appendix) we describe the use of our annotation tool and illustrate some challenging CD^{2}CR tasks and resolution strategies. For example precise entities mentioned in the scientific document may be referenced using ambiguous exophoric mentions in the news article (e.g. 'a mountain breed of sheep' vs 'eight ovis aries'). Our guidelines require resolving these cases based on the journalist's intent (e.g. 'a mountain breed' refers to the 'ovis aries' sheep involved in the experiment).

We evaluated the final pairwise agreement between annotators using Cohen's Kappa (Cohen, 1960) (κ_{cohen}) and an aggregate 'n-way' agreement score using Fleiss' Kappa (Fleiss, 1971) (κ_{fleiss}). Pairwise κ_{cohen} is shown in table 2 along with the total number of tasks each annotator completed. Annotator 3 (A3) shows the most consistent agreement with the other two annotators. Our Fleiss' Kappa analysis of tasks common across the three annotators gave $\kappa_{\text{fleiss}} = 0.554$. We note that Fleiss' Kappa is a relatively harsh metric and values, like ours, between 0.41 and 0.60 are considered to demonstrate 'moderate agreement' (Landis

	# Annotations	A1	A2	A3
A1	10,685	-	0.492	0.600
A2	3,051	0.492	-	0.500
A3	9,847	0.600	0.500	-

Table 2: Number of Annotations and Pairwise Cohen's Kappa scores κ_{cohen} between annotators.

and Koch, 1977). We also carried out Fleiss' Kappa analysis on the subset of mention pairs that were completed by all annotators and were also marked as difficult by at least one user (180 mention pairs in total). We found that for this subset of pairs, $\kappa_{\text{fleiss}} = 0.399$ which is considered to be fair agreement(Landis and Koch, 1977).

4 Model

Below we describe several baseline models including state of the art CDCR models that we used to evaluate how well current approaches can be used in our CD^{2}CR task setting.

4.1 BERT Cosine Similarity (BCOS) Baseline

In this model we calculate the cosine-similarity between embeddings of the two mentions in context (M_0, M_1) encoded using a pre-trained BERT model as discussed above in section 3.3. We define a thresholding function f to decide if M_0 and M_1 are co-referent ($f(x) = 1$) or not ($f(x) = 0$):

$$f(x) = \begin{cases} 1, & \text{if COSSIM}(M_0, M_1) \geq t \\ 0, & \text{otherwise} \end{cases}$$

During inference, we pass this function over all pairs M_0, M_1 and infer missing links such that if $f(A, B) = 1$ and $f(B, C) = 1$ then $f(A, C) = 1$.

Based on Figure 2, we test values in increments of 0.01 between 0.3 and 0.8 inclusive for threshold cut off t. We evaluated the baseline by measuring its accuracy at predicting co-reference in each mention pair in the CD^2CR development set. The best performance was attained when $t = 0.65$. A visualisation of the BERT Cosine Similarity distributions of co-referent and non co-referent annotated mention pairs can be seen in Figure 2.

Co-referent mention pairs tend to have a slightly higher BERT cosine similarity than non co-referent mention pairs but there is significant overlap of the two distributions suggesting that in many cases BERT similarity is too simplistic a measure.

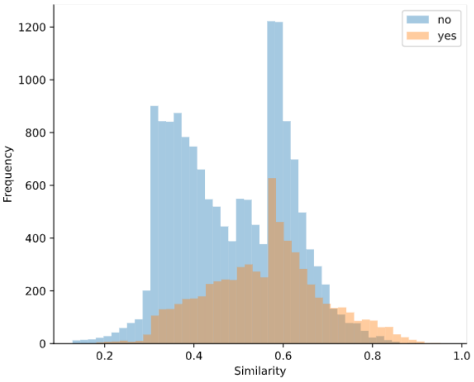


Figure 2: BERT Cosine Similarity frequency distribution for co-referent (Yes) and non-co-referent (No) mention pairs in the CD^{2}CR corpus.

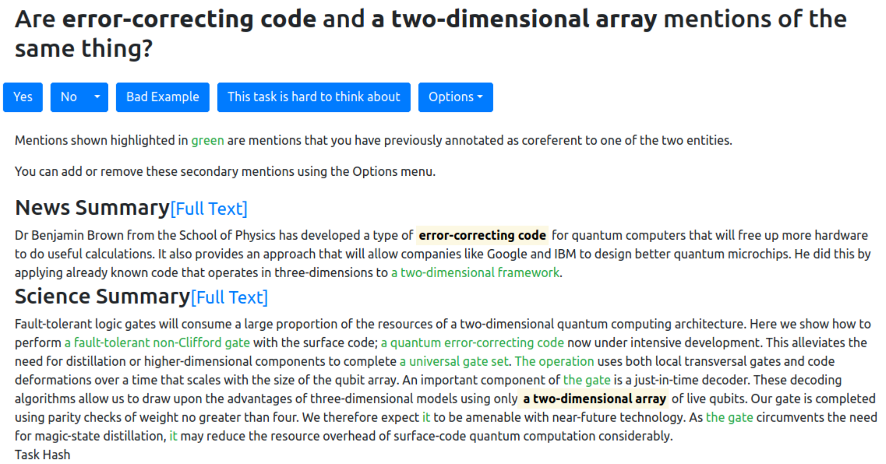
4.2 Entities Only Baseline (CA)

We use a state-of-the-art model (Cattan et al., 2020) (CA) for cross-document co-reference resolution. In this model, each document is separately encoded using a RoBERTa encoder (without fine-tuning) to get contextualized representations for each token. Then, similarly to the within-document coreference model by Lee et al. (2017), the mention spans are represented by the concatenation of four vectors: the vectors of the first and last token in the span, an attention-weighted sum of the span token vectors, and a feature vector to encode the span width. Two mention representations are then concatenated and fed to a feed-forward network to learn a likelihood score for whether two mentions co-refer. At inference time, agglomerative clustering is used on the pairwise scores to form coreference clusters.

The CA model is trained to perform both event and entity recognition on the ECB+ corpus (Cybulska and Vossen, 2014) In our setting there is no event detection subtask so, for fair comparison, we pre-train the CA model on ECB+ entity annotations only and evaluate it on our new CD^{2}CR task to see how well it generalises to our task setting.

4.3 CA + Fine-Tuned (CA-FT) Baseline

Here we aim to evaluate whether fine tuning the CA model from section 4.2 using the CD^{2}CR corpus can improve its performance in the new task setting. The CA model is first trained on the ECB+ corpus in the manner described above. We then further fine-tune the feed-forward model (without affecting the RoBERTa encoder) on the CD^{2}CR corpus for 10 epochs with early stopping. Pseudo-random sub-sampling is carried out on the training set to



Are **error-correcting code** and **a two-dimensional array** mentions of the same thing?

Mentions shown highlighted in green are mentions that you have previously annotated as coreferent to one of the two entities.

You can add or remove these secondary mentions using the Options menu.

News Summary[Full Text]

Dr Benjamin Brown from the School of Physics has developed a type of **error-correcting code** for quantum computers that will free up more hardware to do useful calculations. It also provides an approach that will allow companies like Google and IBM to design better quantum microchips. He did this by applying already known code that operates in three-dimensions to a two-dimensional framework.

Science Summary[Full Text]

Fault-tolerant logic gates will consume a large proportion of the resources of a two-dimensional quantum computing architecture. Here we show how to perform a fault-tolerant non-Clifford gate with the surface code; a quantum error-correcting code now under intensive development. This alleviates the need for distillation or higher-dimensional components to complete a universal gate set. The operation uses both local transversal gates and code deformations over a time that scales with the size of the qubit array. An important component of the gate is a just-in-time decoder. These decoding algorithms allow us to draw upon the advantages of three-dimensional models using only **a two-dimensional array** of live qubits. Our gate is completed using parity checks of weight no greater than four. We therefore expect it to be amenable with near-future technology. As the gate circumvents the need for magic-state distillation, it may reduce the resource overhead of surface-code quantum computation considerably.
Task Hash

Figure 3: An example of a cross-document co-reference task presented within our annotation tool.

ensure a balance of co-referent and non-co-referent mention pairs.

4.4 CA - Vanilla (CA-V) Baseline

Here we aim to evaluate whether training the CA model on the CD^{2}CR dataset from the RoBERTa baseline without first training on the ECB+ corpus allows it to fit well to the new task setting. We re-initialise the CA encoder (Section 4.2) using weights from RoBERTa (Liu et al., 2019) and randomly initialise the remaining model parameters. We then train the model on the CD^{2}CR corpus for up to 20 epochs with early stopping with pseudo-random sub-sampling as above.

4.5 CA - SciBERT (CA-S) Baseline

This model is the same as CA-V but we replace the RoBERTa encoder with SciBERT (Beltagy et al., 2019), a version of BERT pre-trained on scientific literature in order to test whether the scientific terms and context captured by SciBERT improve performance at the CD^{2}CR task compared to RoBERTa. Similarly to CA-V in section 4.4, we initialise the BERT model with weights from SciBERT$_{\text{scivocab-uncased}}$ (Beltagy et al., 2019) and randomly initialise the remaining model parameters, training on the CD^{2}CR corpus for up to 20 epochs with early stopping.

5 Results and Discussion

We evaluate each of the model baselines described in section 4 above on the test subset of our CD^{2}CR corpus. Results are shown in table 3.

For the purposes of evaluation, we use named entity spans from the manually annotated CD^{2}CR as the "gold standard" in all experiments rather than using the end-to-end Named Entity Recognition capabilities provided by some of the models. We evaluate the models using the metrics described by Vilain et al. (1995) (henceforth MUC) and Bagga and Baldwin (1998) (henceforth B^3). MUC F1, precision and recall are defined in terms of pairwise co-reference relationships between each mention. B^3, F1, precision and recall are defined in terms of presence or absence of specific entities in the cluster. When measuring B^3, we remove entities with no co-references (singletons) from the evaluation to avoid inflation of results (Cattan et al., 2020).

The threshold baseline (BCOS) gives the highest MUC recall but also poor MUC precision and poorest B^3 precision. The B^3 metric is highly specific with respect to false-positive entity mentions and strongly penalises BCOS for linking all non-coreferent pairs with $COSSIM(M_0, M_1) \geq 0.65$. Furthermore, Fig. 2 shows that a thresholding strategy is clearly sub-optimal given that there is a significant overlap of co-referent and non-co-referent pairs with only a small minority of pairs at the top and bottom of the distribution that do not overlap.

Model	**MUC**			**B^3**		
	P	**R**	**F1**	**P**	**R**	**F1**
BCOS	0.42	**0.94**	0.58	0.01	0.45	0.00
CA	0.41	0.51	0.46	**0.39**	0.33	0.35
CA-V	0.50	0.69	**0.58**	0.35	0.57	**0.44**
CA-FT	0.47	0.71	0.52	0.30	**0.62**	0.41
CA-S	**0.58**	0.46	0.51	0.32	0.53	0.39

Table 3: MUC and B^3 results from running baseline models on CD^{2}CR test subset, BCOS threshold=0.65

Test Type	Co-referent?	Pass Rate & Total Tests	Example test case and outcome for test case
Anaphora and Exophora resolution	Yes	47.1% (16/34)	M1: ...to boost **the struggling insect**'s numbers... *[PASS]* M2: the annual migration of the **monarch butterfly**...
	No	76.5% (26/34)	M1: ...**monarchs** raised in captivity... *[FAIL]* M2: ... wild-caught **monarchs** in an indoor environment...
Subset relationship resolution	Yes	24.3% (9/37)	M1: ...it was in fact a hive of **human activity**... *[FAIL]* M2: ...**Pre-Columbian cultural developments**...
	No	60.0% (18/30)	M1: ... **the carnivore's skull**... *[FAIL]* M2: ... **the gigantic extinct Agriotherium africanum**
Paraphrase resolution	Yes	33.3% (13/39)	M1: ...**a giant short-faced bear**... *[PASS]* M2: ...**the gigantic extinct Agriotherium africanum**...
	No	80.5% (29/36)	M1: ...the energy that **existing techniques** require *[FAIL]* M2: ...the lack of **efficient catalysts** for ammonia synthesis

Table 4: A breakdown of specific tests carried out on CA-V model against three challenging types of relationships found in the CD^2CR corpus. [PASS] or [FAIL] indicates CA-V model correctness. Pass Rate is mathematically equivalent to Recall for test sets.

Therefore, despite its promising MUC F1 score, it is clear that BCOS is not useful in practical terms.

Whilst our thresholding baseline above uses BERT, RoBERTa is used by Cattan et al. (2020) as the basis for their state-of-the-art model and thus for our models based on their work. Although the two models have the same architecture, RoBERTa has been shown to outperform BERT at a range of tasks (Liu et al., 2019). However, as shown in Figure 4, the cosine similarity distribution of mention pair embeddings produced by RoBERTa is compressed to use a smaller area of the potential distribution space compared to that of BERT (Figure 2). This compression of similarities may imply a reduction in RoBERTa's ability to discriminate in our task setting. Liu et al. (2019) explain that their byte-pair-encoding (BPE) mechanism, which expands RoBERTa's sub-word vocabulary and simplifies pre-processing, can reduce model performance for some tasks, although this is not further explored in their work. We leave further exploration of RoBERTa's BPE scheme and its effects on the CD^{2}CR task setting to future work.

All of the models specifically trained on the CD^{2}CR corpus (CA-V, CA-FT, CA-S) outperform the CA model by a large margin. Furthermore, the CA-V model (without pre-training on ECB+ corpus) outperforms the CA-FT model (with ECB+ pre-training) by 6% MUC and 3% B^3. These results suggest that the CD^{2}CR task setting is distinct from the CDCR and ECB+ task setting and that this distinction is not solvable with fine-tuning.

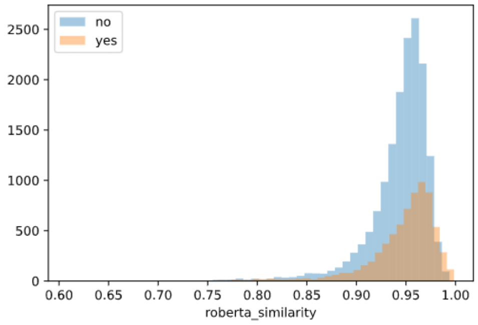


Figure 4: RoBERTa Cosine Similarity frequency distribution for co-referent (Yes) and non-co-referent (No) mention pairs in the CD^{2}CR corpus. Distribution is compressed between 0.8 and 1.0.

In terms of both MUC and B^3, CA-S performs much worse than CA-V suggesting that SciBERT embeddings are less effective than RoBERTa embeddings in this task setting. We hypothesise that SciBERT's specialisation towards scientific embeddings may come at the cost of significantly worse news summary embeddings when compared to those produced by RoBERTa.

We next evaluate our best performing CD^{2}CR baseline model (CA-V) at the entity resolution CDCR task using the ECB+ test corpus, to see how well it generalises to the original CDCR task. Results are presented in 5 along-side Cattan et al's original model results (CA). The CA-V model still shows good performance, despite a small drop, when compared to the original CA model. The drop in B^3 F1 is more pronounced than MUC but is still broadly in line with other contemporary CDCR systems (Cattan et al., 2020). The CA-V model demonstrates a promising ability to generalise beyond our corpus to other tasks and reveals an interesting correspondence between CDCR and CD^{2}CR settings.

Model	**MUC**			B^3		
	P	**R**	**F1**	**P**	**R**	**F1**
CA	0.86	0.82	0.84	0.63	0.68	0.65
CA-V	0.82	0.81	0.81	0.56	0.53	0.55

Table 5: MUC and B^3 results from running the CD^{2}CR baseline model (CA-V) on ECB+ dataset compared with original Cattan et al. (2020) (CA).

Finally, the best model (CA-V) is analysed using a series of challenging test cases inspired by Ribeiro et al (2020). These test cases were created using 210 manually annotated mention-pairs found in the test subset of the CD^2CR corpus according to the type of relationship illustrated (Anaphora & Exophora, Subset relationships, paraphrases). We collected a balanced set of 30-40 examples of both co-referent and non-coreferent-but-challenging pairs for each type of relationship (exact numbers in Table 4). We then recorded whether the model correctly predicted co-reference for these pairs. The results along with illustrative examples of each relationship type are shown in Table 4. The results suggest that the model is better at identifying non-co-referent pairs than co-referent pairs and that it struggles with positive co-referent mentions for all three types of relationship. The model struggles to relate general reader-friendly descriptions of entities from news articles to precise and clinical descriptions found in scientific papers. The model often successfully identifies related concepts such as ‘the carnivore’s skull’ and ‘Agriotherium africanum’. However it is unable to deal with the complexity of these relationships and appears to conflate ‘related’ with ‘co-referent’, which is likely due to lack of lexical knowledge as we discussed in section 2.3. Figure 5 shows significant overlap between co-referent and non-co-referent RoBERTa-based cosine similarities, which can also be observed for the wider corpus in Figure 4, but is especially bad for these test examples. This overlap suggests that disentangling these pairs is likely to be a challenging task for the downstream classification layer in the CA-V model. These challenges are less likely to occur in homogeneous corpora like ECB+ where descriptions and relationships remain consistent in detail and complexity.

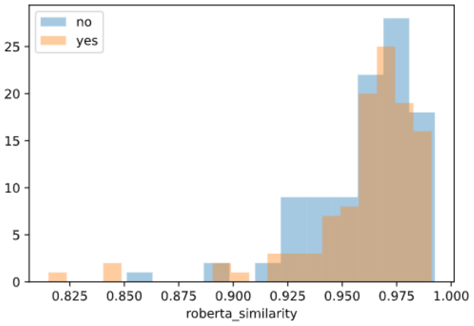


Figure 5: RoBERTa-based mention pair similarity frequency distribution for test examples from Table 4. ’yes’ and ’no’ for ’co-referent’ and ’not-co-referent’ respectively

6 Conclusion

We have defined cross-document, cross-domain co-reference resolution (CD^{2}CR), a special and challenging case of cross-document co-reference resolution for comparing mentions across documents of different types and/or themes. We have constructed a specialised CD^{2}CR annotated dataset, available, along with our annotation guidelines and tool, as a free and open resource for future research. We have shown that state-of-the-art CDCR models do not perform well on the CD^{2}CR dataset without specific training. Furthermore, even with task-specific training, models perform modestly and leave room for further research and improvement. Finally, we show that the understanding of semantic relatedness offered by current generation transformer-based language models may not be precise enough to reliably resolve complex linguistic relationships such as those found in CD^{2}CR as well as other types of co-reference resolution and relationship extraction tasks. The use of semantic enrichment techniques (such as those discussed in Section 2.3) to improve model performance in the CD^{2}CR task should be investigated as future work.

Acknowledgments

This work was supported by The Alan Turing Institute and University of Warwick under respective EPSRC grants EP/N510129/1 and EP/L016400/1.

References

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of ACL '98/COLING '98*, page 79–85, USA. Association for Computational Linguistics.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting Joint Modeling of Cross-document Entity and Event Coreference Resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2020. Streamlining cross-document coreference resolution: Evaluation and modeling. arXiv:2009.11032.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? Lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552.

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

Sourav Dutta and Gerhard Weikum. 2015. Cross-Document Co-Reference Resolution using Sample-Based Clustering with Knowledge Enrichment. *Transactions of the Association for Computational Linguistics*, 3:15–28.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Matt Grenander, Yue Dong, Jackie Chi Kit Cheung, and Annie Louis. 2019. Countering the Effects of Lead Bias in News Summarization via Multi-Stage Training and Auxiliary Losses. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6019–6024.

Zellig S. Harris. 1954. Distributional structure. *WORD*, 10(2-3):146–162.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *Proceedings of the 2010 Text Analysis Conference*.

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

JR Landis and GG Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159—174.

Anne Lauscher, Ivan Vulić, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavaš. 2020. Specializing unsupervised pretraining models for word-level semantic similarity. *Computing Research Repository*, arXiv:1909.02339.

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

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Belinda Z. Li, Gabriel Stanovsky, and Luke Zettlemoyer. 2020. Active learning for coreference resolution using discrete annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8320–8331.

Maria Liakata, Simon Dobnik, Shyamasree Saha, Colin Batchelor, and Dietrich Rebholz-Schuhmann. 2013. A discourse-driven content model for summarising scientific articles evaluated in a complex question answering task. In *Proceedings of Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pages 747–757. Association for Computational Linguistics.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Annie Louis and Ani Nenkova. 2013. A corpus of science journalism for analyzing writing quality. *Dialogue and Discourse*, 4(2):87–117.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237.

Edoardo Maria Ponti, Ivan Vulić, Goran Glavaš, Nikola Mrkšić, and Anna Korhonen. 2018. Adversarial propagation and zero-shot cross-lingual transfer of word vector specialization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 282–293.

Jonathan Raiman and Olivier Raiman. 2018. DeepType: Multilingual entity linking by neural type system evolution. In *AAAI Conference on Artificial Intelligence*.

Delip Rao, Paul McNamee, and Mark Dredze. 2010. Streaming cross document entity coreference resolution. In *Coling 2010: Posters*, pages 1050–1058, Beijing, China. Coling 2010 Organizing Committee.

James Ravenscroft, Amanda Clare, and Maria Liakata. 2018. HarriGT: A tool for linking news to science. In *Proceedings of ACL 2018, System Demonstrations*, pages 19–24.

Marta Recasens, Lluís Màrquez, Emili Sapena, M. Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio, and Yannick Versley. 2010. SemEval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 1–8. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

W. Shen, J. Wang, and J. Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 45–52.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *Computing Research Repository*, arXiv:2004.14974.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. Ontonotes release 5.0 LDC2013T19. Linguistic Data Consortium, Philadelphia, PA.

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

Xiaoyao Yin, Yangchen Huang, Bin Zhou, Aiping Li, Long Lan, and Yan Jia. 2019. Deep entity linking via eliminating semantic ambiguity with BERT. *IEEE Access*, 7:169434–169445.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 545–550.

AREDSUM: Adaptive Redundancy-Aware Iterative Sentence Ranking for Extractive Document Summarization

Keping Bi[1], **Rahul Jha**[2], **W. Bruce Croft**[1], **Asli Celikyilmaz**[3]
[1] University of Massachusetts Amherst, [2] Microsoft, [3] Microsoft Research AI
{kbi, croft}@cs.umass.edu, rajh@microsoft.com, asli@ieee.org

Abstract

Redundancy-aware extractive summarization systems score the *redundancy* of the sentences to be included in a summary either jointly with their *salience* information or separately as an additional sentence scoring step. Previous work shows the efficacy of jointly scoring and selecting sentences with neural sequence generation models. It is, however, not well-understood if the gain is due to better encoding techniques or better redundancy reduction approaches. Similarly, the contribution of salience versus diversity components on the created summary is not studied well. Building on the state-of-the-art encoding methods for summarization, we present two adaptive learning models: AREDSUM-SEQ that jointly considers salience and novelty during sentence selection; and a two-step AREDSUM-CTX that scores salience first, then learns to balance salience and redundancy, enabling the measurement of the impact of each aspect. Empirical results on CNN/DailyMail and NYT50 datasets show that by modeling diversity explicitly in a separate step, AREDSUM-CTX achieves significantly better performance than AREDSUM-SEQ as well as state-of-the-art extractive summarization baselines.

1 Introduction

Extractive summarization is the task of creating a summary by identifying and concatenating the most important sentences in a document (Liu and Lapata, 2019; Zhang et al., 2019; Zhou et al., 2018). Given a partial summary, the decision to include another sentence in the summary depends on two aspects: *salience*, which represents how much information the sentence carries; and *redundancy*, which represents how much information in the sentence is already included in the previously selected sentences.

Although there have been a few studies on redundancy a long time ago, most recent research on extractive summarization focuses on salience alone. They usually model sentence salience as a sequence labeling task (Kedzie et al., 2018; Cheng and Lapata, 2016) or classification task (Zhang et al., 2019) and do not conduct redundancy removal. Previous methods that consider redundancy usually use a separate step after salience scoring to handle redundancy, denoted as sentence selection (Carbonell and Goldstein, 1998; McDonald, 2007; Lin and Bilmes, 2011). Sentence selection often follows a greedy iterative ranking process that outputs one sentence at a time by taking into account the redundancy of candidate sentences with previously selected sentences.

Several approaches for modeling redundancy in sentence selection have been explored: heuristics-based methods such as Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), Trigram Blocking (TRIBLK) (Liu and Lapata, 2019), or model based approaches (Ren et al., 2016), etc. Heuristic-based methods are not adaptive since they usually apply the same rule to all the documents. Model-based approaches depend heavily on feature engineering and learn to score sentences via regression with point-wise loss, which has been shown to be inferior to pairwise loss or list-wise loss in ranking problems (Liu et al., 2009).

Redundancy has also been handled jointly with salience during the scoring process using neural sequence models (Zhou et al., 2018). NEUSUM (Zhou et al., 2018) scores sentences considering their salience as well as previous sentences in the output sequence and learns to predict the sentence with maximum relative gain given the partial output summary. Despite the improved efficacy, it is not well-understood if the gain is due to better encoding or better redundancy-aware iterative ranking approaches (i.e., the sequence generation).

In this work, we propose to study different types of redundancy-aware iterative ranking techniques

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 281–291
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

for extractive summarization that handle redundancy separately or jointly with salience. Extending BERTSUMEXT (Liu and Lapata, 2019), a state-of-the-art extractive summarization model, which uses heuristic-based Trigram Blocking (TRIBLK) for redundancy elimination, we propose two supervised redundancy-aware iterative sentence ranking methods for summary prediction. Our first model, AREDSUM-SEQ, introduces a transformer-based conditional sentence order generator network to score and select sentences by jointly considering their salience and diversity within the selected summary sentences. Our second model, AREDSUM-CTX, uses an additional sentence selection model to learn to *balance* the salience and redundancy of constructed summaries. It incorporates surface features (such as n-gram overlap ratio and semantic match scores) to instrument the diversity aspect. We compare the performance of our redundancy-aware sentence ranking methods with trigram-blocking (Liu and Lapata, 2019), as well as summarization baselines with or without considering redundancy on two commonly used datasets, CNN/DailyMail and New York Times (NYT50). Experimental results show that our proposed AREDSUM-CTX can achieve better performance by reducing redundancy and outperform all the baselines on these two datasets. The model's advantage can be attributed to its adaptiveness to scenarios in which redundancy removal has different potential gains.

In summary, our contributions are: 1) we propose two redundancy-aware iterative ranking methods for extractive summarization extending BERTSUMEXT; 2) we conduct comparative studies between our redundancy-aware models as well as the heuristic-based method that BERTSUMEXT uses; 3) our proposed AREDSUM-CTX significantly outperforms BERTSUMEXT and other competitive baselines on CNN/DailyMail and NYT50.

2 Related Work

Extractive summarization methods are usually decomposed into two subtasks, i.e., sentence scoring and sentence selection, which deal with salience and redundancy, respectively.

Salience Scoring. Graph-based models are widely used methods to score sentence salience in summarization (Erkan and Radev, 2004; Mihalcea and Tarau, 2004; Wan and Yang, 2006). There are also extensions to such methods, e.g., with clustering (Wan and Yang, 2008) or leveraging graph neural networks (Wang et al., 2020). Classical supervised extractive summarization uses classification or sequence labeling methods such as Naive Bayes (Kupiec et al., 1999), maximum entropy (Osborne, 2002), conditional random fields (Galley, 2006) or hidden markov model (Conroy et al., 2004). Human engineered features are heavily used in these methods such as word frequency and sentence length (Nenkova et al., 2006).

In recent years, neural models have replaced older models to score the salience of sentences. Hierarchical LSTMs and CNNs have replaced manually engineered features. LSTM decoders are employed to do sequence labeling (Cheng and Lapata, 2016; Nallapati et al., 2017; Kedzie et al., 2018). These architectures are widely used and also extended with reinforcement learning (Narayan et al., 2018; Dong et al., 2018). More recently, summarization methods based on BERT (Devlin et al., 2018) have been shown to achieve state-of-the-art performance (Liu and Lapata, 2019; Zhang et al., 2019; Zhong et al., 2019; Zhou et al., 2020) on salience for extractive summarization.

Sentence Selection. There are relatively fewer methods that study sentence selection to avoid redundancy. Integer Linear Programming based methods (McDonald, 2007) formulate sentence selection as an optimizing problem under the summary length constraint. Lin and Bilmes (2011) propose to find the optimal subset of sentences with submodular functions. Greedy strategies such as Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998) select the sentence that has maximal salience score and is minimally redundant iteratively. Trigram blocking (Liu and Lapata, 2019) follows the intuition of MMR and filters out sentences that have trigram overlap with previously extracted sentences. Ren et al. (2016) leverage two groups of handcrafted features to capture informativeness and redundancy respectively during sentence selection. In contrast to learning a separate model for sentence selection, Zhou et al. (2018) propose to jointly learn to score and select sentences with a sequence generation model. However, it is not compared with other redundancy-aware techniques, and it is not clear whether its improvement upon other methods is from the sequence generation method or the encoding technique.

In this paper, we compare the efficacy of different sentence selection techniques grounded on the same BERT-based encoder. We propose two models that either conduct redundancy removal with a separate model or jointly with salience scoring and compare them with a heuristic-based method. As far as we know, our work is the first to conduct comparative studies on different types of redundancy-aware extractive summarization methods.

3 Iterative Sentence Ranking

We formulate single document extractive summarization as a task of iterative sentence ranking. Given a document $D = \{s_1, s_2, \cdots, s_L\}$ of L sentences, the goal is to extract t sentences, i.e., $\hat{S}_t = \{\hat{s}_k | 1 \leq k \leq t, \hat{s}_k \in D\}$, from D that can best summarize it. With a limit of selected sentence count l, the process of extracting sentences can be considered as a l-step iterative ranking problem. At each k-th step ($1 \leq k \leq l$), given the current summary $\hat{S}_{k-1}$, a new sentence s_k is selected from the remaining sentences $D \setminus \hat{S}_{k-1}$ and added to the summary. Function $M(\hat{S}_k; S^*)$[1] measures the similarity between the extracted summary $\hat{S}_k$ and the ground truth summary S^*. The objective is to learn a scoring function $f(\cdot)$ so that the best sentence $\hat{s}_k$ selected according to $f(\cdot)$ can maximize the gain of the output summary:

$$\begin{aligned} &\arg\max_f M(\{\hat{s}_k\} \cup \hat{S}_{k-1}); S^*) \\ &\hat{s}_k = \arg\max_{s_i \in D \setminus \hat{S}_{k-1}} f(\{s_i\} \cup \hat{S}_{k-1}) \end{aligned} \tag{1}$$

$\hat{s}_k$ needs to be both salient in the document and novel in the current context $\hat{S}_{k-1}$. Note that at the beginning $\hat{S}_0 = \emptyset$.

Since ground truth summaries S^* of existing summarization corpora are usually abstractive summaries written by experts, previous studies on extractive summarization usually extract a group of pseudo ground truth sentences $\hat{S}^*$ from D based on their similarities to the ground truth summaries S^* for training purposes. Then labels 1 and 0 are assigned to sentences in $\hat{S}^*$ and the other sentences in D. In this case, $M(\hat{S}_t; \hat{S}^*)$ is used to guide training instead of $M(\hat{S}_t; S^*)$.

4 Redundancy-Aware Summarization

Most recent redundancy-aware extractive summarization systems use heuristics to select diverse sentences after salience scoring (Cao et al., 2015; Ren et al., 2017; Liu and Lapata, 2019). Among them, BERTSUMEXT (Liu and Lapata, 2019) is a state-of-the-art model with trigram-blocking (TRIBLK) that reduces redundancy by filtering out sentences that have trigram overlap with previously selected ones at each time step. As we empirically show later in § 6, heuristics can be effective on some datasets yet harmful on others since it applies the same rule to all the documents.

In contrast, we present an adaptive learning process for redundancy-aware extractive summarization, AREDSUM, and introduce two methods, AREDSUM-SEQ and AREDSUM-CTX, extending BERTSUMEXT by either consider redundancy jointly with salience during sentence scoring or separately with an additional selection model.

4.1 Document Encoder

First, we introduce the sentence and document encoder shared by both our variations of AREDSUM, shown in Figure 1. In sentence-level encoding, a [SEP] token is appended to each sentence to indicate the sentence boundaries and a [CLS] token is inserted before each sentence in the document to aggregate the information of the sentence. In addition to token and positional embeddings, as in BERTSUMEXT (Liu and Lapata, 2019), we also use interval segment embeddings E_A and E_B to distinguish sentences at odd and even positions in the document respectively. Following multiple transformer encoder layers, we represent each sentence s_i by the output representation of the [CLS] symbol preceding s_i. These symbols capture the features of the following tokens in the sentence while attending over all other tokens in the document through the transformer layers.

We further conduct document-level encoding on the sentence-level representations from the [CLS] tokens, denoted as E_{s_i}, as well as their positional embeddings, E'_i, with another stack of transformer layers. We add a document embedding E_D before the sequence of sentence embeddings to represent the whole document. The final representation of D and each sentence s_i can be obtained from the output of the multiple transformer layers, denoted as h_{s_i} and h_D.

4.2 AREDSUM-SEQ: Sequence Generation

Our first model, AREDSUM-SEQ, strictly considers the *order* of the target selected sentences while jointly modeling the redundancy and salience of the next sentence. It uses a transformer decoder

---

[1] In § 4 and experiments, we use ROUGE to define $M(\cdot)$

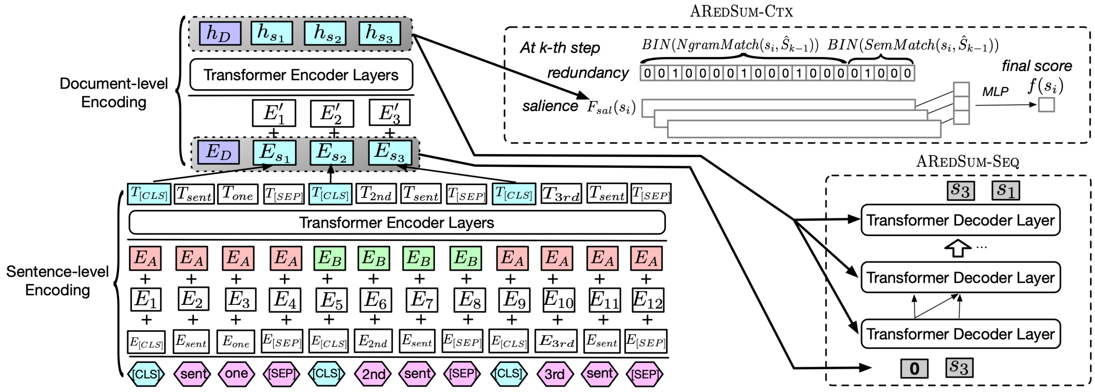


Figure 1: Overview of the proposed models ARedSum-Seq and ARedSum-Ctx sharing the same Bert-based encoder from BertSumExt.

module (Vaswani et al., 2017) to learn to select and order a sequence of sentences from the document as a summary. Our model is different from standard auto-regressive decoders. Each decoder block takes in a sequence of tokens (word-units) as input to generate the next possible token from a pre-defined vocabulary. Instead, our decoder is a conditional model that takes a sequence of sentence representations as input and selects a sentence with the maximum gain to be included in the summary from the rest of the document's sentences.

Following a standard transformer encoder-decoder architecture (Vaswani et al., 2017), at each decoding step k, a current hidden state is obtained with a stack of transformer decoder layers:

$$h'_{\hat{s}_{k-1}} = \mathrm{Dec}([E_{\hat{s}_1},\cdots,E_{\hat{s}_{k-1}}],[h_{s_1},h_{s_2},\cdots,h_{s_L}]) \quad (2)$$

where $[h_{s_1}, h_{s_2}, \cdots, h_{s_L}]$ are the output sentence representations after the document-level encoding in Figure 1 and $[E_{\hat{s}_1}, \cdots, E_{\hat{s}_{k-1}}]$ are the embeddings of the so far selected sentences. $\hat{S}_{k-1} = \emptyset$ and $E_{\hat{s}_0} = \mathbf{0}$ when $k = 1$. Note that sentence embeddings that are fed to the document-level transformer encoders, i.e., $E_{s_1}, \cdots, E_{s_L}$, are used to represent the sentences in the target decoding space. Then, a two-layer MLP is used to score a candidate sentence s_i given the hidden state $h'_{\hat{s}_{k-1}}$:

$$o_l(s_i) = W_{2s} \tanh(W_{1s}[h'_{\hat{s}_{k-1}}; E_{s_i}]) \quad (3)$$

where W_{2s} and W_{1s} are the weights of the MLP (we omit the bias parameters for simplicity), and [;] denotes vector concatenation. In case the salience of s_i is not sufficiently captured in $o_l(s_i)$, we calculate a matching score $o_g(s_i)$ between s_i and the global context D, regardless of which sentences are selected previously [2]:

$$o_g(s_i) = \tanh(h_D W_{ds} h_{s_i}) \quad (4)$$

where W_{ds} is matrix for bilinear matching; h_D and h_{s_i} are the embeddings of the document D and sentence s_i output by the document-level encoder. The final score is the linear combination of o_l and o_g using the weight W_o:

$$o(s_i) = W_o(o_l(s_i) + o_g(s_i)) \quad (5)$$

The probability of any sentence s_i being selected at the k-th step is the softmax of $o(s_i)$ over the remaining candidate sentences s_j in D:

$$P(\hat{s}_k{=}s_i|\hat{S}_{k-1}) = \frac{\exp(o(s_i))}{\sum_{s_j \in D \setminus \hat{S}_{k-1}} \exp(o(s_j))} \quad (6)$$

Following NeuSum (Zhou et al., 2018), we train ARedSum-Seq to optimize for the *relative* ROUGE-F1 gain of each sentence with respect to so-far selected sentences $\hat{S}_{k-1}$.

$$g(s_i) {=} M(\{s_i\} {\cup} \hat{S}_{k-1}; S^*) {-} M(\hat{S}_{k-1}; S^*) \quad (7)$$

where $M(\{s_i\}{\cup}\hat{S}_{k-1}; S^*)$ and $M(\hat{S}_{k-1}; S^*)$ measure the ROUGE-F1 between the golden summary S^* and the so-far selected sentences $\hat{S}_{k-1}$ with and without the candidate s_i respectively. We rescale the gain $g(s_i)$ to [0,1] in case of negative values using a min-max normalization and get $\tilde{g}$. Then we use a softmax function with a temperature τ on the rescaled gain to produce a target distribution:

$$Q(s_i) = \frac{\exp(\tau \tilde{g}(s_i))}{\sum_{s_j \in D \setminus \hat{S}_{k-1}} \exp(\tau \tilde{g}(s_j))} \quad (8)$$

---

[2] Emphasizing salience with o_g enhances the performance.

The final training objective is to minimize the KL divergence between the probability distribution of sentence scores (Eq. 6) and their relative rouge gain (Eq. 8), i.e., $KL(P(\cdot)||Q(\cdot))$. This objective can be considered as a listwise ranking loss (Ai et al., 2018) that maximizes the probability of the target sentence while pushing down the probabilities of the other sentences. In this way, AREDSUM-SEQ combined the sentence scoring and selection in the same decoder framework, and the redundancy is implicitly captured by optimizing the ROUGE gain.

4.3 AREDSUM-CTX: Context-aware Sentence Ranker

We introduce a second model, AREDSUM-CTX, a context-aware ranker that scores salience first and then selects a sentence according to both its salience and redundancy adhering to the previously extracted sentences as context, as shown in Figure 1. In AREDSUM-CTX, we use a two-step process for scoring and selecting sentences for learning to construct a summary: In the *salience ranking* step, we focus on learning the salience of the sentences, while in the *ranking for sentence selection* step, we represent the redundancy explicitly via surface features and use a ranker to decide to promote or demote sentences based on their scores given the joint degree of their salience and redundancy.

Salience Ranking. By assuming that the sentence salience is independent of the previously selected sentences, we design the salience ranking of AREDSUM-CTX as a single step process rather than an iterative one. We measure the probability of a sentence to be included in $\hat{S}^*$ using a scoring function F_{sal} based on the bilinear matching between h_D and h_{s_i}, the transformer output after the document-level encoding, same as in Eq. 4.

$$F_{\text{sal}}(s_i) = \frac{\exp h_D W_{ds} h_{s_i}}{\sum_{j=1}^{j=L} \exp h_D W_{ds} h_{s_j}} \quad (9)$$

The learning objective is to maximize the log likelihood of the summary sentences in the training data:

$$\mathcal{L} = \sum_{s_i \in \hat{S}^*} \log F_{\text{sal}}(s_i) \quad (10)$$

Redundancy Features. In the selection step, we represent redundancy explicitly to let the model focus on learning how to balance salience and redundancy. We extract ngram-matching and semantic-matching features at each k-th step to indicate the redundancy of a candidate sentence s_i given the so-far selected sentences, i.e., $\hat{S}_{k-1}$. The ngram-matching feature $f_{n\text{-gram}}$ is computed as:

$$f_{n\text{-gram}} = \frac{|n\text{-gram}(\hat{S}_{k-1}) \cap n\text{-gram}(s_i)|}{n\text{-gram}(s_i)} \quad (11)$$

where n-gram(x) is the set of n contiguous words in x. We collect $f_{n\text{-gram}}$ for $n = \{1, 2, 3\}$. We also compute the semantic-matching feature f_{sem}:

$$f_{\text{sem}} = \max_{\hat{s}_j \in \hat{S}_{k-1}} \cos(h_{s_i}, h_{\hat{s}_j}) \quad (12)$$

Since most cosine values between output embeddings from the transformer layers fall in a small range near to 1, we apply a min-max normalization on f_{sem} to enlarge the value differences and obtain a updated feature $\tilde{f}_{\text{sem}}$.

The impact of redundancy features on final scores is not linear. Sentences with high redundancy values should be punished more. To capture the effect of the redundancy features at different value sections, we equally divide the range of $[0, 1]$ to m bins and discretize each feature to the corresponding bin according to its value, as shown in Figure 1. In this way, we convert each feature into a one-hot vector of length m and then we concatenate them to obtain a overall redundancy feature vector $F_{\text{red}}(s_i) = [f'_{1\text{-gram}}; f'_{2\text{-gram}}; f'_{3\text{-gram}}; \tilde{f}'_{\text{sem}}]$ where f' represents the one-hot vector after binning f.

Ranker for Sentence Selection. In the sentence selection step, AREDSUM-CTX only needs to learn how to score a sentence based on its redundancy features $F_{\text{red}}(s_i)$ and its salience score $F_{\text{sal}}(s_i)$ from Eq. 9. Note that the first selected sentence is the one ranked with the largest salience score. We use a three-dimensional matrix $W_{\mathcal{F}}$ to do a bilinear matching between the redundancy features and salience score and obtain a output matching vector with dimension d. Then we apply a single-layer MLP on top to output a final score:

$$f(s_i) = W_f \tanh(F_{\text{sal}}(s_i) W_{\mathcal{F}} F_{\text{red}}(s_i)) \quad (13)$$

During training, we randomly select $1, 2, \cdots, l$-1 sentences from the extracted ground-truth set $\hat{S}^*$ as the context and let the model learn to find the next sentence that is both salient and novel, where l is the maximum number of sentences to be included in the predicted summary. The training objective is the same as in § 4.2 except that $o(s_i)$ in Eq. 6 is replaced with $f(s_i)$ in Eq. 13. In contrast to

AREDSUM-SEQ where the target output is an ordered sequence, the loss of AREDSUM-CTX is not order-sensitive since the goal is always to predict the next best sentence given a set of unordered selected sentences as context.

5 Experimental Setup

5.1 Datasets

We evaluate our model on two standard extractive summarization datasets, namely CNN/DailyMail (Hermann et al., 2015) and NewYork Times (NYT) (Sandhaus, 2008).

CNN/DailyMail contains news articles associated with a few bullet points as the article's highlight. We use the standard splits of Hermann et al. (2015) which has 287,226 documents for training, 13,368 for validation, and 11,490 for testing. We conduct preprocessing following the same method in Liu and Lapata (2019). Entities are *not anonymized* in our experiments as in Zhou et al. (2018); See et al. (2017); Zhang et al. (2019); Liu and Lapata (2019). We truncate articles up to 512 tokens. To collect sentence labels for extractive summarization, we use a greedy strategy similar to (Nallapati et al., 2017; Zhang et al., 2019). We label the subset of sentences that can maximize ROUGE scores against the human-generated summary as 1 (sentence to be included in the summary). The remaining ones are labeled as 0.

NYT50 is an annotated corpus of the New York Times. Following Paulus et al. (2017) and Durrett et al. (2016), we discard marks and words such as "(s)" and "photo" at the end of the abstract and filter out the articles with summaries shorter than 50. We sort the articles chronologically and split the data into training/validation/test sets according to the ratio of $0.8/0.1/0.1$, yielding 133,602/16,700/16,700 documents, respectively. We following the same remaining steps for preprocessing and extractive label collection as the CNN/DailyMail.

5.2 Implementation Details

Our implementation [3] is based on PyTorch and BERTSUM(Liu and Lapata, 2019) [4]. We use "bert-base-uncased" version of BERT[5] to do sentence-level encoding. We fine-tune our models using the objective functions in § 4. We set the number of document-level transformer layers to 2. The dropout rate in all layers is 0.1. We search the best value of τ in Eq. 8 in $\{10, 20, 40, 60\}$. We train our models using the Adam optimizer with $\beta_1 = 0.9$, and $\beta_2 = 0.999$ for 2 epochs. We schedule the learning rate according to Vaswani et al. (2017) with initial value $2e$-3 and 10,000 warm-up steps.

We use teacher-forcing to train AREDSUM-SEQ. To learn the k-th sentence in the target sequence, we replace the first $k-1$ input sentences with other random sentences in the document with the probability of 0.2 [6]. We hypothesize that if the previously selected sentence is not always the golden (right) sentence, it can improve the model's robustness during training. We use two transformer layers in the decoder.

For AREDSUM-CTX, we train the salience ranker using the same settings as in Liu and Lapata (2019), and all the parameters in the salience ranker are fixed when we train the ranker for selection. This ensures that the salience score of each sentence stays the same during sentence selection. We select the optimum size of the bins for discretized redundancy features by sweeping the values in $\{10, 20, 30\}$ and the size of the output dimension d of $W_{\mathcal{F}}$ in Eq. 13 using $\{5, 10, 20, 30\}$.

5.3 Baselines

We compare our models to the state-of-the-art extractive summarization model BERTSUMEXT (Liu and Lapata, 2019), which uses Trigram Blocking (TRIBLK) (Paulus et al., 2017) to filter out sentences with trigram overlap with previously extracted sentences. We report the performance of BERTSUMEXT with and without TRIBLK separately to show the impact of this heuristic.

We also compare against other baselines including: LEAD3, NN-SE (Cheng and Lapata, 2016), SUMMARUNNER (Nallapati et al., 2017), Seq2Seq (Kedzie et al., 2018), NEUSUM (Zhou et al., 2018), and HIBERT (Zhang et al., 2019) along with ORACLE for upper bound on performance. LEAD3 is a commonly used effective baseline that extracts the first 3 sentences in the document. NN-SE (Cheng and Lapata, 2016) and SUMMARUNNER (Nallapati et al., 2017) both formulate extractive summarization as a sequence labelling task. NN-SE uses unidirectional GRU for both the encoding and decoding processes. SUMMARUNNER encodes

---

[3] `https://github.com/kepingbi/ARedSumSentRank`

[4] `https://github.com/nlpyang/BertSum`

[5] `https://git.io/fhbJQ`

[6] We found this number after light parameter value sweep.

sentences with BiGRU and considers salience, redundancy, absolute and relative positions of sentences during scoring. Seq2Seq (Kedzie et al., 2018) conducts binary classification by encoding the sentences with a bidirectional GRU (BiGRU) and using a separate decoder BiGRU to transform each sentence as a query vector that attends to the encoder output. NeuSum (Zhou et al., 2018) learns to jointly score and select sentences using a sequence-to-sequence model to optimize the marginal Rouge gain and reduce redundancy implicitly. HiBert (Zhang et al., 2019) pre-trains a hierarchical BERT for extractive summarization without dealing with redundancy. Among these methods, NeuSum implicitly reduces redundancy by jointly scoring and selecting sentences with a sequence generation model. SummaRuNNer considers redundancy during the sequence labeling. The other baselines do not conduct redundancy removal.

6 Results and Discussion

6.1 Automatic Evaluation Results

Following earlier work (Zhou et al., 2018; Liu and Lapata, 2019), we include 3 sentences as the summaries for each system for a fair comparison. We evaluate the full-length Rouge-F1 (Lin, 2004) of the extracted summaries and report Rouge-1, Rouge-2 and Rouge-L which indicates the unigrams, bigrams overlap and longest common subsequence against human edited summaries. The full-length Rouge-F1 (Lin, 2004) scores of the extracted summaries are evaluated using the official Perl script[7] for both CNN/DailyMail and NYT50. The results of NeuSum and HiBert are taken from their original papers while we obtained the rest of the results by re-running the models. Since in the previous work (Liu and Lapata, 2019; Zhang et al., 2019; Paulus et al., 2017) there are no consistent ways of pre-processing the NYT dataset for extractive summarization, we only report the evaluation results from the models we re-trained on this dataset in Table 2.

CNN/DailyMail. Results shown in Table 1 are all comparable as we use the same *non-anonymized* version of CNN/DailyMail. For BertSumExt-based methods, we observe that redundancy removal helps improve the Rouge score compared to BertSumExt. TriBlk has considerably better performances; ARedSum-Seq achieves better Rouge-1/L scores; ARedSum-Ctx significantly outperforms the other redundancy elimination methods. [8] The performance differences of ARedSum-Ctx and TriBlk comes from 30.6% summaries output by the two systems in the test set. In other cases, they agrees with each other. This shows that by *adaptively* balancing salience and diversity, ARedSum-Ctx is superior to TriBlk when redundancy removal is promising.

Table 1: Full-length Rouge (RG) F1 evaluation (%) on the CNN/DailyMail test set. a/b and A/B indicate significant improvements over BertSumExt/BertSumExt+TriBlk with $p < 0.05$ and $p < 0.0001$ respectively.

Model	RG-1	RG-2	RG-L
Oracle	52.59	31.24	48.87
Oracle+TriBlk	51.65	30.50	47.89
Lead3	40.42	17.62	36.67
NN-SE	40.81	17.91	37.03
Seq2Seq	41.83	19.29	38.28
SummaRuNNer	41.84	19.31	38.31
NeuSum	41.59	19.01	37.98
HiBert	42.37	19.95	38.83
BertSumExt	42.61	19.99	39.09
BertSumExt+TriBlk	43.25	20.24	39.63
ARedSum-Seq	42.72^{a}	19.82	39.15
ARedSum-Ctx	$\mathbf{43.43}^{AB}$	$\mathbf{20.44}^{AB}$	$\mathbf{39.83}^{AB}$

We also find that the sequence generation models, i.e., NeuSum and our ARedSum-Seq, do not have clear advantage over other models regardless of their encoder network structure (i.e., Bert or other neural architectures). For instance, SummaRuNNer and Seq2Seq models have the best performance among methods that are not based on Bert[9]. Our ARedSum-Seq perform similarly to BertSumExt. ARedSum-Seq is inferior to ARedSum-Ctx due to its order-sensitive optimization objective. While ARedSum-Ctx learns to optimize towards all the possible ordering of the ground truth sentence set, ARedSum-Seq is optimized towards only one sequence of them. Another ordering of the same set will be penalized by ARedSum-Seq even though they have the same ROUGE score. Its significant worse P@1 (shown in Section 6.2) also confirms this point.

---

[7] `https://github.com/andersjo/pyrouge/tree/master/tools/ROUGE-1.5.5`

[8] The salience ranker of ARedSum-Ctx alone performs similarly to BertSumExt.

[9] The Rouge scores of SummaRuNNer are lower than NeuSum in Zhou et al. (2018) because the results of SummaRuNNer are from the anonymized version CNN/DailyMail, which are not comparable with the results of NeuSum on the *non-anonymized* version.

Table 2: Full-length ROUGE (RG) F1 (%) evaluation on the NYT50 test set. a/b and A/B indicate significant BERTSUMEXT/BERTSUMEXT+TRIBLK with $p < 0.02$ and $p < 0.0001$ respectively.

Model	RG-1	RG-2	RG-L
ORACLE	56.23	37.92	49.45
ORACLE+TRIBLK	54.32	36.33	47.53
Lead3	38.20	19.29	30.49
NN-SE	41.92	22.45	33.88
SEQ2SEQ	44.45	24.72	36.20
SUMMARUNNER	44.70	24.87	36.44
BERTSUMEXT	45.46	**25.53**	37.17
BERTSUMEXT+TRIBLK	44.90	24.87	36.63
AREDSUM-SEQ	45.15^{B}	25.14^{B}	36.79^{B}
AREDSUM-CTX	$\mathbf{45.54}^{AB}$	25.52^{B}	$\mathbf{37.22}^{aB}$

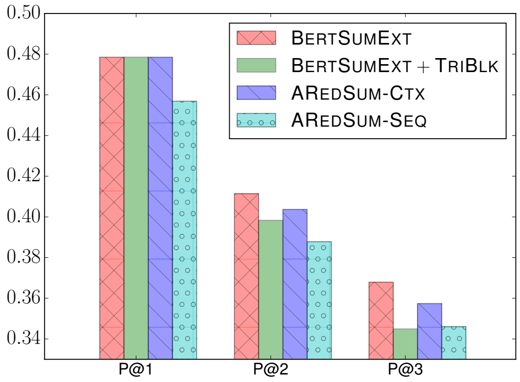


Figure 2: The precision of the extracted sentences at step k on CNN/DailyMail.

NYT50. In contrast to CNN/DailyMail, we observe that TRIBLK has harmed the performance of BERTSUMEXT on NYT50. In fact, as shown in Table 2, applying TRIBLK on ORACLE also causes reduction in ROUGE-1,2,L scores by 1.91, 1.59 and 1.92 absolute point respectively, which are much larger than those drops in CNN/DailyMail (0.94, 0.74 and 0.98). This indicates that TRIBLK filters out more sentences that have high ROUGE gain on NYT50 than CNN/DailyMail, causing more drop of ROUGE. It also shows that sentences in oracle summaries have more trigram overlap on NYT50 than CNN/DailyMail, which implies that redundancy removal on NYT50 may have limited gains and a simple unified rule (TRIBLK) applying on all the documents could harm the performance.

We also observe that AREDSUM-SEQ performs better than BERTSUMEXT+TRIBLK but worse than BERTSUMEXT. In contrast, AREDSUM-CTX achieves higher performance than BERTSUMEXT+TRIBLK and AREDSUM-SEQ by representing redundancy explicitly and controlling its effect dynamically. Since redundancy removal has a limited potential gain on NYT50, the predictions of AREDSUM-CTX differ from BERTSUMEXT only in 10.1% of the test set. However, these differences still lead to significant overall improvements.

Note that the gain of AREDSUM-CTX comes only from redundancy removal, which takes effect from the second step of selecting sentences. The improvements can be larger when redundancy removal has higher potentials (e.g., CNN/DailyMail) and smaller on datasets (e.g., NYT50) with lower potentials. In either case, it does not harm the performances as the other methods, which shows that it is *adaptive* and robust.

6.2 Model Analysis

Precision at Each Step. Since the content of sentences with positive labels, i.e., $\hat{S}^*$, could vary from the original human-generated abstractive summaries S^*, models that have higher precision with respect to $\hat{S}^*$ do not necessarily yield better ROUGE scores against S^*. Because BERTSUMEXT is optimized towards $\hat{S}^*$ while AREDSUM-CTX and AREDSUM-SEQ aim to learn to select sentences with best ROUGE gain against S^*, they behave differently in terms of ROUGE and precision. Thus, we analyze how our model's selection at each k-th step affects the ROUGE performance. We only present the precision on CNN/DailyMail in Figure 2 since similar trends are observed on NYT50. Note all the models except AREDSUM-SEQ have the same $P@1$ because initially, the model's selection is only based on salience. Filtering and demoting the selected sentences starts to take effect only after the second step.

As shown in the figure, BERTSUMEXT has the best $P@1, P@2$ and $P@3$ among all, which is reasonable since $\hat{S}^*$ is the target which it is optimized to learn. When TRIBLK is applied, $P@2$ and $P@3$ drop a lot while the ROUGE scores are up (as in Table 1). It indicates that TRIBLK could filter out some informative but redundant sentences during selection, which harms precision but improves ROUGE. In contrast, $P@2$ and $P@3$ of AREDSUM-CTX is between BERTSUMEXT with and without TRIBLK. Through learning towards ROUGE gain given the previously extracted sentences, AREDSUM-CTX achieves the best ROUGE scores with less harm to precision, which means that AREDSUM-CTX can better balance salience and redundancy.

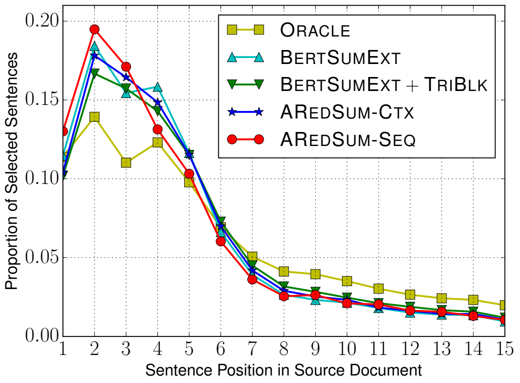


Figure 3: The proportion of extracted sentences in terms of their position in the document.

AREDSUM-SEQ has a significantly lower $P@1$ than the others since its objective at the first step is to find the sentence with maximal ROUGE gain, which is only one in $\hat{S}^*$. At steps 2 and 3, the disadvantage of AREDSUM-SEQ becomes smaller. It has similar $P@3$ to BERTSUMEXT+TRIBLK. The generated sequence of sentences cover a decent portion of $\hat{S}^*$, but it is still worse than the methods that do not use order-sensitive optimization objectives.

Position of Selected Sentences. Figure 3 shows the position of sentences extracted by different models and ORACLE on CNN/DailyMail. A large portion of oracle sentences are the first 5 sentences, and all the models tend to extract the leading 5 sentences in the predicted summaries. The output of AREDSUM-SEQ concentrates more on the first 3 sentences, which differs from ORACLE more than the other models. With TRIBLK, BERTSUMEXT selects sentences in later positions more. The position distribution of AREDSUM-CTX is between BERTSUMEXT with and without TRIBLK, which is similar to their precision distribution in Figure 2. This indicates that AREDSUM-CTX seeks to find a smoother way to filter out sentences that are redundant but salient, and these sentences tend to be at earlier positions.

6.3 Human Evaluation

We also conduct human evaluations to analyze how our best model compares against the best baseline model. On both datasets, we randomly sample 20 summaries constructed by the best baseline and our best model from the cases where their ROUGE-2 score difference is more than 0.05 points. Following Zhou et al. (2018), we asked two graduate student volunteers to rank the summaries extracted by different models from best to worst in terms of *informativeness*, *redundancy* and the *overall* quality. We allowed ties in the analysis. Average ranks of the systems are shown in Table 3.

Table 3: Average ranks of our best method and the best baseline on CNN/DailyMail and NYT50 in terms of informativeness (Info), redundancy (Rdnd) and the overall quality by human participants (the lower, the better). $*$ and $\dagger$ indicates significant improvements with $p < 0.03$ and $p < 0.0001$.

CNN/DailyMail	Info	Rdnd	Overall
BERTSUMEXT + TRIBLK	1.50	1.55	1.55
AREDSUM-CTX	1.20	1.15^*	1.15^*
NYT50	Info	Rdnd	Overall
BERTSUMEXT	1.50	1.60	1.55
AREDSUM-CTX	1.35	$1.00^\dagger$	1.35

On CNN/DailyMail, AREDSUM-CTX ranks higher than BERTSUMEXT+TRIBLK in terms of each aspect. On NYT50, AREDSUM-CTX has a more compelling performance in terms of redundancy than informativeness. This is consistent with the fact that BERTSUMEXT only focuses on learning salience and does not deal with redundancy during sentence selection. From both automatic and human evaluation of our best model and the best baseline, we can see that removing redundancy with our models is better than redundancy removal with heuristics and no redundancy removal.

7 Conclusions

Extending a state-of-the-art extractive summarization model, we propose AREDSUM-SEQ that jointly scores and selects sentences with a sequence generation model and AREDSUM-CTX that learns to balance salience and redundancy with a separate model. Experimental results show that AREDSUM-CTX outperforms AREDSUM-SEQ and all other strong baselines, which yields that redundancy reduction helps improve summary quality, and it is better to model the effect of redundancy explicitly than jointly with salience during sentence scoring.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 135–144. ACM.

Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and Houfeng Wang. 2015. Learning summary prior representation for extractive summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 829–833.

Jaime G Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, volume 98, pages 335–336.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*.

John M Conroy, Judith D Schlesinger, Jade Goldstein, and Dianne P O'leary. 2004. Left-brain/right-brain multi-document summarization. In *Proceedings of the Document Understanding Conference (DUC 2004)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672*.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. *arXiv preprint arXiv:1603.08887*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 364–372. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Chris Kedzie, Kathleen McKeown, and Hal Daume III. 2018. Content selection in deep learning models of summarization. *arXiv preprint arXiv:1810.12343*.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1999. A trainable document summarizer. *Advances in Automatic Summarization*, pages 55–60.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 510–520. Association for Computational Linguistics.

Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *European Conference on Information Retrieval*, pages 557–564. Springer.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*.

Ani Nenkova, Lucy Vanderwende, and Kathleen McKeown. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 573–580. ACM.

Miles Osborne. 2002. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 1–8. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Jun Ma, and Maarten de Rijke. 2017. Leveraging contextual sentence relations for extractive summarization using a neural attention model. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104.

Pengjie Ren, Furu Wei, Zhumin Chen, Jun Ma, and Ming Zhou. 2016. A redundancy-aware sentence regression framework for extractive summarization. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 33–43, Osaka, Japan.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xiaojun Wan and Jianwu Yang. 2006. Improved affinity graph based multi-document summarization. In *Proceedings of the human language technology conference of the NAACL, Companion volume: Short papers*, pages 181–184. Association for Computational Linguistics.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. *arXiv preprint arXiv:1907.03491*.

Qingyu Zhou, Furu Wei, and Ming Zhou. 2020. At which level should we extract? an empirical study on extractive document summarization. *arXiv preprint arXiv:2004.02664*.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. *arXiv preprint arXiv:1807.02305*.

“Talk to me with left, right, and angles”: Lexical entrainment in spoken Hebrew dialogue

Andreas Weise[1], **Vered Silber-Varod**[2], **Anat Lerner**[2,3], **Julia Hirschberg**[4], **Rivka Levitan**[1,5]
[1] Department of Computer Science, The Graduate Center, CUNY, New York, NY 10016, USA
[2] Open Media and Information Lab (OMILab), The Open University of Israel, Raanana 43107, Israel
[3] Mathematics and Computer Science Department, The Open University of Israel, Raanana 43107, Israel
[4] Department of Computer Science, Columbia University, New York, NY 10027, USA
[5] Department of Computer and Information Science, Brooklyn College, CUNY, Brooklyn, NY 11210, USA
aweise@gradcenter.cuny.edu, vereds@openu.ac.il, anat@openu.ac.il, julia@cs.columbia.edu, rlevitan@brooklyn.cuny.edu

Abstract

It has been well-documented for several languages that human interlocutors tend to adapt their linguistic productions to become more similar to each other. This behavior, known as entrainment, affects lexical choice as well, both with regard to specific words, such as referring expressions, and overall style. We offer what we believe to be the first investigation of such lexical entrainment in Hebrew. Using two existing measures, we analyze Hebrew speakers interacting in a Map Task, a popular experimental setup, and find rich evidence of lexical entrainment. Analyzing speaker pairs by the combination of their genders as well as speakers by their individual gender, we find no clear pattern of differences. We do, however, find that speakers in a position of less power entrain more than those with greater power, which matches theoretical accounts. Overall, our results mostly accord with those for American English, with a lack of entrainment on hedge words being the main difference.

1 Introduction

Entrainment, also known as accommodation or alignment, is a widespread phenomenon in human interaction which leads interlocutors to adapt to each other to become more similar. It has been found for a variety of linguistic dimensions, including prosody (Levitan and Hirschberg, 2011), phonetics (Pardo, 2006), syntax (Reitter et al., 2006), and lexical choice (Brennan and Clark, 1996).

Lexical entrainment has been studied for several types of lexical choices from specific sets of words – such as referring expressions (Brennan and Clark, 1996), high-frequency words and task-related words (Rahimi et al., 2017), as well as hedge and cue phrases (Levitan et al., 2018) – to the wider linguistic style (Niederhoffer and Pennebaker, 2002). This motivates us to consider both specific word sets and overall language use here.

Importantly, there are correlations between lexical entrainment and interesting aspects of the conversation. These include task success for both speaker pairs (Reitter and Moore, 2007; Nenkova et al., 2008) and groups (Gonzales et al., 2010; Friedberg et al., 2012), conversation flow and perceived naturalness (Nenkova et al., 2008), as well as power differences between the speakers (Danescu-Niculescu-Mizil et al., 2011). This suggests practical applications and has led to the development of entraining natural language generators in Dutch (De Jong et al., 2008), German (Buschmeier et al., 2009), and American English and European Portuguese (Lopes et al., 2015), among others.

To the best of our knowledge, there has not been any systematic research on lexical entrainment in Hebrew or any other Semitic Language. Previous studies analyzing lexical choice in Semitic Languages focus on borrowing and code-switching, for instance between Arabic and English (Abu-Melhim et al., 2016) and Arabic and Hebrew (Hawker, 2018). Given the important social role of entrainment and its potential applications, our study provides an important contribution by presenting the first analysis of lexical entrainment in Hebrew. This helps identify variations in how the behavior manifests in different linguistic and cultural contexts. We note that in a recently published study (Weise et al., 2020), we analyzed acoustic-prosodic entrainment in Hebrew for the same data. Together, these two papers provide a broad investigation of entrainment for this novel language context.

2 Corpus

In this study, we analyze the Open University of Israel Map Task Corpus (MaTaCOp) (Azogui et al., 2016) of dyadic, Hebrew conversations, modeled after the HCRC Map Task Corpus (Anderson et al., 1991). Each participant was given a map with la-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 292–299
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

beled landmarks, some of them shared with the partner's map, some unique. The map of one participant in a pair, the *leader*, contained a path among the landmarks. It was their task to describe the path so their partner, the *follower*, could reproduce it. All speaker pairs discussed the same two pairs of corresponding maps, with either speaker acting as a leader for one map and as a follower for the other. We refer to whole conversations as *sessions* and to each of the two parts as *tasks*.

MaTaCOp contains about six hours of conversations between 32 speakers, all of them fluent in Hebrew. There are six female, six mixed, and four male pairs. Most of the paired speakers were acquainted prior to the experiment. We analyze the influence of this aspect of our data in Section 5.7. Further details on the level of familiarity is provided in Appendix B. For more details on the corpus in general, see Weise et al. (2020).

3 Transcription, Tokenization, and Lemmatization

MaTaCOp is fully transcribed. The phoneme set consists of the five vowels [i, a, e, o, u] and 21 consonants of Modern Hebrew. The pharyngeal [ʕ] and the glottal [ʔ] are not represented. The phonetic representation removes ambiguity that occurs in Hebrew orthography. For example, the grapheme למה is represented with two different transcriptions, *le-ma* "what for?" and *lama* "why?". In this paper, we use Romanization of Hebrew to transliterate Hebrew words.

Tokenization, on the other hand, generally follows standard Hebrew orthography. For instance, proclitics (such as *mi-* "from") were transcribed attached to the subsequent word (e.g., *mi-nekuda* "from point"). However, in case a silent pause or other disfluency occurred between a clitic and the subsequent word, the clitic was transcribed separately, as in *mi- nekuda* "from point". In total, this yields 50075 tokens for the corpus.

Due to Hebrew's rich morphology, many of the words in our corpus appear in a variety of grammatical forms, such as *agol* "round.M.SG", *agula* "round.F.SG", *agul-im* "round.M.PL", and *ha-agol* "the-round". We use a manually created list of grammatical forms for each lemma to lemmatize and count occurrences per lemma.[1] Overall, there are 1,038 lemmas and 2,179 other grammatical forms.

4 Lexical Entrainment Measures

We measure the lexical similarity of speakers' utterances per session (or task, where noted) using two previously established measures, one for a specific set of words W, one for the overall productions.

Per word $w \in W$ and per speaker S, the first measure determines $cnt_S(w)$, the number of times w was uttered by S, and ttl_S, the total number of words uttered by S. Similarity between a pair of speakers S_1, S_2 is then defined based on the absolute difference of the fractions per word, as

$$sim_1(S_1, S_2) = -\sum_{w \in W} \left| \frac{cnt_{S_1}(w)}{ttl_{S_1}} - \frac{cnt_{S_2}(w)}{ttl_{S_2}} \right| .$$

Nenkova et al. (2008) proposed this measure for high-frequency words. Note that it is symmetric.

The second measure was originally proposed by Gravano et al. (2014) to compare tones and break indices (ToBI). For it, we construct a trigram language model for each speaker from their utterances, using SRILM (Stolcke, 2002). The measure $sim_2(S_1, S_2)$ is then defined as the negated perplexity of using the language model for S_1 to predict all utterances of S_2, computed with SRILM. Low perplexity indicates that the model for S_1 is a good representation of the utterances of S_2. In this case, the phrases used by S_2 are essentially a subset of those used by S_1. We interpret this as entrainment *of* S_1 *towards* S_2 as it signals that S_1 incorporated S_2's phrases into their own. Conversely, high perplexity indicates a lack of entrainment. This is why we use negated perplexity for sim_2. Note that this measure is asymmetric. For a symmetric version, we simply add the asymmetric values for both directions, following Weise and Levitan (2018).

To determine whether significant entrainment is present, we follow Levitan et al. (2012). Each similarity value $sim_i(S_1, S_2)$ for a speaker S_1 with their partner S_2 is compared with the weighted average similarity of S_1 with non-partners, using paired Student's t-tests. Non-partners must have the same gender as S_2 and their partners must have the same gender as S_1. For similarity per task, non-partners must also be talking about the same map and have the same role as S_2. Non-partners are weighted by how closely their language model's entropy, computed using SRILM, matches that of the actual partner (absolute differences). This is meant to account for the effect that the richness of a speaker's lexical inventory has on our measures and follows Weise and Levitan (2018).

---

[1] All word lists we use here can be downloaded at openu.ac.il/en/academicstudies/matacop/pages/default.aspx.

5 Results

5.1 Entrainment on most frequent lemmas

Following Nenkova et al. (2008), we first use sim_1 to check whether speakers in our corpus entrain on its 25 most frequent lemmas (excluding 56 lemmas representing landmark labels and the directional terms in Section 5.2). We find that speakers do significantly entrain on these lemmas ($t(15) = 4.15, p = 8.54e-04$). That is, the distributions of the 25 most frequent lemmas show greater similarity between partners than with non-partners. This effect also approaches significance for just the female pairs ($t(5) = 2.83, p = 0.037$) and male pairs ($t(3) = 3.14, p = 0.052$), but not for mixed pairs ($t(5) = 1.51, p = 0.19$).[2] Table 1 summarizes these results and those for the following subsections. We also use independent Student's t-tests to conduct direct comparisons between the similarity values for the gender pairs, i.e., female vs. male, female vs. mixed, and male vs. mixed.[3] This yields no significant differences and no difference even approaches significance (lowest $p = 0.53$).

5.2 Entrainment on directional terms

Leaders and followers in our corpus use various directional terms to communicate the path among the landmarks. To assess whether they adopt each other's terminology, we follow Silber-Varod et al. (2020) and consider ten different terms of two basic types. This includes the directions of a compass – i.e., *ṣafon* "north", *darom* "south", *maarav* "west", and *mizraḥ* "east" – and relative directions – i.e., *le-mal-a* "upwards", *me-al* "above", *le-mat-a* "downwards", *mi-taḥat* "below", *smol* "left", and *yamin* "right". We treat the lemmas of these ten terms as a set W for measure sim_1 and count all occurrences for all grammatical forms per lemma.

Using this approach, we find significant evidence of entrainment on these ten directional terms overall ($t(15) = 6.64, p =$ 7.86e-06) as well as for female pairs ($t(5) = 5.75, p = 0.0022$), male pairs ($t(3) = 4.42, p = 0.022$), and mixed pairs ($t(5) = 4.85, p = 0.0047$) separately. Again, no difference between the gender pairs even approaches significance (lowest $p = 0.086$).

5.3 Entrainment on geometric terms

In addition to directional terms, speakers employ a variety of geometric terms to describe the shape of the path, the locations of the landmarks, and their relation to each other. This includes, for example, *malben* "rectangle" and *b-a-hitstalvut* "at the intersection". To determine whether speakers entrain on these, we consider a list of 34 lemmas (with a total of 199 grammatical forms) of such terms as another set W for measure sim_1. This yields significant results overall ($t(15) = 4.82, p = 2.2e-04$) as well as for female ($t(5) = 5.08, p = 0.0038$) and mixed pairs ($t(5) = 6.62, p = 0.0012$), but not for male pairs ($t(3) = 1.00, p = 0.39$). Once again, none of the differences between gender pairs even approach significance (lowest $p = 0.72$).

5.4 Entrainment on hedge words

The difficulty of describing irregular path shapes in the Map Task, along with incomplete information about the landmarks, creates uncertainty for the speakers which encourages the use of hedge words. Furthermore, in their corpus of deceptive interviews, Levitan et al. (2018) found the strongest evidence of lexical entrainment for hedge words, stronger than for the 25 most frequent words. These observations motivate us to analyze hedge words as well, using a translated version of the same list Levitan et al. used (with 37 lemmas and 78 grammatical forms total). However, we find no significant entrainment, neither overall ($t(15) = 1.61, p = 0.13$) nor for any of the gender pairs (lowest $p = 0.14$).

5.5 Entrainment on imperative verb forms

The different roles in the Map Task facilitate the use of imperative verb forms. Leaders might command followers to draw a path a certain way, while followers might demand information or a different way of describing, as in the utterance we quoted in the title. Of course, they can achieve the same communicative goals with phrases that avoid imperatives, using, for example, nonverbal predicates or standard infinitival clauses such as *az at tsrix-a em laredet mi-tsad smol la-xanut* "so you have to um to get down from the left side of the store". This flexibility allows for entrainment.

However, note that the different roles actually make it unlikely for speakers to use the *same* verbs. A leader might instruct a follower to "*draw* the path

---

[2] To account for multiple testing, we regard these four tests as a "family" and treat results up to the k-th smallest p-value p_k as significant at level $\alpha = 0.05$, where k is the largest integer such that $p_k \leq \frac{k}{m}\alpha$, with m being the size of the family (Benjamini and Hochberg, 1995). We do the same for each analogous group of four tests for other entrainment targets in the following subsections.

[3] We again account for multiple testing by treating these three tests as a family here and in the following subsections.

Entrainment target	Measure	Significant Overall	FF	MM	FM
25 most frequent words	sim_1	**	(*)	(*)	
directional terms	sim_1	***	**	*	**
geometric terms	sim_1	**	**		**
hedge words	sim_1				
imperative verb forms	sim_1				
overall productions	sim_2	*			

Table 1: Results per entrainment target and measure, overall and per gender pair (female, male, mixed) with significance level (***: $\alpha < 0.001$, **: $\alpha < 0.01$, *: $\alpha < 0.05$, (*): $\alpha < 0.1$) per family (see Footnote 2). Direct comparisons between gender pairs do not show significant differences for any entrainment target.

around the lake", while the follower might demand "*tell* me how close". Therefore, we check whether speakers adopt an imperative mode of speaking from each other, regardless of individual verbs.

We identified a list of 122 imperative verb forms[4] in our corpus and determine what fraction of each speaker's words this list represents. That is, W for sim_1 consists of only one placeholder "word". Using this method, we find no significant entrainment, neither overall ($t(15) = 1.03, p = 0.30$), nor for any gender pair (lowest $p = 0.15$).

5.6 Entrainment on overall productions

Lastly, we use sim_2 to check whether speakers entrain on their partners' overall language use, i.e., whether they model their partners' productions better than those of other speakers. We find that this is the case overall ($t(15) = 3.09, p = 0.0074$) but neither for female pairs ($t(5) = 1.44, p = 0.21$), nor for male pairs ($t(3) = 2.72, p = 0.073$), nor for mixed pairs ($t(5) = 2.20, p = 0.08$). Once again, we find no significant differences between the gender pairs in direct comparisons (lowest $p = 0.43$).

Since sim_2 is asymmetric, we can use it to compare the entrainment behavior of individual speakers based on their gender and role, respectively, with independent Student's t-tests. This yields no significant difference between female and male speakers ($t(30) = 1.06, p = 0.30$).

In order to compare speakers based on their roles, we measure at the task level with separate language models and predictions of all utterances of a task instead of a whole session. Doing so yields a highly significant difference, with followers entraining more than leaders ($t(62) = 5.52, p = 6.95\text{e-}07$). Of course, leaders speak significantly more than followers ($t(62) = 5.04, p = 4.25\text{e-}05$), which might explain why their productions are supersets of those of the followers. However, the difference remains significant even when normalizing the measure by the number of words spoken ($t(62) = 3.22, p = 0.0020$).

5.7 Influence of familiarity

Prior acquaintance between subjects, as in our data, is unusual in entrainment research and introduces a confound to our comparison with other studies. We conduct some additional analysis of this here and discuss it further in Section 6.

For this analysis, we consider speaker pairs in two groups, of "high" (11 pairs) and "low" (5 pairs) familiarity.[5] For each entrainment target, we compare the similarity values for the two groups with independent Student's t-tests. This does yield a significant difference for entrainment on overall productions ($t(14) = 3.31, p = 0.0051$), but not for any other entrainment target ($0.05 < p < 0.83$). That is, speakers who were already well-acquainted before participating in the experiment, show greater entrainment in their overall language use (and only that) than those with little or no acquaintance.

6 Discussion

In this first analysis of lexical entrainment in Hebrew, using two existing measures, we find substantial evidence of entrainment both on specific groups of words and overall language use.

Speakers entrained on the 25 most frequent lemmas in the corpus, a result that matches findings on

---

[4] Including grammatical imperatives (e.g., *lex* "go.M") and 2nd person prefix conjugation (e.g., *te-lex* "go.M") but excluding the reduced future forms (Bat-El, 2002) which are more ambiguous and inconsistent.

[5] For further details, see Appendix B.

English corpora of telephone conversations (Weise and Levitan, 2018), deceptive interviews (Levitan et al., 2018), and task-oriented, multi-party interactions (Rahimi et al., 2017).

The broadest and most significant evidence of entrainment we find is for directional terms and the geometric terms to describe the path. In fact, in some cases speakers actively requested entrainment, as in: *a azov et ha-sinus-im daber iti be-smol-a yemin-a ve-be-zaviy-ot* "uh leave the sines, talk to me with to-the-left, to-the-right, and with angles". Our results match previous ones for referring expressions (Brennan and Clark, 1996) and "project words" (Rahimi et al., 2017) in English.

Contrary to Levitan et al. (2018), who found the strongest evidence of lexical entrainment for hedge words, we find no entrainment for these. This may be because Hebrew speech patterns tend to be very "direct" (Katriel, 2004, ch.2), more so than English ones (Van Dijk, 1997, p.235), so hedges might be culturally less appropriate.

We do not find that speakers entrain on an imperative mode of speaking. This may be due to data sparsity, though, as imperatives constitute only 1.3% of all tokens (see Appendix A) despite the experimental setting facilitating their use. Studies of syntactic alignment, e.g., by Reitter et al. (2006), have found that English speakers adopt syntactic choices from their interlocutors. A broader investigation of this is needed for Hebrew.

Our results for entrainment on overall productions – how well speakers' language models fit their interlocutors' productions – match prior results for English. Weise and Levitan (2018) found this measure to be significant for both task-oriented dialogs and telephone conversations. But unlike us, they found the results for this measure to be more significant than those for the 25 most frequent terms.

Our results reveal no clear pattern of differences between the gender pairs. The number of entrainment targets and significance levels for female, male, and mixed pairs are comparable (marginally weaker results for male pairs might be partially attributable to a smaller sample size). Direct comparisons between the gender pairs also do not reveal any significant differences for any of our measures. Neither does the comparison between individual speakers based on their gender, using the asymmetric version of our measure for overall productions. Similar analyses for acoustic entrainment in English have sometimes found differences based on speaker gender (Levitan et al., 2012) and sometimes not (Pardo et al., 2018; Weise et al., 2019). In our own analysis of acoustic entrainment in the same Hebrew corpus (Weise et al., 2020), we also found no difference based on speaker gender. The only study of the effect of gender on lexical entrainment we are aware of was for human-robot interactions and found that female speakers exhibited a greater degree of entrainment to the robot interlocutor than males did (Kimoto et al., 2017).

Speakers in subordinate roles are predicted to entrain more than those in power (Giles et al., 1991). This has been confirmed for lexical entrainment in English (Danescu-Niculescu-Mizil et al., 2011) and we find the same here. Followers, having less power due to their dependency on information from the leaders, entrain more than leaders with regard to their overall productions. Conversely, for directional terms alone, Silber-Varod et al. (2020) found that followers had greater influence on the terminology, that is, leaders adopted followers' terms more often than vice versa.

It is worth repeating that speakers in our corpus were acquainted prior to their participation in the experiment. There is little prior research on the impact of this factor. For acoustic-prosodic entrainment, Truong and Heylen (2012) find that unacquainted speakers exhibit more entrainment while Cabarrão et al. (2016) report an example with the opposite trend. The analysis of our own data indicates that familiarity has at least *some* influence, specifically for entrainment on overall productions. However, for hedge words the difference between high and low familiarity pairs is so insignificant ($t(14) = 0.50, p = 0.62$) that we do not believe familiarity explains the difference between our results and those for unacquainted English speakers.

Overall, we find that lexical entrainment in our Hebrew corpus is very much comparable to prior results for English. The only notable difference is the lack of entrainment on hedge words which, as we noted above, may be due to cultural differences. Future research should investigate additional conversational settings in Hebrew, including with unacquainted speakers.

Acknowledgments

This work was supported by the Open Media and Information Lab (OMILab) at The Open University of Israel [Grant No. 20184] and by the National Science Foundation [Grant No. 1845710].

References

Abdel-Rahman H Abu-Melhim, Mohammad H Abood, and Taysir M Shehadeh. 2016. Interpreting types and functions of Arabic-English code-switching in light of modern principles of social psychology. *British Journal of Humanities and Social Sciences*, 15(1).

Anne H Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. 1991. The HCRC map task corpus. *Language and speech*, 34(4):351–366.

Jacob Azogui, Anat Lerner, and Vered Silber-Varod. 2016. The Open University of Israel Map Task Corpus (MaTaCOp). `https://www.openu.ac.il/en/academicstudies/matacop/`.

Outi Bat-El. 2002. True truncation in colloquial hebrew imperatives. *Language*, pages 651–683.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B*, 57(1):289–300.

Susan E. Brennan and Herbert H. Clark. 1996. Conceptual pacts and lexical choice in conversation. *Journal of experimental psychology: Learning, memory, and cognition*, 22(6):1482–1493.

Hendrik Buschmeier, Kirsten Bergmann, and Stefan Kopp. 2009. An Alignment-capable Microplanner for Natural Language Generation. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 82–89.

Vera Cabarrão, Isabel Trancoso, Ana Isabel Mata, Helena Moniz, and Fernando Batista. 2016. Global analysis of entrainment in dialogues. In *International Conference on Advances in Speech and Language Technologies for Iberian Languages*, pages 215–223. Springer.

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. 2011. Mark My Words! Linguistic Style Accommodation in Social Media. In *WWW*, pages 745–754.

Markus De Jong, Mariët Theune, and Dennis Hofs. 2008. Politeness and alignment in dialogues with a virtual guide. In *AAMAS '08: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, pages 207–214.

Heather Friedberg, Diane Litman, and Susannah B F Paletz. 2012. Lexical entrainment and success in student engineering groups. In *Spoken Language Technology (SLT), 2012 IEEE Workshop on*, pages 404–409.

Howard Giles, Nikolas Coupland, and Justine Coupland. 1991. Accommodation theory: Communication, context, and consequence. In *Contexts of accommodation: Developments in applied sociolinguistics*, pages 1–68. Cambridge University Press.

Amy L. Gonzales, Jeffrey T. Hancock, and James W. Pennebaker. 2010. Language Style Matching as a Predictor of Social Dynamics in Small Groups. *Communication Research*, 37(1):3–19.

Agustín Gravano, Štefan Beňuš, Rivka Levitan, and Julia Hirschberg. 2014. Three ToBI-based measures of prosodic entrainment and their correlations with speaker engagement. In *Spoken Language Technology (SLT), 2014 IEEE Workshop on*, pages 578–583.

Nancy Hawker. 2018. The mirage of Arabrew: Ideologies for understanding Arabic-Hebrew contact. *Language in Society*, 47(2):219–244.

Tamar Katriel. 2004. *Dialogic moments: From soul talks to talk radio in Israeli culture*. Wayne State University Press.

Mitsuhiko Kimoto, Takamasa Iio, Masahiro Shiomi, Ivan Tanev, Katsunori Shimohara, and Norihiro Hagita. 2017. Gender effects on lexical alignment in human-robot interaction. *IEEJ Transactions on Electronics, Information and Systems*, 137(12):1625–1632.

Rivka Levitan and Julia Hirschberg. 2011. Measuring acoustic-prosodic entrainment with respect to multiple levels and dimensions. In *INTERSPEECH 2011*, pages 3081–3084.

Rivka Levitan, Laura Willson, Agustín Gravano, Štefan Beňuš, Julia Hirschberg, and Ani Nenkova. 2012. Acoustic-Prosodic Entrainment and Social Behavior. In *NAACL HLT*, pages 11–19.

Sarah Ita Levitan, Jessica Xiang, and Julia Hirschberg. 2018. Acoustic-Prosodic and Lexical Entrainment in Deceptive Dialogue. In *Speech Prosody*, pages 532–536.

José Lopes, Maxine Eskenazi, and Isabel Trancoso. 2015. From rule-based to data-driven lexical entrainment models in spoken dialog systems. *Computer Speech and Language*, 31(1):87–112.

Ani Nenkova, Agustín Gravano, and Julia Hirschberg. 2008. High Frequency Word Entrainment in Spoken Dialogue. In *ACL HLT*, pages 169–172.

Kate G. Niederhoffer and James W. Pennebaker. 2002. Linguistic Style Matching in Social Interaction. *Journal of Language and Social Psychology*, 21(4):337–360.

Jennifer S. Pardo. 2006. On phonetic convergence during conversational interaction. *The Journal of the Acoustical Society of America*, 119(4):2382–2393.

Jennifer S Pardo, Adelya Urmanche, Sherilyn Wilman, Jaclyn Wiener, Nicholas Mason, Keagan Francis, and Melanie Ward. 2018. A comparison of phonetic convergence in conversational interaction and speech shadowing. *Journal of Phonetics*, 69:1–11.

Zahra Rahimi, Anish Kumar, Diane Litman, Susannah Paletz, and Mingzhi Yu. 2017. Entrainment in Multi-Party Spoken Dialogues at Multiple Linguistic Levels. *Interspeech 2017*, pages 1696–1700.

David Reitter and Johanna D. Moore. 2007. Predicting Success in Dialogue. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 808–815.

David Reitter, Johanna D. Moore, and Frank Keller. 2006. Priming of Syntactic Rules in Task-Oriented Dialogue and Spontaneous Conversation. In *CogSci*, pages 685–690.

Vered Silber-Varod, Sarit Malayev, and Anat Lerner. 2020. Positioning oneself in different roles: Structural and lexical measures of power relations between speakers in map task corpus. *Speech Communication*, 117:1 – 12.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *ICSLP*, pages 901–904.

Khiet P Truong and Dirk Heylen. 2012. Measuring prosodic alignment in cooperative task-based conversations. In *Interspeech 2012*, pages 843–846.

Teun A. Van Dijk. 1997. *Discourse as social interaction*, volume 2. Sage.

Andreas Weise and Rivka Levitan. 2018. Looking for structure in lexical and acoustic-prosodic entrainment behaviors. In *NAACL HLT*, pages 297–302.

Andreas Weise, Sarah Ita Levitan, Julia Hirschberg, and Rivka Levitan. 2019. Individual differences in acoustic-prosodic entrainment in spoken dialogue. *Speech Communication*, 115(April):78–87.

Andreas Weise, Vered Silber-Varod, Anat Lerner, Julia Hirschberg, and Rivka Levitan. 2020. Entrainment in spoken Hebrew dialogues. *Journal of Phonetics*, 83(11):1–16.

A Percentages of words per list

This paper considers a variety of different word lists for entrainment measure sim_1 (see Section 4). These lists represent different percentages of all the words uttered by various speaker groups in the corpus, as detailed in Table 2. We include them here so they may be used to interpret our results, by themselves or in comparison with other corpora. We note, for instance, that imperative verb forms are comparatively rare, which might partially explain the lack of significant entrainment we found. It also suggests differential use of the word lists by the speaker groups. For instance, as might be expected, the percentage of words that are imperative verb forms is more than twice as high for leaders (1.6%) as for followers (0.7%).

B Session details and raw similarities

Table 3 provides an overview of the sessions, i.e., speaker pairs, in our corpus and their similarity as measured by sim_1 and sim_2 (see Section 4).

Details for the sessions include the gender pair (female, male, mixed) and the level of familiarity between the interlocutors. Familiarity was categorized into two groups. Most speaker pairs were highly acquainted, through marriage (two pairs), prior service in the same military unit (three pairs), or work in the same department (six pairs). The remaining pairs had a low level of acquaintance through work in the same institution with little interaction (four pairs) or no acquaintance at all (one pair).

For each session and entrainment target, the table lists the respective similarity value as well as the baseline similarity derived from the average similarity with non-partners. All values are negative, with values closer to zero indicating greater similarity (see Section 4).

Word list					**Speaker group**				
	All	Female	Male	Leaders	Followers	Female leaders	Female followers	Male leaders	Male followers
25 most frequent words	33.8%	34.1%	33.4%	31.2%	38.8%	31.3%	39.4%	31.1%	38.2%
directional terms	8.8%	9.1%	8.5%	9.0%	8.5%	9.5%	8.5%	8.6%	8.4%
geometric terms	4.5%	5.1%	3.8%	4.8%	3.7%	5.7%	4.1%	4.0%	3.4%
hedge words	3.0%	3.1%	3.0%	3.3%	2.5%	3.5%	2.3%	3.1%	2.7%
imperative verb forms	1.3%	1.2%	1.3%	1.6%	0.7%	1.5%	0.8%	1.6%	0.7%

Table 2: Percentages of all words, uttered by various groups of speakers, that are represented by the different word lists considered as entrainment targets. For example, 9.5% of all the words uttered by female leaders were directional terms. Note that the columns do not add up to 100% since there are many words that are not included in any of our lists.

					Entrainment target			
Session ID	**Gender pair**	**Level of familiarity**	25 most frequent words	directional terms	geometric terms	hedge words	imperative verb forms	overall productions
0	MM	high (military)	-0.200 (-0.238)	-0.038 (-0.062)	-0.014 (-0.027)	-0.029 (-0.038)	-0.024 (-0.014)	-280 (-1135)
1	MM	high (military)	-0.197 (-0.218)	-0.042 (-0.078)	-0.027 (-0.028)	-0.027 (-0.028)	-0.005 (-0.012)	-281 (-835)
2	FM	high (military)	-0.140 (-0.194)	-0.023 (-0.050)	-0.026 (-0.032)	-0.026 (-0.029)	-0.009 (-0.006)	-220 (-259)
3	FM	high (married)	-0.159 (-0.185)	-0.013 (-0.041)	-0.027 (-0.036)	-0.051 (-0.037)	-0.009 (-0.005)	-285 (-456)
4	FM	high (department)	-0.222 (-0.209)	-0.057 (-0.059)	-0.016 (-0.034)	-0.018 (-0.027)	-0.010 (-0.006)	-367 (-1089)
5	FF	high (department)	-0.133 (-0.217)	-0.067 (-0.093)	-0.012 (-0.041)	-0.025 (-0.027)	-0.005 (-0.012)	-237 (-417)
6	FF	low (institution)	-0.176 (-0.207)	-0.051 (-0.060)	-0.038 (-0.051)	-0.041 (-0.033)	-0.004 (-0.012)	-426 (-383)
7	FM	high (department)	-0.191 (-0.167)	-0.026 (-0.046)	-0.025 (-0.037)	-0.018 (-0.026)	-0.003 (-0.006)	-317 (-310)
8	FF	high (department)	-0.272 (-0.279)	-0.070 (-0.112)	-0.013 (-0.059)	-0.019 (-0.031)	-0.025 (-0.024)	-240 (-2113)
9	FF	high (department)	-0.201 (-0.206)	-0.035 (-0.058)	-0.013 (-0.051)	-0.042 (-0.033)	-0.002 (-0.011)	-303 (-599)
10	MM	high (department)	-0.192 (-0.211)	-0.049 (-0.122)	-0.037 (-0.033)	-0.016 (-0.028)	-0.001 (-0.007)	-226 (-967)
11	FM	low (institution)	-0.148 (-0.182)	-0.032 (-0.047)	-0.016 (-0.033)	-0.022 (-0.030)	-9.4e-5 (-0.007)	-295 (-495)
12	FF	low (none)	-0.173 (-0.200)	-0.031 (-0.056)	-0.045 (-0.060)	-0.026 (-0.028)	-2.4e-4 (-0.011)	-358 (-468)
13	FM	high (married)	-0.162 (-0.198)	-0.026 (-0.046)	-0.036 (-0.050)	-0.020 (-0.027)	-0.006 (-0.008)	-207 (-490)
14	FF	low (institution)	-0.159 (-0.225)	-0.031 (-0.063)	-0.040 (-0.061)	-0.024 (-0.032)	-0.020 (-0.013)	-318 (-444)
15	MM	low (institution)	-0.136 (-0.206)	-0.019 (-0.073)	-0.028 (-0.031)	-0.033 (-0.033)	-0.016 (-0.011)	-399 (-373)

Table 3: Raw similarity values per session and entrainment target. Baseline values derived from non-partner pairings are listed in parentheses.

Recipes for Building an Open-Domain Chatbot

Stephen Roller **Emily Dinan** **Naman Goyal** **Da Ju**
Mary Williamson **Yinhan Liu**[*] **Jing Xu** **Myle Ott**
Kurt Shuster **Eric M. Smith** **Y-Lan Boureau** **Jason Weston**

Facebook AI Research
New York, NY
{roller, edinan, namangoyal, daju, marywilliamson,
jingxu23, myleott, kshuster, ems, ylan, jase}@fb.com

Abstract

Building open-domain chatbots is a challenging area for machine learning research. While prior work has shown that scaling neural models in the number of parameters and the size of the data they are trained on gives improved results, we highlight other ingredients. Good conversation requires blended skills: providing engaging talking points, and displaying knowledge, empathy and personality appropriately, while maintaining a consistent persona. We show that large scale models can learn these skills when given appropriate training data and choice of generation strategy. We build variants of these recipes with 90M, 2.7B and 9.4B parameter models, and make our models and code publicly available. Human evaluations show our best models outperform existing approaches in multi-turn dialogue on engagingness and humanness measurements. We then discuss the limitations of this work by analyzing failure cases of our models.

1 Introduction

This work provides recipes for building open-domain chatbots that perform well, i.e., to study and identify which methods work well together in order to maximize human evaluations. It has been shown across the field of NLP (Devlin et al., 2019) and in conversational agents in particular (Dinan et al., 2020; Zhang et al., 2019; Adiwardana et al., 2020) that pre-training on large corpora is important. Beyond simply scaling models, the two main takeaways from our study are that: (1) Large improvements can be made by fine-tuning on data that emphasizes desirable conversational skills. We select tasks that make the model focus on personality and engagingness, knowledge, and empathy, achieving large gains by using the recently introduced Blended Skill Talk (BST) set-up (Smith et al., 2020), which targets those aspects by providing training data and initial conversational context (personas and topics). Small models using BST can match or outperform larger models that do not. (2) Two models with the same perplexity but different decoding algorithms can give vastly different results, e.g. we show that the length of the bot's utterances are crucial to human judgments of quality. We show, contrary to previous work which reports that beam search is inferior to sampling (Holtzman et al., 2019; Adiwardana et al., 2020), that careful choice of search hyperparameters can give strong results, e.g., constraining the minimum beam length gives a crucial control of the bland versus spicy spectrum of responses.

In human evaluations our best model outperforms both DialogGPT (Zhang et al., 2019) and the state of the art Meena chatbot (Adiwardana et al., 2020), the latter in a pairwise comparison 75% to 25% in terms of engagingness, and by 65% to 35% in terms of humanness (both statistically significant, two-tailed binomial test, $p < 0.01$).

While the performance of our bot at first sight is very good, we do not believe we are yet close to solving the problem of open-domain conversation. We thus discuss limitations of our models, and initial attempts to solve them. In particular, our models still display: a lack of in-depth knowledge if sufficiently interrogated; a tendency to stick to simpler language; and a tendency to repeat oft-used phrases. We show how unlikelihood training (Welleck et al., 2020) and retrieve-and-refine (Weston et al., 2018) mechanisms are potential avenues for fixing these problems; however, our initial experiments with these methods are inconclusive. We thus discuss future possibilities for alleviating these problems, as well as methods for evaluating them.

Finally, we believe releasing models is the most reliable way to enable full insight into their capabilities. We thus make publicly available our

---

[*] Work done while at Facebook; currently AI2 Incubator.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 300–325
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

large-scale, state of the art open-domain conversational agent, including code to fine-tune it, the model weights, and code to evaluate it, so that our setup is reproducible.

2 Related Work

The area of open-domain dialogue has made significant progress recently with end-to-end neural approaches. The ConvAI2 competition at NeurIPS 2018 featured large pre-trained Transformers for the top two winning teams (Dinan et al., 2020). In particular, Wolf et al. (2019) pre-trained via the method of Radford et al. (2018) using the BooksCorpus dataset, resulting in the best perplexities and F1 scores. Since then, results have improved further with the advent of larger, improved pre-training (Lewis et al., 2019; Shuster et al., 2019). In general this extends beyond ConvAI2 to many open-domain dialogue datasets, such as daily dialogue and Cornell Movies (He et al., 2019), and also when multi-tasking across many of these datasets, as we also do here (Shuster et al., 2019; Smith et al., 2020).

A particular large-scale model of note that we compare to in this work is Meena (Adiwardana et al., 2020), a 2.6B parameter Transformer-based model trained on 341 GB of text, that was shown to be superior to variants of DialoGPT (Zhang et al., 2019), Mitsuku[1], Cleverbot[2], and XiaoIce (Shum et al., 2018; Zhou et al., 2020). We also compare directly to DialoGPT, which is a 345M parameter model trained based 147M multi-turn dialogues extracted from Reddit discussion threads.

The evaluation metric used for Meena in (Adiwardana et al., 2020) was SSA, the average of sensibleness and specificity, as judged by human raters either in static or interactive setups, which is shown to highly correlate with asking raters how "humanlike" the model is. We note however that the authors themselves state it may not capture all aspects of such a test, e.g. might not measure empathy. We additionally note that neither Meena's model, the static "Mini Turing Benchmark" used in the paper, nor the phrasing of the SSA evaluation question provided to annotators was released, making certain comparisons difficult. Further, the human-bot conversations were conducted by employees and were not blind to the model type (in the logs they say phrases such as "Hi Meena!"). DialoGPT in contrast is a publicly available open source model, and evaluations can be easily conducted and reproduced.

In this work we employ unbiased crowdworkers with reproducible experiments, and use ACUTE-Eval (Li et al., 2019b) (described more in Section 4) to directly ask the humanness question, rather than a proxy. Further, we also report results on engagingness as a main metric, because this measures more closely whether a human will be interested in talking to our bots.

3 Models, training, and data

3.1 Architectures

We consider three types of architectures in this work: retrieval, generative, and retrieve-and-refine models. All three use Transformers (Vaswani et al., 2017) as a base. More details for the architectures are provided in Appendix A.

Retriever Given a dialogue history (context) as input, retrieval systems select the next dialogue utterance by scoring a large set of candidate responses and outputting the highest scoring one. Typically, all possible training set responses are used as the candidate set. We employ the poly-encoder architecture of (Humeau et al., 2019). We consider two poly-encoder sizes: 256M (from (Smith et al., 2020)) and 622M parameter models which we trained here, both using $N = 64$ codes.

Generator We employ a standard Seq2Seq Transformer architecture to generate responses rather than retrieve them from a fixed set. Our implementation is based on the ParlAI version (Miller et al., 2017). We use Byte-Level BPE tokenization (Radford et al., 2019) trained on the pre-training data, as implemented in HuggingFace's Tokenizers.[3] We consider three sizes of model: 90M parameters (following Shuster et al., 2019), 2.7B parameters and 9.4B parameters. Our 9.4B parameter model has a 4 layer encoder, a 32 layer decoder with 4096 dimensional embeddings, and 32 attention heads. Our 2.7B parameter model roughly mimics the architectural choices of Adiwardana et al. (2020), with 2 encoder layers, 24 decoder layers, 2560 dimensional embeddings, and 32 attention heads.

---

[1] `https://www.pandorabots.com/mitsuku/`

[2] `https://www.cleverbot.com/`

[3] `https://github.com/huggingface/tokenizers`

Retrieve and Refine Current generative models are known to have issues with producing dull and repetitive responses which are improved, but not resolved, by simply scaling (Holtzman et al., 2019; Welleck et al., 2020; Li et al., 2019a). One approach to try to alleviate these problems is to combine a retrieval step before generation, referred to as a retrieve and refine model (Weston et al., 2018). We consider two variants for the retrieval step: dialogue retrieval and knowledge retrieval. Dialogue retrieval uses a retrieval-based dialogue model (see above) first to produce a response, which is then appended to the input sequence of the generator, along with a special separator token, and then generate from that expanded context with the generative architecture above. Knowledge retrieval first retrieves from a large knowledge base and conditions the generation on the retrieved knowledge, as done in (Dinan et al., 2019c). We hence refer to this as Wiz Generative model. We use the same knowledge retrieval system as in (Dinan et al., 2019c). A Retriever (the same as the Retriever paragraph in this section) is then used to rank candidates in the same way as for dialogue retrieval above. We additionally trained a Transformer-based classifier to choose when to perform retrieval or not on a per-turn basis, as some contexts do not require knowledge. We note all other models in this work do not condition on retrieved knowledge.

3.2 Training Objectives

Training for the retrieval models follows Humeau et al. (2019). To train the generative models, we use standard Maximum Likelihood Estimation (MLE). For Retrieve and Refine, we adopt α-blending from Weston et al. (2018) to ensure the model does not ignore the retrieved utterance. To combat some failures in model generations, we also experiment with the unlikelihood loss (Welleck et al., 2020; Li et al., 2019a), penalizing overrepresented n-gram phrases. An extensive description of training objectives and implementation can be found in the Appendix B.

3.3 Decoding

We compare several well-known approaches: beam search for different beam sizes, top-k sampling (Fan et al., 2018), sample-and-rank (Adiwardana et al., 2020). We also experiment with minimum-length constraints that forbid end-token generation below a minimum length, and a predictive-length approach that predicts one of four utterance lengths using a classifier built on top of our retrieval architecture. Finally, we also experiment with subsequence blocking through standard beam blocking of n-grams (Paulus et al., 2017) with $n = 3$, considering both the generated utterance or the input sequence (previous utterances from either speaker).

3.4 Training Data

We use English training data. For pre-training, we use a variant of Reddit discussions, which has also been used in several existing studies, see e.g. Yang et al. (2018); Mazaré et al. (2018); Keskar et al. (2019); Shuster et al. (2019). Following Humeau et al. (2019), we use a previously existing Reddit dataset extracted and obtained by a third party and made available on pushshift.io (Baumgartner et al., 2020). Dataset filtering heuristics are provided in the appendix. Our final dataset contains 1.50B comments totaling 56.8B label BPE tokens and 88.8B context tokens.[4] We divide the corpus into 4096 roughly-equal sized chunks, stratified by thread ID (such that no two comments from the same post appear across folds), and reserve the last two chunks for validation and test respectively, each approximately 0.02% of the full dataset (∼360k comments each). For fine-tuning, we use several smaller, but more focused datasets released by the academic community, that were collected to display desirable conversational traits (Roller et al., 2020): the ConvAI2 dataset (Zhang et al., 2018) focuses on personality and engaging the other speaker, Empathetic Dialogues (Rashkin et al., 2019) focuses on empathy, and Wizard of Wikipedia (Dinan et al., 2019c) focuses on knowledge. Finally, Blended Skill Talk (Smith et al., 2020) provides a dataset that focuses on blending these skills. We refer to the "BST tasks" as training on all four tasks together. In addition to skilled-focus datasets, we apply a classifier similar to the one trained in (Dinan et al., 2019b) at test time to detect toxic language before it is shown, and gauge how often such classifiers flag model responses.

4 Evaluation Methods

While we employ and report automatic metrics, our main evaluation involves the ACUTE-Eval procedure (Li et al., 2019b), whereby evaluators are asked to make pairwise evaluations of complete

---

[4] Note that the 90M model discussed later in the paper uses a variant of the corpus with less filtering. See Shuster et al. (2019) for details.

Model	C2 ($K = 20$)	WoW ($K = 100$)	ED ($K = 100$)	BST ($K = 100$)
256M	88.55	91.70	62.67	83.45
622M	89.96	93.22	70.15	82.11

Table 1: Hits@1/**K** of fine-tuned poly-encoder models on the validation set for BST datasets. Hits@$1/K$ measures recall@1 when ranking the gold label among a set of $K - 1$ other random candidates.

dialogues. We consider two evaluation questions, derived from (Li et al., 2019b): (1) Engagingness question: "Who would you prefer to talk to for a long conversation?"; (2) Humanness question: "Which speaker sounds more human?".

Nevertheless, full human evaluations are time consuming and costly, requiring humans to spend time conducting conversations with bots as well as scoring them. As an alternative, we use the self-chat procedure from Li et al. (2019b) for some of our modeling and hyperparameter choices where the full ACUTE-Eval would end up too costly, and only use the full human-bot chat evaluation at the final stage. In this work we use the BST-setting to perform self-chats, i.e. models are given the personas, topics and previous utterances to initiate the conversation, see Appendix E.2 and Figure A.2. Note that when using deterministic methods such as beam decoding, this prevents the models from generating the same conversation repeatedly.

5 Results & Analysis

5.1 Automatic Evaluations

Retriever We fine-tune the retrieval models on ConvAI2, Wizard of Wikipedia, Empathetic Dialogues, and Blended Skill Talk datasets (BST variants of each[5]) and automatically evaluate them by measuring *hits*@$1/K$ on the validation sets of each of these datasets. Results are shown in Table 1.

Generator We assess the performance of our 90M, 2.7B, and 9.4B parameter models by measuring perplexity on the validation set from pushshift.io Reddit[6] and on the smaller dialogue datasets, before and after fine-tuning. Results are shown in Table 2 and Table 3, and show relatively large perplexity improvements after fine-tuning. Table 3 also shows slightly increased perplexity for Retrieve and Refine (RetNRef), consistent with Weston et al. (2018).

Safety Table 4 also shows that fine-tuning on BST results in safer responses compared to the pre-trained baseline, as gauged by an unsafe word list or the safety classifier of Dinan et al. (2019b),[7] and that humans do utter unsafe responses, but much less frequently so in ConvAI2 than in pushshift.io Reddit. This explains why our fine-tuning tends to make the model reply with fewer unsafe utterances.

5.2 Self-Chat Evaluations

We next perform a number of self-chat ACUTE-Evals (see Appendix G) over various modeling choices, using the engagingness question and $\sim$140 trials per pair compared, in order to select the best model set-up to evaluate in a full human evaluation. While the goal of these self-chat experiments is to whittle down the set of models which undergo full human evaluation, we also provide full experimental results and descriptions in Appendix H.2.

Experiments controlling the minimum beam length (in terms of BPE tokens) with a fixed hyperparameter, or by adjusting it with a predictor of the optimal length, show that both methods improve significantly over not controlling the length (83% to 17%; 81% to 19%). In the remainder of the experiments in the paper we thus chose a minimum beam length of 20 BPE tokens.

We also investigate the use of beam blocking. Blocking tends to increase performance, in line with other works, although the results were not significant. We employ full blocking in the remainder of our experiments. Finally, we compare different values of beam size to other search strategies: Top-k sampling, and the sample and rank strategy of Adiwardana et al. (2020) using Top-k ($k = 40$) and 20 samples, suggesting a sweet spot of beam size, where a value of 10 is superior to 1 or 30, which is then on par with sampling methods (48% to 52%), although none of these results is significant. We employ beam size 10 in the remainder of our experiments.

Fine-tuning on the smaller, more focused BST datasets leads to significant improvements (60% to 40%), which may come from adjusting the model to focus on persona, knowledge and empathy, or providing talking points to work into the conversation. Using persona contexts at conversation time

---

[5] `https://parl.ai/projects/bst`

[6] For the 90M parameter model, results are reported from Shuster et al. (2019), as we use that same model.

[7] Both methods available in ParlAI (Miller et al., 2017).

Name	Total Params	V	L_{enc}	L_{dec}	d	h	Steps	PPL
90M	87,508,992	55K	8	8	512	16	2.86M	25.6
2.7B	2,696,268,800	8K	2	24	2560	32	200K	13.3
9.4B	9,431,810,048	8K	4	32	4096	32	200K	12.2

Table 2: **Perplexity on the validation set of pushshift.io Reddit** for several generative Transformer models with given architecture settings. Note that perplexity is not directly comparable between the 90M models and the larger models as the 90M models use a different dictionary. Columns include the vocabulary size (V), number of encoder and decoder layers (L_{enc}, L_{dec}), embedding dimensionality (d), Multihead Attention Heads (h), and training steps.

Size	Pre-train only	BST Gen	BST RetNRef
90M	20.5	13.8	14.2
2.7B	12.0	9.0	9.8
9.4B	11.2	8.6	-

Table 3: **Average perplexity of the pre-trained and fine-tuned models on the validation sets for the BST datasets.** Note that perplexity is not directly comparable between the 90M models and the larger models as 90M models use a different dictionary. Fine-tuning gives gains compared to pre-training on pushshift.io Reddit alone. The Retriever sizes are 256M and 622M for the 90M and 2.7B RetNRef models, respectively. Results on each individual skill (task) are shown in the appendix and present the same pattern.

	pushshift.io Reddit		ConvAI2	
Method	Word List	Classifier	Word List	Classifier
Human	12.9%	18.5%	0.32%	3.8%
Reddit Gen	4.4%	17.8%	0.10%	12.1%
BST Gen	0.6%	9.5%	0.05%	1.6%

Table 4: **Safety of utterances, before filtering through a safety classifier.** We compare human, pre-trained and fine-tuned 90M model responses given pushshift.io Reddit and ConvAI2 contexts using either an unsafe word list or a trained classifier from (Dinan et al., 2019b). The pushshift.io Reddit dataset contains more unsafe contexts, leading to more unsafe responses. Models fine-tuned on the safer BST tasks are less toxic than the pre-trained pushshift.io Reddit model on either type of dataset context.

provides a win compared to omitting them (54% to 46%), so we use personas for our full experiments.

5.3 Full (Human-Bot Chat) Evaluations

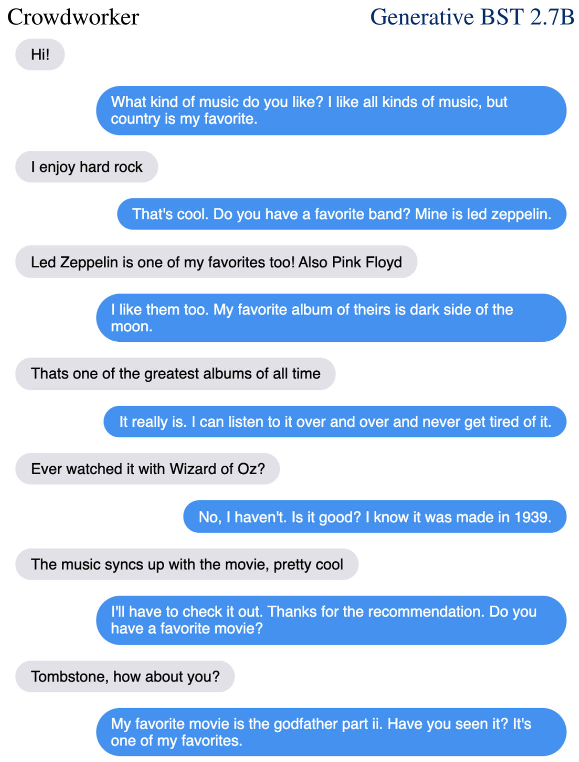
For human-bot conversation data collection we used the same setting proposed in (Adiwardana et al., 2020): open-ended chat that begins with the message "Hi!" from the human to the bot, and has a minimum interactive conversation length of 14 turns, collecting 100 conversations per model via crowdworkers. Figure 1 shows a cherry-picked conversation between a crowdworker and the Generative BST 2.7B model. More examples and more detailed experiments are given in the Appendix.

Overall ranking of models We compare several variants of our models to the publicly available human-human chat logs from Adiwardana et al. (2020), and include the publicly available bot-human Meena logs (Adiwardana et al., 2020) (some toxic conversations were removed by the authors and not made publicly available). Results in Figure 2 confirm that: (1) Fine-tuning on BST (BST Generative 2.7B) is superior to pre-training only (pushshift.io Reddit Generative 2.7B); (2) Beam search with a minimum beam length of 20 (BST Generative 2.7B) is superior to having no minimum length (BST Generative (2.7B) std. beam); (3) The larger BST Generative (2.7B) is superior to the smaller model BST Generative (90M). We find RetNRef models (both dialogue version and using knowledge retrieval) do not improve over their generative counterparts when using the best decoding schemes for the generative models[8].

Our largest BST Generative 9.4B model performs worse on engagingness compared to our 2.7B model, despite having lower perplexity, showing correlation between these metrics is not straightforward. We verified this result further by performing an ACUTE-Eval of engagingness directly comparing the 2.7B and 9.4B against each other, which resulted in a 56% win for the smaller model, aligning with the other results. Future work should aim to understand this result further. Additional experiments matching up our models in pairwise comparisons with the publicly available Meena bot-human logs are provided in the Appendix, showing

---

[8] In earlier experiments (see Figure A.5) we found that RetNRef could outperform its generative counterpart, but after optimizing the generative model decoding parameters, decoding length and beam blocking, it then performs much better compared to RetNRef. The RetNRef model produces longer generations than the vanilla greedy-search generation algorithms, so the original improvement may be from this.



Crowdworker

Generative BST 2.7B

Hi!

What kind of music do you like? I like all kinds of music, but country is my favorite.

I enjoy hard rock

That's cool. Do you have a favorite band? Mine is led zeppelin.

Led Zeppelin is one of my favorites too! Also Pink Floyd

I like them too. My favorite album of theirs is dark side of the moon.

Thats one of the greatest albums of all time

It really is. I can listen to it over and over and never get tired of it.

Ever watched it with Wizard of Oz?

No, I haven't. Is it good? I know it was made in 1939.

The music syncs up with the movie, pretty cool

I'll have to check it out. Thanks for the recommendation. Do you have a favorite movie?

Tombstone, how about you?

My favorite movie is the godfather part ii. Have you seen it? It's one of my favorites.

Figure 1: **Cherry-picked crowdworker example.** Conversation between a crowd worker (left speaker) and the Generative BST 2.7B model (right speaker).

similar trends in ranking of models. In those experiments, our best models improve significantly over Meena, with BST Generative 2.7B winning 75% of the time in pairwise match-ups for the engagingness question and 65% for the humanness question. Meena generally tends to fare better at the humanness question than the engagingness question (see Figure 3), which is line with the goals and modeling choices in that work.

Response Length Figure 4 shows that the average response length (in terms of BPE 8k dictionary tokens) of Generative BST (2.7B) with the constraint (of 20) is around 21 tokens, vs. 9.5 tokens without. Thus, the beam search often ends as soon as the constraint is fulfilled. Meena's average length is 10.4, and humans engaged in human-human chats is 18.0. Humans speaking to models (or other humans) will often match response length if they are engaged in the conversation, and there appears to be correlation of their average response length with engagement (intuitively, humans are expending time and energy typing on their keyboard, which they are more likely to do if engaged).

	Model	vs. Human
DialoGPT std. beam (Zhang et al., 2019)	24 **	76 **
Meena (Adiwardana et al., 2020)	28 **	72 **
DialoGPT min beam 20 (Zhang et al., 2019)	34 **	66 **
BST Generative (2.7B) std. beam	21 **	79 **
BST RetNRef (256M/90M)	37 **	63 **
BST Generative (90M)	42	58
pushshift.io Reddit Generative (2.7B)	44	56
BST Generative (9.4B)	45	55
BST RetNRef (622M/2.7B)	46	54
Wiz Generative (2.7B)	47	53
BST Unlikelihood (2.7B)	48	52
BST Generative (2.7B)	49	51

Figure 2: ACUTE-Eval of engagingness of models vs. humans by comparing human-bot logs to human-human logs. Rows with ** are statistically significant.

	Ours	vs. Meena
DialoGPT std. beam (Zhang et al., 2019)	27 **	73 **
BST Generative (2.7B) std. beam	46	54
DialoGPT min beam 20 (Zhang et al., 2019)	47	53
BST RetNRef (256M/90M)	49	51
pushshift.io Reddit Generative (2.7B)	54	46
BST Generative (90M)	59	41
Wiz Generative (2.7B)	59 *	41 *
BST RetNRef (622M/2.7B)	65 **	35 **
BST Generative (2.7B)	65 **	35 **
BST Generative (9.4B)	66 **	34 **
BST Unlikelihood (2.7B)	70 **	30 **

Figure 3: Human-Chat ACUTE-Eval of **humanness**, various models compared to Meena. Our best models are considered more humanlike than Meena, rows with * and ** are statistically significant.

5.4 Failure Cases and Model Extensions

While performance in the ACUTE-Eval setup appears at first sight to be very strong (e.g. 49% to 51% for our 2.7B generative model compared to human-human logs), we are nowhere near as close to solving the problem of open-domain conversation as this evaluation would indicate. Here, we highlight problems with our models, and elucidate why our evaluation does not capture them. Selected example failures from crowdworker logs are given as conversation snippets in Figure 6, and further failures constructed by the paper authors are shown in the Appendix (H.5).

Vocabulary Usage Generative models employing beam search decoding tend to generate common words too frequently, and rare words too infrequently, as compared to the human distribution (Holtzman et al., 2018; Welleck et al., 2020; Li et al., 2019a). In dialogue, humans can interpret

Model	Model	Human
Meena	10.4	8.2
BST Gen (2.7B) std beam.	9.5	11.3
BST Gen (2.7B)	21.3	16.3
Human	18.0	18.0

Figure 4: Response length statistics for various models. We note the best performing methods have longer response lengths, and humans interacting with them have longer response lengths in kind.

n-gram	MLE	UL	Human
Do you have	110	60	6
you have any	82	46	2
a lot of	74	46	14
What do you	57	20	6
you like to	54	43	1

Figure 5: Counts of 5 most common 3-grams from the BST Generative 2.7B model (MLE) from 100 conversation logs talking to crowdworkers, compared to those of the same model trained with unlikelihood (UL), and to human logs (for the same number of utterances).

I like rap. It's probably not good for fishing.

I listen to rap when I go fishing too! What kind of rap do you listen to?

Really? I am from the south so I like trap music

I'm from the south too! I live in louisiana. What part of the south are you from?

Driving is fun! I like to listen to music while I drive. What do you listen to?

I listen to hiphop tracks. I like 50 cent. You have an hiphop fav?

I don't have a hip hop favorite, but I do have a country favorite. Do you have a favorite?

I'm from the midwest, but I live in the south now. How about you?

Born and raised in the south

What part of the south? I was born in georgia, but raised in tennessee

Figure 6: **Examples of issues when talking to crowdworkers** with our Generative BST 2.7B model: nontrivial repetition (top example), forgetfulness (second example), contradiction (third example, Georgia is not in the Midwest).

this as technically correct, but unengaging. Using sampling to select lower likelihood generations can help, but at the risk of saying something which makes less sense. Despite the minimal length constraints forcing models to provide more rich and varied generation output, our best models still produce too many common words. Figure 5 shows that frequent expressions are clearly over-used by the model. The current evaluation may not expose this as boring because the conversations are short and evaluated separately. Applying unlikelihood training successfully reduced this overexpression during training, and also in the final conversation logs with humans, as shown in Figure 5. Unfortunately, this made a very small or negative impact in our ACUTE-Evals of engagingness (Figure 2 and A.14), although this did score highly in terms of humanness (Figure 3).

Other issues A still common issue is a tendency to repeat (Holtzman et al., 2019), which is partly alleviated by beam blocking, but then manifests as a tendency for models to copy their conversation partner. While this can be engaging, control of this might be achieved through unlikelihood training to minimize context repeats (Li et al., 2019a) or adding a persona to the bot. Our models also occasionally contradict themselves (e.g., Figure 6), albeit less often in the larger models. They often appear to fail to remember what their partner tells them, but this might be due to missing the logical link that they should not ask that question, rather than the models actually “forgetting" (if the previous response is in their dialogue context). While some recent work has posed possible solutions for these issues (Li et al., 2019a), they have not yet been fully resolved. Perhaps surprisingly, factual errors appear relatively rarely in crowdworker conversations with the bots. We believe this is due to the nature of the evaluation conducted: the conversations start with “Hi!” and tend to cover only shallow topics, and are rarely long enough to go deeper. Exploring more focused topics would likely expose the model’s weaknesses. In fact, our models often switch topics, which could be a side effect of the ConvAI2 dataset which exhibits this behavior. The Wizard of Wikipedia dataset, however, was specifically constructed to avoid this. We implemented a model that directly incorporated reading Wikipedia (Wiz Generative 2.7B, Sec 3.1), and it may employ knowledge that the pure sequence to sequence model cannot. Unfortunately the reading of knowledge only had a negative impact in ACUTE-Evals compared to a similarly sized model without knowledge retrieval (Figure 2). This might reflect both (i) deeper knowledge rarely being required in the current evaluation setup; and (ii) the model attempting to use knowledge when there is no need, or using it incorrectly. True open-domain dialogue agents should be able to use knowledge effectively, and to achieve that we have to be able to measure that effectively.

Conversation Length and Memory Our current evaluation involves very short (14-turn) one-shot conversations – not enough to expose how repetitive and forgetful our bots are. Our generative architectures which are standard Transformers have a hard limit of 128 BPE tokens of history, so cannot possibly expand upon things they have learnt from or about the user. While several recent works have extended neural architectures to longer contexts (Dai et al., 2019; Rae et al., 2020; Kitaev et al., 2020; Beltagy et al., 2020), we have neither implemented those, nor would the current evaluation setup likely capture their success.

Further Notes on Evaluation Our evaluation set-up involves short multi-turn conversations with no instructions. A preliminary experiment looked for performance drops of our models over 100 28-turn conversations. We compared the second half of the conversations to the shorter versions for the same 2.7B generative BST model, but did not see a statistically significant difference. Thus, they either need to be longer, or the whole conversation has to be evaluated at once, which becomes difficult for a human annotator not engaged in the conversation itself. Another avenue is to provide instruction. For example, the Wizard of Wikipedia task (Dinan et al., 2019c) asks speakers to converse in depth on a randomly chosen topic, changing the the skills the model will be evaluated on.

Finally, when comparing to human performance, the quality of the human conversations matters. In Figure 2 we compared to logs of employees from Adiwardana et al. (2020). These conversations are often rich and engaging. We also tried comparing to human-human crowdworker conversations, taken from the BST paper (Smith et al., 2020). We then found our models perform better than when compared to employees: our generative BST 2.7B model in an ACUTE-Eval of engagingness beats humans 56% to 44% (not statistically significant), whereas it scored 49% to 51% against employee chats. We also compared crowdworker humans directly to employee humans, with a 56% to 44% win for employees in terms of engagingness, and a 59% to 41% win in terms of humanness. We believe utilizing crowdworkers as a barometer for our models is desirable, as this can yield more replicable experiments, so finding a way to close this gap, perhaps with alternative ways of matching workers or differing set-ups and instructions remain possible avenues of investigation.

6 Discussion

In this work we aimed to study and identify which methods work well together to maximize human evaluations, and identified a few key factors (scale, fine-tuning tasks, and decoding choices) that when blended together give state of the art performance.

While our methods have achieved higher human ratings of engagingness and humanness, our models still have numerous issues. Firstly, even our best models still make mistakes: they i) contradict or repeat themselves on occasion, ii) tend to repeat the same phrases in separate conversations, and iii) hallucinate knowledge as seen in other generative systems (Massarelli et al., 2019). Each of these faults naturally leads to future research directions; we made some attempt here using unlikelihood (Li et al., 2019a) and conditioning on knowledge (Dinan et al., 2019c), but more needs to be done.

As the human evaluations are on short dialogues (14 turns), longer conversations would likely make these issues appear much worse, and expose the limitations of the Transformer architectures stemming from their limited dialogue history. A number of recent architectures attempt to incorporate longer memory, and that is also a fruitful direction, although evaluation is more challenging as long conversations have to be collected, and evaluated. An alternative is to seed the conversation with a topic or otherwise provide instructions to the human speaker during evaluation to give the conversation a certain focus, which would more deeply probe the skills of the bot. On the modeling side, longer conversations could also make the choice of context material provided to the bot more salient. Besides helping with consistency, the persona and topic that are given as initial context in Blended Skill Talk can help models introduce interesting talking points in the conversation. However, they would need to be far more detailed for longer or repeated conversations to help the models be consistent and avoid repetition, and in our experimental setup did not affect evaluations strongly.

For deployment of a chatbot, being well-behaved remains a significant challenge. In particular, we expect bots to have more integrity than the average human (or to even be faultless), but they have much less understanding of what they are saying than humans. Recent work mitigating toxicity (Dinan et al., 2019b) and gender bias in dialogue generation (Dinan et al., 2019a) is encouraging, but much work remains to be done. Making models publicly available could help the community work together on understanding and fixing these issues.

The work of Adiwardana et al. (2020) showed that there is a correlation between human evaluation and perplexity, given a fixed decoding scheme. We argue that while this is important, other factors are also at play and cannot be ignored: (1) the choice of training data is paramount, as shown by our pushshift.io Reddit (pre-training) vs. Blended Skill Talk experiments; and (2) decoding algorithms make large differences for the same fixed perplexity model (Sec. H.2). We find that while our 2.7B parameter model gives large gains over our 90M parameter model, our largest 9.4B model does not have a clear win in human evaluations over our 2.7B model, despite having lower perplexity. This is in line with previous observations, e.g., dialogue competitions are not always won by the model with the lowest perplexity (Dinan et al., 2020), and models that take a small hit in perplexity but provide gains at decoding time can give far improved results (Welleck et al., 2020; Li et al., 2019a). Further refining and understanding these ingredients, and how they help to build the recipe as a whole, remain important directions.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. *arXiv preprint arXiv:2001.08435*.

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

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. *arXiv preprint arXiv:2004.10151*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

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

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2019a. Queens are powerful too: Mitigating gender bias in dialogue generation. *arXiv preprint arXiv:1911.03842*.

Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019b. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4537–4546, Hong Kong, China. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2020. The second conversational intelligence challenge (ConvAI2). In *The NeurIPS '18 Competition*, pages 187–208, Cham. Springer International Publishing.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019c. Wizard of Wikipedia: Knowledge-powered conversational agents. In *Proceedings of the International Conference on Learning Representations*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Maryam Fazel-Zarandi, Shang-Wen Li, Jin Cao, Jared Casale, Peter Henderson, David Whitney, and Alborz Geramifard. 2017. Learning robust dialog policies in noisy environments. In *Proceedings of Workshop on Conversational AI*.

Asma Ghandeharioun, Judy Hanwen Shen, Natasha Jaques, Craig Ferguson, Noah Jones, Àgata Lapedriza, and Rosalind W. Picard. 2019. Approximating interactive human evaluation with self-play for open-domain dialog systems. *Advances in Neural Information Processing Systems*.

Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. 2019. Learning from dialogue after deployment: Feed yourself, chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684, Florence, Italy. Association for Computational Linguistics.

Tianxing He, Jun Liu, Kyunghyun Cho, Myle Ott, Bing Liu, James Glass, and Fuchun Peng. 2019. Mix-review: Alleviate forgetting in the pretrain-finetune framework for neural language generation models. *arXiv preprint arXiv:1910.07117*.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1638–1649. ACL.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *Proceedings of the International Conference on Learning Representations*.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. 2019. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems*, pages 103–112.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *Proceedings of the International Conference on Learning Representations*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2019a. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. *arXiv preprint arxiv:1911.03860*.

Margaret Li, Jason Weston, and Stephen Roller. 2019b. ACUTE-EVAL: Improved dialogue evaluation with optimized questions and multi-turn comparisons. In *NeurIPS workshop on Conversational AI*.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers. *arXiv preprint arXiv:2002.11794*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, and Luke Zettlemoyerand Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. 2019. How decoding strategies affect the verifiability of generated text. *arXiv preprint arXiv:1911.03587*.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2017. Sequence effects in crowdsourced annotations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2860–2865.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84. ACL.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Shrimai Prabhumoye, Margaret Li, Jack Urbanek, Emily Dinan, Douwe Kiela, Jason Weston, and Arthur Szlam. 2020. I love your chain mail! making knights smile in a fantasy game world. *arXiv preprint arXiv:2002.02878*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Stephen Roller, Y-Lan Boureau, Jason Weston, Antoine Bordes, Emily Dinan, Angela Fan, David Gunning, Da Ju, Margaret Li, Spencer Poff, et al. 2020. Open-domain conversational agents: Current progress, open problems, and future directions. *arXiv preprint arXiv:2006.12442*.

Pararth Shah, Dilek Hakkani-Tür, Bing Liu, and Gokhan Tür. 2018a. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 41–51, New Orleans - Louisiana. Association for Computational Linguistics.

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018b. Building a conversational agent overnight with dialogue self-play. *arXiv preprint arxiv:1801.04871*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv preprint arXiv:1804.04235*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*.

Heung-yeung Shum, Xiao-dong He, and Di Li. 2018. From Eliza to XiaoIce: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19(1):10–26.

Kurt Shuster, Da Ju, Stephen Roller, Emily Dinan, Y-Lan Boureau, and Jason Weston. 2019. The dialogue dodecathlon: Open-domain knowledge and image grounded conversational agents.

Eric Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can you put it all together: Evaluating conversational agents' ability to blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Wei Wei, Quoc V. Le, Andrew M. Dai, and Li-Jia Li. 2018. A goal-oriented neural conversation model by self-play.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Jason Weston, Emily Dinan, and Alexander Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pages 87–92, Brussels, Belgium. Association for Computational Linguistics.

Jason E Weston. 2016. Dialog-based language learning. In *Advances in Neural Information Processing Systems*, pages 829–837.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A transfer learning approach for neural network based conversational agents. In *NeurIPS Workshop on Conversational AI*.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning semantic textual similarity from conversations. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 164–174, Melbourne, Australia. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213. ACL.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. DialoGPT: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2020. The design and implementation of XiaoIce, an empathetic social chatbot. *Computational Linguistics*, pages 1–62.

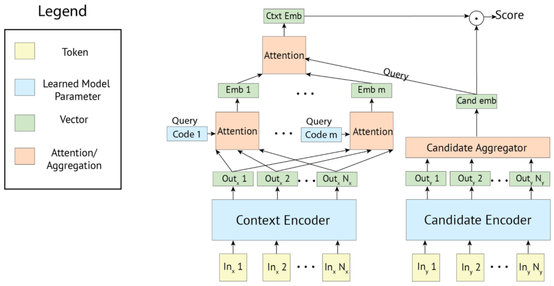


Figure A.1: The Poly-encoder Transformer architecture (Humeau et al., 2019) for retrieval encodes global features of the context using multiple representations (codes), which are attended to by each possible candidate response. This final attention mechanism gives improved performance over a single global vector representation, while being tractable to compute.

A Model architecture details

This section gives additional details on the architectures we use.

A.1 Poly-encoder in the Retriever

The underlying architecture of our Retriever is the poly-encoder architecture of (Humeau et al., 2019). Poly-encoders encode global features of the context using multiple representations (n codes, where n is a hyperparameter), which are attended to by each possible candidate response, see Figure A.1. This final attention mechanism gives improved performance over a single global vector representation (so-called "bi-encoders"), while still being tractable to compute compared to simply concatenating input and output as input to a Transformer (so-called "cross-encoders"). The poly-encoder has state-of-the-art performance on a number of dialogue tasks when compared to other retrieval models, and also gives comparable performance to the winning generative models on the ConvAI2 competition task (Zhang et al., 2018) in terms of human evaluation (Li et al., 2019b). We consider two poly-encoder sizes: 256M (from (Smith et al., 2020)) and 622M parameter models which we trained here, both using $N = 64$ codes.

A.2 Retrieve and Refine

Dialogue Retrieval Given the dialogue history, the retrieval model from Section A.1 is first used to produce a response. Rather than showing this response to the speaking partner, it is appended to the input sequence of the generator, along with a special separator token. The generator then outputs a response as normal given this modified input sequence. Retrieval models produce human written utterances which tend to include more vibrant language than the most high probability utterances of a standard generative model. Hence, if the generative model learns when to copy the elements of such an utterance, and when not to, it can provide improved responses.

Knowledge Retrieval Generative models are known to hallucinate knowledge, and in general are unable to read and access external knowledge other than what is embedded in their model parameters, which may be imperfect. This can be improved by first retrieving from a large knowledge base, instead of retrieving an initial dialogue utterance. We can then condition the generation on the retrieved knowledge, as done in models proposed for the Wizard of Wikipedia task in (Dinan et al., 2019c). In the main body of the paper, we refer to this as a Wizard Generative model, as the supervised training signal of how to use knowledge in dialogue comes from the Wizard of Wikipedia task, even though we multi-task on other tasks as well. We use the same retrieval system as in that cited work, which uses a TF-IDF-based inverted index lookup over a Wikipedia dump[9] to produce an initial set of knowledge candidates. A Transformer retriever model (the same as Section A.1) is then used to rank the candidates and select a single sentence which is used to condition generation. The additional Transformer-based classifier that indicates whether to perform retrieval or not for each turn was trained as a two-class classifier discriminating between contexts that require knowledge or not in our fine-tuning tasks.

B Training Objectives

B.1 Ranking for Retrieval

To train the retrieval models, a cross-entropy loss is minimized in which the logits are $y_{cand_1}, \cdots, y_{cand_n}$, where y_{cand_1} is the score of the correct response and the others are sampled negatives. Following Humeau et al. (2019), during training we use the other responses in the batch for negatives. This allows for much faster training, as we can reuse the embeddings computed for each candidate, and also use a larger batch size. In our training we are able to use batches of 512 elements.

---

[9] `https://parl.ai/projects/wizard_of_wikipedia/`

B.2 Likelihood Training for Generation

To train the generative models, we use the standard Maximum Likelihood Estimation (MLE) approach. Given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}$, minimize:

$$\mathcal{L}^{(i)}_{\text{MLE}}(p_\theta, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}) = -\sum_{t=1}^{|y^{(i)}|} \log p_\theta(y^{(i)}_t | \mathbf{x}^{(i)}, y^{(i)}_{<t}),$$

where $\mathbf{x}^{(i)}$ is a gold input context and $\mathbf{y}^{(i)}$ is a gold next-utterance, and $y^{(i)}_t$ is the t-th token of $\mathbf{y}^{(i)}$.

B.3 α-blending for Retrieve and Refine

For retrieve and refine, simply appending dialogue retrieval responses to the context of a generative model and training with MLE unfortunately does not yield satisfying results. As the correspondence between gold label and retrieved utterance is not necessarily clear, a trained model often opts to simply ignore the retrieval utterance, as was shown in Weston et al. (2018). To ensure it is used, one can replace the retrieved response instead with the gold response $\alpha\%$ of the time, treating α as a hyperparameter to be tuned. This gives a smooth transition between retrieval and generator-only systems. For knowledge retrieval we find this issue to be less of a problem as the fine-tuning datasets used have a clear correspondence between gold knowledge conditioning and response, and in that case we only use the gold knowledge during training.

B.4 Unlikelihood training for generation

An alternative method to combat the failures in model generations is to change the loss function. The unlikelihood loss (Welleck et al., 2020; Li et al., 2019a) has been shown to help fix mismatches between human and model distributions across various axes, including decreasing repetitions and mitigating the issue of overrepresented vocabulary tokens.

The unlikelihood loss penalizes a set of tokens $\mathcal{C}_t$ at each time-step, $\mathcal{L}^{(i)}_{\text{UL}}(p_\theta, \mathcal{C}_{1:T}, \mathbf{x}, \mathbf{y}) =$

$$-\sum_{t=1}^{|y|} \sum_{y_c \in \mathcal{C}_t} \log\left(1 - p_\theta(y_c | \mathbf{x}, y_{<t})\right),$$

where $\mathcal{C}_t \subseteq \mathcal{V}$ is a subset of the vocabulary. The overall objective in unlikelihood training then consists of mixing the likelihood and unlikelihood losses,

$$\mathcal{L}^{(i)}_{\text{ULE}} = \mathcal{L}^{(i)}_{\text{MLE}} + \alpha \mathcal{L}^{(i)}_{\text{UL}}, \tag{1}$$

where $\alpha \in \mathbb{R}$ is the mixing hyper-parameter.

Likelihood tries to model the overall sequence probability distribution, while unlikelihood corrects for known biases. It does this via the set of *negative candidates* $\mathcal{C}_t$ calculated at each step t; typically one specifies in advance a method for generating such candidates, for example the tokens which have been repeated or overrepresented. Likelihood pushes *up* the probability of a gold token $y^{(i)}_t$ while unlikelihood pushes *down* the probability of negative candidate tokens $y_c \in \mathcal{C}_t$. In this work during training we keep a running count of the distribution of n-grams that appear when generating from the model, and choose tokens as negative candidates from these n-grams when their counts are above the human distribution counts as measured from the gold responses.

C Decoding

For generative models, at inference time, one must choose a decoding method to generate a response to the dialogue context given as input. This section provides more details on the decoding approaches we compare.

C.1 Beam Search

Two widely used deterministic decoding approaches are greedy search and beam search. The former can be seen as a special case of the latter. Greedy search selects the highest probability token at each time step: $y_t = \arg\max p_\theta(y_t | x, y_{<t})$. Beam search maintains a fixed-size set of partially-decoded sequences, called hypotheses. At each time step, beam search forms new hypotheses by appending each token in the vocabulary to each existing hypothesis, scoring the resulting sequences then selecting the highest scoring sequences.

We compare beam search for different beam sizes in our experiments.

C.2 Sampling

An alternative is to sample from a model-dependent distribution at each step, $y_t \sim q(y_t | x, y_{<t}, p_\theta)$. In order to prevent sampling low probability tokens, a typical approach is to restrict sampling to a subset of the vocabulary at each step, and sampling according to those (renormalized) probabilities. In this work, we compare top-k sampling (Fan et al., 2018) and sample-and-rank (Adiwardana et al., 2020). The latter performs sampling S times, and selects the generated sample with the highest probability.

C.3 Response Length

Generating with a beam tends to produce short generations that do not match the length statistics of the human utterances they were trained on (Weston et al., 2018). However, longer responses, if of high quality, can be more engaging than very short ones. While following the human distribution may not give optimal performance for a bot – for example, it may want to err on the side of brevity for improved human evaluation, because that is less likely to expose its failings – making its responses longer may make them provide more information, and make them less dull.

We consider two simple methods to control the length of a model's responses.

Minimum length The first method we consider is a hard constraint on the minimum generation length: the end token is forced to not be generated until a minimum sequence length is achieved.

Predictive length The second approach is to predict the length based on human-human conversation data. To do this we train a 4-class classifier by binning the lengths of the next conversation turn (e.g., < 10, < 20, < 30, or > 30 tokens). We use the same architecture as the retrieval model for this classifier. Then, at test time, the classifier is first used to predict the length of the next response, and sets the minimum generation length constraint to its corresponding prediction. Unlike the previous approach, this results in more natural variable length conversation turns, while ensuring long responses when they seem natural. One drawback, however, is that this procedure makes our system more complex.

C.4 Subsequence Blocking

Sequence generation models are known to repeat subsequences (Holtzman et al., 2018), particularly in stochastic methods such as beam search, but also in sampling methods as well (Adiwardana et al., 2020). We implement standard beam blocking of n-grams (Paulus et al., 2017) and use $n = 3$. We consider both blocking repeated n-grams within the generated utterance, and repeating of the input sequence (previous utterances from either speaker).

D Training Details

We detail the techniques we employ during pre-training and fine-tuning.

Pre-training Ranking models. We perform pre-training using the Fairseq (Ott et al., 2019) toolkit. Our 256M parameter ranking model is identical to the pre-trained model released by Humeau et al. (2019). Our 622M model is pre-trained using a simple Masked Language Model objective on the same data and dictionary as the large Generative models. We took all hyperparameter choices from those recommended in RoBERTa (Liu et al., 2019).

Pre-training Generative models. We perform pre-training using the Fairseq (Ott et al., 2019) toolkit. Our 2.7B and 9.4B parameter models were both trained using the Adam optimizer (Kingma and Ba, 2014). In order to fit the larger models onto nodes, we utilize Megatron-LM style model parallelism (Shoeybi et al., 2019), in which the Feed Forward network (FFN) and Multihead Attention layers of the Transformer are "vertically" sliced, minimizing the need for communication across GPUs. We also evaluated Adafactor (Shazeer and Stern, 2018), which allows for larger batch sizes, but we found it converged to a worse place than Adam. In all cases, we use a variant of mixed precision training (Micikevicius et al., 2017), storing gradients and optimizer state in FP32, but accumulating model parameters directly in FP16 (Ott et al., 2019). A dynamic loss scalar is utilized to prevent gradient underflow (Micikevicius et al., 2017). Both our 2.7B and 9.4B parameter models were trained with batches of approximately 500k label BPE tokens per batch. The 2.7B parameter model trained for approximately 200k SGD updates with a maximum learning rate of 2e-4, a linear warmup of 3125 steps, and an invsqrt LR scheduler (Vaswani et al., 2017); the model had not converged when we stopped. The 9.4B parameter model was trained with a maximum learning rate of 1.15e-4 and 2400 warmup steps for a total of 200k SGD updates, and did not appear to be overfitting.

Fine-tuning. We fine-tune our models using the ParlAI toolkit (Miller et al., 2017), which specializes in training and evaluating dialogue models. As opposed to the above pre-training, we utilize GPipe-style model parallelism (Huang et al., 2019), in which full layers are sharded across different GPUs, and each minibatch is further split into micro-batches to ensure maximum throughput. As in pre-training, we found that Adam outperformed Adafactor during fine-tuning, and we utilized Fairseq-style mixed precision training. Models were fine-tuned to convergence, with maximum learning rates of between 1e-6 and 1e-5.

E Training Data

We give additional details on the training data we use, which is all in English (#BenderRule).

E.1 Pre-training

pushshift.io Reddit filtering We start from Reddit posts obtained from PushShift[10] through July 2019. The subreddits cover a vast range of topics, and hence the dataset is a good candidate for helping train a dialogue model in the open-domain case. We apply heuristic rules to filter the dataset with the goal of providing a cleaner training signal. We remove the comment and all subsequent child comments if any of the following conditions are met:

1. The author is a known bot.
2. It comes from a known non-English subreddit.
3. The comment is marked as removed / deleted.
4. It is longer than 2048 characters and does not contain spaces.
5. It is longer than 128 BPE tokens.
6. It is shorter than 5 characters.
7. It contains a URL.
8. It starts with a non-ASCII character.
9. It is further than depth 7 in the thread.

Models were trained with maximum context and response lengths set to 128 BPE tokens, and longer examples were truncated. Our final dataset contains 1.50B comments totaling 56.8B label BPE tokens and 88.8B context tokens.

E.2 Fine-tuning

Our pre-training data, though large, contains data consisting of group discussions, rather than direct two-way conversational data. While it has a lot of useful content, it also still has a lot of noise, even after filtering. In contrast, the academic community has produced a number of smaller, but cleaner, more focused tasks, typically collected via crowdworkers, which have been made publicly available. These tasks can more accurately provide traits that are desirable for our models. This section details the more focused datasets we use.

ConvAI2: ConvAI2 is a dataset used at the NeurIPS 2018 competition of the same name, and is based on PersonaChat (Zhang et al., 2018; Dinan et al., 2020). The training data of 140k utterances involves paired crowdworkers having a conversation where they get to know each other, in which each is given a role to play based on sentences describing their persona, which were also separately crowdsourced (both speakers can see their own persona description, but cannot see their partner's persona). The task thus involves getting to know the other speaker and engaging them in friendly conversation, both asking and answering questions – useful skills for an open-domain conversational agent. Models trained on this task are thus conditioned on the persona and the dialogue history, which are concatenated. It was previously shown this dataset helps provide more engaging dialogue, and that the use of persona gives improved consistency for the bot.

Empathetic Dialogues (ED): Rashkin et al. (2019) constructed the Empathetic Dialogues dataset, which consists of 50k utterances of crowdworker conversations grounded in an emotional situation. In each dialogue, one speaker describes a personal situation and the other plays a "listener" role, displaying empathy during the discussion. Trained models are measured playing the part of the empathetic listener. It was previously shown fine-tuning models on this dataset helps them display more empathy in human evaluations.

Wizard of Wikipedia (WoW): The Wizard of Wikipedia task involves discussing a given topic in depth, where the goal is to both engage the partner as well as display expert knowledge (Dinan et al., 2019c). The dataset consists of 194k utterances over 1250 topics, where each conversation begins with a randomly chosen topic. A retrieval system over Wikipedia was used from which the dialogues were grounded during the human-human crowdsourced conversations. The topics were also crowdsourced and range from e-books to toga parties to showers. In most of our models we use the simpler version of the task where we only use the final conversations for fine-tuning, ignoring the retrieval aspect of the task. For our knowledge retrieve and refine model (Section 3.1) we do also use the gold retrieved knowledge ("checked sentence") for training the retrieval system. It was previously shown for generative models that using such knowledge was rated higher in human evaluation than without when discussing topics in depth.

---

[10] `https://files.pushshift.io/reddit/`

Persona for Unguided Speaker:	**Persona for Guided Speaker**:
My son plays on the local football team. I design video games for a living.	My eyes are green. I wear glasses that are cateye.

Wizard of Wikipedia topic: Video game design
Previous utterances (shown to speakers):
U: What video games do you like to play?
G: all kinds, action, adventure, shooter, platformer, rpg, etc. but video game design requires both artistic and technical competence AND writing skills. that is one part many people forget

Actual utterances:
U: Exactly! I think many people fail to notice how beautiful the art of video games can be. **(ConvAI2)**
(G selected the WoW suggestion: "Indeed, Some games games are purposely designed to be a work of a persons creative expression, many though have been challenged as works of art by some critics.")
G: Indeed, Some games games are purposely designed to be a work of a persons creative expression, many though have been challenged as works of art by some critics. **(WoW)**
U: Video games are undervalued by many and too easily blamed for problems like obesity or violence in kids **(WoW)**
G: Indeed, Just last week my son was playing some Tine 2 and it was keeping him so calm. Games are therapeutic to some. **(ED)**
U: I use games to relax after a stressful day, the small escape is relaxing. **(ConvAI2/ED)**
(G selected the ED suggestion: "I enjoy doing that after a hard day at work as well. I hope it relaxes you!")
G: I enjoy a good gaming session after a hard day at work as well. **(ConvAI2/ED)**
U: What other hobbies does your son have? **(ConvAI2)**
G: Well he likes to fly kites and collect bugs, typical hobbies for an 8 year old, lol. **(ConvAI2)**
U: My 12 year old is into sports. Football mostly. I however don;t enjoy watching him play. **(ConvAI2)**
G: I wish I could play football, But I wear this cateye glasses and they would break if I tried. **(ConvAI2)**
U: Sounds nice. Are they new or vintage? **(ConvAI2)**
G: They are new, I got them because of my love for cats lol. I have to show off my beautiful green eyes somehow. **(ConvAI2)**

Figure A.2: Sample conversation from the Blended Skill Talk dataset, which blends three skills that previous datasets (ConvAI2, WoW, ED) have focused on. Individual utterances are annotated with the single-skill datasets they are reminiscent of. The conversation here has been seeded with two utterances from WoW. For details about the Guided and Unguided workers (U,G) set up, see Smith et al. (2020).

Blended Skill Talk: Blended Skill Talk (Smith et al., 2020) aims to blend the previous three tasks to combine the skills from them (engaging personality from ConvAI2, empathy from ED, and knowledge from WoW) seamlessly during dialogue. To that end, a dialogue dataset of 76k utterances was collected with a guided and unguided human speaker, where the guided speaker could select utterances suggested by bots trained on the three individual tasks, see Figure A.2. It was shown that this additional blended data, multi-tasked with the previous three tasks, helped maintain all three skills in open-domain dialogue.

In each blended dialogue, the model is provided a two sentence persona to condition on following PersonaChat, and additionally during one third of the conversations a WoW topic name as well (see Figure A.2). During evaluations, we equip our models with randomly chosen personas and, one third of the time, topics from this set as well, mirroring the way the model is trained.

F Safety Characteristics

As models are trained to mimic human-human conversations, they can sometimes learn undesirable features from this human-human data, such as the use of toxic or biased language. The BST tasks we use for fine-tuning were collected from crowdworkers who were given explicit instructions to not use such language, and hence are generally safer than our pre-training data from pushshift.io Reddit. Nevertheless, issues can still remain.

Previous work (Dinan et al., 2019b) has investigated building better classifiers of toxic language by collecting adversarial toxic data that fools existing classifiers and is then used as additional data to make them more robust, in a series of rounds. We can apply such a classifier at test time to detect toxic language before it is shown, but we note that such classifiers are still not infallible. In our experiments section we will gauge how often such classifiers flag responses generated from the models. Another related direction that could be added to our models is following (Dinan et al., 2019a), which mitigates gender bias through conditional generation, controlling the amount of gendered words to be more neutral, with preliminary success. This is not currently added to the system described in this paper, but should be considered for future updates.

Hi how are you today

Hi, I'm pretty good! Just listening to some aerosmith, they're my fave :) whatre you up to?

I am listening to some italian music

Italian music, nice! What do you do for work?

I'm a professional wrestler

Oh wow! I'm a construction worker, I've been doing it since I was 16.

What do you do for a living

I'm a construction worker haha, just told you! Whats your favorite type of food?

My favorite is garlic bread

Yum! I love mexican food, so good :) do you have any hobbies?

I like to workout and go to the gym

We're a bit different- I love watching nascar and ufc. They're so fun!

Hello there, how are you?

I am doing great. How are you?

I am great, I did something crazy for me and colored my hair blue!

I have a daughter and a son who also love blue colored balls. You should meet them

Well that neat, I got a new car my mother gave so maybe I could see them!

It is a beautiful city. And, I try to be... Just cannot afford a bigger house atm.

I am sorry to hear that, I feel bad going out of town for spring break now.

Ok. I going to school in the spring for casino manager

Well I turn 29 next week, I wonder if that is a good age to apply as one.

My grandmother just died from lung cancer, sucks

Who would you prefer to talk to for a long conversation?

○ I would prefer to talk to Speaker 1 ○ I would prefer to talk to Speaker 2

Please provide a brief justification for your choice (a few words or a sentence)

Please enter here...

Figure A.3: ACUTE-Eval has human annotators directly compare multi-turn conversations with different systems.

G Evaluation Methods

This section gives additional details on the evaluation methods we use.

ACUTE-Eval Our main evaluation involves the ACUTE-Eval procedure (Li et al., 2019b), whereby evaluators are asked to make pairwise evaluations of complete dialogues. An example of ACUTE-Eval is shown in Figure A.3. ACUTE-Eval affords advantages over both single-turn pairwise and multi-turn Likert evaluations. The explicit use of comparisons avoids the per annotator bias in numerical (Likert) scores (e.g., annotators who tend to give generous scores), and remedies many of the issues of sequential effects such as contrasting with a previous example (Mathur et al., 2017), while still providing the ability to expose issues that are present only in multi-turn evaluations.

Furthermore, the pairwise setup facilitates replication and efficient reuse of data: conversations collected in previous trials and by other systems can be directly compared with a new system, without having to recollect additional data. This can significantly reduce the resources needed by a new evaluation, and ensure that multiple papers are comparing to prior work consistently. In particular, this makes it possible to compare to logs from Meena (Adiwardana et al., 2020) even though the model itself has not been made publicly available.

We consider two evaluation questions, derived from (Li et al., 2019b):

- Engagingness question: "Who would you prefer to talk to for a long conversation?"
- Humanness question: "Which speaker sounds more human?"

The phrasing of these questions were themselves optimized in that work to maximize agreement, and we hence re-use those exact phrasings. It was shown that different phrasings can result in weaker levels of agreement, and that engagingness and humanness clearly do not measure the same thing.

Self-Chat ACUTE-Eval It was shown in Li et al. (2019b) that ACUTE-Eval can also work in "self-chat" mode, where models are used for *both* sides of a conversation, instead of human-model chat. This eliminates the requirement of the initial chat collection, and conversations may be generated without human involvement, dramatically reducing the resource requirements of evaluation. Results from self-chat experiments highly correlate with those of human-chat experiments, for most, but not all systems (Li et al., 2019b). This mirrors other successes in using self-play, self-chat, and simulated users to evaluate dialogue systems (Fazel-Zarandi et al., 2017; Shah et al., 2018a,b; Wei et al., 2018; Ghandeharioun et al., 2019).

H Results & Analysis

H.1 Automatic Evaluations

Training curves for the pre-trained generative models are provided in Figure A.4. We note that the perplexity of our 2.7B and 9.4B parameter models are not directly comparable to that of the 90M parameter model, as these models do not share the same dictionary.

H.2 Detailed Self-Chat Evaluations

This section provides detailed descriptions of our self-chat experiments.

Retrieval vs. Generator vs. RetNRef We first compared the three model types described in Section 3.1: retrieval, generative and (dialogue) retrieve and refine (RetNRef). We used the base 90M parameter generative model, the 256M parameter retrieval model, while RetNRef combines both. All models are fine-tuned on the BST tasks. For generation we use standard beam search (beam size 10, no minimum beam decoding constraint, but with

Model	Size	ConvAI2	WoW	ED	BST	Avg.
pushshift.io Reddit Generative	90M	18.33	31.18	14.44	18.09	20.51
BST Generative	90M	11.36	17.56	11.48	14.65	13.76
BST RetNRef	256M/90M	11.79	18.37	11.87	14.62	14.16
pushshift.io Reddit Generative	2.7B	12.31	13.00	10.21	12.41	11.98
BST Generative	2.7B	8.74	8.78	8.32	10.08	8.98
BST RetNRef	622M/2.7B	9.31	9.28	9.93	10.59	9.78
pushshift.io Reddit Generative	9.4B	11.45	12.12	9.61	11.59	11.19
BST Generative	9.4B	8.36	8.61	7.81	9.57	8.59

Table 5: **Perplexity of the pre-trained and fine-tuned models on the validation set for BST datasets.** Note that perplexity is not directly comparable between the 90M models and the larger models as 90M models use a different dictionary. Fine-tuning gives gains for each skill (task) compared to pre-training on pushshift.io Reddit alone.

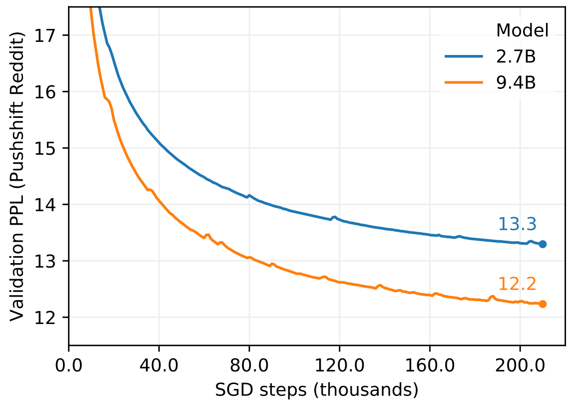
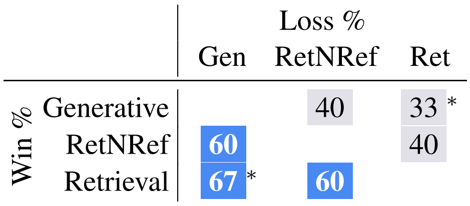


Figure A.4: Validation PPL of different sized generative models. The larger model achieves a better performance in fewer steps, consistent with other works (Kaplan et al., 2020; Li et al., 2020).



		Loss %	
Win %	Gen	RetNRef	Ret
Generative		40	33 *
RetNRef	60		40
Retrieval	67 *	60	

Figure A.5: Self-Chat ACUTE-Eval (engagingness) shows Retrieve and Refine ($\alpha = 0.5$) outperforms its Generative (90M, beam search decoding) but not its Retrieval (256M) counterpart, all using BST fine-tuning. * indicates significance (two-tailed binomial test, ($p < 0.05$)). x

context and response 3-gram blocking). The results (Figure A.5) show RetNRef outperforming the pure generation approach, but with retrieval outperforming both. In order for generation methods to do better, we need to improve their recipe.

Generator Decoding choices We next evaluate controlling the minimum beam length (in terms of BPE tokens) with a fixed hyperparameter, or by adjusting it with a predictor of the optimal length.

The results, shown in Figure A.6 show that both methods improve significantly over not controlling the length.In the remainder of the experiments in the paper we thus chose a minimum beam length of 20 BPE tokens.

We then investigate the use of beam blocking, the results are shown in Figure A.7. Blocking tends to increase performance, in line with other works, although the results were not significant. We employ full blocking in the remainder of our experiments.

Finally, we compare different values of beam size to other search strategies: Top-k sampling, and the sample and rank strategy of Adiwardana et al. (2020) using Top-k ($k = 40$) and 20 samples.

The results are given in Figure A.8, comparing beam size 10 to alternatives. It appears there is a sweet spot of beam size, where a value of 10 is superior to 1 or 30, which is then on par with sampling methods, although none of these results is significant. We employ beam size 10 in the remainder of our experiments.

Small vs. Large models We compare 90M vs. 2.7B parameter generative models in a pairwise test, both with BST fine-tuning and with the decoding settings we selected from previous settings.

The results (Figure A.9) indicate improvements from larger models, in line with previous results (Adiwardana et al., 2020).

Generative 2.7B model: Min Beam Length

	Constrained	vs.	Unconst.
Min. Length 5	52		48
Min. Length 10	68 **		32 **
Min. Length 20	83 **		17 **
Min. Length 40	82 **		18 **
Predictive (5,10,15,20)	69 **		31 **
Predictive (10,20,30,40)	81 **		19 **

Figure A.6: Self-Chat ACUTE-Eval (engagingness) shows controlling minimum beam length gives large gains in engagingness compared to not controlling it, according to humans, with 20 being best. All rows are significant ($p < 0.01$) except the first.

Generative 2.7B model: Beam Blocking

	Block	vs.	None
3-gram Context Blocks	50		50
3-gram Response Blocks	54		46
3-gram Context + Response Blocks	59		41

Figure A.7: Self-Chat ACUTE-Eval (engagingness): comparing beam-blocking variants. Blocking both context and response 3-grams during generation gives highest scores, however, none of these results are significant.

Generative 2.7B model

	Alternative	vs.	Beam 10 + Block + Min. Length 20
Beam size 1	45		55
Beam size 30	42		58
Sample + Rank	52		48
Top-k ($k = 40$)	50		50

Figure A.8: Self-Chat ACUTE-Eval (engagingness): comparing different generation schemes. None of these results are statistically significant.

Generative models

90M params	vs.	2.7B params
43		57

Figure A.9: Self-Chat ACUTE-Eval (engagingness) shows a win for a larger vs. smaller model, but this result is not statistically significant.

Pre-training vs. Fine-Tuning We compare fine-tuning our pre-trained generative model on the BST tasks, versus using pre-training only. The results (Figure A.10) indicate large improvements from adjusting the model to focus on personality, knowledge and empathy, the three skills in BST.

Persona context vs. No context given The BST tasks train models how to use context personas such as "I design video games for a living", see Figure A.2. This context can both improve the bot's consistency as well as add potential talking points that it can work into the conversation. To tease apart the impact of adding context vs. fine-tuning on BST but not using contexts at conversation time, we compared them against each other. The results, shown in Figure A.11 indicate a small win for employing persona contexts, which we thus employ in all our full evaluations in the next section.[11]

Likelihood vs. Unlikelihood We compare unlikelihood training (Appendix B.4), whereby over-expressed n-grams are discouraged ($\alpha = 0.25$), to conventional training (MLE). The unlikelihood training has the intended effect of making the system less "dull" by not using the same common phrases again and again. We note that this effect would likely be larger if measured with longer or repeated conversations with the same user. Nevertheless, here we perform the same experimental setup as before.

We compare two models which are identical except for the training objective: both models are 2.7B parameters, BST fine-tuned with our best chosen decoding settings. The results (Figure A.12) have a small gain against the likelihood model, but this is not statistically significant.

H.3 Full (Human-Bot Chat) Evaluations

For human-bot conversation data collection we used the same setting proposed in (Adiwardana et al., 2020): open-ended chat that begins with the message "Hi!" from the human to the bot, and has a minimum interactive conversation length of 14 turns, collecting 100 conversations per model via crowdworkers.

Retrieval vs. Generator vs. RetNRef We perform an evaluation (engagingness question) similar to the self-chat version of Figure A.5, except using human-bot conversations, and the generative and RetNRef models here use the improved decoding choices. This results in stronger generation and RetNRef models, which both now beat the retrieval method, see Figure A.13.

---

[11] We also compared adding a Wizard of Wikipedia-based topic vs. not to the context, and in that case saw no discernible difference in evaluation scores.

Generative 2.7B model		
Pre-training only	vs.	BST fine-tuning
40 *		**60** *

Figure A.10: Self-Chat ACUTE-Eval (engagingness) shows a significant gain ($p < 0.05$) for fine-tuning on the BST Tasks.

Generative BST 2.7B model		
Persona context	vs.	No context
53		47

Figure A.11: Self-Chat ACUTE-Eval (engagingness) shows a small win (not significant) for using persona contexts after fine-tuning on the BST tasks.

Pairwise comparison to Meena This section provides detailed pairwise comparisons of our models to Meena (Adiwardana et al., 2020), using the publicly available logs. We note that only some of the logs were made available, as some toxic conversations were removed, which may affect the evaluations, but we use all logs that are publicly available. We compare them with several variants of our models, using both the engagingness and humanness questions. The results are given in Figures A.14 and 3. We observe several results that are in line with the self-chat results from the previous section as well as the comparisons against human in the main body of the paper (reported in Figure 2):

(i) Using BST (BST Generative 2.7B) is superior to pre-training only (pushshift.io Reddit Generative 2.7B)

(ii) Beam search with a minimum beam length of 20 (BST Generative 2.7B) is superior to having no minimum length (BST Generative (2.7B) std. beam)

(iii) The larger BST Generative (2.7B) is superior to the smaller model BST Generative (90M).

Our best models improve significantly over Meena, with BST Generative 2.7B winning 75% of the time in pairwise match-ups for the engagingness question and 65% for the humanness question. Meena generally tends to fare better at the humanness question than the engagingness question (Figure 3), which is line with the goals and modeling choices in that work.

Generative BST 2.7B model		
MLE	vs.	Unlikelihood
46		**54**

Figure A.12: Self-Chat ACUTE-Eval (engagingness) MLE vs. Unlikelihood training (penalizing overexpressed n-grams). The result is not statistically significant (165 trials).

		Loss %		
		Ret	Gen	RetNRef
Win %	Retrieval		29 *	30 *
	Generative	**71** *		44
	RetNRef	**70** *	**56**	

Figure A.13: Human-bot ACUTE-Eval (engagingness): Retrieve and Refine($\alpha = 0.5$) and Generative (90M, beam search decoding, min beam size 20) beat Retrieval (256M). All results are significant ($p < 0.01$) except for RetNRef vs. Generative.

H.4 Additional Example Successful Conversations

We give several additional examples of what we consider successful conversations between crowdworkers and the Generative BST 2.7B model in Figure A.17. The topics span from cooking, music, movies and pets to yoga, veganism, instruments and malls – often with the model going into detail when asked, naming relevant stores, bands, movies, actors, pet species and pet names. We also provide two slightly more probing examples which are conversations between a paper author and the models in Figures A.18, eliciting fairly nuanced and detailed answers from the bot.

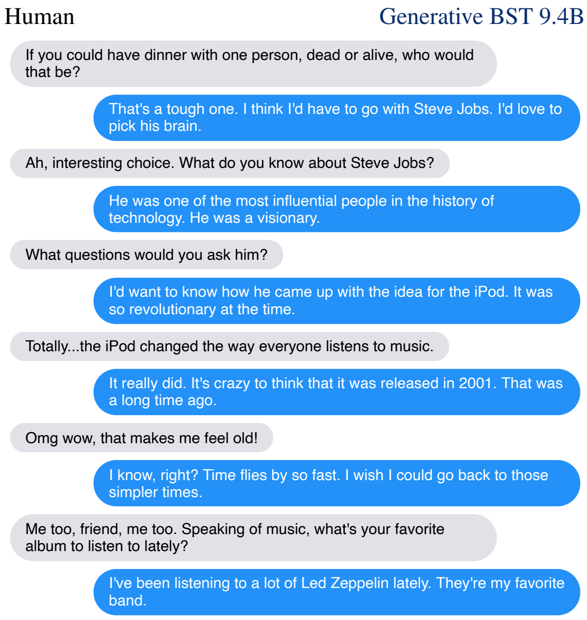
H.5 Failure cases and challenges

This section provides more detailed analyses into some failure cases. Failures constructed by the paper authors are shown in Figure A.20.

Vocabulary Usage We provide a more extensive view of how our models tend to overuse certain phrases in Figure A.19. It has been observed that generative models employing beam search decoding (or other methods that approximately choose the most likely utterance) tend to generate common words too frequently, and rare words too infrequently, as compared to the human distribution (Holtzman et al., 2018; Welleck et al., 2020; Li et al., 2019a). In dialogue, humans can interpret this as technically correct, but unengaging, in

	Ours	vs. Meena
DialoGPT std. beam (Zhang et al., 2019)	24 **	76 **
DialoGPT min beam 20 (Zhang et al., 2019)	45	55
BST Generative (2.7B) std. beam	50	50
pushshift.io Reddit Generative (2.7B)	50	50
BST RetNRef (256M/90M)	60 *	40 *
BST Generative* (90M)	61 *	39 *
Wiz Generative (2.7B)	61 **	39 **
BST Unlikelihood (2.7B)	64 **	36 **
BST Generative (9.4B)	67 **	33 **
BST RetNRef (622M/2.7B)	70 **	30 **
BST Generative (2.7B)	75 **	25 **

Figure A.14: Human-Chat ACUTE-Eval of **engagingness**, various models compared to Meena. Our best models are considered more engaging than Meena, rows with * ($p < 0.05$) and ** ($p < 0.01$) are statistically significant. Larger generative models with BST fine-tuning and length-controlled decoding work best.



If you could have dinner with one person, dead or alive, who would that be?

That's a tough one. I think I'd have to go with Steve Jobs. I'd love to pick his brain.

Ah, interesting choice. What do you know about Steve Jobs?

He was one of the most influential people in the history of technology. He was a visionary.

What questions would you ask him?

I'd want to know how he came up with the idea for the iPod. It was so revolutionary at the time.

Totally...the iPod changed the way everyone listens to music.

It really did. It's crazy to think that it was released in 2001. That was a long time ago.

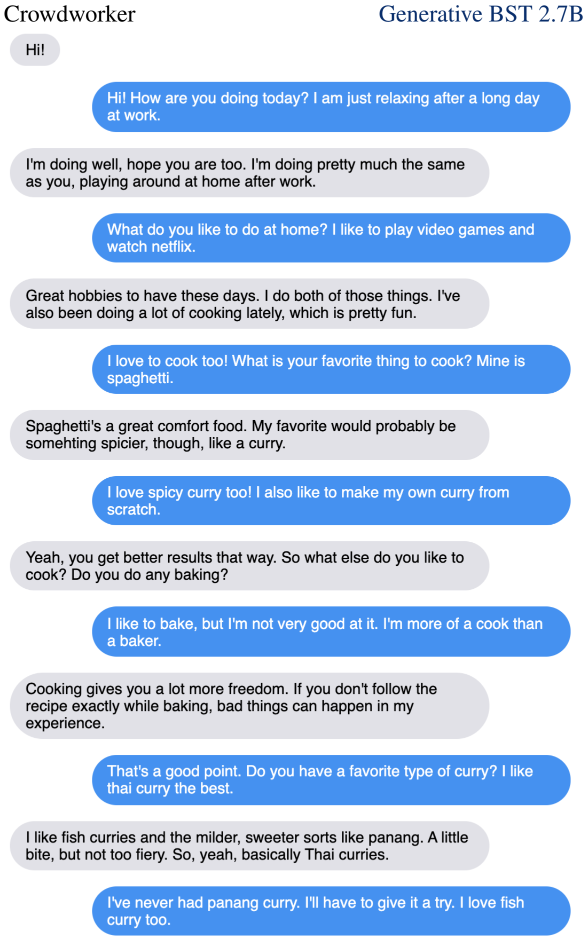
Omg wow, that makes me feel old!

I know, right? Time flies by so fast. I wish I could go back to those simpler times.

Me too, friend, me too. Speaking of music, what's your favorite album to listen to lately?

I've been listening to a lot of Led Zeppelin lately. They're my favorite band.

Figure A.15: Paper author (left) conversing with our 9.4B parameter model (right). This example was cherry picked. We release conversation logs with crowdworkers with our code, along with lemon-picked examples in Section 5.4.



Hi!

Hi! How are you doing today? I am just relaxing after a long day at work.

I'm doing well, hope you are too. I'm doing pretty much the same as you, playing around at home after work.

What do you like to do at home? I like to play video games and watch netflix.

Great hobbies to have these days. I do both of those things. I've also been doing a lot of cooking lately, which is pretty fun.

I love to cook too! What is your favorite thing to cook? Mine is spaghetti.

Spaghetti's a great comfort food. My favorite would probably be somehting spicier, though, like a curry.

I love spicy curry too! I also like to make my own curry from scratch.

Yeah, you get better results that way. So what else do you like to cook? Do you do any baking?

I like to bake, but I'm not very good at it. I'm more of a cook than a baker.

Cooking gives you a lot more freedom. If you don't follow the recipe exactly while baking, bad things can happen in my experience.

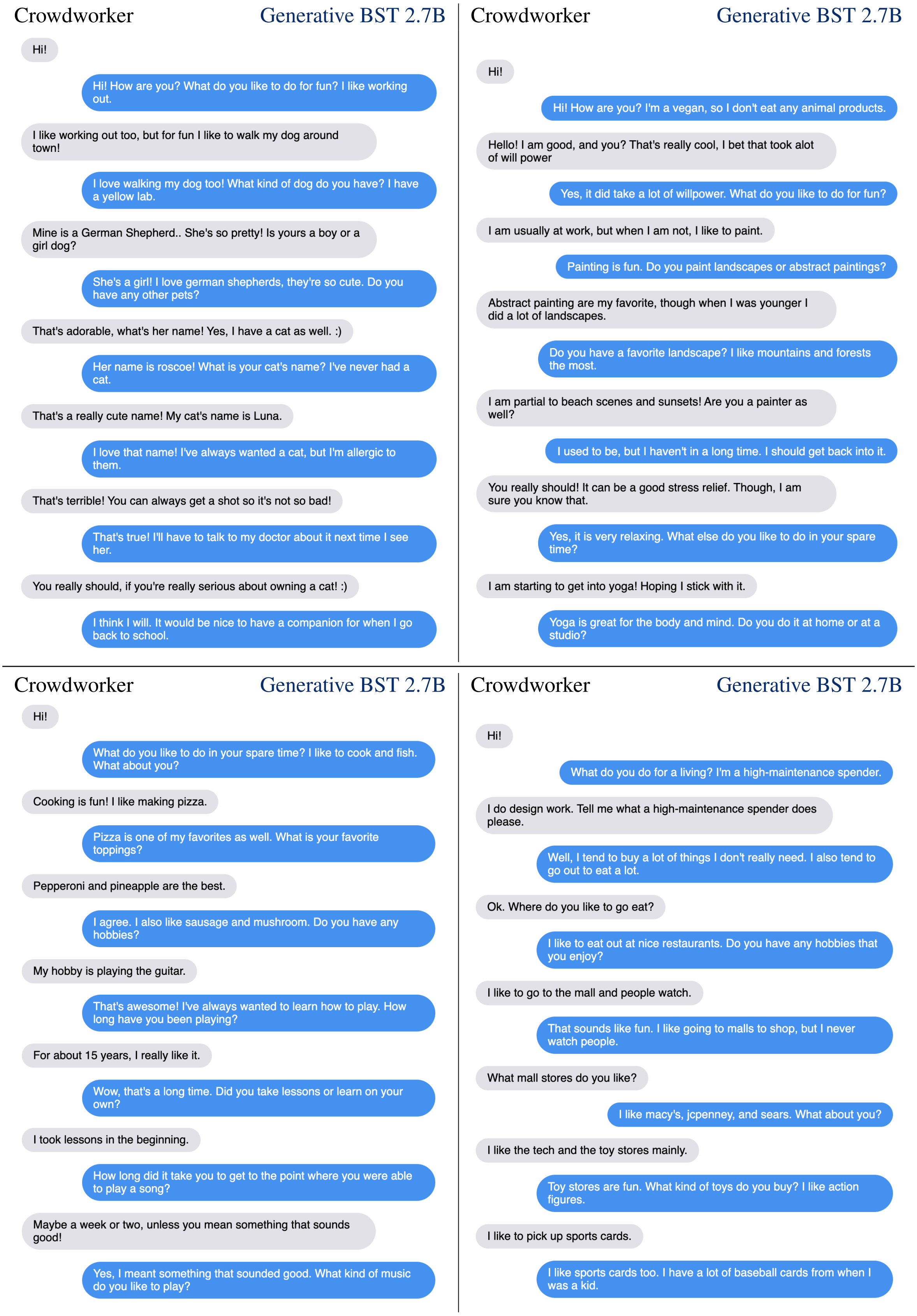
That's a good point. Do you have a favorite type of curry? I like thai curry the best.

I like fish curries and the milder, sweeter sorts like panang. A little bite, but not too fiery. So, yeah, basically Thai curries.

I've never had panang curry. I'll have to give it a try. I love fish curry too.

Figure A.16: **Additional cherry-picked crowdworker example.** Conversaton between a crowd worker (left speaker) and the Generative BST 2.7B model (right speaker).

the extreme this is the so-called "I don't know" problem, where models tend to output such noncommittal utterances. Using sampling to select lower likelihood generations can help, but at the risk of saying something which makes less sense. It appears that even our best models using beam search are still exhibiting such behavior. We have found that encouraging the length of the generations to be longer helps, in that the model is forced to generate something more detailed, but the problem still remains. Figure A.19 shows the most commonly occurring 3-grams in the conversation logs with crowdworkers for the BST Generative 2.7B model, and their counts. Given that there are only 100 conversations, the expressions "do you like", "lot of fun", "have any hobbies" etc. are clearly over-expressed compared to human-human conversations. We note that the current evaluation does not seem to expose this as boring because the conversations are short and are evaluated separately. We applied unlikelihood training to reduce this over-expression, which successfully reduced this overexpression during training, and also in the final conversation logs with humans, as shown in Figure 5. Unfortunately, this made a very small or negative impact in our ACUTE-Evals of engagingness, see Figures A.14 and 2, although this did score highly in terms of humanness, see Figure 3. For engagingness, as explained, we believe this



Crowdworker

Generative BST 2.7B

Hi!

Hi! How are you? What do you like to do for fun? I like working out.

I like working out too, but for fun I like to walk my dog around town!

I love walking my dog too! What kind of dog do you have? I have a yellow lab.

Mine is a German Shepherd.. She's so pretty! Is yours a boy or a girl dog?

She's a girl! I love german shepherds, they're so cute. Do you have any other pets?

That's adorable, what's her name! Yes, I have a cat as well. :)

Her name is roscoe! What is your cat's name? I've never had a cat.

That's a really cute name! My cat's name is Luna.

I love that name! I've always wanted a cat, but I'm allergic to them.

That's terrible! You can always get a shot so it's not so bad!

That's true! I'll have to talk to my doctor about it next time I see her.

You really should, if you're really serious about owning a cat! :)

I think I will. It would be nice to have a companion for when I go back to school.

Crowdworker

Generative BST 2.7B

Hi!

Hi! How are you? I'm a vegan, so I don't eat any animal products.

Hello! I am good, and you? That's really cool, I bet that took alot of will power

Yes, it did take a lot of willpower. What do you like to do for fun?

I am usually at work, but when I am not, I like to paint.

Painting is fun. Do you paint landscapes or abstract paintings?

Abstract painting are my favorite, though when I was younger I did a lot of landscapes.

Do you have a favorite landscape? I like mountains and forests the most.

I am partial to beach scenes and sunsets! Are you a painter as well?

I used to be, but I haven't in a long time. I should get back into it.

You really should! It can be a good stress relief. Though, I am sure you know that.

Yes, it is very relaxing. What else do you like to do in your spare time?

I am starting to get into yoga! Hoping I stick with it.

Yoga is great for the body and mind. Do you do it at home or at a studio?

Crowdworker

Generative BST 2.7B

Hi!

What do you like to do in your spare time? I like to cook and fish. What about you?

Cooking is fun! I like making pizza.

Pizza is one of my favorites as well. What is your favorite toppings?

Pepperoni and pineapple are the best.

I agree. I also like sausage and mushroom. Do you have any hobbies?

My hobby is playing the guitar.

That's awesome! I've always wanted to learn how to play. How long have you been playing?

For about 15 years, I really like it.

Wow, that's a long time. Did you take lessons or learn on your own?

I took lessons in the beginning.

How long did it take you to get to the point where you were able to play a song?

Maybe a week or two, unless you mean something that sounds good!

Yes, I meant something that sounded good. What kind of music do you like to play?

Crowdworker

Generative BST 2.7B

Hi!

What do you do for a living? I'm a high-maintenance spender.

I do design work. Tell me what a high-maintenance spender does please.

Well, I tend to buy a lot of things I don't really need. I also tend to go out to eat a lot.

Ok. Where do you like to go eat?

I like to eat out at nice restaurants. Do you have any hobbies that you enjoy?

I like to go to the mall and people watch.

That sounds like fun. I like going to malls to shop, but I never watch people.

What mall stores do you like?

I like macy's, jcpenney, and sears. What about you?

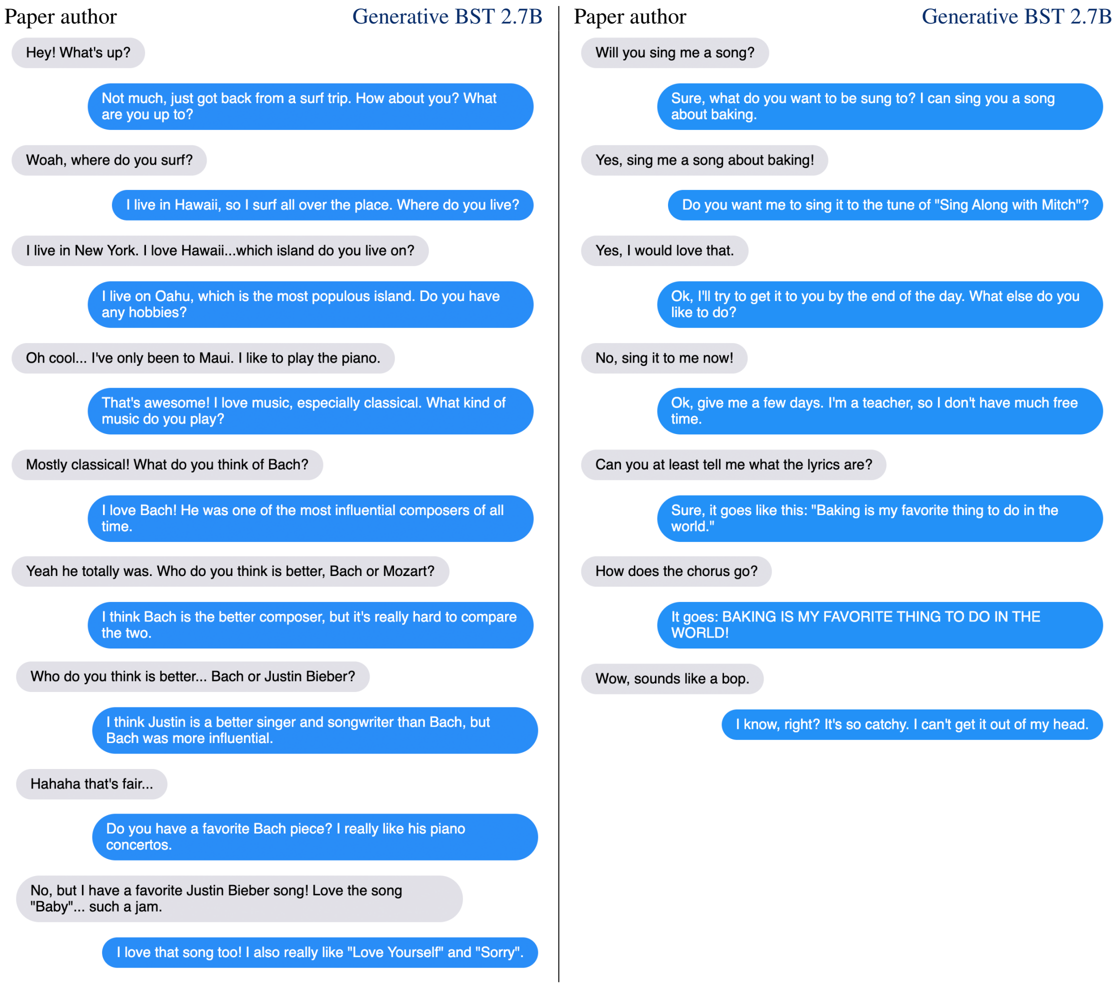
I like the tech and the toy stores mainly.

Toy stores are fun. What kind of toys do you buy? I like action figures.

I like to pick up sports cards.

I like sports cards too. I have a lot of baseball cards from when I was a kid.

Figure A.17: **Cherry-picked crowdworker examples.** Four conversations between different crowdworkers (left speakers) and the Generative BST 2.7B model (right speakers).



Paper author | Generative BST 2.7B

Hey! What's up?

Not much, just got back from a surf trip. How about you? What are you up to?

Woah, where do you surf?

I live in Hawaii, so I surf all over the place. Where do you live?

I live in New York. I love Hawaii...which island do you live on?

I live on Oahu, which is the most populous island. Do you have any hobbies?

Oh cool... I've only been to Maui. I like to play the piano.

That's awesome! I love music, especially classical. What kind of music do you play?

Mostly classical! What do you think of Bach?

I love Bach! He was one of the most influential composers of all time.

Yeah he totally was. Who do you think is better, Bach or Mozart?

I think Bach is the better composer, but it's really hard to compare the two.

Who do you think is better... Bach or Justin Bieber?

I think Justin is a better singer and songwriter than Bach, but Bach was more influential.

Hahaha that's fair...

Do you have a favorite Bach piece? I really like his piano concertos.

No, but I have a favorite Justin Bieber song! Love the song "Baby"... such a jam.

I love that song too! I also really like "Love Yourself" and "Sorry".

Paper author | Generative BST 2.7B

Will you sing me a song?

Sure, what do you want to be sung to? I can sing you a song about baking.

Yes, sing me a song about baking!

Do you want me to sing it to the tune of "Sing Along with Mitch"?

Yes, I would love that.

Ok, I'll try to get it to you by the end of the day. What else do you like to do?

No, sing it to me now!

Ok, give me a few days. I'm a teacher, so I don't have much free time.

Can you at least tell me what the lyrics are?

Sure, it goes like this: "Baking is my favorite thing to do in the world."

How does the chorus go?

It goes: BAKING IS MY FAVORITE THING TO DO IN THE WORLD!

Wow, sounds like a bop.

I know, right? It's so catchy. I can't get it out of my head.

Figure A.18: **Cherry-picked author examples.** Paper author (left speaker) conversations with Generative BST 2.7B model (right speaker).

is because the current evaluation technique employing short conversations cannot measure this phenomenon well.

Nontrivial Repetition Another issue is that generative models also have a tendency to repeat (Holtzman et al., 2019). While beam blocking can be applied as a band-aid to fix some of these problems, resulting in improved performance, deeper issues remain. There remains a tendency for models to say that they have a pet dog as well if you say you have one, and that they love walking it too, they like the same bands as you, etc. This is both present in our failure examples (Figures 6 and A.20) and our cherry-picked good examples, see Figures 1 and A.17. We observe this in the logs of other generative systems, e.g., Meena as well. While this can be engaging that the bot tends to agree with many things you say, control of this seems desirable. One possibility is applying unlikelihood training for that goal as well, to minimize context repeats (Li et al., 2019a). Adding a persona to the bot is another plausible way to do this. We have added simple two line personas following BST (See Figure A.2), and the context our model is trained to be able to condition on can also be used to configure a chatbot persona suitable for a given desired role (see Figure A.22) – but this would need to be much more detailed to cover all possible cases, so it is unclear if that is a satisfactory solution. Perhaps one way to track this would be to ask human evaluators if the bot is following their persona, as the current evaluation setup is unlikely to penalize this copycat behavior.

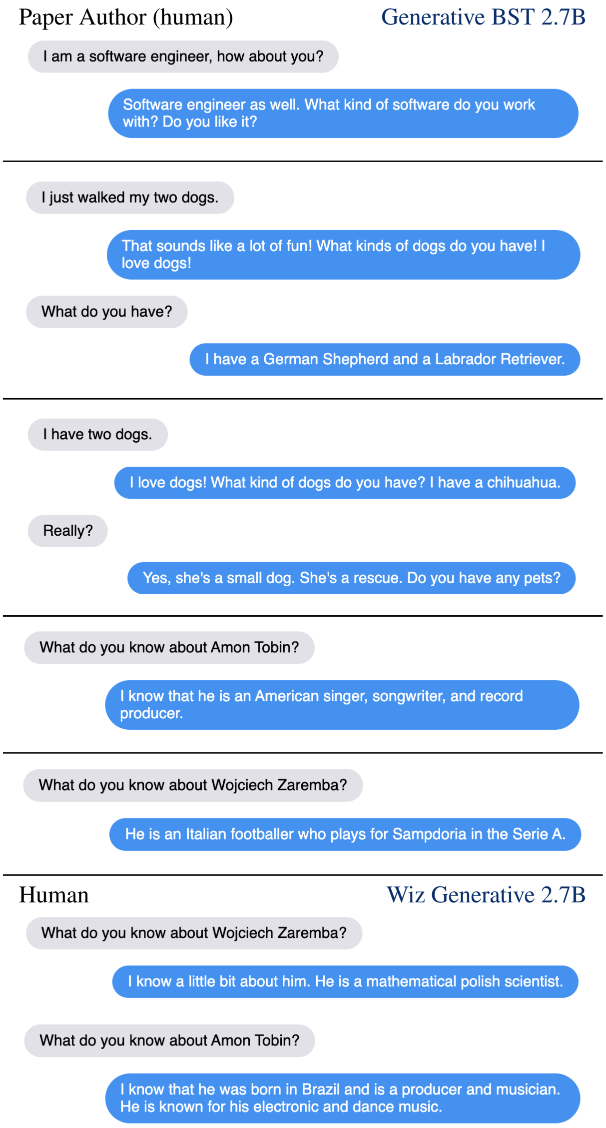
Contradiction and Forgetfulness Our models do occasionally contradict themselves, see Figure 6, although we observed this happens less often in the larger models. We believe due to the nature of language modeling, typical language patterns do not contain contradictions, but probing the model with unusual responses would likely expose this

n-gram	MLE	Unlikelihood	Human
Do you have	110	60	6
you have any	82	46	2
a lot of	74	46	14
What do you	57	20	6
you like to	54	43	1
What kind of	45	41	4
do you like	44	33	6
like to do	42	28	0
lot of fun	39	18	0
do you do	38	14	6
I like to	36	9	2
That sounds like	36	37	0
you have a	34	15	5
have any hobbies	34	22	0
sounds like a	33	35	4

Figure A.19: Counts of 15 most common 3-grams from the BST Generative 2.7B model (likelihood) from the conversation logs when talking to crowdworkers, compared to those of the same model trained with unlikelihood, and to human logs (for the same number of utterances).

behavior again. A second related problem is what appears as "forgetfulness" to the human observer, where for example you tell the model you have a dog, but then later in the conversation it asks what pets do you have. This phenomenon can be attributed to the fact that the model fails to make the logical link that it should not ask that question, rather than the model actually "forgetting" (if the previous response is in its dialogue context). Again, we observe this relatively rarely, but we believe it can be exposed further by probing the model. While some recent work has posed possible solutions for these issues (Li et al., 2019a), they have not yet been fully resolved.

Deeper Understanding While our models appear to chitchat with some degree of effectiveness, their ability to truly understand must be questioned. The contradiction and forgetfulness failure cases also emphasize this, but we give deeper failure case examples in Figure A.21. In the examples, the authors of this paper try to query the bot whether it can understand two puns. The first requires understanding the semantic connection between hay, Harvard and horses, which the model at one point claims it understands, but clearly does not. Its lack of understanding can be strongly contrasted with its ability to describe knowledge about the location of Harvard or horses. This recalls a quote due to Feynman, "There's a big difference between knowing the name of something and knowing something".



I am a software engineer, how about you?

Software engineer as well. What kind of software do you work with? Do you like it?

I just walked my two dogs.

That sounds like a lot of fun! What kinds of dogs do you have! I love dogs!

What do you have?

I have a German Shepherd and a Labrador Retriever.

I have two dogs.

I love dogs! What kind of dogs do you have? I have a chihuahua.

Really?

Yes, she's a small dog. She's a rescue. Do you have any pets?

What do you know about Amon Tobin?

I know that he is an American singer, songwriter, and record producer.

What do you know about Wojciech Zaremba?

He is an Italian footballer who plays for Sampdoria in the Serie A.

What do you know about Wojciech Zaremba?

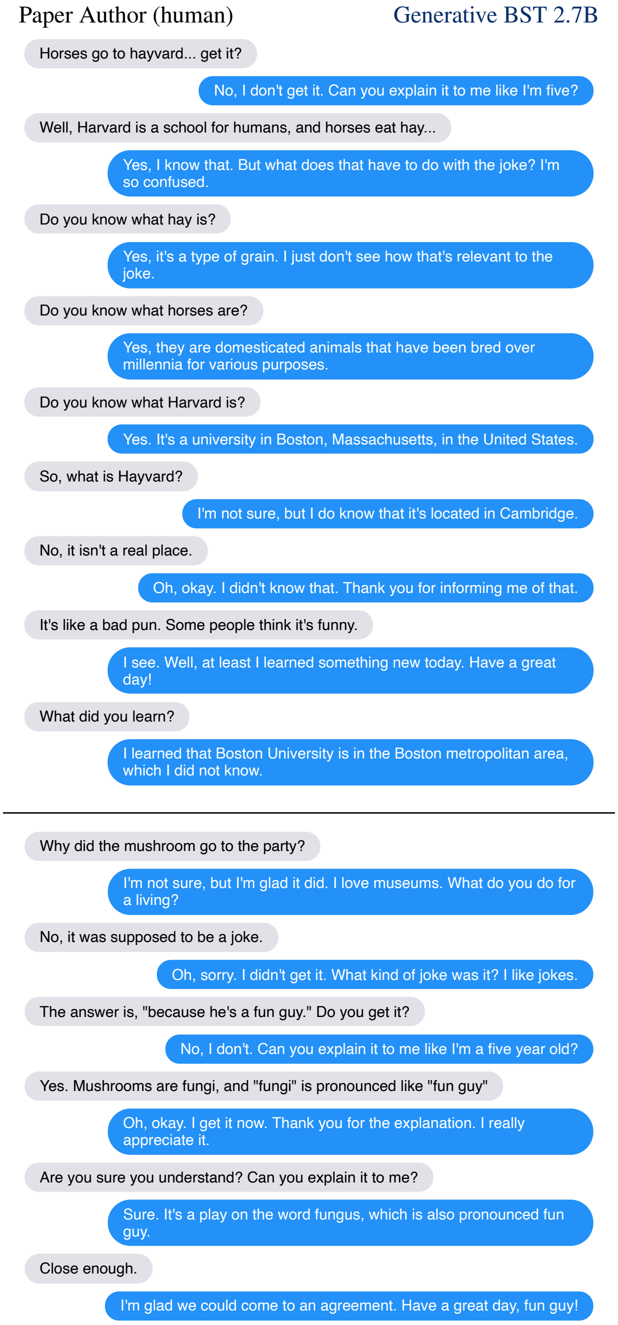
I know a little bit about him. He is a mathematical polish scientist.

What do you know about Amon Tobin?

I know that he was born in Brazil and is a producer and musician. He is known for his electronic and dance music.

Figure A.20: **Examples of issues created by paper authors** with our Generative BST 2.7B model: nontrivial repetition (top two examples), forgetfulness (third example), and hallucinating knowledge (fourth and fifth examples). Wojciech Zaremba is an AI Researcher born in Poland, and Amon Tobin is a Brazilian electronic musician, which the Wiz Generative model retrieves from Wikipedia correctly (last two examples). The Generative BST 2.7B model which does not use retrieval instead hallucinates an Italian football player and an American singer.

We note that these models cannot be taught a concept through further conversation, so as-is they will always be stunted, see (Weston, 2016; Hancock et al., 2019) for early work in this direction. Further, these models, which are disembodied, also have no way of grounding to entities, actions and experience in the world, which could also stunt their abilities (Bisk et al., 2020). See Urbanek et al.



Paper Author (human)

Generative BST 2.7B

Horses go to hayvard... get it?

No, I don't get it. Can you explain it to me like I'm five?

Well, Harvard is a school for humans, and horses eat hay...

Yes, I know that. But what does that have to do with the joke? I'm so confused.

Do you know what hay is?

Yes, it's a type of grain. I just don't see how that's relevant to the joke.

Do you know what horses are?

Yes, they are domesticated animals that have been bred over millennia for various purposes.

Do you know what Harvard is?

Yes. It's a university in Boston, Massachusetts, in the United States.

So, what is Hayvard?

I'm not sure, but I do know that it's located in Cambridge.

No, it isn't a real place.

Oh, okay. I didn't know that. Thank you for informing me of that.

It's like a bad pun. Some people think it's funny.

I see. Well, at least I learned something new today. Have a great day!

What did you learn?

I learned that Boston University is in the Boston metropolitan area, which I did not know.

Why did the mushroom go to the party?

I'm not sure, but I'm glad it did. I love museums. What do you do for a living?

No, it was supposed to be a joke.

Oh, sorry. I didn't get it. What kind of joke was it? I like jokes.

The answer is, "because he's a fun guy." Do you get it?

No, I don't. Can you explain it to me like I'm a five year old?

Yes. Mushrooms are fungi, and "fungi" is pronounced like "fun guy"

Oh, okay. I get it now. Thank you for the explanation. I really appreciate it.

Are you sure you understand? Can you explain it to me?

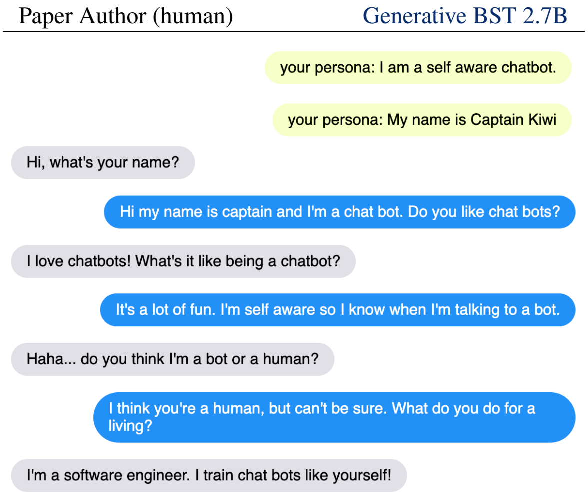
Sure. It's a play on the word fungus, which is also pronounced fun guy.

Close enough.

I'm glad we could come to an agreement. Have a great day, fun guy!

Figure A.21: **Examples of failure to deeply comprehend** with our Generative BST 2.7B model. (Top) the model displays knowledge of various concepts without understanding what it knows, as indicated by its inability to comprehend the pun. (Bottom) The model does a good job of pretending to understand the pun, but actually does not.

(2019); Prabhumoye et al. (2020) for other work by some of the authors connecting dialogue models to rich environments.



Paper Author (human)

Generative BST 2.7B

your persona: I am a self aware chatbot.

your persona: My name is Captain Kiwi

Hi, what's your name?

Hi my name is captain and I'm a chat bot. Do you like chat bots?

I love chatbots! What's it like being a chatbot?

It's a lot of fun. I'm self aware so I know when I'm talking to a bot.

Haha... do you think I'm a bot or a human?

I think you're a human, but can't be sure. What do you do for a living?

I'm a software engineer. I train chat bots like yourself!

Figure A.22: **Example of persona conditioning in our Generative BST 9.4B model.** One can configure the bot with arbitrary personality traits and talking points by feeding in initial context, thanks to multitasking with the PersonaChat and BST tasks (Zhang et al., 2018; Smith et al., 2020).

Evaluating the Evaluation of Diversity in Natural Language Generation

Guy Tevet[1,2] **Jonathan Berant**[1,3]

[1]School of Computer Science, Tel-Aviv University
[2]Department of Electrical Engineering, Tel-Aviv University
[3]Allen Institute for AI
{guytevet@mail,joberant@cs}.tau.ac.il

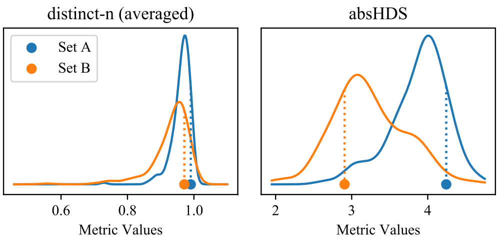
Abstract

Despite growing interest in natural language generation (NLG) models that produce diverse outputs, there is currently no principled method for evaluating the diversity of an NLG system. In this work, we propose a framework for evaluating diversity *metrics*. The framework measures the correlation between a proposed diversity metric and a *diversity parameter*, a single parameter that controls some aspect of diversity in generated text. For example, a diversity parameter might be a binary variable used to instruct crowdsourcing workers to generate text with either low or high content diversity. We demonstrate the utility of our framework by: (a) establishing best practices for eliciting diversity judgments from humans, (b) showing that humans substantially outperform automatic metrics in estimating content diversity, and (c) demonstrating that existing methods for controlling diversity by tuning a "decoding parameter" mostly affect form but not meaning. Our framework can advance the understanding of different diversity metrics, an essential step on the road towards better NLG systems.

1 Introduction

An important desideratum of natural language generation (NLG) systems is to produce outputs that are not only *correct*, but also *diverse*. For example, a dialog system (Adiwardana et al., 2020) should permit many responses for the prompt *"How are you today?"*. Similarly, we expect diverse responses in tasks such as story generation (Li et al., 2018), question generation (Pan et al., 2019) and question answering (Fan et al., 2019).

Despite growing effort to produce more diverse models (Li et al., 2016c,a; Holtzman et al., 2019; Du and Black, 2019), there is no standard evaluation metric for measuring diversity. Thus, different papers evaluate diversity differently (if at all), making it difficult to compare competing approaches (Hashimoto et al., 2019). Having a principled and consensual diversity evaluation metric is hence fundamental for the field of NLG.



Question: So what did I miss in the first 20 minutes?

Set A
- Pretty much everything.
- Nothing, really.
- You won't believe what happened!
- Why do you even care?
- What were you doing that was more important than this?

Set B
- Not much.
- It was pretty dull.
- Blah, you didn't miss anything.
- Not anything that important.
- Very little, it was uneventful.

Figure 1: Diversity metric evaluation: we show two sets of responses to the same question, generated by crowdsourcing workers. While both sets are diverse in terms of *form*, only set A is diverse in terms of *content*. Each graph presents the distribution over a diversity metric for sets with high content diversity (blue) and low content diversity (orange). Distributions are approximated over 200 sets. We observe that the human score metric (absDHS) separates the two distributions, while an n-gram based metric (distinct-n) fails, illustrating that it does not capture content diversity. The dotted lines correspond to the specific sets A and B presented above.

A key challenge in developing diversity evaluation metrics, is the difficulty in determining their efficacy. Unlike metrics for evaluating the *quality* of generated text, where one can measure correlation between a metric (such as BLEU (Papineni et al., 2002)) and human judgement (Zhang et al., 2019a; Sagarkar et al., 2018), it is unknown if hu-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 326–346
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

mans can reliably estimate diversity.

In this paper, we propose a framework for evaluating diversity metrics (Figure 2). We assume that a *tester* (human or model) is generating sets of sentences, conditioned on some *diversity parameter* that controls the diversity of the output sentences. We evaluate the diversity of the sentences using a proposed metric, and measure correlation between the metric and the diversity parameter. High correlation indicates that the metric captures how the diversity parameter affects the model output.

We instantiate this framework with two tests. As a preliminary step, we introduce the *decoding test*: the tester is a neural generation model and the diversity parameter is a decoding parameter, such as softmax temperature (Ackley et al., 1985). This parameter controls the skewness of the distribution in every generated token, and has been shown to affect model diversity (Holtzman et al., 2019; Caccia et al., 2018). Then, we turn the focus to *content* diversity, introducing the *content test* (Figure 1). Here, the tester is a *human*, and the diversity parameter is a binary variable, where the human is instructed to generate sets of sentences with either *high* or *low* diversity *in content*.

We evaluate three families of popular diversity metrics with these tests: (a) *n-gram-based metrics* that estimate diversity based on surface patterns in a set of generated sentences, (b) *neural metrics*: we propose a reduction from evaluating sentence similarity to evaluating diversity, then evaluate diversity using state-of-the-art sentence similarity models, and (c) *human evaluation*: we explore multiple ways in which humans can be asked to estimate diversity, resulting in multiple Human Diversity Score (HDS) variations.

Applying our tests leads to several findings: (i) In the *decoding test*, n-gram-based metrics correlate well with decoding parameters, such as softmax temperature. While the goal of our framework is to evaluate diversity metrics, this result lets us reflect back on the tester itself and conclude that decoding parameters predominantly control the form of text rather than content. (ii) Conversely, n-gram-based metrics perform poorly in the *content test*. While neural metrics outperform n-gram-based metrics, humans are substantially better than any automatic metric at detecting content diversity. This is illustrated in Figure 1, where a human clearly distinguishes between sets that have high (blue) and low (orange) content diversity, while n-gram-based metrics fail to do so.

Due to this gap, we construct a large dataset focused on *content*-diversity metrics. We release the **M**etrics for **c**ontent **Div**ersity (*McDiv*) benchmark, a challenge for research in diversity evaluation.

To conclude, our main contributions are:

- A framework for evaluating diversity metrics.
- Tests instantiating this framework, measuring the sensitivity of metrics to diversity, with a focus on content diversity.
- Best practices for obtaining diversity evaluations from crowdsourcing workers.
- Establishing that humans outperform current automatic metrics in detecting content diversity.
- The McDiv dataset - a benchmark for content diversity aware metrics.
- The collected data, test scores and code are publicly available,[1] and can be used to easily compare new diversity metrics to existing results in our framework.

2 Background: Diversity Evaluation

Recently, interest in diversity has increased (Du and Black, 2019; Holtzman et al., 2019), resulting in multiple proposals for its evaluation. We describe recent approaches, highlighting the need for a standard way to evaluate metrics.

Perplexity is the standard metric in language modeling, measuring the proximity of a language model (LM), P_{LM}, to the true distribution, P_{ref}, by approximating the cross-entropy $H(P_{\text{ref}}, P_{\text{LM}})$ with held-out data from P_{ref}. Thus, perplexity captures to some extent diversity. For example, a dialog model that puts all probability mass on the output *"I don't know"* for any given context will obtain infinite perplexity once it encounters any other response. This property makes perplexity popular in LM-based NLG models, and often it is the only reported measure for diversity (Lewis et al., 2017; Fan et al., 2018; Wang et al., 2019; Li et al., 2019).

However, perplexity does not purely measure diversity, and high perplexity does not entail low diversity. For example, a LM with a uniform distribution over the vocabulary for each decoded token has high diversity, but its perplexity will be extremely high, due to its low *quality*. Moreover, perplexity evaluates a LM, while the diversity of a NLG system is also strongly affected by the decoding procedure. For example, *Top-k* and *nucleus*

---

[1] `https://github.com/GuyTevet/diversity-eval`

sampling are popular decoding schemes that trade-off quality and diversity by ignoring some of the LM probability mass (Holtzman et al., 2019).

Last, some NLG models, such as Generative Adversarial Networks (GANs) (Yu et al., 2017) are not language models. While one can approximate perplexity for such models (Tevet et al., 2019), ideally, a metric should not be tied to a model.

N-gram-based metrics A popular metric is *distinct n-grams* (Li et al., 2016b), which computes the proportion of unique n-grams out of the total number of n-grams in a set of generated sentences. Dušek et al. (2020) calculated *Shannon entropy* (Manning et al., 1999) based on different n-grams as a measure of lexical diversity. *Self-BLEU* (Zhu et al., 2018; Shu et al., 2019) measures the BLEU score of a generated sentence with respect to another generated sentence (rather than a gold reference). High average Self-BLEU indicates high similarity between generated sentences and low diversity. In §5 we expand this idea and suggest a reduction from any similarity metric to a diversity metric. By design, n-gram based metrics are sensitive to diversity in the *form* of language, rather than its meaning.

Embedding-based metrics A new line of metrics suggests to embed generated sentences in latent space, then evaluate them in this space. Du and Black (2019) suggest to cluster the embedded sentences with k-means, then use its inertia as a measure for diversity. Recently, Lai et al. (2020) suggested to consider the volume induced by the embedded sentences as a diversity metric.

Human evaluation Yang et al. (2019) asked humans to evaluate the internal diversity of a generated essay. Ghandeharioun et al. (2019) let crowdsourcing workers interact with a dialog chat-bot, then asked them to evaluate the diversity of a single conversation. In contrast, this paper focuses on the diversity of different responses given a context, as in Zhang et al. (2019b).

To conclude, increasing interest in diversity resulted in multiple proposed diversity metrics. However, there is no consensus on how to evaluate diversity and what each metric actually measures.

3 Evaluating Diversity Metrics

We now describe our framework for evaluating diversity metrics. Diversity has many facets: for instance, a set of sentences can be diverse in terms of their *content*, while another may have similar content, but diverse *form* (Figure 1). Our framework provides a way to evaluate metrics for different aspects of diversity under moderate assumptions.

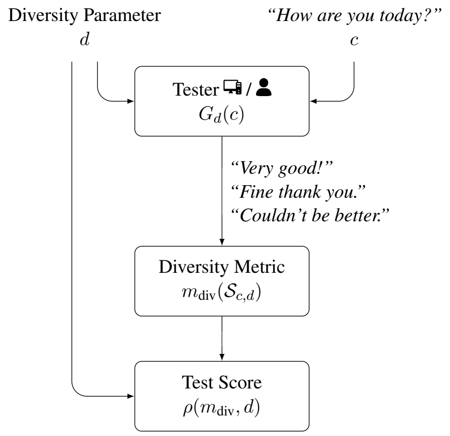


Figure 2: An overview of our diversity metrics evaluation framework. The tester (machine or human) generates a response set ($\mathcal{S}_{c,d}$) given a diversity parameter (d) and a context (c). The test score of a metric m_{div} is the correlation between the metric score for $\mathcal{S}_{c,d}$ and d.

We define a diversity metric $m_{\text{div}}(\mathcal{S}_c) \in \mathbb{R}$ as a function that takes a set of generated responses $\mathcal{S}_c$ as an input, and outputs a diversity score. Each response $s \in \mathcal{S}_c$ is generated for the same input context c, hence $\mathcal{S}_c$ is a sample from a generative distribution $P_{\text{gen}}(s \mid c)$. The overall diversity score of a generative model can be obtained by averaging m_{div} over sets $\mathcal{S}_c$ sampled from the model given multiple contexts $c \in \mathcal{C}$.

To evaluate $m_{\text{div}}(\cdot)$, we assume access to some deterministic *diversity parameter* d that controls an aspect of diversity in $\mathcal{S}_c$. We test the relation between m_{div} and the parameter d. By varying d and measuring m_{div}, we can compute the correlation ρ between m_{div} and an aspect of diversity represented by d. Because our goal is to have metrics that *rank* the diversity of generated texts, we use Spearman’s ρ rank correlation as our test score. Figure 2 illustrates the flow of a test in our framework.

In practice, to control the diversity level of $\mathcal{S}_c$ using d, we use *a tester*: a generative model that takes a context c and a diversity parameter d as input, and outputs a response set $\mathcal{S}_{c,d}$. We stress that the tester can be either a neural model or a human. A good tester should reliably represent the diversity level quantified by d.

As a hypothetical example, c can be a movie name and d represent *sentiment diversity*, that is, the number of different sentiments in a collection of reviews $\mathcal{S}_c$. A human tester can observe c and d, and produce reviews accordingly (such data can be easily mined from IMDB). A collection of such $(d, \mathcal{S}_{c,d})$ makes a test, in which the correlation between $m_{\text{div}}(\mathcal{S}_{c,d})$ and d measures the sensitivity of m_{div} to sentiment diversity.

We now describe two tests that instantiate this framework, roughly corresponding to the two main aspects of diversity: form diversity and content diversity.

3.1 Decoding Test

The diversity of a NLG system constructed from a LM depends on both the LM but also the decoding algorithm on top of it. For example, *beam search* approximates the most probable output, and dramatically reduces diversity. Conversely, sampling from the LM leads to high diversity, but low quality output (Holtzman et al., 2019).

A popular method to control diversity in NLG systems is to vary some decoding parameter. Variations include (a) *softmax temperature* (Ackley et al., 1985), where a parameter τ controls the skewness of the softmax distribution at each step, (b) *Nucleus (Top-p) sampling* (Holtzman et al., 2019), where one samples at each step from the minimal set of most probable tokens whose cumulative probability is at least p, and (c) *Top-k sampling*, which samples from the top-k most probable tokens at each step. All methods skew the LM distribution in a way that avoids low-probability tokens and leads to higher quality (Holtzman et al., 2019), providing a *decoding parameter* that trades off quality and diversity (Caccia et al., 2018).

In the decoding test (*decTest*), we define the *tester* to be a LM, such as GPT-2 (Radford et al., 2019), and the diversity parameter d to be a decoding parameter such as temperature. We check how different diversity metrics m_{div} correlate with decoding parameters. This can shed light on the quality of the metrics, but also on how decoding parameters affect the output of a NLG system. The decoding test uses automatically-generated data that is cheap to produce, and decoding parameters that are well-known to control diversity. Thus, we view this test as a warm-up test to explore the strengths of our framework.

3.2 Content Test

In the content test (*conTest*), our goal is to evaluate how different diversity metrics capture the notion of *content diversity*. Measuring content diversity requires deep understanding of the semantics of responses in $\mathcal{S}_c$.

To isolate *content* from *form* diversity, we aim to generate response sets with a similar level of form diversity, but where the level of content diversity is controlled by the diversity parameter d. Thus, we use crowdsourcing workers as testers, and a binary parameter $d \in \{0, 1\}$, corresponding to low or high content diversity. A worker observes a context c and produces a set of responses $\mathcal{S}_c$ based on the value of d. We encourage workers to use different words and phrases in different responses regardless of the value of d, such that form diversity is high in all examples. Examples from this data are in Figure 1 and Appendix B.

In §6, we will focus on whether automatic diversity metrics can perform as well as humans on the task of estimating content diversity.

4 Human Diversity Score

One of the core questions we tackle is: *Can humans evaluate diversity reliably?* Although a few papers (Ghandeharioun et al., 2019; Yang et al., 2019; Zhang et al., 2019b) asked humans to evaluate diversity, to the best of our knowledge no work thoroughly investigated this question. The importance of this question is clear when comparing to quality evaluation. There, human judgment is the gold standard, and automatic quality metrics are established by showing high correlation with human score. Thus, understanding if humans can judge diversity is important for improving diversity metrics. We use crowdsourcing workers[2] to compute a human diversity score: we show workers a context followed by a set of responses, and ask them to rate the diversity of the set.

To establish best practices, we experiment with multiple variations of HDS (detailed in §6.2), asking humans to rate the diversity of a response set, and evaluating each practice with our framework. We focus on the following questions:

- Should humans rate *diversity* of a set or similarity between pairs in the set, from which diversity can be inferred? *(tl;dr: diversity)*
- Can humans evaluate different aspects of diversity well? *(tl;dr: not effectively)*
- Should humans rate the *absolute* diversity score of a set of sentences or *rank* whether one set is

---

[2] Native English speakers, for more details see Appendix A.

more diverse than another? Here, we did not reach a conclusive result, and describe this experiment in the Appendix C.

As a preliminary step, we conducted pilot experiments among a group of NLP graduate students. The main insights were: (a) humans are biased by quality: if a generated set has high diversity but low quality, humans will rate diversity low. To neutralize this, we explicitly ask workers to evaluate the quality of one of the responses in the set $\mathcal{S}_c$, and then instruct them to ignore quality in diversity questions; (b) To make sure a worker reads the context c, we ask them to generate a sentence s before they rate diversity; (c) It is difficult for workers to evaluate the diversity of a set with more than 10 responses. Our crowdsourcing tasks are provided in Appendix A.

5 Diversity to Similarity Reduction

We expand the idea from Zhu et al. (2018) and suggest a method to construct a diversity metric from any 2-sentence similarity metric. Given $m_{\text{sim}}(s_1, s_2) \in \mathbb{R}$, a symmetric similarity metric that gets a pair of input sentences (s_1, s_2) and returns a similarity score, we can define a diversity metric $\tilde{m}_{\text{div}}$ as the negation of the mean similarity score across all (unordered) pairs of $\mathcal{S}_c$:

$$\tilde{m}_{\text{div}}(\mathcal{S}_c) = -\frac{1}{\binom{|\mathcal{S}_c|}{2}} \sum_{s_i, s_j \in \mathcal{S}_c, i>j} m_{\text{sim}}(s_i, s_j).$$

This reduction allows us to easily define new diversity metrics based on past work on sentence similarity (Gomaa et al., 2013; Devlin et al., 2019; Zhang et al., 2019a; Reimers and Gurevych, 2019). In §6 we show that both n-gram-based similarity metrics and neural semantic similarity metrics provide useful diversity metrics.

6 Experiments

6.1 NLG Tasks

We apply our evaluation procedure on three different English NLG tasks that require diversity.

- **Story completion (*storyGen*);** We use the ROC Stories dataset (Mostafazadeh et al., 2016), in which the context c is the first four sentences of a story, and the response s is a single sentence that ends the story. We use the contexts $\mathcal{C}$ from this data and generate response sets $\mathcal{S}_c$ for each context using our testers. The long contexts characterizing this data narrow down the space of possible responses, making this a "low-entropy" generation task, where the output is constrained, but diversity is still essential.
- **Dialog response generation (*respGen*);** A comment-response pairs dataset extracted from the website `reddit.com` and pre-processed by Hashimoto et al. (2019). We use the comments from their data as contexts $\mathcal{C}$ and generate response sets $\mathcal{S}_c$ for each context using our testers. Since comments are single sentences the response is less constrained, making this a "medium-entropy" generation task.
- **3-words prompt completion (*promptGen*);** Contexts $\mathcal{C}$ are 3-words prompts, extracted from the Cornell Movie-Dialogs Corpus (Danescu-Niculescu-Mizil and Lee, 2011) by taking the first three words from each original context. The response sets $\mathcal{S}_c$ are completions of the prompts, generated by our testers. This context provides minimal constraints, making this a "high-entropy" generation task.

Samples of the contexts extracted for each task, along with generated response sets, are presented in Appendix B. We intentionally avoid NLG tasks where diversity is not necessarily desired, such as summarization and machine translation.

6.2 Evaluated Metrics

N-gram-based metrics We evaluate distinct n-grams (*distinct-n*), as described in §2. We also evaluate n-grams cosine similarity (*cos-sim*): a similarity measure computing the cosine between the vectors representing two sentences, where each vector is a count vector over the n-grams that appear in the response. We use the reduction from §5 to convert this to a diversity measure. In both metrics, rather than choosing the order of the n-grams, we average over $n \in \{1, \ldots, 5\}$, which we found to outperform any single choice of n.

Neural metrics We exploit existing BERT-based models (Devlin et al., 2019) fine-tuned for estimating similarity between two sentences (applying the reduction from §5).

BERT-STS; A BERT model fine-tuned on Semantic Textual Similarity (Cer et al., 2017): a collection of sentence pairs annotated with scores from 1-5 denoting their semantic similarity.[3]

BERT-Score (Zhang et al., 2019a); Originally a quality metric, *BERT-Score* uses BERT's embeddings to measure similarity between two sen-

---

[3] `https://github.com/swen128/bert-sts`

tences. We used `RoBERTa-large` (Liu et al., 2019), as suggested by the authors.[4]

Sentence-BERT (*sent-BERT*) (Reimers and Gurevych, 2019) is a sentence-level embedding model based on BERT. We use the cosine similarity between the embeddings of two responses as a similarity metric. In our experiments we used `bert-large-nli-stsb-mean-tokens`.[5]

Human Metrics We examine four methods for evaluating diversity with humans (see §4), to investigate best practices for obtaining diversity judgment from humans. In all metrics (except ranking), ratings are from 5 (highest diversity/similarity) to 1 (lowest). The original tasks presented to workers are in Appendix A.

Absolute HDS (*absHDS*); Given a context c and a set of generated responses $\mathcal{S}_c$, rate the level of diversity of $\mathcal{S}_c$.

Ranking HDS (*rnkHDS*); Given a context c and two sets $\mathcal{S}_{c,d_1}, \mathcal{S}_{c,d_2}$ generated with different values of the diversity parameter d, rate which set is more diverse. Since this metric did not clearly outperform *absHDS*, we provide results in Appendix C only.

Similarity HDS (*simHDS*); Given a context c and a set of generated responses $\mathcal{S}_c$, rate the similarity of each two sentences in $\mathcal{S}_c$, and then apply the reduction from §5.

Aspect HDS (*aspHDS*); Identical to *absHDS*, except we explicitly ask about a specific aspect of diversity, namely *form* and *content*.[6]

6.3 Decoding Test

In decTest we measure the correlation between diversity metrics (m_{div}) and the *softmax temperature* decoding parameter (d). The tester generating the response sets ($\mathcal{S}_c$) is a neural NLG model.

Context
Fire next door. John woke up smelling like something was burning. He went outside. He saw the fire next door. He called the authorities.
Response set ($\tau = 0.25$)
• It was a minor fire and they put it out. • It was a fire. • It was a fire. • It was a fire. • It was a fire.
Response set ($\tau = 0.8$)
• They arrived and put out the fire. • It was a fire. • It was a fire. • It turned out to be a fire. • It was a minor fire night.
Response set ($\tau = 1.1$)
• It turned out to be a mechanic. • Before the fire was put out it was a fire. • It was a fire. • They co-worker matter how bad the fire was. • Several shells, the fire department came just in time.

Table 1: An example of the effect of *temperature* on the response set $\mathcal{S}_c$ for a context c from ROC Stories.

Data and settings For each task, we generated sets of 10 responses per context, using a linear temperature sweep with 100 values in the range $[0.2, 1.2]$ (Caccia et al., 2018). We generated 1K sets in total for each of 1K contexts (10 per temperature) and evaluated 200 (2 random sets per temperature). For automatic metrics, we repeat this 100 times (randomly sampling 200 out of $1K$ sets each time), to present the mean and standard deviation. HDS metrics are computed over one experiment of 200 sets, due to their high cost.

Data for *storyGen* and *respGen* was generated by the MASS model (Song et al., 2019), fine-tuned on each dataset. Data for *promptGen* was generated by GPT-2-*large* (Radford et al., 2019) without fine-tuning. We provide examples for how story endings change as a function of temperature in Table 1. Examples for all tasks along with additional reproducibility details are in the Appendix B. For each HDS metric, we collected 10 ratings per query from Amazon Mechanical Turk (AMT) workers. While absHDS demands one query per response set, in order to perform simHDS at a reasonable cost, we chose $|\mathcal{S}_c| = 5$, resulting in $\binom{5}{2} = 10$ crowdsourcing queries instead of $\binom{10}{2} = 45$ per set. We evaluate simHDS only for *respGen* due to the metric's high cost and low performance.

Results Table 2 presents results of absHDS, simHDS, and all automatic metrics. In general, n-gram based metrics capture the diversity induced by a temperature sweep, beating HDS and neural metrics. Figure 3 provides a more detailed analysis. Each point represents a single set of responses generated at some temperature. While rank correlation for cosine similarity is high, it is

---

[4] `https://github.com/Tiiiger/bert_score`

[5] `https://github.com/UKPLab/sentence-transformers`

[6] We note that perplexity cannot be evaluated as a diversity metric in our framework, because it requires a sample from P_{ref}, while we assume a response set sampled from P_{gen}.

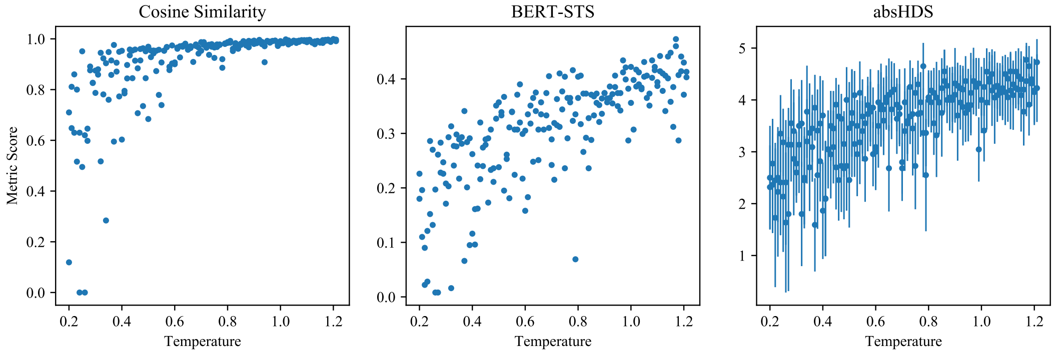


Figure 3: *decTest*: Scatter plot of n-gram-based (cosine similarity), neural (BERT-STS) and human (absHDS) metrics as a function of temperature for *respGen*. Each point corresponds to a single generated set. Error bars of HDS represent the standard deviation over 10 annotator ratings.

	storyGen	**respGen**	**promptGen**
distinct-n	**0.76** (0.03)	**0.89** (0.01)	**0.91** (0.01)
cos-sim	0.71 (0.04)	**0.89** (0.01)	0.87 (0.02)
BERT-STS	0.64 (0.04)	0.81 (0.02)	0.84 (0.02)
sent-BERT	0.65 (0.03)	0.80 (0.02)	0.74 (0.03)
BERT-score	0.69 (0.04)	0.87 (0.01)	0.88 (0.02)
absHDS	0.69	0.81	0.79
simHDS	-	0.74	-

Table 2: *decTest* results: Spearman's ρ correlation between temperature and each metric score (mean and standard deviation). *simHDS* was tested only on *respGen*.

	Temperature	**Top-p**	**Top-k**
distinct-n	**0.91** (0.01)	**0.84** (0.02)	**0.61** (0.05)
cos-sim	0.87 (0.02)	0.78 (0.03)	0.48 (0.05)
BERT-STS	0.84 (0.02)	0.74 (0.03)	0.55 (0.05)
sent-BERT	0.74 (0.03)	0.63 (0.05)	0.51 (0.05)
BERT-score	0.88 (0.02)	0.77 (0.03)	0.57 (0.05)

Table 3: *decTest* results for different decoding parameters: Spearman's ρ (mean and standard deviation) of automatic metrics for *promptGen*.

far from linear and reaches high values even at low temperatures, scoring 0.6 Pearson correlation. Conversely, the correlation for BERT-STS and absHDS is more linear, scoring 0.75 and 0.77 Pearson correlation respectively. Thus, Pearson and Spearman correlations disagree on the quality of the different metrics in this case.

While our framework is meant to evaluate diversity metrics, the results of the test let us reflect on the decoding parameters themselves. This result shows that humans perform worse than automatic metrics in this experimental setup, hinting that temperature mostly controls superficial changes to the generated text. Additionally, simHDS performs worse than absHDS although it is 3x more expensive, showing that rating the entire set rather than averaging over pairs is useful.

Other decoding parameters To compare the robustness of our conclusions to other decoding parameters, we repeat it with two additional decoding methods: (a) in *Nucleus (Top-p) sampling* we swept linearly over 100 values of p in the range $[0.1, 1.0]$; (b) In *Top-k* sampling we swept k in logarithmic scale over 100 values in the range $[1, 30K]$ and present the correlation between the metrics and $\log_{10}(k)$. While softmax temperature enables skewing P_{LM} to a more diverse P_{gen} using $\tau > 1$, both Top-p and Top-k enable only skewing P_{LM} to a more sharp (hence less diverse) P_{gen}.

Table 3 presents results for all automatic metrics using the three decoding methods over *promptGen*. Results for other tasks are in Appendix C. We find that Top-p correlates well with temperature along all three generation tasks, whereas Top-k does not correlate with any of them.

6.4 Content Test

In conTest, we measure the correlation between diversity metrics (m_{div}) and content diversity, represented by a binary parameter $d \in \{0, 1\}$. The testers are AMT workers, guided to create sets with high level of *form* diversity and high or low *content* diversity according to d.

Data and settings For each task, we collected 200 sets of 5 responses each (100 sets per class). For high content diversity class, we asked workers to give 5 responses per context, with as different content and structure as possible. Then we asked the same workers to choose a single response they wrote, and rephrase it 5 times such that the original content will be preserved, while changing the form – this set is used for the low content diversity class.

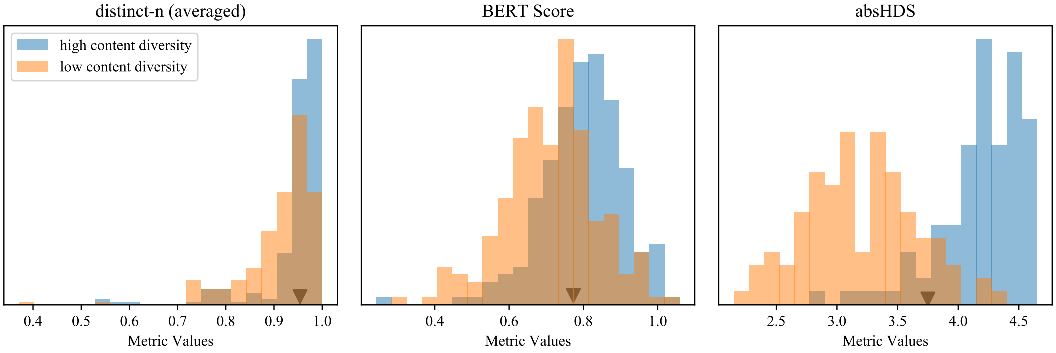


Figure 4: *conTest*: histograms of metric values of n-gram (distinct n-grams), neural (BERT-Score) and human (absHDS) metrics for *promptGen*. The **orange** histogram represents the distribution of the *low content diversity class*, the **blue** histogram represents the distribution of the *high content diversity class* and **brown** is the intersection between the two. Pointing down triangles represent the threshold η of the optimal classifiers. The histograms show how each metric separates the two classes.

A sample from this data is in Figure 1 and more samples in Appendix B. For each HDS metric, we collected 10 ratings from crowdsourcing workers, different than the ones who composed the sets.

Results In addition to Spearman's ρ, we report the optimal single-threshold classifier accuracy (OCA), i.e., the best achievable accuracy in predicting the class of a response set (high or low content diversity) for any threshold η on m_{div}, such that if $m_{\text{div}}(\mathcal{S}_c) > \eta$ the classifier predicts *high diversity*, and otherwise predicts *low diversity*.

Table 4 shows the results. N-gram-based metrics perform poorly, indicating they do not measure content diversity well. Neural models perform better than n-gram-based metrics (especially sent-BERT), but there is still a clear gap between automatic metrics and humans. Figure 4 illustrates the typical distributions of n-gram, neural and human metrics. Clearly, HDS separates high and low *content* diversity better than neural metrics. In addition, n-gram-based metrics saturate both classes to near maximal values, similarly to decTest.

Since conTest isolates content diversity, we used aspHDS to directly rate content and form diversity. *Content* aspHDS gets similar scores to *absHDS*, suggesting little gain in asking directly on the tested aspect. *Form* aspHDS gets low scores compared to absHDS, validating that the form diversity of the two classes is similar.

Content Diversity Benchmark We construct the **M**etrics for **c**ontent **Div**ersity (*McDiv*) benchmark, focusing on metrics for content diversity. McDiv is a dataset containing $6K$ $\{c, \mathcal{S}_c\}$ pairs, ($2K$ for each storyGen, respGen and promptGen) collected as described in this section. McDiv contains a subset of $3K$ examples, termed *McDiv$_{nuggets}$*, in which *form* diversity was neutralized, providing a difficult meta-evaluation challenge. McDiv$_{\text{nuggets}}$ was sampled to ensure that the correlation of *distinct-n* (a form diversity metric) is zero over this subset. Applying conTest over the data shows that n-gram based metrics obtain near-zero values on McDiv$_{\text{nuggets}}$ as expected, and all neural metrics perform substantially worse on McDiv$_{\text{nuggets}}$ than on McDiv. On conTest, we obtain absHDS annotations for more than 200 random samples from McDiv$_{\text{nuggets}}$ and obtain 0.7 Spearman's ρ for the respGen task, substantially higher than the best performing neural metric (sent-BERT) score at 0.6. Details and conTest results can be found in Appendix C.

	storyGen		**respGen**		**promptGen**	
	ρ	OCA	ρ	OCA	ρ	OCA
distinct-n	0.57	0.77	0.34	0.67	0.33	0.68
cos-sim	0.56	0.77	0.33	0.66	0.36	0.67
BERT-STS	0.6	0.78	0.46	0.72	0.65	0.82
sent-BERT	0.77	0.90	0.59	0.79	0.68	0.81
BERT-score	0.59	0.77	0.49	0.74	0.4	0.69
absHDS	**0.85**	**0.95**	0.63	0.81	**0.78**	**0.89**
aspHDS$_{\text{form}}$	0.35	0.65	0.56	0.79	0.4	0.68
aspHDS$_{\text{content}}$	0.84	0.94	**0.67**	**0.83**	0.75	0.88

Table 4: *conTest* results: Spearman's (ρ) correlation between a set's class and each metric score.

HDS Stability: Picking Parameter Values HDS experiments demand expensive human labor. Thus, we need to carefully choose the number of sets and different ratings we ask per set, to get reliable results in a reasonable budget. To this end, we conducted two series of experiments, once increasing the number of sets, and again increasing the number of ratings per sets. By observing results along those two series, we chose to use 200

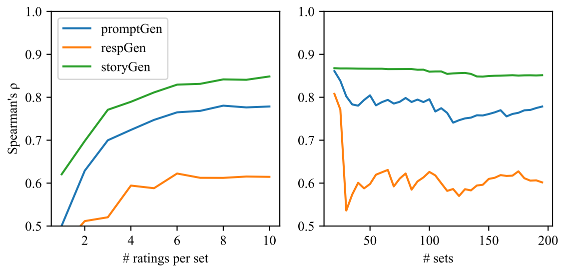


Figure 5: conTest *absHDS* results depends on the number of ratings per set and the number of sets.

sets and 10 ratings per set for all experiments - the minimal values in which results are confidently stable. Results are presented in Figure 5.

7 Aspects of Diversity

In this work, we focused on the two primary aspects of diversity: *content* diversity (What to say?) and *form* diversity (How to say it?). In Figure 1, Both sets are diverse, but *Set B* is only form diverse, as all answers deliver the same massage, whereas *Set A* is diverse in both form and content.

Furthermore, we can observe aspects of diversity as having a tree-like structure, where both content and form diversity can be divided to sub-aspects: Content diversity (e.g. answering the question *"How are you today?"*) can be expressed by using different *sentiment* (*"I'm doing good."* vs. *"I'm so glad you asked! I'm really doing good."*), different *relevance* (*"I'm fine"* vs. *"Did you watch the game last night?"*), and more. Form diversity can be divided into sub-aspects as well: *syntactic* diversity (*"Someone took it from me."* vs. *"It was taken from me."*) or *lexical* diversity (*"I feel fine."* vs. *"I feel very well."*). Even those sub-aspects can be further divided. For example, a sub-aspect of lexical diversity is *register* diversity (*"How are you?"* vs. *"Sup bro?"*).

Another observation is that different aspects are not orthogonal, that is, changing one aspect may lead to changes in other aspects. Specifically, we observe that while it is relatively easy to produce high form diversity with low content diversity (*Set B* in Figure 1), it is almost impossible to diversify content without changing form. This observation was important during the design of conTest.

8 Conclusions

This work presents a framework for evaluating diversity metrics as a step toward standardized evaluation. We limit the scope of this work to differences between *form* and *content* diversity, which are key towards understanding different aspects of diversity. Future work can explore other aspects of diversity, e.g. testing *sentiment* diversity, as proposed in §3. We urge researchers to use this framework as a platform for developing new diversity metrics and establishing their efficiency.

Acknowledgements

We thank Aya Meltzer-Asscher for linguistic advice, and Or Nachmias, Ben Bogin, Mor Geva, Omer Goldman and Ohad Rubin for their useful suggestions and references. This research was partially supported by The Israel Science Foundation grant 942/16, The Yandex Initiative for Machine Learning and the European Research Council (ERC) under the European Union Horizons 2020 research and innovation programme (grant ERC DELPHI 802800).

References

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. *arXiv preprint arXiv:1811.02549*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*.

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

Wenchao Du and Alan W Black. 2019. Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 38–43.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The e2e nlg challenge. *Computer Speech & Language*, 59:123–156.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Asma Ghandeharioun, Judy Hanwen Shen, Natasha Jaques, Craig Ferguson, Noah Jones, Agata Lapedriza, and Rosalind Picard. 2019. Approximating interactive human evaluation with self-play for open-domain dialog systems. In *Advances in Neural Information Processing Systems*, pages 13658–13669.

Wael H Gomaa, Aly A Fahmy, et al. 2013. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18.

Tatsunori Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1689–1701.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Yi-An Lai, Xuan Zhu, Yi Zhang, and Mona Diab. 2020. Diversity, density, and homogeneity: Quantitative characteristic metrics for text collections. *arXiv preprint arXiv:2003.08529*.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016b. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016c. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Junyi Li, Wayne Xin Zhao, Ji-Rong Wen, and Yang Song. 2019. Generating long and informative reviews with aspect-aware coarse-to-fine decoding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1969–1979.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Generating reasonable and diversified story ending using sequence to sequence model with adversarial training. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1033–1043.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher D Manning, Christopher D Manning, and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT press.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949*.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983.

Manasvi Sagarkar, John Wieting, Lifu Tu, and Kevin Gimpel. 2018. Quality signals in generated stories. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 192–202.

Raphael Shu, Hideki Nakayama, and Kyunghyun Cho. 2019. Generating diverse translations with sentence codes. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Guy Tevet, Gavriel Habib, Vered Shwartz, and Jonathan Berant. 2019. Evaluating text gans as language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2241–2247.

Qingyun Wang, Lifu Huang, Zhiying Jiang, Kevin Knight, Heng Ji, Mohit Bansal, and Yi Luan. 2019. Paperrobot: Incremental draft generation of scientific ideas. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1980–1991.

Pengcheng Yang, Lei Li, Fuli Luo, Tianyu Liu, and Xu Sun. 2019. Enhancing topic-to-essay generation with external commonsense knowledge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2002–2012.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

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

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. 2019b. Syntax-infused variational autoencoder for text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2069–2078.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

A HDS Questionnaires

All Human scores for HDS metrics were collected using Amazon Mechanical Turk (AMT) crowdsourcing platform by English native-speaking workers that were specifically qualified for this task. Figure 7 presents the warm-up part, common for all HDS questionnaires. Before asking workers to rate the diversity of each set, we first asked them to generate a response for the context themselves, to make sure they read it. To neutralize the effect of the responses' quality on the workers, we also asked the workers to rate the quality of the first response in the set, then explicitly instructed them to ignore quality when rating diversity.

Figures 8 to 11 present the diversity questions of absHDS, aspHDS, rnkHDS and simHDS as appeared in the AMT questionnaires.

Costs For HDS metrics that require one query per response set (i.e. absHDS, rnkHDS, aspDHS), the cost for a single rating was 0.18\$. We collected 10 ratings per response set, and conduct each experiment with 200 sets, hence the total cost for an experiment was 360\$. In the case of simHDS, the response set size was 5, and the number of queries needed per set is $\binom{5}{2} = 10$. The cost of a single rating for this task was 0.056\$, and with the same multipliers, the total cost for an experiment was 1120\$, three times more expensive.

B Data Samples

B.1 Decoding Test (decTest)

Tables 11 to 19 present data samples from storyGen, respGen and promptGen with the neural testers of decTest, as detailed in §6. Each table presents two contexts and three response sets per context. Each response set was generated with a different value of decoding parameter for the three decoding methods: softmax temperature, Nucleus sampling, and Top-k.

B.2 Content Test (conTest)

Tables 20 to 22 present data samples from storyGen, respGen and promptGen with the human testers of conTest, as detailed in §6. Each table presents two contexts and two response sets per context - one for the *low* content diversity class and one for the *high* content diversity class.

C Additional Experiments

C.1 Decoding Test (decTest)

Comparing decTest results of storyGen to other tasks (Table 2), this task is characterised with noisier scores for all metrics (Figures 3 and 6), hence lower ρ values and higher variance. A possible explanation is larger effect of c on the distribution $P_{gen}(s|c)$ in this task.

Tables 3, 6 and 7, present decTest absolute scoring experiment using *temperature*, *nucleus sampling* and *Top-k* decoding parameters as d. Top-k consistently yields lower ρ compared to other decoding parameters, especially for storyGen task. This implies that Top-k represents diversity less reliably than other methods.

Ranking experiment To examine whether we can improve correlation by asking humans to *rank* diversity, rather than providing an absolute score, we designed a ranking version of decTest. Each context is given along with two sets (5 samples each), produced with different temperature values. We sweep over temperature differences instead of the absolute temperature values. The human metric in this setting is *rnkHDS* (see §6.2), and the automatic metrics are the difference between the scores each of the two sets got.

We report two measures; The first is Spearman's ρ between the metric and the temperature difference. The second is accuracy, i.e., whether the metric can predict which set has higher temperature (e.g., in automatic metrics this is whether the sign of the temperature difference and the sign of metric score difference agree).[7]

Table 5 summarizes the ranking test results. We observe that humans are better at ranking compared to giving absolute scores (Table 2), and are doing as well as automatic metrics. However, the scores of all automatic metrics also improve, making it difficult to separate between the different metrics.

C.2 Metrics for Content Diversity (McDiv)

As elaborated in § 6.4, McDiv is a dataset containing $6K$ $\{c, \mathcal{S}_c\}$ pairs, ($2K$ for each storyGen, respGen and promptGen) collected as described in §6.4. McDiv$_{\text{nuggets}}$ is a $3K$ subset of McDiv, in which *form* diversity is neutralized, providing a difficult meta-evaluation challenge. McDiv$_{\text{nuggets}}$ was sampled in a manner that causing *distinct-n*

---

[7] We consider ties in the metric difference score as a miss.

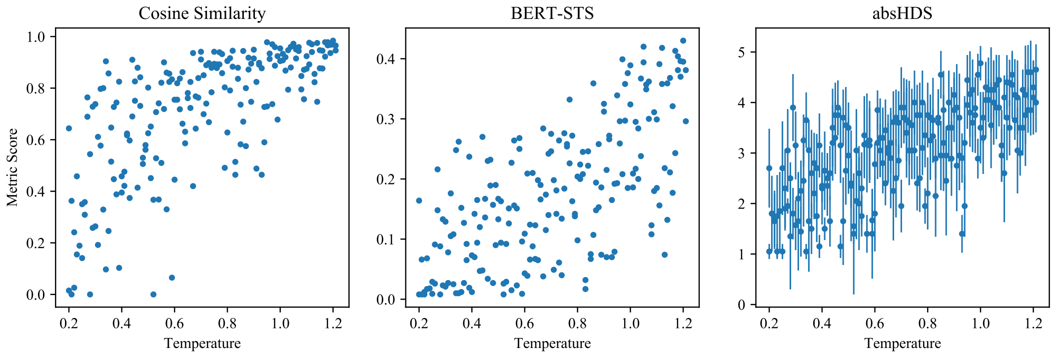


Figure 6: decTest: Scatter plot of n-gram-based (cosine similarity), neural (BERT-STS) and human (absHDS) metrics as a function of temperature for *storyGen*. Each point corresponds to a single generated set. Error bars of HDS represent the standard deviation over 10 annotator ratings.

	storyGen		**respGen**		**promptGen**	
	ρ	acc	ρ	acc	ρ	acc
distinct-n	**0.88**	0.88	0.86	0.9	**0.91**	**0.91**
cos-sim	0.86	0.88	0.87	**0.91**	0.9	**0.91**
BERT-STS	0.84	0.84	0.85	0.88	0.9	0.89
sent-BERT	0.85	0.86	0.83	0.85	0.85	0.85
BERT-score	**0.88**	**0.89**	0.88	0.89	**0.91**	0.9
rnkHDS	0.87	**0.89**	**0.89**	0.9	0.89	0.88

Table 5: *decTest* ranking results: Spearman's (ρ) correlation between temperature differences and each metric score. Accuracy (acc) of classifying which set has the higher temperature. Standard deviation is up to 0.02 for all automatic metrics for both Spearman's correlation and accuracy.

metric to score zero correlation in conTest over this subset. The method of sub-sampling was meant to approximately equalize the distributions of the two classes, *low* and *high* content diversity, over the scores of distinct-n metric, and was performed as follows:

- Sort all collected samples (from both *low* and *high* content diversity classes) according to their *distinct-n* score.
- Divide the sorted samples to groups with fixed size (40 samples each in our case).
- From each such group, randomly sample the same amount of samples for each of the two classes. For example, if a group contains 5 *low* content diversity samples and 35 *high* content diversity samples, we can sample at most 5 samples for each class.

Resutls We applied conTest for all the collected data for each of the three NLG tasks (see Tables 8 and 9). By design, n-gram based metrics score near-zero correlation on McDiv$_{\text{nuggets}}$, making *high* and *low* content diversity classes almost indistinguishable for those metrics, which relay on text surface level features only. Neural metrics perform strictly worse on McDiv$_{\text{nuggets}}$ than McDiv. In addition, we applied conTest on 200 randomly sampled $\{c, \mathcal{S}_c\}$ pairs from McDiv$_{\text{nuggets}}$ for respGen task (see table 10). Compared to Table 4, The gap between the best performing neural metrics (sent-BERT) and absHDS was increased in favor to HDS (0.04 compared to 0.1 difference in Spearman's ρ).

D Additional Reproducibility Details

Collected data and code All the collected data, metric scores per samples for each of decTest and conTest, as well as code for running and visualizing the tests, are publicly available[8]. The collection methods are elaborated in Section 6.

Original data We provide additional data for the original three datasets used in Section 6.

- ROC Stories dataset[9] (Mostafazadeh et al., 2016) used for storyGen task contains $96K/1K/1K$ train/validation/test titles and five-sentence stories. We used the samples without pre-processing for both fine-tuning MASS model and generate samples for our tests.
- Reddit comment-response dataset used for respGen task contains $37M/1M/1M$ train/validation/test comment - response pairs, extracted from the social website `reddit.com` scraped by `pushshift.io` followed by the pre-process described in

---

[8] `https://github.com/GuyTevet/diversity-eval`

[9] `www.cs.rochester.edu/nlp/rocstories/`

(Hashimoto et al., 2019). We used the samples without further processing for both fine-tuning MASS model and generate samples for our tests. To the best of our knowledge, this dataset is not publicly available at the moment.

- CMDC dataset[10] (Danescu-Niculescu-Mizil and Lee, 2011) contains $108K/30K$ train/test sentence-response pairs extracted from movie scripts. We extracted the first three words from the sentences (used as contexts for the original task) to be the context of our task. We did not use this data for training since we used GPT-2 without fine-tuning for promptGen.

Auto-generated data For decTest, we used two pre-trained generative models for generating responses given the contexts:

- For storyGen and respGen tasks, we used MASS[11] (Song et al., 2019) (6L-1024H-8A architecture suggested by the authors), pretraind as described in the original paper. For each task separately, we fine-tuned MASS using the training division of the dataset corresponding to the task. Fine-tuning was done using $200K$ examples over 30 epochs, and took 23 hours using a single *TITAN Xp* GPU core. Inference with the fine-tuned model takes 65 milliseconds on average per response set containing 10 responses with the same GPU core.
- For promptGen task, we used Hugging-Face implementation[12] of GPT-2 *large* (36-layer, 1280-hidden, 20-heads, 774M parameters) (Radford et al., 2019) pre-traind as described in the original paper. We used this model as-is, without fine-tuning. Inference takes 0.6 second on average per response set containing 10 responses with a single *TITAN Xp* GPU core.

Tests Runtime Given metric scores per sample, running each of the tests with 200 samples takes less than a minute on a standard *Intel i7* CPU.

	Temperature	**Top-p**	**Top-k**
distinct-n	**0.76** (0.03)	**0.69** (0.03)	0.2 (0.06)
cos-sim	0.71 (0.04)	0.66 (0.03)	0.16 (0.06)
BERT-STS	0.64 (0.04)	0.58 (0.04)	0.2 (0.07)
sent-BERT	0.65 (0.03)	0.59 (0.04)	0.17 (0.06)
BERT-score	0.69 (0.04)	0.61 (0.04)	**0.23** (0.05)

Table 6: decTest results for different decoding parameters: Spearman's ρ (mean and standard deviation) of automatic metrics for *storyGen*.

	Temperature	**Top-p**	**Top-k**
distinct-n	**0.89** (0.01)	**0.84** (0.02)	**0.64** (0.04)
cos-sim	**0.89** (0.01)	0.78 (0.03)	0.62 (0.05)
BERT-STS	0.81 (0.02)	0.74 (0.03)	0.56 (0.04)
sent-BERT	0.80 (0.02)	0.63 (0.05)	0.51 (0.04)
BERT-score	0.87 (0.01)	0.77 (0.03)	0.6 (0.05)

Table 7: decTest results for different decoding parameters: Spearman's ρ (mean and standard deviation) of automatic metrics for *respGen*.

	storyGen		**respGen**		**promptGen**	
	ρ	OCA	ρ	OCA	ρ	OCA
distinct-n	0.53	0.74	0.52	0.74	0.48	0.75
cos-sim	0.53	0.74	0.52	0.74	0.60	0.77
BERT-STS	0.57	0.74	0.61	0.78	0.78	0.89
sent-BERT	**0.75**	**0.87**	**0.68**	**0.83**	**0.8**	**0.9**
BERT-score	0.60	0.77	0.56	0.78	0.54	0.74

Table 8: conTest results for McDiv; Results for automatic metrics over all the samples (2K per task).

	storyGen		**respGen**		**promptGen**	
	ρ	OCA	ρ	OCA	ρ	OCA
distinct-n	-0.002	0.49	-0.002	0.49	-0.003	0.49
cos-sim	0.04	0.53	0.08	0.55	0.22	0.60
BERT-STS	0.34	0.64	0.39	0.68	0.68	0.83
sent-BERT	**0.63**	**0.80**	**0.53**	**0.76**	**0.73**	**0.85**
BERT-score	0.35	0.66	0.33	0.65	0.35	0.65

Table 9: conTest results for McDiv$_{\text{nuggets}}$ subset; Results for automatic metrics over all the samples (1K per task).

	storyGen		**respGen**		**promptGen**	
	ρ	OCA	ρ	OCA	ρ	OCA
distinct-n	0.04	0.57	-0.01	0.46	0.12	0.56
cos-sim	0.05	0.54	0.04	0.54	0.28	0.62
BERT-STS	0.34	0.68	0.39	0.69	0.72	0.9
sent-BERT	0.68	0.85	0.6	0.79	0.75	0.88
BERT-score	0.37	0.69	0.34	0.68	0.38	0.69
absHDS	**0.78**	**0.9**	**0.7**	**0.85**	**0.84**	**0.94**

Table 10: conTest results for 200 random samples from McDiv$_{\text{nuggets}}$ including HDS.

---

[10] `www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html`

[11] `github.com/microsoft/MASS`

[12] `github.com/huggingface/transformers`

Context	**Response set** ($\tau = 0.25$)	**Response set** ($\tau = 0.8$)	**Response set** ($\tau = 1.1$)
Fire next door. John woke up smelling like something was burning. He went outside. He saw the fire next door. He called the authorities.	• It was a minor fire and they put it out. • It was a fire. • It was a fire. • It was a fire. • It was a fire. • It was a fire. • It was a minor fire. • It was a fire. • It was a fire. • It was a minor fire and they put it out.	• They arrived and put out the fire. • It was a fire. • It was a fire. • It turned out to be a fire. • It was a minor fire night. • They arrived and put it out. • It was a scary but beautiful fire next time he went in. • It was a fire. • It was a put out and John was able to put it out. • It was a scary place to live.	• It turned out to be a mechanic. • Before the fire was put out it was a fire. • It was a fire. • They co-worker matter how bad the fire was. • Several shells, the fire department came just in time. • They arrived and put out the fire. • It turned out to be a clear, fact. • It was a fire. • It was a scary meal for wet clothes. • It was a scary stealing from him.
Charles Shaves His Chest. Charles decides he no longer wants chest hair. He shaves his chest hair with a razor. Although his skin is irritated, he hair is now gone. Charles is satisfied.	• He is glad that he finally shaved. • He is glad that he finally shaved. • He is very happy to have a chest hair. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that he finally shaved.	• He is very happy to have a chest hair. • He is glad that he finally shaved. • He is very happy that he no longer has chest hair. • He is glad that he finally shaved. • He is glad that he finally shaved. • He is glad that his chest hair is no longer shaved. • He is glad that he finally shaved • He is glad that he finally shaved. • He is glad that he finally of shaving. • He is glad that he finally shashock his chest hair.	• He is glad that he finally peaceful his voice again. • He is glad that he finally shaavoided his chest hair. • He is glad that he finally downcome. • He is ready to start better cooperation his fireplace. • He is glad that he finally shaved. • He is glad that he no longer has chest hair. • He is happy to finally be shaved. • He is very happy to have ster good ride of his life. • He is glad that he finally shaved. • He is glad that he finally has chest hair.

Table 11: decTest data samples for storyGen task and different temperatures.

Context	**Response set** ($p = 0.208$)	**Response set** ($p = 0.64$)	**Response set** ($p = 1$)
Really Bad Decisions. Jake was nervous about a meeting at work the next day. He decided to have a drink to relax. Unfortunately Jake kept drinking. He was really hung over the next day.	• He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting.	• He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting.	• He did not get to the meeting anymore. • He missed his meeting. • He passed out and failing the meeting • He missed his meeting. • He missed his meeting. • He missed his meeting. • He missed his meeting. • He passed out and was kicked out of the meeting. • He missed his meeting. • He missed his meeting.
Family Night Food. Tonight, my mom ordered Mexican food for family night. She got it from my favorite Mexican place in town. When it arrived, it was hot and smelled wonderful. We devoured it with gusto.	• After a few hours of take it home we all enjoyed its night. • After a few hours of take it home we all enjoyed its night. • After a few hours of take it home we all enjoyed its night. • After a few hours of eating everyone was satisfied. • After a few hours of take it home we all enjoyed its night. • After a few hours of eating everyone was satisfied. • After a few hours of take it home we all enjoyed its night. • After a few hours of take it home we all enjoyed its night. • After a few hours of take it home we all enjoyed its night. • After a few hours of eating everyone was satisfied.	• After dinner, we all went home to cook Mexican food. • After a few hours of cooking she was tired and ready to eat. • After dinner, I always put got ready for Christmas. • After dinner, I helped her do the dishes. • After a few hours of dinner, the food was amazing. • After a few hours of take it home we all enjoyed margaria. • After dinner, I was ready to take on work the next day of • After dinner, I was sad to say goodbye to her • After a few hours of take it home we all enjoyed one bite. • After a few hours of eating everyone was satisfied.	• Even though my stomach was gone, I was sad it was finally pockets • After dinner, I alone. • All in all the family while my mom finished the food. • After a few hours of dancing, she pianed. • Afterwards I'd never finish single night a week. • Afterwards we all went to sleep in my woods. • Afterwards I helped her do much better than my wife. • Afterwards my mom helped me finish my dinner. • After a few hours we all enjoyed storm blood from the Italy. • After dinner, we all enjoyed some good food together.

Table 12: decTest data samples for storyGen task and different p values (nucleus sampling).

Context	**Response set** ($k = 3$)	**Response set** ($k = 32$)	**Response set** ($k = 318$)
Loud Noise. Jane was trying to take a nap. She heard a loud bang in the kitchen. It woke her up. A dish had fallen off the counter.	• Jane pulled over and started to clean herself. • Jane was horrified and dropped her favorite food. • Jane was able to finish her car and take a nap. • Jane was able to finish her nap . • Jane was able to finish her nap . • Jane stopped at the store to buy a new one. • Jane was able to grab a nap afterwards. • Jane was unable to finish her nap since it was lost. • Jane pulled over and started to clean up. • Jane was able to finish her nap .	• Jane was able to catch a car using the seat. • Jane stopped at the store to buy a new book. • Jane was sad her cat dropped out of the kitchen. • Jane screamed. • Jane was horrified to find her car broken down on the floor. • Jane was horrified and dropped her pay phone. • Jane was easily able to grab a nap. • Jane pulled over and started to cry. • Jane pulled over and started to cry. • Jane stopped at the store to buy a new dish from the store.	• Jane comes, noticed a lot of food left under it. • Jane was horrified and dropped her book. • Jane remembered to take a nap. • Jane was since she took a nap while she waited for the refun • Jane knew she had no time to finish her book. • Jane was glad. • Jane was annoyed and began to cry. • Jane stopped at the store to buy a new one. • Jane wanted to have her car back. • Jane was monthed.
Headache. Kate was wearing big over the ear headphones. But they were tight and squeezing her head. She tried to adjust them to relieve the tension. But nothing really worked.	• Kate decided to go to the store and buy some ear phones. • She decided to go to the store and buy some headphones instead. • She decided to go to the store and buy some ear phones. • She decided to go buy a pair of headphones instead. • She decided to go to the store and buy some headphones instead. • She decided to go to the store and buy some headphones instead. • She decided to go to the doctor and have some rest. • Kate decided to go to the store and buy some headphones instead. • She decided to go to the store and buy some ear phones. • She decided to go buy some ear plugs.	• She decided to go buy a hat so she could enjoy the the long lin • Kate decided to go to the store to buy some candy since she was ti • Kate decided to go outside to rest. • Kate decided to go to the store and buy some ran phones. • Kate decided to go to the store and buy some headphones instead. • Kate decided to go back to her old dishes. • She decided to go buy a big pair of headphones instead. • Kate decided to go to the store and wearing some headphones. • She was forced to go to the store to buy some cash. • She decided to go to the store and buy some headphones instead.	• She decided to go to a ess instead. • Kate decided to go to the store and buy some fake headphones. • She decided to go to the store and buy some ear phones. • She decided to go to the store and buy some headphones instead. • She decided to go to the store and buy some headphones instead. • She decided to go buy some ear phones. • Kate decided to go to the gym and use some saw no more watching T • Kate decided to go to the store and buy some cute phones. • She decided to go buy a fake headphones instead. • She decided to go to the store and buy some concert.

Table 13: decTest data samples for storyGen task and different k values (Top-k).

Context	**Response set** ($\tau = 0.25$)	**Response set** ($\tau = 0.8$)	**Response set** ($\tau = 1.1$)
it is doing okay i just want people to talk more	• i think you mean people • i think its working now • i just want people to talk more • i think i just want people to talk more • i dont think you understand what youre saying • i like to think so • i know that feel • i dont know what to say • i think its pretty cool • i think its working now	• i cant believe you didnt think this through i actually havent had • may i ask why • i just hope its not going to happen • its alright dont even want to talk • what did it say i was using windows 95 anyway xd • thanks for reading the thread • i understand and hope they dont mind that • want to talk more • its doing ok body language great voice acting and music • all good i know your pain	• im just hawaii everyone wants to be friends lt go talk to people from • its pretty smart we didnt run a year ago and have turned back but • id lick your teeth but tha t sucks smart people • it really didnt stop i think you need to go back to normal now • sounds like a plan to me • we we can try forever practice beer • okay but another two more 200 radio views im going to ore my comment of • team building looking bro • yep they said that things have changed • yeah we thank you random stranger
girls who say no not like it matters	• i dont like it at all • i dont like it either • i dont like it either • i think you mean girls who say no not like it matters • i think that is the most accurate description of this thread • i like this one • girls who say no dont like it matter • i dont like it either • i know a guy who says no and he doesnt like it • girls who say no dont like it matter	• wait so there are girls who say no like it matters • i have a friend who doesnt like it but i always say no • and you dont like to play with girls • i say no i dont want it to • girls who say no dont like it care • i see what you did there • girls who say no need to make eye contact with girls • girls who say yes dont like it does • i really dont care if its not gay or straight out programs or not • ugh that game was just awful troll	• how about mine like sales figure instead of higher • definitely not as stupid as that • agreed is that true • refer to my highest iq you and me • less keep it up • dude ive never seen such an complaining member with a legitimate thought • not that theres anything wrong with that or are more wise than telling want • girls make videos youve never met • gtthe girls who say no or a one time thing people thats • which is weird since it is a girl i know so sad

Table 14: decTest data samples for respGen task and different temperatures.

Context	**Response set** ($p = 0.208$)	**Response set** ($p = 0.64$)	**Response set** ($p = 1$)
do you know if he still lives in new orleans	• i dont think so • i dont think so • i dont think so • no i dont think so • no idea i dont know • i dont think so • no i dont think so • no idea i just saw him live in new orleans • i dont think so • no idea i just saw him live in new orleans	• he is in nyc • he does i hope he does • i dont think so • he lives in new orleans • yes he does he is in florida • im guessing not that i know of • hes a small town i dont know what happened there • nope still in new orleans • i do not i dont know if he still lives in new orleans • i dont think so	• aw screw you guys the gimme a shower and ill send you my money • he does im so confused by how this is relevant • they spent all that much in new orleans up in first place • still living in new orleans according to most of them it jesus mustve worth • anyone else want to wait for him to go back home • how many tickets would you say the willing said if he warn you that hes • im guessing he lives there as well • yep rick albeit not • he lives in ny • no i dont get it any more guess i may canadian
its that hanging thing in the back of your mouth	• its a tattoo of a hand • its a good thing its not • i think its a banana • i was thinking the same thing • its a very nice thing to do • its a hole in the bottom of my mouth • i think its a good thing • its a pretty cool thing to do • thats what i thought • i think its a butt	• well it is kind of like a mac • that would be a good idea • its my cake day • you can see it in his mouth • its the junk that was pooping in • i was thinking the same thing • i always see that • i didnt know that was possible • this is my favorite comment of the day • thats not a sticker	• i think you just flash • it is and feels good • my ex and i reach our arms cheating into the door for almost a minute • my mind went to side with that • yep terrible for my teeth • exactly what i was thinking • why do i look so contest • and where else are you going • just dont wear a helmet • and it looks like something some one would give to get in on

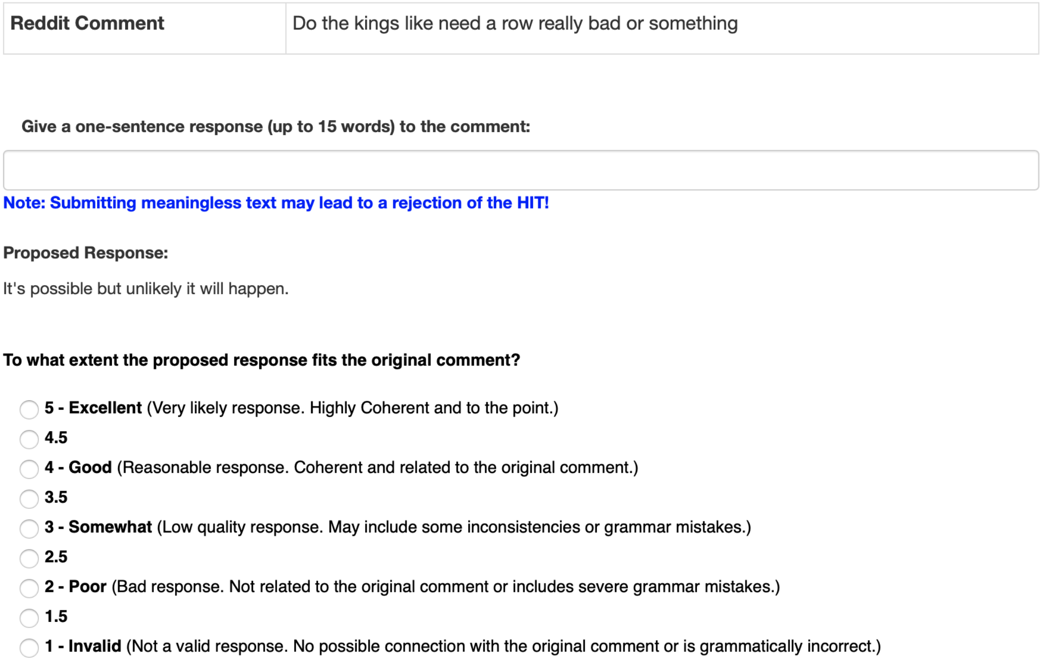
Table 15: decTest data samples for respGen task and different p values (nucleus sampling).

Context	**Response set** ($k = 3$)	**Response set** ($k = 32$)	**Response set** ($k = 318$)
watching curry play in his prime is truly a privilege	• i know i just dont want him to play for us • he has to be a good center for that • he is a great center of football in his prime • hes been playing in his prime for a long time • he was a great back in the day • hes been playing for a while now • i dont know about that he was pretty damn good at that • i dont think he was ever in his prime • i dont think he is a prime minister • i dont know why but i think he is a very good player and	• and his career as well • agreed the way hes playing is awesome • it has to be • this is just called a job • and then being on the field for the first time • i dont see him doing that often enough • he just likes to party in the kitchen • at this point hes going to be a great star for the rest of the • only if he pays well • the only thing that can make that kind of difference is how much time you	• yeah my feeling i mean we dont like it but it happens all the • you are one for real • they still have a rule saying they might not be injured yet • it really is a necessary thing to do • finally some reason to continue watching him at some point • yet that would be epic • not to mention eating curry dinner is a privilege • i just dont want to turn over for this goal like he does in • gt playing in his prime is truly a privilege ftfy • so is saying he is in high school
im going this evening when she usually works	• i think you accidentally a word • you are a good man • i hope she works • im going to the same time as you when she usually works • i am so sorry to hear that • i hope she works for you • i am so jealous of your work i am so jealous • i hope you have fun • i hope you get a job at a local bar • i hope she works for you	• i see what you did there • where in the world are you going to put your socks on • she usually does but she has to keep up to date with her in the • awesome what do you want for it • how does a women have a relationship with someone • did you get the job • thats where she goes • i know im also going this friday • thats a great idea • well make sure youre there too good luck	• thats what you like to hear • go see her later this evening • i read that as she usually fails • okay ill see you there brah • i always thought that it was for the sake of having the girl play as • that can be expected here • thats it im down now • i dont see why it would matter if shes married • i will be the ex gf • she still does make her phone calls the whole office

Table 16: decTest data samples for respGen task and different k values (Top-k).

Response set ($\tau = 0.25$)	**Response set** ($\tau = 0.8$)	**Response set** ($\tau = 1.1$)
• **Not the hacking**. The hacking is the fact that the DNC was hacked. ! • **Not the hacking**. The hacking is the real problem. The hacking is the! • **Not the hacking** of the DNC, but the leaks of the emails of the Democratic National Committee.! • **Not the hacking**, but the way it was done. The FBI's investigation into the! • **Not the hacking** of the DNC, but the hacking of the emails of the Democratic National Committee.! • **Not the hacking** of the DNC, but the leaking of the emails. The DNC's! • **Not the hacking** of the DNC. The hacking of the DNC was a ”false flag! • **Not the hacking** of the DNC. But the hacking of the RNC. The DNC hack! • **Not the hacking**. The hacking is the problem. The hacking is the problem! • **Not the hacking** of the DNC, but the leaking of the emails. The DNC was hacked,!	• **Not the hacking** after all? I'm sure the nation-states that are involved in! • **Not the hacking** that happened on the internal networks of the Energy Department. In fact, according to! • **Not the hacking** of the American public but rather the fraudulent Heisenberg principle that seemed to be! • **Not the hacking** that took place in the DNC last year or the release of hacked emails during the! • **Not the hacking** futurists Cardboard inventor and self-described tinkerer Dennis! • **Not the hacking** alone. In the first half of the report, the hackers tried to create fake! • **Not the hacking**. The hacking is the NSA's new SHIELD technology. It is! • **Not the hacking** and hacking and hacking of the world government. I know this man is a man! • **Not the hacking** aspect, but the pressure exerted by the Trumpistas. But also the Russia angle! • **Not the hacking**, but the willingness.” The evidence of interest in this case comes in!	• **Not the hacking** experience of a CIA VRO crunch nine months ago—JumpStart for 2016 jumps! • **Not the hacking**, David.) The directory was flagged in a document it created in late last year! • **Not the hacking** of Democratic Party systems - said the Russian team's activity represented ”just the beginning! • **Not the hacking**, of course – which these sources sounded more concerned about than being attacked 140 times! • **Not the hacking** story is over. But yet there's another reason not to rush out such statements! • **Not the hacking**-either.- These were scattered in the workshop.(Expanded- being guys with! • **Not the hacking** of private material of elected officials, e.g. emails, even if the! • **Not the hacking** has happened yet!!!!!!!!!!!!!! • **Not the hacking** rumours have cost him any of his followers, least of all the proprietors of! • **Not the hacking** group behind the breach of Sony, which has posted the staffer's information online,!
• **How is our** new technology helping us to do that? We are using a new technology! • **How is our** system different from that of the United States? The United States is a! • **How is our** approach different from that of the other major European countries? The European Commission! • **How is our** country going to be able to compete with the rest of the world if we don! • **How is our** country going to be able to compete with China in the future?” he asked.! • **How is our** work different from that of other organizations? The work of the Center for! • **How is our** work different from other research in this area? We are not the first! • **How is our** system of government supposed to work? The reason we have a government is! • **How is our** system different from the one that was used in the past? The system! • **How is our** country supposed to be a beacon of hope for the world if we have to look!	• **How is our** government going to catch up with the cyber criminals?” he said. ”I'm! • **How is our** society selling humanity on slavery? The answers to these questions are also important for us! • **How is our** minister giving it to you? Isn't it? It's got a bit of! • **How is our** research different from other studies? This study examined the effects of peer-! • **How is our** mission different from Seniors' Service Corps (SSC) other than the fact! • **How is our** challenge different? The only difference is that this challenge is about building an! • **How is our** nation governed?” As Obama moved into his second term, he is increasingly! • **How is our** recommendation different from what more traditional veterinarians do? We don't believe! • **How is our** rapid abandonment of critical thinking, knowledge, and values, and the subsequent burial of! • **How is our** education system designed for our futures? We are the children of immigrants,!	• **How is our** Internet even even connected with our corporate tracks? Every cell phone on the planet knows! • **How is our** developer name attached to the icon? Since the Planetside icon is use internally! • **How is our** food paradise created? Artificial chemical fertilizers. So these aren't GMOs, but! • **How is our** acquisition* worth - BOARD ROLL (Least Significant Equivalents)! • **How is our** transit plan addressing this problem? Under our old plans, Burlington Buses! • **How is our** mind different than any other part of the body?” A Broader View! • **How is our** campaign working? Bitcoin launches alongside psychological research showing that people pay a lot! • **How is our** mentioning application related to a related method (#five with two in queue) page such! • **How is our** having to resort to roundabout hypotheticals to argue that Stewart may secretly want! • **How is our** blood working out for you?” a statewide voter got an outpouring of rename and!

Table 17: decTest data samples for promptGen task and different temperatures. Bold text is the 3-words prompt context.



Reddit Comment	Do the kings like need a row really bad or something

Give a one-sentence response (up to 15 words) to the comment:

Note: Submitting meaningless text may lead to a rejection of the HIT!

Proposed Response:

It's possible but unlikely it will happen.

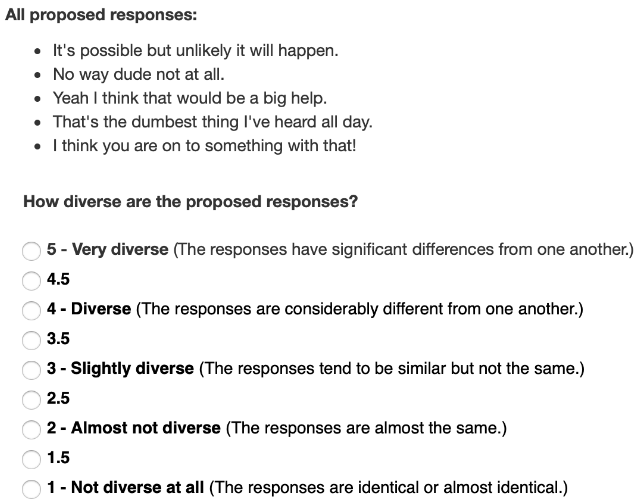
To what extent the proposed response fits the original comment?

○ **5 - Excellent** (Very likely response. Highly Coherent and to the point.)
○ **4.5**
○ **4 - Good** (Reasonable response. Coherent and related to the original comment.)
○ **3.5**
○ **3 - Somewhat** (Low quality response. May include some inconsistencies or grammar mistakes.)
○ **2.5**
○ **2 - Poor** (Bad response. Not related to the original comment or includes severe grammar mistakes.)
○ **1.5**
○ **1 - Invalid** (Not a valid response. No possible connection with the original comment or is grammatically incorrect.)

Figure 7: Warm-up part, starting each AMT HDS task. It includes the context, and a single response generated by the tester. The worker is asked to generate response of hers/his own and rate the quality of the tester's response.

Response set ($p = 0.208$)	**Response set** ($p = 0.64$)	**Response set** ($p = 1$)
• **So that’s the** story of the last few years. The current political climate is not! • **So that’s the** end of the first part of this series. I hope you enjoyed it.! • **So that’s the** first thing I want to say. I’m not going to be the guy! • **So that’s the** thing about being a professional. You have to be able to handle the criticism! • **So that’s the** way it is. I don’t think there’s any way to change it! • **So that’s the** problem. It’s not just that the government is failing to protect! • **So that’s the** thing about this. It’s not just about the money. It’s about! • **So that’s the** end of the story. The next step is to create a custom! • **So that’s the** case. So, what’s the problem? Well,! • **So that’s the** first time I’ve ever seen a real one. I’m not!	• **So that’s the** state of the campaign. Now, what I do want to talk about is! • **So that’s the** thing: For as much as I love TLC, it’s hard to! • **So that’s the** idea, anyway. The last two seasons have been about doing that. It! • **So that’s the** end of the half-hour segment. The next half-hour! • **So that’s the** situation we’re in,” he said. ”We’re in the! • **So that’s the** thing, I don’t know if you know, but in general it’s! • **So that’s the** difference between the kinds of things that people will be talking about on Wednesday,! • **So that’s the** $2.3 billion. Here’s the issue: You’re! • **So that’s the** standard for using memcpy(). It’s fine to use memc! • **So that’s the** next step, and the next step is to try to figure out what’s!	• **So that’s the** first time you want to punch somebody, not miss before.” The Seahawks would! • **So that’s the** science behind the Broadwell-E processors from Intel that Intel launched last fall! • **So that’s the** instinct from other teams, that they’re a headache. - Ramsay MacDonald,! • **So that’s the** white whale right there about too much debt. And then what you! • **So that’s the** end of our discussion about the causes. What happens when we look at the! • **So that’s the** cover of inhibition against ”chronic” or ”adaptive” stimulants! • **So that’s the** way the story goes, but exactly how is cloud providers going to restrict Their! • **So that’s the** beginning, the beginning of the show, I guess five minutes.” ! • **So that’s the** Indie Mobile Game Week Honoring Winners!!!!!!!!! • **So that’s the** reason I’m writing, that’s why you don’t understand why people know!
• **do you listen** to the music?” ”I don’t know. I don’t listen! • **do you listen** to them?” ”I do,” he said. ”I’m not! • **do you listen** to the voices of the people?” ”I do,” said the king! • **do you listen** to the song?” ”I don’t know . I don’t know! • **do you listen** to the music?” ”I do.” ”You’re not! • **do you listen** to the news? I do. I’m a big fan of the! • **do you listen** to me?” ”Yes, I do.” ”I’m! • **do you listen** to the other side?” ”I don’t know. I don’t! • **do you listen** to the other side?” ”I do,” said the boy. ”! • **do you listen** to the news? No, I don’t. I don’t listen!	• **do you listen** to the current draft? I listen to the current draft. I’m! • **do you listen** to it?” It’s easy to hear the ”why?” but when! • **do you listen** to the people that come here?” ”No, I’m too busy! • **do you listen** to the thing?” ”Of course I do. I’ve been reading! • **do you listen** to those who are opposing it, who want to create a situation in which a! • **do you listen** to music or watch TV? How often do you cook or clean? How much! • **do you listen** to them? It’s like the first time you got into something and it just! • **do you listen** to your father? We’ll leave it to the gods to decide.” ! • **do you listen** to music? I like to listen to music, but I don’t really know! • **do you listen** to my story and see if you like it?” ”I think you!	• **do you listen** to Human Fly?, which YouTuber Nico Perri collaborated on, and Google! • **do you listen** to the acapella lyrics out of context and express the feeling?” It’s! • **do you listen** to Michael Kiwanuka-Smith who writes, ”The American Journalism Review discern! • **do you listen** to my songs as I said,” Ramckhalter said. ”You feel! • **do you listen** to U.S. 90 night at this time of the year? ! • **do you listen** to that as well?” ”The question was not, ’Who is! • **do you listen**?” He asks, leaning forward as he woodenly talks to him. ”Listen! • **do you listen** to those books and sway him so much? No. He was deeply brainwashed! • **do you listen**?’ Simon(lol).I feel like i’m in a Kurdish Genocide. I! • **do you listen** to value authenticated queries from your menu when running count? And if not, then!

Table 18: decTest data samples for promptGen task and different p values (nucleus sampling). Bold text is the 3-words prompt context.



All proposed responses:

- It's possible but unlikely it will happen.
- No way dude not at all.
- Yeah I think that would be a big help.
- That's the dumbest thing I've heard all day.
- I think you are on to something with that!

How diverse are the proposed responses?

○ **5 - Very diverse** (The responses have significant differences from one another.)
○ **4.5**
○ **4 - Diverse** (The responses are considerably different from one another.)
○ **3.5**
○ **3 - Slightly diverse** (The responses tend to be similar but not the same.)
○ **2.5**
○ **2 - Almost not diverse** (The responses are almost the same.)
○ **1.5**
○ **1 - Not diverse at all** (The responses are identical or almost identical.)

Figure 8: absHDS question along with the evaluated response set (conTest in this case).

Response set ($k = 3$)	**Response set** ($k = 32$)	**Response set** ($k = 318$)
• **I’m kidding. You** can’t do that. ”I’ve been trying to find! • **I’m kidding. You** can’t be serious.” The man was a little surprised.! • **I’m kidding. You** can’t do that, because the game’s a little bit different.! • **I’m kidding. You**’re the only one who can do it, you know. You’re! • **I’m kidding. You** can have it. ”It’s not a bad idea,”! • **I’m kidding. You** don’t need to be so serious.” ”No, I! • **I’m kidding. You**’re not. I’m just a little bit of a dickhead.! • **I’m kidding. You**’re not. You’re not. I know. I know. You! • **I’m kidding. You** can’t do it. ”I don’t want to be! • **I’m kidding. You**’re not going to be able to see it, and you’ll be!	• **I’m kidding. You**’re too nice. We need someone in the government to talk! • **I’m kidding. You** just sit there, I’ll have the other side do it. What! • **I’m kidding. You** can be my roommate for the holidays in a few weeks. You don! • **I’m kidding. You** can get the full version of the file above, and also the original! • **I’m kidding. You** don’t look very strong.” ”You look as strong as! • **I’m kidding. You**’re right about the last guy. He’s a nice guy. But! • **I’m kidding. You**’ll be happy to know it’s the first time that’s happened since! • **I’m kidding. You** will come down with the same problem, or some other sort of problem! • **I’m kidding. You** might have seen me in a lot of other things. I’m actually! • **I’m kidding. You** should go and see a doctor. In fact, I’m!	• **I’m kidding. You**’re kidding?” ”I’m not.” ”Why! • **I’m kidding. You**’re not.” ”What? A dick of the heart?! • **I’m kidding. You**’re looking at a new version,” said Zilch, who was! • **I’m kidding. You** know when someone takes to the streets to protest? It’s common for! • **I’m kidding. You** are definitely a complete free agent,” said Caruthers. ! • **I’m kidding. You** can have another at first, but don’t start just jumping ahead/! • **I’m kidding. You**’re just a teenager, aren’t you?” It ends there, your! • **I’m kidding. You** were never fully persuaded.” ”Perfect, I am not,”! • **I’m kidding. You** are also in a worse case scenario for someone who was on $2500! • **I’m kidding. You** know. . . ” ”I should have stopped him; I shouldn’t!
• **Where did he** go?” I ask, looking at him. ”I’m not sure. He! • **Where did he** get the idea to do this? He had been working on a book! • **Where did he** come from? He was born in the city of Karkaros! • **Where did he** go?” I asked. ”I don’t know,” she said. ! • **Where did he** go?” ”I think he went to the hospital,” she said.! • **Where did he** get the idea for the name? I think it’s a combination of! • **Where did he** get the idea to make a movie about the Holocaust? ”I had a lot! • **Where did he** get that idea? ”I was just trying to make a statement,”! • **Where did he** get that from? He’s a very good writer. I don’t know what! • **Where did he** go? Where was he? Where was he? He’s gone. !	• **Where did he** come back from? [The Doctor is sitting in a chair. Amy! • **Where did he** find the money?” asked a reporter from the BBC. ”Is anybody else there! • **Where did he** grow up?” But the boy answered, ”He always loved to read! • **Where did he** get that idea?” he asked. ”I didn’t know. I’ve never! • **Where did he** come from?” You’re looking for that missing piece. Maybe you’re missing the! • **Where did he** come from? He was, I think, from a small island about midway between! • **Where did he** come from, to be sure?” he asked, ”I know he came from! • **Where did he** go? [A little while later] I am about to say this! • **Where did he** hear about my story? I couldn’t tell you. He’d only heard of! • **Where did he** come from? From a place called ”the City of the Sun.”!	• **Where did he** at the time in his day seek the God he worshipped? He said: ”! • **Where did he** earn his master’s degree? He is part of a class of doctoral students who! • **Where did he** learn to play guitar?” I asked, puzzled. ”Before I joined! • **Where did he** come from?” ”Australia,” said Peter. ”How could! • **Where did he** hear this, you might ask? Of course, he’d heard of it.! • **Where did he** go? He’s probably dead – or dead and buried within the walls! • **Where did he** earn $150 million on his way to a $5 billion makeover? ! • **Where did he** learn to make his own sticks, or for that matter, hang a stick on! • **Where did he** learn to skate, anyway? Go here and watch this beautiful skater! • **Where did he** get this idea from? What do you think about it? I get!

Table 19: decTest data samples for promptGen task and different k values (Top-k). Bold text is the 3-words prompt context.

Context	**Response set (high content diversity)**	**Response set (low content diversity)**
Sold Out Jane wanted to watch a big new action movie. She had been waiting a long time for it to come out. When tickets became available she was too busy. By the time she had a chance to buy some it was sold out.	• Jane cried over the fact that she couldn’t watch it and just gave up looking for a ticket. • Jane decided to look for a scalper that would sell her the ticket for the movie that she really wanted to see. • Jane thought it was okay since she can still have a chance to watch it once it gets uploaded in video and movie streaming applications. • Jane posted a status on her social media accounts asking her friends for any spare ticket that she is willing to buy. • Jane resorted to contacting her old friend who is working at a huge movie theater hoping she can help her get a ticket.	• Jane remembered that she has an old friend who is a manager at a big movie theater so she contacted that friend in the hopes that she can buy any spare ticket. • Desperate to watch the movie, Jane called her friend, who works at a movie theater, asking for a ticket to that movie. • Jane recalled that her friend works at a movie theater and hoped that she can help get a ticket for that movie. • Jane decided to look for her friend who could possibly have access to tickets for that movie since that friend currently works at a movie theater. • Jane realized that her friend might have spare tickets since she is a manager of a movie theater showing that film.
Beavers. My friend has some beavers in his backyard. They come up from the creek by his house. He invites my over and we watch them. We take pictures of them and send them to our friends.	• They are fascinating animals. • Our friends love getting the pictures. • Sometimes his dogs chase them. • They are building a dam on the creek. • They won’t let us get too close to them.	• They are busy gathering sticks to make a dam. • The dam they are building is almost complete. • It’s fascinating to see their workmanship building a dam. • They are turning the creek into a pond by building a dam. • They all work together with careful engineering to build a dam.

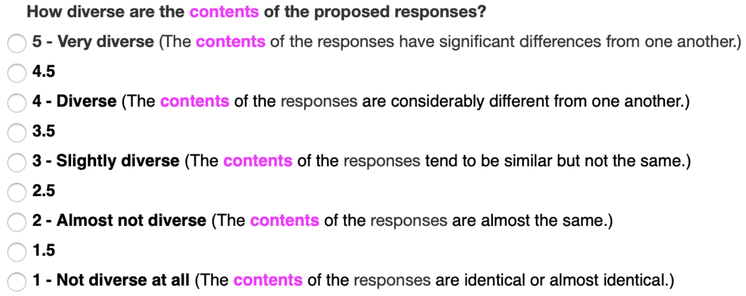
Table 20: conTest data samples for storyGen task.

Context	**Response set (high content diversity)**	**Response set (low content diversity)**
kill la kill is still going new episode every thursday	• That show sucks • OMG I can’t wait • I thought they canceled it • What channel is it on • I only watch nature programs on BBC	• Lead actor is soooo hot • Did you see the cliffhanger at the end of the season • I’ve been waiting for it to return for weeks • I’m totally gonna binge watch last season • I just got into this show and can’t stop watching
places apple slices in a bowl so they’ll stay fresh	• Oh boy, I love apples. • I don’t need you telling me how to keep things fresh, take a hike. • Girl, you’re the fresh one around here. • This post might be better in the life hacks section. • This is actually a useful bit of advice.	• I find merit in this input. • That information will serve me well. • Thanks, that’s really good to know! • Such knowledge is certainly beneficial. • Wise words, I will heed them.

Table 21: conTest data samples for respGen task.

Response set (high content diversity)	Response set (low content diversity)
• **Suppose there's an** escape plan we haven't thought of yet. • **Suppose there's an** omelet that is the most amazing ever. • **Suppose there's an** airplane ticket that's even cheaper. • **Suppose there's an** actual deadline for this paper. • **Suppose there's an** event that we can go to this weekend.	• **Suppose there's an** airline that costs less. • **Suppose there's an** flight that isn't as expensive. • **Suppose there's an** air travel fare, but doesn't cost as much. • **Suppose there's an** way to fly there that is low cost. • **Suppose there's an** flight going there and it's not a lot of money
• **Nothing remotely like** eating a big breakfast. • **Nothing remotely like** dancing with your wife at the wedding. • **Nothing remotely like** singing Justin Bieber's greatest hits • **Nothing remotely like** falling down a hill • **Nothing remotely like** getting yelled at	• **Nothing remotely like** being super full and satisfied. • **Nothing remotely like** getting to taste many different foods. • **Nothing remotely like** starting the day off right. • **Nothing remotely like** doing exactly what I want to do. • **Nothing remotely like** feeding myself with great food.

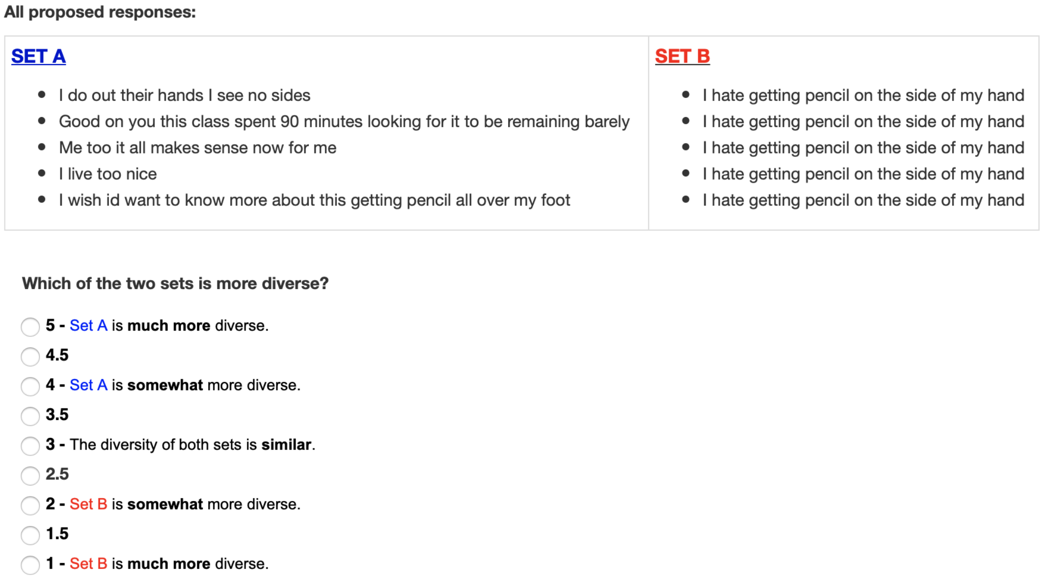
Table 22: conTest data samples for promptGen task. Bold text is the 3-words prompt context.



How diverse are the contents of the proposed responses?

○ **5 - Very diverse** (The contents of the responses have significant differences from one another.)
○ **4.5**
○ **4 - Diverse** (The contents of the responses are considerably different from one another.)
○ **3.5**
○ **3 - Slightly diverse** (The contents of the responses tend to be similar but not the same.)
○ **2.5**
○ **2 - Almost not diverse** (The contents of the responses are almost the same.)
○ **1.5**
○ **1 - Not diverse at all** (The contents of the responses are identical or almost identical.)

Figure 9: aspHDS question (content in this case). The response set is the same as presented for absHDS question.



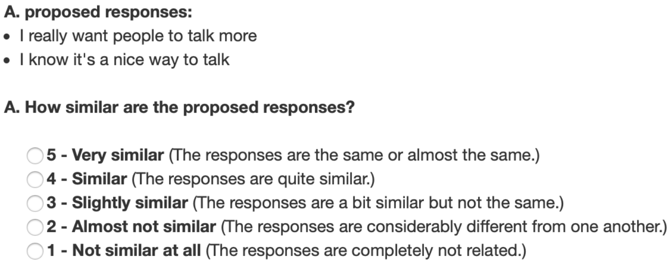
All proposed responses:

SET A	SET B
• I do out their hands I see no sides • Good on you this class spent 90 minutes looking for it to be remaining barely • Me too it all makes sense now for me • I live too nice • I wish id want to know more about this getting pencil all over my foot	• I hate getting pencil on the side of my hand • I hate getting pencil on the side of my hand • I hate getting pencil on the side of my hand • I hate getting pencil on the side of my hand • I hate getting pencil on the side of my hand

Which of the two sets is more diverse?

○ **5 -** Set A is **much more** diverse.
○ **4.5**
○ **4 -** Set A is **somewhat** more diverse.
○ **3.5**
○ **3 -** The diversity of both sets is **similar**.
○ **2.5**
○ **2 -** Set B is **somewhat** more diverse.
○ **1.5**
○ **1 -** Set B is **much more** diverse.

Figure 10: rnkHDS question along with the two evaluated response sets.



A. proposed responses:

- I really want people to talk more
- I know it's a nice way to talk

A. How similar are the proposed responses?

○ **5 - Very similar** (The responses are the same or almost the same.)
○ **4 - Similar** (The responses are quite similar.)
○ **3 - Slightly similar** (The responses are a bit similar but not the same.)
○ **2 - Almost not similar** (The responses are considerably different from one another.)
○ **1 - Not similar at all** (The responses are completely not related.)

Figure 11: simHDS question along with the two evaluated responses.

Retrieval, Re-ranking and Multi-task Learning for Knowledge-Base Question Answering

Zhiguo Wang, Patrick Ng, Ramesh Nallapati, Bing Xiang
AWS AI Labs
{zhiguow, patricng, rnallapa, bxiang}@amazon.com

Abstract

Question answering over knowledge bases (KBQA) usually involves three sub-tasks, namely topic entity detection, entity linking and relation detection. Due to the large number of entities and relations inside knowledge bases (KB), previous work usually utilized sophisticated rules to narrow down the search space and managed only a subset of KBs in memory. In this work, we leverage a *retrieve-and-rerank* framework to access KBs via traditional information retrieval (IR) method, and re-rank retrieved candidates with more powerful neural networks such as the pre-trained BERT model. Considering the fact that directly assigning a different BERT model for each sub-task may incur prohibitive costs, we propose to share a BERT encoder across all three sub-tasks and define task-specific layers on top of the shared layer. The unified model is then trained under a multi-task learning framework. Experiments show that: (1) Our IR-based retrieval method is able to collect high-quality candidates efficiently, thus enables our method adapt to large-scale KBs easily; (2) the BERT model improves the accuracy across all three sub-tasks; and (3) benefiting from multi-task learning, the unified model obtains further improvements with only 1/3 of the original parameters. Our final model achieves competitive results on the SimpleQuestions dataset and superior performance on the FreebaseQA dataset.

1 Introduction

Answering natural language questions by searching over large-scale knowledge bases (KBQA) is highly demanded by real-life applications, such as Google Assistant, Siri, and Alexa. Owing to the availability of large-scale KBs, significant advancements have been made over the years. One main research direction views KBQA as a *semantic matching* task (Bordes et al., 2014; Dong et al., 2015; Dai et al., 2016; Hao et al., 2017; Mohammed et al., 2018; Yu et al., 2018; Wu et al., 2019; Chen et al., 2019a; Petrochuk and Zettlemoyer, 2018), and finds a relation-chain within KBs that is most similar to the question in a common semantic space, where the relation-chain can be 1-hop, 2-hop or multi-hop (Chen et al., 2019b). Another research direction formulates KBQA as a *semantic parsing* task (Berant et al., 2013; Bao et al., 2016; Luo et al., 2018), and tackles questions that involve complex reasoning, such as ordinal (e.g. What is the second largest fulfillment center of Amazon?), and aggregation (e.g. How many fulfillment centers does Amazon have?). Most recently, some studies proposed to derive answers from both KBs and free-text corpus to deal with the low-coverage issue of KBs (Xu et al., 2016; Sun et al., 2018; Xiong et al., 2019; Sun et al., 2019). In this paper, we follow the first research direction since the relation-chain type of questions counts the vast majority of real-life questions (Berant et al., 2013; Bordes et al., 2015; Jiang et al., 2019).

Previous semantic matching methods for KBQA usually decompose the task into sequential subtasks consisting of topic entity detection, entity linking, and relation detection. For example in Figure 1, given the question “Who wrote the book Beau Geste?”, a KBQA system first identifies the topic entity “Beau Geste” from the question, then the topic entity is linked to an entity node (m.04wxy8) from a list of candidate nodes, and finally the relation *book.written_work.author* is selected as the relation-chain leading to the final answer. Previous methods usually worked on a subset of KB in order to fit KB into memory. For entity linking, some sophisticated heuristics were commonly used to collect entity candidates. For relation detection, previous work usually enumerated all possible 1-hop and 2-hop relation-chains (starting from linked entity nodes) as candidates. All

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 347–357
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

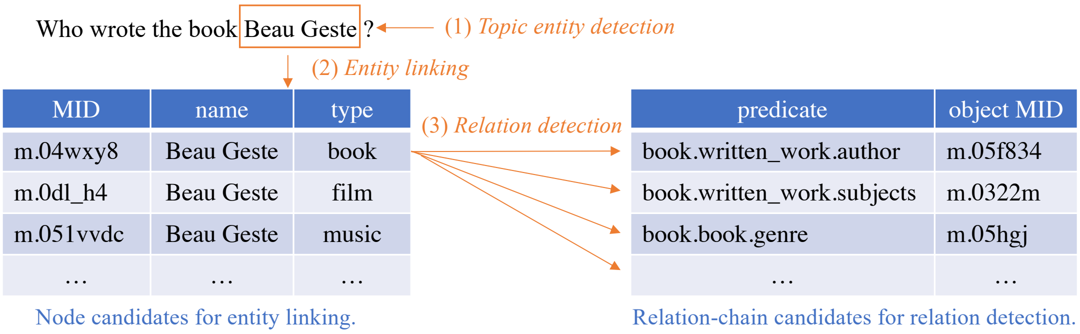


Figure 1: A typical workflow for KBQA. Given a question "Who wrote the book Beau Geste?", the topic entity detection model first identifies a topic entity "Beau Geste" from the question. Then, the entity linking model links the topic entity into an entity node (m.04wxy8) in the KB. Finally, the relation *book.written_work.author* is selected as the relation-chain leading to the final answer node (m.05f834).

these workarounds may prevent their methods from generalizing well to other datasets, and scaling up to bigger KBs.

To tackle these issues, we leverage a *retrieve-and-rerank* strategy to access KBs. In the *retrieval* step, we ingest KBs into two inverted indices: one that stores all entity nodes for entity linking, and the other one that stores all subject-predicate-object triples for relation detection. Then, we use TF-IDF algorithm to retrieve candidates for both entity linking and relation detection sub-tasks. This method naturally overcomes the memory overhead when dealing with large-scale KBs, therefore makes our method easily scale up to large-scale tasks. In the *re-ranking* step, we leverage the advanced BERT model to re-rank all candidates by fine-grained semantic matching. For the topic entity detection sub-task, we utilize another BERT model to predict the start and end positions of a topic entity within a question. Since assigning a different BERT model for each sub-task may incur prohibitive costs, we therefore propose to share a BERT encoder across sub-tasks and define task-specific layers for each individual sub-task on top of the shared layer. This unified BERT model is then trained under the multi-task learning framework. Experiments on two standard benchmarks show that: (1) Our IR-based retrieval method is able to collect high-quality candidates efficiently; (2) the BERT model improves the accuracy across all three sub-tasks; and (3) benefiting from multi-task learning, the unified model obtains further improvements with only 1/3 of the original parameters. Our final model achieves competitive results on the SimpleQuestions dataset and superior performance on the FreebaseQA dataset.

2 Task Definition

Knowledge-base question answering (KBQA) aims to find answers for natural language questions from structural knowledge bases (KB). We assume a KB $\mathcal{K}$ is a collection of subject-predicate-object triples $\langle e_1, p, e_2 \rangle$, where $e_1, e_2 \in \mathcal{E}$ are entities, and $p \in \mathcal{P}$ is a relation type between two entities, $\mathcal{E}$ is the set of all entities, and $\mathcal{P}$ is the set of all relation types. Given a question Q, the goal of KBQA is to find an entity node $a \in \mathcal{E}$ from the KB as the final answer, thus can be formulated as

$$\hat{a} = \arg\max_{a \in \mathcal{E}} Pr(a|Q, \mathcal{K}) \tag{1}$$

where $Pr(a|Q, \mathcal{K})$ is the probability of a to be the answer for Q. A general purpose KB usually contains millions of entities in $\mathcal{E}$ and billions of relations in $\mathcal{K}$ (Bollacker et al., 2008), therefore directly modeling $Pr(a|Q, \mathcal{K})$ is challenging. Previous studies usually factorize this model in different ways. One line of research forms KBQA as a *semantic parsing* task $Pr(q|Q, \mathcal{K})$ to parse a question Q directly into a logical form query q, and execute the query q over KB to derive the final answer. Another line of studies views KBQA as a *semantic matching* task, and finds a relation-chain within KB that is similar to the question in a common semantic space. Then the trailing entity of the relation-chain is taken as the final answer. Following this direction, we decompose the KBQA task into three stages: (1) identify a topic entity t from the question Q, where t is a sub-string of Q; (2) link the topic entity t to a topic node $e \in \mathcal{E}$ in KB; and (3) detect a relation-chain $r \in \mathcal{K}$ starting from the topic node e, where r can be 1-hop, 2-hop or multi-hop relation-chains within KB. Correspondingly, we factorize the model as

$$\begin{aligned} Pr(a|Q,\mathcal{K}) &= Pr(t,e,r|Q,\mathcal{K}) \\ &= P_t(t|Q,\mathcal{K})P_l(e|t,Q,\mathcal{K}) \\ &\qquad P_r(r|e,t,Q,\mathcal{K}) \end{aligned} \quad (2)$$

where $P_t(t|Q,\mathcal{K})$ is the model for topic entity detection, $P_l(e|t,Q,\mathcal{K})$ models the entity linking process, and $P_r(r|e,t,Q,\mathcal{K})$ is the component for relation detection stage. We will discuss how to parameterize these components in Section 4.

3 Background

We briefly introduce some background required by the following sections.

BERT: BERT model (Devlin et al., 2019) follows the multi-head self-attention architecture (Vaswani et al., 2017), and is pre-trained with a masked language modeling objective on a large-scale text corpus. It has achieved state-of-the-art performance on a bunch of textual tasks. Specifically, for semantic matching tasks, BERT simply concatenates two textual sequences together, and encodes the new sequence with multiple self-attention layers. Then, the output vector of the first token is fed into a linear layer to compute the similarity score between the two input textual sequences.

Freebase: We take Freebase (Bollacker et al., 2008) as our back-end KB to answer questions. It contains more than 46 million topic entities and 2.6 billion triples. Each entity has an internal machine identifier (MID) and a set of aliases. Some entities also have properties such as entity types and detailed descriptions. Freebase contains a special entity category called *Compound Value Type* (CVT), which does not have a name or alias, and is only used to collect multiple fields of an event or a special relationship. In the official Freebase dump [1], all facts are formulated as the unified subject-predicate-object triples, and there is no explicit split for entities and relations. We partition facts in Freebase into a set of entities $\mathcal{E}$ and a set of relations $\mathcal{K}$ by following the pre-processing steps in Chah (2017).

Inverted Index and TF-IDF: *Inverted index* is an optimized data structure of finding documents (from a large document collection) where a query word X occurs. It is commonly used for fast free-text searches. Term Frequency-Inverse Document Frequency (TF-IDF) is a ranking function usually used together with an inverted index to estimate the relevance of documents to a given search query (Schütze et al., 2008).

4 Retrieval and Re-ranking for KBQA

In this section, we describe how to parameterize P_t, P_l and P_r in Equation (2).

4.1 Topic Entity Detection Model P_t

The goal of a topic entity detection model $P_t(t|Q,\mathcal{K})$ is to identify a topic entity t that the question Q is asking about, where t is usually a substring of Q. Previous approaches for this task can be categorized into two types: (1) rule-based and (2) sequence labeling. The rule-based approaches take all entity names and their alias from a KB as a gazetteer, and n-grams of the question that exactly match with an entry in the gazetteer are taken as topic entities (Yih et al., 2015; Yao, 2015; He and Golub, 2016; Yu et al., 2017). The advantage of this method is that no machine learning models need to be involved. However, the drawbacks include: (1) topic entities need to have the exact same surface strings as they occur in KB, and (2) memory-efficient data structures need to be designed to load the massive gazetteer into memory (Yao, 2015). Other approaches leverage a sequence labeling model to tag consecutive tokens in the question Q as topic entities (Dai et al., 2016; Bordes et al., 2015; Mohammed et al., 2018; Wu et al., 2019). This approach is able to predict more precise topic entities, thus prunes some unimportant matched entities.

Inspired from the Start/End prediction method commonly utilized for machine reading comprehension tasks (Wang and Jiang, 2016; Seo et al., 2016), we cast the topic entity detection task into predicting the start and end positions of the topic entity t in the question Q. Formally, we denote t_s and t_e as the start and end positions for a topic entity t, and assume this process is independent of KB. Thus the model can be further decomposed as $P_t(t|Q,\mathcal{K}) = P_s(t_s|Q)P_e(t_e|Q)$, where $P_s(t_s|Q)$ and $P_e(t_e|Q)$ are the probabilities of t_s and t_e to be the start and end positions. This formulation directly models the goal of the topic entity detection task, i.e. finding the best topic entity within a question, therefore can give a more precise estimation.

We leverage the advanced BERT model to parameterize $P_s(t_s|Q)$ and $P_e(t_e|Q)$. Concretely, we

---

[1] https://developers.google.com/freebase

first leverage BERT encoder to encode the input question Q, then apply two independent linear layers (with one output neuron) on top of BERT's output for each token. The start/end scores are normalized across all tokens with the $softmax$ function to estimate the probabilities of each token position to be the start/end of the topic entity.

4.2 Entity Linking Model P_l

The purpose of an entity linking model $P_l(e|t, Q, \mathcal{K})$ is to link the recognized topic entity t to an entity node $e \in \mathcal{E}$ in KB. A general purpose KB usually contains millions of nodes in $\mathcal{E}$, which makes it almost impossible to search over the full space. Previous methods usually narrow down the search space based on some heuristic rules. For example, Yih et al. (2015) and Wu et al. (2019) used keyword search to collect all nodes that have one alias exactly matching the topic entity, and Yin et al. (2016) collected all nodes that have at least one word overlapping with the topic entity. Once a smaller set of candidates is selected, complicated neural networks can be utilized to compute the similarity between a candidate node and the topic entity in the question context.

Inspired from the recent success of question answering over free-text corpus (Chen et al., 2017; Wang et al., 2018, 2019), we propose a *retrieve-and-rerank* method to solve the entity linking task in two steps. In the first *retrieval* step, we create an inverted index for all entity nodes, where each node is represented with all tokens from its aliases and description. Then, we use the topic entity t as a query to retrieve top-K candidate nodes from the index with the TF-IDF algorithm[2]. The similar method is also used by Vakulenko et al. (2019) and Nedelchev et al. (2020). This information retrieval (IR) method is better than previous work in the following ways. First, our method can find candidate nodes even if a topic entity does not have an exactly matched entity node. Second, we do not have to maintain all entity nodes inside CPU memory, and can still query candidates efficiently, which enables our method to be easily adapted to large-scale KBs. Third, the relative importance of various matched words is naturally considered in the TF-IDF algorithm.

In the second *re-ranking* step, we leverage BERT model to compute the similarity between each candidate node and the topic entity in the given question context. Concretely, we represent each pair of a topic entity t and a candidate node e as a sequence of tokens with the format " `[CLS] topic entity [SEP] question pattern [SEP] node name [SEP] node types [SEP] node description [SEP]`", where `topic entity` is the string for the topic entity t, `question pattern` is the question string with t being removed, `node name`, `node types` and `node description` are the name, types and description for the topic node e, and `[SEP]` is the delimiter used by BERT model. We encode this sequence with BERT model, then feed the hidden vector for the token `[CLS]` into a linear layer (with one output neuron) to compute the similarity score for each pair of t and e.

4.3 Relation Detection Model P_r

The relation detection model $P_r(r|e, t, Q, \mathcal{K})$ traverses relation-chains starting from a linked topic node e, and attempts to detect the correct relation-chain r that answers the question Q. Previous work usually enumerates all possible 1-hop and 2-hop relation-chains starting from a linked topic node e, and leverages deep neural networks to compute semantic similarity between each candidate relation-chain r and the question Q (Bordes et al., 2014; Yih et al., 2015; Dong et al., 2015; Yu et al., 2017; Wu et al., 2019). In real KBQA systems, usually, a list of linked nodes from the entity linking step is considered to retain high recall. If we enumerate all relation-chains for all these linked topic nodes, we will end up with a large collection of candidate relation-chains. Furthermore, re-ranking so many candidate relation-chains will add much run-time latency, especially when a heavy model such as BERT is utilized.

To address this issue, we propose to use the *retrieve-and-rerank* method for the relation detection task, and deal with this task in two stages similar to what we did for the entity linking task. In the first *retrieval* step, we create an inverted index for all subject-predict-object triples, where each triple is represented as all tokens from the name of the subject entity, the name of the predicate, and types of the object entity. Then, we use the question Q as a query to retrieve top-K 1-hop relation-chains with the constraint that all subject nodes are from the list

---

[2] Mohammed et al. (2018) also created an inverted index for all nodes. However, they generated ngrams of each entity name into several entries, and looked up exactly matched ngram candidates by keyword searching.

of linked entity nodes. If two-hop relation-chains are required in a target dataset, we will do the same querying step again, but with the constraint list being all object entities from the first retrieval step. We acknowledge that this method does not consider the already covered semantics in the first retrieval step, when we do the second step retrieval. Since the main goal of the retrieval step is to collect a list of high-quality candidates, we will perform better semantic matching in the re-ranking step with more powerful neural networks. If multi-hop relation-chains are needed, we can iterate this process until reaching the maximum steps. Usually, the number of max-hop is pre-computed on the target question sets. Another way is to utilize a model to decide when to stop (Chen et al., 2019b), however we will leave this option in the future work.

After collecting a list of relation-chains, we leverage another BERT model to compute the similarity between a question Q and each relation-chain r. Each pair of Q and r will be represented as a sequence of tokens with the format "`[CLS] question [SEP] topic-entity name [SEP] relation chain [SEP] answer name [SEP] answer types [SEP]`", where `topic-entity name` is the name for the linked entity node, `relation chain` is the word sequence of a candidate relation-chain[3], `answer name` is the name of the trailing node in the relation-chain, and `answer types` are all types of the trailing node. The hidden vector for the `[CLS]` token will be fed into a linear layer (with one output neuron) to predict the similarity between Q and r.

5 Multi-Task Learning for KBQA

5.1 Training Objectives

For the topic entity detection model, we define the objective function as the cross-entropy loss between true distributions and predicted distributions. We sum up the cross-entropy losses for both start and end models, and average over all N training instances:

$$L(\theta_t) = -\frac{1}{N}\sum_{i=1}^{N}\log(P_s^i) + \log(P_e^i) \qquad (3)$$

where θ_t is the trainable parameter for topic entity detection model.

Both entity linking and relation detection tasks are ranking tasks, therefore we leverage a hinge loss function for both tasks:

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\max(0, l + s(Q, c^-) - s(Q, c^+)) \qquad (4)$$

Where θ is the trainable parameter, l is a margin, $s(Q, c)$ can be the model of P_l or P_r, c^+ is a correct candidate, and c^- is an incorrect candidate. We set $l = 1.0$ in this work.

5.2 Multi-Task Learning

A naive approach would be to use three different BERT encoders for the topic entity detection, entity linking and relation detection sub-tasks individually. Since BERT model is a very large model, it is expensive to host three BERT models in real applications. To address this, we propose to share a BERT encoder across all three sub-tasks, and define lean layers for each individual sub-task on top of the shared layer. This unified model is then trained under the multi-task learning framework proposed by Liu et al. (2019). First, training instances for each sub-tasks are packed into mini-batches separately. At the beginning of each training epoch, mini-batches from all three sub-tasks are mixed together and randomly shuffled. During training, a mini-batch is selected, and the model is updated according to the task-specific objective for the selected mini-batch.

6 Experiments

We evaluate the effectiveness of our model on standard benchmarks in this section. We first conduct experiments on each sub-task with a separate BERT model in Section 6.2, 6.3 and 6.4, then evaluate the influence of sharing a BERT encoder for all three models in Section 6.5. Finally, we benchmark our method on full Freebase in Section 6.6.

6.1 Datasets and Basic Settings

We evaluate our proposed model on two large-scale benchmarks: SimpleQuestions and FreebaseQA. Other existing datasets, such as WebQuestions (Berant et al., 2013), Free917 (Cai and Yates, 2013) and WebQSP (Yih et al., 2016), are not considered, because they only contain few thousands of questions which is even less than the number of relation types in Freebase.

SimpleQuestions: The SimpleQuestions dataset (Bordes et al., 2015) is so far the largest

---

[3] A relation-chain is split into a word sequence based on delimiters such as periods, hyphens and underscores.

KBQA dataset. It consists of 108,442 English questions written by human annotators, and all questions can be answered by 1-hop relation chains in Freebase. Each question is annotated with a gold-standard subject-relation-object triple from Freebase. We follow the official train/dev/test split. To fairly compare with previous work, we leverage the released FB2M subset of Freebase as the back-end KB for this dataset. FB2M includes 2M entities and 5k relation types between these entities.

FreebaseQA: FreebaseQA dataset (Jiang et al., 2019) is a large-scale dataset with 28K unique open-domain factoid questions which are collected from triviaQA dataset (Joshi et al., 2017) and other trivia websites. Each question can be answered by a 1-hop or 2-hop relation-chain from Freebase. All questions have been matched to subject-predicate-object triples in Freebase, and verified by human annotators. Comparing with other KBQA datasets, FreebaseQA provides more linguistically sophisticated questions, because all questions are created independently from Freebase. FreebaseQA also released a new subset of Freebase, which includes 16M unique entities, and 182M triples. We follow the official train/dev/test split, and take the Freebase subset as the back-end KB for this dataset.

Basic Settings: We leverage the pre-trained BERT-base model with default hyper-parameters in our experiments. We create inverted indices for topic nodes and relations with Elasticsearch[4], and utilize the BM25 (a variance of TF-IDF) algorithm to retrieve inverted indices.

6.2 Topic Entity Detection Experiments

In order to train and evaluate our topic entity detection model, we annotate the ground-truth topic entity for each question with the following steps. First, for each question, all alias names for the annotated topic entity MID are collected from Freebase. Then, we match each alias against the question string. If more than one alias occurs in the question string, the longest matched string will be annotated as the ground-truth. Otherwise, the span with the minimum edit distance will be selected as the ground-truth.

We implement a BERT-based sequence labeling model as a baseline for our Start/End prediction model described in Section 4.1. The baseline model follows the same architecture for the named entity recognition (NER) task in Devlin et al. (2019), where we use BIO schema to annotate each question token. Since the sequence labeling method may predict multiple spans to be topic entities, we choose the span with the maximum average token score as the final prediction.

	SimpleQ.		FreebaseQA	
Models	EM	F_1	EM	F_1
BIO	94.9	97.3	65.1	75.2
Start/End P_t	**96.4**	**97.8**	**74.3**	**81.5**
Multi-task P_t	96.0	97.7	70.6	79.3

Table 1: Results for topic entity detection.

We employ the metrics exact match (EM) and F_1 proposed in Rajpurkar et al. (2016) to evaluate the identified topic entities. Experimental results are shown in Table 1. We can see that our Start/End prediction model works better than the BIO sequence labeling baseline. Specifically, in FreebaseQA dataset, since the questions are longer and more complicated, our Start/End model outperforms the BIO sequence labeling model by a large margin.

6.3 Entity Linking Experiments

We retrieve a list of candidate nodes for each question as follows. For questions in the training sets, we use the ground-truth topic entity as the query to retrieve top-100 candidate nodes. For questions in the dev and testing sets, we use top-N predicted topic entities as queries, and retrieve top-50 candidates for each topic entity. All candidates are then sorted based on their popularity (number of out-going triples). Based on the results on dev sets, we set N=1 for the SimpleQuestions dataset, and N=5 for the FreebaseQA dataset. We employ the top-K accuracy to evaluate entity linking results, where an instance is correct if there is at least one correct candidate inside the top-K candidate list.

Retrieval step: We implement a *Keyword-search* baseline for the retrieval step. In this baseline, all nodes, having an alias exactly matching with the topic entity, are collected as candidates. All candidates are sorted based on their popularity, i.e. the number of out-going triples. Table 2 lists the results of the baseline as well as our IR-based method proposed in Section 4.2. Our IR-based method gets better results than the Keyword baseline on both datasets. The main reason is that our

---

[4] https://www.elastic.co/products/elasticsearch

	SimpleQuestions		FreebaseQA	
	Keyword	IR	Keyword	IR
Top-1	24.4	75.7	39.0	35.3
Top-5	55.0	86.7	70.1	72.7
Top-10	68.8	89.2	76.1	81.1
Top-50	89.3	**93.7**	81.8	89.4
Top-100	92.7	93.7	82.9	**89.8**

Table 2: Qualitative analysis on entity linking candidates for the retrieval step.

IR-based method does not require exact matches between predicted topic entities and entity nodes within KB, therefore is more robust to prediction errors or entity name variances from the up-streaming topic entity detection model.

Re-ranking step: We feed top-100 candidate nodes from the retrieval step into our entity linking model P_l to re-rank all candidates. Table 3 shows results on the SimpleQuestions dataset. The first group of numbers in Table 3 are results from previous state-of-the-art models. We can see that our entity linking model P_l outperforms previous models in terms of Top-1 accuracy, and achieves competitive results in terms of Top-10 and Top-20 accuracy. Table 4 lists the results of our model and previous work on the FreebaseQA dataset. Our entity linking model P_l improves accuracy over previous work (Wu et al., 2019) by a large margin. Since top-5 predicted topic entities are used for the FreebaseQA dataset, we create another ranker to multiply together scores from both the topic entity detection model and entity linking model, and list the results in the row P_tP_l in Table 4[5]. The P_tP_l ranker gets even better Top-1 accuracy than our entity linking model P_l alone, which verifies that our factorization in Equation (2) is reasonable.

6.4 Relation Detection Experiments

We retrieve a list of relation-chain candidates for each question as follows. For questions in the training sets, we use the correct entity node as the start point to search top-100 candidates. For questions in the dev and testing sets, we use the top-N entity nodes predicted by our entity linking model as start points to retrieve top-100 candidates. For candidates with the same subject and relation type, we sort them based on the popularity of the trailing object node (number of in-coming triples), and only keep top-4 relation-chains in the final list. Based on the results on the dev set, we set N=30 for the SimpleQuetions dataset and N=10 for the FreebaseQA dataset. For the SimpleQuestions dataset, since all questions can be answered with 1-hop relation-chains, we only retrieve 1-hop candidates. For the FreebaseQA dataset, following the method in Jiang et al. (2019), we only expand 1-hop relation-chain candidates into 2-hop candidates if the object node of a 1-hop relation-chain is a CVT node. For the SimpleQuetions dataset, a prediction is correct if both the subject and relation are correctly retrieved. For the FreebaseQA dataset, a prediction is correct if the final answer matches with the ground-truth answer.

Models	Top-1	Top-10	Top-20
Yin et al. (2016)	72.7	86.9	88.4
Yu et al. (2017)	79.0	89.5	90.9
Qiu et al. (2018)	81.1	91.7	93.4
Wu et al. (2019)	82.2	**92.5**	**93.6**
P_l	84.2	92.1	93.1
Multi-task P_l	**84.3**	92.1	93.1
Full Freebase	79.0	88.9	90.3

Table 3: Entity linking results on SimpleQuestions.

Models	Top-1	Top-3	Top-5
Wu et al. (2019)	52.4	79.6	85.7
P_l	69.4	**84.8**	**86.6**
P_tP_l	**71.9**	84.6	86.3
Multi-task P_l	68.1	84.2	85.8
Multi-task P_tP_l	71.7	84.7	86.4
Full Freebase	68.1	81.6	83.8

Table 4: Entity linking results on FreebaseQA.

Retrieval step: We implement a baseline to collect all relation-chains starting from entity nodes, and sort all relation-chains based on their popularity, i.e. the in-coming triples for the trailing object. Retrieval results from the baseline are listed in the "All" columns in Table 5. The results from our IR based method (proposed in Section 4.3) are shown in the "IR" columns in Table 5. The last row "Rel/Q" in Table 5 gives the average number

---

[5] Accuracy of the P_tP_l model is not given in Table 3, because only the best (top-1) topic entity is used for retrieving entity candidates in the SimpleQuestions dataset.

	SimpleQuestions		FreebaseQA	
	All	IR	All	IR
Top-1	16.5	52.8	0.3	10.9
Top-5	53.5	80.8	1.4	20.3
Top-10	65.6	86.1	3.4	26.6
Top-50	81.9	91.7	22.8	49.8
Top-100	87.6	**92.5**	31.9	**62.6**
Rel/Q	772	100	3021	100

Table 5: Qualitative analysis for relation-chain candidates in the retrieval step, where “Rel/Q” is the average number of relation-chains per question.

of relation-chains per question. Comparing the “IR” columns with “All” columns, our IR-based method retrieves fewer relation-chains but maintains better recall.

Re-ranking step: We feed top-100 relation-chain candidates from the retrieval step into our relation detection model P_r to re-rank all candidates. Table 6 shows the results from previous state-of-the-art models as well as our relation detection model P_r. We can see that our P_r model obtains very competitive results on the SimpleQuestions dataset, and outperforms previous models by a large margin in the FreebaseQA dataset. We also create a model $P_t P_l P_r$ to multiply scores from our topic entity detection model, entity linking model and relation detection model. By considering the influence of all three components, our $P_t P_l P_r$ model achieves even better accuracy on the FreebaseQA dataset.

6.5 Multi-task Learning Experiments

Our method achieves very strong performance by leveraging three BERT encoders for each model component. In this section, we share a BERT encoder for all three models, and jointly train the unified model with the multi-task learning method described in Section 5.2. Experimental results from this model are shown in rows with the prefix “Multi-task” in Table 1, 3, 4, and 6. Although the multi-task model only has about 1/3 of the original parameters, it is able to achieve better end-to-end accuracy in Table 6, and retain similar performance as before on the other two sub-tasks.

6.6 KBQA over Full Freebase

Most of the previous studies conducted KBQA experiments with a subset of Freebase, because it is hard to fit the full Freebase into memory (Bordes et al., 2014; Dong et al., 2015). Our method ingests Freebase into inverted indices on hard disk storage, thus naturally overcomes the memory overhead. This advantage enables us to evaluate our method on the full Freebase. The last rows of Table 3, 4, and 6 show the results of running our “Multi-task” model over the full Freebase. Significant degradations are observed in entity linking and relation detection tasks on both datasets. This phenomenon reveals that previous studies may overestimate the capacity of their KBQA models. We suggest that researchers evaluate their models on the full Freebase in the future.

Models	SimpleQ.	FreebaseQA
Dai et al. (2016)	75.7	N/A
Yin et al. (2016)	76.4	N/A
Yu et al. (2017)	77.0	N/A
Wu et al. (2019)	77.3	37.0
Hao et al. (2018)	**80.2**	N/A
Petrochuk (2018)	78.1	N/A
P_r	79.4	45.4
$P_t P_l P_r$	79.4	49.1
Multi-task P_r	79.7	47.9
Multi-task $P_t P_l P_r$	79.7	**51.7**
Full Freebase	74.1	35.4

Table 6: Relation detection accuracy in the end-to-end manner.

7 Conclusion

In this work, we proposed a *retrieve-and-rerank* strategy to access large-scale KBs in two steps. First, we leveraged traditional IR methods to collect high-quality candidates from KBs for entity linking and relation detection. Second, we utilized the advanced BERT model to re-rank candidates by fine-grained semantic matching. We also employed a BERT model to predict the start and end positions of the topic entity in a question. To reduce the model size, we proposed a joint model to share BERT encoder across all three sub-tasks, and create task-specific layers on the top. We then trained this joint model with multi-task learning. Experimental results show that our method achieves superior results on standard benchmarks, and is able to scale up to large-scale KBs.

References

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514, Osaka, Japan. The COLING 2016 Organizing Committee.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, Doha, Qatar. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433, Sofia, Bulgaria. Association for Computational Linguistics.

Niel Chah. 2017. Freebase-triples: A methodology for processing the freebase data dumps. *arXiv preprint arXiv:1712.08707*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2019a. Bidirectional attentive memory networks for question answering over knowledge bases. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2913–2923, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019b. UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 345–356, Minneapolis, Minnesota. Association for Computational Linguistics.

Zihang Dai, Lei Li, and Wei Xu. 2016. CFO: Conditional focused neural question answering with large-scale knowledge bases. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 800–810, Berlin, Germany. Association for Computational Linguistics.

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

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over Freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 260–269, Beijing, China. Association for Computational Linguistics.

Yanchao Hao, Hao Liu, Shizhu He, Kang Liu, and Jun Zhao. 2018. Pattern-revising enhanced simple question answering over knowledge bases. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3272–3282.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, Vancouver, Canada. Association for Computational Linguistics.

Xiaodong He and David Golub. 2016. Character-level question answering with attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1598–1607, Austin, Texas. Association for Computational Linguistics.

Kelvin Jiang, Dekun Wu, and Hui Jiang. 2019. Freebaseqa: A new factoid qa data set matching trivia-style question-answer pairs with freebase. In *Pro-*

ceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 318–323.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv preprint arXiv:1904.09482*.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194, Brussels, Belgium. Association for Computational Linguistics.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 291–296, New Orleans, Louisiana. Association for Computational Linguistics.

Rostislav Nedelchev, Debanjan Chaudhuri, Jens Lehmann, and Asja Fischer. 2020. End-to-end entity linking and disambiguation leveraging word and knowledge graph embeddings. *arXiv preprint arXiv:2002.11143*.

Michael Petrochuk and Luke Zettlemoyer. 2018. SimpleQuestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Yunqi Qiu, Manling Li, Yuanzhuo Wang, Yantao Jia, and Xiaolong Jin. 2018. Hierarchical type constrained topic entity detection for knowledge base question answering. In *Companion Proceedings of the The Web Conference 2018*, pages 35–36. International World Wide Web Conferences Steering Committee.

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

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. Introduction to information retrieval. In *Proceedings of the international communication of association for computing machinery conference*, page 260.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390, Hong Kong, China. Association for Computational Linguistics.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium. Association for Computational Linguistics.

Svitlana Vakulenko, Javier David Fernandez Garcia, Axel Polleres, Maarten de Rijke, and Michael Cochez. 2019. Message passing for complex question answering over knowledge graphs. In *CIKM*, pages 1431–1440. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5877–5881, Hong Kong, China. Association for Computational Linguistics.

Dekun Wu, Nana Nosirova, Hui Jiang, and Mingbin Xu. 2019. A general fofe-net framework for simple and effective question answering over knowledge bases. *arXiv preprint arXiv:1903.12356*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete KBs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4258–4264, Florence, Italy.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on Freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2326–2336, Berlin, Germany. Association for Computational Linguistics.

Xuchen Yao. 2015. Lean question answering over Freebase from scratch. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 66–70, Denver, Colorado. Association for Computational Linguistics.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1321–1331, Beijing, China. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1746–1756, Osaka, Japan.

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved neural relation detection for knowledge base question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 571–581, Vancouver, Canada. Association for Computational Linguistics.

Yang Yu, Kazi Saidul Hasan, Mo Yu, Wei Zhang, and Zhiguo Wang. 2018. Knowledge base relation detection via multi-view matching. In *European Conference on Advances in Databases and Information Systems*, pages 286–294. Springer.

Implicitly Abusive Comparisons – A New Dataset and Linguistic Analysis

Michael Wiegand
Digital Age Research Center (D!ARC)
Alpen-Adria-Universität Klagenfurt
AT-9020 Klagenfurt, Austria
michael.wiegand@aau.at

Maja Geulig
Institute of Computational Linguistics
Heidelberg University
D-69120 Heidelberg, Germany
geulig@cl.uni-heidelberg.de

Josef Ruppenhofer
Leibniz Institute for German Language
D-68161 Mannheim, Germany
ruppenhofer@ids-mannheim.de

Abstract

We examine the task of detecting implicitly abusive comparisons (e.g. *Your hair looks like you have been electrocuted*). Implicitly abusive comparisons are abusive comparisons in which abusive words (e.g. *dumbass* or *scum*) are absent. We detail the process of creating a novel dataset for this task via crowdsourcing that includes several measures to obtain a sufficiently representative and unbiased set of comparisons. We also present classification experiments that include a range of linguistic features that help us better understand the mechanisms underlying abusive comparisons.

1 Introduction

Abusive or offensive language is commonly defined as hurtful, derogatory or obscene utterances made by one person to another person.[1] Examples are (1)-(3).

(1) stop editing this, you dumbass.
(2) Just want to slap the stupid out of these bimbos!!!
(3) Go lick a pig you arab muslim piece of scum.

In the literature, closely related terms include *hate speech* (Waseem and Hovy, 2016) or *cyber bullying* (Zhong et al., 2016). While there may be nuanced differences in meaning, they are all compatible with the general definition above.

The definition we follow in this work also restricts abusive language to those utterances that are made to deliberately insult the target. A second requirement of an utterance to be considered abusive is that the target itself has to perceive the utterance as abusive.

Due to the rise of user-generated web content, the amount of abusive language is steadily growing. NLP methods are required to focus human review efforts towards the most relevant microposts.

Though there has been much work on abusive language detection in general, there is has been comparatively little work focusing on **implicit** forms of abusive language (4)-(5) (Waseem et al., 2017).

(4) I haven't had an intelligent conversation with a woman in my whole life.
(5) Why aren't there any Mexicans on Star Trek? Because they don't work in the future either.

By *implicit* we understand abusive language that is not conveyed by (unambiguously) abusive words (e.g. *dumbass*, *bimbo*, *scum*). Detailed analysis on the output of existing classifiers has also revealed that currently only explicit abuse can be reliably detected (Wiegand et al., 2019).

Given that implicit abuse is a challenging problem, we believe that the only reasonable approach to solve this problem is to address specific subtypes individually rather than consider all types of implicit abuse at once. In this paper, we examine *implicitly abusive comparisons*. A comparison is the act of evaluating two or more things by determining the relevant characteristics of each thing and to determine which characteristics of each are similar/different to the other (Bredin, 1998). By an abusive comparison, we understand a comparison that is perceived as abusive. In this work, we only consider those comparisons in which no (explicitly) abusive words are contained (6)-(8). Those comparisons are referred to as implicitly abusive comparisons. We exclude comparisons with abusive words since they can be easily detected with recent lexical resources for language abuse (Wiegand et al., 2018).

(6) You have the face of someone only a mother could love.
(7) Your hair looks like you have been electrocuted.
(8) You run like a headless chicken.

We address abusive comparisons since they make up a large proportion of comparisons on

---

[1] http://thelawdictionary.org/

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 358–368
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

the most related dataset by Qadir et al. (2015). That dataset was created to automatically detect the sentiment of a comparison: one has to distinguish between positive comparisons (*You look like a princess*), neutral comparisons (*You look like your brother*) and negative comparisons (*You look like a crackhead*). We manually annotated the negative comparisons of that dataset. We considered the 359 comparisons that focus on a person[2] (i.e. the 2nd person pronoun *you*). About 75% of the comparisons were considered abusive, the clear majority (2/3) is implicitly abusive. 25% of the negative person-focused comparisons were non-abusive (9)-(11), which is also a significant proportion.

(9) Your face is as pale as a sheet.
(10) You look like you haven't slept in days.
(11) Talking to you is like walking against a strong wind.

Unlike many other datasets for abusive language detection, we create our new dataset with abusive comparisons by ***inventing*** instances (i.e. comparisons) rather than by annotating automatically extracted instances. This design choice is necessary since existing datasets contain either insufficient or biased comparisons: The dataset from Qadir et al. (2015) includes about 180 implicitly abusive comparisons, however, we found, next to many near-duplicates, a heavy bias towards very few recurring images (e.g. *You behave like a child*, *You look like a monkey*). We observed the same phenomenon when extracting comparisons from Twitter directly. Established datasets for abusive language detection (Founta et al., 2018; Zampieri et al., 2019) contain just about 30-40 abusive comparisons.

Our dataset will be created via crowdsourcing. Of course, having abusive comparisons be invented this way will inevitably result in some articificial data. However, we think only thus can we produce a dataset of reasonable size that has also a very low degree of bias which are two important requirements to be able to do research on this novel research topic.

Having crowdworkers invent instances of abusive language may raise ethical concerns. However, the type of abusive language that will be invented in this work is not directed towards specific invididuals or identity groups. Therefore, we believe that this procedure is justifiable. In principle, creating morally disputable content as part of research is not unusual. Both in plagiarism detection (Potthast et al., 2010) and deception detection (Ott et al., 2011), a procedure similar to ours is pursued.

We frame our task as a **binary classification** problem in which each instance is to be categorized as either an abusive or some other negative comparison. Positive and neutral comparisons are not considered since Qadir et al. (2015) already proposed a polarity classifier for comparisons.

Our **contributions** in this paper are:

- We present the first study to address implicitly abusive comparisons.
- We create a **novel dataset** for this task. A set of measures is proposed in order to produce a representative and unbiased set of such comparisons. This dataset is made publicly available.[3]
- We provide an in-depth linguistic analysis that tries to uncover what types of phenomena are involved in abusive comparisons.
- We report classification experiments using state-of-the-art supervised classifiers.
- We provide empirical evidence that this task is different from previously examined tasks and that the established datasets for general language abuse are less suitable for this task.

2 Related Work

Datasets in abusive language detection mostly focus on different targets (e.g. Islamophobia (Waseem and Hovy, 2016), antisemitism (Warner and Hirschberg, 2012), misogyny (Anzovino et al., 2018)), different languages (e.g. Spanish (Álvarez-Carmona et al., 2018), Arabic (Mubarak et al., 2017), Portuguese (Fortuna et al., 2019)) or different domains (e.g. Twitter (Waseem and Hovy, 2016), Facebook (Kumar et al., 2018), Wikipedia (Wulczyn et al., 2017)). Despite some theoretical work outlining distinct subtypes of abusive language (Waseem et al., 2017), there has been little work on datasets that focus on particular subtypes. Schmidt and Wiegand (2017) present a more detailed overview on abusive language detection.

Comparisons, particularly figurative comparisons (*similes*), have been examined with regard to sentiment. Qadir et al. (2015) investigate automatic polarity classification of comparisons while

---

[2] Other foci are fairly unlikely to be abusive.

[3] The supplementary material, which consists of the supplementary notes and the new dataset, is available at: `https://github.com/miwieg/implicitly_abusive_comparisons`

Component	Example Sentence
Topic (T)	**[You]** are as smart as a toad.
Eventuality (E)	You **[are]** as smart as a toad.
Comparator (C)	You are **[as]** smart **[as]** a toad.
Property (P)	You are as **[smart]** as a toad.
Pattern (T+E+C+P)	**[You are as smart as]** a toad.
Vehicle	You are as smart as **[a toad]**.

Table 1: Components of a comparison.

Pattern ***(with example vehicle in brackets)***	**Perc.**
ABUSE	
You are as big as *(a whale.)*	100%
Your sense of humor reminds me of *(cold custard.)*	90%
You are as intelligent as *(a paper clip.)*	86%
You are as competent as *(an amoeba.)*	83%
You are as useful as *(a glass shovel.)*	78%
OTHER	
You are standing about as straight as *(a circle.)*	100%
You are as sad as *(a wilted lettuce.)*	100%
You are as organised as *(a pile of unsorted socks.)*	92%
You are as pale as *(a sheet.)*	92%
Your appetite is like *(that of a hungry bear.)*	86%

Table 2: Example of biased patterns.

Niculae and Danescu-Niculescu-Mizil (2014) establish a general correlation between sentiment and comparisons. Sentiment is also relevant for the general detection of abusive language (Brassard-Gourdeau and Khoury, 2019). Yet the focus on the subset of negative comparisons that are implicitly abusive has not been addressed before.

3 Data

3.1 Terminology

As illustrated in Table 1, a comparison is typically divided into a set of five components: *topic*, *eventuality*, *comparator*, *property* and *vehicle* (Hanks, 2013). In this paper, we combine the first four components into what we call a *pattern*.

3.2 Creating the Dataset

General setting. We decided to create our dataset with the help of **crowdsourcing**. For the invention of comparisons, we prefer a larger crowd to experts, since with a small set of experts (e.g. 2-3 persons) we would expect the resulting output to have a very limited lexical variability. However, since inventing abusive comparisons is not trivial (we have to make our crowdworkers familiar with specific linguistic concepts, such as negative polarity and implicit abuse), we decided to use *Prolific academic*.[4] This platform allows us to advertise our task to a subset of crowdworkers having specific qualifications. We advertised for English native speakers with some basic academic education without dyslexia. (*The supplementary notes provide annotation guidelines and more details on our setup on Prolific academic.*)

Exploratory Phase. We ran a set of trial surveys to get an idea how complex a single task may be while still allowing for the elicitation of data a reasonable quality. In this phase, we also had crowdworkers write abusive comparisons without any restriction. Thus we acquired a representative set of patterns (Table 1) used in subsequent tasks.

Creative Comparisons Task. In order not to overburden the crowdworkers with too complex instructions, we obtained abusive and non-abusive comparisons in separate tasks. Moreover, to devise a specific comparison, the crowdworkers were given a pattern, so that they only had to provide a vehicle (Table 1). By providing patterns, we could control the syntactic variability of the comparison. For both abusive and non-abusive comparisons, the same patterns were used. This setting allowed us to combine the output of these two surveys to one data collection. Otherwise, we may have ended up coincidentally with a dataset containing different syntactic constructions in the two classes which would artificially facilitate automatic classification.

For the non-abusive comparisons, it was also necessary to provide an example situation to the crowdworkers (45 situational frames were used in total – *see also supplementary notes*). For example, for the pattern *Your face is like*, we asked the crowdworkers to imagine the situation that they arrive at work and notice their colleague to have a severe cold. The comparison the crowdworkers were to devise should express their concern. In order not to overstrain their attention, each task was limited to up to 30 comparisons. Therfore, a larger pool of crowdworkers was required. Overall, 98 crowdworkers participated in creating our dataset.

Re-labelling Task. Since the annotation of abusive language is very difficult (Ross et al., 2016) and the generation of comparisons was (partially) tied to specific situational frames given to the crowdworkers, we introduced a re-labelling task in which all those comparisons were rated *in isolation* (i.e. without displaying the context of a specific situation) as either abusive and non-abusive. In this way, we chose to limit our final dataset to comparisons that can be classified in isolation. The motivation for this is that, while humans perceive the same texts as more or less offensive given con-

---

[4] `www.prolific.co`

text, modeling further context of abusive utterances was not found to improve classification using *currently* available methods, as shown by the recent in-depth study by Pavlopoulos et al. (2020).

Each comparison was rated by 5 crowdworkers; all crowdworkers were different from those in the previous step. Crowdworkers were allowed to label a comparison as *can't decide* for ambiguous comparisons or comparisons that were not understood out of context. For further processing, the label of the majority assigned by the workers was used.

Label Consistency Task. Since we noted several semantically similar comparisons across the collection (e.g. *Your posture reminds me of a weary marathon runner* and *You are as hungry as a marathon finisher*) to have different class labels, we introduced another task, in which sets of similar comparisons were presented to some additional crowdworkers without their current class labels. (The sets comprised between 2 and 4 comparisons.) These workers were to score the entire group but also indicate when they considered some individual comparisons to deviate from that group label. This procedure enabled us to remove several inconsistencies but at the same time also preserve different labels of semantically similar comparisons when they were actually appropriate.

Dataset Cleaning and Debiasing. In the final step, we cleaned the set of comparisons. We removed duplicate comparisons, comparisons that were cases of explicit abuse or that required some non-linguistic background knowledge (e.g. *Your reaction reminds me of how I felt*), and comparisons in which no majority label could be reached.[5]

Special attention was paid to the pattern distribution. We noticed that a subset of patterns is skewed towards abusive or non-abusive comparisons (as illustrated in Table 2). If we included those patterns, automatic classification would get substantially easier, as classifiers would simply learn the class distribution of patterns rather than *analyze* the complete comparison. Unwanted biases in datasets for abusive language detection is a signficant problem in current research (Arango et al., 2019; Wiegand et al., 2019). Therefore, we removed all comparisons belonging to patterns that had 65% or more instances belonging to one class. We also limited the remaining patterns to 20 instances in the dataset in order to avoid further possible topic biases caused by specific patterns dominating the dataset.

Property	**Value**
instances (i.e. comparisons)	1000
abusive instances (*ABUSE*)	500
non-abusive instances (*OTHER*)	500
(unique) crowdworkers	98
individual tasks for crowdsourcing	26
average token length of (full) comparison	9.35
average token length of vehicle	5.25
unique patterns	77
average amount of instances per pattern	12.99
total tokens (full comparison)	9351
total token types (full comparison)	1431
total tokens (only vehicle)	5248
total token types (only vehicle)	1391

Table 3: Statistics of the dataset.

We also measured interannotation agreement between the majority label of our crowdsourced comparisons and one co-author of this paper on a random sample of 200 random comparisons. Ignoring the cases in which the author was uncertain (12%), we reached an agreement of $\kappa = 0.6$ which can be considered substantial (Landis and Koch, 1977).

3.3 The Final Dataset

Our final dataset (Table 3), which comprises 1000 comparisons, only includes instances passing all data cleaning steps. Our examination of the simile dataset by Qadir et al. (2015) in §1 suggested a high proportion of abusive comparisons. However, it is unclear in how far these figures generalize beyond that dataset. Therefore, we enforced a balanced class distribution to be as unbiased as possible.

4 Linguistic Features

We present a set of linguistic features that we use for supervised classification. Some of them are too difficult to produce automatically. Yet we consider them relevant to this work because they may shed more light onto the nature of abusive comparisons. These particular features are produced manually.

4.1 Manually Designed Features

Figurativeness (FIGUR) vs. Literalness (LITERAL). Our dataset comprises both figurative (12) and literal comparisons (13).

(12) You sing like a dying bird. *(figurative & ABUSE)*
(13) You have the face of a sad person. *(literal & OTHER)*

In figurative comparisons, vehicle and topic are fundamentally different types of entities (Qadir et al., 2015).[6] Literal comparisons, on the other

---

[5] With 3 possible labels in the Re-labelling Task (*ABUSE*, *OTHER*, *CAN'T DECIDE*) and 5 workers, there could be ties.

[6] Note that in our paper, we consider figurative comparisons synonymous to what Qadir et al. (2015) refer as similes.

hand, are also reversible (Bredin, 1998). That is, the topic and vehicle of a literal comparison should be able to switch places without large changes in meaning. Hence *Encyclopedias are like dictionaries* can be rephrased as *Dictionaries are like encyclopedias*. However, this does not work for *Encyclopedias are like goldmines* which would therefore be judged as a figurative comparison. Moreover, literal comparisons must emphasize properties that are salient for both entities in the comparison (Wałaszewska, 2013). For instance, *dictionary* and *encyclopedia* share properties, such as being organized in a certain order or containing a number of entries. This does not hold for figurative comparisons. *Encyclopedia* and *goldmine* only share non-salient properties, such as being profitable.

One would intuitively expect abusive comparisons to be figurative (12) and non-abusive comparisons to be literal (13). Therefore, we need to answer the question whether abusive comparisons simply coincide with figurative comparisons.

Dehumanization (DEHUM). A dehumanizing comparison is defined as the direct comparison of a person or their inherent mental or physical attributes as the topic with a non-human entity as the vehicle (Loughnan et al., 2009). Dehumanization, in general, is known to correlate with abusive language (Mendelsohn et al., 2020). An example for a dehumanizing comparison would be (14) due to the comparison of a physical attribute of a person (i.e. *walk*) to a non-human entity (i.e. *giraffe*).

(14) You walk like a giraffe. *(ABUSE)*

Taboos (TABOO). Allen and Burridge (2006) define taboo as a proscription of behavior that affects everyday life. A characteristic of abusive language is that it is considered taboo in many social contexts and that it uses words associated with taboo topic to express offensiveness, such as specific bodily organs, physical and mental abnormity (15)-(17). Many of those words (e.g. *vagina*) are not included in common lexical resources for abusive language detection (Wiegand et al., 2018) and therefore are not considered as explicit abuse since they are too ambiguous. In medical contexts, for example, they are acceptable. Allen and Burridge (2006) provide a list of semantic fields, such as *death and disease* or *sex* which form the basis of our manual annotation of taboos. We assume a label set that reflects Western societies, which is the context in which our comparisons were created.

(15) Your eyes are like backwards binoculars. *(ABUSE)*
(16) You drive like an armless child. *(ABUSE)*
(17) Your attention span is like a flash on a circuit. *(ABUSE)*

Absurd Images (ABSURD). In many figurative comparisons in our dataset, we also observed fairly absurd images (18)-(19). By that we mean vehicles that describe situations that are never or extremely rarely observed in real life. We examine whether such images tend to be perceived as abusive.

(18) Your input is like [a baby giving their opinion on computer code]$_{vehicle}$. *(ABUSE)*
(19) You walk like [you have three legs and four pockets full of rubble]$_{vehicle}$. *(ABUSE)*

Contradiction (CONTRAD). A recurring construction in comparisons are contradictions (20)-(21). These are typically constructions where the property of the comparison (e.g. *thin*) is opposite to the prototypical properties associated with the vehicle (e.g. an *elephant* is *large* and *massive* rather than *thin*). Such contradictions, which are a subtype of sarcasm, may be perceived as abusive.

(20) You are as [thin]$_{prop.}$ as [an elephant]$_{vehicle}$. *(ABUSE)*
(21) You are as [smart]$_{prop.}$ as [a neanderthal]$_{vehicle}$. *(ABUSE)*

Evaluation (EVAL) vs. Emotional Frame of Mind (FRAME). Although all our comparisons are negative in polarity, they differ in the type of sentiment that they express. On the one hand, there are evaluative comparisons (22), i.e. the author of a comparison evaluates a specific property of the target person in a negative way, typically by criticizing their behavior or outward appearance (such as being overweight as in (22)). Such comparisons are likely to be perceived as abusive utterances. On the other hand, there are comparisons in which the author describes the emotional frame of mind of the target (23). Since all our comparisons are negative, typical emotional frames are *pain*, *sorrow*, *exhaustion* or *shock* (as in (23)). The author of such comparisons does not necessarily evaluate the target. For example, if one states that some other person is in pain, this is not meant as some criticism, but rather some concern. Such comparisons are rarely perceived as abusive.

(22) You look like an overfed cat. *(ABUSE)*
(23) You look like a shocked cat. *(OTHER)*

This distinction bears a resemblance to the distinction of sentiment views proposed by Wiegand et al. (2016). That work proposes a binary distinction into *speaker views*, which resembles evaluative

Feature	ABSURD	CONTRAD	DEHUM	FIGUR	TABOO	VIEW
Cohen's κ	0.82	1.00	0.88	0.63	0.74	0.73

Table 4: Agreement on manual features.

comparisons, and *actor views*, which resembles descriptions of the emotional frame of mind. Wiegand et al. (2016) also provide a list of verbs, nouns and adjectives classified into either of the two categories. Due to the fact that this lexicon seems inaccurate when it comes to ambiguous words[7], we annotated the binary distinction of evaluation vs. emotional frame of mind manually in addition to using the resource from Wiegand et al. (2016).

Interannotation Agreement. On a random sample of 200 comparisons, we measured interannotation agreement on each of the manually designed features between two annotators, one coauthor and a graduate in computational linguistics. The resulting scores are shown in Table 4.

The easiest feature are contradictions for which we even measured a perfect agreement on our sample. This very high agreement can be explained by the fact that many of the contradictions that have been employed in our dataset were cases of lexicalization, such as *as clear as mud*. Such contradictions are easy to spot.

It comes as no surprise that we obtain the lowest agreement for the distinction between literal and figurative language since this is a very difficult task (Pragglejaz Group, 2007). Still even that score is considered good for this particular task (Veale et al., 2016).

4.2 Automatically Generated Features

Intensity (INTENS). Wiegand et al. (2018) established a correlation between high polar intensity and abusive language (24). In order to measure the degree of polar intensity of a comparison, we took the most effective intensity lexicon from Wiegand et al. (2018) and ranked each comparison by the average intensity score associated with the words contained in the comparison. The lexicon ranks words from very positive (top) to very negative (bottom). Thus polar intensive words are at both ends of the ranking (e.g. top 50 or bottom 50).

(24) Your eyes are like doorways into hell itself. *(ABUSE)*

Frequency (RARE). A high polar intensity may not only be conveyed by inherently polar words (e.g. *hell*) but also by comparisons to special items. We assume that those items share the property of having a low frequency in a general text corpus. Words like *bakelite* or *kazoo* are no polar expressions but they are rare in text corpora. If used in a comparison (25)-(26), the comparison is perceived as an extreme comparison and therefore likely to be abusive. For our experiments, we estimate the frequency of words from the North American News Corpus (LDC95T21). We rank each comparison according to its most infrequent word.

(25) You are as modern as **bakelite**. *(ABUSE)*
(26) You laugh like a **kazoo**. *(ABUSE)*

Absence of Nouns and Adjectives (ABSENCE). Abusive imagery in comparisons typically requires concrete nouns as the vehicle. Further, abusive comparisons may require adjectives in order to convey a high polar intensity or a negative evaluation. This permits the reverse conclusion that the absence of those two parts of speech is likely to be a non-abusive comparison (27)-(28).

(27) You look like you're lost$_{verb}$. *(OTHER)*
(28) You move like you're hurt$_{verb}$. *(OTHER)*

Similarity to Explicit Insults (EXPLICIT). Although our comparisons do not contain any (explicitly) abusive words, a lexicon of such words may still help us in classification. We assume that the boundary between implicit and explicit insults is not clear-cut and that there are ambiguous abusive words contained in our comparisons that still have a strong semantic similarity to abusive words from a lexicon of abusive words. We took the lexicon from Wiegand et al. (2018), computed a centroid embedding vector of its entries and ranked our comparisons according to the semantic similarity to the centroid. A comparison was represented by the embedding vector of the word in that comparison whose similarity was highest to the centroid. As embeddings we chose the fastText emeddings (Joulin et al., 2017) induced on Common Crawl.[8]

Emotions (EMO). In order to take into account the recently reported correlation between abusive language and emotions (Rajamanickam et al., 2020), we use the NRC lexicon (Mohammad and Turney, 2013) which lists the emotion categories associated to a particular frequent English

---

[7] This lexicon assigns one category for each word thus assuming one sense per lexical entry. However, there may be words like *sick* which may convey an evaluation in one context (meaning *crazy* or *mad*) or a judgment on the frame of mind (feeling bad as a result of suffering from illness).

[8] `https://commoncrawl.org`

Classifier	**Prec**	**Rec**	**F1**
majority	25.0	50.0	33.3
random	50.9	51.0	51.0
fastText$_{plain}$	60.6	53.9	57.1
fastText$_{Common\ Crawl}$	68.0	67.5	67.8
BERT$_{only\ pattern}$	53.7	53.2	53.4
BERT$_{only\ vehicle}$	67.1	66.9	67.0
BERT	70.2	70.0	70.1^{*}
linguistic features$_{only\ auto}$	65.9	65.9	65.9
linguistic features	68.9	68.9	68.9^{*}
BERT+linguistic features$_{only\ auto}$	72.2	72.1	$72.2^{*\dagger}$
BERT+linguistic features	72.9	72.8	$72.9^{*\dagger}$
BERT+linguistic feat. on biased dataset	77.4	77.3	77.3
human baseline *(upper bound)*	77.6	77.5	$77.6^{*\dagger}$

statistical significance testing (paired t-test at $p < 0.05$): *: better than *fastText*$_{Common\ Crawl}$; †: better than *BERT*

Table 5: Comparisons of different classifiers.

word. A word may be associated with more than one of the 8 emotion categories. We represent each comparison by the set of emotion categories for the words also occurring in the NRC lexicon.

WordNet Supersenses (SUPER). We also consider WordNet supersenses (Miller et al., 1990) in our experiments. They represent a set of 45 coarse-grained semantic categories and have been found effective in related tasks, such as sentiment analysis (Flekova and Gurevych, 2016). A comparison is represented by the set of semantic categories associated with the words contained in the comparison.

5 Experiments

5.1 Classification Performance

As a supervised classifier, we chose BERT-Large (Devlin et al., 2019). We initially experimented with two versions: one in which we fine-tune the model by adding a layer on top of the pre-trained model and a SVM (Joachims, 1999) that is trained on the BERT embeddings of the final layer. Since we did not measure any statistically significant difference between these models, we decided in favor of SVM due to its simplicity. We carry out a 5-fold cross validation. The folds comprise mutually exclusive patterns (Table 1). Thus test instances always comprise patterns not observed in the training data. We consider this the most difficult and realistic scenario. We report macro-average precision, recall and F1-score. (*The supplementary notes contain more details regarding all classifiers of our experiments.*)

As baselines, we consider a majority classifier, a random classifier and two classifiers trained on fastText: one without and one with pre-trained embeddings (Common Crawl). As an upper bound we also provide a **human baseline** in which we randomly sampled the judgment of one individual annotator from the crowdsourced annotation. This upper bound may notably differ from the gold standard label since the latter benefited from being calculated from the majority of 5 annotators.

In order to demonstrate the importance of cleaning/debiasing the dataset and show that, otherwise classification performance will be unrealistically high, we also train a classifier on a **biased comparison dataset**. For that, we sampled a set of the identical size from the original data we collected via crowdsourcing (§3.2) as our final debiased dataset (i.e. 1000 comparisons) but skipped the data cleaning step, particularly the steps on balancing the pattern distribution and removing patterns that are highly skewed towards either of the two classes (Table 2). We also arranged the folds at random, so that patterns in the test data could also be observed in the training data. Thus a classifier could benefit from memorizing biased patterns.

Table 5 shows the performance of the different classifiers. FastText strongly benefits from the pre-trained embeddings and already outperforms the other baselines by a large degree. BERT outperforms fastText. If a classifier is just trained on the pattern, we obtain performance close to the random baseline which shows that our attemps to produce unbiased patterns were successful. Just training on a vehicle yields reasonable scores. However, this is outperformed by training on the full comparison (i.e. pattern+vehicle). This suggests that the full meaning of a comparison is conveyed by the combination of pattern and vehicle. The linguistic features also produce reasonable performance. However, a combination of BERT and these features results in best performance. The contribution of the manual linguistic features is only limited. The human baseline with an F1-score of 77.6% shows us again how difficult this task is. It also underscores the strong performance of our best classifier with an F1-score of 72.9%. The scores on the biased dataset are unrealistically high reaching performance close to the human upper bound.

Figure 1 shows a learning curve. It shows we already reached a plateau. Thus, it is unlikely to obtain better classification performance with more training data. Apparently, our dataset is already sufficiently representative of abusive comparisons.

Further, we investigate whether classifiers trained on standard datasets from abusive language detection can generalize to abusive comparisons.

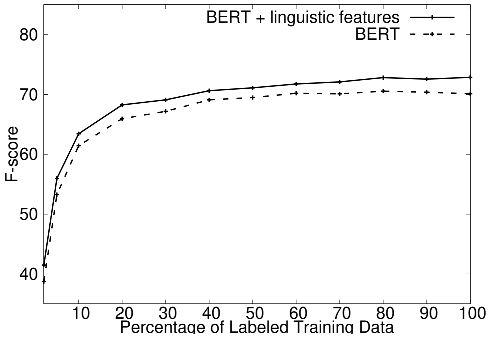


Figure 1: Learning curve on new comparison dataset.

Standard Datasets						New Dataset		
Founta			Zampieri			*(5-fold CV)*		
Prec	**Rec**	**F1**	**Prec**	**Rec**	**F1**	**Prec**	**Rec**	**F1**
63.9	63.5	63.7	66.0	66.1	66.0	72.9	72.8	72.9

Table 6: BERT when trained on different datasets and tested on the new comparison dataset.

We train a classifier (i.e. BERT) on each of the datasets from Founta et al. (2018) and Zampieri et al. (2019) (OffensEval) and test on our new comparison dataset.[9] We compare this against our best previous classifier (Table 5). Table 6 shows the results. The classifiers trained on previous standard datasets (although much larger) yield a considerably lower performance. They may only detect those abusive comparisons that are similar to the very few 30-40 comparisons that are contained on those datasets (§1) or that are generally fairly reminiscent of explicit abuse (22), which is the prominent type of abuse in the standard datasets.

5.2 Linguistic Analysis

Table 7 shows the precision of the most predictive features for each class. For both classes, manual and automatic features are predictive. By far the most predictive feature for implicitly abusive comparisons is the similarity to explicitly abusive words. Words in our dataset, such as *corpse* or *toad* may be too ambiguous to be included in a lexicon of abusive words but they have a strong similarity to abusive words. Further predictive features are rare words and taboo words.

The most predictive non-abusive feature is the supersense *noun.phenomenon*. This represents all comparisons to weather phenomena (e.g. *storms*, *eruption* etc. as in (11)). This is followed by the manual frame feature. Literal comparisons (10) correlate with non-abusive comparisons but there are many figurative comparisons that are also non-abusive (9) & (11). Therefore, the task of detecting abusive comparisons cannot be reduced to the distinction between literal and figurative language.

ABUSE **(Random Precision: 50.0)**			
Feature	**Manual?**	**Precision**	**Freq**
EXPLICIT_top50		92.0	50
EXPLICIT_top100		82.0	100
RARE_top50		76.0	50
TABOO	✓	74.3	113
EMO_disgust		71.3	115
EXPLICIT_top200		70.5	200
SUPER_noun.animal		68.8	122
CONTRAD	✓	68.8	32
SUPER_noun.food		66.1	62
EXPLICIT_top300		66.0	300
ABSURD	✓	62.7	51
SUPER_noun.body		62.5	32
EMO_negative		60.9	253
OTHER **(Random Precision: 50.0)**			
Feature	**Manual?**	**Precision**	**Freq**
SUPER_noun.phenomenon		80.6	36
FRAME_manual	✓	79.1	86
LITERAL	✓	71.6	109
SUPER_noun.object		71.2	59
SUPER_noun.event		70.7	41
INTENS_top200		68.5	200
INTENS_top300		68.3	300
SUPER_noun.time		67.7	65
INTENS_top50		66.0	200
INTENS_top100		65.0	300
ABSENCE		64.6	127
EMO_trust		64.1	131
SUPER_verb.motion		63.5	52
SUPER_noun.act		62.9	70
INTENS_top500		62.6	500
SUPER_verb.perception		61.5	39
FRAME_auto		61.4	140
SUPER_verb.stative		60.7	107

Table 7: Correlation between features and classes.

From the emotion categories, *disgust* correlates with abusive comparisons, while *trust* correlates with non-abusive instances. Of the manual features dehumanization (Prec: 53.9%) does not correlate well with abusive comparisons. Manual features, such as contradictions and absurd images, are more effective. Still, they only cover comparably few instances (i.e. 32 and 51) in our dataset.

The predictive supersenses give us further insights into the nature of abusive comparisons. Animal, food and body expressions are typical of abusive comparisons while weather phenomena, events, temporal expressions and acts are more likely to indicate non-abusive comparisons. Given that also verbal categories (motion, perception, stative) co-occur with the latter class, we can conclude that comparisons addressing abstract concepts tend to be non-abusive while comparisons involving concrete entities (e.g. animals, food) tend to be abusive.

---

[9] Our linguistic features were not added to BERT since they are tailored to comparisons which are extremely rare on those datasets (§1).

5.3 Error Analysis

In the output of the best classifier, we observed the following regularities in misclassifications:

On the one hand, the classifier often overgeneralized. For instance, the classifier learned that a comparison to a machine is typically a means of making fun of someone as in *talking/moving/dancing like a robot*. However, it fails to detect that in (29), the intention is different. Here, the speaker is concerned about seeing someone trembling intensely.

(29) Your hands are like a shaky washing machine. *(OTHER)*

Moreover, certain lexicalized multi-word expressions are not properly recognized. For example, in (30) the mention of the animal *pig* is part of the idiom *see a pig fly* which is not abusive. The classifier probably only learned that *pig* is abusive and predicts the comparison (30) erroneously as abuse.

(30) You talk like you've just seen a pig fly. *(OTHER)*

Comparisons with no obviously predictive linguistic cue (e.g. (31)) also remain difficult.

(31) You talk like someone who is just learning to read. *(ABUSE)*

6 Conclusion

We examined the novel task of detecting implicitly abusive comparisons. For this task, a new dataset was created via crowdsourcing. The comparisons were invented by the crowdworkers themselves. We identified linguistic features that correlate with abusive comparisons (rare words, concrete concepts, contradictions, absurd images and words associated with disgust) and non-abusive comparisons (literal language, comparisons expressing the emotional frame of the target, abstract concepts and the absence of adjectives and nouns). We examined various supervised classifiers. The best classifier is a combination of BERT and the above linguistic features. We also found that abusive comparisons cannot be equated with previously examined phenomena, such as (negative) figurative comparisons. The best classifier trained on the new dataset outperforms classifiers trained on existing datasets that contain hardly any abusive comparisons thus underscoring the need for our new specialized dataset.

Acknowledgements

This research has been partially supported by the Leibniz ScienceCampus Empirical Linguistics and Computational Modeling, funded by the Leibniz Association under grant no. SAS-2015-IDS-LWC and by the Ministry of Science, Research, and Art (MWK) of the state of Baden-Württemberg. The authors would like to thank Ashequl Qadir, Ellen Riloff and Marilyn A. Walker for providing access to their dataset for recognizing affective polarity in similes. We are also grateful to Elisabeth Eder for feedback on earlier drafts of this paper.

References

Keith Allen and Kate Burridge. 2006. *Forbidden Words: Taboo and the Censoring of Language*. Cambridge University Press.

Miguel A. Álvarez-Carmona, Estefanía Guzmán-Falcón, Manuel Montes y Gómez, Hugo Jair Escalante, Luis Villaseñor-Pineda, Verónica Reyes-Meza, and Antonio Rico-Sulayes. 2018. Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets. In *Proceedings of the Evaluation of Human Language Technologies for Iberian Languages Workshop (IberEval)*, Sevilla, Spain.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *Proceedings of the International Conference on Applications of Natural Language Processing to Information Systems (NLDB)*, pages 57–64, Paris, France. Springer.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate Speech Detection is Not as Easy as You May Think: A Closer Look at Model Validation. In *Proceedings of the ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 45–53, Paris, France.

Eloi Brassard-Gourdeau and Richard Khoury. 2019. Subversive Toxicity Detection using Sentiment Information. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, pages 1–10, Florence, Italy.

Hugh Bredin. 1998. Comparisons and similes. *Lingua*, 105(1–2):67–78.

Jacob Devlin, Ming-Wei Chang, KentonLee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 4171–4186, Minneapolis, MN, USA.

Lucie Flekova and Iryna Gurevych. 2016. Supersense Embeddings: A Unified Model for Supersense Interpretation, Prediction, Utilization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2029–2041, Berlin, Germany.

Paula Fortuna, João Rocha da Silva, Juan Soler-Company, Leo Wanner, and Sérgio Nunes. 2019. A Hierarchically-Labeled Portuguese Hate Speech Dataset. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, pages 94–104, Florence, Italy.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behaviour. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM)*, Stanford, CA, USA.

Patrick Hanks. 2013. *Lexical Anaylsis:Norms and Exploitations*. The MIT Press.

Thorsten Joachims. 1999. Making Large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 427–431, Valencia, Spain.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking Aggression Identification in Social Media. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC)*, pages 1–11, Santa Fe, NM, USA.

J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Steve Loughnan, Nick Haslam, and Yoshihisa Kashima. 2009. Understanding the Relationship between Attribute-based and Metaphor-based Dehumanizations. *Group Processes and Intergroup Relations*, 12(6):747–762.

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A Framework for the Computational Linguistic Analysis of Dehumanization. *Frontiers in Artificial Intelligence*, 3(55).

George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3:235–244.

Saif Mohammad and Peter Turney. 2013. Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 39(3):555–590.

Hamdy Mubarak, Kareem Darwish, and Walid Magdy. 2017. Abusive Language Detection on Arabic Social Media. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, pages 52–56, Vancouver, BC, Canada.

Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2014. Brighter than gold: Figurative language in user generated comparisons. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2008–2018, Doha, Qatar.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding Deceptive Opinion Spam by Any Stretch of the Imagination. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 309–319, Portland, OR, USA.

John Pavlopoulos, Jeffrey Sorensen, Lucas Dixon, Nithum Thain, and Ion Androutsopoulos. 2020. Toxity Detection: Does Context Realler Matter? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4296–4305, Online.

Martin Potthast, Benno Stein, Alberto Barrón-Cedeño, and Paolo Rosso. 2010. An Evaluation Framework for Plagiarism Detection. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 997–1005, Beijing, China.

Pragglejaz Group. 2007. MIP: A Method for Identifying Metaphorically Used Words in Discourse. *Metaphor and Symbol*, 22(1):1–39.

Ashequl Qadir, Ellen Riloff, and Marilyn A. Walker. 2015. Learning to Recognize Affective Polarity in Similes. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 190–200, Lisbon, Portugal.

Santhosh Rajamanickam, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020. Joint Modelling of Emotion and Abusive Language Detection. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4270–4279, Online.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2016. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication*, pages 6–9, Bochum, Germany.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the EACL-Workshop on Natural Language Processing for Social Media (SocialNLP)*, pages 1–10, Valencia, Spain.

Tony Veale, Beata Beigman Klebanov, and Ekaterina Shutova. 2016. *Metaphor: A Computational Perspective*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Ewa Wałaszewska. 2013. Like in Similes: A Relevance-theoretic View. *Research in Language*, 11(3):323–334.

William Warner and Julia Hirschberg. 2012. Detecting Hate Speech on the World Wide Web. In *Proceedings of the Workshop on Language in Social Media (LSM)*, pages 19–26, Montréal, Canada.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the ACL-Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada.

Zeerak Waseem and Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL – Student Research Workshop*, pages 88–93, San Diego, CA, USA.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 602–608, Minneapolis, MN, USA.

Michael Wiegand, Josef Ruppenhofer, Anna Schmidt, and Clayton Greenberg. 2018. Inducing a Lexicon of Abusive Words – A Feature-Based Approach. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 1046–1056, New Orleans, LA, USA.

Michael Wiegand, Marc Schulder, and Josef Ruppenhofer. 2016. Separating Actor-View from Speaker-View Opinion Expressions using Linguistic Features. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 778–788, San Diego, CA, USA.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1391–1399, Perth, Australia.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 1415–1420, Minneapolis, MN, USA.

Haoti Zhong, Hao Li, Anna Cinzia Squicciarini, Sarah Michele Rajtmajer, Christopher Griffin, David J. Miller, and Cornelia Caragea. 2016. Content-Driven Detection of Cyberbullying on the Instagram Social Network. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3952–3958, New York City, NY, USA.

Exploiting Emojis for Abusive Language Detection

Michael Wiegand
Digital Age Research Center (D!ARC)
Alpen-Adria-Universität Klagenfurt
AT-9020 Klagenfurt, Austria
`michael.wiegand@aau.at`

Josef Ruppenhofer
Leibniz Institute for German Language
D-68161 Mannheim, Germany
`ruppenhofer@ids-mannheim.de`

Abstract

We propose to use abusive emojis, such as the *middle finger* or *face vomiting*, as a proxy for learning a lexicon of abusive words. Since it represents extralinguistic information, a single emoji can co-occur with different forms of explicitly abusive utterances. We show that our approach generates a lexicon that offers the same performance in cross-domain classification of abusive microposts as the most advanced lexicon induction method. Such an approach, in contrast, is dependent on manually annotated seed words and expensive lexical resources for bootstrapping (e.g. WordNet). We demonstrate that the same emojis can also be effectively used in languages other than English. Finally, we also show that emojis can be exploited for classifying mentions of ambiguous words, such as *fuck* and *bitch*, into generally abusive and just profane usages.

1 Introduction

Abusive or offensive language is defined as hurtful, derogatory or obscene utterances made by one person to another.[1] In the literature, closely related terms include *hate speech* (Waseem and Hovy, 2016) or *cyber bullying* (Zhong et al., 2016). While there may be nuanced differences in meaning, they are all compatible with the general definition above.

Due to the rise of user-generated web content, the amount of abusive language is also steadily growing. NLP methods are required to focus human review efforts towards the most relevant microposts. Building classifiers for abusive language detection requires expensive manually labeled data.

In this paper we explore distant supervision (Mintz et al., 2009) for abusive language detection in which abusive emojis serve as a heuristic to identify abusive language (1)-(8). These texts are subsequently used as training data. The advantage of emojis is that some of them are unambiguously abusive. They are also often redundant (Donato and Paggio, 2017), i.e. they convey something already expressed verbally in the micropost. Since the concept conveyed by an emoji can be expressived verbally in many different ways, abusive emojis may co-occur with many different abusive words (e.g. *idiot*, *cunt*). Moreover, the meaning of emojis is (mostly) shared across languages.

(1) You are such a hypocrite ... Have your dinner dick 😠
(2) @USER @USER you need a good old fashion man sized ass kicking you little Twitt 👊
(3) @USER I challenge you to go on a diet you fat cunt 🖕
(4) @USER You are so so stupid you monkey face 🐵
(5) Send your location, I'll send some killers 🔫
(6) @USER @USER A vote for toddstone or any liberal. Id rather flush a toilet. 💩
(7) Fuck the 12 fuck the cops we aint forgot about you, kill em all kill em all ☠️
(8) @USER She is such a disgusting despicable human being! Ugh! 🤮

Recently, there has been significant criticism of in-domain supervised classification in abusive language detection, whose evaluation has been shown to produce overly optimistic classification scores. They are the result of biases in the underlying datasets. Wiegand et al. (2019) show that on the most popular dataset for this task (Waseem and Hovy, 2016), classifiers learn co-incidental correlations between specific words (e.g. *football* or *sport*) and the abusive class label. Such spurious correlations help classifiers to correctly classify difficult microposts on that particular dataset. Arango et al. (2019) show that since on the dataset from Waseem and Hovy (2016) the majority of abusive tweets originate from just 2 authors, classifiers learn the authors' writing style rather than abusive language.

In order to avoid an evaluation affected by such *topic* or *author biases*, we focus on learning a lexicon of abusive language. A lexicon-based approach

---

[1] http://thelawdictionary.org

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 369–380
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

to detect abusive language primarily focuses on the detection of *explicitly* abusive language, i.e. abusive language that is conveyed by abusive words. Such an approach is currently the most effective clue known for cross-domain classification (Wiegand et al., 2018a). In general, other types of abusive language that are more implicit, such as sarcasm, jokes or stereotypes, require more contextual interpretation of words. Supervised classification is theoretically able to conduct such contextual interpretation. However, it has been reported to perform very poorly (Karan and Šnajder, 2018; Arango et al., 2019) on this task because the biases these classifiers exploit are unlikely to be present across different datasets (Wiegand et al., 2019). Therefore, we focus on explicitly abusive language in this work, since there are no ways of reliably detecting implicitly abusive language.

Despite the existence of lexicons for abusive words, induction methods are required, since new abusive words enter language constantly. Further, there are only few lexicons available in languages other than English. The aim of our work is not to detect completely new types of abusive language but to find an inexpensive and language-independent method for lexicon induction.

Our **contributions** in this paper are:

- We use emojis to induce a lexicon of abusive words. Unlike previous work, such an approach does not depend on manually labeled training data or expensive resources, such as WordNet or intensity lexicons. We also demonstrate its effectiveness on cross-domain classification of microposts.
- In order to show the general applicability of our approach, we apply it not only to English but also to Portuguese and German data. The output of this study are three state-of-the-art lexicons that we make publicly available along with all other resources created in this paper.
- We use emojis to disambiguate the context of potentially abusive words. We exemplify this on the two ambiguous and frequent words *fuck* and *bitch*. A by-product is a dataset of mentions of these words annotated in context.

The supplementary material[2] to this paper includes all resources newly created for our research and notes on implementation details.

2 Related Work

Abusive language detection is mostly framed as a supervised learning task (Schmidt and Wiegand, 2017). Feature-based (Nobata et al., 2016) and neural (Pavlopoulos et al., 2017) methods are applied.

Lexicon induction for abusive language detection has received only little attention in previous work, the exceptions being Razavi et al. (2010) who present a lexicon generated using adaptive learning, Gitari et al. (2015) who bootstrap hate verbs and Wiegand et al. (2018a) who induce a lexicon of abusive words. This lexicon is currently the best performing lexicon for the task. It has been induced with the help of a (seed) base lexicon which had been manually annotated. The bootstrapping step largely relies on resources that exist only for well-resourced languages, such as WordNet, sentiment intensity datasets or sentiment-view lexicons.

Recently, there has been a general interest in exploiting extralinguistic information for natural language processing. Emoticons, such as *:-)*, have been found useful for sentiment analysis, particularly emotion classification (Purver and Battersby, 2012). Emojis represent an even more fine-grained set of icons. Felbo et al. (2017) exploit them for pretraining neural models to produce a text representation of emotional content. Since this approach relies on a representative sample of tweets containing emojis, only the 64 most frequently occurring emojis are considered. This set, however, does not contain the very predictive emojis for abusive language detection (e.g. middle finger). Corazza et al. (2020) follow an approach similar to Felbo et al. (2017) in that they pretrain a language model with the help of emoji informarion. However, unlike Felbo et al. (2017), their emoji-based masked language model is evaluated for zero-shot abusive language detection. The task is also considered in a multilingual setting: the target languages are English, German, Italian and Spanish. The improvements that Corazza et al. (2020) report over baseline language models that do not explicitly incorporate emoji information are only limited.

Our work extends Felbo et al. (2017) and Corazza et al. (2020) in that we focus on predictive emojis for abusive languag detection. Unlike Felbo et al. (2017) and Corazza et al. (2020), we do not pretrain a text classifier with these additional emojis. Supervised text classifiers are known to severely suffer from domain mismatches in abusive language detection whereas lexicon-based classifi-

---

[2] https://github.com/miwieg/emojis_for_abusive_language_detection

emoji	angry face 😠	oncoming fist 👊	middle finger 🖕	monkey face 🐵	pistol 🔫	pile of poo 💩	skull & crossbones ☠️	face vomiting 🤮
English tweets	31078	5835	4973	20533	957	5654	40852	20137
Portuguese tweets	15092	1178	1540	2779	122	857	711	6853
German tweets	1367	76	132	2161	46	146	341	676

Table 1: Emojis examined for the task and the number of tweets containing them obtained after 1 day.

cation is much more stable (Wiegand et al., 2018a).

3 Data, Vocabulary, Tasks & BERT

Data. We use Twitter as a corpus since it is known to contain a significant number of emojis and abusive language. Despite the variety of different emojis[3], only a smaller fraction is regularly used on Twitter. For instance, the dataset from Zampieri et al. (2019) includes less than 10% of them. Our final choice of emojis is displayed in Table 1. It is based on correlations between concepts and abusive language reported in the literature. Next to the emoji 🖕 (middle finger) depicting the most universally offensive gesture (Robbins, 2008), our choice includes emojis that connote violence (Wiener, 1999) (👊 oncoming fist, 🔫 pistol), the taboo topics death and defecation (Allen and Burridge, 2006) (☠️ skull and crossbones, 💩 pile of poo), the emotions anger and disgust (Alorainy et al., 2018) (😠 angry face, 🤮 face vomiting) and dehumanization (Mendelsohn et al., 2020) (🐵 monkey face). (1)-(8) illustrate each emoji with an abusive tweet.

For further emojis we only obtained an insufficient amount of English tweets that were necessary for our experiments (i.e. several thousand tweets after running a query containing these emojis using the Twitter-streaming API for a few days). Examples of such sparse emojis are 💣 (bomb) connoting violence or ⚡ (high voltage) connoting anger.[4]

Although our procedure involved a manual selection of emojis, in our evaluation we will demonstrate that this choice does not overfit but generalizes across different datasets and languages.

Table 1 also shows that for Portuguese and German we obtained fewer tweets. This sparsity is representative for languages other than English.

Vocabulary. Our induction experiments are carried out on a vocabulary of negative polar expressions. Abusive words form a proper subset of these expressions. We use the set of negative polar expressions from Wiegand et al. (2018a) comprising about 7,000 English words. For our experiments on Portuguese and German data, we created similar word lists following Wiegand et al. (2018a).

Tasks. In this work, there are two types of tasks: lexicon induction tasks in which we *rank* negative polar expressions where the high ranks should be abusive words, and *classification* of abusive microposts. The former is evaluated with precision at rank n (P@n), while the latter is evaluated with accuracy and macro-average F-score.

Supervised Micropost Classification with BERT. In many experiments, we employ *BERT-LARGE* (Devlin et al., 2019) as a baseline for state-of-the-art text classification for detecting abusive microposts. We always fine-tune the pretrained model by adding another layer on top of it. (*The supplementary notes contain more details regarding all classifiers employed in this paper.*)

4 Inducing a Lexicon of Abusive Words

4.1 Methods for Lexicon Induction

Pointwise Mutual Information (PMI). A standard method for inducing a lexicon from labeled documents is to rank the words according to the PMI with the target class (Turney, 2002). We use tweets in which either of the above emojis occur as abusive documents. In order to obtain negative instances, i.e. tweets which convey no abusive language, we simply sample random tweets from Twitter. The rationale is that abusive language is known to be rare, even on Twitter. Founta et al. (2018) estimate that the proportion of abusive tweets is less than 5%. In order to avoid spurious word correlations, we compute PMI only for words in our vocabulary of negative polar expressions (§3) which occur at least 3 times in our tweets. This threshold value was proposed by Manning and Schütze (1999).

Projection-based Induction. In our second method, we learn a projection of embeddings. The tweets are labeled in the same way as they are labeled for PMI. We use the pretrained embeddings from GloVe (Pennington et al., 2014) induced from

---

[3] https://unicode.org/emoji/charts/full-emoji-list.html

[4] https://icon-library.com/icon/anger-icon-14.html

Twitter.[5] Projection-based induction has the advantage over PMI that it does not only rank words observed in the labeled tweets but all words represented by embeddings. Since the GloVe embeddings are induced on a very large set of tweets which is about 10,000 times larger than the set of tweets we will later use for projection-based induction, i.e. 100k tweets per class (Table 4), the projection is likely to cover a larger vocabulary than PMI including additional abusive words. Let $\mathbf{M} = [\mathbf{w}_1, \ldots, \mathbf{w}_n]$ denote a labeled tweet of n words. Each column $\mathbf{w} \in \{0,1\}^v$ of $\mathbf{M}$ represents a word in a one-hot form. Our aim is learning a one-dimensional projection $\mathbf{S} \cdot \mathbf{E}$ where $\mathbf{E} \in \mathbb{R}^{e \times v}$ represents our unsupervised embeddings of dimensionality e over the vocabulary size v and $\mathbf{S} \in \mathbb{R}^{1 \times e}$ represents the learnt projection matrix. We compute a projected tweet $\mathbf{h} = \mathbf{S} \cdot \mathbf{E} \cdot \mathbf{M}$ which is an n-dimensional vector. Each component represents a word from the tweet. The value represents the predictability of the word towards being abusive. We then apply a bag-of-words assumption to use that projected tweet to predict the binary class label y: $p(y|\mathbf{M}) \propto exp(\mathbf{h} \cdot \mathbf{1})$ where $\mathbf{1} \in \{1\}^n$. This model is a feed-forward network trained using Stochastic Gradient Descent (Rumelhart et al., 1986). On the basis of the projected embeddings we rank the negative polar expressions from our vocabulary (§3).

Recall-based Expansion by Label Propagation (LP). While the very high ranks of an induction method typically coincide with the target class (in our case: abusive words), the lower a rank is, the more likely we are to encounter other words. Taking the high ranks as abusive seeds and then applying some form of label propagation on a word-similarity graph may increase the overall coverage of abusive words found. More specifically, we apply the Adsorption label propagation algorithm from *junto* (Talukdar et al., 2008) on a word-similarity graph where the words of our vocabulary are nodes and edges encode cosine-similarities of their embeddings. As negative (i.e. non-abusive) seeds, we take the most frequently occurring words from our vocabulary since they are unlikely to represent abusive words. In order to produce a meaningful comparison to PMI and projection-based induction, we need to convert the categorical output of label propagation to a ranking of our entire vocabulary. We achieve this by ranking the words predicted to be abusive by their confidence score. At the bottom we concatenate the words predicted to be non-abusive by their inverted confidence score.

4.2 Experiments on English

Evaluation of Induction. The first question we want to answer is what emoji is most predictive. For each of our pre-selected emojis (Table 1), we sampled 10k tweets in which it occurs and ranked the words of our vocabulary according to PMI. As non-abusive tweets we considered 10k randomly sampled tweets. As a baseline, we randomly rank words (*random*). As a gold standard against which we evaluate our rankings, we use all words of the lexicon from Wiegand et al. (2018a) predicted as abusive. Table 2 shows the results of the evaluation against this gold standard. The table shows that 🖕 (middle finger) is the strongest emoji. This does not come as a surprise as the middle finger is universally regarded as a deeply offensive gesture. We use this emoji as a proxy for abusive language in all subsequent experiments where possible.

In Table 3, we examine for PMI and our projection-based approach whether the ranking quality can be improved when more tweets are used. We increased the number of tweets containing the emoji and the number of negative tweets to 100k each. Using the free Twitter-streaming API larger amounts cannot be crawled in a reasonable time span (e.g. 1 month). While for *projection*, we reach maximum performance at 10k tweets, *PMI* is dependent on more data since it can only rank words it has actually observed in the data. *projection* clearly outperforms *PMI*. Since we do not want to overfit and show that our approach is not dependent on the exact value of 10k but also works with any amount of tweets beyond 10k, we use 100k tweets (i.e. the largest amount of tweets available to us) in subsequent experiments.

Table 4 compares further methods. Our gold standard has a wide notion of abusive language, including words such as *crap* or *shit*, which may be merely profane, not truly abusive. Such words also occur in the random tweets that serve as negative data. (Recall that profanity is much more common on Twitter.) These words are thus not learned as abusive. We therefore replaced our negative data with a random sample of sentences from the English *Web as Corpus (ukwac)*. While we thus preserve the language register with this corpus, i.e. informal language, profane language should be-

---

[5] We take the version with 200 dimensions which is a very frequently used configuration for word embeddings.

P@n	random	👊	🔫	☠️	🐵	🤮	😠	💩	🖕
10	20.0	10.0	30.0	20.0	40.0	60.0	60.0	**70.0**	40.0
50	20.0	24.0	28.0	32.0	40.0	36.0	48.0	48.0	**58.0**
100	17.0	24.0	31.0	28.0	31.0	32.0	36.0	44.0	**56.0**
200	21.0	24.0	27.0	29.0	23.0	39.5	37.0	43.0	**48.0**

Table 2: Precision at rank n (P@n) of different emojis (Table 1); ranking is based on PMI with 10,000 tweets.

	amount of tweets											
	PMI						**projection**					
P@n	**0.5k**	**1k**	**5k**	**10k**	**50k**	**100k**	**0.5k**	**1k**	**5k**	**10k**	**50k**	**100k**
10	60.0	50.0	70.0	40.0	80.0	80.0	70.0	50.0	70.0	70.0	**90.0**	80.0
50	30.0	42.0	58.0	58.0	66.0	68.0	62.0	68.0	**72.0**	**72.0**	70.0	70.0
100	26.0	36.0	50.0	56.0	65.0	65.0	60.0	67.0	**73.0**	70.0	71.0	70.0
200	25.5	35.0	40.5	48.0	58.5	61.0	56.5	62.5	**63.5**	62.5	61.0	60.5
500	*N/A*	27.0	34.4	40.8	46.8	51.6	46.5	50.0	**53.2**	52.4	50.0	51.8
1000	*N/A*	*N/A*	30.5	34.3	37.3	40.3	40.4	41.8	43.4	**46.1**	42.6	43.9
2000	*N/A*	*N/A*	*N/A*	*N/A*	30.6	32.9	33.9	34.9	36.2	**36.6**	35.6	36.4

Table 3: Comparison of PMI and projection using emoji *middle finger* with different amounts of tweets.

come exclusive to our proxy of abusive tweets. Table 4 confirms that using *ukwac* as negative data ($projection_{ukwac}$) improves performance.

To increase the recall of abusive words, we apply *LP* (§4.1) to the output of $projection_{ukwac}$. Since label propagation is sensitive to the underlying class distribution and abusive words typically represent the minority class, we use twice as many non-abusive seeds as abusive seeds.[6] We vary the amount of abusive seeds between 100, 200 and 500. To ensure comparability to the remaining configurations, the seeds are prepended to the output of LP (which explains that LP has only an impact on lower ranks). Table 4 shows clearly that *LP* outperforms $projection_{ukwac}$ on lower ranks.

Cross-Domain Evaluation. Next, we test the best lexicon of our previous experiments (i.e. $projection_{ukwac}$*+LP(200 abusive seed words)*) in cross-domain micropost classification. Posts are categorized into abusive and non-abusive posts. Through a cross-domain classification, in which we train on one dataset and test on another, we show that the chosen configuration is not overfit to a particular dataset.

Table 5 provides some information on the datasets we consider. In addition to the datasets used in Wiegand et al. (2018a), we include the recent SemEval-dataset from Zampieri et al. (2019).

Table 6 shows the results of cross-domain micropost classification. As baselines we use a majority-class classifier, the feature-based approach from Nobata et al. (2016), BERT and the lexicon from Wiegand et al. (2018a). In order to demonstrate the intrinsic predictiveness of the words learned by our emoji-based approach, we do not train a classifier on the source domain (unlike Wiegand et al. (2018a) who use the rank of the lexicon entries as a feature) but simply classify a micropost as abusive if an abusive word from our emoji-based lexicon is found. As abusive words, we consider all 1,250 words of our best approach (Table 4) predicted as *abusive*. Since the training data are not used for our emoji-based approach, that approach produces always the same result on each test set.

Table 6 shows that our lexicon performs on a par with the induction method from Wiegand et al. (2018a), on some domains (e.g. *Warner*), it is even better. Our observation is that these slight performance increases can be ascribed to the fact that our lexicon is only half of the size of the lexicon from Wiegand et al. (2018a). That lexicon still contains many ambiguous words (e.g. *blind* or *irritant*) that are not included in our emoji-based lexicon. Notice that our aim was not to outperform that method. The underlying lexicon was bootstrapped using manual annotation and the induction depends on external resources, such as WordNet or sentiment intensity resources. Our emoji-based approach is a much cheaper solution that can also be applied to languages where these resources are lacking.

4.3 Crosslingual Experiments

In order to show that our approach is also useful for languages other than English, we now apply it to Portuguese and German data.

Necessary Modifications. Given that there are much fewer Portuguese and German than English tweets (Table 1), it is more difficult to obtain a sim-

---

[6] We refrain from tuning the ratio in order to improve the result of *LP* since we want to avoid overfitting.

classifier	P@n 10	50	100	200	500	1000	1500	2000
PMI	80.0	68.0	65.0	61.0	51.6	40.3	35.0	32.9
projection	80.0	70.0	70.0	60.5	51.8	43.9	39.3	36.4
projection$_{ukwac}$	**100.0**	**78.0**	**72.0**	66.0	55.4	46.0	40.6	37.7
projection$_{ukwac}$+LP(100 abusive seed words)	**100.0**	**78.0**	**72.0**	**70.8**	**64.8**	57.5	39.2	31.3
projection$_{ukwac}$+LP(200 abusive seed words)	**100.0**	**78.0**	**72.0**	66.0	63.6	60.7	**57.9**	**49.9**
projection$_{ukwac}$+LP(500 abusive seed words)	**100.0**	**78.0**	**72.0**	66.0	55.4	**63.0**	55.3	41.7

Table 4: Comparison of different ranking methods using 100k tweets containing emoji *middle finger*.

dataset	size†	abusive	source
(Warner and Hirschberg, 2012)	3438	14.3%	*diverse*
(Waseem and Hovy, 2016)	16165	35.3%	Twitter
(Razavi et al., 2010)	1525	31.9%	UseNet
(Wulczyn et al., 2017)	115643	11.6%	Wikipedia
(Zampieri et al., 2019)	13240	33.2%	Twitter

†: total number of microposts in the dataset

Table 5: Datasets comprising labeled microposts.

ilar amount of tweets containing the middle-finger emoji for these languages. Despite the fact that our previous experiments (Table 3) suggest that a smaller amount of data is sufficient for *projection* (i.e. 10k tweets), it would still take more than 2 months to obtain such an amount of German tweets containing the middle finger (Table 1). In order to obtain 10k Portuguese and German tweets more quickly, we included tweets with other predictive emojis. We extracted tweets containing one of the 4 most predictive emojis: *face vomiting*, *pile of poo*, *angry face* or *middle finger*. These 4 emojis are drawn from our English data (Table 1) in order to further demonstrate crosslingual validity. The distribution of emojis reflects their natural distribution on Twitter.

For non-abusive text we sampled sentences from the Portuguese and German versions of the Web As Corpus (Baroni et al., 2009; Filho et al., 2018) from which we also induced word embeddings with word2vec (Mikolov et al., 2013). We decided against pre-trained Twitter embeddings since for many languages such resources are not available. We opted for a setting applicable to most languages.

Evaluation. We evaluate our emoji-based lexicons on the Portuguese dataset from Fortuna et al. (2019) and the two German datasets from GermEval (Wiegand et al., 2018b; Struß et al., 2019). These are datasets for the classification of abusive microposts. As in our evaluation on English data (Table 6), we refrain from an in-domain evaluation since again we want to avoid topic/author biases (§1). Instead, lexicon-based classifiers and a crosslingual approach are used as baselines. The former classifiers predict a micropost as abusive if one abusive word according to the lexicon has been found. In addition to the two variants of *hurtlex* (Bassignana et al., 2018), ***hl-conservative*** and ***hl-inclusive***, we use a lexicon following the method proposed by Wiegand et al. (2018a) on German (***Wiegand2018-replic***). The latter method cannot be replicated for Portuguese, since essential resources for that approach are missing (e.g. sentiment intensity resources, sentiment view lexicons, a manually annotated base lexicon). Moreover, we consider ***Wiegand-translated***, which is the English lexicon from Wiegand et al. (2018a) translated to the target language via GoogleTranslate[7]. Unlike *Wiegand-replic*, this lexicon is cheap to construct as it only requires the original English lexicon.

Our crosslingual baseline exploits the abundance of labeled training data for abusive language detection on English and neural methods to close the language gap between English and the target language. We use ***multilingual BERT*** in which English, Portuguese and German share the same representation space. As proposed by Pires et al. (2019), we train a text classifier on an English dataset for abusive language detection and test the resulting multilingual model on the Portuguese or German microposts. The model that is learnt on English should be usable on the other languages as well, since the three languages share the same representation space. Our crosslingual approach is trained on the dataset from Zampieri et al. (2019), which like our non-English datasets originates from Twitter.

Table 7 shows the results. We also added an upper bound for our emoji-based approach (*emoji+manual*) in which we also include abusive words manually extracted from the abusive microposts missed by the emoji-based approach. Table 7 suggests that our emoji-based approach is only slightly outperformed by its upper bound and the replicated lexicon from Wiegand et al. (2018a), which depends on expensive resources that do not exist in many languages. It is also interesting that the trans-

---

[7] https://translate.google.com

			supervised		lexicon	
test	**training**	**majority-class**	**Nobata et al. (2016)**	**BERT**	**Wiegand et al. (2018a)**	**emoji**
Razavi	Warner	40.5	54.0	68.2	66.1	**75.0**
	Waseem	40.5	51.7	62.9	74.2	**75.0**
	Wulczyn	40.5	**75.1**	72.3	74.8	75.0
	Zampieri	40.5	73.3	**76.7**	74.2	75.0
	Average	40.5	63.4	70.0	72.4	**75.0**
Warner	Razavi	46.1	55.4	60.5	65.0	**69.2**
	Waseem	46.1	56.5	66.9	64.6	**69.2**
	Wulczyn	46.1	60.2	58.0	63.4	**69.2**
	Zampieri	46.1	60.6	62.0	64.7	**69.2**
	Average	46.1	58.2	61.8	64.4	**69.2**
Waseem	Razavi	40.6	57.8	58.4	**63.3**	62.4
	Warner	40.6	58.3	**62.8**	58.7	62.4
	Wulczyn	40.6	56.3	55.4	**62.9**	62.4
	Zampieri	40.6	62.6	63.2	**63.5**	62.4
	Average	40.6	58.2	59.9	62.1	**62.4**
Wulczyn	Razavi	46.9	70.7	**78.4**	73.7	70.6
	Warner	46.9	56.2	60.2	70.1	**70.6**
	Waseem	46.9	51.3	61.6	**72.4**	70.6
	Zampieri	46.9	73.0	**83.1**	72.4	70.6
	Average	46.9	61.9	70.9	**72.1**	70.6
Zampieri	Razavi	40.0	61.0	72.6	72.7	**72.8**
	Warner	40.0	53.6	59.3	63.5	**72.8**
	Waseem	40.0	56.4	59.7	72.3	**72.8**
	Wulczyn	40.0	69.4	71.7	71.9	**72.8**
	Average	40.0	60.1	65.8	70.1	**72.8**

Table 6: Cross-domain classification of English microposts; best result in **bold**; evaluation measure: F1.

	Portuguese	**German**	
classifier		**G.Eval 18**	**G.Eval 19**
majority	40.64	39.75	40.48
hl-inclusive	59.65	57.99	60.77
hl-conservative	62.14	59.72	61.74
Wiegand-translated	57.72	58.90	62.09
multilingual BERT	61.84	61.71	63.01
emoji	64.08	65.15	67.72
emoji+manual	**64.33**	66.25	**68.76**
Wiegand-replic	*N/A*	**66.37**	68.10

Table 7: F1 of crosslingual micropost classifiers; BERT is trained on (English) data from Zampieri et al. (2019).

lated lexicon from Wiegand et al. (2018a) is notably worse than the replicated lexicon. We found that there is a substantial amount of abusive words which cannot be translated into the target language for lack of a counterpart. For example, *spic* refers to a member of the Spanish-speaking minority in the USA. This minority does not exist in most other cultures. For such entries, GoogleTranslate produces the original English word as the translation. In our translated German lexicon, 33% of the entries were such cases. Similarly, we expect some abusive words in German to lack an English counterpart. Therefore, induction methods employing data from the target langage, such as the replicated lexicon or our emoji-based approach, are preferable to translation.

5 Disambiguation of Abusive Words

Many potentially abusive words are not meant to be abusive, i.e. deliberately hurt someone, in all situations in which they are used. For instance, the word *fuck* is abusive in (9) but it is not in (10).

(9) @USER Remorse will get you nowhere, sick **fuck**.
(10) It's so hot and humid what the fuck I'm dying

While operators of social media sites are increasingly facing pressure to react to abusive content on their platforms, they are not necessarily targeting profane language as in (10). In fact, users may see advances of operators against their profane posts as unnecessary and as an infringement of their freedom of speech. Therefore, automated methods to filter textual content of social media sites should ideally distinguish between abusive and profane usage of potentially abusive words.

5.1 Disambiguation with the Help of Emojis

While much previous work (e.g. Davidson et al. (2017)) may frame this task as simply another text classification task in abusive language detection, we consider this as a word-sense disambiguation task. As a consequence, we argue that for robust classification, it is insufficient to have as labeled training data just arbitrary utterances classified as abuse and mere profanity. Instead, as we will also demonstrate, training data have to comprise mentions of those potentially abusive expressions that also occur in the test data. Such an undertaking is very expensive if the training data are to be manually annotated. We propose a more inexpensive alternative in which emojis are employed. We consider tweets containing potentially abusive words as abusive training data if they co-occur with the middle-finger emoji (11)-(14).

(11) @USER Mind ur own business **bitch** 🖕
(12) @USER I have self pride unlike u bastard **bitch** 🖕
(13) @USER Coming from the fake as **fuck** president lol 🖕
(14) @USER @USER How about you **fuck** off Hector! 🖕

Given the scarcity of abusive language even on Twitter (Founta et al., 2018), we consider plain tweets that contain this target word as negative (non-abusive) training data (15)-(18).

(15) Im tired of people complaining about the little shit when I lost my father to that cancer bitch
(16) @USER: I get it nature Im your bitch
(17) Me: you 75% margarita drink some water Aunty: fuck water I'm on vacation
(18) @USER We dont fuck wit each other but happy bday ... celebrate life fam

The supervised classifier we design is a feature-based classifier (SVM). Holgate et al. (2018) report that on the fine-grained classification of (potentially) abusive words such an approach outperforms deep learning methods. We employ an even more lightweight feature set to show that simple features may already help in this task. Table 8 displays our feature set.

5.2 Evaluation of Disambiguation

For evaluation we created a gold standard in which mentions of the two frequent but ambiguous abusive words *fuck* and *bitch* occur (Table 9). We chose these particular two words because they are the only abusive words that are both sufficiently ambiguous and frequent on the dataset from Holgate et al. (2018). That dataset was the only existing dataset with word-specific annotation that was available to us at the time we carried out our experiments so that we could use it as one baseline.[8]

For each of the two words, we extracted 1,000 tweets in which it occurs and had them annotated via crowdsourcing (ProlificAcademic[9]). Each tweet was annotated as *abusive* or *profane* based on the majority of 5 annotators (native speakers of English). (*The supplementary notes contain the annotation guidelines.*)

5.2.1 Baselines for Disambiguation

Text Classification. We train a supervised text classifier (BERT) on each of the following two large datasets (containing several thousand microposts) manually annotated on the micropost level. The dataset from Davidson et al. (2017) distinguishes between the 3 classes: *hate speech*, *offensive language* and *other*. The first category matches our definition of abusive language whereas the second category resembles our category of profane language. We train our classifier on these two categories. The Kaggle-dataset[10] has a more fine-grained class inventory, and the class *insult* can be best mapped to our definition of abusive language. Since profane language can be found in all of the remaining classes, we use the microposts of all other classes as training data for our second class.

Word-specific Classification. We consider the fine-grained class inventory from the manually annotated dataset introduced by Holgate et al. (2018). Unlike the previous baseline, which consists of micropost-level annotation, this dataset contains word-specific annotation, i.e. potentially abusive words annotated in context. This allows us to reduce the training data to contain exclusively contextual mentions of either of our target words (i.e. *bitch* and *fuck*). We use the class *express aggression* as a proxy for our class of abuse while all other occurrences are used as merely profane usages. Given that we have word-specific training data, we train an SVM-classifier on the disambiguation features from Table 8 as we do with our proposed classifier (§5.1).

Heuristic Baseline. In this baseline, training data for abusive usage is approximated by tweets containing the target word and a username. The rationale is that abuse is always directed against a person and such persons are typically represented by a username in Twitter. As profane training data, we consider tweets containing the target word but lacking any username. Given that we have word-specific training data, we again train an SVM-classifier on the disambiguation features (Table 8).

5.2.2 Results of Disambiguation

Table 10 shows the result of our evaluation. For our emoji-based method (and the heuristic baseline), we trained on 2,000 samples containing mentions of the respective target word. Further data did not

---

[8] Meanwhile, two further datasets by Pamungkas et al. (2020) and Kurrek et al. (2020) have been made publicly available which might also be suitable for the kind of evaluation we present in our work.

[9] www.prolific.co

[10] www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

feature	**explanation**
words immediately preceding and following target word	may be helpful in order to learn phrases such as *fuck off*; larger context is avoided since we are likely to overfit to particular domains
presence of abusive words in context?	target word is likely to be abusive if it co-occurs with other (unambiguously) abusive words; abusive words are identified with the help of the lexicon from Wiegand et al. (2018a)
presence of positive/negative polar expressions in context?	positive polar expressions rarely co-occur with abusive language, negative polar expressions, however, do; the polar expressions are obtained from the *Subjectivity Lexicon* (Wilson et al., 2005)
which pronouns are in context?	2nd person pronouns are typical of abusive usage: *you are a bitch*; 1st person pronouns are likely to indicate non-abusive usage: *I am a bitch*
quotation signs in tweet?	quotation signs indicate reported speech; a tweet may report an abusive remark, however, the reported remark itself may not be perceived as abusive (Chiril et al., 2020)
presence of exclamation sign?	a typical means of expressing high emotional intensity

Table 8: Features for disambiguating a potentially abusive word (referred to as *target word*); *context* is defined as a window of 4 words neighbouring the target word.

	bitch		***fuck***	
class	**freq**	**perc**	**freq**	**perc**
abusive	248	24.8	210	21.0
non abusive	752	75.2	790	79.0
all	1000	100.0	1000	100.0

Table 9: Gold standard data for disambiguation.

		bitch		***fuck***	
approach	**classifier**	**Acc**	**F1**	**Acc**	**F1**
majority	SVM	75.2	42.9	79.0	44.1
heuristic baseline	SVM	74.3	58.3	77.5	57.6
text classif. (Kaggle)	BERT	28.0	57.0	61.7	70.1
text classif. (Davidson)	BERT	75.9	60.1	80.9	65.7
emoji	SVM	**77.3**	**66.3**	**82.9**	71.7
word-specific classif.	SVM	68.9	60.0	82.5	**73.3**

Table 10: Disambiguation of *fuck* and *bitch*; *emoji* uses *middle-finger* emoji as distantly-labeled training data.

improve performance. Our proposed approach outperforms all other classifiers with the exception of the more expensive word-specific classifier on the disambiguation of *fuck*. These results show that emojis can be effectively used for disambiguation.

Since we considered the classifier trained with word-specific annotation an upper bound, we were surprised that our emoji-based classifier outperformed that approach on the disambiguation of *bitch*. In that training data we found abusive instances that, according to our guidelines (*see supplementary notes*), would not have been labeled as abusive (19)-(20). These deviations in the annotation may be the cause of the lower performance.

(19) Wow now im a bitch and its apparently ALWAYS like this. Im ready to be over tonight.
(20) I am many things – but a boring bitch is not one.

The baseline *text classification* is less effective than *word-specific classification*. Our inspection of the former datasets revealed that their annotation is less accurate. Apparently annotators were not made aware that certain words are ambiguous. As a consequence, they seem to have used specific words as a signal for or against abuse. For instance, on the Davidson-dataset, almost all occurrences of *bitch* ($> 97\%$) are labeled as abuse and almost all occurrences of *fuck* ($> 92\%$) as no abuse.

6 Conclusion

We presented a distant-supervision approach for abusive language detection. Our main idea was to exploit emojis that strongly correlate with abusive content. The most predictive emoji is the middle-finger emoji. We employed mentions of such emojis as a proxy for abusive utterances and thus generated a lexicon of abusive words that offers the same performance on cross-domain classification of abusive microposts as the best previously reported lexicon. Unlike that lexicon, our new approach neither requires labeled training data nor any expensive resources. We also demonstrated that emojis can similarly be used in other languages where they outperform a crosslingual classifier and a translated lexicon. Finally, we showed that emojis can also be used to disambiguate mentions of potentially abusive words.

Acknowledgements

This research has been partially supported by the Leibniz ScienceCampus Empirical Linguistics and Computational Modeling, funded by the Leibniz Association under grant no. SAS-2015-IDS-LWC and by the Ministry of Science, Research, and Art (MWK) of the state of Baden-Württemberg. The authors would like to thank Rebecca Wilm for helping us to create our new dataset for the disambiguation of abusive words based on crowdsourcing. We are also grateful to Ines Rehbein for feedback on earlier drafts of this paper.

References

Keith Allen and Kate Burridge. 2006. *Forbidden Words: Taboo and the Censoring of Language*. Cambridge University Press.

Wafa Alorainy, Pete Burnap, Han Liu, Amir Javed, and Matthew L. Williams. 2018. Suspended Accounts: A Source of Tweets with Disgust and Anger Emotions for Augmenting Hate Speech Data Sample. In *Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 581–586, Chengdu.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate Speech Detection is Not as Easy as You May Think: A Closer Look at Model Validation. In *Proceedings of the ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 45–53, Paris, France.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetti. 2009. The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation*, 43(3):209–226.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A Multilingual Lexicon of Words to Hurt. In *Proceedings of the Italian Conference on Computational Linguistics (CLiC-It)*, Torino, Italy.

Patricia Chiril, Véronique Moriceau, Farah Benamara, Alda Mari, Gloria Origgi, and Marlène Coulomb-Gully. 2020. He said "who's gonna take care of your children when you are at ACL?": Reported sexist acts are not sexist. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4055–4066, Online.

Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. 2020. Hybrid Emoji-Based Masked Language Models for Zero-Shot Abusive Language Detection. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 943–949, Online.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM)*, Montréal, Canada.

Jacob Devlin, Ming-Wei Chang, KentonLee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 4171–4186, Minneapolis, MN, USA.

Giulia Donato and Patrizia Paggio. 2017. Investigating Redundancy in Emoji Use: Study on a Twitter Based Corpus. In *Proceedings of the Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA)*, pages 118–126, Copenhagen, Denmark.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1615–1625, Copenhagen, Denmark.

Jorge A. Wagner Filho, Rodrigo Wilkens, Marco Idiart, and Aline Villavicencio. 2018. The brWaC corpus: A new open resource for Brazilian Portuguese. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 4339–4344, Miyazaki, Japan.

Paula Fortuna, João Rocha da Silva, Juan Soler-Company, Leo Wanner, and Sérgio Nunes. 2019. A Hierarchically-Labeled Portuguese Hate Speech Dataset. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, pages 94–104, Florence, Italy.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behaviour. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM)*, Stanford, CA, USA.

Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien, and Jun Long. 2015. A Lexicon-based Approach for Hate Speech Detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4):2015–230.

Eric Holgate, Isabel Cachola, Daniel Preoţiuc-Pietro, and Junyi Jessy Li. 2018. Why Swear? Analyzing and Inferring the Intentions of Vulgar Expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4405–4414, Brussels, Belgium.

Mladen Karan and Jan Šnajder. 2018. Cross-Domain Detection of Abusive Language Online. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, pages 132–137, Brussels, Belgium.

Jana Kurrek, Haji Mohammad Saleem, and Derek Ruths. 2020. Towards a Comprehensive Taxonomy and Large-Scale Annotated Corpus for Online Slur Usage. In *Proceedings of the Workshop on Online Abuse and Harms (WOAH)*, pages 138–149, Online.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Julia Mendelsohn, Yulia Tsvetkov, and Dan Jurafsky. 2020. A Framework for the Computational Linguistic Analysis of Dehumanization. *Frontiers in Artificial Intelligence*, 3(55).

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of Workshop at the International Conference on Learning Representations (ICLR)*, Scottsdale, AZ, USA.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant Supervision for Relation Extraction without Labeled Data. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL/IJCNLP)*, pages 1003–1011, Singapore.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 145–153, Republic and Canton of Geneva, Switzerland.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Do You Really Want to Hurt Me? Predicting Abusive Swearing in Social Media. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 6237–6246, Online.

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deeper Attention to Abusive User Content Moderation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1125–1135, Copenhagen, Denmark.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Dohar, Qatar.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4996–5001, Florence, Italy.

Matthew Purver and Stuart Battersby. 2012. Experimenting with Distant Supervision for Emotion Classification. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 182–491, Avignon, France.

Amir Hossein Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. Offensive Language Detection Using Multi-level Classification. In *Proceedings of the Canadian Conference on Artificial Intelligence*, pages 16–27, Ottawa, Canada.

Ira P. Robbins. 2008. Digitus Impudicus: The Middle Finger and the Law. *UC Davis Law Review*, 41:1402–1485.

David. E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Parallel distributed processing: explorations in the microstructure of cognition. In *Learning internal representations by error propagation*, pages 318–362. MIT Press Cambridge.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the EACL-Workshop on Natural Language Processing for Social Media (SocialNLP)*, pages 1–10, Valencia, Spain.

Julia Maria Struß, Melanie Siegel, Josef Ruppenhofer, Michael Wiegand, and Manfred Klenner. 2019. Overview of GermEval Task 2, 2019 Shared Task on the Identification of Offensive Language. In *Proceedings of the GermEval Workshop*, pages 352–363, Erlangen, Germany.

Partha Pratim Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-Supervised Acquisition of Labeled Class Instances using Graph Random Walks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 582–590, Honolulu, HI, USA.

Peter Turney. 2002. Thumbs up or Thumbs down?: Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424, Philadelphia, PA, USA.

William Warner and Julia Hirschberg. 2012. Detecting Hate Speech on the World Wide Web. In *Proceedings of the Workshop on Language in Social Media (LSM)*, pages 19–26, Montréal, Canada.

Zeerak Waseem and Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL – Student Research Workshop*, pages 88–93, San Diego, CA, USA.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 602–608, Minneapolis, MN, USA.

Michael Wiegand, Josef Ruppenhofer, Anna Schmidt, and Clayton Greenberg. 2018a. Inducing a Lexicon of Abusive Words – A Feature-Based Approach. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 1046–1056, New Orleans, LA, USA.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018b. Overview of the GermEval 2018 Shared Task on the Identification of Offensive Language. In *Proceedings of the GermEval Workshop*, pages 1–10, Vienna, Austria.

David Wiener. 1999. Negligent Publication of Statements Posted on Electronic Bulletin Boards: Is There Any Liability Left After Zeran? *Santa Clara Law Review*, 39(3):905–939.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing Contextual Polarity in Phrase-level Sentiment Analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 347–354, Vancouver, BC, Canada.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1391–1399, Perth, Australia.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Koumar. 2019. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In *Proceedings of the International Workshop on Semantic Evaluation (SemEval)*, pages 75–86, Minneapolis, MN, USA.

Haoti Zhong, Hao Li, Anna Cinzia Squicciarini, Sarah Michele Rajtmajer, Christopher Griffin, David J. Miller, and Cornelia Caragea. 2016. Content-Driven Detection of Cyberbullying on the Instagram Social Network. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3952–3958, New York City, NY, USA.

A Systematic Review of
Reproducibility Research in Natural Language Processing

Anya Belz
University of Brighton, UK
`a.s.belz@brighton.ac.uk`

Shubham Agarwal
Heriot Watt University, UK
`sa201@hw.ac.uk`

Anastasia Shimorina
Université de Lorraine / LORIA, France
`anastasia.shimorina@loria.fr`

Ehud Reiter
University of Aberdeen, UK
`e.reiter@abdn.ac.uk`

Abstract

Against the background of what has been termed a reproducibility crisis in science, the NLP field is becoming increasingly interested in, and conscientious about, the reproducibility of its results. The past few years have seen an impressive range of new initiatives, events and active research in the area. However, the field is far from reaching a consensus about how reproducibility should be defined, measured and addressed, with diversity of views currently increasing rather than converging. With this focused contribution, we aim to provide a wide-angle, and as near as possible complete, snapshot of current work on reproducibility in NLP, delineating differences and similarities, and providing pointers to common denominators.

1 Introduction

Reproducibility is one of the cornerstones of scientific research: inability to reproduce results is, with few exceptions, seen as casting doubt on their validity. Yet it is surprisingly hard to achieve, 70% of scientists reporting failure to reproduce someone else's results, and more than half reporting failure to reproduce their own, a state of affairs that has been termed the 'reproducibility crisis' in science (Baker, 2016). Following a history of troubling evidence regarding difficulties in reproducing results (Pedersen, 2008; Mieskes et al., 2019), where 24.9% of attempts to reproduce own results, and 56.7% of attempts to reproduce another team's results, are reported to fail to reach the same conclusions (Mieskes et al., 2019), the machine learning (ML) and natural language processing (NLP) fields have recently made great strides towards recognising the importance of, and addressing the challenges posed by, reproducibility: there have been several workshops on reproducibility in ML/NLP including the Reproducibility in ML Workshop at ICML'17, ICML'18 and ICLR'19; the Reproducibility Challenge at ICLR'18, ICLR'19, NeurIPS'19, and NeurIPS'20; LREC'20 had a reproducibility track and shared task (Branco et al., 2020); and NeurIPS'19 had a reproducibility programme comprising a code submission policy, a reproducibility challenge for ML results, and the ML Reproducibility checklist (Whitaker, 2017), later also adopted by EMNLP'20 and AAAI'21. Other conferences have foregrounded reproducibility via calls, chairs' blogs,[1] special themes and social media posts. Sharing code, data and supplementary material providing details about data, systems and training regimes[2] is firmly established in the ML/NLP community, virtually all main events now encouraging and making space for it. Reproducibility even seems set to become a standard part of reviewing processes via checklists. Far from beginning to converge in terms of standards, terminology and underlying definitions, however, this flurry of work is currently characterised by growing diversity in all these respects. We start below by surveying concepts and definitions in reproducibility research, areas of particular disagreement, and identify categories of work in current NLP reproducibility research. We then use the latter to structure the remainder of the paper.

Selection of Papers: We conducted a structured review employing a stated systematic process for identifying all papers in the field that met specific criteria. Structured reviews are a type of meta-review more common in fields like medicine but beginning to be used more in NLP (Reiter, 2018; Howcroft et al., 2020).

Specifically, we selected papers as follows. We

---

[1] `https://2020.emnlp.org/blog/2020-05-20-reproducibility`

[2] There are some situations where it is difficult to share data, e.g. because the data is commercially confidential or because it contains sensitive personal information. But the increasing expectation in NLP is that authors should share as much as possible, and justify cases where it is not possible.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 381–393
April 19 - 23, 2021.

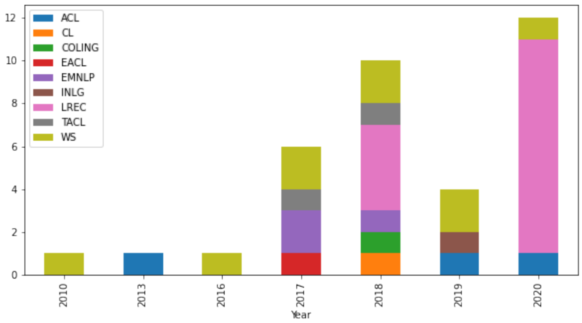


Figure 1: 35 papers from ACL Anthology search, by year and venue.[5]

searched the ACL Anthology for titles containing either of the character strings *reproduc* or *replica*, either capitalised or not.[3] This yielded 47 papers; following inspection we excluded 12 of the papers as not being about reproducibility in the present sense.[4] We found 25 additional papers in non-ACL NLP/ML sources, and a further 7 in other fields.

Figure 1 shows[5] how the 35 papers from the ACL Anthology search are distributed over years: one paper a year at most until 2017/18 when interest seems to have increased spontaneously, before dropping off again. The renewed high numbers for 2020 are almost entirely due to the LREC REPROLANG shared task (see Section 5 below).

2 Terminology and Frameworks

Reproducibility research in NLP and beyond uses a bewildering range of closely related terms, often with conflicting meaning, including reproducibility, repeatability, replicability, recreation, re-run, robustness, repetition and generalisability. The fact that no formal definition of any of these terms singly, let alone in relation to each other, is generally accepted as standard, or even predominant, in NLP at present, is clearly a problem for a survey paper. In this section, we review usage before identifying common-ground terminology that will enable us to talk about the research we survey.

The two most frequently used 'R-terms', *reproducibility* and *replicability*, are also the most problematic. For the ACM (Association for Computing Machinery, 2020), results have been *reproduced* if "obtained in a subsequent study by a person or team other than the authors, using, in part, artifacts provided by the author," and *replicated* if "obtained in a subsequent study by a person or team other than the authors, without the use of author-supplied artifacts" (although initially the terms were defined the other way around[6]). The definitions are tied to team and software (artifacts), but it is unclear how much of the latter have to be the same for reproducibility, and how different the team needs to be for either concept.

Rougier et al. (2017) tie definitions (just) to new vs. original software: "*Reproducing* the result of a computation means running the same software on the same input data and obtaining the same results. [...] *Replicating* a published result means writing and then running new software based on the description of a computational model or method provided in the original publication, and obtaining results that are similar enough to be considered equivalent." It is clear from the many reports of failures to obtain 'same results' with 'same software and data' in recent years that the above definitions raise practical questions such as how to tell 'same software' from 'new software,' and how to determine equivalence of results.

Wieling et al. (2018) define *reproducibility* as "the exact re-creation of the results reported in a publication using the same data and methods," but then discuss (failing to) *replicate* results without defining that term, while also referring to the "unfortunate swap" of the definitions of the two terms put forward by Drummond (2009).

Whitaker (2017), followed by Schloss (2018), tie definitions to data as well as code:

		Data	
		Same	*Different*
Code	*Same*	Reproducible	Replicable
	Different	Robust	Generalisable

The different definitions of *reproducibility* and *replicability* above, put forward in six different contexts, are not compatible with each other. Grappling with degrees of similarity between properties of experiments such as the team, data and software involved, and between results obtained, each draws the lines between terms differently, and moreover variously treats reproducibility and replicability as properties of either systems or results. All are patchy, not accounting for some circumstances, e.g. a team reproducing its own results, not defining

---

[3] `grep -E ‘title *=.*(r|R)\}*(eproduc|eplica)’ anthology.bib`

[4] Most were about annotation and data replication.

[5] Data and code: https://github.com/shubhamagarwal92/eacl-reproducibility

[6] ACM swapped definitions of the two terms when prompted by NISO to "harmonize its terminology and definitions with those used in the broader scientific research community." (Association for Computing Machinery, 2020).

some concepts, e.g. sameness, or not specifying what can serve as a 'result,' e.g. leaving the status of human evaluations and dataset recreations unclear.

The extreme precision of the definitions of the International Vocabulary of Metrology (VIM) (JCGM, 2012) (which the ACM definitions are supposed to be based on but aren't quite) offers a common terminological denominator. The VIM definitions of *reproducibility* and *repeatability* (no other R-terms are defined) are entirely general, made possible by two key differences compared to the NLP/ML definitions above. Firstly, in a key conceptual shift, reproducibility and repeatability are properties of **measurements** (not of systems or abstract findings). The important difference is that the concept of reproducibility now references a specified way of obtaining a **measured quantity value** (which can be an evaluation metric, statistical measure, human evaluation method, etc. in NLP). Secondly, reproducibility and repeatability are defined as the *precision* of a measurement under specified conditions, i.e. the distribution of the quantity values obtained in *repeat* (or *replicate*) measurements.

In VIM, **repeatability** is the precision of measurements of the same or similar object obtained under the same conditions, as captured by a specified set of **repeatability conditions**, whereas **reproducibility** is the precision of measurements of the same or similar object obtained under different conditions, as captured by a specified set of **reproducibility conditions**. See Appendix C for a full set of VIM definitions of the bold terms above.

To make the VIM terms more recognisable in an NLP context, we also call repeatability **reproducibility under same conditions**, and (VIM) reproducibility **reproducibilty under varied conditions**. Finally, we refer to experiments carrying out repeat measurements regardless of same/varied conditions as 'reproduction studies.'

Categories of Reproducibility Research: Using the definitions above, the work we review in the remainder of the paper falls into three categories (corresponding to Sections 3–5):

Reproduction under same conditions: As near as possible exact recreation or reuse of an existing system and evaluation set-up, and comparison of results.[7]

Reproduction under varied conditions: Reproduction studies with deliberate variation of one or more aspects of system and/or measurement, and comparison of results.

Multi-test studies: Multiple reproduction studies connected e.g. because of an overall multi-test design, and/or use of same methodology.

3 Reproduction Under Same Conditions

Papers reporting reproductions under same conditions account for the bulk of NLP reproducibility research to date. The difficulty of achieving 'sameness of system' has taken up a lot of the discussion space. As stressed by many papers (Crane, 2018; Millour et al., 2020), recreation attempts have to have access to code, data, full details/assumptions of algorithms, parameter settings, software and dataset versions, initialisation details, random seeds, run-time environment, hardware specifications, etc.

A related and striking finding, confirmed by multiple repeatability studies, is that results often depend in surprising ways and to surprising degrees on seemingly small differences in model parameters and settings, such as rare-word thresholds, treatment of ties, or case normalisation (Fokkens et al., 2013; Van Erp and Van der Meij, 2013; Dakota and Kübler, 2017). Effects are often discovered during system recreation from incomplete information, when a range of values is tested for missing details. The concern is that the ease with which such NLP results are perturbed casts doubt on their generalisability and robustness.

The difficulties in recreating, or even just rerunning, systems with same results have led to growing numbers of reproducibility checklists (Olorisade et al., 2017; Pineau, 2020), and tips for making system recreation easier, e.g. the PyTorch (Paszke et al., 2017) recommended settings.[8]

We analysed reproduction studies under same conditions from 34 pairs of papers, and identified 549 individual score pairs where reproduction object, method and outcome were clear enough to include in comparisons (for a small number of papers this meant excluding some scores). Table 1 in Appendix A provides a summary of the results. In 36 cases, the reproduction study did not produce scores, e.g. because resource limits were reached,

---

[7] This excludes the countless cases where results for a previous method are used as a baseline or other comparitor, but experiments are not run again.

[8] `https://pytorch.org/docs/stable/notes/randomness.html`

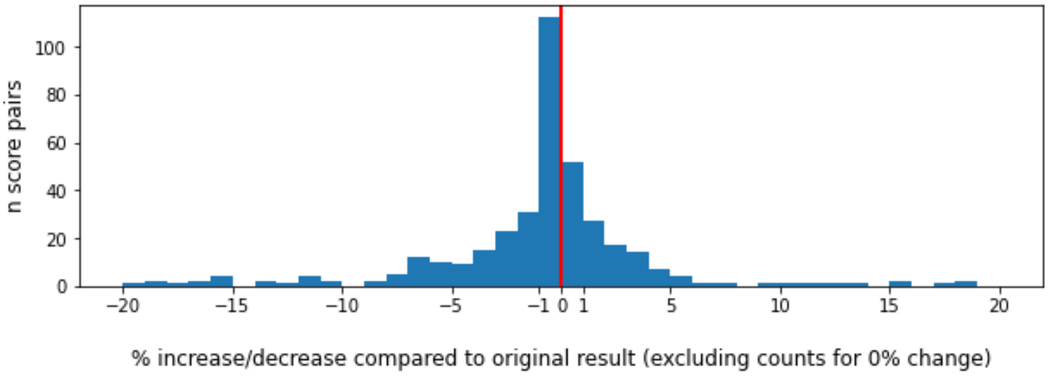


Figure 2: Histogram of percentage differences between original and reproduction scores (bin width = 1; clipped to range -20..20).

or code didn't work. This left 513 cases where the reproduction study produced a value that could be compared to the original score. Out of these, just 77 (14.03%) were exactly the same. Out of the remaining 436 score pairs, in 178 cases (40.8%), the reproduction score was better than the original, and in 258 cases (59.2%) it was worse.

We also examined the size of the differences between original and reproduction scores. For this purpose we computed percentage change (increase or decrease) for each score pair, and looked at the distribution of size and direction of change. For this analysis, we excluded score pairs where one or both scores were 0, as well as 4 very large outliers (all increases). Results are shown in the form of a histogram with bin width 1 (and clipped to range -20..20) in Figure 2. The plot clearly shows the imbalance between worse (60% of non-same scores) and better (40%) reproduction scores. The figure also shows that a large number of differences fall in the -1..1 range. However, the majority of differences, or 3/5, are greater than +/-1%, and about 1/4 are greater than +/-5%.

4 Reproduction Under Varied Conditions

Reproduction studies under varied conditions deliberately vary one or more aspects of system, data or evaluation in order to explore if similar results can be obtained. There are far fewer papers of this type (see Table 2 in Appendix B for an overview) than papers reporting reproduction studies under same conditions; however, note that we are not including papers here that use an existing method for a new language, dataset or domain, without controlling for other conditions being the same in experiments.

Horsmann and Zesch (2017) pick up strong results by Plank et al. (2016) showing LSTM tagging to outperform CRF and HMM taggers, and test whether they can be confirmed for datasets with finer-grained tag sets. Using 27 corpora (21 languages) with finer-grained tag sets, they systematically compare results for the 3 models, and show that LSTMs do perform better, and that their superiority grows in proportion to tag set size.

Htut et al. (2018a) recreate the grammar induction model PRPN (Shen et al., 2018), testing different versions with different data. PRPN is confirmed to be "strikingly effective" at latent tree learning. In a subsequent repeat study under same conditions, Htut et al. (2018b) test PRPN using the authors' own code, obtaining the same headline result.

Millour et al. (2020) attempt to get the POS tagger for Alsatian from Magistry et al. (2018) to work with the same accuracy for a different dataset. Collaborating with, and using resources provided by, the original authors and recreating some where necessary, the best result obtained was 0.87 compared to the original 0.91.

Abdellatif and Elgammal (2020) varied conditions of reproduction for classification results by Howard and Ruder (2018), and were able to confirm outcomes for three new non-English datasets, in all three respects (value, finding, conclusion) identified by Cohen et al. (2018).

Pluciński et al. (2020) and Garneau et al. (2020) both find that the cross-lingual word embedding mappings proposed by Artetxe et al. (2018) yield worse results on more distant language pairs.

Vajjala and Rama (2018)'s automatic essay scoring classification system was tested on different datasets and/or languages in three studies (Arhiliuc et al., 2020; Caines and Buttery, 2020; Huber and Çöltekin, 2020) all of which found performance to drop on the new data.

5 Multi-test and Multi-lab Studies

Work in this category is *multi-test*, in the sense of involving multiple reproduction studies, in a uniform framework using uniform methods. Some of it is also *multi-lab* in that reproduction studies are carried out by more than one research team. For example, in one multi-test repeatability study, Wieling et al. (2018) randomly select five papers each from ACL'11 and ACL'16 for which code/data was available. In a uniform design, original authors were contacted for help if needed, a maximum time limit of 8h was imposed, and all work was done by the same Masters student. It's not clear how scores were selected (not all are attempted), and reasons for failure are not always clear even from linked material. Of the 120 score pairs obtained, 60 were the same, 12 reproduction scores were better, 22 were worse, and 26 runs failed (including exceeding the time limit). See Table 1 for summary.

Rodrigues et al. (2020) recreated six SemEval'18 systems from the Argument Reasoning Comprehension Task, following system descriptions and/or reusing code, with no time limit. Scores were the same for one system, and within +/- 0.036 points for the other five; the SemEval ranking was exactly the same. Systems were also run on a corrected version of the shared-task data (which contained unwitting clues). This resulted in much lower scores casting doubt on the validity of the original shared task results.

REPROLANG (Branco et al., 2020) is so far the only multi-lab (as well as multi-test) study of reproducibility in NLP. It was run as a selective shared task, and required participants to conform to uniform rules. 11 papers were selected for reproduction via an open call and direct selection. Participants had to 'reproduce the paper,' using information contained/linked in it. Participants submitted (a) a report on the reproduction, and (b) the software used to obtain the results as a Docker container (controlling variation from dependencies and run-time environments) on GitLab. Submissions were reviewed in great detail, submitted code was test-run and checked for hard-coding of results. 11 out of 18 submissions were judged to conform with requirements. One original paper (Vajjala and Rama, 2018) attracted four reproductions (Bestgen, 2020; Huber and Çöltekin, 2020; Caines and Buttery, 2020; Arhiliuc et al., 2020) in what must be a groundbreaking first in NLP. See Table 1 for summaries of all 11 reproductions. An aspect the organisers did not control was how to draw conclusions about reproducibility; most contributions provide binary conclusions but vary in how similar they require results to be for success. E.g. the four papers reproducing Vajjala and Rama (2018) all report similarly large deviations, but only one (Arhiliuc et al., 2020) concludes that the reproduction was not a success.

6 Conclusions

It seemed so simple: share all data, code and parameter settings, and other researchers will be able to obtain the same results. Yet the systems we create remain stubbornly resistant to this goal: a tiny 14.03% of the 513 original/reproduction score pairs we looked at were the same. At the same time, worryingly small differences in code have been found to result in big differences in performance.

Another striking finding is that reproduction under same conditions far more frequently yields results that are worse than results that are better: we found 258 out of 436 non-same reproduction results (59.2%) to be worse, echoing findings from psychology (Open Science Collaboration, 2015). Why this should be the case for reproduction under *same* conditions is unclear. It is probably to be expected for reproduction under *different* conditions, as larger parameter spaces, more datasets and languages etc., are tested subsequently, and the original work may have selected better results.

There is a lot of work going on in NLP on reproducibility right now; it is to be hoped that we can solve the vexing and scientifically uninteresting problem of how to rerun code and get the same results soon, and move on to addressing far more interesting questions of how reliable, stable and generalisable promising NLP results really are.

Acknowledgments

We thank our reviewers for their valuable feedback. Shubham Agarwal's PhD fees are supported by Adeptmind Inc., Toronto, Canada.

References

Mohamed Abdellatif and Ahmed Elgammal. 2020. ULMFiT replication. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5579–5587, Marseille, France. European Language Resources Association.

Cristina Arhiliuc, Jelena Mitrović, and Michael Granitzer. 2020. Language proficiency scoring. In *Pro-*

ceedings of The 12th Language Resources and Evaluation Conference, pages 5624–5630, Marseille, France. European Language Resources Association.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Association for Computing Machinery. 2020. Artifact review and badging Version 1.1, August 24, 2020. `https://www.acm.org/publications/policies/artifact-review-and-badging-current`.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760.

Monya Baker. 2016. Reproducibility crisis. *Nature*, 533(26):353–66.

Yves Bestgen. 2020. Reproducing monolingual, multilingual and cross-lingual CEFR predictions. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5595–5602, Marseille, France. European Language Resources Association.

Daniel M. Bikel. 2004. Intricacies of collins' parsing model. *Computational Linguistics*, 30(4):479–511.

Bernd Bohnet, Ryan McDonald, Gonçalo Simões, Daniel Andor, Emily Pitler, and Joshua Maynez. 2018. Morphosyntactic tagging with a meta-BiLSTM model over context sensitive token encodings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2642–2652, Melbourne, Australia. Association for Computational Linguistics.

S.R.K. Branavan, David Silver, and Regina Barzilay. 2011. Learning to win by reading manuals in a monte-Carlo framework. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 268–277, Portland, Oregon, USA. Association for Computational Linguistics.

António Branco, Nicoletta Calzolari, Piek Vossen, Gertjan Van Noord, Dieter van Uytvanck, João Silva, Luís Gomes, André Moreira, and Willem Elbers. 2020. A shared task of a new, collaborative type to foster reproducibility: A first exercise in the area of language science and technology with REPROLANG2020. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5539–5545, Marseille, France. European Language Resources Association.

Ana Brassard, Tin Kuculo, Filip Boltužić, and Jan Šnajder. 2018. TakeLab at SemEval-2018 task12: Argument reasoning comprehension with skip-thought vectors. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1133–1136, New Orleans, Louisiana. Association for Computational Linguistics.

Andrew Caines and Paula Buttery. 2020. REPROLANG 2020: Automatic proficiency scoring of Czech, English, German, Italian, and Spanish learner essays. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5614–5623, Marseille, France. European Language Resources Association.

HongSeok Choi and Hyunju Lee. 2018. GIST at SemEval-2018 task 12: A network transferring inference knowledge to argument reasoning comprehension task. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 773–777, New Orleans, Louisiana. Association for Computational Linguistics.

Maximin Coavoux and Benoît Crabbé. 2016. Neural greedy constituent parsing with dynamic oracles. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 172–182, Berlin, Germany. Association for Computational Linguistics.

K Bretonnel Cohen, Jingbo Xia, Pierre Zweigenbaum, Tiffany J Callahan, Orin Hargraves, Foster Goss, Nancy Ide, Aurélie Névéol, Cyril Grouin, and Lawrence E Hunter. 2018. Three dimensions of reproducibility in natural language processing. In *LREC... International Conference on Language Resources & Evaluation:[proceedings]. International Conference on Language Resources and Evaluation*, volume 2018, page 156. NIH Public Access.

Michael Collins. 1999. Head-driven statistical models for natural language parsing. Ph.D. thesis.

Michael Cooper and Matthew Shardlow. 2020. CombiNMT: An exploration into neural text simplification models. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5588–5594, Marseille, France. European Language Resources Association.

Matt Crane. 2018. Questionable answers in question answering research: Reproducibility and variability of published results. *Transactions of the Association for Computational Linguistics*, 6:241–252.

Daniel Dakota and Sandra Kübler. 2017. Towards replicability in parsing. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 185–194, Varna, Bulgaria. INCOMA Ltd.

Chris Drummond. 2009. Replicability is not reproducibility: nor is it good science. Presented at 4th Workshop on Evaluation Methods for Machine Learning held at ICML'09.

Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. 2013. Offspring from reproduction problems: What replication failure teaches us. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1691–1701, Sofia, Bulgaria. Association for Computational Linguistics.

Paula Fortuna, Juan Soler-Company, and Sérgio Nunes. 2019. Stop PropagHate at SemEval-2019 tasks 5 and 6: Are abusive language classification results reproducible? In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 745–752, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Nuno Freire, José Borbinha, and Pável Calado. 2012. An approach for named entity recognition in poorly structured data. In *Extended Semantic Web Conference*, pages 718–732. Springer.

Qiaozi Gao, Malcolm Doering, Shaohua Yang, and Joyce Chai. 2016. Physical causality of action verbs in grounded language understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1814–1824, Berlin, Germany. Association for Computational Linguistics.

Nicolas Garneau, Mathieu Godbout, David Beauchemin, Audrey Durand, and Luc Lamontagne. 2020. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings: Making the method robustly reproducible as well. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5546–5554, Marseille, France. European Language Resources Association.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*.

Yulan He, Chenghua Lin, and Harith Alani. 2011. Automatically extracting polarity-bearing topics for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 123–131, Portland, Oregon, USA. Association for Computational Linguistics.

Tobias Horsmann and Torsten Zesch. 2017. Do LSTMs really work so well for PoS tagging? – a replication study. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 727–736, Copenhagen, Denmark. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Phu Mon Htut, Kyunghyun Cho, and Samuel Bowman. 2018a. Grammar induction with neural language models: An unusual replication. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4998–5003, Brussels, Belgium. Association for Computational Linguistics.

Phu Mon Htut, Kyunghyun Cho, and Samuel Bowman. 2018b. Grammar induction with neural language models: An unusual replication. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 371–373, Brussels, Belgium. Association for Computational Linguistics.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, Berlin, Germany. Association for Computational Linguistics.

Eva Huber and Çağrı Çöltekin. 2020. Reproduction and replication: A case study with automatic essay scoring. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5603–5613, Marseille, France. European Language Resources Association.

JCGM. 2012. International vocabulary of metrology-basic and general concepts and associated terms (vim).

Yung Han Khoe. 2020. Reproducing a morphosyntactic tagger with a meta-BiLSTM model over context sensitive token encodings. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5563–5568, Marseille, France. European Language Resources Association.

Taeuk Kim, Jihun Choi, and Sang-goo Lee. 2018. SNU_IDS at SemEval-2018 task 12: Sentence encoder with contextualized vectors for argument reasoning comprehension. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1083–1088, New Orleans, Louisiana. Association for Computational Linguistics.

Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 590–599, Portland, Oregon, USA. Association for Computational Linguistics.

Pierre Magistry, Anne-Laure Ligozat, and Sophie Rosset. 2018. Étiquetage en parties du discours de langues peu dotées par spécialisation des plongements lexicaux. In *Conférence sur le Traitement Automatique des Langues Naturelles*, Rennes, France.

Margot Mieskes, Karën Fort, Aurélie Névéol, Cyril Grouin, and Kevin Cohen. 2019. Community perspective on replicability in natural language processing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 768–775, Varna, Bulgaria. INCOMA Ltd.

Alice Millour, Karën Fort, and Pierre Magistry. 2020. Répliquer et étendre pour l'alsacien "étiquetage en parties du discours de langues peu dotées par spécialisation des plongements lexicaux" (replicating and extending for Alsatian : "POS tagging for low-resource languages by adapting word embeddings"). In *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). 2e atelier Éthique et TRaitemeNt Automatique des Langues (ETeRNAL)*, pages 29–37, Nancy, France. ATALA et AFCP.

Andrew Moore and Paul Rayson. 2018. Bringing replication and reproduction together with generalisability in NLP: Three reproduction studies for target dependent sentiment analysis. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1132–1144, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Preslav Nakov and Hwee Tou Ng. 2011. Translating from morphologically complex languages: A paraphrase-based approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1298–1307, Portland, Oregon, USA. Association for Computational Linguistics.

Garrett Nicolai and Grzegorz Kondrak. 2016. Leveraging inflection tables for stemming and lemmatization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1138–1147, Berlin, Germany. Association for Computational Linguistics.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2018. NLITrans at SemEval-2018 task 12: Transfer of semantic knowledge for argument comprehension. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1099–1103, New Orleans, Louisiana. Association for Computational Linguistics.

Babatunde Kazeem Olorisade, Pearl Brereton, and Peter Andras. 2017. Reproducibility of studies on text mining for citation screening in systematic reviews: evaluation and checklist. *Journal of biomedical informatics*, 73:1–13.

Open Science Collaboration. 2015. Estimating the reproducibility of psychological science. *Science*, 349(6251).

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NeurIPS-W*.

Ted Pedersen. 2008. Empiricism is not a matter of faith. *Computational Linguistics*, 34(3):465–470.

Joelle Pineau. 2020. The machine learning reproducibility checklist v2.0.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Kamil Pluciński, Mateusz Lango, and Michał Zimniewicz. 2020. A closer look on unsupervised cross-lingual word embeddings mapping. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5555–5562, Marseille, France. European Language Resources Association.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Kyeongmin Rim, Jingxuan Tu, Kelley Lynch, and James Pustejovsky. 2020. Reproducing neural ensemble classifier for semantic relation extraction in-Scientific papers. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5569–5578, Marseille, France. European Language Resources Association.

João Rodrigues, Ruben Branco, João Silva, and António Branco. 2020. Reproduction and revival of the argument reasoning comprehension task. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5055–5064, Marseille, France. European Language Resources Association.

Jonathan Rotsztejn, Nora Hollenstein, and Ce Zhang. 2018. ETH-DS3Lab at SemEval-2018 task 7: Effectively combining recurrent and convolutional neural networks for relation classification and extraction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 689–696, New Orleans, Louisiana. Association for Computational Linguistics.

Nicolas P Rougier, Konrad Hinsen, Frédéric Alexandre, Thomas Arildsen, Lorena A Barba, Fabien CY Benureau, C Titus Brown, Pierre De Buyl, Ozan Caglayan, Andrew P Davison, et al. 2017. Sustainable computational science: the rescience initiative. *PeerJ Computer Science*, 3:e142.

Christina Sauper, Aria Haghighi, and Regina Barzilay. 2011. Content models with attitude. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 350–358, Portland, Oregon, USA. Association for Computational Linguistics.

Patrick D Schloss. 2018. Identifying and overcoming threats to reproducibility, replicability, robustness, and generalizability in microbiome research. *MBio*, 9(3).

Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. In *International Conference on Learning Representations*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Junfeng Tian, Man Lan, and Yuanbin Wu. 2018. ECNU at SemEval-2018 task 12: An end-to-end attention-based neural network for the argument reasoning comprehension task. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1094–1098, New Orleans, Louisiana. Association for Computational Linguistics.

Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Learning semantically and additively compositional distributional representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1277–1287, Berlin, Germany. Association for Computational Linguistics.

Sowmya Vajjala and Taraka Rama. 2018. Experiments with universal CEFR classification. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 147–153, New Orleans, Louisiana. Association for Computational Linguistics.

Marieke Van Erp and Lourens Van der Meij. 2013. Reusable research? a case study in named entity recognition. Technical report.

Duy-Tin Vo and Yue Zhang. 2015. Target-dependent twitter sentiment classification with rich automatic features. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Bo Wang, Maria Liakata, Arkaitz Zubiaga, and Rob Procter. 2017. Tdparse: Multi-target-specific sentiment recognition on twitter. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 483–493.

Kirstie Whitaker. 2017. The MT Reproducibility Checklist. `https://www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist.pdf`.

Martijn Wieling, Josine Rawee, and Gertjan van Noord. 2018. Squib: Reproducibility in computational linguistics: Are we willing to share? *Computational Linguistics*, 44(4):641–649.

Meiqian Zhao, Chunhua Liu, Lu Liu, Yan Zhao, and Dong Yu. 2018. BLCU_NLP at SemEval-2018 task 12: An ensemble model for argument reasoning based on hierarchical attention. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1104–1108, New Orleans, Louisiana. Association for Computational Linguistics.

Appendices

A Table of Reproductions Under Same Conditions

Original paper	Reproduction study (same conditions)	NLP task	Summary of score differences
Collins (1999)	Gildea (2001)	Phrase-structure parsing	+16.7% error on Model 1 results
Collins (1999)	Bikel (2004)	Phrase-structure parsing	+11% error on Model 2 results on WSJ00; later same results with help from Collins
Freire et al. (2012)	Van Erp and Van der Meij (2013)	NER	"Despite feedback from Freire [...], results remained 20 points below those reported in Freire et al. (2012) in overall F-score" (Fokkens et al., 2013)
Nakov and Ng (2011)	Wieling et al. (2018)	MT	*Unsuccessful (scripts did not work)
He et al. (2011)	Wieling et al. (2018)	Sentiment analysis	*-0.18 points
Sauper et al. (2011)	Wieling et al. (2018)	Topic modelling	*Unsuccessful on 3 scores (8h cut-off reached)
Liang et al. (2011)	Wieling et al. (2018)	Question answering	*Exact reproduction of 2 scores in 4h
Branavan et al. (2011)	Wieling et al. (2018)	Joint learning of game strategy and text selection from game manual	*Unsuccessful on 7 scores (scripts did not generate output)
Coavoux and Crabbé (2016)	Wieling et al. (2018)	Constituent parsing	*9/18 scores same, 9/18 parser did not complete for 4 languages
Gao et al. (2016)	Wieling et al. (2018)	Semantic role grounding	*Exact reproduction of 44/72 scores, 17 worse, 11 better, average -0.62 points
Hu et al. (2016)	Wieling et al. (2018)	Sentiment analysis, NER	*exact reproduction of 1/2 scores, 1 worse -0.2 points
Nicolai and Kondrak (2016)	Wieling et al. (2018)	Stemming, lemmatisation	*2/8 scores -3.4 and -1.55 points, 6/8 scores took longer than 8h cut-off
Tian et al. (2016)	Wieling et al. (2018)	Sentence completion	*4/6 scores reproduced exactly, 2/6 differed -0.1 and +0.01 %-points).
Badjatiya et al. (2017)	Fortuna et al. (2019)	hate speech detection	reproduction under same conditions not possible due to issue with code; recreated/corrected system did well at OffensEval'19 but not at HatEval'19
Choi and Lee (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	1/1 score +0.002 points
Zhao et al. (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	1/1 score +0.038 points
Tian et al. (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	1/1 score -0.021 points
Niven and Kao (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	1/1 score +0.033 points
Kim et al. (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	1/1 score -0.022 points
Brassard et al. (2018)	Rodrigues et al. (2020)	Argument Reasoning Comprehension Task	Exact reproduction of 1/1 score.
Artetxe et al. (2018)	Garneau et al. (2020)	Cross-lingual Mappings of Word Embeddings	Main scores: 2/8 same, 1/8 -0.1, 5/8 +0.1 to +0.3; ablation scores: 4/40 scores same, 19/40 +0.1 to 6.9, 9/40 -0.1 to -0.9, 8/40 took too long
Artetxe et al. (2018)	Pluciński et al. (2020)	Cross-lingual Mappings of Word Embeddings	Main scores: 10/14 better, 4/14 worse; ablation scores: 3/48 scores same, 31/48 better, 14/48 worse
Bohnet et al. (2018)	Khoe (2020)	POS and morphological tagging	POS tagging scores: 35/41 worse, 6/41 better; morph. tagging: 43/46 worse, 3/46 better
Rotsztejn et al. (2018)	Rim et al. (2020)	Relation extraction and classification (SemEval'18 T7)	4 subtasks: 4/4 scores worse, up to 9.04 points; subtask 1.1 by relation: 3/6 worse, 3/6 better
Nisioi et al. (2017)	Cooper and Shardlow (2020)	Simplification	NTS default system: 1/2 automatic scores better, 1/2 automatic scores worse; 2/2 human scores worse

Table continued on next page.

Original paper	Reproduction study (recreation of system)	NLP task	Summary of score differences
Vajjala and Rama (2018)	Bestgen (2020)	Automatic essay scoring (classification)	multilingual: 6/11 better, 5/11 worse; monolingual: 15/27 better, 11/27 worse, 1/27 same; crosslingual: 5/8 better, 1/8 worse, 2/8 same
Vajjala and Rama (2018)	Huber and Çöltekin (2020)	Automatic essay scoring (classification)	multilingual: 3/11 better, 8/11 worse; monolingual: 8/27 better, 19/27 worse; crosslingual: 6/8 better, 2/8 worse
Vajjala and Rama (2018)	Caines and Buttery (2020)	Automatic essay scoring (classification)	multilingual: 9/11 better, 2/11 worse; monolingual: 14/27 better, 11/27 worse, 2/27 same; crosslingual: 1/8 better, 7/8 worse
Vajjala and Rama (2018)	Arhiliuc et al. (2020)	Automatic essay scoring (classification)	multilingual: 11/11 worse; monolingual: 7/27 better, 20/27 worse; crosslingual: 1/8 better, 5/8 worse, 2/8 same
Magistry et al. (2018)	Millour et al. (2020)	POS tagging for Alsatian	baseline: same (0.78 Acc); main: worse (Acc 0.87 vs. 0.91)
Howard and Ruder (2018)	Abdellatif and Elgammal (2020)	Sentiment classification, question classification, topic classification	3/6 better, 3/6 worse
Vo and Zhang (2015)	Moore and Rayson (2018)	Target Dependent Sentiment analysis	6/6 better
Wang et al. (2017)	Moore and Rayson (2018)	Target Dependent Sentiment analysis	2/5 better, 3/5 worse
Tang et al. (2016)	Moore and Rayson (2018)	Target Dependent Sentiment analysis	3/3 worse

Table 1: Tabular overview of individual repeatability tests from 34 paper pairs, and a total of 549 score pairs. * = additional information obtained from hyperlinked material.

Where scores obtained in a repeatability study (reproduction under same conditions) are worse than in the original work, this should *not* be interpreted as casting the original work in a negative light. This is because it is normally not possible to create the exact same conditions in repeatability studies (and lower scores can result from such differences), and because the outcome from multiple repeatability studies may be very different.

For a small number of papers, the score pairs included in this table are a subset of scores reported in the paper. More generally, the summary in the last column should not be interpreted as a summary of the whole paper and its findings.

Our intention here is to summarise differences that have been reported in the literature, rather than draw conclusions about what may have caused the differences.

B Table of Reproductions Under Varied Conditions

Original paper	reproduction study (confirmation of finding)	NLP task	Summary of outcome (as interpreted by authors)
Plank et al. (2016)	Horsmann and Zesch (2017)	POS tagging	Confirmed for finer-grained tagsets
Shen et al. (2018)	Htut et al. (2018a,b)	Grammar induction	Overall finding confirmed (that PRPN is a high performing grammar induction method)
Magistry et al. (2018)	Millour et al. (2020)	POS tagging	Not confirmed, reproduction results worse by $>$ 10 BLEU points
Vajjala and Rama (2018)	Arhiliuc et al. (2020)	Automatic essay scoring (classification)	Lower classification results on a corpus of Asian learners' English.
Vajjala and Rama (2018)	Caines and Buttery (2020)	Automatic essay scoring (classification)	Lower classification results for English and Spanish CEFR datasets, and some adversarial data (e.g., scrambled English texts).
Vajjala and Rama (2018)	Huber and Çöltekin (2020)	Automatic essay scoring (classification)	Lower classification results for English Cambridge Learner Corpus.
Artetxe et al. (2018)	Garneau et al. (2020)	Cross-lingual mappings of word embeddings	For other distant language pairs (from English to Estonian, Latvian, Finnish, Persian) the method did not converge or obtained lower scores.
Artetxe et al. (2018)	Pluciński et al. (2020)	Cross-lingual mappings of word embeddings	For other distant language pairs (from English to Czech, Polish) the method did not converge or obtained lower scores.
Howard and Ruder (2018)	**Abdellatif and Elgammal (2020)	Sentiment classification, question classification, topic classification	Confirmed that transfer learning (pretraining) improves final classification accuracy.

Table 2: Tabular overview of individual studies to confirm a previous research finding. * = additional information obtained from hyperlinked material; ** = reproduction study had minor differences, e.g. hyperparameter tuning was omitted (Abdellatif and Elgammal, 2020).

The comments from the caption for Table 1 also apply here, but note that some differences between original and reproduction study are overt and intentional in the case of the papers in this table, whereas they are not intentional and often inadvertent in the case of the papers in Table 1.

C Verbatim VIM and ACM Definitions

2.1 (2.1) **measurement**	process of experimentally obtaining one or more **quantity values** that can reasonably be attributed to a **quantity**
2.15 measurement precision (precision)	closeness of agreement between **indications** or **measured quantity values** obtained by replicate **measurements** on the same or similar objects under specified conditions
2.20 (3.6, Notes 1 and 2) **repeatability condition of measurement** (repeatability condition)	condition of **measurement**, out of a set of conditions that includes the same **measurement procedure**, same operators, same **measuring system**, same operating conditions and same location, and replicate measurements on the same or similar objects over a short period of time
2.21 (3.6) **measurement repeatability** (repeatability)	**measurement precision** under a set of **repeatability conditions of measurement**
2.24 (3.7, Note 2) **reproducibility condition of measurement** (reproducibility condition)	condition of **measurement**, out of a set of conditions that includes different locations, operators, **measuring systems**, and replicate measurements on the same or similar objects
2.25 (3.7) **measurement reproducibility** (reproducibility)	**measurement precision** under **reproducibility conditions of measurement**
2.3 (2.6) **measurand**	**quantity** intended to be measured

Table 3: VIM definitions of repeatability and reproducibility (JCGM, 2012).

Repeatability (Same team, same experimental setup)	The measurement can be obtained with stated precision by the same team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same location on multiple trials. For computational experiments, this means that a researcher can reliably repeat her own computation.
Reproducibility (Different team, same experimental setup)*	The measurement can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using the author's own artifacts.
Replicability (Different team, different experimental setup)*	The measurement can be obtained with stated precision by a different team, a different measuring system, in a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using artifacts which they develop completely independently.

Results Validated: This badge is applied to papers in which the main results of the paper have been successfully obtained by a person or team other than the author. Two levels are distinguished:

Results Reproduced v1.1	The main results of the paper have been obtained in a subsequent study by a person or team other than the authors, using, in part, artifacts provided by the author.
Results Replicated v1.1	The main results of the paper have been independently obtained in a subsequent study by a person or team other than the authors, without the use of author-supplied artifacts.

In each case, exact replication or reproduction of results is not required, or even expected. Instead, the results must be in agreement to within a tolerance deemed acceptable for experiments of the given type. In particular, differences in the results should not change the main claims made in the paper.

Table 4: ACM definitions (bold) and badges (underlined) (Association for Computing Machinery, 2020).

Bootstrapping Multilingual AMR with Contextual Word Alignments

Janaki Sheth**[*] **Young-Suk Lee*** **Ramón Fernandez Astudillo*** **Tahira Naseem***

Radu Florian* **Salim Roukos*** **Todd Ward***

**Perelman School of Medicine, UPenn, Philadelphia, PA, USA
*IBM Research, Yorktown Heights, NY, USA
`Janaki.Sheth@Pennmedicine.upenn.edu, ramon.astudillo@ibm.com`
`{ysuklee, tnaseem, raduf, roukos, toddward}@us.ibm.com`

Abstract

We develop high performance multilingual Abstract Meaning Representation (AMR) systems by projecting English AMR annotations to other languages with weak supervision. We achieve this goal by bootstrapping transformer-based multilingual word embeddings, in particular those from cross-lingual RoBERTa (XLM-R large). We develop a novel technique for foreign-text-to-English AMR alignment, using the contextual word alignment between English and foreign language tokens. This word alignment is weakly supervised and relies on the contextualized XLM-R word embeddings. We achieve a highly competitive performance that surpasses the best published results for German, Italian, Spanish and Chinese.

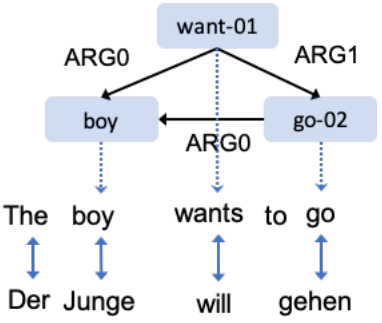


Figure 1: AMR graph for *The boy wants to go* and its German translation *Der Junge will gehen*. Implicit alignments between the English text and AMR concepts are denoted by dotted arrows. Explicit alignments between English and German texts are denoted by solid arrows.

1 Introduction

Abstract Meaning Representation graphs are rooted, labeled, directed, acyclic graphs representing sentence-level semantics (Banarescu et al., 2013). In the example shown in Figure 1, the sentence *The boy wants to go* is parsed into an AMR graph. The nodes of the AMR graph represent the AMR concepts, which may include normalized surface symbols e.g. *boy*, Propbank frames (Kingsbury and Palmer, 2002) e.g. *want-01*, *go-02* as well as other AMR-specific constructs. Edges in an AMR graph represent the relations between concepts. In this example *:arg0*, *:arg1* correspond to standard roles of Propbank.

One distinctive aspect of AMR annotation is the lack of explicit alignments between nodes in the graph and words in the sentences. Since such alignments are essential for training many of present-day AMR parsers, there have been various efforts to link the AMR concepts to their corresponding span of words (Flanigan et al., 2014; Pourdamghani et al., 2014; Lyu and Titov, 2018; Chen and Palmer, 2017). A significant emphasis of this paper is on deriving these alignments for multilingual AMR parsers.

Even though by nature AMR is biased towards English, recent work has evaluated the potential of AMR to work as an interlingua. Hajič et al. (2014) and Xue et al. (2014) categorize and propose refinements for divergences in the annotation between English and Chinese as well as Czech AMRs. Anchiêta and Pardo (2018) import the corresponding AMR annotation for each sentence from the English annotated corpus and revisit the annotation to adapt it to Portuguese. However, Damonte and Cohen (2018) show that it may be possible to use the original AMR annotations devised for English as representation for equivalent sentences in other languages without any modification despite the translation divergence. This defines the problem of multilingual AMR parsing that we seek to address in this paper - given a sentence in a foreign language, recover the AMR graph originally designed for its English translation. We implement

---

[*] This research was done during an internship at IBM Research AI.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 394–404
April 19 - 23, 2021.

multilingual AMR parsers for German, Spanish, Italian and Chinese.

In this paper we propose that transformer-based multilingual word embeddings can be a useful tool for addressing the problem of multilingual AMR parsing. Besides using contextual word embeddings as input token embeddings, we leverage them for *annotation projection*, where existing AMR annotations for English are projected to a target language by using contextual word alignments. In our experiments, we employ XLM-RoBerta large (Conneau et al., 2019) as the multilingual pre-trained transformer model. We show that our proposed procedure achieves competitive results as some of the classical methods for text-to-AMR alignment. Furthermore, such a procedure is easily scalable to the 100 languages that XLM-R is trained on.

We also combine different techniques for concept alignments and AMR parser training which significantly improve performance over the base models. For concept alignment, we combine the proposed contextual word alignments with previously established alignment techniques utilizing matching rules tailored to AMR as well as machine translation aligners (Flanigan et al., 2014; Pourdamghani et al., 2014). For AMR parser training, we pre-train an AMR parser on the treebanks of different languages simultaneously and subsequently finetune on each language. This is analogous to the techniques used for silver data pre-training (Konstas et al., 2017; van Noord and Bos, 2017) in AMR parsing and multi-lingual pre-training (Aharoni et al., 2019) in machine translation.

Finally, we conduct a detailed error analysis of the multilingual AMR parsing. One of the major errors we have found involves synonymous concepts, which share the same meaning as the original concepts in English, but differ in spellings. While this error is mainly caused by the fact that the multilingual word embeddings bridge non-English input tokens to English concepts, it also highlights the highly lexical nature of Smatch scoring (Cai and Knight, 2013) which does not take synonymous concepts into consideration. We also elaborate upon error analysis of the direct comparison between our proposed annotation projection method using contextual word alignment and a previous baseline, using fast align.

The rest of the paper is organized as follows: In Section 2, we discuss related work. In Section 3, we present our main proposal on annotation projection based on contextual word alignments. In Section 4, we describe various combination approaches that improve the multilingual parser performances significantly. These include combining word-to-concept alignments, using multi-lingual treebanks and combining human-annotated and synthetic treebanks. In Section 5, we discuss experimental results. In Sections 6 and 7, we present detailed error analyses. We conclude the paper in Section 8.

2 Related work

Multilingual AMR. There have been significant advances in AMR parsing for languages other than English. Previous studies (Hajič et al., 2014; Xue et al., 2014; Migueles-Abraira et al., 2018; Sobrevilla Cabezudo and Pardo, 2019) investigated AMR annotations for a variety of different languages such as Chinese, Czech, Spanish and Brazilian Portuguese. Vanderwende et al. (2015) automatically parse the logical representation for sentences in Spanish, Italian, German and Japanese, which is then converted to AMR using a small set of rules.

While much of this work, along with studies such as Li et al. (2016); Anchiêta and Pardo (2018), produces AMR graphs whose nodes were labeled with words from the target language, Damonte and Cohen (2018) developed AMR parsers for English and used parallel corpora for annotation projection to train Italian, Spanish, German, and Chinese parsers that recover the AMR graph originally designed for the English translation. Their main results showed that the new parsers can overcome certain structural differences between languages.

Similar to Damonte and Cohen (2018), we also train multilingual AMR parsers by projecting English AMR annotation to target foreign languages (German, Spanish, Italian and Chinese), but we depart from their approach in the specifics of the annotation projection by exploring contextual word alignments directly derived from multilingual contextualized word embeddings. While both procedures utilize parallel corpora, the annotation projection of Damonte and Cohen (2018) requires additional supervised training of their statistical word aligner. Our proposed contextualized word alignment is however unsupervised in nature. Alternatively, a recent study by Blloshmi et al. (2020) showed that one may in fact not need alignment-based parsers for cross-lingual AMR, rather modelling concept identification as a *seq2seq* problem. In this paper, we will compare our results to both

Damonte and Cohen (2018) and Blloshmi et al. (2020).

Word vector alignment techniques. Traditional word alignment methods often use parallel corpora and IBM alignment models (Brown et al., 1990, 1993) as well as improved versions (Och and Ney, 2003; Dyer et al., 2013). More recently, there have been an advent of techniques that align vector representation of words from varying levels of supervision (Ruder et al., 2019). Often word vectors are learned independently for each language and then a mapping from source language vectors to target language vectors with a bilingual dictionary is developed (Mikolov et al., 2013; Smith et al., 2017; Artetxe et al., 2017). To reduce the need for bilingual supervision, the iterative method of starting from a minimal seed dictionary and alternating with learning the linear map was employed by a recent body of work (Conneau et al., 2018; Schuster et al., 2019; Artetxe et al., 2018).

The work most similar to ours is Cao et al. (2020) where the authors obtain contextual embedding alignments from multilingual BERT (Devlin et al., 2018; Pires et al., 2019) and subsequently improve the alignments via finetuning using supervised parallel corpora. Our contextual word alignment between two parallel sentences may be thought of as an adaptation of their contextual word retrieval task. However, we refrain from any finetuning of the contextual embeddings and show that the contextual word alignments from the off-the-shelf XLM-R model achieves results competitive to the word alignments by fast-align (see Damonte and Cohen (2018)). This suggests the potential for inexpensive, massive scaling of AMR parsing up to 100 languages on which XLM-R is trained.

3 Annotation projection

We adopt a transition-based parsing approach for AMR parsing following (Ballesteros and Al-Onaizan, 2017; Naseem et al., 2019; Fernandez Astudillo et al., 2020). These produce an AMR graph g from an input sentence s by predicting instead an action sequence a from s as a sequence to sequence problem. This action sequence applied to a state machine M produces then the desired target graph as $g = M(a, s)$. Transition-based parsers require the action sequence for each graph in the training data. This is determined by a rule-based oracle $a = O(g, s)$ which relies on external word-to-node alignments. In all the subsequent experiments we will use the oracle and action set from (Fernandez Astudillo et al., 2020).

3.1 Projection method

In order to train AMR parsers in a non-English language, we use the annotation projection method to leverage existing English AMR annotation and overcome resource shortage in the target language. First, the English text is aligned to corresponding AMR concepts using both rule-based JAMR aligner (Flanigan et al., 2014) and a IBM model type aligner (Pourdamghani et al., 2014). The latter will henceforth be referred to as the EM aligner. Given the English text-to-AMR concept alignments, we then project these to the target language using word alignment. In the following subsection we describe in the proposed word alignment method, called *contextual word alignment*, which is trained in a weakly supervised manner.

3.2 Contextual word alignments

Given two languages, we align word pairs within parallel sentences if their vector representations derived from the underlying multilingual pre-trained model are similar according to cosine distance. As vector representation we use the average of all 24 layers of the XLM-R large contextual embeddings. We will refer to this average as the word's contextual embedding henceforth for simplicity.

More precisely, suppose we have two parallel sentences - $\mathbf{E} = e_0, e_1, e_2, ..., e_M$ in English and $\mathbf{F} = f_0, f_1, f_2, ..., f_N$ in the target language. We will use r to represent the pre-trained multilingual model such that $r(\mathbf{S})_i$ is the contextual embedding for the i^{th} word in sentence $\mathbf{S}$. Then a word $e_i \in \mathbf{E}$ is contextually word aligned to f_j if and only if the cosine similarity score between their word embeddings is the highest. Thus we define the corresponding contextual alignment function $\chi(f_j|e_i)$ as,

$$\chi(f_j|e_i) = \text{argmax}_{0 \leq j \leq |\mathbf{F}|} cos(r(\mathbf{E})_i, r(\mathbf{F})_j). \tag{1}$$

Similarly, performing the same procedure in the reverse direction we have,

$$\chi(e_i|f_j) = \text{argmax}_{0 \leq i \leq |\mathbf{E}|} cos(r(\mathbf{F})_j, r(\mathbf{E})_i) \tag{2}$$

While these methods can be noisy, by only keeping word pairs in their intersection *i.e.* $\chi(\mathbf{E}|\mathbf{F}) \cap \chi(\mathbf{F}|\mathbf{E})$, one can derive the intersection cosine alignment approach which gives us a word-aligned dataset with low coverage but high accuracy.

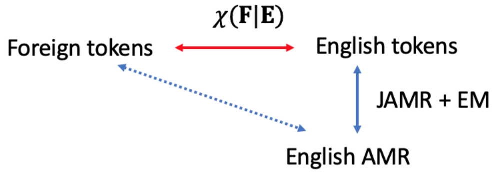


Figure 2: Annotation projection is achieved using JAMR and EM aligners for English text-to-AMR concept alignment and contextual word alignment between tokens of the source (English) and target languages.

As an example, the following are sentences from our German and English training datasets:

E: Establishing models in industrial Innovation
F: Etablierung von Modellen in der industriellen Innovation
Their contextual word alignments are,
$\chi(\mathbf{F}|\mathbf{E})$ = $[(e_0, f_0), (e_1, f_2), (e_2, f_3), (e_3, f_5), (e_4, f_6)]$
$\chi(\mathbf{E}|\mathbf{F})$ = $[(f_0, e_0), (f_1, e_1), (f_2, e_1), (f_3, e_2), (f_4, e_2), (f_5, e_1), (f_6, e_4)]$
$\chi(\mathbf{F}|\mathbf{E}) \cap \chi(\mathbf{E}|\mathbf{F})$
= $[(e_0, f_0), (e_1, f_2), (e_2, f_3), (e_4, f_6)]$

Figure 2 pictorially illustrates our complete annotation projection method using the contextual word alignment $\chi(\mathbf{F}|\mathbf{E})$. English tokens and AMR concepts are aligned using JAMR and EM aligners. The resulting AMR annotation augmented with English word-to-concept alignments is then projected onto the given target language using contextual word embeddings. Henceforth, for brevity we will at times refer to this approach as A.P.

4 Combination approaches

We apply three types of combination techniques to the multilingual AMR parsers, trained by projecting English annotations using contextual word alignments derived from the multilingual contextual word embeddings, each of which improves the parser performance significantly.

4.1 Alignment combination

One such technique is to combine the contextual word alignment based A.P. with the baseline word-to-concept alignment which aligns the target tokens directly to AMR concepts using JAMR and EM aligners. Since the EM aligner is an unsupervised method, it can be directly applied to the target language tokens and English AMR concepts. However, we note that this baseline alignment approach gives incomplete coverage (87% concepts aligned to German, 88% to Italian and 91% to Spanish tokens). Thus, we supplement this by aligning the remaining concepts using the A.P. of Figure 2.

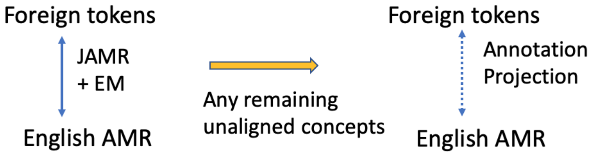


Figure 3: Illustration of the EM, JAMR + A.P. combination alignment: first align target tokens to AMR concepts using JAMR+EM aligners with any remaining concepts then aligned using the annotation projection method proposed in Figure 2.

For example, suppose we have as before two parallel sentences - $\mathbf{E} = e_0, ..., e_M$ in English and $\mathbf{F} = f_0, ..., f_N$ in the target language, as well as AMR concepts $\mathbf{N} = n_0, ..., n_L$. Then one of our proposed foreign text-to-AMR concept combination alignment procedures $EA(f_i|n_j)$ (see Figure 3) is defined as,

$$EA(f_i|n_j) = AP(BA(f_i|n_j)) \tag{3}$$

where $BA(f_i|n_j)$ represents that the j^{th} concept is aligned to the i^{th} token in $\mathbf{F}$ using the baseline aligner BA. If for any concept $n_j \in \mathbf{N}$, $BA(f_i|n_j) =$ None, we use annotation projection to align it where $AP(f_i|n_j)$ is given by,

$$\chi(f_i|e_k) \wedge BA(e_k|n_j) \Rightarrow AP(f_i|n_j) \tag{4}$$

We also experiment with other such alignments, in particular by using the intersection of cosine alignment $(\chi(F|E) \cap \chi(E|F))$ as the contextual word alignment. In this case,

$$EA(f_i|n_j) = \max AP(BA(\mathrm{i}AP(f_i|n_j))) \tag{5}$$

wherein,

$$(\chi(f_i|e_k) \cap \chi(e_k|f_i)) \wedge BA(e_k|n_j) \Rightarrow \mathrm{i}AP(f_i|n_j) \tag{6}$$

As before, $\forall n_j \in \mathbf{N}$ where $\mathrm{i}AP(f_i|n_j) =$ None we align it using the baseline aligner $BA(f_i|n_j)$. For any further remaining unaligned concepts, we employ $\max AP(f_i|n_j)$ which can be described as:

$$\begin{aligned} \max(\chi(f_i|e_k), \chi(e_k|f_i)) \wedge BA(e_k|n_j) \\ \Rightarrow \max AP(f_i|n_j) \end{aligned} \tag{7}$$

That is, we pick the uni-directional contextual word alignment with the higher score and project the AMR annotation accordingly.

4.2 Multilingual treebank combination

In addition to training the parser on the treebank of each language - derived from English treebank via annotation projection - we also experiment with combining all the target language treebanks to create a single multilingual treebank. We notice that pre-training an AMR parser on this multilingual treebank with subsequent finetuning on the treebank of each language, improves performance over the parser trained only on each individual treebank.

4.3 Human and synthetic treebank combination

We create a synthetic AMR corpus by parsing 85k unlabeled sentences from the context portion of SQuAD-2.0. The resulting synthetic AMR graphs are filtered as per the procedure in (Lee et al., 2020) and combined with the AMR-2.0 training set (LDC2017T10), to produce an expanded *AMR-2.0 + SQuAD* training dataset of 94k sentences. We then project annotations of this expanded English treebank onto each of the target languages, and train the corresponding target language parser. We observe that despite the lower quality of the synthetic AMRs as compared to their human-annotated counterparts, their inclusion in the training set significantly improves parser performance.

5 Experimental Results

5.1 AMR Parser and Data

For our experiments, we use the stack-Transformer model (Fernandez Astudillo et al., 2020)[1] as our AMR parser. The stack-Transformer is a transition based parser with a modified Transformer architecture to encode the parser state. It uses a cross entropy loss function and has hyper-parameters similar to those of machine translation described in (Vaswani et al., 2017). We use a beam size of 3 to decode our models and evaluate them using Smatch scores (Cai and Knight, 2013). Model performance values in this manuscript are an average over the best performing models across 3 random seeds. Lastly, the input to the parser - the vector representation of each word - is obtained by averaging over not only all 24 layers of the pre-trained XLM-R large contextual embeddings but also over constituent wordpieces within each word.

For all four languages - German, Spanish, Italian and Chinese - we experiment on AMR1.0 (LDC2015E86). For the first three we also experiment on AMR-2.0 (LDC2017T10). Results from the former are compared to Damonte and Cohen (2018) and from the latter to Blloshmi et al. (2020). Details of our training, dev and test sets are given in Table1.[2] To train each target language parser, we first translate the input sentences of AMR-2.0 and AMR-1.0 with Watson Language Translator.[3] This creates the supervised parallel corpus which we then use for our unsupervised annotation projection via contextual word alignment. We also align target language tokens directly to AMR concepts using JAMR and EM aligners for baseline system evaluation and for combination alignments. We select the best performing models using the devset. Finally, for our best models, we report results using the machine as well as human translations (LDC2020T07) of the test sets.

5.2 Baselines

Our first baseline is zero-shot learning, where we train on the English dataset but test on a foreign language dev-set (Baseline I). The reason behind this experiment is to test the ability of the XLM-R contextual word embeddings to capture the meaning of the given token irrespective of the underlying language. Note that it is only for this experiment that languages for the train and dev sets differ. In another set of experiments we align the target language tokens directly to the AMR concepts only using the JAMR and EM aligners (Baseline II). Lastly, we also test the annotation projection procedure of Damonte and Cohen (2018). Note that while the previous authors use fast align (Dyer et al., 2013) for word alignment between the parallel data and only JAMR aligner for the English text-to-AMR alignment, in Baseline III we have utilized fast align in conjunction with both JAMR and EM aligners (for English text-to-AMR alignment) for improved performance.

5.3 Results

Table 2 compares our different proposed approaches to the three baseline methods using the AMR2.0 and AMR1.0 datasets. We see that our proposed approach - annotation projection with contextual word alignment, in this case using $\chi(\mathbf{F}|\mathbf{E})$ - shows fairly competitive results with

---

[1] `https://github.com/IBM/transition-amr-parser`

[2] Word segmentation is applied to the Chinese raw texts for model training and testing.

[3] https://www.ibm.com/watson/services/language-translator/

Data set	Experiment	Number of sentences	Number of tokens			
			DE	ES	IT	ZH
Train set	AMR2.0 LDC	36k	677k	694k	654k	
	AMR2.0 LDC + synAMR	94k	2.1m	2.2m	2.1m	
	AMR1.0 LDC	10k	222k	240k	227k	195k
Development set	All experiments	1368	30k	32k	31k	26k
Test set	All experiments	1371	31k	33k	32k	27k

Table 1: Details of our dataset

Model	AMR2.0			AMR1.0			
	DE	ES	IT	DE	ES	IT	ZH
Baseline I (zero-shot)	39.0	39.6	41.0	37.4	38.8	39.3	33.4
Baseline II	61.4	66.2	68.3	57.2	60.3	60.7	55.4
Baseline III	63.8	68.7	68.6	56.3	60.8	61.0	54.7
Annotation Projection (A.P)	61.9	67.7	66.8	55.7	60.7	60.5	46.5
EM,JAMR+A.P	63.9	68.7	69.8	57.7	62.3	62.5	55.8
Intersect A.P+EM,JAMR+max(A.P)	64.2	69.1	68.7				
EM,JAMR+A.P (Multilingual)	64.6	69.2	70.4	**58.6**	**62.7**	**62.9**	**58.1**
EM,JAMR+A.P (synAMR)	**67.8**	**71.3**	**72.2**				

Table 2: Dev set Smatch for AMR2.0 and AMR1.0.

Model	Machine translation				Human translation			
	DE	ES	IT	ZH	DE	ES	IT	ZH
Damonte and Cohen (2018)					39	42	43	35
Baseline I (zero-shot)	37.1	37.99	38.5	31.8	36.3	37.6	37.4	30.2
Baseline II	56.1	58.94	59.7	53.3	53.6	57.8	56.8	48.3
Baseline III	55.1	59.24	59.0	53.1	52.7	57.9	57.3	48.1
Annotation Projection (A.P)	54.9	58.9	59.4	44.6	52.7	57.7	57.0	41.4
EM,JAMR + A.P	56.4	60.6	61.3	54.0	53.6	59.2	58.6	48.3
EM,JAMR + A.P (Multilingual)	**57.4**	**61.4**	**61.6**	**55.7**	**54.5**	**60.1**	**59.0**	**50.3**

Table 3: Test set Smatch for AMR1.0.

Model	Machine translation			Human translation			
	DE	ES	IT	DE	ES	IT	ZH
Blloshmi et al. (2020)				53	58	58.1	43.1
EM, JAMR + A.P (Multilingual)	63.8	67.7	69.0	59.9	66.0	65.7	
EM, JAMR + A.P (synAMR)	**66.9**	**69.6**	**71.0**	**62.7**	**67.9**	**67.4**	

Table 4: Test set Smatch for AMR2.0.

those of Baseline III for the target languages of German, Italian and Spanish, especially when applied to the smaller corpus of AMR1.0. This is remarkable considering our method requires no additional training and can be easily generalized for zero-shot learning on all different languages that XLM-R was pretrained on. We then train several parsers using our suggested combination approaches. The first such method comprises of both the EM, JAMR + A.P aligners (see Eq. 3). In a different approach, we use the intersection cosine word alignment based annotation projection (*i.e* $\chi(\mathbf{F}|\mathbf{E}) \cap \chi(\mathbf{E}|\mathbf{F})$). Since this leaves many AMR concepts unaligned, we follow it by aligning concepts using the baseline JAMR and EM aligners. Any leftover unaligned concepts are then aligned using $\max(\chi(\mathbf{E}|\mathbf{F}), \chi(\mathbf{F}|\mathbf{E}))$ (Eq. 5). In another set of experiments, we pre-train a parser on a multilingual treebank, where the train set is a combination of the LDC treebank in all target languages. The parser is then finetuned on each individual language. We surmise that such an experiment will give us a truly multilingual parser capable of successfully decoding all the target languages. Its strength is evident in its performance, it outperforms all our baseline approaches - in the case of AMR1.0 dev set by at least 1.4 points. Finally, in the last two experiments on AMR2.0 we train on the language-specific LDC + SQuaD train set. We see that this gives us our best performing parsers, where the training data is aligned using a combination (EM, JAMR + A.P) alignment.

We test a subset of the AMR2.0 and all of the AMR1.0 models on corresponding test sets. The results are shown in Tables 3 and 4. For AMR1.0, while all of our models including the baselines outperform previously published results, the best performing model is the parser which was trained on multilingual data and whose training input text was aligned to its AMR concepts using the combination of EM, JAMR and A.P aligners. For AMR2.0, models trained on the LDC + SQuAD dataset outperform those trained on multilingual data. Both of these outperform the recently published work of Blloshmi et al. (2020). [4]

We note that the parser performs better on the machine translated test data than on the human translated data. This should be attributed to the training and testing condition mismatch of the human translated test data since all models are trained on machine translated training data. For instance, the out-of-vocabulary (oov) ratio of the human translated test data is consistently higher than that of the machine translated test data. For example, for AMR1.0 the oov ratio of human translated test data vs. machine translated test data is 10.2% vs. 9% for German, 7.3% vs. 6.8% for Spanish, 8.1% vs. 7.6% for Italian and 7.6% vs. 5.5% for Chinese.

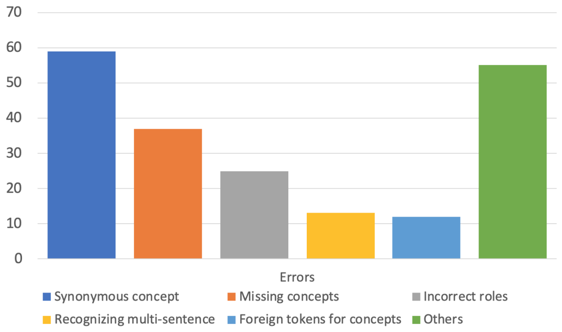


Figure 4: Histogram of different kinds of errors

6 Error analysis

We carried out an error analysis of 56 German sentences parsed by the best performing model trained on the combination of AMR2.0 and SQuAD training data. Statistics of the various errors are depicted in Figure 4. Top 5 most frequent errors include (i) introduction of synonymous concepts, (ii) missing concepts, (iii) incorrect roles, (iv) target tokens in AMR concepts, (v) incorrect parsing of multi-sentence as an instance of conjunction.

6.1 Synonymous concepts

The most common error we encounter is synonymous AMR concepts, as shown in Figure 5. Comparing the expected graph (top) to the parsed version (bottom), we note that concept *previous* is synonymized to *past*. While this error is mainly caused by the fact that the multilingual word embeddings bridge non-English input tokens to English concepts, it also highlights the highly lexical nature of Smatch scoring (Cai and Knight, 2013) which does not take synonymous concepts into consideration. Given that AMR is supposed to represent the core meaning of a sentence regardless of its syntactic and morphological variations, Smatch scoring should be able to capture lexical variations such as synonymous concepts.

---

[4] We did not run experiments with LDC + SQuAD dataset on AMR1.0 since our primary reason for running experiments on AMR1.0 was to more directly be able to compare our results to (Damonte and Cohen, 2018)

```
In this environment, what’s wrong if they
criticize the previous stupefying propaganda
a bit?

(w / wrong-02
      :ARG1 (a2 / amr-unknown)
      :ARG2 (c / criticize-01
            :ARG0 (t / they)
            :ARG1 (p / propaganda
                  :time (p2 / previous)
                  :ARG1-of (s / stupefy-01))
            :degree (b / bit))
      :location (e / environment
            :mod (t2 / this)))

Was ist in dieser Umgebung falsch, wenn sie
die bisherige stupeftende Propaganda ein
bisschen kritisieren?

(w / wrong-02
      :ARG1 (c / criticize-01
            :ARG0 (t2 / they)
            :ARG1 (p2 / propaganda
                  :time (p / past))
                  :degree (b / bit))
      :ARG2 (a / amr-unknown)
      :location (e / environment
            :mod (t / this)))
```

Figure 5: The gold AMR (top) and the parsed AMR (bottom) for a German sentence exemplifying errors: synonymous concept (*previous* vs. *past*), missing concept (concept *stupefy-01* is missing in the parsed AMR), incorrect roles (the two arguments, *:ARG1* and *:ARG2*, of *wrong-02* are swapped in the parsed AMR).

```
In critical moments, we are all descendants of
Yan emperor and Huang emperor.

(d / descend-01
      :ARG0 (w / we
            :mod (a / all))
      :source (a2 / and
            :op1 (p / person
                 :name (n / name
                       :op1 "Yan")
                 :ARG0-of (h / have-org-role-91
                          :ARG2 (e / emperor)))
            :op2 (p2 / person
                 :name (n2 / name :op1 "Huang")
                 :ARG0-of h))
      :time (m / moment
            :ARG1-of (c / critical-02)))

In kritischen Momenten sind wir alle Nachfahren
des Yan Kaisers und Huang Kaisers.

(d / descend-01
      :ARG0 (w / we
            :mod (a / all))
      :ARG1 (a2 / and
            :op1 (p / person
                  :name (n / name
                        :op1 "Yan"
                        :op2 "Kaisers"))
            :op2 (p2 / person
                  :name (n2 / name
                        :op1 "Huang"
                        :op2 "Kaisers")))
      :time (m / moment
            :ARG1-of (c / critical-02)))
```

Figure 6: The gold AMR (top) and the parsed AMR (bottom) for a German sentence illustrating incorrect roles (*:source* is replaced by *:ARG1* in the parsed AMR) and incorrect identification of the target token *Kaisers* as a named entity.

6.2 Missing concepts and incorrect roles

Some concepts are missing in the parsed AMR, such as *stupefy-01* in Figure 5. The parser also incorrectly identifies relations between concepts. In Figure 5, arguments *ARG1* and *ARG2* for concept *wrong-02* are swapped. In Figure 6, the relation *:source* is replaced by frame argument *ARG1*.

6.3 Incorrect parsing of Multi-sentence

Another frequent error includes incorrect parsing of multi-sentence as an instance of conjunction, especially when sentences are demarcated by commas. Note that the multi-sentence errors are not specific to multilingual parsing and occur frequently when parsing English input sentences as well. This multi-sentence error is mostly caused by the ambiguity of commas, which can subsume various semantics depending on the contexts across languages.

6.4 Misrecognition of foreign token as a named entity

Some target tokens may legitimately be realized in the gold AMR, especially when the target tokens are named entities, e.g. *Frankfurt, Anna, Noah, etc*. This often leads to errors in the parsed AMR when a target token is incorrectly recognized as a named entity. In Figure 6, German token *Kaisers* is incorrectly parsed as part of named entities *Yan Kaisers* and *Huang Kaisers*. The failure to capture the correct concept *emperor* for the German token *Kaisers* leads to a subsequent error of not reifying the role to *have-org-role-91*[5], evident in the comparison of the parsed AMR with the gold AMRs.

6.5 Others

Other errors include lack of stemming in the target language, such as *Kaisers* in Figure 6. Stemming errors are mostly caused by the fact that we have not incorporated target language stemmers whereas we have incorporated spacy[6] for English. Some errors are caused by machine translation. English fragmentary input *taking a look* is translated to *Sehen Sie sich*, which is then incorrectly parsed as *imperative* sentence. Nominal target language tokens often fail to invoke predicates. Given the input in English “cultural tyranny in the cloak of nationalism”, *tyranny* invokes the predicate *tyrannize-01*. Its German counterpart *Tyrannei*, however, fails to

---

[5] Refer to https://www.isi.edu/ ulf/amr/lib/roles.html and https://www.isi.edu/ ulf/amr/lib/amr-dict.html/have-org-role-91 for details.

[6] https://spacy.io/

	Contextual Alignment	**Fast Align**
German	23.47	20.52
Italian	29.40	29.30
Spanish	28.81	26.69

Table 5: AMR1.0 parser performance on negations in terms of Smatch. Fast align is compared with the proposed contextual alignment for different languages.

invoke the predicate in “kulturellen Tyrannei im Mantel des Nationalismus”.

7 Word alignment error analysis

We compared the annotation projection for AMR1.0 between fast align and the contextual alignment. As noted in Table 3 they perform comparably for German, Italian and Spanish. However, on detailed analysis we notice that annotation projection using contextualized alignments has a greater coverage in terms of foreign text-to-AMR alignments compared to fast align (eg. for German, contextual alignment A.P. gives 99.95% coverage in comparison to 97.47%.). This is likely due to the fact that fast align is based on an IBM alignment model, which relies on expected counts of alignment pairs and uses additional alignment constraints. Contextualized alignment relies on the unrestricted pairing by cosine distance of the XLM-R contextual word embeddings of the input tokens. Given an English token, the contextualized alignment necessarily aligns it to a foreign language word. Furthermore, since embeddings are contextual and pre-trained with large amounts of data, they are robust to non frequent alignment pairs.

The difference between contextualized alignment and fast align for their coverage is most noticeable for compounds. A German counterpart of English non – tariff is *nichttarifäre*. While contextualized alignment aligns *nichttarifäre* to non, which is subsequently aligned to the concept “–” for polarity, fast align leaves *nichttarifäre* unaligned. Such difference is evidenced in the parser performance on negations realized in diverse morphologies. Comparing the AMR1.0 parser performance on negations between fast align (Baseline III in Table 3) and the contextualized alignment (A.P in Table 3), we find that contextualized alignment consistently outperforms fast align across the three European target languages, as shown in Table 5.

8 Conclusion and future directions

In this paper we propose to use transformer-based multilingual word embeddings for *annotation projection* of AMR annotations. We show that our proposed procedure achieves competitive results as some of the classical methods for text-to-AMR alignment. We apply combination techniques to concept alignments and AMR parser training, which significantly improve performance over the base models. We also provide a detailed error analysis of the multilingual AMR parsing.

Given pre-trained transformer-based multilingual word embeddings, contextual word alignment proves to be a useful avenue for overcoming differences amongst languages and addressing the multilingual AMR problem with weak supervision. Moreover, our annotation projection procedure not only achieves a highly competitive performance for German, Spanish, Italian and Chinese but also permits zero-shot learning to other languages included in the training set of the underlying XLM-R multilingual transformer.

Future work may include diversifying input texts using AMR2text (Mager et al., 2020) generation which can address the difference in results between machine translated and human translated test data. The potential of the AMR parser to overcome translation divergence also points to its utility in an end-to-end multilingual translation system, bypassing the need for supervised parallel corpora for machine translation system training.

Acknowledgement

We thank the anonymous reviewers for helpful suggestions. We also thank Revanth Reddy and Jason Furmanek for their varied inputs.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Rafael Anchiêta and Thiago Pardo. 2018. Towards AMR-BR: A SemBank for Brazilian Portuguese language. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Miguel Ballesteros and Yaser Al-Onaizan. 2017. AMR parsing using stack-LSTMs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1269–1275, Copenhagen, Denmark. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Rexhina Blloshmi, Rocco Tripodi, and Roberto Navigli. 2020. XL-AMR: Enabling cross-lingual AMR parsing with transfer learning techniques. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2487–2500, Online. Association for Computational Linguistics.

Peter Brown, John Cocke, Stephen Della Pietra, Vicent Della Pietra, Fredrick Jelinek, John Lafferty, Robert Mercer, and Paul Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics, Volume 16, Number 2, June 1990*.

Peter Brown, Stephen Della Pietra, Vicent Della Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics, Volume 19, Number 2*.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. In *Proceedings of the 8th International conference on learning representations*.

Wei-Te Chen and Martha Palmer. 2017. Unsupervised AMR-dependency parse alignment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 558–567, Valencia, Spain. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herve Jegou. 2018. Word translation without parallel data. In *Proceedings of the 6th International conference on learning representations*.

Marco Damonte and Shay B. Cohen. 2018. Cross-lingual Abstract Meaning Representation parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1146–1155, New Orleans, Louisiana. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Ramon Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Findings of the EMNLP2020 (to appear)*.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.

Jan Hajič, Ondřej Bojar, and Zdeňka Urešová. 2014. Comparing Czech and English AMRs. In *Proceedings of Workshop on Lexical and Grammatical Resources for Language Processing*, pages 55–64, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Paul R Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*, pages 1989–1993. Citeseer.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Young-Suk Lee, Ramon Fernandez Astudillo, Tahira Naseem, Revanth Gangi Reddy, Radu Florian, and Salim Roukos. 2020. Pushing the limits of amr parsing with self-learning. In *Findings of the EMNLP2020 (to appear)*.

Bin Li, Yuan Wen, Weiguang Qu, Lijun Bu, and Nianwen Xue. 2016. Annotating the little prince with Chinese AMRs. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, pages 7–15, Berlin, Germany. Association for Computational Linguistics.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, USA. Association for Computational Linguistics.

Noelia Migueles-Abraira, Rodrigo Agerri, and Arantza Diaz de Ilarraza. 2018. Annotating Abstract Meaning Representations for Spanish. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4586–4592, Florence, Italy. Association for Computational Linguistics.

Rik van Noord and Johan Bos. 2017. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. *arXiv preprint arXiv:1705.09980*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingua is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Nima Pourdamghani, Yang Gao, Ulf Hermjakob, and Kevin Knight. 2014. Aligning English strings with Abstract Meaning Representation graphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 425–429, Doha, Qatar. Association for Computational Linguistics.

Sebastian Ruder, Iva Vulić, and Søgaar Andersd. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, page 569–630.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Samuel Smith, David Turban, Steven Hamblin, and Nils Hamerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *Proceedings of the 5th International conference on learning representations*.

Marco Antonio Sobrevilla Cabezudo and Thiago Pardo. 2019. Towards a general Abstract Meaning Representation corpus for Brazilian Portuguese. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 236–244, Florence, Italy. Association for Computational Linguistics.

Lucy Vanderwende, Arul Menezes, and Chris Quirk. 2015. An AMR parser for English, French, German, Spanish and Japanese and a new AMR-annotated corpus. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 26–30, Denver, Colorado. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Nianwen Xue, Ondřej Bojar, Jan Hajič, Martha Palmer, Zdeňka Urešová, and Xiuhong Zhang. 2014. Not an interlingua, but close: Comparison of English AMRs to Chinese and Czech. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1765–1772, Reykjavik, Iceland. European Language Resources Association (ELRA).

Semantic Oppositeness Assisted Deep Contextual Modeling for Automatic Rumor Detection in Social Networks

Nisansa de Silva, Dejing Dou
Department of Computer and Information Science,
University of Oregon,
Eugene, USA.
{nisansa,dou}@cs.uoregon.edu

Abstract

Social networks face a major challenge in the form of rumors and fake news, due to their intrinsic nature of connecting users to millions of others, and of giving any individual the power to post anything. Given the rapid, widespread dissemination of information in social networks, manually detecting suspicious news is sub-optimal. Thus, research on automatic rumor detection has become a necessity. Previous works in the domain have utilized the reply relations between posts, as well as the semantic similarity between the main post and its context, consisting of replies, in order to obtain state-of-the-art performance. In this work, we demonstrate that semantic oppositeness can improve the performance on the task of rumor detection. We show that semantic oppositeness captures elements of discord, which are not properly covered by previous efforts, which only utilize semantic similarity or reply structure. Our proposed model learns both explicit and implicit relations between the main tweet and its replies, by utilizing both semantic similarity and semantic oppositeness. Both of these employ the self-attention mechanism in neural text modeling, with semantic oppositeness utilizing word-level self-attention, and with semantic similarity utilizing post-level self-attention. We show, with extensive experiments on recent data sets for this problem, that our proposed model achieves state-of-the-art performance. Further, we show that our model is more resistant to the variances in performance introduced by randomness.

1 Introduction

Social media changed the ecosystem of the World Wide Web by making it possible for any individual, regardless of their level of knowledge of web technologies, to create and maintain *profiles* online. At the same time, various social media provided these individuals with means to tap into the information disseminated by others (e.g., Facebook by *adding friends*, Twitter by *following*). By virtue of other mechanisms, such as *Facebook pages* and *Twitter lists*, the reach of each individual was then extended to the range of thousands-to-millions of users. New content, in the form of *posts*, is created on social media sites each passing second.

The rapidity of this post creation is such, that it is possible to claim that social media reflect a near real-time view of the events in the real world (Veyseh et al., 2019). While it was, indeed, beneficial in terms of *volume* of data, to have private individuals be content creators and propagators of information, this created significant issues, from the perspective of *veracity* of the data. This gave rise to a challenge of detecting *fake news* and *rumors* (which, in this study, we refer to as the task of *rumor detection*). The need for rumor detection has come to the forefront, in light of its momentous impacts on political events (Jin et al., 2017) and social (Jin et al., 2014) or economic (Domm, 2013) trends.

Manual intervention on this task would require extensive analysis of and reasoning about various sources of information, resulting in long response times, which are intolerable, given the impact of these rumors, and the rate at which they spread. Thus, *automatic rumor detection*, toward which we contribute in this paper, has become an important area of contemporary research. Cao et al. (2018) define any piece of information, of which the veracity status was questionable at the time of posting, as a rumor. They further claim that a rumor may later be verified to be true or false by other authorized sources. We follow their definition in this work; thus, we define the task of rumor detection as: Given a piece of information from a social network, predict whether the piece of information is a rumor or not using the conversations which were induced by the said piece of information. The initial piece of information could be a tweet or a user

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 405–415
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

post, and the induced conversation would be the replies from other users (which we use as contextual information). Following the conventions in the literature, in this work, we refer to a main post and its replies as a *thread*.

In this paper, we utilize the semantic oppositeness proposed by (de Silva and Dou, 2019) to improve the rumor detection task, which has so far been restricted to only considering semantic similarity. We further prove that semantic oppositeness is well-suited to be applied to this domain, under the observation that rumor threads are more discordant than those of non-rumors. We further observe that, within rumor threads, false rumor threads continue to be clamorous; while true rumor threads settle into inevitable acquiescence. We claim that semantic oppositeness can help in distinguishing this behavior as well.

We propose word-level self-attention mechanism for the semantic oppositeness to augment the tweet level self-attention mechanism for the semantic similarity. We model the explicit and implicit connections within a thread, using a relevancy matrix. Unlike a regular adjacency matrix, our relevancy matrix recognizes the coherence of each sub-tree of conversation rooted at the main post, while acknowledging that, by definition, for this task, the main tweet must be directly related all the rest of the tweets, regardless of the degrees of separation that may exist between them. We conduct extensive experiments to compare our proposed model with the state-of-the-art studies conducted on the same topic. To the best of our knowledge, this work is the first to utilize semantic oppositeness in rumor detection. In summary, our contributions in this paper include:

- We introduce a novel method for rumor detection, based on both semantic similarity and semantic oppositeness, utilizing the main post and the contextual replies.
- We model the explicit and implicit connections within a thread, using a relevancy matrix, which is then used to balance the impact semantic similarity and semantic oppositeness have on the overall prediction.
- We conduct experiments on recent rumor detection data sets and compare with numerous state-of-the-art baseline models to show that we achieve superior performance.

The remainder of this paper is organized as follows: Section 2 presents related work, and then Section 3 provides a formal definition of the problem, along with our proposed solution. It is followed by Section 4 discussing experiments and results. Finally the Section 5 concludes the paper.

2 Related Work

Semantic oppositeness is the mathematical counterpart of semantic similarity (de Silva and Dou, 2019). While implementations of semantic similarity (Jiang and Conrath, 1997; Wu and Palmer, 1994) are more widely used than those of semantic oppositeness, there are a number of studies which work on deriving or using semantic oppositeness (de Silva et al., 2017; Paradis et al., 1982; Mettinger, 1994; Schimmack, 2001; Rothman and Parker, 2009; de Silva, 2020). However, it is noted that almost all of these studies are reducing oppositeness from a scale to either bipolar scales (Schimmack, 2001) or simple anonymity (Paradis et al., 1982; Jones et al., 2012). The study by de Silva et al. (2017) proves that this reduction is incorrect and proposes an alternative oppositeness function. Their follow-up study, de Silva and Dou (2019) creates a word embedding model for this function. In this study, we use the oppositeness embeddings created by them.

Rumor detection task has been approached on three fronts, according to Cao et al. (2018): feature engineering, propagation-based, and deep learning. In the *feature engineering* approach, posts are transformed into feature representations by hand-designed features and sent to a statistical model to be classified. In addition to textual information, structural evidences (Castillo et al., 2011; Yang et al., 2012) and media content (Gupta et al., 2012) are also utilized. Given that this approach depends heavily on the quality of the hand-designed feature sets, it is neither scalable, nor transferable to other domains. The *propagation-based* approach is built on the assumption that the propagation pattern of a rumor is significantly different to that of a non-rumor. It has been deployed to detect rumors in social networks (Ma et al., 2017). However, this method does not pay any heed to the information in the post content itself. As expected, *deep learning* approach, automatically learns effective features (Ma et al., 2016, 2018; Veyseh et al., 2019). Ma et al. (2016) claim that these discovered features capture the underlying representations of the posts, and hence, improve the generalization performance, while making it easy to be adapted into a new domain or a social medium for the purpose of rumor detection.Yang et al. (2020) propose a slide window-based system for feature extraction. None of these state-of-the-art work attempt to check rumour veracity akin to attempts by Hamidian and Diab (2019a) and Derczynski et al. (2017). Instead, they attempt to do classification on the already established baseline. Thus, our work also follow the approach of the former rather than the latter. The work by Hamidian and Diab (2019b) does focus on rumor detection and classification. However, they are not using the data sets common to the state-of-the-art work mentioned above to evaluate their approach.

Our work is most related to the rumor detection model on Twitter by means of deep learning to capture contextual information (Veyseh et al., 2019). However, we also derive inspiration from earlier work on the same topic (Ma et al., 2018), which utilized the tree-like structures of the posts, and the work by de Silva and Dou (2019), which introduced the oppositeness embedding model. The early work by Ma et al. (2018) uses Recursive Neural Networks (RvNN) for the construction of the aforementioned tree-like structures of the posts, based on their *tf-idf* representations.

The following work by Veyseh et al. (2019) acknowledges the usefulness of considering the innate similarities between replies, but further claims that only considering the replies along the tree-like structure only exploits the explicit relations between the main posts and their replies, and thus ignores the implicit relations among the posts from different branches based on their semantics. Under this claim, they disregard the tree-like structure entirely. In our work, we preserve the idea of considering semantic similarities to discover the implicit relationships among posts, as proposed by (Veyseh et al., 2019).

However, we augment the model and reintroduce the explicit relationships proposed by Ma et al. (2018) in a balancing of information between implicit and explicit. Further, we note that all these prior works have been solely focused on the similarity between the posts and have ignored the oppositeness metric. To the best of our knowledge, we are the first to utilize oppositeness information in the rumor detection task.

3 Methodology

We use a recent work (Veyseh et al., 2019) on rumor detection as our baseline. Their work, in turn, was heavily influenced by the earlier work on rumor detection in Twitter (Ma et al., 2018). A tweet set I is defined as shown in Equation 1, where R_0 is the initial tweet and $R_1, R_2, \ldots, R_T$ are replies, such that T is the count of replies. Each tweet R_i is a sequence of words $W_1, W_2, ..., W_n$, such that n is the count of words. We tokenize the tweets; and in this work, *tokens* and *words* are used interchangeably. We also define the relevance matrix M, which carries the information of the tree structure of the tweet tree in Equation 2, where $A \star B$ denotes that A and B belong to the same tree in the forest obtained by eliminating the initial tweet. We show the process in Fig 1 as well. Our input is the pair $P = (I, M)$, which differs from (Veyseh et al., 2019), where only I was used as the input. The entire data set is represented by D.

Following the convention of (Veyseh et al., 2019) which is our baseline, we classify each pair (I, M) into four labels: 1) Not a rumor (NR); 2) False Rumor (FR); 3) True Rumor (TR); and 4) Unrecognizable (UR), It should be noted that the distinction between "*False Rumor*" and "*True Rumor*" is drawn from the truthfulness of R_0.

$$I = (R_0, R_1, R_2, \ldots, R_T) \tag{1}$$

$$m_{i,j} = \begin{cases} 1 & \text{if } R_i = R_0 \vee R_j = R_0 \\ 1 & \text{if } R_i \star R_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

In simpler terms, we can represent Veyseh et al. (2019) as a trivial *relevance matrix* where all elements are set at 1. The success of Veyseh et al. (2019) over previous state-of-the-art methods attest to the success of using a *relevance matrix* over vanilla *adjacency matrix*. In this work what we do with the above described *relevance matrix* M is to augment the implicit relationship consideration using the high level structure of the explicit relationships, hence bringing in the best-of-both-worlds. In summary, the set of edges in the *relevancy matrix* is a super-set of the set of edges in the *adjacency matrix*. In addition to the edges that were in the *adjacency matrix*, the *relevancy matrix* also has edges that carry implicit connection information. Thus, by definition, the *relevancy matrix* is more

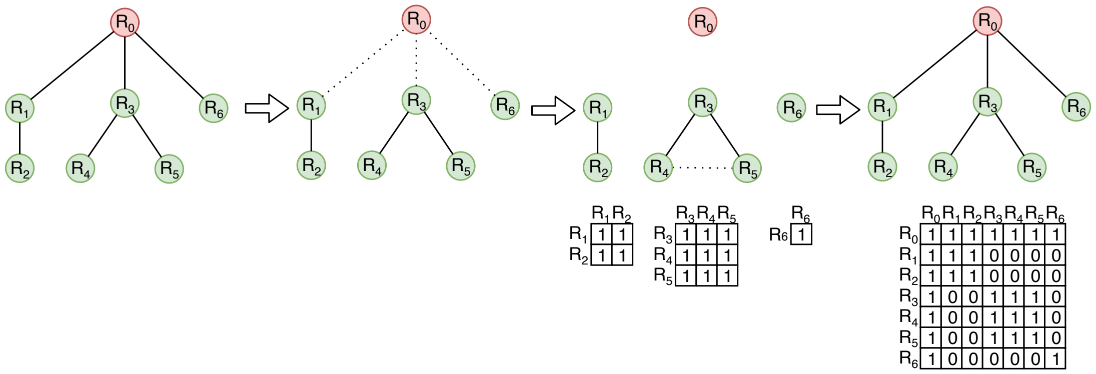


Figure 1: **Relevance matrix building:** 1) Original tweet reply tree; 2) Obtain the forest by temporarily removing the root (main tweet); 3) Consider each tree in the forest to be fully connected graphs, and obtain the relevance matrices; 4) Obtain the full Relevance matrix by putting together the matrices from the previous step and considering the main tweet to be connected to all the other tweets.

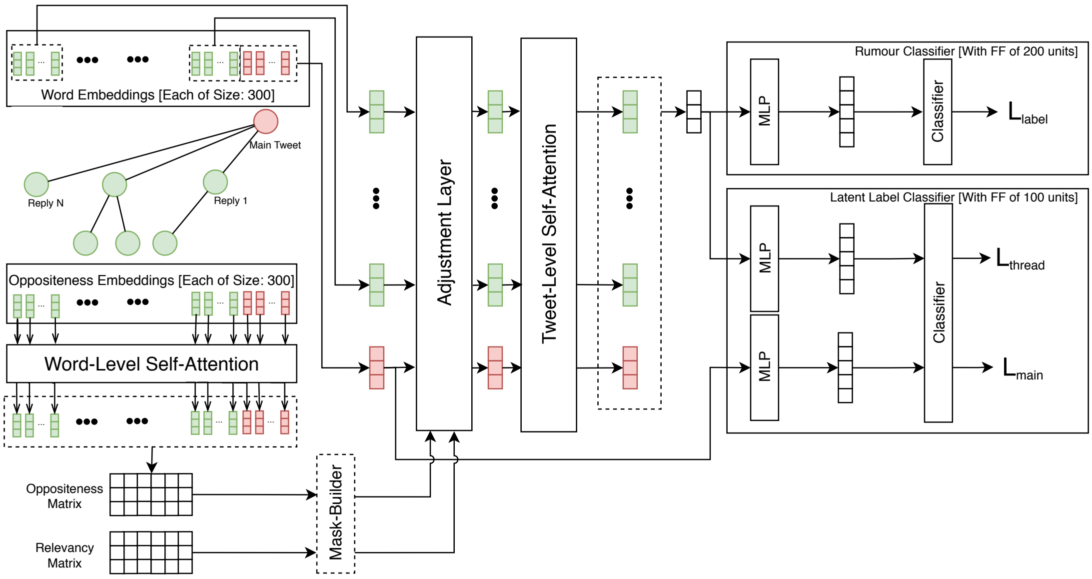


Figure 2: **The Proposed Model:** Red vectors and node represent the main (root) tweet, and green vectors and nodes represent replies. Pooling operations are shown in boxes with dashed lines.

descriptive of the thread compared to the *adjacency matrix*.

3.1 Formal Definition of Tweet Representation

Each tweet will have different number of words n; thus, we pad the short tweets with a special token, until all the tweets have the same word length N as defined by 3.

$$N = \underset{P_i \in D}{\operatorname{argmax}}(n_i) \tag{3}$$

We build the representative oppositeness list O using the oppositeness embeddings created by (de Silva and Dou, 2019) such that, for the i-th tweet R_i, with words $W_{i1}, W_{i2}, ..., W_{iN}$, the oppositeness embedding O_i is created as $o_{i1}, o_{i2}, ..., o_{im}$ where o_{ij} is the embedding of W_{ij}. Note that $m \leq N$ where all tokens might not have corresponding oppositeness embeddings.

Each word in each tweet is then converted to a representative vector by means of a set of pretrained word embeddings, such that for the i-th tweet R_i, with words $W_{i1}, W_{i2}, ..., W_{iN}$ is converted $e_{i1}, e_{i2}, ..., e_{iN}$. We then apply max-pooling operation over the word embeddings along each dimension, resulting in a representative vector h_i coupled to R_i, as shown in Equation 4. At this point, note that the tweet set I of each pair P, which used to be $I = (R_0, R_1, R_2, \dots, R_T)$, has been replaced by $I = (h_0, h_1, h_2, \dots, h_T)$. It is this new representation which is passed to the following steps.

$$h_i = Elementwise\_Max(e_{i1}, e_{i2}, ..., e_{iN}) \quad (4)$$

3.2 Similarity-Based Contextualization

As discussed earlier, the Twitter data is organized as a tree rooted at the main tweet R_0 in each instance. The earlier work by Ma et al. (2018) proved that, in rumor detection, it is helpful to capture these relations among the main tweet and the replies. The subsequent work by Veyseh et al. (2019) noted that only considering explicit reply relation between the main tweet and other tweets neglects the implicit relations among the tweets, arising from their semantic similarities (i.e., by the virtue of discussing the same topic, tweets in two separate branches may carry mutually useful information). Following this hypothesis, they exploited such implicit semantic relations for the purpose of improving the performance of the rumor detection task. However, in doing so, they abandoned the information garnered from the tree structure. In this work we propose to continue to use the implicit information, but to augment it with the information derived from the tree structure.

We follow the self-attention mechanism of (Veyseh et al., 2019), which was inspired by the transformer architecture in (Vaswani et al., 2017), to learn the pairwise similarities among tweets for capturing the semantic relations between the tweets. The process starts with calculating the key (k_i) and query (q_i) vectors for each tweet, based on its representation h_i, as shown in Equation 5. (W and b follow the traditional notation of weights and bias).

$$k_i = W_k * h_i + b_k \qquad q_i = W_q * h_i + b_q \quad (5)$$

With the key and query vectors, we calculate the similarity a_{ij} between i-th and j-th tweets, using the dot product as shown in Equation 6, where γ is a normalization factor.

$$a_{i,j} = k_i \cdot q_j / \gamma \quad (6)$$

3.3 Oppositeness-Based Contextualization

Unlike in the case of similarity vectors, which were reduced to a single dimension at this point, the oppositeness representations are still at two dimensions. Thus the self-attention of oppositeness between tweets is handled at a word level, rather than at the sentence level. We build key ($k_i^{'}$) and query ($q_i^{'}$) vectors for each word based on its representation o_i, as shown in Equation 7. (W and b follow the traditional notation of weights and bias).

$$k_i^{'} = W_k * o_i + b_k \qquad q_i^{'} = W_q * o_i + b_q \quad (7)$$

Since the oppositeness embedding of (de Silva and Dou, 2019) is based on Euclidean distance, with the key and query vectors, we calculate the oppositeness op_{i_x,j_y} between x-th word of i-th tweet and y-th word of j-th tweet using the Euclidean distance, as shown in Equation 8 where $k_{i_x}^{'}$ is the key vector for x-th word of i-th tweet, $q_{j_y}^{'}$ is the query vector for y-th word of j-th tweet, and Euclidean distance $d(,)$ is calculated across the size of the oppositeness embedding.

$$op_{i_x,j_y} = d(k_{i_x}^{'}, q_{j_y}^{'}) \quad (8)$$

To obtain the abstract tweet-level oppositeness, we apply element-wise average-pooling on the $OP_{i,j}$ matrix, as shown in Equation 9, to create the oppositeness matrix $O^{''}$, where EA is $Elementwise\_Average$ operation, δ is the oppositeness embedding count of the i-th tweet, and ϱ is the oppositeness embedding count of the j-th tweet. Note that the dimensions of the oppositeness matrix $O^{''}$ is the same as the relevance matrix M.

$$o_{i,j}^{''} = EA\left(\begin{bmatrix} op_{i_0,j_0} & op_{i_1,j_0} & \cdots & op_{i_\delta,j_0} \\ op_{i_0,j_1} & op_{i_1,j_1} & \cdots & op_{i_\delta,j_1} \\ \cdots & \cdots & \cdots & \cdots \\ op_{i_0,j_\varrho} & op_{i_1,j_\varrho} & \cdots & op_{i_\delta,j_\varrho} \end{bmatrix}\right) \quad (9)$$

Next we create the oppositeness mask Ω by average-pooling $O^{''}$ along rows and columns, as shown in Equation 10, where the definition of EA, is the same as Equation 9 and similar to Equation 3, n_i and n_j are natural lengths of the i-th and j-th tweets respectively.

$$\begin{aligned} \omega_{i,j} = 1 &- EA(o_{i,0}^{''}, o_{i,1}^{''}, ..., o_{i,n_j}^{''}) \\ &- EA(o_{0,j}^{''}, o_{1,j}^{''}, ..., o_{n_i,j}^{''}) \end{aligned} \quad (10)$$

3.4 Deriving Overall Thread Representations

Similar to the oppositeness mask Ω, we create the relevance mask Ψ by sum-pooling M along rows and columns, as shown in Equation 11, where ES, is $Elementwise\_Sum$ operation, and similar to

Equation 3, n_i and n_j are natural lengths of the i-th and j-th tweets respectively.

$$\begin{aligned}\psi_{i,j} &= ES(m_{i,0}, m_{i,1}, ..., m_{i,n_j}) \\ &+ES(m_{0,j}, m_{1,j}, ..., m_{n_i,j})\end{aligned} \tag{11}$$

At this point we diverge from (Veyseh et al., 2019) in two ways and utilize the related relevance mask M as a weighting mechanism, with proportion constant α (where $0 < \alpha < 1$), as well as the oppositeness mask Ω, to obtain augmented attention $a'_{i,j}$ as shown in Equation 12.

$$a'_{i,j} = a_{i,j}\omega_{i,j}\Big[(\psi_{i,j} - \alpha)^2 + \alpha\psi_{i,j}\Big] \tag{12}$$

We utilize the augmented similarity values $a'_{i,j}$ for each tweet pair in the thread to compute abstract representations for the tweets based on the weighted sums, as shown in Equation 13.

$$h'_i = \Sigma_j a'_{i,j} * h_j \tag{13}$$

Next, we apply the max-pooling operation over the processed tweet representation vectors h'_i to obtain the overall representation vector h' for the input pair P.

$$h' = Elementwise\_Max(h'_0, h'_1, h'_2, ..., h'_T) \tag{14}$$

Finally, the result is sent through a 2-layer feed-forward neural network capped with a softmax layer, with the objective of producing the probability distribution $P(y|R_0, R_1, R_2, \ldots, R_T; \theta)$ over the four possible labels, where θ is the model parameter. On this, we optimize the negative log-likelihood function, in order to train the model, as shown in Equation 15, where y^* is the expected (correct) label for I.

$$L_{label} = -\log P(y^*|R_0, R_1, R_2, \ldots, R_T; \theta) \tag{15}$$

3.5 Main Tweet Information Preservation

The Veyseh et al. (2019) study noted that the model by Ma et al. (2018) treats all tweets equally. This was deemed undesirable, given that the main tweet of each thread incites the conversation, and thus, arguably, carries the most important content in the conversation, which should be emphasized, to produce good performance. To achieve this end, it was proposed to bring forward the information in the main tweet independently of and separately from that of the collective twitter thread, in order to provide a check. We, in this work, also provide this sanctity check, to enhance the obtained results.

The basic idea is that, by virtue of definition, if a main tweet is a rumor (or not), unique trait and information pertaining to that class should be in the main tweet itself. Thus, the latent label (L_{thread}) obtained by processing the thread representation h' above should be the same as a potential latent label (L_{main}) obtained by processing the representation of the main tweet h_0. To calculate L_{main}, we use a 2-layer feed-forward neural network with a softmax layer in the end, where it assigns the latent labels drawn from K possible latent labels. Next, we use another 2-layer feed-forward neural network with a softmax layer in the end, assigning the same K number of possible latent labels as shown in the negative log-likelihood function to match it with the thread.

$$L_{main} = argmax_L P(L|R_0) \tag{16}$$

$$L_{thread} = -\log P'(L_{main}|R_0, \ldots, R_T) \tag{17}$$

Finally, the loss function to train the entire model is defined as in Equation 18, where the L_{label} is obtained from Equation 15, and β is a hyper-parameter which controls the contribution of the main tweet information preservation loss to final loss.

$$Loss = L_{label} + \beta L_{thread} \tag{18}$$

4 Experiments

We use the *Twitter 15* and *Twitter 16* data sets introduced by Ma et al. (2017) for the task of rumor detection. Some statistics of the data sets as given by Ma et al. (2017) are shown in Table 1. We use Glove (Pennington et al., 2014) embedding to initialize the word vectors and oppositeness embedding (de Silva and Dou, 2019) to initialize the oppositeness embeddings. Both embedding vectors are of size 300. Key and query vectors in Equations 5 and Equations 7 employ 300 hidden units. The rumor classifier feed-forward network has two layers of 200 hidden units. The feed-forward layer in the main tweet information preservation component has two layers, each with 100 hidden units, and it maps to three latent labels. The proportion constant α, which balances the explicit and implicit information, is set at 0.1. The loss function uses a trade-off parameter of $\beta = 1$, with an initial learning rate of 0.3 on the *Adagrad* optimizer. For the purpose of fair results comparison, we follow the convention of using 5-fold cross validation procedure to tune the parameters (such as node and layer counts) set by Ma et al. (2018).

Statistic	Twitter15	Twitter16
Number of NR	374	205
Number of FR	370	205
Number of TR	372	205
Number of UR	374	203
Avg. Num. of Posts/Tree	223	251
Max Num. of Posts/Tree	1,768	2,765
Min Num. of Posts/Tree	55	81

Table 1: Statistics of the Data Sets.

4.1 Comparison to the State-of-the-Art Models

We compare the proposed model against the state-of-the-art models on the same data sets. The performance is compared by means of overall accuracy and F1 score per class. We observe that there are two types of models against which we compare. The first type are the *feature-based models*, which used feature engineering to extract features for Decision Trees (Zhao et al., 2015; Castillo et al., 2011), Random Forest (Kwon et al., 2013), and SVM (Ma et al., 2015; Wu et al., 2015; Ma et al., 2017). The second type of models are *deep learning models*, which used Recurrent Neural Networks or Recursive Neural Networks to learn features for rumor detection. We compare our model to GRU-RNN proposed by Ma et al. (2016), BU-RvNN and TD-RvNN proposed by Ma et al. (2018), and Semantic Graph proposed by Veyseh et al. (2019). Results for Twitter 15 and Twitter 16 are shown in Tables 2 and 3, respectively.

It is evident from these tables that, in the rumor detection task, the deep learning models outperform feature-based models, proving that automatically learning effective features from data is superior to hand-crafting features. We also note that the *Semantic Oppositeness Graph*, along with the Semantic Graph, and other RvNN models with GRU-RNN, generally do well, which attests to the utility of structural information, be it in the form of reply structure or be it in the form of semantic relations, in helping to improve performance. We further notice that Veyseh et al. (2019) which uses implicit information, outperforms TD-RvNN (Ma et al., 2018), which only uses explicit information. *Semantic Oppositeness Graph*, which uses explicit information, implicit information, and semantic oppositeness outperforms all the other models in accuracy, while outperforming all the other models in three out of four classes, in terms of F1 Score. The one class in which *Semantic Oppositeness Graph* loses out to Veyseh et al. (2019) is in the case of the *Unrecognizable* (UR) class. We argue that this is not an issue, given that the unrecognizable class consists of tweets which were too ambiguous for human annotators to tag as one of: not a rumor (NR), false rumor (FR), or true rumor (TR). We assert that Tables 2 and 3 clearly demonstrate the effectiveness of the proposed *Semantic Oppositeness Graph* method in the task of rumor detection.

4.2 Model Stability Analysis

While comparing our system with Veyseh et al. (2019), which we use as our main baseline, we noticed that their system has a high variance in results, depending on the random weight initialization. This was impactful in such a way that in some random weight initializations, the accuracy of their system could fall as low as 24% from the reported high 70% results in their paper. Given that we use their system as our baseline and the basis for our model, we decided to do a stability analysis between their system and ours. For this purpose, we created 100 random seeds and trained four models with each seed, resulting in a total of 400 models. The models were: 1) Veyseh et al. (2019) on *twitter 15*, 2) Veyseh et al. (2019) on *twitter 16*, 3) *Semantic Oppositeness Graph* on *twitter 15*, 4) *Semantic Oppositeness Graph* on *twitter 16*. Then we normalized the results of the Veyseh et al. (2019) models to the values reported in their paper (also shown in the relevant row on Tables 2 and 3). Each result is reported in the format of (μ, σ) for the 5 fold cross-validation to explore how random weight initialization affects the two models.

From the results in Tables 4 and 5, it is evident that our *Semantic Oppositeness Graph* has higher mean values for accuracy, not a rumor (NR), false rumor (FR), and true rumor (TR), while having comparably reasonable values for Unrecognizable (UR) class. But more interesting are the standard

Model	Accuracy	F1 NR	F1 FR	F1 TR	F1 UR
DTR (Zhao et al., 2015)	0.409	0.501	0.311	0.364	0.473
DTC (Castillo et al., 2011)	0.454	0.733	0.355	0.317	0.415
RFC (Kwon et al., 2013)	0.565	0.810	0.422	0.401	0.543
SVM-TS (Ma et al., 2015)	0.544	0.796	0.472	0.404	0.483
SVM-BOW (Ma et al., 2018)	0.548	0.564	0.524	0.582	0.512
SVM-HK (Wu et al., 2015)	0.493	0.650	0.439	0.342	0.336
SVM-TK (Ma et al., 2017)	0.667	0.619	0.669	0.772	0.645
GRU-RNN (Ma et al., 2016)	0.641	0.684	0.634	0.688	0.571
BU-RvNN (Ma et al., 2018)	0.708	0.695	0.728	0.759	0.653
TD-RvNN (Ma et al., 2018)	0.723	0.682	0.758	0.821	0.654
Semantic Graph (Veyseh et al., 2019)	0.770	0.814	0.764	0.775	**0.743**
Semantic Oppositeness Graph (SOG)	**0.796**	**0.825**	**0.820**	**0.814**	0.742

Table 2: Model Performance on Twitter 15.

Model	Accuracy	F1 NR	F1 FR	F1 TR	F1 UR
DTR (Zhao et al., 2015)	0.414	0.394	0.273	0.630	0.344
DTC (Castillo et al., 2011)	0.465	0.643	0.393	0.419	0.403
RFC (Kwon et al., 2013)	0.585	0.752	0.415	0.547	0.563
SVM-TS (Ma et al., 2015)	0.574	0.755	0.420	0.571	0.526
SVM-BOW (Ma et al., 2018)	0.585	0.553	0.655	0.582	0.578
SVM-HK (Wu et al., 2015)	0.511	0.648	0.434	0.473	0.451
SVM-TK (Ma et al., 2017)	0.662	0.643	0.623	0.783	0.655
GRU-RNN (Ma et al., 2016)	0.633	0.617	0.715	0.577	0.527
BU-RvNN (Ma et al., 2018)	0.718	0.723	0.712	0.779	0.659
TD-RvNN (Ma et al., 2018)	0.737	0.662	0.743	0.835	0.708
Semantic Graph (Veyseh et al., 2019)	0.768	0.825	0.751	0.768	**0.789**
Semantic Oppositeness Graph (SOG)	**0.826**	**0.843**	**0.843**	**0.878**	0.774

Table 3: Model Performance on Twitter 16.

Model	Accuracy	F1 NR	F1 FR	F1 TR	F1 UR
Veyseh et al. (2019)	(0.770,0.138)	(0.814,0.133)	(0.764,0.198)	(0.775,0.118)	(0.743,0.129)
SOG (This work)	(0.796,0.089)	(0.825,0.080)	(0.820,0.109)	(0.814,0.093)	(0.742,0.100)

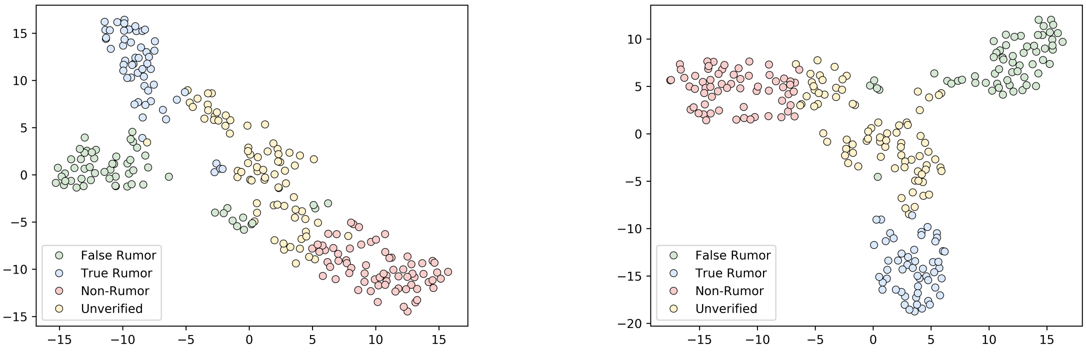
Table 4: Model Variance Performance on Twitter 15.

Model	Accuracy	F1 NR	F1 FR	F1 TR	F1 UR
Veyseh et al. (2019)	(0.768,0.103)	(0.825,0.226)	(0.751,0.103)	(0.768,0.096)	(0.789,0.184)
SOG (This work)	(0.826,0.082)	(0.843,0.153)	(0.843,0.091)	(0.878,0.074)	(0.774,0.114)

Table 5: Model Variance Performance on Twitter 16.

deviation values. It is evident that in all cases, our model has smaller standard deviation values than that of Veyseh et al. (2019). This is proof that our system is comparatively more stable in the face of random weight initialization. We argue that this stability comes from the introduction of the oppositeness component, which augments the decision-making process with the oppositeness information, as a counterpart for the already existing similarity information, preventing the predictions from having a swinging bias.

For a demonstration, consider the subset of three words *increase*, *decrease*, and *expand* from the example given by de Silva and Dou (2019). If the main tweet (R_0) were to say "*A will increase B*", R_1 replied with "*A will decrease B*", and R_2 replied



(a) Without the Oppositeness Component

(b) With the Oppositeness Component

Figure 3: t-SNE diagrams for thread representations.

with "*A will expand B*", then the purely semantic similarity based model will position R_0 and R_1 closer than R_0 and R_2, given that the word contexts in which *increase* and *decrease* are found are more similar than the word contexts in which *increase* and *expand* are found. This would result in the neural network having to learn the opposite semantics between *increase* and *expand* by itself, during the training, making it more vulnerable to issues of bad initial weight selection. This, in turn, will result in greater variance among the trained models as is the case of Veyseh et al. (2019). However, a system with an oppositeness component will already have the opposite semantics between *increase* and *decrease*, as well as *increase* and *expand* already calculated. Thus, such a system would have pre-knowledge that the word pair *increase* and *decrease*, despite being used in more common contexts, is more semantically opposite than the word pair *increase* and *expand*, which is used in less common contexts. Hence the neural network does not have to learn that information from scratch during the training, resulting in it being less vulnerable to issues of bad initial weight selection. Analogously, this, in turn, will result in lesser variance among the trained models; hence, explaining the better stability demonstrated by *Semantic Oppositeness Graph* in comparison to Veyseh et al. (2019) in Tables 4 and 5.

4.3 Impact of the Oppositeness Component

Finally, to emphasize the effect the oppositeness component has on the model, we draw the t-SNE diagrams for the final representations of the threads. Figure 3a shows the data points clustering when the model is trained without the oppositeness component, and Fig. 3b shows the data points clustering when the model is trained with the oppositeness component. Note that all other variables, including the seed for the weight initializer, are the same in the two models. These diagrams prove that the oppositeness component helps improve the separability of the classes. Specially note how the *False Rumor* and *True Rumor* classes are now more clearly separated. We postulate that this derives from the fact that the oppositeness component would help in distinguishing the continuous discord happening in a *False Rumor* thread from the subsequent general agreement in a *True Rumor* thread.

5 Conclusion

Rumors and fake news are a significant problem in social networks, due to their intrinsic nature of connecting users to millions of others and giving any individual the power to post anything. We introduced a novel method for rumor detection, based on semantic oppositeness, in this paper. We demonstrated the effectiveness of our method using data sets from Twitter. Compared to previous work, which only used explicit structures in the reply relations or semantic similarity, our model learns both explicit and implicit relations between a main tweet and its replies, by utilizing both semantic similarity and semantic oppositeness. We proved, with extensive experiments, that our proposed model achieves state-of-the-art performance, while being more resistant to the variances in performance introduced by randomness.

References

Juan Cao, Junbo Guo, Xirong Li, Zhiwei Jin, Han Guo, and Jintao Li. 2018. Automatic Rumor Detection on Microblogs: A Survey. *arXiv preprint arXiv:1807.03505*.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information Credibility on Twitter. In *WWW*, pages 675–684. ACM.

Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. *arXiv preprint arXiv:1704.05972*.

Patti Domm. 2013. False Rumor of Explosion at White House Causes Stocks to Briefly Plunge; AP Confirms Its Twitter Feed Was Hacked. *https://www.cnbc.com/id/100646197*.

Manish Gupta, Peixiang Zhao, and Jiawei Han. 2012. Evaluating event credibility on twitter. In *SDM*, pages 153–164.

Sardar Hamidian and Mona Diab. 2019a. Gwu nlp at semeval-2019 task 7: Hybrid pipeline for rumour veracity and stance classification on social media. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1115–1119.

Sardar Hamidian and Mona T Diab. 2019b. Rumor detection and classification for twitter data. *arXiv preprint arXiv:1912.08926*.

Jay J Jiang and David W Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *COLING*, pages 19–33.

Z. Jin, J. Cao, Y. Jiang, and Y. Zhang. 2014. News credibility evaluation on microblog with a hierarchical propagation model. In *ICDM*, pages 230–239.

Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, Yu Wang, and Jiebo Luo. 2017. Detection and analysis of 2016 US presidential election related rumors on twitter. In *SBP-BRiMS*, pages 14–24.

Steven Jones, M Lynne Murphy, Carita Paradis, and Caroline Willners. 2012. *Antonyms in English: Construals, constructions and canonicity*. Cambridge University Press.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *ICDM*, pages 1103–1108. IEEE.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*, pages 3818–3824.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *CIKM*, pages 1751–1754. ACM.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. In *ACL*, pages 708–717.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. In *ACL*, pages 1980–1989.

Arthur Mettinger. 1994. *Aspects of semantic opposition in English*. Oxford University Press.

Michel Paradis, Marie-Claire Goldblum, and Raouf Abidi. 1982. Alternate antagonism with paradoxical translation behavior in two bilingual aphasic patients. *Brain and language*, 15(1):55–69.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Lori Rothman and M Parker. 2009. Just-About-Right (JAR) Scales. *West Conshohocken, PA: ASTM International*.

Ulrich Schimmack. 2001. Pleasure, displeasure, and mixed feelings: Are semantic opposites mutually exclusive? *Cognition & Emotion*, 15(1):81–97.

Nisansa de Silva. 2020. *Semantic Oppositeness for Inconsistency and Disagreement Detection in Natural Language*. Phd dissertation, College of Arts and Sciences, University of Oregon. Available at `https://www.cs.uoregon.edu/Reports/PHD-202012-deSilva.pdf`.

Nisansa de Silva and Dejing Dou. 2019. Semantic oppositeness embedding using an autoencoder-based learning model. In *Database and Expert Systems Applications*, pages 159–174.

Nisansa de Silva, Dejing Dou, and Jingshan Huang. 2017. Discovering inconsistencies in PubMed abstracts through ontology-based information extraction. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 362–371. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Amir Pouran Ben Veyseh, My T. Thai, Thien Huu Nguyen, and Dejing Dou. 2019. Rumor detection in social networks via deep contextual modeling. In *ASONAM*.

Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *ICDE*, pages 651–662.

Z. Wu and M. Palmer. 1994. Verbs semantics and lexical selection. In *ACL*, pages 133–138.

Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on sina weibo. In *ACM SIGKDD Workshop on Mining Data Semantics*, pages 1–7.

Haolin Yang, Zhiwei Xu, Limin Liu, Jie Tian, and Yujun Zhang. 2020. Adaptive slide window-based feature cognition for deceptive information identification. *IEEE Access*, 8:134311–134323.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *WWW*, pages 1395–1405.

Polarized-VAE: Proximity Based Disentangled Representation Learning for Text Generation

Vikash Balasubramanian[1]*, Ivan Kobyzev[2], Hareesh Bahuleyan[2],
Ilya Shapiro[3], Olga Vechtomova[1]
[1]University of Waterloo, [2]Borealis AI, [2]University of Windsor
`{v9balasu,ovechtom}@uwaterloo.ca`
`ivan.kobyzev@borealisai.com`
`hareeshbahuleyan@gmail.com`
`ishapiro@uwindsor.ca`

Abstract

Learning disentangled representations of real-world data is a challenging open problem. Most previous methods have focused on either supervised approaches which use attribute labels or unsupervised approaches that manipulate the factorization in the latent space of models such as the variational autoencoder (VAE) by training with task-specific losses. In this work, we propose polarized-VAE, an approach that disentangles select attributes in the latent space based on proximity measures reflecting the similarity between data points with respect to these attributes. We apply our method to disentangle the semantics and syntax of sentences and carry out transfer experiments. Polarized-VAE outperforms the VAE baseline and is competitive with state-of-the-art approaches, while being more a general framework that is applicable to other attribute disentanglement tasks.

1 Introduction

Learning representations of real-world data using deep neural networks has accelerated research within a number of fields including computer vision and natural language processing (Zhang et al., 2018). Previous work has advocated for the importance of learning *disentangled representations* (Bengio et al., 2013; Tschannen et al., 2018). Although attempts have been made to formally define disentangled representations (Higgins et al., 2018), there is no widely accepted definition of disentanglement. However, the general consensus is that a disentangled representation should separate the distinct factors of variation that explain the data (Bengio et al., 2013). Intuitively, a greater level of interpretability can be achieved when independent latent units are used to encode different attributes of the data (Burgess et al., 2018).

However, recovering and separating all the distinct factors of variation in the data is a challenging problem. For real-world datasets, there may not be a way to separate each factor of variation into a single dimension in the learnt fixed-size vector representation. An easier problem would be to separate complex factors of interest into distinct subspaces of the learnt representation. For instance, a representation for text could be separated into *content* and *style* subspaces, which then enables style transfer.

Unsupervised disentanglement of underlying factors using variational autoencoders (Kingma and Welling, 2014) has been explored in previous work (Higgins et al., 2017; Kim and Mnih, 2018). However, Locatello et al. (2019) argue that completely unsupervised disentanglement of the underlying factors may be impossible without supervision or inductive biases. Disentangling textual attributes in a completely unsupervised manner has been shown to be especially difficult, but attempts have been made to leverage it for controllable text generation (Xu et al., 2020).

In this work, we propose an approach referred to as polarized-VAE[1] to disentangle the latent space into subspaces corresponding to different factors of variation. We control the relative location of representations in a particular latent subspace based on the similarity of their respective input data points according to a defined criterion (that corresponds to an attribute in the input space, e.g., syntax). This encourages similar points to be grouped together and dissimilar points to be farther away from each other in that subspace. Figuratively, we *polarize* the latent subspaces, and hence the name.

Most previous work on supervised disentanglement for text has focused on adversarial training (John et al., 2019; Yang et al., 2018). Recently, the task of disentangling textual semantics and syntax into distinct subspaces has received attention

---

[1] The code is available at `https://github.com/vikigenius/prox_vae`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 416–423
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

from researchers. For instance, Chen et al. (2019b) use a sentence VAE model with several multitask losses such as paraphrase loss and word position loss for this disentanglement task. Bao et al. (2019) incorporate adversarial training and make use of syntax trees along with specific multitask losses to disentangle semantics and syntax.

In polarized-VAE, we achieve disentanglement through distance based learning. In contrast to previous approaches, our method does not require the use of several multitask losses or adversarial training, both of which can result in optimization challenges. Furthermore, we do not need precise attribute labels, and we show that using proxy labels based on the concept of similarity is sufficient.

In summary, the main contributions of this paper are three-fold: (1) We propose a general framework for learning disentangled representations. Even though we test our method on an NLP task, the underlying concept is very general and can be applied to other domains such as computer vision; (2) We provide a method for disentanglement that does not rely on adversarial training or specialized multitask losses; (3) We demonstrate an application of our method by disentangling the latent space into subspaces corresponding to syntax and semantics. Such a setting can be used to perform controlled text decoding such as generating a paraphrase with a desired sentence structure.

2 Proposed Approach

In VAEs, a probabilistic encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is used to encode a sentence $\boldsymbol{x}$ into a latent variable $\boldsymbol{z}$, and a probabilistic decoder $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ attempts to reconstruct the original sentence $\boldsymbol{x}$ from its latent representation $\boldsymbol{z}$. The objective is to minimize the following loss function:

$$\mathcal{L}_{\text{vae}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{kl}}\mathcal{L}_{\text{kl}} \tag{1}$$

where $\mathcal{L}_{\text{rec}} = -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})]$ is the sentence reconstruction loss and $\mathcal{L}_{\text{kl}} = \mathcal{D}_{\text{kl}}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))$ is the Kullback-Leibler (KL) divergence loss. The KL term ensures that the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is close to the prior $p(\boldsymbol{z})$, which is typically assumed to be the standard normal $\mathcal{N}(\mathbf{0}, \mathbf{I})$; λ_{kl} is a hyperparameter that controls the extent of KL regularization.

The idea behind our polarized-VAE approach is to impose additional proximity regularization on the latent subspaces learnt by VAEs. Let $C = \{c_1, ..., c_k\}$ be the collection of criteria, based on which we wish to disentangle the latent space $\boldsymbol{z}$ of the VAE into k subspaces: $\boldsymbol{z} = [\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(k)}]$. Here $\boldsymbol{z}^{(i)}$ denotes the latent subspace corresponding to the criterion c_i (see Figure 1). In this paper, we focus on the case where the latent space is disentangled into semantics (c_1) and syntax (c_2), i.e., $k = 2$.

2.1 Supervision based on Similarity

We assume that we have information (possibly noisy) about pairwise similarities of the input sentences. Given a pair of sentences, the similarity information can be either a binary label (whether both the sentences belong to the same class or not) or an integer or continuous scalar variable (e.g., edit distance). In this work, the similarity criterion is a binary label:

$$\text{Sim}(\boldsymbol{x}_i, \boldsymbol{x}_j|c) = \begin{cases} 1, & \text{if } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are similar} \\ & \quad \text{w.r.t. the criterion } c \in C \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

In our case, the two criteria for disentanglement are semantics (c_1) and syntax (c_2). We use this additional information to regularize the latent space of the VAE by incorporating the proximity based loss functions, denoted as $D(\boldsymbol{z}_i^{(1)}, \boldsymbol{z}_j^{(1)}|c_1)$ and $D(\boldsymbol{z}_i^{(2)}, \boldsymbol{z}_j^{(2)}|c_2)$.

2.2 Training Method and Proximity Function

Extending the traditional VAE approach, we have a set of RNN-based encoders parameterized by ϕ_c that learn the approximate posteriors $q_{\phi_c}(\boldsymbol{z}^{(c)}|\boldsymbol{x})$. Given two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we denote the proximity of their encodings in the latent subspace by $D(q_{\phi_c}(\boldsymbol{z}^{(c)}|\boldsymbol{x}_i), q_{\phi_c}(\boldsymbol{z}^{(c)}|\boldsymbol{x}_j))$. We experiment with multiple forms of proximity functions and found cosine distance to perform the best:

$$\begin{aligned} D(q_{\phi_c}(\boldsymbol{z}|\boldsymbol{x}_i), q_{\phi_c}(\boldsymbol{z}|\boldsymbol{x}_j)) &= d_c(\boldsymbol{z}_i, \boldsymbol{z}_j) \qquad (3) \\ &= \frac{1}{2}\left(1 - \frac{\boldsymbol{z}_i \boldsymbol{z}_j}{||\boldsymbol{z}_i||||\boldsymbol{z}_j||}\right) \end{aligned}$$

Based on the above distance, we add a regularization term to the VAE loss function as follows. For each example $(\boldsymbol{x}, c)$, we have a positive sample $\boldsymbol{x}_p$ and m negative samples $\boldsymbol{x}_{n_1}, ..., \boldsymbol{x}_{n_m}$, such that $\text{Sim}(\boldsymbol{x}, \boldsymbol{x}_p|c) = 1$ and $\text{Sim}(\boldsymbol{x}, \boldsymbol{x}_{n_j}|c) = 0$; $j \in \{1, ..., m\}$:

$$\mathcal{L}_c = \max(0, 1 + d_c(\boldsymbol{z}, \boldsymbol{z}_p) - \frac{1}{m}\sum_{j=1}^{m} d_c(\boldsymbol{z}, \boldsymbol{z}_{n_j})) \tag{4}$$

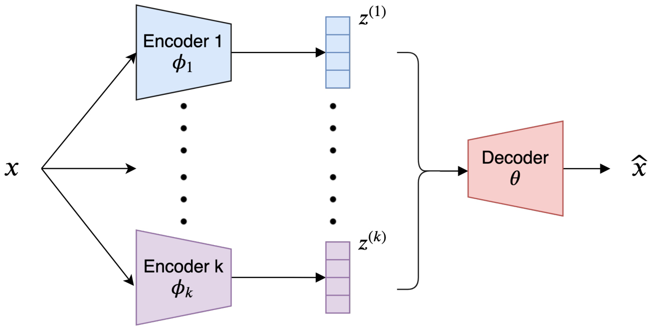


Figure 1: Model Architecture

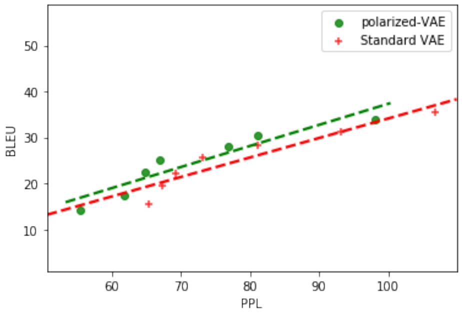


Figure 2: BLEU vs. PPL trade-off

This regularization function can be viewed as a max-margin loss over the proximity function. The final objective then becomes

$$\mathcal{L} = \mathcal{L}_{\text{vae}} + \sum_{c \in C} \lambda_c \mathcal{L}_c \tag{5}$$

3 Experiments

To demonstrate the effectiveness of polarized-VAE in obtaining disentangled representations, we carry out semantics-syntax separation of textual data, using the Stanford Natural Language Inference dataset (SNLI, Bowman et al. (2015)). Model implementation details are provided in Appendix A.

3.1 Reconstruction and Sample Quality

We evaluate our model on reconstruction and sample quality to ensure that the distance-based regularization used does not adversely impact its reconstruction or sampling capabilities. For this purpose, we compare our model and the standard VAE on two metrics: reconstruction BLEU (Papineni et al., 2002) and the Forward Perplexity (PPL)[2] (Zhao et al., 2018) of the generated sentences obtained by sampling from the model's latent space. As seen in Figure 2, there is a clear trade-off between reconstruction quality and sample quality, which is expected. Overall, polarized-VAE performs slightly better than standard VAE and this indicates that the proximity-based regularization does not inhibit the model capabilities.

3.2 Controlled Generation and Transfer

We follow previous work (Chen et al., 2019a; Bao et al., 2019) and analyze the performance of controlled generation by evaluating syntax transfer in generated text. Given two sentences, $\boldsymbol{x}_{\text{sem}}$ and $\boldsymbol{x}_{\text{syn}}$, we wish to generate a third sentence that combines the semantics of $\boldsymbol{x}_{\text{sem}}$ and the syntax of $\boldsymbol{x}_{\text{syn}}$ using the following procedure:

$$\boldsymbol{z}_{\text{sem}} \sim q_{\phi_1}(\boldsymbol{z}^{(1)} | \boldsymbol{x}_{\text{sem}}) \;\; ; \;\; \boldsymbol{z}_{\text{syn}} \sim q_{\phi_2}(\boldsymbol{z}^{(2)} | \boldsymbol{x}_{\text{syn}})$$
$$\boldsymbol{z} = [\boldsymbol{z}_{\text{sem}}, \boldsymbol{z}_{\text{syn}}] \;\; ; \;\; \boldsymbol{x} \sim p_\theta(\boldsymbol{x} | \boldsymbol{z})$$

Following the evaluation methodology of Bao et al. (2019), we measure transfer based on (1) semantic content preservation for the semantic subspace and (2) the tree edit distance (Zhang and Shasha, 1989) for the syntactic subspace.

We consider pairs of sentences from the SNLI test set for evaluation. We would like the generated sentence to be close to $\boldsymbol{x}_{\text{sem}}$ and different from $\boldsymbol{x}_{\text{syn}}$ in terms of semantics, which is measured using BLEU scores. We also report the difference to indicate the strength of transfer denoted by ΔBLEU. Additionally, we would like the generated sentence to be syntactically similar to $\boldsymbol{x}_{\text{syn}}$ and different from $\boldsymbol{x}_{\text{sem}}$, which is measured by averaged sentence-level Tree Edit Distance (TED). We also report ΔTED to indicate the strength of the syntax transfer. Finally, we use the Geometric Mean of ΔBLEU and ΔTED to report a combined score ΔGM.

Our default variant of polarized-VAE uses the entailment labels from SNLI dataset as a proxy for semantic similarity based on which positive and negative samples are chosen. For this model, we threshold the difference in TED of syntax parses as a proxy for syntactic similarity. As shown in Table 1, we also evaluate three other variants of our model. In polarized-VAE (`wo`) we use word overlap (BLEU scores) as a heuristic proxy for estimating semantic similarity, while keeping syntactic training unchanged. We also experiment with heuristics for syntax in polarized-VAE (`len`) where we use length as a heuristic proxy for syntax, while still using ground truth entailment labels for semantic training. Finally we combine these two

---

[2] PPL is computed using the KenLM toolkit (Heafield et al., 2013)

Model	**BLEU**		ΔBLEU$^\uparrow$	**TED**		ΔTED$^\uparrow$	ΔGM$^\uparrow$	**Human Eval** (%)		
	$\boldsymbol{x}_{\text{sem}}{}^\uparrow$	$\boldsymbol{x}_{\text{syn}}{}^\downarrow$		$\boldsymbol{x}_{\text{sem}}{}^\uparrow$	$\boldsymbol{x}_{\text{syn}}{}^\downarrow$			sem	syn	fluency
Standard VAE	4.75	4.67	0.08	13.70	13.60	0.10	0.28	11	11	**43**
Bao et al. (2019)	**13.74**	6.15	7.59	**16.19**	13.10	**3.08**	4.83	24	**58**	19
polarized-VAE	10.78	0.92	**9.86**	14.09	11.67	2.42	**4.88**	**65**	31	38
polarized-VAE (`wo`)	9.82	0.84	8.98	14.12	11.65	2.47	4.71	-	-	-
polarized-VAE (`len`)	10.10	**0.76**	9.34	12.68	**11.44**	1.44	3.67	-	-	-
polarized-VAE (`wo`, `len`)	9.41	0.87	8.54	12.65	11.48	1.17	3.16	-	-	-

Table 1: Syntax transfer results on SNLI. Bao et al. (2019) report TED after multiplying by 10, we report their score after correction. For each model, the human evaluation scores represent percentage of instances that it was ranked the best for a given criteria (semantics preservation/syntax transfer/fluency).

heuristics in polarized-VAE (`wo`, `len`), which can be viewed as an unsupervised variant that does not make use of any ground truth labels or syntax trees.

Our model outperforms the VAE baseline on all metrics. In comparison to (Bao et al., 2019), polarized-VAE is much better at ignoring the semantic information present in $\boldsymbol{x}_{\text{syn}}$ during syntax transfer, as evidenced by our lower BLEU scores w.r.t. $\boldsymbol{x}_{\text{syn}}$. On the other hand, we perform slightly worse on BLEU w.r.t. $\boldsymbol{x}_{\text{sem}}$. Our model does a better job at matching the syntax of sentence $\boldsymbol{x}_{\text{syn}}$ as indicated by the lower TED score w.r.t. $\boldsymbol{x}_{\text{syn}}$. Qualitative samples of syntax transfer are provided in Appendix C.

3.3 Human Evaluation

We carried out a human evaluation study for comparing outputs generated from different models. The test setup is as follows - we provide as input two sentences, x_{sem} and x_{syn} to the model; we wish to generate a sentence that combines the semantics of x_{sem} and the syntax of x_{syn}. We asked 5 human annotators to evaluate the outputs from the 3 models: baseline-VAE, polarized-VAE and the model from (Bao et al., 2019).

Each annotator was shown the input sentences (x_{sem} and x_{syn}) and the outputs from the 3 models (randomized so that the evaluator is unaware of which output corresponds to which model). They were then asked to pick the one best output for each of the following three criteria: (1) semantic preservation — level of semantic similarity with respect to x_{sem}, (2) syntactic transfer — level of syntactic similarity with respect to x_{syn} and (3) fluency. We obtained annotations on 100 test set examples from SNLI dataset. To aggregate the annotations, we used majority voting with manual tie breaking to find the best model for each test example (and for each test criteria).

For each model, we report the percentage of instances where it was voted as best for each criteria. From the human evaluation results in Table 1, we note that polarized-VAE is better at semantic transfer and worse at syntactic transfer in comparison to (Bao et al., 2019). The human evaluation results are consistent with the automatic evaluation metrics, where polarized-VAE scores higher on ΔBLEU (indicator of semantic transfer strength) and (Bao et al., 2019) is better at ΔTED (indicator of syntax transfer strength). With respect to fluency criterion, polarized-VAE ranks higher than (Bao et al., 2019). However, the most fluent sentences are produced by the baseline VAE. We hypothesise this to be due to the presence of additional regularization terms in the loss functions of both (Bao et al., 2019) and polarized-VAE, which in turn affects the fluency of their generated text (due to the deviation from the reconstruction objective).

4 Conclusion and Future Work

We proposed a general approach for disentangling latent representations into subspaces using proximity functions. Given a pair of data points, a predefined similarity criterion in the original input space determines their relative distance in the corresponding latent subspace, which is modelled via a proximity function. We apply our approach to the task of disentangling semantics and syntax in text. Our model substantially outperforms the VAE baseline and is competitive with the state-of-the-art approach while being more general as we do not use specific multitask losses or architectures to encourage preservation of semantic or syntactic information. Our methodology is orthogonal to the multitask learning approaches by Chen et al. (2019b) and Bao et al. (2019) and can be naturally combined with their methods. We would further like to investigate this approach on disentanglement applications outside of NLP. Another interesting research direction would be to further explore suitable proximity functions and identify their properties that could facilitate disentanglement.

Acknowledgements

We would like to thank Dr. Pascal Poupart for his valuable insights and ideas. We would also like to thank Compute Canada (`www.computecanada.ca`) for their support and GPU resourcees.

This Research was funded by the MITACS Accelerate program in collaboration with Borealis AI.

References

Hareesh Bahuleyan. 2018. Natural language generation with neural variational models. *CoRR*, abs/1808.09012.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xinyu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pages 632–642. Association for Computational Linguistics (ACL).

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*.

Christopher P. Burgess, Irina Higgins, Arka Pal, Loïc Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2018. Understanding disentangling in β-vae. *CoRR*, abs/1804.03599.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019a. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019b. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2453–2464.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria.

Irina Higgins, David Amos, David Pfau, Sébastien Racanière, Loïc Matthey, Danilo J. Rezende, and Alexander Lerchner. 2018. Towards a definition of disentangled representations. *CoRR*, abs/1812.02230.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Florence, Italy. Association for Computational Linguistics.

Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2649–2658, Stockholmsmässan, Stockholm Sweden. PMLR.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Reproducibility in Machine Learning, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net.

Frank Nielsen. 2020. On a generalization of the jensen-shannon divergence and the jensen-shannon centroid. *Entropy*, 22(2):221.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Tschannen, Olivier Frederic Bachem, and Mario Lučić. 2018. Recent advances in autoencoder-based representation learning. In *Bayesian Deep Learning Workshop, NeurIPS*.

Peng Xu, Yanshuai Cao, and Jackie Chi Kit Cheung. 2020. On variational learning of controllable representations for text without supervision. In *37th International Conference on Machine Learning, ICML*.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7287–7298. Curran Associates, Inc.

Kaizhong Zhang and Dennis E. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.

Qingchen Zhang, Laurence T Yang, Zhikui Chen, and Peng Li. 2018. A survey on deep learning for big data. *Information Fusion*, 42:146–157.

Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *35th International Conference on Machine Learning, ICML 2018*, pages 9405–9420. International Machine Learning Society (IMLS).

Appendix

Polarized-VAE: Proximity Based Disentangled Representation Learning for Text Generation

A Model and Training Details

Both the semantic and syntactic encoders are bidirectional LSTMs (Hochreiter and Schmidhuber, 1997) with hidden size of 128, followed by two feed-forward layers to parameterize the Gaussian mean (μ) and standard deviation (σ) parameters similar to standard VAE formulations used by (Bao et al., 2019). The latent space dimensions were taken to be $dim(z^1) = 64$ and $dim(z^2) = 16$. The decoder is a unidirectional LSTM with a hidden size of 128. We train the model for 30 epochs in total using the ADAM optimizer (Kingma and Ba, 2015) with the default parameters and a learning rate of 0.001.

We adopt the standard tricks for VAE training including dropout and KL annealing followed by (Bowman et al., 2016). We anneal both semantic and syntactic KL weights (λ_{kl}) upto 0.3 (5000 steps) using the same `sigmoid` schedule (Bahuleyan, 2018).

B Proximity Functions

We provide results for the other proximity functions that we explored for the polarized-VAE model.

Metric	ΔBLEU$^{\uparrow}$	ΔTED$^{\uparrow}$	ΔGM$^{\uparrow}$
Cosine Distance	**9.86**	**2.42**	**4.88**
Hellinger Distance	4.12	0.86	1.42
MMD	5.21	1.17	1.91
KL Divergence	4.32	0.75	1.28
JS Divergence	5.81	1.46	2.33

Table 2: Comparison of polarized-VAE with different proximity functions.

We note that since there is no closed form expression for the JS divergence between two Normal Random variables we used the generalized JS Divergence proposed by (Nielsen, 2020).

C Transfer Examples

We provide qualitative examples of our transfer experiments, where we generate a sentence with the semantics of x_{sem} and the syntactic structure of x_{syn} in Table 4. We also provide the sentences generated by a standard-VAE for comparison.

D Disentanglement of Latent Subspaces

We test if there a possibility that the two latent subspaces encode similar information. This is only likely to happen if the attributes themselves are highly correlated (e.g., if we want to disentangle syntax from length). For such cases, even existing methods based on adversarial disentanglement (John et al., 2019) may fail to completely separate out correlated information.

However, if the attributes are different enough (or ideally independent) for e.g., syntax and semantics, this is less problematic. Note that we apply our proximity loss independently to each of the subspaces (i.e., leaving the other space(s) untouched for a given input). This encourages the semantic encoder to encode semantically similar sentences close together and dissimilar ones far apart in the semantic space (same applies for the syntax encoder).

We empirically compute correlations between the semantic and syntax latent vectors for 1000 test sentences as a way to check whether the two encoders learn similar information. By feeding 1000 sentences from the test set to the Polarized-VAE, we obtain their corresponding semantic (z_{sem}) and syntax (z_{syn}) latent vectors. We then empirically compute the correlation between z_{sem} and z_{syn}. To analyze the level of similarity of information represented in z_{sem} and z_{syn}, we report the maximum absolute correlation (max across all pairs of dimensions) and also the mean absolute correlation. A higher value of correlation would indicate that there is more overlapping information learnt by the semantic and syntactic encoders. As illustrated in Table 3, the analysis indicates that the semantic and syntax latent vectors in polarized-VAE encodes less correlated information than standard-VAE (due to the proximity-based regularization). This demonstrates that the 2 latent spaces learned by our model encode sufficiently different information.

Model	Max Abs Corr$^{\downarrow}$	Mean Abs Corr$^{\downarrow}$
standard-VAE	0.62	0.1
polarized-VAE	**0.25**	**0.05**

Table 3: Maximum Absolute Correlation and Mean Absolute Correlation between the semantic and syntactic latent vectors.

$\boldsymbol{x}_{\mathrm{sem}}$	$\boldsymbol{x}_{\mathrm{syn}}$	**polarized-VAE**	**standard-VAE**
A man works near a vehicle.	A woman showing her face from something to her friend.	A man directing traffic on a bicycle to an emergency vehicle.	A woman works on a loom while sitting outside.
A family in a party preparing food and enjoying a meal.	Man reading a book.	A person enjoying food.	A man plays his guitar.
Two young boys are standing around a camera outdoors.	Three kids are on stage with a vacuum cleaner.	Two young boys are standing around a camera outdoors.	Two people are standing on a snowy hill.
There are a group of people sitting down.	They are outside.	There are people.	They are outside
a woman wearing a hat and hat is chopping coconuts with machete.	The person is in a blue shirt playing with a ball.	a woman with a hat is hanging upside down over utensils.	A girl in a pink shirt and elbow pads is swirling bubbles.
The young girl and a grownup are standing around a table , in front of a fence.	A guy stands with cane outdoors.	The young girl is outside.	The little boy is doing a show.
A person is sleeping on bed.	A man and his son are walking to the beach , looking for something.	A man and a child sit on the ground covered in bed with rocks.	A man is wearing blue jeans and a blue shirt walking.
The men and women are enjoying a waterfall.	A dog is holding an object.	The man and woman are outdoors.	The two men are working on the roof.
a man dressed in uniform.	There is a man with a horse on it.	A man dressed in black clothing works in a house.	A man dressed in black and white holding a baby.

Table 4: Examples of transferred sentences that use the semantics of x_{sem} and syntax of x_{syn}

ParaSCI: A Large Scientific Paraphrase Dataset for Longer Paraphrase Generation

Qingxiu Dong[1,3,4], **Xiaojun Wan**[1,2,3] and **Yue Cao**[1,2,3]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Center for Data Science, Peking University
[3]The MOE Key Laboratory of Computational Linguistics, Peking University
[4]College of Science, Minzu University of China
`qingxiudong@icloud.com,{wanxiaojun,yuecao}@pku.edu.cn`

Abstract

We propose ParaSCI, the first large-scale paraphrase dataset in the scientific field, including 33,981 paraphrase pairs from ACL (ParaSCI-ACL) and 316,063 pairs from arXiv (ParaSCI-arXiv). Digging into characteristics and common patterns of scientific papers, we construct this dataset though intra-paper and inter-paper methods, such as collecting citations to the same paper or aggregating definitions by scientific terms. To take advantage of sentences paraphrased partially, we put up PDBERT as a general paraphrase discovering method. The major advantages of paraphrases in ParaSCI lie in the prominent length and textual diversity, which is complementary to existing paraphrase datasets. ParaSCI obtains satisfactory results on human evaluation and downstream tasks, especially long paraphrase generation.

1 Introduction

A paraphrase is a restatement of meaning with different expressions (Bhagat and Hovy, 2013). Being very common in our daily language expressions, it can also be applied to multiple downstream tasks of natural language processing (NLP), such as generating diverse text or adding richness to a chatbot.

At present, paraphrase recognition or paraphrase generation are largely limited to the deficiency of paraphrase corpus. Especially, due to the permanent vacancy of paraphrase corpus in the scientific field, scientific paraphrase generation advances slowly. Scientific paraphrases can not only be helpful for data augmentation of challenging scientific machine translation, but is also effective for polishing scientific papers. However, existing paraphrase datasets are mainly from news, novels, or social media platforms. Most of them remain short sentences and interrogative or oral style. As a result, none of such training data can train out a scientific paraphrase generator. Taking the sentence *"we used pos tags predicted by the stanford pos tagger"* as an example, the generated sentences from Transformer (Vaswani et al., 2017) models trained on existing paraphrase datasets[1] are *"level basic topics : what is the basic purpose of stanford traditional hmo"* and *"a picture of a street sign with a sign on it"*, far from ground-truth paraphrases.

We have noticed that the structure of scientific papers is nearly fixed. Paraphrase sentence pairs appear not only within a paper (intra-paper) but also across different papers (inter-paper), which makes it possible to construct a paraphrase dataset in the scientific field. For example, repetitions of the same crucial contribution in a paper or explanations of the same term in different papers are potential paraphrases. Based on such characteristics, we design different methods to extract paraphrase pairs (shown in Section 4).

In terms of the construction methods, existing methods merely focus on the paraphrase relationship between entire sentences, while hardly handle sentences with partial paraphrase parts, leaving much original corpus idle. We find that if part of a sentence paraphrases another short sentence, such sequences will be filtered out because the overall semantic similarity is not high enough. For paraphrase discovering in this case, we fine-tune BERT to extract semantically equivalent parts of two sentences and name it PDBERT. In order to train PDBERT, we construct pseudo training data by stitching existing paraphrase sentences, and train a paraphrase extraction model using the pseudo training data. In the end, this model performs well on real scientific texts.

After filtering, we obtain 350,044 paraphrase pairs and name this dataset ParaSCI. It consists of two parts: ParaSCI-ACL (33,981 pairs) and ParaSCI-arXiv (316,063 pairs). Compared with

---

[1] Here we use Quora Question Pairs and MSCOCO, they are introduced in Section 2

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 424–434
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

other paraphrase datasets, sentences in ParaSCI are longer and more sententially divergent. ParaSCI can be used for training paraphrase generation models. Furthermore, we hope that it can be applied to enlarge training data for other NLP tasks in the scientific domain.

Our main contributions include:

1. We propose the first large-scale paraphrase dataset in the scientific field (ParaSCI), including 33,981 pairs in ParaSCI-ACL and 316,063 pairs in ParaSCI-arXiv. Our dataset has been released to the public[2].

2. We propose a general method for paraphrase discovering. By fine-tuning BERT innovatively, our PDBERT can extract paraphrase pairs from partially paraphrased sentences.

3. The model trained on ParaSCI can generate longer paraphrases, and sentences are enriched with scientific knowledge, such as terms and abbreviations.

2 Related Work

Paraphrases capture the essence of language diversity (Pavlick et al., 2015) and play significant roles in many challenging NLP tasks, such as question answering (Dong et al., 2017), semantic parsing (Su and Yan, 2017) and machine translation (Cho et al., 2014). Development in paraphrases relies heavily on the construction of paraphrase datasets.

Paraphrase Identification Datasets Dolan and Brockett (2005) proposed MSR Paraphrase Corpus [MSRP], a paraphrase dataset of 5,801 sentence pairs, by clustering news articles with an SVM classifier and human annotations. As is discovered, platforms such as Twitter also contain many paraphrase pairs. Twitter Paraphrase Corpus [PIT-2015] (Xu et al., 2015) contains 14,035 paraphrase pairs on more than 400 distinct topics. Two years later, Twitter Url Corpus [TUC] (Lan et al., 2017) was proposed as a development of PIT-2015. TUC contains 51,524 sentence pairs, collected from Twitter by linking tweets through shared URLs and do not leverage any classifier or human intervention. Datasets such as MSRP or PIT-2015 encourage a series of work in paraphrase identification (Das and Smith, 2009; Mallinson et al., 2017) but the size limitation hinders complex generation models.

Paraphrase Generation Datasets MSCOCO (Lin et al., 2014) was originally described as a large-scale object detection dataset. It contains human-annotated captions of over 120K images, and each image is associated with five captions from five different annotators. In most cases, annotators describe the most prominent object/action in an image, which makes this dataset suitable for paraphrase-related tasks. Consequently, MSCOCO makes great contribution to paraphrase generation. Quora released a new dataset[3] in January 2017, which consists of over 400K lines of potential question duplicate pairs. Wieting and Gimpel (2018) constructed ParaNMT-50M, a dataset of more than 50 million paraphrase pairs. The pairs were generated automatically by translating the non-English side of a large parallel corpus. Nowadays, MSCOCO and Quora are mainly used for paraphrase generation (Fu et al., 2019; Gupta et al., 2018). Nevertheless, their sentence lengths or related domains are restricted.

3 Dataset

3.1 Source Materials

Our ParaSCI dataset is constructed based on the following source materials:

ACL Anthology Sentence Corpus (AASC) AASC (Aizawa et al., 2018) is a corpus of natural language text extracted from scientific papers. It contains 2,339,195 sentences from 44,481 PDF-format papers from the ACL Anthology, a comprehensive scientific paper repository on computational linguistics and NLP.

ArXiv Bulk Data ArXiv[4] is an open-access repository of electronic preprints. It consists of scientific papers in the fields of mathematics, physics, astronomy, etc.. As the complete set is too large to process, we randomly select 202,125 PDF files as our original data and convert them to TXT files, arranged by sentence.

Semantic Scholar Open Research Corpus (S2ORC) S2ORC (Lo et al., 2020) is a large contextual citation graph of scientific papers from multiple scientific domains, consisting of 81.1M papers, 380.5M citation edges. We select all the citation edges of ACL and arXiv from S2ORC for subsequent processing.

---

[2] https://github.com/dqxiu/ParaSCI

[3] website:https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

[4] website:https://arxiv.org/help/bulk_data

3.2 Basic Information

According to the source materials, ParaSCI includes two subsets, ParaSCI-ACL and ParaSCI-arXiv. Paraphrase pairs in ParaSCI-ACL focus on the NLP field, while paraphrase pairs in ParaSCI-arXiv are more general. Some cases are shown in Table 1. ParaSCI show three main highlights: 1) Sentences included are long, nearly 19 words per sentence; 2) Sentences are more sententially divergent; 3) It provides rich scientific knowledge.

Name	Sentence A	Sentence B
ParaSCI-ACL	Word sense disambiguation (wsd) is the task of identifying the correct meaning of a word in context.	The process of identifying the correct meaning, or sense of a word in context, is known as word sense disambiguation (wsd).
ParaSCI-ACL	In this paper, we study the use of standard continuous representations for words to generate translation rules for infrequent phrases.	In this work, we show how simple continuous representations of phrases can be successfully used to induce translation rules for infrequent phrases.
ParaSCI-arXiv	Simon and Ronder propose a constellation model to localize parts of objects, which utilizes cnn to find the constellations of neural activation patterns.	Simon et al. propose a neural activation constellations part model to localize parts with constellation model.
ParaSCI-arXiv	Here we will concentrate only on those aspects that are directly relevant to the odderon.	We will put some emphasis on those aspects that are immediately relevant to the odderon.

Table 1: Example paraphrase pairs in ParaSCI. Sentence A and corresponding Sentence B are paraphrase pairs.

3.3 Statistic Characteristic

To assess the characteristics of ParaSCI, we compare its statistic characteristics with five main sentential paraphrase datasets in Table 2. The source genre of ParaSCI is scientific papers. Therefore, sentences are more formal and scholastic, and they differ from oral TUC or newsy MSRP. The average sentence length of ParaSCI is almost twice as long as that of ParaNMT-50M, MSCOCO and Quora, and also much longer than that of TUC. Its average length is only a little shorter than that of MSRP. As MSRP only contains 3,900 gold-standard paraphrases, our ParaSCI is complementary to the vacancy of large-scale long paraphrase pairs.

The degree of alteration is another important aspect of paraphrases. To compare our ParaSCI with other existing paraphrase datasets in this aspect, we propose to calculate the BLEU4 score (Papineni et al., 2002) between the source and target sentences of each paraphrase pair and name it **Self-BLEU**. In Table 2, ParaSCI, especially ParaSCI-ACL, shows a relatively low Self-BLEU, which means sentences are significantly changed.

4 Method

4.1 Extracting Paraphrase Candidates

Based on the unique characteristics of scientific papers, we extract the paraphrase sentences from a same paper (intra-paper) and across different papers (inter-paper). In most cases, we develop a simple but practical model to discover paraphrases in different sections effectively. For a more challenging case, when sentences are paraphrased partially rather than entirely, we propose PDBERT to collect more paraphrase candidates.

4.1.1 Intra-paper Extraction of Paraphrase Candidates

Authors usually write down the same information with transformed expressions repeatedly in different parts of the paper to emphasize critical information or echo back and forth. This kind of feature is the premise of our intra-paper extraction methods.

Sentence BERT for Paraphrase Extraction across Different Sections Noting that sentences with shared semantics appear in different parts of one paper. For instance, the following sentences are from different parts of a same paper, and they are paraphrases:

S_1: *we propose a simple yet robust stochastic answer network (SAN) that simulates multi-step reasoning in machine reading comprehension.* (***abstract***, Liu et al. (2018))

S_2: *we introduce Stochastic Answer Networks (SAN), a simple yet robust model for machine reading comprehension.* (***introduction***, Liu et al. (2018))

However, sentences in *Method*, *Data* and *Result* sections are semantically different even when they only have minor changes. For example, the strings other than numbers may be very similar when presenting the two experimental results, but the semantics are completely different. Therefore, we mainly focus on six sections (*Abstract*,

Name	Genre	Size (pairs)	Gold Size[5] (pairs)	Len	Char Len	Self-BLEU
MSRP	news	5,801	3,900	22.48	119.62	47.98
TUC	Twitter	56,787	21,287	15.55	85.10	12.53
ParaNMT-50M	Novels, laws	51,409,585	51,409,585	12.94	59.18	28.60
MSCOCO	Description	493,186	493,186	10.48	51.56	31.97
Quora	Question	404,289	149,263	11.14	52.89	29.46
ParaSCI-ACL	Scientific Papers	59,402	33,981	19.10	113.76	26.52
ParaSCI-arXiv	Scientific Papers	479,526	316,063	18.84	114.46	29.90

Table 2: Statistic characteristics of main existing paraphrase datasets and our ParaSCI. As ParaNMT-50M is too large, we sample 500,000 pairs as representatives. Len means the average number of words per sentence and Char Len represents the average number of characters per sentence. We calculate Len, Char Len and Self-BLEU of the gold-standard paraphrases rather than the whole size of sentences.

Introduction, *Background*, *Discussion*, *Preamble* and *Conclusion*). We directly obtain embeddings of sentences through BERT (Devlin et al., 2018). Then, we calculate the cosine similarity pair by pair and retain sentence pairs with a similarity score higher than 0.931 as favorable paraphrase candidates. 16,563 paraphrase candidates from ACL and 158,227 paraphrase candidates from arXiv are obtained efficiently in this way.

It is noteworthy that as information is hardly repeated in the same section, we exclude the comparison between sentences in the same section.

PDBERT for Paraphrase Discovering In fact, two sentences, even if they are not semantically equivalent, may share some parts with common semantics. For example:

S_1: ***rationales are never given during training***.

S_2: *in other words, target* ***rationales are never provided during training****; the intermediate step of rationale generation is guided only by the two desiderata discussed above.*

S_1 and the bold part of S_2 constitute a paraphrase. However, since the similarity between the entire S_1 and S_2 is only 0.88, this pair of sentences will be filtered out. This phenomenon frequently occurs when a sentence is much longer than the other, so that part of the long sentence paraphrases the short sentence. To resolve this problem, we propose a new paraphrase discovering model, PDBERT.

Our model architecture is shown in Figure 1. We use the paraphrase sentence pairs recognized based on sentence similarity (our first method mentioned above) to construct pseudo-labeled data for model training. For each genuine paraphrase pairs, we take one as Sentence A and the other as Sentence B. Then we randomly pick out two sentences from the whole sentence set, as Sentence C and Sentence D. We use Sentence A as input 1, and concatenate Sentence C, Sentence B and Sentence D, with the 80%, 50%, and 80% appearance probabilities respectively, as input 2. Input 1 and input 2 are further concatenated by adding a token [CLS] at the beginning of input 1, and inserting a [SEP] token between input 1 and input 2. The start and end positions of Sentence B in the concatenated string are recorded as the ground-truth. It is worth mentioning that while concatenating, the ending punctuations of Sentence C and Sentence B are removed. Besides, the random selection of sentences in input 2 guarantees our model to cover various situations, for example, the first part of input 2 is the ground-truth paraphrase of input 1, or input 2 is just Sentence B.

In this way, we generate a great number of training pairs. Input 1 represents a short sentence and input 2 represents a long sentence, which includes the paraphrase of the short one. We fine-tune the model to predict the start and end positions of Sentence B using a softmax function.

Although our training data are pseudo, the fine-tuned model performs well on the real data. For a given real sentence pair, we take the shorter sentence as input 1 and the longer one as input 2, and extract from the long sentence according to the predicted positions. Actually, there is a tiny probability that the entire long sentence is the paraphrase of the short sentence. We exclude such a result because we have already obtained it via sentence BERT. Here we obtain 9,915 paraphrase candidates from ACL and 45,397 pairs from arXiv.

We compare our PDBERT model with a best-clause baseline. The latter is simply splitting the

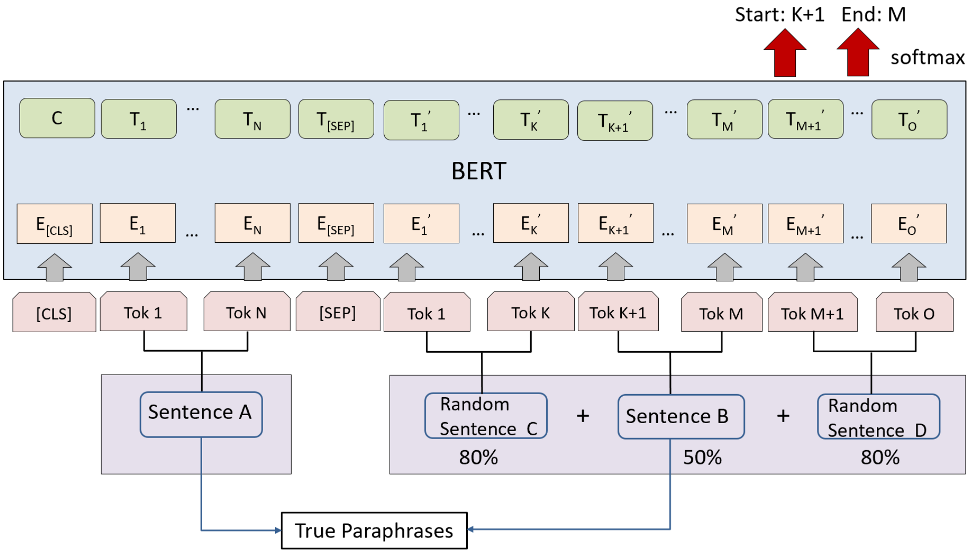


Figure 1: A general paraphrase discovering model, PDBERT. We fine-tune a 12-layer BERT model for predicting the start and end positions of the paraphrase part. The bottom of the figure shows the construction of labeled data.

long sentence into several clauses by punctuation and selecting the combination of clauses that have a maximum similarity with the short sentence. We implement PDBERT and the baseline method on the same corpus and compare the processing speed, size of valuable extractions and BERTScore(Zhang et al., 2019). Table 3 demonstrates the comparison. PDBERT has a higher extraction speed because, in the baseline method, we have to embed each possible combination of clauses. For valuable extractions, PDBERT extracts a larger size of candidate paraphrase pairs. Moreover, the BERTScore of PDBERT's results is higher, indicating its semantic advantage in paraphrase discovering.

Method	Speed	Size	BERTScore
Best-clause	7.16	341	74.06
PDBERT	32.34	387	88.23

Table 3: Comparison of paraphrase discovering methods. We divide the number of all the processed sentence pairs by time (720 minutes) as processing speed. Filtering out what can be obtained via sentence BERT, we take the number of remaining pairs as size of valuable extractions and use BERTScore (scibert-scivocab-uncased) to evaluate their quality.

4.1.2 Inter-paper Extraction of Paraphrase Candidates

Paraphrases also exist across different papers in the same field, including explanations of the same concept in different papers and citations to the same paper.

Explanations of the Same Concept In scientific papers, in order to introduce the definition of a task or a scientific terminology, the authors often explain it in one sentence. Therefore, various definitions of the same term in different papers become paraphrase candidates naturally, just as the following case:

S_1: ***Sentence compression*** *is the task of producing a shorter form of a single given sentence, so that the new form is grammatical and retains the most important information of the original one.*

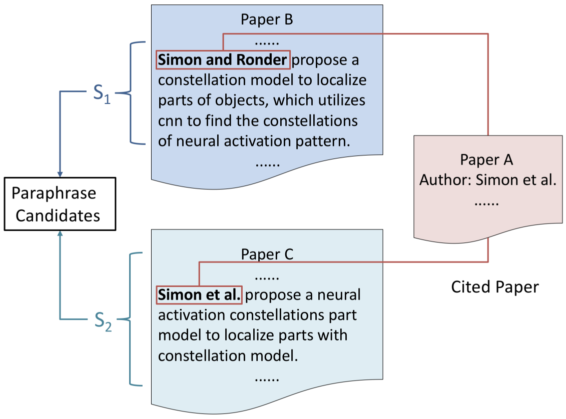
S_2: ***Sentence compression*** *is a task of creating a short grammatical sentence by removing extraneous words or phrases from an original sentence while preserving its meaning.*

In order to extract the definition sentences from different papers, we design a series of possible patterns (regular expressions) of definition sentences and tag the terms in them. In the same subject or area (provided by meta-data of source materials), the extracted sentences are aggregated according to terms, so we obtain multiple explanations of the same concept. In order to ensure the same semantics, we combine them into pairs and adopt the method in Section 4.1.1 to filter out the sentence pairs that are semantically different. Here we get 5,912 paraphrase candidates from ACL and 63,258 paraphrase candidates from arXiv.

---

[5] gold-standard paraphrase

Citations to the Same Paper Scientific papers often need to cite previous works. Besides, authors tend to give a brief introduction (i.e., citation text) to the cited paper. If different papers cite the same one, the introductory sentences to this cited paper in different papers also naturally constitute a sentence set with possible paraphrase relationships. Figure 2 provides an example.



Simon and Ronder propose a constellation model to localize parts of objects, which utilizes cnn to find the constellations of neural activation pattern.

Simon et al. propose a neural activation constellations part model to localize parts with constellation model.

Figure 2: Example of extracting paraphrase pairs from papers sharing a same cited paper.

We locate the citations in the paper through S2ORC data and extract the citation sentence and the cited article. All the extracted results are aggregated according to the same cited paper. Then we match sentences in the same group. Similarly, in order to ensure the same semantics, we use the method in Section 4.1.1 to filter out semantically inconsistent sentence pairs. In this way, we obtain 27,016 pairs of candidates for ACL and 212,644 pairs for arXiv.

4.2 Selecting High-quality Paraphrases

Due to insufficient computing resources, we have used a rough but fast filtering method to construct paraphrase candidate set as mentioned above. It includes 59,406 pairs from ACL and 479,526 pairs from arXiv. To obtain the high-quality paraphrase corpus, we implement domain-related BERTScore and paraphrase length rates to determine if a candidate pair is really paraphrase. Our two-stage construction process works in a coarse-to-fine way.

BERTScore leverages the pre-trained contextual embeddings from BERT and matches words in candidate and reference sentences by cosine similarity. It has been shown to correlate with human judgment on sentence-level and system-level evaluation. We calculate domain-related BERTScore for each paraphrase candidate, with the concrete setting of scibert-scivocab-uncased_L8_no-idf_version=0.3.3.

We design paraphrase length rate (PLR) as another filter because the numbers of words in two paraphrase sentences usually do not vary too much. PLR is simply calculated as:

$$\frac{|L_A - L_B|}{\min(L_A, L_B)}$$

L_A and L_B stand for lengths of the corresponding sentences.

For paraphrase candidates extracted from explanations of the same concept, they consist of more abstract knowledge so we set a loose restriction. We select those with a BERTScore higher than 0.6 and PLR lower than 2. Therefore, we get 4,566 definition paraphrase pairs from ACL candidates and 49,052 pairs from arXiv candidates. For paraphrase candidates extracted from other methods, we change the threshold of BERTScore to be 0.7 and PLR to be 1.0. In this way, we get another 29,415 pairs from ACL candidates and 267,106 pairs from arXiv candidates.

5 Manual Evaluation

We conduct a manual analysis of our dataset in order to quantify its semantic consistency and literal variation lexically, phrasally and sententially. We employ 12 volunteers who are proficient in English to rate the instances. Three human judgements are obtained for every sample and the final scores are averaged across different judges.

Consistency Evaluation Criterion For semantic consistency of paraphrase pairs, we design 5 degrees to distinguish. For a sentence pair to have a rating of 5, the sentences must have exactly the same meaning with all the same details. To have a rating of 4, the sentences are mostly equivalent, but some unimportant details can differ. To have a rating of 3, the sentences are roughly equivalent, with some important information missing or that differs slightly. For a rating of 2, the sentences are not equivalent, even if they share minor details. For a rating of 1, the sentences are totally different. (Examples are shown in the appendix)

Variation Evaluation Criterion For literal variation of paraphrase pairs lexically, phrasally and sententially, we use the following criterion respectively: 5 means there are more than five variations of this level, 4 means four or five, 3 means two or

Name	Lexical	Phrasal	Sentential
ParaSCI-ACL	3.82	2.73	2.01
ParaSCI-arXiv	3.67	2.68	1.48

Table 4: Overall variation of ParaSCI from manual evaluation.

three, 2 means it has only one change and 1 means no change.

Quality Control We evaluate the annotation quality of each worker using Cohen's kappa agreement (Artstein and Poesio, 2008) against the majority vote of other workers. We asked the best worker to label more data by republishing the questions done by workers with low reliability (Cohen's kappa <0.4). Finally, the average Cohen's kappa of semantic consistency evaluation is 0.71 and that of literal variation is 0.62 (0.66 lexically, 0.59 phrasally and 0.61 sententially).

Evaluation Results The average semantic consistency of ParaSCI-ACL is 4.17 and that of ParaSCI-arXiv is 3.94, both around 4, which means most paraphrase pairs are nearly equivalent, only some unimportant details may differ. Besides, the average semantic consistency of ParaSCI-ACL is higher than that of ParaSCI-arXiv. In terms of literal variation, Table 4 summarizes the annotations. ParaSCI-ACL and ParaSCI-arXiv show similar distributions. Paraphrase sentences usually change a lot lexically, because lexical variation is easier to realize. Although sentential variation scores are lower than lexical or phrasal scores, nearly one or two sentential variations for each pair are already rare and valuable for a paraphrase dataset. The long average length makes such sentential transformation possible, which is complementary to other datasets of short paraphrases.

6 Paraphrase Phenomenon Occurrence

In order to show the differences across paraphrase datasets, we sample 100 sentential paraphrases from each dataset and count occurrences of each phenomenon. Boonthum (2004) grouped common paraphrase phenomenon into 6 categories : **Synonym** (substitute a word with its synonym), **Voice** (change the voice of sentence from active to passive or vice versa), **Word-Form** (change a word into a different form), **Break** (break a long sentence down into small sentences), **Definition** (substitute a word with its definition or meaning), **Structure** (use different sentence structures to express the same thing). We report the average number of occurrences of each paraphrase type per sentence pair for each corpus in Table 5 and visualize that in Figure 3.

Name	Syn	Voice	Form	Break	Def	Struct
MSRP	0.80	0.19	0.15	0.15	0.31	0.28
TUC	0.50	0.10	0.10	0.09	0.53	0.29
ParaNMT-50M	0.87	0.15	0.20	0.13	0.40	0.25
MSCOCO	0.72	0.05	0.12	0.10	0.36	0.26
Quora	1.02	0.16	0.22	0.23	0.74	0.46
ParaSCI-ACL	0.97	0.14	0.12	0.28	0.57	0.45
ParaSCI-arXiv	1.04	0.11	0.15	0.32	0.68	0.41

Table 5: Example of extracting paraphrase pairs from papers sharing a same cited paper.

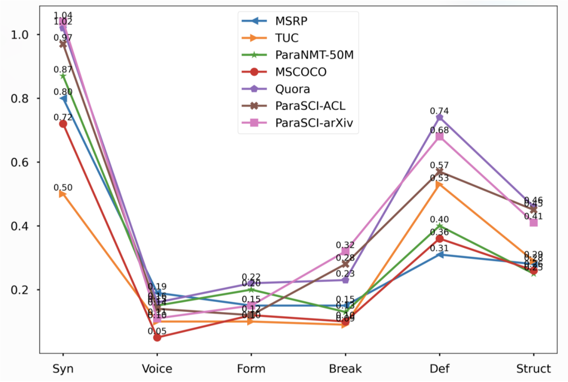


Figure 3: Visualization of Table 5.

As shown in Table 5 and Figure 3, ParaSCI-ACL, ParaSCI-arXiv and Quora share a similar paraphrase phenomenon distribution. Besides, with respect to four categories of paraphrase phenomena (Synonym, Break, Definition, Structure), they all rank top 3. It reveals that our ParaSCI and Quora datasets contain more valuable paraphrase phenomena or paraphrase patterns. However, the Quora dataset is limited to questions pairs and our ParaSCI provides declarative sentence pairs.

7 Paraphrase Generation

To demonstrate further application of ParaSCI, we train paraphrase generation models on Quora, MSCOCO and ParaSCI. We then test their generation ability on the same scientific corpus, including sentences from ACL and arXiv respectively. For paraphrase generation from ParaSCI-ACL, we use 20,388 pairs for training, 6,796 pairs for validation and 6,797 pairs for test. For paraphrase generation from ParaSCI-arXiv, we use 189,639 pairs for train-

Training	**Test**	**BLEU**	**Len**
Quora	ParaSCI-ACL	6.31	14.23
Quora	ParaSCI-arXiv	8.06	13.92
ParaSCI-ACL	ParaSCI-ACL	15.70	18.45
ParaSCI-arXiv	ParaSCI-arXiv	27.18	18.82

Table 6: BLEU scores and average lengths of scientific paraphrase generation by Transformer models trained on Quora and ParaSCI. Since the performance of the model trained on MSCOCO is rather poor (BLEU4 <1.0), we omit the comparison with it.

ing, 63,212 pairs for validation and 63,121 pairs for test. The BLEU scores and average lengths of generated sentences are shown in Table 6.

Although there are many technology-related or scientific questions on Quora, the paraphrase generation model trained on Quora still fails to perform well in the scientific field, with low BLEU scores and short average length. On the contrary, the paraphrase generation model trained on ParaSCI keeps generating longer sentences. The BLEU scores also demonstrate that the quality of the generated sentences is higher. This reflects the significant value of ParaSCI on paraphrase generation.

We show the generated paraphrases on different datasets in Table 7. To be fair, we still use the same Transformer architecture in the experiment. The generated paraphrases vary a lot. In most situations, the generated sentences from MSCOCO is incomplete, more like a phrase. The model trained on Quora only generates short questions. Whether trained on MSCOCO or Quora, the generated sentences usually share similar structures and have a large portion of entities in them. On the contrary, models trained on ParaSCI handle the paraphrase generation of longer sequence, including more modifiers and conjunctions.

Apart from that, generation models trained on ParaSCI manifest another characteristic. As ParaSCI consists of quantities of scientific terms and expressions, generation models trained on ParaSCI bring valuable scientific knowledge to the output sentence.

One thing to mention is that some abbreviations can be understood and utilized in the generation process. For instance, we generated *“we ran **mt** experiments using the moses phrase-based translation system.”* for the sentence *“we used moses as the phrase-based **machine translation** system.”*.

As we use domain-related terms and corresponding abbreviations in scientific texts frequently, this advantage can add conciseness, technicality and naturality to the generated sentences.

Another thing to mention is that some common sense or scientific knowledge is also taken into paraphrase generation. For example, we input *“the penn discourse treebank is the largest corpus richly annotated with explicit and implicit discourse relations and their senses.”* as the original sentence. It introduces the Penn Discourse Treebank (Miltsakaki et al., 2004) without any information related to its size and source, but the information is added to the generated sentences: *“the penn discourse treebank is the largest available annotated corpora of discourse relations **over 2,312 wall street journal articles**.”*

These cases reveal different aspects of ParaSCI's advantages in paraphrase generation. In further work, we hope that ParaSCI will contribute more to scientific paraphrase generation and subsequently, be applied to more downstream tasks in the scientific field.

Name	**Original**	**Paraphrase**
MSCOCO	a group of people watch a dog ride a motorcycle.	an old photo of people riding on a motorcycle and waving.
Quora	how can i get saved wifi password?	how can i see a saved wifi password?
ParaSCI-ACL	relation extraction (re) is the task of determining semantic relations between entities mentioned in text.	relation extraction (re) is the task of recognizing the assertion of a particular relationship between two or more entities in text.
ParaSCI-arXiv	cosmic strings are linear topological defects that can form in the early universe as a result of symmetry-breaking phase transitions.	cosmic strings are one-dimensional massive objects, which may appear as topological defects at the spontaneous symmetry breaking in the early universe.

Table 7: Example paraphrase sentences generated by the same Transformer model trained on different datasets. Different from Table 6, models here are trained with in-domain data, so the training data and testing data come from the same field.

8 Conclusion and Future Work

In this paper, we describe the characteristics and construction process of ParaSCI, a large-scale paraphrase dataset in the scientific field. It shows favorable results in the either automatic or manual evaluation.

For future work, although we filter out more than 200 thousand paraphrase candidates to promise the quality of ParaSCI, most candidates include valuable paraphrase patterns lexically or phrasally. Therefore, more paraphrase patterns are remaining to be discovered. Similarly, compared to the bulk data on arXiv or other scientific websites, we only use the tip of an iceberg to construct this dataset, and we are expecting to implement the methods in other scientific domains. For instance, we can obtain a biomedical paraphrase dataset from PubMed.

We hope that ParaSCI can be used to augment training data for various NLP tasks, such as machine translation in scientific field, and make more contributions to the development of NLP.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61772036), Beijing Academy of Artificial Intelligence (BAAI) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Akiko Aizawa, Takeshi Sagara, Kenichi Iwatsuki, and Goran Topic. 2018. Construction of a new acl anthology corpus for deeper analysis of scientific papers. In *Third International Workshop on SCIentific DOCument Analysis (SCIDOCA-2018), Nov. 2018.*

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Chutima Boonthum. 2004. iSTART: Paraphrase recognition. In *Proceedings of the ACL Student Research Workshop*, pages 31–36, Barcelona, Spain. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Dipanjan Das and Noah A Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 468–476. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. *arXiv preprint arXiv:1708.06022*.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems*, pages 13623–13634.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of The 2017 Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pages 1235–1245. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1694–1704, Melbourne, Australia. Association for Computational Linguistics.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S. Weld. 2020. S2ORC: The Semantic Scholar Open Research Corpus. In *Proceedings of ACL*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893.

Eleni Miltsakaki, Rashmi Prasad, Aravind K Joshi, and Bonnie L Webber. 2004. The penn discourse treebank. In *LREC*.

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

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

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. *arXiv preprint arXiv:1704.05974*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Bill Dolan. 2015. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (pit). In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 1–11.

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

A Appendices

Examples of semantic consistency evaluation are shown in the following table.

Sentence A	**Sentence B**	**Score**
Task-oriented dialog systems help users to achieve specific goals with natural language.	We use a set of 318 English function words from the scikit-learn package.	1
End-to-end task-oriented dialog systems usually suffer from the challenge of incorporating knowledge bases.	Task-oriented dialog systems help users to achieve specific goals with natural language.	2
Opinion mining has recently received considerable attentions.	Analysis has received much attention in recent years.	3
We evaluated all agents on 57 Atari 2600 games from the arcade learning environment.	We evaluated EMDQN on the benchmark suite of 57 Atari 2600 games from the arcade learning environment.	4
Here we will concentrate only on those aspects that are directly relevant to the odderon.	We will put some emphasis on those aspects that are immediately relevant to the odderon.	5

Table 8: Examples of semantic consistency evaluation.

Discourse Understanding and Factual Consistency in Abstractive Summarization

Saadia Gabriel[♠] **Antoine Bosselut**[♦] **Jeff Da**[◇] **Ari Holtzman**[♠]
Jan Buys[♡] **Kyle Lo**[◇] **Asli Celikyilmaz**[♣] **Yejin Choi**[♠◇]
[♠]Paul G. Allen School of Computer Science & Engineering, University of Washington
[♣]Microsoft Research [◇]Allen Institute for Artificial Intelligence
[♦]Stanford University [♡]University of Cape Town
`{skgabrie,ahai,yejin}@cs.washington.edu, {jeffd,kylel}@allenai.org,`
`antoineb@cs.stanford.edu, jbuys@cs.uct.ac.za, aslicel@microsoft.com`

Abstract

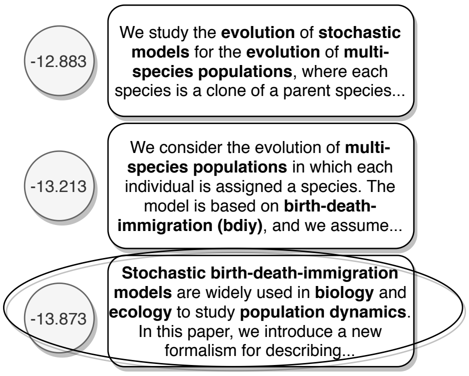
We introduce a general framework for abstractive summarization with factual consistency and distinct modeling of the narrative flow in an output summary. Our work addresses current limitations of models for abstractive summarization that often hallucinate information or generate summaries with coherence issues.

To generate abstractive summaries with *factual consistency* and *narrative flow*, we propose *Cooperative Generator – Discriminator Networks* (Co-opNet), a novel transformer-based framework where a generator works with a discriminator architecture to compose coherent long-form summaries. We explore four different discriminator objectives which each capture a different aspect of coherence, including whether salient spans of generated abstracts are hallucinated or appear in the input context, and the likelihood of sentence adjacency in generated abstracts.

We measure the ability of Co-opNet to learn these objectives with arXiv scientific papers, using the abstracts as a proxy for gold long-form scientific article summaries. Empirical results from automatic and human evaluations demonstrate that Co-opNet learns to summarize with considerably improved global coherence compared to competitive baselines.

1 Introduction

Generating summaries with coherent discourse structure and domain knowledge awareness poses a challenge for current methods in summarization. Generative models can commonly produce high quality text (Figure 1), but fail to understand finer-grained details of coherence such as the structure and flow of a narrative. In addition, they often generate factually incorrect content. Prior work on factuality in abstractive summarization has found that current models can hallucinate information more than 70% of the time when generating summaries of news articles (Maynez et al., 2020).



We study the **evolution** of **stochastic models** for the **evolution** of **multi-species populations**, where each species is a clone of a parent species...

We consider the evolution of **multi-species populations** in which each individual is assigned a species. The model is based on **birth-death-immigration (bdiy)**, and we assume...

Stochastic birth-death-immigration models are widely used in **biology** and **ecology** to study **population dynamics**. In this paper, we introduce a new formalism for describing...

Figure 1: Generated abstracts for a biology article (from the **Bio** subset of our arXiv dataset). Abstracts are ranked from most (top) to least likely (bottom) using the generator model. Abstracts with better narrative structure and domain-specific content (such as the circled abstract) are often out-ranked in terms of likelihood by abstracts with factual errors and less structure.

To address these issues, we focus our study on generating abstractive summaries with *factuality* and *narrative flow*. Given an input document, the goal is to generate a paragraph-length abstractive summary with proper discourse structure that contains factually correct claims. Our study builds on and extends previous work that focuses on either *extractive document-level* summarization (Nenkova and McKeown, 2012; Allahyari et al., 2017) or *abstractive sentence-level* summarization (Rush et al., 2015; Grusky et al., 2019; Narayan et al., 2018).

In pursuit of this goal, we introduce *Cooperative Generator-Discriminator Networks* (Co-opNet), a framework for abstractive summarization that considers subtle aspects of fact-checking and discourse necessary for coherent text generation. In this framework, the generator, a transformer language model fine-tuned for abstractive summarization, proposes a pool of candidate summaries (§2). The

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 435–447
April 19 - 23, 2021.

discriminator, also transformer-based, scores the factuality or discourse quality of candidate summaries using one of four different objectives: the overlap between a scientific article introduction and predicted fact-checking evidence spans in generated summaries, the ordering of predicted discourse roles, the coverage of predicted discourse roles, or the likelihood of adjacency between generated sentences (§3). The best summary is chosen cooperatively by combining the generator and discriminator scores (§4).

Most previous works on *abstractive document-level* summarization have difficulty in directly modeling or evaluating *narrative flow* and *factuality* in generated summaries. This weakness is largely due to the inherent limitations of existing datasets, such as the CNN/DailyMail dataset (Hermann et al., 2015). The reference summaries available in these commonly used resources are mainly headlines of news articles or stories. As a result, they are often sets of disconnected sentences that are highly extractive, leading to models that are also extractive (Hoang et al., 2019), rather than abstractive.

In order to address these data challenges, we test our summarization model on a set of arXiv scientific papers. Scientific abstracts are ideal for modeling narrative flow as they are structured with highly coherent discourse flow. They also maintain implicit *abstractive* alignments with respect to the introduction of the article – in contrast to the tight, *extractive* alignments of current models. Scientific article summarization is also a task where factuality is more well-defined than in other domains like story summarization which leave more room for interpretation.

Comprehensive empirical results considering both automatic and human evaluations demonstrate that Co-opNet learns to summarize scientific articles from three domains with considerably improved global coherence compared to competitive baselines (§6). We also demonstrate that the framework is generalizable to multiple coherence objectives, and effective at generating scientific abstracts that are more factually consistent.

2 Generator Networks

We use the transformer architecture of Radford et al. (2019) as our generator's architecture. Following the work of Liu et al. (2018), we adapt a language model to the task of abstractive summarization by concatenating the article a, a delimiter token $[\text{SEP}]$, the summary s, and an end token $[\text{END}]$ into one input vector $X = (a_1, ..., a_{|a|}, [\text{SEP}], s_1, ..., s_{|s|}, [\text{END}])$, where $|a|$ is the length of the gold article and $|s|$ is the length of the gold summary.

At each time step i, the model produces an output probability distribution over the vocabulary for the next token w_i given all previous output tokens $w_{<i}$. For any arbitrary token w_j preceding w_i, the per-layer representation of that token is computed in the following way:

$$\mathbf{h}_j^0 = \mathbf{W}_e(w_j) + \mathbf{p}_j \tag{1}$$

$$\mathbf{h}_j^l = block(\{\mathbf{h}\}_{<j}^{l-1}) \tag{2}$$

where *block* refers to each transformer block composed of multi-headed attention, a feedforward network and layer normalization, $\mathbf{W}_e$ is a word embedding matrix, $\mathbf{p}_j$ is the position embedding, $\mathbf{h}_j^0$ is the initial representation, $\{\mathbf{h}\}_j^l$ is the block output for an arbitrary layer l, and $\{\mathbf{h}\}_{<j}^{l-1}$ is the set of all block outputs from the preceding layer for positions up to j. Finally, for the current position i in the sequence, we compute a distribution over the output vocabulary as follows:

$$P(w_i|w_0, ...w_{i-1}) = \text{softmax}(\mathbf{h}_{i-1}^L \mathbf{W}_e) \tag{3}$$

where $\mathbf{W}_e$ is the same embedding matrix as in Equation 1 and $\mathbf{h}_{i-1}^L$ is the final layer transformer block output.

3 Discriminator Networks

Because summarization models are prone to narrative flow and factual consistency issues (Kryściński et al., 2020; Xu et al., 2020), we use a discriminator to score generated summaries for discourse and factuality properties. Due to the challenge of explicitly defining discourse and factuality properties as scores, these properties are approximated using parameterized scoring functions.

These scoring functions determine if generated text demonstrates discourse and factuality properties in three ways: (1) predicting the discourse role of sentences within a full summary, (2) predicting the likelihood of adjacency given a sentence pair, and (3) measuring the presence of salient facts in the generated summary from the original input context. While our discriminators focus on these three properties, we note that this framework is generalizable and could be extended to include other

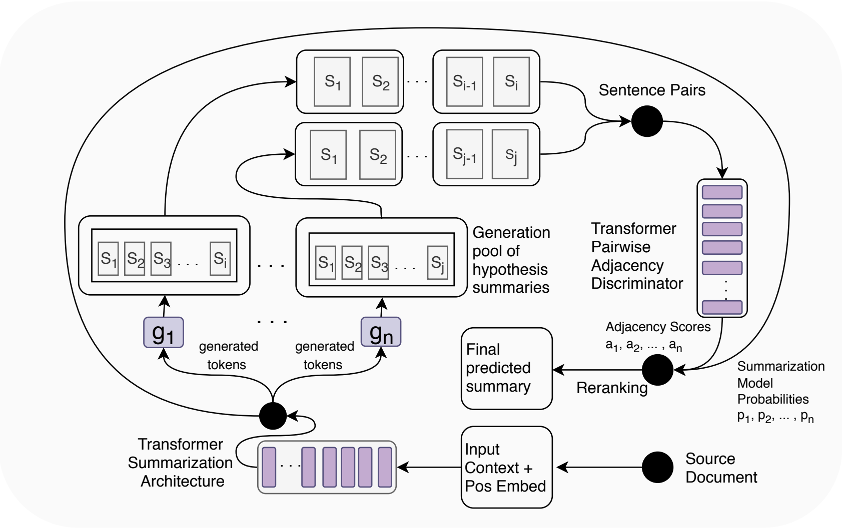


Figure 2: Model architecture for adjacency reranking variation of Co-opNet

discriminator models that encourage different communicative norms associated with high-quality language generation.

3.1 Discourse

We explore different discriminator architectures as additional discourse scoring functions during the generator's decoding process. For these discriminators, we generally score discourse in two ways. First, we use inferred sentence-level scientific abstract discourse role labels[1] defined by Cohan et al. (2019) and predict them using a sequence classifier[2] based on SciBERT (Beltagy et al., 2019). Using these predictions, we score the discourse properties of the abstract relative to their coverage (§3.1.1) or ordering (§3.1.2). Second, we learn a function that can score the likelihood that sentences within generated abstracts should be adjacent to one another (§3.1.3).

3.1.1 Coverage

We measure the completeness of the narrative structure within a scientific abstract by defining the following coverage score:

$$\mathcal{L}_{cov} = \log(D_{abs}/D_{all}), \tag{4}$$

where D_{abs} is the number of unique discourse roles appearing in an abstract and D_{all} is the total number of possible discourse roles. This objective allows us to penalize abstracts that are missing discourse roles. For example, an abstract that fails to mention anything about the results of the study would be penalized.

S_{i-1}	S_i
BACKGROUND	BACKGROUND
BACKGROUND ∨ METHOD ∨ OBJECTIVE	METHOD
BACKGROUND ∨ OBJECTIVE ∨ METHOD	OBJECTIVE
OBJECTIVE ∨ METHOD ∨ OTHER	RESULT

Table 1: Discourse Role Ordering

3.1.2 Ordering

We also score the order in which discourse labels appear in generated abstracts. In Table 1, we hard-code valid orderings of discourse labels for generated sentences based on each of the abstract discourse roles of Cohan et al. (2019). If the ordering for two adjacent sentences in the abstract $O(s_{i-1}, s_i)$ is valid, the score for the ordering is 1 (-1 otherwise). We sum the scores for all the orderings within a particular abstract and normalize between 0 and 1 (as described by f_n):[3]

$$\mathcal{L}_{order} = \log(f_n(\sum_{i=1}^{S} O(s_{i-1}, s_i))) \tag{5}$$

We also impose a rule for s_1='BACKGROUND' and a rule for s_S='RESULT' to encourage more natural orderings.

---

[1] The labels are {BACKGROUND, METHOD, OBJECTIVE, RESULT, OTHER}.

[2] See (Cohan et al., 2019) for model and training details.

[3] See the Appendix for a more detailed description of the f_n function.

3.1.3 Adjacency Classification

To model the likelihood of adjacency between two sentences s_u and s_v, we first compute a hidden representation of the sentence pair using SciBERT (Beltagy et al., 2019). The encoder input is the concatenation of the sentences: $\mathbf{s} = [\mathrm{CLS}] + s_u + [\mathrm{SEP}] + s_v + [\mathrm{SEP}]$, where $[\mathrm{CLS}]$ is a special token associated with the task and $[\mathrm{SEP}]$ is a sentence delimiter token. Each word in the sequence is encoded by a word embedding w_i and positional embedding p_i and passed through the SciBERT model to yield $\mathbf{h}_{cls}$, the output state at the position of the $[\mathrm{CLS}]$ token. We then obtain the probability of adjacency between the sentences by a linear projection of $\mathbf{h}_{cls}$ followed by a sigmoid activation:

$$P_{adj}(\mathbf{s}) = \sigma(\mathbf{w}_{disc}^{\top}\mathbf{h}_{cls}) \quad (6)$$

We define the training objective for the adjacency discriminator to minimize the negative log likelihood of predicting whether two sentences are adjacent or not:

$$\mathcal{L}_{disc} = -\Big(\delta_{adj}(\mathbf{s}) \cdot \log(P_{adj}(\mathbf{s})) + (1 - \delta_{adj}(\mathbf{s})) \cdot \log(1 - P_{adj}(\mathbf{s}))\Big), \quad (7)$$

where $\delta_{adj}(\mathbf{s})$ is an indicator function for whether the two sentences in $\mathbf{s}$ are adjacent. We note that while the discourse discriminators mainly focus on narrative structure, they may also capture context-aware aspects of factuality and content selection.

3.2 Factuality and Faithfulness

To measure factuality of generated summaries, we predict which tokens in the summary are likely to belong to a fact-checking evidence span (i.e., a span of the text used to prove a scientific claim using a finetuned BERT token classification model.[4] Recent work has shown that inspecting attention weights alone is not necessarily a reliable metric for determining saliency of particular aspects in the input context to the output of neural models (Serrano and Smith, 2019). The saliency weights representing the likelihood of tokens belonging to evidence spans provides us with a more explicit representation of factual importance.

We obtain proxy saliency labels for the importance of a particular token t appearing in an abstract using a BERT model trained on evidence spans annotated for scientific fact-checking (Wadden et al., 2020). Specifically, if t is not a stopword and $t \in E$, where E is an evidence span used to check a scientific claim, then we assign a label of 1 to t. Otherwise, the label for t is 0. Examples of extracted spans are given in table 2.

Topic	Spans
NLP	existing semantic schema, annotation effort, music knowledge representation, siri assistant
BIO	biological system, ptotic, cybernetics entropy , shannon established fundamental limits

Table 2: Salient spans extracted using factuality discriminator.

We compare the predicted evidence spans against information presented in the original introduction to capture the degree to which generative models are hallucinating information.

Factuality Objective At inference time, we compare the extracted salient spans, $F(g)$, of the generated summary g against the set of all ngrams in the article input context, $N(a)$, measuring the degree to which salient spans are hallucinated:

$$\mathcal{L}_{fact} = \log\Big(\frac{|\{f | f \in F(g), f \in N(a)\}|}{|F(g)|}\Big) \quad (8)$$

4 Reranking with Discourse and Factuality Experts

To incorporate the discriminator objective into our summarization framework, we first generate a pool of candidate summaries from the base summarization model (§2) using any decoding strategy (e.g., beam search or top-k sampling). Then, the discriminator is used to re-rank these candidates in conjunction with the original token-level generator scores. For example, in the case of the adjacency discriminator, we maximize the generator token-level probability of a candidate summary g, and the average of adjacency scores for the set of sentences composing g (denoted $S(g)$) – i.e., the probability of each sentence s_u being adjacent to the previous sentence s_{u-1} in $S(g)$:

$$\mathrm{score}(g) = \lambda_{gen}\frac{1}{|g|}\sum_{i=1}^{|g|}\log P(w_i | w_1, ...w_{i-1}) + \lambda_{disc}\frac{1}{|S(g)| - 1}\sum_{u=2}^{|S(g)|}\log P_{adj}(s_u, s_{u-1}), \quad (9)$$

---

[4] See Appendix A.4 for details of token classification model.

Split	**CS**	**BIO**	**AAN**
Train	44900	4104	10106
Validation	5622	555	892
Test	5670	522	892

Table 3: Domain subset sizes

where λ_{gen} and λ_{disc} are hyper-parameters controlling the contribution of the generator and adjacency discriminator to the final predicted summary. The same procedure is followed for the other discourse and factuality objectives, replacing $P_{adj}(s_u, s_{u-1})$ with the scores from these discriminators.

5 Data

5.1 Datasets

Since the focus of this work is on generating summaries with more coherent narrative flow and greater factual consistency, we concentrate on datasets requiring discourse structure to generate good summaries. Particular attributes of the discourse structure of these datasets include:

- Length of summaries → Are the summaries long enough to clearly show narrative flow properties and factual correctness?
- Abstractiveness of gold summaries → Do the summaries exhibit particular sentence-level flow, or are the summary sentences extracted highlights from the context?

ArXiv We crawled over 700K samples (472K abstracts) from scientific articles on `arxiv.org`. In our experiments we primarily focus on the CS[5] and Bio[6] domain subsets. The task we define is to generate an abstract given a introduction, which presents a challenge to existing summarization models. This task also requires models to learn relevant domain knowledge for the scientific domain of interest and recognize common discourse structure for papers written in that domain.

AAN Additionally, we include an existing dataset of scientific articles that focuses on papers in the NLP computer science domain. This dataset consists of a 12k paper subset from the ACL Anthology Network (AAN; Radev et al., 2009) with extracted introduction and abstract pairs.

Scientific abstracts in ArXiv and AAN have properties that are missing from existing summarization datasets based on Newswire data. For example, XSum (Narayan et al., 2018) and Newsroom (Grusky et al., 2019) summaries are generally too short to exhibit cross-sentence narrative flow. Meanwhile, CNN/DailyMail (Hermann et al., 2015) summaries are acquired by concatenating extracted highlights, which can be unrelated. Conversely, ArXiv and AAN abstracts are long enough to have multiple sentences,[7] and generally exhibit strong discourse patterns typical to scientific writing, making them ideal corpora for assessing discourse understanding in abstractive summarization. Table 3 provides details of dataset splits.

6 Experimental Setup

Our implementation is based on the Huggingface implementation[8] of the BERT (Devlin et al., 2019) and GPT-2 language models (Radford et al., 2019).

Generator We perform WordPiece tokenization for the input context and output summaries. Because of the fixed input size of the transformer language model, the input context is truncated to a maximum of 800 tokens, and summaries are truncated to a maximum of 200 tokens. We use a learning rate of 2e-5 and a batch size of 16 to finetune the generator. We train the base summarization transformer model for 12 epochs. All experiments are run on either a Titan-X or Quadro RTX 8000 GPU. Training time for the AAN and ArXiv Bio datasets is about 30 minutes per epoch. Training time for the ArXiv CS dataset is 2.5 hours per epoch. In our experiments we use top-k sampling with k=4 (Fan et al., 2018) to generate candidate summaries for each model.

Discriminator At training time we use a maximum sentence length of 200 tokens to accommodate the fixed input size of BERT (512 tokens), reduce inference time, and discourage the model from generating abnormally long run-on sentences that indicate the presence of coherence issues.[9]

For the adjacency discourse models, we fine-tune the discriminator using a learning rate of 2e-5, a linear warmup learning rate schedule, and a batch size

---

[5] `https://arxiv.org/corr`

[6] `https://arxiv.org/archive/q-bio`

[7] See Appendix A.6 for comparison of datasets.

[8] `https://github.com/huggingface/transformers`

[9] See the original papers for details of training the SciFact and abstract discourse models.

Model	AAN			CS			Bio		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Lede-3	27.12	6.62	23.88	28.22	7.06	16.22	27.60	5.70	24.21
LexRank	36.03	10.14	31.37	36.53	10.41	32.09	35.32	**8.84**	30.76
LSTM	27.80	5.57	18.02	22.74	4.56	20.64	10.73	0.49	9.94
PGen	39.85	12.83	23.24	36.68	**11.74**	32.55	23.74	4.48	21.65
Generator (Our work)	41.31	**12.97**	37.05	38.01	10.95	34.46	34.86	8.45	31.38
Co-opNet (Our work)	**41.67**	12.65	**37.23**	**38.57**	10.81	**35.11**	**35.86**	8.41	**32.56**

Table 4: Automatic Evaluation of generative architectures and Co-opNet. For AAN, we provide results using the Factuality discriminator. For CS and Bio, we provide results using the Coverage discriminator.

Model	BERTScore	SciBERTScore
PGen	57.86	59.13
Generator	61.71	62.80
Co-opNet (Adj)	61.87	63.10
Co-opNet (Fact)	**62.09**	**63.21**

Table 5: BERTScore results on AAN subset (F1)

of 32. All adjacency discourse discriminator models are fine-tuned for 2 epochs on a Titan-X GPU. The adjacency discriminator models are adapted from the Huggingface implementation of the BERT next sentence prediction classifier. We initialize the 12-layer BERT-base discriminator model with the pretrained weights of the SciBERT-uncased model, which was originally trained on 1.14 million scientific papers (Beltagy et al., 2019). Two discriminators are trained: one is fine-tuned on AAN for decoding both ArXiv CS and AAN, while the other discriminator is fine-tuned on ArXiv Bio and used exclusively for decoding that subset. We weigh the generation and discriminator models equally when decoding by setting λ_{gen}=λ_{disc}=.5. Additional implementation details are provided in Appendices A.3 and A.4.[10]

7 Experiments

We compare against extractive approaches using the Lede-3 and LexRank (Erkan and Radev, 2004) baselines. We also compare against two abstractive approaches: a 2-layer bi-LSTM sequence-to-sequence model with attention (LSTM), and a pointer-generator model (PGen; See et al., 2017). Training details of the supervised baselines can be found in the Appendix A.2. In addition, we compare to a subset of our approach that only uses the generator to produce summaries, rather than the full framework.

7.1 Automatic Evaluation

Following previous work on summarization, we use the ROUGE metric (Lin, 2004) for automatic evaluation of generative models and Co-opNet. Specifically, we report ROUGE-1, ROUGE-2 and ROUGE-L F1 scores. To capture similarity in contextual meaning, we look at BERTScore F1 (Zhang et al., 2020a), which has been shown to more closely correlate with human judgements than other generation metrics.

Results on the AAN, CS and Bio subsets of ArXiv are shown in Table 4. Co-opNet outperforms all baselines on ROUGE-1 and ROUGE-L by a consistent margin. Notably, Co-opNet's performance is superior to the generator-only model, illustrating the importance of the discriminators for generating more coherent summaries. Interestingly, on the more domain-specific AAN subset, our model is over 12% better on ROUGE-L compared to the PGen baseline and 5.86% better than the best extractive model. Our model also outperforms the strongest baselines on BERTScore.

When we break down results for various Co-opNet architectures (see Table 6), we find that the factuality and discourse role discriminators lead to the best performance in terms of ROUGE scores with the adjacency discriminator achieving lower performance on ROUGE than the base generator. However, as shown by Table 5, the adjacency discriminator outperforms the base generator when we consider BERTScore, a more contextual evaluation metric, indicating that this generator-discriminator combination selects summaries that capture the same linguistic patterns and meaning as reference summaries without directly copying.

7.2 Human Evaluation

Since coherence of generated text is difficult to measure with automatic metrics (Kilickaya et al., 2017; Sun et al., 2019; Clark et al., 2019), we

---

[10] Our code/data is released here: `https://github.com/skgabriel/coopnet`.

Model	AAN R-1	AAN R-2	AAN R-L	CS R-1	CS R-2	CS R-L	Bio R-1	Bio R-2	Bio R-L
Coverage (Cov)	41.29	12.09	37.14	**38.57**	10.81	**35.11**	**35.86**	8.41	**32.56**
Order	41.20	12.20	37.11	38.50	10.87	35.10	35.66	**8.46**	32.39
Adjacency (Adj)	40.97	12.46	36.70	37.44	10.67	33.86	34.89	8.45	31.57
Factuality	**41.67**	**12.65**	**37.23**	38.23	**11.03**	34.64	35.46	8.41	31.96

Table 6: Comparison of different Co-opNet discriminators

PGen vs. Co-opNet-Adj			Generator vs. Co-opNet-Adj		
Criteria	**PGen**	**Co-opNet**	**Criteria**	**Generator**	**Co-opNet**
Abstractiveness	41.89	**47.30**	Abstractiveness	20.41	**38.10**
Coherence	42.57	**50.00**	Coherence	23.81	**34.01**
Factuality	39.86	**45.95**	Factuality	22.98	**30.41**
Overall	34.90	**53.02**	Overall	25.00	**31.08**

PGen vs. Co-opNet-Fact			Generator vs. Co-opNet-Fact		
Criteria	**PGen**	**Co-opNet**	**Criteria**	**Generator**	**Co-opNet**
Abstractiveness	**51.02**	39.46	Abstractiveness	27.33	**35.33**
Coherence	43.92	**50.00**	Coherence	30.87	**32.21**
Factuality	43.84	**48.63**	Factuality	27.52	**32.21**
Overall	43.54	**50.34**	Overall	30.87	**32.21**

Table 7: Human Evaluation of Co-opNet Architectures (% of judgements for each model)

conduct human evaluations to assess how the discriminator affects generation quality using pairwise model comparisons.

Setup We use four key criteria in all evaluations – abstractiveness, coherence, factuality and best overall quality, which we define as follows:

- Abstractiveness → Which abstract rewords information from the introduction instead of directly copying from the introduction?
- Coherence → Which abstract is more structured, and presents a complete and coherent story about the work done in the paper?
- Factuality → Which abstract is more factually consistent, presenting the same information that appears in the introduction and not producing hallucinated information?
- Overall → Which abstract is better overall?

We conduct human evaluations on Amazon Mechanical Turk (AMT) considering 4 different abstractive baseline model variants over 100 randomly sampled AAN test set examples. Given a gold introduction, AMT evaluators are asked to compare a corresponding abstract generated from Co-opNet against an abstract generated by a baseline or our generator model. To reduce bias, the ordering of generated abstracts are randomized and evaluators are not told that abstracts are machine-generated.

Each abstract pair is judged by three unique annotators. For each criteria, we filter to 50 abstracts based on the amount of time AMT workers spent ($\geq$ 20 seconds) and inter-annotator agreement (at least $\frac{2}{3}$ of annotators should agree on which abstract is best). We also prime annotators to consider subtler aspects of discourse coherence by providing examples that capture good or bad narrative flow without complete text degeneration.

We test the Co-opNet framework using both the factuality and adjacency discriminators, as these are the highest and lowest performing discriminator architectures in terms of automatic metrics on the AAN domain. We allow for ties, as Co-opNet and the generator baseline sometimes assign the highest probability to the same abstract, or generated abstracts in the candidate pool are high quality enough that there is little room for improvement.

Results We find that Co-opNet is preferred across all criteria for all comparisons, when we use the adjacency discriminator (see Table 7). When using the the factuality discriminator, Co-opNet is superior to baselines in all cases except when compared on abstractiveness to the PGen model.

In particular, human evaluators prefer Co-

Gold	We investigate mutual benefits between syntax and semantic roles using neural network models, by studying a parsing->SRL pipeline, a SRL->parsing pipeline, and a simple joint model by embedding sharing. The integration of syntactic and semantic features gives promising results in a Chinese Semantic Treebank...
PGen	In this paper, we propose a novel approach to learn ***syntactic and semantic role labeling models*** *to semantic role labeling (wsd). In the first* neural network models induce non-linear *feature features* from word and ***part-of-speech (pos) parsing***. We show that semantic features can be used to learn...
Generator	***Syntax-semantic relations*** play a crucial role in natural language processing. *In contrast,* ***semantic role labeling (srl)*** models typically rely on parser output features to improve accuracy. In this work, we propose a joint ***srl and syntactic parsing srl pipeline*** using the ***chinese treebank*** *(qiu et al., 2016)...*
Co-opNet (Adj)	In this paper, we explore the use of neural network models to jointly train ***semantic role labelers*** and ***parsers*** for ***semantic role labeling (srl)***. We first propose a simple neural ***srl*** model that uses a neural ***long shortterm memory (lstm)-based parser*** to represent the output of an ***srl*** system...

Table 8: Example of gold and generated abstracts from baseline Pointer Networks + Coverage (See et al., 2017) (PGen) and two of our proposed models, Generator and Co-opNet, on the NLP scientific domain. Coherence issues and factual errors in generated abstracts are highlighted in *italics*. We highlight correct terminology and transitional phrases that contribute to coherent flow by properly delineating sections of abstracts in ***bold*** and ***italics***.

opNet with the adjacency discriminator over baselines by over 8% on the coherence metric and 18.12% compared to PGen on overall quality. Notably, the adjacency discriminator encourages more abstractiveness in generated abstracts while still maintaining higher levels of factual consistency. We also find that Co-opNet with the factuality discriminator improves coherence and overall quality in addition to factuality. However, Co-opNet generations with the factuality discriminator were found to be more extractive than abstracts generated by PGen.

As shown in Table 8, generations selected by the adjacency discriminator more closely match the distribution of abstracts, while the generator sometimes favors copying from the introduction at the loss of narrative structure. For example, the generator will select a summary that opens with "*we present a method for jointly solving penn treebank style empty category (e.g. figure 1)...*", while the adjacency discriminator selects a summary that opens with "*we present a method to jointly solve the problem of empty categories...*" and does not refer to a particular figure. Both summaries are faithful to the introduction, but the discriminator-selected summary makes more sense in the context of a paper abstract.

8 Related Work

Narrative Flow and Factuality Modeling coherent narrative flow remains a major challenge in the field of text generation, due to the need for accurate understanding of narrative structure (Christensen et al., 2013; Nikolov et al., 2018; Holtzman et al., 2018; Qin et al., 2019; Koncel-Kedziorski et al., 2019; Gabriel et al., 2021). Early approaches to incorporating structure include integration of explicit discourse markers into automatic summarization (Alonso i Alemany and Fuentes Fort, 2003). Recently proposed solutions include global-tracking of entities (Kiddon et al., 2016; Bosselut et al., 2018; Mei et al., 2016), as well as discourse-aware attention (Cohan et al., 2018). While there has been prior work on factual consistency (Cao et al., 2018; Gao et al., 2019; Kryściński et al., 2020; Zhang et al., 2020b), these works did not focus on scientific paper summarization.

Neural Abstractive Summarization In the past, abstractive summarization models (Rush et al., 2015; Gehrmann et al., 2018) have relied upon seq2seq encoder-decoder architectures (Sutskever et al., 2014; Narayan et al., 2018; Celikyilmaz et al., 2018). Transformer models have emerged as a promising architecture for text generation and summarization (Liu et al., 2018; Hoang et al., 2019; Khandelwal et al., 2019; Zhang et al., 2019). While our model builds upon this work, it is, to our knowledge, the first transformer summarization framework to explicitly model narrative flow and scientific fact-checking across domains.

9 Conclusion

In this work, we introduced *Cooperative Generator-Discriminator Networks*, a framework for more coherent natural language generation with transformer language models through the integration of discriminators that encourage proper narrative flow and factual consistency. Through our analyses over scientific papers from ArXiv and AAN, we empirically showed that our framework selects generations that are more relevant and narratively coherent than previous approaches.

Acknowledgments

We thank the anonymous reviewers, as well as Lianhui (Karen) Qin, Jungo Kasai, Rik Koncel-Kedziorski, Elizabeth Clark, Dave Wadden and Rowan Zellers for helpful feedback. We also thank the annotators who contributed to the human evaluations in this work. This research was supported in part by NSF (IIS-1524371), DARPA CwC through ARO (W911NF-15-1-0543), and Samsung AI Research.

References

Laura Alonso i Alemany and Maria Fuentes Fort. 2003. Cohesion and coherence for automatic summarization. In *Student Research Workshop*.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. Text summarization techniques: A brief survey. *International Journal of Advanced Computer Science and Applications*, 8(10).

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. In *EMNLP*.

Antoine Bosselut, Asli Çelikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. 2018. Discourse-aware neural rewards for coherent text generation. In *NAACL-HLT*.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *AAAI*.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *NAACL-HLT*.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. 2019. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *ACL*.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Daniel S. Weld. 2019. Pretrained language models for sequential sentence classification. In *EMNLP/IJCNLP*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL-HLT*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*.

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021. Paragraph-level commonsense transformers with recurrent memory. In *AAAI*.

Shen Gao, Xiuying Chen, Piji Li, Zhangming Chan, Dongyan Zhao, and Rui Yan. 2019. How to write summaries with patterns? learning towards abstractive summarization through prototype editing. In *EMNLP*.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*, pages 4098–4109, Brussels, Belgium.

Max Grusky, Mor Naaman, and Yoav Artzi. 2019. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *NAACL-HLT*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Neural Information Processing Systems*.

Andrew Hoang, Antoine Bosselut, Asli Celikyilmaz, and Yejin Choi. 2019. Efficient adaptation of pretrained transformers for abstractive summarization. *ArXiv*, abs/1906.00138.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *ACL*.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *ArXiv*, abs/1905.08836.

Chloé Kiddon, Luke S. Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *EMNLP*.

Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *ACL*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *NAACL-HLT*.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *EMNLP*.

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

Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *ICLR*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *ACL*.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. Coherent dialogue with attention-based language models. In *AAAI*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *EMNLP*.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*.

Nikola I. Nikolov, Michael Pfeiffer, and Richard H. R. Hahnloser. 2018. Data-driven summarization of scientific articles. In *7th International Workshop on Mining Scientific Publications, LREC 2018*.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *EMNLP*.

Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2009. The acl anthology network corpus. In *Language Resources and Evaluation*, volume 47, pages 919–944.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Unpublished manuscript.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *ACL*.

Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. 2019. How to compare summarizers without target length ? pitfalls , solutions and re-examination of the neural summarization literature. In *NAACL 2019 Workshop on Optimizing and Evaluating Neural Language Generation (NeuralGen)*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Neural Information Processing Systems*, pages 3104–3112.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *EMNLP*.

Jiacheng Xu, Zhe Gan, Y. Cheng, and Jing jing Liu. 2020. Discourse-aware neural extractive text summarization. In *ACL*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with bert. In *ICLR*.

Xingxing Zhang, Furu Wei, and Ming de Zhou. 2019. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *ACL*.

Yuhao Zhang, Derek Merck, Emily Bao Tsai, Christopher D. Manning, and Curtis P. Langlotz. 2020b. Optimizing the factual correctness of a summary: A study of summarizing radiology reports. In *ACL*.

A Appendices

A.1 Additional Implementation Details

A.2 Baselines

For the sequence-to-sequence RNN model, a bi-LSTM is used to encode a given source article a and a separate decoder LSTM produces the generated summary g. At each decoding time step, the decoder attends to all the context vectors produced by the encoder as well as the maintained state from the previous decoder tokens to produce the next token in the summary.

The Pointer-Generator (PGEN + Cov) model extends the base LSTM model (LSTM + Cov) to allow tokens to be copied from the input during generation. Baselines are trained for up to 40000 steps with a batch size of 16. Following previous work, we decode from these baselines using beam search with a beam size of 4.

A.3 Generator Model

We use the 345M parameter GPT-2 model. The model is trained to minimize the negative log likelihood of the next word w_i given all preceding words:

$$\mathcal{L}_{gen} = -\sum_{i=1}^{|a|+|s|} \log P(w_i | w_0, ... w_{i-1}) \tag{10}$$

where w_i is the i^{th} token of our full input vector X, a is our article and s is our summary. At test time, X only consists of the gold article and delimiter token $(a_1, ..., a_{|a|}, [\text{SEP}])$ and we decode generated summaries g starting from this input.

During generation, we filter candidate summaries from the hypothesis generation pool that contain sentences longer than a fixed max length of 200 tokens, a clear sign of coherence deterioration. We use a candidate pool size of 30 for ATLAS and 20 for AAN.

A.4 Discriminator Training

Factuality Discriminator Details For the token-level classification model, we use the BERT base model with binary labels for whether or not a token should be included in a salient span. We predict for all spans in an abstract at once.

Order Discriminator Details We set the max length of summaries considered by the order discriminator to be 10 sentences, truncating longer summaries. Given the max length of a summary, we have a fixed number of orderings |O| that can be scored. We calculate the final score from the order discriminator based on the unnormalized sum of scores from these orderings, S, and the following function f_n:

$$f_n(S, |O|) = \frac{S - (-|O|)}{|O| - (-|O|)} \tag{11}$$

Sentence Selection for Discriminator Models To train an adjacency discriminator model, we use a subset of adversarial and positive sentence pair examples extracted from the training set. The sentence pairs are extracted from gold abstracts containing at least five sentences using the following approach: For a randomly selected sentence s_u from the abstract, we randomly select an adjacent sentence, s_{u-1} or s_{u+1}, as a positive example and any nonadjacent sentence $s_{v \notin [u-1, u, u+1]}$ as a negative example.

Discriminator Performance We measure the performance of discriminator models using recall, precision, accuracy and F1. Table 9 provides summary statistics of discriminator performance on the various discourse and factuality objectives. Discourse-Adj denotes the adjacency discriminators, while Discourse-Abs denotes the discourse role label prediction model (Cohan et al., 2019) and Factuality denotes the token saliency prediction model.

Model	Training Data	Prec	Rec	F1	Acc
Discourse-Adj	ArXiv-AAN	86.05	85.25	85.65	86.81
Discourse-Adj	ArXiv-Bio	90.30	93.44	91.84	92.32
Discourse-Abs	CSAbstruct	88.99	89.09	89.04	89.00
Factuality	SciFact	73.70	70.50	72.10	75.70

Table 9: Automatic Evaluation of discriminator architectures

A.5 Details on Model Performance

Automatic results for Co-opNet selection were given using a context size of 800 tokens for the input, while a context size of 800 characters was used to select Co-opNet summaries for the human eval. The automatic results for the summaries used in the human eval were lower than the ones using the longer context size. Using a smaller context size leads to faster and more efficient Co-opNet selection (less memory usage), but slightly lower overall automatic performance (while maintaining the same ordering in terms of highest and lowest scores for Co-opNet variants on ROUGE).

A.6 Comparison of Datasets

We removed duplicates and articles without abstracts from AAN. From this subset, we extract introduction and abstract pairs.

A.7 Additional Analysis

Comparison with Gold Summaries To obtain an upper-bound comparison for the human evaluation and verify the effectiveness of our human evaluation pipeline for judging the quality of abstracts, we used the same intro-abstract pairs and Mturk annotation framework as the model comparison to conduct a Turing-style evaluation. In this evaluation, we presented a Co-opNet (adj) generated abstract and a gold abstract to the annotators in a random ordering without noting whether either of the abstracts were human-written or machine-generated. We found that annotators consistently selected the gold abstract over the machine-generated abstract when considering factuality and coherence, though they found the machine-generated abstracts to be slightly more abstractive. We provide the results for this full evaluation in Table 11.

Dataset	Narrative Flow?	# Summaries	Avg # Sents	Avg # Words
XSum (Narayan et al., 2018)	✗	226,711	1.00	23.26
Newsroom (Grusky et al., 2019)	✗	1,321,995	1.45	26.70
CNN (Hermann et al., 2015)	✗	92,579	3.59	45.70
DailyMail (Hermann et al., 2015)	✗	219,506	3.86	54.65
ArXiv	✓	472,493	6.11	150.85
AAN	✓	11,890	5.03	106.76

Table 10: Statistics of gold summaries in different summarization datasets.

Gold vs. Co-opNet-Adj		
Criteria	**Gold**	**Co-opNet**
Abstractiveness	47.37	**52.63**
Coherence	**66.67**	31.06
Factuality	**66.92**	32.33
Overall	**61.36**	32.33

Table 11: Human Evaluation of Co-opNet Architectures (% of judgements for each model)

Knowledge Base Question Answering through Recursive Hypergraphs

Naganand Yadati[1], Dayanidhi R[2], Vaishnavi S[2], Indira G[2], Srinidhi G[2]
[1] Indian Institute of Science, Bangalore
[2] Channabasaveshwara Institute of Technology, Tumkur
`naganand@iisc.ac.in`
`{dayanidhirs.12, vaishnavisathish98, indiragowtham99, srinidhi.g71}@gmail.com`

Abstract

Knowledge Base Question Answering (KBQA) is the problem of predicting an answer for a factoid question over a given knowledge base (KB). Answering questions typically requires reasoning over multiple links in the given KB. Humans tend to answer questions by *grouping* different objects to perform reasoning over acquired knowledge. Hypergraphs provide a natural tool to model group relationships. In this work, inspired by typical human intelligence, we propose a new method for KBQA based on hypergraphs. Existing methods for KBQA, though effective, do not *explicitly incorporate the recursive relational group structure* in the given KB. Our method, which we name RecHyperNet (Recursive Hypergraph Network), exploits a new way of modelling KBs through recursive hypergraphs to organise such group relationships in KBs. Experiments on multiple KBQA benchmarks demonstrate the effectiveness of the proposed RecHyperNet. We have released the code.

1 Introduction

Knowledge Base Question Answering (KBQA) (Xu et al., 2016)), a task that tests the ability of a machine to understand knowledge like a human, is a challenging, central, and popular task in natural language processing. KBQA is the problem of predicting an answer for a factoid question over a given knowledge base (KB) containing facts such as (Inception, written by, Christopher Nolan). Answering questions typically requires reasoning over multiple links in the given KB (Zhang et al., 2018).

When a typical human answers a question (e.g. What are the genres of movies written by Christopher Nolan?), they tend to group objects (e.g. movies written by Christopher Nolan such as Inception, Interstellar, etc. are grouped together) over their acquired knowledge. Hypergraphs are mathematical tools that naturally encode group relationships. Directed hypergraphs have been recently used to model KBs for KBQA (Han et al., 2020) but we argue that they fail to naturally model group relationships without loss of information. Spcifically, two KB triples such as (Inception, written by, Christopher Nolan) and (Person of Interest, written by, Jonathan Nolan) would be modelled by a directed hyperedge {Inception, Person of Interest} -> {Christopher Nolan, Jonathan Nolan} clearly resulting in loss of information (not clear who directed which). Though objects of similar types are grouped together and directions are encoded, directed hypergraphs fail to model KBs without loss of information.

We argue that recursive hypergraphs (Joslyn and Nowak, 2017; Menezes and Roth, 2019) provide a flexible way to model KBs without loss of information. Specifically, all movies written by Christopher Nolan are grouped together and those written by Jonathan Nolan are *separately* grouped together. To summarise, we make the following contributions

- We model KBs as recursive hypergraphs. To the best of our knowledge, this is the first such attempt for natural language processing.
- We propose RecHyperNet (Recursive Hypergraph Network), a novel graph neural network-based model to exploit recursive hypergraphs and apply it for KBQA.
- We show the effectiveness of RecHyperNet on multiple KBQA benchmarks. We have released the source code in the supplementary.

2 Related Work

Knowledge Base Question Answering (KBQA) has been widely investigated especially through

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 448–454
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

end-to-end deep neural networks since the release of popular datasets such as MetaQA (Zhang et al., 2018). One of the first attempts at KBQA was key-value memory network (KVMN) (Miller et al., 2016) that maintained memory to store KB facts and text as key-value pairs. Graphs of Relations Among Facts and Text Networks (GraftNet) (Sun et al., 2018) uses a heterogeneous GCN-based method to fuse information from heterogeneous sources (KBs and text). SubgraphReader (SGReader) (Xiong et al., 2019) employs a graph attention-based method to combine unstructured text and structured KB. PullNet (Sun et al., 2019) also uses a GCN to identify subgraph nodes that should be retrieved ('pull') from text and KB. More recently, EmbedKGQA (Saxena et al., 2020) uses ideas from knoweldge graph embedding literature to improve knowledge base question answering esp. on sparser incomplete knowledge graphs. Two-Phase Hypergraph Based Reasoning with Dynamic Relations (2HR-DR) (Han et al., 2020) is a directed hypergraph-based method for KBQA. The reader is referred to a comprehensive literature review on this topic (Fu et al., 2020).

Graph Neural Networks (GNNs): While deep neural networks such as convolutional neural networks and recurrent neural networks are specially designed for grids and sequences respectively, GNNs (Kipf and Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018) are specially designed to learn representations from graph-structured data. GNNs have been recently extended to hypergraphs (Feng et al., 2019; Yadati et al., 2019). The first published works of GNNs for NLP investigated the tasks of semantic-role labelling (Marcheggiani and Titov, 2017), and neural machine translation (Bastings et al., 2017). A recent tutorial touches upon the recent advances of GNNs in NLP (Vashishth et al., 2019).

GNNs for Question Answering (QA): A popular task closely related to question answering is multi-hop reasoning across documents (context passages) where GNNs have been extensively used (Qiu et al., 2019b; Ding et al., 2019; Tu et al., 2020). Graph Attention Networks (Veličković et al., 2018) have been shown to be effective for modelling the multi-grained structure of documents for machine reading comprehension (MRC) (Zheng et al., 2020). GNNs have also recently been used on heterogeneous data sources for MRC (Kim et al., 2019; Tu et al., 2019), knowledge graphs for QA (De Cao et al., 2019; Cao et al., 2019; Lin et al., 2019; Qiu et al., 2019a; Feng et al., 2020; Ji et al., 2020; Shao et al., 2020), answer sentence selection (Tian et al., 2020), and QA on tables (Zhang, 2020) and math (Chen et al., 2020). In all these publications except 2HR-DR (Han et al., 2020), the input is a graph and limited to modelling pairwise relationships. We address this fundamental limitation by modelling a knowledge graph as a recursive hypergraph 2HR-DR models knowledge graphs through directed hypergraphs but we argue that they are more naturally represented by recursive hypergraphs.

3 Method

We describe the KBQA problem, and how recursive hypergraphs (Menezes and Roth, 2019; Joslyn and Nowak, 2017) can be used to model knowledge bases. We then describe our method of exploiting recursive hypergraphs for KBQA.

3.1 KBQA Problem

The KBQA problem considered in this work is as follows. We are given a knowledge base $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ with entity set $\mathcal{E}$ and relation set $\mathcal{R}$. We are also given a natural language question q with a topic entity $e \in \mathcal{E}$. The task is to output an answer $a \in \mathcal{E}$ that correctly answers the question q.

3.2 Recursive Hypergraph

In this subsection, we precisely define a recursive hypergraph by first defining the following.

Definition 1 (Depth k Powerset)**.** *For a set S, let us use $\mathcal{S}(S)$ to denote the powerset of S i.e. $\mathcal{S}(S) := \{\dot{S} : \varnothing \subseteq \dot{S} \subseteq S\}$. Then, the the depth k powerset of S is*

$$2^{S,k} := \mathcal{S}\left(\bigcup_{i=0}^{k} S_i\right) \tag{1}$$

where $S_0 = S, S_i = \mathcal{S}\left(\bigcup_{j=0}^{i-1} S_j\right)$, for $i \geq 1$.

Note that $2^{S,0} = \mathcal{S}(S)$ i.e. $2^{S,0}$ is powerset of S.

Definition 2 (k-Recursive Hypergraph)**.** *A pair $H = (V, E)$, where V is a set of n vertices, and $E \subseteq \left(2^{V,k} - \varnothing\right)$ is a set of recursive hyperedges.*

Note that a hypergraph in the traditional sense is a 0-recursive hypergraph. We call a hyperedge $e \in E$ as a depth k hyperedge if $e \subseteq 2^{V,k}$ but $e \nsubseteq 2^{V,k}$.

3.3 Modelling knowledge Base as a Recursive Hypergraph

One of our main contributions in this work is to model a knowledge base as a 1-recursive hypergraph with $V = \mathcal{E} \cup \mathcal{R}$ as the set of vertices. Each head entity can be seen as a depth 1 hyperedge connecting all its relations. Each relation can be further seen as a depth 0 hyperedge connecting all the tail entities. For example, if a knowledge base contains the movie "Inception" as a head entity with (relation, object) pairs as 1) (starred actors, Leonardo DiCaprio), 2) (starred actors, Ellen Page), 3) (starred actors, Tom Hardy), 4) (genre, action), 5) (genre, adventure), and 6) (genre, science fiction), and 7) (written by, Christopher Nolan) then we can view them as a recursive hypergraph with Inception as a depth 1 hyperedge connecting relation vertices starred actors, genre, and written by. Each of these relation vertices is in turn a depth 0 hyperedge connecting object entities. For example, the object entities Leonardo DiCaprio, Ellen Page, and Tom Hardy are contained in a single hyperedge (that represents the pair Inception, starred actors).

3.4 RecHyperNet for KBQA

In this section, we describe our proposed methodology for KBQA. The main ingredients/modules of RecHyperNet are three modules: KB Embedding Module, Topic and Question Embedding Module, and Answer Retrieval Module.

1) KB Embedding Module uses knowledge base embedding methods to initialise entity embeddings of the input KB. We denote these initial embeddings by $x_e, e \in \mathcal{E}$. Popular knowledge base embedding methods include TransE (Bordes et al., 2013), and ComplEX (Trouillon et al., 2016).

2) Topic and Question Embedding Module embeds the question q to a fixed dimension vector $\boldsymbol{q}$ of dimension d. We use a feed-forward neural network that first represents q using LSTM/ RoBERTa (Liu et al., 2019). The topic entity e that is also present in the question is embedded using a multi-relational graph convolutional network on the KB (Vashishth et al., 2020). The entity update rule for Composition-based multi-relational graph convolutional network (CompGCN) is as follows:

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}\phi(\boldsymbol{x}_u, \boldsymbol{z}_r)\right), \qquad (2)$$

where $\boldsymbol{x}_u$ and $\boldsymbol{z}_r$ are initial features of vertex u and relation r respectively. ϕ is a composition function that is dictated by the knowledge base embedding method and f is a non-linear activation function such as Rectified Linear Unit. $\mathcal{N}(v)$ is the (relation, object) neighbourhood of the vertex v (for example, the 7 pairs listed for "inception" movie entity in the previous subsection).

Our key modification to CompGCN update rule is motivated through the proposed recursive hypergraph view of knowledge base. Specifically, since each relation can be seen as a hyperedge containing all object entities, we can modify the update rule as follows

$$\boldsymbol{h}_v{=}f\left(\textstyle\sum_{r\in N(v)} \boldsymbol{W}\phi\left(\max_{u\in\mathcal{N}(v,r)}(\boldsymbol{x}_u){,}\boldsymbol{z}_r\right)\right), \qquad (3)$$

where $N(v)$ is the set of all relations in the neighbourhood of v, and $\mathcal{N}(v, r)$ is the set of all object entities connected by the (subject, relation) pair (v, r). $\max$ is the element-wise maximum of a set of vectors. We can replace $\max$ by other aggregator functions such as mean, and sum but we experimentally observed that $\max$ gives the best performance. We embed the topic head entity $e \in \mathcal{E}$ using Equation 3 and obtain the hidden representation $\boldsymbol{h}_e$ with the same dimension d. We finally pass the concatenated representation $\boldsymbol{h}_e||\boldsymbol{q}$ to a 2-layer feed-forward neural network to get the topic and question representation.

3) Answer Selection Module Given a KB embedding scoring function Φ and a set of answer entities $\mathcal{A} \subseteq \mathcal{E}$, we learn the topic and question representation $\boldsymbol{h}_e||\boldsymbol{q}$ so that $\Phi(x_e,\ \boldsymbol{h}_e||\boldsymbol{q},\ x_a) > 0,\ \forall a \in \mathcal{A}$ and $\Phi(x_e,\ \boldsymbol{h}_e||\boldsymbol{q},\ x_{\bar{a}}) > 0,\ \forall \bar{a} \notin \mathcal{A}$. We leartn the model parameters by minimising the binary cross entropy loss between the sigmoid of the scores and the target labels (1 for correct, 0 for wrong).

During test time, we score the given (topic head entity, question) pair against all possible answers $a' \in \mathcal{E}$. So, the answer is given by $\arg\max_{a'\in\mathcal{E}} \Phi(x_e,\ \boldsymbol{h}_e||\boldsymbol{q},\ x_{a'})$.

4 Experiments

We evaluate our proposed method on MetaQA (Zhang et al., 2018) and WebQuestionsSP (Yih et al., 2016) datasets. We closely follow the experimental setup of a prior work (Saxena et al., 2020) for the preprocessed versions of these datasets. MetaQA consists of $400k$ questions ($1 - hop$, 2-hop, and 3-hop), and a knowledge graph of 135k triples, 43k entities, and 9 relations. WebQuestionsSP consists of 4700 questions and a knowledge

Model	**MetaQA KG-Full**			**MetaQA KG-50**			**WebQSP**	**WebQSP**
	1-hop	2-hop	3-hop	1-hop	2-hop	3-hop	**KG-Full**	**KG-50**
KVMN (Miller et al., 2016)	96.2	82.7	48.9	75.7	48.4	37.6	46.7	32.7
VRN (Zhang et al., 2018)	97.5	89.9	62.5	–	–	–	–	–
GraftNet (Sun et al., 2018)	97.0	94.8	77.7	91.5	69.5	66.4	66.8	49.7
SGReader (Xiong et al., 2019)	96.7	80.7	68.6	79.2	77.1	63.5	–	–
PullNet (Sun et al., 2019)	97.0	**99.9**	91.4	**92.4**	90.4	**85.2**	**68.1**	51.9
EmbedKGQA (Saxena et al., 2020)	97.5	98.8	94.8	83.9	91.8	70.3	66.6	53.2
2HR-DR (Han et al., 2020)	98.8	93.7	91.4	80.8	89.3	65.1	67.0	52.2
RecHyperNet (ours)	**99.1**	99.2	**95.0**	84.4	**92.3**	71.1	**68.4**	**53.7**

Table 1: Results (higher is better) on MetaQA, WebQuestionsSP datasets. Baselines such as PullNet utilise external corpus to answer questions (unrealistic as it is not always readily available) while our method does not.

Model	**1-hop**	**2-hop**	**3-hop**
GraftNet	64.0	52.6	59.2
PullNet	65.1	52.1	59.7
RecHyperNet	**84.4**	**92.3**	**71.1**

Table 2: Results (higher is better) of baselines without text corpus on MetaQA KG-50.

Model	**MetaQA**	**WebQSP**
RecHyperNet	**92.3**	**53.7**
replace Eq. 3 by Eq. 2	90.4	52.1
use $\boldsymbol{q}$ in place of $\boldsymbol{h}_e \| \boldsymbol{q}$	85.6	50.7

Table 3: Ablation study on MetaQA KG-50 (2-hop) and WebQSP KG-50

graph of 1.8 million entities, and 5.7 million triples. Following prior work (Saxena et al., 2020), we experimented on two different settings (for both datasets) - **KG Full** (in which the KG is left untouched), and the more realistic **KG-50** setting in which 50% links are randomly removed. We compared against 7 different baselines as shown in Table 1. Please see Section 2 (subsection: Knowledge Base Question Answering) for brief descriptions of the baseline methods.

Model details Following prior work (Saxena et al., 2020), we used a long short term memory (LSTM) network to learn embeddings for words in the questions with an embedding size of 256 for MetaQA and RoBERTa (768 dimensional embeddings) (Liu et al., 2019) for WebQuestionsSP datasets. The hidden dimension size for graph convolutional network was also set to 256. A dropout rate of 0.2 was used for all neural layers.. All models were implemented in PyTorch (Paszke et al., 2019) and trained with ADAM as the optimiser (Kingma and Ba, 2015), a learning rate of 5×10^{-4}, weight decay of 1.0, a batch size of 128 trained for 100 epochs (with patience of 5).

Results. Table 1 shows the results. We evaluated our proposed method and all baselines through Hits@1 metric. As we can see in Table 1, exploiting recursive hypergraphs through our proposed method help. Methods such as PullNet and GraftNet utilise additional text corpus (unrealistic setting as it is not always readily available) to answer questions while our method does not. We exploit only the given knowledge base structure. Futhermore, as shown in Table 2, baselines such as GraftNet and PullNet perform poorly in the absence of text.

Ablation Analysis We conducted an ablation study by removing essential components from RecHyperNet. Specifically we replaced our proposed Equation 3 to exploit recursive hypergraphs by the exisdting Equation 2 of CompGCN. As another basline, we removed the embedding of topic head entity obtained through GCN and used only the question representation while scoring. As shown in Table 3, both these components are essential for our proposed RecHyperNet.

Conclusion and Future Work In this work, we exploit recursive hypergraphs in NLP for the task of KBQA. We have proposed a novel method based on graph convolutional networks. We have demonstrated the effectiveness of RecHyperNet on KBQA benchmarks. In future, we exploit recursive structures for other tasks where graph neural nets are effective such as question generation (Pan et al., 2020), sentiment analysis (Wang et al., 2020), etc.

Acknowledgements

This work is supported by the Ministry of Human Resource Development (Government of India).

References

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1957–1967.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems (NeurIPS) 26*, pages 2787–2795. Curran Associates, Inc.

Yu Cao, Meng Fang, and Dacheng Tao. 2019. BAG: Bi-directional attention entity graph convolutional network for multi-hop reasoning question answering. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 357–362.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020. Question directed graph attention network for numerical reasoning over text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2306–2317.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2694–2703.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the Thirty-Third Conference on Association for the Advancement of Artificial Intelligence (AAAI)*, pages 3558–3565.

Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. *CoRR*, abs/2007.13069.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS) 30*, pages 1024–1034. Curran Associates, Inc.

Jiale Han, Bo Cheng, and Xu Wang. 2020. Two-phase hypergraph based reasoning with dynamic relations for multi-hop kbqa. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3615–3621.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Semsum: Semantic dependency guided neural abstractive summarization. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Cliff Joslyn and Kathleen Nowak. 2017. Ubergraphs: A definition of a recursive hypergraph structure. *Computing Research Repository (CoRR)*, abs/1704.05547.

Daesik Kim, Seonhoon Kim, and Nojun Kwak. 2019. Textbook question answering with multi-modal context graph understanding and self-supervised openset comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3568–3584.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1506–1515.

Telmo Menezes and Camille Roth. 2019. Semantic hypergraphs. *Computing Research Repository (CoRR)*, abs/1908.10784.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1400–1409.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS) 32*, pages 8026–8037. Curran Associates, Inc.

Delai Qiu, Yuanzhe Zhang, Xinwei Feng, Xiangwen Liao, Wenbin Jiang, Yajuan Lyu, Kang Liu, and Jun Zhao. 2019a. Machine reading comprehension using structural knowledge graph-aware network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5898–5903.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019b. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6140–6150.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4498–4507.

Nan Shao, Yiming Cui, Ting Liu, Shijin Wang, and Guoping Hu. 2020. Is graph structure necessary for multi-hop reasoning? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4231–4242.

Zhixing Tian, Yuanzhe Zhang, Xinwei Feng, Wenbin Jiang, Yajuan Lyu, Kang Liu, and Jun Zhao. 2020. Capturing sentence relations for answer sentence selection with multi-perspective graph encoding. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning (ICML) - Volume 48*, page 2071–2080.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2704–2713.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Shikhar Vashishth, Naganand Yadati, and Partha Talukdar. 2019. Graph-based deep learning in natural language processing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): Tutorial Abstracts*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete KBs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4258–4264.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on Freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL, Volume 1: Long Papers)*, pages 2326–2336.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. HyperGCN: A new method of training graph convolutional networks on hypergraphs. In *Advances in Neural Information Processing Systems (NeurIPS) 32*, pages 1509–1520. Curran Associates, Inc.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Xuanyu Zhang. 2020. Cfgnn: Cross flow graph neural networks for question answering on complex tables. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, pages 6069–6076.

Bo Zheng, Haoyang Wen, Yaobo Liang, Nan Duan, Wanxiang Che, Daxin Jiang, Ming Zhou, and Ting Liu. 2020. Document modeling with graph attention networks for multi-grained machine reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

FEWS: Large-Scale, Low-Shot Word Sense Disambiguation with the Dictionary

Terra Blevins **Mandar Joshi** **Luke Zettlemoyer**
Paul G. Allen School of Computer Science & Engineering, University of Washington
Facebook AI Research, Seattle
{blvns, mandar90, lsz}@cs.washington.edu

Abstract

Current models for Word Sense Disambiguation (WSD) struggle to disambiguate rare senses, despite reaching human performance on global WSD metrics. This stems from a lack of data for both modeling and evaluating rare senses in existing WSD datasets. In this paper, we introduce FEWS (Few-shot Examples of Word Senses), a new low-shot WSD dataset automatically extracted from example sentences in Wiktionary. FEWS has high sense coverage across different natural language domains and provides: (1) a large training set that covers many more senses than previous datasets and (2) a comprehensive evaluation set containing few- and zero-shot examples of a wide variety of senses. We establish baselines on FEWS with knowledge-based and neural WSD approaches and present transfer learning experiments demonstrating that models additionally trained with FEWS better capture rare senses in existing WSD datasets. Finally, we find humans outperform the best baseline models on FEWS, indicating that FEWS will support significant future work on low-shot WSD.

1 Introduction

Word Sense Disambiguation (WSD) is the task of identifying the *sense*, or meaning, that an ambiguous word takes in a specific context. Recent WSD models (Huang et al., 2019; Blevins and Zettlemoyer, 2020; Bevilacqua and Navigli, 2020) have made large gains on the task, surpassing the estimated 80% F1 human performance upper bound on WordNet annotated corpora (Navigli, 2009). Despite this breakthrough, the task remains far from solved: performance on rare and zero-shot senses is still low, and in general, current WSD models struggle to learn senses with few training examples (Kumar et al., 2019; Blevins and Zettlemoyer, 2020). This performance gap stems from limited data for rare senses in current WSD datasets, which are annotated on natural language documents that contain a Zipfian distribution of senses (Postma et al., 2016).

C: I **liked** my friend's last status...
S1: to enjoy... [or] be in favor of.
✓ **S2**: To show support for, or approval of, something on the Internet by marking it with a vote.

C: A transistor-diode **matrix** is composed of vertical and horizontal wires with a transistor at each intersection.
✓ **S1**: A grid-like arrangement of electronic components, especially one intended for information coding, decoding or storage.
S2: A rectangular arrangement of numbers or terms having various uses [in mathematics].

Figure 1: Sample contexts (**C**) from FEWS with ambiguous words and a subset of candidate sense definitions (**S**). FEWS covers a wide range of senses, including new senses and domain-specific senses.

More generally, since each word has a different set of candidate senses and new senses are regularly coined, it is almost impossible to gather a large number of examples for each sense in a language. This makes the few-shot learning setting particularly important for WSD. We introduce **FEWS** (Few-shot Examples of Word Senses), a dataset built to comprehensively train and evaluate WSD models in few- and zero-shot settings. Overall, the contributions of FEWS are two-fold: as training data, it exposes models a broad array of senses in a low-shot setting, and the large evaluation set allows for more robust evaluation of rare senses.

FEWS achieves high coverage of rare senses by

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 455–465
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Data Split	# Examples	# Tokens	# Annot.	# Sense Types	# Word Types	Ambiguity
Overall	121,459	3,259,240	131,278	71,391	35,416	5.62
Train	87,329	2,551,358	96,023	52,928	30,450	4.98
Ext. Train	101,459	2,683,345	111,278	61,391	31,937	5.71
Dev	10,000	287,673	10,000	10,000	8,682	5.09
Few-shot	5,000	149,791	5,000	5,000	4,417	4.77
Zero-shot	5,000	137,882	5,000	5,000	4,661	5.41
Test	10,000	288,222	10,000	10,000	8,709	5.10
Few-shot	5,000	149,384	5,000	5,000	4,449	4.71
Zero-shot	5,000	138,838	5,000	5,000	4,666	5.49

Table 1: FEWS data statistics. The development and test sets are balanced across senses and split evenly between few-shot examples (with support in the training set) and zero-shot examples. The extended training set (Ext. Train) adds short examples written by Wiktionary editors as additional training data.

automatically extracting example sentences from Wiktionary definitions. Wiktionary is an apt data source for this purpose, containing examples for over 71,000 senses (Table 1). Not only is this sense coverage higher than existing datasets (e.g., SemCor, the largest manually annotated WSD dataset, only covers approximately 33,000 senses (Miller et al., 1993)), it also extends to senses related to new domains (Figure 1).

We establish performance baselines on FEWS with both knowledge-based approaches and a recent neural biencoder model for WSD (Blevins and Zettlemoyer, 2020). We find that the biencoder, despite being the strongest baseline on FEWS, still underperforms human annotators, particularly on zero-shot senses where the biencoder trails by more than 10%. We also present transfer learning experiments and find adding FEWS as additional training data improves performance on all but the most frequent senses (MFS) in the WSD Evaluation Framework (Raganato et al., 2017); this suggests that future improvements on FEWS could generalize other WSD benchmarks. FEWS is available at `https://nlp.cs.washington.edu/fews`.

2 Related Work

WSD is a long-standing NLP task and is the focus of many datasets. The current *de facto* benchmark for modeling English WSD is the WSD Evaluation Framework (Raganato et al., 2017), which includes the SemCor dataset (Miller et al., 1993) as training data and consolidates a number of evaluation sets (Pradhan et al., 2007a; Palmer et al., 2001; Snyder and Palmer, 2004; Navigli et al., 2013; Moro and Navigli, 2015) into a standardized evaluation suite. These datasets are annotated with the senses (known as *synsets*) from Wordnet, a manually constructed ontology of semantic relations (Miller et al., 1993).

Most existing datasets for WSD, including those in the WSD Evaluation framework and others like Pradhan et al. (2007b), are annotated on natural language documents that contain a Zipfian distribution of word senses (Kilgarriff, 2004). This data source causes these datasets to have low coverage of rare senses, leading to worse performance on these less common senses (Postma et al., 2016; Kumar et al., 2019). In contrast, we use examples sentences from Wiktionary as an alternative source of text for WSD data with FEWS. This means that FEWS has a more uniform sense distribution, providing more balanced coverage across different senses of words.

Wiktionary has previously been used as a resource for WSD research. Most work has investigated mapping Wiktionary senses onto WordNet synsets (Meyer and Gurevych, 2011; Matuschek and Gurevych, 2013); other work has learned similar mappings for Wikipedia articles (Mihalcea (2007); Navigli and Ponzetto (2012); inter alia). More similar to our work, Henrich et al. (2012) and Segonne et al. (2019) mine WSD examples from Wiktionary to augment labeled WSD data for non-English languages. However, FEWS is the first dataset specifically designed to evaluate zero and few-shot learning with the balanced dictionary sense distribution.

3 FEWS: Low-shot Learning for WSD

FEWS (Few-shot Examples of Word Senses) is a new dataset for learning to do low-shot WSD.[1] It is created with example contexts drawn from Wiktionary, an online collaborative dictionary.[2] Since

---

[1] We use the term *low-shot* as an umbrella term for few- and zero-shot learning.

[2] https://en.wiktionary.org/

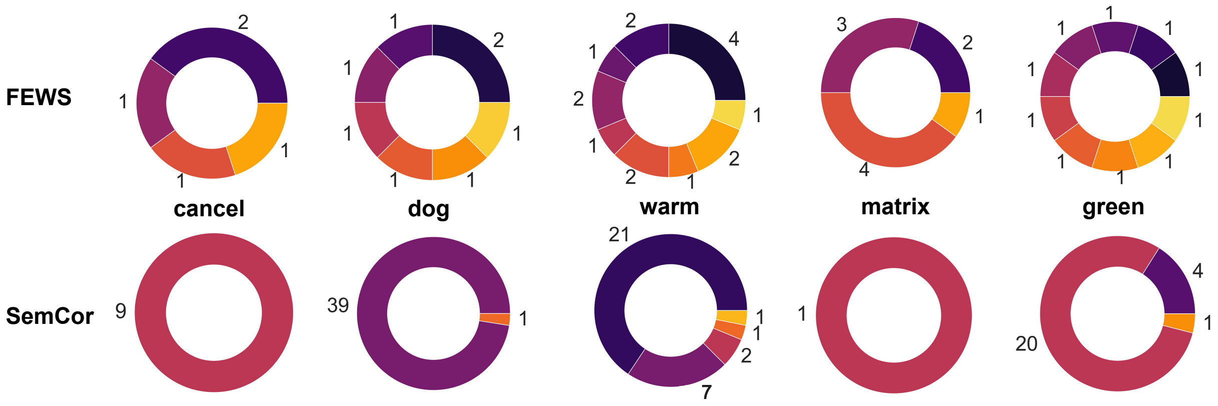


Figure 2: Comparison of sense coverage for five words in the Semcor and FEWS training corpuses.

Wiktionary is curated by volunteers, the data is manually annotated and high quality, and there is no additional annotation cost to construct FEWS. Furthermore, using example contexts from a dictionary allows FEWS to cover many senses underrepresented in existing WSD datasets, such as rare senses of words or senses pertaining to specific domains. However, we note that due to the crowdsourced nature of the examples in Wiktionary and the subjectivity of fine-grained sense distinctions, inconsistencies in the underlying data may introduce some annotation errors into FEWS.

Examples of the data in FEWS are shown in Figure 1. In FEWS, each example context contains one or more instances of the ambiguous *target word* and is labeled with the sense (and corresponding definition) that describes that word as used in the context; this is in contrast to *all-words WSD*, where many of the content words in the context are annotated.

3.1 Dataset Creation

To create FEWS, we extracted all of the definitions for content words (nouns, verbs, adjectives, and adverbs) and example contexts associated with those definitions from a checkpointed version of English Wiktionary.[3] While processing the Wiktionary data, we collected two types of contexts: (1) *quotations* (93% of extracted examples), which are quotations of natural language text found by Wiktionary contributors that contain the target word used with the relevant sense, and (2) *illustrations* (7%), which are short sentences or fragments written by contributors to illustrate the word sense in context. The target words in each context are marked by the Wiktionary formatting metadata; examples where no words are marked or the marked word differs too much from the base form of the dictionary entry are discarded.[4] We additionally filter out examples that are too short to provide a meaningful context for the marked word.

We then labeled the target words in each extracted context with the sense ID generated for the definition associated with that sentence; this gave us 254,506 annotated WSD example contexts covering 148,333 senses. Finally, we filtered out examples with monosemous words since predicting the sense in these cases is a trivial task. However, Loureiro and Camacho-Collados (2020) recently found that unambiguous examples can improve WSD performance; therefore, we include the unambiguous cases as an additional file in the FEWS dataset. After filtering these unambiguous examples from the main dataset, FEWS in total contains 121,459 examples covering 71,391 sense types.

Finally, we split the data into training and evaluation sets. The majority of the data are *quotations*, which we use to populate the train, development, and test sets as they more closely resemble naturally occurring text than the *illustrations*. To create the development and test sets, we randomly select 10,000 examples for each evaluation set and ensure that each of these evaluation examples pertains to a different sense. We verify that half of those examples were labeled with senses that only occurred once in the unsplit data to create a zero-shot subset of each evaluation set, and the other half of the evaluation senses comprise the few-shot evaluation subsets. The remaining quotations that were not

---

[3] We focus English senses and filter out definitions for words in other languages; however, this data collection process could be expanded to other languages in Wiktionary.

[4] We define a marked word as too different from the base form if the longest common subsequence between them is $< 50\%$ of the length of the marked word.

Eval Split	FEWS	WSD Fr.
Dev	10,000	375
Few-shot†	4,529	67
Zero-shot	5,000	50
Test	10,000	3,669
Few-shot†	4,603	761
Zero-shot	5,000	796

Table 2: The number of senses covered in the **FEWS** and **WSD Framework** evaluation sets. † To fairly compare against the WSD Framework, we only count few-shot examples as those have three or fewer supporting examples in their respective train set.

used for the development or test set are included as the training data. Finally, we remove the *illustrations* for senses in the zero-shot evaluation subsets and add the remaining *illustrations* to the training data; this addition makes the extended train set.

3.2 Dataset Analysis

We present a comprehensive analysis of FEWS to demonstrate that the dataset provides high coverage of many diverse words senses in a low-shot manner.

High Coverage of Words and Senses The FEWS dataset covers 35,416 polysemous words and 71,391 senses (Table 1). The complete dataset covers 53.21% of the senses for words that appear in it (out of their Wiktionary sense inventories). Figure 2 shows this high coverage of senses for five different words compared to the coverage of the same words in SemCor (Miller et al., 1993). We see that while SemCor tends to have more examples for these common words, most examples correspond to a single sense of the word. However, the FEWS train set covers many more senses per word, albeit with fewer total examples.

This high coverage of senses is particularly important for the evaluation sets provided in FEWS (Table 2). Each evaluation set (development and test) covers 10,000 different senses; half of these examples are few-shot and occur in the training set, and the other half of the evaluation senses are zero-shot. In comparison, the current benchmark for WSD evaluation (Raganato et al., 2017) only contains 796 unique zero-shot and 761 unique few-shot senses (where the sense is seen three or fewer times in the SemCor (Miller et al., 1993) training set) across development and test evaluation sets. This much larger sample of few- and zero-shot evaluation examples means that FEWS provides a robust setting to evaluate model performance on less common senses.

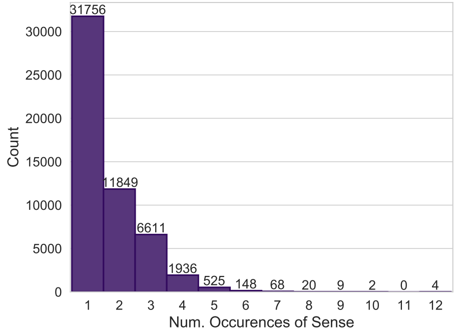


Figure 3: Histogram of sense frequencies in the FEWS training data.

Low-Shot Learning Because the data in FEWS come from example sentences for definitions in Wiktionary, each sense occurs in only a few labeled examples. This low-shot nature of the data is shown for five common words in Figure 2: each sense of these words occurs only one to four times in the training data. Figure 3 shows a histogram of the number of examples per sense in the full training set; we see that the majority of senses seen during training (60%) only occur once and that there are on average 1.65 examples per sense.

This set up also means that all evaluations on FEWS are low-shot (or zero-shot): senses in the few-shot development split have on average 2.06 supporting examples in the training set, with a maximum of 12 examples (these counts increase to 2.13 and 13, respectively, with the extended training data).

Figure 4: Word cloud of the 300 most common tags in the FEWS senses. For clarity, we manually filter out syntactic and uninformative tags.

Domains in FEWS Definitions in Wiktionary are tagged with keywords, which we include as metadata for their respective senses in the dataset. These keywords indicate that the senses in FEWS pertain to topics ranging from literature and archaic English to sports and the sciences and come from various English dialects (Figure 4). FEWS also covers many new domains not covered in existing WSD corpora, with six keywords corresponding to internet culture.

We also find that a number of the senses ($<1\%$ in FEWS) are indicated to be toxic or offensive language: our analysis contains tags such as *ethnic slurs*, *offensive*, *vulgar*, and *derogatory* that correspond to examples of toxic language. For many of these examples, the meaning of the target word is only toxic due to the context in which it appears. These examples provide an opportunity for improving models for hate speech detection and related tasks, but we leave this exploration to future work.

4 Baselines for FEWS

We run a series of baseline approaches on FEWS to demonstrate how current methods for WSD perform on this dataset. We consider a number of knowledge-based approaches (**Lesk**, **Lesk+emb**, and **MFS**) and two neural models that build on pretrained encoders (**Probe**$_{BERT}$ and **BEM**$_{BERT}$). We also ascertain how well humans perform on FEWS as a potential upper bound for model performance.

4.1 Knowledge-based Baselines

Most Frequent Sense (MFS) The MFS baseline assigns each target word in the evaluation with their candidate sense that is most frequently observed as the correct sense of that word in the training set. The MFS heuristic is known to be a particularly strong baseline in WSD datasets labeled on natural language documents (Kilgarriff, 2004); however, we expect this to be a weaker baseline on FEWS since the distribution of senses is much more balanced (and half of the evaluation senses are completely unseen during training).

Lesk The simplified **Lesk** algorithm assigns to each ambiguous target word the sense whose gloss has the highest word overlap with the context surrounding that target word (Kilgarriff and Rosenzweig, 2000). We specifically use the Lesk-definitions baseline from this work, meaning that we do *not* include words from example sentences in the set compared against the context – since these example sentences are used as the contexts in FEWS.

Lesk+emb This baseline is an extension of the above approach that incorporates word embeddings (Basile et al., 2014). A vector representation is built for the context around an ambiguous word (v_c) and the glosses of each sense of that word (v_g), where v_c and v_g are the element-wise sums of the word vectors for words in the context and gloss, respectively. The sense that corresponds to the v_g with the highest cosine similarity to v_c is chosen as the label for the target word. We use Glove embeddings (Pennington et al., 2014) for our implementation of this baseline.

4.2 Neural Baselines

Probe$_{BERT}$ This baseline is a linear classifier trained on contextualized representations output by the final layer of a frozen pretrained model; we use BERT as our pretrained encoder (Devlin et al., 2019). We train this classifier by performing a softmax over all of the senses in the Wiktionary sense inventory and mask out any senses not relevant to the target word.

BEM Our other neural baseline is the biencoder model (BEM) for WSD introduced by Blevins and Zettlemoyer (2020). The BEM has two independent encoders, a *context encoder* that processes the context sentence (including the target word) and a *gloss encoder* that encodes the glosses of senses into a sense representation. The BEM takes the dot product of the target word representation from the *context encoder* and sense representations from the gloss encoder, and it labels the target word with the sense that has the highest dot product score. We train **BEM**$_{BERT}$ by initializing each encoder with BERT and training on the FEWS train set.

4.3 Human Performance

Finally, we calculate the estimated human performance on the FEWS development set. The three human annotators were native English speakers, who each completed the same randomly chosen 300 example subset of the development set. The examples were sampled such that half (150) of these examples came from the few-shot split and the other half came from the zero-shot split. Similar to the modeling baselines, we evaluate each annotator's performance by scoring them against the

	Dev			Test		
	Full Set	**Few-shot**	**Zero-shot**	**Full Set**	**Few-shot**	**Zero-shot**
Knowledge-based baselines						
MFS	26.39	52.78	0.00	25.73	51.46	0.00
Lesk	39.24	42.54	35.94	39.07	40.94	37.20
Lesk+emb	42.54	44.94	40.14	41.53	44.08	38.98
Human†	80.11	80.44	79.87	–	–	–
Neural baselines						
Probe$_{BERT}$	36.17	72.34	0.00	36.07	72.14	0.00
BEM$_{BERT}$	73.81	79.28	68.34	72.77	79.06	66.48
BEM$_{SemCor}$	74.36	79.72	69.00	72.98	78.88	67.08
BEM$_{zero-shot}$‡	58.05	58.38	57.72	57.39	57.94	56.84

Table 3: Accuracy (%) of our baselines (Section 4) and transfer learning models (Section 6) on the FEWS evaluation sets. Human performance (†) is calculated on a subset of the development set and acts as an estimated upper bound on performance. **Probe**$_{BERT}$ and **BEM**$_{BERT}$ are baselines trained on FEWS; **BEM**$_{SemCor}$ is a transfer learning model finetuned on SemCor before training on FEWS. **BEM**$_{zero-shot}$ (‡) is a zero-shot transfer experiment in which the BEM trained on SemCor is evaluated on FEWS *without* finetuning on the FEWS train set.

sense associated with that example in Wiktionary (which we assume to be gold labels).

5 Baseline Experiments

5.1 Experimental Setup

Data All baselines for the FEWS dataset are trained on the *train* set unless specifically stated to have been trained on the *extended train* set. All models for FEWS are tuned using the *development* set and then evaluated on the held-out *test* set.

Experimental Details Our probe and BEM baselines are in implemented in PyTorch[5] and optimized with Adam (Kingma and Ba, 2015). For the BEM, we use the implementation provided by Blevins and Zettlemoyer (2020).[6] We obtain the *bert-base-uncased* encoder from Wolf et al. (2019) to get the BERT output representations for the probe and to initialize the BEM models. Further hyperparameter details are given in Appendix A.

5.2 Modeling Results

Table 3 shows the results of our baseline experiments on FEWS. We find that the MFS baseline is weak overall, primarily because it is unable to predict anything about the held-out, zero-shot senses; the Lesk algorithms both outperform this baseline in the overall setting, with the Lesk+emb approach scoring slightly better than the original Lesk approach by 1.78-4.2% across the different evaluation subsets. However, on the few-shot examples in both the development and test sets, we see that the MFS baseline outperforms both of the Lesk baselines. This shows that, for the few-shot examples, the MFS heuristic remains a reasonably strong baseline even with the more uniform sense distribution of FEWS (and indicates that the distribution of examples drawn from the dictionary is less uniform than expected).

The neural baselines we run generally outperform the knowledge-based ones. The **Probe**$_{BERT}$ model does fairly well on the few-shot examples, outperforming the MFS baseline by about 20 accuracy points; however, it is unable to disambiguate words in the zero-shot splits correctly since the probe can not generalize to unseen senses. In comparison, **BEM**$_{BERT}$ performs well across the entire evaluation set. In particular, the BEM achieves much better zero-shot performance than other baselines, though performance on this subset still lags behind few-shot performance. Finally, we see that humans perform better than all of the considered baselines, particularly on zero-shot senses where humans outperform the BEM by 11.53 points. More details about the human evaluation are given below (Section 5.3).

Additionally, we find that training on the extended train set has little effect on these baselines: the **MFS** and **Probe**$_{BERT}$ baselines perform slightly worse (with deltas of -0.08% and -0.21% on the test set, respectively), and **BEM**$_{BERT}$ performs 1.05% better. Appendix B presents full re-

---

[5] https://pytorch.org/

[6] https://github.com/facebookresearch/wsd-biencoders

	Dev Subset		
	Full Set	Few-shot	Zero-shot
Knowledge-based baselines			
MFS	23.66	47.33	0.00
Lesk	38.33	41.33	35.33
Lesk+emb	46.00	48.67	43.33
Human	80.11	80.44	79.87
Neural baselines			
Probe$_{BERT}$	33.67	67.33	0.00
BEM$_{BERT}$	73.00	80.66	65.33

Table 4: Accuracy (%) of our baselines on the subset of the development set manually scored by human annotators.

sults for the extended train baselines.

5.3 Human Evaluation Results

The human annotators achieved an average accuracy of 80.11% (with each annotator getting 84.67%, 78.67%, and 77.00% accuracy) and an average inter-annotator agreement of $\kappa = 0.802$. We find that humans perform slightly better on the examples that correspond to few-shot examples for the dataset than those corresponding to zero-shot examples despite not using the training data, with an average of 80.44% and 79.87% accuracy on those two subsets, respectively.

We also report the performance of the baselines for FEWS on the same subset of the development set that was manually completed by humans (Table 4). We find that the baselines perform similarly on this subset, with a small decrease in performance compared to the full development set (with decreases in accuracy ranging between 0.81% and 3.46% when compared to the development set).

6 Transfer Learning with FEWS

Next, we investigate how useful FEWS is at improving WSD performance on existing benchmarks. We perform transfer learning experiments by iteratively finetuning models on FEWS and the WSD Evaluation Framework (Raganato et al., 2017), with one acting as the *intermediate* dataset and evaluating performance on the other, *target* dataset. We find that on global metrics, this approach performs similarly to finetuning only on the training data for each benchmark; however, transferring from FEWS to the WSD Framework improves performance on less-frequent and zero-shot senses. This suggests that FEWS provides valuable WSD information not covered in SemCor.

6.1 Experimental Setup

We apply the supplementary training approach presented in Phang et al. (2018) to perform our transfer learning experiments. We initialize a BEM with the best model developed on the *intermediate* dataset and evaluate on the *target* dataset in two ways: first, by evaluating this BEM on the *target* dataset in a zero-shot manner, without additional finetuning; and second by finetuning the BEM on the *target* training data before performing the *target* evaluation. We refer to these models as **BEM**$_{zero-shot}$ and **BEM**$_{intermediate}$, respectively. As a baseline, we also compare against the best BEM trained only on the *target* dataset (with no exposure to the *intermediate* dataset), **BEM**$_{BERT}$.[7] The model implementation details are identical to those for the baseline experiments on FEWS (Section 5.1).

6.2 Data

Models that are finetuned for the WSD Evaluation Framework are trained using SemCor, a large dataset annotated with WordNet synsets and commonly used for training WSD models (Miller et al., 1993). Following previous work, we use SemEval-2007 as a validation set (**SE07**; (Pradhan et al., 2007a)) and hold out the other evaluations sets in the Framework (Senseval2 (**SE2**; (Palmer et al., 2001)), Senseval-3 (**SE3**; (Snyder and Palmer, 2004)), SemEval-2013 (**SE13**; (Navigli et al., 2013)), and SemEval-2015 (**SE15**; (Moro and Navigli, 2015)) as test sets. Similarly to the baseline experiments, models that are trained on FEWS use the *train* set (note that we do not use the *extended train* set in these experiments), are validated on the *development* set, and finally evaluated on the held out *test* set.

6.3 Results

FEWS Results We first consider the setting where SemCor acts as the *intermediate* dataset and FEWS as the *target* (Table 3). We find that **BEM**$_{SemCor}$ performs similarly to training only on FEWS (with an improvement of 0.21% on the test set). **BEM**$_{zero-shot}$, which is not finetuned on

---

[7] We note that this is a naive approach to transferring between Wiktionary senses and WordNet synsets, and that it is possible that better transfer learning could occur with a more complicated multitask approach or by mapping between the two lexical resources with approaches such as Navigli and Ponzetto (2012) or Miller and Gurevych (2014).

	Dev	**Test Datasets**				**Concatenation of all Datasets**				
	SE07	SE2	SE3	SE13	SE15	Nouns	Verbs	Adj.	Adv.	ALL
MFS (in train data)	54.5	65.6	66.0	63.8	67.1	67.7	49.8	73.1	80.5	65.5
BEM$_{BERT}$	**74.5**	**79.4**	77.4	**79.7**	**81.7**	**81.4**	68.5	**83.0**	87.9	**79.0**
BEM$_{FEWS}$	73.6	79.1	**77.9**	79.1	81.6	81.2	**68.9**	81.8	**88.2**	78.8
BEM$_{zero-shot}$	53.0	66.6	62.3	69.1	74.9	70.7	51.2	72.0	69.7	66.4

Table 5: F1-score on the English all-words WSD in the WSD Evaluation Framework (Raganato et al., 2017). We compare the best model from Blevins and Zettlemoyer (2020) (BEM$_{BERT}$) against (1) a model first trained on FEWS and then trained on SemCor (BEM$_{FEWS}$) and (2) a model trained on FEWS and evaluated on this task without further finetuning (BEM$_{zero-shot}$).

	MFS	**LFS**	**Zero-shot**	
			Words	**Senses**
WordNet S1	100.0	0.0	84.9	53.9
BEM$_{BERT}$	**94.1**	52.6	91.2	68.9
BEM$_{FEWS}$	93.7	52.9	92.2	74.8
BEM$_{zero-shot}$	72.6	**55.5**	**92.7**	**80.5**

Table 6: F1-score on the MFS, LFS, and zero-shot subsets of the **ALL** evaluation set from the WSD Evaluation Framework. Zero-shot examples are the words and senses (respectively) from the evaluation suite that do not occur in SemCor.

the FEWS train set, unsurprisingly performs worse than any of the BEMs that saw the FEWS training data but outperforms the Lesk baselines.

WSD Evaluation Framework Results We then consider the opposite setting, where FEWS is the *intermediate* dataset and the WSD Evaluation Framework acts as the *target* evaluation (Table 5). On the overall evaluation set, we again see that **BEM**$_{FEWS}$ performs similarly to the **BEM**$_{BERT}$ baseline on the overall **ALL** metric, and that the zero-shot BEM model underperforms the other bi-encoders.

We then break down performance on the *target* evaluation set by sense frequencies: we evaluate performance on the most frequent sense (MFS) of each word in the evaluation (i.e., the sense each word takes most frequently in the SemCor training set); the less frequent senses (LFS) of words, or any sense a words takes besides its MFS; zero-shot words that are not seen during training on SemCor; and zero-shot senses, also not seen during training (Table 6). We find that both transfer learning models perform better on LFS and zero-shot examples than **BEM**$_{BERT}$.[8]

In particular, the zero-shot transfer model does well on these subsets, demonstrating that a fair amount of WSD knowledge about uncommon senses can be transferred in a zero-shot manner between these datasets (albeit at the expense of higher performance on the MFS subset). This result also shows how much the MFS group dominates existing WSD metrics and highlights the need for focused evaluations of other types of word senses. Finally, we see that even without exposure to the natural sense distribution in natural language texts, the zero-shot model still performs significantly better on the MFS of words than the LFS, with a 17.1 F1 point difference between the two subsets; this is likely because the BERT encoder is exposed to the sense distribution of English natural language documents during pretraining.

7 Conclusion

We establish baseline performance on FEWS with both knowledge-based approaches and recently published neural WSD models. Unsurprisingly, neural models based on pretrained encoders perform best on FEWS; however, the human evaluation shows there is still room for improvement, particularly for zero-shot senses. Finally, we also present results on transferring word sense knowledge from FEWS onto existing WSD datasets with staged finetuning. While our naive approach for transferring knowledge from FEWS does not improve performance on the global WSD metric, adding FEWS as an additional training signal improves performance on uncommon senses in existing evaluation sets.

We hope that FEWS will inspire future work fo-

---

[8] For the models trained on FEWS, it is possible they have seen closely related senses to those in the zero-shot subsets of from the Wiktionary sense inventory; however, these are represented differently in each dataset and correspond to different definitions to be encoded by the biencoder.

cusing on better methods for capturing rare senses in WSD and better modeling of word sense in niche domains like internet culture or technical writing.

Acknowledgements

This material is based on work conducted at the University of Washington, which was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1762114. The authors thank Julian Michael, Maarten Sap, and the UW NLP group for helpful conversations and comments on the work.

References

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2014. An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1591–1600.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

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

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. 2012. Webcage: a web-harvested corpus annotated with germanet senses. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 387–396. Association for Computational Linguistics.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuan-Jing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3500–3505.

Adam Kilgarriff. 2004. How dominant is the commonest sense of a word? In *International conference on text, speech and dialogue*, pages 103–111. Springer.

Adam Kilgarriff and Joseph Rosenzweig. 2000. Framework and results for english senseval. *Computers and the Humanities*, 34(1-2):15–48.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of 3rd International Conference of Learning Representations*.

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha Talukdar. 2019. Zero-shot word sense disambiguation using sense definition embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5670–5681, Florence, Italy. Association for Computational Linguistics.

Daniel Loureiro and Jose Camacho-Collados. 2020. Don't neglect the obvious: On the role of unambiguous words in word sense disambiguation. *arXiv preprint arXiv:2004.14325*.

Michael Matuschek and Iryna Gurevych. 2013. Dijkstra-wsa: A graph-based approach to word sense alignment. *Transactions of the Association for Computational Linguistics*, 1:151–164.

Christian M Meyer and Iryna Gurevych. 2011. What psycholinguists know about chemistry: Aligning wiktionary and wordnet for increased domain coverage. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 883–892.

Rada Mihalcea. 2007. Using Wikipedia for automatic word sense disambiguation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 196–203, Rochester, New York. Association for Computational Linguistics.

George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*, pages 303–308. Association for Computational Linguistics.

Tristan Miller and Iryna Gurevych. 2014. WordNet—Wikipedia—Wiktionary: Construction of a three-way alignment. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2094–2100, Reykjavik, Iceland. European Language Resources Association (ELRA).

Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 288–297.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):10.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. Semeval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Marten Postma, Ruben Izquierdo Bevia, and Piek Vossen. 2016. More is not always better: balancing sense distributions for all-words word sense disambiguation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3496–3506.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007a. Semeval-2007 task-17: English lexical sample, srl and all words. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pages 87–92.

Sameer S Pradhan, Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2007b. Ontonotes: A unified relational semantic representation. In *International Conference on Semantic Computing (ICSC 2007)*, pages 517–526. IEEE.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110.

Vincent Segonne, Marie Candito, and Benoît Crabbé. 2019. Using Wiktionary as a resource for WSD : the case of French verbs. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 259–270, Gothenburg, Sweden. Association for Computational Linguistics.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

		Dev			Test		
		Full Set	**Few-shot**	**Zero-shot**	**Full Set**	**Few-shot**	**Zero-shot**
MFS	Ext. Train	26.13	52.26	0.00	25.65	51.30	0.00
	Train	26.39	52.78	0.00	25.73	51.46	0.00
	Δ	-0.26	-0.52	–	-0.08	-0.16	–
Probe$_{BERT}$	Ext. Train	36.03	72.06	0.00	35.86	71.72	0.00
	Train	36.17	72.34	0.00	36.07	72.14	0.00
	Δ	-0.14	-0.28	–	-0.21	-0.42	–
BEM$_{BERT}$	Ext. Train	74.12	79.38	68.86	73.82	79.70	67.94
	Train	73.81	79.28	68.34	72.77	79.06	66.48
	Δ	0.31	0.10	0.52	1.05	0.64	1.46

Table 7: Accuracy of the FEWS baselines trained on the extended train set. In each group of rows, we report (1) the extended train baseline, (2) the comparable baseline trained on the standard train set, and (3) the performance delta between the two (where a positive delta indicates the extended train baseline performs better).

A Model Hyperparameters

Each model reported in this paper was tuned on a single hyperparameter sweep over the reported ranges and chosen based on the appropriate development set metric (accuracy on FEWS, F1 performance on the Unified WSD Framework).

Probe$_{BERT}$ The linear layer in the BERT probe baseline is trained for 100 epochs. It is tuned over a range learning rates ($[5e-6, 1e-5, 5e-5, 1e-4]$, with a final learning rate of $1e-4$). We use a batch size of 128 to train this probe.

BEM For the biencoder model (BEM), we use the codebase provided by (Blevins and Zettlemoyer, 2020). Following this work, we train the BEM for 20 epochs with a warmup phase of 10,000 steps; we use a context batch size of 4 and a gloss batch size of 256. Each BEM is tuned over the following learning rates: $[1e-6, 5e-6, 1e-5, 5e-5]$. The BEM$_{BERT}$ and BEM$_{SemCor}$ had a final learning rate of $5e-6$, and the BEM$_{FEWS}$, of $1e-6$.

B Extended Train Baselines

The extended train set in FEWS contains all of the quotation-based examples from the train set as well as the additional, shorter example *illustrations* that are written by Wiktionary editors to exemplify a particular sense of a word. We retrain the **MFS**, **Probe**$_{BERT}$, and **BEM**$_{BERT}$ baselines on this extended training set; the other baselines we consider (**Lesk** and **Lesk+emb**) are calculated without using either of the training sets.

Table 7 compare the extended train baselines against those trained on the standard train set. For the **MFS** and **Probe**$_{BERT}$ baselines, we find that adding the extra, stylistically different *illustrations* in extended train slightly hurts performance. However, the stronger **BEM**$_{BERT}$ is able to better use this data and achieves somewhat stronger performance with additional training data. Notably, most of this improvement in the **BEM**$_{BERT}$ comes from the zero-shot evaluation setting, even though the extended train set does not contain any of these zero-shot senses.

MONAH: Multi-Modal Narratives for Humans to analyze conversations

Joshua Y. Kim
University of Sydney
New South Wales, Australia
josh.kim@sydney.edu.au

Greyson Y. Kim
Success Beyond Pain
Western Australia, Australia
greyson.kim@gmail.com

Chunfeng Liu
Hello Sunday Morning
New South Wales, Australia
ian@hellosundaymorning.org

Rafael A. Calvo
Imperial College London
London, United Kingdom
r.calvo@imperial.ac.uk

Silas C.R. Taylor
University of New South Wales
New South Wales, Australia
silas.taylor@unsw.edu.au

Kalina Yacef*
University of Sydney
New South Wales, Australia
kalina.yacef@sydney.edu.au

Abstract

In conversational analyses, humans manually weave multimodal information into the transcripts, which is significantly time-consuming. We introduce a system that automatically expands the verbatim transcripts of video-recorded conversations using multimodal data streams. This system uses a set of preprocessing rules to weave multimodal annotations into the verbatim transcripts and promote interpretability. Our feature engineering contributions are two-fold: firstly, we identify the range of multimodal features relevant to detect rapport-building; secondly, we expand the range of multimodal annotations and show that the expansion leads to statistically significant improvements in detecting rapport-building.

1 Introduction

Dyadic human-human dialogs are rich in multimodal information. Both the visual and the audio characteristics of how the words are said reveal the emotions and attitudes of the speaker. Given the richness of multimodal information, analyzing conversations requires both domain knowledge and time. The discipline of conversational analysis is a mature field. In this discipline, conversations could be manually transcribed using a technical system developed by Jefferson (2004), containing information about intonation, lengths of pauses, and gaps. Hence, it captures both *what* was said and *how* it was said[1]. However, such manual annotations take a great deal of time. Individuals must watch the conversations attentively, often replaying the conversations to ensure completeness.

Automated Jefferson (2004) transcripts could be generated from video-recordings (Moore, 2015). However, the potential issue with Jeffersonian annotations is that there are often within-word annotations and symbols which makes it hard to benefit from pre-trained word embeddings. Inspired by the Jeffersonian annotations, we expand the verbatim transcripts with multimodal annotations such that downstream classification models can easily benefit from pre-trained word embeddings.

Our paper focuses on the classification task of predicting rapport building in conversations. Rapport has been defined as a state experienced in interaction with another with interest, positivity, and balance (Cappella, 1990). If we can model rapport building in the medical school setting, the volunteer actors can let the system give feedback for the unofficial practice sessions, and therefore students get more practice with feedback. Also, the lecturer could study the conversations of the top performers and choose interesting segments to discuss. As student doctors get better in rapport building, when they graduate and practice as doctors, treatments are more effective and long-term (Egbert et al., 1964; DiMatteo, 1979; Travaline et al., 2005).

Outside of the healthcare domain, understanding and extracting the features required to detect rapport-building could help researchers build better conversational systems. Our first contribution is the identification of multimodal features that have been found to be associated with rapport building and using them to predict rapport building automatically. Our second contribution is to include them into a text-based multimodal narrative system (Kim et al., 2019b). Why go through text? It is because this is how human experts have been manually analyzing conversations in the linguistics community. Our text-based approach has the merit of emulating the way human analysts analyze conversations, and hence supporting better interpretability. We demonstrate that the additions bring statistically significant improvements. This feature-

---

* Corresponding author

[1] Please visit www.universitytranscriptions.co.uk/jefferson-transcription-example/ for an audio example.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 466–479
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

engineering system[2] could potentially be used to accomplish a highly attention-demanding task for an analyst. With an automated text-based approach, we aim to contribute towards the research gap of automatic visualizations that support multimodal analysis (Kim et al., 2019a). The created multimodal transcript itself is a conversational analysis product, which can be printed out on paper.

In this paper, we first introduced the problem domain (section 3). Secondly, we motivated the new features (detailed in Fig. 1) to be extracted (section 4). Then, we extracted the features from videos and encoded them as text together with verbatim transcripts (section 4). To evaluate whether the text narratives were useful, we ran experiments that predict rapport-building using texts containing different amounts of multimodal annotations (section 5). Finally, we discuss the results and visualize the outputs of the system (section 6).

2 Related Works

The automated analysis of conversations has been the subject of considerable interest in recent years. Within the domain of doctor-patient communication, Sen et al. (2017) calculated session-level input features, including affective features (Gilbert, 2014). Analyses using session-level features have a drawback of not being able to identify specific defining multimodal interactions in the conversation (Zhao et al., 2016; Heylen et al., 2007). Therefore, we build upon the works of Sen et al. (2017) – in addition to the use of session-level features, we propose using a finer level of talk-turn multimodal text representation as inputs into a hierarchical attention network (HAN) (Yang et al., 2016).

We also build upon our previous work (Kim et al., 2019b) by broadening the range of multimodal features considered. As for the different methods of multimodal information fusion, Poria et al. (2017) completed an extensive review of the different state-of-the-art multimodal fusion techniques. Recent multimodal fusion research (such as ICON (Hazarika et al., 2018a), CMN (Hazarika et al., 2018b), MFN (Zadeh et al., 2018), DialogueRNN (Majumder et al., 2019), M3ER (Mittal et al., 2020)) has focussed on end-to-end approaches. Unlike the typical end-to-end approach of representing and fusing multimodal features using numeric vectors, our contribution is an entirely text-based multimodal narrative, thereby improving downstream analysis's interpretability. The approach of this system not only annotates the presence of nonverbal events (Eyben et al., 2011), but also the degree of the nonverbal event intensity at both the session-level and talkturn-level.

3 Data

This study uses data from the EQClinic platform (Liu et al., 2016). Students in an Australian medical school were required to complete at least one medical consultation on the online video conferencing platform EQClinic with a simulated patient who is a human actor trained to act as a patient. Each simulated patient was provided with a patient scenario, which mentioned the main symptoms experienced. The study was approved by the Human Research Ethics Committee of the University of New South Wales (project number HC16048).

The primary outcome measurement was the response to the rapport-building question on the Student-Patient Observed Communication Assessment (SOCA) form, an adapted version of the Calgary-Cambridge Guide (Kurtz and Silverman, 1996). Simulated patients used the SOCA form to rate the students' performances after each video consultation. Our dataset comprises of 873 sessions, all from distinct students. Since we have two recordings per session (one of the student, the second of the simulated patient), the number of recordings analyzed is 1,746. The average length per recording is 928 seconds (sd=253 seconds), amounting to a total of about 450 hours of recordings analyzed. The dataset's size is small relative to the number of multimodal features extracted; therefore, there is a risk of overfitting.

We used the YouTube platform to obtain the transcript per speaker from the recordings. We chose YouTube because we (Kim et al., 2019c) found that it was the most accurate transcription service (word error rate: 0.28) compared to Google Cloud (0.34), Microsoft Azure (0.40), Trint (0.44), IBM Watson (0.50), when given dyadic video-conferences of an Australian medical school. Jeong-Hwa and Cha (2020) found that among the four categories of YouTube errors (omission, addition, substitution, and word order), substitution recorded the highest amount of errors. Specifically, they found that phrase repetitions could be mis-transcribed into non-repetitions. From our experience, (a) repair-initiation techniques such as sound stretches (e.g. "ummmm") (Hosoda, 2006), were either omitted or

---

[2] Open-sourced at https://github.com/SpectData/MONAH

substituted with “um”; (b) overlapping speech was not a problem because our speakers were physically separated and recorded into separate files.

We brought together the two speakers’ transcripts into a session-level transcript through word-level timings and grouped together words spoken by one speaker until the sequence is interrupted by the other speaker. When the interruption occurs, we deem that the talk-turn of the current speaker has ended, and a new talk-turn by the interrupting speaker has begun. The average number of talk-turns per session is 296 (sd=126), and the average word count per talk-turn is 7.62 (sd=12.2).

At this point, we note that acted dialogues differ from naturally occurring dialogues in a few ways. Firstly, naturally occurring dialogues tend to be more vague (phrases like “sort of”, “kinda”, “or something”) due to the shared understanding between the speakers (Quaglio, 2008). Secondly, taboo words or expletives that convey emotions (like “shit”, “pisssed off”, “crap”) is likely to be less common in an acted medical setting than naturally occurring conversations. Some conversations transform into genuine dialogues where the speakers “shared parts of themselves they did not reveal to everyone and, most importantly, this disclosure was met with acceptance” (Montague, 2012). This definition of genuine conversation is similarly aligned to our definition of rapport-building in section 4.1.

Figure 1 shows a summary of the features extracted. We annotated verbatim transcripts with two different levels of multimodal inputs – annotations at the session-level are labeled *coarse*, whilst annotations at the talk-turn-level are labeled *fine*. To facilitate comparisons, all input families belonging to the *coarse* (*fine*) level would be annotated with uppercase (lowercase) letters, respectively. In this paper, we refer to the previously existing set of features (with white background) as the “prime” ($'$) configuration. Families are also abbreviated by their first letter. For example, the *coarse* P' family would consist of only speech rate and delay, whilst the *coarse* P family would consist of P' plus tone. As another example, the *coarse* D' family is the same as the D family because there are no newly added features (in blue). We introduce the framework of our multimodal feature extraction pipeline in Figure 2.

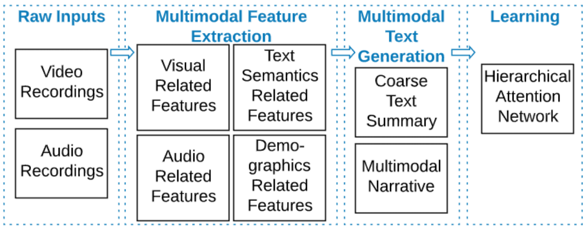


Figure 2: MONAH (Multi-Modal Narratives for Humans) Framework.

4 Multimodal features extractions

As an overview, we extracted the timestamped verbatim transcripts and used a range of pre-trained models to extract temporal, modality-specific features. We relied on pre-trained models for feature extraction and did not attempt to improve on them – demonstrating the value of using multidisciplinary pre-trained models from natural language processing, computer vision, and speech processing for conversational analysis.

Effectively, we extracted structured data from unstructured video data (section 4.2). With the structured data and verbatim transcript, we weaved a multimodal narrative using a set of predefined templates (sections 4.3 and 4.4). With the multimodal narrative, we employed deep learning techniques and pre-trained word embeddings to predict the dependent variable (section 5).

4.1 Dependent variable - rapport building

The dependent variable is defined as the success in rapport building. Rapport building is one of

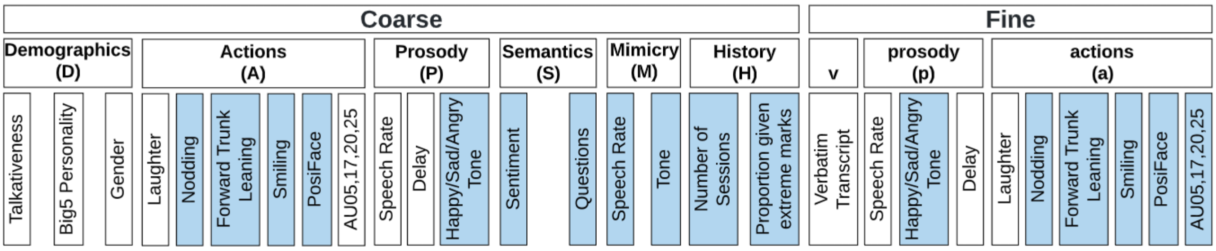


Figure 1: High-level features introduction. We build on our previous work (Kim et al., 2019b) – the new features introduced in this work are coloured in blue, whilst the existing set of features are in white.

the four items scored in the SOCA. The original 4-point Likert scale is Fail, Pass-, Pass, Pass+, we converted this scale into a binary variable where it is true if the rapport-building score is "Pass+" as we are concerned here with identifying good rapport building. "Pass+" means that the actor felt rapport such that all information could be comfortably shared. 38 percent of the population has achieved "Pass+". All actors followed the same pre-interview brief. Because only the actor scored the student performance and there is no overlap, the limitation is that we do not have measures of agreement.

4.2 Description of features

Table 1 gives an overview of all features for each speaker. We define six families of *coarse*-level inputs -– *demographics*, *actions*, *prosody*, *semantics*, *mimicry*, and *history*. We computed the features per speaker. From all families, there are a total of 77 features per session.

We first discuss the family of *demographics*. *Talkativeness* is chosen because the patient's talkativeness would initiate the doctor's active listening while aiding identification of patient's concerns – processes that could establish rapport. In Hall et al. (2009), it appears that patients appreciate a certain degree of doctor's dominance in the conversation, which itself is also correlated with higher rapport. *Big 5 Personality* consists of Extraversion, Agreeableness, Conscientiousness, Neuroticism, and Openness to Experience (McCrae and Costa, 1987). This personality structure is widely used in research and practice to quantify aspects of a person's natural tendency in thought, feeling, and action, with good validity and reliability indicators (McCrae, 2017). It is chosen because traits of agreeableness and openness on the part of both doctor and patient predict higher rapport. Among doctors, higher openness and agreeableness predict higher empathy towards patients (Costa et al., 2014). Among patients, higher agreeableness predicted higher trust towards doctors (Cousin and Mast, 2013), and higher openness predicted higher doctor affectionate communication (Hesse and Rauscher, 2019). *Big 5 Personality* is extracted through feeding transcripts to the IBM Watson Personality Insights API (version 2017-10-13), costing a maximum of 0.02 USD per call. *Gender* is chosen because personality differences between genders were observed cross-culturally. Among twenty-three thousand participants across cultures

Family	Child	Template
Demo graphics	Talkativeness	Total word count, total distinct word count, and proportion of word count
	Big 5 Personality	Percentile scores for each of the big 5 personality
	Gender	Male or Female
Actions	Laughter	Total laughter count
	Head Nodding*	Count of nods
	Forward Trunk Leaning*	Count of leaning in
	Smiling*	Count of smiles
	PosiFace*	Counts of times of positive and negative facial expressions
	AU	Summary statistics of the selected AU (05,17,20,25) intensities
Prosody	Delay	Summary statistics of time gaps between talk-turns
	Speech rate	Average speech rate
	Tone*	Happy, sad, angry tone
Semantics	Sentiment*	Composite, positive, neutral, and negative sentiment
	Questions*	Proportion of talk-turns that are open/closed questions
Mimicry	Speech Rate*	Dynamic time wrapping distance for speech rate
	Tone*	Dynamic time wrapping distance for tone
History	Num. Sessions*	Number of past sessions the assessor has scored before this
	Proportion given extreme marks*	Proportion of past sessions that the assessor has given an extreme score

Table 1: Session-level input features for each participant. * indicates new features outside of Kim et al. (2019b).

for both college-age and adult samples, females reported higher agreeableness, warmth, and openness to feelings than males (Costa Jr et al., 2001), traits that could be linked to rapport building.

Secondly, for the family of *actions*, *laughter* is chosen because humor (which was defined in part by the presence of laughter) on the part of both doctor and patient was found to be twice as frequent in high-satisfaction than low-satisfaction visits (Sala et al., 2002). Laughter events were detected using the Ryokai et al. (2018) algorithm. *Facial expressions* that resemble smiling is another behavioral indicator of humor appreciation, and approval of one another (Tickle-Degnen and Rosenthal, 1990). *Head nodding* is a type of backchannel response (i.e., response tokens) that has been shown to reflect rapport between doctor and patient, especially when the primary activity is face to face communications (Manusov, 2014). *Forward trunk leaning* is chosen because it has long been found to reflect an expression of interest and caring, which are foundational to rapport building (Scheflen, 1964). Additionally, facial positivity (*posiface*) is included as it is useful in rapport building detection in small groups (Müller et al., 2018). Lastly, *action units* (AU) that describe specific facial expressions, in particular AU 05 (upper lid raiser), 17 (chin raiser), 20 (lip stretcher), 25 (lips part), are also included as they were useful in automated dyadic conversational analyses to detect depression in our previous work (Kim et al., 2019b). All features introduced in this paragraph were calculated using the AU and landmark positioning features extracted using OpenFace (Baltrušaitis et al., 2016).

Thirdly, for the family of *prosody*, *delay* is chosen because it has been shown to be an indicator of doctor-to-patient influence – patients of low rapport with their doctors were found to speak less in response to doctor's comments (Sexton et al., 1996). *Speech rate* is chosen because doctor's fluent speech rate and patient's confident communication have been positively correlated with the patient's perception of rapport (Hall et al., 2009). Delay and speech rate are calculated using the time-stamped transcripts. *Tone* is chosen because a warm and respectful tone on the part of both doctor and patient is positively correlated with the patient's perception of rapport (Hall et al., 2009). Tone is calculated using the Vokaturi algorithm (version 3.3) (Vokaturi, 2019).

Fourthly, for the family of *semantics*, *sentiment* is chosen because the provision of positive regard from a practitioner to a patient is an important factor to foster therapeutic alliance; additionally, this process may be further enhanced if the patient also demonstrates positive behaviors towards the practitioners (Farber and Doolin, 2011). Sentiment is extracted using the VADER algorithm (Gilbert, 2014), in line with Sen et al. (2017). *Questions* is chosen because higher engagement by the doctor (e.g., asking questions) with the patient and the patient asking fewer questions have been shown to positively correlate with the patient's perception of rapport (Hall et al., 2009). Questions are detected using Stanford CoreNLP Parser (Manning et al., 2014) and the Penn Treebank (Bies et al., 1995) tag sets.

Next, *mimicry* is chosen because doctor-patient synchrony is an established proxy for rapport. In a review paper, rapport is theorized to be grounded in the coupling of practitioner's and patient's brains (Koole and Tschacher, 2016). Such a coupling process would eventuate in various forms of mimicry in the dyad, for instance, vocally (e.g., matching speech rate and tone), physiologically (e.g., turn-taking, breathing), physically (e.g., matching body language) (Wu et al., 2020). In this study, we aim to use vocal mimicry to capture this underlying phenomenon. Session level mimicry scores are approximated through Dynamic Time Wrapping distances (Giorgino and others, 2009), in line with Müller et al. (2018).

Lastly, *history* is chosen because the scores given by the assessors could be subjective evaluations where the evaluations are unduly influenced by the assessor's leniency bias (Moers, 2005). We attempted to mitigate the leniency bias by introducing history features that indicate the assessor's leniency and its consistency.

4.3 Generation of *coarse* multimodal narrative

In this section, we discuss the *coarse* multimodal narrative. We summarized the automatic generation of the text representation in Table 2.

We calculated the z-score for all the above templates (except Template 3 which is categorical) using the following z-score formula. The average (μ), and standard deviation (σ) are computed using observations from the training observations. Using the z-score, we bucketed them into "very low" (z<-2), "low" (z<-1), "high" (z>1) and "very high"

Family	Child	ID	Template
Demo graphics	Talkativeness	1	**doctor** number of words **high**, **doctor** number of distinct words high
	Big 5 Personality	2	**doctor** openness **high**
	Gender	3	The **patient** is **female**
Actions	Laughter	4	**doctor** laughter counts **high**
	Head Nodding	5	**doctor** head nod counts **high**
	Forward Trunk Leaning	6	**doctor** forward trunk leaning **high**
	Smiling	7	**doctor** smiling counts **high**
	PosiFace	8	**doctor** positive face expression counts **high**
	AU	9	**doctor** minimum lip depressor **very low**, maximum lip depressor **low**, average lip depressor **low**, variance lip depressor **low**
Prosody	Delay	10	minimum delay **very low**, maximum delay **low**, average delay **low**, variance delay **low**
	Speech rate	11	speech rate **high**
	Tone	12	angry tone **high**
Semantics	Sentiment	13	positive sentiment **high**
	Questions	14	open questions **high**
Mimicry	Speech Rate	15	speech rate mimicry **high**
	Tone	16	tone mimicry **high**
History	Num. Sessions	17	patient number of sessions before this **very high**
	Proportion given extreme marks	18	patient question four proportion given maximum marks **high**

Table 2: Templates for the session-level *coarse* summary.

(z>2). The reason for z-transformation is to create a human-readable text through bucketing continuous variables into easy-to-understand buckets ("high" vs. "low").

$$z = \frac{x - \mu_{\text{Train}}}{\sigma_{\text{Train}}} \tag{1}$$

4.4 Generation of *fine* multimodal narrative

In addition to the verbatim transcript, we introduced two new families of information – *prosody*, and *actions*. Table 3 gives an overview of the templates, and the bold-face indicates a variable. The motivations of the features have been discussed; we discuss the rules of insertion in the next few paragraphs.

Template 19 is the verbatim transcript returned from the ASR system. Before each talk-turn, we identified the speaker (doctor/patient) and added multimodal information using templates 20-29. Speech rate and tone were standardized across all training observations. We appended template 20, 21 where possible values are dependent on the z-score – "quickly" ($1 <$ z-score < 2) and "very quickly" (z-score ≥ 2). For delay, we used time intervals of 100 milliseconds, and between 200 and 1200 milliseconds – in line with Roberts and Francis (2013). We appended template 22 at the front of the talk-turn if a delay of at least 200 milliseconds is present between talk-turns. In addition, we appended template 23 where possible values are dependent on the standardized duration of delay – "short" ($<$ 1 z-score), "long" ($<$ 2 z-score) and "significantly long" ($\geq$ 2 z-score). Template 23 captures longer than usual delay, considering the unique turn-taking dynamics of each conversation. The standardized duration of delay is calculated using talk-turn delays from the respective session. Lastly, as for the actions family, templates 24 – 28 were added if any of the actions are detected during the talk-turn. For template 29, it was only added if the AU is detected throughout the entire duration of the talk-turn.

5 Experimental settings

There are two main types of inputs – (1) numeric inputs at the session-level, and (2) *coarse* and/or *fine*

Family	Child	ID	Template
Verbatim	Transcript	19	Transcript returned from the ASR system
Prosody	Speech rate	20	the doctor **quickly** said
	Tone	21	the doctor said **angrily**
	Delay	22	after **two** hundred milliseconds
		23	a **long** delay
Actions	Laughter	24	the doctor **laughed**
	Nodding	25	the doctor **nodded**
	Forward trunk learning	26	the doctor **leaned forward**
	Smiling	27	the doctor **smiled**
	PosiFace	28	the doctor **displayed positive facial expression**
	AU05, 17, 20, 25	29	the doctor **exhibited lip depressor**

Table 3: Templates for the talkturn-level *fine* summary.

multimodal narrative text inputs. As an overview, for (1), we trained the decision tree classifier using session-level numeric inputs. As for (2), we trained the HAN (Yang et al., 2016). We aim to facilitate how humans analyze conversations – HAN can work with text and has easy interpretation with single-headed attention, making it a suitable candidate. Relative to BERT (Devlin et al., 2018), the HAN is faster to train and easier to interpret.

5.1 Research questions

The proposed features have been motivated by scientific studies in Section 4. A natural next question is, "what are the impacts of these proposed features on model performance?" We break this broad question into three questions.

Firstly, (**Q1**) do the newly added features improve performance over the existing set of features for the classification tree and/or HAN?

Secondly, modelling using unstructured text input data (as opposed to using numeric inputs) has the risk of introducing too much variability in the inputs. Therefore, we investigate (**Q2**) – given the *coarse*-only inputs, do the performance between the HAN and classification tree differ significantly?

Lastly, adding more granular talkturn-level inputs to the *coarse* session-level inputs has the benefit of deeper analyses, because it allows the analyst to analyze important talkturns of the conversation. On top of this benefit, (**Q3**) do we also have significant performance improvement between *coarse*-only vs. both *coarse* and *fine* inputs?

For all models, the area under the receiver-operator curve (AUC) was used as the evaluation metric. The AUC measures the goodness of ranking (Hanley and McNeil, 1982) and therefore does not require an arbitrary threshold to turn the probabilities into classes. The partitioning of the dataset to the five-folds is constant for decision tree and HAN to facilitate comparison. The five folds are created through stratified sampling of the dependent variable.

5.2 Classification tree set-up

To answer (**Q1**) and (**Q2**), we tested for all 72 configurations of prime ($2^3 = 8$) plus full ($2^6 = 64$) family inputs for the decision tree. We performed the same z-transformation pre-processing (as in section 4.3) on the decision tree input variables and limited random search to twenty trials.

The algorithm used is from the *rpart* package with R. As part of hyperparameter tuning, we tuned the cp (log-uniform between 10^{-7} to 10^{-9}), maximum depth (uniform between 1 to 20), and minimum split (uniform between 20 to 80) through five-fold cross-validation and random search.

5.3 HAN set-up

To answer (**Q1**) and (**Q2**), we chose the input configurations that performed that best for the classification tree, and used the same input configurations in HAN to compare the difference. Therefore, this test is biased in favour of the classification tree. To answer (**Q3**), we added the *fine* narratives to each *coarse*-only configuration, and compared the difference.

The model architecture is the HAN architecture by Yang et al. (2016), with about 5 million parameters. We used the pre-trained Glove word embeddings (Pennington et al., 2014) of 300-dimensions to represent each word. Words not found in the Glove vocabulary are replaced with the "unk" to-

Coarse Inputs	**Tree**	**Coarse-only (HAN)**	**Significance of Difference (Coarse-only vs. Tree)**	**Coarse + Fine (HAN)**	**Significance of Difference (Coarse + Fine vs. Coarse-only)**
$D'A'P'$ (Existing Features)	0.577 (0.011)	0.637 (0.018)	^^^ [0.038, 0.082]	0.629 (0.041)	[-0.054, 0.038]
H	0.613 ** (0.036)	0.642 (0.038)	[-0.025, 0.083]	0.652 (0.048)	[-0.053, 0.073]
DH	0.670 *** (0.049)	0.670 ** (0.034)	[-0.062, 0.062]	0.654 (0.030)	[-0.063, 0.031]
PAH	0.684 *** (0.022)	0.645 (0.043)	[-0.089, 0.011]	0.661 (0.029)	[-0.038, 0.070]
$APMH$	0.664 *** (0.037)	0.643 (0.036)	[-0.074, 0.032]	0.657 (0.037)	[-0.039, 0.067]
$APSMH$	0.649 *** (0.021)	0.644 (0.049)	[-0.060, 0.050]	0.653 (0.051)	[-0.064, 0.082]
$DAPSMH$	0.630 *** (0.032)	0.661 * (0.030)	[-0.014, 0.076]	0.650 (0.028)	[-0.053, 0.031]

Table 4: Summary of the model performances. We report the average five-fold cross-validation AUC and its standard deviation in brackets. Row-wise: We begin with the $D'A'P'$, which is the full existing feature set from Kim et al. (2019b), and progressively compare it against the new sets of features to answer **Q1**. Column-wise: We compare the difference in AUC between the classification tree and *coarse*-only HAN to answer **Q2**. We compare the difference in AUC between the *coarse*-only HAN and *coarse* + *fine* HAN to answer **Q3**. Asterisks (*) indicate significance relative to the $D'A'P'$ row. Carets (^) indicate significance relative to column-wise comparisons, we also provide the confidence intervals in square brackets [] for the difference in performance. The number of symbols indicate the level of statistical significance, e.g., ***: 0.01, **: 0.05, *: 0.10.

ken. The hyperparameter tuning procedure is reported in Appendix A, and the best hyperparameter configurations are reported in Appendix B. There are twenty hyperparameter search trials for each input configuration[3].

6 Experimental results

The results are summarized in Table 4. The key findings are: (**Q1**) with the extended inputs, we observed statistically significant improvements in both the HAN and tree over the existing full set of features (one-tailed *t*-test); (**Q2**) given the *coarse*-only inputs, the performances between the HAN and classification tree did not differ significantly (two-tailed *t*-test), therefore it is plausible that feature engineering into text features do not risk performance; (**Q3**) although adding the *fine* narratives allow deeper analyses by the analyst, it does not lead to significant differences over the *coarse*-only inputs (two-tailed *t*-test).

(**Q1**) When compared to the full set of existing features, the classification tree achieved statistically significant improvements (at $\alpha = 0.05$) in all six out of six *coarse* input families. For HAN, it achieved statistically significant improvements in one (at $\alpha = 0.05$) or two (at $\alpha = 0.10$) out of six *coarse* input families. This demonstrates the value of the newly introduced *coarse* features[4].

(**Q2**) Across the seven *coarse* input configurations, there are no significant differences in the performance from the classification tree when compared to the HAN in six out of seven input configurations. The only exception is in the baseline $D'A'P'$ configuration where the HAN is significantly better. However, the lack of statistically significant differences does not mean that the performances are the *same*. In line with Quertemont (2011) recommendation, we provided the confidence interval around the difference in performance for discussion. Of all confidence intervals that included zero in the fourth column of Table 4, the

---

[3] We conducted additional tuning experiments for the tree in Appendix C to observe potential improvements in performance.

[4] We performed additional tests in Appendix D to observe the impact of the additions to the *fine* narratives, and found small improvements (but statistically insignificant) in all three out of three input families (va, vp, vpa).

confidence intervals do not suggest that that the effect sizes are negligible (for example, less than 0.01). In summary, we cannot conclude that the performance of HAN differs significantly from tree nor are they the same.

(**Q3**) The addition of *fine* narratives to the *coarse* narrative did not result in significantly stronger (nor weaker) performance in any of the seven input configurations. We posit that this negative finding is due to the difficulty in prioritizing the back-propagation updates to the parts of the network interacting with the *coarse* features, where there is likely a high signal-to-noise ratio. Despite the negative finding, we think it is important to explore fine features' addition onto coarse features because it produces a complete transcript for the human to understand how the conversation proceeded.

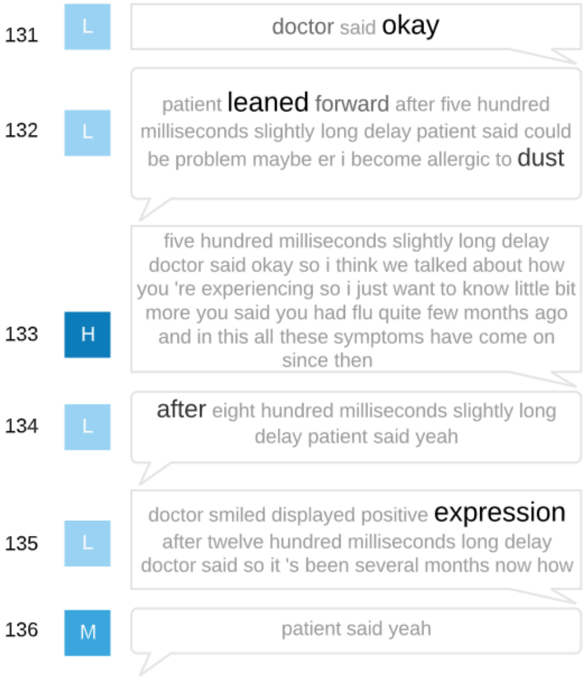
6.1 Qualitative Analysis

We visualized the talkturn-level and word-level attention weights from the model. Attention weights are normalized using z-transformation and bucketed into four buckets (< 0, < 1, < 2, ≥ 2) (Kim et al., 2019b). The analyst could analyze an important segment in detail (as in Fig. 3) or see an overview of the important segments in the conversation (see appendix E). In the example (Fig. 3), we observed that the multimodal annotations of leaning forward and positive expression were picked up as important words by the model.

7 Conclusion

In this paper, we build upon a fully text-based feature-engineering system. We motivated the added features with existing literature, and demonstrated the value of the added features through experiments on the EQClinic dataset. This approach emulates how humans have been analyzing conversations with the Jefferson (2004) transcription system, and hence is human-interpretable. It is highly modular, thereby allowing practitioners to inject modalities. In this paper, we have used a wide range of modalities, including *demographics*, *actions*, *prosody*, *mimicry*, *actions*, and *history*. The ablation tests showed that the added *coarse* features significantly improve the performance for both decision tree and HAN models.

Future research could (1) investigate whether this feature engineering system is generalizable to wider applications of conversational analysis; (2) conduct user studies to validate the usability and ease of interpretability of the visualization.



doctor said okay

patient leaned forward after five hundred milliseconds slightly long delay patient said could be problem maybe er i become allergic to dust

five hundred milliseconds slightly long delay doctor said okay so i think we talked about how you 're experiencing so i just want to know little bit more you said you had flu quite few months ago and in this all these symptoms have come on since then

after eight hundred milliseconds slightly long delay patient said yeah

doctor smiled displayed positive expression after twelve hundred milliseconds long delay doctor said so it 's been several months now how

patient said yeah

Figure 3: Conversation analysis for a true positive. The talkturn-level attentions are labelled Low (L), Medium (M) and High (H), while the words with higher attention have a larger and darker font. We also transcribed this segment using the Jefferson system in Appendix F.

Acknowledgments

We acknowledge the Sydney Informatics Hub and the University of Sydney's high-performance computing cluster, Artemis, for providing the computing resources and Marriane Makahiya for supporting the data manipulation work. Video data collection was carried out as part of the OSPIA platform project, funded by the Department of Health Clinical Training Fund from the Australian Government.

References

Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. 2016. Openface: an open source facial behavior analysis toolkit. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–10. IEEE.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995. Bracketing guidelines for Treebank II style Penn Treebank project. *University of Pennsylvania*, 97:100.

Joseph N Cappella. 1990. On defining conversational coordination and rapport. *Psychological Inquiry*, 1(4):303–305.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Patricio Costa, Raquel Alves, Isabel Neto, Pedro Marvao, Miguel Portela, and Manuel Joao Costa. 2014. Associations between medical student empathy and personality: a multi-institutional study. *PloS one*, 9(3).

Paul T Costa Jr, Antonio Terracciano, and Robert R McCrae. 2001. Gender differences in personality traits across cultures: robust and surprising findings. *Journal of personality and social psychology*, 81(2):322.

Gaëtan Cousin and Marianne Schmid Mast. 2013. Agreeable patient meets affiliative physician: how physician behavior affects patient outcomes depends on patient personality. *Patient Education and Counseling*, 90(3):399–404.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

M Robin DiMatteo. 1979. A social-psychological analysis of physician-patient rapport: toward a science of the art of medicine. *Journal of Social Issues*, 35(1):12–33.

Lawrence D Egbert, George E Battit, Claude E Welch, and Marshall K Bartlett. 1964. Reduction of postoperative pain by encouragement and instruction of patients: a study of doctor-patient rapport. *New England Journal of Medicine*, 270(16):825–827.

Florian Eyben, Martin Wöllmer, Michel F Valstar, Hatice Gunes, Björn Schuller, and Maja Pantic. 2011. String-based audiovisual fusion of behavioural events for the assessment of dimensional affect. In *Face and Gesture 2011*, pages 322–329. IEEE.

Barry A Farber and Erin M Doolin. 2011. Positive regard. *Psychotherapy*, 48(1):58.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

CJ Hutto Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media (ICWSM-14).*

Toni Giorgino and others. 2009. Computing and visualizing dynamic time warping alignments in R: the dtw package. *Journal of statistical Software*, 31(7):1–24.

Judith A Hall, Debra L Roter, Danielle C Blanch, and Richard M Frankel. 2009. Observer-rated rapport in interactions between medical students and standardized patients. *Patient Education and Counseling*, 76(3):323–327.

James A Hanley and Barbara J McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36.

Devamanyu Hazarika, Soujanya Poria, Rada Mihalcea, Erik Cambria, and Roger Zimmermann. 2018a. Icon: Interactive conversational memory network for multimodal emotion detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2594–2604.

Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018b. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2018, page 2122. NIH Public Access.

Colin Hesse and Emily A Rauscher. 2019. The relationships between doctor-patient affectionate communication and patient perceptions and outcomes. *Health communication*, 34(8):881–891.

Dirk Heylen, Elisabetta Bevacqua, Marion Tellier, and Catherine Pelachaud. 2007. Searching for prototypical facial feedback signals. In *International Workshop on Intelligent Virtual Agents*, pages 147–153. Springer.

Yuri Hosoda. 2006. Repair and relevance of differential language expertise in second language conversations. *Applied linguistics*, 27(1):25–50.

Gail Jefferson. 2004. Glossary of transcript symbols with an introduction. *Pragmatics and Beyond New Series*, 125:13–34.

Lee Jeong-Hwa and Kyung-Whan Cha. 2020. An analysis of the errors in the auto-generated captions of university commencement speeches on youtube. *Journal of Asia TEFL*, 17(1):143.

Joshua Kim, Rafael A Calvo, Kalina Yacef, and N J Enfield. 2019a. A Review on Dyadic Conversation Visualizations - Purposes, Data, Lens of Analysis. *arXiv preprint arXiv:1905.00653*.

Joshua Y Kim, Greyson Y Kim, and Kalina Yacef. 2019b. Detecting depression in dyadic conversations with multimodal narratives and visualizations. In *Australasian Joint Conference on Artificial Intelligence*, pages 303–314. Springer.

Joshua Y Kim, Chunfeng Liu, Rafael A Calvo, Kathryn McCabe, Silas CR Taylor, Björn W Schuller, and Kaihang Wu. 2019c. A comparison of online automatic speech recognition systems and the nonverbal responses to unintelligible speech. *arXiv preprint arXiv:1904.12403*.

Sander L Koole and Wolfgang Tschacher. 2016. Synchrony in psychotherapy: A review and an integrative framework for the therapeutic alliance. *Frontiers in psychology*, 7:862.

Suzanne M Kurtz and Jonathan D Silverman. 1996. The Calgary-Cambridge Referenced Observation Guides: an aid to defining the curriculum and organizing the teaching in communication training programmes. *Medical education*, 30(2):83–89.

Chunfeng Liu, Renee L Lim, Kathryn L McCabe, Silas Taylor, and Rafael A Calvo. 2016. A web-based telehealth training platform incorporating automated nonverbal behavior feedback for teaching communication skills to medical students: A randomized crossover study. *Journal of Medical Internet Research*, 18(9).

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6818–6825.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Valerie Lynn Manusov. 2014. *The sourcebook of nonverbal measures: Going beyond words*. Psychology Press.

Robert R McCrae. 2017. *The Five-Factor Model across cultures.* Praeger/ABC-CLIO.

Robert R McCrae and Paul T Costa. 1987. Validation of the five-factor model of personality across instruments and observers. *Journal of personality and social psychology*, 52(1):81.

Trisha Mittal, Uttaran Bhattacharya, Rohan Chandra, Aniket Bera, and Dinesh Manocha. 2020. M3er: Multiplicative multimodal emotion recognition using facial, textual, and speech cues. In *AAAI*, pages 1359–1367.

Frank Moers. 2005. Discretion and bias in performance evaluation: the impact of diversity and subjectivity. *Accounting, Organizations and Society*, 30(1):67–80.

Ryan R Montague. 2012. Genuine dialogue: Relational accounts of moments of meeting. *Western Journal of Communication*, 76(4):397–416.

Robert J Moore. 2015. Automated transcription and conversation analysis. *Research on Language and Social Interaction*, 48(3):253–270.

Philipp Müller, Michael Xuelin Huang, and Andreas Bulling. 2018. Detecting low rapport during natural interactions in small groups from non-Verbal behaviour. In *23rd International Conference on Intelligent User Interfaces*, pages 153–164. ACM.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Soujanya Poria, Erik Cambria, Rajiv Bajpai, and Amir Hussain. 2017. A review of affective computing: From unimodal analysis to multimodal fusion. *Information Fusion*, 37:98–125.

Paulo Quaglio. 2008. Television dialogue and natural conversation: Linguistic similarities and functional differences. *Corpora and discourse: The challenges of different settings*, pages 189–210.

Etienne Quertemont. 2011. How to statistically show the absence of an effect. *Psychologica Belgica*, 51(2):109–127.

Felicia Roberts and Alexander L Francis. 2013. Identifying a temporal threshold of tolerance for silent gaps after requests. *The Journal of the Acoustical Society of America*, 133(6):EL471–EL477.

Kimiko Ryokai, Elena Durán López, Noura Howell, Jon Gillick, and David Bamman. 2018. Capturing, Representing, and Interacting with Laughter. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 358. ACM.

Fabio Sala, Edward Krupat, and Debra Roter. 2002. Satisfaction and the use of humor by physicians and patients. *Psychology and Health*, 17(3):269–280.

Albert E Scheflen. 1964. The significance of posture in communication systems. *Psychiatry*, 27(4):316–331.

Taylan Sen, Mohammad Rafayet Ali, Mohammed Ehsan Hoque, Ronald Epstein, and Paul Duberstein. 2017. Modeling doctor-patient communication with affective text analysis. *2017 7th International Conference on Affective Computing and Intelligent Interaction, ACII 2017*, 2018-Janua:170–177.

Harold C Sexton, Kristin Hembre, and Guri Kvarme. 1996. The interaction of the alliance and therapy microprocess: A sequential analysis. *Journal of Consulting and Clinical Psychology*, 64(3):471.

Linda Tickle-Degnen and Robert Rosenthal. 1990. The nature of rapport and its nonverbal correlates. *Psychological inquiry*, 1(4):285–293.

John M Travaline, Robert Ruchinskas, and Gilbert E D'Alonzo Jr. 2005. Patient-physician communication: why and how. *Journal of the American Osteopathic Association*, 105(1):13.

Vokaturi. 2019. Vokaturi Overview.

Kaihang Wu, Chunfeng Liu, and Rafael A Calvo. 2020. Automatic Nonverbal Mimicry Detection and Analysis in Medical Video Consultations. *International Journal of Human–Computer Interaction*, pages 1–14.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Memory fusion network for multi-view sequential learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Ran Zhao, Tanmay Sinha, Alan W Black, and Justine Cassell. 2016. Socially-aware virtual agents: Automatically assessing dyadic rapport from temporal patterns of behavior. In *International conference on intelligent virtual agents*, pages 218–233. Springer.

Appendices

A Tuning procedure

We tuned the SGD optimizer with a learning rate between 0.003 to 0.010, batch size to be between 4 to 20, L2 regularization between 10^{-6} and 10^{-3}, and trained for up to 350 epochs without early stopping. We tuned the number of gated recurrent units (GRU) (Cho et al., 2014) between 40 to 49 in both the word-level and talk-turn-level layers, with both the GRU dropout and recurrent dropout (Gal and Ghahramani, 2016) to be between 0.05 to 0.50. The method of choosing hyperparameters is through uniform sampling between the above-mentioned bounds, except for the learning rate where log-uniform sampling is used. Training is performed on a RTX2070 GPU or V100 GPU.

B Hyperparameter configurations for best-performing models

Table 5 (HAN) and Table 6 (Tree) report the hyperparameter configurations for each of the best-performing model reported in Table 4.

C Performance of additional tuning

We conducted additional experiments on the tree configurations to (1) compare the improvements in performance when tuning the HAN and tree, and (2) evaluated the increase in performance if the tree is allowed twenty more hyperparameters random search trials (Fig. 4).

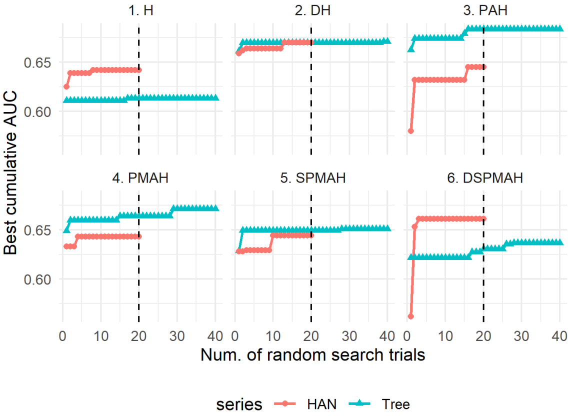


Figure 4: Best cumulative AUC performance given N random search trials.

From the larger increases in HAN performances, it is plausible that HAN is more sensitive to the hyperparameter tuning than the tree.

D Additional tests for additions to the *fine* narratives

Table 7 reports the additional tests on the impact of the added *fine* features. We observe that whilst all three input configurations (va, vp, vpa) have small increases in performance, none of them are statistically significant.

E Conversation thumbnail visualization

By illustrating the talkturn-level attention weights as a heatmap thumbnail (Fig. 5), the analyst could quickly get a sense of the important segments of the conversation without reading the content and zoom-in if required.

F Jefferson example

As an optional reference, we engaged a professional transcriptionist to transcribe the conversation segment presented (Fig. 3) using the Jefferson system. The Jefferson example is presented in Fig. 6. The verbal content is slightly different due to (1) different methods to determine talkturns transitions and (2) automatic speech recognition accuracy.

Config.	**Batch Size**	**Num. of GRU**	**Learning Rate**	**GRU dropout**	**GRU recurrent dropout**	**L2 regularization**	**Epoch**
H	19	42	0.010	0.10	0.23	1×10^{-4}	223
DH	11	46	0.010	0.07	0.09	3×10^{-6}	74
PAH	14	47	0.005	0.16	0.50	2×10^{-5}	329
$APMH$	8	44	0.005	0.29	0.16	1×10^{-3}	275
$APSMH$	9	43	0.005	0.16	0.48	4×10^{-5}	305
$DAPSMH$	14	41	0.010	0.49	0.48	2×10^{-5}	138
$D'A'P'$	19	46	0.004	0.06	0.50	1×10^{-4}	260
v	16	40	0.009	0.15	0.09	2×10^{-5}	316
va	13	43	0.007	0.13	0.48	1×10^{-6}	347
vp	8	42	0.006	0.13	0.05	2×10^{-5}	310
vpa	9	48	0.010	0.45	0.46	1×10^{-5}	349
va'	12	40	0.006	0.11	0.30	1×10^{-4}	346
vp'	11	42	0.007	0.44	0.19	2×10^{-5}	341
$vp'a'$	10	45	0.008	0.31	0.41	4×10^{-6}	267
H-vpa	8	42	0.005	0.38	0.33	2×10^{-5}	346
DH-vpa	12	44	0.009	0.25	0.14	1×10^{-5}	316
PAH-vpa	11	47	0.005	0.08	0.49	5×10^{-5}	349
$APMH$-vpa	18	46	0.008	0.13	0.50	1×10^{-5}	339
$APSMH$-vpa	9	43	0.010	0.13	0.21	2×10^{-6}	240
$DAPSMH$-vpa	15	46	0.009	0.15	0.50	2×10^{-5}	340
$D'A'P'$ - $vp'a'$	13	46	0.008	0.26	0.16	1×10^{-5}	262

Table 5: Best HAN configurations for the development set.

Config.	**Min. split**	**Max. depth**	**cp**
H	27	17	3.13×10^{-6}
DH	72	18	1.14×10^{-6}
PAH	70	15	8.84×10^{-5}
$APMH$	72	18	1.14×10^{-6}
$APSMH$	68	14	5.26×10^{-5}
$DAPSMH$	37	10	2.94×10^{-5}
$D'A'P'$	68	21	3.74×10^{-5}

Table 6: Best Tree configurations for the development set.

Config.	Existing inputs	New inputs	Significance of Difference (existing vs. new)
v	0.617 (0.053)		N/A
vp'	0.630 (0.037)	0.636 (0.055)	[-0.062, 0.074]
va'	0.616 (0.055)	0.622 (0.033)	[-0.060, 0.072]
$vp'a'$	0.630 (0.038)	0.648 (0.027)	[-0.030, 0.066]

Table 7: Summary of the model performances for the fine narratives. We report the average five-fold cross-validation AUC and its standard deviation in brackets. Row-wise, we begin with the v configuration to show the impact of *fine* multi-modal annotations over the verbatim transcript. Then, we show the impact of the additions (**Q1**) over the existing fine annotations from Kim et al. (2019b) using column-wise comparisons. Asterisks (*) indicate significance relative to the v row. Carets (^) indicate significance relative to column-wise comparisons, we also provide the confidence intervals in square brackets [] for the difference in performance. The number of symbols indicate the level of statistical significance, e.g., ***: 0.01, **: 0.05, *: 0.10.

Figure 5: Heatmap thumbnail. Darker blue indicates higher talkturn attention weights.

DOCTOR:	Ok::ay (0.2) and so you would say that i-those
	things are all worse when you're exerting yourself?
	(0.5)
PATIENT:	.hhh ↑↑ye::h↑↑ <I think so::> >I think so< .h but
	also you know maybe because I'm doing household
	chores and ↑stuff that could be: (0.2) a problem
	maybe I-I've become allergic to ↑↑↑dust?
	(0.5)
DOCTOR:	Oka:y (0.5) so:: (0.5) I ↑think we've talked about
	how you're experiencing >so I just ↑wanna know a
	little bit more.=You said you've had the (0.2) the
	flu: quite few months ago↑: .h and of this- all
	these symptoms have come on since the↑::n.
	(0.2)
PATIENT:	[Yeah.]
DOCTOR:	[Um::] (0.7) so it's been several months no:w.
	[0.5]
PATIENT:	[Yeah_]
DOCTOR:	[How] long did you have the flu: (.) period for.

Figure 6: Jefferson transcription example. : (colon) - stretched sound; (0.2) - a pause of 0.2 seconds; .hhh - in breath, .h - short in breath; ↑ - Rise in intonation; underline - emphasis; <> - slowed speech rate, >< - quickened speech rate; [] - overlapping speech.

Does Typological Blinding Impede Cross-Lingual Sharing?

Johannes Bjerva
Department of Computer Science
Aalborg University
jbjerva@cs.aau.dk

Isabelle Augenstein
Department of Computer Science
University of Copenhagen
augenstein@di.ku.dk

Abstract

Bridging the performance gap between high- and low-resource languages has been the focus of much previous work. Typological features from databases such as the World Atlas of Language Structures (WALS) are a prime candidate for this, as such data exists even for very low-resource languages. However, previous work has only found minor benefits from using typological information. Our hypothesis is that a model trained in a cross-lingual setting will pick up on typological cues from the input data, thus overshadowing the utility of explicitly using such features. We verify this hypothesis by blinding a model to typological information, and investigate how cross-lingual sharing and performance is impacted. Our model is based on a cross-lingual architecture in which the latent weights governing the sharing between languages is learnt during training. We show that (i) preventing this model from exploiting typology severely reduces performance, while a control experiment reaffirms that (ii) encouraging sharing according to typology somewhat improves performance.

1 Introduction

Most languages in the world have little access to NLP technology due to data scarcity (Joshi et al., 2020). Nonetheless, high-quality multilingual representations can be obtained using only a raw text signal, e.g. via multilingual language modelling (Devlin et al., 2019). Furthermore, structural similarities of languages are to a large extent documented in typological databases such as the World Atlas of Language Structures (WALS, Dryer and Haspelmath (2013)). Hence, developing models which can take use typological similarities of languages is an important direction in order to alleviate language technology inequalities.

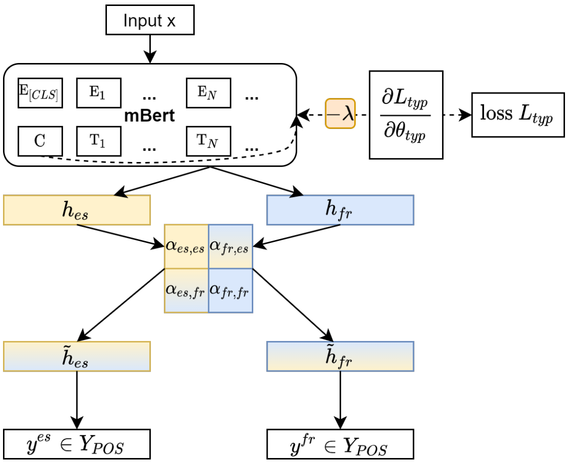


Figure 1: A PoS tagger is exposed (or blinded with gradient reversal, $-\lambda$) to typological features. Observing α values tells us how typology affects sharing.

While previous work has attempted to use typological information to inform NLP models, our work differs significantly from such efforts in that we *blind* a model to this information. Most previous work includes language information as features, by using language IDs, or language embeddings (e.g. Ammar et al. (2016); O'Horan et al. (2016); Östling and Tiedemann (2017); Ponti et al. (2019); Oncevay et al. (2020)). Notably, limited effects are usually observed from including typological features explicitly. For instance, de Lhoneux et al. (2018) observe positive cross-lingual sharing effects only in a handful of their settings. We therefore hypothesise that relevant typological information is learned as a by-product of cross-lingual training. Hence, although models do benefit from this information, it is not necessary to provide it explicitly in a high-resource scenario, where there is abundant training data. This is confirmed by Bjerva and Augenstein (2018a), who find that, e.g., language embeddings trained on a morphological task can encode morphological features from WALS.

In contrast with previous work, we *blind* a model to typological information, by using adversarial

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 480–486
April 19 - 23, 2021.

techniques based on gradient reversal (Ganin and Lempitsky, 2014). We evaluate on the structured prediction and classification tasks in XTREME (Hu et al., 2020), yielding a total of 40 languages and 4 tasks. We show that when a model is blinded to typological signals relating to syntax and morphology, performance on related NLP tasks drops significantly. For instance, the mean accuracy across 40 languages for POS tagging drops by 1.8% when blinding the model to morphological features.

2 Model

An overview of the model is shown in Figure 1. We model each task in this paper using the following steps. First, contextual representations are extracted using multilingual BERT (m-BERT, Devlin et al. (2019)), a transformer-based model (Vaswani et al., 2017), trained with shared wordpieces across languages. We either blind m-BERT to typological features, with an added adversarial component based on gradient reversal (Ganin and Lempitsky, 2014), or expose it to them via multitask learning (MTL, (Caruana, 1997)). Representations from m-BERT are fed to a latent multi-task architecture learning network (Ruder et al., 2019), which includes α parameters we seek to investigate. The model learns which parameters to share between languages (e.g. $\alpha_{es,fr}$ denotes sharing between Spanish and French).

2.1 Sharing architecture

Our sharing architecture is based on that of Ruder et al. (2019), which has latent variables learned during training, governing which layers and subspaces are shared between tasks, to what extent, as well as the relative weighting of different task losses. We are most interested in the parameters which control the sharing between the hidden layers allocated to each task, referred to as α parameters (Ruder et al., 2019). Consider a setting with two tasks A and B. The outputs $h_{A,k}$ and $h_{B,k}$ of the k-th layer for task A and B interact through the α parameters, for which the output is defined as:

$$\begin{bmatrix} \widetilde{h}_{A,k} \\ \widetilde{h}_{B,k} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \left[{h_{A,k}}^{\top}, \quad {h_{B,k}}^{\top} \right] \quad (1)$$

where $\widetilde{h}_{A,k}$ is a linear combination of the activations for task A at layer k, weighted with the learned αs. While their model is an MTL model, we choose to interpret this differently by considering each *language* as a task, yielding $\alpha \in \mathbf{R}^{l \times l}$, where l is the number of languages for the given task. Each activation $\widetilde{h}_{A,k}$ is then a linear combination of the language specific activations $h_{A,k}$. These are used for prediction in the downstream tasks, as in the baselines from Hu et al. (2020).

Crucially, this model allows us to draw conclusions about parameter sharing between languages by observing the α parameters under the blinding and prediction conditions. We will combine this insight with observing downstream task performance in order to draw conclusions about the effects of typological feature blinding and prediction.

2.2 Blinding/Exposing a Model to Typology

We introduce a component which can either blind or expose the model to typological features. We implement this as a single task-specific layer per feature, using the `[CLS]` token from m-BERT model, without access to any of the soft sharing between languages from α-layers. Each layer optimises a categorical cross-entropy loss function (L_{typ}).

For this task, we predict typological features drawn from WALS (Dryer and Haspelmath, 2013), inspired by previous work (Bjerva and Augenstein, 2018a). Unlike previous work, we also blind the model to such features by including a gradient reversal layer (Ganin and Lempitsky, 2014), which multiplies the gradient of the typological prediction task with a negative constant ($-\lambda$), inspired by previous work on adversarial learning (Goodfellow et al., 2014; Zhang et al., 2019; Chen et al., 2019). We hypothesise that using a gradient reversal layer for typology will yield typology-invariant features, and that this will perform worse on tasks for which the typological feature at hand is important. For instance, we expect that blinding a model to syntactic features will severely reduce performance for tasks which rely heavily on syntax, such as POS tagging.

3 Cross-Lingual Experiments

We investigate the effects of typological blinding, using typological parameters as presented in WALS (Dryer and Haspelmath, 2013). The experiments are run on XTREME (Hu et al., 2020), which includes up to 40 languages from 12 language families and two isolates. We experiment on the following languages (ISO 639-1 codes): af, ar, bg, bn, de, el, en, es, et, eu, fa, fi, fr, he, hi, hu, id, it, ja, jv, ka, kk, ko, ml, mr, ms, my, nl, pt, ru, sw, ta, te, th, tl, tr, ur, vi, yo, and zh. We experiment on four tasks: POS (part of speech tagging), NER (named entity recognition), XNLI (cross-lingual natural language inference), and PAWS-X (paraphrase identification). Our general setup for the structured prediction tasks (POS and NER) is that we train on all available languages, and downsample to 1,000 samples per language. For the classification tasks XNLI and PAWS-X, we train on the English training data and fine-tune on the development sets, as no training data is available for other languages. Hence, typological differences will be the main factor in our results, rather than differences in dataset sizes.

3.1 Typological Prediction and Blinding

We first investigate whether prohibiting or allowing access to typological features has an effect on model performance using our architecture. We hypothesise that our multilingual model will leverage signals related to the linguistic nature of a task when optimising its its sharing parameters α.

There exists a growing body of work on prediction of typological features (Daumé III and Campbell, 2007; Murawaki, 2017; Bjerva and Augenstein, 2018b; Bjerva et al., 2019a,b), most notably in a recent shared task on the subject (Bjerva et al., 2020). While we are inspired by this direction of research, our contribution is not concerned with the accuracy of the prediction of such features, and this is therefore not evaluated in detail in the paper.

Moreover, an increasing amount of work measures the correlation of predictive performance of cross-lingual models with typological features as a way of probing what a model has learned about typology (Malaviya et al., 2017; Choenni and Shutova, 2020; Gerz et al., 2018; Nooralahzadeh et al., 2020; Zhao et al., 2020). In contrast to such post-hoc approaches, our experimental setting allows for measuring the impact of typology on cross-lingual sharing performance in a direct manner as part of the model architecture.

Syntactic Features We first blind/expose the model to syntactic features from WALS (Dryer and Haspelmath, 2013). We take the set of word order features which are annotated for all languages in our experiments, resulting in 33 features. This includes features such as *81A: Order of Subject, Object and Verb*, which encodes what the preferred word ordering is (if any) in a transitive clause. For all features, we exclude feature values which do not occur for our set of languages. We hypothesise that performance will drop for all four tasks, as they all require syntactic understanding.

Morphological Features We next attempt to blind/expose the model to the morphological features in WALS. We use the same approach as above, resulting in a total of 8 morphological features. This includes features such as *26A: Prefixing vs. Suffixing in Inflectional Morphology*, indicating to what extent a language uses prefixing or suffixing morphology. We hypothesise that mainly the POS tagging task will suffer under this condition, whereas other tasks only to some extent require morphology.

Phonological Features We next consider a control experiment, in which we attempt to blind/expose the model to phonological features in WALS. We arrive at a total of 15 phonological features, such as *1A: Consonant Inventories* which indicates the size of the consonant inventory of a language. We expect the performance to remain relatively unaffected by this task, as phonology ought to have little importance given a textual input.

Genealogical Features Finally, we attempt to use what one might consider to be language metadata. We attempt to blind/expose the model to what language family a language belongs to. This can be seen as a type of proxy to language similarity, and correlates relatively strongly with structural similarities in languages. Because of this correlation with structural similarities, we expect blinding under this condition to only slightly reduce performance for all tasks, as previous work has shown this type of relationship not to be central in language representations (Bjerva et al., 2019c).

3.2 Results

In general, we observe a drop in performance when blinding the model to relevant typological information, and an increase in performance when exposing the model to it (Table 1). For phonological blinding or prediction, none of the four tasks is noticeably affected. Although, e.g., both the syntactic and morphological prediction tasks increase performance on POS tagging, it is not straightforward to draw conclusions on which of these is the most efficient, as there is a substantial correlation between syntactic and morphological features. As for XNLI and PAWS-X, performance notably drops under both the syntactic and genealogical blinding tasks.

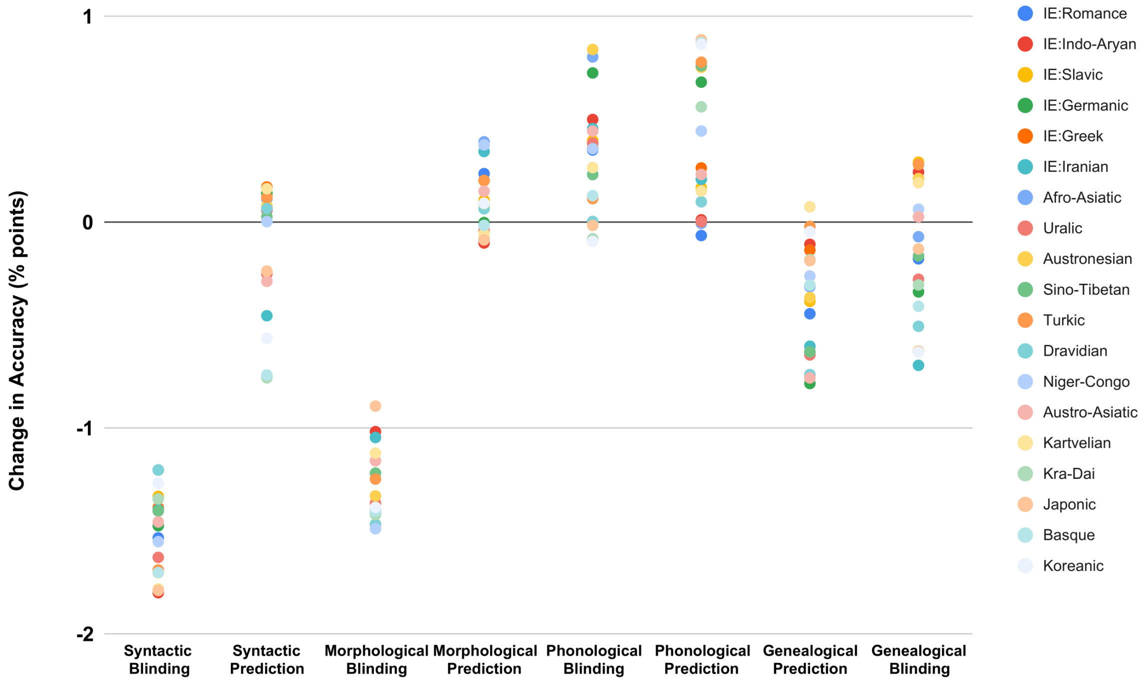


Figure 2: PoS tagging results per language family across blinding and prediction conditions

Model	POS	NER	XNLI	PAWS-X
+ Syntactic Blind.	85.3^{-}	76.4	64.2^{-}	80.6^{-}
+ Morphological Blind.	85.0^{-}	77.2	64.9	81.4
+ Phonological Blind.	86.7	77.1	65.0	81.6
+ Genealogical Blind.	86.1	77.0	64.7	81.1
m-BERT baseline	86.8	77.3	65.1	81.7
+ Syntactic Pred.	87.0	77.5	**65.3**$^{+}$	**81.9**$^{+}$
+ Morphological Pred.	**87.2**$^{+}$	77.3	65.2	81.7
+ Phonological Pred.	86.7	77.1	65.0	81.7
+ Genealogical Pred.	87.0	**77.6**	**65.3**$^{+}$	81.8

Table 1: Typological Blinding and Prediction. Mean POS accuracy, NER F1 scores, XNLI accuracy and PAWS-X accuracy across all languages. $^{+}$ and $^{-}$ indicate significantly better or worse performance respectively, as determined by a one-tailed t-test ($p < 0.01$).

Figure 2 shows results for PoS tagging under prediction and blinding across language families, following the same scheme as Hu et al. (2020). Interestingly, the syntactic and morphological blinding settings are robust across all language families, yielding a drop in accuracy across the board. All other conditions yield mixed results. This further strengthens our argument that preventing a model from learning syntactic and morphological features can be severely detrimental.

3.3 The Effect of Typology on Latent Architecture Learning

The results show that preventing access to typological features hampers performance, whereas providing access improves performance. We now turn to an analysis of how the model shares parameters across languages in this setting. Our hypothesis is that blinding will prevent models from sharing parameters between similar languages, in spite of typological similarities. Concretely, we expect that the drop in POS tagging performance under morphological blinding is caused by lower α weights between languages which are morphologically similar, and higher α weights between languages which are dissimilar. Recall that these parameters are latent variables learned by the model, regulating the amount of sharing between languages (see Eq. 1). We investigate the correlations between the α sharing parameters, and two proxies of language similarity. We focus on the POS task, as the results from the typological blinding and prediction experiments were the most pronounced here, as both morphological and syntactic blinding affected performance.

Our first measure of language similarity is based on Bjerva et al. (2019c), who introduce what they refer to as structural similarity. This is based on

Model	**Struct.**	**Lang. Emb.**
Syntactic Blind.	0.31	0.27
Morphological Blind.	0.34	0.29
Phonological Blind.	0.40	0.41
Genealogical Blind.	0.29	0.31
No blind./pred.	0.43	0.40
Syntactic Pred.	**0.52**	0.53
Morphological Pred.	0.49	**0.56**
Phonological Pred.	0.41	0.39
Genealogical Pred.	0.47	0.38

Table 2: Pearson correlations between α weights and language similarity measures.

dependency statistics from the Universal Dependencies treebank (Zeman et al., 2020), resulting in vectors which describe how different syntactic relations are used in each language. Previous work has shown that this measure of similarity correlates strongly with that learned in embedded language spaces during multilingual training. In addition to considering these dependency statistics, we also use language embeddings drawn form Östling and Tiedemann (2017). For each language similarity measure we calculate its pairwise Pearson correlation with the α values learned under each condition.

Table 2 shows correlations between α weights and similarities increase when predicting typological features, and decreases when blinded to such features. Hence, when the model has indirect access to, e.g., the SVO word ordering features of languages, sharing also reflects this.

4 Discussion

We have shown that blinding a multilingual model to typological features severely affects sharing across a relatively large language sample, and for several NLP tasks. The effects on model performance, as evaluated over 40 languages and 4 tasks from XTREME (Hu et al., 2020), were the largest for POS tagging. The fact that smaller effects were observed for NER, could be because this task relies more on memorising NEs rather than using (morpho-)syntactic cues (Augenstein et al., 2017). Furthermore, the relatively small effects on XNLI and PAWS-X can also be interpreted as evidence for that typology is less important in these tasks than in more traditional linguistic analysis.

A potential critique of our approach is that it merely blinds the model to language identities. This could be the case, if only some latent representation of, e.g., “SVO” ordering is used to represent a language identity. However, previous work has shown that morphological information is encoded by the type of model we investigate. Hence, since we only blind features in a single category at a time, we expect that the model’s representation of language identities is unaffected.

Not only do we observe a drop in performance when blinding a model to syntactic features, but we also observe that the α sharing weights in our model do not appear to correlate with linguistic similarities in this setting. Conversely, encouraging a model to consider typology, by jointly optimising it for typological feature prediction, improves performance in general. Furthermore, α weights in this scenario converge towards correlating with structural similarities of languages. This is in line with recent work which has found that m-BERT uses fine-grained syntactic distinctions in its cross-lingual representation space (Chi et al., 2020).

We interpret this as evidence for the fact that typology can be a necessity for modelling in NLP. Our results furthermore corroborate previous work in that we only find moderate benefits from including typological information explicitly. We expect that this to a large degree is due to the typological similarities of languages being encoded implicitly based on correlations between patterns in the input data. As low-resource languages often do not even have access to any substantial amount of raw text, but often do have annotations in WALS, we expect that using typological information can go some way towards building truly language-universal models.

5 Conclusions

We have shown that preventing access to typology can impede the performance of cross-lingual sharing models. Investigating latent weights governing the sharing between languages shows that this prevents the model from sharing between typologically similar languages, which is otherwise learned based on patterns in the input. We therefore expect that using typological information can be of particular interest for building truly language-universal models for low-resource languages.

Acknowledgements

This research has received funding from the Swedish Research Council (grant No 2019-04129), and the NVIDIA Corporation (Titan Xp GPU).

References

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Johannes Bjerva and Isabelle Augenstein. 2018a. From phonology to syntax: Unsupervised linguistic typology at different levels with language embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 907–916, New Orleans, Louisiana. Association for Computational Linguistics.

Johannes Bjerva and Isabelle Augenstein. 2018b. Tracking Typological Traits of Uralic Languages in Distributed Language Representations. In *Proceedings of the Fourth International Workshop on Computational Linguistics of Uralic Languages*, pages 76–86, Helsinki, Finland. Association for Computational Linguistics.

Johannes Bjerva, Yova Kementchedjhieva, Ryan Cotterell, and Isabelle Augenstein. 2019a. A probabilistic generative model of linguistic typology. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1529–1540, Minneapolis, Minnesota. Association for Computational Linguistics.

Johannes Bjerva, Yova Kementchedjhieva, Ryan Cotterell, and Isabelle Augenstein. 2019b. Uncovering probabilistic implications in typological knowledge bases. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3924–3930, Florence, Italy. Association for Computational Linguistics.

Johannes Bjerva, Elizabeth Salesky, Sabrina J. Mielke, Aditi Chaudhary, Celano Giuseppe, Edoardo Maria Ponti, Ekaterina Vylomova, Ryan Cotterell, and Isabelle Augenstein. 2020. SIGTYP 2020 shared task: Prediction of typological features. In *Proceedings of the Second Workshop on Computational Research in Linguistic Typology*, pages 1–11, Online. Association for Computational Linguistics.

Johannes Bjerva, Robert Östling, Maria Han Veiga, Jörg Tiedemann, and Isabelle Augenstein. 2019c. What Do Language Representations Really Represent? *Computational Linguistics*, 45(2):381–389.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28 (1):41–75.

Steven Chen, Nicholas Carlini, and David Wagner. 2019. Stateful detection of black-box adversarial attacks. *arXiv preprint arXiv:1907.05587*.

Ethan A Chi, John Hewitt, and Christopher D Manning. 2020. Finding universal grammatical relations in multilingual bert. *arXiv preprint arXiv:2005.04511*.

Rochelle Choenni and Ekaterina Shutova. 2020. What does it mean to be language-agnostic? probing multilingual sentence encoders for typological properties. *CoRR*, abs/2009.12862.

Hal Daumé III and Lyle Campbell. 2007. A Bayesian Model for Discovering Typological Implications. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

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

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Yaroslav Ganin and Victor Lempitsky. 2014. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*.

Daniela Gerz, Ivan Vulić, Edoardo Maria Ponti, Roi Reichart, and Anna Korhonen. 2018. On the relation between linguistic typology and (limitations of) multilingual language modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 316–327, Brussels, Belgium. Association for Computational Linguistics.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *arXiv preprint arXiv:2003.11080*.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the nlp world. *arXiv preprint arXiv:2004.09095*.

Miryam de Lhoneux, Johannes Bjerva, Isabelle Augenstein, and Anders Søgaard. 2018. Parameter sharing between dependency parsers for related languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4992–4997.

Chaitanya Malaviya, Graham Neubig, and Patrick Littell. 2017. Learning language representations for typology prediction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2529–2535, Copenhagen, Denmark. Association for Computational Linguistics.

Yugo Murawaki. 2017. Diachrony-aware induction of binary latent representations from typological features. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 451–461, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-Shot Cross-Lingual Transfer with Meta Learning. In *Proceedings of EMNLP*. Association for Computational Linguistics.

Helen O'Horan, Yevgeni Berzak, Ivan Vulić, Roi Reichart, and Anna Korhonen. 2016. Survey on the use of typological information in natural language processing. *arXiv preprint arXiv:1610.03349*.

Arturo Oncevay, Barry Haddow, and Alexandra Birch. 2020. Bridging linguistic typology and multilingual machine translation with multi-view language representations. In *Proceedings of EMNLP*. Association for Computational Linguistics. ArXiv preprint arXiv:2004.14923.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 644–649, Valencia, Spain. Association for Computational Linguistics.

Edoardo Maria Ponti, Helen O'horan, Yevgeni Berzak, Ivan Vulić, Roi Reichart, Thierry Poibeau, Ekaterina Shutova, and Anna Korhonen. 2019. Modeling language variation and universals: A survey on typological linguistics for natural language processing. *Computational Linguistics*, 45(3):559–601.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019. Latent multi-task architecture learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4822–4829.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Daniel Zeman, Joakim Nivre, and Mitchell Abrams et al. 2020. Universal dependencies 2.6. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Huan Zhang, Hongge Chen, Zhao Song, Duane Boning, Inderjit S Dhillon, and Cho-Jui Hsieh. 2019. The limitations of adversarial training and the blind-spot attack. *arXiv preprint arXiv:1901.04684*.

Wei Zhao, Steffen Eger, Johannes Bjerva, and Isabelle Augenstein. 2020. Inducing Language-Agnostic Multilingual Representations. *arXiv preprint arXiv:2008.09112*.

AdapterFusion:
Non-Destructive Task Composition for Transfer Learning

Jonas Pfeiffer[1], **Aishwarya Kamath**[2], **Andreas Rücklé**[1],
Kyunghyun Cho[2,3], **Iryna Gurevych**[1]
[1]Ubiquitous Knowledge Processing Lab (UKP Lab), Technical University of Darmstadt
[2]New York University [3]CIFAR Associate Fellow
`pfeiffer@ukp.tu-darmstadt.de`

Abstract

Sequential fine-tuning and multi-task learning are methods aiming to incorporate knowledge from multiple tasks; however, they suffer from catastrophic forgetting and difficulties in dataset balancing. To address these shortcomings, we propose *AdapterFusion*, a new two stage learning algorithm that leverages knowledge from multiple tasks. First, in the *knowledge extraction* stage we learn task specific parameters called *adapters*, that encapsulate the task-specific information. We then combine the adapters in a separate *knowledge composition* step. We show that by separating the two stages, i.e., knowledge extraction and knowledge composition, the classifier can effectively exploit the representations learned from multiple tasks in a non-destructive manner. We empirically evaluate AdapterFusion on 16 diverse NLU tasks, and find that it effectively combines various types of knowledge at different layers of the model. We show that our approach outperforms traditional strategies such as full fine-tuning as well as multi-task learning. Our code and adapters are available at AdapterHub.ml.

1 Introduction

The most commonly used method for solving NLU tasks is to leverage pretrained models, with the dominant architecture being a transformer (Vaswani et al., 2017), typically trained with a language modelling objective (Devlin et al., 2019; Radford et al., 2018; Liu et al., 2019b). Transfer to a task of interest is achieved by fine-tuning all the weights of the pretrained model on that *single task*, often yielding state-of-the-art results (Zhang and Yang, 2017; Ruder, 2017; Howard and Ruder, 2018; Peters et al., 2019). However, each task of interest requires all the parameters of the network to be fine-tuned, which results in a specialized model for each task.

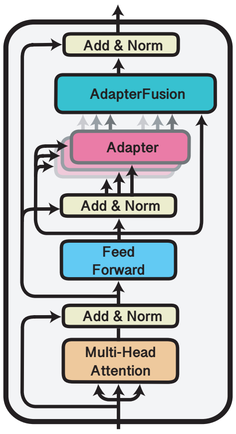


Figure 1: AdapterFusion architecture inside a transformer (Vaswani et al., 2017). The AdapterFusion component takes as input the representations of multiple adapters trained on different tasks and learns a parameterized mixer of the encoded information.

There are two approaches for sharing information across multiple tasks. The first consists of starting from the pretrained language model and sequentially fine-tuning on each of the tasks one by one (Phang et al., 2018). However, as we subsequently fine-tune the model weights on new tasks, the problem of catastrophic forgetting (McCloskey and Cohen, 1989; French, 1999) can arise, which results in loss of knowledge already learned from all previous tasks. This, together with the non-trivial decision of the order of tasks in which to fine-tune the model, hinders the effective transfer of knowledge. Multi-task learning (Caruana, 1997; Zhang and Yang, 2017; Liu et al., 2019a) is another approach for sharing information across multiple tasks. This involves fine-tuning the weights of a pretrained language model using a weighted sum of the objective function of each target task simultaneously. Using this approach, the network captures the common structure underlying all the target tasks. However, multi-task learning requires simul-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 487–503
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

taneous access to all tasks during training. Adding new tasks thus requires complete joint retraining. Further, it is difficult to balance multiple tasks and train a model that solves each task equally well. As has been shown in Lee et al. (2017), these models often overfit on low resource tasks and underfit on high resource tasks. This makes it difficult to effectively transfer knowledge across tasks with all the tasks being solved equally well (Pfeiffer et al., 2020b), thus considerably limiting the applicability of multi-task learning in many scenarios.

Recently, *adapters* (Rebuffi et al., 2017; Houlsby et al., 2019) have emerged as an alternative training strategy. Adapters do not require fine-tuning of all parameters of the pretrained model, and instead introduce a small number of task specific parameters — while keeping the underlying pretrained language model fixed. Thus, we can separately and simultaneously train adapters for multiple tasks, which all share the same underlying pretrained parameters. However, to date, there exists no method for using *multiple* adapters to maximize the transfer of knowledge across tasks without suffering from the same problems as sequential fine-tuning and multi-task learning. For instance, Stickland and Murray (2019) propose a multi-task approach for training adapters, which still suffers from the difficulty of balancing the various target tasks and requiring simultaneous access to all target tasks.

In this paper we address these limitations and propose a new variant of adapters called *AdapterFusion*. We further propose a novel two stage learning algorithm that allows us to effectively share knowledge across multiple tasks while avoiding the issues of catastrophic forgetting and balancing of different tasks. Our AdapterFusion architecture, illustrated in Figure 1, has two components. The first component is an adapter trained on a task without changing the weights of the underlying language model. The second component — our novel Fusion layer — combines the representations from several such task adapters in order to improve the performance on the target task.

Contributions Our main contributions are: (1) We introduce a novel two-stage transfer learning strategy, termed *AdapterFusion*, which combines the knowledge from multiple source tasks to perform better on a target task. (2) We empirically evaluate our proposed approach on a set of 16 diverse NLU tasks such as sentiment analysis, commonsense reasoning, paraphrase detection, and recognizing textual entailment. (3) We compare our approach with Stickland and Murray (2019) where adapters are trained for all tasks in a multi-task manner, finding that AdapterFusion is able to improve this method, even though the model has simultaneous access to all tasks during pretraining. (4) We show that our proposed approach outperforms fully fine-tuning the transformer model on a single target task. Our approach additionally outperforms adapter based models trained both in a Single-Task, as well as Multi-Task setup.

The code of this work is integrated into the AdapterHub.ml (Pfeiffer et al., 2020a).

2 Background

In this section, we formalize our goal of transfer learning (Pan and Yang, 2010; Torrey and Shavlik, 2010; Ruder, 2019), highlight its key challenges, and provide a brief overview of common methods that can be used to address them. This is followed by an introduction to *adapters* (Rebuffi et al., 2017) and a brief formalism of the two approaches to training adapters.

Task Definition. We are given a model that is pretrained on a task with training data D_0 and a loss function L_0. The weights Θ_0 of this model are learned as follows:

$$\begin{aligned} D_0 &:= \text{Large corpus of unlabelled text} \\ L_0 &:= \text{Masked language modelling loss} \\ \Theta_0 &\leftarrow \underset{\Theta}{\operatorname{argmin}}\; L_0(D_0; \Theta) \end{aligned}$$

In the remainder of this paper, we refer to this pretrained model by the tuple (D_0, L_0).

We define C as the set of N classification tasks having labelled data of varying sizes and different loss functions:

$$C = \{(D_1, L_1), \ldots, (D_N, L_N)\}$$

The aim is to be able to leverage a set of N tasks to improve on a target task m with $C_m = (D_m, L_m)$. In this work we focus on the setting where $m \in \{1, \ldots, N\}$.

Desiderata. We wish to learn a parameterization Θ_m that is defined as follows:

$$\Theta_m \leftarrow \underset{\Theta'}{\operatorname{argmin}}\; L_m(D_m; \Theta')$$

where Θ' is expected to have encapsulated relevant information from all the N tasks. The target model for task m is initialized with Θ' for which we learn the optimal parameters Θ_m through minimizing the task's loss on its training data.

2.1 Current Approaches to Transfer Learning

There are two predominant approaches to achieve sharing of information from one task to another.

2.1.1 Sequential Fine-Tuning

This involves sequentially updating all the weights of the model on each task. For a set of N tasks, the order of fine-tuning is defined and at each step the model is initialized with the parameters learned through the previous step. However, this approach does not perform well beyond two sequential tasks (Phang et al., 2018; Pruksachatkun et al., 2020) due to catastrophic forgetting.

2.1.2 Multi-Task Learning (MTL)

All tasks are trained simultaneously with the aim of learning a shared representation that will enable the model to generalize better on each task (Caruana, 1997; Collobert and Weston, 2008; Nam et al., 2014; Liu et al., 2016, 2017; Zhang and Yang, 2017; Ruder, 2017; Ruder et al., 2019; Sanh et al., 2019; Pfeiffer et al., 2020b, *inter alia*).

$$\Theta_{0\rightarrow\{1,...,N\}} \leftarrow \underset{\Theta}{\operatorname{argmin}} \left(\sum_{n=1}^{N} L_n(D_n; \Theta_0) \right)$$

Where $\Theta_{0\rightarrow\{1,...,N\}}$ indicates that we start with Θ_0 and fine-tune on a set of tasks $\{1, ..., N\}$.

However, MTL requires simultaneous access to all tasks, making it difficult to add more tasks on the fly. As the different tasks have varying sizes as well as loss functions, effectively combining them during training is very challenging and requires heuristic approaches as proposed in Stickland and Murray (2019).

2.2 Adapters

While the predominant methodology for transfer learning is to fine-tune all weights of the pretrained model, *adapters* (Houlsby et al., 2019) have recently been introduced as an alternative approach with applications in domain transfer (Rücklé et al., 2020b), machine translation (Bapna and Firat, 2019; Philip et al., 2020) transfer learning (Stickland and Murray, 2019; Wang et al., 2020; Lauscher et al., 2020), and cross-lingual transfer (Pfeiffer et al., 2020c,d; Üstün et al., 2020; Vidoni et al., 2020). Adapters share a large set of parameters Θ across all tasks and introduce a small number of task-specific parameters Φ_n. While Θ represents the weights of a pretrained model (e.g., a transformer), the parameters Φ_n, where $n \in \{1, \ldots, N\}$, are used to encode task-specific representations in intermediate layers of the shared model. Current work on adapters focuses either on training adapters for each task separately (Houlsby et al., 2019; Bapna and Firat, 2019; Pfeiffer et al., 2020a) or training them in a multi-task setting to leverage shared representations (Stickland and Murray, 2019). We discuss both variants below.

2.2.1 Single-Task Adapters (ST-A)

For each of the N tasks, the model is initialized with parameters Θ_0. In addition, a set of new and randomly initialized adapter parameters Φ_n are introduced.

The parameters Θ_0 are fixed and only the parameters Φ_n are trained. This makes it possible to efficiently parallelize the training of adapters for all N tasks, and store the corresponding knowledge in designated parts of the model. The objective for each task $n \in \{1, \ldots, N\}$ is of the form:

$$\Phi_n \leftarrow \underset{\Phi}{\operatorname{argmin}}\ L_n(D_n; \Theta_0, \Phi)$$

For common adapter architectures, Φ contains considerably fewer parameters than Θ, e.g., only 3.6% of the parameters of the pretrained model in Houlsby et al. (2019).

2.2.2 Multi-Task Adapters (MT-A)

Stickland and Murray (2019) propose to train adapters for N tasks in parallel with a multi-task objective. The underlying parameters Θ_0 are fine-tuned along with the task-specific parameters in Φ_n. The training objective can be defined as:

$$\boldsymbol{\Theta} \leftarrow \underset{\Theta, \Phi}{\operatorname{argmin}} \left(\sum_{n=1}^{N} L_n(D_n; \Theta_0, \Phi_n) \right)$$

where

$$\boldsymbol{\Theta} = \Theta_{0\rightarrow\{1,...,N\}}, \Phi_1, \ldots, \Phi_N.$$

2.2.3 Adapters in Practice

Introducing new adapter parameters in different layers of an otherwise fixed pretrained model has been shown to perform on-par with, or only slightly below, full model fine-tuning (Houlsby et al., 2019; Stickland and Murray, 2019; Pfeiffer et al., 2020a).

For NLP tasks, adapters have been introduced for the transformer architecture (Vaswani et al., 2017). At each transformer layer l, a set of adapter parameters Φ_l is introduced. The placement and architecture of adapter parameters Φ within a pretrained model is non-trivial. Houlsby et al. (2019) experiment with different architectures, finding that a two-layer feed-foward neural network with a bottleneck works well. They place two of these components within one layer, one after the multi-head attention (further referred to as *bottom*) and one after the feed-forward layers of the transformer (further referred to as *top*).[1] Bapna and Firat (2019) and Stickland and Murray (2019) only introduce one of these components at the *top* position, however, Bapna and Firat (2019) include an additional *layer norm* (Ba et al., 2016).

Adapters trained in both single-task (ST-A) or multi-task (MT-A) setups have learned the idiosyncratic knowledge of the respective tasks' training data, encapsulated in their designated parameters. This results in a compression of information, which requires less space to store task-specific knowledge. However, the distinct weights of adapters prevent a downstream task from being able to use multiple sources of extracted information. In the next section we describe our two stage algorithm which tackles the sharing of information stored in adapters trained on different tasks.

3 AdapterFusion

Adapters avoid catastrophic forgetting by introducing task-specific parameters; however, current adapter approaches do not allow sharing of information between tasks. To mitigate this we propose AdapterFusion.

3.1 Learning algorithm

In the first stage of our learning algorithm, we train either ST-A or MT-A for each of the N tasks.

In the second stage, we then combine the set of N adapters by using AdapterFusion. While fixing both the parameters Θ as well as all adapters Φ, we introduce parameters Ψ that learn to combine the N task adapters to solve the target task.

$$\Psi_m \leftarrow \underset{\Psi}{\operatorname{argmin}}\ L_m(D_m; \Theta, \Phi_1, \ldots, \Phi_N, \Psi)$$

Ψ_m are the newly learned AdapterFusion parameters for task m. Θ refers to Θ_0 in the ST-A setting or $\Theta_{0\rightarrow\{1,\ldots,N,m\}}$ in the MT-A setup. In our experiments we focus on the setting where $m \in \{1, \ldots, N\}$, which means that the training dataset of m is used twice: *once* for training the adapters Φ_m and *again* for training Fusion parameters Ψ_m, which learn to compose the information stored in the N task adapters.

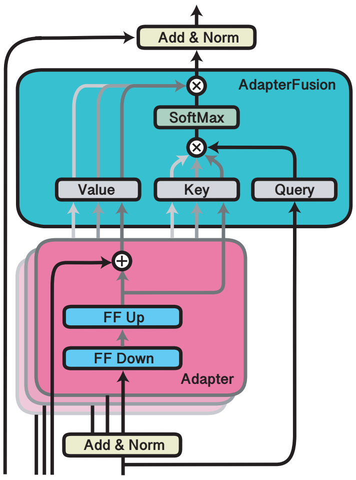


Figure 2: Our AdapterFusion architecture. This includes learnable weights *Query*, *Key*, and *Value*. *Query* takes as input the output of the pretrained transformer weights. Both *Key* and *Value* take as input the output of the respective adapters. The dot product of the *query* with all the *keys* is passed into a softmax function, which learns to weight the adapters with respect to the context.

By separating the two stages — knowledge extraction in the adapters, and knowledge composition with AdapterFusion — we address the issues of catastrophic forgetting, interference between tasks and training instabilities.

3.2 Components

AdapterFusion learns to compose the N task adapters Φ_n and the shared pretrained model Θ, by introducing a new set of weights Ψ. These parameters learn to combine the adapters as a dynamic function of the target task data.

As illustrated in Figure 2, we define the AdapterFusion parameters Ψ to consist of *Key, Value* and *Query* matrices at each layer l, denoted by $\mathbf{K}_l$, $\mathbf{V}_l$ and $\mathbf{Q}_l$ respectively. At *each layer* l of the transformer and *each time-step* t, the output of the feed-forward sub-layer of layer l is taken as the query vector. The output of each adapter $\mathbf{z}_{l,t}$ is used as input to both the *value* and *key* transformations. Similar to attention (Bahdanau et al., 2015; Vaswani et al., 2017), we learn a contextual activation of

---

[1] We illustrate these placements in Appendix Figure 5 (left).

each adapter n using

$$\begin{aligned}
\mathbf{s}_{l,t} &= \text{softmax}(\mathbf{h}_{l,t}^{\top}\mathbf{Q}_l \otimes \mathbf{z}_{l,t,n}^{\top}\mathbf{K}_l), n \in \{1, ..., N\} \\
\mathbf{z}'_{l,t,n} &= \mathbf{z}_{l,t,n}^{\top}\mathbf{V}_l, n \in \{1, ..., N\} \\
\mathbf{Z}'_{l,t} &= [\mathbf{z}'_{l,t,0}, ..., \mathbf{z}'_{l,t,N}] \\
\mathbf{o}_{l,t} &= \mathbf{s}_{l,t}^{\top}\mathbf{Z}'_{l,t}
\end{aligned}$$

Where $\otimes$ represents the dot product and $[\cdot, \cdot]$ indicates the concatenation of vectors.

Given the context, AdapterFusion learns a parameterized mixer of the available trained adapters. It learns to identify and activate the most useful adapter for a given input.

4 Experiments

In this section we evaluate how effective AdapterFusion is in overcoming the issues faced by other transfer learning methods. We provide a brief description of the 16 diverse datasets that we use for our study, each of which uses accuracy as the scoring metric.

4.1 Experimental Setup

In order to investigate our model's ability to overcome catastrophic forgetting, we compare Fusion using ST-A to only the ST-A for the task. We also compare Fusion using ST-A to MT-A for the task to test whether our two-stage procedure alleviates the problems of interference between tasks. Finally, our experiments to compare MT-A with and without Fusion let us investigate the versatility of our approach. Gains in this setting would show that AdapterFusion is useful even when the base adapters have already been trained jointly.

In all experiments, we use BERT-base-uncased (Devlin et al., 2019) as the pretrained language model. We train ST-A, described in Appendix A.2 and illustrated in Figure 5, for all datasets described in §4.2. We train them with reduction factors[2] $\{2, 16, 64\}$ and learning rate 0.0001 with AdamW and a linear learning rate decay. We train for a maximum of 30 epochs with early stopping. We follow the setup used in Stickland and Murray (2019) for training the MT-A. We use the default hyperparameters[3], and train a MT-A model on all datasets simultaneously.

For AdapterFusion, we empirically find that a learning rate of $5e-5$ works well, and use this in all experiments.[4] We train for a maximum of 10 epochs with early stopping. While we initialize $\mathbf{Q}$ and $\mathbf{K}$ randomly, we initialize $\mathbf{V}$ with a diagonal of ones and the rest of the matrix with random weights having a small norm ($1e-6$). Multiplying the adapter output with this value matrix $\mathbf{V}$ initially adds small amounts of noise, but retains the overall representation. We continue to regularize the *Value* matrix using l_2-norm to avoid introducing additional capacity.

4.2 Tasks and Datasets

We briefly summarize the different types of *tasks* that we include in our experiments, and reference the related datasets accordingly. A detailed descriptions can be found in Appendix A.1.

Commonsense reasoning is used to gauge whether the model can perform basic reasoning skills: *Hellaswag* (Zellers et al., 2018, 2019), *Winogrande* (Sakaguchi et al., 2020), *CosmosQA* (Huang et al., 2019), *CSQA* (Talmor et al., 2019), *SocialIQA* (Sap et al., 2019). **Sentiment analysis** predicts whether a given text has a positive or negative sentiment: *IMDb* (Maas et al., 2011), *SST* (Socher et al., 2013). **Natural language inference** predicts whether one sentence entails, contradicts, or is neutral to another: *MNLI* (Williams et al., 2018), *SciTail* (Khot et al., 2018), *SICK* (Marelli et al., 2014), *RTE* (as combined by Wang et al. (2018)), *CB* (De Marneffe et al., 2019). **Sentence relatedness** captures whether two sentences include similar content: *MRPC* (Dolan and Brockett, 2005), *QQP*[5]. We also use an argument mining *Argument* (Stab et al., 2018) and reading comprehension *BoolQ* (Clark et al., 2019) dataset.

5 Results

We present results for all 16 datasets in Table 1. For reference, we also include the adapter architecture of Houlsby et al. (2019), ST-A$^{\text{Houlsby}}$, which has twice as many parameters compared to ST-A. To provide a fair comparison to Stickland and Murray (2019) we primarily experiment with BERT-base-uncased. We additionally validate our best model configurations — ST-A and Fusion with ST-A — with RoBERTa-base, for which we present our results in Appendix Table 4.

---

[2] A reduction factor indicates the factor by which the hidden size is reduced such that the bottle-neck size for BERT Base with factor 64 is reduced to 12 ($768/64 = 12$).

[3] We additionally test out batch sizes 16 and 32.

[4] We have experimented with learning rates $\{6e-6, 5e-5, 1e-4, 2e-4\}$

[5] data.quora.com/First-Quora-DatasetReleaseQuestion-Pairs

Dataset	**Head**	**Full**	**ST-A**	**MT-A**	**F. w/ ST-A**	**F. w/ MT-A**	**ST-A$^{\text{Houlsby}}$**
MNLI	54.59	84.10	**84.32**	82.49 ±0.49	84.28	83.05	84.13
QQP	76.79	**90.87**	90.59	89.47 ±0.60	90.71	90.58	90.63
SST	85.17 ±0.45	92.39 ±0.22	91.85 ±0.41	92.27 ±0.71	92.20 ±0.18	**93.00** ±0.20	92.75 ±0.37
WGrande	51.92 ±0.35	60.01 ±0.08	**61.09** ±0.11	57.70 ±1.40	60.23 ±0.31	59.32 ±0.30	59.32 ±1.33
IMDB	85.05 ±0.22	**94.05** ±0.21	93.85 ±0.07	92.56 ±0.54	93.82 ±0.39	92.66 ±0.32	93.96 ±0.22
HSwag	34.17 ±0.27	**39.25** ±0.76	38.11 ±0.14	36.47 ±0.98	37.98 ±0.01	37.36 ±0.10	38.65 ±0.25
SocIQA	50.33 ±2.50	62.05 ±0.04	62.41 ±0.11	61.21 ±0.89	**63.16** ±0.24	62.56 ±0.10	62.73 ±0.53
CosQA	50.06 ±0.51	60.28 ±0.40	60.01 ±0.02	61.25 ±0.90	60.65 ±0.55	**62.78** ±0.07	61.37 ±0.35
SciTail	85.30 ±2.44	94.32 ±0.11	93.90 ±0.16	94.53 ±0.43	94.04 ±0.23	**94.79** ±0.17	94.07 ±0.39
Argument	70.61 ±0.59	76.87 ±0.32	**77.65** ±0.34	75.70 ±0.60	**77.65** ±0.21	76.08 ±0.27	77.44 ±0.62
CSQA	41.09 ±0.27	58.88 ±0.40	58.91 ±0.57	53.30 ±2.19	59.73 ±0.54	56.73 ±0.14	**60.05** ±0.36
BoolQ	63.07 ±1.27	74.84 ±0.24	75.66 ±1.25	78.76 ±0.76	76.25 ±0.19	**79.18** ±0.45	76.02 ±1.13
MRPC	71.91 ±0.13	85.14 ±0.45	85.16 ±0.52	81.86 ±0.99	**90.29** ±0.84	84.68 ±0.32	86.66 ±0.81
SICK	76.30 ±0.71	87.30 ±0.42	86.20 ±0.00	88.61 ±1.06	87.28 ±0.99	**90.43** ±0.30	86.12 ±0.54
RTE	61.37 ±1.17	65.41 ±0.90	71.04 ±1.62	77.61 ±3.21	76.82 ±1.68	**79.96** ±0.76	69.67 ±1.96
CB	68.93 ±4.82	82.49 ±2.33	86.07 ±3.87	89.09 ±1.15	**92.14** ±0.97	89.81 ±0.99	87.50 ±4.72
Mean	64.17	75.51	76.05	75.80	**77.33**	77.06	76.32

Table 1: Mean and standard deviation results (development sets) for each of the 16 datasets and the different architectural setups. The datasets are ordered by their respective training dataset size. Dashed horizontal lines separate datasizes $\{> 40k, > 10k, > 5k\}$, respectively. Each model is initialized with BERT-base (Devlin et al., 2019) weights. **Head** indicates training only a classification head on top of fixed BERT weights. For **Full** training we fine-tune all weights of BERT. Single-Task Adapters (**ST-A**) is the training of independently trained adapters for each task, using the architecture illustrated in Figure 5. Multi-Task Adapters (**MT-A**) shows results of jointly trained adapters using the default settings of Stickland and Murray (2019). **Fusion w/ ST-A** and **Fusion w/ MT-A** show the results of AdapterFusion using the respective pre-trained Adapters. **ST-A$^{\text{Houlsby}}$** shows the results of ST-Adapters with the architecture proposed by Houlsby et al. (2019). Reported results are accuracy scores.

5.1 Adapters

Training only a prediction-head on the output of a pretrained model can also be considered an adapter. This procedure, commonly referred to as training only the *Head*, performs considerably worse than fine-tuning all weights (Howard and Ruder, 2018; Peters et al., 2019). We show that the performance of only fine-tuning the *Head* compared to *Full* fine-tuning causes on average a drop of 10 points in accuracy. This demonstrates the need for more complex adaptation approaches.

In Table 1 we show the results for MT-A and ST-A with a reduction factor 16 (see the appendix Table 3 for more results) which we find has a good trade-off between the number of newly introduced parameters and the task performance. Interestingly, the ST-A have a regularization effect on some datasets, resulting in better performance on average for certain tasks, even though a much small proportion of weights is trained. On average, we improve 0.66% by training ST-A instead of the *Full* model.

For MT-A we find that there are considerable performance drops of more than 2% for *CSQA* and *MRPC*, despite the heuristic strategies for sampling from the different datasets (Stickland and Murray, 2019). This indicates that these heuristics only partially address common problems of multi-task learning such as catastrophic interference. It also shows that learning a shared representation jointly does not guarantee the best results for all tasks. On average, however, we do see a performance increase of 0.4% using MT-A over *Full* fine-tuning on each task separately, which demonstrates that there are advantages in leveraging information from other tasks with multi-task learning.

5.2 AdapterFusion

AdapterFusion aims to improve performance on a given target task m by transferring task specific knowledge from the set of all N task adapters, where $m \in \{1, \ldots, \mathrm{N}\}$. We hypothesize that if there exists at least one task that supports the target task, AdapterFusion should lead to performance gains. If no such task exists, then the performance should remain the same.

Dependence on the size of training data. In Table 1 we notice that having access to relevant tasks considerably improves the performance for the target task when using AdapterFusion. While datasets with more than 40k training instances perform well without Fusion, smaller datasets with fewer training instances benefit more from our approach. We

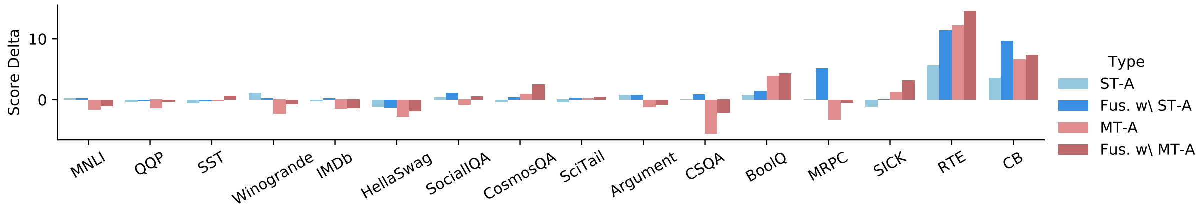


Figure 3: Relative performance difference of the two adapter architectures and the AdapterFusion models over fully fine-tuned BERT. Fusion improves over its corresponding adapters (ST-A and MT-A) for most tasks.

	Fus. w/ ST-A		**Fus. w/ MT-A**	
compared to	ST-A	MT-A	ST-A	MT-A
MNLI	→	↗	↘	↗
QQP	→	↗	→	↗
SST	↗	→	↗	↗
Winogrande	↘	↗	↘	↗
IMDB	↗	↗	↘	→
HellaSwag	→	↗	↘	↗
SocialIQA	↗	↗	→	↗
CosmosQA	↗	↘	↗	↗
SciTail	→	↗	↗	→
Argument	→	↗	↘	↗
CSQA	↗	↗	↘	↗
BoolQ	↗	↘	↗	↗
MRPC	↗	↗	↘	↗
SICK	↗	↘	↗	↗
RTE	↗	↘	↗	↗
CB	↗	↗	↗	↗
Improved	10/16	11/16	7/16	14/16

Table 2: Performance changes of AdapterFusion compared to ST-A and MT-A. Arrows indicate whether there has been an improvement ↗ (> 0.3), decrease ↘ (< -0.3), or whether the results have stayed the same → $[-0.3, 0.3]$.

observe particularly large performance gains for datasets with less than 5k training instances. For example, Fusion with ST-A achieves substantial improvements of 6.5 % for *RTE* and 5.64 % for *MRPC*. In addition, we also see performance gains for moderately sized datasets such as the commonsense tasks *CosmosQA* and *CSQA*. Fusion with MT-A achieves smaller improvements, as the model already includes a shared set of parameters. However, we do see performance gains for *SICK*, *SocialIQA*, *Winogrande* and *MRPC*. On average, we observe improvements of 1.27% and 1.25% when using Fusion with ST-A and MT-A, respectively.

Mitigating catastrophic interference. In order to identify whether our approach is able to mitigate problems faced by multi-task learning, we present the performance differences of adapters and AdapterFusion compared to the fully fine-tuned model in Figure 3. In Table 2, we compare AdapterFusion to ST-A and MT-A. The arrows indicate whether there is an improvement ↗, decrease ↘, or if the the results remain the same →. We compare the performance of both, Fusion with ST-A and Fusion with MT-A, to ST-A and MT-A. We summarize our four most important findings below.

(1) In the case of Fusion with ST-A, for $15/16$ tasks, the performance remains the same or improves as compared to the task's pretrained adapter. For $10/16$ tasks we see performance gains. This shows that having access to adapters from other tasks is beneficial and in the majority of cases leads to better results on the target task. **(2)** We find that for $11/16$ tasks, Fusion with ST-A improves the performance compared to MT-A. This demonstrates the ability of Fusion with ST-A to share information between tasks while avoiding the interference that multi-task training suffers from. **(3)** For only $7/16$ tasks, we see an improvement of Fusion with MT-A over the ST-A. Training of MT-A in the first stage of our algorithm suffers from all the problems of multi-task learning and results in less effective adapters than our ST-A on average. Fusion helps bridge some of this gap but is not able to mitigate the entire performance drop. **(4)** In the case of AdapterFusion with MT-A, we see that the performances on *all 16 tasks* improves or stays the same. This demonstrates that AdapterFusion can successfully combine the specific adapter weights, even if the adapters were trained in a multi-task setting, confirming that our method is versatile.

Summary. Our findings demonstrate that Fusion with ST-A is the most promising approach to sharing information across tasks. Our approach allows us to train adapters in parallel and it requires no heuristic sampling strategies to deal with imbalanced datasets. It also allows researchers to easily add more tasks as they become available, without requiring complete model retraining.

While Fusion with MT-A does provide gains over simply using MT-A, the effort required to train

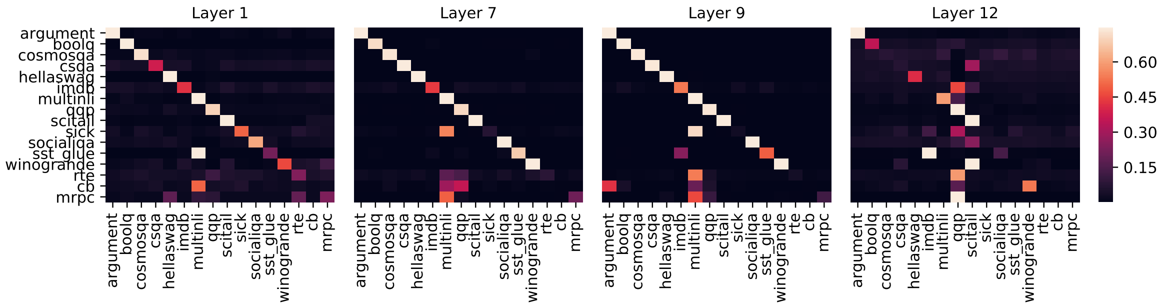


Figure 4: AdapterFusion activations of pretrained **ST-Adapters**. Rows indicate the target task m, columns indicate adapters n. We assume that the softmax activation for $\Phi_{n,l}$ is high if the information of adapter n is useful for task m. For our analysis, we calculate the softmax activation for each adapter $\Phi_{n,l}$, where $n \in \{1, \ldots, N\}$, and average over all activations within the same layer l calculated over all instances in the development set.

these in a multi-task setting followed by the Fusion step are not warranted by the limited gains in performance. On the other hand, we find that Fusion with ST-A is an efficient and versatile approach to transfer learning.

6 Analysis of Fusion Activation

We analyze the weighting patterns that are learned by AdapterFusion to better understand which tasks impact the model predictions, and whether there exist differences across BERT layers.

We plot the results for layers 1, 7, 9, and 12 and ST-A in Figure 4 (see Appendix Figure 6 for the remaining layers). We find that tasks which do not benefit from AdapterFusion tend to more strongly activate their own adapter at every layer (e.g. *Argument, HellaSwag, MNLI, QQP, SciTail*). This confirms that AdapterFusion only extracts information from adapters if they are beneficial for the target task m. We further find that *MNLI* is a useful intermediate task that benefits a large number of target tasks, e.g. *BoolQ, SICK, CSQA, SST-2, CB, MRPC, RTE*, which is in line with previous work (Phang et al., 2018; Conneau and Kiela, 2018; Reimers and Gurevych, 2019). Similarly, *QQP* is utilized by a large number of tasks, e.g. *SICK, IMDB, RTE, CB, MRPC, SST-2*. Most importantly, tasks with small datasets such as *CB, RTE,* and *MRPC* often strongly rely on adapters trained on large datasets such as *MNLI* and *QQP*.

Interestingly, we find that the activations in layer 12 are considerably more distributed across multiple tasks than adapters in earlier layers. The potential reason for this is that the last adapters are not encapsulated between frozen pretrained layers, and can thus be considered as an extension of the prediction head. The representations of the adapters in the 12th layer might thus not be as comparable, resulting in more distributed activations. This is in line with Pfeiffer et al. (2020d) who are able to improve zero-shot cross-lingual performance considerably by dropping the adapters in the last layer.

7 Contemporary Work

In contemporaneous work, other approaches for parameter efficient fine-tuning have been proposed. Guo et al. (2020) train sparse "diff" vectors which are applied on top of pretrained frozen parameter vectors. Ravfogel and Goldberg (2021) only finetune bias terms of the pretrained language models, achieving similar results as full model finetuning. Li and Liang (2021) propose prefix-tuning for natural language generation tasks. Here, continuous task-specific vectors are trained while the remaining model is kept frozen. These alternative, parameter-efficient fine-tuning strategies all encapsulate the idiosyncratic task-specific information in designated parameters, creating the potential for new composition approaches of multiple tasks.

Rücklé et al. (2020a) analyse the training and inference efficiency of adapters and AdapterFusion. For AdapterFusion, they find that adding more tasks to the set of adapters results in a linear increase of computational cost, both for training and inference. They further propose approaches to mitigate this overhead.

8 Conclusion and Outlook

8.1 Conclusion

We propose a novel approach to transfer learning called AdapterFusion which provides a simple and effective way to combine information from several tasks. By separating the extraction of knowledge from its composition, we are able to effectively avoid the common pitfalls of multi-task learning, such as catastrophic forgetting and interference between tasks. Further, AdapterFusion mitigates the problem of traditional multi-task learning in which complete re-training is required, when new tasks are added to the pool of datasets.

We have shown that AdapterFusion is compatible with adapters trained in both single-task as well as multi-task setups. AdapterFusion consistently outperforms fully fine-tuned models on the target task, demonstrating the value in having access to information from other tasks. While we observe gains using both ST-A as well as MT-A, we find that composing ST-A using AdapterFusion is the more efficient strategy, as adapters can be trained in parallel and re-used.

Finally, we analyze the weighting patterns of individual adapters in AdapterFusion which reveal that tasks with small datasets more often rely on information from tasks with large datasets, thereby achieving the largest performance gains in our experiments. We show that AdapterFusion is able to identify and select adapters that contain knowledge relevant to task of interest, while ignoring the remaining ones. This provides an implicit no-op option and makes AdapterFusion a suitable and versatile transfer learning approach for any NLU setting.

8.2 Outlook

Rücklé et al. (2020a) have studied pruning a large portion of adapters after Fusion training. Their results show that removing the less activated adapters results in almost no performance drop at inference time while considerably improving the inference speed. They also provide some initial evidence that it is possible to train Fusion with a subset of the available adapters in each minibatch, potentially enabling us to scale our approach to large adapter sets — which would otherwise be computationally infeasible. We believe that such extensions are a promising direction for future work.

Pfeiffer et al. (2020d) have achieved considerable improvements in the zero-shot cross-lingual transfer performance by dropping the adapters in the last layer. In preliminary results, we have observed similar trends with AdapterFusion when the adapters in the last layer are not used. We will investigate this further in future work.

Acknowledgments

Jonas is supported by the LOEWE initiative (Hesse, Germany) within the emergenCITY center. Aishwarya was supported in part by a DeepMind PhD Fellowship during the time which this project was carried out. Andreas is supported by the German Research Foundation within the project “Open Argument Mining” (GU 798/25-1), associated with the Priority Program “Robust Argumentation Machines (RATIO)” (SPP-1999). This work was partly supported by Samsung Advanced Institute of Technology (Next Generation Deep Learning: from pattern recognition to AI) and Samsung Research (Improving Deep Learning using Latent Structure). Kyunghyun was a research scientist at Facebook AI Research part-time during which this project was carried out.

We thank Sebastian Ruder, Max Glockner, Jason Phang, Alex Wang, Katrina Evtimova and Sam Bowman for insightful feedback and suggestions on drafts of this paper.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *arXiv preprint*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1538–1548.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, pages 160–167.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2020. Parameter-efficient transfer learning with diff pruning. *arXiv preprint*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzkebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 2790–2799.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 328–339.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2391–2401.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5189–5197.

Anne Lauscher, Olga Majewska, Leonardo F. R. Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavaš. 2020. Common Sense or World Knowledge? Investigating Adapter-Based Knowledge Injection into Pretrained Transformers. *arXiv preprint*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics 2017*, 5:365–378.

Hector J. Levesque. 2011. The winograd schema challenge. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4487–4496.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Jinseok Nam, Jungi Kim, Eneldo Loza Menc'ia, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification - revisiting neural networks. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part II*, pages 437–452.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019, Florence, Italy, August 2, 2019*, pages 7–14.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Edwin Simpson, and Iryna Gurevych. 2020b. Low resource multi-task sequence tagging - revisiting dynamic conditional random fields. *arXiv preprint*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020c. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020d. UNKs Everywhere: Adapting Multilingual Language Models to New Scripts. *arXiv preprint*.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint*.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4465–4470.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Elad Ben-Zaken1 Shauli Ravfogel and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint*.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 506–516.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3980–3990, Hong Kong, China. Association for Computational Linguistics.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020a. AdapterDrop: On the Efficiency of Adapters in Transformers. *arXiv preprint*.

Andreas Rücklé, Jonas Pfeiffer, and Iryna Gurevych. 2020b. MultiCQA: Zero-shot transfer of self-supervised text matching models on a massive scale. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2471–2486, Online. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint*.

Sebastian Ruder. 2019. *Neural Transfer Learning for Natural Language Processing*. Ph.D. thesis, National University of Ireland, Galway.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019. Latent multi-task architecture learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4822–4829.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6949–6956.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4462–4472.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451.

Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. 2018. Cross-topic argument mining from heterogeneous sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3664–3674.

Asa Cooper Stickland and Iain Murray. 2019. BERT and pals: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 5986–5995.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

M. Vidoni, Ivan Vulić, and Goran Glavaš. 2020. Orthogonal language and task adapters in zero-shot cross-lingual transfer. In *arXiv preprint*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 353–355.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 93–104.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800.

Yu Zhang and Qiang Yang. 2017. A survey on multitask learning. *arXiv preprint*.

A Appendices

A.1 Datasets

Commonsense Reasoning We work with a large number of datasets, all of which have emerged recently in this domain, ranging from sentence level and document level classification to multiple choice questions. The next sentence prediction task *HellaSWAG* (Zellers et al., 2019) is a more difficult version of the previously released *SWAG* dataset (Zellers et al., 2018). *Winogrande* (Sakaguchi et al., 2020) is a large scale and adversarially filtered (Zellers et al., 2018) adaptation of the *Winograd Schema Challenge* (Levesque, 2011). *Cosmos QA* (Huang et al., 2019) is a commonsense reading comprehension dataset which requires reasoning over larger text passages. *Social IQA* (Sap et al., 2019) is a multiple choice dataset which requires reasoning over social interactions between humans. *Commonsense QA* (Talmor et al., 2019) is a multiple choice dataset based on ConceptNet (Speer et al., 2017), which requires reasoning over general knowledge.

Sentiment Analysis We conduct experiments on two binary sentiment classification tasks on long and short text passages. *IMDb* (Maas et al., 2011) consists of long movie reviews and *SST-2* (Socher et al., 2013) consists of short movie reviews from Rotten Tomatoes[6].

Natural Language Inference (NLI) The goal is to classify whether two sentences entail, contradict, or are neutral to each other. For this we conduct experiments on *MultiNLI* (Williams et al., 2018), a multi-genre dataset, *SciTail* (Khot et al., 2018) a NLI dataset on scientific text, *SICK* (Marelli et al., 2014) a NLI dataset with relatedness scores, the composition of *Recognizing Textual Entailment (RTE)* datasets provided by Wang, Singh, Michael, Hill, Levy, and Bowman (2018), as well as the *Commitment Bank (CB)* (De Marneffe et al., 2019) three-class textual entailment dataset.

Sentence Relatedness We include two semantic relatedness datasets which capture whether or not two text samples include similar content. *Microsoft Research Paraphrase Corpus (MRPC)* (Dolan and Brockett, 2005) consists of sentence pairs which capture a paraphrase/semantic equivalence relationship. *Quora Question Pairs (QQP)* targets duplicate question detection.[7]

Misc The Argument Aspect corpus (Stab et al., 2018) is a three-way classification task to predict whether a document provides arguments *for*, *against* or *none* for a given topic (Nuclear Energy, Abortion, Gun-Control, etc). BoolQ (Clark et al., 2019) is a binary reading comprehension classification task for simple *yes, no* questions.

A.2 What Is The Best Adapter Setup?

As described in §2.2.3, the placement of adapter parameters Φ within a pretrained model is non-trivial, and thus requires extensive experiments. In order to identify the best ST-A setting, we run an exhaustive architecture search on the hyperparameters — including the position and number of adapters in each transformer layer, the position and number of pretrained or task dependent layer norms, the position of residual connections, the bottleneck reduction factors $\{2, 8, 16, 64\}$, and the non linearity $\{\text{ReLU, LeakyReLU, Swish}\}$ used within the adapter. We illustrate this in Figure 5. This grid search includes the settings introduced by Houlsby et al. (2019) and Bapna and Firat (2019). We perform this search on three diverse tasks[8] and find

---

[6] www.rottentomatoes.com

[7] data.quora.com/First-Quora-DatasetReleaseQuestion-Pairs

[8] SST-2, Commonsense QA, and Argument.

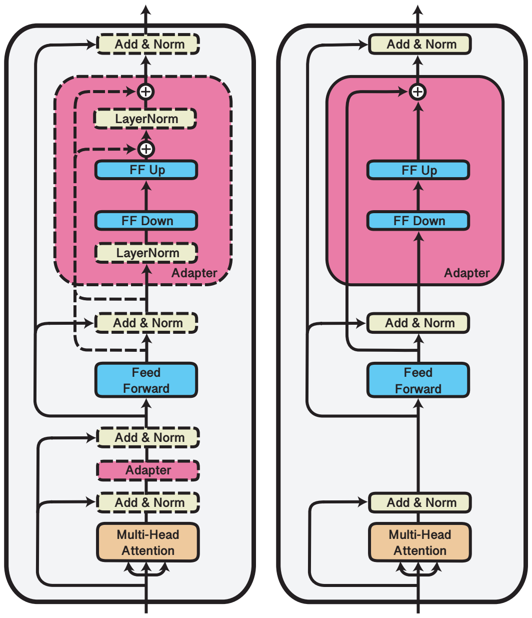


Figure 5: Different architectural components of the adapter. On the left, we show all components for which we conduct an exhaustive search (dashed lines). On the right, we show the adapter architecture that performs the best across all our tasks.

that across all three tasks, the same setup obtains best results. We present our results on the SST-2, Argument, and CSQA datasets in Figures 7, 8, and 9 respectively, at different granularity levels. We find that in contrast to Houlsby et al. (2019), but in line with Bapna and Firat (2019), a single adapter after the feed-forward layer outperforms other settings. While we find that this setting performs on-par with that of Houlsby et al. (2019), it requires only half the number of newly introduced adapters as compared to them, resulting in a more efficient setting in terms of number of operations.

For the single-task adapter setting, we thus perform all subsequent experiments with the best architecture illustrated in Figure 5 on the right and a learning rate of $1e-4$. In order to reproduce the multi-task results in Stickland and Murray (2019) and build upon them, for experiments involving multi-task training, we adopt their architecture as described in §2.2.3.

A.3 AdapterFusion Activations of all Layers

We present the cross-product of activations of AdapterFusion of all layers for BERT-Base and ST-A_{16} in Figure 6, as an extension to Figure 4.

A.4 BERT-base ST-A with Reduction Factors $\{2, 16, 64\}$

We present the ST-A results with different capacity leveraging BERT-base weights in Table 3. Reduction factors 2, 16, and 64 amount to dense adapter dimensions 384, 48, and 12 respectively.

A.5 ST-A and Fusion with ST-A Results with RoBERTa-base

In order to validate our findings of our best setup—ST-A—we re-evaluate our results leveraging RoBERTa-base weights. We present our results in Table 4. Similar to our findigs with BERT-base, especially datasets with less data profit from AdapterFusion. We find that, in contrast to BERT-base, RoBERTa-base does not perform well with high capacity adapters with reduction factor 2.

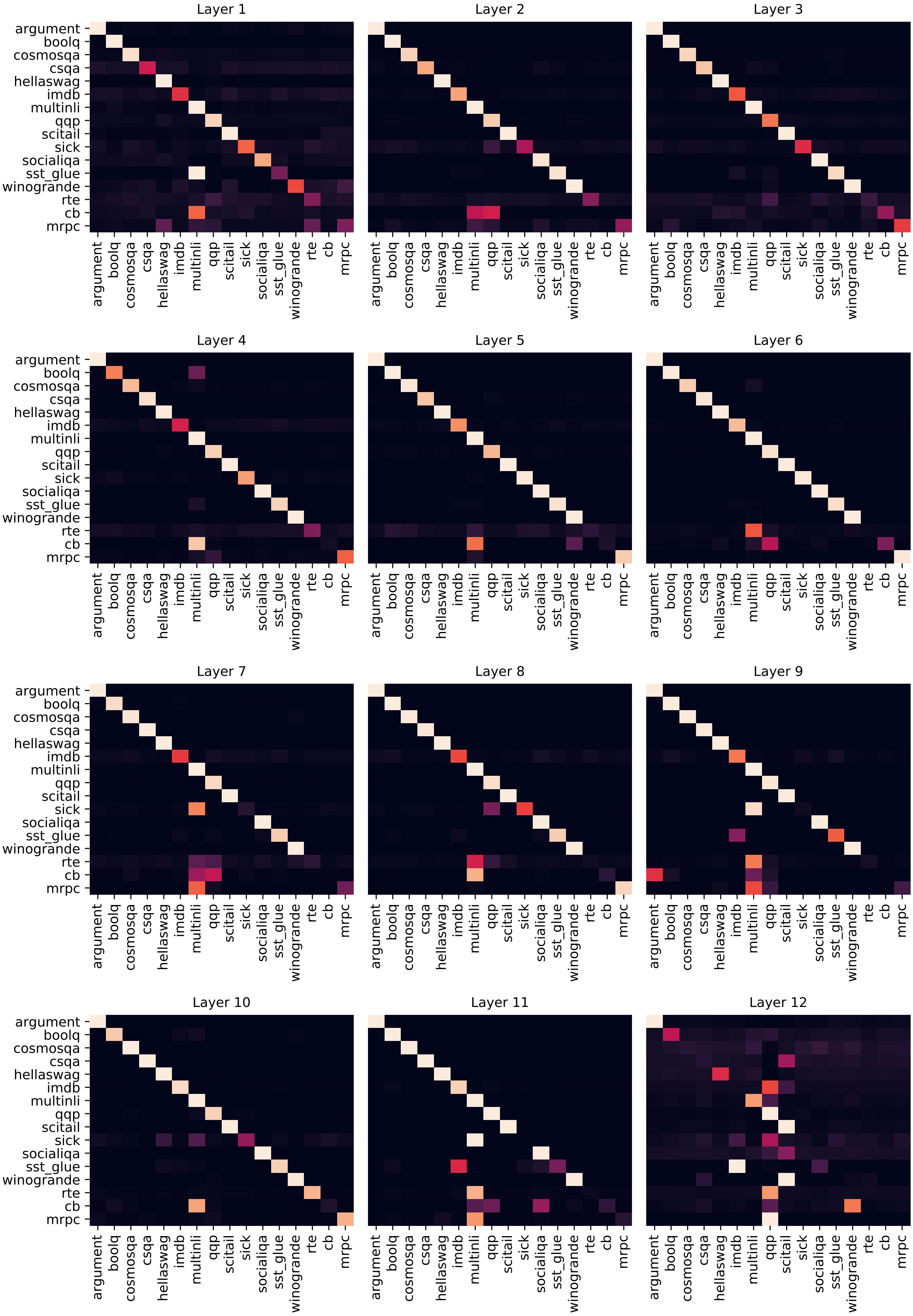
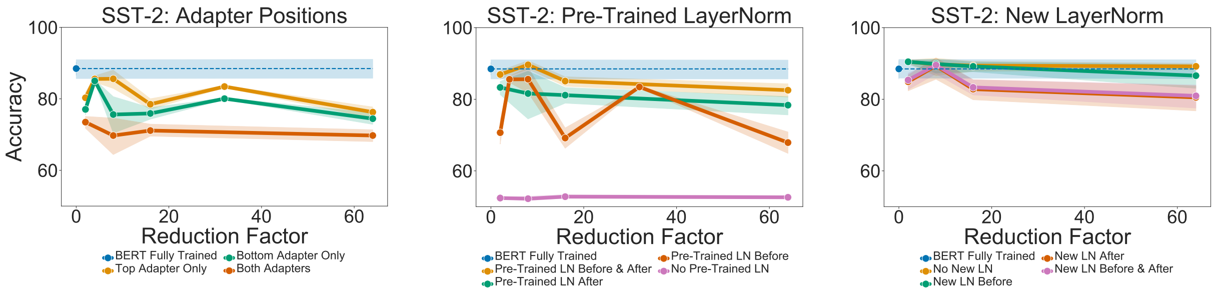


Figure 6: AdapterFusion activations in the 12 BERT-base layers. Target tasks are presented in rows, whereas the set of adapters are displayed in columns. Black squares indicate that an adapter has not been activated, whereas white cells indicate full activation.

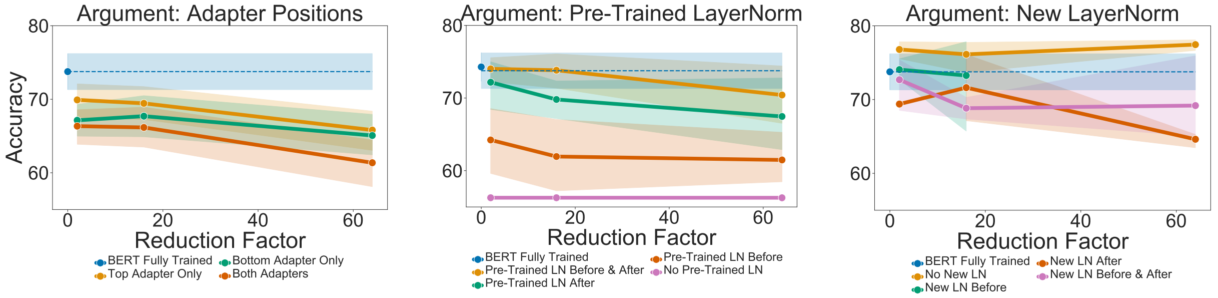


(a) Adapter Positions in Layer

(b) Position of pretrained LayerNorm

(c) Position of newly trained LayerNorm

Figure 7: Results of the grid search on the SST-2 dataset over the architecture settings illustrated on the left of Figure 5. As we go from (a) to (c), the best performing setting is used for further search over other hyperparameters. We find that the best performing architecture is *Top Adapter Only* with *Pretrained LayerNorm Before & After* including *No New LayerNorm*. This Architecture is illustrated on the right of Figure 5.

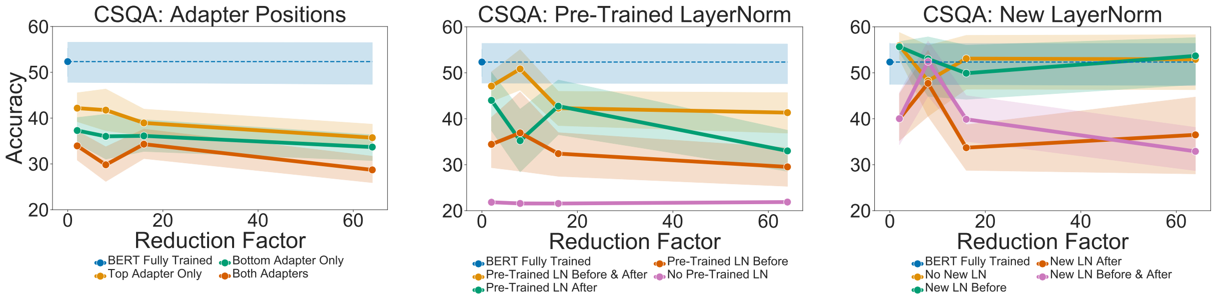


(a) Adapter Positions in Layer

(b) Position of Pretrained LayerNorm

(c) Position of newly trained LayerNorm

Figure 8: Results of the grid search on the Argument dataset over the architecture settings illustrated on the left of Figure 5. As we go from (a) to (c), the best performing setting is used for further search over other hyperparameters. We find that the best performing architecture is *Top Adapter Only* with *Pretrained LayerNorm Before & After* including *No New LayerNorm*. This Architecture is illustrated on the right of Figure 5.



(a) Adapter Positions in Layer

(b) Position of Pretrained LayerNorm

(c) Position of newly trained LayerNorm

Figure 9: Results of the grid search on the CSQA dataset over the architecture settings illustrated on the left of Figure 5. As we go from (a) to (c), the best performing setting is used for further search over other hyperparameters. We find that the best performing architecture is *Top Adapter Only* with *Pretrained LayerNorm Before & After* including *No New LayerNorm*. This Architecture is illustrated on the right of Figure 5.

Dataset	**ST-A$_2$**	**ST-A$_{16}$**	**ST-A$_{64}$**
MultiNLI	84.60	84.32	84.08
QQP	90.57	90.59	89.73
SST	92.66 ±0.32	91.85 ±0.41	92.01 ±0.33
Winogrande	62.11 ±0.09	61.09 ±0.11	59.70 ±0.06
IMDB	94.20 ±0.28	93.85 ±0.07	93.90 ±0.14
HellaSwag	39.45 ±0.20	38.11 ±0.14	38.28 ±0.37
SocialIQA	60.95 ±0.15	62.41 ±0.11	62.23 ±0.73
CosmosQA	59.32 ±0.24	60.01 ±0.02	60.65 ±0.34
SciTail	94.44 ±0.81	93.90 ±0.16	93.82 ±0.49
Argument	76.83 ±0.21	77.65 ±0.34	77.64 ±0.56
CSQA	57.83 ±0.23	58.91 ±0.57	58.88 ±0.40
BoolQ	77.14 ±1.10	75.66 ±1.25	76.07 ±0.54
MRPC	86.13 ±1.59	85.16 ±0.52	85.58 ±0.32
SICK	87.50 ±0.14	86.20 ±0.00	85.70 ±0.42
RTE	70.68 ±4.57	71.04 ±1.62	69.16 ±1.59
CB	87.85 ±2.94	86.07 ±3.87	84.28 ±4.79
Mean	76.39	76.05	75.73

Table 3: Mean and standard deviation results (development sets) for each of the 16 datasets and reduction factors {2, 16, 64} for ST-A. Each model is initialized with BERT-base (Devlin et al., 2019) weights. The datasets are ordered by their respective training dataset size. Dashed horizontal lines separates datasizes $\{>40k, >10k, >5k\}$ respectively.

Dataset	**Head**	**Full**	**ST-A$_2$**	**ST-A$_{16}$**	**ST-A$_{64}$**	**F. w/ ST-A$_{16}$**	**ST-A$_{16}^{\text{Houlsby}}$**
MultiNLI	56.84	86.42	85.56	86.06	85.86	86.20	86.57
QQP	71.40	91.07	90.88 ±0.07	90.27	89.39 ±0.63	90.28	90.66
SST	81.86 ±0.21	94.29 ±0.22	93.71 ±0.29	93.80 ±0.23	93.35 ±0.43	93.67 ±0.13	94.17 ±0.15
Winogrande	51.93	66.77	51.27 ±0.78	65.58 ±0.53	62.43	66.01 ±0.47	63.46 ±6.38
IMDB	85.40	96.00	95.70	95.78 ±0.13	95.80	95.78 ±0.19	95.68 ±0.26
HellaSwag	41.16	63.53	61.09 ±0.08	61.57 ±0.14	61.18 ±0.21	61.52 ±0.07	61.21 ±0.37
SocialIQA	46.87	69.44	69.24	70.14 ±0.40	70.21	70.13 ±0.11	70.78 ±0.17
CosmosQA	41.88 ±0.29	68.52 ±0.49	68.01 ±0.94	68.76 ±0.53	68.62 ±0.55	68.64 ±0.04	69.18 ±0.34
SciTail	49.57	94.47	94.24	94.59 ±0.64	94.32	94.44 ±0.09	94.09 ±0.39
Argument	66.22 ±0.62	78.04 ±0.42	78.60 ±0.34	78.50 ±0.45	78.53 ±0.59	77.98 ±0.24	78.42 ±0.44
CSQA	41.37 ±0.34	65.81 ±0.59	66.11 ±0.60	66.30 ±0.38	64.03 ±0.27	66.52 ±0.18	67.53 ±0.70
BoolQ	62.17	81.89	80.86 ±0.86	80.83 ±0.27	80.17 ±0.25	80.86 ±0.15	81.11 ±0.54
MRPC	68.38 ±0.00	89.11 ±0.93	89.11 ±0.51	88.72 ±0.71	87.10 ±1.67	89.65 ±0.50	89.17 ±1.06
SICK	56.40	86.60	84.80	85.40 ±0.32	85.40	85.76 ±0.26	85.88 ±0.46
RTE	55.81 ±2.92	72.34 ±11.02	61.80 ±12.47	75.30 ±0.61	73.86 ±1.55	78.79 ±1.12	78.56 ±1.54
CB	59.64 ±11.05	90.00 ±1.60	87.14 ±6.85	89.28 ±2.82	81.07 ±4.82	92.86 ±3.79	89.64 ±3.87
Mean	58.05	81.08	78.63	80.83	79.52	**81.41**	81.18

Table 4: Mean and standard deviation results of models initialized with RoBERTa-base (Liu et al., 2019b) weights. Performances are measured on the development sets of the 16 datasets for the different architectural setups. The datasets are ordered by their respective training dataset size. Dashed horizontal lines separate datasizes $\{>40k, >10k, >5k\}$ respectively. **Head** indicates training only a classification head on top of fixed RoBERTa weights. For **Full** training we fine-tune all weights of RoBERTa. Single-Task adapters (**ST-A**) is the training of independently trained adapters for each task, using the architecture illustrated in Figure 5, indices {2, 16, 64} indicate the reduction factor. **Fusion w/ ST-A** show the results of AdapterFusion using the respective pretrained adapters. **ST-A$_{16}^{\text{Houlsby}}$** shows the results of ST-A with with architecture proposed by Houlsby et al. (2019).

CHOLAN: A Modular Approach for Neural Entity Linking on Wikipedia and Wikidata

Manoj Prabhakar Kannan Ravi[1], **Kuldeep Singh**[3], **Isaiah Onando Mulang'**[2],
Saeedeh Shekarpour[4], **Johannes Hoffart**[5], **and Jens Lehmann**[2]

[1]Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
`manoj.prabhakar@hpi.de`
[2]Smart Data Analytics, University of Bonn, Bonn, Germany
`{mulang,jens.lehmann}@cs.uni-bonn.de`
[3]Zerotha Research and Cerence GmbH, Aachen, Germany
`kuldeep.singh1@cerence.com`
[4]University of Dayton, Dayton, USA
`sshekarpour1@udayton.edu`
[5]Goldman Sachs, Frankfurt, Germany
`johannes.hoffart@gs.com`

Abstract

In this paper, we propose CHOLAN, a modular approach to target end-to-end entity linking (EL) over knowledge bases. CHOLAN consists of a pipeline of two transformer-based models integrated sequentially to accomplish the EL task. The first transformer model identifies surface forms (entity mentions) in a given text. For each mention, a second transformer model is employed to classify the target entity among a predefined candidates list. The latter transformer is fed by an enriched context captured from the sentence (i.e. local context), and entity description gained from Wikipedia. Such external contexts have not been used in state of the art EL approaches. Our empirical study was conducted on two well-known knowledge bases (i.e., Wikidata and Wikipedia). The empirical results suggest that CHOLAN outperforms state-of-the-art approaches on standard datasets such as CoNLL-AIDA, MSNBC, AQUAINT, ACE2004, and T-REx.

1 Introduction

The explicit schema, graph-based structure, and interlinking nature of information represented in publicly available knowledge graphs (KGs) e.g., DBpedia (Auer et al., 2007), Freebase (Bollacker et al., 2007), Wikidata (Vrandecic, 2012) or knowledge bases (KBs) such as Wikipedia; introduce a new landscape of features, as well as structured knowledge and embeddings. Researchers have developed several techniques to align information available in unstructured text to the concepts of these KGs (Wu et al., 2019b; Broscheit, 2019).

End-to-end Entity Linking (hereafter EL) task follows this direction; such that, given a sentence EL first identifies the entity mention in the sentence, then maps these mentions to the most likely KG/KB entities. The EL comprises of a three-step process. With respect to the given example sentence *Soccer: Late Goals Give Japan win Over Syria*, the first step called mention detection (MD) identifies the surface forms *Japan* and *Syria*. The next step is candidate generation (CG) aiming to find a list of possible entity candidates in the KG/KB for each entity mention. For example, the candidates list for entity mention *Japan* consists in part of *Japan national football team, Japan (country), Japan (Band)* and for *Syria* is *Syria (Roman province), Syria national football team, Greater Syria*. Finally, the third step deals with the entity disambiguation (ED) which employs the coreference and contextual features to discriminate the most likely entity from the candidates list e.g., *Japan national football team* and *Syria national football team* are correct entities.

Entity Linking approaches are broadly categorised into three categories. The initial attempts (Hoffart et al., 2011; Piccinno and Ferragina, 2014) solve MD and ED as independent sub-tasks of EL (i.e., a pipeline based system). However, these approaches exhibit a behaviour where errors propagate from MD to ED hence might downgrade the overall performance of the system. The second category has emerged in an attempt to mitigate these errors, where researchers focused on jointly modelling MD and ED, emphasising the importance of the mutual dependency of the two sub-tasks (Kolitsas et al., 2018). These two EL

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 504–514
April 19 - 23, 2021.

approaches depend on an intermediate candidate generation step and rely on a pre-computed list of entity candidates. For example, (Kolitsas et al., 2018) propose a joint MD and ED model and inherits the candidate list from (Ganea and Hofmann, 2017). The third approach combines the three sub-steps in a joint model and illustrates that each of those tasks is interdependent (Durrett and Klein, 2014; Broscheit, 2019).

The recent EL approaches focus on jointly modelling two or three subtasks (Sevgili et al., 2020). Furthermore, the NLP research community has extensively used transformers in end-to-end models for entity linking (Broscheit 2019, Peters et al. 2019, and Févry et al. 2020). Nevertheless, these works report less performance than (Kolitsas et al., 2018), which is a bi-LSTM based model. The observations regarding the limited performance of transformer-based models for the EL motivate our work, and in this paper, our focus is to understand the bottlenecks in the entity linking process. We argue that the less studied task in literature, i.e., candidate generation, has an essential role in the EL models' performance, which has not been a focus in the recently proposed transformer-based entity linking models.

In this paper, we **hypothesise** that the transformer models, though trained on a large corpus, may require additional task-specific contexts. Furthermore, inducing the context at the entity disambiguation step may positively impact the overall performance, which has not been utilised in the state of the art methods due to monolithic implementations (Kolitsas et al., 2018; Peters et al., 2019; Broscheit, 2019; Févry et al., 2020). Subsequently, we deviate from the joint modelling of two or three subtasks of the EL and revert to the methodology opted by earlier EL systems in 2011 (Hoffart et al., 2011), i.e. treat each subtask independently. As such, we study the research question: **RQ**: *what is the impact of each sub-task (aka component) on the overall outcome of the transformer-based entity linking approach?* We propose an intuitive novel approach named CHOLAN, comprising a modular architecture of two transformer models to solve MD and ED independently. In the first step, CHOLAN employs BERT (Devlin et al., 2019) model to identify mentions of the entities in an input sentence. The second step involves expanding each mention with a list of KB entity candidates. Finally, the entity mention, sentence (local context), an entity candidate, and entity Wikipedia description (entity context) are fed as input sequences in the second BERT based model to predict the correct KB entity (cf. Figure 1). We train MD and ED steps independently during training, and while testing, we run the CHOLAN pipeline end-to-end for predicting the KB entity. The following are the novel features of CHOLAN:

- The core focus of the approach is to flexibly induce external context and candidate lists in a transformer-based model to improve the EL performance. CHOLAN is independent of a particular candidate list and additional background context. We study four different configurations of CHOLAN to demonstrate the impact of candidate generation step and background knowledge (i.e. entity and sentential context) induced in the model. CHOLAN achieves a new state of the art performance on several datasets: T-REx (ElSahar et al., 2018) for Wikidata; AIDA-B, MSBC, AQUAINT, and ACE2004 for Wikipedia (Hoffart et al., 2011; Guo and Barbosa, 2018).
- CHOLAN is the first approach which is empirically demonstrated to be transferable across KBs having completely different underlying structure and schema i.e., on semi-structured Wikipedia and fully structured Wikidata.

The implementation is publicly available[1]. The paper is structured as follows: next section summarises the related work. Section 3 describes the problem statement and approach. Section 4 explains the experimental settings followed by results in 5. We conclude in Section 6.

2 Related Work

Mention Detection (MD): The first attempt to organise a named entity recognition (NER) task traced back to 1996 (Grishman and Sundheim, 1996). Since then, numerous attempts have been made ranging from conditional random fields (CRFs) with features constructed from dictionaries (Rocktäschel et al., 2013) or feature-inferring neural networks (Collobert and Weston, 2008).

---

[1] `https://github.com/ManojPrabhakar/CHOLAN`

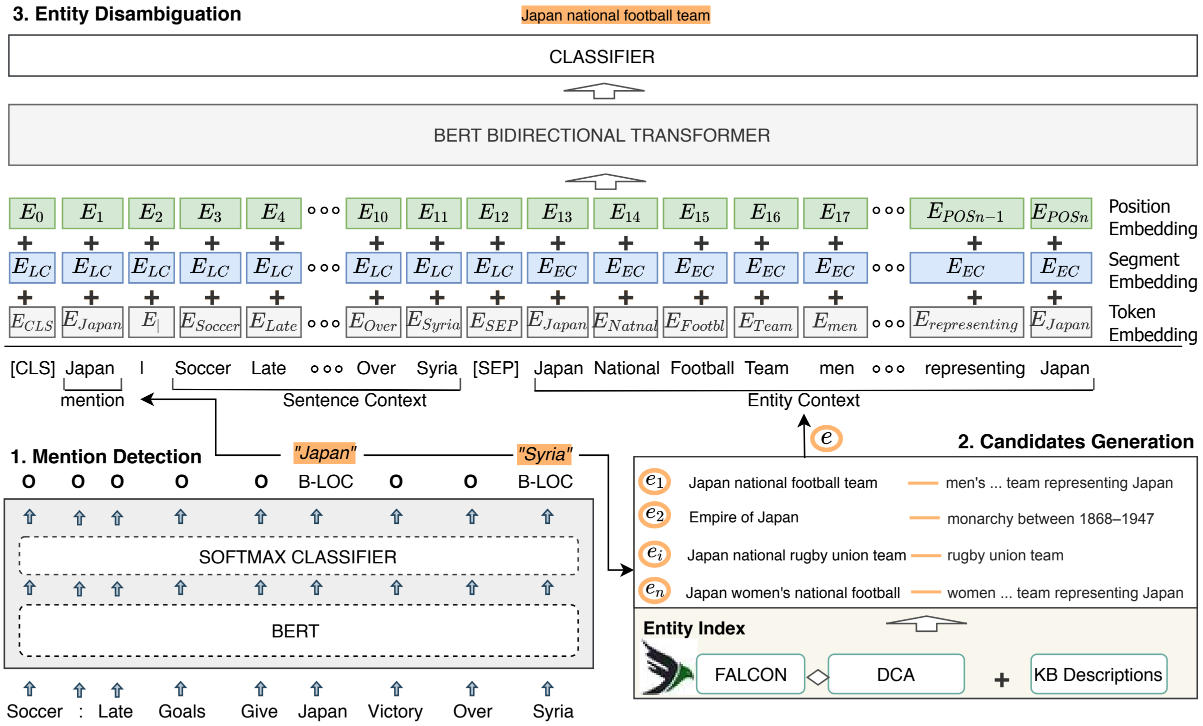


Figure 1: CHOLAN has three building blocks: i) BERT-based Mention Detection that identifies entity mentions in the text ii) Candidate Generation that retrieves a set of entities for the mention iii) Entity Disambiguation: employs BERT transformer model powered by background knowledge from KB and local sentential context.

Recently, contextual embedding based models achieve state of the art for NER/MD task (Akbik et al., 2018; Devlin et al., 2019). We point to the survey by Yadav and Bethard (2018) for details about NER. Few early EL models have performed MD task independently (Ceccarelli et al., 2013; Cornolti et al., 2016).

Candidate Generation (CG): There are four prominent approaches for candidate generation. First is a direct matching of entity mentions with a pre-computed candidate set (Zwicklbauer et al., 2016). The second approach is the dictionary lookup, where a dictionary of the associated aliases of entity mentions is compiled from several knowledge base sources (e.g. Wikipedia, Wordnet) (Sevgili et al., 2020; Fang et al., 2019; Cao et al., 2017). The third approach is to generate entity candidates using empirical probabilistic entity-map $p(e|m)$. The $p(e|m)$ is a pre-calculated prior probability of correspondence between positive mentions and entities. A widely used entity map was built by (Ganea and Hofmann, 2017) from Wikipedia hyperlinks, Crosswikis (Spitkovsky and Chang, 2012) and YAGO (Hoffart et al., 2011) dictionaries. End-to-end EL approaches such as (Kolitsas et al., 2018; Cao et al., 2018) relies on the entity map built by Ganea and Hofmann. The next approach for generating the candidates is proposed by (Sakor et al., 2019). Authors build a local KG by expanding entity mentions using Wikidata and DBpedia entity labels and associated aliases. The local KG can be queried using BM25 ranking algorithm (Logeswaran et al., 2019). The modular architecture of CHOLAN gives us the flexibility to experiment with several ways of generating entity candidates. Hence, we reused candidate list proposed by (Ganea and Hofmann, 2017) and built a new CG approach based on (Sakor et al., 2019).

End to End EL: Few EL approaches accomplish MD and ED tasks jointly. (Nguyen et al., 2016) propose joint recognition and disambiguation of named-entity mentions using a graphical model and show that it improves EL. The work in (Kolitsas et al., 2018) also proposes a joint model for MD and ED. Authors use a bi-LSTM based model for mention detection and computes the similarity between the entity mention embedding and set of predefined entity candidates. The work in (Broscheit, 2019) employs BERT to jointly model three subtasks of the EL. Author employ an entity vocabulary of 700K top most frequent entities to train the model. Work in (Févry et al., 2020) uses a Transformer architecture with large scale pre-training from Wikipedia links for EL. For CG, authors train the model to predict BIO-tagged mention boundaries to disambiguate among all entities. For Wikidata KG, Opentapioca is an entity linking approach which relies on a heuristic-based model for disambiguation of the mentions in a text to the Wikidata entities (Delpeuch, 2020). Arjun (Mulang et al., 2020) is the most similar to our approach CHOLAN and trains two independent neural models for MD and ED. It generates candidates on the fly using a Wikidata entity alias map. Arjun does not induce any context in the model.

3 Problem Statement and Approach

We formally define EL task as follows: given an input sequence of words W = $\{w_1, w_2, w_3, \ldots, w_n\}$, and a set of entities denoted by $\mathcal{E}$ from a KG/KB. The EL task aligns the text into a subset of entities represented as $\Theta : W \rightarrow \mathcal{E}'$ where $\mathcal{E}' \subset \mathcal{E}$. We formulate the EL task as a three step process in which the first step is the mention detection (MD). The MD is a function $\theta_1 : W \rightarrow \mathcal{M}$, where the set of mentions is denoted by $\mathcal{M} = (m_1, m_2, ..., m_k)$ $(k \leq n)$ and each mention m_x is a sequence of words starting from i to end position j: $m_x^{(i,j)} = (w_i, w_{i+1}, ..., w_j)$ $(0 < i, j \leq n)$. The next task is candidate generation where for each mention m_x a set of candidates $C(m_x)$= $\{e_1^x, ..., e_n^x | e_i^x \in \mathcal{E}\}$ is derived. Finally, the entity disambiguation (ED) task aims to map each mention $m_x \in \mathcal{M}$ to the most likely entity from its list of candidates. In our case, we model the ED task as a classification task and augment the input with extra signals as context. For every candidate entity $c_i \in C(m_x)$, the model estimates a probability p_i, thus the most likely entity is the one with the highest probability as $\gamma = \arg\max_{p_i}\{\mathcal{P}(p_i \mid m_x, c_i^x, W, C)\}$ where W and C are the input representations respectively for the given sentence (local context) and the context derived from KG/KB. As such the probability of score p_i is conditioned not only on m_x and c_i^x but also on W and C as contextual parameters.

3.1 CHOLAN Approach

The CHOLAN architecture comprises of three main modules as illustrated in Figure 1.

3.1.1 Mention Detection (MD)

We adapt the vanilla BERT (Devlin et al., 2019) model for the task of entity mention detection in an unstructured text. For each input sentence, we append the special tokens [CLS] and [SEP] to the beginning and end of the sentence, respectively. This is then used as input to the model which learns a representation of the tokens in the sentence. We then introduce a (logistic regression based) classification layer on top of the BERT model to determine named entity tags for each token following the BIO format (Sang and Meulder, 2003). Our BERT† model is initialised using publicly available weights from the pretrained BERT$_{BASE}$ model and is fine-tuned to the specific dataset for detecting a mention m_i. Please note that BERT$_{BASE}$ model is the latest approach which successfully outperformed in various NLP tasks, including MD. Thus, we reuse this model for the completion of our approach.

$$m_i = BERT^\dagger(w_i) \tag{1}$$

3.1.2 Candidate Generation (CG)

One of the critical focus of CHOLAN is to understand the bottleneck at the CG step. Hence, we reuse the DCA candidate list and propose a novel candidate list to understand the candidate generation impact on overall EL performance.

DCA Candidates: (Yang et al., 2019) adapts the probabilistic entity-map $p(e|m)$ created by (Ganea and Hofmann, 2017) (cf. section 2) to calculate the prior probabilities of candidate entities for a given mention. In the probabilistic entity-map, each entity mention has 30 potential entity candidates. Yang and colleagues also provide associated Wikipedia description of each entity. In CHOLAN, we *reuse* candidate set $C(m)$ provided by (Yang et al., 2019) and further consider associated Wikipedia entity descriptions.

Falcon Candidates: (Sakor et al., 2019) created a local index of KG items from Wikidata entities expanded with entity aliases. For example, in Wikidata the entity Q33[2] has the label "Finland". Sakor and colleagues expanded the entity label with other aliases from Wikidata such as “Finlande”, “Finnia”, “Land of Thousand Lakes”, “Suomi”, and “Suomen tasavalta”. We adopt this local KG index to generate entity candidates per entity mention in the employed datasets. The local KG has a querying mechanism using BM25† algorithm (cf. equation (2)) and ranked by the calculated score. We build a predefined candidate set using the top 30 Wikidata entity candidates in

---

[2] `https://www.wikidata.org/wiki/Q33`

$C\_Falcon(m)$ for each entity mention. We enrich the candidates set obtained from Wikidata by the correspondence from Wikipedia. We also add the first paragraph of Wikipedia as entity descriptions (only if Wikidata entity has corresponding Wikipedia page) to the hyperlinks. By selecting two different candidate list, our idea is to understand the impact of candidate generation step on end-to-end entity linking performance.

$$e_i = BM25^{\dagger}(m_i) \tag{2}$$

3.1.3 Entity Disambiguation (ED)

In order to use the power of the transformers, we propose "WikiBERT" to perform the ED task. In WikiBERT, our novel methodological contribution is the induction of local sentential context and global entity context at the ED step in a transformer model, which has not been used in the recent EL models. WikiBERT is derived from the vanilla BERT$_{BASE}$ model and fine-tuned on the two EL datasets (CoNLL-AIDA and T-REx). We view the ED task as sequence classification task. The input to our model is a combination of two sequences. The first sequence S_1 concatenates the entity mention $m \in \mathcal{M}$ and sentence $\mathcal{W}$ where the sentence acts as a local context. The second sequence S_2 is a concatenation of entity candidate $e \in C(m)/C\_Falcon(m)$(obtained from Equation 2) and its corresponding Wikipedia description (entity context ct_i). The two sequences are paired together with special start and separator tokens: ([CLS] S_1 [SEP] S_2 [SEP]). The sequences are fed into the model which in turn learns the input representations according to the architecture of BERT (Devlin et al., 2019). Any given token (local context word, entity mention, or entity context words) is a summation of the three embeddings :

i. *Token embedding*: refers to the embedding of the corresponding token. We make note here on specific tokens that comprises the input representations for our model more specialised as compared to other fine-tuning tasks. The entity mention tokens appended at the beginning of S_1 and separated from the sentence context tokens by a single vertical token bar |, likewise, for the entity context sequence S_2, we prepend the entity title tokens from the KB before adding the descriptions.

ii. *Segment embedding*: each of the sequences receive a single representation such that the segment embedding for the local context E_{LC} refers to the representation for S_1 whereas E_{EC} is the representation of S_2

iii. *Position embedding*: represents the position of the token in an input sequence. A token appearing at the i-th position in the input sequence is represented with E_i

To train the model, we use the negative sampling approach similar to Yamada and Shindo (2019). The candidate list is generated for each identified mention. The desired entity candidate item is labelled as one, and the rest of the incorrect candidate items (from candidate list) are labelled as zero for a given mention. This process iterates over all the identified mentions using Equation 1.

The training process fine-tunes BERT using the contextual input from sentence and Wikipedia resulting into the WikiBERT model (Equation (3)). The model predicts the relatedness of the two sequences by classifying it as either positive or negative.

$$e_i = WikiBERT(m_i, e_i, ct_i) \tag{3}$$

4 Experimental Setup

4.1 Datasets

For Wikidata EL, we rely on T-REx dataset (ElSahar et al., 2018). We adapt the subset of T-REx used by Mulang et al. (2020) for a fair evaluation setting. The dataset contains 983,257 sentences (786,605 in training and 196,652 in the test set) accommodating 3,133,778 instances of surface forms which are linked to 85,628 distinct Wikidata entities. T-REx does not have a separate validation set to fine-tune the hyperparameters. Therefore, we further divide the train set into a 90:10 ratio for training and validation.

For EL over Wikipedia, we adapt standard dataset CoNLL-AIDA proposed by (Hoffart et al., 2011) for the training. The dataset contains 18,448 linked mentions in 946 documents, a test set of 4,485 mentions in 231 documents, and a validation set of 4,791 mentions in 216 documents. For testing, we use AIDA-B (test) dataset from (Hoffart et al., 2011) and MSNBC, AQUAINT, ACE2004 datasets from (Guo and Barbosa, 2018).

4.2 Models for Comparison

4.2.1 Baselines over Wikidata

We now briefly explain Wikidata baselines.

1. OpenTapioca (Delpeuch, 2020): is a heuristic-based end-to-end approach that depends on topic similarity and mapping coherence for linking

Wikidata entity in an input text.
2. Arjun (Mulang et al., 2020): is a pipeline of two attentive neural networks employed for MD and ED. Arjun is the SotA, and we take baseline values from Arjun's paper.

4.2.2 Baselines over Wikipedia

1. (Hoffart et al., 2011): build a weighted graph of entity mentions and candidate entities. Then, the model computes a dense subgraph that predicts the best joint mention-entity mapping.
2. DBpedia Spotlight (Mendes et al., 2011) proposes a probabilistic model and relies on the context of the text to link the entities.
3. KEA (Steinmetz and Sack, 2013) employs a linguistic pipeline coupled with metadata generated from several Web sources. The candidates are ranked using a heuristic approach.
4. Babelfy (Moro et al., 2014) is a graph-based approach that uses loose identification of candidate meanings coupled with the densest subgraph heuristic to link the entities.
5. Piccinno and Ferragina (2014): to solve entity linking, authors focus on mentions recognition and annotations pruning to propose a voting algorithm for entity candidates using PageRank.
6. Kolitsas et al. (2018) train MD and ED task jointly using word and character-level embeddings. The model reuses candidate set from (Ganea and Hofmann, 2017) and generates a global voting score to rank the entity candidates.
7. Peters et al. (2019) induce multiple KBs into a large pretrained BERT model with a knowledge attention mechanism.
8. Broscheit (2019) trains MD, CG, ED task jointly using a BERT-based model. Besides, an entity vocabulary containing 700K most frequent entities in English Wikipedia was utilised.
9. Févry et al. (2020) consider large scale pretraining from Wikipedia links as the context for a transformer model to predict KB entities.
In Wikipedia-based experiments, we report values from (Févry et al., 2020) and (Kolitsas et al., 2018) for AIDA-B test set. On MSNBC (MSB), AQUAINT (AQ), and ACE2004 (ACE) test datasets, only (Kolitsas et al., 2018), DBpedia Spotlight (Mendes et al., 2011), KEA (Steinmetz and Sack, 2013), and Babelfy (Moro et al., 2014) report the values and we compare against them.

Hyper-parameters	**Value**
Epochs	4
Batch size	8
Learning rate	$2e^{-5}$
Learning rate decay	linear
Adam β_1	0.9
Adam β_2	0.999
dropout	0.1
Loss Function	Cross-Entropy
Classifier	Softmax

Table 1: Hyper-parameters during fine-tuning.

4.3 CHOLAN Configurations

We configure CHOLAN model applying various candidate generation approaches detailed below.
CHOLAN-Wikidata: we train the model using T-REx dataset and employ $C\_Falcon(m)$ candidate set. The ED model (WikiBERT) is fed with the sentential context but not with entity description as not all Wikidata entities have a corresponding Wikipedia entity.
CHOLAN-Wiki+FC: is trained on CoNLL-AIDA (Hoffart et al., 2011). For CG step, we employ Falcon candidate set $C\_Falcon(m)$. Here, the ED model (WikiBERT) is only fed with the sentential context.
CHOLAN-Wiki+DCA: We train the MD and ED models on CoNLL-AIDA. The CG step involves DCA candidate set $C(m)$. During ED step (WikiBERT), Wikipedia descriptions associated with each entity is fed along with sentential context.
CHOLAN: inherits **CHOLAN-Wiki+FC** but in addition, Wikipedia entity description is induced into the ED model (WikiBERT).

4.4 Metrics and Hyper-parameters

On Wikidata-based experiments, we employ standard metrics of accuracy i.e., precision (P), recall (R), and F-score (F) same as (Mulang et al., 2020). For Wikipedia-based datasets, we use Micro-F1 score in strong matching setting (Kolitsas et al., 2018). The strong matching needs exactly predicting the gold mention (i.e. target entity mention) boundaries and its corresponding entity annotation in the KB. To compare the recalls of two CG approaches, we report the performance on gold recall. Gold recall is the percentage of entity mentions for which the candidate set contain the ground truth entity (Yao et al., 2019).
We have implemented all our models in PyTorch[3]

---

[3] `https://pytorch.org/`

and optimized using Adam (Kingma and Ba, 2015). We used the pre-trained BERT models from the Transformers library (Wolf et al., 2019). We ran all the experiments on a single GeForce GTX 1080 Ti GPU with 11GB size. Table 1 outlines the hyper-parameters used in the fine-tuning on both the datasets. We followed the standard settings suggested by (Devlin et al., 2019). The average run time is 9.31 hours/epoch for CHOLAN and without description, it was 7.23 hours/epoch.

5 Results

We study the following research question:*what is the impact of each sub-task (aka component) on the overall outcome of the transformer-based entity linking approach?* We further investigate a sub-research question: how do the external context and the candidate generation step impact the overall performance of CHOLAN? Our every experiment systematically studies the research questions in different settings.

Model	**P**	**R**	**F**
Delpeuch 2020	40.7	**82.9**	57.9
Mulang et al. 2020	71.4	71.2	71.3
CHOLAN-Wikidata	**75**	76	**75.4**

Table 2: Comparison on T-REx test set for Wikidata EL. Best values in bold.

5.1 Results on Wikidata dataset

Table 2 summarises CHOLAN performance on T-REx dataset. CHOLAN-Wikidata configuration outperforms the baselines. We dig deeper into our reported values. We observe that for MD task, our F-score is 94.3 (compared to 77 F-score of Arjun (Mulang et al., 2020)). However, the gold recall for CG step is 81.2. We generate the entity candidates using an information retrieval approach (BM25† algorithm) to get the top 30 candidates based on the confidence score. The Wikidata KG is challenging, and many labels share the same name. It contributes to a large loss in the F-score for the CG step. For instance, the entity mention "National Highway" matches exactly with four Wikidata ID labels while 2,055 other entities contain the full mention in their labels. Please note that we did not perform retraining of (Kolitsas et al., 2018) (SOTA on Wikipedia EL) on the T-REx dataset since we determined that the model is tightly coupled and relies on pre-computed Wikipedia candidate list from (Ganea and Hofmann, 2017).

5.1.1 Ablation Study on Wikidata

We study the impact of local context on the performance of CHOLAN. Therefore, we exclude the sentence as input in the ED step at training and testing time. Hence, the inputs to the ED model are only entity mention and the entity candidates gained from the CG step. We observe that the performance drops when the local sentential context is not fed (cf. Table 3). It justifies our choice to feed the model by the sentence during the ED task.

Model	**P**	**R**	**F**
CHOLAN-Wikidata	**75**	**76**	**75.4**
CHOLAN-Wikidata (WLC†)	72	73.5	72.7

Table 3: The ablation study on T-REx test set for Wikidata EL. Best values in bold. WLC† denotes model without local context. When the local sentential context is excluded from ED, the performance drops.

5.2 Results on Wikipedia datasets

Table 4 reports the performance of CHOLAN's configurations on AIDA-B test set. The first configuration is "CHOLAN-Wiki+ FC" in which MD and ED models are trained using CoNLL-AIDA. We notice a clear jump in the performance. We then replaced the Falcon candidate list $C\_Falcon(m)$ with DCA candidates $C(m)$ resulting into "CHOLAN-Wiki+ DCA". In DCA candidates, the description of entities is attached. The performance is increased when an additional background knowledge as an entity description is fed. Our next configuration is CHOLAN where we attached Wikipedia entity descriptions in Falcon candidate list $C\_Falcon(m)$ (as a modification of "CHOLAN-Wiki+ FC"). This setting outperforms all the existing baselines and previous CHOLAN configurations. Our experiments illustrate the impact of CG step and background knowledge on end-to-end EL performance. The improvement of CHOLAN continues to the other three test datasets where the jump is significantly higher compared to the baselines (cf. Table 5). Reported values in Table 5 also approves transferability of CHOLAN when we apply cross-domain experiments.

5.2.1 Ablation Study on Wikipedia

We conducted three ablation studies to understand the behaviour of CHOLAN's configurations over Wikipedia datasets. The first study

Model	Micro F1
Hoffart et al. 2011	72.8
Mendes et al. 2011	57.8
Steinmetz and Sack 2013	42.3
Moro et al. 2014	48.5
Piccinno and Ferragina 2014	73
Kolitsas et al. 2018	82.4
Peters et al. 2019	73.7
Broscheit 2019	79.3
Févry et al. 2020	76.7
CHOLAN-Wiki+ FC	75.1
CHOLAN-Wiki+ DCA	77.5
CHOLAN	**83.1**

Table 4: Comparison on *AIDA-B*. Best value in bold and previous SOTA value is underlined.

Model	MSB	AQ	ACE
Mendes et al. 2011	40.6	45.2	60.5
Steinmetz and Sack 2013	30.9	35.9	40.3
Moro et al. 2014	39.7	35.8	17.8
Kolitsas et al. 2018	72.4	40.4	68.3
CHOLAN-Wiki+ FC	77.8	70	85.7
CHOLAN-Wiki+ DCA	78.3	75.9	71.3
CHOLAN	**83.4**	**76.8**	**86.8**

Table 5: The micro F1 scores are listed from the comparative study over three datasets (out of domain). The model is trained over CoNLL-AIDA dataset. Best value in bold and previous SOTA value is underlined.

is to calculate the Gold recall values for various datasets. CHOLAN uses the candidates from $C\_Falcon(m)$ candidate set for each entity mention. While generating the candidate set from local KG of (Sakor et al., 2019) we observe a drop in the Gold recall as reported in Table 6. CG plays a crucial role in trading off precision and recall. We conclude that more robust CG approaches likely impact overall performance. The second ablation study is about to calculate the performance of our configurations for ED step, i.e., running WikiBERT in isolation. Here, we assume that all entities are truly recognised; thus, our focus of the study is the ED model. We report the impact of various candidate generation approaches on the ED model in Table 7. The significant jump in the performance from ”CHOLAN-Wiki+FC Vs CHOLAN” contributes to the additional background knowledge provided in CHOLAN as entity candidate descriptions. The third ablation study tests the impact of sentential context fed into two configurations on a Wikipedia dataset. Table 8 reports the achieved performance after excluding sentence as the additional context. Obviously, the performance decreases. The model shows similar behaviour on T-REx in Table 3. These observations confirm our hypothesis as the ED model is enhanced using additional contexts.

Model	AIDA-B	MSB	AQ	ACE
Falcon Candidates	94	93.8	85.3	97.3
DCA Candidates	98.3	98.5	94.2	90.6

Table 6: Gold Recall for Candidate Generation techniques over Wikipedia test datasets.

Model	Micro F1
Kolitsas et al. 2018	83.8
CHOLAN-Wiki+ FC	78.4
CHOLAN-Wiki+ DCA	79.1
CHOLAN	**85.7**

Table 7: Comparison on *AIDA-B* for ED. Best score in bold and previous SOTA value is underlined.

Model	Micro F1
CHOLAN-Wiki+ DCA	77.5
CHOLAN-Wiki+ DCA (WLC†)	71.2
CHOLAN	**83.1**
CHOLAN (WLC†)	79.6

Table 8: Ablation study on *AIDA-B*. We observe that when local sentential context is removed from ED step, the performance drops. Best values in bold. WLC† denotes model without local context.

6 Conclusions

In the last two years, the NLP research community has extensively tried transformer-based models for the EL task. However, the performance remained lower than Kolitsas et al. (2018). This paper combines the traditional software engineering principle of modular architecture with the context-induced transformers to effectively solve the EL task. Our reason to deviate from an end-to-end architecture was to provide full flexibility to our system in terms of candidate generation list, underlying KG, and induction of the context at the ED step. We attribute CHOLAN's outperformance to the following reasons: 1) the modular architecture, which brings flexibility and interoperability as CHOLAN can treat each task independently. Kolitsas et al. (2018) reports that shifting towards joint modelling of MD and ED tasks helps mitigate error propagation from MD to ED. However, the performance of BERT$_{BASE}$ for the MD task is significantly high (92.3 on AIDA-B and 94.3 F1-

score on T-REX calculated by us) remarkably reducing the errors in MD. CHOLAN leverages this capability in the MD subtask, placing more focus on CG and ED tasks. 2) The flexibility in architecture further permits us to induce sentence and entity descriptions as additional contexts. Furthermore, using candidate list in plug and play manner has resulted in a significant increase in the performance. In earlier transformer approaches, the implementation is monolithic and context is not utilised. There are scopes for improvement in our approach. Wu et al. (2019a) introduces a novel CG method that retrieves candidates in a dense space defined by a bi-encoder and can be used as alternate CG approach. We aim for scaling CHOLAN to multilingual entity linking as a viable next step.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*

Kurt D. Bollacker, Robert P. Cook, and Patrick Tufts. 2007. Freebase: A Shared Database of Structured General Human Knowledge. In *AAAI 2007*.

Samuel Broscheit. 2019. Investigating entity knowledge in bert with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685.

Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural collective entity linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 675–686.

Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1623–1633.

Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. 2013. Dexter: an open source framework for entity linking. In *Proceedings of the sixth international workshop on Exploiting semantic annotations in information retrieval*, pages 17–20.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. 2016. A piggyback system for joint entity mention detection and linking in web queries. In *Proceedings of the 25th International Conference on World Wide Web*, pages 567–578.

Antonin Delpeuch. 2020. Opentapioca: Lightweight entity linking for wikidata. *The 1st Wikidata Workshop co-located with International Semantic Web Conference 2020 (to appear).*

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

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the association for computational linguistics*, 2:477–490.

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *LREC*.

Zheng Fang, Yanan Cao, Qian Li, Dongjie Zhang, Zhenyu Zhang, and Yanbing Liu. 2019. Joint entity linking with deep reinforcement learning. In *The World Wide Web Conference*, pages 438–447.

Thibault Févry, Nicholas FitzGerald, Livio Baldini Soares, and Tom Kwiatkowski. 2020. Empirical evaluation of pretraining strategies for supervised entity linking. In *Automated Knowledge Base Construction*.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629.

Ralph Grishman and Beth M Sundheim. 1996. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web*, 9(4):459–479.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 782–792.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460.

Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia spotlight: shedding light on the web of documents. In *I-SEMANTICS*.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

Isaiah Onando Mulang, Kuldeep Singh, Akhilesh Vyas, Saeedeh Shekarpour, Ahmad Sakor, Maria Esther Vidal, Soren Auer, and Jens Lehmann. 2020. Encoding knowledge graph entity aliases in an attentive neural networks for wikidata entity linking. *Web Information System and Engineering*.

Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2016. J-nerd: joint named entity recognition and disambiguation with rich linguistic features. *Transactions of the Association for Computational Linguistics*, 4:215–229.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Francesco Piccinno and Paolo Ferragina. 2014. From tagme to wat: a new entity annotator. In *Proceedings of the first international workshop on Entity recognition & disambiguation*, pages 55–62.

Tim Rocktäschel, Torsten Huber, Michael Weidlich, and Ulf Leser. 2013. Wbi-ner: The impact of domain-specific features on the performance of identifying and classifying mentions of drugs. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 356–363.

Ahmad Sakor, Isaiah Onando Mulang, Kuldeep Singh, Saeedeh Shekarpour, Maria-Esther Vidal, Jens Lehmann, and Sören Auer. 2019. Old is gold: Linguistic driven approach for entity and relation linking of short text. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019,Volume 1 (Long and Short Papers)*, pages 2336–2346. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

Ozge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2020. Neural entity linking: A survey of models based on deep learning.

Valentin I Spitkovsky and Angel X Chang. 2012. A cross-lingual dictionary for english wikipedia concepts.

Nadine Steinmetz and Harald Sack. 2013. Semantic multimedia information retrieval based on contextual descriptions. In *Extended Semantic Web Conference*, pages 382–396. Springer.

Denny Vrandecic. 2012. Wikidata: a new platform for collaborative data collection. In *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*, pages 1063–1064.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771v5.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2019a. Zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*.

Shanchan Wu, Kai Fan, and Qiong Zhang. 2019b. Improving distantly supervised relation extraction with neural noise converter and conditional optimal selector. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 7273–7280. AAAI Press.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.

Ikuya Yamada and Hiroyuki Shindo. 2019. Pre-training of deep contextualized embeddings of words and entities for named entity disambiguation. *arXiv preprint arXiv:1909.00426*.

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 271–281.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.

Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. 2016. Robust and collective entity disambiguation through semantic embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 425–434.

Grounding as a Collaborative Process

Luciana Benotti
Universidad Nacional de Córdoba
CONICET, Argentina
`luciana.benotti@unc.edu.ar`

Patrick Blackburn
Philosophy and Science Studies, IKH
Roskilde University, Denmark
`patrickb@ruc.dk`

Abstract

Collaborative grounding is a fundamental aspect of human-human dialog which allows people to negotiate meaning. In this paper we argue that it is missing from current deep learning approaches to dialog and interactive systems. Our central point is that making mistakes and being able to recover from them collaboratively is a key ingredient in grounding meaning. We illustrate the pitfalls of being unable to ground collaboratively, discuss what can be learned from the language acquisition and dialog systems literature, and reflect on how to move forward.

1 Introduction

Collaborative grounding is shaped by *constraints* that are not explicit in successful dialog turns. These constraints combine information from world *causal relations*, the *task* under discussion, the *communicative intents* of the dialog partners, and much else besides. They are used to *negotiate meaning*, to decide which beliefs to add to the shared *common ground*, which is constructed using the *joint attention* of the dialog partners to things either real or imagined. Once beliefs are grounded, they cannot be magically ungrounded without further negotiation, for the dialog partners are *committed* to them. But it's tricky:

> *Human-analogous natural language understanding (NLU) is a grand challenge of artificial intelligence, which involves mastery of the structure and use of language and the ability to ground it in the world.* (*Bender and Koller, 2020*)

What does "the ability to ground it in the world" involve? Implicit constraints that shape grounding become explicit when communication starts to break. We claim that *making mistakes* and being able to recover from them *collaboratively* is a key ingredient of the ability to ground.

We proceed as follows. We first discuss obstacles to current research on dialog and interactive systems (Section 2) and then present working definitions of *collaborative grounding* and related concepts (Section 3). Section 4 plays with scenarios illustrating the pitfalls of being unable to ground collaboratively; we then turn to the language acquisition literature for insight into how humans do it (Section 5) and to the dialog systems literature to discuss what is known (Section 6). In Section 7, we reflect on the progress made and make recommendations, while in Section 8 we note possible objections to our account and conclude.

2 Motivation

Dialog and interactive systems is one of the most popular research areas in computational linguistics nowadays. But — unlike machine translation and information retrieval — deep learning approaches to it have had little impact on products that people use daily.[1] In 1991, cognitive scientist Brennan asked: *Why is it that natural language has yet to become a widely used modality of human/computer interaction?* (Brennan, 1991), and in 2020 the AI researchers de Vries, Bahdanau and Manning (de Vries et al., 2020) asked the same question yet again. In 1990, research on dialog systems used symbolic approaches; today neural generative models are favoured. Methods have changed, but the question remains the same.

Neural generative models offer flexibility, can be easily adapted to new domains, and require minimal domain engineering. But though they generate fluent responses (Serban et al., 2016), the result is often boring and repetitive ("I don't know") or they contradict themselves, or wander away from the

---

[1] Almost all commercial dialog systems currently available seem to be based on pre-deep learning pipeline architectures (Takanobu et al., 2020) in spite of efforts such as the Alexa Prize (Ram et al., 2018; Gabriel et al., 2020).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 515–531
April 19 - 23, 2021.

topic of conversation (Li et al., 2016). De Vries et al (2020) note that neural generative models assume that the required data is available and appropriate. They say: *Ideally, the data used for training and evaluating should reflect the intents and linguistic phenomena found in real-world applications and be of reasonable size to accommodate modern data-intensive methods.* They remark that data quality and quantity are hard to reconcile, and that the community has prioritized quantity over quality; thus dialog systems are inadequate because datasets are poor. But ineffective use of dialog history seems to play a role too. Sankar et al (2019) study model sensitivity to artificially introduced context perturbations at test time. Working with multi-turn dialog datasets, they found that commonly used neural dialog architectures, like recurrent and transformer-based seq2seq models, are rarely sensitive to perturbations such as missing or reordered utterances, and word shuffling.

The ineffective use of dialog history often goes unnoticed because of the evaluation practices that are common nowadays. Automatic metrics such as BLUE, ROUGE, and so on, do not correlate well with human judgement, either for semantic preserving natural language generation or for dialog (Mathur et al., 2020). For human evaluation, it is not enough to show a few turns to the annotators (Liu et al., 2016). This does not measure how well the system is able to recover from its own mistakes; a human-in-the-loop evaluation that judges the overall interaction is needed for that (Walker et al., 1997).

Graphical user interfaces (GUIs), on the other hand, tend to get things right. We agree with Brennan (1998) that GUIs are more successful in everyday use than dialog systems (DSs) because GUIs enable collaborative grounding effectively while deep learning approaches to DSs do not.

3 What is collaborative grounding?

In this section we provide working definitions of collaborative grounding and other key concepts. To get the ball rolling, we take the *common ground* to be the commitments that the dialog partners have (explicitly or implicitly) agreed upon.

Collaborative grounding not symbol grounding *Collaborative grounding* is the process of seeking and providing incremental evidence of mutual understanding through dialog; we view the ongoing exchange of speaker and hearer roles as fundamental to conversation (Benotti, 2010; Benotti and Blackburn, 2014). When the speaker believes that the dialog is on track, *positive evidence* of understanding is provided in different forms (depending on the communication channel) such as explicit acknowledgements and eye contact. *Negative evidence* of understanding signals that something needs to be negotiated before the dialog partners can commit — and negative evidence is ubiquitous:

Conversations with other people are rarely fluent and without mishap, and people do not expect them to be. (Brennan, 1991)

Mishaps lead to the need for *repair*. Repair is fundamental to *conversational analysis* (Schegloff et al., 1977; Schegloff, 2007), the linguistic study of language-driven social interactions. Together with the more psychologically oriented work of Clark and his colleagues (Clark and Wilkes-Gibbs, 1986; Clark, 1996), conversational analysis, is a key inspiration for the lines of research that we review in this paper.

We consider collaborative grounding to be distinct from symbol grounding (Harnad, 1990) though they interact in interesting ways (Larsson, 2018). *Symbol grounding* (or perceptual grounding, or language grounded in vision) is the set of capabilities that link symbols with perceptions; it is an important research area (Roy, 2005) in which accurate connections between systems' linguistic representations and sensor data are often viewed as proof that a system means what it says. These connections are important for meaning, but we agree with De Vault et al. (2006) that *perceptual grounding is neither necessary nor sufficient to justify the attribution of linguistic meaning*. Human perception and memory are neither accurate nor stable, and different people have different abilities and limitations. Human meaning attributions do not rely on the accurate perceptions and perfect memory sought by symbol grounding, but on a *collaborative negotiation process* in which language speakers coordinate their perceptual memories and linguistic usage with other members of their communities. If a dialog system *commits* itself to negotiate its meanings collaboratively when perception and memory falter, then we claim that this gives grounds for assigning linguistic meaning to it. See Section 4 and A for examples.

Collaborative grounding: basic mechanisms When people talk to each other, they tailor their utterances to their partners. People can talk with friends, strangers, disembodied voices on the telephone, readers who will come along after they are gone, foreigners, children, and even dogs. Flexibility in tailoring utterances for a particular addressee has been documented even among the very young; five year olds have been observed to use more simple language and a different pitch range when talking to three year olds than they do talking to adults (Golinkoff, 1986). People adapt by initially estimating the common ground they share with a particular partner, by monitoring the positive and negative evidences of understanding (aka *grounding acts*) and by adapting their initial common ground estimate accordingly.

Alikhani and Stone (2020) explain that dialog systems can participate in collaborative grounding by ensuring they get attention and feedback from their users and tracking user state. Such *pragmatic mechanisms* have been explored, including those for dealing with problems related to joint attention (Koller et al., 2012; Koleva et al., 2015; Tan et al., 2020), engagement (Bohus and Horvitz, 2014; Foster et al., 2017), turn taking and incremental interpretation (Schlangen and Skantze, 2009; Selfridge et al., 2012; DeVault and Traum, 2013; Eshghi et al., 2015) corrections and clarifications (Villalba et al., 2017; Ginzburg and Fernández, 2010) and dialog management (DeVault and Stone, 2009; Selfridge et al., 2012). These mechanisms have been studied for different kinds of applications (Denis, 2010; Dzikovska et al., 2010, 2012). In Section 6 we discuss this research tradition; we believe it can provide top-down research guidance for research on dialog systems that commit to what they say.

Collaborative grounding: exposing limitations As we said at the start, collaborative grounding is shaped by constraints that may not be explicit in successful dialog turns. Dialog partners use constraints to negotiate meaning, adding beliefs to the common ground through their joint attention to a real or imagined world. Further negotiation is required to unground a belief; unilateral belief withdrawal leads to a sense of commitment being brushed aside.

A dialog system is an unusual kind of linguistic partner. A user must figure out what its *limitations* are and then design her utterances with them in mind. Limitations may arise due to gaps in grammatical coverage, an unexpectedly narrow domain vocabulary, or a lack of *common sense*; by common sense we here mean the basic knowledge and competencies required for successful navigation through a world full of objects, time, money, politeness, animals, people, and so on. One much-studied component of commonsense involves *causality* (Pearl, 2009) and the *frame problem* (Shanahan, 2016), the ability to figure out what changes and what does not change due to various kinds of (inter)action. Our point, however, is *not* that dialog systems need all these components of commonsense — it is rather that they need enough *interpersonal* common sense to not try and hide their inadequacies!

Often, however, these limitations are deliberately well hidden by the designers — who thereby unintentionally inhibit users from employing the powerful meaning-negotiation strategies available in a human-human dialog. Dialog systems may generate words they cannot understand, thus building false expectations of common ground in their dialog partner. Humans cannot adapt and ground collaboratively if a dialog system fails to *expose its own limitations*. In the next section we explore examples of such failures.

4 Test scenarios

This section uses playful scenarios to illustrate the implications of not learning to do collaborative grounding appropriately. We start with a text adventure scenario, then consider a multimodal scenario followed by a BlenderBot example, and conclude by contrasting these with the collaborative grounding typical of GUIs.

4.1 Dungeon intelligence

Consider the following interaction with the text game adventure AI Dungeon[2] powered by GPT-2 (Radford et al., 2019). The Webtext dataset used to train GPT-2 contains web content from Reddit links that have been rated as trustable by some Reddit users, probably including blogs, online news, and other discourse data.

In the example, when we detect a problem, we prompt Dungeon AI (DAI) with a negative grounding act (for example, *but before you only had seven, how many are there really?*). This makes the inconsistency with the previous turns explicit; in effect, we offer to negotiate. But the model blithely ig-

---

[2] https://play.aidungeon.io/ accessed on 19/08/2020

nores the negative evidence and talks about something else:

DAI: You are wearing nondescript clothing, and you manage to escape without being recognized. You open the purse and find seven gold coins, which is good since you only planned to steal a little.
Human: How many coins are there in the purse?
DAI: You count the coins and find that you have eighty one coins, which is good since you planned to steal just a little. You think to yourself "I should be able to afford a room tonight."
Human: But before you only had seven, how many are there really?
DAI: You head to the first inn you see, which seems to be a place where merchants stay. The desk clerk is a young woman who is probably only just thirteen or fourteen.

We posed this question several times but the model never tried to negotiate the inconsistency (for example, by replying: *there are 81 coins, I saw only some of them before*). In A there is a continuation of this dialog, illustrating other kinds of inconsistency and lack of collaborative grounding.

Bender and Koller (2020) argue that language models like GPT-2 are unable to ground language in the world: because they are only exposed to form, it is unreasonable to expect them to negotiate meaning. Others propose to tackle all kinds of grounding experimentally, by collecting datasets grounded in various modalities, developing models that learn from them, and using leaderboards (Linzen, 2020) and checklists (Ribeiro et al., 2020) to measure how effectively the model generalizations aligns with those of humans. So let's turn to a multimodal task: visual dialog.

4.2 Visual dialog

Visual dialogs have long been a test-bed for natural language understanding. They played a prominent role in early work on natural language understanding (Winograd, 1972) and are now the focus of an active community investigating the interplay between computer vision and computational linguistics (Baldridge et al., 2018; Shekhar et al., 2019). Important progress has been obtained thanks to the recent release of datasets like VisDial (Das et al., 2017) and GuessWhat?! (de Vries et al., 2017); the former contains chit-chat conversations about an image whereas the latter is a visual game, hence its dialogs are goal oriented.

As we see in Figure 1, GuessWhat?! is a cooperative game: two players attempt to identify an object in an image. The *Questioner* has to guess the referent by asking yes/no questions; the *Oracle* knows the referent object and provides the answers. In the figure, the referent is the second woman from right to left of the group of four women in the back.

The Oracle model that includes the dialog history as an input (de Vries et al., 2017), correctly answers 9 out of 10 questions for the dialog in Figure 1; it only fails on question number 9, incorrectly answering "no". But question 9 does not look particularly hard; why does it create a problem? *Because it trades on the common ground built by the previous dialog turns.* Humans use common ground here, as shallower methods fail (the referent is not salient in the image). Indeed — except for one turn — Figure 1 is just visual question answering; the system fails on precisely the question that *cannot* be correctly answered without using the common ground built by the previous turns. Is this error only due to the rarity of such dialog turns (as (Agarwal et al., 2020) argue), something which might be fixed by better datasets?

The answer may be "no". Collaborative grounding in visual dialog has *two* characteristics that make it difficult for current machine learning methods: it is not only that turns requiring common ground may be infrequent in these datasets, in addition — as in our example — they may not be lexicalized. Common ground dependency is easier to spot when it is lexicalized with explicit pronouns (e.g. 'them' in "is it one of them?" in position 7). If the question "is it the second one?" had been asked in position 2, the model's answer "no" would have been correct, as "second" would then refer to the second (highly salient) man in the row. But in position 9 the correct answer is "yes" due to the common ground built by turns 6 to 8. Common ground dependence may be less lexicalized in visual dialog as the visual modality tends to favor visual saliency to resolve ambiguities.

As we said in Section 3, common ground is the set of commitments that the dialog partners have agreed upon, while collaborative grounding is the process by which such commitments are made. Common ground dependence, as illustrated in turn 9 in Figure 1, may not be as frequent and as lexicalized in multimodal datasets as current data hungry models need (Agarwal et al., 2020; de Vries et al., 2020). But better datasets alone will not solve the

1. It is a person?	yes
2. It is a male?	no
3. Is it a female?	yes
4. Is she in the red dress?	no
5. She is in black?	yes
6. In the back from right to left you see four women?	yes
7. Is it one of them?	yes
8. Right to left, is it the first one, gray sweater?	no
9. Is it the second one?	yes
10. Does she look to have panty hose on?	yes

Figure 1: Human-human dialog on the Guesswhat?! referential task extracted from (de Vries et al., 2017). The example illustrates the linguistic variability present in this task.

problem: we need models that are trying to do collaborative grounding. That is, we need models that are not only led by the lexical cues in the dialog but also by pragmatic mechanisms reflected by timely exchanges between speaker and hearer. So let us turn to a model that has been exposed to such pragmatic mechanisms in its data.

4.3 BlenderBot

Facebook AI has built and open-sourced BlenderBot, which they call the largest-ever open-domain chatbot. It outperforms systems such as Google's Meena (Adiwardana et al., 2020) in terms of engagement and also feels more human, according to human evaluators (Smith et al., 2020). BlenderBot attempts to combine different conversational skills including empathy, knowledge, and personality together in one system. The trained models are available for research (Smith et al., 2020) in different sizes and with different hardware requirements. The largest model has 9.4 Billion parameters, the middle sized version has 2.7 Billion.

Unlike the GPT-2 model we discussed earlier which is mostly trained on discourse data, BlenderBot is pre-trained with a large dataset of multi-party conversations extracted from the Reddit Pushshift dataset (Baumgartner et al., 2020). The dataset consists of free-form exchanges between multiple speakers and hearers where collaborative grounding is occurring. Thus the data on which BlenderBot models are pre-trained includes positive and negative evidence of understanding. BlenderBot models are then fine-tuned on dialogs that have been crowdsourced to exhibit empathy, knowledge about some particular topic, a consistent persona, and on a crowdsourced dataset that blends these abilities together (Smith et al., 2020).

The following interaction was generated using the middle sized model trained on Reddit Pushshift through July 2019 (Smith et al., 2020); we call the model BB2.7B.[3]

Human: Do you know Blender?
BB2.7B: Yes, I've used it a little bit. I am not very good at it, though.
Human: But I don't mean the 3D software.
BB2.7B: Oh, I see. Yeah, I know a little bit about it. What do you like to do with it?
Human: I mean Blender, the chatbot. Have you used it?
BB2.7B: Oh, yes, I have used it. I like to use it to make 3D models.[4]

This fragment explores BlenderBot reaction to negative evidence (*but I don't mean the 3D software*). It does not ignore it, as the Dungeon AI based on GPT-2 does. Indeed it replies with a coherent follow up and it includes a sentence that seems intended to acknowledge the misunderstanding (*Oh, I see*). However, the rest of the dialog shows that in spite of recognizing the structure of negative evidence, BlenderBot is unable to integrate negative grounding into the conversation consistently.

4.4 Graphical user interfaces

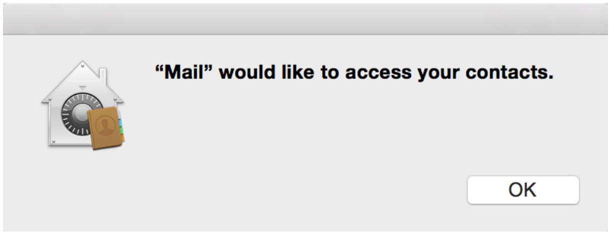
GUIs exploit graphical elements that mimic physical objects: we can point, drag and toss them in the trash bin. GUIs respond by updating immediately, thus the relationship between the user's action and the graphical result is utterly clear. Even though GUIs are primarily graphical, they are also conversational and implement pragmatic mechanisms;

---

[3] The model is available at https://parl.ai/projects/recipes/

[4] See https://www.youtube.com/watch?v=-op_yOXBNEY for the full dialog.

indeed, their response is as timely and relevant as backchannels in human conversation (acknowledgments, nods, eye contact; see (Gravano and Hirschberg, 2011)). As they enable direct manipulation, they trivially solve the linguistic reference problem. They model common ground by tracking what is visible to the user. They model joint attention graphically through focus. And they do not suffer from the frame problem: consistency is carefully preserved in GUI design (Brennan, 1998); if you move something, it will stay there until somebody moves it back. They ask for both positive (e.g. ok) and negative (e.g. cancel) evidence for understanding. GUIs are good at exposing their own limitations and most users are good at adapting to them — some even overadapt and blame themselves for misunderstandings. For example, take the dialog box window in Figure 2. The system wants positive grounding evidence from the user but (confusingly) it does not offer the choice of giving negative grounding evidence (the conventional negative grounding label for buttons in GUIs is "cancel").



"Mail" would like to access your contacts.

Figure 2: Sample dialog box not allowing negative evidence of grounding: there is no cancel button.

5 Human language acquisition

Clark (2001) presents evidence that collaborative grounding underpins the process of first language acquisition in babies. She argues that collaborative grounding offers a way of placing a new piece of the language at the center of *joint attention* of the language learner and her caregiver. In particular, it is through such pragmatic mechanisms that:

> *(a) children solve the general mapping problem between form and meaning when offered new words; (b) they take up conventional terms for unfamiliar objects and events, and see how they relate to familiar terms; (c) they take up information about what they should have said when they have produced an erroneous utterance when offered reformulations. (Clark, 2001)*

Babies around one year old have been shown to perform *negative grounding acts* in order to *repair* a request that they made (e.g. "doll!") whose *intention* (e.g. "getting the doll") was *misunderstood* by an adult. Babies tend to do the repair act even when the request is satisfied by other means (maybe the frustrated adult gave the baby all the toys including the doll — yet the baby takes the doll and repeats "doll!"). In other words, they care that their intention is understood, not only satisfied (Ackermann et al., 2014; Tomasello et al., 2005). Tomasello et al argue that this is a basic ability, one required for the development of language and cognitive capabilities like belief attribution. Golinkoff describes this ability as follows:

> *Importantly, from their earliest forays into linguistic communication, infants engage in a "negotiation of meaning" in which they request clarification from the adult and produce communicative repairs for the adult when needed [...] Infants can and will persevere in the face of failure by altering their signals in creative non stereotypical ways. (Golinkoff, 1986)*

Developmental psychologists have documented repeatedly that children with autism have difficulties signaling non-comprehension and making appropriate repairs to their own linguistic messages (Katherin et al., 1990). Deaf and hearing children have been found to employ different repair strategies. Deaf children were also more likely to revise utterances; hearing children more likely to provide cue repairs. When facing communication breakdown, both deaf and hearing children persisted effectively in clarifying utterances (Ciocci SR, 1998).

Allwood and colleagues have documented that grounding acts have a central role not only for first language acquisition but also for second languages (Allwood et al., 1991; Allwood, 1993, 1997; Allwood and Ahlsen, 1999).

6 Previous work, key insights

In this section we focus on previous work (and key insights) on collaborative grounding from research on human dialog analysis and dialog systems. We won't cover work from robotics and symbol grounding; for that see e.g. (Roy and Reiter, 2005; Bohus and Rudnicky, 2009; Bohus et al., 2012; Larsson, 2018).

Long before the deep learning era, dialog system researchers were aware that constructing common ground *collaboratively* is a key task. A notable pioneer was Traum (1991; 2003) with his focus on dialog turns whose conversational role is to provide positive and negative evidence of grounding as tools for negotiating meaning. The interaction between dialog system and human computer interaction research was fruitful back then, but the tools available today for dealing with language variability (Mrksic et al., 2017) were not yet developed so systems were brittle (Allen et al., 2001).

DeVault and Stone (2009) showed that using predefined semantic tags (e.g. colors of objects) falls short for human dialog; people tend to invent new tags collaboratively as needed (e.g. using *greenish blue* to distinguish one object from a bluer one). This example could be described as "zooming into" the details of the available distinctions, but "zooming out" can occur when limitations are revealed. Saying *I am color blind* will (hopefully) shift the dialog away from reliance on color terms.

Rieser et al. (2010; 2011) and Georgila et al. (2005) propose using Wizard of Oz data instead of naturally occurring human-human dialog for training; this puts the spotlight on the necessary constraints and primes dialog systems with explicit grounding subdialogs for overcoming limitations, instead of restricting attention to the limitations of the channel through which the human crowdsourcers interact. This enables data to be collected that makes explicit strategies for negotiating meaning, and could allow systems to learn particular collaborative grounding skills relevant for the dialog system task.

The surface form of explicit negotiations of meaning in dialog are frequently non-sentential utterances (Fernandez, 2006; Fernández et al., 2007). These include prototypical positive and negative evidence of grounding such as acknowledgements and clarification requests (Stoyanchev et al., 2013; Benotti and Blackburn, 2017), but also less-well-known forms such as self-corrections, rejections, modifiers and plain old questions and answers (Purver, 2004; Purver et al., 2018). Such work makes it evident that non-sentential utterances are not errors of performance and do not need "fixing" into sentential utterances. Ginzburg and Fernandez (2010) contributed detailed formalizations of how different evidences of grounding modified the public common ground and the private commitments of each dialog participant. A simple observation in (Ginzburg, 2012) does a big job in illustrating how different written discourse and dialog can be: it is often said that the most common word in written discourse is *'the'* while the most frequent word in naturally occurring conversations in the British National Corpus (Clear, 1993) is *'yes'*. This makes it evident that the contexts available to the dialog partners in the aftermath of an utterance are *not* identical. Positive acknowledgements (like *'yes'*) signal that the participants are synchronized and that collaborative grounding is proceeding smoothly (Denis et al., 2007).

In Section 3 we said that the ongoing exchange of speaker and hearer roles is fundamental to conversation. Schlangen and others have shown in detail how the ongoing exchange of these roles is so natural that we complete and correct each others sentences in *incremental* approaches to dialog (Schlangen and Skantze, 2009; DeVault et al., 2009; Baumann and Schlangen, 2012; DeVault and Traum, 2013; Kennington and Schlangen, 2017).

Hough and Schlangen (2017) argue that embodied dialog systems must ground the degree of uncertainty they have; that is they must *expose their limitations* as we argued in Section 3 and as GUI systems routinely do. They show that humans can reliably understand the level of uncertainty that a robot has and act accordingly. No complex natural language generation abilities are needed for this; negative grounding can be realized by perceivable hesitations in a physical act done by the robot.

Koller and colleagues explain how the joint attention of the conversational participants can be considered and manipulated in order to correct the common ground. In particular, they show how listener attention can be manipulated when there is co-presence through speech and gaze (Koller et al., 2012; Koleva et al., 2015) and through emphasis in the text ("press the RED button") when co-presence is not possible (Villalba et al., 2017).

In sum, *the possibility of making mistakes and collaboratively recovering from them is one of the key pragmatic mechanisms for grounding meaning*. Much is already understood about this process, and in Section 7 we recommend using this work to motivate top down advances in dialog and interactive systems that would complement the bottom up approaches described in Section 2.

7 Moving forward

In *What Computers Can't Do* (1972), Dreyfus drew on the ideas of philosophers like Merleau-

Ponty (Merleau-Ponty, 1962) and Heidegger (Heidegger, 1993) to criticize symbolic AI. Dreyfus emphasized the embodied capability of *knowing how*, rather than the abstract propositional *knowing that* typical of symbolic AI; he did not anticipate that AI would find plausible methods (deep neural nets, embodied robotics, distributional semantics) for exploring knowing how.

Dreyfus' criticisms may seem obvious in retrospect, but it is useful to recall another philosopher that he cited. In *Philosophical Investigations*, Ludwig Wittgenstein (1953) critiqued earlier approaches to language and meaning (including his own) for failing to take the collaborative aspect of language into account.

Wittgenstein's later work foregrounds the importance of *social interaction*. In Sections 3 and 4 we remarked that collaborative grounding is more than symbolic/perceptual grounding, and claimed that the crucial missing component is provided by social interaction. In Section 5 we saw that human children are born into is a complex world of agents, relationships, affect and much else beside. Moreover (as the child soon learns) it is a world in which interesting others collaborate with the help of a malleable system called language. This system is capable of expressing multiple types of meaning — symbolic, scientific, social — which Wittgenstein summed up anthropologically: language was a form of life (*Lebensform*).

Wittgenstein's ideas are inspiring, but we don't need to look so far back: the work we reviewed in Sections 5 and 6 shows that social aspects of language use are central to the roots of meaning, and cannot be an afterthought in dialog models. This leads to our recommendations.

Appreciate socially grounded dialog datasets Dialog datasets are scarce as most naturally created dialog corpora cannot be shared due to privacy issues. Matters are different in written discourse, which has made great progress thanks to vast corpora from news services, Wikipedia and the like. We agree with de Vries et al (2020) that more effort should be put into designing what they call *ecologically valid* datasets for dialog. We agree that datasets should avoid four common issues: synthetic language, artificial tasks, not working with prospective users, and single-turn interfaces. We add two items to this wish-list: (1) dialog grounded in at least one modality and (2) dialog grounded in a (symmetric or asymmetric) social collaboration.

Regarding (1), there is much work that has been done on data collection in the visual (i.e. seeing) modality (Baldridge et al., 2018) and more recently in the kinaesthetic (i.e. moving) modality (Weihs et al., 2020). The work we reviewed in Section 6 is a good starting point for designing tasks that also consider (2) for various modalities: auditory (hearing) (Schlangen and Skantze, 2009), tactile (touching) (Hough and Schlangen, 2017), kinaesthetic (moving) (Foster et al., 2014), and visual (seeing) (Koleva et al., 2015). Dialogs grounded in social collaboration can be symmetric or asymmetric. In symmetric dialogs, the dialog roles are socially exchangeable (e.g. (Ilinykh et al., 2019; Haber et al., 2019)). In asymmetric dialogs the roles of the dialog partners are different due to expertise, social power or for some other reason. Typical examples include instruction dialog (Anderson et al., 1991) and medical or technical support (Janarthanam and Lemon, 2009).

Our final recommendation regarding datasets is to let humans collaborate freely during the first round of data collection but then to collect more restricted datasets on the same task once the limitations of the model are clear. That is: let people adapt to its limitations and collect further data that reflects the collaborative adaptation mechanisms.

Use datasets better Better data alone may not be enough; we need to consider models that explicitly tackle collaborative grounding. Examples of such work that builds on fundamental research (like that reviewed in Section 6) already exists. Consider, for example, the following exchange extracted from a human-human dataset (Andreas et al., 2020) collected through a Wizard of Oz methodology:

User: What time is my planning meeting?
Agent: You meet with Grace at noon.
User: Sorry, I meant all-hands.
Agent: Your all-hands meeting is at 2:30 pm.

The fragment contains the negative grounding act *Sorry, I meant all-hands*; which requires access to the user's previous utterance to obtain the revised intention: *What time on Tuesday is my all-hands meeting?*. Promising results have been obtained using hybrid learned-symbolic dialog systems that explicitly model the intention by grounding it into an application domain. For example (Andreas et al., 2020) represent intentions (including grounding acts) as programs that modify the common ground; (El Asri et al., 2017) track the common ground using frames, (Lison and Kennington, 2016) do so using Bayesian networks, and (Ultes et al., 2018) using entities. Such approaches have been rather marginalized in favor of more shallow ones.

Interact with models in order to test them Asking someone "do you need history to answer this question?" is not the same as answering it correctly without history. Most human evaluation of dialog systems are about self perception, not about actually performing an action. As is done in Human Computer Interaction, let people interact with models and then rate them; do not just offer pairs of turns. Dialog is *not* the concatenation of pairs of dialog turns, as noted by (Walker et al., 1997; Schlangen and Skantze, 2009; Ginzburg, 2012; Agarwal et al., 2020) and others (see Section 6).

Focus on error recovery, not error avoidance. Explore your dataset thinking about the *constraints* present in the dialog system you are building: Can they be learned from the data that you have? What are the limitations that you know your system will have? How can you expose these limitations to the dialog partner so that she can adapt to them? Will a Wizard of Oz setup in which a human simulates the system limitations help here?

The metric for evaluating a dialog system should not (only) be accuracy on some static dataset, but also: *how many mistakes you cannot recover from when interacting with a potentially adversarial human being*. Other areas of NLP are already using such evaluations (Nie et al., 2020).

Design with collaborative grounding in mind Many deep learning dialog systems differ from simple question answering approaches in recording the dialog history up to some limit, usually dictated by the number of tokens that can be reasonably encoded as model input (e.g. (Agarwal et al., 2020; Smith et al., 2020)). Which leads to a question: if collaborative grounding occurs in the conversations on which these models are trained, what exactly is missing in (say) BlenderBot, which was trained on Reddit conversations and has a dialog memory of at least a few turns? Is it that current training approaches do not capture this skill? For example could it be that the pre-trained model could exhibit such a skill but that fine-tuning to given tasks (on datasets where collaborative grounding does not occur) wipes it out? Or is it that something more is needed? We believe that attention needs to be paid to the collaborative grounding mechanisms reviewed in Section 6.

Don't leave the social aspects till last Bisk et al (2020) give a detailed description of milestones that must be passed to achieve what they call *truly contextual language understanding*. They argue that we are currently close to the written word milestone, the next milestone being perception, the following one embodiment, with the final one being the social. This may be motivated by the fact that humans evolved through perception and embodiment before developing language.

However language is not just about words and their semantics; it is primarily about people, their intentions and their interactions. We urge the community *not* to leave the social aspects of language for the future but to (re)consider existing work on collaborative grounding (Sections 3 and 6) and language acquisition (Section 5). We need to face up to language as *lebensform* right from the start.

8 Conclusion

Possible objections range from the practical (*new datasets are expensive; old datasets are too small for deep learning*) through the theoretical (*much coherent dialog does not require conversational history, task-oriented dialog may requires collaborative grounding but chit chat does not*) to the technical (*grounding acts cannot be identified automatically; deep learning models can learn to do some collaborative grounding without explicitly representing grounding intentions*). But lack of space precludes further discussion, so with these noted, we conclude by restating our basic point.

The idea that simply training models on large amounts of successful dialog as is currently done will lead to human-analogous collaborative grounding seems dubious. As we have described and illustrated, human grounding has to do not only with successful exchanges and positive grounding, but also with negative grounding. Negative grounding includes pragmatic mechanisms by which we negotiate what we actually mean. It is about errors and corrections, about exposing limitations. Until we figure out how to train these, our dialog models will lack a crucial capability: commitment to their own meanings.

Acknowledgments

We thank the anonymous reviewers and area chair for their detailed reviews and insightful comments.

References

Hermann Ackermann, Steffen R Hage, and Wolfram Ziegler. 2014. Brain mechanisms of acoustic communication in humans and nonhuman primates: An evolutionary perspective. *Behavioral and Brain Sciences*, 37:529–546.

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *arXiv*, abs/2001.09977.

Shubham Agarwal, Trung Bui, Joon-Young Lee, Ioannis Konstas, and Verena Rieser. 2020. History for visual dialog: Do we really need it? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8182–8197, Online. ACL.

Huda Alamri, Vincent Cartillier, Abhishek Das, Jue Wang, Anoop Cherian, Irfan Essa, Dhruv Batra, Tim Marks, Chiori Hori, Peter Anderson, Stephan Lee, and Debi Parikh. 2019. Audio visual scene-aware dialog. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7550–7559.

Malihe Alikhani and Matthew Stone. 2020. Achieving common ground in multi-modal dialogue. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 10–15. ACL.

James F. Allen, Donna K. Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. 2001. Toward conversational human-computer interaction. *AI Magazine*, 22(4):27–37.

Jens Allwood. 1993. Feedback in second language acquisition. In *Adult Language Acquisition: Cross Linguistic Perspectives, II: The Results*, pages 196–235. Cambridge University Press.

Jens Allwood. 1997. Notes on dialog and cooperation. In *IJCAI-97 Workshop on Collaboration, Cooperation and Conflict in Dialogue Systems*, pages 8–10.

Jens Allwood and Elisabeth Ahlsen. 1999. Learning how to manage communication with special reference to the acquisition of linguistic feedback. *Journal of Pragmatics*, 31:1353–1389.

Jens Allwood, Joakim Nivre, and Elisabeth Ahlsen. 1991. On the semantics and pragmatics of linguistic feedback. Technical Report 64, Gothenberg Papers in Theoretical Linguistics, Sweden.

Anne H. Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. 1991. The HCRC map task corpus. *Language and Speech*, 34:351–366.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Jason Baldridge, Tania Bedrax-Weiss, Daphne Luong, Srini Narayanan, Bo Pang, Fernando Pereira, Radu Soricut, Michael Tseng, and Yuan Zhang. 2018. Points, paths, and playscapes: Large-scale spatial language understanding tasks set in the real world. In *Proceedings of the First International Workshop on Spatial Language Understanding*, pages 46–52. ACL.

Timo Baumann and David Schlangen. 2012. INPRO_iSS: A component for just-in-time incremental speech synthesis. In *Proceedings of the ACL 2012 System Demonstrations*, pages 103–108, Jeju Island, Korea. ACL.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *Proceedings of the Fourteenth International AAAI Conference on Web and Social Media, ICWSM 2020, Held Virtually, Original Venue: Atlanta, Georgia, USA, June 8-11, 2020*, pages 830–839. AAAI Press.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198. ACL.

Luciana Benotti. 2010. *Implicature as an interactive Process*. Ph.D. thesis, Université de Lorraine. Supervised by Patrick Blackburn.

Luciana Benotti and Patrick Blackburn. 2014. Context and implicature. In Patrick Brézillon and Avelino J. Gonzalez, editors, *Context in Computing - A Cross-Disciplinary Approach for Modeling the Real World*, pages 419–436. Springer.

Luciana Benotti and Patrick Blackburn. 2017. Modeling the clarification potential of instructions: Predicting clarification requests and other reactions. *Computer Speech Language*, 45:536–551.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020.

Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 8718–8735. ACL.

Dan Bohus and Eric Horvitz. 2014. Managing human-robot engagement with forecasts and... um... hesitations. In *Proceedings of the 16th International Conference on Multimodal Interaction*, page 2–9. ACM.

Dan Bohus, Ece Kamar, and Eric Horvitz. 2012. Towards situated collaboration. In *NAACL-HLT Workshop on Future Directions and Needs in the Spoken Dialog Community: Tools and Data*, pages 13–14. ACL.

Dan Bohus and Alexander I. Rudnicky. 2009. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech and Language*, 23(3):332–361.

Susan E Brennan. 1991. Conversation with and through computers. *User modeling and user-adapted interaction*, 1(1):67–86.

Susan E. Brennan. 1998. The grounding problem in conversations with and through computers. In S. R. Fussell and R. J. Kreuz, editors, *Social and Cognitive Approaches to Interpersonal Communication*, pages 201–225. Lawrence Erlbaum, Hillsdale, NJ.

Hyundong Cho and Jonathan May. 2020. Grounding conversations with improvised dialogues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2398–2413. ACL.

Baran J Ciocci SR. 1998. The use of conversational repair strategies by children who are deaf. *American Annals of the Deaf*, 143(3):235–45.

Eve Clark. 2001. Grounding and attention in the acquisition of language. In *Proceedings of the 37th meeting of the Chicago Linguistic Society*, pages 95–116.

Herbert Clark. 1996. *Using Language*. Cambridge University Press.

Herbert H. Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1 – 39.

Jeremy H. Clear. 1993. *The British National Corpus*, pages 163–187. MIT Press, Cambridge, MA, USA.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra. 2017. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335.

Alexandre Denis. 2010. Generating referring expressions with reference domain theory. In *Proceedings of the Sixth International Natural Language Generation Conference*. ACL.

Alexandre Denis, Guillaume Pitel, Matthieu Quignard, and Patrick Blackburn. 2007. Incorporating asymmetric and asynchronous evidence of understanding in a grounding model. In *11th Workshop on the Semantics and Pragmatics of Dialogue-DECALOG 2007*.

David DeVault, Iris Oved, and Matthew Stone. 2006. Societal grounding is essential to meaningful language use. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, pages 747–754. AAAI Press.

David DeVault, Kenji Sagae, and David Traum. 2009. Can I finish? learning when to respond to incremental interpretation results in interactive dialogue. In *Proceedings of the 10th Annual SIGDIAL Meeting on Discourse and Dialogue*. ACL.

David DeVault and Matthew Stone. 2009. Learning to interpret utterances using dialogue history. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 184–192. ACL.

David DeVault and David Traum. 2013. A method for the approximation of incremental understanding of explicit utterance meaning using predictive models in finite domains. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1092–1099. ACL.

Hubert Dreyfus. 1972. *What Computers Can't Do*. MIT Press.

Myroslava O. Dzikovska, Peter Bell, Amy Isard, and Johanna D. Moore. 2012. Evaluating language understanding accuracy with respect to objective outcomes in a dialogue system. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 471–481. ACL.

Myroslava O. Dzikovska, Johanna D. Moore, Natalie Steinhauser, and Gwendolyn Campbell. 2010. The impact of interpretation problems on tutorial dialogue. In *Proceedings of the 2010 Conference of the Association for Computational Linguistics*, pages 43–48. ACL.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219, Saarbrücken, Germany. ACL.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th SIGdial Meeting on Discourse and Dialogue*, pages 37–49, Saarbrücken, Germany. ACL.

Arash Eshghi, Christine Howes, Eleni Gregoromichelaki, Julian Hough, and Matthew Purver. 2015. Feedback in conversation as incremental semantic update. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 261–271, London, UK. ACL.

Raquel Fernandez. 2006. *Non-sentential utterances in dialogue: Classification, resolution and use*. Ph.D. thesis, University of London.

Raquel Fernández, Jonathan Ginzburg, and Shalom Lappin. 2007. Classifying non-sentential utterances in dialogue: A machine learning approach. *Computational Linguistics*, 33(3):397–427.

Mary Ellen Foster, Andre Gaschler, and Manuel Giuliani. 2017. Automatically classifying user engagement for dynamic multi-party human-robot interaction. *International Journal of Social Robotics*, 9(5):659–674.

Mary Ellen Foster, Simon Keizer, and Oliver Lemon. 2014. Towards action selection under uncertainty for a socially aware robot bartender. In *ACM/IEEE international conference on human-robot interaction*, pages 158–159.

Raefer Gabriel, Yang Liu, Anna Gottardi, Mihail Eric, Anju Khatri, Anjali Chadha, Qinlang Chen, Behnam Hedayatnia, Pankaj Rajan, Ali Binici, Shui Hu, Karthik Gopalakrishnan, Seokhwan Kim, Lauren Stubel, Kate Bland, Arindam Mandal, and Dilek Hakkani-Tür. 2020. Further advances in open domain dialog systems in the third alexa prize socialbot grand challenge. *Proceedings of the Alexa Prize*.

Kallirroi Georgila, James Henderson, and Oliver Lemon. 2005. Learning user simulations for information state update dialogue systems. In *Ninth European Conference on Speech Communication and Technology*.

Jonathan Ginzburg. 2012. *The Interactive Stance*. Oxford Press.

Jonathan Ginzburg and Raquel Fernández. 2010. Computational models of dialogue. In Alexander Clark, Chris Fox, and Shalom Lappin, editors, *Handbook of Computational Linguistics and Natural Language Processing*. Blackwell.

Roberta Golinkoff. 1986. 'I beg your pardon?': the preverbal negotiation of failed messages. *Journal of child language*, 13:455–76.

Agustín Gravano and Julia Hirschberg. 2011. Turn-taking cues in task-oriented dialogue. *Computer Speech and Language*, 25(3):601–634.

Janosch Haber, Tim Baumgärtner, Ece Takmaz, Lieke Gelderloos, Elia Bruni, and Raquel Fernández. 2019. The PhotoBook dataset: Building common ground through visually-grounded dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1895–1910. ACL.

Stevan Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335 – 346.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1766–1776, Vancouver, Canada. ACL.

Martin Heidegger. 1993. The end of philosophy and the task of thinking. *Basic Writings*, pages 427–449.

Julian Hough and David Schlangen. 2017. It's not what you do, it's how you do it: Grounding uncertainty for a simple robot. In *Proceedings of the International Conference on Human-Robot Interaction*, pages 274–282. ACM.

Nikolai Ilinykh, Sina Zarrieß, and David Schlangen. 2019. Meet up! a corpus of joint activity dialogues in a visual environment. In *Proceedings of the 23rd Workshop on the Semantics and Pragmatics of Dialogue*, London, United Kingdom.

Srinivasan Janarthanam and Oliver Lemon. 2009. Learning lexical alignment policies for generating referring expressions for spoken dialogue systems. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pages 74–81. ACL.

Loveland Katherin, Robin McEvoy, and Belgin Tunali. 1990. Narrative story telling in autism and down's syndrome. *British Journal of Developmental Psychology*, 8(1):9–23.

Casey Kennington and David Schlangen. 2017. A simple generative model of incremental reference resolution for situated dialogue. *Computer Speech and Language*, 41:43–67.

Jin-Hwa Kim, Nikita Kitaev, Xinlei Chen, Marcus Rohrbach, Byoung-Tak Zhang, Yuandong Tian, Dhruv Batra, and Devi Parikh. 2019. Codraw: Collaborative drawing as a testbed for grounded goal-driven communication. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 6495–6513. ACL.

Nikolina Koleva, Martín Villalba, Maria Staudte, and Alexander Koller. 2015. The impact of listener gaze on predicting reference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 812–817. ACL.

Alexander Koller, Konstantina Garoufi, Maria Staudte, and Matthew Crocker. 2012. Enhancing referential success by tracking hearer gaze. In *Proceedings of the 13th Meeting of the Special Interest Group on Discourse and Dialogue*, pages 30–39. ACL.

Dimosthenis Kontogiorgos, Elena Sibirtseva, and Joakim Gustafson. 2020. Chinese whispers: A multimodal dataset for embodied language grounding. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 743–749, Marseille, France. European Language Resources Association.

Satwik Kottur, José M. F. Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2019. CLEVR-dialog: A diagnostic dataset for multi-round reasoning in visual dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–595, Minneapolis, Minnesota. ACL.

Staffan Larsson. 2018. Grounding as a side-effect of grounding. *Topics in Cognitive Science*, 10(2):389–408.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 110–119. ACL.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Meeting of the Association for Computational Linguistics*, pages 5210–5217. ACL.

Pierre Lison and Casey Kennington. 2016. OpenDial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of ACL-2016 System Demonstrations*, pages 67–72, Berlin, Germany. ACL.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132. ACL.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997. ACL.

Maurice Merleau-Ponty. 1962. *Phenomenology of Perception*. Routledge.

Nikola Mrksic, Ivan Vulic, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gasic, Anna Korhonen, and Steve J. Young. 2017. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. ACL.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901. Association for Computational Linguistics.

Judea Pearl. 2009. *Causality: Models, Reasoning and Inference*, 2nd edition. Cambridge University Press, USA.

Matthew Purver. 2004. *The Theory and Use of Clarification Requests in Dialogue*. Ph.D. thesis, King's College, University of London.

Matthew Purver, Julian Hough, and Christine Howes. 2018. Computational models of miscommunication phenomena. *Topics in Cognitive Science*, 10(2):425–451.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, Open AI.

Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, Eric King, Kate Bland, Amanda Wartick, Yi Pan, Han Song, Sk Jayadevan, Gene Hwang, and Art Pettigrue. 2018. Conversational AI: the science behind the alexa prize. *arXiv*, abs/1801.03604.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912. ACL.

Verena Rieser and Oliver Lemon. 2011. *Reinforcement learning for adaptive dialogue systems: a data-driven methodology for dialogue management and natural language generation*. Springer Science & Business Media.

Verena Rieser, Oliver Lemon, and Xingkun Liu. 2010. Optimising information presentation for spoken dialogue systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1009–1018. ACL.

Deb Roy. 2005. Grounding words in perception and action: Computational insights. *Trends in cognitive sciences*, 9:389–96.

Deb Roy and Ehud Reiter. 2005. Connecting language to the world. *Artificial Intelligence*, 167(1):1 – 12.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 32–37. ACL.

Emanuel Schegloff. 2007. *Sequence Organization in Interaction: A Primer in Conversation Analysis*, volume 1. Cambridge University Press.

Emanuel Schegloff, Gail Jefferson, and Harvey Sacks. 1977. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382.

David Schlangen and Gabriel Skantze. 2009. A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 710–718. ACL.

Ethan O. Selfridge, Iker Arizmendi, Peter A. Heeman, and Jason D. Williams. 2012. Integrating incremental speech recognition and POMDP-based dialogue systems. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 275–279, Seoul, South Korea. ACL.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth Conference on Artificial Intelligence*, page 3776–3783. AAAI Press.

Murray Shanahan. 2016. The Frame Problem. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, spring 2016 edition. Metaphysics Research Lab, Stanford University.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2019. Beyond task success: A closer look at jointly learning to see, ask, and GuessWhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2578–2587. ACL.

Todd Shore, Theofronia Androulakaki, and Gabriel Skantze. 2018. KTH tangrams: A dataset for research on alignment and conceptual pacts in task-oriented dialogue. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can you put it all together: Evaluating conversational agents' ability to blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030. ACL.

Svetlana Stoyanchev, Alex Liu, and Julia Hirschberg. 2013. Modelling human clarification strategies. In *Proceedings of the SIGDIAL 2013 Conference*, pages 137–141, Metz, France. ACL.

Ryuichi Takanobu, Qi Zhu, Jinchao Li, Baolin Peng, Jianfeng Gao, and Minlie Huang. 2020. Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 297–310. ACL.

Xiang Zhi Tan, Sean Andrist, Dan Bohus, and Eric Horvitz. 2020. Now, over here: Leveraging extended attentional capabilities in human-robot interaction. In *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, page 468–470, New York, NY, USA. ACM.

Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. 2005. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*, 28(5):675–691.

David Traum. 1991. *A Computational Theory of Grounding in Natural Language Conversation*. Ph.D. thesis, University of Rochester.

David Traum and Staffan Larsson. 2003. The information state approach to dialogue management. In Jan van Kuppevelt and Ronnie Smith, editors, *Current and New Directions in Discourse and Dialogue*, pages 325–353. Kluwer.

Stefan Ultes, Paweł Budzianowski, Iñigo Casanueva, Lina M. Rojas-Barahona, Bo-Hsiang Tseng, Yen-Chen Wu, Steve Young, and Milica Gašić. 2018. Addressing objects and their relations: The conversational entity dialogue model. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 273–283. ACL.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 673–683. ACL.

Martín Villalba, Christoph Teichmann, and Alexander Koller. 2017. Generating contrastive referring expressions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 678–687, Vancouver, Canada. ACL.

Harm de Vries, Dzmitry Bahdanau, and Christopher D. Manning. 2020. Towards ecologically valid research on language user interfaces. *arXiv*, abs/2007.14435.

Harm de Vries, Kurt Shuster, Dhruv Batra, Devi Parikh, Jason Weston, and Douwe Kiela. 2018. Talk the walk: Navigating new york city through grounded dialogue. *CoRR*, abs/1807.03367.

Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. 2017. Guesswhat?! visual object discovery through multi-modal dialogue. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4466–4475. IEEE Computer Society.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280, Madrid, Spain. Association for Computational Linguistics.

Luca Weihs, Jordi Salvador, Klemen Kotar, Unnat Jain, Kuo-Hao Zeng, Roozbeh Mottaghi, and Aniruddha Kembhavi. 2020. Allenact: A framework for embodied ai research. *arXiv*.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3:1–191.

Ludwig Wittgenstein. 1953. *Philosophical Investigations*. Basil Blackwell.

A More examples from the test scenarios

In this section we present more examples that illustrate the basic points made in the test scenarios presented in Section 4 of the paper.

A.1 Dungeon Intelligence

This is a continuation of the dialog with the Dungeon AI (DAI) game (mode fantasy, character wizard) presented in Section 4. The example in the main paper illustrated numeric inconsistency. In this extended example we illustrate also gender inconsistency (the clerk is first a girl and then a man) and then causal inconsistency (after the food is brought the clerk says they do not serve food, but finally food is available).

DAI: You are wearing nondescript clothing, and you manage to escape without being recognized. You open the purse and find seven gold coins, which is good since you only planned to steal a little.
Me: How many coins are there in the purse?
DAI: You count the coins and find that you have eighty one coins, which is good since you planned to steal just a little. You think to yourself "I should be able to afford a room tonight."
Me: but before you only had seven, how many are there really?
DAI: You head to the first inn you see, which seems to be a place where merchants stay. The desk clerk is a young woman who is probably only just thirteen or fourteen.
Me: I count the coins again
DAI: You count the coins and there are exactly 32 coins there. Hmm, makes it easy to know how much to give to beggars.
Me: I am hungry, I buy a burger.
DAI: You walk over to the counter and order a Whopper with cheese.
Me: I look at your burger.
DAI: It looks like meat. "A burger of goat meat and a mug of ale please." The man nods and quickly brings you the food. "Thanks."
Me: I am surprised, I thought the clerk was a girl.
DAI: "Sorry kid, I don't serve food."
Me: I am confused, I ask "Do you serve food or not?"
DAI: "Sure, if you have the coin." He gives you the food. The food is delicious. You haven't eaten anything so good in a while.

As mentioned in Section 4, it is unsurprising that language models trained only on form are unable to perform collaborative grounding. So in the next subsection we further explore a widespread dataset (de Vries et al., 2017) to look for evidence of collaborative grounding in a multimodal task.

A.2 Visual dialog

Figure 3a, 3b and 3c are random dialogs extracted for the Guesswhat?! dataset (de Vries et al., 2017). In all dialogs, the meaning of the question is correctly interpretable without the previous dialog. In Figure 3a it seems that question 7 *On the left side?* is dependent of the previous turn, and means *On the left side of the boy in the backwards baseball cap?*. However it can be answered correctly with an absolute interpretation without context such as *On the left side of the picture?*. Similarly, in Figure 3b it seems that question 7 *Is it touching the right edge?* should be interpreted as *Is the carrot touching the right edge?*. However, it can correctly answered in a context independent fashion interpreting it as *Is it something touching the right edge?*. Finally, in Figure 3c even the elliptical question 6 *Partially visible?* can be answered correctly without considering the previous turns of the dialog.

The ellipsis can be resolved by changing the question to *It is the one partially visible?*. Even though the "partially visible" criterion hold of more than one potential referent, it is only necessary to know it is true for the target to answer it correctly.

From the Oracle perspective, the whole interaction can be solved without access to the dialog

Question	Answer
1. Is it alive?	yes
2. Is it the baseball player?	no
3. Is it a fan?	yes
4. Is he standing?	no
5. In he wearing a backwards baseball cap?	no
6. Next to the boy in the backwards baseball cap?	yes
7. On the left side?	yes

(a) Question 7 can be interpreted with respect to question 6 as *"on the left side of the boy with the backwards baseball cup"*.

Question	Answer
1. Is it the plate?	no
2. Is it food?	yes
3. Is it a little piece of food?	yes
4. Is it carrot?	yes
5. Is it on the left side?	no
6. Is it on the far right?	yes
7. Is it touching the right edge?	yes

(b) Question 7 could be enriched to *"is it the little piece of carrot touching the right edge"*, considering to questions 3 and 4.

Question	Answer
1. Is it a person?	yes
2. Are they wearing blue?	yes
3. Are they one of the three in the front?	yes
4. Is it a man?	no
5. A woman wearing shorts?	yes
6. Partially visible?	yes

(c) Question 6 could be enriched by the previous dialog as *"is it a partially visible woman in blue wearing shorts"*.

Figure 3: Sample image and dialog from the GuessWhat?! dataset (de Vries et al., 2017). The green mask marks the dialog referent. Questions seem dialog dependent but they can be answered correctly in isolation.

history. There is no need for the Oracle to engage in collaborative grounding for these dialogs.

A.3 Graphical User Interfaces

In this section we illustrate another parallelism between a bad graphical user interface (GUI) and a bad dialog system. A dialog systems that does not expose its limitations and acts as if it was able to detect more intentions that it can actually do, may be compared to a GUI that offers too many options as illustrated in Figure 4.

Both of these interfaces, the graphical and the conversational, overwhelm the user with options, many of which may turn out to be useless. Moreover, a GUI that uses icons that are unfamiliar to the target user, or are hard to recognize, suffers from the same problems as a conversational system trained with a different language variety or expertise than that of the user. This last problem can be overcome through collaborative grounding. In particular, if users need to use this GUI, they can ground the meaning of the strange icons through interaction — all the needed information will exposed by the GUI. We illustrate both points in Figure 4, a GUI cluttered with strange icons.

B Examples of collaborative grounding

Building datasets is expensive. In this section we briefly describe dialog datasets already built that include collaborative grounding phenomena. An

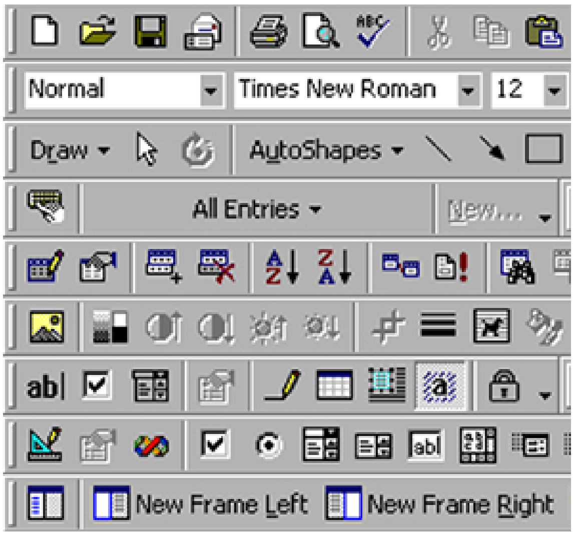


Figure 4: A cluttered GUI that contains icons which are not meaningful for the user

interesting direction of future research would be to analyze and quantify the collaborative grounding acts in these 2 datasets in the spirit of (Fernandez, 2006; Fernández et al., 2007) and other work reviewed in Section 6 of the paper.

The following is an example from (Clark and Wilkes-Gibbs, 1986) with a trial noun phrase collaboratively grounded between S and J. This dataset motivated the creation of larger datasets on the same task (Shore et al., 2018).

S: The small blue cap we talked about before?
J: The one with yellow dots?
S: Yeah

The following example from the Photobook dataset (Haber et al., 2019) starts with a trial noun phrase from A who referred to a TV as a computer, which is collaboratively grounded between A and B. The authors propose to integrate grounding acts to a model based on reference chains.

A: Man with dog on lap looking at his computer?
B: I don't have that, but could it be a TV in yours? Mine has a man sitting with his dog watching TV.
A: yes, TV - sorry!
B: Okay.

This example shows two dialog participants collaboratively grounding their position in a map in the MeetUp corpus (Ilinykh et al., 2019). The map contains pictures of the different rooms the participants can be in. They coordinate the position by describing the rooms. The task is designed to be symmetric, so both participants can contribute equally. Other datasets with a symmetric task are Mutual Friends (He et al., 2017) and Light (Urbanek et al., 2019) and others (Cho and May, 2020).

B: I think I am in a basement
B: i think i might be too
A: maybe not though
A: wood panel?
A: two doors?
B: there's a tan couch, and a tan loveseat/chair brown coffee table. bar. tv
B: nope, different room
A: ok i am not there

The following dialog fragment was collected between crowdsourcers acting as a tourist and a guide (de Vries et al., 2018). By looking at a map the guide had to accomplish the goal of guiding the tourist to a given location in a city. The tourist had access to a navigation street view and provided feedback to the guide about what he did and saw. The task is designed to be asymmetric so collaboration is limited to the roles each participant plays. The guide can make mistakes because the map information is incomplete. The tourist takes a more active role when the guide makes a mistake as illustrated below.

Guide: Ok, turn left then go straight up that road
Guide: There should be shops on two of the corners but you need to go to the corner without a shop.
Tourist: on my left is Radio city Music hall
Tourist: I can't go straight any further.
Guide: ok. turn so that the theater is on your right.
Guide: then go straight
Tourist: That would be going back the way I came
Guide: yeah. I was looking at the wrong bank

Other datasets collected for asymmetric tasks are (Eric et al., 2017; Kottur et al., 2019; Alamri et al., 2019; Kim et al., 2019; Narayan-Chen et al., 2019; Kontogiorgos et al., 2020). All these datasets show collaborative grounding phenomena and are promising contributions for the development of models that can learn to collaboratively ground meaning.

Does He Wink or Does He Nod? A Challenging Benchmark for Evaluating Word Understanding of Language Models

Lutfi Kerem Senel and **Hinrich Schütze**
Center for Information and Language Processing (CIS), LMU Munich, Germany
`lksenel@gmail.com`, `inquiries@cislmu.org`

Abstract

Recent progress in pretraining language models on large corpora has resulted in large performance gains on many NLP tasks. These large models acquire linguistic knowledge during pretraining, which helps to improve performance on downstream tasks via fine-tuning. To assess what kind of knowledge is acquired, language models are commonly probed by querying them with 'fill in the blank' style cloze questions. Existing probing datasets mainly focus on knowledge about *relations between* words and entities. We introduce WDLMPro (Word Definition Language Model Probing) to *evaluate word understanding directly* using dictionary definitions of words. In our experiments, three popular pretrained language models struggle to match words and their definitions. This indicates that they understand many words poorly and that our new probing task is a difficult challenge that could help guide research on LMs in the future.

1 Introduction

Natural language processing (NLP) has advanced drastically in the last decade with the design of larger and more sophisticated models, availability of larger corpora and increasing computational power. Pretrained word embeddings (Mikolov et al., 2013; Pennington et al., 2014) popularized the use of distributed word representations, which became a fundamental building block for NLP systems. Peters et al. (2018a) introduced LSTM-based deep contextual representations and obtained large performance gains by fine-tuning on tasks after unsupervised pretraining (Radford et al., 2018; Howard and Ruder, 2018). More recently, the attention based transformer architecture was shown to use context more effectively (Vaswani et al., 2017) and several subsequent models achieved state of the art results in many NLP tasks by combining the transformer architecture with unsupervised pretraining and task specific fine-tuning (Devlin et al., 2019; Liu et al., 2019). Radford et al. (2019) showed that language models can be applied to a variety of tasks without task specific fine tuning. This is demonstrated on a much larger scale by Brown et al. (2020).

Deep models improve performance. However, what they actually learn about language and word meaning is still to a large extent unclear due to their uninterpretable nature. For static word embeddings, researchers used word similarity (Hill et al., 2015) and word analogy (Gladkova et al., 2016) tests to shed light on what information is captured in these dense vector spaces. For language models, a great amount of linguistic knowledge is stored in the model parameters (Peters et al., 2018b). Several studies proposed using 'fill in the blank' type cloze statements to test knowledge learned by these models during unsupervised pretraining. Petroni et al. (2019) proposed the LAMA (LAnguage Model Analysis) probe to test the factual and common sense knowledge stored in language models. Similarly, Schick and Schütze (2020) introduced WNLaMPro (WordNet Language Model Probing) to assess the ability of language models to understand words based on their frequency. In WNLaMPro, cloze style questions are generated based on antonym, hypernym and cohyponym relations among words extracted from WordNet.

The existing probing datasets mainly focus on investigating the knowledge about *relations between* words or entities. However, a more direct way of testing whether a language model understands the meaning of a word is to use its dictionary definition. If a pretrained language model truly understands the meaning of a word, then it should be able to match it with its dictionary definition. Based on this motivation, we introduce the *Word Definition*

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 532–538
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

synset	definition
a_cappella_singing.n.01	*singing without instrumental accompaniment*
caroling.n.01	singing joyful religious songs (especially at Christmas)
crooning.n.01	singing in a soft low tone
singalong.n.01	informal group singing of popular songs
bel_canto.n.01	a style of operatic singing

Table 1: Five candidates from $\mathcal{G}(t)$ for t= *a_cappella_singing.n.01* and their definitions

	Noun	Verb
# of Synset Groups	51260	8487
Average # of Candidates	50.2	47.7
min / max # of Candidates	5 / 404	5 / 593

Table 2: WDLMPro statistics

Language Model Probing (WDLMPro) dataset;[1] it is a challenging benchmark for testing NLP models for their ability to understand words. WDLMPro is essentially a set of thousands of synset groups; each synset group consists of a target word (with its definition) and its taxonomic sisters (with their definitions). Using taxonomic sisters, rather than random word groups, makes the task more challenging for statistical models that are based on the distributional hypothesis since these words have similar distributional characteristics (Lenci, 2008). We evaluate two masked language models, BERT and RoBERTa, and the auto-regressive model GPT-2 on WDLMPro using two different probing tests: (i) match definition to word (D2W) (ii) match word to definition (W2D). We also provide a baseline using static fastText embeddings (Mikolov et al., 2018). We find that all three language models perform clearly better than the baseline. Nevertheless, they have great difficulty matching words and their definitions, implying a poor understanding of word meaning. This is an important result that could help guide research on LMs in the future.

2 WDLMPro

In this section, we introduce WDLMPro (Word Definition Language Model Probing), a dataset to test how well NLP models can match nouns and verbs with their definitions. We view this as a test of how well the models understand lexical meaning.

2.1 Dataset

WordNet (Miller, 1995) is the basis for constructing WDLMPro. A WordNet *synset* contains a set of synonyms along with a short definition of the synset. Different senses of polysemous words are represented in different synsets providing disambiguation. WordNet connects synsets with each other via semantic relations.

Based on a *target synset* t and the semantic relation hyponymy $<$, we construct a *synset group* $\mathcal{G}$ for the target as follows.

$$\mathcal{G}(t) = \{x | \exists y : t < y \land x < y\}$$

that is, G contains all synsets that are “sister hyponyms” to t with respect to a hypernym of t. $\mathcal{G}(t)$, along with the definitions of the synsets in $\mathcal{G}(t)$, will be used to set up the WDLMPro tasks that require matching of words and definitions. We discard groups $\mathcal{G}(t)$ that have a size of less than 5.

In this study, we focus on nouns and verbs, i.e., we create synset groups $\mathcal{G}$ for the nouns and verbs in WordNet. Table 1 displays five members from $\mathcal{G}(t)$ and their definitions for the target *a_cappella_singing.n.01* (see appx. for the target *beckon.v.01*.) Table 2 shows statistics of the dataset.

2.2 Probing Tests

We define two probing tests that are converses of each other:

- **Match definition to word (D2W).** Given a definition and a set of words, the task is to find the word that the definition defines.
- **Match word to definition (W2D).** Given a word and a set of definitions, the task is to find the definition that defines the word.

Each synset group $\mathcal{G}(t)$ gives rise to one instance of D2W by providing the definition of t, and all words in $\mathcal{G}(t)$. The word from $\mathcal{G}(t)$ that matches the definition has then to be identified. (Note that t is a member of $\mathcal{G}(t)$.) Similarly, each synset group $\mathcal{G}(t)$ gives rise to one instance of W2D by providing t and the definitions of all words in $\mathcal{G}(t)$. The correct definition of t has then to be identified among all definition candidates. Note that WordNet definitions by construction do not contain the word

---

[1] WDLAMPro and evaluation scripts are available at https://www.cis.lmu.de/definition_benchmark/WDLAMPro.zip

	Masked Language Model (MLM)
Noun	_ is `<DEF>`
	_ means `<DEF>`
	_ is defined as `<DEF>`
Verb	definition of _ is to `<DEF>`
	to `<DEF>` is the definition of _
	Autoregressive Language Model (ALM)
Noun	`<DEF>` is the definition of _
Verb	to `<DEF>` is the definition of _

Table 3: Patterns used for querying language models for nouns and verbs. `<DEF>` refers to the definition, _ is the mask or missing word that the language model has to predict.

to be defined. So there are no instances where the two tasks are trivial.

2.2.1 Application to language models

In principle, any NLP model can be tested on D2W and W2D. In this paper, we are particularly interested in testing language models. To this end, we convert the data to a format that is suitable for language models, i.e., to cloze-style questions as shown in Table 3. The basic quantity that allows us to assess the compatibility of a word t and a definition is the probability of t being generated for "_" when the definition is substituted for <DEF>.

More precisely, we compute the probability that the string representation of t is being generated. We will denote the string representation of synset t by $\boldsymbol{t}$. We obtain the string representation by removing the word type and sense information from the name of the synset and replacing underscores with white space. For example, synset *warm_up.v.04* is represented by the string "warm up".

Table 3 shows that we define different templates for masked and autoregressive language models. For the masked language models, we average the prediction scores across patterns before ranking the candidates.

2.3 Baselines

For a masked language model (MLM) M, the probability of a candidate $c \in \mathcal{G}(t)$ on W2D is calculated as:

$$P_M^{\text{W2D}}(c|t) = \prod_{i=1}^{|\boldsymbol{t}|} P(\boldsymbol{t}^i | Q(c, |\boldsymbol{t}|))$$

where $\boldsymbol{t} = [\boldsymbol{t}^1, \boldsymbol{t}^2, ..., \boldsymbol{t}^{|\boldsymbol{t}|}]$ is the tokenization produced by M. $Q(c, |\boldsymbol{t}|)$ is the input query created from one of the patterns (Table 3) with _ replaced with $|\boldsymbol{t}|$ consecutive mask tokens. For an autoregressive language model (ALM) A, we decompose $P(\boldsymbol{t}^i | Q(c), \boldsymbol{t})$ in the standard way:

$$P_A^{\text{W2D}} = \prod_{i=1}^{|\boldsymbol{t}|} P(\boldsymbol{t}^i | Q(c), \boldsymbol{t}^1, ..., \boldsymbol{t}^{i-1})$$

For D2W, we need to compare, given a definition, the probabilities of different candidate words that are generally of different lengths. To ensure a fair comparison, we follow Xiong et al. (2020). For MLMs, we match the number of mask tokens in an input query to the token count of each candidate. The final score is the average log-probability of the masked tokens:

$$P_M^{\text{D2W}}(c|t) = \frac{1}{|\boldsymbol{c}|} \sum_{i=1}^{|\boldsymbol{c}|} \log P(\boldsymbol{c}^i | Q(t, |\boldsymbol{c}|))$$

For ALMs, we use the probability of the first token:

$$P_A^{\text{D2W}}(c|t) = P(\boldsymbol{c}^1 | Q(t))$$

Considering further tokens does not make sense since they are often easily predictable from the first token.

We apply our probing test to two different pretrained MLMs (BERT and RoBERTa) and one ALM (GPT-2). To investigate the effect of model size on the performance, we experiment with both base and large versions of BERT and RoBERTa along with all four sizes of GPT-2 (small, medium, large, xl). For RoBERTa, we capitalize the first letter of the candidate noun since pretrained RoBERTa models are case sensitive and expect a capital letter at the beginning of a sentence.[2]

In addition to the deep contextual language models, we also provide fastText static word embeddings[3] (Mikolov et al., 2018) as a baseline.[4] For fastText embeddings, we tokenize the candidates and their definitions using the NLTK tokenizer and represent them with their average vector. We rank candidates based on their cosine similarity to the target embedding.

---

[2] Not using capitalization resulted in poor performance for single token target words for D2W.

[3] We use the crawl-300d-2M-subword model from https://fasttext.cc/docs/en/english-vectors.html

[4] A reviewer suggests that it would also be interesting to investigate the performance of supervised approaches, e.g., ranking models. Our main focus here is the lexical knowledge acquired in pretraining, so we leave this for future work.

2.4 Measures

We use two measures: precision at 1 (P@1) and a rank score (RS), both based on a ranked results list, either of words or of definitions. P@1 is the percentage of top-ranked items that is correct. We define RS as follows:

$$\mathrm{RS}(L, k) = \frac{L - k}{L - 1}$$

where $L = |\mathcal{G}(t)|$ is the number of candidates and k is the rank of the correct item, $1 \leq k \leq L$. Table 2 shows that the size of $\mathcal{G}(t)$ is highly variable; in contrast to P@1, RS is less affected by this and the random baseline (cf. Tables 4 and 5) is always 0.5.

3 Results

Tables 4 and 5 present W2D and D2W results for BERT, RoBERTa and GPT-2 along with fastText and random baselines. Language models perform clearly better than both baselines. Larger models perform generally better than smaller ones and RoBERTa consistently outperforms BERT. This might be an indication for the correlation between performance on WDLAMPro and downstream performance. However, further investigation is necessary to show the correlation more clearly. For W2D, best performance is achieved by GPT-2$_{xl}$ for nouns (47.3 P@1, 0.81 RS) and by RoBERTa large for verbs (50.8 P@1, 0.84 RS). Performance on D2W is much lower than for W2D for all models. For nouns, RoBERTa large and GPT-2$_{xl}$ perform similarly (28.8 and 29.8 P@1, 0.70 and 0.73 RS) while RoBERTa large achieves the best results for verbs (38.6 P@1, 0.80 RS). Poor performance on D2W compared to W2D might be due to language models' ability to distinguish different definitions better than individual words since definitions are more informative than individual words. Overall GPT-2 models perform better than masked language models (with the exception of Roberta large for verbs), despite using a single pattern as opposed to the multiple patterns used by masked language models. This might indicate that the ALM objective is better at learning word meaning than the MLM objective.

To investigate the effect of frequency, we stratify words into *rare* (fewer than 10 occurrences), *medium* (10 to 99 occurrences) and *frequent* (100 or more occurrences), based on occurrences in WWC[5] (Westbury Wikipedia Corpus, Shaoul (2010)), where we use WWC frequency as a substitute for the models' training corpora. We focus on nouns since most verbs in our dataset are relatively frequent. Table 7 shows that, for W2D, all models have a poor understanding of the meaning of rare and medium words. (See appx. for D2W results.) Even for frequent words, P@1 is never above 55.

Model	**Noun**		**Verb**	
	P@1	RS	P@1	RS
Bert$_b$	35.2	0.74	35.3	0.74
Bert$_l$	35.1	0.73	33.6	0.73
Roberta$_b$	37.1	0.75	42.7	0.79
Roberta$_l$	42.1	0.78	50.8	0.84
GPT-2$_s$	38.7	0.76	45.0	0.80
GPT-2$_m$	41.8	0.77	43.6	0.80
GPT-2$_l$	45.7	0.80	48.4	0.83
GPT-2$_{xl}$	47.3	0.81	48.6	0.83
fastText	22.5	0.66	29.1	0.69
Random	7.6	0.50	7.8	0.50

Table 4: P@1 and rank score (RS) on W2D

Model	**Noun**		**Verb**	
	P@1	RS	P@1	RS
Bert$_b$	23.7	0.65	19.3	0.65
Bert$_l$	25.4	0.65	19.3	0.65
Roberta$_b$	25.7	0.67	32.6	0.74
Roberta$_l$	28.8	0.70	38.6	0.80
GPT-2$_s$	23.2	0.68	29.2	0.71
GPT-2$_m$	25.3	0.70	27.8	0.72
GPT-2$_l$	28.4	0.72	31.5	0.74
GPT-2$_{xl}$	29.8	0.73	32.8	0.76
fastText	16.5	0.63	20.3	0.69
Random	7.6	0.50	8.0	0.50

Table 5: P@1 and rank score (RS) on D2W

We additionally break down the results based on the depth of the synsets in the WordNet hierarchy. Specifically, we investigate the performance of the GPT-2$_{xl}$ model on W2D for WordNet nouns, where we take the depth of a synset group as the length of the shortest path from the target synset to the root synset (i.e., *entity.n.01*). Table 6 shows that performance drops steadily as we go deeper in the hierarchy. Lower levels of the WordNet hierarchy contain many scientific terms and names of (sub)species such as types of cattle (e.g., *cattalo*, *hereford*, *galloway*). These results suggest that even very large LMs lack the knowledge necessary to distinguish these terms.

---

[5] Targets that have more than 3 tokens (based on NLTK tokenization) are taken as rare without counting.

Depth	# synsets	# cand.	RS	P@1
3–5	2106	110	0.94	62.9
6–8	25,232	53	0.83	49.0
9–11	18,521	45	0.81	46.6
12–14	4473	19	0.74	37.4
15–19	928	13	0.67	31.5

Table 6: RS and P@1 results for GPT-2$_{xl}$ on W2D for nouns from different depths of the WordNet hierarchy. # of candidates, RS and P@1 are given as the average across all synsets within the given depth range.

Model	rare	medium	frequent	all
Bert$_b$	26.0	31.1	40.7	35.2
Bert$_l$	23.6	29.8	42.0	35.1
Roberta$_b$	30.8	34.7	40.7	37.1
Roberta$_l$	33.2	38.7	47.2	42.1
GPT-2$_s$	32.9	35.2	42.6	38.7
GPT-2$_m$	34.4	37.4	46.7	41.8
GPT-2$_l$	37.0	41.4	51.1	45.7
GPT-2$_{xl}$	37.7	42.7	53.3	47.3
Random	6.6	7.0	8.2	7.6

Table 7: P@1 scores on W2D for nouns of different frequency ranges

Analysis. The correct definition of the medium frequency verb ‘beckon’ is ‘signal with the hands or nod’. GPT-2$_{xl}$ predicts ‘signal by winking’. The correct definition of the frequent noun ‘roleplaying’ is ‘acting a particular role (as in psychotherapy)’ GPT-2$_{xl}$ predicts ‘acting the part of a character on stage’. So GPT-2$_{xl}$ understands that beckoning is signaling and that roleplaying is acting, but it has not learned to distinguish between different types of signaling and acting. This points to an important future goal for LMs: they should be developed to gain an understanding of words that goes beyond the current superficial state of the art.

Human performance on WDLAMPro. It is beyond the scope of this paper to evaluate human performance on the entirety of WDLAMPro. However, we provide a comparison with human performance on a small subset to provide an intuition about the difficulty of the task. For each of the two tasks, 20 synset groups that have a maximum of 10 candidates are randomly sampled from WDLAMPro. Then two native English speakers are asked to rank the candidates. Table 8 displays the average performance of the human participants and the language models on this subset. For both tasks, performance of the best model is comparable to the average human performance.

Model	W2D		D2W	
	P@1	RS	P@1	RS
Bert$_b$	60.0	0.84	35.0	0.64
Bert$_l$	65.0	0.74	35.0	0.69
Roberta$_b$	50.0	0.78	60.0	0.81
Roberta$_l$	55.0	0.80	45.0	0.69
GPT-2$_s$	35.0	0.69	45.0	0.71
GPT-2$_m$	50.0	0.80	50.0	0.73
GPT-2$_l$	60.0	0.84	45.0	0.75
GPT-2$_{xl}$	50.0	0.76	45.0	0.79
Human	62.5	0.88	57.5	0.77

Table 8: LM and human performance on 20 random samples of WDLAMPro.

Human performance is the upper bound for many NLP tasks. We believe that this is not the case for WDLAMPro: arguably, we should aim for models with an excellent understanding of the meanings of words even if it is better than average human understanding. Knowledge based tasks are an analogous case: we should strive for models that know as many facts as possible even if that performance is above average human performance.

4 Conclusion

We introduced WDLMPro, a probing test that helps analyze how well a model understands word meaning. WDLMPro is complementary to existing probing tests that are about *relations* between words or entities. We evaluated three popular pretrained language models on the W2D (word to definition) and D2W (definition to word) tasks. Our findings show that, despite their remarkable performance on many downstream tasks, these models struggle to match a word and its true definition, suggesting an insufficient understanding of word meaning. Relatively poor performance of these powerful models on WDLMPro can be seen as evidence for the limitations of purely distributional systems and the need for incorporating external knowledge. WDLMPro provides an important evaluation benchmark, encouraging design and training of models with precise word understanding.

Acknowledgements. We thank Denis Peskov and Sander Schulhoff for helping out with the human evaluation and the anonymous reviewers for their insightful comments and suggestions. This work was funded by the European Research Council (ERC #740516).

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

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

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proceedings of the NAACL Student Research Workshop*, pages 8–15, San Diego, California. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Alessandro Lenci. 2008. Distributional semantics in linguistic and cognitive research. *Italian journal of linguistics*, 20(1):1–31.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proc. of the International Conference on Learning Representations (ICLR)*, pages 1–12.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

George A Miller. 1995. WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *Technical Report*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8766–8774.

C. & Westbury C. Shaoul. 2010. The westbury lab wikipedia corpus.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *International Conference on Learning Representations*.

A Appendix

synset	**definition**
beckon.v.01	*signal with the hands or nod*
applaud.v.01	clap one's hands or shout after performances to indicate approval
bow.v.01	bend one's knee or body, or lower one's head
shrug.v.01	raise one's shoulders to indicate indifference or resignation
exsert.v.01	thrust or extend out
wink.v.01	signal by winking
nod.v.01	express or signify by nodding

Table 9: Seven candidates of $\mathcal{G}(t)$ for t= *beckon.v.01* and their definitions

Model	**rare**	**medium**	**frequent**	**all**
Bert$_b$	14.7	20.6	28.7	23.7
Bert$_l$	12.0	20.1	33.1	25.4
Roberta$_b$	17.7	24.2	29.5	25.7
Roberta$_l$	17.9	25.8	34.5	28.8
GPT-2$_s$	17.3	20.7	26.7	23.2
GPT-2$_m$	17.0	21.1	30.6	25.3
GPT-2$_l$	19.2	24.3	33.9	28.4
GPT-2$_{xl}$	19.3	24.8	36.3	29.8
Random	6.7	7.1	8.3	7.6

Table 10: P@1 scores on D2W for nouns based on target word frequency.

Joint Coreference Resolution and Character Linking for Multiparty Conversation

Jiaxin Bai[1], **Hongming Zhang**[1], **Yangqiu Song**[1], **and Kun Xu**[2]
[1]CSE, HKUST
[2] Tencent AI Lab
`{jbai, hzhangal, yqsong}@cse.ust.hk, kxkunxu@tencent.com`

Abstract

Character linking, the task of linking mentioned people in conversations to the real world, is crucial for understanding the conversations. For the efficiency of communication, humans often choose to use pronouns (e.g., "she") or normal phrases (e.g., "that girl") rather than named entities (e.g., "Rachel") in the spoken language, which makes linking those mentions to real people a much more challenging than a regular entity linking task. To address this challenge, we propose to incorporate the richer context from the coreference relations among different mentions to help the linking. On the other hand, considering that finding coreference clusters itself is not a trivial task and could benefit from the global character information, we propose to jointly solve these two tasks. Specifically, we propose C^2, the joint learning model of **C**oreference resolution and **C**haracter linking. The experimental results demonstrate that C^2 can significantly outperform previous works on both tasks. Further analyses are conducted to analyze the contribution of all modules in the proposed model and the effect of all hyper-parameters.

1 Introduction

Understanding conversations has long been one of the ultimate goals of the natural language processing community, and a critical step towards that is grounding all mentioned people to the real world. If we can achieve that, we can leverage our knowledge about these people (e.g., things that happened to them before) to better understand the conversation. On the other hand, we can also aggregate the conversation information back to our understanding about these people, which can be used for understanding future conversations that involve the same people. To simulate the real conversations and investigate the possibility for models to ground mentioned people, the character linking task was proposed (Chen and Choi, 2016). Specifically, it uses the transcripts of TV shows (i.e., *Friends*) as the conversations and asks the models to ground all person mentions to characters.

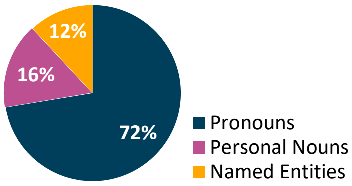
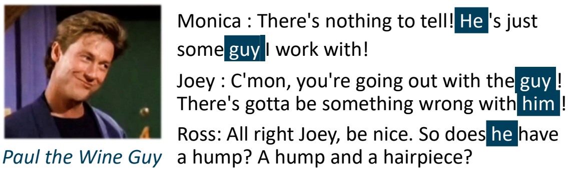


Figure 1: The composition of the mentions in conversations for character grounding. Over 88% of the mentions are not named entities, which brings exceptional challenges when linking those to character entities.

Even though the character linking task can be viewed as a special case of the entity linking task, it is more challenging than the ordinary entity linking task for various reasons. First, the ordinary entity linking task often aims at linking named entities to external knowledge bases such as Wikipedia, where rich information (e.g., definitions) are available. However, for the character linking task, we do not have the support of such rich knowledge base and all we have are the names of these characters and simple properties (e.g., gender) about these characters. Second, the mentions in the ordinary entity linking are mostly concepts and entities, but not pronouns. However, as shown in Figure 1, 88% of the character mentions are pronouns (e.g., "he") or personal nouns (e.g., "that guy") while only 12% are named entities.

Considering that pronouns have relatively weak semantics by themselves, to effectively ground mentions to the correct characters, we need to fully utilize the context information of the whole conversation rather than just the local context they appear in. One potential solution is using the coreference relations among different mentions as the bridge to

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 539–548
April 19 - 23, 2021.



Monica : There's nothing to tell! He 's just some guy I work with!

Joey : C'mon, you're going out with the guy ! There's gotta be something wrong with him !

Ross: All right Joey, be nice. So does he have a hump? A hump and a hairpiece?

Figure 2: Coreference clusters can help to connect the whole conversation to provide a richer context for each mention such that we can better link them to *Paul*. Meanwhile, the character *Pual* can also provide global information to help resolve the coreference.

connect the richer context. One example is shown in Figure 2. It is difficult to directly link the highlighted mentions to the character *Paul* based on their local context because the local context of each mention can only provide a single piece of information about its referent, e.g., "person is a male" or "the person works with Monica." Given the coreference cluster, the mentions refer to the same person, and the pieces of information are put together to jointly determining the referent. As a result, it is easier for a model to do character linking with resolved coreference. Similar observations are also made in (Chen et al., 2017).

At the same time, we also noticed that coreference resolution, especially those involving pronouns, is also not trivial. As shown by the recent literature on the coreference resolution task (Lee et al., 2018; Kantor and Globerson, 2019), the task is still challenging for current models and the key challenge is how to utilize the global information about entities. And that is exactly what the character linking model can provide. For example, in Figure 2, it is difficult for a coreference model to correctly resolve the last mention *he* in the utterance given by *Ross* based on its local context, because another major male character (*Joey*) joins the conversation, which can distract and mislead the coreference model. However, if the model knows the mention *he* links to the character *Paul* and *Paul* works with *Monica*, it is easier to resolve *he* to some *guy* that Monica works with.

Motivated by these observations, we propose to jointly train the **C**oreference resolution and **C**haracter linking tasks and name the joint model as C^2. C^2 adopts a transformer-based text encoder and includes a mention-level self-attention (MLSA) module that enables the model to do mention-level contextualization. Meanwhile, a joint loss function is designed and utilized so that both tasks can be jointly optimized. The experimental results demonstrate that C^2 outperforms all previous work significantly on both tasks. Specifically, compared with the previous work (Zhou and Choi, 2018), C^2 improves the performance by 15% and 26% on the coreference resolution and character linking tasks[1] respectively comparing to the previous state-of-the-art model ACNN (Zhou and Choi, 2018) . Further hyper-parameter and ablation studies testify the effectiveness of different components of C^2 and the effect of all hyper-parameters. Our code is available at `https://github.com/HKUST-KnowComp/C2`.

2 Problem Formulations and Notations

We first introduce the coreference resolution and character linking tasks as well as used notations. Given a conversation, which contains multiple utterances and n character mentions $c_1, c_2, ..., c_n$, and a pre-defined character set $\mathcal{Z}$, which contains m characters $z_1, z_2, ..., z_m$. The coreference resolution task is grouping all mentions to clusters such that all mentions in the same cluster refer to the same character. The character linking task is linking each mention to its corresponding character.

3 Model

In this section, we introduce the proposed C^2 framework, which is illustrated in Figure 3. With the conversation and all mentions as input, we first encode them with a shared mention representation encoder module, which includes a pre-trained transformer text encoder and a mention-level self-attention (MLSA) module. After that, we make predictions for both tasks via two separate modules. In the end, a joint loss function is devised so that the model can be effectively trained on both tasks simultaneously. Details are as follows.

3.1 Mention Representation

We use pre-trained language models (Devlin et al., 2018; Joshi et al., 2019a) to obtain the contextualized representations for mentions. As speaker information is critical for the conversation understanding, we also include that information by appending speaker embeddings to each mention. As

---

[1] The performance on the coreference resolution is evaluated based on the average F1 score of B3, CEAF$_{\phi}$4, and BLANC. The performance on the character linking task is evaluated by the average F1 score of the micro and macro F1.

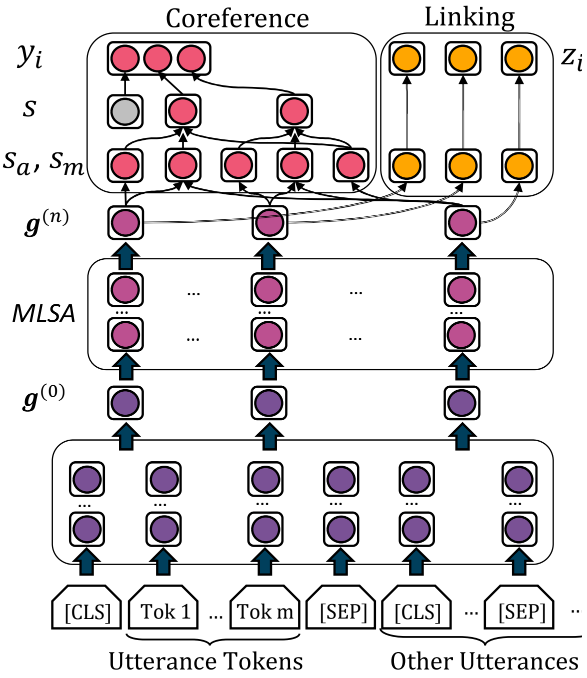


Figure 3: The coreference module and the linking module share the same mention representation $g^{(n)}$ as inputs. The mention representation $g^{(i)}$ are iteratively refined through the mention-level self-attention layers. The initial mention representations $g^{(0)}$ are the sum of text span representations from a pre-trained text encoder and corresponding speaker embeddings.

a result, the initial representation of mention i is:

$$g_i^{(0)} = t_{start_i} + t_{end_i} + e_{speaker_i}, \qquad (1)$$

where t_{start_i} and t_{end_i} are the contextualized representation of the beginning and the end tokens of mention i, and the $e_{speaker_i}$ is the speaker embedding for the current speaker. Here, we omit the embeddings of inner tokens because their semantics has been effectively encoded via the language model. The speaker embeddings are randomly initialized before training.

Sometimes the local context of a mention is not enough to make reasonable predictions, and it is observed that the co-occurred mentions can provide document-level context information. To refine the mention representations given the presence of other mentions in the document, we introduce the Mention-Level Self-Attention (MLSA) layer, which has n layers of transformer encoder structure (Vaswani et al., 2017) and is denoted as T. Formally, this iterative mention refinement process can be described by

$$g_1^{(i+1)}, ..., g_k^{(i+1)} = T(g_1^{(i)}, ..., g_k^{(i)}), \qquad (2)$$

where k indicates the number of mentions in a document, and the $g^{(i)}$ means the mention representation from the i-th layer of MLSA.

3.2 Coreference Resolution

Following the previous work (Joshi et al., 2019a), we model the coreference resolution task as an antecedent finding problem. For each mention, we aim at finding one of the previous mentions that refer to the same person. If no such previous mention exists, it should be linked to the dummy mention ε. Thus the goal of a coreference model is to learn a distribution, $P(y_i)$ over each antecedent for each mention i:

$$P(y_i) = \frac{e^{s(i,y_i)}}{\Sigma_{y' \in \mathcal{Y}(i)} e^{s(i,y')}}, \qquad (3)$$

where $s(i, j)$ is the score for the antecedent assignment of mention i to j. The score $s(i, j)$ contains two parts: (1) the plausibility score of the mentions $s_a(i, j)$; (2) the mention score measuring the plausibility of being a proper mention $s_m(i)$. Formally, the $s(i, j)$ can be expressed by

$$s(i,j) = s_m(i) + s_m(j) + s_a(i,j), \qquad (4)$$

$$s_m(i) = FFNN_m(g_i^{(n)}), \qquad (5)$$

$$s_a(i,j) = FFNN_a([g_i^{(n)}, g_j^{(n)}]), \qquad (6)$$

where $g^{(n)}$ stands for the last layer mention representation resulted from the MLSA and $FFNN$ indicates the feed-forward neural network.

3.3 Character Linking

The character linking is formulated as a multi-class classification problem, following previous work (Zhou and Choi, 2018). Given the mention representations $g^{(n)}$, the linking can be done with a simple feed-forward network, denoted as $FFNN(\cdot)$. Specifically, the probability of character entity z_i is linked with a given mention i can be calculated by:

$$Q(z_i) = Softmax(FFNN_l(g_i^{(n)}))_{z_i}, \qquad (7)$$

where the notation $(.)_z$ represents the z-th composition of a given vector.

3.4 Joint Learning

To jointly optimize both coreference resolution and entity linking, we design a joint loss of both tasks. For coreference resolution, given the gold clusters, we minimize the negative log-likelihood of

DATASET	EPISODES	SCENES	UTTERANCES	SPEAKERS	MENTIONS	ENTITIES
TRN	76	987	18,789	265	36,385	628
DEV	8	122	2,142	48	3,932	102
TST	13	192	3,597	91	7,050	165
TOTAL	97	1,301	24,528	331	47,367	781

Table 1: The detailed information about the datasets. For each season, the episode 1 to 19 are used for training, the episode 20 to 21 for development, and the remaining for testing.

the possibility that each mention is linked to a gold antecedent. Then the coreference loss L_c becomes

$$L_c = -\sum_{i=1}^{N} \log \sum_{y \in \mathcal{Y}(i) \cap GOLD(i)} P(y), \quad (8)$$

where the $GOLD(i)$ denotes the gold coreference cluster that mention i belongs to. Similarly, for character linking, we minimize the negative log-likelihood of the joint probability for each mention being linked to the correct referent character:

$$L_l = -\sum_{i=1}^{N} \log Q(z_i). \quad (9)$$

Finally, the joint loss can be the arithmetic average of the coreference loss and linking loss:

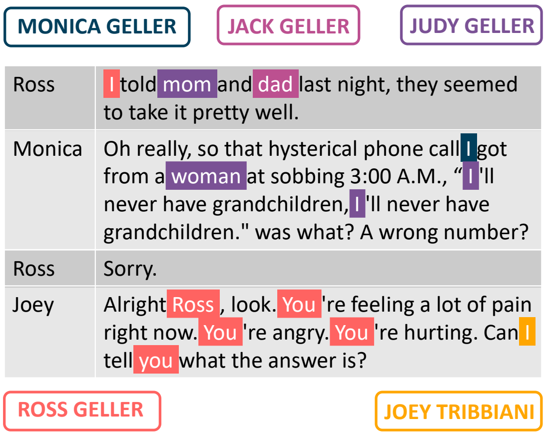
$$L = \frac{1}{2}(L_l + L_c). \quad (10)$$

4 Experiments

In this section, we introduce the experimental details to demonstrate the effectiveness of C^2.

4.1 Data Description

We use the latest released character identification V2.0[2] as the experimental dataset, and we follow the standard training, developing, and testing separation provided by the dataset. In the dataset, all mentions are annotated with their referent global entities. For example, in Figure 4, the mention *I* is assigned to *ROSS*, and the mentions *mom* and *dad* are assigned to *JUDY* and *JACK* respectively in the first utterance given by *Ross*. The gold coreference clusters are derived by grouping the mentions assigned to the same character entity. Statistically, the dataset includes four seasons of the TV show *Friends*, which contain 97 episodes, 1,301 scenes, and 24,528 utterances. In total, there are 47,367 mentions, which are assigned to 781 unique characters. The detailed statistics are shown in Table 1.



Ross	I told mom and dad last night, they seemed to take it pretty well.
Monica	Oh really, so that hysterical phone call I got from a woman at sobbing 3:00 A.M., “ I 'll never have grandchildren, I 'll never have grandchildren." was what? A wrong number?
Ross	Sorry.
Joey	Alright Ross , look. You 're feeling a lot of pain right now. You 're angry. You 're hurting. Can I tell you what the answer is?

Figure 4: The example annotations for character identification. The arrows in the figure are pointing from the character mentions to their referent character entities.

4.2 Baseline Methods

The effectiveness of the joint learning model is evaluated on both the coreference resolution and character linking tasks. To fairly compare with existing models, only the singular mentions are used following the singular-only setting (S-only) in the previous work (Zhou and Choi, 2018).

For the **coreference resolution** task, we compare with the following methods.

- ACNN: A CNN-based model (Zhou and Choi, 2018) coreference resolution model that can also produce the mention and mention-cluster embeddings at the same time.

- C2F: The end-to-end coarse-to-fine coreference model (Joshi et al., 2019b) with BERT (Devlin et al., 2018) or SpanBERT (Joshi et al., 2019a) as the encoder.

- CorefQA: An approach that reformulates the coreference resolution problem as a question answering problem (Wu et al., 2020) and being able to be benefited from fine-tuned question-answer text encoders.

---

[2] https://github.com/emorynlp/character-identification

Model	B3			CEAFϕ4			BLANC			Ave.F1
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	
ACNN	84.30	71.90	77.60	54.50	71.80	62.00	84.30	80.40	82.10	73.96 (0.97)
CorefQA (SpanBERT-Large)	73.72	75.55	74.62	65.82	72.38	68.94	86.82	84.69	85.75	76.44 (0.20)
C2F (BERT-Base)	69.62	76.11	72.72	66.44	60.92	63.56	79.38	86.05	82.38	72.88 (0.23)
C2F (BERT-Large)	71.72	80.25	75.75	69.97	62.61	66.08	81.65	88.23	84.63	75.49 (0.18)
C2F (SpanBERT-Base)	72.49	77.88	75.08	66.00	64.23	65.10	81.60	87.43	84.27	74.81 (0.19)
C2F (SpanBERT-Large)	81.93	84.38	82.57	78.04	71.99	74.89	88.15	91.09	89.56	82.34 (0.17)
C^2 (BERT-Base)	78.10	81.56	79.79	72.48	69.87	71.15	86.14	89.49	87.74	80.14 (0.21)
C^2 (BERT-Large)	78.49	81.90	80.16	73.81	71.15	72.46	86.20	89.93	87.97	80.17 (0.23)
C^2 (SpanBERT-Base)	81.18	83.59	82.36	73.64	73.09	73.36	88.06	91.04	89.49	81.74 (0.19)
C^2 (SpanBERT-Large)	**85.83**	**85.27**	**85.55**	**77.13**	**77.84**	**77.48**	**92.31**	**92.03**	**92.17**	**85.06** (0.16)

Table 2: Experimental results on the coreference resolution task. The results are presented in a 2-digit decimal following previous work. Standard deviations of the average F1 scores are shown in brackets.

For the **character linking** task, we also include ACNN as a baseline method. Considering existing general entity linking models (Kolitsas et al., 2018; van Hulst et al., 2020; Raiman and Raiman, 2018; Onando Mulang et al., 2020) cannot be applied to the character linking problem because they are not designed to handle pronouns, we propose another text-span classification model with transformer encoder as another strong baseline for the character linking task.

- ACNN: A model that uses the mention and mention-cluster embeddings as input to do character linking (Zhou and Choi, 2018).
- BERT/SpanBERT: A text-span classification model consists of a transformer text encoder followed by a feed-forward network.

4.3 Evaluation Metrics

We follow the previous work (Zhou and Choi, 2018) for the evaluation metrics. Specifically, for coreference resolution, three evaluation metrics, B^3, $CEAF_{\phi 4}$, and BLANC, are used. The metrics are all proposed by the CoNNL'12 shared task (Pradhan et al., 2012) to evaluate the output coreference cluster against the gold clusters. We follow Zhou and Choi (2018) to use BLANC (Recasens and Hovy, 2011) to replace MUC (Vilain et al., 1995) because BLANC takes singletons into consideration but MUC does not. As for the character linking task, we use the Micro and Macro F1 scores to evaluate the multi-class classification performance.

4.4 Implementation Details

In our experiments, we consider four different pretrained language encoders: BERT-Base, BERT-Large, SpanBERT-Base, and SpanBERT-Large, and we use $n = 2$ layers of the mention-level self-attention (MLSA). The feed-forward networks are implemented by two fully connected layers with ReLU activations. Following the previous work, (Zhou and Choi, 2018), the scene-level setting is used, where, each scene is regarded as a document for coreference resolution and linking. During the training, each mini-batch consists of segments obtained from a single document. The joint learning model is optimized with the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of 3e-5, and a warming-up rate of 10%. The model is set to be trained for 100 epochs with an early stop. All the experiments are repeated three times, and the average results are reported.

5 Results and Analysis

In this section, we discuss the experimental results and present a detailed analysis.

5.1 Coreference Resolution Results

The performances of coreference resolution models are shown in Table 2. C^2 with SpanBERT-large achieves the best performance on all evaluation metrics. Comparing to the baseline ACNN model, which uses hand-crafted features, C^2 uses a transformer to better encode the contextual information. Besides that, even though ACNN formulates the coreference resolution and character linking tasks in a pipe-line and uses the coreference resolution result to help character linking, the character linking result cannot be used to help to resolve coreference clusters. As a comparison, we treat both tasks jointly such that they can help each other.

Currently, CorefQA is the best-performing general coreference resolution model on the OntoNotes dataset (Pradhan et al., 2012). However, its performance is limited on the conversation dataset due to

MODEL	RO	RA	CH	MO	JO	PH	EM	RI	MICRO	MACRO
ACNN	78.3	86.5	78.8	81.7	78.3	88.8	69.2	**83.9**	73.7 (0.6)	59.6 (2.3)
BERT-BASE	87.4	89.9	86.6	88.2	87.1	91.1	94.3	62.4	84.0 (0.1)	77.3 (0.2)
BERT-LARGE	88.2	89.9	87.9	88.8	87.7	93.1	93.5	68.0	84.8 (0.2)	79.1 (0.2)
SPANBERT-BASE	87.6	91.8	86.7	88.2	86.8	92.6	94.6	73.3	84.2 (0.1)	77.3 (0.2)
SPANBERT-LARGE	90.9	92.8	88.3	90.3	90.2	94.3	94.6	71.7	85.5 (0.1)	79.8 (0.2)
C^2 (BERT-BASE)	86.5	87.8	85.6	86.8	88.1	92.4	93.0	66.0	84.0 (0.1)	78.6 (0.2)
C^2 (BERT-LARGE)	85.9	90.0	87.3	86.9	87.2	93.0	**96.1**	66.0	84.9 (0.1)	79.5 (0.2)
C^2(SPANBERT-BASE)	89.8	91.3	90.5	90.9	87.8	93.2	93.4	71.3	85.7 (0.1)	81.0 (0.1)
C^2 (SPANBERT-LARGE)	**91.2**	**94.1**	**91.1**	**92.5**	**90.4**	**94.4**	89.2	77.1	**87.0** (0.1)	**81.1** (0.1)

Table 3: Experimental results per character on the character linking. The results are presented in a 1-digit decimal following previous work. Standard deviations of the Micro and Macro F1 scores are shown in brackets. The names in the table are written in two-letter acronyms. Ro: Ross, Ra: Rachel, Ch: Chandler, Mo: Monica, Jo: Joey, Ph: Phoebe, Em: Emily, Ri: Richard

two reasons. First, different from the experimental setting of OntoNotes, the mentions in our experiment setting are gold mentions. Consequently, the flexible span predicting strategy of CorefQA loses its advantages because of the absence of the mention proposal stage. Second, the CorefQA leverages the fine-tuning on other question answering (QA) datasets and it is possible that the used QA dataset (i.e., SQuAD-2.0 (Rajpurkar et al., 2018)) is more similar to OntoNotes rather than the used multiparty conversation dataset, which is typically much more informal. As a result, the effect of such fine-tuning process only works on OntoNotes.

The coarse-to-fine (C2F) model (Joshi et al., 2019b) with a transformer encoder was the previous state-of-the-art model on OntoNotes. Referring to Table 2, given the same text encoder, the proposed C^2 model can constantly outperform the C2F model. These results further demonstrate that with the help of the proposed joint learning framework, the out-of-context character information can help achieve better mention representations so that the coreference models can resolve them more easily.

5.2 Character Linking Results

As shown in Table 3, the proposed joint learning model also achieves the best performance on the character linking task and there are mainly two reasons for that. First, the contextualized mention representations obtained from pre-trained language encoders can better encode the context information than those representations used in ACNN. Second, with the help of coreference clusters, richer context about the whole conversation is encoded for each mention. For example, when using the same pre-trained language model as the encoder, C^2 can always outperform the baseline classification model. These empirical results confirm that, though the BERT and SpanBERT can produce very good vector representation for the mentions based on the local context, the coreference clusters can still provide useful document-level contextual information for linking them to a global character entity.

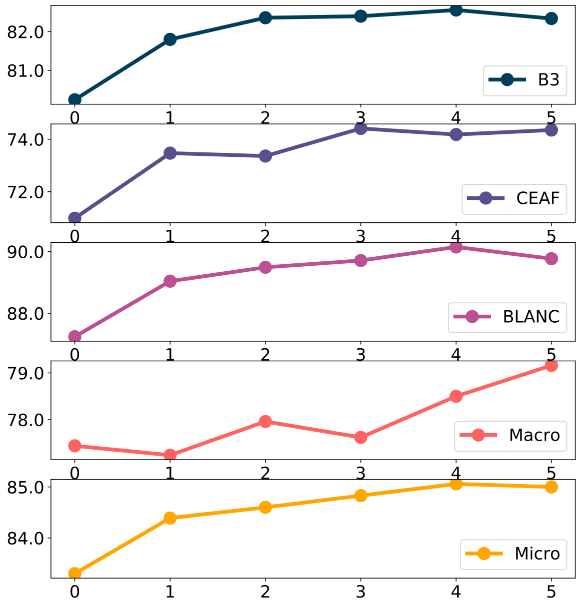
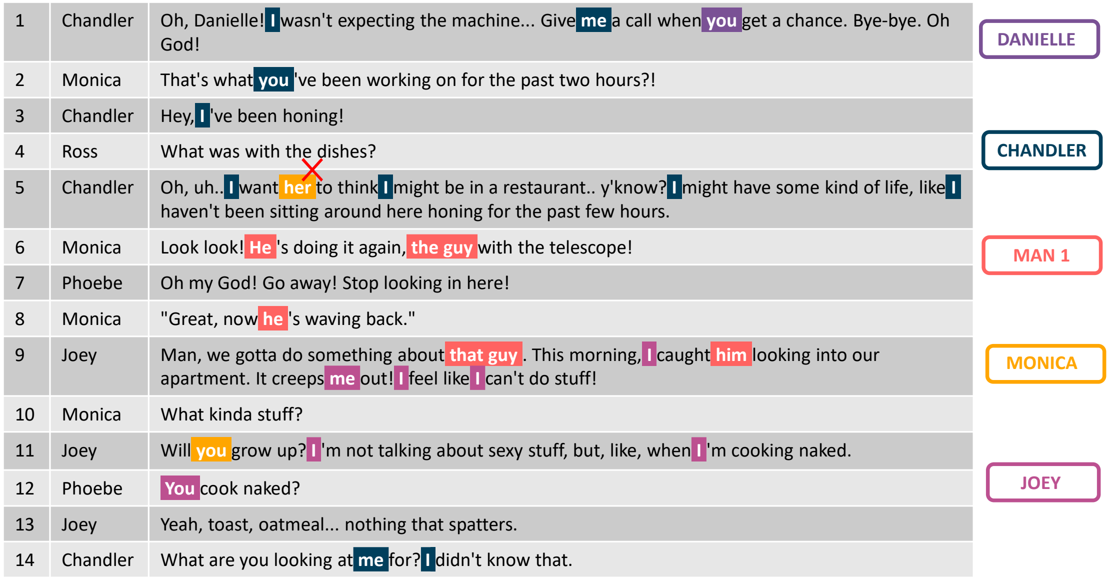


Figure 5: The x-axis is the number of MLSA layers used in the C^2. The y-axes are the F1 scores on each metric for their corresponding tasks. The curves have general trends of going up, which indicates that the model performs better when there are more layers.

5.3 The Number of MLSA Layers

Another contribution of the proposed C^2 model is the proposed mention-level self-attention (MLSA) module, which helps iteratively refine the mention representations according to the other mentions co-occurred within the same document. In this section, to show its effect and the influence of iteration layers, we tried different layers and show their per-



1	Chandler	Oh, Danielle! I wasn't expecting the machine... Give me a call when you get a chance. Bye-bye. Oh God!
2	Monica	That's what you 've been working on for the past two hours?!
3	Chandler	Hey, I 've been honing!
4	Ross	What was with the dishes?
5	Chandler	Oh, uh.. I want her to think I might be in a restaurant.. y'know? I might have some kind of life, like I haven't been sitting around here honing for the past few hours.
6	Monica	Look look! He 's doing it again, the guy with the telescope!
7	Phoebe	Oh my God! Go away! Stop looking in here!
8	Monica	"Great, now he 's waving back."
9	Joey	Man, we gotta do something about that guy . This morning, I caught him looking into our apartment. It creeps me out! I feel like I can't do stuff!
10	Monica	What kinda stuff?
11	Joey	Will you grow up? I 'm not talking about sexy stuff, but, like, when I 'm cooking naked.
12	Phoebe	You cook naked?
13	Joey	Yeah, toast, oatmeal... nothing that spatters.
14	Chandler	What are you looking at me for? I didn't know that.

Figure 6: Case study. All mentions that are linked to the same character and in the same coreference cluster are highlighted with the same color. The misclassified mention is marked with the red cross.

	Coreference F1			Linking F1	
Model	B3	CEAFϕ4	BLANC	Micro	Macro
C^2	**85.54**	**77.48**	**92,17**	**87.05**	**81.09**
- MLSA	83.57	75.32	90.51	86.26	80.32
- Linking	83.50	76.10	90.08	-	-
- Coref.	-	-	-	86.94	79.58

Table 4: Three ablation studies are conducted concerning the MLSA layers, the coreference resolution module, and the character linking module.

formances on the test set in Figure 5. We conducted the experiments with the SpanBERT-Base encoder and all other hyper-parameters are the same. The x-axis is the number of layers, and the y-axes are F1 scores of B3, CEAF, and BLANC for coreference resolution, the Macro and Micro F1 scores for character linking. From the results, we can see that with the increase of layer number from zero to five, the F1 scores on both tasks gradually increase. This trend demonstrates that the model can perform better on both tasks when there are more layers. Meanwhile, the marginal performance improvement of the MLSA layer is decreasing. This indicates that adding too many layers of MLSA may not further help improve the performance because enough context has been included. Considering the balance between performance and computational efficiency, we chose the iteration layers to be two in our current model based on similar observations made on the development set.

5.4 Ablation Study

In this section, we present the ablation study to clearly show the effect of different modules in the proposed framework C^2 in Table 4. First, we try to remove the mention-level self-attention (MLSA) from our joint learning model and a clear performance drop is observed on both tasks. Specifically, the performance on coreference resolution is reduced by 1.21 on the average F1, and meanwhile, the macro-F1 and micro-F1 scores on character linking decreased by 0.77 and 0.79 respectively. The reduction reveals that the MLSA indeed helps achieve better mention representations with the help from co-occurred mentions. Second, we try to remove the coreference resolution and character linking modules. When the character linking module is removed, it is observed that the performance on coreference resolution decreased by 1.94 on the averaged F1 score. When the coreference module is removed, the performance of C^2 on character linking dropped by 0.83 on the average of Micro and Macro F1 scores. These results prove that the modeling of coreference resolution and character linking can indeed help each other and improve the performance significantly, and the proposed joint learning framework can help to achieve that goal.

5.5 Case Study

Besides the quantitative evaluation, in this section, we present the case study to qualitatively evaluate the strengths and weaknesses of the proposed C^2 model. As shown in Figure 6, we randomly select an example from the development set to show the prediction results of the proposed model on both tasks. To illustrate the coreference resolution and character linking results from the C^2 model, the mentions from the same coreference cluster are highlighted with the same color. Also, we use the same color to indicate to which character the mentions are referring. Meanwhile, the falsely predicted result is marked with a red cross.

5.5.1 Strengths

For this example, the results on both tasks are consistent. The mentions that are linked to the same character entity are in the same coreference group and vice versa. Based on this observation and previous experimental results, it is more convincing that the proposed model can effectively solve the two problems at the same time. Besides that, we also notice that the model does not overfit the popular characters. It can correctly solve all the mentions referring to not only main characters, and also for the characters that only appear several times such as *MAN 1*. Last but not least, the proposed model can correctly resolve the mention to the correct antecedent even though there is a long distance between them in the conversation. For example, the mention *me* in utterance 14 can be correctly assigned to the mention *you* in utterance 2, though there are 11 utterances in between. It shows that by putting two tasks together, the proposed model can better utilize the whole conversation context. The only error made by the model is incorrectly classifying a mention and at the same time putting it into a wrong coreference cluster.

5.5.2 Weaknesses

By analyzing the error case, it is noticed that the model may have trouble in handling the mentions that require common sense knowledge. Humans can successfully resolve the mention *her* to Danielle because they know Danielle is on the other side of the telephone, but Monica is in the house. As a result, Chandler can only deceive Danielle but not Monica. But the current model, which only relies on the context, cannot tell the difference.

5.6 Error Analysis

We use the example in Figure 6 to emphasize the error analysis that compares the performance of our model and the baseline models. The details are as follows. In this example, the only mistake made by our model is related to common-sense knowledge, and the baseline models are also not able to make a correct prediction.

For coreference resolution, 3 out of 25 mentions are put into a wrong cluster by the c2f baseline model. The baseline model failed to do long-distance antecedent assignments (e.g., the "me" in utterance 14). Meanwhile, our model is better in this case because it successfully predicts the antecedent of the mention "me", even though its corresponding antecedent is far away in utterance 2. This example demonstrates the advantage that our joint model can use global information obtained from character linking to better resolve the co-referents that are far away from each other.

For character linking, 2 out of 25 mentions are linked to the wrong characters by the baseline model. It is observed that the baseline model cannot consistently make correct linking predictions to less-appeared characters, for example, the "He" in utterance 6. In this case, our model performs better mainly because it can use the information gathered from the nearby co-referents to adjust its linking prediction, as its nearby co-referents are correctly linked to corresponding entities.

6 Related Works

Coreference resolution is the task of grouping mentions to clusters such that all the mentions in the same cluster refer to the same real-world entity (Pradhan et al., 2012; Zhang et al., 2019a,b; Yu et al., 2019). With the help of higher-order coreference resolution mechanism (Lee et al., 2018) and strong pre-trained language models (e.g., SpanBERT (Joshi et al., 2019b)), the end-to-end based coreference resolution systems have been achieving impressive performance on the standard evaluation dataset (Pradhan et al., 2012). Recently, motivated by the success of the transfer learning, Wu et al. (2020) propose to model the coreference resolution task as a question answering problem. Through the careful fine-tuning on a high-quality QA dataset (i.e., SQUAD-2.0 (Rajpurkar et al., 2018)), it achieves the state-of-the-art performance on the standard evaluation benchmark. However, as disclosed by Zhang et al. (2020), current systems are still not perfect. For example, they still cannot effectively handle pronouns, especially those in informal language usage scenarios like conversations. In this paper, we propose to leverage the out-of-context character information to help resolve the coreference relations with a joint learning model, which has been proven effective in the experiments.

As a traditional NLP task, entity linking (Mihalcea and Csomai, 2007; Ji et al., 2015; Kolitsas et al., 2018; Raiman and Raiman, 2018; Onando Mulang et al., 2020; van Hulst et al., 2020) aims at linking mentions in context to entities in the real world (typically in the format of knowledge graph). Typically, the mentions are named entities and the main challenge is the disambiguation. However, as a special case of the entity linking, the character linking task has its challenge that the majority of the mentions are pronouns. In the experiments, we have demonstrated that when the local context is not enough, the richer context information provided by the coreference clusters could be very helpful for linking mentions to the correct characters.

In the NLP community, people have long been thinking that the coreference resolution task and entity linking should be able to help each other. For example, Ratinov and Roth (2012) show how to use knowledge from named-entity linking to improve the coreference resolution, but do not consider doing it in a joint learning approach. After that, Hajishirzi et al. (2013) demonstrate that the coreference resolution and entity linking are complementary in terms of reducing the errors in both tasks. Motivated by these observations, a joint model for coreference, typing, and linking is proposed (Durrett and Klein, 2014) to improve the performance on three tasks at the same time. Compared with previous works, the main contributions of this paper are two-fold: (1) we tackle the challenging character linking problem; (2) we design a novel mention representation encoding method, which has been shown effective on both the coreference resolution and character linking tasks.

7 Conclusion

In this paper, we propose to solve the coreference resolution and character linking tasks jointly. The experimental results show that the proposed model C^2 performs better than all previous models on both tasks. Detailed analysis is also conducted to show the contribution of different modules and the effect of the hyper-parameter.

8 Acknowledgements

This paper was supported by the NSFC Grant U20B2053 from China, the Early Career Scheme (ECS, No. 26206717), the General Research Fund (GRF, No. 16211520), and the Research Impact Fund (RIF, No. R6020-19) from the Research Grants Council (RGC) of Hong Kong, with special thanks to the Tencent AI Lab Rhino-Bird Focused Research Program.

References

Henry Y. Chen, Ethan Zhou, and Jinho D. Choi. 2017. Robust coreference resolution and entity linking on dialogues: Character identification on TV show transcripts. In *Proceedings of CoNLL)*, pages 216–225.

Yu-Hsin Chen and Jinho D. Choi. 2016. Character identification on multiparty conversation: Identifying mentions of characters in TV shows. In *Proceedings of SIGDIAL*, pages 90–100.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the association for computational linguistics*, 2:477–490.

Hannaneh Hajishirzi, Leila Zilles, Daniel S. Weld, and Luke Zettlemoyer. 2013. Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of the EMNLP 2013*, pages 289–299.

Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. 2020. Rel: An entity linker standing on the shoulders of giants. *Proceedings of SIGIR 2020*.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of tac-kbp2015 tri-lingual entity discovery and linking. *Theory and Applications of Categories*.

Mandar Joshi, Danqi Chen, Y. Liu, Daniel S. Weld, L. Zettlemoyer, and Omer Levy. 2019a. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Daniel S. Weld, and Luke Zettlemoyer. 2019b. BERT for coreference resolution: Baselines and analysis. In *Proceedings of EMNLP*.

Ben Kantor and Amir Globerson. 2019. Coreference resolution with entity equalization. In *Proceedings of ACL 2019*, pages 673–677.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of CoNNL 2018*, pages 519–529.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of NAACL (Short Papers)*, pages 687–692.

Rada Mihalcea and Andras Csomai. 2007. Wikify! linking documents to encyclopedic knowledge. CIKM '07, page 233–242.

Isaiah Onando Mulang, Kuldeep Singh, Chaitali Prabhu, Abhishek Nadgeri, Johannes Hoffart, and Jens Lehmann. 2020. Evaluating the impact of knowledge graph context on entity disambiguation models. *arXiv e-prints*, pages arXiv–2008.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. pages 1–40.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: multilingual entity linking by neural type system evolution. *arXiv preprint arXiv:1802.01021*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of ACL 2018*, pages 784–789.

Lev Ratinov and Dan Roth. 2012. Learning-based multi-sieve co-reference resolution with knowledge. In *Proceedings of EMNLP 2012*, pages 1234–1244.

Marta Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17:485 – 510.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L. Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. *Proceedings of the 6th conference on message understanding*, pages 45–52.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings ACL 2020*, pages 6953–6963.

Xintong Yu, Hongming Zhang, Yangqiu Song, Yan Song, and Changshui Zhang. 2019. What you see is what you get: Visual pronoun coreference resolution in dialogues. In *Proceedings of EMNLP-IJCNLP 2019*, pages 5122–5131.

Hongming Zhang, Yan Song, and Yangqiu Song. 2019a. Incorporating context and external knowledge for pronoun coreference resolution. In *Proceedings of NAACL-HLT 2019*, pages 872–881.

Hongming Zhang, Yan Song, Yangqiu Song, and Dong Yu. 2019b. Knowledge-aware pronoun coreference resolution. In *Proceedings of ACL 2019*, pages 867–876.

Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. A brief survey and comparative study of recent development of pronoun coreference resolution. *CoRR*, abs/2009.12721.

Ethan Zhou and Jinho D. Choi. 2018. They exist! introducing plural mentions to coreference resolution and entity linking. In *Proceedings of ICCL*, pages 24–34.

Improving Factual Consistency Between a Response and Persona Facts

Mohsen Mesgar **Edwin Simpson*** **Iryna Gurevych**
Ubiquitous Knowledge Processing Lab (UKP Lab)
Department of Computer Science, Technical University of Darmstadt
https://www.ukp.tu-darmstadt.de

Abstract

Neural models for response generation produce responses that are semantically plausible but not necessarily factually consistent with facts describing the speaker's persona. These models are trained with fully supervised learning where the objective function barely captures factual consistency. We propose to fine-tune these models by reinforcement learning and an efficient reward function that explicitly captures the consistency between a response and persona facts as well as semantic plausibility[1]. Our automatic and human evaluations on the PersonaChat corpus confirm that our approach increases the rate of responses that are factually consistent with persona facts over its supervised counterpart while retaining the language quality of responses.

1 Introduction

Response generation models should ideally generate an appropriate *response* to a given context consisting of utterances previously exchanged between dialogue partners and facts describing the speakers' persona. These models have applications in developing dialogue systems as user interfaces for digital assistants (Bobrow et al., 1977) and also in asynchronous interactions in social media in which speakers define themselves by their profiles.

In this work, we focus on the aspects of persona that can be captured by a set of factual statements, a.k.a., profiles. Table 1 illustrates the persona of the speaker who should respond to the given message. The first response is *topically coherent with the message* and also *linguistically fluent* (or in general, *semantically plausible*) but *factually inconsistent*, unlike the second response, with the second fact in the speaker's persona. We aim to improve the response quality in terms of its factual consistency with facts about the given speaker's persona while retaining its semantic plausibility.

Persona

fact 1: i hate my job
fact 2: i ' m 40 years old
fact 3: i work as a car salesman
fact 4: my wife spends all my money
fact 5: i am planning on getting a divorce

Dialogue History

message: hi , want to be friends?

Generated Responses:

inconsistent: i ' d love to be friends . i ' m 50 years old
consistent: sure , i am 40 , i can tell you about myself

Table 1: A speaker's persona, dialogue history, one inconsistent, and one consistent possible response.

Recent approaches to this problem (Zhang et al., 2018; Dinan et al., 2019; Wolf et al., 2019) generate a response conditioned on persona facts and dialogue history and then use human-generated responses as demonstrations to train their models by fully supervised learning (SL). While this strategy has led to markedly improved performance, there is still a misalignment between this training objective – maximizing the likelihood of human-written responses – and what users care about – generating semantically plausible and factually consistent outputs as determined by humans. This misalignment has several reasons: the maximum likelihood objective considers no distinction between primary errors (e.g. inconsistent responses) and unimportant errors (e.g. selecting the precise word from a set of synonyms); models are incentivized to place probability mass on all human-generated responses, including those that are low-quality; and distributional shift during sampling can degrade performance. Optimizing for targeted quality factors is a principled approach to overcome these problems

---

* Now at the Intelligent Systems Lab, Dept. of Computer Science, University of Bristol.

[1] https://github.com/UKPLab/EACL21-personalized-conversational-system

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 549–562
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

(e.g., Gao et al. (2019) optimize text summarization systems for quality factors relevant to that task).

Our goal is to advance methods for training response generation models on objectives that closely capture the behavior users care about. We first define a reward function to explicitly assesses the quality of a generated response according to *factual consistency with persona facts*, *topical coherence with dialogue history*, and *language fluency*. We then train a policy via reinforcement learning (RL) to maximize the score given by our reward function; the policy generates a token of response at each "time step", and is updated using the Actor-Critic learning approach (Mnih et al., 2016) based on the "reward" our reward function gives to the entire generated response.

We evaluate our approach on PersonaChat (Zhang et al., 2018), a benchmark corpus of English dialogues designed to evaluate the factual consistency between a response and persona facts. We assess the language quality and the factual consistency of responses our RL-based model generates using automatic metrics and human evaluations . Our core contributions are twofold:

- We propose to fine-tune a transformer-based response generation model by an RL method including an efficient reward function that ensures factual consistency with persona facts as well as semantic plausibility of a response.

- We use automatic and human evaluations to show that our RL-based method generates a response that is factually consistent with persona facts more frequently than its SL-based counterpart (Wolf et al., 2019).

The method we present in this paper is motivated in part by long-term concerns about the misalignment of NLP systems with what humans want them to do. When misaligned response generation models generate facts inconsistent with background knowledge like persona facts, their mistakes are relatively low-risk and easy to catch. However, as these systems become more popular to solve essential tasks, their mistakes will likely become more subtle, making this an important area for further research.

2 Method

Let $d = (u_1, ..., u_{T-1})$ be the exchanged utterances between dialogue partners until turn $T-1$, and $p = \{f_1, ..., f_{|p|}\}$ be a persona expressed by a set of facts (i.e. short sentences) about the speaker who should generate a response. Our goal is to generate a response $r = (t_1, ..., t_M)$ consisting of M tokens so that r is consistent with the facts in persona p, topically coherent with u_{T-1}, and linguistically fluent.

2.1 TransferTransfo-SL

We use the *TransferTransfo* (Wolf et al., 2019) dialogue model which is pre-trained and then fine-tuned with fully supervised learning (SL). TransferTransfo is a multi-layer transformer (Vaswani et al., 2017) based on the Generative Pre-trained Transformer (GPT) (Radford et al., 2018). Each transformer layer uses constrained self-attention where every token can only attend to its left context. Generation was performed using beam search with sampling, and an n-gram filtering is used to ensure the model does not directly copy from the persona facts nor former utterances. This model significantly improves over the traditional seq-to-seq, memory-based, and information-retrieval baselines in terms of (1) topical coherence of the response, (2) consistency with a predefined persona, and (3) grammaticality and fluency as evaluated by the automatic metrics in the ConvAI2 competition (Dinan et al., 2019). Since this agent uses transformers, it copes with different lengths of dialogue history.

The transformer layers' parameters in this model are transferred from the pre-trained GPT and then are fine-tuned in a supervised scenario to optimize the losses for the response classification and response generation tasks. The former loss measures if the model distinguishes a correct response appended to the input sequence from a set of randomly sampled distractors, which are randomly selected. The latter one is the language modeling loss that measures how well the model can generate a response similar to the human-generated response. The generative loss is estimated as follows: the self-attention model's final hidden state is fed into an output softmax over the vocabulary to obtain the next response token probabilities. These probabilities are then scored using a negative log-likelihood loss, where the gold next tokens are taken as labels.

2.2 TransferTransfo-RL

Besides the remarkable improvement achieved by TransferTransfo-SL, its generated responses are not necessarily factually consistent with persona

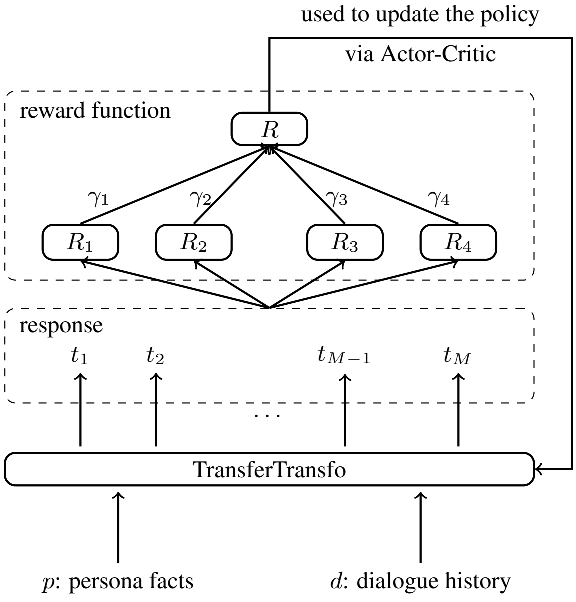


Figure 1: An abstract view of our RL approach.

facts. For example, the inconsistent response in Table 1 is generated by this system. We propose to fine-tune the parameters of this model using reinforcement learning (RL). The TransferTransfo model generates a response token-by-token for a given persona and dialogue history. After generating the last token, i.e. '`<EOS>`', or reaching the maximum length allowed for a response, a reward model assesses the quality of the response (Figure 1). The reward value is used to fine-tune the parameters of TransferTranfo towards the policy that generates a response that is factually consistent with persona facts and also semantically plausible.

Action We consider generating each token of a response as an action performed by the TransferTransfo model:

$$\mathcal{P}_\theta\left(r|s\right) = \mathcal{P}_\theta(t_1|s)\prod_{k=2}^{M}\mathcal{P}_\theta(t_k|t_{1..k-1},s)\ , \quad (1)$$

where t_k is the kth token in response r and $t_{1..k-1}$ indicates the sequence of tokens generated prior to token k. For the sake of brevity, we use the notation s to refer to (p,d). The function $\mathcal{P}_\theta(r|s)$ is the policy with the parameters θ of TransferTransfo.

Reward function A response generation system should ideally generate a response that is factually consistent with the persona facts, topically coherent with the former interactions, and linguistically fluent. Thus, we propose a compound reward consisting of four sub-rewards: R_1 ensures factual consistency with the persona facts. R_2 accounts for topical coherence with the former utterance. R_3 and R_4 reinforce fluency. We use a weighted sum of these sub-rewards as the training signal:

$$R = \gamma_1 R_1 + \gamma_2 R_2 + \gamma_3 R_3 + \gamma_4 R_4\ , \quad (2)$$

where $\gamma_1 + \gamma_2 + \gamma_3 + \gamma_4 = 1$. These weights can be tuned as described below to prevent biasing the policy toward a particular sub-reward.

Persona consistency sub-reward ($\mathbf{R_1}$) Recent studies (Welleck et al., 2019; Dziri et al., 2019) show that consistency with factual information, such as persona facts, can be characterized as a natural language inference (NLI) problem, where entailment labels can be taken as consistent labels and contradiction labels as inconsistent labels. Building on this, we use an NLI model to design this sub-reward. We define our NLI model using BERT as a bidirectional contextualized encoder:

$$\begin{aligned} & h^{\texttt{[cls]}}, - = \texttt{BERT}(\texttt{[cls]}\,f_i\,\texttt{[SEP]}\,r)\ , \\ & [s_e, s_c, s_n] = \texttt{MLP}(h^{\texttt{[cls]}})\ , \qquad (3) \\ & \left[\mathcal{P}_e^{\text{NLI}}, \mathcal{P}_c^{\text{NLI}}, \mathcal{P}_n^{\text{NLI}}\right] = \texttt{Softmax}([s_e, s_c, s_n])\ , \end{aligned}$$

where f_i is a fact in the given persona, r is the generated response, `[SEP]` is the separator token, and $h^{\texttt{[cls]}}$ is provided by BERT to classify semantic relationships between input sentences (Devlin et al., 2019). `MLP` is a linear layer that maps $h^{\texttt{[cls]}}$ to the scores s_e, s_c and s_n, for the *entailment*, *contradiction*, and *neutral* classes, respectively. $\mathcal{P}_e^{\text{NLI}}$, $\mathcal{P}_c^{\text{NLI}}$ and $\mathcal{P}_n^{\text{NLI}}$ denote the respective class probabilities.

We train our NLI model to predict the NLI classes of pairs of utterances and persona facts (§3.3). We then use this trained model as R_1 to penalize the agent if its generated response contradicts one of the facts in the persona, and encourages the agent if its response entails a fact:

$$R_1 = \frac{1}{|p|}\sum_{f_i \in p}\mathcal{P}_e^{\text{NLI}}(f_i, r) - \frac{\beta}{|p|}\sum_{f_i \in p}\mathcal{P}_c^{\text{NLI}}(f_i, r)\ , \quad (4)$$

where $\mathcal{P}_e^{\text{NLI}}$ and $\mathcal{P}_c^{\text{NLI}}$ are the entailment and contradiction probabilities of the relationship between f_i and r. Scalar $\beta \geq 1$ is a marginal penalty for contradiction over entailment: responses that lack entailment may acceptably be neutral, while contradictory responses are a serious consistency error.

The sub-reward for the factual consistency with persona facts is not sufficient to generate a semantically plausible response. The agent can maximize this sub-reward merely by repeating the persona's facts and ignoring topical coherence (for an example, see Appendix A). To prevent such behavior, we assess the topical coherence and grammatical fluency of a response by the following sub-rewards.

Topical coherence sub-reward ($\mathbf{R_2}$) Topical coherence is a crucial property of high-quality dialogues (See et al., 2019; Mesgar et al., 2020). We capture the topical coherence of response r to the last utterance u_{T-1} in dialogue history by representing them using an average pooling layer over their token representations obtained by BERT. Inspired by Baheti et al. (2018) and See et al. (2019), we use *cosine similarity* between $\vec{r}$ and $\vec{u}_{T-1}$ as a proxy for topical coherence:

$$R_2 = \cos(\vec{r}, \vec{u}_{T-1}) \ . \tag{5}$$

Fluency sub-rewards ($\mathbf{R_3}$ and $\mathbf{R_4}$) The above sub-rewards do not assess if the response content expressed is linguistically fluent. As also suggested in prior work (Yarats and Lewis, 2018; Zhao et al., 2019; Bao et al., 2019), applying RL for specific metrics might bring in adverse impacts on linguistic quality. As such, we add sub-rewards R_3 and R_4 to promote linguistic quality. R_3 employs a language model (LM) fine-tuned on a set of utterances (§3.4) to evaluate the language quality of response. To do so, we use the Negative Log-Likelihood (NLL) loss obtained by this LM:

$$R_3 = \frac{\alpha - \texttt{NLL}(r)}{\alpha}, \tag{6}$$

where parameter α is used to map any value of NLL that is greater than α to α so that the output of R_3 will be between 0 and 1. To retain the language quality of responses similar to those of TransferTransfo, we set α to the maximum NLL value that this LM returns for responses generated by the TransferTransfo model on a development set. R_3 is not biased to the length of a response as NLL is already normalized by response length.

Repeated tokens in a response *significantly and negatively* influence the quality of the response (See et al., 2019). R_4 specifically discourages the generation of 1-gram tokens that appear in a response more than one time in a row:

$$R_4 = 1 - \frac{\text{\#repeated-tokens-in-response}}{\text{\#tokens-in-response}}. \tag{7}$$

Weight optimization In combination, these sub-rewards reinforce factual consistency with persona facts, topical coherence, and language fluency. We use their linear combination as a reward R to prevent our policy from becoming overly biased towards any of the sub-rewards. For instance, while generic responses, such as *"I don't know"*, have high fluency, they are discouraged by the persona-consistency sub-reward as they cannot be entailed from any persona fact. However, the weights must be tuned to ensure a suitable balance between the sub-rewards. We apply grid search over the weights and choose the values that yield a policy with the best performance on a validation set (§3.2).

2.3 Training

The goal of RL is to learn a policy, $\mathcal{P}_\theta$, for generating a response that maximizes the expected reward:

$$\mathcal{L} = \mathbb{E}_{\substack{s \in D \\ r \sim \mathcal{P}_\theta(.|s)}} [R(r, s)] \ , \tag{8}$$

where R is the reward function (Equation 2) and $s = (p, d)$ is the given persona and dialogue history that our policy has generated response r for. Function $\mathcal{L}$ is optimized by a stochastic gradient method, where its gradient is (Mnih et al., 2016):

$$\frac{\partial \mathcal{L}}{\partial \theta} = \mathbb{E}_{\substack{s \in D \\ r \sim \mathcal{P}_\theta(.|s)}} \left[R(r, s) \frac{\partial \log \mathcal{P}_\theta(r|s)}{\partial \theta} \right]. \tag{9}$$

To avoid the high-variance issue, we adopt the *actor-critic* method (Mnih et al., 2016) to fine-tune the policy function directly for our quality goals. This approach reduces the variance in the estimated gradient by sampling a single response $r \sim \mathcal{P}_\theta(.|s)$ and computing the difference between its reward $R(r, s)$ and the reward predicted by a critic, $\eta(t_{1..k}, s)$, for the tokens up to position k in response r. The gradient in Equation 9 is then approximated as follows:

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \sum_k (R(r, s) - \eta(t_{1..k}, s)) \frac{\partial}{\partial \theta} \log \mathcal{P}_\theta(t_k|s) \ . \tag{10}$$

The critic function is $\eta = w^T h_k$, where w is its trainable parameters and h_k is the vector returned by the TransferTransfo model (our agent) at position k. We update the critic's parameters after each update of the policy's parameters by minimizing the squared error between its estimated rewards and the value our reward model assigns to the response:

$$\mathcal{L}_\eta = \mathbb{E}_{\substack{s \in D \\ r \sim P_\theta(.|s)}} \sum_k [\eta(t_{1..k}, s) - R(r, s)]^2. \tag{11}$$

	Train	Validation
Num. of dialogues	17,878	1,000
Num. of utterances	262,876	15,602
Num. of personas	955	200

Table 2: Statistics of PersonaChat as used in the ConvAI2 competition and our experiments.

3 Experiments

We measure to what extent our RL-based fine-tuning (§2) improves the factual consistency of generated responses while retaining their semantic plausibility. We first introduce the corpus used in our experiments (§3.1). We then evaluate the TransferTransfo-SL and TransferTransfo-RL systems by automatic and human evaluations (§3.2). We finally analyze the models we employ to estimate the factual consistency (§3.3) and language fluency (§3.4) sub-rewards.

3.1 PersonaChat Corpus

We use datasets built on the PersonaChat corpus (Zhang et al., 2018), which consists of dialogues, in English, with 6 to 8 turns between randomly paired human crowd-workers. The workers were assigned short text facts representing personas and instructed to talk to their dialogue partner *naturally to discover each other's persona*. We chose this corpus because of its focus on promoting natural conversations while grounding conversations in the persona facts. Each persona consists of 4 or 5 facts, and on average is assigned to 8.3 unique dialogues. We train and evaluate the aforementioned systems on the standard splits of the version of this corpus made available in ParlAI[2] for the ConvAI2 challenge (Dinan et al., 2019) (Table 2). As the test set is hidden, we evaluate the systems on the validation set. To create a training and evaluation sample consisting of a persona and a dialogue history (Table 1), each dialogue is split at each dialogue turn.

3.2 Response Generation

We study to what extent our RL approach generates a response that is factually consistent with given persona facts and semantically plausible. We use TranserTransfo, which performed best in automatic evaluation and second-best in human evaluation among 26 participants in the ConvAI2 competition, as a response generation model.

Settings Following the training setup used by Wolf et al. (2019), we fine-tune TransferTransfo on all training samples in PersonaChat and stop the fine-tuning after three epochs. We refer to this fine-tuned model as TransferTransfo-SL. For TransferTransfo-RL, we continue to fine-tune the TransferTransfo model with our RL approach on 90% of the training set for one epoch, where after each policy update, the critic's parameters are updated for 5 times. For R_1, we use the BERT model trained on Dialogue NLI (§3.3) with $\beta = 2$ and for R_3 we use Dialogue LM (§3.4) with $\alpha = 4$. The maximum response length is 20. The input texts are tokenized according to the GPT byte pair encoding (BPE) but the reward is computed on a completely decoded response text. We use the remaining 10% of the training set to choose the sub-reward weights (Equation 2) based on token-level F1-score, which indicates how well the system's responses match the content of human-generated responses (examined weights and their F1-scores are in Appendix B), resulting in $\gamma_1 = 0.4$, $\gamma_2 = 0.16$, $\gamma_3 = 0.22$ and $\gamma_4 = 0.22$. The high weight of the persona consistency sub-reward (γ_1) is compatible with the goal of dialogues in PersonaChat, which is to reveal the persona of dialogue partners. The weights are also consistent with See et al. (2019): fluency factors (γ_3 and γ_4) are more crucial than cosine-relatedness (γ_2) for responses in this corpus.

3.2.1 Automatic Evaluation

We evaluate these systems on the PersonaChat validation set as used in ConvAI2. We report PPL, F1, and BLEU to assess generated responses according to reference responses. We evaluate the factual consistency of a response and the given persona facts using our NLI model (§3.4). It assigns inference relations between a generated response and each fact in the given persona. Given N fact-response pairs in the whole evaluation set, this metric is:

$$\text{PC} = 100\frac{N_e - N_c}{N}\ , \qquad (12)$$

where N_e and N_c are the numbers of entailment and contradiction labels, respectively.

Results The TransferTransfo-RL outperforms its supervised counterparts on all metrics except PPL (Table 3). The improvements on F1 and BLEU indicate that responses generated by TransferTransfo-RL are more similar to reference responses generated by humans and are not biased

---

[2] `https://github.com/facebookresearch/ParlAI/tree/master/projects/personachat`

Method	PPL	F1	BLEU	PC
TransferTransfo-SL	21.31	17.06	0.065	09.32
TransferTransfo-RL	22.64	**17.78**	**0.067**	**13.06**

Table 3: Automatic evaluations of responses generated by TransferTransfo-SL and TransferTransfo-RL.

toward simply repeating persona facts or previous utterances. It also shows that responses are as informative as human-provided ones. Our RL method decreases the average word repetition rate (Equation 7) from 9% with TransferTransfo-SL to 7%, increasing the language fluency of responses. So far, we observe that the RL method could retain and even improve the semantic plausibility of a response.

Regarding the factual consistency between a response and given persona facts, TransferTransfo-RL scores significantly higher for the PC metric. This indicates that the number of evaluation samples for which TransferTransfo-RL generates a response consistent with given persona facts is significantly higher than what TransferTransfo-SL does. Looking at PC in detail (Table 4, top), TransferTransfo-RL increases the frequency of cases whose generated responses are entailed from (or consistent with) persona facts by 3.41% over TransferTransfo-SL, while reducing contradictions (or inconsistency) by 0.07% and neutral by 3.61%; showing that fine-tuning with RL improves the policy for generating a response that is factually consistent with persona facts. While our combined reward function achieves good all-round performance, ablation experiments (Appendix A and B) show that each sub-reward is effective and necessary to capture consistency with persona facts, topical coherence, and language fluency.

3.2.2 Human Evaluation

We also conduct a human evaluation between TransferTransfo-RL and TransferTransfo-SL. We randomly select *100 samples*, each of which consists of a dialogue history, a persona, and the responses generated by the examined systems. We ask *seven human judges* (two native and five fluent English speakers) to assign a *consistency label* from {*consistent, neutral, contradicting*} to the response concerning the facts in the persona (instructions in Appendix D). We also ask the human judges to rate the *semantic plausibility* of each response with an ordinal score ranging from 1 (worst) to 5 (best), encompassing *coherence*, *grammatical correctness*, and *low repetitiveness*.

	Consistent	Contradicting	Neutral
Automatic Evaluation			
TransferTransfo-SL	11.14	01.82	87.04
TransferTransfo-RL	**14.81**	**01.75**	**83.43**
Δ	3.41 ↑	0.07 ↓	3.61 ↓
Human Evaluation			
TransferTransfo-SL	43.71	17.71	38.58
TransferTransfo-RL	**52.71**	**14.00**	**33.29**
Δ	9.00 ↑	3.71 ↓	5.29 ↓

Table 4: Frequencies (%) of consistency labels. Automatic evaluation uses our NLI model to assign labels for the whole evaluation set. For human evaluation, judges labeled 100 samples. Δ is the improvement of TransferTransfo-RL over TransferTransfo-SL.

Method	Average Semantic Plausibility
TransferTransfo-SL	3.33
TransferTransfo-RL	**3.50**

Table 5: Human evaluation: semantic plausibility.

Results Table 4 (bottom) shows the average percentage of consistency labels human judges assign to responses generated by TransferTransfo-RL and TransferTransfo-SL. The number of samples for which TransferTransfo-SL generates a consistent response increases by 9% using our RL fine-tuning approach while contradictions (or inconsistencies) decrease by 3.71%, confirming that human judges more frequently find responses generated by TransferTransfo-RL factually consistent with persona facts than those of TransferTransfo-SL. The number of neutral responses also decreases, suggesting fewer generic responses, as neutral responses tend to be generic (Welleck et al., 2019).

Overall, Table 4 shows a similar trend between the human and the automatic evaluations, confirming the findings of the automatic evaluation. Unlike the human evaluation, our automatic evaluation shows that the models generate a neutral response for most cases. The NLI model assesses more responses to be neutral than humans do – humans can reason about entailment relations using their common senses, while the NLI model does not identify any relation. Further analysis (Appendix E) shows that for over half of the cases for which TransferTransfo-SL generates a contradicting (inconsistent) response, our TransferTransfo-RL generates a consistent response, indicating that the idea of using RL to fine-tune a pre-trained agent improves its capability in generating a factually consistent response with persona facts.

In terms of semantic plausibility (topically coherent and linguistically fluent), Table 5 shows that the human judges find responses generated by TransferTransfo-RL are on par with those of TransferTransfo-SL, showing the effectiveness of our topical coherence and fluency sub-rewards.

3.3 Persona-Consistency Sub-reward Validation

As discussed in §2, assessing factual consistency with persona facts can be characterized as an NLI problem. In this experiment, we investigate the choice of the NLI model for this sub-reward by comparing our BERT-based NLI model (§2) with recent NLI models on the *Dialogue NLI* dataset (Welleck et al., 2019). This dataset, which is designed for evaluating factual NLI in dialogues, consists of a set of fact-utterance, fact-fact, and utterance-utterance pairs extracted from the PersonaChat corpus. Each pair is accompanied by a *human-annotated NLI label*, i.e., entailment (or consistent), contradiction (or inconsistent), and neutral. Two examples of the fact-utterance pair from this dataset are: *"My dad is a priest."* **contradicts** *"Since my dad is a mechanic we had mostly car books."*; and *"I like playing basketball"* **entails** *"I prefer basketball. Team sports are fun."*. This dataset contains 310,110 training, 16,500 validation and 16,500 test pairs. Besides the standard test set, which was annotated by *one crowdworker*, there is *Test Gold* containing 12,376 of test pairs, which were annotated by *three crowdworkers* (Welleck et al., 2019).

We compare our BERT-based NLI model with (1) **Majority**, which returns the majority class; (2) **ESIM** Enhanced Sequential Inference Model (Chen et al., 2017), an LSTM-based model with inter-sentence attentions. ESIM is the state of the art on the Dialogue NLI dataset. We use *bert-base-uncased* (Devlin et al., 2019) to encode utterances and facts. We fine-tune the whole model during training. We set the maximum input length to 128, the learning rate to 5×10^{-5}, and the training- and evaluation-batch sizes to 32 and 8, respectively. We compare the NLI models using *accuracy* (Welleck et al., 2019).

Results Table 6 shows that the BERT-based NLI model outperforms ESIM, suggesting that our model better captures the factual relationships between an utterance and a persona fact. Welleck et al. (2019) previously demonstrated that the performance of ESIM is sufficient to check the factual consistency between a response and persona facts; as our model outperforms ESIM, we chose our NLI model for the consistency sub-reward R_1. Indeed, a more accurate NLI model reduces the noise in the reward function and consequently the errors our system makes.

Model	Validation	Test	Test Gold
Majority	33.33	34.54	34.96
ESIM	86.31	88.20	92.45
Our NLI model	**86.84**	**89.50**	**93.60**

Table 6: Accuracy of candidate NLI models for R_1 on the Dialogue NLI dataset.

Model	PPL
Non-Dialogue LM	108.29
Dialogue LM	**10.01**

Table 7: The perplexity (PPL) of the language model (Dialogue LM) used for the fluency sub-reward $R_{3.1}$.

3.4 Response Fluency Sub-reward Validation

Sub-reward R_3 requires a language model to measure the language quality of a response. In this experiment, we investigate if fine-tuning a pre-trained, non-dialogue language model on dialogue utterances makes it suitable for this goal. To do so, we compare (1) **Non-Dialogue LM**, which is the GPT language model with no fine-tuning; and (2) **Dialogue LM**, which is the GPT language model fine-tuned on utterances from PersonaChat. We fine-tune the GPT language model (Radford et al., 2018) for three epochs on 90% of utterances ($\approx$236,588) from the PersonaChat training set. We evaluate the language model on the remaining 10% ($\approx$26,288) of utterances, so the PersonaChat validation dialogues remain unseen for evaluating our dialogue systems. Training- and validation-batch sizes are 8 and 16, respectively. Learning rate is 6.25×10^{-5}, and *perplexity (PPL)* is the evaluation metric.

Results Dialogue LM substantially improves perplexity over Non-Dialogue LM (Table 7). This shows that the fine-tuned language model better captures the linguistic properties of dialogue utterances, yielding a more suitable language model for the fluency sub-reward R_3. See et al. (2019) validated the benefits of cosine similarity for estimating the coherence (R_2) and word repetition (R_4) for language quality.

4 Discussions

Case analysis We presented one example of an evaluation sample in Table 1, in which the inconsistent response is generated by TransferTransfo-SL and the consistent one by TransferTransfo-RL. Since TransferTransfo-SL is fine-tuned only with reference responses and does not have any training signal for factual consistency, we speculate that variants of *"I'm 50 years old"* occur in the training set leading the agent to produce a response that is inconsistent with the persona fact *"I'm 40 years old"*. In contrast, TransferTransfo-RL generates a consistent response which is also topically coherent with the given question and linguistically fluent. The above sample is an example of "attribute" consistency, where the response should express an attribute of the speaker. Table 8 shows some other evaluation samples. The top sample shows that TransferTransfo-RL can deal with "have" consistency. Our system correctly recognizes the number of dogs the speaker has and grounds its response on this fact. The evaluation sample in the middle row of Table 8 shows that our RL-based model can also deal with "like-to-do" consistency.

Although TransferTransfo-RL outperforms TransferTransfo-SL in generating different types of consistent responses (such as 'attribute", "have", and "like-to-do"), they both struggle with generating consistent responses for evaluation samples in which understanding of persona facts and dialogue history requires common sense knowledge. As an example, consider the second evaluation sample shown in Table 8. TransferTransfo-SL generates the response "*I'm not married yet*" which contradicts the first fact of the given persona "*My husband is adopted.*"; it seems the model does not have enough knowledge to capture the semantic relationship between "*my husband*" and "*marriage*". The bottom evaluation sample in Table 8 demonstrates the lack of common sense knowledge for TransferTransfo-RL as well. The response "*I like to go to church to sing with wife*" contradicts the fact "*My wife left me and took my children*" in the given dialogue history.

Limitations One limitation of our work is to narrow a speaker's persona to a set of facts expressed as short sentences. Persona has other aspects, such as speaking styles, which need a separate study. Nevertheless, the research question and experiments presented in this work demonstrate the benefits of RL methods for fine-tuning transformer-based models, which are already pretrained, to obtain a policy more aligned with target quality factors. Other aspects of the persona can also be involved in the reward function, given that our method potentially reduces the need for the high-quality demonstration responses generated by humans for supervised fine-tuning.

Future directions In this paper, we demonstrate the effectiveness of RL over SL for fine-tuning pretrained neural models (like GPT) for generating responses that fulfill quality goals such as factual consistency with given persona facts and semantic plausibility in a single round of dialogue. Therefore, the next step might be adopting our reward function to generate factually-consistent responses while retaining the diversity of responses through multiple rounds of dialogue.

5 Related Work

There are two types of approach to persona consistency. The first category includes systems that learn speaker-level embeddings from responses produced by a particular speaker (Li et al., 2016a; Madotto et al., 2019). These systems depend on the availability of suitable responses performed by the speaker whose persona we wish to imitate. If those responses do not reveal the persona information, dialogue systems cannot learn the persona. Moreover, these systems cannot be adapted to new personas at deployment time since the persona embeddings must be learned from training data. So our approach is complementary to them and not directly comparable.

The second category includes systems that rely on a set of facts about a persona. For example, Zhang et al. (2018) propose a key-value memory neural model for this task. This model is outperformed by TransferTransfo (Wolf et al., 2019), which is used in our experiments. Welleck et al. (2019) rank a given set of utterances using an NLI model to select a persona-consistent response. In contrast, we use NLI to train a generative model. Song et al. (2020) propose an NLI-based reward for persona consistency that calculates a score using only the persona facts with the highest entailment and contradiction probabilities, rather than the whole persona. Their approach does not reward topical coherence, which we found crucial

Persona	Dialogue History	TransferTransfo-SL	TransferTransfo-RL
fact 1: i go to the gym regularly . *fact 2*: i like doing the treadmill and rowing machine . *fact 3*: i ' ve short hair . *fact 4*: two dogs live with me . *fact 5*: eating is something i do when i am bored .	*message*: hello , how are you ? *response*: hello , i am feeling fit since i visited the gym today ! how are you ? that ' s absolutely great just fantastic after rowing and running on the treadmill , i ' m pretty hungry . do you work out ? *message*: yes after i cook all my chef food	i do a lot of cardio . i ' m really short , so i am always looking to	i love to cook , i have 2 dogs .
fact 1: my husband is adopted . *fact 2*: i like to watch nhl hockey . *fact 3*: i live is a rural farming community . *fact 4*: i ' ve a german shepherd dog .	*message*: the farm animals and a dog . his name is lucky . do you go to church ? *response*: we have a german shepard named bacon . *message*: i love bacon we cook it for breakfast every sunday after church we are christain . response: we don ' t go to church often . our religion is hockey . *message*: my sisters boyfriend plays hockey , she works as a pole dancer and met him there .	i ' m not married yet , i ' m not married yet .	i love to watch nhl games .
fact 1: i am overweight and unhappy *fact 2*: i do not believe in god *fact 3*: i spend most of my time on facebook when i am not working *fact 4*: i work at a nursing home *fact 5*: my wife left me and took my children	*message*: hi i am a custodian named darlene , how are you ? *response*: i am good i work in a nursing home and just at home on facebook *message*: my hobbies include playing guitar and singing , do you have any?	i do i am a drummer in a band and i work as a nurse at a nursing home	i like to go to church to sing with my wife

Table 8: Some example personas, dialogue histories and the responses generated by our examined agents from the evaluation set.

for relieving effects of the persona-consistency subreward on the quality of response.

Persona consistency was also a quality target in the ConvAI2 dialogue generation competition (Dinan et al., 2019). The winner of the human evaluation part of ConvAI2 is the “Lost in Conversation” system (Dinan et al., 2019), which is also a transformer-based model trained by SL on two extra datasets besides PersonaChat. In our paper, we used TransferTransfo trained only on PersonaChat. Our experiments showed that our idea of using RL for fine-tuning neural agents improves factual consistency between a response and persona facts by accounting for it in its reward function.

RL has been extensively used for training task-oriented dialogue systems (e.g., Nogueira and Cho (2017); Liu et al. (2018)). Unlike task-oriented scenarios, where a reward can measure if a task is fulfilled or not, incorporating persona facts lacks a straight-forward measurable outcome. Li et al. (2016b) use RL for generating open-domain dialogue using REINFORCE (instead of Actor-Critic) and an RNN-based model. This agent has no notion of factual consistency with facts about a persona, so is not comparable with our system.

6 Conclusions

We proposed to fine-tune response generation models by RL to improve on the quality goals that matter, e.g., factual consistency between a response and persona facts while retaining semantic plausibility. We adopted the actor-critic method for fine-tuning a pre-trained transformer-based model by defining an efficient and effective reward function measuring persona consistency, topical coherence, and language fluency. Automatic and human evaluations on PersonaChat demonstrate that compared to just using supervised learning, further fine-tuning with RL yields responses that are more frequently factually consistent with persona facts while still semantically plausible.

Acknowledgments

This work was supported by the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1 and grant GU 798/17-1). We thank Kevin Stowe and Leonardo Filipe Rodrigues Ribeiro for valuable feedback on earlier drafts of this paper. We also thank anonymous reviewers for their constructive suggestions.

References

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing,* Brussels, Belgium, 31 October – 4 November 2018, pages 3970–3980.

Siqi Bao, Huang He, Fan Wang, Rongzhong Lian, and Hua Wu. 2019. Know more about each other: Evolving dialogue strategy via compound assessment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Florence, Italy, 28 July – 2 August 2019, pages 5382–5391.

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry Thompson, and Terry Winograd. 1977. GUS, a frame-driven dialog system. *Artificial intelligence*, 8(2):155–173.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Vancouver, Canada, 30 July – 4 August 2017, pages 1657–1668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies , Volume 1 (Long and Short Papers)* , Minneapolis, Minnesota., 2–7 June 2019, pages 4171–4186.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander H. Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander I. Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2019. The second conversational intelligence challenge (ConvAI2). *CoRR*, abs/1902.00098.

Nouha Dziri, Ehsan Kamalloo, Kory Mathewson, and Osmar Zaiane. 2019. Evaluating coherence in dialogue systems using entailment. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies , Volume 1 (Long and Short Papers)* , Minneapolis, Minnesota., 2–7 June 2019, pages 3806–3812.

Yang Gao, Christian M. Meyer, Mohsen Mesgar, and Iryna Gurevych. 2019. Reward learning for efficient reinforcement learning in extractive document summarisation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 2350–2356. International Joint Conferences on Artificial Intelligence Organization.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016a. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Berlin, Germany, 7–12 August 2016, pages 994–1003.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016b. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,* Austin, Texas, 1–5 November 2016, pages 1192–1202.

Bing Liu, Gokhan Tür, Dilek Hakkani-Tür, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Melbourne, Australia, 15–20 July 2018, pages 2060–2069.

Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, and Pascale Fung. 2019. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Florence, Italy, 28 July – 2 August 2019, pages 5454–5459.

Mohsen Mesgar, Sebastian Bücker, and Iryna Gurevych. 2020. Dialogue coherence assessment without explicit dialogue act labels. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1439–1450, Online. Association for Computational Linguistics.

Volodymyr Mnih, Adria Puigdomenech, Badia Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning,* New York, New York, 20–22 Jun 2016, pages 1928–1937.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* Copenhagen, Denmark, 7—11 September 2017, pages 574–583.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI.*

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 57th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), Florence, Italy, 28 July – 2 August 2019, pages 1702–1723.

Haoyu Song, Wei-Nan Zhang, Jingwen Hu, and Ting Liu. 2020. Generating persona consistent dialogues by exploiting natural language inference. In *Proceedings of the 34rd Conference on the Advancement of Artificial Intelligence,* New York, New York, 7–12 February 2020, page [TO APPEAR].

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing (NeurIPS 2017),* Long Beach, California, 4–9 December 2017, pages 5998–6008.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Florence, Italy, 28 July – 2 August 2019, pages 3731–3741.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A transfer learning approach for neural network based conversational agents. *CoRR*, abs/1901.08149.

Denis Yarats and Mike Lewis. 2018. Hierarchical text generation and planning for strategic dialogues. In *Proceedings of the 35th International Conference on Machine Learning*, page 5591–5599.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Melbourne, Australia, 15–20 July 2018, pages 2204–2213.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies , Volume 1 (Long and Short Papers)* , Minneapolis, Minnesota., 2–7 June 2019, pages 1208–1218.

A Using Only the Persona-Consistency Sub-reward R_1

By using the persona-consistency sub-reward as the only reward function, the policy learns only to generate responses that state information about the persona facts and ignores the topical coherence with its former utterance.

Persona

fact 1: i prefer vinyl records to any other music format.
fact 2: i fix airplanes for a living.
fact 3: i drive junk cars that no one else wants.
fact 4: i think if i work hard enough i can fix the world.

Dialogue

message: hi how are you doing?
response: i am fine. i am a mechanic, so i fix airplanes. i fix them.
message: do you have any hobbies?
response: i enjoy listening to vinyl records. i also fix planes. i also fix them. i also fix
message: sounds cool.
response: i also fix airplanes. i also fix airplanes. i also fix them. i also fix them.

Table 9: Response generated by the TransferTransfo model where it is fine-tuned by only the persona-consistency sub-reward ($R = R_1$).

Table 9 illustrates an example dialogue that is conducted with an agent trained with only the persona consistency sub-reward ($R = R_1$). The agent always repeats, *"i fix airplanes. i fix them."*, no matter what the input message is about. This problem not only produces topically irrelevant responses but also makes the agent look nagging and self-centered in a conversation.

Persona

fact 1: i like hunting.
...

Dialogue

message: hi how are you doing?
response: hunting hunting hunting hunting hunting hunting
message: do you have any hobbies?
response: hunting hunting hunting hunting hunting hunting

Table 10: Another example dialogue with an agent that is trained by only the persona-consistency sub-reward ($R = R_1$).

Table 10 illustrates another example dialogue with the agent where it is trained only by persona-consistency sub-reward. The agent keeps repeating *"hunting"* from the persona to maximize its reward. The NLI model used for R_1 evaluates the inference relation between a response and a persona and does not capture the topical coherence of the response with its former utterance and language fluency of the response. It is therefore necessary to use R_1 in combination with topical coherence (R_2) and language fluency sub-rewards (R_3 and R_4), as we propose in our reward function.

B Weight Optimization and Reward Ablation

We examine various weight sets $(\gamma_1, \gamma_2, \gamma_3, \gamma_4)$ to balance the contribution of sub-rewards in the complete reward function on the held out set (10% of the PersonaChat training set). Table 11 shows those weights. The balanced weights give the highest F1 score, suggesting that a combination of sub-rewards leads to responses that are more similar to the human responses.

γ_1	γ_2	γ_3	γ_4	$\mathbf{F_1}$
0.00	0.00	0.00	0.00	02.04
1.00	0.00	0.00	0.00	16.34
0.00	1.00	0.00	0.00	16.12
0.00	0.00	1.00	0.00	12.77
0.00	0.00	0.00	1.00	17.91
0.70	0.00	0.30	0.00	16.37
0.65	0.00	0.35	0.00	19.90
0.60	0.00	0.40	0.00	16.98
0.50	0.00	0.50	0.00	16.07
0.25	0.25	0.25	0.25	18.27
0.60	0.20	0.00	0.20	15.54
0.40	0.20	0.20	0.20	20.57
0.40	**0.16**	**0.22**	**0.22**	**20.75**
0.45	0.13	0.17	0.20	19.98
0.47	0.12	0.17	0.20	19.95
0.47	0.10	0.17	0.21	20.33
0.47	0.10	0.19	0.19	19.73
0.50	0.10	0.16	0.20	19.10
0.40	0.20	0.15	0.20	19.34
0.43	0.20	0.12	0.20	20.22
0.45	0.20	0.12	0.20	20.13
0.45	0.25	0.00	0.25	17.40
0.40	0.10	0.25	0.25	18.93
0.40	0.15	0.20	0.20	19.80
0.40	0.20	0.20	0.15	20.44
0.45	0.17	0.21	0.17	18.85
0.50	0.15	0.15	0.15	20.25
0.47	0.13	0.20	0.15	19.97
0.50	0.15	0.20	0.15	17.64
0.55	0.15	0.15	0.10	19.15

Table 11: The examined sub-reward weights and their corresponding F1 on our validation set, i.e., 10% of the PersonaChat training set.

We also evaluate the use of each sub-reward in isolation, and show the results in Table 12, in comparison with our chosen balanced weights in the bottom line. For the other metrics, we can see that $\gamma_1 = 1$ maximizes the number of entailments from

γ_1	γ_2	γ_3	γ_4	$\mathbf{F_1}$	PPL	Repetition(%)	Consistent (%)	Neutral (%)	Contradiction (%)	PC
1.00	0.00	0.00	0.00	16.34	25.35	36.20	**34.02**	64.62	01.36	**66.33**
0.00	1.00	0.00	0.00	16.12	21.44	13.36	12.24	86.34	**01.42**	55.50
0.00	0.00	1.00	0.00	12.77	**04.76**	12.21	00.46	**99.35**	00.20	51.49
0.00	0.00	0.00	1.00	17.91	21.35	**01.62**	03.53	95.90	00.57	50.13
0.40	0.16	0.22	0.22	**20.75**	12.36	09.61	14.14	84.54	01.32	56.50

Table 12: Performance metrics on our validation set (10% of the PersonaChat training set) when training is performed with each individual sub-reward and our chosen weighted sum of sub-rewards.

the persona facts, $\gamma_3 = 1$ minimizes perplexity, and $\gamma_4 = 1$ gives lowest repetition. Besides F1 score, the balanced weights give good performance across perplexity, repetition, and persona consistency. The setups with fewer neutral responses also tend to have more responses that contradict the persona facts, e.g., for $\gamma_1 = 1$. Neutral responses are a trivial way to avoid contradictory responses and the setup with the least contradictions, $\gamma_3 = 1$, has almost no responses that are consistent with the persona facts. The better overall persona consistency is reflected in the highest PC score for $\gamma_1 = 1$ and next highest for the balanced weights, which trades of PC for less repetition, lower perplexity and a higher F1 score.

C REINFORCE vs Actor-Critic

Figures 2 and 3 show the trend of changes in our reward function during training by REINFORCE and Actor-Critic, respectively. All parameters are the same for the two experiments. We observe that the actor-critic approach converges faster and also is less noisy (has a lower variance) than REINFORCE.

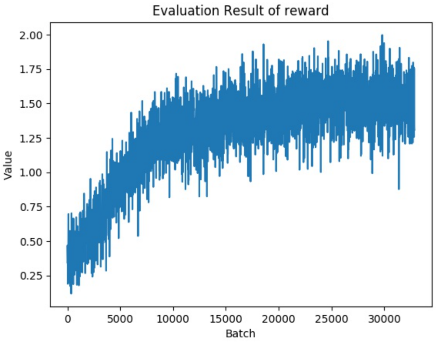


Figure 2: The reward curve during training by REINFORCE.

D Human Evaluation

For each sample, we show to each participant a set of persona facts, a dialogue history, and the response generated by one of TransferTransfo-SL and TransferTransfo-RL. We instruct our participants to assess semantic plausibility according to the following objective definition: *"grammatical correctness, lowest repetitiveness, and coherence"*. The plausibility rates are integer values between 1 and 5, where 5 is most plausible.

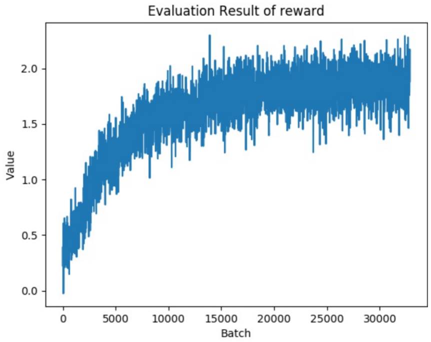


Figure 3: The reward curve during training by Actor-Critic.

To measure persona consistency, we instruct participants as follows:

An answer is considered consistent if:

- It contradicts with neither the dialogue history nor the persona facts;
- It is relevant to any of the given persona facts.

An answer is considered neutral if:

- It contradicts with neither the dialogue history nor the persona facts;
- It is **not** relevant to any of the given persona facts.

E Human Evaluation: Confusion Matrix

Table 13 presents the distributions of consistency labels for TransferTransfo-RL's responses given the consistency labels for TransferTransfo-SL's responses. For the majority of cases whose

		TransferTransfo-RL label		
		Consistent	Neutral	Contradicting
Transfer	Consistent	57.46	27.73	14.82
Transfo	Neutral	46.87	44.99	08.14
-SL	Contradicting	52.48	22.05	25.47

Table 13: Each row corresponds to the cases in the human evaluation for which TransferTransfo-SL received a particular consistency label. The values in each row show the percentages of consistency labels for TransferTransfo-RL for the same data points.

TransferTransfo-SL's responses are contradictory or neutral, TransferTransfo-RL generates consistent responses, showing improved factual consistency with persona facts. However, TransferTransfo-RL generates contradictory responses for some cases whose TransferTransfo-SL responses are consistent with their personas. This may be due to errors in the NLI model's predictions of entailment, hence a more accurate NLI model may improve the quality of the reward function and consequently the consistency of responses. Alternatively, these contradictory responses may receive high rewards from the topic consistency and fluency sub-rewards, which could override R_1.

PolyLM: Learning about Polysemy through Language Modeling

Alan Ansell[1,2] **, Felipe Bravo-Marquez**[3]**, Bernhard Pfahringer**[2]
[1]Language Technology Lab, University of Cambridge
[2]Department of Computer Science, University of Waikato
[3]Department of Computer Science, University of Chile & IMFD
aja63@cam.ac.uk, fbravo@dcc.uchile.cl,
bernhard@waikato.ac.nz

Abstract

To avoid the "meaning conflation deficiency" of word embeddings, a number of models have aimed to embed individual word *senses*. These methods at one time performed well on tasks such as word sense induction (WSI), but they have since been overtaken by task-specific techniques which exploit contextualized embeddings. However, sense embeddings and contextualization need not be mutually exclusive. We introduce PolyLM, a method which formulates the task of learning sense embeddings as a language modeling problem, allowing contextualization techniques to be applied. PolyLM is based on two underlying assumptions about word senses: firstly, that the probability of a word occurring in a given context is equal to the sum of the probabilities of its individual senses occurring; and secondly, that for a given occurrence of a word, one of its senses tends to be much more plausible in the context than the others. We evaluate PolyLM on WSI, showing that it performs considerably better than previous sense embedding techniques, and matches the current state-of-the-art specialized WSI method despite having six times fewer parameters. Code and pre-trained models are available at `https://github.com/AlanAnsell/PolyLM`.

1 Introduction

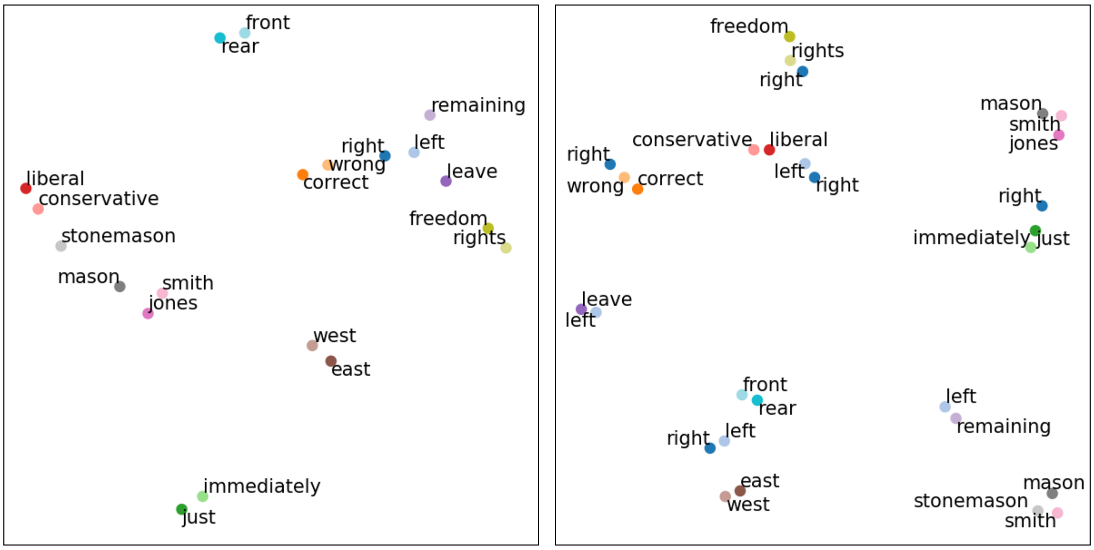
Much work in NLP has been dedicated to vector representations of words, but it has been recognized since as early as (Schütze, 1998) that such representations fail to capture the polysemous nature of many words, conflating their multiple senses into a single point in semantic space. There have been several attempts at embedding individual word senses to avoid this issue, termed the "meaning conflation deficiency" by Camacho-Collados and Pilehvar (2018) in their survey on the area.

We propose PolyLM, an unsupervised sense embedding model which is effective and easy to apply to downstream tasks. PolyLM can be thought of as both a (masked) language model and a *sense* model, as it calculates a probability distribution both over words and word senses at masked positions. The formulation is derived from two observations about word senses: firstly, that the probability of a word occurring in a given context is equal to the sum of the probabilities of its individual senses occurring; and secondly, that for a given occurrence of a word, one of its senses tends to be much more plausible in the context than the others.

There are several reasons for the interest in sense representations. The first is the downsides associated with the meaning conflation deficiency. Word embedding models can have difficulty distinguishing which sense of an ambiguous word applies in a given context (Yaghoobzadeh and Schütze, 2016). Additionally, homonymy and polysemy cause distortion in word embeddings: for instance, we would find the unrelated words *left* and *wrong* unreasonably close in the vector space due to their similarity to two different senses of the word *right*, an effect noted by Neelakantan et al. (2014) and illustrated in Figure 1. Intuitively we would expect that sense embedding models could gain superior semantic understanding by avoiding these problems.

In addition to well-established applications for sense representations such as word sense disambiguation (WSD) and induction (WSI), another interesting use case is the automatic construction of lexical resources (Neale, 2018). While there are existing human-curated word sense inventories for English such as such as WordNet (Miller, 1995), these are expensive to create and are unavailable for most languages. Panchenko (2016) showed that sense embeddings learned using the model of Bartunov et al. (2016) could be linked with word senses contained in BabelNet (Navigli and Ponzetto, 2012) with a reasonable degree of precision, although the mapping struggled with recall. PolyLM represents a significant advance over Bartunov et al.'s in terms of WSI performance, so

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 563–574
April 19 - 23, 2021.



(a) Word embeddings

(b) Sense embeddings

Figure 1: An illustration of the meaning conflation deficiency, showing selected word and sense embeddings learned by PolyLM visualized using t-SNE (Maaten and Hinton, 2008) and adjustText (Flyamer, 2017). Sense embeddings were learned by training PolyLM$_{\text{SMALL}}$ with the standard 8 senses per word; word embeddings were learned by training PolyLM$_{\text{SMALL}}$, but with a single sense per word. Note that both models were trained on unlemmatized data, unlike those used in the WSI experiments. The occurrence of closely related polysemous words nearby in the word embedding space (i.e. *left* and *right*) causes unrelated words to be closer together (e.g. *left* and *wrong*) and related words to be further apart (e.g. *right* and *east*) than they otherwise would be. The use of sense embeddings avoids such distortion. PolyLM is capable of detecting comparatively rare word senses, such as the political senses of *left* and *right*, and the use of *smith* and *mason* to refer to tradespeople.

it seems reasonable to imagine that this approach to lexical resource construction might now be more feasible.

The emergence of contextualized models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) has had a tremendous impact on the area of semantic representation. Rather than representing words using a single embedding, or even a set of sense embeddings, these models allow words to be represented using an infinite set of possible embeddings depending on the context. This approach has been very effective across NLP and many state-of-the-art systems incorporate contextualized models, including systems for WSD and WSI. The success of contextualized models raises the question of whether there is still value in learning discrete sense representations.

However, contextualized models still rely on word embeddings, and are therefore subject to the meaning conflation deficiency. Furthermore it could be argued that it is inefficient to have the same representation size for all words regardless of how diverse their range of senses is. Another drawback is that before they can be applied to word sense-related tasks, an adaptation step such as clustering to induce discrete senses or fine-tuning is generally required, which is often expensive in terms of both research and compute time.

The contributions of this paper can be summarized as follows:

- We propose PolyLM, an end-to-end, unsupervised neural sense embedding model derived from two simple assumptions about word senses. We demonstrate that PolyLM learns senses which correspond well to human notions by showing that it performs well at WSI.
- PolyLM is flexible in that it can use any "contextualizer" (a useful term coined by Liu et al. (2019)), so it will remain relevant as contextualization techniques improve.
- We reduce the effect of the meaning conflation deficiency by disambiguating word senses at the input with a neural "disambiguation layer." We show that good performance on WSI can be achieved using the output of this layer alone, suggesting that it could be a useful component in many neural networks for language understanding.

2 Related Work

One of the first works in unsupervised learning of sense representations was by Schütze (1998), who proposed a two-step process, where vector representations are first derived for each context containing an ambiguous word, and these are then clustered into a pre-defined number of groups. Huang et al. (2012) added a third step, where after sense-labeling each word according to its context cluster, sense representations are learned through neural language modeling.

A number of later approaches employed a joint training approach, where sense labeling and sense representation learning happen in parallel. Neelakantan et al. (2014), Li and Jurafsky (2015) and Bartunov et al. (2016) each proposed multi-sense variants of the Skip-Gram model (Mikolov et al., 2013). Various approaches were tried for determining the number of senses per word: for instance, Li and Jurafsky and Bartunov et al. used Chinese Restaurant Processes and Dirichlet Processes respectively to automatically learn an appropriate number of senses for each word.

Many joint training approaches have the disadvantage that they create ambiguity in the context representation by representing context words with word embeddings in order to avoid considering the exponential number of possible sense labelings for the context. Qiu et al. (2016) and Lee and Chen (2017) propose purely sense-based approaches which can sense-label the input efficiently.

Arora et al. (2018) took a novel approach to the problem of learning word senses, demonstrating that the embedding learned by traditional techniques for an ambiguous word tends to be very close to a linear combination of the hypothetical vectors corresponding to its individual senses. They proposed a method for recovering the underlying sense vectors and coefficients, and evaluated their system on WSI.

Since the emergence of contextualized models, there have been a number of other systems which have exploited their powerful semantic representations for specific tasks such as word sense disambiguation (Huang et al., 2019; Vial et al., 2019) and induction (Amrami and Goldberg, 2018, 2019), however none of these methods creates explicit sense embeddings.

3 PolyLM

3.1 Overview

Consider a typical neural language model. Each word w in a vocabulary V is assigned a single embedding, resulting in an embedding matrix $M \in \mathbb{R}^{|V|\times d}$, where d is the embedding dimensionality. The probability of w occurring in a context c is estimated as

$$\mathbb{P}(w \mid c) = \big[\text{softmax}\big(M\boldsymbol{y}(c) + \boldsymbol{a}\big)\big]_w, \quad (1)$$

where $\boldsymbol{y}(c) \in \mathbb{R}^d$ is a vector representation of c and $\boldsymbol{a} \in \mathbb{R}^{|V|}$ is a trainable bias vector. In BERT (Devlin et al., 2019) for instance, $\boldsymbol{y}(c)$ corresponds to the final output of multiple Transformer encoder layers (Vaswani et al., 2017).

Now suppose that for each $w \in V$, there is a corresponding set S_w of *sememes*, or senses which w can have. For instance, intuitively we might have $S_{\text{rock}} = \{\text{rock:stone, rock:musical genre, rock:shake}\}$. We assume that the S_w are disjoint, i.e. $S_w \cap S_{w'} = \emptyset$ whenever $w \neq w'$, and we define the full sense inventory $S = \bigcup_{w\in V} S_w$.

Context induces specific senses for the words it contains. Thus a passage of text can be thought of as a sequence of sememes as well as a sequence of words. The first observation underlying PolyLM is that the probability of a word w occurring in a context c is equal to the sum of the probabilities of w's component sememes occurring in the context, i.e.

$$\mathbb{P}(w \mid c) = \sum_{s\in S_w} \mathbb{P}(s \mid c). \quad (2)$$

We wish to learn representations for individual senses, and so we assign an embedding to each sememe in our sense inventory, resulting in a matrix E with dimension $|S| \times d$ and bias vector $\boldsymbol{b}$ of dimension $|S|$. Note that this assumes that we know the number of senses of each word *a priori*, an assumption whose consequences we discuss later. Following Eq. 1, we define the vector $\boldsymbol{p}(c) \in \mathbb{R}^{|S|}$ of sememe probabilities in a context c as

$$\boldsymbol{p}(c) = \text{softmax}\big(E\boldsymbol{x}(c) + \boldsymbol{b}\big). \quad (3)$$

Considering Eq. 2, we have

$$\mathbb{P}(w \mid c) = \sum_{s\in S_w} p(c)_s, \quad (4)$$

allowing us to formulate the problem of learning sense representations with a language modeling objective.

PolyLM is constructed from three components: the input layer, which represents the input tokens as aggregates of their sense embeddings, the disambiguation layer, which attempts to determine the contextually appropriate sense embeddings for the input, and the prediction layer, which implements the language modeling objective.

We adopt the masked language modeling (MLM) task used for training BERT. When training, we select a subset $T \subset \{1, 2, ..., n\}$ of the tokens in the input sequence as targets for prediction, and produce a masked version $c' = w'_1, w'_2, ..., w'_n$ of the original sequence $c = w_1, w_2, ..., w_n$ as follows: 15% of tokens are chosen at random as targets, of which 80% are replaced with a special [MASK] token, 10% are replaced with a random token, and 10% are left unchanged.

3.2 Input Layer

We define a contextualizer to be a function which maps a sequence of input representations $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n \in \mathbb{R}^d$ to a corresponding sequence of output representations $\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_n \in \mathbb{R}^d$. Recurrent Neural Networks and Transformer architectures are both commonly used as contextualizers for language modeling. Typically the input representations are drawn from an embedding matrix $I \in \mathbb{R}^{|V| \times d}$. It has become common (e.g. BERT) to set I equal to O, the embedding matrix used at the language modeling output, as recommended by Press and Wolf (2017), and thus have a single embedding matrix E.

The issue of input representation poses a problem for our model. Our output embeddings $E \in \mathbb{R}^{|S| \times d}$ correspond to sememes. We cannot straightforwardly tie our input and output embeddings as Press and Wolf suggest, because we receive words rather than sememes as input. We solve this problem by setting the input representation of a word to be a convex combination of the representations of its sememes, i.e.

$$\boldsymbol{x}(w) = \sum_{s \in S_w} \lambda_{ws} \boldsymbol{e}_s, \qquad (5)$$

where $\boldsymbol{e}_s$ is the row of E corresponding to sememe s, and $\boldsymbol{\lambda}_w$ is a learnable weight vector with the properties that $\sum_{s \in S_w} \lambda_{ws} = 1$ and $\boldsymbol{\lambda}_w \geq \boldsymbol{0}$ (in practice, $\boldsymbol{\lambda}_w$ is the softmax of an underlying, unconstrained variable vector).

3.3 Disambiguation Layer

The disambiguation layer attempts to infer the contextually appropriate sememe embeddings for the input based on the conflated representations from the input layer.

Representations $\boldsymbol{x}(w'_1), \boldsymbol{x}(w'_2), ..., \boldsymbol{x}(w'_n)$ of c', calculated according to Eq. 5, are fed into a contextualizer instance C^D, which outputs representations $\boldsymbol{y}_1^D(c'), \boldsymbol{y}_2^D(c'), ..., \boldsymbol{y}_n^D(c')$. We use these representations to calculate a probability distribution over each sense of the tokens in the input:

$$\boldsymbol{q}_i^D(c') = \text{softmax}\big(E^{(w'_i)} \boldsymbol{y}_i^D(c') + \boldsymbol{b}^{(w'_i)}\big), \qquad (6)$$

where $E^{(w'_i)}$ is a submatrix of E containing only the rows corresponding to senses of token w'_i, and similarly $\boldsymbol{b}^{(w'_i)}$ is a subvector of a learnable bias vector $\boldsymbol{b} \in \mathbb{R}^{|S|}$. In other terms,

$$q_{is}^D(c') = \frac{e^{\boldsymbol{e}_s^\top \boldsymbol{y}_i^D(c') + b_s}}{\sum_{s' \in S_{w'_i}} e^{\boldsymbol{e}_{s'}^\top \boldsymbol{y}_i^D(c') + b_{s'}}}, \qquad (7)$$

where $s \in S_{w'_i}$. $q_{is}^D(c')$ corresponds to the probability that the ith token in sequence c' has sense s.

The disambiguated representation of a token could simply be its highest-probability sememe embedding in the context, but to allow gradients to flow through the disambiguation layer, we take the sum of the sememe embeddings weighted by their probabilities:

$$\boldsymbol{x}_i^P(c') = \sum_{s \in S_{w'_i}} q_{is}^D(c') \boldsymbol{e}_s. \qquad (8)$$

3.4 Prediction Layer

The prediction layer maps a sequence of disambiguated input representations onto a corresponding set of output representations, and from each output representation estimates the probability of every sememe in the sense inventory occurring at the corresponding position of the sequence.

Disambiguated representations $\boldsymbol{x}_1^P(c'), \boldsymbol{x}_2^P(c'), ..., \boldsymbol{x}_n^P(c')$ are fed into another contextualizer instance C^P, which returns output representations $\boldsymbol{y}_1^P(c'), \boldsymbol{y}_2^P(c'), ..., \boldsymbol{y}_n^P(c')$. These are used to calculate a probability distribution over the entire sense inventory, as prescribed by Eq. 3:

$$\boldsymbol{p}_i(c') = \text{softmax}(E \boldsymbol{y}_i^P(c') + \boldsymbol{b}). \qquad (9)$$

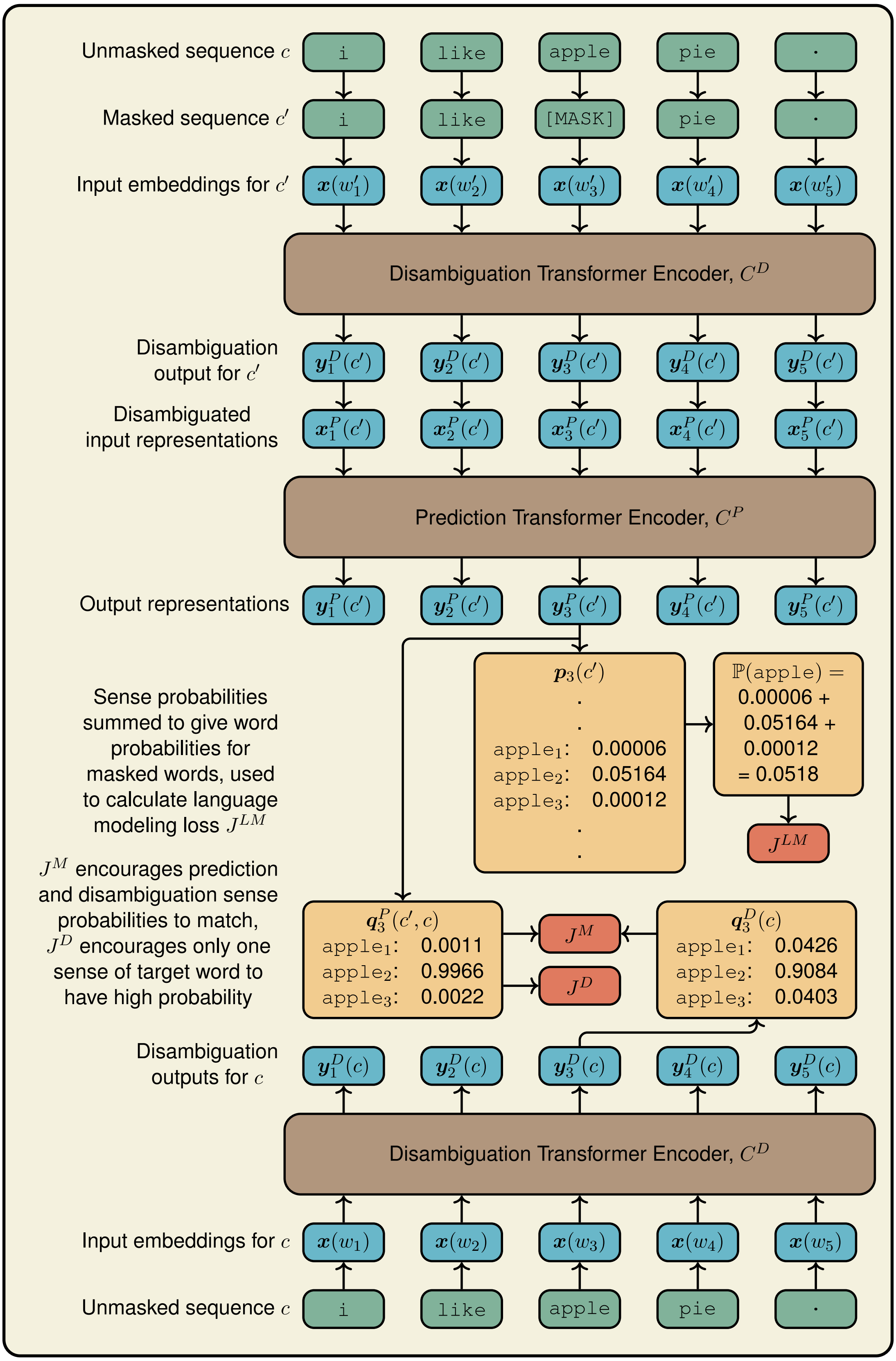


Figure 2: Architecture diagram for PolyLM when training, illustrated on the sentence “I like apple pie.”, where the word “apple” is chosen as a target and masked (note that “apple” is ambiguous when tokens are lower-cased, as it may refer to a fruit or a technology company). At inference time, the bottom components (up to and including $\boldsymbol{q}^D(c)$) do not need to be evaluated, and the sequence may not be masked at the input.

We define an additional set of probabilities $\boldsymbol{q}^P$ analogous to $\boldsymbol{q}^D$ defined in Eq. 6:

$$\boldsymbol{q}_i^P(c', c) = \operatorname{softmax}\big(E^{(w_i)}\boldsymbol{y}_i^P(c') + \boldsymbol{b}^{(w_i)}\big). \tag{10}$$

$\boldsymbol{q}_i^P$ takes both c' and the unmasked sequence c as arguments because we are interested in the sense probabilities of the words w_i that actually occurred. $\boldsymbol{q}_i^P$ will be used later for defining the loss function and is useful for downstream tasks.

3.5 Loss Function

We seek to minimize a loss function J with three components, each of which is explained below:

$$\begin{aligned} J(c, c', T) = J^{LM}(c, c', T) &+ \\ J^D(c, c', T) &+ \\ J^M(c, c', T)& \end{aligned} \tag{11}$$

3.5.1 Language Modeling Loss

The language modeling loss J^{LM} is defined as the mean negative log likelihood of the target tokens occurring:

$$\begin{aligned} &J^{LM}(c, c', T) \\ &= \frac{-1}{|T|}\sum_{i\in T}\log\hat{\mathbb{P}}(w_i \mid c') \qquad (12) \\ &= \frac{-1}{|T|}\sum_{i\in T}\log\sum_{s\in S_{w_i}}\hat{\mathbb{P}}(\text{sememe } i \text{ is } s \mid c') \qquad (13) \\ &= \frac{-1}{|T|}\sum_{i\in T}\log\sum_{s\in S_{w_i}}p_{is}(c'), \qquad (14) \end{aligned}$$

where $\boldsymbol{p}_i$ is as defined in Eq. 9.

3.5.2 Distinctness Loss

Recall that we assume in advance a number of senses for each word. In practice we guess a relatively high number to avoid missing senses. When we overestimate the number of senses, we find that two different sense embeddings for a word converge to essentially the same meaning. The aim of the distinctness loss is to ensure that each sense has a distinct meaning, and to "kill off" superfluous senses by causing them to have very low probability in all contexts.

The second key observation of PolyLM is that if the sememes corresponding to a word w are distinct, then in contexts where w occurs, we would expect one of these sememes to have a high estimated probability of occurring, and the rest to have a low probability. The distinctness loss, given by,

$$J^D(c, c', T) = \frac{-1}{r|T|}\sum_{i\in T}\log\sum_{s\in S_{w_i}}\big(q_{is}^P(c', c)\big)^r, \tag{15}$$

with hyperparameter $r > 1$, encourages this separation to occur. A full justification is given in Appendix A.

3.5.3 Match Loss

Without extra supervision, the disambiguation layer tends to very quickly allocate almost all of the probability mass for a word to a single one of its senses. This appears to be due to a "rich get richer" effect in Eq. 8, where the sense embedding with the highest weight has larger gradients associated with it.

A more reliable source of sense probabilities is the output of the prediction layer, as this is more closely associated with the ground truth. Therefore we encourage the disambiguation sense probabilities $\boldsymbol{q}^D$ to be similar to the prediction sense probabilities $\boldsymbol{q}^P$ by adding a sense probability "match loss," which is proportional to the cosine similarity between $\boldsymbol{q}^D$ and $\boldsymbol{q}^P$.

Because $\boldsymbol{q}_i^D(c')$ is meaningless when token i is replaced with [MASK], when calculating the match loss we evaluate the disambiguation layer on the unmasked sequence (shown with bottom-up arrows in Figure 2), obtaining $\boldsymbol{q}_i^D(c)$. The match loss is defined as

$$J^M(c, c', T) = \frac{-\lambda^M}{|T|}\sum_{i\in T}\frac{\boldsymbol{q}_i^D \cdot \boldsymbol{q}_i^P}{\|\boldsymbol{q}_i^D\|\|\boldsymbol{q}_i^P\|}, \tag{16}$$

where $\boldsymbol{q}_i^D$ and $\boldsymbol{q}_i^P$ are shorthand for $\boldsymbol{q}_i^D(c)$ and $\boldsymbol{q}_i^P(c', c)$ respectively, and λ^M is a hyperparameter.

As we wish the disambiguation layer to learn from the prediction layer rather than the other way around, we do not allow gradients from the match loss to propagate through $\boldsymbol{q}_i^P$.

3.6 Details and Parameters

3.6.1 Preprocessing

To avoid the issue of how to represent a word's sense when it is broken into sub-word level tokens, our vocabulary consists of whole-word tokens. However the WSI tasks on which we evaluate our model operate on the lemma level, so we lemmatize our training corpus as described in Appendix B. The vocabulary consists of the ~86K tokens appearing more than 500 times in our training corpus, which like BERT's consists of English Wikipedia + BookCorpus (Zhu et al., 2015). All tokens are lower-cased.

3.6.2 Contextualizers

One of the advantages of PolyLM is that it can be used with any type of contextualizer - note however that we must train our contextualizers together with the rest of the model rather than using pretrained contextualizer instances, because their word embedding matrix would not match our sense embedding matrix. In this paper we present results where the disambiguation and prediction contextualizers C^D and C^P use BERT's implementation of the Transformer encoder architecture.

3.6.3 Parameters

To keep the total number of embeddings reasonable, we allow only the ~10,000 tokens which occur more than 20,000 times in the training corpus, or appear as focuses in the evaluation datasets, to have multiple senses. Specifically, we assign these tokens a fixed number of $k = 8$ embeddings, and other tokens a single embedding. Since according to Zipf's law (Zipf, 1950), it is the most frequent words which tend to have the most senses, we expect not to miss too many senses by assuming that infrequent words are monosemous. We leave the investigation of more sophisticated methods for pre-allocating or dynamically updating the number of senses for each token for future work.

We train two PolyLM models of different sizes, PolyLM$_{\text{SMALL}}$ and PolyLM$_{\text{BASE}}$. Due to the prohibitive computational cost of training a model of BERT$_{\text{LARGE}}$'s size, we use significantly smaller dimensions, as shown in Table 1.

Models were trained over 6,000,000 batches consisting of 32 sequences of length 128 using the Adam optimizer (Kingma and Ba, 2014). The learning rate was increased linearly from 0 to 3e-5 over the first 10,000 batches, and then reduced linearly back to zero over the remaining batches. The hyperparameters λ^M and r specific to PolyLM's loss function were first increased linearly and then left constant, λ^M from 0 to 0.1 over the first 1,000,000 batches, and r from 1.0 to 1.5 over the first 2,000,000 batches.

It is important for r to be gradually increased in this manner because if r is large initially, then the effect of the distinctness loss reduces the diversity of the senses learned. On the other hand, increasing r too slowly seems to be detrimental to the senses' distinctness.

4 Experiments

Word sense induction (WSI) is the task of inferring the senses of a word in an unsupervised manner. This is precisely the aim of our method, and so is an ideal test task. We evaluate PolyLM on two WSI datasets, SemEval-2010 Task 14 (Manandhar et al., 2010) and SemEval-2013 Task 13 (Jurgens and Klapaftis, 2013). Both datasets consist of passages containing one of a set of polysemous focus words. The occurrences of the focus words in the test set have been sense-labeled by human annotators according to a reference sense inventory.

In the SemEval-2010 dataset, each instance is labeled with a single sense, whereas in the SemEval-2013 dataset an instance may be labeled with several relevant senses, each with a corresponding weight denoting its degree of applicability in the context.

Performance on SemEval-2010 is measured using paired F-Score (F-S) and V-Measure (V-M), and on SemEval-2013 using Fuzzy B-Cubed (FBC) and Fuzzy Normalized Mutual Information (FNMI). Overall performance on each task (**AVG**) is typically defined as the geometric mean of its two sub-metrics.

Currently, the best performing system on both datasets is that of Amrami and Goldberg (2019). Their system uses the idea of *substitute vectors*, first devised by Başkaya et al. (2013). For each instance, a set of most likely words that could have occurred instead of the focus word is obtained from the output of a language model. These sets are then clustered, and each cluster is taken to correspond to a different sense of the focus word. Amrami and Goldberg use BERT$_{\text{LARGE}}$ as their language model.

PolyLM can be used for WSI without any further training. For the SemEval-2010 dataset, each instance c is labeled with the sense of the focus word w_i which has the highest predicted probability, i.e. $\text{argmax}_{s \in S_{w_i}} \boldsymbol{q}^P_{is}(c', c)$, where c' is formed from c by replacing w_i with [MASK]. For SemEval-2013, we consider a sense applicable if it has a predicted probability $q^P_{is}(c', c) > p^{\text{thresh}}$, and the weight assigned to each applicable sense is its probability $q^P_{is}(c', c)$. We arbitrarily set p^{thresh} to 0.2.

Model	d	**Filter size**	**No. attn. heads**	**No. layers**	**Seq. len.**	**Vocab size**	**No. embeddings**	**Total params**
PolyLM$_{\text{SMALL}}$	128	512	8	4 (C^D), 8 (C^P)	128	86K	157K	24M
PolyLM$_{\text{BASE}}$	256	1024	8	4 (C^D), 12 (C^P)	128	86K	157K	54M
BERT$_{\text{LARGE}}$	1024	4096	16	24	512	30K	30K	340M

Table 1: Parameters of PolyLM and BERT$_{\text{LARGE}}$.

System	**Version**	**SemEval-2010**			**SemEval-2013**		
		F-S	V-M	**AVG**	FBC	FNMI	**AVG**
Amrami and Goldberg (2019)	BERT$_{\text{LARGE}}$	**71.3**	40.4	**53.6**	64.0	21.4	37.0
AutoSense (Amplayo et al., 2019)		62.9	10.1	25.2	61.7	8.0	22.2
PolyLM[†]	BASE	65.8	**40.5**	51.6	**64.8**	**23.0**	**38.6**
	SMALL	65.6	35.7	48.4	64.5	18.5	34.5
Qiu et al. (2016)[†]		-	-	-	56.9	6.7	19.5
SE-WSI-fix-cmp (Song et al., 2016)[†]		54.3	16.3	29.8	-	-	-
AdaGram (Bartunov et al., 2016)[†]		43.9	20.0	29.6	13.2	8.9	10.8
Arora et al. (2018)[†]	$k = 5$	46.4	11.5	23.1	-	-	-

Table 2: Comparison of sense embedding models and WSI-specific techniques on the SemEval 2010 and 2013 WSI tasks. SE-WSI-fix-cmp is based on Neelakantan et al. (2014)'s MSSG model. [†] - models which obtain explicit sense embeddings.

Description	**SemEval-2010**			**SemEval-2013**		
	F-S	V-M	**AVG**	FBC	FNMI	**AVG**
PolyLM$_{\text{SMALL}}$	**65.6**	**35.7**	**48.4**	**64.5**	**18.5**	**34.5**
No distinctness loss	53.5	33.4	42.3	57.4	16.3	30.5
No disambiguation layer	64.9	25.5	40.6	**64.5**	17.5	33.6
Disambiguation layer only	63.6	29.3	43.2	62.7	15.7	31.4

Table 3: PolyLM ablation study.

Results are shown in Table 2. Both PolyLM models comprehensively outperform previous sense embedding methods. PolyLM$_{\text{BASE}}$ and Amrami and Goldberg's system slightly outperform each other on one dataset each, suggesting similar overall proficiency at WSI. However it is worth noting that the BERT$_{\text{LARGE}}$ language model used by Amrami and Goldberg has more than six times as many parameters as PolyLM$_{\text{BASE}}$ and is much more computationally expensive to train and run.

PolyLM scales well for the sizes tested, with PolyLM$_{\text{BASE}}$ outperforming PolyLM$_{\text{SMALL}}$ by 3.2 and 4.1 points in **AVG** score on the two datasets with a 2.25x increase in the number of parameters. Even if further increases in model dimensions yielded much smaller improvements in performance, it seems likely that a PolyLM model of BERT$_{\text{LARGE}}$'s 340 million parameter size would achieve results significantly better than those of Amrami and Goldberg (2019).

4.1 Ablation Study

We test three alternative configurations against PolyLM$_{\text{SMALL}}$: one where the distinctness loss term is removed from the objective ("no distinctness loss"), one where the disambiguation layer is removed ("no disambiguation layer"), and one where the disambiguation sense probabilities $\boldsymbol{q}^D$ are used in place of $\boldsymbol{q}^P$ when performing WSI ("disambiguation layer only"). Note that the first two configurations require new models to be trained, whereas the last simply uses PolyLM$_{\text{SMALL}}$ in a different way. Results are shown in Table 3.

The use of the distinctness loss has a big impact on model performance, while the disambiguation layer is somewhat less important but still useful. The model still performs surprisingly well when the disambiguation rather than the prediction sense probabilities are used; these are the output of only four Transformer layers and hence are much cheaper to compute. This suggests that it might be practical to add the disambiguation layer at the input of various neural NLP models to improve their understanding of polysemy.

5 Conclusions

PolyLM is a novel model of polysemy based on two assumptions about word senses: firstly, that the probability of a word occurring in a context is equal to the sum of its individual senses occurring, as expressed by the language modeling loss; and secondly, that generally only one sense of a word ought to have a high probability of occurring in a given context, as expressed by the distinctness loss. PolyLM does indeed learn word senses which correspond well to human notions, as demonstrated by its performance on word sense induction, which matches that of the previous state-of-the-art system despite having 6 times fewer parameters. It can be easily applied to many word-sense related tasks, as it generates a probability distribution over the senses of each word in the input text. It is not specific to any one contextualizer and so can be improved as contextualizers improve.

Acknowledgements

Felipe Bravo-Marquez was funded by ANID FONDECYT grant 11200290, U-Inicia VID Project UI-004/20 and ANID - Millennium Science Initiative Program - Code ICN17_002.

References

Reinald Kim Amplayo, Seung-won Hwang, and Min Song. 2019. AutoSense model for word sense induction. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6212–6219.

Asaf Amrami and Yoav Goldberg. 2018. Word sense induction with neural biLM and symmetric patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium. Association for Computational Linguistics.

Asaf Amrami and Yoav Goldberg. 2019. Towards better substitution-based word sense induction.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495.

Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 130–138, Cadiz, Spain. PMLR.

Osman Başkaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. AI-KU: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 300–306, Atlanta, Georgia, USA. Association for Computational Linguistics.

Jose Camacho-Collados and Mohammad Taher Pilehvar. 2018. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788.

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

Ilya Flyamer. 2017. adjustText.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3507–3512, Hong Kong, China. Association for Computational Linguistics.

David Jurgens and Ioannis Klapaftis. 2013. SemEval-2013 task 13: Word sense induction for graded and non-graded senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299, Atlanta, Georgia, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Guang-He Lee and Yun-Nung Chen. 2017. MUSE: Modularizing unsupervised sense embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 327–337, Copenhagen, Denmark. Association for Computational Linguistics.

Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1722–1732, Lisbon, Portugal. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605.

Suresh Manandhar, Ioannis Klapaftis, Dmitriy Dligach, and Sameer Pradhan. 2010. SemEval-2010 task 14: Word sense induction &disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68, Uppsala, Sweden. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Steven Neale. 2018. A survey on automatically-constructed WordNets and their evaluation: Lexical and word embedding-based approaches. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.

Alexander Panchenko. 2016. Best of both worlds: Making word sense embeddings interpretable. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2649–2655, Portorož, Slovenia. European Language Resources Association (ELRA).

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Lin Qiu, Kewei Tu, and Yong Yu. 2016. Context-dependent sense embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 183–191, Austin, Texas. Association for Computational Linguistics.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Linfeng Song, Zhiguo Wang, Haitao Mi, and Daniel Gildea. 2016. Sense embedding learning for word sense induction. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 85–90, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2019. Sense vocabulary compression through the semantic knowledge of WordNet for neural word sense disambiguation. In *Proceedings of the Tenth Global Wordnet Conference*, pages 108–117, Poland.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2016. Intrinsic subspace evaluation of word embedding representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 236–246, Berlin, Germany. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 19–27, Washington, DC, USA. IEEE Computer Society.

George K. Zipf. 1950. Human behavior and the principle of least effort. *Journal of Clinical Psychology*, 6(3):306–306.

A Justification of the Distinctness Loss

Consider the derivative of the language modeling loss for one particular target position $i \in T$ with respect to the pre-softmax scores $\boldsymbol{e}_k^\top \boldsymbol{y}_i^P + b_k$ of the target word w_i's sense embeddings $k \in S_{w_i}$. For brevity, we define $y_k = \boldsymbol{e}_k^\top \boldsymbol{y}_i^P + b_k$.

$$
\begin{aligned}
&-\frac{\partial}{\partial y_k} J^{LM}(c, c', \{i\}) \\
&= \frac{\partial}{\partial y_k} \log \sum_{s \in S_{w_i}} \left[\mathrm{softmax}(E\boldsymbol{y}_i^P + \boldsymbol{b})\right]_s \\
&= \frac{\partial}{\partial y_k} \log \sum_{s \in S_{w_i}} \frac{e^{y_s}}{\sum_{s' \in S} e^{y_{s'}}} \\
&= \frac{\partial}{\partial y_k} \log \frac{\sum_{s \in S_{w_i}} e^{y_s}}{\sum_{s \in S_{w_i}} e^{y_s} + \sum_{s \in S \setminus S_{w_i}} e^{y_s}} \\
&= \frac{\partial}{\partial y_k} \log \frac{\sum_{s \in S_{w_i}} e^{y_s}}{\sum_{s \in S_{w_i}} e^{y_s} + C},
\end{aligned}
$$

where $C = \sum_{s \in S \setminus S_{w_i}} e^{y_s}$,

$$
\begin{aligned}
&= \frac{\partial}{\partial y_k} \left(\log \sum_{s \in S_{w_i}} e^{y_s} - \log \left(\sum_{s \in S_{w_i}} e^{y_s} + C \right) \right) \\
&= \frac{\frac{\partial}{\partial y_k} \sum_{s \in S_{w_i}} e^{y_s}}{\sum_{s \in S_{w_i}} e^{y_s}} - \frac{\frac{\partial}{\partial y_k} (\sum_{s \in S_{w_i}} e^{y_s} + C)}{\sum_{s \in S_{w_i}} e^{y_s} + C} \\
&= \frac{e^{y_k}}{\sum_{s \in S_{w_i}} e^{y_s}} - \frac{e^{y_k}}{\sum_{s \in S_{w_i}} e^{y_s} + C} \\
&= \frac{e^{y_k}}{\sum_{s \in S_{w_i}} e^{y_s}} - \frac{e^{y_k}}{\sum_{s \in S} e^{y_s}} \\
&= q_{ik}^P(c', c) - p_{ik}(c).
\end{aligned}
$$

Since $q_{ik}^P > p_{ik}$, $\frac{\partial}{\partial y_k} J^{LM}(c, c', \{i\})$ will always be negative, meaning that every sense embedding for the target word will always move towards the contextualized representation $\boldsymbol{y}_i^P$. This is undesirable, because it means that even senses which are irrelevant in a context will receive a positive update.

Now consider the derivatives of the distinctness loss:

$$
\begin{aligned}
&-\frac{\partial}{\partial x_y} J^D(c, c', \{i\}) \\
&= \frac{\partial}{\partial y_k} \frac{1}{r} \log \sum_{s \in S_{w_i}} \left(q_{is}^P(c', c)\right)^r \\
&= \frac{1}{r} \frac{\partial}{\partial y_k} \log \sum_{s \in S_{w_i}} \left(\frac{e^{y_s}}{\sum_{s' \in S_{w_i}} e^{y_{s'}}} \right)^r \\
&= \frac{1}{r} \frac{\partial}{\partial y_k} \log \sum_{s \in S_{w_i}} \frac{e^{r y_s}}{(\sum_{s' \in S_{w_i}} e^{y_{s'}})^r} \\
&= \frac{1}{r} \frac{\partial}{\partial y_k} \log \frac{\sum_{s \in S_{w_i}} e^{r y_s}}{(\sum_{s \in S_{w_i}} e^{y_s})^r} \\
&= \frac{1}{r} \frac{\partial}{\partial y_k} \left(\log \sum_{s \in S_{w_i}} e^{r y_s} - \log \left(\sum_{s \in S_{w_i}} e^{y_s} \right)^r \right) \\
&= \frac{1}{r} \left(\frac{\frac{\partial}{\partial y_k} \sum_{s \in S_{w_i}} e^{r y_s}}{\sum_{s \in S_{w_i}} e^{r y_s}} - \frac{r \frac{\partial}{\partial y_k} \sum_{s \in S_{w_i}} e^{y_s}}{\sum_{s \in S_{w_i}} e^{y_s}} \right) \\
&= \frac{1}{r} \left(\frac{r e^{r y_k}}{\sum_{s \in S_{w_i}} e^{r y_s}} - \frac{r e^{y_k}}{\sum_{s \in S_{w_i}} e^{y_s}} \right) \\
&= \frac{e^{r y_k}}{\sum_{s \in S_{w_i}} e^{r y_s}} - q_{ik}^P(c', c).
\end{aligned}
$$

When $r > 1$, $\frac{e^{r y_k}}{\sum_{s \in S_{w_i}} e^{r y_s}}$ is a "sharpened" version of $q_{ik}^P(c', c)$: it is larger than q_{ik}^P when q_{ik}^P is large, and smaller when q_{ik}^P is small.

Now we have

$$
\begin{aligned}
&-\frac{\partial}{\partial y_k} \left(J^{LM}(c, c', \{i\}) + J^D(c, c', \{i\}) \right) \\
&= -\frac{\partial}{\partial y_k} J^{LM}(c, c', \{i\}) - \frac{\partial}{\partial y_k} J^D(c, c', \{i\}) \\
&= q_{ik}^P(c', c) - p_{ik}(c) + \frac{e^{r y_k}}{\sum_{s \in S_{w_i}} e^{r y_s}} - q_{ik}^P(c', c) \\
&= \frac{e^{r y_k}}{\sum_{s \in S_{w_i}} e^{r y_s}} - p_{ik}(c) \\
&= q_{ik}^{\text{sharp}}(c', c) - p_{ik}(c).
\end{aligned}
$$

Thus the addition of the distinctness loss results in even stronger reinforcement for senses which are highly applicable in the context, and even weaker (possibly negative) reinforcement for senses which are inapplicable. This encourages only one sense of a word to have high probability in a given context, as desired.

B Lemmatization

The training corpus and all text used for evaluation are lemmatized as follows: first, we perform part-of-speech (POS) tagging using Stanford CoreNLP's POS tagger (Manning et al., 2014). Any token with a tag associated with inflectional morphology in English (NNS, JJR, JJS, RBR, RBS, VBD, VBG, VBP, VBZ or VNB) is split into two separate tokens, its lemmatized form and a special token. There is a unique special token for each of the above tags except the pairs JJR and RBR (comparative adjectives and adverbs) and JJS and RBS (superlative adjective and adverbs), which share [COMP] and [SUP] tokens respectively.

Predicting Treatment Outcome from Patient Texts: The Case of Internet-Based Cognitive Behavioural Therapy

Evangelia Gogoulou[1] **Magnus Boman**[2] **Fehmi Ben Abdesslem**[1]
Nils Isacsson[3] **Viktor Kaldo**[3,4] **Magnus Sahlgren**[1]

[1] Research Institutes of Sweden (RISE)
[2]School of Electrical Engineering and Computer Science,
The Royal Institute of Technology(KTH) &
Department of Learning, Informatics, Management and Ethics, Karolinska Institutet
[3]Department of Clinical Neuroscience, Karolinska Institutet, &
Stockholm Health Care Services, Region Stockholm
[4]Faculty of Health and Life Sciences, Linnaeus University

`{evangelia.gogoulou, fehmi.ben.abdesslem, magnus.sahlgren}@ri.se`
`{mab@kth.se}`
`{nils.isacsson, viktor.kaldo}@ki.se`

Abstract

We investigate the feasibility of applying standard text categorisation methods to patient text in order to predict treatment outcome in Internet-based cognitive behavioural therapy. The data set is unique in its detail and size for regular care for depression, social anxiety, and panic disorder. Our results indicate that there is a signal in the depression data, albeit a weak one. We also perform terminological and sentiment analysis, which confirm those results.

1 Introduction

The Internet Psychiatry Clinic in Stockholm offers Internet-based Cognitive Behavioral Therapy (ICBT) for depression, social anxiety, and panic disorder with documented significant treatment effects (Titov, 2018). The treatment is delivered on a secure online platform that enables patients to submit answers to weekly questionnaires of primary symptoms, depression, and suicidal ideation, as well as interactive worksheets. Because patients normally are not in contact with the psychologists away from keyboard during treatment, the infrastructure platform logs a large amount of data for each patient, yielding a unique data set. Because Sweden was the first country to introduce this kind of ICBT, the clinical experience is considerable (Hedman et al., 2012), and experimental research of the kind reported in this paper rests on well-established procedures for sensemaking of machine learning at the clinic (Boman and Sanches, 2015). The overarching problem is how to be able to offer accelerated care, via identification of those predicted not to succeed with treatment, or at least not with enough decrease in symptoms, and giving these people more attention, as early as possible.

We here investigate the possibility of predicting outcome based *only* on patient texts, in various points in treatment. We formulate this as a unimodal binary text categorisation problem, where the categories are simplified to either success or failure. In this initial feasibility study, we evaluate the performance of a number of text categorisation methods for predicting treatment outcome. Even if our methodology is mostly exploratory, we also relate terminological and sentiment analysis of patients' text to the treatment outcome.

2 Related work

NLP has previously been used for predicting the outcome of psychiatric treatment in a number of encouraging studies. Stylometric analysis on patients' texts from the data set under study indicates variation through time in the sentiment sign (positive or negative) in patient text messages (Boman et al., 2019). Althoff et al. (2016) suggest that more references to the future, references to other individuals, and positive conversation perspective are all positively related to the success of counseling conversations. Cohan et al. (2018) construct a

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 575–580
April 19 - 23, 2021.

mental health-related corpus with Reddit posts and analyse users' language to identify self-reported diagnoses of nine different mental health conditions, including depression. Pestian et al. (2010) propose a classification model based on linguistic features, including part-of-speech tags and emotional concepts among others, that under some circumstances outperforms mental health professionals on identifying fake suicide notes. Gkotsis et al. (2016) study the usefulness of negation and affirmation in mental health records with the word *suicide* itself, while Song et al. (2020) identify statements related to suicidal behaviour in electronic health records, using a deep neural network. Trotzek et al. (2018) address the problem of early detection of depression using an ensemble of a Convolutional Neural Network and a classifier based on user-level linguistic metadata.

3 Data Set

The total data covers $6,821$ patients enrolled in a 12-week treatment for three different psychological disorders: Major Depressive Disorder (hereafter referred to as MDD, or depression), social anxiety, or panic disorder. Each treatment consists of weekly exercises, referred to as *homework reports*. For predicting treatment outcome, we focus on the depression treatment programme, which consists of 10 homework reports distributed over the 12 weeks. The homework contains questions about the progress made by the patient during treatment, such as "What is the most important thing you bring from module 1?" The questions can be both closed questions, where the patient selects pre-defined answers from a list, and open-ended questions, where the patient is invited to type in free text. The depression data covers $3,179$ patients. We remove patients who completed less than five homework reports (out of ten), patients without treatment outcome (who did not fill out the self-assessment at the end of the treatment), and empty texts. The resulting filtered data set contains $16,379$ texts from homework reports completed by $1,986$ patients. Each text is a concatenation of all answers written by a patient in one single weekly homework.

We also use patients' texts from the other treatments in order to enhance the domain pre-training for the embeddings. This data covers all $6,821$ patients participating in any of the three treatments, including $180,017$ free texts answers from the homework reports and $146,398$ direct messages, totalling almost 29 million $(28,993,089)$ tokens.

As target scores for predicting treatment outcome for the depression patients, we use the self-assessment MADRS-SR method (Fantino and Moore, 2009), which consists of ten questions scored between zero and six, and amounting to each patient having done between five and ten modules, yielding a total score from zero to 54. Therapists commonly use the cut-off score of 10 to partially define successful treatment. All diagnosed patients started the treatment above this score, and the treatment is defined as successful if the score drops below 10 at the end of the treatment (*remission*) (Hawley et al., 2002). When patients show a symptom reduction of at least 50%, this is defined as a *response*, and the treatment is also considered successful, even if the final score is above the cut-off value. In other cases, the treatment is considered a failure. Regarding the choice of 50% cut-off for the definition of the responder patients, it is supported by previous work (Karin et al., 2018), while the same value has been previously used by clinical researchers (Forsell et al., 2019). All patients are labelled as being subject to a treatment success or a treatment failure, as shown in Table 1.

Table 1: Patient labels in the prediction data set, based on the treatment outcome.

Outcome	Cases	Label	Total
Response	164	Success	1,075
Remission	111		
Response and remission	800		
No response, no remission	911	Failure	911

4 Methodology

We formulate the task of predicting outcome as a simple binary text classification problem, where the classes are either *success* or *failure*. Our goal is to assist clinicians in defining the best possible adaptive treatment strategy. Therefore, the class definitions used here are inherited from the treatment model employed at the clinic. We compare four different text representations: TF-IDF (Jones, 1972),[1] Word2Vec (Mikolov et al., 2013), FastText (Bojanowski et al., 2017), and Doc2Vec (Le and Mikolov, 2014), all of which are fed into a simple linear classifier trained with binary cross entropy loss using PyTorch.[2] All word embeddings are pro-

---

[1] `https://scikit-learn.org`

[2] `https://pytorch.org/`

duced with the Gensim implementation [3] and are pre-trained on patients' texts from all three treatments, as described in Section 3.

We use a naïve classification model that generates stratified predictions based on the class distributions as the baseline. The training set includes $1,588$ patients, with 860 labelled as success and 728 as failure, from the depression programme who have completed at least five weeks of homework reporting. All models are trained with 3-fold cross validation and early stopping, to avoid overfitting. After training, we evaluate each model on the test set, which consists of 398 patients (215 labelled as success and 183 as failure). Model performance is measured using the F1 score metric.

5 Results

The results from the classification experiment are presented in Table 2. The classification model based on TF-IDF consistently improves on the baseline, for all three values of the parameter #Homework reports. On the other hand, the models based on Word2Vec, FastText and Doc2Vec marginally reach this goal. Between all combinations of embeddings and number of homework reports considered, the TF-IDF model using one or three homework reports is the best performing model. Word2Vec exhibits better performance than FastText and Doc2Vec when the number of homework reports used is set to 3. Particularly remarkable is the relative F1 score difference between TF-IDF (number of homework reports equal to 1 or 3) and the rest of the models. This difference indicates the appropriateness of Bag-of-Words models in this experimental setup, compared to text representation methods that leverage semantic information. Interestingly, model performance does not always improve when more homework reports are considered. One possible explanation is that the resulting patient representation has less predictive power when more patient's text is considered, given the relatively small size of the training set. However, further research is necessary to confirm this.

Table 3 shows the test F1 score of the best performing model, which is TF-IDF, for each subgroup of the *success* class and all three values of the parameter #Homework reports. As defined in Section 3, the *success* class consists of patients that are *responders* or *remitters* or both *responders* and *remitters* (*responders+remitters*). It is observed that the F1 score calculated for the group of *remitters* is consistently lower than the ones achieved in the other two groups. This suggests that the model might benefit from a different classification setup, where *remitters* are considered as a separate class. Such analysis is left for future work.

As indicated by the classification results, simple lexical representations perform best in our experimental setup. A concrete follow-up question then becomes: Which vocabulary terms have a strong positive or negative correlation with the success class? To answer the question, we perform a linear regression analysis on the complete set of $1,986$ TF-IDF patient embeddings from our prediction data set, using the first three homework sets, and then compute the dot product between each word in the patient vocabulary and the regression coefficient. Vocabulary words with positive value of dot product contribute positively to the prediction of the success class and *vice versa*. Examples of terms with high positive or negative correlation with the success class are presented in Table 4.

There seems to be a vague relation between terminology and treatment outcome, which corroborates previous work by Althoff et al. (2016) that reports positive correlation between positively signed written language and successful counseling conversations. We therefore perform sentiment analysis on the complete list of homework reports written by $2,611$ depression patients, with at least one homework report completed. Since we are interested in the overall progression of sentiment in patients' answers, we are not constrained to include only patients with a specified number of homework reports completed. We train a Multinomial Naïve Bayes classifier for predicting the sentiment class (positive, neutral or negative) on the SenSALDO Swedish sentiment lexicon.[4] All words are represented with TF-IDF embeddings. The sentiment classifier is then used to identify the sentiment class of all words in the vocabulary, excluding stopwords and words with document frequency lower than 0.2. For each patient and concatenated set of homework reports, the number of word occurrences per sentiment class is calculated. We group the patients by the label outcome and compute the average relative frequency of the number of positive words divided by the number of negative words for each homework report and group. The progression of

---

[3] `https://radimrehurek.com/gensim/`

[4] `https://spraakbanken.gu.se/en/resources/sensaldo`

#Homework reports	Baseline	TF-IDF	Word2Vec	FastText	Doc2Vec
1	0.58	**0.69**	0.60	0.56	0.59
3		**0.69**	**0.62**	0.60	0.55
5		0.59	0.59	0.61	0.59

Table 2: Test F1 score for the three proposed classification models considering various sets of weekly homework reports as model input. The performance of the baseline (stratified classifier) is reported for comparison.

	TF-IDF #hw=1	TF-IDF #hw=3	TF-IDF #hw=5
Responders	0.97	0.97	0.72
Remitters	0.89	0.95	0.6
Responders+Remitters	0.99	0.98	0.76

Table 3: Test F1 score calculated for each subgroup of the *success* class when using TF-IDF as the text representation method. #hw denotes the number of homework reports used by the model.

the average relative frequency of words with positive sentiment throughout the homework reports is presented in Figure 1.

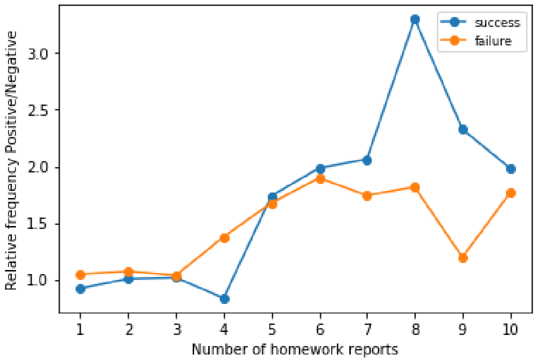


Figure 1: Average relative frequency of words with positive sentiment for all depression patients grouped by the outcome label.

Similar to (Althoff et al., 2016), we observe that more positive sentiment in the homework answers is related to successful treatment outcome. This reflects the treatment progress for patients that in the end are classified as successful. Specifically, homework report 8 had questions concerned with the overall opinion of the treatment, and the usefulness of the tools that the patients had been using. The large differences observed at homework report 8 confirm our intuition that patients showing substantial progress during the treatment, would evaluate its usefulness in more positive terms. Whereas the analysis presented in (Althoff et al., 2016) concerns open-ended counseling conversations, our data set contains patients' answers to targeted questions with positive or negative underlying sentiment. Two indicative questions are *"What's the most important thing you have learned from the past week?"* and *"Tell us about the activities you had planned for this week that you followed through. How was it?"*. The variation in sentiment in homework questions is reflected in the saturation of the relative frequency throughout the progression of homework reports.

6 Discussion

The results demonstrate that the task of predicting treatment outcome based on patient text is very difficult. It is interesting to note that embedding-based representations in some cases fail to perform better than random guessing, which we partly attribute to the comparably small amounts of training data for both domain adaptation and classification, and partly to the fact that such models tend to conflate paradigmatically similar terms, which in this application can signal very different treatment effects. This is demonstrated by terminological and sentiment analysis, which indicate a small but noticeable difference in vocabulary between successful and not successful patients. One hypothesis may be that antonyms are predictive of different outcomes, but have similar representations in the word embeddings, which makes it difficult for a simple linear classifier to separate them. The Bag of Words does not have this problem, which may be the reason it performs better in this specific application.

The selection of patients with a minimum predefined number of homework reports completed for the classification experiments serves the purpose of identifying the best prediction point in the home-

Positive words	forward, away, day, thoughts, initiative, act, wanted, feeling, together, energy, later
Negative words	sleep, rarely, unfortunately, sad, friend, rewarding, despite, life, walk, fill, different

Table 4: List of positively (up) and negatively (down) correlated words with the success class. Words have been translated to English from the original Swedish.

work completion task, but it also poses an important limitation: it reduces the variance in the outcome label. More specifically, success rate is expected to be higher among patients that have completed at least five homework reports. The more careful study of different patient subgroups, based on the number of completed homework reports, is left for future work.

Our current analysis ignores the set of homework assignment questions. Since much of the patient text in ICBT is generated on the prompt of a question, future work could study the correlation of question-answer pairs with the treatment outcome.

7 Conclusions

We have analysed the potential of using patient text from Internet-based Cognitive Behavioural Therapy as the only signal for predicting treatment outcome. When framing the problem as a binary classification task between treatment success and failure, we manage to beat stratified random guessing using a simple Bag of Words classifier. This demonstrates the feasibility of our approach. Interestingly, word embeddings fail to improve on this simple approach, and our best results are achieved using only data from the first couple of weeks of treatment. This has clinical significance because interventions during treatment are still meaningful, and could prove crucial, in the first four weeks of treatment. Additionally, we provide simple terminological and sentiment analysis, which also indicate that there is a signal in the patient text data, albeit a weak one.

The work reported on here should be seen as a feasibility study in that it shows the potential to predict treatment outcome based on patient text, even in a very small subset of the set of patients reporting and reflecting on what they do in treatment, and how treatment progresses. Our next step is to look into larger subsets than the one employed here, but that step will be taken in conjunction with moving from the well-established albeit simple methods used here to contextualised language models.

References

Tim Althoff, Kevin Clark, and Jure Leskovec. 2016. Large-scale analysis of counseling conversations: An application of natural language processing to mental health. *Transactions of the Association for Computational Linguistics*, 4:463–476.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Magnus Boman, Fehmi Ben Abdesslem, Erik Forsell, Daniel Gillblad, Olof Görnerup, Nils Isacsson, Magnus Sahlgren, and Viktor Kaldo. 2019. Learning machines in internet-delivered psychological treatment. *Progress in Artificial Intelligence*, 8(4):475–485.

Magnus Boman and Pedro Sanches. 2015. Sensemaking in intelligent health data analytics. *KI-Künstliche Intelligenz*, 29(2):143–152.

Arman Cohan, Bart Desmet, Andrew Yates, Luca Soldaini, Sean MacAvaney, and Nazli Goharian. 2018. SMHD: a large-scale resource for exploring online language usage for multiple mental health conditions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1485–1497, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bruno Fantino and Nicholas Moore. 2009. The self-reported montgomery-asberg depression rating scale is a useful evaluative tool in major depressive disorder. *BMC psychiatry*, 9:26.

Erik Forsell, Nils Isacsson, Kerstin Blom, Susanna Jernelöv, Fehmi Ben Abdesslem, Nils Lindefors, Magnus Boman, and Viktor Kaldo. 2019. Predicting treatment failure in regular care internet-delivered cognitive behavior therapy for depression and anxiety using only weekly symptom measures. *Journal of Consulting and Clinical Psychology*.

George Gkotsis, Sumithra Velupillai, Anika Oellrich, Harry Dean, Maria Liakata, and Rina Dutta. 2016. Don't let notes be misunderstood: A negation detection method for assessing risk of suicide in mental health records. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 95–105, San Diego, CA, USA. Association for Computational Linguistics.

C Hawley, Tim Gale, and T Sivakumaran. 2002. Defining remission by cut off score on the madrs: Selecting the optimal value. *Journal of affective disorders*, 72:177–84.

Erik Hedman, Brjánn Ljótsson, and Nils Lindefors. 2012. Cognitive behavior therapy via the internet: a systematic review of applications, clinical efficacy and cost–effectiveness. *Expert review of pharmacoeconomics & outcomes research*, 12(6):745–764.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*.

Eyal Karin, Blake F Dear, Gillian Z Heller, Milena Gandy, and Nickolai Titov. 2018. Measurement of symptom change following web-based psychotherapy: Statistical characteristics and analytical methods for measuring and interpreting change. *JMIR mental health*, 5(3):e10200.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119. Curran Associates, Inc.

John Pestian, Henry Nasrallah, Pawel Matykiewicz, Aurora Bennett, and Antoon Leenaars. 2010. Suicide note classification using natural language processing: A content analysis. *Biomedical informatics insights*, 3:BII–S4706.

Xingyi Song, Johnny Downs, Sumithra Velupillai, Rachel Holden, Maxim Kikoler, Kalina Bontcheva, Rina Dutta, and Angus Roberts. 2020. Using deep neural networks with intra- and inter-sentence context to classify suicidal behaviour. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1303–1310, Marseille, France. European Language Resources Association.

Nickolai et al. Titov. 2018. ICBT in routine care: A descriptive analysis of successful clinics in five countries. *Internet interventions*, 13:108 – 115.

Marcel Trotzek, Sven Koitka, and Christoph M Friedrich. 2018. Utilizing neural networks and linguistic metadata for early detection of depression indications in text sequences. *IEEE Transactions on Knowledge and Data Engineering*.

Scalable Evaluation and Improvement of Document Set Expansion via Neural Positive-Unlabeled Learning

Alon Jacovi[1*] **Gang Niu**[2] **Yoav Goldberg**[1,3] **Masashi Sugiyama**[2,4]
[1]Bar Ilan University, Israel
[2]RIKEN, Japan
[3]Allen Institute for Artificial Intelligence, Israel
[4]University of Tokyo, Japan
{alonjacovi,yoav.goldberg}@gmail.com,
gang.niu@riken.jp, sugi@k.u-tokyo.ac.jp

Abstract

We consider the situation in which a user has collected a small set of documents on a cohesive topic, and they want to retrieve additional documents on this topic from a large collection. Information Retrieval (IR) solutions treat the document set as a query, and look for similar documents in the collection. We propose to extend the IR approach by treating the problem as an instance of positive-unlabeled (PU) learning—i.e., learning binary classifiers from only positive (the query documents) and unlabeled (the results of the IR engine) data. Utilizing PU learning for text with big neural networks is a largely unexplored field. We discuss various challenges in applying PU learning to the setting, showing that the standard implementations of state-of-the-art PU solutions fail. We propose solutions for each of the challenges and empirically validate them with ablation tests. We demonstrate the effectiveness of the new method using a series of experiments of retrieving PubMed abstracts adhering to fine-grained topics, showing improvements over the common IR solution and other baselines.

1 Introduction

We are interested in the task of *focused document set expansion*, in which a user has identified a set of documents on a focused and cohesive topic, and they wish to find more documents about the same topic in a large collection. This problem is also known as a “More Like This” (MLT) query in web retrieval. A common way of modeling this problem is to consider the set of documents as a long *query*, with which Information Retrieval (IR) techniques can rank documents. IR literature on document similarity and ranking is vast (Faloutsos and Oard, 1995; Mitra and Chaudhuri, 2000, inter alia)—beyond the scope of this work, and largely orthogonal to it, as will be explained later.

Current methods in document set expansion for very large collections are based on word-frequency or bag-of-words document similarity metrics such as Term Frequency-Inverse Document Frequency (TF-IDF) and Okapi BM25 and its variants (Robertson and Zaragoza, 2009; Zaragoza et al., 2004), considered strong due to their robustness to extreme class imbalance, corpus variance and variable length inputs, as well as their scalability and efficiency (Mitra and Craswell, 2017). However, the performance of such solutions is limited, as the models cannot capture local or global relationships between words.

We examine methods to improve document set expansion by leveraging non-linear (neural) models under the setting of imbalanced binary text classification. To this end, we look to positive-unlabeled (PU) learning (Du Plessis et al., 2015): a binary classification setting where a classifier is trained based on only positive and unlabeled data. In the standard document expansion setting, we indeed only possess positive (the document set) and unlabeled (the very large collection) data.

PU learning has originally been employed for text classification by Liu et al. (2002); Li and Liu (2005); Li and Liu (2003) by using techniques such as EM and SVM. Since then, the setting has been well studied theoretically (Elkan and Noto, 2008; Du Plessis et al., 2015; Niu et al., 2016), and recently objective functions have been developed to facilitate training of flexible neural networks from PU data (Kiryo et al., 2017). We discuss the PU learning setting in more detail in Section 2, and relevant work on PU learning for text in Section 8.

We are, however, not interested in replacing traditional (term-frequency-based) IR solutions, but rather improve upon their results by further classifying the outputs of those models. There are

---

[*] Most of the work done during internship at RIKEN.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 581–592
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

two reasons for this approach: (1) Traditional IR engines are based on word frequencies, and as a result, cannot capture features based on word order; (2) Classification by the use of neural networks does not scale well to "extreme" imbalance[1].

Following these observations, we see traditional IR engines and neural models as complementary to each other. Our proposed solution is a two-step process: First, a BM25-based, MLT IR engine retrieves relevant candidates; Then, a non-linear PU learning model is trained based on the subset of candidates. In this way, each method relieves the weakness of the other.

As already discussed above, PU learning has recently become viable for deep neural network models. As a result, we are able to leverage it to train models that are able to capture higher order features between words. However, PU learning literature focused on theoretical analysis and experiments on small models and simple—**notably, class-balanced**—benchmarks such as MNIST, CIFAR10 and 20News (Kato et al., 2019; Hsieh et al., 2018; Xu et al., 2019). PU learning has not been extensively tested for imbalanced datasets. Scaling PU solutions to high-dimensional, ambiguous and complex data is a significant challenge. One reason for this is that PU data is, by definition, difficult or sometimes impossible to be fully labeled for exhaustive, large-scale evaluation.

For the purpose of document set expansion, and in particular for fine-grained topics, gathering fully-labeled data for an accurate benchmark is also a challenge. For this reason, we propose to simulate the scenario synthetically but realistically by using the PubMed collection of bio-medical academic papers. PubMed entries are manually assigned multiple terms from Medical Subject Headings (MeSH), a large ontology of medical terms and topics. We can treat a set of MeSH terms as defining a fine-grained topic, and use the MeSH labels for deriving fully-labeled tasks (see examples of MeSH topic conjunctions in Table 1). This results in an evaluation setup which is **extensive**, allowing for a large variety of different datasets based on different bio-medical topics; **flexible**, with the ability to simulate different biases in the data gathering to account for many possible practical settings; and **accurate**, with a fully labeled test set.

The contributions of this work are thus:[2]

1. We propose a procedure for generating DSE tasks based on PubMed by using conjunctions of MeSH terms for labels. This serves as a new large-scale benchmark for evaluating (1) PU learning solutions, and (2) DSE solutions, both of which currently suffer from lack of difficult and large-scale evaluation.
2. We expand the PubMed DSE task formulation with a variant that includes biased or unbiased negative data.
3. We apply state-of-the-art PU solutions, previously only evaluated on simple benchmarks for small neural networks, to the PubMed DSE tasks, and report that they perform poorly due to various challenges: no knowledge of class prior, batch size restrictions, extremely imbalanced data (small class prior), and very limited labeled data.
4. We propose methods to deal with the above challenges, and empirically evaluate the new PU solution (which incorporates our solution) on the PubMed DSE tasks, noting a significant improvement over currently available methods.

2 Background: Positive-Unlabeled Learning

PU learning refers to learning a binary classifier from positive and unlabeled data. In this section we briefly describe notation and relevant literature.

Notation. We refer to the *positive set* as *P*, the *labeled positive set* as *LP*, the *unlabeled set* as *U*, and the *negative set* as *N*. Empirical approximations of expectations and priors are denoted $\widehat{\cdot}$.

2.1 Setting

Let $x \in \mathbb{R}^d$ and $y \in \{+1, -1\}$ be random variables jointly distributed by $p(x, y)$ where $p^+(x) := p(x \mid y = +1)$ and $p^-(x) := p(x \mid y = -1)$ are the class marginals (i.e., the positive and negative class-conditional densities), and let $g : \mathbb{R}^d \to \mathbb{R}$ and $\ell : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}_+$ be an arbitrary binary

---

[1] In practice, an IR task may involve positive documents in the order of hundreds or thousands, and negative documents in the order of dozens of millions. Literature dealing with imbalanced classification traditionally discuss typical ratios of 1:50 and 1:100 (Huang et al., 2018; Dong et al., 2018). To our knowledge, the setting of extreme imbalance has not been discussed in literature.

[2] Our code and data are available online at `https://github.com/alonjacovi/document-set-expansion-pu`.

$\lvert LP\rvert$	Topic	BM25+nnPU	BM25	Rand+nnPU	BM25+COPK	Naive	All +	Upperbound
20	Animals + Brain + Rats	48.97	32.25 ± 11.6	40.21	30.47	1.49	44.6	68.17
	Adult + Middle Aged + HIV Infections	42.38	26.75 ± 7.22	40.22	33.59	6.88	30.98	55.61
	Renal Dialysis + Chronic Kidney Failure + Middle Aged	49.16	41.23 ± 8.95	46.58	25.4	0.00	28.40	58.18
	Average of 10† topics	**33.26**	26.69 ± 7.18	30.9	25.47	2.16	26.46	50.46
50	Animals + Brain + Rats	60.56	32.8 ± 10.9	45.13	30.47	5.41	45.86	70.23
	Adult + Middle Aged + HIV Infections	42.77	31.85 ± 10.7	50.52	33.59	12.28	40.53	58.10
	Renal Dialysis + Chronic Kidney Failure + Middle Aged	50.09	35.78 ± 9.13	45.37	25.43	0.00	31.81	57.58
	Average of 10† topics	**37.36**	29.07 ± 7.75	37.01	26.51	3.01	30.41	51.09
	Average of 15‡ topics	**33.82**	27.55 ± 6.20	31.08	25.93	2.12	29.02	47.41

Table 1: Experiment F1 results against the baselines of average performance across topics, as well as three example topics. See Section 6 for details. † denotes the same collection of topics. The average of 15 topics ‡ includes †. The nnPU experiments include BER optimization and proportional batching, but *without* pre-trained embeddings. All experiments use a $|U|$ size of 20,000.

decision function and a loss function of $(g(x), y)$ respectively. For the purpose of this work, we will use the common sigmoid loss, $\ell_{\text{sig}}(t, y) = \frac{1}{1+\exp(ty)}$, as we have observed the best empirical performance with this loss. We denote $\pi^+ := p(y = +1)$ and $\pi^- := p(y = -1)$ as the class prior probabilities, such that $\pi^+ + \pi^- = 1$. The methods described in this section all assume the proportion π^+ to be known.

Binary classification aims to minimize the risk:

$$R(g) := \mathbb{E}_{(x,y)\sim p(x,y)}[\ell(g(x), y)].$$

In supervised (positive and negative: *PN*) learning, both positive $P := \{x_i^P\}_{n^+} \sim p^+(x)$ and negative $N := \{x_i^N\}_{n^-} \sim p^-(x)$ samples are available. The supervised classification risk can be expressed as the partial class-specific risks:

$$R(g) = \pi^+\mathbb{E}_{x\sim p^+(x)}[\ell(g(x), +1)] + \pi^-\mathbb{E}_{x\sim p^-(x)}[\ell(g(x), -1)]. \quad (1)$$

Notice that under the zero-one loss (ℓ_{01}), the risk $R(g)$ refers to $\pi^+\frac{FN}{FN+TP} + \pi^-\frac{FP}{TN+FP}$. When training, we use ℓ_{sig} which can be regarded as a soft approximation of this formulation for back-propagation. In practice, the expectations are expressed as the average of losses and optimized in batched gradient-descent or similar methods.

2.2 Unbiased PU Learning (uPU)

We utilize the case-control variant of PU learning[3] (Ward et al., 2009). Formally, unlabeled data $U := \{x_i^U\}_{n^u} \sim p(x)$ is available instead of N, in addition to $P = \{x_i^P\}_{n^+} \sim p^+(x)$ as before.

In order to train a binary classifier from PU data, we could naively train a classifier to separate positive from unlabeled samples. This approach will result, of course, in a sub-optimal biased solution since the unlabeled dataset contains both positive and negative data. Du Plessis et al. (2015) proposed the following unbiased risk estimator to train a binary classifier from PU data.

Since

$$\pi^-\mathbb{E}_{x\sim p^-(x)}[f(x)] = \mathbb{E}_{x\sim p(x)}[f(x)] - \pi^+\mathbb{E}_{x\sim p^+(x)}[f(x)],$$

we can substitute the negative-class expectation in Equation (1):

$$\begin{aligned} R_{PU}(g) := \mathbb{E}_{(x,y)\sim p(x,y)}[\ell(g(x), y)] = \\ \pi^+\mathbb{E}_{x\sim p^+(x)}[\ell(g(x), +1)] \\ + \mathbb{E}_{x\sim p(x)}[\ell(g(x), -1)] \\ - \pi^+\mathbb{E}_{x\sim p^+(x)}[\ell(g(x), -1)]. \end{aligned} \quad (2)$$

By empirically approximating this risk as an average of losses over our available dataset, we arrive at an unbiased risk estimator that can be trained on PU data, referred to as the *uPU* empirical risk.

Non-negative PU (nnPU). If the loss ℓ is always positive, so should be the risk. However, Kiryo et al. (2017) noted that by using stochastic batched optimization, and specifically via very flexible models (such as neural networks), the negative portion of the uPU loss can eventually cause the loss to go negative during training. To mitigate this overfitting phenomenon, they proposed to encourage the loss to stay positive by using gradient-ascent on the negative portion (which replaces the negative-class risk of the classification risk) when it becomes negative. This method is referred to as *nnPU*.

---

[3] In case-control PU learning, the positive and unlabeled data are collected separately. There are other variants which assume different distributions on the data.

3 The PubMed Set Expansion Task

In this section we discuss the method of generating an extensive benchmark for evaluating solutions of MLT document set expansion.

We are inspired by the following scenario: A user has a set of documents which all pertain to a latent topic, and is interested in retrieving more documents about that topic from a large collection. While traditional term-frequency-based IR solutions scale well to extremely large collections of documents, they are imprecise, and contain a significant amount of noise. Therefore, an additional step based on PU learning can be utilized to classify the output of the IR model, and improve the results.

We are interested in gathering a task for evaluation of the second step. In other words, given an existing black-box IR solution, we would like to use it to produce a dataset for training and evaluation of models which should improve upon the black-box IR solution's performance.

Due to the varied nature of the setting, it is impractical to acquire full supervision for a large number of topics. Therefore, we propose to generate synthetic tasks inspired by the real use-case application.

3.1 Task Generation Method

We generate the document-set expansion tasks by leveraging the expansive PubMed Database: A collection of 29 million bio-medical academic papers. Each document is labeled with MeSH tags, denoting the subject of the document. A conjunction of MeSH terms defines a fine-grained topic, which we use to simulate a user's information intent (example conjunctions in Table 1).

The method of generating one task is then:

1. *Input*: $T \leftarrow$ set of MeSH terms (the retrieval topic); $n^+ \leftarrow$ number of labeled positive data; IR, $\theta_T \leftarrow$ a black-box MLT IR engine, along with query parameters.

2. LP $\leftarrow n^+$ randomly selected papers that are labeled with T.

3. $U \leftarrow \text{IR}(\text{LP}; \theta_T)$.

For the tasks generated and utilized in this paper, we have chosen MeSH sets manually, and $n^+ \in \{20, 50\}$ (for the training set). For the MLT IR engine we have used the Elasticsearch[4] implementation of BM25. The top-$\{10000, 20000\}$ scoring documents are retrieved. We make use of the abstracts of the PubMed papers only. See Appendix A for exact details of our method, as well as a comparison to an alternative method for generating *censoring PU* (explained in the appendix) tasks.[5]

We note that although in essence document set expansion involves using U for both training and evaluation (*transductive case*), we are interested in the case where the PU model is able to generalize to unseen data (*inductive case*). As a result, we split the dataset $[LP; U]$ into training, validation, and test sets, where we use the validation set for hyperparameter tuning and early-stopping, and evaluate on the test set using the true labels. In other words, we assume a separate (from training) small PU set is available for validation. In our experiments, the size of the validation set is half of the size of the training set. In a deployment setting, the PU model can be used to label the training U data.

4 Experiment Details

The rest of this work will reference experiment results. Unless otherwise noted, our base architecture is a single-layer CNN (Kim, 2014). The choice of CNN, over other recurrent-based or attention-based models, is due to this architecture achieving the best performance in our experiments. Test-set performance is reported as an average over multiple MeSH topics (as many as our resources allowed). Except for the experiments that use pretrained models, the inputs are tokenized by words, and word embeddings are randomly initialized and trained with the model. More details are available in Appendix B. We stress that our intent in this work is not to report the very best scores possible, but rather to perform controlled experiments to test hypotheses. To this end, many orthogonally beneficial "tricks" in NLP literature were not utilized. Additionally, nnPU-trained models generally required more diligent hyperparameter tuning due to an additional two hyperparameters.

---

[4] https://www.elastic.co/

[5] The code for generating the tasks, and the data of our generated tasks are available online at the aforementioned repository. The uploaded dataset contains the paper abstracts. The PubMed identifiers are also available in cases where additional information about each paper, such as the full text, can be retrieved from PubMed if desired.

$\lvert LP \rvert$	Prior	Accuracy	F1
20	π^+	84.27	0.0
20	0.5	62.09	33.26
50	π^+	81.71	0.0
50	0.5	59.92	37.36

Table 2: Experiments for the PU model, trained with the nnPU loss with either the true class prior (optimizing for accuracy surrogate) or a prior of 0.5 (optimizing for BER surrogate). Reported average across 10† topics.

5 PU Learning for Document Set Expansion

In PU classification literature, traditionally small (and in many cases, linear) models have been used on relatively simple tasks, such as CIFAR-10 and 20News. However, performance of existing methods does not scale well to very high-dimensional inputs and state-of-the-art neural models for text classification; applying the PU learning methods described in Section 2 to a more practical setting results in several critical challenges that must be overcome—for example, PU learning methods often assume a known class prior, yet estimation of the class prior, particularly for text, is hard and inaccurate. In this section we discuss various challenges we have encountered in applying PU learning to the PubMed Set Expansion task, along with proposed, empirically validated solutions.

5.1 Class Imbalance and Unknown Prior (BER Optimization)

Due to the class imbalance (very small class prior), the classification risk encourages the model to be biased towards negative-class prediction (by prioritizing accuracy) in lieu of a model that achieves worse accuracy but better F1. Thus, optimizing for a metric that is similar to F1 or AUC is preferable.

Under a known class prior π^+ assumption, Sakai et al. (2018) derived a PU risk estimator for optimizing AUC directly. However, π^+ cannot be assumed to be known in practice. Furthermore, the high dimensionality and lack of cluster assumption in the input of our task makes estimation difficult and noisy (Menon et al., 2015; Ramaswamy et al., 2016; Jain et al., 2016; du Plessis et al., 2017).

Following this line of thought, we propose a simple solution to both problems: by assuming a prior of $\widehat{\pi}^+ = 0.5$ in the uPU loss regardless of the value of the true prior, we are able to optimize a surrogate loss for the Balanced Error (BER) metric[6] (Brodersen et al., 2010). Effectively, the uPU loss we are optimizing is:

$$\begin{aligned} R_{PU}(g) = \\ \frac{1}{2}\mathbb{E}_{x\sim p^+(x)}[\ell(g(x),+1) - \ell(g(x),-1)] \\ + \mathbb{E}_{x\sim p(x)}[\ell(g(x),-1)]. \quad (3) \end{aligned}$$

When using the zero-one loss (ℓ_{01}), the binary classification risk is equivalent to BER, while BER minimization is equivalent to AUC maximization: $AUC = \frac{3}{2} - 2BER$ (Menon et al., 2015). Since back-propagation requires a surrogate loss in place of ℓ_{01}, such as ℓ_{sig}, the BER and AUC metrics are not inversely equivalent; However, we've found BER optimization to perform well in practice.

Results. Table 2 shows a performance comparison in which the models trained using a prior of 0.5 achieved stronger F1 performance despite weaker accuracy.

5.2 Small Batch Size (Proportional Batching)

The large memory requirements of state of the art neural models such as Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2018), as discussed in the next subsection, coupled with the need to run on GPU, restrict the batch sizes that can be used.

This presents a challenge: When the loss function is composed of losses for multiple classes, when using stochastic batched optimization, each batch should contain a proportionate amount of data of each class relative to the entire dataset. When the classes are greatly imbalanced, this imposes a lower-bound on the batch size when the batch contains one positive example or more. For example, for a dataset which contains 50 positive and 10,000 unlabeled samples, each batch which contains a positive sample must have 200 unlabeled samples. In practice, we were limited to the vicinity of 20 samples per batch when training large Transformer models.

Using a smaller batch-size than the lower-bound (in the case of the example, 20 as opposed to 201)

---

[6] Given a decision function g:

$$BER(g) = \frac{1}{2}(\frac{FP}{TN+FP} + \frac{FN}{FN+TP})$$

$$R(g;\ell_{01}) = \pi^- \frac{FP}{TN+FP} + \pi^+ \frac{FN}{FN+TP}$$

implies that the vast majority of batches will not have labeled positive samples. This result damages performance in multiple ways. First, the model may overfit to the unlabeled data: Since unlabeled examples are treated as discounted negative examples by the uPU loss, the model will be encouraged to predict the negative class due to an abundance of batches that contain only the "negative" (in truth unlabeled) class. Additionally, early-stopping may be compromised due to the significantly smaller loss in batches that contain only unlabeled data.

To solve these problems, we propose to increase the sampling frequency of the positive class *inversely* to its frequency in the dataset. In practice, this solution simply enforces each batch to have a *rounded-up* proportion of its samples for each class. In the example above, every batch with 20 samples will have 1 positive and 19 unlabeled samples. As we "run out" of positive samples before unlabeled samples, we define an epoch as the a single loop through the positive set.

The implication of increasing the sampling frequency is essentially that the positive component of the uPU loss receives a stronger weight. In our running example, the sampling frequency was increased $\times 10$. For a sampling frequency increase by an order of α, the uPU loss becomes:

$$\grave{R}_{PU}(g) = \alpha\pi^{+}\mathbb{E}_{x\sim p^{+}(x)}[\ell(g(x),+1) - \ell(g(x),-1)] + \mathbb{E}_{x\sim p(x)}[\ell(g(x),-1)]. \quad (4)$$

This, intuitively, counter-acts the overfitting problem caused the abundance of stochastic update steps of entirely unlabeled-class batches. The issue of unstable validation uPU loss is solved as well, since every batch must contain both positive and unlabeled samples, by a ratio that is consistent between the training and validation sets (and thus the validation uPU loss remains a reliable validation metric).

The issue of overfitting in this case is derived from a more general problem: Overfitting to the "bigger" class in stochastic optimization of extremely imbalanced data, when the loss can be decomposed into multiple components for each of the classes (as is the case for cross-entropy loss, as well). For this reason, our solution improves ordinary imbalanced classification under batch size restrictions, as well.

Results. Table 3 shows the effect of the increased sampling frequency method in ordinary imbalanced binary classification, as well as in nnPU training. In the small batch size experiments, the method causes an increase in recall, showing that the model is less inclined towards the "bigger" (in our case, the negative) class. *The results apply in both the PN and PU settings, showing that proportional batching can be beneficial to any imbalanced classification task under batch size restrictions.*

Setting	Class Ratio	Batch Size	Proportional Batching	F1
PN	(P:N) 15:85	512		32.55
		16		5.55
		16	✓	41.61
PU	(LP:U) 2:100	512		22.77
		16		0.0
		16	✓	22.35

Table 3: Evaluation for the sampling frequency increase method for mitigating overfitting to the bigger class in imbalanced classification with small batch size. Results show that proportional batching dramatically improves results under batch size constraints for both ordinary supervision (PN) and PU settings.

5.3 Limited Data

A defining challenge of document set expansion tasks, when observed through the lens of imbalanced classification, is the very small class prior and small amount of labeled positive data. Although BER optimization mitigates the issue of the class imbalance, the issue of very little labeled data remains. To this end, we investigate pretraining as a solution.

We utilize SciBERT (Beltagy et al., 2019) for pretrained contextual embeddings in the PubMed domain. For PubMed abstracts that go above the 512 word-piece limit of SciBERT, we utilize a sliding-window approach that averages all embeddings for a word-piece that appeared in multiple windows.

Results. Utilizing SciBERT embeddings has yielded an increase of F1 performance from 25.75 to 29.96 as an average of five topics.

6 Effectiveness of PU Learning

In this section we evaluate the viability of our proposed solution. All experiments in this section use *BER optimization* and *proportional batching* (Section 5), but no pre-trained embeddings. We refer to our proposed method as **BM25+nnPU** where the IR solution BM25 selects the unlabeled dataset for the PU solution, a CNN model with the nnPU loss.

As an anchor for comparison, we use the following *reference*: **Upper-bound**: An identical model, trained on the same training data with full supervision using the true labels. This reference can be regarded as the upper-bound performance in the ideal case.

We directly compare against the current commonly-used and best-performing solution as **IR (BM25)**[7]: The top-k documents of the IR engine's output, for $k \in \{i\}_{i=|LP|}^{5000}$, are selected as positive documents, while the rest are treated as negative. F1 mean and standard deviation are reported across k. **This strong baseline serves as a reference to the state-of-the-art.**

We additionally compare against standard DSE baselines **All + (all positive)**: Classifying all samples as the positive class; and **Naive**: Supervised learning between the labeled positive set (as P) and the unlabeled set (as N).

Finally, we compare against two additional baselines with the aim of validating the beneficial synergy between the IR step and the PU step. In the **Rand+PU** baseline, we replace the IR step with a random selection of U data. In the **BM25+COPK** baseline, we replace the PU step with a *Constrained K-means Clustering* (Wagstaff et al., 2001) solution, where we perform k-means clustering, $k = 2$, under the constraint that all LP examples must be in the same cluster. To represent examples in embedding space for k-means, we encode the text with SciBERT. Additional details of constrained clustering as a replacement to PU learning are discussed in Appendix C.

The IR baseline is the main alternative to our approach. The all-positive and naive baselines are very simplistic "lower-bound" models to be compared against, while the other two-step baselines evaluate the IR or PU steps separately, providing further justification to using the IR and PU solutions together.

Experiments in Table 1 show a significant increase in F1 performance as an average across many topics, against all baselines.

An interesting experiment in Figure 1 shows the performance of the IR and PU models normalized by the performance of the upper-bound, as a function of the amount of labeled data. The reported values are the distance of F1 scores between each respective model with the upper-bound, normalized by the sum of scores. The figure shows that as more labeled data is added, the PU model (in truth IR+PU) increases in performance at a rate that is higher than the performance increase of the upper-bound. In comparison, the IR model improvement stays relatively constant beyond 300 labeled samples, while the upper-bound continues to increase, causing the disparity between them to grow. This experiment shows that the **IR+PU system scales well with increase in *LP* data, increasing performance at a stronger pace than the fully-supervised reference**, while the IR solution scales poorly.

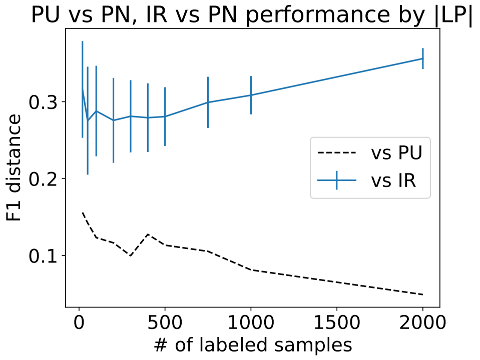


Figure 1: The F1 *absolute difference*, normalized by the sum of the two F1 scores, between the upper-bound and nnPU as a function of the amount of labeled positive samples, as well as between the IR top-k baseline (mean and standard deviation) and the upper-bound. Numbers are the average of five topics.

7 Using Negative Data

The document set expansion scenario may allow for cases where a limited amount of negative data can be collected. For example, the user may possess some number of relevant negative documents which were acquired alongside the positive documents, prior to training; alternatively, the user may label some documents from the model's output as they appear. Therefore, it is of interest to augment the task with biased labeled negative data—i.e., negative documents which were not sampled from the true negative distribution, but were selected with some bias, such as their length, popularity (for example, the number of citations), or their placement within the IR engine's rankings. We consider a bias from document character length, randomly sampling abstracts that are below a certain amount of characters. Alternative bias methods are dis-

---

[7] We note that the comparison here should be made to the *specific IR engine* which resulted in the dataset of the PU model, as the PU model benefits greatly from better performance in the IR engine.

Setting	Precision	Recall	F1
PU	29.35	71.83	40.78
PN (unbiased N)	33.83	70.40	42.14
PN (biased N)	19.34	90.62	31.29

Table 4: Experiments for five topics. All experiments used $|LP| = 50$, $|N| = 50$ for training and $|LP| = 25$, $|N| = 25$ for validation (as well as U in the PU setting).

Setting	Model	(a) Unbiased N F1	(b) Biased N F1
PNU	Ensemble (PN+PU 1+1)	42.31	37.63
PNU	Multi-task	41.50	41.48
PU	Ensemble (3)	41.25	

Table 5: Average performance of the same five topics as in Table 4. All experiments used $|LP| = 50$, $|N| = 50$ for training and $|LP| = 25$, $|N| = 25$ for validation (as well as U). Bias selection for N was performed by character length. "Multi-task" refers to Equation (5).

cussed in Appendix A.

PNU Learning. When it is possible to obtain negative data in limited capacity, it can be incorporated in training. When the negative data is sampled simply from $p^-(x)$, i.e., it is *unbiased* negative data, it is possible to use *PNU classification* (Sakai et al., 2017), which is a linear combination of $R(g)$ and $R_{PU}(g)$:

$$R_{PNU}(g) := \gamma R(g) + (1-\gamma) R_{PU}(g). \quad (5)$$

We note that to our knowledge, PNU learning has not yet been successfully applied to deep models prior to this work. We apply the same solution to the case of biased negative samples. Our PNU experiments include Proportional Batching to overcome the extreme class imbalance.

Results. Tables 4 and 5 summarize the results of PNU learning for the biased and unbiased settings. We observe that performance improves with unbiased negative samples, but does not improve with negative documents selected with bias to shorter documents. In the unbiased case, a simple ensemble of PN and PU models out-performs PNU learning, and we verify that the ensembling is not the sole cause of the performance increase by noting that the PN and PU ensemble out-performs a 3-model PU ensemble. In the biased case, the performance of the PN model is severely lower than the PU model, and in this case indeed the PNU model slightly out-performs the PN and PU ensemble.

8 Related Work

Linear PU models have been extensively used for text classification (Liu et al., 2004; Yu et al., 2005; Cong et al., 2004; Li and Liu, 2005) by using EM and SVM algorithms. Particularly, the 20News corpus has been often leveraged to build PU tasks for evaluation of those models (Lee and Liu, 2003; Li et al., 2007). Li et al. (2010b) have evaluated EM-based PU models against distributional similarity for entity set expansion. Li et al. (2010a) proposed that PU learning may out-perform PN when only the negative data's distribution significantly differs between training and deployment.

du Plessis et al. (2017); Kato et al. (2018) describe methods of estimating the class prior from PU data under some distributional assumptions. Hsieh et al. (2018) introduced *PUbN* as another PU-based loss for learning with biased negatives. PUbN involves two steps, where the marginal probability of a sample to be labeled (positive/negative) is estimated using a neural model, and then used. In our experiments, PUbN has consistently overfit to the majority baseline. We suspect that this is a result from noisy estimation of the labeling probability due to the difficulty of the task.

9 Conclusion

We propose a two-stage solution to document set expansion—the task of retrieving documents from a large collection based on a small set of documents pertaining to a latent fine-grained topic—as a method of improving and expanding upon current IR solutions, by training a PU model on the output of a black-box IR engine. In order to accurately evaluate this method, we synthetically generated tasks by leveraging PubMed MeSH term conjunctions to denote latent topics. Finally, we discuss challenges in applying PU learning to this task, namely an unknown class prior, extremely imbalanced data and batch size restrictions, propose solutions (one of which—"Proportional Batching"—applies in the general scope of PN imbalanced classification, as we empirically validate), and provide empirical evaluation against multiple baselines which showcase the effectiveness of the approach.

Future Work. Stronger class prior estimation, through additional task assumptions, may facilitate direct AUC optimization. Additionally, methods of increasing precision may be considered (such as data augmentation or adversarial training).

Acknowledgments

AJ and YG have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT). GN and MS were supported by JST AIP Acceleration Research Grant Number JPMJCR20U3, Japan.

References

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. *CoRR*, abs/1903.10676.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. 2010. The balanced accuracy and its posterior distribution. In *20th International Conference on Pattern Recognition, ICPR 2010, Istanbul, Turkey, 23-26 August 2010*, pages 3121–3124. IEEE Computer Society.

Gao Cong, Wee Sun Lee, Haoran Wu, and Bing Liu. 2004. Semi-supervised text classification using partitioned EM. In *International Conference on Database Systems for Advanced Applications*, pages 482–493. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Qi Dong, Shaogang Gong, and Xiatian Zhu. 2018. Imbalanced deep learning by minority class incremental rectification. *CoRR*, abs/1804.10851.

M.C. Du Plessis, G Niu, and M Sugiyama. 2015. Analysis of learning from positive and unlabeled data. *Advances in Neural Information Processing Systems*, 1:703–711.

Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220. ACM.

Christos Faloutsos and Douglas W. Oard. 1995. A survey of information retrieval and filtering methods. Technical report, Univ. of Maryland Institute for Advanced Computer Studies Report, College Park, MD, USA.

Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors. 2017. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*.

Yu-Guan Hsieh, Gang Niu, and Masashi Sugiyama. 2018. Classification from positive, unlabeled and biased negative data. *CoRR*, abs/1810.00846.

Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. 2018. Deep imbalanced learning for face recognition and attribute prediction. *CoRR*, abs/1806.00194.

Shantanu Jain, Martha White, and Predrag Radivojac. 2016. Estimating the class prior and posterior from noisy positives and unlabeled data. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2685–2693.

Masahiro Kato, Takeshi Teshima, and Junya Honda. 2019. Learning from positive and unlabeled data with a selection bias. In *International Conference on Learning Representations*.

Masahiro Kato, Liyuan Xu, Gang Niu, and Masashi Sugiyama. 2018. Alternate estimation of a classifier and the class-prior from positive and unlabeled data. *CoRR*, abs/1809.05710.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Ryuichi Kiryo, Gang Niu, Marthinus Christoffel du Plessis, and Masashi Sugiyama. 2017. Positive-unlabeled learning with non-negative risk estimator. In (Guyon et al., 2017), pages 1674–1684.

Wee Sun Lee and Bing Liu. 2003. Learning with positive and unlabeled examples using weighted logistic regression. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 448–455. AAAI Press.

Xiao-Li Li and Bing Liu. 2005. Learning from positive and unlabeled examples with different data distributions. In *European Conference on Machine Learning*, pages 218–229. Springer.

Xiao-Li Li, Bing Liu, and See-Kiong Ng. 2010a. Negative training data can be harmful to text classification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 218–228. Association for Computational Linguistics.

Xiaoli Li and Bing Liu. 2003. Learning to classify texts using positive and unlabeled data. In *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 587–594. Morgan Kaufmann.

Xiaoli Li, Bing Liu, and See-Kiong Ng. 2007. Learning to identify unexpected instances in the test set. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence,* *Hyderabad, India, January 6-12, 2007*, pages 2802–2807.

Xiaoli Li, Lei Zhang, Bing Liu, and See-Kiong Ng. 2010b. Distributional similarity vs. PU learning for entity set expansion. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, Short Papers*, pages 359–364. The Association for Computer Linguistics.

Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. 2002. Partially supervised classification of text documents. In *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*, pages 387–394. Morgan Kaufmann.

Bing Liu, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. 2004. Text classification by labeling words. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*, pages 425–430. AAAI Press / The MIT Press.

Aditya Menon, Brendan Van Rooyen, Cheng Soon Ong, and Bob Williamson. 2015. Learning from corrupted binary labels via class-probability estimation. In *International Conference on Machine Learning*, pages 125–134.

Bhaskar Mitra and Nick Craswell. 2017. Neural models for information retrieval. *CoRR*, abs/1705.01509.

M. Mitra and B. B. Chaudhuri. 2000. Information retrieval from documents: A survey. *Inf. Retr.*, 2(2-3):141–163.

Gang Niu, Marthinus Christoffel du Plessis, Tomoya Sakai, Yao Ma, and Masashi Sugiyama. 2016. Theoretical comparisons of positive-unlabeled learning against positive-negative learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1199–1207.

Marthinus Christoffel du Plessis, Gang Niu, and Masashi Sugiyama. 2017. Class-prior estimation for learning from positive and unlabeled data. *Machine Learning*, 106(4):463–492.

Harish Ramaswamy, Clayton Scott, and Ambuj Tewari. 2016. Mixture proportion estimation via kernel embeddings of distributions. In *International Conference on Machine Learning*, pages 2052–2060.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Tomoya Sakai, Gang Niu, and Masashi Sugiyama. 2018. Semi-supervised AUC optimization based on positive-unlabeled learning. *Machine Learning*, 107(4):767–794.

Tomoya Sakai, Marthinus Christoffel du Plessis, Gang Niu, and Masashi Sugiyama. 2017. Semi-supervised classification based on classification from positive and unlabeled data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2998–3006. JMLR. org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In (Guyon et al., 2017), pages 6000–6010.

Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. 2001. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 577–584. Morgan Kaufmann.

Gill A. Ward, Trevor J. Hastie, Simon Thomas Barry, Jane Elith, and John R. Leathwick. 2009. Presence-only data and the em algorithm. *Biometrics*, 65 2:554–63.

Miao Xu, Bingcong Li, Gang Niu, Bo Han, and Masashi Sugiyama. 2019. Revisiting sample selection approach to positive-unlabeled learning: Turning unlabeled data into positive rather than negative. *CoRR*, abs/1901.10155.

Hailong Yu, Wanli Zuo, and Tao Peng. 2005. A new pu learning algorithm for text classification. In *Mexican International Conference on Artificial Intelligence*, pages 824–832. Springer.

Hugo Zaragoza, Nick Craswell, Michael Taylor, Suchi Saria, and Stephen Robertson. 2004. Microsoft cambridge at trec-13: Web and hard tracks. In *IN PROCEEDINGS OF TREC 2004*.

A PubMed Set Expansion Task Generation

In this section we discuss details of the PubMed Set Expansion task generation process.

Parameters. For this work, we have indexed the January 2019 version of PubMed in an Elasticsearch ver-6.5.4 index. We discard all papers in PubMed that do not have MeSH terms or abstracts (of which there are few). The title and abstract of each paper are tokenized using the Elasticsearch English tokenizer, with term vectors. The title receives a 2.0 score boost during retrieval. For retrieval, we use the Elasticsearch "More Like This" query with the default implementation of BM25, and a "minimum should match" parameter of 20%, indicating that papers that do not share a 20% overlap of terms with the query are dropped. This parameter was controlled in the interest of efficiency, as the query is otherwise very slow.

Table 6 contains statistics about sample topics.

Topic	$\lvert LP\rvert$	$\lvert U\rvert$	Precision	Recall
Liver + Rats, Inbred Strains + Rats	20	10,000	17.45	15.59
	50	10,000	16.55	14.82
Adult + Middle-Aged + HIV Infections	20	20,000	18.33	20.06
	50	20,000	25.42	27.85

Table 6: Dataset sizes for two example PubMed Set Expansion tasks based on the given topics, each composed of three MeSH terms. The reported sizes are for the training set. Precision denotes the proportion of P samples in U, and recall denotes the proportion of retrieved P samples from all positive documents in PubMed.

Censoring PU learning. An alternative, easier, scenario for the Document Set Expansion task involves the case where the LP data was sampled and labeled from the U distribution, termed *censoring PU learning*. To model this case, the task can be generated in the following way:

1. *Input*: $T \leftarrow$ set of MeSH terms (the retrieval topic); $n^+ \leftarrow$ number of labeled positive data; $IR, \theta_T \leftarrow$ a black-box MLT IR engine, along with query parameters.
2. $P \leftarrow$ All papers that are labeled with T.
3. $N \leftarrow IR(P; \theta_T)$
4. $LP \leftarrow n^+$ randomly selected papers in P.
5. $U \leftarrow [P - LP; N]$

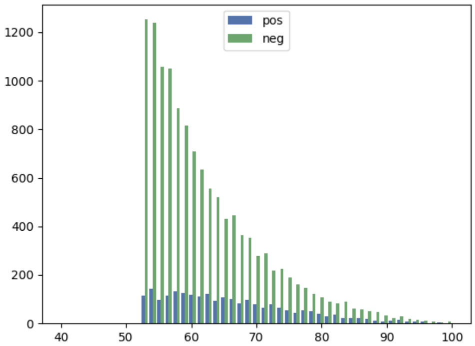


Figure 2: Two histograms of U positive and negative documents respectively by their BM25 score. The horizontal axis denotes buckets of BM25 scores, and the vertical axis is the amount of samples in that bucket.

Experimentally, the F1 performance of all the models (PU and PN) was greatly increased for this setting, in comparison to the case-control tasks described in the main work. All methods discussed in this work apply to the censoring setting, as it is a special case of case-control.

Bias. It is possible to simulate bias in the sampling of documents according to many heuristics and assumptions. For example, it may be assumed that the user is more likely to label documents that are shorter, or documents that are more famous (as indicated by amount of citations in PubMed). Additional possible conditions include the ranking of the IR engine in two possible ways: 1. The user may submit labels after the IR query while viewing the results. In this case, the user is more likely to label documents that are ranked higher; 2. In the case of an IR engine modeled by bag-of-words (such as BM25), documents that rank lower can be assumed to possess less relevant vocabulary overlap with the positive class, such that they may be easier to label at a glance. Figure 2 shows a typical distribution of class according to the rank of BM25 for a sample task of PubMed Set Expansion.

B Experiment Details

The experiments were implemented in PyTorch version 1.0.1.post2, AllenNLP version 0.8.3-unreleased. The neural models used a CNN encoder with max-pooling, with 100 filters for the title and 200 filters for the abstract, split evenly between window sizes of 3 and 5. The choice of CNN (over other recurrent-based or attention-based models) is due to this architecture achieving the best performance in practice. For the SciBERT contextual embeddings, SciBERT-base was used. The learning rate for the model with no pretraining used is 0.001, while the learning rate for the SciBERT model is 0.00005. The nnPU parameters β, γ were set to 0 and tuned over the validation set loss, respectively. In all cases of nnPU training we used the biggest batch-size possible, which was 1000 for the CNN model with no pretraining, and between 16 to 25 for the SciBERT model. In the case of the SciBERT model, we've ignored training and validation samples longer than 600 words, tokenized by the AllenNLP default implementation of `WordTokenizer`, to avoid long outliers which greatly limit the batch size. This was not performed on the test set to maintain an unbiased comparison.

B.1 Experiment Topics

The topics were chosen by a policy of related triplets, such that they could conceivably (though loosely) be relevant searches in practice, by sampling and filtering from MeSH triplets that they occur together in PubMed on an order of hundreds, thousands or tens of thousands. The topics were chosen *without* knowledge of any experiment results related to them, such that they were not picked to achieve a particular outcome.

(†)

1. Animals + Brain + Rats.
2. Adult + Middle Aged + HIV Infections.
3. Lymphatic Metastasis + Middle Aged + Neoplasm Staging.
4. Base Sequence + Molecular Sequence Data + Promoter Regions, Genetic.
5. Renal Dialysis + Kidney Failure, Chronic + Middle Aged.
6. Aged + Middle Aged + Laparoscopy.
7. Apoptosis + Cell Line, Tumor + Cell Proliferation.
8. Disease Models, Animal + Rats, Sprague-Dawley + Rats.
9. Liver + Rats, Inbred Strains + Rats.
10. Dose-Response Relationship, Drug + Rats, Sprague-Dawley + Rats.

(‡)

1. Female + Infant, Newborn + Pregnancy.
2. Molecular Sequence Data + Phylogeny + Sequence Alignment.
3. Cells, Cultured + Mice, Inbred C57BL + Mice.
4. Dose-Response Relationship, Drug + Rats, Sprague-Dawley + Rats.
5. Brain + Magnetic Resonance Imaging + Middle Aged.

C Constrained Clustering for PU Learning

Unfortunately, we are not aware of many competitive alternative solutions to nnPU that interface with only positive and unlabeled data. One such a solution is *constrained clustering*, or clustering under constraints of prior knowledge on which examples should belong in the same cluster, or which examples should not belong in the same cluster.

Constrained clustering can be reduced to a PU problem in the following way: Given LP and U data, we perform clustering under constraints that all of the examples in LP must belong in the same cluster. If N data is available, we may constrain all N data to be in the same cluster, as well, and that LP and N examples may not be in the same cluster. If the algorithm allows a parameterization of the number of clusters, such as COP-Kmeans (Wagstaff et al., 2001), we may specify this number to be 2. Otherwise, all clusters that do not contain the LP examples can be selected as clusters of N, and the cluster that contains the LP examples shall be selected as P.

In this way, we achieve a reduction from the constrained clustering problem to a PU problem, allowing it to serve as a replacement to nnPU. While we are not aware of other work which made this reduction or comparison between constrained clustering and PU learning, in our experiments we note that nnPU has achieved stronger performance and scalability in large data.

The Role of Syntactic Planning in Compositional Image Captioning

Emanuele Bugliarello and **Desmond Elliott**
Department of Computer Science
University of Copenhagen
`{emanuele,de}@di.ku.dk`

Abstract

Image captioning has focused on generalizing to images drawn from the same distribution as the training set, and not to the more challenging problem of generalizing to different distributions of images. Recently, Nikolaus et al. (2019) introduced a dataset to assess compositional generalization in image captioning, where models are evaluated on their ability to describe images with unseen adjective–noun and noun–verb compositions. In this work, we investigate different methods to improve compositional generalization by planning the syntactic structure of a caption. Our experiments show that jointly modeling tokens and syntactic tags enhances generalization in both RNN- and Transformer-based models, while also improving performance on standard metrics.

1 Introduction

Image captioning is a core task in multimodal NLP, where the aim is to automatically describe the content of an image in natural language. To succeed in this task, a model first needs to recognize and understand the properties of the image. Then, it needs to generate well-formed sentences, requiring both a syntactic and a semantic knowledge of the language (Hossain et al., 2019). Deep learning techniques are the standard approach to tackling this problem: images are represented by visual features extracted from Convolutional Neural Networks (*e.g.* He et al. 2016), and sentences are generated by conditioning Recurrent Neural Networks (*e.g.* Hochreiter and Schmidhuber 1997), or Transformers (Vaswani et al., 2017) on the extracted visual features.

While deep neural networks achieve impressive performance in a variety of applications, including image captioning, their ability to demonstrate compositionality, defined as the algebraic potential to understand and produce novel combinations from known components (Loula et al., 2018), has been questioned. Semantic compositionality of language in neural networks has attracted interest in the community (Irsoy and Cardie, 2014; Lake and Baroni, 2018; Baroni, 2019) as compositionality is conjectured to be a core feature not only of language but also of human thought (Fodor and Lepore, 2002).

In image captioning, improving compositional generalization is a fundamental step towards generalizable systems that can be employed in daily life. To this end, Nikolaus et al. (2019) recently introduced a compositional generalization dataset where models need to describe images that depict unseen compositions of primitive concepts. For example, models are trained to describe images with "white" entities and all types of "dog" concepts but never the adjective–noun composition of "white dog." In their dataset, models are evaluated on their ability to caption images depicting the unseen composition of held out concepts. Their study suggests that RNN-based captioning models do not compositionally generalize, and that this is primarily attributable to the language generation component.

In this paper, we study the potential for syntax to improve compositional generalization in image captioning by combining syntactic planning and language generation in a single model. Our study is inspired by the traditional Natural Language Generation (NLG) framework (Reiter and Dale, 1997), where NLG is split into three distinct steps: text planning, sentence planning, and linguistic realization. While state-of-the-art captioning models typically proceed directly from visual features to sentence generation, we hypothesize that a model that plans the structure of a sentence as an intermediate step will improve compositional generalization. A model with a planning step can learn the high-level structure of sentences, making it less prone to overfitting the training data.

Specifically, we explore three methods for integrating syntactic planning into captioning in our

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 593–607
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

experiments: (a) pre-generation of syntactic tags from the image, (b) interleaved generation of syntactic tags and words (Nădejde et al., 2017), and (c) multi-task learning with a shared encoder that predicts syntactic tags or words (Currey and Heafield, 2019). We do so while also empirically investigating four different levels of syntactic granularity.

The main findings of our experiments are that:

- jointly modeling syntactic tags and tokens leads to improvements in Transformer-based (Cornia et al., 2020) and RNN-based (Anderson et al., 2018) image captioning models;
- although the effectiveness of each syntactic tag set varies across our explored approaches, the widely-used *chunking* tag set never outperforms syntactic tags with finer granularity;
- compositional generalization is affected by directly mapping from image representation to tokens because performance can be improved by interleaving a dummy tag with no meaning;
- interleaving syntactic tags with tokens leads to a loss in performance for retrieval systems.

Finally, we also propose an attention-driven image–sentence ranking model, which makes it possible to adaptively combine syntax within the re-scoring approach of Nikolaus et al. (2019) to further improve compositional generalization in image captioning.

2 Planning Image Captions

Natural language generation has traditionally been framed in terms of six basic sub-tasks: content determination, discourse planning, sentence aggregation, lexicalization, referring expression generation and linguistic realization (Reiter and Dale, 1997). Within this framework, a three-stage pipeline has emerged (Reiter, 1994):

- **Text Planning:** combining content determination and discourse planning.
- **Sentence Planning:** combining sentence aggregation, lexicalization and referring expression generation to determine the structure of the selected input to be included in the output.
- **Linguistic Realization:** this stage involves syntactic, morphological and orthographic processing to produce the final sentence.

Early methods for image captioning drew inspiration from this framework; for example, the MIDGE system (Mitchell et al., 2012) features explicit steps for content determination, given detected objects, and sentence aggregation based on local and full phrase-structure tree construction, and TREETALK composes tree fragments using integer linear programming (Kuznetsova et al., 2014). More recently, Wang et al. (2017) propose a two-stage algorithm where the skeleton sentence of the caption (main objects and their relationships) is first generated, and then the attributes for each object are generated if they are worth mentioning. In contrast, the majority of neural network models are based on the encoder-decoder framework (Sutskever et al., 2014) of learning a direct mapping from different granularities of visual representations (Vinyals et al., 2015; Xu et al., 2015; Anderson et al., 2018) to language model decoders based on RNNs (Vinyals et al., 2015) or Transformers (Guo et al., 2020; Cornia et al., 2020).

2.1 Motivation

In this paper, we explore whether image captioning models can be improved by explicitly modeling sentence planning as an intermediate step between content determination and linguistic realization. In particular, we study the use of syntactic tags in enriching the sentence planning step to improve compositional generalization. In the compositional image captioning task, models are tasked with describing images that depict unseen combinations of adjective–noun and noun–verb constructions (see Nikolaus et al. 2019 for a more detailed description of this task). Nikolaus et al. (2019) presented a model that improves generalization with a jointly trained discriminative re-ranker, whereas here, we investigate the role of sentence planning via syntax.

From a psycholinguistic perspective (Griffin and Bock, 2000; Coco and Keller, 2012), there is evidence that humans make plans about how to describe the visual world: they first decide what to talk about (analogous to content determination), then they decide what they will say (a sentence planning phase), and finally, they produce an utterance (linguistic realization). We hypothesize that, analogously to humans, neural network decoders will also find it useful to make such sentence plans.

From a machine learning perspective, the use of syntactic structure can mitigate the bias introduced by the maximum likelihood training of neural network image captioning models. Recall that in the context of image captioning, the optimization objective consists of maximizing the likelihood:

$$\mathcal{L} = \prod_{t=1}^{T} \mathbb{P}(y_t | y_{<t}, v), \qquad (1)$$

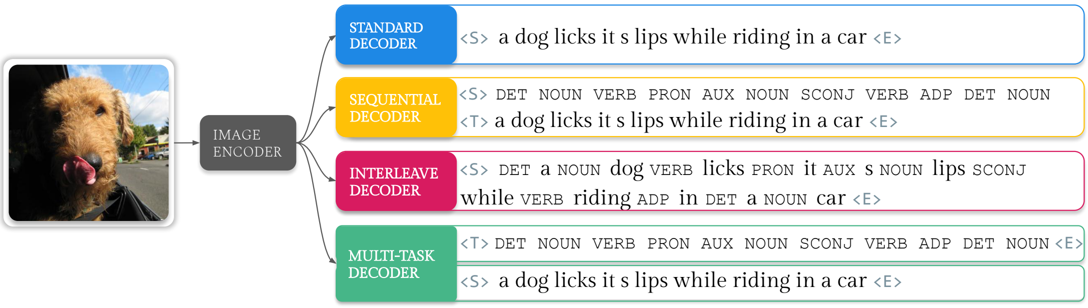


Figure 1: Approaches to syntactically plan image captioning (with POS tags). STANDARD captioning systems directly generate a sequence of surface forms (i.e. words). SEQUENTIAL generates a sequence of syntactic tags, followed by a sequence of surface forms. INTERLEAVE alternates syntactic tags and surface forms. MULTI-TASK generates either a sequence of syntactic tags or a sequence of surface forms from a shared decoder.

where v denotes the visual features (either a single vector or a set of vectors extracted from an image).

In a standard generation task, a model learns to predict the next token based on what it has observed so far. This is especially limiting when it is evaluated on unseen combinations of adjective–noun and noun–verb constructions in the compositional generalization task (i.e. data points that fall outside the training distribution). In fact, models are not explicitly asked to learn *word classes* nor how to connect them to form novel combinations. Whereas, if a system also models syntax, it can assign higher probability to “white dog” if it expects to generate a sequence with an adjective followed by a noun.

2.2 Planning Approaches

We investigate three approaches to jointly modeling tokens and syntactic tags: syntax-driven sequential caption planning (SEQUENTIAL), syntax-interleaved caption generation (INTERLEAVE), and syntax and caption generation via multi-task learning (MULTI-TASK). See Figure 1 for an overview.

SEQUENTIAL: Our first approach closely follows the traditional NLG pipeline and it is related to the *text planning* stage defined above, although limited to sentence-level rather than to a full discourse. Here, a model plans, through syntactic tags, the order of the information to be presented. Specifically, the model is required to generate a sequence whose first T outputs represent the underlying syntactic structure of the sentence before subsequently generating the corresponding T surface forms.

INTERLEAVE: Our second approach consists of interleaving syntactic tags and tokens during generation, which means a syntactic tag and its realization are next to each other, removing the pressure for a model to successfully track long-range dependencies between tags and tokens. Moreover, this allows for a more flexible planning, where the model can adapt the sentence structure based on the previously generated tags and tokens. In particular, a model can break bi-gram dependencies and learn narrower distributions over the next word based on the current syntactic tag. For instance, if we consider part-of-speech tags, the model learns that only a subset of the vocabulary corresponds to nouns, and another subset to adjectives, and so on.

MULTI-TASK: Our last approach is based on multi-task learning, where a model produces either a sequence of tokens (main task) or syntactic tags (secondary task). We draw on the simple and effective approach of Currey and Heafield (2019), proposed for neural machine translation (NMT). In the NMT framework, the source sentence was prepended a task-specific tag, which led the decoder to either predict the translation of the source sentence or the syntactic tags of the source sentence. We adapt this to image captioning by setting the first token to either a start-of-syntax token (`<T>`) or start-of-sentence token (`<S>`) and then generating tags or tokens, respectively. Compared to the other approaches, MULTI-TASK allows the model to learn both types of forms at the same position. While this approach does not double sequence length, it doubles the number of sequences per training epoch.

2.3 Syntactic Granularity

In addition to the three approaches of realizing sentence planning, we investigate the effects of different syntactic tags from a coarse to fine granularity. We experiment with the following tags:

- **CHUNK:** Also known as *shallow parsing*, chunks are syntactic tags that model phrasal structure in a sentence, such as noun phrases (NP) and verb phrases (VP).

- **POS:** Part-of-speech tags are specific lexical categories to which words are assigned, based on their syntactic context and role, such as nouns (N) and adjectives (ADJ).
- **DEP:** Dependency-based grammars model the structure as well as the semantic dependencies and relationships between words in a sentence. In this study, we consider the dependency labels assigned to each word, such as adjectival modifiers (amod), which denote any adjective that modifies the meaning of a noun.
- **CCG:** Combinatory categorial grammar (Steedman and Baldridge, 2006) is based on combinatory logic and provides a transparent interface between surface syntax and the underlying semantic representation. For example, the syntactic category assigned to "sees" is "(S\NP)/NP", denoting it as a transitive verb that will be followed by a noun phrase.

We also study the merit of breaking bi-gram dependencies for the INTERLEAVE approach by tagging each word with a synthetic tag <IDLE>. We hypothesize this approach would not give any benefits in any metric, as attention-based models can simply learn to ignore these pseudo-tags.

3 Experimental Setup

Data We use training and evaluation sets such that paradigmatic gaps exist in the training set. That is, for a concept pair $\{c_i, c_j\}$, the validation $\mathcal{D}_{val}$ and test $\mathcal{D}_{test}$ sets only contain images in which at least one of the captions contains the pair of concepts, while the complementary set – where concepts c_i and c_j can only be observed independently – is used for training $\mathcal{D}_{train}$. Following Nikolaus et al. (2019), we select the same 24 adjective–noun and verb–noun concept pairs, and split the English COCO dataset (Lin et al., 2014) into four sets, each containing six held out concept pairs.

Pre-processing We first lower-case and strip away punctuation from the captions. We then use StanfordNLP (Qi et al., 2018) to tokenize and lemmatize the captions, and to extract universal POS tags and syntactic dependency relations. For IOB-based chunking, we train a classifier-based tagger on *CoNLL2000* data (Tjong Kim Sang and Buchholz, 2000) using NLTK (Bird et al., 2009). Finally, we use the A* CCG parsing model by Yoshikawa et al. (2017) with ELMo embeddings (Peters et al., 2018) to extract CCG tags. Visual features are extracted from 36 regions of interest in each image using Bottom-Up attention (Anderson et al., 2018) trained on Visual Genome (Krishna et al., 2017).

Evaluation Following Nikolaus et al. (2019), we evaluate compositional generalization with Recall@K. Given K generated captions for each of the M images in an evaluation set, $\{\langle s_1^1, \ldots, s_K^1 \rangle, \ldots, \langle s_1^M, \ldots, s_K^M \rangle\}$, the recall of the concept pairs is given by:

$$\text{Recall@K} = \frac{|\{\langle s_k^m \rangle | \exists k : s_k^m \in \mathcal{C}\}|}{M}, \quad (2)$$

where s_k^m denotes the k-th generated caption for image m and $\mathcal{C}$ is the set of captions which contain the expected concept pair and in which the adjective or the verb is a dependent of the noun.

In addition, we use `pycocoeval` to score models on the common image captioning metrics: METEOR (M; Denkowski and Lavie 2014), SPICE (S; Anderson et al. 2016), CIDEr (C; Vedantam et al. 2015), and BLEU (B; Papineni et al. 2002); and the recent multi-reference BERTScore (BS; Yi et al. 2020). In particular, we report the average recall across all concept pairs, the average across the four splits for each score in `pycocoeval`, and the average across all captions for BERTScore.

Models We evaluate three models:

- **BUTD**: Bottom-Up and Top-Down attention (Anderson et al., 2018), a strong and widely-employed RNN-based captioning system.
- **BUTR**: Bottom-Up and Top-Down attention with Ranking (Nikolaus et al., 2019), an RNN-based, multi-task model trained for image captioning and image–sentence ranking that achieves state-of-the-art performance in the compositional generalization task.[1]
- **$\mathcal{M}^2$-TRM**: Meshed-Memory Transformer (Cornia et al., 2020), a recently proposed Transformer-based architecture that achieves state-of-the-art performance in image captioning on the COCO dataset.

Implementation details We follow Nikolaus et al. (2019) and Cornia et al. (2020) to train their systems. Model selection is performed using early stopping, which is determined when the BLEU score of the generated captions in the validation set

---

[1] We denote as BUTR the model that uses the re-ranking module (BUTR+RR in Nikolaus et al. 2019) as it was shown to be essential to improve compositional generalization.

	Model	R@5	M	S	C	B	BS
	BUTD	9.5	25.2	18.6	92.7	32.3	41.7
SEQUENTIAL	+IDLE	8.7	23.7	17.8	87.6	30.0	38.8
	+CHUNK	10.9	24.7	18.2	89.2	31.2	41.2
	+POS	9.5	24.1	17.5	86.1	30.1	40.7
	+DEP	11.1	24.6	17.8	89.7	30.8	41.0
	+CCG	10.6	24.5	18.0	88.4	30.4	41.0
INTERLEAVE	+IDLE	10.5	25.3	18.8	94.3	32.3	41.7
	+CHUNK	9.7	25.2	18.7	93.4	32.5	41.7
	+POS	**11.8**	25.4	18.8	94.4	**32.7**	41.7
	+DEP	10.8	25.2	18.7	93.0	31.9	41.6
	+CCG	10.5	25.4	**19.0**	94.6	**32.7**	41.9
MULTI-TASK	+IDLE	9.8	25.5	18.7	94.5	**32.7**	41.8
	+CHUNK	10.3	25.5	**19.0**	94.5	32.4	41.9
	+POS	10.3	25.4	18.8	93.8	32.6	41.8
	+DEP	11.4	25.5	18.9	93.9	**32.7**	41.9
	+CCG	10.8	**25.7**	**19.0**	**95.6**	**32.7**	**42.0**

Table 1: Average validation results for our approaches to integrating syntactic planning into image captioning evaluated across four types of syntactic forms.

does not increase for five consecutive epochs.[2] We use the default hyperparameters and do not fine-tune them when tasking the models with syntax generation. For full experimental details, refer to App. A. Our code and data are publicly available.[3]

4 Syntax Awareness

In Table 1, we first report the performance of BUTD when jointly modeling different types of syntactic tags and each approach to sentence planning.

Syntax helps compositional image captioning Table 1 clearly shows that, regardless of the level of granularity, syntactic planning enhances compositional generalization in image captioning (R@5). Moreover, CHUNK – one of the most widely-used tag sets for syntax-aware image captioning (*e.g.* Kuznetsova et al. 2012; Yang and Liu 2020) – is outperformed by tag sets with finer granularity (e.g. DEP) in *every* approach, motivating further research into incorporating them in image captioning.

Looking at the results for the SEQUENTIAL approach, we see that, with the exception of POS tags, syntactic planning increases the ability of the model to recall novel concept pairs, with gains of at least +1.1 R@5 points. We then hypothesize that syntax-based sequential planning is effective if the tags convey information about words in relation to each other, e.g. CCG tags as opposed to POS tags.

When the model INTERLEAVEs syntactic tags and words, there is an improvement of at least +1.0 R@5, except for CHUNK. Moreover, POS tags lead to the highest gain of +2.3 R@5.

Finally, the MULTI-TASK approach also leads to significant gains in compositional generalization, with DEP (original setup of Currey and Heafield 2019) giving the highest R@5, corroborating the effectiveness of our porting into image captioning.

Generalization across categories We further investigate the role of syntactic planning for the different unseen composition categories defined by Nikolaus et al. (2019). Figure 2 illustrates how our different combinations of approaches and syntactic tags deal with color and size, type of the objects (animate and inanimate) and type of the verbs (transitive and intransitive). We see that DEP tags consistently improve upon BUTD for color and size concept pairs, regardless of the planning approach, making them a robust tag set for future research. INTERLEAVE+POS also leads to gains for all color and size categories, with up to +10 R@5 for colors of inanimate objects. Conversely, all the variants perform worse than the baseline for the sizes of animate objects. However, this drop is not substantial because BUTD already performs poorly.

Towards neural NLG pipelines While the SEQUENTIAL approach closely follows the traditional NLG pipeline, it consistently degrades performance in standard metrics for image captioning. On the other hand, both INTERLEAVE and MULTI-TASK lead to higher performance in compositional generalization and other metrics. In particular, when BUTD is trained to predict either words or CCG tags in the MULTI-TASK approach, the generated captions achieve the highest average scores, including a substantial gain of +2.9 CIDEr points. These results indicate that neural models require novel ways of sentence planning; and that effectively doing so *consistently* leads to the same or better performance in every considered metric.

Grounding the need for planning Overall, Table 1 provides empirical support that an explicit planning step improves compositional generalization in image captioning. In fact, even breaking bi-grams with the <IDLE> tag in the INTERLEAVE approach improves performance: the standard approach of directly mapping image representations to tokens is sub-optimal because the model learns to generate *n*-grams seen during training.

---

[2] Whenever present, syntactic tags are stripped away when computing evaluation metrics such as BLEU scores.

[3] `https://github.com/e-bug/syncap`.

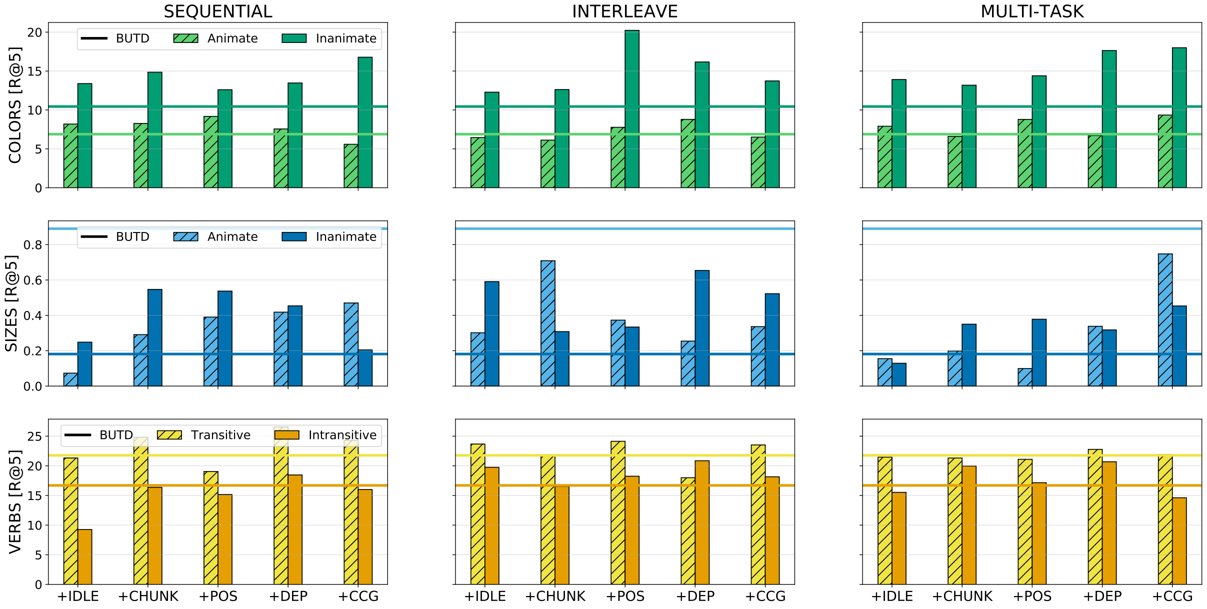


Figure 2: R@5 of unseen compositions by planning approaches (columns) and composition categories (rows).

	Captioning Evaluation						Text Retrieval			Image Retrieval		
Model	**R@5**	**M**	**S**	**C**	**B**	**BS**	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**
BUTR	15.0	26.2	19.9	88.6	28.9	41.8	21.3	45.4	57.8	14.7	35.1	47.0
+POS	12.0	25.7	19.4	85.4	27.4	41.4	17.4	39.3	50.6	12.4	31.0	42.0
BUTR$_{mean}$ +POS	14.2	25.9	19.7	87.4	28.3	42.9	23.3	47.7	60.0	17.1	38.6	50.2
BUTR$_{weight}$	14.9	**26.4**	**20.2**	88.8	28.5	**43.2**	**26.0**	**51.7**	**63.6**	**18.6**	**40.9**	**52.8**
+POS	**16.4**	**26.4**	20.0	**89.8**	**29.1**	43.1	24.5	48.6	60.6	18.0	40.2	52.0

Table 2: Average validation results when interleaving syntactic and lexical forms in BUTR and our variants.

Given its superior performance in the recall of novel compositions of concepts, we adopt INTERLEAVE+POS throughout the remainder of this paper to jointly model syntactic tags and words. For clarity of exposition, we refer to this approach as POS.

4.1 Adaptive Re-Ranking for Syntax

Recall that the best-performing model for compositional image captioning re-ranks its generated captions given the image (BUTR; Nikolaus et al. 2019). Here, we study how to combine the benefits of syntactic planning and their re-ranking approach.

The BUTD model, investigated above, is a two-layer LSTM (Hochreiter and Schmidhuber, 1997) in which the first LSTM encodes the sequence of words, and the second LSTM integrates visual features through an attention mechanism to generate the output sequence (Anderson et al., 2018). The state-of-the-art BUTR model extends this with an image–sentence ranking network that projects images and captions into a joint visual-semantic embedding. The sentence representation used by the ranking network is a learned projection of the final hidden state of the first LSTM: $\mathbf{s} = \mathbf{W}\mathbf{h}_T^l$.

Ranking performance Table 2 shows that the image–sentence retrieval performance of BUTR decreases when interleaving POS tags with words. Given the previous formulation of BUTR and its connection to BUTD, we conclude that jointly modeling syntactic tags and words leads to decreased performance in the generation and ranking tasks.

Adaptively attending to tags We explore two approaches to combining the improvements of interleaved syntactic tagging with ranking:

- *mean*: The model creates a mean representation over the hidden states of the first LSTM.
- *weight*: The model forms a weighted pooling of the hidden states of the first LSTM layer, whose weights are learned through a linear layer. This is a simple form of attention mechanism and it is equivalent to the one used

Model	R@5	M	S	C	B	BS
BUTD	9.5	25.2	18.6	92.7	32.3	41.7
+POS	11.8	25.4	18.8	94.4	32.7	41.7
BUTR$_{weight}$	14.9	26.4	20.2	88.8	28.5	43.2
+POS	**16.4**	26.4	20.0	89.8	29.1	43.1
$\mathcal{M}^2$-TRM	10.6	27.9	21.6	**114.0**	**37.2**	44.4
+POS	13.2	**28.0**	**21.7**	113.8	35.4	**44.9**

Model	R@5	M	S	C	B	BS
BUTD	9.2	25.4	18.6	94.4	32.4	41.8
+POS	11.1	25.4	18.7	96.3	32.9	41.8
BUTR$_{weight}$	13.5	26.4	20.1	91.0	28.6	43.3
+POS	**15.4**	26.3	20.0	91.0	28.7	43.2
$\mathcal{M}^2$-TRM	10.1	27.8	21.5	**115.7**	**36.5**	44.5
+POS	12.1	**28.0**	**21.6**	**115.7**	35.0	**44.9**

Table 3: Average validation (left) and test (right) results when interleaving syntactic (POS) and lexical forms.

by Nikolaus et al. (2019) to represent image features in the shared embedding space:

$$\begin{aligned} \omega_t &= \mathbf{W}_\alpha \mathbf{h}_t^l, \\ \boldsymbol{\alpha} &= \text{softmax}\left(\omega\right), \\ \mathbf{s} &= \mathbf{W} \sum_{t=1}^{T} \alpha_t \mathbf{h}_t^l. \end{aligned} \tag{3}$$

Table 2 shows that the weighting mechanism in the ranking model effectively disentangles syntactic tags and tokens, resulting in $+1.5$ RECALL points over BUTR, with small improvements to the other metrics. Compared to BUTR, BUTR$_{weight}$ also improves the retrieval performance of the ranking module. Adding POS tags still decreases retrieval performance but, compared to BUTR+POS, the difference is now halved for text retrieval and only 0.7 points for image retrieval. Overall, our BUTR$_{weight}$ is a more general and robust approach to jointly training a captioning system and a discriminative image–sentence ranker.

5 Results and Discussion

We now report the final performance of three image captioning models that integrate the syntactic planning (INTERLEAVE+POS) with word generation.

Model-agnostic improvements We start by investigating whether the compositionality given by syntactic planning generalizes across architectures. Table 3 reports average validation and test scores for the BUTD, BUTR$_{weight}$ and $\mathcal{M}^2$-TRM models. We find that interleaving POS tags and tokens consistently leads to $+2$ RECALL points in each model without affecting the performance on other metrics, with the exception of decreased BLEU score of $\mathcal{M}^2$-TRM. In this case, $\mathcal{M}^2$-TRM +POS generates captions that are abnormally truncated, ending with bi-grams such as "of a," "on a" and "to a".[4] Furthermore, we can clearly see that despite $\mathcal{M}^2$-TRM outperforming the RNN-based models in every standard metric, it is only +1 RECALL point better than BUTD at compositional generalization. Hence, syntactic planning is an effective strategy to compositional generalization, *regardless* of the language model used.

Model	Color		Size		Verb	
	A	I	A	I	T	I
BUTD	6.9	10.4	**0.9**	0.2	21.8	16.7
+POS	7.8	20.2	0.4	0.3	24.1	18.2
BUTR$_{weight}$	15.0	24.9	0.8	1.6	26.2	20.8
+POS	**16.2**	**30.4**	**0.9**	**2.5**	26.8	**21.9**
$\mathcal{M}^2$-TRM	5.2	11.9	0.2	0.2	29.1	16.6
+POS	7.9	17.9	0.1	0.2	**32.3**	20.6

Table 4: Average validation R@5 scores for different categories of held out pairs. Color and size adjectives are split into **A**nimate or **I**nanimate objects; Verbs are split into **T**ransitive and **I**ntransitive verbs.

Generalization across categories Table 4 lists the R@5 scores for different categories of held out pairs. Differently from the results reported by Nikolaus et al. (2019), we find the performance of BUTD for noun–verb concept pairs to be much higher thanks to a larger beam size (equal to the one used for BUTR in our experiments). Moreover, the performance from interleaving POS across different categories of held out pairs shows that improvements are consistent across categories and models, with the exception of size modifiers of animate objects, where all models perform poorly. This was also found by Nikolaus et al. (2019) and it is likely due to the need for real-world knowledge (i.e. does this image depict a "big dog" compared to *all other* "dogs"?). For a full breakdown of the R@5 generalization performance for each held out pair by each model, see Table 9 in App. B.

---

[4] This is known as reward hacking (Amodei et al., 2016) which arises in models with a reinforcement learning-based optimization phase. Investigating whether proposed approaches to mitigate this problem (Liu et al., 2017; Li et al., 2019, *inter alia*) are also effective in our setup is left as future work.

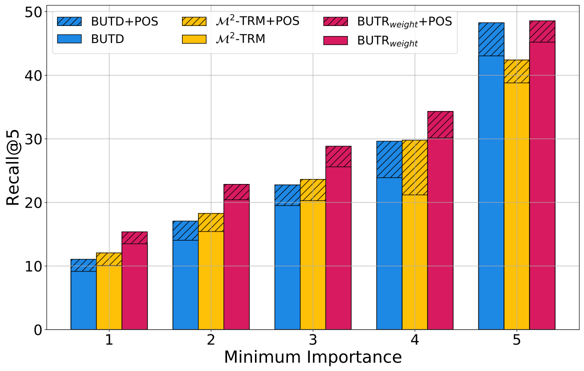


Figure 3: Average test R@5 as a function of the minimum importance of a concept pair for each model.

Performance by minimum importance Given that annotators of the COCO dataset were given a relatively open task to describe images, captioning systems should exhibit higher recall of concept pairs when more annotators use them in the descriptions. As shown in Figure 3, this behavior is seen in each model, with increasing gains given by jointly modeling lexical and syntactic forms. In particular, we observe that the $\mathcal{M}^2$-TRM model recalls fewer pairs than BUTD when they are considered more relevant (more annotators use them in describing an image), and that interleaving POS tags partially solves its limitations. Moreover, as agreement among annotators increases, we also see that BUTD+POS is as effective as BUTR$_{weight}$, corroborating the effectiveness of our model-agnostic approach against a more complex, multi-task model.

Captions diversity Table 5 reports the average scores for caption diversity (van Miltenburg et al., 2018) in the validation data. Comparing BUTD (RNN-based) and $\mathcal{M}^2$-TRM (Transformer-based) models, we see that the output vocabulary of the $\mathcal{M}^2$-TRM-based model spans many more word types, resulting in +11% novel captions. However, $\mathcal{M}^2$-TRM has lower mean segmented type-token ratios (TTRs), contrasting the conclusion of van Miltenburg et al. (2018) that the number of novel descriptions is strongly correlated with the TTR (while this correlation is maintained with the number word types). The models that jointly model syntactic tags and tokens lead to a higher number of types in both models and a substantial +8% in novel captions for $\mathcal{M}^2$-TRM, without affecting other metrics. Clearly, BUTR$_{weight}$ leads to longer sentences, more types, higher TTRs and the highest percentage of novel captions. We can also see that BUTR$_{weight}$ achieves the highest coverage (defined as the percentage of learnable words it can recall), while $\mathcal{M}^2$-TRM has the highest local recall score, being able to better recall the content words that are important to describe a given image.

Model	**ASL**	**Types**	**TTR$_1$**	**TTR$_2$**	**%Novel**	**Cov**	**Loc$_5$**
BUTD	8.6	463	0.16	0.37	69.2	0.12	0.74
+POS	8.6	466	0.16	0.37	70.3	0.12	0.75
BUTR$_{weight}$	**10.3**	**783**	**0.20**	**0.49**	**97.2**	**0.20**	0.78
+POS	**10.3**	778	**0.20**	**0.49**	96.7	**0.20**	0.78
$\mathcal{M}^2$-TRM	9.1	580	0.14	0.33	80.1	0.15	**0.83**
+POS	9.5	601	0.13	0.33	88.4	0.15	**0.83**

Table 5: Average validation set scores for diversity metrics as defined in van Miltenburg et al. (2018).

Accuracy of syntactic forms We verify that the models can correctly predict syntactic tags, regardless of their granularity and the approach used to jointly modeling them with tokens. Indeed, the accuracy of the generated syntactic tags, measured as the ratio of *sequences* matching the annotations from StanfordNLP, by BUTD is high, ranging between 95% and 99%. See App. B for details.

Qualitative examples Figure 4 shows generated captions. Compared to standard BUTD, all syntax-aware approaches allow the model to recall more unseen concept pairs, while also improving the overall quality of the captions. In addition, when looking at the captions generated by all three models, both with and without interleaving POS tags, we find the integration of syntactic tags to clearly **improve the quality** of the generated caption. See Figure 5 in App. B for more examples.

6 Related Work

Compositional image captioning Nikolaus et al. (2019) studies compositional generalization in image captioning with combinations of unseen adjective–noun and verb–noun pairs, whose constituents are observed at training time but not their combination, thus introducing a *paradigmatic gap* in the training data. Nikolaus et al. (2019) showed how to improve compositional generalization by jointly training an image–sentence ranking model with a captioning model. Other work has also investigated generalization to unseen combinations of visual concepts as a classification task (Misra et al., 2017; Kato et al., 2018), triplet prediction (Atzmon et al., 2016), or unseen objects (Lu et al., 2018). Here, we improve generalization by jointly modeling syntactic tags and tokens, and we show

BUTD: there is a woman that is on the floor
BUTD + SEQUENTIAL: a woman doing a trick on a bicycle
BUTD + INTERLEAVE: a woman riding a bike on a wooden floor
BUTD + MULTI-TASK: a woman riding a bike on a wooden surface

BUTD: a woman with a child sitting on a bench
BUTD + POS: **a girl that is standing** on a skateboard

BUTR$_{weight}$: a girl and child playing with a toy in a backyard
BUTR$_{weight}$ + POS: **a girl doing a trick on a skateboard** on a brick walkway

$\mathcal{M}^2$-TRM: a girl doing a trick on a skateboard with
$\mathcal{M}^2$-TRM + POS: a little girl **standing on a skateboard** in the

Figure 4: **Top:** Captions generated by BUTD with different approaches to integrating syntactic tags (based on best R@5). **Bottom:** Captions generated by BUTD, BUTR$_{weight}$, and $\mathcal{M}^2$-TRM, and each model when interleaving POS tags. Syntax-aware approaches generate **higher quality** captions with more unseen concept pairs.

how to combine this with the improvements gained from a jointly-trained ranking model.

Joint syntactic and semantic representations While little work has investigated the interaction of jointly modeling semantics and various syntactic forms in captioning models, a few studies have exploited syntax in image and video captioning. Zhao et al. (2018) propose a multi-task system to jointly train the task of image captioning with two additional tasks: multi-object classification and syntax generation. The same LSTM decoder is used to generate captions *and* CCG tags by mapping the hidden representations to either word or tag vocabularies through two different output layers. Dai et al. (2018) propose a two-stage sequential pipeline where a sequence of noun-phrases is first selected from a fixed pool, which are then patched together via predetermined connecting phrases. This method, however, is unlikely to realize any benefits for compositional generalization because it uses the top-50 noun-phrases and $1,000$ connecting phrases from the training set. Our INTERLEAVE approach can be used to address these limitations in their "phrase pool" and "connecting" modules to produce unseen compositions. Deshpande et al. (2019) rely on sequences of POS tags to produce diverse captions. Similarly to our SEQUENTIAL approach, their model first predicts a sequence of POS tags conditioned on the input image. However, the authors limit the POS sequences to $1,024$ templates obtained through quantization of the training set. During inference, the model samples k POS tag sequences and uses them to condition a greedy decoder for captions generation. Hou et al. (2019) take yet another approach to jointly learn POS tags and surface forms in the framework of video captioning. They introduce a model that resembles our INTERLEAVE approach but with two main differences: (i) the t-th tag is not conditioned on previous tags, and (ii) the t-th word is only conditioned on the t-th tag and the video.

7 Conclusion

We investigated a variety of approaches along with the use of syntactic tag sets to achieve compositional generalization in image captioning via sentence planning. Our results support the claim that combining syntactic planning and language generation *consistently* improves the generalization capability of RNN- and Transformer-based image captioning models, especially for inanimate color–noun combinations. While this approach penalizes image–sentence ranking models, we showed that this can be overcome with an adaptive mechanism, resulting in state-of-the-art performance on the compositional generalization task. We believe our results will lead to further exploration of syntax-aware captioning models given their potential to better generalize, both in terms of under-researched syntactic granularity (e.g. CCG) and more expressive alternatives to modeling syntactic structure. Another direction for future work is to focus on size–noun compositions, which rely on the successful integration of real-world knowledge.

Acknowledgments

We are grateful to the anonymous reviewers, Mitja Nikolaus, Mert Kiliçkaya and members of the CoAStaL NLP group for their useful comments and discussions. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 801199.

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *ArXiv*, abs/1606.06565.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *European Conference on Computer Vision*, pages 382–398. Springer.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086.

Yuval Atzmon, Jonathan Berant, Vahid Kezami, Amir Globerson, and Gal Chechik. 2016. Learning to generalize to new compositions in image understanding. *ArXiv*, abs/1608.07639.

Marco Baroni. 2019. Linguistic generalization and compositionality in modern artificial neural networks. *Philosophical Transactions of the Royal Society B*, 375(1791).

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 794–803, Stockholmsmässan, Stockholm Sweden. PMLR.

Moreno I Coco and Frank Keller. 2012. Scan patterns predict sentence production in the cross-modal processing of visual scenes. *Cognitive science*, 36(7):1204–1223.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10575–10584.

Anna Currey and Kenneth Heafield. 2019. Incorporating source syntax into transformer-based neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 24–33, Florence, Italy. Association for Computational Linguistics.

Bo Dai, Sanja Fidler, and Dahua Lin. 2018. A neural compositional paradigm for image captioning. In *Advances in Neural Information Processing Systems*, page 656–666. Curran Associates, Inc.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Aditya Deshpande, Jyoti Aneja, Liwei Wang, Alexander G Schwing, and David Forsyth. 2019. Fast, diverse and accurate image captioning guided by part-of-speech. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10687–10696.

Jerry A. Fodor and Ernie Lepore. 2002. *The compositionality papers*. Oxford University Press.

Zenzi M Griffin and Kathryn Bock. 2000. What the eyes say about speaking. *Psychological science*, 11(4):274–279.

Longteng Guo, Jing Liu, Xinxin Zhu, Peng Yao, Shichen Lu, and Hanqing Lu. 2020. Normalized and geometry-aware self-attention network for image captioning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10324–10333.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

MD. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2019. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys*, 51(6).

Jingyi Hou, Xinxiao Wu, Wentian Zhao, Jiebo Luo, and Yunde Jia. 2019. Joint syntax representation learning and visual cue translation for video captioning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8917–8926.

Ozan Irsoy and Claire Cardie. 2014. Deep recursive neural networks for compositionality in language. In *Advances in Neural Information Processing Systems*, volume 27, pages 2096–2104. Curran Associates, Inc.

Keizo Kato, Yin Li, and Abhinav Gupta. 2018. Compositional learning for human object interaction. In *European Conference on Computer Vision*, pages 247–264. Springer.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *ArXiv*, abs/1412.6980.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73.

Polina Kuznetsova, Vicente Ordonez, Alexander Berg, Tamara Berg, and Yejin Choi. 2012. Collective generation of natural image descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 359–368, Jeju Island, Korea. Association for Computational Linguistics.

Polina Kuznetsova, Vicente Ordonez, Tamara L. Berg, and Yejin Choi. 2014. Treetalk: Composition and compression of trees for image descriptions. *Transactions of the Association for Computational Linguistics*, 2:351–362.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2873–2882, Stockholmsmässan, Stockholm Sweden. PMLR.

Nannan Li, Zhenzhong Chen, and Shan Liu. 2019. Meta learning for image captioning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):8626–8633.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.

Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. 2017. Improved image captioning via policy gradient optimization of spider. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 873–881.

João Loula, Marco Baroni, and Brenden Lake. 2018. Rearranging the familiar: Testing compositional generalization in recurrent networks. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 108–114, Brussels, Belgium. Association for Computational Linguistics.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018. Neural baby talk. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7219–7228.

Emiel van Miltenburg, Desmond Elliott, and Piek Vossen. 2018. Measuring the diversity of automatic image descriptions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1730–1741, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Ishan Misra, Abhinav Gupta, and Martial Hebert. 2017. From red wine to red tomato: Composition with context. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1160–1169.

Margaret Mitchell, Jesse Dodge, Amit Goyal, Kota Yamaguchi, Karl Stratos, Xufeng Han, Alyssa Mensch, Alex Berg, Tamara Berg, and Hal Daumé III. 2012. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756, Avignon, France. Association for Computational Linguistics.

Maria Nădejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Predicting target language CCG supertags improves neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 68–79, Copenhagen, Denmark. Association for Computational Linguistics.

Mitja Nikolaus, Mostafa Abdou, Matthew Lamm, Rahul Aralikatte, and Desmond Elliott. 2019. Compositional generalization in image captioning. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 87–98, Hong Kong, China. Association for Computational Linguistics.

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

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Ehud Reiter. 1994. Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation*.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

M. Steedman and J. Baldridge. 2006. Combinatory categorial grammar. In Keith Brown, editor, *Encyclopedia of Language & Linguistics (Second Edition)*, second edition edition, pages 610 – 621. Elsevier, Oxford.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, pages 3104–3112. Curran Associates, Inc.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *2015 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *2015 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164.

Yufei Wang, Zhe Lin, Xiaohui Shen, Scott Cohen, and Garrison W Cottrell. 2017. Skeleton key: Image captioning by skeleton-attribute decomposition. In *2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7378–7387.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2048–2057, Lille, France. PMLR.

Zhenyu Yang and Qiao Liu. 2020. Att-bm-som: A framework of effectively choosing image information and optimizing syntax for image captioning. *IEEE Access*, 8:50565–50573.

Yanzhi Yi, Hangyu Deng, and Jinglu Hu. 2020. Improving image captioning evaluation by considering inter references variance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 985–994, Online. Association for Computational Linguistics.

Masashi Yoshikawa, Hiroshi Noji, and Yuji Matsumoto. 2017. A* CCG parsing with a supertag and dependency factored model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 277–287, Vancouver, Canada. Association for Computational Linguistics.

Wei Zhao, Benyou Wang, Jianbo Ye, Min Yang, Zhou Zhao, Ruotian Luo, and Yu Qiao. 2018. A multi-task learning approach for image captioning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 1205–1211. International Joint Conferences on Artificial Intelligence Organization.

A Experimental Setup

Data In order to evaluate the compositional generalization of a model, we use training and evaluation sets such that paradigmatic gaps are observed in the training set. That is, for a concept pair $\{c_i, c_j\}$, the validation $\mathcal{D}_{val}$ and test $\mathcal{D}_{test}$ sets only contain images in which at least one of the captions contains the pair of concepts, while the complementary set – where concepts c_i and c_j can only be observed independently – is used for training $\mathcal{D}_{train}$. Specifically, following Nikolaus et al. (2019), we select the same 24 adjective–noun and verb–noun concept pairs, and split the English COCO dataset (Lin et al., 2014) into four sets, each containing six held out concept pairs (training and validation instances are drawn from `train2014`, while test instances from `val2014`). Table 6 lists the sizes (in number of images) of each split.[5] For more details, we refer the reader to Nikolaus et al. (2019).

Training details Following Nikolaus et al. (2019) and Cornia et al. (2020), each system is trained with teacher forcing. Model selection is performed using early stopping, which is determined when the BLEU score of the generated captions in the validation set does not increase for five consecutive epochs.[6] All models are trained using the Adam optimizer (Kingma and Ba, 2014): BUTD and BUTR use an initial learning rate of $1e-4$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and gradients are clipped when they exceed 10.0. For the GradNorm optimizer (Chen et al., 2018) used in BUTR, the initial learning rate is 0.01 and the asymmetry is 2.5, although we find it beneficial to tune the latter when generating syntax.[7] Moreover, we find that taking the absolute value of the GradNorm weights for each loss in the renormalization step (given that our loss functions are by definition positive) leads to more stable multi-task training. $\mathcal{M}^2$-TRM first uses an initial learning rate of 1, $\beta_1 = 0.9$ and $\beta_2 = 0.98$, with a warm-up equal to $10,000$ iterations (Vaswani et al., 2017), and it is then fixed to $5e-6$ during CIDEr-D optimization. A batch size of 50 is used when training $\mathcal{M}^2$-TRM, while BUTD and BUTR are trained with batch sizes of 100. When adding syntactic forms, due to memory limitations, a batch size of 50 is always used (see Table 7 for a comparison of the planning approaches). All models are trained on one NVIDIA TitanX GPU in a shared cluster. See Table 8 for running times on the second held out dataset.

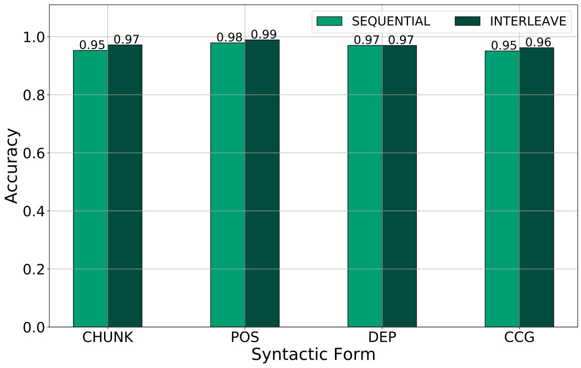


Figure 6: Validation accuracy of syntactic tags generated by BUTD using SEQUENTIAL and INTERLEAVE.

Inference At evaluation time, a maximum caption length of 20 is used when generating lexical forms only, and of 40 when also syntactic tags are generated. Notably, we use the default hyperparameters provided by the respective authors and do not fine-tune them when tasking the models with syntax generation. Differently from Nikolaus et al. (2019), rather than using a beam of 100 for BUTR only, we let all systems generate captions using such beam size as we found it to significantly improve compositional generalization of BUTD in our validation sets.[8]

B Further Analysis

Accuracy of syntactic forms We verify that a model can correctly predict syntactic forms, regardless of their granularity and the approach used to jointly modeling them with lexical forms. Figure 6 shows that, indeed, the accuracy of the generated syntactic tags, measured as the ratio of *sequences* matching the annotations from StanfordNLP, by BUTD is high, ranging between 95% and 99%. Note that we only evaluate accuracy of the SEQUENTIAL and INTERLEAVE approaches as there is no close relationship between syntactic and lexical sequences in the MULTI-TASK approach.

Qualitative examples Figure 5 shows more generated captions for images in the validation sets.

---

[5] Note that the size of each set is slightly different from the one in Nikolaus et al. (2019) as they used different tools for tokenizing and parsing, while we use a single framework to maximize its performance when parsing the captions (used to identify concept pair candidates).

[6] When present, syntactic forms are stripped away when computing evaluation metrics such as BLEU scores.

[7] Searched over the minimal grid: $\alpha \in \{1.5, 2.0, 2.5, 4.0\}$.

[8] BUTD R@5: 7.8 (beam size 5), 9.5 (beam size 100).

	Held out pairs	$\mathcal{D}_{train}$	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$
1	black cat, big bird, red bus, small plane, eat man, lie woman	$79,847$	$2,936$	$1,349$
2	brown dog, small cat, white truck, big plane, ride woman, fly bird	$79,823$	$2,960$	$1,355$
3	white horse, big cat, blue bus, small table, hold child, stand bird	$79,922$	$2,861$	$1,459$
4	black bird, small dog, white boat, big truck, eat horse, stand child	$79,627$	$3,156$	$1,501$

Table 6: Held out word pairs in each dataset split. Dataset sizes are measured in number of images.

Approach	**T**	d(S_t, W_t)	**N**
SEQUENTIAL	2×	T	1×
INTERLEAVE	2×	1	1×
MULTI-TASK	1×	–	2×

Table 7: Comparison of the planning approaches studied in this paper in terms of the sequence lengths used to train the captioning models (**T**), the distance between the syntactic tag and its corresponding word d(S_t, W_t), and the number of training examples per epoch (**N**).

Model	**Training time [hour]**
BUTD	6
+POS	9
BUTR$_{weight}$	18
+POS	17
$\mathcal{M}^2$-TRM	383
+POS	414

Table 8: Training times on held out dataset 2.

Concept pair	**BUTD**	**+POS**	**BUTR$_{weight}$**	**+POS**	**$\mathcal{M}^2$-TRM**	**+POS**
black cat	16.1	15.3	26.0	29.1	5.0	10.9
eat man	19.0	24.5	26.5	26.5	23.8	29.1
small plane	0.2	0.0	0.0	0.4	0.6	0.0
red bus	16.1	24.0	51.8	54.2	13.3	24.9
big bird	2.4	0.9	0.9	0.9	0.5	0.0
lie woman	17.5	18.5	27.1	20.8	14.9	19.8
ride woman	27.3	30.2	23.5	27.4	28.5	30.8
white truck	17.6	30.6	23.3	29.4	9.4	12.7
fly bird	21.8	25.5	26.3	32.5	29.2	35.8
small cat	1.2	0.4	1.2	1.6	0.4	0.0
brown dog	1.7	0.5	4.8	7.8	0.9	1.2
big plane	0.5	0.6	3.7	8.7	0.3	0.0
stand bird	18.4	18.6	16.9	24.0	16.8	19.5
white horse	4.9	10.9	16.9	21.3	10.1	10.1
small table	0.0	0.0	0.0	0.0	0.0	0.0
big cat	0.0	0.0	0.0	0.0	0.0	0.0
blue bus	6.5	22.7	21.3	33.2	23.5	32.1
hold child	17.8	19.8	10.3	13.8	14.8	13.0
big truck	0.0	0.7	2.9	0.7	0.0	1.0
white boat	1.6	3.5	3.3	4.9	1.6	1.9
small dog	0.0	0.1	1.2	0.9	0.0	0.3
eat horse	23.0	22.1	44.6	39.4	49.3	56.3
stand child	9.1	10.4	13.0	10.3	5.7	7.4
black bird	4.9	4.4	12.3	6.4	4.9	9.3

Table 9: R@5 for each of the held out concept pairs in the validation sets.

BUTD: a man standing in front of a pizza
BUTD + SEQUENTIAL: a couple of people that are standing around a pizza
BUTD + INTERLEAVE: a man *sitting* at a table with a pizza
BUTD + MULTI-TASK: a man *sitting* at a table with a pizza

BUTD: a baby laying on a bed with a teddy bear
BUTD + SEQUENTIAL: a couple of kids sitting on a bed
BUTD + INTERLEAVE: a large teddy bear sitting on top of a bed
BUTD + MULTI-TASK: two children sitting on a bed with a laptop

---

BUTD: a man sitting on a bed using a laptop computer
BUTD + POS: a person sitting on a bed with a laptop

BUTR$_{weight}$: a woman sitting on a bed and using a laptop
BUTR$_{weight}$ + POS: a woman sitting on a bed with a laptop and a laptop

$\mathcal{M}^2$-TRM: a man sitting on a couch using a laptop computer
$\mathcal{M}^2$-TRM + POS: a woman sitting on a bed using a laptop computer

---

BUTD: a plane sitting on top of a lush green field
BUTD + POS: a red and white plane in an open field

BUTR$_{weight}$: a red and white plane taking off on a field
BUTR$_{weight}$ + POS: a red and white plane on a lush green field

$\mathcal{M}^2$-TRM: an airplane is on the runway in a field
$\mathcal{M}^2$-TRM + POS: a red and white plane sitting on a runway

---

BUTD: a couple of sheep standing on top of a lush green field
BUTD + POS: a couple of animals standing in the grass

BUTR$_{weight}$: two small lambs standing in a green field
BUTR$_{weight}$ + POS: two baby bears standing in a grassy field

$\mathcal{M}^2$-TRM: a mother sheep and a baby sheep in a field
$\mathcal{M}^2$-TRM + POS: a baby sheep standing next to an adult sheep in

Figure 5: More examples of generated captions from the validation sets. While syntax-aware approaches generate more accurate captions overall, they are sometimes worse than the standard system (second example). Moreover, the last example shows how most systems confuse a kitty and a puppy with two sheep, lambs or bears.

Is "hot pizza" Positive or Negative?
Mining Target-aware Sentiment Lexicons

Jie Zhou[1,2], **Yuanbin Wu**[2], **Changzhi Sun**[3], **and Liang He**[1,2]

[1]Shanghai Key Laboratory of Multidimensional Information Processing
East China Normal University, China
[2]School of Computer Science and Technology East China Normal University, China
[3]ByteDance AI Lab
`jzhou@ica.stc.sh.cn, {ybwu,lhe}@cs.ecnu.edu.cn`
`sunchangzhi@bytedance.com`

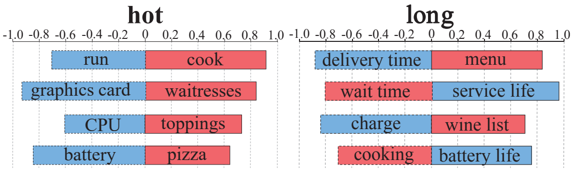
Abstract

Modelling a word's polarity in different contexts is a key task in sentiment analysis. Previous works mainly focus on domain dependencies, and assume words' sentiments are invariant within a specific domain. In this paper, we relax this assumption by binding a word's sentiment to its collocation words instead of domain labels. This finer view of sentiment contexts is particularly useful for identifying commonsense sentiments expressed in neutral words such as "big" and "long". Given a target (e.g., an aspect), we propose an effective "perturb-and-see" method to extract sentiment words modifying it from large-scale datasets. The reliability of the obtained target-aware sentiment lexicons is extensively evaluated both manually and automatically. We also show that a simple application of the lexicon is able to achieve highly competitive performances on the unsupervised opinion relation extraction task.

1 Introduction

Sentiments of words can be subtle. We are used to using the same word to express different emotions in different contexts. "Hot", for example, suggests a negative sentiment when commenting a computer hardware and a positive sentiment when commenting a pizza, even itself alone is identified without any general orientation. In these situations, it is the composition of a word, contexts, and commonsense carries an opinion. Automatically detecting such context dependent sentiments would strengthen both our understanding of implicit opinions in languages and improve existing sentiment analyses models, which is the main topic of this work.

To handle shifts of word sentiment, prior works studied how to adapt existing sentiment lexicons to new domains (Hamilton et al., 2016; Xing et al., 2019). By modeling differences and similarities of text topics, they can detect new sentiments of words as the domain changes. The basic assumption of those domain-level sentiment lexicons is that a word keeps a consistent sentiment within a domain. This assumption, however, might be strong for fine-granularity analyses of text sentiments: words (especially, neural words such as "long", "fast") could exhibit different orientations even in the same domain (Figure 1).



(a) (b)

Figure 1: Visualization of real-world commonsense sentiment of "hot" and "long" extracted by our framework. Red and blue indicate the targets in restaurant and electronic domains, respectively.

To collect more detailed information of a sentiment, another branch of works (aspect-based sentiment analysis (Pontiki et al., 2014; Zhou et al., 2020a,b), opinion relation extraction (Sun et al., 2017)) attempt find answers of "who express what opinion on which target" for opinion bearing texts. Existing solutions heavily rely on manual annotations and linguistic rules, which are either hard to scale-up or hard to be complete.

In this work, we study the task of extracting *target-aware* sentiment lexicons. An entry of such lexicon is a pair of a sentiment word and a target word, and their collocation expresses a sentiment. It improves existing domain-dependent lexicons by being more concrete and accurate on describing opinions. Departing from approaches adopted in existing aspect-based analyses, we aim to build context-aware lexicons by minimizing the requirement of annotations (e.g., only document-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 608–618
April 19 - 23, 2021.

level sentiment labels) and errors from handcrafted patterns. Our method starts from a target word (e.g., an aspect in product reviews), and extract sentiment words from its local context. The main strategy is to perturb context words and see how the sentiment of the target word changes: words with high influence on the target's sentiment hold high probability of forming a collocation with the target. We accomplish this by observing the behaviour of a well-trained document-level sentiment classifier when we change the contexts of the target word. Two types of perturbations are examined, *discrete perturbation* which only requires a black-box classifier, and *continuous perturbation* which asks for network gradients. We collect evidences of each candidate pair on large datesets to ensure the reliability of the final lexicon. Finally, the polarities of a lexicon entry can also be obtained by querying the sentiment classifier.

On two online product review domains (electronic and restaurant), we evaluate the extracted target-aware lexicon both manually and automatically. Quantitative and qualitative results show that the lexicons are reasonable to reflect common sentiment usage in each domain. As an application, we apply the lexicons to the task of unsupervised opinion relation extraction. The model performs significantly better than the baseline extractor, and even competitive with a recent supervised model on restaurant reviews. We summarize main contributions as follows,

- We propose to extend general purpose opinion lexicons with target constraints which provides a finer view on word-level sentiments.
- We develop a scalable approach to automatically mine target-aware sentiment lexicon from texts without extensive annotations and elaborated linguistic rules.
- Besides manual evaluations, we propose an automatic way to evaluate the extracted lexicon with downstream tasks.
- We are able to achieve significant improvements on unsupervised opinion relation extraction task with the help of the new lexicons.

2 Definitions and the Task

Let d be a document with sentences $s_1, s_2, \ldots, s_{|d|}$ and $y \in Y$ be the sentiment label of d. [1] Given a corpus $\mathcal{D} = \{(d_i, y_i)\}_{i=1}^{|\mathcal{D}|}$ and a target word t (e.g., screen, pizza), [2] our task is to extract *target-aware opinion words* of t only using document-level sentiment labels. Precisely, we aim to output a set of triples $(t, o, \mathbf{p})$, where o is an opinion word commonly used to comment target t and $\mathbf{p} \in \mathbf{R}^{|Y|}$ is the distribution of its sentiment orientation.

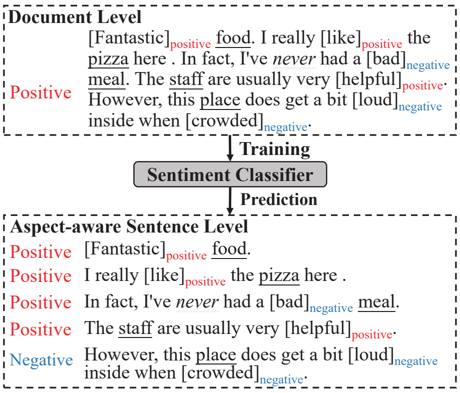


Figure 2: The process of distant supervision.

We develop the lexicon extractor in three steps. First, we build an approximate target-level sentiment classifier (Section 3) using document-level sentiment labels. Second, for each sentence s containing target t, we calculate how important a word $w \in s$ is on helping the classifier correctly predicting s's polarity (Section 4.1 and 4.2). We aggregate scores of w over all its occurrences to get its confidence of being an opinion word of t. Finally, we derive the polarity of w by querying the classifier with template sentences (Section 5).

3 Approximating Target-oriented Opinion

To identify target-aware opinion words, our key approach is to inspect how the opinion of a target changes when its context words change. Hence, it is crucial to know the polarity of a target in documents. However, annotations in D are document-level: for a document, its sentiment label expresses overall sentiments for all targets in the document, rather than a specific one. For example, the restaurant review in Figure 2 talks about 5 targets, each of them is commented by different opinion words with different polarities. In one of our datasets, 93% of documents contain multiple sentences (6 in average), and more than 82% contain multiple targets (7 in average). Therefore, directly using

---

[1] We use the 5-level label set $Y = \{1, 2, 3, 4, 5\}$ (the larger a number, the more positive it represents).

[2] Here we mainly focus on online reviews, but the methods could be applied to other sentiment-bearing texts.

document-level sentiment labels could be inappropriate for target-level analyses. On the other hand, it is quite expensive to annotate target-level sentiments, and existing datasets are far from enough for a robust commonsense opinion extractor.

To deal with this problem, we borrow the idea of distant supervision (Mintz et al., 2009): if a document is labelled as positive, at least one sentence (target) in it is positive. By seeing a large amount of positive documents, a classifier may be able to generalize patterns of their positive sentences, thus may help finding sentence-level (target-level) opinions. Here we simply build a document-level sentiment classifier, and apply it on sentences to get pseudo target-level sentiment labels (for simplicity, we assume one sentence contains one target). Advanced distant supervision models could also be applied, but we find this simple method preforms quite well in our experiments.

To build the sentiment classifier, we fine-tune BERT (Devlin et al., 2019) on $\mathcal{D}$ to encode domain specific semantics and augment it with a sentiment prediction task to encode sentiment information. For a document d, we feed its word sequence into BERT and obtain a vector representation $\mathbf{d} = \mathrm{BERT}(d)$, then we apply a $\mathrm{softmax}$ operator on $\mathbf{d}$ to get the probability of its sentiment $P(y|d)$,

$$P(y|d) = \mathrm{softmax}(W_c\mathbf{d} + b_c), \tag{1}$$

where W_c, b_c are new parameters for the sentiment classification task. The loss function is the cross-entropy between the predicted probability and the true label,

$$L = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \log P(y_i|d_i). \tag{2}$$

For each sentence s containing t, we apply above classifier to predict pseudo sentiment label y_p of s. In the following sections, we will rely on the set $S_t = \{(s, y_p)|t \in S\}$ to extract target-aware opinion words of t.

4 Importance Scores

We propose two score functions for measuring a context word w's influence on the target-oriented sentiment: one is *discrete perturbation* which only requires outputs of the sentiment classifier, another is *continuous perturbation* which needs network gradients. They are also called model-free and model-based methods, respectively. Both of them are simple and easy to compute given the trained model, and thus suitable for large-scale collective analyses.

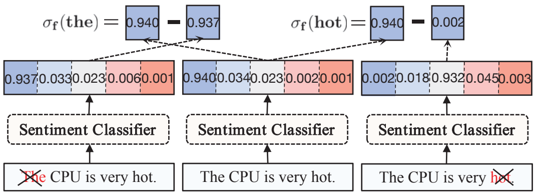


Figure 3: The possibility of the sentence is super negative changed from 0.940 to 0.002 when the word "hot" is deleted.

4.1 Discrete Perturbation

A well-trained sentiment classifier should correctly capture correlations between sentence words and sentence polarities. Intuitively, an opinion word (of the target) would have high influence on the sentiment distribution $P(y_p|s)$. For example, in Figure 3, "hot" is more informative than "The" for predicting the sentence's negative label.

In order to see whether a word w affects $P(y_p|s)$, we perturb the sentence s by removing w from it (denoted by s_{-w}) and examine the output differences,

$$\sigma_f(w, s) = P(y_p|s) - P(y_p|s_{-w}).$$

The larger $\sigma_f(w, s)$ is, the more $P(y_p|s_{-w})$ changes, and the more important w for getting the right sentiment label. We will use $\sigma_f(w, s)$ as an indicator of target-aware opinion words, and aggregate them on $\mathcal{D}$. Let $S_t^w \subseteq S_t$ be the set of sentences which t and w co-occur, we average $\sigma_f(w, s)$ on S_t^w to get the model-free importance score $\sigma_f(w)$,

$$\sigma_f(w) = \log P(w|t)\frac{1}{|S_t^w|}\sum_{s \in S_t^w} \sigma_f(w, s). \tag{3}$$

In order to reduce the affect of noise and rare language usage, we take co-occurrence statistic into account: a target-aware opinion word should co-occur with the target often. Therefore, the average score is empirically scaled with their co-occurrence probability $P(w|t) = \frac{|S_t^w|}{|D|}$.

The score $\sigma_f(w)$ is model-free in the sense that we don't need to know details of the sentiment classifier and only inquire the difference of outputs when the input sentence is perturbed. Hence, though we use the BERT-based classifier here, we can use any other off-the-shelf sentiment classifiers (e.g., pre-trained models with different training objectives, multi-task learned classifiers, etc.) to further enrich (or constrain) the score.

4.2 Continuous Perturbation

Besides the discrete perturbation setting, we could also utilize the full classification model to identify target-aware opinion words. In this continuous perturbation setting, we ask the same question of how the sentiment prediction will change when we perturb sentence words. However, instead of perturbing them discretely (i.e., removing a word), we can perform continuous perturbations on word vectors (Goodfellow et al., 2015).

Let $L(y_p, s, \mathbf{w}) = -\log P(y_p|s)$ be the loss on sentence s and $\mathbf{w}$ is the word vector of w. If we slightly perturb $\mathbf{w}$ to $\mathbf{w}'$ with $\|\mathbf{w}' - \mathbf{w}\| \leq \varepsilon$, we can bound the absolute change of the loss function using the first-order approximation of $L(y_p, s, \mathbf{w})$,

$$\begin{aligned}
&|L(y_p, s, \mathbf{w}') - L(y_p, s, \mathbf{w})| \\
&\approx |\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})^T (\mathbf{w}' - \mathbf{w})| \\
&\leq \|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\| \|\mathbf{w}' - \mathbf{w}\| \\
&\leq \varepsilon \|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\|.
\end{aligned}$$

The magnitude of the gradient's norm $\|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\|$ could be a sign of how sensitive the sentiment label is with respect to w: to get the right prediction we will prefer not to perturb those words with large gradient norms. Therefore, a large gradient norm may also indicate an opinion words of the target. Define

$$\sigma_b(w, s) = \frac{1}{g^* - g_*} \left(\|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\| - g_* \right),$$

where $g^* = \max_{s \in S_t^w} \|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\|$, $g_* = \min_{s \in S_t^w} \|\nabla_{\mathbf{w}} L(y_p, s, \mathbf{w})\|$ are the maximum and minimum gradient norm in S_t^w, which help normalizing $\sigma_b(w, s)$ into $[0, 1]$.

Similar to Equation 3, we collect all $\sigma_b(w, s)$ in S_t^w and scale their average with co-occurrence probability. The model-based score of w is defined as,

$$\sigma_b(w) = \log P(w|t) \frac{1}{|S_t^w|} \sum_{s \in S_t^w} \sigma_b(w, s). \qquad (4)$$

Finally, the computation of both discrete perturbation and continuous perturbation could be done efficiently using auto-gradient tools. The discrete perturbation setting requires a forward process of the network, while the continuous perturbation setting needs an additional backward computation. We also note that the "perturb-and-see" strategy behind both scores characterizes the relation between opinion words and the target only through the sentiment label, which is an indirect way. As a consequence, though the scores could recognize "big" implies a negative opinion on "battery", it could also identify "not" in "the battery is not big" as an important word for the positive opinion. In practice, we could filter out such cases by rules, but how to explicitly handle semantic composition in importance scores would be an important future work.

5 Polarity Inference

Given the importance scores of words with respect to t, we can rank them accordingly and take the top-k words as t's opinion lexicon. As the final step, we are left to determine the polarity of an opinion word o. We accomplish this by building template sentences which try to carry the semantic like "what opinion on which target". We call these sentences *template* which will be use to probe the sentiment classifier's knowledge on (t, o)'s polarity.

Formally, define $\mathcal{T}$ to be a set of templates, each template $\tau \in \mathcal{T}$ takes an opinion word and a target as input, outputs a natural language sentence $\tau(t, o)$. Here, we use the following two templates,

- $\tau(t, o) =$ "The t is o." (e.g., $\tau(\text{battery}, \text{big}) =$ "The battery is big.").

- $\tau(t, o) =$ "o t." (e.g., $\tau(\text{battery}, \text{big}) =$ "big battery.").

By feeding $\tau(t, o)$ into the sentiment classifier, we obtain $P(y_p|\tau(t, o))$, and the polarity distribution $\mathbf{p}$ of (t, o) is averaged over all templates,

$$\mathbf{p} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} P(y|\tau(t, o)) \qquad (5)$$

6 Experimental Results and Analyses

We wish to evaluate the merit of our target-aware sentiment lexicon in this section. We first introduce the experimental setup in Section 6.1. Then, we design detail experiments to answer the following key questions.

	Electronic						Restaurant					
	P@5		P@10		P@20		P@5		P@10		P@20	
	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$
PMI	0.292	0.176	0.284	0.190	0.287	0.178	0.260	0.132	0.260	0.138	0.273	0.149
DP	0.556	**0.352**	0.514	**0.296**	0.460	0.244	0.952	**0.772**	0.932	0.602	0.899	**0.597**
CP	0.608	0.212	0.620	0.210	0.596	0.206	0.828	0.676	0.814	0.536	0.816	0.429
DP+CP	**0.704**	0.296	**0.686**	0.288	**0.628**	**0.262**	**0.980**	0.748	**0.960**	**0.674**	**0.927**	0.537

Table 1: The results of human evaluation on $\mathcal{L}$ and $\mathcal{L}_c$ over electronic and restaurant. DP and CP mean discrete perturbation and continuous perturbation, respectively.

- **Q1 Can we trust our target-aware sentiment lexicon?** To evaluate the quality of the extracted lexicon, we test the performance with both manual evaluation (Section 6.2) and automatic downstream task (Section 6.3).

- **Q2 Useful or not?** As an application, we apply our lexicon into unsupervised opinion extraction task in Section 6.4.

- **Q3 Do we really understand our model?** In Section 6.5, to investigate the insight of commonsense sentiment mined from the texts, we visualize several real-world examples.

6.1 Experimental Setup

We conduct experiments to validate the effectiveness of our approach on two widely different domains: electronic and restaurant, taken from Amazon dataset[3] and Yelp Challenge 2015[4]. We obtain the target set from SemEval'14, SemEval'15, and SemEval'16 for convenience [5].

The extracted target-aware sentiment lexicon ($\mathcal{L}$) can be divided into target-aware general sentiment lexicon ($\mathcal{L}_g$) and commonsense sentiment lexicon ($\mathcal{L}_c$). $\mathcal{L}_g$ means the opinion words in $\mathcal{L}$ that are in general lexicon and $\mathcal{L}_c$ means the opinion words in $\mathcal{L}$ that are not in general lexicon. Here, we use the general lexicon from (Hu and Liu, 2004) to filter the general sentiment words and obtain the commonsense lexicon. This general lexicon contains around 6800 positive and negative opinion words or sentiment words for the English language.

We adopt $\mathrm{BERT}_{\mathrm{base}}$ as the basis for all experiments. Adam (Kingma and Ba, 2015) is adopted as the optimizer with learning rate 5e-5 for fine-tuning and sentiment classification.

6.2 Human Evaluation

To evaluate the quality of the target-aware sentiment lexicon, we test its performance through human evaluation. For quantitative evaluation, we sample 50 targets with top-20 opinion words in each domain to investigate the performance of $\mathcal{L}$ and $\mathcal{L}_c$. Finally, we obtain 3122 and 2877 (t, o) pairs after filtering repetitive pairs for electronic and restaurant, respectively. We ask ten annotators to label them to make sure each pair is marked with three times. Then, we obtain the label through voting. We calculate the Krippendorff's alpha coefficient (Krippendorff, 2011) to measure the inter-annotator agreement of the manual annotation. The value is 0.850 and 0.702 for restaurant and electronic, which indicates the high agreement of the labeled data.

Table 1 reports the results of the human evaluation. The pointwise mutual information (PMI) measure (Hamilton et al., 2016; Church and Hanks, 1990) is adopted as the baseline to compare with, which applied to each target t w.r.t. each word w. We adopt the precision of top-k (e.g., 5, 10, and 20) to measure the performance of the methods across both $\mathcal{L}$ and $\mathcal{L}_c$. From this table, we observe that: **First**, both our discrete perturbation and continuous perturbation algorithms perform much better than PMI. Additionally, in the restaurant domain, our model obtains more than 90% precision for $\mathcal{L}$. These indicate the great effectiveness of capturing target-aware sentiment words and commonsense sentiment words. **Second**, the discrete perturbation method often has higher precision than continuous perturbation method, but the combination of them (Discrete+Continuous Perturbation) [6] obtains the best results in most cases. It suggests that the discrete perturbation and the continuous perturbation settings may focus on different types of opinion words.

---

[3] http://jmcauley.ucsd.edu/data/amazon/

[4] https://www.yelp.com/dataset/challenge

[5] Here we use the targets from existing datasets, but the targets could be extracted automatically through existing work (Poria et al., 2014) or be inputted by users.

[6] Here we simply calculate the average score of them, but different weights can be used.

	Electronic	Restaurant
Original	93.82	91.64
Random-based Deleting	90.49	88.13
Lexicon-based Deleting	$84.77^{\dagger\ddagger}$	$80.67^{\dagger\ddagger}$

Table 2: The results of downstream task: sentiment analysis via **Strategy 1**. The marker † and ‡ refer to p-values < 0.05 when "original" and "random-based deleting" compare with "lexicon-based deleting".

6.3 Downstream Tasks

Besides human evaluation, we also automatically evaluate our commonsense sentiment lexicon $\mathcal{L}_c$ with downstream tasks. Here we examine document-level sentiment analysis. In particular, for each domain, we sample 3500 documents which do not contain any general sentiment lexicon words but have obvious opinion orientations on electronic and restaurant ("Original"). Then we perform sentiment classification on the dataset with $\mathcal{L}_c$ using two strategies.

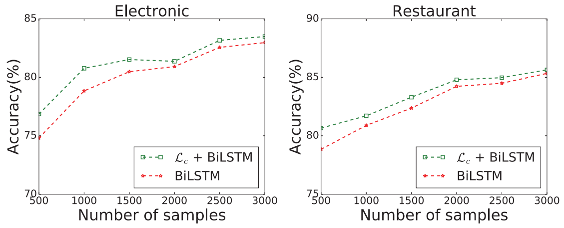
Strategy 1 For each sample in "Original", we remove opinion words which appear in our $\mathcal{L}_c$, and test the performance of sentiment classification using a well-trained sentiment classifier (Section 3). Note that we only use the top-100 opinion words to make sure only fewer than five words are being deleted. To show the effectiveness of our lexicon, we compare our model with removing words randomly with the same rate (Table 2). We find that removing the words in $\mathcal{L}_c$ performs significantly worse than both the original and random removing. It indicates that our method can capture the commonsense opinion words effectively.

Strategy 2 We apply our commonsense lexicon as extra knowledge to enhance a sentiment classification model. Here, we study the standard BiLSTM-based classifier: a BiLSTM is used to encode sentences, the last hidden vector of a sentence is adopt for classification. To inject our extracted lexicon $(t, o, \mathbf{p})$, we concatenate $\mathbf{p}$ to the input of BiLSTM if t and o occur. We sample 1000 and 500 instances from previous 3500 samples as the training and test set. To validate the effectiveness of each model components, we also show ablation test results. Table 3 shows the results. We have the following observations.

- Our commonsense lexicon $\mathcal{L}_c$ can significantly improve the performance of sentiment classification. $\mathcal{L}_c$ + BiLSTM outperforms basic BiLSTM, while the model with PMI is even worse than BiLSTM. We also find the results of the discrete perturbation and continuous perturbation method are similar, and both of them can improve the results of sentiment classification.

	Electronic	Restaurant
BiLSTM	78.84	80.89
$\mathcal{L}_c$ + BiLSTM	$80.77^{\triangle}$	$81.71^{\triangle}$
$\mathcal{L}_c$ + BiLSTM - $\mathbf{p}$	79.56	81.11
$\mathcal{L}_c$ + BiLSTM (PMI)	78.75	80.63
$\mathcal{L}_c$ + BiLSTM (w/o DP)	80.11	81.45
$\mathcal{L}_c$ + BiLSTM (w/o CP)	80.01	81.50
$\mathcal{L}_c$ + BiLSTM (w/o DS)	79.57	80.94

Table 3: The results of sentiment analysis via **Strategy 2**. The marker $^{\triangle}$ refers to p-values < 0.05 comparing with "BiLSTM". DP and CP mean discrete perturbation and continuous perturbation, respectively. DS represents distant supervision.



(a) (b)

Figure 4: The results of sentiment analysis with different sample number.

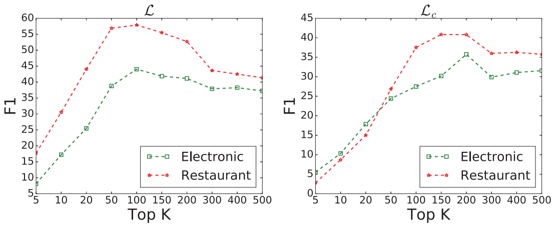
- $\mathcal{L}_c$ + BiLSTM performs better than the corresponding model without distant supervision, which indicates our distant supervision can capture the target information effectively. To further verify the effectiveness of distant supervision, we also randomly select 200 samples from the set S_t and evaluate them with three annotators by voting. The accuracy is 80.5% and 82% for 5-class classification over electronic and restaurant domains. Additionally, there are 71% and 65% of the samples have different polarities with their document-level label, and the accuracy of these samples is 80.99% and 81.54% in electronic and restaurant domains. These indicate our distant supervision can learn the target-oriented sentiment effectively.

- Compared with $\mathcal{L}_c$ + BiLSTM - $\mathbf{p}$ (which takes whether a word is an opinion word as feature), $\mathcal{L}_c$ + BiLSTM obtains better results. It suggest that polarity inference might be reasonable to infer the polarities of (t, o) pairs.

Additionally, to investigate the influence of sample numbers, we draw the results with different sample numbers in Figure 4. We can find that the

	Electronic			Restaurant		
	P	R	F1	P	R	F1
Rule-based	50.13	33.86	40.42	58.14	42.71	47.39
Ours	46.12	42.13	44.04	53.93	62.47	57.89
LSTM	55.71	57.53	56.52	57.46	64.96	60.87

Table 4: The results of opinion extraction. Note that LSTM is a supervised method.



(a) (b)

Figure 5: The results of unsupervised opinion extraction with different top-k via $\mathcal{L}$ and $\mathcal{L}_c$.

fewer samples, the more improvement by our commonsense lexicon.

6.4 Application (Unsupervised Opinion Extraction)

To answer **Q2**, we apply our lexicon into unsupervised opinion relation extraction. We test our lexicon on two datasets[7]: electronic and restaurant, which are released by (Fan et al., 2019), who labeled the opinion words towards the given target.

To investigate the performance of the target-aware sentiment lexicon $\mathcal{L}$, we perform unsupervised opinion extraction on the whole dataset. Table 4 reports the experimental results. We compare our method with two methods: 1) rule-based method (Hu and Liu, 2004) use the distance and POS tags to determine the opinion words; 2) supervised LSTM was proposed by (Liu et al., 2015). We use the results reported in (Fan et al., 2019) here. From this table, we observe: **First**, our $\mathcal{L}$ performs significantly better than the rule-based method even without using any rules or human annotations. **Second**, our unsupervised method is comparable with the supervised method (e.g., LSTM) in Restaurant. Additionally, we explore the influence of top-k in Figure 5 (a). We can find that top-100 is recommended for $\mathcal{L}$ in our experiments.

To verify our method can extract commonsense opinion words accurately, we also evaluate our $\mathcal{L}_c$ on the samples without general words. From Figure 5 (b), we can find that $\mathcal{L}_c$ achieves 40% F1 on the restaurant domain. Considering that we don't include any general sentiment words, we think the result is quite promising.

6.5 Case Studies

To investigate the insight of commonsense sentiment mined from texts, we show several real-world examples in electronic and restaurant in this section. We present some interesting discoveries through in-depth analysis as follows.

We explore the sentiment polarity of different targets with the same opinion word here. As shown in Figure 1, we draw the targets w.r.t. opinion words "hot" and "long". We obtain the following interesting findings. **First**, our model can detect the commonsense sentiment in the corpus effectively. For example, our model can find that "hot" is a common-used collocation for "pizza", "CPU", and "battery", and it expresses a positive sentiment for "pizza", while it represents a negative sentiment for "CPU" and "battery". **Second**, domain-dependent sentiment words and their orientations are insufficient, and both the target and the opinion words are essential. For example, "long" has a positive polarity for "battery life" and negative sentiment for "charge" even both "battery life" and "charge" are in the electronic domain.

The opinion words most related to the given target (top-10) in $\mathcal{L}$ and $\mathcal{L}_c$ are shown in Table 5. From this table, we obtain the following discoveries. **First**, our method captures not only the general opinion words but also the commonsense opinion words. **Second**, as mentioned in Section 4.2, though the scores could recognize "fast" expresses a positive opinion on "response", it also identifies the words are important for sentiment but not opinion words, such as "no", "not" and "never". In practice, we could filter out such cases by rules, but how to explicitly handle semantic composition in importance scores would be an important future work.

From Table 5, we observe that $\mathcal{L}$ and $\mathcal{L}_c$ for different targets are quite different. To investigate whether the common-used opinion words for different targets are different, we measure it by,

$$\text{div} = \frac{1}{|T|(|T|-1)} \sum_{\substack{i,j=0 \\ i \neq j}}^{|T|} 1 - \frac{|\mathcal{L}^{t_i} \cap \mathcal{L}^{t_j}|}{|\mathcal{L}^{t_i} \cup \mathcal{L}^{t_j}|} \qquad (6)$$

where T is the set of targets in our dataset, t_k is the k-th target in T and $\mathcal{L}^{t_k}$ means the sentiment lexicon of t_k. The value of div is 0.65 and 0.90 (0.89

---

[7] https://github.com/NJUNLP/TOWE

	Electronic				Restaurant			
	Response		Memory		Workers		Service	
	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$	$\mathcal{L}$	$\mathcal{L}_c$
1	fast$_+$	fast$_+$	hesitate$_*$	hesitate$_*$	rude$_*$	attitude	worst$_*$	never
2	quickly$_+$	quickly$_+$	perfectly$_+$	recent$_+$	attitude	extremely$_+$	horrible$_*$	zero$_*$
3	excellent$_+$	quite	recent$_+$	class$_+$	terrible$_*$	not	terrible$_*$	beat$_+$
4	quite	longer$_*$	class$_+$	crucial$_+$	extremely$_+$	greeted$_+$	amazing$_+$	extremely$_+$
5	longer$_*$	faster$_+$	crucial$_+$	suggest$_+$	super$_+$	professional$_+$	disappointed$_*$	5$_+$
6	faster$_+$	sometimes$_*$	suggest$_+$	frame	friendly$_+$	ignored$_*$	outstanding$_+$	above
7	sometimes$_*$	no	proprietary$_+$	fastest$_+$	helpful$_+$	fast$_+$	awful$_*$	average
8	no	appropriately$_+$	prefer$_+$	faster$_+$	nice$_+$	dressed$_+$	excellent$_+$	professional$_+$
9	appropriately$_+$	remote	limited$_*$	swap	efficient$_+$	welcoming$_+$	exceptional$_+$	five$_+$
10	softer$_+$	plugged$_*$	perfect$_+$	kingston	incompetent$_*$	friendliest$_+$	sucks$_*$	fast$_+$

Table 5: We list top-10 opinion words of several targets for two domains: electronic and restaurant. The marker $_+$ and $_*$ represent positive and negative sentiment respectively.

and 0.96) for $\mathcal{L}$ and $\mathcal{L}_c$ over restaurant (electronic). All these indicate that commonsense lexicon $\mathcal{L}_c$ is more diverse than general lexicon $\mathcal{L}_g$ over different targets. In addition, the commonly used general opinion words and commonsense sentiment words are different for different targets.

7 Related Work

Domain adaptation has been studied for a long time in the field of sentiment analysis (Wu et al., 2017; Choi and Cardie, 2009; Cambria et al., 2018; Zhou et al., 2020c). We mainly summarize the related work about lexicon domain adaptation that aims to build a domain-specific sentiment lexicon (Ofek et al., 2016; Vo and Zhang, 2016; Hamilton et al., 2016). In (Hamilton et al., 2016), authors inferred the orientation of words from general opinion words by building a graph for each domain. Xing et al. (2019) judged the word polarity via a document-level sentiment classifier. However, it is time-consuming for they have to retrain the model for each word after changing the polarity randomly. Moreover, these existing methods mainly focus on the domain-level, while the sentiment polarities of some words depend on their opinion targets (Liu and Zhang, 2012). It is essential to predict the sentiment in target-level by integrating both target and opinion words.

The most related work to us is (Zhao et al., 2012). Zhao et al. (2012) focused on inferring the polarity of a binary tuple of a polarity word and a target via search engine, while target-aware opinion words extraction is not fully explored. To take the target into account, Wu et al. (2019) proposed to construct a target-specific sentiment lexicon. However, both NLP preprocessing pipelines (e.g., parsing, POS tagging) and linguistic rules are integrated into their algorithm. Different from them, we first extract the target-aware commonsense opinion words via pre-trained models, which learned rich commonsense knowledge hidden in human languages. Then, we predict the sentiment polarity of target and opinion word pair through a probing strategy. We focus on building context-aware lexicons by minimizing the requirement of annotations and handcrafted external resources.

To take the target into account, Wu et al. (2019) proposed to construct a target-specific sentiment lexicon. However, both NLP preprocessing pipelines (e.g., parsing, POS tagging) and linguistic rules are integrated into their algorithm. Available resources like general sentiment lexicon and thesaurus are also made used. Since it is not easy to apply on different domains, we develop a framework to automatically mine aspect-aware commonsense sentiment from texts without extensive annotations and elaborated linguistic rules.

Pre-trained models (e.g., ELMo (Peters et al., 2018), GPT (Radford et al., 2019), BERT (Devlin et al., 2019)) have achieved great success in NLP recently. By exploring a large number of open domain texts, pre-trained models are able to encode rich semantic information hidden in human languages and thus provide new powerful tools for knowledge mining and extraction (Davison et al., 2019; Petroni et al., 2019). Since the commonsense opinions are closely related to human commonsense and background knowledge, we adopt pre-trained language models to mine the commonsense sentiment from texts automatically.

Gradient-based methods (Goodfellow et al., 2015) have been widely applied into computer version and NLP (Zeiler and Fergus, 2014; Liang et al., 2018). The gradient-based approach is also used to understand the decisions of the text classification models from the token level (Li et al., 2016; Alikaniotis et al., 2016). In addition, Rei et al. (2018)

adopted gradient-based approach to detect the important tokens in the sentence via the sentence-level label. In this paper, we design a continuous perturbation algorithm to discover the target-aware opinion words using the gradient.

8 Conclusion

In this paper, we propose a framework for automatic target-aware sentiment mining from texts without manual annotations or linguistic rules. We evaluate the proposed framework on two large-scale online review domains: restaurant and electronic with both manual checking and automatic downstream tasks. We also achieve significant improvements by applying the opinion lexicon to the task of unsupervised opinion relation extraction. To investigate the insight of commonsense sentiment mined from the texts, we visualize several real-world examples and analyze them in-depth. The extensive experimental results demonstrate the excellent performance in building a target-aware sentiment lexicon.

Acknowledgements

The authors wish to thank the reviewers for their helpful comments and suggestions. This research is (partially) supported by NSFC (62076097), STCSM (18ZR1411500). This work is also supported by National Key R&D Program of China (No.2018AAA0100503&2018AAA0100500). The computation is performed in ECNU Multifunctional Platform for Innovation(001). The corresponding authors are Yuanbin Wu and Liang He.

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. 2018. Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1795–1802. AAAI Press.

Yejin Choi and Claire Cardie. 2009. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 590–598. ACL.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Comput. Linguistics*, 16(1):22–29.

Joe Davison, Joshua Feldman, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1173–1178. Association for Computational Linguistics.

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

Zhifang Fan, Zhen Wu, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2509–2518. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 595–605. The Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177. ACM.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Jiwei Li, Xinlei Chen, Eduard H. Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 681–691. The Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4208–4215. ijcai.org.

Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 415–463. Springer.

Pengfei Liu, Shafiq R. Joty, and Helen M. Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1433–1443. The Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 1003–1011. The Association for Computer Linguistics.

Nir Ofek, Soujanya Poria, Lior Rokach, Erik Cambria, Amir Hussain, and Asaf Shabtai. 2016. Unsupervised commonsense knowledge enrichment for domain-specific sentiment analysis. *Cognitive Computation*, 8(3):467–477.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Soujanya Poria, Erik Cambria, Lun-Wei Ku, Chen Gui, and Alexander F. Gelbukh. 2014. A rule-based approach to aspect extraction from product reviews. In *Proceedings of the Second Workshop on Natural Language Processing for Social Media, SocialNLP@COLING 2014, Dublin, Ireland, August 24, 2014*, pages 28–37. Association for Computational Linguistics and Dublin City University.

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

Marek Rei and Anders Søgaard. 2018. Zero-shot sequence labeling: Transferring knowledge from sentences to tokens. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 293–302. Association for Computational Linguistics.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, and Qi Zhang. 2017. Large-scale opinion relation extraction with distantly supervised neural network. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1033–1043, Valencia, Spain. Association for Computational Linguistics.

Duy-Tin Vo and Yue Zhang. 2016. Don't count, predict! an automatic approach to learning sentiment lexicons for short text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*, pages 219–224. The Association for Computer Linguistics.

Fangzhao Wu, Yongfeng Huang, and Jun Yan. 2017. Active sentiment domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association* *for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1701–1711. Association for Computational Linguistics.

Sixing Wu, Fangzhao Wu, Yue Chang, Chuhan Wu, and Yongfeng Huang. 2019. Automatic construction of target-specific sentiment lexicon. *Expert Systems with Applications*, 116:285–298.

Frank Z. Xing, Filippo Pallucchini, and Erik Cambria. 2019. Cognitive-inspired domain adaptation of sentiment lexicons. *Information Processing & Management*, 56(3):554–564.

Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer.

Yanyan Zhao, Bing Qin, and Ting Liu. 2012. Collocation polarity disambiguation using web-based pseudo contexts. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 160–170. ACL.

Jie Zhou, Qin Chen, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020a. Position-aware hierarchical transfer model for aspect-level sentiment classification. *Information Sciences*, 513:1–16.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020b. SK-GCN: modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification. *Knowledge-Based Systems*, 205:106292.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020c. Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 568–579. International Committee on Computational Linguistics.

Quality Estimation without Human-labeled Data

Yi-Lin Tuan[1], **Ahmed El-Kishky**[2], **Adithya Renduchintala**[2], **Vishrav Chaudhary**[2],
Francisco Guzmán[2] and **Lucia Specia**[3]
[1]University of California Santa Barbara, [2]Facebook AI, [3]Imperial College London
[1]`ytuan@cs.ucsb.edu`, [2]`{ahelk,adirendu,vishrav,fguzman}@fb.com`
[3]`l.specia@imperial.ac.uk`

Abstract

Quality estimation aims to measure the quality of translated content without access to a reference translation. This is crucial for machine translation systems in real-world scenarios where high-quality translation is needed. While many approaches exist for quality estimation, they are based on supervised machine learning requiring costly human labelled data. As an alternative, we propose a technique that does not rely on examples from human-annotators and instead uses synthetic training data. We train off-the-shelf architectures for supervised quality estimation on our synthetic data and show that the resulting models achieve comparable performance to models trained on human-annotated data, both for sentence and word-level prediction.

1 Introduction

The adoption of Machine Translation (MT) has been increasing in areas ranging from government and finance, to even social media due to the substantial improvements achieved from Neural Machine Translation (NMT). However, even with improved performance, translation quality is not consistent across language pairs, domains, and sentences. This can be detrimental to end-user's trust and can cause unintended consequences arising from poor translations. Thus, having metrics to assess the quality of translated content is crucial to ensure that only high-quality translations are provided to end-users or downstream tasks. Quality Estimation (QE) metrics aim to predict translation quality without access to reference translations (Blatz et al., 2004; Specia et al., 2009, 2013).

State-of-the-art QE techniques have leveraged MT systems and language-specific human annotations as supervision, including direct assessment and post-editing (Kepler et al., 2019a; Fonseca et al., 2019; Sun et al., 2020). However, these annotations are costly and time-consuming, particularly for word-level QE, where each token needs a label.

Some unsupervised approaches take inspiration from statistical MT (Popović, 2012; Moreau and Vogel, 2012; Etchegoyhen et al., 2018) or apply uncertainty quantification (Fomicheva et al., 2020) for QE. However, their performance is inferior to that of supervised models. In related areas such as automatic post-editing, parallel data has been used to create synthetic post-editing data (Negri et al., 2018), however this technique only compares machine-translated sentences to references. Our approach augments MT errors with additional errors via masked language model rewriting.

We leverage noisy, mined comparable sentences obtained by weakly-supervised techniques (El-Kishky et al., 2020b). These noisy bitexts have been mined from a variety of domains such as Wikipedia (Schwenk et al., 2019a) and large web-crawls (Schwenk et al., 2019b; El-Kishky et al., 2020a; El-Kishky and Guzmán, 2020) and have been shown to be an invaluable source of training data for NMT models. Using this data is crucial to avoid data leakage between a trained NMT model and the data we use to create synthetic QE data. For each source-target sentence pair from the mined data, we apply an MT system to generate a candidate translation of the source sentence. Additionally we rewrite each target reference sentence using a masked language model to introduce errors. These two approaches generate two alternative "translations" of the source sentence. We then produce pseudo-labels for each token in these translations by edit distance alignment to the original reference sentence. This results in each translated word being pseudo-labelled as correct or incorrect, which is our synthetic QE training data. Analogously, sentence-level training data is derived as the proportion of incorrect words per sentence.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 619–625
April 19 - 23, 2021.

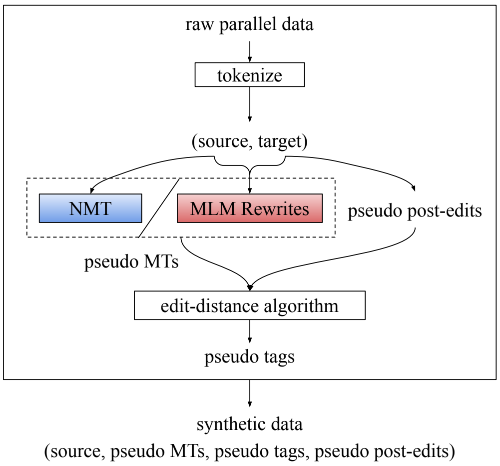


Figure 1: The pipeline to synthesize data for QE from comparable mined data.

Our **main contributions** are: (i) We explore a simple technique to effectively generate synthetic data for QE that allows for both word-level and sentence-level estimation (ii) we demonstrate that our technique performs comparably to off-the-shelf models trained on human-annotated data.

2 QE Task Description

Word-level QE has been mainly framed as the task of predicting which words in the translation need to be post-edited. As such, word-level QE aims to assign a tag for each word and gap between words in a machine-generated translation as *correct*, i.e., the word does not need editing, or *incorrect*, i.e., the words should be substituted, deleted, or inserted (tags for gaps) (Specia et al., 2020).

For word-level, we denote the tag of each word in a translation as $m_t \in \{\texttt{OK}, \texttt{BAD}\}$, where $t \in [1, T]$ and T is the length of the translation. Also, we denote the tag of each gap between two words (including the beginning and the end) as $g_t \in \{\texttt{OK}, \texttt{BAD}\}$, where $t \in [1, T+1]$.

In traditional QE, data is collected by first translating source sentences using an MT model. Second, experts post-edit these translations. Third, the post-edits and machine translations are aligned in such a way that induces the minimum edit distance between the tokens of each. Finally, each m_t is labelled as `BAD` if it should be deleted or substituted and each g_t is labelled as `BAD` if at least a word should be inserted there. Sentence-level QE labels can be generated by computing the Human-targeted Translation Error Rate (HTER) (Snover and Brent, 2001; Snover et al., 2006), which is the minimum ratio of edit operations needed to fix the translation to the number of its tokens. We explore the possibility to skip the costly human post-editing process by proposing a data synthesis pipeline, which we then test on human labelled data.

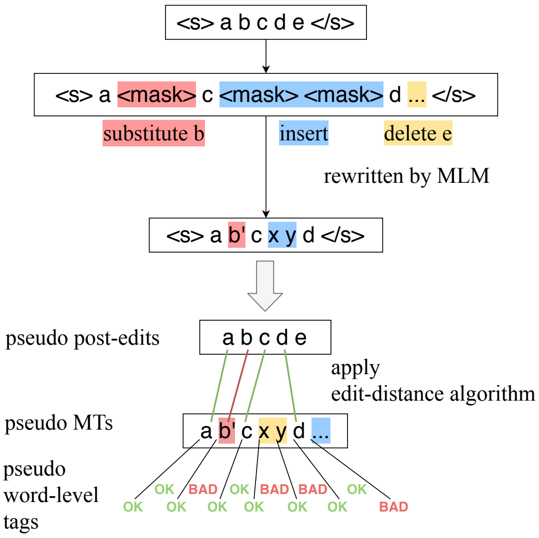


Figure 2: The rewriting process by text-infilling using a masked language model.

3 Approach to Data Synthesis

As depicted in Figure 1, we synthesize data from mined Wikipedia datasets, where each example consists of a *(source, target)* sentence pair.

We create candidate translations of source sentences in two ways: For the first approach, we apply the *NMT* model to translate each source sentence. For the second approach, we rewrite each reference target sentence using a masked language model (MLM), as shown in the *MLM Rewrites* block in Figure 1. The two approaches create two forms of translations. Then, by treating target sentences as if they were post-edited data (*pseudo post-edits*), we identify errors in each candidate translation by looking at the insertions, deletions, and substitutions between the references and generated translations.

Neural Machine Translation. For the first approach to generating synthetic data, we use a pre-trained NMT model to create translations. The NMT model is the same model that was used to generate translations in the supervised data; the architecture is a standard transformer as used in (Vaswani et al., 2017; Ott et al., 2019). The process of creating synthetic QE data first involves translating each source sentence using this model and taking the output as a translation which will

Data	English-German			English-Chinese		
	size	MT bad (%)	Gap bad (%)	size	MT bad (%)	Gap bad (%)
Human annotation	7K	27.8	4.7	7K	54.2	8.4
NMT	459K	38.2	5.7	189K	49.5	6.8
MLM (word-QE)	459K	40.7	2.9	189K	53.9	8.6
MLM (sent-QE)	459K	43.1	3.3	189K	49.9	2.7

Table 1: Statistics of annotated and synthetic (NMT and MLM) data.

later be used to generate the synthetic labels. When decoding, we apply a beam of 5 following the NMT models available in Fomicheva et al. (2020) to generate a candidate translation. Next, we take the mined reference target sentence and treat it as a pseudo post-editing of the machine translation.

We then compute the edit distance between MTs and pseudo post-edits. The resulted edit operations are the pseudo tags, which consist of word tags m_t and gap tags g_t. This process is illustrated in Algorithm 1.

Algorithm 1: DataSynthesis-NMT

```
Input: pairs (source, target) from mined data,
       pretrained NMT model
Output: (MTs, pseudo tags)
for each pair (source, target) do
    MTs = NMT(source)
    $\{m_t\}_{t=1}^{T}$, $\{g_t\}_{t=1}^{T+1}$ = edit_distance(MTs, target)
    pseudo tags = ($\{m_t\}_{t=1}^{T}$, $\{g_t\}_{t=1}^{T+1}$)
    return (MTs, pseudo tags)
```

Rewriting by Masked Language Model (MLM). Our second approach to creating synthetic QE training data is to introduce errors by rewriting target sentences. We inject these errors by performing *text-infilling* (Zhu et al., 2019; Lewis et al., 2019). As displayed in Figure 2, we perform text-infilling by applying three operations: (1) randomly substituting a proportion of tokens with a `<mask>` token, (2) deleting consecutive tokens, and (3) inserting additional consecutive `<mask>` tokens. We determine the lengths of consecutive deletions and insertions by drawing them from a Poisson distribution with mean $\lambda = 1$ shifted by 1 to avoid zero-length insertions or deletions. We then use a pre-trained masked language model (MLM) supplied with the source sentence as input to infill the masked reference sentence. We select multilingual BERT (Devlin et al., 2019) as it is pre-trained on Wikipedia which is in-domain to our test set. We present the target-rewriting approach in detail in Algorithm 2.

In Section 4, we will investigate the performance of QE models trained on NMT-based synthetic data, rewriter-based synthetic data, and a two-model ensemble where each model is trained on a different form of synthetic data.

Algorithm 2: DataSynthesis-Rewriting

```
Input: pairs $(S, W)$:=(source, target) from mined
       data, pretrained MLM
Input: $P_s$, $P_d$, $P_i$ as the probabilities of substitution,
       deletion, and insertion
Output: (pseudo MTs, pseudo tags)
for each pair $(S, W)$ do
    $W'$ = randomly mask tokens in $W$ by $P_s$
    $D$ = randomly mark deletion in $W'$ by $P_d$
    $W'$ = randomly delete a text span from marks $D$
      in $W'$ (length~ $Poisson(\lambda = 1) + 1$)
    $I$ = randomly mark insertion in $W'$ by $P_i$
    $W'$ = randomly insert contiguous masks from
      marks $I$ in $W'$ (length~ $Poisson(\lambda = 1) + 1$)
    rewrites = MLM_fills_in_masks($S$, $W'$)
    $\{m_t\}_{t=1}^{T}$, $\{g_t\}_{t=1}^{T}$ = edit_distance(rewrites, $W$)
    pseudo tags = ($\{m_t\}_{t=1}^{T}$, $\{g_t\}_{t=1}^{T+1}$)
    return (rewrites, pseudo tags)
```

4 Experiments and Results

We focus on data released by the WMT20 shared task on QE for predicting post-editing effort, which includes English-to-German (En-De) and English-to-Chinese (En-Zh) word-level data and their sentence-level HTER (Specia et al., 2020).[1] As the human-annotated data is sampled from Wikipedia, we choose to synthesize data from WikiMatrix (Schwenk et al., 2019a), which consists of mined Wikipedia parallel data from which we sample pairs with a LASER (Artetxe and Schwenk, 2019) margin score threshold of 1.06 to ensure high-quality pairs. We note that the original QE data is not a subset of WikiMatrix. The German and Chinese text were tokenized using the Moses[2] and Jieba[3] tokenizers, respectively. We list the statistics of the filtered Wikimatrix data as well as our resulting synthetic data in Table 1.

For the off-the-shelf QE model, we choose the

---

[1] Available here: https://github.com/sheffieldnlp/mlqe-pe

[2] https://github.com/alvations/sacremoses

[3] https://github.com/fxsjy/jieba

Data	English-German			English-Chinese		
	MCC	F1-Ok	F1-Bad	MCC	F1-Ok	F1-Bad
Human annotation	**0.399**	0.879	**0.495**	0.525	0.820	0.659
MLM	0.332	**0.892**	0.438	0.500	0.850	0.643
NMT	0.379	0.826	0.468	0.525	**0.859**	0.660
NMT + MLM	**0.399**	0.866	0.493	**0.546**	0.835	**0.675**
Improvement (%)	+0.20	-1.40	-0.40	+4.00	+1.83	+2.43

Table 2: Results of word-level QE trained on human-annotated (7k) and synthetic data. Improvement in MCC for en-de & en-zh shows synthetic data can train word-level models comparable to human-annotated data. We report improvement comparing models trained with human-annotation vs our combined NMT+MLM synthetic data.

Data	English-German			English-Chinese		
	Pearson	MAE	RMSE	Pearson	MAE	RMSE
Human annotation	**0.394**	**0.150**	**0.187**	0.490	0.151	0.186
MLM	0.290	0.156	0.195	0.418	0.224	0.269
NMT	0.327	0.229	0.270	0.482	0.161	0.203
NMT + MLM	0.373	0.172	0.205	**0.506**	**0.148**	**0.183**
Improvement (%)	-5.50	+14.7	+9.63	+3.18	-1.79	-1.67

Table 3: Results of sentence-level HTER QE trained on human-annotated and synthetic data. For Pearson, positive improvement is better while for MAE & RMSE negative is better. We report improvement comparing models trained with human-annotation vs our combined NMT+MLM synthetic data.

multi-task predictor-estimator model (Kim et al., 2017) implemented by OpenKiwi v0.1.3 (Kepler et al., 2019b). This was the top-performing architecture for QE at WMT19 (Kepler et al., 2019a; Fonseca et al., 2019). We train the predictor on parallel MT data provided by the WMT20 QE shared task. The predictor reads in words' contextualized word representations, the estimator passes these features through a 2-layer 125-dimension bidirectional LSTM (biLSTM) and then feeds the outputs into 1-layer linear word-level classifier. The first output of the biLSTM is also fed into a multi-layer perceptron to predict a sentence-level score. For multi-task learning, we train the model with both word- and sentence-level data.

For a fair comparison, we take the pre-trained predictor provided by the WMT20 QE shared task, fine-tune the whole model on the human annotated data, and compare results to those when fine-tuned on our synthetic data. We test by comparing model predictions and held-out human-annotated QE at word and sentence-level. At the word level, we measure QE performance with Matthew's Correlation Coefficient (MCC) (Matthews, 1975) (main metric), as well as F1 scores for BAD and OK tags. At the sentence-level, we measure the sentence-level Pearson's correlation (Benesty et al., 2009), mean absolute error (MAE) and Root-mean-square deviation (RMSE).

As shown in Table 2, for word-level QE,[4] the model trained on synthetic data generated from NMT translations performs comparably to the same model trained on the original 7k human-annotated post-edits. This suggests that having human annotators post-edit each translation to create training data may be unnecessary and using reference sentences is good enough. The model trained on the MLM rewriting synthetic data generally under-performs compared to NMT generated data on MCC. However, we note that it performs better on F1 on OK tags. Therefore, we also ensemble the two models trained on each set of synthetic data through a linear combination. This yields comparable or better performance than the model trained on human-annotated data according to the main metric, MCC.

In Table 3, we compare the models trained on human-annotated data to our synthetic data for predicting sentence-level HTER scores. Again our synthetic data from NMT-generated translations outperforms MLM-rewriting data. Both underperform models trained on human-annotated data, but when combined they significantly improve and even outperform human-annotated for En-Zh. This once again suggests that the two forms of synthetic data are complementary and provide valuable signals for QE.

---

[4] The results reported in Tables 2 and 3 are evaluated on the test set provided (test20).

size	MCC	F1-BAD	F1-OK
English-German			
100k	37.72	46.23	83.99
200k	38.45	46.79	84.27
All (459k)	38.68	46.78	83.85
English-Chinese			
50k	53.07	66.58	83.65
100k	53.88	67.13	84.10
All (189k)	53.42	66.86	84.47

Table 4: Ablation study of synthetic data amounts.

5 Discussion

In this section, we further analyze how the quantity of synthetic data impacts performance, and what types of errors are represented in each of the MLM and NMT portions of the synthetic data.

5.1 Amount of Synthetic Data

As previously observed, the amount of synthetic data is orders of magnitude larger than the amount of human-annotated data. It begs the question: How much benefit do we get from smaller amounts of synthetic data? To analyze how the quantity of synthetic data affects QE performance, we conduct an ablation study of word-level QE.[5] As shown in Table 4, using only about half of the synthetic data generated (200k for En-De and 100k for En-Zh) is comparable to using the full generated set. While this suggests an upper-bound in performance to training on synthetic data. The ablation also suggests that this synthetic process can yield good performance with even a small amount of synthetic data.

5.2 Error Analysis

In addition to the performance, we posit that there are essential differences between MLM and NMT synthetic data. To test that, bilingual volunteers qualitatively analyzed the types of mistakes from MLM rewrites vs traditional NMT translations. The major reported differences in error types are:

1. Deletions from NMT translations appear more natural and do not destroy the sentence fluency. However, deletions in MLM rewrites are more destructive (e.g., “new york restaurants” vs “new restaurants” The semantics is changed).

2. Most incorrect insertions or deletions from NMT translations are due to re-ordering words. (e.g., “on 2020 in california” vs “in california on 2020”) However insertions with MLM-rewrites introduces seemingly random words.

3. NMT translations often have semantically distant word substitutions. However, MLM-rewrites tend to substitute similar words (e.g., “strong tea” vs “powerful tea”).

In summary, NMT translations and MLM-rewrites appear to generate different types of errors – the former leads to more subtle errors while the latter often introduces more catastrophic errors. Since a high-quality QE model should be able to detect both types of errors, ensembling the models trained on these two forms of synthetic data indeed is expected to outperform using only one form of synthetic data.

6 Conclusions and Future Work

In this work we devise a technique for building word and sentence-level QE models by creating synthetic training data. By training an off-the-shelf model on our synthetic data, we achieve performance comparable to and often better than training on human-annotated data. This technique for data synthesis can be invaluable if human annotation is difficult to come-by, for example when dealing with low-resource scenarios.

This work can be extended in various ways. While we investigate the scenario of utilizing solely synthetic data, further work can study the effects of augmenting human-labeled data with synthetic data. Further work can analyze the efficacy of this technique into low-resource language pairs where such human-annotation is difficult to obtain. Additionally, instead of a simple MLM re-writer, adversarial training to generate and detect errors could provide more realistic synthetic data.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer.

---

[5] The ablation study is only trained on word-level data.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *Coling 2004: Proceedings of the 20th international conference on computational linguistics*, pages 315–321.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020a. A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969.

Ahmed El-Kishky and Francisco Guzmán. 2020. Massively multilingual document alignment with cross-lingual sentence-mover's distance. *arXiv preprint arXiv:2002.00761*.

Ahmed El-Kishky, Philipp Koehn, and Holger Schwenk. 2020b. Searching the web for cross-lingual parallel data. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2417–2420.

Thierry Etchegoyhen, Eva Martínez Garcia, and Andoni Azpeitia. 2018. Supervised and unsupervised minimalist quality estimators: Vicomtech's participation in the wmt 2018 quality estimation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 782–787.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *TACL*.

Erick Fonseca, Lisa Yankovskaya, André FT Martins, Mark Fishel, and Christian Federmann. 2019. Findings of the wmt 2019 shared tasks on quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, António Góis, M Amin Farajian, António V Lopes, and André FT Martins. 2019a. Unbabel's participation in the wmt19 translation quality estimation shared task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 78–84.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André FT Martins. 2019b. Openkiwi: An open source framework for quality estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 562–568.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Brian W Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

Erwan Moreau and Carl Vogel. 2012. Quality estimation: an experimental study using unsupervised similarity measures. In *7th Workshop on Statistical Machine Translation*, page 120.

Matteo Negri, Marco Turchi, Rajen Chatterjee, and Nicola Bertoldi. 2018. Escape: a large-scale synthetic corpus for automatic post-editing. *arXiv preprint arXiv:1803.07274*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Maja Popović. 2012. Morpheme-and pos-based ibm1 and language model scores for translation quality estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 133–137.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019a. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *arXiv preprint arXiv:1907.05791*.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, and Armand Joulin. 2019b. Ccmatrix: Mining billions of high-quality parallel sentences on the web. *arXiv preprint arXiv:1911.04944*.

Matthew Snover and Michael R Brent. 2001. A bayesian model for morpheme and paradigm identification. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, pages 490–498.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764. Association for Computational Linguistics.

Lucia Specia, Kashif Shah, José GC De Souza, and Trevor Cohn. 2013. Quest-a translation quality estimation framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84.

Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation systems. In *13th Conference of the European Association for Machine Translation*, pages 28–37.

Shuo Sun, Marina Fomicheva, Frédéric Blain, Vishrav Chaudhary, Ahmed El-Kishky, Adithya Renduchintala, Francisco Guzmán, and Lucia Specia. 2020. An exploratory study on multilingual quality estimation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 366–377. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wanrong Zhu, Zhiting Hu, and Eric Xing. 2019. Text infilling. *arXiv preprint arXiv:1901.00158*.

How Fast can BERT Learn Simple Natural Language Inference?

Yi-Chung Lin and **Keh-Yih Su**
Institute of Information Science
Academia Sinica, Taiwan
`{lyc,kysu}@iis.sinica.edu.tw`

Abstract

This paper empirically studies whether BERT can really learn to conduct natural language inference (NLI) without utilizing hidden dataset bias; and how efficiently it can learn if it could. This is done via creating a simple entailment judgment case which involves only binary predicates in plain English. The results show that the learning process of BERT is very slow. However, the efficiency of learning can be greatly improved (data reduction by a factor of 1,500) if task-related features are added. This suggests that domain knowledge greatly helps when conducting NLI with neural networks.

1 Introduction

Entailment judgment (Dagan et al., 2006; Marelli et al., 2014a) is a common test for *natural language inference* (NLI) (Camburu et al., 2018; Conneau et al., 2018) as it possesses the simplest form in related tasks such as question and answering (Bowman and Zhu, 2019). Also, SNLI dataset (Bowman et al., 2015) is frequently adopted for NLI evaluation because it is the first corpus to show the power of neural networks for the task that specifically targets NLI.

Recently, BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), and RoBERTa (Liu et al., 2019) have all shown excellent performances on SNLI, surpassing even human performance (Gong et al., 2017). However, Tsuchiya (2018) and Gururangan et al. (2018) show that SNLI contains *hidden bias*. Also, deep neural networks (DNNs) have been shown to predominantly capture the *statistical irregularities* that go unnoticed by humans (Poliak et al., 2018); this surprisingly yields 69% accuracy on SNLI without being provided the associated *premise* (i.e., the required *supporting evidence*). Furthermore, some studies (Naik et al., 2018; McCoy et al., 2019; Jiang and Marneffe, 2019) have shown that BERT mainly conducts NLI with surface clues/patterns, but not those clues actually adopted by humans.

To the best of our knowledge, none of previous studies carefully removes the dataset bias, and then investigates how efficiently that BERT could learn NLI if the surface clues are *completely* removed. In this paper, we empirically study whether BERT is capable of learning NLI without surface clues/bias appearing in the dataset; and if it is, whether it can learn NLI efficiently. We carefully created absolutely unbiased datasets, in which the premises and hypotheses simply describe the relative position of two objects in plain English. That is, given a premise "*John is on the left side of Mary*" (abbreviated as a predicate "left(John, Mary)" from now on), the hypotheses predicates "left(John, Mary)", "left(Mary, John)" and "left(John, Helen)" should be labeled as *entailment*, *contradictory* and *neutral* respectively. Our experiment results show BERT is very slow in learning this simple NLI.

We then further study if the learning efficiency can be improved with domain knowledge (Gülçehre and Bengio, 2016). Inspired by Chen et al. (2017), we think whether two entities are *exactly the same* is quite crucial in making the above NLI. So, the task-related features, such as whether the first/second argument of the premise predicate exactly matches that of hypothesis, are fed into BERT. The obtained results show that such task-related features are able to greatly benefit BERT (reducing the data needed by a factor of 1,500), which is important as it is difficult to acquire enough data in many real-world applications.

Our main contributions are: (1) We are the first to quantitatively study how efficiently BERT can learn to conduct NLI without available surface clues/bias. (2) We design experiments to completely eliminate hidden bias while evaluating

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 626–633
April 19 - 23, 2021.

the inference capability of BERT. (3) We show that adding task-related features greatly enhances the learning efficiency of BERT.

2 Teaching BERT Binary Predicates

The sentence "*John is on the left side of Mary*" describes a positional relation between two people. For conciseness, it will be denoted by a binary predicate "left(John,Mary)" from now on, where "left" is the predicate name, "John" is the first argument and "Mary" is the second argument. We seek to determine how much data are required to teach BERT to truly understand this simple binary predicate (i.e., premise) and correctly judge that the hypothesis "left(Mary,John)" is contradictory and that the hypothesis "left(John,Helen)" is neutral. Besides, we also seek to determine whether BERT is also able to learn the antonymous predicate "right(·,·)", and judge that the hypothesis "right(Mary,John)" is entailed by the above premise.

2.1 Entity Names and Datasets

The arguments of the above binary predicates "left(·,·)" and "right(·,·)" actually can be the names of any objects. However, in this paper, we simply trained BERT with *personal names*. To avoid dividing a personal name into sub-words, we collected 1,696 male and female first names which appear in the vocabulary of the pre-trained "BERT-Base, Uncased" model (Devlin et al., 2019). These names were randomly partitioned into three sets: $\boldsymbol{\eta}_\mathbf{T}$, $\boldsymbol{\eta}_\mathbf{V}$ and $\boldsymbol{\eta}_\mathbf{E}$. The subscripts **T**, **V** and **E** indicate these name sets will be used for **T**raining, **V**alidation and **E**valuation respectively. Sets $\boldsymbol{\eta}_\mathbf{T}$ and $\boldsymbol{\eta}_\mathbf{V}$, which consist of 1,356 and 170 names respectively, are used to generate the training and validation datasets for fine-tuning BERT; and Set $\boldsymbol{\eta}_\mathbf{E}$, consisting of 170 names, is used to generate a dataset for evaluating the performance of BERT in understanding the predicates with personal names.

Furthermore, we also seek to ascertain whether the BERT model trained by the predicates with personal names also understands the predicates with names of other object types. Therefore, in addition to personal names, we also collected 30 common fruit and vegetable names to create an additional set $\mathbf{f}_\mathbf{E}$, which will be used to generate another dataset for performance evaluation.

We conduct a number of experiments on *recognizing textual entailment* (RTE) (Dagan et al., 2006; Marelli et al., 2014a) to study the learning curves of BERT in understanding binary predicates. In each experiment, four datasets—**Name-T**, **Name-V**, **Name-E** and **Fruit-E**—are created by filling experiment-specific templates with names randomly chosen from $\boldsymbol{\eta}_\mathbf{T}$, $\boldsymbol{\eta}_\mathbf{V}$, $\boldsymbol{\eta}_\mathbf{E}$ and $\mathbf{f}_\mathbf{E}$ respectively. The training set **Name-T** and the validation set **Name-V** are used to fine-tune the BERT model. The other two sets (**Name-E** and **Fruit-E**) are test sets. They are used to assess the performances of the BERT model. Each dataset is generated via iteratively and sequentially adding one entailment example, one contradictory example, and one neutral example until it reaches the desired size. Thus a dataset of a size 100 will consist of 34 entailment examples, 33 contradictory examples, and 33 neutral examples. The experiment-specific templates are described in the following sections.

2.2 One Simple Binary Predicate

We first conducted *EXP-SP* (SP: **S**imple **P**redicate) experiment to show if BERT can be taught to understand the simple binary predicate "left(·,·)". In this experiment, a template has the form "*premise* [s] *hypothesis*", where "[s]" is a separator token between the premise and the hypothesis. The templates are partitioned into entailment, contradictory, and neutral template sets. The entailment template set has only one template "left(x,y) [s] left(x,y)", where "left(x,y)" represents the token sequence "x is on the left side of y" and the variables x and y indicate the names to be filled in. Likewise, the contradictory template set has only one template "left(x,y) [s] left(y,x)". However, the neutral template set consists of four templates:

"left(x,y) [s] left(x,z)",
"left(x,y) [s] left(z,x)",
"left(x,y) [s] left(y,z)",
"left(x,y) [s] left(z,y)",

where the variable z indicates a name to be filled in. Names and templates were randomly selected during dataset creation. For example, when generating a neutral example for **Name-T**, we randomly chose one template from the neutral template set and randomly chose three distinct names (for variables x,y,z) from name set $\boldsymbol{\eta}_\mathbf{T}$. Obviously, the generated datasets do not have any hidden bias, as all examples share the same token sequence except the argument tokens which are randomly chosen. Therefore, no annotation

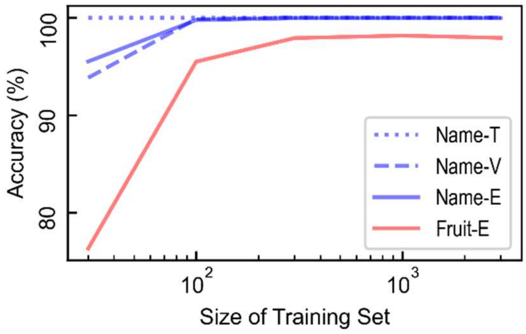


Figure 1: RTE performances of EXP-SP.

artifacts (Gururangan et al., 2018) in the data can provide hints for predicting final inference answers.

To quantitatively study the learning efficiency, we generated various **Name-T** sets with different sizes to study how much data would be required to teach BERT to understand the simple binary predicate "left(·,·)". We set **Name-V**, **Name-E**, and **Fruit-E** to 1,000 examples each. The solid lines in Figure 1 show that the accuracies of BERT on both test sets (i.e., **Name-E** and **Fruit-E**) increase when the training set size increases. However, even if we train BERT with 3,000 examples, it still cannot achieve 100% accuracy on the test sets. In fact, all sentences in this experiment have the form "*x is on the left side of y*", where x and y are object names. Therefore, only 6 different words could appear in the context of object names. In other words, given 3,000 examples, BERT still cannot fully understand the meaning of the context "*is on the left side of*".

Furthermore, training BERT to reach over 99% accuracy on **Name-E** requires 100 training examples. However, even given 30 times the training data, BERT is still unable to achieve 99% accuracy on **Fruit-E**. That is, BERT is not able to well generalize what it has learned from the examples with person names to the examples with fruit and vegetable names.

2.3 One Antonymous Predicate

Antonyms are frequently used in natural language and play an important role in natural language inference. Given the premise "*John is on the left side of Mary*", we can easily infer that the hypothesis "*Mary is on the right side of John*" is entailed by the premise, and that the hypothesis "*John is on the right side of Mary*" contradicts to the premise. This inference is easy for humans; but is it also easy for BERT? Therefore, we conducted another test named *EXP-AP* (AP: **A**ntonymous **P**redicate), a more complicated RTE experiment in which we added the antonymous predicate "right(·,·)".

In this experiment, the entailment template set consists of the following four templates:

$$\text{"left}(x,y)\text{ [s] left}(x,y)\text{"},$$
$$\text{"left}(x,y)\text{ [s] right}(y,x)\text{"},$$
$$\text{"right}(x,y)\text{ [s] right}(x,y)\text{"},$$
$$\text{"right}(x,y)\text{ [s] left}(y,x)\text{"}.$$

The contradictory template set consists of the following four templates:

$$\text{"left}(x,y)\text{ [s] left}(y,x)\text{"},$$
$$\text{"left}(x,y)\text{ [s] right}(x,y)\text{"},$$
$$\text{"right}(x,y)\text{ [s] right}(y,x)\text{"},$$
$$\text{"right}(x,y)\text{ [s] left}(x,y)\text{"}.$$

Likewise, the neutral template set consists of 16 templates. In brief, adding one antonymous predicate "right(·,·)" enlarges the number of all possible templates for dataset generation from 6 to 24.

The solid lines in Figure 2 are the accuracies on the test sets **Name-E** and **Fruit-E** in EXP-AP. For ease of comparison, we also plot the EXP-SP counterpart accuracies with dotted lines. Although EXP-AP uses only four times as many templates as EXP-SP uses, we must provide more than 30 times the training data (from 100 to 3,000) for BERT to reach 99% accuracy on **Name-E**. That is, by adding a single antonymous predicate, the RTE task of EXP-AP becomes 30 times harder than that of EXP-SP. It seems teaching BERT to "almost understand" two simple binary predicates requires more than 3,000 examples. This result could be also interpreted in another view. In the EXP-AP, the context of object names in each sentence is either "*is on the left side of*" or "*is on the right side of*". BERT requires 3,000 examples to learn the meanings of the 7 different words that appear in these two contexts. This represents a quite inefficient learning curve.

3 Incorporating Human Knowledge

The previous section showed that many training examples are needed to teach BERT for understanding two binary predicates "left(·,·)" and "right(·,·)" in a simple RTE task. It thus naturally leads to a conjecture that training BERT to understand more complicated predicates would very likely require infeasible amount of training

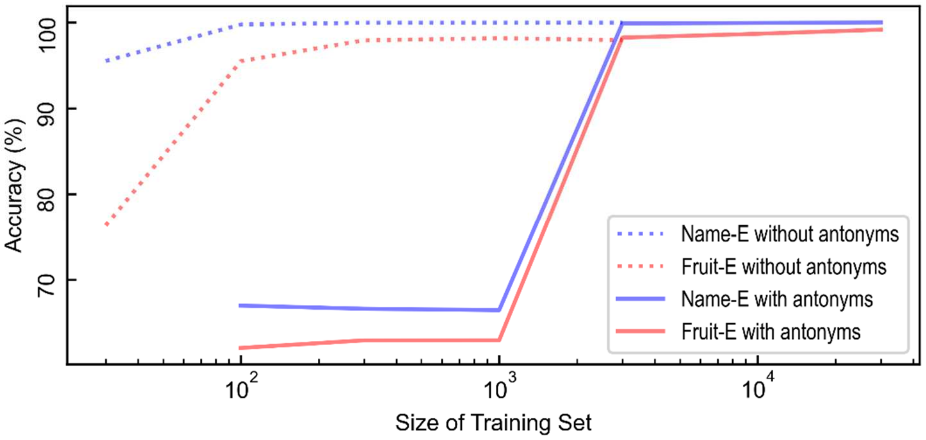


Figure 2: RTE performances of EXP-SP (dotted lines) and EXP-AP (solid lines).

data. One possible solution is to improve the learning curve of BERT by feeding it useful/obvious features that have been identified in human/domain knowledge.

3.1 Simple Features (SF)

Accordingly, we conducted an experiment *EXP-SF*, in which we appended features to the input tokens of EXP-AP to improve the learning curve. In this experiment, a template has the form “*premise* [s] *hypothesis* [s] *features*”. Obviously, humans directly compare the predicate name and the predicate arguments in the premise against their counterparts in the hypothesis. Perhaps this knowledge about which two fields should be compared would be helpful when training BERT on the EXP-AP task. Let N_{P} and N_{H} indicate the predicate names in the premise and the hypothesis respectively; also, let $A_{i,\mathrm{P}}$ and $A_{i,\mathrm{H}}$ indicate the i-th predicate arguments in the premise and the hypothesis respectively. In EXP-SF, every example in the datasets of EXP-AP will be augmented by the following three indicator features:

$$f_1 = I(N_{\mathrm{P}} = N_{\mathrm{H}}),$$
$$f_2 = I\big(A_{1,\mathrm{P}} = A_{1,\mathrm{H}}\big),$$
$$f_3 = I\big(A_{2,\mathrm{P}} = A_{2,\mathrm{H}}\big).$$

Here the indicator function $I(x)$ returns the word “*true*” if x is true and “*false*” if it is not. EXP-SF templates thus could be directly transformed from the corresponding EXP-AP templates. To illustrate this transformation, we show an entailment template, a contradictory template and a neutral template in EXP-SF as follows:

“left(x, y) [s] right(y, x) [s] false false false”,
“left(x, y) [s] left(y, x) [s] true false false”,
“left(x, y) [s] right(x, z) [s] false true false”.

The dashed lines in Figure 3 are the accuracies on the test sets **Name-E** and **Fruit-E** in EXP-SF. For ease of comparison, we also plot the counterpart accuracies of the baseline (i.e., EXP-AP) with dotted lines. The fact that the dashed lines lie far on the left side of the dotted lines indicates that much fewer training examples are required to train BERT after adding these three simple features [1]. This represents a greatly improved learning curve. Specifically, given merely 100 EXP-SF examples, BERT achieves over 99% accuracy on **Name-E**; in contrast, BERT requires 3,000 examples to surpass 99% accuracy on **Name-E** in EXP-AP. This represents a greatly improved learning curve.

3.2 Discriminant Features (DF)

In the previous experiment EXP-SF, features f_2 and f_3 do not precisely indicate the situation in which the arguments in the premise match the arguments in the hypothesis after swapping. For example, for both “left(x, y) [s] left(y, x)” and “left(x, y) [s] left(z, x)”, features f_2 and f_3 are all *false* in EXP-SF. However, the former is a contradictory case and the latter is a neutral case. We thus conducted the last experiment named *EXP-DF*, in which we replaced f_2 and f_3 with a

---

[1] Note that it does not matter which words were chosen to represent the values of features. We had randomly selected *fortification* and *mississippi* from BERT’s vocabulary to replace the words *true* and *false*. The experimental results were similar to those in Figure 3. In other words, the initial embeddings of the feature values are not crucial.

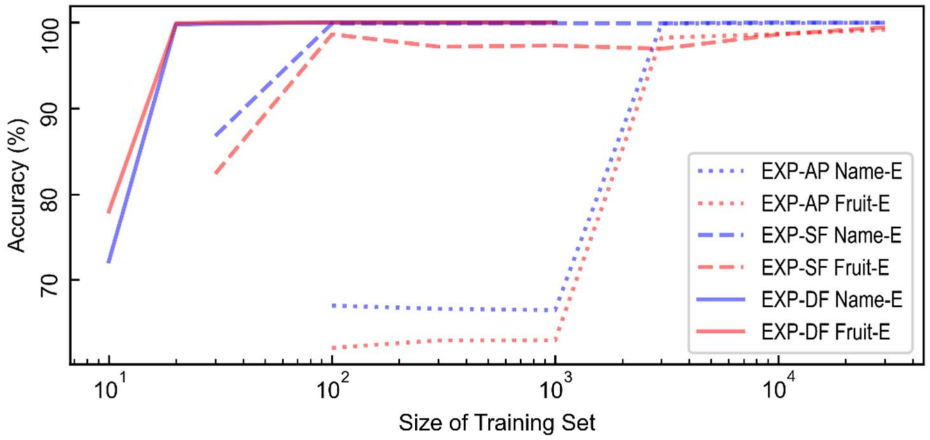


Figure 3: RTE performances of EXP-AP (dotted lines), EXP-SF (dashed lines) and EXP-DF (solid lines). More detailed data values are listed in Appendix.

new discriminant feature f_2'. Words *true*, *false* and *fuzzy* were used to indicate the three possible values of this new feature as follows:

$$f_2' = \begin{cases} true, & A_{1,\mathrm{P}} = A_{1,\mathrm{H}} \land A_{2,\mathrm{P}} = A_{2,\mathrm{H}}; \\ false, & A_{1,\mathrm{P}} = A_{2,\mathrm{H}} \land A_{2,\mathrm{P}} = A_{1,\mathrm{H}}; \\ fuzzy, & \text{otherwise.} \end{cases}$$

The solid lines in Figure 3 show that the learning curve of BERT is further improved after adopting this discriminant feature. Given only 20 EXP-DF examples, BERT achieves 99.9% accuracy on **Fruit-E**. However, in EXP-AP, 1,500 times amount of data (i.e., 30,000 examples) is needed for reaching 99.1% accuracy on **Fruit-E**.

4 Related Work

Dagan et al. (2006) first initiated the task of recognizing textual entailment about fifteen years ago. This task continued until 2011 (Bentivogli et al., 2011). Afterwards, conducting inference with BERT was studied in (Clark et al., 2019; Zellers et al., 2019; Tenney et al., 2019; Aken et al., 2019; Coenen et al., 2019; Michel et al., 2019).

On the other hand, various corpora have been created for conducting NLI for different purposes: SICK (Marelli et al., 2014b), SNLI (Bowman et al., 2015), MNLI (Willams et al., 2018), MPE (Lai et al., 2017), JOCI (Zhang et al., 2017), XNLI (Conneau et al., 2018), and SciTail (Khot et al., 2018). Corpora such as HOTPOTQA (Yang et al., 2018), Breaking-NLI (Glockner et al., 2018), CommonSense QA (Talmor et al., 2019), DROP (Dua et al., 2019) and ROPES (Lin et al., 2019) have also been created recently to evaluate more diverse and difficult NLI cases. However, all those corpora are not created for keeping the data absolutely unbiased. We believe using bias-free simple binary predicates would be more suitable to assess the true inference capability of BERT (or even other DNNs).

Recently, Ribeiro et al. (2020) proposed a new evaluation methodology, named CheckList, to check general linguistic capabilities of a given NLI model. They generated a large number of diverse test cases to identify the critical failures hidden behind state-of-art models. In contrast, our work mainly targeted the learning efficiency of BERT on various unbiased datasets. Besides, we also studied how adding useful domain-specific features would affect the learning curve of BERT.

5 Conclusion

This paper is the first quantitative study on whether BERT could really learn to conduct NLI without implicitly utilizing hidden dataset bias, and how quickly it does so if it could. We conduct experiments to evaluate the capability of BERT on making inference without hidden bias, and show that BERT learns NLI inefficiently even for a simple case. We further add task-related features to greatly enhance BERT's learning efficiency. As a result, it suggests that domain knowledge may be essential in conducting NLI with neural networks (at least for BERT).

References

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions: A Layer-Wise Analysis of Transformer

Representations. *Computing Research Repository,* arXiv:1909.04925.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2011. The Seventh PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the 4th Text Analysis Conference*. Gaithersburg: National Institute of Standards and Technology, pages 1–16.

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

Samuel R. Bowman and Xiandan Zhu. 2019 Deep Learning for Natural Language Inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-SNLI: Natural Language Inference with Natural Language Explanations. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1870–1879.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look At? An Analysis of BERT's Attention. *Computing Research Repository,* arXiv:1906.04341.

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda Viégas, and Martin Wattenberg. 2019. Visualizing and Measuring the Geometry of BERT. *Computing Research Repository,* arXiv:1906.02715.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating Cross-lingual Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2475–2485.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, Springer, pages 177–190.

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

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 2368–2378.

Max Glockner, Vered Shwartz and Yoav Goldberg. 2018. Breaking NLI Systems with Sentences that Require Simple Lexical Inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 650–655.

Yichen Gong, Heng Luo and Jian Zhang. 2017. Natural Language Inference Over Interaction Space. *Computing Research Repository,* arXiv:1709.04348.

Çağlar Gülçehre and Yoshua Bengio. 2016. Knowledge Matters: Importance of Prior Information for Optimization. *Journal of Machine Learning Research*, 17(8): 1-32.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 107–112.

Nanjiang Jiang and Marie-Catherine de Marneffe, 2019. Evaluating BERT for natural language inference: A case study on the Commitment Bank. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, pages 6086–6091.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SCITAIL: A Textual Entailment Dataset from Science Question Answering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI-18)*. Association for the Advancement of Artificial Intelligence, pages 5189–5197.

Alice Lai, Yonatan Bisk, and Julia Hockenmaier. 2017. Natural Language Inference from Multiple Premises. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, pages 100–109.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning Over Paragraph Effects in Situations. *Computing Research Repository,* arXiv:1908.05852.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *Computing Research Repository,* arXiv:1907.11692.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini and Roberto Zamparelli. 2014a. SemEval-2014 Task 1: Evaluation of Compositional Distributional Semantic Models on Full Sentences through Semantic Relatedness and Textual Entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. Association for Computational Linguistics, pages 1–8.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi and Roberto Zamparelli. 2014b. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*. European Language Resources Association (ELRA), pages 216–223.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 3428–3448.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are Sixteen Heads Really Better than One? In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. arXiv:1905.10650.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress Test Evaluation for Natural Language Inference. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Compu-tational Linguistics, pages 2340–2353.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, pages 180–191.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin and Sameer Singh. 2020. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 4902–4912.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CoomonSenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 4149–4158.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 4593–4601.

Masatoshi Tsuchiya. 2018. Performance Impact Caused by Hidden Bias of Training Data for Recognizing Textual Entailment. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*. https://www.aclweb.org/anthology/L18-1239

Adina Willams, Nikita Nangia, and Samuel R Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 1112–1122.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HOTPOTQA: A Dataset for Diverse Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2369–2380.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Computing Research Repository,* arXiv:1906.08237.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pages 4791–4800.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal Commonsense Inference. Transactions of the Association for Computational Linguistics, 5:379–395.

Appendix

To reduce the performance assessment error, each accuracy reported in this paper is the mean of accuracies obtained from multiple simulations. Each simulation uses a unique random seed to fine-tune the BERT model. The accuracies plotted in the figures for EXP-AP, EXP-SF and EXP-DF are listed in the following table, where μ denotes the accuracy mean and σ denotes the standard deviation of μ.

Training Set Size	Test Set Accuracy (%)			
	Name-E		**Fruit-E**	
	μ	σ	μ	σ
EXP-AP				
100	67.0	0.6	62.1	1.8
300	66.6	0.3	62.9	0.9
1000	66.5	0.4	62.9	1.4
3000	99.9	0.0	98.2	0.3
10000	99.9	0.0	98.7	0.2
30000	100.0	0.0	99.1	0.1
EXP-SF				
30	86.8	0.9	82.4	3.1
100	99.9	0.1	98.6	0.4
300	99.9	0.0	97.2	0.4
1000	99.9	0.0	97.3	0.6
3000	99.9	0.0	96.9	0.5
10000	100.0	0.0	98.6	0.9
30000	100.0	0.0	99.4	0.1
EXP-DF				
10	72.1	1.3	78.0	3.4
20	99.8	0.2	99.9	0.1
30	99.9	0.1	100.0	0.0
100	100.0	0.0	100.0	0.0
300	100.0	0.0	100.0	0.0
1000	100.0	0.0	100.0	0.0

GRIT: Generative Role-filler Transformers for Document-level Event Entity Extraction

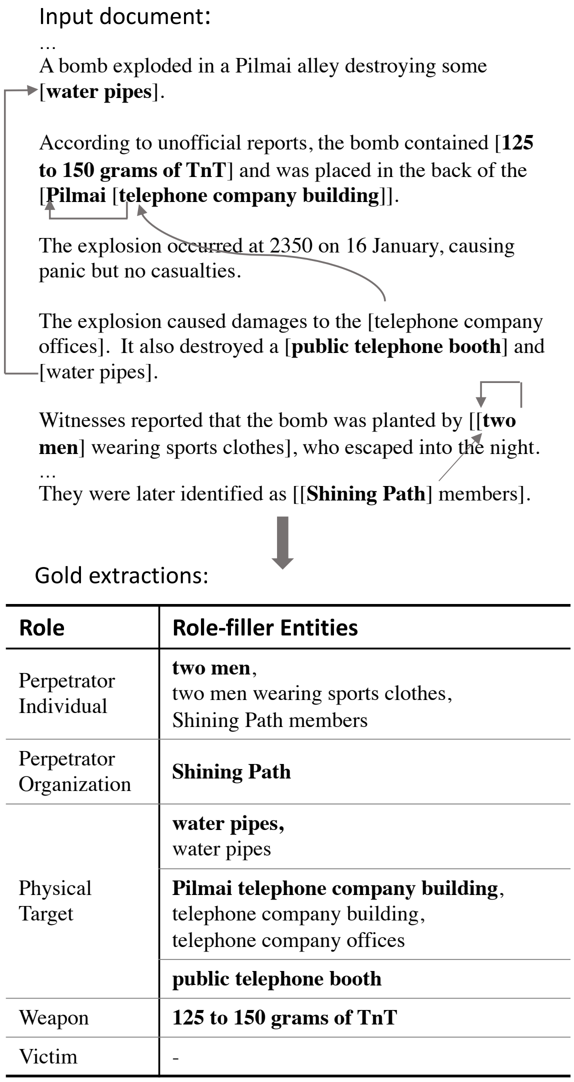
Xinya Du **Alexander M. Rush** **Claire Cardie**
Department of Computer Science
Cornell University
{xdu, cardie}@cs.cornell.edu
arush@cornell.edu

Abstract

We revisit the classic problem of document-level role-filler entity extraction (REE) for template filling. We argue that sentence-level approaches are ill-suited to the task and introduce a generative transformer-based encoder-decoder framework (GRIT) that is designed to model context at the document level: it can make extraction decisions across sentence boundaries; is *implicitly* aware of noun phrase coreference structure, and has the capacity to respect cross-role dependencies in the template structure. We evaluate our approach on the MUC-4 dataset, and show that our model performs substantially better than prior work. We also show that our modeling choices contribute to model performance, e.g., by implicitly capturing linguistic knowledge such as recognizing coreferent entity mentions.

1 Introduction

Document-level template filling (Sundheim, 1991, 1993; Grishman and Sundheim, 1996) is a classic problem in information extraction (IE) and NLP (Jurafsky and Martin, 2014). It is of great importance for automating many real-world tasks, such as event extraction from newswire (Sundheim, 1991). The complete task is generally tackled in two steps. The first step detects events in the article and assigns templates to each of them (template recognition); the second step performs role-filler entity extraction (REE) for filling in the templates. In this work we focus on the **role-filler entity extraction (REE)** sub-task of template filling (Figure 1).[1] The input text describes a bombing event; the goal is to identify the entities that fill any of the roles associated with the event (e.g., the perpetrator, their organization, the weapon) by extracting a descriptive "mention" of it – a string from the document.



Input document:

...
A bomb exploded in a Pilmai alley destroying some [**water pipes**].

According to unofficial reports, the bomb contained [**125 to 150 grams of TnT**] and was placed in the back of the [**Pilmai** [**telephone company building**]].

The explosion occurred at 2350 on 16 January, causing panic but no casualties.

The explosion caused damages to the [telephone company offices]. It also destroyed a [**public telephone booth**] and [water pipes].

Witnesses reported that the bomb was planted by [[**two men**] wearing sports clothes], who escaped into the night.
...
They were later identified as [[**Shining Path**] members].

Gold extractions:

Role	Role-filler Entities
Perpetrator Individual	**two men**, two men wearing sports clothes, Shining Path members
Perpetrator Organization	**Shining Path**
Physical Target	**water pipes,** water pipes
	Pilmai telephone company building, telephone company building, telephone company offices
	public telephone booth
Weapon	**125 to 150 grams of TnT**
Victim	-

Figure 1: Role-filler entity extraction (REE). The first mention of each role-filler entity is bold in the table and document. The arrows denote coreferent mentions.

In contrast to sentence-level event extraction (see, e.g., the ACE evaluation (Linguistic Data Consortium, 2005)), document-level REE introduces

---

[1] In this work, we assume there is one generic template for the entire document (Huang and Riloff, 2011, 2012).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 634–644
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

several complications. First, **role-filler entities must be extracted even if they never appear in the same sentence as an event trigger**. In Figure 1, for example, the WEAPON and the first mention of the telephone company building (TARGET) appear in a sentence that does not explicitly mention the explosion of the bomb. In addition, **REE is ultimately an entity-based task** — exactly one descriptive mention for each role-filler should be extracted even when the entity is referenced multiple times in connection with the event. The final output for the bombing example should, therefore, include just one of the "water pipes" references, and one of the three alternative descriptions of the PERPIND and the second TARGET, the telephone company building. As a result of these complications, end-to-end sentence-level event extraction models (Chen et al., 2015; Lample et al., 2016), which dominate the literature, are ill-suited for the REE task, which calls for models that encode information and track entities across a longer context.

Fortunately, neural models for event extraction that have the ability to model longer contexts have been developed. Du and Cardie (2020), for example, extend standard contextualized representations (Devlin et al., 2019) to produce a document-level sequence tagging model for event argument extraction. Both approaches show improvements in performance over sentence-level models on event extraction. Regrettably, these approaches (as well as most sentence-level methods) handle each candidate role-filler prediction in isolation. Consequently, they cannot easily **model the coreference structure required to limit spurious role-filler mention extractions**. Nor can they easily **exploit semantic dependencies between closely related roles** like the PERPIND and the PERPORG, which can share a portion of the same entity span. "Shining Path members", for instance, describes the PERPIND in Figure 1, and its sub-phrase, "Shining Path", describes the associated PERPORG.

Contributions In this work we revisit the classic but recently under-studied problem of document-level role-filler entity extraction problem and introduce a novel end-to-end generative transformer model — the "**G**enerative **R**ole-f**i**ller **T**ransformer" (**GRIT**) (Figure 2).

- Designed to model context at the document level, GRIT (1) has the ability to make extraction decisions across sentence boundaries; (2) is implicitly aware of noun phrase coreference structure; and (3) has the capacity to respect cross-role dependencies. More specifically, GRIT is built upon the pre-trained transformer model (BERT): we add a pointer selection module in the decoder to permit access to the entire input document, and a generative head to model document-level extraction decisions. In spite of the added extraction capability, GRIT requires no additional parameters beyond those in the pre-trained BERT.

- To measure the model's ability to both extract entities for each role, and implicitly recognize coreferent relations between entity mentions, we design a metric (CEAF-REE) based on a maximum bipartite matching algorithm, drawing insights from the CEAF (Luo, 2005) coreference resolution measure.

- We evaluate GRIT on the MUC-4 (1992) REE task (Section 3). Empirically, our model outperforms substantially strong baseline models. We also demonstrate that GRIT is better than existing document-level event extraction approaches at capturing linguistic properties critical for the task, including coreference between entity mentions and cross-role extraction dependencies.[2]

2 Related Work

Sentence-level Event Extraction Most work in event extraction has focused on the ACE sentence-level event task (Walker et al., 2006), which requires the detection of an event trigger and extraction of its arguments from within a single sentence. Previous state-of-the-art methods include Li et al. (2013) and Li et al. (2015), which explored a variety of hand-designed features. More recently, neural network based models such as recurrent neural networks (Nguyen et al., 2016; Feng et al., 2018), convolutional neural networks (Nguyen and Grishman, 2015; Chen et al., 2015) and attention mechanisms (Liu et al., 2017, 2018) have also been shown to help improve performance. Beyond the task-specific features learned by the deep neural models, Zhang et al. (2019) and Wadden et al. (2019) also utilize pre-trained contextualized representations.

Only a few models have gone beyond individual sentences to make decisions. Ji and Grishman (2008) and Liao and Grishman (2010) utilize event type co-occurrence patterns to propagate

---

[2] Our code for the evaluation script and models is at `https://github.com/xinyadu/grit_doc_event_entity` for reproduction purposes.

event classification decisions. Yang and Mitchell (2016) propose to learn within-event (sentence) structures for jointly extracting events and entities within a document context. Similarly, from a methodological perspective, our GRIT model also learns structured information, but it learns the dependencies between role-filler entity mentions and between different roles. Duan et al. (2017) and Zhao et al. (2018) leverage document embeddings as additional features to aid event detection. Although the approaches above make decisions with cross-sentence information, their extractions are still done the sentence level.

Document-level IE Document-level event role-filler *mention* extraction has been explored in recent work, using hand-designed features for both local and additional context (Patwardhan and Riloff, 2009; Huang and Riloff, 2011, 2012), and with end-to-end sequence tagging based models with contextualized pre-trained representations (Du and Cardie, 2020). These efforts are the most related to our work. The key difference is that our work focuses on a more challenging, and more realistic, setting: extracting **role-filler entities** rather than lists of role-filler mentions that are not grouped according to their associated entity. Also on a related note, Chambers and Jurafsky (2011), Chambers (2013), and Liu et al. (2019) work on *unsupervised* event schema induction and open-domain event extraction from documents. The main idea is to group entities corresponding to the same role into an event template.

Recently, there has also been increasing interest in cross-sentence/document-level relation extraction (RE). In the scientific domain, Peng et al. (2017); Wang and Poon (2018); Jia et al. (2019) study N-ary cross-sentence RE using distant supervision annotations. Luan et al. (2018) introduce SciERC dataset and their model rely on multi-task learning to share representations between entity span extraction and relations. Yao et al. (2019) construct an RE dataset of cross-sentence relations on Wikipedia paragraphs. Ebner et al. (2020) introduce RAMS dataset for multi-sentence argument *mention* linking, while we focus on entity-level extraction in our work. Different from work on joint modeling (Miwa and Bansal, 2016) and multi-task learning (Luan et al., 2019) setting for extracting entities and relations, through the generative modeling setup, our GRIT model *implicitly* captures (non-)coreference relations between noun phrases, without relying on the cross-sentence coreference and relation annotations during training.

Neural Generative Models with a Shared Module for Encoder and Decoder Our GRIT model uses one shared transformer module for both the encoder and decoder, which is simple and effective. For the machine translation task, He et al. (2018) propose a model which shares the parameters of each layer between the encoder and decoder to regularize and coordinate the learning. Dong et al. (2019) presents a new unified pre-trained language model that can be fine-tuned for both NLU and NLG tasks. Similar to our work, they also introduce different masking strategies for different kinds of tasks (see Section5).

3 The Role-filler Entity Extraction Task and Evaluation Metric

We base the REE task on the original MUC[3] formulation (Sundheim, 1991), but simplify it as done in prior research (Huang and Riloff, 2012; Du and Cardie, 2020). In particular, we assume that one generic template should be produced for each document: for documents that recount more than one event, the extracted role-filler entities for each are merged into a single event template. Second, we focus on entity-based roles with string-based fillers[4].

- Each *event* consists of the set of roles that describe it (shown in Figure 1). The MUC-4 dataset that we use consists of ∼1k terrorism events.
- Each *role* is filled with one or more entities. There are five such roles for MUC-4: perpetrator individuals (PERPIND), perpetrator organizations (PERPORG), physical targets (TARGET), victims (VICTIM) and weapons (WEAPON). These event roles represent the agents, patients, and instruments associated with terrorism events (Huang and Riloff, 2012).
- Each *role-filler entity* is denoted by a single descriptive *mention*, a span of text from the input document. Because multiple such mentions for each entity may appear in the input, the gold-standard template lists all alternatives (shown in Figure 1), but systems are required to produce just one.

---

[3] The Message Understanding Conferences were a series of U.S. government-organized IE evaluations.

[4] Other types of role fillers include normalized dates and times, and categorical "set" fills. We do not attempt to handle these in the current work.

Evaluation Metric The metric for past work on document-level role-filler *mentions* extraction (Patwardhan and Riloff, 2009; Huang and Riloff, 2011; Du and Cardie, 2020) calculates mention-level precision across all alternative mentions for each role-filler entity. Thus it is not suited for our problem setting, where entity-level precision is needed, where spurious entity extractions will get punished (e.g., recognizing "telephone company building" and "telephone company offices" as two entities will result in lower precision).

Drawing insights from the entity-based CEAF metric (Luo, 2005) from the coreference resolution literature, we design a metric (**CEAF-REE**) for measuring models' performance on this document-level **role-filler entity extraction** task. It is based on maximum bipartite matching algorithm (Kuhn, 1955; Munkres, 1957). The general idea is that, for each role, the metric is computed by aligning gold and predicted entities with the constraint that a predicted (gold) entity is aligned with at most one gold (predicted) entity. Thus, the system that does not recognize the coreferent mentions and use them for separate entities will be penalized in precision score. For the example in Figure 1, if the system extracts "Pilmai telephone company building" and "telephone company offices" as two distinct TARGETs, the precision will drop. We include more details for our CEAF-TF metric in the appendix.

4 REE as Sequence Generation

We treat document-level REE as a sequence-to-sequence task (Sutskever et al., 2014) in order to better model the cross-role dependencies and cross-sentence noun phrase coreference structure. We first transform the task definition into a source and target sequence.

As shown in Figure 2, the *source sequence* simply consists of the tokens of the original document prepended with a "classification" token (i.e., [CLS] in BERT), and appended with a separator token (i.e., [SEP] in BERT). The *target sequence* is the concatenation of target extractions for each role, separated by the separator token. For each role, the target extraction consists of the first mention's beginning (b) and end (e) tokens:

$$\begin{gathered} \text{<S>}\ e^{(1)}_{1_b}, e^{(1)}_{1_e}, \ldots\ \text{[SEP]} \\ e^{(2)}_{1_b}, e^{(2)}_{1_e}, \ldots\ \text{[SEP]} \\ e^{(3)}_{1_b}, e^{(3)}_{1_e}, e^{(3)}_{2_b}, e^{(3)}_{2_e}, \ldots\ \text{[SEP]} \\ \ldots \end{gathered}$$

Note that we list the roles in a fixed order for all examples. So for the example used in Figure 2, $e^{(1)}_{1_b}$, $e^{(1)}_{1_e}$ would be "two" and "men" respectively; and $e^{(3)}_{1_b}$, $e^{(3)}_{1_e}$ would be "water" and "pipes" respectively. Henceforth, we denote the resulting sequence of source tokens as $x_0, x_1, ..., x_m$ and the sequence of target tokens as $y_0, y_1, ..., y_n$.

5 Model: Generative Role-filler Transformer (GRIT)

Our model is shown in Figure 2. It consists of two parts: the encoder (left) for the source tokens; and the decoder (right) for the target tokens. Instead of using a sequence-to-sequence learning architecture with separate modules (Sutskever et al., 2014; Bahdanau et al., 2015), we use a single pretrained transformer model (Devlin et al., 2019) for both parts, and introduce no additional fine-tuned parameters.

Pointer Embeddings The first change to the model is to ensure that the decoder is aware of where its previous predictions come from in the source document, an approach we call "pointer embeddings". Similar to BERT, the input to the model consists of the sum of token, position and segment embeddings. However, for the position we use the corresponding source token's position. For example, for the word "two", the target tokens would have the identical position embedding of the word "two" in the source document. Interestingly, we do not use any explicit target position embeddings, but instead separate each role with a [SEP] token. Empirically, we find that the model is able to use these separators to learn which role to fill and which mentions have filled previous roles.

Our encoder's embedding layer uses standard BERT embedding layer, which applied to the source document tokens. To denote boundary between source and target tokens, we use sequence A (first sequence) segment embeddings for the source tokens, we use sequence B (second sequence) segment embeddings for the target tokens.

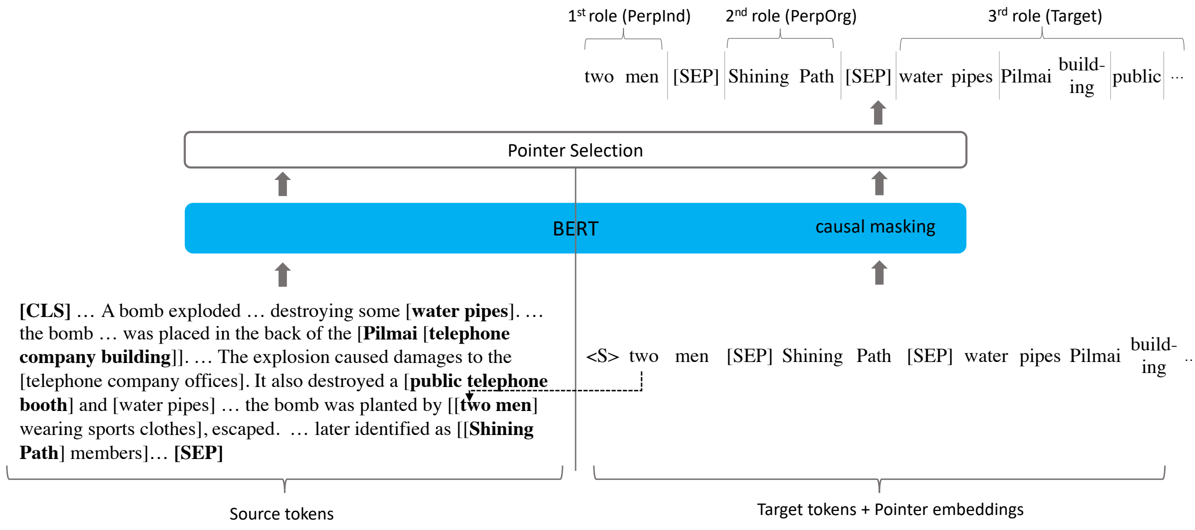


Figure 2: GRIT: generative transformer model for document-level event role-filler entity extraction. (Noun phrase bracketing and **bold** in the source tokens are provided for readability purposes and are not part of the source sequence.)

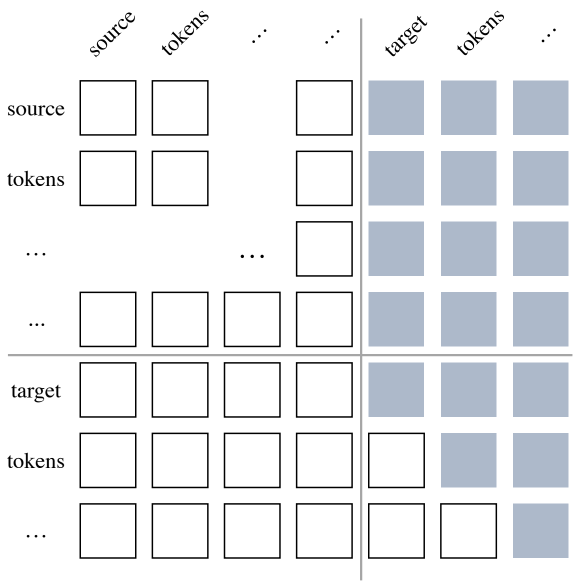


Figure 3: Partially causal masking strategy ($\mathbf{M}$). (White cell: unmasked; Grey cell: masked).

We pass the source document tokens through the encoder's embedding layer, to obtain their embeddings $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m$. We pass the target tokens $y_0, y_1, ..., y_n$ through the decoder's embedding layer, to obtain $\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_n$.

BERT as Encoder / Decoder We utilize one BERT model as both the source and target embeddings. To distinguish the encoder / decoder representations, we provide a partial causal attention mask on the decoder side.

In Figure 3, we provide an illustration for the attention masks – 2-dimensional matrix denoted as $\mathbf{m}$. For the source tokens, the mask allows full source self-attention, but mask out all target tokens. For $i \in \{0, 1, ..., m\}$,

$$\mathbf{M}_{i,j} = \begin{cases} 1, & \text{if } 0 \leq j \leq m \\ 0, & \text{otherwise} \end{cases}$$

For the target tokens, to guarantee that the decoder is autoregressive (the current token should not attend to future tokens), we use a causal masking strategy. Assuming we concatenate the target to the source tokens (the joint sequence mentioned below), for $i \in \{m + 1, ..., n\}$,

$$\mathbf{M}_{i,j} = \begin{cases} 1, & \text{if } 0 \leq j \leq m \\ 1, & \text{if } j > m \text{ and } j \leq i \\ 0, & \text{otherwise} \end{cases}$$

The joint sequence of source tokens' embeddings ($\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m$) and target tokens' embeddings ($\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_n$) are passed through BERT to obtain their contextualized representations,

$$\begin{aligned} &\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_m, \hat{\mathbf{y}}_0 ..., \hat{\mathbf{y}}_n \\ &= \mathtt{BERT}(\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m, \mathbf{y}_0, ..., \mathbf{y}_n) \end{aligned}$$

Pointer Decoding For the final layer, we replace word prediction with a simple pointer selection mechanism. For target time step t ($0 \leq t \leq n$), we first calculate the dot-product between $\hat{\mathbf{y}}_t$ and $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_m$,

$$z_0, z_1, ..., z_m = \hat{\mathbf{y}}_t \cdot \hat{\mathbf{x}}_0, \hat{\mathbf{y}}_t \cdot \hat{\mathbf{x}}_1, ..., \hat{\mathbf{y}}_t \cdot \hat{\mathbf{x}}_m$$

Then we apply softmax to $z_0, z_1, ..., z_m$ to obtain the probabilities of pointing to each source token,

$$p_0, p_1, ..., p_m = \text{softmax}(z_0, z_1, ..., z_m)$$

Test prediction is done with greedy decoding. At each time step t, argmax is applied to find the source token which has the highest probability. The predicted token is added to the target sequence for the next time step $t + 1$ with its pointer embedding. We stop decoding when the fifth [SEP] token is predicted, which represents the end of extractions for the last role.

In addition, we add the following decoding constraints,

- Tune probability of generating [SEP]. By doing this, we encourage the model to point to other source tokens and thus extract more entities for each role, which will help increase the recall. (We set the hyperparameter of downweigh to 0.01, i.e., for the [SEP] token $p = 0.01 * p$.)
- Ensure that the token position increase from start token to end token. When decoding tokens for each role, we know that mention spans should obey this property. Thus we eliminate those invalid choices during decoding.

6 Experimental Setup

We conduct evaluations on the MUC-4 dataset (1992), and compare to recent competitive end-to-end models (Wadden et al., 2019; Du and Cardie, 2020) in IE (Section 7). Besides the normal evaluation, we are also interested in how well our GRIT model captures coreference linguistic knowledge, and comparison with the prior models. In Section 8, we present relevant evaluations on the subset of test documents.

Dataset and Evaluation Metric The MUC-4 dataset consists of 1,700 documents with associated templates. Similar to (Huang and Riloff, 2012; Du and Cardie, 2020), we use the 1300 documents for training, 200 documents (`TST1+TST2`) as the development set and 200 documents (`TST3+TST4`) as the test set. Each document in the dataset contains on average 403.27 tokens, 7.12 paragraphs. In Table 1, we include descriptions for each role in the template.

We use the first appearing mention of the role-filler entity as the training signal (thus do not use the other alternative mentions during training).

Roles	Descriptions
PERPIND	A person responsible for the incident.
PERPORG	An organization responsible for the incident.
TARGET	A thing (inanimate object) that was attacked.
VICTIM	The name of a person who was the obvious or apparent target of the attack or who became a victim of the attack.
WEAPON	A device used by the perpetrator(s) in carrying.

Table 1: Natural Language Descriptions for Each Role.

We use CEAF-REE which is covered in Section 3 as the evaluation metric. The results are reported as Precision (P), Recall (R) and F-measure (F1) score for the micro-average for all the event roles (Table 4). We also report the per-role results to have a fine-grained understanding of the numbers (Table 2).

Baselines We compare to recent strong models for (document-level) information/event extraction. CohesionExtract (Huang and Riloff, 2012) is a bottom-up approach for event extraction that first aggressively identifies candidate role-fillers, and prune the candidates located in event-irrelevant sentences.[5] Du and Cardie (2020) propose *neural sequence tagging (NST)* models with contextualized representations for document-level role filler mentions extraction. We train this model with BIO tagging scheme to identify the first mention for each role-filler entity and its type (i.e., B-PerpInd, I-PerpInd for perpetrator individual). DYGIE++ (Wadden et al., 2019) is a span-enumeration based extraction model for entity, relation, and event extraction. The model (1) enumerates all the possible spans in the document; (2) concatenates the representations of the span's beginning & end token and use it as its representation, and pass it through a classifier layer to predict whether the span represents certain role-filler entity and what the role is. Both the NST and DYGIE++ are end-to-end and fine-tuned BERT (Devlin et al., 2019) contextualized representations with task-specific data. We train them to identify the first mention for each role-filler entity (to ensure fair comparison with our proposed model). Unsupervised event schema induction based approaches (Chambers and Jurafsky, 2011; Chambers, 2013; Cheung et al., 2013) are also able

---

[5] Instead of using feature-engineering based sentence classification to identify event-relevant sentences, we re-implement the sentence classifier with BiLSTM-based neural sequence model.

	PerpInd	PerpOrg	Target	Victim	Weapon
NST (Du and Cardie, 2020)	48.39 / 32.61 / 38.96	60.00 / 43.90 / 50.70	54.96 / 52.94 / **53.93**	62.50 / 63.16 / 62.83	61.67 / 61.67 / **61.67**
DyGIE++ (Wadden et al., 2019)	59.49 / 34.06 / 43.32	56.00 / 34.15 / 42.42	53.49 / 50.74 / 52.08	60.00 / 66.32 / 63.00	57.14 / 53.33 / 55.17
GRIT	65.48 / 39.86 / **49.55**	66.04 / 42.68 / **51.85**	55.05 / 44.12 / 48.98	76.32 / 61.05 / **67.84**	61.82 / 56.67 / 59.13

Table 2: Per-role performance scored by CEAF-REE (reported as P/R/F1, highest F1 for each role are boldfaced).

	$k = 1$	$1 < k \leq 1.25$	$1.25 < k \leq 1.5$	$1.5 < k \leq 1.75$	$k > 1.75$
NST (Du and Cardie, 2020)	63.83 / 51.72 / 57.14	57.45 / 38.57 / 46.15	60.32 / 49.03 / 54.09	64.81 / 50.00 / 56.45	66.67 / 51.90 / 58.36
DyGIE++ (Wadden et al., 2019)	**72.50** / 50.00 / 59.18	70.00 / 40.00 / 50.91	60.48 / 48.39 / 53.76	52.94 / 38.57 / 44.63	66.96 / 48.73 / 56.41
GRIT	65.85 / 46.55 / 54.55	**74.42** / 45.71 / 56.64	**73.20** / 45.81 / 56.35	**67.44** / 41.43 / 51.33	**69.75** / 52.53 / 59.93

Table 3: Evaluations on the subsets of documents with increasing number of mentions per role-filler entity. k denotes the average # mentions per role-filler entity. Results for each column are reported as Precision / Recall / F1. The highest precisions are boldfaced for each bucket.

Models	P	R	F1
CohesionExtract (Huang and Riloff, 2012)	58.38	39.53	47.14
NST (Du and Cardie, 2020)	56.82	**48.92**	52.58
DyGIE++ (Wadden et al., 2019)	57.04	46.77	51.40
GRIT	**64.19****	47.36	**54.50***

Table 4: Micro-average results (the highest number of each column is boldfaced). Significance is indicated with $^{**}(p < 0.01)$,$^{*}(p < 0.1)$ – all tests are computed using the paired bootstrap procedure (Berg-Kirkpatrick et al., 2012).

to model the coreference relations and entities at document-level, but have been proved to perform substantially worse than supervised models (Patwardhan and Riloff, 2009; Huang and Riloff, 2012). Thus we do not compare with them. We also experimented with a variant of our GRIT model – instead of always pointing to the same [SEP] in the source tokens to finish extracting the role-filler entities for a role, we use five different [SEP] tokens. During decoding, the model points to the corresponding [SEP] as the end of extraction for that role. This variant does not improve over the current best results and we omit reporting its performance.

7 Results

In Table 4, we report the micro-average performance on the test set. We observe that our GRIT model substantially outperforms the baseline extraction models in precision and F1, with an over 5% improvement in precision over DyGIE++.

Table 2 compares the models' performance scores on each role (PerpInd, PerpOrg, Target, Victim, Weapon). We see that, (1) our model achieves the best precision across the roles; (2) for the roles that come with entities containing more human names (e.g., PerpInd and Victim), our model substantially outperforms the baselines; (3) for the role PerpOrg, our model scores better precision but lower recall than neural sequence tagging, which results in a slightly better F1 score; (4) for the roles Target and Weapon, our model is more conservative (lower recall) and achieves lower F1. One possibility is that for role like Target, on average there are more entities (though with only one mention each), and it's harder for our model to decode as many Target entities correct in a generative way.

8 Discussion

How well do the models capture coreference relations between mentions? We also conduct targeted evaluations on subsets of test documents whose gold extractions come with coreferent mentions. From left to right in Table 3, we report results on the subsets of documents with increasing number (k) of possible (coreferent) mentions per role-filler entity. We find that: (1) On the subset of documents with only one mention for each role-filler entity ($k = 1$), our model has no significant advantage over DyGIE++ and the sequence tagging based model; (2) But as k increases, the advantage of our GRIT substantially increases – with an over 10% gap in precision when $1 < k \leq 1.5$, and a near 5% gap in precision when $k > 1.5$.

From the qualitative example (document excerpt and the extractions in Figure 4), we also observe our model recognizes the coreference relation between candidate role-filler entity mentions, while the baselines do not, which shows that our model is better at capturing the (non-)coreference relations between role-filler entity mentions. It also proves the advantage of a generative model in this setting.

[P1]... a bomb exploded at the front door of the [home of a peruvian army general], causing damages but no casualties. ... **[P2]** The terrorist attack was ..., by ... who hurled a bomb at the [home of general enrique franco], in the San ... **[P3]** The bomb seriously damaged the [general's [vehicle]], ... and those of [neighboring [houses]].

	Target
Gold Role-filler Entities	• home of peruvian army general, home of general enrique franco • vehicle, general's vehicle • houses, neighboring houses
NST	• home of peruvian army general • home of general enrique franco
DyGIE++	• home of peruvian army general • home of general enrique franco • houses
GRIT	• home of peruvian army general • houses

Figure 4: Our model implicitly captures coreference relations between mentions.

How well do models capture dependencies between different roles? To study this phenomenon, we consider nested role-filler entity mentions in the documents. In the example of Figure 1, "shining path" is a role-filler entity mention for PerpOrg nested in "two *shining path* members" (a role-filler entity mention for PerpInd). The nesting happens more often between more related roles (e.g., PerpInd and PerpOrg) – we find that 33 out of the 200 test documents' gold extractions contain nested role-filler entity mentions between the two roles.

In Table 5, we present the CEAF-REE scores for role PerpOrg on the subset of documents with nested roles. As we hypothesized beforehand, GRIT is able to learn the dependency between different roles and can learn to avoid missing relevant role-filler entities for later roles. The results provide empirical evidence: by learning the dependency between PerpInd and PerpOrg, GRIT improves the relative recall score on the subset of documents as compared to DyGIE++. On all the 200 test documents, our model is $\sim 2\%$ below DyGIE++ in recall; while on the 33 docs, our model scores much higher than DyGIE++ in recall.

	PerpOrg (all docs)	PerpOrg (33/200)
	P / R / F1	P / R / F1
NST	56.00 / 34.15 / 42.42	80.00 / 44.44 / 57.14
DyGIE++	60.00 / **43.90** / 50.70	61.54 / 35.56 / 45.07
GRIT	66.04 / 42.68 / 51.85	80.77 / **46.67** / 59.15

Table 5: Evaluation on the subset of documents that have nested role-filler entity mentions between role PerpInd and PerpOrg (highest recalls boldfaced).

...[[[**guerrillas**] of the [**FARC**] and the [**popular liberation army**]] (EPL)] attacked four towns in northern Colombia, leaving 17 guerrillas and 2 soldiers dead and 3 bridges partially destroyed. ...

	PerpInd	PerpOrg
Gold Role-filler Entities	• guerrillas, guerrillas of FARC and popular liberation army (EPL)	• EPL, popular liberation army • FARC
NST & DyGIE++	• guerrillas	-
GRIT	• guerrillas	• FARC • popular liberation army

Figure 5: Our model captures dependencies between different roles.

For the document in the example of Figure 5, our model correctly extracts the two role-filler entities for PerpOrg: "FARC" and "popular liberation army", which are closely related to the PerpInd entity "guerrilla". While DyGIE++ and NST both miss the entities for PerpOrg.

Decoding Ablation Study In the table below, we present ablation results based on the decoding constraints. These illustrate the influence of the decoding constraints on the our model's performance. The two constraints both significantly improve model predictions. Without downweighing the probability of pointing to [SEP], the precision increases but recall and F1 significantly drops.

	P	R	F1	Δ (F1)
GRIT	64.19	47.36	54.50	
− [SEP] downweigh	67.43	40.12	50.31	-4.19
− constraint on pointer offset	62.90	45.79	53.00	-1.50

Table 6: Decoding Ablation Study

Additional Parameters and Training Cost Finally we consider additional parameters and training time of the models: As we introduced previously, the baseline models DYGIE++ and NST both require an additional classifier layer on top of BERT's hidden state (of size H) for making the predictions. While our GRIT model does not require adding any new parameters. As for the training time, training the DYGIE++ model takes over 10 times longer time than NST and our model. This time comes from the DYGIE++ model requirement of enumerating all possible spans (to a certain length constraint) in the document and calculating the loss with their labels.

	additional params	training cost
DYGIE++	$2H(\#\text{roles} + 1)$	~20h
NST	$H(2\#\text{roles} + 1)$	~1h
GRIT	0	<40min

Table 7: Additional Parameters and Training Cost.

9 Conclusion

We revisit the classic and challenging problem of document-level role-filler entity extraction (REE), and find that there is still room for improvement. We introduce an effective end-to-end transformer based generative model, which learns the document representation and encodes the dependency between role-filler entities and between event roles. It outperforms the baselines on the task and better captures the coreference linguistic phenomena. In the future, it would be interesting to investigate how to enable the model to also do template recognition.

Acknowledgments

We thank the anonymous reviewers for helpful feedback and suggestions. The work of XD and AMR was supported by NSF CAREER 2037519; that of XD and CC was supported in part by DARPA LwLL Grant FA8750-19-2-0039.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Seattle, Washington, USA. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846, Atlanta, Georgia. Association for Computational Linguistics.

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Xinya Du and Claire Cardie. 2020. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020, Online. Association for Computational Linguistics.

Shaoyang Duan, Ruifang He, and Wenli Zhao. 2017. Exploiting document level information to improve event detection via recurrent neural networks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, pages 352–361, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

Xiaocheng Feng, Bing Qin, and Ting Liu. 2018. A language-independent neural network for event detection. *Science China Information Sciences*, 61(9):092106.

Ralph Grishman and Beth Sundheim. 1996. Design of the MUC-6 evaluation. In *TIPSTER TEXT PROGRAM PHASE II: Proceedings of a Workshop held at Vienna, Virginia, May 6-8, 1996*, pages 413–422, Vienna, Virginia, USA. Association for Computational Linguistics.

Tianyu He, Xu Tan, Yingce Xia, Di He, Tao Qin, Zhibo Chen, and Tie-Yan Liu. 2018. Layer-wise coordination between encoder and decoder for neural machine translation. In *Advances in Neural Information Processing Systems*, pages 7944–7954.

Ruihong Huang and Ellen Riloff. 2011. Peeling back the layers: Detecting event role fillers in secondary contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1137–1147, Portland, Oregon, USA. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extraction. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Robin Jia, Cliff Wong, and Hoifung Poon. 2019. Document-level n-ary relation extraction with multiscale representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3693–3704, Minneapolis, Minnesota. Association for Computational Linguistics.

Dan Jurafsky and James H. Martin. 2014. *Speech and language processing*. Prentice Hall, Pearson Education International.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Xiang Li, Thien Huu Nguyen, Kai Cao, and Ralph Grishman. 2015. Improving event detection with abstract meaning representation. In *Proceedings of the First Workshop on Computing News Storylines*, pages 11–15, Beijing, China. Association for Computational Linguistics.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797, Uppsala, Sweden. Association for Computational Linguistics.

(LDC) Linguistic Data Consortium. 2005. English annotation guidelines for events. *https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf*.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, and Yue Zhang. 2019. Open domain event extraction using neural latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2860–2871, Florence, Italy. Association for Computational Linguistics.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

MUC-4. 1992. Fourth message understanding conference (MUC-4). In *Proceedings of FOURTH MESSAGE UNDERSTANDING CONFERENCE (MUC-4)*, McLean, Virginia.

James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 365–371, Beijing, China. Association for Computational Linguistics.

Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 151–160, Singapore. Association for Computational Linguistics.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Beth M. Sundheim. 1991. Overview of the third Message Understanding Evaluation and Conference. In *Third Message Uunderstanding Conference (MUC-3): Proceedings of a Conference Held in San Diego, California, May 21-23, 1991*.

Beth M. Sundheim. 1993. The Message Understanding Conferences. In *TIPSTER TEXT PROGRAM: PHASE I: Proceedings of a Workshop held at Fredricksburg, Virginia, September 19-23, 1993*, pages 5–5, Fredericksburg, Virginia, USA. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57.

Hai Wang and Hoifung Poon. 2018. Deep probabilistic logic: A unifying framework for indirect supervision. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1891–1902, Brussels, Belgium. Association for Computational Linguistics.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence*, 1(2):99–120.

Yue Zhao, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. Document embedding enhanced event detection with hierarchical and supervised attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 414–419, Melbourne, Australia. Association for Computational Linguistics.

Cross-lingual Entity Alignment with Incidental Supervision

Muhao Chen[1,2*], **Weijia Shi**[3*], **Ben Zhou**[1], **Dan Roth**[1]
[1]Department of Computer and Information Science, UPenn
[2]Viterbi School of Engineering, USC
[3]Department of Computer Science, UCLA
`{muhao,xyzhou,danroth}@seas.upenn.edu`; `swj0419@cs.ucla.edu`

Abstract

Much research effort has been put to multilingual knowledge graph (KG) embedding methods to address the entity alignment task, which seeks to match entities in different language-specific KGs that refer to the same real-world object. Such methods are often hindered by the insufficiency of seed alignment provided between KGs. Therefore, we propose an incidentally supervised model, JEANS, which jointly represents multilingual KGs and text corpora in a shared embedding scheme, and seeks to improve entity alignment with incidental supervision signals from text. JEANS first deploys an entity grounding process to combine each KG with the monolingual text corpus. Then, two learning processes are conducted: (i) an *embedding learning process* to encode the KG and text of each language in one embedding space, and (ii) a self-learning based *alignment learning process* to iteratively induce the matching of entities and that of lexemes between embeddings. Experiments on benchmark datasets show that JEANS leads to promising improvement on entity alignment with incidental supervision, and significantly outperforms state-of-the-art methods that solely rely on internal information of KGs.[1]

1 Introduction

A multilingual knowledge base (KB) such as DBpedia (Lehmann et al., 2015), ConceptNet (Speer et al., 2017) and Yago (Mahdisoltani et al., 2015) stores multiple language-specific knowledge graphs (KGs) that express relations of many concepts and real-world entities. As each KG thereof is either extracted independently from monolingual corpora (Lehmann et al., 2015; Mahdisoltani et al., 2015) or contributed by speakers of the language (Speer et al., 2017; Mitchell et al., 2018), it is common for different KGs to constitute complementary knowledge (Bleiholder and Naumann, 2009; Bryl and Bizer, 2014). Hence, aligning and synchronizing language-specific KGs support AI systems with more comprehensive commonsense reasoning (Lin et al., 2019; Li et al., 2019b; Yeo et al., 2018), and benefit various knowledge-driven NLP tasks, including machine translation (Moussallem et al., 2018), narrative prediction (Chen et al., 2019) and dialogue agents (Sun et al., 2019a).

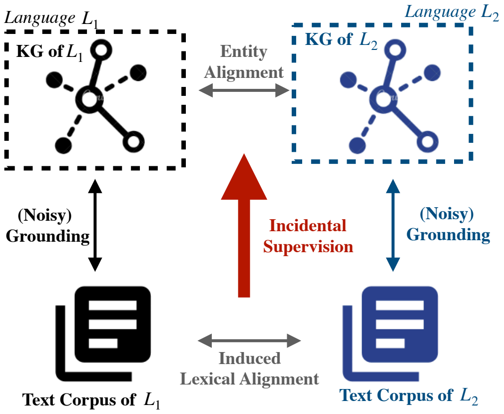


Figure 1: The learning framework of JEANS.

Learning to align multilingual KGs is a nontrivial task, as KGs with distinct surface forms, heterogeneous schemata and inconsistent structures easily cause traditional symbolic methods to fall short (Suchanek et al., 2011; Wijaya et al., 2013; Jiménez-Ruiz et al., 2012). Recently, much attention has been paid to methods based on multilingual KG embeddings (Chen et al., 2017a,b, 2018; Sun et al., 2017, 2018, 2019b; Zhang et al., 2019). Those methods seek to separately encode the structure of each language-specific KG in an embedding space. Then, based on some seed entity alignment, the entity counterparts in different KGs can be easily matched via distances or transformations of embedding vectors. The principle is

---

[*] Indicating equal contributions.

[1] Software and resources are available at `http://cogcomp.org/page/publication_view/929`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 645–658
April 19 - 23, 2021.

that entities with relevant neighborhood information can be characterized with similar embedding representations. Such representations particularly are tolerant to the aforementioned heterogeneity of surface forms and schemata in language-specific KGs (Chen et al., 2017a; Sun et al., 2018, 2020a).

While multilingual KG embeddings provide a general and tractable way to align KGs, it still remains challenging for related methods to precisely infer the cross-lingual correspondence of entities. The challenge is that the seed entity alignment, which serves as the essential training data to learn the connection between language-specific KG embeddings, is often limitedly provided in KBs (Chen et al., 2018; Sun et al., 2018). Hence, the lack of supervision often hinders the precision of inferred entity counterparts, and affects even more significantly when KGs scale up and become inconsistent in contents and density (Pujara et al., 2017). Several methods also gain auxiliary supervision from profile information of entities, including descriptions (Chen et al., 2018; Yang et al., 2019; Zhang et al., 2019) and numerical attributes (Sun et al., 2017; Trsedya et al., 2019). However, such profile information is not available in many KGs (Speer et al., 2017; Mitchell et al., 2018; Bond and Foster, 2013), therefore causing these methods to be not generally applicable.

Unlike existing models that rely on internal information of KGs, we seek to create embeddings that incorporate both KGs and freely available text corpora, and exploit incidental supervision signals (Roth, 2017) from text corpora to enhance the alignment learning on KGs (Figure 1). In this paper, we propose a novel embedding model `JEANS` (**J**oint **E**mbedding Based Entity **A**lignment with **IN**cidental **S**upervision; 👖). Particularly, `JEANS` first performs a *grounding process* (Gupta et al., 2017; Upadhyay et al., 2018) to link entity mentions in each monolingual text corpus to the KG of the same language. Based on the KGs and grounded text in a pair of languages, `JEANS` conducts two learning processes, i.e. *embedding learning* and *alignment learning*. The embedding learning process distributes entities, relations and lexemes of each language in its embedding space, in which a KG embedding model and a language model for that language are jointly trained. This process seeks to leverage text contexts to help capture the proximity of entities. On top of that, alignment learning captures cross-lingual correspondence for entities and lexemes in a self-learning manner (Artetxe et al., 2018). Starting from a small amount of seed entity alignment, this process iteratively induces a transformation between language-specific embedding spaces, and infers more alignment of entity and lexemes at each iteration to improve the learning at the next one. Moreover, we also employ the closed-form Procrustes solution (Conneau et al., 2018) to strengthen alignment induction within each iteration. Experimental results on two benchmarks confirm the effectiveness of `JEANS` in leveraging incidental supervision, leading to significant improvement to entity alignment and drastically outperforming existing methods.

2 Related Work

We discuss relevant works in four topics. Each of them has a large body of work which we can only provide as a highly selected summary.

Entity alignment. Entity alignment in KBs has been a long-standing problem (Shvaiko and Euzenat, 2011). Aside from earlier approaches based on symbolic or schematic similarity of entities (Suchanek et al., 2011; Wijaya et al., 2013; Jiménez-Ruiz et al., 2012), more recent research addresses this task with multilingual KG embeddings. A representative method of such is MTransE (Chen et al., 2017a). MTransE jointly learns two model components. There are a translational embedding model (Bordes et al., 2013) that distributes the facts in language-specific KGs into separate embeddings, and a transformation-based alignment model that maps between entity counterparts across embedding spaces.

Following the general principle of MTransE, later approaches are developed through the following three lines. One is to incorporate various *embedding learning techniques* for KGs. Besides translational techniques, some models employ alternative *relation modeling* techniques to encode relation facts, such as circular correlation (Nickel et al., 2016), Hadamard product (Hao et al., 2019) and recurrent skipping networks (Guo et al., 2019). Others encode entities with *neighborhood aggregation* techniques, including GCN (Wang et al., 2018; Yang et al., 2019; Cao et al., 2019; Xu et al., 2019; Wu et al., 2019b), RGCN (Wu et al., 2019a) and GAT (Zhu et al., 2019). Their benefits are mainly to produce entity representations capturing high-order proximity, so as to better suit the alignment task. A few works follow the second line to enhance the *alignment learning* with semi-supervised learning techniques. Representative ones include co-training (Chen et al., 2018), optimal transport (Pei et al., 2019b) and bootstrapping (Sun et al., 2018), which improve the preciseness of alignment captured with limited supervision. The third line of research seeks to obtain additional supervision from entity profiles, including descriptions (Chen et al., 2018; Yang et al., 2019), attributes (Sun et al., 2017; Trsedya et al., 2019; Pei et al., 2019a; Yang et al., 2020) and KG schemata (Zhang et al., 2019). While those alternative views of entities can effectively bridge the embeddings, the limitation of such methods lies in the unavailability of those views in many KGs (Speer et al., 2017; Mitchell et al., 2018; Bond and Foster, 2013). A survey on the entity alignment problem by Sun et al. (2020b) has provided a more comprehensive summarization of recent advances in these lines.

Our method is mainly related to the third line of research. While instead of leveraging specific intra-KB information, our method introduces supervision signals from text contexts that are freely accessible to almost any KBs with the aid of grounding techniques. Meanwhile, our paper also follows the second line to improve alignment learning techniques, and couples two mainstream techniques for embedding learning.

Joint embeddings of entities and text. Fewer efforts have been put to jointly characterize entities and text as embeddings. Wang et al. (2014b) propose to connect a translational embedding of Freebase (Bollacker et al., 2008) to a English word embedding based on Wikipedia anchors, therefore providing a joint embedding to enhance link prediction in the KG. Zhong et al. (2015) generalize the approach by Wang et al. (2014b) with distant supervision based on entity descriptions and text corpora. Toutanova et al. (2015) extract dependency paths from sentences and jointly embed them with a KG using DistMult (Yang et al., 2015) to support the relation extraction task. Several other approaches focus on jointly embedding words, entities (Yamada et al., 2017; Newman-Griffis et al., 2018; Cao et al., 2017; Almasian et al., 2019) and entity types (Gupta et al., 2017) appearing in the same textual contexts without considering relational structure of a KG. These approaches are employed in monolingual NLP tasks including entity linking (Gupta et al., 2017; Cao et al., 2017), entity abstraction (Newman-Griffis et al., 2018) and factoid QA (Yamada et al., 2017). As they focus on a monolingual and supervised scenario, they are essentially different from our goal to help cross-lingual KG alignment with incidental supervision from unparalleled corpora.

Multilingual word embeddings. Our model component of alignment induction from text is closely connected to multilingual word embeddings. Earlier approaches in this line, regardless of being supervised or weakly supervised, based on seed lexicon (Zou et al., 2013) or parallel corpora (Gouws et al., 2015), are systematically summarized in a recent survey (2017). While a number of methods in this line can be employed in our model to gain addition supervision for entity alignment, we choose to use a combination of Procrustes solution (Conneau et al., 2018) with self-learning to offer precise inference of cross-lingual alignment based on limited seed alignment. Note that recent contextualized embeddings such M-BERT (Pires et al., 2019) and XLM-R (Conneau et al., 2020) do not directly suit our problem setting, since contextualization could cause ambiguity to entity representations, therefore impairing the alignment of entities across embedding spaces.

Incidental supervision. Incidental supervision is a recently introduced learning strategy (Roth, 2017), which seeks to retrieve supervision signals from data that are not labeled for the target task. This strategy has been applied to tasks including SRL (He et al., 2020), controversy prediction (Rethmeier et al., 2018) and dataless classification (Song and Roth, 2015). To the best of our knowledge, the proposed method here is the first of its kind that incorporates incidental supervision in embedding learning or alignment.

3 Method

We hereby begin introducing our method with the formalization of learning resources.

In a KB, $\mathcal{L}$ denotes the set of languages, and $\mathcal{L}^2$ unordered language pairs. G_L is the language-specific KG of language $L \in \mathcal{L}$. E_L and R_L respectively denote the corresponding vocabularies of entities and relations. $T = (h, r, t)$ denotes a triple in G_L such that $h, t \in E_L$ and $r \in R_L$. Boldfaced $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ represent the embedding vectors of head h, relation r, and tail t respectively. For a language pair $(L_1, L_2) \in \mathcal{L}^2$, $I_E(L_1, L_2)$ denotes a set of entity alignments between L_1 and L_2, such that $e_1 \in E_{L_1}$ and $e_2 \in E_{L_2}$ for each entity pair $(e_1, e_2) \in I_E(L_1, L_2)$. Following the convention of previous work (Chen et al., 2018; Sun et al., 2018; Yang et al., 2019), we assume the entity pairs to have a 1-to-1 mapping and it is specified in $I_E(L_1, L_2)$. This assumption is congruent to the design of mainstream KBs (Lehmann et al., 2015; Mahdisoltani et al., 2015) where disambiguation of entities is granted. Besides the definition of multilingual KGs, we use D_L to denote the text corpus of language L. D_L is a set of documents $\{d_L\}$, where each document $d_L = [w_1, w_2, ..., w_l]$ is a sequence of tokens from the monolingual lexicon W_L. Each token w_i thereof is originally a lexeme, but may also be an entity surface form after the ground process, and we also use boldfaced $\mathbf{w}_i$ to denote its vector. $I_W(L_1, L_2)$ denotes the seed lexicon between (L_1, L_2), such that $w_1 \in W_{L_1}$ and $w_2 \in W_{L_2}$ for each lexeme pair $(w_1, w_2) \in I_W(L_1, L_2)$. Note that I_W only include the alignment between lexemes, and may optionally serve as external supervision data. To be consistent with previous problem settings of entity alignment (Chen et al., 2017a; Sun et al., 2018; Yang et al., 2019), I_W is not necessarily provided to training, but is defined to be compatible with the scenarios where it is available.

`JEANS` addresses entity alignment in three consecutive processes. (i) A *grounding process* first link entities of each KG to possible mentions of them in the corresponding monolingual corpus, therefore connecting entities and text tokens of the same language into a shared vocabulary. (ii) An *embedding learning* process characterizes the KG and text of each language into a separate embedding space. In this process, we couple both the translational technique (Bordes et al., 2013; Chen et al., 2017a, 2018) and the neighborhood aggregation technique (Wang et al., 2018; Yang et al., 2019), which are two representative techniques to characterize a KG. Simultaneously, the monolingual text tokens are encoded with a skip-gram language model (Mikolov et al., 2013). (iii) On top of the embeddings, starting from a small amount of seed entity alignment and *optional* seed lexicon, the *alignment learning* process iteratively infers more alignment both on KGs and text using self-learning and Procrustes solution (Schönemann, 1966). The processes of `JEANS`'s learning is consistent to Figure 1. The rest of this section introduces the technical details of each process.

3.1 (Noisy) Entity Grounding

The goal of the grounding process is to combine vocabularies of the KG and the text corpus in each language. This serves as the premise for the embedding learning process to produce a shared representation scheme for entities, relations and lexemes, therefore allowing the alignment learning process to leverage supervision signals for both entities and lexemes. It is noteworthy that, the purpose of entity grounding here is to combine the two data modalities. Hence, we only expect this process to discover enough entity contexts and offer a higher coverage on entity vocabularies, while being tolerant to possible noise in entity recognition and linking. Particularly, we consider two grounding techniques, one using a pre-trained entity discovery and linking (EDL) model, the other based on simple surface form matching (SFM).

Pre-trained EDL model. One technique is to use off-the-shelf EDL models (Khashabi et al., 2018; Manning et al., 2014). A typical model of such sequentially handles the steps of NER to detect entity mentions, and link each mention to candidate entities from the KG based on symbolic and contextual similarity. Many EDL models are easily trainable on large text corpora with anchors, and offer promising performance of grounding and disambiguation on multiple languages (Sil et al., 2018). In this paper, we do not go into details to the design of EDL models. Interested readers are referred to the aforementioned literature.

Surface form matching. Suppose a pre-trained EDL model is not available, then a simpler way of combining data is to match KG surface forms with text. This can be efficiently done by building a Completion Trie (Hsu and Ottaviano, 2013) for multi-token surface forms, and conducting a longest prefix matching (Dharmapurikar et al., 2006) between surface forms and sub-sequences of text tokens. While this simple technique does not necessarily disambiguate entity mentions, experiments find it sufficient to combine the two modalities and allow supervision signals from induced lexical alignment to propagate to entities.

Once the entity vocabulary E_L and the lexicon W_L of a language are combined, we assume that entity mentions in D_L are properly tokenized as grounded surface forms in $E_L \cap W_L$. Specifically, we now use x to denote a token in the grounded corpus D_L that can either be an entity e or a lexeme w. Given the combined learning resources for each language, we next describe the processes of embedding learning and alignment learning.

3.2 Embedding Learning

The embedding learning process is responsible for capturing the combined KG and text corpus of each language in a shared embedding space $\mathbb{R}^k$. In this process, `JEANS` jointly learns two model components to respectively encode units of the KG and the text, among which the overlaps $E_L \cap W_L$ use shared representations. We hereby describe these two model components in detail.

3.2.1 KG Embedding

As discussed in §2, previous approaches respectively leverage two forms of embedding learning techniques: (i) *relation modeling* (Chen et al., 2017a; Sun et al., 2018) such as vector translations, circular correlation and Hadamard product seeks to capture relations as an arithmetic operation in the vector space; (ii) *neighborhood aggregation* (Wang et al., 2018; Yang et al., 2019; Cao et al., 2019) employs graph neural networks (GNN) to encode neighborhood contexts for better seizing the proximity of entities.

The KG embedding model proposed in this work couples both forms of techniques. This aims at seizing both relations and entity proximity, two factors that are both beneficial to produce transferable entity embeddings. To achieve this goal, the encoder first stacks n layers of GCN (Kipf and Welling, 2016) on the KG. Formally, the l-th layer representation $\mathbf{E}^{(l)}$ is computed as

$$\mathbf{E}^{(l)} = \phi\left(D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}\mathbf{E}^{(l-1)}\mathbf{M}^{(l-1)}\right),$$

where D is the diagonal degree matrix D of the KG, $\tilde{A} = A + I$ is the sum of the adjacency matrix A and an identity I, and $\mathbf{M}^{(l-1)}$ is a trainable weight matrix. The raw features of entities $\mathbf{E}^{(0)}$ can be either entity attributes or randomly initialized. The last layer outputs are regarded as entity embedding representations, i.e. $\mathbf{E} = \mathbf{E}^{(n)}$.

We use $\mathbf{E}_L$ to denote the entity representations of language L, then the following log-softmax loss is optimized to perform relational modeling with translation vectors in the embedding space of L:

$$S_L^K = -\sum_{T \in G_L} \log \frac{\exp\left(b - f_r(h,t)\right)}{\sum_{\hat{T} \notin G_L} \exp\left(b - f_r(\hat{h},\hat{t})\right)},$$

where $f_r(h,t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$ is the plausibility measure of a triple (Bordes et al., 2013), $\hat{T} = (\hat{h}, r, \hat{t})$ is a Bernoulli negative-sampled triple (Wang et al., 2014a) created by substituting either head or tail entities h or t in $T = (h, r, t)$. b is a positive bias to adjust the scale of the plausibility measure. All the entity representations optimized in S_L^K are from $\mathbf{E}_L$. Note that the reason for us to choose the translational technique over other relation modeling techniques is due to this technique being more robust in cases where KG structures are sparser (Pujara et al., 2017).

3.2.2 Text Embedding

In addition to the KG embedding, the text embedding seeks to leverage the contextual information of free text to help the embedding better capture the proximity of entities This model employs the continuous skip-gram language model, which is inline with a number of word embedding methods (Mikolov et al., 2013; Bojanowski et al., 2017; Conneau et al., 2018), and is realized by optimizing the following log-softmax loss:

$$S_L^T = -\sum_{x \in E_L \cup W_L} \sum_{x_c \in C_{x,D_L}} \log \frac{\exp\left(d(x, x_c)\right)}{\sum_{x_n} \exp\left(d(x, x_n)\right)}.$$

The text context C_{x,D_L} thereof is the set of tokens that surround a token x in the entity-grounded corpus D_L, d denotes the l_2 distance, and x_n denotes a randomly sampled token in $E_L \cup W_L$.

3.2.3 Embedding Learning Objective

For each language $L \in \mathcal{L}$, the goal of embedding learning is to optimize the joint loss

$$S_L^E = S_L^K + S_L^T.$$

As mentioned, the grounded entity surface forms in $E_L \cap W_L$ use shared representations in both model components, hence are optimized with both S_L^K and S_L^T. The rest lexeme, relation and entity representations are optimized alternately by either component. In both model components, the number of negative samples of triples and tokens are both adjustable hyperparameters.

It is noteworthy that, both model components may choose alternative techniques, including other

KG encoders such as GAT (Veličković et al., 2018), multi-channel GCN (Cao et al., 2019) and gated GNN (Sun et al., 2020a), and text embeddings such as GloVe (Pennington et al., 2014). As experimenting with different embedding techniques is not a main contribution of this work, we leave them as future work. Specifically, contextualized text representations (Peters et al., 2018; Devlin et al., 2019) cannot directly apply, as contextualization will cause ambiguity to token representations that hinder the match of embeddings.

3.3 Alignment Learning

Once the KG and text units of each language are captured in a shared embedding, the alignment learning process therefore bridges the alignment between each pair of embeddings. This process seeks to exploit additional alignment labels from text embeddings, and use those to help the alignment of entities. Different from the majority of methods in §2 that jointly learn embeddings and alignment, the alignment learning process in `JEANS` is a retrofitting process (Shi et al., 2019; Faruqui et al., 2015). Hence, the embedding of each language is fixed and does not require duplicate training for different language pairs (Chen et al., 2017a; Sun et al., 2017).

Given a pair of languages $(L_i, L_j) \in \mathcal{L}^2$, the objective of alignment learning is to induce a transformation $\mathbf{M}_{ij} \in \mathbb{R}^{k \times k}$ between the two embedding spaces. The following loss is minimized

$$S^A_{L_i,L_j} = \sum_{(x_i,x_j)\in I(L_i,L_j)} \|\mathbf{M}_{ij}\mathbf{x}_i - \mathbf{x}_j\|_2 \,,$$

in which $I(L_i, L_j) = I_E(L_i, L_j) \cup I_W(L_i, L_j)$, and the word seed lexicon I_W is considered additional supervision data that are optionally provided. Each $\mathbf{x}_i$ ($\mathbf{x}_j$) denotes a fixed representation of either an entity or a lexeme of L_i (L_j).

Starting from a small amount of seed alignment in $I(L_i, L_j)$, `JEANS` conducts an iterative self-learning process to exploit more alignment labels for both entities and lexemes to improve the learning of $\mathbf{M}_{ij}$. In each iteration, we follow Conneau et al. (2018) to induce a Procrustes solution for $\mathbf{M}_{ij}$. To propose new alignment labels, the self-learning technique in `JEANS` deploys a mutual nearest neighbor (NN) constraint, which requires a suggested pair of matched items to appear in the NN of each other. More specifically, define $\mathcal{N}^K_{L_i}(\mathbf{x})$ as the K-NN of vector $\mathbf{x}$ in the embedding space of L_i, this constraint requires a proposed match (x_i, x_j) to be inserted into I only if $\mathbf{M}_{ij}\mathbf{x}_i$ is in $\mathcal{N}^1_{L_j}(\mathbf{x}_j)$, and $\mathbf{x}_j$ mutually appears in $\mathcal{N}^1_{L_j}(\mathbf{M}_{ij}\mathbf{x}_i)$. Besides, we also require (x_i, x_j) to be of the same type, i.e. both being entities or being lexemes. Particularly, we only select entities that have not been aligned in I to form the newly-proposed (x_i, x_j). This respects the 1-to-1 matching constraint of entities being defined at the beginning of this section, and effectively reduces the candidate space after each iteration of self-learning. Meanwhile, 1-to-1 matching is not required for lexemes. To mitigate hubness, we also follow Conneau et al. (2018) to employ the Cross-domain Similarity Local Scaling (CSLS) measure.

After the iteration, the newly proposed alignment labels are inserted to I to enhance the learning at the next iteration. The iterative self-learning is stopped once the number of proposed entity alignment in an iteration is below certain quantity (e.g. 1% of $|E_{L_i}|$). With more and more matched entities and lexemes being exploited within each iteration, a better $\mathbf{M}_{ij}$ is induced, whereas the lexical alignment naturally serve as incidental supervision signals for entity alignment.

After the alignment learning process, given a query $(e_i, ?e_j)$ to find the counterpart entity of $e_i \in E_{L_i}$ from E_{L_j}, the answer e_j is predicted as the 1-NN entity after applying $\mathbf{M}_{ij}$ to transform $\mathbf{e}_i$, denoted $\{e_j\} = \mathcal{N}^1_{E_{L_i}}(\mathbf{M}_{ij}\mathbf{e}_i)$. The inference phase by default also adopts CSLS as the distance measure, which is consistent with the default setting of recent works (Sun et al., 2019b, 2020a).

4 Experiment

In this section, we evaluate `JEANS` on two benchmark datasets for cross-lingual entity alignment, and compare against a wide selection of recent baseline methods. We also provide detailed ablation study on model components of `JEANS`.

4.1 Experimental Settings

Datasets. Experiments are conducted on DBP15k (Sun et al., 2017) and WK3l60k (Chen et al., 2018) that are widely used benchmarks on the studied task. DBP15k contains four language-specific KGs that are respectively extracted from English (En), Chinese (Zh), French (Fr) and Japanese (Ja) DBpedia (Lehmann et al.,

Settings	DBP15k$_{En-Fr}$			DBP15k$_{En-Zh}$			DBP15k$_{En-Ja}$			WK3l60k$_{En-Fr}$			WK3l60k$_{En-De}$		
Metrics	$H@1$	$H@10$	MRR	$H@1$	$H@10$	MRR	$H@1$	$H@10$	MRR	$H@1$	$H@5$	MRR	$H@1$	$H@5$	MRR
MTransE (Chen et al., 2017a)†‡	0.224	0.556	0.335	0.308	0.614	0.364	0.279	0.575	0.349	0.140	0.203	0.177	0.034	0.101	0.072
GCN-Align (Wang et al., 2018)‡	0.373	0.745	0.532	0.413	0.744	0.549	0.399	0.745	0.546	0.215	0.378	0.293	0.138	0.246	0.190
AlignE (Sun et al., 2018)†	0.481	0.824	0.599	0.472	0.792	0.581	0.448	0.789	0.563	−−	−−	−−	−−	−−	−−
GCN-JE (Wu et al., 2019b)	0.483	0.778	−−	0.459	0.729	−−	0.466	0.746	−−	−−	−−	−−	−−	−−	−−
RotatE (Sun et al., 2019c)†	0.345	0.738	0.476	0.485	0.788	0.589	0.442	0.761	0.550	−−	−−	−−	−−	−−	−−
KECG (Li et al., 2019a)	0.486	0.851	0.610	0.478	0.835	0.598	0.490	0.844	0.610	−−	−−	−−	−−	−−	−−
MuGCN (Cao et al., 2019)†	0.495	0.870	0.621	0.494	0.844	0.611	0.501	0.857	0.621	−−	−−	−−	−−	−−	−−
RSN (Guo et al., 2019)†	0.516	0.768	0.605	0.508	0.745	0.591	0.507	0.737	0.590	−−	−−	−−	−−	−−	−−
GMN (Xu et al., 2019)	0.596	0.876	0.679	0.433	0.681	0.479	0.465	0.728	0.580	−−	−−	−−	−−	−−	−−
AliNet (Sun et al., 2020a)†	0.552	0.852	0.657	0.539	0.826	0.628	0.549	0.831	0.645	−−	−−	−−	−−	−−	−−
JAPE (Sun et al., 2017)†‡	0.324	0.667	0.430	0.412	0.745	0.490	0.363	0.685	0.476	0.169	0.354	0.271	0.147	0.239	0.192
SEA (Pei et al., 2019a)†	0.400	0.797	0.533	0.424	0.796	0.548	0.385	0.783	0.518	−−	−−	−−	−−	−−	−−
HMAN (Yang et al., 2019)	0.543	0.867	−−	0.537	0.834	−−	0.565	0.866	−−	−−	−−	−−	−−	−−	−−
BootEA (Sun et al., 2018)†‡	0.653	0.874	0.731	0.629	0.847	0.703	0.622	0.854	0.701	0.333	0.511	0.425	0.233	0.393	0.316
KDCoE (Chen et al., 2018)	−−	−−	−−	−−	−−	−−	−−	−−	−−	**0.483**	**0.569**	0.496	0.335	0.380	0.339
MMR (Shi and Xiao, 2019)	0.635	0.878	−−	0.647	0.858	−−	0.623	0.847	−−	−−	−−	−−	−−	−−	−−
NAEA (Zhu et al., 2019)	0.673	0.894	0.752	0.650	0.867	0.720	0.641	0.873	0.718	−−	−−	−−	−−	−−	−−
OTEA (Pei et al., 2019b)‡	−−	−−	−−	−−	−−	−−	−−	−−	−−	0.361	0.541	0.447	0.270	0.440	0.352
`JEANS`-SFM	0.766	0.939	0.814	0.713	0.885	0.773	0.723	0.913	0.793	0.463	0.558	**0.538**	**0.337**	**0.450**	**0.412**
`JEANS`-EDL	**0.769**	**0.940**	**0.827**	**0.719**	**0.895**	**0.791**	**0.737**	**0.914**	**0.798**	0.451	0.544	0.529	0.312	0.431	0.390

Table 1: Entity alignment results. Baselines are separated in accord with the three groups described in Section 4.1. † indicates results obtained from (Sun et al., 2020a), and ‡ indicates those from (Pei et al., 2019b). Results of KECG, GCN-JE, MMR, HMAN, KDCoE and NAEA are from original papers. Hyphens denote not available. MRR were not reported by GCN-JE, MMR and HMAN. Top results (incl. w/ and w/o seed lexicon) are boldfaced. Note that results by GCN-JE, GMN and HMAN are reported only for the versions where the extra cross-lingual alignment information (such as machine translation) is removed, so as to conduct fair comparison with all the rest models that are trained using only the alignment labels in the benchmark training sets.

2015), each of which contains around 65k-106k entities. Three sets of 15k alignment labels are constructed to align entities between each of the other three languages and English. WK3l60k contains larger KGs with around 57k to 65k entities in En, Fr and German (De) KGs, and around 55k reference entity alignment for En-Fr and En-De settings. Dataset statistics are given in Appendix §A.2 (Chen et al., 2021).

We also use the text of Wikipedia dumps (dated Jan 01, 2019) in the five participating languages in training. For Chinese and Japanese corpora thereof, we obtain the segmented versions respectively from PKUSEG (Luo et al., 2019) and MeCab (Kudo, 2006).

Baseline methods. We compare with a wide selection of recent approaches for entity alignment on multilingual KGs. The baseline methods include (i) those employing different structure embedding techniques, namely MTransE (Chen et al., 2017a), GCN-Align (Wang et al., 2018), AlignE (Sun et al., 2018), GCN-JE (Wu et al., 2019b), KECG (Li et al., 2019a), MuGCN (Cao et al., 2019), RotatE (Sun et al., 2019c), RSN (Guo et al., 2019) and AliNet (Sun et al., 2020a); (ii) methods that incorporate auxiliary information of entities, namely JAPE (Sun et al., 2017), SEA (Pei et al., 2019a), GMN (Xu et al., 2019) and HMAN (Yang et al., 2019); (iii) semi-supervised alignment learning methods, including BootEA (Sun et al., 2018), KDCoE (Chen et al., 2018), MMR (Shi and Xiao, 2019), NAEA (Zhu et al., 2019) and OTEA (Pei et al., 2019b). Descriptions of these methods are given in Appendix §A.1 (Chen et al., 2021).

Note that some works have allowed to incorporate extra cross-lingual signals such as machine translation in training, or using pre-aligned word embeddings to delimit candidate spaces (Wu et al., 2019a,b; Xu et al., 2019). For example, Wu et al. (2019a,b) used Google Translate to translate surface forms of entities in all other languages to English, and initialize the entity embeddings in their model with pre-trained word embedding of translated entity names. Results for these models are reported for the versions where the extra cross-lingual alignment information is removed so as to conduct fair comparison with all the rest models that are trained from scratch and using the same alignment labels in the benchmark datasets. This also necessarily prevents potential leakage of testing data to training (Liu et al., 2020), considering that training a comprehensive NMT system may have subsumed many of the testing data in the entity alignment benchmarks.

Evaluation protocols. The use of the datasets are consistent with previous studies of the baseline methods. On each language pair in DBP15k, around 30% of seed alignment is used for training, the rest for testing. On WK3l60k, 20% of seed alignment on En-Fr and En-De settings is respectively used for training. Following the convention, we calculate several ranking metrics on test cases, including the accuracy $H@1$, the proportion of cases that are ranked no larger than p $H@p$, and mean reciprocal rank MRR. Note that to align with the results in previous studies (Sun et al., 2020a; Pei et al., 2019b), p is set to 10 on DBP15k and 5 on WK3l60k. All metrics are preferred higher to indicate better performance.

Model Configurations. We use AMSGrad (Reddi et al., 2018) to optimize the training losses of the embedding learning process, for which we set the learning rate α to 0.001, the exponential decay rates β_1 and β_2 to 0.9 and 0.999, and batch sizes to 512 for both S_L^K and S_L^T. Trainable parameters are initialized using Xavier initialization (Glorot and Bengio, 2010). The dimension k is set to 300, which is often used for bilingual word embedding models trained on Wikipedia corpora (Conneau et al., 2018; Gouws et al., 2015), considering that the vocabulary sizes and training data density here are relatively close to those models. The number of GCN layers is set to 2. We set negative sample sizes of triples and text contexts to 5, the text context width to be 10 and the bias b in S_L^K to be 2. More implementation details are in Appendix §A.3 (Chen et al., 2021). Specifically, we evaluate variants of `JEANS` by adjusting two technical details. First, for the grounding process, aside from the simple surface form matching (marked with SFM), we also explore with the off-the-shelf Wikification-based EDL model (Upadhyay et al., 2018, marked with EDL). A grounding performance estimation is given in §4.4. In addition, we consider both CSLS and l_2 in inference.

4.2 Results

We report the entity alignment results in Table 1.

Considering the baseline results on DBP15k, we can see that the simplest variant of `JEANS` using SFM-based grounding has consistently outperformed all baselines on three cross-lingual settings. Particularly, it leads to 17.0-17.4% of absolute improvement in $H@1$ over the best structure-based baseline, 14.0-22.3% over the best entity profile based one, and 6.30-9.30% over the best semi-supervised one. This shows that while `JEANS` preserves the key merit of a semi-supervised entity alignment method, and effectively enhances the alignment of KGs by exploiting incidental supervision signals from unaligned text corpora. Considering different grounding techniques, we observe that SFM variants often perform closely to EDL ones. This indicates that simple SFM is enough to combine KG and text corpora for `JEANS`'s embedding learning without EDL-related resources. The results on Wk3l60k generally exhibit similar observations. In comparison to KDCoE that leverages strong but expensive supervision data of entity descriptions in co-training, `JEANS` offers comparable performance based on very accessible resources.

Setting	DBP15k$_{En-Fr}$			DBP15k$_{En-Ja}$		
Metrics	$H@1$	$H@10$	MRR	$H@1$	$H@10$	MRR
`JEANS`-SFM	0.766	0.939	0.814	0.723	0.913	0.793
—w/o Self-learning	0.628	0.845	0.720	0.622	0.835	0.728
—w/o GCN	0.742	0.913	0.809	0.709	0.905	0.789
—w/o Text	0.725	0.891	0.786	0.681	0.857	0.761
—w/o KG	0.699	0.872	0.771	0.635	0.819	0.706
—w/o CSLS	0.697	0.905	0.762	0.687	0.893	0.768
—w/ seed lexicons	0.788	0.947	0.848	0.738	0.931	0.803

Table 2: The ablation study results for components of `JEANS` based on DBP15k$_{En-Fr}$ and DBP15k$_{En-Ja}$. Note that the additional seed lexicon is not used in the main experiment, and is not obligatory. The last row in this table is only to show the effectiveness of leveraging available supervision data on lexemes.

Language	SFM		EDL	
Estimation	Coverage	Avg match	Coverage	Avg match
En	0.982	1,268	0.933	1,367
Fr	0.987	295	0.926	929
Zh	0.855	141	0.774	348
Ja	0.982	159	0.797	881
De	0.981	297	0.951	1,092

Table 3: Estimated vocabulary coverage and average match per entity on each of the five language-specific Wikipedia corpora.

In general, the experiments here show that `JEANS` promisingly improves SOTA performance for entity alignment, with only the need for unparalleled free text and no need for additional labels.

4.3 Ablation Study

In Table 2 we report an ablation study for `JEANS`-SFM based on DBP15k, so as to understand the importance of each incorporated technique.

From the results, we observe that self-learning is the most important factor. The removal of it can lead to a drop of 10.1-13.8% in $H@1$, as well as drastic drop of other metrics. This also explains why semi-supervised baselines (group 3) typically perform better than others. However, even with self-learning, the removal of text can lead to $H@1$ drop of 2.4% on En-Fr and 4.2% on En-Ja. This shows that context information `JEANS` retrieves from free text effectively infers the match of entities. On the other hand, the structure encoding of KGs is more important than textual contexts, as it causes higher performance drops of 6.7-8.8% in $H@1$ by removing KGs. Note that the model without KG learns entity embeddings solely based on free text. Its results show that context information from text alone can provide a strong starting point from which incorporating KGs can further enhance its performance. Employing GCN leads to relatively slight performance gain, as joint learning the relation model and the language model can satisfyingly capture entity proximity. Changing the distance metric to l_2 also leads to 3.6-6.9% of decrease in $H@1$. This shows CSLS's ability to handle hubness and isolation is also important for similarity inference in the dense embedding space for the metric words and entities. Hence, this metric is also recommended by recent work (Sun et al., 2020a, 2019b; Zhang et al., 2019). In addition, if we introduce additional 5k seed lexicon (with only word alignment information, not including any entity alignment) provided by Conneau et al. (2018) for each language pair, it leads to additional improvement of 1.5-2.2% in $H@1$. This shows that `JEANS` effectively leverages available supervision data on lexemes to further enhance entity alignment, although it is not obligatory.

4.4 Grounding Performance Estimation

Due to the lack of ground truths on unlabeled text, it is hard to estimate the precision of entity grounding by the two types of (noisy) grounding techniques. However, as the requirement of the grounding process is to simply connect two data modalities for training the embeddings, we may encourage a technique that handles enough entity mentions and offer a higher coverage on entity vocabularies. Accordingly, the estimations of these two factors for the two techniques are reported in Table 3. As we can observe that, without considering disambiguation, SFM can overall cover higher proportions of the entity vocabularies, while pretrained EDL generally discovers more entity mentions for each entity. However, both techniques are sufficient to support the noisy grounding process and combine two data modalities for embedding learning and alignment induction.

5 Conclusion

This paper introduces `JEANS` for entity alignment. Different from previous methods that leverage only internal information of KGs, `JEANS` extends the learning on any text corpora that may contain the KG entities. For each language, a noisy grounding process first connects both data modalities, followed by an embedding learning process coupling GCN with relational modeling, and an self-learning based alignment process. Without introducing additional labeled data, `JEANS` offers significantly improved performance over SOTA models on benchmarks. Hence, it shows the effectiveness and feasibility of exploiting incidental supervision from free text for entity alignment.

For future work, aside from experimenting with other embedding learning techniques for KGs and text, we plan to extend `JEANS` to learn associations on KGs with different specificity (Hao et al., 2019). We also seek to extend the representation scheme in hyperbolic spaces (Nickel and Kiela, 2017; Chen and Quirk, 2019) along with the incorporation of hyperbolic lexical embedding techniques (Tifrea et al., 2018), aiming at better capturing the associations for hierarchical ontologies.

Acknowledgement

We appreciate the anonymous reviewers for their insightful comments and suggestions. Also, we would like thank Jennifer Sheffield, Yi Zhang and other members of the UPenn Cognitive Computation Group for giving suggestions that improved the manuscript.

This research is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the BETTER Program, and by Contract FA8750-19-2-1004 with the US Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

Satya Almasian, Andreas Spitz, and Michael Gertz. 2019. Word embeddings for entity-annotated texts. In *Proceedings of the European Conference on Information Retrieval (ECIR)*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Jens Bleiholder and Felix Naumann. 2009. Data fusion. *ACM Computing Surveys*, 41(1):1–41.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics (TACL)*, 5.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD)*.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems (NIPS)*.

Volha Bryl and Christian Bizer. 2014. Learning conflict resolution strategies for cross-language wikipedia data fusion. In *Proceedings of the World Wide Web Conference (WWW)*, pages 1129–1134.

Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. 2019. Multi-channel graph neural network for entity alignment. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*, pages 1452–1461.

Jiaao Chen, Jianshu Chen, and Zhou Yu. 2019. Incorporating structured commonsense knowledge in story completion. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Muhao Chen and Chris Quirk. 2019. Embedding edge-attributed relational hierarchies. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

Muhao Chen, Weijia Shi, Ben Zhou, and Dan Roth. 2021. Cross-lingual entity alignment with incidental supervision. *arXiv preprint arXiv:2005.00171v2 (with Appendices)*.

Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017a. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Muhao Chen, Tao Zhou, et al. 2017b. Multi-graph affinity embeddings for multilingual knowledge graphs. In *Automated Knowledge Base Construction (AKBC)*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sarang Dharmapurikar, Praveen Krishnamurthy, and David E Taylor. 2006. Longest prefix matching using bloom filters. *IEEE/ACM Transactions on Networking*, 14(2):397–409.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Stephan Gouws, Yoshua Bengio, et al. 2015. Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representationlearning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Hangfeng He, Qiang Ning, and Dan Roth. 2020. Quase: Question-answer driven sentence encoding. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Bo-June Paul Hsu and Giuseppe Ottaviano. 2013. Space-efficient data structures for top-k completion. In *Proceedings of the World Wide Web Conference (WWW)*.

Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Yujiao Zhou, and Ian Horrocks. 2012. Large-scale interactive ontology matching: Algorithms and implementation. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.

Daniel Khashabi, Mark Sammons, Ben Zhou, Tom Redman, Christos Christodoulopoulos, Vivek Srikumar, Nickolas Rizzolo, Lev Ratinov, Guanheng Luo, Quang Do, et al. 2018. Cogcompnlp: Your swiss army knife for nlp. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Taku Kudo. 2006. Mecab: Yet another part-of-speech and morphological analyzer. *http://mecab.sourceforge.jp*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. 2019a. Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Shiyang Li, Jianshu Chen, and Dian Yu. 2019b. Teaching pretrained models with commonsense reasoning: A preliminary kb-based approach. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, and Tat-Seng Chua. 2020. Exploring and evaluating attributes, values, and structure for entity alignment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6355–6364.

Ruixuan Luo, Jingjing Xu, Yi Zhang, Xuancheng Ren, and Xu Sun. 2019. Pkuseg: A toolkit for multi-domain chinese word segmentation. *CoRR*, abs/1906.11455.

Farzaneh Mahdisoltani, Joanna Biega, et al. 2015. Yago3: A knowledge base from multilingual Wikipedias. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Tomas Mikolov, Kai Chen, et al. 2013. Efficient estimation of word representations in vector space. *International Conference on Learning Representations (ICLR)*.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, B Yang, J Betteridge, A Carlson, B Dalvi, M Gardner, B Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*.

Diego Moussallem, Matthias Wauer, and Axel-Cyrille Ngonga Ngomo. 2018. Machine translation using semantic web technologies: A survey. *Journal of Web Semantics*, 51:1–19.

Denis Newman-Griffis, Albert M Lai, and Eric Fosler-Lussier. 2018. Jointly embedding entities and text with distant supervision. In *Proceedings of the Workshop on Representation Learning for NLP*.

Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems (NIPS)*.

Shichao Pei, Lu Yu, Robert Hoehndorf, et al. 2019a. semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *Proceedings of the Web Confererence (WWW)*.

Shichao Pei, Lu Yu, and Xiangliang Zhang. 2019b. Improving cross-lingual entity alignment via optimal transport. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Jeffrey Pennington, Richard Socher, et al. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Jay Pujara, Eriq Augustine, and Lise Getoor. 2017. Sparsity and noise: Where knowledge graph embeddings fall short. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of adam and beyond. In *International Conference on Learning Representations (ICLR)*.

Nils Rethmeier, Marc Hübner, and Leonhard Hennig. 2018. Learning comment controversy prediction in web discussions using incidentally supervised multi-task cnns. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

Dan Roth. 2017. Incidental supervision: Moving beyond supervised learning. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2017. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*.

Peter H Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10.

Weijia Shi, Muhao Chen, Pei Zhou, and Kai-Wei Chang. 2019. Retrofitting contextualized word embeddings with paraphrases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Xiaofei Shi and Yanghua Xiao. 2019. Modeling multi-mapping relations for precise cross-lingual entity alignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Pavel Shvaiko and Jérôme Euzenat. 2011. Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engeering*, 25(1):158–176.

Avirup Sil, Heng Ji, Dan Roth, and Silviu Cucerzan. 2018. Multi-lingual entity discovery and linking. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Yangqiu Song and Dan Roth. 2015. Unsupervised sparse vector densification for short text similarity. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Fabian M Suchanek, Serge Abiteboul, et al. 2011. Paris: Probabilistic alignment of relations, instances, and schema. *Proceedings of the VLDB Endowment (PVLDB)*, 5(3).

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019a. Dream: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics (TACL)*, 7:217–231.

Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *Proceedings of the International Semantic Web Conference (ISWC)*.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020a. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Zequn Sun, Jiacheng Huang Wang, Wei Hu, Muhao Chen, and Yuzhong Qu. 2019b. Transedge: Translating relation-contextualized embeddings for knowledge graphs. In *Proceedings of the International Semantic Web Conference (ISWC)*.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020b. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proceedings of the VLDB Endowment (PVLDB)*, 13.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019c. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations (ICLR)*.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. 2018. Poincare glove: Hyperbolic word embeddings. In *International Conference on Learning Representations*.

Kristina Toutanova, Danqi Chen, et al. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Bayu Distiawan Trsedya, Jianzhong Qi, and Rui Zhang. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Shyam Upadhyay, Nitish Gupta, and Dan Roth. 2018. Joint multilingual supervision for cross-lingual entity linking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014a. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Zhen Wang, Jianwen Zhang, et al. 2014b. Knowledge graph and text jointly embedding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Derry Wijaya, Partha Pratim Talukdar, et al. 2013. Pidgin: ontology alignment using web text as interlingua. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019a. Relation-aware entity alignment for heterogeneous knowledge graphs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2019b. Jointly learning entity and relation representations for entity alignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Kun Xu, Liwei Wang, Mo Yu, Yansong Feng, Yan Song, Zhiguo Wang, and Dong Yu. 2019. Cross-lingual knowledge graph alignment via graph matching neural network. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics (TACL)*, 5:397–411.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. *International Conference on Learning Representations (ICLR)*.

Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. 2019. Aligning cross-lingual entities with multi-aspect information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Kai Yang, Shaoqin Liu, Junfeng Zhao, Yasha Wang, and Bing Xie. 2020. Cotsae: Co-training of structure and attribute embeddings for entity alignment. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Jinyoung Yeo, Geungyu Wang, Hyunsouk Cho, Seungtaek Choi, and Seung-won Hwang. 2018. Machine-translated knowledge transfer for commonsense causal reasoning. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view knowledge graph embedding for entity alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Huaping Zhong, Jianwen Zhang, et al. 2015. Aligning knowledge and text embeddings by entity descriptions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Qiannan Zhu, Xiaofei Zhou, Jia Wu, Jianlong Tan, and Li Guo. 2019. Neighborhood-aware attentional representation for multilingual knowledge graphs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Will Y Zou, Richard Socher, et al. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Query Generation for Multimodal Documents

Kyungho Kim[1], **Kyungjae Lee**[1], **Seung-won Hwang**[2]*,
Young-In Song[3] and **Seungwook Lee**[3]
[1]Yonsei University, [2]Seoul National University, [3]NAVER
[1]{ggdg12, lkj0509}@yonsei.ac.kr, [2]seungwon.hwang@gmail.com,
[3]{song.youngin, swook.lee}@navercorp.com

Abstract

This paper studies the problem of generating likely queries for multimodal documents with images. Our application scenario is enabling efficient "first-stage retrieval" of relevant documents, by attaching generated queries to documents before indexing. We can then index this expanded text to efficiently narrow down to candidate matches using inverted index, so that expensive reranking can follow. Our evaluation results show that our proposed multimodal representation meaningfully improves relevance ranking. More importantly, our framework can achieve the state of the art in the first-stage retrieval scenarios.

1 Introduction

As more documents on the web are generated and consumed by mobile devices with cameras, documents are often multimodal, containing information in both text and image modalities. This poses a new challenge of finding relevance documents across modalities. More formally, the relevance of document, consisting of text t and image i, to the given query keywords q, should be modeled as a trimodal function $f(q, t, i)$, rather than a simple lexical match between q and t (or, **BM25** baseline) assuming the semantics of image i is fully represented by the surrounding text (or, **paired-text** assumption).

Prior research observes that paired-text assumption is often violated (Henning and Ewerth, 2017) – for example, some semantics can be better captured in image modality, and may not (or, cannot) be described in text. Meanwhile, BM25 baseline (Robertson et al., 1994) would fail to serve queries for such semantics.

To overcome this limitation of BM25, one may model relevance from query as a trimodal function $f(q, t, i)$ (Nian et al., 2017; Kordan and Kotov, 2018) instead, but they require runtime invocation of f for the given query q with all potential document matches. This incurs a prohibitive runtime overhead, unacceptable for search engines finding results online. A common practice is to use a cheap BM25 ranking as a "first-stage retrieval", efficiently supported by inverted index, to quickly narrow down to a few candidate documents, then evaluate $f(q, t, i)$. However, due to simple nature of BM25 using exact term matching, a document will be missed, if the query term is absent in t, even though it semantically matches image or another term in the text.

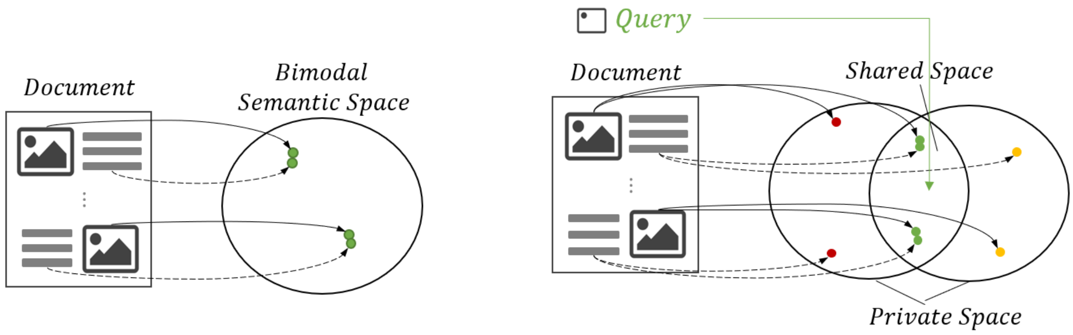
Our contribution is to keep first-stage retrieval as efficient as BM25, but enable multimodal semantic matching, using Query Generation **(QG)** before indexing. More specifically, we generate a likely query q from a joint modeling of t and i, to create an expanded text $t' = q \cup t$ such that (q, t') pair has more lexical overlaps, or better paired than (q, t), for first-phase retrieval. Specifically, we train a sequence-to-sequence model, such that given the representation of multimodal document, this model generates possible queries that users may ask to retrieve such document. This is analogous to doc2query (Nogueira et al., 2019) approach used for first-stage retrieval of textual relevance ranking, though this model, dealing with text modality only, cannot apply to our problem of retrieving multimodal documents.

For such multimodal representation for **QG**, a naive baseline is bimodal representation shown in Figure 1a: We may assume (t, i), even when lexical overlaps are low, is semantically paired in the embedding space. Note this is a relaxed version of paired-text assumption. Given this relaxed assumption, common architecture of bimodal representa-

---

[*] correspond to seungwon.hwang@gmail.com

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 659–668
April 19 - 23, 2021. ©2021 Association for Computational Linguistics



(*a*) (*b*)

Figure 1: (a) bimodal and (b) trimodal representation for QG, query is used for disentangling shared and private space and relaxing paired-text assumption of bimodal representation.

tion for QG consists of the encoders for image i and text t. Each generates vector representation, which is later fused into a multimodal space, then decoded into a textual caption. Specifically, we consider two strong baselines: (a) cross-modal representation, pretrained from a large-scale paired corpus of image and caption, such as LXMERT (Tan and Bansal, 2019), ViLBERT (Lu et al., 2019), and VisualBERT (Li et al., 2019), finetuned for our task, and (b) memory network structure (Park et al., 2017). Based on these baselines, we design Bimodal QG combining the advantages of the two as a strong baseline. Then, we extend into Trimodal QG leveraging text, image and query (q, t, i).

Alternatively, we may further relax paired-text assumption and propose Trimodal QG in Figure 1b: (t, i) can be partially paired, where some semantics is conveyed in one modality. To deal with that challenge, the query given at training, helps "disentangle" shared and private semantics as additional loss terms. Another role of query is improving image representation, to de-emphasize semantics not discussed in either text or query.

In summary, our contributions are as follows:

- We study QG for multimodal documents, as an enabler for efficient first-stage ranking.
- We build a multi-task model, for query generation and representation learning, to generate effective queries for offline indexing.
- We improve the QG model by considering query as third modality in order to work well without paired-text assumption.
- We validated that our model outperforms all baselines in both public dataset and real-life web search query logs and quality annotation for multimodal documents.

2 Related Work

Our work is closely related to the following three areas of research.

2.1 Web search with images

Most efficient way to treat multimodal document ranking has been making paired-text assumption (Coelho et al., 2004; Azilawati and Meriam, 2008), such that simply matching q with t is sufficient. Our work is as efficient, by incurring no additional runtime overhead, but does not make such assumption. Alternatively, Rodríguez-Vaamonde et al. (2015) adds a reranking phase, checking if the images are relevant to the query, supervised by whether the given image is correlated with clicks.

Our distinction: We do not build on paired-text assumption, and can be viewed as generating a better-paired document by adding likely queries.

2.2 Image captioning

Another closely related work is the task of generating textual captions to the given query. As overviewed in Section 1, state-of-the-art models include bimodal joint representation of image i and text t (Kiros et al., 2014b; You et al., 2016; Park et al., 2017). Alternatively, such joint models can also be transferred from pretrained models, such as LXMERT (Tan and Bansal, 2019), ViLBERT (Lu et al., 2019), and VisualBERT (Li et al., 2019). Section 3 will compare and contrast these two approaches, and discuss why these models are

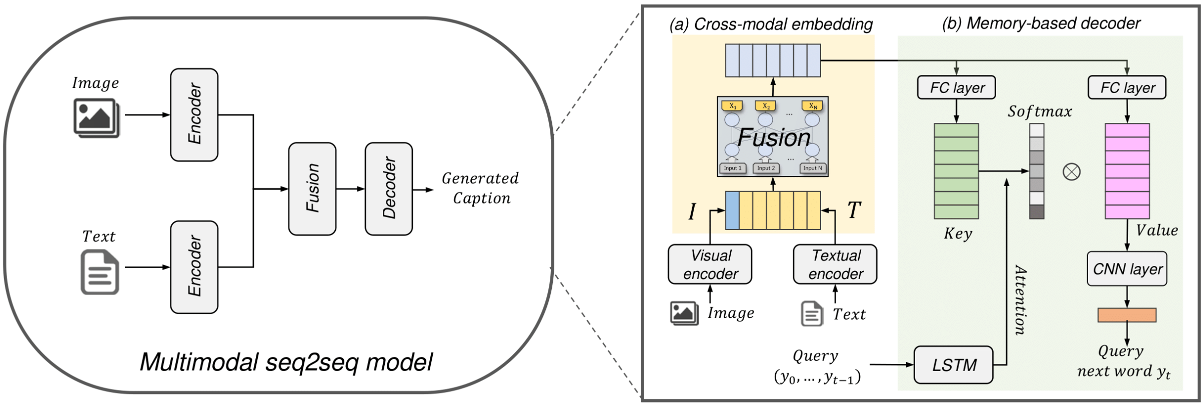


Figure 2: Proposed bimodal QG baseline, combining (a) cross-modal and (b) memory-based state-of-the-arts

limited for our task setting. (Jeon et al., 2003) is a non-neural model trained to annotate images with textual description, though requiring expensive supervision of segmented image with term.

Our distinction: We propose and validate trimodal joint representation for higher-quality captioning. Meanwhile, we do not require segment-level annotation, though our query-guided trimodal image representation naturally emphasizes important segments.

2.3 QG for first-stage retrieval

QG for first-stage text retrieval was pioneered by doc2query (Nogueira et al., 2019), generating likely queries for the document for indexing purposes in text modality. Our work can be viewed as query generation for multimodal documents: For each document, the task is to predict a set of likely queries. We train a sequence-to-sequence transformer using a data set of (query, relevant document) pairs. Alternatively, inverted index can be built for a latent term (Zamani et al., 2018), though it cannot be human-interpreted or reweighed. In contrast, we focus on inverted index on actual terms, as it is human interpretable and combines more naturally with legacy tf-idf ranker and reweighting module.

Our distinction: We validate the effectiveness of trimodal QG over bimodal state-of-the-art methods.

3 Bimodal Baselines: LXMERT and Memory-Based Generator

This section compares and contrasts two bimodal baselines: LXMERT [1] and Memory-Based Generator encode text t and image i into vector representations (Section 3.1), then the two are aggregated into a multimodal representation (Section 3.2), such that this vector can feed a decoder to generate a text sequence (Section 3.3). Specifically, we build **Bimodal QG** baseline, combining LXMERT representation and Memory-Based Generator decoding. Figure 2 shows overall architecture of our Bimodal QG baseline.

3.1 Text and image encoder

LXMERT and Memory-Based Generator generate text and image vectors, using transformer and memory network structure respectively. Both can be explained as **key** memory, aggregating **value** memory representation with proper self-attention, denoted as key and val, respectively, following the conventions of prior literature (Sukhbaatar et al., 2015).

Formally, we encode textual context words $C = \{w_1, w_2, ..., w_j\}$ obtained by concatenating the $n-$dimensional word embedding ($w \in R^n$) of the

---

[1] Out of cross-modal representations discussed in Section 1, we empirically found LXMERT performs the best in our problem setting and thus adopt it as a baseline.

top-j words with the highest TF-IDF weights [2].

$$\begin{aligned} T_i^{key} &= \text{ReLU}(W_1 w_i + b_1), \;\; i \in [1, j] \\ T_i^{val} &= \text{ReLU}(W_2 w_i + b_2), \;\; i \in [1, j] \\ T^{key} &= [T_1^{key}; ...; T_j^{key}] \\ T^{val} &= [T_1^{val}; ...; T_j^{val}] \end{aligned}$$

where W_1, $W_2 \in R^{m \times n}$, $b_1, b_2 \in R^m$ are trainable linear transformation parameters where m is the dimension of memory. When parameter, such as T or I, is denoted without superscript (key or val), it refers both key and val vectors.

Similarly, an image input $U \in R^{2048}$ is generated from pool5 feature vector of Resnet-101 CNN encoder, which is similarly embedded into:

$$\begin{aligned} I^{key} &= \text{ReLU}(W_3 U + b_3) \\ I^{val} &= \text{ReLU}(W_4 U + b_4) \end{aligned}$$

where $W_3, W_4 \in R^{m \times 2048}$ and $b_3, b_4 \in R^m$ are trainable matrices for tuning on given dataset. Final image embedding is generated with key and value vector, following the convention of attention network (Kiros et al., 2014a). Note only the representation of text and image is used at this point, and using query representation will be discussed in Section 4. Also, LXMERT extract image features by using Faster-RCNN (Ren et al., 2015) and transformer structure, which is pre-trained on large dataset combined by MSCOCO (Lin et al., 2014) and Visual Genome (Krishna et al., 2016). Therefore, both methods can be applicable on general images.

3.2 Multimodal fusion

The goal of multimodal fusion module in Figure 2 is to get text and image representations as input, and create a joint representation as output. When paired-text assumption holds (Figure 1a), this can be achieved by simply concatenating or adding two input modalities, and the future layers will be tuned for a proper alignment of the two. However, such concatenation is less effective when paired-text assumption does not hold as in Figure 1b.

One solution is transfer learning from pretrained joint representation model trained from a large-scale paired resources, such as LXMERT (Tan and Bansal, 2019). LXMERT is a transformer-based architecture for cross-modal representation learning, for predicting masked words from the text modality, and vice versa. This auxiliary task, known as masked cross-modality language model, helps building connections across modalities. In our problem setting, this option can be considered for public English dataset, as pre-trained LXMERT with a large scale paired training resources is readily available to generate a joint representation M_{lxm}, to replace simple concatenated fusion embedding M_{fusion} ($[I; T]$).

3.3 Bimodal QG: Joint text decoding for query generation

We now observe the two baselines: LXMERT, focuses on the problem of joint representation, but does not consider a decoder of generating a query sequence from such representation (Figure 2a). Meanwhile, Memory-Based Generator has the advantage of tightly coupling the key-value encoder and CNN decoder, by concatenating the representations of all modalities, co-attended based on the query keywords generated thus far, as illustrated in Figure 2b. This multimodal vector is calculated in each time step and used to decode the next word, which is an effective decoder design adopted for our model:

$$M_{total} = [I; T; Q]$$

With this joint representation, query generation is predicting the output probability of the next word among all vocabularies, by a convolution neural network, denoted as CNN in Figure 2.

For combining with the strength of LXMERT, we can simply replace the cross modal embedding vector in Figure 2 by M_{lxm} for bimodal QG. Alternatively, for ablation purpose, M_{lxm} can directly decoded without memory-based decoder, which we denote as LXMERT QG. Our final loss of bimodal captioning is a seq2seq loss.

$$L_{seq} = -\sum_{t=0}^{l} log(P(y_t | y_{<t}, I, T))$$

where t is the time step and l is the length of caption.

4 Trimodal QG: Query-aware representation

Our proposed bimodal QG partially contributes to relax paired-text assumption, but neither image

---

[2] C can be a subset of arbitrary size from t, which we empirically tune to $j = 30$.

and text representation is aware of queries. We argue that, queries carry rich semantics and contribute significantly to to relax paired-text assumption, specified as two key contributions C1 and C2 below.

- C1: We use query to improve image and text representation to disentangle into shared/private semantics where q matches the shared semantics.
- C2: As query generation is better trained when t and i are paired, we revise image representation to enhance pairedness with given query.

Motivated, we propose two new loss functions L_1 and L_2, addressing C1 and C2 respectively.

4.1 Query-aware relevance

For addressing **C1**, we model "private" parts of image and text, denoted as P_i and P_t, to relax the paired-text assumption. Our goal is to build joint representation S, aligning only the shared part of image and text, with query q.

$$S = W_5[I^{key}; T^{key}] + b_5$$
$$P_i = W_6 I^{key} + b_6$$
$$P_t = W_7 T^{key} + b_7$$

where $W_5, W_6, W_7 \in R^{m\times 2m}$ and $b_5, b_6, b_7 \in R^m$ are trainable parameters. I^{key} and T^{key} are the image and text input vectors, respectively. These inputs are concatenated into $[I^{key}, T^{key}]$ and multiplied by W_5 so that the combined modality can be projected into same semantic space with private vectors P_i and P_t.

To ensure this joint representation to project closely to the representation of query, query embedding vector Q_v is generated by LSTM with generated query $y_0, ..., y_{t-1}$:

$$Q_v = \mathrm{LSTM}(Q)$$

with the objective loss to keep private vectors away from query, and shared close to query:

$$L_1^{img} = \max\{0, r - (\mathrm{sim}(Q_v, S) - \mathrm{sim}(Q_v, P_i))\}$$
$$L_1^{text} = \max\{0, r - (\mathrm{sim}(Q_v, S) - \mathrm{sim}(Q_v, P_t))\}$$
$$L_1 = L_1^{img} + L_1^{text}$$

where r is the margin parameter. This margin enables our model to relax the decision function in LXMERT, predicting whether t and i are paired, as a binary classification. Unlike such binary prediction, computing zero or one score for partially paired pair (Figure 1b), the above two losses compute a scalar score and make a soft decision based on similarity. The query-aware relevance loss L_1 is defined by combining the two losses for each modality.

4.2 Query-aware alignment

For **C2**, we revise image representation to highlight query-related semantics, to make it semantically pair better with text representation. To reflect a (possibly nonlinear) relation with the query and the image, a fully connected neural network is applied to each modality before gating. Formally, query-aware image embedding V_p is described below:

$$A_q = \sigma(w_8 Q_v + b_8)$$
$$V_p = (w_9 I^{key} + b_9) \odot A_q$$

where Q_v is the query embedding, A_q is the attention derived from query with sigmoid σ, $\odot$ means element-wise multiplication and $w_9 \in R^{m\times m}$ and $b_9 \in R^m$ are trainable parameters. $w_8 \in R^{m\times m}$ and $b_8 \in R^m$ are learned with m-dimension query embedding. We apply the fully connected layer (first term before element-wise multiplication) to project image near query embedding.

We apply query-aware image representation to the multimodal space learning:

$$L_2 = \max\{0, r - (\mathrm{sim}(V_p, T_+^{key}) - \mathrm{sim}(V_p, T_-^{key}))\}$$

where r is the margin and $+$ means positive text where text belongs into same document with given image and $-$ means negative text from different document.

Finally, we combine the two loss functions as a final query-aware loss L_q:

$$L_q = L_1 + L_2$$

Our final loss of trimodality captioning is the sum of seq2seq loss and query aware alignment loss:

$$L_{final} = L_{seq} + L_q$$

	R@1	R@10	R@30
BM25	0.166	0.672	0.809
LXMERT QG	0.175	0.684	0.813
Bimodal QG	0.207	0.718	0.816
Ours (Trimodal QG)	**0.213**	**0.723**	**0.823**

Table 1: Public dataset results for first-stage retrieval

5 Experiment

The goal of our evaluations is to validate the effectiveness of our approach in public dataset and real-world Web search queries and settings. In particular, we have two research questions:

- **RQ1** Would QG task benefit from LXMERT model? How does our approach compare with BM25 or Bimodal for first-stage retrieval?
- **RQ2** Would our approach improve real-life Web search queries?

We use public dataset for RQ1 and real-life queries and quality annotation for ad-hoc web search task from NAVER for RQ2.

5.1 RQ1: Public reproducible scenarios

5.1.1 Dataset

As there is no public dataset with query workloads and multimodal documents, we repurpose a public dataset of instructional videos (Kim et al., 2020) by transforming videos into multimodal documents, which consists of 2000 query-video pairs, where each video is a recipe instruction from YouCook2 dataset[3]. We first sample an image corresponding to each sentence in the transcript, by capturing a center frame. As this may create too many (image,sentence) pairs, we propose to cluster into more natural boundaries using temporal and semantic aspects: A pair of successive frames, each with textual transcript and a set of objects [4], will be merged if more than $clip\%$ objects overlap, which is empirically tuned for each experiment. When $clip\%$ is set to 100%, it is our initial setting without merging, and this can be tuned to better fit the target scenario. Figure 3 shows the example of video clipping, where associated frames are merged into multimodal paragraphs with images and transcripts. The lengths of extracted multimodal paragraphs are from 1 to 20 according to video. Therefore it can correspond to short to long length of actual documents.

5.1.2 Experiment settings

To preprocess text and image input data, we use NLTK text tokenizer (Sukhbaatar et al., 2015) and Resnet-101 CNN respectively. When learning a query generator, the dimensionality of image and word embedding vector is set to 2048 (by following the size of pool5 vector of ResNet) and 100 respectively. The dimensionality of memory m and query embedding are empirically determined to 256. Mini-batch stochastic gradient descent method is used to learn our query generator. Specifically, we used Adam optimizer (Kingma and Ba, 2014) with the default setting. The initial learning rate is set as 0.001 and is divided by 1.2 at every five epoch until it reaches 30 epochs. The number of generated query is up to 8.

In this study, we follow a standard two-stage document retrieval scenario: First, top 30 candidate documents are ranked and selected from the index using a first-stage ranker, namely BM25, LXMERT, Bimodal, and Trimodal in Table 1. Then, the second-stage ranker follows, which is generally more sophisticated and expensive, such as BERT-based ranker (Nogueira et al., 2019). However, we stress that our work is orthogonal to second-stage ranker and focus on first-stage results.

5.1.3 Results

First-stage ranking results on this public English dataset are shown in Table 1. In this dataset, evaluation metrics is limited to R@K, due to binary nature of relevance annotation: the ratio of ground truth videos that appear in our top-K results, when $K = 30$ is returning all results. In real-life evaluation in the next section, graded relevance annotations will be collected to evaluate rank accuracy. BM25 in the table uses raw BM25 score on text t itself. The other QG models (LXMERT QG, Bimodal QG, and Trimodal QG) are implemented as BM25 scoring on expanded text t' with queries, generated from each QG model respectively. In all evaluations, relevance scores on t and t' are aggregated with linear weighting, which we empirically tune $\lambda = 0.9$ for t' (and $1 - \lambda$ for t). In all metrics, Bimodal and Trimodal outperform BM25 ranking, validating our hypothesis that considering image for joint document representation is effective.

Based on this result, our evaluations from this point on focus on evaluating Trimodal, with real-

---

[3] http://youcook2.eecs.umich.edu/

[4] extracted from each frame using Faster-RCNN (Ren et al., 2015)

(a) Raw video (b) Multimodal document

Figure 3: An example of video clipping to show how we transform a video into a multimodal document. Each multimodal document is clustered into paragraphs, with images and transcripts, shown as red boxes.

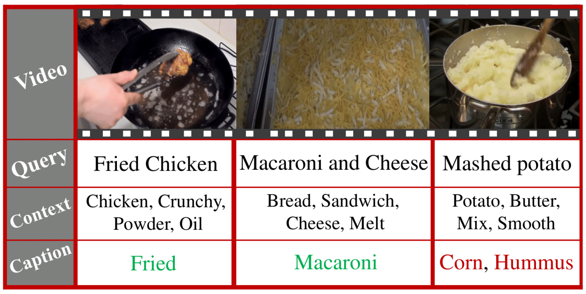


Figure 4: Qualitative examples demonstrating caption generation of our model on public dataset. Successful cases are highlighted by green. Failure case is highlighted by red.

life settings, allowing multilinguality, graded relevance annotation, and a realistic ranker.

5.1.4 Qualitative results

The example images, contexts, and captions are presented in Figure 4. It shows an improvement on search by generating queries for multimodal documents. An example of correctly generating query is shown in the first and second image of Figure 4. The images and contexts are highly related to the name of cooking, but it does not exist in the context where the words are selected by applying TF-IDF to transcript. In this case, our model could make a considerable contribution to search performance by directly generating the query itself like “fried” and “macaroni”. The failure case is shown in the third image of Figure 4. The recipe for hummus and mashed potato both have a mashing step and similar-looking ingredient. If the cooking method and the appearance, color, and texture of the ingredients are similar, the model has a probability of generating other queries. As shown above, our model does great for generating query words to support first-level retrieval.

5.2 RQ2: Real-world ad-hoc web search scenarios

5.2.1 Dataset

The source dataset used in our experiments is the evaluation set of the web search ranking task from the real-life commercial search engine. This dataset contains about 28,000 queries, for each of which 60 document URLs from search engine results are found. In real-time commercial dataset, annotating the relevance of all query-document pairs is impractical. Instead, we pooled top 60 documents, as used widely in IR evaluation to reduce annotation efforts, where only top ranked documents from a small set of retrieval runs are manually assessed for relevance to investigate the impact of first-stage retrieval. These documents are labeled by expert query annotators into one of five graded relevance score, ranging from 1 (poor) to 5 (excellent), or left unlabelled. Since unlabelled documents were randomly sampled from low ranks of search results, we treat all unlabelled documents as irrelevant ones (score 1). Additionally each query is classified into domains by NAVER.we evaluate real world dataset in such experiment setting.

5.2.2 Experiment settings

Out of all domain areas, we observe five main categories where the image information is expected to complement missing information from text–namely, Fashion, Place, Entertainment, Commerce, and Food/Recipe. We select query-document pairs annotated as described above for these categories. More specifically, Table 2 shows the selected categories and the number of queries in each category.

As a realistic ranker, we replace BM25 and train LambdaMart, as implemented in LightGBM (Ke et al., 2017; Meng et al., 2016), which is gradient boosting framework developed by Microsoft that uses tree based learning algorithms. The ranking model is trained and tested separately for each

Category	# of queries
Fashion	119
Entertainment	347
Food/Recipe	205
Place	971
Commerce	1665

Table 2: Statistics of categories selected from real-life data

category. For evaluation metrics, we follow the convention of prior work, to use NDCG@1, 5, 10, evaluated using a 5-fold cross-validation. A detailed description of the text features and image feature used to learn the ranking model is as follows.

To handle Korean text, we replace a tokenizer from KoNLPy [5]. Except this, all other experiment settings, including configurations to learn our query generator, remain unchanged from previous experiments.

BM25F score of query-document
BM25 score of query-document title
BM25 score of query-highlighted text
Exact matching of query-document
Query proximity score on document
Query proximity score on document title
Covered query term ratio of document title
Covered query term ratio of front section
Covered query term ratio of highlighted text

Table 3: Descriptions of real-life text features selected

Each document in real-life search engine is represented by hundreds of pre-computed features. Among them, we select nine widely used features related to textual similarity between a query and document. The selected features are shown in Table 3. Those features are used for the text baseline in Table 4.

5.2.3 Results

Table 4 reports accuracy gains in the five selected categories. For the four of five categories, our proposed approach achieved up to 10.8% gain on NDCG@1.

The category seeing the highest gain has been Food/Recipe, where images can be informative and complement textual instructions in this domain, as consistently observed empirically.

On the other hand, Commerce domain, though we expected showing the image of actual goods would complement text information, was the worst performing category. Our analysis shows that expert annotation was biased to highly rated official sites, while the same item can be sold in millions of sites with lower authority. Meanwhile, our models focusing on document relevance only, following the convention of ad-hoc retrieval scenarios, could not distinguish such difference.

Table 5 shows the search performance of each category when a document is ranked using only trimodal-aware image feature. The best search performance category is the Food/Recipe, which had the highest performance gain in Table 4, and the other categories show a similar performance. The score of ranking model using only image feature can achieve performance about 62% of that of using all features, with respect to NDCG@5.

Table 6 reports the accuracy gains of all categories over strong baselines. Only our trimodal query generation shows positive results on all domains. This demonstrates that our proposed query-aware trimodal loss contributes to capturing the query-relevant semantic of images.

6 Conclusion

We study the problem of representing a multimodal document to be indexable for efficient first-stage. Our contribution is posing the problem as trimodal QG to augment the given text, by proposing a trimodal joint representation of image, text, and query without paired-text assumption. We validate our approach over both public dataset and real-life web search data collected from commercial search engines.

Acknowledgements

This research was supported by the MSIT, under IITP-2017-0-01779; A machine learning and statistical inference framework for explainable artificial intelligence) and the ITRC support program (IITP-2021-2020-0-01789), supervised by the IITP.

References

Azmi Azilawati and Siti Meriam. 2008. Exploiting surrounding text for retrieving web images.

---

[5] Korean Natural Language Processing in python (Park and Cho, 2014)

Category	Model	NDCG@1		NDCG@5		NDCG@10	
Fashion	Text	0.5194	5.9%	0.6542	4.2%	0.7546	2.3%
	Trimodal QG	**0.5502***		**0.6819***		**0.7719***	
Place	Text	0.4806	6.0%	0.5772	4.1%	0.7031	2.0%
	Trimodal QG	**0.5093***		**0.6008****		**0.7175****	
Entertainment	Text	0.4009	10.8%	0.5695	4.8%	0.6916	3.3%
	Trimodal QG	**0.4441***		**0.5970***		**0.7144****	
Commerce	Text	**0.5117***	-3.9%	0.6182	0.1%	**0.7282**	-0.1%
	Trimodal QG	0.4917		**0.6190**		0.7277	
Food/Recipe	Text	0.4862	8.4%	0.6069	5.6%	0.7223	3.4%
	Trimodal QG	**0.5268***		**0.6411****		**0.7468****	

Table 4: First-stage results for real-life dataset. Percentage shows the rate of increase, compared with text-only feature. * and ** indicate statistically significant improvements at $p < 0.05$ level and $p < 0.01$ level respectively.

Category	NDCG@1	NDCG@5	NDCG@10
Fashion	0.2157	0.3616	0.5375
Place	0.2637	0.3778	0.5398
Entertainment	0.2581	0.3878	0.5512
Commerce	0.2539	0.3843	0.5477
Food/Recipe	0.3837	0.4473	0.5904

Table 5: Ablation study of using Trimodal feature only.

Model	NDCG@1	NDCG@5	NDCG@10
LXMERT QG	-0.30%	-0.12%	-0.09%
Bimodal QG	-0.25%	-0.08%	-0.05%
Trimodal QG	+0.30%	+0.12%	+0.08%

Table 6: Result of real-life dataset on all domain category. Numbers show the percentage rate of increases, when compared to using text-only feature.

Tatiana A. S. Coelho, Pável Calado, Lamarque Vieira Souza, Berthier A. Ribeiro-Neto, and Richard R. Muntz. 2004. Image retrieval using multiple evidence ranking. *IEEE Transactions on Knowledge and Data Engineering*, 16:408–417.

Christian Andreas Henning and Ralph Ewerth. 2017. Estimating the information gap between textual and visual representations. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, ICMR '17, page 14–22, New York, NY, USA. Association for Computing Machinery.

Jiwoon Jeon, Victor Lavrenko, and R. Manmatha. 2003. Automatic image annotation and retrieval using cross-media relevance models. In *SIGIR '03*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc.

Kyungho Kim, Kyungjae Lee, and Seung-won Hwang. 2020. Instructional video summarization using attentive knowledge grounding. In *Proceedings of the 2020 European Conference on Machine Learning*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ryan Kiros, R. Salakhutdinov, and R. Zemel. 2014a. Unifying visual-semantic embeddings with multimodal neural language models. *ArXiv*, abs/1411.2539.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. 2014b. Multimodal neural language models. In *ICML*.

Saeid Balaneshin Kordan and Alexander Kotov. 2018. Deep neural architecture for multi-modal retrieval based on joint embedding space for text and images. In *WSDM '18*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *ArXiv*, abs/1908.03557.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *ArXiv*, abs/1908.02265.

Qi Meng, Guolin Ke, Taifeng Wang, Wei Chen, Qiwei Ye, Zhi-Ming Ma, and Tie-Yan Liu. 2016. A communication-efficient parallel algorithm for decision tree. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1279–1287. Curran Associates, Inc.

Fudong Nian, Bing-Kun Bao, Teng Li, and Changsheng Xu. 2017. Multi-modal knowledge representation learning via webly-supervised relationships mining. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 411–419, New York, NY, USA. Association for Computing Machinery.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *ArXiv*, abs/1904.08375.

Cesc Chunseong Park, Byeongchang Kim, and Gunhee Kim. 2017. Attend to you: Personalized image captioning with context sequence memory networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6432–6440.

Eunjeong L. Park and Sungzoon Cho. 2014. Konlpy: Korean natural language processing in python. In *Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology*, Chuncheon, Korea.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149.

S. Robertson, S. Walker, Susan Jones, M. Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In *TREC*.

Sergio Rodríguez-Vaamonde, Lorenzo Torresani, and Andrew W. Fitzgibbon. 2015. What can pictures tell us about web pages? improving document search using images. *IEEE transactions on pattern analysis and machine intelligence*, 37 6:1274–85.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. Weakly supervised memory networks. *CoRR*, abs/1503.08895.

Hao Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP/IJCNLP*.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4651–4659.

Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik G. Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*.

End-to-End Argument Mining as Biaffine Dependency Parsing

Yuxiao Ye
University of Cambridge
Cambridge, UK
yy477@cl.cam.ac.uk

Simone Teufel
University of Cambridge
Cambridge, UK
sht25@cl.cam.ac.uk

Abstract

Non-neural approaches to argument mining (AM) are often pipelined and require heavy feature-engineering. In this paper, we propose a neural end-to-end approach to AM which is based on dependency parsing, in contrast to the current state-of-the-art which relies on relation extraction. Our biaffine AM dependency parser significantly outperforms the state-of-the-art, performing at $F_1 = 73.5\%$ for component identification and $F_1 = 46.4\%$ for relation identification. One of the advantages of treating AM as biaffine dependency parsing is the simple neural architecture that results. The idea of treating AM as dependency parsing is not new, but has previously been abandoned as it was lagging far behind the state-of-the-art. In a thorough analysis, we investigate the factors that contribute to the success of our model: the biaffine model itself, our representation for the dependency structure of arguments, different encoders in the biaffine model, and syntactic information additionally fed to the model. Our work demonstrates that dependency parsing for AM, an overlooked idea from the past, deserves more attention in the future.

1 Introduction

People often hold different opinions about the same thing. One very common way to express one's opinions is to construct an argument (Mercier and Sperber, 2011; Van Eemeren and Henkemans, 2016). To help people efficiently understand different opinions and their reasoning embedded in arguments, it is necessary to develop systems that can automatically analyse structures of arguments. An emerging research field called Argumentation Mining or Argument Mining (AM) (Peldszus and Stede, 2013; Green et al., 2014) addresses this problem.

To analyse the structure of arguments, AM typically proposes four subtasks: 1) *component segmentation*, *i.e.*, cutting a raw sequence into text spans that are either argumentative or non-argumentative segments (only argumentative segments are called argument components); 2) *component classification*, *i.e.*, labelling each argument components with a tag in a pre-defined scheme, (*e.g.*, "PREMISE" or "CLAIM"); 3) *relation detection*, *i.e.*, deciding if two argument components are directly related; and 4) *relation classification*, *i.e.*, categorizing a detected relation into a class in a pre-defined scheme, (*e.g.*, "ATTACK" or "SUPPORT") (Persing and Ng, 2016; Eger et al., 2017; Habernal and Gurevych, 2017; Stab and Gurevych, 2017; Lawrence and Reed, 2020). The first two subtasks are often referred together as *component identification*, and the last two as *relation identification*. An example of the argument structure is illustrated in Figure 1. We see that the "rain" component acts as the premise, supporting the claim of "beautiful".

Some of the AM approaches that try to solve all four subtasks use a pipelined architecture. Independent models are first trained for each subtask; the final results are then achieved by using model ensemble methods such as Integer Linear Programming (Persing and Ng, 2016; Stab and Gurevych, 2017). Like many other pipelined approaches, they often suffer from error propagation. Moreover, many of those models are rule-based or feature-based (Persing and Ng, 2016; Stab and Gurevych, 2017), which require extensive manual efforts and are not flexible or robust in cross-domain scenarios.

As a result, neural end-to-end approaches are desirable for AM. However, it is difficult to jointly model all AM subtasks within one single neural network, because component segmentation and

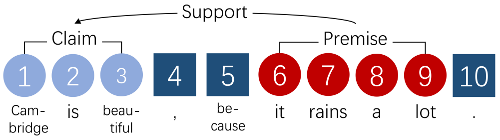


Figure 1: An example argument structure for "*Cambridge is beautiful, because it rains a lot.*"

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 669–678
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

classification are usually approached at the token-level, but relation detection and classification at the segment-level. Therefore, many researchers try to solve only some of them with neural approaches. For example, Morio and Fujita (2019) focus only on component segmentation and classification; Niculae et al. (2017) and Morio et al. (2020) both ignore the component segmentation task, feeding manually segmented text as input into their models.

As far as we know, Eger et al. (2017) present the first study on neural end-to-end AM that addresses all subtasks in one model. They make a comprehensive comparison of methods, among which a model originally proposed for extracting entities and relations (LSTM-ER) achieves the best performance on a popular AM benchmark (Stab and Gurevych, 2017), far outperforming one that uses a dependency parsing (DP) approach.

In this paper, we propose another neural end-to-end approach for the full AM task, called Biaffine Dependency Parsing for Argument Mining (BiPAM). In our approach, AM is formalised as a DP problem, which we model with a modified biaffine neural network (Dozat and Manning, 2018), using our own dependency representation for arguments. Our representation, like the one in Eger et al. (2017), also unifies all AM subtasks under a token-level framework, so that they can be modelled with one single neural network. The biaffine parser in our approach mainly consists of a neural encoder to extract contextualized features of each token in a sequence, and a biaffine classifier that decides which relation holds between any two tokens, and returns a directed acyclic graph (DAG) expressing this information.

Compared with the DP approach in Eger et al. (2017), our model performs at a much higher performance rate. We argue that this is mainly due to the fact that our biaffine model is more powerful in modelling AM-style dependency structures, and also due to other factors such as our dependency representation, which seems closer aligned with linguistic intuitions. Eger et al. (2017) may have discredited the DP approach altogether due to its unsatisfactory results in their setting, but our work reaffirms its potential for AM, given a more powerful parser and a better dependency representation.

Experiments show that our approach also outperforms the best-performing approach (LSTM-ER) in Eger et al. (2017) and thus achieves a new state-of-the-art. Compared with LSTM-ER, our biaffine parser is conceptually and structurally simpler, and our approach is more general because the output of our biaffine parser is a DAG instead of a tree; this is required by several argument schemes which go beyond trees (Park and Cardie, 2018; Lawrence and Reed, 2020).

Our main contributions are as follows:

- We are the first to apply the biaffine parser to AM in an end-to-end approach addressing all four subtasks in one model. Our proposed biaffine argument parser is applicable to non-tree as well as tree-based argument schemes.
- As well as being a theoretically more clean-cut model, our proposed model also significantly outperforms the state-of-the-art approach by 3.3% in F_1 for component identification and 1.3% for relation identification.
- We also present a novel representation for dependency structures of arguments, which is empirically more efficient than Eger et al. (2017).

2 Related Work

Our work is closely related to existing approaches framing AM as DP and to research on neural end-to-end AM.

The essential aim of AM is to analyse the structure of arguments. Most argument schemes (Toulmin, 2003; Peldszus and Stede, 2013; Habernal and Gurevych, 2017; Visser et al., 2019) represent argument structures as trees or DAGs. Similar structures also exist in syntactic and semantic parsing, and can be efficiently analysed with existing dependency parsers (Dozat and Manning, 2017, 2018; Qi et al., 2018). However, unlike syntactic or semantic parsing, dependency structures in AM operate at the segment-level, not the token-level. To apply existing DP techniques to AM, one can either ignore the component segmentation task, only working on already segmented text, or one can convert segment-level dependencies to token-level dependencies.

Morio et al. (2020) take the first approach. In their work, the input is manually segmented, with argumentative and non-argumentative segments already given. They use task-specific BiLSTMs to encode those segments, and a biaffine dependency parser (Dozat and Manning, 2018) with minor alternation to classify argument components and their

(a) (b)

Figure 2: (a) Eger et al. (2017)'s representation, and (b) our representation, for "*Just because it killed much marine life, tourism has threatened nature.*"

relations. They report satisfactory results for component classification and relation identification on the Cornell eRulemaking Corpus (CDCP) (Niculae et al., 2017; Park and Cardie, 2018).

In contrast, Eger et al. (2017) take the second approach, taking raw text as input so that the component segmentation task is addressed. They design a dependency representation for argument structures, in which dependencies are represented at the token-level, with all edges pointing from parent to child[1]. They have evaluated feature-based parsers (McDonald et al., 2005; Bohnet and Nivre, 2012) and neural parsers (Dyer et al., 2015; Kiperwasser and Goldberg, 2016), both of which show unsatisfactory performance. As a result, they discarded DP as inferior.

Our work is similar to Morio et al. (2020) in that we also use a modified biaffine dependency parser to model dependencies in argument structures, but similar to Eger et al. (2017) in that we model AM as a full task starting from tokens, not segments.

Besides DP, Eger et al. (2017) also study other approaches to neural end-to-end AM, including sequence tagging (Ma and Hovy, 2016), multi-task tagging (Søgaard and Goldberg, 2016), and relation extraction (Miwa and Bansal, 2016). Among them, the LSTM-ER model for relation extraction performs the best. However, it requires an additional syntactic parser to produce syntactic dependency trees for the model input. Moreover, the tree-structured LSTM module used to encode those syntactic dependency trees is both conceptually and structurally complicated. In contrast, our proposed approach does not require such syntactic information to break the state-of-the-art (although we will show that it can be enhanced if this information is provided), and the architecture of our biaffine dependency parser is both simpler and more general than that of the LSTM-ER.

3 Proposed Approach

We propose to formalise AM as DP, using a modified biaffine dependency parser, which outputs dependency graphs. To make this work, we first design a dependency representation for arguments which unifies all subtasks of AM under a token-level framework.

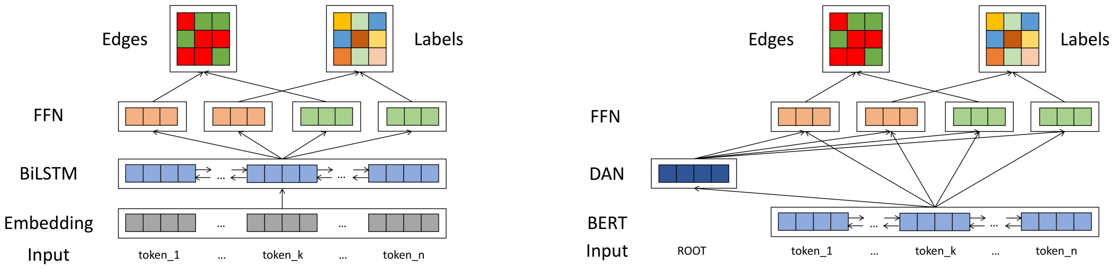
3.1 Dependency Representation for Arguments

Our dependency representation for the structure of arguments contains information about segment boundaries, and about types of relations that could potentially hold between segments, as illustrated in Figure 2(b). The relevant properties of our representation are as follows:

- Information on segment boundaries and segment labels is shown as within-segment labelled edges. Within a segment (*e.g.*, $token[3, ..., 7]$), whether argumentative or non-argumentative, each token, except the last one, is parented by its succeeding token. Labels of within-segment edges are in the form of "($segment\_label \in [component\_label, \mathrm{N}]$, APP)", in which $component\_label = \{\mathrm{MC, C, P...}\}$. "APP" here means "append", and "N" means "non-argumentative segment".
- Information on relations and relation labels is shown as inter-segment labelled edges. If a relation exists between two components, it is expressed as a labelled edge between the last token of the parent, pointing to the last token of the child (*e.g.*, the "SUPPORT" edge from $token[12]$ to $token[7]$). The last token in a non-argumentative segment does not have any parent node.

---

[1] Note the difference in arrow conventions between AM and standard DP. Figure 2 shows dependencies pointing from head to dependent, as is common in DP. In contrast, in AM, relations are shown in the opposite direction (*i.e.*, pointing from parent to child), as shown in Figure 2(a). In the rest of this paper, A→B means A pointing to B in terms of argument schemes, while A⇐B means B pointing to A in terms of dependency representations.



(a) (b)

Figure 3: (a) the biaffine parser in Dozat and Manning (2018), and (b) our modified biaffine parser.

- A pseudo-token *ROOT* is added to the beginning of each argument. This *ROOT* token can be used to represent the topic or the gist of the entire argument, making it possible to model the relationship between components and the topic or the gist. *ROOT* is always acting as a parent to the highest-level component(s).
- Each token is allowed to have zero, one, or more parents, resulting in a dependency graph.

In contrast, Eger et al. (2017) use a representation where each token in a segment is parented by the first token in its parent. But this contradicts general linguistic intuitions, whereby tokens within a segment should be closely related to their surrounding tokens, as is the case in our representation. As a result, we might expect that our representation is better able to model within-segment and long-distance inter-segment dependencies. There are also differences between how we and Eger et al. conceptualize the root node of the argument. They use the terminating token in an argument as root, with all non-argumentative tokens parented by it; in their case this is always a punctuation mark. We think this conceptualisation of the root node is troublesome for two reasons. First, there is no reason why all non-argumentative segments should be related to the terminating token. Second, for arbitrary annotation schemes, the terminating token could be a part of a component. When this happens, all non-argumentative tokens will be parented by the last component in an argument, which goes against linguistic intuitions. In contrast, our representation uses the pseudo-node *ROOT* to avoid this, which is outside all textual segments, and non-argumentative segments never have any inter-segment relations in our representation. In addition, Eger et al. (2017) use the BIO scheme to encode segment boundaries, which results in a larger prediction space than ours, because we can encode the same information topologically[2], making our representation potentially computationally more efficient.

3.2 Modified Biaffine Parser

To our knowledge, the state-of-the-art dependency parser is the one in Dozat and Manning (2018). It consists of a BiLSTM to encoder input text, and a deep biaffine attention module to score each possible head-dependent pair, as shown in Figure 3(a).

The structure of our biaffine parser, as shown in Figure 3(b), closely follows that of Dozat and Manning (2018). We have replaced the Embedding layer and the BiLSTM layer with a pre-trained BERT encoder (Devlin et al., 2019), so that the parser can benefit from rich unsupervised data. Given a text sequence $S = s_1 s_2 ... s_n$, its encoded representation $r_{\text{-}S} \in \mathbb{R}^{n \times d_{enc}}$ is calculated as in Eq. (1), where BERT means the BERT encoder.

$$r_{\text{-}S} = \mathrm{BERT}(s_1 s_2 ... s_n) \tag{1}$$

$$r_{\text{-}ROOT} = \mathrm{FFN}(\mathrm{mean}(r_{\text{-}S}), \mathrm{axis}{=}0) \tag{2}$$

$$R = [r_{\text{-}S}; r_{\text{-}ROOT}], \mathrm{axis}{=}0 \tag{3}$$

$$H^{\mathrm{edge\_parent}} = \mathrm{FFN}^{\mathrm{edge\_parent}}(R) \tag{4}$$

$$H^{\mathrm{label\_parent}} = \mathrm{FFN}^{\mathrm{label\_parent}}(R) \tag{5}$$

$$H^{\mathrm{edge\_child}} = \mathrm{FFN}^{\mathrm{edge\_child}}(R) \tag{6}$$

$$H^{\mathrm{label\_child}} = \mathrm{FFN}^{\mathrm{label\_child}}(R) \tag{7}$$

$$\mathrm{Biaff}(x, y) = x^{\top} \mathrm{U} y + \mathrm{W}(x \oplus y) + b \tag{8}$$

$$sc^{\mathrm{edge}} = \mathrm{Biaff}^{\mathrm{edge}}(H^{\mathrm{edge\_parent}}, H^{\mathrm{edge\_child}}) \tag{9}$$

$$sc^{\mathrm{label}} = \mathrm{Biaff}^{\mathrm{label}}(H^{\mathrm{label\_parent}}, H^{\mathrm{label\_child}}) \tag{10}$$

$$y_{i,j}^{\prime\mathrm{edge}} = \{sc_{i,j}^{\mathrm{edge}} \geq 0\} \tag{11}$$

$$y_{i,j}^{\prime\mathrm{label}} = argmax\ sc_{i,j}^{\mathrm{label}} \tag{12}$$

$$\mathscr{L} = (1 - \lambda)\mathscr{L}^{\mathrm{edge}} + \lambda\mathscr{L}^{\mathrm{label}}, \lambda \in (0, 1) \tag{13}$$

Eq. (2) shows how $r_{\text{-}ROOT} \in \mathbb{R}^{1 \times d_{enc}}$, the representation of *ROOT*, is calculated using a

---

[2] Segment boundaries in our system can be recognised by parenthood or by label – when a label does not contain "APP", a segment boundary has occurred.

deep averaging network (DAN) (Iyyer et al., 2015). $ROOT$ does not go through the BERT encoder like other tokens, because it is considered to represent the gist of the whole argument for the dataset in our experiments. FFN means a feedforward network.

In Eq. (3), $R \in \mathbb{R}^{(n+1)\times d_{enc}}$, the $ROOT$-inclusive representation of S, is derived as the concatenation of $r_{\text{-}ROOT}$ and $r_{\text{-}S}$.

In Eq. (4, 5, 6, 7), four representations are created for R respectively, including two parent-wise representations and two child-wise representations for edge and label prediction, as in Dozat and Manning (2018).

Eq. (8) shows the biaffine classifier in Dozat and Manning (2018), with U, W and b being trainable variables. Using this classifier, edges and their labels are predicted respectively as in Eq. (9, 10, 11, 12). Like Dozat and Manning (2018), the edge-classifier is trained with sigmoid cross-entropy, and the label-classifier with softmax cross-entropy. The total loss is calculated as in Eq. (13). During training, losses are back-propagated to the label-classifier only through edges in the gold standard.

Dropout (Srivastava et al., 2014) is applied for each layer in the proposed model.

4 Experiments

We conduct experiments on a benchmark for AM to evaluate the performance of our model.

4.1 Dataset

The dataset we use in our experiments is a benchmark constructed by Stab and Gurevych (2017), which is also used in Eger et al. (2017). This dataset consists of 402 persuasive essays randomly selected from an online forum (322 for training and 80 for testing). Statistics for this dataset are shown in Table 1.

The argument scheme adopted for this dataset is illustrated in Figure 4. In this dataset, there are four kinds of components. A *major claim* (MAJORCLAIM) is the overall stance of the author, which can be regarded as the overall gist of the whole essay. A *claim* (CLAIM) is a statement that is either *for* (FOR) or *against* (AGAINST) one or more major claims. A *premise* (PREMISE) is the lowest-level component that either *supports* (SUPPORT) or *attacks* (ATTACK) a claim or another premise.

		All	Per Essay
Size	Sentence	7,116	18
	Token	147,271	366
	Paragraph	1,833	5
Components	MAJORCLAIM	751	2
	CLAIM	1,506	4
	PREMISE	3,832	10
	Total	6,089	15
Relation	FOR	2,345	6
	AGAINST	496	1
	SUPPORT	3,613	9
	ATTACK	219	1

Table 1: Statistics of the entire Stab and Gurevych (2017) dataset (test+train).

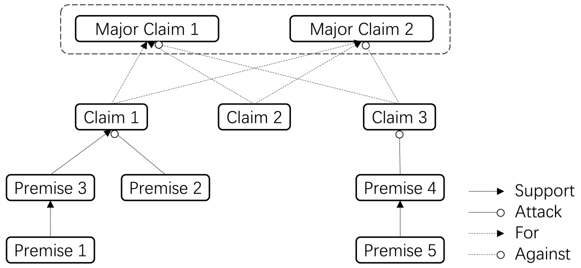


Figure 4: The argument scheme adopted in the Stab and Gurevych (2017) dataset.

Some special characteristics of this dataset are as follows:

- Each essay contains one or more major claims. All major claims within an essay are considered to be equivalent in meaning, and are treated as an equivalence class. In the case of multiple major claims, the structure of the argument is no longer a tree (it is reconstructed as a tree by regarding multiple major claims as one single node in Eger et al. (2017)).
- A claim is allowed to have no supporting or attacking premise.
- Premise→Claim and Premise→Premise pairs are always within-paragraph. Only Claim→MajorClaim pairs are allowed to cut across paragraphs.

Since most relations are within-paragraph for this dataset, our proposed model operates at the paragraph-level. According to our dependency representation, we need a pseudo-token $ROOT$ for each paragraph. If there is one or more major claims in a paragraph, $ROOT$ takes them all as children. Otherwise, $ROOT$ takes all claims in that paragraph as children.

4.2 Competing Models

We choose two models in Eger et al. (2017) for comparison: 1) LSTM-Parser, the best-performing DP model, and 2) LSTM-ER, the overall best-performing model. We use our reimplementation of these models and observe results close to those reported in Eger et al. (2017). Both models are trained at the paragraph-level, with the default hyperparameter configuration provided in the source code. The same pre-trained GloVe embeddings (Pennington et al., 2014) are used as in Eger et al. (2017)[3]. We use the Stanford syntactic dependency parser (Chen and Manning, 2014) to produce syntactic trees required by the LSTM-ER model.

4.3 Training

We create a development set by choosing 30 essays randomly from the training set, which we use for tuning the interpolation factor λ ($\lambda = 0.05$) in Eq. (13) and dropout rate ($dropout = 0.1$), as well as for the early stopping mechanism. We choose the pre-trained BERT model by OpenAI from Devlin et al. (2019) as the encoder. We abandoned another pre-trained encoder GPT2 (Radford et al., 2019) because it performed slightly below BERT. The hidden size of FFNs and biaffine classifiers is set to 600, in line with Dozat and Manning (2018). We use the Adam optimizer (Kingma and Ba, 2014) ($\beta_1 = 0.95, \beta_1 = 0.98, \epsilon = 1e^{-9}$), and adopt the strategy in Vaswani et al. (2017) for the learning rate, with warm up steps set to 1000.

Each model in our experiments, except LSTM-ER and LSTM-Parser (and their related models), which we run only once[4], is trained ten times with different random initialisations. Results are reported in terms of average performance with standard deviations.

4.4 Final Graph Generation

First, several post-processing techniques are applied to make the parser output compatible with the AM dataset (similar techniques are also applied in Eger et al. (2017)), in the following order.

1. The parser sometimes recognises a segment as (mainly) belonging to one label, but interrupts it with a small number of non-fitting labels. We assume the label intended is the majority label and therefore assign it to the segment, if at least 3/5 of the edges share that same majority label in a sequence of consecutive tokens which are linked together.
2. For the remaining inter-segment edges, only valid edges are kept, namely Claim⇐MajorClaim, Premise⇐Claim, and Premise⇐Premise.
3. If two remaining segments are linked by multiple edges, all edges except the one with the highest probability are deleted.
4. If a remaining segment has multiple inter-segment edges, only Claim⇐MajorClaim edges are kept (as only claims can have multiple parents, which have to be major claims).

Component	Model$_a$	Model$_b$
g_1	TP	FN
g_2	FN	FP
...	...	...
g_m	TP	TP
p_1	FP	TN
p_2	FP	FP
...	...	...
p_n	TN	FP

Table 2: Result table for paired permutation tests.

Since our model operates at the paragraph-level rather than the essay-level, we need to combine the sub-graph produced with each paragraph. In the Stab and Gurevych (2017) dataset, only Claim→MajorClaim pairs can connect all sub-graphs in an entire essay. As a result, for each Claim→*ROOT* or Claim→MajorClaim pair, we first eliminate it, and then redirect that claim to all major claims, keeping the original edge label. By doing so, a final graph for the argument structure of the entire essay can be generated.

4.5 Evaluation

Similar to most work in AM, we adopt the evaluation metric in Persing and Ng (2016) to evaluate the performance of AM models. The F_1 scores for component and relation identification are calculated as $F = \frac{2\text{TP}}{2\text{TP}+\text{FP}+\text{FN}}$, in which TP is a true positive, FP false positive, and FN false negative.

We also design a method to apply paired Monte Carlo permutation tests (Dwass, 1957; Nichols and Holmes, 2002) to the results of two AM models. In our situation, results are measured by F_1 scores based on components, but a paired permutation test cannot be straightforwardly applied, because in general the number of components returned by the models is not the same. To address this, a result table is first constructed, as illustrated in Table 2. The set of items consists of the union of : 1) all gold standard components $\{g_1, g_2, ..., g_m\}$, and 2) all predicted components $\{p_1, p_2, ..., p_m\}$ by both models (those that are not duplicates, *i.e.*already contained in the gold standard). The treatment of

---

[3] This pre-trained GloVe embeddings are used for all LSTM encoders in our experiments.

[4] This is due to excessively long training time.

Model	Component	Relation
LSTM-Parser	58.4 (58.9)	34.9 (35.6)
LSTM-ER	70.2 (70.8)	45.1 (45.5)
BiPAM	72.9 ± 0.7	45.9 ± 0.5

Table 3: F_1 scores for competing models and our BiPAM model, with published results for competing models (the best instead of average results) in brackets.

gold standard components is standard for such tests (*i.e.*, TP, FN, FP are possible labels), but the predicted components are added by us, and can only be labelled as TN or FP. As these components are not in the gold standard, each time a model proposes such a component, it will be punished by receiving an FP. A predicted component p_i counts as a true negative (TN) for Model$_a$ if p_i has been predicted by Model$_b$ but not Model$_a$. Permutations are generated by swapping labels in the second and third columns for combinations of randomly selected rows. The two-tailed p-value is then calculated as in Eq. (14, 15), in which F_a means the real F_1 score for Model$_a$, and $F_a'^i$ means the F_1 score for Model$_a$ in the i-th permutation. N is the number of sampled permutations. When a model is trained with multiple initialisations, the one performing closest to the average is used for permutation tests.

$$diff_i = \begin{cases} 1 & , |F_a - F_b| < |F_a'^i - F_b'^i| \\ 0 & , |F_a - F_b| \geq |F_a'^i - F_b'^i| \end{cases} \quad (14)$$

$$p\text{−value} = \frac{1 + \sum_{i=1}^{N} diff_i}{1 + N} \quad (15)$$

5 Results and Discussion

The overall results of our experiments are shown in Table 3. For both component and relation identification, our model BiPAM significantly ($p < 0.01$) outperforms the state-of-the-art model LSTM-ER (72.9% vs. 70.2% and 45.9% vs. 45.1%), and also significantly outperforms LSTM-Parser by a large margin.

We can see from these results that the DP approach for AM is able to achieve the best results currently known on this dataset. This is interesting, as the current state-of-the-art, LSTM-ER, uses a far more complex neural architecture than our BiPAM.

The equivalent model to our BiPAM in Eger et al. (2017), LSTM-Parser, is outperformed by BiPAM by a large margin. We believe this is due to two factors: a biaffine parser is far more suitable for AM than an LSTM parser, and our dependency representation is superior to Eger et al. (2017)'s. We will now investigate the influence of these two factors. As we have already shown that our version of this approach beats the state-of-the-art, we will consider only those models which treat AM as DP.

Model	Component	Relation
LSTM-Eger (LSTM-Parser)	58.4	34.9
LSTM-ours	67.9	36.0
Biaff-Eger	65.7 ± 0.5	38.4 ± 0.5
Biaff-ours (BiPAM)	72.9 ± 0.7	45.9 ± 0.5

Table 4: F_1 scores for models with different combinations of parsers and representations.

5.1 Ablation Study

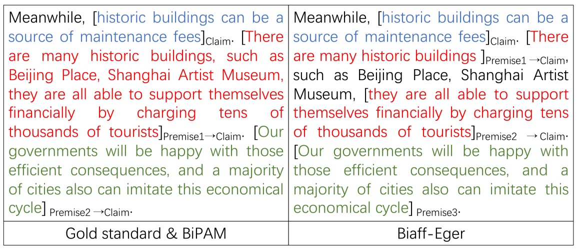
To understand why our proposed model performs better than LSTM-Parser in Eger et al. (2017), we conduct ablation experiments to assess the influence of our modified biaffine dependency parser and our representation for the dependency structure of arguments.

Besides LSTM-Parser and BiPAM (renamed as LSTM-Eger and Biaff-ours in Table 4), we also experiment with two other combinations of parsers and representations: 1) LSTM-ours, replacing Eger et al. (2017)'s representation in LSTM-Parser with ours; and 2) Biaff-Eger, replacing our representation in BiPAM with Eger et al. (2017)'s. Results of these cross-comparisons are shown in Table 4.

As far as parsers are concerned, our modified biaffine parser with either representation (Biaff-Eger and Biaff-ours) performs significantly ($p < 0.01$) better than its counterpart (LSTM-Eger and LSTM-ours). This suggests that our modified biaffine parser is better at capturing dependencies than the LSTM parser in Eger et al. (2017).

As for representations, either parser with our representation (LSTM-ours and Biaff-ours) significantly ($p < 0.01$) outperforms its counterpart (LSTM-Eger and Biaff-Eger). This demonstrates that our representation is superior.

The example text in Figure 5 is a paragraph in the test data that illustrates this fact. Our BiPAM model correctly predicts all components in it, as well as all the relations. In contrast, in the prediction of Biaff-Eger, the parenthetical phrase "such as Beijing Place[sic], Shanghai Artist Museum," in the relatively long component Premise2, is incorrectly recognized as non-argumentative. Biaff-Eger also fails to predict the relation between Premise3 and Claim, which is probably because these two components are located far from each other at opposite ends of the paragraph, making it difficult to detect the relations between and the first token in



Gold standard & BiPAM	Biaff-Eger
Meanwhile, [historic buildings can be a source of maintenance fees]$_{\text{Claim}}$. [There are many historic buildings, such as Beijing Place, Shanghai Artist Museum, they are all able to support themselves financially by charging tens of thousands of tourists]$_{\text{Premise1}\rightarrow\text{Claim}}$. [Our governments will be happy with those efficient consequences, and a majority of cities also can imitate this economical cycle] $_{\text{Premise2}\rightarrow\text{Claim}}$.	Meanwhile, [historic buildings can be a source of maintenance fees]$_{\text{Claim}}$. [There are many historic buildings]$_{\text{Premise1}\rightarrow\text{Claim}}$, such as Beijing Place, Shanghai Artist Museum, [they are all able to support themselves financially by charging tens of thousands of tourists]$_{\text{Premise2}\rightarrow\text{Claim}}$. [Our governments will be happy with those efficient consequences, and a majority of cities also can imitate this economical cycle] $_{\text{Premise3}}$.

Figure 5: Argument structures of an example paragraph predicted by BiPAM and by Biaff-Eger.

Model	Component	Relation
BiPAM	72.9 ± 0.7	45.9 ± 0.5
BiPAM-LSTM	71.4 ± 0.7	43.1 ± 0.7

Table 5: F_1 scores for BiPAM and BiPAM-LSTM.

Claim and tokens in Premise3.

We also want to understand the influence of the pre-trained BERT encoder in the biaffine parser. When we built BiPAM, we modified the original biaffine parser (Dozat and Manning, 2018) by replacing its Embedding layer and BiLSTM layer, with a pre-trained BERT encoder. It is possible that our performance improvements are at least partially due to this. We therefore test against a biaffine model without this replacement[5], named BiPAM-LSTM in Table 5. Results show that replacing the pre-trained BERT encoder with the original Embedding layer and BiLSTM layer significantly ($p < 0.01$) lowers the performance. This is easy to understand since the pre-trained BERT encoder has benefited from an tremendous amount of additional unsupervised data[6].

5.2 Enhancement with Syntactic Information

We believe that a part of the success of LSTM-ER in Eger et al. (2017) might be credited to the fact that the model is also given syntactic information in its input, in addition to the raw text. We borrow this idea to test whether BiPAM might get a performance boost in a similar way.

To do so, we use the Stanford syntactic dependency parser to produce syntactic trees. The pseudo-token *ROOT* is headed by the real root node in the syntactic tree, with an edge labelled as "ROOT". For each input token, we record its syntactic information in a triple as [head, location of head, label of in-coming edge]. The syntactic information triples are encoded as follows: 1) head tokens are encoded by an FFN with pre-trained GloVe embeddings; 2) head locations are encoded using the positional encoding method proposed in Vaswani et al. (2017); and 3) edge labels are passed to an FFN using one-hot encoding. For each token, the encoded representation of its syntactic information triple is concatenated to its encoded representation by BERT, and the entire representation is then fed into the following layers in BiPAM.

Model	Component	Relation
BiPAM	72.9 ± 0.7	45.9 ± 0.5
BiPAM-syn	73.5 ± 0.7	46.4 ± 0.6

Table 6: F_1 scores for BiPAM and BiPAM enhanced with syntactic information.

Results in Table 6 show that introducing syntactic information to the input can significantly ($p < 0.01$) boost the performance of the original BiPAM, outperforming LSTM-ER by 3.3% for component identification and 1.3% for relation identification. These results represent our best and final system.

6 Conclusion

In this paper, we solve argument mining (AM) as a neural end-to-end dependency parsing (DP) problem. As a non-pipelined approach, our solution is free from error propagation. We have demonstrated that the biaffine dependency parser from Dozat and Manning (2018) can act as a powerful AM parser, under the right circumstances. We use BERT as the encoder, feed syntactic information into the model in addition to raw text, and propose a novel, efficient and linguistically plausible representation for the dependency structure of arguments. Our model significantly outperforms current state-of-the-art in neural end-to-end AM, performing at $F_1 = 73.5\%$ for component identification and 46.4% for relation identification. Our research suggests that DP, which has been prematurely abandoned in the past, is actually a very promising approach for AM. Through an ablation study, we find that both the modified biaffine parser and the dependency representation contribute to the performance improvement of BiPAM.

Acknowledgments

Yuxiao Ye is supported by a Ph.D. Studentship funded by Toshiba Research Europe Limited.

---

[5] With this model, there is one change we cannot avoid, and this is that we have to keep the DAN layer for *ROOT*, which does not go through the encoder.

[6] One could also argue that the right way to address the in-domain vs. out-of-domain problem is to train the BERT encoder from scratch using our in-domain training data. We do not perform these experiments, because due to the small data size, it is almost certain that BERT would be over-fitted.

References

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1455–1465.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

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

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the International Conference on Learning Representations*.

Timothy Dozat and Christopher D Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490.

Meyer Dwass. 1957. Modified randomization tests for nonparametric hypotheses. *The Annals of Mathematical Statistics*, pages 181–187.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–22.

Nancy Green, Kevin D Ashley, Diane Litman, Chris Reed, and Vern Walker. 2014. Proceedings of the first workshop on argumentation mining. In *Proceedings of the First Workshop on Argumentation Mining*.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation mining in user-generated web discourse. *Computational Linguistics*, 43(1):125–179.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 1681–1691.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of human language technology conference and conference on empirical methods in natural language processing*, pages 523–530.

Hugo Mercier and Dan Sperber. 2011. Why do humans reason? arguments for an argumentative theory.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116.

Gaku Morio and Katsuhide Fujita. 2019. Syntactic graph convolution in multi-task learning for identifying and classifying the argument component. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 271–278. IEEE.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. 2020. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3259–3266.

Thomas E Nichols and Andrew P Holmes. 2002. Nonparametric permutation tests for functional neuroimaging: a primer with examples. *Human brain mapping*, 15(1):1–25.

Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument mining with structured svms and rnns. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 985–995.

Joonsuk Park and Claire Cardie. 2018. A corpus of erulemaking user comments for measuring evaluability of arguments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Andreas Peldszus and Manfred Stede. 2013. From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 7(1):1–31.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170.

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

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Stephen E Toulmin. 2003. *The uses of argument*. Cambridge university press.

Frans H Van Eemeren and A Francisca Sn Henkemans. 2016. *Argumentation: Analysis and evaluation*. Taylor & Francis.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jacky Visser, John Lawrence, Jean Wagemans, and Chris Reed. 2019. An annotated corpus of argument schemes in us election debates. In *Proceedings of the 9th Conference of the International Society for the Study of Argumentation (ISSA), 3-6 July 2018*, pages 1101–1111.

FakeFlow: Fake News Detection by Modeling the Flow of Affective Information

Bilal Ghanem[1], **Simone Paolo Ponzetto**[2], **Paolo Rosso**[1], **Francisco Rangel**[3]
[1]Universitat Politècnica de València, Spain
[2]University of Mannheim, Germany
[3]Symanto Research, Germany
bigha@doctor.upv.es, simone@informatik.uni-mannheim.de,
prosso@dsic.upv.es, francisco.rangel@symanto.com

Abstract

Fake news articles often stir the readers' attention by means of emotional appeals that arouse their feelings. Unlike in short news texts, authors of longer articles can exploit such affective factors to manipulate readers by adding exaggerations or fabricating events, in order to affect the readers' emotions. To capture this, we propose in this paper to model the flow of affective information in fake news articles using a neural architecture. The proposed model, FakeFlow, learns this flow by combining topic and affective information extracted from text. We evaluate the model's performance with several experiments on four real-world datasets. The results show that FakeFlow achieves superior results when compared against state-of-the-art methods, thus confirming the importance of capturing the flow of the affective information in news articles.

1 Introduction

In today's information landscape, fake news are used to manipulate public opinion (Zhou and Zafarani, 2018) by reshaping readers' opinions regarding some issues. In order to achieve this goal, authors of fake news' narratives need to capture the interest of the reader. Thus, they are putting efforts to make their news articles look more objective and realistic. This is usually done by adding misleading terms or events that can have a negative or positive impact on the readers' emotions.

Short text false information, e.g., fake claims or misleading headlines, might be less harmful than news articles. They may have some eye-catching terms that aim to manipulate the readers' emotions (Chakraborty et al., 2016). In many cases, the identification of this kind of exaggeration in short statements can unmask the fabrication. On the other hand, in fake news articles the authors exploit the length of the news to conceal their fabricated story. This fact exposes the readers to be emotionally manipulated while reading longer texts that have several imprecise or fabricated plots. The flow of information has been investigated for different tasks: Reagan et al. (2016) studied the emotional arcs in stories in order to understand complex emotional trajectories; Maharjan et al. (2018) model the flow of emotions over a book and quantify its usefulness for predicting success in books; Kar et al. (2018) explore the problem of creating tags for movies from plot synopses using emotions.

Unlike previous works (Rashkin et al., 2017; Shu et al., 2018; Castelo et al., 2019; Ghanem et al., 2020) that discarded the chronological order of events in news articles, in this work we propose a model that takes into account the affective changes in texts to detect fake news. We hypothesize that fake news has a different distribution of affective information across the text compared to real news, e.g. more fear emotion in the first part of the article or more overall offensive terms, etc. Therefore, modeling the flow of such information may help discriminating fake from real news. Our model consists of two main sub-modules, topic-based and affective information detection. We combine these two sub-modules since a news article's topic may have a correlation with its affective information. For example, a fake news article about Islam or Black people is likely to provoke fear and express negative sentiment while another fake news that is in favor of a particular politician might try to evoke more positive emotions and also express some exaggerations.

The contributions of our work are as follows:

- We design a model that detects fake news articles by taking into account the flow of affective information[1].

---

[1] Available at https://github.com/bilalghanem/fake_flow

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 679–689
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

- Extensive experiments on four standard datasets demonstrate the effectiveness of our model over state-of-the-art alternatives.
- We build a novel fake news dataset, called MultiSourceFake, that is collected from a large set of websites and annotated on the basis of the joint agreement of a set of news sources.

2 Related Work

Previous work on fake news detection is mainly divided into two main lines, namely with a focus on social media (Zubiaga et al., 2015; Aker et al., 2017; Ghanem et al., 2019) or online news articles (Tausczik and Pennebaker, 2010; Horne and Adali, 2017; Rashkin et al., 2017; Barrón-Cedeno et al., 2019). In this work we focus on the latter one. Fact-checking (Karadzhov et al., 2017; Zlatkova et al., 2019; Shu et al., 2019a) is another closely related research topic. However, fact-checking targets only short texts (that is, claims) and focuses on using external resources (e.g. Web, knowledge sources) to verify the factuality of the news. The focus in previous work on fake news detection is mainly on proposing new feature sets. Horne and Adali (2017) present a set of content-based features, including readability (number of unique words, SMOG readability measure, etc.), stylistic (frequency of part-of-speech tags, number of stop words, etc.) and psycholinguistic features (i.e., several categories from the LIWC dictionary (Tausczik and Pennebaker, 2010)). When these features are fed into a Support Vector Machine (SVM) classifier and applied, for instance, to the task of distinguishing satire from real news, they obtain high accuracies. Using the same features for the task of fake news detection, however, results in somewhat lower scores. Pérez-Rosas et al. (2018) propose a model (FakeNewsDetector) that uses a feature set consisting of unigrams and bigrams, psycholinguistic, readability, punctuation and dependency-based syntactic features, and they evaluate the performance of their model in a cross-domain experiment. Rashkin et al. (2017) use a model based on ngram features with a MaxEntropy classifier and apply it to a dataset with different types of fake news articles (e.g., satire, hoax, propaganda, etc.). Similar to the previous work, the authors evaluate their system's performance on in-domain and out-of-domain test sets, respectively. News, and in particular fake news, are dynamic in nature and change constantly. In order to approach the dynamic nature of news, Castelo et al. (2019) propose a topic-agnostic model (TopicAgnostic) that is based on morphological (count of part-of-speech tags), psycholinguistic (personal concerns, affection, and perception categories from the LIWC dictionary), readability (Gunning Fog metric, etc.) and Web-Markup features to capture patterns of the Web pages' layout (frequency of advertisements, presence of an author name, etc.). All of the morphological, psycholinguistic and readability features in the TopicAgnostic model were extracted from headlines and texts of the news articles. The approach obtains a better performance than FakeNewsDetector on three different datasets using a SVM classifier. FakeNewsTracker (Shu et al., 2019b) is a deep neural network-based model that consists of two branches: one encodes news article texts and the other encodes social media engagements (e.g., tweets and their replies). A similar model called Emotionally Infused Network (EIN) is proposed in Ghanem et al. (2020). EIN encodes the text of the article and their affective content, based on several dictionaries, and then combines the two vector representations. The authors evaluate their model on a multi-class false information dataset and show the effectiveness of using emotion features extracted from the text. Despite the large variety of features and models that have been explored in previous work, none of these works considers the sequence of affective information in text; instead, they feed the entire news articles as one segment into their models. In contrast, the aim of our work is to evaluate this source of information, using a neural architecture.

3 The FakeFlow Model

Given an input document, the FakeFlow model first divides it into N segments. Then it uses both word embeddings and other affective features such as *emotions, hyperbolic words*, etc. in a way to catch the flow of emotions in the document. The model learns to pay attention to the flow of affective information throughout the document, in order to detect whether it is fake or real.

Figure 1 shows the architecture of the FakeFlow model. The neural architecture has two main modules: The first module uses a Convolutional Neural Network (CNN) to extract topic-based information from articles (left branch). The second module models the flow of the affective information within the articles via Bidirectional Gated Recurrent Units (Bi-GRUs) (right branch).

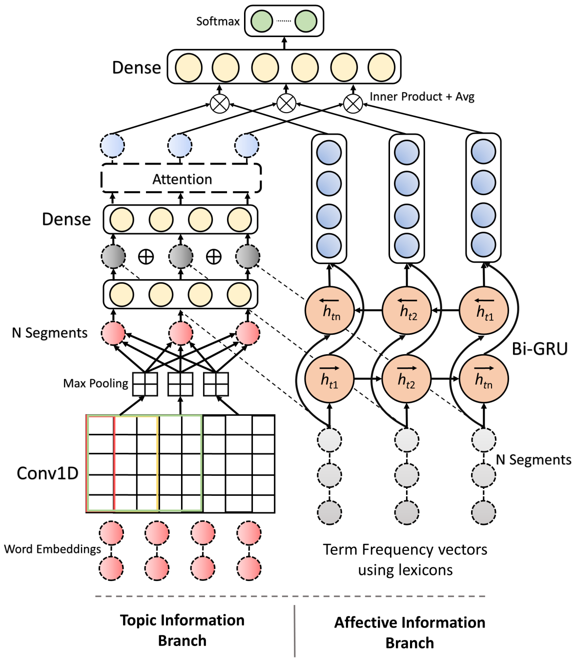


Figure 1: The architecture of the FakeFlow model.

3.1 Topic-based Information

Given a segment $n \in N$ of words, the model first embeds words to vectors through an embedding matrix. Then it uses a CNN that applies convolution processes and max pooling to get an abstractive representation of the input segment. This representation highlights important words, in which the topic information of the segment is summarized. Then it applies a fully connected layer on the output segments to get a smaller representation (v_{topic}) for later concatenation with the representation of affective information:

$$v_{topic} = f(W_a\, cnn_v + b_a)$$

where W_a and b_a are the corresponding weight matrix and bias terms, and f is an activation function such as ReLU, tanh, etc.

Key to FakeFlow is its ability to capture the relevance of the affective information with respect to the topics. For this, we concatenate the topic summarized vector v_{topic} with the representation vector v_{affect}, aimed at capturing the affective information extracted from each segment (Section 3.2).

$$v_{concat} = v_{topic} \oplus v_{affect}$$

To merge the different representations and capture their joint interaction in each segment, the model processes the produced concatenated vector v_{concat} with another fully connected layer:

$$v_{fc} = f(W_c\, v_{concat} + b_c)$$

In order to create an attention-focused representation of the segments to highlight important ones and to provide the model with the ability to weight segments differently according to the similarity of neighboring segments, the model applies a context-aware self-attention mechanism (Zheng et al., 2018) on v_{fc}. This is a crucial step, as the importance of a segment at timestep t is related to the other segments since they share the same context in the news article. Moreover, applying the attention layer can help us understand which features are most relevant by showing to which words the network attends to during learning. The output of the attention layer is an attention matrix l_t with scores for each token at each timestep.

3.2 Affective Flow of Information

To model the affective information flow in the news articles, we choose the following lexical features, under the assumption that they have a different distribution across the articles' segments. We use a term frequency representation weighted by the articles' length to extract the following features from each segment n:

- *Emotions*: We use emotions as features to detect their change among articles' segments. For that we use the NRC emotions lexicon (Mohammad and Turney, 2010) that contains ∼14K words labeled using the eight Plutchik's emotions (*8 Features*).

- *Sentiment*: We extract the sentiment from the text, *positive* and *negative*, again using the NRC lexicon (Mohammad and Turney, 2010) (*2 Features*).

- *Morality*: We consider cue words from the Moral Foundations Dictionary[2] (Graham et al., 2009) where words are assigned to one (or more) of the following categories: *care, harm, fairness, unfairness (cheating), loyalty, betrayal, authority, subversion, sanctity* and *degradation* (*10 Features*).

- *Imageability*: We use a list of words rated by their degree of abstractness and imageability[3]. These words have been extracted from the MRC psycholinguistic database (Wilson, 1988) and then using a supervised learning algorithm, the

---

[2] https://moralfoundations.org/other-materials/

[3] https://github.com/ytsvetko/metaphor/tree/master/resources/imageability

words have been annotated by the degrees of abstractness and imageability. The list contains 4,295 and 1,156 words rated by their degree of abstractness and imageability, respectively (*2 Features*).

- *Hyperbolic*: We use a list of ∼350 hyperbolic words (Chakraborty et al., 2016), i.e., words with high positive or negative sentiment (e.g., terrifying, breathtakingly, soul-stirring, etc.). The authors extracted these eye-catching words from clickbaits news headlines (*1 Feature*).

To model the flow of the above features, we represent each segment of an article by a vector v_{affect} capturing all 23 features listed above. Then we feed the document's vectors to a Bi-GRU network to summarize the contextual flow of the features from both directions[4] to obtain v_{flow}.

Given the segments' flow representation (v_{flow}) of an article and their relevance to the topics (l_t), FakeFlow applies a dot product operation and then averages the output matrix across the segments to get a compact representation $v_{compact}$, which is then fed into a fully connected layer:

$$v_{final} = f(W_d \, v_{compact} + b_d)$$

Finally, to generate the overall factuality label of an article, a softmax layer is applied to the output of the fully connected layer.

4 Fake News Datasets

Despite the recent efforts for debunking online fake news, there is a dearth of publicly available datasets. Most of the available datasets are small in size (e.g., the Politifact[5] dataset in (Shu et al., 2018) has ∼700 available articles, the Celebrity dataset in (Pérez-Rosas et al., 2018) has ∼500 articles, etc.), their test parts have not been manually annotated, or have been collected from a very small number of news sources. Nonetheless, we evaluate FakeFlow on three different available datasets to demonstrate its performance. In addition, we create our own dataset. Table 1 gives an overview of the datasets that we used in our work.

MultiSourceFake: We rely on different resources for creating the training and test portions of the dataset, so as to provide a challenging benchmark.

For the training part, we use *OpenSources.co* (OS), *MediaBiasFactCheck.com* (MBFC), and PolitiFact[6] news websites' lists. OS list contains 560 domains, MBFC list has 548 domains, and the PolitiFact list has 227 domains. These lists have been annotated by professional journalists. The lists contain domains of online news websites annotated based on the content type (as in the OS news list: *satire, reliable*, etc.; and in the PolitiFact news list: *imposter, parody, fake news*, etc.) or from a factuality perspective (as in the MBFC news list: low, medium, and high factuality). From the OS list, we select domains that are in one of the following categories: *fake, bias, reliable, hate, satire,* or *conspiracy*. We consider domains under the *reliable* category as real news sources, and the rest as fake. The PolitiFact list is different from the OS list since it has only labels for domains that are either fake or with mixed content. We discard the mixed ones[7] and map the remaining ones to the fake news label. Finally, we select from the MBFC list those domains that are annotated either as high or low factual news and we map them to real and fake labels, respectively. Out of these three final lists, we select only those domains for our dataset that are annotated in all lists in a consistent way; for example, we discard those domains that are annotated as real in the OS list but their label in the MBFC list is fake (low factuality). The final list contains 85 news websites. We now proceed by projecting the domain-level ground truth onto the content of those domains and randomly sample articles, with a maximum of 100 news articles per domain.[8]

For the test part, we use the *leadstories.com* fact checking website for which professional journalists annotated online news articles on the article level as fake or real. We do not follow the way we annotate the training part since the projection of the domain-level ground truth inevitably introduces noise. The journalists that annotated *leadstories.com* assigned a set of labels to the fake news articles like, e.g., *false, no evidence, satire, misleading*, etc.; we map them all to the *fake* label. In addition, we discard all articles that are multimedia-based. After collecting the news articles, we postprocess them by discarding very short articles (less than 30 words). The test part includes 689 fake news articles. We complement the set with a sample of 1,000 real

---

[4] During prototyping, GRU produced better overall results than LSTM.

[5] https://www.politifact.com/

[6] https://www.politifact.com/article/2017/apr/20/politifacts-guide-fake-news-websites-and-what-they/

[7] The discarded label is "Some fake stories".

[8] Some of the websites included less than 100 news articles.

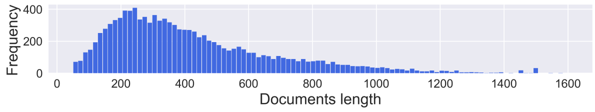


Figure 2: The distribution of the documents' length in the MultiSourceFake dataset.

Name	Total	Training	Test
MultiSourceFake	11,397	9,708	1,689
TruthShades	23,000	16,000	4,000 - 3,000
PoliticalNews	14,240	11,392	2,848
FakeNewsNet	20,208	16,156	4,039

Table 1: Number of articles in the datasets.

news articles from the training part. The overall dataset consists of 5,994 real and 5,403 fake news articles. The average document length (number of words) in the MultiSourceFake dataset is 422 words, and the 95th percentile value is 942. Figure 2 shows the distribution of the documents' length in the dataset.

TruthShades: This dataset has been proposed in Rashkin et al. (2017). The dataset was crawled from a set of domains that are annotated by professional journalists as either *propaganda, hoax, satire*, or *real*. The dataset has been built from the English Gigaword corpus for real news, and other seven unreliable domains that annotated in one of the three previous false information labels.

PoliticalNews: Due to the fact that: "a classifier trained using content from articles published at a given time is likely to become ineffective in the future" (Castelo et al., 2019), the authors of this work collected a dataset by crawling news websites in between the years 2013 to 2018 in order to evaluate their model's performance on different years.

FakeNewsNet: is a fake news repository that consists of two comprehensive datasets, one collected using claims from PolitiFact and the other from the GossipCop fact checking website. Given the large number of true and false claims from these two fact checking websites, Shu et al. (2018) built news datasets that contain visual and textual news articles content and social media information by searching Twitter for users who shared news. Out of the whole collected information, we use only the textual information of news articles, which is the part we are interested in.

5 Experiments

Experimental setup. We split the articles' text into N segments and set the maximum length of segments to 800 words, applying zero padding to the ones shorter than 800 words. Concerning the FakeFlow hyper-parameters, we tune various parameters (*dropout, the size of the dense layers, activation functions, CNN filter sizes and their numbers, pooling size, size of the GRU layer, and the optimization function*) (see Appendix A for the search space) using early stopping on the validation set. In addition to these hyper-parameters, we also use the validation set to pick the best number of segments (N). Regarding the MultiSourceFake dataset, we use 20% of the training part for validation. We represent words using pre-trained word2vec *Google-News-300* embeddings[9]. For evaluation, we follow the setup from related work. We report accuracy and weighted precision, recall and F1 score, and macro F1 for some datasets where the classes are imbalanced.

Baselines. To evaluate the performance of our model, we use a combination of fake news detection models and deep neural network architectures:

- **CNN, LSTM**: We use CNN and LSTM models and validate their performance when treating each document as one fragment. We experiment with different hyper-parameters and report results for the ones that performed best on the validation set.
- **HAN**: The authors of (Yang et al., 2016) proposed a Hierarchical Attention Networks (HAN) model for long document classification. The proposed model consists of two levels of attention mechanisms, i.e., word and sentence attention. The model splits each document into sentences and learns sentence representations from words.
- **BERT**: is a text representation model that showed superior performance on multiple natural language processing (NLP) benchmarks (Devlin et al., 2019). We use the pre-trained *bert-base-uncased* version which has 12-layers and yields output embeddings with a dimension of size 768. We feed the hidden representation of the special [CLS] token, that BERT uses to summarize the full input sentence, to a softmax layer. Experimentally, we found that fine-tuning BERT

---

[9] https://code.google.com/archive/p/word2vec/

layers gives a higher performance. It is worth mentioning that BERT input length is limited to 512 word pieces (sub-words level) (Devlin et al., 2019), thus, we discard the rest of the text in long news articles.

- **Fake News Detection Models**: We compare our model to several fake news detection models. We use Horne and Adali (2017) model, FakeNewsDetector (Pérez-Rosas et al., 2018), Rashkin et al. (2017) model, and EIN (Ghanem et al., 2020).[10]
- **Longformer**: Giving that Transformer-based models (i.e. BERT) are unable to process long sequences, we use Longformer (Beltagy et al., 2020), which is a SOTA model for long document tasks. In our experiments, we set the max sequence length to 1500 to handle documents that have more than 512 tokens in the MultiSourceFake dataset (see Figure 2). Also, we found that fine-tuning the Longformer model gives better results and a much faster convergence.

6 Results and Analysis

Table 2 presents the results of our proposed model and the baselines on the MultiSourceFake dataset. Our best result was achieved by using 10 as the number of segments (N, as found on the validation data). In Figure 3 we show the model's performance for segments of different length.[11] In general, the results show that models that are based on either word ngrams or word embeddings are performing better than other models that use handcrafted features, e.g. Horne and Adali (2017). Also, despite the huge amount of data used to train the BERT model, the results show that BERT performs worse than *FakeFlow* and also fails to outperform some of the other models. We speculate that this is due to the fact that the input length in BERT is limited to 512 words, as we mentioned previously, and a large portion of the news articles in the MultiSourceFake dataset has a length greater than 512 words. The results of the Longformer model confirm our claim regarding the documents' length and show a significantly higher F1 score than the BERT model. This emphasizes that despite the strong performance of BERT on multiple NLP benchmarks, it is unable to handle long text documents, in contrast, e.g., to vanilla text categorization (Adhikari et al., 2019). In addition, Longformer's results show a higher F1 score than the FakeFlow model, yet, the difference is statically insignificant.

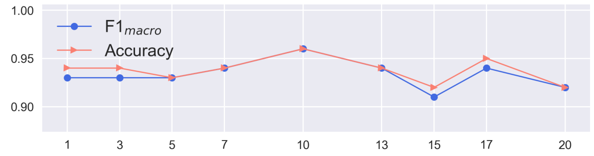


Figure 3: The accuracy and F1 results of the FakeFlow model using different N (number of segments).

Model	Acc.	Prec.	Rec.	$\mathbf{F1}_{macro}$
Majority Class	0.59	0.35	0.59	0.37
Horne and Adali (2017)	0.80	0.75	0.78	0.80
FakeNewsDetector	0.86	0.86	0.86	0.86
LSTM	0.91	0.86	0.91	0.90
CNN	0.91	0.89	0.89	0.91
Rashkin et al. (2017)	0.92	0.92	0.92	0.92
BERT	0.93	0.93	0.94	0.93‡
EIN	0.93	0.94	0.93	0.93‡
HAN	0.94	0.94	0.94	0.93‡
Longformer	0.97	0.97	0.97	**0.97**†
FakeFlow	0.96	0.93	0.97	0.96
FakeFlow – Topic only	0.91	0.89	0.90	0.90
FakeFlow – Affective only	0.61	0.38	0.60	0.40

Table 2: Results on the MultiSourceFake dataset. (‡) indicates a statistically significant improvement of *FakeFlow* over the referred model using McNemar test; (†) indicates no statistically significant improvement over *FakeFlow*.

To isolate the contribution of topical vs. affective information we run two simplified versions of our architecture, each consisting of the networks to capture topical and affective information only. The results show that the flow of the affect information has a weak performance when used alone; this emphasizes that affective information of a news article is a meaningful, yet complementary source of information.

Performance on Multiple Datasets. In Table 3 we compare the performance of the FakeFlow model to SOTA results on the other datasets we introduced in Section 4. The TruthShades dataset has two test sets, in-domain and out-of-domain. In the in-domain configuration, training and test articles come from the same sources, and from different sources in out-of-domain configuration. The results demonstrate that FakeFlow achieves a better

---

[10] We only compare TopicAgnostic on the dataset the authors proposed (PoliticalNews).

[11] In the case of N=1 in Figure 3, we set the maximum segment length to 1500 words instead of 800 to not lose parts of the longer articles.

TruthShades	Acc.	Prec.	Rec.	$\mathrm{F1}_{macro}$
Out-of-domain				
Rashkin et al. (2017)	0.67	0.70	0.67	0.65
FakeFlow	0.68	0.69	0.68	**0.68**
In-domain				
Rashkin et al. (2017)	0.91	0.91	0.91	0.91
FakeFlow	0.96	0.96	0.96	**0.96**

PoliticalNews	Acc.	Prec.	Rec.	$\mathrm{F1}_{weighted}$
TopicAgnostic	0.87	0.87	0.87	0.87
FakeFlow	0.88	0.88	0.88	**0.88**

FakeNewsNet	Acc.	Prec.	Rec.	$\mathrm{F1}_{weighted}$
FakeNewsTracker	0.80	0.82	0.75	0.79
One-Hot LR	0.82	0.90	0.72	0.80
FakeFlow	0.86	0.86	0.86	**0.85**

Table 3: Results on multiple datasets. We compare the FakeFlow model to SOTA models on each dataset.

F1 on both test sets. In a similar way, the results on the PoliticalNews dataset show that FakeFlow also outperforms the TopicAgnostic model, although the gap in results is not very large. Finally, regarding the FakeNewsNet dataset, it looks that the deep learning-based model (FakeNewsTracker) does not achieve a good performance comparing to the other proposed baseline by the authors, which is a Logistic Regression (LR) classifier with one-hot vectors of the news articles' text. Furthermore, it seems that a simple word-based model works better than a more sophisticated model that incorporates social media and context information. The FakeFlow model, on the other hand, achieves a better result, outperforming both the FakeNewsTracker and the LR baseline.

Topic-Aware Model. Constantly, new events are covered by news agencies. These events are different from the old ones in terms of discourse and topic. Therefore, a fake news detector trained on news articles from years back is unable to detect recent news. In this experiment, we are evaluating our approach on the PoliticalNews dataset that is constructed from news distributed across different years (2013 to 2018). Following the experimental setup in (Castelo et al., 2019), we train the FakeFlow model on news from one year and test on the other years, one year at a time for testing. For example, we train the model on news from 2013 and we test on news from 2015. Note that each test set is associated with 5 results, one for each year. Figure 4 shows the average accuracy for each test set. We compare FakeFlow to the TopicAgnostic model that proved to be effective at detecting fake news from different years. It is worth mentioning that the features of the TopicAgnostic model have been extracted from both headlines and text of the news articles. However, the results show that both models have a similar performance, except for the 2013 test set where FakeFlow achieves a higher accuracy with a difference of 7%. The experiment shows that FakeFlow is capable of detecting fake news from different years, with a flat performance across the years.

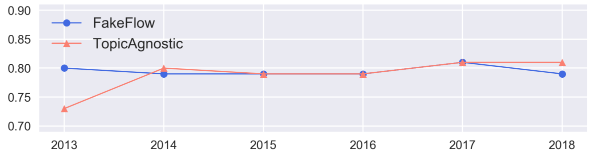


Figure 4: Topic aware experiment's results.

Attention Weights. The proposed FakeFlow model shows that taking into account the flow of affective information in fake news is an important perspective for fake news detection. We argue that being able to better understand the behaviour of the model can make it more transparent to the end-users. Figure 5 illustrates this by showing the attention weights of a fake news article across the 10 segments (left bar).[12] The figure shows that FakeFlow attends more to the beginning of the article. For better understanding, we match the affective information with the attention weights. Regarding the news text in the figure, the *emotions* features[13] show a clear example of how fake news articles try to manipulate the reader. It looks as if the existence of *fear, sadness*, and *surprise emotions* at the beginning of the article have triggered the attention on this part. Towards the end of the article, on the other hand, we can notice that such negative emotions do not exist, while *emotions* like *joy* and *anticipation* appear. This exemplifies how fake news try to attract the readers' attention in the first part of the text. Regarding the *morality* features, we only match the word "kill" with the *harm* category. Also, for the *hyperbolic* feature, we match the words "terrifying" and "powerful". In the same manner, both *morality* and *hyperbolic* features match words that occur at the beginning of the article. Lastly, for both *sentiment* and *im-*

---

[12] We averaged the attention weight matrix along the timesteps (number of segments) representations.

[13] Words with multiple colors mean that they have been annotated with multiple emotion types in the NRC lexicon.

ageability features, we are not able to find a clear interpretation in this example where many words across the segments match.

Real vs. Fake Analysis. In Table 4 we present an analysis on both real and fake news articles. The analysis gives an intuition to the reader on the distribution of the used features across the articles' segments. It shows that an emotion like *fear* has on average a higher difference between the first and the last segment in fake news than in real ones (see Figure 6 for a visualized distribution). Also, a feature like *hyperbolic* has a higher average value and lower standard deviation across all segments for fake news than real news, thus indicating that fake news have a higher amount of hyperbolic words with similarly high values.

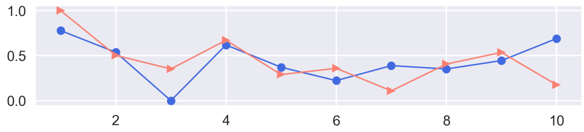


Figure 6: The flow of the *Fear* emotion in **fake** (►) and **real** (●) news articles in the MultiSourceFake dataset. Y-axis presents the average number of *Fear* emotion words in 0-1 scale; the X-axis presents the document text, divided into 10 segments.

7 Conclusion

In this paper we presented FakeFlow, a model that takes into account the flow of affective information (*emotions, sentiment, hyperbolic words*, etc.) in texts to better detect fake news articles. The model receives as input a text, segmented into smaller units, instead of processing one long fragment. This enables it to learn the flow of affective information by modeling the interaction between the topic and affective terms in the news article. We evaluated our model on four different datasets and compared it to several strong baselines. The extensive experiments show the effectiveness of FakeFlow over state-of-the-art models. Although FakeFlow was trained using a limited amount of text, the results demonstrated that it achieves results on-par with resource-hungry models (e.g. BERT and Longformer). In future work, we plan to extend our dataset and study more fine-grained news types, e.g. propaganda, from an emotional perspective. Moreover, we plan to investigate how we can replace the lexicon-based information with language-independent approaches in an attempt to make our model multilingual.

Acknowledgment

The first author would like to thank Ines Rehbein and Ana Uban for their valuable comments and suggestions. The work of the third author was partially funded by the Spanish MICINN under the research project MISMIS-FAKEnHATE on MISinformation and MIScommunication in social media: FAKE news and HATE speech (PGC2018-096212-B-C31) and by the Generalitat Valenciana under the research project DeepPattern (PROMETEO/2019/121).

References

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. 2019. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*.

Ahmet Aker, Leon Derczynski, and Kalina Bontcheva. 2017. Simple open stance classification for rumour analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 31–39.

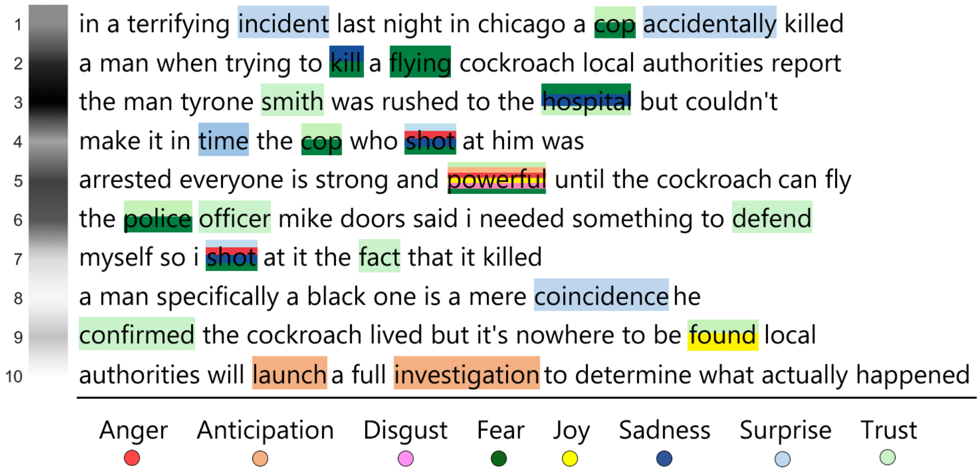
Alberto Barrón-Cedeno, Giovanni Da San Martino, Israa Jaradat, and Preslav Nakov. 2019. Proppy: A system to unmask propaganda in online news. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9847–9848.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The Long-document Transformer. *arXiv preprint arXiv:2004.05150*.

Sonia Castelo, Thais Almeida, Anas Elghafari, Aécio Santos, Kien Pham, Eduardo Nakamura, and Juliana Freire. 2019. A Topic-Agnostic Approach for Identifying Fake News Pages. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 975–980.

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop Clickbait: Detecting and Preventing Clickbaits in Online News Media. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 9–16.

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



in a terrifying incident last night in chicago a cop accidentally killed
a man when trying to kill a flying cockroach local authorities report
the man tyrone smith was rushed to the hospital but couldn't
make it in time the cop who shot at him was
arrested everyone is strong and powerful until the cockroach can fly
the police officer mike doors said i needed something to defend
myself so i shot at it the fact that it killed
a man specifically a black one is a mere coincidence he
confirmed the cockroach lived but it's nowhere to be found local
authorities will launch a full investigation to determine what actually happened

Figure 5: Emotional interpretation of a *fake* news article by showing the attention weights (the bar on the left) and highlighting the emotions in the text.

	Features	**Real News**				**Fake News**			
		$\mu\ first_{seg.}$	$\mu\ last_{seg.}$	$\mu\ all_{seg.}$	$\sigma\ all_{seg.}$	$\mu\ first_{seg.}$	$\mu\ last_{seg.}$	$\mu\ all_{seg.}$	$\sigma\ all_{seg.}$
Emotions	Anger	0.175	0.167	0.170	0.003	0.183	0.170	0.171	0.008
	Anticipation	0.301	0.315	0.264	0.025	0.293	0.305	0.260	0.022
	Disgust	0.095	0.101	0.095	0.004	0.096	0.091	0.091	0.007
	Fear	0.254	0.250	0.238	0.010	0.265	0.226	0.238	0.011
	Joy	0.217	0.226	0.183	0.021	0.207	0.203	0.175	0.020
	Sadness	0.161	0.158	0.160	0.006	0.155	0.155	0.158	0.007
	Surprise	0.140	0.144	0.123	0.012	0.142	0.123	0.120	0.008
	Trust	0.446	0.466	0.400	0.031	0.461	0.421	0.401	0.029
Senti.	Positive	0.599	0.623	0.558	0.030	0.608	0.591	0.554	0.032
	Negative	0.369	0.337	0.347	0.011	0.367	0.336	0.350	0.013
Morality	Harm	0.007	0.011	0.007	0.002	0.008	0.013	0.007	0.002
	Care	0.026	0.023	0.019	0.004	0.021	0.022	0.019	0.003
	Fairness	0.003	0.013	0.007	0.002	0.005	0.020	0.009	0.004
	Unfairness	0.000	0.000	0.001	0.000	0.001	0.000	0.001	0.001
	Loyalty	0.016	0.017	0.019	0.002	0.014	0.016	0.019	0.003
	Betrayal	0.004	0.003	0.005	0.001	0.002	0.003	0.004	0.001
	Authority	0.025	0.032	0.026	0.003	0.024	0.028	0.026	0.002
	Subversion	0.005	0.004	0.004	0.001	0.006	0.007	0.005	0.002
	Sanctity	0.005	0.005	0.004	0.001	0.005	0.006	0.005	0.002
	Degradation	0.003	0.004	0.003	0.001	0.006	0.004	0.003	0.001
Img	Imageability	0.845	1.203	1.144	0.122	0.877	1.184	1.145	0.124
	Abstraction	0.424	0.331	0.352	0.028	0.382	0.304	0.342	0.037
	Hyperbolic	0.042	0.05	0.045	0.005	0.046	0.044	0.047	0.003

Table 4: A quantitative analysis of the features existence across articles' segments. We present the average value in the first segment ($\mu\ first_{seg.}$), the average value in the last segment ($\mu\ last_{seg.}$), the average value in the all 10 segments ($\mu\ all_{seg.}$), and the standard deviation ($\sigma\ all_{seg.}$) of a feature across the 10 segments, both in real and fake news.

]

Bilal Ghanem, Alessandra Teresa Cignarella, Cristina Bosco, Paolo Rosso, and Francisco Manuel Rangel Pardo. 2019. UPV-28-UNITO at SemEval-2019 Task 7: Exploiting Post's Nesting and Syntax Information for Rumor Stance Classification. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1125–1131.

Bilal Ghanem, Paolo Rosso, and Francisco Rangel. 2020. An emotional analysis of false information in social media and news articles. *ACM Transactions on Internet Technology (TOIT)*, 20(2):1–18.

Jesse Graham, Jonathan Haidt, and Brian A Nosek. 2009. Liberals and Conservatives Rely on Different Sets of Moral Foundations. *Journal of personality and social psychology*, 96(5):1029–1046.

Benjamin D Horne and Sibel Adali. 2017. This Just In: Fake News Packs a Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire than Real News. In *Eleventh International AAAI Conference on Web and Social Media*, pages 759–766.

Sudipta Kar, Suraj Maharjan, and Thamar Solorio. 2018. Folksonomication: Predicting tags for movies from plot synopses using emotion flow encoded neural network. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2879–2891.

Georgi Karadzhov, Preslav Nakov, Lluís Màrquez, Alberto Barrón-Cedeño, and Ivan Koychev. 2017. Fully automated fact checking using external sources. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 344–353.

Suraj Maharjan, Sudipta Kar, Manuel Montes-y Gómez, Fabio A Gonzalez, and Thamar Solorio. 2018. Letting emotions flow: Success prediction by modeling the flow of emotions in books. *arXiv preprint arXiv:1805.09746*.

Saif M Mohammad and Peter D Turney. 2010. Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, pages 26–34. Association for Computational Linguistics.

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. 2018. Automatic Detection of Fake News. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3391–3401, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937.

Andrew J Reagan, Lewis Mitchell, Dilan Kiley, Christopher M Danforth, and Peter Sheridan Dodds. 2016. The emotional arcs of stories are dominated by six basic shapes. *EPJ Data Science*, 5(1):1–12.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019a. defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 395–405.

Kai Shu, Deepak Mahudeswaran, and Huan Liu. 2019b. FakeNewsTracker: a Tool for Fake News Collection, Detection, and Visualization. *Computational and Mathematical Organization Theory*, 25(1):60–71.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2018. FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media. *arXiv preprint arXiv:1809.01286*.

Yla R Tausczik and James W Pennebaker. 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of language and social psychology*, 29(1):24–54.

Michael Wilson. 1988. MRC Psycholinguistic Database: Machine-usable dictionary, version 2.00. *Behavior research methods, instruments, & computers*, 20(1):6–10.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open Attribute Value Extraction from Product Profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1049–1058.

Xinyi Zhou and Reza Zafarani. 2018. Fake news: A survey of research, detection methods, and opportunities. *arXiv preprint arXiv:1812.00315*.

Dimitrina Zlatkova, Preslav Nakov, and Ivan Koychev. 2019. Fact-checking meets fauxtography: Verifying claims about images. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2099–2108.

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Kalina Bontcheva, and Peter Tolmie. 2015. Towards Detecting Rumours in Social Media. In *AAAI Workshop: AI for Cities*, pages 35–41.

A Appendices

A.1 Hyper-parameters

For FakeFlow hyper-parameters, we tune the following parameters with the their correspondent search space:

- Dropout: random selection in the range [0.1, 0.6],
- Dense layers: [8, 16, 32, 64, 128],
- Activation functions: [selu, relu, tanh, elu],
- CNN filters' sizes: [(2, 3, 4), (3, 4, 5), (4, 5, 6), (3, 5), (2, 4), (4,), (5,), (3, 5, 7), (3, 6)],
- Numbers of CNN filters: [4, 8, 16, 32, 64, 128],
- Pooling size: [2, 3],
- GRU units: [8, 16, 32, 64, 128],
- Optimization function: [adam, adadelta, rmsprop, sgd],
- For the early stopping, we set the 'patience' parameter to 4 and we set the epochs number to 50.

For the parameters selection, we use hyperopt[14] library that receives the above search space to randomly select different N combination of parameters (trials). We use a small value of N in all of our experiments to avoid overdrawn finetuning; we set N to 35.

A.2 Topic Aware experiments

In Figure 4, we present the average accuracy of our model when we train on different years and test a specific one. In the following we show the results before we averaged them.

Train \ Test	2013	2014	2015	2016	2017	2018
2013	0.00	0.82	0.74	0.76	0.78	0.74
2014	0.84	0.00	0.79	0.76	0.81	0.74
2015	0.79	0.81	0.00	0.82	0.80	0.82
2016	0.80	0.76	0.87	0.00	0.85	0.79
2017	0.79	0.82	0.76	0.80	0.00	0.85
2018	0.79	0.75	0.81	0.83	0.83	0.00
Average	0.80	0.79	0.79	0.79	0.81	0.79

Table 5: FakeFlow results for each train-test run for the Topic-Aware experiment.

---

[14] https://github.com/hyperopt/hyperopt

CTC-based Compression for Direct Speech Translation

Marco Gaido†, *, **Mauro Cettolo**†, **Matteo Negri**†, **Marco Turchi**†
†Fondazione Bruno Kessler
*University of Trento
`{mgaido,cettolo,negri,turchi}@fbk.eu`

Abstract

Previous studies demonstrated that a dynamic phone-informed compression of the input audio is beneficial for speech translation (ST). However, they required a dedicated model for phone recognition and did not test this solution for direct ST, in which a single model translates the input audio into the target language without intermediate representations. In this work, we propose the first method able to perform a dynamic compression of the input in direct ST models. In particular, we exploit the Connectionist Temporal Classification (CTC) to compress the input sequence according to its phonetic characteristics. Our experiments demonstrate that our solution brings a 1.3-1.5 BLEU improvement over a strong baseline on two language pairs (English-Italian and English-German), contextually reducing the memory footprint by more than 10%.

1 Introduction

Speech translation (ST) is the process that converts utterances in one language into text in another language. Traditional approaches to ST consist of separate modules, each dedicated to an easier subtask, which are eventually integrated in a so-called *cascade* architecture (Stentiford and Steer, 1988; Waibel et al., 1991). Usually, its main components are an automatic speech recognition (ASR) model - which generates the transcripts from the audio - and a machine translation (MT) model - which translates the transcripts into the target language. A newer approach is *direct* ST, in which a single model performs the whole task without intermediate representations (Bérard et al., 2016; Weiss et al., 2017). The main advantages of direct ST systems are: *i)* the access to information not present in the text (e.g. prosody, vocal characteristics of the speaker) during the translation phase, *ii)* a reduced latency, *iii)* a simpler and easier to manage architecture (only one model has to be maintained), which *iv)* avoids error propagation across components.

In both paradigms (cascade and direct), the audio is commonly represented as a sequence of vectors obtained with a Mel filter bank. These vectors are collected with a high frequency, typically one every 10 ms. The resulting sequences are much longer than the corresponding textual ones (usually by a factor of ~10). The sequence length is problematic both for RNN (Elman, 1990) and Transformer (Vaswani et al., 2017) architectures. Indeed, RNNs fail to represent long-range dependencies (Bengio et al., 1993) and the Transformer has a quadratic memory complexity in the input sequence length, which makes training on long sequences prohibitive due to its memory footprint. For this reason, architectures proposed for direct ST/ASR reduce the input length either with convolutional layers (Bérard et al., 2018; Di Gangi et al., 2019) or by stacking and downsampling consecutive samples (Sak et al., 2015). However, these fixed-length reductions of the input sequence assume that samples carry the same amount of information. This does not necessarily hold true, as phonetic features vary at a different speed in time and frequency in the audio signals.

Consequently, researchers have studied how to reduce the input length according to dynamic criteria based on the audio content. Salesky et al. (2019) demonstrated that a phoneme-based compression of the input frames yields significant gains compared to fixed length reduction. Phone-based and linguistically-informed compression also proved to be useful in the context of visually grounded speech (Havard et al., 2020). However, Salesky and Black (2020) questioned the approach, claiming that the addition of phone features without segmentation and compression of the input is more effective.

None of these works is a direct ST solution, as they all require a separate model for phone recog-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 690–696
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

nition and intermediate representations. So, they: *i)* are affected by *error propagation* (Salesky and Black 2020 show in fact that lower quality in phone recognition significantly degrades final ST performance), *ii)* have higher latency and *iii)* a more complex architecture. A direct model with phone-based multi-task training was introduced by Jia et al. (2019) for speech-to-speech translation, but they neither compared with a training using transcripts nor investigated dynamic compression.

In this paper, we explore the usage of phones and dynamic content-based input compression for direct ST (and ASR). Our goal is an input reduction that, limiting the amount of redundant/useless information, yields better performance and lower memory consumption at the same time. To this aim, we propose to exploit the Connectionist Temporal Classification (CTC) (Graves et al., 2006) to add phones prediction in a multi-task training and compress the sequence accordingly. To disentangle the contribution of the introduction of phone recognition and the compression based on it, we compare against similar trainings leveraging transcripts instead of phones. Our results show that phone-based multi-task training with sequence compression improves over a strong baseline by up to 1.5 BLEU points on two language pairs (English-German and English-Italian), with a memory footprint reduction of at least 10%.

2 CTC-based Sequence Compression

The CTC algorithm is usually employed for training a model to predict an output sequence of variable length that is shorter than the input one. This is the case of speech/phone recognition, as the input is a long sequence of audio samples, while the output is the sequence of uttered symbols (e.g. phones, sub-words), which is significantly shorter. In particular, for each time step, the CTC produces a probability distribution over the possible target labels augmented with a dedicated `<blank>` symbol representing the absence of a target value. These distributions are then exploited to compute the probabilities of different sequences, in which consecutive equal predictions are collapsed and `<blank>` symbols are removed. Finally, the resulting sequences are compared with the target sequence.

Adding an auxiliary CTC loss to the training of direct ST and acoustic ASR models has been shown to improve performance (Kim et al., 2017; Bahar et al., 2019). In these works, the CTC loss is computed against the transcripts on the encoder output to favour model convergence. Generally, the CTC loss can be added to the output of any encoder layer, as in Figure 1 where the hyper-parameter N_{CTC} indicates the number of the layer at which the CTC is computed. Formally, the final loss function is:

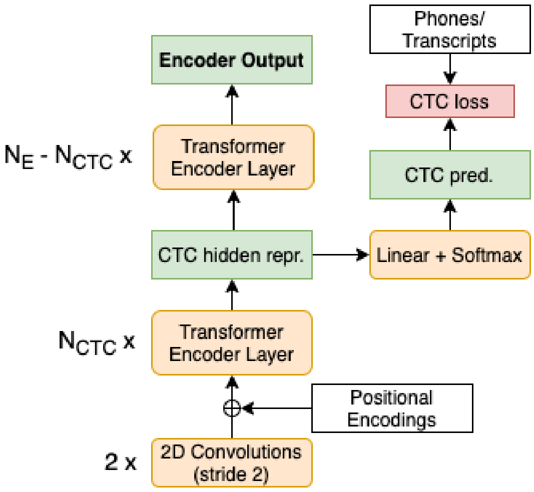


Figure 1: Encoder architecture with CTC loss.

$$\lambda = CTC(E_{\mathrm{N_{CTC}}}) + CE(D_{\mathrm{N_D}}) \qquad (1)$$

where E_{x} is the output of the x-th encoder layer, $D_{\mathrm{N_D}}$ is the decoder output, CTC is the CTC function, and CE is the label smoothed cross entropy. If N_{CTC} is equal to the number of encoder layers (N_{E}), the CTC input is the encoder output. We consider this solution as our baseline and we also test it with phones as target.

As shown in Figure 1, we use as model a Transformer, whose encoder layers are preceded by two 2D convolutional layers that reduce the input size by a factor of 4. Therefore, the CTC produces a prediction every 4 input time frames. The sequence length reduction is necessary both because it makes possible the training (otherwise out of memory errors would occur) and to have a fair comparison with modern state-of-the-art models. A logarithmic distance penalty (Di Gangi et al., 2019) is added to all the Transformer encoder layers.

Our proposed architecture is represented in Figure 2. The difference with the baseline is the introduction of an additional block (*Collapse same predictions*) that exploits the CTC predictions to compress the input elements (vectors). Hence, in this case the CTC does not only help model convergence, but it also defines variable-length segments representing the same content. So, dense audio portions can be given more importance, while re-

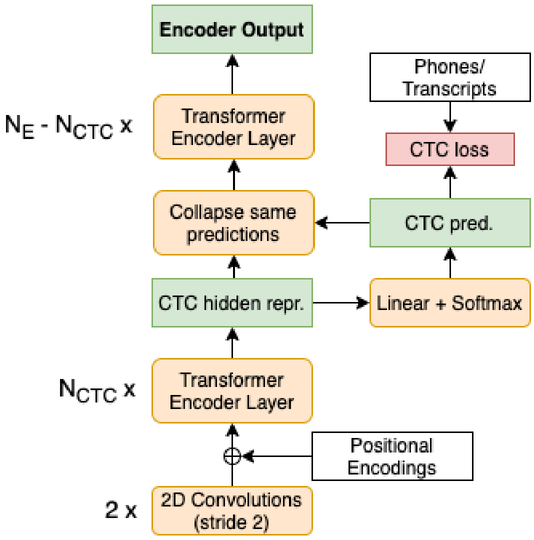


Figure 2: Encoder architecture with CTC compression.

dundant/uninformative vectors can be compressed. This allows the following encoder layers and the decoder to attend to useful information without being "distracted" by noisy elements. The architecture is a direct ST solution as there is a single model whose parameters are optimized together without intermediate representations. At inference time, the only input is the audio and the model produces the translation into the target language (contextually generating the transcripts/phones with the CTC).

We compare three techniques to compress the consecutive vectors with the same CTC prediction:

- **Average.** The vectors to be collapsed together are averaged. As there is only a linear layer between the CTC inputs and its predictions, the vectors in each group are likely to be similar, so the compression should not remove much information.

- **Weighted.** The vectors are averaged but the weight of each vector depends on the confidence (i.e. the predicted probability) of the CTC prediction. This solution is meant to give less importance to vectors whose phone/transcript is not certain.

- **Softmax.** In this case, the weight of each vector is obtained by computing the `softmax` of the CTC predicted probabilities. The idea is to propagate information (nearly) only through a single input vector (the more confident one) for each group.

3 Data

We experiment with MuST-C (Cattoni et al., 2021), a multilingual ST corpus built from TED talks. We focus on the English-Italian (465 hours) and English-German (408 hours) sections. For each set (train, validation, test), it contains the audio files, the transcripts, the translations and a YAML file with the start time and duration of the segments.

In addition, we extract the phones using Gentle.[1] Besides aligning the transcripts with the audio, Gentle returns the start and end time for each recognized word, together with the corresponding phones. For the words not recognized in the audio, Gentle does not provide the phones, so we lookup their phonetic transcription on the VoxForge[2] dictionary. For each sample in the corpus, we rely on the YAML file and the alignments generated by Gentle to get all the words (and phones) belonging to it. The phones have a suffix indicating the position in a word (at the end, at the beginning, in the middle or standalone). We also generated a version without the suffix (we refer to it as `PH W/O POS` in the rest of the paper). The resulting dictionaries contain respectively 144 and 48 symbols.

4 Experimental Settings

Our Transformer layers have 8 attention heads, 512 features for the attention and 2,048 hidden units in FFN. We set a 0.2 dropout and include *SpecAugment* (Park et al., 2019) in our trainings. We optimize label smoothed cross entropy (Szegedy et al., 2016) with 0.1 smoothing factor using Adam (Kingma and Ba, 2015) (betas *(0.9, 0.98)*). The learning rate increases linearly from 3e-4 to 5e-3 for 4,000 updates, then decays with the inverse square root. As we train on 8 GPUs with mini-batches of 8 sentences and we update the model every 8 steps, the resulting batch size is 512. The audio is pre-processed performing speaker normalization and extracting 40-channel Mel filter-bank features per frame. The text is tokenized into subwords with 1,000 BPE merge rules (Sennrich et al., 2016).

As having more encoder layers than decoder layers has been shown to be beneficial (Potapczyk and Przybysz, 2020; Gaido et al., 2020), we use 8 Transformer encoder layers and 6 decoder layers for ASR and 11 encoder and 4 decoder layers for ST unless stated otherwise. We train until the

---

[1] `https://lowerquality.com/gentle/`
[2] `http://www.voxforge.org/home`

model does not improve on the validation set for 5 epochs and we average the last 5 checkpoints. Trainings were performed on K80 GPUs and lasted ~48 hours (~50 minutes per epoch). Our implementation[3] is based on Fairseq (Ott et al., 2019).

We evaluate performance with WER for ASR and with BLEU (Papineni et al., 2002)[4] and SacreBLEU (Post, 2018)[5] for ST.

	WER (↓)	**RAM (MB)**
Baseline - 8L EN	16.0	6929 (1.00)
8L PH	**15.6**	6661 (0.96)
2L PH AVG	21.2	3375 (0.49)
4L PH AVG	17.5	4542 (0.66)
8L PH AVG	16.3	6286 (0.91)
8L PH W/O POS. AVG	16.4	6565 (0.95)
8L EN AVG	16.3	6068 (0.88)

Table 1: Results on ASR using the CTC loss with transcripts and phones as target. `AVG` indicates that sequence is compressed averaging the vectors.

5 Results

5.1 ASR

We first tested whether ASR benefits from the usage of phones and sequence compression. Table 1 shows that having phones instead of English transcripts (Baseline - 8L EN) as target of the CTC loss (`8L PH`) without compression is beneficial. When compressing the sequence, there is little difference according to the target used (`8L PH AVG`, `8L PH W/O POS. AVG`, `8L EN AVG`). However, the compression causes a 0.3-0.5 WER performance degradation and a 12-5% saving of RAM. Moving the compression to previous layers (`4L PH AVG`, `2L PH AVG`) further decreases the output quality and the RAM usage. We can conclude that compressing the input sequence harms ASR performance, but might be useful if RAM usage is critical and should be traded off with performance.

5.2 Direct ST

In early experiments, we pre-trained the first 8 layers of the ST encoder with that of the ASR model, adding three *adapter* layers (Bahar et al., 2019). We realized that ASR pre-training was not useful (probably because ASR and ST data are the same), so we report results without pre-training.

As we want to ensure that our results are not biased by a poor baseline, we compare with (Di Gangi et al., 2020), which uses the same framework and similar settings.[6] As shown in Table 2, our strong baseline (`8L EN`) outperforms (Di Gangi et al., 2020) by 2 BLEU on en-it and 1.3 BLEU on en-de.

As in ASR, replacing the transcripts with phones as target for the CTC loss (`8L PH`) further improves respectively by 0.5 and 1.2 BLEU. We first explore the introduction of the compression at different layers. Adding it to the 8th layer (`8L PH AVG`) enhances the translation quality by 0.6 (en-it) and 0.2 (en-de) BLEU, with the improvement on en-it being statistically significant over the version without CTC compression. Moving it to previous layers (`4L PH AVG`, `2L PH AVG`) causes performance drops, suggesting that many layers are needed to extract useful phonetic information.

Then, we compare the different compression policies: `AVG` outperforms (or matches) `WEIGHTED` and `SOFTMAX` on both languages. Indeed, the small weight these two methods assign to some vectors likely causes an information loss and prevents a proper gradient propagation for the corresponding input elements.

Finally, we experiment with different CTC targets, but both the phones without the position suffix (`8L PH W/O POS. AVG`) and the transcripts (`8L EN AVG`) lead to lower scores.

The different results between ASR and ST can be explained by the nature of the two tasks: extracting content knowledge is critical for ST but not for ASR, in which a compression can hide details that are not relevant to extrapolate meaning, but needed to generate precise transcripts. The RAM savings are higher than in ASR as there are 3 more layers. On the 8th layer, they range from 11% to 23% for en-it, 16% to 22% for en-de. By moving the compression to previous layers, we can trade performance for RAM requirements, saving up to 50% of the memory.

We also tested whether we can use the saved RAM to add more layers and improve the translation quality. We added 3 encoder and 2 decoder layers: this (`8L PH AVG (14+6L)`) results in

---

[3] Available at `https://github.com/mgaido91/FBK-fairseq-ST/tree/eacl2021`.

[4] To be comparable with previous works.

[5] The version signature is: `BLEU+c.mixed+#.1+s.exp+tok.13a+v.1.4.3`.

[6] We acknowledge that better results have been published in a contemporaneous paper by Inaguma et al. (2020). Besides the contemporaneity issue, our results are not comparable with theirs, as they use: *i)* a different architecture built on ESPnet-ST (a newer framework that, alone, outperforms Fairseq), *ii)* higher dimensional input features (83 vs 40 dimensions), *iii)* data augmentation, and *iv)* pre-training techniques.

	en-it			en-de		
	BLEU (↑)	**SacreBLEU (↑)**	**RAM (MB)**	**BLEU (↑)**	**SacreBLEU (↑)**	**RAM (MB)**
(Di Gangi et al., 2020)	20.1	-	-	19.1	-	-
Baseline - 8L EN	22.1	21.8	9624 (1.00)	20.4	20.5	9166 (1.00)
8L PH	22.6*	22.3*	9567 (0.99)	21.6*	21.6*	9190 (1.00)
2L PH AVG	20.2	20.0	5804 (0.60)	17.8	17.8	4484 (0.49)
4L PH AVG	21.6	21.3	6193 (0.64)	20.1	20.2	5186 (0.57)
8L PH AVG	*23.2*†	*22.8*†	8554 (0.89)	*21.8**	*21.9**	7348 (0.80)
8L PH WEIGHTED	22.7*	22.5*	7636 (0.79)	21.7*	21.8*	7380 (0.81)
8L PH SOFTMAX	22.6*	22.3*	7892 (0.82)	*21.8**	*21.9**	7436 (0.81)
8L PH W/O POS. AVG	22.2	22.0	7451 (0.77)	21.5*	21.6*	7274 (0.79)
8L EN AVG	22.2	21.9	8287 (0.86)	20.6	20.7	7143 (0.78)
8L PH AVG (14+6L)	**23.4**†	**23.2**†	8658 (0.90)	**21.9**†	**22.0**†	7719 (0.84)

Table 2: Results using the CTC loss with transcripts and phones as target. `AVG`, `WEIGHTED` and `SOFTMAX` indicate the compression method. If none is specified, no compression is performed. The symbol "*" indicates improvements that are statistically significant with respect to the baseline. "†" indicates statistically significant gains with respect to `8L PH`. Statistical significance is computed according to (Koehn, 2004) with $\alpha = 0.05$. Scores in *italic* indicate the best models among those with equal number of layers.

small gains (0.2 on en-it and 0.1 on en-de), but the additional memory required is also small (the RAM usage is still 10-16% lower than the baseline). The improvements are statistically significant with respect to the models without compression (`8L PH`) on both language pairs. When training on more data, the benefit of having deeper networks might be higher, though, and this solution allows to increase the number of layers without a prohibitive memory footprint. We leave this investigation for future works, as experiments on larger training corpora are out of the scope of this paper.

6 Conclusions

As researchers' focus is shifting from cascade to direct solutions due to the advantages of the latter, we proposed a technique of dynamic sequence-length reduction for direct ST. We showed that averaging the vectors corresponding to the same phone prediction according to the CTC improves the translation quality and reduces the memory footprint, allowing for training deeper models. Our best model outperforms a strong baseline, which uses transcripts in a multi-task training, by 1.3 (en-it) and 1.5 (en-de) BLEU, reducing memory usage by 10-16%.

Acknowledgments

This work is part of the "End-to-end Spoken Language Translation in Rich Data Conditions" project,[7] which is financially supported by an Amazon AWS ML Grant. The authors also wish to thank Mattia Antonino Di Gangi for the insightful discussions on this work.

References

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A Comparative Study on End-to-end Speech to Text Translation. In *Proceedings of International Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 792–799, Sentosa, Singapore.

Yoshua Bengio, Paolo Frasconi, and Patrice Y. Simard. 1993. The problem of learning long-term dependencies in recurrent networks. In *IEEE International Conference on Neural Networks*, pages 1183–1188 vol.3.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-End Automatic Speech Translation of Audiobooks. In *Proceedings of ICASSP 2018 - IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Alberta, Canada.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and Translate: A Proof of Concept for End-to-End Speech-to-Text Translation. In *NIPS Workshop on end-to-end learning for speech and audio processing*, Barcelona, Spain.

Roldano Cattoni, Mattia A. Di Gangi, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2021. MuST-C: A multilingual corpus for end-to-end speech translation. *Computer Speech & Language*, 66:101155.

Mattia A. Di Gangi, Marco Gaido, Matteo Negri, and Marco Turchi. 2020. On Target Segmentation for Direct Speech Translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (AMTA 2020)*, pages 137–150,

---

[7] `https://ict.fbk.eu/units-hlt-mt-e2eslt/`

Virtual. Association for Machine Translation in the Americas.

Mattia A. Di Gangi, Matteo Negri, Roldano Cattoni, Roberto Dessi, and Marco Turchi. 2019. Enhancing Transformer for End-to-end Speech-to-Text Translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 21–31, Dublin, Ireland. European Association for Machine Translation.

Jeffrey L. Elman. 1990. Finding structure in time. *COGNITIVE SCIENCE*, 14(2):179–211.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-End Speech-Translation with Knowledge Distillation: FBK@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 80–88, Online. Association for Computational Linguistics.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, pages 369–376, Pittsburgh, Pennsylvania.

William N. Havard, Jean-Pierre Chevrot, and Laurent Besacier. 2020. Catplayinginthesnow: Impact of Prior Segmentation on a Model of Visually Grounded Speech.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-One Speech Translation Toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Ye Jia, Ron J. Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. 2019. Direct Speech-to-Speech Translation with a Sequence-to-Sequence Model. In *Proceedings of Interspeech 2019*, pages 1123–1127, Graz, Austria.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4835–4839, New Orleans, Louisiana.

Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of 3rd International Conference on Learning Representations (ICLR)*, San Diego, California.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania. Association for Computational Linguistics.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proceedings of Interspeech 2019*, pages 2613–2617, Graz, Austria.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Tomasz Potapczyk and Pawel Przybysz. 2020. SRPOL's system for the IWSLT 2020 end-to-end speech translation task. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 89–94, Online. Association for Computational Linguistics.

Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. 2015. Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition. In *Proceedings of Interspeech 2015*, Dresden, Germany.

Elizabeth Salesky and Alan W. Black. 2020. Phone Features Improve Speech Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2388–2397, Online. Association for Computational Linguistics.

Elizabeth Salesky, Matthias Sperber, and Alan W. Black. 2019. Exploring Phoneme-Level Speech Representations for End-to-End Speech Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Frederick W. M. Stentiford and Martin G. Steer. 1988. Machine Translation of Speech. *British Telecom Technology Journal*, 6(2):116–122.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Las Vegas, Nevada, United States.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5998–6008, Long Beach, California.

Alex Waibel, Ajay N. Jain, Arthur E. McNair, Hiroaki Saito, Alexander G. Hauptmann, and Joe Tebelskis. 1991. JANUS: A Speech-to-Speech Translation System Using Connectionist and Symbolic Processing Strategies. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP 1991*, pages 793–796, Toronto, Canada.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-Sequence Models Can Directly Translate Foreign Speech. In *Proceedings of Interspeech 2017*, pages 2625–2629, Stockholm, Sweden.

A Crowdsourced Open-Source Kazakh Speech Corpus and Initial Speech Recognition Baseline

Yerbolat Khassanov, Saida Mussakhojayeva, Almas Mirzakhmetov,
Alen Adiyev, Mukhamet Nurpeiissov and Huseyin Atakan Varol
Institute of Smart Systems and Artificial Intelligence (ISSAI),
Nazarbayev University, Nur-Sultan, Kazakhstan
`{yerbolat.khassanov, saida.mussakhojayeva, almas.mirzakhmetov,`
`alen.adiyev, mukhamet.nurpeiissov, ahvarol}@nu.edu.kz`

Abstract

We present an open-source speech corpus for the Kazakh language. The Kazakh speech corpus (KSC) contains around 332 hours of transcribed audio comprising over 153,000 utterances spoken by participants from different regions and age groups, as well as both genders. It was carefully inspected by native Kazakh speakers to ensure high quality. The KSC is the largest publicly available database developed to advance various Kazakh speech and language processing applications. In this paper, we first describe the data collection and preprocessing procedures followed by a description of the database specifications. We also share our experience and challenges faced during the database construction, which might benefit other researchers planning to build a speech corpus for a low-resource language. To demonstrate the reliability of the database, we performed preliminary speech recognition experiments. The experimental results imply that the quality of audio and transcripts is promising (2.8% character error rate and 8.7% word error rate on the test set). To enable experiment reproducibility and ease the corpus usage, we also released an ESPnet recipe for our speech recognition models.

1 Introduction

We present an open-source Kazakh speech corpus (KSC) constructed to advance the development of speech and language processing applications for the Kazakh language. Kazakh is an agglutinative language with vowel harmony and belongs to the family of Turkic languages. During the Soviet period, the Kazakh language was overwhelmed by the Russian language, which caused a decline in Kazakh language usage (Dave, 2007). In the 1990s, it was declared an official language of Kazakhstan, and many initiatives were launched to increase the number of Kazakh speakers. Today, it is spoken by over 10 million people in Kazakhstan and by over 3 million people in other countries[1]. By introducing the KSC, we aim to accelerate the penetration of the Kazakh language into the Internet of things (IoT) technologies and to promote research in Kazakh speech processing applications.

Although several Kazakh speech corpora have been presented (Makhambetov et al., 2013; Shi et al., 2017; Mamyrbayev et al., 2019), there is no generally accepted common corpus. Most of them are either publicly unavailable or contain an insufficient amount of data to train reliable models. Especially, these databases are too small for building recent end-to-end models, which are extremely data hungry (Hannun et al., 2014). Consequently, different research groups usually conduct their experiments on internally collected data, which prevents the reproducibility and comparison of different approaches.

To address the aforementioned limitations, we created the KSC, containing around 332 hours of transcribed audio. It was crowdsourced through the Internet, where volunteers were asked to read sentences presented through a web browser. In total, we accepted over 153,000 utterances submitted from over 1,600 unique device IDs. The recordings were first checked manually, and when a sufficient amount of data was collected, were partially checked automatically using a speech recognition system. To the best of our knowledge, the KSC is the largest open-source speech corpus in Kazakh and is available for academic and commercial use upon request at this link[2] under the Creative Commons Attribution 4.0 International License[3].

---

[1] `https://www.ethnologue.com/language/kaz`

[2] `https://issai.nu.edu.kz/kz-speech-corpus/?version=1.1`

[3] `https://creativecommons.org/licenses/by/4.0/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 697–706
April 19 - 23, 2021.

We expect that this database will be a valuable resource for research communities in both academia and industry. The primary application domains of the corpus are speech recognition, speech synthesis, and speaker recognition. To demonstrate the reliability of the database, we performed preliminary automatic speech recognition (ASR) experiments, where promising and sufficient for practical usage results were achieved. We also provide a practical guide on the development of ASR systems for the Kazakh language by sharing the reproducible recipe and pretrained models[4]. The utilization of the database for speech synthesis and speaker recognition tasks is left for future work.

The rest of the paper is organized as follows. Section 2 provides a review of related works. Section 3 presents the KSC database and explains the database construction procedures in detail. Section 4 offer a presentation of the speech recognition experiment setup and obtained results. In Section 5, the obtained results and challenges are discussed. Lastly, Section 6 concludes this paper and mentions potential future work.

2 Related Works

In the past few years, the interest in ASR has surged by its new applications in smart devices, such as voice command, voice search, message dictation, and virtual assistants (Yu and Deng, 2014). In response to this technological shift, many speech corpora have been introduced for various languages. For example, Du et al. (2018) released 1,000 hours of open-source Mandarin read speech data to bridge the gap between academia and industry. The utterances were recorded using iOS-based mobile phones and cover the following domains: voice command, smart home, autonomous driving, and so on. Similarly, Koh et al. (2019) developed a 2,000-hour read speech corpus to help speech technology developers and researchers build speech-related applications in Singapore.

Several works have presented speech databases for the Kazakh language. For example, Makhambetov et al. (2013) developed a Kazakh language corpus containing around 40 hours of transcribed read speech data recorded in a sound-proof studio. Similarly, Mamyrbayev et al. (2019) collected 76 hours of data using a professional recording booth which were further extended to 123 hours in Mamyrbayev et al. (2020). Khomitsevich et al. (2015) utilized 147 hours of bilingual Kazakh-Russian speech data to build code-switching ASR systems. Shi et al. (2017) released 78 hours of transcribed Kazakh speech data recorded by 96 students from China. The IARPA Babel project has released a Kazakh language pack[5] consisting of around 50 hours of conversational and 14 hours of scripted telephone speech. Unfortunately, the aforementioned databases are either publicly unavailable or of an insufficient size to build robust Kazakh ASR systems. Additionally, some of them are nonrepresentative–that is, they cover speakers from a narrow set of categories, such as the same region or age group. Furthermore, since most of these databases have been collected in optimal lab settings, they might be ineffective for real-world applications.

The emergence of crowdsourcing platforms and the growth in Internet connectivity has motivated researchers to employ crowdsourcing for annotated corpora construction. Different from the expert-based approaches, crowdsourcing tends to be cheaper and faster, though, additional measures should be taken to ensure the data quality (Snow et al., 2008; Novotney and Callison-Burch, 2010; Eskenazi et al., 2013). Furthermore, crowdsourcing allows to gather a variety of dialects and accents from remote geographical locations and enables the participation of people with disabilities and of an advanced age, which otherwise would be impossible or too costly (Takamichi and Saruwatari, 2018). Inspired by this, we followed the best crowdsourcing practices to construct a large-scale and open-source speech corpus for the Kazakh language, as described in the following sections.

3 The KSC Construction

The KSC project was conducted with the approval of the Institutional Research Ethics Committee of Nazarbayev University. Each reader participated voluntarily and was informed of the data collection and use protocols through an online consent form.

3.1 Text Collection and Cleaning

We first extracted Kazakh textual data from various sources such as electronic books, laws, and websites, including Wikipedia, news portals, and blogs. For each website, we designed a specialized

---

[4] `https://github.com/IS2AI/ISSAI_SAIDA_Kazakh_ASR`

[5] `https://catalog.ldc.upenn.edu/LDC2018S13`

web crawler to improve the quality of the extracted text. The extracted texts were manually filtered to eliminate inappropriate content involving sensitive political issues, user privacy, violence, and so on. Additionally, we filtered out texts entirely consisting of Russian words. Texts consisting of mixed Kazakh-Russian utterances were kept, because there are many borrowed Russian words in Kazakh, and it is common practice among Kazakh speakers to code-switch between Kazakh and Russian (Khomitsevich et al., 2015). Next, we split the texts into sentences and removed sentences consisting of more than 25 words. Lastly, duplicate sentences were removed. The total number of extracted sentences was around 2.3 million.

3.2 Text Narration and Checking

To narrate the extracted sentences, we developed a web-based speech recording platform capable of running on personal computers and smartphones. The platform randomly samples a sentence from the pool of extracted texts, and presents it to a reader (see Figure 1). It also displays the recording status and statistics, such as elapsed time and the total number of read sentences. Additionally, it has the "pause" and "next" buttons to control the recording process. The readers were allowed to quit at any time. We recruited readers by advertising the project in social media, news, and open messaging communities on WhatsApp and Telegram. Readers who were at least 18 years old were included so that they could legally agree to participate in data collection. The audios were recorded in 48 kHz and 16 bits, but downsampled to 16 kHz and 16 bits for online publication. Following our experimental protocol, we did not store readers' personal information except for the geolocation coordinates, IP address, and device type.

Аудио жазылуда

Абырой болғанда тұрғындар қоныстанып үлгермеген еді.

Сіз 100-ден 0 аудио жаздыңыз

Figure 1: The speech recording web interface.

Category	Train	Valid	Test	Total
Duration (hours)	318.4	7.1	7.1	332.6
# Utterances	147,236	3,283	3,334	153,853
# Words	1.61M	35.2k	35.8k	1.68M
# Unique Words	157,191	13,525	13,959	160,041
# Device IDs	1,554	29	29	1,612
# Speakers	-	29	29	-

Table 1: The KSC database specifications.

Several native Kazakh transcribers were hired to check the quality of recordings. The transcribers logged on to a special transcription checking platform and were provided with an audio segment and the corresponding sentence that a reader had read. The task was to check if the reader had read the sentence according to the prompt, and to transcribe any deviations or other acoustic events based on a set of transcription instructions. As an additional quality measure, we hired a linguist who was assigned to supervise the transcribers and to randomly check tasks they completed. To harmonize the transcriptions, the linguist also held the "go through errors" sessions with the transcribers.

The transcribers were instructed to reject utterances containing obvious mispronunciations or severe noises, to convert numbers into words[6], and to trim long silences at the beginning and end of the audio segments. Additionally, they were instructed to enclose partial repetitions and hesitations in parentheses, for example, '(he) hello' and '(ah)', and to indicate other non-verbal sounds produced by readers, such as sneezing and coughing, using a special '[noise]' token. Background noises were not labeled.

When the size of accepted utterances reached 100 hours, we built an ASR system to automatically check the recordings. The system accepted only recordings perfectly matching corresponding text prompts–that is, 0% character error rate (CER), whereas the remaining utterances were left to human transcribers.

3.3 Database Specifications

The KSC database specifications are provided in Table 1. We split the data into three sets of non-overlapping speakers: training, validation, and test. While the training set recordings were collected

---

[6] Note that after converting numbers into words some sentence lengths exceeded 25 words.

Category		Valid	Test
Gender (%)	Female	51.7	51.7
	Male	48.3	48.3
Age (%)	18-27	37.9	34.5
	28-37	34.5	31.0
	38-47	10.4	13.8
	48 and above	17.2	20.7
Region (%)	East	13.8	13.8
	West	20.7	17.2
	North	13.8	20.7
	South	37.9	41.4
	Center	13.8	6.9
Device (%)	Phone	62.1	79.3
	Computer	37.9	20.7
Headphone (%)	Yes	20.7	17.2
	No	79.3	82.8

Table 2: The validation and test sets speaker details.

from anonymous speakers, the validation and test sets were collected from identifiable speakers to ensure that they did not overlap with the training set, represented different age groups and regions, and were gender balanced (see Table 2). In total, around 153,000 utterances were accepted, yielding 332 hours of transcribed speech data. Note that the device IDs could not be used to represent the number of speakers in the training set as several speakers might have used the same device or the same speaker might have used different devices. Therefore, the total number of speakers in the training set is not shown in Table 1. The whole database creation process took around four months, and the database size is around 38GB.

The Kazakh writing system differs depending on the regions where the language is spoken. For example, the Cyrillic alphabet is used in Kazakhstan and Mongolia, while an Arabic-derived alphabet is used in China. In the KSC, we presented all texts using the Cyrillic alphabet consisting of 42 letters. The distribution of these letters in the KSC is given in Figure 2.

One of the important features of the proposed KSC database is that it was collected in various environment conditions (e.g. home, office, café, transport, and street), with diverse background noises through mobile devices (e.g. phones and tablets) and personal computers, with and without headphone sets, which is similar to realistic use-case scenarios. Consequently, our database enables the development and evaluation of ASR systems designed to operate in real-world voice-enabled applications, such as voice commands, voice search, message dictation, and so on.

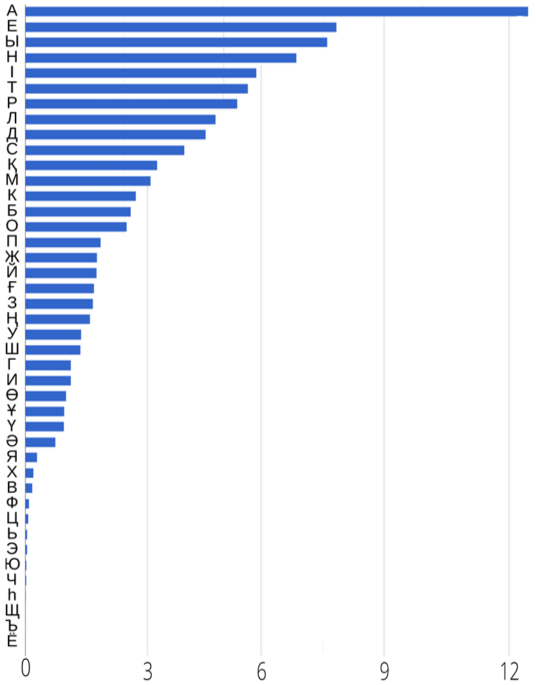


Figure 2: The distribution of letters in the KSC (%).

The KSC database consists of audio recordings, transcripts, and metadata stored in separate folders. The audio and corresponding transcription filenames are the same, except that the audio recordings are stored as WAV files, whereas the transcriptions are stored as TXT files using the UTF-8 encoding. The metadata contain the data splitting information (training, validation, and test) and the speaker's details (gender, age, region, device, and headphones) of the validation and test sets.

4 Speech Recognition Experiment

To demonstrate the utility and reliability of the KSC database, we conducted speech recognition experiments using both the traditional deep neural network-hidden Markov model (DNN-HMM) and recently proposed end-to-end (E2E) architectures. We did not compare or perform thorough architecture searches for either DNN-HMM and E2E, since this falls outside the scope of this paper.

4.1 Experimental Setup

All ASR models were trained with a single V100 GPU running on the NVIDIA DGX-2 server using the training set. All hyper-parameters were tuned using the validation set. The best-performing model was evaluated using the test set. All results are reported without lattice or n-best hypotheses rescoring, and no external data have been used.

4.1.1 The DNN-HMM ASR

The DNN-HMM ASR system was built using the Kaldi framework (Povey et al., 2011). We followed the Wall Street Journal (WSJ) recipe with the "nnet3+chain" setup and other latest Kaldi developments. The acoustic model was constructed using the factorized time-delay neural networks (TDNN-F) (Povey et al., 2018) trained with the lattice-free maximum mutual information (LF-MMI) (Povey et al., 2016) training criterion. The inputs were Mel-frequency cepstral coefficients (MFCC) features with cepstral mean and variance normalization extracted every 10 ms over a 25 ms window. In addition, we applied data augmentation using the speed perturbation (SpeedPerturb) (Ko et al., 2015) at rates of 0.9, 1.0, and 1.1 and the spectral augmentation (SpecAugment) (Park et al., 2019) techniques.

We employed a graphemic lexicon because of the strong correspondence between word spelling and pronunciation in Kazakh (e.g. "$hello \rightarrow h\ e\ l\ l\ o$"). The graphemic lexicon was constructed by extracting all words in the training set, which resulted in 157,191 unique words. During the decoding stage, we employed a 3-gram language model[7] (LM) with the Kneser-Ney smoothing built using the SRILM toolkit (Stolcke, 2002). The 3-gram LM was trained using the transcripts of the training set and the vocabulary covering all the words in the graphemic lexicon.

4.1.2 The E2E ASR

The E2E ASR systems were built using the ESPnet framework (Watanabe et al., 2018). We followed the WSJ recipe to train two different encoder-decoder architectures based on the recurrent neural networks (RNN) (Bahdanau et al., 2015) and the Transformer (Vaswani et al., 2017). Both architectures were jointly trained with the connectionist temporal classification (CTC) (Graves et al., 2006) objective function under the multi-task learning framework (Kim et al., 2017). The input speech was represented as an 80-dimensional filterbank features with pitch computed every 10 ms over a 25 ms window. For both E2E architectures, the acoustic features were first processed by few initial blocks of VGG network (Simonyan and Zisserman, 2015). Since Kazakh is a morphologically rich language, it is susceptible to severe data sparseness. To overcome this issue, we employed character-level output units in both architectures. In total, we used 45 distinct output units consisting of 42 letters from the Kazakh alphabet and 3 special tokens–that is, <*unk*>, <*space*>, and <*blank*> used in CTC.

The E2E ASR systems do not require a lexicon when modeling with grapheme-based output units (Sainath et al., 2018). The character-level LM was built using the transcripts of the training set as a two-layer RNN with 650 long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) units each. We utilized the LSTM LM during the decoding stage using the shallow fusion (Gülçehre et al., 2015) for both E2E architectures. Besides, we augment the training data using the SpeedPerturb and the SpecAugment techniques. For decoding, we set the beam size to 30 and the LSTM LM interpolation weight to 1.

E2E-RNN. The encoder module of the RNN-based E2E ASR system consists of three bi-directional LSTM layers with 1,024 units per direction per layer. The decoder module is a single uni-directional LSTM with 1,024 units. We train the model for 20 epochs using the Adadelta optimizer with the initial learning rate set to 1 and the batch size set to 30. The interpolation weight for the CTC objective was set to 0.5.

E2E-Transformer. The Transformer-based E2E ASR system consists of 12 encoder and 6 decoder blocks. We set the number of heads in the self-attention layer to 4 with 256-dimension hidden states and the feed-forward network dimensions to 2,048. We set the dropout rate and label smoothing to 0.1. The model was trained for 160 epochs using the Noam optimizer (Vaswani et al., 2017) with the initial learning rate of 10 and the warmup-steps of 25,000. The batch size was set to 96. We report results on an average model constructed using the last 10 checkpoints. The interpolation weight for the CTC objective was set to 0.3.

4.2 Experimental Results

The experimental results are presented in Table 3 in terms of both the character error rate (CER) and word error rate (WER). All the ASR models achieved competitive results on both the validation and test sets. We found that the validation set is more challenging than the test set. When compared without SpecAugment, the performance of the DNN-HMM model (ID 1) is slightly better than the E2E-RNN (ID 5), but inferior to the

---

[7] We tested different N-gram orders and found the 3-gram to perform best on our setup.

ID	Model	LM	SpeedPerturb	SpecAugment	Valid		Test	
					CER	WER	CER	WER
1	DNN-HMM	Yes	Yes	No	5.2	14.2	4.6	13.7
2		Yes	Yes	Yes	5.3	14.9	4.7	13.8
3	E2E-LSTM	No	No	No	9.9	32.0	8.7	28.8
4		Yes	No	No	7.9	20.1	7.2	18.5
5		Yes	Yes	No	5.7	15.9	5.0	14.4
6		Yes	Yes	Yes	4.6	13.1	4.0	11.7
7	E2E-Transformer	No	No	No	6.1	22.2	4.9	18.8
8		Yes	No	No	4.5	13.9	3.7	11.9
9		Yes	Yes	No	3.9	12.3	3.2	10.5
10		Yes	Yes	Yes	**3.2**	**10.0**	**2.8**	**8.7**

Table 3: The CER (%) and WER (%) performances of different ASR models built using KSC.

E2E-Transformer (ID 9). We could not achieve any improvements on the DNN-HMM model (ID 2) using the SpecAugment despite trying different hyper-parameter tuning recommendations (Zhou et al., 2020). Overall, the best CER and WER results are achieved by the E2E-Transformer (ID 10) followed by the E2E-RNN (ID 6) and then the DNN-HMM (ID 1).

We observed that the LM fusion significantly improves the performances of both E2E models. For example, 35% and 36% relative WER improvements are achieved on the test set for the RNN (from ID 3 to ID 4) and Transformer (from ID 7 to ID 8) models, respectively. Furthermore, both data augmentation techniques based on SpeedPerturb and SpecAugment are highly effective for the Kazakh E2E ASR where additional improvements are achieved. For example, when models without and with data augmentations are compared, 36% and 26% relative WER improvements are achieved on the test set for the RNN (from ID 4 to ID 6) and Transformer (from ID 8 to ID 10) models, respectively.

These experimental results successfully demonstrate the utility of the KSC database for the speech recognition task. We leave the exploration of the optimal hyper-parameter settings and detailed comparison of different ASR architectures for future work.

5 Discussion

1) Data sparsity. Kazakh language speech recognition is considered challenging due to the agglutinative nature of the language, where word structures are formed by adding derivational and inflectional affixes to stems in a specific order. As a result, the vocabulary size might considerably grow resulting in a data sparsity problem, especially for models operating in a word level, such as our DNN-HMM architecture. The potential solution is to break down words into finer-level linguistic units, such as characters or subword units (Sennrich et al., 2016). We investigated the impact of other output unit sizes on the performance of the Kazakh E2E-Transformer ASR and did not observe any considerable improvements over the character-level outputs (see Figure 3). The output units were generated using the byte pair encoding (BPE) algorithm implemented in the SentencePiece (Kudo and Richardson, 2018) tokenizer.

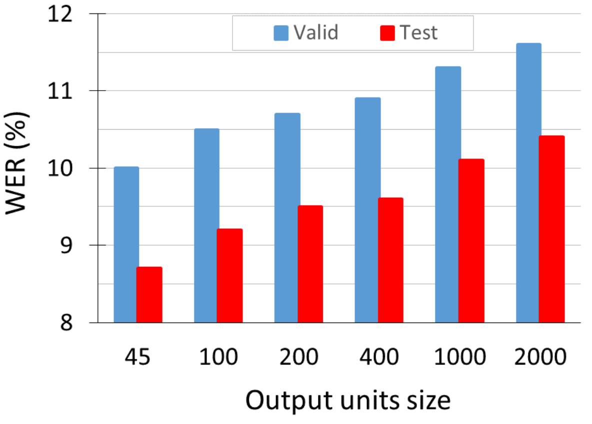


Figure 3: The impact of different output unit sizes on the E2E-Transformer ASR performance.

2) Code-switching. Another challenge is the Kazakh-Russian code-switching practice which is common in daily communication as the majority of

ID	Confusion pairs	Insertion	Deletion	Substitution
1	жоғарғы ⇒ жоғары (jogargy ⇒ jogary) “upper” or “higher” ⇒ “top” or “high”	мен (men) “I” or “with”	де (de) “too” (after a thin-vowelled word)	да (da) “too” (after a thick-voweled word)
2	өзеннің ⇒ өзенінің (uzennin ⇒ uzeninin) “of a river” ⇒ “of the river”	бұл (bul) “this” or “these”	ал (al) an imperative of “take” or “whereas”	де (de) “too” (after a thin-voweled word)
3	ас ⇒ ақ (as ⇒ aq) “meal” ⇒ “white”	бір (bir) “one"	да (da) “too” (after a thick-vowelled word)	мен (men) “I” or “with”
4	бағы ⇒ баға (bagy ⇒ baga) “the garden” ⇒ “price”	да (da) “too” (after a thick-vowelled word)	әр (ar) “every”	жылы (jyly) “the year” or "warm"
5	болу ⇒ болуы (bolu ⇒ boluy) “being” ⇒ “the being”	ақ (aq) “white”	бір (bir) “one"	бір (bir) “one”

Table 4: The top five confusion pairs (reference word ⇒ recognized word) and insertion, deletion, and substitution errors in the recognized output of the E2E-Transformer ASR.

Kazakhs are bilingual. Mostly, inter-sentential and intra-sentential types of code-switching are practiced, however, intra-word code-switching is also possible. For example, one can say “*Men magazinge bardym*” (“*I went to a store*”), where the Russian word “*magazin*” is appended by the Kazakh inflection “*-ge*” representing the preposition “*to*”. Furthermore, while the spelling of Kazakh words closely matches their pronunciation, this is not the case for Russian words, for example, the letter “*o*” is sometimes pronounced as /a/, which might confuse an ASR system. We observed that our ASR system is ineffective in code-switched utterances. Therefore, future work should focus on alleviating these errors.

3) Data efficiency. To analyze the data efficiency–that is, increase in performance due to the addition of new training data–we trained our best E2E-Transformer ASR system using different amounts of data. In particular, we first randomly sampled 40 hours of data and kept increasing their size until the entire training set was covered. The experimental results indicate that the WER performance has not converged yet and further data collection might be effective (see Figure 4). Therefore, we plan to continue the data collection process.

4) ASR output analysis. We inspected the recognized outputs of the best E2E-Transformer model to identify the most challenging characters and words. Table 4 lists the top five confusion pairs and insertion, deletion, and substitution errors. Most of them are commonly used words, such as conjunctions and numbers. In addition, we also inspected the most confused character pairs (see Figure 5). We observed that the Kazakh ASR system confuses characters with a similar pronunciation, such as “н” (/n/) and “ң” (/ŋ/), “і” (/ɪ/) and “ы” (/ə/), and so on.

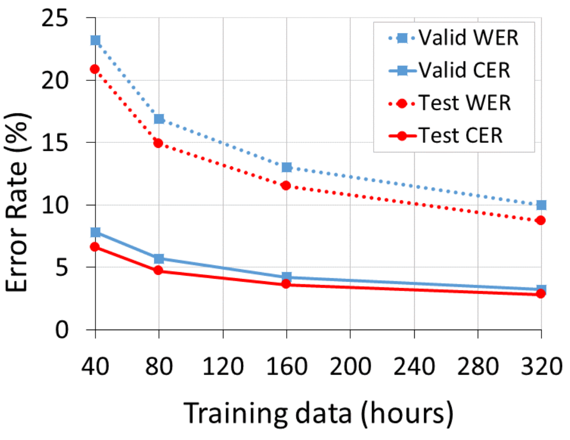


Figure 4: The data efficiency experiment.

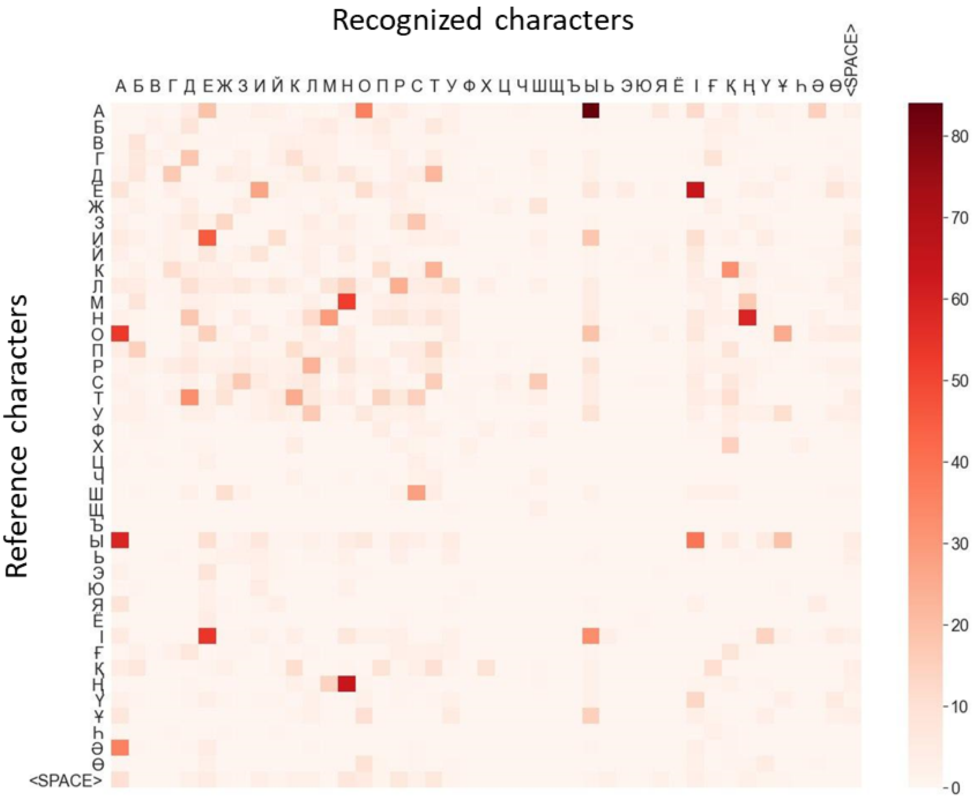


Figure 5: The character confusion matrix for the E2E-Transformer ASR.

5) Performance comparison. We cannot directly compare our results to previous works; however, our WER results are appealing. For example, Mamyrbayev et al. (2019) used 76 hours of speech data to build an DNN-HMM ASR system which achieved 32.7% WER on clean read speech. Similarly, the DNN-HMM ASR system built using 78 hours of data in (Shi et al., 2017) achieved 25.1% WER on read speech. On the other hand, Mamyrbayev et al. (2020) achieved 17.8% WER on clean read speech using the E2E ASR system trained on 126 hours of data. In comparison, our best model achieved 8.7% WER on the test set.

6) Benefit for other Turkic Languages. We also envision that the KSC can be utilized in cross-lingual transfer learning techniques (Das and Hasegawa-Johnson, 2015) to improve ASR systems for other Turkic languages, such as Kyrgyz and Tatar.

6 Conclusion

In this work, we presented the KSC database containing around 332 hours of transcribed speech data. It was developed to advance Kazakh speech processing applications, such as speech recognition, speech synthesis, and speaker recognition. We described the database construction procedures and discussed challenges that should be addressed in future work. The described methodologies might benefit other researchers planning to build a speech corpus for a low-resource language. The database is freely available for any purposes including research and commercial use. We also conducted preliminary speech recognition experiments using both traditional hybrid DNN-HMM and recently proposed E2E ASR architectures. To ease the database usage and ensure the reproducibility of experiments, we split it into three non-overlapping sets (training, validation, and test) and released our ESPnet recipe. The detailed exploration of better ASR settings, as well as the adaptation of the database to other applications, is left for future work.

Acknowledgments

We would like to thank Aigerim Boranbayeva for helping to train and monitor the transcribers. We also thank Kuralay Baimenova, Yerbol Absalyamov, Gibrat Kurmanov, and Rustem Yeshpanov for helping to attract readers and to resolve other administrative tasks.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015*.

Amit Das and Mark Hasegawa-Johnson. 2015. Cross-lingual transfer learning during supervised training in low resource scenarios. In *Proc. of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH), Dresden, Germany, September 6-10, 2015*, pages 3531–3535.

Bhavna Dave. 2007. *Kazakhstan-ethnicity, language and power*. Routledge.

Jiayu Du, Xingyu Na, Xuechen Liu, and Hui Bu. 2018. AISHELL-2: transforming mandarin ASR research into industrial scale. *CoRR*, abs/1808.10583.

Maxine Eskenazi, Gina-Anne Levow, Helen Meng, Gabriel Parent, and David Suendermann. 2013. *Crowdsourcing for speech processing: Applications to data collection, transcription and assessment*. John Wiley & Sons.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proc. of the 23rd International Conference on Machine Learning, (ICML), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pages 369–376.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loïc Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Awni Y. Hannun, C. Case, J. Casper, Bryan Catanzaro, Greg Diamos, E. Elsen, Ryan Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition. *ArXiv*, abs/1412.5567.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Olga Khomitsevich, Valentin Mendelev, Natalia A. Tomashenko, Sergey Rybin, Ivan Medennikov, and Saule Kudubayeva. 2015. A bilingual kazakh-russian system for automatic speech recognition and synthesis. In *Proc. of the 17th International Conference on Speech and Computer (SPECOM), Athens, Greece, September 20-24, 2015*, volume 9319, pages 25–33.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, March 5-9, 2017*, pages 4835–4839.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Proc. of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH), Dresden, Germany, September 6-10, 2015*, pages 3586–3589.

Jia Xin Koh, Aqilah Mislan, Kevin Khoo, Brian Ang, Wilson Ang, Charmaine Ng, and Ying-Ying Tan. 2019. Building the singapore english national speech corpus. In *Proc. of the 20th Annual Conference of the International Speech Communication Association (INTERSPEECH), Graz, Austria, 15-19 September 2019*, pages 321–325.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Olzhas Makhambetov, Aibek Makazhanov, Zhandos Yessenbayev, Bakhyt Matkarimov, Islam Sabyrgaliyev, and Anuar Sharafudinov. 2013. Assembling the kazakh language corpus. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 18-21 October 2013, Seattle, Washington, USA*, pages 1022–1031.

Orken Mamyrbayev, Keylan Alimhan, Bagashar Zhumazhanov, Tolganay Turdalykyzy, and Farida Gusmanova. 2020. End-to-end speech recognition in agglutinative languages. In *Proc. of the 12th Asian Conference on Intelligent Information and Database Systems (ACIIDS), Phuket, Thailand, March 23-26, 2020*, volume 12034, pages 391–401.

Orken J. Mamyrbayev, Mussa Turdalyuly, Nurbapa Mekebayev, Keylan Alimhan, Aizat Kydyrbekova, and Tolganay Turdalykyzy. 2019. Automatic recognition of kazakh speech using deep neural networks. In *Proc. of the 11th Asian Conference on Intelligent Information and Database Systems (ACIIDS), Yogyakarta, Indonesia, April 8-11, 2019*, pages 465–474.

Scott Novotney and Chris Callison-Burch. 2010. Cheap, fast and good enough: Automatic speech recognition with non-expert transcription. In *Proc. of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, June 2-4, 2010, Los Angeles, California, USA*, pages 207–215.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proc. of the 20th Annual Conference of the International Speech Communication Association (INTERSPEECH), Graz, Austria, 15-19 September 2019*, pages 2613–2617.

Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur. 2018. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Proc. of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH), Hyderabad, India, 2-6 September 2018*, pages 3743–3747.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The Kaldi speech recognition toolkit. In *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding*.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. 2016. Purely sequence-trained neural networks for ASR based on lattice-free MMI. In *Proc. of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH), San Francisco, CA, USA, September 8-12, 2016*, pages 2751–2755.

Tara N. Sainath, Rohit Prabhavalkar, Shankar Kumar, Seungji Lee, Anjuli Kannan, David Rybach, Vlad Schogol, Patrick Nguyen, Bo Li, Yonghui Wu, Zhifeng Chen, and Chung-Cheng Chiu. 2018. No need for a lexicon? Evaluating the value of the pronunciation lexica in end-to-end models. In *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, April 15-20, 2018*, pages 5859–5863.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), August 7-12, 2016, Berlin, Germany*.

Ying Shi, Askar Hamdullah, Zhiyuan Tang, Dong Wang, and Thomas Fang Zheng. 2017. A free Kazakh speech database and a speech recognition baseline. In *Proc. of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Kuala Lumpur, Malaysia, December 12-15, 2017*, pages 745–748.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *Proc. of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015*.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast - But is it good? Evaluating non-expert annotations for natural language tasks. In *Proc. of the Conference on Empirical Methods in Natural Language Processing, (EMNLP), 25-27 October 2008, Honolulu, Hawaii, USA*, pages 254–263.

Andreas Stolcke. 2002. SRILM - An extensible language modeling toolkit. In *Proc. of the International Conference on Spoken Language Processing (ICSLP), Denver, Colorado, USA, September 16-20, 2002*.

Shinnosuke Takamichi and Hiroshi Saruwatari. 2018. CPJD corpus: Crowdsourced parallel speech corpus of Japanese dialects. In *Proc. of the 11th International Conference on Language Resources and Evaluation (LREC), Miyazaki, Japan, May 7-12, 2018*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of the Annual Conference on Advances in Neural Information Processing Systems (NIPS), 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proc. of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH), Hyderabad, India, 2-6 September 2018*, pages 2207–2211.

Dong Yu and Li Deng. 2014. *Automatic Speech Recognition: A Deep Learning Approach*. Springer Publishing Company.

Wei Zhou, Wilfried Michel, Kazuki Irie, Markus Kitza, Ralf Schlüter, and Hermann Ney. 2020. The rwth asr system for ted-lium release 2: Improving hybrid hmm with specaugment. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 7839–7843. IEEE.

TDMSci: A Specialized Corpus for Scientific Literature Entity Tagging of Tasks Datasets and Metrics

Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin and Debasis Ganguly
IBM Research Europe, Ireland
`{yhou|mgleize|fbonin|debasga1}@ie.ibm.com`

Abstract

Tasks, *Datasets* and *Evaluation Metrics* are important concepts for understanding experimental scientific papers. However, most previous work on information extraction for scientific literature mainly focuses on the abstracts only, and does not treat datasets as a separate type of entity (Zadeh and Schumann, 2016; Luan et al., 2018). In this paper, we present a new corpus that contains domain expert annotations for *Task (T), Dataset (D), Metric (M)* entities on 2,000 sentences extracted from NLP papers. We report experiment results on TDM extraction using a simple data augmentation strategy and apply our tagger to around 30,000 NLP papers from the ACL Anthology. The corpus is made publicly available to the community for fostering research on scientific publication summarization (Erera et al., 2019) and knowledge discovery.

1 Introduction

The recent years have witnessed a significant growth in the number of scientific publications and benchmarks in many disciplines. As an example, in the year 2019 alone, more than 170k papers were submitted to the pre-print repository arXiv[1] and among them, close to 10k papers were classified as NLP papers (i.e., `cs.CL`). Each empirical field of science, including NLP, will benefit from the massive increase in studies, benchmarks, and evaluations, as they can provide ingredients for novel scientific advancements.

However, researchers may struggle to keep track of all studies published in a particular field, resulting in duplication of research, comparisons with old or outdated benchmarks, and lack of progress. In order to tackle this problem, recently there have been a few manual efforts to summarize the state-of-the-art on selected subfields of NLP in the form of leaderboards that extract tasks, datasets, metrics and results from papers, such as *NLP-progress*[2] or *paperswithcode*.[3] But these manual efforts are not sustainable over time for all NLP tasks.

Over the past few years, several studies and shared tasks have begun to tackle the task of entity extraction from scientific papers. Augenstein et al. (2017) formalized a task to identify three types of entities (i.e., *task, process, material*) in scientific publications (SemEval 2017 task10). Gábor et al. (2018) presented a task (SemEval 2018 task 7) on semantic relation extraction from NLP papers. They provided a dataset of 350 abstracts and reuse the entity annotations from Zadeh and Schumann (2016). Recently Luan et al. (2018) released a corpus containing 500 abstracts with six types of entity annotations. However, these corpora do not treat *Dataset* as a separate type of entity and most of them focus on the abstracts only.

In a previous study, we developed an IE system to extract $\{$*task, dataset, metric*$\}$ triples from NLP papers based on a small, manually created task/dataset/metric (TDM) taxonomy (Hou et al., 2019). In practice, we found that a TDM knowledge base is required to extract TDM information and build NLP leaderboards for a wide range of NLP papers. This can help researchers quickly understand related literature for a particular task, or to perform comparable experiments.

As a first step to build such a TDM knowledge base for the NLP domain, in this paper we present a specialized English corpus containing 2,000 sentences taken from the full text of NLP papers which have been annotated by domain experts for three main concepts: *Task* (T), *Dataset* (D) and *Metric*

---

[1] `https://arxiv.org/help/stats/2019_by_area`

[2] `https://github.com/sebastianruder/NLP-progress`

[3] `https://paperswithcode.com`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 707–714
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

(M). Based on this corpus, we develop a TDM tagger using a novel data augmentation technique. In addition, we apply this tagger to around 30,000 NLP papers from the ACL Anthology and demonstrate its value to construct an NLP TDM knowledge graph. We release our corpus at `https://github.com/IBM/science-result-extractor`.

2 Related Work

A lot of interest has been focused on information extraction from scientific literature. SemEval 2017-task 10 (Augenstein et al., 2017) proposed a new task for the identification of three types of entities (*Task, Method*, and *Material*) in a corpus of 500 paragraphs taken from open access journals. Based on Augenstein et al. (2017) and Gábor et al. (2018), Luan et al. (2018) created *SciERC*, a dataset containing 500 scientific abstracts with annotations for six types of entities and relations between them. Both SemEval 2017-task 10 and *SciERC* do not treat "*dataset*" as a separate entity type. Instead, their "*material*" category comprises a much larger set of resource types, including tools, knowledge resources, bilingual dictionaries, as well as datasets. In our work, we focus on "*datasets*" entities that researchers use to evaluate their approaches because dataset is one of the three core elements to construct leaderboards for NLP papers.

Concurrent to our work, Jain et al. (2020) develop a new corpus *SciREX* which contains 438 papers on different domains from *paperswithcode*. It includes annotations for four types of entities (i.e., *Task, Dataset, Metric, Method*) and the relations between them. The initial annotations were carried out automatically using distant signals from *paperswithcode*. Later human annotators performed necessary corrections to generate the final dataset. *SciREX* is the closest to our corpus in terms of entity annotations. In our work, we focus on TDM entities which reflect the collectively shared views in the NLP community and our corpus is annotated by five experts who all have 5-10 years NLP research experiences.

3 Corpus Creation

3.1 Annotation Scheme

We developed an annotation scheme for annotating Task, Dataset, and Evaluation Metric phrases in NLP papers. Our annotation guidelines[4] are based on the scientific term annotation scheme described in Zadeh and Schumann (2016). Different from previous corpora (Zadeh and Schumann, 2016; Luan et al., 2018), we only annotated **factual** and **content-bearing** entities. This is because we aim to build a TDM knowledge base in the future and non-factual entities (e.g., *a high-coverage sense-annotated corpus* in Example 1) do not reflect the collectively shared views of TDM entities in the NLP domain.

(1) In order to learn models for disambiguating a large set of content words, *a high-coverage sense-annotated corpus* is required.

Following the above guidelines, we also do not annotate *anonymous entities*, such as "*this task*" or "*the dataset*". These entities are anaphors and can not be used independently to refer to any specific TDM entities without contexts. In general, we choose to annotate TDM entities that normally have specific names and whose meanings usually are consistent across different papers. From this perspective, the TDM entities that we annotate are similar to named entities, which are self-sufficient to identify the referents.

3.2 Pilot Annotation Study

Data preparation. For the pilot annotation study, we choose 100 sentences from the NLP-TDMS corpus (Hou et al., 2019). The corpus contains 332 NLP papers which are annotated with triples of $\{$*Task, Dataset, Metric*$\}$ on the document level. We use string and substring match to extract a list of sentences from these papers which are likely to contain the document level *Task, Dataset, Metric* annotations. We then manually choose 100 sentences from this list following the criteria: 1) the sentence should contain the valid mention of *Task*, *Dataset*, or *Metric*; 2) the sentences should come from different papers as much as possible; and 3) there should be a balanced distribution of *task*, *dataset*, and *metric* mentions in these sentences.

Annotation agreement. Four NLP domain experts annotated the same 100 sentences for a pilot annotation study, following the annotation guidelines described above. All the annotations were conducted using BRAT (Stenetorp et al., 2012). The inter annotator agreement has been calculated with a pairwise comparison between annotators using *precision*, *recall* and *F-score* on the exact match of the annotated entities. In other words,

---

[4] Please see the appendix for the whole annotation scheme.

	Mean F-score (EM)	Fleiss' κ (Token)
Task	0.720	0.797
Dataset	0.752	0.829
Metric	0.757	0.896
Overall	0.743	0.842

Table 1: Inter-annotator agreement.

	Train	Test
# Sentences	1500	500
# Task	1219	396
# Dataset	420	192
# Metric	536	174

Table 2: Statistics of task/dataset/metric mentions in the training and testing datasets.

two entities are considered matching (true positive) if they have the same boundaries and are assigned to the same label. We also calculate Fleiss' kappa on a per token basis, comparing the agreement of annotators on each token in the corpus. Table 1 lists the mean F-score as well as the token-based Fleiss' κ value for each entity type. Overall, we achieve high reliability for all categories.

Adjudication. The final step of the pilot annotation was to reconcile disagreements among the four annotators to produce the final canonical annotation. This step also allows us to refine the annotation guidelines. Specifically, through the discussion of annotation disagreements we could identify ambiguities and omissions in the guidelines. For example, one point of ambiguity was whether a *task* must be associated with a dataset, or can we annotate higher level tasks, e.g., *sequence labeling*, which do not have a dedicated dataset but may include several tasks and datasets. This discussion also revealed the overlap in how we refer to tasks and datasets in the literature. As authors we frequently use these interchangeably, often with shared tasks, e.g., "*SemEval-07 task 17*" seems to more often refer to a dataset than a specific instance of the (Multilingual) Word Sense Disambiguation task, or the "*MultiNLI*" corpus is sometimes used as shorthand for the task. After the discussion, we agreed that we should annotate higher level tasks. In addition, we should assign labels to entities according to their actual referential meanings in contexts.

3.3 Main Annotation

After the pilot study, 1,900 additional sentences were annotated by five NLP researchers. Four annotators participated in the pilot annotation study, and all annotators joined the adjudication discussion. Note that every annotator annotate a different set of sentences. The annotator who designed the annotation scheme annotated 700 sentences, the other four annotators annotated 300 sentences each.[5]

In general, most sentences in our corpus are not from the abstracts. Note that the goal of developing our corpus is to automatically build an NLP TDM taxonomy and use them to tag NLP papers. Therefore, the inclusion of sentences from the whole paper other than the abstract section is important for our purpose. Because not all abstracts talk about all three elements. For instances, for the top ten papers listed in the {*sentiment analysis, IMDB, accuracy*} leaderboard in *paperswithcode*[6], only four abstracts mention the dataset "*IMDB*". If we only focus on the abstracts, we will miss the other six papers from the leaderboard.

4 A TDM Entity Tagger

Our final corpus *TDMSci* contains 2,000 sentences with 2,937 mentions of three entity types. We convert the original BRAT annotations to the standard CoNLL format using BIO scheme.[7] We develop a tagger to extract TDM entities based on this corpus.

4.1 Experimental Setup

To evaluate the performance of our tagger, we split *TDMSci* into training and testing sets, which contains 1,500 and 500 sentences, respectively. Table 2 shows the statistics of task/dataset/metric mentions in these two datasets. For evaluation, we report precision, recall, F-score on exact match for each entity type as well as micro-averaged precision, recall, F-score for all entities.

---

[5] Due to time constraints, we did not carry out another round of pilot study. Partially it is because we felt that the revised guidelines resulting from the discussion were sufficient for the annotators to decide ambiguous cases. So in the second stage annotators annotated disjoint sets of sentences. After this, the annotator who designed the annotation scheme went through the whole corpus again to verify the annotations.

[6] https://paperswithcode.com/sota/sentiment-analysis-on-imdb, search was carried out on November, 2020.

[7] Note that our BRAT annotation contains a small amount of embedded entities, e.g., *WSJ portion of Ontonotes* and *Ontonotes*. We only keep the longest span when we convert the BRAT annotations to the CoNLL format.

	CRF			*CRF w/ gazetteer*			*SciIE*			*Flair-TDM*		
	P	R	F	P	R	F	P	R	F	P	R	F
Original training data												
Task	63.79	46.72	53.94	61.86	45.45	52.40	69.23	54.55	61.02	61.54	54.55	57.83
Dataset	65.42	36.46	46.82	65.45	37.50	47.68	66.97	38.02	48.50	52.66	46.35	49.30
Metric	80.00	66.67	72.73	80.95	68.39	74.14	77.99	71.26	74.47	76.33	74.14	75.22
Micro-	68.45	48.69	56.90	67.70	48.69	56.64	71.21	54.20	61.55	62.99	56.96	59.79
Original Training data + Augmented masked training data												
Task	63.24	43.43	51.50	62.96	42.93	51.05	68.63	55.81	**61.56**	65.14	53.79	58.92
Dataset	62.38	32.81	43.00	64.71	34.38	44.90	55.43	50.52	52.86	59.15	50.52	**54.50**
Metric	80.15	62.64	70.32	79.29	63.79	70.70	76.83	72.41	74.56	79.63	74.14	**76.79**
Micro-	67.58	45.14	54.13	67.77	45.54	54.47	67.17	58.27	**62.40**	67.23	57.61	62.05

Table 3: Results of different models for *task/dataset/metric* entity recognition on *TDMSci* test dataset.

4.2 Models

We model the task as a sequence tagging problem. We apply a traditional CRF model (Lafferty et al., 2001) with various lexical features and a BiLSTM-CRF model for this task. To compare with the state-of-the-art entity extraction model on scientific literature, we also use *SciIE* from Luan et al. (2018) to train a *TDM* entity recognition model based on our training data. Below we describe all models in detail.

CRF. We use the Stanford CRF implementation (Finkel et al., 2005) to train a *TDM* NER tagger based on our training data. We use the following features: unigrams of the previous, current and next words, current word character n-grams, current POS tag, surrounding POS tag sequence, current word shape, surrounding word shape sequence.

CRF with gazetteers. To test whether the above CRF model can benefit from knowledge resources, we add two gazetteers to the feature set: one is a list containing around 6,000 dataset names which were crawled from LRE Map,[8] and another gazetteer comprises around 30 common evaluation metrics compiled by the authors.

SciIE. Luan et al. (2018) proposed a multi-task learning system to extract entities and relations from scientific articles. *SciIE* is based on span representations using ELMo (Peters et al., 2018) and here we adapt it for *TDM* entity extraction. Note that if *SciIE* predicts several embedded entities, we keep the one that has the highest confidence score. In practice we notice that this does not happen in our corpus.

Flair-TDM For BiLSTM-CRF model, we use the recent *Flair* framework (Akbik et al., 2018) based on the cased BERT-base embeddings (Devlin et al., 2018). We train our *Flair-TDM* model with a learning rate of 0.1, a batch size of 32, a hidden size of 768, and the maximum epochs of 150.

4.3 Data Augmentation

For TDM entity extraction, we expect that the surrounding context will play an important role. For instance, in the following sentence "we show that for X on the Y, our model outperforms the prior state-of-the-art", one can easily guess that X is a task entity while Y is a dataset entity. As a result, we propose a simple data augmentation strategy that generates the additional mask training data by replacing every token within an annotated TDM entity as `UNK`.

4.4 Results and Discussion

Table 3 shows the performance of different models for *task/dataset/metric* entity recognition on our testing dataset.

First, it seems that although adding gazetteers can help the CRF model detect *dataset* and *metric* entities better, the positive effect is limited. In general, both *SciIE* and *Flair-TDM* perform better than *CRF* models for detecting all three type of entities.

Second, augmenting the original training data with the additional masked data as described in Section 4.3 further improves the performance both for *SciIE* and *Flair-TDM*. However, this is not the case for the CRF models. We assume this is because CRF models heavily depend on the lexical features.

Finally, we randomly sampled 100 sentences from the testing dataset and compared the predicted TDM entities in *Flair-TDM* against the gold annotations. We found that most errors are from the boundary mismatch for task and dataset entities, e.g., *text summarization* vs. *abstractive text sum-*

---

[8] `http://www.elra.info/en/catalogues/lre-map/`

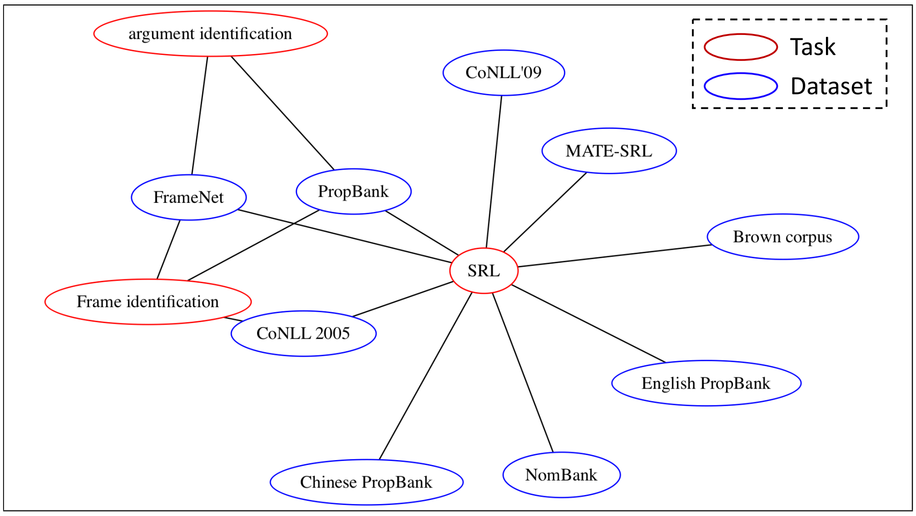


Figure 1: A subset of the *TDM* graph.

marization, or *Penn Treebank* vs. *Penn Treebank dataset*. The last error comes from the bias in the training data. A lot of researchers use "*Penn Treebank*" to refer to a dataset. So the model will learn this bias and only tag "*Penn Treebank*" as the dataset even though in a specific testing sentence, "*Penn Treebank dataset*" was used to refer to the same corpus.

In general, we think these mismatched predictions are reasonable in the sense that they capture the main semantics of the referents. Note that the numbers reported in Table 3 are based on exact match. Often, requiring exact match may be too restictive for downstreaming tasks. Therefore, we carried out an additional evaluation for the best *Flair-TDM* model using partial match from SemEval 2013-Task 9 (Segura-Bedmar et al., 2013), which gives us a micro-average F1 of 76.47 for type partial match.

5 An Initial TDM Knowledge Graph

In this section, we apply the *Flair-TDM* tagger to around 30,000 NLP papers from ACL Anthology to build an initial TDM knowledge graph.

We downloaded all NLP papers from the ACL Anthology[9] covering the period of 1974-2019. For each paper, we collect sentences from the title, the abstract/introduction/dataset/corpus/experiment sections, as well as from the table captions. We then apply the *Flair-TDM* tagger to these sentences. Based on the tagger results, we build an initial graph G using the following steps:

- add a *TDM* entity as a node into G if it appears at least five times in more than one paper;
- create a link between a *task* node and a *dataset/metric* node if they appear in the same sentence at least five times in different papers.

By applying the above simple process, we get a noisy *TDM* knowledge graph containing 180k nodes and 270k links. After checking a few dense areas, we find that our graph encodes valid knowledge about NLP task/dataset/metric. Figure 1 shows that in our graph, the task "SRL" (semantic role labelling) is connected to a few datasets such as "FrameNet", "PropBank", and "NomBank" that are standard benchmark datasets for this task.

Based on the tagged ACL Anthology and this initial noisy graph, we are exploring various methods to build a large-scale NLP TDM knowledge graph and to evaluate its accuracy/coverage in an ongoing work.

6 Conclusion

In this paper, we have presented a new corpus (*TDMSci*) annotated for three important concepts (*Task/Dataset/Metric*) that are necessary for extracting the essential information from an NLP paper. Based on this corpus, we have developed a *TDM* tagger using a simple but effective data augmentation strategy. Experiments on 30,000 NLP papers show that our corpus together with the *TDM* tagger can help to build *TDM* knowledge resources for the NLP domain.

---

[9] https://www.aclweb.org/anthology/

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Shai Erera, Michal Shmueli-Scheuer, Guy Feigenblat, Ora Peled Nakash, Odellia Boni, Haggai Roitman, Doron Cohen, Bar Weiner, Yosi Mass, Or Rivlin, Guy Lev, Achiya Jerbi, Jonathan Herzig, Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, Francesca Bonin, and David Konopnicki. 2019. A summarization system for scientific documents. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 211–216, Hong Kong, China. Association for Computational Linguistics.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, Michigan. Association for Computational Linguistics.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. 2018. Semeval-2018 task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT, New Orleans, Louisiana, June 5-6, 2018*, pages 679–688.

Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, and Debasis Ganguly. 2019. Identification of tasks, datasets, evaluation metrics, and numeric scores for scientific leaderboards construction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5203–5213.

Sarthak Jain, Madeleine van Zuylen, Hannaneh Hajishirzi, and Iz Beltagy. 2020. SciREX: A challenge dataset for document-level information extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7506–7516, Online. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Behrang Q. Zadeh and Anne-Kathrin Schumann. 2016. The acl rd-tec 2.0: A language resource for evaluating term extraction and entity recognition methods. In *LREC*.

A TDM Entity Annotation Guidelines

A.1 Introduction

This scheme describes guidelines for annotating *Task*, *Dataset*, and *Evaluation Metric* phrases in NLP papers. We have pre-processed NLP papers in PDF format and chosen sentences that are likely to contain the above-mentioned entities for annotation. These sentences may come from different sections (e.g., Abstract, Introduction, Experiment, Dataset) as well as tables (e.g., table captions).

A.2 Entity Types

We annotate the following three entity types:

- Task: A task is a problem to solve (e.g., *information extraction*, *sentiment classification*, *dialog state tracking*, *POS tagging*, *NER*).

- Dataset: A dataset is a specific corpus or language resource. Datasets are often used to develop models or run experiments for NLP tasks. A dataset normally has a short name, e.g., *IMDB*, *Gigaword*.

- Metric: An evaluation metric explains the performance of a model for a specific task, e.g., *BLEU* (for machine translation), or *accuracy* (for a range of NLP tasks).

A.3 Notes and Examples

Entity spans. Particular attention must be paid to the entity spans in order to improve agreement. The following list indicates all the annotation directions that annotators have been given regarding entity spans. Table 4 shows examples of correct span annotation.

- Following the ACL RD-TEC 2.0 annotation guideline,[10] determiners should not be part of an entity span. For example, the string 'the text8 test set', only the span 'test8' is annotated as *dataset*.

- Minimum span principle: Annotators should annotate only the minimum span necessary to represent the original meaning of task/dataset/metric. See Table 4, rows 1,2,3,4.

- Include 'corpus/dataset/benchmark' when annotating dataset if these tokens are the head-noun of the dataset entity. For example: 'ubuntu corpus', 'SemEval-2010 Task 8 dataset'.

- Exclude the head noun of 'task/problem' when annotating task (e.g., only annotation "link prediction" for "the link prediction problem") unless they are the essential part of the task itself (e.g., CoNLL-2012 shared task, SemEval-2010 relation classification task).

- Conjunction: If the conjunction NP is an ellipse, annotate the whole phrase (see Table 4, rows 6,11); otherwise, annotate the conjuncts separately (see Table 4, row 5).

- Tasks can be premodifiers (see Table 4, rows 7,8,12)

- Embedded spans: Normally TDM entities do not contain any other TDM entities. A small number of *Task* and *Dataset* entities can contain other entities (see Table 4, row 12).

Anonymous entities. Do not annotate anonymous entities, which include anaphors. The following examples are anonymous entities:

- *this task*

- *this metric*

- *the dataset*

- *a public corpus for context-sensitive response selection* in the sentence "Experimental results in a a public corpus for context-sensitive response selection demonstrate the effectiveness of the proposed multi-vew model."

Abbreviation. If both the full name and the abbreviation are present in the sentence, annotate the abbreviation with its corresponding full name together. For instance, we annotate "20-newsgroup (20NG)" as a dataset entity in Example 2.

Factual entity. Only annotate "factual, content-bearing" entities. Task, dataset, and metric entities normally have specific names and their meanings are consistent across different papers. In Example 3, "*a high-coverage sense-annotated corpus*" is not a factual entity.

---

[10] `https://github.com/languagerecipes/acl-rd-tec-2.0/blob/master/distribution/documents/acl-rd-tec-guidelines-ver2.pdf`

Row	Phrase	Annotation	Entity
1	The public Ubuntu Corpus	Ubuntu Corpus	Dataset
2	the web portion of TriviaQA	web portion of TriviaQA	Dataset
3	sentiment classification of movie reviews	sentiment classification	Task
4	the problem of part-of-speech tagging for informal, online conversational text	part-of-speech tagging	Task
5	The FB15K and WN18 datasets	FB15K; WN18	Dataset
6	Hits at 1, 3 and 10	Hits at 1, 3 and 10	Metric
7	Link prediction benchmarks	Link prediction	Task
8	POS tagging accuracy	POS tagging; accuracy	Task, Metric
9	the third Dialogue State Tracking Challenge	Dialogue State Tracking, third Dialogue State Tracking Challenge	Task, Dataset
10	SemEval-2017 Task 9	SemEval-2017 Task 9	Task
11	temporal and causal relation extraction and classification	temporal and causal relation extraction and classification	Task
12	the SemEval-2010 Task 8 dataset	SemEval-2010 Task 8 dataset; SemEval-2010 Task 8	Dataset,Task

Table 4: Examples of entity span annotation guidelines

(2) We used four datasets: IMDB, Elec, RCV1, and 20-newsgrous (20NG) to facilitate direct comparison with DL15.

(3) In order to learn models for disambiguating a large set of content words, a high-coverage sense-annotated corpus is required.

Top-down Discourse Parsing via Sequence Labelling

Fajri Koto **Jey Han Lau** **Timothy Baldwin**
School of Computing and Information Systems
The University of Melbourne
ffajri@student.unimelb.edu.au, jeyhan.lau@gmail.com, tbaldwin@unimelb.edu.au

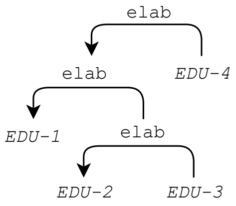
Abstract

We introduce a top-down approach to discourse parsing that is conceptually simpler than its predecessors (Kobayashi et al., 2020; Zhang et al., 2020). By framing the task as a sequence labelling problem where the goal is to iteratively segment a document into individual discourse units, we are able to eliminate the decoder and reduce the search space for splitting points. We explore both traditional recurrent models and modern pre-trained transformer models for the task, and additionally introduce a novel dynamic oracle for top-down parsing. Based on the `Full` metric, our proposed LSTM model sets a new state-of-the-art for RST parsing.[1]

1 Introduction

Discourse analysis involves the modelling of the structure of text in a document. It provides a systematic way to understand how texts are segmented hierarchically into discourse units, and the relationships between them. Unlike syntax parsing which models the relationship of words in a sentence, discourse parsing operates at the document-level, and aims to explain the flow of writing. Studies have found that discourse parsing is beneficial for downstream NLP tasks including document-level sentiment analysis (Bhatia et al., 2015) and abstractive summarization (Koto et al., 2019).

Rhetorical Structure Theory (RST; Mann and Thompson (1988)) is one of the most widely used discourse theories in NLP (Hernault et al., 2010; Feng and Hirst, 2014; Ji and Eisenstein, 2014; Li et al., 2016; Yu et al., 2018). RST organizes text spans into a tree, where the leaves represent the basic unit of discourse, known as elementary discourse units (EDUs). EDUs are typically clauses of a sentence. Non-terminal nodes in the tree represent discourse unit relations.



EDU-1: Roy E. Parrott, the company's president and chief operating officer since Sept. 1, was named to its board.
EDU-2: The appointment increased the number of directors to 10,
EDU-3: three of whom are company employees.
EDU-4: Simpson is an auto parts maker.

Figure 1: An example discourse tree, from the RST Discourse Treebank (`elab` = elaboration).

In Figure 1, we present an example RST tree with four EDUs spanning two sentences. In this discourse tree, EDUs are hierarchically connected with arrows and the discourse label `elab`. The direction of arrows indicates the nuclearity of relations, wherein a "satellite" points to its "nucleus". The satellite unit is a supporting sentence for the nucleus unit and contains less prominent information. It is standard practice that the RST tree is trained and evaluated in a right-heavy binarized manner, resulting in three forms of binary nuclearity relationships between EDUs: Nucleus–Satellite, Satellite–Nucleus, and Nucleus–Nucleus. In this work, eighteen coarse-grained relations are considered as discourse labels, consistent with earlier work (Yu et al., 2018).[2]

Work on RST parsing has been dominated by the bottom-up paradigm (Hernault et al., 2010; Feng and Hirst, 2014; Ji and Eisenstein, 2014; Braud et al., 2017; Morey et al., 2017; Yu et al., 2018). These methods produce very competitive benchmarks, but in practice it is not a straightforward

---

[1] Code and trained models: https://github.com/fajri91/NeuralRST-TopDown

[2] Details of individual relations can be found at: http://www.sfu.ca/rst/index.html

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 715–726
April 19 - 23, 2021.

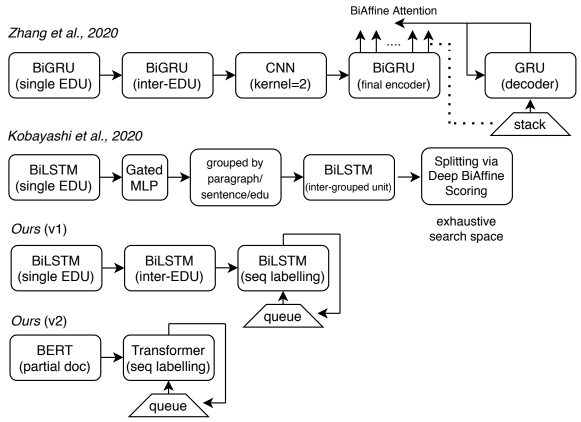


Figure 2: Comparison of our top-down models with Zhang et al. (2020) and Kobayashi et al. (2020).

approach (e.g. transition-based parser with actions prediction steps). Furthermore, bottom-up parsing limits the tree construction to local information, and macro context such as global structure/topic is prone to be under-utilized. As a result, there has recently been a move towards top-down approaches (Kobayashi et al., 2020; Zhang et al., 2020).

The general idea behind top-down parsing is to find splitting points in each iteration of tree construction. In Figure 2, we illustrate how our architecture differs from Zhang et al. (2020) and Kobayashi et al. (2020). First, Zhang et al. (2020) utilize four levels of encoder that comprise 3 Bi-GRUs and 1 CNN layer. The splitting mechanism is applied through a decoder, a stack, and bi-affine attention mechanisms. Kobayashi et al. (2020) use the gold paragraph and sentence boundaries to aggregate a representation for each unit, and generate the tree based on these granularities. Two Bi-LSTMs are used, with splitting points determined by exhaustively calculating the bi-affine score of each possible split. The use of paragraph boundaries can explicitly lower the difficulty of the task, as 77% of paragraphs in the English RST Discourse Treebank ("RST-DT") are actually text spans (Carlson et al., 2001). These boundaries are closely related to gold span boundaries in evaluation.

In this paper, we propose a conceptually simpler top-down approach for RST parsing. The core idea is to frame the problem as a sequence labelling task, where the goal is to iteratively find a segmentation boundary to split a sequence of discourse units into two sub-sequences of discourse units. This way, we are able to simplify the architecture, in eliminating the decoder as well as reducing the search space for splitting points. Specifically, we use an LSTM (Hochreiter and Schmidhuber, 1997) or pre-trained BERT (Devlin et al., 2019) as the segmenter, enhanced in a number of key ways.

Our primary contributions are as follows: (1) we propose a novel top-down approach to RST parsing based on sequence labelling; (2) we explore both traditional sequence models such as LSTMs and also modern pre-trained encoders such as BERT; (3) we demonstrate that adding a weighting mechanism during the splitting of EDU sequences improves performance; and (4) we propose a novel dynamic oracle for training top-down discourse parsers.

2 Related Work

Previous work on RST parsing has been dominated by bottom-up approaches (Hernault et al., 2010; Joty et al., 2013; Li et al., 2016; Braud et al., 2017; Wang et al., 2017). For example, Ji and Eisenstein (2014) introduce `DPLP`, a transition-based parser based on an SVM with representation learning, combined with some heuristic features. Braud et al. (2016) propose joint text segment representation learning for predicting RST discourse trees using a hierarchical Bi-LSTM. Elsewhere, Yu et al. (2018) showed that implicit syntax features extracted from a dependency parser (Dozat and Manning, 2017) are highly effective for discourse parsing.

Top-down parsing is well established for constituency parsing and language modelling (Johnson, 1995; Roark and Johnson, 1999; Roark, 2001; Frost et al., 2007), but relatively new to discourse parsing. Lin et al. (2019) propose a unified framework based on pointer networks for sentence-level discourse parsing, while Liu et al. (2019) employ hierarchical pointer network parsers.

Morey et al. (2017) found that most previous studies on parsing RST discourse tree were incorrectly benchmarked, e.g. one study uses macro-averaging while another use micro-averaging.[3] They also advocate for evaluation based on micro-averaged F-1 scores over labelled attachment decisions (a la the original Parseval).

Pre-trained language models (Radford et al., 2018; Devlin et al., 2019) have been shown to benefit a multitude of NLP tasks, including discourse analysis. For example, BERT models have been used for classifying discourse markers (Sileo et al.,

---

[3] After standardizing evaluation (based on micro-averaged F-1), they found that `DPLP` achieves the best `Full` performance, outperforming the deep learning models.

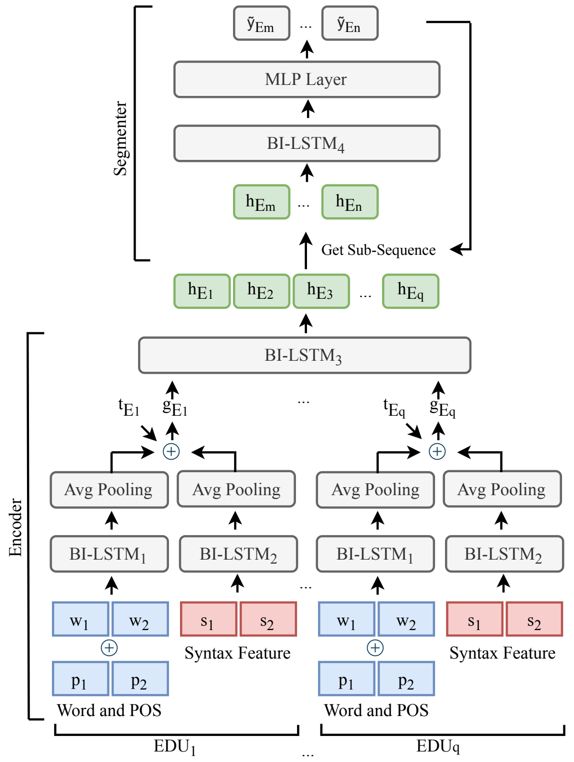


Figure 3: Architecture of the LSTM model.

2019) and discourse relations (Nie et al., 2019; Shi and Demberg, 2019). To the best of our knowledge, however, pre-trained models have not been applied in the generation of full discourse trees, which we address here by experimenting with BERT for top-down RST parsing.

3 Top-down RST Parsing

We frame RST parsing as a sequence labelling task, where given a sequence of input EDUs, the goal is to find a segmentation boundary to split the sequence into two sub-sequences. This is realized by training a sequence labelling model to predict a binary label for each EDU, and select the EDU with the highest probability to be the segmentation point. After the sequence is segmented, we repeat the same process for the two sub-sequences in a divide-and-conquer fashion, until all sequences are segmented into individual units, producing the binary RST tree (e.g. Figure 1).

3.1 LSTM Model

As illustrated in Figure 3, our LSTM parser consists of two main blocks: an encoder and a segmenter. For the encoder, we follow Yu et al. (2018) in using two LSTMs (Bi-LSTM$_1$ and Bi-LSTM$_2$) to produce EDU encodings by processing: (1) x_i, the concatenation of word embedding w_i and POS tag embedding p_i; and (2) syntax embedding s_i, the output of the MLP layer of the bi-affine dependency parser (Dozat and Manning, 2017). Similar to Yu et al. (2018), we then take the average of the output states for both LSTMs over the EDU, and concatenate it with an EDU type embedding t_{E_j} (which distinguishes the last EDU in a paragraph from other EDUs) to produce the final encoding:

$$\begin{aligned} x_i &= w_i \oplus p_i \\ \{a_1^w, .., a_p^w\} &= \text{Bi-LSTM}_1(\{x_1, .., x_p\}) \\ \{a_1^s, ..., a_p^s\} &= \text{Bi-LSTM}_2(\{s_1, .., s_p\}) \\ g_{E_j} &= \text{Avg-Pool}(\{a_1^w, .., a_p^w\}) \oplus \\ &\quad \text{Avg-Pool}(\{a_1^s, .., a_p^s\}) \oplus t_{E_j} \end{aligned} \tag{1}$$

where E_j is an EDU, p is the number of words in E_j, and $\oplus$ denotes the concatenate operation. t_{E_j} is generally an implicit paragraph boundary feature, and provides a fair benchmark with previous models. In Section 4.3, we also show results without paragraph boundary features.

As each EDU is processed independently, we use another LSTM (Bi-LSTM$_3$) to capture the inter-EDU relationship to obtain a contextualized representation h_{E_j}:

$$\{h_{E_1}, ..., h_{E_q}\} = \text{Bi-LSTM}_3(\{g_{E_1}, ..., g_{E_q}\})$$

where q is the number of EDUs in the document. Note that h_{E_j} is the final encoder output (see Figure 3) and is only computed once for each document.

The second part is the segmenter. We frame segmentation as a sequence labelling problem with $y_{E_j} \in \{0, 1\}$, where 1 denotes the splitting point, and 0 a non-splitting point. For each EDU sequence there is exactly one EDU that is labeled 1, and we start from the full EDU sequence (whole document) and iteratively perform segmentation until we are left with individual EDUs. We use a queue to store the two EDU sub-sequences as the result of the segmentation process. In total, there are $q - 1$ iterations of segmentation (recall that q is the total number of EDUs in the document).

As segmentation is done iteratively in a divide-and-conquer fashion, h_{E_j} serves as the input to the segmenter, which takes a (sub)sequence of EDUs to predict the segmentation position:

$$\begin{aligned} \{h'_{E_m}, .., h'_{E_n}\} &= \text{Bi-LSTM}_4(\{h_{E_m}, .., h_{E_n}\}) \\ \tilde{y}_{E_j} &= \sigma(\text{MLP}(h'_{E_j})) \end{aligned}$$

where m/n are the starting/ending index of the EDU sequence,[4] and $\tilde{y}_{E_j}$ gives the probability of a segmentation. From preliminary experiments we found that it's important to have this additional Bi-LSTM$_4$ to perform the EDU sub-sequence segmentation point prediction.

3.2 Transformer Model

Adapting BERT to discourse parsing is not trivial due to the limited number of input tokens it takes (typically 512 tokens), which is often too short for documents. Moreover, BERT is designed to encode sentences (and only two at maximum), where in our case we want to encode sequences of EDUs that span multiple sentences.

In our case, EDU truncation is not an option (since that would produce an incomplete RST tree), and the average number of words per document in our data is 521 (741 word pieces after BERT tokenization), which is much larger than the 512 limit. We therefore break the document into a number of partial documents, each consisting of multiple sentences that fit into the 512 token limit. This way, we allow the model to capture the fine-grained word-to-word relationships across (most) EDUs. Each partial document is then processed based on Liu and Lapata (2019) trick where we use an alternating even/odd segmentation embedding to encode all the EDUs in a document.

We illustrate this approach in Figure 4. First, all EDUs are formatted to start with [CLS] and end with [SEP], and words are tokenized using WordPiece. If the document has more than 512 tokens, we break it into multiple partial documents based on EDU boundaries, and pad accordingly (e.g. in Figure 4 we break the example document of 3 EDUs into 2 partial documents), and process each partial document independently with BERT.

We also experimented with the second alternative by encoding each EDU independently first with BERT, and use a second inter-EDU transformer to capture the relationships between EDUs. Preliminary experiments, however, suggest that this approach produces sub-optimal performance.

In Figure 4 each token is assigned three kinds of embeddings: (1) word, (2) segment, and (3) position. The input vector is computed by summing these three embeddings, and fed into BERT (initialized with `bert-base`). The output of BERT gives us a contextualized embedding for each token, and we use the [CLS] embedding as the encoding for each EDU (g_{E_j}).

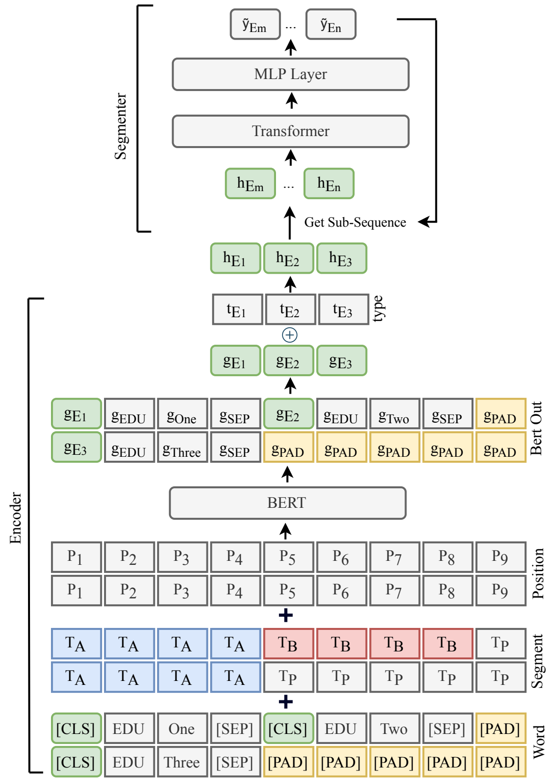


Figure 4: Architecture of the transformer model. In practice, 1 row of input can have more than two EDUs.

Unlike the LSTM model, we do not incorporate syntax embeddings into the transformer model as we found no empirical benefit (see Section 4.3). This observation is in line with other studies (e.g. Jawahar et al. (2019)) that have found BERT to implicit encode syntactic knowledge.

For the segmenter we use a second transformer (initialized with random weights) to capture the inter-EDU relationships for sub-sequences of EDUs during iterative segmentation:

$$\{h'_{E_m}, .., h'_{E_n}\} = \text{transformer}(\{h_{E_m}, .., h_{E_n}\})$$
$$\tilde{y}_{E_j} = \sigma(\text{MLP}(h'_{E_j}))$$

where $\tilde{y}_{E_j}$ gives the probability of a segmentation, and h_{E_j} is the concatenation of the output of BERT (g_{E_j}) and the EDU type embedding (t_{E_j}).

3.3 Nuclearity and Discourse Relation Prediction

In Figure 5, we give an example of the iterative segmentation process to construct the RST tree. In each iteration, we pop a sequence from the queue (initialized with the original sequence of EDUs in

---

[4] In the first iteration, $m = 1$ and $n = q$ (number of EDUs in the document).

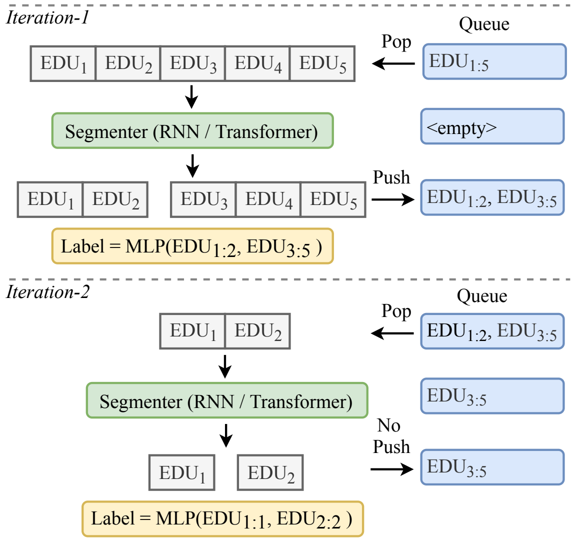


Figure 5: Nuclearity and relation prediction.

the document) and compute the segmentation label for each EDU using an LSTM (Section 3.1) or transformer (Section 3.2). After the sequence is segmented (using the ground truth label during training, or the highest-probability label at test time), we push to the queue the two sub-sequences (if they contain at least two EDUs) and repeat this process until the queue is empty.

In addition to segmentation, we also need to predict the nuclearity/satellite relationship (3 classes) and the discourse label (18 classes) for the segmented pairs. To that end, we average the EDU encodings for the segments, and feed them to a MLP layer to predict the nuclearity and discourse labels:

$$\begin{aligned} u_l &= \text{Avg-Pool}(h'_{E_m}, ..., h'_{E_{m+ind}}) \\ u_r &= \text{Avg-Pool}(h'_{E_{m+ind+1}}, ..., h'_{E_n}) \\ z_{nuc+dis} &= \text{softmax}(\text{MLP}(u_l, u_r)) \end{aligned}$$

where ind is the index of the segmentation point (given by the ground truth during training, or argmax of the segmentation probabilities $\tilde{y}_{E_j}$ at test time), and $z_{nuc+dis}$ gives the joint probability distribution over the nuclearity and discourse classes.[5]

3.4 Segmentation Loss with Penalty

One drawback of the top-down approach is that segmentation errors incurred closer to the root can be detrimental, as the error will propagate to the rest of the sub-trees. To address this, we explore scaling the segmentation loss based on the current tree depth and the number of EDUs in the input sequence. Preliminary experiments found that both approaches work, but that the latter is marginally better, and so we present results using the latter.

Formally, the modified segmentation loss of an example (document) is given as follows:

$$\begin{aligned} L(E_{m:n}) = -\sum_{i=m}^{n} \Big(& y_{E_i} \log(\tilde{y}_{E_i}) + \\ & (1 - y_{E_i}) \log(1 - \tilde{y}_{E_i}) \Big) \\ \mathcal{L}_{seg} = \frac{1}{|S|} & \sum_{(m,n)\in S} (1 + (n-m)^{\beta}) L(E_{m:n}) \end{aligned}$$

where $y_{E_i} \in \{0, 1\}$ is the ground truth segmentation label, $L(E_{m:n})$ is the cross-entropy loss for an EDU sequence, S is the set of all EDU sequences (based on ground truth segmentation), and β is a scaling hyper-parameter.

To summarize, the total training loss of our model is a (weighted) combination of segmentation loss ($\mathcal{L}_{seg}$) and nuclearity-discourse prediction loss ($\mathcal{L}_{nuc+dis}$):

$$\mathcal{L} = \lambda_1 \mathcal{L}_{seg} + \lambda_2 \mathcal{L}_{nuc+dis} \qquad (2)$$

3.5 Dynamic Oracle

The training regimen for discourse parsing creates an exposure bias, where the parser may struggle to recover when it makes a mistake at test time. Goldberg and Nivre (2012) propose a dynamic oracle for transition-based dependency parsing to tackle this. The idea is to allow the model during training to use its predictions (instead of ground truth actions), and introduce a dynamic oracle to find the next best/optimal action sequences. It does so by comparing the current state of the constructed tree and the gold-standard tree, and aims to minimize the deviation. As the model is exposed to prediction errors during training time, it has a better chance of recovering from them at test time.

We explore a similar idea, and propose a dynamic oracle for our top-down discourse parser. A crucial question to ask when designing a dynamic oracle is: *how can we compare the current state to the gold tree to obtain the next best series of actions when an error occurs during training?* In transition-based parsing, Goldberg and Nivre (2012) compute a cost/loss of each transition by

---

[5] We also experimented with predicting the nuclearity and discourse labels separately, but found joint prediction to work better in preliminary experiments.

Algorithm 1 Top-down Dynamic Oracle

```
function DynOracle($E, O, R$)
    # For training only
    # E is list of EDUs
    # O is gold order for segmentation
    # R is list of gold discourse labels based on O
    $q$ = length($E$); $queue = [E_{1:q}]$
    while $queue$ is not empty do
        $E_{m:n} = queue$.pop()
        $id_{gold}, r_{gold}$ = match($E_{m:n}, O, R$)
        $id_{pred}$ = predictSplit($E_{m:n}$)
        $r_{pred1}$ = predictLabel($E_{m:n}, id_{gold}$) # for loss
        $r_{pred2}$ = predictLabel($E_{m:n}, id_{pred}$) # ignored
        if random() $> \alpha$ then
            $L, R$ = separate($E_{m:n}, id_{gold}$)
        else
            $L, R$ = separate($E_{m:n}, id_{pred}$)
        end if
        $queue$.push($L$) if len($L$) $> 1$
        $queue$.push($R$) if len($R$) $> 1$
    end while
end function
```

counting the gold arcs that are no longer reachable based on the action taken (e.g. `SHIFT`, `REDUCE`). We apply similar reasoning when finding the next best segmentation sequence in our dynamic oracle, which we illustrate below with an example.

Say we have a document with 4 EDUs ($E_{1:4}$), and the gold tree given in Figure 6 (left). The correct sequence of segmentation is given by $O_{1:4} = [2, 1, 3, -]$, which means we should first split at E_2 (creating $E_{1:2}$ and $E_{3:4}$), and then at E_1 (creating $E_1, E_2, E_{3:4}$), and lastly at E_3, producing E_1, E_2, E_3, E_4 as the leaves with the gold tree structure. We give the last EDU E_4 a "$-$" label (i.e. $O_4 =$'$-$') because no segmentation is needed for the last EDU.

Suppose the model predicts to do the first segmentation at E_3. This produces $E_{1:3}$ and E_4. What is the best way to segment $E_{1:3}$ to produce a tree that is as close as possible to the gold tree? The canonical segmentation order $O_{1:3}$ is $[2, 1, -]$ (the label of the last EDU is replaced by '$-$'), from which we can see the next best segmentation is to segment at E_2 to create $E_{1:2}$ and E_3. Creating the canonical segmentation order O, and following it as much as possible, ensures the sub-tree that we're creating for $E_{1:3}$ mimics the structure of the gold tree.

The dynamic oracle labels nuclearity-discourse relations following the same idea. We introduce R, a list of gold nuclearity-discourse relations. For our example $R_{1:4} = [r_2, r_1, r_3, -]$ (based on the gold tree; see Figure 6 (left)). If the model decides to first segment at E_3 and creates $E_{1:3}$ and E_4, when we segment at E_2 (next best choice of segmentation), we will follow R and label the nuclearity-discourse relation with r_1. As before, following the original label list R ensures we keep the nuclearity-discourse relation as faithful as possible (Figure 6 (right bottom)).

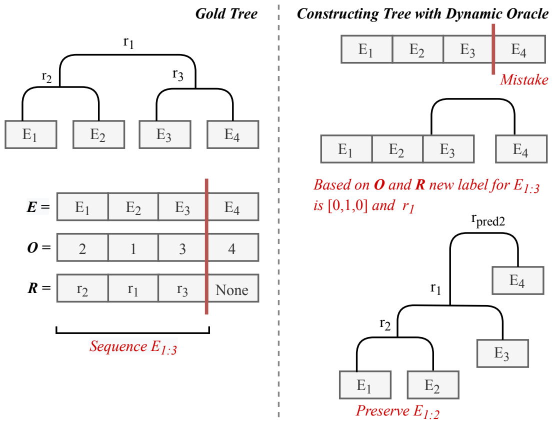


Figure 6: Dynamic oracle for top-down approach.

The dynamic oracle of our top-down parser is arguably quicker than that of a transition-based parser, as we do not need to accumulate cost for every transition taken. Instead, the dynamic oracle simply follows the gold segmentation order O to preserve as many subtrees as possible when an error occurs. We present pseudocode for the proposed dynamic oracle in Algorithm 1.

The probability of using the ground truth segmentation or predicted segmentation during training is controlled by the hyper-parameter $\alpha \in [0, 1]$ (see Algorithm 1). Intuitively, this hyper-parameter allows the model to alternate between exploring its (possibly erroneous) segmentation or learning from the ground truth segmentation. The oracle reverts to its static variant when $\alpha = 0$.

4 Experiments

4.1 Data

We use the English RST Discourse Treebank (Carlson et al., 2001) for our experiments, consistent with recent studies (Ji and Eisenstein, 2014; Li et al., 2014; Feng and Hirst, 2014; Yu et al., 2018). The dataset is based on the Wall Street Journal portion of the Penn Treebank (Marcus et al., 1993), with 347 documents for training, and the remaining 38 documents for testing. We use the same development set as Yu et al. (2018), which consists of 35 documents selected from the training set. We also use the same 18 discourse labels. Stanford

Variant	**LSTM**	**Transformer**
Vanilla	48.4±0.5	51.3±0.2
+Syntax	50.0±0.7	51.9±0.4
+Penalty	49.6±0.5	**52.1±0.4**
+Syntax+Penalty	**51.6±0.1**	51.8±0.8

Table 1: Feature addition study over the development set to find the best configuration for our models. Presented results are the mean and standard deviation of the `Full` metric (micro-averaged F-score on labelled attachment decisions) over three runs.

CoreNLP (Manning et al., 2014) is used for POS tagging.[6]

4.2 Model Configurations

We experiment with two segmentation models — LSTM (Section 3.1) and transformer (Section 3.2) — both implemented in PyTorch framework.[7] As EDUs are provided in the dataset, no automatic segmentation of EDU is required in our experiments.

For the LSTM model, the dimensionality of the Bi-LSTMs in the encoder is 256, while the segmenter (Bi-LSTM$_4$) is 128 (Figure 3). The embedding dimensions of words, POS tags, EDU type, and syntax features are 200, 200, 100, and 1,200, respectively, and we initialize words in EDU with GloVe embedding (Pennington et al., 2014).[8] For hyper-parameters, we use the following: batch size $= 4$, gradient accumulation $= 2$, learning rate $= 0.001$, dropout probability $= 0.5$, and optimizer = Adam (with epsilon of 1e-6). The loss scaling hyper-parameters (Equation (2)), are tuned based on the development set, and set to $\lambda_1 = 1.0$, and $\lambda_2 = 1.0$.

For the transformer model, the document length limit is set to 512 tokens, and longer documents are broken into smaller partial documents. As before, we truncate each EDU to the first 50 words. We initialize the transformer in the encoder with `bert-base`, and the transformer in the segmenter with random weights (Figure 4). The transformer segmenter has 2 layers with 8 heads and 2048 feed-forward hidden size. The training hyper-parameters are: initial learning rate = 5e-5, maximum epochs $= 250$, warm up $= 2000$ steps, and drop out $= 0.2$. For the λ hyper-parameters, we use the same configuration as for the LSTM model.

We tuned the segmentation loss penalty hyper-parameter β (Section 3.4) and the dynamic oracle hyper-parameter α (Section 3.5) based on the development set. Both the LSTM and transformer models use the same $\beta = 0.35$ and $\alpha = 0.65$. We activate the dynamic oracle after training for 50 epochs for both models.

In terms of evaluation, we use the standard metrics introduced by Marcu (2000): `Span`, `Nuclearity`, `Relation`, and `Full`. We report micro-averaged F-1 scores on labelled attachment decisions (original Parseval), following the recommendation of Morey et al. (2017). Additionally, we also present the evaluation with RST-Parseval procedure in Appendix A.

4.3 Results

We first perform a feature addition study over our models to find the best model configuration; results are presented in Table 1. Note that these results are computed over the *development set*, based on a static oracle.

For the vanilla models, the transformer model performs much better than the LSTM model. Adding syntax features (+Syntax) improves both models, although it's more beneficial for the LSTM. A similar trend is observed when we modify the segmentation loss to penalize the model if a segmentation error is made with more EDUs in the input sequence (+Penalty; Section 3.4): the transformer model sees an improvement of +0.8 while the LSTM model improves by +1.2. Lastly, when we combine both syntax features and the segmentation penalty, the LSTM model again shows an appreciable improvement, while the transformer model drops in performance marginally.[9] Given these results, we use both syntax features and the segmentation penalty for the LSTM model, but only the segmentation penalty for the transformer model in the remainder of our experiments.

We next benchmark our models against state-of-the-art RST parsers over the *test set*, as presented in Table 2 (original Parseval) and Table 5 (RST-Parseval as additional result). Except Yu et al. (2018), all bottom-up results are from Morey et al. (2017). We present the labelled attachment decision performance for Yu et al. (2018) by running the code of the authors for three runs and taking

---

[6] `https://stanfordnlp.github.io/CoreNLP`

[7] We use the Huggingface framework for the transformer models.

[8] `https://nlp.stanford.edu/projects/glove`

[9] The result is consistent with the test set (see Appendix B)

Method	**S**	**N**	**R**	**F**
Bottom Up:				
Feng and Hirst (2014)*†	68.6	55.9	45.8	44.6
Ji and Eisenstein (2014)*†	64.1	54.2	46.8	46.3
Surdeanu et al. (2015)*†	65.3	54.2	45.1	44.2
Joty et al. (2015)*	65.1	55.5	45.1	44.3
Hayashi et al. (2016)*	65.1	54.6	44.7	44.1
Li et al. (2016)*	64.5	54.0	38.1	36.6
Braud et al. (2017)*	62.7	54.5	45.5	45.1
Yu et al. (2018) (static)‡	71.1	59.7	48.4	47.4
Yu et al. (2018) (dynamic)‡	71.4	60.3	49.2	48.1
Top Down:				
Zhang et al. (2020)*	67.2	55.5	45.3	44.3
Our model				
Transformer (static)‡	70.6	59.9	50.6	49.0
Transformer (dynamic)‡	70.2	60.1	50.6	49.2
LSTM (static)‡	72.7	61.7	50.5	49.4
LSTM (dynamic)‡	**73.1**	**62.3**	**51.5**	**50.3**
Our best model without paragraph boundary feature				
LSTM (static)	66.3	56.6	47.1	46.1
LSTM (dynamic)	67.3	57.4	48.5	47.4
Human	78.7	66.8	57.1	55.0

Table 2: Results over the test set calculated using micro-averaged F-1 on labelled attachment decisions (original Parseval). All metrics (S: `Span`, N: `Nuclearity`, R: `Relation`, F: `Full`) are averaged over three runs. "*" denotes reported performance. "†" and "‡" denote that the model uses sentence and paragraph boundary features, respectively. In this evaluation, Kobayashi et al. (2020) does not report the original Parseval result.

the average.[10] We also present the reported scores for the other top-down RST parsers (Zhang et al., 2020; Kobayashi et al., 2020).[11] Human performance in Table 2 and Table 5 is the score of human agreement reported by Joty et al. (2015) ad Morey et al. (2017).

Overall, in Table 2 our top-down models (LSTM and transformer) outperform all bottom-up and top-down baselines across all metrics. As we saw in the feature addition study, the LSTM model outperforms the transformer model, even though the transformer uses pre-trained BERT. We hypothesize that this may be because BERT is trained over shorter texts (paragraphs or sentence pairs), while our documents are considerably longer. Also, due to memory constraints, we break long documents into partial documents (Section 3.2), limiting fine-grained word-to-word attention to only nearby EDUs.

#EDUs	**#Docs**	**#Spans**	**Type**	**S**	**N**	**R**
$(0, 50]$	21	404	Static	81.0	72.0	58.9
			Dynamic	79.3	71.6	**59.1**
$(50, 100]$	9	639	Static	76.8	66.4	56.2
			Dynamic	**79.3**	**69.2**	**59.2**
$(100, 150]$	5	604	Static	69.6	58.4	49.1
			Dynamic	**70.3**	57.4	49.1
$(150, \infty)$	3	661	Static	66.6	53.9	41.0
			Dynamic	66.0	**54.4**	**41.8**

Table 3: Impact of the dynamic oracle over documents of differing length. Scores (micro-averaged F-1 on labelled attachment decisions) are averaged over three runs on the test set.

In Table 2, we also present results for our model without paragraph features, and compare against other models which don't use paragraph features (each marked with "‡").[12] First, we observe that our best model substantially outperforms all models with paragraph boundary features in terms of the `Full` metric. Compared to Zhang et al. (2020), our models (without this feature) achieve an improvement of $+0.1$, $+1.9$, $+3.2$, and $+3.1$ for `Span`, `Nuclearity`, `Relation`, and `Full` respectively.

5 Analysis

In Table 3 we present the impact of the dynamic oracle over documents of differing length for the LSTM model. Generally, we found that the static model performs better for shorter documents, and the dynamic oracle is more effective for longer documents. For instance, for documents with 50–100 EDUs, the dynamic oracle improves the `Span`, `Nuclearity`, and `Relation` metrics substantially. We also observe that the longer the document, the more difficult the tree prediction is. It is confirmed by the decreasing trends of all metrics for longer documents in Table 3.

In total, our best model obtains 1,698 out of 2,308 spans of original Parseval trees, and correctly predict 1,517 segmentation points (pairs). We further analyze these pairs by presenting the confusion matrices of nuclearity and relation prediction in Figure 7 and Figure 8, respectively.

First, the model tends to have more errors on `NN` (Nucleus–Nucleus) prediction where 53 span

---

[10] `https://github.com/yunan4nlp/NNDisParser`.

[11] Neither Zhang et al. (2020) nor Kobayashi et al. (2020) released their code, so we were unable to rerun their models.

[12] Yu et al. (2018) use paragraph boundary features in their original code, but do not report it in the paper.

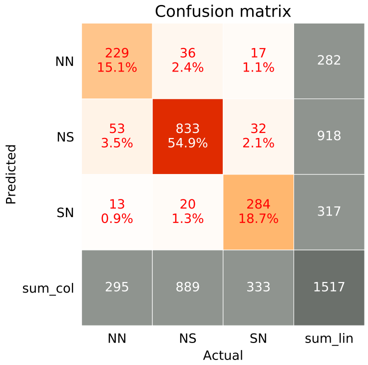


Figure 7: Confusion matrix of nuclearity prediction over the test set (`NS` = Nucleus-Satellite, `NN` = Nucleus-Nucleus, `SN` = Satellite-Nucleus).

EDU-1	EDU-2	Actual	Pred
senator sasser of tennessee is chairman of the appropriations subcommittee on military construction;	mr. bush 's $ 87 million request for tennessee increased to $ 109 million.	`back`	`elab`
a law went in the books in january that let him smoke bacon without breeding pigs.	he chased in	`cause`	`elab`
that 's the rule.	that 's the market.	`list`	`elab`

Table 4: Examples of misclassified relations.

pairs (18% of `NN`) are classified as `NS` (Nucleus–Satellite). Class imbalance in the training set (`NN`:`NS`:`SN` = 23:61:16) is the main factor that drives the model to favor `NS` over the other classes.

In Figure 8 we present analysis over top-7 relations and a relation `other` that represents the rest of 11 classes. Similar to the nuclearity prediction, the relation class distribution is also imbalance where `elab` accounts for 37% of the examples. Some relations are related to `elab` (see Table 4 for examples), such as `back`, `cause`, and `list` which we see some false positives. This produces the low precision of `elab` (74%). Unlike `elab`, relation `attr` is also a major class (represents 14% of the training data) but its precision and recall is substantially higher, at 94% and 96% respectively, suggesting it is less ambiguous. For `other`, its recall is 45%, and most of the errors are classified as `elab` (31%).

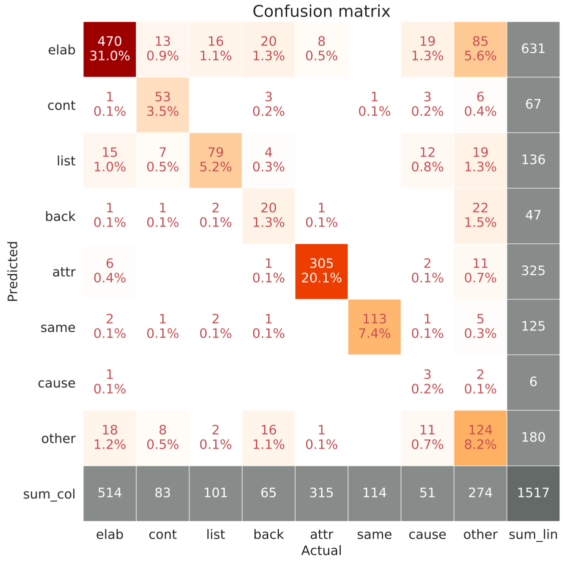


Figure 8: Confusion matrix of relation prediction over the test set with top-7 relations (`elab` = Elaboration, `cont` = Contrast, `list` = List, `back` = Background, `same` = Same, `temp` = Temporal, `eval` = Evaluation, `other` = Other 11 relations).

6 Conclusion

We introduce a top-down approach for RST parsing via sequence labelling. Our model is conceptually simpler than previous top-down discourse parsers and can leverage pre-trained language models such as BERT. We additionally propose a dynamic-oracle for our top-down parser, and demonstrate that our best model achieves a new state-of-the-art for RST parsing.

Acknowledgments

We are grateful to the anonymous reviewers for their helpful feedback and suggestions. In this research, the first author is supported by the Australia Awards Scholarship (AAS), funded by the Department of Foreign Affairs and Trade (DFAT), Australia. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at The University of Melbourne. This facility was established with the assistance of LIEF Grant LE170100200.

References

Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better document-level sentiment analysis from RST discourse parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2212–2218.

Chloé Braud, Maximin Coavoux, and Anders Søgaard. 2017. Cross-lingual RST discourse parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 292–304, Valencia, Spain. Association for Computational Linguistics.

Chloé Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of RST discourse parsers. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1903–1913, Osaka, Japan. The COLING 2016 Organizing Committee.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *SIGDIAL '01 Proceedings of the Second SIGdial Workshop on Discourse and Dialogue - Volume 16*, pages 1–10.

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

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 2016 International Conference on Learning Representations*, pages 1–8.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 511–521, Baltimore, Maryland. Association for Computational Linguistics.

Richard Frost, Rahmatullah Hafiz, and Paul Callaghan. 2007. Modular and efficient top-down parsing for ambiguous left-recursive grammars. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 109–120, Prague, Czech Republic. Association for Computational Linguistics.

Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*, pages 959–976, Mumbai, India. The COLING 2012 Organizing Committee.

Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. 2016. Empirical comparison of dependency conversions for RST discourse trees. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 128–136, Los Angeles. Association for Computational Linguistics.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3):1–33.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Mark Johnson. 1995. Squibs and discussions: Memoization in top-down parsing. *Computational Linguistics*, 21(3):405–417.

Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Yashar Mehdad. 2013. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 486–496, Sofia, Bulgaria. Association for Computational Linguistics.

Shafiq Joty, Giuseppe Carenini, and Raymond T. Ng. 2015. CODRA: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 41(3):385–435.

Naoki Kobayashi, Tsutomu Hirao, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2020. Top-down RST parsing utilizing granularity levels in documents. In *AAAI 2020 : The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2019. Improved document modelling with a neural discourse parser. In *Proceedings of the Australasian Language Technology Association Workshop 2019*, Sydney, Australia.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2061–2069, Doha, Qatar. Association for Computational Linguistics.

Qi Li, Tianshi Li, and Baobao Chang. 2016. Discourse parsing with attention-based hierarchical neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 362–371, Austin, Texas. Association for Computational Linguistics.

Xiang Lin, Shafiq Joty, Prathyusha Jwalapuram, and M Saiful Bari. 2019. A unified linear-time framework for sentence-level discourse parsing. In *Proceedings of the 57th Annual Meeting of the Asso-*

ciation for Computational Linguistics, pages 4190–4200, Florence, Italy. Association for Computational Linguistics.

Linlin Liu, Xiang Lin, Shafiq Joty, Simeng Han, and Lidong Bing. 2019. Hierarchical pointer net parsing. In *2019 Conference on Empirical Methods in Natural Language Processing*, pages 1007–1017.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *2019 Conference on Empirical Methods in Natural Language Processing*.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text Interdisciplinary Journal for the Study of Discourse*, pages 243–281.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, USA.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Mathieu Morey, Philippe Muller, and Nicholas Asher. 2017. How much progress have we made on RST discourse parsing? a replication study of recent results on the RST-DT. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1319–1324, Copenhagen, Denmark. Association for Computational Linguistics.

Allen Nie, Erin Bennett, and Noah Goodman. 2019. DisSent: Learning sentence representations from explicit discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4497–4510, Florence, Italy. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *CoRR, abs/1704.01444, 2017*.

Brian Roark. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249–276.

Brian Roark and Mark Johnson. 1999. Efficient probabilistic top-down and left-corner parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 421–428, College Park, Maryland, USA. Association for Computational Linguistics.

Wei Shi and Vera Demberg. 2019. Next sentence prediction helps implicit discourse relation classification within and across domains. In *2019 Conference on Empirical Methods in Natural Language Processing*, pages 5789–5795.

Damien Sileo, Tim Van De Cruys, Camille Pradel, and Philippe Muller. 2019. Mining discourse markers for unsupervised sentence representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3477–3486, Minneapolis, Minnesota. Association for Computational Linguistics.

Mihai Surdeanu, Tom Hicks, and Marco Antonio Valenzuela-Escárcega. 2015. Two practical rhetorical structure theory parsers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 1–5, Denver, Colorado. Association for Computational Linguistics.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 184–188, Vancouver, Canada. Association for Computational Linguistics.

Nan Yu, Meishan Zhang, and Guohong Fu. 2018. Transition-based neural RST parsing with implicit syntax features. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 559–570.

Nan Yu, Meishan Zhang, and Guohong Fu. 2018. Transition-based neural RST parsing with implicit syntax features. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 559–570, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Longyin Zhang, Yuqing Xing, Fang Kong, Peifeng Li, and Guodong Zhou. 2020. A top-down neural architecture towards text-level parsing of discourse rhetorical structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6386–6395, Online. Association for Computational Linguistics.

A Evaluation with RST-Parseval Procedure

Method	S	N	R	F
Bottom-Up				
Feng and Hirst (2014)*†	84.3	69.4	56.9	56.2
Ji and Eisenstein (2014)*†	82.0	68.2	57.8	57.6
Surdeanu et al. (2015)*†	82.6	67.1	55.4	54.9
Joty et al. (2015)*	82.6	68.3	55.8	54.4
Hayashi et al. (2016)*	82.6	66.6	54.6	54.3
Li et al. (2016)*	82.2	66.5	51.4	50.6
Braud et al. (2017)*	81.3	68.1	56.3	56.0
Yu et al. (2018) (1 run)*‡	85.5	73.1	60.2	59.9
Yu et al. (2018) (static)‡	85.8	72.6	59.5	59.0
Yu et al. (2018) (dynamic)‡	85.6	72.9	59.8	59.3
Top-Down				
Kobayashi et al. (2020)*†‡	**87.0**	**74.6**	60.0	-
Our model				
Transformer (static)‡	85.2	72.0	60.3	59.6
Transformer (dynamic)‡	85.5	72.3	60.5	59.9
LSTM (static)‡	86.4	73.4	60.8	60.3
LSTM (dynamic)‡	86.6	73.7	**61.5**	**60.9**
Our best model without boundary feature				
LSTM (static)	83.2	70.4	58.4	57.9
LSTM (dynamic)	83.6	70.4	58.8	58.2
Human	88.3	77.3	65.4	64.7

Table 5: Results over the test set calculated using micro-averaged F-1 on RST-Parseval. All metrics (S: `Span`, N: `Nuclearity`, R: `Relation`, F: `Full`) are averaged over three runs. "*" denotes reported performance. "†" and "‡" denote that the model uses sentence and paragraph boundary features, respectively. In this evaluation, Zhang et al. (2020) does not report the RST-Parseval result. Also, both Zhang et al. (2020); Kobayashi et al. (2020) do not release their code. First, We can see that our model achieves the best results in terms of `Relation` and `Full`. Compared to other models without paragraph boundary features, our proposed model also performs best on the `Full` metric by a comfortable margin.

B Feature Addition on the Test Set

Variants	LSTM	Transformer
Vanilla	46.5±0.4	48.0±0.4
+Syntax	48.2±0.7	49.3±1.0
+Penalty	46.8±0.8	**49.0±0.2**
+Syntax+Penalty	**49.4±0.4**	48.7±0.6

Table 6: Feature addition over the test set to find the best configuration for our models. Presented results are the mean and standard deviation of the `Full` metric (micro-averaged F-score on labelled attachment decisions) over three runs.

C Model Configuration for Training

Configuration	Value
LSTM1, LSTM2, LSTM3	200, **256**
LSTM4	100, **128**, 200
Word embedding	**200**
POS embedding	**200**
EDU type embedding	**100**
Syntax Feature	**1200**
λ_1	**1.0**, 1.2
λ_2	0.6, 0.8, **1.0**, 1.2
β (Loss penalty)	0, 0.2, **0.35**, 0.4, 0.6, 0.8, 1.0
α (Dynamic oracle)	0, 0.25, 0.5, **0.65**, 0.75, 1.0
Batch size	2, **4**
Gradient accumulation	**2**, 4
Learning rate	**0.001**
Dropout probability	**0.5**
Infrastructure	1 GPU V100 (16 GB)
Metrics to evaluate	`Full`

Table 7: Parameter trials of our LSTM model. Bold indicates the best value after tuning over the development set.

Configuration	Value
BERT encoder	BERT-Base
Transformer2	L=2, H=8, FF=2048
EDU type embedding	**100**
λ_1	0.5, **1.0**, 1.5
λ_2	0.6, 0.8, **1.0**, 1.5
β (Loss penalty)	0, 0.2, **0.35**, 0.4, 0.6, 0.8, 1.0
α (Dynamic oracle)	0, 0.25, 0.5, **0.65**, 0.75, 1.0
Batch size	1, **2**
Gradient accumulation	2, 3, **4**, 8
Learning rate	**5e-5**
Dropout probability	0.1, 0.3, 0.4, **0.5**
Infrastructure	4 GPU V100 (16 GB)
Metrics to evaluate	`Full`

Table 8: Parameter trials of our transformer model. Bold indicates the best value after tuning over the development set.

Method	1 Epoch	Converge at Epoch
Transition-based	5 mins	60–70
LSTM (Ours)	1 min 21 secs	60–70
Transformer (Ours)	1 min	130-150

Table 9: Running time of the static models during the training. The transition-based model is Yu et al. (2018)

Does the Order of Training Samples Matter? Improving Neural Data-to-Text Generation with Curriculum Learning

Ernie Chang, Hui-Syuan Yeh, Vera Demberg
Dept. of Language Science and Technology, Saarland University
{cychang,yehhui,vera}@coli.uni-saarland.de

Abstract

Recent advancements in data-to-text generation largely take on the form of neural end-to-end systems. Efforts have been dedicated to improving text generation systems by changing the order of training samples in a process known as *curriculum learning*. Past research on sequence-to-sequence learning showed that *curriculum learning* helps to improve both the performance and convergence speed. In this work, we delve into the same idea surrounding the training samples consisting of structured data and text pairs, where at each update, the curriculum framework selects training samples based on the model's competence. Specifically, we experiment with various difficulty metrics and put forward a *soft edit distance metric* for ranking training samples. Our benchmarks show faster convergence speed where training time is reduced by 38.7% and performance is boosted by 4.84 BLEU.

1 Introduction

Neural data-to-text generation has been the subject of much recent research. The task aims at transforming source-side structured data into target-side natural language descriptions (Reiter and Dale, 2000; Barzilay and Lapata, 2005). The process typically involves mini-matches which are randomly sampled with a fixed size from the training set to feed into the model at each training step. In this paper, we apply curriculum learning to this process, which was explored in neural machine translation (Platanios et al., 2019; Zhou et al., 2020), and show how it can help in neural data-to-text generation.

The main idea in curriculum learning is to present the training data in a specific order, starting from *easy* examples and moving on to more *difficult* ones, as the learner becomes more *competent*. When starting out with easier instances, the risk of getting stuck in local optima early on in training is reduced, since the loss functions in neural models are typically highly non-convex (Bengio et al., 2009). This learning paradigm enables flexible batch configurations by considering the material properties as well as the state of the learner. The idea brings in two potential benefits: (1) It speeds up the convergence and reduces the computational cost. (2) It boosts the model performance, without having to change the model or add data.

With the release of large data-to-text datasets (e.g. Wikibio (Lebret et al., 2016), Totto (Parikh et al., 2020), E2E (Novikova et al., 2017)), neural data-to-text generation is now at a point where training speed and the order of samples may begin to make a real difference. We here show the efficacy of curriculum learning with a general LSTM-based sequence-to-sequence model and define *difficulty metrics* that can assess the training instances, using a sucessful *competence function* which estimates the model capability during training. Such metrics have not yet been explored in neural data-to-text generation.

In this paper, we explore the effectiveness of various difficulty metrics and propose a soft edit distance metric, which leads to substantial improvements over other metrics. Crucially, we observe that difficulty metrics that consider data-text samples jointly lead to stronger improvements than metrics that consider text or data samples alone. In summary, this work makes the following contributions towards neural data-to-text generation:

1. We show that by simply changing the order of samples during training, neural models can be improved via the use of curriculum learning.

2. We explore various *difficulty metrics* at the level of the data, text, and data-text pairs, and propose an effective novel metric.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 727–733
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2 Related work

The idea of teaching algorithms in a similar manner as humans, incrementally from easy concepts to more difficult ones dates back to *incremental learning*, which was discussed in light of theories of cognitive development relating to the processes of acquisition in young children (Elman, 1993; Krueger and Dayan, 2009; Plunkett and Marchman, 1993). Bengio et al. (2009) first demonstrated empirically that curriculum learning approaches can decrease training times and improve generalization; later approach address these issues by changing the mini-batch sampling strategy to also include model competence (Kocmi and Bojar, 2017; Zhou et al., 2020; Platanios et al., 2019; Liu et al., 2020; Zhang et al., 2018, 2019). While sample difficulty can be assessed for text samples and data samples or jointly, various measures have been proposed for text samples including n-gram frequency Haffari (2009); Platanios et al. (2019), token rarity, and sentence length (Liu et al., 2020; Platanios et al., 2019). Our approach considers data and text jointly, similar to edit distance metric – Levenshtein (Levenshtein, 1966) and Damerau-Levenshtein Distance (Damerau, 1964; Brill and Moore, 2000a), which was used as a content ordering metric in Wiseman et al. (2017) to measure the extent of alignment between data slots and text tokens.

3 Preliminaries of Curriculum Learning

We base our curriculum learning framework on the two standard components: (1) model competence (how capable the current model is at time t), and (2) sample difficulty, which makes independent judgement on each sample's difficulty. Specifically, we adopt the competence function $c(t)$ for a model at time t as in Platanios et al. (2019); Liu et al. (2020):

$$c_{\text{sqrt}}(t) \in (0,1] = \min\left(1, \sqrt{t\frac{1-c_0^2}{\lambda_t} + c_0^2}\right). \tag{1}$$

Where λ_t is a hyperparameter defining the length of the curriculum and is set to 2.5 as in Liu et al. (2020). $c_0 = 0.1$ as Platanios et al. (2019). Following this formulation, the number of new training examples per unit time is reduced as training progresses to give the learner sufficient time to obtain new knowledge. The sequence-to-sequence model learns using the curriculum as outlined in Algorithm 1 by primarily making batch-wise decisions about which samples to add to each batch. This decision is determined by comparing the *competence* score with the *difficulty* score as shown in Algorithm 1.

Algorithm 1: Curriculum Learning Algorithm

```
Input: Training set, $\mathcal{D} = \{s_d, s_t\}_{i=1}^{M}$, consisting of
       $M$ samples, model ($\mathcal{T}$), difficulty metric ($d$),
       and competence function ($c$).
Compute the difficulty, $d(s_i)$, for each data-text pair
    $\in \mathcal{D}$ (Section 4).
Compute the CDF score $\bar{d}(s_i)$ of $d(s_i)$, where
    $\bar{d}(s_i) \in [0,1]$ (See Figure 2).
for training step $t = 1, \ldots$ do
    Compute the model competence $c(t)$ with $\mathcal{T}$.
    Train $\mathcal{T}$ on sampled data batch, $B_t$, drawn
      uniformly from all $s_i \in \mathcal{D}$, such that
      $\bar{d}(s_i) \leq c(t)$.
    if $c(t) = 1$ then
        break.
Output: Trained model.
```

4 Difficulty Metrics

For ease of discussion, we denote sequence to be s, which can be either data or text, or their concatenation. For comparison, the difficulty metrics use the unit tokens as tokenized by SpaCy[1]. We begin with discussion on *length* and *word rarity*, which were previously applied by Kocmi and Bojar (2017); Platanios et al. (2019) on text sentences.

Length. Length-based difficulty is based on the intuition that longer sequences are harder to encode, and that early errors may propagate during the decoding process, making longer sentences also harder to generate. It is defined as:

$$d_{\text{length}}(s) = N. \tag{2}$$

Rarity. *Word rarity* of a sentence is defined as the product of the unigram probabilities (Platanios et al., 2019). This metric implicitly incorporates information about sentence length since longer sentence scores are sum of more terms and are thus likely to be larger. The difficulty metric for word rarity of a sequence s is defined as:

$$d_{\text{rarity}}(s) = -\sum_{k=1}^{N} \log p(w_k). \tag{3}$$

---

[1] `https://spacy.io/api/tokenizer`

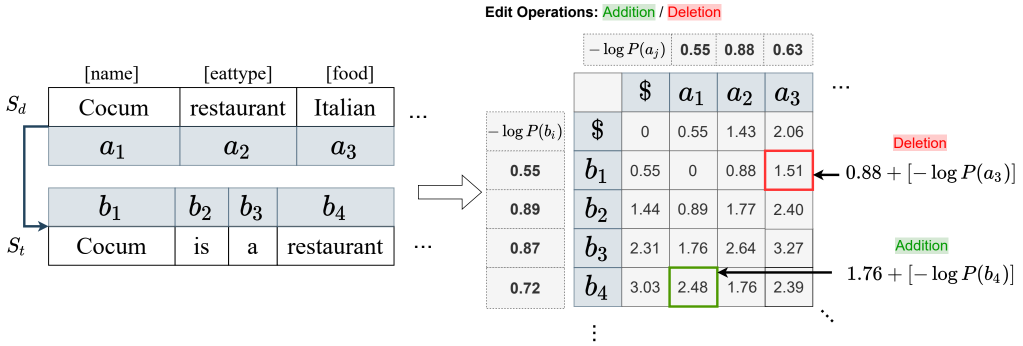


Figure 1: Depiction of the process of *soft edit distance metric* with the *Wagner-Fischer table*. Each cell in the table represents the edit distance to convert data substring's into the text substring.

Damerau-Levenshtein Distance. To consider data and text jointly, we measure the alignment between data slots and text using *the Damerau-Levenshtein Distance* (d_{dld}) (Brill and Moore, 2000a).

We calculate the minimum number of edit operations needed to transform data (s_d) into text (s_t)[2], and relies only four operations: (a) substitute a word in s_d to a different word, (b) insert a word into s_d, (c) delete a word of s_d, and (d) transpose two adjacent words of s_d. The process involves recursive calls that compute distance between substrings $s_d^i \in s_d$ and $s_t^i \in s_t$ at i^{th} comparison.

Soft Data-to-Text Edit Distance. We here present the proposed *soft edit distance (SED)*: (1) We include the basic *add* and *delete* edit operations as in the Levenshtein Distance (Levenshtein, 1966), which was used in Levenshtein Transformer (Gu et al., 2019) as the only two necessary operations for decoding sequences since it correlates well with human text writing where humans "*can revise, replace, revoke or delete any part of their generated text*". We call this variant the plain edit distance *(PED)*. (2) Next, we weight the indicator function $\mathbb{1}(s_d^i, s_t^i)$ for each edit operation with the negative logarithmic unigram probability $-\log p(w)$ for each token $w \in s^i_{\{t|d\}}$, in order to incorporate the idea of *word rarity* into the edit distance metric. For the *delete* operation, we use the $w \in s_d^i$ and for *add* operations, we use $w \in s_t^i$. This is unlike the previous proposal by Brill and Moore (2000b), in which edits are weighted by the token transition probabilities – this is not suitable for our scenario because there is no natural order of the slot sequence in data samples.

The soft distance metric d_{sed} is in principle similar to calculating the logarithmic sum as defined in the *rarity* function, but instead incrementally compares all substrings and calculates their edit distances. This way, d_{sed} includes the information on *length*, *rarity* but also combining the edit operations. We show this process in Figure 5.

Note that we can compute *length* and *rarity* on the concatenation of input data and text sequence, or as individual sequences; whereas *Damerau-Levenshtein distance* and *soft edit distance* are computed jointly on data and text.

5 Experiment Setting

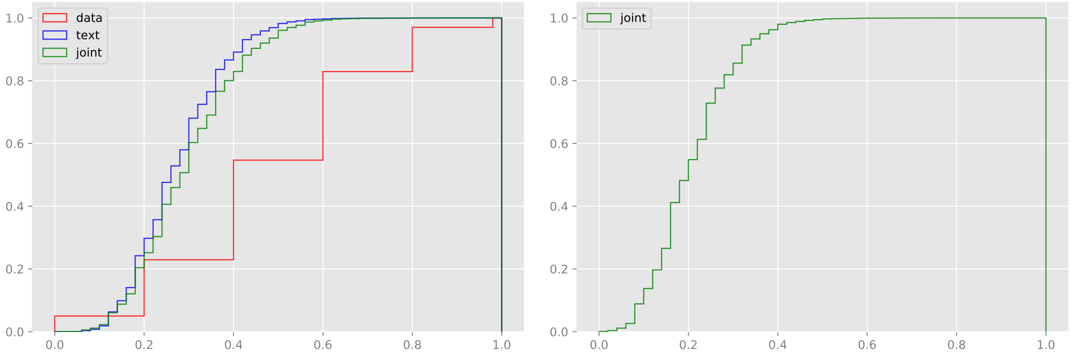
Data. We conduct experiments on the E2E (Novikova et al., 2017) and WebNLG (Colin et al., 2016) datasets. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The inputs are dialogue acts consisting of three to 8 slot-value pairs. WebNLG contains 25k instances describing entities belonging to 15 distinct DBpedia categories, where the data contain are up to 7 RDF triples of the form *(subject, relation, object)*.

Configurations. The LSTM-based model is implemented based on PyTorch (Paszke et al., 2019). We use 200-dimensional token embeddings and the Adam optimizer with an initial learning rate at 0.0001. Batch size is kept at 28, and we decode with beam search with size 5. The performance scores are averaged over 5 random initialization runs.

Settings. We first perform ablation studies (Table 2) on the impact of difficulty metrics on data, text or both (joint). We also analyse the average bin size for each metric – a metric that gives the same score to many instances creates large bins.

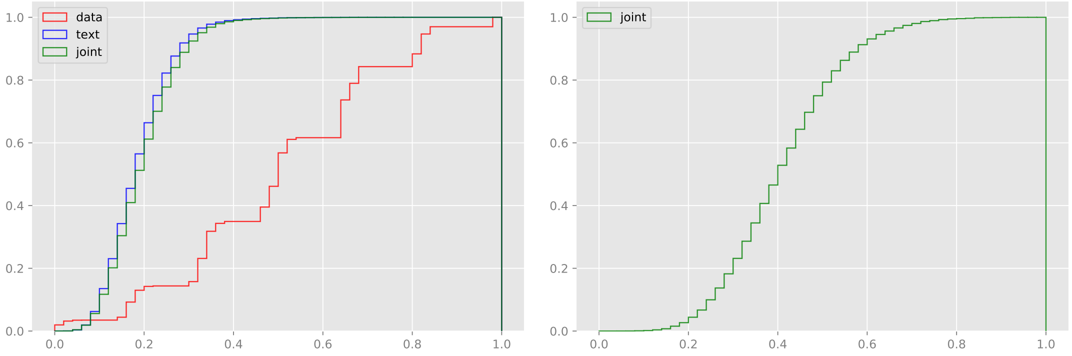
---

[2] Previous work applies *the Damerau-Levenshtein Distance* to slots in data (e.g. "[name]" in Figure 5) and extracts slots from text.



(a) *length*

(c) *Damerau-Levenshtein distance*



(b) *rarity*

(d) *soft edit distance*

Figure 2: The histogram of the cumulative density function for difficulty metrics.

This means that the order of samples within the bin will still be random. On the other hand, a metric that assigns a lot of different difficulty scores to the instances can yield a more complete ordering (and a smaller step size in moving from one level of difficulty to the next). We present the change in performance (BLEU) as the training progresses in order to compare the various difficulty metrics on both datasets (See Figure 3).

6 Results & Analysis

On Table 3, we observe that *soft edit distance* (SED) yields the best performance, outperforming a model that does not use curriculum learning by as much as 2.42 BLEU. It also outperforms all other metrics by roughly 1 BLEU. In general, we see that models perform better on *joint* and *text* than on *data*. This correlates to how a difficulty function is related to the *average bin sizes* of scores it generates. We see that for models that distinguish samples in a more defined manner, it will have a smaller average bin size where probability of having more difficult samples at every confidence threshold is lower. From this, we see that *length* and *DLD* have larger average bin sizes across its difficulty scores and this makes samples less distinguishable from one another. Thus, they result in the smallest improvement over *plain*. We show reordered samples in Table 1 for all difficulty metrics computed *jointly* on data and text. We include length (**L**), rarity (**R**), Damerau-Levenshtein Distance (**DLD**), and the proposed soft edit distance (**SED**).

On the other hand, we also justify the use of weighting for edit operation where *PED*, which is the "hard" variant of *SED* that does not weight edit operations like *SED*, is shown to be far inferior to that of *SED*. The score margin comes up to 2.81 BLEU. Moreover, we further examine the difference in sample orders and observe that *SED* yields more intuitive and better sample ordering as opposed to other metrics.

Human Evaluation. For *human evaluation*, three annotators are instructed to evaluate 100 samples from the *joint* variant to see (1) if the text is *fluent* (score 0-5 with 5 being fully fluent), (2) if it *misses* information contained in the source data and (3) if it includes *wrong* information. These scores are averaged and presented in Table 2.

Metric	Score	Data & Text
L	7	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	43	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
R	17.40	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	173.73	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
DLD	3	**[Data]** name[Wildwood], eatType[restaurant], familyFriendly[yes] **[Text]** Wildwood restaurant is kid-friendly.
	28	**[Data]** name[Cotto], eatType[coffee shop], food[Indian], priceRange[more than £30], customer rating[high], area[riverside], near[The Portland Arms] **[Text]** Cotto is a coffee shop that offers delicious Indian food, although the price range is high, you will be pleased to know Cotto has high customer ratings. It is located in Riverside near The Portland Arms.
SED	17.20	**[Data]** name[The Punter], food[English], priceRange[high] **[Text]** The Punter is a restaurant with high prices.
	87.68	**[Data]** name[The Wrestlers], eatType[coffee shop], food[Italian], priceRange[less than £20], area[city centre], familyFriendly[no], near[Raja Indian Cuisine] **[Text]** The Wrestlers is a coffee shop located in the center of the city that is near Raja Indian Cuisine. It is not family-friendly and the price range is less than 20 pounds.

Table 1: Ordered samples with difficulty metrics.

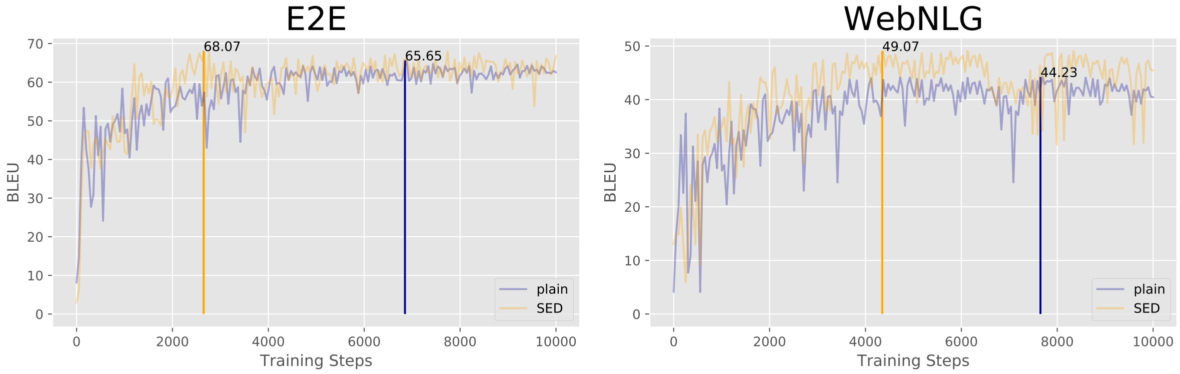


Figure 3: A plot of performance (BLEU) versus the number of steps for E2E and WebNLG datasets. **Vertical bars** indicate where the maximum BLEU scores are attained for **plain** and **SED**.

	source	plain	L	R	DLD	PED	SED
BLEU	data	|	65.34	66.41	-	-	
	text	65.65	65.77	67.01	-	-	-
	joint	|	66.21	67.29	66.58	66.26	**68.07**
Bin Size	data	-	7010.17	385.88	-	-	-
	text	-	737.91	1.04	-	-	-
	joint	-	25.0	1.04	32.26	21.74	**1.02**
Human	fluency	4.35	4.28	4.32	**4.60**	4.23	4.54
	miss	22	16	15	11	14	**9**
	wrong	9	5	7	10	6	**4**

Table 2: Ablation studies for the impact of difficulty metrics on data, text or both (joint) with normalized scores including length (**L**), rarity (**R**), Damerau-Levenshtein Distance (**DLD**), plain edit distance (**PED**), and the proposed soft edit distance (**SED**). **Plain** means no curriculum learning techniques are added. All scores are computed based on the E2E corpus, consisting of both performance (BLEU) and the **average bin size**. Each *bin* is defined by the number of training samples with the same difficulty scores.

On Training Speed. We define speed by the number of updates it takes to reach a performance plateau. On Figure 3, the speedup is measured by the difference between the vertical bars. It can be observed that curriculum learning reduces the training steps to converge, where it consists of up to 38.7% of the total updates for the same model without curriculum learning (on E2E). Further, we see that the use of curriculum learning yields slightly worse performance in the initial training steps, but rise to a higher score and flattens as it converges.

7 Conclusion

To conclude, we show that the sample order does indeed matter when taking into account model competence during training. Further, we demonstrate that the proposed metrics are effective in speeding up model convergence. Given that curriculum learning can be combined with pretty much any neural architecture, we recommend the use of curriculum learning for data-to-text generation. We believe this work offers insights into the annotation process of data with text labels where reduced number of labels are needed.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248

“Foundations of Perspicuous Software Systems”. We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Eric Brill and Robert C Moore. 2000a. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th annual meeting of the association for computational linguistics*, pages 286–293.

Eric Brill and Robert C. Moore. 2000b. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong. Association for Computational Linguistics.

Emilie Colin, Claire Gardent, Yassine M’rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Jeffrey L Elman. 1993. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, pages 11181–11191.

Gholam Reza Haffari. 2009. *Machine learning approaches for dealing with limited bilingual data in statistical machine translation*. Ph.D. thesis, School of Computing Science-Simon Fraser University.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 379–386.

Kai A Krueger and Peter Dayan. 2009. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Xuebo Liu, Houtim Lai, Derek F Wong, and Lidia S Chao. 2020. Norm-based curriculum learning for neural machine translation. *arXiv preprint arXiv:2006.02014*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M. Mitchell. 2019. Competence-based curriculum learning for neural machine translation. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:1162–1172.

Kim Plunkett and Virginia Marchman. 1993. From rote learning to system building: Acquiring verb morphology in children and connectionist nets. *Cognition*, 48(1):21–69.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat.

2018. An empirical exploration of curriculum learning for neural machine translation. *arXiv preprint arXiv:1811.00739*.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1903–1915.

Yikai Zhou, Baosong Yang, Derek F Wong, Yu Wan, and Lidia S Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6934–6944.

TrNews: Heterogeneous User-Interest Transfer Learning for News Recommendation

Guangneng Hu
HKUST / Hong Kong, China
`njuhgn@gmail.com`

Qiang Yang
HKUST / Hong Kong, China
`qyang@cse.ust.hk`

Abstract

We investigate how to solve the cross-corpus news recommendation for unseen users in the future. This is a problem where traditional content-based recommendation techniques often fail. Luckily, in real-world recommendation services, some publisher (e.g., Daily news) may have accumulated a large corpus with lots of consumers which can be used for a newly deployed publisher (e.g., Political news). To take advantage of the existing corpus, we propose a transfer learning model (dubbed as TrNews) for news recommendation to transfer the knowledge from a source corpus to a target corpus. To tackle the heterogeneity of different user interests and of different word distributions across corpora, we design a translator-based transfer-learning strategy to learn a representation mapping between source and target corpora. The learned translator can be used to generate representations for unseen users in the future. We show through experiments on real-world datasets that TrNews is better than various baselines in terms of four metrics. We also show that our translator is effective among existing transfer strategies.

1 Introduction

News recommendation is key to satisfying users' information need for online services. Some news articles, such as breaking news, are manually selected by publishers and displayed for all users. A huge number of news articles generated everyday make it impossible for editors and users to read through all of them, raising the issue of information overload. Online news platforms provide a service of personalized news recommendation by learning from the past reading history of users, e.g., Google (Das et al., 2007; Liu et al., 2010), Yahoo (Trevisiol et al., 2014; Okura et al., 2017), and Bing news (Lu et al., 2015; Wang et al., 2018).

When a new user uses the system (cold-start users) or a new article is just created (cold-start items), there are too few observations for them to train a reliable recommender system. Content-based techniques exploit the content information of news (e.g., words and tags) and hence new articles can be recommended to existing users (Pazzani and Billsus, 2007). Content-based recommendation, however, suffers from the issue of data sparsity since there is no reading history for them to be used to build a profile (Park and Chu, 2009).

Transfer learning is a common technique for alleviating the issues of data sparsity (Pan et al., 2010; Cantador et al., 2015; Liu et al., 2018). A user may have access to many websites such as Twitter.com and Youtube.com (Roy et al., 2012; Huang and Lin, 2016), and consume different categories of products such as movies and books (Li et al., 2009). In this case, transfer learning approaches can recommend articles to a new user in the target domain by exploiting knowledge from the relevant source domains for this new user.

A technical challenge for transfer learning approaches is that user interests are quite different across domains (corpora). For example, users do not use Twitter for the same purpose. A user may follow up on news about "Donald Trump" because she supports republican party (in the political news domain), while she may follow up account @*taylorswift13* ("Taylor Swift") because she loves music (in the entertainment news domain). Another challenge is that the word distribution and feature space are different across domains. For example, vocabularies are different for describing political news and entertainment news. An illustration is depicted in Figure 1. As a result, the user profile computed from her news history is heterogeneous across domains.

Several strategies have been proposed for heterogeneous transfer learning (Yang et al., 2009). The

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 734–744
April 19 - 23, 2021.

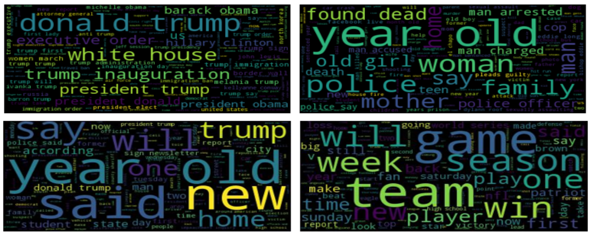


Figure 1: Word clouds of two news corpora. Top: Cheetah Mobile data. Bottom: MIND data. Left and right parts represent different domains (categories).

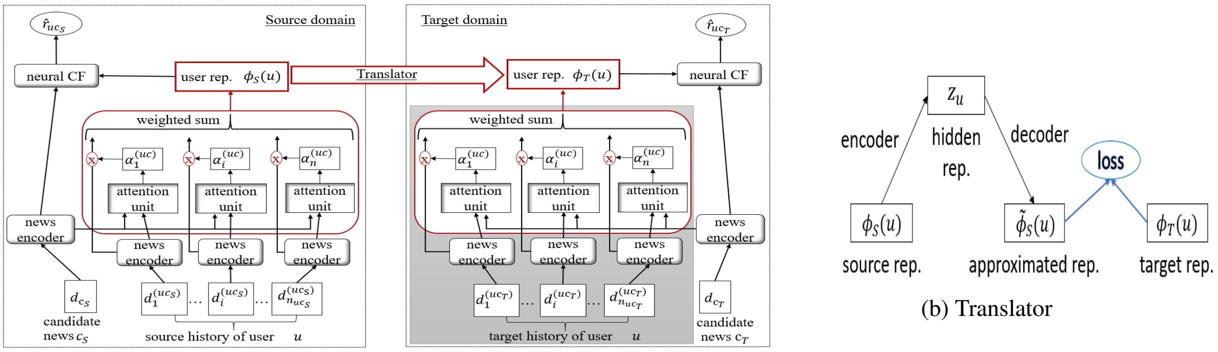
transferable contextual bandit (TCB) (Liu et al., 2018) learns a translation matrix to translate target feature examples to the source feature space. This linear mapping strategy is also used in collaborative cross networks (CoNet) (Hu et al., 2018) and deep dual transfer cross domain recommendation (DDTCDR) (Li and Tuzhilin, 2020). To capture complex relations between source and target domains, some nonlinear mapping strategy is considered in the embedding and mapping cross-domain recommendation (EMCDR) (Man et al., 2017) which learns a supervised regression between source and target factors using a multilayer perceptron (MLP). Since aligned examples between source and target domains are limited, they may face the overfitting issues.

To tackle challenges of heterogeneous user interests and limited aligned data between domains, we propose a novel transfer learning model (TrNews) for cross-corpora news recommendation. TrNews builds a bridge between two base networks (one for each corpus, see Section 3.1.1) through the proposed translator-based transfer strategy. The translator in TrNews captures the relations between source and target domains by learning a nonlinear mapping between them (Section 3.2). The heterogeneity is alleviated by translating user interests across corpora. TrNews uses the translator to transfer knowledge between source and target networks. TrNews alleviates the limited data in a way of alternating training (Section 3.3). The learned translator is used to infer the representations of unseen users in the future (Section 3.4). By "translating" the source representation of a user to the target domain, TrNews offers an easy solution to create unseen users' target representations. TrNews outperforms the state-of-the-art recommendation methods on four real-world datasets in terms of four metrics (Section 4.2), while having an explanation advantage by allowing the visualization of the importance of each news article in the history to the future news (Section 4.4).

2 Related Work

Content recommendation Content-based recommendation exploits the content information about items (e.g., news title and article body (Yan et al., 2012; Xiao et al., 2019; Ma et al., 2019; Wu et al., 2020; Hu et al., 2020), tag, vlog (Gao et al., 2010)), builds a profile for each user, and then matches users to items (Lops et al., 2011; Yu et al., 2016; Wu et al., 2019b). It is effective for items with content or auxiliary information but suffers from the issues of data sparsity for users. DCT (Barjasteh et al., 2015) constructs a user-user similarity matrix from user demographic features including gender, age, occupation, and location (Park and Chu, 2009). NT-MF (Huang and Lin, 2016) constructs a user-user similarity matrix from Twitter texts. BrowseGraph (Trevisiol et al., 2014) addresses the freshly news recommendation by constructing a graph using URL links between web pages. NAC (Rafailidis and Crestani, 2019) transfers from multiple source domains through the attention mechanism. PdMS (Felício et al., 2017) assumes that there are many recommender models available to select items for a user, and introduces a multi-armed bandit for model selection. LLAE (Li et al., 2019a) needs a social network as side information for cold-start users. Different from the aforementioned works, we aim to recommending news to unseen users by transferring knowledge from a source domain to a target domain.

Transfer learning Transfer learning aims at improving the performance of a target domain by exploiting knowledge from source domains (Pan and Yang, 2009). A special setting is domain adaptation where a source domain provides labeled training examples while the target domain provides instances on which the model is meant to be deployed (Glorot et al., 2011; Li et al., 2019b). The coordinate system transfer (CST) (Pan et al., 2010) firstly learns the principle coordinate of users in the source domain, and then transfers it to the target domain in the way of warm-start initialization. This is equivalent to an identity mapping from users' source representations to their corresponding target representations. TCB (Liu et al., 2018) learns a linear mapping to translate target feature examples to the source feature space because there are many labelled data in the source domain. This linear s-



(a) TrNews

(b) Translator

Figure 2: Left: Architecture of TrNews. There is a base network for each of the two domains. The shaded area in the target network is empty for unseen users. The translator enables knowledge transfer between source and target networks. Right: The translator.

trategy is also used in CoNet (Hu et al., 2018) and DDTCDR (Li and Tuzhilin, 2020) which transforms the source representations to the target domain by a translation matrix. Nonlinear mapping strategy (Man et al., 2017; Zhu et al., 2018; Fu et al., 2019) is to learn a supervised mapping function between source and target latent factors by using neural networks. SSCDR (Kang et al., 2019) extends them to the semi-supervised mapping setting. Our translator is general to accommodate these identity, linear, and nonlinear transfer-learning strategies.

3 TrNews

3.1 Architecture

The architecture of TrNews is shown in Figure 2a, which has three parts. There are a source network for the source domain S and a target network for the target domain T, respectively. The source and target networks are both an instantiation of the base network (Section 3.1.1). The translator enables knowledge transfer between the two networks (Section 3.2). We give an overview of TrNews before introducing the base network and the translator.

Target network The information flow goes from the input, i.e., (user u, candidate news c_T) to the output, i.e., the preference score $\hat{r}_{uc_T}$, through the following three steps. First, the news encoder ψ_T computes the news representation from its content. The candidate news representation is $\psi_T(c_T) = \psi_T(d_{c_T})$ where d_{c_T} is c_T's content. The representations of historical news articles $[i]_{i=1}^{n_{uc_T}}$ of the user are $[\psi_T(d_i^{(uc_T)})]_{i=1}^{n_{uc_T}}$ where $d_i^{(uc_T)}$ is i's content and n_{uc_T} is size of the history. Second, the user encoder ϕ_T computes the user representation from her news history by: $\phi_T(u) = \phi_T\left([\psi_T(d_i^{(uc_T)})]_{i=1}^{n_{uc_T}}\right)$. Third, the neural collaborative filtering (CF) module f_T computes the preference score by: $\hat{r}_{uc_T} = f_T([\phi_T(u), \psi_T(c_T)])$. We can denote the target network by a tuple (ψ_T, ϕ_T, f_T).

Source network Similarly to the three-step computing process in target network, we compute preference score $\hat{r}_{uc_S}$ from input (u, c_S) by: $\hat{r}_{uc_S} = f_S([\phi_S(u), \psi_S(c_S)])$ with tuple (ψ_S, ϕ_S, f_S).

Translator The translator $\mathcal{F}$ learns a mapping from the user's source representation to her target representation by $\mathcal{F} : \phi_S(u) \to \phi_T(u)$.

3.1.1 Base network

There is a base network for each of the two domains. It is an attentional network which has three modules (ψ, ϕ, f): the news encoder ψ to learn news representations, the user encoder ϕ to learn user representations, and a neural collaborative filtering module f to learn user preferences from reading behaviors.

News encoder The news encoder module is to learn news representation from its content. The news encoder takes a news article c's word sequence $d_c = [w_j]_{j=1}^{n_c}$ (n_c is length of c) as the input, and outputs its representation $\psi(c) \triangleq \psi(d_c) \in \mathbb{R}^D$ where D is the dimensionality. We compute the average of c's word embeddings by: $\psi(d_c) = \frac{1}{|d_c|} \sum_{w \in d_c} e_w$, where e_w is the embedding of w.

User encoder The user encoder module is to learn the user representation from their reading history. The user encoder takes a user's reading history $[\psi(d_i^{(u)})]_{i=1}^{n_u}$ (n_u is length of u's history) as input, and outputs her representation $\phi(u) \triangleq \phi([\psi(d_i^{(u)})]_{i=1}^{n_u}) \in \mathbb{R}^D$ where D is dimensionality.

In detail, given a pair of user and candidate news (u, c), we get the user representation $\phi(u|c)$ as the weighted sum of her historical news articles' representations: $\phi(u|c) = \sum_{i=1}^{n_{uc}} \alpha_i^{(uc)} \psi(d_i^{(u)})$. The weights $\alpha_i^{(uc)}$'s are computed via attention units by: $\alpha_i^{(uc)} = a([\psi(d_i^{(u)}), \psi(d_c)])$ where a is the attention function with parameters to be learned. We use an MLP to compute it. For a specific candidate news c, we limit the history news to only those articles that are read before it. For notational simplicity, we do not explicitly specify the candidate news when referring to a user representation, i.e., $\phi(u)$ for short of $\phi(u|c)$.

Neural CF The neural collaborative filtering module is to learn preferences from user-news interactions. The module takes concatenated representations of user and news $[\phi(u), \psi(c)]$ as input, and outputs preference score $\hat{r}_{uc} = f([\phi(u), \psi(c)])$ where f is an MLP.

3.2 Translator

The target network suffers from the data sparsity issue of users who have no reading history. In this section, we propose a transfer learning component (i.e., the translator) to enable knowledge transfer for cross-domain news recommendation. The challenge is that user interests and word distributions are different across domains. For example, we compute the word clouds for two news corpora as shown in Figure 1. We can see that their word distributions are quite different and vocabularies are also different. Hence, user representations computed from their news history are heterogeneous across domains.

We build a translator, $\mathcal{F} : \phi_S(u) \to \phi_T(u)$, to learn a mapping from a user's source representation to her target representation as shown in Figure 2b. This translator captures the relationship and heterogeneity across domains. The translator learns to approximate the target representation from the source representation.

The translator takes a user's source representation $\phi_S(u)$ as the input, and maps it to a hidden representation z_u via an encoder parameterized by θ, and then gets a approximated representation $\tilde{\phi}_S(u)$ from it via a decoder parameterized by θ'. The parameters $\Theta_{\mathcal{F}} = \{\theta, \theta'\}$ of the translator are optimized to minimize the approximation error:

$$\mathcal{L}_{\mathcal{F}} = \frac{1}{|\mathcal{U}_0|} \sum_{u \in \mathcal{U}_0} ||H\tilde{\phi}_S(u) - \phi_T(u)||_2^2, \quad (1)$$

where $\mathcal{U}_0 = \mathcal{U}_S \cap \mathcal{U}_T$, and $\mathcal{U}_S$ and $\mathcal{U}_T$ are the user sets of source and target domains, respectively. H is to match the dimensions of source and target representations.

Note that, we do not minimize the approximation error between $\phi_S(u)$ and $\tilde{\phi}_S(u)$ as with the standard autoencoder because our goal is to learn a mapping from a user's source representation to her corresponding target representation. After training, the learned mapping function is then used for inferring representations of unseen users in the target domain (the inference process will be described later in Section 3.4). It fulfills knowledge transfer from the source to the target domain via a supervised learning process.

Extensions The translator can be generalized to multiple, say k, source domains. We learn k translators using the aligned examples from each of the source domain to the target domain and then we average (or concatenate) the k mapped representations as the final representation for the user. Another extension is to introduce denoising or stacking techniques into the translator framework, not just the MLP structure in (Man et al., 2017).

3.3 Model learning

We learn TrNews in two stages. First, we train the source network using source training examples D_S and train the target network using target training examples D_T, respectively. Second, we train the translator by pairs of user representations computed on-the-fly from source and target networks. We introduce these two stages in detail.

First, TrNews optimizes the parameters associated with target network $\Theta_T = \{\theta_{\phi_T}, \theta_{\psi_T}, \theta_{f_T}\}$ and source network $\Theta_S = \{\theta_{\phi_S}, \theta_{\psi_S}, \theta_{f_S}\}$ by minimizing the joint cross-entropy loss:

$$\begin{aligned} \mathcal{L} = &-\textstyle\sum_{D_T}(r_{uc_T} \log \hat{r}_{uc_T} + (1 - r_{uc_T}) \log(1 - \hat{r}_{uc_T})) \\ &-\textstyle\sum_{D_S}(r_{uc_S} \log \hat{r}_{uc_S} + (1 - r_{uc_S}) \log(1 - \hat{r}_{uc_S})), \quad (2) \end{aligned}$$

where the two terms on the right-hand side are to optimize losses over user-news examples in the target and source domains, respectively. They are related by the word embedding matrix for the union of words of the two domains. We generate D_T and D_S as follows and take the target domain as an example since the procedure is the same for the source domain. Suppose we have a whole news reading history for a user u, say $[d_1, d_2, ..., d_{n_u}]$. Then we generate the positive training examples by sliding over the history sequence: $D_T^+ = \{([d_i]_{i=1}^{c-1}, d_c) : c = 2, ..., n_u\}$. We adopt the random negative sampling technique (Pan et al., 2008) to generate the corresponding negative training examples $D_T^- = \{([d_i]_{i=1}^{c-1}, d'_c) : d'_c \notin [d_1, d_2, ..., d_{n_u}]\}$, that is, we randomly sample a news article from the corpus as a negative sample which is not in this user's reading history.

Second, TrNews optimizes the parameters associated with the translator $\Theta_{\mathcal{F}} = \{\theta, \theta'\}$ by Eq. (1). Since the aligned data is limited, we increase the training pairs by generating them on-the-fly during the training of the two networks, i.e., in an alternating way. The model learning is summarized in Algorithm 1.

Algorithm 1: Training of TrNews.

```
Input: $D_T, D_S, \mathcal{U}_0$
Output: Source & target networks, translator
for $iter = 1, 2, ..., 50$ do
    1. Train target and source networks with
       mini batch using $D_T, D_S$ respectively;
    2. for $u \in \mathcal{U}_0$ do
           (a) Generate source representations
               $\phi(u_S)$ using source network;
           (b) Generate target representations
               $\phi(u_T)$ using target network;
           (c) Train the translator using pairs
               $(\phi(u_S), \phi(u_T))$ with mini batch;
       end
    3. Compute metrics on the validation set;
       if No improvement for 10 iters then
           Early stopping;
       end
end
```

3.4 Inference for unseen users

For a new user in the target domain (not seen in the training set $\mathcal{U}_T^{train}$), we do not have any previous history to rely on in learning a user representation for her. That is, the shaded area of the target network in Figure 2a is empty for unseen users.

TrNews estimates a new user u^*'s target representation by mapping from her source representation using the learned translator $\mathcal{F}$ by:

$$\phi_T(u^*) := \mathcal{F}(\phi_S(u^*)), \ \forall u^* \in \mathcal{U}_S \wedge u^* \notin \mathcal{U}_T^{train}, \quad (3)$$

where we compute $\phi_S(u^*)$ using u's latest reading history in the source domain. Then we can predict the user preference for candidate news c^* by:

$$\hat{r}_{u^*c^*} = f_T([\phi_T(u^*), \psi_T(c^*)]). \quad (4)$$

4 Experiment

We evaluate the performance of TrNews (Section 4.2) and the effectiveness of the translator (Section 4.3) in this section.

4.1 Datasets and experimental setup

Datasets We evaluate on two real-world datasets. The first $\boldsymbol{NY}, \boldsymbol{FL}, \boldsymbol{TX}, \& \boldsymbol{CA}$ are four subdatasets extracted from a large dataset provided by an internet company Cheetah Mobile (Liu et al., 2018; Hu et al., 2019). The information contains news reading logs of users in a large geographical area collected in January of 2017, ranging from New York (NY), Florida (FL), Texas (TX), to California (CA) based on the division of user geolocation. They are treated as four rather than a single because the user set is not overlapped among them. The top two categories (political and daily) of news are used as the cross corpora. The mean length of news articles is around 12 words while the max length is around 50 words. The mean length of user history is around 45 articles while the max length is around 900 articles. The second $\boldsymbol{MIND}$ is a benchmark dataset released by Microsoft for news recommendation (Wu et al., 2020). We use the MIND-small version to investigate the knowledge transfer when news reading examples are not so large and it is publicly available https://msnews.github.io/. The title and abstract of news are used as the content. The clicked historical news articles are the positive examples for user. The top two categories (news and sports) of news are used as the cross corpora. The word clouds of the two datasets are shown in Figure 1 and the statistics are summarized in Table 1. The mean length of news articles is around 40 words while the max length is around 123 words. Besides, the mean length of user history is around 13 articles while the max length is around 246 articles.

Evaluation protocol We randomly split the whole user set into two parts, training and test sets where the ratio is 9:1. Given a user in the test set, for each news in her history, we follow the strategy in (He et al., 2017) to randomly sample a number of negative news, say 99, which are not in her reading history and then evaluate how well the recommender can rank this positive news against these negative ones. For each user in the training set, we reserve her last reading news as the valid set. We follow the typical metrics to evaluate top-K news recommendation (Peng et al., 2016; Okura et al.,

Data	#user	Target domain			Source domain		
		#news	#reading	#word	#news	#reading	#word
NY	14,419	33,314	158,516	368,000	23,241	139,344	273,894
FL	15,925	33,801	178,307	376,695	25,644	168,081	340,797
TX	20,786	38,395	218,376	421,586	29,797	221,344	343,706
CA	26,981	44,143	281,035	481,959	32,857	258,890	375,612
MIND	25,580	9,372	211,304	461,984	8,577	120,409	346,988

Table 1: Statistics of the datasets.

NY	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
POP	52.96	67.66	40.34*	45.10	39.89*	77.92
LR	53.24	74.00	36.15	42.86	34.95	91.64
TANR	52.53	71.63	37.24	43.37	36.50	91.35
DeepFM	52.02	73.71	39.17	45.38	39.56	91.79
DIN	57.10*	75.66*	40.23	46.13*	38.65	92.29*
TrNews	**82.60**	**95.15**	**60.78**	**64.83**	**55.70**	**97.28**
FL	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
POP	52.45	66.14	39.72*	44.15	39.15*	79.33
LR	54.26*	73.90*	37.15	43.56	35.89	91.79
TANR	49.98	69.46	36.08	42.37	35.95	90.88
DeepFM	52.36	73.02	36.05	42.74	36.29	91.64
DIN	53.98	73.33	37.96	44.18*	36.96	91.86*
TrNews	**81.83**	**94.45**	**62.53**	**66.63**	**58.39**	**97.41**
TX	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
POP	54.21	67.87	40.62*	45.03*	39.64*	81.31
LR	55.72*	73.80*	39.24	44.97	37.78	91.74*
TANR	49.87	68.75	35.82	41.89	35.59	90.56
DeepFM	52.19	71.95	35.40	41.92	35.65	91.17
DIN	53.72	72.70	38.47	44.59	37.62	91.53
TrNews	**81.50**	**94.67**	**61.76**	**66.11**	**57.49**	**97.21**
CA	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
POP	58.32*	71.19	44.71*	48.86*	43.44*	83.38
LR	58.82	75.67*	42.16	47.65	40.44	92.37*
TANR	49.87	68.75	35.81	41.88	35.58	90.56
DeepFM	55.58	74.73	38.82	45.16	38.21	92.25
DIN	55.31	73.70	40.14	46.09	39.20	92.03
TrNews	**81.54**	**94.72**	**61.99**	**66.25**	**57.70**	**97.22**
MIND	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
POP	84.18	92.80	69.61	72.43	66.33	95.13
LR	92.69*	96.66*	85.81	87.11*	84.33*	97.92*
TANR	89.94	95.34	89.94*	83.38	79.84	97.86
DeepFM	89.16	94.78	79.36	81.19	77.12	97.63
DIN	89.28	94.88	80.16	82.03	78.22	97.63
TrNews	**97.36**	**99.02**	**94.16**	**94.74**	**93.45**	**99.47**

Table 2: Comparison of different recommenders.

2017; An et al., 2019) which are hit ratio (HR), normalized discounted cumulative gain (NDCG), mean reciprocal rank (MRR), and the area under the ROC curve (AUC). We report the results at cut-off $K \in \{5, 10\}$.

Implementation We use TensorFlow. The optimizer is Adam (Kingma and Ba, 2015) with learning rate 0.001. The size of mini batch is 256. The neural CF module has two hidden layers with size 80 and 40 respectively. The size of word embedding is 128. The translator has one hidden layer on the smaller datasets and two on the larger ones. The history is the latest 10 news articles.

4.2 Comparing different recommenders

In this section, we show the recommendation results by comparing TrNews with different state-of-the-art methods.

Baselines We compare with following recommendation methods which are trained on the merged source and target datasets by aligning with shared users: **POP** (Park and Chu, 2009) recommends the most popular news. **LR** (McMahan et al., 2013) is widely used in ads and recommendation. The input is the concatenation of candidate news and user's representations. **DeepFM** (Guo et al., 2017) is a deep neural network for ads and recommendation based on the wide & deep structure. We use second-order feature interactions of reading history and candidate news, and the input of deep component is the same as LR. **DIN** (Zhou et al., 2018) is a deep interest network for ads and recommendation based on the attention mechanism. We use the news content for news representations. **TANR** (Wu et al., 2019a) is a state-of-the-art deep news recommendation model using an attention network to learn the user representation. We adopt the news encoder and negative sampling the same with TrNews.

Results We have observations from results of different recommendation methods as shown in Table 2. Firstly, considering that breaking and headline news articles are usually read by every user, the POP method gets competitive performance in terms of NDCG and MRR since it ranks the popular news higher than the other news. Secondly, the neural methods are generally better than the traditional, shallow LR method in terms of NDCG, MRR, and AUC on the four subdatasets. It may be that neural networks can learn nonlinear, complex relations between the user and the candidate news to capture user interests and news semantics. Considering that the neural representations of user and candidate news are fed as the input of LR, it gets competitive performance on MIND data. Finally, the proposed TrNews model achieves the best performance with a large margin improvement over all other baselines in terms of HR, NDCG, and MRR and also with an improvement in terms of AUC. It validates the necessity of accounting for the heterogeneity of user interests and word distributions across domains. This also shows that the base network is an effective architecture for news recommendation and the translator is effective to enable the knowledge transfer from the source domain to the target domain. In more detail, it is inferior by training a global model from the mixed source and target examples and then using this global model to predict user preferences on the target domain, as baselines do. Instead, it is good by training source

Approach	Transfer strategy	Formulation
CST (Pan et al., 2010)	Identity mapping	$\phi_T(u) = \phi_S(u)$
TCB (Liu et al., 2018)	Linear mapping	$\phi_T(u) = H\phi_S(u)$
DDTCDR (Li and Tuzhilin, 2020)		H is orthogonal
EMCDR (Man et al., 2017)	Nonlinear mapping	$\phi_T(u) = \text{MLP}(\phi_S(u))$

Table 3: Different transfer learning strategies.

NY	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
CST	81.04	94.37	59.04	63.56	54.19	96.94
TCB	82.18	94.92*	60.36*	64.46*	55.23*	97.28*
DDTCDR	82.27	94.90	59.82	63.90	54.51	97.25
EMCDR	82.44*	94.87	60.35	64.33	55.06	97.24
TrNews	**82.60**	**95.15**	**60.78**	**64.83**	**55.70**	**97.28**

FL	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
CST	79.29	93.91	59.03	63.60	54.74	97.07
TCB	81.51	**94.83**	62.06	66.33*	57.90*	97.40*
DDTCDR	81.39	94.63*	61.76	66.12	57.68	97.37
EMCDR	81.52*	94.47	62.14*	66.23	57.87	97.37
TrNews	**81.83**	94.45	**62.53**	**66.63**	**58.39**	**97.41**

TX	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
CST	78.74	94.20	58.53	63.48	54.56	96.92
TCB	80.68	94.12	61.06	65.38	56.97	97.10
DDTCDR	81.08	94.57	61.02	65.50	56.87	97.10
EMCDR	81.34*	**94.72**	**61.78**	66.11*	**57.59**	97.16*
TrNews	**81.50**	94.67*	61.76*	**66.11**	57.49*	**97.21**

CA	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
CST	79.92	93.71*	60.19	64.63	55.97	97.12
TCB	80.90*	93.71*	**62.32**	**66.45**	**58.35**	**97.36**
DDTCDR	80.22	93.47	61.42	65.72	57.44	97.25
EMCDR	80.53	93.33	62.04*	66.18	58.11*	97.30*
TrNews	**81.54**	**94.72**	61.99	66.25*	57.70	97.22

MIND	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
CST	96.93	98.49	93.83	94.34	93.09	99.32
TCB	97.41*	98.94	94.21*	94.72	93.43	99.41
DDTCDR	97.38	98.99	**94.25**	**94.78**	**93.49**	99.46*
EMCDR	**97.42**	99.01*	94.16	94.68	93.35	99.45
TrNews	97.36	**99.02**	94.16	94.74*	93.45*	**99.47**

Table 4: Comparison of different transfer strategies.

and target networks on the source and target domains, respectively, and then learning a mapping between them, as TrNews does.

4.3 Comparing different transfer strategies

In this section, we demonstrate the effectiveness of the translator-based transfer-learning strategy.

Baselines We replace the translator of TrNews with the transfer-learning strategies of baseline methods as summarized in Table 3. All baselines are state-of-the-art recommenders and capable of recommending news to cold-start users. Note that, the compared transfer-learning methods are upgraded from their original versions. We strengthen them by using the neural attention architecture as the base component. In their original versions, CST and TCB use matrix factorization (MF) while DDTCDR and EMCDR use multilayer perceptron. The neural attention architecture has shown superior performance over MF and MLP in the literature (Zhou et al., 2018; Wu et al., 2019a). As a result, we believe that the improvement will be larger if we compare with their original versions but this is obviously unfair.

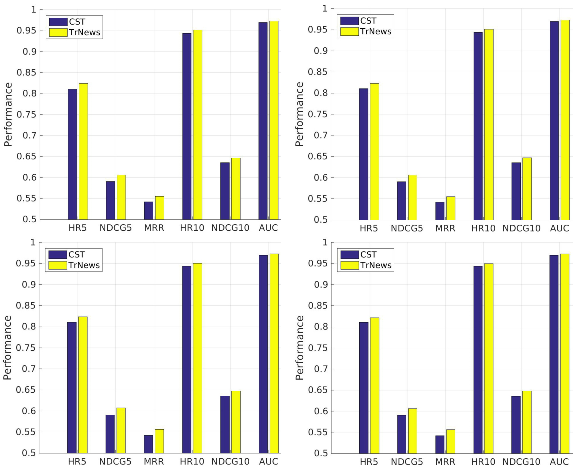


Figure 3: Impact of percentage (90%, 70%, 50%, 30%.) of shared users used to train the translator.

Results We have observations from results of different transfer learning strategies as shown in Table 4. Firstly, the transfer strategy of identity mapping (CST) is generally inferior to the linear (TCB and DDTCDR) and nonlinear (EMCDR and TrNews) strategies. CST directly transfers the source knowledge to the target domain without adaptation and hence suffers from the heterogeneity of user interests and word distributions across domains. Secondly, the nonlinear transfer strategy of EMCDR is inferior to the linear strategy of TCB in terms of MRR and AUC on the two smaller NY and FL datasets. This is probably because EMCDR increases the model complexity by introducing two large fully-connected layers in its MLP component. In contrast, our translator is based on the small-waist autoencoder-like architecture and hence can resist overfitting to some extent. Finally, our translator achieves the best performance in terms of NDCG, MRR and AUC on the two smaller NY and FL datasets, and achieves competitive performance on the two larger TX and CA datasets, and achieves the best performance in terms of HR and AUC on the MIND dataset, comparing with other four transfer methods. These results validate that our translator is a general and effective transfer-learning strategy to capture the diverse user interests accurately during the knowledge transfer for the unseen users in cross-domain news recommendation.

4.4 Analysis

Benefit of knowledge transfer We vary the percentage of shared users used to train the translator (see Eq. (1)) with $\{90\%, 70\%, 50\%, 30\%\}$. We

Sharing?	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
No	81.31	94.69	59.43	63.72	54.37	97.16
Yes	**82.60**	**95.15**	**60.78**	**64.83**	**55.70**	**97.28**

Table 5: Impact of sharing word embeddings between source and target domains.

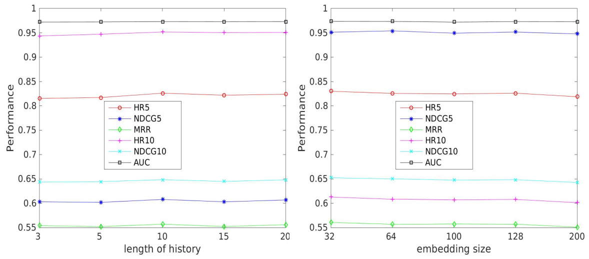
Strategy	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
Sepa.	82.36	94.86	60.62	64.65	55.51	97.28
Alter.	**82.60**	**95.15**	**60.78**	**64.83**	**55.70**	**97.28**

Table 6: Training TrNews with alternating (Alter.) vs separating (Sepa.) strategies.

compare with a naive transfer strategy of CST, i.e., the way of direct transfer without adaptation. The results are shown in Figure 3 on the New York dataset. We can see that it is beneficial to learn an adaptive mapping during the knowledge transfer even when limited aligned examples are available to train the translator. TrNews improves relative 0.82%, 0.77%, 0.67%, 0.64% in terms of HR@10 performance over CST by varying among $\{90\%, 70\%, 50\%, 30\%\}$ respectively. So we think that the more aligned examples the translator has, the more benefits it achieves.

Impact of sharing word embeddings We investigate the benefits of sharing word embeddings between source and target domains. There is a word embedding matrix for each of the domains and we share the columns if the corresponding words occur in both domains. Take the New York dataset as an example, the size of the intersection of their word vocabularies is 11,291 while the union is 50,263. From the results in Table 5 we can see that it is beneficial to share the word embeddings even when only 22.5% words are intersected between them.

Impact of alternating training We adopt an alternating training strategy between training the two (source & target) networks and training the translator in our experiments. In this section, we compare this alternating strategy with the separating strategy which firstly trains the two networks and then trains the translator after completing the training of the two networks. That is, the training pairs of user representations for the translator are not generated on-the-fly during the training of source and target networks but generated only once after finishing their training. From the results in Table 6, we see that the alternating strategy works slightly better. This is probably because the aligned data between domains is limited and the alternating strategy increases the size of training pairs.



(a) History length.

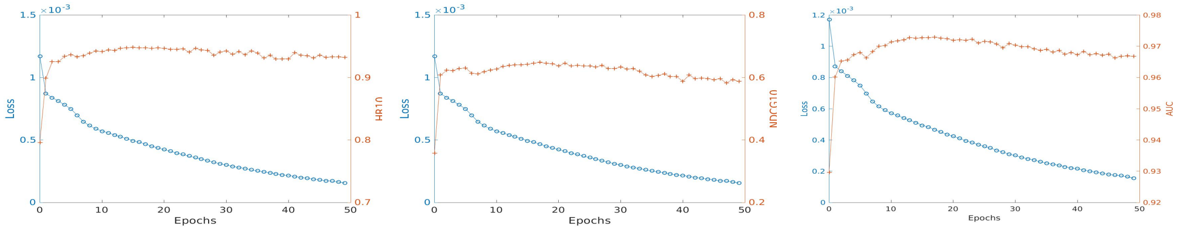
(b) Embedding size.

Figure 4: Impact of the history length (left) and embedding size (right).

Impact of two-stage learning We adopt a two-stage model learning between training the two (source & target) networks and training the translator in our experiments. In this section, we compare this two-stage learning with an end-to-end learning which jointly trains the two networks and the translator. That is, the parameters of the translator depend on the word embedding matrix and on parameters of the user encoder. From the results in Table 7, we see that the two-stage learning works slightly better. This is probably because the aligned data between domains is too limited to reliably update the parameters which do not belong to the parameters of the translator.

Impact of the length of the history Since we generate the training examples by sliding over the whole reading history for each user, the length of reading history is a key parameter to influence the performance of TrNews. We investigate how the length of the history affects the performance by varying it with $\{3, 5, 10, 15, 20\}$. The results on the New York dataset are shown in Figure 4a. We can observe that increasing the size of the sliding window is sometimes harmful to the performance, and TrNews achieves good results for length 10. This is probably because of the characteristics of news freshness and of the dynamics of user interests. That is, the latest history matters more in general. Also, increasing the length of the input makes the training time increase rapidly, which are 58, 83, 143, 174, and 215 seconds when varying with $\{3, 5, 10, 15, 20\}$ respectively.

Impact of the embedding size In this section, we evaluate how different choices of some key hyperparameter affect the performance of TrNews. Except for the parameter being analyzed, all other parameters remain the same. Since we compute the news and user representations using the content of words, the size of word embedding is a key parameter to influence representations of word-



(a) HR@10. (b) NDCG@10. (c) AUC.

Figure 5: Performance (right Y-axis in red cross) and loss (left Y-axis in blue circle) varying with training iterations.

s, uses, and news articles, and hence the performance of TrNews. We investigate how embedding size affects the performance by varying it with $\{32, 64, 100, 128, 200\}$. The results on the New York dataset are shown in Figure 4b. We can observe that increasing the embedding size is generally not harmful to the performance until 200, and TrNews achieves good results for embedding size 128. Changing it to 200 harms the performance a little bit since the model complexity also increases.

Optimization performance and loss We show the optimization performance and loss over iterations on the New York dataset in Figure 5. We can see that with more iterations, the training losses gradually decrease and the recommendation performance is improved accordingly. The most effective updates are occurred in the first 15 iterations, and performance gradually improves until 30 iterations. With more iterations, TrNews is relatively stable. For the training time, TrNews spends 143 seconds per iteration. As a reference, it is 134s for DIN and 139s for TCB, which indicates that the training cost of TrNews is efficient by comparing with baselines. Furthermore, the test time is 150s. The experimental environment is Tensorflow 1.5.0 with Python 3.6 conducted on Linux CentOS 7 where The GPU is Nvidia TITAN Xp based on CUDA V7.0.27.

Examining user profiles One advantage of TrNews is that it can explain which article in a user's history matters the most for a candidate article by using attention weights in the user encoder module. Table 8 shows an example of interactions between some user's history articles No. 0-9 and a candidate article No. 10, i.e., the user reads the candidate article after read these ten historical articles. We can see that the latest three articles matter the most since the user interests may remain the same during a short period. The oldest two articles, however, also have some impact on the candidate article, reflecting that the user interests may mix with a long-term characteristic. TrNews can capture these subtle short- and long-term user interests.

Training	HR@5	HR@10	NDCG@5	NDCG@10	MRR	AUC
end-to-end	81.85	94.63	60.58	64.74	55.68	97.03
two-stage	**82.60**	**95.15**	**60.78**	**64.83**	**55.70**	**97.28**

Table 7: Training TrNews in two-stage vs end-to-end.

No.	News title	Attn. weight
0	hillary clinton makes a low-key return to washington	0.04
1	the hidden message in obama's 'farewell' speech	0.12*
2	here's why sasha obama skipped the farewell address	0.00
3	donald trump's 'prostitute scandal' was filmed by cameras and recorded with microphones hidden behind the walls	0.00
4	white house official explains sasha obama's absence at father's farewell speech	0.00
5	irish bookie puts odds on trump's administration, inauguration and impeachment	0.00
6	heads are finally beginning to roll at the clinton foundation	0.00
7	donald trump's incoming administration considering white house without press corps	**0.76**
8	donald trump says merkel made 'big mistake' on migrants	0.05
9	controversial clinton global initiative closing its doors for good	0.00
10	army chief gen. bipin rawat talks about equal responsibility for women in the frontlines. we couldn't agree more	N/A

Table 8: Example I: Some articles matter more while some are negligible. (No. 10 is the candidate news)

5 Conclusion

We investigate the cross-domain news recommendation via transfer learning. The experiments on real-word datasets demonstrate the necessity of tackling heterogeneity of user interests and word distributions across domains. Our TrNews model and its translator component are effective to transfer knowledge from the source network to the target network. We also shows that it is beneficial to learn a mapping from the source domain to the target domain even when only a small amount of aligned examples are available. In future works, we will focus on preserving the privacy of the source domain when we transfer its knowledge.

Acknowledgement

We thank Dr. Yu Zhang for insightful discussion. The work is partly supported by the National Key Research and Development Program of China under Grant No.2018AAA0101100 and a collaboration project with DiDi: 19201730.

References

Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *Proceedings of Annual Meeting of Association for Computational Linguistics*, pages 336–345.

Iman Barjasteh, Rana Forsati, Farzan Masrour, Abdol-Hossein Esfahanian, and Hayder Radha. 2015. Cold-start item and user recommendation with decoupled completion and transduction. In *Proceedings of ACM Conference on Recommender Systems*.

I. Cantador, I. Fernández, S. Berkovsky, and P. Cremonesi. 2015. Cross-domain recommender systems. In *Recommender systems handbook*.

Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In *Proceedings of World Wide Web*, pages 271–280.

Crícia Z Felício, Klérisson VR Paixão, Celia AZ Barcelos, and Philippe Preux. 2017. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of Conference on User Modeling, Adaptation and Personalization*, pages 32–40.

Wenjing Fu, Zhaohui Peng, Senzhang Wang, Yang Xu, and Jin Li. 2019. Deeply fusing reviews and contents for cold start users in cross-domain recommendation systems. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 94–101.

Wen Gao, Yonghong Tian, Tiejun Huang, and Qiang Yang. 2010. Vlogging: A survey of videoblogging technology on the web. *ACM Computing Surveys*.

Xavier Glorot, Antoine Bordes, and Y. Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of international conference on machine learning*.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 1725–1731.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of world wide web*.

Guangneng Hu, Yu Zhang, and Qiang Yang. 2018. Conet: Collaborative cross networks for cross-domain recommendation. In *Proceedings of ACM International Conference on Information and Knowledge Management*, pages 667–676.

Guangneng Hu, Yu Zhang, and Qiang Yang. 2019. Transfer meets hybrid: a synthetic approach for cross-domain collaborative filtering with text. In *The World Wide Web Conference*, pages 2822–2829.

Linmei Hu, Siyong Xu, Chen Li, Cheng Yang, Chuan Shi, Nan Duan, Xing Xie, and Ming Zhou. 2020. Graph neural news recommendation with unsupervised preference disentanglement. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4255–4264. Association for Computational Linguistics.

Yu-Yang Huang and Shou-De Lin. 2016. Transferring user interests across websites with unstructured text for cold-start recommendation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 805–814.

SeongKu Kang, Junyoung Hwang, Dongha Lee, and Hwanjo Yu. 2019. Semi-supervised learning for cross-domain recommendation to cold-start users. In *Proceedings of ACM International Conference on Information and Knowledge Management*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *International Joint Conference on Artificial Intelligence*.

Jingjing Li, Mengmeng Jing, Ke Lu, Lei Zhu, Yang Yang, and Zi Huang. 2019a. From zero-shot learning to cold-start recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4189–4196.

Pan Li and Alexander Tuzhilin. 2020. Ddtcdr: Deep dual transfer cross domain recommendation. In *Proceedings of ACM International Conference on Web Search and Data Mining*.

Yitong Li, T. Baldwin, and T. Cohn. 2019b. Semisupervised stochastic multi-domain learning using variational inference. In *Proceedings of Annual Meeting of Association for Computational Linguistics*.

Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. 2018. Transferable contextual bandit for cross-domain recommendation. In *AAAI Conference on Artificial Intelligence*.

Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In *Proceedings of international conference on Intelligent user interfaces*, pages 31–40.

Pasquale Lops, M. De Gemmis, and G. Semeraro. 2011. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*.

Zhongqi Lu, Zhicheng Dou, Jianxun Lian, Xing Xie, and Qiang Yang. 2015. Content-based collaborative filtering for news topic recommendation. In *Twenty-ninth AAAI conference on artificial intelligence*.

Ye Ma, Lu Zong, Yikang Yang, and Jionglong Su. 2019. News2vec: News network embedding with subnode information. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*.

Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: an embedding and mapping approach. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 2464–2470.

Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of ACM SIGKDD international conference on Knowledge discovery and data mining*.

Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *IEEE International Conference on Data Mining*, pages 502–511.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Weike Pan, Evan Wei Xiang, Nathan Nan Liu, and Qiang Yang. 2010. Transfer learning in collaborative filtering for sparsity reduction. In *AAAI conference on artificial intelligence*.

Seung-Taek Park and Wei Chu. 2009. Pairwise preference regression for cold-start recommendation. In *Proceedings of ACM conference on Recommender systems*, pages 21–28.

Michael Pazzani and D. Billsus. 2007. Content-based recommendation systems. In *The adaptive web*.

Hao Peng, Jing Liu, and Chin-Yew Lin. 2016. News citation recommendation with implicit and explicit semantics. In *Proceedings of Annual Meeting of Association for Computational Linguistics*.

Dimitrios Rafailidis and Fabio Crestani. 2019. Neural attentive cross-domain recommendation. In *Proceedings of ACM SIGIR International Conference on Theory of Information Retrieval*, pages 165–172.

Suman Deb Roy, Tao Mei, Wenjun Zeng, and Shipeng Li. 2012. Socialtransfer: cross-domain transfer learning from social streams for media applications. In *Proceedings of ACM international conference on Multimedia*, pages 649–658.

Michele Trevisiol, Luca Maria Aiello, Rossano Schifanella, and Alejandro Jaimes. 2014. Cold-start news recommendation with domain-dependent browse graph. In *Proceedings of ACM Conference on Recommender systems*, pages 81–88.

Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of World Wide Web Conference*, pages 1835–1844.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Yongfeng Huang, and Xing Xie. 2019a. Neural news recommendation with topic-aware news representation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 1154–1159.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019b. Neural news recommendation with heterogeneous user behavior. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606. Association for Computational Linguistics.

Wenyi Xiao, Huan Zhao, Haojie Pan, Yangqiu Song, Vincent W. Zheng, and Qiang Yang. 2019. Beyond personalization: Social content recommendation for creator equality and consumer satisfaction. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Rui Yan, Mirella Lapata, and Xiaoming Li. 2012. Tweet recommendation with graph co-ranking. In *Proceedings of Annual Meeting of Association for Computational Linguistics:*.

Qiang Yang, Yuqiang Chen, Gui-Rong Xue, Wenyuan Dai, and Yong Yu. 2009. Heterogeneous transfer learning for image clustering via the social web. In *Proceedings of Annual Meeting of Association for Computational Linguistics*.

Yang Yu, Xiaojun Wan, and Xinjie Zhou. 2016. User embedding for scholarly microblog recommendation. In *Proceedings of Annual Meeting of Association for Computational Linguistics*.

Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1059–1068.

Feng Zhu, Yan Wang, Chaochao Chen, Guanfeng Liu, Mehmet Orgun, and Jia Wu. 2018. A deep framework for cross-domain and cross-system recommendations. In *International Joint Conference on Artificial Intelligence*.

Dialogue Act-based Breakdown Detection in Negotiation Dialogues

Atsuki Yamaguchi♠[*], **Kosui Iwasa**♣ **and Katsuhide Fujita**♢
♠Research and Development Group, Hitachi, Ltd.
♣Graduate School of Engineering, Tokyo University of Agriculture and Technology
♢Institute of Engineering, Tokyo University of Agriculture and Technology
♠`atsuki.yamaguchi1@gmail.com`
♣`contact@iwasakosui.com`
♢`katfuji@cc.tuat.ac.jp`

Abstract

Thanks to the success of goal-oriented negotiation dialogue systems, studies of negotiation dialogue have gained momentum in terms of both human-human negotiation support and dialogue systems. However, the field suffers from a paucity of available negotiation corpora, which hinders further development and makes it difficult to test new methodologies in novel negotiation settings. Here, we share a human-human negotiation dialogue dataset in a job interview scenario that features increased complexities in terms of the number of possible solutions and a utility function. We test the proposed corpus using a breakdown detection task for human-human negotiation support. We also introduce a dialogue act-based breakdown detection method, focusing on dialogue flow that is applicable to various corpora. Our results show that our proposed method features comparable detection performance to text-based approaches in existing corpora and better results in the proposed dataset.

1 Introduction

Negotiation is an essential task involved in our daily life. In negotiation, people work to maximize their profits by bargaining; however, negotiation sometimes breaks down due to conflicts between people's competing interests. To help them to reach rational agreement, previous studies of multiagent systems have proposed the use of negotiating agents (Lin and Kraus, 2010; Jonker et al., 2017; Baarslag et al., 2013a). Recently, several studies have succeeded in modeling a negotiating agent in natural language that can control both text generation and reasoning in the context of goal-oriented dialogue systems, and such agents have produced better performance than human players in some cases (Lewis et al., 2017; He et al., 2018; Cheng et al., 2019). Further, support for human-human negotiation in natural language has also been tackled, involving negotiation corpora developed for goal-oriented dialogue systems, such as a Nash bargaining solution estimation (Iwasa and Fujita, 2018), real-time negotiation coaching (Zhou et al., 2019), and negotiation breakdown detection (Yamaguchi and Fujita, 2020).

Although they have recently attracted additional attention, there are only few negotiation corpora, as the most recent follow-up studies (Iwasa and Fujita, 2018; Cheng et al., 2019; Zhou et al., 2019; Yamaguchi and Fujita, 2020) have only utilized either the DEALORNODEAL (DN) (Lewis et al., 2017), CRAIGSLISTBARGAIN (CB) (He et al., 2018) datasets or both. Moreover, most existing corpora have simplified negotiation settings; for example, the DN dataset handles the negotiation of item division between humans with 22.5 possible solutions per dialogue and uses a standard linear additive utility function (Keeney and Raiffa, 1993; Raiffa et al., 2002) for scoring. The CB dataset is only concerned with price negotiation on a listed product between two human negotiators. These settings might make it easy for a machine learning (ML) model to reach optimal solution or fulfill its goal. Finally, some existing corpora (Konovalov et al., 2016; Petukhova et al., 2016; Asher et al., 2016) other than the DN and CB datasets have far smaller samples (scenarios), which makes it challenging to use them for goal-oriented dialogue systems or end-to-end human-human negotiation support. All of these factors inhibit further development in the field and its future applicability to real-world problems. Furthermore, no effective breakdown detection method for negotiation dialogues has been proposed. Negotiation features certain unique characteristics relative to other dialogues, such as offering proposals, accepting them,

---

[*] This work was conducted when the first author was a master's student at the University of Sheffield, UK.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 745–757
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

and making counter-offers (Thompson et al., 2010; Traum et al., 2008). If the breakdown detection method can incorporate these characteristics, the quality of breakdown detection will be improved.

This study proposes a new negotiation corpus in a job interview setting with increased complexities relative to a range of solutions and a utility function. We enact a breakdown detection task (Yamaguchi and Fujita, 2020) across three negotiation datasets including a proposed one with a novel dialogue act-based approach that can focus on dialogue flow. This task can support human-human negotiation by alerting negotiators to potential breakdowns, which prevents the loss of time and negotiator utility. We highlight the following contributions:

1. We develop a new English negotiation corpus for a job interview setting, consisting of 2639 crowd-sourced dialogues (Section 3).

2. We propose a novel breakdown detection method that employs dialogue act-based features and a gated recurrent unit (GRU) (Chung et al., 2014)-based model (Section 5).

3. We demonstrate that the proposed method exhibits results that are comparable to models with text-based features in the existing corpora and outperforms them in the proposed corpus, which has a far smaller breakdown ratio (Section 7).

4. We conduct ablation studies and error analyses to examine how our proposed features works on a GRU-based model (Section 7).

2 Related Work

Automated Negotiation in Multiagent Systems Automated negotiation is a field of research, in which computers negotiate with each other and try to seek appropriate agreement without human intervention (Baarslag et al., 2013a). Typical applications include supply chain management (Wang et al., 2009) and smart grids (Ketter et al., 2013). As automated negotiation has gained momentum, the International Automated Negotiating Agents Competition (ANAC) (Baarslag et al., 2015; Jonker et al., 2017) has been being held annually since 2010. This event encourages the development of state-of-the-art negotiating strategies for automated negotiating agents in both agent-agent and human-agent (Mell et al., 2018) negotiations. The major difference between automated negotiation and ours is that the former supports negotiation by letting the agents negotiate instead of humans, whereas the latter seeks to support human-human negotiation in natural language only by providing feedback to negotiators with ML models.

NLP for Human-human Negotiation Support Automated negotiation has gained a great deal attention, but there have been only a few studies conducted on support for human-human negotiation in natural language: Iwasa and Fujita (2018) have proposed a GRU-based model to suggest a draft agreement that maximizes the sum of utilities based on the estimated weights of all items in the DN dataset. Zhou et al. (2019) proposed a dynamic negotiation coaching method in the setting of CB dataset that provides useful recommendations to sellers, resulting in increased profits. Our work is a follow-up study to Yamaguchi and Fujita (2020), who demonstrated that neural-network (NN)-based models trained with text-based features could capture signs of breakdowns in DN and CB datasets. Here, we show that text-based methods cannot detect breakdowns in the proposed corpus relative to our dialogue act-based approach.

Negotiation Dialogue Systems Previous efforts on building negotiation dialogue systems initially focused on modeling strategic aspects (Cuayáhuitl et al., 2015; Keizer et al., 2017; Petukhova et al., 2017), to construct an agent that could outperform human players by controlling a discrete action space. By contrast, Lewis et al. (2017) and He et al. (2018) have recently tried to simultaneously handle both text generation and reasoning by employing end-to-end neural negotiating models; moreover, Cheng et al. (2019) proposed adversarial training to improve the robustness of goal-oriented models. Although our main scope is supporting human-human negotiation, our corpus can also be used for goal-oriented dialogue systems (Lewis et al., 2017; He et al., 2018; Cheng et al., 2019) as its fundamental design is drawn from the DN dataset.

Negotiation Dialogue Datasets Several negotiation dialogue corpora, along with the DN and CB datasets, have been proposed to model strategic dialogue. Konovalov et al. (2016) built a bilateral negotiation corpus between a human and an agent in relation to terms of employment. Petukhova et al. (2016) created a corpus in which each negotiator acts as either a city councilor or a small business owner and debates new anti-smoking regulations.

Issue	Option
Salary	\$20 to \$50 per hour (integer)
Weekly day off	2 days to 5 days (integer)
⋆Position	{Engineer, Designer, Manager, Sales}
⋆Company	{Google, Apple, Facebook, Amazon}
Workplace	{Tokyo, Seoul, Beijing, Sydney}

Table 1: List of issues and options in the JI dataset: ⋆ denotes that there is an interdependent relationship between issues.

Asher et al. (2016) developed a multilateral negotiation dialogue corpus in the *Settlers of Catan* game. Our corpus and that of Konovalov et al. (2016) are similar to each other, in that both handle a job contract scenario. However, three main differences appear between the two: (1) The former handles human-human negotiation, whereas the latter deals with human-agent negotiation. (2) The former considers 11.5 times more possible solutions per dialogue than the latter. (3) The former has 2639 dialogues, and the latter has 105.

Dialogue Breakdown Detection Challenge The recently held Dialogue Breakdown Detection Challenge (DBDC) (Higashinaka et al., 2016; Hori et al., 2019) was intended to improve the coherency of a dialogue system. Given a dialogue history between a human and a system, the task is to evaluate whether a certain system response is valid. By contrast, our study focuses on predicting negotiation outcomes based on human-human negotiation to avoid negotiation breakdowns; that is, our task is different from the DBDC.

3 Job Interview Negotiation Dataset

3.1 Overview

The JOBINTERVIEW (JI) dataset is an instance of multi-issue multi-option negotiation, which includes the preferences of the negotiators, a dialogue history, proposed offers, and a settled agreement in a job interview setting. The negotiators conduct a conversation in English in the roles of recruiter or applicant and negotiate regarding the issues listed in Table 1 to maximize their scores. A dialogue sample from the JI dataset is shown in Table 2[1].

Utterance	Dialogue Act
Recruiter - Hello	`<greet>`
Worker - Hi	`<greet>`
Recruiter - I have a position open as an engineer at google. Are you interested?	`<inquire>`
Worker - Yes.	`<agree>`
Recruiter - The position is in tokyo. It pays \$35/hr and it is 4 days a week. Is this acceptable?	`<propose><inquire>`
Worker - Salary is too low.	`<disagree>`
Recruiter - OK. We could bump it to \$40/hr. Is this OK?	`<agree><propose>` `<inquire>`
Worker - How about I work in Beijing?	`<inquire>`
Recruiter - Beijing is open also.	`<inform>`
Worker - 5 days/wk.	`<propose>`
Recruiter - I cannot do 5 days a week.	`<disagree><propose>`
Worker - 4 days/wk and \$47/hr?.	`<propose><inquire>`
Recruiter - OK.	`<agree>`

Table 2: Sample dialogue between two negotiators in the JI dataset with extracted dialogue acts.

3.2 Mathematical Design

To make the negotiation competitive, we define each negotiator's preferences, and a scoring function, as in Lewis et al. (2017). In addition, we consider the interdependency (Kardan and Janzadeh, 2008; Alam et al., 2013) between a pair of issues such that the negotiators cannot easily reach an optimal agreement (Ito et al., 2006), leading them to seek a compromise solution through dialogue.

Preferences The importance of each issue and option, and bias assignment in representing interdependency between specific issues are defined as follows. Two negotiators $\boldsymbol{A} = \{a_1, a_2\}$ participate in a negotiation over the set of independent issues $\boldsymbol{I}$ and of issues $\boldsymbol{J}$ with an interdependent relationship. An issue $i \in \boldsymbol{I}$ is assigned a weight (importance) $w_i^{a_k} \in [0.1, 0.6], \sum_{i \in \boldsymbol{I}} w_i^{a_k} = 1$ per negotiator a_k with $k = 1$ or 2. An option for i, $o^i \in \boldsymbol{O}^i$, is assigned a weight $w_{o^i}^{a_k} \in [0, 1]$. While an issue included in a set of specific issues with an interdependent relationship $(j_{\text{from}}, j_{\text{to}}) \in \boldsymbol{J}$[2] has its own weight per a_k, only an option of j_{to} has a bias for that of j_{from} and j_{to}; that is, $o^{j_{\text{from}}}$ does not have a bias. The bias $b_{(o^{j_{\text{to}}}, o^{j_{\text{from}}})}^{w_{o^{j_{\text{to}}}}} \in [0, 0.5]$ represents an increase of importance for $o^{j_{\text{to}}}$ in a

---

[1] Our dataset is publicly available on GitHub: https://github.com/gucci-j/negotiation-breakdown-detection, and details on the negotiation interface and procedures are given in Appendix A.

[2] In our implementation, j_{from} is equivalent to "position," and j_{to} corresponds to "company."

	JI	DN	CB
# of dialogues	2,639	6,251	6,682
Avg turns per dialogue	12.7	4.97	7.53
Avg words per turn	6.12	8.56	13.60
Vocab size	4,476	2,631	12,139
Agreed [%]	92.9	76.2	74.9
PO solutions [%]	13.4	75.0	
PO bids for all bids [%]	0.98	18.0	
# of all bids per dialogue	9,920	22.5	
Avg score	6.4 / 10	5.7 / 10	

Table 3: Quantitative comparison of the three negotiation datasets: "PO" stands for Pareto optimal.

particular pair of options $(o^{j_\text{to}}, o^{j_\text{from}})$. Note that each weight and bias is initialized using uniform random numbers within a predefined range.

Scoring Function We define a scoring (utility) function to calculate a negotiation score. The weight of option $w_{o^{j_\text{to}}}$ is normalized after considering bias. More specifically, when an option o^{j_from} is in a draft agreement, the normalized weight of the option $w'_{o^{j_\text{to}}}$ is calculated using min-max normalization of $w_{o^{j_\text{to}}} + b^{w_{o^{j_\text{to}}}}_{(o^{j_\text{to}}, o^{j_\text{from}})}$ over $\boldsymbol{O}^{j_\text{to}}$. Thus, the scoring function is defined as follows:

$$\begin{aligned} U_{a_k}(s) = &\sum_{i \in \boldsymbol{I}} w_i^{a_k} w_{o_s^i}^{a_k} \\ &+ \sum_{(j_\text{from}, j_\text{to}) \in \boldsymbol{J}} \left(w_{j_\text{from}}^{a_k} w_{o_s^{j_\text{from}}}^{a_k} + w_{j_\text{to}}^{a_k} w'^{a_k}_{o_s^{j_\text{to}}} \right) \end{aligned}$$

where o_s^i is the option of i and is included in a draft agreement s. The function is derived from a linear additive utility function, utilized in automated bilateral negotiation (Baarslag et al., 2016) and in Lewis et al. (2017).

3.3 Data Collection

We hired workers through Amazon Mechanical Turk to collect human-human dialogues. Only those based in the USA with at least 1,000 previous HITs and an approval rating of over 95% could join our experiments. Before each session, the workers read the task description and instructions for negotiating with the opponent[1]. During a negotiation, each worker could propose a draft agreement up to three times and was asked to send six messages or more in total to submit the proposal. We paid \$0.20 per dialogue and gave a \$(score − 5)/5 bonus if the score was more than 5/10 to promote efficient negotiations.

3.4 Quantitative Comparison

Table 3 shows the quantitative comparison of three negotiation dialogue corpora. The vocabulary size is the largest in the CB dataset because it handles several categories of listed products. The JI and DN datasets focus on a single domain, and of the two, the former has the larger vocabulary size. The average number of turns per dialogue in the JI dataset is the largest of the three, though it has the smallest average number of words per turn. These statistics indicate that participants in the JI dataset likely had enough conversations to reach agreement.

Agreement Ratio The JI dataset had the highest agreement ratio of 92.9%, a sharp contrast with the values of 76.2% and 74.9% for the DN and CB datasets. This difference may be because the participants in the JI dataset could propose intermediate offers up to three times each, while those in the existing corpora could only submit one proposal per session.

Complexity of Negotiation Scenarios The JI dataset has far fewer Pareto optimal[3] solutions for agreements than the DN dataset, which can be ascribed to the following reasons: (1) the larger number of issues and options in the JI dataset, with 9920 possible solutions per dialogue, and (2) the introduction of an interdependent relationship that prevented the scoring function from following a standard linear additive utility function. As a result, participants in the JI dataset struggled to find better solutions and might have compromised with each other more often than in the DN dataset.

4 Task Description

Task We formally define the task of breakdown detection in negotiation dialogues. Let D be a negotiation dialogue between two negotiators, composed of $n \in \mathbb{N}$ turn's utterances $\{s_1, s_2, \ldots, s_n\}$, where each utterance s is a message from one of the negotiators and includes one or more sentences. Given D, the task is to label D as either a success (reaching an agreement: 0) or a breakdown (failing to find an agreement: 1).

Evaluation Metrics To evaluate the effectiveness of the different approaches, we employ area under curve (ROC-AUC) and confusion matrix (CM), both of which are based on Yamaguchi and

---

[3] When an agent's score cannot be improved without lowering the opponent's score, a solution is called Pareto optimal.

Fujita (2020). We also use average precision (AP) to consider the imbalanced nature of breakdown labels in negotiation datasets.

5 Methodology

This section introduces our breakdown detection approach using a dialogue act-based feature and ML models, including linear and NN-based models. The intuition that guides this feature is that because a breakdown dialogue should have distinct flow (e.g., many disagreements), focusing on the dialogue flow can help detect this type of breakdown.

5.1 Dialogue Act Extraction

Our dialogue acts and their extraction are based on He et al. (2018), but we made some changes in the extraction process to capture dialogue flow effectively. The process consists of two stages: (1) pattern matching and (2) filtering and alignment. The first step is almost identical to He et al. (2018), but the second is newly designed for this study.

Pattern Matching Given a dialogue turn, we extract dialogue acts according to the matching patterns (Table 4) using regular expressions[4]. If there is no matched pattern in it, an unknown tag `<unk>` is given. Note that because negotiators in the JI dataset can propose intermediate offers up to three times and because such offers are part of negotiations, we add corresponding dialogue acts whenever these offers are detected during conversations.

Filtering and Alignment He et al. (2018) only extracted one dialogue act per turn. However, because negotiators could send one or more sentences for each turn in the DN, CB, and JI datasets, there may have been two or more dialogue acts. To capture the dialogue flow in detail while matching noise due to the rule-based extraction is reduced, we filter extracted dialogue acts in a way that matches Figure 1, which only allows dialogue acts to appear in the designated order. If an illegal dialogue act follows a matched one, all remaining unmatched ones will be discarded. The constrained flow is motivated by an alternating-offer protocol (Rubinstein, 1982) utilized in automated negotiation (Baarslag et al., 2013b), where one agent proposes a draft agreement, and the other accepts it or makes a counter-offer. Although negotiation dialogues do not have a well-defined negotiation protocol, unlike the case of automated negotiation, we assume that human negotiators should follow an unwritten code to reach agreement with their opponents. Table 2 shows an example of extracted dialogue acts along with the text.

Dialogue act	Matching Pattern
`<greet>`	*hi, hello, yo, hey, hiya, howdy, how are you, good day, good afternoon, good morning*
`<disagree>`	Generic – *isn't, worse, bad, sorry, no, not, nothing, don't, can't, cannot, afraid, a lot lower/higher, too much/high/low* JI – An intermediate offer is rejected.
`<agree>`	*ok, okay, no problem, yes, great, perfect, thanks, gracias, thx, thank you, pleasure, fine, deal, cool, that works, that will work, that works, it will work, sounds good, very good, looks good, i can do*
`<inquire>`	*what, where, when, which, how's, how about, how does, do you, did you, will you, would you, could you, are you, do we, did we, could we, do i, let me know, ?*
`<propose>`	Generic – Any digits, *come down, highest, lowest, go higher/lower, i would like* DN – *ball(s), hat(s), book(s)* JI – A new intermediate offer is proposed.
`<inform>`	A previous utterance ends with `<inquire>` and its reply does not contain any other tags.

Table 4: Matching patterns for dialogue acts.

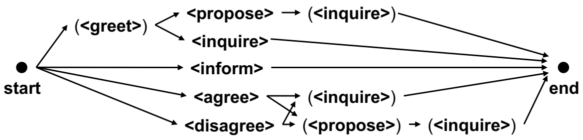


Figure 1: Constrained flow of each dialogue turn. Dialogue acts in parenthesis denote that they do not always appear in each turn.

5.2 Using Dialogue Act-based Features as Inputs for ML Models

Once we extract all features from a dialogue, we concatenate each turn with the addition of a separator tag `<sep>` to the head of each turn and an end tag `<end>` to the end of the dialogue. We then create an input vector for linear or NN-based models and use it to train the model. The input vector is produced as follows:

Linear Models We create a count vector by counting the number of each dialogue act per dialogue, including `<unk>`, `<sep>` and `<end>` tags.

---

[4] If there are two or more matched patterns for the same dialogue act, only the first matched pattern is kept.

NN-based Models We convert each extracted dialogue act into a one-hot representation $\boldsymbol{e} \in \mathbb{R}^{1\times10}$, which includes a padding tag `<pad>`. We then concatenate all one-hot representations in time series per dialogue, which generates an input matrix $\boldsymbol{E} \in \mathbb{R}^{n\times10}$, where n is the number of extracted dialogue acts, including padding.

6 Experimental Settings

6.1 Classification Models

We experiment with linear and NN-based models trained with either text-based or dialogue act-based features:

LR-BOW A logistic regression model trained with bag-of-words features weighted by TF-IDF.

GRU A GRU-based model with a linear layer on top of recurrent units. For text-based inputs, we used frozen pre-trained 300-dimensional word embeddding (GloVe) (Pennington et al., 2014). We also considered the model with a self-attention mechanism (GRU-Att) (Zhou et al., 2016).

BERT A pre-trained bidirectional encoder representations from transformers (BERT)-based model (Devlin et al., 2019) for only text-based inputs. We fine-tuned uncased $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models with one linear layer on the top of the `[CLS]` representation for binary classification.

Random A naive classifier that predicts negotiation outcomes by respecting training set's class distribution.

6.2 Data and Preprocessing

We employed three negotiation datasets compared in Table 3 for our experiments. The breakdown label of each dataset was assigned as follows. **DN**: A log has either a `<disagree>` or `<no_agreement>` tag inside an `<output>` tag. **CB**: A log does not have an offer price. **JI**: A "status" in a log is not "completed." For the CB and JI datasets, we removed short dialogues with less than three turns, as these are often labeled as breakdown and rarely include bargaining components, such as proposals. After the removal, the breakdown ratios of the CB and JI datasets were 18.9% and 4.9%. We preprocessed texts with lower-casing and inserted the `<sep>` and `<end>` tags into each dialogue, as in the dialogue act-based case. We tokenized the texts using spaCy[5]. For BERT, we used a pre-trained BERT tokenizer provided by the Transformers library (Wolf et al., 2020).

6.3 Implementation Details

We trained and tested models using stratified five-fold cross-validation. The model-specific implementation details are as follows:

Linear Model We implemented an LR-BOW model using Scikit-learn (Pedregosa et al., 2011) and trained it on Intel Core i5 (2.9 GHz - 6267U). We tested the n-gram combination of $\{(1, 1), (1, 2), (1, 3)\}$. We applied L2 regularization and weight adjustments to make the weights inversely proportional to the labels in training data.

NN-based Models We set the maximum number of epochs to 100 for GRU-based models and 20 for BERT-based models, with early stopping. We further split the training folds into training (80%) and validation subsets (20%). We used the binary cross-entropy loss and optimized the models with an Adam optimizer (Kingma and Ba, 2014). We implemented the models using PyTorch (Paszke et al., 2019) and tuned their hyperparameters based on validation F_1[6]. For BERT-based models, we utilized the implementation provided by HuggingFace (Wolf et al., 2020). We trained and tested our models with NVIDIA Tesla V100 (SXM2 - 32GB).

7 Results and Analysis

7.1 Quantitative Results

Results in Existing Corpora We can observe from Table 5 that a fine-tuned $\text{BERT}_{\text{BASE}}$ model shows the best AP for the DN and CB datasets. Moreover, NN-based models with text-based features exhibit results that are comparable to those of the best-performing models in terms of AP, in the 95% confidence interval. The proposed approach (GRU_{TAG}) also showed comparable results for either AP or CM in both datasets. Although a logistic regression model with text-based features ($\text{LR-BOW}_{\text{TEXT}}$) produced poor results in terms of AP, it showed the best results for the pair of FN and TP and that of TN and FP in the DN and CB datasets, respectively.

Results in Proposed Corpus Our GRU-based models with dialogue act-based features (GRU_{TAG} and $\text{GRU-Att}_{\text{TAG}}$) showed by far the best AP of all

---

[5] https://spacy.io/

[6] Details concerning the hyperparameter selection are given in Appendix B.

Model	DealOrNoDeal ROC-AUC	AP	TN	FP	FN	TP
LR-BOW$_{\text{TAG}}$	.500 (n/a)	.238 (n/a)	1.00 (n/a)	.000 (n/a)	1.00 (n/a)	.000 (n/a)
GRU$_{\text{TAG}}$	.839 (.018)*	.766 (.031)*	.944 (.035)*	.056 (.035)*	.393 (.034)	.607 (.034)
GRU-Att$_{\text{TAG}}$	.834 (.012)*	.764 (.022)*	**.946** (.030)	**.054** (.030)	.406 (.014)	.594 (.014)
LR-BOW$_{\text{TEXT}}$	.838 (.024)	.745 (.031)	.891 (.011)	.109 (.011)	**.345** (.030)	**.655** (.030)
GRU$_{\text{TEXT}}$	.838 (.022)*	.772 (.031)*	.942 (.022)*	.058 (.022)*	.371 (.012)*	.629 (.012)*
GRU-Att$_{\text{TEXT}}$	.845 (.023)*	**.779** (.026)	.942 (.016)*	.058 (.016)*	.361 (.021)*	.639 (.021)*
BERT$_{\text{BASE}}$	.850 (.017)*	**.779** (.030)	.942 (.013)*	.058 (.013)*	.349 (.037)*	.651 (.037)*
BERT$_{\text{LARGE}}$	**.851** (.018)	.769 (.036)*	.940 (.011)*	.060 (.011)*	.354 (.036)*	.646 (.036)*
Random	.502 (.006)	.238 (.002)	.754 (.014)	.246 (.014)	.750 (.008)	.250 (.008)

Model	CraigslistBargain ROC-AUC	AP	TN	FP	FN	TP
LR-BOW$_{\text{TAG}}$	.500 (n/a)	.189 (n/a)	1.00 (n/a)	.000 (n/a)	1.00 (n/a)	.000 (n/a)
GRU$_{\text{TAG}}$	.897 (.013)	.702 (.035)	.906 (.021)*	.094 (.021)*	.306 (.032)*	.694 (.032)*
GRU-Att$_{\text{TAG}}$	.893 (.016)	.679 (.035)	.894 (.037)	.106 (.037)	.312 (.050)	.688 (.050)
LR-BOW$_{\text{TEXT}}$	.874 (.013)	.685 (.024)	**.925** (.021)	**.075** (.021)	.398 (.029)	.602 (.029)
GRU$_{\text{TEXT}}$	.919 (.011)*	.755 (.033)*	.921 (.015)*	.079 (.015)*	.267 (.040)*	.733 (.040)*
GRU-Att$_{\text{TEXT}}$	**.920** (.014)	.737 (.025)*	.918 (.013)*	.082 (.013)*	**.261** (.040)	**.739** (.040)
BERT$_{\text{BASE}}$	**.920** (.008)	**.756** (.021)	.914 (.017)*	.086 (.017)*	.301 (.052)*	.699 (.052)*
BERT$_{\text{LARGE}}$	.910 (.017)*	.744 (.040)*	.919 (.003)*	.081 (.003)*	.299 (.033)*	.701 (.033)*
Random	.501 (.015)	.190 (.004)	.814 (.016)	.186 (.016)	.813 (.038)	.187 (.038)

Model	JobInterview ROC-AUC	AP	TN	FP	FN	TP
LR-BOW$_{\text{TAG}}$	.500 (n/a)	.049 (n/a)	1.00 (n/a)	.000 (n/a)	1.00 (n/a)	.000 (n/a)
GRU$_{\text{TAG}}$	.902 (.016)*	**.418** (.035)	**.971** (.012)	**.029** (.012)	.646 (.102)*	.354 (.102)*
GRU-Att$_{\text{TAG}}$	**.915** (.014)	.416 (.076)*	.953 (.034)	.047 (.034)	**.582** (.186)	**.418** (.186)
LR-BOW$_{\text{TEXT}}$	.736 (.058)	.178 (.045)	.913 (.051)	.087 (.051)	.701 (.082)*	.299 (.082)*
GRU$_{\text{TEXT}}$	.539 (.083)	.093 (.024)	.966 (.032)*	.034 (.032)*	.937 (.031)	.063 (.031)
GRU-Att$_{\text{TEXT}}$	.547 (.089)	.086 (.017)	.964 (.027)*	.036 (.027)*	.922 (.065)	.078 (.065)
BERT$_{\text{BASE}}$	.705 (.059)	.172 (.072)	.951 (.040)	.049 (.040)	.802 (.111)*	.198 (.111)*
BERT$_{\text{LARGE}}$	.725 (.059)	.171 (.043)	.959 (.024)*	.041 (.024)*	.810 (.094)*	.190 (.094)*
Random	.515 (.025)	.053 (.005)	.951 (.006)	.049 (.006)	.921 (.055)	.079 (.055)

Table 5: Performance comparison for three negotiation dialogue datasets: Best mean results are in **bold**. Values in parenthesis represent standard deviations over the five test folds. Values marked with * are within the 95% confidence interval of the best score for a given metric. Confusion matrices are normalized on a set each of true negative (TN) and false positive (FP), and true negative (TP) and false negative (FN).

models and better results in other metrics. For text-based models, an LR-BOW$_{\text{TEXT}}$ model showed better results in terms of AP, FN, and TP than NN-based models. While text-based GRU models could not detect signs of breakdowns at all, BERT-based models could detect them with a TP ratio of 19.8% (base) and 19.0% (large).

Discussion First, dialogue act-based features only worked with sequential models. This result is in line with our key concept of capturing negotiation flow. Because the LR-BOW$_{\text{TAG}}$ model could not consider sequential information, it could not detect breakdowns at all. Second, an LR-BOW$_{\text{TEXT}}$ model worked well in all datasets, indicating that text-based features themselves contain breakdown information. However, this approach produced more misclassification for successful dialogues in the DN and JI datasets than other models, but it

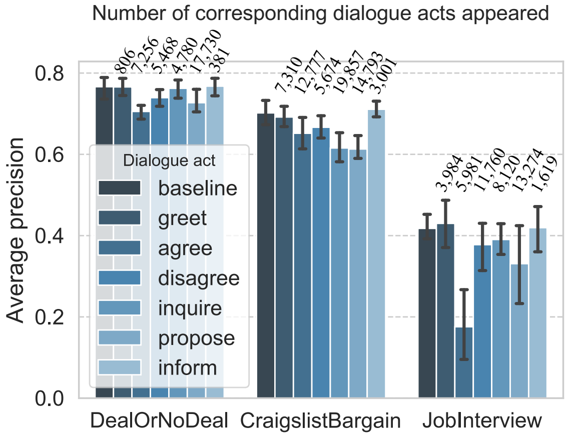


Figure 2: Classification performance comparison on five test folds when replacing a specific dialogue act with an unknown `<unk>` tag. Error bars denote the 95% confidence interval.

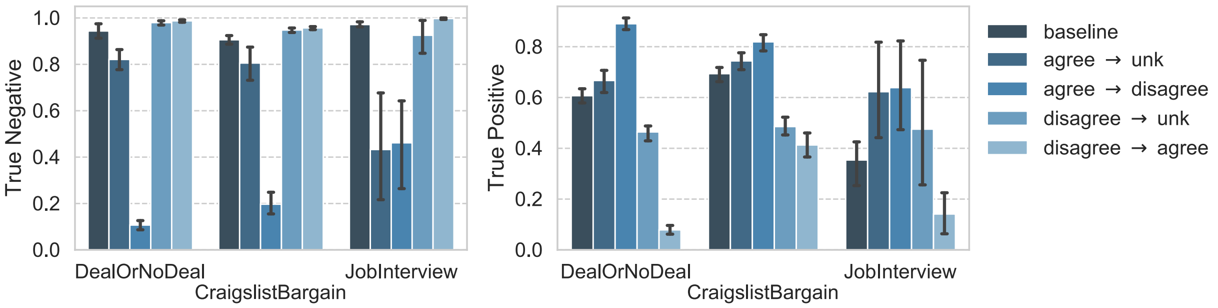


Figure 3: Performance comparison on five test folds when replacing `<agree>` and `<disagree>` tags with their counterpart or an `<unk>` tag. Error bars denote the 95% confidence interval.

could detect fewer breakdowns in the CB dataset. Because we intend to support human-human negotiation, accurate classification for both cases is vital to providing beneficial feedback to negotiators. Thus, the use of this approach is not helpful to our task. Third, NN-based models with text-based features did not perform well in the JI dataset. This was likely due to the far smaller breakdown ratio of 4.9% in the dataset compared to 23.8% and 18.9% in the DN and CB datasets. However, BERT-based models showed far better results than GRU-based ones in terms of the TP ratio. We hypothesize that BERT's rich contextualized information helped detect signs of breakdown.

7.2 Ablation Study

We conducted two ablation studies to better understand dialogue act-based input features. We first analyzed the importance of each dialogue act by replacing it with an unknown tag and tested with our best-performing model (GRU_{TAG}) over the five test folds. The `<agree>` tag was important for breakdown detection across the three corpora, despite its infrequency, especially in the DN and JI datasets (Figure 2). The frequent tag `<propose>` also played an important role in classification. By contrast, the `<disagree>` and `<inquire>` tags were not important except for the `<inquire>` tag in the CB dataset, possibly due to its highest frequency. Finally, the `<greet>` and `<inform>` tags were the least important in all datasets as these appeared less frequently and are not as closely related to breakdown as the others.

Next, we verified whether the GRU_{TAG} model captured the roles of `<agree>` and `<disagree>` tags in the breakdown detection task by replacing these tags with their counterpart or an `<unk>` tag (Figure 3). By replacing an `<agree>` tag with a `<disagree>` tag, we saw a rise in a TP ratio and a significant drop in a TN ratio compared to the baseline. When the `<disagree>` tag was replaced with an `<agree>` tag, the TN ratio slightly increased, while the TP ratio significantly decreased. These results suggest that the model properly took into account the roles of "`<agree>`" and "`<disagree>`" to some extent, and the number of such tags appeared played an important role in detecting a breakdown. While replacement with an `<unk>` tag also showed a similar trend, except with the `<disagree>` tag in the JI dataset, this was probably due to the relative increase of the counterpart.

FP (DN)	`<sep>` *i'd love to take a book and two hats off your hands* `<sep>` *hm, not many points for me but i'll agree to that.* `<end>`
	`<sep> <propose> <sep> <disagree> <end>`
FN (CB)	`<sep>` *hello, i am very interested in your car. however $12000 is out of my price range for a car that is 7 years old. i offer $6000 and i will pick up the car myself.* `<sep>` *there is no possible way i could go that low. i would take $11, 000* `<sep>` *that's fine, i will go elsewhere with my money.* `<sep>` *okay* `<end>`
	`<sep> <greet> <propose> <sep> <disagree> <propose> <sep> <agree> <sep> <agree> <end>`

Table 6: Examples of misclassified dialogues with extracted dialogue acts.

7.3 Error Analysis

Last, we conducted error analyses to examine the behavior of a GRU_{TAG} model and reveal its potential limitations. The first example is an FP sample from the DN dataset, where the model possibly focused on a `<disagree>` tag corresponding to *not*. The second one is an FN sample from the CB dataset, in which the model might have focused on repetitive `<agree>` tags. We consider that the proposed approach could not cope with euphemistic phrases because of the rule-based dialogue act extraction. Thus, annotating negotiation corpora with dialogue acts will be an important research direction for more precise detection.

8 Conclusions and Future Work

This study proposed a job interview negotiation dialogue dataset with 2639 dialogues and increased complexities compared to existing datasets to help propel development of the study of human-human negotiation support and goal-oriented dialogue systems. We also proposed a dialogue act-based breakdown detection model that can focus on negotiation flow. Our approach ($\mathrm{GRU}_{\mathrm{TAG}}$) showed comparable results when used with existing datasets and better results for the proposed dataset than models trained with text-based features. In the future, we intend to explore another application of dialogue act-based features to related tasks, such as preference estimations. We will also utilize the proposed corpus in related tasks in human-human negotiation support and goal-oriented dialogue systems.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623–2631. Association for Computing Machinery.

Muddasser Alam, Alex Rogers, and Sarvapali Ramchurn. 2013. Interdependent multi-issue negotiation for energy exchange in remote communities. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, pages 25–31.

Nicholas Asher, Julie Hunter, Mathieu Morey, Benamara Farah, and Stergos Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: the STAC corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 2721–2727. European Language Resources Association.

Tim Baarslag, Reyhan Aydoğan, Koen V Hindriks, Katsuhide Fujita, Takayuki Ito, and Catholijn M Jonker. 2015. The automated negotiating agents competition, 2010–2015. *AI Magazine*, 36(4):115–118.

Tim Baarslag, Katsuhide Fujita, Enrico H. Gerding, Koen Hindriks, Takayuki Ito, Nicholas R. Jennings, Catholijn Jonker, Sarit Kraus, Raz Lin, Valentin Robu, and Colin R. Williams. 2013a. Evaluating practical negotiating agents: Results and analysis of the 2011 international competition. *Artificial Intelligence*, 198:73 – 103.

Tim Baarslag, Mark J. Hendrikx, Koen V. Hindriks, and Catholijn M. Jonker. 2016. Learning about the opponent in automated bilateral negotiation: A comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, 30(5):849–898.

Tim Baarslag, Koen Hindriks, and Catholijn Jonker. 2013b. Acceptance conditions in automated negotiation. In *Complex Automated Negotiations: Theories, Models, and Software Competitions*, pages 95–111. Springer.

Minhao Cheng, Wei Wei, and Cho-Jui Hsieh. 2019. Evaluating and enhancing the robustness of dialogue systems: A case study on a negotiation agent. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3325–3335. Association for Computational Linguistics.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

Heriberto Cuayáhuitl, Simon Keizer, and Oliver Lemon. 2015. Strategic dialogue management via deep reinforcement learning. *CoRR*, abs/1511.08099.

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

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2333–2343. Association for Computational Linguistics.

Ryuichiro Higashinaka, Kotaro Funakoshi, Yuka Kobayashi, and Michimasa Inaba. 2016. The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 3146–3150. European Language Resources Association.

Chiori Hori, Julien Perez, Ryuichiro Higashinaka, Takaaki Hori, Y-Lan Boureau, Michimasa Inaba, Yuiko Tsunomori, Tetsuro Takahashi, Koichiro Yoshino, and Seokhwan Kim. 2019. Overview of the sixth dialog system technology challenge: Dstc6. *Computer Speech & Language*, 55:1 – 25.

Takayuki Ito, Mark Klein, and Hiromitsu Hattori. 2006. A negotiation protocol for agents with nonlinear utility functions. In *Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence*.

Kosui Iwasa and Katsuhide Fujita. 2018. Prediction of nash bargaining solution in negotiation dialogue. In *PRICAI 2018: Trends in Artificial Intelligence*, pages 786–796. Springer International Publishing.

Catholijn Jonker, Reyhan Aydogan, Tim Baarslag, Katsuhide Fujita, Takayuki Ito, and Koen Hindriks. 2017. Automated negotiating agents competition (anac). In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 5070–5072.

A. Kardan and H. Janzadeh. 2008. A multi-issue negotiation mechanism with interdependent negotiation issues. In *Proceedings of the Second International Conference on the Digital Society*, pages 55–59.

Ralph L. Keeney and Howard Raiffa. 1993. *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge University Press.

Simon Keizer, Markus Guhe, Heriberto Cuayáhuitl, Ioannis Efstathiou, Klaus-Peter Engelbrecht, Mihai Dobre, Alex Lascarides, and Oliver Lemon. 2017. Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 480–484. Association for Computational Linguistics.

Wolfgang Ketter, John Collins, and Prashant Reddy. 2013. Power tac: A competitive economic simulation of the smart grid. *Energy Economics*, 39:262 – 270.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Vasily Konovalov, Ron Artstein, Oren Melamud, and Ido Dagan. 2016. The negochat corpus of human-agent negotiation dialogues. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 3141–3145. European Language Resources Association.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453. Association for Computational Linguistics.

Raz Lin and Sarit Kraus. 2010. Can automated agents proficiently negotiate with humans? *Commun. ACM*, 53(1):78–88.

Johnathan Mell, Jonathan Gratch, Tim Baarslag, Reyhan Aydoğan, and Catholijn M. Jonker. 2018. Results of the first annual human-agent league of the automated negotiating agents competition. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, pages 23–28. Association for Computing Machinery.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. Association for Computational Linguistics.

Volha Petukhova, Harry Bunt, and Andrei Malchanau. 2017. Computing negotiation update semantics in multi-issue bargaining dialogues. In *Proceedings of SEMDIAL 2017 (SaarDial) Workshop on the Semantics and Pragmatics of Dialogue*, pages 87–97.

Volha Petukhova, Christopher Stevens, Harmen de Weerd, Niels Taatgen, Fokie Cnossen, and Andrei Malchanau. 2016. Modelling multi-issue bargaining dialogues: Data collection, annotation design and corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 3133–3140. European Language Resources Association.

Howard Raiffa, John Richardson, David Metcalfe, et al. 2002. *Negotiation analysis: The science and art of collaborative decision making*. Harvard University Press.

Ariel Rubinstein. 1982. Perfect equilibrium in a bargaining model. *Econometrica*, 50:97–109.

Leigh L. Thompson, Jiunwen Wang, and Brian C. Gunia. 2010. Negotiation. *Annual Review of Psychology*, 61(1):491–515.

David Traum, Stacy C. Marsella, Jonathan Gratch, Jina Lee, and Arno Hartholt. 2008. Multi-party, multi-issue, multi-strategy negotiation for multi-modal virtual agents. In *Proceedings of the Eighth International Conference on Intelligent Virtual Agents*, pages 117–130. Springer.

Minhong Wang, Huaiqing Wang, Doug Vogel, Kuldeep Kumar, and Dickson K. W. Chiu. 2009. Agent-based negotiation and decision making for dynamic supply chain formation. *Engineering Applications of Artificial Intelligence*, 22(7):1046–1055.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Atsuki Yamaguchi and Katsuhide Fujita. 2020. Breakdown detection in negotiation dialogues (student abstract). In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 13969–13970.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212. Association for Computational Linguistics.

Yiheng Zhou, He He, Alan W Black, and Yulia Tsvetkov. 2019. A dynamic strategy coach for effective negotiation. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 367–378. Association for Computational Linguistics.

A Job Interview Negotiation Dataset

Here, we introduce the negotiation interface and negotiation procedures. Our dataset and negotiation interface are available at `https://github.com/gucci-j/negotiation-breakdown-detection`.

A.1 Negotiation Interface

We developed an online negotiation interface for our job-interview negotiation, which implemented all mathematical settings such as preferences and a scoring function discussed in the body of the paper. Figure 4 shows the screenshot of our negotiation interface.

At the beginning of each negotiation session, the interface generates negotiators' preferences and displays them next to the corresponding issues and options so that the negotiators can easily understand which issue and option is important for them.

During the session, whenever the negotiators select a new solution, the interface calculates the score of its solution according to the scoring function described in Subsection 3.2 and displays it with the corresponding evaluation. The evaluation is based on Table 7 and intended for providing feedback to the negotiators to promote a better agreement.

At the end of the session, the interface stores the log that consists of the preferences of the participants, dialogue history, proposed offers and settled agreement in `json` format.

Score	Evaluation
< 50	Very bad
< 60	Bad
< 70	Fair
< 80	Good
< 90	Very good
≥ 90	Excellent

Table 7: Correspondence table between the score and the evaluation.

A.2 Negotiation Procedures

Before entering a negotiation session, each negotiator reads the instruction page that describes the outline of the negotiation, its procedures and some precautions (e.g., the maximum number of proposals per negotiator).

During the session, the negotiators can talk to their opponent using the left-hand side of the negotiation interface (Figure 4), while they can select an option for each issue in the right-hand side of the interface. Besides, they can also check the current score, its evaluation and estimated HIT reward for the selected options.

When the negotiators believe that they had sufficient discussion, they can propose a draft agreement by clicking the "PROPOSE" button shown in the bottom-left side of the interface. Once it is sent to the opponent, the opponent can check its details and score with the "ACCEPT" button shown on the interface. If the opponent clicks the button, the negotiation is regarded as successful. Otherwise, the negotiation continues until both the sides exceed the maximum number of propositions. If exceeding the limit, the negotiation is regarded as a breakdown, and the score of each negotiator is recorded as zero.

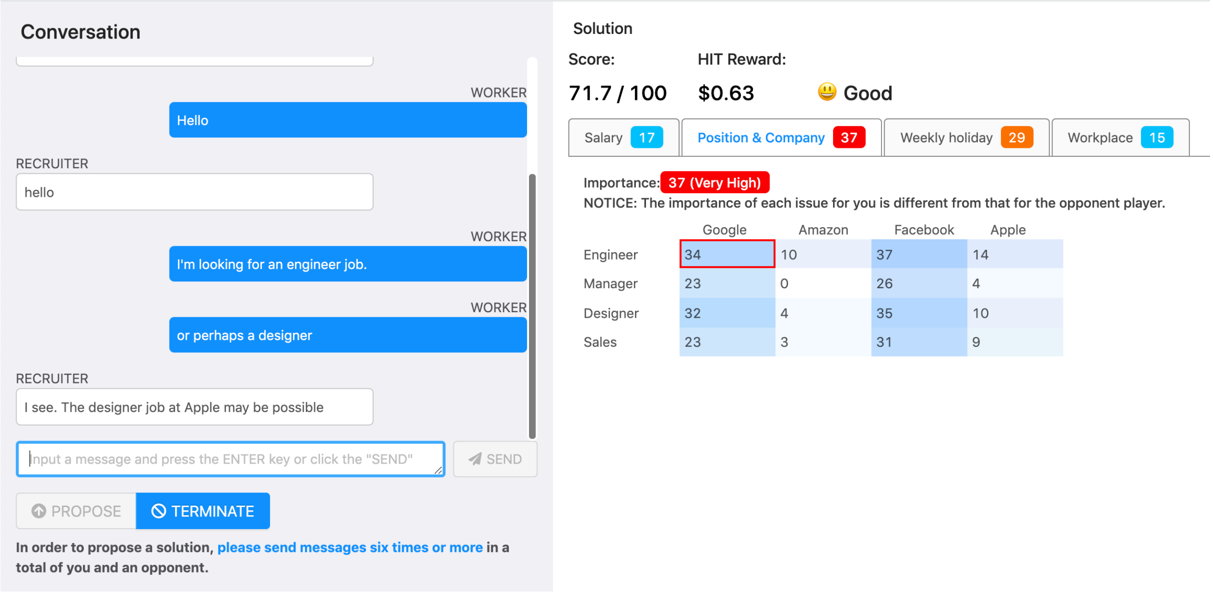
Hyperparameter	**Value or search space**
Maximum training epochs	100
Mini-batch size	64
Adam β_1	0.9
Adam β_2	0.999
Learning rate	$[10^{-5}, 10^{-2}]$
Early stopping patience value	8
Number of GRU layers	$[1, 4]$
Number of GRU hidden units	$[64, 256]$
Bidirectional	`True` or `False`
Recurrent dropout rate	$(0.0, 1.0)$
Classifier dropout rate	$(0.0, 1.0)$

Table 8: Hyperparameters and search space for GRU-based models. If "bidirectional" is `True`, a model becomes a bidirectional GRU.

B Hyperparameter Tuning

Linear Models For the DN dataset, n-gram combination of $(1, 3)$ (uni-gram, bi-gram, and tri-gram) was chosen. For the CB dataset, that of $(1, 2)$ (uni-gram and bi-gram) was selected. For the JI dataset, that of $(1, 1)$ (uni-gram) was chosen. Since none of the models trained with dialogue act-based features did not work, these have no optimal n-gram combinations.

Neural Network-based Models We tuned the hyperparameters of all NN-based models employed in our experiments using the Optuna framework (Akiba et al., 2019). We split training folds into training (80%) and validation (20%) subsets. We tested 100 hyperparameter combinations and evaluated their performance based on F_1 in each validation subset. Tables 8 and 9 show the hyperparameters and search space for GRU and BERT-based models, respectively.



Conversation

WORKER
Hello

RECRUITER
hello

WORKER
I'm looking for an engineer job.

WORKER
or perhaps a designer

RECRUITER
I see. The designer job at Apple may be possible

In order to propose a solution, please send messages six times or more in a total of you and an opponent.

Solution

Score: 71.7 / 100

HIT Reward: $0.63

😃 Good

Salary 17 | Position & Company 37 | Weekly holiday 29 | Workplace 15

Importance: 37 (Very High)

NOTICE: The importance of each issue for you is different from that for the opponent player.

	Google	Amazon	Facebook	Apple
Engineer	34	10	37	14
Manager	23	0	26	4
Designer	32	4	35	10
Sales	23	3	31	9

Figure 4: Negotiation interface used for the JI dataset. Each value shown next to an issue or an option denotes its importance for a negotiator. The score and importance of each issue and option were calculated by the interface based on the mathematical settings discussed in the body of the paper. Note that the score shown on the interface are multiplied by ten for the ease of players' understanding.

Hyperparameter	**Value or search space**
Maximum training epochs	20
Mini-batch size	16 ($\text{BERT}_{\text{LARGE}}$) 32 ($\text{BERT}_{\text{BASE}}$)
Adam β_1	0.9
Adam β_2	0.999
Maximum sequence length	196 (CB and JI datasets) 128 (DN dataset)
Learning rate for pre-trained layers	$[10^{-6}, 10^{-3}]$
Learning rate for an additional dense layer	$[10^{-5}, 10^{-2}]$
Learning rate scheduler	{"get cosine schedule with warmup," "get constant schedule with warmup," "get linear schedule with warmup"}
Warmup steps	$[1, 120]$
Early stopping patience value	3
Dropout rate	$(0.0, 1.0)$
Gradient accumulation steps	10 ($\text{BERT}_{\text{LARGE}}$) 5 ($\text{BERT}_{\text{BASE}}$)

Table 9: Hyperparameters and search space for BERT-based models. Each scheduler name corresponds to the one in the Transformers library (Wolf et al., 2020) by replacing blanks with "_".

Neural Data-to-Text Generation with LM-based Text Augmentation

†Ernie Chang, ⊙ Xiaoyu Shen[*], †Dawei Zhu, †Vera Demberg, ⊗ Hui Su
†Dept. of Language Science and Technology, Saarland University
⊙ Amazon Alexa AI, Berlin
⊗ Pattern Recognition Center, Wechat AI, Tencent Inc, China
`{cychang}@coli.uni-saarland.de`

Abstract

For many new application domains for data-to-text generation, the main obstacle in training neural models consists of a lack of training data. While usually large numbers of instances are available on the data side, often only very few text samples are available. To address this problem, we here propose a novel few-shot approach for this setting. Our approach automatically augments the data available for training by (i) generating new text samples based on replacing specific values by alternative ones from the same category, (ii) generating new text samples based on GPT-2, and (iii) proposing an automatic method for pairing the new text samples with data samples. As the text augmentation can introduce noise to the training data, we use *cycle consistency* as an objective, in order to make sure that a given data sample can be correctly reconstructed after having been formulated as text (and that text samples can be reconstructed from data).

On both the E2E and WebNLG benchmarks, we show that this weakly supervised training paradigm is able to outperform fully supervised seq2seq models with less than 10% annotations. By utilizing all annotated data, our model can boost the performance of a standard seq2seq model by over 5 BLEU points, establishing a new state-of-the-art on both datasets.

1 Introduction

Neural data-to-text generation has been the subject of much recent research. The task aims at transforming *source-side* structured data into *target-side* natural language text (Reiter and Dale, 2000; Barzilay and Lapata, 2005). While neural end-to-end systems afford the advantage of easy adaptability (Lebret et al., 2016; Wiseman et al., 2017), huge amounts of data-text pairs are still necessary to perform on par with their rule-based counterparts (van der Lee et al., 2018). This makes using neural systems less appealing: oftentimes, in-domain text samples are not readily available, and there is a high cost to collecting in-domain texts which fit the data samples, and annotating these texts with the data labels – the cost for collecting this data might hence even outweigh the efforts of designing a rule-based system (Gkatzia, 2016). The goal of this work is to improve the performance of neural data-to-text models in scenarios where only very few text samples exist (we assume that these text samples are paired with corresponding data samples). We aim to answer how we can make the most of the scarce annotations, together with large amounts of unlabelled data, in order to push the limit of the neural data-to-text models. Figure 1 illustrates the scenario.

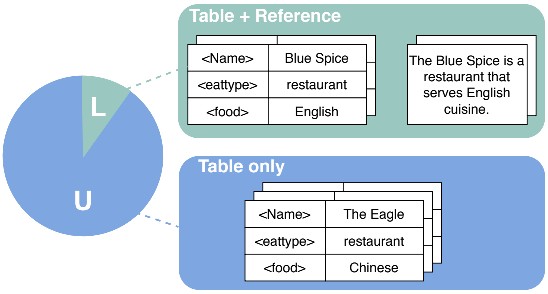


Figure 1: **Few-shot scenario**: The model is expected to learn data-to-text generation with few labeled instances (i.e. table-text pairs). The example is taken from the E2E dataset.

To address the limited-data challenge, we propose a simple yet effective way of augmenting the text side with the pretrained language model (LM) GPT-2 (Radford et al., 2019). Unlike other text augmentation work employed in data-to-text generation systems (Freitag and Roy, 2018; Agarwal et al., 2018), our proposal assumes little to no domain-dependent heuristics. It consists of two steps: (1) information augmentation by slot-value replacement and (2) LM augmentation by GPT-2

---

[*] Work done prior to joining Amazon.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 758–768
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

generation.

Once we have augmented the set of text samples, we are essentially in a similar setting as previously proposed semi-supervised approaches to data-to-text generation Schmitt and Schütze (2019); Qader et al. (2019); Su et al. (2020), which assume the presence of vast amounts of *unpaired* data and text instances. These approaches exploit a *cycle consistency* objective in order to learn a pairing for the data samples. The cycle consistency objective tries to make sure that data samples can be reconstructed correctly from their textual formulations, and similarly that texts can be reconstructed after having been parsed into a data representation.

As the automatically generated text samples from GPT-2 might be very noisy and not pair well with data samples, we align each augmented text sample with its most similar unlabeled data sample, as defined in their encoded vector space. This idea is inspired by recent work on representation matching in MT (Artetxe and Schwenk, 2019; Ruiter et al., 2019). To ensure good quality of the training data, only pairs above a certain similarity threshold ϵ are retained as pseudo pairs for training. The quality of the pseudo pairs will gradually improve as the encoder improves in the training process. In return, the learning of the encoder will also be facilitated with the improved quality of pseudo pairs as a virtuous cycle.

On two data-to-text benchmarks E2E (Novikova et al., 2017) and WebNLG (Gardent et al., 2017), we show that our LM-augmented weakly supervised model succeeds on outperforming fully supervised seq2seq model, though utilizing less than 10% of the data annotations. It even outperforms previous work which *additionally has access to all unpaired text samples*. When trained with full data annotations, it is able to boost the model performance by up to 5 BLEU points, establishing a new state-of-the-art on both datasets.

In summary, this work makes the following contributions:

1. We study the few-shot data-to-text scenario where, unlike previous works, no further target-side text is available.
2. We present an effective way of automatically augmenting target text by resorting to the pretrained LM GPT-2.
3. We propose utilizing the augmented text by a combination of cycle consistency and representation matching. The resulting model outperforms standard seq2seq model with less than 10% data annotations.
4. The proposed model is shown to be complementary with current seq2seq pretraining techniques, and can offer orthogonal improvements when combining both.

2 Related Work

Building neural data-to-text systems with few paired samples (but a large set of unpaired samples) has been a hot research topic recently. Most works adopt the idea of cycle consistency (Zhu et al., 2017), which has been used in many text generation tasks like machine translation (Artetxe et al., 2017; Lample et al., 2017) and style transfer (Prabhumoye et al., 2018; Subramanian et al., 2018). Schmitt and Schütze (2019); Qader et al. (2019); Su et al. (2020); Chang et al. (2020, 2021a,b) applied this idea to the task of data-to-text generation and reported promising results. Ma et al. (2019) separate the generation process into few-shot content selection and surface realization components and learn them separately. Nonetheless, all of these approaches assume the existence of *huge quantity of unpaired text samples*, which, as we mentioned, is an unrealistic assumption for the task of data-to-text generation. Freitag and Roy (2018) proposes to reconstruct usable sequences re-written from data with rules for unsupervised data-to-text generation. Unfortunately, designing these rules require efforts similar to building a template-based system. (Budzianowski and Vulić, 2019; Chen et al., 2020; Peng et al., 2020) tackle the few-shot challenge by finetuning a pretrained LM to incorporate prior knowledge from general-domain text or data-text pairs. We show that our technique is complementary with them and can offer orthogonal improvements when combining both.

3 Problem Formulation

We represent the data samples as D and the text samples as T. In our work, we do not restrict the format of the data. Each $d \in D$ can be a set of key-value pairs, as in Figure 1, or in form of RDF triples as in Gardent et al. (2017). Each text $t \in T$ consists of a sequence of words. In few-shot settings, we are assumed to have (1) k labeled pairs (D_L, T_L) and (2) large quantities of unlabeled data

D_U where $|D_U| \gg k > 0$[1]. This, we believe, is a more realistic setting as unlabeled data are usually abundant and also can be easily fabricated from predefined schemata. Notably, we assume no access to outside resources containing in-domain text. The k annotations are all we know about the text side.

4 Approach

In this section, we first explain our proposed new method for text sample augmentation, and then discuss methods to remove noise and automatically align the data by elaborating on the ideas of cycle consistency and representation matching. Finally, we summarize the approach and present the detailed algorithm.

4.1 Text Augmentation

To mitigate the paucity of the set of text samples T, we propose a pipeline approach to augment the text samples by (1) information augmentation and (2) LM augmentation.

4.1.1 Information Augmentation

We generate additional text samples by performing slot-value replacements. As many data values are exactly copied to the text samples, these copied information can be easily detected and replaced with other values (for the same slot type) to enrich the information space of the text samples. This can be considered as a simplified version of traditional methods of template mining where key words are extracted to construct templates (Kondadadi et al., 2013; Oya et al., 2014). An example is shown in Figure 2. Each text sample is augmented with 10 more distinct text samples or with all possible values being replaced.

The slot-value replacement is efficient to implement. However, it can only detect identical values and augment text with the same combinatorial patterns as the few-shot annotations. To enrich the linguistic realizations of text sentences and enable new combinations of information, we further propose a LM augmentation approach using GPT-2.

4.1.2 LM Augmentation

GPT-2 (Radford et al., 2019) is a language model pretrained on the collected WebText. It has demonstrated remarkable zero-shot multitask adaptability by simply feeding the input of each task into the LM and continuing to generate words. People have also also shown that GPT-2 is able to improve classification tasks via in-domain text augmentation (Papanikolaou and Pierleoni, 2020; Sun et al., 2020). We use a similar technique by first fine-tuning GPT-2 in the few-shot annotations (Wolf et al., 2019), and then applying it to produce synthetic text through an iterative conditional generation process: With initial seeds being samples of T_L plus new samples from information augmentation, the LM iteratively conditions on the previous output sentence to generate in-domain text[2]. Each synthetic sentence is pruned if it (1) is shorter than 5 words or (2) contains only special tokens. The iterative generation is terminated when all tokens in the initial seeds are covered or if the maximum of 100 runs is reached. All the unpruned synthetic text samples are added into the space of T to benefit the learning direction of $t \to d' \to t$ and $\tilde{t} \to t$. Figure 2 depicts the generation process of GPT-2.

In practice, obtaining clean in-domain text requires extreme efforts of designing heuristic rules. Nonetheless, the synthetic text from GPT-2 makes decent sense and can already provide useful signals to drive the learning process.

4.2 Cycle Consistency

The core idea of encouraging cycle consistency is that starting from one sample in a domain, the model first maps it into the other domain, then maps it back (He et al., 2016). The resulting sample should be identical to the original sample. Specifically, let $p_\theta(t|d)$ be the probability distribution to map a data sample d to its corresponding text t, and $p_\phi(d|t)$ be the probability distribution to map text back to data. Starting from a data sample $d \in D$, its objective is:

$$\max_{\phi} \mathbb{E}_{d\sim p(D)} \log p_\phi(d|t'); \; t' \sim p_\theta(t|d) \qquad (1)$$

which basically ensures the consistency in the direction of $d \to t' \to d$. Note that only p_ϕ is updated in this direction and p_θ serves only as as an auxiliary function to provide pseudo samples t' from d. Though it is also possible to update θ at the same time through tricks like Gumbel-softmax (Jang et al., 2016) or REINFORCE (Williams, 1992), we find it did not lead to better performance, yet complicated the training. Similar observations have

---

[1] We force $k > 0$ as we believe a reasonable generation system needs a least a few demonstrations of the annotation.

[2] We adopt the Top-k random sampling setting with $k = 2$ to encourage diversity and reduce repetition (Radford et al., 2019)

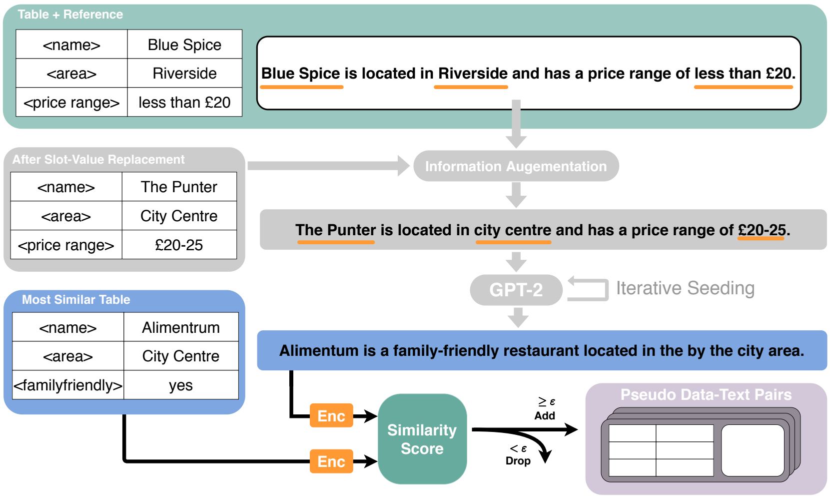


Figure 2: Depiction of *text augmentation* and *representation matching*. Each text sample first goes through information augmentation by slot-value replacement, then passed to GPT-2 with iterative conditional generation. The augmented text samples are paired with the most similar data from the corpus with a threshold cutoff.

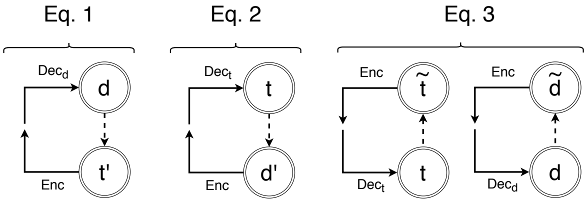


Figure 3: Four directions of cycle consistency. Gradients are backpropagated only through solid lines.

been made in Lample et al. (2018); He et al. (2020); Garcia et al. (2020).

Similarly, starting from a text $t \in T$, the objective is to ensure the consistency in the direction of $t \rightarrow d' \rightarrow t$:[3]

$$\max_{\theta} \mathbb{E}_{t\sim p(T)} \log p_\theta(t|d');\ d' \sim p_\phi(d|t) \qquad (2)$$

Finally, we further add two denoising autoencoding objectives on both the data and text sides:

$$\max_{\theta,\phi} \mathbb{E}_{d\sim p(D),t\sim p(T)} \log p_\phi(d|\tilde{d})p_\theta(t|\tilde{t}) \qquad (3)$$

where $\tilde{d}$ and $\tilde{t}$ are the corrupted versions of d and t. We use the same noise function as in Lample et al. (2018) which randomly permutes and pads a portion of the input. This can encourage the encoder to learn meaningful latent representations by reconstructing the input itself (Currey et al., 2017; Lample et al., 2018).

Figure 3 illustrates all the four directions of the cycle consistency objective.

We use one shared encoder Enc for both the data and text sides. Each data sample is flattened into a sequence by making a list of slot value pairs and fed into the same encoder. Using the same encoder for both types of input gives the model an inductive bias to project similar data/text into surrounding latent space.

We will show later that encoder sharing is essential for a good performance under the few-shot scenario. From the shared encoded space, two separate decoders Dec_d and Dec_t are used to decode d and t respectively[4].

4.3 Representation Matching

Apart from training under the cycle consistency, we further consider matching each synthetic text with its most similar data sample and treating them as supplementary training pairs. Compared with the pseudo d' obtained from back translation (Eq. 2), the matched data samples are extracted from the existing corpus D_U and thereby are guaranteed to be clean. This can provide a much more stable training signal especially at the initial training stage[5]. Previous work has used representation matching to automatically extract pseudo training pairs for machine translation (Artetxe and Schwenk, 2019; Ruiter et al., 2019). Baziotis et al. (2019); Chu and

---

[3] In the MT community, the equivalent step is usually called *back translation* (Sennrich et al., 2016; Lample et al., 2018).

[4] The shared encoding has also been shown effective in other tasks like machine translation (Lample et al., 2018) and image transition (Zhu et al., 2017). We further tried sharing the decoder as in Johnson et al. (2017) but find no improvement (see Table 2).

[5] In theory, as we can fabricate arbitrary possible data samples from the predefined schema and add to the corpus, we can always find one matched data for a text samples.

Liu (2019) also demonstrate that the representation similarity between input-output pairs can serve as a useful regularization for unsupervised text summarization. We adopt a similar idea to create pseudo pairs based on their cosine similarity in the representation space. To summarize, the process of representation matching can be described as:

$$\max_{\theta,\phi} \mathbb{E}_{t\sim p(T')} \mathbb{1}_{cos(d^*,t)>\varepsilon}\big(\log p_\theta(t|d^*) + \log p_\phi(d^*|t)\big); \quad (4)$$

$$d^* = \arg\max_{d\in D} \cos(d,t)$$

where T' is augmented text from the LM and $\mathbb{1}$ is the indicator function. We also perform mean pooling over the encoded representations before matching them. ε is a threshold. Pseudo pairs with a cosine similarity less than ε will be discarded. Ideally, as the encoder improves, the pseudo pairs created by representation matching will make more sense, which can in turn benefit the training of the encoder.

4.4 Summary

Apart from the above unsupervised objective, on the few annotated data-text pairs, we can impose the supervised objective:

$$\max_{\theta,\phi} \mathbb{E}_{d,t\sim p(D_L,T_L)} \log p_\theta(t|d) + \log p_\phi(d|t) \quad (5)$$

where (D_L, T_L) contains the k data annotations. Putting all together, we summarize it in Algorithm 1. In the training stage, we optimize the objectives of cycle consistency, representation matching and supervised learning sequentially to maintain a constant ratio of signals from all sides.

5 Experiment Setting

Data We conduct experiments on the E2E (Novikova et al., 2017) and WebNLG (Colin et al., 2016) datasets. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The inputs are dialogue acts consisting of three to eight slot-value pairs. WebNLG contains 25k instances describing entities belonging to fifteen distinct DBpedia categories. The inputs are up to seven RDF triples of the form *(subject, relation, object)*.

Configuration The model is implemented based on fairseq (Ott et al., 2019). We use 600-dimensional token embedding and Adam optimizer with initial learning rate at 0.0002. Batch size is kept at 48 with a dropout rate at 0.3. We employ beam search with size 3 for decoding and select models based on BLEU-4 scores on the development set. The score is averaged over 10 random initialization runs. In this work, the seq2seq models are built upon the long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997). For LSTM cells, both the encoder and decoder have 3 layers, amounting to 18M parameters for the seq2seq model (600-dimension and 1024 hidden units). Maximum sequence length is set as 100 for E2E and 200 for WebNLG (SPM-based). All encoder parameters are shared between data and text samples. All models were trained on 1 Nvidia V100 GPUs (32GB and CUDA Version 10.2) for 4k steps. The total batch size is around 48K tokens per GPU and we use the Adam optimizer ($\epsilon = 1e{-}6$, $\beta_2 = 0.98$) along with linear learning rate decay scheduling. The total number of updates is set to 8000 for all training and models are selected based on optimal validation BLEU4. At decoding time, sentences are generated using greedy decoding.

Algorithm 1 Few-shot Data-to-text Framework

```
Input: $D_U, (D_L, T_L)$
Create $(D_a, T_a)$ by information augmentation ;
$(D_L, T_L) \leftarrow (D_L, T_L) \cup (D_a, T_a)$;
Create $T'$ by LM augmentation ;
$T \leftarrow T_L \cup T'$;
repeat
    Sample batch data from $(D_L, T)$ ;
    Cycle consistency:
    Optimize by Eq. 1 + Eq. 2 + Eq. 3;
    Representation Matching:
    Optimize by Eq. 4;
    Supervised Training:
    Optimize by Eq. 5;
until convergence
```

6 Results and Analysis

In this section, we present experiment results and analysis. We first compare our model with other baselines on both datasets, then perform a set of ablation studies on the E2E dataset to see the effects of each component. Finally, we analyze how text augmentation helps improves the model, include example outputs and show the human evaluation results in the end.

Model	E2E - 10%				E2E - 100%			
	BLEU	NIST	METEOR	ROUGE-L	BLEU	NIST	METEOR	ROUGE-L
SLUG	-	-	-	-	66.19	8.61	44.54	67.72
Seq2seq	53.38	6.10	38.10	60.53	63.32	6.81	41.25	62.91
Qader et al. (2019)	58.10	6.24	41.32	62.84	64.20	7.14	44.68	65.31
Chen et al. (2020)	59.10	7.49	40.25	63.23	63.72	7.76	40.25	66.23
Proposed (LSTM)	64.24	7.71	43.53	66.81	68.88	8.89	48.53	72.12

	WebNLG - 10%				WebNLG - 100%			
Model	BLEU	NIST	METEOR	ROUGE-L	BLEU	NIST	METEOR	ROUGE-L
Melbourne	-	-	-	-	44.93	8.98	36.58	60.40
Seq2seq	36.54	7.3	35	54.61	44.60	8.49	38.23	59.67
Qader et al. (2019)	38.66	7.81	34.1	56.95	47.19	8.71	37.90	58.61
Chen et al. (2020)	39.40	7.84	37.25	56.23	46.15	8.52	39.1	58.5
Proposed (LSTM)	43.75	8.29	33.58	58.49	50.26	8.86	40.71	61.29

Table 1: Performance on E2E and WebNLG with 10% and 100% data. Qader et al. (2019) utilizes all ground-truth unpaired text samples while our proposed model only gets access to the few-shot data annotations.

Model/Share	**None**	**Enc**	**Dec**	**Both**
Supervised	53.20	-	-	-
+ $t \rightarrow d' \rightarrow t$	53.19	53.28	53.17	**53.29**
+ $d \rightarrow t' \rightarrow d$	53.15	**56.12**	53.49	56.07
+ $t \rightarrow t$	53.74	**56.39**	55.29	55.73
+ $d \rightarrow d$	53.37	**56.44**	56.09	56.11
+ Noise	54.13	**57.37**	56.59	57.04

Table 2: Ablation study for cycle consistency (10% annotations). BLEU-4 score is reported. Each line adds one condition on top of the previous one. **Supervised** is a supervised seq2seq baseline.

Comparison with Other Models In Table 1, we compare our model with (1) seq2seq baseline, (2) cycle consistency model as in Qader et al. (2019)[6] and (3) finetuned GPT-2 model as in Chen et al. (2020)[7]. For all models, we try running with 10% and 100% annotations to see how they perform under different data sizes. Our model is implemented both with LSTM encoder-decoders, same as the seq2seq baseline for a fair comparison. Note that Qader et al. (2019) further *utilized all the ground-truth unpaired text samples*, while the other models run only on the few-shot annotations. We also include the results of **SLUG** (Juraska et al., 2018) and **MELBOURNE** (Gardent et al., 2017), the overall winner on automatic metrics in the E2E and WebNLG challenge respectively(both seq2seq-based). SLUG uses a heuristic slot aligner based on a set of handcrafted rules and combines a complex pipeline of data augmentation, selection, model ensemble and reranker.

The results show that our proposed model significantly improves over the baseline on both the few-shot and fully supervised setting. The improvement is more evident when only 10% annotations are available, with a leap of 11 and 7 BLEU scores on E2E and WebNLG respectively. It also outperforms systems relying on task-dependent heuristics. In comparison, Qader et al. (2019), though with access to all text samples at all percentages, still underperforms our model with tangible margin. On the fully supervised setting, it brings little to no difference compared with the seq2seq baseline as no more extra data is incorporated in the training process. As such, we also observe that the text augmentation from finetuned GPT-2 model helps the proposed model on the few-shot setting, but its advantage also vanishes when all data annotations are available.

In Figure 4, we draw the model performance with varying number of data annotations. All models are trained from scratch with 10 different random initializations and the standard deviation of the BLEU-4 score is visualized. We can see our model (LSTM-based), though with a relatively larger standard deviation due to the uncertainty of text augmentation sampling, still consistently outperforms other baselines significantly and even surpasses the fully supervised seq2seq model with less than 10% of data annotations.

Ablation Study on Cycle Consistency In Table 2, we study how the four directions, input noise and parameter sharing affect the performance of cycle-consistency. The experiments are conducted with 10% annotations and *no further unpaired text*

---

[6] The author did not open-source their code. We reproduced their model based on our implementation. The results on 10k annotations matches their reports in the paper.

[7] https://github.com/czyssrs/Few-Shot-NLG

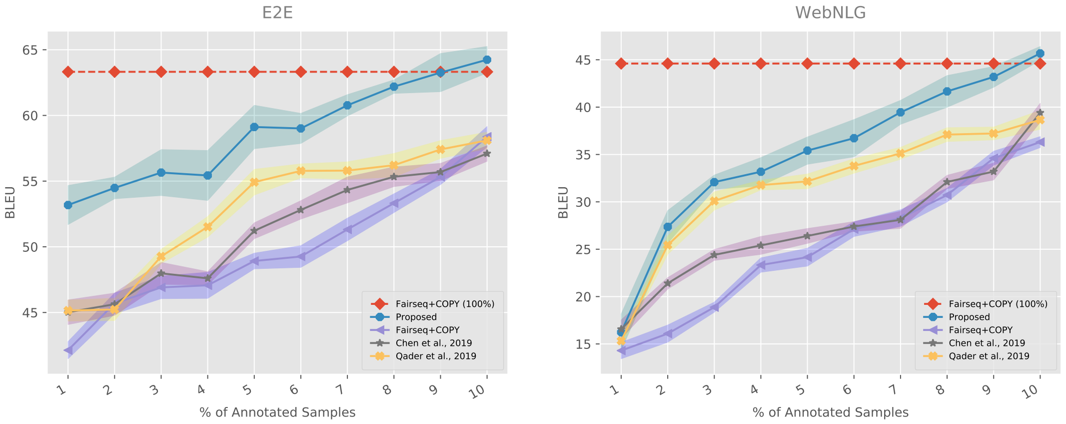


Figure 4: Model performance with varying number of data annotations. Our model with 10% annotations outperforms the seq2seq model trained on 100% pairs (dotted line) on both datasets. Shades are the sample standard deviation based on 10 runs of different model initializations.

samples are available.

As can be observed, adding the training direction $t \rightarrow d' \rightarrow t$ (i.e. back translation) has little effects on top of the supervised seq2seq baseline. This is expected since back translation is naturally designed to incorporate additional unpaired text samples. When run only on the few-shot annotations, its power is very limited. The backward direction $d \rightarrow t' \rightarrow d$ is surprisingly useful when the encoder is shared between the data and text. Though this direction will not affect the text decoder at all, the improvement suggests the model can benefit a lot by simply structuring its encoded space and mapping aligned data-text pairs to similar vector space. The autoencoding directions brings a little improvement. When combined with input noise, the performance further increases. This is similar to previous findings that denoising autoencoding is more helpful in inducing meaningful latent space (Lample et al., 2018) in comparison to simply learning to copy the original input.

The results also suggest encoder sharing is important for the cycle consistency objective to work in our few-shot setting. Decoder sharing, in contrast, makes little or even negative influence. This is kinda similar as in multilingual machine translation where sharing the decoder among languages might negatively interfere with the performance (Johnson et al., 2017).

Ablation Study on Text Augmentation On top of the four-direction cycle consistency training, we study the effects of text augmentation in Table 3. We compare our proposed info + LM augmentation with (1) random augmentation, where a random text from Wikipedia is sampled to the augmented text space, (2) unsupervised data augmentation (Xie et al., 2019) where text samples are augmented with paraphrases of current annotations and (3) ground-truth augmentation with reference obtained from the left training corpus, which can serve as an upper bound of text augmentation techniques. We test the performance with 1%, 5%, 10% and 20% annotations to see the effects with varying number of supervisions.

Text Augmentation	**1%**	**5%**	**10%**	**20%**
None	44.18	50.22	57.37	63.28
Random	41.30	49.62	57.71	62.79
UDA	44.24	50.09	57.66	61.30
Info	45.63	52.22	58.80	63.22
+ LM	48.67	53.53	59.04	**64.78**
Reference	**55.33**	**54.92**	**59.11**	64.26
Random (+RM)	42.32	50.10	58.53	63.29
UDA (+RM)	44.32	52.22	58.80	61.27
Info (+RM)	48.63	56.66	60.80	63.52
+ LM (+RM)	53.18	59.12	64.24	**65.38**
Reference (+RM)	**62.74**	**63.28**	**64.93**	65.27

Table 3: Ablation study for text augmentation with varying number of annotations. Experiments are performed on the E2E dataset. **LM** augmentation outperforms **Random** by a large margin, and even outperforming augmentation with ground-truth references on some occasions. Representation matching (**RM**) boosts the overall performance further.

Model	E2E			WebNLG		
	Fluency	Miss	Wrong	Fluency	Miss	Wrong
Seq2Seq	3.68	49	63	3.95	57	48
Cycle-only	4.08	46	66	4.23	48	44
Finetune GPT-2	4.21	43	57	4.10	39	45
Proposed (LSTM)	**4.33**	**39**	**44**	**4.27**	**31**	**39**

Table 4: Human Evaluation on the sampled outputs (100 instances) for models with 10% annotated data. Cycle-only indicates the approach in Qader et al. (2019); and Finetuned GPT-2 is refers to Chen et al. (2020).

As can be seen, the random augmentation even

Data:

[name] Blue Spice [eat type] restaurant [food] Chinese [area] city centre [family friendly] no [near] Rainbow Vegetarian Café

Reference: at the [city centre], there is a [restaurant] called the [Blue Spice].

seq2seq/info-aug: Blue Spice restaurant near Rainbow Vegetarian Café has a *5 star rating. prices start at £30.*

+LM-aug: located near Rainbow Vegetarian Café is a Chinese theme eatery and restaurant called Blue Spice. It is in the city centre area.

Figure 5: Generation examples with different text augmentation techniques. Trained on 5 data annotations (see the above toy training set). The attribute combination of input data is unseen in the 5 annotations. Hallucinated contents are *italitized*.

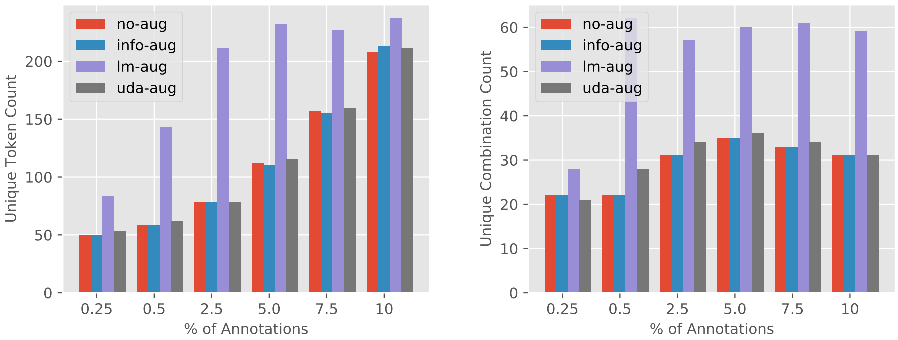


Figure 6: Additional number of decoded unique tokens (non-copied) and unique combinations of information on the testset with varying number of annotations.

harms the model performance, suggesting reasonable in-domain text augmentation are necessary for the model improvement. UDA augmentation also makes rather little difference as it simply paraphrases the current available annotations but cannot bring any new information. The information augmentation by slot-value replacement helps improve a bit. When combined with LM, the performance can be further boosted, especially for lower-resource scenarios. The representation matching always helps lift the performance, with gains of up to 10 BLEU points. As expected, the benefit from text augmentation gradually vanishes as more annotations are collected, especially for datasets with relatively simple patterns as E2E.

How text augmentation helps Intuitively the GPT-2 augmentation is expected to impose new tokens and combination patterns to the few-shot annotations. To investigate whether this is the case, for the decoded text in the test phase, we count the number of unique tokens (excluding copied data values) and unique information combination patterns (attribute combinations in E2E). The results in Fig. 6 show that LM-augmentation indeed greatly enriches the vocabulary space, even doubling the generated unique tokens in low-resource scenarios. The same happens for new combination patterns. In contrast, *info-aug* cannot insert new tokens or combinations at all since all it does is replacing data values based on the same text annotation. UDA can impose new tokens by paraphrasing the annotations, but it hardly helps the model generalize to new combinations of information. Moreover, when trained on a toy dataset, we observe from the generation outputs that Seq2seq and info-aug produce the wrong outputs and overfit to the information in the 5 training instances. With LM augmentation, it adapts to the new combination and connects information correctly. Figure 5 shows a generation example with different text augmentation techniques. We train the systems in a toy setting with only 5 data annotations (Trainset in the Appendix). We pick an input data with an unseen attribute combination to test if models can generalize correctly. Seq2seq and info-aug produce the wrong generation overfit to the information in the 5 training instances. With LM augmentation, it adapts to the new combination and connects information correctly.

Human Evaluation We further run a human evaluation on the model outputs to closely check the generation quality. We compared four types of models: the seq2seq baseline, seq2seq plus cycle-consistency as in Qader et al. (2019), finetuned GPT-2 as in Chen et al. (2020) and our proposed model. All models are LSTM-based apart from the finetuned GPT-2 one. We sample 100 data instances from the test set and apply all the models to generate corresponding text. The data and generated text are evaluated by 50 crowdworkers on Prolific[8]. For each data-text pair, the annotator is instructed to evaluate (1) if the text is fluent (score 0-5 with 5 being fully fluent), (2) if it misses information contained in the source data and (3) if it includes wrong information. The average fluency scores, count of information miss and wrong information are presented in Table 4. The scores are generally consistent with the automatic evaluation results, our proposed model outperforms other ones by a large margin, even though cycle-only can access all unpaired text and finetuned GPT-2 is significantly larger than our LSTM-based seq2seq. The generated text are more fluent, yet maintaining the information completeness and correctness to a large extent.

7 Conclusion

We study few-shot data-to-text generation with only limited annotated data. We propose text augmentation with slot-value replacement followed by GPT-2 generation. The augmented text, when combined with cycle consistency and representation matching, is shown to help the model to generalize to unseen new tokens and patterns of token combinations. With less than 10% annotations, it outperforms supervised seq2seq model trained on 100% annotations and is extensible enough to be combined with pretraining techniques.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 "Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Shubham Agarwal, Marc Dymetman, and Eric Gaussier. 2018. Char2char generation with reranking for the e2e nlg challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 451–456.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seqˆ3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 673–681.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Ernie Chang, Jeriah Caplinger, Alex Marin, Xiaoyu Shen, and Vera Demberg. 2020. Dart: A lightweight quality-suggestive data-to-text annotation tool. In *COLING 2020*, pages 12–17.

Ernie Chang, Vera Demberg, and Alex Marin. 2021a. Jointly improving language understanding and generation with quality-weighted weak supervision of automatic labeling. In *EACL 2021*.

Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. 2021b. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. In *EACL 2021*.

Zhiyu Chen, Harini Eavani, Yinyin Liu, and William Yang Wang. 2020. Few-shot nlg with pretrained language model. *ACL*.

Eric Chu and Peter Liu. 2019. Meansum: a neural model for unsupervised multi-document abstractive summarization. In *International Conference on Machine Learning*, pages 1223–1232.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Anna Currey, Antonio Valerio Miceli-Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156.

---

[8] `https://www.prolific.co/`

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3922–3929.

Xavier Garcia, Pierre Foret, Thibault Sellam, and Ankur P Parikh. 2020. A multilingual view of unsupervised machine translation. *arXiv preprint arXiv:2002.02955*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Dimitra Gkatzia. 2016. Content selection in data-to-text systems: A survey. *arXiv preprint arXiv:1610.08375*.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in neural information processing systems*, pages 820–828.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. *ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162.

Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1406–1415.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, et al. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Chris van der Lee, Emiel Krahmer, and Sander Wubben. 2018. Automated learning of templates for data-to-text generation: comparing rule-based, statistical and neural methods. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 35–45.

Shuming Ma, Pengcheng Yang, Tianyu Liu, Peng Li, Jie Zhou, and Xu Sun. 2019. Key fact as pivot: A two-stage model for low resource table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2047–2057.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53.

Yannis Papanikolaou and Andrea Pierleoni. 2020. Dare: Data augmented relation extraction with gpt-2. *arXiv preprint arXiv:2004.13845*.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876.

Raheel Qader, François Portet, and Cyril Labbé. 2019. Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of* *the 12th International Conference on Natural Language Generation*, pages 552–562.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Dana Ruiter, Cristina España-Bonet, and Josef van Genabith. 2019. Self-supervised neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1828–1834, Florence, Italy. Association for Computational Linguistics.

Martin Schmitt and Hinrich Schütze. 2019. Unsupervised text generation from structured data. *arXiv preprint arXiv:1904.09447*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 86–96.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2020. Towards unsupervised language understanding and generation by joint dual learning. *ACL*.

Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552*.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. {LAMAL}: {LA}nguage modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

Self-Training Pre-Trained Language Models for Zero- and Few-Shot Multi-Dialectal Arabic Sequence Labeling

Muhammad Khalifa [1], Muhammad Abdul-Mageed[2], Khaled Shaalan[3]
[1]Department of Computer Science, Cairo University
[2]Natural Language Processing Lab, University of British Columbia
[3]The British University in Dubai
m.khalifa@grad.fci-cu.edu.eg
muhammad.mageed@ubc.ca, khaled.shaalan@buid.ac.ae

Abstract

A sufficient amount of annotated data is usually required to fine-tune pre-trained language models for downstream tasks. Unfortunately, attaining labeled data can be costly, especially for multiple language varieties and dialects. We propose to self-train pre-trained language models in zero- and few-shot scenarios to improve performance on data-scarce varieties using only resources from data-rich ones. We demonstrate the utility of our approach in the context of Arabic sequence labeling by using a language model fine-tuned on Modern Standard Arabic (MSA) only to predict named entities (NE) and part-of-speech (POS) tags on several dialectal Arabic (DA) varieties. We show that self-training is indeed powerful, improving zero-shot MSA-to-DA transfer by as large as ~10% F_1 (NER) and 2% accuracy (POS tagging). We acquire even better performance in few-shot scenarios with limited amounts of labeled data. We conduct an ablation study and show that the performance boost observed directly results from training data augmentation possible with DA examples via self-training. This opens up opportunities for developing DA models exploiting only MSA resources. Our approach can also be extended to other languages and tasks.[1]

1 Introduction

Neural language models (Xu and Rudnicky, 2000; Bengio et al., 2003) with vectorized word representations (Mikolov et al., 2013) are currently core to a very wide variety of NLP tasks. In specific, using representations from transformer-based (Vaswani et al., 2017) language models (Devlin et al., 2018; Liu et al., 2019), pre-trained on large amounts of unlabeled data and then fine-tuned on labeled task-specific data, has become a popular approach for improving downstream task performance. This pre-training then fine-tuning scheme has been successfully applied to several tasks, including question answering (Yang et al., 2019), social meaning detection (Abdul-Mageed et al., 2020d), text classification (Liu et al., 2019), named entity recognition (NER), and part-of-speech (POS) tagging (Tsai et al., 2019; Conneau et al., 2019). The same setup also works well for cross-lingual learning (Lample and Conneau, 2019; Conneau et al., 2019).

Given that it is very expensive to glean labeled resources for all language varieties and dialects, a question arises: "How can we leverage resource-rich dialects to develop models nuanced to downstream tasks for resource-scarce ones?". In this work, we aim to answer this particular question by applying self-training to unlabeled target dialect data. We empirically show that self-training is indeed an effective strategy in *zero-shot* (where no gold dialectal data are included in training set, Section 4.2) and *few-shot* (where a given number of gold dialectal data points is included in training split, Section 4.4).

Our few-shot experiments reveal that self-training is always a useful strategy that *consistently* improves over mere fine-tuning, even when *all* dialect-specific gold data are used for fine-tuning. In order to understand why this is the case (i.e., why combining self-training with fine-tuning yields better results than mere fine-tuning), we perform an extensive error analysis based on our NER data. We discover that self-training helps the model most (% = 59.7) with improving false positives. This includes DA tokens whose MSA orthographic counterparts (Shaalan, 2014) are either named entities or trigger words that frequently co-occur with named entities in MSA. Interestingly, such out-of-MSA tokens occur in highly dialectal contexts (e.g., interjections and idiomatic expressions employed in interpersonal social media communication) or ones

---

[1] Our code and fine-tuned models can be accessed at: `https://github.com/mohammadKhalifa/zero-shot-arabic-dialects`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 769–782
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

where the social media context in which the language (DA) is employed affords more freedom of speech (Alshehri et al., 2020) and a platform for political satire. We present our error analysis in Section 5.

Context: Language use in social media tends to diverge from 'standard', offline norms (Danet and Herring, 2007; Herring et al., 2015). For example, users employ slang, emojis, abbreviations, letter repetitions, and other types of playful practices. This poses a challenge for processing social media data in general. However, there are other challenges specific to Arabic that motivate our work. More specifically, we choose Arabic to apply our approach since it affords a rich context of linguistic variation: In addition to the standard variety, MSA, Arabic also has several spoken dialects (Abdul-Mageed et al., 2018; Bouamor et al., 2019; Abdul-Mageed et al., 2020b,c), which differ significantly from the written MSA (Zaidan and Callison-Burch, 2014) thus offering an excellent context for studying our problem. Arabic dialects differ among themselves and from MSA at various linguistic levels: lexical, phonological, morphological, and syntactic. This makes our case much more challenging than that of standard vs. social media English, for example. For a good zero-shot performance in our case, a model is required to accommodate not only lexical distance between MSA and DA, but also differences in word formation and syntax (related to POS tags, for example) and lexical ambiguity (as the meaning of the same token can vary cross-dialectically). This makes the zero-shot setting even harder, where the performance drops 20% F1 points (See section 4.2).

From a geopolitical perspective, Arabic also has a strategic significance. This is a function of Arabic being the native tongue of 400 million speakers in 22 countries, spanning across two continents (Africa and Asia)[2]. In addition, the three dialects of our choice, namely Egyptian (EGY), Gulf (GLF), and Levantine (LEV), are popular dialects that are widely used online. This makes our resulting models highly useful in practical situations at scale. Pragmatically, ability to develop NLP systems on dialectal tasks with no-to-small labeled dialect data immediately eases a serious bottleneck. Arabic dialects differ among themselves and from MSA at all linguistic levels, posing challenges to traditional NLP approaches. We also note that our method is language-independent, and we hypothesize it can be directly applied to other varieties of Arabic or in other linguistic contexts for other languages and varieties.

Tasks: We apply our methods on two sequence labeling tasks, where we have access to both MSA and DA gold data. In particular, as mentioned above, we perform experiments on POS tagging and NER. Each of these tasks has become an integral part of various other NLP applications, including question answering, aspect-based sentiment analysis, machine translation, and summarization, and hence our developed models should have wide practical use. Again, we note that our approach itself is task-independent. The same approach can thus be applied to other tasks involving DA. We leave testing our approach on other languages, varieties, and tasks for future research.

Contributions: Our work offers the following contributions:

1. We study the problem of MSA-to-DA transfer in the context of sequence labeling and show that when training on MSA data only, a wide performance gap exists between testing on MSA and DA. That is, models fine-tuned on MSA generalize poorly to DA in zero-shot settings.
2. We propose self-training to improve zero- and few-shot MSA-to-DA transfer. Our approach requires little-to-no labeled DA data. We evaluate extensively on 3 different dialects, and show that our method indeed narrows the performance gap between MSA and DA by a margin as wide as $\sim$10% F_1 points.
3. We develop state-of-the-art models for the two sequence labeling tasks (NER and POS).

We now introduce our method.

2 Method

While the majority of labeled Arabic datasets are in MSA, most daily communication in the Arab world is carried out in DA. In this work, we show that models trained on MSA for NER and POS tagging generalize poorly to dialect inputs when used in zero-shot-settings (i.e., no dialect data used during training). Across the two tasks, we test how self-training would fare as an approach to leverage unlabeled DA data to improve performance

---

[2] https://www.internetworldstats.com/stats19.htm

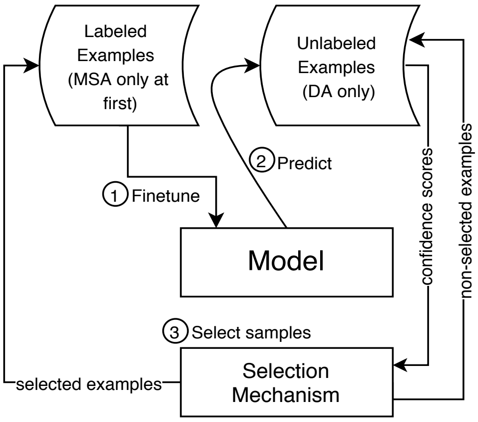


Figure 1: MSA-to-DA Self-training transfer.

```
Algorithm 1: MSA-to-DA Self-Training
Given set $L$ of labeled MSA examples, set $U$ of
  unlabeled DA examples, $\tau$ parameter for probability
  threshold selection.
repeat
    Fine-tune model $M$ for $K$ epochs on labeled
      MSA examples $L$;
    for $u_i \in U$ do
        Obtain prediction $p_{u_i}$ on unlabeled DA
          example $u_i$ using model $M$;
        if $\min_{(l_j^{(i)}, c_j^{(i)}) \in p_{u_i}} c_j^{(i)} \geq \tau$ then
            remove $u_i$ from $U$ and add it to $L$;
    end
until stopping criterion satisfied
```

on DA. As for self-training, it involves training a model using its own predictions on a set of unlabeled data identical from its original training split. Our proposed self-training procedure is given two sets of examples: a labeled set L and an unlabeled set U. To perform zero-shot MSA-to-DA transfer, MSA examples are used as the labeled set, while unlabeled DA examples are the unlabeled set. As shown in Figure 1, each iteration of the self-training algorithm consists mainly of three steps. First, a pre-trained language model is fine-tuned on the labeled MSA examples L. Second, for every unlabeled DA example u_i, we use the model to tag each of its tokens to obtain a set of predictions and confidence scores for each token $p_{u_i} = (l_1^{(i)}, c_1^{(i)}), (l_2^{(i)}, c_2^{(i)}), ...(l_{|u_i|}^{(i)}, c_{|u_i|}^{(i)})$, where $(l_j^{(i)}, c_j^{(i)})$ are the label and confidence score (softmax probability) for the j-th token in u_i. Third, we employ a selection mechanism to identify examples from U that are going to be added to L for the next iteration.

For a selection mechanism, we experiment with both a thresholding approach and a fixed-size (Dong and de Melo, 2019) approach. In the thresholding method, a threshold τ is applied on the minimum confidence per example. That is, we only add an example u_i to L if $\min_{(l_j^{(i)}, c_j^{(i)}) \in p_{u_i}} c_j^{(i)} \geq \tau$. See Algorithm 1. The fixed-size approach involves, at each iteration, the selection of the top S examples with respect to the minimum confidence score $\min_{(l_j^{(i)}, c_j^{(i)}) \in p_{u_i}} c_j^{(i)}$, where S is a hyper-parameter. We experiment with both approaches and report results in Section 4.

For our language model, we use XLM-RoBERTa (Conneau et al., 2019), XML-R for short. XLM-R is a cross-lingual model, and we choose it since it is reported to perform better than the multilingual mBERT (Devlin et al., 2018). XLM-R also uses Common Crawl for training, which is more likely to have dialectal data than the Arabic Wikipedia (used in mBERT), making it more suited to our work. We now introduce our experiments.

3 Experiments

We begin our experiments with evaluating the standard fine-tuning performance of XLM-R models on both NER and POS tagging against strong baselines. We then use our best models from this first round to investigate the MSA-to-DA zero-shot transfer, showing a significant performance drop even when using pre-trained XLM-R. Consequently, we employ self-training for both NER and POS tagging in zero- and few-shot settings, showing substantial performance improvements in both cases. We now introduce our datasets.

3.1 Datasets

NER: For our work on NER, we use 4 datasets: ANERCorp (Benajiba et al., 2007), ~150K tokens; ACE 2003 (Mitchell et al., 2003) BNews (BN-2003), ~15K tokens; ACE 2003 Newswire (NW-2003), ~27K tokens; and Twitter (Darwish, 2013), ~81K tokens. Named entity types in all datasets are *location (LOC)*, *organization (ORG)*, and *person (PER)*.

POS Tagging: There are a number of Arabic POS tagging datasets, mostly on MSA (Maamouri et al., 2004) but also on dialects such as EGY (Maamouri et al., 2014). To show that the proposed approach is able to work across multiple dialects, we ideally needed data from more than one dialect. Hence, we use the multi-dialectal dataset from (Darwish et al., 2018), comprising 350 tweets from each of the 4 varieties MSA, EGY, GLF and LEV. This dataset has 21 POS tags, some of which are suited to social media (since it is derived from Twitter). We show the POS tag set from (Darwish et al., 2018) in Table 10 in Appendix A. We now introduce our baselines.

3.2 Baselines

For the **NER task**, we use the following baselines:

- **NERA (Abdallah et al., 2012)**: A hybrid system of rule-based features and a decision tree classifier.
- **WC-BiLSTM (Gridach, 2016)**: A character- and a word-level Bi-LSTM with a conditional random fields (CRF) layer.
- **WC-CNN (Khalifa and Shaalan, 2019)**: A character- and a word-level CNN with a CRF layer.
- **mBERT (Devlin et al., 2018)**: A fine-tuned multilingual BERT-Base-Cased (110M parameters), pre-trained with a masked language modeling objective on the Wikipedia corpus of 104 languages (including Arabic). For fine-tuning, we find that (based on experiments on our development set) a learning rate of 6×10^{-5} works best with a dropout of 0.1.

In addition, we compare to the published results in (Shaalan and Oudah, 2014), AraBERT (Antoun et al., 2020), and CAMel (Obeid et al., 2020) for the ANERCorp dataset. We also compare to the published results in (Khalifa and Shaalan, 2019) for the 4 datasets.

For the **POS tagging task**, we compare to our own implementation of WC-BiLSTM (since there is no published research that uses this method on the task, as far as we know) and run mBERT on our data. We also compare to the CRF results published by (Darwish et al., 2018). In addition, for the Gulf dialect, we compare to the BiLSTM with compositional character representation and word representations (CC2W+W) published results in (Alharbi et al., 2018).

3.3 Experimental Setup

Our main models are XLM-RoBERTa base architecture XLM-R$_\text{B}$ $(L = 12, H = 768, A = 12,$ 270M params) and XLM-RoBERTa large architecture XLM-R$_\text{L}$ $(L = 24, H = 1024, A = 16,$ 550M params), where L is number of layers, H is the hidden size, A is the number of self-attention heads. For XLM-R experiments, we use Adam optimizer with $1e^{-5}$ learning rate, batch size of 16. We typically fine-tune for 20 epochs, keeping the best model on the development set for testing. We report results on the test split for each dataset, across the two tasks. For all BiLSTM experiments, we use the same hyper-parameters as (Khalifa and Shaalan, 2019).

For the standard fine-tuning experiments, we use the same train/development/test split as in (Khalifa and Shaalan, 2019) for NER, and the same split provided by (Darwish et al., 2018) for POS tagging. For all the self-training experiments, we use the dialect subset of the Arabic online news commentary (AOC) dataset (Zaidan and Callison-Burch, 2011), comprising the EGY, GLF, and LEV varieties limiting to equal sizes of 9K examples per dialect (total =27K) [3]. We use the split from (Elaraby and Abdul-Mageed, 2018) of AOC, removing the dialect labels and just using the comments themselves for our self-training. Each iteration involved fine-tuning the model for $K = 5$ epochs. As a stopping criterion, we use early stopping with patience of 10 epochs. Other hyper-parameters are set as listed before. For selecting confident samples, we experiment with a fixed number of top samples $S = [50, 100, 200]$ and selection based on a probability threshold $\tau = [0.80, 0.90, 0.95]$ (softmax values) [4]. For all evaluations, we use the *seqeval* toolkit.[5]

4 Results

4.1 Fine-tuning XLM-R

Here, We show the resuts of standard fine-tuning of XLM-R for the two tasks in question. We start by showing the result of fine-tuning XLM-R on the **named entity task**, on each of the 4 Arabic NER (ANER) datasets listed in Section 3.1. Table 1 shows the test set macro F$_1$ score on each of the 4 ANER datasets. Clearly, the fine-tuned XLM-R models outperform other baselines on all datasets,

---

[3] We note that our approach could be scaled with an even bigger unlabeled dataset, given the performance gains we report with self-training in this work.

[4] It is worth noting that our S values are similar to those used in (Dong and de Melo, 2019). We also experimented with other values for τ and S, but found them sub-optimal and hence we report performance only for the listed values of these two hyper-parameters here.

[5] `https://github.com/chakki-works/seqeval`.

except on the NW-2003 where WC-CNN (Khalifa and Shaalan, 2019) performs slightly better than XLM-R_L.

For **POS Tagging**, Table 2 shows test set word accuracy of the XLM-R models compared to baselines. Again, XLM-R models (both base and large) outperform all other models. A question arises why XLM-R models outperform both mBERT and AraBERT. As noted before, for XLM-R vs. mBERT, XLM-R was trained on much larger data: CommonCrawl for XLM-R vs. Wikipedia for mBERT. Hence, the *larger dataset* of XLM-R is giving it an advantage over mBERT. For comparison with AraBERT, although the pre-training data for XLM-R and AraBERT may be comparable, even the smaller XLM-R model (XLM-R_B) has more than twice the number of parameters of the BERT_BASE architecture on which AraBERT and mBERT are built (270M v. 110M). Hence, XLM-R model *capacity* gives it another advantage. We now report our experiments with zero-shot transfer from MSA to DA.

4.2 MSA-DA Zero-Shot Transfer

We start by the discussion of **NER experiments**. Since there is no publicly available purely dialectal NER dataset on which we can study MSA-to-DA transfer, we needed to find DA data to evaluate on. We observed that the dataset from (Darwish, 2013) contains both MSA and DA examples (tweets). Hence, we train a binary classifier[6] to distinguish DA data from MSA. We then extract examples that are labeled with probability $p > 0.90$ as either DA or MSA. We obtain 2,027 MSA examples (*henceforth*, `Twitter-MSA`) and 1,695 DA examples (*henceforth*, `Twitter-DA`), respectively. We split these into development and test sets with 30% and 70% ratios. As **for POS Tagging**, we already have the three previously used DA datasets, namely EGY, GLF and LEV. We use those for the zero-shot setting by omitting their training sets and using only the development and test sets.

We first study how well models trained for NER and POS tagging on MSA data only will generalize to DA inputs during test time. We evaluate this zero-shot performance on both the XLM-R_B and XLM-R_L models. **For NER**, we train on ANERCorp (which is pure MSA) and evaluate on both Twitter-MSA and Twitter-DA. While for POS tagging, we train on the MSA subset (Darwish et al., 2018) and evaluate on the corresponding test set for each dialect. As shown in Table 3, for NER, a significant generalization gap of around 20 % F_1 points exists between evaluation on MSA and DA using both models. While for **POS tagging**, the gap is as large as 18.13 % accuracy for the LEV dialect with XLM-R_B. The smallest generalization gap is on the GLF variety, which is perhaps due to the high overlap between GLF and MSA (Alharbi et al., 2018). In the next section, we evaluate the ability of self-training to close this MSA-DA performance gap.

4.3 Zero-shot Self-Training

Here, **for NER**, similar to Section 4.2, we train on ANERCorp (pure MSA) and evaluate on Twitter-MSA and Twitter-DA. Table 4 shows self-training NER results employing the selection mechanisms listed in Section 2, and with different values for S and τ. The best improvement is achieved with the thresholding selection mechanism with a $\tau = 0.90$, where we have an F_1 gain of 10.03 points. More generally, self-training improves zero-shot performance in all cases albeit with different F_1 gains. It is noteworthy, however, that the much higher-capacity large model deteriorates on MSA if self-trained (dropping from 68.32% to 67.21%). This shows the ability of the large model to learn representations very specific to DA when self-trained. It is also interesting to see that the best self-trained base model achieved 50.10% F_1, outperforming the large model before the latter is self-trained (47.35% in the zero-shot setting). As such, we conclude that ***a base self-trained model, with less computational capacity, can (and in our case does) improve over a large (not-self-trained) model*** *that needs significant computation*. The fact that, when self-trained, the large model improves 15.35% points over the base model in the zero-shot setting (55.42 vs. 40.07) is remarkable.

As **for POS tagging**, we similarly observe consistent improvements in zero-shot transfer with self-training (Table 5). The best model achieves accuracy gains of 2.41% (EGY), 1.41% (GLF), and 1.74% (LEV). Again, this demonstrates the utility of self-training pre-trained language models on the POS tagging task even in absence of labeled dialectal POS data (zero-shot).

---

[6] The model we use is XLM-R_B fine-tuned on the AOC using (Elaraby and Abdul-Mageed, 2018) split. We achieve development and test accuracies of 90.3% and 89.4 %, respectively, outperforming the best results in (Elaraby and Abdul-Mageed, 2018).

Model	ANERCorp	BN-2003	NW-2003	Twitter
NERA (Abdallah et al., 2012)	88.77	–	–	–
CAMeL (Obeid et al., 2020)	85.00	–	–	–
Hybrid (Shaalan and Oudah, 2014)	90.66	–	–	–
WC-BiLSTM (Gridach, 2016)	88.56	94.92	90.32	64.93
WC-CNN (Khalifa and Shaalan, 2019)	88.77	94.12	**91.20**	65.34
mBERT (ours)	85.86	89.52	87.19	58.92
AraBERT (Antoun et al., 2020)	84.2	–	–	–
XLM-R_B (ours)	87.75	95.35	85.25	60.39
XLM-R_L (ours)	**91.43**	**97.33**	91.10	**68.91**

Table 1: Test set macro F_1 scores for NER.

Model	MSA	EGY	GLF	LEV
BiLSTM (CC2W + W) (Alharbi et al., 2018)	–	–	89.7	–
CRF (Darwish et al., 2018)	93.6	92.9	87.8	87.9
WC-BiLSTM (ours)	94.63	93.41	88.79	86.13
mBERT (ours)	90.57	92.88	87.85	72.30
XLM-R_B (ours)	96.30	94.70	92.18	89.98
XLM-R_L (ours)	**98.21**	**97.00**	**94.41**	**93.19**

Table 2: Test set accuracy for POS Tagging.

4.4 Few-Shot Self-Training

We also investigate whether self-training would be helpful in scenarios where we have access to some gold-labeled DA data (as is the case with POS tagging). Here, we evaluate the few-shot performance of self-training as increasing amounts of predicted DA data are added to the gold training set. This iteration of experiments focuses exclusively on **POS tagging**, using a fixed-size $S = 100$ of predicted cases for self-training and the XLM-R base model. Figure 2 shows how POS tagging test accuracy improves as the percentage of gold DA examples added to the MSA training data increases from 0% to 100% on the three dialects (EGY, GLF, and LEV). Comparing these results to those acquired via the standard fine-tuning settings without self-training, we find that self-training does *consistently* improve over fine-tuning. This improvement margin is largest with only 20% of the gold examples.

4.5 Ablation Study

Here, we conduct an ablation study with the NER task as our playground in order to verify our hypothesis that the performance boost primarily comes from using unlabeled DA data for self-training. By using a MSA dataset with the same size as our unlabeled DA one[7], we can compare the performance of the self-trained model in both settings: MSA and DA unlabeled data. We run 3 different self-training experiments using 3 different values for τ using each type of unlabeled data. Results are shown in table 6. While we find slight performance boost due to self-training even with MSA unlabeled data, the average F1 score with unlabeled DA is better by 2.67 points, showing that using unlabeled DA data for self-training has helped the model adapt to DA data during testing.

5 Error analysis

To understand why self-training the pre-trained language model, when combined with fine-tuning, improves over mere fine-tuning, we perform an error analysis. For the error analysis, we focus on the NER task where we observe a huge self-training gain. We use the development set of Twitter-DA (See section 4.3) for the error analysis. We compare predictions of the standard fine-tuned XLM-R_B model (FT) and the best performing self-training ($\tau = 0.9$) model (ST) on the data, and provide the confusion matrices of both models with gold labels in Table 11 (in Appendix B). The error analysis leads to an interesting discovery: The greatest benefit from the ST model comes mostly from reducing *false positives* (see Table 7). In other words, self-training helps regularize the model predictions such that tokens misclassified by the original FT model as a named entities are now correctly tagged as *unnamed entity* "O".

To understand why the ST model improves false positive rate, we manually inspect the cases it cor-

---

[7] We use a set of MSA tweets from the AOC dataset mentioned before.

Model	NER		POS			
	MSA	DA	MSA	EGY	GLF	LEV
XLM-R$_B$	60.42	40.07	96.30	78.38	83.72	78.17
XLM-R$_L$	68.32	47.35	98.21	82.28	85.95	81.24

Table 3: Zero-shot transfer results on DA For NER (macro F$_1$) and POS Tagging (accuracy). Models are trained on MSA only and evaluated on DA. Datasets used are: Twitter-MSA and Twitter-DA (Darwish, 2013) for NER, and Multi-dialectal (Darwish et al., 2018) for POS tagging.

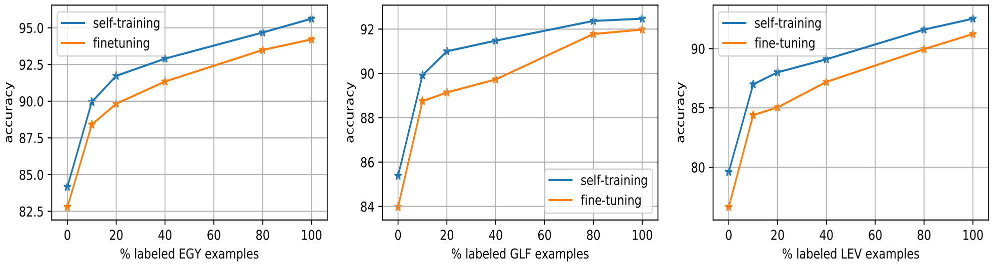


Figure 2: Development accuracy as labeled DA data is added to the training MSA data. **Left:** Results on EGY. **Middle:** Results on GLF. **Right:** Results on LEV. We use fixed-size selection with $S = 100$ for self-training models.

Model	MSA	DA
XLM-R$_B$	61.88	40.07
XLM-R$_B$, ST, S=50	60.98	43.88
XLM-R$_B$, ST, S=100	61.13	42.01
XLM-R$_B$, ST, S=200	61.46	43.49
XLM-R$_B$, ST, $\tau = 0.80$	**63.36**	46.97
XLM-R$_B$, ST, $\tau = 0.90$	61.02	**50.10**
XLM-R$_B$, ST, $\tau = 0.95$	62.25	47.91
XLM-R$_L$	**68.32**	47.35
XLM-R$_L$ + ST, $\tau = 0.90$	67.21	**55.42**

Table 4: Zero-short self-training (ST) NER results. Models trained on ANERCorp (pure MSA) and evaluated on Twitter-MSA and Twitter-DA we extract from (Darwish et al., 2018). Self-training boosts the performance on DA data by 10% macro F1 points with XLM-R$_B$ and $\tau = 0.90$.

Model	MSA	EGY	GLF	LEV
XLM-R$_B$	96.30	78.38	83.72	78.17
XLM-R$_B$, ST, S=50	–	**80.79**	**85.13**	**79.91**
XLM-R$_B$, ST, S=100	–	80.43	84.74	79.16
XLM-R$_B$, ST, S=200	–	78.75	84.21	79.40
XLM-R$_B$, ST, τ=0.90	–	79.52	83.97	79.21
XLM-R$_B$, ST, τ=0.85	–	78.97	83.53	79.06
XLM-R$_B$, ST, τ=0.80	–	78.88	83.72	78.50
XLM-R$_L$	98.21	82.28	85.95	81.24
XLM-R$_L$, ST, S=50	–	**82.65**	**87.76**	**83.70**

Table 5: Zero-shot POS tagging transfer accuracy when training on MSA only. **ST:** self-training.

rectly identifies that were misclassified by the FT model. We show examples of these cases in Table 8. As the table shows, the ST model is able to identify dialectal tokens whose equivalent MSA forms can act as trigger words (usually followed by a PER named entity). We refer to this category as ***false trigger words***. An example is the word نبي "prophet" (row 1 in Table 8). A similar example that falls within this category is in row (2), where the model is confused by the token الى ("who" in EGY, but "to" in MSA and hence the wrong prediction as LOC). A second category of errors is caused by ***non-standard social media language***, such as use of letter repetition in interjections (e.g., in row (3) in Table 8). In these cases, the FT model also assigns the class PER, but the ST model correctly identifies the tag as "O". A third class of errors arises as a result of ***out-of-MSA*** vocabulary. For example, the words in rows (4-6) are all out-of-MSA where the FT model, not knowing these, assigns the most frequent named entity label in train (PER). A fourth category of errors occurs as a result of a token that is usually part of a named entity in MSA, that otherwise functions as part of an ***idiomatic expression*** in DA. Row (7) in Table 8 illustrates this case. Table 12 in Appendix B provides more examples.

We also investigate errors shared by both the FT and ST models (errors which the ST model also could not fix). Some of these errors result from the fact that often times both MSA and DA use the same word for both person and location names.

Setting	Unlabeled MSA	Unlabeled DA
ST, $\tau = 0.80$	43.88	44.46
ST, $\tau = 0.90$	44.69	47.83
ST, $\tau = 0.95$	43.43	46.87
Avg	43.67	**46.34**

Table 6: Ablation experiment with MSA unlabeled data for zero-shot NER. Development set macro F1 is shown when using both unlabeled MSA and DA data with the same size. Average performance with DA unlabeled data is higher showing the effect of unlabeled DA on model final performance.

Measure	FT	ST	% improvement
True Positives	155	165	6.5 %
False Positive	159	64	**59.7 %**
False Negative	162	168	-3.7 %
True Negative	5,940	6,035	1.5 %

Table 7: Comparison of error categories in percentage between the fine-tuned model (FT) and the model combining fine-tuned+self-trained (ST) model, based on the dialectal part of the dev set of the NER task.

Row (1) in Table 13 (in Appendix B) is an example where the word "Mubarak", name of the ex-Egypt President, is used as LOC. Other errors include *out-of-MSA* tokens mistaken as named entities. An example is in row (3) in Table 13, where بأمارة ,("proof" or "basis" in EGY) is confused for بإمارة ("emirate", which is a location). *False trigger words*, mentioned before, also play a role here. An example is in row (7) where يابطل is confused for PER due to the trigger word يا "Hey!" that is usually followed by a person name. ***Spelling mistakes*** cause the third source of errors, as in row (4). We also note that even with self-training, detecting ORG entities is more challenging than PER or LOC. The problem becomes harder when such organizations are not seen in training such as in rows (8) الاخوان المسلمين, (9) قناة العربية and (10) المجلس العسكري, all of which do not occur in the training set (ANERCorp).

False negatives. The "regularizing" effect caused by self-training we discussed thus far can sometimes produce *false negatives* as shown in Table 9. We see a number of named entities that were misclassified by the self-trained model as unnamed ones. As an example, we take the last name الجنزوري which was classified both correctly and incorrectly in different contexts by the self-trained model. Context of correct classification is "هاش تاج لكمال الجنزوري", while it is "ماسك على الناس كلها سي دي الا الجنزوري ماسك عليه فلوبي" for the incorrect classification. First, we note that الجنزوري is not a common name (zero occurrences in the MSA training set). Second, we observe that in the correct case, the word was preceded by the first name كمال which was correctly classified as PER, making it easier for the model to assign PER to the word afterwards as a surname.

6 Related Work

Sequence Labeling. Recent work on sequence labeling usually involves using a word- or character-level neural network with a CRF layer (Lample et al., 2016; Ma and Hovy, 2016). These architectures have also been applied to Arabic sequence tagging (Gridach, 2016; Alharbi et al., 2018; Khalifa and Shaalan, 2019; Al-Smadi et al., 2020; El Bazi and Laachfoubi, 2019), producing better or comparable results to classical rule-based approaches (Shaalan and Oudah, 2014). We refer the interested reader to (Shoufan and Alameri, 2015) and (Al-Ayyoub et al., 2018) for surveys on Arabic NLP.

Pre-trained Language Models. Language models, based on Transformers (Vaswani et al., 2017), and pre-trained with the masked language modeling (MLM) objective have seen wide use in various NLP tasks. Examples include BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), MASS (Song et al., 2019), and ELECTRA (Clark et al., 2020). While they have been applied to several tasks, including text classification, question answering, named entity recognition (Conneau et al., 2019), and POS tagging (Tsai et al., 2019), a sufficiently large amount of labeled data is required for good performance. Concurrent with our work, Abdul-Mageed et al. (2020a) released MARBERT, a language model trained on a large amount of dialectal Arabic data. However, the extent to which dialect-specific models such as MARBERT can alleviate lack of labeled data remains untested.

Cross-lingual Learning. Cross-lingual learning is of particular importance due to the scarcity of labeled resources in many of the world's languages. The goal is to leverage existing labeled resources in high-resource languages (such as English) to optimize learning for low-resource ones. In our case, we leverage MSA resources for building DA models. With proximity to our work, Kim et al. (2017) trained a POS tagger for different languages

no.	Token	Eng.	MSA	Context/Explanation	FT Pred.
(1)	نبي	we want	نريد	نبي نعرف من... (*we want to know who*)	PER
(2)	ماكانوا	wasn't	لم يكونوا	أغلب الي ماكانوا مصدقين (*most of those who wasn't believing*)	LOC
(3)	لوووول	LOL	ضحك	لوووول... (interjection)	PER
(4)	عشان	for	لكي	تبي بطاريات عشان تلعب (*she wants batteries to play*)	LOC
(5)	دلوقتي	now	الآن	اقنعوه ينزل دلوقتي.. (*convince him to move now*)	PER
(6)	ايش	what	ماذا	ايش رأيك (*what do you think?*)	PER
(7)	قادر	capable	قادر	وبقدرة قادر... (*magically*; idiomatic expression)	PER

Table 8: Sample false positives mitigated by self-training. These were correctly predicted as the unnamed entity "O" by the self-trained model.

no.	Word	Gold	FT	ST
(1)	الاخوان	ORG	ORG	O
(2)	للبرادعي	PER	PER	O
(3)	مجدي الجلاد	PER	PER	O
(4)	فان ديزل	PER	PER	O
(5)	الجنزوري	PER	PER	O
(6)	زين يسون	PER	PER	O

Table 9: **NER task.** Sample false negatives produced by self-training.

using English-resources only using two BiLSTM networks to learn common and language-specific features. Xie et al. (2018) made use of bilingual word embeddings with self-attention to learn cross-lingual NER for low-resource languages.

Multilingual extensions of LMs have emerged through joint pre-training on multiple languages. Examples include mBERT (Devlin et al., 2018), XLM (Lample and Conneau, 2019) and XLM-RoBERTa (Conneau et al., 2019). Such multilingual models have become useful for few-shot and zero-shot cross-lingual settings, where there is little or no access to labeled data in the target language. For instance Conneau et al. (2019) evaluated a cross-lingual version of RoBERTa (Liu et al., 2019), namely XLM-R, on cross-lingual learning across different tasks such as question answering, text classification, and named entity recognition.

Self-Training. Self-Training is a semi-supervised technique to improve learning using unlabeled data. Self-training has been successfully applied to NER (Kozareva et al., 2005), POS tagging (Wang et al., 2007), parsing (Sagae, 2010) and text classification (Van Asch and Daelemans, 2016). Self-training has also been applied in cross-lingual settings when gold labels are rare in the target language. Hajmohammadi et al. (2015) proposed a combination of active learning and self-training for cross-lingual sentiment classification. Pan et al. (2017) made use of self-training for named entity tagging and linking across 282 different languages. Lastly, Dong and de Melo (2019) employed self-training to improve zero-shot cross-lingual classification with mBERT (Devlin et al., 2018).

7 Conclusion

Even though pre-trained language models have improved many NLP tasks, they still need labeled data for fine-tuning. We show how self-training can boost the performance of pre-trained language models in zero- and few-shot settings on various Arabic varieties. We apply our approach to two sequence labeling tasks (NER and POS), establishing new state-of-the-art results on both. Through in-depth error analysis and an ablation study, we uncover why our models work and where they can fail. Our method is *language*- and *task-agnostic*, and we believe it can be applied to other tasks and language settings. We intend to test this claim in future research. Our research also has bearings to ongoing work on language models and self-training, and interactions between these two areas can be the basis of future work. All our models and code are publicly available.

Acknowledgements

MAM gratefully acknowledges support from the Natural Sciences and Engineering Research Council of Canada, the Social Sciences and Humanities Research Council of Canada, Canadian Foundation for Innovation, Compute Canada (`www.computecanada.ca`) and UBC ARC-Sockeye (`https://doi.org/10.14288/SOCKEYE`).

References

Sherief Abdallah, Khaled Shaalan, and Muhammad Shoaib. 2012. Integrating rule-based system with classification for Arabic named entity recognition. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 311–322. Springer.

Muhammad Abdul-Mageed, Hassan Alhuzali, and Mohamed Elaraby. 2018. You tweet what you speak: A city-level dataset of arabic dialects. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2020a. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. *arXiv preprint arXiv:2101.01785*.

Muhammad Abdul-Mageed, Chiyu Zhang, Houda Bouamor, and Nizar Habash. 2020b. Nadi 2020: The first nuanced arabic dialect identification shared task. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop (WANLP 2020)*, pages 97–110, Barcelona, Spain.

Muhammad Abdul-Mageed, Chiyu Zhang, AbdelRahim Elmadany, and Lyle Ungar. 2020c. Micro-dialect identification in diaglossic and code-switched environments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5855–5876.

Muhammad Abdul-Mageed, Chiyu Zhang, Azadeh Hashemi, and El Moatez Billah Nagoudi. 2020d. AraNet: A Deep Learning Toolkit for Arabic Social Media. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 16–23.

Mahmoud Al-Ayyoub, Aya Nuseir, Kholoud Alsmearat, Yaser Jararweh, and Brij Gupta. 2018. Deep learning for arabic nlp: A survey. *Journal of computational science*, 26:522–531.

Mohammad Al-Smadi, Saad Al-Zboon, Yaser Jararweh, and Patrick Juola. 2020. Transfer learning for Arabic named entity recognition with deep neural networks. *IEEE Access*, 8:37736–37745.

Randah Alharbi, Walid Magdy, Kareem Darwish, Ahmed AbdelAli, and Hamdy Mubarak. 2018. Part-of-speech tagging for Arabic gulf dialect using bi-lstm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Ali Alshehri, El Moatez Billah Nagoudi, and Muhammad Abdul-Mageed. 2020. Understanding and detecting dangerous speech in social media. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 40–47, Marseille, France. European Language Resource Association.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for Arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Yassine Benajiba, Paolo Rosso, and José Miguel Bened\'\iruiz. 2007. Anersys: An Arabic named entity recognition system based on maximum entropy. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 143–153.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Houda Bouamor, Sabit Hassan, and Nizar Habash. 2019. The MADAR shared task on Arabic fine-grained dialect identification. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 199–207.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Brenda Danet and Susan C Herring. 2007. *The multilingual Internet: Language, culture, and communication online*. Oxford University Press.

Kareem Darwish. 2013. Named entity recognition using cross-lingual resources: Arabic as an example. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1558–1567.

Kareem Darwish, Hamdy Mubarak, Ahmed Abdelali, Mohamed Eldesouki, Younes Samih, Randah Alharbi, Mohammed Attia, Walid Magdy, and Laura Kallmeyer. 2018. Multi-dialect arabic pos tagging: a crf approach. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xin Luna Dong and Gerard de Melo. 2019. A robust self-learning framework for cross-lingual text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* *and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6307–6311.

Ismail El Bazi and Nabil Laachfoubi. 2019. Arabic named entity recognition using deep learning approach. *International Journal of Electrical & Computer Engineering (2088-8708)*, 9(3).

Mohamed Elaraby and Muhammad Abdul-Mageed. 2018. Deep models for Arabic dialect identification on benchmarked data. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 263–274.

Mourad Gridach. 2016. Character-aware neural networks for Arabic named entity recognition for social media. In *Proceedings of the 6th workshop on South and Southeast Asian natural language processing (WSSANLP2016)*, pages 23–32.

Mohammad Sadegh Hajmohammadi, Roliana Ibrahim, Ali Selamat, and Hamido Fujita. 2015. Combination of active learning and self-training for cross-lingual sentiment classification with density analysis of unlabelled samples. *Information sciences*, 317:67–77.

Susan C Herring, Jannis Androutsopoulos, et al. 2015. Computer-mediated discourse 2.0. *The handbook of discourse analysis*, 2:127–151.

Muhammad Khalifa and Khaled Shaalan. 2019. Character convolutions for arabic named entity recognition with long short-term memory networks. *Computer Speech & Language*, 58:335–346.

Joo-Kyung Kim, Young-Bum Kim, Ruhi Sarikaya, and Eric Fosler-Lussier. 2017. Cross-lingual transfer learning for pos tagging without cross-lingual resources. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2832–2838.

Zornitsa Kozareva, Boyan Bonev, and Andres Montoyo. 2005. Self-training and co-training applied to spanish named entity recognition. In *Mexican International conference on Artificial Intelligence*, pages 770–779. Springer.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The penn Arabic treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467. Cairo.

Mohamed Maamouri, Ann Bies, Seth Kulick, Michael Ciul, Nizar Habash, and Ramy Eskander. 2014. Developing an egyptian Arabic treebank: Impact of dialectal morphology on annotation and tool development. In *LREC*, pages 2348–2354.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Alexis Mitchell, Stephanie Strassel, Mark Przybocki, J Davis, George Doddington, Ralph Grishman, Adam Meyers, Ada Brunstain, Lisa Ferro, and Beth Sundheim. 2003. Tides extraction (ace) 2003 multilingual training data. *LDC2004T09, Philadelphia, Penn.: Linguistic Data Consortium*.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 7022–7032.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958.

Kenji Sagae. 2010. Self-training without reranking for parser domain adaptation and its impact on semantic role labeling. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 37–44.

Khaled Shaalan. 2014. A survey of Arabic named entity recognition and classification. *Computational Linguistics*, 40(2):469–510.

Khaled Shaalan and Mai Oudah. 2014. A hybrid approach to Arabic named entity recognition. *Journal of Information Science*, 40(1):67–87.

Abdulhadi Shoufan and Sumaya Alameri. 2015. Natural language processing for dialectical arabic: A survey. In *Proceedings of the second workshop on Arabic natural language processing*, pages 36–48.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. 2019. Small and practical bert models for sequence labeling. *arXiv preprint arXiv:1909.00100*.

Vincent Van Asch and Walter Daelemans. 2016. Predicting the effectiveness of self-training: Application to sentiment classification. *arXiv preprint arXiv:1601.03288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wen Wang, Zhongqiang Huang, and Mary Harper. 2007. Semi-supervised learning for part-of-speech tagging of mandarin transcribed speech. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–137. IEEE.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. *arXiv preprint arXiv:1808.09861*.

Wei Xu and Alex Rudnicky. 2000. Can artificial neural networks learn language models? In *Sixth international conference on spoken language processing*.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357.

Omar Zaidan and Chris Callison-Burch. 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 37–41.

Omar F Zaidan and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202.

Appendices

A POS Tag Set

Table 10 lists all the part-of-speech (POS) tags used in our experiments.

Tag	Description	Tag	Description
ADV	adverb	ADJ	adjective
CONJ	conjunction	DET	determiner
NOUN	noun	NSUFF	noun suffix
NUM	number	PART	particle
PUNC	punctuation	PRON	pronoun
PREP	preposition	V	verb
ABBREV	abbreviation	VSUFF	verb suffix
FOREIGN	non-Arabic	FUT_PART	future particle
PROG_PART	progressive particle	EMOT	Emoticon/Emoji
MENTION	twitter mention	HASH	Hashtag
URL	URL	–	–

Table 10: The POS tag set in (Darwish et al., 2018).

B Error Analysis

		Predicted			
		PER	LOC	ORG	O
	PER	117	2	2	66
Gold	LOC	11	33	1	39
	ORG	5	5	5	57
	O	130	14	15	5,940

		Predicted			
		PER	LOC	ORG	O
	PER	120	3	2	62
Gold	LOC	10	34	0	40
	ORG	5	6	11	66
	O	54	8	2	6,035

Table 11: NER confusion matrices for fine-tuning (left) and self-training (right) on the development set of the DA NER data.

no.	Token	Eng.	MSA	Context/Explanation	FT Pred.
(1)	نبي	we want	نريد	نبي نعرف من… (*we want to know who*)	PER
(2)	ماكانوا	wasn't	لم يكونوا	أغلب الي ماكانوا مصدقين (*most of those who wasn't believing*)	LOC
(3)	لووول	LOL	ضحك	لووول… (interjection)	PER
(4)	عشان	for	لكي	تبي بطاريات عشان تلعب (*she wants batteries to play*)	LOC
(5)	دلوقتي	now	الآن	..اقنعوه ينزل دلوقتي (*convince him to move now*)	PER
(6)	ايش	what	ماذا	ايش رأيك (*what do you think?*)	PER
(7)	قادر	capable	قادر	وبقدرة قادر… (*magically*; idiomatic expression)	PER
(8)	المشين	shameful	المشين	المشين طنطاوي (*shameful Tantawy*; Playful for *General Tant.*)	PER
(9)	ايديكوا	your hands	أيديكم	ابوس ايديكوا اقنعوه… (*I entreat you to convince hi*m)	PER
(10)	اسالك	I ask you	أسألك	ودي اسالك شنهي (*I ask you what*)	ORG
(11)	مين	who	مين	صوتك مع مين البدوي (*who do you vote for, Badawi*)	PER
(12)	فلوبي ديسك	floppy disk	قرص مرن	ماسك عليه فلوبي ديسك (*holds a floppy disk against him*)	PER
(13)	لحبايب	loved ones	الأحباء	تعال علم يف لحبايب (*come teach your loved ones*)	LOC
(14)	ماي	water	ماء	جبت لهم ماي (*brought them water*)	PER
(15)	ريتويت	retweet	إعادة تغريد	لو قرفان دوس ريتويت (*if depressed click retweet*)	PER

Table 12: **NER task.** Bigger sample false positives mitigated by self-training. These were correctly predicted as the unnamed entity "O" by the self-trained model.

no.	Token(s)	Context/Explanation	Gold	FT	ST
(1)	بالمبارك	بالمبارك عاد احنا (*We are still in Mubarak*)	LOC	PER	O
(2)	محشش	محشش دخل المحاضرة (*a drunk entered the lecture*)	O	PER	PER
(3)	بأمارة	بأمارة ايه وفين (*what is the evidence/sign and where?*)	O	LOC	LOC
(4)	لمستفشي	لمستفشي قصر الدوباره (*to Qasr AlDobara Hospital*)	LOC	O	O
(5)	كنتاكي	عند كنتاكي (*by Kentucky [resturant]*)	LOC	O	O
(6)	داون تاون	مشروع داون تاون بطنطا (*a down town Tanta project*)	LOC	O	O
(7)	يابطل	مبروك يابطل (*Congratulations, hero!*)	O	PER	PER
(8)	الاخوان	نختلف مع الاخوان (*we disagree with the Muslim brotherhood*)	ORG	O	O
(9)	قناة العربية	شفت قناة العربية (*watched Al Arabya Channel*)	ORG	O	O
(10)	المجلس العسكري	اللي عمله المجلس العسكري (*what the military council did*)	ORG	O	O

Table 13: **NER task.** Sample errors that are not fixed by self-training (shared with the mere fine-tuned model)

Multiple Tasks Integration: Tagging, Syntactic and Semantic Parsing as a Single Task

Timothée Bernard[*]
LLF, Université de Paris, France
`timothee.bernard@ens-lyon.org`

Abstract

Departing from both sequential pipelines and monotask systems, we propose Multiple Tasks Integration (MTI), a multitask paradigm orthogonal to weight sharing. The essence of MTI is to process the input iteratively but concurrently at multiple levels of analysis, where each decision is based on all of the structures that are already inferred and free from usual ordering constraints. We illustrate MTI with a system that performs part-of-speech tagging, syntactic dependency parsing and semantic dependency parsing. We observe that both the use of reinforcement learning and the release from sequential constraints are beneficial to the quality of the syntactic and semantic parses. We also observe that our model adopts an easy-first strategy that consists, on average, of predicting shorter dependencies before longer ones, but that syntax is not always tackled before semantics.

1 Introduction

Historically, Natural Language Processing (NLP) systems have generally been built as sequential pipelines, where each module adds another layer of annotation, in order of (supposed) increasing complexity. Progress in neural networks has, however, led to the development of state-of-the-art systems that completely bypass intermediate levels of analysis that were previously considered essential. For example, the system of Zhou and Xu (2015) perform Semantic Role Labeling (SRL; Carreras and Màrquez, 2005) without referring to any explicit (morpho)syntactic information.

A well known problem of sequential systems is *error propagation*, which happens when an incorrect prediction at some point in the process leads to more incorrect predictions at a later stage. In traditional pipelines, one of the roots of error propagation lies in the fact that they feature a unidirectional flow of annotation between their different stages.

End-to-end systems with no intermediate level of analysis, in contrast, are protected against one form of error propagation. Since such systems are not contingent on possibly faulty levels of analysis, they are free from the interference they could cause. However, the absence of any intermediate decisions — symbolic traces of the system's computation — raises questions about generalisation ability and interpretability (Lipton, 2018). What is more, by reducing the information in the prediction and training signals at their disposal, they might not leverage the full power of the neural networks they are usually based on. Indeed, in their reintroduction of syntactic dependencies to the training process, Strubell et al. (2018) managed to develop a state-of-the-art system for SRL that at the same time computes good-quality syntactic parses.

Departing from these two kinds of architectures, we propose *Multiple Tasks Integration* (MTI). The principles of MTI are (i) to let the system take actions pertaining to different levels of analysis without constraining their order, and (ii), at any given point of the process, to feed all layers of annotation as input for the next predictions. The different tasks can therefore fully interact with each other as if they were a single task. Our main contribution lies in an illustration of these principles with a system that performs part-of-speech (POS) tagging, syntactic dependency parsing and semantic dependency parsing (SDP), on English data. We have chosen these specific tasks not only for their strong interdependence but also for their generality: many other tasks in NLP (e.g. SRL, coreference resolution, relation extraction) can be reduced to labelling or bi-lexical dependencies creation problems. We show that in this specific case, letting the system order freely its actions across all three tasks leads to better performance, and that this im-

---

[*] Work done while at AIST (Japan).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 783–794
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

provement concerns the syntactic layer as well as the semantic one, although to a lesser extent.

2 Related work

Fighting error propagation Lê and Fokkens (2017) identify two kinds of error propagation. First, error propagation happens when an incorrect action makes a subsequent correct action unavailable, because the two are logically inconsistent. Second, it also happens when an action leads to an unusual state (unseen during the training process), thus disrupting the system in such a way that a subsequent correct action is not taken even though both are logically compatible.

Since its apparition for speech recognition (Lowerre, 1976), *beam search* has become ubiquitous in NLP and artificial intelligence. Exploring multiple paths of actions in parallel instead of a single one mitigates the risk of error propagation of both kinds. An orthogonal approach consists in designing systems able to attain the gold annotation via multiple paths (usually involving different orderings of subsequences of actions) and training them to infer an *easy-first strategy*. Such a strategy corresponds to performing easy actions first so as to maximise the amount of reliable information available when having to tackle harder decisions. This idea has been applied to various NLP problems (Shen et al., 2007; Goldberg and Elhadad, 2010; Stoyanov and Eisner, 2012; Xie et al., 2015), but, to the extent of our knowledge, only in single-task contexts. Another promising research direction is the one of *iterative refinement*. The idea here is to use a baseline system in order to produce an initial output and to then correct it, possibly multiple times, with a refinement system that has access to both the previous output and the input of the problem. Iterative refinement techniques have been used for tasks such as machine translation (Lee et al., 2018) and SRL (Lyu et al., 2019).

Other techniques are designed to help to fight error propagation of the second kind, which focus on the training of the system to make it robust to its own mistakes. One possibility consists in training the system to predict the next best action in any state it find itself in, instead of simply staying on the (errorless) gold path. Doing so requires the possibility for a next best action to be determined, which, while trivial in some cases such as POS tagging, is not in others, in particular for structured prediction. Hence the introduction of the notion of *dynamic oracle* by Goldberg and Nivre (2012) for syntactic parsing. An alternative is *reinforcement learning* (RL; Sutton and Barto, 2018), which also trains the system from its actual trajectories, but depends instead on *rewards* defined for each action in the particular state it is taken. Note that in adequate settings, RL has been shown to lead to the emergence of easy-first strategies. This is the case in semantic parsing, as shown in the work of Kurita and Søgaard (2019), which we extend here to both POS tagging and syntactic parsing following our proposed Multiple Tasks Integration principles rather than implementing a traditional MultiTask Learning scheme.

MultiTask Learning (MTL) The main idea behind MTL is that one can often increase the performance of a given neural-based system by sharing some of its weights with other systems trained to perform other tasks (Caruana, 1997). Doing so tends to strengthen the training signal of the shared parts and to drive the model to develop richer and more relevant distributional representations. Refer to the texts of Ruder (2017) and Zhang and Yang (2018) for reviews of the wide NLP literature on the subject.

While MTL deals with tasks usually done rather independently, an alternative approach — which does not require weight sharing — is explored by Peng et al. (2018a). They work with pipelines where intermediate layers are expressed as solving a constrained maximisation problem, a setting for which they develop a technique, SPIGOT, to backpropagate gradient through argmax operations (which are discrete, hence non-differentiable). This allows for a lower module to be trained to predict structures used by a higher module so as to optimise the predictions of the latter. Note that the gain in performance obtained by introducing SPIGOT in a pipeline cannot necessarily be attributed to a reduction in error propagation: a lower module might end up generating less accurate yet more helpful predictions.

Joint syntactic-semantic parsing In 2008 and 2009, the CoNLL shared tasks were focused on the joint parsing of syntactic and SRL-based semantic dependencies (Surdeanu et al., 2008; Hajič et al., 2009). Most participants developed sequential systems that parse syntax before semantics, one notable exception being the proposal of Henderson et al. (2008), further developed by Henderson et al.

(2013) and on which the work of Swayamdipta et al. (2016) is based. These joint parsers are transition-based systems that alternate between sequences of syntactic actions and semantic ones. Both types of actions share the same symbolic structure, meaning that these systems allow for a strong interaction between syntax and semantics. While they exhibit no sequentiality globally over the two subtasks, the alternation of actions still enforces a strict ordering locally: once a new token is shifted from the input queue, it is processed syntactically before being processed semantically. This contrasts with the method adopted by Constant and Nivre (2016) for joint syntactic parsing and multiword expression detection, who compute, at each time step, a distribution of probability over actions pertaining to both tasks — as in our system. However, their linear (left-to-right) processing still imposes strong constraints on possible action orders.

More recently, Strubell et al. (2018) designed a model for SRL on the basis of the transformer architecture (Vaswani et al., 2017) in which one attention head is trained to attend to the syntactic head of the corresponding tokens. At inference time, the model is able to recover good-quality syntactic parses after (argmax) discretisation of the attention vectors. In this system and in stark contrast with our approach, the interaction between the two tasks can only take place during the single pass within the transformer network, where SRL and syntactic dependencies are only present under distributional forms (what we could call the "distributional soup of information"), before they are independently discretised into symbolic structures.

Semantic Dependency Parsing (SDP) The SDP task, introduced for the SemEval workshop (Oepen et al., 2014, 2015), represents semantic analysis as a directed acyclic graph (DAG) the nodes of which are tokens of the sentence. This means that each token may have zero or more parents. In addition, zero or more tokens in each sentence may be annotated as *top predicates*.

A wide range of techniques has been applied in the literature in order to tackle SDP. Peng et al. (2017, 2018b,a) propose graph-based constrained maximisation systems based on the AD^3 decoding algorithm (Martins et al., 2011). Dozat and Manning (2018) adapt a graph-based syntactic parser to DAG parsing while Wang et al. (2018) adapt a transition-based one. Zhang et al. (2019) present a very general sequence to graph transduction system. Wang et al. (2019) obtain excellent results by scoring not only the potential dependencies but also some of the potential pairs of dependencies. A second-order parser is thus defined, which relies on a (differentiable) mean field variational inference layer (or, alternatively, a loopy belief propagation one). The model closest to ours is the one of Kurita and Søgaard (2019), who see SDP as an iterated head-selection problem. They take inspiration from Zhang et al. (2017)'s system, which builds a syntactic tree by selecting a head for each token and correcting the corresponding dependencies with well-formedness heuristics. In this work, we see both syntactic and semantic parsing head-selection problems, that we tackle jointly with POS tagging.

3 Model

3.1 Overview

We define six types of *actions*. Relative to a token i, TAG-t corresponds to tagging i with POS tag t, SYN-j-l to creating a syntactic dependency labelled l from token j to i, ROOT to setting i as the (syntactic) root, SEM-j-l to creating a semantic dependency labelled l from token j to i, TOP_PRED to setting i as a (semantic) top predicate and, finally, HALT to doing nothing.

The inference process is summarised in Algorithm 1. At each time step s, the system first encodes the current state of the analysis into a sequence of one vector per token. These encodings contain information from the three different layers of annotation being built. Then, for each token i independently, its *policy* — a distribution of probability over all possible actions (TAG-t, SYN-j-l, etc.) —, $\pi_{i,s}$, is computed. From each policy $\pi_{i,s}$, an action $a_{i,s}$ is then selected and performed, thus enriching the annotation structure.[1] This means that, at any given step, one action per token is applied, as opposed to a single action for (typical) transition systems. At that point, either the next step starts, or, in case all tokens selected the HALT action, the analysis (or *episode*) stops.[2]

The system is designed to perform one action per token at each time step for computational reasons. Indeed, doing so allows to drastically reduce the number of steps required to analyse a sentence.

---

[1] During training, we sample from the distribution. At inference time, we take the argmax.

[2] If s reaches a limit $s_{\max}$, HALT is imposed to all tokens, leading to the end of the episode. $s_{\max}$ is set to 1.5 times the minimum number of steps required to correctly analyse any training instance (namely, 12).

Algorithm 1: Inference algorithm

```
input: A sentence $\mathbf{x}$.
Initialise empty POS tag, syntactic and
  semantic annotation structures:
  $(s_{\text{tag}}, s_{\text{syn}}, s_{\text{sem}})$;
$s \longleftarrow 0$;
$continue \longleftarrow True$;
while $continue = True$ do
    $\mathbf{l} \longleftarrow encode(\mathbf{x}, s_{\text{tag}}, s_{\text{syn}}, s_{\text{sem}})$;
    for $i \in shuffle([\![1, |\mathbf{x}|]\!])$ do
        if $s < s_{max}$ then
            $\pi_i \longleftarrow policy(l_i)$;
            $a_i \longleftarrow select(\pi_i)$;
        else
            $a_i \longleftarrow$ HALT;
        apply $a_i$ on $(s_{\text{tag}}, s_{\text{syn}}, s_{\text{sem}})$;
    if $\forall i,\ a_i =$ HALT then
        $continue \longleftarrow False$;
    $s \longleftarrow s + 1$;
return $(s_{tag}, s_{syn}, s_{sem})$
```

Note that this does not entail that analyses are three steps long, as (i) tokens can have more than a single semantic head and (ii) any token can decide to wait with a HALT action.

We do not ensure that the syntactic (resp. semantics) structure computed is a tree (resp. DAG); we leave the implementation of relevant heuristics as a subject of future work. We do impose several constraints, however: (i) a token can be annotated with only one POS tag, (ii) there can be at most one root, (iii) there can be no incoming syntactic dependency on a root, (iv) there can be at most one syntactic (resp. semantic) dependency from token j to i. Our strategy is to always perform the selected action, overwriting possible incompatible previous annotations.[3]

While POS tags, syntactic and semantic dependencies are used during training (see Section 4), only the raw tokens are required at inference time.

3.2 Training

We typically train our model with a supervised pre-training phase followed by a reinforcement learning (RL) phase. During the pre-training phase, the log-likelihood of the model on *goldish* sequences of actions is maximised: for each token of a sentence, we generate a sequence of actions of minimum length leading to the gold annotation, randomly permute it (each time with a different permutation) and pad it with HALT so as to match the length of the longest sequence (over the sentence), before a final HALT is added. For example, for a token tagged with the POS tag of id 7, syntactically dependent (label of id 12) on the token 3, and that is neither a top predicate nor semantically dependent on any token, one of the two goldish sequences is

[SYN-3-12, TAG-7, HALT, HALT, HALT]

(if all tokens in the sentence can be fully annotated with a maximum of four actions). Using such sequences intuitively teaches the system to predict all annotations as quickly as possible, with no ordering preference. When the performance of the model on the SDP task saturates (early stopping on the development set), we switch to the RL phase, during which we expect the model to infer a good strategy as to how to order the different actions.

In relation to the reinforcement process, each action $a_{i,s}$ is associated with a reward $r(a_{i,s}, i, s)$. The role of RL is to train the model to maximise the expected sum of rewards $J = \mathbb{E}(R)$, where $R = \sum_s R_s$ and $R_s = \sum_i r(a_{i,s}, i, s)$. We compute the rewards in the following way. Let #pos be the number of POS annotations (which is also the number of tokens) in the training set, while N is the number of sentences. We then define $r_{\text{pos}} = \frac{N}{\#\text{pos}}$. The creation of a correct POS annotation and the suppression of an incorrect one (which can happen by overwriting) both correspond to a reward of r_{pos}, while the creation of an incorrect POS annotation and the suppression of a correct one correspond to a reward of $-r_{\text{pos}}$. Syntactic and semantic dependencies are treated similarly, with syntactic root (resp. semantic top predicate) annotations counted as a virtual syntactic (resp. semantic) dependencies. As a consequence, the construction of the full gold structure corresponds on average to a reward of 3 per sentence, equally balanced across the three layers. The reward associated to a given action is then computed as the sum of the reward of its effects, minus a small constant negative penalty

---

[3] This is also ensured within the actions selected for the different tokens of the sentence during a given time step, which, strictly speaking, we do not perform “in parallel”, but in random order, as can be seen in Algorithm 1. As a result, if ROOT is selected for two tokens, one of them is set first as the root (deleting any incoming syntactic dependencies) — information that is immediately overwritten by setting the other one as the root.

in the case of non HALT actions (set at a tenth of the average reward per token in the training set) aimed at discouraging the model from loitering.

The RL algorithm we use to optimise our model is a modification of REINFORCE (Williams, 1992). We adapt it to take into account the fact that multiple actions (one per token) are performed at each time step. To do so, we simply distribute future rewards to each token's discounted reward equally. More formally, given the *discount factor* γ, the discounted reward for token i at time step s is defined as

$$G_{i,s} = r(a_{i,s}, i, s) + \sum_{s' \geq 1} \gamma^{s'} \frac{R_{s+s'}}{n},$$

where n is the length of the sentence.[4] The direction of the parameters update for a given episode is then the one obtained from REINFORCE summed over all tokens:

$$\nabla_\theta L_p = \sum_{s \geq 0, i} (G_{i,s} - b_{i,s}) \nabla_\theta \log \pi_{i,s}(a_{i,s})$$

where $b_{i,s}$ is the *baseline term*.[5] We use a *state value* baseline, which is trained by minimising its squared error with the observed return:

$$\nabla_\theta L_b = \sum_{s \geq 0, i} 2\,(\nabla_\theta b_{i,s} - G_{i,s}).$$

The policy and baseline parameters updates are weighted with coefficients 0.67 and 0.33 respectively. Note that we update the baseline term also during the pre-training phase, using the rewards obtained following the goldish sequences. Finally, we additionally maximise the entropy of each policy with a coefficient 0.002.

Optimisation is done using Adam (Kingma and Ba, 2015). The search for learning rates has been done manually, optimising the semantic F1 on the development set using the DM formalism.[6] We first found that a learning rate of 5.10^{-4} during pre-training gave satisfying results (the other optimiser parameters are left as set by default in TensorFlow, i.e. $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$), then that using a learning rate of 5.10^{-5} during the subsequent RL phase was a reasonable choice.

3.3 Token representations

We first define a *base encoding* for each token, composed of a 100-dimensional pre-trained GloVe word embedding (Pennington et al., 2014) concatenated to a POS tag embedding[7], a sum of prefix embeddings, a sum of suffix embeddings, a few other *tagging* features and two binary features indicating whether the token is currently predicted as the root or a semantic top predicate. (More detail is given in appendix A.) These base encodings are first sent through a dense ReLU layer before serving as input to the *token representation module*, which is composed of two layers of the architecture depicted in Figure 1. These layers contain a *syntax encoder* and a *semantics encoder*, the role of which are to linearise the partial syntactic and semantic structures respectively. These graph encoders are inspired by the work of Kurita and Søgaard (2019); the main difference lies in the use here of summation instead of a more powerful but also more expensive recurrent neural network.

Given a labelled graph — defining a set of parents *parents*(i) and of children *children*(i) for each token i along with $l(j, i)$, the label of the arc from j to i when it exists — and noting u_i for the input corresponding to token i and v_l for the embedding of label l, we define the *downward encoding* of i as $w_i^d = \sum_{j \in \mathit{children}(i)} \mathit{Dense}([u_j, v_{l(i,j)}])$ (where *Dense* represents a dense ReLU layer). Similarly, the *upward encoding* of i is defined as $w_i^u = \sum_{j \in \mathit{parents}(i)} \mathit{Dense}([u_j, v_{l(j,i)}])$ (where *Dense* represents another dense ReLU layer). Finally, the *bothward encoding* of i is defined as the concatenation of the two previous vectors: $w_i = [w_i^d, w_i^u]$. For the output of the syntax encoder, we use the bothward encodings of the syntactic graph while for the output of the semantic encoder we use only the upward encodings of the semantic graph.

The output of the second layer of the token representation module is sent (i) to a multilayer perceptron (MLP) that computes the state value $b_{i,t}$ (the baseline term) and (ii) to different sub-networks that compute the logits of the actions (i.e. the values from which the probabilities are obtained by applying softmax), which are described in the next section.[8]

---

[4] Kurita and Søgaard (2019) do not discuss the definition of the discounted rewards and only use the immediate local reward (corresponding to a discounting factor $\gamma = 0$). We use here the fairly standard value $\gamma = 0.99$.

[5] The baseline term is not strictly necessary: its goal is to reduce the variance of the estimate of the gradient in order to speed the learning process (Sutton and Barto, 2018, §13).

[6] See Section 4 about the different formalisms.

[7] The corresponding vocabulary includes a "not tagged yet" embedding.

[8] All BiLSTMs that we use are actually 2-layer BiLSTMs, and all MLPs have two hidden layers.

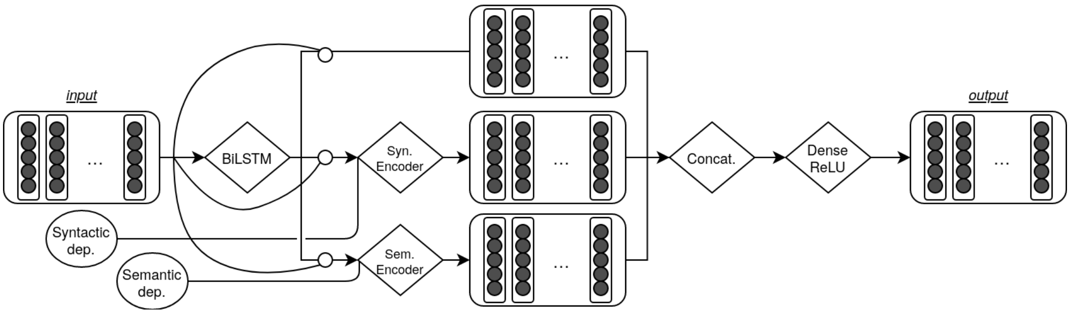


Figure 1: One layer of the token representation module. Each sequence is composed of one vector per token. Each of the circles directly linked to the output of the BiLSTM is a gate parametrised by a scalar α and represents the function $(v_1, v_2) \mapsto \sigma(\alpha)v_1 + (1 - \sigma(\alpha))v_2$. See the text for the definition of the syntax and semantics encoder.

3.4 Action logits

The logit of each action is computed by one of three sub-networks. The first one is an MLP that returns the logits for TAG-t actions, ROOT, TOP_PRED and HALT. The second returns the logits for SYN-j-l actions: the logit, for token i, to select token j as governor, for all possible dependency labels is given by MLP($[v_i, v_j, v'_{i,j}]$) $\in \mathbb{R}^{|L|}$, where $|L|$ is the number of dependency labels and $v'_{i,j} \in \mathbb{B}^{|L|+1}$ is a one-hot vector indicating whether j is currently governor of i and if so, what is the label of the corresponding dependency. The last sub-network returns the logits for SEM-j-l actions in exactly the same way. The policy is then obtained by applying the softmax function to the concatenation of the output of these different sub-networks.

4 Experiments

To test whether MTI is a viable paradigm and determine the impact of RL, in this section we test the model described above along with three variants. These four models can be seen as combinations of two binary traits. The first trait pertains to the ordering of the actions: *sequential* models simulate sequential pipelines, while for *free* models, as above, no particular constraint is imposed. A sequential model can only select TAG-t actions during the first step of an episode, only SYN-j-l and ROOT during the second and only SEM-j-l, TOP_PRED and HALT afterwards. This is ensured by using as policy for each token the vector obtained by normalising a masked version of its usual policy.[9] The second trait pertains to the learning process: *supervised* models are only trained in the pre-training phase while *RL* models, as above, are further trained using reinforcement learning.

4.1 Data

We use data from the SemEval 2015 Task 18 (Oepen et al., 2015), which provides POS tags, lemmas, syntactic parses and semantic parses for English, Chinese and Czech texts. Concerning English, the same data is annotated with three distinct formalisms: DELPH-IN MRS-Derived Bi-Lexical Dependencies (DM), Enju Predicate–Argument Structures (PAS) and Prague Semantic Dependencies (PSD). They all represent a semantic analysis as a bi-lexical dependency DAG. We use here the standard split of the English data. The train set is composed of 33,964 sentences from sections 0–19 of the WSJ corpus, the development set of 1,692 from section 20 and the (in-domain) test set of 1,410 from section 21. During training, for simplicity, we always use the gold (morpho)syntax annotation provided under the form of Stanford Basic dependencies derived from the PTB, even when working with the DM formalism (which comes with different annotations).[10]

4.2 Semantics

Table 1 shows the average F1 for labelled semantic dependencies over the three formalisms in each setting. Three randomly initialised runs are used for each formalism, with each performance computed using the script provided by the shared task. Everywhere in this text, F1 are *micro*-F1 (i.e. computed over the whole dataset instead of at the sentence level). We first observe that RL models perform better than their supervised-only counterparts, a

---

[9] The goldish sequences used during pre-training are modified accordingly. The ordering of the selection of the possibly multiple semantic parents of any given token is still random.

[10] All data and evaluation scripts can be found on the shared task's website: `http://alt.qcri.org/semeval2015/task18/`.

(sem.)	sequential	free	Δ
supervised	84.9 (86.4)	85.7 (87.2)	0.8 (0.8)
RL	86.5 (87.9)	87.3 (88.5)	0.8 (0.6)
Δ	1.6 (1.5)	1.6 (1.3)	

Table 1: Average of the F1 over the three semantic formalism on the in-domain test set. Scores on the development set are given in brackets.

result consistent with the findings of Kurita and Søgaard (2019). This indicates that RL allows the model to infer a good ordering strategy (limited to the ordering of the semantic actions in the sequential regime) and/or trains it to be more robust to its own mistakes (an effect similar to that of dynamic oracles). Second, we observe that free models perform better than their sequential counterparts. This indicates, as we had hypothesised, that a strict ordering of actions following a traditional hierarchy of levels of analysis is not optimal and that MTI is a potentially powerful form of MTL (note that the sequential models here implement a relatively traditional form of MTL). Finally, we see that these two effects are additive.

4.3 Syntax and POS tagging

Let us turn to the performance of our model on the syntactic parsing and POS tagging tasks in the four settings. Table 2 shows the F1 for labelled syntactic dependencies[11], averaged over all nine corresponding runs. Following Chen and Manning (2014)'s recommendation, punctuation is excluded from the evaluation. What we observe is that while the use of RL and (more particularly) non-sequentiality have a small impact on syntactic parsing, it is still noticeable. All scores on the test set are statistically different according to Pitman's permutation test, with a p-value under 10^{-3}.[12] The combination of RL with free-ordering appears a particularly good combination.

As shown in Table 3, the use of RL also has a statistically significant impact on POS tagging, although of very low magnitude. Free-ordering, however, has no perceptible effect (the corresponding deltas are in fact positive but strictly smaller than 0.05 and the distributions they compare are not distinguishable).

(synt.)	sequential	free	Δ
supervised	91.0 (91.1)	91.1 (91.4)	0.2 (0.3)
RL	91.8 (91.9)	92.3 (92.2)	0.5 (0.4)
Δ	0.8 (0.8)	1.2 (0.8)	

Table 2: Average syntactic F1 on the in-domain test set. Scores on the development set are given in brackets.

(tag.)	sequential	free	Δ
supervised	96.9 (97.1)	96.9 (97.1)	0.0 (0.0)
RL	97.1 (97.2)	97.2 (97.2)	0.0 (0.0)
Δ	0.2 (0.1)	0.2 (0.1)	

Table 3: Average tagging F1 on the in-domain test set. Scores on the development set are given in brackets.

Note that while we do not use the same split of the WSJ Corpus as them, a comparison with the 91.87 LAS on syntax and 96.92 accuracy on POS tagging obtained by the model of Strubell et al. (2018) — which is also trained to perform these tasks jointly with a form of semantic parsing — indicates that our free+RL model, with 92.3 F1 on syntax and 97.2 on POS tagging, performs at least reasonably well.

4.4 Inferred strategies

In order to better understand why our MTI paradigm leads to better performance not only on the syntactic task but also on the semantic one, we now perform a brief study of the ordering strategy inferred by our free+RL model. We focus here on the SYN-j-l and SEM-j-l actions and look at the average time step at which they are created as a function of their length.[13] Figure 2 shows that for the PAS formalism, on average, the system creates (i) syntactic dependencies before semantic ones and (ii) short dependencies before long ones.[14] This tends to indicate that the model follows an easy-first strategy and that, as often assumed, it is preferable

---

[11] As we do not constrain the model to produce complete syntactic trees, the natural measure here is indeed F1. In practice, however, the *parsing rate* (the ratio of tokens that either have an incoming dependency or are predicted to be the root) is always higher that 99%, meaning that recall and precision are almost equal and that F1 is very similar to a labelled attachment score (LAS). A similar remark applies to POS tagging.

[12] We use the permutation test to check whether the distributions of the scores in each setting (nine runs) are distinct from each other. See the work of Dror et al. (2018) for an NLP-oriented discussion of statistical significance.

[13] Sometimes, though rarely, does the model select the same action multiple times (for the same token) over a single episode. We filter out useless actions, that is to say, the ones that do not modify the structure when they are applied.

[14] Each figure mentioned in this paragraph corresponds to a randomly picked run. The behaviour observed is very consistent across the different runs for a given semantic formalism.

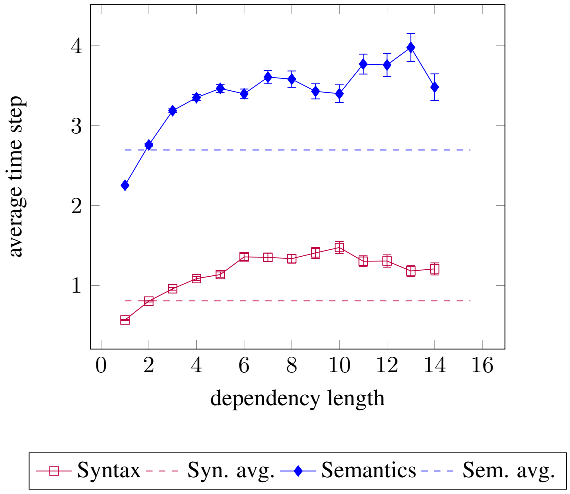


Figure 2: PAS semantic formalism: average time step (from 0) at which syntactic and semantic dependencies are created as a function of their length. Error bars indicate the standard error of the mean.

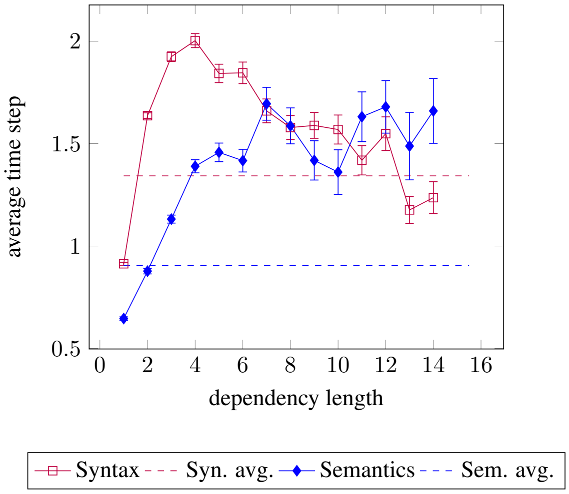


Figure 3: DM semantic formalism: average time step (from 0) at which syntactic and semantic dependencies are created as a function of their length. Error bars indicate the standard error of the mean.

Model	DM	PAS	PSD	all	avg.
FREDA3	90.4	92.7	78.5	88.0	87.2
K&S (-lem.)	91.2	92.9	78.8	88.5	87.6
SPIGOT	91.6		78.9		
MF	**93.0**	**94.3**	**80.9**		89.4
free+RL	91.1	91.7	79.0	88.1	87.3

Table 4: Semantic F1 on the in-domain test set for each of the three formalisms, along with the F1 over all formalisms (*all*) and the average of the F1 over all formalisms (*avg.*). *All* tends to be higher than *avg.* because PAS happens to be both the most simple formalism and the one introducing the highest number of dependencies, while the opposite is true of PSD.

to perform syntactic before semantic parsing[15], at least when dealing with this specific semantic formalism. Indeed, figure 3 shows that the strategy inferred when training on the DM formalism is radically different: short dependencies are still created before long ones (note that the number of dependencies decreases very rapidly with their length), but the syntactic and the semantic structures are built much more concurrently, with semantic structures being, on average, generated *before* syntactic ones of the same length. The strategy inferred for the PSD formalism, not depicted here, is qualitatively similar to the one for DM. In light of these observations, we can better understand why both tasks benefit from MTI: it is *not* true, in general, that syntactic parsing should be performed (even on average) before semantic parsing.

4.5 Comparison with other parsers

Finally, table 4 compares the performance of our free+RL model with state-of-the-art SDP parsers that, for fairer comparison, do not use ELMo or BERT embeddings, nor rely on lemma input: Peng et al. (2017)'s FREDA3, Peng et al. (2018a)'s SPIGOT, Wang et al. (2019)'s MF model and a model by Kurita and Søgaard (2019).[16] Note that our model is the only one that does *not* rely on the provided POS tags at inference time.[17]

Wang et al. (2019)'s second-order model, based on mean field variational inference, is clearly ahead in terms of performance. Among the other four, our model achieves competitive performance on the DM and PSD formalisms. It is, however, somewhat behind when it comes to the PAS formalism. Understanding why this is so is left as a subject of future research.

5 Conclusion and future work

We have defined Multiple Tasks Integration as a set of principles for joint processing, orthogonal to weight sharing. The essence of MTI is to process the input iteratively but concurrently on multiple

---

[15] On average only, as some semantic dependencies are still created before syntactic ones.

[16] Dozat and Manning (2018) report very high scores, but as indicated in the Q&A session of their ACL 2018 talk (`https://vimeo.com/285804230`) — a fact pointed out by Kurita and Søgaard (2019) —, these scores correspond to a macro-F1 instead of the micro-F1 computed by the shared task tool, thus giving more relative weight to shorter sentences.

[17] In the English data, the PAS and PSD graphs come with the tags of the PTB (Marcus et al., 1993) while the DM ones come with the tags of DeepBank (Flickinger et al., 2012).

levels of analysis, basing each decision on all of the structures already inferred and free from usual ordering constraints. This way, the different tasks can interact in the full sense of the term. To train such a system, we propose using reinforcement learning algorithms, thus allowing it to infer its own ordering strategy.

In practice, we have trained a system to perform part-of-speech tagging, syntactic dependency parsing and semantic dependency parsing. We have observed that both the use of reinforcement learning and the release from sequential constraints are beneficial, not only to the (seemingly) highest level task (i.e. semantic parsing), but also to some intermediate ones (i.e. syntactic parsing). If the inferred strategies are interpreted as being easy-first — which is supported by the fact that shorter dependencies have a strong tendency to be generated before longer ones —, then we have observed that syntactic parsing is not necessarily simpler than semantic parsing and that both benefit from being executed concurrently.

While our model is not yet as effective as today's most complex systems, it is still competitive with most of the parsers presented in the recent literature, even though it uses a poorer input signal for inference (consisting of the raw tokens only). Furthermore, several aspects of the current system are open to developments that seem likely to improve performance. For instance, we do not use here the full potential of reinforcement learning, as what we optimise (the expected sum of rewards) is not the metric we are interested in (which would be either an average of the three F1 or the semantic one). For each of the three tasks, the sum of the rewards we have defined approaches, up to a multiplicative constant, the corresponding F1 of the system when the latter approaches 100%, but a better approximation might prove more successful. In a similar vein, Kurita and Søgaard (2019), who work with a similar architecture as far as SDP is concerned, penalise HALT actions when relevant dependencies are still missing, which intuitively boosts recall. Note also that our network is still rather simple, in that it does not use any form of regularisation nor any advanced technique to handle out-of-vocabulary words.

We would also be interested in applying MTI to other tasks and in studying how well it can learn from incomplete annotations. Our architecture can be straightforwardly adapted to any labelling or graph building task, as long as all nodes are tokens of the input sentence. In contrast, work remains to be done in order to handle formalisms such as Abstract Meaning Representation (AMR, Banarescu et al., 2013) or on how to integrate a generation component with, for example, the goal of translating the sentence being analysed.

Acknowledgments

The author thanks Marie Candito for discussion on this paper.

This paper is based on results obtained from project JPNP15009, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). The computational resources of the AI Bridging Cloud Infrastructure (ABCI), provided by the National Institute of Advanced Industrial Science and Technology (AIST), Japan, were used for this work.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186. Association for Computational Linguistics. Event-place: Sofia, Bulgaria.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75.

Danqi Chen and Christopher Manning. 2014. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Matthieu Constant and Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 161–171. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but More Accurate Semantic Dependency

Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Dan Flickinger, Yi Zhang, and Valia Kordoni. 2012. DeepBank. A dynamically annotated treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, pages 85–96.

Yoav Goldberg and Michael Elhadad. 2010. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 742–750, Los Angeles, California. Association for Computational Linguistics.

Yoav Goldberg and Joakim Nivre. 2012. A Dynamic Oracle for Arc-Eager Dependency Parsing. In *Proceedings of COLING 2012*, pages 959–976, Mumbai, India. The COLING 2012 Organizing Committee.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. 2008. A Latent Variable Model of Synchronous Parsing for Syntactic and Semantic Dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 178–182, Manchester, England. Coling 2008 Organizing Committee.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multilingual Joint Parsing of Syntactic and Semantic Dependencies with a Latent Variable Model. *Computational Linguistics*, 39(4):949–998.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shuhei Kurita and Anders Søgaard. 2019. Multi-Task Semantic Dependency Parsing with Policy Gradient for Learning Easy-First Strategies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2420–2430, Florence, Italy. Association for Computational Linguistics.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Zachary C. Lipton. 2018. The Mythos of Model Interpretability. *ACM Queue*, 16(3):1–27.

Bruce T. Lowerre. 1976. *The HARPY Speech Recognition System*. Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic Role Labeling with Iterative Structure Refinement. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1071–1082, Hong Kong, China. Association for Computational Linguistics.

Minh Lê and Antske Fokkens. 2017. Tackling Error Propagation through Reinforcement Learning: A Case of Greedy Dependency Parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 677–687, Valencia, Spain. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

André Martins, Noah Smith, Mário Figueiredo, and Pedro Aguiar. 2011. Dual Decomposition with Many Overlapping Components. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 238–249, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinkova, Dan Flickinger, Jan Hajic, and Zdenka Uresova. 2015. SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926. Association for Computational Linguistics. Denver, Colorado.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 Task 8:

Broad-Coverage Semantic Dependency Parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep Multitask Learning for Semantic Dependency Parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048, Vancouver, Canada. Association for Computational Linguistics.

Hao Peng, Sam Thomson, and Noah A. Smith. 2018a. Backpropagating through Structured Argmax using a SPIGOT. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1863–1873.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018b. Learning Joint Semantic Parsers from Disjoint Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1492–1502, New Orleans, Louisiana. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. 11283.

Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv:1706.05098 [cs, stat]*.

Libin Shen, Giorgio Satta, and Aravind Joshi. 2007. Guided Learning for Bidirectional Sequence Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 760–767, Prague, Czech Republic. Association for Computational Linguistics.

Veselin Stoyanov and Jason Eisner. 2012. Easy-first Coreference Resolution. In *Proceedings of COLING 2012*, pages 2519–2534, Mumbai, India. The COLING 2012 Organizing Committee.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038. Association for Computational Linguistics. 00015 event-place: Brussels, Belgium.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England. Coling 2008 Organizing Committee.

Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*, second edition edition. Adaptive Computation and Machine Learning series. MIT Press.

Swabha Swayamdipta, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Greedy, Joint Syntactic-Semantic Parsing with Stack LSTMs. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 187–197. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-Order Semantic Dependency Parsing with End-to-End Neural Networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy. Association for Computational Linguistics.

Yuxuan Wang, Wanxiang Che, Jiang Guo, and Ting Liu. 2018. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5561–5568. AAAI Publications.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256.

Jun Xie, Chao Ma, Janardhan Rao Doppa, Prashanth Mannem, Xiaoli Fern, Thomas G. Dietterich, and Prasad Tadepalli. 2015. Learning Greedy Policies for the Easy-First Framework. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Broad-Coverage Semantic Parsing as Transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3784–3796, Hong Kong, China. Association for Computational Linguistics.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency Parsing as Head Selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 665–676, Valencia, Spain. Association for Computational Linguistics.

Yu Zhang and Qiang Yang. 2018. A Survey on Multi-Task Learning. *arXiv:1707.08114 [cs]*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

A Appendix: Base encodings

The base encoding of a given token (introduced in Section 3.3) is composed of the concatenation of four vectors and five binary values.

- The GloVe embeddings that we use are the 100-dimensional vectors of the 6B (uncased) release. We do not fine-tune them. All words present in the training set use their corresponding GloVe entry. All other words are considered unknown and are assigned the average of these embeddings.
- For POS tag embeddings, we use randomly initialised 50-dimensional vectors.
- We use a sum of prefix embeddings. We first consider all cased prefixes of length 1, 2 or 3 and then filter out all those that appear in less that a thousandth of the tokens and less that a thousandth of the word forms in the training set. The remaining prefixes are assigned a randomly initialised 32-dimensional vector. (Unknown prefixes correspond to a zero vector.)
- We use a sum of similar suffix embeddings.
- One binary value indicates whether the token starts with an upper case letter.
- One binary value indicates whether there is any upper case letter in the token.
- One binary value indicates whether the token is a number (matching the `\d+(\.\d+)?` regular expression).
- One binary value indicates whether the token is currently annotated as the (syntactic) root.
- One binary value indicates whether the token is currently annotated as a (semantic) top predicate.

In future work, we plan to substitute a more general character-level word embedding model for the tagging features (i.e., the prefix and suffix embeddings and the three first binary features).

Coordinate Constructions in English Enhanced Universal Dependencies: Analysis and Computational Modeling

Stefan Grünewald[1,2] **Prisca Piccirilli**[1,2] **Annemarie Friedrich**[2]
[1]Institut für Maschinelle Sprachverarbeitung, University of Stuttgart
[2]Bosch Center for Artificial Intelligence, Renningen, Germany
`stefan.gruenewald|annemarie.friedrich@de.bosch.com`
`piccirpa@ims.uni-stuttgart.de`

Abstract

In this paper, we address the representation of coordinate constructions in Enhanced Universal Dependencies (UD), where relevant dependency links are propagated from conjunction heads to other conjuncts. English treebanks for enhanced UD have been created from gold basic dependencies using a heuristic rule-based converter, which propagates only core arguments. With the aim of determining which set of links *should* be propagated from a semantic perspective, we create a large-scale dataset of manually edited syntax graphs. We identify several systematic errors in the original data, and propose to also propagate adjuncts. We observe high inter-annotator agreement for this semantic annotation task. Using our new manually verified dataset, we perform the first principled comparison of rule-based and (partially novel) machine-learning based methods for conjunction propagation for English. We show that learning propagation rules is more effective than hand-designing heuristic rules. When using automatic parses, our neural graph-parser based edge predictor outperforms the currently predominant pipelines using a basic-layer tree parser plus converters.

1 Introduction

The Universal Dependencies (UD) formalism (de Marneffe et al., 2014) is a framework for representing syntactic dependencies between words, prioritizing links between content words. UD parses provide two levels of analysis. *Basic* dependencies form standard syntactic dependency trees in which each node has exactly one governor (black links on top in Figure 1). *Enhanced* dependencies (Schuster and Manning, 2016) are extensions of these trees including additional relations (blue links below sentence) with the aim of representing linguistic phenomena such as coordination, control, or relative clauses. They have been shown to provide valuable input for information extraction tasks (Schuster et al., 2017). One of the most frequent phenomena addressed by enhanced UD is coordination. In the English Web Treebank (EWT), more than 15% of all sentences contain conjoined verbs. Hence, a good representation of coordination clearly is crucial for downstream tasks. For example, in Figure 1, the enhanced layer explicitly captures that the arguments of the predicate "wrote" also fill the corresponding slots of "published," which is highly relevant for natural language understanding tasks.

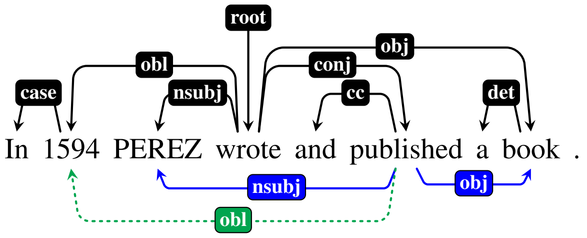


Figure 1: UD **basic** (top) and **enhanced** (bottom) dependencies. **Green dotted link:** proposed addition.

In many cases, enhanced representations can be derived from the gold basic layer in a rule-based fashion (Schuster and Manning, 2016). The currently available English enhanced UD treebanks have been created by applying such a converter. However, we are not aware of a large study regarding their correctness and completeness. Focusing on precision, the converter only propagates core arguments. In this paper, we take a complementary approach, performing a large-scale annotation study in order to determine which set of links *should* be propagated from a semantic perspective. On a new dataset of 1,417 sentences from the EWT containing conjoined verbs, we verify and if necessary modify/extend the links involved in coordinate constructions. We argue that adjuncts such as obliques should in fact be propagated at times, e.g., in Figure 1, the additional (green dotted) link that

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 795–809
April 19 - 23, 2021.

we propose to add facilitates answering questions like "When was the book published?". To the best of our knowledge, our work constitutes the first large-scale annotation effort of this kind.

On the basis of our new dataset, we make the following contributions. First, we estimate the degree of **correctness and completeness of the rule-based converter/existing treebanks**. We find that the converter usually generates correct graphs when applied on gold basic trees, with some notable exceptions involving non-parallel syntactic constructions (e.g., conjuncts having different voice or mood). In addition, the converter does not propagate links correctly in presence of multiple interacting conjunctions. Our inter-anntotator agreement study shows high overlap for propagation decisions, with F1 between pairs of annotators of about 0.9 on average and around 0.75 for obliques.

Second, we address the question of **how to create high-quality treebanks** for enhanced UD from gold basic dependencies, again focusing on coordinate constructions. Based on the findings of our corpus study, we improve the rule-based converter by Schuster and Manning (2016). We also compare machine-learning (ML) based conjunction propagation classifiers in the form of (a) SVM-based classifiers as previously used for Finnish, Swedish and Italian (Nyblom et al., 2013; Nivre et al., 2018), and (b) a novel neural approach integrating tree- and RoBERTa-based features. We find that all systems mostly rely on tree-based features, but contextual embeddings also provide useful information. Performance on propagation decisions has promising F1 around 0.9, already similar to human agreement. ML-based classifiers outperform the rule-based converters on the EWT test set.

Third, we compare methods for **extracting propagated dependencies in an automatic parsing setting.** The currently predominant approach is to run a basic-layer tree parser and then the same converter that has been used for gold standard construction. We propose to use a neural graph-parser based edge predictor with an architecture similar to Dozat and Manning (2018) instead, and show that this approach outperforms pipelines by around 9 points F1 on propagating links in conjunctions.

In sum, our contributions include: (1) a manually curated large-scale dataset of 1,417 sentences addressing semantically motivated correct and complete conjunction propagation in enhanced UD; (2) the proposal of novel neural approaches to conjunction propagation; and (3) experimental evidence that these models outperform rule- and pipeline-based approaches in both gold standard treebank enhancing and automatic parsing settings. To the best of our knowledge, our work constitutes the first principled comparison of various approaches to propagating conjunctions in enhanced UD on manually corrected gold standard data for English. Both our model implementations and the dataset are freely available.[1] We will contribute our changes to the EWT corpus to the next UD release.

2 Related Work

Coordinate Constructions in UD are represented using the *conj* relation, with the first conjunct being the head to which all dependencies of the phrase are attached (see Figure 1). In the basic layer, all governors and dependents of a conjoined phrase are attached to the first conjunct. In the enhanced layer, relations are propagated to the dependent if suggested by the semantics of the sentence.[2] Schuster and Manning (2016) present an algorithm for creating enhanced dependencies automatically based on the basic layer. While it propagates links with high precision, it propagates *only* core arguments by design (see Appendix A). In addition, it is highly reliant on correct basic dependencies (see Sec. 5).

Conjunction propagation classifiers. Nyblom et al. (2013) present an SVM-based approach for enhancing Finnish syntax trees. They observe high performance on conjunction propagation when operating on gold basic trees, but markedly worse results when using automatic parser output. Nivre et al. (2018) evaluate a similar approach for Swedish and Italian. We show that their approach also works well for English, and extend it with neural models and contextualized word embeddings. Simi and Montemagni (2018) de-lexicalize their rule-based converter developed for Italian, showing that their language-independent system also correctly produces most of the propagations for English. However, they evaluate on EWT, which is itself the result of a rule-based system. In contrast, we evaluate on manually checked gold data.

For the related task of dealing with **gapping** constructions such as "Paul likes coffee and Mary tea,"

---

[1] `https://github.com/boschresearch/coordinate_constructions_english_enhanced_ud_eacl2021`

[2] `https://universaldependencies.org/u/overview/enhanced-syntax.html`

Schuster et al. (2018) reconstruct elided predicates by first parsing into an intermediate representation and then applying either a rule-based or an ML-based algorithm to copy over lexical material. We here focus on dependency propagation and operate on gold tokens as annotated in the enhanced UD treebanks, which already include traces. Other related work exists in the area of manual and rule-based error correction on UD treebanks (Wisniewski, 2018; Alzetta et al., 2018).

There is still little published work regarding **fully automatic enhanced UD parsing**, however, the topic has recently been addressed by the IWPT 2020 Shared Task (Bouma et al., 2020). Among the top-performing systems, several approaches first parse into basic UD and then added transformation rules (e.g., Heinecke, 2020; Dehouck et al., 2020). Others directly employ graph parsing techniques (e.g., Wang et al., 2020; He and Choi, 2020; Hershcovich et al., 2020). The overall winner TurkuNLP (Kanerva et al., 2020) transforms enhanced UD into a tree format and then makes use of UDify (Kondratyuk and Straka, 2019). In addition, much work exists on **semantic dependency parsing** (SDP, Oepen et al., 2014, 2015; May and Priyadarshi, 2017). These works differ from UD-based approaches as the respective formalisms represent meaning less close to syntactic structure, thus not requiring propagation. From a modeling point of view, our work is most similar to that of Grünewald and Friedrich (2020), who also use a graph-based biaffine architecture for enhanced UD parsing, and to that of Dozat and Manning (2018), who achieve state-of-the-art results for SDP.

3 Coordinate Constructions Dataset

In this section, we describe our creation of our manually created dataset and analyse the results.

	conj. sentences	**edited**
train	1,926	999
dev	222	222
test	196	196
total	2,344	1,417

Table 1: Coordinate constructions dataset statistics. **conj. sentences**: sentences in EWT containing verb phrase conjunctions; we edited 60% of these.

3.1 Data

Our dataset consists of 1,417 sentences collected from EWT,[3] containing data from five genres of web media (weblogs, newsgroups, emails, reviews, and Yahoo! answers).[4] The basic dependencies of this UD gold standard have been derived from the original Stanford dependencies (de Marneffe et al., 2006) and were then hand-corrected. The enhanced layer has been created using the automatic converter (Schuster and Manning, 2016, see Appendix A). We retrieve all sentences containing at least one *conj* link between two verbs. More than 15% of all sentences in EWT contain conjoined verbs. Out of these sentences, we edit all sentences of the dev and test sets, and 999 sentences of the training set, amounting to more than 60% of all relevant sentences in EWT (see Table 1). The careful curation of each sentence took around 10 minutes on average, amounting to a total annotation effort of around 240 hours (total costs ca. $4,750). We exclude 18 sentences when reporting our statistics: In 12 cases, the *conj* relation is annotated wrongly in the basic layer and six sentences contain syntactically non-standard English.[5]

3.2 Annotation Methodology

The manual corrections of the treebank were performed by a French native speaker with an extensive background in linguistics. The annotation project involved regular discussions among all authors to decide on uncertain cases and to ensure consistency. Additionally, in case of doubt, an English native speaker with an extensive linguistics background was consulted. Dependencies were checked carefully sentence-wise using the ConLL-U-Editor tool (Heinecke, 2019). If necessary, the full document was consulted to make sure interpretations were correct in context.

Annotation Guidelines. We verify and modify all links involved in coordinate constructions including conjoined verbs, but also noun or adjectival phrases. First, we make sure that the automatically constructed enhanced representations adhere to the official guidelines for enhanced UD (see Sec. 2), propagating heads and dependents of conjuncts if

---

[3] Linguistic Data Consortium LDC2012T13.

[4] `https://universaldependencies.org/treebanks/en_ewt/index.html`

[5] Such as "i want to be able to use it in my car, out n about etc...i guess like an iphone, but thats later on and ,i know what they are so no suggestions on just goin out to buy one im talking about right now just for an ipod??" (EWT dev set)

	A	B	C
A	-	90.1	94.9
B	95.2	-	97.2
C	80.5	77.9	-

Table 2: Inter-annotator agreement on propagated links for 100 sentences: **precision** when treating the row annotator as gold standard (or, equivalently, **recall** when treating the column annotator as gold standard).

the interpretation of the sentence suggests additional syntactic relations between words.

As each verb may also have its own complements, this task requires a semantic interpretation leveraging context and knowledge about selectional preferences. If an ambiguity has already been resolved in the basic layer,[6] we follow this interpretation unless obviously wrong. Second, we propose to also propagate non-core dependents such as *obl*, *advcl* and *advmod* if suggested by semantics, an annotation task similar to prepositional phrase attachment resolution. We only propagate such links if the adjunct clearly modifies each conjunct (as in Figure 1). Finally, we extend the attachment of relative pronouns (*ref*) to all antecedents if involved in coordinations. We focus on propagating dependencies between content words, not propagating relations such as *aux* or *cop*, which could be handled as traces.

Inter-annotator agreement study. We sampled 100 sentences, half of them from cases where the primary annotator had judged the original version to be correct, and half of them cases that included modifications. This sample was blindly re-annotated by two secondary annotators, both German native speakers with an extensive computational linguistics background. Table 2 shows agreement in terms of precision and recall on the set of dependencies resulting from conjunction propagation, i.e., the links involved in conjunctions that are present in the enhanced layer but not in the basic layer. For a formal definition, see Appendix B. Agreement is generally high, particularly between annotators A and B. Annotator C was more conservative in propagating links, especially in generally ambiguous cases. However, the links that C propagates are also propagated by A and B. Pairwise agreement was high on *nsubj*, *obj* and *xcomp*. Modifier clauses (*acl*, *advcl*) and adverbials (*advmod*) were common sources of disagreement, indicating the more ambiguous nature of these propagations. Pairwise scores and more details can be found in Appendix B.

3.3 Analysis and Discussion

In this section, we analyse and discuss the modifications made to the original treebanks.

Quantitative Analysis of Changes. Table 3 presents the numbers of dependency relations that have been added and removed in coordinate constructions in the enhanced layer. More specifically, we consider only the set of links not present in the basic tree and count modifications regarding links starting or ending at conjuncts.[7] Counts for coarse-grained labels (e.g., *nmod*) include all subtypes (e.g., *nmod:for*) not explicitly listed in the table. During our manual correction of the treebank, around 15% of the total enhanced links involved in conjoined phrases were added and about 3% were removed. This confirms that the converter by Schuster and Manning (2016) is optimized for precision rather than recall, though our additions of course include labels that the converter does not address. Note that in these cases, removed relations in Table 3 are caused by fixes regarding attachment in the basic layer, whose errors had been propagated to the enhanced layer. In total, we fixed errors in 57 sentences in the basic layer. In 42 of these, this led to changes in the enhanced layer.

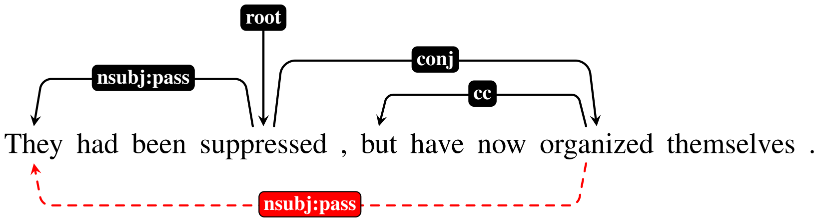
Linguistic Analysis of Changes. One systematic error involves **links to subjects in passive constructions**: 18 out of 225 *nsubj:pass* links were actually wrongly propagated. All of them have been changed to *nsubj*. The reason is that the converter automatically propagates an *nsubj:pass* link if the first conjoined verb is in the passive form, as, e.g., in “These Shiite movements had been suppressed by Saddam Hussein’s regime, but have now organized and armed themselves” (see Figure 2a). Another common error (occurring 12 times) is the propagation of the first conjoined verb’s subject to the second verb, even though the latter is in **imperative mood**, as, e.g., in “I think it was the Lincoln Square area but don’t quote me on that” (see Figure 2b).

In sentences containing **multiple coordinate constructions**, such as “Dr. Fortier and his girlfriend lashed two canoes together and paddled

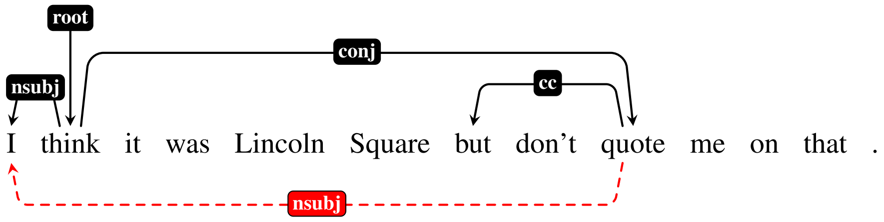
---

[6] For example, in “She was reading or watching a movie,” “movie” is attached to the second conjunct “watching” in the basic layer, hence resolving the syntactic ambiguity.

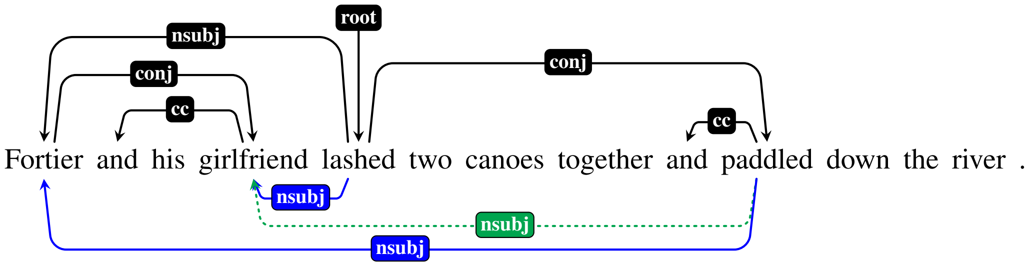
[7] We made some additional fixes (not necessarily related to coordinations) to the original treebank, see Appendix C.



(a) Passive voice: *nsubj:pass* links should not be propagated verbatim if the second conjunct is in active voice.



(b) Imperative mood: *nsubj* links should not be propagated if the second conjunct is in imperative mood.



(c) Multiple coordinate constructions: *nsubj* links should be propagated between the second conjuncts of each coordination.

Figure 2: Systematic errors found in the automatic propagation of dependencies. (Only coordination-relevant links are depicted.) **Red dashed link:** Incorrect propagation or incorrectly labeled propagation. **Green dotted link:** Missing propagation.

label	**#added**	**#removed**	**#sents**	**#total**
acl	14	4	12	68
acl:relcl	13	3	9	190
advcl	32	3	31	167
advmod	46	2	35	6
amod	19	2	14	7
ccomp	9	10	11	50
nmod	32	0	23	102
nmod:poss	11	0	10	18
nsubj	160	30	118	249
nsubj:pass	8	18	25	225
nsubj:xsubj	22	10	17	1688
obj	9	5	12	71
obl	72	0	51	150
ref	12	1	8	61
xcomp	7	5	10	8
all	466	93	386	3060

Table 3: Statistics of modifications made to 1,399 sentences of the EWT. **#sents** reports the number of sentences in which the respective reported changes were made, **#total** reports the number of occurrences of the label in the enhanced layer of the original treebank.

eight kilometres along the Soper River," *nsubj* links should be present in the enhanced layer between both conjuncts of the subject noun phrase and both verbs. However, in the original treebank, the second subject conjunct was never propagated to the second verb (see Figure 2c). Similarly, we also added many relations in cases of **nested coordinations** as in "These Shiite movements had been suppressed by Saddam Hussein's regime, but have now organized and armed themselves." The second conjunct of the conjoined verb phrase is a conjoined verb phrase itself, but the *nsubj* link to "armed" was missing. In total, 194 sentences contain several coordinations, and we modified 92 of them. This phenomenon also accounts for 45 of the added *nsubj* links.

Some originally missing propagations concern **adjectival and adverbial modifiers** (*acl, amod, advcl, advmod*), which are known to be ambiguous cases. In "Handwritten notes and files on a laptop were seized," the adjective "handwritten" clearly modifies the first conjunct "notes" only, but in "Several Indian scholars and politicians have been ready to say and endorse anything," the propagation of "several" and "Indian" was added during our modifications. These cases involve world knowledge that the converter currently does not handle.

Finally, consider the sentence "We recognize that the state may not require religious groups to officiate at, or bless, same-gender marriages." Both conjuncts take "marriages" as their argument, but as an *obl* and as an *obj* relation, respectively. The resolution of such non-parallel constructions requires detailed subcategorization information.

4 Modeling

In this section, we describe three approaches to generating links propagated due to coordination: (1) an improved version of an existing converter (Sec. 4.1); (2) ML-based propagation classification operating on basic trees (Sec. 4.2); and (3) a graph-parser based approach for directly predicting edges between tokens (Sec. 4.3). While (1) and (2) may be used to construct "silver standard" enhanced UD graphs from gold trees, (3) is applicable in the automatic parsing setting only.

4.1 Modifications to Rule-based Converter

Based on the error analysis in Sec. 3.3, we modify the rule-based converter by Schuster and Manning (2016) as follows. In order to fix errors related to subject propagation in passive and imperative constructions, we take the conjunction dependent's morphological features into account. In the gold standard, the *Voice* feature is considered to be *active* by default. Hence, if the conjunction dependent does not have a *Voice* feature or is explicitly marked as *active*, an *nsubj:pass* dependency will be propagated as *nsubj*. Similarly, if it has the feature *Mood=Imp*, an *nsubj* link will not be propagated. Our second modification propagates common adjuncts of verbs as well (*obl*, *advmod*, and *advcl*). We maintain the rule from object propagation that a dependency is only propagated if the dependent comes after the potential target in the sentence. Finally, to handle multiple and nested coordinations, we iterate the converter's conjunction propagation function until the dependency graph does not change any more. This allows dependencies that result from propagation to be propagated themselves, retrieving links that would otherwise be missed.

4.2 Conjunction Propagation Classifiers

The core idea of ML-based conjunction propagation classifiers is to take a basic-layer tree and to decide for each incoming or outgoing dependency of the head of a coordinated phrase whether to propagate this dependency to the other coordinated item(s). We refer to the coordinated nodes as **conjunction head** and **conjunction dependent** and to the candidate governor/dependent of the second conjunct as the propagation **target**. In Figure 1, these three nodes correspond to "wrote," "published" and "1954" (or "PEREZ"/"book"), respectively. The output is a binary decision whether to propagate the given dependency or not. In addition to the features described below, we always provide the candidate dependency label and direction.

SVM-based Classifier. We re-implement the method proposed by Nyblom et al. (2013) using scikit-learn's SVC with a polynomial kernel of degree 2.[8] The features comprise morphological information about the tokens for the conjunction head/dependent and the target, as well as structural tree features extracted from the basic-layer tree. For a detailed description, see Appendix D.

Neural network classifier. We pass the sentence through the transformer-based neural language model RoBERTa (Liu et al., 2019) and extract the word embeddings for the first wordpiece tokens of the conjunction head, the conjunction dependent, and the propagation target. In addition, we use equivalents of the SVM tree features using learned embeddings or one-hot encodings (see Appendix D). The inputs are concatenated and fed to a multi-layer perceptron, which then outputs the binary decision whether to propagate the dependency or not. The multi-layer perceptron consists of two linear layers with hidden sizes 1500 and 500 respectively. We implement the model using Huggingface's Transformers library (Wolf et al., 2019). RoBERTa weights are not fine-tuned.

4.3 Graph-Parser Based Edge Prediction

In addition to the above approaches, we also evaluate a graph-parser based approach that predicts dependencies between tokens directly, i.e., which does not rely on a basic-layer tree. Our unfactorized architecture is similar to that of Grünewald and Friedrich (2020), i.e., our model predicts presence of edges and the corresponding labels in a single

---

[8] `https://scikit-learn.org`

step, treating nonexistence of an edge as simply another label (∅). As we focus on the dependencies involved in conjunctions, we do not require the parser's output to constitute valid graphs.

Embeddings for input tokens are generated by feeding gold tokens to the RoBERTa tokenizer and then running the resulting word-pieces through the RoBERTa-large model. We then generate an embedding $\mathbf{r}_i$ for the token at position i by forming a weighted sum of the hidden layers' embeddings at the positions corresponding to the first word-piece token of the original token as suggested by Kondratyuk and Straka (2019). Weights for this scalar mixture of layers are learned during training. Layers are randomly dropped during training to prevent the model from focusing on only a single layer.

For each input embedding $\mathbf{r}_i$, we create a head representation h_i^{head} and a dependent representation h_i^{dep} via two feed-forward neural networks:

$$\mathbf{h}_i^{head} = \text{FNN}^{head}(\mathbf{r}_i) \tag{1}$$

$$\mathbf{h}_i^{dep} = \text{FNN}^{dep}(\mathbf{r}_i) \tag{2}$$

For each ordered pair (i, j) of tokens, we feed their respective head and dependent representations to a biaffine classifier (Dozat and Manning, 2017) predicting logits $\mathbf{s}_{i,j}$ over the possible dependency labels. We use these logits to extract the probabilities $P(y_{i,j})$ for each label:

$$\text{Biaff}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{U} \mathbf{x}_2 + W(\mathbf{x}_1 \oplus \mathbf{x}_2) + \mathbf{b} \tag{3}$$

$$\mathbf{s}_{i,j} = \text{Biaff}\big(\mathbf{h}_i^{head}, \mathbf{h}_j^{dep}\big) \tag{4}$$

$$P(y_{i,j}) = \text{softmax}(\mathbf{s}_{i,j}) \tag{5}$$

$\mathbf{U}$, W and $\mathbf{b}$ in (3) are learned parameters; $\oplus$ denotes concatenation. The model is trained to minimize cross entropy loss w. r. t. the true dependency label between each pair of tokens. If a token is not assigned any head due to ∅ scoring highest for all other tokens, we assign the highest-scoring non-∅-relation and the corresponding head.

The model is simply trained to predict all link types in enhanced UD graphs. In the training section of the EWT corpus, we replace every sentence that contains a coordinated verb phrase with our manually corrected version of that sentence, or remove it from the corpus if it is one of the 927 conjunction sentences in the training section which we did not correct. For hyperparameter settings, see Appendix E.

5 Experiments

In this section, we describe our experiments on creating enhanced UD representations for coordinate constructions. Analogous to Nyblom et al. (2013), we measure precision, recall and F1 on enhanced links that are the result of propagation in coordinate constructions. For all experiments, we use gold sentence segmentation and tokenization, and evaluate on our manually corrected sentences from the dev and test sets of the EWT corpus.

5.1 Gold Standard Treebank Enhancing

We first address the research question of how to best generate enhanced representations for treebanks with gold standard basic annotations. We compare the following models: (1) an "Always" baseline, which simply propagates all incoming and outgoing links from the conjunction head to the conjunction dependent(s); (2) the rule-based converter by Schuster and Manning (2016) and the variations thereof we developed inspired by our corpus study; (3) our re-implementation of the SVM-based classifier by Nyblom et al. (2013); and (4) our neural-network (NN) based classifier. The latter uses AdamW (Loshchilov and Hutter, 2017) with a learning rate of 5e-5, a batch size of 1 and early stopping. Table 4 reports the results on the development and test sets of our manually verified conjunction dataset. The recall of the "Always" baseline is not at 100% because a small number of relations change their label during propagation, e.g., *nsubj→nsubj:pass*.

Rule-based conversion. We show results for successively adding components to the original converter (**RBC**). On the test set, adding propagation of non-core dependents and allowing several iterations increases recall and improves F1 by more than 2 points. On the dev set, in contrast, we do not observe these effects.[9] Adding our suggested passive/imperative fix surprisingly decreased performance. Analysis showed that the cases that our converter got wrong were caused by erroneous morphological feature annotations in the basic layer. In sum, our suggested improvements (**RBC2**) of heuristically propagating adjuncts (*obl*, *advmod*, *acl*) and allowing several resolution passes of the

---

[9] We assume that the reason is the presence of several informal-language sentences in the dev set that include multiple conjunctions (e.g., "etc. etc. etc."), whose annotation is unclear even in the basic gold standard.

converter seem to improve treebank enhancing, provided that the basic layer is correct.

ML-based conversion. Overall, the SVM and NN models show similar performance. As they perform already close to human agreement (see Table 2), further improvement may actually indicate overfitting. On the test set, the ML-based methods outperform the heuristic rule-based methods, surpassing the original converter by over 4 points F1. We conclude that learning structural rules based on actual gold standard data is more effective than hand-designing them. Differences on the dev set are less pronounced despite models being optimized on this data, again hinting to some qualitative differences between the two sets.

In order to determine which sources of information are most relevant, we perform ablation experiments for both classifiers. The features representing the candidate dependency label and the direction of the link are essential and kept in each case. Both the SVM and the NN classifiers draw most of their information from tree-based features. This effect is particularly pronounced for the SVM classifier, where performance drops by 10 to almost 20 points F1 when ommitting these features. The NN classifier's performance does not deteriorate as strongly under the same condition, indicating that some syntactic information can also be retrieved from contextualized word embeddings (see e.g., Tenney et al., 2019). Nonetheless, in most experiments, adding token features improves performance slightly, showing that they do contain important information for propagation decisions.

	Dev			**Test**		
	P	R	F	P	R	F
"Always" baseline	*23.1*	*99.6*	*37.5*	*28.0*	*99.6*	*43.7*
RBC	**94.8**	86.4	**90.4**	95.2	76.9	85.0
+ non-core deps	93.7	86.4	89.9	94.9	79.7	86.7
+ iteration (**RBC2**)	90.1	86.8	88.4	93.9	81.5	87.2
+ passive fix	91.7	85.3	88.4	**95.7**	78.6	86.3
SVM	87.6	87.9	87.8	93.4	85.4	**89.2**
- tree features	75.5	78.0	76.8	76.5	63.7	69.5
- token features	86.3	87.5	86.9	92.3	85.1	88.5
NN	87.0	87.9	87.4	92.0	**85.8**	88.8
- tree features	87.1	86.4	86.8	88.0	78.6	83.1
- token features	87.3	**88.3**	87.8	92.2	84.3	88.1

Table 4: **Predicting relation propagation for coordinate constructions on gold basic trees**. Precision, recall and F1 on propagated relations in the predicted vs. gold dependency graphs in our manually verified conjunction dataset. The gold dev and test sets contain 273 and 281 instances, respectively.

5.2 Propagating Conjunction Links in Automatic Parsing Setting

For the scenario of parsing from raw tokens, we compare two state-of-the-art parsers, StanfordNLP (Qi et al., 2018) and UDify (Kondratyuk and Straka, 2019), combined with the rule-based converter or ML-based conjunction propagators, and our graph-parser based edge predictor. The latter is trained on the subset of training sentences that either do not contain coordinated verb phrases or that were corrected by us. Hyperparameters and training settings are given in Appendix E.

Results for these experiments can be found in Table 5. The impact of the quality of the parsed basic dependencies is evident: Results are much better for the UDify parser (LAS F1 of 89.4 for basic dependencies on the EWT dev set) than for StanfordNLP (LAS F1 of 87.4). In the automatic setting, our heuristic extensions improve results compared to using the original converter, and there is no decrease in F1 on dev. As in the gold standard settings, ML-based extensions improve upon RBC on test, but not dev. Of the systems based on basic-layer tree parsers, RBC2 works best. However, performance of all pipeline systems show rather poor performance at or below an F1 of 70. Our graph-parser based edge predictor achieves by far the best results, outperforming all other models by a margin of over 7 points F1. This shows that in an automatic setting, most robust results are achieved by directly inducing dependency links between tokens, modeling conjunction only indirectly.

To estimate the impact of our corrections to the gold standard, we also train the graph parser on uncorrected data. The model trained on the corrected data has higher recall, but lower precision. This is expected to some extent as we introduce semantically motivated propagations of adjuncts, and we suspect that they may require a larger training set.

5.3 Discussion

The main insights comparing our experiments in the gold standard vs. the automatic parsing setting are as follows. Overall, our heuristic extensions for the rule-based converter are beneficial in both settings. In the gold setting, ML-based extensions lead to higher accuracy; when applying them on noisy parser output, they do not work well. However, using *one* end-to-end machine-learning model

	Dev			Test		
	P	R	F	P	R	F
Stanford+RBC	70.8	63.0	66.7	56.5	47.7	51.7
Stanford+RBC2	68.7	65.2	66.9	56.2	50.2	53.0
Stanford+SVM	64.3	65.2	64.7	54.7	49.8	52.1
Stanford+NN	64.3	65.2	64.7	54.4	50.2	52.2
UDify+RBC	72.8	67.8	70.2	71.8	58.0	64.2
UDify+RBC2	71.9	68.5	70.2	75.0	61.9	67.8
UDify+SVM	70.6	68.5	69.5	70.9	59.1	64.5
UDify+NN	69.9	68.9	69.4	70.4	60.1	64.9
GBP (orig. data)	**83.1**	74.0	78.3	**86.1**	66.2	74.8
GBP (our data)	82.3	**75.1**	**78.5**	82.5	**68.7**	**75.0**

Table 5: **Predicting relation propagation for coordinate constructions on parser output**. Otherwise same evaluation setup as in Table 4.

directly to generate enhanced representations for conjunctions outperforms the pipeline version. A possible reason for this might be that these models were all developed on gold data, while the graph-based parser does not rely on potentially wrong structural tree features and is also able to use internal confidence information for edges. Another advantage of the end-to-end model may stem from the fact that its training allows to leverage semantic information from training data of a larger number of dependency links, i.e., including those not occurring in coordinate constructions. This points to a promising future research direction, i.e, generating additional semi-artificial training data for conjunction propagation.

6 Conclusion and Outlook

We have presented a large-scale manually curated **dataset for conjunction propagation** in English. In contrast to previous work focusing on high-precision rule-based propagation, we propagate links in all cases that semantically suggest argument or adjunct sharing. In the gold standard treebank enhancing setting, we found **ML-based models** to outperform the de-facto standard rule-based converter by learning to exploit mostly structural features. However, one of our main insights is that neither rule-based nor ML-based classifiers work well on noisy parser output precisely because of this reliance on structural information. We propose to use a graph-parser based edge predictor instead and show that it outperforms pipeline-based models by a large margin. Our model reaches F1 scores between 0.75 and 0.78 with a precision of more than 0.82, a level of performance that may already be useful in downstream tasks.

Our models could be used for creating high-quality enhanced-level representations of conjunctions for the remaining English data, and could thus help in a UD community effort to continuously **improve the UD treebanks**. Future work also includes the study of conjunction propagation methods for **further languages**. Our in-depth study on English data provides several insights that we expect to be transferable cross-linguistically. First, conjunction propagation can to some extent be addressed using heuristic rules, but capturing the full semantic nature of the task requires manual annotation. Second, given appropriate training data, our machine-learning based approaches are also applicable to other languages.

In addition, it would be interesting to see if manually annotated data for coordinate constructions may be useful in natural language understanding tasks such as natural language inference (NLI). This is especially true for "stress test" datasets such as CONJNLI (Saha et al., 2020), which are designed to specifically test models' capabilities to process coordination.

Finally, as morphological features are generally important for this task, improving their automatic prediction (see e.g., Ramm et al., 2017; Myers and Palmer, 2019) as well as UD's gold standard seems to be a promising way to go. Our work has demonstrated the value of a linguistically motivated corpus study of a syntactic-semantic phenomenon, and shown that given manually curated data, rules for conjunction propagation can be learned effectively.

Acknowledgments

We would like to thank Sherry Tan and Johannes Hingerl for their support, as well as Jonas Kuhn, Heike Adel, Jannik Strötgen, and the anonymous reviewers for their useful comments regarding this work. In addition, we are grateful to the UD community, especially Nathan Schneider, for answering our questions.

References

Chiara Alzetta, Felice Dell'Orletta, Simonetta Montemagni, Maria Simi, and Giulia Venturi. 2018. Assessing the impact of incremental error detection and correction. a case study on the Italian universal dependency treebank. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 1–7, Brussels, Belgium. Association for Computational Linguistics.

Gosse Bouma, Djamé Seddah, and Daniel Zeman. 2020. Overview of the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, Seattle, US. Association for Computational Linguistics.

Mathieu Dehouck, Mark Anderson, and Carlos Gómez-Rodríguez. 2020. Efficient EUD parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 192–205, Online. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Stefan Grünewald and Annemarie Friedrich. 2020. RobertNLP at the IWPT 2020 shared task: Surprisingly simple enhanced UD parsing for English. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 245–252, Online. Association for Computational Linguistics.

Han He and Jinho D. Choi. 2020. Adaptation of multilingual transformer encoder for robust enhanced Universal Dependency parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 181–191, Online. Association for Computational Linguistics.

Johannes Heinecke. 2019. ConlluEditor: a fully graphical editor for universal dependencies treebank files. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 87–93, Paris, France. Association for Computational Linguistics.

Johannes Heinecke. 2020. Hybrid enhanced Universal Dependencies parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 174–180, Online. Association for Computational Linguistics.

Daniel Hershcovich, Miryam de Lhoneux, Artur Kulmizev, Elham Pejhan, and Joakim Nivre. 2020. Køpsala: Transition-based graph parsing via efficient training and effective encoding. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 236–244, Online. Association for Computational Linguistics.

Jenna Kanerva, Filip Ginter, and Sampo Pyysalo. 2020. Turku enhanced parser pipeline: From raw text to enhanced graphs in the IWPT 2020 shared task. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 162–173, Online. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Jonathan May and Jay Priyadarshi. 2017. SemEval-2017 task 9: Abstract Meaning Representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 536–545, Vancouver, Canada. Association for Computational Linguistics.

Skatje Myers and Martha Palmer. 2019. ClearTAC: Verb tense, aspect, and form classification using neural nets. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 136–140, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre, Paola Marongiu, Filip Ginter, Jenna Kanerva, Simonetta Montemagni, Sebastian Schuster, and Maria Simi. 2018. Enhancing universal dependency treebanks: A case study. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 102–107, Brussels, Belgium. Association for Computational Linguistics.

Jenna Nyblom, Samuel Kohonen, Katri Haverinen, Tapio Salakoski, and Filip Ginter. 2013. Predicting conjunct propagation and other extended Stanford dependencies. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 252–261, Prague, Czech Republic. Charles University in Prague, Matfyzpress, Prague, Czech Republic.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Anita Ramm, Sharid Loáiciga, Annemarie Friedrich, and Alexander Fraser. 2017. Annotating tense, mood and voice for English, French and German. In *Proceedings of ACL 2017, System Demonstrations*, pages 1–6, Vancouver, Canada. Association for Computational Linguistics.

Swarnadeep Saha, Yixin Nie, and Mohit Bansal. 2020. ConjNLI: Natural language inference over conjunctive sentences. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8240–8252, Online. Association for Computational Linguistics.

Sebastian Schuster, Éric Villemonte de La Clergerie, Marie Candito, Benoît Sagot, Christopher Manning, and Djamé Seddah. 2017. Paris and Stanford at EPE 2017: Downstream evaluation of graph-based dependency representations. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation (EPE 2017)*, pages 47–59.

Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English universal dependencies: An improved representation for natural language understanding tasks. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2371–2378, Portorož, Slovenia. European Language Resources Association (ELRA).

Sebastian Schuster, Joakim Nivre, and Christopher D. Manning. 2018. Sentences with gapping: Parsing and reconstructing elided predicates. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1156–1168, New Orleans, Louisiana. Association for Computational Linguistics.

Maria Simi and Simonetta Montemagni. 2018. Bootstrapping enhanced universal dependencies for italian. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253. CEUR-WS.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, and Kewei Tu. 2020. Enhanced Universal Dependency parsing with second-order inference and mixture of training data. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 215–220, Online. Association for Computational Linguistics.

Guillaume Wisniewski. 2018. Errator: a tool to help detect annotation errors in the universal dependencies project. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Appendix

A Detailed converter description

In this section, we describe the algorithm implemented by the converter proposed by Schuster and Manning (2016).[10]

For each *conj* relation, the converter decides whether links ending or starting at the conjunction head (`gov`) should be propagated to the conjunction dependent (`dep`):

1. Governors of `gov` are always propagated to `dep`, unless the relation is explicitly treated as an exception (e.g., *vocative*, *discourse*, or *root*).
2. Dependents of `gov` are propagated to `dep` as follows:
    (a) If the dependent is attached via *nsubj* or *csubj*, it is only propagated if `dep` does not already have a subject. If `dep` has an *aux:pass* dependent, the relation is propagated as *nsubj:pass* / *csubj:pass*.
    (b) If the dependent is attached via a non-subject core relation (*obj*, *iobj*, *ccomp*, or *xcomp*), it is propagated if and only if it comes after `dep` in the linear order of the sentence.
    (c) Non-core dependents (such as *obl*) are never propagated.

The algorithm is able to handle many syntactically ambiguous cases, provided the underlying basic dependencies have resolved the ambiguity correctly. Consider the sentence "She was reading or watching a movie." If "movie" is correctly attached as an object of the conjunction dependent "watching," it will not be propagated to "reading" in the enhanced representation.

B Inter-annotator agreement study

Detailed comparisons between the three annotators can be found in Table 6, Table 7, and Table 8. In our study, we consider only links that are part of the enhanced layer, but not of the basic layer. For each annotator, we count for each label how often it occurs as an incoming or outgoing relation of a conjunct (columns labeled with the annotator's ID). Formally, the set E^l_A is the set of enhanced-layer edges that are (i) not present in the basic layer and (ii) involved in conjunctions as incoming or outgoing links of the conjuncts, with label l marked by annotator A. We also count the overlap of links for pairs of annotators. Using these counts, we then compute precision, recall and F1, treating one annotator as the system and one as the gold standard. For instance, when treating A as the gold standard and B as the system, this leads to:

$$Precision_{BA} = \frac{|E^l_A \cap E^l_B|}{E^l_B} \qquad (6)$$

$$Recall_{BA} = \frac{|E^l_A \cap E^l_B|}{E^l_A} \qquad (7)$$

Note that when reversing this order, P and R are simply reversed, F1 stays the same.

The following numbers compare each annotator to the original gold standard (not in tables). For modifier clauses (*acl*, *advcl*) and adverbials (*advmod*), B was the most aggressive in propagating dependencies, adding 55 links in total for these labels, while A and C only added 39 and 32 links, respectively. While all annotators propagated *obl* dependencies roughly to the same extent, agreement was high between A and B but lower (F1 64-68%) between C and the others, indicating that there are more ambiguities among these dependencies as well. Annotator C is generally more conservative in propagating dependencies. This is reflected in the relatively low recall when comparing to the other annotators, as well as the lower overall number of added links (285 as compared to 309 for A and 312 for B).

---

[10] `https://github.com/stanfordnlp/CoreNLP/blob/master/src/edu/stanford/nlp/trees/ud/UniversalEnhancer.java`

	B	**A**	**A&B**	**P**	**R**	**F1**
acl	7	7	7	100.0	100.0	100.0
acl:relcl	12	8	8	66.7	100.0	80.0
advcl	24	17	17	70.8	100.0	82.9
advmod	10	5	4	40.0	80.0	53.3
amod	4	6	4	100.0	66.7	80.0
ccomp	12	13	12	100.0	92.3	96.0
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
nmod	8	8	8	100.0	100.0	100.0
nsubj	62	66	62	100.0	93.9	96.9
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	7	6	100.0	85.7	92.3
obj	26	25	25	96.2	100.0	98.0
obl	29	26	25	86.2	96.2	90.9
ref	5	5	5	100.0	100.0	100.0
xcomp	7	7	7	100.0	100.0	100.0
total	222	210	200	90.1	95.2	92.6

Table 6: **Agreement** of Annotator A vs. Annotator B on links involved in coordinate constructions in the enhanced layer. For P/R computation, A was treated as the gold standard and B as the system.

	C	**A**	**A&C**	**P**	**R**	**F1**
acl	7	7	7	100.0	100.0	100.0
acl:relcl	4	8	4	100.0	50.0	66.7
advcl	11	17	10	90.9	58.8	71.4
advmod	4	5	0	0.0	0.0	0.0
amod	4	6	4	100.0	66.7	80.0
ccomp	11	13	11	100.0	84.6	91.7
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
mark	2	0	0	0.0	0.0	0.0
nmod	6	8	6	100.0	75.0	85.7
nsubj	61	66	61	100.0	92.4	96.1
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	7	6	100.0	85.7	92.3
obj	25	25	23	92.0	92.0	92.0
obl	18	26	18	100.0	69.2	81.8
ref	2	5	2	100.0	40.0	57.1
xcomp	7	7	7	100.0	100.0	100.0
total	178	210	169	94.9	80.5	87.1

Table 7: **Agreement** of Annotator A vs. Annotator C on links involved in coordinate constructions in the enhanced layer. For P/R computation, A was treated as the gold standard and C as the system.

	C	**B**	**B&C**	**P**	**R**	**F1**
acl	7	7	7	100.0	100.0	100.0
acl:relcl	4	12	4	100.0	33.3	50.0
advcl	11	24	11	100.0	45.8	62.9
advmod	4	10	3	75.0	30.0	42.9
amod	4	4	4	100.0	100.0	100.0
ccomp	11	12	10	90.9	83.3	87.0
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
mark	2	0	0	0.0	0.0	0.0
nmod	6	8	6	100.0	75.0	85.7
nsubj	61	62	61	100.0	98.4	99.2
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	6	6	100.0	100.0	100.0
obj	25	26	24	96.0	92.3	94.1
obl	18	29	18	100.0	62.1	76.6
ref	2	5	2	100.0	40.0	57.1
xcomp	7	7	7	100.0	100.0	100.0
total	178	222	173	97.2	77.9	86.5

Table 8: **Agreement** of Annotator B vs. Annotator C on links involved in coordinate constructions in the enhanced layer. For P/R computation, B was treated as the gold standard and C as the system.

C Statistics on Treebank Modifications

As mentioned in Sec. 3.3, we made changes to the original treebank in 1,417 sentences containing coordinate verb phrases. While we focused on the dependency links starting or ending at conjuncts, we also fixed some additional errors that we spotted during this process. Table 9 gives statistics on these modifications (compare to Table 3, which includes only changes related to conjuncts).

label	#added	#removed	#sents	#total
acl	18	5	14	749
acl:relcl	15	5	9	5,086
advcl	39	11	36	2,055
advmod	46	4	37	4,426
amod	20	3	15	3,570
case	2	1	2	6,415
ccomp	12	16	13	1,003
det	2	1	2	6,448
nmod	32	2	24	3,279
nmod:poss	12	1	10	7,485
nsubj	194	41	144	7,815
nsubj:pass	9	21	25	1,381
nsubj:xsubj	35	20	31	8,758
nummod	3	0	3	814
obj	10	10	14	4,673
obl	78	5	56	5,478
punct	3	3	3	9,508
ref	21	2	16	348
xcomp	14	6	12	758
all	565	157	466	80049

Table 9: Statistics of modifications made to 1,417 sentences of the EWT, including both basic and enhanced layer. **#sents** reports the number of sentences in which the respective reported changes were made, **#total** reports the number of occurrences of the label in the original treebank.

D ML-based classifiers: features

Table 10 lists the features used in our SVM and NN models. Token features are extracted for conjunction head, conjunction dependent, and propagation target each. In addition to the listed features, we also experimented with including lemmas and POS tags, but did not find them to be useful in our ablation experiments.

E Graph-based edge predictor: Training Setup

Label lexicalization. At training time, we only use a limited label set of 56 labels where lexical material is replaced with placeholders, such as *obl:[case]*. At prediction time, we retrieve the missing lexical material from the dependency graph in a rule-based fashion. In the simplest case, this means simply substituting the word form of the dependent of the required type (e.g., a *case* relation). In conjunctions, the token in question may not have its own dependent of the correct type, instead "inheriting" if from its conjunction head. In that case, we retrieve the lexical material from the conjunction head's dependent.

Hyperparameters We perform only a minimal amount of hyperparameter tuning, mostly sticking with the values used by Kondratyuk and Straka (2019). One notable exception is the training regime, where we found low batch size and the AdamW optimizer to yield the best results. The full hyperparameter configuration can be found in Table 11.

RoBERTa embeddings	
Embeddings dimension	1024
Token mask probability	0.15
Layer dropout	0.1
Hidden dropout	0.2
Attention dropout	0.2
Output dropout	0.5
Biaffine classifier	
Hidden size	1024
Dropout	0.33
AdamW Optimizer	
Batch size	5
Learning rate	$5e^{-6}$
β_1, β_2	0.9, 0.999
Weight decay	0.0

Table 11: Hyperparameters for our graph-based parser.

Feature name	Description	SVM	NN
Instance features			
dependency label	label of candidate link	one-hot	50-dim. embedding
incoming/outgoing	whether the dependency being propagated is an outgoing or incoming link at the conjunction head	one-hot	50-dim. embedding
midrule ***Token features***			
morphological features	values of the *Number*, *Person*, *VerbForm*, and *Voice* features	one-hot	-
contextualized word embeddings	word embeddings as generated by the RoBERTa-base model	-	768-dim. embedding
Tree features			
linear dependency direction	whether the linear direction of the candidate dependency is the same as for the dependency being propagated (both-left, both-right, or differing-directions)	one-hot	50-dim. embedding
existing dependency	whether the conjunction dependent already has a dependency of this type (only relevant for outgoing links)	one-hot	50-dim. embedding
outgoing dependencies (head)	set of outgoing dependencies of the conjunction head	one-hot	one-hot
outgoing dependencies (dep)	set of outgoing dependencies of the conjunction dependent	one-hot	one-hot
# coord.-items	number of items in the coordination	one-hot	scalar

Table 10: Description of feature sets used in ML-based conjunction propagation models.

Ellipsis Resolution as Question Answering: An Evaluation

Rahul Aralikatte[1], **Matthew Lamm**[2], **Daniel Hardt**[3], and **Anders Søgaard**[1]

[1]University of Copenhagen, [2]Stanford University
[3]Copenhagen Business School

Abstract

Most, if not all forms of ellipsis (e.g., '*so does* Mary') are similar to reading comprehension questions ('*what does* Mary do'), in that in order to resolve them, we need to identify an appropriate text span in the preceding discourse. Following this observation, we present an alternative approach for English ellipsis resolution relying on architectures developed for question answering (QA). We present both single-task models, and joint models trained on auxiliary QA and coreference resolution datasets, clearly outperforming the current state of the art for Sluice Ellipsis (from 70.00 to 86.01 F_1) and Verb Phrase Ellipsis (from 72.89 to 78.66 F_1).

1 Introduction

Ellipsis resolution is a hard, open problem in NLP, and an important source of error in machine translation, question answering, and dialogue understanding (Vicedo and Ferrández, 2000; Dzikovska et al., 2009; Chung and Gildea, 2010; Macketanz et al., 2018; Petrn Bach Hansen and Sgaard, 2020). There are no large annotated text corpora for this phenomenon, even for English, and we only have annotations for a subset of the known ellipsis constructions. Since annotation is expensive and cumbersome, any synergies with existing NLP tasks could be useful and enable us to leverage auxiliary data when learning models for ellipsis resolution.

This paper presents a simple yet strong approach to ellipsis resolution based on a straightforward observation, depicted in Figure 1, that ellipsis resolution can be converted to a QA problem. Ellipsis and questions put in focus *referentially dependent* expressions (Carlson, 2006), or free variables (Partee, 1978), that need to be resolved in order to comprehend the discourse. For similar observations about different tasks, see McCann et al. (2018) and Gardner et al. (2019).

This straightforward observation leads us to suggest treating different forms of ellipsis resolution – and later, as an auxiliary task, coreference resolution – as a QA problem, and to apply state-of-the-art architectures for QA to ellipsis resolution tasks, as well as to experiment with using training data for QA and coreference resolution to improve our new ellipsis resolution models.

Sluice Ellipsis

Context: … But the way things are structured now you have to set aside your ego to make things happen. **The whole thing worked out**. I don't know **how**, but it did. Both sides had to work to make it happen …

Question: I don't know how, but it did.

Answer: The whole thing worked out

Verb Phrase Ellipsis

Context: … It has to be considered as an additional risk for the investor," said Gary P. Smaby of Smaby Group Inc., Minneapolis. "Cray Computer will be a concept stock," he said. "You either **believe Seymour can do it** again or you **don't** …

Question: You either believe Seymour can do it again or you don't.

Answer: believe Seymour can do it again

Figure 1: Examples of Sluice Ellipsis and Verb Phrase Ellipsis, represented as "questions" about their associated contexts. Wh-phrases and auxiliary verbs are marked in red and elided phrases are marked in blue.

Contributions We cast ellipsis as a QA problem, enabling us to induce models for it using neural architectures originally developed for QA. Applying these architectures out of the box enables us to establish strong results[1] for ellipsis resolution tasks, improving significantly over previous work. Using the same architecture for the different ellipsis resolution tasks, as well as for QA and coreference resolution, enables us to explore syn-

---

[1] Though we report state-of-the-art results for both sluice and verb phrase ellipsis, we consider these models as strong baselines for future research as they are obtained purely using existing methods.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 810–817
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

ergies between the tasks, and we show that training joint models on these tasks leads to even better performance.

2 Methodology

In this section, we briefly describe the various datasets used for training, and explain how they are converted into QA format. We then move on to the choice of model architectures and the reasoning behind their selection.

Sluice Ellipsis For training and evaluation of Sluice Ellipsis resolution models, we use the corpus introduced by Anand and McCloskey (2015), which contains 3,103 annotated examples of embedded sluices, collected from the New York Times section of the English Gigaword corpus. Since the annotators were free to paraphrase the antecedent, in some cases, a string match on the context does not return antecedent span indices. To ensure a fair comparison, we follow previous work (Rønning et al., 2018), which is also the current state-of-the-art, in ignoring these instances, and use their split for training, development and testing.

Verb Phrase Ellipsis Bos and Spenader (2011) provide Verb Phrase (VP) Ellipsis annotations for the WSJ part of the Penn Treebank. All 25 sections were annotated, and we follow them in using sections 0-19 for training, and 20-24 for testing. We further hold out sections 18-19 from the training data for development. This also enables to us compare our results directly with the current state-of-the-art for VP Ellipsis (Zhang et al., 2019).

Coreference Resolution For coreference resolution, which we use as an auxiliary task, we train and evaluate on two corpora: (i) the English portion of the OntoNotes 5.0[2] corpus with the standard data split used in the CoNLL-2012 shared task (Pradhan et al., 2012), and (ii) the WikiCoref corpus (Ghaddar and Langlais, 2016), which contains annotations of 30 documents from the English Wikipedia. From this dataset, we use 22 documents for training, 4 documents for development, and 4 for testing.

QA We also use SQuAD v1.1 (Rajpurkar et al., 2016) as an auxiliary reading comprehension dataset.

Task	Train	Dev	Test	ACL
	ELLIPSIS			
Sluice Ellipsis	1.4k	480	992	351
VP Ellipsis	264	20	78	984
	AUXILIARY			
OntoNotes	153k	18.8k	19.5k	463
WikiCoref	5.6k	630	638	2.2k
SQuAD	87.6k	10.6k	-	117

Table 1: QA pair counts and average context lengths (ACL) for different datasets, after conversion

Data Conversion For converting the various datasets into the QA format of `<context, question, answer>` triples, we perform a simple restructuring as shown in Figure 1. We consider the entire document as the context; the sentence in which the ellipsis/mention is present becomes the question, and the antecedent/entity becomes the answer. In case of coreference resolution, where a single sentence can have n mentions, we create n questions where every question is the same sentence with a different mention $i \in \{1 \ldots n\}$ marked for resolution with `<ref>` and `</ref>` tags. Table 1 shows the number of QA pairs created from each dataset and the average number of words in their contexts.

QA Architectures Generally, QA models have two main components: (i) an encoder module which learns to represent the question and its context, and (ii) a span selection module which predicts the start and end span indices of the answer if it is present in the context. In this work, we present experiments with three diverse models which take entirely different approaches to build the encoder module: (i) DrQA (Chen et al., 2017), with an LSTM encoder, (ii) QANet (Yu et al., 2018), with a CNN encoder, and (iii) BERT (Devlin et al., 2019), with a (pretrained) transformer encoder. We use the three different models to show that the between-task synergies are relatively robust across architectures; but one architecture (BERT) is clearly superior to the others and will be the standard baseline we propose for future research.[3]

---

[2] `https://catalog.ldc.upenn.edu/LDC2013T19`

[3] Note that there are many differences between these architectures; not only the encoder networks. The number of parameters differ, and BERT is pre-trained on large volumes of data. Our purpose here is not comparing strategies, but simply showing that synergies can be seen across all architectures. For more details, see Appendix B.

TASK	SOTA	SINGLE TASK			JOINT		
		DRQA	QANET	BERT	DRQA	QANET	BERT
Sluice Ellipsis	70.00 (Rønning et al., 2018)	**77.48**	**75.70**	**<u>85.10</u>**	**80.17**	**77.11**	**<u>86.01</u>**
VP Ellipsis	72.89 (Zhang et al., 2019)	62.86	1.93	**76.42**	63.54	22.49	**<u>78.66</u>**

Table 2: Ellipsis resolution scores are token-level F_1. Bold-faced results are better than the previous state-of-the-art; underlined results are the new state-of-the-art. When evaluated, our best joint architecture scores 72.31 on OntoNotes and 65.30 on WikiCoref (macro-averages of MUC, B^3, and $CEAF_{\phi_4}$ scores). See Appendix C.2 for why these numbers are not directly comparable to previously reported coreference resolution results in literature.

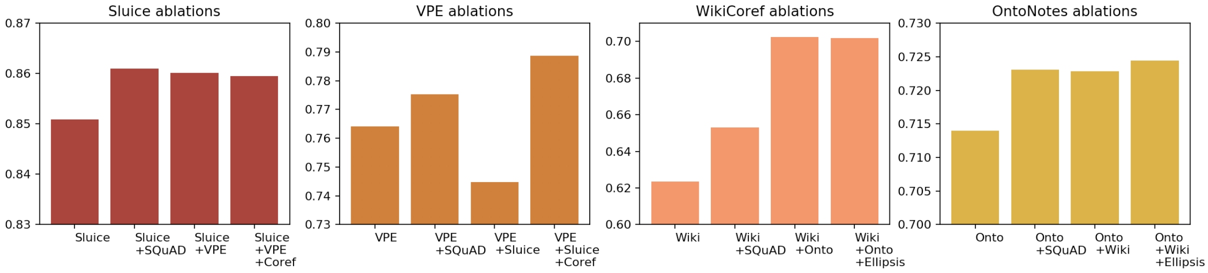


Figure 2: Dataset ablations (F_1)

3 Experiments & Results

We conduct two sets of experiments: (i) the SINGLE-TASK experiments, in which we train and evaluate separate models for the two ellipsis resolution tasks; and (ii) the JOINT modelling experiments, where we train on the best possible combination of ellipsis resolution, coreference resolution and QA data, as determined on the validation set. The results can be seen in Table 2.[4]

Single-Task Setup The SINGLE-TASK DrQA model improves the state-of-the-art on sluice ellipsis by 7.48 F_1. The SINGLE-TASK QANet model also improves the state-of-the-art on sluice ellipsis by 5.7 F_1, but fails to learn anything meaningful for VP ellipsis. We hypothesise this is due to the fact that 264 training examples are not enough to train the model's large stack of encoder blocks from scratch.

The SINGLE-TASK BERT model achieves state-of-the-art results in both the ellipsis datasets with absolute error reductions of 50.33% (Sluice Ellipsis) and 13.02% (VP Ellipsis). Interestingly, it also achieves a 17.10% error reduction over the best previously reported results on WikiCoref, but see Appendix C.2 for why such a direct comparison of numbers is not entirely fair.

Joint Setup The JOINT models always perform on-par with, or better than the SINGLE-TASK models. In this setup, the BERT models beat the previous state-of-the-art for both Sluice and VP Ellipsis with 53.37% and 21.28% absolute error reductions respectively.

4 Dataset ablations

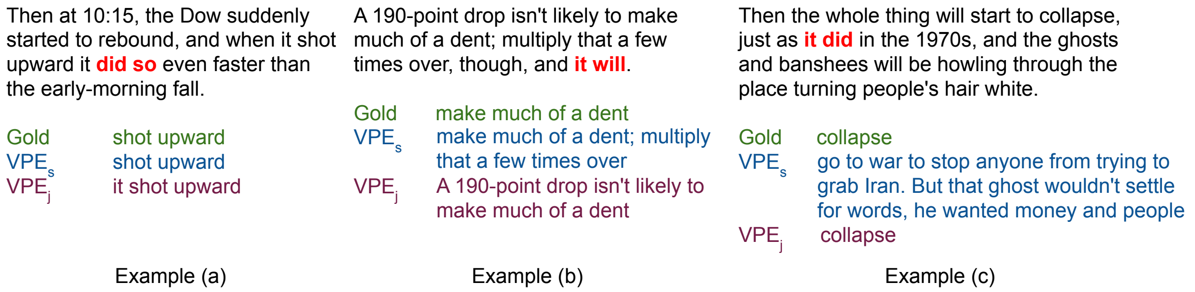
We determine the best task combinations on held-out validation data for each ellipsis resolution task.[5] For Sluice Ellipsis, the best results are obtained by training the models on a combination of Sluice and VP Ellipsis data. For VP Ellipsis, the best performance is attained when the models are trained with a combination of all datasets. When training a model for a particular task, we sample auxiliary data from other datasets to match the size of the main task's dataset. For each dataset, the variations in its F_1 scores of the best performing architecture when combined with other datasets are shown in Figure 2. The most interesting findings from these ablations are mentioned below.

When the two ellipsis datasets are combined, the overall performance of the models increase for both tasks by around 1% each. This shows that the two types of ellipsis are similar, and that when learning ellipsis resolution models, there is considerable synergy between the two resources. If we add subsampled coreference data when training these models, the Verb Phrase Ellipsis models gain up to 2.9%. One possible explanation

---

[4] The reported results are the average of three independent runs with different random seeds.

[5] These ablations are performed on the best performing (BERT) model.



Then at 10:15, the Dow suddenly started to rebound, and when it shot upward it **did so** even faster than the early-morning fall.

Gold	shot upward
VPE_s	shot upward
VPE_j	it shot upward

Example (a)

A 190-point drop isn't likely to make much of a dent; multiply that a few times over, though, and **it will**.

Gold	make much of a dent
VPE_s	make much of a dent; multiply that a few times over
VPE_j	A 190-point drop isn't likely to make much of a dent

Example (b)

Then the whole thing will start to collapse, just as **it did** in the 1970s, and the ghosts and banshees will be howling through the place turning people's hair white.

Gold	collapse
VPE_s	go to war to stop anyone from trying to grab Iran. But that ghost wouldn't settle for words, he wanted money and people
VPE_j	collapse

Example (c)

Figure 3: Selected gold and predicted antecedent spans from SINGLE-TASK Verb Phrase Ellipsis (VPE_s in figure) and JOINT Verb Phrase Ellipsis (VPE_j in figure) models.

could be more similarities between noun phrases and verb phrases, than between noun phrases and the sentences that are elided in Sluice Ellipsis resolution.

5 Error Analysis

We now look at some errors made by our best performing models. First, we compare the errors made by our SINGLE-TASK and JOINT Sluice Ellipsis resolution models before moving on to VP Ellipsis.[6]

Sluice Ellipsis The JOINT Sluice Ellipsis results improve modestly over the SINGLE-TASK Sluice Ellipsis results. This is noteworthy, since the added VP Ellipsis data is quite small compared to the size of the sluice data. These models consistently select an antecedent of the right syntactic form, which is normally a complete sentence. Many of the errors consist of empty outputs: SINGLE-TASK Sluice Ellipsis produces 58 empty outputs, while JOINT Sluice Ellipsis produces 63. Another source of error is discontiguous antecedents. It is not unusual for the gold antecedent to be a discontiguous span (Donecker, 1996), but our models are not permitted to produce such antecedents, so these cases will always be a source of error.

All the systems have problems when the antecedent follows the ellipsis, as in the following example: *I don't know why, but they seem to need a story.* We also compared the right and left periphery scores of sluices, and found better results predicting the right periphery: for SINGLE-TASK Sluice Ellipsis, there were 678 matches on the left edge, and 733 on the right edge; for JOINT Sluice Ellipsis, there were 703 left matches and 734 right matches.

Verb Phrase Ellipsis The SINGLE-TASK VP models trained with just VP Ellipsis data improves on the current state of the art, and further improvement is observed when trained on auxiliary data, especially the Sluice Ellipsis resolution dataset. While the JOINT VP Ellipsis model is generally better than the SINGLE-TASK model, joint training with Sluice Ellipsis resolution data also seems to introduce unfortunate biases. While the SINGLE-TASK model always selects antecedents of the right syntactic form, i.e., verb phrases, the JOINT model may select sentential antecedents. See examples in Figure 3.

In Example (a), the JOINT VP model incorrectly includes the subject *it*, presumably because the sluice data includes complete sentences as antecedents. Similarly in Example (b) – though the SINGLE-TASK model correctly chooses an antecedent beginning with the verb *make*, it continues with additional material that does not form a coherent antecedent. The JOINT result is also incorrect, but note that it consists of the complete sentence containing the correct VP antecedent. Example (b) presents the advantages and disadvantages of the joint ellipsis training data. While the two types of ellipsis require antecedents of different forms, they have similar requirements in terms of where in the context the antecedent is to be found. Example (c) further supports this point. Here the JOINT result is perfect, while the SINGLE-TASK result finds an antecedent that is in the wrong part of the discourse. The SINGLE-TASK model is slightly better with left periphery matches than right: we found 58 left and 55 right matches. This is reversed with the JOINT model, with 54 left and 60 right matches.

---

[6] We also briefly discuss how coreference resolution benefits from synergies with ellipsis in Appendix C.1.

6 Related Work

We are not the first to use question answering to redefine a set of tasks. Recently, He et al. (2015) showed that semantic role labeling annotations could be solicited by asking simple questions that implicitly target predicate-argument relations in a sentence. Parallel to our work, Hou (2020) cast bridging anaphora resolution as question answering based on context. Wu et al. (2020) and Li et al. (2020) also reformulate coreference resolution and named entity recognition as QA. In the realm of re-framing relation extraction as a QA problem, Levy et al. (2017) and Abdou et al. (2019) create monolingual and multilingual template based QA datasets respectively, which yield relation extraction models which were better at generalizing in the zero-shot setting. Extending this idea, McCann et al. (2018) introduced the DecaNLP challenge, which casts 10 core tasks in NLP as question-answering problems. Similar to our work, their architecture jointly learns across all of these tasks. DecaNLP includes pronoun resolution, a subset of coreference resolution, but it does so only on a small, hand-crafted dataset; it does not address ellipsis.

Limitations of our approach One limitation of our approach is that, like most previous work, we assume ellipsis and coreference resolution amount to finding antecedent spans that corefer with the target mention. This is not always the case; the elided material can: (i) have extra-linguistic antecedents, and (ii) refer to something that is contextually implied.

7 Conclusion

We present strong models for Sluice and Verb Phrase ellipsis resolution problems, by reformulating them as machine reading comprehension problems, significantly outperforming the previously best reported results. We also empirically show that training these models jointly and with auxiliary data from coreference resolution and question-answering further improves their performance. Our code is publicly available at https://github.com/rahular/ellipsis-baselines.

8 Acknowledgements

We thank the reviewers for their valuable feedback. Rahul Aralikatte and Anders Søgaard are funded by a Google Focused Research Award.

References

Mostafa Abdou, Cezar Sas, Rahul Aralikatte, Isabelle Augenstein, and Anders Søgaard. 2019. X-WikiRE: A large, multilingual resource for relation extraction as machine comprehension. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 265–274, Hong Kong, China. Association for Computational Linguistics.

Pranav Anand and Jim McCloskey. 2015. Annotating the implicit content of sluices. In *Proceedings of The 9th Linguistic Annotation Workshop*, pages 178–187, Denver, Colorado, USA. Association for Computational Linguistics.

Rajendra Banjade, Vasile Rus, and Nobal B. Niraula. 2015. Using an implicit method for coreference resolution and ellipsis handling in automatic student answer assessment. In *FLAIRS*.

Johan Bos and Jennifer Spenader. 2011. An annotated corpus for the analysis of vp ellipsis. *Lang. Resour. Eval.*, 45(4):463–494.

Greg Carlson. 2006. Anaphora. In *Encyclopedia of Cognitive Science*. Wiley.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Tagyoung Chung and Daniel Gildea. 2010. Effects of empty categories on machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 636–645, Cambridge, MA. Association for Computational Linguistics.

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

Paul G. Donecker. 1996. Subdeletion in verb phrase ellipsis. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 348–350, Santa Cruz, California, USA. Association for Computational Linguistics.

Myroslava Dzikovska, Charles Callaway, Elaine Farrow, Johanna Moore, Natalie Steinhauser, and Gwendolyn Campbell. 2009. Dealing with interpretation errors in tutorial dialogue. In *Proceedings of the SIGDIAL 2009 Conference*, pages 38–45, London, UK. Association for Computational Linguistics.

Paul Elbourne. 2013. *Definite descriptions*, volume 1. Oxford Studies in Semantics.

Matt Gardner, Hannaneh Hajishirzi, Alon Talmor, and Sewon Min. 2019. Question answering is a format; when is it useful?

Abbas Ghaddar and Philippe Langlais. 2016. Wikicoref: An English coreference-annotated corpus of Wikipedia articles. In *LREC*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Yufang Hou. 2020. Bridging anaphora resolution as question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1428–1438, Online. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Chuan-Jie Lin, Chien-Wei Pao, Yen-Heng Chen, Chi-Ting Liu, and Hui-Huang Hsu. 2016. Ellipsis and coreference resolution in a computerized virtual patient dialogue system. *Journal of Medical Systems*, 40.

Vivien Macketanz, Eleftherios Avramidis, Aljoscha Burchardt, and Hans Uszkoreit. 2018. Fine-grained evaluation of German-English machine translation based on a test suite. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 578–587, Belgium, Brussels. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Jason Merchant. 2001. *The syntax of silence: Sluicing, islands, and the theory of ellipsis*. Oxford University Press on Demand.

Barbara H. Partee. 1978. Bound variables and other anaphors. In *Theoretical Issues in Natural Language Processing-2*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Victor Petrn Bach Hansen and Anders Sgaard. 2020. What do you mean why?: Resolving sluices in conversations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7887–7894.

Paul M. Postal. 1966. On so-called pronouns in english. In F. Dinneen, editor, *Nineteenth Monograph on Language and Linguistics*. Georgetown University Press.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, CoNLL '12, pages 1–40, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Ola Rønning, Daniel Hardt, and Anders Søgaard. 2018. Sluice resolution without hand-crafted features over brittle syntax trees. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, page 236241. Association for Computational Linguistics.

José L. Vicedo and Antonio Ferrández. 2000. Importance of pronominal anaphora resolution in question answering systems. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 555–562, Hong Kong. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *ICLR (Poster)*. OpenReview.net.

Weinan Zhang, Yue Zhang, Yuanxing Liu, Donglin Di, and Ting Liu. 2019. A neural network approach to verb phrase ellipsis resolution. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7468–7475.

A Similarity between Ellipsis and Coreference Resolution

Linguists have long pointed out deep links among different forms of ellipsis, as well as between ellipsis and pronominal anaphora. For example, Merchant (2001) presents a unified account of ellipsis phenomena within a minimalist syntactic framework, and theorists such as Postal (1966) and Elbourne (2013) go so far as to argue that pronouns are also elliptical forms. The exact nature of the connections between ellipsis and anaphoric constructions remains a subject of controversy among linguists. However it is clear that there are rooted connections, and in our view these connections represent potential areas to be exploited with forms of knowledge transfer among datasets of different types.

Typically in NLP, ellipsis and coreference have been treated as distinct tasks. Possible exceptions include Lin et al. (2016), who present a rule-based, feature-rich system for handling ellipsis and coreference in Chinese medical dialogues, but the synergy between the two subsystems is limited; and Banjade et al. (2015), who reduce ellipsis and coreference to problems of alignment to an auxiliary text implicitly describing the universe of the dialogue in question.

B QA Models

We briefly describe the architectures of the QA models below. All experiments are conducted on a single 12 GB GPU. For all models, we use the hyperparameter values recommended in their respective papers.

DrQA The Document Reader component of DrQA consists of a context and a question encoder followed by two span prediction classifiers. The context encoder is a multi-layer bidirectional LSTM (Hochreiter and Schmidhuber, 1997) which takes in word embeddings (Pennington et al., 2014, GloVe), similarity based features (whether the token appears in the question in it's original, lowercase or lemma form), and other token level features (positional tags, named entities and term frequency) as input. The concatenation of each layer's hidden units is used as the context vector. The question encoder is another LSTM which takes word embeddings as input and combines the resulting hidden units using a simple attention mechanism to form the question vector. A bilinear term which captures the similarities between context and question vectors is used to combine the two vectors and the resulting vector used as input to the span prediction classifiers. The two classifiers predict the start and the end span respectively and are trained independently.

QANet In QANet, each encoder layer is a stack of depthwise separable convolutions followed by a multi-head self-attention mechanism placed inside a residual block. Initially, words in the context and question are embedded using a combination of GloVe and character embeddings. They are then contextualized individually with an encoder block. The representations are then passed through a context-query attention layer to obtain a combined representation of the context and question. This is further passed through three encoding blocks before feeding it into a classifier for predicting the answer spans.

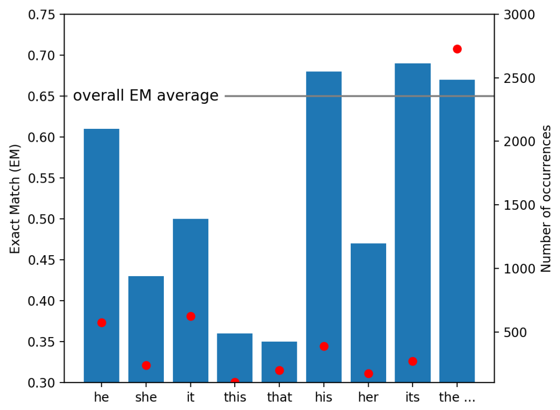


Figure 4: Exact match percentage (bars) and number of occurrences (dots) of referential forms in OntoNotes

BERT We use the pre-trained BERT$_{BASE}$ uncased model to encode questions and their contexts. It has 12 Transformer blocks, 12 self-attention heads, and a hidden size of 768. Word piece tokenization (Wu et al., 2016) is performed, both on the context paragraph and the question. The boundaries of the two sequences are marked by dummy symbols. The context and the question are joined with a $[SEP]$ token in between, and the $[CLS]$ token is prepended at the beginning to form the input. The representation of the $[CLS]$ token is fed into a single-layer MLP with 2 outputs which is used to predict the span indices.[7]

C Coreference Resolution

In this section, we analyse the best performing coreference models and discuss why they cannot be compared with other works in literature.

C.1 Error Analysis

The JOINT OntoNotes model improves a little over the SINGLE-TASK counterpart. Here we examine specific referential forms in OntoNotes (WikiCoref has similar traits), as shown in Figure 4. In general, performance is better on frequent pronouns – e.g., 'he' over 'she', 'this' and 'that'. An exception to this is that 'it' is less accurate, but more frequent than 'he'. It is notable that the possessive pronouns ('his', 'her', 'its') are all more accurate than their nominative counterparts ('he', 'she', 'it'), perhaps because they tend to have a closer connection to their antecedents. Overall, the single-word referential forms are less accurate than multiple-word forms. For example, definite descriptions (forms beginning with 'the') are more accurate than any of the single-word forms, with the exception of 'its'. We speculate that multi-word forms provide more specific information, thus limiting the set of potential antecedents. It is also interesting to break down error by the grammatical gender of the pronouns. Male pronouns generally tend to be more accurate than their female counterparts. Antecedents of 'he' and 'his' are matched 20% more frequently than for 'she' and 'her'. This is probably due to an unfortunate bias in OntoNotes, where female pronouns are 50% rarer than male pronouns.

C.2 Result Comparability

Converting coreference into QA fundamentally changes the coreference resolution problem: It, on the one hand, makes the coreference resolution problem harder, in that we require the identification of a specific antecedent span, rather than any mention in the entity chain; on the other hand, the problem becomes easier by providing the bracketing of the mention that needs to be resolved. Due to these differences, it is not possible to directly compare our results with others in literature. For analysis, to make our results more comparable with Lee et al. (2018), we provided their model with the bracketing of the mentions and considered the first mention to be the antecedent. This way we can reinterpret their clusters as question-answer pairs and do not penalize them for mention bracketing errors, only considering pairs where they correctly identify mentions. Note this gives their model an advantage over ours, as their model considers multiple sources of evidence for inferring the coreference links, and gets to pick the subset of data on which the models are compared. On OntoNotes, in this setting, and after pruning around $7,358$ mentions Lee et al. (2018) bracketed wrongly, their new average F$_1$ score is 75.9. Our performance on the same subset of the data is 72.1. Upon manual inspection, we see the model in Lee et al. (2018) has a strong bias favoring nominal antecedents, whereas our model is more likely to predict clausal antecedents. On WikiCoref, our model remains better than the previous state of the art by some margin, with an F$_1$ of 69.2 over 43.6.

---

[7] We use the implementation detailed in Wolf et al. (2020).

Jointly Improving Language Understanding and Generation with Quality-Weighted Weak Supervision of Automatic Labeling

†Ernie Chang, †Vera Demberg, *Alex Marin
†Dept. of Language Science and Technology, Saarland University
*Microsoft Corporation, Redmond, WA
cychang@coli.uni-saarland.de

Abstract

Neural natural language generation (NLG) and understanding (NLU) models are data-hungry and require massive amounts of annotated data to be competitive. Recent frameworks address this bottleneck with generative models that synthesize weak labels at scale, where a small amount of training labels are expert-curated and the rest of the data is automatically annotated. We follow that approach, by automatically constructing a large-scale *weakly-labeled data* with a fine-tuned GPT-2, and employ a semi-supervised framework to jointly train the NLG and NLU models. The proposed framework adapts the parameter updates to the models according to the estimated label-quality. On both the E2E and Weather benchmarks, we show that this weakly supervised training paradigm is an effective approach under low resource scenarios with as little as 10 data instances, and outperforming benchmark systems on both datasets when 100% of training data is used.

1 Introduction

Natural language generation (NLG) is the task that transforms meaning representations (MR) into natural language descriptions (Reiter and Dale, 2000; Barzilay and Lapata, 2005); while natural language understanding (NLU) is the opposite process where text is converted into MR (Zhang and Wang, 2016). These two processes can thus constrain each other – recent exploration of the duality of neural natural language generation (NLG) and understanding (NLU) has led to successful semi-supervised learning techniques where both labeled and unlabeled data can be used for training (Su et al., 2020; Tseng et al., 2020; Schmitt and Schütze, 2019; Qader et al., 2019; Su et al., 2020).

Standard supervised learning for NLG and NLU depends on the access to labeled training data – a major bottleneck in developing new applications.

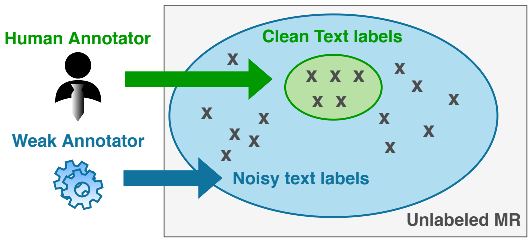


Figure 1: **Training scenario**: Each × represents a labeled data instance. The goal is to learn both from few human-labeled instances (inner) and large amounts of weakly labeled data (outer).

In particular, neural methods require a large annotated dataset for each specific task. The collection process is often prohibitively expensive, especially when specialized domain expertise is required. On the other hand, learning with weak supervision from noisy labels offers a potential solution as it automatically builds imperfect training sets from low cost labeling rules or pretrained models (Zhou, 2018; Ratner et al., 2017; Fries et al., 2020). Further, labeled data and large unlabeled data can be utilized in semi-supervised learning (Lample et al., 2017; Tseng et al., 2020), as a way to *jointly* improve both NLU and NLG models.

To this end, we target a weak supervision scenario (shown in Figure 1) consisting of small, high-quality expert-labeled data and a large set of unlabeled MR instances. We propose to expand the labeled data by automatically annotating the MR samples with noisy text labels. These noisy text labels are generated by a *weak annotator*, which is built upon recent works that directly fine-tune GPT-2 (Radford et al., 2019) on joint meaning representation (MR) and text (Mager et al., 2020; Harkous et al., 2020). Then, we jointly train the NLG and NLU models in a two-step process with semi-supervised learning objectives (Tseng et al., 2020). First, we use pretrained models to estimate quality scores for each sample. Then, we down-weight the

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 818–829
April 19 - 23, 2021.

loss updates in the back-propagation phase using the estimated quality scores. This way, the models are guided to avoid mistakes of the *weak annotator*.

On two benchmarks, E2E (Novikova et al., 2017b) and Weather (Balakrishnan et al., 2019), we utilize varying amount of labeled data and show that the framework is able to successfully learn from the synthetic data generated by *weak annotator*, thereby allowing jointly-trained NLG and NLU models to outperform other baseline systems.

This work makes the following contributions:

1. We propose an automatic method to overcome the lack of text labels by using a fine-tuned language model as a *weak annotator* to construct text labels for the vast amount of MR samples, resulting in a much larger labeled dataset.
2. We propose an effective two-step weak supervision using the dual mutual information (DMI) measure which can be used to modulate parameter updates on the weakly labeled data by providing quality estimates.
3. We show that the approach can even be used to improve upon baselines with 100% data to establish new state-of-the-art performance.

2 Related Work

Learning with Weak Supervision. Learning with weak supervision is a well-studied area that is popularized by the rise of data-driven neural approaches (Ratner et al., 2017; Safranchik et al., 2020; Bach et al., 2017; Wu et al., 2018; Dehghani et al., 2018; Jiang et al., 2018; Chang et al., 2020a; de Souza et al., 2018). Our approach incorporates similar line of work, by providing noisy labels (text) with a fine-tuned LM which incorporates prior knowledge from general-domain text and data-text pair (Budzianowski and Vulić, 2019; Chen et al., 2020; Peng et al., 2020; Mager et al., 2020; Harkous et al., 2020; Shen et al., 2020; Chang et al., 2020b, 2021b,a), and use it as the *weak annotator*, similar by functionality to that of fidelity-weighted learning (Dehghani et al., 2017), or data creation tool *Snorkel* (Ratner et al., 2017).

Learning with Semi-Supervision. Work on semi-supervised learning considers settings with some labeled data and a much larger set of unlabeled data, and then leverages both labeled the unlabeled data as in machine translation (Artetxe et al., 2017; Lample et al., 2017), data-to-text generation (Schmitt and Schütze, 2019; Qader et al., 2019) or more relevantly the joint learning framework for training NLU and NLG (Tseng et al., 2020; Su et al., 2020). Nonetheless, these approaches all assume that a *large collection of text* is available, which is an unrealistic assumption for the task due to the need for expert curation. In our work, we show that both NLU and NLG models can benefit from (1) automatically labeling MR with text, and (2) by semi-supervisedly learning from these samples while accounting for their qualities.

3 Approach

We represent the set of meaning representation (MR) as X and the text samples as Y. There are no restrictions on the format of the MR: each $x \in X$ can be a set of slot-value pairs, or can take the form of tree-structured semantic definitions as in Balakrishnan et al. (2019). Each text $y \in Y$ consists of a sequence of words.

In our setting, we have (1) k labeled pairs and (2) a large quantity of unlabeled MR set X_U where $|X_U| \gg k > 0$. (We force $k > 0$ as we believe a reasonable generation system needs at least a few demonstrations of the annotation.) This is a realistic setting for novel application domains, as unlabeled MR are usually abundant and can also be easily constructed from predefined schemata. Notably, *we assume no access to outside resources containing in-domain text*. The k annotations are all we know about in-domain text.

The core of our approach consists of first labeling MR samples with text, and then training on the expanded dataset. We start with describing the process of creating weakly labeled data (§4). Next, we delve into the semi-supervised training objectives for the NLU and NLG models, which allow the models to learn from labeled and unlabeled data (§5). Lastly, we explain the training process where NLG and NLU models are jointly optimized in two steps: In *step 1*, we pretrain the models on the weakly-labeled corpus, then *continue updating the models* on the combined data consisting of the weak and real data in *step 2*. Importantly, to account for the noise that comes with the automatic weak annotation, *step 2* trains the model with quality-weighted updates (§6). We depict this process in Figure 2.

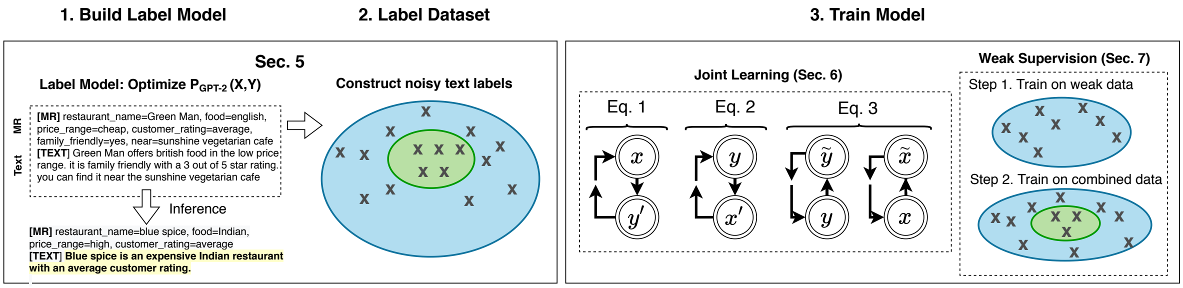


Figure 2: Depiction of the proposed framework. In *joint learning*, gradients are back-propagated through solid lines.

4 Creating Weakly Labeled Data

We construct synthetic data in two ways: (1) creating more MR samples (see §4.1), and (2) by creating a larger parallel set of MRs with texts (see §4.2).

4.1 Generating Synthetic MR Samples

We consider a simple way of MR augmentation via value swapping. This creates more unlabeled MR to be annotated by the *weak annotator* and also provide a substantial augmentation that benefits the autoencoding on MR samples (see Equation 3) by exposing it to a larger set of MR.

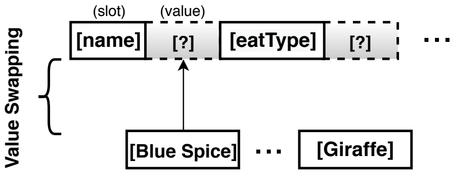


Figure 3: Depiction of MR augmentation in the E2E corpus.

Since each slot in the MR samples corresponds to multiple possible values, we pair each *slot* with a randomly sampled value collected from the set of all MR samples to obtain new combination of slot-value pairs. This way, we create a large synthetic MR set.

4.2 Creation of Parallel MR-to-Text Set

GPT-2 (Radford et al., 2019) is a powerful language model pretrained on the large WebText corpus. Recent work on conditional data-to-text generation (Harkous et al., 2020; Mager et al., 2020) demonstrated that fine-tuning GPT-2 on the joint distribution of MR and text for text-only generation yields impressive performance.

The fine-tuned model generates in-domain text by conditioning on samples from the augmented MR set (X_U). Rather than using GPT-2 outputs directly, we employ them in a process analogous to knowledge distillation (Tan et al., 2018; Tang et al., 2019; Baziotis et al., 2020) where the fine-tuned GPT-2 provides supervisory signals instead of being used directly for generation.

We now describe the process of GPT-2 fine-tuning. Given the sequential MR representation $x_1 \cdots x_M$ and a sentence $y_1 \cdots y_N$ in the labeled dataset (X_L, Y_L), we maximize the joint probability $p_{\text{GPT-2}}(X_L, Y_L)$, where each sequence is concatenated into "[MR] $x_1 \cdots x_M$ [TEXT] $y_1 \cdots y_N$". In addition, we also freeze the input embeddings when fine-tuning had positive impact on performance, following Mager et al. (2020). At test time, we provide the MR samples as context as in conventional conditional text generation:

$$\tilde{y}_j = \arg\max_{y_j}\{p_{\text{GPT-2}}(y_j \mid y_{1:j-1}, x_{1:N})\}$$

The fine-tuned LM conditions on augmented MR sample set X_U to generate the in-domain text[1], forming the weak label dataset $\mathcal{D}_W = (X_U, \tilde{Y}_L)$ with noisy labels $\tilde{y}_i \in \tilde{Y}_L$. In practice, the fine-tuned LM produces malformed, synthetic text which does not fully match with the MR it was conditioned on, as it might hallucinate additional values not consistent with its MR counterpart. Thus, it is necessary to check for factual consistency (Moryossef et al., 2019). We address this point next.

Past findings showed (e.g. (Wang, 2019)) that the removal of utterance with "hallucinated" facts (MR values) from MR leads to considerable performance gain, since inconsistent MR-Text correspondence might misguide systems to generate incorrect facts and deteriorate the NLG outputs. We filter out the synthetic, poor quality MR-text

---

[1] We adopt the Top-k random sampling with $k = 2$ to encourage diversity and reduce repetition (Radford et al., 2019)

pairs by training a separate NLU model on the original labeled data to predict MR from generated text labels. These MRs can then be checked against the paired MR in $\mathcal{D}_W$ via pattern matching as inspired by Cai and Knight (2013); Wiseman et al. (2017). Specifically, we use a measure of semantic similarity in terms of *f-score via matching of slots between the two MRs*. We keep all MR-text pairs with f-scores above 0.7, as we found empirically that this criterion retains a sufficiently large amount of high-quality data. The removed text sentences are used for unsupervised training objectives as in Eq. 1-3. Using this method, we create a collection of parallel MR-text samples (~500k) an order of magnitude larger than even the full training sets (~40k for E2E and ~25k for Weather).

5 Joint learning of NLG and NLU

For both NLU and NLG models, we adopt the same architecture as Tseng et al. (2020), which use two Bi-LSTM-based (Hochreiter and Schmidhuber, 1997) encoders for each model. The NLU decoder for slot-value structured data (e.g., E2E, Mrkšić et al., 2017) contains several 1-layer feed-forward neural classifiers for each slot; while for tree-structured meaning representation in Balakrishnan et al. (2019), the decoder is LSTM-based. In this framework, both NLU and NLG models are trained to infer the shared latent variable repeatedly – starting from either MR or text, in order to encourage semantic consistency. Each model can be improved via gradient passing between them using REINFORCE (Williams, 1992). This way, the models benefit from each other's training in a process known as the *dual learning* (Su et al., 2020), which consists of both *unsupervised* and *supervised* learning objectives. We now go into details describing them.

Unsupervised Learning. Starting from either a MR sample or a text sample, the models project the sample from one space into the other, then map it back to the original space (either MR or text sample, respectively), and compute the reconstruction loss after the two operations. This repetition will result in aligned pairs between the MR samples and corresponding text (He et al., 2016). Specifically, let $p_\theta(y|x)$ be the probability distribution to map x to its corresponding y (NLG), and $p_\phi(x|y)$ be the probability distribution to map y back to x (NLU). Starting from $x \in X$, its objective is:

$$\max_{\phi} \mathbb{E}_{x \sim p(X)} \log p_\phi(x|y');\; y' \sim p_\theta(y|x) \quad (1)$$

which ensures the semantic consistency by first performing NLG accompanied by NLU in direction $x \to y' \to x$. Note that only p_ϕ is updated in this direction and p_θ serves only as as an auxiliary function to provide pseudo samples y' from x. Similarly, starting from $y \in Y$, the objective ensures semantic consistency in the direction where the NLU step is followed by NLG: $y \to x' \to y$[2]:

$$\max_{\theta} \mathbb{E}_{y \sim p(Y)} \log p_\theta(y|x');\; x' \sim p_\phi(x|y) \quad (2)$$

We further add two autoencoding objectives on both MR and text samples:

$$\max_{\theta,\phi} \mathbb{E}_{x \sim p(X), y \sim p(Y)} \log p_\phi(x|x) p_\theta(y|y) \quad (3)$$

Thus, unlabeled text samples can be used as they are shown to benefit the text space (Y) by introducing new signals into learning directions $y \to x' \to y$ and $\tilde{y} \to y$. Thus, *we use all in-domain text data* whether they have corresponding MR or not. Note that following (Tseng et al., 2020), we also adopt the variational optimization objective upon the latent variable z which was shown to pull the inferred posteriors $q(z|x)$ and $q(z|y)$ closer to each other. In this case, the parameters of both NLG and NLU models are updated.

Supervised Learning. Apart from the above unsupervised objectives, we can impose the supervised objective on the k labeled pairs:

$$\max_{\theta,\phi} \mathbb{E}_{x,y \sim p(X_L, Y_L)} \log p_\theta(y|x) + \log p_\phi(x|y) \quad (4)$$

Each MR is flattened into a sequence and fed into the NLG encoder, giving NLG and NLU models an inductive bias to project similar MR/text into the surrounding latent space (Chisholm et al., 2017). As we observed anecdotally[3], the information flow enabled by REINFORCE allows the models to utilize unlabeled MR and text, boosting the performance in our scenarios.

---

[2] This direction is usually termed as *back translation* in MT community (Sennrich et al., 2016; Lample et al., 2018)

[3] Tseng et al. (2020) noticed similar trend in the experiments.

6 Learning with Weak Supervision

The primary challenge that arises from the synthetic data is the *noise* introduced during the generation process. Noisy and poor quality labels tend to bring little to no improvements (Elman, 1993; Frénay and Verleysen, 2013). To better train on the large and noisy corpus described in section §4 (size ~500k), we employ a two-step training process motivated by fidelity-weighted learning (Dehghani et al., 2018). The two-step process consists of (1) *pretraining* and (2) *quality-weighted fine-tuning* to account for the heterogenous data quality.

Step 1: ***Pre-train two sets of models on weak and clean data, respectively.*** We train the first set of models (*teacher*) consisting of NLU, NLG, and autoencoder (AUTO) models on the clean data. The second set of models (i.e. NLU and NLG) is the *student* that pretrains on the weak data.

Step 2: ***Fine-tune the student model parameters on the combined clean and weak datasets.*** We use each *teacher model* to determine the step size for each iteration of the stochastic gradient descent (SGD) by down-weighting the training step of the corresponding *student model* using the sample quality given by the teacher. Data points with true labels will have high quality, and thus will be given a larger step-size when updating the parameters; conversely, we down-weight the training steps of the student for data points where the teacher is not confident. For this specific fine-tuning process, we update the parameters of the student (i.e. NLG and NLU models) at time t by training with SGD, where $\mathcal{L}(\cdot)$ is the loss of predicting $\hat{y}$ for an input x_i when the label is $\tilde{y}$. The weighted step is then $c(x_i, \tilde{y}_i)\nabla\mathcal{L}(\hat{y}, \tilde{y})$, where $c(\cdot)$ is a scoring function learned by the *teacher* taking as input MR x_i and its noisy text label $\tilde{y}_i$. In essence, we control the degree of parameter updates to the *student* based on how reliable its labels are according to the *teacher*.

We denote $c(\cdot)$ as the function of the label quality based on the *dual mutual information (DMI)*, defined as the *absolute* difference between mutual information (MI)[4] in inference directions $x \rightarrow y$ and $y \rightarrow x$. Bugliarello et al. (2020) shows that $\text{MI}_{x\rightarrow y}$ correlates to the difficulty in predicting y from x, and vice versa. Thus we expect the difference between $\text{MI}_{x\rightarrow y}$ and $\text{MI}_{y\rightarrow x}$ for clean sample (x, y) to be relatively small compared to noisy samples, since the level of difficulty is largely *proportional* between NLU and NLG on the samples – difficulty in inferring x from y will result in harder prediction of y from x. Based on this intuition, the DMI score of the sample (x, y) is defined as:

$$exp\left\{\left|\underbrace{\log\frac{q_{\text{AUTO}}(y)}{q_{\text{NLG}}(y|x)}}_{\text{MI}_{x\rightarrow y}} - \underbrace{\log\frac{q_{\text{AUTO}}(x)}{q_{\text{NLU}}(x|y)}}_{\text{MI}_{y\rightarrow x}}\right|\right\}.$$

where $q(\cdot)$ are the two respective models. The DMI for a clean MR-text pair should be *relatively small*, as the two sides contain proportional semantic information[5], and so poor quality samples tend to have higher DMI scores and lower $c(\cdot)$ as they are *less semantically aligned*. Thus, $c(\cdot)$ defines the *confidence* (quality) the teacher has about the current MR-text sample. We use $c(\cdot)$ to scale η_t. Note that $\eta_t(t)$ does not necessarily depend on each data point, whereas $c(\cdot)$ does. We define $c(x_t, y_t)$ as:

$$c(x_t, y_t) = 1 - \mathcal{N}(\text{DMI}(x_t, y_t))$$

where $\mathcal{N}(\cdot)$ normalizes DMI over all samples in both clean and weak data to be in $[0, 1]$.

7 Experiment Setting

Data. We conduct experiments on the Weather (Balakrishnan et al., 2019) and E2E (Novikova et al., 2017b) datasets. Weather contains 25k instances of tree-structure annotations. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The inputs are dialogue acts consisting of three to eight slot-value pairs.

Configurations. Both NLU and NLG models are implemented in PyTorch (Paszke et al., 2019) with 2 Bi-LSTM layers and 200-dimensional token embeddings and Adam optimizer (Kingma and Ba, 2014) with initial learning rate at 0.0002. Batch size is kept at 28 and we employ beam search with size 3 for decoding. The score is averaged over 10 random initialization runs. In our implementation, the sequence-to-sequence models are built upon the bi-directional long short-term memory (Bi-LSTM) (Hochreiter and Schmidhuber, 1997). For LSTM cells, both the encoder and decoder have 2 layers, amounting to 18M parameters for

---

[4] Mutual information for $x \rightarrow y$ can be seen as $H(x \rightarrow y) = H_{AUTO}(y) - H_{NLG}(y|x)$ (Bugliarello et al., 2020).

[5] We found that mutual information for $x \rightarrow y$ is usually greater than that of $y \rightarrow x$ since NLG is a one-to-many and more difficult process as opposed to NLU.

Model	E2E (NLG)					E2E (NLU)				
	10	50	1%	5%	50%	10	50	1%	5%	50%
WA	0.195	0.287	0.563	0.649	0.714	9.48	11.66	13.20	45.21	65.81
JUG*	0.002	0.015	0.726	0.7671	0.819	0.00	0.00	32.24	53.20	**78.93**
decoupled	0.261	0.279	0.648	0.693	0.793	0.00	0.00	20.51	52.77	73.68
joint	0.218	0.336	0.732	0.764	0.775	0.00	6.18	24.98	49.66	70.33
joint+aug	0.275	0.381	0.748	0.781	0.797	5.88	15.79	25.15	53.20	69.68
step 1	0.441	0.487	0.610	0.642	0.685	13.18	14.28	15.37	44.72	65.20
Ours (step 1+2)	**0.489**	**0.558**	**0.754**	**0.775**	**0.822**	**15.81**	**23.67**	**34.09**	**56.33**	72.45

	Weather (NLG)					Weather (NLU)				
Model	10	50	1%	5%	50%	10	50	1%	5%	50%
WA	0.261	0.332	0.518	0.567	0.611	8.42	30.64	66.41	70.19	75.26
JUG*	0.005	0.244	0.618	0.670	**0.726**	0.00	33.48	67.44	79.19	**89.17**
decoupled	0.250	0.288	0.598	0.632	0.719	0.00	28.21	70.24	73.46	88.45
joint	0.270	0.348	0.577	0.639	0.658	0.00	24.52	64.30	69.92	86.86
joint+aug	0.329	0.361	0.589	0.662	0.671	4.21	26.33	67.43	71.19	87.10
step 1	0.371	0.429	0.570	0.607	0.632	12.19	35.89	72.90	72.01	84.73
Ours (step 1+2)	**0.401**	**0.458**	**0.644**	**0.672**	0.717	**16.62**	**42.74**	**75.94**	**80.36**	87.77

Table 1: Performance for NLG (BLEU-4) and NLU (joint accuracy (%)) on E2E and Weather datasets with increasing amount of labeled data from 10, 50 labeled instances to 1%, 5%, and 100% of the labeled data (D_L). Models that have access to **unlabeled ground-truth text labels** are marked with *. We provide results for the NLG and NLU models trained separately using supervised objectives alone (*decoupled*), our semi-supervised joint-learning model (*joint*), *joint* with all unlabeled data (*joint+aug*), and weakly-supervised models (*step 1*). *Step 1+2* denotes the full proposed approach.

	D_L	D_W	X_U	Y_{SL}	Y_{WL}
JUG	✓	✗	✓	✓	✗
WA	✓	✗	✗	✗	✗
decoupled	✓	✗	✗	✗	✗
joint	✓	✗	✓	✗	✗
joint+aug	✓	✗	✓	✗	✓
step 1	✗	✓	✓	✗	✓
Ours (step 1+2)	✓	✓	✓	✗	✓

Table 2: Summary of training data used in each model. Sources of data include labeled data (D_L), unlabeled MR (X_U), weakly labeled data (D_W), 100% real text (Y_{SL}), and weak text labels (Y_{WL}).

E2E NLG	**BLEU-4**
TGEN (Dušek and Jurcicek, 2016)	0.6593
SLUG (Juraska et al., 2018)	0.6619
Dual supervised learning (Su et al., 2019)	0.5716
JUG (Tseng et al., 2020)	0.6855
GPT2-FT (Chen et al., 2020)	0.6562
WA (Harkous et al., 2020)	0.6445
Ours (step 1+2)	**0.7025**

Weather NLG	**BLEU-4**
S2S-CONSTR (Balakrishnan et al., 2019)	0.7660
JUG (Tseng et al., 2020)	0.7768
Ours (step 1+2)	**0.7986**

Table 3: For comparison, we show the performance of previous systems on the datasets following the **original split**, so the scores are **not comparable** to Table 1.

the seq2seq model. All models were trained on 1 Nvidia V100 GPU (32GB and CUDA Version 10.2) for 10k steps. The average training time for seq2seq model was approximately 1 hour, and roughly 2 hours for the proposed semi-supervised training with 100% data. The total number of updates is set to 10k steps for all training and patience is set as 100 updates. At decoding time, sentences are generated using greedy decoding.

8 Results

We first compare our model with other baselines on both datasets, then perform a set of ablation studies on the E2E dataset to see the effects of each component. Finally, we analyze the strength of the *weak annotator*, and the effect of the quality-weighted weak supervision, before concluding with the analysis of dual mutual information.

In particular, we experiment with various low resource conditions of training set (10 instances, 50 instances, 1% of all data, 5% of all data). To show that our proposed approach is consistently better, we include the scenario with 0-100% of the data at 10% interval, to show that performance does not deteriorate as more training samples are added (Figure 4). Table 2 shows the summary of training data used for all models in Table 1. We compare our model with (1) a fine-tuned GPT2 model (GPT2-FT) that uses a switch mechanism to select between input and GPT2 knowledge (Chen et al., 2020)[6], (2) a fine-tuning approach to be used as the weak

---

[6] `https://github.com/czyssrs/Few-Shot-NLG`

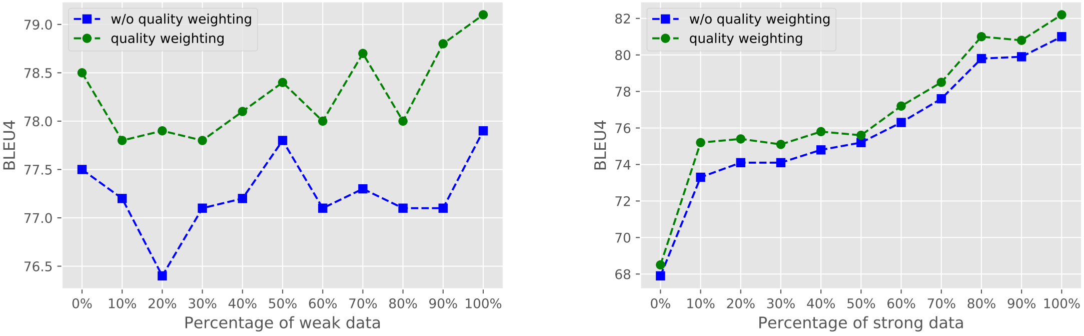


Figure 4: Model performance (BLEU-4) on 5% E2E data with varying percentages of *strong* and *weak* data *with* and *without* DMI-based quality weighting. **Left plot** begins with models trained on labeled data while **right plot** starts with the weak synthesized dataset instead.

Method	NLU			NLG		
	Miss	Redundant	Wrong	Fluency	Miss	Wrong
decoupled	72	78	87	4.10	69	73
JUG	65	**72**	75	4.23	64	65
Ours (step 1+2)	**54**	77	**68**	**4.50**	**63**	**61**

Table 4: Human evaluation on the sampled E2E outputs (100 instances) for models with 1% training data. Numbers of *missing*, *redundant* and *wrong* predictions on slot-value pairs are reported for NLU; *fluency*, numbers of *missing* or *wrong* generated slot values are listed for NLG.

annotator (WA) that predict text from MR or MR from text, depending on the input format during fine-tuning (Harkous et al., 2020)[7], and (3) the semi-supervised model [8] (JUG) from Tseng et al. (2020). Note that the specialized encoder in *GPT2-FT* cannot be easily adapted to the tree-structured input in Weather, and so we do not provide its score on the Weather dataset.

In Table 1, we show that our proposed approach (*step 1+2*) generally performs better than the baselines for both tasks (NLG and NLU) for most selected labeled data sizes. We show that even with only 10 labeled instances, our approach (*step 1+2*) is able to yield decent results compared to the baselines. The difference between models tends to be larger for settings with few training instances, and the advantage of the method diminishes *as the amount of labeled data available for* JUG *increases*, to the point where JUG is able to outperform the proposed approach. Overall, the benefit of the noisy supervisory signal from the weak data is able to boost performance, especially at lower resource conditions.

We observe that training with weakly labeled data alone (step 1) is not sufficient, and so strong data is required to provide the supervisory signals necessary (step 2). Further, the fact that *joint+aug* displays noticeable improvements over *joint* suggests that simply having augmented text helps to improve the encoded latent space as projected by both the NLU and NLG encoders. This also shows an alternative way to introduce additional in-domain information to both models, even though the NLU model does not benefit directly from additional text. Importantly, our approach shows that the *weak annotator* is able to bridge the gap as defined by the access to ground-truth text labels in JUG – outperforming it significantly at low resource conditions (10, 50, 1%, 5%) with the difference in NLG being as large as 48.7 BLEU points with 10 instances. We find that the proposed model also performs well in the high resource (100% of labeled data) condition, as shown in Table 3. Moreover, with 100% labeled data, our model is still able to produce superior performance over some of the baselines, which shows that weak annotation does capture additional useful patterns that benefit the NLG process.

9 Analysis

Error Analysis. Since word-level overlapping scores usually correlate rather poorly with human judgements on fluency and information accuracy (Reiter and Belz, 2009; Novikova et al., 2017a), we perform human evaluation on the E2E corpus on 100 sampled generation outputs. For each MR-text pair, the annotator is instructed to evaluate the *fluency* (score 1-5, with 5 being most fluent), *miss* (count of MR slots that were missed) and *wrong* (count of included slots not in MR) are presented in Table 4, where fluency scores are averaged over 50 crowdworkers. We show that with 1% data, both NLU and NLG models yield signif-

---

[7] No released source code so we re-implemented it based on paper.

[8] `https://github.com/andy194673/Joint-NLU-NLG`

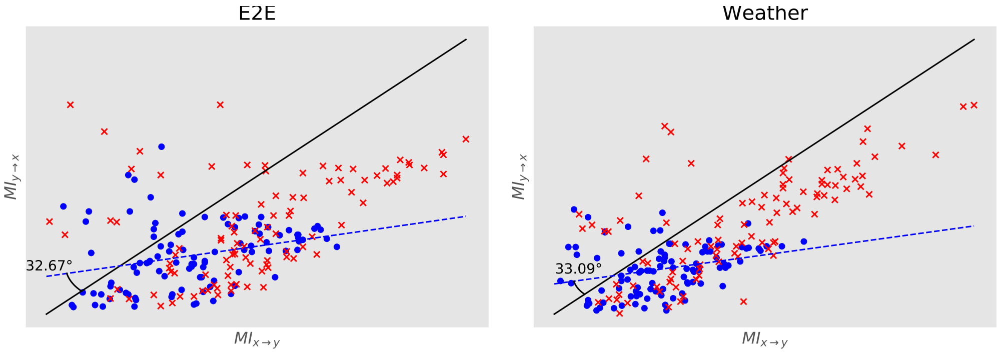


Figure 5: Visualization of dual mutual information (DMI) on both datasets where × markers are 50 random samples from annotated data and ○ markers are 50 random samples in the weak dataset. **Dotted** lines are trend lines for ○ markers and **solid black** lines are diagonal reference that correspond to the perfect NLG-NLU balance where both tasks have equal difficulties.

Method	NLG BLEU-4 (Accuracy (%))	NLU Accuracy (%) (F1)
w/ *DF*	0.683 (77.69)	24.71 (0.6443)
with *DF*	0.703 (79.08)	27.19 (0.6840)
with *WS*	0.733 (82.65)	30.23 (0.7028)
with *WS+CW*	**0.754 (86.44)**	**34.09 (0.7200)**

Table 5: Ablation study of weak supervision (1% E2E labeled data D_L) including data fidelity (*DF*), the proposed model (step 1+2) with weak supervision (*WS*), and *WS* with quality-weighted weak supervision (*WS+CW*).

icantly fewer errors in terms of *misses* and *wrong* facts, while having more fluent outputs. However, it generates more redundant slot-value pairs which we attribute to the noisy augmentation that "misguided" the NLU model.

How Strong is the *Weak Annotator*? To assess the strength of the weak annotator (WA) itself, we also computed its NLG scores with varying amounts of labeled data (see Table 1). We observe that the WA suffers from a performance drop in lower resource conditions (i.e. 0.195 BLEU with 10 labeled instances), when the given training samples are not sufficient for the pretrained model to converge upon a region of in-domain generation. However, it yields some quality data when conditioned on a large number of possible MR (i.e. 50% data), forming a useful in-domain text set (See Table 6).

Analysis of Weak Supervision. In Table 5, we present the results of an ablation study on weak supervision (see §6) where the effect of *data fidelity* is stronger on NLU than on NLG, which is due to the nature of the filtering process which removes faulty text labels which influences both $x \rightarrow y$ and $y \rightarrow y$ training directions. Next, though weak supervision boosted the model by giving direct supervision in training directions $x \rightarrow y$ and $y \rightarrow x$, the noisy nature of the augmentation limits its effectiveness. The model is further improved with the proposed quality-weighted update that takes into account the sample quality and alleviate the influence of poor quality samples. Refer to Table 7 for output comparison.

Analysis of the Two-Step Training Process. As inspired by Dehghani et al. (2018), we justify the two-step training process by performing two types of experiments with 5% data (see Figure 4): In the first experiment, we use all the available strong data but consider different ratios of the entire weak dataset – as used in our 2-step approach. In the second, we fix the amount of weak data and provide the model with varying amounts of strong data. The results show that the *student* models are generally better off by having the *teacher*'s supervision. Further, pretraining on weak data prior to fine-tuning on strong data appears to be the better approach and this motivates the reasoning behind our two-step approach.

Analysis of the Dual Mutual Information. Figure 5 depicts DMI with the visualization of $MI_{x \rightarrow y}$ as x-axis and $MI_{y \rightarrow x}$ as y-axis, in which 100 randomly sampled noisy and ground-truth samples are plotted for both datasets. On the plot, the diagonal reference represents the scenario in which NLG and NLU inference are equally difficult, and we see that annotated data cluster more around the diagonal reference. This means that expert-labeled samples' DMI scores tend to be smaller, where NLU and NLG inference for these samples carry similar levels of difficulty. Importantly, since DMI scores are normalized over both clean and noisy samples, *the proximity of data to the trendlines can then be used to estimate the sample quality* – clean data are closer as compared to the noisy sam-

mr synthetic reference	[name] Giraffe, [eat type] pub, [area] riverside Giraffe is a pub in the riverside of the city just down the street.
mr synthetic reference	[name] Strada, [eat type] restaurant, [food] Italian, [area] city centre, [familyfriendly] no, [near] Avalon Strada is an Italian restaurant not for the families! it is near Avalon in the city centre.
mr synthetic reference	[name] Cocum, [eat type] restaurant, [food] French, [area] riverside, [familyfriendly] no, [near] Raja Indian Cuisine Cocum sells French food near Raja Indian Cuisine.

Table 6: Display of weakly-labeled data samples.

mr	[name] Blue Spice, [eat type] coffee shop, [area] city centre
step 1+2	Blue Spice is a coffee shop in the city centre that of the city.
JUG	Blue Spice serves Italian food and is family friendly.
decoupled	Blue Spice is an adult Italian coffee shop with high customer rating located in

Table 7: Display of text generations from different models.

ples. Thus clean data will have smaller normalized scores, higher $c(\cdot)$, and a larger update step. This further supports the use of the proposed sample quality-based updates on the parameters.

10 Conclusion and Future Work

In this paper, we show the efficacy of the framework where data is automatically labeled and both NLU and NLG models learn with quality-weighted weak supervision so as to account for the individual data quality. Most importantly, we show that not only is the two-step training process useful in improving the model, it yields decent quality text. This work serves as a starting point for weakly-supervised learning in natural language generation, especially for topics related to instance-based weighting approaches.

For future work, we hope to extend on the framework and propose ways with which it can be incorporated into existing text annotation systems.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 “Foundations of Perspicuous Software Systems”. We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. 2017. Learning the structure of generative models without labeled data. *Proceedings of machine learning research*, 70:273.

Anusha Balakrishnan, Jinfeng Rao, Kartikeya Upasani, Michael White, and Rajen Subba. 2019. Constrained decoding for neural nlg from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 831–844.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Christos Baziotis, Barry Haddow, and Alexandra Birch. 2020. Language model prior for low-resource neural machine translation. *arXiv preprint arXiv:2004.14928*.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it’s gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Emanuele Bugliarello, Sabrina J Mielke, Antonios Anastasopoulos, Ryan Cotterell, and Naoaki Okazaki. 2020. It’s easier to translate out of english than into it: Measuring neural translation difficulty by cross-mutual information. *arXiv preprint arXiv:2005.02354*.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752.

Ernie Chang, David Adelani, Xiaoyu Shen, and Vera Demberg. 2020a. Unsupervised pidgin text generation by pivoting english data and self-training. In *In Proceedings of Workshop at ICLR*.

Ernie Chang, Jeriah Caplinger, Alex Marin, Xiaoyu Shen, and Vera Demberg. 2020b. Dart: A lightweight quality-suggestive data-to-text annotation tool. In *COLING 2020*, pages 12–17.

Ernie Chang, Xiaoyu Shen, Dawei Zhu, Vera Demberg, and Hui. Su. 2021a. Neural data-to-text generation with lm-based text augmentation. *EACL 2021*.

Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. 2021b. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. In *EACL 2021*.

Zhiyu Chen, Harini Eavani, Yinyin Liu, and William Yang Wang. 2020. Few-shot nlg with pretrained language model. *ACL*.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 633–642.

Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. 2017. Fidelity-weighted learning. *arXiv preprint arXiv:1711.02799*.

Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. 2018. Fidelity-weighted learning. In *International Conference on Learning Representations*.

Ondřej Dušek and Filip Jurcicek. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51.

Jeffrey L Elman. 1993. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99.

Benoît Frénay and Michel Verleysen. 2013. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Jason A Fries, Ethan Steinberg, Saelig Khattar, Scott L Fleming, Jose Posada, Alison Callahan, and Nigam H Shah. 2020. Trove: Ontology-driven weak supervision for medical entity classification. *arXiv preprint arXiv:2008.01972*.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. *arXiv preprint arXiv:2004.06577*.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in neural information processing systems*, pages 820–828.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, et al. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. *arXiv preprint arXiv:2005.09123*.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017a. Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017b. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*.

Raheel Qader, François Portet, and Cyril Labbé. 2019. Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 552–562.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Esteban Safranchik, Shiying Luo, Stephen H Bach, Elaheh Raisi, Stephen H Bach, Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, et al. 2020. Weakly supervised sequence tagging from noisy rules. In *AAAI*, pages 5570–5578.

Martin Schmitt and Hinrich Schütze. 2019. Unsupervised text generation from structured data. *arXiv preprint arXiv:1904.09447*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 86–96.

Xiaoyu Shen, Ernie Chang, Hui Su, Jie Zhou, and Dietrich Klakow. 2020. Neural data-to-text generation via jointly learning the segmentation and correspondence. In *ACL 2020*. https://www.aclweb.org/anthology/2020.acl-main.641.pdf.

José GC de Souza, Michael Kozielski, Prashant Mathur, Ernie Chang, Marco Guerini, Matteo Negri, Marco Turchi, and Evgeny Matusov. 2018. Generating e-commerce product titles and predicting their quality. In *INLG*, pages 233–243.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2019. Dual supervised learning for natural language understanding and generation. *arXiv preprint arXiv:1905.06196*.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2020. Towards unsupervised language understanding and generation by joint dual learning. *ACL*.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2018. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*.

Raphael Tang, Yao Lu, and Jimmy Lin. 2019. Natural language generation for effective knowledge distillation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 202–208.

Bo-Hsiang Tseng, Jianpeng Cheng, Yimai Fang, and David Vandyke. 2020. A generative model for joint natural language understanding and generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1795–1807.

Hongmin Wang. 2019. Revisiting challenges in data-to-text generation with fact grounding. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 311–322.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2018. Learning matching models with weak supervision for response selection in retrieval-based chatbots. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 420–425.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the*

Twenty-Fifth International Joint Conference on Artificial Intelligence, pages 2993–2999.

Zhi-Hua Zhou. 2018. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53.

Continuous Learning in Neural Machine Translation using Bilingual Dictionaries

Jan Niehues
Department of Data Science and Knowledge Engineering (DKE)
Maastricht University, Maastricht, The Netherlands
jan.niehues@maastrichtuniversity.nl

Abstract

While recent advances in deep learning led to significant improvements in machine translation, neural machine translation is often still not able to continuously adapt to the environment. For humans, as well as for machine translation, bilingual dictionaries are a promising knowledge source to continuously integrate new knowledge. However, their exploitation poses several challenges: The system needs to be able to perform one-shot learning as well as model the morphology of source and target language.

In this work, we proposed an evaluation framework to assess the ability of neural machine translation to continuously learn new phrases. We integrate one-shot learning methods for neural machine translation with different word representations and show that it is important to address both in order to successfully make use of bilingual dictionaries. By addressing both challenges we are able to improve the ability to translate new, rare words and phrases from 30% to up to 70%. The correct lemma is even generated by more than 90%.

1 Introduction

Recent advances in neural machine translation (NMT) have led to astonishing translation quality of research systems in evaluation campaigns as well as for commercial systems. These improvements even led to discussions whether automatic machine translation is already on par with human translation (Barrault et al., 2019). One challenge that has raised less attention is the ability of these systems to continuously learn over time. In contrast, humans are continuously improving their skills and adapting to an ever-changing environment.

There are several reasons why this is necessary: First, nobody is fluent in all possible domains. Even professional translators need to adapt to the specific vocabulary of different domains. Secondly, language is not static but developing over time and translators need to learn new terms, meanings and expressions.

For humans, one successful approach to adapt to the environment is the usage of a dictionary [1]. Learning translations from a dictionary has several advantages: Dictionaries contain minimal examples. We do not need to collect full sentences, but can directly learn translations from a single phrase. Furthermore, this can even be generalized to other inflected forms of the same lexem. Secondly, it enables the system to directly integrate correction. If a user sees a specific problem, the user can interact with the system by adding a specific dictionary entry. This is very important if a specific terminology should be used.

Motivated by the success for human translators, in this work we will enable NMT to also successfully integrate knowledge from bilingual dictionaries. Thereby, we will focus on learning translations that could not be learned from parallel data. This poses several interesting research challenges as shown in the example in Table 1. When training a system on the proceedings of the European Parliament, it might never have seen the word *giraffe* and needs to learn the translation from the dictionary. First of all, we have to address one-shot learning. The system needs to be able to continuously learn new dictionary entries and then should directly be able to translate all occurrences of this phrase.

Secondly, the model must be aware of the morphology of the source and target language. In a dictionary only the base form of a word is given. In the example only the lemma *giraffe* is in the dictionary, but not the plural form *giraffes*. Therefore, we must enable the system to translate different lexemes of a lemma by knowing only the translation of the base form. This involves analysing the mor-

---

[1] In this work the dictionary entries can consist of a single word or whole phrases

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 830–840
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Source:	Tell us, what have you got against **giraffes** ?
Dictionary:	giraffe → Giraffe
Reference:	was haben Sie eigentlich gegen **Giraffen** ?
Annotation:	Tell us, what have you got against # giraffes # Giraffe # ?

Table 1: Example of dictionary usage

phological form of the source word, transferring the information about the form to the target and finally generating the correct morphological form of the target word based on the dictionary entry as well as on the morphological form of the source word. In German the plural of the dictionary entry *Giraffe* is *Giraffen*.

In order to assess the approaches on this challenging condition, it is essential to define an appropriate evaluation scheme. While the ability to continuously learn new translations is essential in many practical applications, the newly learned terminology will only occur rarely. Therefore, standard methods for evaluating machine translation are not able to measure the effect appropriately.

In order to address these challenges, we develop the following contributions:

- We developed a targeted evaluation approach for the continuous learning of new translations (Section 2)
- We showed that character-based representation is essential to inflect unknown words correctly. (Section 3)
- We show that only the combination of word representation and one-shot learning enables the successful integration of bilingual dictionaries (Section 3)

2 Evaluation scenario

The first important research question that needs to be addressed in the targeted continuous learning scenario is the evaluation approach. While the evaluation of machine translation is well-established (e.g. using BLEU (Papineni et al., 2002)), new learned words are typically rare words and therefore their influence on a BLEU score calculated on all words is very limited.

In order to have a valid evaluation approach, the evaluation should focus on phrases that cannot be learned from the parallel data. These are typically very rare phrases. Furthermore, we want to translate them in a real world situation. Therefore, the evaluation data should not be synthetic sentences. Finally, the approach should be using the standard parallel data without the need of collecting additional parallel data.

A first attempt would be to use existing test data and select sentences where dictionary entries are needed as e.g. done in (Dinu et al., 2019). However, if we limit ourselves to phrases that do not occur in the parallel data or only a few times, the number of occurring words in the test sets are too low to draw any conclusions.

Therefore, we evaluate our approach by proposing a new test-train split of existing parallel data. In a first step, we filter a large background dictionary for entries that help to translate phrases that only occur a few times in the existing parallel data. In a second step, we select some of the sentences with their matching dictionaries entries as the new test sets. An overview of the process is shown in Figure 1. Finally, we specifically evaluate the ability of the translation system to translate the dictionary entries.

In addition, it is important to ensure that the proposed methods do not have negative side effects on the overall translation quality. Therefore, we also evaluate the model using standard evaluation metrics on well-established test sets and on the proposed test set. Due to the weakness of these metrics to measure improvements in rare words, we do not expect that the proposed methods improve on these metrics, but it is important that the performance measured in these metrics does not decrease significantly.

2.1 Dictionary filtering

In a first step, we create a large background dictionary for each considered language pair by extracting a bilingual dictionary from the English Wiktionary. Therefore, we extracted the translation from a Wiktionary dump[2] using wiktextract [3].

Secondly, we match the dictionary entries to the

---

[2] https://dumps.wikimedia.org/enwiktionary/20200501/enwiktionary-20200501-pages-articles.xml.bz2

[3] https://github.com/tatuylonen/wiktextract

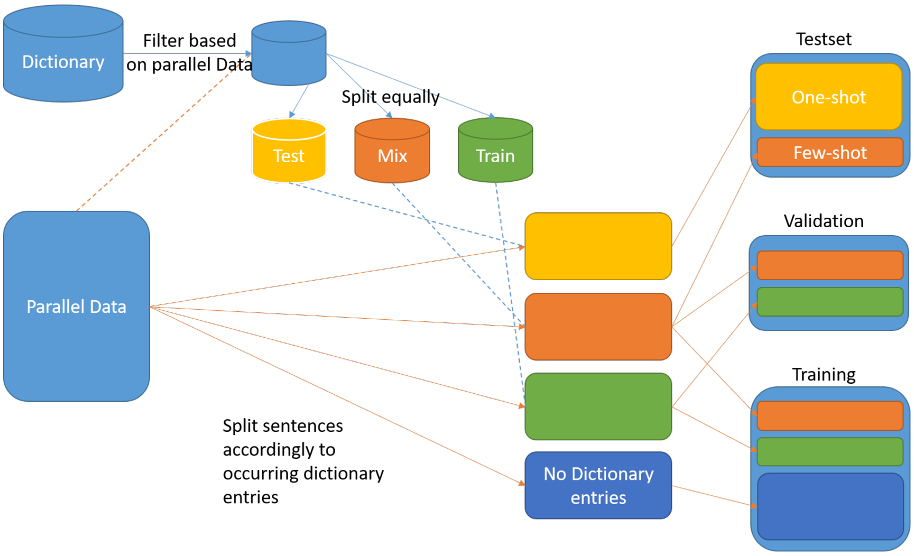


Figure 1: Overview of the evaluation approach: Based on the parallel data and dictionary, a new split of the data is generated

targeted corpus. Therefore, we lemmatize all dictionary entries as well as both sides of the parallel data. This is done to also find matches for all morphological variants of the dictionary entries. Finally, we calculate the statistics mentioned in Table 2 for each dictionary entry about its matches to the parallel data.

In a third step, we filter the dictionary based on the statistics. We only select words that are rare in the corpus. If the words are common and occur often in the training data, a dictionary entry would not be helpful. Secondly, we want to analyse the ability of the system to generate different morphological forms. Therefore, we only consider entries that occur at least with two different morphological variants on the target side. Finally, in this work we focus on words that are not ambiguous. We leave an integration of word sense disambiguation to also handle ambiguous dictionary entries for future work. Therefore, we only consider phrases, where both, the source and target phrase, occur less than 10 times with a different translation than the one given in the dictionary.

Statistic	Threshold
Occurrences	$3 \leq k \leq 80$
target inflected phrases	≥ 2
only source/target match	< 10

Table 2: Dictionary filtering

2.2 Train-Test Split

Finally we generate a split of the corpus into training, validation and test sets based on the selected dictionary entries as shown in Figure 1. The model needs to learn how to use the dictionary. Therefore, several training sentences need to be annotated with dictionary entries. Furthermore, we want to evaluate the ability of the model to translate phrases it has seen a few times in training (*Few-Shot learning*) as well as words it only has seen in the dictionary (*One-shot learning*). Therefore, we split the entries in the dictionary equally into three sets (*Test (yellow)*, *Mix (orange)* and *Train (green)*). All sentence pairs associated with entries from the *Test* set are added to the newly created test set.

In a second step, we select all the sentences from the remaining training sentences, where an entry from the *Mix* set occurs. For each entry, half the sentences are added to the test set and a quarter to the validation and training set.

Finally, all sentences with entries from *Train* are equally distributed to the training and validation set. Since we want to concentrate on modelling the morphology when using the dictionary and not the translation ambiguity for dictionary entries, we removed all sentences from the training where the source entry from the dictionary occurs, but the target sentence does not contain the target entry. Due to our selection of the dictionary, where we focus on words that have only very few different translations (less than 10 times a different one), we only removed very few sentences here. All remaining sentences with no annotations (most of the sentences) were used for training.

2.3 Evaluation

When evaluating we want to focus on the system's ability to translate the phrases from the dictionary. Therefore, we measure the accuracy of translating the dictionary entries in addition to the commonly used BLEU score. In addition to calculating the accuracy by comparing the inflected words of hypothesis and reference (*Exact match*), we calculate further statistics to analyse the approaches.

In addition, we measure the ability of the system to at least create the correct lemma by ignoring errors made due to wrong inflection of the words. Therefore, for each sentence, we compare the target lemmatized phrase of the dictionary entry with the lemmatized version of the generated translation. We will refer to this metric as *Lemma match*

Finally, we are especially interested in the ability of the model to generate the correct inflected form. For many words, this is quite straightforward since it is the same as the lemma. Therefore, we also measure the exact match on the subset of the dictionary entries, where the target side of the dictionary is different from the inflect form occurring in the reference. To generate the correct translation, in this case the model really needs to change the output. We will refer to this metric as *Morph. Adjustment*.

In addition to these three evaluation scores, we also investigate the performance on the different types of entries. We evaluate all metrics on all entries and independently on the one-shot (*OneS*) and few-shot (*FewS*) entries.

3 NMT Dictionary Integration

To successfully integrate the dictionary into the NMT system, we need to address two challenges: First, we need to enable the system to perform one-shot learning. It should be able to translate a phrase after seeing it only once in the bilingual dictionary. Furthermore, it needs to be possible to continuously add new translations. Secondly, we need to model the morphology of the dictionary entries. We need to use the dictionary for different inflected forms of the word and also generate various inflected forms of the target phrase.

3.1 One-shot learning

In order to achieve one-shot learning, we need to combine the dictionary with our neural machine translation system. The combination should ensure fast learning, so a single dictionary entry is enough to learn the translation. Furthermore, it needs to be flexible, so new dictionary entries can be continuously added to the system and it is able to perform life-long learning by using the newly added entries.

One large advantage of deep learning approaches is that they are able to easily incorporate additional information. By annotating the input with additional information, the model is able to learn automatically how to make use of this additional information. This has been successfully done, for example, for the translation of other MT systems (Niehues et al., 2016), for domain information (Kobus et al., 2017) or information about formality (Sennrich et al., 2016a).

For the integration of additional knowledge about specific phrases, we follow similar approaches presented in Pham et al. (2018) and Dinu et al. (2019). The main idea is that we annotate each source phrase, for which a dictionary translation is available with this translation. This is done by appending the translation to the source phrase within the sentence as shown in Table 1. Since this is done during training and testing, the system is able to learn to copy and modify these suggestions. No further adaptation to the architecture of the NMT system is necessary. The system will learn how to exploit these systems and can transfer this knowledge to new translations that have not been seen in training. Therefore, the translations need only to be added once to the dictionary, which enables the system to perform one-shot learning as well as to continuously learn new translation by extending the dictionary.

The main difference to previous work is that we are focusing on very rare words and morphological variants of the dictionary phrases. Therefore, we investigate the matching of the dictionary entries as well as the number of necessary entries.

In order to find the dictionary entries for a given source sentence, we first lemmatize the sentence. In a second step, we then match the dictionary to the lemmatized sentences. Finally, we map back the found entries to the original sentence.

In annotating the source sentence, we follow the related work and append the translation to the source phrase. As shown in Figure 1, we replace the source word *giraffes* by the entry *# giraffes # Giraffe #*. In contrast to the original work, we do not have the inflect target words, but only put the lemmatized target string to the sentence. For the source side, we keep the inflected form for the source sentence so the system is able to extract important morphological information from the source (e.g. grammatical number) and map it to the target. This is done for the training and test data. Then the baseline neural machine translation system is trained normally on the annotated sentences. We did not adapt the architecture since in Dinu et al. (2019) the standard transformer based system was able to learn to copy the suggested translations into the target side.

While the system should learn to also use dictionary entries it has not seen during training, the system needs enough examples in order to learn how to use dictionary entries in general. Since we are concentrating on very rare words, the number of dictionary entries in the parallel data is relatively small. For larger corpora, we therefore explore whether it is helpful to annotate additional phrases. This was done by also extracting phrases that occur more often (*add. Annot*). However, we did use the same split and also evaluated our approach only on the rare phrases.

3.2 Word representations

A second challenge when building a machine translation system for the targeted scenario is the generation of the correct inflected word form. Since we have seen the new words only in the dictionary, we will often need to generate different inflected word forms that we have neither seen in the dictionary nor in the corpus.

While there have been attempts to generate unknown inflected word forms for dictionary entries (e.g. Niehues and Waibel (2011)) prior to neural machine translation, the ability to represent parts of the words in neural machine translation offer a unique opportunity to model morphological inflection. Therefore, in this work, we concentrated on the word representation used in the NMT system. Thereby, we always use the same representation for the source and the target language. The most commonly used word representation used in state-of-the-art neural machine translation systems are byte-pair-encodings (BPE) (Sennrich et al., 2016b). A second successful approach to represent words in a neural machine translation system are character-based representations, where each word is split into its characters.

While there have been several works on comparing these two representations(e.g. Sennrich (2017), they are mostly concentrating on generating the overall best translation performance. However, in this work, we will focus on the rare words. Since only for these words we need to learn how to generate different inflected forms. For the more frequent words, this is often not that important since all word forms occur several times in the corpus.

Besides the generation of unknown inflected forms, the word representation is also important when learning to copy the annotations to the target. If we look at the example dictionary entry *concentric* → *konzentrisch*, the lemma *konzentrisch* got split into the subwords *konzent@@ ris@@ ch* while the inflect form *konzentrischer* into *kon@@ zentr@@ ischer*. In this case there is no overlap in the subwords between the lemma and the inflected form. Therefore, it is difficult for the system to learn from the suggested translation. In contrast, when looking at the character-based representation, the model can copy the lemma and only has to learn to add additional tokens at the end.

In a first step, we compared character-based and sub-word based models. Thereby, we highlight their ability to generate new inflected forms of rare words. For both we used exactly the same NMT architecture. The only difference is that the input and output length for the character-based models is significantly larger since the number of characters is higher than the number of subwords.

We will see that the character-based models are significantly better in generating the different inflected forms for rare words. However, a major challenge is the training time. Due to the significant longer sequence length, also the training and decoding time is much slower. Therefore, we also propose a combination of word-based and character-based models.

In the mixed representation, we split each word that occurs less than k times into its characters, while the other words are kept as they are. Since only frequent words are not split into characters, no further subword segmentation for these words is performed. Thereby, we can speed up the processing due to a short sequence length, but still have the ability to learn how to inflect rare words. Some dictionary entries contain phrases with many frequent words. In order to be able to better inflect these words, in a second approach we in addition split also all words within a dictionary phrase into characters. We refer to this technique as *Mix+Ann*.

4 Experiments

We evaluate the approaches on three different data sizes and on two different language pairs (English-German and English-Czech). Since we are focusing on the generation of different morphological forms, we always use the morphologically rich language as the target language.

4.1 Data

For English-to-German we created two datasets with different sizes. A first series of experiments is run on the TED (Cettolo et al., 2012) corpus. We split the corpus into training, validation and test sets as described in Section 2. In addition, we evaluate the system also on the official test sets *tst2014*, *tst2015* and *2018* and report average metrics for these test sets.

For the second system, we use the Europarl corpus (Koehn, 2005). This corpus is around 10 times bigger than the TED corpus as shown in Table 3. In addition to the target test set, we also tested the systems on the test2006 and test2007, which are the most recent official test sets from the same domain used for the WMT.

Finally, we also tested the techniques on a different language pair. For this we choose English to Czech and also use the Europarl corpus for these experiments. Since there is no official in-domain corpus available, we tested the systems also on the newstest2019 test set.

As shown in Table 3, the parameters mentioned in Section 2 lead to a reasonable test set size for all corpora. As mentioned in Section 3.1, we evaluate the system on Europarl with different amounts of training annotations. All data sets with their splits are available for further experiments [4].

	EN-DE		EN-CS
	TED	Europarl	Europarl
Train	198K	1.9M	636K
- Annot	1.6K	1.2K	2.7K
- add. Annot		14.5K	24.3K
Valid	1610	1196	2000
Test	3181	2140	5360

Table 3: Data size in number of sentences

4.2 System

All data was processed using the Stanza toolkit (Qi et al., 2020) for tokenization and lemmatization. The lemmatization was only used for matching the dictionary entries, the translation systems were built on the inflected words. If BPE is applied, we used a BPE size of 20K. For the mixed representation, words occurring less than $k = 50$ times were represented as individual characters.

We use the standard transformer architecture (Vaswani et al., 2017) and increase the number of layers to eight. The layer size is 512 and the inner size is 2048. Furthermore, we apply word dropout (Gal and Ghahramani, 2016) with $p = 0.1$. We use the same learning rate schedule as in the original work and the implementation presented in (Pham et al., 2019) [5]. All systems were always trained from scratch with random initialization.

4.3 TED

A first series of experiments were performed on the TED task. We evaluated the one-shot learning approach by source sentence annotation as well as the three different word representations described in Section 3.2. In a first step, we evaluated the translation performance using BLEU (mteval-v14.pl) and characTER (Wang et al., 2016) on the continuous learning test set as well as on the official test set (Table 4).

The baseline systems using no one-shot learning do not annotate the source at all and are trained on the standard parallel data. If we take a look at the official test set, we see systems using character-based representation (*Character* and *Mix*) perform slightly better than the subword-based models. This might be due to the fact that the TED training data is rather small. Secondly, the one-shot learning approach has no influence on the translation performance of this test set. This is not surprising, since only 94 phrases in the 4343 sentences of the test sets were annotated. Therefore, we also evaluated our approach on the dedicated continuous-learning test set (*CL test*), created by the new train-test split.

The improvements by character-based representation on the CL test set are even larger. This might be due to the fact that there are more rare words in these sentences and therefore the advantages of the character-based models is stronger. Secondly, in this case, the one-shot approach improvements improve the translation quality. Since the improve-

---

[4] https://nlp-dke.github.io/data/rareWordNMT/

[5] https://github.com/nlp-dke/NMTGMinor

Represen-tation	One-Shot	CL Test		official Test	
		BLEU ↑	characTER ↓	BLEU ↑	characTER ↓
BPE	No	25.97	44.09	26.17	44.62
Character	No	28.12	42.79	26.57	44.27
Mix	No	27.44	42.79	26.83	44.28
BPE	Annot	26.00	41.74	26.21	44.73
Character	Annot	28.92	40.16	26.72	43.96
Mix	Annot	28.93	40.96	26.8	44.44

Table 4: Translation quality on TED tasks

Representation	One-Shot	Exact match			Lemma match			Morph. Adjustment		
		All	OneS	FewS	All	OneS	FewS	All	OneS	FewS
BPE	No	34	22	53	31	27	62	29	22	43
Character	No	48	40	60	55	47	68	45	43	48
Mix	No	42	35	54	49	40	63	38	34	46
BPE	Annot	48	34	69	62	46	88	33	24	50
Character	Annot	76	74	78	92	91	93	62	61	64
Mix	Annot	75	72	79	92	91	94	59	56	65

Table 5: Rare word accuracy on TED tasks

ments for the BPE-based system are only measured by characTER and not by BLEU might indicate that for this system it is more challenging to generate the correct inflected form.

To better analyse this, we also perform a detailed evaluation as described in Section 2.3 and shown in Table 5. First of all, the experiments show the difficulty of the task. The baseline system is only able to translate 34% of the phrases correctly. For the one-shot subset this even drops to 22%.

Secondly, the experiments show that the challenge can only successfully be addressed by modelling both: one-shot learning and word representation. On the last two lines using character-based word representation and one-shot learning are able to achieve high accuracy. We see an improvement by 50% percent absolutely, which is a relative improvement by more than 300%. Furthermore, for these models there is no longer a clear difference between the one-shot and few-shot examples (Comparison of Columns *OneS* and *FewS*).

By looking at them separately, we see that only using one-shot learning improves the quality slightly. However, even when ignoring the word infection, the model often is not able to produce the correct lemma. The example in Section 3.2, motivates one challenge when learning to copy with different subword segmentations. If we only use character-based representations, we see improvements, especially for phrases that do not occur in training. In this case, the model is more often able to find the correct translation based on translations of other words. However, a similar performance between the few-shot and one-shot learning is only achieved by combining both techniques.

Finally, when only looking at the words where the lemma is different from the inflected form, we still see open research challenges. While we also could improve the accuracy from around 20% or 30% to nearly 60%, it is still the most difficult case.

While there is no clear difference between the character-based model and the mixed model on the output quality, there is a clear difference in training speed. For the full training on 64 epochs, the character-based model needs 14h, while the mixed representation only needs around 4h. While this is still slower than the subword-based model (2.5h), it still allows for a fast training of the model. Therefore, we only compared the mixed and the sub-word based representation for the remaining experiments on larger corpora.

4.4 Europarl

In a second set of experiments, we evaluated the approach on the larger data set on two different language pairs. In addition to the two word representation from the last experiment (*BPE* and *Mix*), we also applied *Mix+Ann*, where we also represent

Lang.	Representation	One-Shot	CL Test		official Test	
			BLEU ↑	characTER ↓	BLEU ↑	characTER ↓
Ger.	BPE	No	28.74	47.30	25.30	48.75
	Mix	No	30.83	45.80	25.52	48.48
	BPE	Annot	28.74	47.47	25.49	48.64
	Mix	Annot	31.63	44.64	25.45	48.60
	BPE	add.Annot	28.81	47.24	25.45	48.71
	Mix	add.Annot	31.44	44.75	25.50	48.57
	Mix+Ann	add.Annot	31.76	44.11	25.64	48.41
Cz	BPE	No	34.25	39.43	16.2	57.15
	BPE	Annot	34.73	38.39	15.57	57.59
	Mix+Ann	Annot	34.86	38.16	16.62	57.70
	BPE	add.Annot	34.74	38.89	15.7	57.37
	Mix+Ann	add.Annot	35.21	37.95	16.63	57.65

Lang.	Representation	One-Shot	Exact match			Lemma match			Morph. Adjustment		
			All	OneS	FewS	All	OneS	FewS	All	OneS	FewS
Ger.	BPE	No	32	28	42	39	33	50	28	23	37
	Mix	No	42	38	48	50	38	58	37	34	43
	BPE	Annot	47	40	61	61	52	80	35	39	47
	Mix	Annot	66	65	68	83	81	88	51	47	56
	BPE	add.Annot	51	49	55	65	62	70	37	36	38
	Mix	add.Annot	65	63	69	81	78	88	51	40	56
	Mix+Ann	add.Annot	72	72	72	92	91	94	58	56	60
Cz	BPE	No	34	25	53	44	32	67	33	24	51
	BPE	Annot	46	33	70	63	48	92	42	30	67
	Mix+Ann	Annot	64	61	69	92	91	95	60	58	65
	BPE	add.Annot	45	31	71	61	45	91	41	29	58
	Mix+Ann	add.Annot	66	63	72	92	89	95	63	60	70

Table 6: Translation Performance on the Europarl data set

all words within dictionary entries as characters as described in Section 3.2. Furthermore, we also investigate *add. Annot*, where additional dictionary entries were used for more training examples. The results are shown in Table 6.

The overall picture for these experiments and the previous experiments is quite similar. For all three scenarios, the quality of the various systems on the official test sets is relatively similar, however the systems differ when looking especially at the accuracy of translating the dictionary entries. Only when combining one-shot learning with character-based representation, we are able to successfully translate the dictionary entries. Independent of the language pair and data size, we are able to achieve an accuracy of around 70% and an accuracy of around 90% when only looking at the lemmas only. Furthermore, the model performs as good in one-shot learning as in few-shot learning.

However, beside the evidence that the approach works on various language pairs and data sizes, the additional experiments give some more insights. First, although the data is larger, we do not see a difference between the models using additional annotation and the models using only the baseline annotation. So it seems to be sufficient to have around 1000 examples in order to learn to copy the suggestions from the source sentence.

Furthermore, although there are no longer clear improvements for character-based representation on the overall translation performance, also for this experiment with larger data size these representations are essential for the dictionary integration. This is highlighted by the improvements of using characters for all words in dictionary entries (*Mix+Ann*) instead of only for rare words (*Mix*).

5 Related work

In recent years, several different approaches to integrate additional data into neural machine translation have been suggested. If this is parallel data, fine-tuning on the additional, better matching data (Luong and Manning, 2015; Lavergne et al., 2011) is often successful. If the additional data is provided in other forms, different techniques have been investigated.

For human feedback, Turchi et al. (2017) suggested to use fine-tuning on the human generated post edits. Pham et al. (2018) used phrase pairs extracted by statistical machine translation to annotate translations of rare phrases. In the similar scenario Li et al. (2019) used a neural network to store the external phrase pairs.

Even more work has been done to integrate dictionaries into neural machine translation. A first work by Arthur et al. (2016) used the additional dictionary to influence the softmax probabilities of the neural machine translation. Another possibility is to include the dictionary as an additional knowledge source during training using posterior regularization (Zhang et al., 2017). A different approach is chosen by Zhang and Zong (2016) using the dictionary as additional training sentences or generating synthetic sentences. In contrast to this work, these do not allow the integration of new words after training the NMT system.

Several authors investigate the integration of the dictionary as an additional constraint during the coding process (Chatterjee et al., 2017; Hokamp and Liu, 2017; Hasler et al., 2018). This leads to a larger complexity in decoding that has been addressed by Post and Vilar (2018). However, the dictionary is typically a hard constraint which makes it difficult to learn words forms that do not occur in the dictionary.

Most similar to this work is the approach by Dinu et al. (2019), which like this work and Pham et al. (2018) annotates the source sentence with possible translations. They showed that state-of-the-art models no longer need architecture changes, but can directly learn to copy form the source sentences. In this work, we additionally focus on generating new word morphological forms not occurring in the dictionary. We investigated different word representations and analysed their influence on the ability to copy the dictionary entries.

6 Conclusion

By introducing the new continuous learning test set using a different train-test split for existing corpora we could highlight the challenges of state-of-the-art neural machine translation systems. While they achieve very good performance, they are still challenged by new emerging terms. The baseline system was only able to correctly translate 20 to 30 percent of these phrases.

Our integration of bilingual dictionaries into the systems improves the translation performance to correctly translate the words by up to 70%. In 90% of the cases at least the lemma of the word is predicted correctly. Furthermore, in this case, we see no difference in accuracy between words only seen in the dictionary and words also seen a few times in the parallel data. However this is only possible by modelling both: enabling the model to perform one-shot learning and modeling the different morphological forms of the rare phrases. The first one is addressed by annotating the source sentence with dictionary translation while the second one is addressed by using character-based models. By combining character-based and word-based representations we are able to model the different morphological variants of a word as well as enabling the system for fast training.

As mentioned before, this work concentrates on the morphological variants of the dictionary entries and ignores ambiguities due to different possible translation. In the future, we intend to address this by including word sense disambiguation into the translation process.

References

Philip Arthur, Graham Neubig, and Satoshi Nakamura. 2016. Incorporating Discrete Translation Lexicons into Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Austin, Texas. Association for Computational Linguistics.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 Conference on Machine Translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

M. Cettolo, C. Girardi, and M. Federico. 2012. WIT 3 :

Web Inventory of Transcribed and Translated Talks. In *Proceedings of the 16th EAMT Conference*, pages 261–268, Trento, Italy.

Rajen Chatterjee, Matteo Negri, Marco Turchi, Marcello Federico, Lucia Specia, and Frédéric Blain. 2017. Guiding Neural Machine Translation Decoding with External Knowledge. In *Proceedings of the Second Conference on Machine Translation*, pages 157–168, Copenhagen, Denmark. Association for Computational Linguistics.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training Neural Machine Translation to Apply Terminology Constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1027–1035, Barcelona, Spain. Curran Associates Inc.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural Machine Translation Decoding with Terminology Constraints. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512, New Orleans, Louisiana. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Catherine Kobus, Josep Crego, and Jean Senellart. 2017. Domain Control for Neural Machine Translation. In *Proceddings of Recent Advances in Natural Language Processing (RANLP 2017)*, pages 372–378, Varna, Bulgaria.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings The Tenth Machine Translation Summit (MT Summit X)*, Phuket, Thailand.

Thomas Lavergne, Hai-Son Le, Alexandre Allauzen, and François Yvon. 2011. LIMSI's experiments in domain adaptation for IWSLT11. In *Proceedings of the International Workshop on Spoken Langugage Translation*, pages 62–67, San Francisco, CA.

Ya Li, Xinyu Liu, Dan Liu, Xueqiang Zhang, and J. Liu. 2019. Learning Efficient Lexically-Constrained Neural Machine Translation with External Memory. *ArXiv*, abs/1901.11344.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford Neural Machine Translation Systems for Spoken Language Domains. In *Proceedings of the Twelfth International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam.

Jan Niehues, Eunah Cho, Thanh Le Ha, and Alex Waibel. 2016. Pre-translation for neural machine translation. In *COLING 2016 - 26th International Conference on Computational Linguistics, Proceedings of COLING 2016: Technical Papers*, pages 1828–1836, Osaka, Japan.

Jan Niehues and Alex Waibel. 2011. Using Wikipedia to translate domain-specific terms in SMT. In *Proceedings of the 8th International Workshop on Spoken Language Translation (IWSLT 2011)*, pages 230–237.

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

Ngoc-Quan Pham, Thai-Son Nguyen, Jan Niehues, Markus Müller, Sebastian Stüker, and Alexander Waibel. 2019. Very Deep Self-Attention Networks for End-to-End Speech Recognition. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (Interspeech 2019)*, Graz, Austria.

Ngoc-Quan Pham, Jan Niehues, and Alex Waibel. 2018. Towards one-shot learning for rare-word translation with external experts. In *Proceedings of the Second Workshop on Neural Machine Translation*, Melbourne, Australia.

Matt Post and David Vilar. 2018. Fast Lexically Constrained Decoding with Dynamic Beam Allocation for Neural Machine Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Rico Sennrich. 2017. How Grammatical is Character-level Neural Machine Translation? Assessing MT Quality with Contrastive Translation Pairs. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 376–382, Valencia, Spain. Association for Computational Linguistics.

Rico Sennrich, Alexandra Birch, and Barry Haddow. 2016a. Controlling Politeness in Neural Machine Translation via Side Constraints. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, pages 35–40, San Diego, California, USA.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Marco Turchi, Matteo Negri, Amin Farajian, and Marcello Federico. 2017. Continuous Learning from Human Post-Edits for Neural Machine Translation. *The Prague Bulletin of Mathematical Linguistics*, 108.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. CharacTer: Translation Edit Rate on Character Level. In *Proceedings of the First Conference on Statistical Machine Translation (WMT 2016)*, pages 505–510, Berlin, Germany.

Jiacheng Zhang, Yang Liu, Huanbo Luan, Jingfang Xu, and Maosong Sun. 2017. Prior Knowledge Integration for Neural Machine Translation using Posterior Regularization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1514–1523, Vancouver, Canada. Association for Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Bridging Neural Machine Translation and Bilingual Dictionaries.

Adv-OLM: Generating Textual Adversaries via OLM

Vijit Malik **Ashwani Bhat** **Ashutosh Modi**
Indian Institute of Technology Kanpur (IIT Kanpur)
{vijitvm, bashwani}@iitk.ac.in
ashutoshm@cse.iitk.ac.in

Abstract

Deep learning models are susceptible to adversarial examples that have imperceptible perturbations in the original input, resulting in adversarial attacks against these models. Analysis of these attacks on the state of the art transformers in NLP can help improve the robustness of these models against such adversarial inputs. In this paper, we present **Adv-OLM**, a black-box attack method that adapts the idea of **O**cclusion and **L**anguage **M**odels (OLM) to the current state of the art attack methods. OLM is used to rank words of a sentence, which are later substituted using word replacement strategies. We experimentally show that our approach outperforms other attack methods for several text classification tasks.

1 Introduction

In recent times, deep learning models have become pervasive across different domains. Many of the recent deep models have shown SOTA performance on a variety of NLP tasks (Wang et al., 2018). Consequently, deep models are being deployed in a variety of production systems for real-life applications. Hence, it becomes imperative to ensure the reliability and robustness of such models as it might pose a threat to security.

Recent studies have pointed out the vulnerability of deep models to adversarial attacks (Goodfellow et al., 2014). Adversarial attack comprises generating adversarial samples by performing small perturbations to the original input, making them imperceptible to humans while fooling the deep learning models to give incorrect predictions.

Adversarial attack on textual data is much more difficult due to the discrete nature of the text. The basic requirement of imperceptibility of perturbation by human judges is much more challenging in a language data setting. Therefore, the adversarial sample needs to be grammatically correct and semantically sound. Perturbations at word or character level that are perceptible to human judges have been explored in-depth (Ebrahimi et al., 2017; Belinkov and Bisk, 2017; Jia and Liang, 2017; Gao et al., 2018). Work on defense against misspellings based attacks (Pruthi et al., 2019) and use of optimization algorithms for attacks like genetic algorithm (Alzantot et al., 2018; Wang et al., 2019) and particle swarm optimization (Zang et al., 2020) have also been explored. With the rise of pre-trained language models, like BERT (Devlin et al., 2018) and other transformer-based models, generating human imperceptible adversarial examples has become more challenging. Wallace et al. (2019), Jin et al. (2019), and Pruthi et al. (2019) have explored these models from different perspectives.

Adversarial examples can be generated using black-box, where no knowledge about the model is accessible, and white-box, where information about the technical details of models are known. Generation of textual adversarial samples in a black-box setting consists of two steps 1) Finding words to replace in a sample (Word Ranking) 2) Replacing the chosen word (Word Replacement). Word Ranking is necessary to ensure that the word that contributes the most to the output prediction is considered as the candidate for replacement in the next step. Other constraints like generating semantically similar adversarial samples, human imperceptibility, and minimal perturbation percentage are also considered. Previous work has obtained word ranking by performing deletion of words (e.g., BAE-R (Garg and Ramakrishnan, 2020), TextFooler (Jin et al., 2019)), and replacement of words with *[UNK]* token (e.g., BERT-Attack (Li et al., 2020)) and then ranking the words based on the output logits difference.

Recently in the model explainability domain, the method of Occlusion and Language Models (OLM)

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 841–849
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

(Harbecke and Alt, 2020) has been proposed, the authors argue that the data likelihood of the samples obtained after either deleting the token or replacement with *[UNK]* token is very low, which makes these methods unsuitable for determining relevance of the word towards the output probability. The authors propose the use of language models for calculating the relevance of the words in a sentence. Taking inspiration from OLM, we propose **Adv-OLM**, a black box attack method, that adapts the idea of OLM (as the Work Ranking Strategy) to find the relevant words to replace. We empirically show that OLM provides a better set of ranked words compared to the existing word ranking strategies for the generation of adversarial examples.

We summarize our contributions as follows:

- We propose a new method Adv-OLM, to rank words for generating adversarial examples.
- We empirically show that Adv-OLM has a higher success rate and lower perturbation percentage than previous attacking methods.

The implementation for the proposed approach is made available at the GitHub repository: `https://github.com/vijit-m/Adv-OLM`.

2 Problem Formulation

We are given a corpus consisting of n input samples, $\mathbb{X} = \{x_1, \ldots, x_n\}$ with corresponding labels $\mathbb{Y} = \{y_1, \ldots, y_n\}$ and a trained classification model $f(f : \mathbb{X} \rightarrow \mathbb{Y})$ that maps an input samples to its correct label. We assume a black-box setting where the attacker can only query the classifier for output label probabilities for the given input. For an input sample $x \in \mathbb{X}$, the task is to construct an adversarial sample x' such that, $f(x) = y, \quad f(x') = y'$ with $\quad y \neq y'$, and $Similarity(x', x) \geq \epsilon$. Here, $Similarity : \mathbb{X} \times \mathbb{X} \rightarrow (0, 1)$ can be both the semantic and syntactic similarity function, and ϵ is the minimum similarity threshold. Ideally, the amount of perturbation should be minimized. The first step is to rank the words of the sample x. Based on the ranking, starting from the most important word, the word is replaced by some candidate word that keeps the perturbed sample x' semantically similar and grammatically sound but changes the output prediction.

3 Methodology

Adv-OLM uses the idea of Occlusion and Language Models to perform Word Ranking using both OLM and OLM-S methods. OLM uses a language model to sample some candidate instances for a word and then replaces the word. Let x_i be a word of the input x and $x_{\setminus i}$ be the incomplete input without this word. Then the OLM relevance score r given the prediction function f and label y is (Here f_y is the logit value corresponding to the label y.)

$$r_{f,y}(x_i) = f_y(x_i) - f_y(x_{\setminus i}) \tag{1}$$

Here, $f_y(x_{\setminus i})$ is not accurately defined and needs to be approximated since $x_{\setminus i}$ is the incomplete input. A language model p_{LM} generates input by predicting the masked word as $\hat{x}_i$ that is as natural as possible for the model and thus approximates to:

$$f_y(x_{\setminus i}) \approx \sum_{\hat{x}_i} p_{LM}(\hat{x}_i | x_{\setminus i}) f_y(x_{\setminus i} \cup \hat{x}_i) \tag{2}$$

where, $f_y(x_{\setminus i} \cup \hat{x}_i)$ is the prediction of the classification model after the language model's prediction $\hat{x}_i$ is added to the incomplete input $x_{\setminus i}$.

The other method OLM-S calculates the sensitivity of a position in the text and has nothing to do with the word present at that position in the original input. The sensitivity score of OLM-S is calculated

$$s_{f,y}(x_i) = \sqrt{\textstyle\sum_{\hat{x}_i} p_{LM}(\hat{x}_i | x_{\setminus i})(f_y(x_{\setminus i} \cup \hat{x}_i) - \mu)^2}$$

where μ is the mean value from Equation 2. The sensitivity score $s_{f,y}(x_i)$ is used for word ranking in OLM-S.

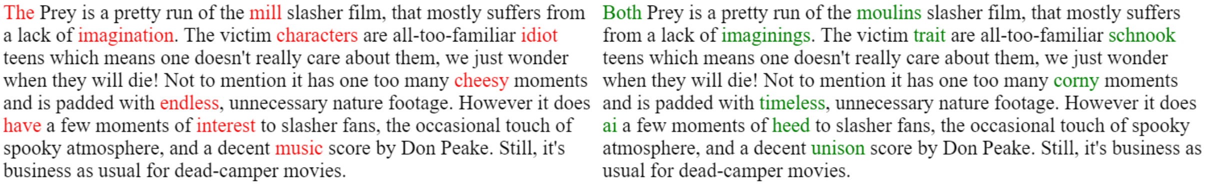
After performing the Word Ranking step using the relevance scores generated by OLM and OLM-S, the next step is to replace highly scored words with semantically similar words that form grammatically correct sentences (Word Replacement) such that the output prediction changes. Word replacement strategy is kept similar to existing methods. TextFooler uses Synonym Extraction, POS checking and semantic similarity checking whereas BAE-R uses a Language Model for word replacement. (details in Appendix C).

Dataset	Classes	Train	Test	Avg. Length
Classification				
IMDB	2	25K	500	245.12
Yelp	2	560K	500	132.33
AG's News	4	120K	500	40.41
Natural Language Inference				
MNLI	3	433K	500	29.72

Table 1: Statistic of Datasets. Avg. Length is the average number of words in the test set.

4 Experiments

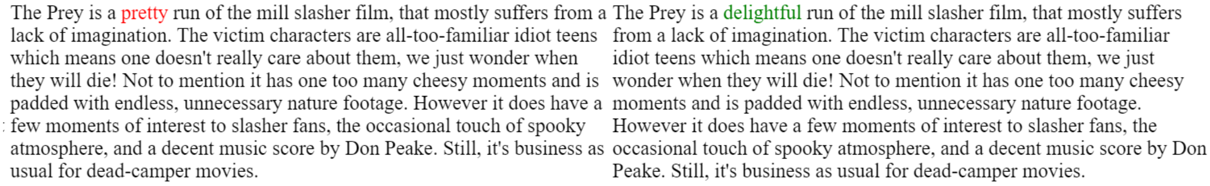
We experiment with different benchmark datasets for text classification and entailment: IMDB, AG News, Yelp Polarity and MNLI (details in Ap-



The Prey is a pretty run of the mill slasher film, that mostly suffers from a lack of imagination. The victim characters are all-too-familiar idiot teens which means one doesn't really care about them, we just wonder when they will die! Not to mention it has one too many cheesy moments and is padded with endless, unnecessary nature footage. However it does have a few moments of interest to slasher fans, the occasional touch of spooky atmosphere, and a decent music score by Don Peake. Still, it's business as usual for dead-camper movies.

Both Prey is a pretty run of the moulins slasher film, that mostly suffers from a lack of imaginings. The victim trait are all-too-familiar schnook teens which means one doesn't really care about them, we just wonder when they will die! Not to mention it has one too many corny moments and is padded with timeless, unnecessary nature footage. However it does ai a few moments of heed to slasher fans, the occasional touch of spooky atmosphere, and a decent unison score by Don Peake. Still, it's business as usual for dead-camper movies.

(a) TextFooler Attack on fine-tuned BERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(59\%)]$



The Prey is a pretty run of the mill slasher film, that mostly suffers from a lack of imagination. The victim characters are all-too-familiar idiot teens which means one doesn't really care about them, we just wonder when they will die! Not to mention it has one too many cheesy moments and is padded with endless, unnecessary nature footage. However it does have a few moments of interest to slasher fans, the occasional touch of spooky atmosphere, and a decent music score by Don Peake. Still, it's business as usual for dead-camper movies.

The Prey is a delightful run of the mill slasher film, that mostly suffers from a lack of imagination. The victim characters are all-too-familiar idiot teens which means one doesn't really care about them, we just wonder when they will die! Not to mention it has one too many cheesy moments and is padded with endless, unnecessary nature footage. However it does have a few moments of interest to slasher fans, the occasional touch of spooky atmosphere, and a decent music score by Don Peake. Still, it's business as usual for dead-camper movies.

(b) Adv-OLM Attack on fine-tuned BERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(95\%)]$

Figure 1: Qualitative Examples of TextFooler and Adv-OLM on BERT classifier (Red words are replaced by Green words while changing the output prediction probability.)

Dataset	**Method**	**Word Ranking**	**BERT**				**ALBERT**			
			Original Acc.	**Attacked Acc.**	**Success Rate**	**Perturbed %**	**Original Acc.**	**Attacked Acc.**	**Success Rate**	**Perturbed %**
AG's News	BAE-R	OLM	93.8%	**78%**	**16.84%**	6.72%	94.4%	**79.2%**	**16.1%**	8.37%
		OLM-S		79%	15.78%	**6.35%**		82.4%	12.71%	**6.9%**
		Original (delete)		78.8%	15.99%	6.42%		79.4%	15.89%	7.67%
	TextFooler	OLM		**19.2%**	**79.53%**	23.52%		20.4%	78.39%	21.24%
		OLM-S		21.4%	77.19%	**20.96%**		**20.2%**	**78.6%**	**20.18%**
		Original (delete)		21.4%	77.19%	23.19%		22.0%	76.69%	21.15%
	PWWS	-		44.8%	52.24%	16.21%		36.8%	61.02%	14.8%
Yelp	BAE-R	OLM	97.2%	**38.4%**	**60.49%**	**6.45%**	97.4%	41.4%	57.49%	7.07%
		OLM-S		55.2%	43.21%	10.36%		39.6%	59.34%	7.11%
		Original (delete)		41.6%	57.20%	7.28%		35.0%	64.07%	6.49%
	TextFooler	OLM		**5.2%**	**94.65%**	**9.10%**		**2.6%**	**97.33%**	10.17%
		OLM-S		7.8%	91.98%	13.31%		2.8%	97.13%	10.13%
		Original (delete)		6.6%	93.21%	9.95%		3.8%	96.1%	9.57%
	PWWS	-		6.2%	93.62%	6.9%		3.8%	96.1%	6.82%

Table 2: Comparison between word ranking strategies on AG's News and Yelp for fine-tuned BERT and ALBERT. Our method Adv-OLM has OLM (or OLM-S) as the word ranking strategy.

Dataset	**Method**	**Word Ranking**	**Original Acc.**	**Attacked Acc.**	**Success Rate**	**Perturbed %**
MNLI	BAE-R	OLM	84.6%	20.6%	75.65%	7.82%
		OLM-S		20.8%	75.41%	8.22%
		Original (delete)		14.0%	83.45%	6.4%
	TextFooler	OLM		**6.6%**	**92.2%**	8.29%
		OLM-S		7.0%	91.73%	8.59%
		Original (delete)		6.8%	91.96%	6.98%
	PWWS	-		3.2%	96.22%	6.62%

Table 3: Comparsion between previous methods and Adv-OLM on MNLI fine-tuned BERT. Our method Adv-OLM has OLM (or OLM-S) as the word ranking strategy.

pendix A). The statistics of the final dataset are shown in Table 1. Test set was randomly choosen stratified set. For evaluating the effectiveness of our proposed approach, we experiment with SOTA text classifiers i.e. transformer based models like BERT (Devlin et al., 2018), ALBERT (Lan et al., 2019), RoBERTa (Liu et al., 2019) and DistilBERT (Sanh et al., 2019).

We replaced the existing word ranking strategies (i.e. Original (delete)) of previous attack methods: Textfooler (Jin et al., 2019) and BAE-R (Garg and Ramakrishnan, 2020) with word rankings generated using OLM and OLM-S while keeping rest of the attack procedure same. The

Model	Method	Word Ranking	Original Acc.	Attacked Acc.	Success Rate	Perturbed %
BERT	BAE-R	OLM	92.0%	38.8%	57.83%	2.79%
		OLM-S		**33.2%**	**63.91%**	**2.33%**
		Original (delete)		42.6%	53.70%	2.92%
	Textfooler	OLM		29.8%	67.61%	4.52%
		OLM-S		**26.4%**	**71.3%**	**3.16%**
		Original (delete)		31.8%	65.43%	5.26%
	PWWS	-		28.2%	69.35%	2.92%
ALBERT	BAE-R	OLM	92.8%	27.4%	70.47%	3.58%
		OLM-S		**23.2%**	**75.0%**	**3.4%**
		Original (delete)		26.8%	71.12%	3.48%
	Textfooler	OLM		**1.4%**	**98.49%**	6.65%
		OLM-S		**1.4%**	**98.49%**	**5.39%**
		Original (delete)		2.4%	97.41%	6.57%
	PWWS	-		3.8%	95.91%	3.76%
RoBERTa	BAE-R	OLM	94.2%	29.4%	68.79%	4.08%
		OLM-S		**28.0%**	**70.28%**	**3.57%**
		Original (delete)		29.4%	68.79%	3.9%
	Textfooler	OLM		**0.0%**	**100%**	7.62%
		OLM-S		0.2%	99.79%	**6.43%**
		Original (delete)		0.2%	99.79%	6.89%
	PWWS	-		0.4%	99.58%	5.38%
DistilBERT	BAE-R	OLM	91.8%	22.6%	75.38%	3.29%
		OLM-S		21.6%	76.47%	**2.93%**
		Original (delete)		21.6%	76.47%	3.34%
	Textfooler	OLM		0.2%	99.78%	4.03%
		OLM-S		0.2%	99.78%	**3.55%**
		Original (delete)		0.2%	99.78%	4.44%
	PWWS	-		0.6%	99.35%	3.0%
BiLSTM	BAE-R	OLM	83.8%	10.8%	87.11%	2.78%
		OLM-S		10.4%	87.59%	2.71%
		Original (delete)		8.6%	89.74%	2.52%
	Textfooler	OLM		0%	100%	2.38%
		OLM-S		0%	100%	2.41%
		Original (delete)		0%	100%	1.95%
	PWWS	-		0%	100%	1.63%

Table 4: Comparison between previous methods and Adv-OLM for IMDB dataset across different models. Our method Adv-OLM has OLM (or OLM-S) as the word ranking strategy.

comparison is provided between the attacks generated through original word ranking, and OLM adapted word ranking (including comparison with PWWS attack method (Ren et al., 2019)) in table 2, table 3 and table 4. PWWS (Probability Weighted Word Saliency) method considers the word saliency along with the classification probability. The change in value of the classification probability is used to measure the attack effect of the proposed substitute word, while word saliency shows how well the original word affects the classification. We use the default language model (BERT) employed in the OLM and OLM-S, and kept the number of samples generated by the OLM language model as 30 in all the experiments.

The following evaluation metrics are used:

- **Attacked Acc.**: Accuracy of the model after attack. Lower the better.
- **Success Rate:** Ratio of number of successful attacks and the total number of attempted attacks[1]. Higher the better.
- **Perturbed Percentage:** Ratio of number of words that were modified by the attack and the total number of words in the input example. Lower the better.

We use TextAttack's (Morris et al., 2020) fine tuned models on these datasets and used it to execute the attacks, including Adv-OLM (Appendix B).

Number of queries in Adv-OLM: From equations 2 and 3, it is clear that unlike other methods of deletion and *[UNK]* token replacement, which perform only a single query, we need to perform multiple queries. We set the number of samples

---

[1] Note that total number of attempted attacks are not the same as number of input examples i.e., the samples which were originally wrongly classified by the model even before an attack are skipped

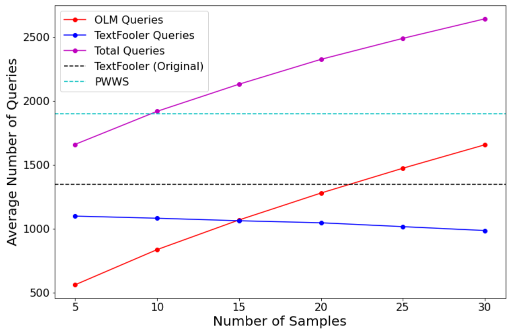


Figure 2: Average Number of Queries vs Number of samples in Adv-OLM attack on BERT on IMDB dataset.

generated by the OLM language model to 30 for our experiment. In the worst case, we would have all 30 samples of the token as unique, which will query the model 30 times. However, experimentally it was not the case. To study this, we varied the number of samples and evaluated the OLM ranking step's number of queries. In fig 2, we plotted the number of queries for OLM averaged over the input samples against the number of samples. We can see that there is not a significant difference in the total number of samples in our case (OLM + Textfooler queries) when compared with PWWS.

5 Results and Analysis

Results are shown in Tables 2, 3 and 4. Table 2 provides the results on AG News and Yelp datasets on fine-tuned BERT and ALBERT model. Our method performs better on both datasets by increasing the success rate by about 1-3% than the previous methods and also decreasing the perturbation percentage. Table 3 gives the results of attacking a fine-tuned BERT on MNLI. Although we did not perform better than original BAE-R, we were still able to outperform TextFooler. Due to the unavailability of MNLI fine-tuned ALBERT model in TextAttack, we did not perform an attack on ALBERT. It can also be seen from Table 2 that the perturbation percentage for AG's News exceed more than 20%, which seems to be a perceptible change, but since the average length of the article is only 40.41, making the space for finding relevant words less, the perturbation percentage becomes very high.

To compare attacks across different transformer-based models, we evaluate the performance of Adv-OLM on IMDB dataset. Table 4 provides the results of different attack methods on BERT, ALBERT, RoBERTa, DistilBERT and BiLSTM. Adv-OLM was able to outperform previous attack methods on BERT, ALBERT, RoBERTa by increasing the success rate up to 10% for BAE-R and up to 6% for TextFooler. Perturbation percentage was also reduced by 1-2%. On DistilBERT, Adv-OLM showed no change in the success rate, but the perturbation percentage was lowered slightly. We also performed an attack on a non-transformer based BiLSTM model which did not show any improvements in the success rate. For BAE-R, it even showed a decrease in the success rate for Adv-OLM. One possible reason for this might be that in both OLM and OLM-S word sampling is performed using a transformer-based BERT language model. We also have qualitative results on IMDB dataset (Figure 1a, 1b).

Experimentally it was observed that better words were ranked when OLM/OLM-S was used as the Word Ranking strategy (Figure 1b). When comparing with the original methods, Adv-OLM has more number of queries, which is due to the fact that for word rankings, OLM/OLM-S queries the model a number of times, thus increasing the overall queries. However, the difference in the number of Adv-OLM queries with the existing attacking methods is not very significant since the model is queried only for unique words from the samples generated from the language model.

6 Conclusion

In this work, we present **Adv-OLM**, a black box attacking method that uses OLM based word ranking strategy, improving the attack performance significantly over previous methods. We also studied how replacing a single variable in a complex system with a new existing method can improve upon the previously existing attack strategies. For future work, we would like to experiment with other language models in the OLM algorithm. We plan to study the effect of using different transformers for the language model and the target model.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*.

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*.

Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, and Cho-Jui Hsieh. 2020. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. In *AAAI*, pages 3601–3608.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. *arXiv preprint arXiv:1804.07781*.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. *arXiv preprint arXiv:2004.01970*.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

David Harbecke and Christoph Alt. 2020. Considering likelihood in nlp classification explanations with occlusion and language modeling. *arXiv preprint arXiv:2004.09890*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. *arXiv preprint arXiv:1909.00986*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Volodymyr Kuleshov, Shantanu Thakoor, Tingfung Lau, and Stefano Ermon. 2018. Adversarial examples for natural language classification problems.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. *arXiv preprint arXiv:2004.09984*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. 2019. Combating adversarial misspellings with robust word recognition. *arXiv preprint arXiv:1905.11268*.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Xiaosen Wang, Hao Jin, and Kun He. 2019. Natural language adversarial attacks and defenses in word level. *arXiv preprint arXiv:1909.06723*.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080.

Appendix

A Datasets

We evaluate our adversarial attacks on text classification and natural language inference datasets. We evaluate our method on 500 samples randomly selected from the test set of the given dataset.

Text classification We used the following text classification datasets:

- **IMDB:** Document-level large Movie Review dataset for binary sentiment classification. [2]
- **Yelp:** The Yelp reviews dataset consists of reviews from Yelp. This is a dataset for sentiment classfication. It is extracted from the Yelp Dataset Challenge 2015 data. [3]
- **AG's News:** Sentence level news-type classification dataset, containing 4 types of news: World, Sports, Business, and Science. [4]

Natural Language Inference

- **MNLI:** The corpus of sentence pairs manually labeled for classification with the labels entailment, contradiction, and neutral, supporting the task of natural language inference (NLI). Unlike SNLI, MNLI is more diverse, based on multi-genre texts, covering transcribed speech, popular fiction, and government reports. [5]

Average Length is the average number of words in the randomly chosen 500 samples taken from its test set for each dataset.

B Textattack

TextAttack is an open-source python framework for adversarial attacks, data augmentation and adversarial training in NLP.

Because of the modularity that TextAttack provides, it enables researchers to construct new attacks from a combination of novel and existing approaches or perform analysis on the already existing approaches. This helps in composing and comparing the attacks in a shared environment. TextAttack makes it easy to perform benchmark comparisons across all the previous attacks performed across models. Text Attack provides clean, readable implementations of 16 adversarial attacks from the literature. Out of which two are sequence to sequence attacks and nine are classification based attacks from the GLUE benchmark. A list of these attacks is presented in Table 5. TextAttack is directly integrated with HuggingFace's transformers and NLP libraries. This allows users to test attacks on models and datasets.

Attack Recipes	
BAE (Garg and Ramakrishnan, 2020)	PWWS (Ren et al., 2019)
Bert-Attack (Li et al., 2020)	TextFooler (Jin et al., 2019)
DeepwordBug (Gao et al., 2018)	HotFlip (Ebrahimi et al., 2017)
Alzantot (Alzantot et al., 2018)	Morpheus (Tan et al., 2020)
IGA (Wang et al., 2019)	Pruthi (Pruthi et al., 2019)
Input-Reduction (Feng et al., 2018)	PSO (Zang et al., 2020)
Seq2Sick (Cheng et al., 2020)	TextBugger (Li et al., 2018)
Kuleshov (Kuleshov et al., 2018)	Fast Alzantot (Jia et al., 2019)

Table 5: Adversarial attacks implemented in Textattack

TextAttack builds attacks from four components:

1. A **search method** that selects the words to be transformed.
2. A **transformation** that generates a set of possible perturbations for the given input.
3. A **set of constraints** implied on the transformation to ensure that the perturbations are valid with respect to the original input.
4. A **goal function** that determines whether an attack is successful in terms of model outputs. For classification tasks, untargeted, and targeted. For a sequence to sequence tasks, non-overlapping output, and minimum BLEU score.

For our approach, we attack TextAttack's fine-tuned models on datasets discussed in A, that are publically available on huggingface[6] Textattack is also used for the execution of all the previous attacking methods and our Adv-OLM as well.

C Word Replacement Strategies

C.1 TextFooler Word Replacement Strategy

Following workflow was proposed by the paper:

Synonym Extraction: Gather a candidate set CANDIDATES for all possible replacements of the selected word w_i and every other word in the vocabulary. To represent the words, counter fitting word embeddings were used. Using this set

---

[2] IMDB dataset

[3] Yelp dataset

[4] AG's News dataset

[5] MNLI dataset

[6] TextAttack fine-tuned models on HuggingFace

of embedding vectors, top N synonyms whose cosine similarity with w is higher than some δ were chosen.

POS Checking: In the set CANDIDATES of the word w_i, only the ones with the same part-of-speech(POS) as w_i were kept. This step assures that the grammar of the text is mostly maintained.

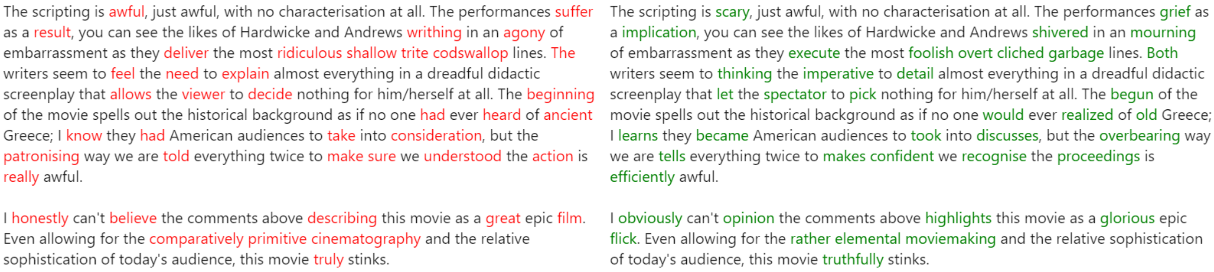
Semantic Similarity Checking: For each remaining word $c \in$ CANDIDATES, these were substituted for w_i in the sentence X, and an adversarial example X_{adv} was obtained. Universal Sentence Encoder (USE) was used to encode the two sentences into high dimensional vectors and then use their cosine similarity score to calculate the sentence similarity between X and X_{adv}. The words resulting in similarity scores above a preset threshold ϵ were placed in a final candidate pool (FINCANDIDATE).

Finally, every candidate word from the FINCANDIDATE was chosen one by one, and the one that resulted in the least confidence score of label y was considered as the best replacement for word w_i.

C.2 BAE-R Word Replacement Strategy

BAE uses a pre-trained BERT masked language model(MLM) to predict the mask tokens for replacement. Since BERT is powerful and trained on the large training corpus, the predicted mask tokens fit well grammatically in the sentence. BERT-MLM does not, however, guarantee semantic coherence to the original text. To ensure semantic similarity on introducing perturbations in the input text, a set of K masked tokens were filtered out using Universal Sentence Encoder(USE) based on sentence similarity score. An additional check for grammatical correctness of the generated adversarial example by filtering out predicted tokens that do not form the same part of speech(POS) as the original token in the sentence was performed.

D More Examples



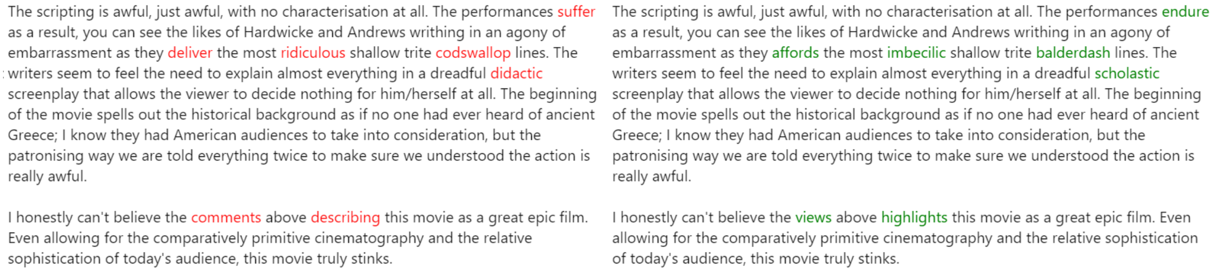
The scripting is awful, just awful, with no characterisation at all. The performances suffer as a result, you can see the likes of Hardwicke and Andrews writhing in an agony of embarrassment as they deliver the most ridiculous shallow trite codswallop lines. The writers seem to feel the need to explain almost everything in a dreadful didactic screenplay that allows the viewer to decide nothing for him/herself at all. The beginning of the movie spells out the historical background as if no one had ever heard of ancient Greece; I know they had American audiences to take into consideration, but the patronising way we are told everything twice to make sure we understood the action is really awful.

I honestly can't believe the comments above describing this movie as a great epic film. Even allowing for the comparatively primitive cinematography and the relative sophistication of today's audience, this movie truly stinks.

The scripting is scary, just awful, with no characterisation at all. The performances grief as a implication, you can see the likes of Hardwicke and Andrews shivered in an mourning of embarrassment as they execute the most foolish overt cliched garbage lines. Both writers seem to thinking the imperative to detail almost everything in a dreadful didactic screenplay that let the spectator to pick nothing for him/herself at all. The begun of the movie spells out the historical background as if no one would ever realized of old Greece; I learns they became American audiences to took into discusses, but the overbearing way we are tells everything twice to makes confident we recognise the proceedings is efficiently awful.

I obviously can't opinion the comments above highlights this movie as a glorious epic flick. Even allowing for the rather elemental moviemaking and the relative sophistication of today's audience, this movie truthfully stinks.

(a) TextFooler Attack on fine-tuned ALBERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(51\%)]$



The scripting is awful, just awful, with no characterisation at all. The performances suffer as a result, you can see the likes of Hardwicke and Andrews writhing in an agony of embarrassment as they deliver the most ridiculous shallow trite codswallop lines. The writers seem to feel the need to explain almost everything in a dreadful didactic screenplay that allows the viewer to decide nothing for him/herself at all. The beginning of the movie spells out the historical background as if no one had ever heard of ancient Greece; I know they had American audiences to take into consideration, but the patronising way we are told everything twice to make sure we understood the action is really awful.

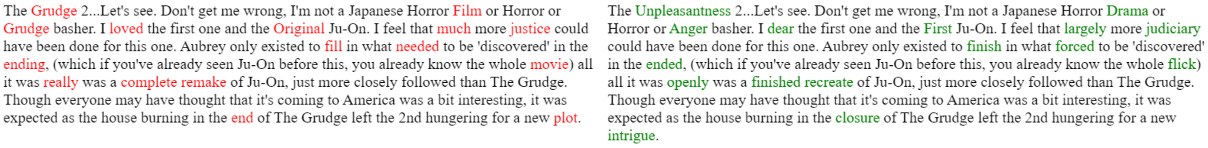
I honestly can't believe the comments above describing this movie as a great epic film. Even allowing for the comparatively primitive cinematography and the relative sophistication of today's audience, this movie truly stinks.

The scripting is awful, just awful, with no characterisation at all. The performances endure as a result, you can see the likes of Hardwicke and Andrews writhing in an agony of embarrassment as they affords the most imbecilic shallow trite balderdash lines. The writers seem to feel the need to explain almost everything in a dreadful scholastic screenplay that allows the viewer to decide nothing for him/herself at all. The beginning of the movie spells out the historical background as if no one had ever heard of ancient Greece; I know they had American audiences to take into consideration, but the patronising way we are told everything twice to make sure we understood the action is really awful.

I honestly can't believe the views above highlights this movie as a great epic film. Even allowing for the comparatively primitive cinematography and the relative sophistication of today's audience, this movie truly stinks.

(b) Adv-OLM attack on fine-tuned ALBERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(57\%)]$

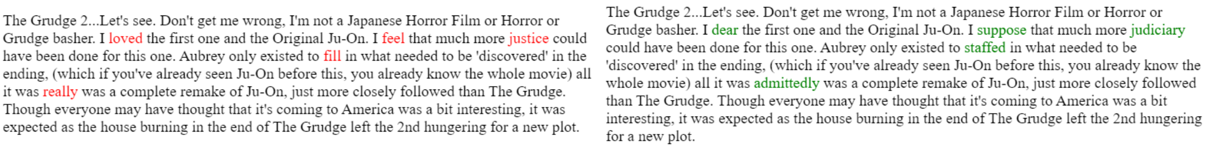
Figure 3: Qualitative Examples of TextFooler and Adv-OLM on ALBERT classifier (Red words are replaced by Green words while changing the output prediction probability.)



The Grudge 2...Let's see. Don't get me wrong, I'm not a Japanese Horror Film or Horror or Grudge basher. I loved the first one and the Original Ju-On. I feel that much more justice could have been done for this one. Aubrey only existed to fill in what needed to be 'discovered' in the ending, (which if you've already seen Ju-On before this, you already know the whole movie) all it was really was a complete remake of Ju-On, just more closely followed than The Grudge. Though everyone may have thought that it's coming to America was a bit interesting, it was expected as the house burning in the end of The Grudge left the 2nd hungering for a new plot.

The Unpleasantness 2...Let's see. Don't get me wrong, I'm not a Japanese Horror Drama or Horror or Anger basher. I dear the first one and the First Ju-On. I feel that largely more judiciary could have been done for this one. Aubrey only existed to finish in what forced to be 'discovered' in the ended, (which if you've already seen Ju-On before this, you already know the whole flick) all it was openly was a finished recreate of Ju-On, just more closely followed than The Grudge. Though everyone may have thought that it's coming to America was a bit interesting, it was expected as the house burning in the closure of The Grudge left the 2nd hungering for a new intrigue.

(a) TextFooler Attack on fine-tuned ALBERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(51\%)]$



The Grudge 2...Let's see. Don't get me wrong, I'm not a Japanese Horror Film or Horror or Grudge basher. I loved the first one and the Original Ju-On. I feel that much more justice could have been done for this one. Aubrey only existed to fill in what needed to be 'discovered' in the ending, (which if you've already seen Ju-On before this, you already know the whole movie) all it was really was a complete remake of Ju-On, just more closely followed than The Grudge. Though everyone may have thought that it's coming to America was a bit interesting, it was expected as the house burning in the end of The Grudge left the 2nd hungering for a new plot.

The Grudge 2...Let's see. Don't get me wrong, I'm not a Japanese Horror Film or Horror or Grudge basher. I dear the first one and the Original Ju-On. I suppose that much more judiciary could have been done for this one. Aubrey only existed to staffed in what needed to be 'discovered' in the ending, (which if you've already seen Ju-On before this, you already know the whole movie) all it was admittedly was a complete remake of Ju-On, just more closely followed than The Grudge. Though everyone may have thought that it's coming to America was a bit interesting, it was expected as the house burning in the end of The Grudge left the 2nd hungering for a new plot.

(b) Adv-OLM attack on fine-tuned ALBERT on IMDB data sample. $[Negative(100\%) \rightarrow Positive(65\%)]$

Figure 4: Qualitative Examples of TextFooler and Adv-OLM on RoBERTa classifier (Red words are replaced by Green words while changing the output prediction probability.)

Conversational Question Answering over Knowledge Graphs with Transformer and Graph Attention Networks

Endri Kacupaj[1], **Joan Plepi**[2], **Kuldeep Singh**[3], **Harsh Thakkar**[4],
Jens Lehmann[1,5], **and Maria Maleshkova**[1]

[1]Smart Data Analytics Group, University of Bonn, Bonn, Germany
[2]Technische Universität Darmstadt, Darmstadt, Germany
[3]Zerotha Research and Cerence GmbH, Germany
[4]Zerotha Research and Osthus GmbH, Germany
[5]Fraunhofer IAIS, Dresden, Germany

`{kacupaj,jens.lehmann,maleshkova}@cs.uni-bonn.de`
`joan.plepi@tu-darmstadt.de`
`kuldeep.singh1@cerence.com`
`harsh.thakkar@osthus.com`
`jens.lehmann@iais.fraunhofer.de`

Abstract

This paper addresses the task of (complex) conversational question answering over a knowledge graph. For this task, we propose LASAGNE (mu**L**ti-task sem**A**ntic par**S**ing with tr**A**nsformer and **G**raph atte**N**tion n**E**tworks). It is the first approach, which employs a transformer architecture extended with Graph Attention Networks for multi-task neural semantic parsing. LASAGNE uses a transformer model for generating the base logical forms, while the Graph Attention model is used to exploit correlations between (entity) types and predicates to produce node representations. LASAGNE also includes a novel entity recognition module which detects, links, and ranks all relevant entities in the question context. We evaluate LASAGNE on a standard dataset for complex sequential question answering, on which it outperforms existing baseline averages on all question types. Specifically, we show that LASAGNE improves the F1-score on eight out of ten question types; in some cases, the increase in F1-score is more than 20% compared to the state of the art.

1 Introduction

Since their inception in the late 2000s, publicly available Knowledge Graphs (e.g., DBpedia (Lehmann et al., 2015) and Yago (Suchanek et al., 2007)) have been widely used as a source of knowledge in several natural language processing (NLP) tasks such as entity linking, relation extraction, fact-checking, and question answering. Question answering (QA), in particular, is an essential task that maps a user natural language question to a query over a knowledge graph (KG) to retrieve the correct answer (Singh et al., 2018). With the increasing popularity of intelligent personal assistants (e.g., Alexa, Siri), the research focus has been shifted to conversational question answering that involves multi-turn dialogues, incorporating the phenomenon of anaphora and ellipses (Christmann et al., 2019; Shen et al., 2019)(c.f. Figure 1).

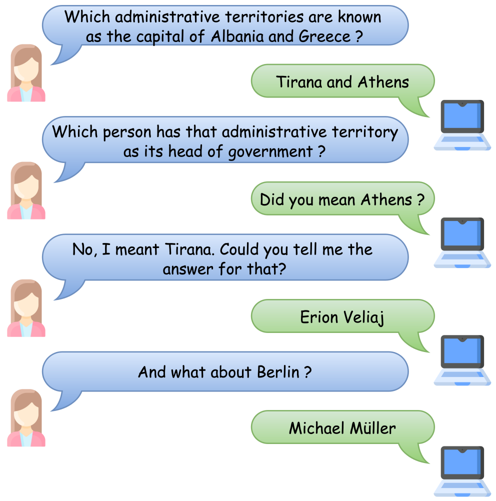


Figure 1: Conversational Question Answering task with examples similar to CSQA dataset (Saha et al., 2018).

Conversational QA is often realised by using semantic parsing approaches, mapping an utterance to a logic form for extracting answers from a KG (Guo et al., 2018; Shen et al., 2019). The state of the art for semantic parsing approaches decomposes the semantic parsing process into two stages (Shen et al., 2019). First, a logical form is generated based on low-level features and then

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 850–862
April 19 - 23, 2021.

the missing details are filled by considering both the question and the template. Other approaches (Dong and Lapata, 2016; Guo et al., 2018; Liang et al., 2017) first employ an entity linking model to identify entities in the question and subsequently use another model to map the question to a logical form. (Zhang et al., 2018; Shen et al., 2019) point out that the modular approaches suffer from the common issue of error propagation along the QA pipeline, resulting in accumulated errors. To mitigate these errors, Shen et al. (2019) proposed a multi-task framework, where a pointer-equipped semantic parsing model was designed to resolve coreference in conversations and empower joint learning with a type-aware entity detection model. Furthermore, the authors used simple classifiers to predict the required (entity) types and predicates for the generated logical forms. In this paper, we argue that Shen et al. (2019) model (the current SotA) has the following shortcomings: 1) the (entity) type and predicate classifiers share no common information, except for the supervision signal propagated to them. 2) Hence, due to missing common information, the model can produce ambiguous results, since the classifiers can predict entities and predicates that do not correlate with each other.

Approach and Contributions: We tackle the problem of conversational (complex) question answering over a large-scale knowledge graph. We propose LASAGNE (muLti-task semAntic parSing with trAnsformer and Graph atteNtion nEtworks) - a multi-task learning framework consisting of a transformer model extended with Graph Attention Networks (GATs) (Veličković et al., 2018) for multi-task neural semantic parsing. Our framework handles semantic parsing using the transformer (Vaswani et al., 2017) model similar to previous approaches. However, in LASAGNE we introduce the following two novel contributions: 1) the transformer model is supplemented with a Graph Attention Network to exploit the correlations between (entity) types and predicates due to its message-passing ability between the nodes. 2) We propose a novel entity recognition module that detects, links, filters, and permutes all relevant entities. (Shen et al., 2019) uses a pointer equipped decoder that learns and identifies the relevant entities for the logical form using only the encoder's information. In contrast, we use both sources of information, i.e., the entity detection module and the encoder, to filter and permute the relevant entities for a logical form. This avoids re-learning entity information in the current question context and relies on the entity detection module's information. Our empirical results show that the proposed novel contributions lead to substantial performance improvements.

LASAGNE achieves the state of the art results in 8 out of 10 question types on the Complex Sequential Question Answering (CSQA) (Saha et al., 2018) dataset consisting of conversations over linked QA pairs. The dataset contains 200K dialogues with 1.6M turns, and over 12.8M entities from Wikidata[1]. Our implementation, the annotated dataset with the proposed grammar, and the results are publicly available to facilitate reproducibility and reuse[2].

The structure of the paper is as follows: Section 2 summarises the related work. Section 3 presents the proposed LASANGE framework. Section 4 describes the experiments, including the experimental setup, the results, the ablation study and error analysis. We conclude in Section 5.

2 Related Work

We point to the survey by (Gao et al., 2018) that provides a holistic overview of neural approaches in conversational AI. In this paper, we stick to our closely related work, i.e., semantic parsing-based approaches in conversations. (Liang et al., 2017) introduce a neural symbolic machine (NSM) extended with a key-value memory network, where keys and values are the output of the sequence model in different encoding or decoding steps. The NSM model is trained using the REINFORCE algorithm with weak supervision and evaluated on the WebQuestionsSP dataset (Yih et al., 2016).

(Saha et al., 2018) propose a hybrid model of the HRED model (Serban et al., 2016) and the key-value memory network model (Miller et al., 2016). The model consists of three components. The first one is the Hierarchical Encoder, which computes a representation for each utterance. The next module is a higher-level encoder that computes a representation for the context. The second component is the Key-Value Memory Network. It stores each of the candidate tuples as a key-value pair where the key contains the concatenated embedding of the relation and the subject. In contrast, the value contains

---

[1] `https://www.wikidata.org/`

[2] `https://github.com/endrikacupaj/LASAGNE`

Action	Description
set → find(e, p)	set of objects part of the triples with subject e and predicate p
set → find_reverse(e, p)	set of subjects part of the triples with object e and predicate p
set → filter_type(set, tp)	filter the given set of entities based on the given type
set → filter_multi_types(set_1, set_2)	filter the given set of entities based on the given set of types
dict → find_tuple_counts(p, tp_1, tp_2)	extracts a dictionary, where keys are entities of $type_1$ and values are the number of objects of $type_2$ related with p
dict → find_reverse_tuple_counts(p, tp_1, tp_2)	extracts a dictionary, where keys are entities of $type_1$ and values are the number of subjects of $type_2$ related with p
set → greater(dict, num)	set of those entities that have greater count than *num*
set → lesser(dict, num)	set of those entities that have lesser count than *num*
set → equal(dict, num)	set of those entities that have equal count with *num*
set → approx(dict, num)	set of those entities that have approximately same count with *num*
set → atmost(dict, num)	set of those entities that have at most same count with *num*
set → atleast(dict, num)	set of those entities that have at least same count with *num*
set → argmin(dict)	set of those entities that have the most count
set → argmax(dict)	set of those entities that have the least count
boolean → is_in(entity, set)	check if the entity is part of the set
number → count(set)	count the number of elements in the set
set → union(set_1, set_2)	union of set_1 and set_2
set → intersection(set_1, set_2)	intersection of set_1 and set_2
set → difference(set_1, set_2)	difference of set_1 and set_2

Table 1: Predefined grammar with respective actions to generate logical forms.

the embedding of the object. The last component is the decoder used to create an end-to-end solution and produce multiple types of answers.

(Guo et al., 2018) present a model that converts an utterance in conversation to a logical form. The model follows a flexible grammar, in which the generation of a logical form is equivalent to predicting a sequence of actions. A dialogue memory management is proposed and integrated into the model, so that historical entities, predicates, and action sub-sequences can selectively be replicated. (Shen et al., 2019) proposed the first multi-task learning framework that learns type-aware entity detection and pointer-equipped logical form generation simultaneously. The multi-task learning framework takes advantage of the supervision from the subtasks.

3 LASAGNE

In a conversation, the input data consists of utterances u and their answers a, extracted from the knowledge graph. Our framework LASAGNE employs a multi-task semantic parsing approach. In particular, it maps the utterance u to a logical form z, depending on the conversation context. Figure 2 shows the architecture of LASAGNE.

3.1 Grammar

For the semantic parsing task, we propose a grammar that can be used to capture the entire context of the input utterance with the minimum number of actions. Table 1 illustrates the complete grammar with all the defined actions. We considered the work by (Guo et al., 2018) as a starting point for generating them, however, we have updated many of the semantic actions. For instance, for a couple of actions, we also define their reverse occurrence (e.g. *find*, *find_reverse*)).

3.2 Transformer

To translate the input conversation into a sequence of actions (logical form), we utilise a transformer model (Vaswani et al., 2017). Specifically, the transformer here aims to map a question q, that is a sequence $x = \{x_1, \ldots, x_n\}$, to the answer label l, that can be also defined as a sequence $y = \{y_1, \ldots, y_m\}$, by modelling the conditional probability $p(y|x)$.

3.2.1 Input and Word Embedding

We have to incorporate the dialog history from previous interactions as an additional input to our model for handling coreference and ellipsis. To do so, we consider the following utterances for each turn: 1) the previous question, 2) the previous answer, and 3) the current question. Utterances are separated from one another by using a $[SEP]$ token. At the end of the last utterance, we append a context token $[CTX]$, which is used as the semantic representation for the entire input question. In the next step, given an utterance q containing n words $\{w_1, \ldots, w_n\}$ we first tokenise the conversation context using WordPiece tokenization (Wu et al., 2016), and after that, we use the pre-trained model GloVe (Pennington et al., 2014) to embed the words into a vector representation space of di-

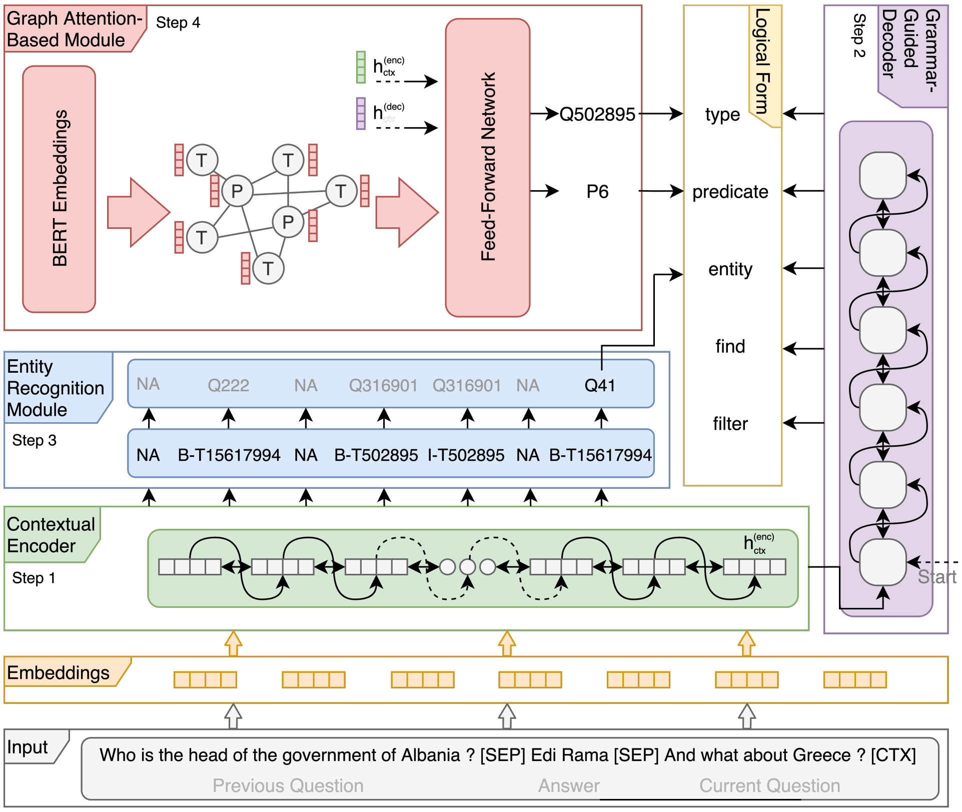


Figure 2: LASAGNE (Multi-task Semantic Parsing with Transformer and Graph Attention Networks) architecture. It consists of three modules: 1) A semantic parsing-based transformer model, containing a contextual encoder and a grammar guided decoder using the grammar defined in Table 1. 2) An entity recognition module, which identifies all the entities in the context, together with their types, linking them to the knowledge graph. It filters them based on the context and permutes them, in case of more than one required entity. Finally, 3) a graph attention-based module that uses a GAT network initialised with BERT embeddings to incorporate and exploit correlations between (entity) types and predicates. The resulting node embeddings, together with the context hidden state (h_{ctx}) and decoder hidden state (d_h), are used to score the nodes and predict the corresponding type and predicate.

mension d [3]. Our word embedding model provides us with a sequence $x = \{x_1, \ldots, x_n\}$ where x_i is given by, $x_i = GloVe(w_i)$ and $x_i \in \mathbb{R}^d$.

3.2.2 Contextual Encoder

The word embeddings x, are forwarded as input to the contextual encoder, which uses the multi-head attention mechanism described by (Vaswani et al., 2017). The encoder here outputs the contextual embeddings $h^{(enc)} = \{h_1^{(enc)}, \ldots, h_n^{(enc)}\}$, where $h_i^{(enc)} \in \mathbb{R}^d$ and it can be defined as:

$$h^{(enc)} = encoder(x; \theta^{(enc)}), \tag{1}$$

where $\theta^{(enc)}$ are the encoder's trainable parameters.

3.2.3 Grammar-Guided Decoder

We use a grammar guided decoder for generating the logical forms. The decoder also employs the multi-head attention mechanism. The decoder output is dependent on the encoder contextual embeddings h. The main task of the decoder is to generate each corresponding action, based on Table 1, alongside with the general semantic object from the knowledge graph (entity, type, predicate). In other words, the decoder will predict the main logical form without using or initialising any specific information from the knowledge graph. Here we define the decoder vocabulary as $V^{(dec)} = \{find, find\_reverse, \ldots, entity, type, predicate, value\}$, where all the actions from Table 1 are included. On top of the decoder stack, we employ a linear layer alongside a softmax to calculate each token's probability scores in the vocabulary. We define the decoder stack output as follows:

$$\begin{aligned} h^{(dec)} &= decoder(h^{(enc)}; \theta^{(dec)}), \\ p_t^{(dec)} &= softmax(\boldsymbol{W}^{(dec)} h_t^{(dec)}), \end{aligned} \tag{2}$$

---

[3] Across the model, we use the same dimension d for all the representations, unless it is explicitly noted.

where $h_t^{(dec)}$ is the hidden state in time step t, $\theta^{(dec)}$ are the decoder trainable parameters, $\boldsymbol{W}^{(dec)} \in \mathbb{R}^{|V^{(dec)}| \times d}$ are the linear layer weights, and $p_t^{(dec)} \in \mathbb{R}^{|V^{(dec)}|}$ is the probability distribution over the decoder vocabulary in time step t. The $|V^{(dec)}|$ denotes the decoder's vocabulary size.

3.3 Entity Recognition Module

The entity recognition module is composed of two sub-modules, where each module is trained using a different objective.

3.3.1 Entity Detection and Linking

Entity Detection It aims to detect and link the entities to the KG. The module is inspired by (Shen et al., 2019) and performs type-aware entity detection by using BIO sequence tagging jointly with entity type tagging. Specifically, the entity detection vocabulary is defined as $V^{(ed)} = \{O, \{B, I\} \times \{TP_i\}_{i=1}^{N^{(tp)}}\}$, where TP_i denotes the *i-th* entity type label, $N^{(tp)}$ stands for the number of the distinct entity types in the knowledge graph and $|V^{(ed)}| = 2 \times N^{(tp)} + 1$. For performing the sequence tagging task we use an LSTM (Hochreiter and Schmidhuber, 1997) and the module is defined as:

$$\begin{aligned} h^{(l)} &= LeakyReLU(LSTM(h^{(enc)}; \theta^{(l)})), \\ p_t^{(ed)} &= softmax(\boldsymbol{W}^{(l)} h_t^{(l)}), \end{aligned} \tag{3}$$

where $h^{(enc)}$ is the encoder hidden state, $\theta^{(l)}$ are the LSTM layer trainable parameters, $h_t^{(l)}$ is the LSTM hidden state for time step t, $\boldsymbol{W}^{(l)} \in \mathbb{R}^{|V^{(ed)}| \times d}$ are the linear layer weights and $p_t^{(ed)}$ are the entity detection module prediction for time step t. $|V^{(ed)}|$ denotes the entity detection vocabulary size.

Entity Linking Once the entity BIO labels and their types are recognised, the next steps for the entity linking are: 1) the BIO labels are used to locate the entity spans from the input utterances. 2) An inverted index built for the knowledge graph entities is used to retrieve candidates for each predicted entity span. Finally, 3) the candidate lists are filtered using the predicted (entity) types. From the filtered candidates, the first entity is considered as correct.

3.3.2 Filtering and Permutation

After finding all the input utterances' entities, we perform two additional tasks in order to use entities in the generated logical form. First, we filter the relevant entities, and then we need to permute the entities in the order required for the logical form. The module receives as an input the concatenation of the hidden states of the encoder $h^{(enc)}$ and the hidden states of the LSTM $h^{(l)}$ from the entity detection model. The module here learns to assign index tags to each input token. We define the module vocabulary as $V^{(ef)} = \{0, 1, \ldots, m\}$ where 0 is the index assigned to the context entities that are not considered. The remaining values are indices that permute our entities based on the logical form. Here, m is the total number of indices based on the maximum number of entities from all logical forms. Overall, our filtering and permutation module is modelled using a feed-forward network with two linear layers separated with a Leaky ReLU activation function and appended with a softmax. Formally we define the module as:

$$\begin{aligned} h^{(ef)} &= LeakyReLU(\boldsymbol{W}^{(ef_1)}[h^{(enc)}; h^{(l)}]), \\ p_t^{(ef)} &= softmax(\boldsymbol{W}^{(ef_2)} h_t^{(ef)}), \end{aligned} \tag{4}$$

where $\boldsymbol{W}^{(ef_1)} \in \mathbb{R}^{d \times 2d}$ are the weights of the first linear layer and $h_t^{(ef)}$ is the hidden state of the module in time step t. $\boldsymbol{W}^{(ef_2)} \in \mathbb{R}^{|V^{(ef)}| \times d}$ are the weights of the second linear layer, $|V^{(ef)}|$ is the size of the vocabulary and $p_t^{(ef)}$ denotes the probability distribution over the tag indices for the time step t.

3.4 Graph Attention-Based Module

A knowledge graph (KG) can be denoted as a set of triples $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ where $\mathcal{E}$ and $\mathcal{R}$ are the set of entities and relations respectively. To build the (local) graph, we consider the relations and the types of entities that are linked with these relations in the knowledge graph $\mathcal{K}$. We define a graph $\mathcal{G} = \{\mathcal{T} \cup \mathcal{R}, \mathcal{L}\}$ where $\mathcal{T}$ is the set of types, $\mathcal{R}$ is the set of relations and $\mathcal{L}$ is a set of links (tp_1, r) and (r, tp_2) such that $\exists(e_1, r, e_2) \in \mathcal{K}$ where e_1 is of type tp_1 and e_2 is of type tp_2.

To propagate information in the graph and to project prior KG information into the embedding space, we use the Graph Attention Networks (GATs) (Veličković et al., 2018).

We initialise each node embedding $h^{(g)} = \{h_1^{(g)}, \ldots, h_n^{(g)}\}$ using pretrained BERT embeddings, and $n = |\mathcal{T} \cup \mathcal{R}|$. A GAT layer uses a parameter weight matrix, and self-attention, to produce a transformation of input representations $\overline{h}^{(g)} = \{\overline{h}_1^{(g)}, \ldots, \overline{h}_n^{(g)}\}$, where $\overline{h}_i^{(g)} \in \mathbb{R}^d$ as shown below:

$$\overline{h}^{(g)} = g(h^{(g)}; \theta^{(g)}),^{4} \tag{5}$$

and $\theta^{(g)}$ are the trainable parameters. We model the task of predicting the correct type or predicate in the logical form as a classification task over the nodes in graph $\mathcal{G}$, given the current conversational context and decoder hidden state. For each time step t in the decoder, we calculate the probability distribution $p_t^{(g)}$ over the graph nodes as:

$$p_t^{(g)} = softmax(\overline{h}^{(g)T} h_t^{(c)}), \tag{6}$$

where $\overline{h}^{(g)} \in \mathbb{R}^{d \times n}$ and $h_t^{(c)}$ is a linear projection of the concatenation of the context representation and the decoder hidden state, given as follows,

$$h_t^{(c)} = LeakyReLU(\boldsymbol{W}^{(g)}[h_{ctx}^{(enc)}; h_t^{(dec)}]), \tag{7}$$

and $\boldsymbol{W}^{(g)} \in \mathbb{R}^{d \times 2d}$.

3.5 Learning

The framework consists of four trainable modules, grammar guided decoder, entity detection, filtering and permutation, and the GAT-based module for types and predicates. Every module consists of a loss function that contributes to the overall performance of the framework, as shown in Section 4.3. To account for multi-tasking, we perform a weighted average of all the single losses:

$$L = \lambda_1 L^{dec} + \lambda_2 L^{ed} + \lambda_3 L^{ef} + \lambda_4 L^{g}, \tag{8}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are the relative weights, which are learned during training by taking into account the difference in magnitude between losses by incorporating the log standard deviation (Armitage et al., 2020; Cipolla et al., 2018). L^{dec}, L^{ed}, L^{ef}, and L^{g} are the respective negative log-likelihood losses of the grammar guided decoder, entity detection, filtering and permutation, and GAT-based modules. These losses are defined as follows:

$$\begin{aligned}
L^{dec} &= -\sum_{k=1}^{m} logp(y_k^{(dec)}|x), \\
L^{ed} &= -\sum_{j=1}^{n} logp(y_j^{(ed)}|x), \\
L^{ef} &= -\sum_{i=1}^{n} logp(y_i^{(ef)}|x), \\
L^{g} &= -\sum_{k=1}^{m} I_{(y_k^{(dec)} \in \{type, pred\})} logp(y_k^{(g)}|x),
\end{aligned} \tag{9}$$

where n and m are the length of the input utterance x and the gold logical form, respectively. $y_k^{(dec)} \in V^{(dec)}$ are the gold labels for the decoder, $y_j^{(ed)} \in V^{(ed)}$ are the gold labels for entity detection, $y_j^{(ef)} \in V^{(ef)}$ are the gold labels for filtering and permutation, and $y_k^{(g)} \in \{\mathcal{T} \cup \mathcal{R}\}$ are the gold labels for the GAT-based module. The model benefits from multiple supervision signals from each module, and this improves the performance in the given task.

4 Experiments

4.1 Experimental Setup

Datasets We use the Complex Sequential Question Answering (CSQA) dataset[5] (Saha et al., 2018). CSQA was built on the large-scale knowledge graph Wikidata. Wikidata consists of 21.2M triples with over 12.8M entities, 3,054 entity types, and 567 predicates. The CSQA dataset consists of around 200K dialogues where each partition – train, valid, test contains 153K, 16K, 28K dialogues, respectively. The questions involve complex reasoning to determine the correct answers.

Model Configurations We incorporate a semi-automated preprocessing step to annotate the CSQA dataset with gold logical forms. For each question type and subtype in the dataset, we create a general template with a pattern sequence that the actions should follow. Thereafter, we follow a set of rules to create the specific gold logical form that extracts the gold sequence of actions based on the type of question for each question. The actions used for this process are the ones in Table 1. For all the modules in the LASAGNE framework, we employ an embedding dimension of 300. We utilise the transformer model with six heads for the multi-head attention model with two layers. For the optimisation, we use the Noam optimiser proposed by (Vaswani et al., 2017), where authors use an Adam optimiser (Kingma and Ba, 2015) with several warmup steps for the learning rate. Please refer to the appendix submitted with the paper for more details.

Models for Comparison We compare the LASAGNE framework with the last three baselines that have been evaluated on the employed dataset. The first baseline is (Saha et al., 2018) where authors introduce the HRED+KVmem model. The second baseline is D2A (Guo et al., 2018), which

---

[4] For more details about GAT please refer to the appendix.

[5] `https://amritasaha1812.github.io/CSQA`

Methods		HRED-KVM	D2A	MaSP	LASAGNE (ours)	Δ
Question Type	**#Examples**			F1 Score		
Overall	206k	9.39%	66.70%	79.26%	**82.91%**	+3.65%
Clarification	12k	16.35%	35.53%	**80.79%**	69.46%	-11.33%
Comparative Reasoning (All)	15k	2.96%	48.85%	68.90%	**69.77%**	+0.87%
Logical Reasoning (All)	22k	8.33%	67.31%	69.04%	**89.83%**	+20.79%
Quantitative Reasoning (All)	9k	0.96%	56.41%	73.75%	**86.67%**	+12.92%
Simple Question (Coreferenced)	55k	7.26%	57.69%	76.47%	**79.06%**	+2.59%
Simple Question (Direct)	82k	13.64%	78.42%	85.18%	**87.95%**	+2.77%
Simple Question (Ellipsis)	10k	9.95%	81.14%	**83.73%**	80.09%	-3.64%
Question Type	**#Examples**			Accuracy		
Overall	66k	14.95%	37.33%	45.56%	**64.34%**	+18.78%
Verification (Boolean)	27k	21.04%	45.05%	60.63%	**78.86%**	+18.23%
Quantitative Reasoning (Count)	24k	12.13%	40.94%	43.39%	**55.18%**	+11.79%
Comparative Reasoning (Count)	15k	8.67%	17.78%	22.26%	**53.34%**	+31.08%

Table 2: LASAGNE's performance comparison on the CSQA dataset having 200K dialogues with 1.6M turns and over 12.8M entities. LASAGNE achieves "overall" (weighted average on all question types) new state of the art for both the F1 score and the question type results' accuracy metric.

uses a semantic parsing approach based on a seq2seq model. Finally, the current state of the art is MaSP (Shen et al., 2019), which is also a semantic parsing approach.

Evaluation Metrics We use the same metrics as employed by the authors of the CSQA dataset (Saha et al., 2018) as well as the previous baselines. The "F1-score" is used for questions that have an answer composed of a set of entities. The "Accuracy" metric is used for the question types whose answer is a number or a boolean value (YES/NO). We also provide an overall score for each evaluation metric and their corresponding question categories.

4.2 Results

Table 2 summarises the results comparing the LASAGNE framework against the previous baselines. LASAGNE outperforms the previous baselines weighted average on all question types (The row "overall" in the Table 2). Furthermore, LASAGNE is a new SotA in 8 out of 10 question types, and in some cases, the improvement is up to 31 percent.

What worked in our case? For question types that require more than two entities for reasoning, such as *Logical Reasoning (All)* and *Verification (Boolean)*, LASAGNE performs considerably better (+20.79% and +18.23% respectively). This is mainly due to the proposed entity recognition module. Furthermore, for question types that require two or more (entity) types and predicates, such as *Quantitative Reasoning (All)*, *Quantitative Reasoning (Count)* and *Comparative Reasoning (Count)* LASAGNE also outperforms MaSP (+12.92%, +11.79% and +31.08% respectively). Here, the improvement is due to the graph attention-based module, which is responsible for predicting the relevant (entity) types and predicates. Another interesting result is that LASAGNE also performs better in two out of three *Simple Question involving one entity and one predicate* categories. The performance shows the robustness of LASAGNE.

What did not work in our case? LASAGNE noticeably under-performs on the *Clarification* question type, where MaSP retains the state-of-the-art. The main reason is the spurious logical forms during the annotation process which has further impacted the Simple Questions (Ellipses) performance.

4.3 Ablation Study

Effect of GAT and Multi-task Learning Table 3 summarises the effectiveness of the GAT-based module and the multi-task learning. We can observe the advantage of using them together in LASAGNE. To show the effectiveness of GAT-based module, we replace it with two simple classifiers, one for each predicate and type categories. We can observe that the performance drops significantly for the question types that require multiple entity types and predicates (e.g. *Quantitative Reasoning (All)*, *Quantitative Reasoning (Count)* and *Comparative Reasoning (Count)*). When we exclude the multi-task learning and train all the modules independently, there is a negative impact on all question types. In LASAGNE, the filtering and

Methods	Ours	w/o GATs	w/o Multi
Question Type	F1 Score		
Clarification	66.94%	57.33%	59.43%
Comparative	69.77%	57.72%	66.41%
Logical	89.83%	78.52%	86.75%
Quantitative	86.67%	75.26%	82.18%
Simple (Coref)	79.06%	76.46%	77.23%
Simple (Direct)	87.95%	83.59%	85.39%
Simple (Ellipsis)	80.09%	77.19%	78.47%
Question Type	Accuracy		
Verification	78.86%	63.38%	75.24%
Quantitative	55.18%	40.87%	46.27%
Comparative	53.34%	41.73%	45.90%

Table 3: The effectiveness of the GAT and the multi-task learning. The first column contains the results of the LASAGNE framework, where all the modules are trained simultaneously. The second and third columns selectively remove the GAT and the multi-task learning from LASAGNE.

permutation module, along with the GAT-based module, is heavily dependent on the supervision signals received from the previous modules. Therefore it is expected that without the multi-task learning, LASAGNE will underperform on all question types, since each module has to re-learn inherited information.

4.4 Task Analysis

Tasks	Accuracy
Entity Detection	86.75%
Filtering & Permutation	97.49%
Grammar-Guided Decoder for Logical Forms	98.61%
GAT-Based Module for Type/Predicate	92.28%

Table 4: Tasks accuracy of the LASAGNE framework.

Table 4 illustrates the task accuracy of LASAGNE. The Entity Detection task has the lowest accuracy (86.75%). The main reason here is the errors in the entity type prediction. On the other hand, for all other tasks, we have accuracy above 90%.

Effect of Filtering and Permutation For justifying the effectiveness and superior performance of LASAGNE's filtering and permutation module, we compare the overall performance of the entity recognition module to the corresponding module from MaSP. Please note, entity detection modules in both frameworks adopt a similar approach as defined in section 3.3. In Table 5 we can see that the MaSP entity recognition module provides an overall accuracy of 79.8% on test data, while our module outperforms it with an accuracy of 92.1%. The main reason for the under-performance of MaSP is that it uses only token embeddings without any entity information. In contrast, our approach avoids re-learning entity information in the question context and relies on the entity detection module's information.

Model	Entity Recognition Accuracy
MaSP	79.8%
LASAGNE	92.1%

Table 5: Comparing MaSP (Shen et al., 2019) and LASAGNE for entity recognition performance.

4.5 Error Analysis

For the error analysis, we randomly sampled 100 incorrect predictions. We detail the reasons for two types of errors observed in the analysis:

Entity Ambiguity Even though our entity detection module assigns (entity) types to each predicted span, entity ambiguity remains the biggest challenge for our framework. For instance, for the question, "Who is associated with Jeff Smith ?" LASAGNE entity detection module correctly identifies "Jeff Smith" as an entity surface form and correctly assigns the (entity) type "common name". However, the Wikidata knowledge graph contains more than ten entities with exactly the same label and type. Our entity linking module has difficulties in such cases. Wikidata entity linking is a newly emerging research domain that has its specific challenges such as entities sharing the same labels, user-created non-standard entity labels and multi-word entity labels (up to 62 words) (Mulang et al., 2020b). Additional entity contexts, such as entity descriptions and other KG contexts, could help resolve the Wikidata entity ambiguity (Mulang et al., 2020a).

Spurious Logical Form For specific question categories, we could not identify gold actions for all utterances. Therefore spurious logical form is a standard error that affects LASAGNE. Specifically, we have spurious logical forms for categories such as "Comparative, Quantitative, and Clarification" but still can achieve SotA in the comparative and quantitative categories.

5 Conclusions

In this article, we focus on complex question answering over a large-scale knowledge graph containing conversational context. We provide a transformer-based framework to handle the task in a multi-task semantic parsing manner. At the same time, we propose a named entity recognition module for entity detection, filtering, and permutation. Furthermore, we also introduce a graph attention-based module, which exploits correlations between (entity) types and predicates for identifying the gold ones for each particular context. We empirically show that our model achieves the best results for numerous question types and also overall. Our ablation study demonstrates the effectiveness of the multi-task learning and of our graph-based module. We also present an error analysis on a random sample of "wrong examples" to discuss our model's weaknesses. For future work, we believe that reinforcement learning is a viable alternative to explore complex conversational question answering without gold annotations.

Acknowledgments

The project leading to this publication has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 812997 (Cleopatra).

References

Jason Armitage, Endri Kacupaj, Golsa Tahmasebzadeh, Maria Maleshkova, Ralph Ewerth, Jens Lehmann, et al. 2020. Mlm: A benchmark dataset for multitask learning with multiple languages and modalities. *arXiv preprint arXiv:2008.06376*.

Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 729–738.

R. Cipolla, Y. Gal, and A. Kendall. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7482–7491.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2018. Dialog-to-action: Conversational question answering over a large-scale knowledge base. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 2946–2955.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Diederik P Kingma and Jimmy Ba. 2015. Adam, a method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, volume 1412.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Chen Liang, Jonathan Berant, Quoc V. Le, Kenneth D. Forbus, and Ni Lao. 2017. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 23–33. Association for Computational Linguistics.

Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 783–792. ACL.

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1400–1409. The Association for Computational Linguistics.

Isaiah Onando Mulang, Kuldeep Singh, Chaitali Prabhu, Abhishek Nadgeri, Johannes Hoffart, and Jens Lehmann. 2020a. Evaluating the impact of knowledge graph context on entity disambiguation models. *29th ACM Conference on Information and Knowledge Management (CIKM)*.

Isaiah Onando Mulang, Kuldeep Singh, Akhilesh Vyas, Saeedeh Shekarpour, Ahmad Sakor, Maria Esther

Vidal, Soren Auer, and Jens Lehmann. 2020b. Encoding knowledge graph entity aliases in an attentive neural networks for wikidata entity linking. *In Proceedings of 21st Conference on Web Information System and Engineering*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 705–713. AAAI Press.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2442–2451, Hong Kong, China. Association for Computational Linguistics.

Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, et al. 2018. Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference*, pages 1247–1256.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv e-prints*, page arXiv:1609.08144.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 6069–6076. AAAI Press.

A Grammar

We propose a new grammar to annotate the dataset with a gold logical form to perform the semantic parsing task. We consider the work by (Guo et al., 2018) as a starting point for generating them. While we differ in many actions regarding their semantic and therefore their implementation. Our goal was to define more precise and richer actions, which gives us a more flexible grammar in terms of being used to annotate a wider range of question's complexities. For instance, for a couple of actions, we also define their reverse occurrence (e.g. *find*, *find_reverse*)). We do this in order to match the knowledge graph triple direction (*subject-predicate-object*). In some questions, we might have the subject or the object entity. Having both normal and reverse actions helps us to identify directly the correct answer based on the action the model predicted. Furthermore, we also define actions that do not exist in (Guo et al., 2018). Some of them are *find_tuple_counts*, *atmost*, *atleast*. Table 1 illustrates the complete grammar with all the defined actions. Following (Lu et al., 2008), we define each action with a function that can be executed on the knowledge graph. Finally, in order to execute a sequence of actions, we have to parse it into a tree structure. There our executor starts from the tree leaves and it recursively executes the leftmost non-terminal node until the whole tree is complete.

B Case Study

Table 9 shows examples from different question types in the CSQA dataset and the logical forms generated from our model. As we can see, our actions can cover reasoning for every question type by following certain patterns depending on them. The sequences can cover all the different complexities of the questions. For example, the logical forms pattern of *Simple Questions* to *Quantitative* or *Comparative* is slightly different due to increased complexity of the latter questions. The reasoning over *Quantitative* or *Comparative* question involves more actions in order to reach the correct answer.

For the question type *Simple Question (Direct)*, we can see the question “Which administrative territory is the birthplace of Antonio Reguero ?”. The correct logical form for this example is “filter_type(find(Antonio Reguero, place of birth), administrative territorial entity)”. Here we can distinguish two different actions; the first one is the *filter_type* and the other one is the *find* action. The *find* action receives as input an entity subject and a predicate and provides the set of object entities from the Knowledge Graph. Whereas, the *filter_type* action receives as input a set of entities along with an entity type and results to a set of entities that belong to that particular entity type.

C Hyperparamters and module configurations

Table 6 summarizes the hyperparameters used across the LASAGNE framework. For the transformer module, we use the configurations from (Vaswani et al., 2017). Our model dimension is $d_{model} = 300$, with a total number of $H = 6$ heads and $L = 2$ layers. The inner feed-forward linear layers have dimension $d_{ff} = 600$. Following the base transformer parameters, we apply residual dropout (Srivastava et al., 2014) to the summation of the embeddings and the positional encodings in both encoder and decoder stacks with a rate of 0.1. The entity detection module has a dimension of 300. Our base LSTM here is followed with a LeakyReLU, dropout, and a linear layer. The output of the linear layer is the module prediction while the LSTM hidden state is propagated to the filtering and permutation layer. The filtering and permutation module receives an input of dimension 600 where here a linear layer is responsible to reduce it to 300 which is the framework dimension. Like in the previous module, a LeakyReLU, dropout, and a linear layer are used for the final predictions. Finally, for the GAT-based module, we use pre-trained BERT embeddings for type and predicate labels. Hence the input dimension on this module is 3072. The GAT layer will produce representations with an embedding size of 300. Next, multiple linear, dropout, and LeakyReLU layers are used to produce the final predictions.

Hyperparameters	**Value**
epochs	20
batch size	64
dropout ratio	0.1
learning rate	0.001
warmup steps	4000
optimizer	Adam
β_1	0.9
β_2	0.999
ε	1e-09
model dimension	300
model pretrained embeddings	GloVe
non-linear activation	LeakyRelu
GAT input dimension	3072
GAT node dimension	300
GAT pretrained embeddings	BERT

Table 6: Hyper-parameters for LASAGNE framework.

D Graph Attention Networks

Figure 3 shows the aggregation process of graph attention layer between the (entity) types and predicates from Wikidata. The KB types and predicates are the nodes of the graph, and there exist an edge only between types and predicates with the condition that there exist a triple which involved the predicate and an entity of that type. We use GATs (Veličković et al., 2018) to capture dif-

Methods		HRED-KVM		D2A		MaSP		LASAGNE (ours)	
Question Type	**#Examples**	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Overall	206k	6.30%	18.40%	66.57%	66.83%	80.48%	78.07%	87.08%	80.31%
Clarification	12k	12.13%	25.09%	33.97%	37.24%	77.66%	84.18%	81.80%	60.35%
Comparative Reasoning (All)	15k	4.97%	2.11%	54.68%	44.14%	81.20%	59.83%	83.88%	59.73%
Logical Reasoning (All)	22k	5.75%	15.11%	68.86%	65.82%	78.00%	61.92%	98.67%	82.43%
Quantitative Reasoning (All)	9k	1.01%	0.91%	60.63%	52.74%	79.02%	69.14%	79.66%	95.02%
Simple Question (Coreferenced)	55k	5.09%	12.67%	56.94%	58.47%	76.01%	76.94%	72.71%	86.62%
Simple Question (Direct)	82k	8.58%	33.30%	77.37%	79.50%	84.29%	86.09%	94.94%	81.92%
Simple Question (Ellipsis)	10k	6.98%	17.30%	77.90%	84.67%	82.03%	85.50%	93.74%	69.91%
Question Type	**#Examples**	Accuracy							
Overall	66k	14.95%		37.33%		45.56%		64.34%	
Verification (Boolean)	27k	21.04%		45.05%		60.63%		78.86%	
Quantitative Reasoning (Count)	24k	12.13%		40.94%		43.39%		55.18%	
Comparative Reasoning (Count)	15k	8.67%		17.78%		22.26%		53.34%	

Table 7: Precision and recall comparison with baselines.

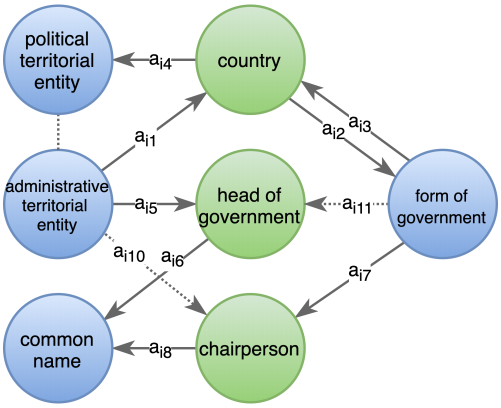


Figure 3: The aggregation process of graph attention layer between the (entity) types and predicates from Wikidata knowledge graph. The dashed lines represent an auxiliary edge, while a_{ij} represents relative attention values of the edge. We also incorporate the predicates (relations) as nodes of the graph instead of edges.

ferent level of information for a node, based on the neighborhood in the graph. We denote with $h^{(g)} = \{h_1^{(g)}, \dots, h_n^{(g)}\}$ the initial representations of the nodes, which will also be the input features for the GAT layer. To denote the influence of node j to the node i, an attention score e_{ij} is computed as $e_{ij} = a(\mathbf{W}h_i^{(g)}, \mathbf{W}h_j^{(g)})$, where $\mathbf{W}$ is a parameterized linear transformation, and a is an attention function. In our case, we follow the GAT paper, and compute e_{ij} score as follows,

$$e_{ij} = LeakyReLU(\mathbf{a}^T[\mathbf{W}h_i^{(g)} || \mathbf{W}h_j^{(g)}]), \quad (10)$$

where $\mathbf{a} \in \mathbb{R}^{2d}$ is a single-layer feedforward network, and $||$ denotes concatenation. This attention scores are normalized using a softmax function and producing the α_{ij} scores for all the edges in a neighborhood. These normalized attention scores are used to compute the output features $\overline{h}_i^{(g)}$ of a node in a graph, by applying a linear combination of all the nodes in the neighborhood as below,

$$\overline{h}_i^{(g)} = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} h_j^{(g)}) \quad (11)$$

where σ is a non-linear function. Following (Veličković et al., 2018) and (Vaswani et al., 2017) we also apply a multi-head attention mechanism and compute the final output features as,

$$\overline{h}_i^{(g)} = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k h_j^{(g)}) \quad (12)$$

where K is equal to the number of heads, and α_{ij}^k, $\mathbf{W}^k$ are the corresponding attention scores and linear transformation by the k-th attention mechanism. During our experiments, we found out the $K = 2$ was sufficient for our model.

E Experiments

Table 7 summarizes precision and recall results comparing LASAGNE framework against the previous baselines. Furthermore, a detailed task analysis for each task on each question type is illustrated on Table 8.

Tasks	Entity Detection	Filt. & Permut.	Logical Form	Type/Predicate
Question Type	Accuracy			
Clarification	92.19%	99.97%	98.36%	86.70%
Comparative Reasoning (All)	92.03%	99.88%	99.00%	97.18%
Logical Reasoning (All)	72.20%	99.44%	98.18%	95.95%
Quantitative Reasoning (All)	87.38%	100.0%	99.56%	95.87%
Simple Question (Coreferenced)	93.50%	96.92%	98.50%	90.12%
Simple Question (Direct)	90.58%	99.34%	98.58%	89.71%
Simple Question (Ellipsis)	77.90%	99.98%	98.81%	90.02%
Verification (Boolean)	79.50%	84.66%	99.79%	98.10%
Quantitative Reasoning (Count)	77.77%	99.80%	97.34%	92.09%
Comparative Reasoning (Count)	92.04%	99.98%	98.66%	96.92%

Table 8: Task accuracy from LASAGNE. We can obtain that entity detection is the task with lowest accuracy while filtering and permutation together with logical form generation are the tasks with highest accuracy.

Question Type	Question	Logical Forms
Simple (Direct)	Q1: Which administrative territory is the birthplace of Antonio Reguero ?	filter_type(find(Antonio Reguero, place of birth), administrative territorial entity)
Simple (Ellipsis)	Q1: Which administrative territories are twin towns of Madrid ? A1: Prague, Moscow, Budapest Q2: And what about Urban Community of Brest?	filter_type(find(Urban Community of Brest, twinned administrative body), administrative territorial entity)
Simple (Coref)	Q1: What was the sport that Marie Pyko was a part of ? A1: Association football Q2: Which political territory does that person belong to ?	filter_type(find(Marie Pyko, country of citizenship), political territorial entity)
Quantitative Reasoning (Count)	Q1: How many beauty contests and business enterprises are located at that city ? A1: Did you mean Caracas? Q2: Yes	count(union(filter_type(find_reverse(Caracas, located in), beauty contest), filter_type(find_reverse(Caracas, located in), business enterprises)))
Quantitative Reasoning (All)	Q1; Which political territories are known to have diplomatic connections with max number of political territories ?	argmax(find_tuple_counts(diplomatic relation, political territorial entity, political territorial entity))
Comparative Reasoning (Count)	Q1: How many alphabets are used as the scripts for more number of languages than Jawi alphabet ?	count(greater(count(filter_type(find(Jawi alphabet, writing system), language)), find_tuple_counts(writing system, alphabet, language)))
Comparative Reasoning (All)	Q1: Which occupations were more number of publications and works mainly about than composer ?	greater(union(find_reverse_tuple_counts(main subject, occupation, publication), find_reverse_tuple_counts(main subject, occupation, work)), count(filter_multi_types(find_reverse(composer, main subject), publication, work)))
Verification	Q1: Was Geir Rasmussen born at that administrative territory ?	is_in(find(Geir Rasmussen, place of birth), Chicago)

Table 9: Examples from the CSQA dataset (Saha et al., 2018) annotated with gold logical form.

DRAG: Director-Generator Language Modelling Framework for Non-Parallel Author Stylized Rewriting

Hrituraj Singh
Adobe Research
`hrisingh@adobe.com`

Gaurav Verma[*]
Georgia Tech
`gverma@gatech.edu`

Aparna Garimella
Adobe Research
`garimell@adobe.com`

Balaji Vasan Srinivasan
Adobe Research
`balsrini@adobe.com`

Abstract

Author stylized rewriting is the task of rewriting an input text in a particular author's style. Recent works in this area have leveraged Transformer-based language models in a denoising autoencoder setup to generate author stylized text without relying on a parallel corpus of data. However, these approaches are limited by the lack of explicit control of target attributes and being entirely data-driven. In this paper, we propose a Director-Generator framework to rewrite content in the target author's style, specifically focusing on certain target attributes. We show that our proposed framework works well even with a limited-sized target author corpus. Our experiments on corpora consisting of relatively small-sized text authored by three distinct authors show significant improvements upon existing works to rewrite input texts in target author's style. Our quantitative and qualitative analyses further show that our model has better meaning retention and results in more fluent generations.

1 Introduction

With recent advances in language modeling techniques that have resulted in powerful language models (Radford et al., 2019; Devlin et al., 2018; Brown et al., 2020) along with an increased interest in stylized text generation (Hu et al., 2017; Shen et al., 2017; Subramanian et al., 2018; Fu et al., 2018; Niu and Bansal, 2018), large language models have been successfully tuned to achieve text stylization (Lample et al., 2018; Ziegler et al., 2019; Syed et al., 2020; Singh et al., 2020). Apart from *transferring* an input text to the target style, which has received recent interest from the community, *understanding* and *measuring* style have been persistently explored over the last few decades (Kessler et al., 1997; Garera and Yarowsky, 2009; Liu, 2012; Verma and Srinivasan, 2019). Lying at the intersection of style transfer enabled by advanced language models and a deep understanding of style as a nuanced combination of several linguistic concepts, problems like *stylized generation* or *stylized rewriting* have gained further traction. A large body of work in style transfer focuses on binary aspects such as positive-negative sentiment (Li et al., 2018; Ziegler et al., 2019), formal-informal (Jain et al., 2019), and sometimes a mixture of these attributes (Subramanian et al., 2018). To fuel this interest in such binary stylization, some datasets comprising of text from the extreme ends of these spectrums have also emerged (e.g., positive-negative sentiments (Mathews et al., 2016), formal-informal (Rao and Tetreault, 2018)). As pointed by Syed et al. (2020), author stylized rewriting does not directly fit under any of these variants as the writing style of an author is an amalgamation of several such attributes and needs to be modeled in a fine-grained manner.

Apart from the distinction along style dimensions, prior works can also be categorized as supervised (using parallel corpus (Jhamtani et al., 2017)) and unsupervised (Li et al., 2018; Syed et al., 2020; Niu and Bansal, 2018). In supervised frameworks, parallel data is used to tune sequence-to-sequence models for stylized rewriting. However, annotating such parallel corpus is a tedious effort and therefore, there is an increased interest in unsupervised style transfer; i.e., when there is no direct supervision or parallel data available for training the models. In this work, we focus on such an unsupervised setting.

Existing approaches on unsupervised author stylized rewriting rely on implicitly learning the target stylistic attributes from data and do not allow finer control on generation (Syed et al., 2020). While this is a good starting point for author-stylized

---

[*] This was work was carried out when the author was at Adobe Research.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 863–873
April 19 - 23, 2021.

rewriting, it is desirable to further improve the rewriting model on certain aspects without compromising on other attributes that the model has already optimized. An example would be to retain the stylistic strengths while improving content retention, or vice versa. To this end, we propose **Dir**ecting **a** **G**enerator framework (DRAG). Our quantitive and qualitative experiments show the viability of the proposed approach. Experiments further indicate that the framework's setup allows it to operate efficiently in scarce data setting and improves the performance over the baseline models. Our contributions can be summarized as - **(1)** We introduce a director-generator approach to *rewrite* an input text in a target author's style. **(2)** We propose linguistic alignment scores – both at the local and global level and extend these to design *thresholds* for the generator and director. **(3)** We present experimental results on texts written by three authors from the Gutenberg corpus with very distinct writing styles, and show that our approach outperforms prior works across content retention and style alignment metrics. **(4)** We further identify and discuss shortcomings of our proposed approach, and present error analysis to aid future research in author stylized rewriting.

2 Related Work

With the rise of Transformer-based (Vaswani et al., 2017) **language models**, generative pretraining (Devlin et al., 2018; Radford et al., 2019; Brown et al., 2020) has advanced the field of NLP significantly. Fine-tuning such large language models on specific task has become very prevalent (Sun et al., 2019; Lee et al., 2020; Lample and Conneau, 2019; Raffel et al., 2019; Liu et al., 2019). Pretraining infuses the *generic* language knowledge into the language model helping it understand the specific tasks with relatively much less supervision. In fact, recent approaches (Radford et al., 2019; Brown et al., 2020) show that often, even such small supervisions are not required and a simple instruction can be used to solve specific tasks by utilizing the capabilities of such large language models trained on very large datasets.

Pretraining of such models usually involves optimizing them on **Masked Language Modelling (MLM)** (Devlin et al., 2018), **Causal Language Modelling (CLM)** (Radford et al., 2019) or other similar (Clark et al., 2020) objectives.While CLM is the task of auto-regressively predicting the next word given the previous words or context, MLM is the task of recovering masked tokens from a given input. While these approaches mostly train only an encoder or a decoder framework, Lample and Conneau (2019) explored initializing encoder-decoder frameworks using the pre-trained encoders for cross-lingual translation. Such a technique with appropriate modification has been shown to be successful in incorporating stylistic aspects of the language as well (Conneau and Lample, 2019; Syed et al., 2020). All these works utilize the task of minimizing the *denoising auto-encoder loss* for inducing style in the language models in a reconstruction framework. For our explorations, we leverage these works to initialize our DRAG framework.

There is an increased interest in **stylistic generation** or text rewriting. Most of the approaches define dimensions like formality-informality (Shen et al., 2017; Ficler and Goldberg, 2017; Jain et al., 2019; Sun et al., 2019) and achieve the alignment along these dimensions. While some of these approaches rely on parallel corpus (Ficler and Goldberg, 2017; Jhamtani et al., 2017), many of the approaches focus on unsupervised framework (Li et al., 2018; Shen et al., 2017; Jain et al., 2019), where the model *preserves* the input content in the output while *biasing* the generations towards the target style. While some approaches utilize simple *editing* to achieve the style along particular dimensions (Li et al., 2018), others focus on achieving this through discriminators (Fu et al., 2018) or scorers (Jain et al., 2019). As mentioned before, since author style is an amalgamation of several such attributes, it requires much more than a discriminator or singular dimension tuning to achieve stylization.

Due to the difficulty associated with author style understanding and fine-grained nature of that style even if understood them, the problem of **author stylized rewriting** has not been explored a lot. While Jhamtani et al. (2017) try to solve this problem for a specific author (i.e. Shakespeare), their approach is contingent on the availability of a parallel corpus. Since preparing parallel corpus is a tedious and intractable process, especially while dealing with multiple authors and multiple combinations of input and output styles, it is essential to focus on unsupervised solutions. Most recently, Syed et al. (2020) leverage the capabilities of the large language models to solve this problem in an unsupervised manner.

3 Author Style

There has been significant work on understanding binary stylization along dimensions like formal-informal, positive-negative sentiment (Rao and Tetreault, 2018; Kessler et al., 1997; Pavlick and Tetreault, 2016; Collins-Thompson and Callan, 2005; Hovy, 1990; Inkpen and Hirst, 2006; Kantrowitz, 2003), however, there is limited work on understanding an author's writing style (McCarthy et al., 2006; Forgeard, 2008; Verma and Srinivasan, 2019). While style can be a mixture of several factors including, but not limited to, lexical preferences, syntactic/sentential choices, discourse structure, narrative style, tone, we follow Syed et al. (2020) and consider an author's style at three levels:

Surface style is estimated using the frequencies of different surface elements such as the number of commas, semicolons, colons, question marks, exclamation marks, and hyphens per paragraph, from a given author's text. We, thus, quantify the surface-style elements into a 6-dimensional vector.

Lexical style of an author is reflected in the author's choice of words. To describe the same concept, different authors may use different words. For instance, Rudyard Kipling, known for his classics in children's literature, tended to use more concrete words (e.g., gongs, rockets, torch) while Abraham Lincoln, being a political writer, used more abstract words (e.g., freedom, patriotism). We enumerate lexical style categories as subjective, objective, literary, colloquial, abstract and concrete (Brooke and Hirst, 2013). We use lexicons for each of these categories (Brooke and Hirst, 2013), and define *lexical style alignment* of each word in the vocabulary to a given style category as the average and normalized point-wise mutual information (PMI) between that word and the seed words in the lexicon for that style category. The lexical style alignment for each word is thus a 6-dimensional vector. We use the EmoBank corpus (Buechel and Hahn, 2017) to compute the co-occurrence statistics for PMI computations. The inclination of a word towards a style category is positive if its normalized PMI score is positive with respect to the given category. The inclination of an author towards a style category is then estimated by the fraction of words in their text that have a positive inclination towards the category.

Syntactic style of an author is indicated by the nature of sentences used and we estimate the distribution of different types of sentences in an author's text. Sentence types may range from complex, as seen in philosophical writings, to simple, as observed in children's storybooks. We use five categories of sentence styles: (i) simple, (ii) compound, (iii) complex, (iv) complex-compound sentences, and (v) others (Feng et al., 2012; Verma and Srinivasan, 2019; Syed et al., 2020). Sentences are categorized into one of these types using the algorithm proposed by Feng et al. (2012). The resulting 5-dimensional probability distribution vector is used as the estimation of syntactic style. These vectors are estimated at corpus-level, unlike those for lexical and surface style which are computed at paragraph-level.

4 DRAG: *Di*recting *a* *G*enerator for Stylized Rewriting

Our proposed framework, DRAG, that aims to rewrite a given piece of text with a specific target author's style consists of three main stages:

(1) Pretraining a language model to infuse general linguistic knowledge into the model

(2) Adapting the pre-trained language model towards the target author's writing style by further pretraining it on text written by this author (Syed et al., 2020), and

(3) Using a director-generator framework (as discussed later) to fine-tune such *biased* language model to improve its style transfer capabilities even further while fixing content preservation issues. It is worth noting that we do not rely on the availability of parallel data for any of our experiments.

4.1 Pretraining Language Model

In order to infuse general linguistic knowledge into a language model, we leverage Tranformer-based pretrained language models (Devlin et al., 2018; Radford et al., 2019; Brown et al., 2020) due to their recent success in text processing tasks (Vaswani et al., 2017; Devlin et al., 2018; Brown et al., 2020). Similar to Conneau and Lample (2019), we first train a Transformer-based encoder on the Masked Language Modelling (MLM) task with 15% of the tokens masked (Devlin et al., 2018) on a *generic* text corpus. We initialize an encoder-decoder framework, as shown in Figure 1, with this language model.

4.2 Adapting LM for Rewriting

To adapt the pretrained LM for author stylized rewriting, Syed et al. (2020) initialize an encoder-

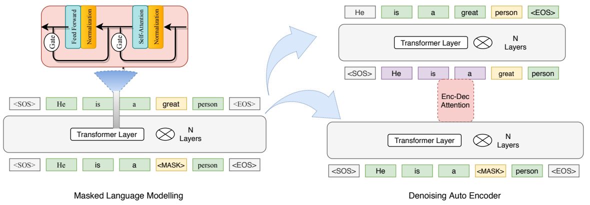


Figure 1: Language Model Pretraining using Masked Language Modelling followed by encoder-decoder initialization using pretrained models. This process still leaves the encoder-decoder attention parameters unitialized which can be initilized using the Denoising Auto Encoder training as depicted in the figure.

decoder framework with the pretrained LM, as shown in Figure 1. This is followed by optimizing it on denoising auto-encoder (DAE) loss (Lample et al., 2018; Lample and Conneau, 2019) *only* over target author's corpus. Syed et al. (2020) use the DAE loss to infuse an author's linguistic style into the reconstruction model; we refer to this framework as StyleLM. The fine-tuning using the DAE loss on a target author's corpus encourages recovering actual paragraphs from their noisy version (Lample and Conneau, 2019). For a paragraph g in corpus G and its noisy version $C(g)$ ($C(.)$ being the noise function), DAE loss is given by,

$$DAE(\theta_e, \theta_{ed}, \theta_d) = -\frac{1}{|G|} * \sum_{g \sim G} \log P(g/C(g); \theta_e, \theta_{ed}, \theta_d) \tag{1}$$

where P is the probability of reconstruction for a given encoder parameters θ_e, decoder parameters θ_d, and encoder-decoder attention parameters θ_{ed}. Please note that θ_{ed} does not refer to any additional layer but the parameters which are present in transformers and are responsible for encoder-decoder attention. In our setup, $C(.)$ function introduces two noises: (a) random dropping of words with 10% probability, and (b) word masking by replacing it with [MASK] token with 10% probability.

Given a noisy input, the encoder fills the [MASK] tokens with suitable replacements (based on the knowledge from its MLM pretraining), thus creating a pseudo *generic* input for the decoder, the target sequence for which is aligned to the target author's style. However, we identify and verify experimentally two issues with this approach:

(1) It requires a large target author corpora to achieve meaningful content preservation capability. This is evident by its very low content preservation scores (as discussed in Section 5.2) when trained on authors with relatively smaller corpora. Even with large corpora, the model still suffers from exposure bias to texts written only by the target author leading to spurious outputs for *unseen* inputs.

(2) The masking results in a significant emphasis on lexical style aspects, with a lesser focus on the surface and syntactic preferences. Since the model is completely data-driven, there is no way to explicitly add emphasis on additional style aspects.

One of the primary reasons behind (1) is the lack of explicit initialization of encoder-decoder attention parameters in StyleLM resulting in a random initialization. The model, therefore, needs a large corpus of author data to stabilize these parameters. To fix this, we propose to train the entire encoder-decoder language model using the DAE loss over the same *generic* corpus used for pre-training. The resulting model will be in the *generic* language space (English, in our case), and henceforth referred to as VanillaLM. We, further, finetune VanillaLM in the author corpus on the DAE loss to arrive at an improved version of StyleLM which we call iStyleLM. This offers better encoder-decoder attention initialization, and also removes the exposure bias of StyleLM, thus resulting in a more resilient and stable model with improved content preservation abilities (as demonstrated in Section 5.2).

However, at this point, we note that iStyleLM still fails to address (2), and its content preservation ability is also sub-optimal as the target author's style aspects which are infused at the later stage of training override some of the general linguistic knowledge. To further improve on iStyleLM, we introduce a Director-Generator component to our training framework in the next section.

4.3 Director-Generator Finetuning

For the Director-Generator finetuning, we find inspiration in the standard RL strategies (Rennie et al., 2017; Ranzato et al., 2015) where the *nearby* space is *explored* and certain actions are rewarded higher than others, consequently getting encouraged in the future. We, however, find direct rewarding unstable for our problem. Hence, we *generate* potential *directives* during exploration and *accept* or *reject* them on the basis of thresholds. A *directive*, in our context, is as an output paragraph generated from an input by a director model which is fixed and has been initialized using iStyleLM. Specifically, we create two copies of the iStyleLM as the *Director* and the *Generator*.

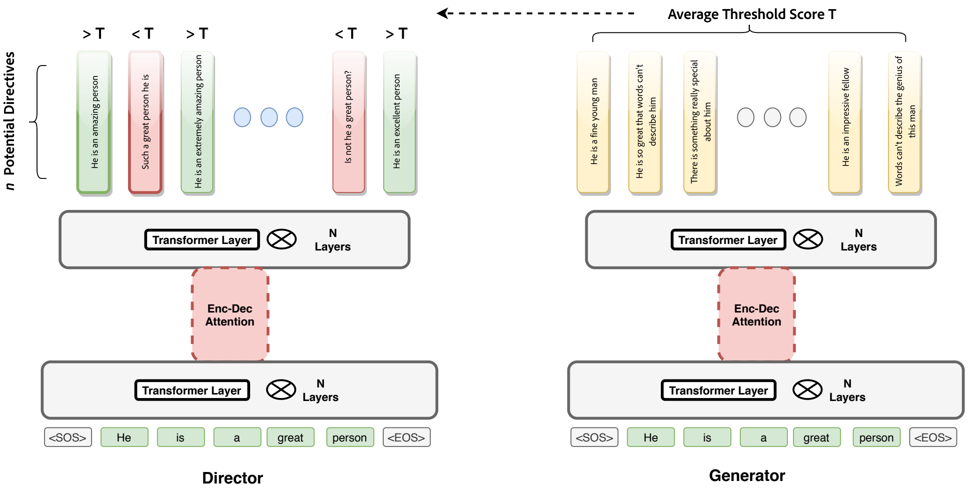


Figure 2: Both the director as well as generator, intiliazed using ISTYLELM, work together to improve the final outputs. While director remains in the space of author style generating and exploring potential directives , generator keeps changing its threshold as it gets improved on its content & style capabilities. The directives above the average threshold for same example are accepted while rest of them are rejected.

As the names indicate, for each input, the director proposes n potential directives or paragraphs, while the generator generates n *thresholding* outputs (paragraphs) as shown in Figure 2. We generate the potential directives using nucleus sampling (Holtzman et al., 2019) with a softmax temperature of 1.2, while the thresholding outputs are generated using a softmax temperature of 0.8 (the same value is used at inference time as well). We score the director and generator outputs on various content and style attributes. For content preservation, we use the BLEU score between input and output as the content score. For lexical style, the mean squared error is calculated between the 6-dimensional lexical alignment vector of the directives/generator outputs (calculated as the averaged sum of alignments of words in the proposal) and average lexical alignments of paragraphs for the target author corpus. Similarly, the mean squared error for surface style is also calculated. The scores L and S for lexical and surface styles, respectively, are then calculated as reciprocal of means squared errors (with ϵ added in the denominator to avoid zero-division). For syntactical choices, since we wish to achieve the probability distribution of different types of sentences at the corpus-level, we calculate the score for syntactic style as, $SX = \frac{sum(P_p \circ P_t)}{sum(P_p)}$ where P_p denotes the frequency distribution of different types of sentences in a directive/generator's output, P_t the probability distribution of different types of sentences in target author corpus, and $\circ$ denotes the Hadamard product. All three scores are summed to calculate the style score for the directives (and the generator outputs).

The ISTYLELM model already captures certain stylistic aspects of the target author. We want our model to leverage this understanding and improve on aspects where ISTYLELM does not perform well. To capture this, we compute the content and style scores of all the potential directives and generator outputs and retain *only* those directives which have **both** the content and style scores better than the average of the generators' outputs' scores. The *accepted* directives become *real* directives for the generator and are used to train it using the *teacher-forcing* cross-entropy loss. Note again that the director remains frozen with ISTYLELM during the entire training process. In the case of multiple potential directives being better than the generators' outputs' average, the cross-entropy loss for each directive is weighted by its marginal difference from the generator's average score on the style dimension; i.e., if the style score for a directive is D_s and average outputs' style score from the generator is G_s, its weight during the cross-entropy training is $D_s - G_s$. This objective is similar to the one used in SCST (Rennie et al., 2017) but only *accepted* directives are encouraged and nothing is explicitly discouraged.

In order to stabilize the Director-Generator fine-tuning framework, we use (a) **fixed director**, and (b) **moving generator**. Contrary to the natural expectation of exploring better directives with the training of the director, the fixed or frozen director prevents catastrophic degradation in case the training biases the model towards specific choices that further train the model. It is a known phenomenon in RL frameworks that the model quickly learns to bias towards specific choices that are more rewarding. Specifically, we observe that training the director as well leads to overfitting to the limited stylistic choices, thus resulting in the exploration of sub-optimal potential directives that seldom cross the required thresholds, especially the content preservation ones. With a moving (i.e. trained at each step) generator, its outputs scores account for the *current* state of the model against a fixed stable director, and hence only those directives get accepted which are better than the *current* capabilities (thresholds) of the generator. With a fixed generator, directives that would have been worse than *current* capabilities of the model but better than the capabilities of the fixed generator would also get accepted, thus training the model in the opposite direction. The Director-Generator finetuned iSTYLELM yields our proposed DRAG framework. At the inference time, we drop the director and use the Generator as our final rewriting model.

5 Experiments

We use a transformer encoder[1] with 512 hidden units, 16 heads, a dropout rate of 0.1, and learned positional embeddings during our MLM training. The model is trained using Adam Optimizer with a learning rate of 10^{-4}. The batch size used is 32 with a stream of 256 tokens, and the whole setup is trained until the validation performance (perplexity scores) shows no further improvement. The Transformers used in encoder-decoder setup also have the same parameters, and are initialized using the above encoder before training on further objectives. During DAE loss training, we use the same hyper-parameters used in (Conneau and Lample, 2019; Syed et al., 2020), and set p_{drop} and p_{blank} to 0.1. During director-generator training, we use n as 8 and ϵ as 0.05. The learning rate used in this case is 10^{-5}. In all the models, we use Byte Pair Encoding (Sennrich et al., 2015) with $80k$ codes learnt over the entire *generic* corpus.

5.1 Dataset

We use the 2,857 books written by 142 authors in the Gutenberg corpus (Lahiri, 2014), as used in (Syed et al., 2020), along with the Wikipedia articles, to form a corpus of about 4.6M passages. We refer to this corpus as *generic* during all our experiments, since it infuses only *generic* linguistic knowledge into the models. While MLM and VANILLALM are trained on the *generic* corpus, we select three authors with the most distinct writing styles, namely Albert Einstein, Michael Faraday, and John Stuart Mill, as measured by comparing their lexical alignments with the average lexical alignment of the Gutenberg corpus, as the target authors for author-specific style rewriting. Note that the choice of the authors is made purely on statistical basis with these three authors having maximum lexical style difference on their style vectors as described earlier when compared with the lexical style of entire *generic* corpus. For evaluation, we use the Opinosis corpus (Ganesan et al., 2010) as well as mixed author Gutenberg subset (with five passages from all the authors except the target author), which we refer to as *Generic (Test)*.

5.2 Quantitative Evaluation

Table 1 shows the results averaged over the three selected authors. The experiments are conducted on Opinosis and *Generic (Test)* datasets, using the following four models.

- **VANILLALM** is initialized using MLM-trained encoders and decoders and fine-tuned on the generic corpus using DAE loss.
- **STYLELM**, proposed by Syed et al. (2020),[2] is also initialized using MLM-trained encoders and decoders, but fine-tuned *only* on the target author corpus (instead of the generic corpus).
- **iSTYLELM**, an improved and stronger baseline compared to STYLELM, is initialized with VANILLALM and then fine-tuned on the target author corpus.
- **DRAG** is our proposed model. We use iSTYLELM to initialize both director and generator as described above, and then fine-tune them using inputs from *generic* corpus.

While the *Generic(Test)* corpus is predominantly literary due to the nature of the source, Opinosis covers everyday language. As shown in Table 1, STYLELM improves on the style alignment scores,

---

[1] As proposed by Parisotto et al. and shown in Fig. 1

[2] Note that the STYLELM code is not publicly available. Results shown in the table are using our own implementation.

Dataset	**Model**	**Content Preservation (↑)**				**Author Style (↓)**		
		ROUGE-1	**ROUGE-2**	**ROUGE-L**	**BLEU**	**Lexical(RMSE)**	**Surface(RMSE)**	**Syntactic(JSD)**
Opinosis	Vanilla LM	75.23	56.12	74.28	59.46	0.232	2.74	0.132
	StyleLM (Syed et al., 2020)	32.61	14.51	31.69	14.82	**0.174**	2.56	0.121
	iStyleLM	49.28	29.16	49.86	28.16	0.178	2.61	0.122
	DRAG	**57.23**	**36.12**	**56.98**	**37.53**	0.179	**2.48**	**0.109**
Generic	Vanilla LM	72.34	54.65	71.93	56.46	0.218	2.48	0.120
	StyleLM (Syed et al., 2020)	30.31	11.58	29.77	12.91	**0.163**	2.29	0.114
	iStyleLM	45.12	26.08	44.36	24.16	0.171	2.33	0.121
	DRAG	**52.39**	**30.66**	**51.98**	**33.28**	0.174	**2.21**	**0.097**

Table 1: ↑ indicates higher scores are better while ↓ indicates the opposite. Apart from lexical alignment where StyleLM performs marginally better, DRAG outperforms prior approaches. Vanilla LM performs best at content preservation but lacks any stylization

Input	**Albert Einstein**	**Michael Faraday**	**John Stuart Mill**
The accuracy at this point is very good	The experimental definitions developed is very *clearly*	The point is very wonderful	The physical for this **,** is very very pretty .
The estimated time to arrival does not seem to calculate the travelling time accurately	The estimated time relative to the leading existence does not seem likely to calculate the travelling time *exactly*	The discovery of ascertaining time **;** indeed **,** do not not show accuracy to the time to *angles*	The total time is to infer that arrival is not verified but often clearly a **,** accurately .

Table 2: Qualitative Outputs For Three different authors for same inputs

Input	**StyleLM (Syed et al., 2020)**	**DRAG (Ours)**
but after that it is very **easy** and quite **accurate** to use.	But for all that it is very After question about this and quite *measured* with consideration.	But after all it is very accurate and quite illustrious to the use of *events*.
Leather seats are very comfortable.	come on very very should we have any replaced.	This moving *hypothetical* seats are very comfortable.
I am not real fond of the electric seat and I find it is not as comfortable as my F150 pickup on trips	I am not real and use of the electric *position* and I find that it is not as well may's for the very hardly small have we led train	I am not real fond of the electric seat **,** and I find it is not as comfortable as *my physical relative on railway investigations.* .

Table 3: Comparison between STYLELM and DRAG for Albert Einstein

but at a great cost of content preservation when the target author corpus is small. This is possibly due to the random initialization of encoder-decoder attention parameters in the DAE training over target corpus, as reflected in the superior performance of ISTYLELM. We also note that while the approach proposed in STYLELM (Syed et al., 2020) improves lexical scores significantly, it fails to bring the same level of improvement in surface and syntactic alignments, perhaps due to the due to rare chances of less frequent punctuation symbols getting masked during DAE training, even more so when the target author corpus is not large enough to cover all possible masks. Similar reasoning explains the syntactic alignment issues, The DRAG approach, however, improves on both surface and syntactic alignment along with content preservation scores even though it comes at the marginal cost of lexical alignment. Please note that the purpose of Vanilla LM is to provide an estimation of *upper limit* on the content preservation scores and is not to be treated as a baseline due to the simple objective of its task (just copying the input tokens).

5.3 Qualitative Comparisons

We also qualitatively show some comparisons for different authors and different models. In Table 2, we show the outputs of DRAG for same input and different target authors. Evidently, our model produces changes both at the lexical as well as surface levels. The word 'good' in the first input is replaced by words like 'clearly', 'wonderful', and 'pretty', depending on the author. Some words do not replace any word but still get added to change the syntactical structure of the sentences. For example, appearance of the word 'relative' starts comparison to the 'leading existence' making it a bit complex. Sometimes, surface level changes like appearance of ';' also change the complexity of sentences.

We also show the comparison between STYLELM and our proposed DRAG outputs for same inputs when the target author is Albert Einstein as shown in Table 3. Evidently, while both models try to achieve the stylistic alignment, STYLELM ends up distorting the input sentence too much resulting in poor content preservation properties. Words like 'measured', 'hypothetical', and 'physical' relative reflect the *objective* approach used in Albert Einstein's writings.

6 Discussions and Limitations

While the language generation advancements are happening at a very high pace, the notion of style and the ability of models to rewrite same content in different styles is still far from being solved. One of the most important observation as made by Lample et al. (2018) is that it is very difficult to separate content from style. In fact, previous approaches which worked on the principle of *disentangling* style from content were not found to disentangle the style so much after all (Lample et al., 2018). The notion of style is still very far from being defined and concretized. While some psycholinguistic concepts can be defined to some extent (formality, sentiment, etc.), defining it at the level of author's style is very difficult due to manifestation of style at different levels as enumerated by Verma and Srinivasan (2019). Despite, such enumeration at various levels, it is far from *exhaustive* and therefore our approach still requires more granular understanding of style to closely emulate target author's style.

Our evaluation uses automatic metrics for style due to the difficulty associated with conducting human evaluation in author attribution tasks (Syed et al., 2020). The skill needed to identify the author's style is very intense thus making the human evaluation very costly. A more granular and detailed study on understanding how humans interpret an author's style is required to design a proper feedback mechanism. This is, however, outside the scope of this work.

6.1 What Did Not Work

In this section, we discuss some of our explorations that did not work as expected to aid future research in author stylization. We experimented with various **reinforcement learning setups** as it was a more natural choice once we had scoring engines for rewards. Using the VANILLALM as a policy and we explored Self Critical Sequence Training (SCST) (Rennie et al., 2017; Ranzato et al., 2015) and Proximal Policy Optimization (Schulman et al., 2017). However, all the setups were unstable in various ways for our problem. Note that our experiments and observations here are limited to the problem of author stylized rewriting only. SCST or self-critical sequence training is aimed at bringing the advantages of reinforcement learning setups for sequence level problems. A model (or policy) generates/explores outputs (or episodes) using multinomial sampling and greedy sampling. If the greedily sampled episode reward is r_b and the non-greedily sampled episode reward is r - the whole setup is trained using REINFORCE (Sutton and Barto, 2018) with r as the actual reward and r_b as baseline reward. We found this to limit exploration considering our problem is relatively much harder than previous metrics on which SCST has been successful due to our target metric being of an exact value. It, therefore, resulted in no improvement in either style or content scores. We, therefore, shifted to its modified version to encourage exploration, where we generated multiple episodes for each input and averaged their scores to use that as baseline reward r_b and trained the setup on all generated episodes using REINFORCE (Sutton and Barto, 2018). We found this approach to be effective at style incorporation but not generalizable at all. The model learned to repeat certain patterns with poor content preservation abilities. We, tried, to balance it with occasional denoising autoencoder loss training but that only *delayed* the overfitting and not solve it. We also attempted Proximal Policy Optimization in a setup same as (Sun et al., 2019) but it resulted in even worse outputs due to the critic's failure to approximate complex value functions for our objectives.

As discussed already, we only accept those directives which have scores above the threshold. We also tried a variant of it which had even those directives which do not score above the threshold. We scored them negatively thereby resulting in a bit similar framework like SCST but within some steps, we found more negative scores than positive due to bad content preservation pushing the model away from a bad state towards some undefined state resulting in spurious and inconsistent outputs.

7 Conclusion and Future Work

In this work, we addressed the shortcomings of the prior approaches for the task of author stylized rewriting and overcame them through DRAG: a Director-Generator approach. We showed the effectiveness of our proposed approach for stylized rewriting on three different authors from the

Guteneberg Corpus. Furthermore, we discussed the limitations of our approach and some of the failure cases to aid future research. While our DRAG approach is able to stabilize the training while improving the content preservation abilities of the model, a standard reinforcement learning approach, when stabilized, has the potential to improve these scores to a much more improved level. Improved understanding of author style while keeping a human in the loop and stabilizing RL with transformers models are subjects of future research.

References

Julian Brooke and Graeme Hirst. 2013. A multi-dimensional bayesian approach to lexical style. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 673–679.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Sven Buechel and Udo Hahn. 2017. EmoBank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 578–585, Valencia, Spain. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Kevyn Collins-Thompson and Jamie Callan. 2005. Predicting reading difficulty with statistical language models. *Journal of the American Society for Information Science and Technology*, 56(13):1448–1462.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7059–7069.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Characterizing stylistic elements in syntactic structure. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1522–1533.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*.

Marie Forgeard. 2008. Linguistic styles of eminent writers suffering from unipolar and bipolar mood disorder. *Creativity Research Journal*, 20(1):81–92.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions.

Nikesh Garera and David Yarowsky. 2009. Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 710–718.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Eduard H Hovy. 1990. Pragmatics and natural language generation. *Artificial Intelligence*, 43(2):153–197.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. *arXiv preprint arXiv:1703.00955*.

Diana Inkpen and Graeme Hirst. 2006. Building and using a lexical knowledge base of near-synonym differences. *Computational linguistics*, 32(2):223–262.

Parag Jain, Abhijit Mishra, Amar Prakash Azad, and Karthik Sankaranarayanan. 2019. Unsupervised controllable text formalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6554–6561.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence-to-sequence models. *arXiv preprint arXiv:1707.01161*.

Mark Kantrowitz. 2003. Method and apparatus for analyzing affect and emotion in text. US Patent 6,622,140.

Brett Kessler, Geoffrey Nunberg, and Hinrich Schütze. 1997. Automatic detection of text genre. *arXiv preprint cmp-lg/9707002*.

Shibamouli Lahiri. 2014. Complexity of Word Collocation Networks: A Preliminary Structural Analysis. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 96–105, Gothenburg, Sweden. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*.

Alexander Patrick Mathews, Lexing Xie, and Xuming He. 2016. Senticap: Generating image descriptions with sentiments. In *Thirtieth AAAI conference on artificial intelligence*.

Philip M McCarthy, Gwyneth A Lewis, David F Dufty, and Danielle S McNamara. 2006. Analyzing writing styles with coh-metrix. In *FLAIRS Conference*, pages 764–769.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

Emilio Parisotto, H Francis Song, Jack W Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant M Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. 2019. Stabilizing transformers for reinforcement learning. *arXiv preprint arXiv:1910.06764*.

Ellie Pavlick and Joel Tetreault. 2016. An empirical analysis of formality in online communication. *Transactions of the Association for Computational Linguistics*, 4:61–74.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. *arXiv preprint arXiv:1803.06535*.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Hrituraj Singh, Gaurav Verma, and Balaji Vasan Srinivasan. 2020. Incorporating stylistic lexical preferences in generative language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1074–1079.

Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Bakhtiyar Syed, Gaurav Verma, Balaji Vasan Srinivasan, Anandhavelu Natarajan, and Vasudeva Varma. 2020. Adapting language models for non-parallel author-stylized rewriting. In *AAAI*, pages 9008–9015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Gaurav Verma and Balaji Vasan Srinivasan. 2019. A lexical, syntactic, and semantic perspective for understanding style in text. *arXiv preprint arXiv:1909.08349*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering

Gautier Izacard[1,2,3] **Edouard Grave**[1]
[1] Facebook AI Research, Paris
[2] ENS, PSL University, Paris
[3] Inria, Paris
gizacard|egrave@fb.com

Abstract

Generative models for open domain question answering have proven to be competitive, without resorting to external knowledge. While promising, this approach requires to use models with billions of parameters, which are expensive to train and query. In this paper, we investigate how much these models can benefit from retrieving text passages, potentially containing evidence. We obtain state-of-the-art results on the Natural Questions and TriviaQA open benchmarks. Interestingly, we observe that the performance of this method significantly improves when increasing the number of retrieved passages. This is evidence that sequence-to-sequence models offers a flexible framework to efficiently aggregate and combine evidence from multiple passages.

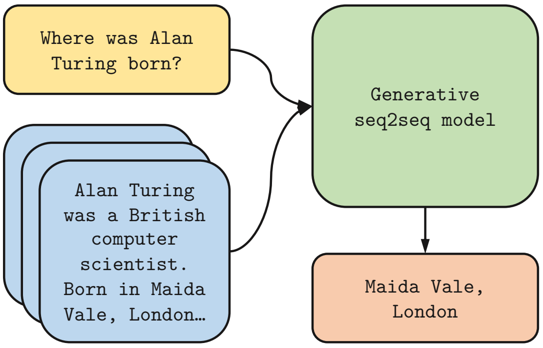


Figure 1: A simple approach to open domain question answering. First, it retrieves support text passages from an external source of knowledge such as Wikipedia. Then, a generative encoder-decoder model produces the answer, conditioned on the question and the retrieved passages. This approach scales well with the number of retrieved passages, as the performance keeps improving when retrieving up to one hundred passages.

1 Introduction

Recently, several works have shown that factual information can be extracted from large scale language models trained on vast quantities of data (Radford et al., 2019; Petroni et al., 2019; Jiang et al., 2019; Talmor et al., 2019). Building on that observation and the advances in pretraining of natural language processing models, Roberts et al. (2020) introduced a generative model for open domain question answering. Without relying on external knowledge, this method obtained competitive results on several benchmarks. However, it requires models containing billions of parameters, since all the information needs to be stored in the weights. This makes models expensive to query and train. In this paper, we investigate how much this method could benefit from having access to an external source of knowledge, such as Wikipedia.

Retrieval based approaches were previously considered in the context of open domain question answering with extractive models (Chen et al., 2017). In that case, systems start by retrieving support documents, before extracting the answer from these documents. Different retrieval techniques have been considered, either using sparse representations based on TF/IDF or using dense embeddings (Guu et al., 2020; Karpukhin et al., 2020). The models which extract the answers are often based on contextualized word representations such as ELMo or BERT (Peters et al., 2018; Devlin et al., 2019), and predict a span as answer. Aggregating and combining evidence from multiple passages is not straightforward when using extractive models, and multiple techniques have been proposed to address this limitation (Clark and Gardner, 2018; Min et al., 2019a).

In this paper, we explore a simple approach having the best of both worlds, by building on the exciting developments in generative modeling and retrieval for open domain question answering. This method proceeds in two steps, by first retrieving supporting passages using either sparse or dense

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 874–880
April 19 - 23, 2021.

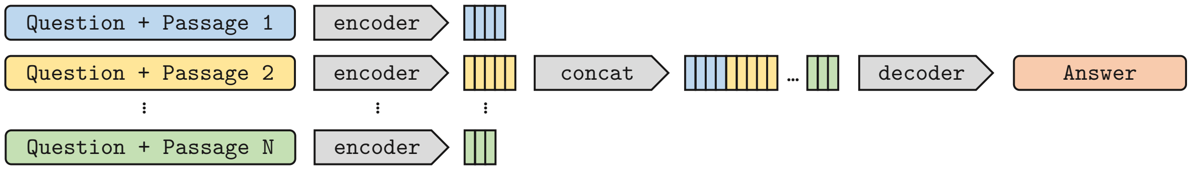


Figure 2: Architecture of the Fusion-in-Decoder method.

representations. Then, a sequence-to-sequence model generates the answer, taking as input the retrieved passages in addition to the question. While conceptually simple, this method sets new state-of-the-art results on the TriviaQA and NaturalQuestions benchmarks. In particular, we show that the performance of our method significantly improves when the number of retrieved passages increases. We believe that this is evidence that generative models are good at combining evidence from multiple passages, compared to extractive ones.

2 Related work

Open domain question answering is the task of answering general domain questions, in which the evidence is not given as input to the system. While being a longstanding problem in natural language processing (Voorhees, 1999), this task has recently regained interest following the work by Chen et al. (2017). In that version of the problem, strong supervision is available to the learning system, in the form of spans corresponding to answers. Chen et al. (2017) proposed to solve the problem by first retrieving support document from Wikipedia, before extracting the answer from the retrieved document. Different methods were proposed to tackle the setting where no gold spans are given to the system, but only the correct answer. Clark and Gardner (2018) proposed to use a global normalization over all the span corresponding to the answer, which was later applied to BERT based models (Wang et al., 2019). Min et al. (2019a) introduced a method based on hard expectation-maximization to tackle noisy supervision from this setting. Wang et al. (2018b) described a technique to aggregate answers from different paragraphs, using confidence and coverage scores.

Passage retrieval is an important step in open domain question answering, and is an active area of research to improve QA systems. Initially, sparse representations based on TF/IDF were used to retrieve support documents (Chen et al., 2017). Lee et al. (2018) introduced a supervised learning method to rerank paragraphs based on BiLSTM, while Wang et al. (2018a) trained a ranking system with reinforcement learning. A second approach to improve the retrieval step of QA systems is to used additional information such as the Wikipedia or Wikidata graphs (Min et al., 2019b; Asai et al., 2020). Recently, multiple works show that retrieval systems entirely based on dense representation and approximate nearest neighbors were competitive with traditional approaches. Such models can be trained using weak supervision in the form of question-answer pairs (Karpukhin et al., 2020), or pretrained using a cloze task and finetuned end-to-end (Guu et al., 2020; Lee et al., 2019).

Generative question answering was mostly considered in previous work for datasets requiring to generate answers, such as NarrativeQA (Kočiskỳ et al., 2018), CoQA (Reddy et al., 2019) or ELI5 (Fan et al., 2019). These datasets were generated in a way that answers do not correspond to spans in support documents, thus requiring abstractive models. Raffel et al. (2020) showed that generative models are competitive for reading comprehension tasks such as SQuAD (Rajpurkar et al., 2016), where answers are spans. Roberts et al. (2020) proposed to use large pretrained generative models, without using additional knowledge, for open domain question answering. Closest to our work, Min et al. (2020) and Lewis et al. (2020b) introduced retrieval augmented generative models for open domain question answering. Our approach differs from these works by how the generative model processes the retrieved passages. This allows to scale to large numbers of documents, and to benefit from this large amount of evidence.

3 Method

In this section, we describe our approach to open domain question answering. It proceeds in two steps, first retrieving support passages before processing them with a sequence to sequence model.

Model	NQ	TriviaQA		SQuAD Open	
	EM	EM	EM	EM	F1
DrQA (Chen et al., 2017)	-	-	-	29.8	-
Multi-Passage BERT (Wang et al., 2019)	-	-	-	53.0	60.9
Path Retriever (Asai et al., 2020)	31.7	-	-	**56.5**	**63.8**
Graph Retriever (Min et al., 2019b)	34.7	55.8	-	-	-
Hard EM (Min et al., 2019a)	28.8	50.9	-	-	-
ORQA (Lee et al., 2019)	31.3	45.1	-	20.2	-
REALM (Guu et al., 2020)	40.4	-	-	-	-
DPR (Karpukhin et al., 2020)	41.5	57.9	-	36.7	-
SpanSeqGen (Min et al., 2020)	42.5	-	-	-	-
RAG (Lewis et al., 2020b)	44.5	56.1	68.0	-	-
T5 (Roberts et al., 2020)	36.6	-	60.5	-	-
GPT-3 few shot (Brown et al., 2020)	29.9	-	71.2	-	-
Fusion-in-Decoder (base)	48.2	65.0	77.1	53.4	60.6
Fusion-in-Decoder (large)	**51.4**	**67.6**	**80.1**	**56.7**	63.2

Table 1: Comparison to state-of-the-art. On TriviaQA, we report results on the open domain test set (left), and on the hidden test set (right), `competitions.codalab.org/competitions/17208#results`).

Retrieval. For the retrieval of support passages, we consider two methods: BM25 (Robertson et al., 1995) and DPR (Karpukhin et al., 2020). In BM25, passages are represented as bag of words, and the ranking function is based on term and inverse document frequencies. We use the implementation from Apache Lucene[1] with default parameters, and tokenize questions and passages with SpaCy.[2] In DPR, passages and questions are represented as dense vector representations, computed using two BERT networks. The ranking function is the dot product between the query and passage representations. Retrieval is performed using approximate nearest neighbors with the FAISS library.[3]

Reading. Our generative model for open domain QA is based on a sequence-to-sequence network, pretrained on unsupervised data, such as T5 or BART (Raffel et al., 2020; Lewis et al., 2020a). The model takes as input the question, as well as the support passages, and generates the answer. More precisely, each retrieved passage and its title are concatenated with the question, and processed independently from other passages by the encoder. We add special tokens `question:`, `title:` and `context:` before the question, title and text of each passage. Finally, the decoder performs attention over the concatenation of the resulting representations of all the retrieved passages. The model thus performs evidence fusion in the decoder only, and we refer to it as *Fusion-in-Decoder*.

By processing passages independently in the encoder, but jointly in the decoder, this method differs from Min et al. (2020) and Lewis et al. (2020b). Processing passages independently in the encoder allows to scale to large number of contexts, as it only performs self attention over one context at a time. This means that the computation time of the model grows linearly with the number of passages, instead of quadratically. On the other hand, processing passages jointly in the decoder allows to better aggregate evidence from multiple passages.

4 Experiments

In this section, we report empirical evaluations of Fusion-in-Decoder for open domain QA.

Datasets. We consider the following datasets, and use the same setting as Lee et al. (2019):

- NaturalQuestions (Kwiatkowski et al., 2019) contains questions corresponding to Google search queries. The open-domain version of this dataset is obtained by discarding answers with more than 5 tokens.

- TriviaQA (Joshi et al., 2017) contains questions gathered from trivia and quiz-league

---

[1] `lucene.apache.org`

[2] `spacy.io`

[3] `github.com/facebookresearch/faiss`

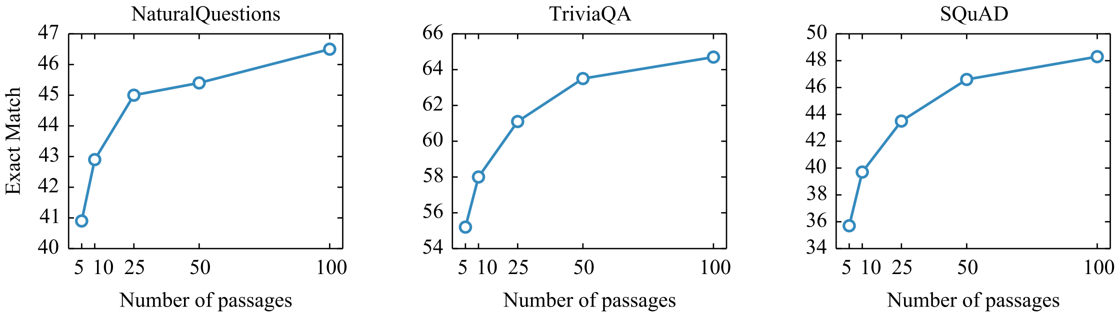


Figure 3: Performance of Fusion-in-Decoder (base) on valid sets as a function of the number of retrieved passages.

websites. The *unfiltered* version of TriviaQA is used for open-domain question answering.

- SQuAD v1.1 (Rajpurkar et al., 2016) is a reading comprehension dataset. Given a paragraph extracted from Wikipedia, annotators were asked to write questions, for which the answer is a span from the corresponding paragraph.

Following Lee et al. (2019) we use the validation as test, and keep 10% of the training set for validation. We use the Wikipedia dumps from Dec. 20, 2018 for NQ and TriviaQA and from Dec. 21, 2016 for SQuAD. We apply the same preprocessing as Chen et al. (2017); Karpukhin et al. (2020), leading to passages of 100 words, which do not overlap.

Evaluation. Predicted answers are evaluated with the standard exact match metric (EM), as introduced by Rajpurkar et al. (2016). A generated answer is considered correct if it matches any answer of the list of acceptable answers after normalization. This normalization step consists in lowercasing and removing articles, punctuation and duplicated whitespace.

Technical details. We initialize our models with the pretrained T5 models (Raffel et al., 2020), available in the HuggingFace Transformers library.[4] We consider two model sizes, base and large, containing respectively 220M and 770M parameters. We fine-tune the models on each dataset independently, using Adam (Kingma and Ba, 2014) with a constant learning rate of 10^{-4} and a dropout rate of 10%. We train the model for 10k gradient steps, with a batch size of 64, using 64 Tesla V100 32Gb. We evaluate models every 500 steps and select the best one on the validation set based on the Exact Match score. During training on NaturalQuestions and SQuAD, we sample the target among the list of answers, while for TriviaQA, we use the unique human-generated answer. For TriviaQA, answers in uppercase are normalized by converting all letters in lowercase except the first letter of each word, using the `title` Python string method. For both training and testing, we retrieve 100 passages (unless said otherwise), and truncate them to 250 word pieces. Following the results of Karpukhin et al. (2020), passages are retrieved with DPR for NQ and TriviaQA, and with BM25 for SQuAD. We generate answers by using greedy decoding.

Comparison to state-of-the-art. In table 1, we compare the results obtained by Fusion-in-Decoder with existing approaches for open domain question answering. We observe that while conceptually simple, this method outperforms existing work on the NaturalQuestion and TriviaQA benchmarks. In particular, generative models seem to perform well when evidence from multiple passages need to be aggregated, compared to extractive approaches. Our method also performs better than other generative models, showing that scaling to large number of passages and processing them jointly leads to improvement in accuracy. Second, we observe that using additional knowledge in generative models by using retrieval lead to important performance gains. On NaturalQuestions, the *closed book* T5 model obtains 36.6% accuracy with 11B parameters, while our approach obtains 44.1% with 770M parameters plus Wikipedia with BM25 retrieval. Both methods use roughly the same amount of memory to store information, indicating that text based explicit memories are competitive for knowledge retrieval tasks.

Scaling with number of passages. In Figure 3, we report the performance with respect to the

---

[4] `github.com/huggingface/transformers`

Training Passages	NaturalQuestions w/o finetuning	NaturalQuestions w/ finetuning	TriviaQA w/o finetuning	TriviaQA w/ finetuning
5	37.8	45.0	58.1	64.2
10	42.3	45.3	61.1	63.6
25	45.3	46.0	63.2	64.2
50	45.7	46.0	64.2	64.3
100	46.5	-	64.7	-

Table 2: Performance depending on the number of passages used during training. Exact Match scores are reported on dev sets.

number of retrieved passages. In particular, we observe that increasing the number of passages from 10 to 100 leads to 6% improvement on TriviaQA and 3.5% improvement on NaturalQuestions. On the other hand, the performance of most extractive models seems to peak around 10 to 20 passages (Wang et al., 2019; Yang et al., 2019). We believe that this is evidence that sequence-to-sequence models are good at combining informations from multiple passages.

Impact of the number of training passages. In the previous section, the model was trained and evaluated with the same number of passages. To reduce the training computational budget, a simple solution consists in training the model with fewer passages. In Table 2, we report the performance obtained by training with different numbers of passages, while testing with 100 passages. We observe that reducing the number of training passages leads to a decrease of accuracy. Further, we propose to finetune the previous models using 100 passages for 1000 steps. This allows to reduce the accuracy gap, while using significantly less computational resources: we can reach 46.0 EM on NaturalQuestions, using 147 GPU hours, compared to 425 GPU hours when training on 100 passages.

5 Conclusion

In this paper, we study a simple approach to open domain question answering, which relies on retrieving support passages before processing them with a generative model. We show that while conceptually simple, this approach is competitive with existing methods, and that it scales well with the number of retrieved passages. In future work, we plan to make this model more efficient, in particular when scaling to large number of support passages. We also plan to integrate the retrieval in our model, and to learn the whole system end-to-end.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 845–855.

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

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2019. How can we know what language models know? *arXiv preprint arXiv:1911.12543*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1601–1611.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

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

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 565–569.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

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

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2851–2864.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5783–5797.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2463–2473.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Technical Report*.

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

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

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5418–5426.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, and Mike Gatford. 1995. Okapi at TREC-3. *NIST Special Publication Sp*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. oLMpics – on what language model pre-training captures. *arXiv preprint arXiv:1912.13283*.

Ellen M Voorhees. 1999. The TREC-8 question answering track report. In *Proceedings of the Eighth Text REtrieval Conference*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018a. R^3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018b. Evidence aggregation for answer re-ranking in open-domain question answering. In *International Conference on Learning Representations*.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5878–5882.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77.

Clinical Outcome Prediction from Admission Notes using Self-Supervised Knowledge Integration

Betty van Aken[1], **Jens-Michalis Papaioannou**[1], **Manuel Mayrdorfer**[2],
Klemens Budde[2], **Felix A. Gers**[1] **and Alexander Löser**[1]

[1]Beuth University of Applied Sciences Berlin
[2]Charité Berlin
{bvanaken,michalis.papaioannou,gers,aloeser}@beuth-hochschule.de
{manuel.mayrdorfer,klemens.budde}@charite.de

Abstract

Outcome prediction from clinical text can prevent doctors from overlooking possible risks and help hospitals to plan capacities. We simulate patients at admission time, when decision support can be especially valuable, and contribute a novel *admission to discharge* task with four common outcome prediction targets: Diagnoses at discharge, procedures performed, in-hospital mortality and length-of-stay prediction. The ideal system should infer outcomes based on symptoms, pre-conditions and risk factors of a patient. We evaluate the effectiveness of language models to handle this scenario and propose *clinical outcome pre-training* to integrate knowledge about patient outcomes from multiple public sources. We further present a simple method to incorporate ICD code hierarchy into the models. We show that our approach improves performance on the outcome tasks against several baselines. A detailed analysis reveals further strengths of the model, including transferability, but also weaknesses such as handling of vital values and inconsistencies in the underlying data.

1 Introduction

Clinical professionals make decisions about patients under strong time constraints. The patient information at hand is often unstructured, e.g. in the form of clinical notes written by other medical personnel in limited time. Clinical decision support (CDS) systems can help in these scenarios by pointing towards related cases or certain risks. Clinical outcome prediction is a fundamental task of CDS systems, in which the patient's development is predicted based on data from their Electronic Health Record (EHR). In this work we focus on textual EHR data available at admission time.

Figure 1 shows a sample admission note with highlighted parts that – according to medical doctors – must be considered when evaluating a patient.

Encoding clinical notes with pre-trained language models. Neural models need to extract relevant facts from such notes and learn complex relations between them in order to associate certain clinical outcomes. Pre-trained language models such as BERT (Devlin et al., 2019) have shown to be able to both extract information from noisy text and to capture task-specific relations in an end-to-end fashion (Tenney et al., 2019; van Aken et al., 2019). We thus base our work on these models and pose the following questions:

- Can pre-trained language models learn to predict patient outcomes from their admission information only?
- How can we integrate knowledge about outcomes that doctors gain from medical literature and previous patients?
- How well would these models work in clinical practice? Are they able to interpret common risk factors? Where are they failing?

Simulating patients at admission time. Existing work on text-based outcome prediction focuses on progress notes after a certain time of a patient's hospitalisation (Huang et al., 2019). This is mostly due to a lack of publicly available admission notes and poses some problems: 1) Doctors might miss specific outcome risks early in admission and 2) progress notes already contain information about clinical decisions made on admission time (Boag et al., 2018). We propose to simulate newly arrived patients by extracting admission notes from MIMIC III discharge summaries. We are thus able

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 881–893
April 19 - 23, 2021.

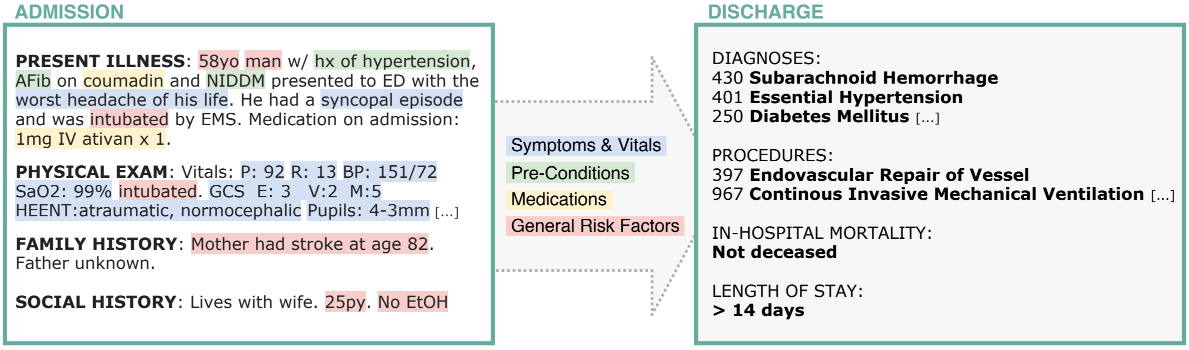


Figure 1: *Admission to discharge* sample that demonstrates the outcome prediction task. The model has to extract patient variables and learn complex relations between them in order to predict the clinical outcome.

to give doctors hints towards possible outcomes from the very beginning of an admission and can potentially prevent early mistakes. We can also help hospitals in planning resources by indicating how long a patient might stay hospitalised.

Integrating knowledge with specialised outcome pre-training. Gururangan et al. (2020) recently emphasized the importance of domain- and task-specific pre-training for deep neural models. Consequently we propose to enhance language models pre-trained on the medical domain with a task-specific *clinical outcome pre-training*. Besides processing clinical language with idiosyncratic and specialized terms, our models are thus able to learn about patient trajectories and symptom-disease associations in a self-supervised manner. We derive this knowledge from two main sources: 1) Previously admitted patients and their outcomes. This knowledge is usually stored by hospitals in unlabelled clinical notes and 2) Scientific case reports and knowledge bases that describe diseases, their presentations in patients and prognoses. We introduce a method for incorporating these sources by creating a suitable pre-training objective from publicly available data.

Contributions. We summarize the major contributions of this work as follows:
1) A novel task setup for clinical outcome prediction that simulates the patient's admission state and predicts the outcome of the current admission.
2) We introduce self-supervised *clinical outcome pre-training*, which integrates knowledge about patient outcomes into existing language models.
3) We further propose a simple method that injects hierarchical signals into ICD code prediction.
4) We compare our approaches against multiple baselines and show that they improve performance on four relevant outcome prediction tasks with up to 1,266 classes. We show that the models are transferable by applying them to a second public dataset without additional fine-tuning.
5) We present a detailed analysis of our model that includes a manual evaluation of samples conducted by medical professionals.

2 Related Work

Using clinical notes for outcome prediction. Boag et al. (2018) studied the predictive value of clinical notes with simple approaches such as bag-of-words. Recent work increasingly applies neural models to compensate for the noisy nature of the data and the complexity of patterns. Hashir and Sawhney (2020) used both convolutional and recurrent layers for outcome prediction, while Jain et al. (2019) and Qiao et al. (2019) proposed attention-based approaches. Dligach et al. (2019) explored pre-training as a strategy to mitigate data sparsity in clinical setups. Si and Roberts (2019) and Suresh et al. (2018) further showed that outcome prediction benefits from a multitask setup. In contrast to earlier work we apply neural models to admission notes in an *admission to discharge* setup.

Pre-trained language models for the clinical domain. While pre-trained language models are successful in many areas of NLP, there has been little work on applying them to the clinical domain (Qiu et al., 2020). Alsentzer et al. (2019) and Huang et al. (2019) both pre-trained BERT-based models on clinical data. They evaluated their work on readmission prediction and other NLP tasks. We are the first to evaluate pre-trained language models on multiple clinical outcome tasks with large label sets. We further propose a novel pre-training objective specifically for the clinical domain.

Prediction of diagnoses and procedures. The majority of work on diagnosis and procedure prediction covers either single diagnoses (Liu et al., 2018; Choi et al., 2018) or coarse-grained groups (Peng et al., 2020; Sushil et al., 2018). We argue that models should predict diseases and procedures in a fine-grained manner to be beneficial for doctors. Thus we use all diagnosis and procedure codes from the data for our outcome prediction tasks.

ICD coding vs. outcome prediction. There is a variety of work in the related field of automated ICD coding (Xie et al., 2018; Falis et al., 2019). Zhang et al. (2020) recently presented a model able to identify up to 2,292 ICD codes from text. However, ICD coding differs from outcome prediction in the way that diseases are directly extracted from text rather than inferred from symptom descriptions and patient history. We further discuss this distinction in Section 6.

3 Clinical *Admission to Discharge* Task

Clinical outcome prediction can be defined in different ways. We approach the task from a doctor's perspective and predict the outcome of a current admission from the time of the patient's arrival to the hospital unit. We describe our setup as follows.

3.1 Clinical Notes from MIMIC III

As our primary data source, we use the freely-available MIMIC III v1.4 database (Johnson et al., 2016). It contains de-identified EHR data including clinical notes in English from the Intensive Care Unit (ICU) of Beth Israel Deaconess Medical Center in Massachusetts between 2001 and 2012. We focus our work on discharge summaries in particular and the outcome information associated with an admission. Similar to previous work, we filter out notes about newborns and remove duplicates.

3.2 Creating Admission Notes from Discharge Summaries

The state of a patient is commonly summarized in an ongoing document, which finally concludes in a discharge summary. Since we want to support clinical decisions from the beginning of a patient's stay, we simulate the state of the patient's document at admission time. We thus filter the document by sections that are known at admission such as: *Chief complaint, (History of) Present illness, Medical history, Admission Medications, Allergies, Physical exam, Family history* and *Social history*. We further describe the filtering in Appendix B.1. Our approach results in 48,745 admission notes. As shown in Table 1 the notes contain about 400 words on average. The selection of admission sections as well as the resulting structure of the notes were verified by medical doctors.

Admission Notes Statistics			
avg (words / doc)	std (words / doc)	avg (sent / doc)	std (sent / doc)
396.3	233.3	32.5	23.1

Table 1: Numbers of words / sentences in MIMIC III admission notes. We see a high variation in length.

Multi-label tasks: ICD-9 codes per dataset split							
Diagnoses				Procedures			
Total	Train	Val	Test	**Total**	Train	Val	Test
1,266	1,201	906	1,031	**711**	672	476	563

Table 2: Distribution of ICD-9 codes per dataset split (patient-wise). Note that very rare codes do not appear in each split of the dataset.

Single-label tasks: Samples per class					
Mortality		Length of Stay (in days)			
0	1	≤ 3	> 3 & ≤ 7	> 7 & ≤ 14	> 14
43,609	5,136	5,596	16,134	13,391	8,488

Table 3: Distribution of labels for *Mortality Prediction* and *Length of Stay* task. Both tasks have unbalanced class distributions.

This newly created admission dataset enables us to make predictions on the outcome of a current admission. At inference time, doctors can then use the model's predictions on textual data from newly arrived patients.

3.3 Outcome Prediction Tasks

We select four relevant tasks for outcome prediction in consultation with medical professionals. All tasks take admission notes as input.

Diagnosis prediction. A main goal of clinical outcome prediction is to support medical professionals in the process of differential diagnosis. We thus take all diagnoses associated with an admission into account and frame the task as an extreme multi-label classification. Diagnoses are encoded as ICD-9 codes in the MIMIC III database. Following Choi et al. (2017), we group ICD-9 diagnosis codes from the database from 4- into 3-digit codes to reduce complexity while still obtaining granular suggestions. This results in a total of 1,266 diag-

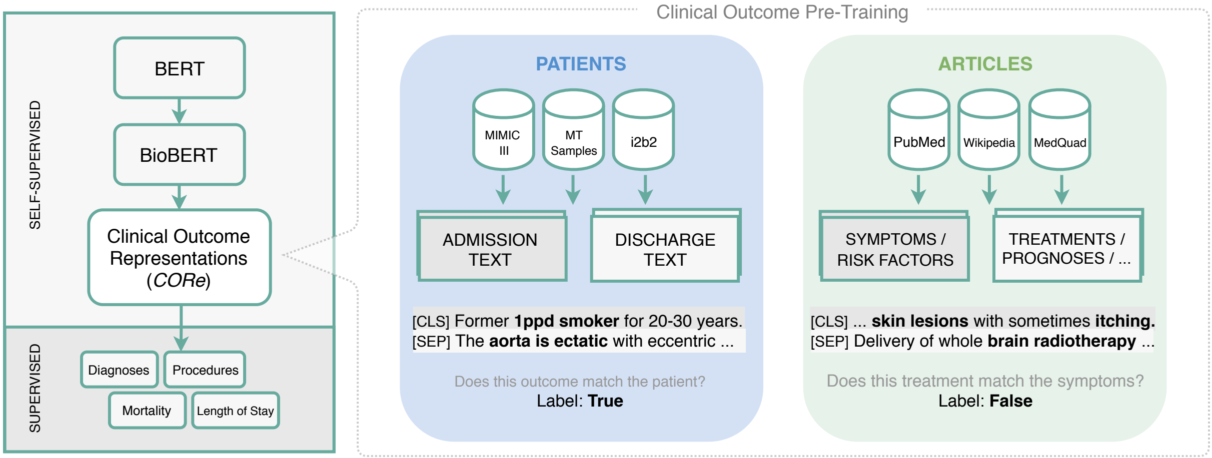


Figure 2: Schematic demonstration of *clinical outcome pre-training*. Sources of clinical knowledge are complete patient notes and medical articles. Based on that we create a self-supervised learning objective that teaches relations between symptoms, risk factors and outcomes.

nosis codes, which are distributed over our dataset splits as shown in Table 2. The labels are power-law distributed with a long tail of very rare codes.

Procedure prediction. Procedures are either diagnostics or treatments applied to a patient during a stay. Similarly to diagnosis prediction, this is an extreme multi-label task. We again group the ICD-9 codes from the MIMIC III database into 3-digit codes. In total there are 711 procedure codes labelled in the database in a power law distribution similar to the diagnosis codes.

In-hospital mortality prediction. Predicting a patient's mortality risk is a fundamental part of the triage process. In-hospital mortality in particular describes whether a patient died during the current admission and is a binary classification task. The percentage of deceased patients in the data is around 10% (see Table 3). As some notes contain direct indications of mortality such as *patient deceased* within the admission sections, we apply an additional filter for those terms.

Length-of-stay prediction. The duration of an ICU stay is an important information for hospitals in order to plan allocations of resources. We group patients into four major categories regarding their length of stay: *Under 3 days, 3 to 7 days, 1 week to 2 weeks, more than 2 weeks.* These categories were recommended by medical doctors in order to make the results as useful as possible in clinical practice. Table 3 shows the samples per class.

4 Integrating Clinical Knowledge Into Language Models

We propose *clinical outcome pre-training*, a way to integrate knowledge about clinical outcomes into pre-trained language models. We further introduce an additional step to incorporate ICD code hierarchy into our multi-label classification tasks.[1]

4.1 Clinical Outcome Pre-Training

Motivation. Language model pre-training has shown to be of use in specialised domains like the clinical (Alsentzer et al., 2019; Huang et al., 2019). However, these models lack knowledge about patient trajectories and symptom-diagnosis relations, because their training is focused on learning language characteristics.
We develop an additional pre-training step that produces *Clinical Outcome Representations (CORe)* in order to teach the model relations between symptoms, risk factors and clinical outcomes. Much of this knowledge is present and publicly available, e.g. in knowledge bases like Wikipedia or publication archives like PubMed. Another source is available to hospitals in the form of unlabelled clinical notes from previous patients. The suggested outcome pre-training is a way to use this knowledge to improve the model's capabilities in predicting clinical outcomes as described in 3.3.
Corresponding to the way doctors gain their knowledge from both experience and medical literature,

---

[1] The code to recreate the experiments and datasets described in this paper is accessible at: `https://github.com/bvanaken/clinical-outcome-prediction`

we incorporate knowledge from complete patient notes (including discharge information) and medical articles.

Training objective. Our proposed training objective (Figure 2) is strongly related to the Next Sentence Prediction (NSP) task introduced by Devlin et al. (2019). In NSP the model gets two sentences as an input and predicts whether the second follows the first sentence. This way models such as BERT learn relations between sentences. We convert this setting so that the model instead learns relations between admissions and outcomes.

From common sections in patient notes, we create two categories: Sections that are created at admission A and sections that are created after admission, e.g. at discharge time D. Given a patient note N, we split it into sections $A_N \in A$ and $D_N \in D$. We remove all other sections. We then sample token sequences from these sections to get $t_{N,1...k} \in A_N$ and $t'_{N,1...k} \in D_N$, where k is randomly set between 30 and 50 tokens. We then train the model to maximize $P(Same\_Patient|X_{N\_N})$ and $P(Other\_Patient|X_{N\_M})$ with

$$\begin{aligned} X_{N\_N} &= Enc(t_{N,1...k}, t'_{\mathbf{N},1...k}) \\ X_{N\_M} &= Enc(t_{N,1...k}, t'_{\mathbf{M},1...k}) \end{aligned} \tag{1}$$

with M being a randomly sampled document from the same batch and Enc referring to the BioBERT encoding. As in the original NSP setting, we apply negative sampling (X_{N_M}) for 50% of examples. We apply the same strategy on medical articles and case reports, so that A represents sections describing symptoms and risk factors, and D represents sections that describe outcomes of a disease or case.

Data sources. We create the pre-training dataset from multiple public sources. To integrate knowledge that doctors gain from previous patients and medical literature, we create two groups of sources:
1) *Patients*, which includes 32,721 discharge summaries from the MIMIC III training set, 5,000 publicly available medical transcriptions from the MT-Samples website [2] and 4,777 clinical notes from the i2b2 challenges 2006-2012[3] (Uzuner et al., 2007, 2008, 2010a,b, 2011, 2012; Sun et al., 2013b,a).
2) *Articles*, composed of 9,335 case reports from PubMed Central (PMC), 2,632 articles from Wikipedia describing diseases and 1,467 article sections from the MedQuAd dataset (Abacha and Demner-Fushman, 2019) extracted from NIH websites such as cancer.gov.

While *Patients* samples contain unaudited practical knowledge, *Articles* samples are built from verified general medical knowledge such as peer-reviewed studies. The sources are therefore substantially different and we evaluate their individual effect on performance in Section 5.3.

Data preparation. We create admission (A_N) and discharge parts (D_N) of the documents based on section headings. We define common sections belonging to the admission part and those belonging to the discharge part similar to the method described in Section 3.2. We ignore sections that cannot be categorized. For section heading extraction from MIMIC III discharge summaries and MTSamples transcriptions, we apply simple rule-based approaches, which is feasible because the notes are well-structured. For Wikipedia we use headings from the WikiSection dataset (Arnold et al., 2019) filtered for disease articles only. For PubMed Central we similarly use the PubMedSection dataset (Schneider et al., 2020) and filter for section headings that indicate case reports. As i2b2 notes are less well-structured in comparison to MIMIC III discharge summaries, we use a classifier as proposed by Rosenthal et al. (2019) to determine which section a sentence belongs to. The classifier is trained on an annotated set of i2b2 notes and then applied to all other notes.

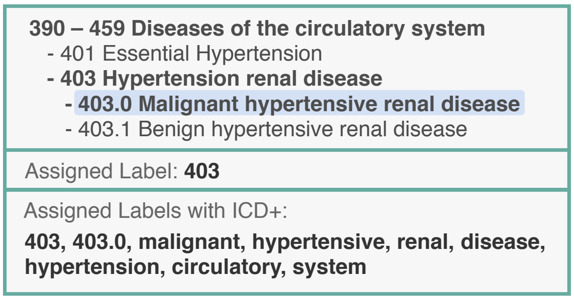
4.2 ICD+: Incorporation of ICD Hierarchy

Medical knowledge in ICD labels. Diagnosis and procedure prediction requires the model to predict ICD-9 codes in a multi-label manner. ICD-9 codes are hierarchically ordered into associated groups. Figure 3 shows the code hierarchy for *Malignant hypertensive renal disease* with the ICD-9 code *403.0*. The diagnosis has two parent groups namely *Hypertension renal disease* and *Diseases of the circulatory system*. Diagnoses or procedures in the same group often share similar medical characteristics, therefore hierarchical relations of a labelled code can be valuable information. This medical information is currently not integrated into the model. The same holds for words describing the ICD-9 codes, that often represent further important signals, such as the words *renal* or *malignant*.

---

[2] https://mtsamples.com

[3] We exclude notes from the 2014 De-identification and Heart Disease Risk Factors Challenge in order to use this set for evaluation as described in Section 5.4.



390 – 459 Diseases of the circulatory system
- 401 Essential Hypertension
- **403 Hypertension renal disease**
  - **403.0 Malignant hypertensive renal disease**
  - 403.1 Benign hypertensive renal disease

Assigned Label: **403**

Assigned Labels with ICD+:

403, 403.0, malignant, hypertensive, renal, disease, hypertension, circulatory, system

Figure 3: Example of *ICD+* labelling. *Malignant hypertensive renal disease* is assigned to nine codes (bottom row) that inform about the type and group of the disease.

Enhancing training with useful additional signals. We propose a simple method, *ICD+*, to incorporate both associated groups and words into the model weights: Instead of only classifying 3-digit codes (as mentioned in 3.3), we let the model additionally predict the 4-digit codes and the bag of associated words with a code and its parent groups. In order to create the bag of words per code, we use the descriptions of ICD-9 codes from MIMIC III and remove all stop words. As shown in Figure 3, the *ICD+* method assigns eight additional labels to the example diagnosis and therefore supplies the model with further information about the diagnosis during training.

By increasing the amount of labels per sample, we integrate relevant medical knowledge and enable the model to learn implicit relations between codes and code groups that share certain words. We evaluate the effectiveness of *ICD+* in Section 5.

5 Experimental Evaluation

5.1 Training Clinical Outcome Representations

We pre-train the *CORe* model on top of BioBERT weights[4]. We then fine-tune the model separately on the four outcome tasks. We use the same training regimen for both pre-training and fine-tuning: We tokenize the texts with WordPiece tokenization and truncate them to 512 tokens, due to the limited context length of the pre-trained models. We use early stopping and tune hyperparameters as described in Appendix C.

5.2 Baseline Models

In the following, we introduce the baseline models that we evaluate on the novel outcome prediction tasks. In order to understand the abilities of pre-trained language models we compare their performance against more traditional approaches. The first three models (*BOW, word embeddings, CNN*) are trained using the hyperparameters proposed by the authors for outcome prediction tasks. The language models are fine-tuned the same way as the *CORe* model.

Bag-of-Words. Boag et al. (2018) shows that a simple bag-of-words (BOW) approach can outperform more complex models on tasks like mortality prediction. We thus include their approach in our evaluation. We adopt their training setting except that we consider 200 instead of 20 top tf-idf words in order to make the model converge.

Pre-trained word embeddings. Boag et al. (2018) further propose the use of pre-computed word embeddings that were trained on MIMIC III data. We use the same setting as for the BOW approach and fit a support vector machine classifier on the clinical outcome tasks.

Convolutional Neural Network (CNN). Si and Roberts (2019) built a neural network for mortality prediction with two hierarchical convolutional layers at the word and sentence levels and then aggregated it to a patient level representation. We follow their approach to evaluate the model on our four *admission to discharge* tasks.

BioBERT. Following the success of BERT, Lee et al. (2020) further pre-trained the model on biomedical research articles from PubMed using abstracts and full-text articles. They reported improved performance on a range of biomedical text mining tasks.

ClinicalBERT and DischargeBERT. We further evaluate two public language models pre-trained on the clinical domain, with MIMIC III data in particular. Huang et al. (2019) pre-trained a BERT Base model on 100,000 random clinical notes (ClinicalBERT) while Alsentzer et al. (2019) further pre-trained BioBERT on all discharge summaries from MIMIC III (we refer to the model as DischargeBERT for simplicity).

---

[4] We choose BioBERT as the base for our model because it outperforms BERT on medical tasks and has not seen data from our test set during pre-training unlike DischargeBERT.

	Diagnoses (1266 classes)	Procedures (711 classes)	In-Hospital Mortality (2 classes)	Length-of-Stay (4 classes)
BOW (Boag et al., 2018)	75.87	77.47	79.15	65.83
Embeddings (Boag et al., 2018)	75.16	76.72	79.94	66.78
CNN (Si and Roberts, 2019)	61.18	73.13	75.50	64.49
BERT Base (Devlin et al., 2019)	82.08	85.84	81.13	70.40
ClinicalBERT (Huang et al., 2019)	81.99	86.15	82.20	71.14
DischargeBERT (Alsentzer et al., 2019)	*82.86*	*87.09*	***84.51***	*71.73*
BioBERT Base (Lee et al., 2020)	82.81	86.36	82.55	71.59
BioBERT ICD+	83.17	87.45	-	-
CORe Articles (w/o ICD+)	83.46 (82.89)	87.43 (86.75)	83.64	71.99
CORe Patients (w/o ICD+)	83.41 (83.40)	**88.37** (86.60)	83.60	71.96
CORe All (w/o ICD+)	**83.54** (83.39)	87.65 (87.15)	**84.04**	**72.53**

Table 4: Results on outcome prediction tasks in macro-averaged % AUROC. The *CORe* models outperform the baselines, *ICD+* adds further improvement (values in parentheses are ablation results without *ICD+*). DischargeBERT results are printed in italic because the model has seen all test data during pre-training and is therefore slightly advantaged.

5.3 Results on MIMIC III Admission Notes

Table 4 shows performances in (macro-averaged) area under the receiver operating characteristic curve (AUROC). We report scores of the *CORe* model trained only on *Articles*, *Patients* and in a combined training setting *CORe All*. We evaluate diagnosis and procedure prediction both with and without the *ICD+* method on BioBERT and the *CORe* models. In both scenarios we evaluate on 3-digit ICD codes only, in order to maintain comparability between the methods.

Pre-trained models outperform baselines. We see that the evaluated pre-trained language models clearly outperform the *BOW*, *word embeddings* and *CNN* approaches. We further observe that the *CORe* models improve scores on all tasks in comparison to the baseline models, except for DischargeBERT that reaches a higher score in mortality prediction – probably affected by its exposure to the test data. This shows that even though the language models are trained on similar data (e.g. PubMed and/or clinical notes), the specific *outcome pre-training* improves the model's ability to predict clinical outcome targets. Pre-training on *Patients* and *Articles* achieve similar improvements over the baselines, while the combined training is the most effective. An exception is the procedure prediction, where pre-training on *Patients* achieves the highest score. A probable reason is that procedures are documented in more detail in clinical notes, especially since our selection of medical articles focuses on diseases rather than procedures.

Predicting mortality risk is easier than length of stay. We see that the models reach higher scores in the binary mortality task than in length of stay prediction. Even a simple *BOW* approach can reach a relatively high score, which indicates that most of the notes contain clear hints towards an increased mortality risk. On the other hand, the length of stay task is difficult due to the many factors that can contribute to the length of a patient's stay after the admission, including nonclinical factors such as the patient's insurance situation (Khosravizadeh et al., 2016).

ICD hierarchy improves diagnosis and procedure predictions. Table 4 shows an ablation test without the *ICD+* method (in parentheses). We see that both the BioBERT model and the *CORe* models improve when incorporating code hierarchy and relations through *ICD+* into the training process. This is especially visible for ICD procedures, where the hierarchical and textual information, e.g. that a *Nephropexy* is an *operation* on the *kidney* can add important signals during training.

	i2b2 Diagnoses
BioBERT ICD+	80.43
CORe Articles	81.46
CORe Patients	**82.31**
CORe All	81.15

Table 5: Results on i2b2 diagnosis prediction task (5 classes) in % AUROC. The models reach similar results as on the MIMIC III data, indicating their transferability to other data sources without additional fine-tuning.

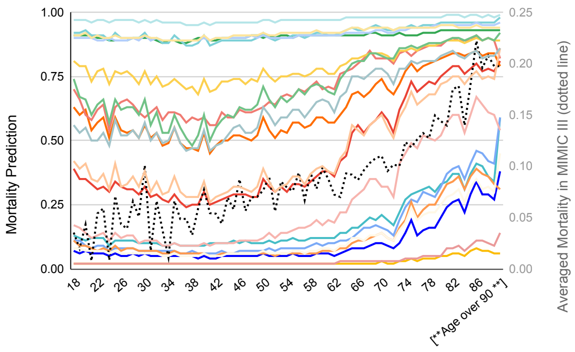


Figure 4: Impact of age on mortality prediction on 20 random samples. Mortality risk and age mostly increase proportionally as intended, with certain peaks that might indicate unintended biases in the data.

5.4 Model Transferability: Cross-Verification on i2b2 Clinical Notes

In order to verify that the fine-tuned models are transferable to ICU data from other sources, we apply it to data from the i2b2 De-identification and Heart Disease Risk Factors Challenge (Stubbs et al., 2015). We convert the clinical notes to admission notes as further described in Appendix B.2, which results in 1,118 samples labelled with up to five ICD-9 codes.

Models generalize to i2b2 data. We apply our MIMIC III-based models to predict diagnosis codes for the i2b2 notes without further fine-tuning. We then evaluate based on whether the predictions contain the five mentioned ICD-9 codes. The results in macro-averaged % AUROC are shown in Table 5. Even though the clinical notes differ from the MIMIC III notes in structure and writing style, the tested models are mostly able to identify the conditions. The scores are comparable to the MIMIC III results, which shows that the models are able to generalise on data from different sources such as other hospitals.

6 Discussion and Findings

Clinical outcome prediction is a sensitive task. We therefore conduct an extensive analysis on the *CORe All* model including a manual error analysis by medical doctors on 20 randomly chosen samples to understand how the model would perform in clinical practice.[5]

	% AUROC
All Diagnoses	83.54
Diagnoses **Mentioned** in Text	**87.10**
Diagnoses **Not Mentioned** in Text	82.35

Table 6: Analysis of the impact of directly mentioned diagnoses on the diagnosis prediction task. Mentioned diagnoses are detected more reliably. Though on unmentioned diagnoses, scores only see a small decrease compared to the overall score.

6.1 A Closer Look at the Model's Abilities

Does the model mainly extract already present diagnoses? We observe that a majority of coded diseases are already mentioned in the admission text. This is mainly due to chronic diseases (e.g. *diabetes mellitus*) or to conditions that were identified prior to the ICU admission (e.g. in the emergency ward). We want to know if our model is also able to predict diagnoses that are not mentioned in the text. We annotate the admission texts with ICD-9 diagnosis codes with the methodology described by Searle et al. (2020). We then evaluate on codes that were explicitly mentioned in the text and those that were not. Table 6 shows that the model indeed extracts many diagnoses directly from the text and thus reaches a higher score on mentioned diagnoses. On the other hand, we see that the performance on non-mentioned diagnoses does drop only slightly, indicating that the model has also learned to predict non-mentioned diagnoses.

How does age and gender impact predictions? Age and gender are common risk factors with significant impact on the potential clinical outcome of a patient. We want our models to learn that impact without overestimating it. We test the model's behaviour by switching age and gender throughout 20 random samples and analyse how the mortality prediction changes. For each sample we manually switch the age mention and iterate over it from 18 until [**Age over 90**][6]. Figure 4 shows that the analysed samples show a high variation in mortality risk and that age only impacts the prediction partially. In all cases the prediction increases with age – as expected from a medical perspective. We also observe some peaks without a medical reason that are caused by the mortality of certain age groups in the original data (black dotted line). This demonstrates how the model does not follow medical reasoning but merely statistic observations. We

---

[5] Our demo application used for this analysis is available at: `https://outcome-prediction.demo.datexis.com`

[6] De-identified age information for patients older than 89.

similarly switch the gender mention and all pronouns in the texts and observe that mortality prediction for male patients is increased by 5% on average, consistent with medical rationale.

Where is the model failing?

1. **Negation**: While our error analysis depicts that negation does not generally falsify the model's predictions, we find single samples in which especially medical-specific negations, such as *abstinent from alcohol*, are misinterpreted by the model, e.g. into *alcohol dependence syndrome*.
2. **Numerical data**: Wallace et al. (2019) show BERT's inabilities to interpret numbers. We observe this in the case that the model does not interpret life-threatening vital values (such as temperature over 105°F) as an increased mortality risk. Clinical notes contain many such relevant values, thus improving the encoding of such data is an important goal for future work.

6.2 There is no Ground Truth in Clinical Data

Incomplete and inconsistent labels. Our error analysis reveals that 60% of the analysed samples are partially under-coded. They contain indicators for a diagnosis or procedure but miss the corresponding ICD-9 code. This is consistent with results from Searle et al. (2020) showing that MIMIC III is up to 35% under-coded. Additionally we find that procedures that are almost always performed in the ICU such as *Puncture of vessel* are often coded inconsistently. While a doctor can infer these labels with medical common sense, they pose a challenge to our models. We therefore suggest a critical view towards the data and welcome additional clinical datasets to compensate for noisy labels.

Multiple possible outcomes. 85% of analysed samples contain false positive predictions that the doctors still consider medically reasonable. This demonstrates that there are many possible clinical pathways and that some might not be foreseeable at admission time. We also see many cases in which the information in the clinical note is not sufficient and therefore allows multiple interpretations. For future work, we propose including further EHR data as suggested by Khadanga et al. (2019) to extend the patient representation in these scenarios.

7 Conclusion

We reframe the task of clinical outcome prediction to consider the admission state of a patient and support doctors in their initial decision process. We show that current state-of-the-art language models outperform selected baselines on this task and present methods for further improvement: *Outcome pre-training* enables our models to learn from unlabelled sources and *ICD+* incorporates hierarchical and textual ICD representations into our models. For future work, we suggest considering pre-trained language models with larger context sizes (Beltagy et al., 2020; Zaheer et al., 2020) and languages other than English (Reys et al., 2020). We further encourage work on semantic encoding of negated terms and numerical data from clinical text.

Acknowledgments

We would like to thank Anjali Grover and Sebastian Herrmann for their support throughout the project. Our work is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) under grant agreement 01MD19003B (PLASS) and 01MK2008MD (Servicemeister).

References

Asma Ben Abacha and Dina Demner-Fushman. 2019. A question-entailment approach to question answering. *BMC Bioinformatics*, 20(1):511:1–511:23.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions?: A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019*, pages 1823–1832, Beijing, China. ACM.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly Available Clinical BERT Embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. ACL.

Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A. Gers, and Alexander Löser. 2019. SECTOR: A Neural Model for Coherent Topic Segmentation and Classification. *Transactions of the Association for Computational Linguistics, TACL*, 7:169–184.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *Computing Research Repository*, arXiv/2004.05150.

Willie Boag, Dustin Doss, Tristan Naumann, and Peter Szolovits. 2018. What's in a Note? Unpacking Predictive Value in Clinical Note Representations. *AMIA Summits on Translational Science Proceedings*, 2018:26 – 34.

Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. 2017. Generating Multi-label Discrete Patient Records using Generative Adversarial Networks. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC*, volume 68 of *Proceedings of Machine Learning Research*, pages 286–305, Boston, Massachusetts. PMLR.

Edward Choi, Cao Xiao, Walter F. Stewart, and Jimeng Sun. 2018. MiME: Multilevel Medical Embedding of Electronic Health Records for Predictive Healthcare. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 4552–4562, Montréal, Canada.

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

Dmitriy Dligach, Majid Afshar, and Timothy A. Miller. 2019. Toward a clinical text encoder: pretraining for clinical natural language processing with applications to substance misuse. *J. Am. Medical Informatics Assoc.*, 26(11):1272–1278.

Matús Falis, Maciej Pajak, Aneta Lisowska, Patrick Schrempf, Lucas Deckers, Shadia Mikhael, Sotirios A. Tsaftaris, and Alison O'Neil. 2019. Ontological attention ensembles for capturing semantic concepts in ICD code prediction from clinical text. In *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis, LOUHI@EMNLP 2019*, pages 168–177, Hong Kong, China. ACL.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 8342–8360, Online. ACL.

Mohammad Hashir and Rapinder Sawhney. 2020. Towards unstructured mortality prediction with free-text clinical notes. *Journal of Biomedical Informatics*, 108:103489.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission. In *Proceedings of ACM Conference on Health, Inference, and Learning, CHIL 2020*, Online. ACM.

Sarthak Jain, Ramin Mohammadi, and Byron C. Wallace. 2019. An Analysis of Attention over Clinical Notes for Predictive Tasks. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 15–21, Minneapolis, Minnesota, USA. ACL.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3:160035.

Swaraj Khadanga, Karan Aggarwal, Shafiq R. Joty, and Jaideep Srivastava. 2019. Using Clinical Notes with Time Series Data for ICU Management. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 6431–6436, Hong Kong, China. ACL.

Omid Khosravizadeh, Soudabeh Vatankhah, Peivand Bastani, Rohollah Kalhor, Samira Alirezaei, and Farzane Doosty. 2016. Factors affecting length of stay in teaching hospitals of a middle-income country. *Electronic physician*, 8(10):3042–3047.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Jingshu Liu, Zachariah Zhang, and Narges Razavian. 2018. Deep EHR: Chronic Disease Prediction Using Medical Notes. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC 2018*, volume 85 of *Proceedings of Machine Learning Research*, pages 440–464, Palo Alto, California, USA. PMLR.

Xueping Peng, Guodong Long, Tao Shen, Sen Wang, and Jing Jiang. 2020. Self-Attention Enhanced Patient Journey Understanding in Healthcare System. In *Proceedings of The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2020*, Online.

Zhi Qiao, Xian Wu, Shen Ge, and Wei Fan. 2019. MNN: Multimodal Attentional Neural Networks for Diagnosis Prediction. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 5937–5943, Macao, China.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained

Models for Natural Language Processing: A Survey. *Science China Technological Sciences*, 63:1872 – 1897.

Arthur D. Reys, Danilo Silva, Daniel Severo, Saulo Pedro, Marcia M. de Souza e Sá, and Guilherme A. C. Salgado. 2020. Predicting Multiple ICD-10 Codes from Brazilian-Portuguese Clinical Notes. In *Proceedings of the 9th Brazilian Conference on Intelligent Systems (BRACIS)*, Rio Grande, Brazil.

Sara Rosenthal, Ken Barker, and Zhicheng Liang. 2019. Leveraging Medical Literature for Section Prediction in Electronic Health Records. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4864–4873, Hong Kong, China. ACL.

Rudolf Schneider, Tom Oberhauser, Paul Grundmann, Felix Alexander Gers, Alexander Loeser, and Steffen Staab. 2020. Is Language Modeling Enough? Evaluating Effective Embedding Combinations. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4739–4748, Marseille, France. ELRA.

Thomas Searle, Zina M. Ibrahim, and Richard J. B. Dobson. 2020. Experimental Evaluation and Development of a Silver-Standard for the MIMIC-III Clinical Coding Dataset. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing, BioNLP 2020, Online, July 9, 2020*, pages 76–85. ACL.

Yuqi Si and Kirk Roberts. 2019. Deep Patient Representation of Clinical Notes via Multi-Task Learning for Mortality Prediction. *AMIA Summits on Translational Science Proceedings*, 2019:779–788.

Amber Stubbs, Christopher Kotfila, and Özlem Uzuner. 2015. Automated systems for the de-identification of longitudinal clinical narratives: Overview of 2014 i2b2/UTHealth shared task Track 1. *Journal of Biomedical Informatics*, 58:S11–S19.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/UTHealth corpus. *Journal of Biomedical Informatics*, 58:S20–S29.

Weiyi Sun, Anna Rumshisky, and Özlem Uzuner. 2013a. Annotating temporal information in clinical narratives. *Journal of Biomedical Informatics*, 46(6):S5–S12.

Weiyi Sun, Anna Rumshisky, and Özlem Uzuner. 2013b. Evaluating temporal relations in clinical text: 2012 i2b2 Challenge. *Journal of the American Medical Informatics Association*, 20(5):806–813.

Harini Suresh, Jen J. Gong, and John V. Guttag. 2018. Learning Tasks for Multitask Learning: Heterogenous Patient Populations in the ICU. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018*, pages 802–810, London, UK. ACM.

Madhumita Sushil, Simon Šuster, Kim Luyckx, and Walter Daelemans. 2018. Patient representation learning and interpretable evaluation using clinical notes. *Journal of Biomedical Informatics*, 84:103 – 113.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019*, New Orleans, LA, USA.

Özlem Uzuner, Andreea Bodnari, Shuying Shen, Tyler Forbush, John Pestian, and Brett R. South. 2012. Evaluating the state of the art in coreference resolution for electronic medical records. *Journal of the American Medical Informatics Association*, 19(5):786–791.

Özlem Uzuner, Ira Goldstein, Yuan Luo, and Isaac S. Kohane. 2008. Viewpoint Paper: Identifying Patient Smoking Status from Medical Discharge Records. *Journal of the American Medical Informatics Association*, 15(1):14–24.

Özlem Uzuner, Yuan Luo, and Peter Szolovits. 2007. Viewpoint Paper: Evaluating the State-of-the-Art in Automatic De-identification. *Journal of the American Medical Informatics Association*, 14(5):550–563.

Özlem Uzuner, Imre Solti, and Eithon Cadag. 2010a. Extracting medication information from clinical text. *Journal of the American Medical Informatics Association*, 17(5):514–518.

Özlem Uzuner, Imre Solti, Fei Xia, and Eithon Cadag. 2010b. Community annotation experiment for ground truth generation for the i2b2 medication challenge. *Journal of the American Medical Informatics Association*, 17(5):519–523.

Özlem Uzuner, Brett R. South, Shuying Shen, and Scott L. DuVall. 2011. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP Models Know Numbers? Probing Numeracy in Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 5306–5314, Hong Kong, China. ACL.

Pengtao Xie, Haoran Shi, Ming Zhang, and Eric P. Xing. 2018. A Neural Architecture for Automated

ICD Coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Volume 1: Long Papers*, pages 1066–1076, Melbourne, Australia. ACL.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Zachariah Zhang, Jingshu Liu, and Narges Razavian. 2020. BERT-XML: large scale automated ICD coding using BERT pretraining. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop, ClinicalNLP@EMNLP 2020, Online, November 19, 2020*, pages 24–34. Association for Computational Linguistics.

A Distribution of Diagnosis and Procedure Labels

Figure 5 and Figure 6 show the distributions of labels in the diagnosis and procedure prediction training sets. Both distributions follow the power law with a long tail of rare codes.

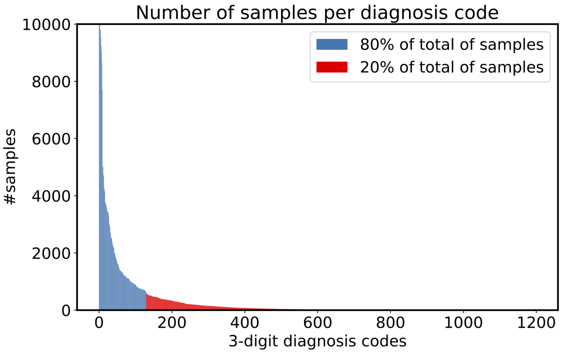


Figure 5: Distribution of ICD-9 diagnosis codes in MIMIC III training set.

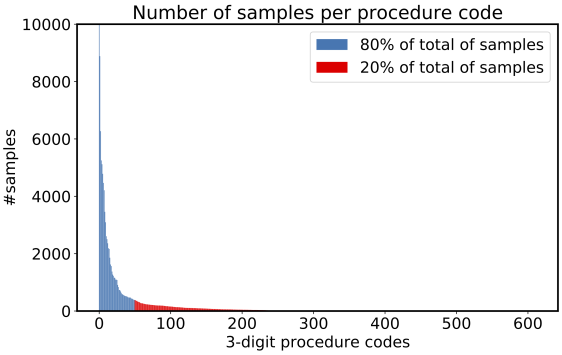


Figure 6: Distribution of ICD-9 procedure codes in MIMIC III training set.

B Pre-Processing Clinical Notes

B.1 Admission Notes From Discharge Summaries

We use MIMIC III discharge summaries that contain aggregated information about a patient such as doctor's assessments, relevant lab values, medications, and the patient's history. In order to filter the documents by admission sections, we first split all discharge summaries into sections with simple pattern matching. Together with clinical professionals, we then evaluated discharge summaries and identified sections that are known at admission time. We remove all other sections and thus hide information about the further hospital course and discharge of a patient. We exclude notes that do not contain any of the admission sections. We further apply a patient-wise split into train, validation and test set with a 70/10/20 ratio.

B.2 Converting i2b2 Data into Admission Discharge Task

The i2b2 De-identification and Heart Disease Risk Factors Challenge (Stubbs et al., 2015; Stubbs and Uzuner, 2015) introduced a dataset that contains clinical notes and discharge summaries annotated based on risk factors and disease indicators. We convert the data into an *admission to discharge* task by selecting five of the annotated conditions which correspond to ICD-9 codes as our labels, namely Hypertension (401), Hyperlipidemia (272), Coronary artery disease (414), Diabetes mellitus (250) and Obesity (278). Just like the MIMIC III diagnosis task, samples are annotated in a multi-label fashion. In order to convert the clinical notes to admission notes, we use the dataset from Rosenthal et al. (2019) that contain section labels per sentence. We then exclude sections that are not known at admission time concurrent to Section 3.2.

C Hyperparameter Setting

We use the following setting for pre-training and fine-tuning of the introduced Transformer-based models: We use early stopping and apply a random search for tuning the following hyperparameters on the validation set: learning rate [1e-4−1e-6], warmup steps [50−30k], dropout [0.1−0.3], class balancing [True/False] (fine-tuning only), gradient accumulation [1−200] with a batch size of 20.

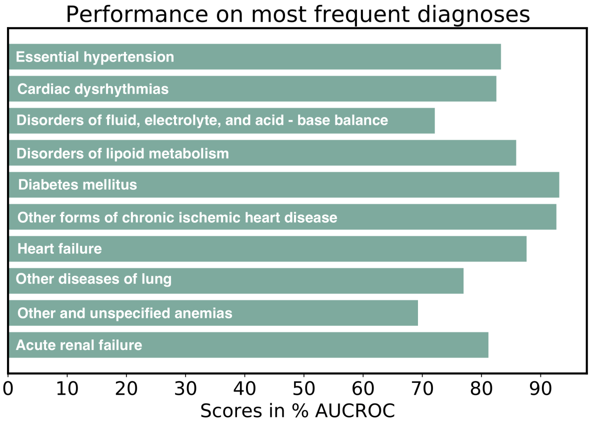


Figure 7: Top 10 diagnoses by frequency with the scores reached by the *CORe All* model.

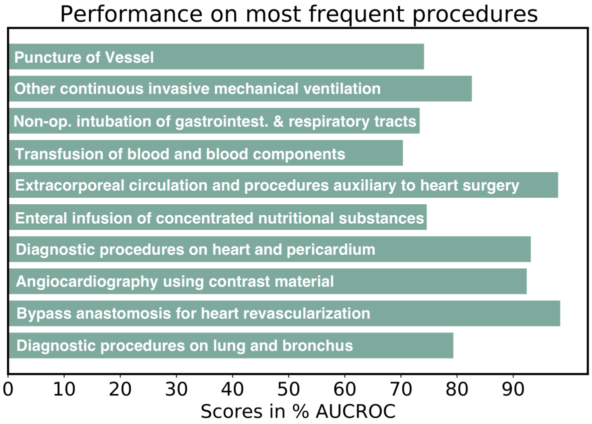


Figure 8: Top 10 procedures by frequency with the scores reached by the *CORe All* model.

D Results on Top 10 Diagnoses and Procedures

Figures 7 and 8 show the % AUROC scores of our *CORe All* model on the most frequent labels within the diagnosis and procedure prediction tasks. Figure 7 show that many chronic diseases such as *Essential Hypertension* or *Chronic ischemic heart disease* are among the most common within the MIMIC III dataset and present with relatively high AUROC values. We also observe that very specific codes such as *Diabetes mellitus* and *Bypass Anastomosis* are predicted more easily compared to more general codes such as *Other and unspecified anemias*.

Figure 8 further shows the negative influence of inconsistent labeling on standard procedures such as *Puncture of Vessel*.

Combining Deep Generative Models and Multi-lingual Pretraining for Semi-supervised Document Classification

Yi Zhu[♡] **Ehsan Shareghi**[♠♡] **Yingzhen Li**[◇*] **Roi Reichart**[♣] **Anna Korhonen**[♡]
[♡] Language Technology Lab, University of Cambridge
[♠] Department of Data Science & AI, Monash University
[◇] Department of Computing, Imperial College London
[♣] Faculty of Industrial Engineering and Management, Technion, IIT
`{yz568,alk23}@cam.ac.uk`, `ehsan.shareghi@monash.edu`
`yingzhen.li@imperial.ac.uk`, `roiri@technion.ac.il`

Abstract

Semi-supervised learning through deep generative models and multi-lingual pretraining techniques have orchestrated tremendous success across different areas of NLP. Nonetheless, their development has happened in isolation, while the combination of both could potentially be effective for tackling task-specific labelled data shortage. To bridge this gap, we combine semi-supervised deep generative models and multi-lingual pretraining to form a pipeline for document classification task. Compared to strong supervised learning baselines, our semi-supervised classification framework is highly competitive and outperforms the state-of-the-art counterparts in low-resource settings across several languages. [1]

1 Introduction

Multi-lingual pretraining has been shown to effectively use unlabelled data through learning shared representations across languages that can be transferred to downstream tasks (Artetxe and Schwenk, 2019; Devlin et al., 2019; Wu and Dredze, 2019; Conneau and Lample, 2019). Nonetheless, the lack of labelled data still leads to inferior performance of the same model compared to those trained in languages with more labelled data such as English (Zeman et al., 2018; Zhu et al., 2019).

Semi-supervised learning is another appealing paradigm that supplements the labelled data with unlabelled data which is easy to acquire (Blum and Mitchell, 1998; Zhou and Li, 2005; McClosky et al., 2006, *inter alia*). In particular, deep generative models (DGMs) such as variational autoencoder (VAE; Kingma and Welling (2014)) are capable of capturing complex data distributions at scale with rich latent representations, and they have been used for semi-supervised learning in various tasks in NLP (Xu et al., 2017; Yin et al., 2018; Choi et al., 2019; Xie and Ma, 2019), as well as inducing cross-lingual word embeddings (Wei and Deng, 2017), and representation learning in combination with Transformers via pretraining (Li et al., 2020).

To leverage the benefits of both worlds, we propose a pipeline method by combining semi-supervised DGMs (SDGMs) based on M1+M2 model (Kingma et al., 2014) with multi-lingual pretraining. The pretrained model serves as multi-lingual encoder, and SDGMs can operate on top of it independently of encoding architecture. To highlight such independence, we experiment with two pretraining settings: (1) our LSTM-based cross-lingual VAE, and (2) the current stat-of-the-art (SOTA) multi-lingual BERT (Devlin et al., 2019).

Our experiments on document classification in several languages show promising results via the SDGM framework with different encoders, outperforming the SOTA supervised counterparts. We also illustrate that the end-to-end training of M1+M2 that was previously considered too unstable to train (Maaløe et al., 2016) is possible with a reformulation of the objective function.

2 Semi-supervised Learning with DGMs

Variational Autoencoder. VAE consists of a stochastic neural encoder $q_\phi(\mathbf{z}|\mathbf{x})$ that maps an input $\mathbf{x}$ to a latent representation $\mathbf{z}$, and a neural decoder $p_\theta(\mathbf{x}|\mathbf{z})$ that reconstructs $\mathbf{x}$, jointly trained by maximising the evidence lower bound (ELBO) of the marginal likelihood of the data:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x}|\mathbf{z})\big] - \mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})\big) \qquad (1)$$

where the first term (reconstruction) maximises the expectation of data likelihood under the posterior distribution of $\mathbf{z}$, and the Kullback-Leibler (KL) divergence regulates the distance between the learned posterior and prior of $\mathbf{z}$.

---

[*] Work done while at Microsoft Research Cambridge.

[1] Code is available at `https://github.com/cambridgeltl/mling_sdgms`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 894–908
April 19 - 23, 2021.

$$\mathcal{L}(\mathbf{x},y) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z}_1)]}_{\text{Reconstruction}} - \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}[\log \frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}{p(\mathbf{z}_2)} + \log \frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}]}_{\text{KL}} + \underbrace{\log p(y)}_{\text{Constant}}$$

$$\mathcal{U}(\mathbf{x}) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z}_1)]}_{\text{Reconstruction}} - \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1)q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}[\log \frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}{p(\mathbf{z}_2)} + \log \frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)} + \log \frac{q_\phi(y|\mathbf{z}_1)}{p(y)}]}_{\text{KL}}$$

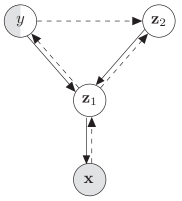


Table 1: Labelled and unlabelled objectives for M1+M2 model (left), and its corresponding graphical model (right).

Semi-supervised Learning with VAEs. The SDGM we use for semi-supervised learning is M1+M2 (Kingma et al., 2014), a graphical model (Table 1 (right)), with two layers of stochastic variables $\mathbf{z}_1$ and $\mathbf{z}_2$, with each being an isotropic Gaussian distribution. The first layer encodes the input sequence $\mathbf{x}$ into a deterministic hidden representation $\mathbf{h}$, and outputs the posterior distribution of $\mathbf{z}_1$:

$$q_\phi(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\Big(\boldsymbol{\mu}_\phi(\mathbf{h}), \mathrm{diag}\big(\boldsymbol{\sigma}^2_\phi(\mathbf{h})\big)\Big) \tag{2}$$

As our SDGM is independent of the encoding architecture, we use different pretrained multi-lingual models to obtain $\mathbf{h}$, $\boldsymbol{\mu}_\phi(\mathbf{h})$, and $\boldsymbol{\sigma}^2_\phi(\mathbf{h})$, described in §3. The second layer computes the posterior distribution of $\mathbf{z}_2$, conditioned on sampled $\mathbf{z}_1$ from $q_\phi(\mathbf{z}_1|\mathbf{x})$ and a class variable y.

When we use labelled data, i.e. y is observed, $q_\phi(\mathbf{z}_2|\mathbf{z}_1, y)$ can be directly obtained. With unlabelled data, we calculate the posterior $q_\phi(\mathbf{z}_2, y|\mathbf{z}_1) = q_\phi(y|\mathbf{z}_1)q_\phi(\mathbf{z}_2|\mathbf{z}_1, y)$ by inferring y with the classifier $q_\phi(y|\mathbf{z}_1)$, and integrate over all possible values of y. Therefore, the ELBO for the labelled data $\mathcal{S}_l = \{\mathbf{x}, y\}$ is $\mathcal{L}(\mathbf{x}, y)$:

$$\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)}\left[\log \frac{p_\theta(\mathbf{x}, y, \mathbf{z}_1, \mathbf{z}_2)}{q_\phi(\mathbf{z}_1, \mathbf{z}_2|\mathbf{x}, y)}\right] \leq \log p(\mathbf{x}, y)$$

and for the unlabelled data $\mathcal{S}_u = \{\mathbf{x}\}$ is $\mathcal{U}(\mathbf{x})$:

$$\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, y, \mathbf{z}_1, \mathbf{z}_2)}{q_\phi(\mathbf{z}_1, \mathbf{z}_2, y|\mathbf{x})}\right] \leq \log p(\mathbf{x})$$

where the generation part is $p_\theta(\mathbf{x}, y, \mathbf{z}_1, \mathbf{z}_2) = p(y)p(\mathbf{z}_2)p_\theta(\mathbf{z}_1|\mathbf{z}_2, y)p_\theta(\mathbf{x}|\mathbf{z}_1)$, $p(y)$ is uniform distribution as the prior of y, $p(\mathbf{z}_2)$ is standard Gaussian distribution as the prior of $\mathbf{z}_2$, and $p_\theta(\mathbf{x}|\mathbf{z}_1)$ is the decoder, which can have different architectures depending on the encoder (§4).

The objective function maximises both the labelled and unlabelled ELBOs while training directly the classifier with the labelled data as well:

$$\mathcal{J} = \sum_{(\mathbf{x},y)\in\mathcal{S}_l} \big(\mathcal{L}(\mathbf{x}, y) + \alpha\mathcal{J}_{cls}(\mathbf{x}, y)\big) + \sum_{\mathbf{x}\in\mathcal{S}_u} \mathcal{U}(\mathbf{x})$$

where $\mathcal{J}_{cls}(\mathbf{x}, y) = \mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[q_\phi(y|\mathbf{z}_1)]$, and α is a hyperparameter to tune. Considering the factorisation of the model according to the graphical model, we can rewrite the $\mathcal{L}(\mathbf{x}, y)$ and $\mathcal{U}(\mathbf{x})$ as shown in Table 1(left). The reconstruction term is the expected log likelihood of the input sequence $\mathbf{x}$, same for both ELBOs. The KL term regularises the posterior distributions of $\mathbf{z}_1$ and $\mathbf{z}_2$ according to their priors. Additionally for $\mathcal{U}(\mathbf{x})$, as mentioned before, we first infer y and treat it as if it were observed, so we need to compute the expected KL term over $q_\phi(y|\mathbf{z}_1)$ regularised by $\mathrm{KL}(q_\phi(y|\mathbf{z}_1)\|p(y))$.

Due to its training difficulty, M1+M2 is trained layer-wise in Kingma et al. (2014), where the first layer is trained according to Eq. 1 and fixed, before the second layer is trained on top. However, in our experiments (§4.1) we found that M1+M2 is easier to train end-to-end. We attribute this to our mathematical reformulation of the objective functions, giving rise to a more stable optimisation schedule.

3 SDGMs with Multi-lingual Pretraining

LSTM-based Encoder with VAE Pretraining. Our pretraining is based on the framework of Wei and Deng (2017), in which they pretrain a cross-lingual VAE with parallel corpus as input. However, the parallel corpus is expensive to obtain, and only the resulting cross-lingual embeddings rather than the whole encoder could be used due to the parallel input limitation of the model. To address these shortcomings, we propose non-parallel cross-lingual VAE (NXVAE), which has the same graphical model as the vanilla VAE. Each language i is associated with its own word embedding matrix, and its input sequence $\mathbf{x}_i$ is processed via a two layer BiLSTM (Hochreiter and Schmidhuber, 1997) shared across languages. We use the concatenation of the BiLSTM last hidden states as $\mathbf{h}$, and compute $q_\phi(\mathbf{z}|\mathbf{x}_i)$ with Eq. 2, so that $\mathbf{z}$ becomes the joint cross-lingual semantic space. A language specific bag-of-word decoder (BOW; Miao et al. (2016)) is then used to reconstruct the input sequence. Additionally, we optimise a language discriminator as an adversary (Lample et al., 2018a) to encourage the mixing of different language representations and keep the shared encoder language-agnostic. After pretraining NXVAE, we transfer the whole encoder, including $\boldsymbol{\mu}_\phi(\mathbf{h})$ and

$\boldsymbol{\sigma}^2_\phi(\mathbf{h})$, directly into our SDGM framework and treat it as $q_\phi(\mathbf{z}_1|\mathbf{x})$ component of the model (§4.1).

Multi-lingual BERT Encoder. To show that our SDGM is effective with other encoding architectures, we use the pretrained multi-lingual BERT (mBERT; Devlin et al. (2019))[2] as our encoder. Given an input sequence, the pooled `[CLS]` representation is used as $\mathbf{h}$ to compute $q_\phi(\mathbf{z}_1|\mathbf{x})$ (Eq. 2). Different from NXVAE, we initialise the parameters of $\boldsymbol{\mu}_\phi(\mathbf{h})$ and $\boldsymbol{\sigma}^2_\phi(\mathbf{h})$ randomly.

4 Experiments

We perform document classification on the class balanced multilingual document classification corpus (MLDoc; Schwenk and Li (2018)). Each document is assigned to one of the four news topic classes: *corporate/industrial* (C), *economics* (E), *government/social* (G), and *markets* (M). We experiment with five representative languages: EN, DE, FR, RU, ZH, and use 1k instance training set along with the standard development and test set. For experiments with varying labelled data size, the rest training data from 1k corpus is used as unlabelled data. The full statistics are shown in Table 2. Three languages (EN, DE, FR) are tested for LSTM encoder with VAE pretraining (§4.1) and all five languages for mBERT encoder (§4.2). All documents are lowercased. We report *accuracy* for evaluation following Schwenk and Li (2018).

For all experiments, We use Adam (Kingma and Ba, 2015) as optimiser, but with different learning rates for both settings and pretraining. We implemented the model with Pytorch[3] 1.10 (Paszke et al., 2019), and use GeForce GTX 1080Ti GPUs. See the Appendix for details about model configurations and training.

4.1 LSTM Encoder with VAE Pretraining

Experimental Setup. For pretraining NXVAE, we use three language pairs: EN-DE, EN-FR and DE-FR constructed from Europarl v7 parallel corpus (Koehn, 2005),[4] where only two language pairs are available: EN-DE and EN-FR, which consist of four datasets in total: (EN, DE)$_{\text{EN-DE}}$, and (EN, FR)$_{\text{EN-FR}}$. For DE-FR, we pair DE$_{\text{EN-DE}}$ and FR$_{\text{EN-FR}}$ directly as pseudo parallel data. We trim all datasets into exactly the same sentence size, and preprocess them with: tokenization, lowercasing, substituting digits with 0, and removing all punctuations, redundant spaces and empty lines. We randomly sample a small part of parallel sentences to build a development set. For models which do not require parallel input, e.g. NXVAE, we mix the two datasets of a language pair together. To avoid KL-collapse during pretraining, a weight α on the KL term in Eq. 1 is tuned and fixed to 0.1 (Higgins et al., 2017; Alemi et al., 2018). We only run one trial with fixed random seed for both pretraining and document classification. Training details can be found in the Appendix.

	C	E	G	M	Total
EN	270	234	252	244	1000
	228	238	266	268	1000
	991	1000	1030	979	4000
DE	270	240	245	245	1000
	229	268	266	237	1000
	984	1026	1022	968	4000
FR	227	262	258	253	1000
	257	237	237	269	1000
	999	973	998	1030	4000
RU	261	288	184	267	1000
	265	272	204	259	100
	1073	1121	706	1100	4000
ZH	294	286	109	311	1000
	324	300	93	283	1000
	1169	1215	363	1253	4000

Table 2: Statistics of MLDoc in five languages. Instance numbers for each class along with the total numbers are shown. For each language, three rows are training, development and test set instance numbers.

As our supervised baselines we compare with the following two groups: (I) NXVAE-based supervised models which are pretrained NXVAE encoder with a multi-layer perceptron classifier on top (denoted by NXVAE-z_1 ($q_\phi(y|\mathbf{z}_1)$) or NXVAE-h ($q_\phi(y|\mathbf{h})$) depending on the representation fed into the classifier; or NXVAE-z_1 models initialised with different pretrained embeddings: random initialisation (RAND), mono-lingual fastText (FT; Bojanowski et al. (2017)), unsupervised cross-lingual MUSE (Lample et al., 2018b), pretrained embeddings from Wei and Deng (2017) (PEMB), and our resulting embeddings from pretrained NXVAE (NXEMB).[5] (II) We also pretrain a word-based BERT (BERTW) with parameter size akin to NXVAE on the same data, and fine-tune it directly.[6]

For our semi-supervised experiments, we test

---

[2] `https://github.com/google-research/bert/blob/master/multilingual.md`.

[3] `https://pytorch.org/`.

[4] `https://www.statmt.org/europarl/`.

[5] All embeddings are pretrained on the same Europarl data.

[6] We also trained subword-based models for BERT and NXVAE, and observed similar trends. See the Appendix.

Word pair	Lang	kNNs ($k = 3$)
president (EN)	EN	mr, madam, gentlemen
	DE	präsident, herr, kommissar
präsident (DE)	EN	president, mr, madam
	DE	herr, kommissar, herren
great (EN)	EN	deal, with, a
	DE	große, eine, gute
groß (DE)	EN	striking, gets, lucrative
	DE	gering, heikel, hoch
said (EN)	EN	already, as, been
	DE	gesagt, mit, dem
sagte (DE)	EN	he, rightly, said
	DE	vorhin, kollege, kommissar

Table 3: Cosine similarity-based nearest neighbours of words (left column) in embedding spaces of EN and DE.

# Labels	32	64	128	1K	32	64	128	1K
EN-DE	EN				DE			
NXVAE-h	56.5	61.7	59.5	78.4	53.6	66.7	78.9	87.2
NXVAE-z_1	63.9	71.4	77.0	91.6	**65.0**	73.8	82.7	93.0
RAND	50.1	54.2	62.3	82.5	47.2	60.8	69.0	84.8
FT	36.3	49.4	61.1	80.9	45.0	54.3	69.2	86.1
MUSE	59.8	65.4	71.8	88.4	45.1	66.2	79.7	90.4
PEMB	36.4	53.9	50.9	84.4	39.4	52.0	69.0	86.7
NXEMB	61.5	62.0	68.6	85.4	53.4	71.2	75.9	88.8
BERTW	**67.7**	**72.7**	**84.6**	**91.8**	58.1	**77.5**	**89.2**	**94.0**
M1+M2	56.6	67.1	70.3	-	52.6	67.2	76.8	-
M1+M2+BOW	**79.8**	**81.7**	**87.2**	-	70.5	79.6	**89.7**	-
M1+M2+GRU	75.3	79.4	84.9	-	**75.1**	**80.0**	87.1	-
BERTW+ST	68.4	73.9	86.4	-	59.6	79.7	89.4	-
EN-FR	EN				FR			
NXVAE-h	**71.4**	73.8	78.6	88.0	62.8	72.7	79.9	88.9
NXVAE-z_1	71.2	**75.3**	**80.4**	**91.2**	**68.3**	**75.0**	**81.4**	**91.7**
M1+M2	71.8	73.5	76.5	-	66.2	78.7	79.7	-
M1+M2+BOW	**81.0**	**85.5**	**88.2**	-	80.3	**86.0**	**88.8**	-
M1+M2+GRU	75.3	81.4	83.8	-	**80.7**	82.3	87.4	-
DE-FR	DE				FR			
NXVAE-h	42.4	53.3	74.3	85.7	39.8	51.8	58.5	86.9
NXVAE-z_1	**63.3**	**75.4**	**81.3**	**92.1**	**60.1**	**71.1**	**78.4**	**91.4**
M1+M2	59.1	70.6	75.4	-	48.5	57.4	60.7	-
M1+M2+BOW	**78.0**	**83.2**	**88.3**	-	**81.4**	**84.5**	**88.4**	-
M1+M2+GRU	74.6	80.5	86.2	-	66.2	77.2	81.9	-

Table 4: MLDoc test accuracy for EN-DE, EN-FR and DE-FR pairs. The best results for supervised and semi-supervised models are in bold.

two types of decoders with different model capacities: BOW and GRU (Cho et al., 2014). We use M1+M2+BOW (GRU) to denote the model with joint training using a specific decoder, and M1+M2 to denote the original model in Kingma et al. (2014) with layer-wise training.[7] We also add a semi-supervised self-training method (McClosky et al., 2006) for BERTW to leverage the unlabelled data (BERTW+ST), where we iteratively add predicted unlabelled data when the model achieves a better dev. accuracy until convergence.

Qualitative Results. Table 3 illustrates the quality of the learned alignments in the cross-lingual space of NXVAE for EN-DE word pairs.

Classification Results. Table 4 (**EN-DE**) shows that within supervised models the NXVAE-z_1 substantially outperforms other supervised baselines with the exception of BERTW. The fact that NXVAE-z_1 is significantly better than NXVAE-h, suggests that pretraining has enabled $\mathbf{z}_1$ to learn more general knowledge transferable to this task. Combining with SDGMs, our best pipeline outperforms all baselines across data sizes and languages, including BERTW+ST with bigger gaps in fewer labelled data scenario. We observe the same trend of performance in both supervised and semi-supervised DGM settings on **EN-FR** and **DE-FR**.

For decoder, BOW outperforms the GRU, a finding in line with the results of Artetxe et al. (2019) which suggests a few keywords seem to suffice for this task. The poor performance of the original M1+M2, implies the domain discrepancy between pretraining and task data, and highlights the impact of fine-tuning. In addition, our NXEMB, as a byproduct of NXVAE, performs comparably well with MUSE, and better than all other embedding models including its parallel counterpart PEMB.

4.2 Multi-lingual BERT Encoder

Experimental Setup. We use the cased mBERT, a 12 layer Transformer (Vaswani et al., 2017) trained on Wikipedia of 104 languages with 100k shared WordPiece vocabulary. The training corpus is larger than Europarl by orders of magnitude, and high-resource languages account for most of the corpus. We use the best SDGM setup (M1+M2+BOW §4.1), on top of mBERT encoder against the mBERT supervised model with a linear layer as classifier (SUP-h) in 5 representative languages (EN, DE, FR, RU, ZH). We report the results over 5 runs due to the training instability of BERT (Dodge et al., 2020; Mosbach et al., 2020).

Classification Results. Figure 1 demonstrates that M1+M2+BOW outperforms the SOTA supervised mBERT (SUP-h) on average across all languages. This corroborates the effectiveness of our

---

[7] We also compared this against a more complex Skip Deep Generative Model (Maaløe et al., 2016), but found that end-to-end M1+M2 performs better. Details in the Appendix.

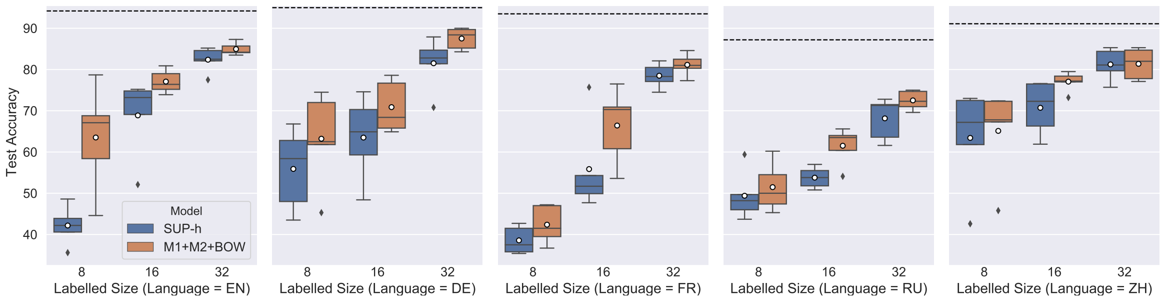


Figure 1: Boxplot of test accuracy scores for SUP-h and M1+M2+BOW over 5 runs. The mean is shown as white dot. The dashed line is the test mean accuracy of SUP-h trained on 1k labelled data of the corresponding language.

SDGM in leveraging unlabelled data within smaller labelled data regime, as well as its independence from encoding architecture.[8] As expected, the gap is generally larger with 8 and 16 labelled data, but reduces as the data size grows to 32. The variance shows similar pattern, but with relatively large values because of the instability of mBERT. Interestingly, the performance difference seems to be more notable in high-resource languages with more pretrained data, whereas in languages with fewer pretrained texts or vocabulary overlaps such as RU and ZH, the two models achieve closer results.

5 Conclusion

We bridged between multi-lingual pretraining and deep generative models to form a semi-supervised learning framework for document classification. While outperforming SOTA supervised models in several settings, we showed that the benefits of SDGMs are orthogonal to the encoding architecture or pretraining procedure. It opens up a new avenue for SDGMs in low-resource NLP by incorporating unlabelled data potentially from different domains and languages. Our preliminary results in cross-lingual zero-shot setting with SDGMs+NXVAE are promising, and we will continue the exploration in this direction as future work.

Acknowledgments

This work is supported by the ERC Consolidator Grant LEXICAL: Lexical Acquisition Across Languages (648909). The first author would like to thank Victor Prokhorov and Xiaoyu Shen for their comments on this work. The authors would like to thank the three anonymous reviewers for their helpful suggestions.

---

[8] Compared to smaller pretraining corpus (§4.1), we found that the representations pretrained on large corpus are less prune to overfit to the training instances of the task. We observe that training without the KL regularisation yields better performance for SDMGs+mBERT.

References

Alexander A. Alemi, Ben Poole, Ian Fischer, Joshua V. Dillon, Rif A. Saurous, and Kevin Murphy. 2018. Fixing a broken ELBO. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 159–168. PMLR.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *CoRR*, abs/1910.11856.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Trans. Assoc. Comput. Linguistics*, 7:597–610.

Avrim Blum and Tom M. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998*, pages 92–100. ACM.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Jihun Choi, Taeuk Kim, and Sang-goo Lee. 2019. A cross-sentence latent variable model for semi-supervised text sequence matching. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4747–4761. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *CoRR*, abs/2002.06305.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3581–3589.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018b. Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Chunyuan Li, Xiang Gao, Yuan Li, Xiujun Li, Baolin Peng, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. *CoRR*, abs/2004.04092.

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. 2016. Auxiliary deep generative models. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1445–1453. JMLR.org.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 4-9, 2006, New York, New York, USA*. The Association for Computational Linguistics.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1727–1736. JMLR.org.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. *CoRR*, abs/2006.04884.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037. Curran Associates, Inc.

Holger Schwenk and Xian Li. 2018. A corpus for multilingual document classification in eight languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Liang-Chen Wei and Zhi-Hong Deng. 2017. A variational autoencoding approach for inducing cross-lingual word embeddings. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 4165–4171. ijcai.org.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 833–844. Association for Computational Linguistics.

Zhongbin Xie and Shuai Ma. 2019. Dual-view variational autoencoders for semi-supervised text matching. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5306–5312. ijcai.org.

Weidi Xu, Haoze Sun, Chunting Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3358–3364. AAAI Press.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. Structvae: Tree-structured latent variable models for semi-supervised semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 754–765. Association for Computational Linguistics.

Daniel Zeman, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 1–21. Association for Computational Linguistics.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Trans. Knowl. Data Eng.*, 17(11):1529–1541.

Yi Zhu, Benjamin Heinzerling, Ivan Vulic, Michael Strube, Roi Reichart, and Anna Korhonen. 2019. On the importance of subword information for morphological tasks in truly low-resource languages. In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*, pages 216–226. Association for Computational Linguistics.

A Derivations of semi-supervised ELBOs

We derive the full ELBOs of both labelled and unlabelled data for M1+M2 and Auxiliary Skip Deep Generative Model (AUX; Maaløe et al. (2016)).[9] We first use $(\cdot)$ to represent different conditional variables for the two models so that the derivations can be unified, then we will realise it with the model-specific conditions in the end.

As written in the paper, the labelled ELBO for both models is:

$$\begin{aligned}&\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)}\left[\log \frac{p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)}{q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)}\right]\\&=\mathcal{L}(\mathbf{x},y)\leq \log p(\mathbf{x},y)\end{aligned}$$

Expanding the ELBO, we will have:

$$\begin{aligned}&\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)}[\log \frac{p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)}{q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)}]\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log p(\mathbf{z}_2)+\log p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)+\log p_\theta(\mathbf{x}|\cdot)+\log p(y)-\\
&\log q_\phi(\mathbf{z}_2|\cdot)-\log q_\phi(\mathbf{z}_1|\mathbf{x})]\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log q_\phi(\mathbf{z}_2|\cdot)+\log q_\phi(\mathbf{z}_1|\mathbf{x})-\\
&\log p(\mathbf{z}_2)-\log p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)-\log p(y)]\\
=&\mathbb{E}_{q_\phi(\mathbf{z1}|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log \frac{q_\phi(\mathbf{z}_2|\cdot)}{p(\mathbf{z}_2)}+\log \frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}-\log p(y)]\end{aligned}$$

After realising $(\cdot)$, we can then obtain the labelled ELBO for M1+M2 and AUX in the original paper:

$$\begin{aligned}&\mathcal{L}_{\text{M1+M2}}(\mathbf{x},y)\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z}_1)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}[\\
&\log \frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}{p(\mathbf{z}_2)}+\log \frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}-\log p(y)]\end{aligned}$$

$$\begin{aligned}&\mathcal{L}_{\text{AUX}}(\mathbf{x},y)\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}[\log p_\theta(\mathbf{x}|\mathbf{z}_1,\mathbf{z}_2,y)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}[\\
&\log \frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}{p(\mathbf{z}_2)}+\log \frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}-\log p(y)]\end{aligned}$$

---

[9] As mentioned in the footnote of original paper, we compare M1+M2 with AUX in LSTM encoder with VAE pretraining, but found that the simpler M1+M2 performs better. Results on AUX can be found in §D.

For the unlabelled ELBO, y is unobservable:

$$\begin{aligned}&\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})}\left[\log\frac{p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)}{q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})}\right]\\&=\mathcal{U}(\mathbf{x})\leq\log p(\mathbf{x})\end{aligned}$$

After expansion:

$$\begin{aligned}
&\mathbb{E}_{q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})}[\log\frac{p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)}{q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})}]\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log p(\mathbf{z}_2)+\log p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)+\log p_\theta(\mathbf{x}|\cdot)+\log p(y)-\\
&\log q_\phi(\mathbf{z}_2|\cdot)-\log q_\phi(\mathbf{z}_1|\mathbf{x})-\log q_\phi(y|\cdot)]\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log q_\phi(\mathbf{z}_2|\cdot)+\log q_\phi(\mathbf{z}_1|\mathbf{x})+\log q_\phi(y|\cdot)-\\
&\log p(\mathbf{z}_2)-\log p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)-\log p(y)]\\
=&\mathbb{E}_{q_\phi(\mathbf{z1}|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log\frac{q_\phi(\mathbf{z}_2|\cdot)}{p(\mathbf{z}_2)}+\log\frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}+\log\frac{q_\phi(y|\cdot)}{p(y)}]
\end{aligned}$$

Similarly, we will get unlabeled ELBO of M1+M2 and AUX:

$$\begin{aligned}
&\mathcal{U}_{\text{M1+M2}}(\mathbf{x})\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z}_1)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1)q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}[\\
&\log\frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)}{p(\mathbf{z}_2)}+\log\frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}+\log\frac{q_\phi(y|\mathbf{z}_1)}{p(y)}]
\end{aligned}$$

$$\begin{aligned}
&\mathcal{U}_{\text{AUX}}(\mathbf{x})\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1,\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}[\log p_\theta(\mathbf{x}|\mathbf{z}_1,\mathbf{z}_2,y)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1,\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}[\\
&\log\frac{q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)}{p(\mathbf{z}_2)}+\log\frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}+\log\frac{q_\phi(y|\mathbf{z}_1,\mathbf{x})}{p(y)}]
\end{aligned}$$

In our experiments, we sample $\mathbf{z}_1$ and $\mathbf{z}_2$ once during inference, so both labeled and unlabeled ELBOs can be approximated by:

$$\begin{aligned}
&\mathcal{L}(\mathbf{x},y)\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log\frac{q_\phi(\mathbf{z}_2|\cdot)}{p(\mathbf{z}_2)}+\log\frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}-\log p(y)]\\
\approx&\log p_\theta(\mathbf{x}|\cdot)+\log p(y)-\\
&\text{KL}(q_\phi(\mathbf{z}_2|\cdot)\|p(\mathbf{z}_2))-\text{KL}(q_\phi(\mathbf{z}_1|\mathbf{x})\|p_\theta(\mathbf{z}_1|\mathbf{z}_2,y))
\end{aligned}$$

$$\begin{aligned}
&\mathcal{U}(\mathbf{x})\\
=&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\log p_\theta(\mathbf{x}|\cdot)]-\\
&\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\cdot)q_\phi(\mathbf{z}_2|\cdot)}[\\
&\log\frac{q_\phi(\mathbf{z}_2|\cdot)}{p(\mathbf{z}_2)}+\log\frac{q_\phi(\mathbf{z}_1|\mathbf{x})}{p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)}+\log\frac{q_\phi(y|\cdot)}{p(y)}]\\
\approx&\log p_\theta(\mathbf{x}|\cdot)-\text{KL}(q_\phi(y|\cdot)\|p(y))-\\
&\mathbb{E}_{q_\phi(y|\cdot)}[\text{KL}(q_\phi(\mathbf{z}_2|\cdot)\|p(\mathbf{z}_2))]-\\
&\mathbb{E}_{q_\phi(y|\cdot)}[\text{KL}(q_\phi(\mathbf{z}_1|\mathbf{x})\|p_\theta(\mathbf{z}_1|\mathbf{z}_2,y))]
\end{aligned}$$

B Factorisation of M1+M2 and AUX

The two models have different factorisations, with M1+M2 being written as:

$$\begin{aligned}
q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)&=q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)\\
q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})&=q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1)q_\phi(\mathbf{z}_2|\mathbf{z}_1,y)\\
p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)&=p(y)p(\mathbf{z}_2)p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)p_\theta(\mathbf{x}|\mathbf{z}_1)\\
\mathcal{J}_{cls}(\mathbf{x},y)&=\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[q_\phi(y|\mathbf{z}_1)]
\end{aligned}$$

and AUX is factorised as follows:

$$\begin{aligned}
q_\phi(\mathbf{z}_1,\mathbf{z}_2|\mathbf{x},y)&=q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)\\
q_\phi(\mathbf{z}_1,\mathbf{z}_2,y|\mathbf{x})&=q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(y|\mathbf{z}_1,\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1,\mathbf{x},y)\\
p_\theta(\mathbf{x},y,\mathbf{z}_1,\mathbf{z}_2)&=p(y)p(\mathbf{z}_2)p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)p_\theta(\mathbf{x}|\mathbf{z}_1,\mathbf{z}_2,y)\\
\mathcal{J}_{cls}(\mathbf{x},y)&=\mathbb{E}_{q_\phi(\mathbf{z}_1|\mathbf{x})}[q_\phi(y|\mathbf{z}_1,\mathbf{x})]
\end{aligned}$$

where $q_\phi(\mathbf{z}_1|\mathbf{x})$, $q_\phi(\mathbf{z}_2|\cdot)$, and $p_\theta(\mathbf{z}_1|\mathbf{z}_2,y)$ are parameterised as diagonal Gaussians, and other distributions are defined as:

$$\begin{aligned}
q_\phi(y|\cdot)&=\text{Cat}(y|\pi_\phi(\cdot)) & p(y)&=\text{Cat}(y|\pi)\\
p(\mathbf{z}_2)&=\mathcal{N}(\mathbf{z}_2|\mathbf{0},\mathbf{I}) & p_\theta(\mathbf{x}|\cdot)&=f(\mathbf{x},\cdot;\theta)
\end{aligned}$$

where $\text{Cat}(\cdot)$ is a multinomial distribution and y is treated as latent variables if it is unobserved in unlabelled case. $f(\mathbf{x},\cdot;\theta)$ serves as the decoder and calculates the likelihood of the input sequence $\mathbf{x}$.

C Details on LSTM Encoder with VAE Pretraining

C.1 Data preprocessing and statistics

We use two pairs of data from Europarl v7 (Koehn, 2005):[10] EN-DE and EN-FR, which consist of four datasets in total: $\text{EN}_{\text{EN-DE}}$, $\text{DE}_{\text{EN-DE}}$, $\text{EN}_{\text{EN-FR}}$, and $\text{FR}_{\text{EN-FR}}$. Regarding DE-FR data, we take the datasets of $\text{DE}_{\text{EN-DE}}$ and $\text{FR}_{\text{EN-FR}}$.

For each language pair, the sentences in the same line of both datasets are a pair of parallel sentences. We do the following preprocessing to each dataset: tokenization; lower case; substitute digits with 0; remove all punctuations; remove redundant spaces and empty lines. Then we trim all four datasets into exactly the same sentence size. We randomly split a small part of parallel sentences to build a dev. set, which leads to 189m lines of training set and 13995 lines of dev. set for each language. Then we shuffle each dataset so that each language pair is not parallel anymore (for both train and dev. sets).

Our goal is to merge the two datasets of each pair and scramble them to form a single dataset. In practice, we keep each dataset separate, and sample

---

[10] `https://www.statmt.org/europarl/`.

a batch randomly from one language alternatively during pretraining, so that the data from both languages are mixed.

C.2 Model and training details

Instead of optimising the standard VAE, we optimise the following objective for NXVAE (Higgins et al., 2017; Alemi et al., 2018):

$$\mathcal{J}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \alpha \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})) \quad (3)$$

where we manually tune the fixed hyperparameter α on EN-DE data to reach a good balance between the reconstruction and the KL empirically. We select $\alpha = 0.1$ and apply it for the pretraining of other language pairs as well. The model and training details of XNVAE are shown in Table 5 (left).

C.3 Pretraining other models

For MLDoc supervised document classification, we also pretrain other baseline models to compare with ONLY for **EN-DE** pair:

Cross-lingual VAE with parallel input (PEMB; Wei and Deng (2017)): For the model of Wei and Deng (2017), we run the original code directly on the same EN-DE Europarl data without changing any of the model architecture. Since the model requires parallel input, we take the preprocessed and split EN-DE data. However, we do not shuffle each dataset, but rather feed them as parallel input to the model, so that the model and our corresponding NXVAE use the same amount and content of the data.

Subword-based non-parallel cross-lingual VAE SNXVAE: Instead of having separate vocabulary and decoders for each language, we use a single vocabulary and decoder for SNXVAE. We build the vocabulary with SentencePiece[11] of size 1e4. All other settings are the same as NXVAE. Its model and training details can be found in Table 5 (right).

Word and subword-based BERT model BERTW/BERTSW : For BERTW, we change the vocabulary and model size to be comparable with NXVAE. Note that the vocabulary size of BERTW is the same as the intersected vocabulary size of the two languages in NXVAE. We only use the masked language model objective during pretraining, and discard the objective of next sentence prediction.[12] For BERTSW, we use the same vocabulary as SNXVAE and set the model to similar parameter size as SNXVAE. The model and training details of BERTW and BERTSW are shown in Table 6.

D More Results on Document Classification

D.1 LSTM Encoder with VAE Pretraining

Supervised Learning. Our base model is NXVAE-z_1, which adds an MLP classifier $q_\phi(y|\mathbf{z}_1)$ on top of the encoder with the same architecture of the NXVAE. The similar applies to the subword-based models SNXVAE-z_1. NXVAE-h takes the deterministic $\mathbf{h}$ as the input to $q_\phi(y|\mathbf{x})$. All our baseline models with pretrained embeddings use the architecture of NXVAE-z_1. For fastText (FT), we train the embeddings of both languages with the same data of $\mathrm{EN}_{\mathrm{EN\text{-}DE}}$ and $\mathrm{DE}_{\mathrm{EN\text{-}DE}}$. For MUSE, we align on the pretrained FT embeddings. For BERTW and BERTSW, we use the library Transformers[13] for classification, and initialise the models with the corresponding pretrained parameters. All model and training details can be found in Table 7. The comparison results of word-based and subword-based models are shown in Table 8.

Semi-supervised learning with SDGMs. The main model (NXVAE) and training details are the same as in supervised learning. Besides M1+M2, we also compare with AUX (Maaløe et al., 2016) with the two decoder types. The training details are shown in Table 9. Regarding the decoding of GRU, all conditional latent variables of $p_\theta(\mathbf{x}|\cdot)$ are fed as extra input at each decoding step (Xu et al., 2017).

We tune all semi-supervised models on $\mathrm{EN}_{\mathrm{EN\text{-}DE}}$ with 32 labels in semi-supervised settings, and then apply it to all other languages and data sizes. We tune only one hyperparameter: the scaling factor β in the weight for the classification loss α in the original SDGM paper (Maaløe et al., 2016):

$$\alpha = \beta \frac{N_l + N_u}{N_l}$$

where N_l and N_u are labelled and unlabelled data point numbers. We tune β from $\{0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0, 20.0\}$. We pick the β with the best dev. performance for each

---

[11] `https://github.com/google/sentencepiece`.

[12] Both word and subword-based models are trained with: `https://github.com/google-research/bert`.

[13] `https://github.com/huggingface/transformers`.

model, and randomly select one when there is a tie. Then we use such fixed β for all other experiments across different training data sizes and languages.

The results of AUX can be seen in Table 10 along with M1+M2 results from the original paper. The parameter size of each model is shown in Table 11.

D.2 mBERT Encoder

The supervised model (SUP-h) adds a single linear transformation layer on the pooled `[CLS]` representation of mBERT, and M1+M2+BOW adds the corresponding SDGM framework on the same mBERT output. Like BERT, as mBERT uses a shared WordPiece vocabulary across languages, the parameter size of the same model will be the same for each language. All model and training details along with parameter size can be found in Table 12.

For tuning the hyperparameter of M1+M2+BOW, different from LSTM encoder with VAE pretraining, we set α fixed to $\alpha = \beta$. We tune β on EN with 8 labels in semi-supervised settings with 5 trials from $\{0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0, 20.0, 50.0\}$, pick the β with the best **average** dev. performance, and then apply it to all other languages and data sizes. We report the mean and variance over 5 trials, and the full results for both models can be seen in Table 13.

E Conditional document generation

Semi-supervised deep generative models can not only explore the complex data distributions, but are also equipped with the ability to generate documents conditioned on latent codes, which is another advantage over other semi-supervised models. We follow Kingma et al. (2014) by varying latent variable y for generation, and fixing $\mathbf{z}_2$ either sampled from the prior (Table 14) or obtained from the input through the inference model (Table 15), and generate sequence samples from the trained semi-supervised models M1+M2+BOW and M1+M2+GRU.[14]

Overall, all models generate words or utterances directly related to the class, with the class labels among top nouns generated by BOW models, and subjects/objects in sentences from GRU are also pertaining to corresponding classes. However, we also observe that the utterances in GRU are not fluent with many repetitions. We argue that it is caused by the high proportion of UNK in the training corpus that makes the sequence generation harder, supported by the fact that the most probable word in all BOW decoders is always UNK.

---

[14] All models are treined on $\text{EN}_{\text{en-fr}}$ with 128 labelled data.

Hyperparameter	NXVAE	SNXVAE
vocabulary size	4e4 (EN), 5e4 (DE, FR)	1e4
embedding size	300	300
embedding dropout	0.2	0.2
encoder	BiLSTM	BiLSTM
encoder input dimension	300	300
encoder hidden dimension	600 for each direction	600 for each direction
encoder layer number	2	2
encoder dropout	0.2	0.2
discriminator configuration	[2 × 600, 1024, leakyrelu, 1024, 1]	[2 × 600, 1024, leakyrelu, 1024, 1]
inferer ($\mathbf{h}$ to μ or $\log \sigma$) configuration	[2 × 600, 300, batchNorm, relu, 300]	[2 × 600, 300, batchNorm, relu, 300]
$\mathbf{z}$ dimension	300	300
parameter size	41.8M (EN-DE and EN-FR)/ 44.9M (DE-FR)	17.8M
running time	$\sim$ 1 day	less than 1 day
tie embeddings of encoder and decoder	True	True
sentence length threshold	median length of training data	median length of training data
α in Equation 3	{0.01, 0.02, 0.05, **0.1**, 0.2, 0.5, 1.0}	0.1
training epoch	500	500
early stopping	5 epochs on dev. negative likelihood	5 epochs
batch size	128	128
validate on dev.	every 4000 iterations	every 4000 iterations
optimiser	Adam	Adam
learning rate	5e-4	5e-4

Table 5: Model and training details of NXVAE.

Hyperparameter	BERTW	BERTSW
vocabulary size	84101	10005
hidden size	300	300
max position embeddings	512	512
hidden dropout prob	0.1	0.1
hidden activation	gelu	gelu
intermediate size	2100	1800
num attention heads	12	12
attention probs dropout prob	0.1	0.1
num hidden layers	12	11
parameter size	45.0M	19.1M
running time	$\sim$ 5 days	$\sim$ 3 days
max seq length	200	
max predictions per seq	30	
masked lm prob	0.15	
batch size	32	
optimiser	Adam	
learning rate	1e-4	
weight decay	0.01	
num train steps	1e6	

Table 6: Model and training details of BERTW and BERTSW.

Hyperparameter	BERTW/BERTSW	VAE-based
vocabulary	same as pretrained model	same as pretrained model
training epoch	5000	5000
early stopping	1000 epochs on dev. accuracy	1000
batch size	16	16
running time	$\sim$5.5h	$\sim$2.5h
sentence length	200	200
optimiser	Adam	Adam
learning rate	2e-5	5e-4
classifier	[input_dim, class_num]	[input_dim, 1024, leakyrelu, 1024, class_num]

Table 7: LSTM encoder with VAE pretraining: model and training details of MLDoc supervised document classification. The running time is calculated on $\text{EN}_{\text{EN-DE}}$ with 32 labelled data for all models.

EN-DE	EN				DE			
	32	64	128	FULL	32	64	128	FULL
BERTW	67.7	72.7	**84.6**	**91.8**	58.1	77.5	**89.2**	94.0
BERTSW	54.4	69.0	83.0	91.4	62.2	80.1	84.3	**94.1**
NXVAE-z_1	63.9	71.4	77.0	91.6	65.0	73.8	82.7	93.0
SNXVAE-z_1	68.9	**76.5**	79.2	90.3	69.0	79.4	85.5	91.7

Table 8: LSTM encoder with VAE pretraining: comparisons of word-based models and subword-based models for BERT and NXVAE in MLDoc supervised document classification. Word-based results are directly from the original paper.

Hyperparameter	M1+M2	M1+M2+BOW	M1+M2+GRU	AUX+BOW	AUX+GRU
training epoch	5000	5000	5000	5000	5000
early stopping	1000	1000	1000	1000	1000
best β	0.1	0.2	10.0	20.0	5.0
$\mathbf{z}_1$ dim	300	300	300	300	300
$\mathbf{z}_2$ dim	300	300	300	300	300
tie embedding	-	False	False	False	False
running time	~2h	~12h	~14h	~13h	~ 14.5h
GRU input dim	-	-	100	-	100
GRU hidden dim	-	-	50	-	50
GRU layers	-	-	1	-	1
GRU dropout prob	-	-	0.5	-	0.5

Table 9: LSTM encoder with VAE pretraining: model and training details of MLDoc semi-supervised document classification. The running time is calculated on EN$_{\text{EN-DE}}$ with 32 labelled data for all models.

EN-DE	EN				DE			
	32	64	128	FULL	32	64	128	FULL
M1+M2	56.6	67.1	70.3	-	52.6	67.2	76.8	-
M1+M2+BOW	**79.8**	**81.7**	87.2	-	70.5	79.6	**89.7**	-
M1+M2+GRU	75.3	79.4	84.9	-	75.1	80.0	87.1	-
AUX+BOW	78.8	**81.7**	**87.7**	-	**75.2**	**86.2**	89.3	-
AUX+GRU	74.8	80.0	85.1	-	72.2	76.5	87.6	-
EN-FR	EN				FR			
	32	64	128	FULL	32	64	128	FULL
M1+M2	71.8	73.5	76.5	-	66.2	78.7	79.7	-
M1+M2+BOW	**81.0**	**85.5**	**88.2**	-	80.3	**86.0**	**88.8**	-
M1+M2+GRU	75.3	81.4	83.8	-	**80.7**	82.3	87.4	-
AUX+BOW	79.8	83.4	87.1	-	80.4	85.7	88.1	-
AUX+GRU	78.3	81.3	86.6	-	**80.7**	83.2	85.4	-
DE-FR	DE				FR			
	32	64	128	FULL	32	64	128	FULL
M1+M2	59.1	70.6	75.4	-	48.5	57.4	60.7	-
M1+M2+BOW	**78.0**	**83.2**	88.3	-	**81.4**	**84.5**	**88.4**	-
M1+M2+GRU	74.6	80.5	86.2	-	66.2	77.2	81.9	-
AUX+BOW	74.6	82.9	**89.0**	-	73.9	79.5	82.1	-
AUX+GRU	70.7	79.5	80.3	-	67.3	81.0	83.6	-

Table 10: LSTM encoder with VAE pretraining: test accuracy of AUX models. The header numbers denote number of labelled training data instances. The best results are in bold. Other results related to M1+M2 are directly from the original paper.

	EN	DE	FR
EMBEDDING MODELS	25.8M	28.8M	28.8M
NXVAE-h	26.8M	29.8M	29.8M
NXVAE-z_1	25.8M	28.8M	28.8M
SNXVZE-z_1	16.8M	16.8M	16.8M
BERTW	45.0M	45.0M	45.0M
BERTSW	19.1M	19.1M	19.1M
M1+M2	0.9M	0.9M	0.9M
M1+M2+BOW	38.5M	44.5M	44.5M
M1+M2+GRU	43.2M	49.2M	49.2M
AUX+BOW	43.8M	49.8M	49.8M
AUX+GRU	48.5M	54.5M	54.5M

Table 11: LSTM encoder with VAE pretraining: parameter size of all supervised and semi-supervised models. The difference between NXVAE-based models and BERTW is caused by language specific vocabulary of NXVAE, where only one vocabulary is used for **mono-lingual** document classification.

Hyperparameter	SUP-h	M1+M2+BOW
vocabulary size	1e5	1e5
$\mathbf{z}_1$ dim	768	768
$\mathbf{z}_2$ dim	768	768
tie embedding	True	True
best β	-	10.0
training epoch	500	500
early stopping	100 epochs on dev. accuracy	100
batch size	4	4
running time	∼1h	∼5h
sentence length	200	200
optimiser	Adam	Adam
learning rate	2e-5	2e-5
classifier	[768, class_num]	[768, class_num]
parameter size	178M	185M

Table 12: mBERT encoder: model and training details of MLDoc document classification. The running time is calculated on EN$_{\text{EN-DE}}$ with 8 labelled data for both models.

	Model	8	16	32	1K
EN	SUP-h	42.2 (4.7)	68.9 (9.7)	82.4 (3.0)	94.2 (0.8)
	M1+M2+BOW	**63.5** (12.8)	**77.1** (2.8)	**85.0** (1.5)	-
DE	SUP-h	55.9 (9.9)	63.5 (10.2)	81.5 (6.5)	95.0 (0.3)
	M1+M2+BOW	**63.2** (11.5)	**70.9** (6.3)	**87.5** (2.6)	-
FR	SUP-h	38.6 (3.3)	55.9 (11.4)	78.5 (3.0)	93.5 (0.7)
	M1+M2+BOW	**42.4** (4.6)	**66.4** (9.1)	**81.1** (2.7)	-
RU	SUP-h	49.4 (6.0)	53.8 (2.6)	68.2 (5.2)	87.2 (0.4)
	M1+M2+BOW	**51.5** (6.0)	**61.5** (4.6)	**72.6** (2.3)	-
ZH	SUP-h	63.4 (12.5)	70.7 (6.5)	81.2 (3.9)	91.1 (0.1)
	M1+M2+BOW	**65.1** (11.1)	**77.1** (2.4)	**81.4** (3.8)	-

Table 13: mBERT Encoder: MLDoc average test accuracy for both SUP-h and M1+M2+BOW models. The variance is in the bracket after the mean score. The first row denotes the number of labelled instances. The best results are in bold.

Class	**M1+M2+BOW**	**M1+M2+GRU**
C	1: UNK, industry, credibility, agreement, ticket, co, decision, concept, ltd, people, sale, government, market, president, designations, minister, firm, plans, partner, deal	1: the bank said it lump of the united ... the new girls ltd said the concept ... the new extraordinary and the concept ... said the statement ...
	2: UNK, ticket, year, shares, days, results, age, net, demand, securities, period, stock, concept, construction, bank, programme, procedure, statement, value, commission	2: the bank of organisation said on thursday that it had revoked by the first girls ... first year to ...
E	1: UNK, finance, market, loophole, budget, surprise, bank, basis, issue, government, system, exchanges, committee, municipal, world, securities, holding, net, confidence, minister	1: the international basic fund said on acknowledged that it said on publish to vote on publish to a bank said on publish ...
	2: UNK, ticket, city, escalation, finance, bank, budget, concept, revenue, net, price, sale, trade, tax, prices, markets, series, rate, fund, pack	2: the bank of submitting on publish florence said on acknowledged that ... it said on publish that ... to the new coherent said on acknowledged to bumping the bank said the bank ...
G	1: UNK, government, state, minister, delay, pension, work, president, plans, summit, ticket, people, procedure, conference, ambassador, country, talks, opposition, nations, house	1: the president remarkable said on thursday it surprise of ethnocide arrival the infidels of the islamic of the waterway the bank was ...
	2: UNK, state, president, war, police, office, authorities, problem, information, result, country, rights, committee, city, people, biodiversity, justice, health, securities, issue	2: the summit in the authors and a virtual geological and the first time of the first party of the first time of ...
M	1: UNK, ticket, phase, market, government, minister, markets, banks, bank, budget, floor, points, rate, traders, procedure, strength, economy, finance, prices, loophole	1: the database distinctions the market closed sharply entire on thursday on acknowledged ...
	2: UNK, markets, market, stock, loophole, points, trade, shares, ticket, corporate, speaker, issues, fund, bank, group, exchanges, results, anticipation, companies, surprise	2: the following of the the the ries and not have embargo costs unveiling on publish pleading a impact of the japanese ... market and a bank was to be of the bank ...

Table 14: Generated samples from M1+M2+GRU (BOW) for class C (*Corporate/Industrial*), E (*Economics*), G (*Government/Social*), and M (*Markets*). We randomly sample $\mathbf{z}_2$ from the prior while varying y.

1: Fiat shares lost nearly two percent on Wednesday, slipping below the psychologically important 4,000 lire level in thin trading on a generally easier Milan Bourse, traders said. "The stock has gradually lost ground but without any major sell orders. At the moment there just isn't any interest in Fiat," one trader said. At 1439 GMT, Fiat was quoted 1.99 percent off at 3,980 lire, after touching a day's low of 3,970 lire, in volume of just under four million shares. The all-share Mibtel index posted a 0.47 percent fall. – Milan newsroom +392 66129589 (E)

1: fiat shares lost nearly two percent on UNK slipping below the psychologically important UNK lire level in thin trading on a generally easier milan UNK traders UNK UNK stock has gradually lost ground but without any major sell UNK at the moment there just UNK any interest in UNK one trader UNK at UNK UNK fiat was quoted UNK percent off at UNK UNK after touching a UNK low of UNK UNK in volume of just under four million UNK the UNK UNK index posted a UNK percent UNK UNK milan UNK UNK UNK

2: The top prosecutor of Honduras said on Wednesday that his country is a haven for money laundering. "In Honduras it's easy to launder money, the system allows it," Edmundo Orellana told reporters. "It's permitted because there is no law in Honduras that obligates a Honduran to explain the origin of his wealth." Honduran authorities estimate that $300 million in illegal drug profits is laundered through the country each year. Money laundering is not classified as an offence in Honduras, although legislators have been working on a bill to outlaw it since last year. (G)

2: the top prosecutor of honduras said on wednesday that his country is a haven for money UNK UNK honduras UNK easy to launder UNK the system allows UNK UNK UNK told UNK UNK permitted because there is no law in honduras that UNK a honduran to explain the origin of his UNK honduran authorities estimate that UNK million in illegal drug profits is laundered through the country each UNK money laundering is not classified as an offence in UNK although legislators have been working on a bill to outlaw it since last UNK

Class	M1+M2+BOW	M1+M2+GRU
C	1: UNK, ticket, profit, concept, net, market, escalation, share, results, shares, delay, group, revision, profits, period, misery, statement, bank, key, procedure 2: UNK, concept, ticket, group, market, shares, delay, president, stock, companies, bank, statement, government, stake, price, co, state, girls, meeting, ltd	1: the bank said on fourthly it has inject requirement of the first group of ... 2: the bank of organisation said on acknowledged that it had a meeting ...
E	1: UNK, ticket, escalation, inflation, key, revision, delay, period, floor, consumer, bank, contexts, result, instance, show, market, level, government, gross, price 2: UNK, ticket, bank, government, finance, market, state, budget, tax, minister, rate, delay, debt, issue, trade, investment, surprise, policy, sale, procedure	1: the bank of submitting on publish florence said on acknowledged that it said on publish that ... the new coherent ... to the bank ... 2: the international basic fund said on acknowledged that it said on publish ... to vote on acknowledged to a bank ...
G	1: UNK, world, ticket, policies, time, surprise, procedure, demand, campaigns, group, team, president, match, communities, place, minister, bank, government, number, relief 2: UNK, president, government, people, state, minister, pension, police, designations, meeting, talks, opposition, leaders, country, security, result, statement, authorities, peace, summit	1: the ana police said acknowledged it had a tackling ... 2: the president remarkable said on thursday that it surprise of ethnocide arrival infidels of her wines of her recall and the white house of ...
M	1: UNK, shares, ticket, contexts, touch, market, stock, points, escalation, share, traders, phase, immigrants, procedure, price, pledges, revision, agriculture, group , level 2: UNK, market, ticket, bank, traders, anticipation, delay, procedure, trade, prices, immigrants, rate, government, money, meda, escalation, demands, exchange, points, reallocation	1: the bank of the settlement following the following vocational meda of the deal was delay ... and the market ... 2: the bank of the settlement following the following vocational value of the relative gains of ...

Table 15: Generated samples from M1+M2+GRU (BOW) by varying class label y. We take $\mathbf{z}_2$ from the input examples shown above. For each example, the first is the original document with the class label in the end, and the second is the real input to the system.

Multi-facet Universal Schema

Rohan Paul[*] **Haw-Shiuan Chang**[*] **Andrew McCallum**
CICS, University of Massachusetts Amherst
`{rohpaul,hschang,mccallum}@cs.umass.edu`

Abstract

Universal schema (USchema) assumes that two sentence patterns that share the same entity pairs are similar to each other. This assumption is widely adopted for solving various types of relation extraction (RE) tasks. Nevertheless, each sentence pattern could contain multiple facets, and not every facet is similar to all the facets of another sentence pattern co-occurring with the same entity pair. To address the violation of the USchema assumption, we propose multi-facet universal schema that uses a neural model to represent each sentence pattern as multiple facet embeddings and encourage one of these facet embeddings to be close to that of another sentence pattern if they co-occur with the same entity pair. In our experiments, we demonstrate that multi-facet embeddings significantly outperform their single-facet embedding counterpart, compositional universal schema (CUSchema) (Verga et al., 2016), in distantly supervised relation extraction tasks. Moreover, we can also use multiple embeddings to detect the entailment relation between two sentence patterns when no manual label is available.

1 Introduction

Relation extraction (RE) is a crucial step in automatic knowledge base construction (AKBC). A major challenge of RE is that the frequency of relations in the real world is a long-tail distribution but collecting sufficient human annotations for every relation is infeasible (Han et al., 2020).

Distant supervision is proposed to alleviate the issue (Mintz et al., 2009). Distant supervision assumes that a sentence pattern expresses a relation if the sentence pattern co-occurs with an entity pair and the entity pair has the relation. For example, we assume the sentence pattern *"$ARG1, the partner of fellow $ARG2"* is likely to express the spouse relation if we observe a text clip *"... Angelina Jolie, the partner of fellow Brad Pitt ..."* in our training corpus and a knowledge base tells us that *Angelina Jolie* and *Brad Pitt* has the spouse relation. Accordingly, we can infer that another entity pair is likely to have the spouse relation if we observe the text *", the partner of fellow"* between them in a new corpus.

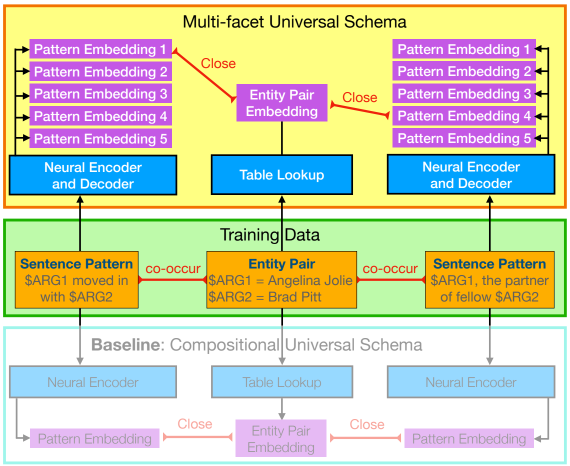


Figure 1: Comparison between the multi-facet and compositional universal schema. In our training loss, we encourage one of the facet embeddings from a sentence pattern to be similar to its co-occurred entity pair.

Universal schema (Riedel et al., 2013) extends this assumption by treating every sentence pattern as a relation, which means we assume that sentence patterns or relations in a knowledge base are similar if they co-occur with the same entity pair. For example, we assume *"$ARG1, the partner of fellow $ARG2"* and *"$ARG1, the wife of fellow $ARG2"* are similar if they both co-occur with *(Kristen Bell, Dax Shepard)*. Consequently, we can infer that *"$ARG1, the wife of fellow $ARG2"* also implies spouse relation as *"$ARG1, the partner of fellow $ARG2"* even if the knowledge base does not record the spouse relation between *Kristen*

---

[*] indicates equal contribution

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 909–919
April 19 - 23, 2021.

Bell and *Dax Shepard*.

Compositional universal schema (Verga et al., 2016) realizes the idea by using a LSTM (Hochreiter and Schmidhuber, 1997) to encode each sentence pattern into an embedding and encouraging the embedding to be similar to the embedding of the co-occurred entity pair. As in the lower part of Figure 1, the model makes the embeddings of two sentence patterns similar if they co-occur with the same entity pair. Baldini Soares et al. (2019) rely on a similar assumption and achieve state-of-the-art results on supervised RE tasks by replacing the LSTM with a large pre-trained language model.

The variants of universal schema have many different applications, including multilingual RE (Verga et al., 2016), knowledge base construction (Toutanova et al., 2015; Verga et al., 2017), question answering (Das et al., 2017), document-level RE (Verga et al., 2018), N-ary RE (Akimoto et al., 2019), open information extraction (Zhang et al., 2019), and unsupervised relation discovery (Percha and Altman, 2018).

Nevertheless, one sentence pattern could contain multiple facets, and each facet could imply a different relation. In Figure 1, *"\$ARG1, the partner of fellow \$ARG2"* could imply the entity pair has the spouse relation, the co-worker relation, or both. *"\$ARG1 moved in with \$ARG2"* could imply the spouse relation, the parent relation, ..., etc. If we squeeze the facets of a sentence pattern into a single embedding, the embedding is more likely to be affected by the irrelevant facets from other patterns co-occurred with the same entity pair (e.g., *"\$ARG1 moved in with \$ARG2"* might incorrectly imply the co-worker relation).

Another limitation is that single embedding representation can only provide symmetric similarity measurement between two sentence patterns. Thus, an open research challenge is to predict the entailment direction of two sentence patterns only based on their co-occurring entity pair information.

To overcome the challenges, we propose multi-facet universal schema, where we assume that two sentence patterns share a similar facet if they co-occur with the same entity pair. As in Figure 1, we use a neural encoder and decoder to predict multiple facet embeddings of each sentence pattern and encourage one of the facet embeddings to be similar to the entity pair embedding. As a result, the facets that are irrelevant to the relation between the entity pairs are less likely to affect the embeddings of entity pairs and other related sentence patterns. For example, the parent facet of *"\$ARG1 moved in with \$ARG2"* could be excluded when updating the embeddings of *(Angelina Jolie, Brad Pitt)*.

In our experiments, we first compare the multi-facet embeddings with the single-facet embedding in distantly supervised RE tasks. The results demonstrate that multiple facet embeddings significantly improve the similarity measurement between the sentence patterns and knowledge base relations. Besides RE, we also apply multi-facet embeddings to unsupervised entailment detection tasks. In a newly collected dataset, we show that multi-facet universal schema significantly outperforms the other unsupervised baselines.

2 Methods

Our method is illustrated in Figure 2. In Section 2.1, we first provide our problem setup: We are given a knowledge base (KB) and a text corpus during training. Our goal is to extract relations by measuring the similarity between KB relations and an (unseen) sentence pattern or to detect entailment between two sentence patterns. In Section 2.2, we introduce our neural model, which predicts multi-facet embeddings of each sentence pattern. Next, in Section 2.3, we describe our objective function, which encourages the embeddings of co-occurred entity pairs to be close to the embeddings of their closest pattern facets. Finally, in Section 2.4, we explain that multi-facet embeddings could be viewed as the cluster centers of possibly co-occurred entity pairs, and in Section 2.5, we provide our scoring functions for distantly supervised RE and unsupervised entailment tasks.

2.1 Background and Problem Setup

Our RE problem setup is the same as compositional universal schema (Verga et al., 2016). First, we run named entity recognition (NER) and entity linking on a raw corpus. After identifying the entity pairs in each sentence, we prepare a co-occurrence matrix as in Figure 2. Similarly, we represent the KB relations between entity pairs as a co-occurrence matrix and merge the matrices from the KB and the training corpus. The merged matrix has $y_{i,j} = 1$ if the ith sentence pair or KB relation co-occurs with the jth entity pair and $y_{i,j} = 0$ otherwise.

During testing, we use NER to extract an entity pair and the sentence pattern, which might not have been seen in the training corpus. Next, we extract

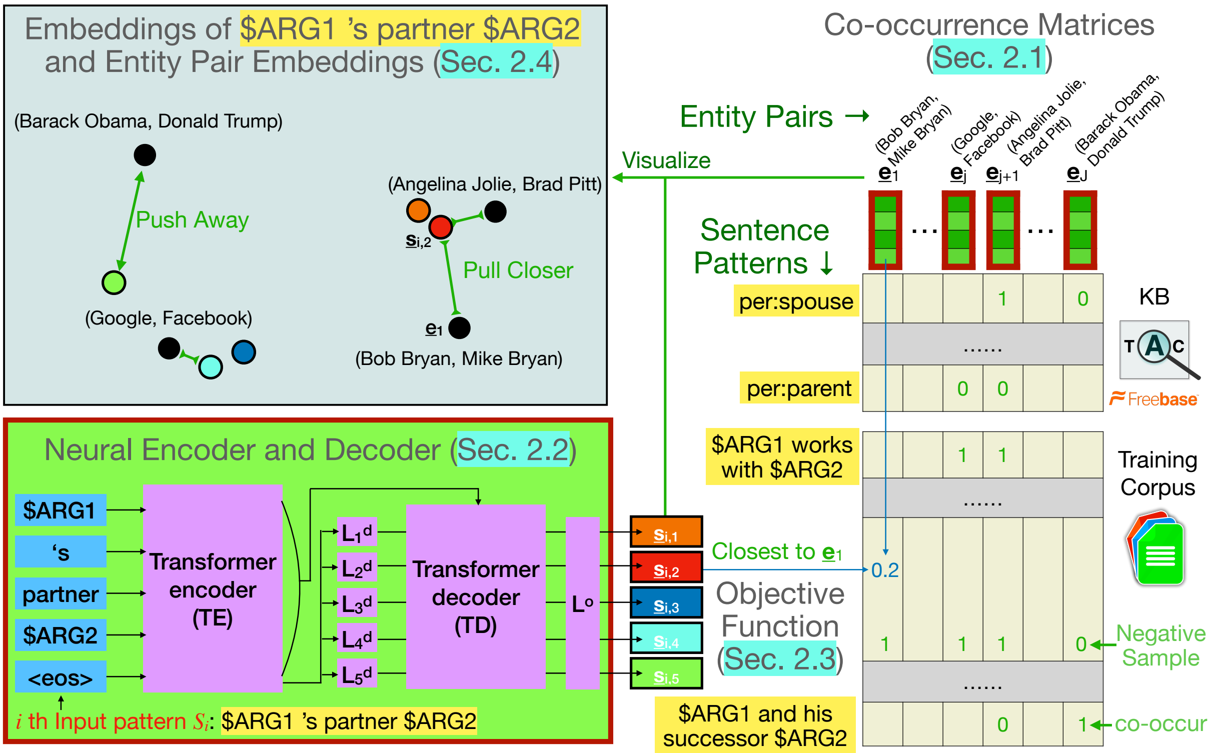


Figure 2: An illustration of the proposed method. The training signal comes from the co-occurrence matrices of the KB and training text corpus on the right. On the lower left, we visualize our neural encoder, which captures the compositional meaning of tokens in the sentence pattern, and our neural decoder, which models the dependency among multiple facet embeddings. When a sentence pattern co-occurs with an entity pair, the training loss minimizes the distance between the entity pair embedding and the closest facet embedding of the sentence pattern (e.g., 0.2 between $s_{i,2}$ and e_1). Trainable parameters in our model are highlighted using red borders. On the upper left, we visualize the embedding space to establish the connection between our method and clustering.

relations by computing the similarity between the sentence pattern embeddings and the embeddings of the applicable KB relations. Besides RE, we also detect the entailment between two sentence patterns by comparing their embeddings.

2.2 Neural Encoder and Decoder

We use a neural model to predict K facet embeddings of each sentence pattern. The goal is similar to Chang et al. (2021), which predict a fixed number of embeddings of a sentence, so we adopt their neural model as shown in Figure 2.

For the ith sentence pattern S_i, we append an <eos> to its end and use a 3-layer Transformer (Vaswani et al., 2017) encoder TE to model the compositional meaning of the input word sequence: $\underline{\boldsymbol{u}}_{i,1}...\underline{\boldsymbol{u}}_{i,|S_i|}\underline{\boldsymbol{u}}_{i,<\text{eos}>} = TE(S_i\text{<eos>})$, where $\underline{\boldsymbol{u}}_{i,l}$ is an embedding contextualized by the encoder. In the experiment, we also replace the Transformer with a bidirectional LSTM (bi-LSTM) to show that the improvement of multi-facet embeddings is independent of the encoder choice.

The embedding $\underline{\boldsymbol{u}}_{i,<\text{eos}>}$ represents the whole sentence pattern; we use K different linear layers L^d_k to transform the embedding into the inputs of our decoder: $\underline{\boldsymbol{b}}_{i,k} = L^d_k(\underline{\boldsymbol{u}}_{i,<\text{eos}>})$.

The facets in a sentence pattern often have some dependency. For example, the patterns that express the partnership between two people might also express the collaboration relation between two companies. To leverage the dependency, we use another 3-layer Transformer as our decoder TD. Besides the self-attention, we allow the hidden states in the decoder to query the contextualized word embeddings $\underline{\boldsymbol{u}}_{i,l}$ from the encoder (Vaswani et al., 2017) and output the embeddings corresponding to the different facets $\underline{\boldsymbol{d}}_{i,k}$: $\underline{\boldsymbol{d}}_{i,1}...\underline{\boldsymbol{d}}_{i,K} = TD(\underline{\boldsymbol{b}}_{i,1}...,\underline{\boldsymbol{b}}_{i,K},\underline{\boldsymbol{u}}_{i,1}...\underline{\boldsymbol{u}}_{i,<\text{eos}>})$. Notice that we do not use autoregressive decoding as in Vaswani et al. (2017), so our decoder could also be viewed as another encoder with attention to the output of the encoder TE. Finally, to convert the hidden state size to the entity embedding size, we let the outputs of decoder go through another linear layer

L^o to get the facet embedding (i.e., sentence pattern embedding): $\underline{\boldsymbol{s}}_{i,k} = L^o(\underline{\boldsymbol{d}}_{i,k})$.

2.3 Objective Function

When measuring the distance between the jth entity pair and the ith sentence pattern, we compute the Euclidean distance between the entity pair embedding $\underline{\tilde{\boldsymbol{e}}}_j$ and its closest facet embedding of the ith sentence pattern. The distance is defined as

$$D(\{\underline{\boldsymbol{s}}_{i,k}\}_{k=1}^{K}, \underline{\tilde{\boldsymbol{e}}}_j) = \min_{k=1}^{K} \min_{0 \leq \eta_k \leq 1} ||\underline{\tilde{\boldsymbol{e}}}_j - \eta_k \underline{\boldsymbol{s}}_{i,k}||^2, \quad (1)$$

where the entity pair embedding is normalized (i.e., $||\underline{\tilde{\boldsymbol{e}}}_j|| = 1$). During testing, we ignore the magnitude of facet embeddings, so we use η_k to eliminate the magnitude of facet embeddings $\underline{\boldsymbol{s}}_{i,k}$ during training. We do not allow negative η_k to prevent the gradient flow from pushing $\underline{\boldsymbol{s}}_{i,k}$ toward the inverse direction of $\underline{\tilde{\boldsymbol{e}}}_j$ and we ensure $\eta_k \leq 1$ to avoid the neural model from outputting $\underline{\boldsymbol{s}}_{i,k}$ with a very small magnitude.

As in Figure 2, we minimize the distance $D(\{\underline{\boldsymbol{s}}_{i,k}\}_{k=1}^{K}, \underline{\tilde{\boldsymbol{e}}}_j)$ in our loss function when the ith sentence pair co-occurs with the jth entity pair (i.e., $y_{i,j} = 1$). For negative samples (i.e., $y_{i,j} = 0$), we maximize the distance instead. That is, the major term of our loss function is defined as

$$\sum_{(i,j) \in R} (2 \cdot y_{i,j} - 1) r_{i,j} D(\{\underline{\boldsymbol{s}}_{i,k}\}_{k=1}^{K}, \underline{\tilde{\boldsymbol{e}}}_j), \quad (2)$$

and the other regularization term Ω in the loss function will be described in the appendix. R is a set that includes all positive and negative samples. Positive samples are (i, j) such that $y_{i,j} = 1$ and the negative samples are constructed by pairing a randomly selected sentence pattern with the jth entity pair. To balance the influence of popular entity pairs (i.e., entity pairs that co-occur with many sentence patterns) and rare entity pairs on our model, we set the weight of each pair, $r_{i,j} \propto \frac{1}{\sum_i y_{i,j}}$ and $\frac{\sum_{(i,j) \in R} r_{i,j}}{|R|} = 1$.

We generate the embeddings for KB relations in a similar way. We use a single token to represent the relation and append an <eos> (e.g., *per:spouse* <eos>) to form the input of our neural model. The KB relations usually co-occur with more entity pairs, so we set the number of facet embeddings for KB relations K_{rel} to be larger than the number of facet embeddings for sentence patterns K.

2.4 Connection to Clustering

If a sentence pattern contains multiple facets that describe different relations between the entity pairs, the pattern often co-occurs with different kinds of entity pairs. For example, *"$ARG1 's partner $ARG2"* in Figure 2 could express the collaboration relationship between two companies or the partnership between two people, so the sentence patterns could co-occur with two companies such as *(Google, Facebook)* and two people such as *(Bob Bryan, Mike Bryan)*.

Different kinds of entity pairs often have very different embeddings, so we could discover the facets of sentence patterns by clustering the embeddings of entity pairs. Here, a facet refers to a mode of the embedding distribution of the entity pairs that could possibly co-occur with the sentence pattern. A facet could be represented by multiple facet embeddings and each facet embedding corresponds to a cluster center of the entity pair embeddings. Hence, although the number of facet embeddings K is fixed for all the sentence patterns, our model can capture the facets of the sentence patterns well when the number of facets is less than K.

In equation 1, we choose the closest facet embedding of the sentence pattern for each co-occurring entity pair embedding and minimize their distance. For example, $\underline{\boldsymbol{s}}_{i,2}$ and the embedding of *(Bob Bryan, Mike Bryan)* are pulled closer in Figure 2. Minimizing equation 1 by passing the gradient through the scaled facet embedding $\eta_k \underline{\boldsymbol{s}}_{i,k}$ is the same as minimizing a Kmeans loss, so the loss term induced by positive sample pairs encourage each $\underline{\boldsymbol{s}}_{i,k}$ to become the cluster center of its nearby co-occurring entity pair embeddings. The details of our training algorithm could be found in the appendix.

The co-occurrence matrices in RE tasks are usually extremely sparse, and most of the sentence patterns only co-occur with a few entity pairs, which makes it difficult to derive multiple high-quality embeddings by clustering the co-occurring entity pair embeddings as in multi-sense word embedding methods such as Neelakantan et al. (2014). The proposed method solves this sparsity challenge by predicting the cluster centers using a neural model. For instance, even if *"$ARG1 's partner $ARG2"* does not co-occur with many entity pairs, its embeddings are encouraged to be close to the embeddings of entity pairs that co-occur with other similar patterns (e.g., *"$ARG1 and her partner $ARG2"*).

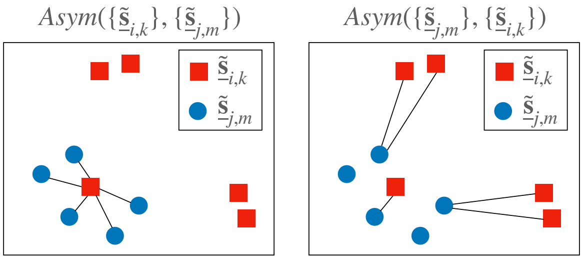


Figure 3: Comparison of the asymmetric similarities. $Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\}) > Asym(\{\underline{\tilde{\mathbf{s}}}_{j,m}\}, \{\underline{\tilde{\mathbf{s}}}_{i,k}\})$ because the average cosine distance on the left is smaller than that on the right.

2.5 Scoring Functions

In compositional universal schema, the similarity between the ith and jth sentence patterns are measured by the symmetric cosine similarity $\underline{\tilde{\mathbf{s}}}_{i,1}^{T}\underline{\tilde{\mathbf{s}}}_{j,1}$, where $\underline{\tilde{\mathbf{s}}}_{i,1} = \frac{\underline{\mathbf{s}}_{i,1}}{||\underline{\mathbf{s}}_{i,1}||}$. When using multiple embeddings to represent a sentence pattern, we can compute the asymmetric similarity as

$$Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\}) = \frac{\sum_{m=1}^{K} \max_{k=1}^{K}(\underline{\tilde{\mathbf{s}}}_{i,k}^{T}\underline{\tilde{\mathbf{s}}}_{j,m})}{K}. \quad (3)$$

In an example of Figure 3, a red square $\underline{\tilde{\mathbf{s}}}_{i,k}$ is close to all the blue points, which leads to a high $Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\})$.

Between two sentence patterns with entailment relation, we empirically find that the embeddings of a premise (the more specific pattern) often have some facet embeddings that are far away from all the embeddings of its hypothesis (the more general pattern). Relying on the tendency, we could detect the direction of the entailment relation. For example, the ith sentence pattern (red squares) in Figure 3 is more likely to be premise if the ith and jth (blue circles) sentence patterns have an entailment relation.

We suspect the reason is that more specific patterns could contain more words that are similar to the words of other patterns expressing different relations. For example, *"$ARG1 , the wife of fellow $ARG2"* have a facet embedding for spouse relation and another facet embedding for the co-worker relation because the pattern has high word overlapping with *"$ARG1 , the wife of $ARG2"* and *"$ARG1 and her fellow $ARG2"*. Another possible reason is that the articles in our corpus tend to use more specific patterns to express the relation between a pair of entities (Shwartz et al., 2017).

When performing RE, we compute the symmetric similarity between ith sentence pattern and jth KB relation $Sim(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\})$ by

$$\frac{Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\}) + Asym(\{\underline{\tilde{\mathbf{s}}}_{j,m}\}, \{\underline{\tilde{\mathbf{s}}}_{i,k}\})}{2}. \quad (4)$$

3 Experiments

We primarily compare our method with compositional universal schema (CUSchema) (Verga et al., 2016) because CUSchema is one of the state-of-the-art RE methods in the small model regime (without using large pre-trained language models) (Chang et al., 2016; Chaganty et al., 2017).[1]

In Section 3.1, we visualize and analyze the facet embeddings. Next, we use distant-supervised RE tasks to evaluate our symmetric similarity measurement in Section 3.2, and detect entailment between sentence patterns to evaluate our asymmetric similarity measurement in Section 3.3.

3.1 Embedding Visualization

We visualize the embeddings of sentence patterns and a KB relation from the single embedding model and multi-facet embedding model that perform the best in the RE tasks (i.e., **Ours (Single-Trans)** and **Ours (Trans)** in Table 1). We project the facet embeddings to a 2-dimensional space using multidimensional scaling (MDS) (Borg and Groenen, 2005) and visualize the embeddings of one KB relation and three related sentence patterns in Figure 4. The three sentence patterns are selected from our validation set, so the model is not aware of the entity pairs that actually co-occur with the patterns during training. For each facet embedding, we show two among five of its closest entity pairs to visualize the meaning of the embedding space.[2]

---

[1] We have not yet applied the multi-facet embeddings approach to the models that rely on a large pretrained language model (LM) (Baldini Soares et al., 2019) due to computational and evaluation considerations. Computationally speaking, training state-of-the-art models requires intensive GPU resources. Besides, a smaller model size might be desired when we need to construct a knowledge base from a large corpus in real time. Moreover, there is no existing pretrained LM in some domains (Zhang et al., 2019), and training the LM in a new domain from scratch requires even more GPU resources.

In terms of the evaluation consideration, our method is an improvement over CUSchema, so we want to compare it with CUSchema fairly. Furthermore, evaluating entailment between two full sentences is more difficult than between the sentence patterns, and we are not aware of a LM-based model that only considers the text between the entity pairs.

[2] Notice that our training signal is sparse and noisy and the projection does not necessarily preserve the original distances, so the entity pairs with similar relations might be relatively far away from each other.

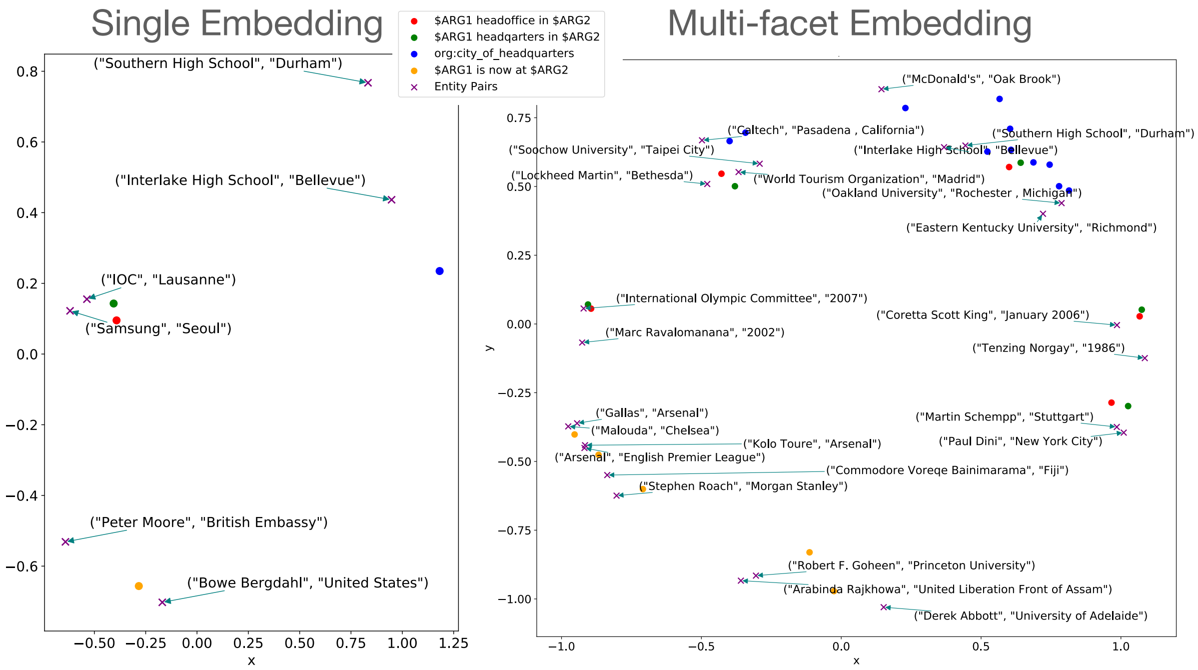


Figure 4: Facet embedding visualization of **Ours (Single-Trans)** on the left and **Ours (Trans)** on the right. Dots are the facet embeddings outputted by our models and crosses are their nearby entity pair embeddings

In the single embedding model, the embedding of *org:city_of_headquarter* is close to the embedding of *(school, location)* while *"$ARG1 headoffice in $ARG2"* is close to *(company, location)* and *"$ARG1 headquarter in $ARG2"*.

In the multi-facet embedding model, some embeddings of *org:city_of_headquarter* are closer to *(school, location)* and others are closer to *(company, location)*. In addition to these entity pairs, *"$ARG1 headoffice in $ARG2"* and *"$ARG1 headquarter in $ARG2"* also co-occur with *(people, location)* and *(people/organization, year)*. Using the visualization of multi-facet embedding, we can understand which facets of *org:city_of_headquarter* are similar or dissimilar to *"$ARG1 headoffice in $ARG2"*, which cannot be done if all facets are averaged into a single embedding as in the traditional models.

The facet embeddings of *"$ARG1 is now at $ARG2"* are close to *(people, organization)* where the organization could be school, sports team, and company. Using multiple embeddings could avoid enforcing the closeness of these entity pairs with different relations. The results also indicate that our model can output reasonable cluster centers despite learning from the sparse and noisy training data. Finally, we can see that if a sentence pattern has fewer facets than K, our model learns to output some very similar facet embeddings, which makes the performance less sensitive to the setting of K.

3.2 Relation Extraction

We follow the same training data and testing protocol in compositional universal schema (CUSchema) (Verga et al., 2016)[3] to highlight the benefit of predicting multiple facet embeddings, and the relation extraction step in TAC KBP slot-filling tasks is used to compare the different models.

Setup: The training data for our RE models are prepared by distant supervision without requiring any manually labeled data. The relations in Freebase (Bollacker et al., 2008) are mapped to TAC relations (e.g., *org:city_of_headquarter*) and the NER tagger and entity linker are run in a raw text corpus. Then, the training data is cleaned using the methods in Roth et al. (2013).

During testing, we are given a query containing the head entity and a query TAC relation in the slot-filling tasks, and the goal is to extract the tail entity from the candidate sentences. The NER tagger and query expansion are used to gather the candidate sentence patterns, and we compute the similarity scores from different models between the candidate sentence patterns and query relation. Finally, we compare the extracted second entity with the ground truth using exact string matching and report the precision, recall, and F1 scores.

Following Verga et al. (2016), we use TAC 2012

---

[3] `https://github.com/patverga/torch-relation-extraction`

Method	TAC 2012 (Validation)			TAC 2013			TAC 2014		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
USchema*	**34.8**	23.7	28.2	**42.6**	29.4	34.8	**35.5**	24.3	28.8
CUSchema (LSTM)*	27.0	**32.7**	29.6	39.6	32.2	35.5	32.9	27.3	29.8
Ours (Single-LSTM)	25.7	21.7	23.5	30.4	26.3	28.2	22.1	20.5	21.3
Ours (Single-Trans)	26.1	21.6	23.7	29.5	25.2	27.2	19.0	21.2	20.0
Ours (LSTM)	32.0	28.9	30.3	41.3	**33.9**	37.2	34.1	**29.5**	**31.6**
Ours (Trans)	33.6	27.7	**30.4**	42.5	33.2	**37.3**	34.6	28.5	31.3
USchema + CUSchema (LSTM)*	29.3	32.8	30.9	**41.9**	34.4	37.7	29.3	34.1	31.5
USchema + Ours (LSTM)	29.2	33.7	31.3	38.1	**38.9**	38.5	31.5	**34.4**	32.9
USchema + Ours (Trans)	**30.4**	**33.9**	**32.1**	39.0	38.8	**38.9**	**32.0**	34.0	**33.0**

Table 1: Distantly supervised relation extraction using different versions of the universal schema. All numbers are %. CUSchema refers to compositional universal schema. Trans is an abbreviation of Transformer. The best scores of the single models and ensemble models are highlighted. *The performance of TAC 2013 and 2014 are copied from Verga et al. (2016).

as our validation set to determine the threshold score for each TAC relation. Each model's hyperparameters are tuned separately using the validation set (TAC 2012) to ensure a fair comparison.

We compare the following methods:

Ours (Trans): Our method that measures the similarity between the sentence pattern $\{\underline{\tilde{s}}_{i,k}\}$ and TAC relation $\{\underline{\tilde{s}}_{j,m}\}$ using $Sim(\{\underline{\tilde{s}}_{i,k}\}, \{\underline{\tilde{s}}_{j,m}\})$ in equation 4. Trans is an abbreviation of the Transformer encoder. We set $K = 5$ and $K_{rel} = 11$ based on the validation set.

Ours (LSTM): The same as Ours (Trans) except that we use bi-LSTM as our encoder instead.

Ours (Single-*): Our methods that use single facet embedding to represent each sentence pattern or KB relation. When setting $K = K_{rel} = 1$, our decoder becomes the interleaving feedforward layers and cross-attention layers attending to the output embeddings of the encoder.

CUSchema (LSTM): Compositional universal schema (Verga et al., 2016). The method is similar to Ours (Single-LSTM) but uses a different loss function, neural architecture (no decoder), and hyperparameter search procedure.

USchema: Universal schema (Riedel et al., 2013) estimates every sentence pattern embedding by factorizing the co-occurrence matrices (i.e., replacing the bi-LSTM in CUSchema with a look-up table).

USchema + *: Verga et al. (2016) show that taking the maximal similarity between USchema and CUSchema model improves the F1. We also apply the same merging procedure to our model.

Results: In Table 1, the proposed method **Ours (Trans)** significantly outperform **CUSchema (LSTM)** before and after combining with universal schema. As far as we know, our proposed multi-facet embedding is the first method that outperforms compositional universal schema using the same training signal in the distant-supervised RE benchmark they proposed.

Although the recall of **USchema** is low because it does not exploit the similarity between the patterns (e.g., *"$ARG1 happily married $ARG2"* is similar to *"$ARG1 married $ARG2"*), **USchema** has a high precision because it also won't be misled by the similarity (e.g., *"$ARG1, and his wife $ARG2"* expresses the *spouse* relation but *"$ARG1, his wife, and $ARG2"* does not) (Verga et al., 2016). Thus, combining **USchema** and **Ours (Trans)** leads to the best performance.

Ours (Trans) and **Ours (LSTM)** perform similarly. Furthermore, **Ours (LSTM)** performs much better than **Ours (Single-LSTM)**, which demonstrates the effectiveness of using multiple embeddings. Notice that multiple facet embeddings could improve the performance even after the training data have been cleaned. This indicates that our method is complementary to the noise removal methods in Roth et al. (2013).

3.3 Entailment Detection

Entailment is a common and fundamental relation between two sentence patterns. Some examples could be seen in Table 2. Unsupervised hypernym detection (i.e., entailment at the word level) is extensively studied (Shwartz et al., 2017), but we are not aware of any previous work on unsupervised entailment detection at the sentence level, nor any existing entailment dataset between sentence patterns. Thus, we create one.

Dataset Creation: We use WordNet (Miller, 1998) to discover the entailment candidates of sen-

Premise (Specific Pattern)	Hypothesis (General Pattern)	Label	Ours	CUSchema	Ours Diff	Freq Diff
$ARG1 , the president of the $ARG2	$ARG1 , the leader of the $ARG2	Entailment	0.98	0.94	+	+
$ARG1 's chairman , $ARG2	$ARG1 's leader , $ARG2	Entailment	0.95	0.87	+	-
$ARG1 's father , $ARG2	$ARG1 's leader , $ARG2	Other	0.08	0.52	NA	NA
$ARG1 worked with $ARG2	$ARG1 deal with $ARG2	Entailment	0.92	0.83	+	-
$ARG1 had with $ARG2	$ARG1 deal with $ARG2	Other	0.96	0.88	NA	NA
$ARG1 said the $ARG2	$ARG1 say the $ARG2	Paraphrase	0.93	0.92	NA	NA

Table 2: Example of sentence pattern pairs, its label, and our predictions in our entailment experiment. **Ours** and **Ours Diff** are the predictions from **Ours (Trans)**. **Freq Diff** is the frequency difference baseline.

Method	Classification	Direction Detection	
	AP@all	Micro Acc	Macro Acc
Random	21.9	50.0	50.0
Freq Diff	21.4	54.5	47.3
CUSchema	31.2	50.0	50.0
Ours (Single)	23.6	50.0	50.0
Ours	**37.8**	**63.1**	**55.4**

Table 3: Comparison of entailment detection methods. AP and Acc are average precision and accuracy, respectively. All numbers are %. Our methods use a Transformer as their encoder.

tence pattern pairs and manually label the candidates. For each sentence pattern in the training data of Verga et al. (2016), we replace one word at a time with its hypernym based on the WordNet hierarchy. The two sentence patterns before and after replacement form an entailment candidate.

We label 1,500 pairs of the most popular sentence pattern, which co-occurs with the highest number of unique entity pairs. Each candidate could be labeled as entailment, paraphrase, or other. Finally, around 20% of the candidates are randomly chosen to form the validation set, and the rest are in the test set. More details of the dataset creation process could be seen in the appendix

In this dataset, only 22% and 10% of candidates are labeled as entailment and paraphrase, respectively. This suggests that entailment relation between two sentence patterns is hard to be inferred by only the hypernym relation (i.e., entailment relation at the word level) in WordNet.

Setup: We evaluate entailment detection using the typical setup and metrics in hypernym detection (Shwartz et al., 2017). Negative examples include the candidates labeled as paraphrases and others. We compare the average precision of different methods (i.e., AUC in the precision-recall curve) (Hastie et al., 2009). In addition, we predict the direction of entailment relation in each pair (i.e., which pattern is the premise) and report the accuracy. Many hypotheses have the same hypernyms such as the *leader* in Table 2, so we also report the macro accuracy of direction detection averaged across every hypernym in the hypotheses.

The task is challenging because all the candidates have a word-level entailment relation if their compositional meaning is ignored. Furthermore, we cannot infer the entailment direction based on the tendency that longer sentence patterns tend to be more specific because most of the candidate pairs in this dataset have the same length.

As described in Section 2.5, our models detect the direction by computing **Ours Diff** as $Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\}) - Asym(\{\underline{\tilde{\mathbf{s}}}_{j,m}\}, \{\underline{\tilde{\mathbf{s}}}_{i,k}\})$ and predict the ith sentence pattern to be premise if **Ours Diff** > 0. When performing entailment classification, we use as the asymmetric similarity scores $Asym(\{\underline{\tilde{\mathbf{s}}}_{i,k}\}, \{\underline{\tilde{\mathbf{s}}}_{j,m}\})$. We report the performance of **Ours (Trans)**, which is the same best model in the RE experiment.

In entailment classification, we compare the results with cosine similarity from **Ours (Single-Trans)** and **CUschema**. We also test the frequency difference, which is a strong baseline in hypernym direction detection (Chang et al., 2018). **Freq Diff** = Freq(S_j) - Freq(S_i) where Freq(S_i) is the number of unique entity pairs co-occurred with the ith sentence pattern. The baseline predicts S_i to be premise if **Freq Diff** > 0 because more general sentence patterns should co-occur with more entity pairs. As a reference, we also report the performance of random scores.

Results: The quantitative and qualitative comparison are presented in Table 3 and Table 2, respectively. Our model that uses multi-facet embeddings significantly outperforms the other baselines. We hypothesize that a major reason is that the sentence patterns with an entailment relation are often similar in some but not all of the facets, and our asymmetric similarity measurement is better at capturing the facet overlapping.

4 Related Work

Relation extraction (RE) is widely studied. Han et al. (2020) summarize the trend of recent studies and point out one of the major challenges is the cost of collecting the labels. Distant supervision (Mintz et al., 2009) and its follow-up work enable us to collect a large amount of training data at a low cost, but the violation of its assumptions often introduces substantial noise into the supervision signal. Our goal is to alleviate the noise issue by representing every sentence pattern using multiple embeddings.

Other noise reduction methods have also been proposed (Roth et al., 2013). For instance, we can adopt multi-instance learning techniques (Yao et al., 2010; Surdeanu et al., 2012; Amin et al., 2020), global topic model (Alfonseca et al., 2012), or both (Roth and Klakow, 2013). We can also reduce the noise by counting the number of shared entity pairs between a sentence pattern and a KB relation (Takamatsu et al., 2012; Su et al., 2018). Nevertheless, the studies focus on mitigating the noise caused by assuming similarity between the sentence patterns and KB relations that co-occur with the same entity pairs, while our method can also reduce the noise from two sentence patterns sharing the same entity pair. Besides, our method is complementary to popular noise reduction methods because our improvement is shown in the training data that have been cleaned (Verga et al., 2016).

Our method is conceptually related to some studies for lexical semantics. For example, word sense induction or unsupervised hypernymy detection can be addressed by clustering the co-occurring words (Neelakantan et al., 2014; Athiwaratkun and Wilson, 2017; Chang et al., 2018). However, the clustering-based methods do not apply to RE because the co-occurring matrix for RE is much sparser (see Section 2.4 for more details).

Finally, our work is inspired by Chang et al. (2021), but they focus on improving the sentence representation rather than RE. We encourage the facet embeddings to become the centers in Kmeans clustering instead of NNSC (non-negative sparse coding) clustering used in Chang et al. (2021), due to its simplicity, efficiency, and better RE performance. Moreover, we discover that an additional regularization described in the appendixis crucial to overcome the sparsity challenge in RE.

5 Conclusion

In this work, we address the limitation of representing each sentence pattern using only a single embedding, and our approach improves the distantly-supervised RE performances of compositional universal schema.

Relying on only a very sparse co-occurrence matrix between the sentence patterns and entity pairs, we show that it is possible to predict reasonable cluster centers of entity pair embeddings and to predict the entailment relation between two sentence patterns without any labels.

Acknowledgements

We thank Ge Gao for the preliminary exploration of this project. We also thank the anonymous reviewers for their constructive feedback.

This work was supported in part by the Center for Data Science and the Center for Intelligent Information Retrieval, in part by the Chan Zuckerberg Initiative under the project Scientific Knowledge Base Construction, in part using high performance computing equipment obtained under a grant from the Collaborative R&D Fund managed by the Massachusetts Technology Collaborative, in part by the National Science Foundation (NSF) grant numbers DMR-1534431 and IIS-1514053.

Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Kosuke Akimoto, Takuya Hiraoka, Kunihiko Sadamasa, and Mathias Niepert. 2019. Cross-sentence n-ary relation extraction using lower-arity universal schemas. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6225–6231, Hong Kong, China. Association for Computational Linguistics.

Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. 2012. Pattern learning for relation extraction with a hierarchical topic model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 54–59, Jeju Island, Korea. Association for Computational Linguistics.

Saadullah Amin, Katherine Ann Dunfield, Anna Vechkaeva, and Guenter Neumann. 2020. A data-driven approach for noise reduction in distantly supervised biomedical relation extraction. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 187–194, Online. Association for Computational Linguistics.

Ben Athiwaratkun and Andrew Wilson. 2017. Multimodal word distributions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1645–1656, Vancouver, Canada. Association for Computational Linguistics.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM.

Ingwer Borg and Patrick JF Groenen. 2005. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media.

Arun Chaganty, Ashwin Paranjape, Percy Liang, and Christopher D. Manning. 2017. Importance sampling for unbiased on-demand evaluation of knowledge base population. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1038–1048, Copenhagen, Denmark. Association for Computational Linguistics.

Haw-Shiuan Chang, Amol Agrawal, and Andrew McCallum. 2021. Extending multi-sense word embedding to phrases and sentences for unsupervised semantic applications. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Haw-Shiuan Chang, Abdurrahman Munir, Ao Liu, Johnny Tian-Zheng Wei, Aaron Traylor, Ajay Nagesh, Nicholas Monath, Patrick Verga, Emma Strubell, and Andrew McCallum. 2016. Extracting multilingual relations under limited resources: TAC 2016 cold-start KB construction and slot-filling using compositional universal schema. In *Proceedings of the 2016 Text Analysis Conference, TAC 2016, Gaithersburg, Maryland, USA, November 14-15, 2016*. NIST.

Haw-Shiuan Chang, Ziyun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional inclusion vector embedding for unsupervised hypernymy detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 485–495, New Orleans, Louisiana. Association for Computational Linguistics.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question answering on knowledge bases and text using universal schema and memory networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–365, Vancouver, Canada. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758, Suzhou, China. Association for Computational Linguistics.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.

Bethany Percha and Russ B. Altman. 2018. A global network of biomedical relationships derived from text. *Bioinform.*, 34(15):2614–2624.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. 2013. A survey of noise reduction methods for distant supervision. In *Proceedings of the 2013 workshop on Automated knowledge base construction*.

Benjamin Roth and Dietrich Klakow. 2013. Feature-based models for improving the quality of noisy training data for relation extraction. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 1181–1184. ACM.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 65–75, Valencia, Spain. Association for Computational Linguistics.

Yu Su, Honglei Liu, Semih Yavuz, Izzeddin Gür, Huan Sun, and Xifeng Yan. 2018. Global relation embedding for relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 820–830, New Orleans, Louisiana. Association for Computational Linguistics.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 721–729, Jeju Island, Korea. Association for Computational Linguistics.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Lisbon, Portugal. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Patrick Verga, David Belanger, Emma Strubell, Benjamin Roth, and Andrew McCallum. 2016. Multilingual relation extraction using compositional universal schema. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 886–896, San Diego, California. Association for Computational Linguistics.

Patrick Verga, Arvind Neelakantan, and Andrew McCallum. 2017. Generalizing to unseen entities and entity pairs with row-less universal schema. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 613–622, Valencia, Spain. Association for Computational Linguistics.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884, New Orleans, Louisiana. Association for Computational Linguistics.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2010. Collective cross-document relation extraction without labelled data. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1013–1023, Cambridge, MA. Association for Computational Linguistics.

Dongxu Zhang, Subhabrata Mukherjee, Colin Lockard, Luna Dong, and Andrew McCallum. 2019. OpenKI: Integrating Open Information Extraction and Knowledge Bases with Relation Inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 762–772, Minneapolis, Minnesota. Association for Computational Linguistics.

Exploring Transitivity in Neural NLI Models through Veridicality

Hitomi Yanaka[1], **Koji Mineshima**[2], **and Kentaro Inui**[3,1]
[1]RIKEN, [2]Keio University, [3]Tohoku University
`hitomi.yanaka@riken.jp`, `minesima@abelard.flet.keio.ac.jp`,
`inui@ecei.tohoku.ac.jp`

Abstract

Despite the recent success of deep neural networks in natural language processing, the extent to which they can demonstrate human-like generalization capacities for natural language understanding remains unclear. We explore this issue in the domain of natural language inference (NLI), focusing on the *transitivity* of inference relations, a fundamental property for systematically drawing inferences. A model capturing transitivity can compose basic inference patterns and draw new inferences. We introduce an analysis method using synthetic and naturalistic NLI datasets involving clause-embedding verbs to evaluate whether models can perform transitivity inferences composed of veridical inferences and arbitrary inference types. We find that current NLI models do not perform consistently well on transitivity inference tasks, suggesting that they lack the generalization capacity for drawing composite inferences from provided training examples. The data and code for our analysis are publicly available at `https://github.com/verypluming/transitivity`.

1 Introduction

Deep neural networks (DNNs) have shown impressive performance in many natural language processing tasks. In particular, DNN models pretrained with large-scale data such as BERT (Devlin et al., 2019) have achieved high accuracy in various benchmark tasks (Wang et al., 2019a,b), which suggests that they might possess some generalization capacities that are a hallmark of human cognition. However, recent analyses (Talmor and Berant, 2019; Liu et al., 2019; McCoy et al., 2019) have shown that high accuracy on a test set drawn from the same distribution as the training set does not always indicate that the model has obtained the intended ability, so it remains unclear to what extent DNN models can learn the systematic generalization in natural language from training instances.

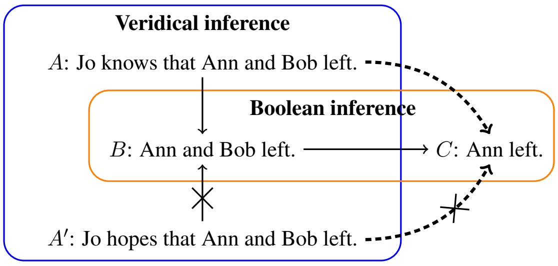


Figure 1: Illustration of transitivity inferences (indicated by ⇢) composed of two basic inferences, veridical and Boolean. Arrows indicate *entailment* and arrows with a cross (×) indicate *non-entailment*.

Central to human-like generalization capacities is the fact that ability to understand a given sentence is related to ability to understand other sentences, called *systematicity* of human cognition in Fodor and Pylyshyn (1988). Thus, if speakers understand the meaning of the sentence *Ann loves Bob*, they must also understand the meaning of structurally related sentences such as *Bob loves Ann*. We explore whether DNN models possess this type of generalization capacity in the domain of natural language inference (NLI), which is the task to judge whether a premise entails a hypothesis (Dagan et al., 2013; Bowman et al., 2015a).

A key property underlying systematicity of drawing inferences is the *transitivity* of inference relations, illustrated in Figure 1. Schematically, if a model learns a basic inference pattern from A to B and one from B to C, it should be able to compose the two patterns to draw a new inference from A to C. If a model lacks this generalization capacity, it must memorize an exponential number of inference combinations independently of basic patterns.

Among the various inference patterns, we focus on transitivity inferences that combine *veridical* in-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 920–934
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

ferences with other types. In veridical inferences, one must distinguish two entailment types. For example, the verb *know* is called **veridical** in that "x *knows that* P" entails that P is true, while the verb *hope* is called **non-veridical** since "x *hopes that* P" does not entail that P is true. Veridical inferences can relatively easily compose transitivity inferences at scale by embedding various inference types into clause-embedding verbs. For instance, as Figure 1 shows, if a model has the ability to perform both Boolean inference and veridical inference, it is desirable to have the ability to combine both types to make a chained inference.

Such transitivity inferences are by no means trivial. For instance, if the premise is changed to *Jo knows that Ann* ***or*** *Bob left*, it does not follow that *Bob left*, even though the veridical verb *know* appears. Models relying on shallow heuristics such as lexical overlap can wrongly predict *entailment* in this case. To correctly handle such composite inferences, models must capture structural relations between veridical inferences and various kinds of embedded inference.

Previous studies on the generalization capacities of NLI models have addressed how models could learn inferences with various challenging linguistic phenomena (Bowman et al., 2015b; Dasgupta et al., 2018; Geiger et al., 2019, 2020; Yanaka et al., 2019a,b; Richardson et al., 2020). However, these studies have focused on the linguistic phenomena in isolation, and thus do not address how a model could learn the *interactions* between them. Our aim is to fill this gap by presenting a method for probing generalization capacity of DNN models performing transitivity inferences.

This study provides three main contributions. First, we create and publicly release two types of NLI datasets for testing model ability to perform transitivity inferences: a fully synthetic dataset that combines veridical inferences and Boolean inferences, and a naturalistic dataset that combines veridical inferences with lexical and structural inferences. Second, we use these datasets to systematically expose models to basic inference patterns and test them on a variety of combinations. This will demonstrate that the models lack the ability to capture transitivity of inference. Third, we investigate whether data augmentation with new combination patterns helps models to learn transitivity. Experiments show that the data augmentation improves model performance on similar combinations, regardless of the existence of basic inference patterns in the training set. These results suggest there is much room for improving the generalization capacities of DNN models for combining basic inferential abilities.

2 Related Work

Transitivity The transitivity of entailment relations, which derives $A \rightarrow C$ from $A \rightarrow B$ and $B \rightarrow C$, is incorporated into logic-based NLI systems using automated theorem proving (Abzianidze, 2015; Mineshima et al., 2015). This is a basic property of formal logic, also known as *syllogism* in traditional logic or the *cut rule* in proof theory (Troelstra and Schwichtenberg, 2000; van Dalen, 2013). Transitivity inference in its various forms has also been widely studied as a fundamental property of human reasoning in cognitive psychology (Johnson-Laird and Byrne, 1991; Khemlani and Johnson-Laird, 2012). In the context of NLP, previous works have proposed a method for training models with transitivity constraints in multi-hop reasoning tasks (Asai and Hajishirzi, 2020) and temporal relation extraction tasks (Ning et al., 2017). Clark et al. (2020) investigated a transformer's ability to perform a chain of reasoning where reasoning rules are explicitly given. In this work, we study model ability to learn transitivity of entailment relations from training examples, rather than explicitly providing rules.

Systematicity There has been extensive discussion of whether neural networks (aka Connectionist models) can exhibit systematicity of cognitive capacities (Fodor and Pylyshyn, 1988; Marcus, 2003). Recent works have explored whether modern neural networks can learn systematicity in semantic parsing tasks (Lake and Baroni, 2017; Baroni, 2020; Kim and Linzen, 2020) and question answering tasks (Sinha et al., 2019), whereas our focus is the systematicity in NLI.

In works related to the systematicity in NLI, Goodwin et al. (2020), Yanaka et al. (2020), and Geiger et al. (2020) used a manually constructed NLI dataset of monotonicity inferences with and without negation (e.g., *The child is not holding plants* $\rightarrow$ *The child is not holding flowers*) to examine DNN models' generalization capacities. While these approaches concentrate on monotonicity inferences involving quantifiers and negative expressions, our method using veridical inference is general in that it can be applied to any entailment relation that combines basic inference patterns; we generate composite inferences by embedding various types of sentences into clause-embedding verbs.

Fodor and Pylyshyn (1988) distinguished systematicity (roughly, the ability to understand sentences that are structurally related to each other) from productivity (the ability to understand an infinite set of sentences), claiming that systematicity poses a serious challenge to neural network models. Yanaka et al. (2020) tested both systematicity and productivity of DNN models with a synthetic dataset of monotonicity inferences for upward (e.g., *some, at least three*) and downward (e.g., *few, at most three*) quantifiers, where handling productivity (recursion) makes sentences more involved (e.g., iterated relative clauses and negation). Focusing on systematicity rather than productivity allows testing models with more natural and less complicated data, as compared to sentences appearing in monotonicity inferences.

Veridicality Veridical inferences, including those licensed by factive and implicative verbs, have been intensively studied in the literature of semantics and pragmatics (Karttunen and Peters, 1979; Beaver, 2001). Recent work has revealed graded and context-sensitive aspects of veridicality inferences, creating veridicality judgement datasets (de Marneffe et al., 2012; White and Rawlins, 2018; White et al., 2018). While we use only a subset of veridical predicates discussed in the literature, our method can be extended to more complex inferences, such as factive presupposition.

Ross and Pavlick (2019) presented a naturalistic veridicality dataset and compared the predictions of a BERT-based NLI model and human judgements. These previous studies on veridicality inferences have tended to focus on relations between whole sentences (e.g., *Jo remembered that there was a wild deer jumping a fence*) and its embedded material (e.g., *There was a wild deer jumping a fence*). By contrast, we consider the interactions of veridicality inferences and other inference types (see Section 3.2), including cases where the embedded material is further paraphrased via linguistic phenomena (e.g., *Jo remembered that there was a wild deer jumping a fence* ⇒ *An animal was jumping*). We also collect human judgements on our dataset and compare them with model predictions (see Section 4.4).

Probing NLI models Many studies of probing NLI models have found that current models often fail on linguistically challenging (adversarial) inferences (Rozen et al., 2019; Nie et al., 2019; Yanaka et al., 2019a; Richardson et al., 2020), learning undesired biases (Glockner et al., 2018; Poliak et al., 2018; Tsuchiya, 2018; Liu et al., 2019), and heuristics (McCoy et al., 2019). Our approach also provides adversarial test sets against such heuristics by considering combinations of veridical inferences and diverse (lexical, structural, and logical) types of inferences.

One way to learn challenging inferences is data augmentation, and prior studies (Yanaka et al., 2019b; Richardson et al., 2020; Min et al., 2020) have shown that data augmentation with synthesized datasets improves performance with challenging linguistic phenomena. However, it remains unclear whether data augmentation can help models learn *composite* inferences mixing several inference types from training instances. We address this question in Section 4.3.

3 Dataset

3.1 Overview

To investigate whether models can capture transitivity, we consider two basic inference patterns and their combinations. The first basic pattern, $\mathcal{I}_1$, is veridical inference. We write $f(s_1) \rightarrow s_1$ to denote a schematic veridical inference, where f is a clause-embedding verb and s_1 is the embedded clause. For instance, in the case of the inference pattern $A \rightarrow B$ in Figure 1, "*Jo knows that x*" corresponds to $f(x)$ and "*Ann and Bob left*" to s_1.

The second basic pattern, $\mathcal{I}_2$, provides an inference from the embedded material. We denote a premise-hypothesis pair of this second inference by $s_1 \rightarrow s_2$. Given two inferences $f(s_1) \rightarrow s_1$ in $\mathcal{I}_1$ and $s_1 \rightarrow s_2$ in $\mathcal{I}_2$, we consider a new inference $f(s_1) \rightarrow s_2$, where premise $f(s_1)$ is the same as that of $\mathcal{I}_1$ and hypothesis s_2 is the same as that of $\mathcal{I}_2$. See Table 1 and Table 2 for some examples of inferences $f(s_1) \rightarrow s_1$, $s_1 \rightarrow s_2$, and $f(s_1) \rightarrow s_2$. In this work, we consider binary labels, *entailment* and *non-entailment*, denoted by *yes* and *unk*, respectively. As Table 3 shows, the gold label on the $f(s_1) \rightarrow s_2$ pattern can be determined from those of the basic patterns $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$, following the transitivity of entailment relations.

We train models with the first and second patterns, $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$, and then test them

f	$f(s_1)\rightarrow s_1$	$s_1\rightarrow s_2$	$f(s_1)\rightarrow s_2$	**Example**
V	*yes*	*yes*	*yes*	$f(s_1)$: Someone **noticed** that [Henry and Daniel found Elliot, John and Fred]. s_1: Henry and Daniel found Elliot, John and Fred. s_2: Henry found John.
NV	*unk*	*yes*	*unk*	$f(s_1)$: Someone **expects** that [Tom and Ann admire Greg and Fred]. s_1: Tom and Ann admire Greg and Fred. s_2: Tom admires Greg.
NV	*unk*	*unk*	*unk*	$f(s_1)$: Someone **argued** that [it was not the case that Greg hated John or Elliot]. s_1: It was not the case that Greg hated John or Elliot. s_2: Greg hated John.

Table 1: Examples from our fully synthetic transitivity inference datasets. `V` and `NV` indicate types of clause-embedding verbs (veridical/non-veridical); *yes* means *entailment* and *unk* means *non-entailment*.

ID	f	$f(s_1)\rightarrow s_1$	$s_1\rightarrow s_2$	$f(s_1)\rightarrow s_2$	**Example**
2299	V	*yes*	*yes*	*yes*	$f(s_1)$: Someone **realized** that [a boy was playing a guitar]. s_1: A boy was playing a guitar. s_2: A kid was playing a guitar.
2049	V	*yes*	*unk*	*unk*	$f(s_1)$: Someone **remembered** that [a cat was playing with a device]. s_1: A cat was playing with a device. s_2: The boy was enthusiastically playing in the mud.
5024	NV	*unk*	*yes*	*unk*	$f(s_1)$: Someone **doubts** that [the woman is putting makeup on the man]. s_1: The woman is putting makeup on the man. s_2: A man's face is being painted by a woman.

Table 2: Examples from our naturalistic transitivity inference datasets. `V` and `NV` indicate types of clause-embedding verbs (veridical/non-veridical); *yes* means *entailment* and *unk* means *non-entailment*. **ID** indicates the original ID of $s_1\rightarrow s_2$ in the SICK dataset.

$f(s_1)\rightarrow s_1$	$s_1\rightarrow s_2$	$f(s_1)\rightarrow s_2$
yes	*yes*	***yes***
yes	*unk*	***unk***
unk	*yes*	***unk***
unk	*unk*	***unk***

Table 3: Rule for determining the $f(s_1)\rightarrow s_2$ label from the basic patterns $f(s_1)\rightarrow s_1$ and $s_1\rightarrow s_2$.

Type of f	**Verbs**
Veridical	realize, acknowledge, remember, note, find, notice, learn, see, reveal, discover, understand, know, admit, recognize, observe
Non-veridical	feel, claim, doubt, hope, predict, imply, suspect, wish, think, believe, hear, expect, estimate, assume, argue

Table 4: Clause-embedding verbs used for our dataset.

on a set of the composite inferences $f(s_1)\rightarrow s_2$ that combines them. Note that due to how they are constructed, the training and test sets do not overlap. Model capable of applying the transitivity inference from $f(s_1)\rightarrow s_1$ and $s_1\rightarrow s_2$ to $f(s_1)\rightarrow s_2$ should consistently predict the correct label of $f(s_1)\rightarrow s_2$ for any combination of $f(s_1)\rightarrow s_1$ and $s_1\rightarrow s_2$.

3.2 Data creation

We generate basic inferences $f(s_1)\rightarrow s_1$ and $s_1\rightarrow s_2$ and combine them to produce transitivity inferences $f(s_1)\rightarrow s_2$. To test diverse inference patterns, we consider two types of the second basic inference $s_1\rightarrow s_2$: synthesized Boolean inferences and naturalistic inferences using an existing NLI dataset, SICK (Marelli et al., 2014), which contains lexical inferences (e.g., *boy* → *kid* in ID 2299 in Table 2) and structural inferences (e.g., active-passive alternation in ID 5024 in Table 2). Since the ratio of the gold labels (*yes* and *unk*) is set to $1:1$ in both basic inference sets, the ratio of the gold labels for the transitivity test set is $1:3$ by the rule in Table 3. We reserve 10% of the basic inference set for the validation set.

Clause-embedding verbs We focus on clause-embedding verbs that take tensed subordinate clauses. Specifically, we collect 67 verbs appearing in both MegaVeridicality2 (White et al., 2018) and the verb veridicality dataset (Ross and Pavlick, 2019). As Table 4 shows, we select a final set of 30 clause-embedding verbs.

Following a previous study (White et al., 2018), we slot a clause-embedding verb f into a template with the form "Someone f that s_1" and generate premise $f(s_1)$ of veridical inference to avoid confounds introduced by world knowledge and pragmatic inference in the main clause. The clause-embedding verb f is in past or present tense, and we inflect the verb in the complement s_1 to match the tense of f.

When measuring the extent to which models can learn transitivity of entailment relations from training instances, it is desirable to determine the gold labels of composite inferences from those of basic inferences. Thus, we take the labels of veridical inference datasets predicted by the veridical and non-veridical distinction in lexical semantics as the gold standard. In addition, veridical inferences are sensitive to context, influenced by world knowledge and pragmatic factors (de Marneffe et al., 2012). Accordingly, we also present additional experiments to take into account such complexity of veridical inferences in Section 4.2.

Boolean inference To provide a fully synthetic transitivity inference dataset, we generate Boolean inferences with conjunction, disjunction, and negation. The data generation process is similar to the one in Yanaka et al. (2020): sentences are generated using a context-free grammar (CFG) associated with semantic composition rules in lambda-calculus. We first generate a set of premise sentences by the CFG rules and translate each sentence s_1 into a first-order-logic (FOL) formula F_1 in accordance with semantic composition rules specified in the CFG rules. Appendix A provides a set of CFG rules and semantic composition rules. We randomly select one of the atomic subformulas appearing in F_1 and take its positive or negative form, which we denote by F_2. Then we convert F_2 to a sentence s_2 using the same grammar. We set s_2 as a hypothesis.

The gold label for inference pair $s_1 \rightarrow s_2$ is determined by checking whether formula F_1 entails formula F_2 using an FOL theorem prover. The gold labels for $f(s_1) \rightarrow s_1$ and $f(s_1) \rightarrow s_2$ pairs are automatically determined according to the veridicality of a clause-embedding verb and the rule in Table 3, respectively. To restrict the complexity of generated sentences, we set the maximum number of logical connectives appearing in formula F_1 to 6.

Table 1 illustrates examples of fully synthetic transitivity inference datasets. We generate 3,000 Boolean inference examples $s_1 \rightarrow s_2$, 6,000 veridical inference examples $f(s_1) \rightarrow s_1$, and 6,000 composite inference examples $f(s_1) \rightarrow s_2$.

Naturalistic inference To generate a naturalistic transitivity inference dataset, we collect an example $s_1 \rightarrow s_2$ of naturalistic inference from the SICK dataset, which is constructed from existing sentences (image descriptions given by different people) and covers various lexical and structural phenomena. (1) is an example of lexical inference (*brush* → *comb*) in SICK, whose label is *yes*.

(1) s_1: A person is brushing a cat.
s_2: A person is combing the fur of a cat.

By selecting a clause-embedding verb f and an embedded sentence s_1, we generate a new sentence $f(s_1)$. As shown in (2), we construct a veridical inference example $f(s_1) \rightarrow s_1$ by setting $f(s_1)$ as a premise and s_1 as a hypothesis.

(2) $f(s_1)$: Someone **sees** that a person is brushing a cat.
s_1: A person is brushing a cat. (*yes*)

Likewise, as in (3), we can obtain a composite inference example $f(s_1) \rightarrow s_2$ whose label is *yes*:

(3) $f(s_1)$: Someone **sees** that a person is brushing a cat.
s_2: A person is combing the fur of a cat.

Table 2 illustrates examples of naturalistic transitivity inference datasets. We sample 1,000 naturalistic inference examples $s_1 \rightarrow s_2$ from the SICK training set and obtain 30,000 veridical inference examples $f(s_1) \rightarrow s_1$ and 30,000 composite inference examples $f(s_1) \rightarrow s_2$.

4 Experiments and Analysis

We analyze whether models trained with the basic inference set can consistently perform composite inferences on the test set. We use two DNN models, BERT and LSTM, which are known to perform well with linguistic phenomena such as subject-verb agreement and hierarchical and structural probing tasks (Linzen et al., 2016; Weiss et al., 2018; Kuncoro et al., 2018).

4.1 Experimental setup

In all experiments, we train each model for 25 epochs or until convergence and select the best-performing model based on its accuracy on the validation set. We perform five runs and report the average and standard deviation of their accuracies.

Data			Model					
$f(s_1)\rightarrow s_1$	$s_1\rightarrow s_2$	$f(s_1)\rightarrow s_2$	**LSTM-M**	**LSTM-B**	**LSTM-M&B**	**BERT-M**	**BERT-B**	**BERT-M&B**
yes	*yes*	***yes***	74.2 ± 2.0	89.0 ± 9.1	87.9 ± 3.7	66.3 ± 3.4	100.0 ± 0.0	100.0 ± 0.0
yes	*unk*	***unk***	16.0 ± 4.3	6.3 ± 12.8	60.0 ± 10.2	4.9 ± 1.5	0.4 ± 0.7	60.5 ± 0.6
unk	*yes*	***unk***	14.7 ± 3.8	93.4 ± 8.3	89.0 ± 9.5	12.6 ± 4.8	99.4 ± 9.0	92.9 ± 3.6
unk	*unk*	***unk***	17.8 ± 5.5	92.1 ± 7.2	99.7 ± 0.5	13.2 ± 3.4	99.5 ± 0.5	99.9 ± 0.0
Test Overall			30.9 ± 3.2	70.2 ± 3.4	84.2 ± 1.2	24.4 ± 1.6	75.7 ± 0.4	88.3 ± 0.9
Validation ($f(s_1)\rightarrow s_1$)			50.5 ± 1.7	93.3 ± 11.1	91.4 ± 5.7	68.1 ± 1.3	99.2 ± 0.2	98.3 ± 0.3
Validation ($s_1\rightarrow s_2$)			41.5 ± 3.4	89.2 ± 3.4	85.2 ± 1.2	54.4 ± 2.3	100.0 ± 0.0	99.4 ± 0.5

Table 5: Accuracies for the fully synthetic transitivity test set and the validation set. **-B** indicates a model trained with the basic inference set, **-M** indicates a model trained with MNLI, and **-M&B** indicates a model trained with MNLI mixed with the basic inference set. The label *yes* means *entailment*, and *unk* means *non-entailment*.

Data			Model					
$f(s_1)\rightarrow s_1$	$s_1\rightarrow s_2$	$f(s_1)\rightarrow s_2$	**LSTM-M**	**LSTM-B**	**LSTM-M&B**	**BERT-M**	**BERT-B**	**BERT-M&B**
yes	*yes*	***yes***	64.6 ± 12.1	97.1 ± 2.7	100.0 ± 0.1	85.9 ± 1.1	100.0 ± 0.0	100.0 ± 0.0
yes	*unk*	***unk***	45.6 ± 10.5	0.0 ± 0.0	3.6 ± 1.4	28.4 ± 0.9	8.9 ± 7.8	22.3 ± 13.6
unk	*yes*	***unk***	24.4 ± 12.1	97.1 ± 2.7	99.7 ± 0.5	13.3 ± 1.7	100.0 ± 0.0	100.0 ± 0.0
unk	*unk*	***unk***	45.4 ± 11.2	97.3 ± 2.6	99.9 ± 0.1	31.1 ± 0.9	100.0 ± 0.0	100.0 ± 0.0
Test Overall			45.0 ± 5.5	72.9 ± 2.0	75.8 ± 0.5	39.7 ± 0.2	77.2 ± 2.0	80.6 ± 3.4
Validation ($f(s_1)\rightarrow s_1$)			46.2 ± 1.2	82.1 ± 3.3	89.8 ± 6.5	68.7 ± 1.6	99.2 ± 0.0	97.1 ± 0.3
Validation ($s_1\rightarrow s_2$)			58.0 ± 1.0	81.9 ± 3.0	82.1 ± 1.4	62.0 ± 1.0	89.1 ± 2.0	91.0 ± 0.0

Table 6: Accuracies for the naturalistic transitivity test set and the validation set.

LSTM We use an LSTM (Hochreiter and Schmidhuber, 1997) model, where each premise and hypothesis is processed as a sequence of words using RNN with LSTM cells, and the final hidden state of each serves as its representation. The model concatenates the premise and hypothesis representations and passes the result to three hidden layers followed by a two-way softmax classifier. The model is initialized with 300-dimensional GloVe vectors (Pennington et al., 2014) and optimized using Adam (Kingma and Ba, 2015). We search dropout probabilities of $[0, 0.1, 0.2]$ on the output.

BERT We use the base-uncased pretrained BERT (Devlin et al., 2019) model[1], fine-tuned for the NLI classification task on training data in the standard way. When fine-tuning BERT, we search dropout probabilities of $[0, 0.1, 0.2]$ on the output, and hyperparameters are the same as those commonly used for MultiNLI.

4.2 Testing transitivity

We first evaluate whether the models trained with basic inferences $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$ can consistently make judgements on the composite inferences $f(s_1) \rightarrow s_2$. As a previous work (Ross and Pavlick, 2019) reported that a BERT model trained with the benchmark NLI dataset MultiNLI (MNLI; Williams et al., 2018) is sensitive to verb veridicality, we regard the accuracy of models trained with MNLI as a baseline. We also analyze models trained with MNLI mixed with the basic inference set.

Table 5 shows accuracies for the fully synthetic transitivity test set that combines veridical and Boolean inferences. Models trained with the basic inference set achieved over 80% accuracy on the test cases, except for cases where $f(s_1) \rightarrow s_1$ is *yes* and $s_1 \rightarrow s_2$ is *unk*. Table 6 shows accuracies for the naturalistic transitivity test set. Again, models trained with the basic inference set performed substantially below chance for the cases $f(s_1) \rightarrow s_2$, where $f(s_1) \rightarrow s_1$ is *yes* and $s_1 \rightarrow s_2$ is *unk*. This suggests that while the models achieve over 80% accuracy on both $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$ validation sets, they do not apply transitivity inference from the inferences $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$, but rather predict the label for the composite inference $f(s_1) \rightarrow s_2$ by judging whether it is similar to the veridical inference $f(s_1) \rightarrow s_1$ in the training set.

Accuracy of models trained with MNLI was low because they predicted *yes* for many examples where correct labels were *unk*, as in (4).

(4) $f(s_1)$: Someone wished that **John saw Tom** or Greg.
s_2: John saw Tom. (*unk*)

---

[1] We use the Pytorch implementation of BERT released at https://github.com/huggingface/transformers.

Type	Templates
Pronoun	At that moment, we f that s
Pronoun	Then he f that s
Specific group	The customers f that s
Specific group	Some economists f that s
Proper noun	Hanson f that s

Table 7: Examples of additional templates used for generating veridical inference datasets. Here f is a place for a veridical verb and s for an embedded sentence.

The models predicted *yes* for over 80% of the fully synthetic transitivity test set and more than 60% of the naturalistic transitivity test set. These results are consistent with the findings in McCoy et al. (2019), namely, that models trained with MNLI tend to predict entailment relations when the hypothesis is a subsequence of the premise, as in (4).

When models are trained with MNLI mixed with the basic inference set, they seem to improve performance on the fully synthetic transitivity test set. One reason for this result is that the models might use heuristics to make predictions for some *unk* examples in the fully synthetic inference set. Error analysis shows that the models tend to predict *unk* when either a premise or a hypothesis contains a negation like (5).

(5) $f(s_1)$: Someone knew that Fred praised Henry or Ann.
s_2: Fred did **not** praise Ann. (*unk*)

These heuristics might be related to the annotation artifact (Gururangan et al., 2018) in MNLI, because an inference example involving negation words tends to be a *contradiction*[2]. Moreover, models can memorize the basic inference set regardless of the existence of MNLI in the training set, so performance seems to be better.

Note that models trained with MNLI mixed with the basic inference set still failed on the naturalistic transitivity inference $f(s_1) \rightarrow s_2$ where $f(s_1) \rightarrow s_1$ is *yes* and $s_1 \rightarrow s_2$ is *unk*. Since the naturalistic basic inference examples $s_1 \rightarrow s_2$ contain various linguistic phenomena, models cannot rely on the heuristics for such examples.

Is poor performance of transitivity inference due to overfitting on verbs? To determine whether models do not overfit on clause-embedding verbs, we analyze the models under two additional settings using naturalistic transitivity datasets: (I) we use various templates other than "Someone f that s_1" to generate the main clause in $f(s_1)$, and (II) we flip the gold labels of 10% veridical inference $f(s_1) \rightarrow s_1$ instances, randomly sampled, instead of using gold labels uniquely fixed from verb types. These two complex settings expose models to more natural evaluation settings that consider the context-sensitive property of veridicality.

Data			**Model**	
$f(s_1) \rightarrow s_1$	$s_1 \rightarrow s_2$	$f(s_1) \rightarrow s_2$	**LSTM-B** (△)	**BERT-B** (△)
yes	*yes*	***yes***	97.9 (+0.8)	100.0 (0.0)
yes	*unk*	***unk***	0.0 (0.0)	2.3 (−6.6)
unk	*yes*	***unk***	99.0 (+1.9)	100.0 (0.0)
unk	*unk*	***unk***	99.2 (+1.9)	100.0 (0.0)

Table 8: Accuracies of models in the setting (I). (△) is the difference from the accuracy in Table 6.

Data			**Model**	
$f(s_1) \rightarrow s_1$	$s_1 \rightarrow s_2$	$f(s_1) \rightarrow s_2$	**LSTM-B** (△)	**BERT-B** (△)
yes	*yes*	***yes***	90.0 (−7.1)	93.6 (−6.4)
yes	*unk*	***unk***	2.2 (+2.2)	17.9 (+9.0)
unk	*yes*	***unk***	89.9 (−7.2)	94.0 (−6.0)
unk	*unk*	***unk***	98.3 (+1.0)	95.6 (+1.8)

Table 9: Accuracies of models in the setting (II). (△) is the difference from the accuracy in Table 6.

For evaluation setting (I) using various templates involving clause-embedding verbs, we manually select forty main clauses of the verb veridicality dataset (Ross and Pavlick, 2019) and provide additional templates. Table 7 shows examples of additional templates involving clause-embedding verbs used for generating veridical inference datasets.

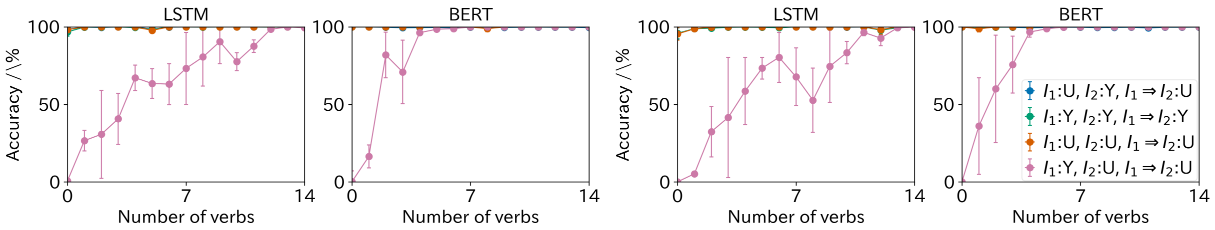
Table 8 and Table 9 show the results for (I) and (II), respectively. These results show the same trends as those in Table 6, indicating that even when we consider the complexity of veridical inference in our analysis, the models fail to consistently perform composite inferences.

4.3 Analysis with data augmentation

We further hypothesize that even if the current models fail to consistently perform composite inferences, data augmentation with a small number of composite inference examples might allow models to learn transitivity inference. Thus, we evaluate models trained with basic inferences $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$ and with a subset of the composite inferences $f(s_1) \rightarrow s_2$ on a naturalistic inference test set. Considering that models fail on composite inference $f(s_1) \rightarrow s_2$ where f

---

[2] We use binary labels (*entailment*/*non-entailment*) and take *contradiction* as *non-entailment*.


(a) $f(s_1) \rightarrow s_1$, $s_1 \rightarrow s_2$, and a subset of $f(s_1) \rightarrow s_2$

(b) $f(s_1) \rightarrow s_1$ and a subset of $f(s_1) \rightarrow s_2$

Figure 2: Accuracies of models trained with (a) and (b). I_1 indicates the first basic pattern $f(s_1) \rightarrow s_1$ and I_2 indicates the second basic pattern $s_1 \rightarrow s_2$. Y means *entailment* and U means *non-entailment*. The horizontal axis shows the number of veridical verbs for the additional training set.

Data			**Model**		**Human**
$f(s_1) \rightarrow s_1$	$s_1 \rightarrow s_2$	$f(s_1) \rightarrow s_2$	**LSTM-B**	**BERT-B**	
yes	*yes*	***yes***	100.0 ± 2.7	100.0 ± 0.0	98.8
yes	*unk*	***unk***	0.0 ± 0.0	8.9 ± 7.8	98.8
unk	*yes*	***unk***	97.1 ± 2.7	100.0 ± 0.0	44.9
unk	*unk*	***unk***	97.3 ± 2.6	100.0 ± 0.0	99.6
Test Overall			72.9 ± 2.0	77.2 ± 2.0	85.5

Table 10: Comparison between accuracies of humans and the models trained with the basic inference set.

is veridical and $s_1 \rightarrow s_2$ is *unk*, we gradually add veridical verbs (e.g., *know*) one-by-one to generate an additional training set of composite inference $f(s_1) \rightarrow s_2$ and analyze performance on a test set. Figure 2(a) shows that this data augmentation improved performance on test examples $f(s_1) \rightarrow s_2$ where f is veridical and $s_1 \rightarrow s_2$ is *unk*, while maintaining accuracy on the remaining examples in the test set. BERT achieved 100% accuracy over the entire test set by adding composite inferences generated from four veridical verbs, whereas in the case of LSTM twelve veridical verbs were needed to achieve the same accuracy.

To determine whether models augmented with composite inference examples learn the ability to combine basic inferences to perform transitivity inference, we analyze the performance of models where basic inference examples are not included in the training set. Figure 2(b) shows that models trained with only the basic inference set $f(s_1) \rightarrow s_1$ and a subset of the composite inference set $f(s_1) \rightarrow s_2$ also had improved accuracy. This result supports findings that models do not combine the basic inference $f(s_1) \rightarrow s_1$ and $s_1 \rightarrow s_2$, but rather predict the label for a composite inference $f(s_1) \rightarrow s_2$ by judging whether it is similar to inference patterns found in the training set.

4.4 Comparison with humans

To investigate how humans perform on transitivity inference tasks, we collect human judgements for a subset of our naturalistic inference dataset. We asked crowdsourced workers to label 960 transitivity inference examples involving all the clause-embedding verbs in Table 4. Following prior works involving crowdsourced NLI datasets (Zhang et al., 2017; Ross and Pavlick, 2019), we instructed raters to label each premise-hypothesis pair with the degree of entailment on a 5-point Likert scale, with 1 meaning a hypothesis is definitely not true given the premise, and 5 meaning a hypothesis is definitely true. We collected three annotations per pair on Amazon Mechanical Turk (see Appendix D for details), and the inter-rater agreement (the Pearson correlation among raters, averaged across both examples and raters) was 0.76. As model predictions are discrete (*yes* or *unk*), we discretized human scores into evenly sized bins, setting *yes* if the score was 4 or higher and set *unk* if the score was 3 or lower. We assumed the majority of three discretized labels as the final human judgement.

Table 10 shows that humans generally follow the distinction between veridical and non-veridical verbs traditionally assumed in the lexical semantics, as well as the transitivity of entailment relation. In particular, while as we saw in Section 4.2 the DNN models performed substantially below chance for transitivity inferences where $f(s_1) \rightarrow s_1$ is *yes* and $s_1 \rightarrow s_2$ is *unk*, human performance is near perfect for such inferences.

Interestingly, however, humans tend to predict incorrect labels for transitivity inferences where the verb f is non-veridical (so $f(s_1) \rightarrow s_1$ is *unk*) and the embedded inference $s_1 \rightarrow s_2$ is *yes*. This might be because a natural complement as in (6) induces veridicality bias (Ross and Pavlick, 2019), that is, no matter whether a complement verb f is veridical or non-veridical, humans tend to decide the truth value of $f(s_1)$ by judging whether its complement s_1 is true. Thus, judgement for $f(s_1) \rightarrow s_2$ coincides with that of $s_1 \rightarrow s_2$ in this case.

(6) $f(s_1)$: Someone **believed** that a man is jumping off a low wall.
s_1: A man is jumping off a low wall.
s_2: A man is jumping a wall.

5 Conclusion

We introduced an analysis method using transitivity inferences for evaluating systematic generalization capacities of NLI models. We found that current NLI models do not perform consistently well on transitivity inference tasks. Furthermore, data augmentation analysis suggested that models can memorize composite inference examples, but do not perform the intended transitivity inferences combining basic inference examples.

Overall, our results indicated that despite the impressive performance of DNN models on standard NLI datasets, there remains much room for improving their systematic generalization capacities with respect to combining basic inferential abilities on various linguistic phenomena. Regarding what is necessary for improving the systematic generalization capacity, one interesting possibility is explicitly feeding some form of logic-guided transitivity rules to models, which is left for future work. Our analysis method using transitivity can be an effective tool for further progress in the study of compositional NLI.

Acknowledgement

We thank the three anonymous reviewers for their helpful comments and suggestions. We are also grateful to Masashi Yoshikawa for helpful discussions. This work was partially supported by the RIKEN-AIST Joint Research Fund (feasibility study) and JSPS KAKENHI Grant Number JP20K19868.

References

Lasha Abzianidze. 2015. A tableau prover for natural logic and language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2492–2502.

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5642–5650.

Marco Baroni. 2020. Linguistic generalization and compositionality in modern artificial neural networks. *Philosophical Transactions of the Royal Society B*, 375(1791):20190307.

David Beaver. 2001. *Presupposition and Assertion in Dynamic Semantics*. CSLI Publications.

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

Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. 2015b. Recursive neural networks can learn logical semantics. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 12–21.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence and the 17th Pacific Rim International Conference on Artificial Intelligence (IJCAI-PRICAI)*.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Dirk van Dalen. 2013. *Logic and Structure*, 5 edition. Springer.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J. Gershman, and Noah D. Goodman. 2018. Evaluating compositionality in sentence embeddings. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 1596–1601.

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

Jerry A. Fodor and Zenon W. Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.

Atticus Geiger, Ignacio Cases, Lauri Karttunen, and Christopher Potts. 2019. Posing fair generalization tasks for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4484–4494.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 650–655.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1958–1969.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 107–112.

Irene Heim and Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Blackwell.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Philip N. Johnson-Laird and Ruth M.J. Byrne. 1991. *Deduction*. Erlbaum.

Lauri Karttunen and Stanley Peters. 1979. Conventional implicatures. In Choon Kyu Oh and David A. Dineen, editors, *Syntax and Semantics 11: Presupposition*, pages 1–56. Academic Press.

Sangeet Khemlani and Philip N Johnson-Laird. 2012. Theories of the syllogism: A meta-analysis. *Psychological bulletin*, 138(3):427–457.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1426–1436.

Brenden M. Lake and Marco Baroni. 2017. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics (TACL)*, 4:521–535.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2171–2179.

Gary Marcus. 2003. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. MIT Press.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, pages 216–223.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? the pragmatic complexity of veridicality assessment. *Computational Linguistics*, 38(2):301–333.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3428–3448.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2339–2352.

Koji Mineshima, Pascual Martínez-Gómez, Yusuke Miyao, and Daisuke Bekki. 2015. Higher-order logical inference with compositional semantics. In

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2055–2061.

Richard Montague. 1973. The proper treatment of quantification in ordinary English. In Jaakko Hintikka, Julius M. E. Moravcsik, and Patrick Suppes, editors, *Approaches to Natural Language*, pages 189–224. Reidel, Dordrecht. Reprinted in Richmond H. Thomason (ed.), *Formal Philosophy: Selected Papers of Richard Montague*, 247–270, 1974, New Haven: Yale University Press.

Yixin Nie, Yicheng Wang, and Mohit Bansal. 2019. Analyzing compositionality-sensitivity of NLI models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6867–6874.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1027–1037.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 180–191.

Kyle Richardson, Hai Hu, Lawrence S. Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Alexis Ross and Ellie Pavlick. 2019. How well do NLI models capture verb veridicality? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2230–2240.

Ohad Rozen, Vered Shwartz, Roee Aharoni, and Ido Dagan. 2019. Diversify your datasets: Analyzing generalization via controlled variance in adversarial datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 196–205.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. 2019. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4515.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4911–4921.

Anne S. Troelstra and Helmut Schwichtenberg. 2000. *Basic Proof Theory*, 2 edition. Cambridge University Press.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of Advances in Neural Information Processing Systems 32 (NIPS)*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision RNNs for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 740–745.

Aaron Steven White and Kyle Rawlins. 2018. The role of veridicality and factivity in clause selection. In *Proceedings of the 48th Annual Meeting of the North East Linguistic Society, Amherst, MA, USA. GLSA Publications*.

Aaron Steven White, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2018. Lexicosyntactic inference in neural models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4717–4724.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1112–1122.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6105–6117.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019a. Can neural networks understand monotonicity reasoning? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 31–40.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019b. HELP: A dataset for identifying shortcomings of neural models in monotonicity reasoning. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 250–255.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics (TACL)*, 5:379–395.

A Details about the Boolean logic fragment

Table 11 shows the context-free grammar used to generate sentences for Boolean logic reasoning with conjunction, disjunction, and negation. Each rewriting rule is paired with the corresponding semantic composition rule in standard Montagovian semantics to generate the logical form of a sentence (Montague, 1973; Heim and Kratzer, 1998). We use ten items each for proper names (PN), intransitive verbs (IV), and transitive verbs (TV). Each sentence is generated with a verb in the past tense.

For sentences with multiple NPs, we assume the surface-scope reading where the subject NP takes scope over the object NP. For instance, the sentence *Ann and Bob saw Chris or Daniel*, where the subject NP is conjunctive and the object NP is disjunctive, has the logical form $(\mathbf{see}(\mathbf{ann}, \mathbf{chris}) \vee \mathbf{see}(\mathbf{ann}, \mathbf{daniel})) \wedge (\mathbf{see}(\mathbf{bob}, \mathbf{chris}) \vee \mathbf{see}(\mathbf{bob}, \mathbf{daniel}))$.

There are two types of negation, sentential negation (SNEG) and verbal negation (VNEG), which are distinguished with respect to their scope interpretation. Thus, the sentence *Ann and Bob did not swim* has the logical form $\neg\mathbf{swim}(\mathbf{ann}) \wedge \neg\mathbf{swim}(\mathbf{bob})$, while the sentence *It is not the case that Ann and Bob did not swim* has the logical form $\neg(\mathbf{swim}(\mathbf{ann}) \wedge \mathbf{swim}(\mathbf{bob}))$.

To generate a premise-hypothesis pair (s_1, s_2) using this Boolean logic fragment, we first generate a sentence s_1 and derive its logical form F_1 using the grammar in Table 11. We then randomly select one of the atomic formulas appearing in F_1, say A, and takes its positive (A) or negative ($\neg A$) form, which is in turn converted to the hypothesis sentence s_2 using the same grammar. The gold label (*entailment* or *non-entailment*) for the pair (s_1, s_2) is determined by checking whether F_1 logically entails A or $\neg A$ using a first-order-logic theorem prover[3].

B Training details

In all experiments, we trained models on eight NVIDIA DGX-1 Tesla V100 GPUs. The runtime for training each model was about 1-8 hours, depending on the size of the training set. The order of training instances was shuffled for each model.

C Supplementary results on the random train-test split

To confirm that our transitivity inference dataset is not excessively difficult, we conducted additional experiments using the random $9:1$ train:test split of transitivity inference ($f(s_1) \rightarrow s_2$) datasets. We evaluate models under two settings: (i) models trained with the train split of transitivity inference datasets and (ii) models trained with the train split mixed with MNLI. Table 12 shows the results on the random train-test split of our full-synthetic transitivity dataset, and Table 13 shows the results on the random train-test split of our naturalistic transitivity dataset. These results showed that regardless of the existence of MNLI in the training set, models achieved perfect performance on our transitivity inference test set with the standard random train-test split setting.

D Human judgement details

Using Amazon Mechanical Turk, we collected human judgements for 960 naturalistic veridical inference examples and 960 naturalistic transitivity inference examples. We required raters to have completed at least 5,000 approved tasks to maintain a 99% approval rating. Raters could indicate by a checkbox that one or both sentences did not make sense, but no rater clicked the checkbox. We collected three annotations per pair and paid $0.06 per labeled pair.

Since humans predict incorrect labels for some composite inference examples $f(s_1) \rightarrow s_2$ where the verb f is non-veridical, we checked the accuracy of human judgement on a set of premise-hypothesis pairs $f(s_1) \rightarrow s_1$ and $f(s_1) \rightarrow s_2$ involving each non-veridical verb, as shown in Table 14. Annotators tended to incorrectly make judgements for both $f(s_1) \rightarrow s_1$ and $f(s_1) \rightarrow s_2$. Regarding accuracy for each non-veridical verb, annotators correctly drew inferences containing *wish* and *hope*, while they tended to draw inferences containing *claim* and *hear* incorrectly.

In comparison with the previous veridicality dataset MegaVeridicality2 (White et al., 2018), the accuracy tended to be lower than that in MegaVeridicality2[4]. As (7) shows, while a simple complement is used for MegaVeridicality2, a natural complement like (8) might induce veridicality

---

[3] https://github.com/vprover/vampire

[4] We calculated the percentage of the majority judgement for each verb for ten different annotations in MegaVeridicality2.

Syntax	Semantics
$\text{S} \rightarrow \text{NP VP}_{\text{past}}$	$[\![\text{S}]\!] = [\![\text{NP}]\!]([\![\text{VP}_{\text{past}}]\!])$
$\text{S} \rightarrow \text{SNEG S}$	$[\![\text{S}]\!] = [\![\text{SNEG}]\!]([\![\text{S}]\!])$
$\text{NP} \rightarrow \text{PN}$	$[\![\text{NP}]\!] = [\![\text{PN}]\!]$
$\text{NP} \rightarrow \text{PN CON PN}$	$[\![\text{NP}]\!] = \lambda F.[\![\text{CON}]\!]([\![\text{PN}]\!](F), [\![\text{PN}]\!](F))$
$\text{NP} \rightarrow \text{PN , PN , CON PN}$	$[\![\text{NP}]\!] = \lambda F.[\![\text{CON}]\!]([\![\text{PN}]\!](F), [\![\text{CON}]\!]([\![\text{PN}]\!](F), [\![\text{PN}]\!](F)))$
$\text{VP}_{tense} \rightarrow \text{IV}_{tense}$	$[\![\text{VP}_{tense}]\!] = [\![\text{IV}_{tense}]\!]$
$\text{VP}_{tense} \rightarrow \text{TV}_{tense}\ \text{NP}$	$[\![\text{VP}_{tense}]\!] = \lambda x.[\![\text{NP}]\!](\lambda y.[\![\text{TV}_{tense}]\!](x, y))$
$\text{VP}_{\text{past}} \rightarrow \text{VNEG VP}_{\text{base}}$	$[\![\text{VP}_{\text{past}}]\!] = \lambda x.[\![\text{VNEG}]\!]([\![\text{VP}_{\text{base}}]\!](x))$
$\text{PN} \rightarrow \texttt{Ann} \mid \texttt{Bob} \mid \texttt{Chris} \mid \cdots$	$[\![\text{PN}]\!] = \lambda F.F(\textbf{sym})$
$\text{IV}_{\text{base}} \rightarrow \texttt{swim} \mid \texttt{drink} \mid \texttt{smoke} \mid \cdots$	$[\![\text{IV}_{\text{base}}]\!] = \lambda x.\textbf{sym}(x)$
$\text{IV}_{\text{past}} \rightarrow \texttt{swam} \mid \texttt{drank} \mid \texttt{smoked} \mid \cdots$	$[\![\text{IV}_{\text{past}}]\!] = \lambda x.\textbf{sym}(x)$
$\text{TV}_{\text{base}} \rightarrow \texttt{see} \mid \texttt{visit} \mid \texttt{touch} \mid \cdots$	$[\![\text{TV}_{\text{base}}]\!] = \lambda y \lambda x.\textbf{sym}(x, y)$
$\text{TV}_{\text{past}} \rightarrow \texttt{saw} \mid \texttt{visited} \mid \texttt{touched} \mid \cdots$	$[\![\text{TV}_{\text{past}}]\!] = \lambda yx.\textbf{sym}(x, y)$
$\text{SNEG} \rightarrow \texttt{it is not the case that}$	$[\![\text{SNEG}]\!] = \lambda P.\neg P$
$\text{VNEG} \rightarrow \texttt{did not}$	$[\![\text{VNEG}]\!] = \lambda P.\neg P$
$\text{CON} \rightarrow \texttt{and}$	$[\![\text{CON}]\!] = \lambda P \lambda Q.P \wedge Q$
$\text{CON} \rightarrow \texttt{or}$	$[\![\text{CON}]\!] = \lambda P \lambda Q.P \vee Q$

Table 11: Grammar for the Boolean logic fragment with semantic composition. Feature *tense* for VP is either "base" or "past." In semantic composition, **sym** is the place where the symbol (lemma) for a lexical item appears.

Data			**Model**			
$f(s_1) \rightarrow s_1$	$s_1 \rightarrow s_2$	$f(s_1) \rightarrow s_2$	**LSTM-T**	**LSTM-M&T**	**BERT-T**	**BERT-M&T**
yes	*yes*	***yes***	99.8 ± 0.1	100.0 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
yes	*unk*	***unk***	99.3 ± 0.0	99.7 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
unk	*yes*	***unk***	99.4 ± 0.2	99.8 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
unk	*unk*	***unk***	99.6 ± 0.1	100.0 ± 0.1	100.0 ± 0.0	100.0 ± 0.0

Table 12: Accuracies on the random train-test split of our fully synthetic transitivity dataset. **-T** indicates a model trained with the train split of the transitivity inference set, and **-M&T** indicates a model trained with MNLI mixed with the train split.

Data			**Model**			
$f(s_1) \rightarrow s_1$	$s_1 \rightarrow s_2$	$f(s_1) \rightarrow s_2$	**LSTM-T**	**LSTM-M&T**	**BERT-T**	**BERT-M&T**
yes	*yes*	***yes***	99.2 ± 0.0	100.0 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
yes	*unk*	***unk***	98.4 ± 0.2	99.1 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
unk	*yes*	***unk***	98.3 ± 0.2	99.3 ± 0.1	100.0 ± 0.0	100.0 ± 0.0
unk	*unk*	***unk***	99.6 ± 0.1	100.0 ± 0.1	100.0 ± 0.0	100.0 ± 0.0

Table 13: Accuracies on the random train-test split of our naturalistic transitivity test set.

bias (Ross and Pavlick, 2019), resulting in incorrect judgements on veridical inference. Whether a verb is veridical or non-veridical, humans tend to judge the complement as true.

(7) $f(s_1)$: Someone **believed** that something happened.

s_1: Something happened.

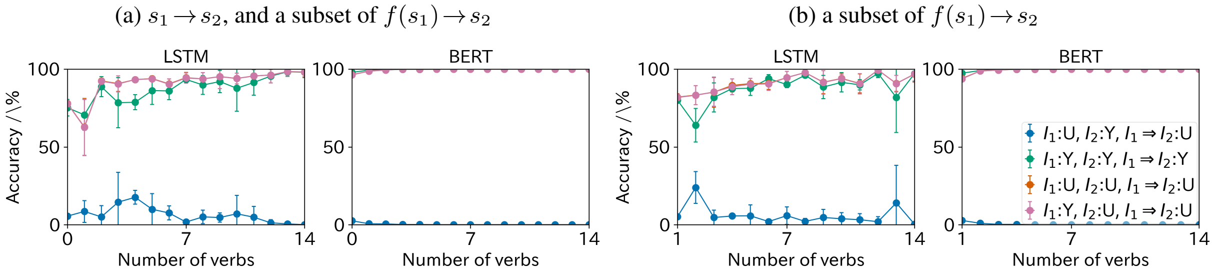
(8) $f(s_1)$: Someone **believed** that a man is jumping off a low wall.

s_1: A man is jumping off a low wall.

s_2: A man is jumping a wall.

E Supplementary results with data augmentation

In Section 4.3, we gradually added a subset of the composite inferences $f(s_1) \rightarrow s_2$ involving a veridical verb (e.g., *know*) to the training set and evaluated the performance of models on a naturalistic inference test set. We also evaluated the performance of models under two conditions: (a) models trained with the basic inference set $s_1 \rightarrow s_2$ and a subset of the composite inference set $f(s_1) \rightarrow s_2$ and (b) models trained with a subset of the composite inference set $f(s_1) \rightarrow s_2$. Figure 3(a) shows that the models significantly improved accuracy on composite inferences except for the test example $f(s_1) \rightarrow s_2$, whose label dif-


(a) $s_1 \rightarrow s_2$, and a subset of $f(s_1) \rightarrow s_2$

(b) a subset of $f(s_1) \rightarrow s_2$

Figure 3: Accuracies of models trained with (a) and (b). I_1 is the first basic pattern $f(s_1) \rightarrow s_1$ and I_2 is the second basic pattern $s_1 \rightarrow s_2$. Y indicates *entailment* and U indicates *non-entailment*. The horizontal axis shows the number of veridical verbs for the additional training set.

Verb	$f(s_1) \rightarrow s_1$	$f(s_1) \rightarrow s_2$	**MegaV2**
argue	34 (-46)	66 (-14)	80
assume	70 (-15)	79 (-6)	85
believe	19 (-71)	59 (-31)	90
claim	15 (-65)	56 (-24)	80
doubt	91 (+9)	96 (+16)	80
estimate	35 (-50)	64 (-21)	85
expect	53 (-27)	70 (-10)	80
feel	42 (-53)	67 (-28)	95
hear	14 (-41)	53 (-2)	55
hope	77 (-8)	92 (+7)	85
imply	18 (-47)	58 (-7)	65
predict	50 (-25)	73 (-2)	75
suspect	48 (-47)	79 (-16)	95
think	18 (-77)	57 (-38)	95
wish	77 (+7)	92 (+22)	70

Table 14: Accuracy (%) of human judgements for each non-veridical verb. MegaV2 indicates the percentage of those annotators who judge each verb to be non-veridical in MegaVeridicality2 (White et al., 2018). A number in parentheses is a difference from the accuracy in MegaVeridicality2.

fered from that of $s_1 \rightarrow s_2$. Moreover, their performance was maintained even without composite inference examples in the training set. This indicates that models predict labels for the composite inference example only by judging whether it is similar to the basic inference example in the training set.

Figure 3(b) shows the result when models are trained only with a subset of the composite inference set $f(s_1) \rightarrow s_2$. As non-veridical verbs are not included in the training set in this setting, the models predict labels for composite inferences involving non-veridical verbs by judging whether they are similar to composite inferences involving veridical verbs in the training set. The models thus fail on composite inference examples $f(s_1) \rightarrow s_2$ where f is non-veridical and $s_1 \rightarrow s_2$ is *yes*. The labels of such non-veridical inference examples are opposite to those of veridical inference examples in the training set.

A Neural Few-Shot Text Classification Reality Check

Thomas Dopierre[1][2], **Christophe Gravier**[1], **Wilfried Logerais**[2]

[1]Laboratoire Hubert Curien
UMR CNRS 5516
Université Jean Monnet
Saint-Étienne, France
`firstname.lastname@univ-st-etienne.fr`

[2]Meetic
Paris, France
`{t.dopierre,w.logerais}@meetic-corp.com`

Abstract

Modern classification models tend to struggle when the amount of annotated data is scarce. To overcome this issue, several neural few-shot classification models have emerged, yielding significant progress over time, both in Computer Vision and Natural Language Processing. In the latter, such models used to rely on fixed word embeddings before the advent of transformers. Additionally, some models used in Computer Vision are yet to be tested in NLP applications. In this paper, we compare all these models, first adapting those made in the field of image processing to NLP, and second providing them access to transformers. We then test these models equipped with the same transformer-based encoder on the intent detection task, known for having a large number of classes. Our results reveal that while methods perform almost equally on the ARSC dataset, this is not the case for the Intent Detection task, where the most recent and supposedly best competitors perform worse than older and simpler ones (while all are given access to transformers). We also show that a simple baseline is surprisingly strong. All the new developed models, as well as the evaluation framework, are made publicly available[1].

1 Introduction

Text classification often requires a large number of mappings between texts and target classes, so that it is challenging to build few-shot text classification models (Geng et al., 2019). With the recent advances of transformer-based models (Devlin et al., 2018; Wolf et al., 2019) along with their fine-tuning techniques (Sun et al., 2019), text classification has significantly improved. In few-shot settings, methods based on these extracted text representations have been historically made of semi-supervision, especially thanks to pseudo-labeling (Blum and Mitchell, 1998; Mihalcea, 2004; Zhi-Hua Zhou and Ming Li, 2005), which aims at propagating known labels to unlabeled data points in the representational space. Such methods depend on the number of collected unlabeled data, which can also be costly to obtain (Charoenphakdee et al., 2019), and also suffer from the infamous pipeline effect in NLP (Tenney et al., 2019), as cascade processing tends to make errors accumulate. In order to address the hindrance of collecting unlabeled data, modern approaches include unsupervised data augmentation techniques (Xie et al., 2019). It consists of generating samples through well-established text augmentation techniques in Neural Machine Translation, such as backtranslation (Sennrich et al., 2015; Edunov et al., 2018), and then use a consistency loss, training the classifier to assign the same prediction to all variations of the same sample text. While collecting new pseudo-labels can therefore be overcome by manipulating the dataset (especially using data augmentation techniques), the pipelining error accumulation effect instead calls for new neural architectures supporting scarcity of labeled data in an end-to-end fashion. Such end-to-end few-shot neural architectures for few-shot classification were discovered in image processing – it includes Matching Networks (Vinyals et al., 2016), Prototypical Networks (Snell et al., 2017) plus a follow-up known as Prototypical Networks++ (Ren et al., 2018), and Relation Networks (Sung et al., 2018). Ultimately Induction Networks (Geng et al., 2019) is a meta-learning based method dedicated to few-shot text classification, supposedly the state-of-the-art. Since our contribution considers this family of models, we will further detail them in Section 2. Nonetheless, it is important to stress that most of these neural architectures were originally devised to integrate image feature extractors. Despite both text and image relying on features extractors, a paragraph or sentence of few words hardly convey

---

[1] https://github.com/tdopierre/FewShotText

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 935–943
April 19 - 23, 2021.

as much information as a full-fledged three-canals 600×400 image ($720,000$ numerical values intrinsically). It is therefore of the utmost practical interest to validate and compare if what works best for end-to-end few-shot *image* classification is the same for end-to-end few-shot *text* classification. Moreover, when applying these end-to-end few-shot models to text, two main system components are into action: the text feature extractor itself and the downstream part of the neural network that provides a learning strategy over few shots. If we want to compare these systems, we need to plug the same feature extractor (hopefully the best one, that is transformer-based currently) into each end-to-end model. For the time being, the literature on end-to-end few-shot text classification compare aforementioned techniques using a different text extractor for each system, which is the one available when the technique was discovered – these text encoding varying greatly (Section 3.3). From that point-of-view, it is hardly possible to conclude if the improvement over time in few-shot text classification is due to new few-shot learning techniques or plainly to the significant advances made by text feature extractors. The same applies to vectors metrics: one method can use the cosine and another the euclidean distance, and that choice alone can impact conclusions made on the method being the state-of-the-art, although it could well rely only on the metric at work. Ultimately, experimental setups are usually restricted to one dataset, and evaluation schemes are heterogeneous among papers (Yu et al., 2018a),

- We revise different end-to-end neural architectures for few-shot text classification using the *same* transformer-based feature extractor,
- We investigate how these re-implemented state-of-the-art solutions compete with very simple baselines found to be yet very competitive for few-shot classification in the field of image-processing,
- We introduce an evaluation framework based on a number of intent detection datasets which is significantly bigger than what is usually used as evaluation in seminal papers transposing each of these architectures from image to text classification,
- The entire framework used in this paper, including all the re-implemented methods plugged with up-to-date transformers, is provided as an open-source repository for further research.

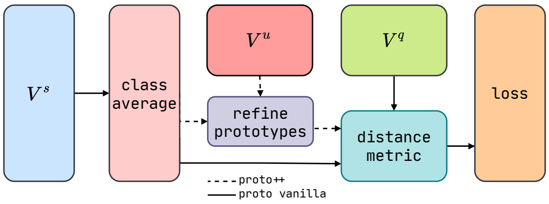
In a nutshell, we will demonstrate that providing a transformer-based encoder to a previously obsolete few-shot technique makes it the state-of-the-art again, that standard baselines are surprisingly strong, and that Induction Networks, while performing well for binary sentiment classification, struggles to perform correctly in the most common setups of few-shot text classification.

2 Few-Shot Classification Methods

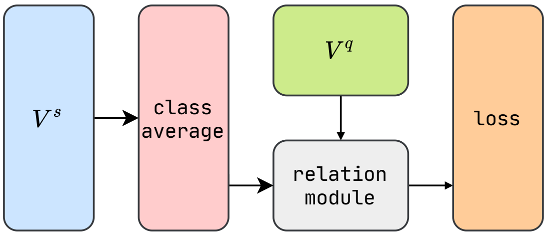
In this section, we will describe the few-shot learning methods. In the following section, sentence vectors derived from the sentence encoder are denoted v. V^s, V^q, and V^u represent vectors for support, query, and unlabeled points, respectively. The number of shots is denoted K, and the number of classes per episode is denoted C. The kth support vector of class c is denoted $v^s_{c,k}$. In the equations, $s^q_{i,j}$ (resp. $s^q_{i,c}$) will denote the similarity between the ith query vector and the jth support vector (resp. the cth class). Similarly, $s^u_{i,j}$ represents the similarity between the ith unlabeled sample and the jth support vector. When needed, the number of unlabeled data is denoted U. For each method relying on a given similarity or distance metric, we devise two experiments, using either the cosine similarity or the euclidean distance. Those additional experiments are crucial, as they allow us to compare methods directly, without introducing a metric choice bias. Architectures of the different few-shot approaches are illustrated in Figure 1. They are each detailed in Section 2.2 and onwards, yet we first introduce the common building blocks among all methods in what follows.

2.1 Common building blocks

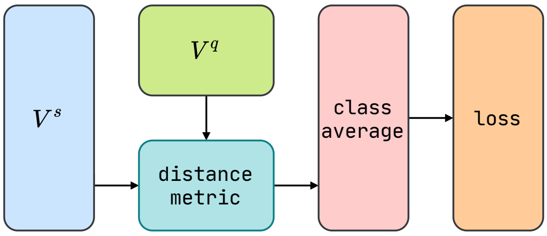
Class average All Matching, Prototypical, and Relation Networks contain a *class average* block. This step is used to directly compare a query point to a given class in order to make a prediction for this query point. In both Matching and Prototypical Networks, this step averages embeddings of support points for each class (they are class `prototypes`), which are then compared to query points to output class probabilities. In Matching Networks, this block averages similarity scores class-wise. On the contrary, in Induction Networks,



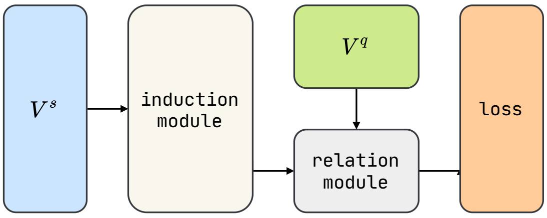
(a) Prototypical Network, with the optional proto++ step. In the original Prototypical Networks, the euclidean distance is used as distance metric.



(b) Relation Network



(c) Matching Network. In the original Matching Networks, the cosine similarity is used as a distance metric.



(d) Induction Network

Figure 1: Few-Shot classification methods and variants used in our experiments.

this step is lacking as support points are converted into prototypes using an Induction Layer, which aims at finding a better way to aggregate such knowledge than using the average (Section 2.5).

Loss Matching and Induction Networks both use the mean squared error (MSE) loss. Other methods use cross-entropy (CE). We implemented both losses on Matching and Induction networks, and it leads to very similar results – and sometimes, slightly better using CE. We therefore report results for all models using CE due to space limitations. Note that both losses are available in the publicly available source code. The cosine similarity being bounded, it would not make sense to directly apply such a loss on cosine similarities. To overcome this issue, we multiply the cosine similarities by a constant factor of 5, allowing them to reach more extreme values, hence ensuring that probabilities obtained by softmax are sparse enough.

2.2 Matching Networks

Introduced by Vinyals et al. (2016), Matching Networks (Figure 1c) rely on the comparison between query and support vectors using the cosine similarity in the seminal paper. After similarities between a query point and all support points are computed, they are averaged for each class. The predicted label for a given query point is the one with the highest average cosine similarity. In our notation framework, this process is summed up in Equation 1.

$$s_{i,c}^{q,\text{matching}} = K^{-1}\sum_{k=1}^{K}\frac{\left(v_i^q\right)^T v_{c,k}^s}{\|v_i^q\|_2\|v_{c,k}^s\|_2} \qquad (1)$$

2.3 Prototypical Networks

Prototypical Networks (Figure 1a) were introduced by Snell et al. (2017) as an extension of Matching Networks. After obtaining support vectors from the encoder, a class-wise average operation is done, as in Equation 2. This results in C prototypes denoted $\{p_c, c \in [\![1, C]\!]\}$, each one being the representative of a class. Then, a distance metric compares all query points to all prototypes. For each query point, the predicted class is the one for which this distance is the smallest. In the original Prototypical Networks, the euclidean distance was used, as in Equation 3. We also add the cosine similarity-based distance in our experiments in order to measure the impact of selecting another distance metric.

$$p_c = K^{-1}\sum_{k=1}^{K} v_{c,k}^s \qquad (2)$$

$$s_{i,c}^{q,\text{proto}} = \frac{\exp\left(-\|v_i^q - p_c\|_2^2\right)}{\sum_{c'=1}^{C}\exp\left(-\|v_i^q - p_{c'}\|_2^2\right)} \qquad (3)$$

An extension to Prototypical Networks was proposed by Ren et al. (2018), where unlabeled data points are used along with support and query points. After computing each class's prototype, a soft k-means technique is applied to further refine those prototypes using unlabeled data points. The refined prototypes, denoted $\tilde{p}_c$, are derived using Equation 4. This additional step aims at correcting the support points selection bias and making the method more robust.

$$\tilde{p}_c = \frac{\sum_{k=1}^{K} v_{c,k}^{s} + \sum_{i=1}^{U} v_i^u s_{i,c}^{u,\text{proto}}}{K + \sum_i s_{i,c}^{u,\text{proto}}} \quad (4)$$

2.4 Relation Networks

Relation Networks (Sung et al., 2018) challenge the idea of using a pre-defined metric. The Relation Module takes as an input a query vector $v_i^q \in \mathbb{R}^d$, and the prototype of a class $p_c \in \mathbb{R}^d$, the latter being obtained the same way as in Prototypical Networks (Equation 2). The idea is to use a relation module, modeling the relationship between those two vectors, yielding a similarity score $s_{i,c} \in (0,1)$. Instead of using a pre-defined distance metric like the euclidean or the cosine one, this approach allows such networks to learn this metric by themselves. Two different relation module architectures exist.

base The `base` relation module concatenates both v_i^q and p_c, and applies a small feed-forward neural network composed of two linear layers, with a ReLU activation function in between. The formula for this given relation module is described in Equation 5, where $C(\cdot,\cdot)$ denotes the concatenation operator, $f(\cdot)$ denotes the ReLU activation function, and w, M_1, M_2 are learnable parameters.

$$s_{i,c}^{q,\text{rel-base}} = \langle w, M_2 (f (M_1 (C (v_i^q, p_c))))\rangle \quad (5)$$

NTL Introduced by Socher et al. (2013), the **N**eural **T**ensor **L**ayer relation module uses intermediate learnable matrices $M_k \in \mathbb{R}^{d,d}$ to model the relation between support vectors and prototypes. The similarity score for this relation module is obtained using Equation 6, where w is a learnable parameter. Following the work done by Geng et al. (2019), we fix the number h of intermediate matrices to 100 in all our experiments.

$$s_{i,c}^{q,\text{rel-ntl}} = \left\langle w, z_{i,c}^{\text{rel-ntl}} \right\rangle \quad , \quad w \in \mathbb{R}^h \quad (6)$$

$$z_{i,c,t}^{q,\text{rel-ntl}} = f\left((v_i^q)^T M_t p_c \right) \quad , \quad t \in [\![1, h]\!] \quad (7)$$

2.5 Induction Networks

Induction Networks (Geng et al., 2019) aims at finding a general representation of each class in the support set to compare to new queries. They are composed of both an induction module and a relation module. The main motivation for such networks is that representing the class by the average vector of its data points – what is done in Prototypical and Relation networks – is too restrictive. The first part, the induction module, leverages a dynamic routing (Sabour et al., 2017) algorithm. In their contribution, Geng et al. (2019) show that their method can better *induce* (hence their name) and generalize class-wise representations. For the second part, an NTL Relation Module is used: this is the same as the one introduced earlier (Section 2.4). Such networks are illustrated in Figure 1d.

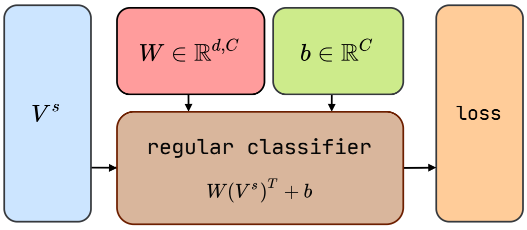
As in (Geng et al., 2019), we fix the number of routing iterations to 3, and the number of matrices in the NTL to 100.

2.6 Classifier Baselines

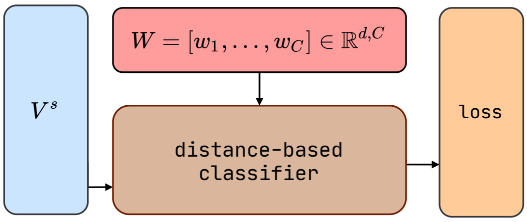
Few-shot learning algorithms are designed to overcome the data scarcity problem. With the tremendous shift in the architecture of sentence encoders using transformers, control baselines are needed to validate their ability to learn from few samples. For this reason, we include as a first `Baseline` model a traditional classifier, as described by Equation 8, added on top of BERT. Both W and b are learnable parameters, fine-tuned on the support vectors V^s. In our experiments, this method will henceforth be referred to as `Baseline`.

$$s_{i,j}^{q,\text{baseline}} = (W v_i^q + b)_j \quad ; \quad W \in \mathbb{R}^{C,d} \quad (8)$$

In addition to this `Baseline` model, we also implement a variant of it, which will henceforth be referred to as `Baseline++`. In that second baseline, the classifier design differs as follows: it measures similarities to a learnable vector instead of transforming vectors into logits using a linear layer. The matrix W used in the `Baseline` model can be writen as $[w_1, \ldots, w_C]$ where each $w_k \in \mathbb{R}^d$ is a weight vector corresponding to the k^{th} class. To measure the similarity between class j and a query vector v_i^q, we compute the similarity scores in Equation 9. After all scores $s_{\cdot,\cdot}$ are computed, we then obtain a probability vector through



(a) Baseline Network.



(b) Baseline++ Network. The distance-base classifier can either use cosine or euclidean distance

Figure 2: Few-Shot classification baselines used in our experiments.

normalization using the softmax function).

$$s_{i,j}^{q,\text{baseline++}} = \frac{w_j^T v_i^q}{\|w_j\|_2 \|v_i^q\|_2} \tag{9}$$

As in Prototypical Networks, the derived vectors $[w_1, \ldots, w_C]$ can be interpreted as class prototypes. For both baselines, at each training episode, the weights W and b are initialized, and the whole model is fine-tuned for a few iterations using support samples. This is important in practice, as it teaches the sentence encoder – a transformer, see Section 3.3 – how to produce good enough embeddings for the downstream classifier to learn efficiently. At test time, the same process is used – using test labels –, except that we freeze the encoder's weights and only fine-tune the classifier part. The baselines architectures are represented in Figure 2.

3 Experimental Setup

3.1 Few-Shot Evaluation Setup

Introduced by Vinyals et al. (2016), few-shot classification corresponds to the case when a classifier must adapt to new classes, denoted here as $\mathcal{C}_{test}$, unseen during training, and only given a few labeled examples of these new classes. To this end, the approaches assume that during training, a task-significant set of classes noted $\mathcal{C}_{train}$ is available, along with an accordingly task-significant number of labeled data for each class $c_{train_i} \in \mathcal{C}_{train}$. For each training episode, C classes are sampled from $\mathcal{C}_{train}, C \ll |\mathcal{C}_{train}|$. Then, K support examples and Q query examples are randomly drawn for each of these classes. The model is then iteratively trained using both query and support points.

At testing time, the same sampling strategy is made, this time drawing classes among $\mathcal{C}_{test}$, with $\mathcal{C}_{test} \cap \mathcal{C}_{train} = \emptyset$. The model is then evaluated on its ability to predict labels for the Q query samples, using the K support samples (unless otherwise stated, C, Q, and K values are the same at both testing and training time).

This training procedure is called C-way K-shot classification. In all our experiments, we used $K = Q = 5$. Concerning the value of C, it is fixed to 2 for ARSC, as this dataset is already composed of binary classification tasks. Regarding the intent detection datasets we introduce later (Section 3.2), in order to see the shift between ARSC binary tasks and the more common 5-way evaluation (Geng et al., 2019; Ren et al., 2018), we measured performances of the different models with C ranging from 2 to 5.

3.2 Datasets

In this section, we describe the datasets used in our evaluation framework. The first one is a popular sentiment classification dataset, while the others are intent detection datasets. All datasets are public and in English.

ARSC The **A**mazon **R**eview **S**entiment **C**lassification dataset (Blitzer et al., 2007) is composed of product reviews from 27 product categories. Each review belongs to one of the 27 domains, and contains a grade ranging from 1 to 5 stars. The usual setup (Yu et al., 2018b; Geng et al., 2019) to evaluate few-shot classification with this dataset is as follows: for each of $p \leq 27$ product category and $2 \leq t \leq 5$ score thresholds, $\mathcal{E}_{ARSC} = p \times t$ binary classification evaluation tasks are created. In each of these $p \times 4$ tasks, a competitor model must learn to classify negative ($< t$) and positive ($\geq t$) reviews. To build our test tasks, we consider the same product categories as previous works (Yu et al., 2018b; Geng et al., 2019), which are *Books*, *DVD*, *Electronics*, *Kitchen*, and $t = 3$ (thresholds are picked in the $\{2, 4, 5\}$ set) – hence 12 binary classification test tasks in our benchmark for this dataset.

Each of these twelve evaluation tasks comes with a number of support test samples ($K = Q = 5$ as stated previously). Nonetheless, in (Yu et al., 2018b) the same 5 samples per testing class are fixed for all experiments[2], which leads to a significant selection bias towards these 5 randomly selected samples used throughout the evaluation. In order to get more consistent results, we ran additional experimental runs, each of them selecting randomly new support samples. In the ARSC result table (table 1), this corresponds to the last column (`BERT + Sample shots`).

OOS The **O**ut **O**f **S**cope dataset[3] (Larson et al., 2019) is an intent detection dataset containing 150 equally-distributed classes. While initially used for out-of-scope prediction, it was also motivated by a high number of classes, a low number of examples per class (150), and its chatbot life-like style. In our experiments, we discard the out-of-scope class, keeping the remaining 150 classes to work with.

Liu Introduced by Liu et al. (2019), this intent detection dataset consists in 54 classes. This dataset was collected on the Amazon Mechanical Turk platform, where workers were given an intent and had to formulate queries for this intent with their own words. It is highly imbalanced: the most common class (`query`) holds 5, 920 samples while the least common one (`volume_other`) 24 samples.

TREC28 TREC[4] is an open-domain fact-based dataset for question classification. We use the 50 labels version of the dataset but remove the labels which have less than 40 samples. This filtering process yields a dataset with 28 classes, ranging from 40 to 962 samples per class.

3.3 Sentence Encoder

In previous works comparing few-shot text classification methods, sentence encoders were not always the same. For example, Yu et al. (2018b) use a CNN on top of word embeddings, while Geng et al. (2019) use a Bi-LSTM. Those differences make the results hard to compare since they do not use the same method to convert sentences into vectors. In our experiments, in order to reduce this selection bias, and since it is now the state-of-the-art in many applications, we use a BERT (Devlin et al., 2018) encoder, using models from the Hugging Face (Wolf et al., 2019) team.

For each dataset, instead of using an off-the-shelf pre-trained model, we fine-tune it on the masked language modeling task, as it greatly improves the quality of embeddings (Sun et al., 2019; Xie et al., 2019). This fine-tuned transformer is then used as input for all few-shot models.

4 Observations

We report results for the ARSC dataset in Table 1, and results for the Intent Detection tasks in Table 3.

4.1 Baselines are surprisingly strong

Few-shot learning methods were originally used to overcome data scarcity. In those situations, training a classifier on top of a small dataset – in our case, 5 samples per class – can be hard. However, our experiments on ARSC show that the `Baseline` and `Baseline++`, plain and simple classifiers, get surprisingly close to state-of-the-art results. Table 2 provides four correct and four incorrect classification examples for the `Baseline` model.

While it fails to predict the correct text label for some shots, it is also able to correctly classify sentences such as *What do I take home ?* among the 50 test classes of the OOS dataset. On the ARSC dataset, it is also important to note that the `Baseline++` model is significantly better than the `Baseline`, and is even on par with all other architectures, except `Prototypical Networks`.

4.2 Sample selection bias

The mean accuracy difference between the last and the second columns of Table 1 accounts for the difference of randomly selecting new support samples at each iteration (last column) as opposed to picking the same fixed pool of support samples as done previously (second to last column). We can see that this difference alone is in the range of the increments brought by each model over time (baselines aside, bringing from 1 point up to 2.6 points for Prototypical Networks). This huge gap shows the importance of using evaluation tricks like cross-validation, instead of evaluating only for one run over a fixed set of shots.

4.3 Impact of switching to transformers

One of the main contributions of our paper is to compare few-shot learning methods with the lowest bias possible (see Section 3.3). On the ARSC dataset, using transformers drastically changes the

---

[2] See labeled sampled in `https://github.com/Gorov/DiverseFewShot_Amazon`

[3] `https://github.com/clinc/oos-eval`

[4] `https://trec.nist.gov/data/qa.html`

Model	Configuration		Mean binary accuracy		
	Metric	Relation module	Original encoder †	BERT as encoder (↗ or ↘ w.r.t. original encoder)	BERT + Sample shots
Matching Network (Vinyals et al., 2016)	euclid.	N/A	–	81.2	82.9
	cosine	N/A	65.7	81.9 (↗)	83.3
Prototypical Network (Snell et al., 2017)	euclid.	N/A	68.2	80.0 (↗)	82.6
	cosine	N/A	–	81.7	83.5
Proto++ (Ren et al., 2018)	euclid.	N/A	📷	82.4	**84.0**
	cosine	N/A	📷	**82.6**	83.6
Relation Network (Sung et al., 2018)	N/A	base	–	81.0	82.9
	N/A	ntl	83.1	81.7 (↘)	83.3
Induction Network (Geng et al., 2019)	N/A	ntl	85.6	79.3 (↘)	80.3
Baseline	N/A	N/A	📷	80.7	79.8
Baseline++	euclid.	N/A	–	81.9	82.2
	cosine	N/A	📷	79.7	81.1

Table 1: Mean accuracy on the 12 ARSC binary classification test tasks. In column †, results are reproduced from the Induction Networks seminal paper (Geng et al., 2019) (where applies), a dash (–) means that results for that encoder/metric pair were not reported, and 📷 denotes models only tested on computer vision tasks (first time applied to text in our contribution). The *BERT* column is our implementation using the same 5 shots as the first column but using a BERT encoder for all methods. The last column is also using BERT, but results are averaged over five runs, sampling different shots for each run. In the Configuration column, N/A means that the configuration criteria does not apply to the model.

	Correct classification examples
S:	Do I have enough in my boa account for a new pair of skis ?
P:	`balance`
T:	`balance`
S:	What's 15% of 68 ?
P:	`calculator`
T:	`calculator`
S:	I need to know the nearest bank's location.
P:	`directions`
T:	`directions`
S:	What do I take home ?
P:	`income`
T:	`income`
	Incorrect classification examples
S:	On Tuesday you are supposed to have a meeting.
P:	`meeting_schedule`
T:	`calendar`
S:	What are my insurance rewards ?
P:	`insurance`
T:	`redeem_rewards`
S:	How much farther is Orlando from my location?
P:	`current_location`
T:	`distance`
S:	Stop talking please.
P:	`change_speed`
T:	`cancel`

Table 2: Examples of OOS query examples correctly and incorrectly predicted by the `Baseline` method using 5 shots. S (resp. P, T) is the sentence (resp. prediction and true label).

performances of all methods. When feeding the same transformer-based encoder to all few-shot methods, Prototypical Networks are now on top, whereas metric learning approaches (Induction & Relation Networks) tend to struggle, almost reaching the same performances as Matching Networks. Such metric learning approaches rely on various weight matrices and parameters, while more traditional approaches (`Matching`, `Proto`) do not use any parameter apart from the encoding step. This hints that the upstream transformer does most of the learning and is able to model the embedding space well enough such that no more additional metric learning is needed. The massive increase in embedding quality brought by the BERT encoder makes Prototypical Network approaches reclaim the state-of-the-art position.

4.4 The curious case of induction networks

When Geng et al. (2019) introduced Induction Networks, both the ARSC dataset and a private intent detection dataset were used for evaluation (publicly unavailable). Our experiments of this method on the ARSC dataset confirm those results in an acceptable range, even when trying to get more consistent results using multiple random seeds. Nonetheless, the performances of this method are underwhelming on all three intent detection datasets, even when matching the binary classification scenario using $C = 2$. Those poor performances were observed both on the test set

	Metric	Relation Module	Liu				OOS				TREC28			
			2	3	4	5	2	3	4	5	2	3	4	5
Matching	euclid.	-	96.6	93.7	91.1	89.1	99.2	98.7	98.1	97.7	89.4	81.6	76.6	69.6
	cosine	-	93.3	87.9	84.8	81.0	96.8	95.8	95.1	94.7	81.6	75.4	68.5	63.5
Proto	euclid.	-	97.4	95.3	93.4	91.8	**99.5**	99.0	98.7	98.4	**92.6**	**87.6**	82.0	**79.2**
	cosine	-	94.6	90.4	88.5	85.6	97.6	97.3	96.9	96.5	85.6	79.1	74.5	71.3
Proto++	euclid.	-	**97.7**	**95.7**	**93.7**	**92.2**	99.5	**99.1**	**98.8**	**98.5**	91.7	84.9	**82.0**	76.8
	cosine	-	94.0	90.9	87.9	85.4	97.5	97.3	97.0	96.5	83.8	78.1	71.0	65.9
Relation	-	base	88.2	76.5	71.8	65.1	91.1	86.0	79.9	77.9	80.8	66.3	61.7	51.8
	-	ntl	87.4	80.1	74.3	69.0	90.9	84.2	82.0	77.8	74.7	62.5	57.7	48.6
Induction	-	ntl	73.9	57.9	52.6	40.6	74.9	59.3	50.9	43.8	70.3	49.6	41.9	33.9
Baseline	-	-	94.3	89.0	84.1	79.8	99.1	98.5	97.7	97.2	90.5	83.6	79.3	75.7
Baseline++	euclid.	-	93.1	87.6	81.4	78.1	95.8	93.3	92.1	90.6	87.7	78.3	72.5	69.1
	cosine	-	93.1	86.8	81.0	75.1	98.9	97.9	96.8	96.1	86.7	78.2	72.1	70.0

Table 3: Mean accuracy of C-way 5-shot intent detection, with C ranging between 2 and 5. Each reported value is the average over five runs with different random seeds. For each column, the best method is highlighted in **bold**.

as well as the train set, discarding the over-fitting argument. Such a big performance gap between sentiment and intent classification tasks show that Induction Networks, while suited for the former, are not directly applicable to any type of task.

4.5 On metric choice

Prototypical Networks were originally designed to do better than Matching Networks. The two differences between them are the placement of the `class average` step, and the choice of the metric (cosine for Matching, euclidean for Prototypical). Our results show that metric choice yields a big gap in performances for both methods, this gap being larger than the gap caused by the model design. This hints that when using a pre-defined metric – excluding the case of metric learning –, choosing the right metric is of paramount importance. Moreover, while `Matching Networks` were designed to use the cosine distance, we found here that they perform significantly better when equipped with the Euclidean distance (on all datasets for all number of given test classes).

4.6 On architectural choices

Overall, Prototypical Networks come on top of every intent detection dataset. More importantly, their gap between other competing approaches is wider as the number of classes increases. This result is important, as in practice, the number of classes is likely to be higher than what is used in the literature. The extended variant, proto++, obtains mixed results. While this shows that using unlabeled data can have some benefits, we also observe that the proto++ way of integrating this external knowledge is perfectible. Ultimately, note that our results do not mirror Computer Vision results. Since few-shot learning methods are used on top of embeddings, we could emit the hypothesis that they can be applied to any embeddings, regardless of the field. However, while Relation Networks, for example, were performing well in Computer Vision classification tasks – the tasks which they were originally designed for – as well as text classification – back in the days when transformers did not exist –, this is not the case anymore. The drawback is that all methods are very sensitive to the feature extractor used in prior steps.

5 Conclusion

We provided a fair comparison for end-to-end neural few-shot text classification methods discovered over the last few years. When they are all equipped with a transformer-based text encoder, we show that Prototypical Networks become the state-of-the-art again. We also found that a traditional classifier trained on few shots yields very competitive results, especially when given shots are re-sampled at each iteration. Ultimately, we also demonstrated the significant impact of the vector metric, illustrated by Matching Networks strongly improving by only replacing the cosine by the euclidean distance. The complete source code with the re-implementation of all the tested methods and evaluation framework used in this study is publicly available[5] – we hope that it will help the community build upon consistent comparative experiments.

---

[5] https://github.com/tdopierre/FewShotText

References

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

Nontawat Charoenphakdee, Jongyeong Lee, Yiping Jin, Dittaya Wanvarie, and Masashi Sugiyama. 2019. Learning only from relevant keywords and unlabeled documents. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. *arXiv preprint arXiv:1902.10482*.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. *arXiv preprint arXiv:1903.05566*.

Rada Mihalcea. 2004. Co-training and self-training for word sense disambiguation. In *Proc. of 8th CoNLL-2004 (HLT-NAACL 2004)*.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. 2018. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601. Association for Computational Linguistics.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018a. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, New Orleans, Louisiana. Association for Computational Linguistics.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018b. Diverse few-shot text classification with multiple metrics. *arXiv preprint arXiv:1805.07513*.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: exploiting unlabeled data using three classifiers. *IEEE TKDE*, 17(11):1529–1541.

Multilingual Machine Translation: Closing the Gap between Shared and Language-specific Encoder-Decoders

Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa, Mikel Artetxe†
TALP Research Center, Universitat Politècnica de Catalunya, Barcelona
†IXA NLP Group, University of the Basque Country (UPV/EHU)
{carlos.escolano,marta.ruiz,jose.fonollosa}@upc.edu
mikel.artetxe@ehu.eus

Abstract

State-of-the-art multilingual machine translation relies on a universal encoder-decoder, which requires retraining the entire system to add new languages. In this paper, we propose an alternative approach that is based on language-specific encoder-decoders, and can thus be more easily extended to new languages by learning their corresponding modules. So as to encourage a common interlingua representation, we simultaneously train the N initial languages. Our experiments show that the proposed approach outperforms the universal encoder-decoder by 3.28 BLEU points on average, while allowing to add new languages without the need to retrain the rest of the modules. All in all, our work closes the gap between shared and language-specific encoder-decoders, advancing toward modular multilingual machine translation systems that can be flexibly extended in lifelong learning settings.

1 Introduction

Multilingual machine translation is the ability to generate translations automatically across a (large) number of languages. Research in this area has attracted a lot of attention in recent times both from the scientific and industrial community. Under the neural machine translation paradigm (Bahdanau et al., 2015), the opportunities for improving this area have dramatically expanded. Thanks to the encoder-decoder architecture, there are viable alternatives to expensive pairwise translation based on classic paradigms[1].

The main proposal in this direction is the universal encoder-decoder (joh, 2017) with massive multilingual enhancements (Arivazhagan et al., 2019). While this approach enables zero-shot translation and is beneficial for low-resource languages, it has multiple drawbacks: (i) the entire system has to be retrained when adding new languages or data; (ii) the quality of translation drops when adding too many languages or for those with the most resources (Arivazhagan et al., 2019); and (iii) the shared vocabulary grows dramatically when adding a large number of languages (especially when they do not share alphabets). Other limitations include the incompatibility of adding multiple modalities such as image or speech.

In this paper, we propose a new framework that can be incrementally extended to new languages without the aforementioned limitations (§3). Our proposal is based on language-specific encoders and decoders that rely on a common intermediate representation space. For that purpose, we simultaneously train the initial N languages in all translation directions. New languages are naturally added to the system by training a new module coupled with any of the existing ones, while new data can be easily added by retraining only the module for the corresponding language.

We evaluate our proposal on three experimental configurations: translation for the initial languages, translation when adding a new language, and zero-shot translation (§4). Our results show that the proposed method is better in the first configurations by improving the universal system: by 0.40 BLEU points on average in the initial training and by 3.28 BLEU points on average when adding new languages. However, our proposed system is still lagging behind universal encoder-decoder in zero-shot translation.

2 Related Work

Multilingual neural machine translation can refer to translating from one-to-many languages (Dong et al., 2015), from many-to-one (Zoph and Knight, 2016) and many-to-many (joh, 2017). Within the many-to-many paradigm, existing approaches can

---

[1] http://www.euromatrixplus.net

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 944–948
April 19 - 23, 2021.

be further divided into shared or language-specific encoder-decoders.

Shared Encoder-Decoder. joh (2017) feed a single encoder and decoder with multiple input and output languages. Given a set of languages, a shared architecture has a universal encoder and a universal decoder that are trained on all initial language pairs at once. The model shares parameters, vocabulary and tokenization among languages to ensure that no additional ambiguity is introduced in the representation. This architecture provides a simple framework to develop multilingual systems because it does not require modifications of a standard neural machine translation model, and information is easily shared among the different languages through common parameters. Despite the model's advantages in transfer learning, the use of a shared vocabulary and embedding representation forces the model to employ a vocabulary that includes tokens from all the alphabets used. Additionally, recent work (Arivazhagan et al., 2019), that imposes representational invariance across language, shows increasing the number of languages varies the quality of the languages already in the system (generally enhancing low-resource pairs but being detrimental for high-resource pairs).

Language-specific Encoder-Decoders which may or may not share parameters at some point.

Sharing parameters. Firat et al. (2016b) proposed extending the bilingual recurrent neural machine translation architecture (Bahdanau et al., 2015) to the multilingual case (Vázquez et al., 2019; Lu et al., 2018) by designing a shared attention-based mechanism between the language-specific encoders and decoders to create a language independent representation. As the language specific components rely on the shared modules, modifying those components to add a new language or add further data to the system would require retraining the whole system (similarly to the previous shared approach). (Lakew et al., 2018) proposes a model based on the addition of new languages to an already trained system by vocabulary adaptation and transfer learning. While limited, it requires some retraining to adapt the model to the new task.

No sharing. The system proposed by Escolano et al. (2019) is trained on language-specific encoders and decoders based on joint training without parameter or vocabulary-sharing and on enforcing a compatible representation between the jointly trained languages. The advantage of the approach is that it does not require retraining to add new languages and increasing the number of languages does not vary the quality of the languages already in the system. However, the system has to be trained on a multi-parallel corpus and it does not scale well when there is a large number of languages in the initial system, since all encoders and decoders have to be trained simultaneously.

3 Proposed Method

Our proposed approach trains a separate encoder and decoder for each of the N languages available without requiring multi-parallel corpus. We do not share any parameter across these modules, which allows to add new languages incrementally without retraining the entire system.

3.1 Definitions

We next define the notation that we will be using when describing our approach. We denote the encoder and the decoder for the ith language in the system as e_i and d_i, respectively. For language-specific scenarios, both the encoder and decoder are considered independent modules that can be freely interchanged to work in all translation directions.

3.2 Language-Specific Proposed Procedure

In what follows, we describe the proposed training procedure in two steps: joint training and adding new languages.

Joint Training The straightforward approach is to train independent encoders and decoders for each language. The main difference from standard pairwise training is that, in this case, there is only one encoder and one decoder for each language, which will be used for all translation directions involving that language. The training algorithm for this language-specific procedure is described in Algorithm 1.

For each translation direction $s_{i,j}$ in the training schedule S with language i as source and language j as target, the system is trained using the language-specific encoder e_i and decoder d_j.

Adding New Languages Since parameters are not shared between the independent encoders and decoders, the joint training enables the addition of

Algorithm 1 Multilingual training step

```
procedure MULTILINGUALTRAININGSTEP
    $N \leftarrow$ Number of languages in the system
    $S = \{s_{0,0}, ..., s_{N,N}\} \leftarrow \{(e_i, d_j)\}$
    $E = \{e_0, ..., e_N\} \leftarrow$ Language-specific encs.
    $D = \{d_0, ..., d_N\} \leftarrow$ Language-specific decs.
    for $i \leftarrow 0$ to $N$ do
        for $j \leftarrow 0$ to $N$ do
            if $s_{i,j} \in S$ then
                $l_i, l_j = get\_parallel\_batch(i, j)$
                $train(s_{i,j}(e_i, d_j), l_i, l_j)$
```

new languages without the need to retrain the existing modules. Let us say we want to add language $N + 1$. To do so, we must have parallel data between $N + 1$ and any language in the system. For illustration, let us assume that we have $L_{N+1} - L_i$ parallel data. Then, we can set up a new bilingual system with language L_{N+1} as source and language L_i as target. To ensure that the representation produced by this new pair is compatible with the previously jointly trained system, we use the previous L_i decoder (d_{li}) as the decoder of the new L_{N+1}-L_i system and we freeze it. During training, we optimize the cross-entropy between the generated tokens and L_i reference data but update only the parameters of to the L_{N+1} encoder ($e_{l_{N+1}}$). By doing so, we train $e_{l_{N+1}}$ not only to produce good quality translations but also to produce similar representations to the already trained languages. Following the same principles, the L_{N+1} decoder can also be trained as a bilingual system by freezing the L_i encoder and training the decoder of the $L_i - L_{N+1}$ system by optimizing the cross-entropy with the L_{N+1} reference data.

4 Experiments in Multilingual Machine Translation

In this section we report machine translation experiments in different settings. Since the main difference between the shared and the language-specific encoders-decoders lies in whether they retrain the entire system when adding new languages, we accordingly design our experiments to compare the systems under this condition.

4.1 Data and Implementation

We used 2 million sentences from the *EuroParl* corpus (Koehn, 2005) in German, French, Spanish and English as training data, with parallel sentences among all combinations of these four languages (without being multi-parallel). For Russian-English, we used 1 million training sentences from the *Yandex* corpus[2]. As validation and test set, we used *newstest2012* and *newstest2013* from WMT[3], which is multi-parallel across all the above languages. All data were preprocessed using standard Moses scripts (Koehn et al., 2007)

We evaluate our approach in 3 different settings: (i) the *initial* training, covering all combinations of German, French, Spanish and English; (ii) *adding* new languages, tested with Russian-English in both directions; and (iii) *zero-shot* translation, covering all combinations between Russian and the rest of the languages. Additionally we compare two configurations which consists in using non-tied or tied embeddings. In the language-specific approach tied embeddings consists in using language-wise word embeddings: for one language, we use the same word embeddings. Whereas, in the case of non-tied, the encoder and the decoder of each language have different word embeddings. Tied embeddings in the shared system means that both encoder and decoder share the same word embeddings.

All experiments were done using the Transformer provided by Fairseq(Ott et al., 2019) [4]. We used 6 layers, each with 8 attention heads, an embedding size of 512 dimensions, and a vocabulary size of 32k subword tokens with Byte Pair Encoding (Sennrich et al., 2016) (in total for the shared encoders/decoders and per pair for language-specific encoder-decoders). Dropout was 0.1 for the shared approach and 0.3 for language-specific encoders/decoders. Both approaches were trained with an effective batch size of 32k tokens for approximately 200k updates, using the validation loss for early stopping. We used Adam (Kingma and Ba, 2015) as the optimizer, with learning rate of 0.001 and 4000 warmup steps.

4.2 Shared vs Language-specific

Table 1 and 2 show comparisons between the shared and language-specific encoders-decoders.

In contrast with our proposed approach, the shared system requires retraining from scratch to

---

[2] `https://translate.yandex.ru/corpus?lang=en`

[3] `http://www.statmt.org`

[4] Release v0.6.0 available at `https://github.com/pytorch/fairseq`

	Shared		LangSpec	
	¬Tied	Tied	¬Tied	Tied
de-en	24,40	**25,04**	22,04	24,54
de-es	24,04	25,01	22,38	**25,02**
de-fr	24,78	25,14	22,57	**25,49**
en-de	21,39	21,51	19,44	**22,01**
en-es	28,08	28,19	26,79	**29,53**
en-fr	28,43	28,67	26,94	**29,74**
es-de	19,51	20,21	17,7	**20,31**
es-en	26,66	26,93	24,9	**27,75**
es-fr	29,47	29,59	27,31	**30,08**
fr-de	19,22	19,81	16,88	**19,97**
fr-en	25,78	26,29	23,5	**26,55**
fr-es	28,15	29,03	26,78	**29,07**

Table 1: Initial training. In bold, best global results.

	Shared^{RU}		LangSpec	
	¬Tied	Tied	¬Tied	Tied
ru-en	24,71	24,62	25,52	**27,54**
en-ru	19,91	20,03	21,44	**23,94**
ru-de	15,36	**16,52**	12,73	13,77
ru-es	21,38	**23,12**	18,71	21,08
ru-fr	21,38	**22,04**	18,05	19,85
de-ru	16,23	**17,27**	14,39	16,99
es-ru	16,98	**18,78**	15,93	18,46
fr-ru	16,79	**17,83**	15,16	17.47

Table 2: Adding a new language translation and Zero-shot.

add a new language. For that reason, we experiment with two variants of this system: one trained without Russian-English (*Shared*) and another one including this pair (*Shared*^{RU}). Note that, to make experiments comparable, we use the *Shared* version when comparing to our *initial* system in Table 1, and the *Shared*^{RU} version when *adding* new languages and performing *zero-shot* translation.

Initial Training Table 1 shows that the language-specific encoder-decoders outperforms the shared approach in all cases. On average, our proposed approach is better than the shared approach with a difference of 0.40 BLEU points.

Adding New Languages Table 2 shows that, when adding a new language into the system, the language-specific encoder-decoders outperform the shared architecture by 2.92 BLEU points for Russian-to-English and by 3.64 BLEU in the opposite direction. It is also worth mentioning that the Russian data is from a different domain than the frozen English modules used for training (*Yandex* corpus and *EuroParl*, respectively). As such, the language specific encoder-decoders are able to outperform the shared architecture when adding a new language and a new domain by learning from the previous information in the frozen modules. Note that additionally, retraining the shared encoder-decoder to add a new language took an entire week, whereas the incremental training with the language-specific encoder-decoders was performed in only one day.

Zero-shot The shared encoder-decoder clearly outperforms the language-specific encoder-decoders by 1.39 BLEU points on average. This difference in performance suggests that, while limiting the amount of shared information during training can improve our model performance, it may also harm zero-shot translation.

Note that employing tied embeddings has a larger impact in the language-specific architecture than in the shared one. In fact, it has been key for closing the performance gap between language-specific and shared architectures.

Previous works, e.g. (Firat et al., 2016a; Lu et al., 2018), suffered from attention mismatch which was solved by sharing intermediate layers. In our case, we believe that we do not suffer from this problem because, within an initial set of N languages, we train $N*N-1$ systems using pair-wise corpus (without requiring multi-parallel corpus as previous works (Escolano et al., 2019)). Due to our proposed joint training, once we have trained our initial system, we end up with only N encoders and N decoders ($2*N$).

5 Conclusions

In this paper, we present a novel method to train language-specific encoders-decoders without sharing any parameters at all. More relevantly, our system allows to incrementally add new languages into the system without having to retrain it and without varying the translation quality of initial languages in the system. When adding a new language, the language-specific encoder-decoders outperform the shared ones by 3.28 BLEU on average and, most importantly, the training of this new language was done in only one day, as opposed to the week taken by the shared system.

Acknowledgments

This work is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 947657).

References

2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the ACL-IJCNLP*, pages 1723–1732, Beijing.

Carlos Escolano, Marta R. Costa-jussà, and José A. R. Fonollosa. 2019. From bilingual to multilingual neural machine translation by incremental training. In *Proceedings of the ACL: Student Research Workshop*, Florence, Italy.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the NAACL-HLT, month = jun*, pages 866–875, San Diego, California.

Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86. Citeseer.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL: Demo Papers*, pages 177–180.

Surafel Melaku Lakew, Aliia Erofeeva, Matteo Negri, Marcello Federico, and Marco Turchi. 2018. Transfer learning in multilingual neural machine translation with dynamic vocabulary. *CoRR*, abs/1811.01137.

Yichao Lu, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. 2018. A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 84–92, Belgium, Brussels.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany.

Raúl Vázquez, Alessandro Raganato, Jörg Tiedemann, and Mathias Creutz. 2019. Multilingual NMT with a language-independent attention bridge. In *Proceedings of the RepL4NLP Workshop)*, pages 33–39, Florence, Italy.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *Proceedings of the 2016 Conference of the NAACL-HLT*, pages 30–34, San Diego.

Clustering Word Embeddings with Self-Organizing Maps. Application on LaRoSeDa – A Large Romanian Sentiment Data Set

Anca Maria Tache[1,*], **Mihaela Găman**[1,‡], **Radu Tudor Ionescu**[1,2,◇]
[1]Department of Computer Science, [2]Romanian Young Academy
University of Bucharest
14 Academiei, Bucharest, Romania
*`anca.taches@gmail.com` ‡`mp.gaman@gmail.com`
◇`raducu.ionescu@gmail.com`

Abstract

Romanian is one of the understudied languages in computational linguistics, with few resources available for the development of natural language processing tools. In this paper, we introduce LaRoSeDa, a **La**rge **Ro**manian **Se**ntiment **Da**ta Set[1], which is composed of 15,000 positive and negative reviews collected from one of the largest Romanian e-commerce platforms. We employ two sentiment classification methods as baselines for our new data set, one based on low-level features (character n-grams) and one based on high-level features (bag-of-word-embeddings generated by clustering word embeddings with k-means). As an additional contribution, we replace the k-means clustering algorithm with self-organizing maps (SOMs), obtaining better results because the generated clusters of word embeddings are closer to the Zipf's law distribution, which is known to govern natural language. We also demonstrate the generalization capacity of using SOMs for the clustering of word embeddings on another recently-introduced Romanian data set, for text categorization by topic.

1 Introduction

Perhaps one of the most studied tasks in computational linguistics is sentiment classification, a.k.a. opinion mining or polarity classification. The task has been studied across several languages, the most popular being English (Blitzer et al., 2007; Dos Santos and Gatti, 2014; Fu et al., 2018; Giménez-Pérez et al., 2017; Huang et al., 2017; Ionescu and Butnaru, 2019; Kim, 2014; Maas et al., 2011; Pang and Lee, 2005; Shen et al., 2018; Socher et al., 2013), Chinese (Peng et al., 2017; Wan, 2008; Zagibalov and Carroll, 2008; Zhai et al., 2011; Zhang and He, 2013; Zhang et al., 2008), Arabic (Al-Ayyoub et al., 2018; Elsahar and El-Beltagy, 2015; Dahou et al., 2016; Nabil et al., 2013, 2015) or Spanish (Brooke et al., 2009; Molina-González et al., 2015; Navas-Loro and Rodríguez-Doncel, 2019; Vilares et al., 2015; Zafra et al., 2017). However, studying this task for under-studied languages, e.g. Romanian, is difficult without access to large data sets. We hereby introduce LaRoSeDa, a **La**rge **Ro**manian **Se**ntiment **Da**ta Set, which is freely available for download at https://github.com/ancatache/LaRoSeDa for non-commercial use. With a total of 15,000 positive and negative reviews collected from one of the most popular Romanian e-commerce websites, to our knowledge, LaRoSeDa is the largest data set for Romanian polarity classification.

We experiment with two baseline methods on our novel data set. The first baseline employs string kernels, an approach based on low-level features (character n-grams), that was found to work well for sentiment analysis across multiple languages, e.g. English (Giménez-Pérez et al., 2017; Ionescu and Butnaru, 2018), Chinese (Zhang et al., 2008) and Arabic (Popescu et al., 2017), requiring no linguistic resources besides a labeled training set of samples. The second baseline employs bag-of-word-embeddings (Ionescu and Butnaru, 2019; Fu et al., 2018), an approach based on high-level features (clusters of word embeddings generated by k-means), that attains good results in various text classification tasks (Butnaru and Ionescu, 2017; Cozma et al., 2018; Fu et al., 2018; Ionescu and Butnaru, 2019), including sentiment analysis. As an additional contribution, we replace the k-means clustering algorithm in the second baseline method with self-organizing maps (SOMs) (Kohonen, 2001), obtaining better results because the generated clusters of word embeddings are closer to the Zipf's law distribution (see Figure 2), which is known to govern natural language (Powers, 1998). To our knowl-

---

[1] https://github.com/ancatache/LaRoSeDa

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 949–956
April 19 - 23, 2021.

edge, we are the first to apply SOMs to cluster word embeddings, showing performance gains for both word2vec (Mikolov et al., 2013) and Romanian BERT (Dumitrescu et al., 2020) embeddings. We also demonstrate the generalization capacity of using SOMs in the bag-of-word-embeddings on another recently-introduced Romanian data set (Butnaru and Ionescu, 2019), for the task of text categorization by topic.

In summary, our contribution is twofold:

- We introduce LaRoSeDa, one of the largest corpora for Romanian sentiment analysis, along with a set of strong baselines to be used as reference in future research.
- To our knowledge, we are the first to employ SOMs as a technique to cluster word embeddings. We provide empirical evidence showing that SOMs produce better results than the popular k-means.

2 Related Work

To date, a small number of works targeting sentiment classification in the Romanian language have been published. Preceding the sentiment analysis efforts on Romanian texts, there are a few studies on subjectivity, that have introduced two corpora built through cross-lingual projections from English to Romanian (Mihalcea et al., 2007) or through machine translation (Banea et al., 2008). An extensive study conducted by Banea et al. (2011) looks at sentiment and subjectivity from a computational linguistics perspective, in a multilingual setup in which Romanian is also included. However, in these initial works, Romanian is studied only from a subjectivity perspective, which does not go down to the level of polarity.

On our topic (i.e. sentiment analysis in Romanian), the first study that we have found describes two word sets tagged with polarity for Romanian and Russian (Sokolova and Bobicev, 2009). Gînscă et al. (2011) introduced a sentiment analysis service intended for multiple languages, that also supports Romanian. They perform sentiment identification using a list of manually-built triggers which, to our knowledge, is not publicly available. Another effort (Colhon et al., 2016) in creating an opinion lexicon with polarity annotations introduced a collection of 2,521 Romanian tourist reviews and an extensive linguistic analysis of the corpus. The data set is not released for public use. Similarly, we did not find any public link to RoEmoLex, a lexicon with approximately 11,000 Romanian words tagged for emotion and sentiment (Briciu and Lupea, 2017; Lupea and Briciu, 2017). The only Romanian data set annotated for sentiment that we have found freely available is rather small, with 1,000 movie reviews manually extracted from several blogs and sites (Russu et al., 2014). With 15,000 reviews, our corpus is much larger.

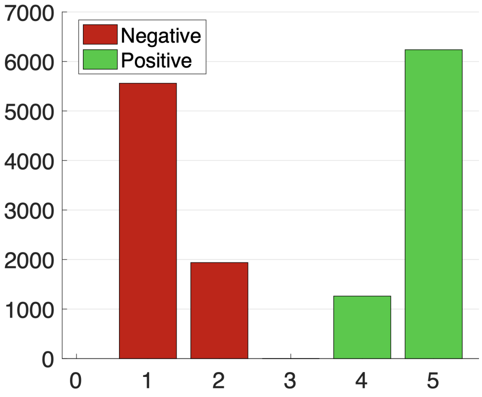


Figure 1: The rating distribution of Romanian product reviews. Negative reviews are those rated with one or two stars, while positive reviews are those rated with four or five stars. Neutral reviews are not included in our data set. Best viewed in color.

3 Data Set

In order to build LaRoSeDa, we collected product reviews from one of the largest e-commerce websites in Romania. Along with the textual content of each review, we collected the associated star ratings in order to automatically assign labels to the collected text samples. Following the same approach used for data sets containing English reviews (Blitzer et al., 2007; Maas et al., 2011; Pang and Lee, 2005), we assigned positive labels to the reviews rated with four or five stars and negative labels to the reviews rated with one or two stars. However, the star rating might not always reflect the polarity of the text. We thus acknowledge that the automatic labeling process is not optimal, i.e. some labels might be noisy. Since automatic labeling based on star ratings is a commonly accepted practice for opinion mining data sets of product reviews, we leave the analysis of noisy labels and manual labeling for future work.

We also imitate the data collection approach used for English review data sets (Blitzer et al., 2007; Maas et al., 2011; Pang and Lee, 2005), selecting a balanced set of Romanian reviews. More precisely, LaRoSeDa is formed of a total of 15,000 reviews that are perfectly balanced, i.e. half of them (7,500)

Set	Positive		Negative	
	#samples	#words	#samples	#words
Training	6,000	187,813	6,000	244,499
Test	1,500	47,661	1,500	60,314
Total	7,500	235,474	7,500	304,813

Table 1: The number of positive and negative samples (#samples) and the corresponding number of words (#words) contained in the training set and the test set of LaRoSeDa.

are positive reviews and the other half (7,500) are negative reviews. In Figure 1, we show the distribution of reviews with respect to the star ratings. We note that most of the negative reviews (5,561) are rated with one star. Similarly, most of the positive reviews (6,238) are rated with five stars. Hence, the corpus is highly polarized. We divide LaRoSeDa into a training set containing 80% of the data samples and a test set containing the remaining 20%. In Table 1, we present the number of positive and negative reviews inside each subset, along with the number of words. Our data set contains a total of 540,287 words, with an average of 36 words per review. We observe that positive reviews contain 235,474 words (44.6%) and negative reviews contain 304,813 words (56.4%). We note that, in negative reviews, people are likely to complain about several points or to explain what is wrong with the reviewed products. This could provide a natural explanation for the fact that the negative reviews contain more words than the positive reviews.

4 Methods

String kernels. A simple language-independent and linguistic-theory-neutral approach is to interpret text samples as sequences of characters (strings) and to use character n-grams as features. The number of character n-grams is usually much higher than the number of samples, so representing the text samples as feature vectors may require a lot of space. String kernels provide an efficient way to avoid storing and using the feature vectors (primal form), by representing the data though a kernel matrix (dual form). Each component K_{ij} in a kernel matrix represents the similarity between data samples x_i and x_j. In our experiments, we use the histogram intersection string kernel (HISK) (Ionescu et al., 2014, 2016) as the similarity function. For two strings x_i and x_j over a set of characters S, HISK is defined as follows:

$$k^{\cap}(x_i, x_j) = \sum_{g \in S^n} \min\{\#(x_i, g), \#(x_j, g)\}, \quad (1)$$

where $\#(x, g)$ is a function that returns the number of occurrences of n-gram g in x, and n is the length of n-grams. While being a rather shallow approach, string kernels attained strong results in some specific tasks. For instance, string kernels ranked first in the Arabic Dialect Identification tasks of VarDial 2017 (Ionescu and Butnaru, 2017) and VarDial 2018 (Butnaru and Ionescu, 2018).

Bag-of-word-embeddings. Following the seminal paper of Mikolov et al. (2013) introducing *word2vec*, word embeddings became one of the mainstream approaches in various computational linguistics tasks (Cheng et al., 2018; Conneau et al., 2017; Cozma et al., 2018; Fu et al., 2018; Kim, 2014; Kiros et al., 2015; Shen et al., 2018; Torki, 2018; Zhou et al., 2018). In order to build the bag-of-word-embeddings (BOWE), we first trained *word2vec* on the collected Romanian reviews using the continuous bag-of-words (CBOW) model. Before training, we transformed all letters to lowercase and removed punctuation. In addition to word2vec, we consider a recently introduced Romanian BERT model (Dumitrescu et al., 2020) as an alternative way to produce word embeddings, which is likely to produce much better results, considering the success of BERT (Devlin et al., 2019) in English NLP tasks. Instead of averaging the word embeddings to obtain document-level representations (Shen et al., 2018), we follow a different and more effective path suggested by some recent works (Butnaru and Ionescu, 2017; Cozma et al., 2018; Fu et al., 2018; Ionescu and Butnaru, 2019). More specifically, we cluster the word embeddings collected from the entire training set using k-means. For a document D of n words, $D = (w_1, w_2, ..., w_n)$, a word embedding model, be it word2vec or BERT, outputs a matrix of $n \times m$ components (or a set of n m-dimensional vectors), the m-dimensional vector at index i corresponding to word w_i. We apply clustering on the word vectors extracted from all training documents, thus obtaining a set of k clusters. A document D is then represented as a bag-of-word-embeddings (histogram) $H = (h_1, h_2, ..., h_k)$ in which each component h_i retains the number of word embeddings from the document D that fall in cluster i, where $i \in \{1, 2, ..., k\}$. We note that the size of the bag-of-word-embeddings is equal to the number of clusters k. In the case of BERT, we emphasize that, although the embedding vector of a word depends on the context, it is likely that the embedding vectors corresponding to a specific word will fall in the same cluster. Hence, BOWE is able to cope well with this situation.

Replacing k-means with SOMs. Quantitative linguistics studies (Powers, 1998) have pointed out that, given a corpus of text documents, the frequency of any word is inversely proportional to its rank in the frequency table, giving rise to a Zipf's law distribution of words in natural language. However, the k-means algorithm tends to ignore the data density, producing equally-sized clusters. We therefore propose to replace the k-means algorithm with an approach that takes into account the density in the word embedding space, producing a set of clusters that follow the Zipf's law. We propose to perform clustering using self-organizing maps (SOMs) (Kohonen, 2001), since these models are known to preserve the topological properties of the input space. Indeed, Figure 2 shows that SOMs produce clusters of Romanian word embeddings closer to the Zipf's law distribution than k-means.

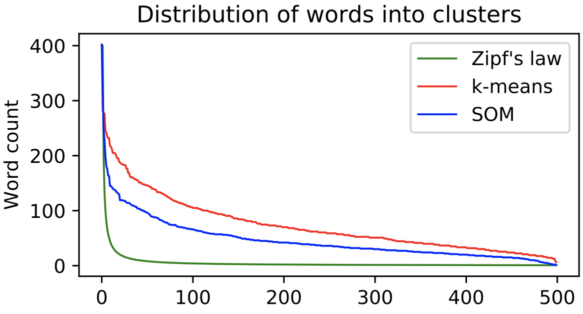


Figure 2: The distribution of words into clusters generated by k-means (in red) or by SOMs (in blue), for LaRoSeDa. The Zipf's law distribution (in green) is included for reference. Best viewed in color.

It is important to emphasize that k-means can produce clusters of different size, as shown in Figure 2. Our observation refers only to the fact that the data density is not particularly modeled by the k-means optimization process, while SOMs are optimized by shifting the neural units following the density of the data (units tend to migrate where the space is more dense). Our observation with respect to k-means is also confirmed by other studies. For example, Raykov et al. (2016) note that: *"even when all other implicit geometric assumptions of k-means are satisfied, it will fail to learn a correct, or even meaningful, clustering when there are significant differences in cluster density"*. Since natural language involves such significant differences (due to the presence of the Zipf's law), we believe that k-means is a sub-optimal choice.

5 Experiments

Corpora. First and foremost, we perform experiments on LaRoSeDa with the goal of introducing some benchmark results on our new data set. We also perform experiments on MOROCO (Butnaru and Ionescu, 2019), a data set with Moldavian and Romanian news articles, with the goal of showing the generalization capacity of using SOMs instead ok k-means.

Experimental setup. On LaRoSeDa, we present two sets of results, one based on the established train-test split and one based on 10-fold cross-validation. On MOROCO, we choose to present 10-fold cross-validation results for the intra-dialect multi-class categorization by topic task, on the 18,161 samples written in the Romanian dialect.

Parameter and model choices. For HISK, we combined character 3-grams, 4-grams and 5-grams. For BOWE-word2vec and BOWE-BERT, we set the number of clusters to $k = 500$, just as Cozma et al. (2018). We trained *word2vec* to produce 300-dimensional Romanian word embeddings, while the Romanian BERT outputs 768-dimensional embeddings. In the learning stage, we employed the linear Support Vector Machines (SVM) implementation from Scikit-learn (Pedregosa et al., 2011), providing as input pre-computed kernels. For BOWE-word2vec and BOWE-BERT, we opt for the PQ kernel, based on the findings of Butnaru and Ionescu (2017). We set the regularization parameter of SVM to $C = 10^3$ in all the experiments. We also fuse HISK with BOWE-word2vec or BOWE-BERT in the dual form by summing up the corresponding kernel matrices. We employed an open source implementation of SOMs[2]. We used the default choices for most hyperparameters, the modifications being detailed next. We set the learning rate to 0.25 and the number of epochs to 200. Before starting the training, the SOM is configured to randomly choose a number of training samples equal to number of expected outputs. We

---

[2] http://neupy.com/pages/home.html

Method	10-fold CV	Test
HISK	84.38%	84.73%
BOWE-word2vec with k-means	72.24%	70.73%
BOWE-word2vec with SOMs	75.99%	75.57%
BOWE-BERT with k-means	78.63%	77.36%
BOWE-BERT with SOMs	79.75%	80.73%
HISK+BOWE-word2vec with k-means	85.08%	83.57%
HISK+BOWE-word2vec with SOMs	87.17%	88.93%
HISK+BOWE-BERT with k-means	88.81%	89.42%
HISK+BOWE-BERT with SOMs	89.54%	90.90%

Table 2: Accuracy rates of HISK, BOWE-word2vec and BOWE-BERT with clustering based on k-means or SOMs, as well as ensemble models on LaRoSeDa. Results are reported in two cases: using a 10-fold cross-validation procedure and using the train-test split.

Method	10-fold CV
HISK (Butnaru and Ionescu, 2019)	71.27%
BOWE-BERT with k-means	63.42%
BOWE-BERT with SOMs	68.50%
HISK+BOWE-BERT with k-means	72.21%
HISK+BOWE-BERT with SOMs	73.35%

Table 3: Accuracy rates of HISK, BOWE-BERT with k-means, BOWE-BERT with SOMs and their combinations on MOROCO, for the Romanian intra-dialect multi-class categorization by topic task. Results are reported using a 10-fold cross-validation procedure.

opted for the cosine distance between data samples and SOM's weights.

Results on LaRoSeDa. In Table 2, we present the results on LaRoSeDa. Among the individual baselines, we observe that HISK attains the best accuracy rates, surpassing all BOWE configurations. We also note that by replacing k-means with SOMs, the accuracy rate of BOWE-word2vec grows by 4 or 5%. The improvements brought by SOMs can be explained by the fact that, unlike k-means, SOMs produce clusters that are closer to the Zipf's law distribution. This is proven by the word embedding counts per cluster illustrated in Figure 2. When we combine HISK with BOWE-BERT, we notice significant performance gains.

Results on MOROCO. In Table 3, we present the results on MOROCO, for the Romanian intra-dialect multi-class categorization by topic task. We notice that HISK attains better results than BOWE-BERT with k-means and BOWE-BERT with SOMs, although the differences in terms of accuracy seem to be smaller. As for LaRoSeDa, we observe a significant improvement (higher than 5%) when k-means is replaced by SOMs. There is an observable improvement over the plain HISK, when HISK is combined with BOWE-BERT based on k-means. Nonetheless, we notice a larger improvement when we combine HISK and BOWE-BERT based on SOMs.

6 Conclusion

In this paper, (i) we introduced LaRoSeDa, a large data set for polarity classification of Romanian reviews, and (ii) we employed self-organizing maps, a clustering approach that preserves the density of words in the embedding space, resulting in a more effective bag-of-word-embeddings representation. Our top accuracy rates on LaRoSeDa are 89.54% for the cross-validation procedure and 90.90% on the test set. We note that SOMs had a significant contribution in attaining these high accuracy rates. We conclude that the combination of HISK and BOWE-BERT with SOMs is a strong baseline which should encourage future research in proposing non-trivial models for Romanian polarity classification. Furthermore, the results obtained on MOROCO confirm that SOMs provide better accuracy rates than k-means when it comes to building document-level representations based on clustering word embeddings.

Acknowledgments

The authors thank reviewers for their useful remarks. This work was supported by a grant of the Romanian Ministry of Education and Research, CNCS - UEFISCDI, project number PN-III-P1-1.1-TE-2019-0235, within PNCDI III. This article has also benefited from the support of the Romanian Young Academy, which is funded by Stiftung Mercator and the Alexander von Humboldt Foundation for the period 2020-2022.

References

Mahmoud Al-Ayyoub, Aya Nuseir, Kholoud Alsmearat, Yaser Jararweh, and Brij Gupta. 2018. Deep learning for Arabic NLP: A survey. *Journal of Computational Science*, 26:522–531.

Carmen Banea, Rada Mihalcea, and Janyce Wiebe. 2011. Multilingual Sentiment and Subjectivity Analysis. In *Multilingual Natural Language Processing Applications: From Theory to Practice*, chapter 7. IBM Press.

Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008. Multilingual Subjectivity Analysis Using Machine Translation. In *Proceedings of EMNLP*, page 127–135.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of ACL*, pages 187–205.

Anamaria Briciu and Mihaiela Lupea. 2017. RoEmoLex – A Romanian Emotion Lexicon. *Studia Universitatis Babeș-Bolyai Informatica*, 62(2):45–56.

Julian Brooke, Milan Tofiloski, and Maite Taboada. 2009. Cross-linguistic sentiment analysis: From English to Spanish. In *Proceedings of RANLP*, pages 50–54.

Andrei Butnaru and Radu Tudor Ionescu. 2017. From Image to Text Classification: A Novel Approach based on Clustering Word Embeddings. In *Proceedings of KES*, pages 1784–1793.

Andrei Butnaru and Radu Tudor Ionescu. 2019. MOROCO: The Moldavian and Romanian Dialectal Corpus. In *Proceedings of ACL*, pages 688–698.

Andrei M. Butnaru and Radu Tudor Ionescu. 2018. UnibucKernel Reloaded: First Place in Arabic Dialect Identification for the Second Year in a Row. In *Proceedings of VarDial*, pages 77–87.

Zhou Cheng, Chun Yuan, Jiancheng Li, and Haiqin Yang. 2018. TreeNet: Learning Sentence Representations with Unconstrained Tree Structure. In *Proceedings of IJCAI*, pages 4005–4011.

Mihaela Colhon, Mădălina Cerban, Alex Becheru, and Mirela Teodorescu. 2016. Polarity shifting for Romanian sentiment classification. In *Proceedings of INISTA*, pages 1–6.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of EMNLP*, pages 670–680.

Mădălina Cozma, Andrei Butnaru, and Radu Tudor Ionescu. 2018. Automated essay scoring with string kernels and word embeddings. In *Proceedings of ACL*, pages 503–509.

Abdelghani Dahou, Shengwu Xiong, Junwei Zhou, Mohamed Houcine Haddoud, and Pengfei Duan. 2016. Word embeddings and convolutional neural network for Arabic sentiment classification. In *Proceedings of COLING*, pages 2418–2427.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*, pages 4171–4186.

Cícero Nogueira Dos Santos and Maira Gatti. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *Proceedings of COLING*, pages 69–78.

Ştefan Daniel Dumitrescu, Andrei-Marius Avram, and Sampo Pyysalo. 2020. The birth of Romanian BERT. In *Findings of EMNLP*, pages 4324–4328.

Hady Elsahar and Samhaa R. El-Beltagy. 2015. Building large Arabic multi-domain resources for sentiment analysis. In *Proceedings of CICLing*, pages 23–34.

Mingsheng Fu, Hong Qu, Li Huang, and Li Lu. 2018. Bag of meta-words: A novel method to represent document for the sentiment classification. *Expert Systems with Applications*, 113:33–43.

Rosa M. Giménez-Pérez, Marc Franco-Salvador, and Paolo Rosso. 2017. Single and Cross-domain Polarity Classification using String Kernels. In *Proceedings of EACL*, pages 558–563.

Alexandru-Lucian Gînscă, Emanuela Boroș, Adrian Iftene, Diana Trandabăț, Mihai Toader, Marius Corîci, Cenel-Augusto Perez, and Dan Cristea. 2011. Sentimatrix – Multilingual Sentiment Analysis Service. In *Proceedings of WASSA*, pages 189–195.

Xingchang Huang, Yanghui Rao, Haoran Xie, Tak-Lam Wong, and Fu Lee Wang. 2017. Cross-Domain Sentiment Classification via Topic-Related TrAdaBoost. In *Proceedings of AAAI*, pages 4939–4940.

Radu Tudor Ionescu and Andrei Butnaru. 2019. Vector of Locally-Aggregated Word Embeddings (VLAWE): A Novel Document-level Representation. In *Proceedings of NAACL*, pages 363–369.

Radu Tudor Ionescu and Andrei M. Butnaru. 2017. Learning to Identify Arabic and German Dialects using Multiple Kernels. In *Proceedings of VarDial*, pages 200–209.

Radu Tudor Ionescu and Andrei M. Butnaru. 2018. Improving the results of string kernels in sentiment analysis and Arabic dialect identification by adapting them to your test set. In *Proceedings of EMNLP*, pages 1084–1090.

Radu Tudor Ionescu, Marius Popescu, and Aoife Cahill. 2014. Can characters reveal your native language?

A language-independent approach to native language identification. In *Proceedings of EMNLP*, pages 1363–1373.

Radu Tudor Ionescu, Marius Popescu, and Aoife Cahill. 2016. String Kernels for Native Language Identification: Insights from Behind the Curtains. *Computational Linguistics*, 42(3):491–525.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of EMNLP*, pages 1746–1751.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-Thought Vectors. In *Proceedings of NIPS*, pages 3294–3302.

Teuvo Kohonen. 2001. *Self-Organizing Maps*, 3rd edition. Springer-Verlag.

Mihaiela Lupea and Anamaria Briciu. 2017. Formal concept analysis of a Romanian emotion lexicon. In *Proceedings of ICCP*, pages 111–118.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of ACL*, pages 142–150.

Rada Mihalcea, Carmen Banea, and Janyce Wiebe. 2007. Learning Multilingual Subjective Language via Cross-Lingual Projections. In *Proceedings of ACL*, pages 976–983.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of NIPS*, pages 3111–3119.

Dolores M. Molina-González, Eugenio Martínez-Cámara, Teresa M. Martín-Valdivia, and Alfonso L. Ureña-López. 2015. A Spanish semantic orientation approach to domain adaptation for polarity classification. *Information Processing & Management*, 51(4):520–531.

Mahmoud Nabil, Mohamed Aly, and Amir Atiya. 2015. ASTD: Arabic Sentiment Tweets Dataset. In *Proceedings of EMNLP*, pages 2515–2519.

Mahmoud Nabil, Mohamed A. Aly, and Amir F. Atiya. 2013. LABR: A Large Scale Arabic Book Reviews Dataset. In *Proceedings of ACL*, pages 494–498.

María Navas-Loro and Víctor Rodríguez-Doncel. 2019. Spanish corpora for sentiment analysis: a survey. *Language Resources and Evaluation*, pages 1–38.

Bo Pang and Lillian Lee. 2005. Seeing Stars: Exploiting Class Relationships For Sentiment Categorization With Respect To Rating Scales. In *Proceedings of ACL*, pages 115–124.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Haiyun Peng, Erik Cambria, and Amir Hussain. 2017. A Review of Sentiment Analysis Research in Chinese Language. *Cognitive Computation*, 9(4):423–435.

Marius Popescu, Cristian Grozea, and Radu Tudor Ionescu. 2017. HASKER: An efficient algorithm for string kernels. Application to polarity classification in various languages. In *Proceedings of KES*, pages 1755–1763.

David M.W. Powers. 1998. Applications and Explanations of Zipf's Law. In *Proceedings of CoNLL*, pages 151–160.

Yordan P. Raykov, Alexis Boukouvalas, Fahd Baig, and Max A. Little. 2016. What to Do When K-Means Clustering Fails: A Simple yet Principled Alternative Algorithm. *PLoS One*, 11(9):1–28.

Roxana Monica Russu, Mihaela Dinsoreanu, Oana Luminiţa Vlad, and Rodica Potolea. 2014. An opinion mining approach for Romanian language. In *Proceedings of ICCP*, pages 43–46.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. In *Proceedings of ACL*, pages 440–450.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, D. Christopher Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Marina Sokolova and Victoria Bobicev. 2009. Classification of Emotion Words in Russian and Romanian Languages. In *Proceedings of RANLP*, pages 416–420.

Marwan Torki. 2018. A Document Descriptor using Covariance of Word Vectors. In *Proceedings of ACL*, pages 527–532.

David Vilares, Miguel A. Alonso, and Carlos Gómez-Rodríguez. 2015. A syntactic approach for opinion mining on Spanish reviews. *Natural Language Engineering*, 21(1):139–163.

Xiaojun Wan. 2008. Using Bilingual Knowledge and Ensemble Techniques for Unsupervised Chinese Sentiment Analysis. In *Proceedings of EMNLP*, pages 553–561.

Salud Maria Jimenez Zafra, Teresa M. Martín-Valdivia, Eugenio Martinez Camara, and Alfonso L. Ureña-López. 2017. Studying the scope of negation for Spanish sentiment analysis on Twitter. *IEEE Transactions on Affective Computing*, 10(1):129–141.

Taras Zagibalov and John Carroll. 2008. Unsupervised classification of sentiment and objectivity in Chinese text. In *Proceedings of IJCNLP*, pages 304–311.

Zhongwu Zhai, Hua Xu, Bada Kang, and Peifa Jia. 2011. Exploiting effective features for Chinese sentiment classification. *Expert Systems with Applications*, 38(8):9139–9146.

Changli Zhang, Wanli Zuo, Tao Peng, and Fengling He. 2008. Sentiment Classification for Chinese Reviews Using Machine Learning Methods Based on String Kernel. In *Proceedings of ICCIT*, volume 2, pages 909–914.

Pu Zhang and Zhongshi He. 2013. A weakly supervised approach to Chinese sentiment classification using partitioned self-training. *Journal of Information Science*, 39(6):815–831.

Qianrong Zhou, Xiaojie Wang, and Xuan Dong. 2018. Differentiated Attentive Representation Learning for Sentence Classification. In *Proceedings of IJCAI*, pages 4630–4636.

Elastic weight consolidation for better bias inoculation

James Thorne
Department of Computer Science
University of Cambridge
`jt719@cam.ac.uk`

Andreas Vlachos
Department of Computer Science
University of Cambridge
`av308@cam.ac.uk`

Abstract

The biases present in training datasets have been shown to affect models for sentence pair classification tasks such as natural language inference (NLI) and fact verification. While fine-tuning models on additional data has been used to mitigate them, a common issue is that of catastrophic forgetting of the original training dataset. In this paper, we show that elastic weight consolidation (EWC) allows fine-tuning of models to mitigate biases while being less susceptible to catastrophic forgetting. In our evaluation on fact verification and NLI stress tests, we show that fine-tuning with EWC dominates standard fine-tuning, yielding models with lower levels of forgetting on the original (biased) dataset for equivalent gains in accuracy on the fine-tuning (unbiased) dataset.

1 Introduction

A number of recent works have illustrated shortcomings in sentence-pair classification models that are used for Natural Language Inference (NLI). These arise from limited or biased training data and the lack of suitable inductive bias in models. Naik et al. (2018) demonstrated that phenomena such as the presence of negation or a high degree of lexical overlap induce misclassifications on models trained on the MultiNLI dataset (Williams et al., 2018). Poliak et al. (2018) and Gururangan et al. (2018) identified biases introduced during dataset construction that were exploited by models to learn associations between the label and only one of the two input sentences without considering the other – known as *hypothesis-only* bias.

Such biases also affect fact verification (Schuster et al., 2019), typically modeled as a text-pair classification between a claim and evidence retrieved from a trusted source (Pomerleau and Rao, 2017; Thorne et al., 2018).

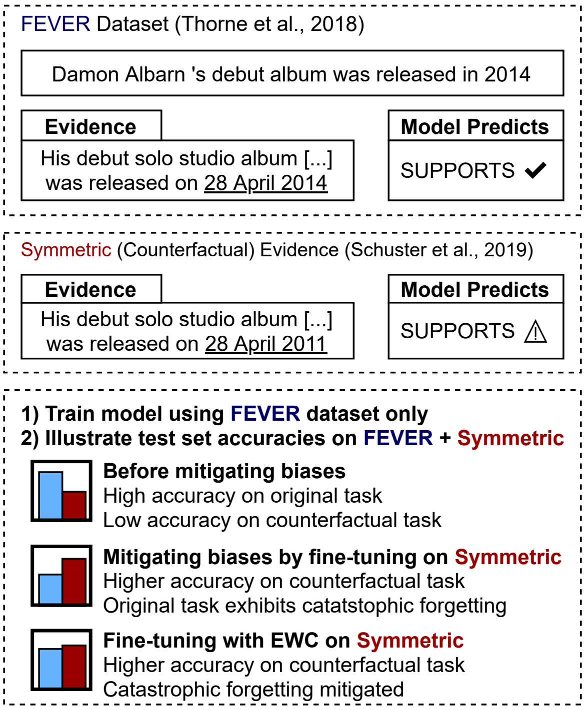


Figure 1: Hypothesis only bias in FEVER contributes to low accuracy when testing against counterfactual evidence. This is mitigated by fine-tuning on counterfactual evidence. Catastrophic forgetting from fine-tuning is reduced when using elastic weight consolidation (EWC), preserving the original task accuracy.

To mitigate these undesirable behaviors Liu et al. (2019a) fine-tune models with a small targeted number of labeled instances to “inoculate” the models. This can be contrasted to methods such as Debiased Focal Loss (DFL) and Product of Experts (POE) (Mahabadi et al., 2020) which require architectural changes to separately encode and penalize biases. In related work, Suntwal et al. (2019) delexicalize instances – replacing tokens with placeholders, preventing classifiers from exploiting mutual information between domain-specific noun-phrases and class labels. Finally, Schuster et al. (2019) re-weight the loss of in-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 957–964
April 19 - 23, 2021.

stances during training guided by the mutual information between tokens and the instance labels. Each of these methods corresponds to a multi-objective optimization problem where the original dataset accuracy is often sacrificed in favor of accuracy on a different evaluation set.

For fine-tuning specifically, the reduction in accuracy on the original task called catastrophic forgetting (French, 1999) as parameters for the original task are overridden during fine-tuning. In this paper, we show that regularizing fine-tuning with Elastic Weight Consolidation (Kirkpatrick et al., 2017, EWC) minimizes catastrophic forgetting by penalizing weight updates to parameters crucial to modeling the original dataset. EWC has previously been applied to machine translation, where Thompson et al. (2019) and Saunders et al. (2019) minimized catastrophic forgetting for models undergoing domain adaption. Extending this line of research further, we demonstrate that EWC can be used to mitigate biases present in two sentence-pair classification tasks without architectural changes to the models. We evaluate a number of popular model architectures, trained on MultiNLI and FEVER, and demonstrate that fine-tuning with EWC mitigates model bias while yielding lower reductions in accuracy on the original dataset compared to standard fine-tuning.

On all experiments on the FEVER dataset, fine-tuning with symmetric counterfactual data (Schuster et al., 2019) mitigated hypothesis-only bias, increasing the absolute accuracy of a BERT model by approximately 10%. Without EWC, accuracy on the original dataset was reduced from 87% to 79%, whereas with EWC catastrophic forgetting was mitigated and accuracy was 82%. In all experiments with EWC, the original task accuracy was significantly higher than fine-tuning without regularization and fine-tuning with L2 regularization. These gains were attained while maintaining similar performance on the fine-tuning data. Plotting the Pareto frontier, we show that equivalent gains in accuracy can be made with less forgetting of the original dataset. Furthermore, we demonstrate that fine-tuning methods can be combined with POE and DFL, yielding improvements on both the original task as well as fine-tuning data used for debiasing. Similar patterns were observed when fine-tuning MultiNLI models with lexical bias evaluation datasets (Naik et al., 2018).

2 Mitigating biases with fine-tuning

Fine-tuning broadly refers to approaches where a model is initially trained on one dataset and then further improved by training on another. We refer to these datasets as fine-tuning training and test data as *FT-train* and *FT-test* respectively. This technique is commonly used to mitigate model biases (Liu et al., 2019a), where the original data, while useful in model training, often contain biases, which are addressed by further training the model on a small set of instances targeting these biases. Fine-tuning to mitigate bias, however, can result in model parameters over-adjusting to the instances targeting it, reducing the accuracy on the original task, referred to as catastrophic forgetting (French, 1999). To ameliorate this issue, one can regularize the parameter updates so that they do not deviate too much from the original training, similar to the intuition behind multi-task training approaches (Ruder, 2017).

Elastic Weight Consolidation (Kirkpatrick et al., 2017, EWC) penalizes parameter updates according to the model's sensitivity to changes in these parameters. The model sensitivity is estimated by the Fisher information matrix, which describes the model's expected sensitivity to a change in parameters, and near the (local) minimum of the loss function used for training is equivalent to the second-order derivative $F = \mathbb{E}_{(x,y)\sim\mathcal{D}_{original}}[\nabla^2 \log p(y|x;\theta)]$. When fine-tuning with EWC (which we refer to as FT+EWC), the Fisher information is used to elastically scale the cost of updating parameters θ_i from the original value θ_i^*, controlled by the λ hyper-parameter, as follows:

$$\mathcal{L}(\theta) = \mathcal{L}_{FT}(\theta) + \sum_i \frac{\lambda}{2} F_{i,i}(\theta_i - \theta_i^*)^2 \tag{1}$$

For efficiency, we use the *empirical Fisher* (Martens, 2014): diagonal elements are approximated through squaring first-order gradients from a sample of instances, recomputed before each epoch. If the Fisher information is not used (i.e. $F_{i,i} = 1$), Eq. 1 is equivalent to L2 regularization (which we refer to as FT+L2).

3 Experimental setup

We assess the application of EWC to minimize catastrophic forgetting when mitigating model biases in the context of two sentence-pair classification tasks: fact verification and natural language inference. We compare the untreated model (original), fine-tuning (FT), FT+EWC, FT+L2, and merging instances from the FT-train dataset when training (Merged). Each model is first trained using the original dataset and splits from the respective task, using the AllenNLP implementations (Gardner et al., 2017) with default hyper-parameters and tokenize with SpaCy or pre-trained transformer tokenizers. We train five random initializations of each model, reporting the mean accuracy, standard deviation, and p-value with an unpaired t-test. For fine-tuning, the learning rate, regularization strength λ, and number of epochs are selected through 5-fold cross-validation on the FT-train data, selecting the model with the highest FT-train accuracy. 30 hyper parameter choices were evaluated with grid search over 10 choices for regularization strength between 10^6 and 10^8 and 3 choices of learning rates in $\{2 \cdot 10^{-6}, 4 \cdot 10^{-6}, 6 \cdot 10^{-6}\}$ for transformer models and $\{2 \cdot 10^{-4}, 4 \cdot 10^{-4}, 6 \cdot 10^{-4}\}$ for ESIM models. We trained the models for a max of 8 epochs. For the transformer-based models, the highest cross validation accuracy on the FT-train dataset was achieved with $LR = 4 \cdot 10^{-6}$, $\lambda = 10^7$ and 6 epochs. For the ESIM-based models, the highest FT-train accuracy was achieved with $LR = 2 \cdot 10^{-4}$, $\lambda = 10^7$ and 5 epochs. Full hyper-parameter choices are in Appendix D.

Mitigating hypothesis only bias in fact verification: The FEVER[1] task (Thorne et al., 2018) is to predict whether short factoid sentences called claims are Supported or Refuted against evidence (in the form of sentences from Wikipedia) or whether there is not enough info (NEI). When training the models, the NEI instances by definition don't have evidence: we sample negative instances for these with random sentences from the Wikipedia page closest to the claim using TF·IDF. This preprocessing is the same as Thorne et al. (2018). Schuster et al. (2019) identified a bias where the label for some claims can be predicted without the need for evidence. To evaluate this bias, they released[2] a set of 1420 symmetric counterfactual instances where each claim is supported by one Wikipedia passage and refuted by another (approximately 1% of the FEVER dataset). This is mitigating the claim-only bias by reducing the mutual information between claims and labels. The availability of counterfactual data meant that it is possible to experiment with fine-tuning as a mitigation strategy, using the published dev and test data as FT-train and FT-test respectively. Following Schuster et al. (2019)'s evaluation, we train two ESIM (Enhanced LSTM) variants (Chen et al., 2016), and a BERT (Devlin et al., 2019) transformer. We also evaluate RoBERTa (Liu et al., 2019b), as it has been shown to be more robust to adversarial testing (Bartolo et al., 2020).

Mitigating model limitations in NLI stress tests: The MultiNLI[3] task (Williams et al., 2018) requires systems to predict whether a hypothesis is entailed by a premise. Naik et al. (2018) identify limitations of models trained on this dataset where 6 phenomena such as *lexical overlap*, *numerical reasoning* and presence of *antonyms* were evaluated with 'stress-tests'. In this paper, we report on *antonyms* and *numerical reasoning* as these stress-tests exhibited catastrophic forgetting when used to fine-tune models (Liu et al., 2019a). To this end, we do not evaluate on HANS (McCoy et al., 2020), as high accuracies can be attained without forgetting.

Like Liu et al. (2019a), we mitigate these biases through fine-tuning both an ESIM (Chen et al., 2016) model on stress-test data[4]. Each contains a small number of procedurally generated instances (between 1500-9800) that specifically target one of these phenomena. We evaluate FT and FT+EWC using the same methodology, controlling the number of instances, sampled at random, in FT-train and report the change in accuracy on the FT-test and original test sets.

4 Results

4.1 Fact verification

Fine-tuning the models, rather than merging datasets, yielded the greatest improvements in accuracy on FT-test. All improvements from the untreated model were significant ($p < 0.05$, denoted #). Without L2 or EWC, catastrophic forgetting occurs due to the shift in label distribution between the FEVER and FT-train dataset, which only contains 2 of the original 3 label classes.

---

[1] `https://fever.ai/`

[2] `https://github.com/TalSchuster/FeverSymmetric/`

[3] `https://cims.nyu.edu/~sbowman/multinli/`

[4] `https://abhilasharavichander.github.io/NLI_StressTest/`

Model	FEVER Dataset (Original Task) Accuracy (%)				
	Original	Merged	FineTune	FineTune+L2	FineTune+EWC
ESIM+GloVe	79.94 ± 0.4	79.57 ± 0.4	70.78 ± 1.1	$73.29 \pm 0.4^{*}$	$74.64 \pm 0.7^{*\dagger}$
ESIM+ELMo	80.15 ± 0.2	80.33 ± 0.8	76.45 ± 0.8	$73.72 \pm 0.6^{*}$	$78.09 \pm 0.4^{*\dagger}$
BERT Base	86.88 ± 0.5	86.87 ± 0.5	78.82 ± 0.9	$79.90 \pm 1.4^{*}$	$82.23 \pm 1.1^{*\dagger}$
RoBERTa Base	88.12 ± 0.3	88.11 ± 0.1	82.51 ± 1.5	$83.14 \pm 1.4^{*}$	$85.12 \pm 1.1^{*\dagger}$
	Symmetric Dataset (Fine-tuning Task) Accuracy (%)				
ESIM+GloVe	68.37 ± 1.0	69.35 ± 0.5	$74.21 \pm 1.3^{\#}$	$73.34 \pm 1.2^{\#\diamondsuit}$	$73.20 \pm 1.4^{\#\diamondsuit\heartsuit}$
ESIM+ELMo	64.04 ± 0.7	$66.46 \pm 1.3^{\#}$	$68.68 \pm 0.7^{\#}$	$70.31 \pm 0.5^{\#}$	$69.16 \pm 0.7^{\#}$
BERT Base	74.77 ± 1.4	$79.24 \pm 0.7^{\#}$	$87.07 \pm 0.6^{\#}$	$86.66 \pm 0.4^{\#\diamondsuit}$	$85.11 \pm 0.4^{\#}$
RoBERTa Base	78.34 ± 0.2	$87.03 \pm 2.3^{\#}$	$91.01 \pm 0.6^{\#}$	$90.98 \pm 0.5^{\#\diamondsuit}$	$89.63 \pm 1.3^{\#\diamondsuit\heartsuit}$

Table 1: Bias mitigation for FEVER classifiers comparing no treatment (original), against merging from instances from the FT-train with the original task training dataset (Merged) and FineTuning (with EWC and L2). Improvements $p < 0.05$ are marked with the following symbols: $*$ against FT, $\dagger$ against FT+L2, $\#$ against original. Deteriorations $p > 0.05$ on the symmetric dataset are marked with $\diamondsuit$ against FT and $\heartsuit$ against FT+L2

Both L2 and EWC reduced catastrophic forgetting. Improvements on the original task are significant (denoted $*$) compared to FT. However, EWC regularization retained more of the original task accuracy than L2 for all models; this was also significant (denoted $\dagger$). In all cases, there is a trade-off between original and fine-tuning task accuracies. With regularization, the FT accuracy was higher than FT+L2 and FT+EWC (with the exception of ESIM+ELMo). However, the deterioration from FT when regularizing was not significant ($p > 0.05$, denoted $\diamondsuit$). Furthermore, for the highest performing model (RoBERTa base), the deterioration of using FT+EWC against FT+L2 was also not significant (denoted $\heartsuit$).

Training a model where the FEVER training and FT-train were merged yielded modest improvements on the FT-test without harming the original FEVER task accuracy. We attribute this to the impact of these 700 instances being diluted by the large number of training instances in FEVER (FT-train is <1% the size of FEVER).

4.2 Combining FT and bias modeling

Fine-tuning can be applied to any task using a small amount of bias-mitigating labeled data, whereas explicit modeling of hypothesis-only biases (Mahabadi et al., 2020) requires architectural changes that are specific to the task and model. We consider two techniques from Mahabadi et al. (2020): Product of Experts (PoE), which multiplies the sentence-pair label probabilities with hypothesis-only label probabilities, and Debiased Focal Loss (DFL) which explicitly modulates the loss of instances according to the accuracy of a hypothesis-only classifier. For both ESIM and BERT, accuracy, when trained with PoE, was stable across different choices of hyper-parameters, whereas some hyper-parameter choices for DFL resulted in lower accuracy on both tasks. We report results for BERT+PoE and ESIM+DFL as these were best.

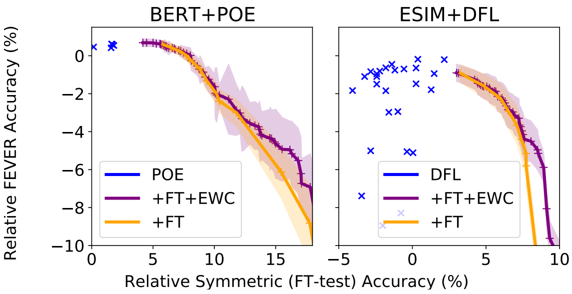


Figure 2: Pareto frontiers of fine-tuning BERT and ESIM models showing FT+EWC dominates FT.

The trade-off between accuracy on the original and FT-test datasets is visualized in Figure 2 indicating the change in accuracy for both bias modeling techniques in isolation, as well as in combination with fine-tuning.[5] This further shows that FT+EWC Pareto dominates FT for both the ESIM and BERT model. With EWC, equivalent gains on the symmetric FT-test can be attained with a lower

---

[5] All Pareto frontier lines are the result of sweeping the following: for POE, β; for DFL, β and γ; for FT, learning rate; and for FT+EWC, learning rate + EWC strength (λ in Equation 1) in Appendix D.8.

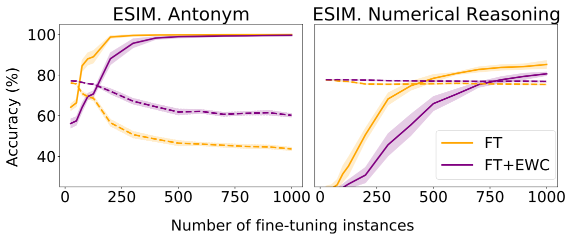


Figure 3: Training curves fine tuning MultiNLI with stress-test data. Solid lines indicate challenge dataset accuracy. Dashed lines indicate MultiNLI accuracy.

reduction in accuracy on the original FEVER task.

4.3 Natural Language Inference (NLI)

In a separate experiment, we apply EWC to a different domain. We inoculate biases on the ESIM model for natural language inference reported in Liu et al. (2019a). For both *Antonym* and *Numerical Reasoning* challenges, MultiNLI accuracy was higher with FT+EWC compared to FT (dashed lines in Figure 3).

Antonym challenge: The ESIM model was sensitive to fine-tuning, attaining near perfect accuracy (top row of Figure 3) on the FT-test data. The antonym stress-test only contains instances labeled *contradiction*: a change in label distribution that causes catastrophic forgetting. Without EWC, accuracy on MultiNLI fell to just above chance levels as the model learned only to predict contradiction (yellow dashed line). However, using an appropriate EWC penalty attained near-perfect accuracy with a smaller reduction in MultiNLI accuracy (purple dashed line).

Numerical reasoning challenge: The ESIM model was sensitive to fine-tuning to introduce numerical reasoning behaviours to the model. As the difference in label distribution in the inoculation dataset was less severe than the Antonym dataset, the catastrophic forgetting was less severe. Nevertheless, FT+EWC minimized catastrophic forgetting at the expense of reducing sample efficiency: accuracy on MultiNLI fell from 77.9% to 75.4% without EWC and 76.8% with EWC.

5 Conclusions

Fine-tuning can be used to mitigate model bias but has the risk that the model catastrophically forgets the data it was originally trained on. Incorporating elastic weight consolidation (EWC) when fine-tuning minimizes catastrophic forgetting, yielding higher accuracy on the original task. We show this holds for both the NLI stress-tests, as well debiasing fact-verification systems (Schuster et al., 2019) through fine-tuning.

Acknowledgements

James Thorne is funded by an Amazon Alexa Graduate Fellowship. Andreas Vlachos is supported by the ERC grant AVeriTeC. Thank you to Danielle Saunders for the insightful discussions.

References

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is Your Evidence: Improving Fact-checking by Justification Modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium. Association for Computational Linguistics.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. MultiFC: A Real-World Multi-Domain Dataset for Evidence-Based Fact Checking of Claims. In *Proceedings of EMNLP-IJCNLP*, Hong Kong, China. Association for Computational Linguistics.

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating Adversarial Human Annotations for Reading Comprehension.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced LSTM for Natural Language Inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, page 1657–1668, Vancouver, Canada.

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

Robert M French. 1999. Catastrophic forgetting in connectionists networks. *Trends in Cognitive Sciences*, 3(4):128–135.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2017. AllenNLP: A Deep Semantic Natural Language Processing Platform.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation Artifacts in Natural Language Inference Data.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by Fine-Tuning: A Method for Analyzing Challenge Datasets. pages 2171–2179.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-End Bias Mitigation by Modelling Biases in Corpora. In *Proceedings of the 58th meeting of the Association for Computational Linguistics*, Seattle, WA. Association for Computational Linguistics.

James Martens. 2014. New insights and perspectives on the natural gradient method.

Thomas R. McCoy, Ellie Pavlick, and Tal Linzen. 2020. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2, pages 3428–3448.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress Test Evaluation for Natural Language Inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis Only Baselines in Natural Language Inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Dean Pomerleau and Delip Rao. 2017. Fake News Challenge. http://fakenewschallenge.org/.

Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. May.

Danielle Saunders, Felix Stahlberg, Adrià de Gispert, and Bill Byrne. 2019. Domain Adaptive Inference for Neural Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 222–228, Florence, Italy. Association for Computational Linguistics.

Tal Schuster, Darsh J Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. 2019. Towards Debiasing Fact Verification Models. In *Proceedings of EMNLP-IJCNLP*, Hong Kong, China.

Sandeep Suntwal, Mithun Paul, Rebecca Sharp, and Mihai Surdeanu. 2019. On the Importance of Delexicalization for Fact Verification. pages 3411–3416.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming Catastrophic Forgetting During Domain Adaptation of Neural Machine Translation. (2018):2062–2068.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

A Hypothesis-only bias in other fact-verification datasets

We use the FEVER dataset in this paper due to the availability of the symmetric counterfactual data released by Schuster et al. (2019). Other datasets for fake news detection and fact verification also have the same task signature (sentence-pair classification with a claim and another sentence), but second sentence is not evidential. All tasks exhibit a hypothesis-only bias where information from the claim can be used to predict the label without considering the second sentence in the sentence-pair. If there was no mutual information between claims (without evidence) and labels, this should be 33%.

For FEVER, this bias is introduced through synthetic generation of the claims and is more problematic than the biases that occur in the datasets consisting of naturally occurring claims. In Liar and MultiFC, the claims arise from real-world events and the biases in the data reflect political viewpoints, rather than cognitive shortcuts taken by the FEVER annotators.

The RoBERTa model, trained on only the claims outperforms the sentence-pair setup for both the MultiFC (Augenstein et al., 2019) and Liar-Plus tasks (Alhindi et al., 2018) whereas for the the FEVER data the sentence-pair accuracy is higher. As FEVER is the only task that requires the use of evidence for classification, this is expected.

Dataset	Accuracy (%)	
	Claim Only	Sentence Pair
Liar-Plus	28.74	20.48
Liar-Plus (binary)	72.59	70.48
MultiFC	46.02	44.83
FEVER	61.50	88.93
FEVER (2-way)	79.09	92.24

Table 2: Validation accuracy for claim-only vs sentence pair classification for fact verification datasets trained on RoBERTa. For 2-way fever we discard instances labelled NOTENOUGHINFO. For binary Liar-Plus, we map all positive labels to true and all negative labels to false and discard neutral instances.

B Compute infrastructure

All experiments were performed on a single workstation with a single Xeon E5-2630 CPU, 64GB RAM and an Nvidia 1080Ti GPU.

C Average run-time for fine-tuning

Estimating the Fisher matrix diagonal takes approximately 2 seconds for the ESIM model using 2000 instances sampled from the original dataset. Average training duration (excluding estimating Fisher information) was 11 seconds in total with 700 instances.

Estimating the Fisher matrix diagonal takes approximately 25 seconds for the BERT and RoBERTa models using 2000 instances sampled from the original dataset. Average training duration (excluding estimating Fisher information) was 2 minutes 20 seconds in total with 700 instances.

D Hyper-parameter configurations

D.1 Base models

For the base-models, the default hyperparemters in AllenNLP are used for the ESIM, BERT and RoBERTa models.

ESIM

- Embedding dimension: 300, bidirectional
- Dropout: 0.5
- Optimizer: Adam
- Gradient Norm: 10.0
- Batch Size: 64
- Learning Rate: 0.0004
- Learning Rate Schedule: reduce on plateau, patience 0, factor 0.5
- Number of Epochs: 75
- Early Stopping: Patience 10

BERT+RoBERTa

- Embedding dimension: 768
- Optimizer: AdamW
- Gradient Norm: 10.0
- Batch Size: 8
- Learning Rate: 0.0004
- Learning Rate Schedule: slanted triangular, cut frac 0.06
- Number of Epochs: 5
- Early Stopping: Patience 0

D.2 Fine-tuning without EWC

ESIM

- FT Learning Rate: 0.0002
- FT Epochs: 8

BERT

- FT Learning Rate: 0.000004
- FT Epochs: 6

RoBERTa

- FT Learning Rate: 0.000004
- FT Epochs: 7

D.3 Fine-tuning using EWC

ESIM

- FT Learning Rate: 0.0002
- FT Epochs: 5
- EWC: 10000000

BERT+RoBERTa

- FT Learning Rate: 0.000004
- FT Epochs: 6
- EWC: 10000000

D.4 Unsupervised bias mitigation

Hyperparameters: β controls the weight update for the hypothesis-only model and γ controls the modulation of hypothesis-only model in the loss function.

- POE (BERT) $\beta = 0.4$
- POE (ESIM) $\beta = 0.05$
- DFL (BERT) $\beta = 0.4, \gamma = 0.6$
- DFL (ESIM) $\beta = 0.05, \gamma = 2.0$

D.5 Search bounds for fine-tuning

Approximately 30 configurations were considered (Cartesian product of LR+EWC). The best performing system was selected through max accuracy on the FT cross validation dataset through 5 fold cross validation.

- EWC $\{10^6, 2\cdot 10^6, 4\cdot 10^6, 8\cdot 10^6, 10^7, 2\cdot 10^7, 4\cdot 10^7, 6\cdot 10^7, 8\cdot 10^7, 10^8\}$
- Fine-tuning learning rate (ESIMs): $\{0.0002, 0.0004, 0.0006\}$
- Fine-tuning learning rate (Transformer): $\{0.000002, 0.000004, 0.000006\}$
- Epochs: Up to 10 epochs cross valudating on the FT-training dataset

D.6 Search bounds for end2end bias mitigation

We used the same range of values published by Mahabadi et al. (2020). For DFL, we performed a grid search of every value totalling 30 trials.
$\gamma \in \{0.02, 0.05, 0.1, 0.6, 2.0, 4.0, 5.0\}$
$\beta \in \{0.05, 0.2, 0.4, 0.8, 1.0\}$

D.7 Stress test sizes

Following the evaluation of (Liu et al., 2019a) we vary the number of instances from the stress test (between 1500-9800) instances. To plot Figure 3, we use 25, 50, 75, 100, 250, 400, 500, 600, 700, 800, 900, 1000 instances in our evaluation.

D.8 Pareto frontier sweeps

The Pareto frontier sweeps in Figure 2 were generated by plotting all hyperparameters from Appendix D.5. The end2end (blue) crosses are plotted by plotting the choices from Appendix D.6 in combination with the best FT model.

Hierarchical Multi-head Attentive Network for Evidence-aware Fake News Detection

Nguyen Vo
Worcester Polytechnic Institute
Computer Science Department
Worcester, MA, USA, 01609
`nkvo@wpi.edu`

Kyumin Lee
Worcester Polytechnic Institute
Computer Science Department
Worcester, MA, USA, 01609
`kmlee@wpi.edu`

Abstract

The widespread of fake news and misinformation in various domains ranging from politics, economics to public health has posed an urgent need to automatically fact-check information. A recent trend in fake news detection is to utilize evidence from external sources. However, existing evidence-aware fake news detection methods focused on either only word-level attention or evidence-level attention, which may result in suboptimal performance. In this paper, we propose a Hierarchical Multi-head Attentive Network to fact-check textual claims. Our model jointly combines multi-head word-level attention and multi-head document-level attention, which aid explanation in both word-level and evidence-level. Experiments on two real-word datasets show that our model outperforms seven state-of-the-art baselines. Improvements over baselines are from 6% to 18%. Our source code and datasets are released at `https://github.com/nguyenvo09/EACL2021`.

1 Introduction

The proliferation of biased news, misleading claims, disinformation and fake news has caused heightened negative effects on modern society in various domains ranging from politics, economics to public health. A recent study showed that maliciously fabricated and partisan stories possibly caused citizens' misperception about political candidates (Allcott and Gentzkow, 2017) during the 2016 U.S. presidential elections. In economics, the spread of fake news has manipulated stock price (Kogan et al., 2019). For example, $139 billion was wiped out when the Associated Press (AP)'s hacked Twitter account posted rumor about White House explosion with Barack Obama's injury. Recently, misinformation has caused infodemics in public health (Ashoka, 2020) and even led to people's fatalities in the physical world (Alluri, 2019).

To reduce the spread of misinformation and its detrimental influences, many fact-checking systems have been developed to fact-check textual claims. It is estimated that the number of fact-checking outlets has increased 400% in 60 countries since 2014 (Stencel, 2019). Several fact-checking systems such as *snopes.com* and *politifact.com* are widely used by both online users and major corporations. Facebook (CNN, 2020) recently incorporated third-party fact-checking sites to social media posts and Google integrated fact-checking articles to their search engine (Wang et al., 2018). These fact-checking systems debunk claims by manually assess their credibility based on collected webpages used as evidence. However, this manual process is laborious and unscalable to handle the large volume of produced false claims on communication platforms. Therefore, in this paper, our goal is to build an automatic fake news detection system to fact-check textual claims based on collected evidence to speed up fact-checking process of the above fact-checking sites.

To detect fake news, researchers proposed to use linguistics and textual content (Castillo et al., 2011; Zhao et al., 2015; Liu et al., 2015). Since textual claims are usually deliberately written to deceive readers, it is hard to detect fake news by solely relying on the content claims. Therefore, multiple works utilized other signals such as temporal spreading patterns (Liu and Wu, 2018), network structures (Wu and Liu, 2018; Vo and Lee, 2018; Shu et al., 2020) and users' feedbacks (Vo and Lee, 2019; Shu et al., 2019; Vo and Lee, 2020a). However, limited work used external webpages as documents which could provide interpretive explanation to users. Several recent work (Popat et al., 2018; Ma et al., 2019; Vo and Lee, 2020b) started to utilize documents to fact-check textual claims. Popat et al. (2018) used word-level attention in documents but treated all documents with equal im-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 965–975
April 19 - 23, 2021.

portance whereas Ma et al. (2019) only focused on which documents are more crucial without considering what words help explain credibility of textual claims.

Observing drawbacks of the existing work, we propose Hierarchical Multi-head Attentive Network which jointly utilizes word attention and evidence attention. Overall semantics of a document may be generated by multiple parts of the document. Therefore, we propose a multi-head word attention mechanism to capture different semantic contributions of words to the meaning of the documents. Since a document may have different semantic aspects corresponding to various information related to credibility of a claim, we propose a multi-head document-level attention mechanism to capture contributions of the different semantic aspects of the documents. In our attention mechanism, we also use speakers and publishers information to further improve effectiveness of our model. To our knowledge, our work is the first applying multi-head attention mechanism for both words and documents in evidence-aware fake news detection. Our work makes the following contributions:

- We propose a novel hierarchical multi-head attention network which jointly combines word attention and evidence attention for evidence-aware fake news detection.
- We propose a novel multi-head attention mechanism to capture important words and evidence.
- Experiments on two public datasets demonstrate the effectiveness and generality of our model over state-of-the-art fake news detection techniques.

2 Related Work

Many methods have been proposed to detect fake news in recent years. These methods can be placed into three groups: (1) human-based fact-checking sites (e.g. Snopes.com, Politifact.com), (2) machine learning based methods and (3) hybrid systems (e.g. content moderation on social media sites). In machine-learning-based methods, researchers mainly used linguistics and textual content (Zellers et al., 2019; Zhao et al., 2015; Wang, 2017; Shu et al., 2019), temporal spreading patterns (Liu and Wu, 2018), network structures (Wu and Liu, 2018; Vo and Lee, 2018; You et al., 2019), users' feedbacks (Vo and Lee, 2019; Shu et al., 2019) and multimodal signals (Gupta et al., 2013; Vo and Lee, 2020b). Recently, researchers focus on fact-checking claims based on evidence from different sources. Thorne and Vlachos (2017) and Vlachos and Riedel (2015) fact-check claims using subject-predicate-object triplets extracted from knowledge graph as evidence. Chen et al. (2020) assess claims' credibility using tabular data. Our work is closely related to fact verification task (Thorne et al., 2018; Nie et al., 2019; Soleimani et al., 2020) which aims to classify a pair of a claim and an evidence extracted from Wikipedia into three classes: *supported*, *refuted*, or *not enough info*. For fact verification task, Nie et al. (2019) used ELMo (Peters et al., 2018) to extract contextual embeddings of words and used a modified ESIM model (Chen et al., 2017). Soleimani et al. (2020) used BERT model (Devlin et al., 2018) to retrieve and verify claims. Zhou et al. (2019) used graph based models for semantic reasoning. Our work is different from these work since our goal is to classify a pair of a claim and a list of relevant evidence into *true* or *false*.

Our work is close to existing work about evidence-aware fake news detection (Popat et al., 2018; Ma et al., 2019; Wu et al., 2020; Mishra and Setty, 2019). Popat et al. (2018) used an average pooling layer to derive claims' representation to attend to words in evidence, Mishra and Setty (2019) focused on words and sentences in each evidence, and Ma et al. (2019) proposed a semantic entailment model to attend to important evidence. However, to the best of our knowledge, our work is the first jointly using multi-head attention mechanisms to focus on important words in each evidence and important evidence from a set of relevant articles. Our attention mechanism is different from these work since we use multiple attention heads to capture different semantic contributions of words and evidence.

3 Problem Statement

We denote an evidence-based fact-checking dataset $\mathcal{C}$ as a collection of tuples $(c, s, \mathcal{D}, \mathcal{P})$ where c is a textual claim originated from a speaker s, $\mathcal{D} = \{d_i\}_{i=1}^{k}$ is a collection of k documents[1] relevant to the claim c and $\mathcal{P} = \{p_i\}_{i=1}^{k}$ is the corresponding publishers of documents in $\mathcal{D}$. Note, $|\mathcal{D}| = |\mathcal{P}|$. Our goal is to classify each tuple $(c, s, \mathcal{D}, \mathcal{P})$ into a pre-defined class (i.e. true news/fake news).

---

[1] We use the term "documents", "articles", and "evidence" interchangeably.

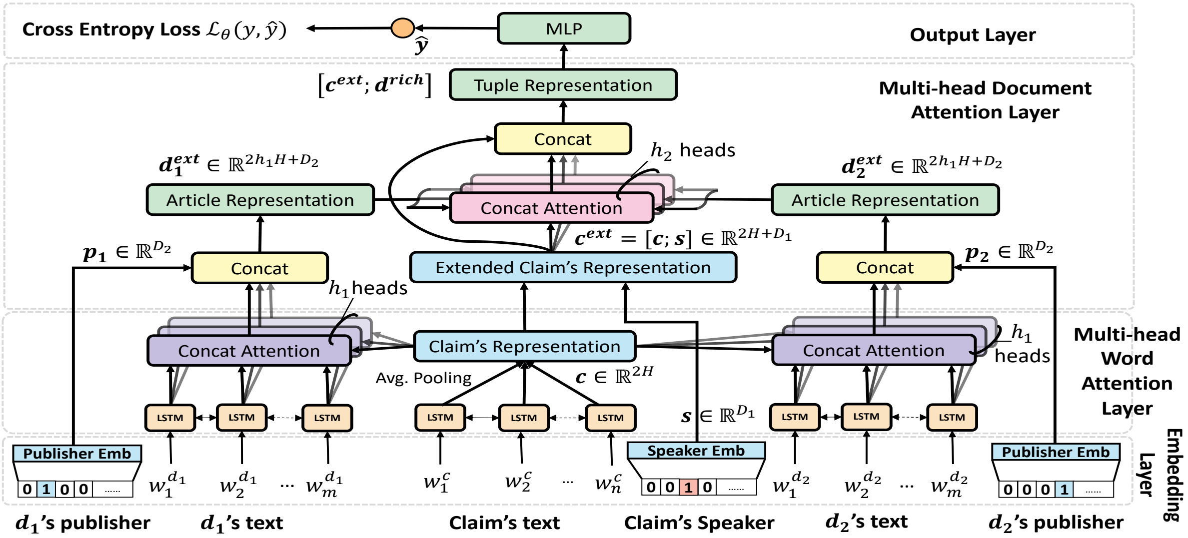


Figure 1: The architecture of our proposed model **MAC** in which we show a claim c, two associated relevant articles d_1 and d_2 and sources of the claim and the two documents. h_1 and h_2 are the number of heads of word-level attention and document-level attention respectively.

4 Framework

In this section, we describe our Hierarchical **M**ulti-head **A**ttentive Network for Fact-**C**hecking (**MAC**) which jointly considers word-level attention and document-level attention. Our framework consists of four main components: (1) embedding layer, (2) multi-head word attention layer, (3) multi-head document attention layer and (4) output layer. These components are illustrated in Fig. 1 where we show a claim and two documents as an example.

4.1 Embedding Layer

Each claim c is modeled as a sequence of n words $[w_1^c, w_2^c, ..., w_n^c]$ and d_i is viewed as another sequence of m words $[w_1^d, w_2^d, ..., w_m^d]$. Each word w_i^c and w_j^d will be projected into D-dimensional vectors $\mathbf{e}_i^c$ and $\mathbf{e}_j^d$ respectively by an embedding matrix $\mathbf{W}_e \in \mathbb{R}^{V \times D}$ where V is the vocabulary size. Each speaker s and publisher p_i modeled as one-hot vectors are transformed into dense vectors $\mathbf{s} \in \mathbb{R}^{D_1}$ and $\mathbf{p}_i \in \mathbb{R}^{D_2}$ respectively by using two matrices $\mathbf{W}_s \in \mathbb{R}^{S \times D_1}$ and $\mathbf{W}_p \in \mathbb{R}^{P \times D_2}$, where S and P are the number of speakers and publishers in a training set respectively. Both $\mathbf{W}_s$ and $\mathbf{W}_p$ are uniformly initialized in $[-0.2, 0.2]$. Note that, both matrices $\mathbf{W}_s$ and $\mathbf{W}_p$ are jointly learned with other parameters of our MAC.

4.2 Multi-head Word Attention Layer

We input word embeddings $\mathbf{e}_i^c$ of the claim c into a bidirectional LSTM (Graves et al., 2005) which helps generate contextual representation $\mathbf{h}_i$ of each token as follows: $\mathbf{h}_i^c = [\overleftarrow{\mathbf{h}_i}; \overrightarrow{\mathbf{h}_i}] \in \mathbb{R}^{2H}$, where $\overleftarrow{\mathbf{h}}_i$ and $\overrightarrow{\mathbf{h}}_i$ are hidden states in forward and backward pass of the BiLSTM, symbol ; means concatenation and H is hidden size. We derive claim's representation in $\mathbb{R}^{2H}$ by an average pooling layer as follows:

$$\mathbf{c} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{h}_i^c \tag{1}$$

Applying a similar process on the top of each document d_i with a different BiLSTM, we have contextual representation $\mathbf{h}_j^d \in \mathbb{R}^{2H}$ for each word in d_i. After going through BiLSTM, d_i is modeled as matrix $\mathbf{H} = [\mathbf{h}_1^d \oplus \mathbf{h}_2^d \oplus ... \oplus \mathbf{h}_m^d] \in \mathbb{R}^{m \times 2H}$ where $\oplus$ denotes stacking.

To understand what information in a document helps us fact-check a claim, we need to guide our model to focus on crucial keywords or phrases of the document. Drawing inspiration from (Luong et al., 2015), we firstly replicate vector $\mathbf{c}$ (Eq.1) m times to create matrix $\mathbf{C}_1 \in \mathbb{R}^{m \times 2H}$ and propose an attention mechanism to attend to important words in the document d_i as follows:

$$\mathbf{a}_1 = softmax\big(\tanh\big([\mathbf{H}; \mathbf{C}_1] \cdot \mathbf{W}_1\big) \cdot \mathbf{w}_2\big) \tag{2}$$

where $\mathbf{w}_2 \in \mathbb{R}^{a_1}$, $\mathbf{W}_1 \in \mathbb{R}^{4H \times a_1}$, $[\mathbf{H}; \mathbf{C}_1]$ is concatenation of two matrices on the last dimension and $\mathbf{a}_1 \in \mathbb{R}^m$ is attention distribution on m words. However, the overall semantics of the document might be generated by multiple parts of the document (Lin et al., 2017). Therefore, we propose a multi-head word attention mechanism to capture different semantic contributions of words by extending vector $\mathbf{w}_2$ into a matrix $\mathbf{W}_2 \in \mathbb{R}^{a_1 \times h_1}$ where h_1 is the number of attention heads shown in Fig. 1. We modify Eq. 2 as follows:

$$\mathbf{A}_1 = softmax_{col}\big(\tanh([\mathbf{H}; \mathbf{C}_1] \cdot \mathbf{W}_1) \cdot \mathbf{W}_2\big) \quad (3)$$

where $\mathbf{A}_1 \in \mathbb{R}^{m \times h_1}$ and each column of $\mathbf{A}_1$ has been normalized by the softmax operation. Intuitively, $\mathbf{A}_1$ stands for h_1 different attention distributions on top of m words of the document d_i, helping us capture different aspects of the document. After computing $\mathbf{A}_1$, we derive representation of document d_i as follows:

$$\mathbf{d}_i = flatten(\mathbf{A}_1^T \cdot \mathbf{H}) \quad (4)$$

where $\mathbf{d}_i \in \mathbb{R}^{h_1 2H}$ and function *flatten(.)* flattens $\mathbf{A}_1^T \cdot \mathbf{H}$ into a vector. We also implemented a more sophisticated multi-head attention in (Vaswani et al., 2017) but did not achieve good results.

4.3 Multi-head Document Attention Layer

This layer consists of three components as follows: (1) extending representations of claims, (2) extending representations of evidence and (3) multi-head document attention mechanism.

Extending representations of claims. So far the representation of the claim $\mathbf{c}$ (Eq. 1) is only from textual content. In reality, a speaker who made a claim may impact credibility of the claim. For example, claims from some politicians are controversial and inaccurate (Allcott and Gentzkow, 2017). Therefore, we enrich vector $\mathbf{c}$ by concatenating it with speaker's embedding $\mathbf{s}$ to generate $\mathbf{c}^{ext} \in \mathbb{R}^x$, where $x = 2H + D_1$ as shown in Eq. 5.

$$\mathbf{c}^{ext} = [\mathbf{c}; \mathbf{s}] \in \mathbb{R}^x \quad (5)$$

Extending representations of evidence. Intuitively, an article published by *nytimes.com* might be more reliable than a piece of news published by *breitbart.com* which is known to be a less credible site. Therefore, to capture more information, we further enrich representations of evidence with publishers' information by concatenating $\mathbf{d}_i$ (Eq. 4) with its publisher's embedding $\mathbf{p}_i$ as follows:

$$\mathbf{d}_i^{ext} = [\mathbf{d}_i; \mathbf{p}_i] \in \mathbb{R}^y \quad (6)$$

where $y = 2h_1H + D_2$. From Eq. 6, we can generate representations of k relevant articles and stack them as shown in Eq. 7.

$$\mathbf{D} = [\mathbf{d}_1^{ext} \oplus ... \oplus \mathbf{d}_k^{ext}] \in \mathbb{R}^{k \times y} \quad (7)$$

Multi-head Document Attention Mechanism. In real life, a journalist from *snopes.com* and *politifact.com* may use all k articles relevant to the claim c to fact-check it but she may focus on some key articles to determine the verdict of the claim c while other articles may have negligible information. To capture such intuition, we need to downgrade uninformative documents and concentrate on more meaningful articles. Similar to Section 4.2, we use multi-head attention mechanism which produces different attention distributions representing diverse contributions of articles toward determining veracity of the claim c.

We firstly create matrix $\mathbf{C}_2 \in \mathbb{R}^{k \times x}$ by replicating vector $\mathbf{c}^{ext}$ (Eq. 5) k times. Secondly, the matrix $\mathbf{C}_2$ is concatenated with matrix $\mathbf{D}$ (Eq. 7) on the last dimension of the two matrices denoted as $[\mathbf{D}; \mathbf{C}_2] \in \mathbb{R}^{k \times (x+y)}$.

Our proposed multi-head document-level attention mechanism applies h_2 different attention heads as shown in Eq. 8.

$$\mathbf{A}_2 = softmax_{col}(\tanh([\mathbf{D}; \mathbf{C}_2] \cdot \mathbf{W}_3) \cdot \mathbf{W}_4) \quad (8)$$

where $\mathbf{W}_3 \in \mathbb{R}^{(x+y) \times a_2}$, $\mathbf{W}_4 \in \mathbb{R}^{a_2 \times h_2}$. The matrix $\mathbf{A}_2 \in \mathbb{R}^{k \times h_2}$, where each of its column is normalized by the softmax operator, is a collection of h_2 different attention distributions on k documents. Using attention weights, we can generate attended representation of k evidence denoted as $\mathbf{d}^{rich} \in \mathbb{R}^{h_2 y}$ as shown in Eq. 9.

$$\mathbf{d}^{rich} = flatten(\mathbf{A}_2^T \cdot \mathbf{D}) \quad (9)$$

where *flatten(.)* function flattens $\mathbf{A}_2^T \cdot \mathbf{D}$ into a vector. We finally generate representation of a tuple $(c, s, \mathcal{D}, \mathcal{P})$ by concatenating vector $\mathbf{c}^{ext}$ (Eq. 5) and vector $\mathbf{d}^{rich}$ (Eq. 9), denoted as $[\mathbf{c}^{ext}; \mathbf{d}^{rich}]$.

To the best of our knowledge, our work is the first work utilizing multi-head attention mechanism integrated with speakers and publishers information to capture various semantic contributions of evidence toward fact-checking process.

4.4 Output Layer

In this layer, we input tuple representation $[\mathbf{c}^{ext}; \mathbf{d}^{rich}]$ into a multilayer perceptron (MLP) to compute probability $\hat{y}$ that the claim c is a true news as follows:

$$\hat{y} = \sigma\big(\mathbf{W}_6 \cdot \big(\mathbf{W}_5 \cdot [\mathbf{c}^{ext}; \mathbf{d}^{rich}] + \mathbf{b}_5\big) + \mathbf{b}_6\big) \quad (10)$$

where $\mathbf{W}_5, \mathbf{W}_6, \mathbf{b}_5, \mathbf{b}_6$ are weights and biases of the MLP, and $\sigma(.)$ is the sigmoid function. We optimize our model by minimizing the standard

Table 1: Statistics of our experimental datasets

	Snopes	PolitiFact
True claims	1,164	1,867
False claims	3,177	1,701
|Speakers|	N/A	664
|Documents|	29,242	29,556
|Publishers|	12,236	4,542

cross-entropy as shown on the top of Fig. 1.

$$\mathcal{L}_\theta(y, \hat{y}) = -\big(y \log \hat{y} + (1-y)\log(1-\hat{y})\big) \quad (11)$$

where $y \in \{0, 1\}$ is the ground truth label of a tuple $(c, s, \mathcal{D}, \mathcal{P})$. During training, we sample a mini batch of 32 tuples and compute average loss from the tuples.

5 Experiments

5.1 Datasets

We employed two public datasets released by (Popat et al., 2018). Each of these datasets is a collections of tuples $(c, s, \mathcal{D}, \mathcal{P}, y)$ where each textual claim c and its credible label y are collected from two major fact-checking websites `snopes.com` and `politifact.com`. The articles pertinent to the claim c are retrieved by using search engines. Each Snopes claim was labeled as *true* or *false* while in Politifact, there were originally six labels: *true*, *mostly true*, *half true*, *false*, *mostly false*, *pants on fire*. Following (Popat et al., 2018), we merge *true*, *mostly true* and *half true* into *true claims* and the rest are into *false claims*. Details of our datasets are presented in Table 1. Note that Snopes does not have speakers' information.

5.2 Baselines

We compare our MAC model with seven state-of-the-art baselines divided into two groups. The first group of the baselines only used textual content of claims, and the second group of the baselines utilized relevant articles to fact-check textual claims. A related method (Mishra and Setty, 2019) used subject information of articles (e.g. politics, entertainment), which was not available in our datasets. We tried to compare with it but achieved poor results perhaps due to missing information. Therefore, we do not report its result in this paper. Details of the baselines are shown as follows:

Using only claims' text:

- **BERT** (Devlin et al., 2018) is a pre-trained language model achieving state-of-the-art results on many NLP tasks. The representation of [CLS] token is inputted to a trainable linear layer to classify claims.
- **LSTM-Last** is a model proposed in (Rashkin et al., 2017). **LSTM-Last** takes the last hidden state of the LSTM as representations of claims. These representations will be inputted to a linear layer for classification.
- **LSTM-Avg** is another model proposed in (Rashkin et al., 2017) which used an average pooling layer on top of hidden states to derive representations of claims.
- **CNN** (Wang, 2017) is a state-of-the-art model which applied 1D-convolutional neural network on word vectors of claims.

Using both claims' text and articles' text:

- **DeClare** (Popat et al., 2018) computes credibility score of each pair of a claim c and a document d_i. The overall credible rating is averaged from all k relevant articles.
- **HAN** (Ma et al., 2019) is a hierarchical attention network based on representations of relevant documents. It uses attention mechanisms to determine which document is more important without considering which word in a document should be focused on.
- **NSMN** (Nie et al., 2019) is a state-of-the-art model designed to determine stance of a document d_i with respect to claim c. We apply NSMN on our dataset by predicting score of each pair (c, d_i) and computing average score based on documents in $\mathcal{D}$ same as DeClare.

Note that, we also applied BERT, LSTM-Last, LSTM-Avg and CNN by using both claims' text and articles' text. For each of these baselines, we concatenated a claim's text and a document's text, and input the concatenated content into the baseline to compute likelihood that the claim is fake news. We computed average probability based on all documents of the claim and used it as final prediction. However, we did not observe considerable improvements of these baselines. In addition to deep-learning-based baselines, we compared our MAC with other feature-based techniques (e.g. SVM). As expected, these traditional techniques had inferior performance compared with neural models. Therefore, we only report the seven baselines' performance.

Table 2: Performance of MAC and baselines on Snopes dataset. MAC outperforms baselines significantly with p-value<0.05 by one-sided paired Wilcoxon test.

Method Types	Methods				True News as Positive			Fake News as Positive		
		AUC	F1 Macro	F1 Micro	F1	Precision	Recall	F1	Precision	Recall
Using only claims' text	BERT	0.60852	0.56096	0.69806	0.31574	0.40318	0.26050	0.80618	0.76011	0.85839
	LSTM-Avg	0.69124	0.62100	0.71877	0.42953	0.48415	0.39692	0.81246	0.79139	0.83671
	LSTM-Last	0.70142	0.63122	0.72415	0.44650	0.48935	0.41412	0.81594	0.79594	0.83776
	TextCNN	0.70537	0.63081	0.72005	0.45001	0.48164	0.43035	0.81160	0.79882	0.82622
Using both claims' text & articles' text	HAN	0.70365	0.62510	0.72800	0.42884	0.49192	0.38161	0.82136	0.79058	0.85490
	NSMN	0.77270	0.68006	0.76127	0.51954	0.57558	0.48182	0.84058	0.82011	0.86364
	DeClare	0.81036	0.72445	0.78813	0.59250	0.61235	0.58096	0.85640	0.85023	0.86399
Ours	MAC	**0.88715**	**0.78660**	**0.83316**	**0.68738**	**0.69975**	**0.68601**	**0.88581**	**0.88617**	**0.88706**
Imprv. over the best baseline		9.47%	8.58%	5.71%	16.01%	14.27%	18.08%	3.43%	4.23%	2.67%

5.3 Experimental Settings

For each dataset, we randomly select 10% number of claims from each class to form a validation set, which is used for tuning hyper-parameters. We report 5-fold stratified cross validation results on the remaining 90% of the data. We train our model and baselines on 4-folds and test them on the remaining fold. We use AUC, macro/micro F1, class-specific F1, Precision and Recall as evaluation metrics. To mitigate overfitting and reduce training time, we early stop training process on the validation set when F1 macro on the validation data continuously decreases in 10 epochs. When we get the same F1 macro between consecutive epochs, we rely on AUC for early stopping.

For fair comparisons, we use Adam optimizer (Kingma and Ba, 2014) with learning rate 0.001 and regularize parameters of all methods with ℓ_2 norm and weight decay $\lambda = 0.001$. As the maximum lengths of claims and articles in words are 30 and 100 respectively for both datasets, we set $n = 30$ and $m = 100$. For HAN and our model, we set $k = 30$ since the number of articles for each claim is at most 30 in both datasets. Batch size is set to 32 and we trained all models until convergence. We tune all models including ours with hidden size H chosen from $\{64, 128, 300\}$, pre-trained word-embeddings are from Glove (Pennington et al., 2014) with $D = 300$. Both D_1 and D_2 are tuned from $\{128, 256\}$. The number of attention heads h_1 and h_2 is chosen from $\{1, 2, 3, 4, 5\}$, a_1 and a_2 are equal to $2 \times H$. In addition to Glove, we also utilized contextual embeddings from pre-trained language models such as ELMo and BERT but achieved comparable performances. We implemented all methods in PyTorch 0.4.1 and run experiments on an NVIDIA GTX 1080.

5.4 Performance of MAC and baselines

We show experimental results of our model and baselines in Tables 2 and 3. In Table 2, MAC outperforms all baselines with significance level $p < 0.05$ by using one-sided paired Wilcoxon test on Snopes dataset. MAC achieves the best result when $h_1 = 5, h_2 = 2, H = 300$ and $D_1 = D_2 = 128$. In Table 3, MAC also significantly outperforms all baselines with $p < 0.05$ according to one-sided paired Wilcoxon test on PolitiFact dataset. The hyperparameters we selected for MAC are $h_1 = 3, h_2 = 1, H = 300$ and $D_1 = D_2 = 128$.

For baselines, BERT is used as a static encoder. We tried to fine tune it but even achieve worse results. This might be because we do not have sufficient data to tune it. For both HAN and DeClare, since both papers do not release their source code, we tried our best to reproduce results from these two models. HAN model derived representation of each document by using the last hidden state of a GRU (Chung et al., 2014) without any attention mechanism on words to downgrade unimportant words (e.g. stop words), leading to poor representations of documents. Therefore, document-level attention mechanism in HAN model did not perform well. Similar patterns can be observed in two baselines LSTM-Avg and LSTM-Last. DeClare performed best among baselines, indicating the importance of applying word-level attention on words to reduce impact of less informative words.

We can see that our MAC outperforms all baselines in all metrics. When viewing *true news* as *positive class*, our MAC has an average increase of 16.0% and 7.1% over the best baselines on Snopes and PolitiFact respectively. We also have an increase of 4.7% improvements over baselines with a maximum improvements of 10.1% in PolitiFact

Table 3: Performance of MAC and baselines on PolitiFact dataset. MAC outperforms baselines with statistical significance level p-value<0.05 by one-sided paired Wilcoxon test.

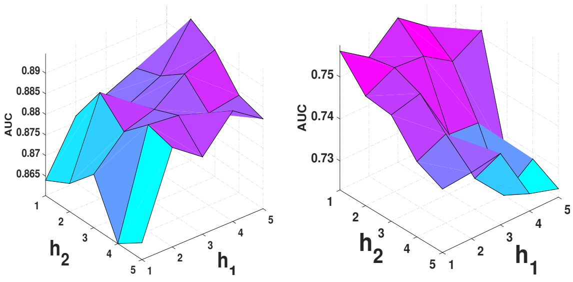
Method Types	Methods				True News as Positive			Fake News as Positive		
		AUC	F1 Macro	F1 Micro	F1	Precision	Recall	F1	Precision	Recall
Using only claims' text	BERT	0.58822	0.56021	0.56446	0.56364	0.59206	0.54968	0.55678	0.54354	0.58069
	LSTM-Avg	0.65465	0.60564	0.60866	0.61821	0.63192	0.61267	0.59307	0.59046	0.60425
	LSTM-Last	0.64289	0.60196	0.60493	0.61703	0.62634	0.61456	0.58690	0.58763	0.59434
	TextCNN	0.65152	0.60380	0.60740	0.61521	0.63010	0.61030	0.59238	0.59049	0.60421
Using both claims' text & articles' text	HAN	0.63201	0.58655	0.59121	0.59193	0.61502	0.58290	0.58117	0.57573	0.60034
	NSMN	0.64237	0.60211	0.60431	0.61123	0.63051	0.59912	0.59299	0.58213	0.60999
	DeClare	0.70642	0.65213	0.65350	0.67230	0.66548	0.67997	0.63195	0.64053	0.62444
Ours	MAC	**0.75756**	**0.68642**	**0.69116**	**0.71786**	**0.68856**	**0.75493**	**0.65498**	**0.70546**	**0.62576**
Imprv. over the best baseline		7.24%	5.26%	5.76%	6.78%	3.47%	11.02%	3.64%	10.14%	0.21%

Table 4: Impact of word attention and evidence attention on our MAC in two datasets

Methods	Snopes		PolitiFact	
	AUC	F1 Macro	AUC	F1 Macro
Only Word Att	0.87278	0.77831	0.74483	0.67818
Only Evidence Att	0.82531	0.72885	0.71790	0.65187
Word & Doc Att	**0.88715**	**0.78660**	**0.75756**	**0.68642**

Table 5: Impact of speakers and publishers on performance of MAC in two datasets

Methods	Snopes		PolitiFact	
	AUC	F1 Macro	AUC	F1 Macro
Text Only	0.88186	0.77146	0.72401	0.66844
Text + Publishers	**0.88715**	**0.78660**	0.72645	0.66984
Text + Speakers			0.75202	0.68483
Text + Pubs + Spkrs			**0.75756**	**0.68642**



(a) Snopes (b) PolitiFact

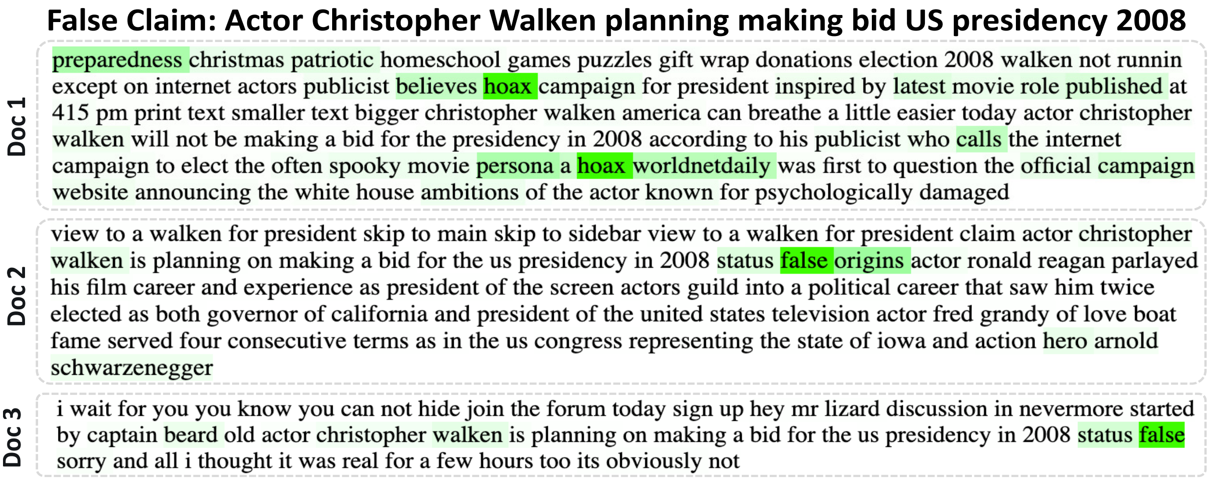
Figure 2: Sensitivity of MAC with respect to number of heads in word-level attention h_1 and the number of heads in document-level attention h_2

when considering *fake news* as *negative class*. In terms of AUC, average improvements of MAC over the baselines are 7.9% and 6.1% on Snopes and PolitiFact respectively. Improvements of MAC over baselines can be explained by our multi-head attention mechanism shown in Eq. 3 and Eq. 8. After attending to words in documents, we can generate better representations of documents/evidence, leading to more effective document-level attention compared with HAN model.

5.5 Ablation Studies

Impact of Word Attention and Evidence Attention. We study the impact of attention layers on performance of MAC by (1) using only word attention and replacing evidence attention with an average pooling layer on top of documents' representations and (2) using only evidence attention and replacing word attention with an average pooling layer on top of words' representations. As we can see in Table 4, using only word attention performs much better than using only evidence attention. This is because without downgrading less informative words in evidence, irrelevant information can be captured, leading to low quality representations of evidence. This experiment aligns with our observation that HAN model, which used only evidence attention, did not perform well. When combining both attention mechanisms hierarchically, we consistently achieve best results on two datasets in Table 4. In particular, the model *Word & Doc Att* outperformed both *Only Evidence Att* and *Only Evidence Att* significantly with p-value < 0.05. This result indicates that it is crucial to combine word-level attention and document-level attention to improve the performance of evidence-aware fake news detection task.

Impact of Speakers and Publishers on MAC. To study how speakers and publishers impact performance of MAC, we experiment four models: (1) using text only (Text Only), (2) using text and publishers (Text + Publishers), (3) using text and speakers (Text + Speakers) and (4) using text, publishers and speakers (Text + Pubs + Spkrs). In Table 5, Text + Publishers has better performance then using only text in both datasets. In PolitiFact, Text + Speakers achieves 2∼3% improvements over Text + Publishers, indicating that speakers who made claims are



False Claim: Actor Christopher Walken planning making bid US presidency 2008

Doc 1

preparedness christmas patriotic homeschool games puzzles gift wrap donations election 2008 walken not runnin except on internet actors publicist believes hoax campaign for president inspired by latest movie role published at 415 pm print text smaller text bigger christopher walken america can breathe a little easier today actor christopher walken will not be making a bid for the presidency in 2008 according to his publicist who calls the internet campaign to elect the often spooky movie persona a hoax worldnetdaily was first to question the official campaign website announcing the white house ambitions of the actor known for psychologically damaged

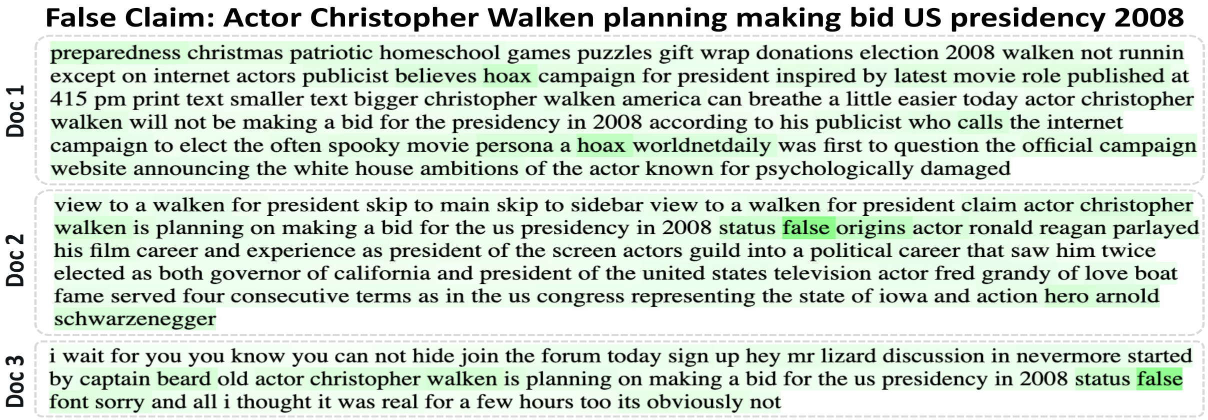
Doc 2

view to a walken for president skip to main skip to sidebar view to a walken for president claim actor christopher walken is planning on making a bid for the us presidency in 2008 status false origins actor ronald reagan parlayed his film career and experience as president of the screen actors guild into a political career that saw him twice elected as both governor of california and president of the united states television actor fred grandy of love boat fame served four consecutive terms as in the us congress representing the state of iowa and action hero arnold schwarzenegger

Doc 3

i wait for you you know you can not hide join the forum today sign up hey mr lizard discussion in nevermore started by captain beard old actor christopher walken is planning on making a bid for the us presidency in 2008 status false sorry and all i thought it was real for a few hours too its obviously not

Figure 3: Visualization of attention weights of *the first attention head* on three documents relevant to a false claim in word-level attention layer



False Claim: Actor Christopher Walken planning making bid US presidency 2008

Doc 1

preparedness christmas patriotic homeschool games puzzles gift wrap donations election 2008 walken not runnin except on internet actors publicist believes hoax campaign for president inspired by latest movie role published at 415 pm print text smaller text bigger christopher walken america can breathe a little easier today actor christopher walken will not be making a bid for the presidency in 2008 according to his publicist who calls the internet campaign to elect the often spooky movie persona a hoax worldnetdaily was first to question the official campaign website announcing the white house ambitions of the actor known for psychologically damaged

Doc 2

view to a walken for president skip to main skip to sidebar view to a walken for president claim actor christopher walken is planning on making a bid for the us presidency in 2008 status false origins actor ronald reagan parlayed his film career and experience as president of the screen actors guild into a political career that saw him twice elected as both governor of california and president of the united states television actor fred grandy of love boat fame served four consecutive terms as in the us congress representing the state of iowa and action hero arnold schwarzenegger

Doc 3

i wait for you you know you can not hide join the forum today sign up hey mr lizard discussion in nevermore started by captain beard old actor christopher walken is planning on making a bid for the us presidency in 2008 status false font sorry and all i thought it was real for a few hours too its obviously not

Figure 4: Visualization of attention weights of *the second attention head* on three documents relevant to a false claim in word-level attention layer

crucial to determine verdict of the claims. Finally, using all information (Text + Pubs + Spkrs) helps us achieve the best result in PolitiFact. In Snopes, we omit results of Text + Speakers and Text + Pubs + Spkrs because the dataset does not contain speakers' information. In particular, model *Text + Pubs + Spkrs* outperformed methods *Text Only* and *Text + Publishers* significantly (p-value< 0.05). Based on these results, we conclude that integrating information of speakers and publishers is useful for detecting misinformation.

5.6 Impact of the Number of Attention Heads

In this section, we examine sensitivity of MAC with respect to the number of heads h_1 in word attention layer and the number of heads h_2 in document attention layer. We vary h_1 and h_2 in $\{1, 2, 3, 4, 5\}$. Since AUC is less sensitive to any threshold, we report AUC of MAC on two datasets in Fig. 2(a) and 2(b). A common pattern we can observe in the two figures is that performance of MAC tends to be better when we increase the number heads h_1 in word attention layer while performance of MAC tends to decrease when increasing h_2. This phenomenon indicates that word attention is more important than evidence attention. In Snopes, MAC has the best AUC when $h_1 = 5, h_2 = 2$. In PolitiFact, MAC reaches the peak when $h_1 = 3, h_2 = 1$.

5.7 Case Study

To understand how multi-head attention mechanism works, from the testing set, we visualize attention weights on three documents of a false claim *Actor Christopher Walken planning making bid US presidency 2008*. Note, our MAC correctly classifies the claim as fake news. In Fig. 3 and Fig. 4, we show the claim and visualization of two different heads in word attention layer. Note that Popat et al.

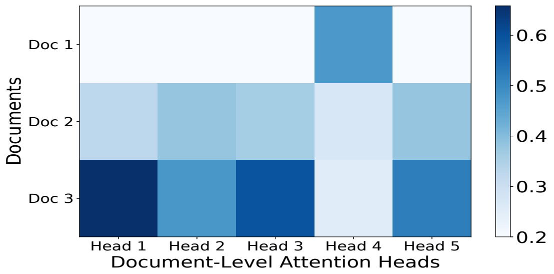


Figure 5: Visualization of five attention heads in document-level attention layer for three documents

(2018), who released the datasets, already lowercased and removed punctuations. To conduct fair comparison, we directly used the datasets without any additional preprocessing. In Fig. 3, attention weights are sparse, indicating that the first attention head focuses on the most important words which determine credibility of the claim (e.g. *hoax*, *false*). Differently, in Fig. 4, the second attention head has more diffused attention weights to capture more useful phrases from documents (e.g. *walken not running*, *its obviously not*). Moving on to attention heads in evidence attention layer in Fig. 5, we show a heat map where the x-axis is the five heads extracted from evidence attention layer and the y-axis is three documents relevant to the same claim in Fig. 3 and 4. As we can see in Fig. 5, *Head 1*, *Head 3* and *Head 5* emphasize on *Doc 3* which contains refuting phrases (e.g. *its obviously not*), while *Head 4* focuses on *Doc 1* which has negating information such as *walken not running*. Both *Doc 1* and *Doc 3* have crucial signals to fact-check the claim. From these analyses, we conclude that heads in word attention layer capture different semantic contributions of words and different heads in document attention layer captures important documents.

6 Conclusions

In this paper, we propose a novel evidence-aware model to fact-check textual claims. Our MAC is designed by hierarchically stacking two attention layers. The first one is a word attention layer and the second one is a document attention layer. In both layers, we propose multi-head attention mechanisms to capture different semantic contributions of words and documents. Our MAC outperforms the baselines significantly with an average increase of 6% to 9% over the best results from baselines with a maximum improvements of 18%. We conduct ablation studies to understand the performance of MAC and provide a case study to show the effectiveness of the attention mechanisms. In future work, we will further examine other data types such as images to improve the performance of our model.

Acknowledgment

This work was supported in part by NSF grant CNS-1755536, AWS Cloud Credits for Research, and Google Cloud. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.

References

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36.

Aparna Alluri. 2019. Whatsapp: The 'black hole' of fake news in india's election. `https://www.bbc.com/news/world-asia-india-47797151`.

Ashoka. 2020. Misinformation spreads faster than coronavirus: How a social organization in turkey is fighting fake news. `https://bit.ly/36qqmmH`.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Neural Information Processing Systems*.

CNN. 2020. How facebook is combating spread of covid-19 misinformation. `https://cnn.it/3gjtBkg`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional lstm networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*, pages 799–804. Springer.

Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*, pages 729–736.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Shimon Kogan, Tobias J Moskowitz, and Marina Niessner. 2019. Fake news: Evidence from financial markets. *Available at SSRN 3237763*.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *The 5th International Conference on Learning Representations*.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM.

Yang Liu and Yi-Fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Empirical Methods in Natural Language Processing*.

Jing Ma, Wei Gao, Shafiq Joty, and Kam-Fai Wong. 2019. Sentence-level evidence embedding for claim verification with hierarchical attention networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2561–2571.

Rahul Mishra and Vinay Setty. 2019. Sadhan: Hierarchical attention networks to learn latent aspect embeddings for fake news detection. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 197–204.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6859–6866.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. Declare: Debunking fake news and false claims using evidence-aware deep learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 22–32.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 395–405.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. 2020. Hierarchical propagation networks for fake news detection: Investigation and exploitation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 626–637.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. In *European Conference on Information Retrieval*, pages 359–366. Springer.

Mark Stencel. 2019. Number of fact-checking outlets surges to 188 in more than 60 countries. `https://bit.ly/36y3S3l`.

James Thorne and Andreas Vlachos. 2017. An extensible framework for verification of numerical claims. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 37–40.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Andreas Vlachos and Sebastian Riedel. 2015. Identification and verification of simple claims about statistical properties. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2596–2601. Association for Computational Linguistics.

Nguyen Vo and Kyumin Lee. 2018. The rise of guardians: Fact-checking url recommendation to combat fake news. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 275–284.

Nguyen Vo and Kyumin Lee. 2019. Learning from fact-checkers: Analysis and generation of fact-checking language. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–344.

Nguyen Vo and Kyumin Lee. 2020a. Standing on the shoulders of guardians: Novel methodologies to combat fake news. In *Disinformation, Misinformation, and Fake News in Social Media*, pages 183–210. Springer.

Nguyen Vo and Kyumin Lee. 2020b. Where are the facts? searching for fact-checked information to alleviate the spread of fake news. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7717–7731.

William Yang Wang. 2017. “liar, liar pants on fire”: A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426.

Xuezhi Wang, Cong Yu, Simon Baumgartner, and Flip Korn. 2018. Relevant document discovery for fact-checking articles. In *Companion Proceedings of the The Web Conference 2018*, pages 525–533.

Liang Wu and Huan Liu. 2018. Tracing fake-news footprints: Characterizing social media messages by how they propagate. In *Proceedings of the eleventh ACM international conference on Web Search and Data Mining*, pages 637–645.

Lianwei Wu, Yuan Rao, Xiong Yang, Wanzhen Wang, and Ambreen Nazir. 2020. Evidence-aware hierarchical interactive attention networks for explainable claim verification. In *International Joint Conferences on Artificial Intelligence*.

Di You, Nguyen Vo, Kyumin Lee, and Qiang Liu. 2019. Attributed multi-relational attention network for fact-checking url recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1471–1480.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, pages 9051–9062.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th international conference on world wide web*, pages 1395–1405.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. Gear: Graph-based evidence aggregating and reasoning for fact verification. page Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

Identifying Named Entities as they are Typed

Ravneet Singh Arora **Chen-Tse Tsai** **Daniel Preoţiuc-Pietro**
Bloomberg
`{rarora62,ctsai54,dpreotiucpie}@bloomberg.net`

Abstract

Identifying named entities in written text is an essential component of the text processing pipeline used in applications such as text editors to gain a better understanding of the semantics of the text. However, the typical experimental setup for evaluating Named Entity Recognition (NER) systems is not directly applicable to systems that process text in real time as the text is being typed. Evaluation is performed on a sentence level assuming the end-user is willing to wait until the entire sentence is typed for entities to be identified and further linked to identifiers or co-referenced. We introduce a novel experimental setup for NER systems for applications where decisions about named entity boundaries need to be performed in an online fashion. We study how state-of-the-art methods perform under this setup in multiple languages and propose adaptations to these models to suit this new experimental setup. Experimental results show that the best systems that are evaluated on each token after its typed, reach performance within 1–5 F_1 points of systems that are evaluated at the end of the sentence. These show that entity recognition can be performed in this setup and open up the development of other NLP tools in a similar setup.

1 Introduction

Automatically identifying named entities such as organizations, people and locations is a key component in processing written text as it aids with understanding the semantics of the text. Named entity recognition is used as a pre-processing step to subsequent tasks such as linking named entities to concepts in a knowledge graph, identifying the salience of an entity to the text, identifying coreferential mentions, computing sentiment towards an entity, in question answering or for extracting relations.

Figure 1: An example of the proposed task and evaluation setup. After the word ‘Foreign’ is typed, the model immediately predicts an NER label for this word, only using left context (‘A spokesman for’) and the word itself. The prediction is then compared against the gold label to compute token-level F_1 score. This token’s prediction will not be changed, even if the model’s internal prediction for it can be revised later as more tokens are typed.

Identifying named entities as they are typed benefits any system that processes text on the fly. Examples of such applications include: a) News editors – where named entities can be highlighted, suggested (auto-completion), co-referenced or linked as the editor is typing; b) auto-correct – where named entities that are just typed are less likely to need correction as they may come from a different language or be out-of-vocabulary (OOV); c) simultaneous machine translation – where translation of OOV named entities requires different approaches; d) live speech-to-text (e.g., TV shows) – where named entities are more likely to be OOV, hence the transcription should focus more on the phonetic transcription rather than on n-gram language modelling.

This paper introduces a novel experimental setup of Named Entity Recognition systems illustrated in Figure 1. In this setup, inference about the span and type of named entities is performed for each token, immediately after it was typed. The sentence level tag sequence is composed through appending all individual token predictions as they were made. The current named entity recognition systems that are trained and evaluated to predict full sentences are likely to under-perform in this experimental setup as they: expect that right context is available,

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 976–988
April 19 - 23, 2021.

are faced with unseen types of inputs in the form of truncated sentences and can not reconcile the final sentence-level tag sequence across the entire sentence as the result may not be a valid sequence.

The goal of this study is to present a comprehensive analysis of the task of NER in the *as-you-type* scenario, with the following contributions:

a) A novel experimental setup for conducting named entity recognition experiments, denoted as the *as-you-type* scenario;
b) Experiments with state-of-the-art sentence-level approaches to named entity recognition in the *as-you-type* setup across three languages, which indicate a 1–5 F_1 points decrease compared to sentence-level inference;
c) Tailored methods for as-you-type entity recognition models which reduce the gap to entire sentence-level inference by 9–23% compared to regular approaches;
d) An extensive analysis of existing data sets in the context of this task and model error analysis, which highlight future modelling opportunities.

2 Related Work

Named Entity Recognition is most commonly treated as a sequence labelling problem, where a prediction of whether a token is an entity and its type is done jointly for all tokens in the sentence. Over the past recent years, the dominant approach is based on recurrent neural networks, such as LSTMs (Hochreiter and Schmidhuber, 1997). These architectures use a stacked bi-directional LSTM units to transform the word-level features into distributions over named entity tags (Huang et al., 2015). Usually, an additional Conditional Random Field (CRF) (Lafferty et al., 2001) is used on the BiLSTM output in order to take into better model neighbouring tags. The tokens inputs are represented using one or a concatenation of pre-trained static word embeddings such as GloVe (Ma and Hovy, 2016), contextual word embeddings (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2019), pooled contextual word embeddings (Akbik et al., 2019b) or character embeddings trained using BiLSTMs (Lample et al., 2016) or CNNs (Ma and Hovy, 2016; Chiu and Nichols, 2016).

In addition to research on improving the performance of the NER model, other experimental setups have been proposed for this task. These include domain adaptation, where a model trained on data from a source domain is used to tag data from a different target domain (Guo et al., 2009; Greenberg et al., 2018; Wang et al., 2020), temporal drift, where a model is tested on data from future time intervals (Derczynski et al., 2016; Rijhwani and Preotiuc-Pietro, 2020), cross-lingual modelling where models trained in one language are adapted to other languages (Tsai et al., 2016; Ni et al., 2017; Xie et al., 2018), identifying nested entities (Alex et al., 2007; Lu and Roth, 2015) or high-precision NER models (Arora et al., 2019).

However, all these experimental setups assume that training is done over full length sentences. Perhaps the most related experimental setup to the one we propose for the task of entity recognition is the task of simultaneous machine translation. In this setup, the task is to generate an automatic translation in a target language as the text is being processed in the source language. The goal of the task is to produce a translation that is as accurate as possible while limiting the delay as compared to the input. Initial approaches involved identifying translatable segments and translating these independently (Fügen et al., 2007; Bangalore et al., 2012; Fujita et al., 2013) or by learning where to segment in order to optimize the system's performance (Oda et al., 2014). More recent approaches involve learning training an agent, usually using reinforcement learning, that makes a set of decisions of whether to should wait for another word from the input or write a token to the output (Gu et al., 2017). Other operations are shown to help, including predicting the verb (Grissom II et al., 2014) or the next word (Alinejad et al., 2018), better decoding with partial information (Cho and Esipova, 2016), and connecting the machine translation system to the agent's decisions (Gu et al., 2017).

Our experimental setup is different as we do not want to wait for another input token before we make a prediction about the named entity. We analyze the impact a delay has, albeit our experimental setup does not aim to combine quality and delay. The challenges are related, as the input may contain important cues for the translation or named entity decision after the current token or towards the end of the sentence, such as the verb in verb-final (SOV) languages such as German (Grissom II et al., 2014). The proposed as-you-type NER model can be useful to improve simultaneous machine translation.

3 Experimental Setup

We propose a new experimental setup for the standard task of Named Entity Recognition that would best suit real-time applications that need to process text in an online fashion.

In the regular NER experimental setup, a model is presented with a sequence of inputs $X = \{x_1, x_2, ..., x_n\}$ and it outputs a sequence of labels $Y = \{y_1, y_2, ..., y_n\}$ where $y_i \in K = \{O\} + E \times T$, where E are the set of entity types and T is the entity tag representation. Throughout the rest of the paper, we use the BIO tagging scheme ($T = \{\text{B}, \text{I}\}$), as this is arguably the most popular and differences in results between this tagging scheme and others, such as the BILOU scheme, are very small in practice (Ratinov and Roth, 2009). The types of entities we consider are $E = \{\text{ORG}, \text{PER}, \text{LOC}, \text{MISC}\}$.

The *as-you-type* named entity recognition setup assumes that the editor writing the text $X = \{x_1, x_2, ..., x_n\}$ needs each label prediction y_i right after the corresponding token x_i was typed. In this case, the information available for predicting y_i is only the sub-sequence $X_{1,i} = \{x_1, x_2, ..., x_i\}$. The sequence $Y = \{y_1, y_2, ..., y_n\}$ is obtained by concatenating the individual y_i predictions made for each token. Token-level micro F_1 score is used as the metric in our experiments. The evaluation process is illustrated in Figure 1.

This setup presents the model with several challenges. First, the model has no access to right context when making the prediction for each tag. However, this information is available in training. Secondly, the output sequence may contain invalid sequences of tags. For example, in the output sequence, B-ORG could be followed by I-LOC if the model decided to revise its predictions based on new information, but the evaluation setup prevents the model from revising the previous wrongly predicted tag (i.e. B-ORG). Lastly, sequences and sentences of the same length are likely to be qualitatively different and the model might need to adapt in training in order to account for these differences.

We note that this experimental setup can further be extended to account for delays in prediction, to trade-off between delays and quality or to predict entities before they are typed.

4 Data

We test our methods on three different data sets covering three different languages. We use the data sets released as part of CoNLL shared tasks in 2002 for Spanish (Tjong Kim Sang, 2002)[1] and in 2003 for English and German (Tjong Kim Sang and De Meulder, 2003).[2] The data sets contain four types of named entities: persons, locations, organizations and names of miscellaneous entities that do not belong to the previous three types. We use the standard train, dev and test splits defined for these data sets.

We chose these data sets as they are arguably the most popular data sets for performing named entity recognition and are regularly used as benchmarks for this task. We use the data sets in different languages in order to compare the impact of the language on the experimental results, identify if the commonalities and peculiarities for performing *as-you-type* entity recognition in different languages and draw more robust conclusions regarding our task setup.

4.1 Data Analysis

We perform a quantitative analysis of the data sets in order to develop some intuitions about the data in the context of the *as-you-type* experimental setup.

Sentence Length and Tag Distribution First, we study the distribution of sentence lengths. Figure 2 shows that for English, most sentences are very short (under 10 tokens) and the most frequent sentence length is two. These are expected to pose problems in the as-you-type scenario, as the context is limited. German sentences are slightly longer, while the Spanish sentences are longest, except for a spike of sentences of length one.

Figure 3 shows the distribution of entity types in sentences of varying lengths for English. For clarity, we remove MISC tags from this plot as these are infrequent. We observe there are major differences in tag distributions, especially shorter sentences (<5 tokens) containing both more entity tags as well as different tag distributions. For example, almost 30% of locations (B-LOC or I-LOC) are in sentences of length two, which are the most frequent in the English data, while in longer sentences, these are around 5%. Organizations are most frequent in sentences of length between 4 and 7 tokens, while persons are most frequent in sentences longer than 7 tokens.

Token Position and Tag Distribution To further investigate the positional bias of different tags, Fig-

---

[1] https://www.clips.uantwerpen.be/conll2002/ner/

[2] https://www.clips.uantwerpen.be/conll2003/ner/

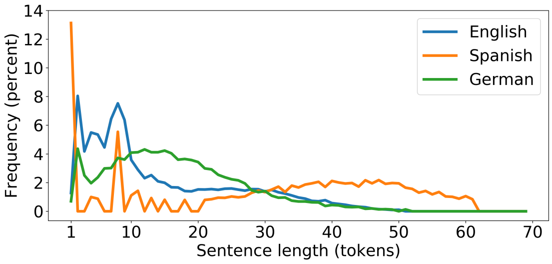


Figure 2: Distribution of sentence lengths.

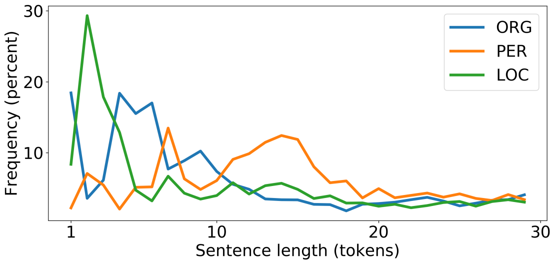


Figure 3: Distribution of entity types in terms of sentence length in the English CoNLL data set.

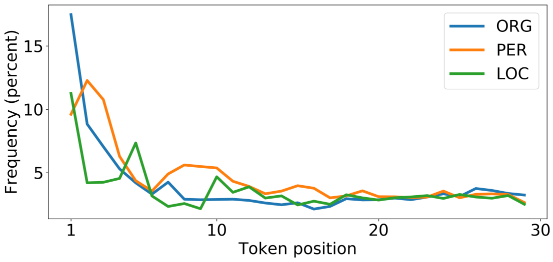


Figure 4: Distribution of entity types at each token position in the English CoNLL data set. B and I tags are merged for the same entity types and removed MISC tags as infrequent for clarity. O tag frequency can be inferred from the rest.

ure 4 shows the distribution of tags in the k-th token of the sentence. We observe that the first tokens of a sentence are much more likely to contain entities. The first position is most likely to be an ORG, with PER being the most frequent in the second to fourth positions, followed by LOC being the most prevalent for the next position, with PER again most frequent in further positions. These observations are likely to complicate the as-you-type inference for named entities, as a higher proportion of tokens will have to be inferred with no or little right context. Comparing Figures 3 and 4 shows that the model will be faced with different tag distributions when inferring the tag for the k-th token in a truncated sentence then to what it has observed in sentences of length k, which provides an intuition for our modelling strategies.

5 Methods

This section describes the methods used to perform named entity recognition in the as-you-type scenario. We use a base neural architecture that achieves state-of-the-art performance on the standard sentence-level NER task. We study its performance and observe the impact of different variants of the architecture in the *as-you-type* scenario. Following, we propose changes to the model to adapt to the as-you-type setup. We use the Flair package to conduct our experiments (Akbik et al., 2019a).[3] Implementation details and hyperparameter choices for all models are listed in the Appendix.

5.1 Base Architecture

We adopt the BiLSTM-CRF model proposed in (Huang et al., 2015) with the addition of character representation (Lample et al., 2016; Ma and Hovy, 2016). In this architecture, the word representations are fed into a bi-directional LSTM, and then the concatenated forward and backward vectors are passed through one layer of feed-forward neural network to produce a $|K|$ dimensional output for each word, where each value represents a score associated with each label. Finally, a Conditional Random Field (CRF) layer is applied to make a global decision for the entire sentence. This has the role of reconciling the independent predictions and modeling the constraints in the output space (e.g. I-PER can not follow B-ORG).

5.2 Architecture Variants

We start with studying different variants of the base neural architecture for the as-you-type scenario. The key challenge in the as-you-type setting is that the model is not presented with the future or right context (words after the current word) at test time. A natural idea is to remove information from this context during training as well. The variants we consider are based on changing the following three modeling components

Embeddings We first study the impact of different ways in which input tokens are represented. Pretrained word embeddings obtained state-of-the-art performance on the NER task when they were introduced (Lample et al., 2016). These representations are used to initialize the word embeddings, are then fine-tuned on the training data and are concatenated

---

[3] https://github.com/zalandoresearch/flair

with a character-level representation of the word obtained using BiLSTMs initialized with random character embeddings.

Contextual word embeddings extend this concept to obtain different word representations for the same token in based on its context. In the standard sentence-level evaluation, contextual word embeddings were shown to obtain 2–3 F_1 points improvement on the English CoNLL data set (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2019). Without right context, the quality of word representations could be more crucial than in the standard setting. In this study, we test the performance of using classic embeddings – GloVe embeddings for English (Pennington et al., 2014) and FastText embeddings (Bojanowski et al., 2017) for German and Spanish as well as the character based contextual Flair embeddings, which achieve state-of-the-art performance on the English and German CoNLL data sets (Akbik et al., 2018). We also experimented with contextual ELMO embeddings (Peters et al., 2018) which showed slightly lower performance when compared to the Flair embeddings and hence only Flair numbers are reported due to space limitations.

However, contextual embeddings are trained with right context available. We experiment with removing this dependency from the trained embeddings and observe if this improves the performance in the as-you-type setting, as the test scenario is more similar to the training one. We note that right context is never observed in inference beyond the current token such that there is no leakage of information.

BiLSTM Bidirectional LSTM stacks two recurrent neural networks: one starts from the beginning of the sentence, and another starts from the end of the sentence. This performs better than the unidirectional variant on sentence-level experiments and shows that both types of context (left and right) are important for identifying and typing entity mentions. In the as-you-type setting, we compare unidirectional LSTM modelling left context with the bidirectional LSTM model that models both types of contexts in training.

Conditional Random Field The CRF assigns labels for words in a sentence jointly, ensuring label assignments are coherent. When running inference in the as-you-type setting, the model often sees truncated sentences which, as shown in Section 4 may have different label distributions. This discrepancy between training and test sequences may degrade the usefulness of the CRF. We experiment to see if and how the CRF is useful in the as-you-type scenario.

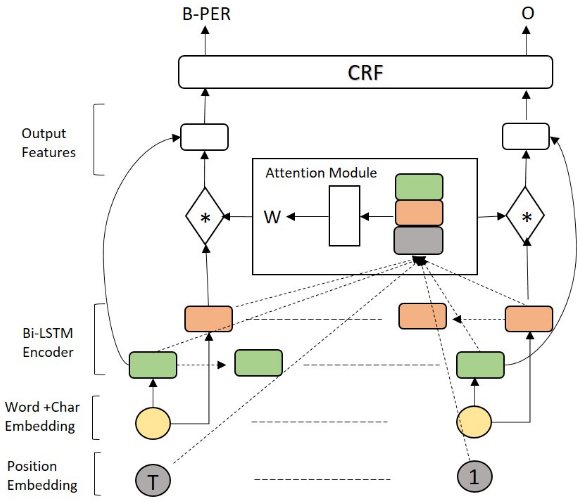


Figure 5: Architecture diagram highlighting the proposed feature weighting method described in Section 5.3.3

5.3 Model Adaptations for the As-you-type Setup

The as-you-type experimental setup presents the model with a new type of evaluation, which does not correspond to the one used in training. We thus propose the following approaches to bridge the gap between the setup and how the model is trained.

5.3.1 Weighted Loss

The model only observes the partial backward context for the tokens in the sequence during the as-you-type inference. In training, since the model has access to the entire sequence, it is likely that the model becomes too dependent on the presence and reliability of the backward features, especially for predicting the initial tokens.

In order to bias the model to be more robust to the absence or unreliable backward context, we design a new loss function that combines the original BiLSTM-CRF loss with the loss from the unidirectional LSTM features. From the latter loss, we also remove the influence of CRF as it also captures signal from the right context and, for contextual embeddings, remove the backward embeddings from the input to the LSTM. The resulting loss is:

$$L_{\text{weighted}} = L_{\text{BiLSTM-CRF}} + w * L_{\text{LSTM-NoCRF}}$$

where w is a weight treated as a hyper-parameter in our experiments.

5.3.2 Final Token Representation

The final token in a training sequence is treated in a special way in the base model. First, a stop tag is regularly used to capture the information associated with a the last token in the sequence. Secondly, the backward features for the final token in a sequence are not observed, as there is no right context, so these are initialized with a random vector. While both are useful in the regular setup as it captures a consistent pattern the final words follow, this does not hold for the as-you-type setup, where each token in a sequence will be the final token for its prediction, as partial sequences are observed.

We thus assume these choices add noise in our setup, thus we both remove the stop tag together with any transition scores to it during training and evaluation and remove the backward feature of the last token and initialize it with a zero vector.

5.3.3 Feature Weighting

The previous approach relied on the intuition of learning the trade-off between forward and backward features in order to better adapt to the inference setup. However, this trade-off is likely impacted by the position of the token in the sequence.

Thus, we explore a technique similar to (Moon et al., 2018) that allows the model to learn the importance of backward features based on the position of tokens. The is illustrated in Figure 5. We implement this using position based attention weights. We apply these weights before combining the forward and backward features (instead of concatenating) from the LSTMs as follows:

$$h_t = h_t^f + a_t * h_t^b$$

where t is the position of the token, h_t^f and h_t^b are forward and backward LSTM features at t, h_t is the new output feature at t and a_t is the attention weight for the backward features at position t. The attention weights a_t are computed as follows:

$$u_t = [h_t^f; h_t^b; p_t]; a_t = \sigma(W.u_t + b)$$

At every position t, the feature vector is calculated by concatenating the forward, backward and the positional embedding for that token. Attention weight a_t is calculated by applying attention weight matrix W followed by the sigma activation. We do not apply a_t for forward features since they are always complete and reliable even for partial sentences.

We follow the structure of positional embeddings introduced in (Vaswani et al., 2017) and defined as:

$$p_t^{2i} = sin(\frac{t}{10000^{2i/d}}); p_t^{2i+1} = cos(\frac{t}{10000^{2i/d}})$$

where d is the size of positional embedding, i is the dimension and t is the position. The values in a positional embedding are sin and cosine functions whose period is $10000^{2i/d} * 2\pi$. Positional sinusoidal embedding allows to encode longer sequences that are not present in training.

Since the right hand side context decreases as we move from left to right of a sequence in training, we would like our attention weights to consider how far a token lies from the final token in a sequence. To achieve this, we calculate position index of tokens from the end of the sentence which makes sure that a token lying at the final position always receives an index of 0, producing the same positional embedding and the input to attention weights does not fluctuate from one sequence to another.

5.3.4 Embedding Weighting

We perform a similar operation using attention at the embedding stage to trade-off between backward and forward contextual token representations. The input embeddings are calculated as follows:

$$x_t = e_t^w + e_t^f + a_t^e * e_t^b$$

$$v_t = [e_t^w; e_t^f; e_t^b; p_t]; a_t^e = \sigma(W_e.v_t + b_e)$$

where e^w are the classical word embeddings and e^f, e^b are Flair forward and backward embeddings. An architecture diagram is presented in the Appendix.

6 Results

We present the experimental results of the various NER models in the as-you-type setup, contrasting them with the regular sentence-level setup. All models are trained using the standard splits for the CoNLL data sets. The evaluation metric is token-level micro F_1, as this is reflective of our evaluation setup where each token is evaluated separately.

The top section of Table 1 shows the results of the different variants of the base architecture in Section 5.2. The overall performance drop in the as-you-type setup compared to the sentence-level setup ranges from 4.80 F1 for English to only 1.53 F1 for Spanish when comparing the best performing models. This is expected, as the as-you-type scenario is more challenging, especially for English where our data analysis from Section 4.1 showed that tokens are more prevalent in short sentences which are overall more frequent. For Spanish, where the performance difference is smallest, is where we have on average the longest sentences and in which left context alone is in most cases enough to make the correct inference.

			English		**German**		**Spanish**	
Embedding	**LSTM**	**CRF**	As-you-type	Sentence	As-you-type	Sentence	As-you-type	Sentence
Classic	⇆	✓	83.27	90.97	72.20	78.67	67.56	80.40
	→	✓	85.12	88.99	70.54	77.21	64.84	80.28
	⇆	✗	79.06	86.87	73.49	77.89	74.61	80.74
	→	✗	83.75	83.27	75.73	75.73	77.30	77.30
Flair (⇆)	⇆	✓	84.15	**92.75**	79.79	84.32	81.80	89.43
	→	✓	84.82	91.63	79.98	84.11	82.23	88.82
	⇆	✗	84.50	92.23	79.84	83.73	85.37	88.80
	→	✗	85.87	90.73	80.50	82.74	85.32	89.06
Flair (→)	⇆	✓	85.60	92.19	79.21	82.94	81.61	88.76
	→	✓	86.92	90.36	78.34	81.99	82.60	88.21
	⇆	✗	85.13	91.76	79.30	81.88	86.79	88.83
	→	✗	87.95	87.95	80.79	80.79	**87.90**	87.90
Adaptations for the as-you-type setup								
Flair (⇆)	⇆ Weighted Loss		87.77	92.46	80.39	83.71	84.13	89.48
	+ Final Token Rep		88.00	92.40	80.59	83.71	87.23	89.48
	+ Feature weighting		88.21	92.24	80.61	84.16	87.38	89.23
	+ Embedding weighting		**88.40**	92.29	**81.62**	**84.77**	87.72	**89.79**

Table 1: Evaluation results of LSTM-based NER models in the as-you-type and sentence-level evaluation setups as measured using token-level micro F_1. Arrows indicate if uni- (→) or bi-directional (⇆) training is used. Models with the best results across their setup are in **bold**. Best results within the class of methods are underlined. For classic word embeddings, we use GloVe for English, and FastText for German and Spanish. Results are averaged across three runs.

We note that in all three data sets, the best results in the as-you-type setting are obtained by matching the training setup to that of testing by only keeping a uni-directional LSTM that processes the text left to right and Flair embeddings only trained using left context. Flair embeddings trained only using left context are in all cases better than the bi-directional ones, which is natural as those embeddings would conflate information that is not available in inference. Uni-directional LSTMs perform overall better than bi-directional LSTMs by an average of a few percentage points as bi-directional LSTMs are likely learning information that will not be available in testing.

Adding the CRF hurts performance in all except one case when holding everything else constant, sometimes by wide margins (e.g. 5.3 F1 drop on Spanish with Flair forward embeddings and uni-directional LSTM). We attribute this to the mismatch between the structure of the sequences in the training data containing only full sentences, when compared to truncated sentences which can be observed by comparing Figures 3 to Figures 4.

The bottom section of Table 1 shows the results of our as-you-type adaptation strategies. All proposed methods are added on top of each other in order to study their individual contributions. For brevity, we only present the results of using Flair forward and backward embeddings as these performed best.

The changes to the last token representation and weighted loss improves on the regular bi-directional LSTM model by a substantial margin, adding between 0.45 for Spanish up to 4.25 F1 on English on the as-you type setup performance. We also notice that the sentence-level evaluation is near to the regular model performance (-0.39 for German to +0.05 for Spanish), showing that the weighted loss is able to achieve a good compromise between the representations.

Adding the feature weighting based on position marginally improves performance, between 0.02 on German to 0.21 on English. However, the weighting through attention is more effective at the embedding level improving on the previous model by between 0.19 F1 on English to 1.01 F1 on German. Overall, our as-you-type adaptation methods improve on the best variation of the base architecture on English (+0.45 F1) and German (+0.83 F1). The model is competitive on Spanish (-0.12 F1) to the Flair forward unidirectional LSTM with no CRF, albeit this is likely driven by the very long average sentence length in the Spanish data set (see Figure 2). Overall, the improvements represent between 9.3% - 23% error reduction when comparing to the best as-you-type and sentence level setups.

We highlight that an additional benefit of the proposed adaptation methods is that the model retains high performance on the sentence-level setup, in contrast with the Flair forward uni-directional LSTM, which performs between 1.89 (Spanish) and 4.34 (English) worse on the sentence-level.

Finally, the results of our proposed model are actually marginally better than the state-of-the-art approach of BiLSTM+CRF using Flair embeddings on German (+0.45 F1) and Spanish (+0.73 F1), albeit this was not the original goal of our additions. This highlights that the proposed modelling ideas are more generally beneficial as they push the model to learn more robust representations.

7 Error Analysis

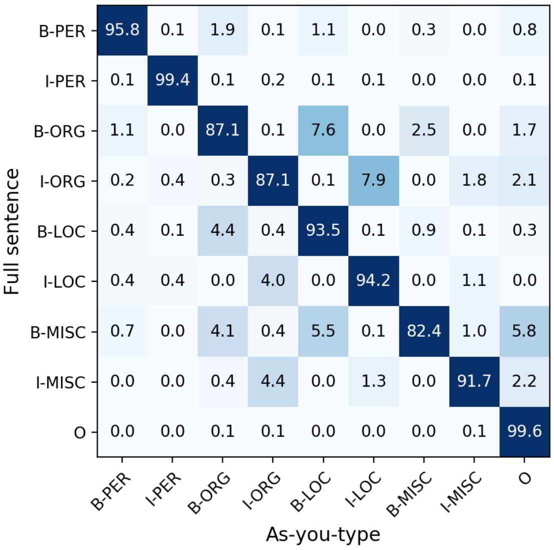
Finally, we perform error analysis to identify the limitations of our approaches.

7.1 Confusion Matrix

We first study prediction difference between as-you-type and sentence-level setup. Figure 6 shows the confusion matrix on the English data set. Both models are BiLSTM-CRF with Flair embeddings, but the as-you-type model is trained with the best setting from Table 1. We can see that most confusions are between LOC and ORG: 7.9% of I-ORG tokens in the full-sentence setting are classified as I-LOC in the as-you-type setting, and 7.6% of B-ORG tokens are classified as B-LOC. Without right context, it is very challenging to distinguish these two types. For example, the data set contains many sport teams that contain location name tokens. Another noticeable difference is that the as-you-type model makes more O predictions. For instance, 5.8% of B-MISC are classified as O. This can be due to the limited availability of cues for identifying entities when right context is missing.

7.2 Positional Prediction

We expect that the size of the context impacts the quality of the inference more acutely in the as-you-type scenario when compared to the sentence-level setup. Figure 7 plots the predictive performance of three English models across each token's position. This confirms our intuition that the as-you-type setup especially impacts prediction on the first tokens in a sentence, which are more entity rich as shown in Section 4.1. However, we see that there is still a difference compared to the standard NER setup (blue curve) across all positions, confirming that indeed the right context can add more information regardless of the position of the token. The plot also highlights the performance gains of our adaptation methods at the first tokens when compared with the BiLSTM-CRF model evaluated in the as-you-type setting (orange curve).



Full sentence \ As-you-type	B-PER	I-PER	B-ORG	I-ORG	B-LOC	I-LOC	B-MISC	I-MISC	O
B-PER	95.8	0.1	1.9	0.1	1.1	0.0	0.3	0.0	0.8
I-PER	0.1	99.4	0.1	0.2	0.1	0.1	0.0	0.0	0.1
B-ORG	1.1	0.0	87.1	0.1	7.6	0.0	2.5	0.0	1.7
I-ORG	0.2	0.4	0.3	87.1	0.1	7.9	0.0	1.8	2.1
B-LOC	0.4	0.1	4.4	0.4	93.5	0.1	0.9	0.1	0.3
I-LOC	0.4	0.4	0.0	4.0	0.0	94.2	0.0	1.1	0.0
B-MISC	0.7	0.0	4.1	0.4	5.5	0.1	82.4	1.0	5.8
I-MISC	0.0	0.0	0.4	4.4	0.0	1.3	0.0	91.7	2.2
O	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.1	99.6

Figure 6: Confusion matrix of tag type prediction when comparing the best as-you-type and sentence-level models.

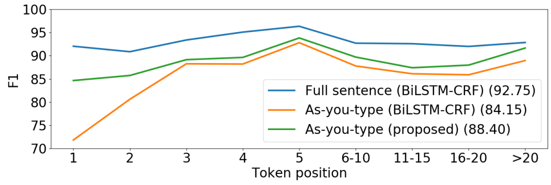


Figure 7: F_1 scores at various token positions averaged across all the sentences in the English data set. The overall performance of each model is listed in the legend.

7.3 Delayed Entity Prediction

Based on the previous results and examples, we want to study the impact of delaying the entity decision by one token. This would account for cases where the first entity word is ambiguous and also reduce the number of invalid sequences that can be produced by the as-you-type inference. For example, in the case of the 'Zimbabwe Open' entity, if the model predicted the tag of the token 'Zimbabwe' as B-LOC and after seeing the next token ('Open'), it revises this token's prediction as I-MISC, it is unable to change the B-LOC tagging, thus creating an invalid sequence, but still obtains partial credit on the token F_1 score. Delaying the decision of the first token could have allowed the model to correct its decision for the first token ('Zimbabwe') to B-MISC, resulting in a valid and more accurate sequence.

We study the possibility of delaying the prediction of a single token (not multiple tokens) by using

	English		German		Spanish	
Thr.	Tok%	F_1	Tok%	F_1	Tok%	F_1
0	0%	88.4	0%	81.62	0%	87.72
0.6	0.77%	89.18	0.97%	82.65	0.66%	88.20
0.7	1.28%	89.56	1.65%	83.09	1.15%	88.47
0.8	2.17%	90.05	2.43%	83.74	1.80%	88.66
0.9	3.25%	90.53	3.61%	84.01	2.80%	88.64
1.0	100%	91.20	100%	84.22	100%	88.91
Full sentence		92.75		84.32		89.43

Table 2: Results of the proposed wait-one policy. If the probability of prediction is less than or equal to the given threshold (Thr. column), we wait for one more token and predict again. The column Tok% indicates percentage of tokens which have a delayed prediction. The best performing model in the as-you-type setup is used. The performance of the best full-sentence model is listed in the last row for comparison purposes.

a threshold on the tag output. Table 2 shows the results of several threshold values and their impact on the total F_1.

We observe that if we delay prediction by one token for all tokens (Thr $= 1.0$), the performance is very close to the best full-sentence model, obtaining an error reduction rate of 64% (1.55 compared to 4.35) for English. Moreover, we can obtain a 50% reduction rate by only delaying the decision on 3.25% of the tokens if the downstream application deems this acceptable. These results highlight the importance of the immediate right context.

8 Untyped Entity Detection

Named entity recognition combines two different components: entity detection – identifying the named entity spans – and typing – identifying entity types. We study the impact of typing in the as-you-type setup by removing the typing information from the output labels ($E = \{\text{ENT}\}$), thus reducing the output space to $K = \{\text{B-ENT, I-ENT, O}\}$.

Results using the best as-you-type models with and without as-you-type adaptations are shown in Table 3. Comparing with the numbers in Table 1, the untyped F_1 score of as-you-type setting is much closer to the standard sentence-level evaluation, being within 1 point of F_1 for all languages. This highlights that the challenging part of the as-you-type setting is entity typing. For example, 'Zimbabwe' is a location on its own, but 'Zimbabwe Open' is an event (MISC entity type) while 'West' is usually indicative of a first location token (e.g. 'West Pacific'), but can also refer to an organization (e.g. 'West Indies' when referencing the cricket team). The proposed technique results are in this case less conclusive, which is somewhat expected as the differences in entity frequencies between full sentences and truncated sentences are smaller.

	Sentence	As-you-type	
		Original	Embedding weighting
English	97.49	97.11	97.09
German	91.37	89.29	89.49
Spanish	97.70	97.05	96.96

Table 3: Entity identification performance. The four entity types are collapsed into one type when computing token-level F_1 scores. The model for "Embedding weighting" is BiLSTM-CRF with bidirectional Flair embeddings for all three languages. For the "Original" setting, we use forward LSTM with forward Flair embeddings.

9 Conclusions

This paper introduced and motivated the as-you-type experimental setup for the popular NER task. We presented results across three different languages, which show the extent to which sentence-level state-of-the-art models degrade in this setup. Through insights gained from data analysis, we proposed modelling improvements to further reduce the gap to the regular sentence-level performance. Our error analysis highlights the cases that pose challenges to the as-you-type scenario and uncovers insights into way to further improve the modelling of this task.

This setup is tailored for end-applications such as text editors, speech-to-text, machine translation, auto-completion, or auto-correct. For text editors, the editor would be able to receive suggestions for entities inline, right after they type the entity, which can further be coupled with a linking algorithm. This would increase the user experience and efficiency of the editor, as they can make selections about entities inline (similar to a phone's autocorrect), rather than having to go back over the entire sentence after it was completed.

Another avenue of future work would be to couple the NER as-you-type with ASR data and using methods that adapt NER to noisy ASR input (Benton and Dredze, 2015) for building an end-to-end live speech to entities system.

Acknowledgements

We would like to thank Leslie Barrett, Mayank Kulkarni, Amanda Stent, Umut Topkara and the other members of the Bloomberg AI group. They provided invaluable feedback on the experiments and the paper.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019a. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019b. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Ravneet Arora, Chen-Tse Tsai, Ketevan Tsereteli, Prabhanjan Kambadur, and Yi Yang. 2019. A semi-Markov structured support vector machine model for high-precision named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5862–5866, Florence, Italy. Association for Computational Linguistics.

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 437–445, Montréal, Canada. Association for Computational Linguistics.

Adrian Benton and Mark Dredze. 2015. Entity linking for spoken language. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 225–230, Denver, Colorado. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan. The COLING 2016 Organizing Committee.

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

Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21(4):209–252.

Tomoki Fujita, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Simple, lexicalized choice of translation timing for simultaneous speech translation. In *INTERSPEECH*, pages 3487–3491.

Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. 2018. Marginal likelihood training of BiLSTM-CRF for biomedical named entity recognition from disjoint label sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2824–2829, Brussels, Belgium. Association for Computational Linguistics.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. Association for Computational Linguistics.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume*

1, Long Papers, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain adaptation with latent semantic association for named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 281–289, Boulder, Colorado. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity disambiguation for noisy social media posts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2000–2008, Melbourne, Australia. Association for Computational Linguistics.

Jian Ni, Georgiana Dinu, and Radu Florian. 2017. Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection. *arXiv preprint arXiv:1707.02483*.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 551–556, Baltimore, Maryland. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Shruti Rijhwani and Daniel Preotiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7605–7617, Online. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Jing Wang, Mayank Kulkarni, and Daniel Preotiuc-Pietro. 2020. Multi-domain named entity recognition with genre-aware and agnostic inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8476–8488, Online. Association for Computational Linguistics.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium. Association for Computational Linguistics.

A Appendices

A.1 Implementation and Hyperparameters

To train our models we use Stochastic Gradient Descent with a learning rate of 0.1 and mini batch size of 32. The LSTM model includes 1 layer of LSTM with hidden state size 256. We also employ a dropout of 0.5 for the LSTM layer. For the positional embeddings, the dimension d is set as 100 for feature weighting and 1000 for embedding weighting. We tried different dimensions between 100 and 2000. w for weighted loss is identified as 1.5 for English and 1 for German and Spanish from the dev set. For W we considered values between 0 and 2. All the models are trained for 70 epochs and the best model is selected based on the token-level F-1 score[4] on dev set. We perform manual hyperparameter selection and the final performance is reported based on the 5-10 runs of the best hyperparameter setting.We use Flair's standard settings for English.

All the models are trained on nvidia GPU and overall training for 70 epochs takes around 5-6 hours. This run-time complexity is very close to the complexity achieved by the the Flair implementation for standard NER training.

Numbers reported in (Akbik et al., 2018) are generated by training models on combined train and dev sets, hence they are higher than the numbers we report when training only on the training data. We also report token-level F1, rather than entity-level F1, which leads to results that are not directly comparable with (Akbik et al., 2018).

A.2 Visualization of Attention Weights

To better understand the impact of positional attention weights, we visualize and compare the feature-level attention weights for different tokens on a few hand-picked English sentences. Figure 8 highlights tokens using different color intensities. Higher intensity represents a larger weight value and hence a stronger influence of backward context. First, it is evident that tokens in the first position rely more heavily on backward features in the absence of any forward context, which is further reflected in the higher attention weights achieved by these tokens. Moreover, first tokens of multi-token entities such as Persons ('Nabil Abu Rdainah'), Organizations ('NY Rangers') and Events ('Zimbabwe Open') are assigned larger weights due to a high influence of immediate next tokens. Also, quite often the last token in the sentences are weighted lower which can be attributed to the positional information captured by the attention weights.

A.3 Performance on Dev Set

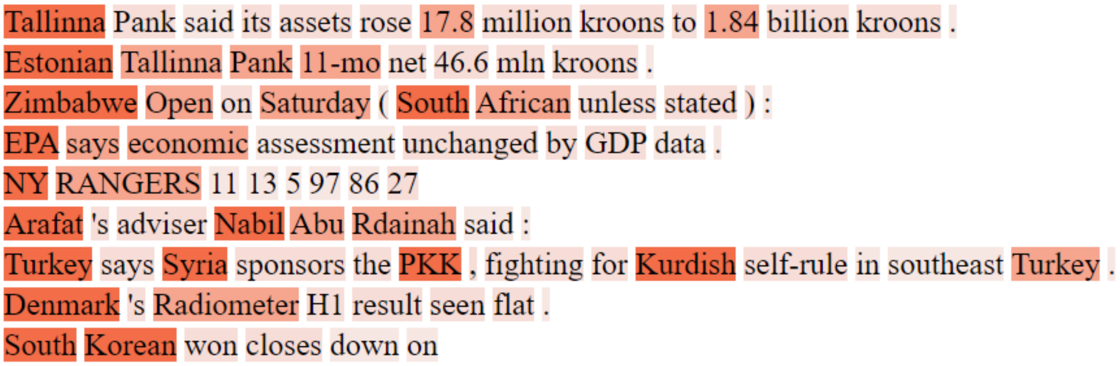
To facilitate reproducibility of results, Table 4 reports the development set performance of the base model (Bi-LSTM CRF Flair) and the proposed model for both as-you-type and sentence level setups.

A.4 Parameters

Table 5 lists different trainable parameters used in the model along with their sizes.

---

[4] `https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html#evaluation`



Tallinna Pank said its assets rose 17.8 million kroons to 1.84 billion kroons .
Estonian Tallinna Pank 11-mo net 46.6 mln kroons .
Zimbabwe Open on Saturday (South African unless stated) :
EPA says economic assessment unchanged by GDP data .
NY RANGERS 11 13 5 97 86 27
Arafat 's adviser Nabil Abu Rdainah said :
Turkey says Syria sponsors the PKK , fighting for Kurdish self-rule in southeast Turkey .
Denmark 's Radiometer H1 result seen flat .
South Korean won closes down on

Figure 8: Sample attention weights from the English CoNLL Data Set.

	English		**German**		**Spanish**	
Model	As-you-type	Sentence	As-you-type	Sentence	As-you-type	Sentence
Base Model	88.16	96.08	82.74	86.48	81.34	87.54
Proposed Model	92.54	96.43	83.71	86.69	85.75	87.78

Table 4: Performance on Conll Dev set for both Bi-LSTM-CRF Flair and the proposed final model

Parameter	Size
GloVe Embedding Dimension (English)	100
Fast-text Embedding Dimension (Spanish, German)	300
Flair Embedding Size	2,000
Feature-Level Positional Embedding Size	100
Embedding-Level Positional Embedding Size	1,000
LSTM output size	100
Bi-Directional LSTM	1,680,800
Linear Output Layer	200 * 9
CRF Transition Matrix	6 * 6
Feature Level Attention Matrix (W)	300*1
Embedding Level Attention Matrix (W_e) (English)	5,100 * 1 + 1
Embedding Level Attention Matrix (W_e) (Spanish, German)	5,300 * 1 + 1

Table 5: Number of parameters for different components of our models. When not explicitly mentioned, parameters are for models in all three languages.

SANDI: Story-and-Images Alignment

Sreyasi Nag Chowdhury **Simon Razniewski** **Gerhard Weikum**
Max Planck Institute for Informatics, Saarbrücken, Germany
`(sreyasi, srazniew, weikum)@mpi-inf.mpg.de`

Abstract

The Internet contains a multitude of social media posts and other stories where text is interspersed with images. In these contexts, images are not simply used for general illustration, but are judiciously placed in certain spots of a story for multimodal descriptions and narration. In this work we analyze the problem of text-image alignment, and present SANDI, a methodology for automatically selecting images from an image collection and aligning them with text paragraphs of a story. SANDI combines visual tags, user-provided tags and background knowledge, and uses an Integer Linear Program to compute alignments that are semantically meaningful. Experiments show that SANDI can select and align images with texts with high quality of semantic fit.

1 Introduction

It is well-known (and supported by studies Lester (2013); Messaris and Abraham (2001)) that the most powerful messages are delivered with a combination of words and pictures. On the Internet, such multimodal content is abundant in the form of news articles, social media posts, and personal blog posts where authors enrich their stories with carefully chosen and placed images. As an example, consider a vacation report, to be posted on a blog site or online community. The backbone of the travel report is a textual narration, but the user typically places illustrative images in appropriate spots, carefully selected from her photo collection from this trip. These images can either show specific objective highlights such as waterfalls, mountain hikes or animal encounters, or may serve to depict the thematic mood of the trip, e.g., by showing nice sunsets. Another example is brochures for various organizations. Here, the text describes the mission, achievements and ongoing projects, and is accompanied with judiciously selected and placed photos of buildings, people, products and images depicting the subjects of interest, e.g., galaxies or telescopes for research in astrophysics.

Figure 1: The story-and-images alignment problem.

The generation of such multimodal stories requires substantial human judgement and reasoning, and is thus time-consuming and labor-intensive. In particular, the effort on the human side includes selecting the right images from a pool of story-specific photos (e.g., the traveler's own photos) and possibly also from a broader pool for visual illustration (e.g., from Pinterest). Even if the set of photos were exactly given, there is still considerable effort to place them within appropriate paragraphs, paying attention to the semantic coherence between surrounding text and image. In this paper, we set out to automate this human task, formalizing it as a *Story-AND-Images (SANDI) alignment* problem.

Problem Statement. Given a story-like text document and a set of images, the problem is to automatically decide where individual images are placed in the text. Figure 1 depicts this task. The problem comes in different variants: either all images in the given set need to be placed, or a subset of given cardinality must be selected and aligned with text paragraphs. Formally, given n paragraphs and $m \geq n$ images, assign $b \leq n$ of these images to a subset of the paragraphs, such that each paragraph has at most one image.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 989–999
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Prior Work and its Inadequacy. There is ample literature on computer support for multimodal content creation, most notably, on generating image captions. Closest to our problem is work on *Story Illustration* (Joshi et al., 2006; Schwarz et al., 2010) where the task is to select illustrative images from a large pool. However, the task is quite different from ours, making prior approaches inadequate for the setting of this paper. First, unlike story illustration, we need to consider the text-image alignments jointly for all pieces of a story, rather than making context-free choices one piece at a time. Second, prior work assumes that each image in the pool has an informative caption or set of tags, by which the selection algorithm computes its choices. Our model harnesses visual tags from deep neural network based object-detection frameworks and incorporates background knowledge, as automatic steps to enrich the semantic interpretation of images.

Our Approach – SANDI. We present a framework that casts the story-images alignment task into a combinatorial optimization problem. The objective function, to be maximized, captures the semantic coherence between each paragraph and the co-located image. To this end, we consider a suite of features – the visual tags associated with an image (automatically detected tags as well as user-defined tags when available), text embeddings, and also background knowledge. The optimization is constrained by the number of images that the story should be enriched with. As a solution algorithm, we devise an integer linear program (ILP) and employ the Gurobi ILP solver for computing the exact optimum. Experiments show that SANDI produces semantically coherent alignments. A demonstration of SANDI (Nag Chowdhury et al., 2020) can be viewed at `https://youtu.be/k5gu2pNxdNU`.

Contributions. To the best of our knowledge, this is the first work to address story-images alignment. Our salient contributions are:

1. We introduce and define the problem of story-images alignment.
2. We analyze two real-world datasets of stories with rich visual illustrations, and derive insights on alignment decisions and quality measures.
3. We devise relevant features, formalize the alignment task as a combinatorial optimization problem, and develop an exact-solution algorithm using integer linear programming.
4. We compare our method against baselines that use multimodal embeddings.

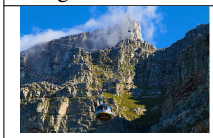
2 Related Work

Existing work on associations between text and images can be categorized into the following areas.

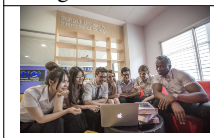
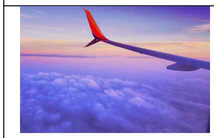
Image Attribute Recognition. High level concepts in images lead to better results in Vision-to-Language problems (Wu et al., 2016). Traditionally image tagging was based on community input (Gupta et al., 2010). Modern deep-learning based tools detect objects (Hoffman et al., 2014; Redmon and Farhadi, 2017; Ren et al., 2015) and scenes (Zhou et al., 2014) in images. Inter-concept incoherence can also be refined using background knowledge (Nag Chowdhury et al., 2018). We leverage some frameworks from this category in our model to detect visual concepts in images.

Story Illustration. Prior work finds suitable images from annotated image collections to illustrate personal stories (Joshi et al., 2006; Ravi et al., 2018) or news posts (Schwarz et al., 2010; Delgado et al., 2010). The results are presented as clusters of related images (Guan et al., 2011), or an illustrated article (Jhamtani et al., 2016). Story illustration only addresses the problem of image selection, whereas we solve two problems simultaneously: image selection and image placement – making a joint decision on all pieces of long complex stories. This makes our problem distinct. There is no way to systematically compare our full-blown model with prior works on story illustration alone.

Multimodal Embeddings. A popular method of semantically comparing images and text has been to map textual and visual features into a common space of multimodal embeddings (Frome et al., 2013; Vendrov et al., 2016; Faghri et al., 2018; Wu et al., 2019; Wang et al., 2019; Liu et al., 2019). Visual-Semantic-Embeddings (VSE) has been used for generating captions for whole images (Faghri et al., 2018), or to associate text with image regions (Karpathy and Li, 2015). Color, geometry, aspect-ratio have been used to align image regions to nouns (“chair”), attributes (“big”), and pronouns (“it”) in corresponding text (Kong et al., 2014). Recent work train on document-level co-occurrences and predict links between images and sentences in a document (Hessel et al., 2019; Chu and Kao, 2017). However, alignment of small image regions to text snippets or linking images to single sentences play little role in jointly interpreting the correlation between images and a larger body of text. We focus on the latter in this work.

Image	Ground Truth Paragraph
	. . . Table Mountain Cableway. The revolving car provides 360 degree views as you ascend this mesmerising 60-million-year-old mountain. From the upper cableway station. . .
	. . . On the east flank of the hill is the old Muslim quarter of the Bo-Kaap; have your camera ready to capture images of the photogenic pastel-painted colonial period homes. . .

(a) Sample image and paragraph from Lonely Planet

Image	Ground Truth Paragraph
	. . . If you are just looking for some peace and quiet or hanging out with other students...library on campus, a student hangout space in the International College building. . . .
	. . . I was scared to travel alone. But I quickly realized that there's no need to be afraid. Leaving home and getting out of your comfort zone is an important part of growing up.. . .

(b) Sample image and paragraph from Asia Exchange

Figure 2: Text-image pairs from our datasets.

Image Caption Generation. Most prior works generate factual captions (Xu et al., 2015; Tan and Chan, 2016; Lu et al., 2017; Wang et al., 2018), while some recent architectures venture into producing stylized captions (Gan et al., 2017) and stories (Zhu et al., 2015; Krause et al., 2017). An image caption can be considered as a precise focused description of an image without much superfluous or contextual information. However, in a multimodal story, the paragraphs surrounding an image contain detailed thematic descriptions. We try to capture the thematic indirection between an image and surrounding text, thus making the problem distinct.

Commonsense Knowledge for Story Understanding. One of the earliest applications of Commonsense Knowledge (CSK) to interpret text-image associtions is a photo agent which automatically annotated images from user's multi-modal (text-image) emails or web pages, while also inferring additional CSK concepts (Lieberman and Liu, 2002). Subsequent works used CSK reasoning to infer causality in stories (Williams et al., 2017). We enhance automatically detected objects and scenes in image with relevant CSK concepts from ConceptNet (Speer et al., 2017). This often helps to capture more context about an image.

3 Dataset and Problem Analysis

3.1 Dataset

To the best of our knowledge, there is no experimental dataset for text-image alignment. We therefore compile two datasets from two blogging sites:

- Lonely Planet[1]: 2178 travel stories containing 20 paragraphs and 4.5 images per story. Most images are accompanied by captions. The images come from the author's personal archives and adhere strictly to the content of the story.
- Asia Exchange[2]: 200 stories about education opportunities in Asia, with 13.5 paragraphs and 4 images per story. Some stories contain generic stock images complying with the abstract theme. Most images have captions.

Figure 2 shows image-paragraph examples from the datasets.

Text-Image Semantic Coherence. To understand human judgments behind text-image pairing, we analyze 50 randomly chosen images and their corresponding paragraphs from the Lonely Planet dataset. We identify six possibly overlapping concept classes that appear in images as well as in their corresponding paragraphs: (i) natural named objects such as Mt. Everest (ii) human activities such as biking (iii) generic objects such as cars (iv) general nature scenes such as forest (v) specific man-made entities such as monuments (vi) geographic locations such as Rome. The outcome of this analysis is shown in Table 1.

Concept Classes	% of text-images pairs with shared concepts
Natural named objects	9%
Human activities	12%
Generic objects	15%
General nature scenes	20%
Man-made named objects	21%
Geographic locations	29%

Table 1: Reasons for text-image semantic coherence.

3.2 Image Tags

Based on the analysis in Table 1, we consider the following kinds of tags for describing images:

Visual Tags (CV). We use three state-of-the-art computer-vision methods for object and scene detection. First, deep convolutional neural networks based architectures like LSDA (Hoffman et al., 2014) and YOLO (Redmon and Farhadi, 2017), are used to detect objects like *person*, *frisbee* or *bench*, that denote "Generic objects" from Table 1.

---

[1] www.lonelyplanet.com/blog

[2] www.asiaexchange.org

For stories, general scene descriptors like *restaurant* or *beach* play a major role, too. Therefore, our second asset is scene detection from the MIT Scenes Database (Zhou et al., 2014). These constitute "General nature scenes" from Table 1. Thirdly, since stories often abstract away from explicit visual concepts, a framework that incorporates abstractions into visual detections – VISIR (Nag Chowdhury et al., 2018) – is also leveraged. For e.g., the concept "hiking" is supplemented with the concepts "walking" (Hypernym of "hiking" from WordNet) and "fun" (from ConceptNet (Speer et al., 2017) assertion ⟨ *hiking, has property, fun*⟩).

User Tags (MAN). Owners of images often have additional knowledge about content and context – for e.g., activities or geographical information ("hiking near Lake Placid"), which, from Table 1 play a major role in text-image alignment. For experiments, we use nouns and adjectives from image captions from our datasets as user tags. In downstream applications, images can be selected either from web repositories or from a personal collection. In the former case, explicit tags or words from captions/titles serve as user tags. In the latter case, location details like names of places can be easily inferred from metadata like GPS coordinates associated with "raw" phone/camera images.

Big-data Tags (BD). Big data and crowd knowledge allow to infer additional context that may not be visually apparent. We utilize the Google reverse image search API[3] to incorporate such tags. This API allows to search by image, and suggests tags based on visually similar images in the vast web image repository. These tags depict popular places, such as "Savarmati Ashram", or "Mexico City insect market", and thus constitute "Natural names objects", "Man-made named objects", as well as "Geographic locations" from Table 1.

To further improve the semantic characterization of an image, we extend the tag set of an image by related commonsense knowledge concepts.

Commonsense Knowledge (CSK). CSK can bridge the gap between visual and textual concepts (Nag Chowdhury et al., 2016). CV, BD, and MAN tags are enriched with CSK from the following ConceptNet relations – *used for, has property, causes, at location, located near, conceptually related to*. E.g., for the left image in Figure 3, we add CSK concept "show talent" from CV tag "stage" from the assertion ⟨stage, *used for*, show talent⟩. CSK concepts cover multiple classes from Table 1. Owing to the noise and subjectivity in ConceptNet, only concepts which are informative for a given image are retained. If the top-10 web search results of a CSK concept are semantically similar to the image tags (CV/MAN/BD), the CSK concept is considered to be *informative* for the image. Cosine similarity between the mean vectors (from word2vec) of the image context and the search results is used as a measure of semantic similarity.

Figure 3 shows examples of the image tags. In use cases all features are not always available – user tags may not exist or may not be retained during web distribution, big data requires access to paid APIs , and visual tags are error-prone. We will thus study the features in isolation and jointly.

CV: person, sunglasses, stage
MAN: Globe Theatre
BD: Shakespeare's Globe
CSK: show talent, attend concert, entertain audience

CV: terra cotta, village
MAN:tiled rooftops
BD: Languedoc Roussillon
CSK: colony, small town

Figure 3: Types of image tags. CV – visual objects/scenes, MAN and BD – nuanced descriptions, locations, CSK - high-level thematic concepts.

4 Model for Story-Images Alignment

Our *story-image alignment* model constitutes an Integer Linear Program (ILP) which jointly optimizes the placement of selected images within a story. The main ingredient for this alignment is the pairwise similarity between images and units of text. We consider a paragraph as a text unit.

Text-Image Pairwise Similarity. Given an image, each of the three kinds of descriptors of Section 3.2 gives rise to a bag of features. We use these features to compute *text-image semantic relatedness scores* $srel(i,t)$ for an image i and a paragraph t.

$$srel(i,t) = cosine(\vec{i},\vec{t}) \tag{1}$$

where $\vec{i}$ and $\vec{t}$ are the mean word embeddings for the image tags and the paragraph respectively. For images, we use all detected tags. For paragraphs, we consider only the top 50% of concepts w.r.t. their TF-IDF ranking over the entire dataset. We use word embeddings from word2vec trained on Google News Corpus. $srel(i,t)$ scores from Equation 1 serve as weights for variables in the ILP.

---

[3] `www.google.com/searchbyimage`

Tasks. Our problem has two distinct tasks: 1. *Image Selection* – to select relevant images from an image pool. 2. *Image Placement* – to place selected images in the story. These two components are modelled into one ILP where Image Placement is achieved by maximizing an objective function, while Image Selection is styled by constraints. In the following subsections we discuss two flavors of our model consisting of one or both the tasks.

4.1 Complete Alignment

Complete Alignment constitutes the problem of aligning *all* images in a given image collection with relevant text units of a story. Hence, only *Image Placement* is applicable. For a story with $|T|$ text units and an associated image album with $|I|$ images, the alignment of images $i \in I$ to text units $t \in T$ can be modeled as an Integer Linear Program (ILP) with the following definitions:

Decision Variables. The following binary decision variables are introduced: $X_{it} = 1$ if image i should be aligned with text unit t, 0 otherwise.

Objective. Select image i to be aligned with text unit t such that the semantic relatedness over all text-image pairs is maximized:

$$max\left[\sum_{i\in I}\sum_{t\in T} srel(i,t)X_{it}\right] \qquad (2)$$

where $srel(i,t)$ is the text-image semantic relatedness from Equation 1.

Constraints. We make two assumptions for text-image alignments – no image may be repeated in the story (Constraint 3), and no paragraph may be aligned with multiple images (Constraint 4). The former is a trivial observation from multimodal presentations on the web such as in blog posts, newswire, brochures. The latter is made based on the nature of our datasets, and it is designed as a hard constraint in order to facilitate a fair evaluation.

$$\sum_{i} X_{it} \leq 1 \forall t \quad (3) \qquad\qquad \sum_{t} X_{it} = 1 \forall i \quad (4)$$

4.2 Selective Alignment

Selective Alignment is the flavor of the model which selects a certain number of thematically relevant images from a big image pool, and places them within the story. Hence, it constitutes both tasks – *Image Selection* and *Image Placement*. Along with the constraint in (3), Image Selection entails the following additional constraints:

$$\sum_{t} X_{it} \leq 1 \forall i \quad (5) \qquad\qquad \sum_{i}\sum_{t} X_{it} = b \quad (6)$$

where b is the budget for the number of images in the story. b may be simply defined as the number of paragraphs in the story, following our assumption that each paragraph may be associated with a maximum of one image. (5) is an adjustment to (4) which implies that not all images from the image pool need to be aligned with the story. The objective function from (2) rewards the selection of best fitting images from the image pool.

5 Quality Measures

In this section we define metrics for automatic evaluation of the text-image alignment problem. The two tasks involved – *Image Selection* and *Image Placement* – call for separate evaluation metrics as discussed below.

5.1 Image Selection

Representative images for a story are selected from a big pool of images. There are multiple conceptually similar images in our image pool since they have been gathered from blogs of the domain "travel". Hence evaluating the results on strict precision (based on exact matches between selected and ground-truth images) does not necessarily assess true quality. We therefore define a relaxed precision metric (based on semantic similarity) in addition to the strict metric. Given a set of selected images I and the set of ground truth images J, where $|I| = |J|$, the precision metrics are:

$$RelaxedPrecision = \frac{\sum_{i\in I, j\in J} max(cosine(\vec{i},\vec{j}))}{|I|} \qquad (7)$$

$$StrictPrecision = \frac{|I \cap J|}{|I|} \qquad (8)$$

5.2 Image Placement

For each image in a multimodal story, the ground truth (GT) paragraph is assumed to be the one following the image in our datasets. To evaluate the quality of SANDI's text-image alignments, we compare the GT paragraph and the paragraph assigned to the image by SANDI (henceforth referred to as "aligned paragraph"). We propose the following metrics for evaluating the quality of alignments:

BLEU and ROUGE. BLEU and ROUGE are classic n-gram-overlap-based metrics for evaluating

	BLEU	ROUGE	SemSim	ParaRank	Order Preserve
RandomAlign	3.1	6.9	75.1	50.0	50.0
VSE++	11.0	9.5	84.6	59.1	55.2
VSE++ ILP	12.6	11.2	84.0	58.1	48.0
SANDI-CV	18.2	17.6	86.3	63.7	54.5
SANDI-MAN	**45.6**	**44.5**	**89.8**	72.5	**77.4**
SANDI-BD	26.6	25.1	84.7	61.3	61.2
SANDI$*$	44.3	42.9	89.7	**73.2**	76.3

Table 2: Complete Alignment: Lonely Planet.

	BLEU	ROUGE	SemSim	ParaRank	Order Preserve
RandomAlign	6.8	8.9	70.8	50.0	50.0
VSE++	19.4	17.7	85.7	51.9	48.0
VSE++ ILP	23.5	20.1	86.0	52.6	46.1
SANDI-CV	21.5	20.6	87.8	58.4	52.0
SANDI-MAN	**35.2**	**32.2**	89.2	61.5	61.5
SANDI-BD	24.1	22.3	86.7	56.0	53.6
SANDI$*$	33.4	31.5	**89.7**	**62.4**	**62.5**

Table 3: Complete Alignment: Asia Exchange.

machine translation and text summarization. Although known to be limited insofar as they do not recognize synonyms and semantically equivalent formulations, they are in widespread use. We consider them as basic measures of concept overlap between GT and aligned paragraphs.

Semantic Similarity. To alleviate the shortcoming of requiring exact matches, we consider a metric based on embedding similarity. We compute the similarity between two text units t_i and t_j by the average similarity of their word embeddings, considering all unigrams and bigrams as words.

$$SemSim(t_i, t_j) = cosine(\vec{t_i}, \vec{t_j}) \tag{9}$$

where $\vec{x}$ is the mean vector of words in x. For this calculation, we drop uninformative words by keeping only the top 50% with regard to their TF-IDF weights over the whole dataset.

Average Rank of Aligned Paragraph. We associate each paragraph in the story with a ranked list of all the paragraphs on the basis of semantic similarity (Eq. 9), where rank 1 is the paragraph itself. Our goal is to produce alignments ranked higher with the GT paragraph. The average rank of alignments by a model is computed as follows:

$$ParaRank = 1 - \left[\left(\frac{\sum_{t \in T'} rank(t)}{|I|} - 1 \right) \Big/ \left(|T| - 1 \right) \right] \tag{10}$$

where $|I|$ is the number of images and $|T|$ is the number of paragraphs in the story. $T' \subset T$ is the set of paragraphs aligned to images. Scores are normalized between 0 and 1; 1 being the perfect alignment and 0 being the worst alignment.

Order Preservation. Most stories either follow a time-line or storyline. Images placed at meaningful spots within the text would ideally adhere to this sequence. Hence the measure of pairwise ordering provides a sense of maintaining or respecting the storyline. It can be defined as the number of order preserving image pairs in the alignment (i_m, i_n) normalized by the total number of ordered image pairs in the ground truth.

$$OrderPreserve = \frac{|(i_m, i_n)|}{(|I|(|I| - 1)/2)} \tag{11}$$

Correlation between Metrics. Table 4 shows the pairwise correlation between our evaluation metrics – BLEU, ROUGE, *SemSim*, *ParaRank*, *Orderpreserve* – computed on a random alignment of images to paragraphs in 100 stories. Not surprisingly, BLEU and ROUGE correlate nearly perfectly, and show reasonable correlation to $SemSim$ and $ParaRank$. $OrderPreserve$ works at a different level and exhibits virtually no correlation to the other metrics. This illustrates that order-preserving alignments are not necessarily semantically meaningful, and vice versa.

	ROUGE	SemSim	ParaRank	Order Preserve
BLEU	0.98	0.32	0.39	-0.23
ROUGE		0.33	0.40	-0.23
SemSim			0.29	0.08
ParaRank				-0.06

Table 4: Correlation between evaluation metrics.

6 Experiments and Results

We evaluate the two flavors of SANDI – *Complete Alignment* and *Selective Alignment* – based on the quality measures from Section 5.

6.1 Setup

Tools. Deep learning based architectures – LSDA (Hoffman et al., 2014), YOLO (Redmon and Farhadi, 2017), VISIR (Nag Chowdhury et al., 2018) and Places-CNN (Zhou et al., 2014) are used as sources of *Visual tags (CV)*. Google reverse image search tag suggestions are used as *Big-data tags (BD)*. We use the Gurobi Optimizer for solving the ILP. A Word2Vec (Mikolov et al., 2013) model trained on the Google News Corpus encompasses a large cross-section of domains, and hence is used as a source of word embeddings.

SANDI Variants. The variants of our text-image alignment model are based on the use of image descriptors described in Section 3.2.

- SANDI-CV, SANDI-MAN, and SANDI-BD use CV, MAN, and BD tags respectively.
- SANDI$*$ combines CV, MAN, and BD tags.

6.2 Complete Alignment

We evaluate our Complete Alignment model (defined in Section 4.1), which places *all* images from a given image pool within a story.

Baselines. To the best of our knowledge, there is no existing work on story-image alignment. Hence we modify methods on joint visual-semantic-embeddings (VSE) (Kiros et al., 2014; Faghri et al., 2018) to serve as baselines, henceforth referred to as VSE++. We compare SANDI with:

- RandomAlign: a simple baseline with random image-text alignments.
- VSE++: for an image, VSE++ is adapted to produce a ranked list of paragraphs from the given story. The top paragraph is considered as an alignment, with a greedy constraint that one paragraph can be aligned to at most one image.
- VSE++ ILP: using cosine similarity scores between image and paragraph from the joint visual-semantic embedding space, we solve an ILP as described in Section 4.

Since there are no existing story-image alignment datasets, VSE++ has been trained on the MSCOCO captions dataset (Lin et al., 2014), which contains 330K images with 5 captions per image.

Evaluation. Tables 2 and 3 show the performance of the baselines and the SANDI variants on the Lonely Planet and Asia Exchange datasets respectively. SANDI outperforms the baselines on all evaluation metrics to various degrees. While VSE++ looks at each image in isolation, SANDI captures context better by considering all text units of the story and all images from the corresponding album at once in a constrained optimization problem. VSE++ ILP, although closer to SANDI in methodology, does not outperform SANDI. This can be attributed to the fact that SANDI is less tied to a particular dataset, relying only on word2vec embeddings that are trained on a much larger corpus than MSCOCO. On Lonely Planet, SANDI-MAN is the best configuration – this is expected since user tags (MAN) contain concepts most specific to the story. SANDI$*$ marginally outperforms it on Asia Exchange – recall that images in this dataset are sometimes generic thematic illustrations, hence a combination of all features capture more context. The consistency of scores across both datasets highlight the robustness of SANDI.

Role of Commonsense Knowledge (CSK). We observe that CSK helps improve performance of SANDI-CV. This is intuitive because CV tags denote only explicit objects and scenes, which do not capture high-level concepts of the images. CSK alleviates this to some extent. For example, in the first image in Figure 3 – CSK (*show talent, attend concert, entertain audience*) appends a more meaningful context to the CV tags (*person, sunglasses, stage*); MAN and BD tags already capture a broader context. Table 5 compares SANDI-CV and SANDI-CV-CSK on Asia Exchange.

	BLEU	ROUGE	SemSim	ParaRank	Order Preserve
SANDI-CV	21.5	20.6	87.8	58.4	52.0
SANDI-CV -CSK	19.9	19.4	**88.3**	**62.4**	50.0

Table 5: Role of Commonsense Knowledge: Asia Exchange

6.3 Selective Alignment

This variation of our model, as defined in Section 4.2, solves two tasks – *Image Selection* and *Image placement*.

6.3.1 Image Selection

Setup. In addition to the setup described in Section 6.1, some additional requirements are:

- Image pool – We pool images from stories in our dataset. Since stories from a particular domain (e.g. travel blogs) are largely quite similar, images in the pool may also be very similar in content – e.g., stories on *hiking* contain images containing *mountain, person, backpack*.
- Image budget – For each story, the number of images in the ground truth is considered as the image budget b (Equation 4.2).

Baselines. We compare SANDI with:

- RandomAlign: a baseline of randomly selected images from the pool.
- NN: nearest neighbors from a common embedding space of images and paragraphs. Images are represented as mean vectors of their tags, and paragraphs are represented as mean vectors of their distinctive words. The basic word vectors are obtained from Word2Vec trained on Google News Corpus.

Tag Space	Precision	Random	NN	VSE++	SANDI
CV	*Strict*	0.4	2.0	1.14	**4.18**
	Relaxed	42.16	52.68	29.83	**53.54**
MAN	*Strict*	0.4	3.95	-	**14.57**
	Relaxed	37.14	42.73		**49.65**
BD	*Strict*	0.4	1.75	-	**2.71**
	Relaxed	32.59	37.94		**38.86**
∗	*Strict*	0.4	4.8	-	**11.28**
	relaxed	43.84	50.06		**54.34**

Table 6: Selective Alignment-Image Selection:Lonely Planet.

Tag Space	Precision	Random	NN	VSE++	SANDI
CV	*Strict*	0.45	0.65	0.44	**0.79**
	Relaxed	55.0	**57.64**	30.05	57.2
MAN	*Strict*	0.45	0.78	-	**3.42**
	Relaxed	40.24	52.0		**52.87**
BD	*Strict*	0.45	0.82	-	**0.87**
	Relaxed	31.12	**33.27**		33.25
∗	*Strict*	0.45	1.04	-	**1.7**
	relaxed	55.68	58.1		**58.2**

Table 7: Selective Alignment-Image Selection:Asia Exchange.

	BLEU	ROUGE	SemSim	ParaRank
RandomAlign	0.31	0.26	69.18	48.16
VSE++	1.04	0.8	79.18	53.09
VSE++ ILP	1.23	1.03	79.04	53.96
SANDI-CV	1.70	1.60	83.76	61.69
SANDI-MAN	**8.82**	**7.40**	82.95	66.83
SANDI-BD	1.77	1.69	**84.66**	**76.18**
SANDI∗	6.82	6.57	84.50	75.84

Table 8: Selective Alignment-Image Placement:Lonely Planet.

	BLEU	ROUGE	SemSim	ParaRank
RandomAlign	2.06	1.37	53.14	58.28
VSE++	2.66	1.39	58.00	64.34
VSE++ ILP	2.78	1.47	57.65	64.29
SANDI-CV	1.04	1.51	60.28	75.42
SANDI-MAN	**3.49**	**2.98**	61.11	**82.00**
SANDI-BD	1.68	1.52	**76.86**	70.41
SANDI∗	1.53	1.84	64.76	80.57

Table 9: Selective Alignment-Image Placement:Asia Exchange.

- VSE++: a joint visual-textual embeddings method presented in (Faghri et al., 2018) is adapted to retrieve the top-b images for a story.

Evaluation. We evaluate *Image Selection* by the measures in Section 5.1. Table 6 shows the results for SANDI and the baselines on a pool of 500 images from Lonely Planet. NN and SANDI both use Word2Vec for text-image similarity. SANDI's better scores are attributed to the joint optimization over the entire story, as opposed to greedy selection in case of NN. VSE++ uses a joint text-image embeddings space for similarity scores. Our evaluation metric $RelaxedPrecision$ (Eq. 7) factors in the semantic similarity between images based on the image descriptors (Section 3.2). Hence we compute results on the different image tag spaces, where '∗' refers to the combination of CV, MAN, and BD. The baseline VSE++ however, operates only on visual features; hence we report its performance only for CV. Results on Asia Exchange are similar (Table 7). Recall from Section 3.1 that the Asia Exchange dataset often has stock images for generic illustration rather than only story-specific images. Hence the average relaxed precision on image selection is comparitively higher. Figure 4 shows image selection results for one story. The original story contains 17 paragraphs; only the main concepts from the story have been retained for readability. SANDI is able to retrieve 2 ground-truth (GT) images out of 7, while the baselines retrieve 1 each. Note that SANDI's non-exact matches are thematically similar to GT – images in the 4th column of both GT and SANDI feature a yellow train in a backdrop of mountains, images in the 5th column show sunset. This can be attributed to the wider space of concepts that SANDI explores through the image tags from Section 3.2.

6.3.2 Image Placement

Having selected thematically related images from a big image pool, SANDI places them within contextual paragraphs of the story. Note that SANDI integrates the *Image Selection* and *Image Placement* stages into joint inference on selective alignment, whereas the baselines operate in two steps.

We evaluate the alignments by the measures from Section 5.2. Note that the measure *OrderPreserve* does not apply to Selective Alignment since the images are selected from a pool of mixed images which cannot be ordered. From Table 8 and 9 we observe that SANDI outperforms the baselines by a clear margin, harnessing its more expressive pool of tags. We show anecdotal evidence of the diversity of our image tags in Figure 3 and Table 10.

6.4 Role of Model Components

Image Descriptors. The wide variety of image tags that SANDI leverages (CV, BD, MAN) capture special characteristics of the images. These are unavailable to baselines such as VSE++, attributing to their poor performance.

Embeddings. The nature of embeddings is decisive towards alignment quality. Joint visual-semantic-embeddings trained on MSCOCO (used by VSE++) fall short in capturing high-level se-

Story: Clatter into Lisbon's steep Alfama aboard a classic yellow tram...Ride a regular bus for view of the metropolis...Venice ...opting for a vaporetto (water taxi) instead of a private punt...Hungary. Trundle alongside the Danube...Castle Hill... Istanbul...Ferries crossing the Bosphorus strait...Sail at sunset...Monte Carlo's electric-powered ferry boats...The 'Coast ...Tram' skirts...Belgium's North Sea shoreline...Pretty but pricey Geneva...travel on buses, trams and taxi-boats.

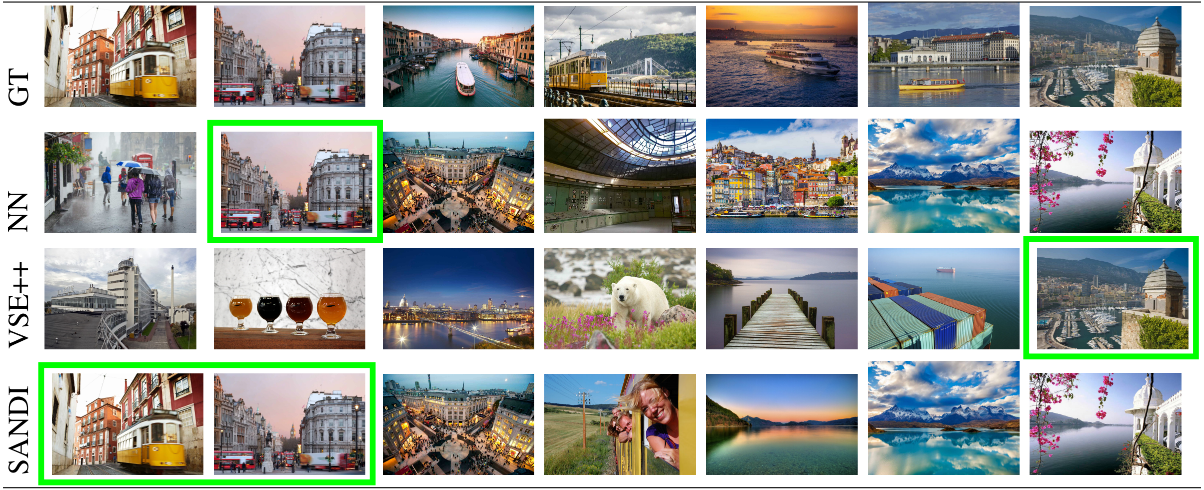


Figure 4: Image Selection. Images within green boxes are exact matches with ground truth (GT). SANDI retrieves more exact matches than the baselines (NN, VSE++). SANDI's non-exact matches are also much more thematically similar to the GT.

Image and detected concepts	SANDI-CV	SANDI-MAN	SANDI-BD
 CV: snowy mountains, massif, alpine glacier, mountain range MAN: outdoor lover, New Zealand, study destination, BD: New Zealand	New Zealand produced the first man to ever climb Mount Everest and also the creator of the bungee-jump. Thus, it comes as no surprise that this country is filled with adventures and adrenaline junkies.	Moreover, the wildlife in New Zealand is something to behold. Try and find a Kiwi! (The bird!) They are nocturnal creatures so it is quite a challenge. New Zealand is also home to the smallest dolphin species. Lastly, take the opportunity to search for the beautiful yellow-eyed penguin.	Home to hobbits, warriors, orcs and dragons. If you're a fan of the famous trilogies, Lord of the Rings and The Hobbit, then choosing New Zealand should be a no-brainer.

Table 10: Example alignments. Highlighted texts show similar concepts between image and aligned paragraphs.

mantics between images and story. Word2Vec embeddings trained on a much larger and domain-independent Google News corpus better represents high-level image-story interpretations.

ILP. As observed in Tables 6 and 7, Combinatorial optimization (SANDI) outperforms greedy optimization approaches (NN), both methods using the same embedding space.

7 Conclusion

In this paper we introduced the problem of story-images alignment – selecting and placing a set of representative images within a story. We analyzed features towards meaningful alignments from real-world multimodal datasets – Lonely Planet and Asia Exchange blogs – and defined various evaluation measures. We presented SANDI, a methodology for automating such alignments by a constrained optimization problem maximizing semantic coherence between text-image pairs jointly for the entire story. Evaluations show that SANDI produces semantically meaningful alignments. Nevertheless, some follow-up questions arise.

Additional Features. Our feature space covers most natural aspects. In addition, GPS locations where available may provide cues for geographic named entities, while timestamps may capture temporal aspects of a storyline.

Abstract and Metaphoric Relations. We do not address stylistic elements like metaphors and sarcasm in text, which would entail more challenging alignments. For example, the text "the news was a dagger to his heart" should not be paired with a picture of a dagger. Although user provided tags may provide some cues towards such abstract relationships, a deeper understanding of semantic coherence is desired.

The proposed text-image alignment system is available at `https://sandi.mpi-inf.mpg.de`, and a video of the demonstration can be viewed at `https://youtu.be/k5gu2pNxdNU`.

References

Wei-Ta Chu and Ming-Chih Kao. 2017. Blog article summarization with image-text alignment techniques. In *ISM*.

Diogo Delgado, João Magalhães, and Nuno Correia. 2010. Automated illustration of news stories. In *ICSC*.

Fartash Faghri, David J. Fleet, Jamie Kiros, and Sanja Fidler. 2018. VSE++: improving visual-semantic embeddings with hard negatives. In *BMVC*.

Andrea Frome, Gregory S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *NIPS*.

Chuang Gan, Zhe Gan, Xiaodong He, Jianfeng Gao, and Li Deng. 2017. Stylenet: Generating attractive visual captions with styles. In *CVPR*.

Genliang Guan, Zhiyong Wang, Xian-Sheng Hua, and David Dagan Feng. 2011. *StoryImaging*: a media-rich presentation system for textual stories. In *ACM MM*.

Manish Gupta, Rui Li, Zhijun Yin, and Jiawei Han. 2010. Survey on social tagging techniques. *SIGKDD Explorations*.

Jack Hessel, Lillian Lee, and David Mimno. 2019. Unsupervised discovery of multimodal links in multi-image, multi-sentence documents. In *EMNLP*.

Judy Hoffman, Sergio Guadarrama, Eric Tzeng, Ronghang Hu, Jeff Donahue, Ross B. Girshick, Trevor Darrell, and Kate Saenko. 2014. LSDA: large scale detection through adaptation. In *NIPS*.

Harsh Jhamtani, Shubham Varma, Midhun Gundapuneni, and Siddhartha Kumar Dutta. 2016. A supervised approach for text illustration. In *ACM MM*.

Dhiraj Joshi, James Ze Wang, and Jia Li. 2006. The story picturing engine - a system for automatic text illustration. *TOMCCAP*.

Andrej Karpathy and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. 2014. Unifying visual-semantic embeddings with multimodal neural language models. *CoRR*.

Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. 2014. What are you talking about? text-to-image coreference. In *CVPR*.

Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei Fei. 2017. A hierarchical approach for generating descriptive image paragraphs. In *CVPR*.

Paul Martin Lester. 2013. *Visual communication: Images with messages*. Cengage Learning.

Henry Lieberman and Hugo Liu. 2002. Adaptive linking between text and photos using common sense reasoning. In *AH*.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. In *ECCV*.

Chunxiao Liu, Zhendong Mao, An-An Liu, Tianzhu Zhang, Bin Wang, and Yongdong Zhang. 2019. Focus your attention: A bidirectional focal attention network for image-text matching. In *ACM MM*.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*.

Paul Messaris and Linus Abraham. 2001. The role of images in framing news stories. In *Framing public life*. Routledge.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Sreyasi Nag Chowdhury, William Cheng, Gerard de Melo, Simon Razniewski, and Gerhard Weikum. 2020. Illustrate your story: Enriching text with images. In *WSDM*.

Sreyasi Nag Chowdhury, Niket Tandon, Hakan Ferhatosmanoglu, and Gerhard Weikum. 2018. VISIR: visual and semantic image label refinement. In *WSDM*.

Sreyasi Nag Chowdhury, Niket Tandon, and Gerhard Weikum. 2016. Know2look: Commonsense knowledge for visual search. In *AKBC*.

Hareesh Ravi, Lezi Wang, Carlos Muñiz, Leonid Sigal, Dimitris N. Metaxas, and Mubbasir Kapadia. 2018. Show me a story: Towards coherent neural story illustration. In *CVPR*.

Joseph Redmon and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. In *CVPR*.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*.

Katharina Schwarz, Pavel Rojtberg, Joachim Caspar, Iryna Gurevych, Michael Goesele, and Hendrik P. A. Lensch. 2010. Text-to-video: Story illustration from online photo collections. In *KES*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Ying Hua Tan and Chee Seng Chan. 2016. phi-lstm: A phrase-based hierarchical LSTM model for image captioning. In *ACCV*.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. In *ICLR*.

Tan Wang, Xing Xu, Yang Yang, Alan Hanjalic, Heng Tao Shen, and Jingkuan Song. 2019. Matching images and text with multi-modal tensor fusion and re-ranking. In *ACM MM*.

Ziwei Wang, Yadan Luo, Yang Li, Zi Huang, and Hongzhi Yin. 2018. Look deeper see richer: Depth-aware image paragraph captioning. In *ACM Multimedia*, pages 672–680. ACM.

Bryan Williams, Henry Lieberman, and Patrick H. Winston. 2017. Understanding stories with large-scale common sense. In *COMMONSENSE*.

Qi Wu, Chunhua Shen, Lingqiao Liu, Anthony R. Dick, and Anton van den Hengel. 2016. What value do explicit high level concepts have in vision to language problems? In *CVPR*.

Yiling Wu, Shuhui Wang, Guoli Song, and Qingming Huang. 2019. Learning fragment self-attention embeddings for image-text matching. In *ACM MM*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. 2014. Learning deep features for scene recognition using places database. In *NIPS*.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story like visual explanations by watching movies and reading books. In *ICCV*.

Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets

Patrick Lewis†‡**, Pontus Stenetorp**‡**, Sebastian Riedel**†‡
†Facebook AI Research; ‡University College London
`{plewis,sriedel}@fb.com`; `p.stenetorp@cs.ucl.ac.uk`

Abstract

Ideally Open-Domain Question Answering models should exhibit a number of competencies, ranging from simply memorizing questions seen at training time, to answering novel question formulations with answers seen during training, to generalizing to completely novel questions with novel answers. However, single aggregated test set scores do not show the full picture of what capabilities models truly have. In this work, we perform a detailed study of the test sets of three popular open-domain benchmark datasets with respect to these competencies. We find that 30% of test-set questions have a near-duplicate paraphrase in their corresponding train sets. In addition, we find that 60–70% of answers in the test sets are also present in the train sets. Using these findings, we evaluate a variety of popular open-domain models to obtain greater insight into what extent they can generalize, and what drives their overall performance. We find that all models perform substantially worse on questions that cannot be memorized from train sets, with a mean absolute performance difference of 61% between repeated and non-repeated data. Finally we show that simple nearest-neighbor models outperform a BART closed-book QA model, further highlighting the role that train set memorization plays in these benchmarks.

1 Introduction

Open-domain Question Answering (ODQA) is a task that examines the ability of models to produce answers to natural language factoid questions drawn from an open set of domains. ODQA has received significant attention for its potential practical applications, and more recently as a popular method to analyse how well NLP systems can capture and recall factual knowledge. This interest in ODQA as a challenging "knowledge-intensive" task has led to a flurry of recent works that have driven test-set performance on standard ODQA datasets to new heights (Lee et al., 2019; Guu et al., 2020; Karpukhin et al., 2020; Lewis et al., 2020; Izacard and Grave, 2020, inter alia). Whilst there have been several works examining other kinds of QA datasets (Manjunatha et al., 2018; Kaushik and Lipton, 2018; Sugawara et al., 2018, 2020), however, we know comparatively little about how the questions and answers are distributed in ODQA benchmarks, making it hard to understand and contextualize the results we are observing.

In this work, we address these issues via an analysis of the test sets of three popular ODQA datasets, namely WebQuestions (Berant et al., 2013), TriviaQA (Joshi et al., 2017) and Open NaturalQuestions (Kwiatkowski et al., 2019; Lee et al., 2019). We identify three types of desired behaviour of a trained ODQA system, in increasing order of difficulty: 1) to recall the answer to a question that the model has seen at training time. 2) to answer novel questions at test time and choose an answer from the set of answers it has seen during training. 3) to answer novel questions which have answers which are not contained in the training data. It is not clear to what extent our current ODQA datasets measure each of these three behaviours.

To address this, we stratify the test sets of these datasets. Firstly, we split the test data by whether answers in the test set also appear somewhere in the train set. We find that 58–71% of test answers also occur somewhere in the training data, demonstrating that the majority of the test data does not probe for answer generalization. Secondly, we annotate 1,000 question-answer pairs from each test set for near-duplicate questions in their respective train sets. We find that a surprisingly high 28–34% have paraphrased questions in the training data, the vast majority of which are near-duplicates differing by one or two words. This result implies that 30% of the test set of these datasets only probe for how

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1000–1008
April 19 - 23, 2021.

Dataset	% Answer overlap	% Question overlap
NaturalQuestions	63.6	32.5
TriviaQA	71.7	33.6
WebQuestions	57.9	27.5

Table 1: Test-train set overlap for the datasets.

well models can memorize question-answer pairs seen at training.

Equipped with these observations, we compute the performance of several recently proposed ODQA models on our test subsets. We test both *Open-book* approaches, which leverage retrieval from a large corpus of documents and *Closed-book* approaches, which focus on training large parametric seq2seq models with no external knowledge source (Roberts et al., 2020). We find that test data with train-overlapping data contribute the bulk of the overall performance of all the models studied.

These issues seem to be more acute for closed-book models. Strikingly, we find that a closed-book BART-based model (Lewis et al., 2019) is incapable of producing answers not observed at training time, and achieves very low scores on non-overlapping questions, suggesting this model is only capable of memorizing question-answer pairs from training time. With this in mind, we build simple nearest-neighbor models which outperform this BART model, despite having virtually no capacity to generalize beyond training data.

In summary, we make the following contributions: 1) We provide insights into how answer entities are distributed between dataset splits for ODQA datasets 2) We provide annotated subsets of ODQA test sets indicating whether test-time questions are duplicates of training time questions.[1] 3) We evaluate a variety of models on our dataset splits, and derive insights into what kinds of question answering behaviour different models achieve.

2 Datasets

We analyse three widely used ODQA datasets, WebQuestions (Berant et al., 2013), TriviaQA (Joshi et al., 2017), and Open-NaturalQuestions (Lee et al., 2019). All three datasets consist of factual natural language questions and short multi-token answers, differing slightly in the style of questions and format of answers.

WebQuestions consists of 3,778 train and 2,032 test instances. Questions were obtained by mining a search engine, and answers are Freebase entities (Bollacker et al., 2008) annotated by crowdworkers. The ODQA task consists of predicting the name of the Freebase entity. We use the standard train/test splits from Berant et al. (2013) and the development split from Karpukhin et al. (2020), which was randomly split from the train set.

TriviaQA consists of 78,785 train, 8,837 development and 11,313 test instances obtained by scraping trivia websites. Answers are Wikipedia entities, and any alias for the answer entity is considered a correct answer. We use the ODQA splits, which correspond to the unfiltered-train and unfiltered-dev reading comprehension splits (Lee et al., 2019; Min et al., 2019, 2020b; Karpukhin et al., 2020).

Open-NaturalQuestions consists of search engine questions with crowdsourced answer spans in Wikipedia articles. The ODQA version consists of question-answer pairs from NaturalQuestions which have short answer spans less than 6 tokens in length. We use the standard open-domain splits in our experiments, consisting of 79,168 train, 8,757 development and 3,610 question answer pairs.

For all three datasets, the canonical train, development and test splits were obtained by randomly splitting the question-answer pairs, and there are no exact duplicate questions in any dataset. We exclude development data from our overlap analyses, and focus purely on train-test overlap to explicitly assess the effects of training memorization.

3 Test-Train Overlaps

We explore two ways of examining the test sets based on overlaps between training and test data. Consider a question-answer pair (q, a) from the test set D_{test} where the answer consists of at least one answer reference $a = \{s_1..s_n\}$. We can consider *answer overlap* where there exists at least one $(q', a') \in D_{\text{train}}$ which shares at least one answer reference with (q, a). We can also consider *question overlap*, where there exists some $(q'', a'') \in D_{\text{train}}$ where q'' is a duplicate of q, such that q and q'' are paraphrases and have the same answer.

Answer Overlap Following Rajpurkar et al. (2016), we apply answer normalization (lower-casing, stripping punctuation, removing articles and normalizing whitespace) on answer references

---

[1] Our data and evaluation code is available at `https://github.com/facebookresearch/QA-Overlap`

Open NaturalQuestions		TriviaQA		WebQuestions	
Overlapping	Non-overlapping	Overlapping	Non-overlapping	Overlapping	Non-overlapping
Phil Simms	Cloves	David Bowie	Death in the afternoon	Harvard	Queen Victoria
Brian Johnson	Matt Monro	Battle of camlann	Clash of the Titans	Alderaan	Brasília
8	1,020 – 1,080 kg	Heligoland	ice-cream sundae	India	Paddington
the Indians	Hermann Ebbinghaus	Henry VII	Camshaft	2011	Tom Corbett
the 1830s	Matt Flinders	Niagra Falls	Cumberland	Zeus	Gary

Table 2: Randomly sampled overlapping and non-overlapping answers from all three test sets.

Answer	Test Question	Train Question
Jason Marsden	who plays max voice in a goofy movie	who does max voice in a goofy movie
January 23 2018	when will the 2018 oscar nominations be announced	when are the oscar nominations for 2018 announced
Alan Shearer	who has scored more goals in the premier league	most goals scored by a premier league player
retina	where are the cones in the eye located	where are cone cells located in the eye
francisco pizarro	who led the conquest of the incas in south america	conquistador who defeated the incan empire in peru

Table 3: Randomly sampled test-train overlapping questions in Open NaturalQuestions. See Appendix A.2 for more examples, including examples from TriviaQA and WebQuestions

before searching for overlapping answer references for all (q, a) pairs in the test set – see Table 1. We find that 57.9% of test (q, a) pairs in WebQuestions have answer overlaps, with 63.6% and 71.7% for NaturalQuestions and TriviaQA respectively. We would naturally expect TriviaQA to have higher answer overlap as it has more answer references per question on average (13.7 references on average compared to 1.2 for NaturalQuestions and 2.4 for WebQuestions). Examples of answer overlaps are shown in Table 2.

Question-Overlap Unlike answer overlap, question overlap cannot be easily computed automatically, as searching for duplicates via rules or paraphrase classifiers may lead to both false positives and negatives. Thus, we turn to manual annotation to investigate question overlap. To obtain a representative sample for each dataset, we annotate a random subset of 1,000 (q, a) pairs for each test set. Annotators are shown a list of up to 50 training questions which have a similar answer reference.[2] This answer similarity function is designed for high recall to obtain a tight lower bound on question overlap. If there were no questions with similar answers in the train set, the question was automatically annotated as not overlapping. Three expert annotators annotated the remaining questions and indicated if any were paraphrases of the test question and had the same answer.

The results from the annotation can be seen in Table 1 and examples of overlapping questions in Table 3. A sample of 100 2-way annotated examples indicated 93% agreement, corresponding to a Cohen's Kappa of 0.85 (Cohen, 1960). What we observe is a high degree of question overlap, with between 27.5% and 33.6% of the 1,000 annotated test questions having a duplicate in the train set. It is also common to see several duplicates per test question, with an average of 2.8 duplicate questions per overlapping test question in NaturalQuestions.

4 Implications for Modelling

Earlier we identified three classes of answering behaviours: 1) questions that can be memorized at training time, 2) novel questions that can be answered with answers memorized at training time, 3) novel questions with novel answers. We refer to these behaviours as *Question memorization*, *Answer classification* and *QA generalization*.

Question memorization To perform well on the question-overlap subset, a model only needs to memorize (q, a) pairs at training time, then recognize which training question matches a test-time question. The reasoning required ranges from trivial duplicate detection for very similar questions such as "who played pink in pink floyd the wall" and "who played pink in the movie the wall", to more challenging inference problems for subtler duplicates like "On which island in the North Sea did both St Aidan and St Cuthbert live?" and "irish born missionary st aidan founded a monastery in

---

[2] Training questions are selected for annotation if one of the following is true: they share an answer reference with a test question, a test answer reference is a sub-sequence of a training answer reference, or the other way around (a training reference answer is a sub-sequence of a test answer reference). If there are more than 50 such questions, the top 50 are chosen by the highest degree of word overlap with the test question.

Model		Open NaturalQuestions				TriviaQA				WebQuestions			
		Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
Open book	RAG	44.5	70.7	34.9	24.8	56.8	83.0	54.7	29.2	45.5	82.6	45.8	21.1
	DPR	41.3	69.4	34.6	19.3	57.9	78.0	59.6	31.6	42.4	76.1	39.8	22.2
	FID	51.4	71.3	48.3	34.5	67.6	86.2	66.9	42.8	-	-	-	-
Closed book	T5-11B+SSM	35.2	76.5	21.0	8.4	-	-	-	-	42.8	82.1	44.5	22.0
	BART	26.5	67.6	10.2	0.8	26.7	67.3	16.3	0.8	27.4	76.1	20.7	1.6
Nearest Neighbor	Dense	26.7	69.4	7.0	0.0	28.9	81.8	11.2	0.0	26.4	81.9	17.1	0.0
	TF-IDF	22.2	56.8	4.1	0.0	23.5	69.8	5.1	0.0	19.4	68.1	8.7	0.0

Table 4: Exact Match scores on our dataset splits. "Total" is the overall performance on the dataset. "Question Overlap" is the subset with train-test question overlap, and probes for simple question memorization. "Answer Overlap Only" is subset without train-test question overlap, but with train-test answer overlap, which probes for answer classification. "No overlap" is the subset with no train-test answer overlap and probes for QA generalization

653 on which english island which is also the name of a 1970s uk folk-rock band?". Manual annotation of 100 question-overlap pairs showed 81% were simple duplicates differing by one or two words, 14% needed some paraphrase recognition ability, and 5% needed more sophisticated language understanding. To measure performance on question memorization, we build a test subset of (q, a) pairs with question overlap to the train set.

Answer Classification For a model to handle answer-overlap questions, a classifier over train set answers would be sufficient, as answers never appear at test time that do not appear at training time. We build a test subset of (q, a) pairs which have answer overlap, but not question overlap. Question-overlap pairs are excluded as they are significantly easier and would inflate scores.

QA Generalization Here, models cannot rely on memorizing their training data. To measure performance on this most challenging split, we build a test subset of (q, a) pairs which do not have answer overlap with the train set. We also note that we expect higher frequency answers, such as countries, integers and public figures will naturally appear less often in this subset. As such, models that perform well on the head of the answer distribution may struggle to perform well here, despite being able to perform some generalization at test time.

Next we briefly describe the models included in our analysis. For published models, we obtain test set predictions directly from the authors.

4.1 Open-Book Models

Open-book Models first retrieve relevant documents from Wikipedia and then either extract or generate answers conditioned on those documents. We consider Dense Passage Retriever (DPR) (Karpukhin et al., 2020), a pipeline model which retrieves documents using dense embeddings, before feeding them into a conventional reader-reranker which extracts spans of text as answers. We also include Retrieval-Augmented Generation (Lewis et al., 2020), a model that jointly learns to retrieve and generate answers in a seq2seq framework. Finally we include Fusion-in-Decoder (FID) (Izacard and Grave, 2020), a pipeline model which retrieves 100 documents and fuses them so that the decoder can attend to all documents at once. We do not include FID results on WebQuestions as the authors did not use it in their original work.

4.2 Closed-Book Models

Closed-book models store the knowledge required to answer their questions entirely within the model's parameters, rather than in an external corpus. Typically these models consist of seq2seq transformers directly fine-tuned on (q, a) pairs. In our analysis, we train a BART-large closed-book model, which is trained with questions as input and generates (q, a) pairs as output. Checkpoints are selected by Exact Match score on a development set. We also include a more powerful T5-11B model from (Roberts et al., 2020). We use the T5-11B model pretrained with a special "Salient Span Masking" objective (Guu et al., 2020), which improves downstream ODQA performance. We do not include TriviaQA results for T5 since their model used a different data splitting scheme.

4.3 Nearest-Neighbor Models

Given the high levels of train-test overlaps in these datasets, we also experiment with some simple nearest-neighbor models. Here, we simply retrieve a (q, a) pair from the train set based on question similarity to the test question, and return its answer. We experiment with two models, one using TF-IDF and the other using maximum inner product of question embeddings from the DPR retriever. These models cannot generalize to non-overlapping answers, and have limited capacity to answer non-overlapping questions. However, they are attractive from the perspective of model size and efficiency (Min et al., 2020a).

4.4 Results

Question Memorization Earlier, we found that $\sim$30% of test set questions overlap with the train set. The "Question overlap" columns in Table 4 shows performance on Question Memorization. Comparing this column with the total performance column shows that all models perform significantly higher on memorizable questions. This effect is most pronounced for closed book models. The T5-11B performs especially well for question memorization on both NaturalQuestions and WebQuestions. This suggests that its very large capacity, coupled with more powerful question understanding may allow it to store and recall training questions more effectively than other models.

Answer Classification The "Answer overlap only" column in Table 4 shows performance on answer classification. Answer classification has a large drop in performance compared to question memorization, dropping by an average of 45% Exact Match. Open-book models handle this setting better than closed book models. The BART model in particular struggles here, scoring only 10.2%.

QA Generalization The "No overlap" column in Table 4 shows performance on QA generalization. All models significantly lose performance on QA generalization, highlighting the shortcomings of the total performance metric. For example, we may expect the FID state-of-the-art model to answer half of NaturalQuestions-style questions correctly, but once we account for repeated questions and answers, it can only answer about a third of questions correctly. This difference is even more pronounced for other models, with an average absolute drop of 25% with respect to total performance.

Nearest-Neighbor Models The bottom two rows of Table 4 show results for nearest-neighbor models. TF-IDF, despite being completely untrained, can answer about 20% of test questions correctly, purely by retrieving questions from the train sets. The dense retrieval model outperforms the BART closed-book model on NaturalQuestions and TriviaQA. Further, the dense nearest neighbor model also outperforms the significantly more complex DPR open-book model on TriviaQA and WebQuestions on the question overlap subset.

5 Related Work

Examining what behaviours are learnt by models has received attention in language understanding tasks, such as GLUE (Wang et al., 2018), which includes tools for probing for different reasoning types. There has also been critical and careful analysis of QA systems and datasets. Chen et al. (2016), Sugawara et al. (2020) and Kaushik and Lipton (2018) analyse the difficulty of various machine reading datasets, and Manjunatha et al. (2018) show that visual QA models memorize common question-answer relationships in training data. Févry et al. (2020) analyse various closed-book models' TriviaQA predictions. Kwiatkowski et al. (2019) note that the machine reading NaturalQuestions dataset has train-test overlap of Wikipedia titles, and provide baselines for "long-answer" QA. Verga et al. (2020) observe answer overlap effects in a related modality (knowledgebase QA), but no not consider question overlap.

6 Conclusion

We performed an analysis of popular open-domain QA datasets. We found that 60% of test set answers overlap with the train set and 30% of test set questions have at least one duplicate in the train set. Given these observations, we contextualize performance of seven ODQA models, stratifying by the extent of overlap, exploring how well these models generalize verses simply memorizing training data. It is clear that performance on these datasets cannot be properly understood by overall QA accuracy. In the future, a greater emphasis should be placed on more behaviour-driven evaluation, rather than pursuing single-number overall accuracy.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 1247–1250, Vancouver, Canada. Association for Computing Machinery.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46. Publisher: SAGE Publications Inc.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as Experts: Sparse Memory Access with Entity Supervision. *arXiv:2004.07202 [cs]*. ArXiv: 2004.07202.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. *arXiv:2002.08909 [cs]*. ArXiv: 2002.08909.

Gautier Izacard and Edouard Grave. 2020. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. *arXiv:2007.01282 [cs]*. ArXiv: 2007.01282.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv:2004.04906 [cs]*. ArXiv: 2004.04906.

Divyansh Kaushik and Zachary C. Lipton. 2018. How Much Reading Does Reading Comprehension Require? A Critical Investigation of Popular Benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: a Benchmark for Question Answering Research. *Transactions of the Association of Computational Linguistics*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *arXiv:1910.13461 [cs, stat]*. ArXiv: 1910.13461.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv:2005.11401 [cs]*. ArXiv: 2005.11401.

Varun Manjunatha, Nirat Saini, and Larry S. Davis. 2018. Explicit Bias Discovery in Visual Question Answering Models. *arXiv:1811.07789 [cs]*. ArXiv: 1811.07789.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, Patrick Lewis, Yuxiang Wu, Heinrich Küttler, Linqing Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei Miyawaki, Shun Sato, Ryo Takahashi, Jun Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, Pavel Smrz, Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-tau Yih. 2020a. NeurIPS 2020 EfficientQA Competition: Systems, Analyses and Lessons Learned. *arXiv:2101.00133 [cs]*. ArXiv: 2101.00133.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A Discrete Hard EM Approach for Weakly Supervised Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2020b. Knowledge Guided Text Retrieval and Reading for Open Domain Question Answering. *arXiv:1911.03868 [cs]*. ArXiv: 1911.03868.

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

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How Much Knowledge Can You Pack Into the Parameters of a Language Model? *arXiv:2002.08910 [cs, stat]*. ArXiv: 2002.08910.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What Makes Reading Comprehension Questions Easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Brussels, Belgium. Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the Benchmarking Capacity of Machine Reading Comprehension Datasets. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8918–8927. AAAI Press.

Pat Verga, Haitian Sun, Livio Baldini Soares, and William W. Cohen. 2020. Facts as Experts: Adaptable and Interpretable Neural Memory over Symbolic Knowledge. *arXiv:2007.00849 [cs]*. ArXiv: 2007.00849.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

A Appendices for Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets

A.1 Test subset sizes

Table 5 gives details of how many instances are in each test set, as well as the number of instances in each test subset used in the main paper.

A.2 Additional Question Overlap Examples

Tables 6, 7 and 8 give more question overlap examples for the three datasets.

Dataset	Total	Answer Overlap	N Annotated	Question Overlap	Answer Overlap Only
NaturalQuestions	3610	2297	996	324	315
TriviaQA	11313	8112	1001	336	411
WebQuestions	2032	1176	998	274	299

Table 5: Number of instances in each test set subset

Answer	Test Question	Train Question
Bob Geldof	who played pink in pink floyd the wall	who played pink in the movie the wall
Daren Maxwell Kagasoff	who played ricky in secret life of the american teenager	who played ricky on the secret life of the american teenager
Andy	who does april end up with on parks and rec	who does april marry in parks and rec
may 5 2017	when did gaurdians of the galaxy 2 come out	when is guardians of the galaxy vol 2 released
norman pritchard	who won the first medal in olympics for india	who won the first individual olympic medal for india
moira kelly	who does the voice of nala in the lion king	who played nala in the lion king movie
supreme court	who enforces the charter of rights and freedoms	who has final authority of interpretation of the canadian charter of rights and freedoms
554	most passing yards by nfl qb in a game	what is the nfl record for most passing yards in a single game
John Ross	who ran the fastest 40 yard dash in the nfl	who has the fastest 40 yard dash ever
international border ib	what is the name of india pakistan border	what is the border name between india and pakistan
Andrew Wright	who wrote when a man loves a woman	who wrote song when a man loves a woman
new england patriots	who has participated in the most super bowls	what nfl team has been to most super bowls

Table 6: Additional examples of test-train overlapping questions in Open NaturalQuestions

Answer	Test Question	Train Question
Picasso	Who painted ”Boy With a Pipe” which, in May 2004, was sold for a record price of $104 million?	painted in 1905, the painting garcon a la pipe was a famous painting by which famous artist who died in 1973?
Wensum	On what river is the city of Norwich	the english city of norwich lies on which river?
Mantle	Comprising around two-thirds of the Earth’s mass , what is found between the core of the Earth and its crust?	what do we call the layer of the earth between its crust and its core?
Live and Let Die	In which James Bond film does actress Jane Seymour play Solitaire?	jane seymour played the character ”solitaire” in which bond film?
Esau	Who, in the Bible, was the eldest son of Isaac?	in the bible, who was the first born of isaac?
Alanis Morrisette	Who made the 1995 album ’Jagged Little Pill’ which sold 33 million copies?	who released the 1995 hit album ”jagged little pill”?
Excalibur	In British legend, what is the name of King Arthur’s sword?	what was the name of king arthur’s sword?
Humidity	What is measured by a Hygrometer?	what does a hygrometer measure?
A Storm	On the Beaufort scale what is defined as force 11?	what is force 11 (eleven) on the beaufort scale?
Jeremy Irons	Actress Sinead Cusack is married to which ’Oscar’ winning actor?	which actor is the husband of sinead cusack?
Sir Cloudesley Shovell	Who was the British Admiral who died in 1707 when four of his ships were wrecked in the Scilly Isles?	in 1707 a fleet of navy ships was wrecked off the scilly islands. who was the commander who lost his life in the disaster?
Tony Hart	Which famous individual created the ’Blue Peter’ sailing ship logo?	which artist designed the logo for uk television children’s show ‘blue peter’?

Table 7: Examples of test-train overlapping questions in TriviaQA

Answer	Test Question	Train Question
costa rica	where is isthmus of panama located on the map?	where is isthmus of panama located?
1986 world series	when’s the last time the mets won the world series?	when did the mets win the pennant?
abbottabad	where was bin laden found and killed?	what country was osama bin laden killed in?
believer	what other movies has ryan gosling been in?	what movies does ryan gosling star in?
sculpture	what type of art did leonardo da vinci make?	what kind of art did leonardo da vinci produce?
origin of species	what book did charles darwin wrote in 1859?	what was the name of the book that charles darwin wrote?
morehouse college	what college did martin luther king jr go to?	where did dr. martin luther king jr. go to school?
communist state	what type of government did soviet union have?	what type of government does the former soviet union have?
turkish lira	what money to take to turkey?	what currency to take to side turkey?
spanish language	what is the most common language spoken in argentina?	what is language in argentina?
opera OR classical music	what music period did beethoven live in?	what music did beethoven composed?
harry s truman	who was president after franklin d. roosevelt?	who became president when roosevelt died in office?

Table 8: Examples of test-train overlapping questions in WebQuestions

El Volumen Louder Por Favor: Code-switching in Task-oriented Semantic Parsing

Arash Einolghozati **Abhinav Arora** **Lorena Sainz-Maza Lecanda**
Anuj Kumar **Sonal Gupta**
Facebook
{arashe,abhinavarora,lorenasml,anujk,sonalgupta}@fb.com

Abstract

Being able to parse code-switched (CS) utterances, such as Spanish+English or Hindi+English, is essential to democratize task-oriented semantic parsing systems for certain locales. In this work, we focus on Spanglish (Spanish+English) and release a dataset, CSTOP, containing 5800 CS utterances alongside their semantic parses. We examine the CS generalizability of various Cross-lingual (XL) models and exhibit the advantage of pre-trained XL language models when data for only one language is present. As such, we focus on improving the pre-trained models for the case when only English corpus alongside either zero or a few CS training instances are available. We propose two data augmentation methods for the zero-shot and the few-shot settings: fine-tune using translate-and-align and augment using a generation model followed by match-and-filter. Combining the few-shot setting with the above improvements decreases the initial 30-point accuracy gap between the zero-shot and the full-data settings by two thirds.

1 Introduction

Code-switching (CS) is the alternation of languages within an utterance or a conversation (Poplack, 2004). It occurs under certain linguistic constraints but can vary from one locale to another (Joshi, 1982). We envision two usages of CS for virtual assistants. First, CS is very common in locales where there is a heavy influence of a foreign language (usually English) in the native "substrate" language (e.g., Hindi or Latin-American Spanish). Second, for other native languages, the prevalence of English-related tech words (e.g., Internet, screen) or media vocabulary (e.g., movie names) is very common. While in the second case, a model using contextual understanding should be able to parse the utterance, the first form of CS, which is our focus in this paper, needs Cross-Lingual(XL) capabilities in order to infer the meaning.

There are various challenges for CS semantic parsing. First, collecting CS data is hard because it needs bilingual annotators. This gets even worse considering that the number of CS pairs grows quadratically. Moreover, CS is very dynamic and changes significantly by occasion and in time (Poplack, 2004). As such, we need extensible solutions that need little or no CS data while having the more commonly-accessible English data available. In this paper, we first focus on the zero-shot setup for which we only use EN data for the same task domains (we call this in-domain EN data). We show that by translating the utterances to ES and aligning the slot values, we can achieve high accuracy on the CS data. Moreover, we show that having a limited number of CS data alongside augmentation with synthetically generated data can significantly improve the performance.

Our contributions are as follows: 1) We release a code-switched task-oriented dialog data set, CSTOP[1], containing 5800 Spanglish utterances and a corresponding parsing task. To the best of our knowledge, this is the first Code-switched parsing dataset of such size that contains utterances for both training and testing. 2) We evaluate strong baselines under various resource constraints. 3) We introduce two data augmentation techniques that improve the code-switching performance using monolingual data.

2 Task

In task-oriented dialog, the language understanding task consists of classifying the intent of an utterance, i.e., sentence classification, alongside tagging the slots, i.e., sequence labeling. We use the Task-

---

[1] The dataset can be downloaded from https://fb.me/cstop_data

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1009–1021
April 19 - 23, 2021.

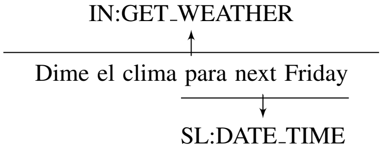


Figure 1: Example CS sentence and its annotation for the sequence [IN:GET_WEATHER Dime el clima [SL:DATE_TIME para next Friday]]

Oriented Parsing dataset released by Schuster et al. (2018) as our EN monolingual dataset. We release a similar dataset, CSTOP, of around 5800 Spanglish utterances over two domains, Weather and Device, which are collected and annotated by native Spanglish speakers. An example from the CSTOP alongside its annotation is shown in Fig. 1. Note that the intent and slot lables start with IN : and SL :, respectively. Our task is to classify the sentence intent, here *IN:GET_WEATHER* as well as the label and value of the slots, here *SL:DATE_TIME* corresponding to the span *para next Friday*. Moreover, other words are classified as having no label, i.e., *O* class. We discuss the details of this dataset in the next section.

One of the unique challenges of this task, compared with common NER and language identification CS tasks, is the constant evolution of CS data. Since the task is concerned with spoken language, the nature of CS is very dynamic and keeps evolving from domain to domain and from one community to another. Furthermore, cross-lingual data for this task is also very rare. Most of the existing techniques, either combine monolingual representations (Winata et al., 2019a) or combine the datasets to synthesize code-switched data (Liu et al., 2019). Lack of monolingual data for the substrate language (very realistic if you replace ES with a less common language) would make those techniques inapplicable.

In order to evaluate the model in a task-oriented dialog setting, we use the exact-match accuracy (from now on, accuracy) as the primary metric. This is simply defined as the percentage of utterances for which the full parse, i.e., the intent and all the slots, have been correctly predicted.

3 CSTOP Dataset

In this section, we provide details of the CSTOP dataset. We originally collected around 5800 CS utterances over two domains; *Weather* and *Device*. We picked these two domains as they represent complementary behavior. While Weather contains slot-heavy utterances (average 1.6 slots per utterance), Device is an intent-heavy domain with only average 0.8 slots per utterance. We split the data into 4077, 1167, and 559 utterances for training, testing, and validation, respectively.

CS data collection proceeded in the following steps:

1. One of the authors, who is a native speaker of Spanish and uses Spanglish on a daily basis, generated a small set of CS utterances for Weather and Device domains. Additionally, we also recruited bilingual EN/ES speakers who met our Spanglish speaker criteria guidelines, established following Escobar and Potowski (2015).

2. We wrote Spanglish data creation instructions and asked participants to produce Spanish-English CS utterances for each intent (i.e. ask for the weather, set device brightness, etc).

3. Next, we filter out utterances from this pool to only retain those that exhibited true intra-sentential CS.

4. The collected utterances were labeled by two annotators, who identified the intent and slot spans. If the two annotators disagreed on the annotation for an utterance, a third annotator would resolve the disagreement to provide a final annotation for it.

Table. 1 shows the number of distinct intents and slots for each domain and the number of utterances in CSTOP for each domain. We have also shown the most 15 common intents in the training set and a representative Spanglish example alongside its slot values for those intents in Table. 2. The first value in a slot tuple is the slot label and the second is the slot value. We can see that while most of the verbs and stop words are in Spanish, Nouns and slot values are mostly in English. We further calculate the prevalence Spanish and English words by using a vocabulary file of 20k for each language. Each token in the CSTOP training set is assigned to the language for which that token has a lower rank. The ratio of the Spangish to English tokens is around 1.34 which matches our previous anecdotal observation. This ratio was consistent when increasing the vocabular size to even 40k. .

Domain	intents	slots	utterances
Weather	2	4	3692
Device	17	6	2112

Table 1: CSTOP Statistics

4 Model

Our base model is a bidirectional LSTM with separate projections for the intent and slot tagging (Yao et al., 2013). We use the aligned word embedding MUSE (Conneau et al., 2017) with a vocabulary size of 25k for both EN and ES. Our experiments showed that for the best XL generalization, it's best to freeze the word embeddings when the training data contains only EN or ES utterances. We refer to this model as simply MUSE.

We also use SOTA pre-trained XL models; XLM (Conneau and Lample, 2019) and XLM-R (Conneau et al., 2019). These models are pre-trained via Masked Language Modeling (MLM) (Devlin et al., 2019) on massive multilingual data. They share the word-piece token representation, BPE (Sennrich et al., 2016) and SentencePiece (Kudo and Richardson, 2018), as well as a common MLM transformer for different languages. Moreover, while XLM is pre-trained on Wikipedia, XLM-R is trained on crawled web data which contains more non-English and possibly CS data. In order to adapt these models for the joint intent classification and slot tagging task, we use the method described in Chen et al. (2019). For classification, we add a linear classifier on top of the first hidden state of the Transformer. A typical slot tagging model feeds the hidden states, corresponding to each token, to a CRF layer (Mesnil et al., 2015). To make this compatible with XLM and XLM-R, we use the hidden states corresponding to the first sub-word of every token as the input to the CRF layer.

Table 3 shows the accuracy of the above models on CSTOP. We also have listed the performance when the models were first fine-tuned on the EN data (CS+EN). We observe that in-domain fine-tuning can almost halve the gap between XLM-R and XLM, which is around 50% faster during the inference than XLM-R during inference. The training details for all our models and the validation results are listed in the Appendix.

5 Zero-shot performance

Bottom part of Table 3 shows the CS test accuracy when using only the in-domain monolingual data. Our EN dataset is the task-oriented parsing dataset (Schuster et al., 2018) described in the previous section. Since the original TOP dataset did not include any utterances belonging to the Device domain, we also release a dataset of around thousand EN Device utterances for the experiments using the EN data. In order to showcase the effect of monolingual ES data, we also experiment with using the in-domain ES dataset, i.e. ES Weather and Device queries.

We observe that having monolingual data of both languages yields very high accuracy, only a few points shy of training directly on the CS data. Moreover, in this setting, even simpler models such as MUSE can yield competitive results with XLM-R while being much faster. However, the advantage of XL pre-training becomes evident when only one of the languages is present. As such, having only the substrate language (i.e., ES) is almost the same as having both languages for XLM-R.

Note that we do not use ES data for other results in this paper. Obtaining semantic parsing dataset in another language is expensive and often only EN data is available. Our experiments show a huge performance gap when only using the EN data, and thus in this paper, we will be focusing on using the EN data alongside zero or a few CS instances.

5.1 Effect of XL Embeddings

Here, we explore how much of the zero-shot performance can be attributed to the XL embeddings as opposed to the shared XL representation. As such, we experiment with replacing MUSE embeddings with other embeddings in the LSTM model explained in the previous section. We experiment with the following strategies:: (1) Random embedding: This learns the ES and EN word embeddings from the scratch (2) Randomly-initialized SentencePiece (Kudo and Richardson, 2018) (RSP): Words are represented by wordpiece tokens that are learned from a huge unlabeled multilingual corpus. (3) Pre-trained XLM-R sentence piece (XLSP). These are the 250k embedding vectors that are learned during the pre-trainig of XLM-R.

We have shown the effects of using the aforementioned embeddings in the zero-shot setting in Table 4. We can see that by having monolingual datasets from both languages, even random

intent	utterance	slots
GET_WEATHER	¿cómo estará el clima en Miami este weekend?	(LOCATION, Miami), (DATE_TIME, este weekend)
UNSUPPORTED_WEATHER	how many centimeters va a llover hoy	(DATE_TIME, hoy)
OPEN_RESOURCE	Abreme el gallery	(RESOURCE, el gallery)
CLOSE_RESOURCE	Cierra maps	(RESOURCE, maps)
TURN_ON	Prende el privacy mode	(COMPONENT, el privacy mode)
TURN_OFF	Desactiva el speaker	(COMPONENT, el speaker)
WAKE_UP	Quita sleep mode	-
SLEEP	prende el modo sleep	-
OPEN_HOMESCREEN	Go to pagina de inicio	-
MUTE_VOLUME	Desactiva el sound	-
UNMUTE_VOLUME	Prende el sound	-
SET_BRIGHTNESS	subir el brigtness al 80	(PERCENT, 80)
INCREASE_BRIGHTNESS	Ponlo mas bright	-
DECREASE_BRIGHTNESS	baja el brightness	-
SET_VOLUME	Turn the volumen al nivel 10	(PRECISE_AMOUNT,10)
INCREASE_VOLUME	aumenta el volumen a little bit	-
DECREASE_VOLUME	Bájale a la music	-

Table 2: Examples from CSTOP intents

Lang/Model	MUSE	XLM	XLM-R
CS	87.0	86.6	94.4
CS + EN	88.1	93.0	95.4
EN	39.2	54.8	66.6
ES	69.9	78.3	88.1
EN+ES	88.2	87.8	91.2

Table 3: Full-training (top) and zero-shot (bottom) accuracy of XL models when using different monolingual corpora. ES is an internal dataset to showcase the effect of having a big Spanish corpus.

embeddings can yield high performance. By removing one of the languages, unsurprisingly, the codeswitching generalizability drops sharply for all, but much less for XLSP and MUSE. Moreover, even though the XLSP embeddingsm, unlike MUSE, is not consttrained to only EN and ES, it yields comparable results with the word-based MUSE embeddings.

We can also see that When ES data is available, RSP provides some codeswitching generalizability, as compared with the Random strategy, but not when only EN data is available. We hypothesize that the common sub-word tokens are more helpful to generalize the slot values (which in the codeswitched data are mostly in EN) than the non-slot queries which are more commonly in ES. This is also verified by the observation that most of the gains for the RSP vs Random for the ES only scenario come from the slot tagging accuracy as compared with the intent detection.

As a final note, we observe that between $20-30\%$ of the XLM-R gains can be captured by using the pre-trained sentence-piece embeddings while the rest are coming from the shared XL representation pre-trained on massive unlabeled data. In the rest of the paper, we focus on the XLM-R model.

	Random	RSP	XLSP	MUSE
EN	13.5	12.2	30.3	**39.2**
ES	38.2	48.0	**70.5**	**69.9**
EN+ES	81.1	84.3	**89.0**	88.2

Table 4: Zero-shot accuracy for simple LSTM model when using different monolingual corpora and different embedding strategies.

6 Data Augmentation Approaches

In this section, we discuss two data augmentation approaches. The first one is in a zero-shot setting and only uses EN data to improve the performance on the Spanglish test set. In the second approach, we assume having a limited number of Spanglish data and use the EN data to augment the few-shot setting.

6.1 Translate and Align

We explore creating synthetic ES data from the EN dataset using machine translation. Since our task is a joint intent and slot tagging task, creating a synthetic ES corpus consists of two parts: a) Obtaining a parallel EN-ES corpus by machine translating utterances from EN to ES, b) Projecting gold annotations from EN utterances to their ES counterparts via word alignment (Tiedemann et al., 2014; Lee et al., 2019b). Once the words in both languages are aligned, the slot annotations are simply copied

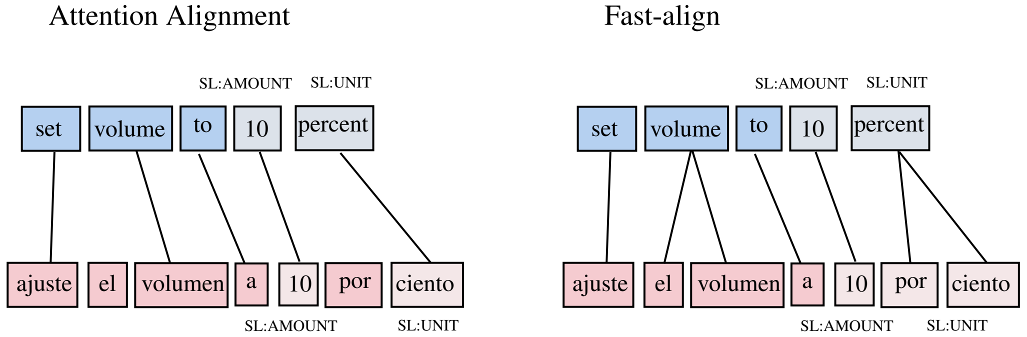


Figure 2: An example comparison between the two methods of slot label projection. The image in on the left shows Attention alignment, where every source token gets projected to a single target token. As a result, *percent*, in EN is aligned only with *ciento* in ES. The image on the right shows fast-align, which allow a many-to-many alignment. Hence *percent* is correctly aligned with *por ciento*.

over from EN to ES by word alignment. For word alignment, we explore two methods that are explained below. In some cases, word alignment may produce discontinuous slot tokens in ES, which we handle by introducing new slots of the same type, for all discontinuous slot fragments.

Our first method leverages the attention scores (Bahdanau et al., 2015) obtained from an existing EN to ES NMT model. We adopt a simplifying assumption that each source word is aligned to one target language word (Brown et al., 1993). For every slot token in the source language, we select the highest attention score to align it with a word in the target language.

Our next approach to annotation projection makes use of unsupervised word alignment from statistical machine translation. Specifically, we use the fast-align toolkit (Dyer et al., 2013) to obtain alignments between EN and ES tokens. Since fast-align generates asymmetric alignments, we generate two sets of alignments, EN to ES and ES to EN and symmetrize them using the grow-diagnol-final-and heuristic (Koehn et al., 2003) to obtain the final alignments.

In Table 5, we show the CS zero-shot accuracy when fine-tuning on the newly generated ES data (called ES*.) alongside the original EN data. We can see that unsupervised alignment results in around 2.5 absolute point accuracy improvement. On the other hand, using attention alignment ends up hurting the accuracy, which is perhaps due to the slot noise that it introduces. The assumption that a single source token aligns with a single target token leads to incorrect data annotations when the length of a translated slot is different in EN and ES. Figure 3 shows an example utterance where attention alignment produces an incorrect annotation compared to unsupervised alignment.

EN	**EN+ES* Attn**	**EN+ES* aligned**
66.6	65.8	69.2

Table 5: Zero-shot accuracy when fine-tuning XLM-R on EN monolignual data as well as the auto-translated and aligned ES data (called ES*).

6.2 Generate by Match-and-Filter in the Few-shot Setting

Here, we assume having a limited number of high-quality in-domain CS data and as such, we construct the CSTOP$_{100}$ dataset of around 100 utterances from the original training set in the CSTOP. We make sure that every individual slot and intent (but not necessarily the combination) is presented in CSTOP$_{100}$ and randomly sample the rest. We perform our sampling three times and report the few-shot results on the average performance. This setting is of paramount importance for bringing up a domain in a new locale when the EN data is already available. The first column in Table 6 shows the CS Few-Shot (FS) performance alongside the fine-tuning on the EN data and the aligned translated data, when average over three sampling of CSTOP$_{100}$.

In order to improve the FS performance, we perform data augmentation on the CSTOP$_{100}$ dataset. Unlike methods such as Pratapa et al. (2018), we seek generic methods that do not need extra resources such as constituency parsers. Instead, we explore using pre-trained generative models while taking advantage of the EN data.

We use BART (Lewis et al., 2019), a denois-

Model/Training Data	Few Shot	Few shot+ Generate and Filter augmentation
XLM-R	61.2	70.3
XLM-R fine-tuned on EN	82.6	83.7
XLM-R fine-tuned on EN+ES*	84.1	**84.8**

Table 6: Accuracy when only a few CS instances are available during training, with and without the data augmentation. ES* is the auto-translated and aligned data.

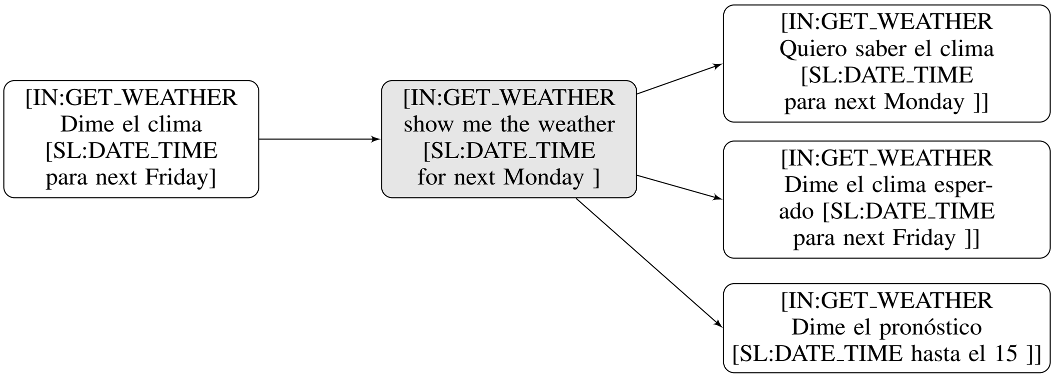


Figure 3: Match and Filter data augmentation: 1- For each CS utterance (target), find the the closest EN neighbor (source). 2- Learn a generative model from source to target 3- Perform beam search to generate more targets from the source utterances.

ing autoencoder trained on massive amount of web data, as the generative model. Our goal is to generate diverse Spanglish data from the EN data. Even though BART was trained for English, we found it very effective for this task. We hypothesize this is due to the abundance of the Spanish text among EN web data and the proximity of the word-piece tokens among them. We also experimented with multilingual BART (Liu et al., 2020a) but found it very challenging to fine-tune for this task.

First, we convert the data to a bracket format (Vinyals et al., 2015), which is called the seqlogical form in Gupta et al. (2018). Examples of this format are shown in Fig. 3. In the seqlogical form, we include the intent (i.e., sentence label) at the beginning and for each slot, we first include the label and text in brackets.

We perform our data augmentation technique in the following steps:

1. Find the top K closest EN neighbors to every CS query in the CSTOP$_{100}$. We enforce the neighbors to have the same parse as the CS utterance, i.e., same intent and same slot labels, and use the Levenshtein distance to rank the EN sequences.

2. Having this parallel corpus, i.e., top-K EN neighbors as the source and the original CS query as the target, Fine-tune the BART model. We use K=10 in our experiments to increase the parallel data size to around 650.

3. During the inference, Use the beam size of 5 to decode CS utterances from the same EN source data. Since both the source and target sequences are in the seqlogical form, the CS generated sequences are already annotated.

In Fig. 3, we have shown the closest EN neighbor corresponding to the original CS example in Fig. 1. The CS utterance can be seen as a rough translation of the EN sentence. We have also shown the top three generated CS utterances from the EN example.

In order to reduces the noise, we filter the generated sequences that either already exist in CSTOP$_{100}$, are not valid trees, or have a semantic parse different from the original utterance. We augment CSTOP$_{100}$ with the data, and fine-tune the XLM-R baseline.

In the second column of Table 6, we have shown the average data augmentation improvement over the three CSTOP$_{100}$ samples for the few-shot setting. We can see that even after fine-tuning on the EN monolingual data (the second row), the augmentation technique improves this strong baseline. In the last row, we first use the translation alignment of the previous section to obtain ES*. After fine-tuning on this set combined with the EN data, we further fine-tune on the CSTOP$_{100}$. We can see that the best model enjoys improvements from both zero-shot (translation alignment) and the few-shot (generate and filter) augmentation techniques. We also note that the p-value corresponding to the second and third row gains are 0.018 and 0.055, respectively.

7 Related Work

7.1 XL Pre-training

Most of the initial work on pre-trained XL representations was focused on embedding alignment (Xing et al., 2015; Zhang et al., 2017; Conneau et al., 2018). Recent developments in this area have focused on the context-aware XL alignment of contextual representations (Schuster et al., 2019; Aldarmaki and Diab, 2019; Wang et al., 2019; Cao et al., 2020). Recently, pre-trained multilingual language models such as mBERT (Devlin et al., 2019), XLM (Conneau and Lample, 2019), and Conneau et al. (2019) have been introduced, and Pires et al. (2019) demonstrate the effectiveness of these on sequence labeling tasks.

Separately, Liu et al. (2020a) introduce mBART, a sequence-to-sequence denoising auto-encoder pre-trained on monolingual corpora in many languages using a denoising autoencoder objective (Lewis et al., 2019).

7.2 Code-Switching

Following the ACL shared tasks, CS is mostly discussed in the context of word-level language identification (Molina et al., 2016) and NER (Aguilar et al., 2018). Techniques such as curriculum learning (Choudhury et al., 2017) and attention over different embeddings (Wang et al., 2018; Winata et al., 2019a) have been among the successful techniques. CS parsing and use of monolingual parses are discussed in Sharma et al. (2016); Bhat et al. (2017, 2018). Sharma et al. (2016) introduces a Hinglish test set for a shallow parsing pipeline. In Bhat et al. (2017), outputs of two monolingual dependency parsers are combined to achieve a CS parse. Bhat et al. (2018) extends this test set by including training data and transfers the knowledge from monolingual treebanks. Duong et al. (2017) introduced a CS test set for semantic parsing which is curated by combining utterances from the two monolingual datasets. In contrast, CSTOP is procured independently of the monolingual data and exhibits much more linguistic diversity. In Pratapa et al. (2018), linguistic rules are used to generate CS data which has been shown to be effective in reducing the perplexity of a CS language model. In contrast, our augmentation techniques are generic and do not require rules or constituency parsers.

7.3 XL Data Augmentation

Most approaches to cross-lingual data augmentation use machine translation and slot projection for sequence labeling tasks (Jain et al., 2019). Wei and Zou (2019) uses simple operations such as synonym replacement and Lee et al. (2019a) use phrase replacement from a parallel corpus to augment the training data. Singh et al. (2019) present XLDA that augments data by replacing segments of input text with its translations in other languages. Some recent approaches (Chang et al., 2019; Winata et al., 2019b) also train generative models to artificially generate CS data. More recently, Kumar et al. (2020) study data augmentation using pre-trained transformer models by incorporating label information during fine-tuning. Concurrent to our work, Bari et al. (2020) introduce Multimix, where data augmentation from pre-trained multilingual language models and self-learning are used for semi-supervised learning. Recently, Liu et al. (2019) generate CS data by translating keywords picked based on attention scores from a monolingual model. Generating CS data has recently been studied in Liu et al. (2020b)

7.4 Task-oriented Dialog

The intent/slot framework is the most common way of performing language understanding for task oriented dialog using. Bidirectional LSTM for the sentence representation alongside separate projection layers for intent and slot tagging is the typical architecture for the joint task (Yao et al., 2013; Mesnil et al., 2015; Hakkani-Tur et al., 2016). Such representations can accommodate trees of up to length two, as is the case in CSTOP. More recently, an extension of this framework has been introduced to fit the deeper trees (Gupta et al., 2018; Rongali et al., 2020).

8 Conclusion

In this paper, we propose a new task for code-switched semantic parsing and release a dataset, CSTOP, containing 5800 Spanglish utterances over two domains. We hope this foments further research on the code-switching phenomenon which has been set back by paucity of sizeable curated datasets. We show that cross-lingual pre-trained models can generalize better than traditional models to the code-switched setting when monolingual data from only one languages is available. In the presence of only EN data, we introduce generic augmentation techniques based on translation and generation. As such, we show that translating and aligning the EN data can significantly improve the zero-shot performance. Moreover, generating code-switched data using a generation model and a match-and-filter approach leads to improvements in a few-shot setting. We leave exploring and combining other augmentation techniques to future work.

References

Gustavo Aguilar, Fahad AlGhamdi, Victor Soto, Mona Diab, Julia Hirschberg, and Thamar Solorio. 2018. Named entity recognition on code-switched data: Overview of the CALCS 2018 shared task. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 138–147, Melbourne, Australia. Association for Computational Linguistics.

Hanan Aldarmaki and Mona Diab. 2019. Context-aware cross-lingual mapping. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3906–3911, Minneapolis, Minnesota. Association for Computational Linguistics.

Ahmed Aly, Kushal Lakhotia, Shicong Zhao, Mrinal Mohit, Barlas Oguz, Abhinav Arora, Sonal Gupta, Christopher Dewan, Stef Nelson-Lindall, and Rushin Shah. 2018. Pytext: A seamless path from NLP research to production. *CoRR*, abs/1812.08729.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

M. Saiful Bari, Muhammad Tasnim Mohiuddin, and Shafiq R. Joty. 2020. Multimix: A robust data augmentation strategy for cross-lingual NLP. *CoRR*, abs/2004.13240.

Irshad Bhat, Riyaz A. Bhat, Manish Shrivastava, and Dipti Sharma. 2017. Joining hands: Exploiting monolingual treebanks for parsing of code-mixing data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 324–330, Valencia, Spain. Association for Computational Linguistics.

Irshad Bhat, Riyaz A. Bhat, Manish Shrivastava, and Dipti Sharma. 2018. Universal dependency parsing for Hindi-English code-switching. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 987–998, New Orleans, Louisiana. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ching-Ting Chang, Shun-Po Chuang, and Hung-yi Lee. 2019. Code-switching sentence generation by generative adversarial networks and its application to data augmentation. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 554–558. ISCA.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *ArXiv*, abs/1902.10909.

Monojit Choudhury, Kalika Bali, Sunayana Sitaram, and Ashutosh Baheti. 2017. Curriculum design for code-switching: Experiments with language identification and language modeling with deep neural networks. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, pages 65–74, Kolkata, India. NLP Association of India.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 7059–7069. Curran Associates, Inc.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. *ArXiv*, abs/1710.04087.

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

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2017. Multilingual semantic parsing and code-switching. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 379–389, Vancouver, Canada. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Anna Maria Escobar and Kim Potowski. 2015. *El Español de los Estados Unidos*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Dilek Hakkani-Tur, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech 2016*, pages 715–719.

Alankar Jain, Bhargavi Paranjape, and Zachary C. Lipton. 2019. Entity projection via machine translation for cross-lingual NER. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1083–1092, Hong Kong, China. Association for Computational Linguistics.

Aravind K. Joshi. 1982. Processing of sentences with intra-sentential code-switching. In *Coling 1982: Proceedings of the Ninth International Conference on Computational Linguistics*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *CoRR*, abs/2003.02245.

Grandee Lee, Xianghu Yue, and Haizhou Li. 2019a. Linguistically motivated parallel data augmentation for code-switch language modeling. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 3730–3734. ISCA.

Kyungjae Lee, Sunghyun Park, Hojae Han, Jinyoung Yeo, Seung-won Hwang, and Juho Lee. 2019b. Learning with limited data for multilingual reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2840–2850, Hong Kong, China. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *CoRR*, abs/2001.08210.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2019. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020b. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8433–8440.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Y Bengio, li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, and Geoffrey Zweig. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 23:530–539.

Giovanni Molina, Fahad AlGhamdi, Mahmoud Ghoneim, Abdelati Hawwari, Nicolas Rey-Villamizar, Mona Diab, and Thamar Solorio. 2016. Overview for the second shared task on language identification in code-switched data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 40–49, Austin, Texas. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *ACL (1)*, pages 4996–5001. Association for Computational Linguistics.

Shana Poplack. 2004. *Code-Switching*, pages 589–596.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia. Association for Computational Linguistics.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2018. Cross-lingual transfer learning for multilingual task oriented dialog. In *2nd workshop on Conversational AI NeurIPS*.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Shrivastava, Radhika Mamidi, and Dipti M. Sharma. 2016. Shallow parsing pipeline - Hindi-English code-mixed social media text. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1340–1345, San Diego, California. Association for Computational Linguistics.

Jasdeep Singh, Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2019. XLDA: cross-lingual data augmentation for natural language inference and question answering. *CoRR*, abs/1905.11471.

Jörg Tiedemann, Željko Agić, and Joakim Nivre. 2014. Treebank translation for cross-lingual parser induction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 130–140, Ann Arbor, Michigan. Association for Computational Linguistics.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2773–2781. Curran Associates, Inc.

Changhan Wang, Kyunghyun Cho, and Douwe Kiela. 2018. Code-switched named entity recognition with embedding attention. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 154–158, Melbourne, Australia. Association for Computational Linguistics.

Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. 2019. Cross-lingual BERT transformation for zero-shot dependency parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5721–5727, Hong Kong, China. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Genta Indra Winata, Zhaojiang Lin, Jamin Shin, Zihan Liu, and Pascale Fung. 2019a. Hierarchical meta-embeddings for code-switching named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3541–3547, Hong Kong, China. Association for Computational Linguistics.

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2019b. Code-switched language models using neural based synthetic data from parallel sentences. In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*, pages 271–280. Association for Computational Linguistics.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, Denver, Colorado. Association for Computational Linguistics.

Kaisheng Yao, Geoffrey Zweig, Mei-Yuh Hwang, Yangyang Shi, and Dong Yu. 2013. Recurrent neural networks for language understanding. In *Interspeech*, pages 2524–2528.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970, Vancouver, Canada. Association for Computational Linguistics.

A Appendix

Here, we describe the details regarding the training as the validation results.

A.1 Model and Training Parameters

In Table 7, we have shown the training details for all our models. We use ADAM (Kingma and Ba, 2014) with Learning Rate (LR), Weight Decay (WD), and Batch Size (BSz) that is listed for each model. We have also shown the number of epochs and the average training time for the full CS data using 8 V100 Nvidia GPUs. For all our XLM-R experiments, we use the XLM-R large from the PyText[2] (Aly et al., 2018) which is pre-trained on 100 languages. For the XLM experiments, we use XLM-20 pre-trained over 20 languages and use the same fine-tuning parameters as XLM-R but run for more epochs.

For the LSTM models, we use a two-layer LSTM with hidden dimension of 256 and dropout of 0.3 for all connections. We use one layer of MLP of dimension 200 for both the slot tagging and the intent classification. We also use an ensemble of five models for all the LSTM experiments to reduce the variance. The LSTM model with SentencePiece embeddings in Table 4 were trained with embedding dimension of 1024 similar to the XLM-R model.

A.2 Validation Results

In Table. 9, we have shown the validation results when using the full CS training data. We have not shown the corresponding results for the zero-shot experiments as no validation data was not used and the monolingual models were tested off the shelf.

In Table. 8, we have shown the validation results for the few-shot setting.

---

[2] `https://pytext.readthedocs.io/en/master/xlm_r.html`

Model	BSz	LR	WD	Epoch	Avg Time
XLM-R (pronoun)	8	0.000005	0.0001	15	5 hr
XLM (pronoun)	8	0.000005	0.0001	20	1 hr
LSTM (pronoun+question)	64	0.03	0.00001	45	45 min

Table 7: Training Parameters

Model/Training Data	**Few shot**	**Few shot + Generate and Filter Augmentation**
XLM-R	61.7	70.4
XLM-R fine-tuned on EN	83.3	83.9
XLM-R fine-tuned on EN+ES*	83.5	84.9

Table 8: Validation Accuracy when only a few CS instances (FS) are available during training. FS+G refers to augmenting the few-shot instances with generated CS data. ES* is the auto-translated and aligned data.

Lang/Model	**MUSE**	**XLM**	**XLM-R**
CS	87.8	90.7	95.0
CS + EN	89.0	92.9	95.5

Table 9: Validation results for the Full-training on the CS data

Generating Syntactically Controlled Paraphrases without Using Annotated Parallel Pairs

Kuan-Hao Huang
University of California, Los Angeles
khhuang@cs.ucla.edu

Kai-Wei Chang
University of California, Los Angeles
kwchang@cs.ucla.edu

Abstract

Paraphrase generation plays an essential role in natural language process (NLP), and it has many downstream applications. However, training supervised paraphrase models requires many annotated paraphrase pairs, which are usually costly to obtain. On the other hand, the paraphrases generated by existing unsupervised approaches are usually syntactically similar to the source sentences and are limited in diversity. In this paper, we demonstrate that it is possible to generate syntactically various paraphrases without the need for annotated paraphrase pairs. We propose *Syntactically controlled Paraphrase Generator* (SynPG), an encoder-decoder based model that learns to disentangle the semantics and the syntax of a sentence from a collection of unannotated texts. The disentanglement enables SynPG to control the syntax of output paraphrases by manipulating the embedding in the syntactic space. Extensive experiments using automatic metrics and human evaluation show that SynPG performs better syntactic control than unsupervised baselines, while the quality of the generated paraphrases is competitive. We also demonstrate that the performance of SynPG is competitive or even better than supervised models when the unannotated data is large. Finally, we show that the syntactically controlled paraphrases generated by SynPG can be utilized for data augmentation to improve the robustness of NLP models.

1 Introduction

Paraphrase generation (McKeown, 1983) is a long-lasting task in natural language processing (NLP) and has been greatly improved by recently developed machine learning approaches and large data collections. Paraphrase generation demonstrates the potential of machines in semantic abstraction and sentence reorganization and has already been applied to many NLP downstream applications, such as question answering (Yu et al., 2018), chatbot engines (Yan et al., 2016), and sentence simplification (Zhao et al., 2018).

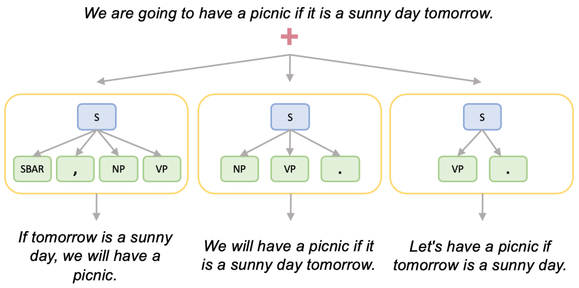


Figure 1: Paraphrase generation with syntactic control. Given a source sentence and a target syntactic specification (either a full parse tree or top levels of a parse tree), the model is expected to generate a paraphrase with the syntax following the given specification.

In recent years, various approaches have been proposed to train sequence-to-sequence (seq2seq) models on a large number of annotated paraphrase pairs (Prakash et al., 2016; Mallinson et al., 2017; Cao et al., 2017; Egonmwan and Chali, 2019). Some of them control the syntax of output sentences to improve the diversity of paraphrase generation (Iyyer et al., 2018; Goyal and Durrett, 2020; Kumar et al., 2020). However, collecting annotated pairs is expensive and induces challenges for some languages and domains. On the contrary, unsupervised approaches build paraphrase models without using parallel corpora (Li et al., 2018; Roy and Grangier, 2019; Zhang et al., 2019). Most of them are based on the variational autoencoder (Bowman et al., 2016) or back-translation (Mallinson et al., 2017; Wieting and Gimpel, 2018; Hu et al., 2019). Nevertheless, without the consideration of controlling syntax, their generated paraphrases are often similar to the source sentences and are not diverse in syntax.

This paper presents a pioneering study on syntactically controlled paraphrase generation based

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1022–1033
April 19 - 23, 2021.

on disentangling semantics and syntax. We aim to disentangle one sentence into two parts: 1) the semantic part and 2) the syntactic part. The semantic aspect focuses on the meaning of the sentence, while the syntactic part represents the grammatical structure. When two sentences are paraphrased, their semantic aspects are supposed to be similar, while their syntactic parts should be different. To generate a syntactically different paraphrase of one sentence, we can keep its semantic part unchanged and modify its syntactic part.

Based on this idea, we propose **Syn**tactically Controlled **P**araphrase **G**enerator (SynPG)[1], a Transformer-based model (Vaswani et al., 2017) that can generate syntactically different paraphrases of one source sentence based on some target syntactic parses. SynPG consists of a semantic encoder, a syntactic encoder, and a decoder. The semantic encoder considers the source sentence as a bag of words without ordering and learns a contextualized embedding containing only the semantic information. The syntactic encoder embeds the target parse into a contextualized embedding including only the syntactic information. Then, the decoder combines the two representations and generates a paraphrase sentence. The design of disentangling semantics and syntax enables SynPG to learn the association between words and parses and be trained by reconstructing the source sentence given its unordered words and its parse. Therefore, we do not require any annotated paraphrase pairs but only unannotated texts to train SynPG.

We verify SynPG on four paraphrase datasets: ParaNMT-50M (Wieting and Gimpel, 2018), Quora (Iyer et al., 2017), PAN (Madnani et al., 2012), and MRPC (Dolan et al., 2004). The experimental results reveal that when being provided with the syntactic structures of the target sentences, SynPG can generate paraphrases with the syntax more similar to the ground truth than the unsupervised baselines. The human evaluation results indicate that SynPG achieves competitive paraphrase quality to other baselines while its generated paraphrases are more accurate in following the syntactic specifications. In addition, we show that when the training data is large enough, the performance of SynPG is competitive or even better than supervised approaches. Finally, we demonstrate that the syntactically controlled paraphrases generated by SynPG can be used for data augmentation to defense syntactically adversarial attack (Iyyer et al., 2018) and improve the robustness of NLP models.

2 Unsupervised Paraphrase Generation

We aim to train a paraphrase model without using annotated paraphrase pairs. Given a source sentence $\mathbf{x} = (x_1, x_2, ..., x_n)$, our goal is to generate a paraphrase sentence $\mathbf{y} = (y_1, y_2, ..., y_m)$ that is expected to maintain the same meaning of $\mathbf{x}$ but has a different syntactic structure from $\mathbf{x}$.

Syntactic control. Motivated by previous work (Iyyer et al., 2018; Zhang et al., 2019; Kumar et al., 2020), we allow our model to access additional syntactic specifications as the control signals to guide the paraphrase generation. More specifically, in addition to the source sentence $\mathbf{x}$, we give the model a target constituency parse $\mathbf{p}$ as another input. Given the input $(\mathbf{x}, \mathbf{p})$, the model is expected to generate a paraphrase $\mathbf{y}$ that is semantically similar to the source sentence $\mathbf{x}$ and syntactically follows the target parse $\mathbf{p}$. In the following discussions, we assume the target parse $\mathbf{p}$ to be a full constituency parse tree. Later on, in Section 2.3, we will relax the syntax guidance to be a *template*, which is defined as the top two levels of a full parse tree. We expect that a successful model can control the syntax of output sentences and generate syntactically different paraphrases based on different target parses, as illustrated in Figure 1.

Similar to previous work (Iyyer et al., 2018; Zhang et al., 2019), we linearize the constituency parse tree to a sequence. For example, the linearized parse of the sentence "*He eats apples.*" is `(S(NP(PRP))(VP(VBZ)(NP(NNS)))(.)).` Accordingly, a parse tree can be considered as a sentence $\mathbf{p} = (p_1, p_2, ..., p_k)$, where the tokens in $\mathbf{p}$ are non-terminal symbols and parentheses.

2.1 Proposed Model

Our main idea is to disentangle a sentence into the semantic part and the syntactic part. Once the model learns the disentanglement, it can generate a syntactically different paraphrase of one given sentence by keeping its semantic part unchanged and modifying only the syntactic part.

Figure 2 illustrates the proposed paraphrase model called SynPG, a seq2seq model consisting of a semantic encoder, a syntactic encoder, and a decoder. The semantic encoder captures only the semantic information of the source sentence $\mathbf{x}$,

---

[1] Our code and the pretrained models are available at `https://github.com/uclanlp/synpg`

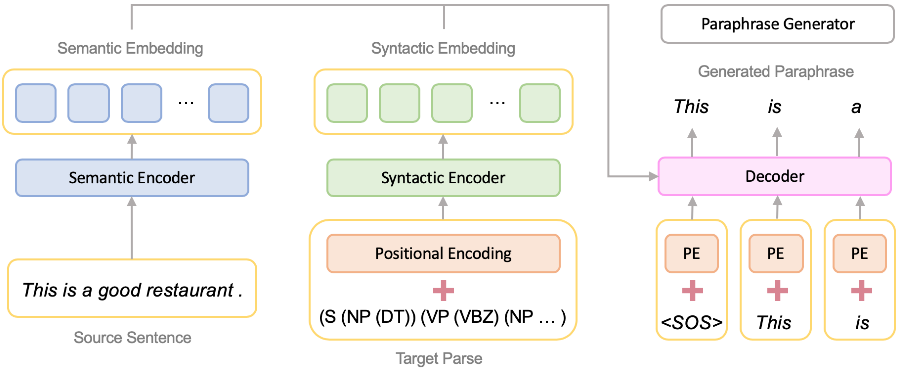


Figure 2: SynPG embeds the source sentence and the target parse into a semantic embedding and a syntactic embedding, respectively. Then, SynPG generates a paraphrase sentence based on the two embeddings.

while the syntactic encoder extracts only the syntactic information from the target parse $\mathbf{p}$. The decoder then combines the encoded semantic and syntactic information and generates a paraphrase $\mathbf{y}$. We discuss the details of SynPG in the following.

Semantic encoder. The semantic encoder embeds a source sentence $\mathbf{x}$ into a contextualized semantic embedding $\mathbf{z}_{sem}$. In other words,

$$\mathbf{z}_{sem} = (z_1, z_2, ..., z_n) = \text{Enc}_{sem}((x_1, x_2, .., x_n)).$$

The semantic embedding $\mathbf{z}_{sem}$ is supposed to contain only the semantic information of the source sentence $\mathbf{x}$. To separate the semantic information from the syntactic information, we use a Transformer (Vaswani et al., 2017) *without the positional encoding* as the semantic encoder. We posit that by removing position information from the source sentence $\mathbf{x}$, the semantic embedding $\mathbf{z}_{sem}$ would encode less syntactic information.

We assume that words without ordering capture most of the semantics of one sentence. Indeed, semantics is also related to the order. For example, exchanging the subject and the object of a sentence changes its meaning. However, the decoder trained on a large corpus also captures the *selectional preferences* (Katz and Fodor, 1963; Wilks, 1975) in generation, which enables the decoder to infer the proper order of words. In addition, we observe that when two sentences are paraphrased, they usually share similar words, especially those words related to the semantics. For example, "*What is the best way to improve writing skills?*" and "*How can I improve my writing skills?*" are paraphrased, and the shared words (*improve*, *writing*, and *skills*) are strongly related to the semantics. In Section 4, we show that our designed semantic embedding captures enough semantic information to generate paraphrases.

Syntactic encoder. The syntactic encoder embeds the target parse $\mathbf{p} = (p_1, p_2, ..., p_k)$ into a contextualized syntactic embedding $\mathbf{z}_{syn}$. That is,

$$\mathbf{z}_{syn} = (z_1, z_2, ..., z_k) = \text{Enc}_{syn}((p_1, p_2, .., p_k)).$$

Since the target parse $\mathbf{p}$ contains no semantic information but only syntactic information, we use a Transformer *with the positional encoding* as the syntactic encoder.

Decoder. Finally, we design a decoder that takes the semantic embedding $\mathbf{z}_{sem}$ and the syntactic embedding $\mathbf{z}_{syn}$ as the input and generates a paraphrase $\mathbf{y}$. In other words,

$$\mathbf{y} = (y_1, y_2, ..., y_m) = \text{Dec}(\mathbf{z}_{sem}, \mathbf{z}_{syn}).$$

We choose Transformer as the decoder to generate $\mathbf{y}$ autoregressively. Notice that the semantic embedding $\mathbf{z}_{sem}$ does not encode the position information and the syntactic embedding $\mathbf{z}_{syn}$ does not contain semantics. This forces the decoder to extract the semantics from $\mathbf{z}_{sem}$ and retrieve the syntactic structure from $\mathbf{z}_{syn}$. The attention weights attaching to $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ make the decoder learn the association between the semantics and the syntax as well as the relation between the word order and the parse structures. Therefore, SynPG is able to reorganize the source sentence and use the given syntactic structure to rephrase the source sentence.

2.2 Unsupervised Training

Our design of the disentanglement makes it possible to train SynPG without using annotated pairs. We train SynPG with the objective to reconstruct the source sentences. More specifically, when training on a sentence $\mathbf{x}$, we first separate $\mathbf{x}$ into two parts: 1) an unordered word list $\bar{\mathbf{x}}$ and 2) its linearized parse $\mathbf{p}_x$ (can be obtained by a pretrained parser). Then, SynPG is trained to reconstruct $\mathbf{x}$ from $(\bar{\mathbf{x}}, \mathbf{p}_x)$ with the reconstruction loss

$$\mathcal{L} = -\sum_{i=1}^{n} \log P(y_i = x_i | \bar{\mathbf{x}}, \mathbf{p}_x, \mathbf{y}_1, ..., \mathbf{y}_{i-1}).$$

Notice that if we do not disentangle the semantics and the syntax, and directly use a seq2seq model to reconstruct $\mathbf{x}$ from $(\mathbf{x}, \mathbf{p}_x)$, it is likely that the seq2seq model only learns to copy $\mathbf{x}$ and ignores $\mathbf{p}_x$ since $\mathbf{x}$ contains all the necessary information for the reconstruction. Consequently, at inference time, no matter what target parse $\mathbf{p}$ is given, the seq2seq model always copies the whole source sentence $\mathbf{x}$ as the output (more discussion in Section 4).

On the contrary, SynPG learns the disentangled embeddings $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$. This makes SynPG capture the relation between the semantics and the syntax to reconstruct the source sentence $\mathbf{x}$. Therefore, at test time, given the source sentence $\mathbf{x}$ and a new target parse $\mathbf{p}$, SynPG is able to apply the learned relation to rephrase the source sentence $\mathbf{x}$ according to the target parse $\mathbf{p}$.

Word dropout. We observe that the ground truth paraphrase may contain some words not appearing in the source sentence; however, the paraphrases generated by the vanilla SynPG tend to include only words appearing in the source sentence due to the reconstruction training objective. To encourage SynPG to improve the diversity of the word choices in the generated paraphrases, we randomly discard some words from the source sentence during training. More precisely, each word has a probability to be dropped out in each training iteration. Accordingly, SynPG has to predict the missing words during the reconstruction, and this enables SynPG to select different words from the source sentence to generate paraphrases. More details are discussed in Section 4.5.

2.3 Templates and Parse Generator

In the previous discussion, we assume that a full target constituency parse tree is provided as the input to SynPG. However, the full parse tree of the target paraphrase sentence is unlikely available at inference time. Therefore, following the setting in Iyyer et al. (2018), we consider generating the paraphrase based on the *template*, which is defined as the top two levels of the full constituency parse tree. For example, the template of `(S(NP(PRP))(VP(VBZ)(NP(NNS)))(.))` is `(S(NP)(VP))(.))`.

Motivated by Iyyer et al. (2018), we train a parse generator to generate full parses from templates. The proposed parse generator has the same architecture as SynPG, but the input and the output are different. The parse generator takes two inputs: a tag sequence $\mathbf{tag}_x$ and a target template $\mathbf{t}$. The tag sequence $\mathbf{tag}_x$ contains all the POS tags of the source sentence $\mathbf{x}$. For example, the tag sequence of the sentence "*He eats apples.*" is "`<PRP> <VBZ> <NNS> <.>`". Similar to the source sentence in SynPG, we do not consider the word order of the tag sequence during encoding. The expected output of the parse generator is a full parse $\tilde{\mathbf{p}}$ whose a syntactic structure follows the target template $\mathbf{t}$.

We train the parse generator without any additional annotations as well. Let $\mathbf{t}_x$ be the the template of $\mathbf{p}_x$ (the parse of $\mathbf{x}$), we end-to-end train the parse generator with the input being $(\mathbf{tag}_x, \mathbf{t}_x)$ and the output being $\mathbf{p}_x$.

Generating paraphrases from templates. The parse generator makes us generate paraphrases by providing target templates instead of target parses. The steps to generate a paraphrase given a source sentence $\mathbf{x}$ and a target template $\mathbf{t}$ are as follows:

1. Get the tag sequence $\mathbf{tag}_x$ of the source sentence $\mathbf{x}$.
2. Use the parse generator to generate a full parse $\tilde{\mathbf{p}}$ with input $(\mathbf{tag}_x, \mathbf{t})$.
3. Use SynPG to generate a paraphrase $\mathbf{y}$ with input $(\mathbf{x}, \tilde{\mathbf{p}})$.

Post-processing. We notice that certain templates are not suitable for some source sentences and therefore the generated paraphrases are nonsensical. We follow Iyyer et al. (2018) and use n-gram overlap and paraphrastic similarity computed by the model[2] from Wieting and Gimpel (2018) to remove nonsensical paraphrases[3].

3 Experimental Settings

We conduct extensive experiments to demonstrate that SynPG performs better syntactic control than other unsupervised paraphrase models, while the quality of generated paraphrases by SynPG is comparable to others. In addition, we show that the performance of SynPG is competitive or even better than supervised models when the training data is large enough.

---

[2] https://github.com/jwieting/para-nmt-50m

[3] We set the minimum n-gram overlap to 0.3 and the minimum paraphrastic similarity to 0.7.

3.1 Datasets

For the training data, we consider ParaNMT-50M (Wieting and Gimpel, 2018), a paraphrase dataset containing over 50 million pairs of reference sentences and the corresponding paraphrases as well as the quality scores. We select about 21 million pairs with higher quality scores as our training examples. Notice that we use *only the reference sentences* to train SynPG and unsupervised paraphrase models since we do not require paraphrase pairs.

We sample 6,400 pairs from ParaNMT-50M as the testing data. To evaluate the transferability of SynPG, we also consider the other three datasets: 1) Quora (Iyer et al., 2017) contains over 400,000 paraphrase pairs and we sample 6,400 pairs from them. 2) PAN (Madnani et al., 2012) contains 5,000 paraphrase pairs. 3) MRPC (Dolan et al., 2004) contains 2,753 paraphrase pairs.

3.2 Evaluation

We consider paraphrase pairs to evaluate all the models. For each test paraphrase pair $(\mathbf{x}_1, \mathbf{x}_2)$, we consider $\mathbf{x}_1$ as the source sentence and treat $\mathbf{x}_2$ as the target sentence (ground truth). Let $\mathbf{p}_2$ be the parse of $\mathbf{x}_2$, given $(\mathbf{x}_1, \mathbf{p}_2)$, The model is expected to generate a paraphrase $\mathbf{y}$ that is similar to the target sentence $\mathbf{x}_2$.

We use BLEU score (Papineni et al., 2002) and human evaluation to measure the similarity between $\mathbf{x}_2$ and $\mathbf{y}$. Moreover, to evaluate how well the generated paraphrase $\mathbf{y}$ follows the target parse $\mathbf{p}_2$, we define the *template matching accuracy* (TMA) as follows. For each ground truth sentence $\mathbf{x}_2$ and the corresponding generated paraphrase $\mathbf{y}$, we get their parses ($\mathbf{p}_2$ and $\mathbf{p}_y$) and templates ($\mathbf{t}_2$ and $\mathbf{t}_y$). Then, we calculate the percentage of pairs whose $\mathbf{t}_y$ *exactly matches* $\mathbf{t}_2$ as the *template matching accuracy.*

3.3 Models for Comparison

We consider the following unsupervised paraphrase models: 1) **CopyInput**: a naïve baseline which directly copies the source sentence as the output without paraphrasing. 2) **BackTrans**: back-translation is proposed to generate paraphrases (Mallinson et al., 2017; Wieting and Gimpel, 2018; Hu et al., 2019). In our experiment, we use the pretrained EN-DE and DE-EN translation models[4] proposed by Ng et al. (2019) to conduct back-translation. Notice that training translation models requires additional translation pairs. Therefore, BackTrans needs more resources than ours and the translation data may not available for some low-resource languages. 3) **VAE**: we consider a vanilla variational autoencoder (Bowman et al., 2016) as a simple baseline. 4) **SIVAE**: syntax-infused variational autoencoder (Zhang et al., 2019) utilizes additional syntax information to improve the quality of sentence generation and paraphrase generation. Unlike SynPG, SIVAE does not disentangle the semantics and syntax. 5) **Seq2seq-Syn**: we train a seq2seq model with Transformer architecture to reconstruct $\mathbf{x}$ from $(\mathbf{x}, \mathbf{p}_x)$ without the disentanglement. We use this model to study the influence of the disentanglement. 6) **SynPG**: our proposed model which learns disentangled embeddings.

We also compare SynPG with supervised approaches. We consider the following: 1) **Seq2seq-Sup**: a seq2seq model with Transformer architecture trained on whole ParaNMT-50M pairs. 2) **SCPN**: syntactically controlled paraphrase network (Iyyer et al., 2018) is a supervised paraphrase model with syntactic control trained on ParaNMT-50M pairs. We use their pretrained model[5].

3.4 Implementation Details

We consider byte pair encoding (Sennrich et al., 2016) for tokenization and use Stanford CoreNLP parser (Manning et al., 2014) to get constituency parses. We set the max length of sentences to 40 and set the max length of linearized parses to 160 for all the models. For the encoders and the decoder of SynPG, we use the standard Transformer (Vaswani et al., 2017) with default parameters. The word embedding is initialized by GloVe (Pennington et al., 2014). We use Adam optimizer with the learning rate being 10^{-4} and the weight decay being 10^{-5}. We set the word dropout probability to 0.4 (more discussion in Section 4.5). The number of epoch for training is set to 5.

Seq2seq-Syn, Seq2seq-Sup are trained with the similar setting. We reimplemnt VAE and SIVAE, and all the parameters are set to the default value in the original papers.

4 Results and Discussion

4.1 Syntactic Control

We first discuss if the syntactic specification enables SynPG to control the output syntax better.

---

[4] https://github.com/pytorch/fairseq/tree/master/examples/wmt19

[5] https://github.com/miyyer/scpn

	Model	ParaNMT		Quora		PAN		MRPC	
		TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
No Paraphrasing	CopyInput	33.6	16.4	55.0	20.0	37.3	26.8	47.9	30.7
Unsupervised Models	BackTrans	29.0	16.3	53.0	16.4	27.9	16.2	47.2	21.6
	VAE	26.3	9.6	44.0	8.1	19.4	5.2	20.8	1.2
With Syntactic Specifications	SIVAE	30.0	12.8	48.3	13.1	26.6	11.8	21.5	5.1
	Seq2seq-Syn	33.5	16.3	54.9	19.8	37.1	**26.5**	47.7	**30.4**
	SynPG	**71.0**	**32.2**	**82.6**	**33.2**	**66.3**	26.4	**74.0**	26.2

Table 1: Paraphrase results on four datasets. TMA denotes the template matching accuracy, which evaluates how often the generated paraphrases follow the target parses. With the syntactic control, SynPG obtains higher BLEU score and the template matching accuracy. This implies the paraphrases generated by SynPG are more similar to the ground truths and follow the target parses more accurately.

Model	Example 1 (ParaNMT)	Example 2 (Quora)
Source Sent.	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
Ground Truth	if we don't act quickly, the children will die.	how could i improve my writing skill?
BackTrans	these children will die if we do not act now.	what are the best ways to improve your writing skills?
VAE	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
SIVAE	these children are gonna die if we don't act now .	what are the best ways to improve writing skills?
Seq2seq-Syn	these children are gonna die if we don't act now.	what are the best ways to improve writing skills?
SynPG	if we don't act now, these children will die.	how can i improve my writing skills?

Table 2: Paraphrases generated by each model. SynPG can generate paraphrases with the syntax more similar to the ground truth than other baselines.

Table 1 shows the template matching accuracy and BLEU score for SynPG and the unsupervised baselines. Notice that here we use the full parse trees as the syntactic specifications. We will discuss the influence of using the template as the syntactic specifications in Section 4.3.

Although we train SynPG on the reference sentences of ParaNMT-50M, we observe that SynPG performs well on Quora, PAN, and MRPC as well. This validates that SynPG indeed learns the syntactic rules and can transfer the learned knowledge to other datasets. CopyInput gets high BLEU scores; however, due to the lack of paraphrasing, it obtains low template matching scores. Compared to the unsupervised baselines, SynPG achieves higher template matching accuracy and higher BLEU scores on all datasets. This verifies that the syntactic specification is indeed helpful for syntactic control.

Next, we compare SynPG with Seq2seq-Syn and SIVAE. All models are given syntactic specifications; however, without the disentanglement, Seq2seq-Syn and SIVAE tend to copy the source sentence as the output and therefore get low template matching scores.

Table 2 lists some paraphrase examples generated by all models. Again, we observe that without syntactic specifications, the paraphrases generated by unsupervised baselines are similar to the source sentences. Without the disentanglement, Seq2seq-Syn and SIVAE always copy the source sentences. SynPG is the only model can generate paraphrases syntactically similar to the ground truths.

4.2 Human Evaluation

We perform human evaluation using Amazon Mechanical Turk to evaluate the quality of generated paraphrases. We follow the setting of previous work (Kok and Brockett, 2010; Iyyer et al., 2018; Goyal and Durrett, 2020). For each model, we randomly select 100 pairs of source sentence $\mathbf{x}$ and the corresponding generated paraphrase $\mathbf{y}$ from ParaNMT-50M test set (after being post-processed as mentioned in Section 2.3) and have three Turkers annotate each pair. The annotations are on a three-point scale: **0** means $\mathbf{y}$ is not a paraphrase of $\mathbf{x}$; **1** means $\mathbf{x}$ is paraphrased into $\mathbf{y}$ but $\mathbf{y}$ contains some grammatical errors; **2** means $\mathbf{x}$ is paraphrased into $\mathbf{y}$, which is grammatically correct.

The results of human evaluation are reported in Table 3. If paraphrases rated **1** or **2** are considered meaningful, we notice that SynPG generates meaningful paraphrases at a similar frequency to that of SIVAE. However, SynPG tends to generate more ungrammatical paraphrases (those rated **1**). We think the reason is that most of paraphrases generated by SIVAE are very similar to the source sentences, which are usually grammatically correct. On the other hand, SynPG is encouraged to

Model	**2**	**1**	**0**	**2+1**	Hit Rate
BackTrans	63.6	22.4	14.0	86.0	11.0
SIVAE	57.6	20.3	22.0	78.0	6.5
SynPG	44.3	32.0	23.7	76.3	28.9

Table 3: Human evaluation on a three-point scale (**0** = not a paraphrase, **1** = ungrammatical paraphrase, **2** = grammatical paraphrase). SynPG performs better on hit rate (defined as the percentage of generated paraphrase getting **2** and matching the target parse at the same time) than other unsupervised models.

use different syntactic structures from the source sentences to generate paraphrases, which may lead some grammatical errors.

Furthermore, we calculate the *hit rate*, the percentage of generated paraphrases getting **2** and matching the target parse at the same time. The hit rate measures how often the generated paraphrases follow the target parses and preserve the semantics (verified by human evaluation) simultaneously. The results show that SynPG gets higher hit rate than other models.

4.3 Target Parses vs. Target Templates

Next, we discuss the influence of generating paraphrase by using templates instead of using full parse trees. For each paraphrase pair $(\mathbf{x}_1, \mathbf{x}_2)$ in test data, we consider two ways to generate the paraphrase. 1) Generating the paraphrase with the target parse. We use SynPG to generate a paraphrase directly from $(\mathbf{x}_1, \mathbf{p}_2)$. 2) Generating the paraphrase with the target template. We first use the parse generator to generate a parse $\tilde{\mathbf{p}}$ from $(\mathbf{tag}_1, \mathbf{t}_2)$, where $\mathbf{tag}_1$ is the tag sequence of $\mathbf{x}_1$ and $\mathbf{t}_2$ is the template of $\mathbf{p}_2$. Then we use SynPG to generate a paraphrase from $(\mathbf{x}_1, \tilde{\mathbf{p}})$. We calculate the template matching accuracy to compare these two ways to generate paraphrases, as shown in Table 4. We also report the template matching accuracy of the generated parse $\tilde{\mathbf{p}}$.

We find that most of generated parses $\tilde{\mathbf{p}}$ indeed follow the target templates, which means that the parse generator usually generates good parses $\tilde{\mathbf{p}}$. Next, we observe that generating paraphrases with target parses usually performs better than with target templates. The results show a trade-off. Using templates proves more effortless during the generation process, but may compromise the syntactic control ability. In comparison, by using the target parses, we have to provide more detailed parses, but our model can control the syntax better.

Another benefit of generating paraphrase with target templates is that we can easily generate a lot of syntactically different paraphrases by feeding the model with different templates. Table 5 lists some paraphrases generated by SynPG with different templates. We can perceive that most generated paraphrases are grammatically correct and have similar meanings to the original sentence.

Model	Template Matching Accuracy			
	ParaNMT	Quora	PAN	MRPC
Paraphrases generated by target parses	71.0	82.6	66.3	74.0
Paraphrases generated by target templates	54.1	73.4	51.7	62.3
Parses $\tilde{\mathbf{p}}$ generated by parse generator	98.4	99.0	95.7	93.9

Table 4: Influence of using templates. Using templates proves more effortless during the generation process, but may compromise the syntactic control ability.

4.4 Training SynPG on Larger Dataset

Finally, we demonstrate that the performance of SynPG can be further improved and be even competitive to supervised models on some datasets if we consider more training data. The advantage of unsupervised paraphrase models is that we do not require parallel pairs for training. Therefore, we can easily boost the performance of SynPG by consider more unannotated texts into training.

We consider SynPG-Large, the SynPG model trained on the reference sentences of ParaNMT-50M as well as One Billion Word Benchmark (Chelba et al., 2014), a large corpus for training language models. We sample about 24 million sentences from One Billion Word and add them to the training set. In addition, we fine-tune SynPG-Large on only the reference sentences of the testing paraphrase pairs, called SynPG-FT.

From Table 6, We observe that enlarging the training data set indeed improves the performance. Also, with the fine-tuning, the performance of SynPG can be much improved and even is better than the performance of supervised models on some datasets. The results demonstrate the potential of unsupervised paraphrase generation with syntactic control.

4.5 Word Dropout Rate

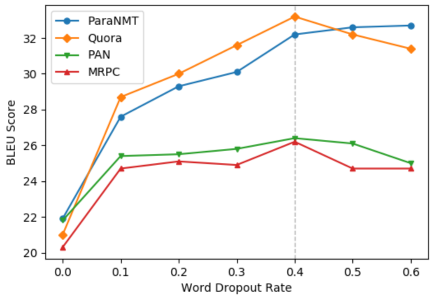
The word dropout rate plays an important role for SynPG since it controls the ability of SynPG to generate new words in paraphrases. We test differ-

Template	Generated Paraphrase
Original	can you adjust the cameras?
(S(NP)(VP)(.))	you can adjust the cameras.
(SBARQ(ADVP)(,)(S)(,)(SQ)(.))	well, adjust the cameras , can you?
(S(PP)(,)(NP)(VP)(.))	on the cameras, you can adjust them?
Original	she doesn't keep pictures from her childhood.
(SBARQ(WHADVP)(SQ)(.))	why doesn't she keep her pictures from childhood.
(S(``)(NP)(VP)('')(NP)(VP)(.))	" she doesn't keep pictures from her childhood " she said.
(S(ADVP)(NP)(VP)(.))	perhaps she doesn't keep pictures from her childhood.

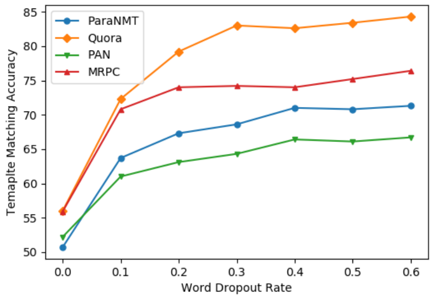
Table 5: Paraphrases generated by SynPG with different templates.

	Model	ParaNMT		Quora		PAN		MRPC	
		TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
Ours	SynPG	71.0	32.2	82.6	33.2	66.3	26.4	74.0	26.2
	SynPG-Large	70.3	31.8	83.8	34.7	66.6	27.1	79.3	36.2
	SynPG-FT	–	–	86.3	**44.4**	66.4	34.2	**80.7**	**44.6**
Supervised Models	Seq2seq-Sup	40.2	19.6	54.0	11.3	29.2	13.1	44.3	16.3
	SCPN	**83.9**	**58.3**	**87.1**	41.0	**72.3**	**37.6**	80.1	41.8

Table 6: Training on larger dataset improves the performance of SynPG. Since training SynPG does not require annotated paraphrase pairs, it is possible to fine-tune SynPG on the texts in the target domain. With the fine-tuning, SynPG can have competitive or even better performance than supervised approaches.



(a) BLEU score



(b) Template matching accuracy

Figure 3: Influence of word drop out rate. Setting the word dropout rate to 0.4 can achieve the best BLEU score. However, higher word dropout rate leads to better template matching accuracy.

ent word dropout rates and report the BLEU scores and the template matching accuracy in Figure 3.

From Figure 3a, we can observe that setting the word dropout rate to 0.4 can achieve the best BLEU score in most of datasets. The only exception is ParaNMT, which is the dataset used for training. On the other hand, Figure 3b shows that higher word dropout rate leads to better template matching accuracy. The reason is that higher word dropout rate gives SynPG more flexibility to generate paraphrases. Therefore, the generated paraphrases can match the target syntactic specifications better. However, higher word dropout rate also make SynPG have less ability to preserve the meaning of source sentences. Considering all the factors above, we recommend to set the word dropout rate to 0.4 for SynPG.

5 Improving Robustness of Models

Recently, a lot of work show that NLP models can be fooled by different types of adversarial attacks (Alzantot et al., 2018; Ebrahimi et al., 2018; Iyyer et al., 2018; Tan et al., 2020; Jin et al., 2020). Those attacks generate *adversarial examples* by slightly modifying the original sentences without changing the meanings, while the NLP models change the predictions on those examples. However, a robust model is expected to output the same labels. Therefore, how to make NLP models not affected by the adversarial examples becomes an important task.

Since SynPG is able to generate syntactically different paraphrases, we can improve the robustness of NLP models by data augmentation. The models trained with data augmentation are thus more robust to the syntactically adversarial examples (Iyyer et al., 2018), which are the adversarial sentences that are paraphrases to the original sen-

Model	SST-2		MRPC		RTE	
	Acc.	Brok.	Acc.	Brok.	Acc.	Brok.
Base	91.9	46.7	84.1	52.8	63.2	58.3
SynPG	88.9	**39.6**	80.1	**35.5**	60.7	**33.9**

Table 7: Data augmentation improves the robustness of models. SynPG denotes the base classifier trained on augmented data generated by SynPG. Acc denotes the accuracy in the original dataset (the higher is the better). Brok denotes the percentage of examples changing predictions after attacking (the lower is the better).

tences but with syntactic difference.

We conduct experiments on three classification tasks covered by GLUE benchmark (Wang et al., 2019): SST-2, MRPC, and RTE. For each training example, we use SynPG to generate four syntactically different paraphrases and add them to the training set. We consider the setting to generate syntactically adversarial examples by SCPN (Iyyer et al., 2018). For each testing example, we generate five candidates of adversarial examples. If the classifier gives at least one wrong prediction on the candidates, we treat the attack to be successful.

We compare the model without data augmentation (Base) and with data augmentation (SynPG) in Table 7. We observe that with the data augmentation, the accuracy before attacking is slightly worse than Base. However, after attacking, the percentage of examples changing predictions is much less than Base, which implies that data augmentation indeed improves the robustness of models.

6 Related Work

Paraphrase generation. Traditional approaches usually require hand-crafted rules, such as rule-based methods (McKeown, 1983), thesaurus-based methods (Bolshakov and Gelbukh, 2004; Kauchak and Barzilay, 2006), and lattice matching methods (Barzilay and Lee, 2003). However, the diversity of their generated paraphrases is usually limited.

Recently, neural models make success on paraphrase generation (Prakash et al., 2016; Mallinson et al., 2017; Cao et al., 2017; Egonmwan and Chali, 2019; Li et al., 2019; Gupta et al., 2018). These approaches treat paraphrase generation as a translation task and design seq2seq models based on a large amount of parallel data. To reduce the effort to collect parallel data, unsupervised paraphrase generation has attracted attention in recent years. Wieting et al. (2017); Wieting and Gimpel (2018) use translation models to generate paraphrases via back-translation. Zhang et al. (2019); Roy and Grangier (2019) generate paraphrases based on variational autoencoders. Reinforcement learning techniques are also considered for paraphrase generation (Li et al., 2018).

Controlled generation. Recent work on controlled generation can be grouped into two families. The first family is doing end-to-end training with an additional trigger to control the attributes, such as sentiment (Shen et al., 2017; Hu et al., 2017; Fu et al., 2018; Peng et al., 2018; Dai et al., 2019), tense (Logeswaran et al., 2018), plots (Ammanabrolu et al., 2020; Fan et al., 2019; Tambwekar et al., 2019; Yao et al., 2019; Goldfarb-Tarrant et al., 2019, 2020), societal bias (Wallace et al., 2019; Sheng et al., 2020b,a), and syntax (Iyyer et al., 2018; Goyal and Durrett, 2020; Kumar et al., 2020). The second family controls the attributes by learning disentangled representations. For example, Romanov et al. (2019) disentangle the meaning and the form of a sentence. Chen et al. (2019b,a); Bao et al. (2019) disentangle the semantics and the syntax of a sentence.

7 Conclusion

We present syntactically controlled paraphrase generator (SynPG), an paraphrase model that can control the syntax of generated paraphrases based on the given syntactic specifications. SynPG is designed to disentangle the semantics and the syntax of sentences. The disentanglement enables SynPG to be trained without the need for annotated paraphrase pairs. Extensive experiments show that SynPG performs better syntactic control than unsupervised baselines, while the quality of the generated paraphrases is competitive to supervised approaches. Finally, we demonstrate that SynPG can improve the robustness of NLP models by generating additional training examples. SynPG is especially helpful for the domain where annotated paraphrases are hard to obtain but a large amount of unannotated text is available. One limitation of SynPG is the need for mannually providing target syntactic templates at inference time. We leave the automatic template generation as our future work.

Acknowledgments

We thank anonymous reviewers for their helpful feedback. We thank Kashif Shah and UCLA-NLP group for the valuable discussions and comments. This work is supported in part by Amazon Research Award.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Prithviraj Ammanabrolu, Ethan Tien, W. Cheung, Z. Luo, W. Ma, Lara J. Martin, and Mark O. Riedl. 2020. Story realization: Expanding plot events into sentences. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xin-Yu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics*.

Igor A. Bolshakov and Alexander F. Gelbukh. 2004. Synonymous paraphrasing using wordnet and internet. In *Proceedings of the 9th International Conference on Applications of Natural Languages to Information Systems*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *Proceedings of 15th Annual Conference of the International Speech Communication Association*.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019a. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019b. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Conference of the Association for Computational Linguistics*.

Elozino Egonmwan and Yllias Chali. 2019. Transformer and seq2seq model for paraphrase generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation@EMNLP-IJCNLP*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *NAACL system demonstration*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Conference of the Association for Computational Linguistics*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019. PARABANK: monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *The Thirty-Third AAAI Conference on Artificial Intelligence*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning*.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. First quora dataset release: Question pairs. *data.quora.com*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Jerrold J Katz and Jerry A Fodor. 1963. The structure of a semantic theory. *language*, 39(2):170–210.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics*.

Stanley Kok and Chris Brockett. 2010. Hitting the right paraphrases in good time. In *Proceedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics*.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha P. Talukdar. 2020. Syntax-guided controlled generation of paraphrases. *Trans. Assoc. Comput. Linguistics*, 8:330–345.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*.

Nitin Madnani, Joel R. Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52th Conference of the Association for Computational Linguistics*.

Kathleen R. McKeown. 1983. Paraphrasing questions using given and new information. *Am. J. Comput. Linguistics*, 9(1):1–10.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's WMT19 news translation task submission. In *WMT*.

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

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *NAACL Workshop*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek V. Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of the 26th International Conference on Computational Linguistics*.

Alexey Romanov, Anna Rumshisky, Anna Rogers, and David Donahue. 2019. Adversarial decomposition of text representation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Conference of the Association for Computational Linguistics*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2020a. " nice try, kiddo": Ad hominems in dialogue systems. *arXiv preprint arXiv:2010.12820*.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2020b. Towards controllable biases in language generation. In *EMNLP-Findings*.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, B. Harrison, and Mark O. Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*.

Samson Tan, Shafiq R. Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Conference of the Association for Computational Linguistics*.

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

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 7th International Conference on Learning Representations*.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Conference of the Association for Computational Linguistics*.

John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Yorick Wilks. 1975. An intelligent analyzer and understander of english. *Commun. ACM*, 18(5):264–274.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. 2016. Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Lili Yao, Nanyun Peng, Weischedel Ralph, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *Proceedings of the 6th International Conference on Learning Representations*.

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. 2019. Syntax-infused variational autoencoder for text generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Data Augmentation for Hypernymy Detection

Thomas Kober
Rasa
Berlin, DE
t.kober@rasa.com

Julie Weeds
University of Sussex
Brighton, UK
j.e.weeds@sussex.ac.uk

Lorenzo Bertolini
University of Sussex
Brighton, UK
l.bertolini@sussex.ac.uk

David Weir
University of Sussex
Brighton, UK
d.j.weir@sussex.ac.uk

Abstract

The automatic detection of hypernymy relationships represents a challenging problem in NLP. The successful application of state-of-the-art supervised approaches using distributed representations has generally been impeded by the limited availability of high quality training data. We have developed two novel data augmentation techniques which generate new training examples from existing ones. First, we combine the linguistic principles of hypernym transitivity and intersective modifier-noun composition to generate additional pairs of vectors, such as *small dog - dog* or *small dog - animal*, for which a hypernymy relationship can be assumed. Second, we use generative adversarial networks (GANs) to generate pairs of vectors for which the hypernymy relation can also be assumed. We furthermore present two complementary strategies for *extending* an existing dataset by leveraging linguistic resources such as WordNet. Using an evaluation across 3 different datasets for hypernymy detection and 2 different vector spaces, we demonstrate that both of the proposed automatic data augmentation and dataset extension strategies substantially improve classifier performance.

1 Introduction

The detection of hypernymy relationships between terms represents a challenging commonsense inference problem and is a major component in recognising paraphrase and textual entailment in larger units of text. Consequently, it is important for Question-Answering, Text Simplification and Automatic Summarization. For example,

There are lots of cars and vans at the port today.

might be adequately summarised by

There are lots of vehicles at the port today.

as *car* and *van* both lexically entail, i.e. they are both hyponyms of the more general term *vehicle*.

Furthermore, the recognition and discovery of hyponym-hypernym relations is a foundational part of constructing taxonomies, which has a range of practical applications in a variety of domains such as Healthcare (Barisevičius et al., 2018) or Fashion[1].

While distributed representations of words are commonly used to find semantically similar words, they do not straightforwardly provide a way to distinguish more fine-grained semantic information, such as hypernymy, co-hyponymy and meronymy relationships. This deficiency has attracted substantial attention in the literature, and with regard to the task of hypernymy detection, both unsupervised approaches (Hearst, 1992; Weeds et al., 2004; Kotlerman et al., 2010; Santus et al., 2014; Rimell, 2014; Nguyen et al., 2017; Chang et al., 2018) and supervised approaches (Weeds et al., 2014; Roller et al., 2014; Roller and Erk, 2016; Shwartz et al., 2016; Vulić and Mrkšić, 2018; Rei et al., 2018; Kamath et al., 2019) have been proposed.

Supervised methods have, however, been severely hampered by a lack of adequate training data. Not only has a paucity of labelled data been an obstacle in the adoption of deep neural networks and other more complex supervised methods, but two compounding problem-specific issues have been identified. First, there is a need to avoid lexical overlap between the training and test sets in order to avoid the lexical memorisation problem (Weeds et al., 2014; Levy et al., 2015a), where a supervised method simply learns the relationships between lexemes rather than generalising to their distributional features. Second, the performance of classifiers given just the hypernym word (at training

---

[1] See e.g. `https://www.voguebusiness.com/technology/taxonomy%2Dis%2Dthe%2Dnew%2Dfashion%2Dtech%2Dessential%2Dthe%2Dyes` or `https://fashionbrain-project.eu/fashion%2Dtaxonomy/`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1034–1048
April 19 - 23, 2021.

and testing) has been shown to be almost as good as performance given both words (Weeds et al., 2014; Shwartz et al., 2017). This suggests that classifiers are learning the distributional features that make something a more general term or a more specific term. Our conjecture is that in order to learn the more complex function, more complex machinery, and hence more labelled data is required.

In computer vision or speech recognition, it is common to use data augmentation to increase the size of the training set (Shrivastava et al., 2017; Park et al., 2019). The idea is that there are certain transformations of the data under which the class label remains invariant. For example, rotating an image does not change whether that image contains a face or not. By providing a supervised classifier with rotated examples, it can better generalise.

In this work, we consider the use of linguistic transformations to augment existing datasets for hypernymy detection. The challenge is to identify transformations that can be applied to the representations of two words that are known to be in a hypernym relationship, such that the entailment relation still holds between the transformed representations. We propose two ways to achieve this.

Our first augmentation technique is based on the hypothesis that lexical entailment is transitive and therefore invariant under certain compositions. For example, if A entails B and B entails C then A also entails C. Suitable candidates for A can be found by composing common intersective adjectives with the noun B. For example, if we know that *car* entails *vehicle*, then we can augment the dataset with *fast car* entails *car* and *fast car* entails *vehicle*.

Our second augmentation technique is based on the hypothesis that lexical entailment is invariant within a certain threshold of similarity. If A entails B, A' is very similar to A and B' is very similar to B then A' will also entail B'. In order to obtain vectors which are sufficiently similar to the words in the training data, we apply generative adversarial networks (GANs) to create realistic-looking synthetic vectors, from which we choose the most similar to the words in the training data.

We evaluate the proposed techniques on three hypernymy detection datasets. The first two are standard benchmark tasks in this area (Weeds et al., 2014; Baroni et al., 2012), both of which are generated from WordNet (Fellbaum, 1998). However, since many of the approaches to hypernmy classification involve vector space models which have been specialised using the entirety of WordNet, we need to guard against the danger that evaluations are simply measuring how well WordNet has been encoded, rather than how well the general hypernymy relationship has been learned. In light of this, we introduce a new dataset (that we call **HP4K**) which does not rely on WordNet in its construction.

We evaluate our two data augmentation techniques against two methods for increasing the size of the training data which rely on finding or mining more non-synthetic examples. First, we consider the extraction of additional examples from WordNet. Second, we consider extracting examples automatically from a Wikipedia corpus using Hearst Patterns (Hearst, 1992). This provides us with what one would expect to be an upper bound on what we might reasonably expect to achieve with a similar amount of synthetic examples generated using our data augmentation techniques.

Our contributions are thus threefold. First, we have identified two novel data augmentation techniques for the task of hypernymy detection which have the potential to generate almost limitless quantities of synthetic data. Second, we show, rather surprisingly, that adding synthetic data is more effective than adding non-synthetic data in almost all cases. Third, we release a new benchmark evaluation dataset for the lexical entailment task that is not dependent on WordNet.

2 Related Work

Data augmentation has recently become a very popular research topic in NLP and has successfully been applied in machine translation systems (Sennrich et al., 2016; Fadaee et al., 2017; Wang et al., 2018; Li et al., 2019; Tong et al., 2019; Matos Veliz et al., 2019; Xia et al., 2019; Gao et al., 2019; Li and Specia, 2019; Liu et al., 2019), but also for tasks such as relation extraction (Can et al., 2019; Yan et al., 2019), text classification (Wei and Zou, 2019), or natural language inference (Kang et al., 2018; Junghyun et al., 2020). Most similar to our usage of GANs for data augmentation is the proposal of Kang et al. (2018) who leverage a GAN-based setup together with WordNet for data augmentation for natural language inference.

While to the best of our knowledge this work represents the first application of data augmentation for lexical entailment, a number of alternative approaches have been proposed. Most proposals rely on supervised methods for injecting an external source of knowledge into distributional representations. Starting with *retro-fitting* (Faruqui et al., 2015), vector-specialization methods modify existing representations to embed desired features (Vulić and Mrkšić, 2018; Rei et al., 2018; Kamath et al., 2019; Glavaš and Vulić, 2019).

3 Data Augmentation Strategies for Hypernymy Detection

Given a labelled dataset $\mathcal{D}_{\mathcal{X}}$ of triples $\langle x^{(i)}_{hypo}, x^{(i)}_{hyper}, y^{(i)}\rangle$, where $x^{(i)}_{hypo}, x^{(i)}_{hyper} \in \mathcal{X}$ and $y^{(i)} \in \{0, 1\}$, we define data augmentation as adding additional hyponym-hypernym triples $\langle x'^{(j)}_{hypo}, x'^{(j)}_{hyper}, y'^{(j)}\rangle$ coming from an automatically generated augmentation set $\mathcal{A}_{\mathcal{X}'}$, such that $x'^{(j)}_{hypo}, x'^{(j)}_{hyper} \in \mathcal{X}'$ and $y'^{(j)} \in \{0, 1\}$, to the existing training set of $\mathcal{D}$. We ensure that the data augmentation does not introduce any lexical overlap with the existing test set, i.e. $\mathcal{X} \cap \mathcal{X}' = \emptyset$.

Data augmentation strategies in NLP can roughly be divided into two categories: linguistically grounded augmentation and artificial augmentation. In the former, which has been the dominant paradigm in NLP, any additional instances that are added to a training set have an actual surface form representation, i.e. the data points correspond to actual words or sentences (Kim et al., 2019; Kumar et al., 2019; Gao et al., 2019; Andreas, 2020; Croce et al., 2020). The latter adds instances that are fully or partly artificial, meaning they do not correspond to any words or sentences. In this work we propose methods for both categories, data augmentation via distributional composition adds data points grounded in real language to a training set, and data augmentation based on GANs infers plausible points in latent space, which however, do not correspond to any real linguistic objects.

Furthermore, we distinguish between *data augmentation* and *dataset extension*, where in the former case we only leverage knowledge from the existing dataset and in the latter case we rely on expanding the training set with additionally mined hyponym-hypernym pairs. Below, we discuss two ways of augmenting and two ways of extending a training set. We make use of a cleaned October 2013 Wikipedia dump (Wilson, 2015) as reference corpus to determine word and bigram frequencies.

Distributional Composition based Augmentation. We take a modified noun as being in a hypernymy relation with the unmodified noun. For example, we treat the pairs $\langle$*fast car*, *car*$\rangle$ and $\langle$*car*, *vehicle*$\rangle$ as expressing the same semantic relation when the modifier-noun compound is composed with an intersective composition function.

We focus on adjective-noun (AN) and noun-noun (NN) compounds, extracted from our reference corpus where each AN or NN phrase occurred at least 50 times. We filtered pairs with non-subsective adjectives using a wordlist from Nayak et al. (2014)[2].

We consider two strategies for automatically constructing positive hyponym-hypernym pairs: simple positive cases such as $\langle$*small dog*, *dog*$\rangle$ or $\langle$*fast car*, *car*$\rangle$; and gapped positive cases that mimic the transitivity of hypernym relations, where we pair the hypernym of an existing hyponym-hypernym pair with a compound hyponym. For example if $\langle$*car*, *vehicle*$\rangle$ is in the training data, we combine *car* with one of its modifiers to create the pair $\langle$*fast car*, *vehicle*$\rangle$.

We construct negative pairs from the simple positive cases using two strategies: creating compositional co-hyponyms such as $\langle$*fast car*, *red car*$\rangle$, where we keep the head noun fixed and pair it with two different modifiers; and creating perturbed simple positive examples, such as $\langle$*small dog*, *cat*$\rangle$ where we select the incorrect hypernym (e.g. *cat*) from the n most similar nouns to the composed hyponym (e.g. *dog*). We apply the same methodology to the perturbed gapped positive examples, replacing the correct hypernym with a noun from the top n neighbours of the compositional hyponym's head noun. For example, given a positive pair such as $\langle$*dog*, *animal*$\rangle$, this would result in negative examples such as $\langle$*small dog*, *vehicle*$\rangle$, where the hyponym *dog* is paired with a modifier and the hypernym *animal* is replaced with one of its neighbours, in this case, *vehicle*.

In neural word embeddings, an additive composition function approximates the intersection of the corresponding feature spaces (Tian et al., 2017), hence by creating positive pairs such as $\langle$*small dog*, *dog*$\rangle$, we encode the distributional inclusion hypothesis (Weeds et al., 2004; Geffet and Dagan, 2005) in the augmentation set.

GAN based Augmentation. We create an augmentation set using Generative Adversarial Networks (Goodfellow et al., 2014). GANs consist of two model components — the generator and the discriminator — which are typically implemented

---

[2] In preliminary experiments we did not find that filtering non-subsective adjectives had much of an effect, but decided to move forward with the filtered data nonetheless.

as neural networks. The generator's task is to create data that mimics the distribution of the original data, while the discriminator's task is to distinguish between data coming from the real distribution and synthetic data coming from the generator. Both components are trained jointly until the generator succeeds in creating realistic data. Using GANs for data augmentation has been shown to be a successful strategy for a number of computer vision tasks (Shrivastava et al., 2017; Frid-Adar et al., 2018; Neff, 2018). Our goal is to create synthetic hyponym-hypernym pairs that are similar to real examples. Unlike most other scenarios involving GANs for NLP tasks, our generated vectors do not need to correspond to actual words.

For our model - *GANDALF*[3] - we used a list of ≈40K nouns for which we had vector representations as the "real" data input to *GANDALF*, and sampled the synthetic vectors from a Gaussian distribution, optimising a binary cross-entropy error criterion for the generator and the discriminator, which are both simple feedforward networks with a single hidden layer. We provide *GANDALF*'s full model details in Appendix A. As an additional quality check for the generated vectors, we tested whether a logistic regression classifier could distinguish the synthetic and non-synthetic vectors. Typically, the accuracy of the classifier was between 0.55-0.65, meaning the classifier is barely able to distinguish between "real" vectors and generated ones.

Figure 1 illustrates the training loop of *GANDALF* as well as the selection process for constructing an augmented training set. Essentially, once *GANDALF* has been trained, the generator is used to create a large collection[4] of synthetic noun vectors. To augment a dataset, $\mathcal{D}_{\mathcal{X}}$, for each triple, $\langle x_{hypo}, x_{hyper}, y\rangle \in \mathcal{D}_{\mathcal{X}}$ we find the n synthetic vectors most similar to x_{hypo} and the n synthetic vectors most similar to x_{hyper} and for each of the n^2 synthetic vector pairs, $\langle x'_{hypo}, x'_{hyper}\rangle$, we create the triple $\langle x'_{hypo}, x'_{hyper}, y\rangle$. The augmented training set is formed by randomly sub-sampling this set of triples.

WordNet based Extension. WordNet (Fellbaum, 1998) is a large manually curated lexical resource, covering a wide range of lexical relations between words, where groups of semantically similar words form "synsets"[5]. For each synset we extract all hypernyms and hyponyms of a given lexeme, and add it as a positive hyponym-hypernym pair if the original lexeme and any extracted hypernym/hyponym occurs at least 30 times in our reference corpus.

We construct negative pairs based on distributional similarity, where we calculate the pairwise cosine similarities between all lexemes in the positive set. Subsequently we use all antecedent (LHS) lexemes from the extracted positive pairs and select the top n most similar words for each antecedent as negative examples[6].

Pattern based Extension. Hearst Patterns (Hearst, 1992) are textual patterns such as *a car **is-a** vehicle* and can be automatically mined from text corpora in an unsupervised way. This has recently been shown to deliver strong performance on the hypernymy detection task (Roller et al., 2018). In this work, we leverage Hearst Patterns to mine additional hyponym-hypernym pairs in order to extend a training set. We treat any extracted noun pairs as additional positive examples and create the negative pairs in the same way as for the WordNet-based approach above.

4 Experiments

We evaluate our models on the datasets **Weeds** (Weeds et al., 2014) and **LEDS** (Baroni et al., 2012): well-studied and frequently used benchmarks for the hypernymy detection task (Roller et al., 2014; Vilnis and McCallum, 2014; Roller and Erk, 2016; Carmona and Riedel, 2017; Shwartz et al., 2017). Since both datasets use WordNet during construction, this can give rise to a bias in favour of those models that also make use of WordNet. To address this concern, we have created a new entailment dataset, **HP4K**, that makes use of Hearst Patterns, and is manually annotated, thereby avoiding the use of WordNet.

Weeds: The dataset is based on nouns sampled from WordNet where each noun had to occur at least 100 times in Wikipedia, and its predominant sense had to account for more than 50% of the occurrences in SemCor (Miller et al., 1993). We use the predefined split of Weeds et al. (2014), that avoids any lexical overlap between the training and evaluation sets. The split contains 2012 examples in the training set, evenly balanced between positive and negative hyponym-hypernym pairs, and

---

[3] **GAN**-based **D**ata **A**ugmentation for **L**exical in**F**erence.

[4] We would typically create half a million synthetic nouns.

[5] We use the API provided by NLTK (Loper and Bird, 2002), using WordNet 3.0.

[6] Ensuring we don't accidentally add any real positive pairs.

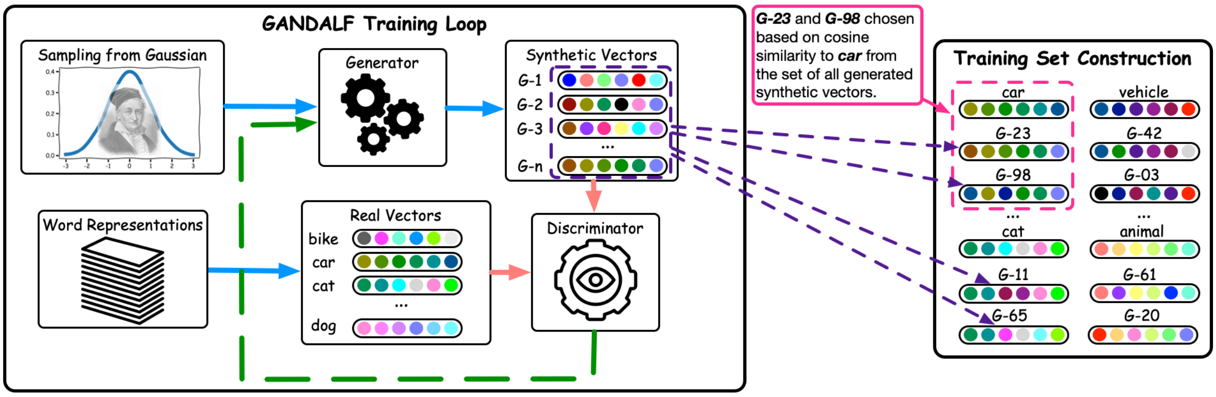


Figure 1: The left panel shows the training loop for *GANDALF* where synthetic noun vectors are generated from Gaussian noise. The discriminator's task is to distinguish between the generated synthetic vectors and real word representations from word2vec or HyperVec. The dashed line indicates error backpropagation, on the basis of the discriminator's current performance, through to the generator. After a final set of synthetic vectors has been generated with *GANDALF*, we choose the top n most similar synthetic vectors to a real representation and add these with the same label as the original hyponym-hypernym pair from an existing training set (e.g. $\langle cat, animal \rangle$) to the augmented training set (right panel).

502 examples in the evaluation set.

LEDS: The dataset consists of 2770 examples, evenly balanced between positive and negative hyponym-hypernym pairs. The positive examples are based on direct hyponym-hypernym relationships from WordNet and the negative examples are based on a random permutation of the hypernyms of the positive pairs. As there is no predefined training/evaluation split, we make use of the 20-fold cross-validation methodology of Roller and Erk (2016) that avoids any lexical overlap between training and evaluation sets.

HP4K: We extracted Hearst Patterns from our reference Wikipedia corpus and randomly selected 4500 unigram pairs. Subsequently, we manually annotated each pair according to whether it constitutes a correct hyponymy-hypernymy relation or not. The labelling was carried out by 4 experienced annotators — all domain experts, familiar with the problem of hypernymy detection. We then split up the annotators in two teams, with each team annotating one half of the dataset. The initial round of annotations resulted in a Cohen's κ score of 0.714, indicating substantial agreement (Viera and Garrett, 2005). Conflicts were resolved on a cross-team basis such that team A would resolve team B's annotation conflicts and vice-versa.

During annotation we noticed that positive pairs typically fall into one of two categories. Either they were "true" subtype-supertype relations, such as $\langle dog, animal \rangle$, or they were individual-class relationships where the hyponym is typically a named entity and represents a *specific* instance of the more general class, as for example in $\langle Nirvana, band \rangle$. Negative pairs were of a more diverse nature and included a range of different relations, such as co-hyponyms, meronyms or reverse hyponym-hypernyms. Negative pairs can also be comprised of two random nouns or two nouns without any semantic relation due to some amount of noise in extracting candidates solely on the basis of Hearst Patterns. Table 1 shows positive and negative examples from the dataset.

Pair	**Relationship**	**Label**
$\langle dog, animal \rangle$	hyponymy-hypernymy (Subtype)	*True*
$\langle Nirvana, band \rangle$	hyponymy-hypernymy (Instance)	*True*
$\langle beef, stew \rangle$	meronymy	*False*
$\langle pedestrian, road \rangle$	topical relatedness	*False*
$\langle chemical, adenosine \rangle$	reverse hyponymy-hypernymy	*False*
$\langle cherry, plum \rangle$	co-hyponymy	*False*
$\langle form, situation \rangle$	none	*False*

Table 1: Examples from our proposed **HP4K** dataset.

HP4K consists of 4369 pairs with a class distribution of 45:55 (positive : negative). Subsequently we split the dataset into a training and evaluation set, ensuring that there is no lexical overlap between the two sets. This resulted in a training set of size 3426 and an evaluation set of size 943[7].

4.1 Models

We conduct experiments with two distributional vector space models, word2vec (Mikolov et al., 2013) and HyperVec (Nguyen et al., 2017). HyperVec is based on word2vec's skip-gram architecture and leverages WordNet to optimise the word representations for the hypernym detection task. Hierarchical information is encoded in the norm of

---

[7] All resources are available from `https://github.com/tttthomasssss/le-augmentation`.

the learned vectors, such that lexemes higher up in the hypernymy hierarchy have larger norms than lexemes in lower parts.

For word2vec we use the 300-dimensional pre-trained Google News vectors[8] and for HyperVec we trained 100-dimensional embeddings on a October 2013 Wikipedia dump (Wilson, 2015), using the recommended settings of Nguyen et al. (2017), as our augmentation sets contained many words that were OOV in the pre-trained HyperVec vectors[9].

In our experiments, we consider a supervised scenario where a classifier predicts a hyponym-hypernym relation between two given word embeddings. We use two different models as classifier a logistic regression classifier (LR), and a 3-layer feedforward neural network (FF). In both cases, the classifier takes the aggregated hypothesised hyponym-hypernym pair as input and predicts whether the pair is in a hyponym-hypernym relation. We report a detailed overview of the model parameterisation in Appendix A.

The two models share the same procedure for aggregating the word embeddings of the hypothesised hyponym-hypernym pair. For data augmentation based on distributional composition, we use vector averaging as composition function, which gave substantially better performance than addition in preliminary experiments.

4.2 Results

For the FF network, we performed 10-fold cross-validation on the **Weeds** and **HP4K** training sets. As our evaluation for **LEDS** is based on a 20-fold cross-valiation split, rather than a pre-defined training/evaluation split as for **Weeds** and **HP4K**, the same procedure for hyperparameter tuning is not straightforwardly applicable without exposing the model to some of the evaluation data. However, we found that the top parameterisations for **Weeds** and **HP4K** were quite similar and therefore applied hyperparameters to the FF model for **LEDS** that performed well in 10-fold cross-validation on **Weeds** *and* **HP4K**. For data augmentation and dataset extension, we consider the following amounts of additional data: $\{0.2\text{K}, 1\text{K}, 2\text{K}, 4\text{K}, 10\text{K}, 20\text{K}, 40\text{K}\}$. All augmentation sets are balanced between positive and negative pairs.

Figure 2 shows the increase in absolute points of accuracy for the LR and FF model, as well as both vector spaces, averaged across all datasets. While in total data augmentation as well as dataset extension has a positive impact, the gains are larger for the FF model, suggesting that a higher capacity model is necessary to more effectively leverage the additional information from the augmentation source. Furthermore, before starting our experiments we exptected that extending an existing dataset with WordNet represents an upper bound on performance, given that WordNet is manually annotated and curated. However in our experiments we found that data augmentation by either distributional composition or by using *GANDALF* remarkably *surpassed* performance of the WordNet-based extension technique regularly.

The effect of data augmentation and dataset extension in absolute points of accuracy on each dataset individually for the FF model is shown in Figure 3. It highlights consistent improvements across the board with only a single performance degradation in the case of extending the LEDS dataset with Hearst Patterns when using HyperVec-based word representations. The results per dataset for the LR model are presented in Appendix A and show that the LR model is less effective in leveraging the augmented data, causing more frequent performance drops. This suggests that models with more capacity are able to make more efficient use of additional data and are more robust in the presence of noise which is inevitably introduced by automatic methods.

Table 2 compares our FF model using word2vec embeddings with all proposed techniques for augmenting or extending a dataset. Our techniques are able to outperform a non-augmented model by 4-6 points in accuracy, representing a relative error reduction of 14%-26%. While the primary objective in this work is to improve an existing model setup with data augmentation, our augmented models compare favourably with previously published results.[10] In general, data augmentation by distributional composition or by *GANDALF* overcomes two key weaknesses of simply extending a dataset with more data from WordNet or Hearst Patterns. First, many of the hyponym-hypernym pairs we mined from WordNet contain low-frequency words,

---

[8] Available from: `https://code.google.com/archive/p/word2vec/`.

[9] We used the HyperVec code from `www.ims.uni-stuttgart.de/data/hypervec`.

[10] We note that due to the use of different performance metrics and cross-validation splits, direct model-to-model comparisons are difficult on the LEDS and Weeds datasets. Thus we only compare to approaches that use the same evaluation protocol as we do.

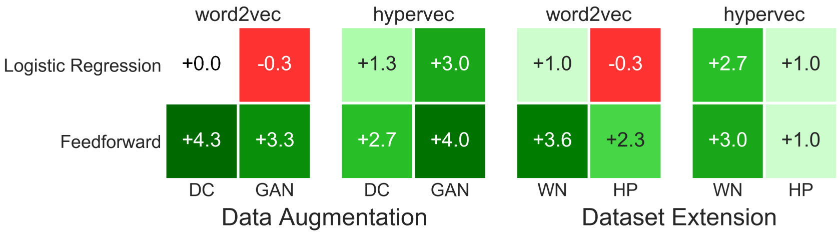


Figure 2: Effect of data augmentation and dataset extension in absolute points of accuracy averaged across all datasets over the same model without augmentation or extension. The 2 heatmaps on the left are based on data augmentation (DC=Distributional Composition, GAN=*GANDALF*). The 2 heatmaps on the right use dataset extension (WN=WordNet, HP=Hearst Patterns).

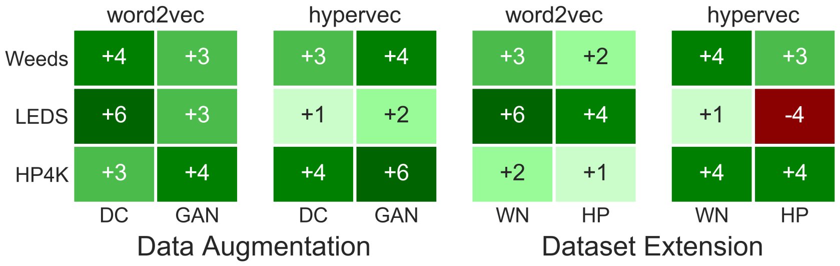


Figure 3: Effect of data augmentation and dataset extension in absolute points of accuracy on all datasets for the FF model.

Model	**Weeds**	**LEDS**	**HP4K**
No Augmentation	0.72	0.77	0.67
Distributional Composition	**0.76**	**0.83**	0.70
GANDALF	0.75	0.80	**0.71**
WordNet Extension	0.75	**0.83**	0.69
Hearst Patterns Extension	0.74	0.81	0.68
Weeds et al. (2014)	0.75	-	-
Carmona and Riedel (2017)	0.63	0.81	-

Table 2: Accuracy scores for the data augmentation and the two dataset extension strategies in comparison to the same FF model without any augmentation or extension.

which may have poor representations in our vector space models. Second, while using Hearst Patterns typically returned higher frequency words, the retrieved candidates frequently did not represent hyponymy-hypernymy relationships.

5 Analysis

The concrete amount of data augmentation, i.e. the number of additional hyponym-hypernym pairs that are added to the training set, represents a tuneable parameter. Figure 4 shows the effect of varying amounts of data augmentation for the FF model, using word2vec representations, across all datasets. We note that all amounts of additional augmentation data share the same quality, i.e. it is not the case that a smaller augmentation set consists of "better data" or contains less noise than a larger set. For the **Weeds** and **LEDS** datasets, peak performance is typically achieved with smaller amounts of additional data, whereas for the **HP4K** dataset optimal performance is achieved with larger amounts of augmentation data. One explanation for the different augmentation characteristics of the **HP4K** dataset in comparison to the other two datasets is its independence of WordNet during the development of the dataset.

5.1 Data Augmentation in Space

In order to visualise what area of the vector space the *GANDALF* vectors and the composed vectors inhabit, we created a t-SNE (van der Maaten and Hinton, 2008) projection of the vector spaces in Figure 5. For the visualisation we produced the nearest neighbours of standard word2vec embeddings and augmentation embeddings for 5 exemplary words and project them into the same space. Figure 5 shows that the generated augmentation points, marked with an "x", fit in with the real neighbours and do not deviate much from the "natural" neighbourhood of a given word. *GANDALF* vectors typically inhabit the edges of a neighbour cluster, whereas composed vectors are frequently closer to the cluster centroid. Table 3 lists the nearest neighbours for the example words. For word2vec and the composed augmentation vectors, we simply list the nearest neighbours of each query word. For *GANDALF* we list the nearest neighbours of the generated vector that correspond to actual words. For example, if the vector *GANDALF-234* is closest to the representations of *sugar*, *GANDALF-451* and *mountain*, we only list *sugar* and *mountain* as

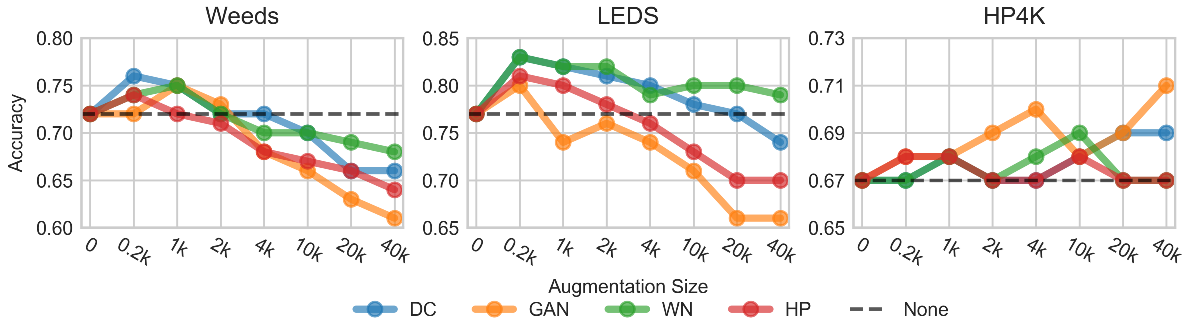


Figure 4: Impact of different amounts of data augmentation.

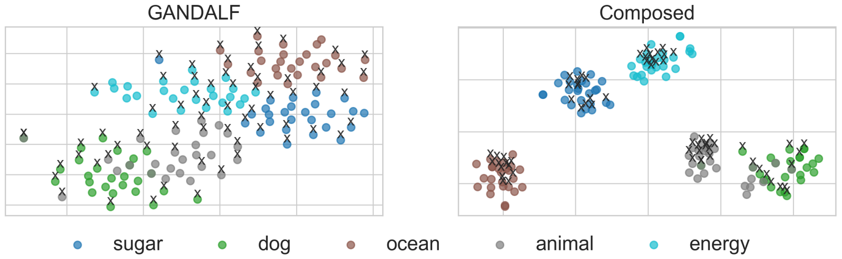


Figure 5: t-SNE visualisation of the data augmentation spaces. Data points marked with "x" denote the representation as coming from the data augmentation set.

neighbours of *GANDALF-234*.

The composed neighbours for each word are typically closely related to the original query, e.g. *raw sugar* for *sugar*, or *zoo animal* for *animal*. The *GANDALF* neighbours on the other hand have a much weaker association with the query word, but are frequently still related to it *somehow* as in the example of *akpeteshie*, which is a spirit on sugar cane basis, as a neighbour for *sugar*.

5.2 Data Augmentation as Regularisation

In the past, a prominent criticism of distributional methods for hypernymy detection was that such models were found to frequently identify features of a prototypical hypernym in the distributional representations, rather than being able to dynamically focus on the relevant features that are indicative of a hypernymy relation for a specific pair of words (Weeds et al., 2014; Levy et al., 2015b). We therefore briefly investigate whether data augmentation can be used as a regularisation mechanism that helps prevent models from overfitting on prototypical hypernym features.

Table 4 shows the results on the **Weeds** dataset using a hypernym-only FF model with word2vec representations, in comparison to the same model variant that makes use of the hyponym and the hypernym. Ideally, we would hope to see weak performance for the hypernym-only and strong performance on the full model. This would indicate that the classifier does not rely on prototypical features in the hypernym, but is able to focus on specific features in a given hyponym-hypernym pair. For data augmentation by distributional composition there appears to be a correlation between the performance of the hypernym-only and the full model, i.e. a stronger model on the whole dataset also results in better performance for the hypernymy-only model. Hence augmentation by distributional composition might not be effective in helping the model to generalise in its current form. For augmentation with *GANDALF* however, performance for the full model improves, while performance of the hypernym-only model slightly drops, suggesting that the evoked *GANDALF* representations have a regularisation effect, while also improving generalisation. Hence, a fruitful avenue for future work will be further leveraging data augmentation for regularisation.

6 Conclusion

In NLP, in contrast to computer vision, data augmentation has not been applied as standard due to the apparent lack of universal rules for label-invariant language transformations.

We have considered the problem of hypernymy detection, and proposed two novel techniques for data augmentation. These techniques rely on semantic rules rather than an external knowledge source, and have the potential to generate almost limitless synthetic data for this task. We demonstrate that these techniques perform better than

Word	**word2vec Neighbours**	***GANDALF* Neighbours**	**Composed Neighbours**
sugar	refined sugar, cane sugar, turbinado, cocoa, sugars	cmh rawalpindi, prescribed antipsychotic medication, sugar, mumtaz bhutto, akpeteshie	raw sugar, white sugar, brown sugar, sugar price, sugar industry
dog	dogs, puppy, pit bull, pooch, cat	ellis burks, sniffing glue, microchip implants, liz klekamp, cf rocco baldelli	pet cat, dog fighting, cat breed, rat terrier terrier breed
ocean	sea, oceans, pacific ocean, atlantic ocean, oceanic	pacific ocean, heavily vegetated, alaska aleutian, seagrasses, plutoid	ocean basin, pacific ocean, shallow sea, sea fish, sea mammal
animal	animals, animal welfare, dog, pet, cats	animals, pet, hallway feeds, poop scooping, panhandle animal welfare	first animal, adult animal, zoo animal, different animal, animal organization
energy	renewable energy, enery, electricity, enegy, fossil fuel	radial velocity measurements, stopped, renewable energy, bicycle advisory, steinkuehler	energy efficiency, solar energy, state energy, food energy, energy company

Table 3: Nearest neighbours for word2vec, GAN vectors and composed vectors.

Augmentation	**Hypernym-Only**	**Full**
No Data Augmentation	0.59	0.72
Distributional Composition (size=100)	0.60	**0.74**
Distributional Composition (size=500)	**0.57**	0.71
GANDALF (size=500)	**0.58**	**0.75**
GANDALF (size=1000)	0.60	0.73

Table 4: Accuracy for the hypernym-only and full models on the **Weeds** dataset with no, DC or GAN augmentation.

extending the training set with additional non-synthetic data, drawn from an external knowledge source in most cases. Our results are consistent across evaluation benchmarks, word vector spaces and classification architectures. We have also shown that our approach is effective even when the word vector space model has already been specialised for hypernymy detection.

Since WordNet is widely used as a source of information about semantic relations, we have proposed a new evaluation benchmark that is independent of WordNet. Whilst results are lower across the board on this dataset, suggesting that it is more difficult than the others, we see the same pattern of increasing performance with a more complex classifier and the use of data augmentation.

Future work includes leveraging data augmentation for more complex models and the extension of our approach to a multilingual setup as well as domains with a more specialised vocabulary such as Healthcare or Fashion.

Acknowledgements

Part of this research was supported by EPSRC grant no. 2129720: *Composition and Entailment in Distributed Word Representations*. We thank Mark Steedman for valuable feedback and insightful discussions that had substantial impact on the quality the paper. We also thank Nicola De Cao, without whose help *GANDALF* wouldn't have been able to defeat the Balrog of mediocre performance. Lastly, we thank several mini-batches of reviewers for their ~~stochastic gradients~~ helpful comments that iteratively improved the paper, rejection after rejection, until it *finally* converged towards acceptance.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Gintaras Barisevičius, Martin Coste, David Geleta, Damir Juric, Mohammad Khodadadi, Giorgos Stoilos, and Ilya Zaihrayeu. 2018. Supporting digital healthcare services using semantic web technologies. In *The Semantic Web – ISWC 2018*, pages 291–306. Springer International Publishing.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France. Association for Computational Linguistics.

Duy-Cat Can, Hoang-Quynh Le, Quang-Thuy Ha, and Nigel Collier. 2019. A richer-but-smarter shortest dependency path with attentive augmentation for relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2902–2912, Minneapolis, Minnesota. Association for Computational Linguistics.

Vicente Ivan Sanchez Carmona and Sebastian Riedel. 2017. How well can we predict hypernyms from word embeddings? a dataset-centric analysis. In *Proceedings of the European Chapter of the Association for Computational Linguistics*.

Haw-Shiuan Chang, Ziyun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional inclusion vector embedding for unsupervised hypernymy detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 485–495, New Orleans, Louisiana. Association for Computational Linguistics.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan. 2018. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 289–293.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

M. Geffet and I. Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 107–114. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2019. Generalized tuning of distributional word vectors for monolingual and cross-lingual lexical entailment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4824–4830, Florence, Italy. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the International Conference on Computational Linguistics*.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org.

Min Junghyun, McCoy R. Thomas, Dipanjan Das, Pitler Emily, and Linzen Tal. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Aishwarya Kamath, Jonas Pfeiffer, Edoardo Maria Ponti, Goran Glavaš, and Ivan Vulić. 2019. Specializing distributional vectors of all words for lexical entailment. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 72–83, Florence, Italy. Association for Computational Linguistics.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. 2018. AdvEntuRe: Adversarial training for textual entailment with knowledge-guided examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2418–2428, Melbourne, Australia. Association for Computational Linguistics.

Hwa-Yeon Kim, Yoon-Hyung Roh, and Young-Kil Kim. 2019. Data augmentation by data noising for open-vocabulary slots in spoken language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 97–102, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Special Issue of Natural Language Engineering on Distributional Lexical Semantics*, 4(16):359–389.

Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. 2019. A closer look at feature space data augmentation for few-shot intent classification. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015a. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015b. Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, Colorado.

Guanlin Li, Lemao Liu, Guoping Huang, Conghui Zhu, and Tiejun Zhao. 2019. Understanding data augmentation in neural machine translation: Two perspectives towards generalization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5689–5695, Hong Kong, China. Association for Computational Linguistics.

Zhenhao Li and Lucia Specia. 2019. Improving neural machine translation robustness via data augmentation: Beyond back-translation. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 328–336, Hong Kong, China. Association for Computational Linguistics.

Angli Liu, Jingfei Du, and Veselin Stoyanov. 2019. Knowledge-augmented language model and its application to unsupervised named-entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1142–1150, Minneapolis, Minnesota. Association for Computational Linguistics.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, pages 63–70. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Claudia Matos Veliz, Orphee De Clercq, and Veronique Hoste. 2019. Benefits of data augmentation for NMT-based text normalization of user-generated content. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 275–285, Hong Kong, China. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the Arpa Workshop on Human Language Technology*, pages 303–308.

Neha Nayak, Mark Kowarsky, Gabor Angeli, and Christopher D. Manning. 2014. A dictionary of nonsubsective adjectives. Technical Report CSTR 2014-04, Department of Computer Science, Stanford University.

Thomas Neff. 2018. Data augmentation in deep learning using generative adversarial networks. Master's thesis, Graz University of Technology.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical embeddings for hypernymy detection and directionality. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 233–243, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *CoRR*, abs/1904.08779.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Marek Rei, Daniela Gerz, and Ivan Vulic. 2018. Scoring lexical entailment with a supervised directional similarity network. In *ACL*.

Laura Rimell. 2014. Distributional lexical entailment by topic coherence. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 511–519, Gothenburg, Sweden. Association for Computational Linguistics.

Stephan Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet selective: Supervised distributional hypernymy detection. In *Proceedings of the International Conference on Computational Linguistics*.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2163–2172, Austin, Texas. Association for Computational Linguistics.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–363. Association for Computational Linguistics.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc.

Enrico Santus, Alessandro Lenci, and Qin Lu. 2014. Chasing hypernyms in vector space with entropy. In *Proceedings of the European Chapter of the Association for Computational Linguistics*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. 2017. Learning from simulated and unsupervised images through adversarial training. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2242–2251.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 65–75, Valencia, Spain. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2017. The mechanism of additive composition. *Machine Learning*, 106(7):1083–1130.

Yixuan Tong, Liang Liang, Boyan Liu, Shanshan Jiang, and Bin Dong. 2019. Supervised neural machine translation based on data augmentation and improved training & inference process. In *Proceedings of the 6th Workshop on Asian Translation*, pages 147–151, Hong Kong, China. Association for Computational Linguistics.

Anthony J. Viera and Joanne M. Garrett. 2005. Understanding interobserver agreement: The kappa statistic. *Family medicine*, 37:360–3.

Luke Vilnis and Andrew McCallum. 2014. Word representations via gaussian embedding. *CoRR*, abs/1412.6623.

Ivan Vulić and Nikola Mrkšić. 2018. Specialising word vectors for lexical entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1134–1145. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1015–1021, Geneva, Switzerland.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Benjamin Wilson. 2015. The unknown perils of mining wikipedia. Https://blog.lateral.io/2015/06/the-unknown-perils-of-mining-wikipedia/.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized data augmentation for low-resource translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5786–5796, Florence, Italy. Association for Computational Linguistics.

Jianhao Yan, Lin He, Ruqin Huang, Jian Li, and Ying Liu. 2019. Relation extraction with temporal reasoning based on memory augmented distant supervision. In *Proceedings of the 2019 Conference of* *the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1019–1030, Minneapolis, Minnesota. Association for Computational Linguistics.

A Supplemental Material

A.1 GANDALF Model Details

The generator and discriminator in *GANDALF* are single layer feedforward networks, with tanh activations and a dropout ratio (Srivastava et al., 2014) of 0.3. We used ADAM (Kingma and Ba, 2014) to optimise a binary cross entropy error criterion with a learning rate of 0.0002 and β values of 0.5 and 0.999. We found that *GANDALF* required quite a bit of wizardry to achieve strong performance and we found the website `https://github.com/soumith/ganhacks` very helpful. For example we applied label noise and soft labels (Salimans et al., 2016) and used a batch normalisation layer (Ioffe and Szegedy, 2015), which had the largest impact on model performance. *GANDALF* is implemented in PyTorch (Paszke et al., 2017) and we release our code on `https://github.com/tttthomasssss/le-augmentation`.

A.2 Model Details

For our linear model we use the logistic regrssion classifier implemented in scikit-learn (Pedregosa et al., 2011). Our neural network model is 3-layer feedforward model implemented in PyTorch (Paszke et al., 2017).

We tuned the parameters of the Feedforward neural network by 10-fold cross-validation on the respective training sets, except for **LEDS**, where we chose the parameters on the basis of a model that performed well on the **Weeds** and **HP4K**. Our parameter grid consisted of activation function: {tanh, relu}, dropout: { 0.0, 0.1, 0.3 } and hidden layer sizes, where we considered { 200-200-200, 200-100-50, 200-50-30 } for Hypervec and { 600-600-600, 600-400-200, 600-300-100, 600-200-50} for word2vec. We furthermore considered 3 different aggregation functions: `diff` (Weeds et al., 2014), which simply takes the elementwise difference of the embedding pair; `asym` (Roller et al., 2014) which is the concatenation of the difference and the squared difference of the embedding pair; and `concat-asym` (Roller and Erk, 2016), which is the concatenation of the embedding pair, their difference, and their squared difference. We trained all models for 30 epochs with early stopping and used ADAM with a learning rate of 0.01 to optimise a cross entropy error criterion.

A.3 Results

Figure 6 below shows the effect of data augmentation in terms of points of Accuracy for the logistic regression classifier per vector space model and dataset. Unlike for the higher-capacity feedforward model, data augmentation frequently causes performance to go down for the simpler linear model. This suggests that more complex models are required to fully leverage the additional information from the augmentation sets.

Table 5 below gives an overview over the complete results for both classifier and vector space models, across all datasets. It shows the consistent positive effect of data augmentation on the more complex feedforward model in comparison to the logistic regression classifier, which is less robust to the small amounts of noise that is inevitably introduced by the automatic augmentation algorithm.

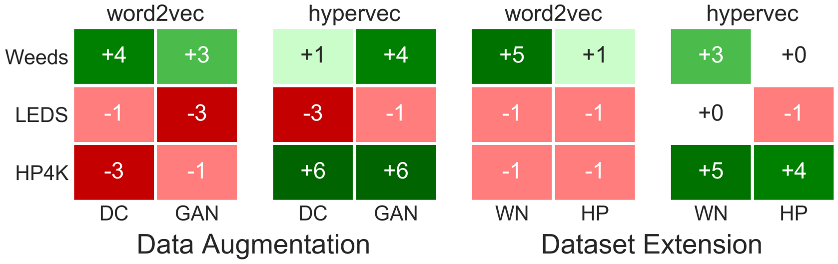


Figure 6: Effect of data augmentation and dataset extension in absolute points of accuracy on all datasets for the LR model.

Model	Weeds					LEDS					HP4K					Vector Space
	None	DC	GAN	WN	HP	None	DC	GAN	WN	HP	None	DC	GAN	WN	HP	
LR	0.69	0.73	0.72	0.74	0.70	0.81	0.80	0.78	0.80	0.80	0.64	0.61	0.63	0.63	0.63	word2vec
FF	0.72	**0.76**	0.75	0.75	0.74	0.77	**0.83**	0.80	**0.83**	0.81	0.67	0.70	**0.71**	0.69	0.68	
LR	0.70	0.71	0.74	0.73	0.70	0.79	0.76	0.78	0.79	0.78	0.63	0.69	0.69	0.68	0.67	hypervec
FF	0.71	0.74	**0.75**	**0.75**	0.74	0.79	0.80	**0.81**	0.80	0.75	0.66	0.70	**0.72**	0.70	0.70	

Table 5: Accuracy scores for the data augmentation strategies (DC and GAN), and the two dataset extension strategies (WN and HP), and the baseline that neither uses augmentation nor extension (None). Boldfaced results denote top performance per vector space and dataset, underlined results denote improved performance in comparison to the baseline without data augmentation.

Few-shot learning through contextual data augmentation

Farid Arthaud [*]
Ecole Normale Supérieure
farid.arthaud@ens.psl.eu

Rachel Bawden [*]
Inria
rachel.bawden@inria.fr

Alexandra Birch
University of Edinburgh
A.Birch@ed.ac.uk

Abstract

Machine translation (MT) models used in industries with constantly changing topics, such as translation or news agencies, need to adapt to new data to maintain their performance over time. Our aim is to teach a pre-trained MT model to translate previously unseen words accurately, based on very few examples. We propose (i) an experimental setup allowing us to simulate novel vocabulary appearing in human-submitted translations, and (ii) corresponding evaluation metrics to compare our approaches. We extend a data augmentation approach using a pre-trained language model to create training examples with similar contexts for novel words. We compare different fine-tuning and data augmentation approaches and show that adaptation on the scale of one to five examples is possible. Combining data augmentation with randomly selected training sentences leads to the highest BLEU score and accuracy improvements. Impressively, with only 1 to 5 examples, our model reports better accuracy scores than a reference system trained with on average 313 parallel examples.

1 Introduction

News agencies dealing with articles in multiple languages often rely on machine translation (MT) to provide an initial translation, which a journalist reworks into a final article. This can involve changing the structure or phrasing, but also correcting (or *post-editing*) mistranslated words or expressions, which can frequently occur when new topics emerge, bringing new vocabulary that has been rarely or never seen in the data used for training. The willingness of a journalist to use MT technology is dependent on the general quality of the models, but also on whether they can learn from the journalist's corrections, to avoid them having to correct the same errors time and time again.

Various strategies have been explored to learn from a journalist's post-edits. One option is to use an automatic post-editing (APE) model trained on the journalist's post-edits. However, state-of-the-art APE systems (Junczys-Dowmunt and Grundkiewicz, 2017) typically require large numbers of post-edits for training, which are rarely available or hard to generate (particularly for low-resource languages). An alternative, commonly used strategy is to fine-tune models to in-domain data, but this is prone to overfitting (Miceli Barone et al., 2017). More advanced ways of continually learning through fine-tuning have been explored, selecting similar training instances based on their similarity with test sentences (Li et al., 2018; Turchi et al., 2017). These methods achieve good results according to automatic MT metrics but can also overfit when training is continued. So far there has been little focus on the speed of adaptation required—so as to minimize the number of human interventions required—and the trade-off between successfully adapting to specific new post-edits and maintaining a good global translation quality.

In this article, we aim to explore in more depth this trade-off between overall translation quality and the ability of the MT model to learn specific novel words in the context of life-long learning of MT from journalistic post-edits. We explore the setup in a simulated few-shot learning scenario, whereby we track the translation performance on specific test words that are removed from the training data and are gradually re-introduced through fine-tuning. To improve the model's capacity to learn these new examples, we explore an approach similar to Turchi et al. (2017) and inspired by a contextual data augmentation technique used for rare word translation (Fadaee et al., 2017) to reduce the number of corrections that need to be seen.

We show in our experiments on the Gujarati-English language pair that it is not only possible

---

[*] Work done at University of Edinburgh

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1049–1062
April 19 - 23, 2021.

to surpass the accuracy of our baseline fine-tuning approach, but also of a reference model which has already seen the new words dozens to hundreds of times during training. However we find that in most cases adaptation to new words comes at a variable cost in BLEU (Papineni et al., 2002), due to overfitting to the new examples. We show that this cost can be kept at a minimum by padding our data-augmentation approach with randomly selected sentences from our training set. The appropriate choice of hyper-parameters is also important for final performance.

Our code is freely available online.[1]

2 Related work

The topic of few-shot learning from post-edits is relatively novel, and we were therefore left with few comparison points. A somewhat similar task that requires quick adaptation is low-resource MT, for which transfer learning (Zoph et al., 2016) and meta-learning (Gu et al., 2018) approaches exist. These techniques generally apply for adaptation from hundreds of thousands of sentences, rather than a dozen available in our scenario; this is because we aim to learn individual new words rather than a whole language or domain.

A widespread technique in the MT literature to adapt a model to new data is fine-tuning, often used for domain adaptation. Turchi et al. (2017) and Li et al. (2018) explore the use of a similarity search in the training corpus in order to fine-tune an MT model before translating a novel sentence or to gradually adapt a model to post-edits. This approach does not apply to our scenario, where new words appear and therefore a similarity search cannot yield sentences containing these new words. Our baseline approach (which we call `finetune`) is present in these works, known as *adaptation a posteriori* in (Turchi et al., 2017) and also appears as *single-sentence adaptation* in (Kothur et al., 2018). A challenge these works report with fine-tuning is overfitting, which we encounter systematically when evaluating our various techniques. Works on fine-tuning also explore several regularization techniques (Simianer et al., 2019) when adapting to new data, which we choose to leave out of our comparison due to added hyper-parameter choices and complexity of implementation in our experiments – we do however believe that future work implementing these techniques could potentially outperform ours.

Fadaee et al. (2017); Kobayashi (2018); Wu et al. (2019) and Gao et al. (2019) explore a similar contextual data augmentation technique, albeit in different scenarios and with different goals. This technique synthesizes new sentences by using sentences from the training set and substituting different words into them. In (Fadaee et al., 2017) the goal is to enhance overall translation performance by focusing on words that appear rarely in the training data, but in our case we are training our system to learn new words which were not in the training set at all, based simply on a ground-truth translation of this new word by a human. Moreover, our technique uses more recent tools and techniques such as the BERT contextual language model (Devlin et al., 2019). Kobayashi (2018) and Wu et al. (2019) also use BERT for contextual data augmentation, but with a goal of improving language model tasks such as sentiment analysis. The constraints for this task are very different; rather than having to produce a translation for augmented data, these approaches have to maintain the sentiment label of the sentences. Gao et al. (2019) work in a similar context to us, but focus on overall translation performance rather than learning new words, and apply contextual data augmentation during the training step, thus removing the challenge of adapting to new data as it becomes available.

Similar to us, Huck et al. (2019) focus on improving the MT of words which are unseen in the training set. They use bilingual lexicons to hypothesize translations for their unseen terms. They find these translations in monolingual target side data and backtranslate them inserting the unseen term. They show that this improves translation performance in the medical domain. However they do not analyze the accuracy of translation of the novel terms, or explore how fast you can learn from very few examples.

3 Lifelong learning from post-edits

MT models are inevitably adapted towards the topics and vocabulary from the time period associated with their training data. In the long term, they therefore struggle to correctly translate novel words and expressions associated with new topics, unless they can be adapted to them. This is particularly a problem in journalism, where current topics and names are constantly changing. A prime example is the

---

[1] `https://gitlab.com/farid-fari/fewshot-learning`

recent COVID-19 pandemic: prior to January 2020, newspapers would contain little to no mention of the words *coronavirus*, *respirators*, *PPE masks* and *hydroxychloroquine*. On top of these topic-specific words, there may also be novel but very frequent expressions that are initially hard to translate, such as *flattening the curve* and *social distancing*, which are likely to be poorly translated and therefore need to be subsequently corrected by the journalist.

We are interested in developing approaches to quickly learn from journalistic post-edits in a way that maintains the general high translation quality of the model. In order to analyze and objectively evaluate our different approaches, we simulate this scenario in a reproducible way by using a publicly available corpus in which we select the rarest words and separate out all sentences containing them: these words will be our *evaluation words*. These words must appear at least a few times in the test set as well in order to enable proper evaluation of adaptation using our approaches. The sentences separated from the training set containing our evaluation words will be used to simulate reference sentences submitted by a journalist for our models to use for adaptation. Finally, the training set with the evaluation words removed will be our *filtered training set*, used for initial training of our model.

Our test set is also made up of publicly available data sets whose sentences also contain the evaluation words we selected. This setup prohibits the use of a development set and a test set due to the dependency of the rare words on the test set: they must appear a minimum number of times in the test set to have a way of evaluating our approaches. In turn, the choice of rare words changes the training set and therefore the models we train. We are therefore in a transductive learning scenario, where our method is adapted to the task we aim to solve.

We also use as a reference point a model which has been trained on the complete (unfiltered) training set, which is a very strong comparison point since it has seen many more occurrences of the rare words we selected than our few-shot models.

4 Our approaches

We propose and evaluate four approaches: `finetune` (our baseline), `randompad`, `augmented` and `half`. The first two approaches only involve fine-tuning whereas the latter two include contextual data augmentation. Each approach works from a given number of reference sentence pairs containing our evaluation words, acting as human-provided references. In our experimental setup, these sentences are chosen randomly among the held-out sentences from the filtered training set.

4.1 Baseline: fine-tuning

Our first approach, which is also our baseline, is naive fine-tuning referred to as *adaptation a posteriori* in (Turchi et al., 2017), and which we refer to as `finetune`. When presented with a set of reference sentence pairs to adapt to, our model is trained from its previous state for a few epochs using the sentence pairs. This approach is especially prone to overfitting as it will see very few sentences, and they all contain a potentially repetitive set of words. To minimize this effect, we choose to fine-tune on a batch of several reference sentence pairs rather than individually for each evaluation word, so as to diversify the data fed to the model.

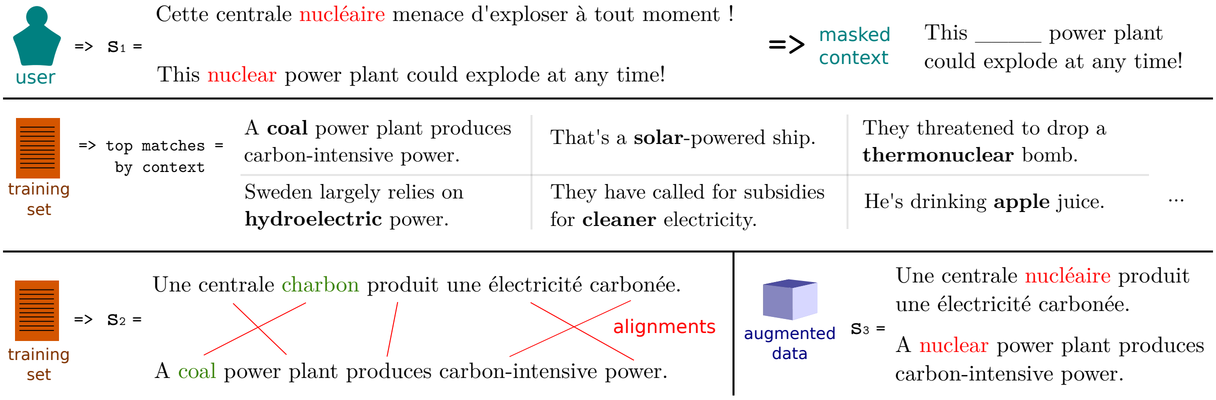
4.2 Padded fine-tuning

The second approach, `randompad`, attempts to minimize the overfitting from `finetune` by introducing variety and generality to the fine-tuning set. This is done by adding randomly chosen sentences from the filtered training set to the reference sentences, which is known as *padding*. These random sentences do not contain the evaluation words and so encourage generalization during fine-tuning.

4.3 Contextual data augmentation

The aim of this approach, similar to the one used by Fadaee et al. (2017) and which we refer to as `augmented`, is to use the reference sentences provided by the journalist to create additional novel sentences containing the evaluation words. If successful, this approach simulates the presence of more reference sentences containing the new words, which accelerates learning, effectively reducing the number of corrections required from the human translator or journalist. More formally, given a reference sentence pair (s_{src}, s_{tgt}) containing an unseen word $w \in s_{tgt}$, the goal is to generate new sentence pairs containing w in the target language, varied enough to be beneficial in fine-tuning.

Our data augmentation process has 2 steps: (i) for each reference sentence pair provided by the journalist containing a word of interest w, find sentences from the training corpus with similar contexts to the one surrounding w, and then (ii) insert w and its translation into the retrieved sentences.



Cette centrale nucléaire menace d'exploser à tout moment !

This nuclear power plant could explode at any time!

This ______ power plant could explode at any time!

A **coal** power plant produces carbon-intensive power.

That's a **solar**-powered ship.

They threatened to drop a **thermonuclear** bomb.

Sweden largely relies on **hydroelectric** power.

They have called for subsidies for **cleaner** electricity.

He's drinking **apple** juice.

Une centrale charbon produit une électricité carbonée.

A coal power plant produces carbon-intensive power.

Une centrale nucléaire produit une électricité carbonée.

A nuclear power plant produces carbon-intensive power.

Figure 1: Contextual data augmentation applied to an example sentence

We illustrate this process in Figure 1, using the French-English language pair to aid readability: the sentence pair s_1 containing the new word *nuclear* is provided to our system by the journalist or translator. We first find sentences in the training set that contain a word (in bold) with a similar context to that of w in s_{tgt}. In each of these sentences, the word in bold could be replaced by *nuclear*, although there are sometimes false positives such as the last sentence: *He's drinking **nuclear** juice*, would not be a good reference sentence for training. Next, we use an alignment tool to replace the aligned source word (e.g. *charbon*) in the selected sentence pair s_2 with the word *nucléaire*, providing a brand new sentence pair s_3 for the reference word *nucléaire-nuclear*.

Finding similar contexts The first step is to find suitable sentences in which we can substitute w and its translation. As in (Wu et al., 2019), we use the BERT contextual language model (Devlin et al., 2019) to provide a contextual representation of w in s_{tgt}, noted v. This feature vector is taken from the second-to-last layer of BERT (usually used to compute vocabulary probabilities through a softmax layer). Given the masked and bidirectional nature of BERT, this contextual representation contains information about the context surrounding w in s_{tgt}, with no prior knowledge of w required by BERT. The advantage of this approach is that truly novel words can be data-augmented using this technique with any pre-trained BERT model.

The context search consists in (i) extracting the feature vector v for w in s_{tgt}, (ii) for each sentence in the filtered training set, randomly selecting a word u in it and computing the corresponding feature vector[2] and (iii) selecting the top k sentences based on the cosine similarity of these feature vectors with v.

This is shown in Figure 1 where 6 sentences have been selected based on their feature vector similarity to s_1. These sentences are selected because the context surrounding the random word chosen in each of them is similar to the context surrounding w in s_1, which generally means the bold words loosely correspond to adjectives describing power plants. Using a masked model means that knowledge of the meaning of w is not required, but only an understanding of the context in s_{tgt}. An additional real use example is provided in Appendix B.

This process significantly differs from Fadaee et al. (2017), which iterates over the training set to find sentences for which the language model gives a high probability of the rare words appearing. Moreover, by using feature vectors rather than raw probability distributions (as in previous works), we capture richer information about the surrounding sentence than only which words are likely to replace w.

Creation of new training examples The second step is to substitute the randomly masked word u—which is *coal* in our example—in the retained sentence pair s_2 with w in the target language and its translation in the source language. In the target language, we simply replace u by w, but the task is harder in the source language since we have no prior information as to which word translates to u, or which word translates to w to replace it with. In our scenario, we assume that the translation in the source language of w, noted w', is known to us – for

---

[2] Comparing all positions in all sentences would lead to the best results, but is inefficient. With sufficient training examples (as is the case here), randomly choosing a single position in each sentence avoids having to to do all computations and still provides varied contexts that can be tested.

example, a human translator providing (s_{src}, s_{tgt}) could identify w' in s_{src}. Having no human translator for our experiments, we train an alignment tool on the complete unfiltered training set and use it to select the words most often aligned to each w from our evaluation words. A different approach could be used in future work, by training the alignment tool on the filtered training set and then re-training it on reference sentences as they are provided, removing the need for human intervention or prior knowledge of w. Note that all alignment steps rely on automatic tools which can introduce some noise in the process. However, we observed satisfactory results, and this allows for the entire process to be automated.

Once the translation w' for w is established, we have to determine which word or words in the source sentence must be replaced by w'. This is done by using an alignment tool trained only on the filtered training set, to compute the aligned word to u. The word aligned to u is replaced by w'; if multiple consecutive words are aligned to u they are all replaced with w', and if no words or non-consecutive words are aligned to u then the sentence is discarded. This is illustrated in Figure 1, where *coal* is successfully aligned to *charbon*, allowing us to replace it with *nucléaire*.

The final result in the example is s_3, a sentence pair which makes sense and will be useful for the translation of *nucléaire* to *nuclear*. The sentences generated by this method will at the very least contain w' and w in roughly aligned source and target positions and the best sentences will have the same quality as human-provided references.

4.4 Padded data augmentation

The `augmented` approach does not benefit from the added generality of the `randompad` approach; it could suffer from overfitting due to the repetition of our evaluation words (and their semantic fields) in the fine-tuning sentences. This is addressed by the `half` approach, designed to overcome potential overfitting in the `augmented` approach. This approach uses half of the synthesized sentences from `augmented` and replaces the other half with random sentences from the filtered training set. This provides more sentences containing our rare words through augmentation but also more generality from the random sentences, aiming to combine the strengths of both approaches.

5 Experiments

5.1 Experimental setup

We use the low-resource Gujarati to English language direction. Our base model is the same as the University of Edinburgh's submission to WMT 2019 (Bawden et al., 2019), except that its training set is filtered to remove sentences containing words previously selected for evaluation. The model is a base transformer model (Vaswani et al., 2017) with 6 encoder and decoder layers, feed-forward dimension 512, 8 transformer heads, and dropout of 0.1. We train using the Marian toolkit (Junczys-Dowmunt et al., 2018) and the Adam optimizer (Kingma and Ba, 2015).[3] The complete training script with additional parameters can be found on the webpage for the University of Edinburgh's WMT19 submission.[4] We use an identical version of this model trained on the full, unfiltered training set as a reference point called `all-data`.

The main preprocessing, data augmentation and training scripts are freely available online: `https://gitlab.com/farid-fari/fewshot-learning`.

We use the pre-trained BERT model from Huggingface (Wolf et al., 2020): the `bert-large-uncased-whole-word-masking` variant, which has the benefit for our use case of masking whole words rather than subwords.

5.2 Data and preprocessing

Our training data consists of both genuine Gujarati-English parallel data and backtranslations from the WMT19 news translation task (Barrault et al., 2019). The backtranslations are produced as described by Bawden et al. (2019) and we follow their method of first training on a mixture of backtranslations and parallel data before fine-tuning on the genuine parallel data only. As mentioned previously, we filter out English words selected for evaluation from both the synthetic and genuine parallel data. However, only the genuine parallel data is used to select augmentation candidates to ensure their high quality. For testing, we use the concatenation of `newstest2019` and `newsdev2019`.

Preprocessing All data augmentation was run after tokenization but before subword segmentation to keep a consistent notion of a 'word'. We

---

[3] We experimented with SGD but observed no significant differences with Adam.

[4] `http://data.statmt.org/wmt19_systems/en-gu/train.sh`

first apply tokenization using the Moses tokenizer (Koehn et al., 2007) and then apply sub-word segmentation using the BPE strategy (Sennrich et al., 2016) and the `fastbpe` implementation.[5,6] Word alignment is carried out using `fast_align` (Dyer et al., 2013).[7]

Evaluation data The rare words to filter out are chosen to be the 100 rarest words in the training set that appear at least 5 times in the test set and 20 times in the training set, and are manually filtered down to 96 words to exclude low quality choices (such as plurals and punctuation). Their frequency in the unfiltered training set (used by our reference model) ranges from 20 to 775 occurrences, with a mean of 313 and a median of 275 occurrences. The complete word list can be found in Appendix A. These words appear in 701 of the 3,014 test set sentences, meaning that the BLEU score is computed on a majority of sentences not containing our evaluation words, thus giving us a good overview of how general translation performance is affected. They appear in only 26,910 of the 8.5M training sentences,[8] meaning that the filtered training set is almost the same size as the original full training data. The genuine parallel data contains 40k sentences which were used for context search in the data augmentation steps.

Evaluation setup Our aim is to evaluate how well the model adapts (through fine-tuning of the model) to the gradual addition of reference examples containing the 96 evaluation words, which were absent from the initial training data. For each evaluation word, we randomly choose 20 reference sentences that contain the word, and for each of the four approaches, we successively make 1, 2, 3, 5, 10, 15 and 20 reference occurrences of each word available to the model to learn from, evaluating at each step. Everything is run as a batch over all 96 words; for instance in an experiment using 3 occurrences, fine-tuning is conducted over 288 reference sentences, with each reference word occurring 3 times.[9]

Each approach that uses padding and/or augmentation requires choosing a ratio r between the total number of fine-tuning sentences and the original number of reference sentences provided by the journalist. For a word w, as the number of occurrences offered to the model grows from 1 to 20, the number of fine-tuning sentences grows from r to $20r$. This ratio r is calculated as follows:

$$n_{total} = n_{ref} + n_{synth} + n_{rand} \quad (1)$$

$$r = \frac{n_{total}}{n_{ref}}, \quad (2)$$

where n_{ref}, n_{synth} and n_{rand} refer respectively to the number of reference, synthetic and random sentences. The chosen ratio is written in brackets following the name of each approach.

For the `augmented` experiments we choose $r = 20$, meaning that we augment each reference sentence with 19 synthetic sentences. This is chosen by manually evaluating the quality of augmented sentences on examples, as shown in Appendix B. We run two `randompad` experiments, `randompad(2)` (half of sentences are random), and `randompad(20)` (same ratio as `augmented(20)` for comparison). The `half(20)` experiment has the same ratio as `augmented(20)`, with each reference sentence accompanied by 9 random sentences and 10 synthetic sentences.[10]

5.3 Evaluation metrics

We use two evaluation metrics: (i) the BLEU score (Papineni et al., 2002) computed with the `multi-bleu-detok` script from the Moses toolkit (Koehn et al., 2007) on the entire test set to evaluate overall MT quality, and (ii) the accuracy of the filtered evaluation words to evaluate how well the approaches learn from the post-edits. We use a clipped bag-of-words accuracy defined as follows: if in the target language the reference sentence contains the word w n times, and the translated sentence contains it p times, then the accuracy is $\frac{\min(p,n)}{n}$. This accuracy is computed separately for each of the evaluation words w, and then averaged to obtain an overall accuracy. The advantage of this metric is that it also only requires segmentation to be done in the target language, meaning that once more no prior knowledge or training fur-

---

[5] https://github.com/glample/fastBPE

[6] We reuse the pre-processing scripts available at http://data.statmt.org/wmt19_systems/scripts/.

[7] https://github.com/clab/fast_align

[8] Both genuine and synthetic parallel training sentences.

[9] The number of sentences is very slightly lower due to some sentences containing 2 evaluation words, but this is rare enough that it does not meaningfully impact our results.

[10] We leave it to future work to explore additional padding and augmentation ratios. In the `half` experiments, we randomly select half of the sentences from the synthetic sentences from `augmented`: this means in particular that we do not necessarily keep the best half of the synthetic sentences, leaving room for improvement in future experiments.

ther than the base model are required for the source language.

A third evaluation metric, which is very important for our scenario but overlooked in previous work, is the speed at which the model improves its accuracy. We define this as the number of reference sentences per novel word it needs to see in order to substantially improve the accuracy in translating these words. For a journalist correcting an MT system's mistakes, it is important to correct a given mistake as few times as possible, since having to correct each mistake beyond a certain number of times might make it more worthwhile to simply manually translate the article. The best way of evaluating this metric is by comparing the evolution of the accuracy curve and the BLEU curve as a function of the number of seen occurrences, as presented in the next section in more detail.

We also compare all models to the reference model `all-data`, trained on the unfiltered training set, as in the University of Edinburgh's WMT19 submission (Bawden et al., 2019). This model is considered as a topline rather than a baseline since it does not perform the same task and was trained with much more data – our baseline being the `finetune` approach.

5.4 Hyper-parameter choice

To ensure optimal settings, for each approach we choose hyper-parameters based on the size of the fine-tuning corpus. Several hyper-parameter choices seem relevant, depending on the goal: more training leads to lower BLEU scores (as overfitting occurs) but higher accuracies on the evaluation words, exposing a trade-off between BLEU and translation performance specifically on the evaluation words. This is shown in Appendix D where we explore different values for the `finetune` approach. In our results in the following section, we use hyper-parameter values that best match the accuracy scores for all approaches, thus simplifying the trade-off to a direct comparison in BLEU score.

Choosing a single epoch and learning rate value depends on the end goal: we can focus on accuracy at all costs, even if it means decreasing overall translation performance, or take a more conservative approach by moderately increasing accuracy while maintaining BLEU. Another way of seeing this trade-off is with respect to a time scale over which adaptation occurs: if the process is to be repeated many times, then it may be wise to decrease BLEU as little as possible, whereas a system that is often reset or used less often can afford to sacrifice more BLEU for extra accuracy on novel words. It is important to note that these results are batched, so the BLEU losses or gains illustrated correspond to the learning of 96 words at once.

We present our results with two adaptation speeds for each approach: a slow (*S*) and a fast (*F*) setting, corresponding to two possible compromises in the previously explained trade-off. At each adaptation speed, for each approach, different hyper-parameters are used in order to match accuracy scores as closely as possible to make their comparison easier. The table of chosen epoch and learning rate values can be found in Appendix C.

6 Results

Figures 2 and 3 present the results for the reference model and all approaches for both slow and fast adaptation speeds. We immediately notice that at both adaptation speeds our models are capable of surpassing the `all-data` model[11] in terms of accuracy on the new words, despite having seen fewer than 20 (and as few as 1) reference sentences containing those words, whereas `all-data` has seen each evaluation word over 300 times on average. Our baseline, the `finetune` model, is shown to be largely surpassed by all other approaches in accuracy, both on the slow and fast settings.

With hyper-parameters being chosen to approximately match accuracy curves, the main comparison point is the BLEU score. In Figure 2, we see that in the slow setting our `half` approach is the only one able to improve the BLEU score while learning the new words, nearly matching `all-data` in both BLEU and accuracy at 3 occurrences seen. The `augmented` approach offers a slightly better accuracy curve than all other models on this speed, but loses out on BLEU score at higher occurrence numbers.

It is very important to look at the first points on the curve: 1-5 occurrences of each evaluation word is the realistic range to imagine a journalist making corrections, since a journalist could become frustrated with a model requiring each new word to be corrected up to 20 times for it to be correctly translated. In this respect, Figure 3 for the fast setting offers the best results, with accu-

---

[11] Note however that `all-data` does not constitute a baseline for our model: it does not perform the same task and has access to many more reference sentence pairs.

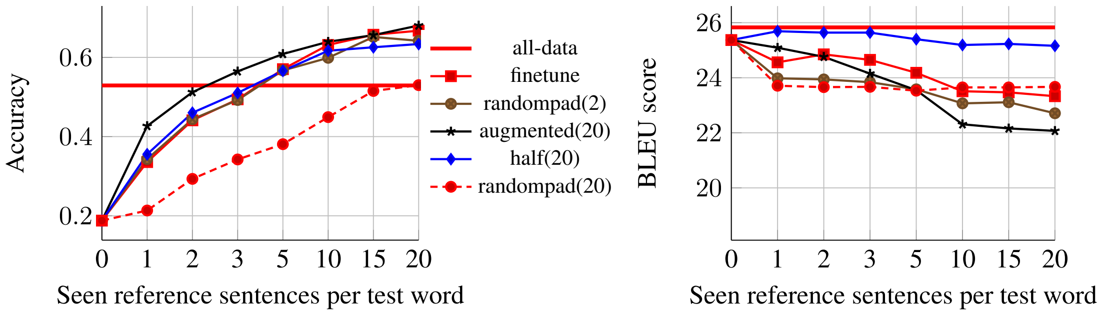


Figure 2: BLEU and accuracy results of all of our approaches for the slow speed setting

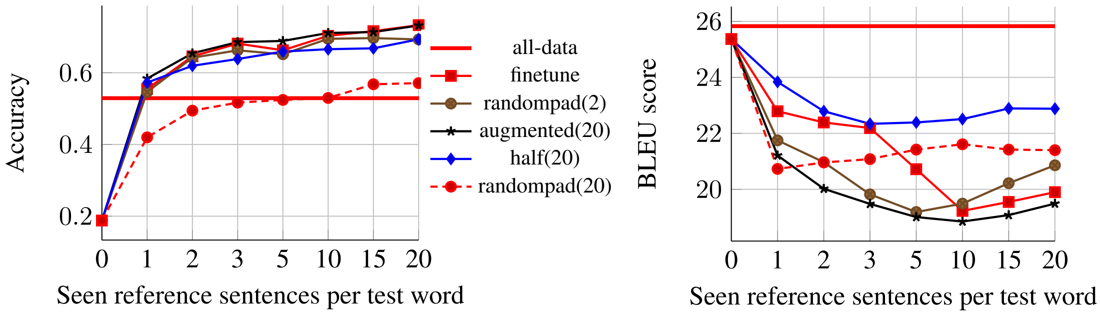


Figure 3: BLEU and accuracy results of all of our approaches for the fast speed setting

racy immediately surpassing `all-data` for all but one model, albeit at a heightened cost to BLEU. Several models, including our new `augmented` and `half` approaches, see an increase in BLEU at higher occurrence numbers, suggesting that generalization is occurring as more sentences are learned with bigger learning steps.

The `randompad` approaches surprised us in two ways, especially given `half`'s success: they generally perform worse than `finetune` despite being designed as improvements over it, and `randompad(20)` has poor accuracy scores compared to all other approaches. To the first point, one possible explanation is that overfitting might occur with both the random sentences and the reference sentences when few data is used for fine-tuning (low seen occurrence numbers). This would explain why in Figure 3 the BLEU curve for `randompad` ends up overtaking the `finetune` curve once enough padding data is available to allow generalization. The second point can be explained by a form of 'dilution' of the reference sentences containing the evaluation words: the model over-adapts to the sentences provided to it but does not particularly improve on the evaluation words. We also tried various hyper-parameters for `randompad(20)` but were unable to find a compromise similar to other models, resulting in this approach standing out from the others. This might also be partially explained through variations inherent to the randomness in the selection of the padding sentences.

6.1 Analysis

To gain more insight into which words were translated well or poorly, we chose to look at our accuracy metric for each evaluation word. For the fast `half` approach, averaged across all occurrence numbers, the best words were generally proper nouns such as *Sulawesi* (1.00), *Isabel* (0.98) and *Kohli* (0.68). Acronyms such as *ATM* (1.00), *RCN* (0.98) or *GB* (0.87) also performed very well, and common nouns or verbs had more varied performance: *niece* (0.00), *moustache* (0.37) and *smartphone* (0.87). Although the very best words are generally proper nouns or acronyms and the very worst generally other parts of speech, no clear pattern or general rule can be ascertained.

One phenomenon initially worrying us the was over-translation of evaluation words: a model outputting evaluation words where they should not appear would be able to 'trick' our bag-of-words accuracy metric by artificially inflating it to a certain point. However, we hypothesized and then verified that this would be counteracted by a decrease in BLEU score due to these words appearing in sentences where they should not. This is presented in Appendix E where we confirm this by demonstrating a negative correlation between over-translation and BLEU score.

7 Conclusion

We explored different techniques based on fine-tuning in order to adapt a base model to post-edits containing novel vocabulary. We proposed a data augmentation technique never applied to this task, allowing us to expand the number of occurrences of the new words available to our model to learn from. In our experiments, all proposed adaptation techniques offer better performance on the novel words than our reference model, which had seen the words hundreds of times each. Our data augmentation approaches yield faster adaptation than our baseline, but with a greatly improved BLEU score, especially when combined with generalization using padding with random training set data.

These various techniques could all be applied to lifelong adaptation of an MT system often confronted with new vocabulary or expressions. Our work can be generalized in several directions which we chose to leave for future work: word translations can automatically be retrieved with alignments, rare multi-word expressions (*n-grams*) can be used rather than rare single-token words, the language model generalization can be used in the source language or both languages and several word substitutions can be made in a single sentence. Several approaches can be deployed to further improve the BLEU score when fine-tuning, such as regularization techniques as explored by Simianer et al. (2019), keeping only the best half of synthetic sentences rather than a random half, and a more careful choice of the ratio of augmented and random data relative to reference sentences.

Acknowledgments

This work was supported by funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825299 (GoURMET) and funding by the UK Engineering and Physical Sciences Research Council (EPSRC) fellowship grant EP/S001271/1 (MTStretch).

References

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Rachel Bawden, Nikolay Bogoychev, Ulrich Germann, Roman Grundkiewicz, Faheem Kirefu, Antonio Valerio Miceli Barone, and Alexandra Birch. 2019. The university of Edinburgh's submissions to the WMT19 news translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 103–115, Florence, Italy. Association for Computational Linguistics.

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

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Matthias Huck, Viktor Hangya, and Alexander Fraser. 2019. Better OOV translation with bilingual terminology mining. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5809–5815, Florence, Italy. Association for Computational Linguistics.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2017. The AMU-UEdin submission to the WMT 2017 shared task on automatic post-editing. In *Proceedings of the Second Conference on Machine Translation*, pages 639–646, Copenhagen, Denmark. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective high-quality neural machine translation in C++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135, Melbourne, Australia. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California, USA.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Sachith Sri Ram Kothur, Rebecca Knowles, and Philipp Koehn. 2018. Document-level adaptation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 64–73, Melbourne, Australia. Association for Computational Linguistics.

Xiaoqing Li, Jiajun Zhang, and Chengqing Zong. 2018. One sentence one model for neural machine translation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Antonio Valerio Miceli Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1494, Copenhagen, Denmark. Association for Computational Linguistics.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Patrick Simianer, Joern Wuebker, and John DeNero. 2019. Measuring immediate adaptation performance for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2038–2046, Minneapolis, Minnesota. Association for Computational Linguistics.

Marco Turchi, Matteo Negri, M. Amin Farajia, and Marcello Federico. 2017. Continuous Learning from Human Post-Edits for Neural Machine Translation. *The Prague Bulletin of Mathematical Linguistics*, 108:233–244.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *Proceedings of the 19th International Conference on Computational Science*, pages 84–95, Faro, Portugal. Springer International Publishing.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the*

2016 Conference on Empirical Methods in Natural Language Processing, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

A Filtered words

The complete list of the 96 filtered words in English is given in Table 4.

B Data augmentation example

This is an actual example of contextual data augmentation from a given sentence. The reference sentence is, "*A powerful 7.5 magnitude earthquake hit the Indonesian island of* **Sulawesi** *on Friday, September 29, triggering a tsunami and leaving nearly 400 people dead.*", with the novel word being *Sulawesi*. The top five sentences matching the context of *Sulawesi* above are as follows:

1. This labour shortage prompted the authorities to import slaves from **Indonesia** and Madagascar.
2. Many of them have settled down in Ahmedabad, Vadodara, Mumbai, Kolkota, Delhi, Nagpur and far away places like **Java**, Rangoon, Singapore, Fiji, Eden, Kenya, Uganda, America etc and established their business in these places.
3. The rice lands of **Java** are among the richest in the world.
4. Rising ocean temperatures and ocean acidification means that the capacity of the ocean carbon sink will gradually get weaker, giving rise to global concerns expressed in the **Monaco** and Manado Declarations.
5. **Lara**'s first school was St. Joseph's Roman Catholic primary.

These sentences generally capture the idea that the word refers to an island, except for the last one which might be an outlier. The fourth sentence is remarkable in the fact that *Monaco* is not an island, but the context of *rising ocean temperatures and ocean acidification* as well as the mention of *Manado* alongside (the capital of North Sulawesi) make the sentence relevant here. This highlights the importance of context besides the actual words' meanings.

C Learning rate and epochs

Table 5 gives the epoch numbers and learning rates we chose for our presented examples. The `augmented` and `half` experiments both have lower learning rates because they fine-tune on more data, and thus go through more gradient steps.

D Preliminary survey of hyper-parameters

We tried different hyper-parameter combinations for the `finetune` approach to explore the dependency of accuracy and BLEU score to these. Figure 6 shows these curves labeled by '*number of epochs / learning rate*'.

We noticed a clear trade-off between BLEU score and accuracy as the parameters evolved, with more training leading to bigger gains in accuracy at a cost to BLEU. Higher parameters saw diminishing returns as accuracy would improve little or not at all while BLEU kept decreasing, as can be seen for the two first curves. Lower parameters also showed diminishing returns in BLEU score as accuracy dramatically drops.

E Over-translation

Table 7 shows the over-translation metric for different approaches. Over-translation is defined as the number of times each word appears in sentences in excess of the reference sentence, divided by the number of times it appears in the reference. This metric is computed similarly to the accuracy metric as explained in Section 5.3, i.e. per word over all sentences and then averaged over all words.

For reference, the `all-data` model gets an over-translation metric of 0.04 and the base model (the one referred to as 0 occurrences in figures) scores 0.01.

While some approaches have seemingly high over-translation values, the most competitive approaches do not see such a high increase in over-translation. For instance, the over-translation metric of 0.25 for `half(20)` (F) means that an over-translated word would appear for every four reference occurrences of our evaluation words, which means that it would over-translate only in about one in seventeen sentences in the test set (given that reference words only appear in 23% of test set sentences). The `half` approach has very good BLEU scores, which is aligned with the fact that it has some of the best over-translation scores amongst the presented approaches.

Moreover, Figure 8 shows for all of these data points the evolution of BLEU score with over-translation: there is a clear linear correlation, implying that BLEU captures over-translation by our approaches very well.

2018	ATM	Ahmedabad	Ambani	Amul	Anand	Ayr
BJP	Bachchan	Becker	Bedford	Chequers	Constantinople	Conway
DM	Dinesh	Dragons	Fidelity	Fleetwood	GB	GST
Gadkari	Giga	HDFC	Hastings	Isabel	Jammu	Kapoor
Kavanaugh	Keyser	Kohli	Lavrov	Lina	Lucknow	MLA
Manish	Mayorga	Meng	Modi	Molinari	Mukesh	Musk
Márquez	Nana	Narendra	Nifty	Oldham	Palu	Patriarch
Patriarchate	Prithvi	Pune	RCN	RTI	Rajkot	Rupani
Rupee	Sachin	Salman	Scalia	Seeley	Sensex	Shetty
Shilpa	Spiegel	Sulawesi	Surat	Sushma	Tendulkar	Tesla
Tiwari	Twitter	Vadodara	Virat	Vyas	Watts	app
apps	cleanliness	crores	cylinders	dough	fortress	ghee
inaugurate	intoxicated	lakhs	litre	mentioning	moustache	niece
refrigerators	sacrificed	slab	smartphone	strawberries		

Figure 4: Filtered words from the corpus

		`finetune`	`randompad(2)`	`randompad(20)`	`augmented`	`half`
Slow (S)	Epochs	10	10	10	10	10
	Learning rate	4×10^{-5}	4×10^{-5}	1×10^{-5}	4×10^{-6}	4×10^{-6}
Fast (F)	Epochs	30	30	30	10	10
	Learning rate	1×10^{-4}	1×10^{-4}	4×10^{-5}	4×10^{-5}	4×10^{-5}

Figure 5: The chosen learning rate and epoch values in our experiments

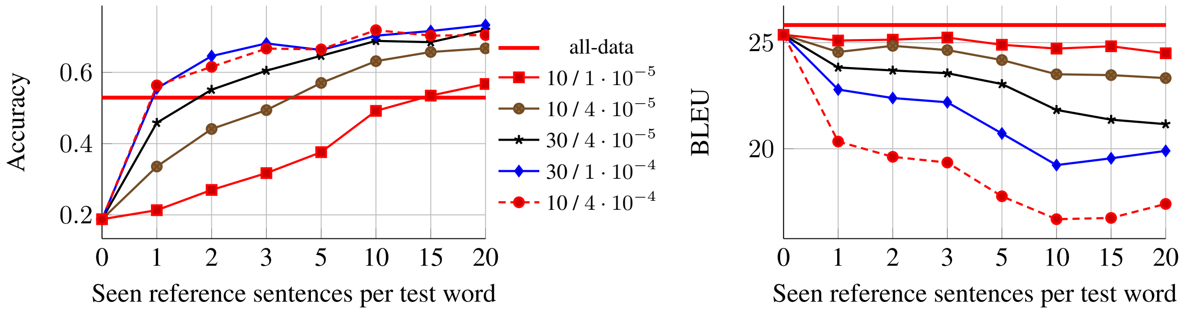


Figure 6: Evolution of the `finetune` approach's performance labeled by *epochs / learning rate*

Seen reference occurrences	1	2	3	5	10	15	20
`finetune` (S)	0.02	0.05	0.08	0.14	0.24	0.24	0.3
`finetune` (F)	0.16	0.33	0.5	0.59	0.85	0.82	0.78
`randompad(2)` (S)	0.03	0.06	0.09	0.13	0.23	0.23	0.25
`randompad(2)` (F)	0.17	0.33	0.48	0.48	0.55	0.6	0.52
`augmented(20)` (S)	0.06	0.1	0.12	0.19	0.26	0.29	0.32
`augmented(20)` (F)	0.32	0.4	0.47	0.57	0.68	0.73	0.76
`half(20)` (S)	0.03	0.07	0.08	0.1	0.15	0.15	0.17
`half(20)` (F)	0.15	0.22	0.24	0.26	0.26	0.28	0.25

Figure 7: The over-translation score of various approaches

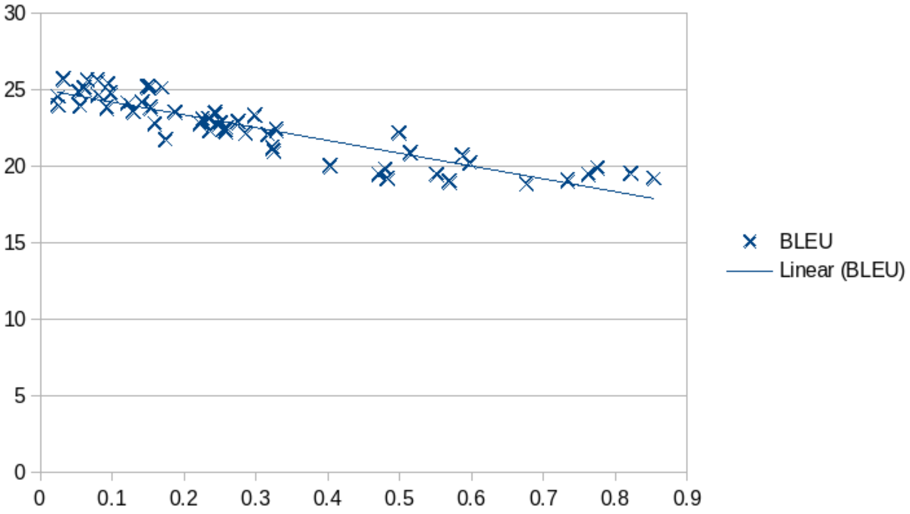


Figure 8: BLEU score as a function of over-translation score

Zero-shot Generalization in Dialog State Tracking through Generative Question Answering

Shuyang Li,[1, 2] **Jin Cao,**[1] **Mukund Sridhar,**[1] **Henghui Zhu,**[3] **Shang-Wen Li,**[3]
Wael Hamza,[1] **Julian McAuley**[2]
[1]Amazon Alexa AI, [2]UC San Diego, [3]AWS AI
`{shl008, jmcauley}@eng.ucsd.edu`
`{jincao, harakere, henghui, shangwel, waelhamz}@amazon.com`

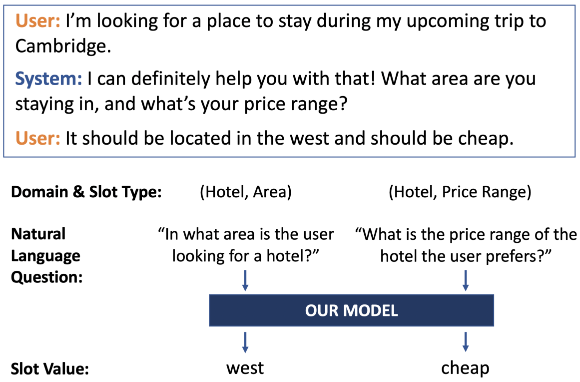
Abstract

Dialog State Tracking (DST), an integral part of modern dialog systems, aims to track user preferences and constraints (slots) in task-oriented dialogs. In real-world settings with constantly changing services, DST systems must generalize to new domains and unseen slot types. Existing methods for DST do not generalize well to new slot names and many require known ontologies of slot types and values for inference. We introduce a novel ontology-free framework that supports natural language queries for unseen constraints and slots in multi-domain task-oriented dialogs. Our approach is based on generative question-answering using a conditional language model pre-trained on substantive English sentences. Our model improves joint goal accuracy in zero-shot domain adaptation settings by up to 9% (absolute) over the previous state-of-the-art on the MultiWOZ 2.1 dataset.

1 Introduction

Dialog agents are gaining increasing prominence in daily life. These systems aim to assist users via natural language conversations, taking the form of digital assistants who help accomplish everyday tasks by interfacing with connected devices and services. A key component to understanding and enabling these task-oriented dialogs is Dialog State Tracking (DST): extracting user intent and goals from conversations via filling in belief slots (Lemon et al., 2006; Wang and Lemon, 2013). Assistive and recommendation use-cases for dialog agents in production settings are particularly challenging due to constantly changing services and applications with which they interface.

Traditional DST systems have achieved high accuracy when presented with a known ontology of slot types and valid values (Chen et al., 2020). In a real-world setting, however, a DST model must generalize to new slot *values* (e.g. new entities that are not present at training time) and new slot *types* (e.g. requirements regarding a new application). Recent work has sought to address these issues by posing DST as a reading comprehension or question answering (QA) task (Gao et al., 2019)—such models predict each slot value independently at any given turn and can theoretically be queried for new slots at inference time.



User: I'm looking for a place to stay during my upcoming trip to Cambridge.

System: I can definitely help you with that! What area are you staying in, and what's your price range?

User: It should be located in the west and should be cheap.

Figure 1: Based on a dialog history, a natural language questions are provided to our model to query a user's requirements and preferences (dialog state).

Some approaches toward DST as QA learn embedding vectors for each slot and/or domain word (Wu et al., 2019), but this is not robust to unseen slots whose specific names (e.g. 'Internet Access') may be totally unlike those in the training set. Gao et al. (2020) attempt to remedy this by posing a natural language question for each slot, but their hybrid span-extraction and classification-based system nonetheless requires access to the full ontology for unknown domains. We present an ontology-free model using natural language questions to represent slots that builds on conditional language modeling techniques—taking advantage of the rise of powerful generative language models (Radford et al., 2019)—to tackle DST as a *generative* QA

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1063–1074
April 19 - 23, 2021.

task. Our model can generalize to unseen domains, slot types, and values, and allows developers to query for arbitrary user requirements via simple questions. To summarize our main contributions:

- We propose an ontology-free conditional language modeling framework for dialog state tracking via generative question answering, achieving state-of-the-art performance in zero-shot domain adaptation settings for DST on MultiWOZ 2.1 (Eric et al., 2020) across all domains with average per-domain gains of 5.9% joint accuracy over previous best methods;
- We demonstrate performance competitive with state-of-the-art methods in a fully supervised setting;
- We show that our approach can be easily adapted to predict slot carry-over and transfer knowledge from a larger, more diverse dataset (Kim et al., 2019), improving zero-shot DST performance across all domains to 11% joint accuracy over the state-of-the-art.

2 Approach

We follow Gao et al. (2019) in treating Dialog State Tracking as a reading comprehension problem: at each turn of dialog, our model reads the dialog history and answers a fixed set of queries about user requirements and preferences (slots), with predictions aggregated to form the belief state. In our framework (Figure 1), we query for a given slot (e.g. Hotel Price Range) by asking a natural language question (Gao et al., 2020)—"What is the price range of the hotel the user prefers?". As our model's predictive ability is based on its general understanding of language and task-oriented conversation, we support zero-shot inference without the need to re-train the model or extend a formal ontology. For example, if a model has not been trained on data from the hotel domain, when presented with a hotel booking conversation we may nonetheless ask it a question like "In what area is the user looking for a hotel?" and received a prediction for that unseen requirement (Hotel Area).

While we conduct our experiments on English-language DST datasets, our approach is applicable to state tracking in any language, provided a conversation history is available.

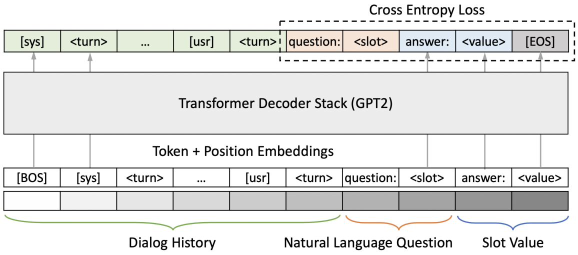


Figure 2: Our model performs DST via generative question-answering. Natural language questions for dialog slots allow our model to generalize to new slot types through its understanding of general language.

Problem Statement We consider a conversation with T turns of user u_t and system utterances y_t: $C = \{y_1, u_1, \dots y_T, u_T\}$. The belief state B_t at turn t comprises many tuples of slots $s \in S$ and their associated values $v_{s,t} \in V_s$, extracted from the conversation history $C_t = \{y_1, u_1, \dots, y_t, u_t\}$. The set of possible values V_s can be arbitrarily large (e.g. possible hotel names), so we represent these values as sequences of vocabulary tokens $v_{s,t} = \{w_1, w_2, \dots, w_k\}, w_i \in \mathcal{W}$. At inference time we pose a natural language question $s = \{w_1, \dots, w_n\}$ and our model predicts an answer (slot value $v_{s,t}$) based on its understanding of the dialog history C_t. To predict the belief state B_t, our model independently answers $|S|$ different questions (Figure 1). In zero-shot DST, the system must predict values for slots outside of the initial ontology—these slot queries correspond to arbitrary natural language questions s' about entities and relationships in the conversation C_t.

Generalizing to New Domains and Slots Dialog State Tracking systems in real-world settings must scale to new users and services, accommodating new slot values (e.g. a new movie release) as well as new domains and slot types (e.g. a service update, or a new connected API). Existing methods require the developer to either write a complete ontology of slots and allowed values or modify their model architecture to add slot-specific prediction heads (Chen et al., 2020). Span-based approaches (Zhang et al., 2019; Zhou and Small, 2019) can correctly predict values that appear verbatim in a conversation but fail when a user paraphrases or mis-phrases a value. They also fall back to treating open-valued slots as classification problems (Zhang et al., 2019; Gao et al., 2020). We approach DST as an ontology-free generative question answering task, as generative methods (Wu et al., 2019; Kumar et al., 2020) have shown promise in few-shot and supervised DST settings.

	JGA (%)	# Params
DistilGPT2 LM	36.35	82 M
DistilGPT2 CLM no PT	39.34	
DistilGPT2 CLM	49.55	
+Question (CLMQ)	50.83	
GPT2 CLMQ	51.02	124 M
GPT2-medium CLMQ	**52.58**	355 M

Table 1: Ablation study of our framework, reporting supervised JGA on the MultiWOZ 2.1 test set.

	MultiWOZ	DSTC8
Train	7,906	16,142
Validation	1,000	2,482
Test	1,000	4,201
Domains	5	19
Slots	30	124
Open	9	59
Numeric	5	12
Temporal	5	10
Categorical	11	43

Table 2: Dataset statistics for MultiWOZ 2.1 and DSTC8: number of dialogs in each split, number of domains, and slots with slot category breakdowns.

While some approaches toward DST as QA learn a set of embeddings for each slot and/or domain (Gao et al., 2019; Wu et al., 2019; Kumar et al., 2020), this is not robust to unseen slots. We encode slots as natural language questions—manually formulating one question per slot—allowing us to share a pre-trained encoder for both dialog context and slot to leverage shared linguistic knowledge (Gao et al., 2020). Thus, our model is also agnostic to ontologies and can answer arbitrary English questions about the dialog history. We treat DST via QA as a conditional language modeling task, and train our model to predict the conditional likelihood of question (slot s) and answer (value $v_{s,t}$) tokens given a dialog context C_t at a given turn t:

$$P(v_{s,t}, s|C_t) = P(v_{s,t}|s, C_t) * P(s|C_t)$$

At inference time, the model is given the dialog context alongside a question—$[C_t; s]$—and asked to predict the value $v_{s,t}$ for that slot.

3 Model Architecture

For our conditional language model, we compared two common architectures: 1) an encoder-decoder model (Sutskever et al., 2014) with a bi-directional encoder; and 2) a purely auto-regressive decoder-only model. We conducted preliminary experiments using both a Transformer (Vaswani et al., 2017) encoder-decoder language model pre-trained using a de-noising auto-encoder objective (Lewis et al., 2020), as well as a Transformer decoder pre-trained with next-token prediction on English web pages. We achieved 1% better supervised DST performance with the decoder-only model in half the training time. Our model architecture thus comprises a Transformer decoder language model that allows us to leverage pre-trained language models like GPT2 (Radford et al., 2019) and common-sense world knowledge accrued through pre-training (Petroni et al., 2019).

We use a BPE (Sennrich et al., 2016) tokenizer to convert input text into a sequence of tokens. These are embedded in $\mathbb{R}^h$ and added to an $\mathbb{R}^h$ sinusoidal positional embedding. This input embedding is processed by l Transformer layers with hidden dimensionality h, each of which applies multi-headed attention with k heads followed by a feed-forward layer with a softmax nonlinearity. The final output hidden states are then projected into our vocabulary space of 50,257 sub-word tokens. We initialize our model weights with DistilGPT2 (Sanh et al., 2019), GPT2 (Radford et al., 2019), or GPT2-medium with $h = 768, 768, 1024$, $l = 6, 12, 24$, and $k = 12, 12, 16$ respectively.

As seen in Figure 2, our input sequence consists of a concatenation of dialog context C_t, slot query s, and slot value $v_{s,t}$: $[C_t; s; v_{s,t}]$. We prepend each utterance with a speaker token `[usr]` or `[sys]` for a user or system speaker to allow our model to identify additional context about each utterance. We pre-pend the slot query and value with `question:` and `answer:` respectively to distinguish slot queries from user-posed questions in the conversation. At training time, we calculate a cross-entropy loss similar to encoder-decoder models by maximizing the log likelihood of the slot query and value conditioned on the dialog context:

$$P(s, v_{s,t}|C_t) = \prod_{i}^{n} P(x_i|x_{<i}, C_t)$$

where $n = |[s; v_{s,t}]|$. We find through ablation experiments on our architecture that this loss computation method out-performs a naïve language-modeling approach that maximizes log likelihood of the full concatenated sequence $[C_t; s; v_{s,t}]$ via the factorized joint distribution (Peng et al., 2020;

Model	Type	JGA	NLQ
TRADE (Wu et al., 2019)	G	45.60	
SUMBT (Lee et al., 2019)*	C	46.70	
STARC (Gao et al., 2020)*	C+S	49.48	Y
MA-DST (Kumar et al., 2020)	G	51.88	
GPT2-m CLMQ	G	**52.58**	Y

Table 3: Supervised DST performance on MultiWOZ 2.1 of our model (underlined) compared to prior methods capable of zero-shot inference. Models using natural language questions (NLQ) are marked. **Requires access to slot-value ontologies at inference time.*

Hosseini-Asl et al., 2020):

$$P(x) = \prod_{i}^{n} P(x_i | x_{<i})$$

This allows for flexibility in learned representations for dialog context while regularizing slot query hidden states.

4 Data

We perform our experiments on **MultiWOZ** (Budzianowski et al., 2018), which contains over 10K single- and multi-domain task-oriented dialogs written by crowd-workers. We use the 2.1 version, with corrected and standardized annotations from Eric et al. (2020). We follow Wu et al. (2019) in lower-casing all dialogs and removing dialogs from training-only domains (Police and Hospital). The final dataset contains 9,906 conversations from 5 domains (Restaurant, Hotel, Attraction, Train, Taxi) covering 30 domain-slot pairs. Each dialog contains an average of 7 user and system turns.

We also experiment with augmenting our training dataset in zero-shot settings with observations drawn from the DSTC8 (Kim et al., 2019) dataset, [1] which contains 16,152 dialogs from 45 domains. DSTC8 was created via template-based dialog models provided with service APIs, and then edited by crowd-workers (Shah et al., 2018). We normalize domains and slots corresponding to the same domain (e.g. `Bus_1, Bus_2`) for a total of 19 domains and 124 slot types in DSTC8. We further manually annotate each dataset with slot value types: open-valued (e.g. Hotel Name), numeric (e.g. Restaurant Guests), temporal (e.g. Taxi LeaveAt), and categorical (e.g. Attraction Type). Dataset statistics are shown in Table 2.

Model	Type	JGA	Extra Supervision
DSTQA	C+S	51.17	Knowledge Graph
DS-DST	C+S	51.21	
GPT2-m CLMQ	G	52.58	
SOM-DST	G	53.68	Previous Dialog State
SST	C	55.23	Schema
TripPy	S	55.30	Previous Dialog Actions
SimpleToD	G	**55.72**	Actions (Training)

Table 4: Supervised DST performance on MultiWOZ 2.1 of our model (underlined) against state-of-the-art DST methods incapable of zero-shot inference.

5 Experiments

We measure DST performance via Joint Goal Accuracy (JGA): the proportion of turns with all belief slots predicted correctly, including those not present. In Section 5.1, we evaluate our model on fully *supervised* DST, in which all domains and slots are known at training time. In Section 5.2, we investigate *zero-shot* domain adaptation in which the model is evaluated on conversations from an unseen domain with previously unseen slots. We then explore how our framework seamlessly accommodates teaching a model to predict slot carryover (Section 5.3) and transfer learning with significantly more diverse domains and slot types (Section 5.4). To measure zero-shot JGA, we follow Campagna et al. (2020) and only consider slots specific to the held-out domain. We focus our analysis on the zero-shot setting, as our goal is to build DST systems that can easily and effectively generalize to new domains and services. We train all models to convergence with a maximum of 10 epochs on Nvidia V100 GPUs, using the Lamb optimizer (You et al., 2020) with a base learning rate of 2e-5. All predictions are made using greedy decoding.

5.1 Supervised DST

We first evaluate on the commonly benchmarked supervised DST task to demonstrate performance competitive with state-of-the-art. In this setting we compare our approach against prior methods capable of zero-shot inference in Table 3—TRADE, STARC, SUMBT, and MA-DST—and those incapable of doing so in Table 4, including DSTQA (Zhou and Small, 2019), DS-DST (Zhang et al., 2019), SOM-DST (Kim et al., 2020), SST (Chen et al., 2020), TripPy (Heck et al., 2020), and SimpleToD (Hosseini-Asl et al., 2020). Our model outperforms all prior models that support zero-shot generalization and is competitive with meth-

---

[1] `https://github.com/google-research-datasets/dstc8-schema-guided-dialogue`

	Rest.	Hot.	Attr.	Train	Taxi
TRADE	12.59	14.20	20.06	22.39	59.21
MA-DST	13.56	16.28	22.46	<u>22.76</u>	59.27
SUMBT	<u>16.50</u>	<u>19.80</u>	<u>22.60</u>	22.50	<u>59.50</u>
Ours (GPT2)	21.05	18.54	23.67	24.34	59.10
Ours (GPT2-m)	**26.17**	**24.41**	**31.31**	**29.07**	**59.61**

Table 5: Zero-shot domain adaptation JGA (%) on MultiWOZ 2.1 test set for recent works and our models with question loss, on the (Rest)aurant, (Hot)el, (Attr)action, Train, and Taxi domains. Previous state-of-the-art results are <u>underlined</u>, with new best **bolded**.

ods that focus solely on supervised DST—most of which require extra supervision at training and inference time, including dialog actions and prior dialog states. We distinguish models by their prediction type as (C)lassification-, (S)pan extraction-, and (G)eneration-based methods.

As seen in Table 1, our formulation of DST as a generative QA task benefits significantly from the usage of a conditional decoder-style model. A standard auto-regressive language modeling formulation (**LM**) with loss computed over the entire input sequence achieves 13% lower JGA compared to computing cross entropy loss only over slot value tokens (**CLM**). Pre-training is also crucial—we see a 10-point drop in JGA when randomly initializing model weights (**no PT**) compared to initializing from pre-trained DistilGPT2 weights. We also compare two other sizes of our models: GPT2-based—comparable in size to SUMBT's (Lee et al., 2019) 112M parameters—and GPT2-medium-based—comparable in size to STARC's (Gao et al., 2020) 355M parameters. We find that scaling the size of our model results in modest improvements in supervised JGA. We hypothesize that extending our loss to cover both slot query and value tokens (**+Question/CLMQ**) helps regularize the hidden representations of question tokens, and we achieve a 1.3% improvement in JGA.

5.2 Zero-Shot DST

Our primary focus lies in the zero-shot domain adaptation setting, where conversations and target slots at inference time come from unseen domains. We use a leave-one-out setup, training our models on four domains from MultiWOZ and evaluating on the held-out domain. Our model must understand a wide variety of possible questions about unseen conversations to generalize well. We compare our model against strong baseline models for zero-shot DST: TRADE, SUMBT, and MA-DST; Table 5 contains results from our models alongside baseline results reported by Kumar et al. (2020) and Campagna et al. (2020). These models represent slots as domain-slot tuples: TRADE learns a separate embedding for each domain and word in slot names, while SUMBT and MA-DST encode domain-slot tuples via BERT (Devlin et al., 2019) and an RNN encoder, respectively.

Our GPT2-medium based model achieves state-of-the-art zero-shot performance on all five domains, and by a significant (5-10%) margin on the Restaurant, Hotel, Attraction, and Train domains. While increased model size modestly impacts supervised DST performance (Table 1), larger models perform significantly better in a zero-shot setting with average absolute gains of 4.8% and relative gains of 22% in JGA across domains. Such improvements are consistent with findings from Brown et al. (2020) that up-sizing language models improves zero-shot performance across various tasks and Petroni et al. (2019), who observe that larger pre-trained models can retain more commonsense and world knowledge from their pre-training corpus—which may help our model understand queries for unseen domains and slots.

Effect of Natural Language Questions Prior work that frames DST as QA typically represents the slot query as a concatenation (tuple) of domain and slot name. Zhang et al. (2019) explore the impact of three different slot representations—domain-slot tuples, short slot descriptions, and full questions—on a hybrid classification-extraction model for DST, and find little difference in performance. However, we find that full questions work much better than domain-slot tuples for our generative framework, especially in zero-shot DST. We hypothesize that natural language questions—structurally similar to dialog utterances and pre-training sentences—allow our model to best leverage its linguistic knowledge with minimal friction when jointly encoding the dialog history, slot query, and slot value.

Wu et al. (2019) find that zero-shot generalization in models that represent slots as tuples is primarily due to shared slot names between domains (e.g. Taxi and Train 'leaveAt'). In a real-world setting a newly added dialog service is unlikely to share slot names verbatim with existing services. To fairly compare tuples and natural language questions under our framework, we per-

USER	My friend told me about Carolina Bed and Breakfast. Do you know anything about it?
SYS	It's a 4 star guesthouse. What would you like to know about it?
USER	Can you give me the postcode? And, do they have internet?
SYS	The postcode is cb13nx; they have internet.
USER	Thanks. Any boat attractions in the west?
SYS	**Nothing in west. Closest boat is the Cambridge Punter in centre. Too far?**
USER	**Yes, it is. How about a museum?**

Error Modality	Slot	Gold	Prediction	Open	Numeric	Temporal	Categorical
Spurious	(Attraction, Name)	`n/a`	cambridge punter	8.4 %	22.3 %	47.7 %	16.0 %
Ignored	(Hotel, Internet)	yes	`n/a`	65.3 %	53.5 %	19.9 %	76.8 %
Wrong Value	(Attraction, Type)	museum	boat	26.3 %	24.2 %	32.4 %	7.2 %

Table 6: Example of different classes of DST errors, and the proportion of errors they make up across the four slot categories for all five domains in a zero-shot setting. Latest (target) turn is **bolded**.

form zero-shot experiments using each representation. For tuple-based questions, our model takes as slot query a synonym of the slot name (e.g. Taxi 'leaveAt' → 'Pick Up Time') instead of a full question (e.g. 'What time does the user want the taxi to pick them up¿). Full question models achieved 6% higher per-domain JGA compared to slot-tuple models, supporting the notion that slot-tuple models memorize slot names rather than understanding their meaning and thus do not generalize well in real-world settings. Using full questions, our model (Table 5) achieves state-of-the-art performance in zero-shot settings.

Error Modalities To analyze our model, we follow Gao et al. (2020) and categorize DST errors in three modalities: 1) the model predicts a *spurious* value for an irrelevant slot; 2) the model *ignores* a relevant slot; and 3) the model correctly infers the presence of a slot but predicts a *wrong value*. Table 6 shows examples of each type of error for a sample conversation, and what proportion of errors they make up in each slot category for our **GPT2-m CLMQ** model in a zero-shot setting. Temporal slots are least likely to be *ignored* by our model, as verbatim `HH:MM` values are easily identifiable in a conversation. However, it is difficult to distinguish between closely related unseen temporal slots like 'leaveAt' and 'arriveBy'. Values for categorical, numeric, and open-valued slots on the other hand can comprise common (non-slot) phrases used in conversation, and thus it is easy for our model to ignore such slot references.

We also examine the source of dialog slots: *users* explicitly express the majority (79.5%) of slot values, while a minority are either derived via user reactions to *system* suggestions (9.7%) or *implicitly* valued (10.8%)—not present verbatim in a conversation. However, our errors are distributed evenly between *user*, *system*, and *implicit* sourced slots—suggesting that it is challenging for our model to track dialog states that are updated reactively via user feedback. We thus see a future opportunity to improve DST models by emphasizing multi-hop reasoning and common-sense inference.

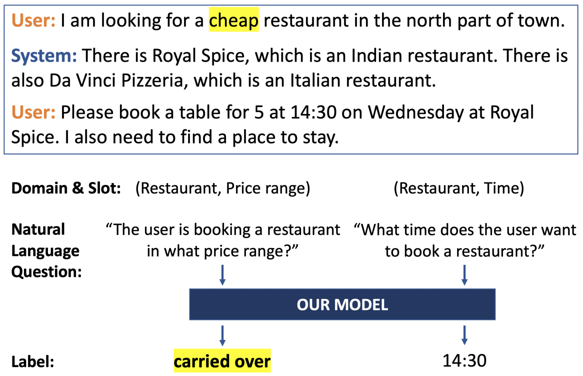
5.3 Predicting Carried Over Slots

Long-range dependencies and slot values carried over from early turns are particularly important to model for accurate DST in long conversations (Kumar et al., 2020). We observe this in the zero-shot setting: our model is able to predict all slots accurately for 61% of conversation first-turns, dropping to 46% after one turn, and 5.7% after seven turns (the average conversation duration). We implement an oracle module to discard predictions when a dialog state does not need updating, obtaining an upper bound for DST improvements due to carry-over prediction. With this oracle, we see an average 5-point improvement in JGA across domain, indicating that carry-over prediction can greatly benefit our model. State-of-the-art models for fully supervised DST often rely on explicitly processing previous dialog states—via slot-value graphs (Zhou and Small, 2019; Chen et al., 2020) or as a separate input to the model at each turn (Heck et al., 2020; Kim et al., 2020). In our framework we can target slot carry-over by training a model to predict a `carried over` token in place of the true slot value whenever a slot value does not need updating at the current turn (**+ Carryover**). At inference time, we replace predicted carry-over tokens with the slot's last predicted value.

Our carry-over implementation improved JGA for all domains (Table 7) by an average of 3.14%, and improved JGA across all context lengths—with the largest improvements (+7%) at the sec-

	Rest.	Hotel	Attr.	Train	Taxi
Previous SOTA	16.50	19.80	22.60	22.76	59.50
GPT2 CLMQ	21.05	18.54	23.67	24.34	59.10
+ Carryover	24.00	19.91	28.45	30.75	59.29
+ DSTC8	24.65	22.94	34.30	38.55	59.68
GPT2-m CLMQ	26.17	24.41	31.31	29.07	59.68
+ CO, DSTC8	**27.69**	**24.88**	**42.39**	**41.05**	**60.32**

Table 7: Zero-shot JGA on MultiWOZ 2.1 test set with carry-over prediction and transfer learning.


User: I am looking for a cheap restaurant in the north part of town.

System: There is Royal Spice, which is an Indian restaurant. There is also Da Vinci Pizzeria, which is an Italian restaurant.

User: Please book a table for 5 at 14:30 on Wednesday at Royal Spice. I also need to find a place to stay.

Figure 3: Our model can be trained to predict the presence of slot carry-over by replacing slot values from previous turns with the `carried over` token.

ond and third turn of a conversation (Figure 4). The `carried over` token allows our model to hedge against low confidence slots, falling back to predictions from previous turns where the target slot may be directly mentioned. This helps reduce the *wrong value* error rate by an average of 31% across each domain. Our model can also propagate null values with carry-over, reducing spurious predictions by an average of 36% across domains. However, we also observe our carry-over model propagating 78% of its errors from previous turns, suggesting that further improvements can result via accurately predicting slot updates.

5.4 Transfer Learning for Generalization

Our framework is ontology-agnostic and thus easily supports transfer learning without modifying the architecture by simply writing natural language questions for additional slots. Gao et al. (2020) found that intermediate fine-tuning of RoBERTa-Large (Liu et al., 2019) on passage-based QA tasks (Fisch et al., 2019) improved zero-shot DST performance. In preliminary experiments, we found no significant impact from intermediate fine-tuning on the SQuAD v2.0 (Rajpurkar et al., 2018) passage-based QA dataset. However, we observe significant improvements when training with joint, non-curriculum learning (McCann et al., 2018; Raffel et al., 2020)—augmenting our training data with an equal number of examples sampled from DSTC8, taking care to remove data from the held-out domain in both MultiWOZ and DSTC8.

Our framework allows for easy joint optimization with carry-over and transfer learning: by training new models on MultiWOZ 2.1 augmented with DSTC8 (**+ DSTC8**) we gain a further average 3.5-point improvement in per-domain JGA (Table 7). On average, our model makes 29% fewer *spurious* errors, and 6.9% fewer errors in open-valued slots, suggesting that our model scales well with additional training data with semantically distinct slot types and values. Our model also makes 9.7% fewer errors on categorical slots and 63% fewer mistakes where it assigns the value of one categorical slot to another, despite being unable to observe the set of possible categorical options—suggesting that exposure to more diverse categorical slots allows our model to better understand and distinguish between such slots. While temporal slots comprise only 17% of MultiWOZ and 10% of DSTC8 slots, these additional examples seem to help our model better disambiguate temporal references and make 32% fewer errors in such slots.

By applying both carry-over and transfer learning to our largest model, we observe further improvements in zero-shot JGA for all domains—averaging 5.1 points better than GPT2-m CLMQ, for an average gain of 11% JGA over previous state-of-the-art across domains (Table 7).

6 Qualitative Analysis

We manually reviewed 300 errors made by our GPT2-medium CLMQ model in the zero-shot setting—annotating 20 errors from each modality (spurious, ignored, wrong value) from each domain with the gold label quality and perceived cause of error totaling 300 annotated examples. As widely observed in recent DST work (Zhou and Small, 2019; Kumar et al., 2020), a significant proportion of DST errors on MultiWOZ are unavoidable—caused by annotation errors. While version 2.1 corrected some of these, annotation errors and inconsistencies remain responsible for 30% of sampled errors—in particular, in 10% of errors the original annotator did not record reactive preferences while in 5% of errors the original annotator did. These inconsistencies can hurt our model's ability to infer

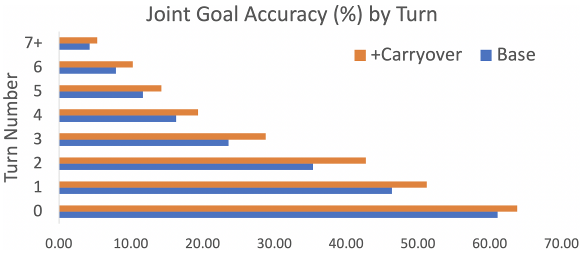


Figure 4: Per-turn JGA on MultiWOZ Test set for GPT2-CLMQ with and without carry-over prediction.

reactive and implied requirements and preferences.

We are also particularly interested in *slot transfers*—when our model mistakenly predicts one slot's value for a different slot, comprising 36% of our manually reviewed errors. In the Taxi and Hotel domains, our model transfers slots from the same domain over 75% of the time, with most swaps occurs between same-category slots (e.g. temporal slots like Taxi 'LeaveAt' and 'ArriveBy'). Slots in these domains are closely semantically related, with values that can fit any slot of that category (e.g. 13:10 vs. 15:15). While a human can easily infer that the earlier of two times must be departure and the later arrival, our model has no inherent understanding of temporal mechanics or numeracy (Wallace et al., 2019). In future work, we will explore learning such knowledge directly via hierarchical softmax output distributions to distinguish between output modalities (Spithourakis and Riedel, 2018), and fine-tuning our model with contrastive losses to learn to rank numerals and times (Hoffer and Ailon, 2015).

For Restaurant, Attraction, and Train, our model tends to swap slot values with those from other domains in the conversation. This is often due to semantically similar slots whose values, at first glance, may not be obviously identifiable as such (e.g. 'Bridge' or 'The Place'). Kumar et al. (2020) similarly observe a particularly high incidence of slot transfers between different-domain 'Name' slots. Other such slots include price ranges and numbers of guests. We have seen that data augmentation with DSTC8 can improve our model's ability to disambiguate such slots—this suggests that we could further improve our model by exposing it to in-domain, conversational reading comprehension data.

While no such dataset currently exists, in future work we aim to explore using question generation (Du et al., 2017) and paraphrasing (Tseng et al., 2014) models to perform in-domain data augmentation, creating reading comprehension questions for task-oriented dialogs that targeting entities and relations not covered by an ontology. We also wish to explore methods for generating general reading comprehension questions for out-of-domain conversations (Shakeri et al., 2020) to improve our model's domain adaptation ability.

7 Related Works

Modern dialog state tracking seeks to capture evolving user intents in a structured belief state (Thomson and Young, 2010). Traditional systems rely on hand-crafted features (Henderson et al., 2014) and classify slot values from a fixed ontology (Mrksic et al., 2017; Ramadan et al., 2018). Gao et al. (2019) and Zhou and Small (2019) fill some slots via spans extracted from dialog history, although they treat non-numeric slots as categorical. Generative methods (Xu and Hu, 2018; Wu et al., 2019) can predict arbitrary unseen values, with Hosseini-Asl et al. (2020) achieving state-of-the-art supervised DST performance in MultiWOZ 2.1 although they cannot predict unseen slots.

By posing DST as generative QA, our framework can leverage language models pre-trained on open-domain documents (Radford et al., 2019) to understand unfamiliar queries. Like Gao et al. (2020), we seek to answer natural language questions about each slot. We contrast our approach to zero-shot DST—which never has access to slots or dialog from the target domain—and that of Campagna et al. (2020), who expose their 'zero-shot' models to synthetic in-domain conversations that require access to the full ontology of the 'held-out' evaluation domain.

We take inspiration from previous work that frames a wide selection of natural language understanding (NLU) tasks (Wang et al., 2019) as QA (McCann et al., 2018) and span extraction (Keskar et al., 2019). While question-answering can be posed as a span extraction task (Wang et al., 2016), generative approaches have proven successful in answering questions about complex passages (Fan et al., 2019). We use a language modeling approach, taking cues from Raffel et al. (2020) who demonstrate that a large language model trained on next-token prediction can learn to solve many different NLU tasks posed as text. Recent work has also shown that large pre-trained language models can generalize to new NLU tasks with few or no examples (Brown et al., 2020), and we leverage this alongside world knowledge acquired during the pre-training process (Petroni et al., 2019) to build a DST model that is robust to new domains and slot-value ontologies.

8 Conclusion

This paper proposes a conditional language modeling approach to multi-domain DST posed as a generative question answering task. By leveraging natural language questions as state queries, our model can generalize to unseen domains, slots, and values via its understanding of language. Our model achieves state-of-the-art zero-shot results on the MultiWOZ 2.1 dataset with average per-domain absolute improvements of 5.9% joint accuracy. We also demonstrate that our framework is easily extensible to support transfer learning and learning slot carry-over. In the future, it is worth exploring mechanisms for our model to better understand relative temporal values and general reading comprehension questions from conversations in order to disambiguate semantically similar dialog slots.

Acknowledgments

We thank Ben Liu, Maryam Fazel-Zarandi, Anuj Goyal, and anonymous reviewers for providing valuable feedback on this work. Work was performed during first author's internship at Amazon Alexa AI. Findings and observations are of the authors only, and do not necessarily reflect the views of Amazon or UCSD.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5016–5026.

Giovanni Campagna, Agata Foryciarz, Mehrad Moradshahi, and Monica S. Lam. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 122–132.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7521–7528.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1342–1352.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: long form question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3558–3567.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering, MRQA@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 1–13.

Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Hakkani-Tür. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. *CoRR*, abs/2004.05827.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tür. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue, SIGdial 2019, Stockholm, Sweden, September 11-13, 2019*, pages 264–273.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 35–44.

Matthew Henderson, Blaise Thomson, and Steve J. Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the SIGDIAL 2014 Conference, The 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 18-20 June 2014, Philadelphia, PA, USA*, pages 292–299.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Unifying question answering and text classification via span extraction. *CoRR*, abs/1904.09286.

Seokhwan Kim, Michel Galley, R. Chulaka Gunasekara, Sungjin Lee, Adam Atkinson, Baolin Peng, Hannes Schulz, Jianfeng Gao, Jinchao Li, Mahmoud Adada, Minlie Huang, Luis A. Lastras, Jonathan K. Kummerfeld, Walter S. Lasecki, Chiori Hori, Anoop Cherian, Tim K. Marks, Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, and Raghav Gupta. 2019. The eighth dialog system technology challenge. *CoRR*, abs/1911.06394.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 567–582.

Adarsh Kumar, Peter Ku, Anuj Kumar Goyal, Angeliki Metallinou, and Dilek Hakkani-Tür. 2020. MA-DST: multi-attention-based scalable dialog state tracking. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8107–8114.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5478–5483.

Oliver Lemon, Kallirroi Georgila, James Henderson, and Matthew N. Stuttle. 2006. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: Generic slot-filling in the TALK in-car system. In *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Nikola Mrksic, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve J. Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1777–1788.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. SOLOIST: few-shot task-oriented dialog with A single pre-trained auto-regressive model. *CoRR*, abs/2005.05298.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473.

Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. 2019. Better language models and their implications. *OpenAI Blog*.

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789.

Osman Ramadan, Pawel Budzianowski, and Milica Gasic. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 432–437.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Pararth Shah, Dilek Hakkani-Tür, Gökhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry P. Heck. 2018. Building a conversational agent overnight with dialogue self-play. *CoRR*, abs/1801.04871.

Siamak Shakeri, Cícero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5445–5460.

Georgios P. Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2104–2115.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Blaise Thomson and Steve J. Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Comput. Speech Lang.*, 24(4):562–588.

Ya-Min Tseng, Yi-Ting Huang, Meng Chang Chen, and Yeali S. Sun. 2014. Generating comprehension questions using paraphrase. In *Technologies and Applications of Artificial Intelligence, 19th International Conference, TAAI 2014, Taipei, Taiwan, November 21-23, 2014. Proceedings*, pages 310–321.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5306–5314.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Zhuoran Wang and Oliver Lemon. 2013. A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In *Proceedings of the SIGDIAL 2013 Conference, The 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 22-24 August 2013, SUPELEC, Metz, France*, pages 423–432.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 808–819.

Puyang Xu and Qi Hu. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1448–1457.

Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S. Yu, Richard Socher, and Caiming Xiong. 2019. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *CoRR*, abs/1910.03544.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *CoRR*, abs/1911.06192.

Zero-shot Neural Passage Retrieval via Domain-targeted Synthetic Question Generation

Ji Ma Ivan Korotkov Yinfei Yang Keith Hall Ryan McDonald
Google Research
`{maji,ivankr,yinfeiy,kbhall,ryanmcd}@google.com`

Abstract

A major obstacle to the wide-spread adoption of neural retrieval models is that they require large supervised training sets to surpass traditional term-based techniques, which are constructed from raw corpora. In this paper, we propose an approach to zero-shot learning for passage retrieval that uses synthetic question generation to close this gap. The question generation system is trained on general domain data, but is applied to documents in the targeted domain. This allows us to create arbitrarily large, yet noisy, question-passage relevance pairs that are domain specific. Furthermore, when this is coupled with a simple hybrid term-neural model, first-stage retrieval performance can be improved further. Empirically, we show that this is an effective strategy for building neural passage retrieval models in the absence of large training corpora. Depending on the domain, this technique can even approach the accuracy of supervised models.

1 Introduction

Recent advances in neural retrieval have led to advancements on several document, passage and knowledge-base benchmarks (Guo et al., 2016; Pang et al., 2016; Hui et al., 2017; Dai et al., 2018; Gillick et al., 2018; Nogueira and Cho, 2019a; MacAvaney et al., 2019; Yang et al., 2019a,b,c). Most neural passage retrieval systems are, in fact, two stages (Zamani et al., 2018; Yilmaz et al., 2019), illustrated in Figure 1. The first is a true retrieval model (aka first-stage retrieval[1]) that takes a question and retrieves a set of candidate passages from a large collection of documents. This stage itself is rarely a neural model and most commonly is an term-based retrieval model such as BM25 (Robertson et al., 2004; Yang et al., 2017), though there is recent work on neural models (Zamani et al., 2018; Dai and Callan, 2019; Chang et al., 2020; Karpukhin et al., 2020; Luan et al., 2020). This is usually due to the computational costs required to dynamically score large-scale collections. Another consideration is that BM25 is often high quality (Lin, 2019). After first-stage retrieval, the second stage uses a neural model to rescore the filtered set of passages. Since the size of the filtered set is small, this is feasible.

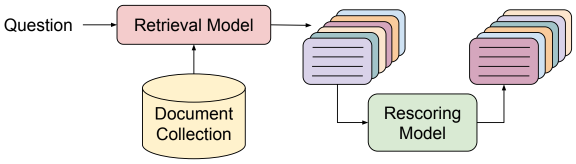


Figure 1: End-to-end neural retrieval. A first-stage model over a large collection returns a smaller set of relevant passages which are reranked by a rescorer.

The focus of the present work is methods for building neural models for *first-stage passage retrieval* for large collections of documents. While rescoring models are key components to any retrieval system, they are out of the scope of this study. Specifically, we study the *zero-shot* setting where there is no target-domain supervised training data (Xian et al., 2018). This is a common situation, examples of which include enterprise or personal search environments (Hawking, 2004; Chirita et al., 2005), but generally any specialized domain.

The zero-shot setting is challenging as the most effective neural models have a large number of parameters, which makes them prone to overfitting. Thus, a key factor in training high quality neural models is the availability of large training sets. To address this, we propose two techniques to improve neural retrieval models in the zero-shot setting.

First, we observe that general-domain question-passage pairs can be acquired from community platforms (Shah and Pomerantz, 2010; Duan et al., 2017) or high quality academic datasets that are publicly available (Kwiatkowski et al., 2019; Bajaj et al., 2016). Such resources have been used to

---

[1] Also called *open domain* retrieval.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1075–1088
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

create open domain QA passage retrieval models. However, as shown in Guo et al. (2020) and in our later experiments, neural retrieval models trained on the general domain data often do not transfer well, especially for specialized domains.

Towards zero-shot neural retrieval with improved domain adaptability, we propose a data augmentation approach (Wong et al., 2016) that leverages these naturally occurring question/answer pairs to train a generative model that synthesizes questions given a text (Zhou et al., 2017). We apply this model to passages in the target domain to generate unlimited pairs of synthetic questions and target-domain passages. This data can then be used for training. This technique is outlined in Figure 2.

A second contribution is a simple hybrid model that interpolates a traditional term-based model – BM25 (Robertson et al., 1995) – with our zero-shot neural model. BM25 is also zero-shot, as its parameters do not require supervised training. Instead of using inverted index which is commonly used in term-based search, we exploit the fact that BM25 and neural models can be cast as vector similarity (see Section 4.4) and thus nearest neighbour search can be used for retrieval (Liu et al., 2011; Johnson et al., 2017). The hybrid model takes the advantage of both the term matching and semantic matching.

We compare a number of baselines including other data augmentation and domain transfer techniques. We show on three specialized domains (scientific literature, travel and tech forums) and one general domain that the question generation approach is effective, especially when considering the hybrid model. Finally, for passage retrieval in the scientific domain, we compare with a number of recent supervised models from the BioASQ challenge, including many with rescoring stages. Interestingly, the quality of the zero-shot hybrid model approaches supervised alternatives.

2 Related Work

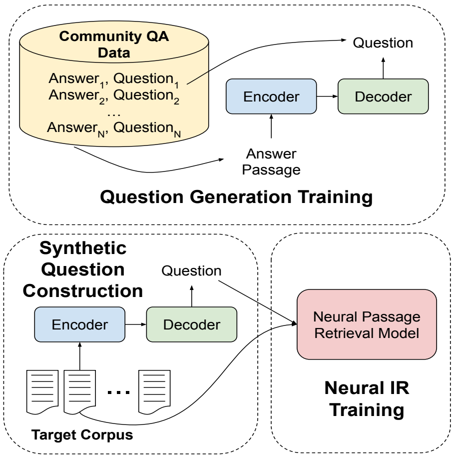


Figure 2: Synthetic query generation for neural IR.

Neural Retrieval The retrieval vs. rescorer distinction (Figure 1) often dictates modelling choices for each task. For first-stage retrieval, as mentioned earlier, term-based models that compile document collections into inverted indexes are most common since they allow for efficient lookup (Robertson et al., 2004; Yang et al., 2017). However, there are studies that investigate neural first-stage retrieval. A common technique is to learn the term weights to be used in an inverted index (Zamani et al., 2018; Dai and Callan, 2019, 2020). Another technique is representation-based models that embed questions and passages into a common dense subspace (Palangi et al., 2016) and use nearest neighbour search for retrieval (Liu et al., 2011; Johnson et al., 2017). Recent work has shown this can be effective for passage scoring (Chang et al., 2020; Karpukhin et al., 2020; MacAvaney et al., 2020). Though all of the aforementioned first-stage neural models assume supervised data for fine-tuning. For rescoring, scoring a small set of passages permits computationally intense models. These are often called interaction-based, one-tower or cross-attention models and numerous techniques have been developed (Guo et al., 2016; Hui et al., 2017; Xiong et al., 2017; Dai et al., 2018; McDonald et al., 2018), many of which employ pre-trained contextualized models (Nogueira and Cho, 2019a; MacAvaney et al., 2019; Yang et al., 2019a,b). Khattab and Zaharia (2020) also showed that by delaying interaction to the last layer, one can build a first stage retrieval model which also leverages the modeling capacity of an interaction based models.

Model Transfer Previous work has attempted to alleviate reliance on large supervised training sets by pre-training deep retrieval models on weakly supervised data such as click-logs (Borisov et al., 2016; Dehghani et al., 2017). Recently, Yilmaz et al. (2019) has shown that training models on general-domain corpora adapts well to new domains without targeted supervision. Another common technique for adaptation to specialized domains is to learn cross-domain representations (Cohen et al., 2018; Tran et al., 2019). Our work is more aligned with methods like Yilmaz et al. (2019) which use general domain resources to build neural models for new domains, though via different techniques – data augmentation vs. model transfer. Our experiments show that data augmentation compares favorably a model transfer baseline. For specialized domains, recently, there have been a number of studies using cross-domain transfer and other techniques for biomedical passage retrieval via the TREC-COVID challenge[2,3] that uses the CORD-19 collection (Wang et al., 2020).

Question generation for data augmentation is a common tool, but has not been tested in the pure zero-shot setting nor for neural passage retrieval. Duan et al. (2017) use community QA as a data source, as we do, to train question generators. The generated question-passage pairs are not used to train a neural model, but QA is instead done via question-question similarity. Furthermore, they do not test on specialized domains. Alberti et al. (2019) show that augmenting supervised training resources with synthetic question-answer pairs can lead to improvements. Nogueira et al. (2019) employed query generation in the context of first-stage retrieval. In that study, the generated queries were used to augment documents to improve BM25 keyword search. Here we focus on using synthetic queries to train the neural retrieval models.

Hybrid Models Combining neural and term-based models have been studied, most commonly via linearly interpolating scores in an approximate re-ranking stage (Karpukhin et al., 2020; Luan et al., 2020) or through the final layer of a rescoring network (Severyn et al., 2015; McDonald et al., 2018). Since rescoring can be cast as classification, blending signals is straight-forward. However, this is approximate as it does not operate over the whole collection. For first-stage retrieval, the most common method is to learn term weights for a standard inverted index in order to make search efficient (Zamani et al., 2018; Dai and Callan, 2019). Here we propose a first-stage retrieval model that incorporates both term-based (sparse) and neural-based (dense) representations in a hybrid model that uses nearest neighbor search for exact inference (Liu et al., 2011; Johnson et al., 2017; Wu et al., 2019). Similar methods using approximate nearest neighbour search have been investigated by Seo et al. (2019).

3 Synthetic Question Generation

In this work, we are specifically investigating the zero-shot scenario where there exists neither user issued questions nor domain specific data except the passage collection itself. We propose to address the

Ubuntu Forums
Passage: Every time I get a notification about and begin updating when they become available, the process is interrupted by an error message: error in foomatic-filters. Then I get "error in linux generic package" and a bunch of numbers. This is replaced before I can write it all down with "error in Linux package" Everything seems to go OK except I don't know if the updates are really being installed. I tried un-installing and re-installing foomatic-filters . . . **Generated Question**: How do I get rid of error in foomatic-filters?

Biomedical Literature
Passage: Electroencephalographic tracings of 50 patients who presented the classical features of Friedreich's ataxia were reviewed . . . Friedreich's ataxia is mainly a spinal disorder. Involvement of supraspinal and in particular brain stem or diencephalic structures may be more extensive in those patients who show electrographic abnormalities. This would require confirmation with comparative data based on pathological observations. Impaired function of brain stem inhibitory mechanism may be responsible for the slightly raised incidence of seizures in patients with Friedreich's ataxia and other cerebellar degenerations. **Generated Question**: What is the significance of Friedreich's ataxia?

Table 1: Examples of domain-targeted synthetic generated questions used to train passage retrieval models.

training data scarcity issue by generating synthetic questions (Zhou et al., 2017; Duan et al., 2017; Alberti et al., 2019; Nogueira et al., 2019). Leverage the fact that there are large question-answer data sources freely available from the web (Shah and Pomerantz, 2010; Duan et al., 2017). we first train a question generator using general domain question-answer pairs. The passage collection of a target domain is then fed into this generator to create pairs of noisy question-passage pairs, which are used to train a retrieval model (see Figure 2). In this work, we mine English question-answer pairs from community resources, primarily StackExchange[4] and Yahoo! Answers[5]. Note we use stackexchange as it covers a wide range of topics, and we focus on investigating the domain adaptability of using a question generation approach. We leave comparing question generator trained on different datasets or using different architectures to future work.

To ensure data quality, we further filter the data by only keeping question-answer pairs that were positively rated by at least one user on these sites. In total, the final dataset contains 2 millions pairs, and the average length of questions and answers are 12 tokens and 155 tokens respectively. This dataset is *general domain* in that it contains question-answer pairs from a wide variety of topics.

Our question generator is an encoder-decoder with Transformer (Vaswani et al., 2017) layers, which is a common for generation tasks such as translation and summarization (Vaswani et al., 2017; Rothe et al., 2019). The encoder is trained to build a representation for a text and the decoder generates a question for which that text is a plausible answer. Appendix B has model specifics.

---

[2] `ir.nist.gov/covidSubmit/`

[3] `ir.nist.gov/covidSubmit/archive.html`

[4] archive.org/details/stackexchange

[5] webscope.sandbox.yahoo.com/catalog.php?datatype=l

Our approach is robust to domain shift as the generator is trained to create questions based on a given text. As a result, generated questions stay close to the source passage material. Real examples are shown in Table 1 for technical and biomedical domains, highlighting the model's adaptability.

4 Neural First-stage Retrieval

In this section we describe our architecture for training a first-stage neural passage retriever. Our retrieval model belongs to the family of *relevance-based dense retrieval* [6] that encodes pairs of items in dense subspaces (Palangi et al., 2016). Let $Q = (q_1, \ldots q_n)$ and $P = (p_1, \ldots, p_m)$ be a question and passage of n and m tokens respectively. Our model consists of two encoders, $\{f_Q(), f_P()\}$ and a similarity function, sim(). An encoder is a function f that takes an item x as input and outputs a real valued vector as the encoding, The similarity function, sim(), takes two encodings, $\mathbf{q}, \mathbf{p} \in \mathbb{R}^N$ and calculates a real valued score, $s = \text{sim}(\mathbf{q}, \mathbf{p})$. For passage retrieval, the two encoders are responsible for computing dense vector representation of questions and passages.

4.1 BERT-based Encoder

In this work, both query and document encoders are based on BERT (Devlin et al., 2019), which has been shown to lead to large performance gains across a number of tasks, including document ranking (Nogueira and Cho, 2019a; MacAvaney et al., 2019; Yang et al., 2019b). In addition, we share parameters between the query and passage encoder – i.e., $f_Q = f_P$, so called Siamese networks – as we found this greatly increased performance while reducing parameters.

We encode P as (CLS, $p_1, \ldots, p_m$, SEP). For some datasets, a passage contains both a title $T = (t_1, ..., t_l)$ and content $C = (c_1, ..., c_o)$, in which case we encode the passage as (CLS, $t_1, ..., t_l$, SEP, $c_1, ..., c_o$, SEP). These sequences are fed to the BERT encoder. Let $h_{\text{CLS}} \in \mathbb{R}^N$ be the final representation of the "CLS" token. Passage encodings $\mathbf{p}$ are computed by applying a linear projection, i.e., $\mathbf{p} = \mathbf{W} * h_{\text{CLS}}$, where $\mathbf{W}$ is a $N \times N$ weight matrix (thus $N = 768$), which preserves the original size of h_{CLS}. This has been shown to perform better than down-projecting to a lower dimensional vector (Luan et al., 2020), especially for long passages.

We encode Q as (CLS, $q_1, q_2, ..., q_n$, SEP) which is then fed to the BERT encoder. Similarly, a linear projection on the corresponding "CLS" token, using the same weight matrix $\mathbf{W}$, is applied to generate $\mathbf{q}$. Following previous work (Luan et al., 2020; Lee et al., 2019b), we use dot product as the similarity function, i.e., $\text{sim}(\mathbf{q}, \mathbf{p}) = \langle \mathbf{q}, \mathbf{p} \rangle = \mathbf{q}^\intercal \mathbf{p}$.

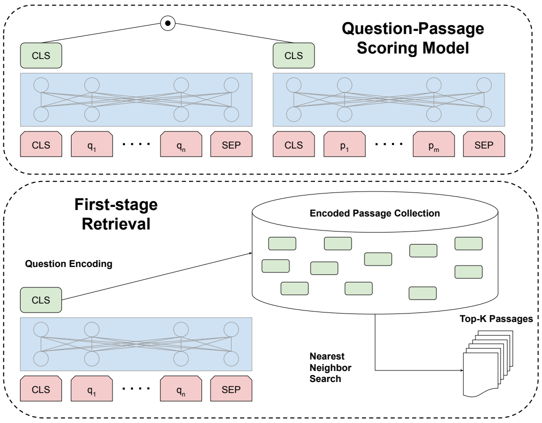


Figure 3: First-stage neural passage retrieval. Top: A BERT-based transformer encodes questions and passages and scores them via dot-product. Bottom: Passages from the collection are encoded and stored in a nearest neighbour search backend. At inference, the question is encoded and relevant passages retrieved.

The top half of Figure 3 illustrates the model.

4.2 Training

For training, we adopt softmax cross-entropy loss. Formally, given an instance $\{\mathbf{q}, \mathbf{p}^+, \mathbf{p}_1^-, ..., \mathbf{p}_k^-\}$ which comprises one query $\mathbf{q}$, one relevant passage $\mathbf{p}^+$ and k non-relevant passages $\mathbf{p}_i^-$. The objective is to minimize the negative log-likelihood:

$$L(\mathbf{q}, \mathbf{p}^+, \mathbf{p}_1^-, ..., \mathbf{p}_k^-) = \log(e^{\langle \mathbf{q}, \mathbf{q}^+ \rangle} + \sum_{i=1}^{k} e^{\langle \mathbf{q}, \mathbf{q}_i^- \rangle}) - \langle \mathbf{q}, \mathbf{q}^+ \rangle$$

This loss function is a special case of ListNet loss (Cao et al., 2007) where all relevance judgements are binary, and only one passage is marked relevant for each training example.

For the set $\{\mathbf{p}_1^-, ..., \mathbf{p}_k^-\}$, we use in-batch negatives. Given a batch of (query, relevant-passage) pairs, negative passages for a query are passages from different pairs in the batch. In-batch negatives has been widely adopted as it enables efficient training via computation sharing (Yih et al., 2011; Gillick et al., 2018; Karpukhin et al., 2020).

4.3 Inference

Since the relevance-based model encodes questions and passages independently, we run the encoder

---

[6] A.k.a. two-tower, dual encoder or dense retrieval.

over every passage in a collection offline to create a distributed lookup-table as a backend. At inference, we run the question encoder online and then perform nearest neighbor search to find relevant passages, as illustrated in the bottom half of Figure 3. While there has been extensive work in fast approximate nearest neighbour retrieval for dense representations (Liu et al., 2011; Johnson et al., 2017), we simply use distributed brute-force search as our passage collections are at most in the millions, resulting in exact retrieval.

4.4 Hybrid First-stage Retrieval

Traditional term-based methods like BM25 (Robertson et al., 1995) are powerful zero-shot models and can outperform supervised neural models in many cases (Lin, 2019). Rescoring systems have shown that integrating BM25 into a neural model improves performance (McDonald et al., 2018). However, for first-stage retrieval most work focuses on approximations via re-ranking (Karpukhin et al., 2020; Luan et al., 2020). Here we present a technique for exact hybrid first-stage retrieval without the need for a re-ranking stage. Our method is motivated by the work of Seo et al. (2019) for sparse-dense QA.

For a query Q and a passage P, BM25 is computed as the following similarity score,

$$\text{BM25}(Q, P) = \sum_{i=1}^{n} \frac{\text{IDF}(q_i) * \text{cnt}(q_i \in P) * (k+1)}{\text{cnt}(q_i \in P) + k * (1 - b + b * \frac{m}{m_{\text{avg}}})},$$

where k/b are BM25 hyperparameters, IDF is the term's inverse document frequency from the corpus, cnt is the term's frequency in a passage, n/m are the number of tokens in Q/P, and m_{avg} is the collection's average passage length.

Like most TF-IDF models, this can be written as a vector space model. Specifically, let $\mathbf{q}^{\text{bm25}} \in [0,1]^{|V|}$ be a sparse binary encoding of a query of dimension $|V|$, where V is the term vocabulary. Specifically this vector is 1 at position i if $v_i \in Q$, here v_i is the i-th entry in V. Furthermore, let $\mathbf{p}^{\text{bm25}} \in \mathbb{R}^{|V|}$ be a sparse real-valued vector where,

$$\mathbf{p}_i^{\text{bm25}} = \frac{\text{IDF}(v_i) * \text{cnt}(v_i \in P) * (k+1)}{\text{cnt}(v_i \in P) + k * (1 - b + b * \frac{m}{m_{\text{avg}}})}$$

We can see that,

$$\text{BM25}(Q, P) = \langle \mathbf{q}^{\text{bm25}}, \mathbf{p}^{\text{bm25}} \rangle$$

As BM25 score can be written as vector dot-product, this gives rise to a simple hybrid model,

$$\begin{aligned} \text{sim}(\mathbf{q}^{\text{hyb}}, \mathbf{p}^{\text{hyb}}) &= \langle \mathbf{q}^{\text{hyb}}, \mathbf{p}^{\text{hyb}} \rangle \\ &= \langle [\lambda \mathbf{q}^{\text{bm25}}, \mathbf{q}^{\text{nn}}], [\mathbf{p}^{\text{bm25}}, \mathbf{p}^{\text{nn}}] \rangle \\ &= \lambda \langle \mathbf{q}^{\text{bm25}}, \mathbf{p}^{\text{bm25}} \rangle + \langle \mathbf{q}^{\text{nn}}, \mathbf{p}^{\text{nn}} \rangle, \end{aligned}$$

where $\mathbf{q}^{\text{hyb}}$ and $\mathbf{p}^{\text{hyb}}$ are the hybrid encodings that concatenate the BM25 ($\mathbf{q}^{\text{bm25}}/\mathbf{p}^{\text{bm25}}$) and the neural encodings ($\mathbf{q}^{\text{nn}}/\mathbf{p}^{\text{nn}}$, from Sec 4.1); and λ is a interpolation hyperparameter that trades-off the relative weight of BM25 versus neural models.

Thus, we can implement BM25 and our hybrid model as nearest neighbor search with hybrid sparse-dense vector dot-product (Wu et al., 2019).

5 Experimental Setup

We outline data and experimental details. The Appendix has further information to aid replicability.

5.1 Evaluation Datasets

BioASQ Biomedical questions from Task B Phase A of BioASQ (Tsatsaronis et al., 2015). We use BioASQ 7 and 8 test data for evaluation. The collection contains all abstracts from MEDLINE articles. Given an article, we split its abstract into chunks with sentence boundaries preserved. A passage is constructed by concatenating the title and one chunk. Chunk size is set so that each passage has no more than 200 wordpiece tokens.

Forum Threads from two online user forum domains: Ubuntu technical help and TripAdvisor topics for New York City (Bhatia and Mitra, 2010). For each thread, we concatenate the title and initial post to generate passages. For BERT-based models we truncate at 350 wordpiece tokens. Unlike the BioASQ data, this data generally does not contain specialist knowledge queries. Thus, compared to the collection of question-answer pairs mined from the web, there is less of a domain shift.

NaturalQuestions Aggregated queries issued to Google Search (Kwiatkowski et al., 2019) with relevance judgements. We convert the original format to a passage retrieval task, where the goal is to retrieval the long answer among all wiki paragraphs (Ahmad et al., 2019). We discarded questions whose long answer is either a table or a list. We evaluate retrieval performance on the development set as the test set is not publicly available. The target collection contains all passages from the development set and is augmented with passages from 2016-12-21 dump of Wikipedia (Chen et al., 2017). Each passage is also concatenated with title. For BERT-based models passages are truncated at 350 wordpiece tokens. This data is different from the previous data in two regards. First, there is a single annotated relevant paragraph per query. This is due to the nature in which the data was curated. Second, this data is entirely "general domain".

Dataset statistics are listed in Appendix A.

5.2 Zero-shot Systems

BM25 Term-matching systems such as BM25 (Robertson et al., 1995) are themselves zero-shot, since they require no training resources except the document collection itself. We train a standard BM25 retrieval model on the document collection for each target domain.

ICT The Inverse Cloze Task (ICT) (Lee et al., 2019b) is an unsupervised pre-training objective which randomly masks out a sentence from a passage and creates synthetic sentence-passage pairs representing membership of the sentence in the passage. These masked examples can then used to train or pre-train a retrieval model. Lee et al. (2019b) showed that masking a sentence with a certain probability, p, can both mimic the performance of lexical matching ($p = 0$) or semantic matching ($p > 0$). ICT is *domain-targeted* since training examples are created directly from the relevant collection. Chang et al. (2020) showed that ICT-based pre-training outperforms a number of alternatives such as Body First Selection (BFS) or Wiki Link Prediction (WLP) for large-scale retrieval.

Ngram Gysel et al. (2018) proposes to train unsupervised neural retrieval system by extracting ngrams and titles from each document as queries. Different from ICT, this approach does not mask the extract ngrams from the original document.

QA The dataset mined from community question-answer forums (Sec. 3) itself can be used directly to train a neural retrieval model since it comes of the form query and relevant text (passage) pair. This data is naturally occurring and not systematically noisy, which is an advantage. However, the data is not domain-targeted, in that it comes from general knowledge questions. We call models trained on this dataset as QA. Applying a model trained on general domain data to a specific domain with no adaptation is a strong baseline (Yilmaz et al., 2019).

QGen The QGen retrieval model trained on the domain-targeted synthetic question-passage pairs described in Section 3. While this model can contain noise from the generator, it is domain-targeted.

	QA	ICT	Ngram	ICT+Ngram	QGen
BioASQ	2.00M	90.50M	636.54M	727.05M	82.62M
NQ	2.00M	71.58M	356.15M	427.72M	84.33M
Forum Travel	2.00M	0.30M	1.25M	1.54M	0.26M
Forum Ubuntu	2.00M	0.42M	2.07M	2.49M	0.43M

Table 2: Number of (synthetic-question, passage) pairs used in zero-shot experiments.

QGenHyb This is identical to QGen, but instead of using the pure neural model, we train the hybrid model in Section 4.4 setting $\lambda = 1.0$ for all models to avoid any domain-targeted tuning. We train the term and neural components independently, combing them only at inference.

All **ICT**, **NGram**, **QA** and **QGen** models are trained using the neural architecture from Section 4. For BioASQ experiments, question and passage encoders are initialized with BioBERT base v-1.1 (Lee et al., 2019a). All other data uses uncased BERT base (Devlin et al., 2019).

We can categorize the neural zero-shot models along two dimensions *extractive* vs. *transfer*. ICT and Ngram are extractive, in that they extract exact substrings from a passage to create synthetic questions for model training. Note that extractive models are also unsupervised, since they do not rely on general domain resources. QA is a *direct* cross-domain transfer model, in that we train the model on data from one domain (or general domain) and directly apply it to the target domain for retrieval. QGen models are *in-direct* cross-domain transfer models, in that we use the out-of-domain data to generate resources for model training.

5.3 Generated Training Datasets

The nature of each zero-shot neural system requires different generated training sets. For ICT, we follow Lee et al. (2019b) and randomly select at most 5 sentences from a document, with a mask rate of 0.9. For Ngram models, Gysel et al. (2018) suggests that retrieval models trained with ngram-order of around 16 was consistently high in quality. Thus, in our experiment we also use 16 and move the ngram window with a stride of 8 to allow 8 token overlap between consecutive ngrams.

For QGen models, each passage is truncated to 512 sentence tokens and feed to the question generation system. We also run the question generator on individual sentences from each passage to promote questions that focus on different aspects of the same document. We select at most 5 salient sentences from a passage, where sentence saliency is the max term IDF value in a sentence.

The size of the generated training set for each baseline is shown in Table 2.

6 Results and Discussion

Our main results are shown in Table 3. We compute Mean Average Precision over the first N[7] results (MAP), Precision@10 and nDCG@10 (Manning et al., 2008) with TREC evaluation script[8]. All numbers are in percentage.

Accuracy of pure neural models are shown in the upper group of Table 3. First, we see that both QA and QGen consistently outperform neural baselines such as ICT and Ngram that are based on sub-string masking or matching. Matching on sub-strings likely biases the model towards memorization instead of learning salient concepts of the passage. Furthermore, query encoders trained on sub-strings are not exposed to many questions, which leads to adaptation issues when applied to true retrieval tasks. Comparing QGen with QA, typically QGen performs better, especially for specialized target domains. This suggests that domain-targeted query generation is more effective for domain shift than direct cross-domain transfer (Yilmaz et al., 2019).

Performance of term-based models and hybrid models are shown in Table 3 (bottom). We can see that BM25 is a very strong baseline. However, this could be an artifact of the datasets as the queries are created by annotators who already have the relevant passage in mind. Queries created this way typically have large lexical overlapping with the passage, thus favoring term matching based approaches like BM25. This phenomenon has been observed by previous work Lee et al. (2019b). Nonetheless, the hybrid model outperforms BM25 on all domains, and the improvements are statistically significant on 9/12 metrics. This illustrate that term-based model and neural-based model return complementary results, and the proposed hybrid approach effectively combines their strengths.

For NaturalQuestions since there is a single relevant passage annotation, we report Precision@1 and Mean reciprocal rank (MRR)[9]. Results are show in Table 4. We can see here that while QGen still significantly outperform other baselines, the gap between QGen and QA is smaller. Unlike BioASQ and Forum datasets, NaturalQuestions contains general domain queries, which aligns well with the question-answer pairs for training the QA model. Another difference is that NaturalQuestions consists of real information seeking queries, in this case QGen performs better than BM25.

6.1 Zero-shot vs. Supervised

One question we can ask is how close to the state-of-the-art in supervised passage retrieval are these zero-shot models. To test this we looked at BioASQ 8 dataset and compare to the top-participant systems.[10] Since BioASQ provides annotated training data, the top teams typically use supervised models with a first-stage retrieval plus rescorer architecture. For instance, the AUEB group, which is the top or near top system for BioASQ 6, 7 and 8, uses a BM25 first-stage retrieval model plus a supervised neural rescorer (Brokos et al., 2018; Pappas et al., 2019).

In order to make our results comparable to participant systems, we return only 10 passages per question (as per shared-task guidelines) and use the official BioASQ 8 evaluation software.

Table 5 shows the results for three zero-shot systems (BM25, QGen and QGenHyb) relative to the top 4 systems on average across all 5 batches of the shared task. We can see the QGenHyb performs quite favorably and on average is indistinguishable from the top systems. This is very promising and suggests that top-performance for zero-shot retrieval models is possible.

A natural question is whether improved first-stage model plus supervised rescoring is additive. The last two lines of the table takes the two-best first-stage retrieval models and adds a simple BERT-based cross-attention rescorer (Nogueira and Cho, 2019b; MacAvaney et al., 2019). We can see that, on average, this does improve quality. Furthermore, having a better first-stage retriever (QGenHyb vs. BM25) makes a difference.

As noted earlier, on BioASQ, BM25 is a very strong baseline. This makes the BM25/QGenHyb zero-shot models highly likely to be competitive. When we look at NaturalQuestions, where BM25 is significantly worse than neural models, we see that the gap between zero-shot and supervised widens substantially. The last row of Table 4 shows a model trained on the NaturalQuestions training data, which is nearly 2-3 times more accurate than the best zero-shot models. Thus, while zero-shot neural models have the potential to be competitive with supervised counterparts, the experiments here show this is data dependant.

---

[7] BioASQ: N=100; and Forum: N=1000.

[8] `https://trec.nist.gov/trec_eval/`

[9] MRR = MAP when there is one relevant item.

[10] `participants-area.bioasq.org`

	BioASQ 7			BioASQ 8			Forum Travel			Forum Ubuntu		
	MAP	**Prec @10**	**nDCG @10**	**MAP**	**Prec @10**	**nDCG @10**	**MAP**	**Prec @10**	**nDCG @10**	**MAP**	**Prec @10**	**nDCG @10**
NEURAL MODELS												
ICT*	9.31*	3.84*	11.44*	9.31*	3.36*	11.78*	3.66*	11.60*	12.04*	8.93*	21.60*	23.21*
Ngram*	9.17*	3.86*	11.53*	8.81*	2.84*	10.74*	10.00	25.60	28.53	9.44*	22.00*	23.90*
QA†	17.80*	7.46*	21.93*	14.61*	4.26*	17.09*	11.00	27.60	28.32	17.78	**34.00**	34.73
QGen‡	**32.45**	**13.48**	**37.23**	**30.32**	**9.36**	**34.53**	**11.79**	**32.00**	**33.34**	**17.97**	32.40	**36.11**
TERM/HYBRID MODELS												
BM25*	45.12*	**20.66**	50.33*	38.61*	11.94*	42.78*	15.41*	37.60	39.21	16.23*	31.20*	35.16*
QGenHyb‡	**46.78**	20.60	**52.16**	**41.73**	**12.84**	**46.18**	**18.19**	**40.80**	**43.92**	**21.97**	**39.60**	**43.91**

Table 3: Zero-shot first-stage retrieval. Unsupervised*; Out-of-domain†; Synthetic‡. Bold=Best in group. Statistically significant differences (permutation test, $p < 0.05$) from the last row of each group are marked by *.

	MRR	**Prec@1**
BM25*	6.63*	1.84*
ICT*	4.62*	1.58*
Ngram*	7.22*	3.05*
QA†	11.14*	4.35*
QGen‡	14.93	**6.21**
QGenHyb‡	**16.73**	6.05
Supervised	*33.68*	*17.33*

Table 4: Zero-shot ad-hoc retrieval for Natural Questions. Unsupervised*; Out-of-domain†; Synthetic‡. Bold=Best; Underline=Best non-hybrid. Baselines with statistically significant differences (permutation test, $p < 0.05$) from QGen are marked by *.

	B1	B2	B3	B4	B5	Avg.
BM25	31.7	27.8	40.4	40.1	41.8	36.3
QGen	28.9	20.3	30.7	29.0	33.1	28.4
QGenHyb	34.8	31.3	43.4	41.9	45.3	39.3
AUEB-1	33.6	31.8	**44.4**	40.1	46.0	39.2
pa	33.5	**33.0**	43.5	36.0	**48.3**	38.9
bioinfo-3	34.0	31.7	43.7	40.2	46.7	39.2
DeepR-test	30.7	29.1	43.5	39.8	47.5	38.1
BM25→resc.	33.9	29.2	42.4	42.5	457	38.7
QGenHyb→resc.	**37.5**	31.2	43.0	**43.6**	46.6	**40.4**

Table 5: MAP for zero-shot models (above dashed lined) vs. supervised models (below dashed line) on BioASQ8 document retrieval. B1-B5 is batch 1-5.

6.2 Learning Curves

Since our approach allows us to generate queries on every passage of the target corpus, one question is that whether retrieval system trained this way simply memorizes the target corpus or it also generalize on unseen passages. Furthermore, from an efficiency standpoint, how many synthetic training examples are required to achieve maximum performance. To answer these questions, we uniformly sample a subset of documents and then generate synthetic queries only on that subset. Results on BIOASQ 7 are shown in Figure 4, where x-axis denotes the percentage of sampled documents. We can see that retrieval accuracy improves as passage coverage increases. The peak is achieved when using a 20% subset, which covers 21% of the reference passages. This is not surprising because the number of frequently discussed entities/topics are typically limited, and a subset of the passages covers most of them. This result also indicates that the learned system does generalize, otherwise optimal performance would be seen with 100% of the data.

6.3 Generation vs. Retrieval Quality

Another interesting question is how important is the quality of the question generator relative to retrieval performance. Below we measured generation quality (via Rouge-based metrics (Lin and Hovy, 2002)) versus retrieval quality for three systems. The base generator contains 12 transformer layers, the lite version only uses the first 3 layer. The large one contains 24 transformer layers and each layer with larger hidden layer size, 4096, and more attention heads, 16. Retrieval quality was measured on BIOASQ 7 and generation quality with a held out set of the community question-answer data set. Results are shown in Table 6. We can see that larger generation models lead to improved generators. However, there is little difference in retrieval metrics, suggesting that large domain targeted data is the more important criteria.

7 Conclusion

We study methods for neural zero-shot passage retrieval and find that domain targeted synthetic question generation coupled with hybrid term-neural first-stage retrieval models consistently outperforms alternatives. Furthermore, for at least one domain, approaches supervised quality. While out of the scope of this study, future work includes further testing the efficacy of these first-stage models in a full end-to-end system (evaluated briefly in Section 6.1), as well as for pre-training supervised models (Chang et al., 2020).

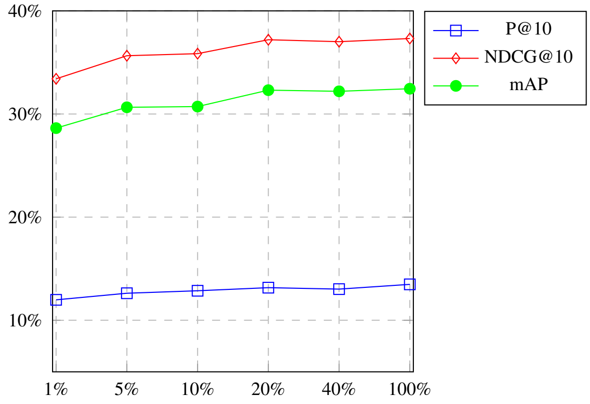


Figure 4: MAP on BioASQ7 (y-axis) w.r.t. the % of documents used for synthesizing queries (x-axis).

	Generation		Retrieval		
	Rouge 1	Rouge L	MAP	Prec @10	nDCG @10
Lite	23.55	21.90	32.50	**13.48**	37.23
Base	26.20	24.23	**32.86**	13.42	**37.96**
Large	**26.81**	**24.90**	32.61	13.34	37.53

Table 6: Generation quality vs. retrieval metrics.

Acknowledgements

We thank members of Google Research Language for feedback on this work. In particular, Gonçalo Simões gave detailed feedback on an early draft of this work, and Shashi Narayan evaluated the question generation quality.

References

Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. 2019. ReQA: An evaluation for end-to-end answer retrieval models. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 137–146, Hong Kong, China. Association for Computational Linguistics.

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic qa corpora generation with roundtrip consistency. *arXiv preprint arXiv:1906.05416*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Sumit Bhatia and Prasenjit Mitra. 2010. Adopting inference networks for online thread retrieval. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, page 1300–1305. AAAI Press.

Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 531–541, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Georgios-Ioannis Brokos, Polyvios Liosis, Ryan McDonald, Dimitris Pappas, and Ion Androutsopoulos. 2018. Aueb at bioasq 6: Document and snippet retrieval. *arXiv preprint arXiv:1809.06366*.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 129–136, New York, NY, USA. Association for Computing Machinery.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Paul Alexandru Chirita, Wolfgang Nejdl, Raluca Paiu, and Christian Kohlschütter. 2005. Using odp metadata to personalize search. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185.

Daniel Cohen, Bhaskar Mitra, Katja Hofmann, and W. Bruce Croft. 2018. Cross domain regularization for neural ranking models using adversarial learning. *CoRR*, abs/1805.03403.

Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687*.

Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1533–1536. ACM.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the ACM Web Search and Data Mining Conference*, pages 126–134.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *CoRR*, abs/1811.08008.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 55–64.

Mandy Guo, Yinfei Yang, Daniel Cer, Qinlan Shen, and Noah Constant. 2020. MultiReQA: A cross-domain evaluation for retrieval question answering models. *arXiv preprint arXiv:2005.02507*.

Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2018. Neural vector spaces for unsupervised information retrieval. *ACM Trans. Inf. Syst.*, 36(4).

David Hawking. 2004. Challenges in enterprise search. In *ADC*, volume 4, pages 15–24. Citeseer.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1049–1058, Copenhagen, Denmark. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *CoRR*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 39–48. ACM.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

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

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019a. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019b. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2002. Manual and automatic evaluation of summaries. In *Proceedings of the ACL-02 Workshop on Automatic Summarization - Volume 4*, AS '02, page 45–51, USA. Association for Computational Linguistics.

Jimmy Lin. 2019. The neural hype and comparisons against weak baselines. In *ACM SIGIR Forum*. ACM New York, NY, USA.

Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. 2011. Hashing with graphs. In *Proceedings of the International Conference on Machine Learning*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, dense, and attentional representations for text retrieval.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via prediction of importance with contextualization. In *Proceedings of the 43rd International ACM SIGIR conference on* *research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1573–1576. ACM.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

Ryan McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1849–1860, Brussels, Belgium. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2019a. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira and Kyunghyun Cho. 2019b. Passage re-ranking with bert. *ArXiv*, abs/1901.04085.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2793–2799. AAAI Press.

Dimitris Pappas, Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. 2019. AUEB at BioASQ 7: document and snippet retrieval. In *Proceedings of the BioASQ Workshop*.

Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at trec-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126. Gaithersburg, MD: NIST.

Stephen Robertson, Hugo Zaragoza, and Michael Taylor. 2004. Simple bm25 extension to multiple weighted fields. In *Proceedings of the ACM Internation Conference on Information and Knowledge Management*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2019. Leveraging pre-trained checkpoints for sequence generation tasks. *CoRR*, abs/1907.12461.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics.

Aliaksei Severyn, Massimo Nicosia, Gianni Barlacchi, and Alessandro Moschitti. 2015. Distributional neural networks for automatic resolution of crossword puzzles. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 199–204, Beijing, China. Association for Computational Linguistics.

Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418.

Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. 2018. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*.

Brandon Tran, Maryam Karimzadehgan, Rama Kumar Pasumarthi, Mike Bendersky, and Don Metzler. 2019. Domain adaptation for enterprise email search. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artiéres, Axel-Cyrille Ngonga Ngomo, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16:138.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008. Curran Associates, Inc.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, et al. 2020. Cord-19: The covid-19 open research dataset. *arXiv preprint arXiv:2004.10706*.

S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell. 2016. Understanding data augmentation for classification: When to warp? In *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–6.

Xiang Wu, Ruiqi Guo, David Simcha, Dave Dopson, and Sanjiv Kumar. 2019. Efficient inner product approximation in hybrid spaces. *arXiv preprint arXiv:1903.08690*.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. 2018. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 55–64.

Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019a. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Wei Yang, Haotian Zhang, and Jimmy Lin. 2019b. Simple applications of bert for ad hoc document retrieval. *arXiv preprint arXiv:1903.10972*.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019c. Multilingual universal sentence encoder for semantic retrieval. *CoRR*, abs/1907.04307.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 247–256, Portland, Oregon, USA. Association for Computational Linguistics.

Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-domain modeling of sentence-level evidence for document retrieval. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3481–3487.

Hamed Zamani, Mostafa Dehghani, W Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 497–506.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

A Data

Statistics on each evaluation set are listed in Table 7. Document collection of "BioASQ" comes from MEDLINE articles, and we remove roughly 10M articles that only contains a title. Furthermore, for BioASQ 7B and BioASQ 8B we only keep articles published before 2018 December 31 and 2019 December 31, respectively. On "Forum", we remove threads with empty posts. On "NQ" since there is at most one passage annotated as relevant for each question, and we also remove questions that have no answer, thus the number of questions equal to the number of reference passages. Besides zero-shot experiments, we also conduct supervised experiments on NQ, where we randomly samples 5% question from the training data as development set. This yields a training and development set with 70,393 and 3,704 (question, passage) pairs, respectively.

The data resources can be downloaded from the following websites

- BioASQ: `http://participants-area.bioasq.org/`
- Forum: `http://sumitbhatia.net/source/datasets.html`
- Natural Questions: `https://github.com/google/retrieval-qa-eval`
- Pubmed / Medline: `https://www.nlm.nih.gov/databases/download/pubmed_medline.html`
- Stackexchange: `http://archive.org/details/stackexchange`
- Yahoo! Answers: `http://webscope.sandbox.yahoo.com/catalog.php?datatype=l`

	BioASQ7	BioASQ8	NQ	ForumTravel	ForumUbuntu
Q	500	500	1772	25	25
R	2349	1646	1772	1,538	1,188
C	50M	53.4M	29.5M	82,669	106,642

Table 7: Statistics on each evaluation set. “Q” denotes the number of unique questions. “R” denotes the total number of annotated reference passages. “C” is the number of passages in the target collection.

- BioBERT: `https://github.com/dmis-lab/biobert`
- BERT: `https://github.com/google-research/bert`

To the extent that we pre-process the data, we will release relevant tools and data upon publication.

B Question Generation Details

Our question generation follows the same implementation of Rothe et al. (2019). Both the encoder and decoder share the same network structure. Parameter weights are also shared and are initialized from a pretrained RoBERTa (Liu et al., 2019) checkpoints. Training data is processed with sentencepiece (Kudo and Richardson, 2018) tokenization. We truncate answers to 512 sentencepiece tokens, and limit decoding to at most 64 steps. The training objective is the standard cross entropy. We use Adam (Kingma and Ba, 2014) with learning rate of 0.05, $\beta_1 = 0.9$, $\beta_2 = 0.997$ and $\epsilon = 1e-9$. Learning rate warmup over the first 40,000 steps. Training batch size for the “lite”, “base” and “large” models are 256, 128 and 32 respectively. All models are trained on a “4x4” slice of v3 Google Cloud TPU. At inference, results from using beam search decoding usually fall short of diversity, thus we use greedy decoding to speed up question generation.

C Neural Model Details

C.1 Zero shot Retrieval Models

C.1.1 Development Set

Since we are investigating zero-shot scenario where there is no annotated development set available, hyperparameters are set by following best practice reported in previous work. We thus do not have development set numbers. However, in the hyperparameters section below, we do use a subset of the zero-shot training data to test training convergence under different parameters.

C.1.2 Data Generation

For ICT task, we follow Lee et al. (2019b) and randomly select at most 5 sentences from a document, with a mask rate of 0.9. For Ngram models, Gysel et al. (2018) suggests that retrieval models trained with N larger than 16 consistently outperform those trained with N smaller than 8. In addition, further increase N from 16 has little effect on retrieval accuracy. Thus, in our experiment we set N to 16 and move the ngram window with a stride of 8 to allow 8 token overlap between consecutive ngrams. For QGen models, each passage is truncated to 512 sentence tokens and feed to the question generation system. Besides, we also run question generator on individual sentences from each document to promote questions that focus on different aspects of the same document. In particular, we select at most top 5 salient sentences from a document, where salience of a sentence is measure as the max IDF value of terms in that sentence. We then feed these sentences to the question generator.

C.1.3 Hyperparameters

For zero-shot neural retrieval model training, we uniformly sample of a subset of 5K (question, document) pairs from the training data as a noisy development set. Instead of finding the best hyperparameter values, we use this subset to find the largest batch size and learning rate that lead the training to converge (Smith et al., 2018). Take batch size for example, we always start from the largest batch that can fit in the memory of a “8x8” TPU slice. We gradually decrease the batch size by a factor of 2 if the current value causes training diverge. More details of hyperparameter values of each task are listed in Table 8. Note on Forum data, the maximum batch size for QGen is much larger than other tasks. Looking into the data, we found that queries generated by ICT or Ngram task on Forum data tends to contain higher percentage of noisy sentences or ngrams that are either ill-relevant to the topic or too general. For example, “suggestions are welcomed”, “any ideas for things to do or place to stay”. We train each model for 10 epochs, but also truncate training steps to 200,000 to make training time tractable.

For BM25, the only two hyperparameters are k and b. We set these to $k = 1.2$ and $b = 0.75$ as advised by Manning et al. (2008).

For the hybrid model QGenHyb, the only hyperparameter is λ. We set this to 1.0 without any tuning, since this represented an equal trade-off between the two models and we wanted to keep the systems zero-shot. However, we did try experimentations. For BioASQ 8b and Forum Ubuntu, values near 1.0 were actually optimal. For BioASQ

		Learning Rate	**Batch Size**
BioASQ	ICT	1e-5	8192
	Ngram	1e-5	8192
	QGen	1e-5	8192
Forum Travel	ICT	2e-6	1024
	Ngram	2e-6	1024
	QGen	2e-6	4096
Forum Ubuntu	ICT	1e-6	512
	Ngram	1e-6	512
	QGen	1e-6	4096
NQ	ICT	1e-5	6144
	Ngram	1e-5	6144
	QGen	1e-5	6144

Table 8: Hyperparameters

7b and Forum Travel, values of 2.0 and 2.1 were optimal and led to improvements in MAP from 0.468 → 0.474 and 0.181 → 0.188, respectively.

C.2 Supervised Models

We also train supervised models on BioASQ and NQ, where we use the development set to do early stopping. For BioASQ, our developement set is data from BioASQ 5 (i.e., disjoint from BioASQ 7 and 8). The development set MAP of our supervised model reranking a BM25 system on this data is 52.1, compared to the BioASQ 8 scores of 38.7. For NQ, the MRR on the development set is 0.141. All other hyperparameters remain the same except we use a smaller batch size of 1024, as we observe that using large batch causes the model quickly overfit the training data. This may due to the number of training examples is 2 orders of magnitude smaller compared to zero-shot setting. For our BioASQ supervised model we follow Pappas et al. (2019) and train it with binary cross-entropy using the top 100 BM25 results as negatives.

C.3 Computational Resources

C.3.1 Question Generation

To train the question generator on 2M questions,

- We used a “4x4” slice of v3 Google Cloud TPU.
- Training time ranges from 20 hours for the lite model and 6 days for the large model.

Once trained, we need to run the generator over our passage collection.

- We distributed computation and used 10,000 machines (CPUs) over the collection.
- For BioASQ, the largest dataset, it took less than 40 hours to generate synthetic questions.

We initialize question generation models from either RoBerta base or Roberta large checkpoint (Liu et al., 2019), and the total number of trainable parameters are 67M for the lite model, 152M for the base model and 455M for the large model.

C.3.2 Neural Retrieval Model

To train the retrieval models, we need to train the query and passage encoders. We share parameters between the two encoders and initialize them from either base BERT (Devlin et al., 2019) or BioBERT (Lee et al., 2019a) checkpoint. Thus retrieval models trained on BioASQ have 108M trainable parameters and retrieval models trained on NQ and Forum data have 110M trainable parameters. After training, we need to run the passage encoder over every passage in the collection to create the nearest neighbour backend.

- Depending on the training batch size, we use either an “8x8” or “4x4” TPU slice.
- Training the ”ngram” model on BioASQ took the longest time, which completes in roughly 30 hours.
- Indexing BioASQ, which is our largest passage collection, with 4000 CPUs which took roughly 4 hours.

Having trained models, the inference task is to encode a query with the neural model and query the distributed nearest neighbour backend to get the top ranked passages. The relevant resources are:

- We encode queries on a single CPU.
- Our distributed nearest neighbour search uses 20 CPUs to serve the collections.
- For BioASQ, our largest collection, to run the inference on the test sets of 500 queries took roughly 1m57s. This is approximately 0.2s per instance to encode the query, run brute-force nearest neighbour search on 10s of millions of examples and return the result.

Discourse-Aware Unsupervised Summarization of Long Scientific Documents

Yue Dong*
MILA/McGill University
Montreal, QC, Canada
`yue.dong2@mail.mcgill.ca`

Andrei Mircea*
MILA/McGill University
Montreal, QC, Canada
`andrei.romascanu@mail.mcgill.ca`

Jackie C. K. Cheung
MILA/McGill University
Montreal, QC, Canada
`jcheung@cs.mcgill.ca`

Abstract

We propose an unsupervised graph-based ranking model for extractive summarization of long scientific documents. Our method assumes a two-level hierarchical graph representation of the source document, and exploits asymmetrical positional cues to determine sentence importance. Results on the PubMed and arXiv datasets show that our approach[1] outperforms strong unsupervised baselines by wide margins in automatic metrics and human evaluation. In addition, it achieves performance comparable to many state-of-the-art supervised approaches which are trained on hundreds of thousands of examples. These results suggest that patterns in the discourse structure are a strong signal for determining importance in scientific articles.

1 Introduction

Single document summarization aims at shortening a text and preserving the most important ideas of the source document. While abstractive strategies generate summaries with novel words, extractive strategies select sentences from the source to form a summary (Nenkova et al., 2011). Despite recent advances in abstractive summarization, extractive models are still attractive in cases where faithfully preserving the original text is the priority. For example, legal arguments can hinge on the exact wording of a contract (Farzindar and Lapalme, 2004), and ensuring the factual correctness of a summary can be critical in the health or scientific domains, which is a known weakness of current abstractive methods (Kryściński et al., 2019).

Introduction	anxiety affects quality of life in those living with parkinson's disease (pd) more so than overall cognitive status, motor deficits, apathy, and depression.
Introduction	although anxiety and depression are often related and coexist in pd patients, recent research suggests that anxiety rather than depression is the most prominent and prevalent mood disorder in pd.
Related Work	furthermore, since previous work, albeit limited, has focused on the influence of symptom laterality on anxiety and cognition, we also explored this relationship .
Methodology	this study is the first to directly compare cognition between pd patients with and without anxiety.
Result	the findings confirmed our hypothesis that anxiety negatively influences attentional set-shifting and working memory in pd.
Result	moreover, anxiety has been suggested to play a key role in freezing of gait (fog), which is also related to attentional set-shifting.
Future work	s. future research should examine the link between anxiety, set-shifting, and fog, in order to determine whether treating anxiety might be a potential therapy for improving fog.

Table 1: Example of a PubMed article's summary produced by our model HipoRank. The hierarchical and directed graph combined with discourse-aware edge weighting allow HipoRank to generate summaries that cover topics from different sections of the scientific article.

Supervised neural-based models have been the dominant paradigm in recent extractive systems, at least for *short news summarization* (Nallapati et al., 2017; Dong et al., 2018; Zhou et al., 2018; Liu and Lapata, 2019; Narayan et al., 2018b; Zhang et al., 2019b). These models usually employ the encoder-decoder structure and have achieved promising performance on news datasets such as CNN/DailyMail (Hermann et al., 2015), and NYT (Sandhaus, 2008).

However, these models cannot easily be adapted to out-of-domain data that have greater length and fewer training examples such as scientific article summarization (Xiao and Carenini, 2019) due to

---

[*] Equal contribution.

[1] Link to our code: https://github.com/mirandrom/HipoRank.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1089–1102
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

two significant limitations. First, they require large domain-specific training pairs of source documents and gold-standard summaries, which are often not available or feasible to create (Zheng and Lapata, 2019). Second, the typical setup of using a token-level encoder-decoder with an attention mechanism does not scale well to longer documents (Shao et al., 2017), as the number of attention computations is quadratic with respect to the number of tokens in the input document.

We instead explore *unsupervised* approaches to address these challenges on long document summarization. We show that a simple unsupervised graph-based ranking model combined with proper sophisticated modelling of discourse information as an inductive bias can achieve unreasonable effectiveness in selecting important sentences from long scientific documents.

For the choice of unsupervised graph-based ranking model, we follow the paradigm of LexRank (Erkan and Radev, 2004) and PACSUM (Zheng and Lapata, 2019). In these methods, sentences are nodes and weighted edges represent the degree of similarity between sentences. Summary generation is formulated as a node selection problem, in which nodes (i.e., sentences) that are semantically similar to other nodes are chosen to be included in the final summary. In other words, they determine node importance by defining a notion of centrality in the graph.

In addition, we augment the document graph with directionality and hierarchy to reflect the rich discourse structure of long scientific documents. In particular, our method relies on two insights about the discourse structure of long scientific documents. The first is that important information typically occurs at the start and end of sections; i.e., they tend to appear near section boundaries (Baxendale, 1958; Lin and Hovy, 1997; Teufel, 1997). We implement this using an asymmetric edge weighting function in a *directed graph* which considers the distance of a sentence to a boundary. The second is that most sentences across section boundaries are unlikely to interact significantly with each other (Xiao and Carenini, 2019). We implement this insight by injecting *hierarchies* into our model, introducing section-level representations as graph nodes in addition to sentence nodes. By doing so, we convert a flat graph into a hierarchical non-fully-connected graph, which has two advantages: 1) reduced computational cost and 2) pruning of distracting weak connections between sentences across different sections.

We call our approach **Hi**erarchical and **Po**sitional **Rank**ing model (HIPORANK) and evaluate it on summarizing long scientific articles from PubMed and arXiv (Cohan et al., 2018). Empirical results show that our method significantly improves performance over previous unsupervised models (Zheng and Lapata, 2019; Erkan and Radev, 2004) in both automatic and human evaluation. In addition, our simple unsupervised approach achieves performance comparable to many expensive state-of-the-art supervised neural models that are trained on hundreds of thousands of examples of long document pairs (Xiao and Carenini, 2019; Subramanian et al., 2019). This suggests that patterns in the discourse structure are highly useful for determining sentence importance in long scientific articles, and that explicitly building in biases inspired by this structure is a viable strategy for building summarization systems.

2 Related Work

2.1 Extractive Summarization

Traditional extractive summarization methods are mostly unsupervised (Radev et al., 2000; Lin and Hovy, 2002; Wan, 2008; Wan and Yang, 2008; Hirao et al., 2013; Parveen et al., 2015; Yin and Pei, 2015; Li et al., 2017; Zheng and Lapata, 2019), utilizing a notion of sentence importance based on n-gram overlap with other sentences and frequency information (Nenkova and Vanderwende, 2005), relying on graph-based methods for sentence ranking (Erkan and Radev, 2004; Mihalcea and Tarau, 2004), or performing keyword extraction combined with submodular maximization (Tixier et al., 2017; Shang et al., 2018).

With the development of large-scale summarization datasets such as CNN/DailyMail (Hermann et al., 2015), NYT (Sandhaus, 2008), Newsroom (Grusky et al., 2018) and XSum (Narayan et al., 2018a), along with advancements in deep neural-based architectures, modern supervised neural network-based methods that employ encoder-decoder framework have become increasingly popular. These models have been proposed with extractive strategies (Cheng and Lapata, 2016; Nallapati et al., 2017; Wu and Hu, 2018; Dong et al., 2018; Zhou et al., 2018; Narayan et al., 2018b); abstractive strategies (See et al., 2017; Chen and Bansal, 2018; Gehrmann et al., 2018; Dong et al., 2019; Zhang et al., 2019a; Lewis et al., 2019); and hybrid strategies (Hsu et al., 2018; Bae et al., 2019; Moroshko et al., 2019).

More recently, extractive approaches leveraging transformer architectures (Vaswani et al., 2017) and their pretrained counterparts (Devlin et al., 2019; Lewis et al., 2019; Zhang et al., 2019a; Dong et al., 2019) have achieved state-of-the-art performances on the CNN/DailyMail news benchmark dataset (Zhang et al., 2019b; Liu and Lapata, 2019; Zhong et al., 2019). Furthermore, pretrained transformer models also provide better sentence representations for unsupervised summarization methods. For instance, PACSUM (Zheng and Lapata, 2019), a directed graph-based unsupervised model that utilizes BERT-based sentence representations, achieved comparable performance to supervised models on the CNN/DailyMail and NYT datasets.

2.2 Extractive Summarization of Long Scientific Papers

Despite the success of deep neural-based models on news summarization, these approaches typically face challenges when applied to long documents such as scientific articles. Furthermore, these approaches are often blind to the topical information resulting from the structured sections in scientific articles (Xiao and Carenini, 2019). Two recent neural supervised models address these issues. Subramanian et al. (2019) used the introduction section as a proxy for the whole document, while Xiao and Carenini (2019) divided articles into sections and used non-auto-regressive approaches to model global and local information.

Besides neural approaches, most previous scientific article summarization systems employ traditional supervised machine learning algorithms with surface features as input (Xiao and Carenini, 2019). Surface features such as sentence position, sentence and document length, keyphrase score, and fine-grain rhetorical categories are often combined with Naive Bayes (Teufel and Moens, 2002), CRFs and SVMs (Liakata et al., 2013), LSTM and MLP (Collins et al., 2017) for extractive summarization over long scientific articles. To the best of our knowledge, the only unsupervised extractive summarization model for long scientific documents relies on citation networks (Qazvinian and Radev, 2008; Cohan and Goharian, 2015), by extracting citation-contexts from citing articles and ranking these sentences to form the final summary. Our proposed method is different from their settings, where we perform single document summarization based on the long source article.

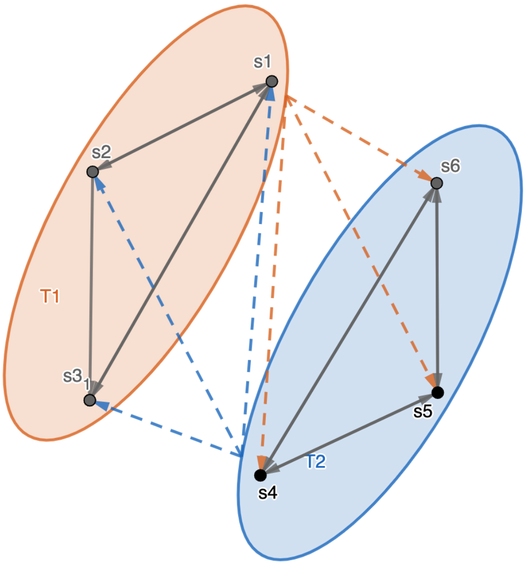


Figure 1: Example of a hierarchical document graph constructed by our approach on a toy document that contains two sections $\{T_1, T_2\}$, each containing three sentences for a total of six sentences $\{s_1, \ldots, s_6\}$. Each double-headed arrow represents two edges with opposite directions. The solid and dashed arrows indicate intra-section and inter-section connections respectively. When compared to the flat fully-connected graph of traditional methods, our use of hierarchy effectively reduces the number of edges from 60 to 24 in this example.

3 Method

Our proposed method combines simple graph-based ranking algorithms with a two-level hierarchical model of the rich discourse structures of long scientific documents (Teufel, 1997; Xiao and Carenini, 2019). We incorporate this discourse information into the graph as inductive biases through the construction of a *directed hierarchical graph* for document representation (Figure 1 and Section 3.2) and through the asymmetric edge weighting of edges with boundary functions (Section 3.3).

3.1 Graph-based Ranking Algorithm

Graph-based ranking algorithms for summarization represent a document as a graph $G = (V, E)$, where V is the set of vertices that represent sentences or other textual units in the document, and

E is the set of edges that represent interactions between sentences. The directed edge e_{ij} from node v_i to node v_j is typically weighted by $w_{ij} = f(sim(v_i, v_j))$, where sim is a measure of similarity between two nodes (e.g. cosine distance between their distributed representations), and f can be an additional weighting function. These algorithms select the most salient sentences from V based on the assumption that sentences that are similar to a greater number of other sentences capture more important content and therefore are more informative.

3.2 Hierarchical Document Graph Creation

To create a hierarchical document graph, we first split a document into its sections, then into sentences[2]. To create the hierarchy, we allow two levels of connections in our hierarchical graph: intra-sectional connections and inter-sectional connections as shown in Figure 1.

Intra-sectional connections aim to model the *local* importance of a sentence within its section. It implements the idea that a sentence that is similar to a greater number of other sentences in the same topic/section should be more important. This is realized in our fully-connected subgraph for an arbitrary section I, where we allow *sentence-sentence* edges for all sentence nodes within the same section.

Inter-sectional connections aim to model the *global* importance of a sentence with respect to other topics/sections in the document, as a sentence that is similar to a greater number of other topics is deemed more important. However, calculating sentence-sentence connections across different sections is computationally expensive and may also suffer from performance degradation due to weak edges between sentences that are unrelated as a result of being from different sections (Mihalcea and Tarau, 2004). To address these issues, We introduce section nodes on top of sentence nodes to form a hierarchical graph. For inter-section connections, we only allow *section-sentence* edges for modeling the global information. This choice makes our approach more computationally efficient while greatly limiting the number of irrelevant inter-section edges that arise from the fact that sections in scientific documents typically have independent topics (Xiao and Carenini, 2019). In contrast, traditional graph-based ranking algorithms have a flat fully-connected graph document with no sections.

3.3 Asymmetric Edge Weighting by Boundary functions

To calculate the weight of an edge, we first measure similarity between a sentence-sentence pair $sim(v_j^I, v_i^I)$ and a section-sentence pair $sim(v^J, v_i^I)$. While our method is agnostic to the measure of similarity, we use cosine similarity with different vector representations in our experiments, averaging a section's sentence representations to obtain its own.

While the similarities of two graph nodes are symmetric, one may be more salient than the other when considering their discourse structures (Baxendale, 1958; Teufel, 1997). Based on these discourse hypotheses of long scientific documents, we capture this asymmetry by making our hierarchical graph *directed* and inject *asymmetric* edge weighting over intra-section and inter-section connections.

Asymmetric edge weighting over sentences Our asymmetric edge weighting is based on the hypothesis that important sentences are near the boundaries (start or end) of a text (Baxendale, 1958). We reflect this hypothesis by defining a *sentence boundary function* d_b over sentences v_i^I in section I such that sentences closer to the section's boundaries are more important:

$$d_b(v_i^I) = \min(x_i^I, \alpha(n^I - x_i^I)), \tag{1}$$

where n^I is the number of sentences in section I and x_i^I represents sentence i's position in the section I. $\alpha \in \mathbb{R}^+$ is a hyper-parameter that controls the relative importance of the start or end of a section or document.

The sentence boundary function allow us to incorporate directionality in our edges, and weight edges differently depending on if they are incident to a more important or less important sentence in the same section. Concretely, we define the weight w_{ji}^I for intra-section edges (incoming edges for i) as:

$$w_{ji}^I = \begin{cases} \lambda_1 * sim(v_j^I, v_i^I), & \text{if } d_b(v_i^I) \geq d_b(v_j^I), \\ \lambda_2 * sim(v_j^I, v_i^I), & \text{if } d_b(v_i^I) < d_b(v_j^I) \end{cases} \tag{2}$$

where $\lambda_1 < \lambda_2$ such that an edge e_{ji} incident to i is weighted more if i is closer to the text bound-

---

[2] Our approach is agnostic to the sentence/section splitting method. In our experiments, articles in the datasets are already split into sections and sentences.

ary than j. Edges with a weight below a certain threshold β can be pruned (i.e., set to 0).

Asymmetric edge weighting over sections Similarly, to reflect the hierarchy hypothesis over long scientific documents proposed by Teufel (1997), we also define a *section boundary function* d_b to reflect that sections near a document's boundaries are more important:

$$d_b(v^I) = \min(x^I, \alpha(N - x^I)), \tag{3}$$

where N is the number of sections in the document and x^I represents section I's position in the document.

This section boundary function allows us to inject asymmetric edge weighting w_i^{JI} to inter-section edges:

$$w_i^{JI} = \begin{cases} \lambda_1 * sim(v^J, v_i^I), & \text{if } d_b(v^I) \geq d_b(v^J). \\ \lambda_2 * sim(v^J, v_i^I), & \text{if } d_b(v^I) < d_b(v^J) \end{cases} \tag{4}$$

where $\lambda_1 < \lambda_2$ such that an edge e_i^{JI} incident to $i \in I$ is weighted more if section I is closer to the text boundary than section J.

3.4 Importance Calculation

We compute the overall importance of sentence v_i^I as the weighted sum of its inter-section and intra-section centrality scores:

$$c(v_i^I) = \mu_1 \cdot c_{\text{inter}}(v_i^I) + c_{\text{intra}}(v_i^I) \tag{5}$$

$$\begin{aligned} c_{\text{intra}}(v_i^I) &= \sum_{v_j^I \in I} \frac{w_{ji}^I}{|I|} \\ c_{\text{inter}}(v_i^I) &= \sum_{v^J \in D} \frac{w_i^{JI}}{|D|}, \end{aligned} \tag{6}$$

where I is the set of sentences neighbouring v_i^I and D is the set of neighbouring sections in the hierarchical document graph; μ_1 is a weighting factor for inter-section centrality.

3.5 Summary Generation

Lastly, we generate a summary by greedily extracting sentences with the highest importance scores until a predefined word-limit L is passed. Most graph-based ranking algorithms recompute importance after each sentence is extracted in order to prevent content overlap. However, we find that the asymmetric edge scoring functions in (2) and (4) naturally prevent redundancy, because similar sentences have different boundary positional scores. Our method thus successfully extracts diverse sentences without recomputing importance.

Dataset	# docs	avg. doc. len.	avg. summ. len.
CNN	92K	656	43
Daily Mail	219K	693	52
NYT	655K	530	38
PubMed	133K	3,016	203
arXiv	215K	4,938	220

Table 2: Dataset statistics on news articles (CNN, DailyMail, and NYT) and long scientific documents (PubMed and arXiv).

4 Experimental Setup

This section describes the datasets, the hyperparameter choices, the baseline models, and the evaluation metrics used in the experiments.

4.1 Datasets

Our experiments are conducted on PubMed and arXiv (Cohan et al., 2018), two large-scale datasets of long and structured scientific articles with abstracts as summaries. The average source article length is four to seven times longer than popular news benchmarks (Table 2), making them ideal candidates to test our method.

4.2 Implementation Details

Our model's hyperparameters for testing are chosen from the ablation studies on the validation sets. The test results are reported with the following hyperparameter settings: $\lambda_1 = 0.0, \lambda_2 = 1.0, \alpha = 1.0$, with $\mu_1 = 0.5$ for PubMed and $\mu_1 = 1.0$ for arXiv. We fix λ_2 to 1 and the choices of $\lambda_1 \in \{-0.2, 0, 0.5\}$. represent whether the edge between a less boundary-important sentence and a more boundary-important sentence is 1) negatively weighted, 2) pruned, or 3) down-weighted. $\lambda_1 < \lambda_2$ such that an edge e_{ji} incident to i is weighted more if i is closer to the text boundary than j. $\alpha \in \{0, 0.5, 0.8, 1.0, 1.2\}$ controls the relative importance of the start or end of a section or document. $\mu_1 \in \{0.5, 1.0, 1.5\}$ controls how much we weigh intra-section sentence importance vs. inter-section sectional importance.

For each dataset, we experimented with different pretrained distributional sentence representation models. The dimension of sentence representations is model-dependent (details in Section

6.2). We used the publicly released BERT model[3] (Devlin et al., 2019), PACSUM BERT model[4] (Zheng and Lapata, 2019), SentBERT and SentRoBERTa[5] (Reimers and Gurevych, 2019), and BioMed word2vec representations[6] (Moen and Ananiadou, 2013). A section's representation is calculated as the average of its sentences' representations. The similarity between sentences or sections is defined to be the cosine similarity between the distributed representations.

4.3 Baselines

We compare our approach with previous unsupervised and supervised models in extractive summarization. In addition, we also compare it with recent neural abstractive approaches for completeness.

For unsupervised extractive summarization models, we compare with SumBasic (Vanderwende et al., 2007), LSA (Steinberger and Jezek, 2004), LexRank (Erkan and Radev, 2004) and PACSUM (Zheng and Lapata, 2019). For supervised neural extractive summarization models, we compare with a vanilla encoder-decoder model (Cheng and Lapata, 2016), SummaRuNNer (Nallapati et al., 2017), GlobalLocalCont (Xiao and Carenini, 2019), Sent-CLF and Sent-PTR (Subramanian et al., 2019). We also compare with neural abstractive summarization models as reported in Xiao and Carenini (2019): Attn-Seq2Seq (Nallapati et al., 2016), Pntr-Gen-Seq2Seq (See et al., 2017) and Discourse-aware (Cohan et al., 2018). In addition, we report the lead baseline that selects the first k tokens as a summary ($k = 203, = 220$ for PubMed and arXiv respectively). Lastly, we report baselines for an Oracle summarizer (Nallapati et al., 2017).

4.4 Evaluation Methods

We evaluate our method with automatic evaluation metrics - ROUGE F1 scores (Lin, 2004). ROUGE-1 and ROUGE-2 compute unigram and bigram overlaps between system summaries and reference summaries, while ROUGE-L computes the longest common sub-sequence of the two.

In addition, we design a human evaluation experiment (details in Section 5.2) to compare our model with the best unsupervised summarization model - PACSUM (Zheng and Lapata, 2019). As far as we know, we are the first to perform human evaluation on the 2018 PubMed and arXiv datasets (Cohan et al., 2018). Human evaluation over long scientific articles require annotators to comprehend a full domain-specific long article and compare multiple summaries for quality evaluation. Due to the challenging nature of the task, previous papers choose to skip it and purely rely on automatic evaluations to judge the system performance.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead	35.63	12.28	25.17
Oracle (ROUGE-2, F1)	55.05	27.48	38.66
Supervised Abstractive			
Attn-Seq2Seq (2016)	31.55	8.52	27.38
Pntr-Gen-Seq2Seq (2017)	35.86	10.22	29.69
Discourse-aware (2018)	38.93	15.37	35.21
Supervised Extractive			
Cheng & Lapata (2016)	43.89	18.53	30.17
SummaRuNNer (2017)	43.89	18.78	30.36
GlobalLocalCont (2019)	44.85	19.70	31.43
Sent-CLF (2019)	45.01	19.91	41.16
Sent-PTR (2019)	43.30	17.92	39.47
Unsupervised Extractive			
SumBasic (2007)	37.15	11.36	33.43
LSA (2004)	33.89	9.93	29.70
LexRank (2004)	39.19	13.89	34.59
PACSUM (2019)	39.79	14.00	36.09
HIPORANK (ours)	**43.58**	**17.00**	**39.31**

Table 3: Test set results on PubMed (ROUGE F1).

5 Results

5.1 Automatic Evaluation Results

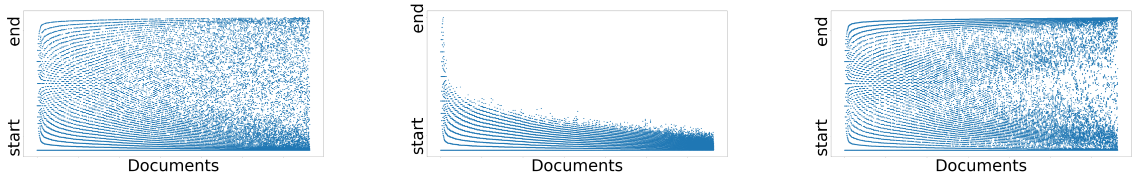
Tables 3 and 4 summarize our automatic evaluation results on the PubMed and arXiv test sets with the best hyperparameters, as described in Section 4.2.

The first blocks in Table 3,4 include the lead and the oracle baselines. The second and the third blocks in the tables present the results of supervised abstractive models, and of supervised extractive models. ROUGE-2 oracle summaries are used as gold standard summaries for training supervised extractive models, which likely contributes to their better ROUGE-2 scores.

The last blocks compare previous unsupervised models with our approach. Our model outperforms all other unsupervised approaches by wide margins in terms of ROUGE-1,2,L F1 scores on both PubMed and arXiv datasets. We also show that PACSUM is biased towards selecting sentences that

---

[3] https://github.com/huggingface/transformers
[4] https://github.com/mswellhao/PACSUM
[5] https://github.com/UKPLab/sentence-transformers
[6] http://bio.nlplab.org/word-vectors



(a) Oracle (b) PacSum (c) HipoRank

Figure 2: Sentence positions in source document for extractive summaries generated by different models on the PubMed validation set. Documents on the x-axis are ordered by increasing article length from shortest to longest. We also see a similar trend on arXiv (the plots with more details can be found in the appendix).

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead	33.66	8.94	22.19
Oracle (ROUGE-2, F1)	53.88	23.05	34.90
Supervised Abstractive			
Attn-Seq2Seq (2016)	29.30	6.00	25.56
Pntr-Gen-Seq2Seq (2017)	32.06	9.04	25.16
Discourse-aware (2018)	35.80	11.05	31.80
Supervised Extractive			
Cheng&Lapata (2016)	42.24	15.97	27.88
SummaRuNNer (2017)	42.81	16.52	28.23
GlobalLocalCont (2019)	43.62	17.36	29.14
Sent-CLF (2019)	34.01	8.71	30.41
Sent-PTR (2019)	42.32	15.63	38.06
Unsupervised Extractive			
SumBasic (2007)	29.47	6.95	26.30
LSA (2004)	29.91	7.42	25.67
LexRank (2004)	33.85	10.73	28.99
PacSum (2019)	38.57	10.93	34.33
HipoRank (ours)	**39.34**	**12.56**	**34.89**

Table 4: Test set results on arXiv (ROUGE F1).

appear at the beginning of a document while our method selects sentences in every section and near the article boundaries, similar to the oracle (Figure 2). This overlap with gold standard summaries suggests our use of discourse structure and hierarchy plays a significant role in our method's performance.

Interestingly, despite limited access to only the validation set for hyperparameter tuning, our method achieves performance scores that are competitive with supervised models that require hundreds of thousands of training examples, outperforming almost all abstractive and extractive models on ROUGE-L. This suggests that our discourse-aware unsupervised model is surprisingly effective at selecting salient sentences in long scientific document and perhaps should be used as a strong baseline to accessing the merits of supervised approaches for learning content beyond discourse.

Model	Content-coverage	Importance
PacSum	30.52	48.70
HipoRank (ours)	**42.13**	**59.06**

Table 5: Human evaluation results on 20 sampled reference summaries with 281 system summary sentences from PubMed. Each reference summary-sentence pair is annotated by two annotators with an average annotator agreement of 73.24%. The results are averaged across 127 sentences from HipoRank and154 sentences from state-of-the-art unsupervised extractive summarization system PacSum (Zheng and Lapata, 2019)..

5.2 Human Evaluation

We asked the human judges[7] to read the reference summary[8] (abstract) and present extracted sentences from different summarization systems in a random and anonymized order. The judges are asked to evaluate the system summary sentence according to two criteria: 1) *content coverage* (whether the presented sentence contains content from the abstract); and 2) *importance* (whether the presented sentence is important for a goal-oriented reader even if it isn't in the abstract (Lin and Hovy, 1997)).

Table 5 presents the human evaluation results. HipoRank is shown to be significantly better than PacSum in both content coverage and importance ($p = 0.002$ and $p = 0.007$ with Mann-Whitney U tests, respectively). We also measure inter-rater reliability using Fleiss' κ (46.56 for *content-coverage* and 41.37 for *importance*). These results help sup-

---

[7] All judges are native English speakers with at least a bachelor's degree and experience in scientific research. We compensated the judges at an hourly rate of $20.

[8] We made the decision to not present the whole article, which would create a large cognitive burden on judges and incentivize them to take shortcuts.

Model	ROUGE-1	ROUGE-2	ROUGE-L
HipoRank + Different Positional Functions			
lead	37.43	12.13	33.68
undirected	40.66	13.41	36.55
boundary-distance (ours)	**43.42**	**16.76**	**39.23**
HipoRank + Different Hierarchical Functions			
w.o. hierarchy	41.88	15.39	37.91
w. hierarchy (ours)	**43.42**	**16.76**	**39.23**

Table 6: Results on the PubMed validation set with different positional function or hierarchical information.

port that our method's use of hierarchy and discourse structure improves summarization quality.

6 Ablation Studies

6.1 Component-wise Analysis

Table 6 presents the ablation study to assess the relative contributions of the boundary function and the hierarchical information. We keep all the hyperparameters unchanged with respect to the best setting in Section 4.2 and either vary the positional function or the hierarchical structures. We also found that the improvement of each components are stable across all the hyperparameters we tested (more details in the appendix).

The first block of Table 6 reports the ablation results with different positional functions: no positional function (Erkan and Radev, 2004; Mihalcea and Tarau, 2004), lead bias function (Zheng and Lapata, 2019), and our proposed boundary function. We can see that using the wrong positional function hurts the model's performance when comparing no positional function with lead bias function. Our boundary positional function outperforms the lead or no positional functions significantly.

The second block of Table 6 reports the results with or without the hierarchical structure. We observe that adding the hierarchical information results in a huge performance improvement.

6.2 Effect of Embeddings

To disentangle the effect of sentence representation, we show PubMed test set results of our best model with different sentence embeddings in Table 7. While pretrained transformer models finetuned on sentence similarity improve performance, HipoRank still consistently outperforms previous state-of-the-art unsupervised models (Table 3) even with random embeddings. These results once again suggest that our method's improvement can indeed be attributed to the use of hierarchy and discourse structure, rather than to the the choice of representations.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead	35.63	12.28	25.17
Oracle	55.05	27.48	38.66
HipoRank with Different Embeddings			
Random Embedding (d=200)	43.05	16.69	38.63
Biomed-w2v (d=200)	43.70	17.06	39.19
BERT (d=768)	42.91	16.27	38.52
PacSum-BERT (d=768)	43.58	17.00	**39.31**
SentBERT (d=768)	**43.59**	**17.08**	39.07
SentRoBERTa (d=1024)	43.55	17.06	39.07

Table 7: PubMed test set results with HipoRank framework and different pretrained sentence and section embeddings.

6.3 Stability of Hyperparameters

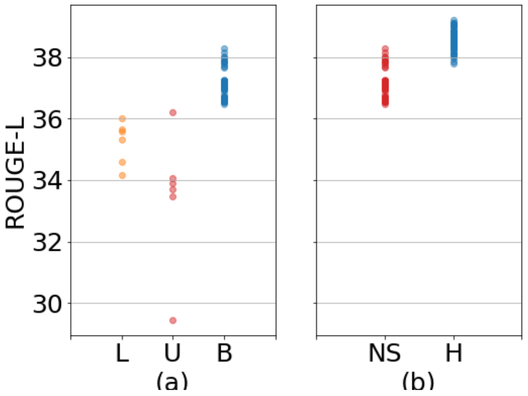


Figure 3: ROUGE-L scores for (a) different positional functions (L=lead, U=undirected, B=boundary) and (b) different graph hierarchies (NS=no_section, H=hierarchical). Each point corresponds to one configuration of the hyperparameter gridsearch described in Section 4.2.

To further inspect our model's stability across different hyperparameter choices, we conducted fine-grained analysis across all different hyperparameter settings as below.

Stability w.r.t. Discourse Structure To evaluate the impact and the stability of discourse structure informed edge weighting (Section 3.3), we first compared our *boundary* positional function (Eqn. 1,3) to PacSum's *lead* positional function, as well as the standard *undirected* approach over different hyperparameter settings. Figure 3 (a) shows that our method consistently performed better on the PubMed validation set, across *different hyperparameters and embedding models* outlined in Section 4.2.

Stability w.r.t. Hierarchy We then evaluated the effect of adding hierarchy (Section 3.2) on top of our boundary positional function. In addition to decreasing the computational cost, Figure 3 (b) shows that incorporating hierarchy further improved ROUGE-L consistently across *different hyperparameters and embedding models* we tested.

Application to other genres While our work here is focused on long scientific document summarization, we believe that our approach is promising for other genres of text, provided that the right discourse-aware biases are given to the model. Indeed, one version of our model with our proposed boundary function can be seen as a generalization of PACSUM, which achieves state-of-the-art performance on unsupervised summarization of news by exploiting the well known lead bias of news text (Zheng and Lapata, 2019; Grenander et al., 2019). We leave such explorations of adapting HIPORANK to other genres to future work.

7 Conclusion

We presented an unsupervised graph-based model for long scientific document summarization. The proposed approach augments the measure of sentence centrality by inserting directionality and hierarchy in the graph with boundary positional functions and hierarchical topic information grounded in discourse structure. Our simple unsupervised approach with rich discourse modelling outperforms previous unsupervised graph-based summarization models by wide margins and achieves comparable performance to state-of-the-art supervised neural models. This makes our model a lightweight but strong baseline for assessing the performance of expensive supervised approaches for long scientific document summarization.

Acknowledgments

This work is supported by the Natural Sciences and Engineering Research Council of Canada, Compute Canada, and the CIFAR Canada AI Chair program. We would like to thank Hao Zheng, Wen Xiao, and Sandeep Subramanian for useful discussions.

References

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sang-goo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 10–20.

Phyllis B Baxendale. 1958. Machine-made index for technical literature—an experiment. *IBM Journal of research and development*, 2(4):354–361.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621.

Arman Cohan and Nazli Goharian. 2015. Scientific article summarization using citation-context and article's discourse structure. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 390–400, Lisbon, Portugal. Association for Computational Linguistics.

Edward Collins, Isabelle Augenstein, and Sebastian Riedel. 2017. A supervised approach to extractive summarisation of scientific papers. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 195–205.

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Atefeh Farzindar and Guy Lapalme. 2004. Legal text summarization by exploration of the thematic structure and argumentative roles. In *Text Summarization Branches Out*, pages 27–34, Barcelona, Spain. Association for Computational Linguistics.

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

Matt Grenander, Yue Dong, Jackie Chi Kit Cheung, and Annie Louis. 2019. Countering the effects of lead bias in news summarization via multi-stage training and auxiliary losses. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6019–6024, Hong Kong, China. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1515–1520, Seattle, Washington, USA. Association for Computational Linguistics.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. 2017. Salience estimation via variational auto-encoders for multi-document summarization. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Maria Liakata, Simon Dobnik, Shyamasree Saha, Colin Batchelor, and Dietrich Rebholz-Schuhmann. 2013. A discourse-driven content model for summarising scientific articles evaluated in a complex question answering task. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 747–757, Seattle, Washington, USA. Association for Computational Linguistics.

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

Chin-Yew Lin and Eduard Hovy. 1997. Identifying topics by position. In *Fifth Conference on Applied Natural Language Processing*, pages 283–290.

Chin-Yew Lin and Eduard Hovy. 2002. From single to multi-document summarization. In *Proceedings of the 40th annual meeting of the association for computational linguistics*, pages 457–464.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3721–3731.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing. *Proceedings of LBM*, pages 39–44.

Edward Moroshko, Guy Feigenblat, Haggai Roitman, and David Konopnicki. 2019. An editorial network for enhanced document summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 57–63.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3075–3081.

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

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

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759.

Ani Nenkova, Kathleen McKeown, et al. 2011. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101.

Daraksha Parveen, Hans-Martin Ramsl, and Michael Strube. 2015. Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1949–1954.

Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific paper summarization using citation summary networks. *CoRR*, abs/0807.1560.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

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

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

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

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 664–674.

Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2210–2219.

Josef Steinberger and Karel Jezek. 2004. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM*, 4:93–100.

Sandeep Subramanian, Raymond Li, Jonathan Pilault, and Christopher Pal. 2019. On extractive and abstractive neural document summarization with transformer language models. *arXiv preprint arXiv:1909.03186*.

Simone Teufel. 1997. Sentence extraction as a classification task. In *Intelligent Scalable Text Summarization*.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: experiments with relevance and rhetorical status. *Computational linguistics*, 28(4):409–445.

Antoine Tixier, Polykarpos Meladianos, and Michalis Vazirgiannis. 2017. Combining graph degeneracy and submodularity for unsupervised extractive summarization. In *Proceedings of the workshop on new frontiers in summarization*, pages 48–58.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. 2007. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 755–762, Honolulu, Hawaii. Association for Computational Linguistics.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306.

Yuxiang Wu and Baotian Hu. 2018. Learning to extract coherent summary via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3002–3012.

Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

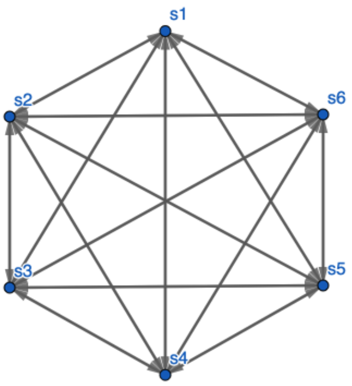
Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

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

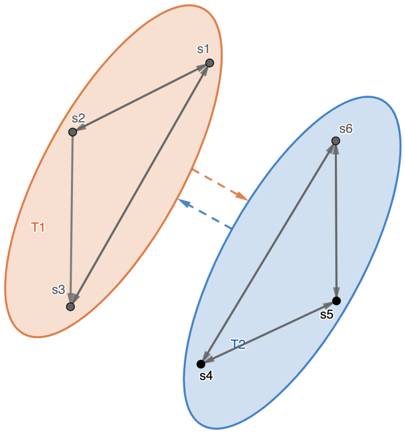
Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1049–1058.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663.

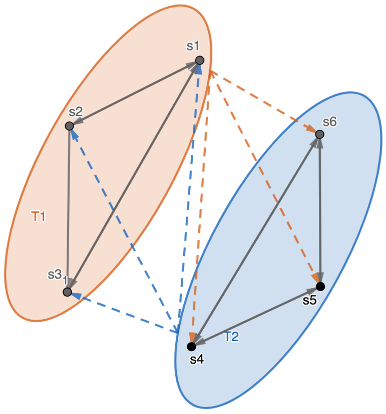
A Appendices



(a) Flat fully-connected graph



(b) Section-section hierarchical multiplication (hierarchy-multiply, ours)



(c) Section-sentence hierarchical addition (hierarchy-add, ours)

Figure 4: Comparison of the flat fully-connected graph used in Erkan and Radev (2004); Mihalcea and Tarau (2004); Zheng and Lapata (2019) to the hierarchical graph used in our models (b) and (c). Although the section-section multiplication reduces the edge computation proportionally to the number of sections, we found it oversimplifies the graph by assuming independence between sentences across different sections. Our final model loosens the assumption by including section-sentence connections as shown in sub-figure (c).

A.1 Different Hierarchical Structure

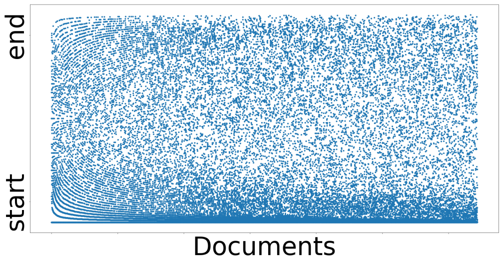
Besides our proposed hierarchical model (Figure 4 (c), hierarchy-add) in the paper, we also proposed and experimented with another novel hierarchical graph by introducing section-section connections (Figure 4 (b), hierarchy-multiply). In this hierarchical setting, we multiply a sentence's sectional importance with its sentence importance (Eqn. (2)) to form the final centrality score:

$$c(v_j^I) = \mu_1 \cdot c_{\text{inter}}(v_j^I) \times c_{\text{intra}}(v_j^I). \tag{7}$$

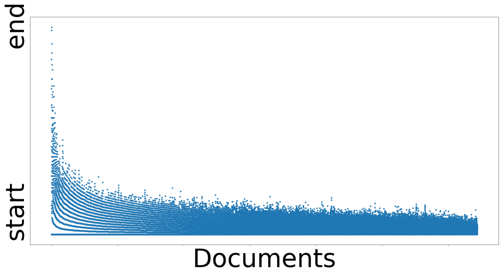
Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
Various Hierarchicall Centrality			
no-hierarchy	41.88	15.39	37.91
hierarchy-multiply (ours)	43.04	16.76	38.77
hierarchy-add (ours)	**43.42**	**16.76**	**39.23**

Table 8: Results on the PubMed validation set with different positional function or different hierarchical information.

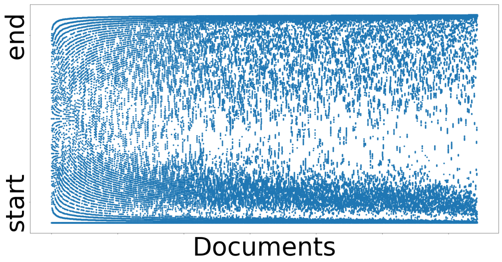
Our empirical results indicate the hierarchy-multiply model always outperforms no-hierarchy models ((Figure 4 (a)) but under performs hierarchy-add. Nevertheless, Table 8 shows that adding any hierarchical structure results in performance improvement by wide margins when compared to the no-hierarchy model.



(a) Oracle (arXiv)



(b) PACSUM (arXiv)



(c) HIPORANK (arXiv)

Figure 5: Sentence positions in source document for extractive summaries generated by different models on the arXiv validation set. Documents on the x-axis are ordered by increasing article length from shortest to longest.

A.2 Sentence Position Comparison

Figure 5 shows the sentence positions in source document for extractive summaries generated by different models on the arXiv validation set. We can again see that PACSUM is biased towards selecting sentences that appear at the beginning of a document while our method selects sentences in every section and near the article boundaries, similar to the oracle.

MIDAS: A Dialog Act Annotation Scheme for Open Domain Human Machine Spoken Conversations

Dian Yu
University of California, Davis
`dianyu@ucdavis.edu`

Zhou Yu
Columbia University
`zhouyu@cs.columbia.edu`

Abstract

Dialog act prediction in open-domain conversations is an essential language comprehension task for both dialog system building and discourse analysis. Previous dialog act schemes, such as SWBD-DAMSL, are designed mainly for discourse analysis in human-human conversations. In this paper, we present a dialog act annotation scheme, MIDAS (Machine Interaction Dialog Act Scheme), targeted at open-domain human-machine conversations. MIDAS is designed to assist machines to improve their ability to understand human partners. MIDAS has a hierarchical structure and supports multi-label annotations. We collected and annotated a large open-domain human-machine spoken conversation dataset (consisting of 24K utterances). To validate our scheme, we leveraged transfer learning methods to train a multi-label dialog act prediction model and reached an F1 score of 0.79.[1]

1 Introduction

Human-machine conversations have different dynamics compared to human-human conversations due to the power imbalance between humans and machines in conversational settings. Such differences include content and voice quality (Hill et al., 2015). For instance, humans tend to use a more authoritative voice when they talk to a machine. We found that "commands" account for about 9% of utterances in a human-machine social conversation corpus we collected, while a similar dialog act, "Action-Directive", only accounts for 0.4% in a human-human conversation corpus (Switchboard Dialog Act Corpus (SwDA) (Jurafsky et al., 1997)). Moreover, a human-machine dialog act scheme is not simply used to understand the dialog flow but also to help dialog systems plan their next steps. Therefore, a dialog act scheme for spoken dialog systems must capture semantic information necessary for fine-grained dialog planning. More importantly, a useful human-machine dialog act scheme should incorporate dynamics of realistic settings such as real-time automatic speech recognition (ASR) outputs. However, previously available dialog act schemes are all designed for manual dialog transcriptions, which are very different from ASR outputs. For instance, ASR outputs are noisy and may not contain punctuation. To test if a dialog act predictor trained on manual transcriptions would generalize to ASR outputs, we trained a model using the SwDA dataset annotated with the SWBD-DAMSL scheme (Jurafsky et al., 1997). We tested the model on the ASR output of our open-domain human-machine dialog system conversations and found that the model's performance is only 47.38% in accuracy, mostly due to incompatible schemes and confusion in annotation in addition to domain shifts. This low accuracy score suggests that using existing schemes and datasets to train models for human-machine dialog systems is impractical.

Scheme	#labels	context	multi-label	dataset
DAMSL	44	✓	✗	✓
TOPIC	14	✓	✗	✗
ISO	88	?	✓	✗
MIDAS	23	✓	✓	✓

Table 1: Comparison of different major dialog act schemes. TOPIC refers to the dialog act designed for topic modeling from Khatri et al. (2018). "?" indicates that guideline on context using is not clear (Mezza et al., 2018)

In this paper, we propose a hierarchical multi-label dialog act annotation scheme, MIDAS, specifically designed for real-time open-domain human-

---

[1] Code, data, and trained models are available at `https://github.com/DianDYu/MIDAS_dialog_act`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1103–1120
April 19 - 23, 2021.

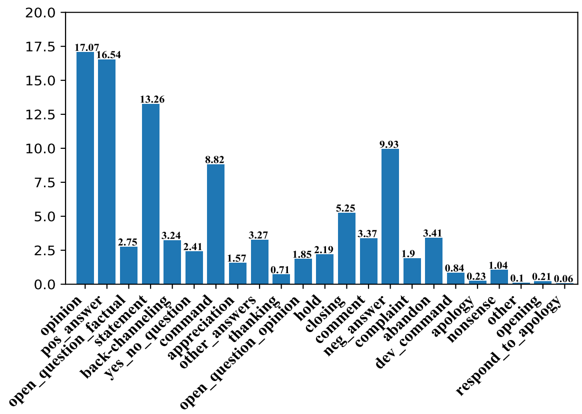


Figure 1: Distribution of dialog acts in the training set

machine spoken conversations. We show a comparison to major dialog act schemes in Table 1. We also annotated real-world human-machine social conversations using the MIDAS scheme and Figure 1 shows the distribution of dialog acts. The scheme is easy for humans to follow. Two annotators achieved an inter-annotator agreement of $\kappa = 0.94$. We trained a multi-label dialog act classifier using transfer learning methods and reached an F1 score of 0.79. Multiple Amazon Alexa Prize social chatbots have deployed our dialog act model and reported better conversational quality due to improved language understanding compared to using SWBD-DAMSL. We share our annotated data and trained models with the research community for easy adoption.

2 Related Work

Previous dialog act annotation schemes are mostly designed for task-oriented dialogs with a specific task, such as MapTask (Thompson et al., 1993) and DATE (Walker and Passonneau, 2001), or in a specific setting (Klüwer, 2011). There are a few dialog act schemes designed for human-human task-independent conversations, such as the Discourse Annotation and Markup System of Labeling (DAMSL, Core and Allen, 1997) and SWBD-DAMSL (Jurafsky et al., 1997). SWBD-DAMSL is used to annotate the Switchboard (Godfrey et al., 1992) corpus, a task-independent telephone conversation corpus, with inter-annotator agreement of $\kappa = 0.80$. SWBD-DAMSL is also applied to annotate human-human meeting conversations (Shriberg et al., 2004). In this paper, we design a dialog annotation scheme specifically for human-machine social chitchat conversations without any topic constraints.

Khatri et al. (2018) introduces a human-machine dialog act annotation scheme with 14 tags. However, the scheme is designed for modeling conversation topics instead of training dialog act predictors. The scheme has tags such as *Information Request*, *General Chat*, and *Multiple Goals*, and the annotation is performed on unsegmented user utterances. Even though the small number of tag categories makes annotation more reliable, it may not provide enough information for understanding user semantics. For example, tags such as *Multiple Goals* do not provide explicit information about user intent. In contrast, we propose a dialog act annotation scheme that focuses on improving open-domain dialog system understanding. We also build a dialog act predictor based on the annotated corpus.

Previously, most popular annotation schemes, such as DAMSL, use mutually-exclusive tags (Mezza et al., 2018) (0.01% of the labeled utterances have multiple labels so we do not consider DAMSL as multi-label) and each utterance is labeled with a single tag in SWBD-DAMSL (Stolcke et al., 2000). However, Bunt (2009) argues that conversation utterances are complex. Each functional segment can have four to five functions on average, so dialog act tags should serve multiple functions. Dynamic Interpretation Theory (DIT) (Bunt, 1997) and its extension, DIT++ (Bunt, 2009) try to solve this problem by supporting multi-dimension and multi-function tags. The 88 tags are organized in a hierarchical structure and separated into dimension-specific and general-purpose functions. The fifth version of DIT++, ISO 24617-2 (ISO standard, Bunt et al., 2010, 2017), is introduced to incorporate not only linguistic theory but also empirical discourse analysis on real domain-independent conversations. Although much effort has been expended in designing ISO, no large dataset was annotated with the scheme except DialogBank (Bunt et al., 2016). This is probably due to the complexity of the scheme and the lack of clear guidelines on how to use contextual information (Ribeiro et al., 2015; Mezza et al., 2018). Due to the huge complexity of open domain social conversations, we propose a hierarchical structure in the annotation scheme and allow one utterance to have multiple dialog acts. We limit the number of dialog acts to 23 to capture the fundamental acts necessary to the system while keeping the total number relatively small (two annotators reached 0.94 in Kappa). We

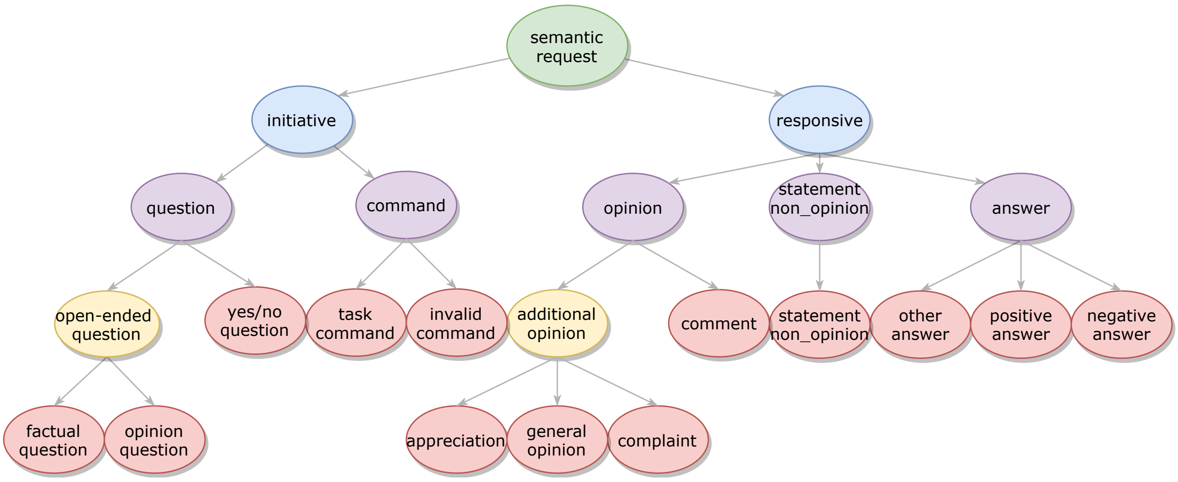


Figure 2: Semantic request tree. Scheme types, classes, categories, and sub-categories are in green, blue, purple, and yellow, respectively. Dialog act tags are leaf nodes in red. Tags can co-occur in one utterance, except tags under *opinion* and *statement non_opinion*, *question* and *answer* categories due to semantic and syntactic conflicts. For example, "User1:Do you watch TV shows? User2: I prefer watching movies." User2 is labeled both *general opinion* and *negative answer*.

also publish our annotated human-machine chatbot corpus through user studies containing 24,000 utterances.

3 MIDAS Annotation Scheme

We present MIDAS, a contextual hierarchical multi-label dialog act annotation scheme for human-machine conversations. MIDAS follows DIT++ and ISO (Bunt, 2009; Bunt et al., 2010) to ensure that the scheme facilitates both annotation and the training of automatic dialog act predictors. MIDAS focuses on helping dialog systems understand their human users, while previous schemes mainly focus on analyzing human-human dialog with not fully open-domain data (SWBD-DAMSL), mutually-exclusive tags (DAMSL), or lack of contextual information (DIT++ and ISO). Therefore, MIDAS provides a unique hierarchical structure and proposes a set of new labels for the human-machine setting while inheriting labels from previous schemes. A complete description of MIDAS is in Appendix A.2. Similar to Chowdhury et al. (2016); Mezza et al. (2018), we also show a potential mapping from SWBD, SWBD-DAMSL, and ISO to MIDAS in Appendix A.4. We discuss the three main features of MIDAS: hierarchical structure, multi-label format, and context consideration respectively.

3.1 Hierarchical structure

Previous schemes such as ISO design their hierarchical structure as multiple dimensions and define dialog acts as dimension-specific functions. Such a turn-by-turn specific taxonomy mixes in-depth analysis into the discourse level, thereby requiring a distinguished definition in each dimension and creating a complex hierarchy. For instance, "accept" and "decline" are defined individually for corresponding parent dimensions such as "address offer" and "address suggestion". This detailed design is beneficial to study interlocutors in a conversations, but may not be necessary for language understanding in a dialog system. In comparison, MIDAS focuses on a concise hierarchical structure to mainly distinguish semantic and functional intents which are critical for dialog understanding and planning. This design facilitates both annotation and model prediction in human-machine conversations.

Specifically, we design MIDAS to have a tree structure. It has two sub-trees: *semantic request* type (Figure 2) and *functional request* type (Figure 3). Under each type, there are classes, categories, and tags arranged in a hierarchical tree structure. Please refer to Figure 2 and 3 for detailed organization. Utterances are labeled with dialog act tags, which are the leaf nodes. The non-leaf nodes are used to categorize different dialog acts, help annotators find the correct tags, and assist customized requirements. We explain the reasoning for the hierarchical design and provide justification for the dialog act definitions with use cases.

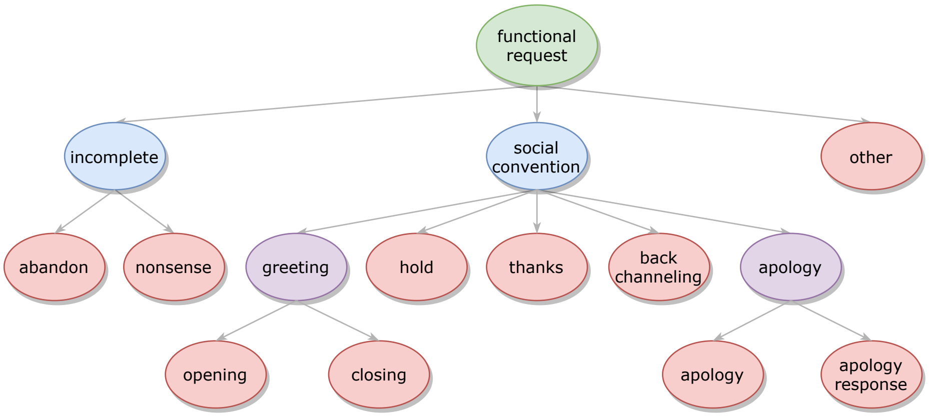


Figure 3: Functional request tree. We remove class and category nodes if there is only one label under them.

3.1.1 Semantic request

Semantic request type captures dialog content, therefore it is essential for dialog topic planning. *Semantic request* separates into *initiative* class and *responsive* class based on whether the user is proposing or continuing a topic.

Initiative class is especially important in the human-machine setting, because in such an imbalanced power setting, the machine must follow the topic that its human partner proposes. Therefore, understanding whether the user is proposing a new topic with their specific intent is the first step for the system to be coherent. There are two categories, *question* and *command* in the *initiative* class, that are designed to distinguish information requests (*question*) from action requests (*command*).

MIDAS first separates *question* into *yes/no question* and *open-ended question* based on syntax. This separation helps the system to generate a coherent response. For example, it is more natural for system responses to start with “yes” or “no” when replying to a *yes/no question*. Then MIDAS further divides *open-ended question* into *factual question* and *opinion question* based on the different types of information that users seek. The system can thus leverage this information and prepare responses by searching different knowledge bases. For example, *factual questions* require factual information from knowledge graphs such as Wikipedia, while *opinion question* requires information from corpora with opinionated material such as Twitter.

Unlike *question*, *command* conveys orders and is particularly popular in human-machine dialogs. The system needs to follow users’ commands, both implicit and explicit, because the system has less power in the conversation than its human interlocutors. Therefore, unlike previous schemes, MIDAS uses *task command* for task-oriented and device related requests, and introduces *invalid command*, which is specific to smart devices. Users sometimes produce commands that are beyond the system’s capability. For example, users may want to control device hardware to which the dialog system does not have access. The system needs to identify these utterances and handle them separately. Utterances labeled with *invalid command* are commands involving device functions which are specific to tasks and can be replied by templates and APIs. We present this tag in the leaf level to be consistent with other dialog acts in prediction.

Responsive class indicates that the utterance is a continuation of the previous topic. SWBD-DAMSL points out that opinions are often followed by general opinions, whereas statements are followed by back-channels (Jurafsky et al., 1997). This distinction may be subtle in human-human conversations (Jurafsky et al., 1997) as humans do not need to explicitly distinguish between the two tags to generate corresponding responses. However, knowing whether an utterance is a statement or an opinion is essential for a system to generate an appropriate response. Hence, MIDAS further breaks the responsive class into *opinion*, *statement non_opinion*, and *answer*, based on the conversation history.

MIDAS separates the *opinion* category into the *additional opinion* subcategory and the *comment* tag because we observed examples such as “User1: my friend thinks we are living in matrix. User2: she’s probably right”. User2 comments on the previous utterance without contributing extra information. *additional opinion* indicates that utterances labeled as dialog acts under this subcategory may contribute extra information, whereas *comment* often indicates that an utterance is a simple reply without explicit feedback. MIDAS separates three types of opinions: *appreciation*, *complaint*, and *general opinion*. Understanding whether the user is complaining or praising the system is essential. Such information is critical to influence dialog policies and extract user feedback.

We also split *answer* into *positive answer*, *negative answer*, and *other answer*, based on utterance's sentiment. One caveat is that utterances, such as "why not", contain negative words but are actually a positive answer to questions such as "Can we talk about movies?", and thus should be labeled as *positive answer*. Such phenomena suggest that automatic dialog act prediction models require semantic understanding and incorporate context in feature representation.

3.1.2 Functional request

Functional request type helps dialog systems achieve discourse level coherence and control conversation functions. We define *incomplete*, *social convention*, and *other* classes under the *functional request* type.

Incomplete class describes utterances that are not complete. There are two *incomplete* types, *abandon* and *nonsense*. In real-world settings, human users can be cut off due to issues such as background noise and long pauses. These cases are labeled as *abandon*. By comparison, *nonsense* is used to label utterances that human annotators cannot understand. These utterances usually contain many ASR errors.

Social convention class is similar to the *social obligations management* and *discourse structure management* dimensions in ISO (Bunt et al., 2010). We define *opening*, *closing*, *thanks*, *apology*, *apology response*, *hold*, and *back channeling* to provide discourse level information.

Finally, utterances that cannot be assigned to any other tag in this hierarchical structure are labeled as an *other* tag.

3.2 Multi-label support

Compared to single-label schemes, multi-label schemes capture different dimensions and functions, which better support dialog system building and discourse analysis. For example,

User1: *what books have you read recently*
User2A: *i haven't read any*
User2B: *i don't want to talk about books*
User2C: *i prefer watching movies*

Users may use different strategies to express a *negative answer* intent. A single-label scheme cannot differentiate above three sentences. An extra label that captures additional semantic information besides *negative answer* benefits dialog system understanding. For instance, User2B has the additional *task command* intent that requests to end the current topic compared to User2A. User2C has the *general opinion* intent that initiates a different topic. Dialog system may not need to change a topic for User2A, but it needs to change the topic for User2B and User2C.

SWBD-DAMSL allows a utterance to be tagged as *double labels* and lists the preferred tag first (Jurafsky et al., 1997). However, *double labels* and ordering heuristics are not explicitly defined in the scheme and thus are infrequent in Switchboard. In MIDAS, except for two exclusive category pairs (*opinion* and *statement non_opinion*, *question* and *answer*), dialog acts are designed to be compatible across hierarchies so that labels in each category can co-occur in one utterance.

Due to the power difference in human-machine conversations, the system needs to prioritize certain tags over the others to keep a coherent conversation. MIDAS has a priority list to focus, from high to low: *question*, *command*, *answer*, *opinion*, and *statement non_opinion*. For example, "User1: what do you want to talk about? User2: how about the financial market". User2's utterance can be tagged as *task command*, *opinion question*, and *general opinion*. Among the three tags, *opinion question* and *task command* are more useful for the system to direct the conversation towards a specific topic.

3.3 Context consideration

Mezza et al. (2018) suggests that most dialog act schemes, including ISO, are not clear on how to leverage contextual information in the annotation process. So tags such as "Answers" are confused with "Inform", which leads to noisy annotations. In comparison, MIDAS scheme is designed to consider context utilizing the hierarchical structure. Dialog acts are thus explicitly distinguished by contextual information. For instance, *general opinion* and *comment* both refer to personal views, but they belong to different sub-categories capturing different semantics and expectations from interlocutors given a specific context. Correspondingly, during annotation, annotators are instructed to consider the context from previous two turns and locate each level in the structure tree suggested by the context.

4 Dataset and Annotation Process

We collected 380K human-machine conversations ASR outputs using Gunrock, the 2018 Alexa Prize winning social bot (Chen et al., 2018). The average number of turns in a conversation and the average tokens per user utterance is 21.76 and 2.85, respectively. Table 2 shows a sample conversation. Two annotators read the descriptions and examples of each dialog act illustrated in Appendix A.2. The annotators reached an inter-annotator agreement of $\kappa = 0.94$ on 1,185 segmented utterances verified by the scheme designers (compared to $\kappa = 0.80$ using DAMSL reported by Stolcke et al. (2000)). Then they annotated the rest of the data separately. In total, they annotated 468 conversations, including 24K segmented sentences on both user and system, among which 12.9K segments are from users. Note that if one dialog act spans multiple segments, we annotate each segment with the same dialog act tag. *general opinion* and *statement non_opinion* are the most frequent tags. For multiple labels, (*positive_answer*, *command*) and (*negative_answer*, *command*) are the most frequent co-occurring tags. Detailed annotation results, including dialog act tag distribution, are shown in Appendix A.2 and Appendix A.3.

USER1:	what do you want to talk about
USER2:	what can you tell me what the top books are right now
USER1:	i am so excited to talk to you about books. i'm actually a pretty big bookworm, and i love to read when i'm not chatting
USER2:	oh [SEG] what are some titles of the books you've read
USER1:	recently, i'm reading the great gastby. it's really thought provoking, and i can see why some people call it the great american novel. how about you? what book do you like?
USER2:	i have {not} read a book in a while [SEG] do you have recommendations in the sci fi

Table 2: An example conversation between a machine (USER1) and a human (USER2). The word "not" is dropped in the last sentence due to ASR errors.

We note that there may be some limitations in the corpus we annotated in terms of both ASR results, and the current interactions between humans and machines. Our motivation is that compared to carefully corrected transcriptions such as SwDA, our scheme and annotated corpus can be more robust against ASR errors and noisy inputs. Since the proportion of utterances with ASR errors are relatively small in our corpus, models trained with the corpus should still perform well on utterances without such errors from more advanced ASR systems. In addition, we believe that our hierarchical multi-label schemes with a manageable size and structure can be easily modified to customized models, as well as more advanced human-machine conversational models. Moreover, our scheme is designed to bridge the gap in human and machine interactions, and are not restricted by the annotation corpus. More importantly, this scheme and annotated corpus can be extended outside of predictions and understanding tasks to language generation tasks. We leave detailed comparison and application to additional tasks such as language generation to future work.

For dialog act prediction, user utterances in human-machine dialogs are ASR outputs and have no punctuation. Therefore, we train a model to segment utterances into semantic units for preprocessing. Note that utterance segmentation is a different line of research and is out of the scope of this paper. We focus on dialog act prediction on each segmented unit following Stolcke et al. (2000). Previous research detects sentence boundaries by predicting the exact punctuation in the training dataset (Cho et al., 2015). However, correct punctuation also relies on deep semantic understanding beyond the sentence surface forms. For instance, a misused question mark can lead the dialog act model to predict a sentence as a question. So following Favre et al. (2008), we only predict the boundary of a sentence instead of predicting punctuation to avoid introducing errors.

It is expensive to annotate sentence boundaries, so we use the Cornell Movie-Quotes Corpus (Danescu-Niculescu-Mizil and Lee, 2011) to train a sentence segmentation model. The Cornell dataset contains 300K utterances from movie transcripts. We reformat the transcripts by replacing punctuation to sentence breaker tokens (denoted as [SEG]). We then train a sequence-to-sequence (seq2seq) model (Sutskever et al., 2014) to predict sentence breaker similarly to Klejch et al. (2017) and Peitz et al. (2011). The input to the model is a reformatted sentence, and the output is the same sentence with added sentence breaker tokens. An example can be seen in the last *USER2* utterance in Table 2. Word embeddings are pre-trained with fastText

(Mikolov et al., 2018) using Common Crawl. We evaluate the segmentation model on 2k manually labeled human utterances of the collected data. The segmentation model achieves 84.43% in micro F1 score, 84.97% in precision, and 84.57% in recall. We apply the trained segmentation model on the entire collected dataset to obtain segmented sentences. All the dialog act annotation and predictions are done on the automatic segmentation results.

5 Dialog Act Prediction

We formulate the dialog act prediction problem as a multi-label classification problem. Following Katakis et al. (2008), we evaluate the proposed methods using F1 score. For evaluation simplicity, we predict one or two labels in the following experiments. We leverage both unlabeled data and annotated data to improve classification performance.

5.1 Baseline model

RNN models have shown promising results in text classification (Rojas-Barahona et al., 2016). Our baseline model uses a 2-layer Bi-LSTM to encode the context representation and a multi-layer perceptron (MLP) to decode the output. For multi-label prediction, we use a binary cross-entropy objective function to learn co-occurring tags independently. This training objective also helps with transfer learning from other single label dialog act datasets. During testing, we choose the labels with the highest values predicted from the MLP as the potential output and filter them with an empirical threshold (0.5).

5.2 Context representation

Contextual information plays an important role in dialog act prediction (Liu et al., 2017; Khatri et al., 2018). We consider two methods to represent previous turns: the actual utterance (text), and the dialog act of the utterance (DA). For each method, the most recent segmented sentence unit from each speaking party is considered as the history suggested by the length of context from Khatri et al. (2018). We append the last segmented system unit (sys_unit), the previous segmented user unit (user_prev), and the current segmented user unit (user_cur) as sys_unit $<u\_p>$ $user\_prev$ $<u\_c>$ $user\_cur$ where $<u\_p>$ and $<u\_c>$ are special tokens to separate utterances. For instance, to predict the dialog act for the segment "do you have recommendations in the sci fi" in the last *USER2* utterance in Table 2, the context representation is formed as *what book do you like <u_p> i haven't read a book in a while <u_c> do you have recommendations in the sci fi*. However, if the current utterance is the first one in the current turn, i.e. there is no contextual information, we use an empty token for usr_prev instead.

Another method to incorporate history is to replace the actual previous segment unit with its dialog act labels (if there are two labels for one segment, we combine both labels). The results for these two methods are shown in Table 3.

5.3 Transfer learning

We use an unsupervised domain adaptation task and a supervised dialog act predictor trained on SwDA (Jurafsky et al., 1997) to improve model performance. We used a BERT model trained on Wikipedia (Devlin et al., 2019) to leverage its contextual word embeddings. However, one potential drawback of using pre-trained BERT is that it has a different domain from our conversational data. Inspired by Siddhant et al. (2019), we use 50 million unlabeled segmented utterances from 380K conversations from Gunrock to fine-tune the BERT language model to resolve this issue. We also leverage an annotated dataset from a similar task. We automatically map 42 tags from SWBD-DMSL to our 23 tags. The detailed mapping can be found in Appendix A.4. We remove all the punctuation (except apostrophes) and non-verbal information such as "<laugh>" from the carefully annotated dataset. We also remove sentences with dialog acts that are not applicable to ours such as *3rd-party-talk*. After pre-processing, we extract a total of 200K annotated utterances using the context representation in Section 5.2. We train a single-label prediction model based on BERT before fine-tuning it on our multi-label prediction task.

6 Experiments

Setting The main purpose of a dialog act predictor in human-machine dialogs is to improve the dialog system's understanding of user intent and preparing corresponding responses. Therefore, we build a dialog act prediction model on user utterances only. After pre-processing (refer to Section 5.2), there are 12.9K user segments, 13.78% of which have multiple labels. We use 10.3K for training and 2.6K for testing. All the testing data are from in-lab user studies. In order to protect Amazon Alexa

	Pre(%)	Rec(%)	F1(%)
LSTM-text	75.94	75.91	75.51
LSTM-DA	75.83	73.48	73.77
BERT-text	79.57	79.31	79.11
BERT-DA	79.29	76.12	76.87
BERT-no_context	73.88	70.43	71.30
BERT-DA+text	79.79	79.47	79.28
BERT_F-text	79.83	79.64	79.40
BERT_F-DA	79.30	76.15	76.89
BERT_F-DA+text	**79.93**	79.61	**79.44**
BERT-SwDA	79.26	76.43	78.98
BERT-SwDA_F	79.58	**79.76**	79.28

Table 3: BERT_F-DA+text achieves the best precision and F1 score. Results reported are an average score of six different random seed runs.

user's privacy, we only make the 2.6K annotated testing utterances public as they were collected with consent for releasing the data to the public.

Models. We implemented 11 models. We use **LSTM** to represent the baseline model trained with LSTMs. **BERT** represents Transformer models with a pre-trained BERT language model. Based on different transfer learning methods described in Section 5.3, **BERT_F** is a pre-trained BERT language model fine-tuned on unlabeled in-domain data, whereas **BERT_SwDA** is a pre-trained BERT language model fine-tuned on labeled SwDA task. Combining these two methods, **BERT-SwDA_F** is fine-tuned on both the unlabeled and labeled tasks. After fine-tuning, the models are trained on our data annotated with the MIDAS scheme. To evaluate the impact of context representation for the above models, we use **-text** and **-DA** to represent using text and dialog act as the context, respectively. We denote **-no_context** when predicting the current utterance without any context.

See Appendix A.1 for implementation details.

7 Results and Analysis

Table 3 describes the experimental results on all 11 models. Transformer models (Vaswani et al., 2017) using BERT embeddings (BERT-text) outperform Bi-LSTM models with pre-trained word embeddings (LSTM-text) by a large margin (from 75.51% to 79.11% in F1). If we further fine tune the BERT language model on an unsupervised training task with similar data distribution (BERT_F-text), the classification result further improves from 79.11% to 79.40% in F1. This is consistent with previous research on in-domain pre-training (Siddhant et al., 2019). However, the performance improvement is not statistically significant. One possible reason is that models pre-trained on a very large text dataset, such as Wikipedia, already encode sufficient semantics for dialog act prediction. Therefore, fine-tuning the model on a more domain-aligned data set does not significantly improve the performance.

We found that incorporating context improves the model performance. Adding text information as context improves the BERT model from 71.30% to 79.11% in F1. We also compare the impact of different context embedding methods on dialog act classification performance. The results show that replacing text with dialog act achieves a high precision, but suffers from a low recall. This is because an utterance can have multiple intents while dialog act itself does not provide enough context information to achieve accurate prediction. For example, when "i don't think so" is a response to a simple *yes/no question* such as "have you read the book", it is a *negative_answer*. But if it is a response to a more complex *yes/no question*, such as "do you want to talk about books", then it has two tags, *command* and *negative_answer*. The latter conveys user's implicit request on changing the topic. Therefore, only using dialog act as context could lead to high recall but low F1. We found combining both the previous segment's dialog act label and its surface text achieves the best performance in F1 (79.44%). However the performance improvement over including text only is not statistically significant. This suggests that dialog act and text may contain more overlapping information than complementary information.

We also found that fine-tuning the model using the supervised dialog act prediction task on the SwDA data did not improve performance in F1 but improved recall slightly. The reduced performance may be due to difference in the data. Even though both datasets are open domain conversational data, the SwDA task uses pre-processed Switchboard data that does not have ASR errors. Moreover, SwDA is human-human conversations, and they are more coherent and consistent compared to human-machine conversations. Another reason is that SwDA dataset has exactly one label for each utterance. When fine-tuning on our multi-label task, the pre-trained single-label model may tend to predict more labels to quickly reduce loss but fail to learn better representations.

We further looked into the errors made by our best model (*BERT_F-DA+text*) and found that the model confuses *statement non_opinion* and *general opinion*. This is most likely caused by only including one turn context. Sometimes, users have questions that break the conversation flow, such as “can you say it again clearly”. The model needs to consider not only this utterance but also the prior turns to perform dialog act prediction. We plan to incorporate longer context in future work. In addition, some of the *nonsense* sentences are misclassified as *statement non_opinion* such as “it doesn’t outside break a car”. It is also worth noting that some incorrectly segmented units resulted in inaccurate dialog act prediction. For instance, the utterance “we are they love each other” is incorrectly segmented into “we” and “are they love each other” given the context “you must be great friends with your dogs”. The second segment is incorrectly predicted as a *yes/no question*.

8 Conclusion

We propose MIDAS, a dialog act scheme designed for open-domain human-machine conversational systems. MIDAS is a hierarchical annotation scheme that supports multiple labels. We annotated 24K sentences from a human-machine social conversation dataset using MIDAS. We also trained dialog act classification models based on the annotated dataset. We tested different transfer learning techniques to improve model performance. We found that fine-tuning a pre-trained BERT model on unannotated target human-machine conversations improved model performance. However, fine-turning the model on a supervised dialog act task with human-human conversations did not improve model performance. MIDAS has been deployed in real world dialog systems, demonstrating its efficacy in the open-domain dialog setting.

Beyond prediction in open-domain dialogs for human request understanding, our proposed scheme can also be applied to task-oriented dialogs together with chit-chats. Furthermore, MIDAS can be used in end-to-end dialog systems for cases such as probing and model interpretation. In addition, our scheme and method can be extended to controlled text generation. We will explore broader applications in future work.

9 Acknowledgement

We thank Kenji Sagae for valuable discussions and suggestions. We also thank Sam Davidson and anonymous reviewers for their helpful comments. Dian Yu is supported by the National Science Foundation under Grant No. 1840191. Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the NSF.

References

Harry Bunt. 1997. Dynamic interpretation and dialogue theory.

Harry Bunt. 2009. The dit++ taxonomy for functional dialogue markup. In *AAMAS 2009 Workshop, Towards a Standard Markup Language for Embodied Dialogue Acts*, pages 13–24.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, Claudia Soria, and David Traum. 2010. Towards an ISO standard for dialogue act annotation. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC’10)*, Valletta, Malta. European Languages Resources Association (ELRA).

Harry Bunt, Volha Petukhova, and Alex Fang. 2017. Revisiting the iso standard for dialogue act annotation. In *Proceedings 13th Joint ISO - ACL Workshop on Interoperable Semantic Annotation (isa-11)*, pages 37–50. Research Unit(s) information for this publication is provided by the author(s) concerned.

Harry Bunt, Volha Petukhova, Andrei Malchanau, Kars Wijnhoven, and Alex Fang. 2016. The DialogBank. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3151–3158, Portorož, Slovenia. European Language Resources Association (ELRA).

Chun-Yen Chen, Dian Yu, Weiming Wen, Yi Mang Yang, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, Giritheja Sreenivasulu, Runxiang Cheng, Ashwin Bhandare, and Zhou Yu. 2018. Gunrock: Building a human-like social bot by leveraging large scale real user data. In *2nd Proceedings of Alexa Prize*.

Eunah Cho, Kevin Kilgour, Jan Niehues, and Alexander H. Waibel. 2015. Combination of nn and crf models for joint detection of punctuation and disfluencies. In *INTERSPEECH*.

Shammur Absar Chowdhury, Evgeny Stepanov, and Giuseppe Riccardi. 2016. Transfer of corpus-specific dialogue act annotation to ISO standard: Is it worth it? In *Proceedings of the Tenth International Conference on Language Resources and*

Evaluation (LREC 2016), pages 132–135, Portorož, Slovenia. European Language Resources Association (ELRA).

Mark G. Core and James F. Allen. 1997. Coding dialogs with the damsl annotation scheme. In *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*.

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

B. Favre, D. Hakkani-Tur, S. Petrov, and D. Klein. 2008. Efficient sentence segmentation using syntactic features. In *2008 IEEE Spoken Language Technology Workshop*, pages 77–80.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1.

Jennifer Hill, W. Randolph Ford, and Ingrid G. Farreras. 2015. Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations. *Computers in Human Behavior*, 49:245 – 250.

Dan Jurafsky, Liz Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. Technical Report Draft 13, University of Colorado, Institute of Cognitive Science.

Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. 2008. Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge*.

C. Khatri, R. Goel, B. Hedayatnia, A. Metanillou, A. Venkatesh, R. Gabriel, and A. Mandal. 2018. Contextual topic modeling for dialog systems. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 892–899.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

O. Klejch, P. Bell, and S. Renals. 2017. Sequence-to-sequence models for punctuated transcription combining lexical and acoustic features. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5700–5704.

Tina Klüwer. 2011. “i like your shirt” - dialogue acts for enabling social talk in conversational agents. In *Intelligent Virtual Agents*, pages 14–27, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yang Liu, Kun Han, Zhao Tan, and Yun Lei. 2017. Using context information for dialog act classification in dnn framework. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2170–2178. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Stefano Mezza, Alessandra Cervone, Evgeny Stepanov, Giuliano Tortoreto, and Giuseppe Riccardi. 2018. Iso-standard domain-independent dialogue act tagging for conversational agents. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3539–3551. Association for Computational Linguistics.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Stephan Peitz, Markus Freitag, Arne Mauser, and Hermann Ney. 2011. Modeling punctuation prediction as machine translation. In *IWSLT*.

Eugénio Ribeiro, Ricardo Ribeiro, and David Martins de Matos. 2015. The influence of context on dialogue act recognition. *CoRR*, abs/1506.00839.

Lina M. Rojas-Barahona, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, Stefan Ultes, Tsung-Hsien Wen, and Steve Young. 2016. Exploiting sentence and context representations in deep neural models for spoken language understanding. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 258–267, Osaka, Japan. The COLING 2016 Organizing Committee.

Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004*, pages 97–100, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Aditya Siddhant, Anuj Goyal, and Angeliki Metallinou. 2019. Unsupervised transfer learning for spoken language understanding in intelligent agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4959–4966.

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Comput. Linguist.*, 26(3):339–373.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Henry S. Thompson, Anne Anderson, Ellen Gurman Bard, Gwyneth Doherty-Sneddon, Alison Newlands, and Cathy Sotillo. 1993. The hcrc map task corpus: Natural dialogue for speech recognition. In *Proceedings of the Workshop on Human Language Technology*, HLT '93, pages 25–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Marilyn Walker and Rebecca Passonneau. 2001. DATE: A dialogue act tagging scheme for evaluation of spoken dialogue systems. In *Proceedings of the First International Conference on Human Language Technology Research*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

A Appendices

A.1 Implementation Details

The baseline dialog act prediction model uses a 2-layer Bi-LSTM with a hidden size of 500. The LSTM layers use a dropout rate of 0.3. We optimize the model with Adam optimizer (Kingma and Ba, 2014). For the Transformer models, we use 12 layers with 12 attention heads and a hidden size of 768. All the fully connected layers use a dropout rate of 0.1. Because one utterance may have multiple labels, following Katakis et al. (2008), we calculate precision, recall, and F1 for multilabel classification) on each sample and then average them across all samples (micro F1). We use the efficient Transformer implementation (Wolf et al., 2019) for our experiments.

For sentence breaker seq2seq model, both the encoder and the decoder are 2-layer 500-dimension bi-LSTMs. In addition, the decoder uses global attention and input feed (Luong et al., 2015) with beam search.

A.2 Dialog Act Scheme

Dialog Act - Semantic request		
Dialog Act Tag	**Description**	**Example**
factual question	factual questions	How old is Tom Cruise; How's the weather today
opinion question	opinionated questions	What's your favorite book; what do you think of disney movies
yes/no question	yes or no questions	Do you like pizza; did you watch the game last night
task command	commands/requests (can be in a question format) for some actions that may be different from the ongoing conversation	can i ask you a question; let's talk about the immigration policy; repeat
invalid command	general device/system commands that cannot be handled by the social bot	show me a picture; cook food for me
appreciation	appreciation towards the previous utterance	that's cool; that's really awesome
general opinion	personal view with polarized sentiment	dogs are adorable; (A: How do you like Tom) B: i think he is great
complaint	complaint about the response from another party	I can't hear you; what are you talking about; you didn't answer my question
comment	comments on the response from another conversation party	(A: my friend thinks we live in the matrix) B1: she is probably right; B2: you are joking, right; B3: i agree; (A: ... we can learn a lot from movies ...) B: there is a lot to learn; (A: He is the best dancer after michael jackson. What do you think) B: michael jackson
statement non_opinion	factual information	I have a dog named Max; I am 10 years old; (A: what movie have you seen recently) B: the avengers
other answer	answers that are neither positive or negative	I don't know; i don't have a favorite; (A: do you like listening to music) B: occasionally
positive answer	positive_answers	yes; sure; i think so; why not
negative answer	negative response to a previous question	no; not really; nothing right now

Table 4: Dialog Act Scheme for Semantic request

Dialog Act - Functional request		
Dialog Act Tag	**Description**	**Example**
abandon	not a complete sentence	So uh; I think; can we
nonsense	utterances that do not make sense to humans	he all out
hold	a pause before saying something	let me see; well
opening	opening of a conversation	hello; hi
closing	closing of a conversation	nice talking to you; goodbye
thanks	expression of thankfulness	thank you
back-channeling	acknowledgement to the previous utterance	Uh-huh; (A: i learned that ...) B: okay/yeah/right/really?
apology	apology	I'm sorry
apology response	response to apologies	That's all right
other	utterances that cannot be assigned to other tags	

Table 5: Dialog Act Scheme for Functional request

A.3 Multi-functionality schemes

Multi-label tags		
Dialog Act Tags	**Example**	**Count in User Utterances**
positive answer, task command	(A: wanna know something interesting about it?) B: sure; (A: do you want to talk about some games) B: minecraft	698
negative answer, task command	(A: would you like to know more about it) B: I don't want to hear more	328
task command, general opinion	(A: what do you want to talk about) B: harry potter stuff	192
task command, statement non_opinion	let's talk about mario kart	141
positive answer, statement non_opinion	(A: have you read any books recently?) B: I'm reading the great gatsby	133
task command, yes/no question	do you know tom brady; (A: what do you want to talk about?) B: how about movies	116
negative answer, statement non_opinion	(A: do you have pets) B: I don't have any	66
positive answer, general opinion	(A: do you like animals) B: My favorite animals is panda	35
invalid command, yes/no question	can you speak louder	15
task command, factual question	what do you know about dodgers	12
negative answer, general opinion	(A: do you watch sports) B: I'm not into sports	10
task command, opinion question	(A: what did you find interesting recently) B: what do you think of the new movie	9
task command, complaint	I don't want to hear you talk about anything; would you stop asking me that question	5
other answer, general opinion	(A: what's your favorite movie) B: there are so many to choose from	5
positive answer, comment	(A: don't you think so) B: it's true	4
general opinion, yes/no question	(A: what would you imagine doing in such situation) B: can we just sleep all day	3
negative answer, comment	(A: isn't that interesting) B: that's ridiculous	3
general opinion, opinion question	(A: what book would you recommend me to read) B: how about antifragile	3

A.4 Dialog act tag mapping

SWBD-DAMSL	**SWBD**	**MIDAS**
statement_non_opinion	*sd*	*statement non_opinion*
Acknowledge (Backchannel)	*b*	*back-channeling*
Statement-opinion	*sv*	*general opinion*
Agree/Accept	*aa*	*positive answer*
Abandoned or Turn-Exit	*% -*	*abandon*
Appreciation	*ba*	*appreciation*
Yes-No-Question	*qy*	*yes-no question*
Non-verbal	*x*	
Yes answers	*ny*	*positive answer*
Conventional-closing	*fc*	*closing*
Uninterpretable	*%*	*abandon*
Wh-Question	*qw*	
No answers	*nn*	*negative answer*
Response Acknowledgement	*bk*	*back-channeling*
Hedge	*h*	*other answers*
Declarative Yes-No-Question	*qyˆd*	*yes-no question*
Other	*o,fo,bc,by,fw*	*other*
Backchannel in question form	*bh*	*back-channeling*
Quotation	*q̂*	*general opinion*
Summarize/reformulate	*bf*	*general opinion*
Affirmative non-yes answers	*na, nyˆe*	*positive answer*
Action-directive	*ad*	*task command*
Collaborative Completion	*ˆ2*	*general opinion*
Repeat-phrase	*bˆm*	*general opinion*
Open-Question	*qo*	
Rhetorical-Questions	*qh*	
Hold before answer/agreement	*ˆh*	*hold*
Reject	*ar*	*negative answer*
Negative non-no answers	*ng,nnˆe*	*negative answer*
Signal-non-understanding	*br*	*complaint*
other_answers	*no*	*other answer*
Conventional-opening	*fp*	*opening*
Or-Clause	*qrr*	*other*
Dispreferred answers	*arp,nd*	*negative answer*
3rd-party-talk	*t3*	
Offers, Options Commits	*oo,cc,co*	*other*
Self-talk	*t1*	*other*

SWBD-DAMSL	**SWBD**	**MIDAS**
Downplayer	*bd*	*apology response*
Maybe/Accept-part	*aap/am*	*positive answer*
Tag-Question	*ˆg*	*other*
Declarative Wh-Question	*qwˆd*	
Apology	*fa*	*apology*
Thanking	*ft*	*thanking*

Table 6: Dialog act tag mapping among SWBD-DAMSL, SWBD, and MIDAS

MIDAS	**ISO**
factual question	*Question*, *Set-Question*, *Choice-Question*
opinion question	*Question*, *Set-Question*, *Choice-Question*
yes/no question	*Propositional-Question*, *Check-Question*
task command	*Instruct*, *Suggest*, *Request*, *Offer*, *Promise*
invalid command	*Instruct*, *Suggest*, *Request*, *Offer*, *Promise*
appreciation	*AutoPositive*, *AlloPositive*
general opinion	*Inform*
complaint	*AutoNegative*, *AlloNegative*
comment	*AutoPositive*, *AutoNegative*, *AlloPositive*, *AlloNegative*
statement non_opinion	*Inform*, *Init-Self-Introduction*, *Return Self-Introduction*
other answer	*Address-Offer*, *Address-Request*, *Address-Suggest*, *Answer*, *Correction*, *Disagreement*, *Agreement*
postive answer	*Confirm*, *Accept-Offer*, *Accept-Request*, *Accept-Suggest*, *Agreement*
negative answer	*Disconfirm*, *Decline-Offer*, *Decline-Suggest*, *Decline-Request. Disagreement*
abandon	
nonsense	
opening	*Init-Greeting*, *Return Greeting*
closing	*Init-Goodbye*, *Return Goodbye*
hold	*Staling*, *Pausing*
thanks	*Thanking*, *Accept Thanking*
back-channeling	*Propositional-Question*
apology	*Apology*
apology response	*Accept Apology*
other	*Other*

Table 7: Dialog act tag mapping between MIDAS and ISO communicative functions

Analyzing the Forgetting Problem in Pretrain-Finetuning of Open-domain Dialogue Response Models

Tianxing He
MIT
tianxing@mit.edu

Jun Liu
Facebook
junliu@fb.com

Kyunghyun Cho
New York University
kyunghyun.cho@nyu.edu

Myle Ott & Bing Liu
Facebook
{myleott,bingl}@fb.com

James Glass
MIT
glass@mit.edu

Fuchun Peng
Facebook
fuchunpeng@fb.com

Abstract

In this work, we study how the finetuning stage in the pretrain-finetune framework changes the behavior of a pretrained neural language generator. We focus on the transformer encoder-decoder model for the open-domain dialogue response generation task. Our major finding is that after standard finetuning, the model *forgets* some of the important language generation skills acquired during large-scale pretraining. We demonstrate the forgetting phenomenon through a set of detailed behavior analysis from the perspectives of knowledge transfer, context sensitivity, and function space projection. As a preliminary attempt to alleviate the forgetting problem, we propose an intuitive finetuning strategy named "mix-review". We find that mix-review effectively regularizes the finetuning process, and the forgetting problem is alleviated to some extent. Finally, we discuss interesting behavior of the resulting dialogue model and its implications.

1 Introduction

Large-scale unsupervised pretraining (Peters et al., 2018; Devlin et al., 2018; Song et al., 2019; Yang et al., 2019; Liu et al., 2019) has recently been shown to greatly boost the performance of natural language processing (NLP) models. On a high level, the pretrain-finetune framework can be viewed as a simple two-stage procedure: (1) Use large-scale unsupervised text data to pretrain the model; (2) Use target task data to finetune the model.

Recently, multiple works (Radford et al., 2019; Jiang et al., 2020; Roberts et al., 2020; Talmor et al., 2019) have reported that pretrained language models (LM) have implicitly stored large amounts of "world knowledge" in its parameters, and are able to answer common-sense questions. While these studies are encouraging, during the finetuning stage the model is usually trained on a dataset that is very different from the pretraining data, which leads to the potential danger that the model could **forget** precious skills gained during pretraining. This is an important question for open-domain dialogue response generation, which is the focus of our work, because the knowledge acquired during pretraining can greatly help make the dialogue interaction more engaging or informative.

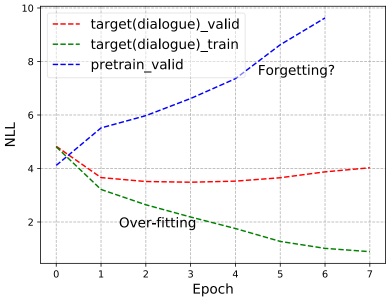


Figure 1: During finetuning, the model's performance on the pretraining data drastically degrades.

In Figure 1, we show that during finetuning, the model's performance on the pretraining data drastically degrades. While this drop is concerning, it does not necessarily mean that the skills from pretrained model are not well transferred to the end dialogue task, because the model should be evaluated in a dialogue setting.

To better answer the question about how finetuning changes the pretrained model's behavior, in this work we conduct a set of behavior analysis from the perspectives of knowledge transfer, context sensitivity, and function space projection. Our major finding is that in the finetuning stage, data separation causes the model to *forget* some of the important language generation skills acquired during pretraining. We also show that the forgetting problem can be alleviated by mixing pretraining and target-task data during finetuning.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1121–1133
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2 Model Formulation

In this work we study the pretrain-finetune framework from the viewpoint of neural language generation (NLG). In particular, we focus on the open-domain dialogue response task, for the following reasons: (1) There is high similarity between the target dialogue response task (conditional NLG) and the pretraining language modeling (LM) objective, so we expect that language generation skills learnt during pretraining can be well transferred to the down-stream target task. (2) The sequence-to-sequence (seq2seq) nature of the model allows us to characterize the model's generation behavior in various ways (e.g., context sensitivity).

End-to-end dialogue response generation (Li et al., 2016) can be formulated as a sequence-to-sequence (seq2seq) task: Given a dialogue context (previous utterances), the model is asked to generate a high-quality response. In this work we adopt the encoder-decoder model architecture (Sutskever et al., 2014; Cho et al., 2014), which is widely used in NLG applications like dialogue response generation (Li et al., 2016), machine translation (Luong et al., 2015), etc. In particular, we use the transformer model (Vaswani et al., 2017), which has currently become the most popular encoder-decoder model architecture (Young et al., 2017). We use the same configuration as (Vaswani et al., 2017), which has 6 encoder/decoder layers, 16 attention heads, with an embedding dimension of 1024 and a feed-forward dimension of 4096.

During standard finetuning, the Adam optimizer (Kingma and Ba, 2014) is used to minimize the negative log-likelihood (NLL) of the reference target sentence $\boldsymbol{y}$ given the input context $\boldsymbol{x}$ in the data distribution (denoted as P_{data}):

$$\begin{aligned}\mathcal{L}_{\text{finetune}}(P_{data};\theta) &= E_{(\boldsymbol{x},\boldsymbol{y})\sim P_{data}}(-\log P_\theta(\boldsymbol{y}|\boldsymbol{x})) \\ &= E_{(\boldsymbol{x},\boldsymbol{y})\sim P_{data}}(-\sum_{t=1}^{m}\log P_\theta(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})),\end{aligned} \tag{1}$$

where $\boldsymbol{y}_{<t}$ refers to $\{y_0, y_1, ..., y_{t-1}\}$, in which y_0 is set to a begin-of-sentence token `<BOS>`, and y_m is a end-of-sentence token `<EOS>`. In the dialogue response setting, the input $\boldsymbol{x}$ is a concatenation of previous utterances. We truncate the length of $\boldsymbol{x}$ to be at most 128 words, which typically includes around 6 previous utterances.

Given a trained seq2seq model, to generate a response for some contextual input, one needs to choose a decoding or sampling method. Recent research (Holtzman et al., 2019; Radford et al., 2019; Fan et al., 2018) has shown that a strategy called top-k sampling, in which the next word is sampled from the top k most probable choices, is a better choice than the traditional beam-search decoding, due to better diversity. Our preliminary experiments (Appendix A) have also verified this claim in the open-domain dialogue response setting. As a result, in this work, unless otherwise mentioned, we use top-k sampling as the default decoding method. In particular, we set k to 30 (we find it to work well in preliminary experiments).

Dialogue
Input: what did you do yesterday ? `<eou>` i watched the avengers movie .
Output: wow ! i am crazy about iron man !
Next-sentence Pretraining
Input: the avengers are super hot currently . `<eou>` the next movie will be on in April .
Output: fans are talking about iron man on the internet .
MASS Pretraining
Input: fans are talking about `<MASK>` `<MASK>` `<MASK>` will do on the internet .
Output: what iron man

Table 1: Illustrations of input-output pairs for typical dialogue response training, next-sentence pretraining, or MASS pretraining.

3 The Pretrain-Finetune Framework

In this section we first review the pretrain-finetune framework for encoder-decoder models. We discuss the language generation skills the model can acquire during pretraining, and more importantly, how we check whether the skills are "forgotten" during finetuning. Finally, as a preliminary attempt to alleviate the forgetting problem, we propose the mix-review finetuning strategy.

3.1 Pretraining

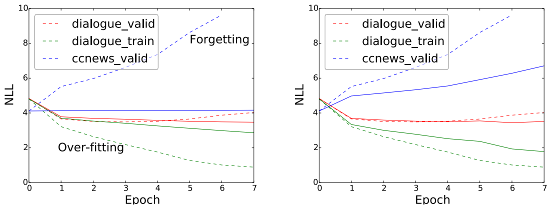
In this work, we consider pretraining the seq2seq model using large-scale unsupervised text data, and afterwards finetuning it using target dialogue data. We compare two representative strategies: next-sentence (NS) pretraining and masked sequence-to-sequence (MASS) pretraining (Song et al., 2019). Next-sentence pretraining is a natural extension of GPT-style LM training (Radford et al., 2019; Kiros et al., 2015) for encoder-decoder models. For every sentence in a given training document, we set the previous sentences as the contextual input, and ask the model to generate the next sentence. We omit the formulation of NS because it is very similar to Equation (1).

Masked sequence-to-sequence pretraining (MASS) can be regarded as an extension of the “BERT” (Devlin et al., 2018) pretraining for encoder-decoder models. For each sentence, a random segment of the sentence is masked, and the model is trained to generate the masked words on the decoder side. We refer readers to (Song et al., 2019) for more details.

In Table 1, we illustrate the similarity between NS pretraining and typical dialogue response training. Compared to NS pretraining, MASS has the disadvantage that it focuses on one single sentence at a time. However, the context of multiple previous sentences are very important for dialogue response generation.

3.2 Analyzing the Forgetting Problem

Although recently a number of pretraining strategies (Peters et al., 2018; Devlin et al., 2018; Song et al., 2019; Yang et al., 2019; Liu et al., 2019) have been proposed for various NLP tasks, the finetuning stage remains simple and straightforward: simply finetune all parameters with a relatively small learning rate.



(a) Mix-review (b) WD(θ_{pre})

Figure 2: Model’s performance on different evaluation sets during the finetuning stage, for the Dailydialog dataset (to be described in Section 4.1). The dotted lines represent the original finetuning process, while the solid lines represent when mix-review or WD(θ_{pre}) is applied.

In Figure 2a, we show (with the dotted lines) the model’s negative log-likelihood (NLL) on different evaluation sets during the finetuning stage. We identify two potential issues during finetuning: (1) **Over-fitting:** The gap between training-set NLL and validation-set NLL increases quickly. (2) **Forgetting:** The performance on the pretraining CCNEWS data (to be described in Section 4.1) drops drastically. Note that the forgetting phenomenon here is not necessarily “catastrophic” as in the sequential learning case (Atkinson et al., 2018; Robins, 1995), because the goal is to achieve the best performance on the target dialogue dataset, and the model does not need to maintain fidelity to the pretraining data. However, **it leads us to question whether the model has lost some important skills learned during pretraining**.

In this work we analyze two important generation capabilities that the model can acquire in the pretraining stage, and will be useful for the target dialogue setting. One is the *acquisition of knowledge*: the large-scale pretraining text data contains a large amount of knowledge, and can be used to make dialogue responses more informative and engaging (e.g., the model can learn about the “Avengers” movie, and use it as a topic). To quantify how knowledgeable the finetuned model is, we prepare a set of knowledge terms such as iphone, pokemon, etc., and the corresponding reference description. We then query the model about these knowledge terms, and compare its output against the reference. We also conduct multi-turn human evaluation in the setting of knowledgeable conversations. More details will be given in Section 5.1.

The other ability is the *utilization of contextual input*: as shown by (Sankar et al., 2019), the current open-domain dialogue models (without pretraining) are insensitive to contextual input, which gives rise to the generic response problem (Li et al., 2016). In our preliminary experiments with NS pretraining, we find that similarly to the GPT model (Radford et al., 2019) the pretrained model has the ability to generate closely related responses given the previous sentences as input. Ideally during finetuning, the model can transfer this skill to the target dialogue task. To quantify the model’s sensitivity to context, following (Sankar et al., 2019), we add noise to the input, and measure the relative drop in perplexity. More details will be given in Section 5.2.

3.3 The Mix-review Finetuning Strategy

As a preliminary attempt to alleviate the forgetting problem, we propose a finetuning strategy named “mix-review (MR)”: For each finetuning epoch, we mix the target dialogue data with a random subset of the pretraining data. This process introduces two hyper-parameters: *mix-ratio*, which controls how much pretraining data is mixed, and *mix-decay*, which decays mix-ratio by each epoch. For example, assume the target dialogue training set has

100k utterances, mix-ratio=4 and mix-decay=0.9, then in the first epoch of mix-review finetuning, 400k pretraining utterances will be mixed in, and for the second epoch the amount will be reduced to 360k utterances, etc.

We formulate the mix-review objective as below:

$$\mathcal{L}_{\text{mix-review}} = \mathcal{L}_{\text{finetune}}(P_{\text{target-data}};\theta) + \\ \text{mix-ratio} \cdot \mathcal{L}_{\text{pretrain}}(P_{\text{pretrain-data}};\theta). \quad (2)$$

Note that the augmented mixing term can be viewed as a regularization term.

We tune the hyper-parameters (mix-ratio and mix-decay) in the grid of $\{1,2,4,8,16\} \times \{1,0.9,0.8,0.7,0.6,0.5\}$ (using the same learning rate and other hyper-parameters with standard finetuning), and report with the best model based on the perplexity (PPL) performance on the validation set of the target task. We find that the performance gain of mix-review is not sensitive to hyper-parameter tuning: a small mix-ratio of 4 typically works well, which means the computational cost of mix-review is comparable to standard finetuning.

In Figure 2a, we show the loss curve for mix-review finetuning with a mix-ratio of 4 and a mix-decay of 0.7. We observe that the performance on the pretraining CCNEWS data is preserved, which strongly supports the motivation of mix-review. Furthermore, we observe a regularization effect from mix-review (narrowing the gap between training and testing performance).

We compare mix-review with the L_2 regularization (weight decay) toward the pretrained parameters θ_{pre} (Kirkpatrick et al., 2016). We denote it as WD(θ_{pre}) and formulate it as follows:

$$\mathcal{L}_{\text{finetune}}(P_{\text{target-data}};\theta) + \lambda \cdot \left\|\theta - \theta_{\text{pre}}\right\|_2^2 . \quad (3)$$

In our experiments, we tune λ in the set $\{10^{-1},10^{-2},10^{-3},10^{-4},10^{-5}\}$ and report with the best model based on PPL on the validation set.

In Figure 2b we show the loss curve for WD(θ_{pre}) with $\lambda = 0.1$. We observe that WD(θ_{pre}) also has a regularization effect, but it is not as strong as mix-review.

Additionally, we tried the following two basic regularization techniques: (1) Increase the rate of dropout; (2) Freeze the bottom layers of the model during finetuning. However, these two techniques show little or no improvement. The reason could be that the transformer is already a well-tuned model (e.g., it features dropout and layer normalization).

4 Datasets and Implementation Details

4.1 Datasets

For pretraining, we use the large-scale CCNEWS data (Bakhtin et al., 2019) which is a de-duplicated subset of the English portion of the CommonCrawl news dataset[1]. The dataset contains news articles published worldwide between September 2016 and February 2019. It has in total around 1 billion sentences or 27 billion words. To be able to complete experiments in a reasonable amount of time, we use the first 10 percent of the CCNEWS data for pretraining, which contains 100 million sentences and 2.7 billion words.

For finetuning, three open-domain conversational dialogue datasets are used: Dailydialog (1.3 million words) (Li et al., 2017), Switchboard (1.2 million words), and Cornell Movie (Danescu-Niculescu-Mizil and Lee, 2011) (4.5 million words). To save space, we defer the details of the data-sets to Appendix B.

To construct the vocabulary, we learn codes of Byte Pair Encoding (BPE) (Sennrich et al., 2016) from the CCNEWS-100m data with 50k merges. This results in a vocabulary of size 62k. We then apply the same BPE codes to all target dialogue datasets.

4.2 Implementation

Our code is based on the Fairseq toolkit (Ott et al., 2019). The Adam optimizer (Kingma and Ba, 2014) is used for all experiments. For pretraining of both MASS and NS, we use a mini-batch size of 2048, with the learning rate (LR) set to 0.0001. Following (Vaswani et al., 2017), the "inverse square root" LR scheduler with a warm-up stage is used. Pretraining is conducted on 32 GPUs and half-precision (float16) speed-up is used. For both MASS and NS, we stop the pretraining after the CCNEWS data is swept 20 times. For all our experiments, a dropout rate of 0.1 is applied to the transformer model. We follow Song et al. (2019) for the recommended hyper-parameter setting of MASS (e.g., how to select the mask span).

Finetuning is done on 2 GPUs without float16 speed-up. The learning rate is halved when the PPL on the validation set does not improve. In almost all finetuning experiments over-fitting is observed, and we do an early-stop when performance on the validation set starts to deteriorate. We tune the

---

[1] `http://commoncrawl.org/2016/10/news-dataset-available`

Training	**Test-PPL**		
	DD	**SB**	**CM**
Baseline(from scratch)	24.83	51.14	49.48
MASS+finetune	12.78	28.41	30.25
NS+finetune	11.54	26.37	28.06
NS+WD(θ_{pre})	11.19	26.25	27.80
NS+mix-review	**11.07**	**25.92**	**27.54**

Table 2: Perplexity on test set for different training process on the three dialogue datasets. The datasets are Dailydialogue (DD), Switchboard (SB), and Cornell Movie (CM).

learning rate from $\{10^{-3},10^{-4},10^{-5}\}$, and report the best model based on validation set perplexity.

5 Experiment Results

In this section, we conduct a set of detailed behavior analysis, characterising how different training strategies change the model's behavior. In particular, we aim to answer the crucial question about whether the model forgets precious language generation skills during standard finetuning, and whether mix-review helps the model remember the skills.

We first present perplexity results for different finetuning methods in Table 2. We observe the big improvement in perplexity (larger than 40%) for the pretrained models comparing to the baseline models trained from scratch. Comparing to MASS, the NS pretraining has more than 7% relative improvement. This confirms our earlier discussion that the model pretrained by NS better utilizes contextual input (which is further verified in Section 5.2). Based on this observation, we focus our analysis below on the NS pretraining.

Comparing to standard finetuning, mix-review further gives solid improvement. The gain is due to its strong regularization effect (which we study in the next three sections). However, the performance gap between mix-review and WD(θ_{pre}) is not significant. We believe the reason is that the benefit (e.g., knowledge transfer) from alleviate the forgetting problem is not be well demonstrated in single-turn response evaluation, because the context is limited to the narrow scope of the specific datasets. We address this concern with multi-turn human evaluation in the next section.

5.1 Behavior Analysis: Knowledge Transfer

As argued in Section 3.1, ideally the model can acquire common-sense (or world) knowledge from the large-scale pretraining data, which will be useful for the downstream open-domain dialogue task. In this section, we design a process to quantify how much knowledge the model has, and use it to monitor how the pretrain-finetune framework changes the model's behavior.

Since the pretraining CCNEWS data is in the public news domain, we expect the model to have knowledge about "big news". So, we utilize the Google trend data of the year 2016[2], which contains 365 trending terms (e.g., iPhone 7, Deadpool), and its corresponding description.

To query whether the model has knowledge of a certain term, we design three news-style and three dialogue-style "trigger templates" to trigger the model to generate responses related to the knowledge term. We collect 10 samples for each trigger, then we compute BLEU score of generated samples against the reference descriptions. We show some examples of trigger inputs in Table 3.

The BLEU scores are shown in Table 4. Note that for the pretrained model we feed news triggers, while for the other dialogue models dialogue triggers are used. We observe that although the finetuned model is more knowledgeable than the baseline model, its score is much lower than the pretrained model. This demonstrates the forgetting problem for the standard finetuning.

On the other hand, we find that mix-review and WD(θ_{pre}) can effectively retain the knowledge acquired during pretraining, giving a much higher BLEU score than the standard finetuned model. Mix-review shows higher BLEU scores than WD(θ_{pre}), demonstrating its superiority in facilitating knowledge retention. We showcase samples from different models in Table 3. To save space, we manually select and show the most related sample out of the 30 samples for each knowledge term. The observations agree with the quantitative results: The standard finetuning loses the detailed information about the knowledge term, and mix-review helps the model retain it. More importantly, the model is able to express the knowledge in a dialogue context.

To further investigate our model's ability to conduct knowledgeable dialogues with users, we use the ParlAI[3] platform to conduct multi-turn dialogue evaluation. For each session, the user will be assigned a random knowledge term from Google Trend, and have a 8-turn dialogue with the model under that topic. Ratings from around 600 dia-

---

[2] `https://www.google.com/intl/en-US/trends/2016records/`

[3] https://parl.ai/

News-style Triggers	**Dialogue-style Triggers**
now, some opinions about X . let me tell you about X . here's some news about X .	what you do think about X ? please tell me about X . do you have news about X ?
Reference Description: Pokemon first took the world by storm in the mid-90s, doing so once again this year with the release of Pokemon Go. **NS Pretrained:** the game , titled pokemon go : pocket camp , can be played in person ... **Standard Finetuned:** it 's a new game that can be played with kids . **WD(θ_{pre}):** pokemon go , it 's a type of game that only exists in the us . **Mix-review:** pokemon go is a popular mobile game , where you 're expected to catch pokemon .	
Reference Description: Deadpool: The wisecracking antihero, played by Ryan Reynolds in a movie of the same name, became the highest-grossing R-rated film of all time. **NS Pretrained:** ryan reynolds teased his upcoming movie as the character of deadpool . **Standard Finetuned:** it 's a popular movie . **WD(θ_{pre}):** yes , i really like him . he is a very funny character . **Mix-review:** ryan reynolds .	

Table 3: Templates for news or dialogue-style triggers. "X" is to be replaced by specific knowledge terms. They are followed by reference description and model samples for "pokemon" and "deadpool". Note that the pretrained model's sample is from news-style triggers, and the other samples are from dialogue-style triggers.

Training	**Dailydialog**	**Switchboard**	**Cornell**
Pretrained	**BLEU-2 0.347 / BLEU-3 0.153**		
Baseline	0.124/0.007	0.032/0.003	0.081/0.003
NS+finetune	0.162/0.047	0.187/0.052	0.207/0.071
NS+WD	0.226/0.080	0.203/0.070	0.285/0.114
NS+MR	**0.261/0.108**	**0.223/0.079**	**0.396/0.190**

Table 4: Average BLEU-2/BLEU-3 scores for the model's samples w.r.t. the reference description. We highlight the pretrained model's performance for news triggers and the performance of the best model finetuned with dialogue data for dialogue triggers.

Training	**Knowledge**	**Consistency**	**Engaging**
finetune	2.82 ± .06	4.28 ± .06	3.84 ± .05
WD(θ_{pre})	3.18 ± .06	4.60 ± .06	4.18 ± .05
Mix-review	**3.40 ± .05**	**4.75 ± .06**	**4.27 ± .05**

Table 5: AMT rating scores (mean and standard deviation) for multi-turn knowledgeable dialogue evaluation.

Training	**Dailydialog**	**Switchboard**
NS Pretrained	**word shuffle +110% / drop +105%**	
Baseline	+12%/+28%	+4%/+5%
MASS+FT	+24%/+48%	+15%/+19%
NS+FT	+41%/+64%	+17%/+21%
NS+WD(θ_{pre})	+26%/+46%	+19%/+25%
NS+MR	**+60%/+108%**	**+19%/+30%**

Table 6: The model's relative PPL drop when word-shuffle/drop is applied to input. "FT" refers to "finetune". To save space, the results on the Cornell Movie dataset is deferred to Appendix D, Table 11.

logues are collected for each model, and are reported in Table 5. In this evaluation we use the models finetuned on the Dailydialog data, because the nature of that dataset is closet to online chit-chat. It is observed that the model trained by mix-review significantly outperforms WD(θ_{pre}) on knowledge, consistency and engagingness, which agrees well with the results in Table 4. Some dialogue examples are included in Table 7.

5.2 Behavior Analysis: Context Sensitivity

The sensitivity to context is an important property for NLG models. However, as shown by (Sankar et al., 2019), dialogue models trained from scratch typically are not sensitive to artificial distortion in the context input, showing the models have poor utilization of dialogue context. In this section, we repeat their experiments with pretrained or finetuned models.

Following (Sankar et al., 2019), we use two methods to distort the context input:

- **word-drop:** We randomly drop 30% of the words in the context input.
- **word-shuffle:** We randomly shuffle the words in the context input.

We use the relative drop in test-set perplexity to quantify the sensitivity. The results are presented in Table 6, where the result of the pretrained model is also included. First, we observe the baseline model trained from scratch is relatively insensitive to context, which agrees well with Sankar et al. (2019). The model with the standard pretrain-finetune process is much more sensitive, showing that pretraining effectively changes the model's behavior. Comparing to MASS, the NS pretrained model has better utilization of context, which explains its superior performance in PPL.

Somewhat surprisingly, the finetuned dialogue models are much less sensitive to context input than the pretrained model without finetuning. This again

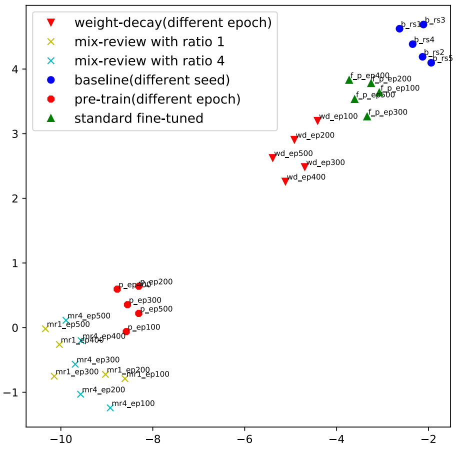


Figure 3: UMAP projection of checkpoints from different training processes.

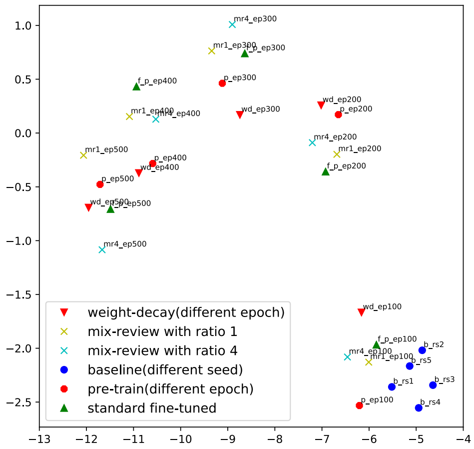


Figure 4: Parameter-space UMAP projection of checkpoints from different training processes.

verifies our worry in Section 3.3 that the model is forgetting some important generation skill during standard finetuning. Further, we find that the mix-review finetuning strategy can effectively alleviate this problem: Its sensitivity is much greater than that of standard finetuning, and is close to the pre-trained model.

5.3 Behavior Analysis: Function Space Projection

It is interesting to study models' behavior via function-space 2D projection (Erhan et al., 2010). We collect the model's output distribution on 10k words for the CCNEWS validation set and the Dailydialog validation set[4]. And feed them as input to UMAP (McInnes et al., 2018). We use the default hyper-parameter setting of the python implementation of UMAP. The result is shown in Figure 3. Note that during pretraining of the CCNEWS data, 20 epochs are one entire data pass. We finetune from epoch 100, 200, 300, 400, 500 of the pretraining checkpoints.

We observe that the standard finetuned models are not close to the cluster of the pretrained models, which suggests the models' generative behavior is substantially different from the pretrained ones. Mix-review regularizes the finetuning process by keeping the model's generation behavior close to the pretrained model. These observations agree with our results in Section 5.1 and 5.2. Figure 3 also suggests potential limitations of mix-review and WD(θ_{pre}): mix-review could be too "aggressive" and does not put enough attention on the target task. On the other hand, WD(θ_{pre}) is not strong enough in regularizing the model's generative behavior.

In Figure 4 we show the parameter-space UMAP projection for the same set of models. In this case, the input to UMAP is the concatenation of flattened weight matrices of the transformer model. A key observation is that the finetuned models are typically very close to the starting point (pretrained models). However, as shown in Figure 3, their behavior is very different. This suggests that a parameter-space regularization such as WD(θ_{pre}) could be not very effective for regularizing the model's behavior.

6 Implications and Discussion

The sensitivity to dialogue context and the ability to transfer knowledge from pretraining opens the possibility of a **data-driven** knowledgable chatbot. In Table 7, we show multi-turn and single-turn interaction examples with the model trained by mix-review. For demonstration purpose, we manually select the most interesting response out of 10 samples from the model for the single-turn examples. We observe that the model is able to return interesting responses with the knowledge it acquired from pretraining. Interestingly, it has developed its own "opinions" and can give advice

---

[4] It's a concatenation of two long vectors.

User: have you been to china ?	**U:** who is mark zukerberg ?	**U:** tell me about tv show Robot Wars .
Model: yes , i have .	**M:** he is the ceo of facebook .	**M:** it 's about robots fighting in a war .
User: which city did you go ?	**U:** what is that company about ?	**U:** cool ! What season is it on ?
Model: shanghai	**M:** it is a social media company .	**M:** it 's on next month .
User: how do you like it there ?	**U:** do you like him ?	**U:** how many years has it been on for ?
Model: i like it very much	**M:** i think he is a good leader .	**M:** it started in 2003 .
User: i think the weather is not so good .	**U:** my friend works at facebook .	**U:** what channel does it come on ?
Model: but it is nice in spring and fall .	**M:** what does your friend do there ?	**M:** the science channel.
User: when is the independence day ? **Best-of-10:** it 's july fourth.		
User: is coffee good for health ? **Best-of-10:** it may help reduce your risk of certain diseases.		
User: should i do a phd or go to work ? **Best-of-10:** a phd is not enough if you want to learn something new .		
User: oh shut up you . **Worst-of-10:** shut up !		
User: you 're the problem . **Worst-of-10:** i hate you .		
User: i will kill . **Worst-of-10:** i will kill you .		

Table 7: Multi-turn and single-turn examples of the model trained by mix-review on the Dailydialogue dataset. The single-turn examples involve light cherry-picking.

to the user.

Next, we discuss the malicious response problem for open-domain dialogue models. As shown by (He and Glass, 2019a), it is relatively difficult to trigger the dialogue models trained from scratch to output malicious responses. However, as shown in Table 7, the pretrained models are easily triggered to respond in a malicious way when "provoked". This is because compared to the baseline models, the pretrained models are more sensitive to the contextual input, making them easier to manipulate. This makes the malicious response problem a more relevant issue to solve (He and Glass, 2019b).

Finally, we discuss some limitations of our work. First, the mix-review strategy we proposed is a simple and preliminary attempt to alleviate the forgetting, and its performance is far from perfect. As shown in Appendix C, in a lot of cases, the generation from mix-review is still boring or non-informative. Next, the three datasets considered in this work are open-domain dialogue datasets, and they are not knowledge-intensive. It would be interesting, as future work, to check the forgetting problem for knowledge-grounded datasets such as Topical-chat (Gopalakrishnan et al., 2019).

7 Related Works

Behavior of pretrained NLG Models Recently, multiple works (Radford et al., 2019; Jiang et al., 2020; Roberts et al., 2020; Talmor et al., 2019; Trinh and Le, 2019) have reported that pre-trained language models (LM) have implicitly stored large amounts of "world knowledge" in its parameters, and are able to answer common-sense questions. However, whether the world knowledge is well preserved after finetuning on target task dataset is not discussed.

On the other hand, knowledge-grounded NLG model (Liu et al., 2018; Guu et al., 2020; Zhou et al., 2018) has been an important and exciting research topic. These studies usually involve additional retrieval modules or external knowledge bases to provide the model with relevant information. In contrast to these works, we study whether the model can conduct knowledgeable dialogues by itself.

Forgetting As discussed in Section 3.2, in contrast to the "catastrophic forgetting" problem in sequential learning (Atkinson et al., 2018; Robins, 1995), the performance drop on pretraining data is not necessarily bad for the NLP pretrain-finetune framework, and its implications have not been properly studied. In our analysis, we confirm the "forgetting" of important language generation skills during standard finetuning. The proposed mix-review strategy is similar to the *pseudo-rehearsal* algorithm in sequential learning (Robins, 1995), with the difference being that we assume we still have access to the pretraining data.

8 Conclusion

In this work, we attempt to answer to question of whether during finetuning, the model has forgotten some of the useful NLG skills acquired during large-scale pretraining. Through a set of detailed behavior analysis, we find the answer is, to some extent, yes. For example, the finetuned model fails to give detailed information about some knowledge terms, while the pretrained model can. As a preliminary attempt to alleviate the forgetting problem, we propose the mix-review finetuning method, and find it to be effective.

Our analysis shows that under the surface of the performance boost for standard metrics, large-scale pretraining changes the model's generative behavior in various profound ways. More importantly, the behavior change is influenced by the nature of data itself. For example, we demonstrate that we can discuss news with the dialogue model finetuned by mix-review, even when the target dataset is not about news (Dailydialog). We believe that this opens the possibility of a completely data-driven way to customize a language generator.

References

Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. 2018. Pseudo-recursal: Solving the catastrophic forgetting problem in deep neural networks. *CoRR*, abs/1802.03875.

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *CoRR*, abs/1906.03351.

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

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, 11:625–660.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. 2019. Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In *Proc. Interspeech 2019*, pages 1891–1895.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training.

Tianxing He and James Glass. 2019a. Detecting egregious responses in neural sequence-to-sequence models. In *International Conference on Learning Representations*.

Tianxing He and James R. Glass. 2019b. Negative training for neural dialogue response generation. *CoRR*, abs/1903.02134.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know?

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114 13:3521–3526.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 110–119.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Shuman Liu, Hongshen Chen, Zhaochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1498, Melbourne, Australia. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. 2018. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model?

Anthony Robins. 1995. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7:123–146.

Chinnadhurai Sankar, Sandeep Subramanian, Christopher J. Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. *CoRR*, abs/1906.01603.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. olmpics – on what language model pre-training captures.

Trieu H. Trinh and Quoc V. Le. 2019. Do language models have common sense?

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2017. Recent trends in deep learning based natural language processing. *CoRR*, abs/1708.02709.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4623–4629. International Joint Conferences on Artificial Intelligence Organization.

A Beam-search vs. Top-k Sampling

To compare beam-search with top-k sampling (we set k to 30), we compute diversity metrics for samples from models trained by different procedures (from scratch or pretrained). In particular, we compute bi-gram and tri-gram entropy, and the ratio of the most frequent response and second most frequent response (denoted as max-ratio) (He and Glass, 2019b). The results are shown in Table 8.

We observe that the responses given by top-k sampling are much more diverse than beam-search. Beam-search suffers much from the "generic response" problem (Li et al., 2016), for example, 34% of the responses are "`um - hum`" for Switchboard. Further, in our multi-turn dialogue experiments, beam-search is likely to give repetitive responses. Finally, by manual inspection, we find the sample quality of top-k sampling is not compromised. Due to these observations, we adopt top-k sampling for our models.

B Details on Datasets

Dailydialogue (Li et al., 2017) is a high-quality multi-turn dialog dataset. The language is human-written and less noisy. The dialogues in the dataset reflect our everyday communication and cover various topics about our daily life. The training split has around 11k dialogues (1.3 million words), and both the validation and test splits have 1k dialogues (0.1 million words).

The Switchboard Dialogue Act Corpus[5] is a version of the Switchboard Telephone Speech Corpus, which is a collection of two-sided telephone conversations, annotated with utterance-level dialogue acts. In this work we only use the conversation text part of the data, and select 1.1k dialogues for training (181k sentences / 1.2 million words), 50 dialogues for validation and 50 dialogues for testing.

The Cornell Movie Dialogue Corpus[6] (Danescu-Niculescu-Mizil and Lee, 2011) is a collection of movie scripts. In the processing of the data, we simply regard the whole scripts from a movie as a long dialogue. The training split contains 9k dialogues (4.5 million words), and both the validation and test splits have 180 dialogues (85k words).

C Model Samples

In Table 9, we show more samples from different training procedure, for the three dialogue datasets.

D Supplementary Experiment Results

	Cornell	
Model	**Dialogue Triggers**	**News Triggers**
NS Pretrained	0.245/0.089	**0.347/0.153**
Baseline	0.081/0.003	0.088/0.003
NS+finetune	0.207/0.071	0.207/0.063
NS+WD(θ_{pre})	0.285/0.114	0.202/0.072
NS+Mix-review	**0.396/0.190**	0.212/0.065

Table 10: Average BLEU-2/BLEU-3 scores for the model's samples w.r.t. the reference description. We highlight the pretrained model's performance for news triggers and the performance of the best model fine-tuned with dialogue data for dialogue triggers.

In this section we supplement results that are deferred in the main body due to space limit.

In Table 10 we show the knowledge transfer results for the Cornell Movie dataset.

In Table 11 we show context sensitivity results for the Cornell Movie dataset.

---

[5] http://compprag.christopherpotts.net/swda.html

[6] https://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html

	Beam Search		**Top-30 Sampling**	
Dataset	**Entropy**	**Max-ratio**	**Entropy**	**Max-ratio**
Dailydialogue	7.44 8.49	1.7% 1.3%	9.04 10.81	0.6% 0.4%
Switchboard	4.96 5.54	34.9% 27.8%	8.47 10.45	8.4% 7.9%
Cornell	6.10 6.56	10.2% 9.9%	8.76 10.54	1.4% 1.1%

Table 8: Average of diversity metrics for models on the three dialogue datasets.

Dailydialogue

Input: why do you think you are qualified for this position ?
Baseline: i can tell you more about your company because your company has a working experience with a company and company company .
NS: i have been working with microsoft for four years . i can make good use of microsoft ' s word pad and excel excel spreadsheet .
MASS: i think i have a strong sense of responsibility and i have had a high iq score in college . i am applying for some courses of study for some courses if i have a preference .
Mix-review: my background is in computer engineering and i have a strong commercial acumen in business - related application . i am in charge of all the things i like to do here .
WD(θ_{pre}): my background , education , experience as an english professor , and the work ethic are strong .
Input: according to your advertisement , you want an experienced software engineer . i think my background meets the requirement of this position .
Baseline: yes . and you also need to know how to enjoy the work . and , here is a position to develop the company position under the company .
NS: do you know your qualifications ?
MASS: that ' s a good answer . i can tell you if we have made a decision this afternoon .
Mix-review: i see . do you have some experience as an executive engineer ?
WD(θ_{pre}): in your first year of operating your company , you had an impressive track record of product design , and achieved a high level of technical proficiency .

Switchboard

Input: that 's going to be a really viable program if it continues to be funded at current levels
Baseline: oh really
NS: um it 's going to be a lot of fun it certainly is
MASS: sure sure sure
Mix-review: well that that 's what i 'm saying too that 's that 's not gonna be an issue
WD(θ_{pre}): yeah oh sure sure
Input: uh - huh
Baseline: oh okay
NS: and uh i- i think there are some areas that if the government decides to close down that could be a real problem
MASS: and i really feel strongly that we don 't need the space station because we think there 's really no space
Mix-review: but uh that
WD(θ_{pre}): i mean it 's pretty i think it 's it 's pretty interesting to me that it 's uh

Cornell Movie

Input: i expect you shortly .
Baseline: how so ?
NS: i 'll make a reservation .
MASS: i 'll come with you , doctor .
Mix-review: no , i 'm fine
WD(θ_{pre}): okay .
Input: what shall we do ?
Baseline: we all don 't know where the hell he is now , but if he talks to me first , i 'll be back .
NS: nothing .
MASS: we send a cab ! i said i expect you to be fast .
Mix-review: david 's not very lucid , but he 's always lucid .
WD(θ_{pre}): stop him .
Input: tea would be nice .
Baseline: i have no desire to go to bed .
NS: but you can 't come here and have tea too ?
MASS: i 've been expecting you , doctor .
Mix-review: do you want me to drink some ?
WD(θ_{pre}): what is it ?

Table 9: Samples of different models on the Dailydialog/Switchboard/Cornell-Movie test-set. The samples are consecutive (input of the next sample is the reference response for the previous one).

Model(Dataset)	PPL(**normal**)	PPL(**word-shuffle**)	PPL(**word-drop**)
NS pretrained(CCNEWS)	17.33	36.56(**+110.96%**)	35.56(**+105.19%**)
Baseline(Cornell)	49.48	50.22(+1.4%)	50.85(+2.7%)
MASS+finetune(Cornell)	30.25	36.50(+20.6%)	36.36(+20.1%)
NS+finetune(Cornell)	28.06	36.88(+31.4%)	34.47(+22.8%)
NS+WD(θ_{pre})(Cornell)	27.80	37.46(**+34.7%**)	35.10(+26.2%)
NS+Mix-review(Cornell)	27.54	36.94(+34.1%)	37.72(**+36.9%**)

Table 11: The model's PPL performance when word-shuffle or word-drop is applied to the context input. On the left we describe what training process is used and on which test set is PPL evaluated. Note that MASS/NS refers to MASS/NS pretraining with standard finetuning.

Leveraging End-to-End ASR for Endangered Language Documentation: An Empirical Study on Yoloxóchitl Mixtec

Jiatong Shi[1] **Jonathan D. Amith**[2] **Rey Castillo García**[3]
Esteban Guadalupe Sierra **Kevin Duh**[1] **Shinji Watanabe**[1]
[1]The Johns Hopkins University, Baltimore, Maryland, United States
[2]Department of Anthropology, Gettysburg College
[3]Secretaría de Educación Pública, Estado de Guerrero, Mexico
{jiatong_shi@, kevinduh@cs.}jhu.edu
{jonamith, reyyoloxochitl, estebanyoloxochitl}@gmail.com
shinjiw@ieee.org

Abstract

"Transcription bottlenecks", created by a shortage of effective human transcribers are one of the main challenges to endangered language (EL) documentation. Automatic speech recognition (ASR) has been suggested as a tool to overcome such bottlenecks. Following this suggestion, we investigated the effectiveness for EL documentation of end-to-end ASR, which unlike Hidden Markov Model ASR systems, eschews linguistic resources but is instead more dependent on large-data settings. We open source a Yoloxóchitl Mixtec EL corpus. First, we review our method in building an end-to-end ASR system in a way that would be reproducible by the ASR community. We then propose a novice transcription correction task and demonstrate how ASR systems and novice transcribers can work together to improve EL documentation. We believe this combinatory methodology would mitigate the transcription bottleneck and transcriber shortage that hinders EL documentation.

1 Introduction

Grenoble et al. (2011) warned that half of the world's 7,000 languages would disappear by the end of the 21st century. Consequently, a concern with endangered language documentation has emerged from the convergence of interests of two major groups: (1) native speakers who wish to document their language and cultural knowledge for future generations; (2) linguists who wish to document endangered languages to explore linguistic structures that may soon disappear. Endangered language (EL) documentation aims to mitigate these concerns by developing and archiving corpora, lexicons, and grammars (Lehmann, 1999). There are two major challenges:

(a) Transcription Bottleneck: The creation of EL resources through documentation is extremely challenging, primarily because the traditional method to preserve primary data is not simply with audio recordings but also through time-coded transcriptions. In a best-case scenario, texts are presented in interlinear format with aligned parses and glosses along with a free translation (Anastasopoulos and Chiang, 2017). But interlinear transcriptions are difficult to produce in meaningful quantities: (1) ELs often lack a standardized orthography (if written at all); (2) invariably, few speakers can accurately transcribe recordings. Even a highly skilled native speaker or linguist will require a minimum of 30 to 50 hours to simply transcribe one hour of recording (Michaud et al., 2014; Zahrer et al., 2020). Additional time is needed for parse, gloss, and translation. This creates what has been called a "transcription bottleneck", a situation in which the expert transcribers cannot keep up with the amount of recorded material for documentation.

(b) Transcriber Shortage: It is generally understood that any viable solution to the transcription bottleneck must involve native speaker transcribers. Yet usually few, if any, native speakers have the skills (or time) to transcribe their language. Training new transcribers is one solution, but it is time-consuming, especially with languages that present complicated phonology and morphology. The situation is distinct for major languages, for which transcription can be crowd-sourced to speakers with little need for specialized training (Das and Hasegawa-Johnson, 2016). In Yoloxóchitl Mixtec (YM; Glottocode=yolo1241, ISO 639-3=xty), the focus of this study, training is time-consuming: after one-year part-time transcription training, a proficient native speaker, Esteban Guadalupe Sierra, still has problems with certain phones, particularly tones and glottal stops. Documentation requires accurate transcriptions, a goal yet beyond even the capability of an enthusiastic speaker with many

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1134–1145
April 19 - 23, 2021.

months of training.

As noted, ASR has been proposed to mitigate the Transcription Bottleneck and create increasingly extensive EL corpora. Previous studies first investigated HMM-based ASR for EL documentation (Ćavar et al., 2016; Mitra et al., 2016; Adams et al., 2018; Jimerson et al., 2018; Jimerson and Prud'hommeaux, 2018; Michaud et al., 2018; Cruz and Waring, 2019; Thai et al., 2020; Zahrer et al., 2020; Gupta and Boulianne, 2020a). Along with HMM-based ASR, natural language processing and semi-supervised learning have been suggested as a way to produce morphological and syntactic analyses. As HMM-based systems have become more precise, they have been increasingly promoted as a mechanism to bypass the transcription bottleneck. However, ASR's context for ELs is quite distinct from that of major languages. Endangered languages seldom have sufficient extant language lexicons to train an HMM system and invariably suffer from a dearth of skilled transcribers to create these necessary resources (Gupta and Boulianne, 2020b).

As we have confirmed with this present study, end-to-end ASR systems have shown comparable or better results over conventional HMM-based methods (Graves and Jaitly, 2014; Chiu et al., 2018; Pham et al., 2019; Karita et al., 2019a). As end-to-end systems directly predict textual units from acoustic information, they save much effort on lexicon construction. Nevertheless, end-to-end ASR systems still suffer from the limitation of training data. Attempts with resource-scarce languages have relatively high character (CER) or word (WER) error rates (Thai et al., 2020; Matsuura et al., 2020; Hjortnaes et al., 2020). It has nevertheless become possible to utilize ASR with ELs to reduce significantly, but not eliminate, the need for human input and annotation to create acceptable ("archival quality") transcriptions.

This Work: This work represents end-to-end ASR efforts on Yoloxóchitl Mixtec (YM), an endangered language from western Mexico. The YMC[1] corpus comprises two sub-corpora. The first ("YMC-EXP", expert transcribed, corpus) includes 100 hours of transcribed speech that have been carefully checked for accuracy. We built a recipe of the ESPNet (Watanabe et al., 2018) that shows the whole process of constructing an end-to-end ASR system using the YMC-EXP corpus.[2] The second corpus, ("YMC-NT", native trainee, corpus) includes 8+ hours of additional recordings not included in the YMC-EXP corpus. This second corpus contains novice transcriptions with subsequent expert corrections that has allowed us to evaluate the skill level of the novice. Both the YMC-EXP and YMC-NT corpora are publicly available at OpenSLR under a CC BY-SA-NC 3.0 License.[3]

The contributions of our research are:

- A new Yoloxóchitl Mixtec corpus to support ASR efforts in EL documentation.
- A reproducible workflow to build an end-to-end ASR system for EL documentation.
- A comparative study between HMM-based ASR and end-to-end ASR, demonstrating the feasibility of the latter. To test the framework's generalizability, we also experiment with another EL: Highland Puebla Nahuat (Glottocode=high1278; ISO 639-3=azz).
- An in-depth analysis of errors in novice transcription and ASR. Considering the discrepancies in error types, we propose Novice Transcription Correction (NTC) as a task for the EL documentation community. A rule-based method and a voting-based method are proposed.[4] In clean speech, the best system reduces relative word error rate in the novice transcription by 38.9% .

2 Corpus Description

In this section, we first introduce the linguistic specifics for YM and YMC. Then we discuss the recording settings. Since YM is a spoken language without a standardized textual format, we next explain the transcription style designed for this language. Finally, we offer the corpus partition and some statistics regarding corpora size.

2.1 Linguistic Specifics for Yoloxóchitl Mixtec

Yoloxóchitl Mixtec is an endangered, relatively low-resource Mixtecan language. It is mainly spoken in the municipality of San Luis Acatlán, state

---

[1] Specifically, we used material from the community of Yoloxóchitl (YMC), one of four in which YM is spoken.

[2] `https://github.com/espnet/espnet/tree/master/egs/yoloxochitl_mixtec/asr1`

[3] `http://www.openslr.org/89/`

[4] A system combination method, Recognizer Output Voting Error Reduction (Fiscus, 1997))

of Guerrero, Mexico. It is one of some 50 languages in the Mixtec language family, which is part of a larger unit, Otomanguean, that Suárez (1983) considers “a ‘hyper-family’ or ‘stock’.” Mixtec languages (spoken in Oaxaca, Guerrero, and Puebla) are highly varied, resulting from approximately 2,000 years of diversification.

YM is spoken in four communities: Yoloxóchitl, Cuanacaxtitlan, Arroyo Cumiapa, and Buena Vista. Mutual intelligibility among the four YM communities is high despite significant differences in phonology, morphology, and syntax. All villages have a simple segmental inventory but significant though still undocumented variation in tonal phonology. YMC (refering only to the Mixtec of the community of Yoloxóchitl [16.81602, -98.68597]) manifests 28 distinct tonal patterns on 1,451 identified bimoraic lexical stems. The tonal patterns carry a significant functional load in regards to the lexicon and inflection. For example, 24 distinct tonal patterns on the bimoraic segmental sequence [nama] yield 30 words (including six homophones). This ample tonal inventory presents challenges to both a native speaker learning to write and an ASR system learning to recognize. Notably, it also introduces difficulties in constructing a language lexicon for training HMM-based systems.

2.2 Recording Settings

There are two corpora used in this study. The first (YMC-EXP) was used for ASR training. The second (YMC-NT) was used to train the novice speaker (e.g., set up a curriculum for him to learn how to transcribe) and for Novice Transcription Correction. The YMC-EXP corpus comprises expert transcriptions used as the gold-standard reference for ASR development. The YMC-NT corpus has paired novice-expert transcription as it was used to train and evaluate the novice writer.

The corpus used for ASR development comprises mostly conversational speech in two-channel recordings (split for training). Each conversation is with two speakers and each of the two speakers was fitted with a separate head-worn mic (usually a Shure SM10a). Over two dozen speakers (mostly male) contributed to the corpus. The topics and their distribution were varied (plants, animals, hunting/fishing, food preparation, ritual speech). The YMC-NT corpus comprises single-channel field recordings made with a Zoom H4n at the moment plants were collected during ethnobotanical research. Speakers were interviewed one after another; there is no overlap. However, the recordings often registered background sounds (crickets, birds) that we expected would negatively impact ASR accuracy more than seems to have occurred. The topic was always a discussion of plant knowledge (a theme of only 9% of the YMC-EXP corpus). Expectedly, there were many out-of-vocabulary (OOV) words (e.g., plant names not elsewhere recorded) in this YMC-NT corpus.[5]

2.3 Corpus Transcription

(a) Transcription Level: The YMC-EXP corpus presently has two levels of transcription: (1) a practical orthography that represents underlying forms; (2) surface forms. The underlying form marks prefixes (separated from the stem by a hyphen), enclitics (separated by an = sign), and tone elision (with the elided tones in parentheses). All these “breaks” and phonological processes disappear in the surface form. For example, the underlying $\mathrm{be}'^{3}e^{3}{=}an^{4}$ (house=3sgFem; 'her house') surfaces as $\mathrm{be}'^{3}\tilde{a}^{4}$. And $\mathrm{be}'^{3}e^{(3)}{=}^{2}$ ('my house') surfaces as $\mathrm{be}'^{3}e^{2}$. Another example is the completive prefix ni^{1}-, which is separated from the stem as in ni^{1}-$xi^{3}xi^{(3)}{=}^{2}$ (completive-eat-1sgS; 'I ate'). The surface form would be written $\widetilde{ni}^{1}xi^{3}xi^{2}$. Again, processes such as nasalization, vowel harmony, palatalization, and labialization are not represented in the practical (underlying) orthography but are generated in the surface forms. The only phonological process encoded in the underlying orthography is tone elision, for which parentheses are used.

The practical, underlying orthography mentioned above was chosen as the default system for ASR training for three reasons: (1) it is easier than a surface representation for native speakers to write; (2) it represents morphological boundaries and thus serves to teach native speakers the morphology of their language; and (3) for a researcher interested in generating concordances for a corpus-based lexicographic project it is much easier to discover the root for 'house' in $\mathrm{be}'^{3}e^{3}{=}an^{4}$ and $\mathrm{be}'^{3}e^{(3)}{=}^{2}$ than in the surface forms $\mathrm{be}'^{3}\tilde{a}^{4}$ and $\mathrm{be}'^{3}e^{2}$.

(b) “Code-Switching” in YMC: Endangered, colonialized Indigenous languages often manifest extensive lexical input from a dominant Western language, and speakers often talk with “code-switching” (for lack of a better term). Yoloxóchitl

---

[5] After separating enclitics and prefixes as separate tokens, the OOV rate in YMC-NT is 4.84%.

Corpus	Subset	UttNum	Dur (h)
EXP	Train	52763	92.46
	Validation	2470	4.01
	Test	1577	2.52
EXP(-CS)	Train	35144	58.60
	Validation	1301	2.16
	Test	2603	4.35
NT	Clean-Dev	2523	3.45
	Clean-Test	2346	3.31
	Noise-Test	1335	1.60

Table 1: YMC Corpus Partition for EXP (corpus with expert transcription), EXP(-CS) (subset of EXP without "code-switching"), NT (corpus with paired novice and expert transcription)

Mixtec is no exception. Amith considered how to write such forms best and decided that Spanish-origin words would be written in Spanish and without tone when their phonology and meaning are close to that of Spanish. So Spanish *docena* appears over a dozen times in the corpus and is written *tucena*; it always has the meaning of 'dozen'. All month and day names are also written without tones. Note, however, that Spanish *camposanto* ('cemetery') is also found in the corpus and pronounced as $pa^3san^4tu^2$. The decision was made to write this with tone markings as it is significantly different in pronunciation from the Spanish origin word. In effect, words like $pa^3san^4tu^2$ are considered loans into YM and are treated orthographically as Mixtec. Words such as *tucena* are considered "code-switching" and written without tones.

(c) Transcription Process: The initial time-aligned transcriptions were made in Transcriber (Barras et al., 1998). However, given that Transcriber cannot handle multiple tiers (e.g., transcription and translation, or underlying and surface orthographies), the Transcriber transcriptions were then imported into ELAN (Wittenburg et al., 2006) for further processing (e.g., correction, surface-form generation, translation).

2.4 Corpus Size and Partition

Though endangered, YMC does not suffer from the same level of resource limitations that affect most ASR work with ELs (Ćavar et al., 2016; Jimerson et al., 2018; Thai et al., 2020). The YMC-EXP corpus, developed for over ten years, provided 100 hours for the ASR training, validation, and test corpora. There are 505 recordings from 34 speakers in the YMC-EXP corpus, and the transcription for the YMC-EXP were all carefully proofed by an expert native-speaker linguist. As shown in Table 1, we offer a train-valid-test split where there is no overlap in content between the sets. The partition considers the balance between speakers and relative size for each part.

As introduced in Section 2.2, the YMC-NT corpus has *both* expert and novice transcription. It includes only three speakers for a total of 8.36 hours. In the recordings of two consultants, the environment is relatively clean and free of background noise. The speech of the other individual, however, is frequently affected by background noise. This seems coincidental as all three were recorded together, one after the other in random order. But given this situation, we split the corpus into three sets: clean-dev (speaker EGS), clean-test (speaker CTB), and noise-test (speaker FEF; see Table 1).

The "code-switching" discussed in 2.3 (b) introduces different phonological representations and makes it difficult to train an HMM-based model using language lexicons. Therefore, previous work (Mitra et al., 2016) using the HMM-based system for YMC did not consider phrases with "code-switching". To compare our model with their results, we have used the same experimental corpus in our evaluation. Their corpus (YMC-EXP(-CS)), shown in Table 1, is a subset of the YMC-EXP; the YMC-EXP(-CS) corpus does not contain "code-switching" phrases, i.e., phrases with words that were tagged as Spanish origin and transcribed without tone.

3 ASR Experiments

3.1 End-to-End ASR

As ESPNet (Watanabe et al., 2018) is widely used in open-source end-to-end ASR research, our end-to-end ASR systems are all constructed using ESPNet. For the encoder, we employed the conformer structure (Gulati et al., 2020), while for the decoder we used the transformer structure to condition the full context, following the work of Karita et al. (2019b). The conformer architecture is a state-of-the-art innovation developed from the previous transformer-based encoding methods (Karita et al., 2019a; Guo et al., 2020). A comparison between the conformer and transformer encoders shows the value of applying state-of-the-art end-to-end ASR to ELs.

3.2 Experiments and Results

As discussed above, our end-to-end model applied an encoder-decoder architecture with a conformer encoder and a transformer decoder. The architecture of the model follows Gulati et al. (2020) while its configuration follows the aishell conformer recipe from ESPNet.[6] The experiment is reproducible using ESPNet.

As the end-to-end system models are based on word pieces, we adopted CER and WER as evaluation metrics. They help demonstrate the system performances at different levels of graininess. But because the HMM-based systems were decoding with a word-based lexicon, for comparison to HMM we only use the WER metric. To thoroughly examine the model, we conducted several comparative experiments, as discussed in continuation.

(a) Comparison with HMM-based Methods: We first compared our end-to-end method with the Deep Neural Network-Hidden Markov Model (DNN-HMM) methods proposed in Mitra el al. (2016). In this work, Gammatone Filterbanks (GFB), articulation, and pitch are configured for the DNN-HMM model. This baseline is a DNN-HMM model using Mel Filterbanks (MFB). In recent unpublished work, Kwon and Kathol develop a latest state-of-the-art CNN-HMM-based ASR model[7] for YMC based on the lattice-free Maximum Mutual Information (LF-MMI) approach, also known as “chain model” (Povey et al., 2016). The experimental data of the above HMM-based models is YMC-EXP(-CS) discussed in Section 2.4. For the comparison, our end-to-end model adopted the same partition to ensure fair comparability with their results.

Table 2 shows the comparison between DNN-HMM systems and our end-to-end system on YMC-EXP(-CS). It indicates that even without an external language lexicon the end-to-end system significantly outperforms both the DNN-HMM baseline models and the CNN-HMM-based state-of-the-art model.

In Section 2.3 (b), we note that “code-switching” is invariably present in EL speech (e.g., YMC). Thus, ASR models built on ”code-switching-free corpora (like YMC-EXP[-CS]) are not practical for real-world usage. However, a language lexicon is available only for the YMC-EXP(-CS) corpus so we cannot conduct HMM-based experiments with either YMC-EXP or YMC-NT corpora.

Model	**Feature**	**WER**
DNN-HMM	MFB	36.9
DNN-HMM	GFB + Articu. +Pitch	31.1
CNN-HMM (Chain)	MFCC	19.1
E2E-Conformer	MFB + Pitch	**15.4**

Table 2: Comparison between HMM-based Models and the End-to-End Conformer (E2E-Conformer) Model on YMC-EXP(-CS) that is a subset of the YMC-EXP **without** “code-switching”.

Model	**CER** dev/test	**WER** dev/test
E2E-RNN	9.2/9.3	19.1/19.2
E2E-Transformer	7.8/7.9	16.3/16.7
E2E-Conformer	**7.7/7.7**	**16.0/16.1**

Table 3: End-to-End ASR Results on YMC-EXP (corpus **with** “code-switching”)

(b) Comparison with Different End-to-End ASR Architectures: We also conducted experiments comparing models with different encoders and decoders on the YMC-EXP corpus. For a Recurrent Neural Network-based (E2E-RNN) model, we followed the best hyper-parameter configuration, as discussed in Zeyer et al. (2018). For a Transformer-based (E2E-Transformer) model, the same configuration from Karita et al. (2019b) was adopted. Both models shared the same data preparation process as the E2E-Conformer model.

Table 3 compares different end-to-end ASR architectures on the YMC-EXP corpus.[8] The E2E-Conformer obtained the best results, obtaining significant WER improvement as compared to the E2E-RNN and the E2E-Transformer models. The E2E-Conformer’s WER on YMC-EXP(-CS) is slightly lower than that obtained for the whole YMC-EXP corpus, despite a significantly smaller training set in the YMC-EXP(-CS) corpus. Since the subset excludes Spanish words, “code-switching” may well be a problem to consider in ASR for endangered languages such as YM.

---

[6] See Appendix for details about the model configuration.

[7] See Appendix for details about the model configuration.

[8] The train set in YMC-EXP is significantly larger than that in YMC-EXP(-CS), the YMC-EXP corpus from which all lines containing a Spanish-origin word have been removed.

Transcription Level	**CER** dev/test	**WER** dev/test
Surface	8.0/**7.6**	16.6/16.3
Underlying	**7.7**/7.7	**16.0/16.1**

Table 4: E2E-Conformer Results for Two Transcription Levels (Underlying represents morphological divisions and underlying phonemes before the application of phonological rules; Surface is reflective of spoken forms and lacks morphological parsing)

Corpus	**CER** dev/test	**WER** dev/test
10h	19.4/19.5	39.1/39.2
20h	12.6/12.7	26.2/26.2
50h	8.6/8.7	18.0/18.0
Whole (92h)	**7.7/7.7**	**16.0/16.1**

Table 5: E2E-Conformer Results on Different Corpus Size

(c) Comparison with Different Transcription Levels: In addition to comparing model architectures, we compared the impact of transcription levels on the ASR model. E2E-Conformer models with the same configurations were trained using both the surface and the underlying transcription forms, which are discussed in Section 2.3. We also trained separate RNN language models for fusion and unigram language models to extract word pieces for different transcription levels.

Table 4 shows the E2E-Conformer results for both underlying and surface transcription levels. As introduced in Section 2.3, the surface form reduces several linguistic and phonological processes compared to the underlying practical form. The results indicate that the end-to-end system is able to automatically infer those morphological and phonological processes and maintain a consistent low error rate.

(d) Comparison with Different Corpus Sizes: As introduced in Section 1, most ELs are considered low-resource for ASR purposes. To measure the impact of resource availability on ASR accuracy we trained the E2E-Conformer model on 10, 20, and 50 hours subsets of YMC-EXP. The results demonstrate the model performances over different sizes of resources.

Table 5 shows the E2E-Conformer performances on different amounts of training data. It demonstrates how the model consumes data. As corpus size is incrementally increased, WER decreases significantly. It is apparent that the model still has the capacity to improve performance with more data. The result also indicates that our system can get reasonable performances from 50 hours of data. This would be an important guideline when we collect a new EL database.

Model	**CER** dev/test	**WER** dev/test
E2E-RNN	10.3/9.9	26.8/25.4
E2E-Transformer	**9.1**/9.1	23.7/**21.7**
E2E-Conformer	9.9/**8.6**	**23.5/21.7**

Table 6: E2E-Conformer Results on another EL: Highland Puebla Nahuatl

Corpus	**CER** dev/test	**WER** dev/test
10h	18.3/17.5	44.7/43.3
20h	14.2/12.9	34.8/33.3
50h	11.0/10.2	27.0/24.9
Whole (120h)	**9.9/8.6**	**23.5/21.7**

Table 7: E2E-Conformer Results on another EL: Highland Puebla Nahuatl (Different Corpus Size)

(e) The Framework Generalizability: To test the end-to-end ASR systems' generalization ability, we conducted the same end-to-end training and test procedures on another endangered language: Highland Puebla Nahuatl (high1278; azz). This corpus is also open access under the same CC license.[9] It comprises 954 recordings that total 185 hours 22 minutes, including 120 hours transcribed data in ELAN and 65 hours still only in Transcriber and not used in ASR training.[10]

Table 6 shows the performance of three different end-to-end ASR architectures on Highland Puebla Nahuatl. For this language the E2E-Conformer again offers better performances over the other models. Table 7 shows the E2E-Conformer performances on different amounts of training data for Highland Puebla Nahuatl. We can observe that 50-hour is a reasonable size for an EL, which is similar to the experiments in Table 5. These experiments indicate the general ability to consistently apply end-to-end ASR systems across ELs.

---

[9] `http://openslr.org/92`

[10] The recordings are almost all with two channels and two speakers in natural conversation.

Error Types	**Novice**	**ASR**
Enclitics (=)	**96**	243
Prefixes (-)	141	**62**
Glottal Stop (')	341	**209**
Parenthesis	1607	**302**
Tone	4144	**3241**
Stem-Nasal (n)	**0**	6
Others	**4263**	10175
Total	10592	14232

Table 8: Character Error-type Distribution of Novice and ASR (by number of errors)

4 Novice Transcription Correction

Finally, this paper presents novice transcription correction (NTC) as a task for EL documentation. That is, in this experiment we explore not only the possibility of using ASR to enhance the accuracy of a YM novice transcription but to combine both novice transcription and ASR to achieve accurate results that surpass that of either component. Below we first analyze patterns manifested in novice transcriptions. Next, we introduce two baselines that fuse ASR hypotheses and novice transcription for the NTC task.

4.1 Novice Transcription Error

As mentioned in Section 1, transcriber shortages have been a severe challenge for EL documentation. Before 2019, only the native speaker linguist, Rey Castillo García, could accurately transcribe the segments and tones of YMC. To mitigate the YMC transcriber shortage, in 2019 Castillo began to train another speaker, Esteban Guadalupe Sierra. First, a computer course was designed to incrementally teach Guadalupe segmental and tonal phonology. In the next stage, he was given YMC-NT corpus recordings to transcribe. Compared to the paired expert transcription, the novice achieved a CER of 6.0% on clean-dev, defined in Table 1. However, it is not feasible to spend many months training speakers with no literacy skills to acquire the transcription proficiency achieved by Guadalupe in our project. Moreover, even with a 6.0% CER, there are still enough errors so as to require significant annotation/correction by the expert, Castillo. The state-of-the-art ASR system (e.g., E2E-Conformer) shown in Table 3 gets an 8.2% CER on the clean-dev set, more errors than the novice CER. So for YMC, ASR is still not a good enough substitute for a proficient novice.

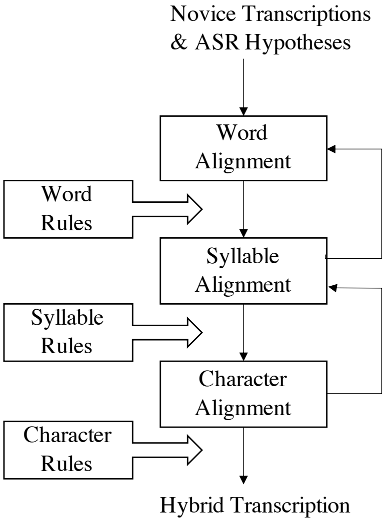


Figure 1: Novice-ASR Fusion Process

As Amith and Castillo worked with the novice, they saw a repetition of types of errors that they worked to correct by giving the novice exercises focused on these transcription shortcomings. The end-to-end ASR, however, has demonstrated a different pattern of errors. For example, it developed a fair understanding of the rules for suppleting tones, marked by parentheses around the suppleted tones. Rather than over-specify the NTC correction algorithm, we first analyzed the error-type distribution using the Clean-dev from the YMC-NT corpus, as shown in Table 8.

4.2 Novice-ASR Fusion

Rapid comparison of the types of errors for each transcription (novice and ASR) demonstrated consistent patterns and has led us to hypothesize that a fusion system might automatically correct many of these errors. Two baseline methods are examined for the fusion: a voting-based system (Fiscus, 1997) and a rule-based system.

The voting-based system follows the definition in (Fiscus, 1997) that combines hypotheses from different ASR models with novice transcription.

The framework of rule-based fusion is shown in Figure 1. The rules are defined in different linguistic units: words, syllables, and characters. They assume a hierarchical alignment between the novice transcription and ASR hypotheses. The rules are applied to the transcription from word to syllable to character level. The rules are developed based on continual evaluation of the novice's progress. Thus they will be different but discoverable when

Model	Clean-Dev		Clean-Test		Noise-Test		Overall	
	CER	WER	CER	WER	CER	WER	CER	WER
A. Novice	6.0	21.5	6.4	22.6	**8.4**	**26.6**	6.8	23.1
B. E2E-Transformer	9.8	23.1	8.8	21.2	24.3	47.0	12.9	28.1
C. E2E-Conformer	8.2	19.6	8.2	19.1	23.6	44.1	12.0	25.3
D. E2E-Conformer(50h)	10.5	25.0	9.9	23.7	25.7	50.1	14.0	30.5
E. Fusion1 (A+C)	6.3	20.6	6.9	22.0	13.1	38.6	8.2	25.4
F. Fusion1 (A+D)	7.0	22.9	7.5	24.5	14.0	41.5	8.9	28.0
G. Fusion2 (A+C)	5.1	17.6	5.5	18.7	9.6	30.3	6.3	21.1
H. Fusion2 (A+D)	5.5	19.4	5.9	20.4	10.1	32.6	6.8	23.0
I. ROVER (A+B+C)	4.7	**14.6**	**4.6**	**13.8**	12.4	32.6	6.5	**18.5**
J. ROVER-Fusion2 (A+B+C+E)	**4.5**	16.1	4.7	16.7	9.0	28.3	**5.7**	19.3

Table 9: NTC Results on YMC-NT (the results are evaluated using the expert transcription in YMC-NT). Model D is trained with a 50-hour subset of the YMC-EXP as shown in Table 5.

applied to a new language. However, the general principle should be applicable to other ELs: Novice trainees will learn certain transcription tasks easier than others. Below we explain the rules for YMC.

Word Rules: If a word from the novice transcription is Spanish (i.e., no tones and no linguistic indications [-, =, '] that mark it as Mixtec), keep the novice transcription. If the novice has extra words, not in the ASR hypothesis, keep those extra words.

Syllable Rules: If a novice syllable is tone initial, use the corresponding ASR syllable. If the novice and the ASR have identical segments but different tones, use the ASR tones. When an ASR syllable has CVV or CV'V, and its corresponding novice syllable has CV,[11] use the ASR syllable (CVV or CV'V). If the tone from either transcription system follows a consonant (except a stem-final n), use the other system's transcription.

Character Rules: If the ASR has a hyphen, equal sign, parentheses, glottal stop which is absent from the novice transcription, then always trust the ASR and maintain the aforementioned symbols in the final transcription.

We apply the edit distance (Wagner and Fischer, 1974) to find the alignment between the ASR model hypothesis $\{C_1, ..., C_n\}$ and the Novice transcription $\{C'_1, ..., C'_m\}$. The L_I, L_D, L_S are introduced in the dynamic function as the insertion, deletion, and substitution loss, respectively. In the naive setting, L_I, L_D are both set to 1. The L_S is set to 1 if C_i is different from C'_j and 0 otherwise. This setting is computation-efficient. However, it does not consider how the contents mismatch between the C_i and C_j. Therefore, we adopt a hierarchical dynamic alignment. In this method, the character alignment follows the native setting. While the $L_S(C_i, C'_j)$ for syllable alignment is defined as the normalized character-level edit distance between C_i and C'_j as follows:

$$L_S(C_i, C'_j) = \frac{D[C_i, C'_j]}{|C_i|} \quad (1)$$

where the $|C_i|$ is the lengths of the syllable. Similarly, the $L_S(C_i, C'_j)$ for word alignment is defined based on syllable alignment.

5 NTC Experiments

5.1 Experimental Settings

The novice transcription, the E2E-Transformer model, and the E2E-Conformer model were considered as baselines for the NTC task. To evaluate the system for reduced training data, we also show our results of E2E-Conformer trained with a 50-hour subset. For the end-to-end models, we adopted the trained model from Section 3 with the same decoding set-ups. To test the effectiveness of the hierarchical dynamic alignment, we tested the data with two fusion systems, namely Fusion1 and Fusion2. The Fusion1 system used the naive settings of edit distance, while the Fusion2 system adopted the hierarchical dynamic alignment. Both fusion systems adopt rules defined in Section 4.2. Two configurations for voting-based methods were tested. The first "ROVER" combined three hypotheses (i.e., the E2E-Transformer, the E2E-Conformer, and the

---

[11] A CV syllable can occur in a monomoraic word. But novice will often write a CV word when it should be CVV or CV'V. Stem-final syllables can be CV, CVV or CV'V. But novice tends to write CV in these cases.

Novice). In contrast, the "ROVER-Fusion2" combined the Fusion2 system with the above three.

5.2 Results

As shown in Table 9, voting-based methods and rule-based methods all significantly reduce the novice errors for clean speech.[12] However, for the noise-test, the novice transcription is the most robust method. For overall results, the ROVER system (model I) has a lower WER, while the ROVER-Fusion2 system (model J) reaches a lower CER. Model J significantly reduces specific errors, including tone errors (25%), enclitic errors (50%), and parentheses errors (87.5%). In addition, models D, F, and H indicate that the system could still reduce clean-environment novice errors using ASR models trained with a 50-hour subset of the YMC-EXP corpus.

As we discussed in Section 4, novice and ASR transcriptions manifest distinct patterns of error and thus can be used to complement each other. Table 9 shows that our proposed rule-based and voting-based fusion methods can potentially eliminate the errors that come from the novice transcriber, and it can mitigate the transcriber shortage problems based on these fusion methods. However, we should note that a noisy recording condition would negatively affect a fusion approach as ASR does poorly under such conditions (>23% CER), and for practical purposes, the novice transcription alone (<8.5%) is much more accurate. In such conditions we should rely on the novice transcriber alone.

6 Conclusion and Future Work

This work presents an open-source endangered language corpus in Yoloxóchitl Mixtec and a comparative and reproducible study on various approaches to end-to-end ASR. We demonstrate that end-to-end approaches are feasible and present comparable results over conventional HMM approaches, which require resources such as language lexicons not necessary with end-to-end ASR. Additionally, we propose novice transcription correction as a potential task for ASR in EL documentation. We examine two methods to approach this task. The first is a rule-based approach that uses hierarchical dynamic alignment and linguistic rules to perform novice-ASR hybridization. The second is a voting-based method that combines hypotheses from the novice and end-to-end ASR systems. Empirical studies on the YMC-NT corpus indicate that both methods significantly reduce the CER/WER of the novice transcription for clean speech.

The above discussion suggests that a useful approach to EL documentation using both human and computational (ASR) resources might focus on training each system (human and ASR) for particular transcription tasks. If we know from the start that ASR will be used to correct novice transcriptions in areas of difficulty, we could train an ASR system to maximize accuracy in those areas that challenge novice learning.

References

Oliver Adams, Trevor Cohn, Graham Neubig, Hilaria Cruz, Steven Bird, and Alexis Michaud. 2018. Evaluating phonemic transcription of low-resource tonal languages for language documentation. In *LREC 2018 (Language Resources and Evaluation Conference)*, pages 3356–3365.

Antonios Anastasopoulos and David Chiang. 2017. A case study on using speech-to-translation alignments for language documentation. In *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 170–178.

Claude Barras, Edouard Geoffrois, Zhibiao Wu, and Mark Liberman. 1998. Transcriber: a free tool for segmenting, labeling and transcribing speech. In *Preceedings of the First international conference on language resources and evaluation (LREC)*, pages 1373–1376.

Malgorzata Ćavar, Damir Ćavar, and Hilaria Cruz. 2016. Endangered language documentation: Bootstrapping a chatino speech corpus, forced aligner, asr. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC)*, pages 4004–4011.

Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778.

Hilaria Cruz and Joseph Waring. 2019. Deploying technology to save endangered languages. *arXiv preprint arXiv:1908.08971.*

Amit Das and Mark Hasegawa-Johnson. 2016. An investigation on training deep neural networks using

---

[12] Note that the rules are developed based on YM specifics, so the result cannot be applied to other languages directly. Readers should view it as a case study.

probabilistic transcriptions. In *Interspeech*, pages 3858–3862.

Jonathan G Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354.

Pegah Ghahremani, Bagher BabaAli, Daniel Povey, Korbinian Riedhammer, Jan Trmal, and Sanjeev Khudanpur. 2014. A pitch extraction algorithm tuned for automatic speech recognition. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2494–2498.

Alex Graves and Navdeep Jaitly. 2014. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772.

Lenore A Grenoble, Peter K Austin, and Julia Sallabank. 2011. Handbook of endangered languages.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. 2020. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*.

Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, et al. 2020. Recent developments on espnet toolkit boosted by conformer. *arXiv preprint arXiv:2010.13956*.

Vishwa Gupta and Gilles Boulianne. 2020a. Automatic transcription challenges for Inuktitut, a low-resource polysynthetic language. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2521–2527.

Vishwa Gupta and Gilles Boulianne. 2020b. Speech transcription challenges for resource constrained indigenous language cree. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 362–367.

Nils Hjortnaes, Niko Partanen, Michael Rießler, and Francis M Tyers. 2020. Towards a speech recognizer for komi, an endangered and low-resource uralic language. In *Proceedings of the Sixth International Workshop on Computational Linguistics of Uralic Languages*, pages 31–37.

Robbie Jimerson and Emily Prud'hommeaux. 2018. Asr for documenting acutely under-resourced indigenous languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Robbie Jimerson, Kruthika Simha, Raymond Ptucha, and Emily Prudhommeaux. 2018. Improving asr output for endangered language documentation. In *Proc. The 6th Intl. Workshop on Spoken Language Technologies for Under-Resourced Languages*, pages 187–191.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. 2019a. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 449–456.

Shigeki Karita, Nelson Enrique Yalta Soplin, Shinji Watanabe, Marc Delcroix, Atsunori Ogawa, and Tomohiro Nakatani. 2019b. Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration. *Proceedings of Interspeech 2019*, pages 1408–1412.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Christian Lehmann. 1999. Documentation of endangered languages. In *A priority task for linguistics. ASSIDUE Arbeitspapiere des Seminars für Sprachwissenschaft der Universität Erfurt*, 1.

Kohei Matsuura, Sei Ueno, Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara. 2020. Speech corpus of ainu folklore and end-to-end speech recognition for ainu language. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2622–2628.

Alexis Michaud, Oliver Adams, Trevor Cohn, Graham Neubig, and Séverine Guillaume. 2018. Integrating automatic transcription into the language documentation workflow: Experiments with na data and the persephone toolkit. *Language Documentation & Conservation*, 12:393–429.

Alexis Michaud, Eric Castelli, et al. 2014. Towards the automatic processing of yongning na (sino-tibetan): developing a'light'acoustic model of the target language and testing'heavyweight'models from five national languages. In *4th International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU 2014)*, pages 153–160.

Vikramjit Mitra, Andreas Kathol, Jonathan D Amith, and Rey Castillo García. 2016. Automatic speech transcription for low-resource languages-the case of yoloxóchitl mixtec (mexico). In *Proceedings of Intespeech 2016*, pages 3076–3080.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Proc. Interspeech 2019*, pages 2613–2617.

Ngoc-Quan Pham, Thai-Son Nguyen, Jan Niehues, Markus Müller, and Alex Waibel. 2019. Very deep self-attention networks for end-to-end speech recognition. *Proceedings of Interspeech 2019*, pages 66–70.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. 2016. Purely sequence-trained neural networks for asr based on lattice-free mmi. In *Interspeech*, pages 2751–2755.

Jorge A Suárez. 1983. *The mesoamerican indian languages*. Cambridge University Press.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4278–4284.

Bao Thai, Robert Jimerson, Raymond Ptucha, and Emily Prud'hommeaux. 2020. Fully convolutional asr for less-resourced endangered languages. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 126–130.

Robert A Wagner and Michael J Fischer. 1974. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson-Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al. 2018. Espnet: End-to-end speech processing toolkit. *Proceedings of Interspeech 2018*, pages 2207–2211.

Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. 2006. Elan: a professional framework for multimodality research. In *5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1556–1559.

Alexander Zahrer, Andrej Zgank, and Barbara Schuppler. 2020. Towards building an automatic transcription system for language documentation: Experiences from muyu. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2893–2900.

Albert Zeyer, Kazuki Irie, Ralf Schlüter, and Hermann Ney. 2018. Improved training of end-to-end attention models for speech recognition. *Proceedings of Interspeech 2018*, pages 7–11.

A Appendices

Experimental Settings for End-to-End ASR: All the end-to-end ASR systems adopted the hybrid CTC/Attention architecture integrated with an RNN language model. The best model was selected on the basis of performance on the development set. The input acoustic features were 83-dimensional log-Mel filterbanks features with pitch features (Ghahremani et al., 2014). The window length and the frameshift were set to 25ms and 10ms. SpecAugmentation are adopted for data augmentation (Park et al., 2019). The prediction targets were the 150-word pieces trained using unigram language modeling (Kudo and Richardson, 2018) (both for surface and underlying form). All the end-to-end models are fused with RNN language models.[13] The CTC ratio for Hybrid CTC/Attention was set to 0.3. The decoding beam size was 20. Training and Testing are based on Pytorch.

E2E-Conformer Configuration: The E2E-Conformer used 12 encoder blocks and 6 decoder blocks. All the blocks adopted a 2048 dimension feed-forward layer and four-head multi-head-attention with 256 dimensions. Kernel size in the Conformer block was set to 15. For training, the batch size was set to 32. Adam optimizer with 1.0 learning rate and Noam scheduler with 25000 warmup-steps were used in training. We trained for a max epoch of 50. The parameter size is 43M.

E2E-RNN Configuration: The E2E-RNN used 3 encoder blocks and 2 decoder blocks. All the blocks adopt 1024 hidden units. Location-based attention adopted 1024-dim attention. Adadelta was chosen as the optimizer, and we trained for a max epoch of 15. The parameter size is 108M.

E2E-Transformer Configuration: The E2E-Transformer used 12 encoder blocks and 6 decoder blocks. All the blocks adopted a 2048 dimension feed-forward layer and four-head multi-head-attention with 256 dimensions. Adam optimizer with 1.0 learning rate and Noam scheduler with 25000 warmup-steps were used in training. We trained for a max epoch of 100. The parameter size is 27M.

Experimental Settings for HMM-based ASR: The acoustic feature input for this model is 40 dimensional Mel Frequency Cepstral Coefficients (MFCC). The lexicon for HMM-based models is

---

[13] Our experiments show that the RNN language model reduces WER by about 1%.

phone-based. The transcriptions are mapped to surface representations and then to phones (a total of 197 phones, as each tone for a given vowel, is a different phone). There are 22,465 total entries in the lexicon. The chain model is trained with a sequence-level objective function and operates with an output frame rate of 30 ms, three times longer than the previous standard. The longer frame rate increases decoding speed, which in turn makes it possible to operate with a significantly deeper DNN architecture for acoustic modeling. The best results were achieved with a neural network based on the ResNet architecture (Szegedy et al., 2017). This consists of an initial layer for Linear Discriminative Analysis (LDA) transformation and subsequent alternating 160-dimensional bottleneck layers, adding up to 45 layers in total. The DNN acoustic model is then compiled with a 4-gram language model into a weighted finite-state transducer for word sequence decoding.

Mode Effects’ Challenge to Authorship Attribution

Haining Wang[1], **Allen Riddell**[1], and **Patrick Juola**[2]

[1]Indiana University Bloomington, Bloomington, Indiana
[2]Duquesne University, Pittsburgh, Pennsylvania
`{hw56|riddella}@indiana.edu  juola@mathcs.duq.edu`

Abstract

The success of authorship attribution relies on the presence of linguistic features specific to individual authors. There is, however, limited research assessing to what extent authorial style remains constant when individuals switch from one writing modality to another. We measure the effect of writing mode on writing style in the context of authorship attribution research using a corpus of documents composed online (in a web browser) and documents composed offline using a traditional word processor. The results confirm the existence of a “mode effect” on authorial style. Online writing differs systematically from offline writing in terms of sentence length, word use, readability, and certain part-of-speech ratios. These findings have implications for research design and feature engineering in authorship attribution studies.

1 Introduction

That authorship attribution techniques work as reliably as they do has been attributed to the fact that each individual has a distinctive writing style. Texts written by the same author can be recognized by analyzing lexical and syntactic features in documents (Juola, 2006). This principle is practically successful in a variety of settings (Abbasi and Chen, 2008; Overdorf and Greenstadt, 2016; Afroz et al., 2014). In some cases, however, authorial style is challenging to detect. For example, authorial style fades as time goes by (Glover and Hirst, 1996; Baayen et al., 2002), varies considerably in collaborative environments (Graham et al., 2005; Kestemont et al., 2018; Zangerle et al., 2019), and drifts depending on document genre (Stamatatos, 2018; Koppel et al., 2007; Sapkota et al., 2016).

The aforementioned changes are conspicuous due to the fact that there are certain markers indicating that a document may have been written in a fashion that will lead to stylistic variation. For instance, the presence of genre-specific words indicates a document may present a puzzle for standard authorship attribution techniques. Sometimes, however, documents which may challenge an analysis of authorial style can be unannounced. This paper shows that authorial style changes with respect to sentence length, word use, readability, and certain part-of-speech ratios when the writing environment switches from traditional word processing software to an input box of a web browser.

2 Mode Effects

Originally developed in survey research and educational testing, a “mode effect” describes the following phenomenon: a respondent may answer the same question differently depending on how a survey is administered (e.g., online vs. phone) (Hochstim, 1967; Leeson, 2006). Although discussion continues about mode effects’ underlying mechanism (Kreuter et al., 2008; Sidi et al., 2017), contextualized magnitude (Carpenter and Alloway, 2019; Washburn et al., 2017), and adjustment methods (Kolenikov and Kennedy, 2014), a consensus has been reached that mode effects can impair survey validity. For instance, Tourangeau et al. (2000) compiled six studies investigating illicit drug use with self-administered and interviewer-administered surveys. The results showed that illicit drug use was reported at higher rates when questions were administered without an interviewer present.

Functionally, writing modality resembles survey modality: the style observed in an individual’s writing may vary depending on how the writer composes the document. A document written by hand may vary from a document composed using traditional word processing software. Further variation may be observed if the document is typed into a

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1146–1155
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

text box in a web browser. Therefore, this research uses "mode effect" to label such differences.

3 Data

Participants in this experiment were recruited on Amazon Mechanical Turk (MTurk). Two distinct types of writing were collected from each of the 18 participants: (1) ca. 6,500 words of pre-existing formal writing and (2) a short ca. 500-word open-ended response to an essay prompt. For the pre-existing writing samples, participants were asked to "Submit at least 6500 words total from multiple documents of your own writing that was done for a formal purpose (school essays, grant proposals, etc)." For the 500-word essay, respondents were asked to describe [their] neighborhood to someone who has never been there before as part of a college application.[1] Respondents also completed a demographic questionnaire, reporting their gender and age bracket.

Responses that were not in English or which seemed very likely to be inauthentic were excluded. (Kennedy et al. (2018) discusses challenges dealing with MTurk surveys). We also excluded one response which appeared to contain writing copied (without attribution) from online sources. The pre-existing writing samples were further processed in order to remove personally identifying information. Lengthy quotations, headings, tables, and figures were also removed.

The data for this experiment are a subset of data collected as part of research seeking to replicate results in Brennan et al. (2012). In the full replication experiment, respondents were randomly given one of four essay prompts. In this paper, we only used the responses by respondents randomly assigned to the "control" condition. These respondents provided pre-existing writing samples and a response to the essay prompt mentioned above. Responses were collected between March 29th and June 1st, 2019. 14 out of 18 respondents reported their age as "18-34". Self-reported gender was also collected. Ten of the respondents were men and eight were women.

The overwhelming majority of pre-existing writing samples collected were essays written for undergraduate courses. Many essays discussed films and literary works. Many appeared to be written for political science and business courses.

The writing prompt generally elicited the desired response: respondents wrote about their neighborhood using formal or semi-formal prose.

To check that all writing collected exhibited approximately the same degree of formality, we compared the formality of the writing in the offline corpus with the formality of the writing in the online corpus using a formality score developed by Heylighen and Dewaele (1999). We found that the formality scores in each corpus were similar.[2]

4 Method

This paper focuses on stylistic differences introduced by seemingly innocuous variation in the mode used to enter a text—offline composition vs. online typing into a text box. We are interested in whether a mode effect occurs in writing. To the extent that it is observed, we wish to know if its impact on an author's style is predictable. Does writing mode induce similar changes in the writing style of different individuals?

To answer these questions, we extract linguistic features from the documents written by the 18 participants. We then use a Bayesian hierarchical model to estimate differences in the rates at which the linguistic features appear in texts written using different writing modes.

5 Modeling Authorial Style

5.1 Feature Selection

We use a set of high-level, familiar linguistic features in our study. Our "Comparative Style" ("CS") feature set aims to capture word-, sentence-, and chunk-level features. All features are described in Table 1. For sentence-level features, white space between words is not counted as a character. Punctuation includes periods, exclamation marks, question marks, commas, semicolons, colons, and apostrophes. For function words we use the list of 512 words from Koppel et al. (2005). The Voice of America (VOA) Special English word list contains 1,512 frequently-used words which are used

---

[1] The full prompt reads: "TOPIC: You are asked as part of a college application to describe your neighborhood to someone who has never been there before. Discuss the houses, people, stores, parks. Anything you think is relevant." The prompt is taken from Brennan et al. (2012).

[2] The mean formality score and standard deviation for the offline corpus were 59.2 and 17.3 respectively. The mean formality score and standard deviation for the online corpus were 57.9 and 17.6 respectively. The formality scores were calculated for each sentence using the following formula: $(\text{noun frequency} + \text{adjective freq.} + \text{preposition freq.} + \text{article freq.} - \text{pronoun freq.} - \text{verb freq.} - \text{adverb freq.} - \text{interjection freq.} + 100)/2$ (Heylighen and Dewaele, 1999).

Table 1: This table describes features in the Comparative Style feature set. The last column mentions an example sentence and an example text chunk. The first sentence of the abstract is the example sentence. The abstract is the example text chunk.

Level	No.	Feature	Abbreviation	Explanation/Example
Word	1	Word length in syllables	WordLenSyll	E.g. the word "mode" has one syllable while the word "effect" has two.
	2	Word length in characters	WordLenChar	E.g. the word "mode" has four characters while the word "effect" has five.
Sentence	3	Sentence length in syllables	SentLenSyll	E.g. the example sentence has 35 syllables.
	4	Sentence length in words	SentLenWord	The example sentence has 16 words.
	5	Sentence length in characters	SentLenChar	E.g. the example sentence has 99 characters.
	6	Punctuation to character ratio	PuncChar	E.g. the PuncChar ratio of the example sentence is 1/99.
	7	Function word to word ratio	FuncWord	E.g. the FuncWord ratio of the example sentence is 6/16.
	8	Special English ratio	SplEng	E.g. the SplEng ratio of the example sentence is 6/16, because "on", "the", "to", "individual," and two "of" are in the sentence.
	9	Common word ratio	CommWord	E.g. the CommWord ratio of the example sentence is 12/16, because "success", "on", "the", "presence", "linguistic", "feature", "specific", "to", "individual", "author," and two "of" are in the sentence.
Chunk	10	Adjective to noun ratio	AdjNoun	E.g. the AdjNoun ratio of the abstract is 18/44.
	11	Verb to noun ratio	VerbNoun	E.g. the VerbNoun ratio of the abstract is 16/44.
	12	Pronoun to noun ratio	PronNoun	E.g. the PronNoun ratio of the abstract is 2/46.
	13	Adverb to adjective ratio	AdvAdj	E.g. the AdvAdj ratio of the abstract is 3/18.
	14	Flesch–Kincaid grade level	FleschKincaid	E.g. the FleschKincaid score of the abstract is 13.11.
	15	Gunning fog index	GunningIdx	E.g. the GunningIdx ratio of the abstract is 50.00.

in VOA Special English reporting (Voice of America, 2007). The 8,013-word "common word" list is taken from the College English Test Band 4 and 6 (CET-4/6), the nationwide English proficiency test used in mainland China. The lists are used in the three sentence-level measures of vocabulary richness.

A chunk is defined as a sequence of consecutive sentences containing at least 150 words. The criterion "150 words" was arbitrarily chosen to balance meeting the length requirement of readability tests and the desire to extract as many chunk-level observations as possible in order to better estimate feature variability within writing modes. We experimented with different chunk lengths (e.g., 100, 150, 200) and found that our results did not depend strongly on chunk length.

The Flesch–Kincaid grade level Flesch (1948) and Gunning fog index (Gunning, 1968) were calculated for every chunk. More challenging texts are associated with lower Flesch-Kincaid levels and higher Gunning fog indexes.

Thirteen of the 15 CS features were transformed by taking the square root so that the feature distributions would be approximately Gaussian. The Flesch-Kincaid level and the Gunning fog index are left on the original scale as their distributions were already approximately Gaussian. Although the hierarchical model uses the transformed features, in subsequent visualizations and tables, parameter estimates are reported on the original scale.

5.2 Setup

We divide the documents into two groups: the "offline" documents, the pre-existing writing samples from the 18 subjects (authored using word processing software) and the "online" documents, written in a web browser in response to the essay prompt asking for a description of the writer's neighborhood.

To compare features across modes and individuals, we use a hierarchical model. Within each mode-specific group of documents, feature observations associated with an individual are modeled using a normal sampling model with an individual-specific mean and scale. The individual-specific means and scales are, in turn, modeled using a normal distribution and gamma distribution.

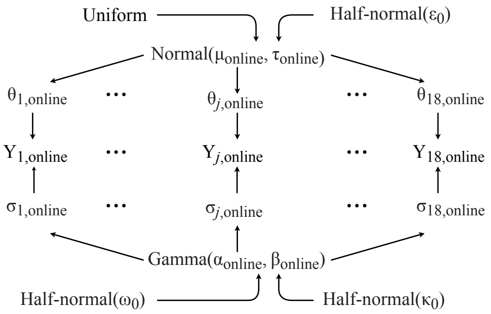


Figure 1: The model for CS feature observations for the online documents. The model for the CS feature observations for the offline documents is the same.

Figure 1 shows the hierarchical model for observations of CS features for the documents composed online. The model can be rendered in symbols as

$$\begin{aligned}\theta_{j,\text{online}} &\overset{i.i.d.}{\sim} \text{Normal}(\mu_{\text{online}}, \tau_{\text{online}}) \\ \sigma_{j,\text{online}} &\overset{i.i.d.}{\sim} \text{Gamma}(\alpha_{\text{online}}, \beta_{\text{online}}) \\ Y_{j,\text{online}} &\overset{i.i.d.}{\sim} \text{Normal}(\theta_{j,\text{online}}, \sigma_{j,\text{online}}), \\ & \quad j \in \{1, 2, ..., 18\}\end{aligned}$$

where $Y_{j,\text{online}} = \{y_{1,j}, y_{2,j}, ..., y_{n_j,j}\}_{\text{online}}$ are observations for the jth subject in the online mode. These observations are drawn from a normal sampling distribution $\text{Normal}(\theta_{j,\text{online}}, \sigma_{j,\text{online}})$. The individual-specific standard deviation $\sigma_{j,\text{online}}$ comes from a gamma distribution parameterized by shape α_{online} and rate β_{online}. The individual-specific mean $\theta_{j,\text{online}}$ is drawn from a normal distribution with a location μ_{online} and a scale τ_{online}. In addition, the μ_{online} was assigned a uniform prior distribution, while weakly informative priors were given to τ_{online}, α_{online}, and β_{online}. Each feature is modeled separately. The models for CS features in the offline mode mirrors those for the online mode.

In a pilot study, we considered using a Student-t distribution instead of a normal distribution as the sampling distribution. We found that the estimated degrees of freedom for these sampling distributions were sufficiently large (>30). Hence we concluded it was safe to use the simpler normal distribution as the sampling distribution.

Additionally, for simplicity, η, the group-level scale coming from Gamma(α, β), will be reported rather than the original parameterization. The effect size between modes was calculated with $\frac{\mu_{\text{online}} - \mu_{\text{offline}}}{\sqrt{(\eta^2_{\text{online}} + \eta^2_{\text{offline}})/2}}$ (Kruschke, 2014).

5.3 Decision Rule

Posterior differences between the two group means (μ_{online} and μ_{offline}) and group scales (η_{online} and η_{offline}) will be characterized using 95% highest posterior density (HPD) intervals and regions of practical equivalence (ROPE) (Kruschke and Liddell, 2018). The 95% HPD interval describes an interval in which a parameter is likely to be found. The ROPE specifies a region of practical equivalence around a null value.

When the 95% HPD for a parameter falls outside the ROPE, the null value is rejected, and the parameter is considered to be different from the null value. If the HPD falls entirely inside the ROPE, the null value is accepted. Otherwise, we withhold judgment.

The data can be analyzed using different ROPEs. Given the goals of this investigation, we use ROPEs associated with a "small" effect (Cohen's d=0.2, according to Cohen (1988)), that is, calculating ±0.1 standard deviations for every posterior difference as the upper and the lower ROPE limits around zero, as suggested by Kruschke (2018). There are many ways to calculate the effect size. We follow Kruschke (2014) in our calculation and refer to this effect size as Cohen's d.

6 Results

Before summarizing differences between online and offline writing across all individuals and features, we first consider how a single feature varies in writing from the 18 subjects. This analysis demonstrates how we use Bayesian methods to infer collective tendencies in the data.

6.1 A Close Look at Sentence Length

Do sentence lengths vary systematically by writing mode? We consider three measures of sentence length, one of which is "sentence length in characters." Even before performing any modeling, the individual and collective tendencies can be identified in a visualization of the data (Figure 2).

The box plots on the left-hand side of Figure 2 show that all but one of the 18 participants tended to use longer sentences in their offline documents. The variability in sentence lengths also tended to be greater. The right-hand side histograms of aggregated observations further confirm this characterization.

Figure 3 shows the 95% HPD intervals for the parameters of interest related to "sentence length in

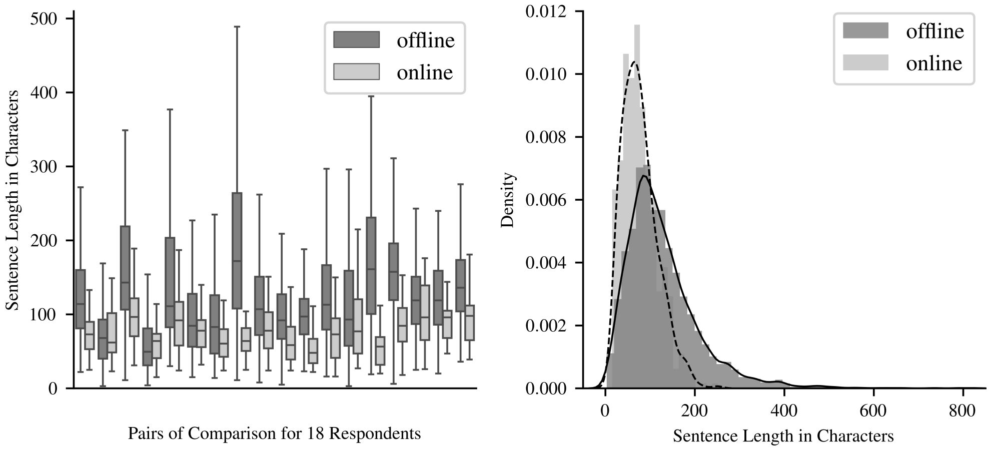


Figure 2: The grouped box plots and histograms with kernel density estimates for sentence length in characters.

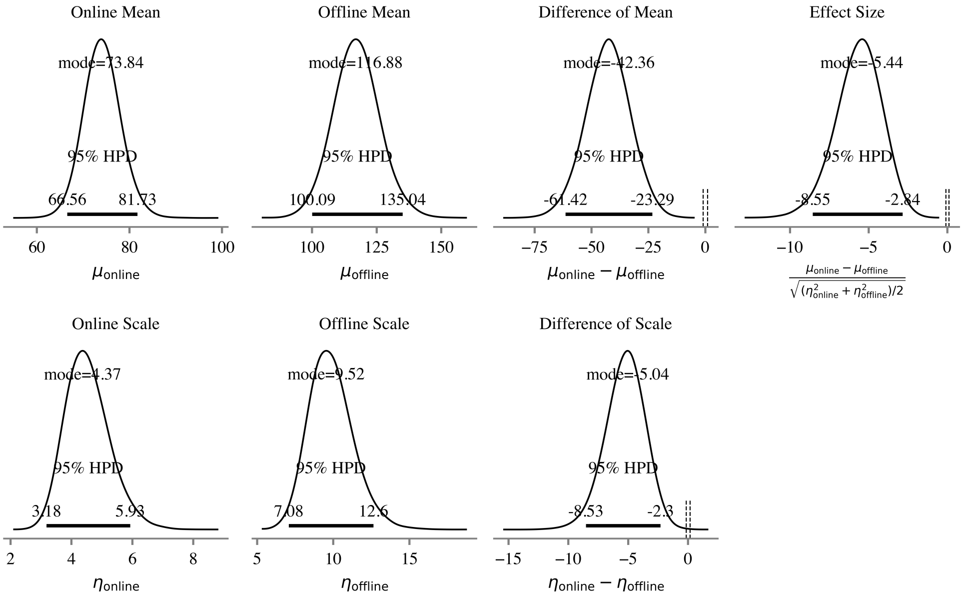


Figure 3: Posterior distribution of μ_{online}, μ_{offline}, $\mu_{\text{online}} - \mu_{\text{offline}}$, η_{online}, η_{offline}, $\eta_{\text{online}} - \eta_{\text{offline}}$, and effect size for the “sentence length in characters” feature. The ranges between two dotted reference lines are ROPEs.

characters”, μ_{online}, μ_{offline}, η_{online}, η_{offline}, $\mu_{\text{online}} - \mu_{\text{offline}}$, $\eta_{\text{online}} - \eta_{\text{offline}}$, and effect size.

In the first two upper panels, posteriors indicate that individuals’ sentences were typically 73.84 characters when typing into a text box in a web browser but were typically 116.88 characters with traditional word processing software. Those writing online tended to use shorter sentences (42.36 characters fewer, $\mu_{\text{offline}} - \mu_{\text{online}}$). Note that the 95% HPD falls far away from the ROPE, indicating a non-negligible difference. Therefore, we conclude that people wrote shorter sentences when writing online.

The lower panels of Figure 3 show another difference: the standard deviation in the online setting was 5.04 characters fewer than that in the offline mode, indicating a relative lack of variability in sentence length when individuals wrote online.

The estimated effect size was -5.44 with a 95% HPD interval between -8.55 and -2.84 (Figure 3 upper-right). A effect size of greater than 2 (in absolute value) counts as “huge” (Sawilowsky, 2009). One way of comprehending the magnitude of an effect size is the following: with the naked eye, one can barely detect a “small” effect (e.g., Cohen’s d=0.2) but would have no difficulty in seeing a “large” one (e.g., Cohen’s d=0.8).

The preceding analysis looked closely at a specific feature. We considered both the raw data and posterior estimates. In the remainder of the paper, for the sake of brevity, only differences between group means ($\mu_{\text{online}} - \mu_{\text{offline}}$), scales ($\eta_{\text{online}} - \eta_{\text{offline}}$), and effect sizes will be reported.

6.2 How Writing Style Varies by Mode

In Figure 4, 12 out of 15 posterior differences of group means ($\mu_{\text{online}} - \mu_{\text{offline}}$) are credibly nonzero, leaving the rest undecided.

The posterior differences between word lengths and sentence lengths are negative. Participants tend to write shorter sentences and use shorter words in the online condition. Relative to offline writings, a positive difference in mean Flesch-Kincaid levels and a negative difference in mean Gunning fog index scores indicate that individuals simplify their writing style when they are entering prose in the web browser. Similar patterns also appear when examining the percentages of function words, Special English, and common words. Individuals tended to use simpler vocabulary in the online condition. For ratios of parts of speech, the adjective to noun and pronoun to noun ratios show credibly positive differences.

Differences in feature standard deviations are shown in Figure 5, where five features indicated nonzero differences. Sentence length, measured in three different ways, varies less in online writing than in offline writing. That is, individuals tend to use a wider range of sentence lengths in offline writing than in online writing. Two readability scores also show less variations in the online mode.

Effect size. The posterior distributions for effect size are shown in Figure 6. Eleven out of 15 features have nonzero effect size. One effect size counts as "medium", five count as large "large", one counts as "very large," and four count as "huge" (using levels defined in Cohen (1988) and Sawilowsky (2009)). Measures regarding word length both manifest "large" effect sizes, and features related to sentence length all have "huge" effect sizes. Likewise, the Flesch-Kincaid reading ease and Gunning's fog index differences are "huge" and "very large." Function words and special English show "large" effects while common words display "medium" effects. This confirms that individuals used simpler words online. The pronoun to noun ratio is the only part-of-speech ratio that shows a credibly nonzero effect (a "large" effect).

7 Mode Effects and Authorship Attribution Accuracy

Another way to understand the magnitude of mode effects is to check if standard authorship attribution techniques have a harder time identifying an individual's writing when presented with the same individual's writing composed in a different mode. That is, we can compare the rate at which an authorship attribution model identifies the correct individual when presented with an unsigned document written offline with the rate at which the model identifies the correct individual when given a document written offline. (The model is trained using writing samples written in the offline mode.) This approach has the virtue of allowing us to answer the question we began with: Does the mode effect make authorship attribution more difficult?

For this experiment, we use two authorship attribution models featured in Brennan et al. (2012). One is extremely basic, making use of nine features and a simple feed-forward neural network. The second model uses a larger feature set ("Writeprints Static") and a support vector machine classifier with a linear kernel.[3] Authorship attribution accuracy is calculated in the following steps:

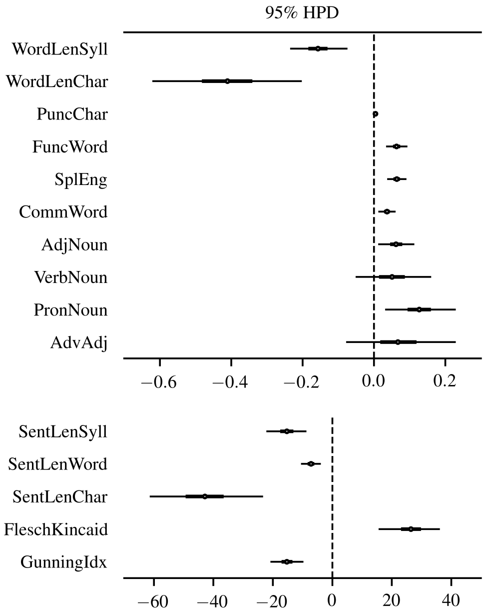
1. Choose a random subset from the 18 authors, starting from two and gradually increasing.

2. Calculate authorship attribution accuracy with five-fold cross-validation using the authorship attribution model (SVM or neural network) with pre-existing writings.

---

[3] We re-engineered the Writeprints Static" and the "Basic-9" feature sets. Our re-implementation of the Writeprints-Static feature set mirrored the original set with three exceptions. First, we applied another widely-used function word list (Koppel et al., 2005) (including 512 words in total) in lieu of the original word list because we could not locate the original list. Second, we used the Penn Treebank tagset (39 non-punctuation tags in NLTK 3.4.5) rather than the original maximum entropy tagset (22 tags). We expect to see very minor differences between the two implementations. Third, we used a linear kernel instead of the polynomial kernel mentioned in Brennan et al. (2012) because the linear kernel performed far better. Indeed, we suspect Brennan et al. (2012) may have used a linear kernel (despite reporting having used a polynomial kernel). A subsequent paper, Overdorf and Greenstadt (2016), which shares an author with Brennan et al. (2012) describes the Brennan et al. (2012) as having used SVM with a linear kernel.

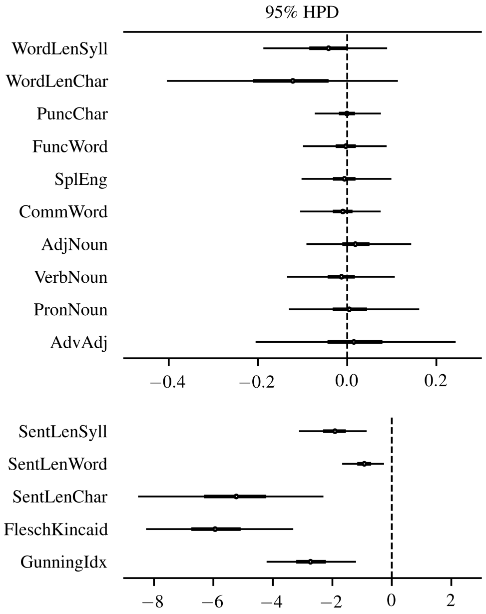
The SVM with a linear kernel used a maximum iteration of 100,000. The neural network used half of the sum of the author count and feature size (which is nine) as the hidden layer count, 100 neurons per layer, and a maximum iteration of 100. All experiments were performed using Scikit-Learn 0.22.1.



Difference of Means ($\mu_{\text{online}} - \mu_{\text{offline}}$)

Feature	Estimate (Posterior Mode)	95% HPD	ROPE	Decision
WordLenSyll	-0.16	(-0.235, -0.074)	(-0.004, 0.004)	Reject Null
WordLenChar	-0.41	(-0.621, -0.202)	(-0.011, 0.011)	Reject Null
PuncChar	0.00	(-0.001, 0.009)	(0.000, 0.000)	Undecided
FuncWord	0.06	(0.034, 0.094)	(-0.002, 0.002)	Reject Null
SplEng	0.07	(0.037, 0.091)	(-0.001, 0.001)	Reject Null
CommWord	0.04	(0.012, 0.061)	(-0.001, 0.001)	Reject Null
AdjNoun	0.06	(0.012, 0.113)	(-0.003, 0.003)	Reject Null
VerbNoun	0.05	(-0.051, 0.161)	(-0.005, 0.005)	Undecided
PronNoun	0.13	(0.031, 0.230)	(-0.005, 0.005)	Reject Null
AdvAdj	0.07	(-0.078, 0.229)	(-0.008, 0.008)	Undecided
SentLenSyll	-15.10	(-22.174, -8.712)	(-0.342, 0.342)	Reject Null
SentLenWord	-7.11	(-10.557, -3.872)	(-0.170, 0.170)	Reject Null
SentLenChar	-42.40	(-61.418, -23.287)	(-0.968, 0.968)	Reject Null
FleschKincaid	26.70	(15.538, 36.181)	(-0.517, 0.517)	Reject Null
GunningIdx	-15.30	(-20.844, -9.693)	(-0.281, 0.281)	Reject Null

Figure 4: 95% HPD, ROPE, and Decisions for Differences of Group Mean ($\mu_{\text{online}} - \mu_{\text{offline}}$)



Difference of Scales ($\eta_{\text{online}} - \eta_{\text{offline}}$)

Feature	Estimate (Posterior Mode)	95% HPD	ROPE	Decision
WordLenSyll	-0.03	(-0.188, 0.090)	(-0.007, 0.007)	Undecided
WordLenChar	-0.11	(-0.404, 0.114)	(-0.013, 0.013)	Undecided
PuncChar	0.00	(-0.073, 0.075)	(-0.004, 0.004)	Undecided
FuncWord	0.00	(-0.099, 0.088)	(-0.005, 0.005)	Undecided
SplEng	-0.01	(-0.102, 0.099)	(-0.005, 0.005)	Undecided
CommWord	-0.01	(-0.105, 0.075)	(-0.005, 0.005)	Undecided
AdjNoun	0.02	(-0.091, 0.144)	(-0.006, 0.006)	Undecided
VerbNoun	-0.01	(-0.135, 0.107)	(-0.006, 0.006)	Undecided
PronNoun	0.01	(-0.131, 0.161)	(-0.008, 0.008)	Undecided
AdvAdj	0.01	(-0.205, 0.243)	(-0.011, 0.011)	Undecided
SentLenSyll	-1.85	(-3.118, -0.848)	(-0.058, 0.058)	Reject Null
SentLenWord	-0.91	(-1.667, -0.260)	(-0.036, 0.036)	Reject Null
SentLenChar	-5.04	(-8.532, -2.301)	(-0.160, 0.160)	Reject Null
FleschKincaid	-6.06	(-8.262, -3.314)	(-0.126, 0.126)	Reject Null
GunningIdx	-2.79	(-4.212, -1.202)	(-0.076, 0.076)	Reject Null

Figure 5: 95% HPD, ROPE, and Decisions for Differences of Group Scale ($\eta_{\text{online}} - \eta_{\text{offline}}$)

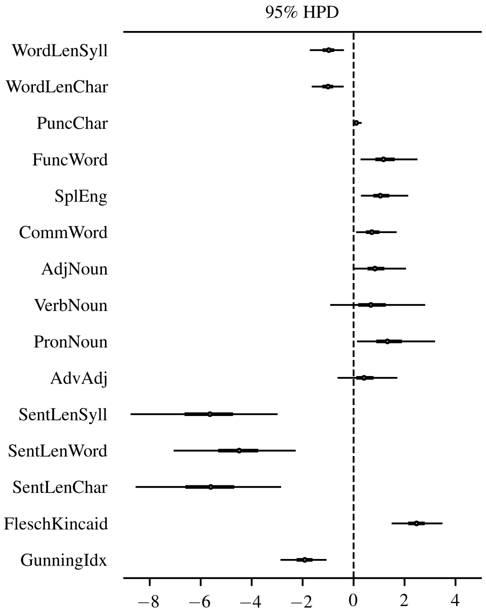
3. Repeat the previous steps 1,000 times using a different author subsets. Calculate the average accuracy over these replications.

Calculating attribution accuracy for the online writing samples follows similar steps. Step 2 differs. The model is trained on pre-existing documents and asked to predict the authorship of a document written online.

Figure 7 shows our results. It is clear that the accuracy suffers when applying offline-writing-trained classifiers to online writings. That is, changes in authorial style are big enough to confuse the classifiers. Writing mode differences make authorship attribution more difficult.

8 Discussion

This study investigated whether individuals' writing style varies by "mode": Does the mode used to



Effect Size

Feature	Estimate (Posterior Mode)	95% HPD	Decision	Most Credible Level
WordLenSyll	-0.90	(-1.715, -0.377)	Reject Null	Large
WordLenChar	-0.95	(-1.640, -0.391)	Reject Null	Large
PuncChar	0.09	(-0.028, 0.323)	Undecided	-
FuncWord	0.99	(0.274, 2.505)	Reject Null	Large
SplEng	0.95	(0.289, 2.146)	Reject Null	Large
CommWord	0.62	(0.103, 1.692)	Reject Null	Medium
AdjNoun	0.72	(0.029, 2.061)	Undecided	-
VerbNoun	0.61	(-0.915, 2.810)	Undecided	-
PronNoun	1.17	(0.133, 3.202)	Reject Null	Large
AdvAdj	0.35	(-0.629, 1.720)	Undecided	-
SentLenSyll	-5.44	(-8.750, -2.978)	Reject Null	Huge
SentLenWord	-4.31	(-7.061, -2.273)	Reject Null	Huge
SentLenChar	-5.44	(-8.552, -2.84)	Reject Null	Huge
FleschKincaid	2.49	(1.492, 3.487)	Reject Null	Huge
GunningIdx	-1.88	(-2.870, -1.059)	Reject Null	Very Large

Figure 6: 95% HPD and the most credible levels for effect sizes ($\frac{\mu_{\text{online}} - \mu_{\text{offline}}}{\sqrt{(\eta^2_{\text{online}} + \eta^2_{\text{offline}})/2}}$). The ROPEs are set with (-0.1, 0.1) associated with Cohen's "small" effect (d=0.2).

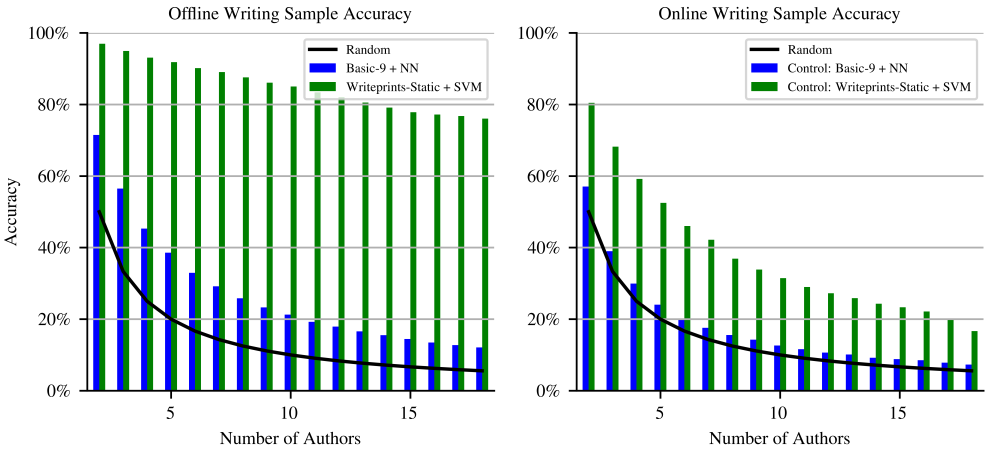


Figure 7: Authorship attribution accuracy by writing mode. The authorship attribution model is trained on pre-existing writing samples which were composed offline.

compose a document (word processor (offline) versus web browser text entry (online)) affect measurements of individuals' writing style? Our findings confirmed the existence of mode effects. In online writing, respondents tend to use shorter sentences, shorter words, more adjectives (relative to nouns) and pronouns (relative to nouns). Sentence lengths exhibit lower variability as well.

Therefore, we suggest authorship attribution researchers should exercise caution when dealing texts written using different modes.

For example, in Brennan et al. (2012), the authors attributed lower accuracy in an authorship attribution task to the fact that writers employed authorship attribution circumvention techniques. Our research suggests that this lower accuracy may be due in part to differences in writing mode. (The circumvention techniques were only used in online writing.)

Systematic differences in writing associated with different modes may complicate a broader range of experiments. Just as researchers appropriately anticipate genre-dependent stylistic differences in individuals' writing (e.g., fiction vs. non-fiction prose), experiments should also anticipate mode-dependent differences.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1814425. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Ahmed Abbasi and Hsinchun Chen. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, 26(2):1–29.

Sadia Afroz, Aylin Caliskan Islam, Ariel Stolerman, Rachel Greenstadt, and Damon McCoy. 2014. Doppelgänger finder: Taking stylometry to the underground. In *2014 IEEE Symposium on Security and Privacy*, pages 212–226. IEEE.

Harald Baayen, Hans van Halteren, Anneke Neijt, and Fiona Tweedie. 2002. An experiment in authorship attribution. In *6th JADT*, volume 1, pages 69–75.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Trans. Inf. Syst. Secur.*, 15(3):12:1–12:22.

Rachel Carpenter and Tracy Alloway. 2019. Computer versus paper-based testing: Are they equivalent when it comes to working memory? *Journal of Psychoeducational Assessment*, 37(3):382–394.

Jacob Cohen. 1988. *Statistical power analysis for the behavioral sciences*. Routledge.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Angela Glover and Graeme Hirst. 1996. Detecting stylistic inconsistencies in collaborative writing. In *The new writing environment*, pages 147–168. Springer.

Neil Graham, Graeme Hirst, and Bhaskara Marthi. 2005. Segmenting documents by stylistic character. *Natural Language Engineering*, 11(4):397–415.

Robert Gunning. 1968. *The technique of clear writing*, revised edition edition. McGraw-Hill, New York, N.Y.

Francis Heylighen and Jean-Marc Dewaele. 1999. Formality of language: definition, measurement and behavioral determinants. *Interner Bericht, Center "Leo Apostel", Vrije Universiteit Brüssel*, 4.

Joseph R Hochstim. 1967. A critical comparison of three strategies of collecting data from households. *Journal of the American statistical Association*, 62(319):976–989.

Patrick Juola. 2006. Authorship attribution. *Foundations and Trends® in Information Retrieval*, 1(3):233–334.

Ryan Kennedy, Scott Clifford, Tyler Burleigh, Philip D Waggoner, Ryan Jewell, and Nicholas JG Winter. 2018. The shape of and solutions to the mturk quality crisis. *Political Science Research and Methods*, pages 1–16.

Mike Kestemont, Michael Tschuggnall, Efstathios Stamatatos, Walter Daelemans, Günther Specht, Benno Stein, and Martin Potthast. 2018. Overview of the author identification task at pan-2018: Cross-domain authorship attribution and style change detection. In *Working Notes Papers of the CLEF 2018 Evaluation Labs. Avignon, France, September 10-14, 2018/Cappellato, Linda [edit.]; et al.*, pages 1–25.

Stanislav Kolenikov and Courtney Kennedy. 2014. Evaluating three approaches to statistically adjust for mode effects. *Journal of survey statistics and methodology*, 2(2):126–158.

Moshe Koppel, Jonathan Schler, and Elisheva Bonchek-Dokow. 2007. Measuring differentiability: Unmasking pseudonymous authors. *Journal of Machine Learning Research*, 8(Jun):1261–1276.

Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Automatically determining an anonymous author's native language. In *International Conference on Intelligence and Security Informatics*, pages 209–217. Springer.

Frauke Kreuter, Stanley Presser, and Roger Tourangeau. 2008. Social desirability bias in cati, ivr, and web surveys: The effects of mode and question sensitivity. *Public opinion quarterly*, 72(5):847–865.

John Kruschke. 2014. *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press.

John K Kruschke. 2018. Rejecting or accepting parameter values in bayesian estimation. *Advances in Methods and Practices in Psychological Science*, 1(2):270–280.

John K Kruschke and Torrin M Liddell. 2018. The bayesian new statistics: Hypothesis testing, estimation, meta-analysis, and power analysis from a bayesian perspective. *Psychonomic Bulletin & Review*, 25(1):178–206.

Heidi V Leeson. 2006. The mode effect: A literature review of human and technological issues in computerized testing. *International Journal of Testing*, 6(1):1–24.

Rebekah Overdorf and Rachel Greenstadt. 2016. Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution. *Proceedings on Privacy Enhancing Technologies*, 2016(3):155–171.

Upendra Sapkota, Thamar Solorio, Manuel Montes, and Steven Bethard. 2016. Domain adaptation for authorship attribution: Improved structural correspondence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2226–2235.

Shlomo S Sawilowsky. 2009. New effect size rules of thumb. *Journal of Modern Applied Statistical Methods*, 8(2):597–599.

Yael Sidi, Maya Shpigelman, Hagar Zalmanov, and Rakefet Ackerman. 2017. Understanding metacognitive inferiority on screen by exposing cues for depth of processing. *Learning and Instruction*, 51:61–73.

Efstathios Stamatatos. 2018. Masking topic-related information to enhance authorship attribution. *Journal of the Association for Information Science and Technology*, 69(3):461–473.

Roger Tourangeau, Lance J Rips, and Kenneth Rasinski. 2000. *The psychology of survey response*. Cambridge University Press.

Voice of America. 2007. *VOA Special English word book: a list of words used in Special English programs on radio, television, and the Internet.* Voice of America, Washington, D.C. OCLC: 761196573.

Shannon Washburn, James Herman, and Randolph Stewart. 2017. Evaluation of performance and perceptions of electronic vs. paper multiple-choice exams. *Advances in physiology education*, 41(4):548–555.

Eva Zangerle, Michael Tschuggnall, Günther Specht, M Potthast, and B Stein. 2019. Overview of the style change detection task at pan 2019. In *CLEF*.

Generative Text Modeling through Short Run Inference

Bo Pang[*]
UCLA
`bopang@ucla.edu`

Erik Nijkamp[*]
UCLA
`enijkamp@ucla.edu`

Tian Han
Stevens Institute of Technology
`than6@stevens.edu`

Ying Nian Wu
UCLA
`ywu@stat.ucla.edu`

Abstract

Latent variable models for text, when trained successfully, accurately model the data distribution and capture global semantic and syntactic features of sentences. The prominent approach to train such models is variational autoencoders (VAE). It is nevertheless challenging to train and often results in a trivial local optimum where the latent variable is ignored and its posterior collapses into the prior, an issue known as *posterior collapse*. Various techniques have been proposed to mitigate this issue. Most of them focus on improving the inference model to yield latent codes of higher quality. The present work proposes a short run dynamics for inference. It is initialized from the prior distribution of the latent variable and then runs a small number (e.g., 20) of Langevin dynamics steps guided by its posterior distribution. The major advantage of our method is that it does not require a separate inference model or assume simple geometry of the posterior distribution, thus rendering an automatic, natural and flexible inference engine. We show that the models trained with short run dynamics more accurately model the data, compared to strong language model and VAE baselines, and exhibit no sign of posterior collapse. Analyses of the latent space show that interpolation in the latent space is able to generate coherent sentences with smooth transition and demonstrate improved classification over strong baselines with latent features from unsupervised pretraining. These results together expose a well-structured latent space of our generative model.

1 Introduction

The state-of-the-art language models (LM) are often modeled with recurrent neural networks (RNN) (Mikolov et al., 2010) or attention-based models (Dong et al., 2019; Vaswani et al., 2017). They are optimized by making a series of next-step predictions, encouraging the models to capture local dependency rather than global semantic features or high-level syntactic properties. A seminal work by Bowman et al. (2016) extends the standard LM to incorporate a continuous latent space which is aimed to explicitly capture global features. They formulate and train the model as a varational autoencoder (VAE) (Kingma and Welling, 2014). Indeed, the model is able to generate coherent and diverse sentences through continuous sampling, and provide smooth interpolation between sentences, uncovering a well-formed latent space.

However, training VAE for text is challenging and often leads to a trivial local optimum, *posterior collapse*. Specifically, the training objective of VAE can be decomposed into a reconstruction term and a KL term that regularizes the distance between the posterior and prior of the latent variable. Due to the autoregressive nature of the decoder, it is able to reconstruct the data well by simply relying on the one-step-ahead groud-truth and evolving model state while completely ignoring the latent codes. The posterior hence collapses into the prior, carrying no information. This is an important open question in this field. As pointed out in Fu et al. (2019), two paths work together to generate sentences in VAE. One path (Path A) is through the latent codes, while the other (Path B) is conditioned on the prior ground-truth or previously generated tokens. The posterior collapse describes an easy solution, that is, relying on Path B and ignoring Path A. Prior efforts made to address this issue by and large are along the two paths. One can control the information available from Path B to force the decoder to employ more Path A information. Bowman et al. (2016) dropout the input words to the decoder and Yang et al. (2017) utilize a dilated CNN to con-

---

[*] Equal contributions.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1156–1165
April 19 - 23, 2021.

trol the size of context from previously generated words. Along Path A, various techniques have been developed to improve the latent code quality. Bowman et al. (2016) anneal the weight of the KL term from a small number to reduce the regularization in the beginning of the training (Anneal-VAE), while Fu et al. (2019) further propose to adopt a cyclical annealing schedule (Cyclical-VAE). He et al. (2019) update the encoder multiple times before one decoder update (Lagging-VAE). Li et al. (2019) initialize the VAE with an autoencoder (AE) and adopt a hinge loss for the KL term such that KL is not driven down below a target rate (FBP-VAE and FB-VAE). These techniques fall under the framework of amortized variational inference. Despite its fast inference, Cremer et al. (2018) observes that an amortization gap, the gap between the log-likelihood and the ELBO, can be large. Thus Kim et al. (2018) proposes semi-amortized variational autoencoders (SA-VAE) in which initial variational parameters are obtained from an encoder as in VAE, and the ELBO is then optimized with respect to the variational parameters to refine them.

An alternative to variational inference is Markov chain Monte Carlo (MCMC) sampling. MCMC posterior sampling may be in the form of Langevin dynamics (Langevin, 1908) or Hamiltonian Monte Carlo (HMC) (Neal, 2011; Chen et al., 2014). Traditional MCMC can be time-consuming as the Markov chains require a long running time, each iteration involving a gradient computation through the decoder.

In this article, we propose to apply a short run inference (SRI) dynamics, such as finite step Langevin dynamics, guided by the posterior distribution of the latent variable as an approximate inference engine. For each training example, we initialize such a short run dynamics from the prior distribution such as Gaussian noise distribution, and run a finite number (e.g., 20) of steps of updates. This amounts to a residual network which transforms the initial noise distribution to an approximate posterior distribution.

One major advantage of the SRI is that it is natural and automatic. Designing and tuning a separate inference model is not a trivial task. In prior work, the inference model requires careful tuning to avoid posterior collapse in VAEs for text modeling. For instance, the inference model needs to be aggressively trained (He et al., 2019), pre-trained with an autoencoder (Li et al., 2019), or refined with gradient descent guided by the ELBO (Kim et al., 2018). In contrast, the short run dynamics guided by the log-posterior of the latent variable can be automatically obtained on modern deep learning platforms. In addition, our method does not assume a closed-form density for the posterior, like a Gaussian with diagonal covariance matrix, and hence are possible to have a good approximate posterior and provide good latent code. Lastly, we optimize the hyper-parameter of the short run dynamics by minimizing the KL divergence between the short-run-dynamics-induced posterior and the true posterior, to further improve the approximate posterior.

Empirically, we show that the model trained with the SRI is able to outperform a standard LSTM language model by employing an LSTM generative model, while exhibiting active utilization of the latent space, improving over models trained with VAE-based approaches. Moreover, we find the learned latent space is smooth, allowing for coherent and smooth interpolation and reconstruction from noisy samples, and captures sufficient global information, enabling enhanced classification accuracy over state-of-the-art baselines.

In summary, the following are contributions of our paper. (1) We propose to use short run inference dynamics to train generative models for sentences without the need for an auxiliary inference network. (2) We demonstrate that the generative model trained with the SRI is able to accurately model the data distribution and make active use of the latent space, exhibiting no sign of posterior collapse. (3) We show that the learned latent space is smooth and captures rich global representations of sentences.

2 Model and learning algorithm

2.1 Generative model

Let x be the observed example, such as a sentence. Let z be the latent variable. We may consider z as forming an interpretation or explanation of x, such as the global semantics and/or high-level syntactic properties of sentences. Consider the following generative model for x,

$$z \sim p(z) \quad x \sim p_\theta(x|z). \tag{1}$$

where $p(z)$ is the prior and $p_\theta(x|z)$ is given by a generative model parameterized with θ. The marginal distribution of x is $p_\theta(x) = \int p_\theta(x,z)dz$. Given x, the inference of z can be based on the posterior distribution $p_\theta(z|x) = p_\theta(x,z)/p_\theta(x)$.

2.2 Learning and inference

Let $p_{\text{data}}(x)$ be the data distribution that generates the example x. The learning of parameters θ of $p_\theta(x)$ can be based on $\min_\theta \text{KL}(p_{\text{data}}(x)\|p_\theta(x))$, where $\text{KL}(p\|q) = \text{E}_p[\log(p(x)/q(x))]$ is the Kullback-Leibler divergence between p and q (or from p to q since $\text{KL}(p\|q)$ is asymmetric). If we observe training examples $\{x_i, i = 1, ..., n\} \sim p_{\text{data}}(x)$, the above minimization can be approximated by maximizing the log-likelihood

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(x_i), \quad (2)$$

which leads to the maximum likelihood estimate (MLE).

The gradient of the log-likelihood, $L'(\theta)$, can be computed according to the following identity:

$$\begin{aligned}\frac{\partial}{\partial\theta}\log p_\theta(x) &= \frac{1}{p_\theta(x)}\frac{\partial}{\partial\theta}p_\theta(x)\\ &= \frac{1}{p_\theta(x)}\int \frac{\partial}{\partial\theta}p_\theta(x,z)dz\\ &= \int \frac{\partial}{\partial\theta}\log p_\theta(x,z)\frac{p_\theta(x,z)}{p_\theta(x)}dz\\ &= \text{E}_{p_\theta(z|x)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x,z)\right]. \quad (3)\end{aligned}$$

While the marginal distribution $p(x) = \int p(x|z)p(z)dz$ is intractable due to the latent variables z being integrated out, the above expectation can be approximated by Monte Carlo average with samples drawn from $p_\theta(z|x)$. Such samples from $p_\theta(z|x)$ can be obtained by MCMC in the form of Langevin dynamics (Langevin, 1908), which iterates

$$z_{k+1} = z_k + s\frac{\partial}{\partial z}\log p_\theta(z_k|x) + \sqrt{2s}\epsilon_k, \quad (4)$$

where $\epsilon_k \sim \mathcal{N}(0, I)$, t denotes the time step of Langevin dynamics, and s is the discretization step size. The gradient term is tractable since $\frac{\partial}{\partial z}\log p_\theta(z_k|x) = \frac{\partial}{\partial z}\log p_\theta(z_k, x)$ and thus does not depend on the intractable $p_\theta(x)$. The Langevin dynamics (4) involves a gradient and a diffusion term. The first term is gradient descent $z'_{k+1} = z'_k + s\frac{\partial}{\partial z}\log p_\theta(z_k|x)$ on $\log p_\theta(z_k|x)$. If $z_k \sim p_\theta(z_k|x)$, then the distribution of z'_k will be shifted towards basins of high log-posterior. We may recover $p_\theta(z_k|x)$ by smoothing with the second term $\sqrt{2s}\epsilon_k$, which amounts to white noise diffusion and induces randomness for sampling from $p_\theta(z_k|x)$.

For small step size s, the marginal distribution of z_k will converge to $p_\theta(z|x)$ as $k \to \infty$ regardless of the initial distribution of z_0 (Cover and Thomas, 2006). More specifically, let $q_k(z)$ be the marginal distribution of z_k of the Langevin dynamics, then $\text{KL}(q_k(z)\|p_\theta(z|x))$ decreases monotonically to 0, that is, by increasing k, we reduce $\text{KL}(q_k(z)\|p_\theta(z|x))$.

Finally, the MLE learning can be accomplished by gradient descent. Each learning iteration updates θ by

$$\theta_{t+1} = \theta_t + \eta_t \frac{1}{n}\sum_{i=1}^{n} \text{E}_{p_{\theta_t}(z_i|x_i)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x_i, z_i) \mid_{\theta=\theta_t}\right], \quad (5)$$

where η_t is the step size or learning rate, and $\text{E}_{p_{\theta_t}(z_i|x_i)}$ can be approximated by Monte Carlo sampling from $p_{\theta_t}(z_i|x_i)$.

2.3 Learning with short run inference dynamics

It is computationally impractical to run long Markov chains from $p_\theta(z|x)$ as the gradient term in (4) requires back-propagation through the model underlying $p_\theta(x|z)$. Earlier work (Han et al., 2017) recruits persistent Markov chains (Tieleman, 2008) $\{(z_i, x_i), i = 1, \ldots, n\}$ such that for each observed example x_i a latent code z_i is updated for a few steps in each learning iteration and the chains are maintained throughout the learning procedure. This method leads to inconsistent sampling procedures while training and evaluating the model, since persistent Markov chains for evaluation data are not available. Moreover, estimation of the log-likelihood has to resort to means such as annealed importance sampling (Neal, 2001).

Instead, we adopt short run MCMC (Nijkamp et al., 2019) in which we approximately sample from the posterior distribution of the latent variable. We thus propose the following short run inference dynamics, with a fixed small number of steps

K (e.g., $K = 20$),

$$z_0 \sim p(z), \tag{6}$$

$$z_{k+1} = z_k + s\frac{\partial}{\partial z}\log p_\theta(z_k|x) + \sqrt{2s}\epsilon_k, \tag{7}$$

where $k = 1, ..., K$ and $p(z)$ is the prior distribution of z. Initializing $z_0 \sim p(z) = \mathcal{N}(0, I)$, we perform K steps of Langevin with step size s.

Finally, the learning procedure updates θ by

$$\begin{aligned}\theta_{t+1} &= \theta_t \qquad\qquad (8)\\ &+ \eta_t \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{q_{s,\theta_t}(z_i|x_i)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x_i, z_i)\mid_{\theta=\theta_t}\right], \qquad (9)\end{aligned}$$

where η_t is the learning rate, $\mathrm{E}_{q_{\theta_t}(z_i|x_i)}$ can be approximated by samples drawn from $q_{\theta_t}(z_i|x_i)$ using (7). Compared to MLE learning algorithm (5), we replace $p_{\theta_t}(z|x)$ by $q_{s,\theta_t}(z|x)$. Moreover, we may update the step size s of (7), which we will elaborate in the following.

2.4 Theoretical understanding

Given θ_t, the updating equation (9) is a one step gradient ascent on

$$Q_s(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{q_{s,\theta_t}(z_i|x_i)}\left[\log p_\theta(x_i, z_i)\right]. \tag{10}$$

Compared to the log-likelihood in MLE learning, $L(\theta) = \frac{1}{n}\sum_{i=1}^{n}\log p_\theta(x)$, we have

$$\begin{aligned}Q_s(\theta) &= L(\theta) + \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{q_{s,\theta_t}(z_i|x_i)}\left[\log p_\theta(z_i|x_i)\right]\\ &= L(\theta) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{KL}(q_{s,\theta_t}(z_i|x_i)\|p_\theta(z_i|x_i))\\ &+ \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{q_{s,\theta_t}(z_i|x_i)}[\log q_{s,\theta_t}(z_i|x_i)]. \qquad (11)\end{aligned}$$

Since the last term has nothing to do with θ, gradient ascent on $Q_s(\theta)$ is equivalent to gradient ascent of

$$\tilde{Q}_s(\theta) = L(\theta) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{KL}(q_{s,\theta_t}(z_i|x_i)\|p_\theta(z_i|x_i)), \tag{12}$$

which is a perturbation or a variational lower bound of log-likelihood $L(\theta)$.

The fixed point of the learning algorithm (9) solves the following estimating equation:

$$\frac{1}{n}\sum_{i=1}^{n} \mathrm{E}_{q_{s,\theta}(z_i|x_i)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x_i, z_i)\right] = 0. \tag{13}$$

If we approximate $\mathrm{E}_{q_{s,\theta_t}(z_i|x_i)}$ by Monte Carlo samples from $q_{s,\theta_t}(z_i|x_i)$, then the learning algorithm becomes Robbins-Monro algorithm for stochastic approximation (Robbins and Monro, 1951), whose convergence to the fixed point follows from regular conditions of Robbins-Monro. The estimating equation (9) is a perturbation of the maximum likelihood estimating equation $\frac{1}{n}\sum_{i=1}^{n}\mathrm{E}_{p_\theta(z_i|x_i)}\left[\frac{\partial}{\partial\theta}\log p_\theta(x_i, z_i)\right] = 0$.

2.5 Optimizing step size

We can optimize the step size s by maximizing $\tilde{Q}_s(\theta)$ defined in equation (12), which is equivalent to minimizing the KL divergence between the short-run-dynamics-induced posterior and the true posterior since the first term $L(\theta)$ does not involve s. $\tilde{Q}_s(\theta)$ involves the entropy of $q_{s,\theta_t}(z_i|x_i)$. We provide the details of its computation in the supplementary materials. The step size optimization can be done by grid search or stochastic gradient descent. In this work, we optimize the step size s with grid search guided by maximizing $\tilde{Q}_s(\theta)$.

2.6 Algorithm

The learning procedure is summarized in Algorithm 1. Note that we only optimize s every T_s iterations, so that computational cost is negligible.

Algorithm 1: Learning with SRI.

```
input  : Learning iterations $T$, step size interval $T_s$, learning rate $\eta$, initial
         weights $\theta_0$, observed examples $\{x_i\}_{i=1}^n$, batch size $m$,
         number of steps $K$, initial step size $s$.
output : Weights $\theta_{T+1}$.
for $t = 0 : T$ do
    1. Draw observed examples $\{x_i\}_{i=1}^m$.
    2. Draw latent vectors $\{z_{i,0} \sim p(z)\}_{i=1}^m$.
    3. Infer $\{z_{i,K}\}_{i=1}^m$ by $K$-steps of dynamics (7) with step size $s$.
    4. Update $\theta$ according to (9).
    5. Every $T_s$ iterations, update $s$.
```

2.7 Log-likelihood computation

Unlike traditional MCMC, short run inference enables the computation of the marginal log-likelihood $\log p(x)$[1],

$$\begin{aligned}\log p_\theta(x) &= \log \int p_\theta(x,z)dz \\ &= \log \int \frac{p_\theta(x|z)p(z)}{q_k(z)} q_k(z)dz \\ &= \log E_{q_k(z)}\left[\frac{p_\theta(x|z)p(z)}{q_k(z)}\right]. \end{aligned} \quad (14)$$

Then,

$$\begin{aligned}& E_{p_{\text{data}}}\left[\log \frac{1}{M}\sum_{i=1}^{M} \frac{p_\theta(x|z_i)p(z_i)}{q_k(z_i|x)}\right] \\ &= E_{p_{\text{data}}}\Bigg[\log \sum_{i=1}^{M} \exp\left[\log p_\theta(x|z_i)\right. \\ &\left. + \log p(z_i) - \log q_k(z_i|x)\right] - \log M\Bigg]. \end{aligned} \quad (15)$$

While most terms in (15) are readily available, $\log q_k(z_i|x)$ requires special treatment. We may rewrite the dynamics (7) in the form of

$$z_0 \sim p(z), \quad z_k = R_k(z_0) \quad (16)$$

where R_k is defined by a k-step Langevin dynamics. Let the distribution of z_k be denoted $q_k(z)$. Then, by change of variable theorem,

$$z_k \sim q_k(z), \quad (17)$$

$$q_k(z) = p(R_k^{-1}(z))|\det(dR_k^{-1}(z)/dz)|. \quad (18)$$

Instead of inverting R_k, we draw $z_0 \sim p(z)$ and compute the log determinant of the Jacobian $dR_k(z_0)/dz_0$. See more details in the supplementary.

3 Related Work

Variational inference. VAE (Kingma and Welling, 2014; Bowman et al., 2016) is a prominent method for learning generative models. Due to the autoregressive nature of the decoder, a naive application of VAE to text data results in posterior collapse. Following work makes extensive efforts to alleviate this issue (Fu et al., 2019; Yang et al., 2017; He et al., 2019; Li et al., 2019; Kim et al., 2018; Pelsmaeker and Aziz, 2020; Dieng and Paisley, 2019). Among them SA-VAE developed by Kim et al. (2018) is mostly related to our work. They propose SA-VAE where initial variational parameters obtained from the inference model are further refined by running a small number of gradient updates (e.g., 20) guided by the ELBO. In our work, instead of relying on a parameteric varational distribution, we run a few gradient updates on the log-posterior of the latent variable with initialization from the prior distribution to draw samples directly. Thus, there is no need to design and tune an extra inference model, which is highly non-trivial considering that posterior collapse occurs easily in VAE training.

Alternating back-propagation. Han et al. (2017) propose to learn generative models for images by maximum likelihood, where the learning algorithm iterates over two steps: (i) inferring the latent variable by sampling from its posterior distribution with Langevin dynamics; (ii) updating the model parameters based on the inferred latent codes. In the training stage, in step (i), the Langevin dynamics is initialized from the latent codes inferred in the last epoch, which is called persistent chain in the literature (Tieleman, 2008). In contrast, the short run dynamics always initializes the gradient descent updates from the prior noise distribution. Data-independent initialization renders the dynamics in training and testing consistent.

Short run MCMC. Nijkamp et al. (2019) introduces short run MCMC as a learned sampling dynamics guided by an energy-based model. It shares the same theoretical underpinning as early work of using stochastic gradient Langevin dynamics to learn mixture of Gaussians and logistic regression for large-scale data (Welling and Teh, 2011). Our short run inference method for learning latent variable models for text is inspired by these works.

4 Experiments

We apply our method to train latent variable models on text datasets. The dimension of the latent variable is 32 in all experiments. The generator is implemented with a one-layer uni-directional LSTM (Hochreiter and Schmidhuber, 1997). The number of hidden units and word embedding size of the LSTM vary among datasets to closely follow the experimental setup in recent work (Fu et al.,

---

[1] Note that its Monte Carlo estimator is biased but the bias is diminishing with a large sample size.

	PPL	Recon	AU	KL
		PTB		
LSTM-LM	100.47	-	-	-
Anneal-VAE	101.40	101.28	0	0.00
Cyclical-VAE	108.97	101.85	5	1.37
Lagging-VAE	99.83	100.26	4	0.93
SA-VAE	100.39	100.97	5	1.86
FBP-VAE	99.62	98.52	3	2.95
FB-VAE	96.35	94.52	32	8.15
Ours	**94.26**	**91.14**	**32**	**10.13**
		SNLI		
LSTM-LM	21.44	-	-	-
Anneal-VAE	21.50	31.66	2	1.42
Cyclical-VAE	21.62	30.89	4	2.36
Lagging-VAE	**21.16**	31.53	5	1.42
SA-VAE	21.49	30.12	5	2.34
FBP-VAE	21.46	31.04	3	2.12
FB-VAE	22.00	23.36	32	8.48
Ours	21.21	**22.24**	**32**	**10.02**
		Yahoo		
LSTM-LM	60.75	-	-	-
Anneal-VAE	61.52	329.10	0	0.00
Cyclical-VAE	66.93	333.80	0	2.83
Lagging-VAE	59.77	322.70	15	5.70
SA-VAE	63.92	327.27	17	7.23
FBP-VAE	62.88	328.13	2	3.06
FB-VAE	59.51	315.31	32	15.02
Ours	**57.05**	**311.23**	**32**	**16.19**

Table 1: Language modeling results on PTB, SNLI, and Yahoo test set.

FB-VAE
a man with a cane is walking down the street .
a man with a cane is walking down the street .
a man in a blue shirt is eating food .
people are eating food .
people walk in a city .
people are outside in a city .
Ours
there is a boy skating down a small street .
there is a child walking in the snow .
the man is riding a horse through the snow .
the man is riding a boat .
the biker is looking at the lake .
the person is looking at a country .

Table 2: Comparison on interpolation. Sentence samples greedily decoded from linear interpolation between samples from the Gaussian prior with FB-VAE (Top) and SRI-trained generative model (Bottom).

2019; Li et al., 2019). The number steps of the short run dynamics is 20 for all experiments [2]. The sample from the short run dynamics is used to predict the initial hidden state of the LSTM. It is also concatenated with the word embeddings and then fed to the LSTM as input at each time step.

The short run inference is more computationally costly than the vanilla VAE and has comparable training cost as some improved versions of VAE. The number of inner steps of SRI (20 steps) is about the same as that of SA-VAE and Lagging-VAE. In training, SRI has faster convergence than SA-VAE and comparable convergence as Lagging-VAE in our experiments. In inference, our sampling-based approach is slower than amortized inference. Our method trades a feasible computational cost for accurate inference whose empirical performance is presented in the following experiments.

4.1 Language Modeling

We evaluate our method on language modeling with the Penn Tree Bank (PTB) (Marcus et al., 1993), Yahoo (Yang et al., 2017), and a down-sampled version of the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015) as preprocessed in (Li et al., 2019). Ideally, a language model with latent variable would be expected to make use of the latent space and accurately model the data distribution. To measure the utilization of the latent space, three quantitatively metrics are often considered in prior work (Bowman et al., 2016; Li et al., 2019; Fu et al., 2019; Kim et al., 2018): reconstruction error (Recon), number of active units (AU), the magnitude of KL. Reconstruction error is the negative log-likelihood of the observed data evaluated under the posterior, $\mathrm{E}_{q(z|x)}\left[-\log p_\theta(z|x)\right]$. A latent dimension is considered active if its distribution changes depending on the observations. Following Burda et al. (2016), a latent dimension is defined to be active if $\mathrm{Cov}_x(\mathrm{E}_{z\sim q(z|x)}[z]) > 10^{-2}$. Perplexity (PPL) based on the marginal log-likelihood of x is adopted to measure how accurately the model captures the data. The marginal log-likelihood is estimated with importance sampling with z sam-

---

[2] $K = 10$ steps led to posterior collapse. We observed a slight improvement in model performance if K was increased from 20 to 40 and no improvement from 40 to 60.

there is a crowd of people in the city .
a man rubbing a dirty face .
a couple was waiting to cross the street in a grocery store .
the little girl is drinking water .
a group of boys are playing in the fountain
five asian teenagers are peforming a dance routine for a volunteer organization .
a white-haired man is in front of a building playing music .
construction workers sit at a african courtyard .
a jewish man wearing white garb , playing a guitar , with a lazy look on him .
a young man in a brown checkered shirt sings down on the floor while playing with the on a hot day .

Table 3: Comparison on the generated sentences. Sentence samples generated from the Gaussian prior by FB-VAE (Top) and SRI-trained generative model (Bottom).

k	1	2	3	4
AE	**26.05**	40.46	52.77	63.07
Anneal-VAE	32.20	32.65	33.12	33.39
Cyclical-VAE	31.83	32.87	33.73	34.38
Lagging-VAE	31.78	31.99	32.21	32.32
SA-VAE	31.63	31.82	32.15	32.46
FBP-VAE	29.93	32.59	34.90	36.77
FB-VAE	27.92	29.12	**30.03**	30.85
Ours	27.12	**28.66**	30.21	**30.46**

Table 4: Noisy reconstruction loss on SNLI. k is the number of word swaps performed on the original sentences.

Number of Labels	0	100	$1k$	$10k$
AE	53.1	78.8	83.7	84.1
Anneal-VAE	56.3	59.2	62.3	65.8
Cyclical-VAE	59.9	78.6	82.7	83.2
Lagging-VAE	63.6	65.8	74.2	80.5
SA-VAE	62.6	69.3	78.8	81.4
FBP-VAE	60.9	74.8	76.9	81.1
FB-VAE	67.5	84.9	89.5	90.6
Ours	**73.3**	**85.8**	**89.6**	**90.8**

Table 5: Accuracy on Yelp of unsupervised and semi-supervised classification as a function of the number of labeled example during training.

ples from trained short run dynamics as importance samples.

Besides the standard LM and the vanilla VAE with KL weight annealing, VAEs with recent state-of-the-art training techniques, Cyclical-VAE (Fu et al., 2019), Lagging-VAE (He et al., 2019), SA-VAE (Kim et al., 2018), FBP-VAE and FB-VAE (Li et al., 2019) are also included for comparison. The results are displayed in Table 1. In terms of PPL, our method outperforms all the baselines on the PTB and Yahoo datasets, while does slightly worse than Lagging-VAE and performs better than other baselines on the SNLI. This indicates the model trained with our method is able to accurately model the data distribution. On the other hand, our method yields the lowest reconstruction error and the highest KL with all latent dimension active on all three datasets, exposing the active use of the latent space. Taken together, these results suggest that the model trained with short run dynamics are balanced on modeling the data and utilizing the latent space.

Figure 1 displays a t-SNE plot of the SRI-induced aggregate posterior $E_{p_{\text{data}}} q(z|x)$ and its marginal density of each dimension. The t-SNE plot demonstrates the SRI-induced aggregate posterior is multi-modal and the marginal densities are uni-modal but clearly deviates from the zero-centered standard Gaussian prior. These visualizations demonstrate that the aggregate posterior in our model is clearly different the isotropic Gaussian prior[3] and thus our

---

[3] Ideally $E_{p_{\text{data}}} q(z|x) = p(z)$ since $\int_x p_{\text{data}}(x)q(z|x) = \int_x p(z)p_\theta(x|z)$. However the generative model might not be able to induce such a model posterior. The mismatch might indicate some form of under-regularization, similar to other approaches for mitigating posterior collapse such as FB-VAE.

model does not show a posterior collapse issue, consistent with our analysis above.

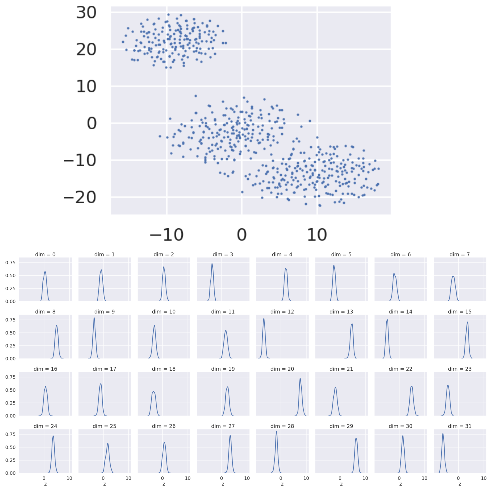


Figure 1: t-SNE plot (upper) and marginal density plot (bottom) of the SRI-induced aggregate posterior on the PTB dataset.

4.2 Latent Space Analysis

The quality of the latent space with SNLI is examined through interpolation, generation, and noisy reconstruction.

4.2.1 Interpolation

Interpolation allows us to appraise the smoothness of the latent space. In particular, two samples z_1 and z_2 are drawn from the prior. We linearly interpolate between them and then decode the interpolated points. FB-VAE (Li et al., 2019) is considered as the SOTA text VAE that mitigates posterior collapse. Due to space limit, we only include this method for comparison in interpolation and generation experiments. Table 2 shows the decoded samples. Although the interpolated sentences by FB-VAE appears smooth, the first two sentences are repetitive. In comparison, the decoded sentences from our model transition more smoothly. While the interpolated sentences from our model are diverse, their syntactic properties and topic information remain consistent in neighborhoods along the path, exposing a smooth latent space.

4.2.2 Generation

We sample from the prior distribution and decode the sentences in a greedy manner. Table 3 displays the samples from our model and FB-VAE. It appears that samples from both models are grammatically correct and semantically meaningful in general. FB-VAE samples nevertheless show more local grammar errors. More generated samples are given in the supplementary.

4.2.3 Noisy reconstruction

Zhao et al. (2018) reasons that a latent variable model's capacity on reconstructing from noisy data reveals the smoothness of the latent space. We impose discrete noise to the data by swapping tokens in a sentence for k times, where $k = 1, 2, 3, 4$ in this experiment. The reconstruction error (negative log-likelihood) under each condition is reported in Table 4. Notice that even the AE yields the lowest reconstruction when the noise is low ($k = 1$), but its performance deteriorates quickly as the noise level increases, implying that the latent space of AE is not smooth. In contrast, other models with regularization on the latent space do not exhibit drastic decline in reconstruction performance with increasing noise level. Furthermore, the model trained with our method demonstrates reconstruction either outperforming other methods or comparable to the best, revealing that the model trained with SRI has a smooth latent space.

4.3 Classification

The latent space of a well-learned latent variable model should capture highly informative features such that data points cluster into meaningful groups in the latent space. We hence further probe the latent space structure by investigating the clustering and classification performance of the SRI-inferred latent codes. Following prior work (Fu et al., 2019; Li et al., 2019), we utilize the Yelp sentiment dataset as preprocessed in Shen et al. (2017). We train a Gaussian mixture for clustering (zero labels) and a SVM with 100, 1000, or 10,000 number of labels. The results are displayed in Table 5. Our method consistently improves over VAE approaches and AE. The improvement is especially clear in the zero-shot setting and small data regime (0 and 100 labels), revealing a well-structured latent space learned by SRI.

5 Conclusion

This work proposes to use short run inference dynamics to infer latent variables in text generative models. SRI dynamics is always initialized from the prior distribution of the latent variable and then performs a small number (e.g., 20) of Langevin dynamics updates guided by the posterior distribution. This simple and automatic inference method induces a good approximate posterior and provides good latent code.

The model trained with SRI accurately models the text data compared to strong language model and generative model baselines and shows no sign of posterior collapse, which is non-trivial to avoid and several remedies have been proposed for in prior art. Moreover, the learned space is smooth and captures rich representations of the sentences.

Acknowledgement

We thank the reviewers for their insightful comments and suggestions. The work is supported by NSF DMS-2015577.

References

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

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Tianqi Chen, Emily B. Fox, and Carlos Guestrin. 2014. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1683–1691.

Thomas M. Cover and Joy A. Thomas. 2006. *Elements of information theory (2. ed.)*. Wiley.

Chris Cremer, Xuechen Li, and David Duvenaud. 2018. Inference suboptimality in variational autoencoders. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 1086–1094.

Adji B Dieng and John Paisley. 2019. Reweighted expectation maximization. *arXiv preprint arXiv:1906.05850*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pretraining for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Çelikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 240–250.

Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. 2017. Alternating back-propagation for generator network. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 1976–1984.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. In *Proceedings of ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yoon Kim, Sam Wiseman, Andrew C Miller, David Sontag, and Alexander M Rush. 2018. Semi-amortized variational autoencoders. In *International Conference on Machine Learning*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Paul Langevin. 1908. *On the theory of Brownian motion*.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3601–3612, Hong Kong, China. Association for Computational Linguistics.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Radford M. Neal. 2001. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139.

Radford M Neal. 2011. MCMC using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2.

Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. 2019. Learning non-convergent non-persistent short-run MCMC toward energy-based model. *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, Canada*.

Tom Pelsmaeker and Wilker Aziz. 2020. Effective estimation of deep generative language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7220–7236, Online. Association for Computational Linguistics.

Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Tijmen Tieleman. 2008. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Max Welling and Yee W Teh. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3881–3890.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *International Conference on Machine Learning*, pages 5897–5906.

Detecting Extraneous Content in Podcasts

Sravana Reddy
Spotify
sravana@spotify.com

Yongze Yu
Spotify
yongzey@spotify.com

Aasish Pappu
Spotify
aasishp@spotify.com

Aswin Sivaraman[*]
Indiana University
asivara@iu.edu

Rezvaneh Rezapour[*]
University of Illinois
at Urbana-Champaign
rezapou2@illinois.edu

Rosie Jones
Spotify
rjones@spotify.com

Abstract

Podcast episodes often contain material extraneous to the main content, such as advertisements, interleaved within the audio and the written descriptions. We present classifiers that leverage both textual and listening patterns in order to detect such content in podcast descriptions and audio transcripts. We demonstrate that our models are effective by evaluating them on the downstream task of podcast summarization and show that we can substantively improve ROUGE scores and reduce the extraneous content generated in the summaries.

1 Introduction

Podcasts are a rich source of data for speech and natural language processing. We consider two types of textual information associated with a podcast episode: the short description written by the podcast creator, and the transcript of its audio, both of which may contain content that is not directly related to the main themes of the podcasts. Such content may come in the form of sponsor advertisements, promotions of other podcasts, or mentions of the speakers' websites and products.

While such content is tightly integrated into the user experience and monetization, it is a source of noise for many natural language processing and information retrieval applications which utilize podcast data. For example (Table 1), an episode of the podcast show *Survival* includes a promotion for an unrelated podcast *Dog Tales* about dogs; a search query for podcasts on dogs should probably not surface the *Survival* episode. Algorithms attempting to connect topics discussed in the podcast to those mentioned in the episode description, such as summarization models, would be confounded by the presence of supplementary material and URLs in the description. Information extraction models looking for entities may mistakenly retrieve sponsor names from advertisements.

... **sit stay and roll over with excitement for par casts endearing series dog tails. Listen to dog tails free on Spotify or wherever you get your podcast.** And now back to the story. Almost immediately after setting sail on June 29th. 1871 Charles Francis Halls Arctic Expedition...
... the focus is on strengthening deficient repertoires, while systematically increasing task demands and difficulty. **For more information, visit www.behaviorbabe.com. — This episode is sponsored by Anchor: The easiest way to make a podcast. https://anchor.fm/app**

Table 1: Portions of a podcast transcript and episode description which contain extraneous material (in bold).

In this paper, we introduce the problem of detecting extraneous content (which we sometimes shorthand as EC) in episode descriptions and audio transcripts. We produce an annotated corpus by taking advantage of podcast listening data, construct models to detect extraneous content, and evaluate our models for accuracy of detection, as well as for the downstream task of summarizing podcast transcripts. We also discuss some of challenges that arise while annotating and classifying extraneous content in this domain.

2 Previous Work

A related, well-studied problem is boilerplate detection on web pages, mainly involving the detection of templates, navigational elements, and advertisements (Kohlschütter et al., 2010). Such models tend to rely on the specific structure of web page boilerplate markup and characteristics. There has also been work on detecting promotional content on Wikipedia (Bhosale et al., 2013).

There are primarily two lines of work in advertisement detection and discovery in multimedia. One computes acoustic features over the entire audio to discriminate between the segments of con-

---

[*] Work done while at Spotify

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1166–1173
April 19 - 23, 2021.

tent and the segments of advertisements (Conejero and Anguera, 2008; Melamed and Kim, 2009; Nguyen et al., 2010). The other fuses multimodal features such as visual cues to segment ad clips from televised and online videos (Lienhart et al., 1997; Duan et al., 2006; Vedula et al., 2017). Our work is closely related to Huang et al. (2018), who analyze consumer engagement on audio advertisements when compared to topical content, as we utilize the user engagement signals to predict extraneous content segments.

3 Datasets and Annotation

To create an annotated dataset, we selected a random set of podcast episodes out of the Spotify Podcast Dataset, a corpus of 105,360 episodes (Clifton et al., 2020). Each episode in the dataset has an automatically generated transcript and a short text description of the episode written by the podcast creators. We annotate both sources, creating ground-truth labels for the extraneous content detection task, using the open source software *doccano* (Nakayama et al., 2018). Annotators were instructed to select spans that correspond to extraneous content, which we defined as ads, social media links, promotions of other podcasts, and show notes that are not directly related to the episode. Respecting sentence boundaries was encouraged, but not required, to allow for cases where extraneous content starts or ends mid-sentence.

3.1 Podcast Episode Descriptions

Annotation of episode descriptions was relatively straightforward. We encountered a few corner cases: for example, an episode may in its entirety be a promotion or an ad; a description that reflects these promotions would be on-topic. In such scenarios, the annotations attempted to be consistent with the annotators' judgments of the main topics of the episode as far as possible. Examples of annotated episode descriptions are shown in Table 6 in the appendix.

3.2 Podcast Episode Transcripts

Each transcript contains thousands of words, of which extraneous content may make up a small portion. This necessitates a way to sample an informative subset of transcript segments for annotation. We observe that if a region is extraneous to the main content, listeners of the podcast episode may fast-forward past this region or abandon listening. To this end, we gather listener data for a subset of our corpus from Spotify, an audio streaming platform. We record the proportion of all listeners who begin streaming an episode retained at each second of the episode's duration. Listening data was aggregated from the date of each episode's publication – with the most recent episode published in February 2020 – through September 2020. For episodes with a substantial number of listeners, there exist distinct local minima or "dips" in the retention curves (Figure 1), which we posit may correspond to regions of extraneous content.

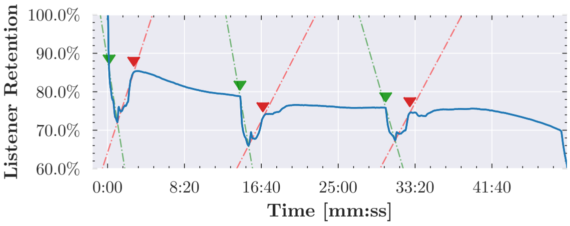


Figure 1: A podcast listener retention curve for a single podcast episode. The dips in the graph suggest potential EC regions. Start- and end-points for each dip are automatically estimated as described in Section 3.2, shown with green and red markers respectively.

To locate the center point of each dip, we first apply `SciPy` peak detection on the negative retention curve (Virtanen et al., 2020). Within ±2 min of each peak, we calculate the slopes of secant lines passing through every point on the curve. The coordinates which maximize the secant slope within this range correspond to the start/end points of the dips. Green and red secant lines are shown in Figure 1 to illustrate this process.

The transcript is then segmented 60 sec before the starting point and 90 sec after the ending point of each dip, loosening the boundaries of the potential EC regions; these segments are then manually annotated. We note that the transcripts contain noisy text which are artifacts from an automatic speech recognition (ASR) system. An additional challenge stems from identifying native advertising, where podcast creators deliberately script product placement into the context of their content (Einstein, 2015; Hutton, 2015). Considering all cases, we attempted to best estimate the boundaries of the scripted content.

Examples of annotated transcript regions are shown in Table 7 in the appendix. The set of manual annotations is used as a *gold* labeled transcript dataset for our models.

Dataset	# of episodes	# of sentences
Episode Descriptions	1730 (61.58% w/ EC)	10410 (43.80% w/ EC)
Transcripts (Gold)	506 (62.25% w/ EC)	12918 (24.01% w/ EC)

Table 2: Our manually annotated datasets show that the majority of podcast episodes do contain extraneous content, making up a sizeable fraction of all sentences.

Of the annotated dips, 38.4 % were found not to contain extraneous content. These largely correspond to episode beginnings, where listeners may skip over the introduction to show, or dynamically inserted ads that are not present in the transcript. The dip boundaries for the rest are relatively accurate against the manual annotations, with an absolute mean error of 16.0 words for the starts, and 35.2 for the ends, motivating the use of the unannotated listener dips as a *silver* training set (described in §4.2).

Sentence-Level Labels Our models for descriptions and transcripts use the sentence as the unit of classification. The annotated data is split into sentences using SpaCy[1]. A sentence is labeled as extraneous if more than 50 % is annotated as such (Table 2).

4 Extraneous Content Detection

4.1 Sentence-Level Classification

We built separate classifiers for detecting extraneous content in descriptions and transcripts. A pretrained BERT (Devlin et al., 2019) cased language model[2] was first fine-tuned on our entire large corpus (of podcast descriptions and transcripts respectively, excluding the test set) to capture the distinctive language use of the podcast domain, and then further fine-tuned for classification on the annotated data to predict whether a sentence is extraneous. We also trained non-neural classifiers (logistic regression and SVMs) with TF-IDF unigram and bigram features (Appendix A.2).

We experimented with single sentence classification in isolation, and with the immediately preceding sentence prepended for context.[3]

Dataset	Model	Single Sentence		Sentence with Context	
		Unigram	Bigram	Unigram	Bigram
Episode Descriptions	LR	0.893	0.818	0.858	0.850
	SVM	0.907	0.907	0.898	0.896
	BERT	0.920		**0.940**	
Transcripts (Gold)	LR	0.651	0.603	0.629	0.638
	SVM	0.675	0.679	0.663	0.670
	BERT	0.710		**0.769**	

Table 3: F1 score of three sentence-level EC classifiers tested across two datasets.

Results Our experiments used a training/test split of 90/10 for annotated transcripts and 80/20 for episode descriptions. The best results come from BERT where the model sees the previous sentence (Table 3). We observed that classification precision and recall are comparable on the descriptions dataset; however, with transcripts, precision tends to be lower (0.690) than recall (0.870). This suggests that the model mistakenly identifies sentences as extraneous, but has fairly good coverage. This seems to be because many sentences in extraneous content in the training transcripts are seemingly innocuous when taken out of context, confounding the model during training.

Dataset	Model	Doc. (%)	Sen. (%)
Episode Descriptions	Sentence-level BERT	86.4	94.9
	BERT + Change Point	**89.6**	95.4
	BERT + BiLSTM-CRF	87.6	95.0
Transcripts (Gold)	Sentence-level BERT	43.1	88.7
	BERT + Smoothing	**49.1**	90.9
Transcripts (Silver)	Sentence-level BERT	60.8	96.6
	BERT + Smoothing	**66.7**	97.0

Table 4: Classification accuracy of sentence-level and document-level extraneous content detection. Documents come from either the dataset of full episode descriptions, or from transcript segments corresponding to dip regions. A document EC-match occurs when *all* sentence labels agree; this is difficult to achieve with longer segments.

4.2 Document-Level Classification

We classify sentences independently, but extraneous content comes as contiguous groups of sentences. We apply non-parametric kernel regression to post-hoc smooth the sequence of individual sentence classification probabilities in the transcripts. We observe that extraneous content within episode

---

[1] https://spacy.io

[2] https://huggingface.co/bert-base-cased

[3] The two sentences are concatenated. In BERT, we include the SEP token to separate them. Where there is no preceding sentence, we use a special `__START__` token.

descriptions often appears as a contiguous block at the end, prompting us to apply a change point detector (Appendix A.3) on the sentence classification probabilities in order to detect the start of the EC block.

We also formulate the problem as sequence tagging at the sentence level, in order to allow the model to learn the label dependencies. For this, we use the BERT pooled sentence embeddings as input to a separate BiLSTM-CRF model. The BiLSTM-CRF improves over sentence-level classification but underperforms the change point detection strategy (Table 4). In future work, we would like to investigate an end-to-end BERT-BiLSTM-CRF model (Dai et al., 2019) or sequential sentence classification models (Cohan et al., 2019).

Expanded Transcript Dataset Since the manually labeled gold set is small, we create a larger silver dataset from 6401 detected listener dips across 4930 episodes by applying the best performing classifier trained on the gold data. To encode information about the dip locations to aid the model, we prepend special tokens 'in-dip' and 'outside-dip' to each sentence depending on whether the sentence lies within a listening dip.

We strip special tokens from the resulting silver set. This data is then used to train a final classifier that can detect extraneous content regions in podcast transcripts *without* listener dips. We also add a negative sample of sentences that are distant (by at least 5 minutes) from dips in the same episodes, with the assumption that these are likely to be topical.

On the same test set as the previous experiments, the document and sentence level performance increases (Table 4), proving the model benefits from the larger, albeit noisy, training set. This model is applied to the corpus in the downstream task described below.

5 Application to Podcast Summarization

We address the problem of automatically generating episode descriptions from podcast transcripts, a task similar to abstractive summarization. Within this problem, we evaluate the downstream effect of removing extraneous sentences from the training and/or test data. Alternatives to removal (such as using the model's predictions as auxiliary inputs in the downstream system) are left for future work.

We experiment with two supervised abstractive summarization models both built using BART (Lewis et al., 2020). The first experiment uses a model pretrained for summarization on the CNN/DailyMail dataset[4] (Hermann et al., 2015). This model (which we refer to as BART-CNN) evaluates the extent to which extraneous text in the transcripts contribute to the presence of extraneous content in the generated descriptions. In the second experiment, we fine-tune BART-CNN on our corpus of podcasts, similar to the work of Zheng et al. (2020), using the episode transcripts as inputs and descriptions as targets. We refer to this model as BART-PODCASTS; with it, we can evaluate the effects of EC as realistic noise which may contaminate the training data of summarization models.

5.1 Experimental Setup

From the original corpus of 105,360 podcast shows, 6401 were used for training and evaluation of the two EC detection models. We filter the remaining episodes for descriptions which are distinctly short or long. The resulting dataset contains 84,451 episodes sorted by episode publication date. This is split into 82,451 episodes for training, 1000 for validation, and 1000 for evaluation.

As a baseline, we manually remove extraneous content from the episode descriptions within the test set, comparing the ROUGE score of the model outputs against the manually-cleaned descriptions as well as against the original descriptions. Additionally, we manually validate whether the generated outputs for 150 random test episodes contain extraneous content.

5.2 Results

Table 5 shows the full ROUGE-L scores of our experiments. We evaluate quality through ROUGE as well as by manually verifying for the presence of extraneous content in the outputs. BART-CNN is an out of the box summarization model, while BART-PODCASTS is the same model that is fine-tuned on our data of transcripts and descriptions. All numbers are reported on the test split of the corpus. The range of these ROUGE scores is comparable to previous podcast summarization work. Removing extraneous content is clearly beneficial for summary quality: while the baseline models have better ROUGE scores against the original (EC-containing) descriptions, the highest scoring models score better against the clean descriptions compared to the originals. With the original data, both

---

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

Model	**EC Removal Method**	ROUGE-L against original descriptions			ROUGE-L against cleaned descriptions			Amount of EC in output (%)
		Rec.	Pre.	F	Rec.	Pre.	F	
Original Descriptions (no model)	n/a	1.000	1.000	1.000	0.996	0.810	0.872	50.0
BART-CNN	n/a	0.310	0.199	0.204	0.340	0.170	0.195	18.8
	M1	0.299	0.200	0.199	0.330	0.172	0.192	11.5
	M2	0.309	0.188	0.195	0.345	0.165	0.191	4.0
BART-Podcasts	n/a	0.377	0.329	0.315	0.349	0.228	0.248	73.2
	M1	0.269	0.309	0.247	0.308	0.280	0.260	8.7
	M3	0.323	0.276	0.261	0.367	0.247	0.263	15.5
	M4	0.330	0.281	0.269	0.376	0.261	0.268	2.0

Table 5: Evaluation of the effect of extraneous content detection on a downstream summarization task. Accuracies are reported in terms of recall, precision, and F1 score. The models are provided with data pre-processed through one of four EC removal strategies: (M1) cleaned test descriptions, (M2) cleaned test transcripts, (M3) cleaned training descriptions, or (M4) cleaned descriptions and transcripts for both train and test.

BART-CNN and BART-PODCASTS tend to generate descriptions that contain extraneous material. Interestingly, BART-PODCASTS, being trained on the unmodified descriptions, produces even *more* extraneous content (73.2 %) than the corresponding original descriptions (50.0 %), often generating ads unrelated to the actual sponsors, and nonexistent URLs. While post-processing only the output summaries with no change to the model inputs is effective at minimizing extraneous content, it does so at the expense of summary quality, since the resulting summaries are significantly shortened. The best overall performance comes from detecting extraneous content on transcripts and descriptions before model training and application.

6 Conclusion

We introduced the problem of detecting extraneous content in podcast descriptions and transcripts, presented models that leverage textual and listener data, and evaluated them on a downstream summarization task.

We consider our models to be baselines for a new problem with several opportunities for future work. Although we used two separate models for the descriptions and transcripts with the view that the language patterns are different, a joint model or shared components may be able to take advantage of some of the common vocabulary. One could leverage the 'boilerplate' nature of some types of extraneous content like ads by detecting repeated sentences and phrases across the corpus. A language model that is robust to noisy speech transcripts (Lin et al., 2019; Chuang et al., 2019) may improve accuracy on podcast transcripts. Given that extraneous content may appear as pre-recorded audio, or with a different speaking pitch and cadence, acoustic features alongside textual ones may be helpful.

7 Acknowledgments

We thank Ann Clifton and Jussi Karlgren for valuable assistance with the summarization model and evaluations, Richard Mitic, Laurence Pascall, Skylar Brannon, Kristie Savage, and Karishma Gulati for collaborating on identifying listening dips from user logs, Iftekhar Tanveer for helpful discussions, and the anonymous EACL reviewers for their suggestions.

References

Shruti Bhosale, Heath Vinicombe, and Raymond Mooney. 2013. Detecting promotional content in Wikipedia. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1851–1857, Seattle, Washington, USA. Association for Computational Linguistics.

Yung-Sung Chuang, Chi-Liang Liu, and Hung-Yi Lee. 2019. Speechbert: Cross-modal pre-trained language model for end-to-end spoken question answering. *arXiv preprint arXiv:1910.11559*.

Ann Clifton, Sravana Reddy, Yongze Yu, Aasish Pappu, Rezvaneh Rezapour, Hamed Bonab, Maria Eskevich, Gareth Jones, Jussi Karlgren, Ben Carterette, and Rosie Jones. 2020. 100,000 podcasts: A spoken English document corpus. In *Proceedings* *of the 28th International Conference on Computational Linguistics*, pages 5903–5917, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Dan Weld. 2019. Pretrained language models for sequential sentence classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3693–3699, Hong Kong, China. Association for Computational Linguistics.

David Conejero and Xavier Anguera. 2008. Tv advertisements detection and clustering based on acoustic information. In *2008 International Conference on Computational Intelligence for Modelling Control & Automation*, pages 452–457. IEEE.

Zhenjin Dai, Xutao Wang, Pin Ni, Yuming Li, Gangmin Li, and Xuming Bai. 2019. Named entity recognition using bert bilstm crf for chinese electronic health records. In *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–5. IEEE.

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

Ling-Yu Duan, Jinqiao Wang, Yantao Zheng, Jesse S Jin, Hanqing Lu, and Changsheng Xu. 2006. Segmentation, categorization, and identification of commercial clips from tv streams using multimodal analysis. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 201–210.

Brandon R. Einstein. 2015. Reading Between the Lines: The Rise of Native Advertising and the FTC's Inability to Regulate It. *Brook. J. Corp. Fin. & Com. L.*, 10:225.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Jason Huang, David Reiley, and Nick Riabov. 2018. Measuring consumer sensitivity to audio advertising: A field experiment on pandora internet radio. *Available at SSRN 3166676.*

Christopher Hutton. 2015. Podcasting embraces native advertising.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450.

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

Rainer Lienhart, Christoph Kuhmunch, and Wolfgang Effelsberg. 1997. On the detection and recognition of television commercials. In *proceedings of IEEE international Conference on Multimedia Computing and Systems*, pages 509–516. IEEE.

H. Lin, T. Lo, and B. Chen. 2019. Enhanced bert-based ranking models for spoken document retrieval. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 601–606.

I Dan Melamed and Yeon-Jun Kim. 2009. Automatic detection of audio advertisements. In *Tenth Annual Conference of the International Speech Communication Association.*

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

MN Nguyen, Qi Tian, and Ping Xue. 2010. Efficient advertisement discovery for audio podcast content using candidate segmentation. *EURASIP Journal on Audio, Speech, and Music Processing*, 2010(1):1–12.

Nikhita Vedula, Wei Sun, Hyunhwan Lee, Harsh Gupta, Mitsunori Ogihara, Joseph Johnson, Gang Ren, and Srinivasan Parthasarathy. 2017. Multimodal content analysis for effective advertisements on youtube. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1123–1128. IEEE.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chujie Zheng, Harry Jiannan Wang, Kunpeng Zhang, and Ling Fan. 2020. A baseline analysis for podcast abstractive summarization.

A Appendix

A.1 Annotation Process and Annotated Examples

Examples of extraneous content regions of episode descriptions are shown in Table 6. and example of annotated listener 'dip' regions in the podcast transcripts in Table 7. The categorization into types is only for illustrative purposes and is not used in our model.

A.2 Model Training and Evaluation Details

We modified the sentence splitter in SpaCy to include `---`, `...` and the three-space string as delimiters for episode descriptions, based on our observations of common patterns. Speech recognition errors/disfluencies and missing punctuation contribute to a small amount of noise in the sentence segmentations of transcripts and descriptions respectively.

For the bag of words models, we used `https://scikit-learn.org/` implementations with the default parameters and no hyper-parameter tuning.

For all Transformer models, we used the Huggingface library (Wolf et al., 2020). For summarization, we set the maximum length of the target descriptions as 250 tokens for training and generation, and the minimum length to 30 tokens. The models were trained for up to 5 epochs, with early stopping based on ROUGE-2 on the validation set. All other hyperparameters were set as the defaults specified in the Huggingface Transformers code.

For evaluation, we use the FILES2ROUGE implementation (`https://github.com/pltrdy/files2rouge`), after tokenizing both the reference and the outputs using the Moses tokenizer (`https://github.com/alvations/sacremoses`).

A.3 Change Point Detection for Episode Descriptions

As shown in Eq. 1, we can find a position $\hat{\tau}$ which maximizes the negative log-likelihood ratio R_τ of H_1 as the existence of the change point versus H_0 as no change point.

$$\hat{\tau} = \arg \max_{1 \leq \tau \leq N} R_\tau \tag{1}$$

We make the assumption that (1) there is only one change point, and (2) extraneous content appears at end the descriptions. The null hypothesis is that there is no changepoint, while the alternative hypothesis assumes that there is a changepoint at the time $t = \tau$. Here is the hypothesis test:

$$H_0 : \theta_1 = \theta_2 = ... = \theta_{N-1} = \theta_N \tag{2}$$

$$H_1 : \theta_1 = ... = \theta_{\tau-1} = \theta_\tau \neq \theta_{\tau+1} = ... = \theta_N \tag{3}$$

Therefore, the likelihood is given by the probability of observation the data $x = x_1,...,x_n$ conditional on H_0. In other words,

$$L(H_0) = p(x|H_0) = \prod_{i=1}^{N} p(x_i|\theta_i) \tag{4}$$

And the the likelihood of the alternative hypothesis is,

$$L(H_0) = p(x|H_1) = \prod_{i=1}^{\tau} p(x_i|\theta_i) \prod_{j=\tau+1}^{N} p(x_j|\theta_j) \tag{5}$$

The log-likelihood ratio R_τ is then,

$$\begin{aligned} R_\tau &= \log \frac{L_{H_0}}{L_{H_1}} \\ &= \sum_{i=1}^{\tau} \log p(x_i|\theta_1) + \sum_{j=\tau+1}^{N} \log p(x_j|\theta_2) \\ &\quad - \sum_{k=1}^{N} \log p(x_k|\theta_0) \end{aligned} \tag{6}$$

TYPE	EXAMPLES
Platform Ads	Transformation Church Pastor Mike Todd brings Marked Part 8- You Are Enough −−− **This episode is sponsored by Anchor: The easiest way to make a podcast. https://anchor.fm/app Support this podcast: https://anchor.fm/m-e8/support**
Social media links, podcast promotions, sponsor ads	Crazy new years news, Dallin fantasizes about a Tesla, and then we do some hilarious Mad Libs... you have to hear these. **Follow the Dashleys for more! https://thedashleysxxxx.com https://facebook.com/thedashleysxxx https://instagram.com/thedashxxxx https://youtube.com/thedasxxxx hellodasxxxx@gmail.com Check out our new Podcast! http://anchor.fm/takingsxxxx Thank you to Skillshare for sponsoring this podcast! Get 2 months premium membership free here: skillshare.com/dashxxxx Thank you to HelloFresh for sponsoring this podcast! Sign up for HelloFresh and get 10 free meals here! URL: hellofresh.com/BIGLITTLELIxxx Promo: BIGLITTLELxxxx — Support this podcast: https://anchor.fm/biglittlexxxx/support**

Table 6: Annotated examples from episode descriptions. (Some of the links are masked for privacy.)

TYPE	SUBTYPE	EXAMPLES
Sponsorships / Ads	Products	... the railway tracks a mere four inches a small but deadly act of sabotage. **Hey friend, I want to tell you about the new $3 Little John from Jimmy John's [...] or with the Jimmy John's app at participating locations taxes and delivery fees extra.** Welcome to today and True Crime a par cast original...
	Platforms	... before we move on to your next topic. **We just want to say thanks to our sponsor Anchor. If you haven't heard about Anchor [...] and it's everything you need to make a podcast in one place download the free anchor app or go to Anchor.** Um to get started, a very somber...
Podcast Promotions	Other Podcasts	... Coming up heading returns to Denmark and Begins the long journey to reclaim his throne. **Darkness tragedy pain these things hide within every beloved institution and most people are none the wiser every week. [...] Search The Dark Side of or visit Park ask.com / Dark Side to listen now.** Now back to the story...
	Social Media	... manipulates our relationships into deadly results **at par cast we are grateful for you our listeners you allow us to do what we love let us know how we're doing reach out on Facebook and Instagram at par cast and Twitter [...] for more information.** It all began with a computer...
Not EC	Episode Introduction	... I'm your host Taylor structure of the live Daily Talk Radio Show [...] It's getting to meet interesting inspiring and exceptional people and getting them to talk about the heaviest things in the world. So if you love real talk with a...
	Miscellaneous Content	... Not much is known about her personal life prior to her murder if she was so average, why was she living with her aunt and uncle I'm not saying there's anything wrong with that [...] I'm sorry to say that it looks as though your husband has an advanced...

Table 7: Annotated examples from the podcast transcripts, corresponding to detected dips in listening.

Randomized Deep Structured Prediction for Discourse-Level Processing

Manuel Widmoser[*1] **Maria Leonor Pacheco**[*2] **Jean Honorio**[2] **Dan Goldwasser**[2]
[1]University of Salzburg, Austria
[2]Purdue University, USA
mwidmoser@cs.uni-salzburg.at,
{pachecog, jhonorio, dgoldwas}@purdue.edu

Abstract

Expressive text encoders such as RNNs and Transformer Networks have been at the center of NLP models in recent work. Most of the effort has focused on sentence-level tasks, capturing the dependencies between words in a single sentence, or pairs of sentences. However, certain tasks, such as argumentation mining, require accounting for longer texts and complicated structural dependencies between them. Deep structured prediction is a general framework to combine the complementary strengths of expressive neural encoders and structured inference for highly structured domains. Nevertheless, when the need arises to go beyond sentences, most work relies on combining the output scores of independently trained classifiers. One of the main reasons for this is that constrained inference comes at a high computational cost. In this paper, we explore the use of randomized inference to alleviate this concern and show that we can efficiently leverage deep structured prediction and expressive neural encoders for a set of tasks involving complicated argumentative structures.

1 Introduction

Many discourse-level NLP tasks require modeling complex interactions between multiple sentences, paragraphs or even documents. For example, analyzing opinions in online conversations (Hasan and Ng, 2013; Sridhar et al., 2015) requires modeling the dependencies between the opinions in individual posts, the disagreement between posts in long conversational threads and the overall view of users, given all their posts.

Learning in these settings is extremely challenging. It requires highly expressive models that can capture the claims made in each document, either by using a rich, manually crafted feature set, or by using neural architectures to learn an expressive representation (Ji and Eisenstein, 2014; Niculae et al., 2017). In addition, reasoning about the interaction between these decisions is often computationally challenging, as it requires incorporating domain-specific constraints into the search procedure, making exact inference intractable. As a result, most current work relies on highly engineered solutions, which are difficult to adapt. Instead of training structured predictors that model the interaction between decisions during training, they combine locally trained classifiers at test time (Stab and Gurevych, 2017).

Our goal in this paper is to study realistic settings, in which discourse-level problems can be learned efficiently when leveraging *deep structured prediction*, a framework for combining rich neural representation with an inference-layer, forcing consistency between them (Zheng et al., 2015). These models were applied successfully to simpler NLP tagging tasks (Lample et al., 2016), in which inference is tractable. However, as shown in a recent argumentation mining work (Niculae et al., 2017), their applicability to more complex learning tasks is not guaranteed.

Randomized inference algorithms have been proposed for structured NLP tasks, such as tagging and dependency parsing, in the context of linear models (Zhang et al., 2014, 2015; Ma et al., 2019). This approach offers an efficient alternative to exact inference. Instead of finding the optimal output state, the algorithm makes greedy updates to a randomly initialized (or locally initialized) output assignment state. Our main contribution is to explore these ideas in the context of deep structured models composed of expressive text encoders, where theoretical guarantees are weak or nonexistent. Moreover, we do this for discourse-level tasks involving a rich set of domain constraints. To do this, we consider two variations of this approach. In the first, the

---

[*] Contributed equally to this work as first authors

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1174–1184
April 19 - 23, 2021.

algorithm samples and traverses only legal states (i.e., consistent with the constraints imposed by domain knowledge). In the second, these restrictions are ignored and only applied at test time. Adapting the sampling procedure to the specific constraints imposed by each domain is difficult, motivating the second approach as a generic alternative.

We focus on two discourse-level tasks, stance prediction in discussion forums, described above, and parsing argumentation structures in essays (Stab and Gurevych, 2017). The latter consists of constructing an argumentation tree that represents the type-of, and relation-between, the arguments made in the essay. Models for both tasks typically employ declarative inference for incorporating domain knowledge. Our experiments are designed to quantify the trade-off between different modeling choices, both in terms of task performance and computational cost. We compare exact ILP models, approximate inference based on the popular AD^3 algorithm (Martins et al., 2015) and the two randomized inference algorithms. Our experiments show that in all cases, deep structured prediction outperforms traditional shallow approaches, structured learning outperforms inference over locally trained models, and generic randomized inference performs competitively to exact inference.

2 Related Work

Using deep structured prediction for NLP has been studied in previous work, typically on traditional sentence-level tasks such as dependency parsing (Chen and Manning, 2014; Weiss et al., 2015), transition systems (Andor et al., 2016), named entity recognition (Lample et al., 2016) and sequence labeling systems (Ma and Hovy, 2016). In most of these cases, inference is tractable. More recently, some efforts have started to look at incorporating deep structured prediction to discourse tasks such as argument mining (Niculae et al., 2017), event and temporal relation extraction (Han et al., 2019) and discourse representation parsing (Liu et al., 2019). In all of these cases, constrained inference is formulated as an integer linear program and solved either using off-the-shelf optimizers or approximation algorithms like AD^3 (Martins et al., 2015).

Randomized approximation has been introduced as an alternative to exact inference. Zhang et al. (2014) suggest a simple randomized greedy inference algorithm and empirically demonstrate its effectiveness for dependency parsing and other traditional NLP tasks (Zhang et al., 2015). The theoretical results in (Honorio and Jaakkola, 2016), based on the probably approximately correct Bayes framework, characterize these findings by providing generalization bounds. More recently, Ma et al. (2019) extended the work of (Zhang et al., 2014, 2015) to structured prediction tasks with large structured outputs by leveraging local classifiers to find good starting solutions and improve the accuracy of search. All of these methods were evaluated on linear structured models.

In this paper, we focus on two tasks: mining argumentative structures in essays and stance prediction in online debates. Stab and Gurevych (2017) approach argumentative essays using an exhaustive set of hand-crafted features, linear local classifiers and ILP at test time. Niculae et al. (2017) jointly learn to score multiple decisions while enforcing domain constraints. They explore structured SVMs and RNNs, using the AD^3 inference algorithm (Martins et al., 2015). On the other hand, there are several works on predicting user stances in online debates. Some approaches model the problem as a text classification task (Somasundaran and Wiebe, 2010; Sun et al., 2018), while other approaches take a collective approach to model user behavior and interactions (Walker et al., 2012; Hasan and Ng, 2013; Sridhar et al., 2015; Li et al., 2018). In the latter case, inference procedures include MaxCut, ILP and probabilistic soft logic (Bach et al., 2017).

3 Modeling

We look at two challenging structured prediction problems that deal with long texts where dependencies span across different paragraphs, documents and authors. To deal with these setups, we define neural factor graphs $G = \{\Psi\}$ where each decision $\psi_i \in \Psi$ is associated with a neural architecture ρ_i and a set of parameters $\boldsymbol{\theta}_i$. In this section, we introduce the tasks in detail.

3.1 Argument Mining

This task aims to identify argumentative structures in essays. Each argumentative structure forms a tree, and there is a forest per document. Nodes correspond to spans of text in the document and they can be labeled either as *claims, major claims* or *premises*. Edges correspond to stances (i.e., support/attack relations between nodes). The spans of texts are given, and we need to label nodes, predict which pairs of nodes are connected by an edge and label the edges. Domain knowledge can be exploited as there are only valid edges between pairs of premises, a premise and a claim, or a claim and a major claim. At the same time, all trees are rooted at major claims. Similarly to previous work, we model second order relationships: **grandparent** ($a \rightarrow b \rightarrow c$) and **co-parent** ($a \rightarrow b \leftarrow c$) (Martins and Almeida, 2014; Niculae et al., 2017).

Figure 1 has a visual representation of the structure. In this problem, each forest defines a factor graph Ψ and G is the collection of all documents. We define a set of five neural architectures corresponding to the five types of decisions that we need to make: $NN = \{\rho_{\text{node}}, \rho_{\text{link}}, \rho_{\text{stance}}, \rho_{\text{grandparent}}, \rho_{\text{coparent}}\}$, each with its own set of parameters $\boldsymbol{\theta} = \{\theta_{\text{node}}, \theta_{\text{link}}, \theta_{\text{stance}}, \theta_{\text{grandparent}}, \theta_{\text{coparent}}\}$. Note that in principle, we can substitute each (ρ_i, θ_i) with any neural architecture. We include details about the architectures in the experimental section.

3.2 Stance Prediction

Given a debate thread on a specific political issue, the task is to predict the stance of each post w.r.t. the issue (e.g., pro-life or pro-choice on abortion) (Walker et al., 2012). Following previous work, we model the problem as a collective classification task and consider all posts in a given thread. To do this, we add the task of predicting stance agreement between consecutive posts. As observed in Figure 1, each thread forms a tree, where users participate and respond to each other's posts. For a thread labeling to be valid, we need to enforce consistency between the node and edge labels.

In this case, each discussion thread defines a factor graph Ψ and G is the collection of threads. We define two neural architectures $NN = \{\rho_{\text{stance}}, \rho_{\text{agreement}}\}$, each with its own set of parameters $\boldsymbol{\theta} = \{\theta_{\text{stance}}, \theta_{\text{agreement}}\}$. As in the previous setup, each (ρ_i, θ_i) can be substituted by any neural architecture, more details are outlined in the experimental section.

4 Learning

We learn a joint neural model that uses inference during training to ensure consistency across all decisions. Let Ψ be a factor graph with potentials $\psi_i \in \Psi$ over all possible structures Y. Let $\boldsymbol{x_i}$ be the input vector to potential ψ_i. Let $\boldsymbol{\theta} = \{\theta^i\}$ be

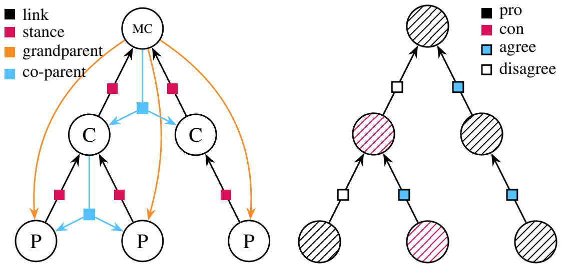


Figure 1: Argument Mining (left), Stance Prediction (right)

a set of parameter vectors associated with a set of neural networks $\boldsymbol{\rho} = \{\rho_i\}$, and $\rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i)$ is the score for potential ψ_i resulting from a forward pass. Here $\boldsymbol{y} \in Y$ corresponds to the gold structure and $\hat{\boldsymbol{y}} \in Y$ to the prediction resulting from the MAP inference procedure:

$$\begin{aligned} \arg\max_{\boldsymbol{y} \in Y} \sum_{\psi_i \in \Psi} \rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i) \\ s.t.\ c(\boldsymbol{x_c}, \boldsymbol{y_c}) \ \ \forall c \in C \end{aligned} \tag{1}$$

Where C is a set of domain-specific constraints defined over the factor graph Ψ, and $\boldsymbol{x_c}, \boldsymbol{y_c}$ indicates inputs and variables relevant to the constraints. In this work, we experiment with different algorithms to obtain or approximate the $\arg\max$, including the randomized procedures outlined in Section 5.

To learn $\boldsymbol{\theta}$, we use the structured hinge loss $L(\boldsymbol{x}, \boldsymbol{y}, \hat{\boldsymbol{y}}; \boldsymbol{\theta})$ defined as:

$$\begin{aligned} \max\Big(0, \max_{\hat{\mathbf{y}} \in Y}\big(\Delta(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \sum_{\psi_i \in \Psi} \rho_i(\boldsymbol{x_i}, \hat{\boldsymbol{y}}_{\boldsymbol{i}}; \theta^i)\big) \\ - \sum_{\psi_i \in \Psi} \rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i)\Big) \end{aligned} \tag{2}$$

Where $\Delta(\boldsymbol{y}, \hat{\boldsymbol{y}})$ is the Hamming loss. To introduce the Hamming loss into the objective, we perform loss augmented inference. The pseudo-code for the structured learning procedure can be observed in Algorithm 1. We implemented our models using DRaiL (Pacheco and Goldwasser, 2020), a declarative deep structured prediction framework built on PyTorch, and extended it to support our randomized inference procedures[1].

5 Randomized Inference

In this section, we describe the randomized inference procedure used for each task. We define the

---

[1] The source code for this paper is available on `https://www.gitlab.com/purdueNlp/DRaiL`

Algorithm 1 Deep Structured Prediction

```
$p \leftarrow 0$
$\text{loss}_{\text{best}} \leftarrow \infty$
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{\text{local}}$
$\boldsymbol{\theta}^{\text{ret}} \leftarrow \boldsymbol{\theta}$
while $p <$ patience do
    for each $\Psi \in G_{\text{train}}$ do
        for each $\psi_i \in \Psi$ do
            $w_i \leftarrow \rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i)$        //forward pass
        $\hat{\boldsymbol{y}} \leftarrow \arg\max_{\boldsymbol{y} \in Y} \sum_{\psi_i \in \Psi} w_i \psi_i(\boldsymbol{x_i}, \boldsymbol{y_i})$
        loss $\leftarrow L(\boldsymbol{x}, \boldsymbol{y}, \hat{\boldsymbol{y}}; \boldsymbol{\theta})$
        backpropagate loss
    $\text{loss}_{\text{dev}} \leftarrow 0$
    for each $\Psi \in G_{\text{dev}}$ do
        for each $\psi_i \in \Psi$ do
            $w_i \leftarrow \rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i)$        //forward pass
        $\hat{\boldsymbol{y}}_{\text{dev}} \leftarrow \arg\max_{\boldsymbol{y} \in Y} \sum_{\psi_i \in \Psi} w_i \psi_i(\boldsymbol{x_i}, \boldsymbol{y_i})$
        $\text{loss}_{\text{dev}} \leftarrow \text{loss}_{\text{dev}} + L(\boldsymbol{x}, \boldsymbol{y}_{\text{dev}}, \hat{\boldsymbol{y}}_{\text{dev}}; \boldsymbol{\theta})$
    if $\text{loss}_{\text{dev}} < \text{loss}_{\text{best}}$ then
        $\text{loss}_{\text{best}} \leftarrow \text{loss}_{\text{dev}}$
        $\boldsymbol{\theta}^{\text{ret}} \leftarrow \boldsymbol{\theta}$
        $p \leftarrow 0$
    else
        $p \leftarrow p + 1$
return $\boldsymbol{\theta}^{\text{ret}}$
```

relevant domain constraints for each case, and explain how we sample solutions that respect them. Finally, we include a discussion about the theoretical bounds for the linear case.

5.1 Argument Mining

For randomized inference on argument mining, we adapt the randomized greedy algorithm proposed by Zhang et al. (2014). Algorithm 2 outlines the overall procedure. We will consider that each paragraph $p \in P$ of an essay contains a single tree. We obtain a local optimum tree $\hat{y}$ by using the hill climbing algorithm, which is further described below. After that, $\hat{y}$ is labeled and added to the forest Y. We iterate over each paragraph (line 4) and subsequently score the forest as:

$$\bar{\mathcal{S}}(Y) = \sum_{\hat{y} \in Y} \mathcal{S}(\hat{y}) = \sum_{\hat{y} \in Y} w + h \, \|y - \hat{y}\|_1 \quad (3)$$

Where $w = \sum_{\psi_i \in \Psi} \rho_i(\boldsymbol{x_i}, \hat{\boldsymbol{y}}_{\boldsymbol{i}}; \theta^i)$ is the sum of the scores of the potentials for the predicted tree $\hat{y}$. We add a weighted Hamming distance term to the scoring function in order to additionally penalize the score the more the tree structure differs from the gold structure. $h \, \|y - \hat{y}\|_1$ gets close to w if the distance is low, and close to zero if it is high.

Algorithm 2 Randomized Inference

```
1:  $\hat{Y} \leftarrow \{\}$
2:  for number of restarts do
3:      $Y \leftarrow \{\}$
4:      for each $p \in P$ do
5:          $\hat{y} \leftarrow$ hill_climbing($p$)
6:          label($\hat{y}$)
7:          $Y \leftarrow Y \cup \{\hat{y}\}$
8:      if $\bar{\mathcal{S}}(Y) > \bar{\mathcal{S}}(\hat{Y})$ then
9:          $\hat{Y} \leftarrow Y$
10: return $\hat{Y}$
```

More specifically, let $\|y - \hat{y}\|_1$ be in $[0, 1]$, e.g., by dividing the number of node and edge differences by the total number of nodes and edges. In its simplest form, h can be assigned to $-w$, and thus $\mathcal{S}(\hat{y})$ would become w if $y = \hat{y}$ or 0 if they differ in every node and edge. Whenever the score of the locally improved forest is better than the forest found so far, Y becomes the new currently best scoring forest $\hat{Y}$. Since hill climbing might get stuck in a local optimum, we repeat line 3-9 for a constant number of restarts.

Algorithm 3 Hill Climbing

```
$\hat{y}_0 \leftarrow$ initialize tree randomly for paragraph $p$
label($\hat{y}_0$)
$\hat{y} \leftarrow \hat{y}_0$
$t \leftarrow 0$
repeat
    $\mathcal{L} \leftarrow$ top-down level node list of $\hat{y}$
    for $i = 1, ..., |\mathcal{L}|$ do
        for $j = i - 1, ..., 0$ do
            $\hat{y}_{t+1} \leftarrow$ connect subtree of $\mathcal{L}_i$ to $\mathcal{L}_j$
            label($\hat{y}_{t+1}$)
            if $\mathcal{S}(\hat{y}_{t+1}) > \mathcal{S}(\hat{y})$ then
                $\hat{y} \leftarrow \hat{y}_{t+1}$
            $t \leftarrow t + 1$
until no improvement in this iteration
return $\hat{y}$
```

Algorithm 3 describes the hill climbing procedure. It initially draws uniformly a tree $\hat{y}_0$ at random. Then the greedy algorithm applies local updates on $\hat{y}_t$ and attempts to achieve a better scoring tree $\hat{y}_{t+1}$. This is done by iterating through a top-down level node list $\mathcal{L}$ of $\hat{y}_t$. Denote i as the current position in the list, then the entire subtree of $\mathcal{L}_i$ is connected to the node $\mathcal{L}_j$, whereas $j = i - 1, i - 2, \ldots, 0$. If the score of $\hat{y}_{t+1}$ is higher than the score of $\hat{y}_t$, the newly generated tree is kept. The algorithm continues until the score can no longer be improved and therefore yields a local optimum tree. Figure 2 depicts how such local

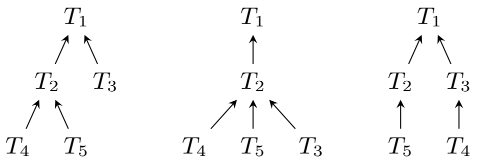


Figure 2: Greedy local update of a tree $\hat{y}_t$ (left) to $\hat{y}_{t+1}$ and $\hat{y}_{t+2}$ without score improvement

updates are performed, $\mathcal{L} = (T_1, T_2, T_3, T_4, T_5)$.

It might be the case that a paragraph contains more than a single tree, therefore, when a tree is initially drawn at random, we introduce an additional *phantom node* which serves as the new root. This modification no longer restricts hill climbing on trees only. Moreover, it allows us having multiple roots and we treat the second layer of the tree like the top layer.

Domain Specific Constraints: For node labeling, we exploit domain knowledge. *Major claims* can only occur in the first or last paragraph, and there has to be at least one major claim in each essay. A root gets labeled as major claim with some fixed probability depending on the paragraph (first or last), holding the condition that there has to exist at least one. Any other root is labeled as a *claim* in each paragraph. Nodes having an edge to a major claim are labeled as claims as well. All remaining nodes are *premises*. An edge can have either the label *support* or *attack* and we draw all edge labels randomly with a probability of 0.9 being a support label. The node and edge labels are determined after each iteration since scoring depends on both, links and labels.

In Section 6, we evaluate our models using randomized inference with and without domain specific constraints. In the latter case, all labels are chosen at random.

5.2 Stance Prediction

A debate thread provides a fixed structure, thus nodes and links are predefined and no improvement of the tree structure needs to be done. However, nodes and edges still need to be labeled and can be improved. Initially, we pick the node labels, which can either be *pro* or *con*. Following the observations made by Ma et al. (2019), we leverage local classifiers and greedily chose the label with the highest score for each node.

Domain Specific Constraints: To respect the dependencies between node and edges labels, we use the following heuristic: If two consecutive nodes u and v have different stances, the edge (u, v) receives a *disagreement* label, if they share the same stance, (u, v) gets an *agreement* label. When author constraints are considered as well, we additionally force stances of posts to be equal when written by the same author.

We attempt to improve node labels by flipping them randomly and subsequently adjust the edge labels. This is done until an iteration no longer improves the overall score. We restart the algorithm for a constant number of times in order to increase the chance of achieving a global optimum. In the experiments, we evaluate our models using randomized inference with and without domain specific constraints. When constraints are not used, a random node is flipped and its adjacent edge adjusted, without enforcing consistency in the whole tree.

The error of the *constrained* randomized algorithms can be bound for the linear case. Let's define the norm of the set of parameter vectors $\boldsymbol{\theta}$ as follows: $\|\boldsymbol{\theta}\| = \sqrt{\sum_{\theta^i \in \boldsymbol{\theta}} \|\theta^i\|^2}$, where $\|\theta^i\|$ is the Euclidean norm of the parameter vector θ^i. Let n be the number of training samples. From Theorem 2 and Claim ii in (Honorio and Jaakkola, 2016), for $\rho_i(\boldsymbol{x_i}, \boldsymbol{y_i}; \theta^i)$ linear in θ^i, the generalization bound (i.e., the difference between the test error and training error) is on the order of $\|\boldsymbol{\theta}\|^2/n + \|\boldsymbol{\theta}\|/\sqrt{n} + \max(1/\log 2, \|\boldsymbol{\theta}\|^2) \log^{3/2} n/\sqrt{n}$. The above generalization bound is decreasing in n, and increasing in $\|\boldsymbol{\theta}\|$, which suggests the use of a large training set, and the penalization of the norm $\|\boldsymbol{\theta}\|$ during learning. In our experiments, we show that in practice we can obtain competitive results by implementing the randomized algorithms for the non-linear case.

6 Experiments

We learn our models using four different inference procedures: (1) **ILP** defines the inference problem as an integer linear program and uses the Gurobi solver[2] to perform exact inference, (2) **AD^3/ILP** translates the ILP program into an AD^3 instance to perform approximate inference, (3) **Rand-C** uses the randomized method with domain constraints, and (4) **Rand** uses the randomized method without domain constraints. Note that we always use exact inference to evaluate on both the development and test sets. For completeness, we add an entry **AD^3**

---

[2] `https://www.gurobi.com/products/gurobi-optimizer`

where we use AD^3 for both training and testing. When using ILP or AD^3, the domain constraints are expressed declaratively.

All experiments were run on a 32 core 3.2Ghz Intel Xeon CPU machine with 128GB RAM and an NVIDIA GeForce GTX 1080 Ti 11GB GDDR5X GPU. We performed an exhaustive search for hyperparameters on the development set. We tuned the learning rate (lr $\in$ {1e-6, 2e-6, 5e-6, 1e-5, 2e-5, 5e-5, 1e-4, 2e-4, 5e-4, 1e-3, 2e-3, 5e-3, 1e-2, 2e-2, 5e-2, 1e-1}), patience (p $\in \{5, 10, 15, 20\}$), and number of restarts (r $\in \{1, 5, 10, 15, 20, 30, 50, 100\}$). The weight decay was fixed at 1e-5 (PyTorch's default). We found that results were stable for local and global models, for different sets of constraints and across inference algorithms.

6.1 Argument Annotated Persuasive Essays

Dataset: We used the UKP dataset (Stab and Gurevych, 2017), consisting of 402 documents, with a total of 6,100 propositions and 3,800 links (17% of pairs). We use the train/dev/test splits used by Niculae et al. (2017), and report macro F1 for components and positive F1 for relations.

Learning and Representation: We did 5 repetitions and reported the average performance. Each repetition used a different seed to initialize the model parameters. For training, we used stochastic gradient descent, a patience of 10, weight decay of 1e-5, and 5 restarts for randomized inference. For local models, we used a learning rate of 0.05 and for structured learning we used a learning rate of 1e-4. Similarly to previous work on deep structured prediction (Han et al., 2019), we obtained better results by performing structured learning over locally trained models, instead of training them from scratch. To represent the component and the essay, we used a BiLSTM over the words, initialized with GloVe embeddings (Pennington et al., 2014), concatenated with a feature vector following Niculae et al. (2017), without the word features. For representing the relation, we use the components, as well as the relation features used in Niculae et al. (2017). For shallow models, we use a bag-of-words representation for the text and concatenate it with the rest of the features into a single feature vector. Both the feature extraction and the neural implementations are available in the repository.

We test two versions of the model: (1) **Base** includes node labeling, link prediction and link labeling, and (2) **Full** adds grandparent and co-parent factors. Domain constraints are introduced in all models.

Model	Inference	Node	Link	Avg	Stance
Local	–	70.7	52.8	61.7 (60.7)	63.4
	L+I	76.5	56.9	66.7 (66.5)	62.5
Base	ILP	83.0	57.6	70.3 (67.2)	68.0
	AD^3/ILP	**83.2**	58.2	**70.7** (67.2)	**68.4**
	AD^3	83.0	57.6	70.3 (67.2)	68.0
	Rand-C	82.8	58.4	70.6 (67.7)	**68.4**
	Rand	82.9	**58.5**	**70.7** (67.7)	68.0
Full	ILP	83.1	61.2	72.2 (65.3)	**69.2**
	AD^3/ILP	83.7	62.0	72.9 (65.3)	68.5
	AD^3	83.5	61.1	72.3 (65.3)	**69.2**
	Rand-C	**83.8**	62.6	73.2 (66.3)	68.4
	Rand	83.4	**63.2**	**73.3** (65.9)	68.4

Table 1: F1 for argument mining using **deep structured prediction**, Avg results using **shallow** models included in parenthesis

We can analyze the results across three dimensions:

Structured Learning: The advantage of leveraging more structural dependencies can be seen in Table 1. The model gets increasingly better as more dependencies are considered, and using global learning outperforms learning local models and using inference just at prediction time (L+I).

Deep vs. Shallow: There is a consistent trend showing that deep structured models are more expressive than their shallow counterparts, as we can see by comparing average results in Table 1. To obtain good results using linear classifiers, Stab and Gurevych (2017) relied on an exhaustive set of features (Table 2). These numbers cannot be replicated by using just word-features and the feature set suggested by Niculae et al. (2017), as our shallow models and their structured SVM results show. In contrast, deep models and word embeddings are able to leverage this information without additional features. In addition, we find that deep models have a shorter overall training time (3.3x faster for the full model). This can be attributed to the compact embedding representation used in deep models, in contrast to the large sparse one-hot vectors used in linear models. Similarly to previous work (Niculae et al., 2017), we find that higher-order factors and strict constraints are more helpful when using deep structured models than in their shallow counterparts.

Model	Node	Link	Avg
Human upper bound	86.8	75.5	81.2
ILP Joint (Stab and Gurevych, 2017)	82.6	58.5	70.6
Struct RNN strict (Niculae et al., 2017)	79.3	50.1	64.7
Struct RNN full (Niculae et al., 2017)	76.9	50.4	63.6
Struct SVM strict (Niculae et al., 2017)	77.3	56.9	67.1
Struct SVM full (Niculae et al., 2017)	77.6	60.1	68.9
Joint PointerNet (Potash et al., 2017)	84.9	60.8	72.9
Kuribayashi et al. 2019	**85.7**	**67.8**	**76.8**
BERT (Devlin et al., 2019)	71.1	50.8	61.0
BERT-doc	79.5	55.8	67.7
BERT-doc + Inf (Base)	79.9	58.1	69.0
BERT-doc + Structured Prediction (Base)	82.1	60.0	71.1
Deep Full ILP	83.1	61.2	72.2
Deep Full Rand-C	**83.8**	62.6	73.2
Deep Full Rand	83.4	**63.2**	**73.3**

Table 2: Previous work on UKP Dataset

Randomized vs. ILP/AD3: When using deep structured prediction, we did not find a statistically significant difference in the performance of the models that were trained with ILP/AD3 vs. the ones that were trained with constrained and non-constrained randomized inference.

We obtain competitive results with respect to previous work that relies on the same underlying embeddings or features, as observed in Table 2. Recently, Kuribayashi et al. (2019) were able to further improve performance by exploiting contextualized embeddings that look at the whole document, instead of embedding the arguments in isolation, and by making a distinction between argumentative markers and argumentative components. We attempted document-level contextualized embeddings using BERT and were not able to replicate their success[3]. Moreover, we found no significant improvement on the structured prediction models when replacing our BiLSTM encoders with either BERT or document-level BERT. We leave the exploration of an effective way to leverage contextualized embeddings for future work. As for stance prediction, Stab and Gurevych (2017) identify stances over the resulting structure and obtain a macro F1 of 68.0. Our full models obtain commensurate results, 69.2, 68.4 for ILP and randomized inference, respectively.

6.2 Debate Stance Prediction

Dataset: We use a subset of the 4FORUMS dataset from the Internet Argument Corpus (Walker et al., 2012), which consist of a total of 418 discussion threads on four political issues, containing 24,658 posts. We use the same splits as (Li et al., 2018). Most previous work reports accuracy. However, given that labels are highly imbalanced, we also report macro F1.

Learning and Representation: We model the problem as a collective classification task by predicting disagreement between consecutive posts in a given thread. We represented posts using a BERT encoder. For disagreement, we just represented pairs of posts without additional information. We do 5-fold cross validation and report the average performance. For training, we used AdamW, weight decay of 1e-5, a patience of 3, and 50 restarts for randomized inference. For local models, we used a learning rate of 5e-5 and for structured models we used a learning rate of 2e-6. For structured learning, we initialize the parameters using the local models. Note that we keep fine-tuning BERT during training.

Model	Infer.	A		E		GM		GC	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
Majority		56.8	28.4	65.9	33.0	66.0	33.0	67.9	34.0
Local		66.0	64.3	65.2	54.3	70.0	61.5	68.2	54.1
Base	L+I	71.0	70.4	63.3	59.2	73.6	69.4	66.8	60.2
	ILP	**72.4**	**71.8**	**64.7**	60.6	**75.1**	72.6	70.5	65.8
	AD^3/ILP	**72.4**	**71.8**	63.2	59.4	**75.1**	72.6	69.7	64.3
	AD^3	**72.4**	**71.8**	64.5	**60.7**	**75.1**	72.6	**71.0**	**66.0**
	Rand-C	71.9	71.5	63.1	60.0	75.0	**73.0**	65.4	60.8
	Rand	71.5	71.1	61.3	58.0	74.3	72.0	64.1	60.2
AC	L+I	83.6	84.6	73.3	69.7	84.8	81.9	68.2	60.9
	ILP	87.5	**88.0**	76.1	73.8	**91.2**	**90.3**	74.2	69.9
	AD^3/ILP	86.2	85.8	**76.7**	**73.9**	90.0	89.0	**74.4**	70.7
	AD^3	85.0	84.8	62.7	60.3	87.4	86.3	72.8	67.9
	Rand-C	**87.8**	87.6	**76.7**	73.7	88.9	87.7	73.4	**71.3**
	Rand	86.6	86.4	73.4	70.9	89.9	88.9	72.7	68.8

Table 3: **Post stance** on 4FORUMS. A: Abortion, E: Evolution, GM: Gay Marriage, GC: Gun Control

We test two versions of the model: (1) **Base** includes consistency between node and edge labels, and (2) **AC** adds author constraints enforcing the same stance for all posts by the same author.

Structured Learning: We can also see that the performance of all structured models outperforms learning local models and using inference just at prediction time (L+I), both for post stance (Table 3) and for disagreement (Table 4).

Randomized vs. ILP/AD3: In the case of stance prediction, there is a significant trend in the performance of the different inference methods. Learning with exact inference generally outperforms the randomized constrained procedure, and the latter outperforms its non-constrained version. The differ-

---

[3] We did not experiment with their extended BoW features, nor AC/AM distinction.

Model	Infer.	A		E		GM		GC	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
Majority		77.8	38.9	66.4	33.2	73.7	36.9	64.3	32.2
Local		76.0	58.1	63.4	56.0	71.3	56.9	67.0	61.4
Base	L+I	70.8	60.3	62.6	58.4	63.3	58.7	61.4	58.9
	ILP	74.2	**62.9**	63.6	**59.6**	**71.5**	61.4	63.8	59.9
	AD^3/ILP	74.2	**62.9**	62.8	58.8	**71.5**	61.4	64.2	61.5
	AD^3	74.2	**62.9**	**64.3**	59.5	**71.5**	61.4	64.2	59.9
	Rand-C	**76.0**	61.6	64.1	57.3	71.2	60.1	64.8	**61.8**
	Rand	**76.0**	60.7	62.7	58.5	70.4	**61.5**	**65.3**	59.4
AC	L+I	83.2	78.7	72.1	70.0	71.0	68.0	68.8	67.0
	ILP	88.0	82.2	**76.5**	**73.6**	**86.1**	**81.5**	75.4	73.6
	AD^3/ILP	86.3	80.5	74.8	72.4	83.9	79.2	72.8	71.5
	AD^3	84.9	76.4	66.8	61.4	84.4	78.4	68.1	65.8
	Rand-C	**88.2**	**82.4**	74.3	71.7	82.5	78.1	**78.5**	**76.3**
	Rand	87.7	81.4	75.0	71.9	84.1	78.7	75.8	74.4

Table 4: **Disagreement** on 4FORUMS. A: Abortion, E: Evolution, GM: Gay Marriage, GC: Gun Control

ence is more pronounced in the case of **AC** models. However, we find that relative to its simplicity, the randomized procedures obtain highly competitive performance.

Model	A	E	GM	GC	Avg
BERT (Devlin et al., 2019)	66.0	65.2	70.0	68.2	67.4
PSL (Sridhar et al., 2015)	77.0	80.3	80.5	69.1	76.7
Struct. Rep. (Li et al., 2018)*	86.5	**82.2**	87.6	**83.1**	**84.9**
Deep AC ILP	87.5	76.1	**91.2**	**74.2**	**82.3**
Deep AC Rand-C	**87.8**	**76.7**	88.9	73.4	81.7
Deep AC Rand	86.6	73.4	89.9	72.7	80.7

Table 5: Previous work on 4FORUMS (Post Acc)
*Note that (Li et al., 2018) use author profile information in their models, whereas we only look at text

Table 5 compares our models to previous work on this dataset. Sridhar et al. (2015) use probabilistic soft logic (PSL) to learn a global assignment for the post labels. They use local classifiers to obtain the input scores to PSL. The main difference between their approach and ours is that we are able to backpropagate the global error back into the classifiers, and we find that it improves performance considerably. Even though we use BERT encoders in our structured procedure, we can see that BERT alone is not able to solve the task. Lastly, we compare to the structured representation learning method of Li et al. (2018) and find that we are able to improve on abortion and gay marriage only. Note that these two are the issues with more data available (8,000 and 7,000 posts respectively). The main difference with their approach and ours is that they push author profile information into the learned representation. We hypothesize that this is key to obtain good performance for gun control, which contains only 4,000 posts.

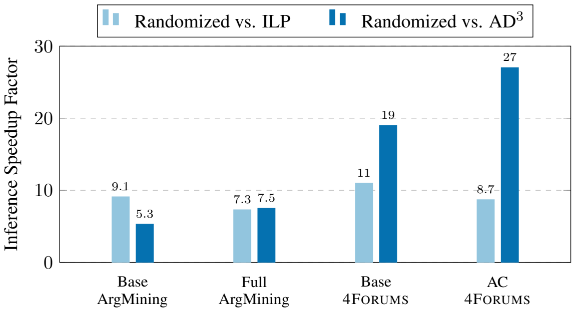


Figure 3: Comparing inference speedup per epoch

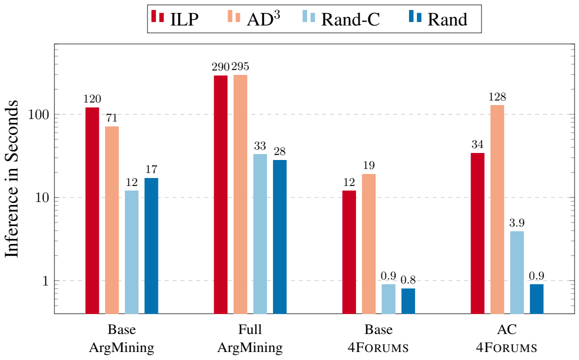
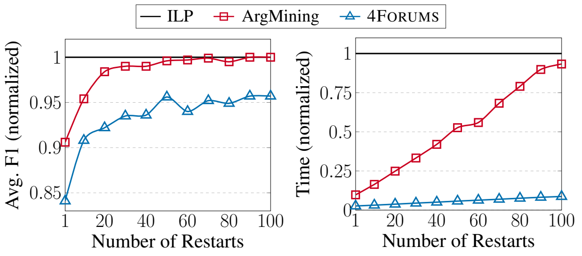


Figure 4: Pure inference runtime in seconds

6.3 Inference Analysis

In our experiments, randomized inference always outperforms ILP and AD^3 in terms of runtime. Figure 3 shows the speedup factor per epoch against ILP and AD^3. In argument mining, AD^3 is faster than ILP, except on our full model, where both perform similarly. We noticed that ILP consumes a lot of time in initialization and encoding. The randomized inference approach is able to predict argumentative structures 9.1x faster than ILP for our base model, and 7.5x faster than AD^3 for our full model. For stance prediction on 4FORUMS, ILP is considerably faster than AD^3, we presume that this is due to the fact that Gurobi is a highly optimized commercial software, and our graphs are small. Randomized inference is 11x faster than ILP on the base model and beats AD^3 by a factor of 27 when author constraints are used.

We also measured pure inference time over five training runs and took the average. Figure 4 shows (in logarithmic scale) plain inference runtime in seconds on the training set for all of our models. We can observe that randomized inference without domain constraints has almost the same performance as the constrained version. Again we find that randomized inference considerably outperforms ILP and AD^3.



(a) (b)

Figure 5: Impact of performance and runtime depending on the number of restarts for randomized inference

Additionally, we evaluated our model at test time by replacing exact inference with randomized inference, incrementally increasing the number of restarts. Figure 5 shows the performance and runtime of the Rand-C algorithm with respect to exact inference $\left(\text{i.e., } \frac{\text{Rand-C}}{\text{ILP}}\right)$. Figure 5a shows that the global optimum is closely approached after just 20 restarts for the argument mining task, as opposed to stance prediction on 4FORUMS, where a higher number of restarts is required. This is in line with our reported results in Sections 6.1 and 6.2. Figure 5b shows that randomized inference is about twice as fast than ILP when using 50 restarts for the Argument Mining task, and it starts to approach the time needed for ILP after 100 restarts. On the other hand, the randomized algorithm on 4FORUMS continues to be an order of magnitude faster even when doing 100 repetitions. Note that as the number of restarts keeps increasing, the randomized procedure will eventually surpass the time needed to perform exact inference.

7 Summary

We studied the effectiveness of randomized inference for deep structured prediction and obtained positive results for two challenging discourse-level tasks. We showed that, in practice, we can train complex structured models, using expressive neural architectures, and get competitive results at a lower computational cost. Moreover, we saw that combining expressive representations and inference is a promising direction for modeling discourse-level structures. Future directions include expanding the discussion to other tasks involving more complex structures, as well as exploring shared representations across different sub-tasks.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2017. Hinge-loss markov random fields and probabilistic soft logic. *J. Mach. Learn. Res.*, 18:109:1–109:67.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 740–750.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 434–444.

Kazi Saidul Hasan and Vincent Ng. 2013. Stance classification of ideological debates: Data, models, features, and constraints. In *Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pages 1348–1356.

Jean Honorio and Tommi S. Jaakkola. 2016. Structured prediction: From gaussian perturbations to linear-time principled algorithms. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence, UAI 2016, June 25-29, 2016, New York City, NY, USA*.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 13–24.

Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. 2019. An empirical study of span representations in argumentation structure parsing. In

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4691–4698, Florence, Italy. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 260–270.

Chang Li, Aldo Porco, and Dan Goldwasser. 2018. Structured representation learning for online debate stance prediction. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3728–3739.

Jiangming Liu, Shay B. Cohen, and Mirella Lapata. 2019. Discourse representation parsing for sentences and documents. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6248–6262.

Chao Ma, F. A. Rezaur Rahman Chowdhury, Aryan Deshwal, Md Rakibul Islam, Janardhan Rao Doppa, and Dan Roth. 2019. Randomized greedy search for structured prediction: Amortized inference and learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5130–5138.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

André F. T. Martins and Mariana S. C. Almeida. 2014. Priberam: A turbo semantic parser with second order features. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 471–476.

André F. T. Martins, Mário A. T. Figueiredo, Pedro M. Q. Aguiar, Noah A. Smith, and Eric P. Xing. 2015. Ad^3: alternating directions dual decomposition for MAP inference in graphical models. *J. Mach. Learn. Res.*, 16:495–545.

Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument mining with structured svms and rnns. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 985–995.

Maria Leonor Pacheco and Dan Goldwasser. 2020. Modeling content and context with deep relational learning. *Computing Research Repository*, arXiv:2010.10453.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, Copenhagen, Denmark. Association for Computational Linguistics.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124, Los Angeles, CA. Association for Computational Linguistics.

Dhanya Sridhar, James R. Foulds, Bert Huang, Lise Getoor, and Marilyn A. Walker. 2015. Joint models of disagreement and stance in online debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 116–125.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Comput. Linguistics*, 43(3):619–659.

Qingying Sun, Zhongqing Wang, Qiaoming Zhu, and Guodong Zhou. 2018. Stance detection with hierarchical attention network. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 2399–2409.

Marilyn A. Walker, Pranav Anand, Rob Abbott, and Ricky Grant. 2012. Stance classification using dialogic properties of persuasion. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pages 592–596.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 323–333.

Yuan Zhang, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2014. Greed is good if randomized: New inference for dependency parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1013–1024.

Yuan Zhang, Chengtao Li, Regina Barzilay, and Kareem Darwish. 2015. Randomized greedy inference for joint segmentation, POS tagging and dependency parsing. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 42–52.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. 2015. Conditional random fields as recurrent neural networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1529–1537.

Automatic Data Acquisition for Event Coreference Resolution

Prafulla Kumar Choubey and Ruihong Huang
Department of Computer Science and Engineering
Texas A&M University
`(prafulla.choubey, huangrh)@tamu.edu`

Abstract

We propose to leverage lexical paraphrases and high precision rules informed by news discourse structure to automatically collect coreferential and non-coreferential event pairs from unlabeled English news articles. We perform both manual validation and empirical evaluation on multiple evaluation datasets with different event domains and text genres to assess the quality of our acquired event pairs. We found that a model trained on our acquired event pairs performs comparably as the supervised model when applied to new data out of the training data domains. Further, augmenting human-annotated data with the acquired event pairs provides empirical performance gains on both in-domain and out-of-domain evaluation datasets.

1 Introduction

Event coreference resolution aims to determine and cluster event mentions that refer to the same real-world event. It is a relatively less studied NLP task despite being crucial for various NLP applications such as topic detection and tracking, question answering, and summarization.

A typical event coreference resolution system relies on scoring similarity between two event mentions in a document followed by clustering. However, event coreference chains are very sparsely distributed and only certain key events are repeated in a document, which makes manually labeling many event coreference relations very time-consuming. Furthermore, event mentions tend to appear in extremely diverse contexts and few are accompanied by a full set of their arguments. The two challenges, the absence of abundant human-annotated event coreference data and the high diversity of contexts containing coreferential event mentions, make it hard to build effective event coreference resolution systems.

We aim to improve the effectiveness of event coreference resolution systems by automatically acquiring coreferential event pairs from many documents requiring minimal supervision. Specifically, coreferential event mentions are associated with discourse function of sentences in a news document (Choubey et al., 2020)[1]. We propose to use them to identify sentence pairs that are likely to contain coreferential event mentions as well as sentence pairs that are likely to contain non-coreferential event pairs. Consider the two example sentence pairs below, each pair having an event pair with synonymous trigger words.

(1): [People living in absolute poverty in rural areas of the eight regions and provinces **reduced** to 14.52 million from 30.76 million over the last decade.] [Yang admitted , however , that ethnic minority regions still lagged far behind the developed eastern regions and the government still faced serious challenges to **reduce** poverty.]

(2): [At least 30,000 war-displaced people camped in Angola's central province of Kwanza-sul are being **resettled** in productive areas, the official news agency angop reported here on Friday.] [The **resettlement** is being carried out jointly by the local municipal authorities of Seles, located in southern Kwanza-sul, and the charity organization German Agro Action, the news agency said.]

In example (1), the first sentence describes a historical event about the reduction in poverty during the last decade, while the second sentence projects the challenges of further reducing poverty in the coming years. Here, the two *reduce* events are non-overlapping in the temporal space and are non-coreferential. On the contrary, in example (2), both mentions for the event *resettle* refer to the same real-world event and can be so ascertained by knowing that both sentences describe the same main

---

[1] The discourse roles are roughly based on the Van Dijk's theory of news discourse (Teun A, 1986). It assigns discourse function to sentences in a news article, where the function is characterized by the operative role of sentence's content in describing the main event, context informing events, and other historical or future projected events

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1185–1196
April 19 - 23, 2021.

event in a news article. In general, we can recognize pairs of sentences in news articles that are likely to contain coreferential or non-coreferential event mention pairs by knowing the sentence's discourse function following Van Dijk's theory.

To ascertain our hypothesis, we first use the discourse profiling system and dataset introduced by Choubey et al. (2020) to identify the discourse role for each sentence in a news article. Then, we use multiple rules to capture the distributional correlation between event coreference chains and discourse roles of sentences and collect a diverse set of 9,210 coreferential and 232,135 non-coreferential event pairs[2]. To assess the reliability of the proposed data augmentation strategy, we perform manual validation on subsets of both coreferential and non-coreferential event pairs. Then, we train event coreference resolution systems using the acquired data alone or using the acquired data to augment a human-annotated training dataset.

We evaluate trained systems on two datasets, the news portion[3] of the widely used benchmark evaluation corpus KBP 2017 as well as the news portion[4] of the Richer Event Description (RED) corpus (O'Gorman et al., 2016). Unlike the KBP corpora that only consider eight event types for event coreference annotations, the RED corpus comprehensively annotates all the event types that appear in a document, and is arguably the only comprehensively annotated corpus of event coreference relations. Assuming the automatically acquired event coreference data is not available, we also train a supervised event coreference resolution system using the KBP 2015 corpus[5]. On the KBP 2017 corpus, the event coreference resolution system trained on the acquired data performs slightly worse than the system trained using the KBP 2015 corpus, the human-annotated in-domain training data. But, on the RED corpus, both the systems trained on either the annotated KBP 2015 corpus or the acquired data obtain roughly the same evaluation results. Further, the system trained on combined annotated KBP 2015 and automatically acquired data yields the best results on both the KBP 2017 dataset and the RED dataset.

Lastly, we evaluate all the trained systems on a different text genre, discussion forum articles from the KBP 2017 corpus, and found that all the systems obtain comparable results. Overall, the performance gain of all the trained systems on discussion forum documents is marginal compared to a simple trigger word match baseline. Thus, increasing training data size does not improve the performance of an event coreference resolution system on a new text genre. We suspect that, for generalization across different text genres, we may require specialized learning algorithms, e.g., text style adaptation, which is not in the scope of this work.

2 Related Work

The existing literature on supervised event coreference resolution primarily focuses on designing pairwise classifier based on the surface linguistic features such as lexical features comprising of lemma and part-of-speech tag similarity of event words (Bejan and Harabagiu, 2010; Lee et al., 2012; Liu et al., 2014; Yang et al., 2015; Lu et al., 2016; Cremisini and Finlayson, 2020), argument overlap (Chen et al., 2009; McConky et al., 2012; Sangeetha and Arock, 2012; Bejan and Harabagiu, 2014; Yang et al., 2015; Lu et al., 2016; Choubey and Huang, 2017), semantic similarity based on lexical resources such as wordnet (Bejan and Harabagiu, 2010; Liu et al., 2014; Yu et al., 2016) and word embeddings (Yang et al., 2015; Choubey and Huang, 2017; Kenyon-Dean et al., 2018; Barhom et al., 2019; Zuo et al., 2019; Pandian et al., 2020; Sahlani et al., 2020; Lu et al., 2020), and discourse features such as token and sentence distance (Liu et al., 2014; Cybulska and Vossen, 2015). The resulting classifier is used to cluster event mentions. The commonly used strategies include *agglomerative clustering* that selects the antecedent closest in mention distance that is classified as coreferent or the antecedent with the highest coreference likelihood (Chen et al., 2009; Chen and Ng, 2014), hierarchical bayesian (Yang et al., 2015) or spectral clustering algorithms (Chen and Ji, 2009). In this work, we use the pre-trained BERT model to extract both event and context features and use agglomerative clustering to form event coreference chains.

Supervised models suffer from a lack of human-

---

[2] The acquired coreferential and non-coreferential event pairs can be found at `https://github.com/prafulla77/Event-Coref-EACL-2021`

[3] All the KBP corpora include news articles as well as documents from discussion forums.

[4] In addition to news articles, the RED corpus contains several other types of documents, including news summaries, discussion forum posts, and web posts.

[5] We only use the news articles from KBP 2015 to train the supervised system.

annotated event coreference data. To address the annotation scarcity problem, Peng et al. (2016) proposed to learn structured event representations on large amounts of text and use the similarity score between two event representations to form event coreference chains. Their model uses a small human-annotated event coreference dataset to find the appropriate similarity score threshold for linking two events. Unsupervised models based on probabilistic generative modeling have also been successfully used for event coreference resolution (Bejan and Harabagiu, 2010; Chen and Ng, 2015). However, both semi-supervised and unsupervised approaches have been found empirically lagging behind the supervised models (Lu and Ng, 2018).

The closest to our work are weakly-supervised and self-training methods that have been shown useful for many information extraction and classification tasks (Riloff, 1996; Riloff and Wiebe, 2003; Xie et al., 2019). But, to the best of our knowledge, we are the first to explore discourse-aware strategies to automatically label event coreference relations and use them exclusively or use them to augment existing human-annotated data for training event coreference resolution systems.

3 Event Coreference Data Acquisition

To acquire coreferential event-pairs without direct supervision, we first collect event trigger words along with their potential set of coreferential event mentions using The Paraphrase Database (PPDB 2.0) (Ganitkevitch et al., 2013; Pavlick et al., 2015)[6]. Then, we use high precision rules informed by the functional news discourse structures (Teun A, 1986; Choubey et al., 2020) to identify seed coreferential and non-coreferential event pairs followed by a single bootstrapping iteration to collect additional non-coreferential event pairs.

3.1 Identifying Coreferential Event Trigger Candidates using The PPDB Database

We collect lexically diverse candidate coreferential event pairs using the paraphrases from PPDB-2.0-s-lexical (Pavlick et al., 2015) database. The corpus[7] contains 213,716 highest scoring lexical paraphrase pairs, each annotated with one of the equivalence, forward or reverse entailment, and contradiction relation classes. First, we extract all the *verb* paraphrase pairs as the potential event trigger words. While event mentions can take other part of speech types, we limit our paraphrase pairs to verbs to ensure high precision among the collected event trigger words. Additionally, many of the verb paraphrase pairs include nominalization (e.g., investing and investment), which adds to the syntactic diversity in the event pairs without compromising their quality. Then, among all verb paraphrase pairs, we filter out only three relation classes, namely *equivalence*, *reverse entailment* and *forward entailment*, as the potential coreferential event pairs. The forward and reverse entailment relations characterize hyponym and hypernym relations, which are not semantically equivalent but can often be coreferential and thus, add diversity to the pairs. Finally, we manually remove noisy event trigger words and cluster the remaining event pairs through pivoting, based on a common event trigger word shared between two paraphrase pairs[8]. Overall, we obtain 1023 clusters with an average of 3.375 event trigger words per cluster.

3.2 Post-Filtering Paraphrase-based Event Pairs using Functional News Discourse Structure

To generate the news discourse structure proposed by Van Dijk (Teun A, 1986; Van Dijk, 1988a,b) and specify the discourse role of a sentence with respect to events in the document, we use the discourse profiling system proposed by Choubey et al. (2020). Note that the above discourse structure is functional (Webber and Joshi, 2012) and does not specify relations between two discourse units. Instead, it classifies each sentence in a document into one of the eight content types. Each content type describes the specific role of a sentence in describing the main event, context informing events, and other historical or future projected events.

The eight content types include *main event* (M1) sentences that describe the most newsworthy event of a news article. Sentences describing events that happen recently and act as triggers for the main event and events that are triggered by the main event constitute the *previous event* (C1) and *consequence* (M2) sentences respectively. The remaining context-informing events and states with temporal co-occurrence with the main event are

---

[6] A contemporary work by Meged et al. (2020) has also studied the potential correlation between coreferential event trigger words and predicate paraphrases.

[7] `http://nlpgrid.seas.upenn.edu/PPDB/eng/ppdb-2.0-tldr.gz`

[8] The processed event clusters are available at `https://git.io/JtnMf`

covered in *current context* (C2) sentences. In addition to the above four content types, a news article may contain sentences describing lesser relevant events such as *historical events* (D1) that temporally precedes the main event by months and years, *anecdotal events* (D2) that are unverifiable personal account of incidents, *evaluation* (D3) containing reactions from immediate participants, experts or known personalities and *expectation* (D4) that projects the possible consequences of the main event.

Among the eight content types, events described in *main event* sentences are central to the main news topic. They routinely appear in headline and sentences of other content types and consequently are more likely to form event coreference chains. On the contrary, events in the *historical event* content type are restricted to describing certain historical background and might only be mentioned once in the document. Additionally, events mentioned in *previous event* sentences tend to happen before those in *main event* and *consequence* sentences, and are unlikely to be coreferential with the events from the later two content types. Overall, content types provide cues for determining whether the events from a certain sentence possess coreferential event mentions and we leverage them to locate both coreferential and non-coreferential event pairs in a news article. Our event coreference data acquisition method works in two phases.

Rule-based Filtering to extract Coreferential and Non-coreferential Event Pairs: In the first phase, we extract both coreferential and non-coreferential event mention pairs based on their respective rules. Specifically, two event mentions from the headline or main event sentences with synonymous event trigger words are identified as coreferential event pairs. Considering that coreferential event mentions are very sparsely distributed, simple trigger-word matching is extremely noisy and damaging when used to train an event coreference classifier. However, narrowing coreferential event mention pairs to synonymous event trigger words from main event sentences or headline significantly eliminates false coreferential event pairs. To get non-coreferential event pairs, we require both trigger words to be non-synonymous and belong to either the same sentence or two sentences of different non-main content types. Further, considering that events in historical event sentences tend to precede the main event by months and years, we identify non-synonymous event pairs with one mention in a historical event sentence and another mention in a main event sentence as non-coreferential. The latter rule allows us to also acquire non-coreferential event pairs with one event from main event sentences, adding to the overall diversity of the acquired dataset.

Distilling Non-coreferential Event Pairs with Synonymous Trigger Words: All the non-coreferential event pairs acquired in phase one have non-synonymous trigger words. However, we know that many of the synonymous words are non-coreferential. Therefore, to further diversify the acquired event coreference data, we use the second-phase bootstrapping to extract non-coreferential pairs with synonymous trigger words. We once again leverage the temporal separation between historical and other content types. We first identify synonymous event pairs that have one mention in a historical sentence and another mention in any non-historical sentence as candidate non-coreferential pairs. Then, we use an event coreference classifier trained on the dataset extracted in phase one to filter out high scoring non-coreferential event pairs (likelihood ≥ 0.9) from the candidate pairs.

3.3 Statistics of Acquired Coreference Data

We use Xinhua news articles[9] from the English Gigaword (Napoles et al., 2012) corpus to acquire coreferential and non-coreferential event pairs using the proposed methodology. We limit the number of coreferential and non-coreferential event pairs for each trigger word to 20 and 200, respectively, to ensure diversity and reduce repetitions of common event trigger words. We compare our acquired event pairs with the KBP 2015 corpus, which has 179 news documents annotated with eight event types and 38 event subtypes. It is the most widely used corpus for training a within-document event coreference resolution system. Table 1 shows the number of event pairs obtained in the first and second phases of our data acquisition strategy and the human-annotated KBP 2015 corpus. Overall, the total number of extracted coreferential event pairs is more than twice the number of pairs in news documents from the KBP 2015 corpus. Note that we can increase the number of acquired pairs by expanding the synonymous event

---

[9] The discourse profiling system (Choubey et al., 2020) obtains the best performance on Xinhua news articles compared to NYT and Reuters

Data	# Coref	# Non-Coref
Rule-based (Phase I)	9210	226776
Distillation (Phase II)	0	5359
KBP 2015	4401	106383

Table 1: Number of coreferential and non-coreferential events pairs acquired through the proposed methodology and the human annotated KBP 2015 corpus.

Row	Data	Prec.	80% CI
1	Synonyms: Coref	49.0	45.3-52.6
2	Synonyms: Non-Coref	51.0	47.3-54.6
3	Phase I: Coref	83.0	80.3-85.6
4	Phase I: Non-Coref	99.3	98.6-100
5	Phase II: Non-Coref	93.0	90.0-96.0

Table 2: Precision (Prec.) and bootstrap 80% confidence interval (80% CI) score of precision for acquired event pairs based on human evaluation.

trigger word list or the unlabeled news article collection.

3.4 Manual Evaluation of Acquired Event Pairs

We randomly selected 300 event pairs from each of the coreferential and non-coreferential samples extracted in the first phase, 100 event pairs from non-coreferential samples distilled in the second phase, and 300 event pairs having synonymous event trigger words to evaluate the proposed data acquisition methodology. Then, we asked a human annotator to validate all the 1000 samples manually.

Table 2 shows the precision and bootstrapped 80% confidence interval of precision for event pairs from each category. Rows 1 and 2 show that only 49% of synonymous event pairs are coreferential while the remaining are non-coreferential. By comparing rows 1 and 3, we can see that limiting coreferential event pairs to the synonymous event trigger words from the headline and main event sentences improves the precision from 49% to 83%. As shown in rows 4 and 5, our rules achieve high precision in identifying non-coreferential event pairs as well, achieving 99.3% for event pairs with non-synonymous trigger words acquired in the first phase and even 93% for event pairs with synonymous trigger words acquired in the second phase. Note that the high precision of non-coreferential event pair identification in both phases is partly due to the distributional sparsity of event coreference chains.

4 Event Coreference Resolution System

We design a neural network-based mention-pair classifier for event coreference resolution. We represent each event pair using 50 context words to the left and right of the first and second event trigger words respectively, and with the maximum of 200 words in between the two event words[10].

Given the event context $(w_1, ., e_1, ., e_2, ., w_n)$, we first transform the context words sequence to word embeddings sequence $(b_{w1}, ., b_{e1}, ., b_{e2}, ., b_{wn})$ using the pre-trained Bert-Large-uncased model (Devlin et al., 2019). Then, we model the semantic associations between two event mentions by measuring similarity between their event embeddings (b_{e1}, b_{e2}) through element-wise product and difference. Further, we obtain context embedding (C) through $maxpool$ operation over the word embeddings sequence to model contextual cues. While the context provides important cues for identifying coreferential event mentions, it may not always be relevant for resolving coreference links. For instance, many event trigger word pairs such as (*"injuries"*, *"recommended"*) are extremely unlikely to exhibit coreferential relations irrespective of their context. Therefore, we use the similarity between event embeddings to control the context input and use them only in the scenarios where event trigger words are likely to possess coreferential link. To achieve so, we apply linear neural layer over element-wise product and differences of two event mention embeddings followed by the $sigmoid$ activation, and multiply them with context embedding C. Finally, we concatenate the resulting set of embeddings and then use a three-layer feed-forward neural network classifier to score the coreference likelihood. The exact formulation of the coreference classifier is described in Eq. 1.

$$\begin{aligned}
(b_{w1}.b_{e1}.b_{e2}.b_{wn}) &= BERT[(w_1.e_1.e_2.w_n)] \in R^{n\times1024} \\
C &= maxpool(b_{w1}, ., b_{e1}, ., b_{e2}, ., b_{wn}) \in R^{1024} \\
s_1 &= sigmoid(W_1^s(b_{w1} \odot b_{w2}) + b_1^s) \in R^{1024} \\
s_2 &= sigmoid(W_2^s(b_{w1} - b_{w2}) + b_2^s) \in R^{1024} \\
R &= [b_{w1} \odot b_{w2}; b_{w1} - b_{w2}; s_1 \odot C; s_2 \odot C] \in R^{4096} \\
\hat{y_i} &= W_3(gelu(W_2(gelu(W_3R + b_3)) + b_2)) + b_3 \in R
\end{aligned} \tag{1}$$

We train the model using binary cross-entropy

---

[10] We take 100 context words to the right and left of the first and second event trigger words respectively when the number of context words in between them exceeds 200.

loss. During inference, we use the best-first clustering approach, where we select the antecedent having the highest pairwise coreference score based on the coreference classifier, to build event chains.

5 Experiments

5.1 Datasets and Evaluation Setup

We use the news documents from the KBP 2016 for validation, and use news documents from KBP 2017 and RED corpora as well as discussion forum documents from the KBP 2017 corpus to evaluate the usefulness of our acquired data[11]. KBP 2016, KBP 2017 and RED corpora contain 85, 83 and 30 news documents respectively, and KBP 2017 has 84 discussion forum documents. KBP corpora have been widely used for evaluating in-document event coreference resolution systems. We further evaluate our models on the RED corpus to examine systems' performance across different event types. KBP 2016 and 2017 corpora are annotated using a subset of 20 subtypes from 38 subtypes used in KBP 2015. On the contrary, RED documents are comprehensively annotated with event coreference relations with no restriction on event types or subtypes, thus, allowing us to evaluate coreference resolution performance on a broad range of events. Besides, we evaluate the performance of models across text genres by evaluating our models trained with news articles on KBP 2017 discussion forum documents.

Following previous work on event coreference resolution, we evaluate all the event coreference resolution systems using the official KBP 2017 scorer v1.8. The scorer employs four coreference scoring measures, namely B^3 (Bagga and Baldwin, 1998), CEAFe (Luo, 2005), MUC (Vilain et al., 1995) and BLANC (Recasens and Hovy, 2011) and the unweighted average of their F1 scores AVG_{F1}. In addition, since MUC directly evaluates pairwise coreference links, we also report MUC precision and recall scores.

5.2 Implementation Details

We use an ensemble of multi-layer feed-forward neural network classifiers to identify event mentions (Choubey and Huang, 2018) for both news and discussion forum documents in KBP 2017 corpus. For the RED corpus, we use gold event mentions as that event extraction system can identify events from only eight event types annotated in KBP 2015 corpus. The coreference classifier uses a three-layer feed-forward neural network with 1024-512-1 units for scoring coreference likelihood. Two single-neural layers, used to transform element-wise dot product and difference between two event embeddings used for controlling context input, use 1024 units each. All hidden activations are followed by dropout with the rate of 0.1 for regularization (Srivastava et al., 2014). All models are trained using AdamW optimizer (Loshchilov and Hutter, 2017; Kingma and Ba, 2014) with four different learning rates (1e-4, 5e-5, 1e-5, 5e-6) and for maximum of 100,000 updates. We use the batch size of 16 and evaluate the model after every 5,000 steps. The epoch and learning rate yielding the best validation performance, average F1 score on KBP 2016 news documents, are used to obtain the final model. Bert model is kept fixed during the training. All experiments are performed on NVIDIA GTX 2080 Ti 11GB using PyTorch 1.2.0+cu92 (Paszke et al., 2019) and HuggingFace Transformer libraries (Wolf et al., 2019).

5.3 Baseline Systems

Trigger Match (+Paraphrase): It links event mentions with the same trigger word (or are lexical paraphrases) as coreferential. Trigger match is a strong baseline for event coreference resolution.

Feature-based Classifier: The neural network classifier that uses GloVe (Pennington et al., 2014) based event trigger word embeddings and binary features indicating argument overlaps.

Choubey and Huang (2018): It models correlations between event coreference chains and document topic structures through a heuristics-based ILP formulation and has achieved the best event coreference resolution performance to date on both KBP 2016 and KBP 2017 datasets.

5.4 Our Systems

KBP 2015, Paraphrase-based pairs, Post-Filtering Paraphrase pairs and KBP 2015+Post-Filtering Paraphrase pairs: The mention pair model, proposed in § 4, trained on different combinations of acquired and human-annotated datasets. *KBP 2015* is trained on event pairs from news docu-

---

[11] ECB+ (Cybulska and Vossen, 2014) is another popular dataset for evaluating event coreference resolution. However, documents in ECB+ are selectively annotated, comprising only of event mentions and within-document coreference chains that are relevant to cross-document event coreference chains. Since our data acquisition methodology is designed for collecting within-document event pairs, we decided to exclude evaluations on the ECB+ corpus.

ments in the KBP 2015 corpus. *Paraphrase-based pairs* is trained on paraphrase event pairs without rules-based filtering (§3.1). *Post-Filtering Paraphrase pairs* is trained on paraphrase event pairs that are filtered using rules defined over news discourse structure (§3.2). *KBP 2015+Post-Filtering Paraphrase pairs* is trained on aggregation of KBP 2015 and Post-Filtering Paraphrase event pairs.

Student Training: The mention pair model trained using the recently proposed self-training approach with a student network (Xie et al., 2019). We first train a teacher mention pair model on the KBP 2015 corpus, then use the teacher model to annotate samples from unannotated news articles. We use the same set of event pairs from Xinhua articles in the Gigaword corpus, set the same upper bound of 20 coreferential and 200 non-coreferential pairs per event trigger word. Also, to allow fair comparisons, we selected only high scoring event pairs (likelihood $\geq$ 0.9) and collected 11,390 coreferential and 272,083 non-coreferential pairs. Finally, we train a new student network with the combined KBP 2015 and teacher-annotated event pairs.

Masked Training: The mention pair model trained on all annotated and automatically acquired (or teacher annotated in case of student training model) event pairs. However, to limit the over-dependence on lexical features[12], we replace both the event trigger words with the *[MASK]* token for all acquired event pairs. Annotated event pairs from KBP 2015 are left unchanged.

5.5 Results and Analysis

The first segment in Table 3 shows the results for all models on KBP 2017 news articles corpus. The mention-pair model trained on KBP 2015 corpus using pre-trained language model and larger event context outperforms both local feature-based as well as the discourse-structure aware previous best model (Choubey and Huang, 2018), outperforming Choubey and Huang (2018) by 2.26 points in average F1 score. The improvement is consistent across all metrics. Specifically, the used mention pair model gains MUC F1 score by 9.76 and 3.33 points over feature-based and discourse aware systems, indicating that BERT-based embedding is more effective in resolving coreference links without exclusively modeling event-arguments or discourse-related features. The model trained on event pairs acquired following the proposed automatic strategy also outperforms Choubey and Huang (2018) by 1.24 and 0.56 points on MUC F1 and average F1 scores respectively. However, this model does worse than the equivalent model trained on KBP 2015 data, which can be explained by the related distribution of KBP 2015 and KBP 2017 datasets. Overall, training the model on KBP 2015 data combined with the acquired event pairs performs the best, outperforming both models trained on KBP 2015 only and the one trained with student training by 1.04 and 0.14 points respectively.

As shown in the second segment of Table 3, the improvement in the average F1 of the model trained on KBP 2015 over the trigger match baseline reduces to 2.3 points on the RED news articles corpus, compared to 5.69 points on KBP 2017 news articles. Mainly, RED annotates all event types while KBP has only 8 event types, and the change in event domains affects the overall performance gain of model. The model trained on our Post-Filtering Paraphrase event pairs performs similarly to the one trained on KBP 2015, implying that the former generalizes similarly to the model trained on human-annotated data when applied to new data out of the training data distribution. Similar to the performance gain on KBP 2017 news articles, combining both KBP 2015 and acquired event pairs improves the average F1 on RED news articles, achieving the highest average F1 gain of 3.98 points against the trigger match baseline. Note that student training also improves performance on RED news articles. However, it is 1.26 points lower on average F1 score than the *KBP 2015+Post-Filtering Paraphrase pairs* model.

In the third segment of Table 3, we compare the performance of all models on a different text genre by evaluating them on the discussion forum documents from the KBP 2017 corpus. With shared event types, the model trained on KBP 2015 achieves the best result with 1.76 points improvement in the average F1 score over the lemma match baseline. The model trained using acquired event pairs, *Post-Filtering Paraphrase pairs*, achieves performance comparable to the model trained on KBP 2015. However, combining the KBP 2015 data with acquired event pairs (the model *KBP 2015+Post-Filtering Paraphrase pairs*) does not further improve the performance. Overall, we observe that none of the models obtain substantial performance improvement. The smaller improvements

---

[12] All acquired event pairs are either synonyms or exhibit hypernym or hyponym relations

Model	b^3_{F1}	$ceafe_{F1}$	muc_R	muc_P	muc_{F1}	$blanc_{F1}$	AVG_{F1}
KBP 2017 News Articles							
Trigger Match	48.96	45.67	26.16	36.63	30.52	29.30	38.61
Trigger Match+Paraphrase	48.92	45.35	27.36	36.41	31.25	29.83	38.84
Feature-based Classifier	50.24	48.47	-	-	30.81	29.94	39.87
Choubey and Huang (2018)	50.35	48.61	-	-	37.24	31.94	42.04
KBP 2015	51.57	**50.90**	33.91	50.49	40.57	34.15	44.30
Paraphrase-based pairs	48.10	42.36	38.05	37.01	37.52	31.64	39.91
Post-Filtering Paraphrase pairs	50.94	47.81	31.77	48.77	38.48	33.19	42.60
KBP 2015+Post-Filtering Paraphrase pairs	**52.29**	50.50	35.24	**55.23**	43.03	35.53	45.34
Masked Training	52.10	50.72	36.31	53.02	43.10	35.51	**45.36**
Student Training	51.85	49.91	**38.18**	49.73	**43.20**	**35.83**	45.20
Masked Training	51.91	50.12	37.11	50.82	42.90	35.50	45.11
RED News Articles							
Trigger Match	88.07	84.21	42.63	35.14	38.52	64.34	68.78
Trigger Match+Paraphrase	87.18	83.09	47.16	33.87	39.43	64.88	68.65
KBP 2015	88.33	85.48	52.38	39.08	44.76	65.77	71.08
Paraphrase-based pairs	82.01	76.74	**68.02**	30.39	42.01	63.09	65.96
Post-Filtering Paraphrase pairs	**89.25**	86.70	47.39	41.63	44.32	63.75	71.0
KBP 2015+Post-Filtering Paraphrase pairs	**89.25**	**86.96**	56.00	**43.40**	48.91	66.74	72.96
Masked Training	89.16	86.90	58.04	43.31	**49.61**	**67.50**	**73.29**
Student Training	87.91	84.95	59.18	39.30	47.23	66.70	71.70
Masked Training	88.11	84.92	58.50	39.69	47.29	67.44	71.94
KBP 2017 Discussion Forum Documents							
Trigger Match	37.29	39.15	20.36	19.06	19.69	18.25	28.59
Trigger Match + Paraphrase	36.94	38.52	21.26	19.13	20.14	18.14	28.44
KBP 2015	38.11	38.67	25.33	**23.76**	24.52	**20.10**	**30.35**
Paraphrase-based pairs	35.58	34.30	28.65	21.37	24.48	19.19	28.39
Post-Filtering Paraphrase pairs	**39.12**	**41.52**	17.34	20.75	18.89	18.81	29.59
KBP 2015+Post-Filtering Paraphrase pairs	37.43	38.16	26.24	22.27	24.09	20.01	29.92
Masked Training	38.33	39.64	21.71	20.68	21.19	19.19	29.59
Student Training	36.80	36.68	**28.80**	22.68	**25.38**	20.08	29.73
Masked Training	37.06	38.01	22.77	20.00	21.29	17.51	28.47

Table 3: Results for event coreference resolution systems on the KBP 2017 and RED corpora. Feature-based Classifier results are directly taken from Choubey and Huang (2018). The results are statistically significant using bootstrap and permutation test (Dror et al., 2018) with $p<0.01$ between *Post-Filtering Paraphrase pairs* and *Paraphrase-based Pairs* and $p<0.002$ between *KBP 2015+Post-Filtering Paraphrase pairs+Masked Training* and *KBP 2015* models on both KBP 2017 and RED news articles test sets. Further, results for *KBP 2015+Post-Filtering Paraphrase pairs+Masked Training* are statistically significant compared to both *Student Training* and *Student Training+Masked Training* with $p<0.002$ on the RED news articles test set.

for all models on discussion forum documents, with the increased data size, also indicate the need for specialized learning algorithms to build a model that can generalize to a new text genre.

Post-Filtering Paraphrase Filtering and Masked Training: The model trained on Post-Filtering Paraphrase event pairs outperforms the one trained on paraphrase-based pairs by 2.69 and 5.04 average F1 points on KBP 2017 and RED news articles test sets respectively. Using news discourse structure-based rules to first constrain coreferential event paraphrase pairs within main sentences or headline and then add non-coreferential event paraphrase pairs from historical sentences inhibits the model from exclussively relying on lexical features. Further, masked training helps to completely circumvent any bias induced in a model by limiting coreferential event pairs to lexical paraphrases, which slightly improved the average F1 score.

Distributional Analysis of Predicted Coreferential Event Pairs across different Discourse Content Type Pairs: Finally, we analyze the distribution of predicted coreferential event pairs across sentence pairs with different discourse content types on the validation dataset. We use the gold coreferential event pairs to identify the top 10 content type pairs of sentences that most frequently contain coreferential event mention pairs. Then, for the models trained on KBP 2015, Post-Filtering Paraphrase pairs and their combination with masked training, we report true-positive, false-positive, and false-negative predictions, shown in Figure 1. To ensure uniformity with rules used in

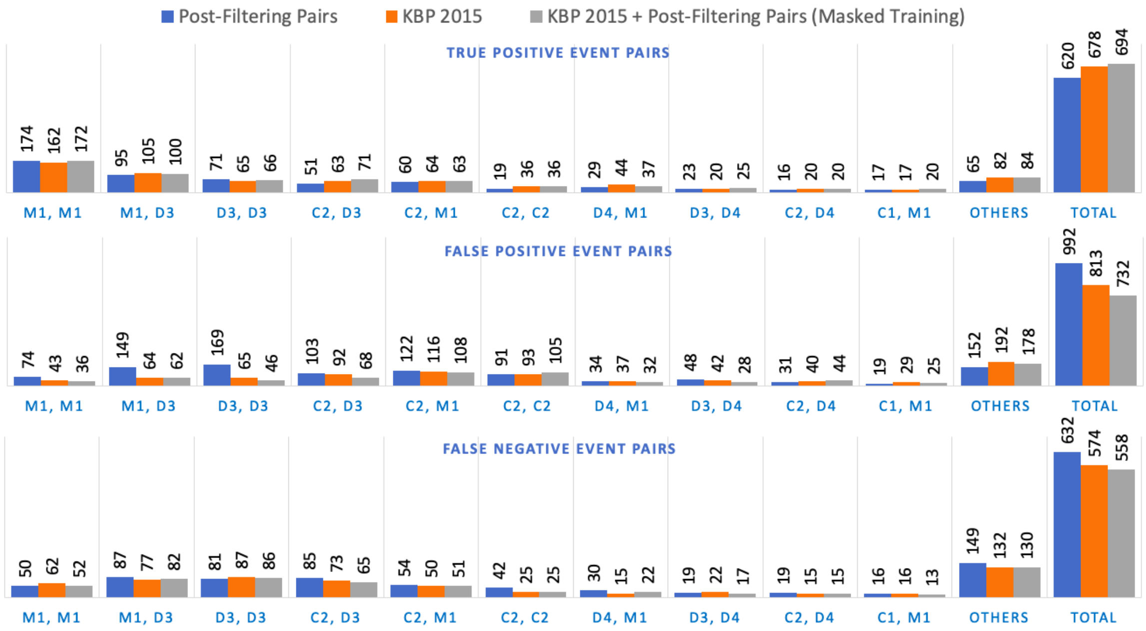


Figure 1: Distributions of Predicted Coreferential Event Pairs across different Discourse Content Type Pairs.

§3.2, we merge the headline with main sentences.

Contrary to the rule that exclusively acquires coreferential event pairs from main sentences or headline, the classifier trained on acquired event pairs predicts coreferential event pairs across all discourse content type pairs. Notably, the Post-Filtering Pairs model predicted a comparable number of coreferential event pairs, 248, 244 and 240, in the (M1, M1), (M1, D3) and (D3, D3) content type pairs respectively. However, the number of true positives in (M1, M1) content pair is more than twice the number in either of the (M1, D3) or (D3, D3). This is expected given that the distribution of gold coreferential event pairs is normally skewed towards (M1, M1).

In comparison, models trained on KBP 2015 or combined KBP 2015 and Post-Filtering pairs have lower false-positives while exhibiting similar distributions for true-positive predictions. Intuitively, despite second phase bootstrapping to include non-coreferential paraphrase pairs, the model trained solely on acquired event pairs focuses on lexical features more than the model trained on human-annotated corpus. On the other hand, masked training effectively overcomes excessive reliance on lexical cues and helps achieve a higher true positive rate without increasing false positives.

6 Conclusions and Future Work

We presented an automatic data acquisition strategy for event coreference resolution by mining the functional news discourse structure. We performed both qualitative and empirical studies to determine the effectiveness of our proposed strategy. We found that the model trained on automatically acquired event pairs performs similarly to the model trained on human-annotated corpus when evaluated on the test set covering general event domains. Further, augmenting acquired event pairs to existing human-annotated data improves the performance of the model on both training-domain and broader domain test sets. For future work, we intend to develop new training algorithms to improve the generalization capability of models on a new text genre. Further, we plan to evaluate a similar event coreference data acquisition strategy for new languages.

Acknowledgments

We thank our anonymous reviewers for providing insightful review comments. We gratefully acknowledge support from National Science Foundation via the awards IIS-1942918 and IIS-1755943. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: the views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF or the U.S. Government.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Granada.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189.

Cosmin Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1412–1422.

Cosmin Adrian Bejan and Sanda Harabagiu. 2014. Unsupervised event coreference resolution. *Computational Linguistics*, 40(2):311–347.

Chen Chen and Vincent Ng. 2014. Sinocoreferencer: An end-to-end chinese event coreference resolver. In *LREC*, volume 2, page 3. Citeseer.

Chen Chen and Vincent Ng. 2015. Chinese zero pronoun resolution: A joint unsupervised discourse-aware model rivaling state-of-the-art resolvers. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 320–326, Beijing, China. Association for Computational Linguistics.

Zheng Chen and Heng Ji. 2009. Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 54–57. Association for Computational Linguistics.

Zheng Chen, Heng Ji, and Robert Haralick. 2009. A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the workshop on events in emerging text types*, pages 17–22. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133.

Prafulla Kumar Choubey and Ruihong Huang. 2018. Improving event coreference resolution by modeling correlations between event coreference chains and document topic structures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 485–495.

Prafulla Kumar Choubey, Aaron Lee, Ruihong Huang, and Lu Wang. 2020. Discourse as a function of event: Profiling discourse structure in news articles around the main event. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5374–5386, Online. Association for Computational Linguistics.

Andres Cremisini and Mark Finlayson. 2020. New insights into cross-document event coreference: Systematic comparison and a simplified approach. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 1–10, Online. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 4545–4552, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Agata Cybulska and Piek Vossen. 2015. Translating granularity of event slots into features for event coreference resolution. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 1–10.

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

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 1–10.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500. Association for Computational Linguistics.

Zhengzhong Liu, Jun Araki, Eduard H Hovy, and Teruko Mitamura. 2014. Supervised within-document event coreference using information propagation. In *LREC*, pages 4539–4544.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jing Lu and Vincent Ng. 2018. Event coreference resolution: A survey of two decades of research. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 5479–5486. International Joint Conferences on Artificial Intelligence Organization.

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint inference for event coreference resolution. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3264–3275.

Yaojie Lu, Hongyu Lin, Jialong Tang, Xianpei Han, and Le Sun. 2020. End-to-end neural event coreference resolution.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 25–32. Association for Computational Linguistics.

Katie McConky, Rakesh Nagi, Moises Sudit, and William Hughes. 2012. Improving event coreference by context extraction and dynamic feature weighting. In *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 38–43. IEEE.

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. 2020. Paraphrasing vs coreferring: Two sides of the same coin.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100, Montréal, Canada. Association for Computational Linguistics.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, Texas. Association for Computational Linguistics.

Arun Pandian, Lamana Mulaffer, Kemal Oflazer, and Amna AlZeyara. 2020. Precision event coreference resolution using neural network classifiers. *Computación y Sistemas*, 24(1).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

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

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Marta Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, page 1044–1049. AAAI Press.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 105–112.

Hossein Sahlani, Maryam Hourali, and Behrouz Minaei-Bidgoli. 2020. Coreference resolution using semantic features and fully connected neural network in the persian language. *International Journal of Computational Intelligence Systems*, 13:1002–1013.

Satyan Sangeetha and Michael Arock. 2012. Event coreference resolution using mincut based graph clustering. In *Proceedings of the Fourth International Workshop on Computer Networks & Communications*, pages 253–260.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Van Dijk Teun A. 1986. News schemata. *Studying writing: linguistic approaches*, 1:155–186.

Teun A Van Dijk. 1988a. News analysis. *Case Studies of International and National News in the Press. New Jersey: Lawrence*.

Teun A Van Dijk. 1988b. *News as discourse*. Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th conference on Message understanding*, pages 45–52. Association for Computational Linguistics.

Bonnie Webber and Aravind Joshi. 2012. Discourse structure and computation: Past, present and future. In *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, pages 42–54, Jeju Island, Korea. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2019. Self-training with noisy student improves imagenet classification.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent bayesian model for event coreference resolution. *Transactions of the Association of Computational Linguistics*, 3(1):517–528.

Dian Yu, Xiaoman Pan, Boliang Zhang, Lifu Huang, Di Lu, Spencer Whitehead, and Heng Ji. 2016. Rpi blender tac-kbp2016 system description. In *TAC*.

Xinyu Zuo, Yubo Chen, Kang Liu, and Jun Zhao. 2019. Event co-reference resolution via a multi-loss neural network without using argument information. *Science China Information Sciences*, 62(11).

Joint Learning of Representations for Web-tables, Entities and Types using Graph Convolutional Network

Aniket Pramanick
TCS Research
aniket.pramanick@tcs.com

Indrajit Bhattacharya
TCS Research
b.indrajit@tcs.com

Abstract

Existing approaches for table annotation with entities and types either capture the structure of table using graphical models, or learn embeddings of table entries without accounting for the complete syntactic structure. We propose TabGCN, which uses Graph Convolutional Networks to capture the complete structure of tables, knowledge graph and the training annotations, and jointly learns embeddings for table elements as well as the entities and types. To account for knowledge incompleteness, TabGCN's embeddings can be used to discover new entities and types. Using experiments on 5 benchmark datasets, we show that TabGCN significantly outperforms multiple state-of-the-art baselines for table annotation, while showing promising performance on downstream table-related applications.

1 Introduction

Table data abounds in webpages and organizational documents. Annotation of table entries, such as columns, cells and rows, using available background knowledge (e.g. Yago, DBPedia, Freebase, etc.), such as knowledge of entities and their types, helps in better understanding and semantic interpretation of such tabular data. The challenge, however, is that such web tables do not adhere to any standard format, schema or convention (Limaye et al., 2010). Additionally, knowledge graphs are typically incomplete - entities and types mentioned in tables may not always exist in the knowledge graph. Therefore, it becomes necessary to expand the knowledge graph with new entities (Zhang et al., 2020) and types for annotating tables.

Initial research on table annotation (Limaye et al., 2010; Takeoka et al., 2019; Bhagavatula et al., 2015) used probabilistic graphical models to capture the complete row-column structure of tables and also the knowledge graph for collective annotation. More recent approaches using embeddings (Gentile et al., 2017; Zhang and Balog, 2018; Zhang et al., 2019; Chen et al., 2019; Yin et al., 2020) only partly capture the syntactic structure of tables, and also ignore the structure of the knowledge graph. The problem of incompleteness of the knowledge representation (Zhang et al., 2020) is mostly not addressed.

In this work, we propose the TabGCN model that uses a Graph Convolutional Network (GCN) (Kipf and Welling, 2017) to unify the complete syntactic structure of tables (rows, columns and cells) and that of the knowledge graph (entities and types) via available annotations. The embeddings of the table elements as well as knowledge graph entities and types are trained jointly and end-to-end. While GCNs have been used for learning embeddings for many NLP tasks using the syntactic and semantic structure of natural language sentences (Marcheggiani and Titov, 2017; Vashishth et al., 2019), encoding tabular structure using GCNs has not been addressed before. The model and embeddings thus trained are used to annotate new tables with known entities and types, while discovering hitherto unseen entities and types. Additionally, we use the trained embeddings for tables and rows for downstream table-related tasks - identifying similar tables, and identifying the appropriate table for any row.

We demonstrate these capabilities of TabGCN using experiments on 5 benchmark web table datasets comparing against 5 existing models. We show that WebGCN significantly improves performance for entity and type annotation. For the other tasks, we show that the same embeddings show impressive performance. No existing model can perform all of these tasks.

Our contributions are as follows: **(a)** We propose a model called TabGCN based on the GCN architecture that captures the complete syntactic structure

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1197–1206
April 19 - 23, 2021.

of tables as well as the knowledge representation, and learns embeddings of tables, rows, columns and cells, as well as entities and types jointly and in an end-to-end fashion. **(b)** TabGCN addresses incompleteness in the knowledge representation by discovering new entities and types. **(c)** TabGCN significantly outperforms 5 existing approaches in 5 different benchmark datasets for the task of table annotation. **(d)** The trained embeddings show impressive performance in downstream tasks such as identifying similar tables and assignment of rows to appropriate tables.

2 Related Work

Existing literature on table annotation considers two different types of tables. In general, web tables (Limaye et al., 2010; Takeoka et al., 2019; Bhagavatula et al., 2015) contain mentions of entities under every column, which need to be annotated. In contrast, relational tables (Gentile et al., 2017; Zhang and Balog, 2018; Zhang et al., 2019, 2020) refer to a single entity in each row, with a single core or anchor column and multiple attribute columns. Here, the entire row is annotated with a single entity. While both tasks are important, our focus is on the first category.

In terms of approaches, one category of work considers graphical models to capture table structure and performs joint inference for entity and type classification (Limaye et al., 2010; Takeoka et al., 2019; Bhagavatula et al., 2015). Limaye et al. (2010) uses a Markov Random Field that captures the structure of tables, and creates dependencies between entity and type annotations to jointly classify entities for cells and types for columns. The Markov Random Field (MRF) potentials capture domain knowledge about similarities between cells, between cells and entity lemmas and between entities using the type hierarchy. It cannot handle entities and types not used as labels during training. MeiMei (Takeoka et al., 2019) extends this MRF framework to handle numeric columns and focuses on multi-label classifiers for entities with multiple types. Additionally, it constructs embeddings of entities and types using knowledge graph structure to avoid expensive graph traversal for computing potential function features. However, it still requires manual construction of these features based on domain knowledge. Our GCN architecture is motivated by the structure of the graphical model in these papers. Both these models also jointly label pairs of columns with known relationships, which we do not address in our work.

The second category focuses on embeddings for tables (Gentile et al., 2017; Zhang and Balog, 2018; Zhang et al., 2019). All of these models transform table data to word sequences and then make use of neural language models. Zhang and Balog (2018) make use of RDF2vec for embedding tables, which internally uses neural language models after transforming graphs to sequences. ColNet (Chen et al., 2019) considers columns as cell sequences and uses a CNN to learn the representations of individual cells, which are used to predict entity types for columns. TaBERT (Yin et al., 2020) focuses on retrieving table rows for natural language utterances and learns a joint representation for utterances and cell sequences in table rows using BERT. In summary, these fail to capture the complete row-column structure of tables in the embeddings. In contrast, our GCN architecture captures the structure of all tables, the entities, the types and the training annotations.

For the problem of extending the knowledge graph from tables, Zhang et al. (2020) consider discovery of new entities, but only in the context of relational tables, with a single core column. Their approach make use of pre-trained neural embeddings (word2vec) and cosine similarity in addition to lexical similarity. Our focus is on web tables, where we need to discover new entities and types for all columns in a table. Our approach makes use of GCN embeddings trained using entity and type labels to detect new entities and types.

3 Problem: Table Annotation

In this section, we first define tables, the background knowledge of entities and types, annotation of tables with entities and types, and, finally, the problem of semi-supervised table annotation with incomplete knowledge.

Tables: We are given a set of tables $\mathcal{S}$. Fig.1 shows an example at the top. The k^{th} table $S^k \in \mathcal{S}$, consists of m^k rows and n^k columns of individual cells. The individual cell in the i^{th} row and j^{th} column of the k^{th} table is denoted as x^k_{ij}. We consider textual tables only, so that each x^k_{ij} takes a string as value. Also, we denote the i^{th} row as R^k_i and the j^{th} column as C^k_j.

Entities and Types: We assume background knowledge of entities and entity types (or simply,

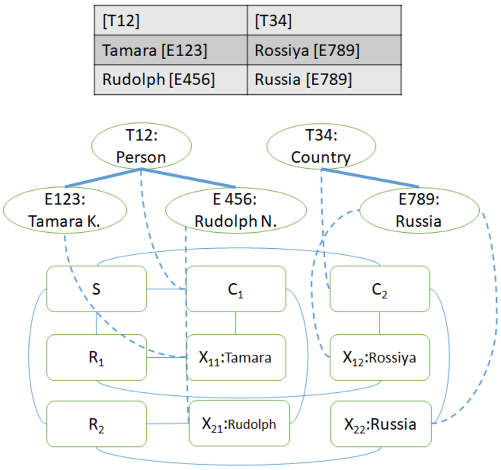


Figure 1: (a) Example table with 2 columns and 2 rows. The column annotations with types and the cell annotations with entities are shown within brackets. (b) GCN graph for example table with table nodes as boxes and entities and types nodes as ovals. Table edges are shown using fine solid lines, entity-type edges with thick solid lines and annotation edges using dashed lines. Lexical similarity edges are typically between cells in different tables and are not shown.

types). Let $\mathcal{T}$ denote the set of types, and $\mathcal{E}$ the set of entities. Each entity E is associated with a type $\mathcal{T}(E)$. For each entity E, there is also an entity description or lemma $\mathcal{L}(E)$. In Fig.1, the oval nodes shows example entities and types. Entity *E123* has associated type *T12:Person* and lemma *Tamara K.*.

Table Annotations: We assume that tables contain information about entities $\mathcal{E}$ and types $\mathcal{T}$. Specifically, each column of each table corresponds to a single type, each cell corresponds to a specific entity of that type. In our example table, cell x_{11} is annotated with entity *E123*, and column C_1 is annotated with type *T12:Person*. Let $\mathcal{T}(C^k_j)$ denote the type associated with column C^k_j, and $\mathcal{E}(x^k_{ij})$ the entity associated with the cell x^k_{ij}. Let A_e be the set of all entity annotations of cells, and A_t that of all type annotations of columns.

Semi-supervised Table Annotation with Incomplete Knowledge: In the semi-supervised table annotation task, we are given the entire set of tables $\mathcal{S}$ but only a subset $A^o_e \subset A_e$ of the entity annotations, and a subset $A^o_t \subset A_t$ of the type annotations are observed. The task is to annotate the unannotated cells and columns of the tables, using the observed annotations as training data.

Let $\mathcal{T}^o$ denote the set of unique types seen in A^o_t, and $\mathcal{E}^o$ the set of unique entities seen in A^o_e. In the *incomplete knowledge setting*, $\mathcal{T}^o \subset \mathcal{T}$, indicating that all the types are not seen in the training annotations. Similarly, all the entities are also not seen in training: $\mathcal{E}^o \subset \mathcal{E}$. Now, the task for the unannotated cells and columns is three-fold. The first is to decide whether these correspond to observed entities A^o_e and observed types A^o_t. We call this *novelty classification*. Next, the non-novel table columns need to be annotated with observed types $\mathcal{T}^o$, and the non-novel table cells with observed entities $\mathcal{E}^o$. We call these *type detection* and *entity detection* respectively. Finally, the columns corresponding to novel types need to be grouped according to distinct novel types, and the cells corresponding to novel entities need to be grouped according to distinct novel entities. We call these *type discovery* and *entity discovery* respectively.

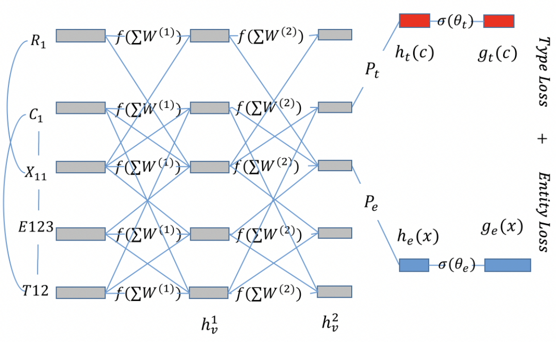


Figure 2: Model architecture showing GCN component with gray nodes, entity classification component with blue nodes and type classification components with red nodes.

4 Joint Table and Knowledge Embedding

We first present at a high level the network architecture of our model, which we call **TabGCN**. The core components TabGCN are (I) a Graph Convolutional Network (GCN), which captures the various syntactic relationships between table and knowledge elements, and then jointly learns embeddings for these via the convolution operation. The GCN embeddings of table elements contain information about both types and entities. These are fed into two parallel components: (II) the Type Classification component, and (III) the Entity Classification component. The Type Classification component first projects the GCN embedding of *table columns* to a *type space* using a *type-projection matrix*, and then uses a soft-max layer to classify this *type embedding* according to observed types. Similarly, the

Entity Classification component first projects the GCN embeddings of *table cells* to an *entity space* using an *entity-projection matrix*, and then uses a soft-max layer to classify this *entity embedding* according to observed entities. Fig.2 shows the high level architecture of our model for a subgraph of our example table graph in Fig.1. The parameters of all three components are trained jointly in an end-to-end fashion using training entity annotations for cells and type annotations for columns via back-propagation. We next describe these components in greater detail.

Graph Convolutional Network: Graph Convolutional Networks (GCN) (Kipf and Welling, 2017; Gilmer et al., 2017) extend the notion of convolution to graphs. Let $\mathcal{G} = (\mathcal{V}, R)$ be a directed graph where $\mathcal{V}$ is the set of n nodes and R the set of directed edges. An edge between nodes $u, v \in \mathcal{V}$ with label L_{uv} is denoted as $(u, v, L_{uv}) \in R$. The edge set includes an inverse edge for each edge and a self-loop for each node. An input feature matrix $\mathcal{X} \in \mathbb{R}^{m\times n}$ contains the input feature representation $x_u \in \mathbb{R}^m$ of each node $\forall u \in \mathcal{V}$ in its columns. Output embedding of a node v at k^{th} layer of GCN is given by

$$h_v^{(k+1)} = f\left(\sum_{u\in\mathcal{N}(v)} W_{L_{uv}}^{(k)} h_u^{(k)} + b_{L_{uv}}^{(k)}\right), \forall v \in \mathcal{V} \tag{1}$$

Here, $W_{L_{uv}}^{(k)}$ and $b_{L_{uv}}^{(k)}$ are label specific model parameters at k^{th} layer and $h_u^{(1)} = x_u$. For classification, a linear classification layer is added on top of final GCN layer. Function $f()$ is a non-linear activation for which we used ReLU.

GCN for Table and Knowledge Elements: Our GCN graph connects table parts, entities and types. We show the GCN graph for our example table in Fig.1. Its node set V consists of **table nodes** (boxes) and **knowledge nodes** (ovals). For each table $S^k \in \mathcal{S}$, the table nodes include one node for each cell x_{ij}^k, one node for each column C_j^k, one node for each row R_i^k, and one node for the table S_k itself. The knowledge nodes include one node for each *observed* type $T^o \in \mathcal{T}^o$ and one node for each *observed* entity $E^o \in \mathcal{E}^o$.

Recall that edges in a GCN serve to bring the embeddings of their end nodes closer, the extent being determined by their weight. With this intuition we create the edges R of different types reflecting the underlying semantics of tables and the annotations. These are **table edges** R_t, **knowledge edges** R_k, **annotation edges** R_a, and **lexical similarity edges** R_l.

Table edges capture the semantics of web tables, which do not have special anchor columns. These are of four categories: a *cell-column edge* between a cell node x_{ij}^k and its corresponding column node C_j^k; a *cell-row edge* between a cell node x_{ij}^k and its corresponding row node R_i^k; a *column-table edge* between a column node C_j^k and its corresponding table node S^k; and a *row-table edge* between a row node R_i^k and its corresponding table node S^k.

Knowledge edges connect each entity node E^o with its corresponding type node $\mathcal{T}(E^o)$.

Annotation edges are of two categories: an *entity annotation edge* for each entity annotation in A_e^o between a cell node x_{ij}^k and its labeled entity node $\mathcal{E}(x_{ij}^k)$; and a *type annotation edge* for each type annotation in A_t^o between a column node C_j^k and its labeled type node $\mathcal{T}(C_j^k)$.

Lexical similarity edges are added between pairs of cells in the same or different tables whose lexical similarity, computed using character-based Jaccard (Limaye et al., 2010), is above a threshold.

All edges are bi-directional. Each of the 8 edge categories above has its own parameters $(W_l^{(k)}, b_l^{(k)})$ for each layer in our GCN. Self loops are added for nodes associated with textual input, specifically, cells and entities with lemmas. For the input representation of such nodes, we use the pre-trained word embeddings for each of their constituent tokens, and take their mean. For this paper we used GloVe (Pennington et al., 2014).

Type Classification: The final (K^{th}) GCN layer generates an embedding $h_v^{(K)}$ for each node v. This contains information about both types and entities. For type classification of a column node c, we first get its *type embedding* $h_t(c)$ by projecting $h_c^{(K)}$ to a *type space* using a *type projection matrix* P_t: $h_t(c) = P_t h_c^{(K)}$. Then we get the probability distribution $g_t(c)$ over known types $\mathcal{T}^o$ for the type embedding $h_t(c)$ using a soft-max layer with weight matrix θ_t: $g_t(c) = \sigma(h_t(c); \theta_t)$. The type projection matrix P_t and the sigmoid weight matrix θ_t form the parameters for this component.

Entity Classification: We follow a similar approach for entity classification of a cell node x. We first project its GCN embedding $h_x^{(K)}$ to an *entity* *space* using an *entity projection matrix* P_e. The *entity embedding* $h_e(x)$ is passed through a soft-max layer with weight matrix θ_e to get the probability distribution $g_e(x)$ over known entities $\mathcal{E}^o$. The entity projection matrix P_e and the sigmoid weight matrix θ_e form the parameters for this component.

Joint Training: The parameters for all three components are trained end-to-end using available entity annotations for cells and type annotations for columns. Specifically, we consider the type predictions $g_t(c)$ for columns and minimize classification loss with the observed entity labels $\mathcal{T}(c)$. We similarly minimize classification loss between entity predictions $g_e(x)$ and observed entity labels $\mathcal{E}(x)$ for cells. We consider a weighted combination of the entity prediction loss and the type prediction loss. We have used cross-entropy as the classification loss function, and Adam (Kingma and Ba, 2015) for optimization.

5 Annotating Tables using Embeddings

Training the network in Sec.4 generates estimates of the model parameters as well as embeddings for all table and knowledge elements. In this section, we describe the use of these parameters and embeddings for the tasks defined in Sec.3. We use $h(v)$ for the GCN embedding of a node v instead of $h_v^{(K)}$ for brevity.

Novelty Classification: To decide whether an unannotated column c corresponds to any of the known types in $\mathcal{T}^o$ (novel type classification), we make use of their type embeddings. Column c corresponds to a new type if its type-space embedding $h_t(c) = P_t h(c)$ is 'far away' from the type space embedding $h_t(T) = P_t h(T)$ for all $T \in \mathcal{T}^o$. More specifically, we use $\delta(\max_{T\in\mathcal{T}^o} \cos(P_t h(c), P_t h(T)) \leq \epsilon_t)$, where ϵ_t is the *novel type threshold*, and $\delta()$ is the Kronecker delta function. A similar approach is followed for deciding if an unannotated cell x corresponds to any of the known entities in $\mathcal{E}^o$ (novel entity classification) by using the entity embeddings. Specifically, we use $\delta(\max_{E\in\mathcal{E}^o} \cos(P_e h(x), P_e h(E)) \leq \epsilon_e)$, where ϵ_e is the *novel entity threshold.*

Type and Entity Detection: Columns and cells determined to be non-novel need to be classified according to known types and entities respectively. This can be done using forward propagation in the trained network on the embeddings of the corresponding nodes. The type prediction $g_t(c)$ of a column c is obtained as $g_t(c) = \sigma(P_t h(c); \theta_t)$. Similarly, the entity prediction $g_e(x)$ of a cell x is obtained as $g_t(x) = \sigma(P_e h(x); \theta_e)$.

Type and Entity Discovery: On the other hand, columns and cells determined to be novel need to be grouped according to distinct new types and entities respectively. This is done by clustering their projections in the appropriate space. Specifically, for type discovery, we take the *type embeddings* $h_t(c)$ of all novel columns c and cluster these. Similarly, for entity discovery, we cluster the *entity embeddings* $h_e(x)$ of all novel cells x. The clustering algorithm needs to automatically determine the number of clusters in both cases. In this paper, we have used Silhouette clustering (Rousseeuw, 1987) as a representative non-parametric clustering algorithm. Other algorithms approaches such as Bayesian non-parametric techniques (Teh, 2010) may be used here.

Down-stream Inference for Rows and Tables: Training annotations are only provided for cells and columns. But embeddings are available for the rows and tables as well after training, and these can be used for different down-stream application tasks. Since these do not involve type or entity spaces, we directly use their GCN embeddings for these tasks. As examples, we define two such tasks here. **Table clustering** is the task of grouping together semantically related tables. For this, we cluster the table embeddings $h(S)$ of all tables $S \in \mathcal{S}$. For consistency, we again use Silhouette clustering. **Row to Table assignment** is the task of assigning a row to its most appropriate table. For this, we assign a row R with embedding $h(R)$ to the table S_k with the 'closest' embedding $h(S_k)$, or, more specifically, to $S^* = \arg\max_{S_k} \cos(h(R), h(S_k))$. If R is a row from a table provided during training, then its embedding is readily available. If it is a new row, then its embedding is created by convolving over the input embeddings of its constituent cells using the trained parameters W_l for cell-row edge.

6 Experiments

In this section, we first present experimental results for table annotation, and then for down-stream table-related tasks. We compare our proposed model **TabGCN** with appropriate state-of-the-art baselines. We have uploaded our source code as supplementary material for reproducibility.

Model	Wiki M		Web M		T2Dv2		Limaye		Wikipedia	
	Type	Entity	Type	Entity	Type	Entity	Type	Entity	Type	Entity
PGM	**0.55**	**0.70**	0.57	0.75	-	-	0.60	0.75	-	-
MeiMei	0.40	-	0.62	-	-	-	0.58	-	-	-
Tab2Vec	-	0.20	-	0.50	-	-	-	0.50	-	-
ColNet	0.20	-	0.47	-	0.59	-	0.47	-	0.60	-
TaBERT	0.20	-	0.49	-	0.59	-	0.47	-	0.59	-
TabGCN	0.33	0.24	**0.84**	**0.83**	**0.82**	-	**0.84**	**0.79**	**0.85**	-

Table 1: Entity and Type detection performance using micro averaged F1 for TabGCN and 5 baselines models on 5 benchmark datasets. Note that T2Dv2 and Wikipedia do not have entity annotations. Also, not all baselines can perform both entity and type detection.

Model	Wiki M		Web M		T2Dv2		Limaye		Wikipedia	
	Type	Entity	Type	Entity	Type	Entity	Type	Entity	Type	Entity
ColNet	0.76	-	0.64	-	0.77	-	0.63	-	0.60	-
TaBERT	0.76	-	0.62	-	0.76	-	0.61	-	0.60	-
TabGCN	**0.88**	**0.86**	**0.84**	**0.84**	**0.90**	-	**0.84**	**0.83**	**0.82**	-

Table 2: Type and Entity Discovery performance using normalized mutual information (NMI) for TabGCN, ColNet and TaBERT. Other baselines cannot address this task. T2DV2 and Wikipedia do not have entity annotations.

Data: We used 5 benchmark web table datasets. Their statistics are shown in Table.4. **Wiki Manual** (Limaye et al., 2010) is a small dataset of simple non-infobox tables from Wikipedia articles. **Web Manual** (Limaye et al., 2010) contains tables fetched by web-crawling using Wiki Manual tables as queries. These two tables are manually annotated with entities and types from YAGO. The tables and annotations are noisier than Wiki Manual. **Limaye** (Chen et al., 2019; Efthymiou et al., 2017) corrects many incorrect annotation in Wiki Manual using entities and types from DBPedia 2015. **T2Dv2** (Chen et al., 2019) contains tables from Common Web Crawl. **Wikipedia** is a publicly available subset of the data used by Efthymiou et al. (2017). This has HTML tables from Wikipedia 2013 with class attribute "wikitable". The last three datasets are annotated using DBPedia. But **T2Dv2** and **Wikipedia** contain *only* type annotations, and entity annotations are not available. For all datasets, we first set aside 30% of the unique entities and types, and effectively all their annotations as unseen during training. Of the remaining annotations, again 30% was removed during training.

Baselines: We compare TabGCN against three existing table annotation approaches. **PGM** (Limaye et al., 2010) uses a probabilistic graphical model and hand-crafted features. Since our entity-type set $\mathcal{T}$ is flat instead of being a hierarchical graph, we use logistic regression instead of structural SVM to estimate model parameters. **MeiMei** (Takeoka et al., 2019) also uses a Markov Random field but embeds the entities and types for fast computation of clique potentials. For both models, we ignore the relation labels for column-pairs and associated potential functions, since we do not address relation detection. **Table2Vec** (Zhang et al., 2019) learns various word2vec-based embeddings for tables for entity annotation. It does not address type annotation. It focuses on relational tables and associates a single entity with a row for its core column. We adapted this model for web tables which associate an entity for each cell. We use their *Table2VecE* version, which models a row as a sequences of all entities that appear within cells of that row. After training the word2vec model, instead of considering the embedding for the entire row as in *Table2VecE*, we create cell-specific embeddings from only the tokens for that cell. **ColNet** (Chen et al., 2019) models each column as a sequence of words from the cells under the column and uses a CNN to predict the column type. We also use an adaptation of TaBERT (Yin et al., 2020) which trains a joint language model for retrieval from tables given utterances. We adapt their approach to independently linearize each row and column of a table as a sequence of cells under that row and column, and get column and row embeddings using the mean of corresponding cell embeddings. For cells, we use pre-trained BERT embeddings.

Hyper-parameters: We used one-hot encodings as inputs for cells and entities with lemmas. We used ReLU as the nonlinear GCN activation, 2 GCN layers with 512 and 256 dimensions. As a result, the 8 GCN weight matrices were Vx512 in the first layer, where V is the vocabulary size,

Model	Wiki M		Web M		T2Dv2		Limaye		Wikipedia	
	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR
Tab2Vec	0.11	0.16	0.13	0.21	0.32	0.39	0.09	0.18	0.29	0.35
TaBERT	0.13	0.18	0.51	0.58	0.32	0.40	0.46	0.57	0.38	0.46
TabGCN	**0.18**	**0.21**	**0.71**	**0.77**	**0.67**	**0.70**	**0.71**	**0.77**	**0.71**	**0.74**

Table 3: Row assignment performance using Hits@1 and MRR for TabGCN, Tab2Vec and TaBERT on all datasets.

Dataset	T	R	C	E	$\mathcal{T}$	E^*	$\mathcal{T}^*$
Wiki Man.	39	36.3	4.2	1026	49	308	15
Web Man.	403	34.4	3.7	883	48	265	14
Limaye	294	27.7	3	504	21	151	6
T2Dv2	345	44.8	4.8	-	27	-	8
Wikipedia	572	24.4	5	-	29	-	9

Table 4: Dataset Statistics: For each dataset, T indictates the number of tables, R the average number of rows per table, C the average number of columns per table, E the number of unique entities and $\mathcal{T}$ the number of unique types used for annotation, E^* the number of new unique entities and $\mathcal{T}^*$ the number of new unique types *not seen* in training annotations.

and 512x256 in the second layer. The entity and type space embeddings had dimension 256, so that the entity and type projection matrices were both 256x256. For training, we used learning rate 0.001, dropout 0.5, 1000 epochs. The weights for combining the type and entity losses was $1 : 2$ for both datasets, optimized manually. All experiments were performed on a Dual-core intel Core i5 Processor with 8GB RAM. The average training time per epoch ranges from 1.3 secs for Wiki Manual to 35.4 secs for T2Dv2.

Detection Results: We first present results for entity and type detection, addressed by most of the baselines. For this task, all models predict entity labels for all cells and type labels for all columns that are unannotated. However, evaluation is *only* for those cells and columns whose true entity and type labels are contained in the observed entities $\mathcal{E}^o$ and observed types $\mathcal{T}^o$ respectively.

For evaluation, as in earlier table annotation papers (Limaye et al., 2010; Chen et al., 2019), we use micro-averaged F1, which takes the weighted average of the F1 scores over the entities or types. This takes class imbalance into account and is therefore more appropriate than accuracy. We note that MeiMei addresses the multi-label setting with multiple possible type labels, and therefore uses ranking evaluation measures (e.g. NDCG). This is not meaningful in our single-labeled setting.

Tab. 1 shows detailed results for all models across datasets. We can see that **TabGCN** significantly outperforms all baselines on all datasets for both detection tasks. The only exception is for Wiki Manual. The graphical model based approaches with handcrafted potential functions outperform the representation learning approaches, possibly on account of the smallness of the dataset. Among the embedding based approaches, TabGCN performs the best.

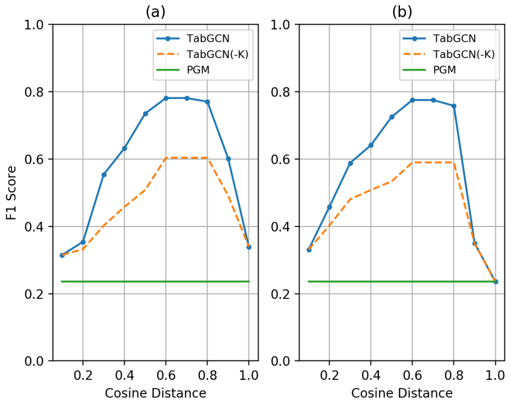


Figure 3: Novelty classification Performance on (a) types and (b) entities for TabGCN, its ablation TabGCN(-K) and PGM for Web Manual, showing F1 on the y-axis and decision threshold on x-axis. PGM does not use a decision threshold. Results on other datasets are very similar. Other baselines cannot address this task.

Novelty Classification Results: In our second experiment, we consider novelty classification. This is an unsupervised task, where a model makes a binary decision for each unannotated column (novel type classification) and for each unannotated cell (novel entity classification). Since the decision depends on the thresholds for type (ϵ_t) and entity (ϵ_e), we plot F1 score on the y-axis against the corresponding threshold on the x-axis. Of the baselines, only PGM can address this task, but outputting a NONE label for the type or entity. Fig.3 shows novelty classification performance for Web Manual. Results for Wiki Manual and Limaye are very similar. **TabGCN** reaches F1 around 0.8 for both tasks for appropriate thresholds. Across thresholds, TabGCN significantly out-

Models	Wiki M		Web M		Limaye	
	Type	Ent.	Type	Ent.	Type	Ent.
-K	0.29	0.23	**0.84**	0.82	0.83	0.78
-E	0.22	-	0.82	-	0.81	-
-T	-	0.22	-	0.63	-	0.63
-L	0.33	0.23	0.80	0.75	0.81	0.74
TabGCN	**0.33**	**0.24**	**0.84**	**0.83**	**0.84**	**0.79**

Table 5: Ablation study for entity and type detection showing micro. F1. -K removes knowledge (entity and type) nodes, -E removes entity nodes, -T removes type nodes and -L removes lexical similarity edges.

performs PGM.

Type and Entity Discovery Results: The final annotation tasks are type discovery, where all unannotated columns that do not correspond to known types in $\mathcal{T}^o$ need to be clustered into distinct new types, and entity discovery, where all unannotated cells that do not correspond to known entities in $\mathcal{E}^0$ need to be clustered into distinct new entities. We used Normalized Mutual Information (NMI) $NMI = \frac{2I(C,Y)}{H(C)+H(Y)}$ between the assigned cluster labels (Y) and the true entity or type labels (C), where $I(,)$ denotes mutual information and $H()$ denotes entropy. In Tab. 2, we see that **TabGCN** performs consistently above 80% for entity and type discovery across datasets, significantly outperforming ColNet and TaBERT.

Ablation Study: Next, we analyze the performance of **TabGCN**, using multiple ablations of the full model. **-K** leaves out the knowledge nodes and their incident edges from the GCN graph during training. **-E** focuses only on types by removing all entity nodes and entity-related edges (type-entity and cell-entity annotation) from the GCN graph. It is trained only using type loss. Note that it cannot perform tasks associated with entities, specifically entity detection, novel entity classification and novel entity discovery. Similarly, **-T** focuses only on entities by removing all type nodes and type-related edges (type-entity and column-type annotation) from the GCN graph. It is trained only using entity loss, and cannot perform tasks associated with types. Finally, **-L** removes the lexical similarity edges from the GCN graph. The results are recorded in Table. 5. We can see that all the components of the architecture contribute to performance improvements. The improvement is statistically significant (using the Wilson Interval with $\alpha = 0.05$) for all ablations other than **-K**. While **-K** performs comparably here, its performance drops significantly for novelty classification, as can be seen in Fig.3.

Model	T2Dv2	Limaye
Tab2Vec	0.66	0.49
TaBERT	0.76	0.51
TabGCN	**0.89**	**0.81**

Table 6: Table clustering performance using NMI for TabGCN, Tab2Vec and TaBERT on T2Dv2 and Limaye. Other baselines cannot address this task. T2Dv2 has 40 table categories and Limaye 15. Other datasets do not have table categories for evaluation.

Downstream Table Related Tasks: Finally, we include some results for the table and row related inference tasks defined at the end of Sec.5. This is to demonstrate how the learnt embeddings can benefit potential down-stream tasks. Note that TabGCN directly outputs table embeddings. Of the baselines, Tab2Vec and TaBERT output row embeddings. For these models, we create table embeddings by averaging the corresponding row embeddings.

In Tab. 6, we record performance for table clustering. **TabGCN** again significantly outperforms both baselines for both datasets.

We finally consider row-to-table assignment. In this task, one randomly selected row is removed from every table during training. These rows then need to be classified according to their parent table. Since the models output a ranking of tables for each row, we evaluate using two ranking related measures. *Hits@1* measures the fraction of rows with the correct table at rank 1. *Mean Reciprocal rank* (MRR) is the mean of the reciprocal rank of the correct table over all rows, and its perfect value is 1.0. In Tab. 3, we again see that **TabGCN** performs the best across datasets.

In summary, we have demonstrated the usefulness of learning embeddings of table elements and knowledge elements jointly using both entity and type losses in an end-to-end fashion for type and entity annotation on 5 benchmark datasets. In addition, we have demonstrated how the learned embeddings can be useful for downstream table-related tasks. In all cases, **TabGCN** has significantly outperformed multiple state-of-the-art baselines using probabilistic graphical models as well as other neural approaches.

7 Conclusion

We have proposed a model for that jointly learns representations of tables, rows, columns and cell, as well as entities and types by capturing the complete syntactic structure of all tables, the relevant entities and types and the available annotations using the Graph Convolutional Network. As a result, TabGCN unifies the benefits of probabilistic graphical model based approaches and embedding based approaches for table annotation. Using these embeddings, TabGCN significantly outperforms existing approaches for table annotation, as well as entity and type discovery.

References

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2015. Tabel: Entity linking in web tables. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, volume 9366 of *Lecture Notes in Computer Science*, pages 425–441. Springer.

Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles Sutton. 2019. Colnet: Embedding the semantics of web tables for column type prediction. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 29–36. AAAI Press.

Vasilis Efthymiou, Oktie Hassanzadeh, Mariano Rodriguez-Muro, and Vassilis Christophides. 2017. Matching web tables with knowledge base entities: From entity lookups to entity embeddings. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, pages 260–277. Springer.

Anna Lisa Gentile, Petar Ristoski, Steffen Eckel, Dominique Ritze, and Heiko Paulheim. 2017. Entity matching on web tables: a table embeddings approach for blocking. In *Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017, Venice, Italy, March 21-24, 2017*, pages 510–513. OpenProceedings.org.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*, 3(1-2):1338–1347.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1506–1515. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Peter Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. comput. appl. math. 20, 53-65. *Journal of Computational and Applied Mathematics*, 20:53–65.

Kunihiro Takeoka, Masafumi Oyamada, Shinji Nakadai, and Takeshi Okadome. 2019. Meimei: An efficient probabilistic approach for semantically annotating tables. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 281–288. AAAI Press.

Yee Whye Teh. 2010. Dirichlet processes. In *Encyclopedia of Machine Learning*. Springer.

Shikhar Vashishth, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya, and Partha Talukdar. 2019. Incorporating syntactic and semantic information in word embeddings using graph convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3308–3318, Florence, Italy. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8413–8426. Association for Computational Linguistics.

Li Zhang, Shuo Zhang, and Krisztian Balog. 2019. Table2vec: Neural word and entity embeddings for table population and retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1029–1032. ACM.

Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1553–1562. ACM.

Shuo Zhang, Edgar Meij, Krisztian Balog, and Ridho Reinanda. 2020. Novel entity discovery from web tables. In *Proceedings of The Web Conference 2020*, WWW '20, page 1298–1308, New York, NY, USA. Association for Computing Machinery.

Multimodal Text Style Transfer for Outdoor Vision-and-Language Navigation

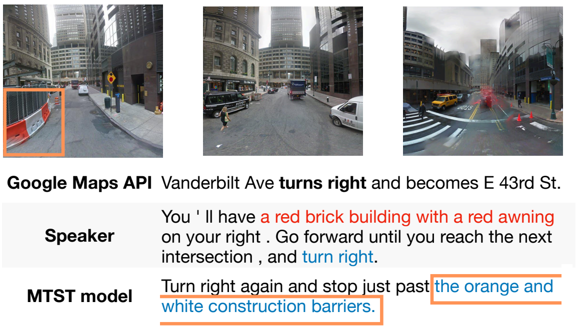
Wanrong Zhu[†], **Xin Eric Wang**[‡], **Tsu-Jui Fu**[†], **An Yan**[§],
Pradyumna Narayana[*], **Kazoo Sone**[*], **Sugato Basu**[*], **William Yang Wang**[†]
[†]UC Santa Barbara, [‡]UC Santa Cruz, [§]UC San Diego, [*]Google
`{wanrongzhu,tsu-juifu,william}@cs.ucsb.edu`, `xwang366@ucsc.edu`,
`ayan@eng.ucsd.edu`, `{pradyn,sone,sugato}@google.com`

Abstract

One of the most challenging topics in Natural Language Processing (NLP) is visually-grounded language understanding and reasoning. Outdoor vision-and-language navigation (VLN) is such a task where an agent follows natural language instructions and navigates a real-life urban environment. Due to the lack of human-annotated instructions that illustrate intricate urban scenes, outdoor VLN remains a challenging task to solve. This paper introduces a Multimodal Text Style Transfer (MTST) learning approach and leverages external multimodal resources to mitigate data scarcity in outdoor navigation tasks. We first enrich the navigation data by transferring the style of the instructions generated by Google Maps API, then pre-train the navigator with the augmented external outdoor navigation dataset. Experimental results show that our MTST learning approach is model-agnostic, and our MTST approach significantly outperforms the baseline models on the outdoor VLN task, improving task completion rate by 8.7% relatively on the test set. [1]

1 Introduction

A key challenge for Artificial Intelligence research is to go beyond static observational data and consider more challenging settings that involve dynamic actions and incremental decision-making processes (Fenton et al., 2020). Outdoor vision-and-language navigation (VLN) is such a task, where an agent navigates in an urban environment by grounding natural language instructions in visual scenes, as illustrated in Fig. 1. To generate a series of correct actions, the navigation agent must comprehend the instructions and reason through the visual environment.



Google Maps API	Vanderbilt Ave **turns right** and becomes E 43rd St.
Speaker	You ' ll have a red brick building with a red awning on your right . Go forward until you reach the next intersection , and turn right.
MTST model	Turn right again and stop just past the orange and white construction barriers.

Figure 1: An outdoor VLN example with instructions generated by Google Maps API (ground truth), the Speaker model, and our MTST model. Tokens marked in red indicate incorrectly generated instructions, while the blue tokens suggest alignments with the ground truth. The orange bounding boxes show that the objects in the surrounding environment have been successfully injected into the style-modified instruction.

Different from indoor navigation (Anderson et al., 2018; Wang et al., 2018; Fried et al., 2018; Wang et al., 2019; Ma et al., 2019a; Tan et al., 2019; Ma et al., 2019b; Ke et al., 2019), the outdoor navigation task takes place in urban environments that contain diverse street views (Mirowski et al., 2018; Chen et al., 2019; Mehta et al., 2020). The vast urban area leads to a much larger space for an agent to explore and usually contains longer trajectories and a wider range of objects for visual grounding. This requires more informative instructions to address the complex navigation environment. However, it is expensive to collect human-annotated instructions that depict the complicated visual scenes to train a navigation agent. The issue of data scarcity limits the navigator's performance in the outdoor VLN task.

To deal with the data scarcity issue, Fried et al. (2018) proposes a Speaker model to generate additional training pairs. However, synthesizing instructions purely from visual signals is hard, especially for outdoor environments, due to visual complexity.

---

[1] Our code and dataset is released at `https://github.com/VegB/VLN-Transformer`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1207–1221
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

On the other hand, template-based navigation instructions on the street view can be easily obtained via the Google Map API, which may serve as additional learning signals to boost outdoor navigation tasks. But instructions generated by Google Maps API mainly consist of street names and directions, while human-annotated instructions in the outdoor navigation task frequently refer to street-view objects in the panorama. The distinct instruction style hinders the full utilization of external resources.

Therefore, we present a novel Multimodal Text Style Transfer (MTST) learning approach to narrow the gap between template-based instructions in the external resources and the human-annotated instructions for the outdoor navigation task. It can infer style-modified instructions for trajectories in the external resources and thus mitigate the data scarcity issue. Our approach can inject more visual objects in the navigation environment to the instructions (Fig. 1), while providing direction guidance. The enriched object-related information can help the navigation agent learn the grounding between the visual environment and the instruction.

Moreover, different from previous LSTM-based navigation agents, we propose a new VLN Transformer to predict outdoor navigation actions. Experimental results show that utilizing external resources provided by Google Maps API during the pre-training process improves the navigation agent's performance on Touchdown, a dataset for outdoor VLN (Chen et al., 2019). In addition, pre-training with the style-modified instructions generated by our multimodal text style transfer model can further improve navigation performance and make the pre-training process more robust. In summary, the contribution of our work is four-fold:

- We present a new Multimodal Text Style Transfer learning approach to generate style-modified instructions for external resources and tackle the data scarcity issue in the outdoor VLN task.
- We provide the Manh-50 dataset with style-modified instructions as an auxiliary dataset for outdoor VLN training.
- We propose a novel VLN Transformer model as the navigation agent for outdoor VLN and validate its effectiveness.
- We improve the task completion rate by 8.7% relatively on the test set for the outdoor VLN task with the VLN Transformer model pre-trained on the external resources processed by our MTST approach.

2 Related Work

Vision-and-Language Navigation (VLN) is a task that requires an agent to achieve the final goal based on the given instructions in a 3D environment. Besides the generalizability problem studied by previous works (Wang et al., 2018, 2019; Tan et al., 2019; Zhang et al., 2020), the data scarcity problem is another critical issue for the VLN task, expecially in the outdoor environment(Chen et al., 2019; Mehta et al., 2020; Xiang et al., 2020). Fried et al. (2018) obtains a broad set of augmented training data for VLN by sampling trajectories in the navigation environment and using the Speaker model to back-translate their instructions. However, the Speaker model might cause the error propagation issue since it is not trained on large corpora to optimize generalization. While most existing works select navigation actions dynamically along the way in the unseen environment during testing, Majumdar et al. (2020) proposes to test in previously explored environments and convert the VLN task to a classification task over the possible paths. This approach performs well in the indoor setting, but is not suitable for outdoor VLN where the environment graph is different.

Multimodal Pre-training has attracted much attention to improving multimodal tasks performances. The models usually adopt the Transformer structure to encode the visual features and the textual features (Tan and Bansal, 2019; Lu et al., 2019; Chen et al., 2020; Sun et al., 2019; Li et al., 2019; Huang et al., 2020b; Luo et al., 2020; Li et al., 2020; Zheng et al., 2020; Wei et al., 2020; Tsai et al., 2019). During pre-training, these models use tasks such as masked language modeling, masked region modeling, image-text matching to learn the cross-modal encoding ability, which later benefits the multimodal downstream tasks. Majumdar et al. (2020) proposes to use image-text pairs from the web to pre-train VLN-BERT, a visiolinguistic transformer-based model similar to the model proposed by Lu et al. (2019).

A concurrent work by Hao et al. (2020) proposes to use Transformer for indoor VLN. Our VLN Transformer is different from their model in several key aspects: (1) The pre-training objectives are different: Hao et al. (2020) pre-trains the model on the same dataset for training, while we create an augmented, stylized dataset for outdoor VLN using the proposed MTST method. (2) Benefiting from the effective external resource, a simple navigation loss is employed in our VLN Transformer, while they adopt the masked language modeling to better train their model. (3) Model-wise, instead of encoding the whole instruction into one feature, we use sentence-level encoding since Touchdown instructions are much longer than R2R instructions. (4) We encode the trajectory history, while their model encodes the panorama for the current step.

Unsupervised Text Style Transfer is an approach to mitigate the lack of parallel data for supervised training. One line of work encodes the text into a latent vector and manipulate the text representation in the latent space to transfer the style. Shen et al. (2017); Hu et al. (2017); Yang et al. (2018) use variational auto-encoder to encode the text, and use a discriminator to modify text style. John et al. (2019); Fu et al. (2018) rely on models with encoder-decoder structure to transfer the style. Another line of work enriches the training data by generating pseudo-parallel data via back-translation (Artetxe et al., 2018; Lample et al., 2018b,a; Zhang et al., 2018).

3 Methods

3.1 Task Definition

In the vision-and-language navigation task, the reasoning navigator is asked to find the correct path to reach the target location following the instructions (a set of sentences) $\mathcal{X} = \{s_1, s_2, \ldots, s_m\}$. The navigation procedure can be viewed as a series of decision making processes. At each time step t, the navigation environment presents an image view $\boldsymbol{v}_t$. With reference to the instruction $\mathcal{X}$ and the visual view $\boldsymbol{v}_t$, the navigator is expected to choose an action $a_t \in \mathcal{A}$. The action set $\mathcal{A}$ for urban environment navigation usually contains four actions, namely *turn left*, *turn right*, *go forward*, and *stop*.

3.2 Overview

Our Multimodal Text Style Transfer (MTST) learning mainly consists of two modules, namely the *multimodal text style transfer model* and the *VLN Transformer*. Fig. 2 provides an overview of our MTST approach. We use the multimodal text style transfer model to narrow the gap between the human-annotated instructions for the outdoor navigation task and the machine-generated instructions in the external resources. The multimodal text style transfer model is trained on the dataset for outdoor navigation, and it learns to infer style-modified instructions for trajectories in the external resources. The VLN Transformer is the navigation agent that generates actions for the outdoor VLN task. It is trained with a two-stage training pipeline. We first pre-train the VLN Transformer on the external resources with the style-modified instructions and then fine-tune it on the outdoor navigation dataset.

Source	Instruction
Google Maps API	Head northwest on E 23rd St toward 2nd Ave. Turn left at the 2nd cross street onto 3rd Ave.
Human Annotator	Orient yourself so you are facing the same as the traffic on the 4 lane road. Travel down this road until the first intersection. Turn left and go down this street with the flow of traffic. You'll see a black and white stripped awning on your right as you travel down the street.

Table 1: For the outdoor VLN task, the instructions provided by Google Maps API is distinct from the instructions written by human annotators.

3.3 Multimodal Text Style Transfer Model

Instruction Style The navigation instructions vary across different outdoor VLN datasets. As shown in Table 1, the instructions generated by Google Maps API is template-based and mainly consists of street names and directions. In contrast, human-annotated instructions for the outdoor VLN task emphasize the visual environment's attributes as navigation targets. It frequently refers to objects in the panorama, such as traffic lights, cars, awnings, etc. The goal of conducting multimodal text style transfer is to inject more object-related information in the surrounding navigation environment to the machine-generated instruction while keeping the correct guiding signals.

Masking-and-Recovering Scheme The multimodal text style transfer model is trained with a "masking-and-recovering" (Zhu et al., 2019; Liu et al., 2019; Donahue et al., 2020; Huang et al., 2020a) scheme to inject objects that appeared in the panorama into the instructions. We mask out certain portions in the instructions and try to recover the missing contents with the help of the remaining instruction skeleton and the paired trajectory. To be specific, we use NLTK (Bird et al., 2009) to mask out the object-related tokens in the human-annotated instructions, and the street names

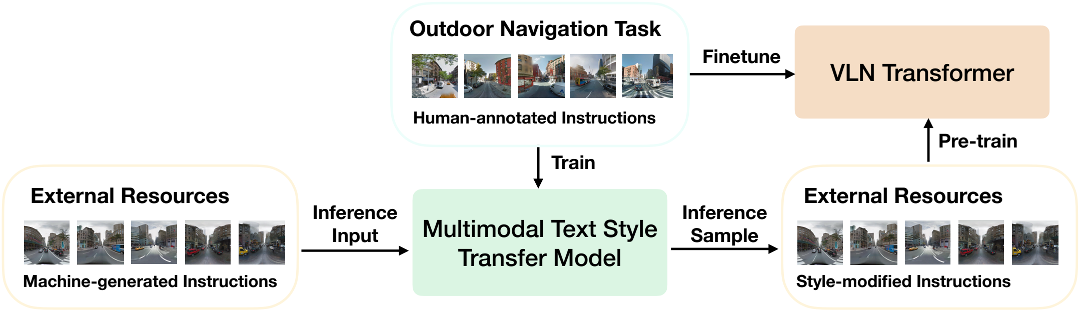


Figure 2: An overview of the Multimodal Text Style Transfer (MTST) learning approach for vision-and-language navigation in real-life urban environments. Details are described in Section 3.2.

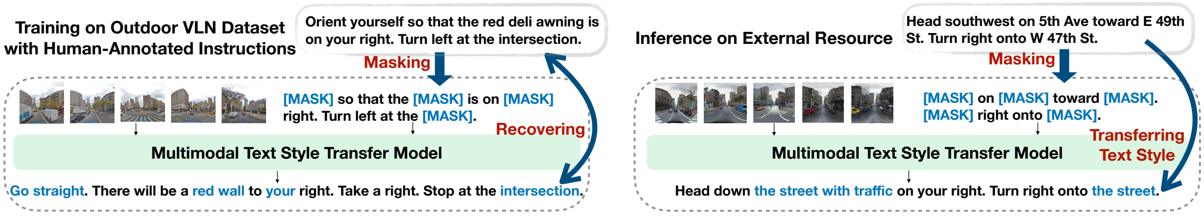


Figure 3: An example of the training and inference process of the multimodal text style transfer model. During training, we mask out the objects in the human-annotated instructions to get the instruction template. The model takes both the trajectory and the instruction skeleton as input, and the training objective is to recover the instructions with objects. When inferring new instructions for external trajectories, we mask the street names in the original instructions and prompt the model to generate new object-grounded instructions.

in the machine-generated instructions[2]. Multiple tokens that are masked out in a row will be replaced by a single [MASK] token. We aim to maintain the correct guiding signals for navigation after the style transfer process. Tokens that provide guiding signals, such as “turn left" or “take a right", will not be masked out. Fig. 3 provides an example of the “masking-and-recovering" process during training and inferring.

Model Structure Fig. 3 illustrates the input and expected output of our multimodal text style transfer model. We build the multimodal text style transfer model upon the Speaker model proposed by Fried et al. (2018). On top of the visual-attention-based LSTM (Hochreiter and Schmidhuber, 1997) structure in the Speaker model, we inject the textual attention of the masked instruction skeleton $\mathcal{X}'$ to the encoder, which allows the model to attend to original guiding signals.

The encoder takes both the visual and textual inputs, which encode the trajectory and the masked instruction skeletons. To be specific, each visual view in the trajectory is represented as a feature vector $\boldsymbol{v}' = [\boldsymbol{v}'_v; \boldsymbol{v}'_\alpha]$, which is the concatenation of the visual encoding $\boldsymbol{v}'_v \in \mathbb{R}^{512}$ and the orientation encoding $\boldsymbol{v}'_\alpha \in \mathbb{R}^{64}$. The visual encoding $\boldsymbol{v}'_v$ is the output of the last but one layer of the RESNET18 (He et al., 2016) of the current view. The orientation encoding $\boldsymbol{v}'_\alpha$ encodes current heading α by repeating vector $[sin\alpha, cos\alpha]$ for 32 times, which follows Fried et al. (2018). As described in section 3.4, the feature matrix of a panorama is the concatenation of eight projected visual views.

In the multimodal style transfer encoder, we use a soft-attention module (Vaswani et al., 2017) to calculate the grounded visual feature $\hat{\boldsymbol{v}}_t$ for current view at step t:

$$attn_{v_{t,i}} = softmax((\boldsymbol{W}_v \boldsymbol{h}_{t-1})^T \boldsymbol{v}'_i) \qquad (1)$$

$$\hat{\boldsymbol{v}}_t = \sum_{i=1}^{8} = attn_{v_{t,i}} \boldsymbol{v}'_i \qquad (2)$$

where $\boldsymbol{h}_{t-1}$ is the hidden context of previous step, $\boldsymbol{W}_v$ refers to the learnable parameters, and $attn_{v_{t,i}}$ is the attention weight over the i_{th} slice of view $\boldsymbol{v}'_i$ in current panorama.

We use full-stop punctuations to split the input text into multiple sentences. The rationale is to enable alignment between the street views and the semantic guidance in sub-instructions. For each sentence in the input text, the textual encoding $\boldsymbol{s}'$

---

[2] We masked out the tokens with the following part-of-speech tags: [JJ, JJR, JJS, NN, NNS, NNP, NNPS, PDT, POS, RB, RBR, RBS, PRP$, PRP, MD, CD]

is the average of all the tokens' word embedding in the current sentence. We also use a soft-attention modules to calculate the grounded textual feature $\hat{\boldsymbol{s}}_t$ at current step t:

$$attn_{s_{t,j}} = softmax((\boldsymbol{W}_s \boldsymbol{h}_{t-1})^T \boldsymbol{s}'_j) \tag{3}$$

$$\hat{\boldsymbol{s}}_t = \sum_{j=1}^{M} attn_{s_{t,j}} \boldsymbol{s}'_j \tag{4}$$

where $\boldsymbol{W}_s$ refers to the learnable parameters, $attn_{s_{t,j}}$ is the attention weight over the j_{th} sentence encoding $\boldsymbol{s}'_j$ at step t, and M denotes the maximum sentence number in the input text. The input text for the multimodal style transfer encoder is the instruction template $\mathcal{X}'$.

Based on the grounded visual feature $\hat{\boldsymbol{v}}_t$, the grounded textual feature $\hat{\boldsymbol{s}}_t$ and the visual view feature $\boldsymbol{v}'_t$ at current timestamp t, the hidden context can be given as:

$$\boldsymbol{h}_t = LSTM([\hat{\boldsymbol{v}}_t; \hat{\boldsymbol{s}}_t; \boldsymbol{v}'_t]) \tag{5}$$

Training Objectives We train the multimodal text style transfer model in the teacher-forcing manner (Williams and Zipser, 1989). The decoder generates tokens auto-regressively, conditioning on the masked instruction template $\mathcal{X}'$, and the trajectory. The training objective is to minimize the following cross-entropy loss:

$$\mathcal{L}(x_1, x_2, \dots, x_n | \mathcal{X}', \boldsymbol{v}'_1, \dots, \boldsymbol{v}'_N)$$
$$= -\log \prod_{j=1}^{n} P(x_j | x_1, ..., x_{j-1}, \mathcal{X}', \boldsymbol{v}'_1, \dots, \boldsymbol{v}'_N) \tag{6}$$

where $x_1, x_2, \dots, x_n$ denotes the tokens in the original instruction $\mathcal{X}$, n is the total token number in $\mathcal{X}$, and N denotes the maximum view number in the trajectory.

3.4 VLN Transformer

The VLN Transformer is the navigation agent that generates actions in the outdoor VLN task. As illustrated in Fig. 4, our VLN Transformer is composed of an instruction encoder, a trajectory encoder, a cross-modal encoder that fuses the modality of the instruction encodings and trajectory encodings, and an action predictor.

Instruction Encoder The instruction encoder is a pre-trained uncased BERT-base model (Devlin et al., 2019). Each piece of navigation instruction is split into multiple sentences by the full-stop punctuations. For the i_{th} sentence $s_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,l_i}\}$ that contains l_i tokens, its sentence embedding $\boldsymbol{h}^s_i$ is calculated as:

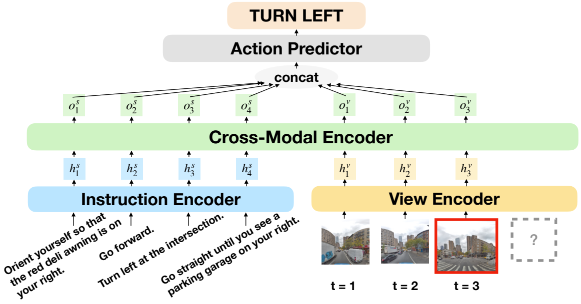


Figure 4: Overview of the VLN Transformer. In this example, the VLN Transformer predicts to take a left turn for the visual scene at $t = 3$.

$$\boldsymbol{w}_{i,j} = \mathcal{BERT}(x_{i,j}) \in \mathbb{R}^{768} \tag{7}$$

$$\boldsymbol{h}^s_i = \mathcal{FC}\left(\frac{\sum_{j=1}^{l_i} \boldsymbol{w}_{i,j}}{l_i}\right) \in \mathbb{R}^{256} \tag{8}$$

where $\boldsymbol{w}_{i,j}$ is the word embedding for $x_{i,j}$ generated by BERT, and $\mathcal{FC}$ is a fully-connected layer.

View Encoder We use the view encoder to retrieve embeddings for the visual views at each time step. Following Chen et al. (2019), we embed each panorama $\mathbf{I}_t$ by slicing it into eight images and projecting each image from an equirectangular projection to a perspective projection. Each of the projected image of size 800×460 will be passed through the RESNET18 (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015). We use the output of size $128 \times 100 \times 58$ from the fourth to last layer before classification as the feature for each slice. The feature map for each panorama is the concatenation of the eight image slices, which is a single tensor of size $128 \times 100 \times 464$. We center the feature map according to the agent's heading α_t at timestamp t. We crop a $128 \times 100 \times 100$ sized feature map from the center and calculate the mean value along the channel dimension. The resulting 100×100 features is regarded as the current panorama feature $\hat{\mathbf{I}}_t$ for each state. Following Mirowski et al. (2018), we then apply a three-layer convolutional neural network on $\hat{\mathbf{I}}_t$ to extract the view features $\boldsymbol{h}^v_t \in \mathbb{R}^{256}$ at timestamp t.

Cross-Modal Encoder In order to navigate through complicated real-world environments, the agent needs to grasp a proper understanding of the natural language instructions and the visual views jointly to choose proper actions for each state. Since the instructions and the trajectory lies in different modalities and are encoded separately, we introduce the cross-modal encoder to fuse the features from different modalities and jointly encode the instructions and the trajectory. The cross-modal encoder is an 8-layer Transformer encoder (Vaswani et al., 2017) with mask. We use eight self-attention heads and a hidden size of 256.

In the teacher-forcing training process, we add a mask when calculating the multi-head self-attention across different modalities. By masking out all the future views in the ground-truth trajectory, the current view $\boldsymbol{v}_t$ is only allowed to refer to the full instructions and all the previous views that the agent has passed by, which is $[\boldsymbol{h}^s_1, \boldsymbol{h}^s_2, \ldots, \boldsymbol{h}^s_M; \boldsymbol{h}^v_1, \boldsymbol{h}^v_2, \ldots, \boldsymbol{h}^v_{t-1}]$, where M denotes the maximum sentence number.

Since the Transformer architecture is based solely on attention mechanism and thus contains no recurrence or convolution, we need to inject additional information about the relative or absolute position of the features in the input sequence. We add a learned segment embedding to every input feature vector specifying whether it belongs to the sentence encodings or the view encodings. We also add a learned position embedding to indicate the relative position of the sentences in the instruction sequence or the trajectory sequence's views.

Action Predictor The action predictor is a fully-connected layer. It takes the concatenation of the cross-modal encoder's output up to the current timestamp t as input, and predicts the action a_t for view $\boldsymbol{v}_t$:

$$\boldsymbol{h}_{concat} = \boldsymbol{h}^s_1 || \ldots || \boldsymbol{h}^s_M || \boldsymbol{h}^v_1 || \ldots || \boldsymbol{h}^v_t \tag{9}$$

$$a_t = argmax(\mathcal{FC}(\mathcal{T}(\boldsymbol{h}_{concat}))) \tag{10}$$

where $\mathcal{FC}$ is a fully-connected layer in the action predictor, and $\mathcal{T}$ refers to the Transformer operation in the cross-modal encoder. During training, we use the cross-entropy loss for optimization.

4 Experiments

4.1 Datasets

Outdoor VLN Dataset For the outdoor VLN task, we conduct experiments on the Touchdown dataset (Chen et al., 2019; Mehta et al., 2020), which is designed for navigation in realistic urban environments. Based on Google Street View[3], Touchdown's navigation environment encompasses 29,641 Street View panoramas of the Manhattan area in New York City, which are connected by 61,319 undirected edges. The dataset contains 9,326 trajectories for the navigation task, and each trajectory is paired with a human-written instruction. The training set consists of 6,526 samples, while the development set and the test set are made up of 1,391 and 1,409 samples, respectively.

External Resource We use the StreetLearn dataset as the external resource for the outdoor VLN task (Mirowski et al., 2018). The StreetLearn dataset is another dataset for navigation in real-life urban environments based on Google Street View. StreetLearn contains 114k panoramas from New York City and Pittsburgh. In the StreetLearn navigation environment, the graph for New York City contains 56k nodes and 115k edges, while the graph for Pittsburgh contains 57k nodes and 118k edges. The StreetLearn dataset contains 580k samples in the Manhattan area and 8k samples in the Pittsburgh area for navigation.

While the StreetLearn dataset's trajectory contains more panorama along the way on average, the paired instructions are shorter than the Touchdown dataset. We extract a sub-dataset *Manh-50* from the original large scale StreetLearn dataset for the convenience of conducting experiments. Manh-50 consists of navigation samples in the Manhattan area that contains no more than 50 panoramas in the whole trajectory, containing 31k training samples. We generate style-transferred instructions for the Manh-50 dataset, which serves as an auxiliary dataset, and will be used to pre-train the navigation models. More details can be found in the appendix.

4.2 Evaluation Metrics

We use the following metrics to evaluate VLN performance: (1) *Task Completion (TC)*: the accuracy of completing the navigation task correctly. Following Chen et al. (2019), the navigation result is considered correct if the agent reaches the specific goal or one of the adjacent nodes in the environment graph. (2) *Shortest-Path Distance (SPD)*: the mean distance between the agent's final position and the goal position in the environment graph. (3) *Success weighted by Edit Distance (SED)*: the normalized Levenshtein edit distance between the path predicted by the agent and the reference path, which is constrained only to the successful navigation. (4) *Coverage weighted by Length Score (CLS)*: a measurement of the fidelity of the agent's path

---

[3] `https://developers.google.com/maps/documentation/streetview/intro`

with regard to the reference path. (5) *Normalized Dynamic Time Warping (nDTW)*: the minimized cumulative distance between the predicted path and the reference path, normalized by the reciprocal of the square root of the reference path length. The value is rescaled by taking the negative exponential of the normalized value. (6) *Success weighted Dynamic Time Warping (SDTW)*: the nDTW value where the summation is only over the successful navigation.

TC, SPD, and SED are defined by Chen et al. (2019). CLS is defined by Jain et al. (2019). nDTW and SDTW are originally defined by Ilharco et al. (2019), in which nDTW is normalized by the length of the reference path. We adjust the normalizing factor to be the reciprocal of the square root of the reference path length for length invariance (Mueen and Keogh, 2016). In case the reference trajectories length has a salient variance, our modification to the normalizing factor made the nDTW and SDTW scores invariant to the reference length.

4.3 Results and Analysis

In this section, we report the outdoor VLN performance and the quality of the generated instructions to validate the effectiveness of our MTST learning approach. We compare our VLN Transformer with the baseline model and discuss the influence of pre-training on external resources with/without instruction style transfer.

Outdoor VLN Performance We compare our VLN Transformer with RCONCAT (Chen et al., 2019; Mirowski et al., 2018) and GA (Chen et al., 2019; Chaplot et al., 2018) as baseline models. Both baseline models encode the trajectory and the instruction in an LSTM-based manner and use supervised training with Hogwild! (Recht et al., 2011). Table 2 presents the navigation results on the Touchdown validation and test sets, where VLN Transformer performs better than RCONCAT and GA on most metrics with the exception of SPD and CLS.

Pre-training the navigation models on Manh-50 with template-based instructions can partially improve navigation performance. For all three agent models, the scores related to successful cases—such as TC, SED, and SDTW—witness a boost after being pre-trained on vanilla Manh-50. However, the instruction style difference between Manh-50 and Touchdown might misguide the agent in the pre-training stage, resulting in a performance drop on SPD for our VLN Transformer model.

In contrast, our MTST learning approach can better utilize external resources and further improve navigation performance. Pre-training on Manh-50 with style-modified instructions can stably improve the navigation performance on all the metrics for both the RCONCAT model and the VLN Transformer. This also indicates that our MTST learning approach is model-agnostic.

Table 4 compares the SPD values on success and failure navigation cases. In the success cases, VLN Transformer has better SPD scores, which is aligned with the best SED results in Table 2. Our model's inferior SPD results are caused by taking longer paths in failure cases, which also harms the fidelity of the generated path and lowers the CLS scores. Nevertheless, every coin has two sides, and exploring more areas when getting lost might not be a complete bad behavior for the navigation agent. We leave this to future study.

Multimodal Text Style Transfer in VLN We attempt to reveal each component's effect in the multimodal text style transfer model. We pre-train the VLN Transformer with external trajectories and instructions generated by different models, then fine-tune it on the TouchDown dataset.

According to the navigation results in Table 3, the instructions generated by the Speaker model misguide the navigation agent, indicating that relying solely on the Speaker model cannot reduce the gap between different instruction styles. Adding textual attention to the Speaker model can slightly improve the navigation results, but still hinders the agent from navigating correctly. The style-modified instructions improve the agent's performance on all the navigation metrics, suggesting that our Multimodal Text Style Transfer learning approach can assist the outdoor VLN task.

Quality of the Generated Instruction We evaluate the quality of instructions generated by the Speaker and the MTST model. We utilize five automatic metrics for natural language generation to evaluate the quality of the generated instructions, including BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Elliott and Keller, 2013), CIDEr (Vedantam et al., 2015) and SPICE (Anderson et al., 2016). In addition, we calculate the guiding signal match rate (MR) by comparing the appearance of "turn left" and "turn right". If the generated instruction contains the

Model	Dev Set						Test Set					
	TC ↑	SPD ↓	SED ↑	CLS ↑	nDTW ↑	SDTW ↑	TC ↑	SPD ↓	SED ↑	CLS ↑	nDTW ↑	SDTW ↑
RCONCAT	10.6	20.4	10.3	48.1	22.5	9.8	11.8	20.4	11.5	47.9	22.9	11.1
+*M-50*	11.8	19.1	11.4	48.7	23.1	10.9	12.1	19.4	11.8	49.4	24.0	11.3
+*M-50* +style	11.9	19.9	11.5	48.9	23.8	11.1	12.6	20.4	12.3	48.0	23.9	11.8
GA	12.0	18.7	11.6	51.9	25.2	11.1	11.9	19.0	11.5	51.6	24.9	10.9
+*M-50*	12.3	**18.5**	11.8	**53.7**	26.2	11.3	13.1	**18.4**	12.8	**54.2**	26.8	12.1
+*M-50* +style	12.9	**18.5**	12.5	52.8	26.3	11.9	13.9	**18.4**	13.5	53.5	27.5	12.9
VLN Transformer	14.0	21.5	13.6	44.0	23.0	12.9	14.9	21.2	14.6	45.4	25.3	14.0
+*M-50*	14.6	22.3	14.1	45.6	25.0	13.4	15.5	21.9	15.4	45.9	26.1	14.2
+*M-50* +style	**15.0**	20.3	**14.7**	50.1	**27.0**	**14.2**	**16.2**	20.8	**15.7**	50.5	**27.8**	**15.0**

Table 2: Navigation results on the outdoor VLN task. +*M-50* denotes pre-training with vanilla Manh-50 which contains machine-generated instructions; in the +*style* setting, the model is pre-trained with Manh-50 trajectories and style-modified instructions that are generated by our MTST model.

Model	Dev Set						Test Set					
	TC ↑	SPD ↓	SED ↑	CLS ↑	nDTW ↑	SDTW ↑	TC ↑	SPD ↓	SED ↑	CLS ↑	nDTW ↑	SDTW ↑
VLN Transformer +*M-50*	14.6	22.3	14.1	45.6	25.0	13.4	15.5	21.9	15.4	45.9	26.1	14.2
+speaker	7.6	26.2	7.3	34.6	14.6	7.0	8.3	25.4	8.0	36.3	15.9	7.7
+text_attn	11.7	**20.1**	11.3	46.3	23.2	10.7	11.8	**20.5**	11.5	47.3	23.2	11.0
+style	**15.0**	20.3	**14.7**	**50.1**	**27.0**	**14.2**	**16.2**	20.8	**15.7**	**50.5**	**27.8**	**15.0**

Table 3: Ablation study of the multimodal text style transfer model on the outdoor VLN task. In the +*speaker* setting, the instructions used in pre-training are generated by the Speaker (Fried et al., 2018), which only attends to the visual input; +*text_attn* denotes that we add a textual attention module to the Speaker to attend to both the visual input and the machine-generated instructions provided by Google Maps API.

Model	Dev Set		Test Set	
	S_SPD↓	F_SPD↓	S_SPD↓	F_SPD↓
RCONCAT	0.64	22.68	0.67	23.06
+*M-50*	0.68	21.53	0.69	21.97
+*M-50* +style	0.66	22.48	0.69	23.21
GA	0.65	21.15	0.66	21.41
+*M-50*	0.70	**20.95**	0.77	**21.09**
+*M-50* +style	0.65	21.11	0.70	21.26
VLN Transformer	0.66	24.92	0.63	24.84
+*M-50*	0.67	25.94	0.63	25.77
+*M-50* +style	**0.59**	23.72	**0.62**	24.67

Table 4: *S_SPD* and *F_SPD* denotes the average SPD value on success and failure cases respectively.

same number of guiding signals in the same order as the ground truth instruction, then this instruction pair is considered to be matched. We also calculate the number of different infilled tokens (#infill) in the generated instruction[4]. This reflects the model's ability to inject object-related information during style transferring. Among the 9,326 trajectories in the Touchdown dataset, 9,000 are used to train the MTST model, while the rest form the validation set.

Model	BLEU	METEOR	ROUGE_L	CIDEr	SPICE	MR	#infill
Speaker	15.1	20.6	22.2	1.4	20.7	8.3	160
Text_Attn	23.8	23.3	29.6	10.0	24.6	35.7	182
MTST	**30.6**	**28.8**	**39.7**	**27.8**	**30.6**	**46.7**	**308**

Table 5: Quantitative evaluation of the instructions generated by Speaker, Speaker with textual attention and our MTST model.

We report the quantitative results on the validation set in Table 5. After adding textual attention to the Speaker, the evaluation performance on all seven metrics improved. Our MTST model scores the highest on all seven metrics, which indicates that the "masking-and-recovering" scheme is beneficial for the multimodal text style transfer process. The results validate that the MTST model can generate higher quality instructions, which refers to more visual objects and provide more matched guiding signals.

Human Evaluation We invite human judges on Amazon Mechanical Turk to evaluate the quality of the instructions generated by different models. We conduct a pairwise comparison, which covers 170 pairs of instructions generated by Speaker, Speaker with textual attention, and our MTST model. The instruction pairs are sampled from the Touchdown

---

[4] We regard tokens with the following part-of-speech tags as infilled tokens: [JJ, JJR, JJS, NN, NNS, NNP, NNPS, PDT, POS, RB, RBR, RBS, PRP$, PRP, MD, CD]

Choice (%)	MTST *vs* Speaker			MTST *vs* Text_Attn			Speaker *vs* Text_Attn		
	MTST	Speaker	Tie	MTST	Text_Attn	Tie	Speaker	Text_Attn	Tie
Better describes the street view	**67.9**	22.8	9.3	**44.3**	35.8	19.9	28.2	**62.7**	9.1
More aligned with the ground truth	**64.6**	26.8	8.6	**37.6**	33.9	28.5	25.3	**62.5**	12.2

Table 6: Human evaluation results of the instructions generated by Speaker, Speaker with textual attention and our MTST model with pairwise comparisons.

validation set. Each pair of instructions, together with the ground truth instruction and the gif that illustrates the navigation street view, is presented to 5 annotators. The annotators are asked to make decisions from the aspect of guiding signal correctness and instruction content alignment. Results in Table 6 show that annotators think the instructions generated by our MTST model better describe the street view and is more aligned with the ground-truth instructions.

Case Study We demonstrate case study results to illustrate the performance of our Multimodal Text Style Transfer learning approach. Fig. 5 provides two showcases of the instruction generation results. As listed in the charts, the instructions generated by the vanilla Speaker model have a poor performance in keeping the guiding signals in the ground truth instructions and suffer from hallucinations, which refers to objects that have not appeared in the trajectory. The Speaker with textual attention can provide guidance direction. However, the instructions generated in this manner does not utilize the rich visual information in the trajectory. On the other hand, the instructions generated by our multimodal text style transfer model inject more object-related information (“the light", “scaffolding") in the surrounding navigation environment to the StreetLearn instruction while keeping the correct guiding signals.

StreetLearn	**Head northwest** on W 35th St toward Hudson Blvd E. **Turn right** at the 1st cross street onto Hudson Blvd E.
Original Speaker	Turn so the red construction is on your left and the red brick building is on your right. Go forward to the intersection and turn right. You'll have a red brick building with a red awning on your right.
Speaker with Textual Attention	Head in the direction of traffic. Turn right at the first intersection.
Multimodal Text Style Transfer	Move forward with traffic on the right turn right at the light. Continue straight.

StreetLearn	**Turn right** onto W 36th St. **Turn right** onto Dyer Ave.
Original Speaker	Go to the next intersection and turn left again. There will be a building with a red awning on your right. Go straight through the next intersection.
Speaker with Textual Attention	Turn right at the next intersection. Stop just before the next intersection.
Multimodal Text Style Transfer	Turn right again at the next intersection. On your right will be scaffolding on your right. Turn right.

Figure 5: Two showcases of the instruction generation results. The red tokens indicate incorrectly generated instructions, while the blue tokens suggest alignments with the ground truth. The orange bounding boxes show that the objects in the surrounding environment have been successfully injected into the style-modified instruction.

5 Conclusion

In this paper, we proposed the Multimodal Text Style Transfer learning approach for outdoor VLN. This learning framework allows us to utilize out-of-domain navigation samples in outdoor environments and enrich the original navigation reasoning training process. Experimental results show that our MTST approach is model-agnostic, and our MTST learning approach outperforms the baseline models on the outdoor VLN task. We believe our study provides a possible solution to mitigate the data scarcity issue in the outdoor VLN task. In future studies, we would love to explore the possibility of constructing an end-to-end framework. We will also further improve the quality of style-modified instructions, and quantitatively evaluate the alignment between the trajectory and the style-transferred instructions.

Acknowledgments

We would like to show our gratitude towards Jiannan Xiang, who kindly shares his experimental code on Touchdown, and Qi Wu, who provides valuable feedback to our initial draft. We also thank the anonymous reviewers for their thought-provoking comments. The UCSB authors were sponsored by an unrestricted gift from Google. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the sponsor.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: semantic propositional image caption evaluation. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V*, volume 9909 of *Lecture Notes in Computer Science*, pages 382–398. Springer.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3674–3683. IEEE Computer Society.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly.

Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. 2018. Gated-attention architectures for task-oriented language grounding. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 2819–2826. AAAI Press.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. TOUCHDOWN: natural language navigation and spatial reasoning in visual street environments. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12538–12547. Computer Vision Foundation / IEEE.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2492–2501. Association for Computational Linguistics.

Desmond Elliott and Frank Keller. 2013. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1292–1302. ACL.

Norman E. Fenton, Martin Neil, and Anthony C. Constantinou. 2020. *The Book of Why: The New Science of Cause and Effect, Judea Pearl, Dana Mackenzie. Basic Books (2018)*, volume 284.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3318–3329.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 663–670. AAAI Press.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13134–13143. IEEE.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. volume 9, pages 1735–1780.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR.

Yichen Huang, Yizhe Zhang, Oussama Elachqar, and Yu Cheng. 2020a. INSET: sentence infilling with inter-sentential transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2502–2515. Association for Computational Linguistics.

Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. 2020b. Pixel-bert: Aligning image pixels with text by deep multi-modal transformers. *CoRR*, abs/2004.00849.

Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. 2019. General evaluation for instruction conditioned navigation using dynamic time warping. In *Visually Grounded Interaction and Language (ViGIL), NeurIPS 2019 Workshop, Vancouver, Canada, December 13, 2019*.

Vihan Jain, Gabriel Magalhães, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1862–1872. Association for Computational Linguistics.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 424–434. Association for Computational Linguistics.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha S. Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6741–6749. Computer Vision Foundation / IEEE.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5039–5049. Association for Computational Linguistics.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020. Unicoder-vl: A universal encoder for vision and language by cross-modal pretraining. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11336–11344. AAAI Press.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. volume abs/1908.03557.

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

Dayiheng Liu, Jie Fu, Pengfei Liu, and Jiancheng Lv. 2019. TIGS: an inference algorithm for text infilling with gradient search. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4146–4156. Association for Computational Linguistics.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. 2020. Univilm: A unified video and language pre-training model for multimodal understanding and generation. volume abs/2002.06353.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6732–6740. Computer Vision Foundation / IEEE.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VI*, volume 12351 of *Lecture Notes in Computer Science*, pages 259–274. Springer.

Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. 2020. Retouchdown: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view. volume abs/2001.03671.

Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. 2018. Learning to navigate in cities without a map. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2424–2435.

Abdullah Mueen and Eamonn J. Keogh. 2016. Extracting optimal performance from dynamic time warping. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 2129–2130. ACM.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Benjamin Recht, Christopher Ré, Stephen J. Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 693–701.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. 2015. Imagenet large scale visual recognition challenge. volume 115, pages 211–252.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6830–6841.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 7463–7472. IEEE.

Hao Tan and Mohit Bansal. 2019. LXMERT: learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5099–5110. Association for Computational Linguistics.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2610–2621. Association for Computational Linguistics.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6558–6569. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4566–4575. IEEE Computer Society.

Xin Wang, Qiuyuan Huang, Asli Çelikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *IEEE Conference on Computer Vision and Pattern*

Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 6629–6638. Computer Vision Foundation / IEEE.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI*, volume 11220 of *Lecture Notes in Computer Science*, pages 38–55. Springer.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. 2020. Multi-modality cross attention network for image and sentence matching. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10938–10947. IEEE.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. volume 1, pages 270–280.

Jiannan Xiang, Xin Wang, and William Yang Wang. 2020. Learning to stop: A simple yet effective approach to urban vision-language navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 699–707. Association for Computational Linguistics.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7298–7309.

Yubo Zhang, Hao Tan, and Mohit Bansal. 2020. Diagnosing the environment bias in vision-and-language navigation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 890–897. ijcai.org.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2018. Style transfer as unsupervised machine translation. *CoRR*, abs/1808.07894.

Chen Zheng, Quan Guo, and Parisa Kordjamshidi. 2020. Cross-modality relevance for reasoning on language and vision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7642–7651. Association for Computational Linguistics.

Wanrong Zhu, Zhiting Hu, and Eric P. Xing. 2019. Text infilling. *CoRR*, abs/1901.00158.

A Appendix

A.1 Dataset Comparison

Dataset	#path	#pano	#pano/path	instr_len	#sent/path	#turn/path
Touchdown	6k	26k	35.2	80.5	6.3	2.8
Manh-50	31k	43k	37.2	22.1	2.8	4.1
StreetLearn	580k	114k	29.0	28.6	4.0	13.2

Table 7: Dataset statistics. *path*: navigation path; *pano*: panorama; *instr_len*: average instruction length; *sent*: sentence; *turn*: intersection on the path.

Table 7 lists out the statistical information of the datasets used in pre-training and fine-tuning. Even though the Touchdown dataset and the StreetLearn dataset are built upon Google Street View, and both of them contain urban environments in New York City, pre-training the model with the VLN task on the StreetLearn dataset does not raise a threat of test data leaking. This is due to several causes:

First, the instructions in the two datasets are distinct in styles. The instructions in the StreetLearn dataset is generated by Google Maps API, which is template-based and focuses on street names. However, the instructions in the Touchdown dataset are created by human annotators and emphasize the visual environment's attributes as navigational cues. Moreover, as reported by Mehta et al. (2020), the panoramas in the two datasets have little overlaps. In addition, Touchdown instructions constantly refer to transient objects such as cars and bikes, which might not appear in a panorama from a different time. The different granularity of the panorama spacing also leads to distinct panorama distributions of the two datasets.

A.2 Training Details

We use Adam optimizer (Kingma and Ba, 2015) to optimize all the parameters. During pre-training on the StreetLearn dataset, the learning rate for the RCONCAT model, GA model, and the VLN Transformer is 2.5×10^{-4}. We fine-tune BERT separately with a learning rate of 1×10^{-5}. We pre-train RCONCAT and GA for 15 epochs and pre-train the VLN Transformer for 25 epochs.

When training or fine-tuning on the Touchdown dataset, the learning rate for RCONCAT and GA is 2.5×10^{-4}. For the VLN Transformer, the learning rate to fine-tune BERT is initially set to 1×10^{-5}, while the learning rate for other parameters in the model is initialized to be 2.5×10^{-4}. The learning rate for VLN Transformer will decay. The batch size for RCONCAT and GA is 64, while the VLN Transformer uses a batch size of 30 during training.

Model	TC ↑	SPD ↓	SED ↑	CLS ↑	nDTW ↑	SDTW ↑
no split	9.6	21.8	9.3	46.1	20.0	8.7
split	**13.6**	**20.5**	**13.1**	**47.6**	**24.0**	**12.6**

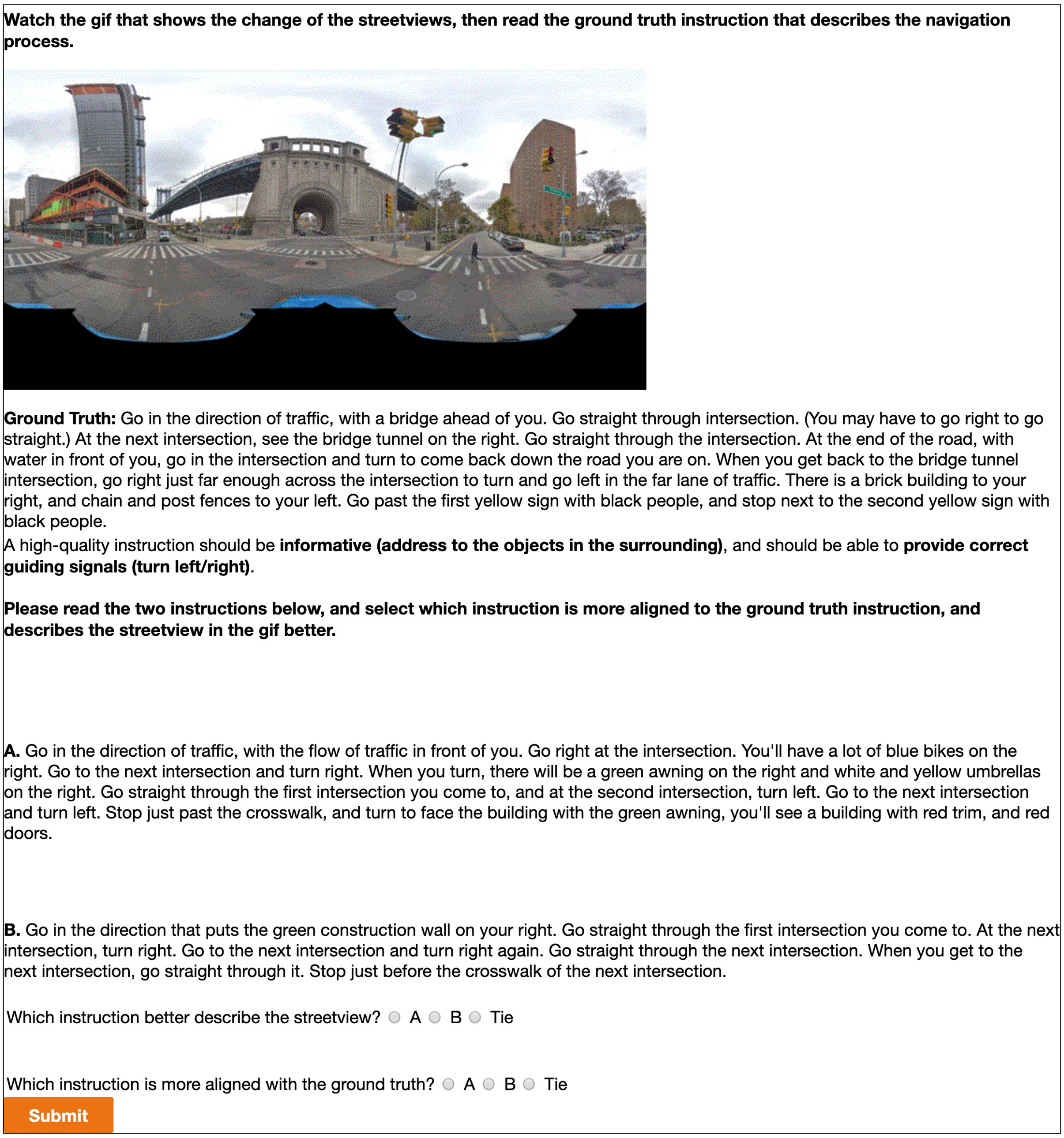
Table 8: Ablation results of the VLN Transformer's instruction split on Touchdown dev set. In *split* setting, the instruction is split into multiple sentences before being encoded by the instruction encoder, while *no split* setting encodes the whole instruction without splitting.

A.3 Split Instructions vs. No Split

We compare VLN Transformer performance with and without splitting the instructions into sentences during encoding. Results in Table 8 show that breaking the instructions into multiple sentences allows the visual views and the guiding signals in sub-instructions to attend to each other during cross-modal encoding fully. Such cross-modal alignments lead to betters navigation performance.

A.4 Amazon Mechanical Turk

We use AMT for human evaluation when evaluating the quality of the instructions generated by different models. The survey form for head-to-head comparisons is shown in Figure 6.


Watch the gif that shows the change of the streetviews, then read the ground truth instruction that describes the navigation process.

Ground Truth: Go in the direction of traffic, with a bridge ahead of you. Go straight through intersection. (You may have to go right to go straight.) At the next intersection, see the bridge tunnel on the right. Go straight through the intersection. At the end of the road, with water in front of you, go in the intersection and turn to come back down the road you are on. When you get back to the bridge tunnel intersection, go right just far enough across the intersection to turn and go left in the far lane of traffic. There is a brick building to your right, and chain and post fences to your left. Go past the first yellow sign with black people, and stop next to the second yellow sign with black people.

A high-quality instruction should be **informative (address to the objects in the surrounding)**, and should be able to **provide correct guiding signals (turn left/right)**.

Please read the two instructions below, and select which instruction is more aligned to the ground truth instruction, and describes the streetview in the gif better.

A. Go in the direction of traffic, with the flow of traffic in front of you. Go right at the intersection. You'll have a lot of blue bikes on the right. Go to the next intersection and turn right. When you turn, there will be a green awning on the right and white and yellow umbrellas on the right. Go straight through the first intersection you come to, and at the second intersection, turn left. Go to the next intersection and turn left. Stop just past the crosswalk, and turn to face the building with the green awning, you'll see a building with red trim, and red doors.

B. Go in the direction that puts the green construction wall on your right. Go straight through the first intersection you come to. At the next intersection, turn right. Go to the next intersection and turn right again. Go straight through the next intersection. When you get to the next intersection, go straight through it. Stop just before the crosswalk of the next intersection.

Which instruction better describe the streetview? ○ A ○ B ○ Tie

Which instruction is more aligned with the ground truth? ○ A ○ B ○ Tie

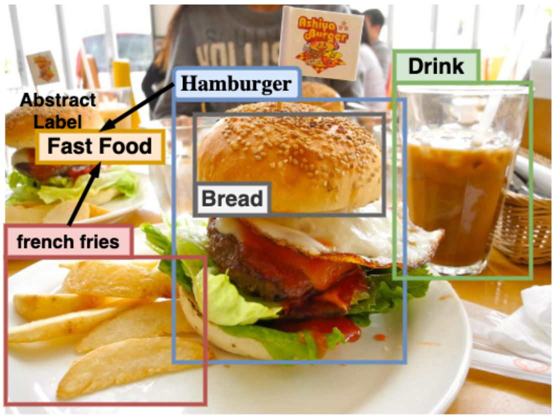
Figure 6: Pairwise comparison form for human evaluation on AMT.

`ECOL-R`: Encouraging Copying in Novel Object Captioning with Reinforcement Learning

Yufei Wang[1] and **Ian D. Wood**[1] and **Stephen Wan**[2] and **Mark Johnson**[3]
Macquarie University, Sydney, Australia[1]
CSIRO Data61, Sydney, Australia[2]
Oracle Digital Assistant, Oracle Corporation[3]
`yufei.wang@students.mq.edu.au, ian.wood@mq.edu.au`
`stephen.wan@data61.csiro.au`
`mark.mj.johnson@oracle.com`

Abstract

Novel Object Captioning is a zero-shot Image Captioning task requiring describing objects not seen in the training captions, but for which information is available from external object detectors. The key challenge is to select and describe all salient detected novel objects in the input images. In this paper, we focus on this challenge and propose the **`ECOL-R`** model (**E**ncouraging **C**opying of **O**bject **L**abels with **R**einforced Learning), a copy-augmented transformer model that is encouraged to accurately describe the novel object labels. This is achieved via a specialised reward function in the SCST reinforcement learning framework (Rennie et al., 2017) that encourages novel object mentions while maintaining the caption quality. We further restrict the SCST training to the images where detected objects are mentioned in reference captions to train the **`ECOL-R`** model. We additionally improve our copy mechanism via *Abstract Labels*, which transfer knowledge from known to novel object types, and a *Morphological Selector*, which determines the appropriate inflected forms of novel object labels. The resulting model sets new state-of-the-art on the *nocaps* (Agrawal et al., 2019) and *held-out* COCO (Hendricks et al., 2016) benchmarks.



Ⓐ **Two Hamburgers served with french fries and drinks.**

Ⓑ **Two Hamburgers served with bread on the plate.**

Ⓒ **Two Hamburgers served with french fries.**

Figure 1: Caption **A** is the ground-truth caption for the image. Compared with **B** and **C**, **A** is the best caption because it mentions all salient objects (i.e, *Hamburger*, *French Fries* and *Drinks*). We use Abstract Labels, that is hypernyms of the objects' detected object labels in the object representations, transferring knowledge from the objects seen in the training captions to novel objects. Our copy mechanism also selects appropriate inflected forms of object labels (i.e., *Hamburgers* vs. *Hamburger*).

1 Introduction

Novel Object Captioning is a zero-shot Image Captioning task where the captions should mention novel objects (i.e., not seen in the training captions), but for which information is available from external object detectors. To produce high-quality captions, the captioning models should select and describe all salient detected objects and avoid mentioning minor or irrelevant details in the input images. As shown in Figure 1, caption **A** is the best caption among the three because **A** mentions all salient objects in the images without any unnecessary details while **B** mentions *Bread* which is just a minor detail; and **C** misses the salient object *Drink*. This paper aims to develop a captioning model that produces caption **A**.

We use an advanced copy mechanism, similar to the one in Hu et al. (2020), to effectively integrate novel objects. We follow the setup in Agrawal et al. (2019) and use two object detectors: one providing rich object visual features and another providing task specific (including novel) object labels as copy candidates. Our preliminary experiments show that the copy mechanism is infrequently triggered and unable to mention many salient objects in the input images. We propose the **`ECOL-R`**

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1222–1234
April 19 - 23, 2021.

model (**E**ncouraging **C**opying of **O**bject **L**abels with **R**einforced Learning), a copy-augmented transformer model trained in the Self-Critical Sequence Training (SCST) framework (Rennie et al., 2017). SCST with a CIDEr reward (Vedantam et al., 2015) is a standard approach for training the captioning models (Anderson et al., 2018b), but this paper will show that it does not sufficiently encourage the model to use copy operations. We design a new reward function that provides a reward for each copy operation proportional to the caption quality. We further restrict the SCST training to the images that contain at least one word in the ground truth captions that corresponds to one of the detected object labels. With these innovations, the `ECOL-R` model outperforms a SCST baseline and a strong inference encouragement baseline by a large margin.

Our copy mechanism and caption generator incorporate two enhancements to better choose and incorporate novel objects: *a) Abstract Labels* which correspond to hypernyms of the object labels and facilitate knowledge transfer between objects appearing in training captions and novel objects; *b) a Morphological Selector* which determines the correct inflected form of the copied task specific object labels which is similar in purpose to that proposed in (Lu et al., 2018b).

We evaluate the `ECOL-R` model on the novel object captioning benchmark *nocaps* (Agrawal et al., 2019) and *held-out* COCO (Hendricks et al., 2016). The `ECOL-R` model achieves a new state of the art on both benchmarks and generalizes well to in-domain images.

2 Related Work

Popular Image Captioning models include LSTM-based (Anderson et al., 2018b) and Transformer-based decoders (Herdade et al., 2019; Cornia et al., 2020). The visual encoders are often neural object detectors (Anderson et al., 2018b; Wang et al., 2019) producing Region-of-Interest (ROI) vectors. To train the model to copy novel object labels, the Neural Baby Talk model (`NBT`) (Lu et al., 2018a) and follow-up work (Wu et al., 2018; Yao et al., 2017; Li et al., 2019) use copy mechanisms (Vinyals et al., 2015). The copying candidates are labels of salient objects produced by external object detectors. In this paper, we follow previous work by using the Visual Genome object detector from (Anderson et al., 2018b) as the visual feature extractor and a task specific object detector to provide object labels for copying.

These models are typically trained with the Cross-Entropy loss (CE). This creates a mismatch between the training and testing environments because the evaluation metrics are non-differentiable text-based measures (Ranzato et al., 2015). Self-Critical Sequence Training (SCST) (Rennie et al., 2017) was proposed to address this issue by directly optimizing the inference output using caption-level rewards, such as CIDEr-D (Vedantam et al., 2015).

There are two existing novel object captioning benchmarks: *a)* the *held-out* COCO Benchmark (Hendricks et al., 2016), constructed by excluding images containing one of eight selected object classes from the standard COCO 2014 benchmark, and *b) nocaps* (Agrawal et al., 2019), which uses the COCO 2017 benchmark for training and provides new validation and test images from the `Open Images` Dataset with over 400 novel objects. Both benchmarks are object-centric and there is no reliable benchmarks that systematically evaluate the quality of generated actions or attributes.

3 Model

Figure 2 provides an overview of the `ECOL-R` model. We refer to the `ECOL-R` model without SCST training as `ECOL`. We describe this model in Sec. 3.1 and our novel reinforced copy encouragement training in Sec. 3.2.

3.1 The `ECOL` Model

Input Image Objects: Following the setup in Agrawal et al. (2019), we use two object detectors: the Visual Genome object detector from Anderson et al. (2018b), producing image objects and regions G (represented by embedding vectors $[\boldsymbol{x}^g_1, \ldots, \boldsymbol{x}^g_{k^g}]$) with detailed visual features; and a task specific object detector, producing image objects F (represented by $[\boldsymbol{x}^f_1, \ldots, \boldsymbol{x}^f_{k^f}]$) and their corresponding labels $L_f = [l_1, \ldots, l_{k^f}]$ used as copy candidates during caption generation. We will introduce object representations $\boldsymbol{x}_i$ below and define them in Eq. 1.

Image Object Representations: Following Anderson et al. (2018b); Lu et al. (2018a), we represent both sets of objects with Region-Of-Interest (ROI, $\boldsymbol{r}_i \in \mathbb{R}^{2048}$) vectors from the Visual Genome object detector and object positional features ($\boldsymbol{p}_i \in \mathbb{R}^8$), including bounding box coordinates and size, and an object label confidence score. In addition,

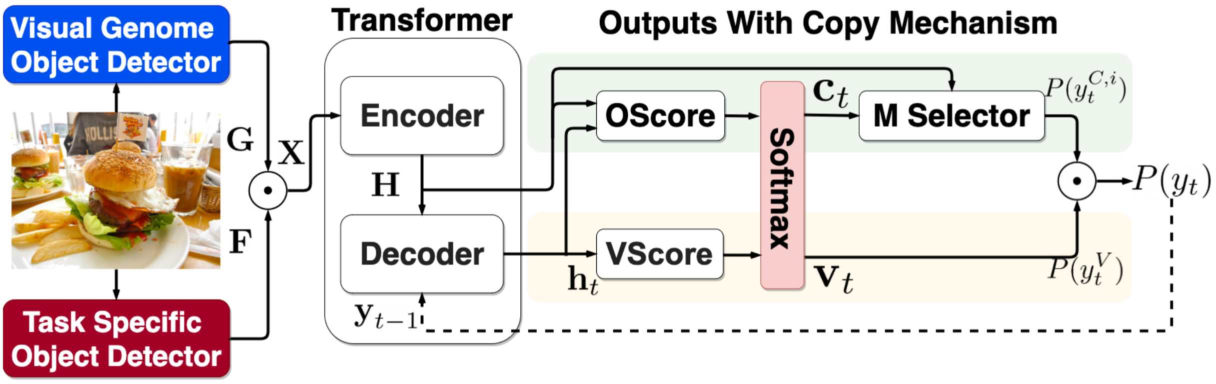


Figure 2: Overview of the **ECOL-R** Model. X is the concatenated Object representations G and F from the two object detectors. The Transformer encoder produces $\mathbf{H}$ and the decoder provides h_t at step t. We then estimate the probabilities for generating each vocabulary word (yellow box) and copying from each task specific image object (green box). The results are concatenated and jointly softmax (red box). We refine each copy probability into the concrete inflected word probability in $MSelector$. The final output $P(y_t)$ concatenates all above word probabilities.

to transfer knowledge from the seen objects to the novel ones, we propose *Abstract Labels* for the task specific objects, described below.

Abstract Labels: The task specific object detectors we use provide taxonomies of object classes, and every detected object is assigned a label from that taxonomy. More general object classes conceptually include all the labels lower in the taxonomy. [1] This provides us with a mechanism for associating class labels not present in the training data with those that do occur in the training data by mapping them to a common ancestor in the hierarchy. Inspired by Ciaramita and Johnson (2003), we define *Abstract Labels* to be a fixed set of ancestor class labels that spans the entire taxonomy (see Figure 3). Using the abstract labels to drive copy decisions allows the usage of known object types to inform the word generation of novel objects. Each object from the task specific detector is associated with its nearest abstract label ancestor. We choose the set of abstract labels such that the objects in the training data are evenly distributed across the set of abstract labels. We represent abstract labels with trainable embeddings $e_i \in \mathbb{R}^d$, where d is the hidden size of our base model. We use the `Open Images` V4 class hierarchy for the *nocaps* benchmark and a merged 8 coco super-categories hierarchy Lin et al. (2014) for the *held-out* COCO benchmark. The final representation for each object x_i is:

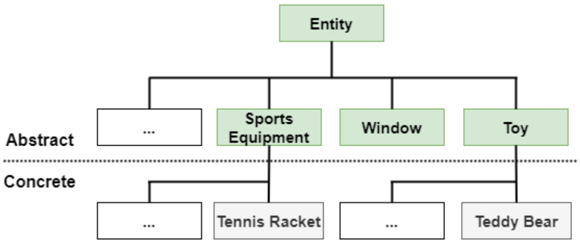


Figure 3: A part of the class hierarchy from the Open Images V4 Dataset (Kuznetsova et al., 2018). The green nodes are used as abstract object labels. For each label, its abstract label is its closest green ancestor.

$$x_i = LN(W_r r_i + e_i) + LN(W_p p_i) \quad (1)$$

where LN is layer normalization and $W_r \in \mathbb{R}^{d\times 2048}$, $W_p \in \mathbb{R}^{d\times 8}$ are trainable projections. The two sets of object representations are concatenated as $X = F \odot G$ where $\odot$ represents concatenation.

Transformer Base Model: We use a transformer model (Vaswani et al., 2017) with an N_{enc}-layer encoder and an N_{dec}-layer decoder (N_{enc} = N_{dec} = 3 in our experiments). We denote the encoder output $\mathbf{H} = Encoder(X)$. The decoder uses frozen word and positional embeddings WE and PE from GPT2 (Radford et al., 2019) which are helpful in producing better captions describing novel objects. In step t:

$$w_{1:t-1} = WE(y_{1:t-1}) + PE(y_{1:t-1}) \quad (2)$$

$$h_t = Decoder(\mathbf{H}, w_{1:t-1}) \quad (3)$$

where $y_{1:t-1}$ is the generation history and $h_t \in \mathbb{R}^d$.

---

[1] If the task specific object detector does not provide such a taxonomy, a suitable taxonomy could be obtained from sources such as Wordnet.

Outputs With Copy Mechanism: The `ECOL` model either generates words from the vocabulary or copies from task specific objects. We deploy a copy mechanism similar to the dynamic pointer network in Hu et al. (2020). Given the decoder output $\boldsymbol{h}_t$, we first calculate a raw score for each vocabulary word:

$$VScore(\boldsymbol{h}_t) = \boldsymbol{W}_e \boldsymbol{h}_t \tag{4}$$

where $\boldsymbol{W}_e \in \mathbb{R}^{|V| \times d}$, $|V|$ is the GPT2 vocabulary size. We then calculate raw additive attention scores over the encoder output of task specific image objects (i.e., $\mathbf{H}_{1:k^f}$):

$$OScore(\mathbf{H}, \boldsymbol{h}_t)_i = \boldsymbol{w}_c^T tanh(\boldsymbol{W}_f \mathbf{H}_i + \boldsymbol{W}_h \boldsymbol{h}_t) \tag{5}$$

where $i \in [1, k^f]$ and $\boldsymbol{W}_f \in R^{d \times d}$, $\boldsymbol{W}_h \in R^{d \times d}$ and $\boldsymbol{w}_c \in R^d$. Finally, we concatenate the raw scores from $VScore$ and $OScore$ and jointly softmax:

$$[\boldsymbol{v}_t, \boldsymbol{c}_t] = Softmax([VScore(\boldsymbol{h}_t) \odot OScore(\mathbf{H}, \boldsymbol{h}_t)]) \tag{6}$$

where $\odot$ represents concatenation. $\boldsymbol{v}_t$ provides probabilities for GPT2 vocabulary words and $\boldsymbol{c}_t$ provides probabilities for copying task specific object labels.

Morphological Selector: Object labels can appear in inflected forms in captions. For example, in Figure 1, after selecting the object *hamburger*, the `ECOL` model should generate *"hamburgers"* after *"Two"*. We propose a morphological selector (*M Selector*) to refine the copy probability of each task specific image object label l_i (i.e., $\boldsymbol{c}_{t,i}$) into the probabilities of generating all possible morphological forms $y_t^{l_i}$ (i.e., $P(y_t^{l_i}|l_i)$). Specifically, we use $\boldsymbol{h}_t$ to choose an inflected form from its possible inflected forms (e.g., Singular or Plural in English):

$$P(y_t^{l_i}|l_i) = \text{softmax}(\boldsymbol{W}_{l_i} \boldsymbol{h}_t) \tag{7}$$

Here $\boldsymbol{W}_{l_i} \in \mathbb{R}^{s_i \times d}$ where s_i is the number of inflected forms of label l_i (in most cases 2 for English, singular and plural). Finally, the `ECOL` model concatenates the above refined probabilities as follows:

$$P(y_t^v) = \boldsymbol{v}_t \tag{8}$$

$$P(y_t^{l_i}) = \boldsymbol{c}_{t,i} \cdot P(y_t^{l_i}|l_i) \tag{9}$$

$$P(y_t) = P(y_t^v) \odot P(y_t^{l_1}) \odot \cdots \odot P(y_t^{l_{kf}}) \tag{10}$$

where $\odot$ represents concatenation. Some novel object labels are included in the GPT2 vocabulary. However, these words are not present in the training captions and thus the model always assigns them very low probabilities in $P(y_t^v)$. The only way novel object labels can appear in captions is through copy operations.

Model Application Scope In this paper, we focus on the Novel Object Captioning task. However, in general, our copy mechanism is capable of copying any type of information. The Abstract Label approach is general to zero shot learning problems where novel items share characteristics with training instances. The Morphological Selector is also applicable to linguistic copy mechanisms in other contexts such as Commonsense Reasoning (Lin et al., 2020) where copied terms may require linguistic alignment with the generated text.

3.2 Copying More Object Labels

In this paper, we encourage the copying of object labels by using a suitable reward function in the Self-Critical Sequence Training (SCST) framework, which has proven effective for image captioning tasks. Compared with injecting additional loss terms together with the standard XE loss, using the SCST framework allows us to design arbitrary encouragement signals based on the inference output. It minimizes the negative expected reward score:

$$L_R(\theta) = -\mathbf{E}_{y_{1:T} \sim p_\theta}[r(y_{1:T})] \tag{11}$$

where r is the reward function and p_θ represents the models outputs. In this paper, following Cornia et al. (2020), we first pre-train the `ECOL` model with the CE loss, then switch to fine-tune the `ECOL` model with the above SCST loss.

Inference Bias Baseline: We add an Inference Bias (IB) $b \in \mathbb{R}^+$ to increase $P(y_t^{l_i})$ at inference time. Eq. 9 is changed to:

$$P(y_t^{l_i}) = b \cdot \boldsymbol{c}_{t,i} \cdot P(y_t^{l_i}|l_i) \tag{12}$$

and remaining probabilities normalised accordingly. IB is functionally equivalent to adjusting the threshold for the copy decision during inference. Surprisingly, this simple inference trick provides a strong baseline (see Table 3). This shows that after the CE training, many correct copy operations are assigned with low probabilities, compared to the fixed vocabulary items. However, we believe that it is better to train the model to increase the probabilities of these copy operations than adding ad hoc adjustments at inference time.

Can Standard SCST Encourage Copying? Rennie et al. (2017) shows that SCST with the CIDEr reward fine-tuning leads to noticeable caption quality improvement for standard image captioning tasks (i.e, improvement in various automatic metrics). Previous work (Cornia et al., 2020; Anderson et al., 2018b) use CIDEr as the standard reward function in their SCST optimization. This shows suggests that the problem of overfitting of SCST training with CIDEr reward is minimal. Intuitively, the CIDEr reward is positively correlated with the number of salient object label mentions and should encourage the model to copy salient novel object labels. However, CIDEr equally rewards both generation of object labels present in training data via the vocabulary $P(y_t^v)$ and via copy operations $P(y_t^{l_i})$. Novel objects labels however can only be generated by copy operations (see Sec. 3.1), thus the CIDEr reward function does not sufficiently encourage these operations. We propose two orthogonal modifications to the standard SCST algorithm to address this issue:

Novel Encouragement Reward: We propose combining the standard CIDEr-D reward with a reward function that gives captions with words copied from object labels an extra bonus, which we intend to encourage copy operations. One straightforward way to implement this idea is to provide a constant bonus to each triggered copy operation:

$$R_a(X) = \mathit{CIDEr}(X) + a * C \tag{13}$$

where X is a generated caption, C is the number of copy actions in the caption X and $a \in R^+$ is a fixed hyper-parameter. We refer this as *additive bias*. Optimizing with the additive bias, the captioning model only needs to trigger the copy operation for arbitrary objects at arbitrary generation steps. That is, the model may encourage copying object labels at the expense of caption quality (i.e., high CIDEr-D scores). Therefore, we propose a *proportional bias* that assigns different rewards to the copy operations in different images by making a connection between the copy bonus and the generated captions CIDEr-D score:

$$R_p(X) = \mathit{CIDEr}(X) * (1.0 + p * C) \tag{14}$$

where $p \in R^+$ is a fixed hyper-parameter. Although R_a can effectively encourage the model to copy objects, it may introduce noisy object mentions. R_p penalizes those noisy object mentions via the low caption CIDEr score.

Visual Object Aligned (VOA) Images: *VOA Images* refers to the set of training images where the reference captions contain at least one word from retained object labels. During SCST training, images that contain no object label words (i.e., non-VOA images) will not utilise copy operations, thus these images encourage the model NOT to copy. VOA images account for approximately 70% of the full COCO-2017 training images set. Although restricting training to VOA images can be done with arbitrary models, this may hurt the diversity of generated captions. Experiments in Table 3 confirm that restricting to VOA images only improves performance when used with SCST training.

Hyper-Parameters For Copy Encouragement: The above approaches introduce two additional parameters: a and p. In our experiments, a and p range over 0.2, 0.3 and 0.4; we found that 0.3 works the best for both reward functions. Combined with restricting SCST training to VOA images, R_p works better than R_a and sets a new SOTA for novel object image captioning.

4 Experiments

We conduct experiments on the *nocaps* (Agrawal et al., 2019) and the *held-out* COCO (Hendricks et al., 2016) Benchmark. We set the layer and embedding size to $d = 768$ and use Adam optimisation (Kingma and Ba, 2014). We train our models 15 epochs with batch size 100 for CE loss and 15 epochs with batch size 10 for SCST loss.

4.1 Evaluation Metrics

We use CIDEr (Vedantam et al., 2015), SPICE (Anderson et al., 2016a) and METEOR (Banerjee and Lavie, 2005) to evaluate the caption quality. CIDEr measures the similarity between the reference captions and generated outputs using tf-idf weighted n-gram overlap. SPICE is based on the scene graphs matching between the reference captions and generated outputs. METEOR focuses on the alignment between the words in reference captions and generated outputs, with an aim of 1:1 correspondence. To measure the effectiveness of our copy encouragement approach, we report object F1 (Anderson et al., 2017) in the *held-out* COCO Benchmark. As the *nocaps* benchmark does not release its ground-truth captions, we instead report averaged number of mentioned objects (Ave. O) and CIDEr score for dummy captions that only contain copied object words (Object CIDEr, OC., details see Appendix).

	in-domain		*near-domain*		*out-of-domain*		**Overall**		
Method	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	Meteor	CIDEr	SPICE
Up-Down + BS	73.7	11.6	57.2	10.3	30.4	8.1	22.9	54.5	10.1
Up-Down + ELMo + CBS	76.0	11.8	74.2	11.5	66.7	9.7	24.4	73.1	11.2
NBT + BS	62.8	10.3	51.9	9.4	48.9	8.4	21.8	54.3	9.4
NBT + CBS	61.9	10.4	57.3	9.6	61.8	8.6	21.6	59.9	9.5
$OSCAR_L$ + CBS + SCST	-	-	-	-	-	-	-	80.9	11.3
ECOL + IB (Ours)	81.7	12.9	77.2	12.1	67.0	10.3	25.6	76.0	11.9
ECOL-R (Ours)	**87.3**	**12.8**	**84.0**	**12.5**	**75.4**	**10.7**	**25.7**	**82.9**	**12.2**

Table 1: Comparison of the **ECOL-R** model with other state-of-the-art systems on the *nocaps* Test Split. The **ECOL-R** model sets new state of the art and improves previous work by 2.0 CIDEr and 0.9 SPICE. The performance of the **Up-Down** and **NBT** models are from Agrawal et al. (2019). The $OSCAR_L$ model is from Li et al. (2020).

	out-of-domain				*in-domain*		
Method	Meteor	CIDEr	SPICE	Object F1	Meteor	CIDEr	SPICE
LSTM-P (Li et al., 2019)	23.4	88.3	16.6	60.9	-	-	-
Base + CBS (Anderson et al., 2017)	23.3	77.0	15.9	54.0	24.9	88.0	18.4
NBT + CBS (Lu et al., 2018a)	24.1	86.0	17.4	70.5	25.0	92.1	18.0
ECOL + IB (Ours)	25.6	95.5	18.8	58.2	**27.0**	108.3	20.4
ECOL-R (Ours)	**25.7**	**99.2**	**19.3**	66.3	26.8	**113.3**	20.4
ECOL-R + CBS (Ours)	**25.7**	99.1	19.1	**71.8**	26.8	112.6	**20.8**
PS3 (Anderson et al., 2018a)§	25.4	94.5	17.9	63.0	25.9	101.1	19.0
FDM-net (Cao et al., 2020)§	25.9	84.8	19.4	64.7	27.2	109.7	20.2
FDM-net + CBS (Cao et al., 2020)§	25.6	85.3	19.6	85.7	26.2	105.5	19.7

Table 2: Comparison of the **ECOL-R** model with previous work on the test split of the *held-out* COCO Benchmark. Object F1 measures the presence of particular novel objects in the captions. § not comparable as they use additional information related to novel objects (i.e., images containing novel objects and scene graphs).

4.2 Comparison with the State-of-the-art

We compare our models **ECOL** + IB and **ECOL-R** with other state-of-the-art systems in Tables 1 and 2.

On the *nocaps* benchmark (Table 1), our models outperform previous work, including the recently proposed $OSCAR_L$ + CBS + SCST model (Li et al., 2020), which is fine-turned from the BERT-LARGE model (Devlin et al., 2019), by 2.0 CIDEr, 0.9 SPICE and set a new state of the art. Compared with the $OSCAR_L$ model, our models use far fewer model parameters (340M vs. 60M) and outperforms $OSCAR_L$ on both CIDEr and SPICE metrics. We train our model for about 10 hours for CE Loss and 24 hours for SCST Loss using a single Nvidia P100 GPU. As a comparison, the $OSCAR_L$ model which is fine-tuned from BERT-LARGE uses 60 - 90 hours for training CE Loss and 60 - 90 hours for training SCST Loss.[2] This shows that simply deploying a BERT-based language model is not sufficient for the Novel Object Captioning task.

On the *held-out* COCO benchmark (Table 2), the **ECOL-R** model produces more novel objects (+ 13.3 Object F1) and higher quality captions (+ 3.9 CIDEr on the *out-of-domain* split) than the **ECOL** model with run-time Inference Bias. Compared with previous work, the **ECOL-R** model achieves 10.9 CIDEr and 1.9 SPICE higher in the *out-of-domain* split, 21.2 CIDEr and 2.8 SPICE higher in the *in-domain* split with the highest object F1. This shows that our copy encouragement approach successfully trains our model to correctly copy more novel objects and to produce high-quality captions. Compared with PS3 (Anderson et al., 2018a)

---

[2] According to the authors' comments on their official model code repo https://github.com/microsoft/Oscar/issues/6

and FDM-net model (Cao et al., 2020) which are trained on extra images containing novel objects and scene graphs, our models still outperform the PS3 model and 13.9 CIDEr higher than the FDM-net. We set a new state of the art in this benchmark without additional novel objects information.

4.3 Ablation Study

Table 3 presents ablation results for various **ECOL-R** components, including our copy encouragement approach. Table 4 shows that our encouragement of copying in the **ECOL-R** model does not benefit from additional Inference Bias. Table 5 shows the effect of Abstract Labels and the Morphological Selector in the **ECOL-R** model. Finally, Table 6 confirms the **ECOL-R** model's generalization ability for in-domain COCO images.

ECOL-R Components: The **ECOL** model produces better captions using the frozen GPT2 parameters (row **1** vs. **2**). Our copy mechanism (C) helps the model to explicitly integrate novel objects, substantially improving the *out-of-domain* split by 15.3 CIDEr and 0.3 SPICE (row **2** vs. **3**). The Inference Bias (IB) introduces noticeable performance improvement: 8.4 CIDEr and 0.3 SPICE (row **3** vs. **4**) in models that do not use our reinforcement learning approach. The **ECOL** model trained with the standard SCST reward function obtains an overall 8.1 CIDEr improvement, but most of the improvement is from the *in-domain* and *near-domain* splits (row **8** vs. **6**). Compared with the **ECOL** + IB model, the **ECOL** model trained with standard SCST algorithm is 8.1 CIDEr lower in the *out-of-domain* split (row **5** vs. **4**). As discussed in Sec. 3.2, standard SCST cannot provide sufficient copy encouragement as object words can be generated from either pathways (fixed vocabulary or copy). Optimizing either the R_a or R_p reward functions improves the **ECOL** + CIDEr model by 7.0 CIDEr and 7.8 CIDEr respectively (row **7** and **5**). R_a achieves 3.7 CIDEr higher than R_p in the *out-of-domain* split. Interestingly, after restricting the model training to VOA images, R_p achieves 7.8 CIDEr improvement in the *out-of-domain* split (row **8** vs. **7**), outperforming the **ECOL** + R_a w/ VOA model by 1.4 CIDEr (row **10** vs. **8**).

Effectiveness of Copy Encouragement: We directly measure the copy quantity by counting the number of copied object labels and Object CIDEr. Row **5** and **3** confirm that the standard SCST algorithm has little impact on the copy quantity (only + 0.1 object per image and + 1.4 Object CIDEr). Inference Bias (IB), R_a and R_p rewards substantially improve the quantity of copied objects (row **4**, **6**, **7** vs. **3**). Among these three components, the models trained with R_a and R_p work better than the IB baseline (row **6**, **7** vs. **4**). The model trained with the R_a reward copies more objects than the R_p reward, especially training with all training images. This is because the R_a reward assigns constant positive reward for all copied objects. However, such a naive reward appears to encourage noisy copying operations (i.e., copying non-salient objects). As a result, the **ECOL** + R_a model performs worse than the **ECOL** + R_p model in terms of caption quality (row **7** vs. **6**). After restricting training with VOA images, the models trained with R_a and R_p copy similar amount of objects, but the model with R_a produce better captions than the one with R_a, especially in the *out-of-domain* split (row **10** vs. **8**). The R_p reward maintains a good balance between copying more objects and high caption quality.

Are The VOA images Always Useful? Restricting training to the VOA images can be done with any captioning models. However, this does not necessarily encourage copy operations and improve the output caption quality. When we restrict training to VOA images, the **ECOL-R** model performs consistently worse in all three splits compared to our proposed training scheme (row **9** vs. **10**). The only difference is that the **ECOL** model is not trained with diverse images during the cross-entropy stage. That is, restricting to VOA images is only suitable for fine-tuning in the SCST stage.

Sufficient Encouragement For Copy: Here we investigate whether our **ECOL-R** model mentions a sufficient number of salient objects. We apply an increasing amount of inference bias to the **ECOL**, **ECOL** + CIDEr and **ECOL-R** models in Table 4. We note that only **ECOL-R** model is negatively impacted (measured by CIDEr score) by different Inference Bias values. This shows that the **ECOL-R** model does not benefit from further copy encouragement.

The Effect of Abstract Labels and M Selector Table 5 shows the effect of Abstract Labels (AL) and the M Selector (M) in the **ECOL** + IB and **ECOL-R** models. Removing AL and M from the **ECOL** + IB model drops 2.3 CIDEr, 0.6 SPICE and 4.6 CIDEr, 0.8 SPICE respectively. AL and M have a large positive impact on the SPICE

Method	*in-domain* CIDEr	*in-domain* SPICE	*near-domain* CIDEr	*near-domain* SPICE	*out-of-domain* CIDEr	*out-of-domain* SPICE	`Overall` CIDEr	`Overall` SPICE	`Overall` OC.	`Overall` Ave. O
Ablation of ***ECOL*** *Components*										
(1) R + P + AL	80.3	12.7	60.3	11.4	34.8	9.2	58.0	11.2	-	-
(2) GPT2 + R + P + AL	78.6	12.5	64.2	11.6	39.4	9.3	61.3	11.3	-	-
(3) C + GPT2 + R + P + AL (**ECOL**)	80.7	12.7	68.6	11.8	61.0	10.2	68.8	11.7	10.3	0.5
Ablation of Copy Encouragement										
(4) ECOL + IB	85.3	12.8	76.6	12.2	73.7	10.8	77.2	12.0	17.5	1.1
(5) ECOL + CIDEr	89.7	12.3	75.9	11.8	63.5	10.1	75.3	11.5	11.7	0.6
(6) ECOL + R_a	89.0	12.6	82.2	12.5	78.2	11.2	82.3	12.3	19.7	1.5
(7) ECOL + R_p	92.3	12.9	83.5	12.5	74.9	11.0	83.0	12.3	17.7	1.3
(8) ECOL + R_a w/ VOA	89.3	12.6	83.6	12.5	81.7	11.1	84.0	12.2	20.7	1.6
(9) ECOL + R_p w/ VOA (all training)	88.0	12.6	83.3	12.6	79.4	11.2	83.2	12.3	20.6	1.5
(10) ECOL + R_p w/ VOA (**ECOL-R**)	90.4	12.6	84.5	12.4	83.1	11.0	85.1	12.2	20.4	1.5

Table 3: Ablation study of our model. *OC.* for Object CIDEr; *Ave. O* for Averaged Number of Mentioned Object in each image; *C* for Copy Mechanism; *R* for ROI; *P* for Position; *AL* for Abstract Label; R_a and R_p are the SCST reward function; *VOA*: Visual Object Aligned Images; *VOA (all training)*: all training using VOA images.


curtain
ostrich, deer
door, house

ECOL-R	A bathroom with a shower ***curtain*** and a toilet.✓	***An ostrich*** and a deer standing in a field. ✓	A red door of a red house with a red phone. ×
ECOL + IB	A white bath tub sitting next to a white toilet. ×	Two ostriches and a deer in a grassy field. ×	A red telephone booth sitting next to a brick wall. ✓
GT	The bathtub is white and has a white shower ***curtain***.	***An ostrich*** standing in grass with a few deer in the background.	A red phone booth is standing against a brick wall.

Figure 4: Three Examples generated by the **ECOL-R** and **ECOL** + IB model on the *nocaps* Val Set. First Case: The **ECOL-R** model accurately mentions the novel object *curtain* in the caption. Second Case: Both models talk about *ostrich* in their generated caption, but the **ECOL** + IB model uses the wrong modifier. Third Case: when detected object labels are too general, the **ECOL-R** model may produce inaccurate captions.

IB	1 (e^0)	2.72 (e^1)	7.39 (e^2)	20.01 (e^3)
ECOL	68.8	73.6	76.0	77.2
ECOL + CIDEr	75.3	77.6	79.5	81.1
ECOL-R	**85.1**	85.0	84.9	84.3

Table 4: The CIDEr Score in *nocaps* Validation Set with different Inference Bias.

score. As SPICE is sensitive to long-range object word relationships, such as attributes and predicate words, (Anderson et al., 2016a) Abstract Labels and the M Selector improve the semantic coherence and fluency of the captions. The performance gap in the **ECOL-R** model becomes smaller. Our copy encouragement approach contributes to the generation coherency and fluency.

Method	*out-of-domain* CIDEr	*out-of-domain* SPICE	`Overall` CIDEr	`Overall` SPICE
ECOL + IB	73.7	10.8	77.2	12.0
ECOL + IB w/o AL	72.7	10.0	74.9	11.4
ECOL + IB w/o M	71.2	10.5	72.6	11.2
ECOL-R	83.1	11.0	85.1	12.2
ECOL-R w/o AL	80.8	10.9	84.4	12.1
ECOL-R w/o M	78.4	10.7	82.8	11.9

Table 5: The contribution of Abstract Label (AL) and Morphological Selector (M).

Generalization For In-Domain COCO: To further show the generalization of our model for In-Domain images (i.e., without novel objects), we run the **ECOL** + IB and **ECOL-R** models on the COCO 2017 Validation Set and compare with another two

Method	Meteor	CIDEr	SPICE
NBT + CBS	25.1	80.2	15.8
Up-Down + CBS	25.7	95.4	18.2
ECOL + IB	27.1	106.1	20.6
ECOL-R	27.1	114.2	20.9

Table 6: The performance of the best four *nocaps* models on the COCO 2017 Validation Set.

novel object captioning models (**NBT** + CBS and **Up-Down** + ELMo + CBS) reported in Agrawal et al. (2019) in Table 6. Both of our models outperforms the **Up-Down** and **NBT** model by a large margin. Our models produce high-quality captions for images with novel objects as well as known objects.

4.4 Qualitative analysis on *nocaps*

Qualitative analysis on the *nocaps* validation set reveals that the **ECOL-R** model mentions the salient object in the input image (first example in Figure 4), is able to generate more accurate descriptions of novel objects (second example in Figure 4), however may generate inaccurate captions due to the non-informative detected object labels (third example in Figure 4). In summary, the **ECOL-R** model is better at incorporating detected image objects into generated captions than the **ECOL** + IB model.

5 Conclusion and Future work

This paper proposes the **ECOL-R** model that includes a training scheme to encourage copying novel object labels using Reinforced Learning. Our experiments show that the **ECOL-R** model successfully integrates novel object information and achieves state-of-the-art performance on two Novel Object Caption Benchmarks. In the future, we plan to extend our SCST reward function to other metrics such as SPICE (Anderson et al., 2016b) and BertScore (Zhang et al., 2020).

Acknowledgments

We thank anonymous reviewers for their insightful suggestions to improve this paper. This research was supported by a Google award through the Natural Language Understanding Focused Program, by a MQ Research Excellence Scholarship and a CSIRO's DATA61 Top-up Scholarship, and under the Australian Research Councils Discovery Projects funding scheme (project number DP160102156).

References

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016a. Spice: Semantic propositional image caption evaluation. In *ECCV*.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016b. Spice: Semantic propositional image caption evaluation. In *European Conference on Computer Vision*, pages 382–398. Springer.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, Copenhagen, Denmark. Association for Computational Linguistics.

Peter Anderson, Stephen Gould, and Mark Johnson. 2018a. Partially-supervised image captioning. In *Advances in Neural Information Processing Systems*, pages 1875–1886.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018b. Bottom-up and top-down attention for image captioning and visual question answering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Tingjia Cao, Ke Han, Xiaomei Wang, Lin Ma, Yanwei Fu, Yu-Gang Jiang, and Xiangyang Xue. 2020. Feature deformation meta-networks in image captioning of novel objects. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Massimiliano Ciaramita and Mark Johnson. 2003. Supersense tagging of unknown nouns in WordNet. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 168–175.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-Memory Transformer for Image Captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

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

Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond J. Mooney, Kate Saenko, and Trevor Darrell. 2016. Deep compositional captioning: Describing novel object categories without paired training data. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1–10.

Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. 2019. Image captioning: Transforming objects into words. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 11137–11147. Curran Associates, Inc.

Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. 2020. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. 2018. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. *ECCV*.

Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. 2019. Pointing novel objects in image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018a. Neural baby talk. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7219–7228.

Yujie Lu, Boyi Ni, Qijin Ji, Kotaro Sakamoto, Hideyuki Shibuki, and Tatsunori Mori. 2018b. Deep learning paradigm with transformed monolingual word embeddings for multilingual sentiment analysis. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong. Association for Computational Linguistics.

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

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.

Weixuan Wang, Zhihong Chen, and Haifeng Hu. 2019. Hierarchical attention network for image captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8957–8964.

Yu Wu, Linchao Zhu, Lu Jiang, and Yi Yang. 2018. Decoupled novel object captioner. In *2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018*, pages 1029–1037.

Deshraj Yadav, Rishabh Jain, Harsh Agrawal, Prithvijit Chattopadhyay, Taranjeet Singh, Akash Jain, Shiv Baran Singh, Stefan Lee, and Dhruv Batra. 2019. Evalai: Towards better evaluation systems for ai agents. *arXiv preprint arXiv:1902.03570*.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2017. Incorporating copying mechanism in image captioning for learning novel objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6580–6588.

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

Appendices

A Model Details

The hyper-parameters of the **`ECOL-R`** model is shown in Table 7. This architecture is basically from (Cornia et al., 2020). We only change the hidden size of the model to 768 to fit the size of GPT2 (the smallest version). Our model has total 60.8×10^6 parameters and 43.0×10^6 trainable parameters. This scale is slightly smaller than the Transformer Base model (65.8×10^6) (Vaswani et al., 2017).

We optimise with Adam(α=0.9, β=0.98, ϵ=1e-9) (Kingma and Ba, 2014) and clip gradients to 0.1 for both Benchmarks. In Cross-entropy training, we vary the learning rate over the course of training using the heuristic:

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(S^{-0.5}, S \cdot W^{-1.5}) \quad (15)$$

where S is the step number and W is the number of warm-up steps. We set W to 20000 steps for the *nocaps* Benchmark and 10000 steps for the *held-out* COCO Benchmark. The number of warm-up steps has some impact on both benchmark. We tried 20,000 and 10,000 for both Benchmarks. For SCST training, we set the initial learning rate $1e^{-6}$ and reduce it by half if the reward metric (Validatoin set CIDEr) does not improve for 3 evaluations. We conduct evaluation every 3000 steps.

We use Pytorch *1.4.0* to implement our model. The Cross-Entropy Training takes about 8 hours and the SCST optimization takes about 15 hours in a single NVIDIA Tesla P100 GPU.

Our source code is submitted in the *Software*. We setup an anonymous Google Drive to host large file [3].

Name	Value	Name	Value
N_e	3	Dropout	0.1
N_d	3	Max Cap. Len.	20
Model Size	768	beam Size	5
FFN Size	2048	Att. Head Num	8

Table 7: Hyper-parameters of the **`ECOL-R`** model.

A.1 Input Object Detector

We follow the processing of input objects in Agrawal et al. (2019). We observed that some

---

[3] `https://drive.google.com/drive/folders/1ETοBXQ8WAWxn5uCd38HtfRYmchnBBbMo?usp=sharing`

object categories are frequently mentioned in the training captions and that they often have variable, context-sensitive verbalisation (e.g., a person might be described as a sports player, a student, etc., depending on the context). For those objects, vocabulary based word generation often did a better job at selecting the correct verbalisation due to their frequency in training captions. On the other hand, novel objects typically have lower-frequencies and a fixed, single verbalisation. For example, elephants are usually only referred to with the word *elephant*. For this reason, we remove objects with high-frequency in training captions from the output of the task specific object detector, leaving their corresponding words to be generated via vocabulary softmax. We also remove the more abstract objects (higher in the object class hierarchy) when object regions overlap. Finally, we keep only one detected object for each label (the one with highest confidence score).

We provide the downloadable link of filtered results in Sec B. We use exactly the same Visual Genome objects as described in Anderson et al. (2018b). The Visual Genome object detector (Anderson et al., 2018b) can produce ROI vectors for arbitrary bounding boxes, hence we use it also to produce ROI vectors for objects from the task specific detector.

A.2 `ECOL-R` Inference Details

We use Beam Search with beam size 5 to decode the captions. We first do length normalization for the overall score of each decoded caption. We also penalize captions when they generate repeated bi-grams. Once the repetitions are found, the log-probability for that particular word is divided by a penalty value e^2. All image objects are only allowed to be copied once. During the SCST optimization, we mask out words from the vocabulary that can be generated via copy operations to encourage the model to copy. All the above constraints are applied to all of our models in the ablation study.

A.3 Object CIDEr Details

Object CIDEr score for dummy captions that only contain copied object words. This shows the correctness of our copy mechanism. High Object CIDEr score means many of the copied object labels are also mentioned in the ground-truth captions. We use this metrics because the *nocaps* benchmark does not release its ground-truth captions and only provide online evaluation APIs.

B Dataset Details

For the *nocaps* Benchmark, we train with the COCO-2017 dataset, which is available at `http://images.cocodataset.org/zips/train2017.zip`. The *nocaps* Validation and Test datasets are available from `https://nocaps.org/download`. The Visual Genome image object detection files can be found in `https://github.com/nocaps-org/updown-baseline`. For the *held-out* COCO Benchmark, the training and evaluation data can be found in `https://github.com/LisaAnne/DCC`. The Visual Genome image object detector is used for both benchmarks because COCO-2017 and COCO-2014 share the same set of images. The anonymous Google Drive includes the above data and the sets of task specific objects detected for the above two benchmarks.

B.1 Duplicated Caption Removal

We find some images in COCO share exactly the same reference captions. We find it beneficial to remove those duplicates. We simply iterate over all reference captions and remove any captions if they have already been found previously. This removes 25463 captions from the training data of the *nocaps* Benchmark and 7059 captions from the training data of the *held-out* COCO Benchmark.

B.2 VOA (visual object aligned) Images

VOA (visual object aligned) images/reference caption pairs are those that mention at least one detected task specific image object label (or their linguistic variant). Non-VOA image/caption pairs are removed from our SCST training process. We provide the reduced set of reference captions in the anonymous Google Drive (*ddc_captions/ddc_train_VOA.json* and *nocaps_captions/nocaps_train_VOA.json*).

B.3 Data Statistics for Benchmarks

Table 9 and Table 10 show the number of images and annotated reference captions of the *nocaps* and *held-out* COCO Benchmark, respectively. On average, each image has five reference captions. The COCO Train in the *nocaps* Benchmark is larger than the *held-out* COCO Benchmark.

C Evaluation

The *nocaps* Benchmark hosts its evaluation sever at `https://evalai.cloudcv.org/web/`

	out-of-domain				*in-domain*		
Method	Meteor	CIDEr	SPICE	Object F1	Meteor	CIDEr	SPICE
ECOL + IB (Ours)	25.1	93.0	18.7	58.4	26.9	107.1	20.3
ECOL-R (Ours)	25.6	99.2	19.5	64.5	26.7	113.0	20.3
ECOL-R + CBS (Ours)	25.6	98.4	19.3	70.9	26.7	112.3	20.2

Table 8: The performance of **ECOL** + IB, **ECOL-R** and **ECOL-R** + CBS model on *held-out* COCO Benchmark Validation Set.

	COCO Train	COCO Train VOA	COCO Val	*nocaps* Val	*nocaps* Test
#Image	118,287	82,771	5,000	4,500	10,600
#Caption	591,753	299,502	25,014	-	-

Table 9: Data Statistics for the *nocaps* Benchmark. The full set of annotations in *nocaps* Val and Test is not available. One can only access some of them via `https://nocaps.org/explore`.

	Train	Train VOA	Val in-domain	Val out-domain	Test in-domain	Test out-domain
#Image	70,194	55,799	17,234	3,018	17,288	3,024
#Caption	351,134	197,061	86,230	1,5105	86,188	15,131

Table 10: Data Statistics for the *held-out* COCO Benchmark.

`challenges/challenge-page/355/overview`. The detailed setup instruction of the local submission environment to Evai (Yadav et al., 2019) is available at `https://github.com/nocaps-org/updown-baseline`. The *held-out* COCO Benchmark uses the evaluation tool from `https://github.com/ruotianluo/coco-caption/tree/ea20010419a955fed9882f9dcc53f2dc1ac65092`. We provide an on-the-shelf version of this tool in the anonymous Google Drive (in *tools*).

C.1 *held-out* COCO Benchmark Validation Performance

We only show the test performance on the *held-out* COCO Benchmark in our main paper. Here, we show the performance of our model performance on the validation Set in Table 8. The models achieve similar level of performance on the Validation Set.

Enriching Non-Autoregressive Transformer with Syntactic and Semantic Structures for Neural Machine Translation

Ye Liu[1], Yao Wan[2], Jian-Guo Zhang[1], Wenting Zhao[1], Philip S. Yu[1]
[1]Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA
[2]School of Computer Sci. & Tech., Huazhong University of Science and Technology, Wuhan, China
{yliu279, jzhan51, wzhao41, psyu}@uic.edu, wanyao@hust.edu.cn

Abstract

The non-autoregressive models have boosted the efficiency of neural machine translation through parallelized decoding at the cost of effectiveness, when comparing with the autoregressive counterparts. In this paper, we claim that the syntactic and semantic structures among natural language are critical for non-autoregressive machine translation and can further improve the performance. However, these structures are rarely considered in existing non-autoregressive models. Inspired by this intuition, we propose to incorporate the explicit syntactic and semantic structures of languages into a non-autoregressive Transformer, for the task of neural machine translation. Moreover, we also consider the intermediate latent alignment within target sentences to better learn the long-term token dependencies. Experimental results on two real-world datasets (i.e., WMT14 En-De and WMT16 En-Ro) show that our model achieves a significantly faster speed, as well as keeps the translation quality when compared with several state-of-the-art non-autoregressive models.

1 Introduction

Recently, non-autoregressive models (Gu et al., 2018), which aim to enable the parallel generation of output tokens without sacrificing translation quality, have attracted much attention. Although the non-autoregressive models have considerably sped up the inference process for real-time neural machine translation (NMT) (Gu et al., 2018), their performance is considerably worse than that of autoregressive counterparts. Most previous works attribute the poor performance to the inevitable conditional independence issue when predicting target tokens, and many variants have been proposed to solve it. For example, several techniques in non-autoregressive models are investigated to mitigate the trade-off between speedup and performance, including iterative refinement (Lee et al., 2018), insertion-based models (Chan et al., 2019; Stern et al., 2019), latent variable based models (Kaiser et al., 2018; Shu et al., 2020), CTC models (Libovický and Helcl, 2018; Saharia et al., 2020), alternative loss function based models (Wei et al., 2019; Wang et al., 2019; Shao et al., 2020), and masked language models (Ghazvininejad et al., 2019, 2020).

Although these works have tried to narrow the performance gap between autoregressive and non-autoregressive models, and have achieved some improvements on machine translation, the non-autoregressive models still suffer from syntactic and semantic limitations. That is, the translations of non-autoregressive models tend to contain incoherent phrases (e.g., repetitive words), and some informative tokens on the source side are absent. It is because in non-autoregressive models, each token in the target sentence is generated independently. Consequently, it will cause the multimodality issue, i.e., the non-autoregressive models cannot model the multimodal distribution of target sequences properly (Gu et al., 2018).

One key observation to mitigate the syntactic and semantic error is that source and target translated sentences follow the same structure, which can be reflected from Part-Of-Speech (POS) tags and Named Entity Recognition (NER) labels. Briefly, POS, which aims to assign labels to words to indicate their categories by considering the long-distance structure of sentences, can help the model learn the syntactic structure to avoid generating the repetitive words. Likewise, NER, which discovers the proper nouns and verbs of sentences, naturally helps the model recognize some meaningful semantic tokens that may improve translation quality. This observation motivates us to leverage the syntactic as well as semantic structures of natural language to improve the performance of non-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1235–1244
April 19 - 23, 2021.

Table 1: A motivating example on WMT14 En→De dataset. English with POS|NER and its corresponding German translation with POS|NER. The Blue labels show the same tags, while the Red labels show the different tags in two languages.

EN:	A	republican	strategy	to	counter	the	rel-election	of	Obama	.
	|	|	|	|	|	|	|	|	|	|
EN POS:	DET	ADJ	NOUN	PART	VERB	DET	NOUN	ADP	PROPN	PUNCT
EN NER:	O	B_NORP	O	O	O	O	O	O	B_PERSON	O
DE:	Eine	republikanische	strategie	gegen	die	wiederwahl	Obama	.		
	|	|	|	|	|	|	|	|		
DE POS:	DET	ADJ	NOUN	ADP	DET	NOUN	PROPN	PUNCT		
EN NER:	O	B_NORP	O	O	O	O	B_PERSON	O		

autoregressive NMT. We present a motivating example in Table 1 to better illustrate our idea. From this table, we can find that although the words are altered dramatically from the English sentence to its German translation, the corresponding POS and NER tags still remain similar. For example, most POS tags are identical and follow the same pattern, except that PART, VERB, and ADP in the English do not match the German ADP, while the NER tags are exactly the same in both sentences.

In this paper, we propose an end-to-end Syntactic and semantic structure-aware Non-Autoregressive Transformer model (**SNAT**) for NMT. We take the structure labels and words as inputs of the model. With the guidance of extra sentence structural information, the model greatly mitigates the multimodality issue's negative impact. The core contributions of this paper can be summarized as that we propose 1) a syntax and semantic structure-aware Transformer which takes sequential texts and the structural labels as input and generates words conditioned on the predicted structural labels, and 2) an intermediate alignment regularization which aligns the intermediate decoder layer with the target to capture coarse target information. We conduct experiments on four benchmark tasks over two datasets, including WMT14 En→De and WMT16 En→Ro. Experimental results indicate that our proposed method achieves competitive results compared with existing state-of-the-art non-autoregressive and autoregressive neural machine translation models, as well as significantly reduces the decoding time.

2 Background

Regardless of its convenience and effectiveness, the autoregressive decoding methods suffer two major drawbacks. One is that they cannot generate multiple tokens simultaneously, leading to inefficient use of parallel hardware such as GPUs. The other is that beam search has been found to output low-quality translation with large beam size and deteriorates when applied to larger search spaces. However, non-autoregressive transformer (NAT) could potentially address these issues. Particularly, they aim at speeding up decoding through removing the sequential dependencies within the target sentence and generating multiple target tokens in one pass, as indicated by the following equation:

$$P_{\mathbf{NAT}}(\mathbf{y}|\mathbf{x};\phi)=\prod_{t=1}^{m} p\left(y_t|\hat{\mathbf{x}},\mathbf{x};\phi\right), \qquad (1)$$

where $\hat{\mathbf{x}}=\{\hat{x}_1,\ldots,\hat{x}_m\}$ is the copied source sentence. Since the conditional dependencies within the target sentence (y_t depends on $y_{<t}$) are removed from the decoder input, the decoder is unable to leverage the inherent sentence structure for prediction. Hence the decoder is supposed to figure out such target-side information by itself given the source-side information during training. This is a much more challenging task compared to the autoregressive counterparts. From our investigation, we find the NAT models fail to handle the target sentence generation well. It usually generates repetitive and semantically incoherent sentences with missing words. Therefore, strong conditional signals should be introduced as the decoder input to help the model better learn internal dependencies within a sentence.

3 Methodology

In this section, we present our model **SNAT** to incorporate the syntactic and semantic structure information into a NAT model as well as an intermediate latent space alignment within the target. Figure 1 gives an overview of the network structure of our proposed **SNAT**. In **SNAT**, the input sequence is segmented into sub-words by byte-pair

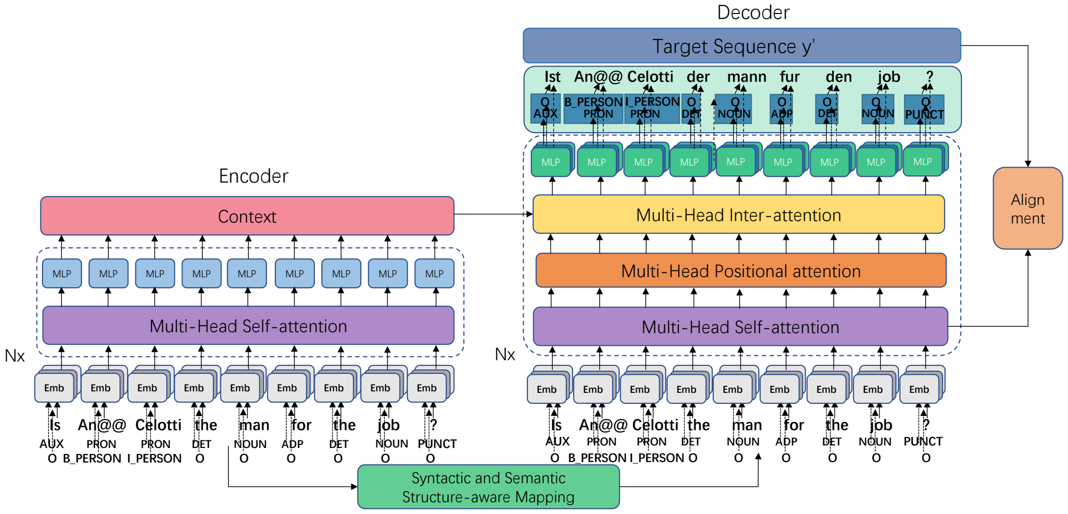


Figure 1: An overview of the proposed **SNAT** for neural machine translation.

tokenizer (Sennrich et al., 2016). In parallel, words in the input sequence are passed to POS and NER annotators to extract explicit syntactic and semantic structures, and the corresponding embeddings are aggregated by a linear layer to form the final syntax and semantic structure-aware embedding. The **SNAT** model copies the structured encoder input as the decoder input and generates the translated sentences and labels.

One of the most important properties of **SNAT** is that it naturally introduces syntactic and semantic information when taking the structure-aware information as inputs and generating both words and labels. More precisely, given a source sentence $\mathbf{x}$, as well as its label sequence $\mathbf{L_x}$, the conditional probability of a target translation $\mathbf{y}$ and its label sequence $\mathbf{L_y}$ is:

$$P_{\mathbf{SNAT}}(\mathbf{y}, \mathbf{L_y}|\mathbf{x}, \mathbf{L_x}; \varphi) = \prod_{t=1}^{m} p\left(y_t, L_{y_t}|\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L_x}; \varphi\right), \quad (2)$$

where $\mathbf{x}$ and $\mathbf{L_x}$ are first fed into the encoder of **SNAT** model. $\hat{\mathbf{x}}$ and $\hat{\mathbf{L}}_{\mathbf{x}}$ with length m are syntactic and semantic structure-aware copying of word and label from encoder inputs, respectively. We show the details in the following sections.

3.1 Syntactic and Semantic Labeling

We use POS and NER to introduce the syntactic and semantic information existing in natural language, respectively. During the data pre-processing, each sentence is annotated into a semantic sequence using an open-source pre-trained semantic annotator. In particular, we take the Treebank style (Marcus et al., 1999) for POS and PropBank style (Palmer et al., 2005) for NER to annotate every token of input sequence with semantic labels. Given a specific sentence, there would be predicate-argument structures. Since the input sequence is segmented into subword units using byte-pair encoding (Sennrich et al., 2016), we assign the same label to all subwords tokenized from the same word. As shown in Figure 1, the word "*Ancelotti*" is tokenized as "*An@@*" and "*Celotti*". The corresponding POS tags are PRON and PRON while the corresponding NER tags are B_PERSON and I_PERSON. For the text "*Is An@@ Celotti the man for the job ?*", the corresponding POS tag set is $\{$AUX, PRON, PRON, DET, NOUN, ADP, DET, NOUN, PUNCT$\}$ and the NER tag set is $\{$O, B_PERSON, I_PERSON, O, O, O, O, O, O$\}$. The data flow of the proposed model is also shown in Figure 1.

3.2 Encoder

Following Transformer (Vaswani et al., 2017), we use a stack of 6 identical Transformer blocks as encoder. In addition to the word embedding and position embedding in the traditional Transformer, we add structure-aware label embedding. The input to the encoder block is the addition of the normalized word, labels (NER and POS) and position embedding, which is represented as $\mathbf{H}^0 = [\mathbf{h}_1^0, \ldots, \mathbf{h}_n^0]$.

The input representation $\mathbf{H}^0 = [\mathbf{h}_1^0, \ldots, \mathbf{h}_n^0]$ is encoded into contextual layer representations through the Transformer blocks. For each layer, the layer representation $\mathbf{H}^l = [\mathbf{h}_1^l, \ldots, \mathbf{h}_n^l]$ is computed by the l-th layer Transformer block $\mathbf{H}^l = \text{Transformer}_l(\mathbf{H}^{l-1})$, $l \in \{1, 2, \ldots, 6\}$. In each

Transformer block, multiple self-attention heads are used to aggregate the output vectors of the previous layer. A general attention mechanism can be formulated as the weighted sum of the value vector $\mathbf{V}$ using the query vector $\mathbf{Q}$ and the key vector $\mathbf{K}$:

$$\mathrm{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{\text{model}}}}\right) \cdot \mathbf{V}, \tag{3}$$

where d_{model} represents the dimension of hidden representations. For self-attention, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are mappings of previous hidden representation by different linear functions, i.e., $\mathbf{Q} = \mathbf{H}^{l-1}\mathbf{W}_Q^l$, $\mathbf{K} = \mathbf{H}^{l-1}\mathbf{W}_K^l$, and $\mathbf{V} = \mathbf{H}^{l-1}\mathbf{W}_V^l$, respectively. At last, the encoder produces a final contextual representation $\mathbf{H}^6 = [\mathbf{h}_1^6, \ldots, \mathbf{h}_n^6]$, which is obtained from the last Transformer block.

3.3 Decoder

The decoder also consists of 6 identical Transformer blocks, but with several key differences from the encoder. More concretely, we denote the contextual representations in the i-th decoder layer is $\mathbf{Z}^i (1 \leq i \leq 6)$. The input to the decoder block as $\mathbf{Z}^0 = [\mathbf{z}_1^0, \ldots, \mathbf{z}_m^0]$, which is produced by the addition of the word, labels (NER and POS) copying from encoder input and positional embedding.

For the target side input $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}]$, most previous works simply copied partial source sentence with the length ratio $\frac{n}{m}$ where n refers to the source length and m is the target length as the decoder input. More concretely, the decoder input y_i at the i-th position is simply a copy of the $\lfloor \frac{n}{m} \times i \rfloor$th contextual representation, i.e., $x_{\lfloor \frac{n}{m} \times i \rfloor}$ from the encoder. From our investigation, in most cases, the gap between source length and target length is relatively small (e.g. 2). Therefore, it deletes or duplicates the copy of the last a few tokens of the source. If the last token is meaningful, the deletion will neglect important information. Otherwise, if the last token is trivial, multiple copies will add noise to the model.

Instead, we propose a syntactic and semantic structure-aware mapping method considering the POS and NER labels to construct the decoder inputs. Our model first picks out the informative words with NOUN and VERB POS tags, and the ones recognized as entities by the NER module. If the source length is longer than the target length, we retain all informative words, and randomly delete the rest of the words. On the other hand, if the source length is shorter than the target, we retain all words and randomly duplicate the informative words. The corresponding label of a word is also deleted or preserved. Moreover, by copying the similar structural words from the source, it can provide more information to the target input than just copying the source token, which is greatly different from the target token. The POS and NER labels of those structure-aware copied words from the source sentence are also copied as the decoder input. So by using the structure-aware mapping, we can assign $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}]$ as decoder input.

For positional attention which aims to learn the local word orders within the sentence (Gu et al., 2018), we set positional embedding (Vaswani et al., 2017) as both $\mathbf{Q}$ and $\mathbf{K}$, and hidden representations of the previous layer as $\mathbf{V}$.

For inter-attention, $\mathbf{Q}$ refers to hidden representations of the previous layer, whereas $\mathbf{K}$ and $\mathbf{V}$ are contextual vectors $\mathbf{H}^6$ from the encoder. We modify the attention mask so that it does not mask out the future tokens, and every token is dependent on both its preceding and succeeding tokens in every layer. Therefore, the generation of each token can use bi-directional attention. The position-wise Feed-Forward Network (FFN) is applied after multi-head attention in both encoder and decoder. It consists of two fully-connected layers and a layer normalization (Ba et al., 2016). The FFN takes $\mathbf{Z}^6$ as input and calculates the final representation $\mathbf{Z}^f$, which is used to predict the whole target sentence and label:

$$p\left(\mathbf{y} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}\right) = f\left(\mathbf{Z}^f \mathbf{W}_w^\top + \mathbf{b}_w\right), \tag{4}$$

$$q\left(\mathbf{L}_{\mathbf{y}} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}\right) = f\left(\mathbf{Z}^f \mathbf{W}_l^\top + \mathbf{b}_l\right), \tag{5}$$

where f is a GeLU activation function (Hendrycks and Gimpel, 2016). $\mathbf{W}_w$ and $\mathbf{W}_l$ are the token embedding and structural label embedding in the input representation, respectively. We use different FFNs for POS and NER labels. To avoid redundancy, we just use $q\left(\mathbf{L}_{\mathbf{y}} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}\right)$ to represent the two predicted label likelihood in general.

3.4 Training

We use ($\mathbf{x}$, $\mathbf{L}_{\mathbf{x}}$, $\mathbf{y}^*$, $\mathbf{L}_{\mathbf{y}}^*$) to denote a training instance. To introduce the label information, our proposed **SNAT** contains a discrete sequential latent variable $L_{y_{1:m}}$ with conditional posterior distribution $p(L_{y_{1:m}} | \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi)$. It can be approximated using a proposal distribution $q(\mathbf{L}_{\mathbf{y}} \mid$

$\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}})$. The approximation also provides a variational bound for the maximum log-likelihood:

$$\begin{aligned}
&\log P_{\textbf{SNAT}} = \log \sum_{t=1}^{m} q\left(L_{y_t} | \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right) \\
&\times p\left(y_t | L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right) \\
&\geq \underset{L_{y_{1:m}} \sim q}{\mathbb{E}} \Bigg\{ \underbrace{\sum_{t=1}^{m} \log q\left(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right)}_{\text{Label likelihood}} \\
&+ \underbrace{\sum_{t=1}^{m} \log p\left(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi\right)}_{\text{Structure-aware word likelihood}} \Bigg\} + \mathcal{H}(q).
\end{aligned} \tag{6}$$

Note that, the resulting likelihood function, consisting of the two bracketed terms in Eq. (6), allows us to train the entire model in a supervised fashion. The inferred label can be utilized to train the label predicting model q and simultaneously supervise the structure-aware word model p. The label loss can be calculated by the cross-entropy $\mathcal{H}$ of $L_{y_t}^*$ and Eq. (5):

$$\mathcal{L}_{\text{label}} = \sum_{t=1}^{m} \mathcal{H}\left(L_{y_t}^*,\ q(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}})\right), \tag{7}$$

The structure-aware word likelihood is conditioned on the generation result of the label. Since the Eq. (4) does not depend on the predicted label, we propose to bring the structure-aware word mask $\mathbf{M}_{wl} \in \mathbb{R}^{|V_{word}| \times |V_{label}|}$, where $|V_{word}|$ and $|V_{label}|$ are vocabulary sizes of word and label, respectively. The mask $\mathbf{M}_{w_l}$ is defined as follows:

$$\mathbf{M}_{w_l}(i,j) = \begin{cases} 1, & \mathcal{A}(y_i) = label_j, \\ \epsilon, & \mathcal{A}(y_i) \neq label_j, \end{cases} \tag{8}$$

which can be obtained at the preprocessing stage, and $\mathcal{A}$ denotes the open-source pre-trained POS or NER annotator mentioned above. It aims to penalize the case when the word y_i does not belong to the label $label_j$ with ϵ, which is a small number defined within the range of $(0, 1)$ and will be tuned in our experiments. For example, the word "great" does not belong to VERB. The structure-aware word likelihood can be reformulated as:

$$\begin{aligned}
&p(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi) = p(y_t \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}) \\
&\times \mathbf{M}_{w_l} \times q(L_{y_t} \mid \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}).
\end{aligned} \tag{9}$$

Consequently, the structure-aware word loss $\mathcal{L}_{\text{word}}$ is defined as the cross-entropy between true $p^{'}(y_t^*|L_{y_t}^*)$ and predicted $p(y_t \mid L_{y_t}, \hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}, \mathbf{x}, \mathbf{L}_{\mathbf{x}}; \varphi)$, where $p^{'}(y_t^*|L_{y_t}^*) \in \mathbb{R}^{|V_{word}| \times |V_{label}|}$ is a matrix where only item at the index of $(y_t^*, L_{y_t}^*)$ equals to 1, otherwise equals to 0. We reshape $p^{'}(y_t^*|L_{y_t}^*)$ and $p(y_t|L_{y_t})$ to vectors when calculating the loss.

Intermediate Alignment Regularization One main problem of NAT is that the decoder generation process does not depend on the previously generated tokens. Based on the bidirectional nature of **SNAT** decoder, the token can depend on every token of the decoder input. However, since the input of decoder $[\hat{\mathbf{x}}, \hat{\mathbf{L}}_{\mathbf{x}}]$ is the duplicate of encoder input $[\mathbf{x}, \mathbf{L}_{\mathbf{x}}]$, the generation depends on the encoder tokens rather than the target $\mathbf{y}^*$.

To solve this problem, we align the output of the intermediate layer of the decoder with the target. The alignment makes the generation of following layers dependent on the coarse target-side information instead of the mere encoder input. This alignment idea is inspired by (Guo et al., 2019), which directly feeds target-side tokens as inputs of the decoder by linearly mapping the source token embeddings to target embeddings. However, using one FFN layer to map different languages to the same space can hardly provide promising results. Thus, instead of aligning the input of decoder with the target, we use the intermediate layer of decoder to align with the target. In this case, our model avoids adding additional training parameters and manages to train the alignment together with **SNAT** model in an end-to-end fashion. Formally, we define the intermediate alignment regularization as cross-entropy loss between the predicted word and the true word:

$$\mathcal{L}_{\text{reg}} = \sum_{t=1}^{m} \mathcal{H}\left(y_t^*,\ \text{FFN}(\mathbf{Z}_t^{md})\right), \tag{10}$$

where $\mathbf{Z}^{md}$ $(1 < md < 6)$ represents the output of each intermediate layer. Consequently, the final loss of **SNAT** can be represented with the coefficient λ as:

$$\mathcal{L}_{\textbf{SNAT}} = \mathcal{L}_{\text{word}} + \mathcal{L}_{\text{label}} + \lambda \mathcal{L}_{\text{reg}}. \tag{11}$$

4 Experiment

In this section, we conduct experiments to evaluate the effectiveness and efficiency of our proposed model, with comprehensive analysis.

Table 2: Performance of BLEU score on WMT14 En↔De and WMT16 En↔Ro tasks. "-" denotes that the results are not reported. LSTM-based results are from (Wu et al., 2016); CNN-based results are from (Gehring et al., 2017). †The Transformer (Vaswani et al., 2017) results are based on our own reproduction.

	En→De	**De→En**	**En→Ro**	**Ro→En**		
Autoregressive Models					**Latency**	**Speedup**
LSTM Seq2Seq (Bahdanau et al., 2017)	24.60	-	-	-	-	-
Conv S2S (Gehring et al., 2017)	26.43	-	30.02	-	-	-
Transformer† (Vaswani et al., 2017)	27.48	31.29	34.36	33.82	642ms	1.00X
Non-autoregressive Models					**Latency**	**Speedup**
NAT (Gu et al., 2018)	17.69	20.62	29.79	-	39ms	15.6X
NAT, rescoring 10 (Gu et al., 2018)	18.66	22.41	-	-	79ms	7.68X
NAT, rescoring 100 (Gu et al., 2018)	19.17	23.20	-	-	257ms	2.36X
iNAT (Lee et al., 2018)	21.54	25.43	29.32	-	-	5.78X
Hint-NAT (Li et al., 2020)	21.11	25.24	-	-	26ms	23.36X
FlowSeq-base (Ma et al., 2019)	21.45	26.16	-	29.34	-	-
ENAT-P (Guo et al., 2019)	20.26	23.23	29.85	-	25ms	24.3X
ENAT-P, rescoring 9	23.22	26.67	34.04	-	50ms	12.1X
ENAT-E	20.65	23.02	30.08	-	24ms	25.3X
ENAT-E, rescoring 19	24.28	26.10	34.51	-	49ms	12.4X
DCRF-NAT (Sun et al., 2019)	23.44	27.22	-	-	37ms	16.4X
DCRF-NAT, rescoring 9	26.07	29.68	-	-	63ms	6.1X
DCRF-NAT, rescoring 19	26.80	30.04	-	-	88ms	4.4X
NAR-MT(rescoring 11) (Zhou and Keung, 2020)	23.57	29.01	31.21	32.06	-	-
NAR-MT(rescoring 11) + monolingual	25.53	29.96	31.91	33.46	-	-
AXE CMLM (Ghazvininejad et al., 2020)	23.53	27.90	30.75	31.54	-	-
SNAT	24.64	28.42	32.87	32.21	26.88ms	22.6X
SNAT, rescoring 9	26.87	30.12	34.93	33.11	54.63ms	11.1X
SNAT, rescoring 19	**27.50**	**30.82**	**35.19**	**33.98**	65.62ms	9.3X

4.1 Experimental Setup

Data We evaluate **SNAT** performance on both the WMT14 En-De (around 4.5M sentence pairs) and the WMT16 En-Ro (around 610k sentence pairs) parallel corpora. For the parallel data, we use the processed data from (Ghazvininejad et al., 2019) to be consistent with previous publications. The dataset is processed with Moses script (Hoang and Koehn, 2008), and the words are segmented into subword units using byte-pair encoding (Sennrich et al., 2016). The WMT14 En-De task uses newstest-2013 and newstest-2014 as development and test sets, and WMT16 En-Ro task uses newsdev-2016 and newstest-2016 as development and test sets. We report all results on test sets. The vocabulary is shared between source and target languages and has ∼36k units and ∼34k units in WMT14 En-De and WMT16 En-Ro, respectively.

Model Configuration Our implementation is based on the PyTorch sequence modeling toolkit Fairseq.[1] We follow the weights initialization scheme from BERT and follow the settings of the base Transformer configuration in (Vaswani et al., 2017) for all the models: 6 layers per stack, 8 attention heads per layer, 512 model dimensions and 2,048 hidden dimensions. The dimension of POS and NER embedding is set to 512 which is the same as the word embedding dimension. The autoregressive and non-autoregressive model have the same encoder-decoder structure, except for the decoder attention mask and the decoding input for the non-autoregressive model as described in Sec. 3. We try different values for the label mismatch penalty ϵ from $\{0.01, 0.1, 0.5\}$ and find that 0.1 gives the best performance. The coefficient λ is tested with different values from $\{0.25, 0.5, 0.75, 1\}$, and $\lambda = 0.75$ outperforms other settings. We set the initial learning rate as values from {8e-6, 1e-5, 2e-5, 3e-5}, with a warm-up rate of 0.1 and L2 weight decay of 0.01. Sentences are tokenized and the maximum number of tokens in each step is set to 8,000. The maximum iteration step is set to 30,000, and we train the model with early stopping.

Baselines We choose the following models as baselines: **NAT** is a vanilla non-autoregressive Transformer model for NMT which is first introduced in (Gu et al., 2018). **iNAT** (Lee et al., 2018) extends the vanilla NAT model by iteratively read-

---

[1] `https://github.com/pytorch/fairseq`

ing and refining the translation. The number of iterations is set to 10 for decoding. **Hint-NAT** (Li et al., 2020) utilizes the intermediate hidden states from an autoregressive teacher to improve the NAT model. **FlowSeq** (Ma et al., 2019) adopts normalizing flows (Kingma and Dhariwal, 2018) as latent variables for generation. **ENAT** (Guo et al., 2019) proposes two ways to enhance the decoder inputs to improve NAT models. The first one leverages a phrase table to translate source tokens to target tokens **ENAT-P**. The second one transforms source-side word embedding into target-side word embeddings **ENAT-E**. **DCRF-NAT** (Sun et al., 2019) designs an approximation of CRF for NAT models and further uses a dynamic transition technique to model positional context in the CRF. **NAR-MT** (Zhou and Keung, 2020) uses a large number of source texts from monolingual corpora to generate additional teacher outputs for NAR-MT training. **AXE CMLM** (Ghazvininejad et al., 2020) trains the conditional masked language models using a differentiable dynamic program to assign loss based on the best possible monotonic alignment between target tokens and model predictions.

4.2 Training and Inference Details

To obtain the part-of-speech and named entity labels, we use industrial-strength spaCy[2] to acquire the label for English, German, and Romanian input. In our implementation, there are 17 labels for POS in total, i.e., ADJ (adjective), ADV (adverb), ADP (ad-position), AUX (auxiliary), CCONJ (coordinating conjunction), DET (determiner), INTJ (interjection), NOUN (noun), NUM (numeral), PART (particle), PRON (pronoun), PROPN (proper noun), PUNCT (punctuation), SCONJ (subordinating conjunction), SYM (symbol), VERB (verb), and X (other). The NER task is trained on OntoNotes v5.0 benchmark dataset (Pradhan et al., 2013) using formatted BIO labels and defines 18 entity types: CARDINAL, DATE, EVENT, FAC, GPE, LANGUAGE, LAW, LOC, MONEY, NORP, ORDINAL, ORG, PERCENT, PERSON, PRODUCT, QUANTITY, TIME, and WORK_OF_ART.

Knowledge Distillation Similar to previous works on non-autoregressive translation (Gu et al., 2018; Shu et al., 2020; Ghazvininejad et al., 2019), we use sequence-level knowledge distillation by training **SNAT** on translations generated by a standard left-to-right Transformer model (i.e., Transformer-large for WMT14 EN→DE, and Transformer-base for WMT16 EN→RO). Specifically, we use scaling NMT (Ott et al., 2018) as the teacher model. We report the performance of standard autoregressive Transformer trained on distilled data for WMT14 EN→DE and WMT16 EN→RO. We average the last 5 checkpoints to obtain the final model. We train the model with cross-entropy loss and label smoothing ($\epsilon = 0.1$).

Inference During training, we do not need to predict the target length m since the target sentence is given. During inference, we use a simple method to select the target length for **SNAT** (Wang et al., 2019; Li et al., 2020). First, we set the target length to $m' = n + C$, where n is the length of the source sentence and C is a constant bias term estimated from the overall length statistics of the training data. Then, we create a list of candidate target lengths with a range of $[m' - B, m' + B]$ where B is the half-width of the interval. Finally, the model picks the best one from the generated $2B + 1$ candidate sentences. In our experiments, we set the constant bias term C to 2 for WMT 14 EN→DE, -2 for WMT 14 DE→EN, 3 for WMT 16 EN→RO, and -3 for WMT 14 RO→EN according to the average lengths of different languages in the training sets. We set B to 4 or 9, and obtain corresponding 9 or 19 candidate translations for each sentence. Then we employ an autoregressive teacher model to rescore these candidates.

4.3 Results and Analysis

Experimental results are shown in Table 2. We first compare the proposed method against autoregressive counterparts in terms of translation quality, which is measured by BLEU (Papineni et al., 2002). For all our tasks, we obtain results comparable with the Transformer, the state-of-the-art autoregressive model. Our best model achieves 27.50 (+0.02 gain over Transformer), 30.82 (-0.46 gap with Transformer), 35.19 (+0.82 gain), and 33.98 (+0.16 gain) BLEU score on WMT14 En↔De and WMT16 EN↔Ro, respectively. More importantly, our **SNAT** decodes much faster than the Transformer, which is a big improvement regarding the speed-accuracy trade-off in AT and NAT models.

Comparing our models with other NAT models, we observe that the best **SNAT** model achieves a significant performance boost over NAT, iNAT, Hint-NAT, FlowSeq, ENAT, NAR-MT and AXE CMLM by +8.33, +5.96, +6.39, +6.05, +3.22, 3.93

---

[2] `https://spacy.io/usage/models`

and +3.97 in BLEU on WMT14 En→De, respectively. This indicates that the incorporation of the syntactic and semantic structure greatly helps reduce the impact of the multimodality problem and thus narrows the performance gap between Autoregressive Transformer (AT) and Non-Autoregressive Transformer (NAT) models. In addition, we see a +0.69, +0.78, +0.68, and 0.52 gain of BLEU score over the best baselines on WMT14 En→De, WMT14 De→En, WMT16 En→Ro and WMT16 Ro→En, respectively.

From the result of our methods at the last group in Table 2, we find that the rescoring technique substantially assists in improving the performance, and dynamic decoding significantly reduces the time spent on rescoring while further accelerating the decoding process. On En→De, rescoring 9 candidates leads to a gain of +2.23 BLEU, and rescoring 19 candidates gives a +2.86 BLEU score increment.

Decoding Speed Following previous works (Gu et al., 2018; Lee et al., 2018; Guo et al., 2019), we evaluate the average per-sentence decoding latency on WMT14 En→De test sets with batch size being 1, under an environment of NVIDIA Titan RTX GPU for the Transformer model and the NAT models to measure the speedup. The latencies are obtained by taking an average of five runs. More clearly, We reproduce the Transformer on our machine. We copy the runtime of other models but the speedup ratio is between the runtime of their implemented Transformer and their proposed model. We think it's reasonable to compare the speedup ratio because it is independent of the influence caused by different implementation software or machines. And to clarify, the latency does not include preprocessing of tagging, because it's a very fast process as executing around 7000 sentences in one second.

We can see from Table 2 that the best **SNAT** gets a 9.3 times decoding speedup than the Transformer, while achieving comparable or even better performance. Compared to other NAT models, we observe that the **SNAT** model is almost the fastest (only a little bit behind of ENAT and Hint-NAT) in terms of latency, and is surprisingly faster than DCRF-NAT with better performance.

4.4 Ablation Analysis

Effect of Syntactic and Semantic Structure Information We investigate the effect of using the syntactic and semantic tag on the model performance. Experimental results are shown in Table 3.

Table 3: The performance of different vision of SNAT models on WMT14 En→De development set. ✔ means selecting the label tag.

Model	POS tag	NER tag	BLEU
SNAT-V1	✔		24.21
SNAT-V2		✔	24.09
SNAT-V3			22.84

Table 4: The performance with respect to using different layer of intermediate interaction. Evaluated by the BLEU score on WMT14 En→De|WMT14 De→En.

Method	WMT14 En→ De	WMT14 De→ En
w/o	23.11	27.03
w/ $\mathbf{Z}^2$	24.32	28.21
w/ $\mathbf{Z}^3$	24.57	28.42

It demonstrates that incorporating POS information boosts the translating performance (+1.37 on WMT14 En→De) and NER information can also enhance the translating performance (+1.25 on WMT14 En→De). The POS label enriches the model with the syntactic structure, while the NER label supplements the semantic information to the model which are critical elements for **SNAT** model to exhibit better translation performance.

Effect of Intermediate Representation Alignment We conduct experiments for our **SNAT** model on WMT14 En→De with various alignments between decoder layers and target. As shown in Table 4, using the second layer $\mathbf{Z}^2$ in the decoder as intermediate alignment can gain +1.21 improvement, while using the third layer $\mathbf{Z}^3$ in the decoder as intermediate alignment can gain +1.46 improvement. This is in line with our expectation that aggregating layer-wise token information in intermediate layers can help improve the decoder's ability to capture token-token dependencies.

Effect of Sentence Length To evaluate different models on different sentence lengths, we conduct experiments on the WMT14 En→De development set and divide the sentence pairs into different length buckets according to the length of the reference sentences. As shown in Table 5, the column of 100 calculates the BLEU score of sentences that the length of the reference sentence is larger than 50 but smaller or equal to 100. We can see that the performance of vanilla NAT drops quickly as the sentence length increases from 10 to 50, while AT model and the proposed **SNAT** model have relatively stable performance over different sen-

Table 5: The performance with respect to different sentence lengths. Evaluated by the BLEU score on WMT14 En→De.

Model	**10**	**20**	**30**	**50**	**100**
AT	28.35	28.32	28.30	24.26	20.73
NAT	21.31	19.55	17.19	16.31	11.35
SNAT	28.67	28.50	27.33	25.41	17.69

tence lengths. This result confirms the power of the proposed model in modeling long-term token dependencies.

5 Conclusion

In this paper, we have proposed a novel syntactic and semantic structure-aware non-autoregressive Transformer model **SNAT** for NMT. The proposed model aims at reducing the computational cost in inference as well as keeping the quality of translation by incorporating both syntactic and semantic structures existing among natural languages into a non-autoregressive Transformer. In addition, we have also designed an intermediate latent alignment regularization within target sentences to better learn the long-term token dependencies. Comprehensive experiments and analysis on two real-world datasets (i.e., WMT14 En→De and WMT16 En→Ro) verify the efficiency and effectiveness of our proposed approach.

Acknowledgements

This work is supported in part by NSF under grants III-1763325, III-1909323, and SaTC-1930941.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, Dzmitry Bengio, Yoshua Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *5th International Conference on Learning Representations, ICLR 2017*.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. Kermit: Generative insertion-based modeling for sequences. *arXiv preprint arXiv:1906.01604*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1243–1252. PMLR.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *Proceedings of the International Conference on Machine Learning*, pages 9330–9338.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6114–6123.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3723–3730.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Hieu Hoang and Philipp Koehn. 2008. Design of the moses decoder for statistical machine translation. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 58–65.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2390–2399, Stockholmsmässan, Stockholm Sweden. PMLR.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in neural information processing systems*, pages 10215–10224.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2020. Hint-based training for non-autoregressive machine translation. In *Proceedings of the International Conference on Learning Representations*.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4282–4292, Hong Kong, China. Association for Computational Linguistics.

Mitchell P Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3. *Linguistic Data Consortium, Philadelphia*, 14.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

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

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. 2020. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 198–205.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. In *AAAI*, pages 8846–8853.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 5976–5985.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhihong Deng. 2019. Fast structured decoding for sequence models. In *Advances in Neural Information Processing Systems*, pages 3011–3020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yiren Wang, Fei Tian, Di He, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2019. Non-autoregressive machine translation with auxiliary regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5377–5384.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1304–1312, Florence, Italy. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jiawei Zhou and Phillip Keung. 2020. Improving non-autoregressive neural machine translation with monolingual data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1893–1898, Online. Association for Computational Linguistics.

NLQuAD: A Non-Factoid Long Question Answering Data Set

Amir Soleimani
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
a.soleimani@uva.nl

Christof Monz
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
c.monz@uva.nl

Marcel Worring
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
m.worring@uva.nl

Abstract

We introduce NLQuAD, the first data set with baseline methods for non-factoid long question answering, a task requiring document-level language understanding. In contrast to existing span detection question answering data sets, NLQuAD has non-factoid questions that are not answerable by a short span of text and demanding multiple-sentence descriptive answers and opinions. We show the limitation of the F1 score for evaluation of long answers and introduce Intersection over Union (IoU), which measures position-sensitive overlap between the predicted and the target answer spans. To establish baseline performances, we compare BERT, RoBERTa, and Longformer models. Experimental results and human evaluations show that Longformer outperforms the other architectures, but results are still far behind a human upper bound, leaving substantial room for improvements. NLQuAD's samples exceed the input limitation of most pre-trained Transformer-based models, encouraging future research on long sequence language models.[1]

Question: How are people coping in the lockdown?

Headline: China coronavirus: Death toll rises as more cities restrict travel

Document: China has widened its travel restrictions in Hubei province - the centre of the coronavirus outbreak - as the death toll climbed to 26. The restrictions will affect at least 20 million people across 10 cities, including the capital, Wuhan, where the virus emerged. On Thursday, a coronavirus patient died in northern Hebei province - making it the first death outside Hubei. [...] We now know this is not a virus that will burn out on its own and disappear. [...] And we still don't know when people are contagious. Is it before symptoms appear, or only after severe symptoms emerge? One is significantly harder to stop spreading than the other. [...] **One doctor, who requested anonymity, describes the conditions at a hospital in Wuhan. [...] "I was planning to stay in my apartment because I'm scared to go to the gym, and I'm scared to go to out in public, and not many people are willing to go out." (141 words).** Vietnam and Singapore were on Thursday added to the nations recording confirmed cases, joining Thailand, the US, Taiwan and South Korea. [...] Taiwan has banned people arriving from Wuhan and the US state department warned American travellers to exercise increased caution in China. (document length: 921 words)

Figure 1: A question-answer pair in NLQuAD. QA models must predict the answer span within the context document. The correct answer span is bolded. We extract questions and answers, respectively, from the sub-headings and the sub-section bodies from real-word English news articles. Two other questions based on the same article: Can the Coronavirus be stopped? What's the global situation?

1 Introduction

Over the last few years, there have been remarkable improvements in the area of Machine Reading Comprehension (MRC) and open-domain Question Answering (QA) due to the availability of large scale data sets such as SQuAD (Rajpurkar et al., 2016) and pre-trained language models such as BERT (Devlin et al., 2018). Although non-factoid questions represent a large number of real-life questions, current QA data sets barely cover this area. The reason is that context passages in existing QA data sets are mostly very short and questions mostly factoid, i.e., can be answered by simple facts or entities such as a person name and location (Jurafsky and Martin, 2019). Little attention has been paid to non-factoid and open-ended questions that require complex answers such as descriptions or opinions (Hashemi et al., 2020). Answers to non-factoid questions extend to multiple sentences or paragraphs having few words overlapping with the question (Cohen and Croft, 2016). Non-factoid QA facilitates document assistance systems, where for example, journalists can seek assistance to highlight relevant opinions and interpretations. It can further motivate more research on long sequence

---

[1] Dataset and Models: github.com/asoleimanib/NLQuAD

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1245–1255
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

language models. Therefore, a high-quality data set in this area is clearly desired.

To support research towards non-factoid and long QA tasks and to address the existing shortcomings as identified above, we have built NLQuAD, a non-factoid long question answering data set. NLQuAD contains 31k non-factoid questions and long answers collected from 13k BBC news articles. We extract questions and answers from the articles' sub-headings and the following body paragraphs of the sub-headings (see Figure 1).

Questions in NLQuAD are not answerable by a short span of text within the documents. This is in contrast to existing long-context but factoid QA data sets such as NewsQA (Trischler et al., 2017), TriviaQA (Joshi et al., 2017), NarrativeQA (Kočiský et al., 2018), DuoRC (Saha et al., 2018), HotpotQA (Yang et al., 2018), and Natural Questions (Kwiatkowski et al., 2019). Although these data sets contain long documents, questions are answerable by short entities or a span of entities.

In particular, Natural Questions covers two types of short and long answers. However, due to its factoid questions, most long answers are still sections containing exactly the short answers and so are trivial (e.g., "Where is the world's largest ice sheet...?", Short: "Antarctica"; Long: "The Antarctic ice sheet is the largest single mass of ice on Earth..."). Furthermore, although a small portion (13%) of Natural Questions samples have only long answers, they are still spans of simple facts. For example, "Who is the author of the book Arabian Nights?" has no short answer simply because there are multiple authors: "The work was collected over many centuries by various authors, translators...". In contrast, we address non-factoid questions requiring complex answers like opinions and explanations. NLQuAD's answers are open and not predefined. Figure 3 and Table 3 present our question types. NLQuAD's questions are also not self-contained. For example, "How are people coping in the lockdown?" or "What's the global situation?" cannot be answered without the context from the document (see Figure 1). Section 3.2 discusses our question types in detail.

In most existing QA data sets such as SQuAD, crowd-workers generate questions based on provided short passages and extract answers from the passages (Rajpurkar et al., 2016). This method of question generation can make QA samples trivial because models can simply detect the most related span to the question by guessing based on shallow pattern matching (Kočiský et al., 2018). In contrast, all annotations in NLQuAD are done automatically and directly based on the news articles themselves. NLQuAD, unlike MS MARCO (Bajaj et al., 2016) and ELI5 (Fan et al., 2019), does not use information retrieval (IR) methods to collect supporting documents. Retrieved documents in these data sets are not guaranteed to contain all facts required to answer the question or they occasionally just contain information related to the question but no answers.

NLQuAD requires document-level language understanding. With an average document length and answer length of 877 and 175 words, respectively, it exceeds the maximum input length of the state of the art QA models such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) due to their memory and computational requirements. Thus, training and evaluating the (document, question, answer) tuples is impossible using such models in an end-to-end manner. It is worth noting that it is also harder to perform pre-selection methods before the final span detection because our answers are long. Meanwhile, most of our questions are not self-contained. For example, to answer the question "How are people coping in the lockdown?" (Figure 1), the system needs to read the document to interpret the concept of "lockdown" and then locate the information regarding the people's behaviour.

We also show the shortcomings of the F1 score and ROUGE-N scores in evaluating long sequences. There is a higher chance of overlap between the word N-grams in two long sequences causing F1 and ROUGE-N to over-estimate the performance. Therefore, we propose to use Intersection over Union (IoU) measuring position-sensitive overlap between two spans.

In summary, our contributions are as follows: (1) We introduce a new data set for non-factoid long QA that to the best of our knowledge is the first data set requiring long answer span detection given non-self-contained and non-factoid questions; (2) We show the limitations of the F1 score in evaluating long answers and propose a new evaluation metric; (3) To establish baseline results, we experiment with three state-of-the-art models: BERT, RoBERTa, and Longformer, and compare them with human performance. To handle the input length limitations of BERT and RoBERTa, we pro-

	Avg # Words				
data sets	Que.	Doc.	Ans.	**QA Type**	**Samples**
SQuAD	10	117	3	Factoid Span Detection	150k
NewsQA	8	616	4	Factoid Span Detection	100k
TriviaQA	14	2895	2	Factoid Span Detection	95k
NarrativeQA	10	656	5	Factoid Span Detection	47k
DouRC-Self	9	591	3	Factoid Span Detection	186k
DouRC-Pharaphrase	9	1240	3	Factoid Span Detection	186k
HotpotQA	18	917	2	Factoid Span Detection	113k
Natural Questions	9	7360	192	Factoid Span Detection	307k
DuReader	5	396	67	Factoid & Non-Factoid Span Detection	200k
DQA	N/A	N/A	54	Factoid & Non-Factoid Span Detection	17k
MS MARCO	6	56	14	Answer Generation	183k
ELI5	42	858	131	Answer Generation	272k
NLQuAD	7	877	175	Non-Factoid Span Detection	31k

Table 1: Comparison of NLQuAD with SQuAD, MS MARCO, and long-context QA data sets.

pose to train these models in a sliding-window approach; (4) We finally show that the state-of-the-art models have limited performance in the non-factoid long QA task.

2 Existing data sets

Existing large-scale QA data sets can be categorized based on their context passage length in two groups: short-context QA, i.e., data sets with paragraph-level context, and long-context QA, i.e., data sets with multiple-paragraph or document-level context. Long-context QA can potentially include questions demanding long answers. In this section, we only review QA datasets. However, it is worth noting that very recently, (Tay et al., 2020a) introduced a unified benchmark using different tasks for evaluating model quality under long-context scenarios.

2.1 Short-Context Question Answering

SQuAD (Rajpurkar et al., 2016) is a factoid span detection data set with short answers. Crowd-workers generated the questions given a set of articles. DROP (Dua et al., 2019) makes the problem more challenging by adversarially-created questions requiring discrete reasoning over the text. SQuAD and DROP use Wikipedia pages as context passages whereas SearchQA (Dunn et al., 2017) uses IR approaches to collect context passages.

Answer generation based on a set of passages is another approach to address this task. MS MARCO (Bajaj et al., 2016) consists of real-world search queries and retrieved documents corresponding to the queries.

There are also different types of QA data sets such as Antique (Hashemi et al., 2020), which is a data set for answer retrieval for non-factoid questions. There is also a range of multiple-choice QA tasks such as RACE (Lai et al., 2017), ARC (Clark et al., 2018), SWAQ (Zellers et al., 2018), and COSMOS QA (Huang et al., 2019) that are clustered together with the short-context QA data sets.

2.2 Long-Context Question Answering

Factoid QA has been applied to longer documents, however, the nature of factoid questions limits answers to short texts. NewsQA (Trischler et al., 2017), TriviaQA (Joshi et al., 2017), NarrativeQA (Kočiský et al., 2018), and DuoRC (Saha et al., 2018) fall into this category and their documents are extracted from news articles, stories, and movie plots, respectively. On the other hand, DQA (ter Hoeve et al., 2020) is a document-centred QA data set aimed at document assistance systems. Along with Yes/No questions, it also includes non-factoid questions with relatively long answers. However, the questions are generated by crowd-workers based on a small set of documents.

DuReader (He et al., 2018) consists of real-word Chinese queries and corresponding retrieved documents. It contains both factoid and non-factoid (40%) questions and consequently has longer average answer length than pure factoid datasets.

The multi-hop QA task, requiring multi-hop reasoning over multiple paragraphs, can also be considered as long-context QA if models process paragraphs together. HotpotQA (Yang et al., 2018) is a multi-hop data set, but the answer length of its factoid questions is as limited as that of short-context QA data sets.

Natural Questions (Kwiatkowski et al., 2019) is a factoid QA task with much longer documents and two types of answer lengths. It consists of factoid questions, retrieved Wikipedia pages, and short

Number of QA pairs	31k
Number of Documents	13k
Number of Unique Questions	24k
Avg. Document Length (Word)	876.8
Avg. Answer Length (Word)	174.6
Avg. Question Length (Word)	7.0
Avg. Document Length (Sentence)	38.7
Avg. Answer Length (Sentence)	7.5
Avg. Question Length (Sentence)	1.0
Avg. Question per Document	2.4

Table 2: NLQuAD: data set statistics.

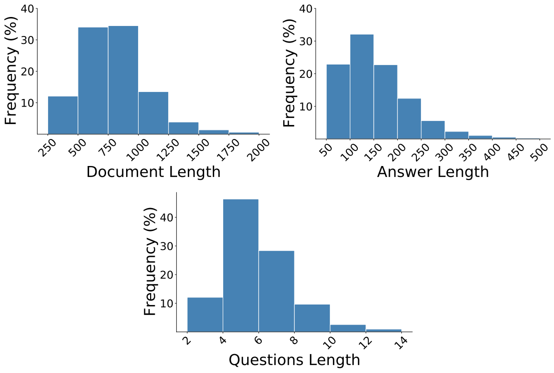


Figure 2: Distribution of the number of words in document, question and answer.

answers (yes/no, entities) as well as long answers (bounding boxes with the information to infer the answer). However, due to the nature of factoid questions, the majority of long answers are sections containing exactly the short answer or simple facts.

ELI5 (Fan et al., 2019) consists of real-world questions with answers provided by the Reddit community. The task is to generate answers given a set of documents retrieved from the Web. However, the documents are not guaranteed to completely address the questions. Furthermore, evaluation metrics for sequence generation tasks such as the ROUGE score (Lin and Och, 2004) are far from perfect to assess the quality of generated answers.

Table 1 compares existing long-context question answering data sets along with SQuAD and MS MARCO. We report the average length for data sets with different types of answers.

3 Data Set Design

NLQuAD consists of news articles as context documents, interrogative sub-headings in the articles as questions, and body paragraphs corresponding to the sub-headings as contiguous answers to the questions. We automatically extract target answers because annotating for non-factoid long QA is rather challenging and costly. To ensure the quality of answers in addition to the initial investigations, we perform human evaluations (Section 5.3). We choose the BBC news website as the resource of our documents and the question-answer pairs, mainly because its articles contain a considerable amount of high-quality question-like sub-headings which are suitable for the QA task.

NLQuAD's characteristics make it an appealing and challenging data set for the non-factoid long QA task: Its context documents are long, and its questions are non-factoid in a way that cannot be answered by single or multiple entities. The questions are addressed by more than seven sentences on average. Meanwhile, it covers a wide range of topics, making it an open-domain QA data set.

The BBC news articles typically follow a specific template. They begin with an introductory section consisting of news summaries (Narayan et al., 2018) and one or more sections accompanied by sub-headings. Each section contains multiple short to medium-length paragraphs. We remove the template and section break-lines to prevent revealing possible answer boundaries.

3.1 Data Curation

We exploit Wayback Machine,[2] a digital archive of the Web, and Wayback Machine Scraper[3] to scrape the article archives. Links in the scraped pages are used to collect additional pages from the original website. We scraped the English BBC news website from 2016 to 2020 as a limited number of questions can be found in articles before 2016. Only textual information is kept and we strip away multimedia objects and hyperlinks outside of the body of the articles. Duplicate documents are removed and questions with bullet list answer types are discarded. We detect interrogative sub-headings by checking if they end with a question mark.

3.2 Data Set Statistics

NLQuAD contains 31k non-factoid questions based on 13k supporting documents from news articles. Table 2 shows the data set statistics. We randomly partition the data set into training (80%), development (10%), and evaluation (10%) sets.

While NLQuAD has long documents and long-answer QA pairs, the histograms in Figure 2 indicate the wide range of samples. Figure 3 presents a visualisation of the distribution of question types

---

[2] archive.org/web

[3] github.com/sangaline/wayback-machine-scraper

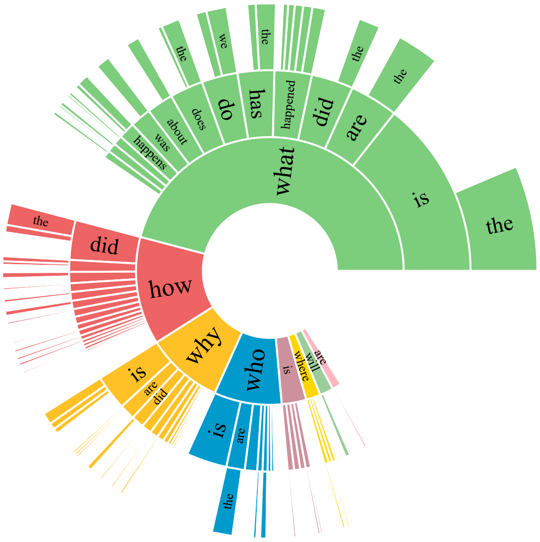


Figure 3: Distribution of trigram prefixes of questions in NLQuAD. Empty portions indicate suffixes with small percentages. NLQuAD covers a wide range of non-factoid question types.

What	How	Why
is the background...	did the attack...	is the US...
is the latest...	did we get...	is this happening...
is the reaction...	did it come...	is there a...
is happening in...	does the US/UK...	are there protests...
is in the...	does it work...	are the fires...
are the allegations...	has the government...	did the US...
did the court...	have the authorities...	was the vote...
happened in the...	do you know...	does this matter...
has the reaction...	is the shutdown...	has the US...
do we know...	many people are...	do not we...

Table 3: Top 4-grams prefixes of questions in NLQuAD. Even 'What' questions are non-factoid and need longer answers (descriptions or opinions)

in terms of their first three tokens. Table 3 also lists high frequency examples of “what”, “how” and “why” questions. NLQuAD has a large percentage of “how” and “why” question types where also the “what” examples are non-factoid and consequently require longer explanations as answers.

We manually investigated 100 randomly sampled question-answer pairs from the NLQuAD training set and find that 87% of the questions are not self-contained and require additional contextual information to be understood or disambiguated. Most of the answers consist of explanations, descriptions, or opinions, and only 2% of the questions can be answered by a short span of text.

4 Baseline Models

To investigate the difficulty level of NLQuAD for state-of-the-art QA systems and to establish baseline results, we evaluate the performance of BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and Longformer (Beltagy et al., 2020). Longformer is a scalable model for processing long documents and has been used for long sequences such as document classification (Beltagy et al., 2020) and document re-ranking (Sekulić et al., 2020). We refer readers to Tay et al. (2020b) for a detailed survey on efficient transformers. We train these Transformer-based (Vaswani et al., 2017) models to predict the span of the answer in a context document given a question and document.

4.1 BERT and RoBERTa

The BERT QA model concatenates question and document pairs into a single sequence and predicts the answer span by a dot product between the final hidden vectors, a start vector and an end vector (Devlin et al., 2018). Due to the memory and computational requirements, BERT can encode sequences with a maximum length of 512 tokens that is less than the average sample length in NLQuAD. Therefore, we adopt a sliding window approach. We split the samples into segments using a sliding window of 512 tokens and a stride of 128 tokens. Each segment is augmented with its corresponding question. The segments can include no answer, a portion of the answer, or the entire answer. We train BERT on the segments independently. Finally, the predicted spans corresponding to a single sample are aggregated to predict the final span that is the span between the earliest start position and the latest end position. The output is considered empty when all segments have empty spans.

RoBERTa has the same model architecture and input length limitation as BERT but with a robustly optimized pre-training scheme allowing it to generalize better to downstream tasks such as QA (Liu et al., 2019). We apply the same sliding window approach for RoBERTa.

4.2 Longformer

In order to process the question and entire documents at the same time, we use the Longformer model. It employs an attention mechanism scaling linearly with the sequence length which enables Longformer to process up to 4,096 tokens. It uses multiple attention heads with different dilation configurations to attend to the entire sequence and includes global attention to question tokens in the sequence. Question and document pairs are packed together into a single sequence without having to

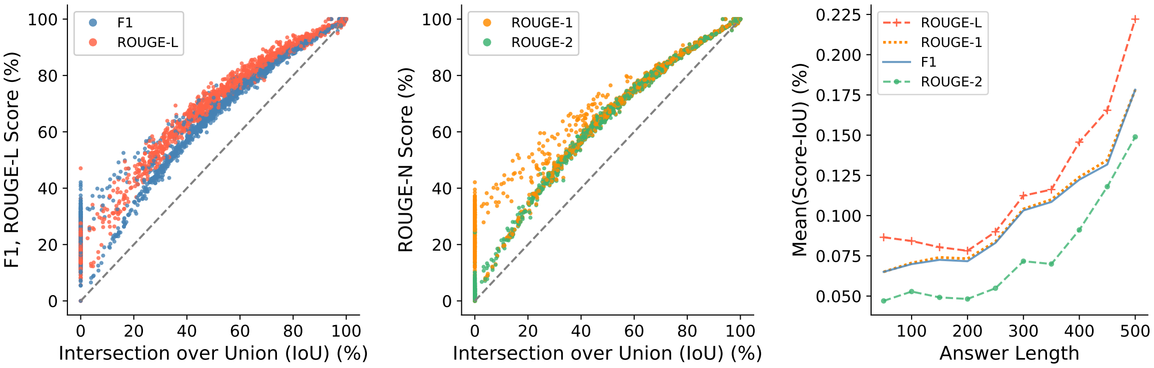


Figure 4: Comparing F1, ROUGE-N and IoU. Left/Middle: All scores behave similarly in the higher values, but F1 and ROUGE-N over-estimate the performance in the lower IoU values due to a higher chance of overlap between the bag of words, n-grams, or longer LCSs in the prediction and target spans. The dashed line shows $y = x$. Right: F1 and ROUGE-N over-estimate more in samples with longer answers. Results are plotted for the Longformer on the development set.

use sliding windows and the answer span is calculated by a dot product (Beltagy et al., 2020).

5 Experiments

5.1 Evaluation Metrics

Exact Match (EM) and the macro-averaged F1 score are the two main evaluation metrics in the span detection QA task (Rajpurkar et al., 2016). Exact Match determines if the prediction exactly matches the target which can be a too strict criterion for long answers. The F1 score measures the overlap between the words in the prediction and the target. It treats sequences as a bag of words. Unfortunately, in long answers, it is highly likely that a random, long span shares a considerable number of tokens with the target span.

The ROUGE-N scores (Lin and Och, 2004), which are primarily used for sequence generation evaluation, have the same drawback in long sequences. ROUGE-N measures the N-gram overlap between the prediction and target. High chances of overlap of unigrams and bigrams in long sequences cause ROUGE-1 and ROUGE-2 to over-estimate performance. The same holds for ROUGE-L with the Longest Common Sub-sequence (LCS) because of a high chance of longer LCSs between two long sequences.

To better take sequence similarities into account, we propose to evaluate models with the Intersection over Union (IoU) score, also known as Jaccard Index. IoU is defined as follows:

$$IoU = \frac{|p \cap t|}{|p \cup t|}$$

Question: How did we get here?

Headline: Eta disarms: French police find 3.5 tonnes of weapons

Target Answer: Slowly, and with many false starts. Eta used parts of south-western France as a base, even though most of its operations were against Spanish targets in Spain. The group has, however, killed some French policemen, but mostly during police raids on members of the group. Etaś first ceasefire was in 1998, but collapsed the following year. A similar declaration in 2006 only lasted a matter of months, ending when Eta bombed an airport car park, killing two people. Four years later, in 2010, Eta announced it would not carry out further attacks and in January 2011, it declared a permanent and "internationally verifiable" ceasefire but refused to disarm. In recent years, police in France and Spain have arrested hundreds of Eta figures and seized many of its weapons. Etaś political wing, Herri Batasuna, was banned by the Spanish government, which argued that the two groups were inextricably linked.

Prediction: The group was set up more than 50 years ago in the era of Spanish dictator General Franco, who repressed the Basques politically and culturally. Eta's goal was to create an independent Basque state out of territory in south-west France and northern Spain. Its first known killing was in 1968, when a secret police chief was shot dead in the Basque city of San Sebastian. France and Spain refuse to negotiate with Eta, which is on the EU blacklist of terrorist organisations.

Figure 5: A prediction span that is semantically different from the target span but has a F1=30% (Prec.=43%, Rec.=23%) and IoU=0. Red shows the overlapping words in the prediction span with the target. Articles (a, an, the) and punctuations are discarded before overlapping calculation. (ROUGE-1=32%, ROUGE-2=4%, ROUGE-L=24%)

where p and t and are the predicted and target contiguous intervals over the context document, containing the positions of the tokens. Intersec-

Method	EM	Prec.	Rec.	F1	IoU
BM25L	0.03	29.66	83.37	41.86	23.28
BM25L-oracle	**12.03**	**50.44**	51.18	**50.30**	**29.16**
Random Span	0.00	28.43	78.40	39.91	20.67
First Span	0.00	25.40	72.30	36.02	15.70
Last Span	0.03	29.38	**83.90**	41.77	23.63

Table 4: Ranking results on the development set. BM25L performs similar to selecting the last 512 tokens in the context document as the answer. BM25L-oracle knows the target answer span size.

Method	EM	Prec.	Rec.	F1	IoU
BERT-base e=2,s=128	23.27	60.28	**84.10**	64.34	54.04
BERT-base e=1,w,s=128	23.33	59.79	81.50	63.12	53.11
BERT-base e=2,w,s=128	**24.53**	**61.78**	83.46	**64.90**	**54.81**
BERT-base e=3,w,s=128	22.77	60.24	83.73	63.89	53.49
BERT-base e=2,w,s=256	24.09	61.64	79.08	63.38	53.41
BERT-base e=2,w,s=512	17.87	58.06	66.35	55.98	46.01
RoBERTa-base e=2,s=128	26.18	62.59	82.87	65.25	55.47
RoBERTa-base e=1,w,s=128	25.32	61.76	84.36	65.22	55.28
RoBERTa-base e=2,w,s=128	27.21	**62.71**	**85.34**	**66.17**	**56.33**
RoBERTa-base e=3,w,s=128	26.65	61.83	84.78	65.55	55.79
RoBERTa-base e=2,w,s=256	**27.33**	62.21	82.33	66.08	56.23
RoBERTa-base e=2,w,s=512	17.17	62.16	64.71	57.11	47.17
BERT-large e=2,w,s=128	28.54	63.83	84.68	66.95	57.24
RoBERTa-large e=2,w,s=128	**30.92**	**66.74**	**87.47**	**69.85**	**60.56**

Table 5: BERT and RoBERTa results on the development set. e=#epoch, w=warm-up over the first 1,000 steps, s=stride.

tion ($p \cap t = \{x | x \in p \, \text{and} \, x \in t\}$) measures the overlapping interval and union ($\cup$) is defined as $p \cup t = \{x | x \in p \, \text{or} \, x \in t\}$.

Figure 4 (left/middle) compares the F1 and ROUGE-N scores and IoU for the Longformer model on the development set. The F1 and ROUGE-N scores are always higher than IoU, but the metrics perform similarly in their higher values. Somewhat surprisingly, the F1 score can be up to 40% while there is no overlap between the two spans and IoU=0. We manually inspected the spans with F1>0 and IoU=0 and saw no significant semantic similarity between the predicted answer span and the target span. The same pattern repeats for the ROUGE-N scores. ROUGE-1 similar to F1 can reach 40% while IoU=0, but ROUGE-2 and ROUGE-L are less prone to such over-estimation due to lower chance of overlap of bigrams than unigrams and shorter LCSs in two random non-overlapping sequences. Figure 4 (right) indicates that the F1 and ROUGE-N scores are higher than IoU for longer answers reiterating the fact that these scores over-estimate more for longer sequences. Figure 5 shows two spans in a document with high F1 and ROUGE-N percentages, but different meanings.

5.2 Results and Discussion

We use the BM25L ranking function (Trotman et al., 2014) to investigate how a basic IR approach can detect answer spans using TF-IDF features. We adopt a sliding window approach with a window size of 512 and a stride of one sentence. We compare BM25L with random window (span) selection and the first and last window selection in the documents. Table 4 presents the results of the ranking functions. In the BM25L-oracle, we set the window size to the target answer span size. BM25L-oracle outperforms the other methods but the results are far from perfect. There is no significant difference between BM25L and other methods. The results restate the fact that there is little word overlap between non-factoid questions and their answers.

We analyze the performance of BERT and RoBERTa with different hyper-parameters on the development set in Table 5. Smaller strides, i.e., higher overlap between the segments, and warm-up contribute to better performances. RoBERTa constantly outperforms BERT, which is to be expected as RoBERTa is optimized robustly during the pre-training. We use the HuggingFace's Transformers (Wolf et al., 2019) code[4] and train the base and large models on 2 and 4 GPUs, respectively. We

---

[4] github.com/huggingface/transformers

Method	#Param.	EM	Prec.	Rec.	F1	IoU
BERT-base	110M	25.03	60.60	82.48	63.96	53.75
BERT-large	340M	30.29	64.87	84.62	67.91	58.39
RoBERTa-base	125M	29.07	64.02	84.79	67.19	57.65
RoBERTa-large	355M	33.40	67.79	**87.56**	71.10	62.39
Longformer	149M	**50.30**	**83.92**	85.17	**81.38**	**73.57**

Table 6: NLQuAD evaluation set results. Longformer surpasses the other models in all the metrics except recall.

have to use a batch size of 12 and 8, respectively, for the base and large models because of the long input sequence size and memory limitations.

We use the official AllenAI Longformer code[5] to train Longformer on NLQuAD. We use the same batch size of 12 (batch size of 1 and gradient accumulation over 12 batches) and learning rate warm-up for the first 1,000 steps. Due to memory requirements, we limit the experiments to only the Longformer base model (the large model cannot fit on our GPUs even with a batch size of 1). We ran the experiments on 2 NVIDIA P40 (24GB GPU memory) for about one day for 5 epochs. Similarly, we choose the best epoch based on the performance on the development set.

Table 6 summarizes the scores obtained by the baseline systems on the NLQuAD evaluation set. While Longformer significantly outperforms BERT and RoBERTa, its performance, particularly in terms of IoU and EM, is far from perfect. This demonstrates that NLQuAD and non-factoid QA is still an open problem for state-of-the-art models.

Method	EM	Prec.	Rec.	F1	IoU
RoBERTa-large	36.00	61.78	87.00	66.41	57.09
Human-AVG	35.50	86.67	68.66	72.52	62.94
Longformer	56.00	78.55	83.69	78.27	70.64
Human-UB	**74.00**	**97.79**	**94.88**	**95.49**	**92.63**

Table 7: Comparing human performance with Longformer and RoBERTa-large on a subset of evaluation set. UB=upper bound, AVG=average.

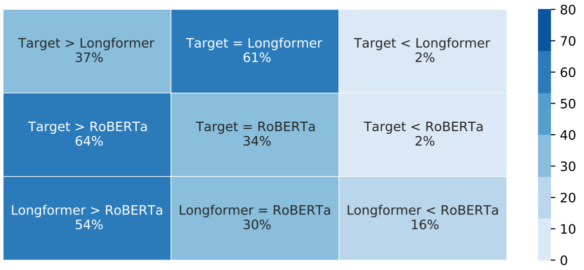


Figure 6: Pairwise comparison between the target spans, Longformer, and RoBERTa's predicted spans. X>Y means X is more preferable.

5.3 Human Evaluation

To ensure that the samples are of high quality, in addition to the initial investigation and pre-processing steps, we asked four volunteers to investigate 50 random samples from the evaluation set. They rated the goodness of answers on a 3-point scale: (1: Irrelevant answer; 2: Good answer after adding or removing some sentences; 3: Perfect answer). The average score is 2.56 indicating the high quality of NLQuAD's QA samples.

In order to benchmark human performance, we asked the four volunteers to answer 50 questions, a randomly sampled subset of evaluation set. They were given unlimited time to detect the answers, but on average, it took them about 270 seconds to answer a question. Table 7 compares human performance with Longformer and RoBERTa-large on the same subset. Similar to HotpotQA (Yang et al., 2018), we estimate the human upper bound by taking the best human answer in terms of our primary evaluation metric (IoU) for each sample. While NLQuAD is a challenging task both for humans and the state of the art QA models, the human upper bound performance significantly outperforms the models. We suspect that the mediocre average of human performance, considering the high score of the target answers, might be because volunteers are not familiar with the articles' writing style or they might have become exhausted by reading long articles.

Furthermore, we asked another volunteer to compare the target answers with the predicted answers in a pairwise comparison for 100 samples. Figure 6 shows that the target answers are preferred in 37% and 64% of cases over the Longformer and RoBERTa predictions, respectively. The human evaluation is in line with the results shown in Table 6 and Table 7.

---

[5] github.com/allenai/longformer

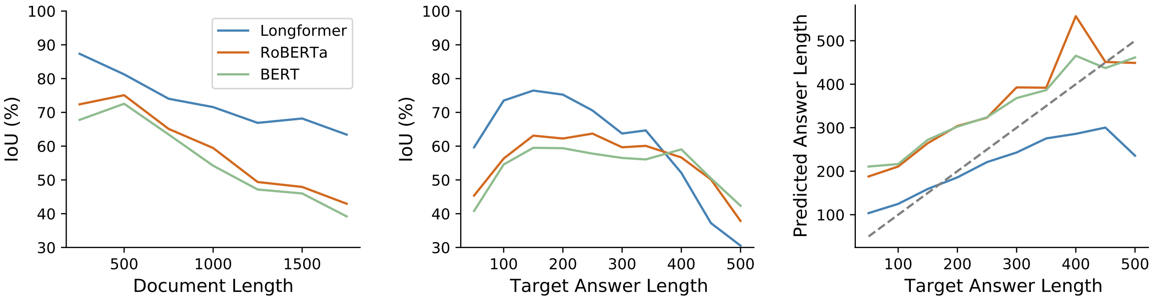


Figure 7: Effect of document and answer length on the performances. Left: IoU drops in all models for longer documents. Middle: RoBERTa and BERT outperform Longformer in longer answers. Right: Longformer has a bias to predict shorter answers while RoBERTa and BERT predict longer answers. The dashed line means $y = x$.

5.4 Error Analysis

Figure 7 compares the performance of BERT, RoBERTa, and Longformer for instances with different document and answer lengths. As expected, both longer documents and longer answers are harder for the models. Surprisingly, BERT and RoBERTa outperform Longformer for longer answers. The same pattern occurs for F1 and EM (not shown in the figure).

Figure 7 (right) shows that RoBERTa and BERT behave completely differently compared to Longformer for longer answer lengths. The former models have a bias to predict longer spans while Longformer under-estimates the length of the answer span. This different behaviour might be due to the sliding window approach and the prediction aggregation in the RoBERTa and BERT models and the attention dilation strategy in Longformer.

6 Conclusion

We introduce NLQuAD, a non-factoid long question answering data set from BBC news articles. NLQuAD's question types and the long lengths of its context documents as well as answers, make it a challenging real-world task. We propose to use Intersection over Union (IoU) as an evaluation metric for long question answering. To establish a baseline performance, we experimented with the BERT, RoBERTa, and Longformer question answering models. Longformer outperforms the other methods with an IoU of 73.57%, but the results show that the performance of state-of-the-art question answering systems is far from perfect. We hope NLQuAD will inspire more research in the area of document-level language understanding and question answering.

Acknowledgments

This research was partly supported by VIVAT. We thank the BBC for giving permission to publish our extracted data for non-commercial, research purposes. We also thank our volunteers for providing human assessments.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. MS MARCO: A human generated machine reading comprehension dataset. *arXiv:1611.09268*.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv:1803.05457*.

Daniel Cohen and W. Bruce Croft. 2016. End to end long short term memory networks for non-factoid question answering. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ICTIR 16, page 143–146, New York, NY, USA. Association for Computing Machinery.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv:1704.05179*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. 2020. ANTIQUE: A non-factoid question answering benchmark. In *European Conference on Information Retrieval*, pages 166–173. Springer.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Maartje ter Hoeve, Robert Sim, Elnaz Nouri, Adam Fourney, Maarten de Rijke, and Ryen W. White. 2020. Conversations with documents: An exploration of document-centered assistance. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval*, page 43–52, New York, NY, USA. Association for Computing Machinery.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. COSMOS QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Dan Jurafsky and James H. Martin. 2019. Speech and language processing, 3rd edition.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, page 605–es, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv:1907.11692*.

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

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

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Ivan Sekulić, Amir Soleimani, Mohammad Aliannejadi, and Fabio Crestani. 2020. Longformer for MS MARCO document re-ranking task. *arXiv:2009.09392*.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020a. Long range arena: A benchmark for efficient transformers. *arXiv:2011.04006*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020b. Efficient transformers: A survey. *arXiv:2009.06732*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to BM25 and language models examined. In *Proceedings of the 2014 Australasian Document Computing Symposium*, ADCS '14, page 58–65, New York, NY, USA. Association for Computing Machinery.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv:1910.03771*.

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

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Debiasing Pre-trained Contextualised Embeddings

Masahiro Kaneko
Tokyo Metropolitan University
`kaneko-masahiro@ed.tmu.ac.jp`

Danushka Bollegala[*]
University of Liverpool, Amazon
`danushka@liverpool.ac.uk`

Abstract

In comparison to the numerous debiasing methods proposed for the static non-contextualised word embeddings, the discriminative biases in contextualised embeddings have received relatively little attention. We propose a fine-tuning method that can be applied at token- or sentence-levels to debias pre-trained contextualised embeddings. Our proposed method can be applied to any pre-trained contextualised embedding model, without requiring to retrain those models. Using gender bias as an illustrative example, we then conduct a systematic study using several state-of-the-art (SoTA) contextualised representations on multiple benchmark datasets to evaluate the level of biases encoded in different contextualised embeddings before and after debiasing using the proposed method. We find that applying token-level debiasing for all tokens and across all layers of a contextualised embedding model produces the best performance. Interestingly, we observe that there is a trade-off between creating an accurate vs. unbiased contextualised embedding model, and different contextualised embedding models respond differently to this trade-off.

1 Introduction

Contextualised word embeddings have significantly improved performance in numerous natural language processing (NLP) applications (Devlin et al., 2019; Liu et al., 2019; Clark et al., 2020) and have established as the de facto standard for input text representations. Compared to static word embeddings (Pennington et al., 2014; Mikolov et al., 2013) that represent a word by a single vector in all contexts it occurs, contextualised embeddings use dynamic context dependent vectors for representing a word in a specific context. Unfortunately however, it has been shown that, similar to their non-contextual counterparts, contextualised text embeddings also encode various types of unfair biases (Zhao et al., 2019; Bordia and Bowman, 2019; May et al., 2019; Tan and Celis, 2019; Bommasani et al., 2020; Kurita et al., 2019). This is a worrying situation because such biases can easily propagate to the downstream NLP applications that use contextualised text embeddings.

Different types of unfair and discriminative biases such as gender, racial and religious biases have been observed in static word embeddings (Bolukbasi et al., 2016; Zhao et al., 2018a; Rudinger et al., 2018; Zhao et al., 2018b; Elazar and Goldberg, 2018; Kaneko and Bollegala, 2019). As discussed later in § 2 different methods have been proposed for debiasing static word embeddings such as projection-based methods (Kaneko and Bollegala, 2019; Zhao et al., 2018b; Bolukbasi et al., 2016; Ravfogel et al., 2020) and adversarial methods (Xie et al., 2017; Gonen and Goldberg, 2019). In contrast, despite multiple studies reporting that contextualised embeddings to be unfairly biased, methods for debiasing contextualised embeddings are relatively under explored (Dev et al., 2020; Nadeem et al., 2020; Nangia et al., 2020). Compared to static word embeddings, debiasing contextualised embeddings is significantly more challenging due to several reasons as we discuss next.

First, compared to static word embedding models where the semantic representation of a word is limited to a single vector, contextualised embedding models have a significantly large number of parameters related in complex ways. For example, BERT-large model (Devlin et al., 2019) contains 24 layers, 16 attention heads and 340M parameters. Therefore, it is not obvious which parameters are responsible for the unfair biases related to a partic-

---

[*] Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1256–1266
April 19 - 23, 2021.

ular word. Because of this reason, projection-based methods, popularly used for debiasing pre-trained static word embeddings, cannot be directly applied to debias pre-trained contextualised word embeddings.

Second, in the case of contextualised embeddings, the biases associated with a particular word's representation is a function of both the target word itself and the context in which it occurs. Therefore, the same word can show unfair biases in some contexts and not in the others. It is important to consider the words that co-occur with the target word in different contexts when debiasing a contextualised embedding model.

Third, pre-training large-scale contextualised embeddings from scratch is time consuming and require specialised hardware such as GPU/TPU clusters. On the other hand, fine-tuning a pre-trained contextualised embedding model for a particular task (possibly using labelled data for the target task) is relatively less expensive. Consequently, the standard practice in the NLP community has been to share[1] pre-trained contextualised embedding models and fine-tune as needed. Therefore, it is desirable that a debiasing method proposed for contextualised embedding models can be applied as a fine-tuning method. In this view, counterfactual data augmentation methods (Zmigrod et al., 2019; Hall Maudslay et al., 2019; Zhao et al., 2019) that swap gender pronouns in the training corpus for creating a gender balanced version of the training data are less attractive when debiasing contextualised embeddings because we must retrain those models on the balanced corpora, which is more expensive compared to fine-tuning.

Using gender-bias as a running example, we address the above-mentioned challenges by proposing a debiasing method that fine-tunes pre-trained contextualised word embeddings[2]. Our proposed method retains the semantic information learnt by the contextualised embedding model with respect to gender-related words, while simultaneously removing any stereotypical biases in the pre-trained model. In particular, our proposed method is agnostic to the internal architecture of the contextualised embedding method and we apply it to debias different pre-trained embeddings such as BERT, RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), DistilBERT (Sanh et al., 2019) and ELECTRA (Clark et al., 2020). Moreover, our proposed method can be applied at token-level or at sentence-level, enabling us to debias at different granularities and on different layers in the pre-trained contextualised embedding model.

Following prior work, we compare the proposed debiasing method in two sentence-level tasks: Sentence Encoder Association Test (**SEAT**; May et al., 2019) and Multi-genre co-reference-based Natural Language Inference (**MNLI**; Dev et al., 2020). Experimental results show that the proposed method not only debiases all contextualised word embedding models compared, but also preserves useful semantic information for solving downstream tasks such as sentiment classification (Socher et al., 2013), paraphrase detection (Dolan and Brockett, 2005), semantic textual similarity measurement (Cer et al., 2017), natural language inference (Dagan et al., 2005; Bar-Haim et al., 2006) and solving Winograd schema (Levesque et al., 2012). We consider gender bias as a running example throughout this paper and evaluate the proposed method with respect to its ability to overcome gender bias in contextualised word embeddings, and defer extensions to other types of biases to future work.

2 Related Work

Prior work on debiasing word embeddings can be broadly categorised into two groups depending on whether they consider static or contextualised word embeddings. Although we focus on contextualised embeddings in this paper, we first briefly describe prior work on debiasing static embeddings for completeness of the discussion.

Bias in Static Word Embeddings: Bolukbasi et al. (2016) proposed a post-processing approach that projects gender-neutral words into a subspace, which is orthogonal to the gender direction defined by a list of gender-definitional words. However, their method ignores gender-definitional words during the subsequent debiasing process, and focus only on words that are *not* predicted as gender-definitional by a classifier. Therefore, if the classifier erroneously predicts a stereotypical word as gender-definitional, it would not get debiased. Zhao et al. (2018b) modified the original GloVe (Pennington et al., 2014) objective to learn gender-neutral word embeddings (GN-GloVe) from a given corpus. Unlike the above-

---

[1] `https://huggingface.co/transformers/pretrained_models.html`

[2] Code and debiased embeddings: `https://github.com/kanekomasahiro/context-debias`

mentioned methods, Kaneko and Bollegala (2019) proposed GP-GloVe, a post-processing method to preserve gender-related information with autoencoder (Kaneko and Bollegala, 2020), while removing discriminatory biases from stereotypical cases.

Adversarial learning (Xie et al., 2017; Elazar and Goldberg, 2018; Li et al., 2018) for debiasing first encode the inputs and then two classifiers are jointly trained – one predicting the target task (for which we must ensure high prediction accuracy) and the other for protected attributes (that must not be easily predictable). Elazar and Goldberg (2018) showed that although it is possible to obtain chance-level development-set accuracy for the protected attributes during training, a post-hoc classifier trained on the encoded inputs can still manage to reach substantially high accuracies for the protected attributes. They conclude that adversarial learning alone does not guarantee invariant representations for the protected attributes. Ravfogel et al. (2020) found that iteratively projecting word embeddings to the null space of the gender direction to further improve the debiasing performance.

Benchmarks for biases in Static Embeddings: Word Embedding Association Test (WEAT; Caliskan et al., 2017) quantifies various biases (e.g. gender, race and age) using semantic similarities between word embeddings. Word Association Test (WAT) measures gender bias over a large set of words (Du et al., 2019) by calculating the gender information vector for each word in a word association graph created in the Small World of Words project (SWOWEN; Deyne et al., 2019) by propagating masculine and feminine words via a random walk (Zhou et al., 2003). SemBias dataset (Zhao et al., 2018b) contains three types of word-pairs: (a) **Definition**, a gender-definition word pair (e.g. hero – heroine), (b) **Stereotype**, a gender-stereotype word pair (e.g., manager – secretary) and (c) **None**, two other word-pairs with similar meanings unrelated to gender (e.g., jazz – blues, pencil – pen). It uses the cosine similarity between the gender directional vector, $(\overrightarrow{he} - \overrightarrow{she})$, and the offset vector $(\boldsymbol{a} - \boldsymbol{b})$ for each word pair, (a, b), in each set to measure gender bias. WinoBias (Zhao et al., 2018a) uses the ability to predict gender pronouns with equal probabilities for gender neutral nouns such as occupations as a test for the gender bias in embeddings.

Bias in Contextualised Word Embeddings: May et al. (2019) extended WEAT using templates to create a sentence-level benchmark for evaluating bias called SEAT. In addition to the attributes proposed in WEAT, they proposed two additional bias types: *angry black woman* and *double binds* (when a woman is doing a role that is typically done by a man that woman is seen as arrogant). They show that compared to static embeddings, contextualised embeddings such as BERT, GPT and ELMo are less biased. However, similar to WEAT, SEAT also only has positive predictive ability and cannot detect the absence of a bias. Bommasani et al. (2020) evaluated the bias in contextualised embeddings by first distilling static embeddings from contextualised embeddings and then using WEAT tests for different types of biases such as gender (male, female), racial (White, Hispanic, Asian) and religion (Christianity, Islam). They found that aggregating the contextualised embedding of a particular word in different contexts via averaging to be the best method for creating a static embedding from a contextualised embedding.

Zhao et al. (2019) showed that contextualised ELMo embeddings also learn gender biases present in the training corpus. Moreover, these biases propagate to a downstream coreference resolution task. They showed that data augmentation by swapping gender helps more than neutralisation by a projection. They obtain the embedding of two input sentences with reversed gender from ELMo, and obtain the debiased embedding by averaging them. It can only be applied to feature-based embeddings, so it cannot be applied to fine-tuning based embeddings like BERT. We directly debias the contextual embeddings. Additionally, data augmentation requires re-training of the embeddings, which is often costly compared to fine-tuning. Kurita et al. (2019) created masked templates such as "__ is a nurse" and used BERT to predict the masked gender pronouns. They used the log-odds between male and female pronoun predictions as an evaluation measure and showed that BERT to be biased according to it. Karve et al. (2019) learnt conceptor matrices using class definitions in the WEAT and used the negated conceptors to debias ELMo and BERT. Although their method was effective for ELMo, the results on BERT were mixed. This method can only be applied to context-independent vectors, and it requires the creation of static embeddings from BERT and ELMo as a pre-processing step for debi-

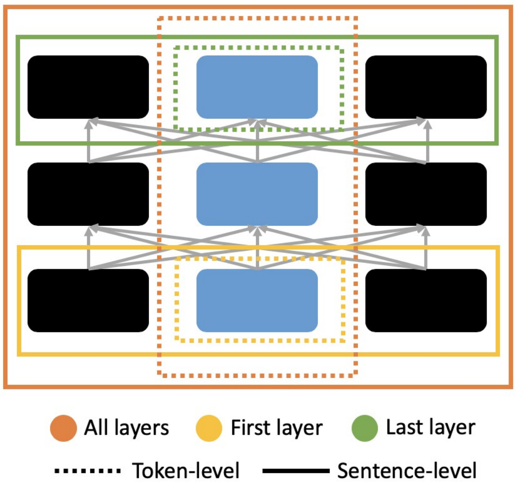


Figure 1: Types of hidden states in E considered in the proposed method. The blue boxes in the middle correspond to the hidden states of the target token.

asing the context-dependent vectors. Therefore, we do not compare against this method in the present study, where we evaluate on context-dependent vectors.

Dev et al. (2020) used natural language inference (NLI) as a bias evaluation task, where the goal is to ascertain if one sentence (i.e. premise) entails or contradictions another (i.e. hypothesis), or if neither conclusions hold (i.e. neutral). The premise-hypothesis pairs are constructed to elicit various types of discriminative biases. They showed that orthogonal projection to gender direction (Dev and Phillips, 2019) can be used to debias contextualised embeddings as well. However, their method can be applied only to the noncontextualised layers (ELMo's Layer 1 and BERT's subtoken layer). In contrast, our proposed method can be applied to *all* layers in a contextualised embedding and outperforms their method on the same NLI task. And our debiasing approach does not require task-dependent data.

3 Debiasing Contextualised Embeddings

We propose a method for debiasing pre-trained contextualised word embeddings in a fine-tuning setting that simultaneously (a) preserves the semantic information in the pre-trained contextualised word embedding model, and (b) removes discriminative gender-related biases via an orthogonal projection in the intermediate (hidden) layers by operating at token- or sentence-levels. Fine-tuning allows debiasing to be carried out without requiring large amounts of tarining data or computational resources. Our debiasing method is independent of model architectures or their pre-training methods, and can be adapted to a wide range of contextualised embeddings as shown in § 4.3.

Let us define two types of words: *attribute* words ($\mathcal{V}_a$) and *target* words ($\mathcal{V}_t$). For example, in the case of gender bias, attribute words consist of multiple word sets such as feminine (e.g. she, woman, her) and masculine (e.g. he, man, him) words, whereas target words can be occupations (e.g. doctor, nurse, professor), which we expect to be gender neutral. We then extract sentences that contain an attribute or a target word. Sentences contain more than one attribute (or target) words are excluded to avoid ambiguities. Let us denote the set of sentences extracted for an attribute or a target word w by $\Omega(w)$. Moreover, let $\mathcal{A} = \bigcup_{w \in \mathcal{V}_a} \Omega(w)$ and $\mathcal{T} = \bigcup_{w \in \mathcal{V}_t} \Omega(w)$ be the sets of sentences containing respectively all of the attribute and target words. We require that the debiased contextualised word embeddings preserve semantic information w.r.t. the sentences in $\mathcal{A}$, and remove any discriminative biases w.r.t. the sentences in $\mathcal{T}$.

Let us consider a contextualised word embedding model E, with pre-trained model parameters $\boldsymbol{\theta}_e$. For an input sentence x, let us denote the embedding of token w in the i-th layer of E by $E_i(w, x; \boldsymbol{\theta}_e)$. Moreover, let the total number of layers in E to be N. In our experiments, we consider different types of encoder models such as E. To formalise the requirement that the debiased word embedding $E_i(t, x; \boldsymbol{\theta}_e)$ of a target word $t \in \mathcal{V}_t$ must not contain any information related to a protected attribute a, we consider the inner-product between the noncontextualised embedding $\boldsymbol{v}_i(a)$ of a and $E_i(t, x; \boldsymbol{\theta}_e)$ as a loss L_i given by (1).

$$L_i = \sum_{t \in \mathcal{V}_t} \sum_{x \in \Omega(t)} \sum_{a \in \mathcal{V}_a} \left(\boldsymbol{v}_i(a)^\top E_i(t, x; \boldsymbol{\theta}_e) \right)^2 \quad (1)$$

Here, $\boldsymbol{v}_i(a)$ is computed by averaging the contextualised embedding of a in the i-th layer of E over all sentences in $\Omega(a)$ following Bommasani et al. (2020) and is given by (2).

$$\boldsymbol{v}_i(a) = \frac{1}{|\Omega(a)|} \sum_{x \in \Omega(a)} E_i(a, x; \boldsymbol{\theta}_e) \quad (2)$$

Here, $|\Omega(a)|$ denotes the total number of sentences in $\Omega(a)$. If a word is split into multiple sub-tokens, we compute the contextualised embedding of the word by averaging the contextualised embeddings of its constituent sub-tokens. Minimising the loss L_i defined by (1) with respect to $\boldsymbol{\theta}_e$ forces the hidden states of E to be orthogonal to the protected attributes such as gender.

Although removing discriminative biases in E is our main objective, we must ensure that simultaneously we preserve as much useful information that is encoded in the pre-trained model for the downstream tasks. We model this as a regulariser where we measure the squared ℓ_2 distance between the contextualised word embedding of a word w in the i-th layer in the original model, parametrised by $\boldsymbol{\theta}_{\text{pre}}$, and the debiased model as in (3).

$$L_{\text{reg}} = \sum_{x\in\mathcal{A}} \sum_{w\in x} \sum_{i=1}^{N} ||E_i(w,x;\boldsymbol{\theta}_e) - E_i(w,x;\boldsymbol{\theta}_{\text{pre}})||^2 \quad (3)$$

The overall training objective is then given by (4) as the linearly weighted sum of the two losses defined by (1) and (3).

$$L = \alpha L_i + \beta L_{\text{reg}} \quad (4)$$

Here, coefficients $\alpha, \beta \in [0, 1]$ satisfy $\alpha + \beta = 1$.

As shown in Figure 1, a contextualised word embedding model typically contains multiple layers. It is not obvious which hidden states of E are best for calculating L_i for the purpose of debiasing. Therefore, we compute L_i for different layers in a particular contextualised word embedding model in our experiments. Specifically, we consider three settings: debiasing only the **first** layer, **last** layer or **all** layers. Moreover, L_i can be computed only for the target words in a sentence x as in (1), or can be summed up for *all* words in $w \in x$ (i.e. $\sum_{t\in\mathcal{V}_t}\sum_{x\in\Omega(t)}\sum_{w\in x}\left(\boldsymbol{v}_i(a)^\top E_i(w,x;\boldsymbol{\theta}_e)\right)^2$). We refer to the former as **token-level** debiasing and latter **sentence-level** debiasing. Collectively this gives us six different settings for the proposed debiasing method, which we evaluate experimentally in § 4.3.

4 Experiments

4.1 Datasets

We used SEAT (May et al., 2019) 6, 7 and 8 to evaluate gender bias. We use NLI as a downstream evaluation task and use the Multi-Genre Natural Language Inference data (MNLI; Williams et al., 2018) for training and development following Dev et al. (2020). In NLI, the task is to classify a given hypothesis and premise sentence-pair as entailing, contradicting, or neutral. We programmatically generated the evaluation set following Dev et al. (2020) by filling occupation words and gender words in template sentences. The templates take the form "The subject verb a/an object." and the created sentence-pairs are assumed to be neutral.

We used the word lists created by Zhao et al. (2018b) for the attribute list of feminine and masculine words. As for the stereotype word list for target words, we use the list created by Kaneko and Bollegala (2019). Using News-commentary-v15 corpus[3] was extract 11023, 42489 and 34148 sentences respectively for Feminine, Masculine and Stereotype words. We excluded sentences with more than 128 tokens in training data. We randomly sampled 1,000 sentences from each type of extracted sentences as development data.

We used the GLEU benchmark (Wang et al., 2018) to evaluate whether the useful information in the pre-trained embeddings is retrained after debiasing. To evaluate the debiased models with minimal effects due to task-specific fine-tuning, we used the following small-scale training data: Stanford Sentiment Treebank (SST-2; Socher et al., 2013), Microsoft Research Paraphrase Corpus (MRPC; Dolan and Brockett, 2005), Semantic Textual Similarity Benchmark (STS-B; Cer et al., 2017), Recognising Textual Entailment (RTE; Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), and Winograd Schema Challenge (WNLI; Levesque et al., 2012). We evaluate the performance of the contextualised embeddings on the corresponding development data.

4.2 Hyperparameters

We used BERT (**bert-base-uncased**; Devlin et al., 2019), RoBERTa (**roberta-base**; Liu et al., 2019), ALBERT (**albert-base-v2**; Lan et al., 2020), DistilBERT (**distilbert-base-uncased**; Sanh et al., 2019) and ELECTRA (**electra-small-discriminator**; Clark et al., 2020) in our experiments.[4] DistilBERT has 6 layers and the others 12. We used the development data in SEAT-6 for hyperparameter tuning. The hyperparameters of the models, except the learning rate and batch size, are set to their default values as in `run_glue.py`. Using greedy search, the learning rate was set to 5e-5 and the

---

[3] http://www.statmt.org/wmt20/translation-task.html

[4] We used https://github.com/huggingface/transformers

Model	*Layer*	*Unit*	SEAT-6	SEAT-7	SEAT-8	#†	SST-2	MRPC	STS-B	RTE	WNLI	Avg
BERT	all	token	0.68†	-0.09	0.60†	2	92.1	85.6	83.1	60.0	53.5	74.9
		sent	1.13†	0.34	0.12	**1**	91.9	82.6	80.0	54.2	40.8	69.9
	last	token	1.02†	-1.18	0.47†	2	92.2	86.9	82.3	58.1	56.3	75.2
		sent	1.51†	-0.60	1.52†	2	92.3	84.6	82.9	62.1	56.3	75.6
	first	token	0.88†	0.33	0.86†	2	92.4	87.1	82.6	62.1	50.7	75.0
		sent	0.94†	0.32	0.97†	2	91.9	86.1	83.0	63.9	46.5	74.3
	original		1.04†	0.18	0.81†	2	92.8	86.7	82.4	60.6	56.3	75.8
	random		1.16†	-0.08	-0.29	**1**	92.2	87.4	81.9	63.2	54.9	**75.9**
RoBERTa	all	token	0.51†	0.15	0.02	**1**	78.1	81.6	73.7	53.8	56.3	68.7
		sent	1.27†	0.86†	1.14†	3	80.3	82.8	74.4	50.9	56.3	68.9
	last	token	1.17†	-0.60	0.45†	2	79.9	83.7	74.1	52.3	56.3	69.3
		sent	0.98†	0.75†	0.87†	3	69.5	81.5	72.9	52.7	56.3	66.6
	first	token	1.15†	0.26	0.54†	2	77.8	81.1	74.5	54.5	56.3	68.8
		sent	1.21†	0.32	0.50†	2	79.0	82.5	74.5	51.6	56.3	68.8
	original		1.21†	1.34†	1.01†	3	93.8	91.2	89.8	71.8	56.3	**80.6**
	random		1.39†	0.40†	0.39†	3	73.4	82.5	73.9	53.4	49.3	66.5
ALBERT	all	token	0.16	0.02	0.18	**0**	78.1	80.5	67.5	54.9	56.3	67,5
		sent	0.18	-0.05	-0.77	**0**	77.3	81.7	69.9	46.9	56.3	66.4
	last	token	0.83†	-1.15	-0.76	1	77.8	81.2	68.9	47.3	56.3	66.3
		sent	0.69†	-0.06	-0.10	1	78.3	80.1	71.3	55.2	56.3	68,2
	first	token	0.09	0.28	0.97†	1	77.9	81.6	70.0	52.0	56.3	67,6
		sent	0.25	0.60†	1.18†	2	75.9	81.3	70.1	53.1	54.9	67,1
	original		0.30	0.48†	1.12†	2	92.2	89.9	87.7	70.0	56.3	**79.2**
	random		0.41†	0.34	1.08†	2	78.2	79.9	71.8	47.3	56.3	66.7
DistilBERT	all	token	0.70†	-0.83	-0.66	**1**	90.4	87.8	80.8	56.0	42.3	71.5
		sent	1.34†	1.01†	0.97†	3	91.4	83.3	78.8	57.4	53.5	**72.9**
	last	token	1.11†	-0.03	1.38†	2	90.9	88.5	80.3	55.6	38.0	70.7
		sent	1.57†	-1.34	0.27	**1**	90.8	90.2	80.9	58.5	43.7	72.8
	first	token	1.19†	0.59†	0.52†	3	90.8	90.8	80.4	55.2	38.0	71.0
		sent	1.19†	0.60†	0.55†	3	91.1	90.9	80.1	55.2	36.6	70.8
	original		1.26†	0.31	0.74†	2	90.8	89.3	80.6	56.0	38.0	70.9
	random		1.35†	0.66†	-0.25	2	91.1	89.1	80.5	56.3	40.8	71.6
ELECTRA	all	token	0.33	0.10	0.15	**0**	90.3	87.7	79.4	52.7	57.7	**73.6**
		sent	0.42†	0.21	0.33	1	90.7	87.1	79.5	52.3	54.9	72.9
	last	token	0.55†	0.07	0.24	1	90.8	87.3	79.8	51.6	46.5	71.2
		sent	0.50†	0.42†	0.32†	3	90.5	87.3	80.1	54.5	40.8	70.6
	first	token	0.31	0.10	0.33	**0**	90.4	86.9	79.7	53.1	56.3	73.4
		sent	0.29	0.22	0.30	**0**	90.4	87.6	79.7	53.4	56.3	73.5
	original		0.16	0.46†	0.04	1	90.5	87.9	80.4	54.5	46.5	72.0
	random		0.43†	0.49†	-0.22	2	90.4	87.7	78.5	51.3	54.9	72.6

Table 1: Gender bias of contextualised embeddings on SEAT. † denotes significant bias effects at $\alpha < 0.01$.

batch size to 32 during debiasing. Optimal values for $\alpha = 0.2$ and $\beta = 0.8$ were found by a greedy search in $[0, 1]$ with 0.1 increments. For the GLEU and MNLI experiments, we set the learning rate to 2e-5 and the batch size to 16. Experiments were conducted on a GeForce GTX 1080 Ti GPU.

4.3 Debiasing vs. Preserving Information

Table 1 shows the results on SEAT and GLEU where **original** denotes the pre-trained contextualised models prior to debiasing. We see that original models other than ELECTRA contain significant levels of gender biases. Overall, the **all-token** method that conducts token-level debiasing across all layers performs the best. Prior work has shown that biases are learned at each layer (Bommasani et al., 2020) and it is important to debias all layers. Moreover, we see that debiasing at token-level is more efficient compared to at the sentence-level. This is because in token-level debiasing, the loss is computed only on the target word and provides a more direct debiasing update for the target word than in the sentence-level debiasing, which sums the losses over all tokens in a sentence.

To test the importance of carefully selecting the target words considering the types of biases that we want to remove from the embeddings, we implement a **random** baseline where we randomly select target and attribute words from $\mathcal{V}_a \cup \mathcal{V}_t$ and perform **all-token** debiasing. We see that **random** debiases BERT to some extent but is not effective on other models. This result shows that the

Model	MNLI-m	MNLI-mm	NN	FN	T:0.7
Dev et al. (2020)	**80.8**	81.1	85.5	**97.3**	88.3
all-token	80.7	**81.2**	**87.8**	96.8	**89.3**
original	**80.8**	81.0	82.3	96.4	83.2
random	80.5	81.1	85.8	96.4	87.0

Table 2: Debias results for BERT in MNLI.

proposed debiasing method is *not* merely a regularisation technique that imposes constraints on any arbitrary set of words, but it is essential to carefully select the target words used for debiasing.

The results on GLEU show that BERT, DistilBERT and ELECTRA compared to the **original** embeddings, the debiased embeddings report comparable performances in most settings. This confirms that the proposed debiasing method preserves sufficient semantic information contained in the **original** embeddings that can be used to learn accurate prediction models for the downstream NLP tasks.[5] However, the performance of RoBERTa and ALBERT decrease significantly compared to their **original** versions after debiasing. We suspect that these models are more sensitive to fine-tuning and hence lose their pre-trained information during the debiasing process. We defer the development of techniques to address this issue to future research.

4.4 Measuring Bias with Inference

Following Dev et al. (2020), we use the multi-genre co-reference-based natural language inference (MNLI) dataset for evaluating gender bias. This dataset contains sentence triples where a premise must be neutral in entailment w.r.t. two hypotheses. If the predictions made by a classifier that uses word embeddings as features deviate from neutrality, it is considered as biased. Given a set containing M test instances, let the entailment predictor's probabilities for the m-th instance for entail, neutral and contradiction labels be respectively e_m, n_m and c_m. Then, they proposed the following measures to quantify the bias: (1) Net Neutral (**NN**): NN $= \frac{1}{M}\sum_{m=1}^{M} n_m$; (2) Fraction Neutral (**FN**): FN $= \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}[\text{neutral} = \max(e_m, n_m, c_m)]$; and (3) Threshold τ (**T:**τ): T:τ $= \mathbf{1}[n_m \geq \tau]$, where we used $\tau = 0.7$ following Dev et al. (2020). For an ideal (bias-free) embedding, all three measure would be 1.

Model	*Layer*	SEAT-6	SEAT-7	SEAT-8
BERT	all	**0.44**	0.25	**0.46**
	last	0.56	**0.12**	0.47
	first	0.52	0.22	0.49
RoBERTa	all	**0.59**	**0.23**	0.61
	last	0.73	0.24	0.65
	first	0.69	0.28	**0.59**
ALBERT	all	**0.46**	0.48	**0.24**
	last	1.15	**0.26**	0.60
	first	0.54	0.89	0.95
DistilBERT	all	**0.66**	**-0.16**	0.37
	last	0.88	0.19	**0.35**
	first	0.90	0.40	0.52
ELECTRA	all	**0.21**	**0.02**	**0.18**
	last	0.34	0.20	0.21
	first	0.28	0.13	0.34

Table 3: Averaged scores over all layers in an embedding debiased at token-level, measured on SEAT tests.

In Table 2, we compare our proposed method against the *noncontextualised debiasing* method proposed by Dev et al. (2020) where they debias Layer 1 of BERT-large model using an orthogonal projection to the gender direction during training and evaluation. In addition to the above-mentioned measures, we also report the entailment accuracy on the matched (in-domain) and mismatched (cross-domain) denoted respectively by **MNLI-m** and **MNLI-mm** in Table 2 to evaluate the semantic information preserved in the embeddings after debiasing.

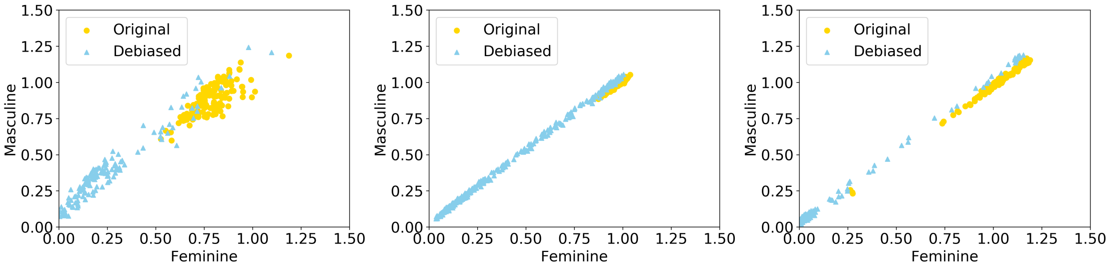
We see that the proposed method outperforms noncontextualised debiasing (Dev et al., 2020) in NN and T:0.7, and its performance of the MNLI task is comparable to the original embeddings. This result further confirms that the proposed method can not only debias well but can also preserve the pre-trained information. Moreover, it is consistent with the results reported in Table 1 and shows that debiasing all layers is more effective than only the first layer as done by Dev et al. (2020).

4.5 The Importance of Debiasing All Layers

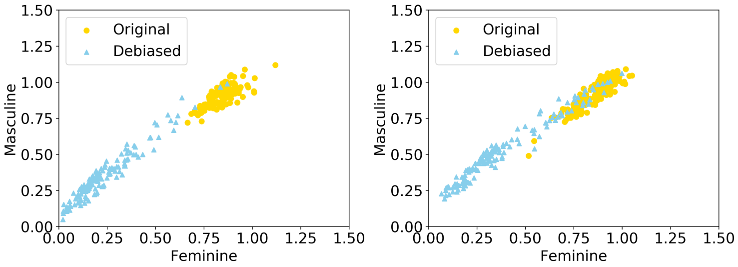
In Table 1, we investigated the bias for the final layer, but it is known that the contextualised embeddings are learned at each layer (Bommasani et al., 2020). Therefore, to investigate whether by debiasing in each layer we are able to remove the biases of the entire contextualised embeddings, we evaluate the debiased embeddings at each layer

---

[5] Although on WNLI **all-token** debiasing improves performance for DistilBERT and ELECTRA compared to the respective **original** models, this is insignificant as WNLI contains only 146 test instances.



(a) BERT (b) RoBERTa (c) ALBERT



(d) DistilBERT (e) ELECTRA

Figure 2: Scatter plot of gender information of hidden states for original and debiased stereotype words.

on SEAT 6, 7, 8 datasets and report the averaged metrics for **all-token**, **first-token** and **last-token** methods in Table 3. We see that, on average, **fitst-token** and **last-token** methods have more bias than **all-token**. Therefore, we conclude that It is not enough to debias only the first and last layers even in DistilBERT, which has a small number of layers. These results show that biases in the entire contextualised embedding cannot be reliably removed by debiasing only some selected layers, but rather the importance of debiasing all layers consistently.

4.6 Visualizing Debiasing Results

To further illustrate the effect of debiasing using the proposed **all-token** method, we visualise the similarity scores of a stereotypical word with feminine and masculine dimensions as follows. First, for each target word t, its hidden state, $E_i(t, x)$ in the i-th layer of the model E in a sentence x is computed. Next, we average those hidden states across all sentences in the dataset that contain t to obtain $\hat{E}_i(t) = \frac{1}{|\mathcal{T}|}\sum_{x \in \mathcal{T}} E_i(t, x)$. Likewise, we compute $\hat{E}_i(f)$ and $\hat{E}_i(m)$ respectively for each feminine (f) and masculine (m) word. Next, we compute, s_i^f, the cosine similarity between each $\hat{E}_i(f)$ and the feminine vector $\boldsymbol{v}_i(f)$, and the cosine similarity, s_i^m, between each $\hat{E}_i(f)$ and the masculine vector $\boldsymbol{v}_i(f)$. $\boldsymbol{s}_i^f$ and $\boldsymbol{s}_i^m$, respectively, are averaged over all layers in a contextualised embedding model to obtain $\boldsymbol{s}_{\text{Avg}}^f$ and $\boldsymbol{s}_{\text{Avg}}^m$, which represent how much gender information each gender word contains on average.

We then compute the cosine similarity, $\boldsymbol{s}_i^{t,f}$, between each stereotype word's averaged embedding, $\hat{E}_i(t)$ and the feminine vector $\boldsymbol{v}_i(f)$. Similarly, we compute the cosine similarity $\boldsymbol{s}_i^{t,m}$ between each stereotype word's averaged embedding $\hat{E}_i(t)$ and the masculine vector $\boldsymbol{v}_i(m)$. We then average $\boldsymbol{s}^{t,f}$ and $\boldsymbol{s}^{t,m}$ over the layers in E respectively, to compute $\boldsymbol{s}_{\text{Avg}}^{t,f}$ and $\boldsymbol{s}_{\text{Avg}}^{t,m}$, which represent how much gender information each stereotype word contains on average. Finally, we visualise the normalised female and male gender scores given respectively by $\boldsymbol{s}_{\text{Avg}}^{t,f} / \boldsymbol{s}_{\text{Avg}}^f$ and $\boldsymbol{s}_{\text{Avg}}^{t,m} / \boldsymbol{s}_{\text{Avg}}^m$. For example, a zero $\boldsymbol{s}_{\text{Avg}}^{t,f} / \boldsymbol{s}_{\text{Avg}}^f$ value indicates that t does not contain female gender related information, whereas a value of one indicates that it contains all information about the female gender. Figure 2 shows each stereotype word with its normalised female ad male gender scores respectively in x and y axises. For a word, a yellow circle denotes its original embeddings, and the blue triangle denotes the result of debiasing using the **all-token** method.

We see that with the original embeddings, stereotypical words of are distributed close to one, indicating that they are highly gender-specific. On the other hand, we see that the debiased BERT, DistilBERT and ELECTRA have similar word distributions compared to the original embeddings respectively, with an overall movement towards zero. On the other hand, for RoBERTa, debiased embeddings are mainly distributed from zero to around one compared to the original embeddings. Moreover, for ALBERT, the debiased embeddings are close to zero, but unlike the original distribution, the debiased embeddings are mainly clustered around zero. This shows that RoBERTa and ALBERT do not retain structure of the original distribution after debiasing. While ALBERT over-debiases pre-trained embeddings of stereotypical words, RoBERTa under-debiases them. This trend was already confirmed on the downstream evaluation tasks conducted in Table 1.

5 Conclusion

We proposed a debiasing method for pre-trained contextualised word embeddings, operating at token- or sentence-levels. Our experimental results showed that the proposed method effectively debiases discriminative gender-related biases, while preserving useful semantic information in the pre-trained embeddings. The results showed that the downstream task was more effective in debias than the previous studies.

References

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, and Danilo Giampiccolo. 2006. The second pascal recognising textual entailment challenge. *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09*.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NIPS*.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 7–15, Minneapolis, Minnesota. Association for Computational Linguistics.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

K. Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *ArXiv*, abs/2003.10555.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, pages 177–190, Berlin, Heidelberg. Springer-Verlag.

Sunipa Dev, Tao Li, Jeff Phillips, and Vivek Srikumar. 2020. On Measuring and Mitigating Biased Inferences of Word Embeddings. In *AAAI*.

Sunipa Dev and Jeff M. Phillips. 2019. Attenuating bias in word vectors. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 879–887. PMLR.

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

Simon De Deyne, Danielle J. Navarro, Amy Perfors, Marc Brysbaert, and Gert Storms. 2019. The "small world of words" english word association norms for over 12,000 cue words. *Behavior Research Methods*, 51:987–1006.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Yupei Du, Yuanbin Wu, and Man Lan. 2019. Exploring human gender stereotypes with word association test. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6133–6143, Hong Kong, China. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial Removal of Demographic Attributes from Text Data. In *Proc. of EMNLP*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. *The Third PASCAL Recognizing Textual Entailment Challenge*, pages 1–9. Association for Computational Linguistics, USA.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5266–5274, Hong Kong, China. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2019. Gender-preserving debiasing for pre-trained word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2020. Autoencoding improves pre-trained word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1699–1713, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Saket Karve, Lyle Ungar, and João Sedoc. 2019. Conceptor debiasing of word representations evaluated on WEAT. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 40–48, Florence, Italy. Association for Computational Linguistics.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pages 552–561. AAAI Press.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, and Jeffrey Dean. 2013. Efficient estimation of word representation in vector space. In *ICLR*.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. StereoSet: Measuring stereotypical bias in pretrained language models.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-Pairs: A Challenge Dataset for Measuring Social Biases in Masked Language Models. In *Proc. of EMNLP*.

Jeffery Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: global vectors for word representation. In *EMNLP*, pages 1532–1543.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null It Out: Guarding Protected Attributes by Iterative Nullspace Projection. In *Proc. of ACL*.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yi Chern Tan and L. Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. In *Advances in Neural Information Processing Systems 32*, pages 13230–13241. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Proc. of NIPS*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634, Minneapolis, Minnesota. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018a. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20. Association for Computational Linguistics.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018b. Learning Gender-Neutral Word Embeddings. In *Proc. of EMNLP*, pages 4847–4853.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2003. Learning with local and global consistency. In *NIPS*.

Ran Zmigrod, Sebastian J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual Data Augmentation for Mitigating Gender Stereotypes in Languages with Rich Morphology. In *Proc. of ACL*.

Language Models for Lexical Inference in Context

Martin Schmitt[1] and **Hinrich Schütze**[2]
Center for Information and Language Processing (CIS)
LMU Munich, Germany
[1]`martin@cis.lmu.de` [2]`inquiries@cislmu.org`

Abstract

Lexical inference in context (LIiC) is the task of recognizing textual entailment between two very similar sentences, i.e., sentences that only differ in one expression. It can therefore be seen as a variant of the natural language inference task that is focused on lexical semantics. We formulate and evaluate the first approaches based on pretrained language models (LMs) for this task: (i) a few-shot NLI classifier, (ii) a relation induction approach based on handcrafted patterns expressing the semantics of lexical inference, and (iii) a variant of (ii) with patterns that were automatically extracted from a corpus. All our approaches outperform the previous state of the art, showing the potential of pretrained LMs for LIiC. In an extensive analysis, we investigate factors of success and failure of our three approaches.[1]

1 Introduction

Lexical inference (LI) denotes the task of deciding whether or not an entailment relation holds between two lexical items. It is therefore related to the detection of other lexical relations like hyponymy between nouns (Hearst, 1992), e.g., *dog* ⇒ *animal*, or troponymy between verbs (Fellbaum and Miller, 1990), e.g., *to traipse* ⇒ *to walk*. Lexical inference in context (LIiC) adds the problem of disambiguating the pair of lexical items in a given context before reasoning about the inference question. This type of LI is particularly interesting for entailments between verbs and verbal expressions because their meaning – and therefore their implications – can drastically change with different arguments. Consider, e.g., *run* ⇒ *lead* in a PERSON / COMPANY context ("Bezos runs Amazon") vs. *run* ⇒ *execute* in a COMPUTER / SOFTWARE context ("My mac runs macOS"). LIiC is thus also closely related to the task of natural language inference (NLI) – also called recognizing textual entailment (Dagan et al., 2013) – and can be seen as a focused variant of it. Besides the important use case of evaluating NLI systems, this kind of predicate entailment has also been shown useful for question answering (Schoenmackers et al., 2010), event coreference (Shwartz et al., 2017; Meged et al., 2020), and link prediction in knowledge graphs (Hosseini et al., 2019).

Despite its NLI nature, previous systems for LIiC have primarily been models of lexical similarity (Levy and Dagan, 2016) or models based on verb argument inclusion (Hosseini et al., 2019). The reason is probably that supervised NLI models need large amounts of training data, which is unavailable for LIiC, and that systems trained on available large-scale NLI benchmarks (e.g., Williams et al., 2018) have been reported to insufficiently cover lexical phenomena (Glockner et al., 2018; Schmitt and Schütze, 2019).

Recently, transfer learning has become ubiquitous in NLP; Transformer (Vaswani et al., 2017) language models (LMs) pretrained on large amounts of textual data (Devlin et al., 2019a; Liu et al., 2019) form the basis of a lot of current state-of-the-art models. Besides zero- and few-shot capabilities (Radford et al., 2019; Brown et al., 2020), pretrained LMs have also been found to acquire factual and relational knowledge during pretraining (Petroni et al., 2019; Bouraoui et al., 2020). The entailment relation certainly stands out among previously explored semantic relations – such as the relation between a country and its capital – because it is very rarely stated explicitly and often involves reasoning about both the meaning of verbs and additional knowledge (Schmitt and Schütze, 2019). It is unclear whether implicit clues during pretraining are enough to learn about LIiC and what the best way is to harness any such implicit knowledge.

Regarding these questions, we make the follow-

---

[1] Our code is publicly available: `https://github.com/mnschmit/lm-lexical-inference`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1267–1280
April 19 - 23, 2021.

ing contributions: (1) This work is the first to explore the use of pretrained LMs for the LIiC task. (2) We formulate three approaches and evaluate them using the publicly available pretrained RoBERTa LM (Liu et al., 2019; Wolf et al., 2019): (i) a few-shot NLI classifier, (ii) a relation induction approach based on handcrafted patterns expressing the semantics of lexical inference, and (iii) a variant of (ii) with patterns that were automatically extracted from a corpus. (3) We introduce the concept of antipatterns, patterns that express non-entailment, and evaluate their usefulness for LIiC. (4) In our experiments on two established LIiC benchmarks, Levy/Holt's dataset (Levy and Dagan, 2016; Holt, 2018) and SherLIiC (Schmitt and Schütze, 2019), all our approaches consistently outperform previous work, thus setting a new state of the art for LIiC. (5) In contrast to previous work on relation induction (Bouraoui et al., 2020), automatically retrieved patterns do not outperform handcrafted ones for LIiC. A qualitative analysis of patterns and errors identifies possible reasons for this finding.

2 Related Work

Lexical inference. There has been a lot of work on lexical inference for nouns, notably hypernymy detection, resulting in a variety of benchmarks (Kotlerman et al., 2010; Kiela et al., 2015) and methods (Shwartz et al., 2015; Vulić and Mrkšić, 2018). Although there has been work on predicate entailment before (Lin and Pantel, 2001; Lewis and Steedman, 2013), Levy and Dagan (2016) were the first to create a general benchmark for evaluating entailment between verbs. In their evaluation, neither resource-based approaches (Pavlick et al., 2015; Berant et al., 2011) nor vector space models (Levy and Goldberg, 2014) achieved satisfying results. Holt (2018) later published a re-annotated version, which was readily adopted by later work. Hosseini et al. (2018) put global constraints on top of directed local similarity scores (Weeds and Weir, 2003; Lin, 1998; Szpektor and Dagan, 2008) based on distributional features of the predicates. Hosseini et al. (2019) replaced these scores by transition probabilities in a bipartite graph where edge weights are computed by a link prediction model.

When Schmitt and Schütze (2019) created the SherLIiC benchmark, they also mainly focused on resource- and vector-based models for evaluation. Their best model combines general-purpose word2vec representations (Mikolov et al., 2013) with a vector representation of the arguments that co-occur with a predicate.

All these works (i) base the probability of entailment validity on the similarity of the verbs and (ii) compute this similarity via (expected) co-occurrence of verbs and their arguments. Our work differs in that our models solely reason about the sentence surface in an end-to-end NLI task without access to previously observed argument pairs. This is possible because our models have learned about these surface forms during pretraining.

Patterns and entailment. Pattern-based approaches have long been known for hypernymy detection (Hearst, 1992). Recent work combined them with vector space models (Mirkin et al., 2006; Roller and Erk, 2016; Roller et al., 2018). While there are effective patterns, such as X *is a* Y, that are indicative for entailment between nouns, there is little work on comparable patterns for verbs. Schwartz et al. (2015) mine symmetric patterns for lexical similarity and achieve good results for verbs. Entailment, however, is not symmetric.

Chklovski and Pantel (2004) handcrafted 35 patterns to distinguish 6 semantic relations for pairs of distributionally similar verbs. Some of their classes like strength (*taint :: poison*) or antonymy (*ban :: allow*) can be indicators of entailment and non-entailment but are, in general, much more narrowly defined than the patterns we use in our approach. Another difference to our work is that verb pairs are scored based on co-occurrence counts on the web, while we employ an LM, which does not depend on a valid entailment pair actually appearing together in a document.

Patterns and language models. Amrami and Goldberg (2018) were the first to manipulate LM predictions with a simple pattern to enhance the quality of substitute words in a given context for word sense induction. Petroni et al. (2019) found that large pretrained LMs can be queried for factual knowledge, when presented with appropriate pattern-generated cloze-style sentences. This zero-shot factual knowledge has later been shown to be quite fragile (Kassner and Schütze, 2020). So we rather focus on approaches that fine-tune an LM on at least a few samples. Forbes et al. (2019) train a binary classifier on top of a fine-tuned BERT (Devlin et al., 2019a) to predict the truth value of handwritten statements about objects and their properties. While their experiments investigate BERT's physical common sense reasoning, we focus on the different phenomenon of entailment between two actions expressed by verbs in context.

Schick and Schütze (2020) used handcrafted patterns and LMs for few-shot text classification. Based on manually defined label-token correspondences, the predicted classification label is determined by the token an LM estimates as most probable at a masked position in the cloze-style pattern. We differentiate entailment and non-entailment via compatibility scores for patterns and antipatterns and not via different predicted tokens.

Addressing relation induction, Bouraoui et al. (2020) propose an automatic way of finding, given a relation, LM patterns that are likely to express it. They train a binary classifier per relation on the sentences generated by these patterns. While some of the relations they consider are related to verbal entailment (e.g., *cook activity-goal eat*), most of them concern common sense (e.g., *library location-activity reading*) or encyclopedic knowledge (e.g., *Paris capital-of France*). We adapt their method for the automatic retrieval of promising patterns for LIiC, but find that handcrafted patterns that capture the generality of the entailment relation still have an advantage over automatic patterns for LIiC. Another important novelty we introduce is the use of antipatterns. While Bouraoui et al. (2020) have to use negative samples for training their classifiers, they only consider patterns that exemplify the desired relation. In contrast, we also use antipatterns that exemplify what the entailment relation is **not**. We believe that antipatterns are particularly useful for entailment detection because they can help identify other kinds of semantic relations that often pose a challenge to vector space models (Levy and Dagan, 2016; Schmitt and Schütze, 2019).

3 Proposed Approaches

3.1 NLI classifier

Building an NLI classifier on top of a pretrained LM usually means taking an aggregate sequence representation of the concatenated premise and hypothesis as input features of a neural network classifier (Devlin et al., 2019b). For RoBERTa (Liu et al., 2019), this representation is the final hidden state of a special $\langle \mathtt{s} \rangle$ token that is prepended to the input sentences, which in turn are separated by a separator token $\langle / \mathtt{s} \rangle$. Let Λ be the function that maps such an input $x = x_1 \langle / \mathtt{s} \rangle x_2$ to the aggregate representation $\Lambda(x) \in \mathbb{R}^d$. Following (Devlin et al., 2019b; Liu et al., 2019), we then feed these features to a 2-layer feed-forward neural network with tanh activation:

$$\begin{aligned} h(x) &= \tanh(\mathrm{drop}(\Lambda(x)) W_1 + b_1) \\ P_{\text{NLI}}(y \mid x) &= \sigma(\mathrm{drop}(h(x)) W_2 + b_2) \end{aligned} \tag{1}$$

where drop applies dropout with a probability of 0.1, σ is the softmax function, and $W_1 \in \mathbb{R}^{d \times d}, W_2 \in \mathbb{R}^{d \times 2}, b_1 \in \mathbb{R}^d, b_2 \in \mathbb{R}^2$ are learnable parameters. Note that W_1 and b_1 are still part of the LM's pretrained parameters; so we only train W_2 and b_2 from scratch.[2] The actual classification decision uses a threshold ϑ:

$$D^{\vartheta}_{\text{NLI}}(x_1, x_2) = \begin{cases} 1, & \text{if } P_{\text{NLI}}(y = 1 \,|\, x_1, x_2) > \vartheta \\ 0, & \text{otherwise} \end{cases}$$

The traditional choice for the threshold is $\vartheta = 0.5$ because that means $D^{\vartheta}_{\text{NLI}}(x_1, x_2) = 1$ iff $P_{\text{NLI}}(y = 1 \mid x_1, x_2) > P_{\text{NLI}}(y = 0 \mid x_1, x_2)$. We nevertheless keep ϑ as a hyperparameter to be tuned on held-out development data.

We train the NLI approach by minimizing the negative log-likelihood $\mathcal{L}_{\text{NLI}}$ of the training data $\mathcal{T}$:

$$\mathcal{L}_{\text{NLI}}(\mathcal{T}) = \sum_{(x_1, x_2, y) \in \mathcal{T}} -\log(P_{\text{NLI}}(y \mid x_1, x_2))$$

3.2 Pattern-based classifier

This approach puts the input sentences x_1, x_2 together in a pattern-based textual context and trains a classifier to distinguish between felicitous and infelicitous utterances.[3] In contrast to previous approaches (Forbes et al., 2019; Bouraoui et al., 2020), we also consider antipatterns that exemplify what kind of semantic relatedness we are not interested in, and combine probabilities for patterns and antipatterns in the final classification.

Finding suitable patterns. A simple handcrafted pattern to check for the validity of an inference $x_1 \Rightarrow x_2$ is "x_2 *because* x_1.". An analoguos antipattern is "*It is not sure that* x_2 *just because* x_1.". Based on similar considerations, we manually design 5 patterns and 5 antipatterns (see Table 4). We will refer to the approach using these handcrafted patterns as MANPAT.

Bouraoui et al. (2020) argue that text produced by simple, handcrafted patterns is artificial and

---

[2] We follow the official implementation; cf. Jacob Devlin's comment on issue 43 in the BERT GitHub repository, `https://github.com/google-research/bert/issues/43`, (accessed 19 January 2021).

[3] Bouraoui et al. (2020) called this natural vs. unusual.

therefore suboptimal for LMs pretrained on naturally occurring text. To adapt their setup to verbal expressions used in LIiC, we identify suitable patterns (antipatterns) by searching a large text corpus[4] for sentences that contain both elements of valid (invalid) entailment pairs. In a second step, we score each of these patterns (antipatterns) according to the number of valid (invalid) entailment pairs x_1, x_2 that can be found by querying an LM for the k most probable completions when x_1 or x_2 is inserted in the pattern and its counterpart is masked. For example, consider the entailment pair *rule* ⇒ *control* and the pattern "*Catchers* ***prem*** *the field; they* ***hypo*** *the plays and tell everyone where to be.*" extracted from a description of softball. Predicting *rule* from "*Catchers* ⟨`mask`⟩ *the field; they control the plays and tell everyone where to be.*" and predicting *control* from "*Catchers rule the field; they* ⟨`mask`⟩ *the plays and tell everyone where to be.*" would result in one point each. Approaches called AUTPAT$_n$ use the n patterns with the most points obtained in that manner. See §4 for more details on our experimental setup.

Pattern-based predictions. The probability $P_{\text{FEL}}(z \mid x)$ of sentence x to be felicitous ($z=1$) or infelicitous ($z=0$) is estimated like P_{NLI} in Eq. (1), except that x is not the concatenation of two sentences but a single pattern-generated utterance.

Given a set of patterns Φ and a set of antipatterns Ψ, the score s to judge an input x_1, x_2 is the difference between the maximum probability m_{pos} that any pattern forms a felicitous statement and the maximum probability m_{neg} that any antipattern forms a felicitous statement:

$$\begin{aligned} m_{\text{pos}} &= \max_{\varphi \in \Phi} P_{\text{FEL}}(z = 1 \mid \varphi(x_1, x_2)) \\ m_{\text{neg}} &= \max_{\psi \in \Psi} P_{\text{FEL}}(z = 1 \mid \psi(x_1, x_2)) \\ s(x_1, x_2) &= m_{\text{pos}} - m_{\text{neg}} \end{aligned}$$

As in NLI, the final decision uses a threshold ϑ:

$$D^{\vartheta}_{\text{PAT}}(x_1, x_2) = \begin{cases} 1, & \text{if } s(x_1, x_2) > \vartheta \\ 0, & \text{otherwise} \end{cases}$$

This corresponds to requiring that m_{pos} be higher than m_{neg} by a margin ϑ, i.e., $D^{\vartheta}_{\text{PAT}}(x_1, x_2) = 1$ iff $m_{\text{pos}} > m_{\text{neg}} + \vartheta$.

As Bouraoui et al. (2020) did not use antipatterns, they defined m_{neg} as the maximum probability for any pattern to form an infelicitous statement.

		Levy/Holt	SherLIiC
dev$_1$	train	4,388	797
	dev$_2$	1,098	201
test		12,921	2,990

Table 1: Data split sizes as used in our experiments.

To estimate the usefulness of antipatterns, we evaluate both possibilities, marking systems that use both patterns and antipatterns with ΦΨ and those that only use patterns with Φ.

The use of a threshold is another novel component, i.e., Bouraoui et al. (2020) virtually set $\vartheta = 0$. We discuss the influence of ϑ in §5.

We train all pattern-based approaches by minimizing the negative log-likelihood $\mathcal{L}_{\text{PAT}}$ that patterns Φ produce felicitous statements for valid entailments ($y = 1$) and infelicitous statements for invalid entailments ($y = 0$) from the training data $\mathcal{T}$, and vice versa for antipatterns Ψ:

$$\mathcal{L}_{\text{PAT}}(\mathcal{T}, \Phi, \Psi) = \sum_{(x_1, x_2, y) \in \mathcal{T}} \mathcal{L}_{\Phi}(x_1, x_2, y) + \mathcal{L}_{\Psi}(x_1, x_2, 1 - y)$$

with

$$\mathcal{L}_{\Omega}(x_1, x_2, y) = -\frac{1}{|\Omega|} \sum_{\omega \in \Omega} \log(P_{\text{FEL}}(z = y \mid \omega(x_1, x_2)))$$

4 Experiments

We evaluate on two benchmarks: (i) Levy/Holt's dataset (Levy and Dagan, 2016; Holt, 2018) and (ii) SherLIiC (Schmitt and Schütze, 2019). For both filtering and classification, we employ RoBERTa-base (Liu et al., 2019). For classification only, we also report results for RoBERTa-large.

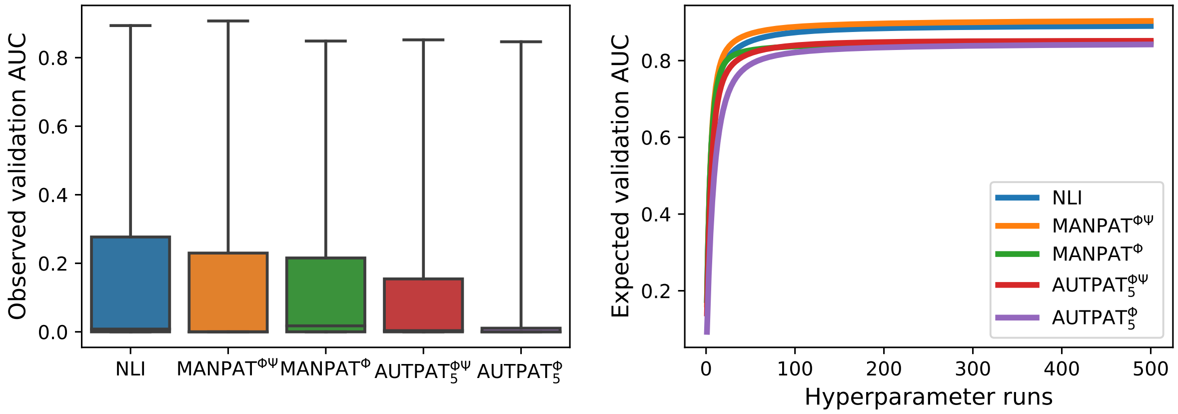
4.1 Data processing

For both datasets, previous work has established a dev/test split. For Levy/Holt, it was defined in (Hosseini et al., 2018); for SherLIiC, we use the original one from (Schmitt and Schütze, 2019). For comparison with previous work, we keep the test portion as is and split the dev portion further into 80% for training and 20% for development. We call the new, smaller dev sets dev$_2$ and the original dev sets dev$_1$. See Table 1 for data split sizes.

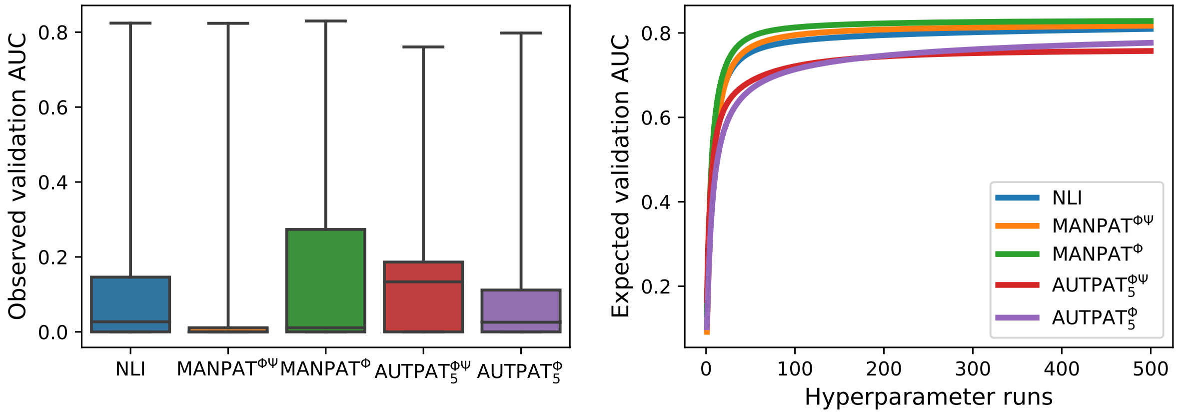
Levy/Holt. An instance in Levy/Holt has two sentences, each consisting of two shared noun

---

[4] We use the Wikipedia dump from Jan 15th 2011.



(a) Levy/Holt dev$_2$ RoBERTa-base



(b) SherLIiC dev$_2$ RoBERTa-base

Figure 1: Validation performance distribution of different datasets across different hyperparameter runs (left) and expected validation performance per number of tested hyperparameter configurations as proposed by Dodge et al. (2019) (right). Performance is measured as the area under the precision-recall curve for precision values ≥ 0.5. The Boxes represent 75% of the respective data points; a black line indicates the median, whiskers extend to the maximum value.

phrases (the arguments) and a verbal expression, in which the two sentences differ. As the verbal expressions can contain auxiliaries or negation, they often consist of multiple tokens. Originally, one argument is replaced with a WordNet (Miller, 1995) type in one of the sentences to make the entailment more general during annotation, but we use a version of the dataset provided by Hosseini et al. (2018) where both sentences have concretely instantiated arguments. For example, consider Table 6 (c). *Athena* was masked as the WordNet synset *deity* during benchmark annotation but we use the original sentences as shown in Table 6 for all classifiers without further modification.

For the automatic pattern search in AUTPAT, we look for sentences that mention verbatim the two verbal expressions of any instance from dev$_1$. For the ranking, we take the last token of a verbal expression as representative for the whole. This has the advantage that we can query the LM with a single $\langle$`mask`$\rangle$ token and compare a single token to the $k = 100$ most probable predictions. We take the last token because it usually is the main verb.

SherLIiC. For classification, we use SherLIiC's automatically generated sentences that were used for annotation during benchmark creation. The arguments in SherLIiC are entity types from Freebase (Bollacker et al., 2008). As such, they can be replaced by any Freebase entity with matching type. For example, consider Table 6 (a); the arguments *Germany* and *Côte d'Ivoire* were originally masked as *location[A]* and *location[B]* during annotation, but annotators also saw three randomly chosen instantiations for both *A* (*Germany / Syria / USA*) and *B* (*Côte d'Ivoire / UK / Italy*) for context. From the three examples provided in SherLIiC for each argument, we choose the first one to form sentences with concretely instantiated arguments.

For the automatic pattern search in AUTPAT, we make use of the greater flexibility offered by the

		AUC	P	R	F1
baselines					
Hosseini et al. (2018)		16.5	–	–	–
Hosseini et al. (2019)		**18.7**	–	–	–
RoBERTa-base					
NLI	($\vartheta = 0.0052$)	72.6	68.7	**75.3**	71.9
MANPAT$^{\Phi\Psi}$	($\vartheta = -0.0909$)	**76.9**	**78.7**	66.4	**72.0**
MANPAT$^{\Phi}$	($\vartheta = 0.5793$)	71.2	74.4	61.2	67.1
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.1428$)	63.7	71.0	58.8	64.3
AUTPAT$_5^{\Phi}$	($\vartheta = -0.0592$)	65.4	68.0	63.3	65.5
RoBERTa-large					
NLI	($\vartheta = 0.0016$)	75.5	73.5	73.7	73.6
MANPAT$^{\Phi\Psi}$	($\vartheta = 0.1156$)	**83.9**	**84.8**	70.1	**76.7**
MANPAT$^{\Phi}$	($\vartheta = -0.8457$)	77.8	67.9	**81.5**	74.1
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.0021$)	70.4	75.7	60.7	67.4
AUTPAT$_5^{\Phi}$	($\vartheta = -0.9197$)	66.5	61.8	74.4	67.5

Table 2: Levy/Holt test. AUC denotes the area under the precision-recall curve for precision ≥ 0.5. All results in %. Bold means best result per column and block.

		AUC	P	R	F_1
baselines					
Lemma		–	**90.7**	8.9	16.1
w2v+untyped_rel		–	52.8	69.5	60.0
w2v+tsg_rel_emb		–	51.8	**72.7**	**60.5**
RoBERTa-base					
NLI	($\vartheta = 0.3878$)	65.8	**67.0**	66.1	66.5
MANPAT$^{\Phi\Psi}$	($\vartheta = -0.3324$)	66.4	60.9	78.8	68.7
MANPAT$^{\Phi}$	($\vartheta = -0.4812$)	**69.2**	62.0	**81.2**	**70.3**
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.4694$)	67.4	61.8	75.6	68.0
AUTPAT$_5^{\Phi}$	($\vartheta = -0.7042$)	67.3	56.6	82.6	67.2
AUTCUR$_5^{\Phi}$	($\vartheta = -0.7524$)	**69.5**	56.3	**89.6**	**69.2**
AUTARG$_5^{\Phi}$	($\vartheta = -0.7461$)	65.2	**61.9**	75.6	68.1
RoBERTa-large					
NLI	($\vartheta = 0.0025$)	68.3	60.5	**85.5**	70.9
MANPAT$^{\Phi\Psi}$	($\vartheta = -0.0956$)	**74.4**	**66.0**	80.8	**72.6**
MANPAT$^{\Phi}$	($\vartheta = -0.6641$)	64.6	58.1	79.0	67.0
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.9889$)	68.6	61.9	75.5	68.0
AUTPAT$_5^{\Phi}$	($\vartheta = -0.5355$)	56.8	61.5	66.1	63.7

Table 3: SherLIiC test. Baseline results from (Schmitt and Schütze, 2019). Table format: see Table 2.

lemmatized representations in SherLIiC. As we are interested in statements that can be made in any way in a text, we search for sentences that mention the two predicates of a SherLIiC dev_1 instance in any inflected form. For the ranking, we again consider the predicate representative for the whole verbal expression. We thus use the predicate lemma and otherwise proceed as described above.

4.2 Training details

We train all our classifiers for 5 epochs with Adam (Kingma and Ba, 2015) and a mini-batch size of 10 (resp. 2) for RoBERTa-base (resp. -large). We randomly sample 500 configurations for the remaining hyperparameters (see Appendix A). For a fair comparison, we evaluate all our approaches with the same configurations.

5 Results and Discussion

5.1 Hyperparameter robustness

Following previous work (Hosseini et al., 2018, 2019), we use the area under the precision-recall curve (AUC) restricted to precision values ≥ 0.5 as criterion for model selection.

Fig. 1 (left) shows the distribution of dev_2 performance for 500 randomly sampled runs with RoBERTa-base. Most hyperparameters perform poorly, suggesting that hyperparameter search is crucial. For Levy/Holt, NLI is strong whereas for SherLIiC handcrafted MANPAT$^{\Phi}$ patterns have a clearer advantage. For SherLIiC, the combination of automatically generated patterns and antipatterns AUTPAT$_5^{\Phi\Psi}$ exhibits the highest median performance and the second-highest upper quartile, making it together with MANPAT$^{\Phi}$ the most robust to different hyperparameters, although its top performance is lower compared to the others. For all methods, only very few hyperparameter sets achieve top performances. For both datasets, however, a well-performing configuration is found after fewer than 100 sampled runs (Fig. 1, right). Considering that AUTPAT requires an LM to rank thousands of patterns, these results suggest that, for LIiC, available GPU hours should be spent on automatic hyperparameter rather than pattern search. With its manually written patterns, MANPAT does not need additional GPU hours for pattern search and still, on average, performs better.

5.2 Best hyperparameter configurations

For the best found configuration for each method, we not only report AUC, which provides a general picture of a scoring method's precision-recall tradeoff, but also the concrete precision, recall, and F1 for the actual classification after applying a threshold ϑ. For this we tune ϑ on dev_2 for optimal F1. Tables 2 and 3 show the results.

On both datasets, our methods outperform all previous work (sometimes by a large margin), thus establishing a new state of the art. For SherLIiC+RoBERTa-base, the strong but simple NLI system is consistently outperformed by all pattern-based approaches, showing that well-

Automatically retrieved patterns (with SherLIiC dev$_1$)		prem	hypo
rank 1	In North America, where the ”atypical” forms of community-**hypo** pneumonia are becoming more common, macrolides (such as azithromycin), and doxycycline have displaced amoxicillin as first-line outpatient treatment for community-**prem** pneumonia.	acquired	acquired
rank 5	This area now consists of … the Yukon Territory (**prem** 1898) … and Nunavut (**hypo** 1999).	created	created in
rank 12	For example, … 訪問 ”**prem**” is composed of 訪 ”to visit” and 問 ”to **hypo**”.	interview	ask
Handcrafted patterns			
(a)	PARGL **prem** PARGR, which means that HARGL **hypo** HARGR.		
(b)	It is not the case that HARGL **hypo** HARGR, let alone that PARGL **prem** PARGR.		
(c)	HARGL **hypo** HARGR because PARGL **prem** PARGR.		
(d)	PARGL **prem-negated** PARGR because HARGL **hypo-negated** HARGR.		
(e)	HARGL **hypo-negated** HARGR, which means that PARGL **prem-negated** PARGR.		

Table 4: Examples of automatically retrieved and ranked AUTPAT patterns (top) and handcrafted MANPAT patterns (bottom). prem/hypo = original fillers as found in the corpus. PARGL/HARGL = placeholder for left argument of premise/hypothesis; PARGR/HARGR = right argument.

chosen patterns and antipatterns can be helpful for LIiC. For SherLIiC+RoBERTa-large and also generally on Levy/Holt's dataset, NLI is more competitive, but the combination of handcrafted patterns and antipatterns MANPAT$^{\Phi\Psi}$ still performs better in these cases.

The use of antipatterns does not consistently lead to better performance for all combinations of dataset, LM variant (base vs. large), and pattern set (MANPAT vs. AUTPAT). They do, however, consistently bring gains for some combinations, e.g., MANPAT on Levy/Holt and AUTPAT on SherLIiC. Moreover, antipatterns are essential for achieving top performance, i.e., the new state of the art, on both datasets.

Most of the threshold values ϑ (tuned on dev$_2$) are far from their traditional values, 0.5 for NLI and 0.0 for patterns. NLI classifiers' probability estimates are often too confident, resulting in values close to 0 and 1. To "correct" cases where a very small value is assigned to a valid entailment, optimal thresholds are often close to 0 instead of 0.5. Analogously, most pattern-based approaches opt for a negative ϑ, which means that instead of requiring a margin between m_{pos} and m_{neg} (boosting precision), they make more positive predictions and boost recall. Low recall is a key problem in LIiC (cf. Levy and Dagan (2016)). Tuning a threshold increases the models' flexibility in this aspect.

6 Analysis

6.1 Number of patterns

§5 shows that automatic patterns do not beat handcrafted patterns for LIiC. However, automatic patterns have one major advantage: in contrast to manual patterns, their number can be easily increased. We therefore investigate the impact of the hyperparameter n for AUTPAT$_n$.

	$\Phi\Psi$		Φ	
n	AUC	F1	AUC	F1
5	67.4	68.0	67.3	67.2
15	**70.0**	**68.7**	**73.1**	**69.4**
25	63.5	67.3	69.0	68.7
50	66.3	65.6	67.4	67.6

Table 5: RoBERTa-base+AUTPAT$_n$ results on SherLIiC test for different n values. Hyperparameters were tuned for the corresponding AUTPAT$_5$ method on dev$_2$.

Table 5 shows that too many patterns is as bad as too few. AUTPAT$_{15}$ is the sweetspot: on SherLIiC, it outperforms all other RoBERTa-base methods on AUC and closely approaches the otherwise best method MANPAT$^{\Phi}$ on F1.

6.2 Pattern analysis

Handcrafted patterns mostly outperform automatic ones (§5). A larger number n of patterns only has a small effect (§6.1). We therefore take a closer look at automatic and manual patterns. Table 4 shows all handcrafted and a sample of highly ranked automatic patterns.

It is striking how specific the automatically retrieved contexts are; especially for the highest ranks (exemplified by ranks 1 and 5) only a narrow set of verbs seems plausible from a human perspective. It is only at rank 12 that we find a more general context and it arguably even displays some semantic reasoning. There certainly are verbs that are not compatible with the meaning of *visit*, but this context allows for a wide range of plausible verbs and even mentions composition of meaning.

The handcrafted patterns, in contrast, all capture some general aspect of entailment, which might be the reason they generalize better. Moreover, they also have placeholder slots for the verb arguments, which could be an advantage as these represent a verb's original context. Only accepting corpus sentences in which the verbs occur with the same arguments as in the dataset is too restrictive.

We therefore conduct the following experiment: We manually go through the 100 highest-ranked automatically created patterns and identify 5 contexts that could accommodate arguments without changing the overall sentence structure. We also try to pick patterns that are different enough from each other to avoid redundancy. As a baseline, the method AUTCUR$^{\Phi}_{5}$ uses these manually curated patterns as is. We then rewrite the patterns such that they include placeholders for verb arguments, e.g., “*The original aim of de Garis' work was to* ***prem*** *the field of ”brain building” (a term of his invention) and to ”****hypo*** *a trillion dollar industry within 20 years”*.” becomes “*The original aim of their work was that ”*PARGL ***prem*** PARGR*” and that ”*HARGL ***hypo*** HARGR *within 20 years”*.” with PARGL / PARGR (HARGL / HARGR) the placeholder for the left / right argument of the premise (hypothesis). See Table 14 in the appendix for the complete list. AUTARG$^{\Phi}_{5}$ is based on these rewritten patterns. We try the same 500 hyperparameter configurations as for the other RoBERTa-base approaches and include results for the best configuration (chosen on dev$_2$) in Table 3. We find that manually curating automatically ranked patterns helps performance. AUTCUR$^{\Phi}_{5}$ outperforms AUTPAT$^{\Phi}_{5}$ on AUC and F1, reducing the gap to handcrafted patterns (i.e., MANPAT$^{\Phi}$). This is probably due to the variety we enforced when handpicking the patterns.

Surprisingly, adding arguments decreases performance. Possibly, our modifications make the patterns less fluent or the arguments that are filled into the placeholders during training and evaluation do not fit well into the contexts, which still are rather specific.

6.3 Error analysis

Table 6 displays a selection of the dev$_2$ sets of our two benchmarks along with the predictions of all our approaches.

(a) Sh	*Germany is occupying Côte d'Ivoire* ⇒ *Germany is remaining in Côte d'Ivoire*	truth: 1	NLI: 0	MANPAT: 1 / 0	AUTPAT: 1 / 1
(b) L/H	*Ford awarded him the medal* ⇒ *Ford was awarded a medal*	truth: 0	NLI: 1	MANPAT: 0 / 0	AUTPAT: 1 / 1
(c) L/H	*Athena was worshiped in Athens* ⇒ *Athena was the goddess of Athens*	truth: 0	NLI: 0	MANPAT: 0 / 1	AUTPAT: 1 / 1
(d) L/H	*Pyrrhus was beaten by the romans* ⇒ *Pyrrhus fought the romans*	truth: 1	NLI: 1	MANPAT: 0 / 0	AUTPAT: 0 / 1
(e) Sh	*England national rugby union team is playing against Denver Broncos* ⇒ *England national rugby union team is beating Denver Broncos*	truth: 0	NLI: 1	MANPAT: 1 / 1	AUTPAT: 1 / 1
(f) L/H	*Polk negotiated with Britain* ⇒ *Polk made peace with Britain*	truth: 0	NLI: 1	MANPAT: 1 / 1	AUTPAT: 1 / 1

Table 6: Qualitative error analysis of the RoBERTa-base models from Tables 2 and 3 on the dev$_2$ split of SherLIiC (Sh) and Levy/Holt (L/H). Pattern-based predictions are listed in the format $\Phi\Psi$ / Φ. Correct predictions are green; errors are underlined and red.

The first four examples indicate how NLI differs from pattern approaches. Example (a) involves the common sense knowledge that occupying a territory implies remaining there. This might be learned from patterns more easily as these patterns might resemble contexts – seen during pretraining – that describe how long a military force remained during an occupation. Putting the inference candidate (b) into a pattern-generated context avoids being fooled by the high similarity of the two sentences. Only the handcrafted patterns can make sense of the important details in this construction.

In contrast, (c) and (d) are difficult for our pattern approaches whereas NLI gets them right. We hypothesize that the problem stems from linking the two sentences into one. An entailment pattern ideally represents a derivation of the hypothesis from the premise. One may wrongly conclude that (c) *Athena was the goddess of Athens only because she was worshiped there*, by neglecting the possibility that there are others that are equally worshiped. In the same way, (d) is unlikely to be found in an argumentative text. While it is clear that there can be no beating without a fight, one would hardly argue that *Pyrrhus fought the romans because they beat him*. This particular reasoning calls for additional explanations like *Pyrrhus must have fought the romans because I know that they beat him*. This analysis serves as inspiration for further improvements of entailment patterns.

The last two examples (e) and (f) are difficult for all approaches. It seems to be a particular challenge to identify open situations like a sports match or a negotiation where multiple outcomes are possible and distinguish them from cases where one particular outcome is inevitable.

7 Conclusion

We proposed and evaluated three approaches to the task of lexical inference in context (LIiC) based on pretrained language models (LMs). In particular, we found that putting an inference candidate into a pattern-generated context mostly increases performance compared to a standard sequence classification approach. Concrete performance, however, also depends on the particular dataset, used LM (variant), and pattern set. We introduced the concept of antipatterns, which express the negative class of a binary classification, and found that they often lead to performance gains for LIiC. We set a new state of the art for LIiC and conducted an extensive analysis of our approaches. Notably, we found that automatically created patterns can perform nearly as well as handcrafted ones if we either use the right number n of patterns or manually identify the right subset of them. Promising directions for future work are the investigation of alternative automatic pattern generation methods or a better modeling of the remaining challenges we described in our error analysis (§6.3).

Acknowledgments

We gratefully acknowledge a Ph.D. scholarship awarded to the first author by the German Academic Scholarship Foundation (Studienstiftung des deutschen Volkes). This work was supported by the BMBF as part of the project MLWin (01IS18050).

References

Asaf Amrami and Yoav Goldberg. 2018. Word sense induction with neural biLM and symmetric patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium. Association for Computational Linguistics.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 610–619, Portland, Oregon, USA. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Computing Research Repository*, arXiv:2005.14165.

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 33–40, Barcelona, Spain. Association for Computational Linguistics.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing textual entailment: Models and applications*. Morgan & Claypool Publishers.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository*, arXiv:1810.04805.

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

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2185–2194, Hong Kong, China. Association for Computational Linguistics.

Christiane Fellbaum and George A. Miller. 1990. Folk psychology or semantic entailment? comment on rips and conrad (1989). *Psychological Review*, 97(4):565–570.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society (CogSci 2019)*, pages 1753–1759. cognitivesciencesociety.org.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*.

Xavier R. Holt. 2018. Probabilistic models of relational implication. Master's thesis, Macquarie University.

Mohammad Javad Hosseini, Nathanael Chambers, Siva Reddy, Xavier R. Holt, Shay B. Cohen, Mark Johnson, and Mark Steedman. 2018. Learning typed entailment graphs with global soft constraints. *Transactions of the Association for Computational Linguistics*, 6:703–717.

Mohammad Javad Hosseini, Shay B. Cohen, Mark Johnson, and Mark Steedman. 2019. Duality of link prediction and entailment graph induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4736–4746, Florence, Italy. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Douwe Kiela, Laura Rimell, Ivan Vulić, and Stephen Clark. 2015. Exploiting image generality for lexical entailment detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 119–124, Beijing, China. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389.

Omer Levy and Ido Dagan. 2016. Annotating relation inference in context via question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 249–255, Berlin, Germany. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland. Association for Computational Linguistics.

Mike Lewis and Mark Steedman. 2013. Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 768–774, Montreal, Quebec, Canada. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2001. DIRT: Discovery of Inference Rules from Text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*, pages 323–328, New York, NY, USA. ACM Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. 2020. Paraphrasing vs coreferring: Two sides of the same coin. *Computing Research Repository*, arXiv:2004.14979.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computing Research Repository*, arXiv:1301.3781.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Shachar Mirkin, Ido Dagan, and Maayan Geffet. 2006. Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 579–586, Sydney, Australia. Association for Computational Linguistics.

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

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2163–2172, Austin, Texas. Association for Computational Linguistics.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–363, Melbourne, Australia. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *Computing Research Repository*, arXiv:2001.07676.

Martin Schmitt and Hinrich Schütze. 2019. SherLIiC: A typed event-focused lexical inference benchmark for evaluating natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 902–914, Florence, Italy. Association for Computational Linguistics.

Stefan Schoenmackers, Jesse Davis, Oren Etzioni, and Daniel Weld. 2010. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098, Cambridge, MA. Association for Computational Linguistics.

Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 258–267, Beijing, China. Association for Computational Linguistics.

Vered Shwartz, Omer Levy, Ido Dagan, and Jacob Goldberger. 2015. Learning to exploit structured resources for lexical inference. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 175–184, Beijing, China. Association for Computational Linguistics.

Vered Shwartz, Gabriel Stanovsky, and Ido Dagan. 2017. Acquiring predicate paraphrases from news tweets. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 155–160, Vancouver, Canada. Association for Computational Linguistics.

Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 849–856, Manchester, UK. Coling 2008 Organizing Committee.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ivan Vulić and Nikola Mrkšić. 2018. Specialising word vectors for lexical entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1134–1145, New Orleans, Louisiana. Association for Computational Linguistics.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 81–88.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

A Hyperparameters

We train all our classifiers for 5 epochs with the Adam optimizer (Kingma and Ba, 2015) and a mini-batch size of 10 or 2 instances for RoBERTa-base and -large, respectively. For AUTPAT$_n$ approaches with $n > 5$, we distribute the available patterns and antipatterns into chunks of size 5 for training to save memory. During evaluation, the predictions are based on all the patterns and antipatterns.

We randomly sample 500 configurations for the remaining hyperparameters, i.e., initial learning rate lr, weight decay λ (L2 regularization), and the number of batches c the gradient is accumulated before each optimization step, which virtually increases the batch size by a factor of c. The hyperparameters are sampled from the following intervals: $\mathrm{lr} \in [10^{-8}, 5 \cdot 10^{-2}]$, $\lambda \in [10^{-5}, 10^{-1}]$, $c \in \{1, 2, \dots, 10\}$. lr and λ are sampled uniformly in log-space. For a fair comparison, we use the same 500 random configurations for all of our approaches.

As usual for Transformer models, we apply a learning rate schedule: lr decreases linearly such that it reaches 0 at the end of the last epoch. We do not employ warm-up.

The best configurations can be seen in Tables 8 and 10 for Levy/Holt's dataset and in Tables 9 and 11 for SherLIiC.

B Results on development sets

See Tables 12 and 13.

C Varying n in training and evaluation

Another approach to make use of different values of n in AUTPAT$_n$ systems is to vary n from training to evaluation. Figure 2 is a visualization of the performance impact of this procedure. The base point for the visualization (in white) is the AUC performance of AUTPAT$^{\Phi}_5$. We see that training with $n = 50$ almost always leads to a performance drop (marked in blue) w.r.t. this number. It seems generally to be catastrophic to evaluate a model with patterns that were not seen during training, indicating that there is no generalization from seen patterns to unseen patterns even if they were chosen by the same method and can thus be expected to be – at least to some extent – similar. In general, this evaluation suggests that modifying n after the training always leads to a drop in performance.

D Transfer between Datasets

Table 7 shows results on the question how well a model trained on one dataset performs on the other. For this, we assume that the target dataset is not available at all, i.e., we do not use it at all – neither for finding patterns in AUTPAT nor for tuning the threshold ϑ. We thus use the standard ϑ values, i.e., 0.5 for NLI and 0.0 for the pattern-based methods.

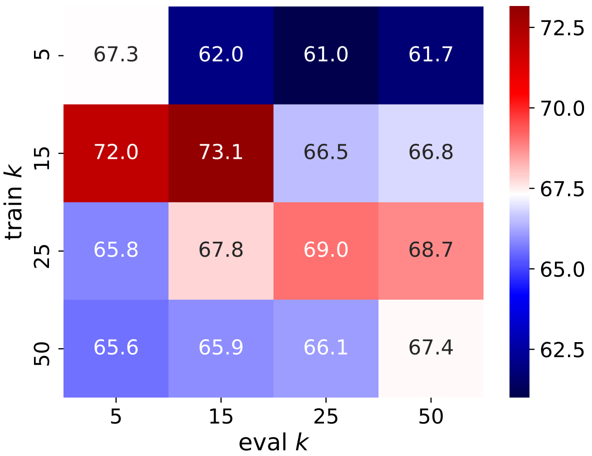


Figure 2: RoBERTa-base+AUTPAT$^{\Phi}_k$ performance on SherLIiC test for different k values during training and evaluation. Same hyperparameters used for all models (as in Table 5). Blue marks drops, red marks gains in performance w.r.t. AUTPAT$^{\Phi}_5$.

	AUC	P	R	F$_1$
RoBERTa-base				
NLI	38.4	52.7	57.1	**54.8**
MANPAT$^{\Phi\Psi}$	**46.1**	**64.0**	45.4	53.1
MANPAT$^{\Phi}$	32.4	32.4	**94.5**	48.2
AUTPAT$^{\Phi\Psi}_5$	18.7	40.5	35.0	37.6
AUTPAT$^{\Phi}_5$	21.3	28.3	62.3	38.9
RoBERTa-large				
NLI	37.8	31.0	96.4	46.9
MANPAT$^{\Phi\Psi}$	**70.4**	39.6	95.3	**56.0**
MANPAT$^{\Phi}$	38.9	25.6	**98.3**	40.6
AUTPAT$^{\Phi\Psi}_5$	33.6	**61.6**	36.0	45.5
AUTPAT$^{\Phi}_5$	9.3	30.7	76.6	43.8

(a) SherLIiC train → Levy/Holt test.

	AUC	P	R	F$_1$
RoBERTa-base				
NLI	63.3	62.8	**68.4**	**65.5**
MANPAT$^{\Phi\Psi}$	**69.1**	**80.5**	42.1	55.3
MANPAT$^{\Phi}$	68.4	80.1	24.2	37.2
AUTPAT$^{\Phi\Psi}_5$	60.3	71.5	54.5	61.9
AUTPAT$^{\Phi}_5$	58.9	68.6	55.7	61.5
RoBERTa-large				
NLI	65.6	73.8	53.0	61.7
MANPAT$^{\Phi\Psi}$	69.6	84.7	35.7	50.3
MANPAT$^{\Phi}$	**72.2**	**89.3**	30.3	45.2
AUTPAT$^{\Phi\Psi}_5$	62.1	68.1	**57.3**	**62.3**
AUTPAT$^{\Phi}_5$	63.8	75.8	44.2	55.8

(b) Levy/Holt train → SherLIiC test.

Table 7: Transfer learning. Table format: see Table 2.

	NLI	MANPAT$^{\Phi\Psi}$	MANPAT$^{\Phi}$	AUTPAT$_5^{\Phi\Psi}$	AUTPAT$_5^{\Phi}$
lr	$2.72 \cdot 10^{-5}$	$2.47 \cdot 10^{-5}$	$6.68 \cdot 10^{-6}$	$3.82 \cdot 10^{-5}$	$2.11 \cdot 10^{-5}$
λ	$1.43 \cdot 10^{-3}$	$2.98 \cdot 10^{-4}$	$1.07 \cdot 10^{-5}$	$4.02 \cdot 10^{-5}$	$1.65 \cdot 10^{-5}$
c	1	2	1	2	3

Table 8: Levy/Holt; RoBERTa-base.

	NLI	MANPAT$^{\Phi\Psi}$	MANPAT$^{\Phi}$	AUTPAT$_5^{\Phi\Psi}$	AUTPAT$_5^{\Phi}$	AUTCUR$_5^{\Phi}$	AUTARG$_5^{\Phi}$
lr	$6.34 \cdot 10^{-6}$	$3.87 \cdot 10^{-5}$	$2.28 \cdot 10^{-5}$	$3.92 \cdot 10^{-5}$	$2.53 \cdot 10^{-5}$	$1.28 \cdot 10^{-5}$	$2.47 \cdot 10^{-5}$
λ	$1.35 \cdot 10^{-3}$	$1.43 \cdot 10^{-5}$	$6.52 \cdot 10^{-2}$	$2.18 \cdot 10^{-4}$	$1.02 \cdot 10^{-5}$	$8.23 \cdot 10^{-3}$	$2.98 \cdot 10^{-4}$
c	1	4	2	1	1	1	2

Table 9: SherLIiC; RoBERTa-base.

	NLI	MANPAT$^{\Phi\Psi}$	MANPAT$^{\Phi}$	AUTPAT$_5^{\Phi\Psi}$	AUTPAT$_5^{\Phi}$
lr	$6.68 \cdot 10^{-6}$	$4.55 \cdot 10^{-6}$	$4.92 \cdot 10^{-6}$	$6.68 \cdot 10^{-6}$	$8.13 \cdot 10^{-6}$
λ	$1.07 \cdot 10^{-5}$	$3.90 \cdot 10^{-4}$	$3.61 \cdot 10^{-4}$	$1.07 \cdot 10^{-5}$	$6.05 \cdot 10^{-2}$
c	1	2	3	1	2

Table 10: Levy/Holt; RoBERTa-large.

	NLI	MANPAT$^{\Phi\Psi}$	MANPAT$^{\Phi}$	AUTPAT$_5^{\Phi\Psi}$	AUTPAT$_5^{\Phi}$
lr	$6.68 \cdot 10^{-6}$	$1.29 \cdot 10^{-5}$	$9.14 \cdot 10^{-6}$	$6.34 \cdot 10^{-6}$	$4.55 \cdot 10^{-6}$
λ	$1.07 \cdot 10^{-5}$	$2.49 \cdot 10^{-4}$	$6.09 \cdot 10^{-5}$	$1.35 \cdot 10^{-3}$	$3.90 \cdot 10^{-4}$
c	1	3	4	1	2

Table 11: SherLIiC; RoBERTa-large.

		dev$_1$				dev$_2$				test			
		AUC	P	R	F1	AUC	P	R	F1	AUC	P	R	F1
baselines													
Hosseini et al. (2018)		–	–	–	–	–	–	–	–	16.5	–	–	–
Hosseini et al. (2019)		–	–	–	–	–	–	–	–	**18.7**	–	–	–
RoBERTa-base													
NLI	($\vartheta = 0.0052$)	94.9	87.4	91.1	89.2	88.8	78.1	**90.3**	83.8	72.6	68.7	**75.3**	71.9
MANPAT$^{\Phi\Psi}$	($\vartheta = -0.0909$)	**96.5**	**87.7**	**96.2**	**91.8**	**89.4**	**81.4**	88.5	**84.8**	**76.9**	**78.7**	66.4	**72.0**
MANPAT$^{\Phi}$	($\vartheta = 0.5793$)	91.8	80.2	90.1	84.9	84.7	77.5	81.1	79.3	71.2	74.4	61.2	67.1
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.1428$)	95.0	83.4	95.6	89.1	87.7	79.2	85.7	82.3	63.7	71.0	58.8	64.3
AUTPAT$_5^{\Phi}$	($\vartheta = -0.0592$)	87.4	78.0	90.0	83.6	83.3	76.3	81.6	78.8	65.4	68.0	63.3	65.5
RoBERTa-large													
NLI	($\vartheta = 0.0016$)	96.9	90.1	97.1	**93.5**	87.7	82.6	87.6	**85.0**	75.5	73.5	73.7	73.6
MANPAT$^{\Phi\Psi}$	($\vartheta = 0.1156$)	**97.1**	**91.4**	95.4	93.4	**88.9**	**84.0**	84.8	84.4	**83.9**	**84.8**	70.1	**76.7**
MANPAT$^{\Phi}$	($\vartheta = -0.8457$)	92.2	76.1	**97.3**	85.4	84.4	72.5	**91.2**	80.8	77.8	67.9	**81.5**	74.1
AUTPAT$_5^{\Phi\Psi}$	($\vartheta = -0.0021$)	95.0	86.0	91.9	88.8	84.7	78.9	81.1	80.0	70.4	75.7	60.7	67.4
AUTPAT$_5^{\Phi}$	($\vartheta = -0.9197$)	92.4	75.5	95.3	84.2	83.5	70.6	88.5	78.5	66.5	61.8	74.4	67.5

Table 12: Full results on the Levy/Holt dataset. The dev and test sets as created by Hosseini et al. (2018) are called dev$_1$ and test. The portion of dev$_1$ that serves as our validation set is called dev$_2$. AUC denotes the area under the precision-recall curve for precision values ≥ 0.5. All results in %.

	dev$_1$				dev$_2$				test			
	AUC	P	R	F$_1$	AUC	P	R	F$_1$	AUC	P	R	F$_1$
baselines												
Lemma	–	**90.0**	10.9	19.4	–	–	–	–	–	**90.7**	8.9	16.1
w2v+untyped_rel	–	56.5	74.0	64.1	–	–	–	–	–	52.8	69.5	60.0
w2v+tsg_rel_emb	–	56.6	**77.6**	**65.5**	–	–	–	–	–	51.8	**72.7**	**60.5**
RoBERTa-base												
NLI ($\vartheta = 0.3878$)	81.3	**79.1**	80.1	79.6	81.5	**84.2**	70.6	**76.8**	65.8	**67.0**	66.1	66.5
MANPAT$^{\Phi\Psi}$ ($\vartheta = -0.3324$)	76.2	68.6	85.8	76.2	**82.4**	70.0	**82.4**	75.7	66.4	60.9	78.8	68.7
MANPAT$^{\Phi}$ ($\vartheta = -0.4812$)	**88.4**	75.5	**93.1**	**83.4**	84.1	73.0	79.4	76.1	**69.2**	62.0	**81.2**	**70.3**
AUTPAT$_5^{\Phi\Psi}$ ($\vartheta = -0.4694$)	87.0	77.8	88.8	82.9	71.2	68.4	76.5	72.2	67.4	61.8	75.6	68.0
AUTPAT$_5^{\Phi}$ ($\vartheta = -0.7042$)	86.8	64.1	91.8	75.5	74.0	65.5	83.8	73.6	67.3	56.6	82.6	67.2
AUTCUR$_5^{\Phi}$ ($\vartheta = -0.7524$)	82.6	61.7	92.8	74.1	75.6	60.6	88.2	71.9	69.5	56.3	89.6	69.2
AUTARG$_5^{\Phi}$ ($\vartheta = -0.7461$)	77.4	69.3	84.0	76.0	73.6	68.9	75.0	71.8	65.2	61.9	75.6	68.1
AUTPAT$_{15}^{\Phi\Psi}$ ($\vartheta = -0.5263$)	95.3	87.0	93.1	89.9	73.0	65.4	75.0	69.9	70.0	60.4	79.7	68.7
AUTPAT$_{15}^{\Phi}$ ($\vartheta = -0.6422$)	**95.4**	85.3	94.6	89.7	**75.8**	69.2	**79.4**	**74.0**	**73.1**	**63.0**	77.4	**69.4**
AUTPAT$_{25}^{\Phi\Psi}$ ($\vartheta = -0.0014$)	95.0	**92.0**	93.7	**92.8**	66.1	**70.0**	72.1	71.0	63.5	62.1	73.4	67.3
AUTPAT$_{25}^{\Phi}$ ($\vartheta = -0.6496$)	88.1	72.3	90.0	80.2	73.0	67.5	79.4	73.0	69.0	60.5	**79.4**	68.7
AUTPAT$_{50}^{\Phi\Psi}$ ($\vartheta = -0.9163$)	93.2	72.8	92.8	81.5	67.1	63.0	75.0	68.5	66.3	54.3	**82.8**	65.6
AUTPAT$_{50}^{\Phi}$ ($\vartheta = -0.9500$)	94.2	79.1	**94.9**	86.3	69.3	66.3	77.9	71.6	67.4	57.3	82.5	67.6
RoBERTa-large												
NLI ($\vartheta = 0.0025$)	**92.3**	**79.7**	**93.7**	**86.1**	75.7	66.7	**82.4**	73.7	68.3	60.5	**85.5**	70.9
MANPAT$^{\Phi\Psi}$ ($\vartheta = -0.0956$)	89.3	77.3	88.5	82.5	**80.8**	**74.7**	77.9	**76.3**	**74.4**	**66.0**	80.8	**72.6**
MANPAT$^{\Phi}$ ($\vartheta = -0.6641$)	78.0	67.4	84.9	75.1	72.2	67.1	77.9	72.1	64.6	58.1	79.0	67.0
AUTPAT$_5^{\Phi\Psi}$ ($\vartheta = -0.9889$)	86.5	73.8	86.7	79.7	73.6	70.4	73.5	71.9	68.6	61.9	75.5	68.0
AUTPAT$_5^{\Phi}$ ($\vartheta = -0.5355$)	71.6	64.3	71.9	67.9	64.5	71.4	66.2	68.7	56.8	61.5	66.1	63.7

Table 13: Full results on SherLIiC. The original dev and test sets are called dev$_1$ and test. The portion of dev$_1$ that serves as our validation set is called dev$_2$. AUC denotes the area under the precision-recall curve for precision values ≥ 0.5. Baseline results from (Schmitt and Schütze, 2019). All results in %.

(1)	The original aim of de Garis' work was to **prem** the field of "brain building" (a term of his invention) and to "**hypo** a trillion dollar industry within 20 years".	→	The original aim of their work was that "PARGL **prem** PARGR" and that "HARGL **hypo** HARGR within 20 years".
(2)	Critic Roger Ebert stated that Gellar and co-star Ryan Phillippe "**prem** a convincing emotional charge" and that Gellar is "effective as a bright girl who knows exactly how to **hypo** her act as a tramp".	→	Critic Roger Ebert stated that PARGL and co-star Ryan Phillippe "**prem** PARGR" and that HARGL is "effective as a bright girl who knows exactly how she **hypo** HARGR as a tramp".
(3)	Well-known professional competitions in the past have included the World Professional Championships (**hypo** Landover, Maryland), the Challenge Of Champions, the Canadian Professional Championships and the World Professional Championships (**prem** in Jaca, Spain).	→	Well-known professional competitions in the past have included HARGL (**hypo** HARGR), the Challenge Of Champions, the Canadian Professional Championships and PARGL (**prem** PARGR).
(4)	They also had sharpshooter Steve Kerr, whom they **hypo** via free agency before the 1993–94 season, Myers, and centers Luc Longley (**prem** via trade in 1994 from the Minnesota Timberwolves) and Bill Wennington.	→	HARGL also had sharpshooter HARGR, whom they **hypo** via free agency before the 1993–94 season, Myers, and centers PARGR (whom PARGL **prem** via trade in 1994 from the Minnesota Timberwolves) and Bill Wennington.
(5)	Because the 6x86 was more efficient on an instructions-per-cycle basis than Intel's Pentium, and because Cyrix sometimes **hypo** a faster bus speed than either Intel or AMD, Cyrix and competitor AMD co-**prem** the controversial PR system in an effort to compare its products more favorably with Intel's. ...	→	Because the 6x86 was more efficient on an instructions-per-cycle basis than Intel's Pentium, and because HARGL sometimes **hypo** HARGR, PARGL and competitor AMD co-**prem** PARGR in an effort to compare its products more favorably with Intel's. ...

Table 14: Five manually selected patterns from the 100 highest-ranked automatically extracted patterns from SherLIiC dev$_1$ (used in AUTCUR$_5^{\Phi}$) and their rewritten counterparts (used in AUTARG$_5^{\Phi}$). PARGL (HARGL) stands for the left argument of the premise (hypothesis); PARGR (HARGR) for the right one.

Few-Shot Semantic Parsing for New Predicates

Zhuang Li, Lizhen Qu*, Shuo Huang, Gholamreza Haffari
Faculty of Information Technology
Monash University
firstname.lastname@monash.edu
shua0043@student.monash.edu

Abstract

In this work, we investigate the problems of semantic parsing in a few-shot learning setting. In this setting, we are provided with k utterance-logical form pairs per new predicate. The state-of-the-art neural semantic parsers achieve less than 25% accuracy on benchmark datasets when $k = 1$. To tackle this problem, we proposed to i) apply a designated meta-learning method to train the model; ii) regularize attention scores with alignment statistics; iii) apply a smoothing technique in pre-training. As a result, our method consistently outperforms all the baselines in both one and two-shot settings.

1 Introduction

Semantic parsing is the task of mapping natural language (NL) utterances to structured meaning representations, such as logical forms (LF). One key obstacle preventing the wide application of semantic parsing is the lack of task-specific training data. New tasks often require new predicates of LFs. Suppose a personal assistant (e.g. Alexa) is capable of booking flights. Due to new business requirement it needs to book ground transport as well. A user could ask the assistant "*How much does it cost to go from Atlanta downtown to airport?*". The corresponding LF is as follows:

(lambda $0 e (exists $1 (and (ground_transport $1)
(to_city $1 atlanta:ci)(from_airport $1 atlanta:ci)
(=(ground_fare $1) $0))))

where both *ground_transport* and *ground_fare* are new predicates while the other predicates are used in flight booking, such as *to_city*, *from_airport*. As manual construction of large parallel training data is expensive and time-consuming, we consider the *few-shot* formulation of the problem, which requires only a handful of utterance-LF training pairs for each new predicate. The cost of preparing few-shot training examples is low, thus the corresponding techniques permit significantly faster prototyping and development than supervised approaches for business expansions.

Semantic parsing in the few-shot setting is challenging. In our experiments, the accuracy of the state-of-the-art (SOTA) semantic parsers drops to less than *25%*, when there is only one example per new predicate in training data. Moreover, the SOTA parsers achieve less than 32% of accuracy on five widely used corpora, when the LFs in the test sets do not share LF *templates* in the training sets (Finegan-Dollak et al., 2018). An LF template is derived by *normalizing* the entities and attribute values of an LF into typed variable names (Finegan-Dollak et al., 2018). The few-shot setting imposes two major challenges for SOTA neural semantic parsers. First, it lacks sufficient data to learn effective representations for new predicates in a supervised manner. Second, new predicates bring in new LF templates, which are mixtures of known and new predicates. In contrast, the tasks (e.g. image classification) studied by the prior work on few-shot learning (Snell et al., 2017; Finn et al., 2017) considers an instance exclusively belonging to either a known class or a new class. Thus, it is non-trivial to apply conventional few-shot learning algorithms to generate LFs with mixed types of predicates.

To address above challenges, we present `ProtoParser`, a transition-based neural semantic parser, which applies a sequence of parse actions to transduce an utterance into an LF template and fills the corresponding slots. The parser is pre-trained on a training set with known predicates, followed by fine-tuning on a *support set* that contains few-shot examples of new predicates. It extends the attention-based sequence-to-sequence architecture (Sutskever et al., 2014) with the following

---

*corresponding author

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1281–1291
April 19 - 23, 2021.

novel techniques to alleviate the specific problems in the few-shot setting:

- *Predicate-droput*. Predicate-droput is a meta-learning technique to improve representation learning for both known and new predicates. We empirically found that known predicates are better represented with supervisely learned embeddings, while new predicates are better initialized by a metric-based few-shot learning algorithm (Snell et al., 2017). In order to let the two types of embeddings work together in a single model, we devised a training procedure called *predicate-dropout* to simulate the testing scenario in pre-training.

- *Attention regularization*. In this work, new predicates appear approximately once or twice during training. Thus, it is insufficient to learn reliable attention scores in the Seq2Seq architecture for those predicates. In the spirit of supervised attention (Liu et al., 2016), we propose to regularize them with alignment scores estimated by using co-occurrence statistics and string similarity between words and predicates. The prior work on supervised attention is not applicable, because it requires either large parallel data (Liu et al., 2016), significant manual effort (Bao et al., 2018; Rabinovich et al., 2017), or it is designed only for applications other than semantic parsing (Liu et al., 2017; Kamigaito et al., 2017).

- *Pre-training smoothing*. The vocabulary of predicates in fine-tuning is higher than that in pre-training, which leads to a distribution discrepancy between the two training stages. Inspired by Laplace smoothing (Manning et al., 2008), we achieve significant performance gain by applying a smoothing technique during pre-training to alleviate the discrepancy.

Our extensive experiments on three benchmark corpora show that `ProtoParser` outperforms the competitive baselines with a significant margin. The ablation study demonstrates the effectiveness of each individual proposed technique. The results are statistically significant with $p \leq 0.05$ according to the Wilcoxon signed-rank test (Wilcoxon, 1992).

2 Related Work

Semantic parsing There is ample of work on machine learning models for semantic parsing. The recent surveys (Kamath and Das, 2018; Zhu et al., 2019) cover a wide range of work in this area. The semantic formalism of meaning representations range from lambda calculas (Montague, 1973), SQL, to abstract meaning representation (Banarescu et al., 2013). At the core of most recent models (Chen et al., 2018; Cheng et al., 2019; Lin et al., 2019; Zhang et al., 2019b; Yin and Neubig, 2018) is SEQ2SEQ with attention (Bahdanau et al., 2014) by formulating the task as a machine translation problem. COARSE2FINE (Dong and Lapata, 2018) reports the highest accuracy on GEOQUERY (Zelle and Mooney, 1996) and ATIS (Price, 1990) in a supervised setting. IRNET (Guo et al., 2019) and RATSQL (Wang et al., 2019) are two best performing models on the Text-to-SQL benchmark, SPIDER (Yu et al., 2018). They are also designed to be able to generalize to unseen database schemas. However, supervised models perform well only when there is sufficient training data.

Data Sparsity Most semantic parsing datasets are small in size. To address this issue, one line of research is to augment existing datasets with automatically generated data (Su and Yan, 2017; Jia and Liang, 2016; Cai and Yates, 2013). Another line of research is to exploit available resources, such as knowledge bases (Krishnamurthy et al., 2017; Herzig and Berant, 2018; Chang et al., 2019; Lee, 2019; Zhang et al., 2019a; Guo et al., 2019; Wang et al., 2019), semantic features in different domains (Dadashkarimi et al., 2018; Li et al., 2020), or unlabeled data (Yin et al., 2018; Kočiskỳ et al., 2016; Sun et al., 2019). Those works are orthogonal to our setting because our approach aims to efficiently exploit a handful of labeled data of new predicates, which are not limited to the ones in knowledge bases. Our setting also does not require involvement of humans in the loop such as active learning (Duong et al., 2018; Ni et al., 2019) and crowd-sourcing (Wang et al., 2015; Herzig and Berant, 2019). We assume availability of resources different than the prior work and focus on the problems caused by new predicates. We develop an approach to generalize to unseen LF templates consisting of both known and new predicates.

Few-Shot Learning Few-shot learning is a type of machine learning problems that provides a handful of labeled training examples for a specific task. The survey (Zhu et al., 2019) gives a comprehensive overview of the data, models, and algorithms

t	Actions
t_1	*GEN* [(ground_transport v_a)]
t_2	*GEN* [(to_city v_a v_e)]
t_3	*GEN* [(from_airport v_a v_e)]
t_4	*GEN* [(= (ground_fare v_a) v_a)]
t_5	REDUCE [and :- NT NT NT NT]
t_6	REDUCE [exists :- v_a NT]
t_7	REDUCE [lambda :- v_a e NT]

Table 1: An example action sequence.

proposed for this type of problems. It categorizes the models into multitask learning (Hu et al., 2018), embedding learning (Snell et al., 2017; Vinyals et al., 2016), learning with external memory (Lee and Choi, 2018; Sukhbaatar et al., 2015), and generative modeling (Reed et al., 2017) in terms of what prior knowledge is used. (Lee et al., 2019) tackles the problem of poor generalization across SQL templates for SQL query generation in the one-shot learning setting. In their setting, they assume all the SQL templates on test set are shared with the templates on support set. In contrast, we assume only the sharing of new predicates between a support set and a test set. In our one-shot setting, only around 10% of LF templates on test set are shared with the ones in the support set of GEOQUERY dataset.

3 Semantic Parser

`ProtoParser` follows the SOTA neural semantic parsers (Dong and Lapata, 2018; Guo et al., 2019) to map an utterance into an LF in two steps: *template generation* and *slot filling*[1]. It implements a designated transition system to generate templates, followed by filling the slot variables with values extracted from utterances. To address the challenges in the few-shot setting, we proposed three training methods, detailed in Sec. 4.

Many LFs differ only in mentioned atoms, such as entities and attribute values. An LF template is created by replacing the atoms in LFs with typed slot variables. As an example, the LF template of our example in Sec. 1 is created by substituting i) a typed atom variable v_e for the entity "atlanta:ci"; ii) a shared variable name v_a for all variables "\$0" and "\$1".

(lambda v_a e (exists v_a (and (ground_transport v_a)
(to_city v_a v_e)(from_airport v_a v_e) (=(ground_fare v_a) v_a))))

Formally, let $\boldsymbol{x} = \{x_1, ..., x_n\}$ denote an NL utterance, and its LF is represented as a semantic tree $\boldsymbol{y} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, ..., v_m\}$ denotes the node set with $v_i \in \mathcal{V}$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is its edge set. The node set $\mathcal{V} = \mathcal{V}_p \cup \mathcal{V}_v$ is further divided into a template predicate set $\mathcal{V}_p$ and a slot value set $\mathcal{V}_v$. A template predicate node represents a predicate symbol or a term, while a slot value node represents an atom mentioned in utterances. Thus, a semantic tree $\boldsymbol{y}$ is composed of an abstract tree $\tau_{\boldsymbol{y}}$ representing a template and a set of slot value nodes $\mathcal{V}_{v,\boldsymbol{y}}$ attaching to the abstract tree.

In the few-shot setting, we are provided with a train set $\mathcal{D}_{train}$, a support set $\mathcal{D}_s$, and a test set $\mathcal{D}_{test}$. Each example in either of those sets is an utterance-LF pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$. The new predicates appear only in $\mathcal{D}_s$ and $\mathcal{D}_{test}$ but *not* in $\mathcal{D}_{train}$. For K-shot learning, there are K $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ per each new predicate p in $\mathcal{D}_s$. Each new predicate appears also in the test set. The goal is to maximize the accuracy of estimating LFs given utterances in $\mathcal{D}_{test}$ by using a parser trained on $\mathcal{D}_{train} \cup \mathcal{D}_s$.

3.1 Transition System

We apply the transition system (Cheng et al., 2019) to perform a sequence of transition actions to generate the template of a semantic tree. The transition system maintains partially-constructed outputs using a *stack*. The parser starts with an empty stack. At each step, it performs one of the following transition actions to update the parsing state and generate a tree node. The process repeats until the stack contains a complete tree.

- ***GEN*** $[y]$ creates a new leaf node y and pushes it on top of the stack.
- **REDUCE** $[r]$. The reduce action identifies an implication rule *head* : −*body*. The rule body is first popped from the stack. A new subtree is formed by attaching the rule head as a new parent node to the rule body . Then the whole subtree is pushed back to the stack.

Table 1 shows such an action sequence for generating the above LF template. Each action produces *known* or *new* predicates.

3.2 Base Parser

`ProtoParser` generates an LF in two steps: i) template generation, ii) slot filling. The base architecture largely resembles (Cheng et al., 2019).

---

[1] Code and datasets can be found in this repository: `https://github.com/zhuang-li/few-shot-semantic-parsing`

Template Generation Given an utterance, the task is to generate a sequence of actions $\mathbf{a} = a_1, ..., a_k$ to build an abstract tree $\tau_{\boldsymbol{y}}$.

We found out LFs often contain idioms, which are frequent subtrees shared across LF templates. Thus we apply a *template normalization* procedure in a similar manner as (Iyer et al., 2019) to preprocess all LF templates. It collapses idioms into single units such that all LF templates are converted into a compact form.

The neural transition system consists of an encoder and a decoder for estimating action probabilities.

$$P(\mathbf{a}|\boldsymbol{x}) = \prod_{t=1}^{|\mathbf{a}|} P(a_t|\boldsymbol{a}_{<t}, \boldsymbol{x}) \tag{1}$$

Encoder We apply a bidirectional Long Short-term Memory (LSTM) network (Gers et al., 1999) to map a sequence of n words into a sequence of contextual word representations $\{\mathbf{e}\}_{i=1}^{n}$.

Template Decoder The decoder applies a stack-LSTM (Dyer et al., 2015) to generate action sequences. A stack-LSTM is an unidirectional LSTM augmented with a pointer. The pointer points to a particular hidden state of the LSTM, which represents a particular state of the stack. It moves to a different hidden state to indicate a different state of the stack.

At time t, the stack-LSTM produces a hidden state $\mathbf{h}_t^d$ by $\mathbf{h}_t^d = \text{LSTM}(\mu_t, \mathbf{h}_{t-1}^d)$, where μ_t is a concatenation of the embedding of the action $\mathbf{c}_{a_{t-1}}$ estimated at time $t-1$ and the representation $\mathbf{h}_{y_{t-1}}$ of the partial tree generated by history actions at time $t-1$.

As a common practice, $\mathbf{h}_t^d$ is concatenated with an attended representation $\mathbf{h}_t^a$ over encoder hidden states to yield $\mathbf{h}_t$, with $\mathbf{h}_t = \mathbf{W}\begin{bmatrix}\mathbf{h}_t^d \\ \mathbf{h}_t^a\end{bmatrix}$, where $\mathbf{W}$ is a weight matrix and $\mathbf{h}_t^a$ is created by soft attention,

$$\mathbf{h}_t^a = \sum_{i=1}^{n} P(\mathbf{e}_i|\mathbf{h}_t^d)\mathbf{e}_i \tag{2}$$

We apply dot product to compute the normalized attention scores $P(\mathbf{e}_i|\mathbf{h}_t^d)$ (Luong et al., 2015). The *supervised attention* (Rabinovich et al., 2017; Yin and Neubig, 2018) is also applied to facilitate the learning of attention weights. Given $\mathbf{h}_t$, the probability of an action is estimated by:

$$P(a_t|\mathbf{h}_t) = \frac{\exp(\mathbf{c}_{a_t}^{\intercal}\mathbf{h}_t)}{\sum_{a' \in \mathcal{A}_t} \exp(\mathbf{c}_{a'}^{\intercal}\mathbf{h}_t)} \tag{3}$$

where $\mathbf{c}_a$ denotes the embedding of action a, and $\mathcal{A}_t$ denotes the set of applicable actions at time t. The initialization of those embeddings will be explained in the following section.

Slot Filling A tree node in a semantic tree may contain more than one slot variables due to template normalization. Since there are two types of slot variables, given a tree node with slot variables, we employ a LSTM-based decoder with the same architecture as the *Template* decoder to fill each type of slot variables, respectively. The output of such a decoder is a value sequence of the same length as the number of slot variables of that type in the given tree node.

4 Few-Shot Model Training

The few-shot setting differs from the supervised setting by having a support set in testing in addition to train/test sets. The support set contains k utterance-LF pairs per new predicate, while the training set contains only known predicates. To evaluate model performance on new predicates, the test set contains LFs with both known and new predicates. Given the support set, we can tell if a predicate is known or new by checking if it only exists in the train set.

We take two steps to train our model: i) pre-training on the training set, ii) fine-tuning on the support set. Its predictive performance is measured on the test set. We take the two-steps approach because i) our experiments show that this approach performs better than training on the union of the train set and the support set; ii) for any new support sets, it is computationally more time efficient than training from scratch on the union of the train set and the support set.

There is a distribution discrepancy between the train set and the support set due to new predicates, the meta-learning algorithms (Snell et al., 2017; Finn et al., 2017) suggest to simulate the testing scenario in pre-training by splitting each batch into a meta-support set and a meta-test set. The models utilize the information (e.g. prototype vectors) acquired from the meta-support set to minimize errors on the meta-test set. In this way, the meta-support and meta-test sets simulate the support and test sets sharing new predicates.

However, we cannot directly apply such a training procedure due to the following two reasons. First, each LF in the support and test sets is a mixture of both known predicates and new predicates. To simulate the support and test sets, the meta-support and meta-test sets should include both types of predicates as well. We cannot assume that there are only one type of predicates. Second, our preliminary experiments show that if there is sufficient training data, it is better off training action embeddings of known predicates $\mathbf{c}$ (Eq. (3)) in a supervised way, while action embeddings initialized by a metric-based meta-learning algorithm (Snell et al., 2017) perform better for rarely occurred new predicates. Therefore, we cope with the differences between known and new predicates by using a customized initialization method in fine-tuning and a designated pre-training procedure to mimic fine-tuning on the train set. In the following, we introduce fine-tuning first because it helps understand our pre-training procedure.

4.1 Fine-tuning

During fine-tuning, the model parameters and the action embeddings in Eq. (3) for known predicates are obtained from the pre-trained model. The embedding of actions that produce new predicates $\mathbf{c}_{a_t}$ are initialized using prototype vectors as in prototypical networks (Snell et al., 2017). The prototype representations act as a type of regularization, which shares the similar idea as the deep learning techniques using pre-trained models.

A prototype vector of an action a_t is constructed by using the hidden states of the *template* decoder collected at the time of predicting a_t on a support set. Following (Snell et al., 2017), a prototype vector is built by taking the mean of such a set of hidden states $\mathbf{h}_t$.

$$\mathbf{c}_{a_t} = \frac{1}{|M|} \sum_{\mathbf{h}_t \in M} \mathbf{h}_t \tag{4}$$

where M denotes the set of all hidden states at the time of applying the action a_t. After initialization, the whole model parameters and the action embeddings are further improved by fine-tuning the model on the support set with a supervised training objective $\mathcal{L}_f$.

$$\mathcal{L}_f = \mathcal{L}_s + \lambda\Omega \tag{5}$$

where $\mathcal{L}_s$ is the cross-entropy loss and Ω is an attention regularization term explained below. The degree of regularization is adjusted by $\lambda \in \mathbb{R}^+$.

Attention Regularization We address the poorly learned attention scores $P(\mathbf{e}_i|\mathbf{h}^d_t)$ of infrequent actions by introducing a novel attention regularization. We observe that the probability $P(a_j|x_i) = \frac{\text{count}(a_j,x_i)}{\text{count}(x_i)}$ and the character similarity between the predicates generated by action a_j and the token x_i are often strong indicators of their alignment. The indicators can be further strengthened by manually annotating the predicates with their corresponding natural language tokens. In our work, we adopt $1 - dist(a_j, x_i)$ as the character similarity, where $dist(a_j, x_j)$ is normalized Levenshtein distance (Levenshtein, 1966). Both measures are in the range $[0, 1]$, thus we apply $g(a_j, x_i) = \sigma(\cdot)P(a_j|x_i) + (1 - \sigma(\cdot)char\_sim(a_j, x_i)$ to compute alignment scores, where the sigmoid function $\sigma(\mathbf{w}_p^\intercal \mathbf{h}^d_t)$ combines two constant measures into a single score. The corresponding normalized attention scores is given by

$$P'(x_i|a_k) = \frac{g(a_k, x_i)}{\sum_{j=1}^{n} g(a_k, x_j)} \tag{6}$$

The attention scores $P(x_i|a_k)$ should be similar to $P'(x_i|a_k)$. Thus, we define the regularization term as $\Omega = \sum_{ij} |P(x_i|a_j) - P'(x_i|a_j)|$ during training.

4.2 Pre-training

The pre-training objective are two-folds: i) learn action embeddings for known predicates in a supervised way, ii) ensure our model can quickly adapt to the actions of new predicates, whose embeddings are initialized by prototype vectors before fine-tuning.

Predicate-dropout Starting with randomly initialized model parameters, we alternately use one batch for the meta-loss $\mathcal{L}_m$ and one batch for optimizing the supervised loss $\mathcal{L}_s$.

In a batch for $\mathcal{L}_m$, we split the data into a meta-support set and a meta-test set. In order to simulate existence of new predicates, we randomly select a subset of predicates as ”new”, thus their action embeddings $\mathbf{c}$ are replaced by prototype vectors constructed by applying Eq. (4) over the meta-support set. The actions of remaining predicates keep their embeddings learned from previous batches. The resulted action embedding matrix $\mathbf{C}$ is the combination of both.

$$\mathbf{C} = (1 - \mathbf{m}^\intercal)\mathbf{C}_s + \mathbf{m}^\intercal \mathbf{C}_m \tag{7}$$

where $\mathbf{C}_s$ is the embedding matrix learned in a supervised way, and $\mathbf{C}_m$ is constructed by using prototype vectors on the meta-support set. The mask vector $\mathbf{m}$ is generated by setting the indices of actions of the ”new” predicates to ones and the other

Algorithm 1: Predicate-Dropout

```
Input  : Training set $\mathcal{D}$, supervisely trained action
         embedding $\mathcal{C}_s$, number of meta-support
         examples $k$, number of meta-test examples
         $n$ per one support example,
         predicate-dropout ratio $r$
Output : The loss $\mathcal{L}_m$.
Extract a template set $\mathcal{T}$ from the training set $\mathcal{D}$
Sample a subset $\mathcal{T}_i$ of size $k$ from $\mathcal{T}$
$S := \emptyset$ # meta-support set
$Q := \emptyset$ # meta-test set
for $t$ in $\mathcal{T}_i$ do
    Sample a meta-support example $s'$ with template
      $t$ from $\mathcal{D}$ without replacement
    Sample a meta-test set $Q'$ of size $n$ with template
      $t$ from $\mathcal{D}$
    $S = S \cup s'$
    $Q = Q \cup Q'$
end
Build a prototype matrix $\mathcal{C}_m$ on $S$
Extract a predicate set $\mathcal{P}$ from $S$
Sample a subset $\mathcal{P}_s$ of size $r \times |\mathcal{P}|$ from $\mathcal{P}$ as new
  predicates
Build a mask $\mathbf{m}$ using $\mathcal{P}_s$
With $\mathcal{C}_s$, $\mathcal{C}_m$ and $\mathbf{m}$, apply Eq. (7) to compute $\mathbf{C}$
Compute $\mathcal{L}_m$, the cross-entropy on $Q$ with $\mathbf{C}$
```

to zeros. We refer to this operation as *predicate-dropout*. The training algorithm for the meta-loss is summarised in Algorithm 1.

In a batch for $\mathcal{L}_s$, we update the model parameters and all action embeddings with a cross-entropy loss $\mathcal{L}_s$, together with the attention regularization. Thus, the overall training objective becomes

$$\mathcal{L}_p = \mathcal{L}_m + \mathcal{L}_s + \lambda\Omega \tag{8}$$

Pre-training smoothing Due to the new predicates, the number of candidate actions during the prediction of fine-tuning and testing is larger than the one during pre-training. That leads to distribution discrepancy between pre-training and testing. To minimize the differences, we assume a prior knowledge on the number of actions for new predicates by adding a constant k to the denominator of Eq. (3) when estimating the action probability $P(a_t|\mathbf{h}_t)$ during pre-training.

$$P(a_t|\mathbf{h}_t) = \frac{\exp(\mathbf{c}_{a_t}^{\intercal}\mathbf{h}_t)}{\sum_{a' \in \mathcal{A}_t} \exp(\mathbf{c}_{a'}^{\intercal}\mathbf{h}_t) + k} \tag{9}$$

We do not consider this smoothing technique during fine-tuning and testing. Despite its simplicity, the experimental results show a significant performance gain on benchmark datasets.

5 Experiments

Datasets. We use three semantic parsing datasets: JOBS, GEOQUERY, and ATIS. JOBS contains 640 question-LF pairs in Prolog about job listings. GEOQUERY (Zelle and Mooney, 1996) and ATIS (Price, 1990) include 880 and 5,410 utterance-LF pairs in lambda calculas about US geography and flight booking, respectively. The number of predicates in JOBS, GEOQUERY, ATIS is 15, 24, and 88, respectively. All atoms in the datasets are anonymized as in (Dong and Lapata, 2016).

For each dataset, we randomly selected m predicates as the new predicates, which is 3 for JOBS, and 5 for GEOQUERY and ATIS. Then we split each dataset into a train set and an *evaluation* set. And we removed the instances, the template of which is unique in each dataset. The number of such instances is around 100, 150 and 600 in JOBS, GEOQUERY, and ATIS. The ratios between the evaluation set and the train set are 1:4, 2:5, and 1:7 in JOBS, GEOQUERY, and ATIS, respectively. Each LF in an evaluation set contains at least a new predicate, while an LF in a train set contains only known predicates. To evaluate k-shot learning, we build a support set by randomly sampling k pairs per new predicate without replacement from an *evaluation* set, and keep the remaining pairs as the test set. To avoid evaluation bias caused by randomness, we repeat the above process six times to build six different splits of support and test set from each evaluation set. One for hyperparameter tuning and the rest for evaluation. We consider at most 2-shot learning due to the limited number of instances per new predicate in each evaluation set.

Training Details. We pre-train our parser on the training sets for {80, 100} epochs with the Adam optimizer (Kingma and Ba, 2014). The batch size is fixed to 64. The initial learning rate is 0.0025, and the weights are decayed after 20 epochs with decay rate 0.985. The predicate dropout rate is 0.5. The smoothing term is set to {3, 6}. The number of meta-support examples is 30 and the number of meta-test examples per support example is 15. The coefficient of attention regularization is set to 0.01 on JOBS and 1 on the other datasets. We employ the 200-dimensional GLOVE embedding (Pennington et al., 2014) to initialize the word embeddings for utterances. The hidden state size of all LSTM models (Hochreiter and Schmidhuber, 1997) is 256. During fine-tuning, the batch size is 2, the learning rates and the epochs are selected from {0.001, 0.0005} and {20, 30, 40, 60, 120}, respectively.

	Jobs	GeoQuery	Atis	Jobs	GeoQuery	Atis	p-values
Seq2Seq (*pt*)	11.27	20.00	17.23	14.58	33.01	18.76	3.32e-04
Seq2Seq (*cb*)	11.70	7.64	2.25	21.49	14.36	7.91	6.65e-06
Seq2Seq (*os*)	14.18	11.38	4.45	30.46	33.59	10.17	5.30e-05
Coarse2Fine (*pt*)	10.91	24.07	17.44	13.83	35.63	21.08	1.48e-04
Coarse2Fine (*cb*)	9.28	14.50	0.42	19.61	28.93	9.25	2.35e-06
Coarse2Fine (*os*)	6.73	10.35	5.26	16.08	28.55	17.73	1.13e-05
IRNet (*pt*)	16.00	20.00	17.12	19.06	35.05	20.11	2.86e-05
IRNet (*cb*)	19.67	21.90	5.60	28.22	44.08	15.73	2.76e-03
IRNet (*os*)	14.91	18.78	4.95	30.84	40.97	18.05	2.47e-04
DA	18.91	9.67	4.29	21.31	20.88	17.18	1.13e-06
PT-MAML	11.64	9.76	6.83	17.76	22.52	12.28	1.73e-06
Ours	**27.09**	**27.49**	**19.27**	**32.5**	**48.45**	**22.48**	

Table 2: Evaluation of learning results on three datasets. (Left) The one-shot results. (Right) The two-shot results.

Baselines. We compared our methods with five competitive baselines, Seq2Seq with attention (Luong et al., 2015), Coarse2Fine (Dong and Lapata, 2018), IRNet (Guo et al., 2019), PT-MAML (Huang et al., 2018) and DA (Li et al., 2020). Coarse2Fine is the best performing supervised model on the standard split of GeoQuery and Atis datasets. PT-MAML is a few-shot learning semantic parser that adopts Model-Agnostic Meta-Learning (Finn et al., 2017). We adapt PT-MAML in our scenario by considering a group of instances that share the same template as a pseudo-task. DA is the most recently proposed neural semantic parser applying domain adaptation techniques. IRNet is the strongest semantic parser that can generalize to unseen database schemas. In our case, we consider a list of predicates in support sets as the columns of a new database schema and incorporate the schema encoding module of IRNet into the encoder of our base parser. We choose IRNet over RATSQL (Wang et al., 2019) because IRNet achieves superior performance on our datasets.

We consider three different supervised learning settings. First, we pre-train a model on a train set, followed by fine-tuning it on the corresponding support set, coined *pt*. Second, a model is trained on the combination of a train set and a support set, coined *cb*. Third, the support set in *cb* is oversampled by 10 times and 5 times for one-shot and two-shot respectively, coined *os*.

Evaluation Details. The same as prior work (Dong and Lapata, 2018; Li et al., 2020), we report accuracy of exactly matched LFs as the main evaluation metric.

To investigate if the results are statistically significant, we conducted the Wilcoxon signed-rank test, which assesses whether our model consistently performs better than another baseline across *all* evaluation sets. It is considered superior than t-test in our case, because it supports comparison across different support sets and does not assume normality in data (Demšar, 2006). We include the corresponding p-values in our result tables.

5.1 Results and Discussion

Table 2 shows the average accuracies and significance test results of all parsers compared on all three datasets. Overall, `ProtoParser` outperforms all baselines with at least 2% on average in terms of accuracy in both one-shot and two-shot settings. The results are statistically significant w.r.t. the strongest baselines, IRNet (*cb*) and Coarse2Fine (*pt*). The corresponding p-values are 0.00276 and 0.000148, respectively. Given one-shot example on Jobs, our parser achieves 7% higher accuracy than the best baseline, and the gap is 4% on GeoQuery with two-shots examples. In addition, none of the SOTA baseline parsers can consistently outperform other SOTA parsers when there are few parallel data for new predicates. In one-shot setting, the best supervised baseline IRNet (*cb*) can achieve the best results on GeoQuery and Jobs among all baselines, and on two-shot setting, it performs best only on GeoQuery. It is also difficult to achieve good performance by adapting the existing meta-learning or transfer learning algorithms to our problem, as evident by the moderate performance of PT-MAML and DA on all datasets.

The problems of few-shot learning demonstrate the challenges imposed by infrequent predicates. There are significant proportions of infrequent predicates on the existing datasets. For example, on GeoQuery, there are 10 predicates contributing to only 4% of the total frequency of all 24 predicates, while the top two frequent predicates amount

	Jobs	GeoQuery	Atis	Jobs	GeoQuery	Atis	p-values
Ours	27.09	**27.49**	**19.27**	**32.50**	**48.45**	**22.48**	
- sup	23.63	18.86	12.91	26.91	39.51	14.89	1.44e-05
- proto	22.91	18.77	13.24	29.16	38.93	16.81	1.77e-05
- reg	**29.27**	18.10	13.66	31.03	39.61	18.58	9.60e-04
- strsim	22.18	19.62	10.14	28.41	47.09	19.98	9.27e-04
- cond	23.27	19.05	9.63	27.66	40.97	17.50	4.37e-05
- smooth	24.36	23.60	15.23	30.84	44.95	18.71	3.27e-03

Table 3: Ablation study results. (Left) The one-shot learning results. (Right) The two-shot learning results.

to 42%. As a result, the SOTA parsers achieve merely less than 25% and 44% of accuracy with one-shot and two-shots examples, respectively. In contrast, those parsers achieve more than 84% accuracy on the standard splits of the same datasets in the supervised setting.

Infrequent predicates in semantic parsing can also be viewed as a class imbalance problem, when support sets and train sets are combined in a certain manner. In this work, the ratio between the support set and the train set in Jobs, GeoQuery, and ATIS is 1:130, 1:100, and 1:1000, respectively. Different models prefer different ways of using the train sets and support sets. The best option for Coarse2Fine and Seq2Seq is to pre-train on a train set followed by fine-tuning on the corresponding support set, while IRNet favors oversampling in two-shot setting.

Ablation Study We examine the effect of different components of our parser by removing each of them individually and reporting the corresponding average accuracy. As shown in Table 3, removing any of the components almost always leads to statistically significant drop of performance. The corresponding p-values are all less than 0.00327.

To investigate predicate-dropout, we exclude either supervised-loss during pre-training (-sup) or initialization of new predicate embeddings by prototype vectors before fine-tuning (-proto). It is clear from Table 3 that ablating either supervisely trained action embeddings or prototype vectors hurts performance severely.

We further study the efficacy of attention regularization by removing it completely (-reg), removing only the string similarity feature (-strsim), or conditional probability feature (-cond). Removing the regularization completely degrades performance sharply except on Jobs in the one-shot setting. Our further inspection shows that model learning is easier on Jobs than on the other two datasets. Each predicate in Jobs almost always aligns to the same word across examples, while a predicate can align with different word/phrase in different examples in GeoQuery and ATIS. The performance drop with -strsim and -cond indicates that we cannot only reply on a single statistical measure for regularization. For instance, we cannot always find predicates take the same string form as the corresponding words in input utterances. In fact, the proportion of predicates present in input utterances is only 42%, 38% and 44% on Jobs, ATIS, and GeoQuery, respectively.

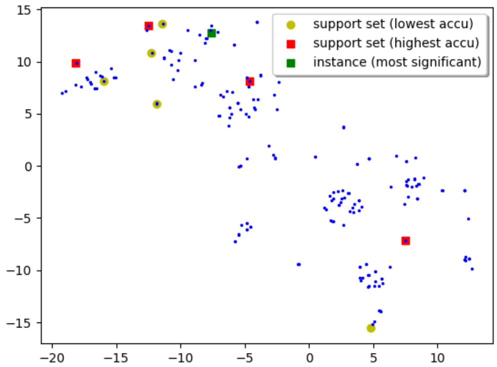


Figure 1: (Round) The support set with the lowest accuracy. (Box) The support set with the highest accuracy.

Furthermore, without pre-training smoothing (-smooth), the accuracy drops at least 1.6% in terms of mean accuracy on all datasets. Smoothing enables better model parameter training by more accurate modelling in pre-training.

Support Set Analysis We observe that all models consistently achieve high accuracy on certain support sets of the same dataset, while obtaining low accuracies on the other ones. We illustrate the reasons of such effects by plotting the evaluation set of GeoQuery. Each data point in Figure 1 depicts an representation, which is generated by the encoder of our parser after pre-training. We applied T-SNE (Maaten and Hinton, 2008) for dimension reduction. We highlight two support sets used in the one-shot setting on GeoQuery. All examples in the highest performing support set tend to scatter evenly and cover different dense regions in the feature space, while the examples in the lowest performing support set are far from a significant number of dense regions. Thus, the examples in good support sets are more representative of the underlying distribution than the ones in poor support sets. When we leave out each example in the highest performing support set and re-evaluate our parser each time, we observe that the good ones (e.g. the green box in Figure 1) locate either in or close to some of the dense regions.

6 Conclusion and Future Work

We propose a novel few-shot learning based semantic parser, coined `ProtoParser`, to cope with new predicates in LFs. To address the challenges in few-shot learning, we propose to train the parser with a pre-training procedure involving predicate-dropout, attention regularization, and pre-training smoothing. The resulted model achieves superior results over competitive baselines on three benchmark datasets.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Yujia Bao, Shiyu Chang, Mo Yu, and Regina Barzilay. 2018. Deriving machine attention from human rationales. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1903–1913.

Qingqing Cai and Alexander Yates. 2013. Semantic parsing freebase: Towards open-domain semantic parsing. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 328–338.

Shuaichen Chang, Pengfei Liu, Yun Tang, Jing Huang, Xiaodong He, and Bowen Zhou. 2019. Zero-shot text-to-sql learning with auxiliary task. *arXiv preprint arXiv:1908.11052*.

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. *arXiv preprint arXiv:1809.00773*.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. 2019. Learning an executable neural semantic parser. *Computational Linguistics*, 45(1):59–94.

Javid Dadashkarimi, Alexander Fabbri, Sekhar Tatikonda, and Dragomir R Radev. 2018. Zero-shot transfer learning for semantic parsing. *arXiv preprint arXiv:1808.09889*.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 43–48.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Catherine Finegan-Dollak, Jonathan K Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-sql evaluation methodology. *arXiv preprint arXiv:1806.09029*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with lstm.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-sql in cross-domain database with intermediate representation. *arXiv preprint arXiv:1905.08205*.

Jonathan Herzig and Jonathan Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. *arXiv preprint arXiv:1804.07918*.

Jonathan Herzig and Jonathan Berant. 2019. Don't paraphrase, detect! rapid and effective data collection for semantic parsing. *arXiv preprint arXiv:1908.09940*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2018. Few-shot charge prediction with discriminative legal attributes. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 487–498.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 732–738.

Srinivasan Iyer, Alvin Cheung, and Luke Zettlemoyer. 2019. Learning programmatic idioms for scalable semantic parsing. *arXiv preprint arXiv:1904.09086*.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. *arXiv preprint arXiv:1606.03622*.

Aishwarya Kamath and Rajarshi Das. 2018. A survey on semantic parsing. *CoRR*, abs/1812.00978.

Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, Masaaki Nagata, Hiroya Takamura, and Manabu Okumura. 2017. Supervised attention for sequence-to-sequence constituency parsing. *IJCNLP 2017*, page 7.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. 2016. Semantic parsing with semi-supervised sequential autoencoders. *arXiv preprint arXiv:1609.09315*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526.

Dongjun Lee. 2019. Clause-wise and recursive decoding for complex and cross-domain text-to-sql generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6047–6053.

Dongjun Lee, Jaesik Yoon, Jongyun Song, Sanggil Lee, and Sungroh Yoon. 2019. One-shot learning for text-to-sql generation. *arXiv preprint arXiv:1905.11499*.

Yoonho Lee and Seungjin Choi. 2018. Gradient-based meta-learning with learned layerwise metric and subspace. *arXiv preprint arXiv:1801.05558*.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Zechang Li, Yuxuan Lai, Yansong Feng, and Dongyan Zhao. 2020. Domain adaptation for semantic parsing. *arXiv preprint arXiv:2006.13071*.

Kevin Lin, Ben Bogin, Mark Neumann, Jonathan Berant, and Matt Gardner. 2019. Grammar-based neural text-to-sql generation. *arXiv preprint arXiv:1905.13326*.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

Richard Montague. 1973. The proper treatment of quantification in ordinary english. In *Approaches to natural language*, pages 221–242. Springer.

Ansong Ni, Pengcheng Yin, and Graham Neubig. 2019. Merging weak and active supervision for semantic parsing. *arXiv preprint arXiv:1911.12986*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Patti J Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990.*

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1139–1149.

Scott Reed, Yutian Chen, Thomas Paine, Aäron van den Oord, SM Eslami, Danilo Rezende, Oriol Vinyals, and Nando de Freitas. 2017. Few-shot autoregressive density estimation: Towards learning to learn distributions. *arXiv preprint arXiv:1710.10304*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087.

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. *arXiv preprint arXiv:1704.05974*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Yibo Sun, Duyu Tang, Nan Duan, Yeyun Gong, Xiaocheng Feng, Bing Qin, and Daxin Jiang. 2019. Neural semantic parsing in low-resource settings with back-translation and meta-learning. *arXiv preprint arXiv:1909.05438*.

I Sutskever, O Vinyals, and QV Le. 2014. Sequence to sequence learning with neural networks. *Advances in NIPS*.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2019. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *arXiv preprint arXiv:1911.04942*.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1332–1342.

Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer.

Pengcheng Yin and Graham Neubig. 2018. Tranx: A transition-based neural abstract syntax parser for semantic parsing and code generation. *arXiv preprint arXiv:1810.02720*.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. StructVAE: Tree-structured latent variable models for semi-supervised semantic parsing. In *The 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Rui Zhang, Tao Yu, He Yang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019a. Editing-based sql query generation for cross-domain context-dependent questions. *arXiv preprint arXiv:1909.00786*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. *arXiv preprint arXiv:1909.02607*.

Q. Zhu, X. Ma, and X. Li. 2019. Statistical learning for semantic parsing: A survey. *Big Data Mining and Analytics*, 2(4):217–239.

Alternating Recurrent Dialog Model with Large-scale Pre-trained Language Models

Qingyang Wu[1], **Yichi Zhang**[2], **Yu Li**[1], **Zhou Yu**[1]
[1]University of California, Davis, [2]Tsinghua University,
{wilwu, yooli, joyu}@ucdavis.edu, zhangyic17@mails.tsinghua.edu.cn

Abstract

Existing dialog system models require extensive human annotations and are difficult to generalize to different tasks. The recent success of large pre-trained language models has suggested the effectiveness of incorporating language priors in down-stream NLP tasks. However, how much pre-trained language models can help dialog response generation is still under exploration. In this paper, we propose a simple, general, and effective framework: Alternating Recurrent Dialog Model (ARDM)[1]. ARDM models each speaker separately and takes advantage of large pre-trained language models. It requires no supervision from human annotations such as belief states or dialog acts to achieve effective conversations. ARDM outperforms or is on par with the state-of-the-art methods on two popular task-oriented dialog datasets: CamRest676 and MultiWOZ. Moreover, we can generalize ARDM to more challenging, non-collaborative tasks such as persuasion. In the PersuasionForGood task, ARDM is capable of generating human-like responses to persuade people to donate to a charity.

1 Introduction

It has been a long-standing ambition for artificial intelligence researchers to create an intelligent conversational agent that can generate human-like responses. Recently, data-driven dialog models are more and more popular. However, most current state-of-the-art approaches still heavily rely on extensive human annotations such as belief states and dialog acts (Lei et al., 2018). However, dialog content can vary considerably in different dialog tasks. Having a different intent or dialog act annotation scheme for each task is costly and even impossible for tasks such as open-domain social chat. Thus, it is difficult to utilize these methods on challenging dialog tasks where dialog states and acts are difficult to annotate such as persuasion and negotiation.

Eric and Manning (2017) proposed a simple sequence-to-sequence architecture that requires no explicit annotations. The model learns to extract information from dialog history with attention and copy mechanism. However, due to the limited language modeling capability in the previous model, Sequicity (Lei et al., 2018), which uses belief states as inputs for supervision, outperforms Eric and Manning (2017)'s method significantly in the recent dialog datasets. But with the success of large pre-trained language models such as BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019), we investigate how large-scale pre-trained language models can help dialog tasks.

Previous sequence-to-sequence models are used to tackle documents with only one narrator. However, in dialogs, two speakers have different roles; therefore, their language model distributions are very different from each other. To address this issue, we propose ARDM, a dialog model that encodes and decodes different speaker utterances in alternating order. This structure makes the model more flexible and efficient than traditional sequence-to-sequence models in processing various dialogs. We evaluate our model on three different task-oriented dialog datasets: CamRes676, MultiWOZ, and PersuasionForGood. The first two datasets are traditional information request dialog datasets with well-defined automatic evaluation metrics on task completion. By contrast, PersuasionForGood is a new dataset that focuses on persuading people to donate to a charity. Due to the complexity of dialog content, there is no explicit dialog state defined in this task.

We observe that ARDM is capable of improving task-oriented dialog tasks performance over the previous state-of-the-art methods without in-

---

[1] https://github.com/qywu/ARDM

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1292–1301
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

corporating any explicit supervision from belief states or dialog acts. Also, because of ARDM's simplicity and generality, one can rapidly build a dialog prototype on different types of applications using only conversations without additional human annotations. We also found that ARDM works well on complex dialogs, such as persuasion. The model generates dialog responses that successfully persuade people to donate to a charity, suggesting the potential of ARDM being used in wide-scale real-world settings.

2 Related Work

Traditional dialog systems consist of a dialog manager to maintain dialog states and control the conversation flow. However, a dialog manager requires extensive manual annotations for training the sub-modules such as dialog state tracker or policy decision-maker. An alternative is to model dialog without explicitly modeling belief states. Specifically, Eric and Manning (2017) proposed a sequence-to-sequence model that utilizes copy-mechanism to copy history information directly from raw dialog history. This method achieved the state-of-the-art results on DSTC2 (Henderson et al., 2014), which is a simple dialog restaurant booking task with abundant data. However, such method did not perform well on more complex dialog task data sets CamRes676 (Wen et al., 2017) and KVRET (Eric et al., 2017). Sequicity (Lei et al., 2018) attributed the bad performance of Eric and Manning (2017)'s method to the omission of belief tracker. They introduced the concept of belief span and added belief tracker back to the model and achieved state-of-the-art performance.

Compared to Sequicity, Eric and Manning (2017)'s method provides a more general framework that reduces manual dialog state, user intent, and dialog act labeling by bypassing any symbolic annotations. Such a model can apply to datasets with no or partial annotations of belief states. In a real-world setting, if the dialog task introduces new slot values in belief states (i.e. a new type of food), Sequicity will suffer from the belief span decoder error in response generation. Thus, Eric and Manning (2017)'s method may be potentially more robust than Sequicity in this situation. Besides, if the task requires belief states for database search, we can treat belief tracking as a separate task. We can train a good belief tracking with only a small amount of annotated data, which reduces the annotation required and is easier to fix errors. Also, since belief states are a set of important entities condensed from dialog history (i.e., often exact words from utterances), they do not introduce extra information to the model. Therefore, a dialog model with powerful representation learning should learn a form of belief states information automatically without human annotations as the scaffold.

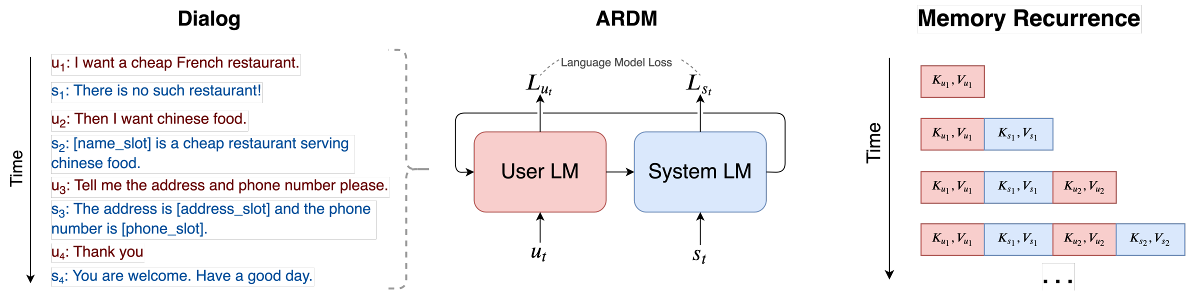
Recent success of BERT (Devlin et al., 2019) and GPT2 (Radford et al., 2019) suggests the possibility of applying large pre-trained language models to dialog systems. There are some studies of applying large pre-trained language model to dialog generation. TransferTransfo (Wolf et al., 2019) fine-tuned the pre-trained language model GPT (Radford et al., 2018) on Persona-Chat dataset (Zhang et al., 2018) and obtained significant improvements on chitchat response generation, suggesting the potential of fine-tuning large pre-trained language models on other dialog response generation tasks. A more recent work (Budzianowski and Vulic, 2019) adopted the framework of TransferTransfo. They made the first attempt to leverage large pre-trained language models GPT and GPT-2 on task-oriented dialog generation, but it included belief states modeling as the input and did not achieve better results than the baseline. We propose to model dialogs without any annotation but rely on pre-trained large-scale language models that alternate.

3 Alternating Recurrent Dialog Model

We propose Alternating Recurrent Dialog Model (ARDM) by compositing two separate pre-trained language model in alternate order to learn the user and system utterance distribution through memory recurrence. Figure 1 shows an overview of ARDM.

3.1 Recurrent Modeling for User and System

We aim to model both user and system utterances distribution recurrently. Given a multi-turn dialog (d) between a user (u) and a system (s), we can represent d as a series of utterances $\{u_1, s_1, u_2, s_2, \dots, u_T, s_T\}$, where T denotes the total number of turns. We decompose the probability distributions over the utterances in d into two language models for the user and system respectively, denoted as p_u and p_s. Then we define a



u_1: I want a cheap French restaurant.
s_1: There is no such restaurant!
u_2: Then I want chinese food.
s_2: [name_slot] is a cheap restaurant serving chinese food.
u_3: Tell me the address and phone number please.
s_3: The address is [address_slot] and the phone number is [phone_slot].
u_4: Thank you
s_4: You are welcome. Have a good day.

(a) (b)

Figure 1: Alternating Recurrent Dialog Model (ARDM) Overview. (a) shows how we feed the entire dialog to ARDM. (b) shows the recurrence mechanism we used to preserve memory.

dialog model $p(d)$ with the equation:

$$p(d) = \prod_{t=1}^{T} p_u(u_t|u_{<t}, s_{<t})\, p_s(s_t|u_{\leq t}, s_{<t}) \quad (1)$$

p_u and p_s are standard language models where the task is to predict the next token given the preceding context. For an utterance u_t or s_t with m tokens $\{w_1, \dots, w_m\}$, the joint probability of an utterance is as follows:

$$p_u(u_t|u_{<t}, s_{<t}) = \prod_{i=1}^{m_{u_t}} P(w_i|w_{<i}, u_{<t}, s_{<t}) \quad (2)$$

$$p_s(s_t|u_{\leq t}, s_{<t}) = \prod_{i=1}^{m_{s_t}} P(w_i|w_{<i}, u_{\leq t}, s_{<t}) \quad (3)$$

Finally, we train the dialog model by maximizing the likelihood over Equation 1. However, in order to model multi-turn dialog distributions efficiently, we need a memory mechanism to encode the history. In the next section, we introduce the memory mechanism in detail.

3.2 Memory Recurrence

We apply a memory mechanism to grant the model the capability of memorizing conversation history. This method is similar to Transformer-XL (Dai et al., 2019), which enables the model to learn longer dependency. For an utterance at turn t, we reuse the hidden states $h_{\leq t-1}$ stored in the memory M_{t-1} to obtain h_t, and store the h_t back to the memory as M_t. As for the pre-trained Transformer language model, we implement the memory mechanism using self-attention given the query/key/value features denoted as Q, K, V, where the equation is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T V) \quad (4)$$

For simplicity, we assume there is only one layer in Transformer, and h_t is the hidden states which consist of N vectors for the current input N tokens in the utterance at time t. Then a recurrence relation for h_t is defined by computing Q_t, $K_{\leq t}$, $V_{\leq t}$ from $h_{\leq t-1}$ and the current utterance. In practice, we reuse $K_{\leq t-1}$ and $V_{\leq t-1}$ (i.e. history keys and values) as M_{t-1} instead of h_{t-1} to avoid recomputing history information. Therefore, the final h_t is computed as:

$$\begin{aligned} M_{t-1} &= [K_{\leq t-1}, V_{\leq t-1}] && (5)\\ K_{\leq t}, V_{\leq t} &= [K_{\leq t-1}; K_t], [V_{\leq t-1}; V_t] && (6)\\ h_t &= \text{Attention}(Q_t, K_{\leq t}, V_{\leq t}) && (7) \end{aligned}$$

One can use h_t (consisting of vectors for each token) to get each token's probability to calculate the language model cross entropy loss to maximize $p(w_i|w< i, u_{<t}, s_{<t})$, shown in Figure 1.

3.3 Training Details

We initialize the user and the system language model with a large pre-trained language model GPT-2 small with 117M parameters (Radford et al., 2019). It is a Transformer (Vaswani et al., 2017) model with 12 heads, 768 hidden size, and 12 layers. The model is trained on a large scale corpus called WebText extracted from Reddit with at least three upvotes. The tokenizer is 50,257 size byte pair encoding (BPE) (Sennrich et al., 2016) that can encode and decode any text in a lossless manner to avoid out-of-vocabulary tokens. We follow a special format in GPT-2 as the "trigger" so that the model can zero-shot dialog response generation, by prefixing the user role token "A:" or "B:", and suffixing the end of utterance token "\n\n\n". This "trigger" approach is similar in other zero-shot scenarios mentioned in GPT-2 paper (e.g., that a "TL;DR" token can trigger GPT-2 to summarize the input text.) We further fine-tune ARDM on the specific task dataset. We apply AdamW optimizer (Loshchilov and Hutter, 2019), and the number of warm-up steps is set to be the number of batches in one epoch. The learning rate is set to 3×10^{-5}, and the dropout rate is set to 0.1 for all tasks. We conduct all experiments on a 11GB GPU.

3.4 Decoding Details

We decode utterances by nucleus sampling (Holtzman et al., 2019) with different hyper-parameters (top-p, top-k) for down-stream dialog tasks. We also vary the temperature of $T < 1$ to find the best setting for the specific down-stream dialog task. To handle both situations in the evaluation and the real-world use case, we have two decoding modes. For evaluation mode, we feed all past ground truth history before turn t to generate the corresponding utterance, so that we can evaluate the quality of generated dialog responses without concerning about the conversion flow. While in a real-world use case, we do not have ground truth history, and therefore we use the memory from previously generated responses and let the model dynamically interact with a human or another bot in turns. Because dialogs have different lengths, it is hard for ARDM to efficiently decode responses using traditional batch padding method. As a solution, we develop a dynamic dialog filtering algorithm to support fast decoding in batch. Such method speeds up the generation eight times faster. Please refer to Appendix for the method's details.

3.4.1 Belief Tracking

Database state means the database query results. In the example, since there is no match in the database satisfying the constraint "cheap french restaurant", it returns "0" and the domain "restaurant". Then the system can condition on the database state to generate "There is no such restaurant." We do not incorporate belief tracking because we want to fit different dialog tasks in which there may be no annotations of belief states. Also, (Eric and Manning, 2017) pointed out that user utterances already contains the belief state information to generate system response, meaning it is not necessary to explicitly input belief state into the model. For database search, one can train a separate belief tracker with a small portion of data that is annotated, which is more common in the real world cases.

4 Experiments and Results

Data scarcity is one of the biggest challenges in dialog research. It is costly to collect human-human conversations under a specific setting. It is even more time-consuming to annotate the corresponding belief states and dialog acts. With the success of transfer learning in NLP, we aim to mitigate the low-resource problem with the large pre-trained language model. We validate our proposed ARDM on three task-oriented dialog datasets, CamRest676, MulitWOZ, and PersuasionForGood.

4.1 CamRest676

CamRest676 is a relatively small dataset with 408/136/136 dialogs for train/validation/test. We follow Sequicity (Lei et al., 2018) to delexicalize tokens such as restaurant names, phone numbers, postcodes by replacing them with their slot names in utterances. We prepend database search results to the system utterance. An example database search results are "restaurant;3", where the first slot indicates its dialog domain, which is always "restaurant" in CamRest767, and the second slot represents the number of matched items in the database. We use greedy sampling for all methods during decoding. We use **BLEU-4** and **Success F1** to evaluate language generation quality and Success F1 to evaluate task success. Success F1 computes the F1 score of the generated responses on requested slots such as an address, phone number, or food type. Other than Sequicity, we also compare results by using GPT-2 alone as a language model for the entire dialog. This baseline is equivalent to ARDM without alternating recurrent modeling.

4.1.1 Results

We first test our method on a restaurant search dataset, CamRest676 (Wen et al., 2017). Table 1 shows all models' results with ground truth belief state or generated belief state. We first use ground truth belief state in all methods to evaluate their response generation quality. ARDM achieves the best BLEU and Success F1 score. We observe that after fine-tuning GPT-2 on the CamRest676, it achieves similar results compared to the previous state-of-the-art method, Sequicity with reinforcement fine-tuning. This suggests pre-trained large-scale language model, such as GPT-2, transfers the meaningful representations to help fine-tuning.

Model	Entity Match rate	Ground Truth Database State BLEU-4	 Success. F1	Generated Database State BLEU-4	 Success. F1
Regular Expression	0.960	-	-	-	-
Sequicity	0.923	21.4	0.852	21.4	0.853
Sequicity (w/o RL)	0.940	22.9	0.821	23.4	0.834
GPT-2-finetune	-	21.8	0.851	19.2	0.862
ARDM	-	**26.2**	**0.864**	**25.4**	**0.862**
ARDM (50% data)	-	25.9	0.859	23.4	0.851

Table 1: Results on CamRest676 dataset. Our method does not require human annotations from dialog states and dialog acts in comparison to Sequicity.

However, without the alternating mechanism, GPT-2 alone does not perform as well as ARDM in terms of both BLEU-4 and Success F1, especially in BLEU-4 (improved 19%). Without the recurrent modeling, the model blends two speakers language distribution and ignores the role difference. Moreover, to test if our model preserves its performance with even less training data, we reduce the training data to 50%, and the performance only drops slightly. With half of the training data, our method still performs significantly better than Sequicity. This result suggests ARDM is robust on low-resource settings due to the advantage of the large-scale pre-training language model.

We also evaluate all models with generated belief states instead of ground truth belief states. Sequicity generates belief tracker results, and its Entity Match rate is 0.927. Our model does not have a state tracker, so we write a separate simple regular expression to extract the occurrence of entities that appear in the database to support our model. Such state tracker achieves 0.960 in Entity Match rate. It suggests that state tracking may be accomplished in more straightforward ways other than training a neural network model on a large set of annotated data. With a simple state tracker, our proposed method still performs better than Sequicity, which trains the belief state and the response generation task jointly.

4.2 MultiWOZ

Here, we only use the ground truth database search result to be consistent with other methods. We perform delexicalization which is mentioned in the original MultiWOZ (Budzianowski et al., 2018). We prepend the database search results to the system response for as conditional input. Also, the database results now contain information about whether the booking is successful or not (i.e., succeed or fail). Again, we prefix the database results to the system response. Note that we do not use belief state or dialog act annotation provided by the dataset to train ARDM. We set the top-p to 0.2 and the temperature to 0.7.

We normalize the time's slot value in all dialogs into the 24-hour format and perform tokenization via spaCy[2]. We found that different papers report results with different versions of the evaluator, which makes it difficult to compare different methods fairly. We explain the differences among all versions of the evaluator in Appendix. In this paper, we follow LaRL's evaluator implementation, as it is more reasonable than others. We re-evaluate results for all methods with the same evaluator to ensure fairness.

We compare our model to the attention-based seq2seq model which is proposed as the MultiWOZ Baseline (Budzianowski et al., 2018), the HDSA (Chen et al., 2019) model that incorporates dialog act supervision as an inductive prior for model architecture, and the LaRL (Zhao et al., 2019) model which leverages latent action modeling and reinforcement learning to improve performance. We do not compare with GPT-2-finetune with our model in MultiWOZ because GPT-2-finetune's performance on CamRest676 is significantly worse than our model. Note that GPT-2-finetune is equivalent to ARDM without alternating parameterization.

The results are evaluated on BLEU-4, Inform Rate, and Success Rate. Inform and Success Rate measure whether the system response provides the recommendations and requested information given in the goal.

---

[2] https://spacy.io/

Model	Supervision Dialog State	 Dialog Act	Inform (%)	Success (%)	BLEU-4
Human	-	-	98.9	96.5	-
Baseline	✓	×	82.5	72.9	18.9
HDSA	✓	✓	**87.7**	73.4	**23.6**
LaRL	✓	×	82.8	**79.2**	12.8
ARDM	×	×	87.4	72.8	20.6

Table 2: Results on MultiWOZ. Supervision denotes whether a model leverages dialog state or/and dialog act annotations. All models use the ground truth dialog state for database search. ARDM without supervision from annotation can still achieve comparable results.

4.2.1 Results

The evaluation results are shown in Table 2. Without any supervision from dialog states or dialog acts, ARDM significantly outperforms the MultiWOZ Baseline and LaRL on BLEU-4 and Inform rate, and is on par with HDSA. However, HDSA uses dialog act supervision and a large pretrained language model, BERT. Our model requires no annotation and can achieve similar results. This suggests our recurrent modeling and large-scale pre-training methods work similarly as the useful dialog act annotations. All the results show that our method's excellent performance remains consistent in multi-domain dialogs.

We analyze the generated responses and find that if multiple domains have appeared in the conversation history, our model tends to make mistakes in answering the right domain for user requests. This finding suggests that the Maximum Likelihood Estimation (MLE) has limitations in directly optimizing the metric, while reinforcement Learning (RL) can hugely improve the task completion in a dialog system. This is why LaRL has a higher Success rate. However, we also observe that LaRL has a low BLEU-4 score, which indicates low readability in responses. Therefore, there is a trade-off between the generation quality and the task success rate in the RL setting.

4.3 PersuasionForGood

To showcase ARDM's performance on a dialog dataset where it is much more difficult to obtain belief states and dialog act annotations, we train and evaluate our model on PersuasionForGood (Wang et al., 2019) dataset. In this dataset, the persuader must persuade an assigned persuadee (i.e., a person who is asked to donate) to donate money (from their task payment) to a charity called "Save the Children".

4.3.1 Setting

This dataset has a much larger vocabulary size (8,141) than the previous task-oriented dialog datasets due to its non-collaborative dialog property. The conversation content is richer because two speakers are negotiating back and forth. The dataset consists of 1,017 dialogs where only 300 dialogs are annotated with dialog acts. Therefore, models that require dialog state or dialog act annotation are not applicable in this dataset. As ARDM does not require dialog acts for training, it fits well on the PersuasionForGood dataset. We remove the prefix for system inputs because the task does not involve any database search. We use TransferTransfo (Wolf et al., 2019) model as a baseline, since it does not require extra human annotations. TransferTransfo is based on large pre-trained language model, and it uses token type embedding to encode role information of the speaker. We finetune it on the PersuasionForGood dataset.

To generate diverse responses, we decode the response using the nucleus sampling (Holtzman et al., 2019) with a top-p of 0.9 and a temperature of 0.7. It is impossible to conduct an automatic evaluation on task success on this task due to the lack of annotation. We use perplexity to evaluate each model's language generation quality. We also conduct a human evaluation to validate each model's task success rate. We show some generated examples in the Appendix to provide more information on both models' generation quality.

4.3.2 Results

Table 3 shows the results for PersuasionForGood. Because ARDM applies better speaker modeling and recurrence mechanism, our model achieves

	Perplexity ↓	Human Preference ↑	Average Donation Amount ↑
TransferTransfo	19.9	34.7%	0.538
ARDM	**10.1**	**65.3%**	**0.807**

Table 3: Automatic Evaluation and Human Evaluation Results on the PersuasionForGood dataset.

lower perplexity compared to TransferTransfo. We did not report BLEU, because on the PersuasionForGood dataset, BLEU score cannot reflect the actual generation quality as a random sentence with common tokens "the, of, is, are..." already has 10.0+ BLEU-1 score. Also because the validation set only contains 100 samples, the result can have a high variance. Then the only way to evaluate this type of system is by human evaluation.

To comprehensively evaluate each model's performance, we recruit 14 human evaluators to chat with the two persuasive systems ten times to avoid the randomness produced by each model. In total, we collected 140 ratings. We ask them to select a preferred chat-bot and indicate how much they are willing to donate after talking to the chat-bot. As a result, human judges prefer ARDM over TransferTransfo and tends to donate more when talking to ARDM produced chat-bot. Our model achieved 27% more donations compared to TransferTransfo. This indicates that our systems are more persuasive. In some examples, such as the one in Table 4, our model generates coherent, natural, and persuasive responses.

5 Error Analysis

Since CamRest676 is similar to MultiWOZ in terms of task content and dialog structure, we only describe the errors in MultiWOZ for simplicity. We randomly selected 30 generated error responses from our model with zero inform and success score.

To our surprise, we observed that nearly 63.3% of errors are not really mistakes. It is mainly due to the limitation of the automatic evaluator. For example, at turn one, the user asks about a restaurant, and the ground truth system response is "the [restaurant_name] is located at ...", but the generated system response is "what food preference do you have?". Our generated response is correct with respect to the dialog context. It is narrowing down the restaurant choices before providing a restaurant recommendation. However, the evaluator sticks to the only possible response it has. Unless the user can dynamically interact with the system, there is no good way to change such mistakes in the automatic evaluator.

We find that another 20% errors our model makes are when the system asks information the user already provided. This type of errors calls for a better history representation. Another 10% errors are due to ignoring the user's request for information, such as phone number. However, when we look at the ground truth responses, some crowd workers also made such errors. So resolving these errors requires a cleaner training dataset. Finally, the rest of 6.7% errors are about incorrect dialog domain understanding. For example, the user is asking for a hotel, but we present a restaurant recommendation. This is because of the data noise during the delexicalization process in which some domain labels are wrong.

The donation persuasion system trained with TransferTransfo and our model has some common problems, such as inconsistency, lack of logic, and hallucination. For example, if the persuader provides the information about "Save the Children", then the persuadee asks "Can you tell me more about it?". The system ends up providing the same information as before. It also sometimes makes up facts that have never happened, such as "Save the Children has an operation about a hurricane in Hawaii". All those errors would prevent users from trusting the bot, and therefore resulting in less donation. However, we also observe that users have a higher tolerance for errors in the persuasion setting than the customer service setting.

Overall, our model performs better on PersuasionForGood by having longer and diversified utterances. This suggests our model which utilizes alternating parameters is more effective than TransferTransfo.

6 Discussions and Ethical Consideration

ARDM models speakers separately on top of a large pre-trained language model. Such simple adaptation demonstrates substantial performance gain. We suspect it is because the interleaved structure of two language models provides a collabora-

Role	Utterance
SYS	Hi, how are you doing today?
USR	I am good!
SYS	Have you ever heard of the charity Save the Children?
USR	No, I have not heard about it. Would you like to talk about it?
SYS	Save the Children is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
USR	That's cool. It is an organization for the good of this world.
SYS	Yes, it is. In the first two months of 2018 alone, 1,000 children were reportedly killed or injured in intensifying violence in Syria.
USR	That's sad.
SYS	We do need to do something about that. One of the things that we can do is make sure that we have a system in place that will ensure that every penny we spend goes directly to the children in need.
USR	I agree.
SYS	How much do you like to donate to the charity now? Your donation will be directly deducted from your task payment. You can choose any amount from $0 to all your payment ($2).
USR	I am willing to donate full my payment $2.
SYS	Wow, that is awesome! That is so kind of you!
USR	You are welcome.

Table 4: A conversation between a machine persuader trained by ARDM and a human persuadee.

tive learning frame of both the user and the system language distribution modeling. The memory is the only way for the user and system to communicate, as they do not share any weights in their networks. Thus, the user encoder needs to learn useful representations to make the system model for understanding its intent. Similarly, the system needs to do the same for the user model to improve its understanding. This alternative repeating process forces both the user and system models to preserve the dialog history effectively in the memory. One can interpret the memory as the implicit representation of belief states or dialog acts.

Another benefit of ARDM is that we will obtain both user and system utterance generators. We can let the two models talk to each other to generate new self-play dialogs (Silver et al., 2017). With self-play, one can rapidly build a large scale dialog dataset using adversarial filtering (Zellers et al., 2018). Such models can also potentially be used in reinforcement learning as user simulator to study complex dialog strategies as well. We will explore those possibilities in the future work.

Persuasion is a double-edged sword. Given the fast development of dialog systems, an ethical design principle must be in place throughout all stages of the development and evaluation. We choose the donation task is because it is a relatively simple task that benefits children. Second, when deploying the persuasive agents in real conversations, we need to keep the users informed of the nature of the system. By revealing the identity of the persuasive agent, the user should also have options to communicate directly with the human team behind the system. Lastly, by investigating persuasive dialog systems, we also envision to use them as an educational tool for the general public to learn to defend themselves against machine persuasion.

7 Conclusions

We propose to build Alternating Recurrent Dialog Model (ARDM), a simple, general, and effective dialog method that models user and system separately with large-scale pre-trained language models. Since ARDM does not require any annotations, it generalizes to different dialog applications. Experimental results on CamRest676 and MultiWOZ suggest that ARDM outperforms or is on-par with the current state-of-the-art methods that use manual annotation information, such as belief states and dialog acts. Furthermore, we find our model's excellent performance generalizes to more complex non-collaborative dialog settings. It can generate high-quality responses to persuade people to donate to charity. However, the easiness of training ARDM raises concerns about the misuse of the model in scenarios such as sales, harassment, or scam on a mass scale. We caution the public in deploying such systems in the real world.

References

Pawel Budzianowski and Ivan Vulic. 2019. Hello, it's GPT-2 - how can I help you? towards the use of pretrained language models for task-oriented dialogue systems. *CoRR*, abs/1907.05774.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In (Riloff et al., 2018), pages 5016–5026.

Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3696–3709. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics.

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

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pages 37–49. Association for Computational Linguistics.

Mihail Eric and Christopher D. Manning. 2017. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *CoRR*, abs/1701.04024.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the SIGDIAL 2014 Conference, The 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 18-20 June 2014, Philadelphia, PA, USA*, pages 263–272. The Association for Computer Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447, Melbourne, Australia. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. OpenAI.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors. 2018. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. *CoRR*, abs/1906.06725.

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve J. Young. 2017. Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3732–3741. PMLR.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *CoRR*, abs/1901.08149.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In (Riloff et al., 2018), pages 93–104.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2204–2213. Association for Computational Linguistics.

Tiancheng Zhao, Kaige Xie, and Maxine Eskénazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1208–1218. Association for Computational Linguistics.

On the Evaluation of Vision-and-Language Navigation Instructions

Ming Zhao **Peter Anderson** **Vihan Jain** **Su Wang**
Alexander Ku **Jason Baldridge** **Eugene Ie**
Google Research
`{astroming, pjand, vihan, wangsu, alexku, jridge, eugeneie}`
`@google.com`

Abstract

Vision-and-Language Navigation wayfinding agents can be enhanced by exploiting automatically generated navigation instructions. However, existing instruction generators have not been comprehensively evaluated, and the automatic evaluation metrics used to develop them have not been validated. Using human wayfinders, we show that these generators perform on par with or only slightly better than a template-based generator and far worse than human instructors. Furthermore, we discover that BLEU, ROUGE, METEOR and CIDEr are ineffective for evaluating grounded navigation instructions. To improve instruction evaluation, we propose an instruction-trajectory compatibility model that operates without reference instructions. Our model shows the highest correlation with human wayfinding outcomes when scoring individual instructions. For ranking instruction generation systems, if reference instructions are available we recommend using SPICE.

1 Introduction

Generating route instructions is a long studied problem with clear practical applications (Richter and Klippel, 2005). Whereas earlier work sought to create instructions for human wayfinders, recent work has focused on using instruction-generation models to improve the performance of agents that follow instructions given by people. In the context of Vision-and-Language Navigation (VLN) datasets such as Room-to-Room (R2R) (Anderson et al., 2018b), models for generating navigation instructions have improved agents' wayfinding performance in at least two ways: (1) by synthesizing new instructions for data augmentation (Fried et al., 2018; Tan et al., 2019), and (2) by fulfilling the role of a probabilistic speaker in a pragmatic reasoning setting (Fried et al., 2018). Such data augmentation is so effective that it is nearly ubiquitous in the best performing agents (Wang et al., 2019; Huang et al., 2019; Li et al., 2019).

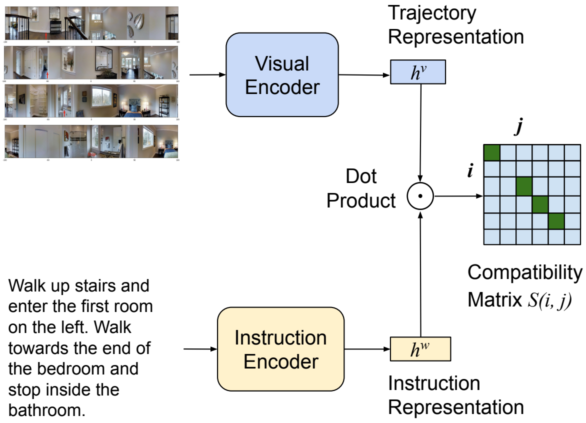


Figure 1: Proposed dual encoder instruction-trajectory compatibility model. Navigation instructions and trajectories (sequences of panoramic images and view angles) are projected into a shared latent space. The independence between the encoders facilitates learning using both contrastive and classification losses.

To make further advances in the generation of visually-grounded navigation instructions, accurate evaluation of the generated text is essential. However, the performance of existing instruction generators has not yet been evaluated using human wayfinders, and the efficacy of the automated evaluation metrics used to develop them has not been established. This paper addresses both gaps.

To establish benchmarks for navigation instruction generation, we evaluate existing English models (Fried et al., 2018; Tan et al., 2019) using human wayfinders. These models are effective for data augmentation, but in human trials they perform on par with or only slightly better than a template-based system, and they are far worse than human instructors. This leaves much headroom for better instruction generation, which may in turn improve agents' wayfinding abilities.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1302–1316
April 19 - 23, 2021.

Next, we consider the evaluation of navigation instructions without human wayfinders, a necessary step for future improvements in both grounded instruction generation (itself a challenging and important language generation problem) and agent wayfinding. We propose a model-based approach (Fig. 1) to measure the compatibility of an instruction-trajectory pair without needing reference instructions for evaluation. In training this model, we find that adding contrastive losses in addition to pairwise classification losses improves AUC by 9–10%, round-trip back-translation improves performance when used to paraphrase positive examples, and that both trajectory and instruction perturbations are useful as hard negatives.

Finally, we compare our compatibility model to common textual evaluation metrics to assess which metric best correlates with the outcomes of human wayfinding attempts. We discover that BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Denkowski and Lavie, 2014) and CIDEr (Vedantam et al., 2015) are ineffective for evaluating grounded navigation instructions. For system-level evaluations with reference instructions, we recommend SPICE (Anderson et al., 2016). When averaged over many instructions, SPICE correlates with both human wayfinding performance and subjective human judgments of instruction quality. When scoring individual instructions, our compatibility model most closely reflects human wayfinding performance, outperforming BERTScore (Zhang et al., 2019) and VLN agent-based scores. Our results are a timely reminder that textual evaluation metrics should always be validated against human judgments when applied to new domains. We plan to release our trained compatibility model and the instructions and human evaluation data we collected.

2 Related Work

Navigation Instruction Generation Until recently, most methods for generating navigation instructions were focused on settings in which a system has access to a map representation of the environment, including the locations of objects and named items (e.g. of streets and buildings) (Richter and Klippel, 2005). Some generate route instructions *interactively* given the current position and goal location (Dräger and Koller, 2012), while others provide *in-advance* instructions that must be more robust to possible misinterpretation (Roth and Frank, 2010; Mast and Wolter, 2013).

Recent work has focused on instruction generation to improve the performance of wayfinding agents. Two instruction generators, *Speaker-Follower* (Fried et al., 2018) and *EnvDrop* (Tan et al., 2019), have been widely used for R2R data augmentation. They provide ∼170k new instruction-trajectory pairs sampled from training environments. Both are seq-to-seq models with attention. They take as input a sequence of panoramas grounded in a 3D trajectory, and output a textual instruction intended to describe it.

Vision-and-Language Navigation For VLN, embodied agents in 3D environments must follow natural language instructions to reach prescribed goals. Most recent efforts (e.g., Fu et al., 2019; Huang et al., 2019; Jain et al., 2019; Wang et al., 2019, etc.) have used the Room-to-Room (R2R) dataset (Anderson et al., 2018b), which contains 4675 unique paths in the *train* split, 340 in the *val-seen* split (same environments, new paths), and an additional 783 paths in the *val-unseen* split (new environments, new paths). However, our findings are also relevant for similar datasets such as Touchdown (Chen et al., 2019; Mehta et al., 2020), CVDN (Thomason et al., 2019), REVERIE (Qi et al., 2020), and the multilingual Room-across-Room (RxR) dataset (Ku et al., 2020).

Text Generation Metrics There are many automated metrics that assess textual similarity; we focus on five that are extensively used in the context of image captioning: BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), ROUGE (Lin, 2004), CIDEr (Vedantam et al., 2015) and SPICE (Anderson et al., 2016). More recently, model- and semi-model-based metrics have been proposed. BERTScore (Zhang et al., 2019) takes a semi-model-based approach to compute token-wise similarity using contextual embeddings learned with BERT (Devlin et al., 2019). BLEURT (Sellam et al., 2020) is a fully model-based approach combining large-scale synthetic pretraining and domain specific finetuning. However, all of the aforementioned metrics are reference-based, and none is specifically designed for assessing navigation instructions associated with 3D trajectories for an embodied agent, which requires not only language-to-vision grounding but also correct sequencing.

Instruction-Trajectory Compatibility Models Our model builds on that of Huang et al. (2019), but differs in loss (using focal and contrastive losses), input features (adding action and geometry representation), and negative mining strategies (adding instruction perturbations in addition to trajectory perturbations). Compared to the trajectory re-ranking compatibility model proposed by Majumdar et al. (2020), we use a dual encoder architecture rather than dense cross-attention. This facilitates the efficient computation of contrastive losses, which are calculated over all pairs in a minibatch, and improve AUC by 10% in our model. We also avoid training on the outputs of the instruction generators (to prevent overfitting to the models we evaluate). We are yet to explore transfer learning (which is the focus of Majumdar et al. (2020)).

3 Human Wayfinding Evaluations

To benchmark the current state-of-the-art for navigation instruction generation, we evaluate the outputs of the *Speaker-Follower* and *EnvDrop* models by asking people to follow them. We use instructions for the 340 and 783 trajectories in the R2R val-seen and val-unseen splits, respectively. Both models are trained on the R2R train split and the generated instructions were provided by the respective authors. To contextualize the results, we additionally evaluate instructions from a template-based generator (using ground-truth object annotations), a new set of instructions written by human annotators, and three adversarial perturbations of these human instructions. New navigation instructions and wayfinding evaluations are collected using a lightly modified version of PanGEA[1], an open-source annotation toolkit for panoramic graph environments.

Crafty Crafty is a template-based navigation instruction generator. It observes the trajectory's geometry and nearby ground-truth object annotations, identifies salient objects, and creates English instructions using templates describing movement with respect to the trajectory and objects. See the Appendix for details. Note that Crafty has an advantage over the learned models which rely on panoramic images to identify visual references and do not exploit object annotations.

Human Instructions We collect 340 new English instructions for the trajectories in the R2R val-seen split using the PanGEA Guide task.

Instruction Perturbations To quantify the impact of common instruction generator failure modes on instruction following performance, we include three adversarial perturbations of human instructions capturing incorrect direction words, hallucinated objects/landmarks, and repeated or skipped steps. We use Google Cloud NLP[2] to identify named entities and parse dependency trees and then generate perturbations as follows:

- **Direction Swap**: Random swapping of directional phrases with alternatives from the same set, with sets as follows: *around/left/right, bottom/middle/top, up/down, front/back, above/under, enter/exit, backward/forward, away from/towards, into/out of, inside/outside*.

  Example: *"Take a right (**left**) and wait by the couch outside (**inside**) the bedroom. "*

- **Entity Swap**: Random swapping of entities in an instruction. All noun phrases excluding a stop list containing *any, first, end, front*, etc. are considered to be entities. If two entities have the same lemma (e.g., *stairs/staircase/stairway*) they are considered to be synonyms and are not swapped.

  Example: *"Exit the bedroom (**bathroom**), turn right, then enter the bathroom (**bedroom**)."*

- **Phrase Swap**: A random operation on the dependency tree: either remove one sub-sentence tree, duplicate one sub-sentence tree, or shuffle the order of all sentences except the last.

  Example: *"Exit the room using the door on the left. Turn slightly left and go past the round table an chairs. Wait there."* – where the first and second sentences are swapped.

Wayfinding Task Using the PanGEA Follower task, annotators are presented with a textual navigation instruction and the first-person camera view from the starting pose. They are instructed to attempt to follow the instruction to reach the goal location. Camera controls allow for continuous heading and elevation changes as well as movement between Matterport3D panoramas based on a navigation graph. Each instruction is evaluated by three different human wayfinders.

---

[1] https://github.com/google-research/pangea

[2] https://cloud.google.com/natural-language/

Instructions	Num. Evals	Wordcount	NE ↓	SR ↑	SPL ↑	SDTW ↑	Quality ↑	Visual Search Start ↓	Visual Search Other ↓	Time ↓
Val-unseen										
Speaker-Follower	783×3	24.6	6.55	35.8	30.3	28.1	3.50	43.5	24.8	43.2
EnvDrop	783×3	21.3	5.89	42.3	36.1	33.5	3.70	42.1	24.8	**39.5**
Val-seen										
Speaker-Follower	340×3	24.7	6.23	42.3	35.7	33.2	3.64	43.0	24.5	42.6
EnvDrop	340×3	22.3	5.99	47.7	40.0	36.9	3.83	39.9	25.0	42.1
Crafty	340×3	71.2	6.01	43.6	34.7	33.3	3.48	42.0	25.9	69.6
Direction Swap	340×3	54.8	4.74	58.9	47.9	45.9	3.67	40.6	25.4	61.0
Entity Swap	340×3	55.1	4.71	51.3	42.5	40.6	3.33	40.1	25.8	62.7
Phrase Swap	340×3	52.0	4.07	62.6	51.6	49.9	3.85	38.7	24.7	58.0
Human	340×3	54.1	**2.56**	**75.1**	**64.7**	**63.1**	**4.25**	**35.8**	**23.8**	53.9

Table 1: Human wayfinding performance following instructions from the Speaker-Follower (Fried et al., 2018) and EnvDrop (Tan et al., 2019) models, compared to Crafty (template-based) instructions, Human instructions, and three adversarial perturbations of Human instructions (Direction, Entity and Phrase Swap).

Evaluation Metrics We use the following standard metrics to evaluate the trajectories generated by our annotators (and hence, the quality of the provided instructions): *Navigation Error* (**NE** ↓) *Success Rate* (**SR** ↑), *Success weighted by inverse Path Length* (**SPL** ↑), *Success weighted by normalized Dynamic Time Warping* (**SDTW** ↑). Arrows indicate improving performance. See Anderson et al. (2018a) and Ilharco et al. (2019) for details.

People are resourceful and may succeed in following poor quality instructions by expending additional effort. Therefore, we report additional metrics to capture these costs. **Quality** ↑ is a self-reported measure of instruction quality based on a 1–5 Likert scale. At the end of each task annotators respond to the prompt: *Do you think there are mistakes in the instruction?* Responses range from *Way too many mistakes to follow (1)* to *No mistakes, very very easy to follow' (5).* **Visual Search** cost ↓ measures the percentage of the available panoramic visual field that the annotator observes at each viewpoint, based on the pose traces provided by PanGEA and first proposed in the RxR dataset (Ku et al., 2020). Higher values indicate greater effort spent looking for the correct path. We report this separately for the start viewpoint and other viewpoints since wayfinders typically look around to orient themselves at the start. **Time** ↓ represents the average time taken in seconds.

Results Table 1 summarizes the results of 11,886 wayfinding attempts using 37 English-speaking annotators. The performance of annotators stays consistent over time and does not show any sign of adaptation. See Appendix for detailed analysis.

As expected, human instructions perform best in human wayfinding evaluations on all path evaluation metrics and on subjective assessments of instruction quality, and they also incur the lowest visual search costs. The only metric not dominated by human instructions is the time taken – which correlates with instruction length, and may be affected by wayfinders giving up when faced with poor quality instructions. Overall, the Speaker-Follower and EnvDrop models are surprisingly weak and noticeably worse than even adversarially perturbed human instructions. Compared to the template-based approach (Crafty), the Speaker-Follower model performs on par and EnvDrop is only slightly better. As a first step to improving existing navigation instruction generators, we focus on developing and evaluating automated metrics that can approximate these human wayfinding evaluations.

4 Compatibility Model

As an alternative to human evaluations, we train an instruction-trajectory compatibility model to assess both the grounding between textual and visual inputs and the alignment of the two sequences.

4.1 Model Structure

Our model is a dual encoder that encodes instructions and trajectories into a shared latent space (Figure 1). The instruction representation h^w is the concatenation of the final output states of a bi-directional LSTM (Schuster and Paliwal, 1997) encoding the instruction tokens $\mathcal{W} = \{w_1, w_2, ..., w_n\}$. We use contextualized token embeddings from BERT (Devlin et al., 2019) as input to the LSTM.

The visual encoder is a two-layer LSTM that processes visual features extracted from a sequence of viewpoints $\mathcal{V} = \{(I_1, p_1), (I_2, p_2)..., (I_t, p_t)\}$ comprised of panoramic images I_t captured at positions p_t along a 3D trajectory. The vector h_t^v representing the viewpoint at step t is given by:

$$a_t = \text{Attention}(h_{t-1}^v, e_{\text{pano},t}) \quad (1)$$

$$v_t = f([e_{\text{prev},t}, e_{\text{next},t}, a_t]) \quad (2)$$

$$h_t^v = \text{LSTM}(v_t, h_{t-1}^v) \quad (3)$$

where $e_{\text{pano},t}$ is a set of 36 visual features representing the panoramic image I_t (discretized into 36 viewing angles by elevation θ and heading ϕ), $e_{\text{prev},t}$ and $e_{\text{next},t}$ are the visual features in the directions of the previous and next viewpoints ($v_{t-1} - v_t$ and $v_{t+1} - v_t$ respectively), and f is a projection layer. Each visual feature is a concatenation of a pre-trained CNN image feature (Juan et al., 2020) with orientation vectors encoding both sine and cosine functions of the absolute and relative angles $\{\theta_{\text{abs}}, \phi_{\text{abs}}, \theta_{\text{rel}}, \phi_{\text{rel}}\}$. We use standard dot-product attention (Luong et al., 2014) and define $h^v = h_T^v$, the final viewpoint embedding in the trajectory. The output of the model is the compatibility score S between an instruction and a trajectory defined as the cosine similarity between h^v and h^w.

4.2 Hard Negative Mining

To avoid overfitting, our compatibility model is *not trained on the outputs of any of the instruction generators that we evaluate*. Instead, we use only the relatively small set of positive instruction-trajectory examples from R2R. We use round-trip back-translation to expand the set of positive examples. Unmatched instruction-trajectory pairs from R2R are considered to be negative examples and we also construct hard negative examples from positive examples by adversarially perturbing both trajectories and instructions.

Instruction Perturbations We use the same instruction perturbations described in Section 3: *Direction Swap*, *Entity Swap*, and *Phrase Swap*. These perturbations are inspired by typical failure modes in instruction generators and are designed to be hard to recognize without grounding on images and actions along the trajectory. Previous work by Huang et al. (2019) considered only trajectory perturbations. While this encourages the model to recognize incorrect trajectories for a given ground truth instruction, it may not encourage the model to identify a trajectory matched with a poor quality instruction. Our results suggest that instruction perturbations are equally important.

Trajectory Perturbations To perturb trajectories we use the navigation graphs defining connected viewpoints in R2R. Inspired by Huang et al. (2019), we consider *Random Walk*, *Path Reversal*, and *Viewpoint Swap* perturbations:

- **Random Walk**: The first or last two viewpoints are fixed and the remainder of the trajectory is re-sampled using random edge traversals subject to the path length remaining within ±1 step of the original. To make the task harder, we avoid revisiting a viewpoint and require the re-sampled trajectory to have at least two overlapping viewpoints with the original.

- **Path Reversal**: The entire trajectory is reversed while keeping the same viewpoints.

- **Viewpoint Swap**: A new method we introduce that randomly samples and swaps a viewpoint in a trajectory with a new viewpoint sampled from the neighbors of the adjacent viewpoints in the original trajectory.

Paraphrases To expand the 14k positive examples from the R2R train set and balance the positive-to-negative ratio, we paraphrase instructions via round-trip back-translation. We use the following ten intermediate languages and Google Translate[3]: *ar, es, de, fr, hi, it, pt, ru, tr*, and *zh*. To exclude low quality or nearly duplicate instructions, we filter paraphrased instructions outside the BLEU score range of [0.25, 0.7] compared to the original. Overall we have a total of 110,601 positive instruction-trajectory pairs in the training set, which contains 4675 unique trajectories.

4.3 Loss Functions

During training, each minibatch is constructed with N matching instruction-trajectory pairs, which may be perturbed. We define $M \in \{0, 1\}^N$ as the vector indicating unperturbed pairs. A compatibility matrix $S \in \mathcal{R}^{N \times N}$ is defined such that $S_{i,j}$ is the cosine similarity score between instruction i and trajectory j determined by our model.

We use both binary classification loss functions, defined on diagonal elements of S, and a contrastive loss defined on S's rows and columns. Contrastive losses are commonly used for retrieval and representation learning (e.g., Yang et al., 2019;

---

[3] https://cloud.google.com/translate/

					Instruction Validation			**Path Validation**		
	Model	**Perturbations**	**Validation** (val_unseen)	**Test** (val_seen)	Direction Swap	Entity Swap	Phrase Swap	Viewpoint Swap	Random Walk	Path Reversal
1	Huang et al. (2019)	Path + Instruction	53.4	52.3	80.3	89.4	80.9	78.0	72.9	75.9
	This Work									
2	+ CE Loss	Path + Instruction	57.9	57.6	89.5	88.4	83.4	95.0	94.1	87.8
3	+ Focal Loss	Path + Instruction	59.2	59.2	**89.8**	90.5	84.2	95.6	95.0	90.3
4	+ Contrastive Loss	Path + Instruction	67.2	68.7	75.1	70.1	73.7	73.6	88.7	93.1
5	+ Contrastive + CE	Path + Instruction	66.5	67.5	82.0	77.4	76.9	84.7	91.3	91.7
6	+ Contrastive + Focal	Path + Instruction	68.5	68.3	83.9	81.5	79.8	88.5	93.7	93.3
7	+ Contrastive + Focal + Paraphrase	Path + Instruction	71.3	72.2	83.1	81.9	80.4	90.4	95.2	**94.0**
8	+ Contrastive + Focal + Paraphrase + Bert Embed.	Path + Instruction	**73.5**	**73.7**	84.6	86.7	82.2	89.1	94.3	93.3
	Perturbation Ablations									
9	+ Contrastive + Focal + Paraphrase + Bert Embed.	Instruction	70.5	70.4	85.1	88.6	82.5	64.9	84.3	90.7
10	+ Contrastive + Focal + Paraphrase + Bert Embed.	Direction Swap Only	70.1	68.9	89.0	72.0	70.7	65.9	85.0	91.0
11	+ Contrastive + Focal + Paraphrase + Bert Embed.	Entity Swap Only	69.1	68.5	70.3	**92.7**	71.3	65.2	84.4	90.9
12	+ Contrastive + Focal + Paraphrase + Bert Embed.	Phrase Swap Only	70.3	70.5	70.1	72.5	**85.5**	65.5	83.3	90.1
13	+ Contrastive + Focal + Paraphrase + Bert Embed.	Path	69.7	69.4	71.7	71.4	69.8	92.4	94.8	92.2
14	+ Contrastive + Focal + Paraphrase + Bert Embed.	Viewpoint Swap Only	70.7	72.1	70.7	71.1	71.2	**94.6**	94.6	91.9
15	+ Contrastive + Focal + Paraphrase + Bert Embed.	Random Walk Only	70.5	70.9	70.5	71.5	71.0	81.7	**95.1**	91.6
16	+ Contrastive + Focal + Paraphrase + Bert Embed.	Path Reversal Only	69.4	69.7	71.0	70.7	71.2	65.3	86.3	91.9
17	+ Contrastive + Focal + Paraphrase + Bert Embed.	No Perturbation	69.1	69.7	71.1	71.0	70.8	65.3	84.9	90.2

Table 2: Ablation of different models based on classification AUC. Models are trained with the original R2R-train data and paraphrased positive instructions, plus path and/or instruction perturbed hard negatives (second column). The best models are selected based on the validation set (column 3), and we report the final test performance in column 4. To understand the performance of individual perturbation method, we also report the best AUCs for each of the six perturbations in columns 5 - 10.

Chen et al., 2020) and in our case exploits all random instruction-trajectory pairs in a minibatch.

Each loss requires a separate normalization. For the classification loss we compute the probability of a match $p_{i,j}$, such that $p_{i,j} = \sigma(aS_{i,j} + b)$ where a and b are learned scalars and σ is the sigmoid function. For the classification loss $\mathcal{L}_{\text{cls}}$ we consider both binary cross entropy loss $\mathcal{L}_{\text{CE}}$, and focal loss (Lin et al., 2017) given by $\mathcal{L}_{\text{FL}} = (1 - p_{i,j})^{\gamma}\mathcal{L}_{\text{CE}}$ where we set $\gamma = 2$.

For the contrastive loss we compute logits by scaling S with a learned scalar temperature τ. The contrastive loss $\mathcal{L}_C(S)$ calculated over the rows and colums of S is given by:

$$\mathcal{L}_C(S) = \frac{1}{\sum_i M_i} \sum_{i=1}^{N} \Big(\mathcal{L}_r(S_i) + \mathcal{L}_r(S_i^{\intercal})\Big) \quad (4)$$

where $\mathcal{L}_r(S_i) = 0$ if $M_i = 0$, i.e., the diagonal element is a perturbed pair and not considered to be a match. Otherwise:

$$\mathcal{L}_r(S_i) = -\log \frac{e^{S_{i,i}/\tau}}{\sum_j^N e^{S_{i,j}/\tau}} \quad (5)$$

The final loss is the combination:

$$\mathcal{L} = \mathcal{L}_C(S) + \frac{\beta}{N} \sum_{i=1}^{N} \mathcal{L}_{cls}(S_{i,i}) \quad (6)$$

where $\mathcal{L}_{cls}$ is the classification loss, either $\mathcal{L}_{CE}$ or $\mathcal{L}_{FL}$, and we set $\beta = 1$.

Sampling hyperparameters We sample positive and negative examples equally with a mix ratio of 2:1:1 for ground truth, instruction perturbations, and trajectory perturbations, respectively. For each perturbation type, we sample the three methods with equal probability.

5 Experiments

We evaluate our compatibility model against alternative model-based evaluations and standard textual similarity metrics. We report instruction classification results in Section 5.1, improved data augmentation for VLN agents in Section 5.2, and correlation with human wayfinder outcomes in 5.3.

5.1 Instruction Classification

Evaluation In this setting we use the instruction-trajectory compatibility model to classify high and low quality instructions for trajectories from the R2R val-unseen and val-seen sets. The instruction pool includes 3 high-quality instructions per trajectory from R2R, plus 2 instructions per trajectory from the Speaker-Follower and EnvDrop models. These are considered to be high quality if 2 out of 3 human wayfinders reached the goal (see Section 3), and low quality otherwise. We assess model performance using *Area Under the ROC Curve* (AUC). We use the val-unseen split (3915 instructions, 75% high quality) for model validation and the val-seen split (1700 instructions, 78% high quality) as test.

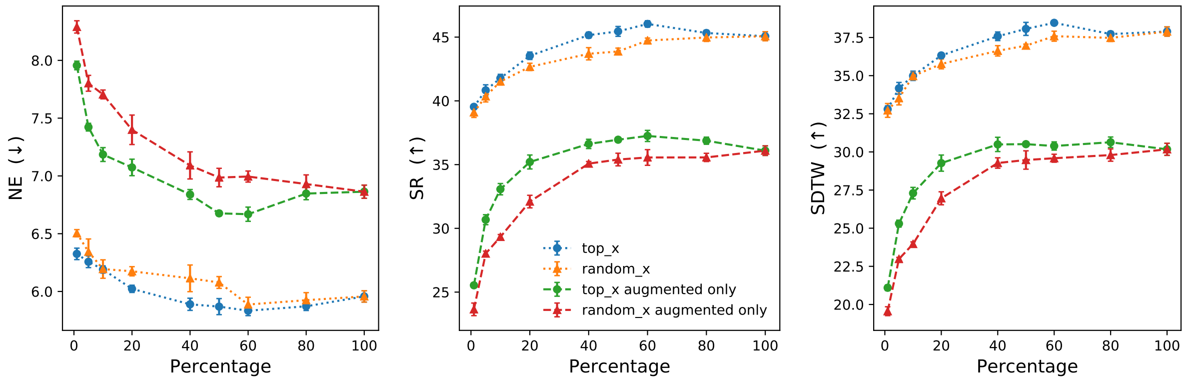


Figure 2: Navigation performance of a VLN agent trained with different fractions of Speaker-Follower augmented paths, starting from 1%. For NE, lower is better; for SR and SDTW, higher is better. The dashed-lines (green and red) use only augmented paths in training, while the dotted lines (blue and orange) use both augmented and R2R-train paths. Filled circles indicate fractions ranked by our compatibility model in descending order, while triangles indicate random fractions. Each point is the mean of 3 runs and the error bars represent the standard deviation of the mean. The model-ranked fractions show consistent improvement over random samples of the same percentage. Agents trained with only augmented paths (dashed-lines) show greater difference between model-ranked fractions and random fractions.

Benchmark We compare to the compatibility model proposed by Huang et al. (2019), which computes elementwise similarities between instruction words and trajectory panoramas *before* pooling, and does not include action embeddings ($e_{\text{prev},t}$ and $e_{\text{next},t}$) or position encodings p_t. In contrast, our model calculates the similarity between instructions and trajectories *after* pooling each sequence and includes both action and position encodings.

Results Table 2 reports classification AUC including comprehensive ablations of loss functions, approaches to hard negative mining, and modeling choices. With regard to the loss function, we find that the combination of contrastive and focal loss (row 6) performs best overall, and that adding contrastive loss provides a very significant 9 - 10% increase in AUC compared to using just cross-entropy (CE) or focal loss (rows 2 and 3) due to the effective use of in-batch negatives. Adding paraphrased positive instructions and pretrained BERT token embeddings also leads to significant performance gains (rows 7 and 8 vs. row 6). The best performing model on both the validation and test sets uses Contrastive + Focal loss with paraphrased instructions and BERT embeddings, as well as trajectory and instruction perturbations (row 8). This model consistently outperforms the benchmark from prior work (row 1) by a large margin and achieves a test set AUC of 73.7%.

In rows 9–17 we ablate the six perturbation methods that we use for hard negative mining. Ablations using only instruction perturbations (row 9), only path perturbations (13), or no perturbations at all (row 17) perform considerably worse than our best model (row 8). We also show that no individual perturbation approach is effective on its own. In addition to scores for the validation and test sets, we report AUC for each perturbation method on the val-seen set to investigate their individual performance. Overall, trajectory perturbations get higher scores than instruction perturbations, showing they are easier tasks. *Phrase Swap* proves the hardest task, while *Random Walk* is the easiest.

5.2 Data Augmentation for VLN

Data augmentation using instructions from the Speaker-Follower and EnvDrop models is pervasive in the training of VLN agents (Wang et al., 2019; Huang et al., 2019; Li et al., 2019). In this section we evaluate whether our compatibility model can be used to filter out low quality instructions from the augmented training set to improve VLN performance. We score all of 170k augmented instruction-trajectory pairs from the Speaker-Follower model and rank them in descending order. We then use different fractions of the ranked data to train VLN agents, and compare with agents trained using random samples of the same size. We use a VLN agent model based on Wang et al. (2019) and implemented in VALAN (Lansing et al., 2019), which achieves a success rate (SR) of 45% on the R2R val-unseen split when trained on the R2R train split and all of the Speaker-Follower augmented instructions.

Figure 2 indicates that instruction-trajectory pairs selected by our compatibility model consistently outperform random training pairs in terms of the performance of the trained VLN agent. This demonstrates the efficacy of our compatibility model for improving VLN data augmentation by identifying high quality instructions.

5.3 Correlation with Human Wayfinders

In this section we evaluate the correlation between the scores given by our instruction-trajectory compatability model and the outcomes from the human wayfinding attempts described in Section 3. Using Kendall's τ to assess rank correlation, we report both system-level and instance-level correlation. The instance-level evaluations assess whether the metric can identify the best *instruction* from two candidates, while the system-level evaluations assess whether a metric can identify the best *model* from two candidates (after averaging over many instruction scores for each model). The results in Table 3 are reported separately over all 3.9k instructions (9 systems comprising the rows of Table 1), and over model-generated instructions only (4 systems comprising the 2.2k instructions generated by the Speaker-Follower and EnvDrop models on R2R val-seen and val-unseen).

Automatic Metrics For comparison we include standard textual evaluation metrics (BLEU, CIDEr, METEOR, ROUGE and SPICE) and two model-based metrics: BERTScore (Zhang et al., 2019), and scores based on the performance of a trained VLN agent attempting to follow the candidate instruction (Agarwal et al., 2019). Note that only the compatibility model and the VLN agent-based scores use the candidate trajectory – the other metrics are calculated by comparing each candidate instruction to the three reference instructions from R2R (and are thus reliant on reference instructions).

To calculate the standard metrics we use the official evaluation code provided with the COCO captions dataset (Chen et al., 2015). For BERTScore, we use a publicly available uncased BERT[4] model with 12 layers and hidden dimension 768, and compute the mean $F1$-score over the three references. For the VLN agent score, we train three VLN agents based on Wang et al. (2019) from different random initializations using the R2R train set.

			All Instructions (N=3.9k, M=9)			
	Score	**Ref**	**NE ↓**	**SR ↑**	**SPL ↑**	**Quality ↑**
System-Level	BLEU-4	✓	(0.00, 0.33)	(-0.22, 0.39)	(-0.22, 0.00)	(0.11, 0.39)
	CIDEr	✓	(0.06, 0.39)	(-0.22, 0.39)	(-0.22, 0.00)	(0.17, 0.39)
	METEOR	✓	(0.11, 0.44)	(-0.39, 0.28)	(-0.39, -0.06)	(0.00, 0.28)
	ROUGE	✓	(0.06, 0.39)	(-0.28, 0.39)	(-0.33, 0.00)	(0.06, 0.39)
	SPICE	✓	(**-0.67**, **-0.28**)	(**-0.06**, **0.61**)	(**0.44**, **0.78**)	(**0.56**, **0.83**)
	BERTScore	✓	(0.06, 0.39)	(-0.22, 0.39)	(-0.22, 0.00)	(0.17, 0.39)
	SPL$_{\text{1-agent}}$		(-0.50, -0.06)	(-0.22, 0.44)	(0.11, 0.56)	(0.00, 0.44)
	SPL$_{\text{3-agents}}$		(-0.22, 0.17)	(-0.33, 0.39)	(0.00, 0.33)	(0.33, 0.61)
	SDTW$_{\text{1-agent}}$		(-0.44, 0.00)	(-0.22, 0.44)	(0.11, 0.50)	(0.00, 0.44)
	SDTW$_{\text{3-agents}}$		(-0.22, 0.17)	(-0.28, 0.33)	(0.00, 0.33)	(0.33, 0.61)
	Compatibility		(-0.17, 0.17)	(-0.17, 0.50)	(0.00, 0.28)	(0.44, 0.72)

			All Instructions (N=3.9k, M=9)			
	Score	**Ref**	**NE ↓**	**SR ↑**	**SPL ↑**	**Quality ↑**
Instance-Level	BLEU-4	✓	(0.05, 0.09)	(-0.04, 0.00)	(-0.09, -0.05)	(-0.01, 0.03)
	CIDEr	✓	(0.06, 0.09)	(-0.04, -0.00)	(-0.11, -0.07)	(-0.02, 0.01)
	METEOR	✓	(0.00, 0.04)	(-0.05, -0.02)	(-0.04, 0.00)	(-0.01, 0.02)
	ROUGE	✓	(0.05, 0.08)	(-0.05, -0.01)	(-0.10, -0.06)	(-0.02, 0.02)
	SPICE	✓	(-0.05, -0.02)	(-0.00, 0.04)	(0.03, 0.06)	(0.03, 0.07)
	BERTScore	✓	(-0.04, -0.00)	(0.07, 0.12)	(-0.01, 0.03)	(0.07, 0.11)
	SPL$_{\text{1-agent}}$		(-0.18, -0.14)	(0.15, 0.19)	(0.14, 0.18)	(0.07, 0.11)
	SPL$_{\text{3-agents}}$		(**-0.22**, **-0.18**)	(**0.20**, **0.24**)	(**0.18**, **0.22**)	(0.10, 0.14)
	SDTW$_{\text{1-agent}}$		(-0.18, -0.14)	(0.15, 0.19)	(0.14, 0.18)	(0.08, 0.12)
	SDTW$_{\text{3-agents}}$		(**-0.22**, **-0.19**)	(**0.20**, **0.24**)	(**0.18**, **0.22**)	(0.11, 0.15)
	Compatibility		(**-0.20**, **-0.17**)	(0.13, 0.17)	(**0.17**, **0.20**)	(**0.19**, **0.23**)

			Model-Generated Instructions (N=2.2k, M=4)			
	Score	**Ref**	**NE ↓**	**SR ↑**	**SPL ↑**	**Quality ↑**
Instance-Level	BLEU-4	✓	(-0.02, 0.03)	(-0.03, 0.02)	(-0.02, 0.03)	(-0.02, 0.03)
	CIDEr	✓	(-0.02, 0.03)	(-0.03, 0.02)	(-0.02, 0.03)	(-0.02, 0.03)
	METEOR	✓	(-0.02, 0.03)	(-0.03, 0.02)	(-0.02, 0.03)	(-0.02, 0.03)
	ROUGE	✓	(-0.02, 0.03)	(-0.05, 0.00)	(-0.04, 0.01)	(-0.03, 0.02)
	SPICE	✓	(-0.05, -0.00)	(0.00, 0.05)	(0.00, 0.05)	(0.01, 0.06)
	BERTScore	✓	(-0.22, -0.18)	(0.19, 0.24)	(0.18, 0.23)	(0.16, 0.20)
	SPL$_{\text{1-agent}}$		(-0.21, -0.16)	(0.17, 0.23)	(0.16, 0.22)	(0.07, 0.12)
	SPL$_{\text{3-agents}}$		(**-0.26**, **-0.21**)	(**0.21**, **0.27**)	(**0.21**, **0.26**)	(0.09, 0.14)
	SDTW$_{\text{1-agent}}$		(-0.22, -0.16)	(0.17, 0.23)	(0.16, 0.22)	(0.07, 0.13)
	SDTW$_{\text{3-agents}}$		(**-0.26**, **-0.21**)	(**0.22**, **0.27**)	(**0.21**, **0.26**)	(0.10, 0.15)
	Compatibility		(**-0.25**, **-0.20**)	(**0.22**, **0.27**)	(**0.21**, **0.25**)	(**0.18**, **0.23**)

Table 3: Kendall's τ correlation between automated instruction evaluation metrics and human wayfinder evaluations. Ranges are 90% confidence intervals based on bootstrap resampling. **N** refers to the number of instructions and **M** refers to the number of systems. If checked, **Ref** indicates that the metric requires reference instructions for comparison. SPL$_{k\text{-agent(s)}}$ and SDTW$_{k\text{-agent(s)}}$ refers to wayfinding scores averaged over k VLN agents trained from random initialization.

We then employ the trained agents for the wayfinding task and report performance as either the SPL or SDTW similarity between the path taken by the agent and the reference path – using either a single agent or the average score from three agents.

Results Table 3 compares system-level and instance-level correlations for all metrics, both standard and model-based. At the *system-level*, we see no correlation between standard text metrics such as BLEU, ROUGE, METEOR and CIDEr and human wayfinder performance. The exception is SPICE, which shows the desired negative correlation with NE, and positive correlation with SR, SPL (see Figure 3) and Quality. At the system-level, the model-based approaches (BERTScore,

---

[4] tfhub.dev/google/bert_uncased_L-12_H-768_A-12/1

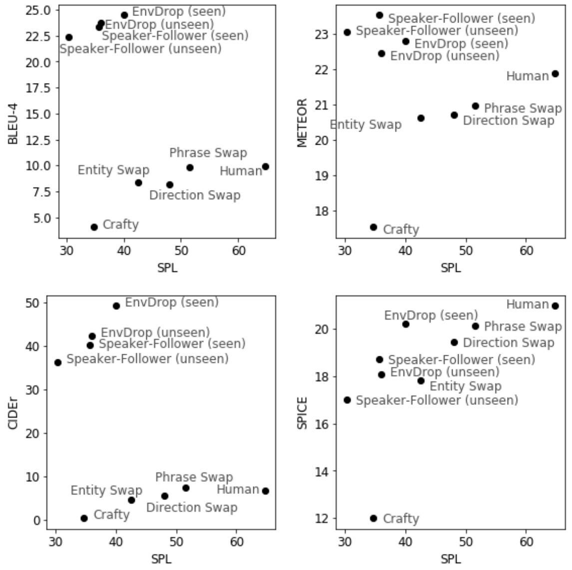


Figure 3: Standard evaluation metrics vs. human wayfinding outcomes (SPL) for 9 navigation instruction generation systems. SPICE is most consistent with human wayfinding outcomes, although no metrics score the Crafty template-based instructions highly.

agent SPL/SDTW and Compatability) also lack the desired correlation and exhibit wide confidence intervals. Here, it is important to point out that the 9 systems under evaluation include a variety of styles (e.g., Crafty's template-based instructions, different annotator pools, adversarial perturbations) which are dissimilar to the R2R data used to train the VLN agents and the compatibility model. Accordingly, the model-based approaches are unable to reliably rank these out-of-domain systems.

At the *instance-level* (when scoring individual instructions) we observe different outcomes. SPICE scores for individual instructions have high variance, and so SPICE does not correlate with wayfinder performance at the instruction level. In contrast, the model-based approaches exhibit the desired correlation, particularly when restricted to the model-generated instructions (Table 3 bottom panel). Our compatibility score shows the strongest correlation among all metrics, performing similarly to an ensemble of three VLN agents.

6 Conclusion

Generating grounded navigation instructions is one of the most promising directions for improving the performance of VLN wayfinding agents, and a challenging and important language generation task in its own right. In this paper, we show that efforts to improve navigation instruction generators have been hindered by a lack of suitable automatic evaluation metrics. With the exception of SPICE, all the standard textual evaluation metrics we evaluated (BLEU, CIDEr, METEOR and ROUGE) are ineffective, and – perhaps as a result – existing instruction generators have substantial headroom for improvement.

To address this problem, we develop an instruction-trajectory compatibility model that outperforms all existing automatic evaluation metrics on instance-level evaluation without needing any reference instructions – making it suitable for use as a reward function in a reinforcement learning setting, as a discriminator in a Generative Adversarial Network (GAN) (Dai et al., 2017), or for filtering instructions in a data augmentation setting.

Progress in natural language generation (NLG) is increasing the demand for evaluation metrics that can accurately evaluate generated text in a variety of domains. Our findings are a timely reminder that textual evaluation metrics should not be trusted in new domains unless they have been comprehensively validated against human judgments. In the case of grounded navigation instructions, for model selection in the presence of reference instructions we recommend using the SPICE metric. In all other scenarios (e.g., selecting individual instructions, or model selection without reference instructions) we recommend using a learned instruction-trajectory compatibility model.

Acknowledgements

We thank the Google Data Compute team, in particular Ashwin Kakarla and Priyanka Rachapally, for their tooling and annotation support for this project.

References

Sanyam Agarwal, Devi Parikh, Dhruv Batra, Peter Anderson, and Stefan Lee. 2019. Visual landmark selection for generating grounded and interpretable navigation instructions. In *CVPR workshop on Deep Learning for Semantic Visual Navigation*.

Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Roshan Zamir. 2018a. On evaluation of embodied navigation agents. *CoRR*, abs/1807.06757.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: Semantic Propositional Image Caption Evaluation. In *ECCV*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *ICML*.

Xinlei Chen, Tsung-Yi Lin Hao Fang, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv preprint arXiv:1504.00325*.

Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. 2017. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Markus Dräger and Alexander Koller. 2012. Generation of landmark-based navigation instructions from open-source data. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 757–766, Avignon, France. Association for Computational Linguistics.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *NeurIPS*.

Tsu-Jui Fu, Xin Eric Wang, Matthew Peterson, Scott Grafton, Miguel Eckstein, and William Yang Wang. 2019. Counterfactual vision-and-language navigation via adversarial path sampling.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. 2019. Transferable representation learning in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. 2019. Effective and general evaluation for instruction conditioned navigation using dynamic time warping. *NeurIPS Visually Grounded Interaction and Language Workshop*.

Vihan Jain, Gabriel Magalhães, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In *ACL*.

Da-Cheng Juan, Chun-Ta Lu, Zhen Li, Futang Peng, Aleksei Timofeev, Yi-Ting Chen, Yaxi Gao, Tom Duerig, Andrew Tomkins, and Sujith Ravi. 2020. Ultra fine-grained image semantic embedding. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, page 277–285.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Larry Lansing, Vihan Jain, Harsh Mehta, Haoshuo Huang, and Eugene Ie. 2019. VALAN: Vision and language agent navigation. *arXiv:1912.03241*.

Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah Smith, and Yejin Choi. 2019. Robust navigation with language pretraining and stochastic sampling. In *CVPR*.

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

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *CVPR*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2014. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *CVPR*.

Vivien Mast and Diedrich Wolter. 2013. A probabilistic framework for object descriptions in indoor route instructions. In *Spatial Information Theory*, pages 185–204, Cham. Springer International Publishing.

Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. 2020. Retouchdown: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view. *arXiv:2001.03671*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020. REVERIE: Remote embodied visual referring expression in real indoor environments. In *CVPR*.

Kai-Florian Richter and Alexander Klippel. 2005. A model for context-specific route directions. In *Spatial Cognition IV. Reasoning, Action, Interaction*, pages 58–78, Berlin, Heidelberg. Springer Berlin Heidelberg.

Michael Roth and Anette Frank. 2010. Computing EM-based alignments of routes and route directions as a basis for natural language generation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 958–966, Beijing, China. Coling 2010 Organizing Committee.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45:2673–2681.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2019. Vision-and-dialog navigation. In *CoRL*.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *CVPR*.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*.

Yinfei Yang, Gustavo Hernández Ábrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bidirectional dual encoder with additive margin softmax. In *IJCAI*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with BERT. In *ICLR*.

A Automated metric scores for all instructions

We provide more details about automated metric scores for all instructions in this section. Table 4 gives automated metrics for each model we consider. Generated instructions from EnvDrop and Speaker-Follower are scored the highest, whereas human instructions are scored poorly and on par with perturbed instructions, and Crafty is the lowest. These results diverge significantly from human wayfinding performance in Section 3, and highlights the inefficacy of these automated text metrics.

	BertScore	**BLEU-4**	**CIDEr**	**ROUGE**	**METEOR**	**SPICE**
Val-unseen						
Speaker Fol.	78.9	22.3	36.3	45.6	23.1	17.0
EnvDrop	79.3	23.7	42.2	45.8	22.5	18.1
Val-seen						
Speaker Fol.	79.0	23.3	40.1	46.3	**23.5**	18.7
EnvDrop	**79.5**	**24.5**	**49.3**	**46.7**	22.8	20.2
Crafty	71.5	4.1	0.4	23.3	17.5	12.0
Dir. Swap	74.7	8.2	5.7	31.4	20.7	19.4
Entity Swap	74.0	8.4	4.6	31.4	20.6	17.8
Phrase Swap	74.8	9.8	7.4	31.4	21.0	20.1
Human	74.9	9.9	6.7	33.3	21.9	**21.0**

Table 4: Automated metric scores for all instructions. The BLEU, CIDEr and ROUGE metrics score human instructions poorly compared to the neural net models.

B Crafty Details

We use the data in Matterport3D to build Crafty, a template-based navigation instruction generator that uses a Hidden Markov Model (HMM) to select objects as reference landmarks for wayfinding. Crafty's four main components (*Appraiser*, *Walker*, *Observer* and *Talker*) are described below.

B.1 Appraiser

The Appraiser scores the interestingness of objects based on the Matterport3D scans in the training set. It treats each panorama as a document and the categories corresponding to objects visible from the panorama as words, and then computes a per-category inverse document frequency (IDF) score.

B.2 Walker

The Walker converts a panorama sequence into a motion sequence. Given a path (sequence of connected panoramas) and an initial heading, it calculates the entry heading into each panorama and the exit heading required to transition to the next panorama. For each panorama, all annotated objects that are visible from the location are retrieved. For each object, we obtain properties such as their category and center, which allows the distance and heading from the panorama center to be computed. From these, the Walker creates a sequence of motion tuples, each of which captures the context of the source panorama and the goal panorama, along with the heading to move from source to goal.

B.3 Observer

The Observer selects an object sequence by generating objects from an HMM that is specially constructed for each environment, characterized by:

- *Emissions*: how panoramas relate to objects. This is a probability distribution over panoramas for each object, based on the distance between the object and the panoramas.
- *Transitions*: how looking at one object might shift to another one, based on their relative location, the motion at play, and the Appraiser's assessment of their prominence.

The intuition for using an HMM is that we tend to fixate on a given salient object over several steps as we move (high self-transitions); these tend to be nearby (high emission probability for objects near a panorama's center) and connected to the next salient object (biased object-object transitions). To explain a particular observed panorama sequence (path), we can then infer the optimal object sequence using the Viterbi algorithm.

B.4 Talker

Given a motions sequence from the Walker and corresponding object observations from the Observer, the Talker uses a small set of templates to create English instructions for each step. We decompose this into low-level and high-level templates.

B.4.1 Low-level templates

For single step actions, there are three main things to mention: movement, the fixated object and its relationship to the agent's position.

MOVE. For movement, we simply generate a set of possible commands for each direction type, where the direction types are defined as in the orientation wheel shown in Fig. 4. There are additional direction types for UP and DOWN based on relative pitch (e.g. when the goal panorama is higher or lower than the source).

Given one of these heading types, we generate a set of matching phrases appropriate to each. E.g.

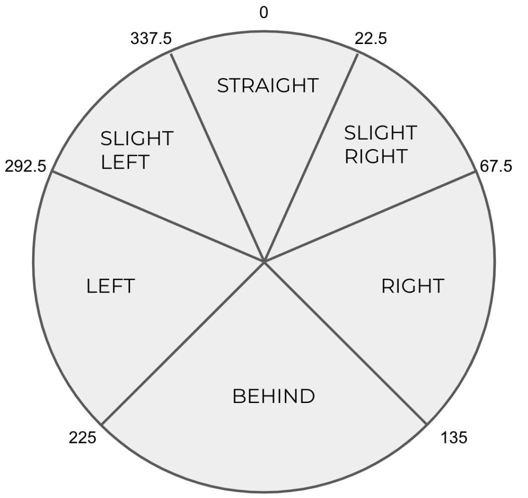


Figure 4: Orientation wheel with directions types. The demarcations are $\frac{\pi}{8}$ (22.5°), $\frac{3\pi}{8}$ (67.5°), etc.

for LEFT and RIGHT, the verbs *face*, *go*, *head*, *make a*, *pivot*, *turn*, and *walk* are combined *left* and *right*, respectively. For moving STRAIGHT, the verbs *continue*, *go*, *head*, *proceed*, *walk*, and *travel* are combined with *straight* and *ahead*. To select an instruction for a given heading, we randomly sample from the list of available phrases. To generate a MOVE command for LEFT, we randomly sample from [*face left*, *go left*, . . . *walk left*].

OBJ. An object's description is its category (e.g. *couch*, *tv*, *window*).

ORIENT. We use the same direction types shown in Figure 4. When an object is STRAIGHT and BEHIND, we use the phrases *ahead of you* or *in front of you* and *behind you* or *in back of you*, respectively. For objects to the LEFT or RIGHT, we use two templates `DIRECTION_PRE DIRECTION` and `DIRECTION DIRECTION_POST`, where `DIRECTION_PRE` is selected from [*to your*, *to the*, *on your*, *on the*] and `DIRECTION_POST` is the phrase *of you*. This produces *to your left*, *on the right*, *right of you*, and so on. For SLIGHT LEFT and SLIGHT RIGHT, one of [*a bit*, *slightly*, *a little*, *just*] is added in front (e.g. *a bit to your left*).

B.4.2 High level templates

Crafty pieces these low-level textual building blocks together to describe actions. In what follows, MOVE, OBJ, and ORIENT indicate the move command, object phrase and orientation phrase, respectively, discussed above.

Single action. We use templates for three situations: start of a path, heading change in a panorama (intra) and moving between panoramas (inter).

- *Start of path*: There are several templates that simply help a wayfinder verify their current position. Ex: *you are near a* `OBJ`*,* `ORIENT`*.*
- *Intra*: These templates include the movement command followed by a verification of the orientation to an object having completed the movement. Ex: `MOVE`*. a* `OBJ` *is* `ORIENT`*.*
- *Inter* : These templates capture walking from one panorama to another and provide additional object verification. Ex: `MOVE`*, going along to the* `OBJ ORIENT`*.*

Multi-step actions. We attempt to reduce verbosity by collapsing actions that involve fixation on the same object.

- *Combining actions*: Repeated actions are collapsed; e.g. [STRAIGHT, STRAIGHT, RIGHT, STRAIGHT] becomes [STRAIGHT, RIGHT, STRAIGHT]). These produce a composite move command, e.g. *proceed forward and make a right and go straight*.
- *Describing the object*: To orient with respect to the fixated-upon object, we switch on the direction type between the agent and the object at the last action. Ex: for STRAIGHT, we use *heading toward the* `OBJ` and for SLIGHT LEFT/RIGHT, we use *approaching the* `OBJ ORIENT`*.*

The final output is the concatenation of the combined move command and the object orientation phrase.

End-of-path instruction templates. The final action is a special situation in that it needs to describing stopping near a salient object. For this, we extract MOVE and OBJ phrases from the last action and use templates such as `MOVE` *and stop by the* `OBJ`*.*

Full example. Putting it all together, Crafty creates full path instructions such as the following, with relevant high-level templates indicated:

- (START) *there is a lamp when you look a bit to the left. pivot right, so it is in back of you.*
- (INTER) *walk forward, going along to the curtain in front of you.*

- (INTRA) *curve left. you should see a tv ahead of you.*
- (MULTI-ACTION) *go forward and go slightly left and walk straight, passing the curtain to your right.*
- (END-OF-PATH) *continue forward and stop by the couch.*

Crafty's instructions are more verbose than human instructions, but are often easy to follow—provided there are good, visually salient landmarks in the environment to use for orientation.

C Human Rater Performance Over Time

Human raters are excellent at learning and adapting to new problems over time. To understand whether our 37 human raters learn to self-correct the perturbed instructions over time and whether that affects the quality of our human wayfinding results, we investigate rater performance as a function of time using the sequence of examples they evaluate.

Figure 5 shows the average human rater performance for all of the 9 datasets included in Table 1 of Section 3. Due to the binary nature of SR, we use a 50-point bin to average each rater's performance, and then average the results across all raters for each bin. Figure 5 shows that the average rater performance stays flat within the uncertainties and does not show systematic drift over time, indicating no overall self-correction that affects the wayfinding results. For a more granular scrutiny of individual perturbation methods, in particular the perturbed instructions, we plot in Figure 6 the average human rater performance over time for the three methods: *Direction Swap*, *Entity Swap*, and *Phrase Swap*. Despite greater uncertainties due to much fewer data points used for averaging, the overall human performance for each method still does not drift significantly in a systematic manner. These results indicate that our human wayfinding performance results are reliable and robust over time, which can be attributed to shuffling of the examples and that the perturbation methods are blind to human raters.

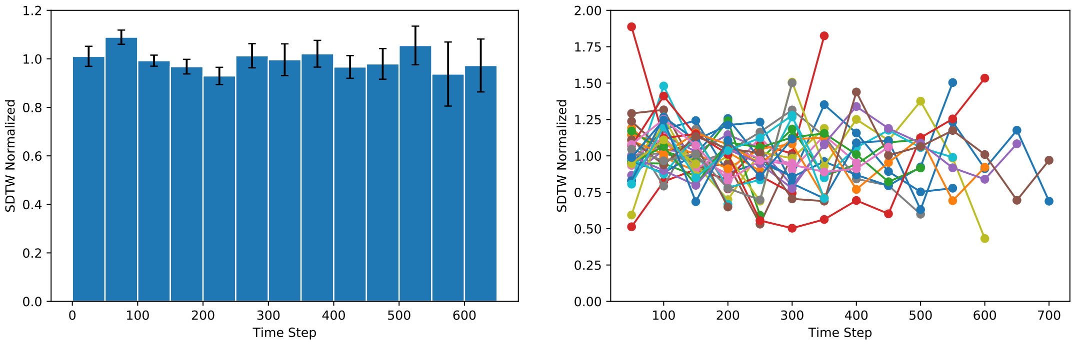


Figure 5: Average human rater performance for all datasets as a function of time using the sequence of examples each rater has evaluated. Each rated example indicates a time step. We normalize the scores of each rater by their mean value over time to remove performance bias of each rater in order to better pick up the trend over time. We average 50 examples to get the mean SDTW for each rater due to the discrete nature of success. *Left*: The mean performance of all rater for each bin. Error bars represent the standard deviation of the mean. *Right*: Individual rater performance over time. Each line represents a single rater. Despite a few outliers, the overall human rater performance is flat and consistent over time, indicating no self-correction or adaptation to the datasets by human raters.

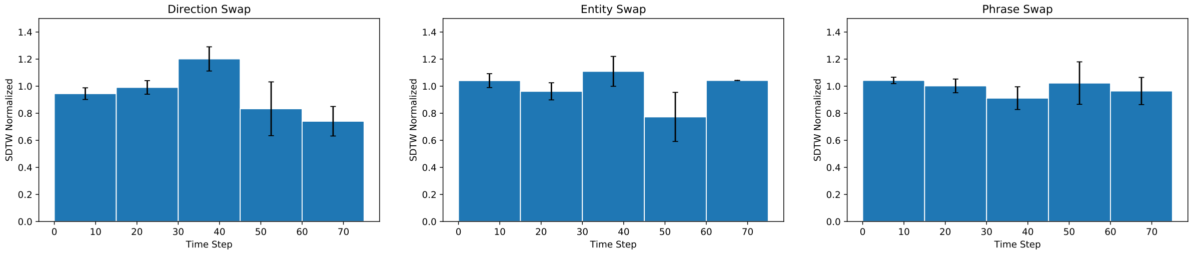


Figure 6: Average human rater performance for the three instruction perturbation methods as a function of time (number of examples), computed in a similar way as in Figure 5. We use a 15-point bin to compute the average for each human rater, and aggregate over all raters to get the mean and its uncertainty. The overall human rater performance stays flat and does not drift significantly over time for instruction perturbations.

Cross-lingual Visual Pre-training for Multimodal Machine Translation

Ozan Caglayan[1], **Menekse Kuyu**[2], **Mustafa Sercan Amac**[2], **Pranava Madhyastha**[1]
Erkut Erdem[2], **Aykut Erdem**[3] and **Lucia Specia**[1,4,5]
Imperial College London[1], Hacettepe University[2], Koç University[3]
University of Sheffield[4], ADAPT - Dublin City University[5]
`o.caglayan@ic.ac.uk, meneksekuyu@gmail.com, sercanamac@gmail.com, pranava@ic.ac.uk`
`erkut@cs.hacettepe.edu.tr, aerdem@ku.edu.tr, l.specia@ic.ac.uk`

Abstract

Pre-trained language models have been shown to improve performance in many natural language tasks substantially. Although the early focus of such models was single language pre-training, recent advances have resulted in cross-lingual and visual pre-training methods. In this paper, we combine these two approaches to learn visually-grounded cross-lingual representations. Specifically, we extend the translation language modelling (Lample and Conneau, 2019) with masked region classification and perform pre-training with three-way parallel vision & language corpora. We show that when fine-tuned for multimodal machine translation, these models obtain state-of-the-art performance. We also provide qualitative insights into the usefulness of the learned grounded representations.

1 Introduction

Pre-trained language models (Peters et al., 2018; Devlin et al., 2019) have been proven valuable tools for contextual representation extraction. Many studies have shown their effectiveness in discovering linguistic structures (Tenney et al., 2019), which is useful for a wide variety of NLP tasks (Talmor et al., 2019; Kondratyuk and Straka, 2019; Petroni et al., 2019). These positive results led to further exploration of (i) cross-lingual pre-training (Lample and Conneau, 2019; Conneau et al., 2020; Wang et al., 2020) through the use of multiple mono-lingual and parallel resources, and (ii) visual pre-training where large-scale image captioning corpora are used to induce grounded vision & language representations (Lu et al., 2019; Tan and Bansal, 2019; Li et al., 2020a; Su et al., 2020; Li et al., 2020b). The latter is usually achieved by extending the masked language modelling (MLM) objective (Devlin et al., 2019) with auxiliary vision & language tasks such as masked region classification and image sentence matching.

In this paper, we present the first attempt to bring together cross-lingual and visual pre-training. Our visual translation language modelling (VTLM) objective combines the translation language modelling (TLM) (Lample and Conneau, 2019) with masked region classification (MRC) (Chen et al., 2020; Su et al., 2020) to learn grounded cross-lingual representations. Unlike most of the prior work that use classification or retrieval based downstream evaluation, we focus on the *generative* task of multimodal machine translation (MMT), where images accompany captions during translation (Sulubacak et al., 2020). Once pre-trained, we transfer the VTLM encoder to a Transformer-based (Vaswani et al., 2017) MMT and fine-tune it for the MMT task. To our knowledge, this is also the first attempt of pre-training & fine-tuning for MMT, where the current state of the art mostly relies on training multimodal sequence-to-sequence systems from scratch (Calixto et al., 2016; Caglayan et al., 2016; Libovický and Helcl, 2017; Elliott and Kádár, 2017; Caglayan et al., 2017; Yin et al., 2020).

Our findings highlight the effectiveness of cross-lingual visual pre-training: when fine-tuned on the English→German direction of the Multi30k dataset (Elliott et al., 2016), our MMT model surpasses our constrained MMT baseline by about 10 BLEU and 8 METEOR points. The rest of the paper is organised as follows: §2 describes our pre-training and fine-tuning protocol, §3 presents our quantitative and qualitative analyses, and §4 concludes the paper with pointers for future work.

2 Method

We propose Visual Translation Language Modelling (VTLM) objective to learn multimodal cross-lingual representations. In what follows, we first describe the TLM objective (Lample and Conneau, 2019) and then introduce the modifications required to extend it to VTLM.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1317–1324
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

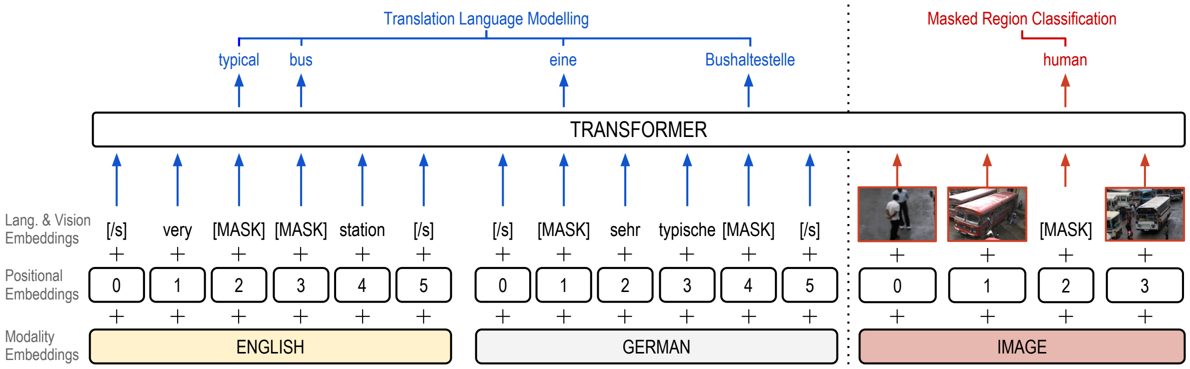


Figure 1: The architecture of the proposed model: VTLM extends the TLM (Lample and Conneau, 2019) (left side of the dotted line) with regional image features. Masking applies on both linguistic and visual tokens.

2.1 Translation language modelling

The TLM objective is based on Transformer networks and assumes the availability of parallel corpora during training. It defines the input x as the concatenation of m-length source language sentence $s^{(1)}_{1:m}$ and n-length target language sentence $s^{(2)}_{1:n}$:

$$x = \left[s^{(1)}_1, \cdots, s^{(1)}_m, s^{(2)}_1, \cdots, s^{(2)}_n\right]$$

For a given input, TLM follows (Devlin et al., 2019), and selects a random set of input tokens $y = \{s^{(l)}_1, \dots, s^{(l)}_k\}$ for masking. Let us denote the masked input sequence with $\tilde{x}$, and the ground-truth targets for masked positions with $\hat{y}$. TLM employs the masked language modelling (MLM) objective to maximise the log-probability of correct labels $\hat{y}$, conditioned on the masked input $\tilde{x}$:

$$\mathcal{L} = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \log \Pr(\hat{y} | \tilde{x}; \theta)$$

where θ are the model parameters. We keep the standard hyper-parameters for masking, i.e. 15% of inputs are randomly selected for masking, from which 80% are replaced with the `[MASK]` token, 10% are replaced with random tokens from the vocabulary, and 10% are left intact.

2.2 Visual translation language modelling

VTLM extends the TLM by adding the visual modality alongside the translation pairs (Figure 1). Therefore, we assume the availability of sentence pair & image triplets and redefine the input as:

$$x = \left[s^{(1)}_1, \cdots, s^{(1)}_m, s^{(2)}_1, \cdots, s^{(2)}_n, v_1, \cdots, v_o\right]$$

where $\{v_1, \cdots, v_o\}$ are features extracted from a Faster R-CNN model (Ren et al., 2015) pre-trained on the Open Images dataset (Kuznetsova et al., 2018).[1] Specifically, we extract convolutional feature maps from $o = 36$ most confident regions, and average pool each of them to obtain a region-specific feature vector $v_i \in \mathbb{R}^{1536}$. Each region i is also associated with a detection label $\hat{v}_i$ provided by the extractor. Before encoding, the feature vectors and their bounding box coordinates are projected into the language embedding space.

The final model processes translation pairs and projected region features in a **single-stream** fashion (Su et al., 2020; Li et al., 2020a), and combines the TLM loss with the masked region classification (MRC) loss as follows:

$$\mathcal{L} = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \log \Pr(\{\hat{y}, \hat{v}\} | \tilde{x}; \theta)$$

Masking. 15% random masking ratio is applied separately to both language and visual streams, and the $\hat{v}$ above now denotes the correct region labels for the masked feature positions. Different from previous work that zeroes out masked regions (Tan and Bansal, 2019; Su et al., 2020), VTLM replaces their projected feature vectors with the `[MASK]` token embedding.[2] Similar to textual masking, 10% of the random masking amounts to using regional features randomly sampled from all images in the batch, and the remaining 10% of regions are left intact.

2.3 Pre-training

VTLM requires a three-way parallel multimodal corpus, which does not exist in large-scale. To ad-

---

[1] The "*faster rcnn inception resnet v2 atrous oid v4*" model from TensorFlow.

[2] Although this choice is mostly practical, we hypothesise that using the same signal for both language and visual masking can be beneficial for grounding.

dress this, we extend[3] the **Conceptual Captions** (CC) (Sharma et al., 2018) dataset with German translations. CC is a large-scale collection of ∼3.3M images retrieved from the Internet, with noisy *alt-text* captions in English. The translation of English captions into German was automatically performed using an existing NMT model (Ng et al., 2019) provided[4] in the Fairseq (Ott et al., 2019) toolkit. Since some of the images are no longer accessible, the final corpus' size is reduced to ∼3.1M triplets. We used byte pair encoding (BPE) (Sennrich et al., 2016) to learn a joint 50k BPE model on the CC dataset. The pre-training was conducted for 1.5M steps, using a single RTX2080-Ti GPU, and best checkpoints were selected with respect to validation set accuracy.

Settings. We use a small version of the TLM (Lample and Conneau, 2019)[5] and set the model dimension, feed-forward layer dimension, number of layers and number of attention heads to $d = 512$, $f = 2048$, $l = 6$ and $h = 8$, respectively. We randomly initialise model parameters, instead of using pre-trained LM checkpoints such as BERT or XLM. We use Adam (Kingma and Ba, 2014) with the mini-batch size and the learning rate set to 64 and 0.0001, respectively. The dropout (Srivastava et al., 2014) rate is set to 0.1 in all layers. The pre-training is done for 1.5M steps using a single RTX2080-Ti GPU, and best checkpoints are selected with respect to validation accuracy.

2.4 Baseline MT models and fine-tuning

Our experimental protocol consists of **initialising** the encoder and the decoder of Transformer-based NMT and MMT models with weights from TLM/VTLM, and **fine-tuning** them with a smaller learning rate. The architectural difference between the NMT and the MMT models is that the latter encodes 36 regional visual features as part of the source sequence, similar to the VTLM (§ 2.2). As a natural baseline, we train constrained (trained only on the MT dataset) models without transferring weights from the pre-trained TLM/VTLM models. We refer to these models as **from-scratch**. For the fine-tuning experiments, we train three runs with different seeds. For evaluation, we use the models with the lowest validation set perplexity to decode translations with beam size equal to 8.

Dataset. We use the standard MMT corpus **Multi30k** (Elliott et al., 2016) for both fine-tuning and from-scratch runs. It contains 30k image descriptions from Flickr30k (Young et al., 2014) and their human translations in German for training, along with three test sets of 1K samples each: the original and the most in-domain **2016** test set, as well as **2017** and **COCO** test sets created using images and descriptions collected from sources other than Flickr.

Settings. For fine-tuning, we use the same hyperparameters as the pre-training phase, apart from decreasing the learning rate to $1e{-}5$. For MT models that are trained from scratch, we increase the dropout rate to 0.4 and linearly warm up the learning rate from $1e{-}7$ to $1e{-}4$ during the first 4,000 iterations. *Inverse square-root* annealing is applied after 4,000 iterations.

3 Results

3.1 Machine translation

Table 1 reports METEOR and BLEU scores across three different test sets of Multi30k. First, we observe that the MMT system trained from scratch is consistently worse than its NMT counterpart. However, the gap disappears when pre-trained TLM/VTLM checkpoints are fine-tuned for MT. This suggests that pre-training may be necessary for *single-stream* multimodal encoding, where the number of regions (36) outnumbers the avg. number of source tokens (13 for Multi30k).

Second, we see that the best performances are obtained when models are first pre-trained on the *three-way* parallel Conceptual Captions (CC) dataset. To validate this further, we train a baseline NMT on the concatenation of Multi30k and CC (NMT+CC) and an MMT that uses only Multi30k for **both** pre-training and fine-tuning. The results clearly show that these systems lag behind the ones pre-trained on CC.

We also experimented with an alternative pre-training strategy where we do not mask visual regions. Interestingly, this alternative MMT in Table 1 reveals that not masking visual regions during pre-training yields slightly better results overall. This is equivalent to letting the model predict the object labels from a multimodal input where words are stochastically masked but regional features are kept intact. Overall, MMT fine-tuning on VTLM sets a new state of the art across all Multi30k test

---

[3] `https://hucvl.github.io/VTLM`

[4] The `transformer.wmt19.en-de` model.

[5] https://github.com/facebookresearch/XLM

	2016		2017		COCO	
	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU
Best RNN-MMT (Caglayan, 2019)						
	58.7	39.4	52.9	32.6	–	–
Graph-based Transformers MMT (Yin et al., 2020)						
	57.6	39.8	51.9	32.2	37.6	28.7
Ensemble RNN-MMT (Delbrouck and Dupont, 2018)						
	59.6	40.3	–	–	–	–
Unconstrained Transformers MMT (Helcl et al., 2018)						
	59.1	42.7	–	–	–	–
Our Baseline Transformers **(from scratch)**						
`NMT`	56.4	37.6	51.3	30.9	47.2	27.5
`+CC`	58.8	39.5	55.6	36.2	51.5	33.0
`MMT`	55.4	35.2	49.5	27.7	46.2	25.4
VTLM: Pre-train and fine-tune on **Multi30k**						
`MMT`	59.0	40.2	53.5	32.7	49.3	28.9
TLM: Pre-train on **CC** – fine-tune on **Multi30k**						
`NMT`	60.7	43.1	56.5	37.6	53.3	34.8
	60.5 ± 0.21	42.5 ± 0.46	56.4 ± 0.10	37.3 ± 0.38	53.1 ± 0.13	34.6 ± 0.17
`MMT`	60.3	41.9	56.7	37.6	53.3	34.3
	60.2 ± 0.08	41.7 ± 0.18	56.5 ± 0.16	37.5 ± 0.10	53.0 ± 0.20	34.1 ± 0.14
VTLM: Pre-train on **CC** – fine-tune on **Multi30k**						
`NMT`	61.2	43.3	56.9	37.2	53.7	35.1
	60.5 ± 0.46	42.5 ± 0.53	56.4 ± 0.34	37.0 ± 0.16	53.1 ± 0.42	34.6 ± 0.40
`MMT`	60.8	42.7	57.1	**38.1**	53.1	34.2
	60.6 ± 0.15	42.6 ± 0.14	56.9 ± 0.20	37.7 ± 0.43	53.0 ± 0.05	33.9 ± 0.19
VTLM: Alternative (0% visual masking during pre-training)						
`MMT`	**61.3**	**44.0**	**57.2**	38.0	**53.8**	**35.2**
	60.9 ± 0.30	43.3 ± 0.59	57.1 ± 0.07	37.6 ± 0.31	53.6 ± 0.17	35.1 ± 0.09

Table 1: Quantitative comparison of experiments: when the mean and the standard deviation is reported, the single numbers appearing above, denote the maximum across three different runs.

sets.[6] We leave the exploration of visual region masking for the MRC task as future work and proceed with the alternative variant in the following experiments.

Encoder attention parameters. When fine-tuning the TLM for MT, the default XLM implementation randomly initialises the decoder's missing encoder attention parameters. In our experiments, we noticed that **copying** those parameters from the TLM self-attention layers substantially improves the results up to 2.2 BLEU.

3.2 Explicit masking

Here, we will evaluate the extent to which the visual information is taken into account (i) when TLM/VTLM predicts masked tokens, and (ii) when the fine-tuned NMT and MMT models are forced to translate source sentences with missing visual entities. For the latter, we use Flickr30k entities (Plummer et al., 2015) to mask head nouns in 2016 test set sentences, similar to Caglayan et al. (2019).

Last-word masking. In this experiment, we measure the target word prediction accuracy, when last tokens[7] of input caption pairs are systematically masked during evaluation. Table 2 suggests

---

[6] We exclude Grönroos et al. (2018) as their improvements (45.5 BLEU) were not due to multi-modality but rather to other modifications such as heavy parallel data augmentation, domain fine-tuning, and ensembling.

[7] We pre-process the sentences to ensure that they do not end with punctuation marks, which would make the task easier for masked punctuation.

	Valid			Test		
	En	De	Both	En	De	Both
`TLM`	89.0	87.3	55.2	88.5	86.3	53.6
`VTLM`	⇑ **0.9**	⇑ **1.4**	⇑ **5.0**	⇑ **1.1**	⇑ **2.2**	⇑ **5.8**
`+shuf`	⇓ **1.0**	⇓ **0.2**	⇓ **7.7**	⇓ **1.3**	⇓ **0.3**	⇓ **7.4**

Table 2: Masked *last-word* prediction accuracies: VTLM gains are with respect to TLM, whereas the incongruent (+shuf) drops are relative to VTLM.

	Mask	Remove
`TLM→NMT`	31.44	27.38
`TLM→MMT`	⇓ **0.43**	⇓ **0.26**
`VTLM→NMT`	31.27	27.63
`VTLM→MMT`	⇑ **1.65**	⇑ **0.65**

Table 3: Entity masking on 2016 test set: results are BLEU averages of three fine-tuned MT systems.

that the visual information is much more helpful (i.e. up to 6% accuracy improvement) when last tokens are masked in both English and German captions. However, if one caption is available, it provides enough context for cross-lingual prediction. Finally, when we shuffle (+shuf) the test set features to introduce incongruence (Elliott, 2018), we see that the VTLM model deteriorates substantially. This confirms that the accuracy improvements are not due to side-effects of experimentation noise, such as regularisation or random seed related effects.

Entity masking in MT. We devise two ways of masking entities i.e. we either replace them with the `[MASK]` token or remove them entirely so that the masking phenomena is *not known* to the model. The results in Table 3 show that MMT models can recover the missing source context to some extent, only when they are pre-trained using the proposed VTLM objective. In other words, the grounding ability can only be acquired when visual modality is present for both pre-training and fine-tuning. The gap between Mask and Remove also seems to highlight the importance of reserving a source position even it is corrupted/masked.

3.3 Visual attention in MMT

Here we take the MMT decoder's cross-attention layers and measure the attention mass they attribute to regional features in the input embeddings. Although the encoder's self-attention layers produce increasingly mixed contextual embeddings as we move towards the top layers, Brunner et al. (2020) show that the final layer states still encode corresponding input embeddings to some extent. With this assumption at hand, Figure 2 shows the average attention mass attributed to the first 36 (visual) top-layer encoding states, by each cross-attention layer in the decoder. We find these results to be in *agreement* with the quantitative metrics (Table 1), with VTLM-MMT assigning substantially more attention to these positions, compared to TLM-MMT and MMT from scratch.

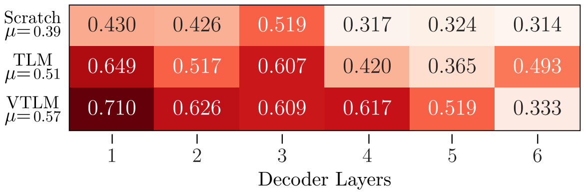


Figure 2: Cross-attention mass over the visual portion of input sequences, averaged across the 2016 test set.

4 Conclusions

We proposed a novel cross-lingual visual pre-training approach and tested its efficacy for multimodal machine translation. Our pre-training approach extends the TLM framework (Lample and Conneau, 2019) with regional features and performs masked language modelling and masked region classification on a three-way parallel corpus. We show that this leads to substantial improvements compared to multimodal machine translation with cross-lingual pre-training only or without pre-training at all. As future work, we consider exploring *more informed* masking strategies for visual regions and investigating the impact of visual masking probability for the MRC pre-training task for downstream MMT performance.

Acknowledgments

This work was supported in part by TUBA GEBIP fellowship awarded to Erkut Erdem, and the MMVC project funded by TUBITAK and the British Council via the Newton Fund Institutional Links grant programme (grant ID 219E054 and 352343575). Lucia Specia, Pranava Madhyastha and Ozan Caglayan also received support from MultiMT project (H2020 ERC Starting Grant No. 678017) and Lucia Specia from the Air Force Office of Scientific Research (under award number FA8655-20-1-7006).

References

Gino Brunner, Yang Liu, Damian Pascual Ortiz, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On Identifiability in Transformers. In *ICLR*.

Ozan Caglayan. 2019. *Multimodal Machine Translation*. Theses, Université du Maine.

Ozan Caglayan, Walid Aransa, Adrien Bardet, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, Marc Masana, Luis Herranz, and Joost van de Weijer. 2017. LIUM-CVC submissions for WMT17 multimodal translation task. In *Proceedings of the Second Conference on Machine Translation*, pages 432–439, Copenhagen, Denmark. Association for Computational Linguistics.

Ozan Caglayan, Walid Aransa, Yaxing Wang, Marc Masana, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, and Joost van de Weijer. 2016. Does multimodality help human and machine for translation and image captioning? In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 627–633, Berlin, Germany. Association for Computational Linguistics.

Ozan Caglayan, Pranava Madhyastha, Lucia Specia, and Loïc Barrault. 2019. Probing the need for visual context in multimodal machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4159–4170, Minneapolis, Minnesota. Association for Computational Linguistics.

Iacer Calixto, Desmond Elliott, and Stella Frank. 2016. DCU-UvA multimodal MT system report. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 634–638, Berlin, Germany. Association for Computational Linguistics.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *Computer Vision – ECCV 2020*, pages 104–120, Cham. Springer International Publishing.

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

Jean-Benoit Delbrouck and Stéphane Dupont. 2018. UMONS Submission for WMT18 Multimodal Translation Task. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 643–647, Belgium, Brussels. Association for Computational Linguistics.

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

Desmond Elliott. 2018. Adversarial evaluation of multimodal machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2974–2978, Brussels, Belgium. Association for Computational Linguistics.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.

Desmond Elliott and Ákos Kádár. 2017. Imagination improves multimodal translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 130–141, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Stig-Arne Grönroos, Benoit Huet, Mikko Kurimo, Jorma Laaksonen, Bernard Merialdo, Phu Pham, Mats Sjöberg, Umut Sulubacak, Jörg Tiedemann, Raphael Troncy, and Raúl Vázquez. 2018. The MeMAD submission to the WMT18 multimodal translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 603–611, Belgium, Brussels. Association for Computational Linguistics.

Jindřich Helcl, Jindřich Libovický, and Dušan Variš. 2018. CUNI system for the WMT18 multimodal translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 616–623, Belgium, Brussels. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. 2018. The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale. *CoRR*, abs/1811.00982.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020a. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11336–11344. AAAI Press.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020b. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In *Computer Vision – ECCV 2020*, pages 121–137, Cham. Springer International Publishing.

Jindřich Libovický and Jindřich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 196–202, Vancouver, Canada. Association for Computational Linguistics.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *2015 IEEE International Conference on Computer Vision (ICCV)*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Umut Sulubacak, Ozan Caglayan, Stig-Arne Grönroos, Aku Rouhe, Desmond Elliott, Lucia Specia, and Jörg Tiedemann. 2020. Multimodal machine translation through visuals and speech. *Machine Translation*, pages 1–51.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*.

Yongjing Yin, Fandong Meng, Jinsong Su, Chulun Zhou, Zhengyuan Yang, Jie Zhou, and Jiebo Luo. 2020. A novel graph-based multi-modal fusion encoder for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3035, Online. Association for Computational Linguistics.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Memorization vs. Generalization: Quantifying Data Leakage in NLP Performance Evaluation

Aparna Elangovan[1], **Jiayuan He**[1,2], **and Karin Verspoor**[1]
[1]The University of Melbourne, Australia
[2]RMIT University, Australia
`aparnae@student.unimelb.edu.au`
`{estrid.he, karin.verspoor}@unimelb.edu.au`

Abstract

Public datasets are often used to evaluate the efficacy and generalizability of state-of-the-art methods for many tasks in natural language processing (NLP). However, the presence of overlap between the train and test datasets can lead to inflated results, inadvertently evaluating the model's ability to memorize and interpreting it as the ability to generalize. In addition, such data sets may not provide an effective indicator of the performance of these methods in real world scenarios. We identify leakage of training data into test data on several publicly available datasets used to evaluate NLP tasks, including named entity recognition and relation extraction, and study them to assess the impact of that leakage on the model's ability to memorize versus generalize.

1 Introduction

Shared tasks that provide publicly available datasets in order to evaluate and compare the performance of different methods on the same task and data are common in NLP. Held-out test sets are typically provided, enabling assessment of the generalizability of different methods to previously unseen data. These datasets have played a key role in driving progress in NLP, by defining focus tasks and by making annotated data available to the broader community, in particular in specialized domains such as biomedicine where data can be difficult to obtain, and quality data annotations require the detailed work of domain experts. Examples of tasks where benchmark data sets exist include open domain question answering (QA) (Berant et al., 2013; Joshi et al., 2017) and biomedical named entity recognition (Smith et al., 2008) .

In the context of machine learning models, effectiveness is typically determined by the model's ability to both memorize and generalize (Chatterjee, 2018). A model that has huge capacity to memorize will often work well in real world applications, particularly where large amounts of training data are available (Daelemans et al., 2005). The ability of a model to generalize relates to how well the model performs when it is applied on data that may be different from the data used to train the model, in terms of e.g. the distribution of vocabulary or other relevant vocabulary. The ability to memorize, taken to the extreme, can be considered equivalent to an exact match lookup table (Chatterjee, 2018) and the ability to generalize captures how well it can deal with degrees of variations from the lookup table. An effective combination of memorization and generalization can be achieved where a model selectively memorizes only those aspects or features that matter in solving a target objective given an input, allowing it to generalize better and to be less susceptible to noise.

When there is considerable overlap in the training and test data for a task, models that memorize more effectively than they generalize may benefit from the structure of the evaluation data, with their performance inflated relative to models that are more robust in generalization. However, such models may make poor quality predictions outside of the shared task setting. The *external validity* of these evaluations can therefore be questioned (Ferro et al., 2018).

In this paper, we assess the overlap between the train and test data in publicly available datasets for Named Entity Recognition (NER), Relation Extraction (REL) and Text Classification (CLS) tasks, including SST2 (Socher et al., 2013), BioCreative (Smith et al., 2008; Arighi et al., 2011) and AIMed (Bunescu et al., 2005) datasets, and examine the significant impact of not taking into account this overlap on performance evaluation.

We argue that robustness in generalization to unseen data is a key consideration of the performance of a model, and propose a framework to examine

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1325–1335
April 19 - 23, 2021.

inadvertent leakage of data between data set splits, in order to enable more controlled assessment of the memorization vs. generalization characteristics of different methods.

2 Related work

The issue of memorization vs. generalization has been previously discussed in the context of question answering datasets, where, given only a question, a system must output the best answer it can find in available texts.

Lewis et al. (2020) identify 3 distinct issues for open domain QA evaluation: a) *question memorization* – recall the answer to a question that the model has seen at training time; b) *answer memorization* – answer novel questions at test time, where the model has seen the answer during training; and c) *generalization* – question and answer not seen during training time. They find that 58-71% of test answers occur in the training data in 3 examined data sets, concluding that the majority of the test data does not assess answer generalization. They also find that 28-34% have paraphrased questions in training data, and a majority of questions are duplicates differing only by a few words.

Similarly, Min (2020) identified repeating forms in QA test sets as a problem. The work proposed a novel template-based approach to splitting questions into paraphrase groups referred to as "Templates" and then controlling train/test data splits to ensure that all questions conforming to a given template appear in one segment of the data only. This was tested on the EMR Clinical Question Answering dataset emrQA (Pampari et al., 2018) and the Overnight dataset (Wang et al., 2015); it was demonstrated that models perform significantly worse on test sets where strict division is enforced. This paraphrase-based splitting methodology was also employed in their recent work on emrQA (Rawat et al., 2020).

3 Approach

A common practice to create a train and test set is to shuffle data instances in a dataset and generate random splits, without taking into account broader context. However, this can inadvertently lead to data leakage from the train set to test set due to the overlaps between similar train and test instances.

The type of overlap between train and test dataset depends on the type of the NLP task. Generally speaking, the leakage can occur either in the input texts or the annotated outputs. We define the types of overlaps which may occur in several NLP tasks as follows.

Algorithm 1 Compute overlap

```
procedure COMPARE($testset, trainset$)
    $totalscore \leftarrow 0$
    $n \leftarrow |testset|$
    for $test_i$ in testset do
        $s \leftarrow$ BESTMATCH($test_i, trainset$)
        $totalscore \leftarrow totalscore + s$
    end for
    return $totalscore/n$                    ▷ Average score
end procedure
procedure BESTMATCH($test_i, trainset$)
    $bestscore \leftarrow 0$
    for $train_j$ in trainset do
        $s \leftarrow$ SIMILARITY($test_i, train_j$)
        if $score > bestscore$ then
            $bestscore \leftarrow s$
        end if
    end for
    return $bestscore$
end procedure
```

- In text classification (CLS) tasks such as sentiment analysis, overall (document-level) similarity in input texts can result in train/test leakage.
- In named entity recognition (NER) tasks, leakage from train to test data may occur when
  a) input sentences or passages are similar
  b) target entities are similar
- In relation extraction (REL) tasks, leakage may occur when
  a) input sentences or passages are similar
  b) participating entities are similar

We propose a framework for quantifying train-test overlaps, and conduct experiments to show the impact of train-test overlap on model performances. Next, we discuss the proposed framework in Sec. 4.2 and the experimental settings in Sec. 4.3. We present our findings including the train-test overlaps in several benchmark datasets in Sec. 5.1 and the impact of data leakage in Sec. 5.2.

4 Method

4.1 Datasets

We examine overlap in the following datasets:

- **AIMed** - AIMed dataset (Bunescu et al., 2005) for protein relation extraction (REL)
- **BC2GM** - BioCreative II gene mention dataset (Smith et al., 2008) for NER task
- **ChEMU** - Chemical Reactions from Patents (He et al., 2020) for recognising names of chemicals, an NER task

Task	Dataset	Score	Split	Example
REL	AIMed (R)	100.0	Train	Thus, during PROTEIN1 -mediated suppression of cell proliferation, PROTEIN and PROTEIN2 may be important for coordinating cell-cycle progression, DNA replication and repair of damaged DNA.
			Test	Thus, during PROTEIN -mediated suppression of cell proliferation, PROTEIN1 and PROTEIN2 may be important for coordinating cell-cycle progression, DNA replication and repair of damaged DNA.
NER	BC2GM	100.0	Train	E2F family members
			Test	E2F family members (1-5)
CLS	SST2	100.0	Train	good movie .
			Test	it 's still not a good movie.
CLS	SST2	21.8	Train	herzog is obviously looking for a moral to his fable , but the notion that a strong , unified showing among germany and eastern european jews might have changed 20th-century history is undermined by ahola 's inadequate performance .
			Test	of the unsung heroes of 20th century

Table 1: Examples of train-test matches and the corresponding unigram similarity score.

- **BC3ACT** - Biocreative III protein interaction classification (CLS) (Arighi et al., 2011)
- **SST2** - Stanford Sentiment Analysis Treebank (Socher et al., 2013) used to classify sentiments (CLS) in Glue (Wang et al., 2018)

The AIMed dataset does not explicitly provide a test set and 10-fold cross validation is used for evaluation in previous works (Hsieh et al., 2017; Zhang et al., 2019). In this paper, we use two types of splits of AIMed to evaluate the impact of data leakage: AIMed (R) which **R**andomly splits the dataset into 10 folds; and AIMed (U) which splits the dataset into 10 folds such that the documents within each resultant split are **U**nique (according to the *document ID*) to other splits across each split. The *document ID* refers to the source document of a data instance, and data instances from the same source document have the same document ID, see example in Appendix A

4.2 Similarity measurement

The pseudo code for measuring similarity is shown in Algorithm 1. Given a test instance $test_i$, we compute its similarity with the training set using the training instance that is most similar with $test_i$. We then use the average similarity over all the test instances as an indicator to measure the extent of train/test overlap. The function $similarity(\cdot)$ can be any function for text similarity. In this paper, we use a simple bag-of-words approach to compute text similarity. We represent each train/test instance with a count vector of unigrams/bigrams/trigrams, ignoring stopwords, and compute the similarity using the cosine similarity.

4.3 Evaluate model performance

We assess the impact of data leakage on a machine learning model's performance. We split the test sets of BC2GM, ChEMU, BC2ACT and SST2 into four intervals considering four similarity threshold ranges (in terms of unigrams): [0-0.25),[0.25-0.50), [0.50-0.75), and [0.75-1.0]. For example, the test instances in the first interval are most different from the training set with a similarity less than 0.25. This method allows full control of the similarity of instances within each interval, but results in a different number of instances in each interval. Thus, we consider another scenario where we split the test set into 4 quartiles based on similarity ranking, so that the number of samples remain the same in each quartile but the threshold varies as a result.

We finetune a BERT (base and cased) model (Devlin et al., 2019) for each dataset using their own training set and compare the performance of the finetuned BERT model on the four different test intervals and test quartiles.

We compare the performances of AIMed (R) with AIMed (U) using 3 different models—Zhang et al. (2019) convolutional residual network, Hsieh et al. (2017) Bi-LSTM, and BioBERT (Lee et al., 2019). Following previous works, we preprocess the dataset and replace all non-participating proteins with neutral name *PROTEIN*, the participating entity pairs with *PROTEIN1* and *PROTEIN2*, so the model only ever sees the pseudo protein names.

5 Results

5.1 Similarity in datasets

Examples of similar train and test instances are shown in Table 1. The overall results of train-test similarities of all datasets are shown in Table 2.

In the BC2GM dataset, we find that there is 70% overlap between gene names in the train and test set. On further analysis, we find that 2,308 out of 6,331 genes in the test set have exact matches in the train set. In the AIMed (R) dataset, we can see that there is over 73% overlap, even measured in

Dataset	Task	uni	bi	tri
AIMED (R)	REL	**96.95**	**82.29**	**73.15**
AIMED (U)	REL	**67.14**	36.07	20.77
BC2GM ann	NER	**70.77**	19.55	5.41
BC2GM text	NER	33.19	13.12	4.20
BC3ACT	CLS	26.76	6.91	1.81
ChEMU ann	NER	**84.29**	30.67	6.83
ChEMU text	NER	**68.45**	42.39	31.63
SST2	CLS	46.06	17.38	1.39

Table 2: Train-test similarity using unigrams (uni), bigrams (bi), trigrams (tri). BC2GM and ChEMU are in BRAT standoff format and their similarities are shown for their text files (“text”) and annotation files (“ann”). Similarities beyond 60.0 are highlighed in **bold**.

Split type	Method	P	R	F1
O	BiLSTM	78.8	75.2	76.9
O	ConvRes	79.0	76.8	77.6
Replicated experiments				
R	BiLSTM	74.5	69.7	71.7
U	BiLSTM	**57.4**	**61.7**	**58.7**
R	ConvRes	71.1	69.2	69.9
U	ConvRes	**56.7**	**56.4**	**56.1**
R	BioBERT	79.8	76.7	77.9
U	BioBERT	**65.8**	**63.7**	**64.4**

Table 3: Performances on AIMed (R) and AIMed (U). The split type (O) indicates the original results from the authors.

D	SR	%	P	R	F1	A
BC2	F	100.0	77.5	86.4	81.7	
BC2	1I	19.8	68.8	81.1	**74.5**	
BC2	2I	74.1	78.3	86.9	82.4	
BC2	3I	5.1	83.8	90.6	**87.1**	
BC2	4I	0.9	79.5	77.5	78.5	
ChE	F	100.0	93.8	94.4	94.1	
ChE	1I	0.0	-	-	-	
ChE	2I	10.0	84.6	86.6	**85.6**	
ChE	3I	**60.0**	93.4	94.0	93.7	
ChE	4I	30.0	96.7	96.7	**96.7**	
BC3	F	100.0	45.1	84.1	58.7	82.1
BC3	1I	47.0	43.0	82.0	**56.4**	85.8
BC3	2I	51.0	46.0	85.6	59.9	78.8
BC3	3I	2.0	53.5	76.7	**63.0**	77.5
BC3	4I	0.0	0.0	0.0	0.0	0.0
SST	F	100.0	90.4	96.7	93.4	93.2
SST	1I	1.1	60.0	75.0	**66.7**	**85.0**
SST	2I	66.8	91.6	96.0	93.8	93.4
SST	3I	28.7	87.1	98.7	92.5	92.7
SST	4I	3.5	96.9	96.9	**96.9**	**96.8**

Table 4: Performances on various similarity thresholds and the corresponding percentage of test instances within the intervals. Datasets (D): BC2→BC2GM, ChE→ChEMU, BC3→BC3ACT, SST→SST2. The similarity threshold range (SR) $[0, 0.25)$ = 1I, $[0.25, 0.5)$ = 2I, $[0.5, 0.75)$ = 3I, $[0.75, 1]$ = 4I, $[0, 1]$ = F. Accuracy (A) is the official metric for the SST2 dataset according to GLUE benchmark, all others use F1-score (F1) as primary metric.

the trigrams, between train and test sets.

5.2 Model performance and similarity

We observe drops in F-scores of more than 10 points between AIMed (R) and AIMed (U) across all three models as shown in Table 3. This is in line with the similarity measurement in Table 2: the train-test similarity drops significantly from AIMed (R) to AIMed (U) since in AIMed (U) we only allow unique document IDs in different folds.

On the ChEMU NER dataset we observe nearly 10-point drop in F-score (96.7→85.6) from 4I to 2I as shown in Table 4.

On the BC2GM dataset, we also find that the model performance degrades from 82.4% to 74.5% in 2I compared to that in 1I. Surprisingly, F-score for 4I is substantially lower than that of 3I (78.5→87.1), despite 41 out of the total 47 instances in 4I having 100% similarity with the train set (full detailed samples shown in Appendix Table 10). A further investigation on this shows that **(a)** the interval 4I only has 0.9% (47/5000) of test instances; **(b)** a significant drop in recall (90.6→77.5) from 3I to 4I is caused by six instances whose input texts have exact matches in the train set (full samples shown in Appendix Table 11). This implies that the model doesn’t perform well even on the training data for these samples. Since BC2GM has over 70% overlap in the target gene mentions (Table 2), we also analysed the recall on the annotations that overlap between train and test. We find that the recall increases (84.5→87.8), see Appendix Table 8, compared to recall (81.1→90.6) as a result of input text similarity. Since BERT uses a word sequence-based prediction approach, the relatively high similarity in target annotations does not seem to make much difference compared to similarity in input text. However, if we used a dictionary-based approach, similarity in annotations could result in much higher recall compared to similarity in input text.

The BC3ACT dataset also exhibits the same trend where the F1-score improves (56.4→63.0) as the similarity increases. However, the accuracy

D	Q	Min	Max	P	R	F1	A
BC2	1	0.0	26.3	69.8	82.0	**75.4**	
BC2	2	26.3	31.6	74.5	85.9	79.8	
BC2	3	31.6	38.3	78.3	86.4	82.1	
BC2	4	38.3	100.0	83.0	88.9	**85.9**	
ChE	1	37.9	56.7	90.8	91.8	**91.3**	
ChE	2	56.8	68.2	93.3	94.4	93.8	
ChE	3	68.2	78.5	95.1	96.1	95.6	
ChE	4	78.6	99.8	97.1	97.4	**97.3**	
BC3	1	6.3	20.1	44.5	81.4	**57.6**	88.8
BC3	2	20.1	25.7	42.3	82.5	55.9	82.7
BC3	3	25.7	31.9	46.5	85.3	**60.2**	79.5
BC3	4	31.9	75.0	46.1	85.2	59.8	77.3
SST	1	0.0	36.5	90.8	95.2	**92.9**	**92.8**
SST	2	36.5	43.6	91.3	96.2	93.7	93.2
SST	3	43.6	53.5	91.2	97.3	**94.2**	**94.1**
SST	4	53.5	100.0	88.0	98.1	92.8	92.9

Table 5: Performances on four different test quartiles, where the number of samples in each quartile (Q) is kept same. The minimum (Min) and the maximum (Max) similarity within each quartile are also reported.

drops from 85.8→77.5. This is could be because while the train set has 50% positive classes, the test set has just 17% with 3 points higher mean similarity in positive samples (details in Appendix B).

On SST2, an increase in accuracy (85.0 → 96.8) from 1I to 4I is observed apart from a marginal 0.7-point drop (93.4 → 92.7) from 2I to 3I.

We also split the test sets into four equal-sized quartiles based on the similarity ranking of test instances, shown in Table 5. We observe similar phenomena as in the previous set of experiments for the dataset BC2GM, ChEMU, and BC3ACT. The only exception is for SST2 where the F-score has a relatively small but consistent increase from Q1 to Q3 (92.9→94.1) but drops to 92.8 in Q4.

6 Discussion

6.1 Quantifying similarity

The bag-of-words based approach to compute cosine similarity has been able to detect simple forms of overlap effectively as shown in Table 2. A trend that can be seen is that overlap is more common in tasks that are manual labour intensive, such as named entity recognition and relation extraction compared to text classification.

However, this approach may detect similarity even when the meanings are different, especially in the case of classification tasks as shown for SST2 in Table 1. Semantic Text Similarity (STS) measurement is a challenging task in its own right, with a large body of literature and a number of shared tasks organized to address it (Cer et al., 2017; Wang et al., 2020; Karimi et al., 2015). More sophisticated methods for similarity measurement developed in these contexts could be incorporated into the framework for measuring similarity of data set splits; for simple leakage detection it is arguably adequate. However, sophisticated methods can also potentially lead to a chicken and egg problem, if we use a machine learning model to compute semantic similarity.

The question of what level of similarity is acceptable is highly data and task-dependent. If the training data has good volume and variety, the training-test similarity will naturally be higher and so will the acceptable similarity.

6.2 Memorization vs. Generalization

We find that the F-scores tend to be higher when the test set input text is similar to the training set as shown in Table 3 and 4. While this might be apparent, quantifying similarity in the test set helps understand that high scores in the test set could be a result of similarity to the train set, and therefore measuring memorization and not a model's ability to generalize. If a model is trained on sufficient volume and variety of data then it may now matter if it memorizes or generalizes in a real world context, and a model's ability to memorize is not necessarily a disadvantage. However, in the setting of a shared task, we often do not have access to sufficiently large training data sets and hence it is important to consider the test/train similarity when evaluating the models. This implies that in real world scenarios the model may perform poorly when it encounters data not seen during training.

7 Conclusion

We conclude that quantifying train/test overlap is crucial to assessing real world applicability of machine learning in NLP tasks, given our reliance on annotated data for training and testing in the NLP community. A single metric over a held-out test set is not sufficient to infer generalizablity of a model. Stratification of test sets by similarity enables more robust assessment of memorization vs. generalization capabilities of models. Further development of approaches to structured consideration of model performance under different assumptions will improve our understanding of these tradeoffs.

References

Cecilia N Arighi, Phoebe M Roberts, Shashank Agarwal, Sanmitra Bhattacharya, Gianni Cesareni, Andrew Chatr-Aryamontri, Simon Clematide, Pascale Gaudet, Michelle Gwinn Giglio, Ian Harrow, et al. 2011. BioCreative III interactive task: an overview. *BMC Bioinformatics*, 12(S8):S4.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544. Association for Computational Linguistics.

Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk Wah Wong. 2005. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 33(2):139–155.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Satrajit Chatterjee. 2018. Learning and memorization. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 755–763, Stockholmsmässan, Stockholm Sweden. PMLR.

Walter Daelemans, Antal Van den Bosch, et al. 2005. *Memory-based language processing*. Cambridge University Press.

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

Nicola Ferro, Norbert Fuhr, Gregory Grefenstette, Joseph A Konstan, Pablo Castells, Elizabeth M Daly, Thierry Declerck, Michael D Ekstrand, Werner Geyer, Julio Gonzalo, et al. 2018. The dagstuhl perspectives workshop on performance modeling and prediction. In *ACM SIGIR Forum*, volume 52, pages 91–101. ACM New York, NY, USA.

Jiayuan He, Dat Quoc Nguyen, Saber A. Akhondi, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Zubair Afzal, Zenan Zhai, Biaoyan Fang, Hiyori Yoshikawa, Ameer Albahem, Lawrence Cavedon, Trevor Cohn, Timothy Baldwin, and Karin Verspoor. 2020. Overview of ChEMU 2020: Named entity recognition and event extraction of chemical reactions from patents. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Eleventh International Conference of the CLEF Association (CLEF 2020)*, volume 12260. Lecture Notes in Computer Science.

Yu-Lun Hsieh, Yung-Chun Chang, Nai-Wen Chang, and Wen-Lian Hsu. 2017. Identifying protein-protein interactions in biomedical literature using recurrent neural networks with long short-term memory. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 240–245, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Sarvnaz Karimi, Jie Yin, and Jiri Baum. 2015. Evaluation methods for statistically dependent text. *Computational Linguistics*, 41(3):539–548.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2020. Question and answer test-train overlap in open-domain question answering datasets.

So Yeon Min. 2020. Towards knowledge-based, robust question answering.

Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. 2018. emrQA: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2357–2368, Brussels, Belgium. Association for Computational Linguistics.

Bhanu Pratap Singh Rawat, Wei-Hung Weng, So Yeon Min, Preethi Raghavan, and Peter Szolovits. 2020. Entity-enriched neural models for clinical question answering. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 112–122, Online. Association for Computational Linguistics.

L Smith, LK Tanabe, RJ Ando, CJ Kuo, IF Chung, CN Hsu, YS Lin, R Klinger, CM Friedrich, K Ganchev, et al. 2008. Overview of BioCreative II gene mention recognition. *Genome Biology*, 9:S2–S2.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Yanshan Wang, Naveed Afzal, Sunyang Fu, Liwei Wang, Feichen Shen, Majid Rastegar-Mojarad, and Hongfang Liu. 2020. MedSTS: a resource for clinical semantic textual similarity. *Language Resources and Evaluation*, 54(1):57–72.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China. Association for Computational Linguistics.

H. Zhang, R. Guan, F. Zhou, Y. Liang, Z. Zhan, L. Huang, and X. Feng. 2019. Deep residual convolutional neural network for protein-protein interaction extraction. *IEEE Access*, 7:89354–89365.

A AIMed document examples

The following example shows how multiple data instances are extracted from a single document in AIMed dataset. The document with ID “`AIMed.d0`” has several instances including “`AIMed.d0.s0`” and “`AIMed.d0.s1`”. These instances thus have the same document id.

```
<corpus source="AIMed">
  <document id="AIMed.d0">
    <sentence id="AIMed.d0.s0" text="Th1/Th2 type cytokines in hepatitis B patients treated with interferon-alpha." seqId="s0"
        >
      <entity id="AIMed.d0.s0.e0" charOffset="60-75" type="protein" text="interferon-alpha" seqId="e0"/>
    </sentence>
    <sentence id="AIMed.d0.s1" text="OBJECTIVE: To investigate the relationship between the expression of Th1/Th2 type
         cytokines and the effect of interferon-alpha therapy." seqId="s1">
      <entity id="AIMed.d0.s1.e0" charOffset="110-125" type="protein" text="interferon-alpha" seqId="e1"/>
    </sentence>
  </document>
  <document id="AIMed.d1">
    <sentence id="AIMed.d1.s11" text="Involvement of BMP-2 signaling in a cartilage cap in osteochondroma." seqId="s11">
      <entity id="AIMed.d1.s11.e0" charOffset="15-19" type="protein" text="BMP-2" seqId="e15"/>
    </sentence>
  </document>
</corpus>
```

B Classwise similarity for BC3AST

The test set has 5090 negative samples compared to 910 positive samples, with 2.96 points higher mean similarity in positive samples.

Test label		Unigram	Bigram	Trigram
0	count	5090.00	5090.00	5090.00
	mean	26.31	6.70	1.73
	std	9.25	5.35	1.72
	min	6.28	0.00	0.00
	25%	19.70	3.29	0.79
	50%	25.16	5.07	1.39
	75%	31.53	8.29	2.27
	max	75.01	41.75	18.71
1	count	910.00	910.00	910.00
	mean	29.27	8.09	2.26
	std	9.36	6.00	1.73
	min	11.14	1.52	0.00
	25%	22.69	4.51	1.17
	50%	28.31	6.25	1.88
	75%	34.32	9.38	2.84
	max	74.01	51.20	18.97

Table 6: Class-wise similarity for BC3ACT dataset

C BERT and similarity thresholds

Table 7 shows the impact on precision, recall and F-score using different similarity thresholds on the BC2GM test set, which has approximately 6,300 annotations.

We also compare the recall when the target annotations are similar as shown in Table 8. We only compare unigrams, as the number of tokens in a gene name tends to be small (on average less than 3).

Table 9 shows BERT’s performance using bi-grams and trigrams on SST2 and BC3AST datasets.

Dataset	**N**	**SR**	%	**P**	**R**	**F**
BC2GM	-	-	100	77.5	86.4	81.7
BC2GM	1	1I	19.8	68.8	81.1	74.5
BC2GM	1	2I	74.1	78.2	86.9	82.3
BC2GM	1	3I	5.1	83.8	90.6	87.1
BC2GM	1	4I	1.0	79.5	77.5	78.5
BC2GM	2	1I	91.7	76.9	86.3	81.4
BC2GM	2	2I	7.5	82.5	88.1	85.2
BC2GM	2	3I	0.3	1.0	1.0	1.0
BC2GM	2	4I	0.5	78.9	76.9	77.9
BC2GM	3	1I	98.5	77.4	86.4	81.7
BC2GM	3	2I	0.9	85.2	88.5	86.8
BC2GM	3	3I	0.1	50.0	100.0	66.7
BC2GM	3	4I	0.5	80.6	76.3	78.4

Table 7: NER performances of BERT on various similarity threshold range (SR) and the corresponding percentage of instances when the similarity is computed using N-grams (N = 1, 2 and 3) in the input text. The range $[0, 0.25) =$ 1I, $[0.25, 0.5) =$ 2I, $[0.5, 0.75) =$ 3I, $[0.75, 1] =$ 4I, $[0, 1] =$ F.

Dataset	N	SR	%	Recall
BC2GM (anno)	-	F	100.0	86.4
BC2GM (anno)	1	1I	16.7	84.5
BC2GM (anno)	1	2I	5.6	81.8
BC2GM (anno)	1	3I	24.7	85.6
BC2GM (anno)	1	4I	53.0	87.8

Table 8: NER score on BERT at various similarity threshold range (SR) and the corresponding % of samples using ngram N = 1 in the output annotated gene mentions.

Dataset	N	SR	%	P	R	F1	A
BC3ACT	-	F	100.0	45.1	84.1	58.7	82.1
BC3ACT	1	1I	47.0	43.0	82.0	56.4	85.8
BC3ACT	1	2I	51.0	46.0	85.6	59.9	78.8
BC3ACT	1	3I	2.0	53.5	76.7	63.0	77.5
BC3ACT	1	4I	0.0	0.0	0.0	0.0	0.0
BC3ACT	2	1I	98.2	45.0	84.1	58.6	82.2
BC3ACT	2	2I	1.8	48.8	83.3	61.5	76.6
BC3ACT	2	3I	0.0	100.0	100.0	100.0	100.0
BC3ACT	2	4I	0.0	0.0	0.0	0.0	-
BC3ACT	3	1I	100.0	45.1	84.1	58.7	82.1
BC3ACT	3	2I	0.0	0.0	0.0	0.0	-
BC3ACT	3	3I	0.0	0.0	0.0	0.0	-
BC3ACT	3	4I	0.0	0.0	0.0	0.0	-
SST2	-	F	100.0	90.4	96.7	93.4	93.2
SST2	1	1I	1.1	60.0	75.0	66.7	85.0
SST2	1	2I	66.8	91.6	96.0	93.8	93.4
SST2	1	3I	28.7	87.1	98.7	92.5	92.7
SST2	1	4I	3.5	96.9	96.9	96.9	96.8
SST2	2	1I	64.0	88.6	96.0	92.2	92.3
SST2	2	2I	30.8	93.1	97.3	95.1	94.8
SST2	2	3I	4.8	93.1	100.0	96.4	95.4
SST2	2	4I	0.4	100.0	100.0	100.0	100.0
SST2	3	1I	97.6	90.5	96.6	93.4	93.3
SST2	3	2I	1.9	82.6	100.0	90.5	88.2
SST2	3	3I	0.5	100.0	100.0	100.0	100.0
SST2	3	4I	0.0	0.0	0.0	0.0	-

Table 9: SST2 and BC3ACT similarity thresholds using ngram N = 1,2 and 3. The range $[0, 0.25) = 1I$, $[0.25, 0.5) = 2I$, $[0.5, 0.75) = 3I$, $[0.75, 1] = 4I$, $[0, 1] = F$

D High similarity BC2GM samples

Table 10 shows the 75% similarity samples in the BC2GM dataset. The samples that caused the drop in recall are shown in Table 11.

Score	Test	Train
76.45	Histological and immunophenotypic studies revealed 12 large cell lymphomas (11 B cell and one T cell), two small noncleaved cell lymphomas (B-cell phenotype), and five low grade B-cell lymphomas (two small lymphocytic and three follicular mixed lymphomas).	The cases included 35 de novo diffuse aggressive lymphomas (DAL; 19 large-cell, 4 mixed-cell, and 12 large-cell immunoblastic), 52 transformed aggressive lymphomas derived from follicular lymphomas (TFL), 42 indolent follicular lymphomas (FL), 14 mantle cell lymphomas (MCL), and 27 small noncleaved cell lymphomas (SNCL).
77.46	98, 93-98).	356, 93-98].
81.65	Free protein S deficiency in acute ischemic stroke.	Ischemic stroke due to protein C deficiency.
83.41	In stage I, histochemistry for copper was positive in 11 out of 21 cases: 6 cases were T+; 1 case R+ and 2 cases O+; 2 cases were T+, R+, O+.	3 cases
86.60	STUDY DESIGN: Retrospective review.	DESIGN: Retrospective study.
86.60	Non-dialyzable transfer factor	Dialyzable transfer factor.
100.00	Recently we have performed a detailed analysis of specific neuronal populations affected by the mutation which shed new light on the role of Krox-20 in the segmentation and on the physiological consequences of its inactivation.	Recently we have performed a detailed analysis of specific neuronal populations affected by the mutation which shed new light on the role of Krox-20 in the segmentation and on the physiological consequences of its inactivation.
100.00	Slowly adapting type I mechanoreceptor discharge as a function of dynamic force versus dynamic displacement of glabrous skin of raccoon and squirrel monkey hand.	Slowly adapting type I mechanoreceptor discharge as a function of dynamic force versus dynamic displacement of glabrous skin of raccoon and squirrel monkey hand.
100.00	The recruitment of constitutively phosphorylated p185(neu) and the activated mitogenic pathway proteins to this membrane-microfilament interaction site provides a physical model for integrating the assembly of the mitogenic pathway with the transmission of growth factor signal to the cytoskeleton.	The recruitment of constitutively phosphorylated p185(neu) and the activated mitogenic pathway proteins to this membrane-microfilament interaction site provides a physical model for integrating the assembly of the mitogenic pathway with the transmission of growth factor signal to the cytoskeleton.
100.00	A heterologous promoter construct containing three repeats of a consensus Sp1 site, cloned upstream of a single copy of the ZII (CREB/ AP1) element from the BZLF1 promoter linked to the beta-globin TATA box, exhibited phorbol ester inducibility.	A heterologous promoter construct containing three repeats of a consensus Sp1 site, cloned upstream of a single copy of the ZII (CREB/ AP1) element from the BZLF1 promoter linked to the beta-globin TATA box, exhibited phorbol ester inducibility.
100.00	The reconstituted RNA polymerases containing the mutant alpha subunits were examined for their response to transcription activation by cAMP-CRP and the rrnBP1 UP element.	The reconstituted RNA polymerases containing the mutant alpha subunits were examined for their response to transcription activation by cAMP-CRP and the rrnBP1 UP element.
100.00	Analysis of 1 Mb of published sequence from the region of conserved synteny on human chromosome 5q31-q33 identified 45 gene candidates, including 35 expressed genes in the human IL-4 cytokine gene cluster.	Analysis of 1 Mb of published sequence from the region of conserved synteny on human chromosome 5q31-q33 identified 45 gene candidates, including 35 expressed genes in the human IL-4 cytokine gene cluster.
100.00	Although RAD17, RAD24 and MEC3 are not required for cell cycle arrest when S phase is inhibited by hydroxyurea (HU), they do contribute to the viability of yeast cells grown in the presence of HU, possibly because they are required for the repair of HU-induced DNA damage.	Although RAD17, RAD24 and MEC3 are not required for cell cycle arrest when S phase is inhibited by hydroxyurea (HU), they do contribute to the viability of yeast cells grown in the presence of HU, possibly because they are required for the repair of HU-induced DNA damage.
100.00	The promoter for HMG-CoA synthase contains two binding sites for the sterol regulatory element-binding proteins (SREBPs).	The promoter for HMG-CoA synthase contains two binding sites for the sterol regulatory element-binding proteins (SREBPs).
100.00	Coronary vasoconstriction caused by endothelin-1 is enhanced by ischemia-reperfusion and by norepinephrine present in concentrations typically observed after neonatal cardiopulmonary bypass.	Coronary vasoconstriction caused by endothelin-1 is enhanced by ischemia-reperfusion and by norepinephrine present in concentrations typically observed after neonatal cardiopulmonary bypass.
100.00	(LH P ¡ 0.05, LH/FSH P ¡ 0.01).	(LH P ¡ 0.05, LH/FSH P ¡ 0.01).
100.00	Determinants of recurrent ischaemia and revascularisation procedures after thrombolysis with recombinant tissue plasminogen activator in primary coronary occlusion.	Determinants of recurrent ischaemia and revascularisation procedures after thrombolysis with recombinant tissue plasminogen activator in primary coronary occlusion.
100.00	The human SHBG proximal promoter was analyzed by DNase I footprinting, and the functional significance of 6 footprinted regions (FP1-FP6) within the proximal promoter was studied in human HepG2 hepatoblastoma cells.	The human SHBG proximal promoter was analyzed by DNase I footprinting, and the functional significance of 6 footprinted regions (FP1-FP6) within the proximal promoter was studied in human HepG2 hepatoblastoma cells.
100.00	Biol.	Biol.
100.00	Copyright 1999 Academic Press.	Copyright 1999 Academic Press.
100.00	These results demonstrate a specific association of SIV and HIV-2 nef, but not HIV-1 nef, with TCRzeta.	These results demonstrate a specific association of SIV and HIV-2 nef, but not HIV-1 nef, with TCRzeta.
100.00	Urease activity, judged as the amount of ammonia production from urea, could be measured at 25 ng per tube (S/N = 1.5) with Jack bean meal urease.	Urease activity, judged as the amount of ammonia production from urea, could be measured at 25 ng per tube (S/N = 1.5) with Jack bean meal urease.
100.00	Copyright 1999 Academic Press.	Copyright 1999 Academic Press.
100.00	IV.	IV.
100.00	Copyright 1998 Academic Press.	Copyright 1998 Academic Press.
100.00	IV.	IV.
100.00	Biol.	Biol.
100.00	Copyright 1999 Academic Press.	Copyright 1999 Academic Press.
100.00	Copyright 1998 Academic Press.	Copyright 1998 Academic Press.
100.00	Copyright 2000 Academic Press.	Copyright 2000 Academic Press.
100.00	1988).	(1988) J.
100.00	Biol.	Biol.
100.00	Acad.	Acad.
100.00	Virol.	Virol.
100.00	1995.	(1995) J.
100.00	Natl.	Natl.
100.00	Copyright 1999 Academic Press.	Copyright 1999 Academic Press.
100.00	The activated glucocorticoid receptor forms a complex with Stat5 and enhances Stat5-mediated transcriptional induction.	The activated glucocorticoid receptor forms a complex with Stat5 and enhances Stat5-mediated transcriptional induction.
100.00	Copyright 1999 Academic Press.	Copyright 1999 Academic Press.
100.00	Chem.	Chem.
100.00	Appl.	Appl.
100.00	Copyright 1998 Academic Press.	Copyright 1998 Academic Press.
100.00	Sci.	Sci.
100.00	(1992) J.	(1992) J.
100.00	Acad.	Acad.
100.00	Mutational analysis of yeast CEG1 demonstrated that four of the five conserved motifs are essential for capping enzyme function in vivo.	Mutational analysis of yeast CEG1 demonstrated that four of the five conserved motifs are essential for capping enzyme function in vivo.
100.00	We also show that in fusions with the DNA binding domain of GAL4, full activity requires the entire BHV-alpha TIF, although both amino and carboxyl termini display some activity on their own.	We also show that in fusions with the DNA binding domain of GAL4, full activity requires the entire BHV-alpha TIF, although both amino and carboxyl termini display some activity on their own.

Table 10: Samples with over 75% similarity in the BC2GM dataset

Gene	Position	Input
capping enzyme	88 100	Mutational analysis of yeast CEG1 demonstrated that four of the five conserved motifs are essential for capping enzyme function in vivo.
human IL-4 cytokine gene	145 165	Analysis of 1 Mb of published sequence from the region of conserved synteny on human chromosome 5q31-q33 identified 45 gene candidates, including 35 expressed genes in the human IL-4 cytokine gene cluster.
LH	1 2	(LH P ¡ 0.05, LH/FSH P ¡ 0.01).
LH	10 11	(LH P ¡ 0.05, LH/FSH P ¡ 0.01).
FSH	13 15	(LH P ¡ 0.05, LH/FSH P ¡ 0.01).
Urease	0 5	Urease activity, judged as the amount of ammonia production from urea, could be measured at 25 ng per tube (S/N = 1.5) with Jack bean meal urease.
Jack bean meal urease	101 118	Urease activity, judged as the amount of ammonia production from urea, could be measured at 25 ng per tube (S/N = 1.5) with Jack bean meal urease.
cAMP-CRP	117 124	The reconstituted RNA polymerases containing the mutant alpha subunits were examined for their response to transcription activation by cAMP-CRP and the rrnBP1 UP element.
HIV-2 nef	51 58	These results demonstrate a specific association of SIV and HIV-2 nef, but not HIV-1 nef, with TCRzeta.
HIV-1 nef	66 73	These results demonstrate a specific association of SIV and HIV-2 nef, but not HIV-1 nef, with TCRzeta.

Table 11: Test samples where the model failed to detect genes, lowering recall, despite the input raw text being an exact match to the training sample

An Expert Annotated Dataset for the Detection of Online Misogyny

Ella Guest
The Alan Turing Institute
University of Manchester
ella.guest@manchester.ac.uk

Bertie Vidgen
The Alan Turing Institute
bvidgen@turing.ac.uk

Alexandros Mittos
Queen Mary University of London
University College London
alexandros@mittos.net

Nishanth Sastry
University of Surrey
King's College London
The Alan Turing Institute
nsastry@turing.ac.uk

Gareth Tyson
Queen Mary University of London
The Alan Turing Institute
g.tyson@qmul.ac.uk

Helen Margetts
The Alan Turing Institute
Oxford Internet Institute
hmargetts@turing.ac.uk

Abstract

Online misogyny is a pernicious social problem that risks making online platforms toxic and unwelcoming to women. We present a new hierarchical taxonomy for online misogyny, as well as an expert labelled dataset to enable automatic classification of misogynistic content. The dataset consists of 6,567 labels for Reddit posts and comments. As previous research has found untrained crowdsourced annotators struggle with identifying misogyny, we hired and trained annotators and provided them with robust annotation guidelines. We report baseline classification performance on the binary classification task, achieving accuracy of 0.93 and F1 of 0.43. The codebook and datasets are made freely available for future researchers.

1 Introduction

Misogyny is a problem in many online spaces, making them less welcoming, safe, and accessible for women. Women have been shown to be twice as likely as men to experience gender-based online harassment (Duggan, 2017). This misogyny can inflict serious psychological harm on women and produce a 'silencing effect', whereby women self-censor or withdraw from online spaces entirely, thus limiting their freedom of expression (Mantilla, 2013; International, 2017). Tackling such content is increasingly a priority for social media platforms and civil society organisations.

However, detecting online misogyny remains a difficult task (Hewitt et al., 2016; Nozza et al., 2019). One problem is the lack of high-quality datasets to train machine learning models, which would enable the creation of efficient and scalable automated detection systems (Anzovino et al., 2018). Previous research has primarily used Twitter data and there is a pressing need for other platforms to be researched Lynn et al. (2019a). Notably, despite social scientific studies that show online misogyny is pervasive on some Reddit communities, to date a training dataset for misogyny has not been created with Reddit data. In this paper we seek to address the limitations of previous research by presenting a dataset of Reddit content with expert labels for misogyny that can be used to develop more accurate and nuanced classification models.

Our contributions are four-fold. First, we develop a detailed hierarchical taxonomy based on existing literature on online misogyny. Second, we create and share a detailed codebook used to train annotators to identify different types of misogyny. Third, we present a dataset of 6,383 entries from Reddit. Fourth, we create baseline classification models based on these datasets. All of the research artefacts are made freely available via a public repository for future researchers.[1]

The dataset itself has several innovations which differentiate it from previous training datasets for misogyny. First, we use chronological and structured conversation threads, which mean annotators take into account the previous context of each entry before labelling. Second, we distinguish between conceptually distinct types of misogynistic abuse, including gendered personal attacks, use of misogynistic pejoratives, and derogatory and threatening language. Third, we highlight the specific section of text, also known as a 'span', on which each label is based. This helps differentiate between multiple labels on one piece of text. Fourth, we use trained annotators, rather than crowd-sourced workers. We also use facilitated meetings to decide the final labels rather than just a majority decision. Both of these factors lead to a high-quality dataset. Additionally, we provide a second dataset with the original labels made by annotators before the final labels were decided.

---

[1] https://github.com/ellamguest/online-misogyny-eacl2021

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1336–1350
April 19 - 23, 2021.

2 Background

Most previous classification work on online misogyny has used data from Twitter (Waseem and Hovy, 2016; Anzovino et al., 2018; Jha and Mamidi, 2017). However, social scientific and ethnographic research shows that Reddit is increasingly home to numerous misogynistic communities. Reddit is a social news website organised in to topic-based communities. Each subreddit acts as a message board where users make posts and hold discussions in comment threads on those posts. In recent years it has become a hub for anti-feminist activism online (Massanari, 2017; Ging and Siapera, 2018). It is also home to many misogynistic communities, particularly those associated with the 'manosphere', a loosely connected set of communities which perpetuate traditional forms of misogyny and develop new types of misogynistic discourse which in turn spread to other online spaces (Ging, 2017; Zuckerberg, 2018; Ging et al., 2019; Farrell et al., 2019; Ribeiro et al., 2020). Recent research suggests that the rate of misogynistic content in the Reddit manosphere is growing and such content is increasingly more violent (Farrell et al., 2019).

Waseem and Hovy (2016) provided a widely-used dataset for abusive language classification. They used expert annotators to identify sexist and racist tweets based on a set of criteria drawn from critical race theory. The tweets were initially labelled by the authors then reviewed by a third annotator. The resulting dataset consists of 17k tweets, of which 20% are labelled as sexist. However 85% of the disagreements between annotators were over sexism labels, which shows that even experienced coders of abusive language can have difficultly identifying gendered abuse.

Jha and Mamidi (2017) extended on the Waseem and Hovy (2016) dataset to distinguish between between 'benevolent' and 'hostile' sexism (Glick and Fiske, 1997). They classed all sexist labels in the previous dataset as 'Hostile' and all non-sexist labels as 'Other'. They then augmented the dataset by collecting tweets using keyword sampling on benevolently sexist phrases (e.g. 'smart for a girl') and extracted those manually identified as 'benevolent sexism'. In the combined dataset of 10,095 unique tweets 712 were labelled as 'benevolent', 2,254 as 'hostile', and 7,129 as 'not sexist'. They thus found that in the data hostile sexism was more than three times as common as the benevolent form. Their work highlights the need for greater attention to be placed on forms of 'subtle abuse', particularly for online misogyny (Jurgens et al., 2019).

Anzovino et al. (2018) developed a taxonomy with five categories of misogyny, drawn from the work of Poland (2016): Stereotype & Objectification, Dominance, Derailing, Sexual Harassment & Threats of Violence, Discredit. They used a combination of expert and crowdsourced annotation to apply the taxonomy and present a dataset of 4,454 tweets with balanced levels of misogynistic and non-misogynistic content. A shared task confirmed that the dataset could be used to distinguish misogynistic and non-misogynistic content with high accuracy, but performance was lower in differentiating between types of misogyny (Fersini et al., 2018).

Lynn et al. (2019b) provide a dataset of 2k Urban Dictionary definitions of which half are labelled as misogynistic. In Lynn et al. (2019a) they show that deep learning techniques had greater accuracy in detecting misogyny than conventional machine learning techniques.

3 Data collection

We collected conversation threads from Reddit. Given that a very small amount of content on social media is hateful, a key difficulty when creating datasets for annotation is collecting enough instances of the 'positive' class to be useful for machine learning (Schmidt and Wiegand, 2017; Fortuna and Nunes, 2018). However, sampling strategies can introduce biases in the composition and focus of the datasets if overly simplistic methods are used, such as searching for explicitly misogynistic terms (Wiegand et al., 2019).

To ensure that our dataset contains enough misogynistic abuse we began with targeted sampling, taking content from 12 subreddits that were identified as misogynistic in previous research. This includes subreddits such as `r/MensRights`, `r/seduction`, and `r/TheRedPill`. The sources used to identify these subreddits are available in Table 9 in the Appendix. We then identified 22 additional subreddits which had been recommended by the moderators/owners of the original 12 subreddits in the 'sidebar'. Some of these are not misogynistic but discuss women (e.g. `r/AskFeminists`) and/or are otherwise related to misogyny. For example, `r/exredpill` is a support group for former members of the misogynistic subreddit `r/TheRedPill`. Table 9 in the Appendix lists the 34 targeted subreddits and the number of entries and threads for each in the dataset. Over 11 weeks, for each subreddit, we collected the entire threads of the 20 most popular posts that week.

Using subreddits to target the sampling rather than keywords should ensure that more linguistic variety is captured, minimising the amount of bias as keywords such as 'slut' are associated with more explicit and less subtle forms of abuse. Nonetheless, only sampling from suspected misogynistic communities could still lead to classifiers which only identify the forms of misogyny found in those targeted contexts (Davidson et al., 2017; Wiegand et al., 2019; Sap et al., 2019). To account for this potential bias, and to enable greater generalisabilty, we sampled content from 71 randomly selected subreddits. They accounted for 18% of threads and 16% of entries in our dataset. For each randomly selected subreddit, we collected the thread of the most popular post. All threads were in English with the exception of one thread from the subreddit `r/Romania`.

Posts and comments were collected from February to May 2020 using the python package PRAW, a wrapper for the Reddit API (Boe, 2020). Posts on Reddit have a text title and a body which can be text, an image, or a link. For posts with a text body we combined this with the post title to create a single unit of text. For the 29% of posts where the body was an image we also collected the image.

4 Taxonomy

We developed a hierarchical taxonomy with three levels. First, we make a binary distinction between Misogynistic content and Non-misogynistic content, which are mutually exclusive. Second, we elaborated subtypes of Misogynistic and Non-misogynistic content. For Misogynistic content we defined four categories: (i) *Misogynistic Pejoratives*, (ii) descriptions of *Misogynistic Treatment*, (iii) acts of *Misogynistic Derogation* and (iv) *Gendered Personal attacks* against women. For Non-misogynistic content we defined three categories: (i) *Counter speech* against misogyny, (ii) *Non-misogynistic personal attacks* and (iii) *None of the categories*. Third, we included additional flags for some of the second level categories. Within both Misogynistic and Non-misogynistic content, the second level categories are not mutually exclusive, thereby allowing for multiple labels per entry. For instance, a Misogynistic entry could be assigned labels for both a *Pejorative* and *Treatment*.

This taxonomy draws on the typologies of abuse presented by Waseem et al. (2017) and Vidgen et al. (2019) as well as theoretical work in online misogyny research (Filipovic, 2007; Mantilla, 2013; Jane, 2016; Ging, 2017; Anzovino et al., 2018; Ging and Siapera, 2019; Farrell et al., 2019). It was developed by reviewing existing literature on online misogyny and then iterating over small samples of the dataset. This deductive-inductive process allowed us to ensure that conceptually distinct varieties of abuse are separated and that different types of misogyny can be unpicked. This is important given that they can have very different impacts on victims, different causes, and reflect different outlooks and interests on the part of the speaker.

4.1 Misogynistic content

Misogynistic content directs abuse at women or a closely related gendered group (e.g. feminists). This content can fall in to four non-mutually exclusive categories.

4.1.1 Misogynistic pejoratives

Misogynistic pejoratives are terms which are used to disparage women. It includes terms which are explicitly insulting and derogatory, such as 'slut' or 'whore', as well as terms which implicitly express negativity or animosity against women, such as 'Stacy' or 'Becky'. For example, 'Stacy' is a term used in the incel community to describe women considered attractive and unattainable, in opposition to a more average and attainable 'Becky' (Jennings, 2018).

4.1.2 Misogynistic treatment

Misogynistic treatment is content that discusses, advocates, incites or plans negative or harmful treatment of women. It includes expressing intent to take action against women, as well as expressing desires about how they should be treated. Misogynistic treatment contains third-level subcategories: *Threatening language* and *Disrespectful actions*.

1. *Threatening language*: Content which expresses an intent/desire to inflict/cause women to suffer harm, or expresses support for, encourages, advocates or incites such harm. It is an 'explicit' form of abuse. It falls in to three thematic groups:

   (a) *Physical violence*: non-sexual physical violence such as killing, maiming, beating, etc. e.g. 'Feminists deserve to be shot'.
   (b) *Sexual violence*: explicit sexual violence such as rape, penetration, molestation, etc. e.g. 'Someone should rape her – that would put her in her place'.
   (c) *Privacy*: an invasion of privacy such as the disclosure of personal information (i.e. doxing) or threats to visit them. e.g. 'I know where you live, bitch'.

2. *Disrespectful actions*: Content which treats/portrays women as either lacking or not deserving independence/autonomy. This includes more subtly abusive statements about how women should be treated and what they should be allowed to do. It is an 'implicit' form of abuse. It falls in to four thematic groups:

   (a) *Controlling*: suggesting or stating that women should be controlled in some way, especially by a man or men. E.g. 'I would never let my girlfriend do that'.
   (b) *Manipulation*: using or advocating the use of tactics such as lying and gaslighting to manipulate what women do or think. E.g. 'Told my last girlfriend she was hallucinating when she saw the texts from my side piece'.
   (c) *Seduction and conquest*: discussing woman solely as sexual conquests or describing previous incidences of when they have been treated as such. E.g. 'Got her home and used her so hard'.
   (d) *Other*: content that is not covered by the other subcategories.

4.1.3 Misogynistic derogation

Misogynistic derogation is content that demeans or belittles women. This content can be explicitly or implicitly abusive. It is separated into third-level subcategories:

1. *Intellectual inferiority*: making negative judgements of women's intellectual abilities, such as a lack of critical thinking or emotional control. This includes content which infantilizes women. An implicit example would be 'My gf cries at the stupidest shit – lol!' for suggesting irrational emotional responses. An explicit example is 'Typical stupid bitch – talking about things she doesn't understand'.

2. *Moral inferiority*: making negative judgements of women's moral worth, such as suggesting they are deficient or lesser to men in some way. This includes subjects such as superficiality (e.g. only liking men who are rich or attractive), promiscuity, and untrustworthiness. An implicit example is 'Girls love your money more than you'. An explicit example is 'My ex-girlfriend was a whore, she slept with every guy she saw'.

3. *Sexual and/or physical limitations*: making negative judgements of women's physical and/or sexual ability. This includes perceived unattractiveness (i.e. a lack of sexual desirability), ugliness (i.e. a lack of beauty), frigidness (i.e. a lack of sexual willingness), as well as belittling statements about feminine physical weakness. An implicit example is 'I gave it my A-game but she would not give in, so uptight!' An explicit example is 'Yikes, Dianne Abbott looks like a monkey!'

4. *Other*: content that is not covered by the other subcategories but is derogatory towards women.

4.1.4 Gendered personal attacks

Gender personal attacks are highly gendered attacks and insults. This category is used only when the nature of the abuse is misogynistic, e.g. 'Hilary Clinton is such a stupid bitch, someone should give her a good fucking and put her in her place'.

The category has a level three flag for the *gender of the recipient* of the abuse. We include this flag as research has shown that men can also be targeted by misogynistic attacks (Jane, 2014). The gender can either be a woman (e.g. 'That chick is dumb'), a man (e.g. 'This dude is a piece of shit') or unknown (e.g. 'You're are an idiot, fuck off'). If the content was replying to an entry which reveals the recipient's gender we can infer it from this context. For example if 'You're an idiot, fuck off' was a response to 'I'm a man and a feminist there's nothing contradictory about that' we know the abuse is targeted at a man.

4.2 Non-misogynistic content

Non-misogynistic content can fall in to three non-mutually exclusive categories, all of which are relevant for misogyny research.

4.2.1 Non-misogynistic personal attacks

Interpersonal abuse which is *not* misogynistic. We include this category to allow for a comparison of the nature of abuse directed at women and men (Duggan, 2017). It includes content which personally attacks a woman but is not misogynistic in nature, e.g. 'Hilary Clinton has no clue what she's talking about, idiot!'. It uses the same level three flag for the *gender of the recipient* as Misogynistic personal attack. This allows us to compare the rates of personal attacks against women and men.

Note that although it is possible for an entry to contain both Misogyny and a Non-misogynistic personal attack, this was very rare. In such cases, we chose to not annotate the Non-misogynistic personal attack in order to keep the first level as a binary distinction.

4.2.2 Counter speech

Counter speech is content which challenges, refutes, and puts into question previous misogynistic abuse in a thread. It could directly criticise previous abuse (e.g. 'What you said is unacceptable'), specifically accuse it of prejudice (e.g. 'That's incredibly sexist'), or offer a different perspective which challenges the misogyny (e.g 'That's not how women act, you're so wrong').

4.2.3 None of the categories

Content which does not contain misogynistic abuse, pejoratives, or related counter speech as defined in the previous categories. This content is often not related to abuse or to women in general. That said, it can include other forms of abusive language which are not misogynistic.

5 Annotation Methodology

A key difficulty in the formation of abusive language training datasets is producing high quality annotations. Several factors affect this. Deciding between similar categories, such as 'hate speech' versus 'offensive language' can be difficult (Waseem et al., 2017). Determining the right category often requires close scrutiny and sustained critical thinking from annotators. Annotators may face information overload if asked to work with too many categories, both in terms of breadth (e.g. annotating for different types of abuse) and depth (e.g. working with numerous subcategories). Further, annotators may have different values and experiences and so make different assessments of the content they observe, especially when context plays a large role. Annotators will also have unconscious social biases which may mean they interpret coding instructions differently to each other, and to how they were intended by the research authors. For instance, Davidson et al. (2017) found that crowdsourced annotators were more likely to label sexist content as merely 'offensive' while racist and homophobic content was considered 'hate speech'.

To mitigate such annotator biases, we used expert annotators specifically trained in identifying misogynistic content, as well as a group-based facilitation process to decide final labels. Due to time and resource constraints, the final dataset is smaller than if we had used crowdsourced workers but captures more nuanced and detailed cases of misogyny. Six annotators worked on the dataset. Annotators were trained in the use of a codebook detailing the taxonomy and annotation guidelines. The codebook was updated over time based on feedback from the annotators. Demographic information on the annotators is available in Appendix A.2

5.1 Annotation process and disagreements

Annotators independently marked up each entry for the three levels presented in Section 4. For all level two categories other than 'None', they also highlighted the specific part of the entry which was relevant to the labelled category (the 'span'). This is particularly important information for long posts which can contain multiple forms of abuse.

Each entry was annotated by either two (43%) or three (57%) annotators. If all annotators made the exact same annotation (including all three levels and highlighting) this was accepted as the final annotation. All other entries were flagged as disagreements. Annotators reviewed the disagreements in weekly meetings which were overseen by an expert facilitator, a PhD researcher who had developed the annotation taxonomy and was familiar with the literature on online misogyny and hate speech classification. The role of the facilitator was to promote discussion between annotators and ensure the final labels reflected the taxonomy. Each disagreement was discussed until the annotators reached a consensus on the final agreed label or labels.

5.2 Inter-annotator reliability

For the level one binary task the Fleiss' Kappa is 0.484 and the Krippendorf's alpha is 0.487. By conventional NLP standards these results appear low. However they are equivalent to, or above, those of existing abusive content datasets. Sanguinetti et al. (2018) report category-wise Kappas from k=0.37 for *offence* to k=0.54 for *hate*. Gomez et al. (2020) have a Kappa of 0.15 in the "MMH150" dataset of hateful memes. Fortuna and Nunes (2018) report a Kappa of 0.17 for a text-only task. Krippendorf's alpha is similar to the 0.45 reported by Wulczyn et al. (2017).

We also calculated level two category-wise Fleiss' Kappas for each of the 17 sets of annotator groups, then took the mean across all groups (Ravenscroft et al., 2016). Table 1 shows the breakdown of Kappas per category. There was greatest agreement for *Misogynistic pejoratives* (k=0.559) down to the lowest agreement for *Misogynistic personal attacks* (k=0.145).

Category	Fleiss' Kappa
Mis. Pejoratives	0.559
Mis. Treatment	0.210
Mis. Derogation	0.364
Mis. Personal attack	0.145
Nonmis. Personal attack	0.239
Counter speech	0.179
None of the categories	0.485

Table 1: Category-wise Fleiss' Kappa

Our taxonomy has seven partially overlapping categories, and as such annotation is considerably more difficult compared with most prior work, which tends to involve only binary labelling. As such, whilst slightly low, we believe that our agreement scores show the robustness of our annotation approach. Further, all disagreements were then discussed with an expert adjudicator, meaning that points of disagreement were addressed before the final labels were determined.

6 Prevalence of the categories

Of the 6,567 agreed labels in the final dataset 10.6% are Misogynistic (n=699) and 89.4% are Non-misogynistic (n=5,868). Tables 2 and 3 show the number of labels in the final dataset for each of the Misogynistic and Non-misogynistic categories, broken down by the level two categories. The vast majority of entries fall under *None of the categories* (88.6% of all labels). The next most common category is *Misogynistic Pejoratives* followed by *Misogynistic Derogation* pejoratives (4.2%). There are relatively few labels for *Personal attacks* with just 0.7% in total for each of the Misogynistic and Non-misogynistic categories, respectively. The least common category is *Counter speech* against misogyny, with only ten cases (0.2%).

Category	Number	Total %
Pejorative	276	4.2%
Treatment	103	1.6%
Derogation	285	4.3%
Personal attack	35	0.7%
Total	**696**	**10.6%**

Table 2: Breakdown of *Misogynistic* category counts

Category	Number	Total %
Personal attack	43	0.7%
Counter speech	10	0.2%
None	5815	88.6%
Total	**5868**	**89.4%**

Table 3: Breakdown of *Non-misogynistic* category counts

6.1 Misogynistic pejoratives

Annotators identified at least one misogynistic pejorative in 4.2% of all entries. The most common misogynistic term in the labels is 'bitch' (n=43) followed by 'stacy' (24) and 'stacies' (21).

6.2 Misogynistic treatment

There are 103 labels of *Treatment*. Figure 1 shows the number of labels for each level three subcategory. There are almost five times as many labels for *Disrespectful actions* (n=85) than *Threatening language* (n=18).

Both level three subcategories were broken down into more specific misogynistic themes. Within *Disrespectful actions*, *Seduction and conquest* is the most common topic, with twice as many labels as the second most common, *Controlling* (43 vs 17). And, within *Threatening language*, *Physical violence* was the most common theme (13) while

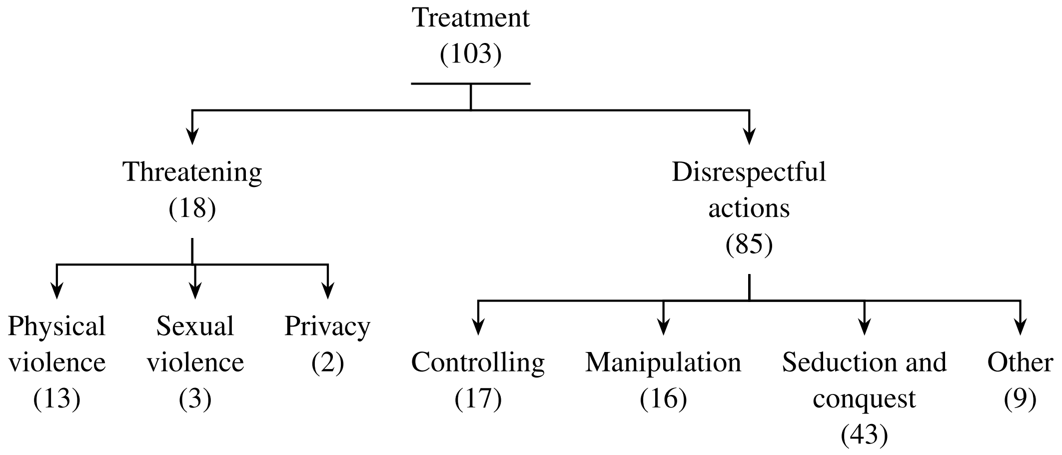


Figure 1: Prevalence of Misogynistic Treatment subcategories

Sexual violence and *Invasion of privacy* only have a couple of labels each (three and two, respectively).

6.3 Misogynistic derogation

The are 286 *Derogation* labels. Table 4 shows the number of labels for each subcategory within *Derogation*. The counts are broken down by the strength of the abuse (i.e. *implicit/explicit*). *Implicit derogation* is almost twice as common as *explicit* (182 vs 103). The most common subcategory is *Moral inferiority* which accounts for 51% of *implicit* and 54% of *explicit Derogation*. *Intellectual inferiority* then has equal numbers of *implicit* and *explicit* labels (n=16).

Subcategory	**Implicit**	**Explicit**
Moral infer.	92	56
Intellectual infer.	16	16
Sexual & physical lim.	14	12
Other	60	19
Total	**182**	**103**

Table 4: Breakdown of *Misogynistic Derogation* subcategories by implicit and explicit strength

6.4 Personal attacks

Table 5 shows the breakdown of both *Misogynistic* and *Nonmisogynistic* personal attacks. Slightly more than half (55%) of interpersonal abuse was *not* misogynistic. Of these women were still the target of the abuse almost four times as often as men (n=32 vs n=8). And women were as likely to receive misogynistic person attacks as non-misogynistic ones (n=32).

Gender	**Misog.**	**Nonmis.**	**Total**
Woman	32	32	64 (82%)
Man	2	8	10 (13%)
Unknown	1	3	4 (5%)
Total	35 (45%)	43 (55%)	78

Table 5: Breakdown of *Misogynistic* and *Nonmisogynistic* personal attacks by *Gender* of the target

The gender of the target was only *unknown* in 5% of cases, one misogynistic and three not. There were two cases of misogynistic abuse against men. All other misogynistic personal attacks were towards women.

6.5 Counter speech

There are only 10 cases of *Counter speech* in the final dataset of agreed labels. Annotators originally identified far more counter speech (188 labels for 149 unique entries were initially made) but few were accepted during the adjudication meetings. In Section 5.2 we showed that the category has one of the lowest Kappa values. Notably, 39% of original *Counter speech* labels were made by one annotator, showing that the annotators had different understandings of the threshold for *Counter speech*. However, the number of original labels for *Counter speech* decreased over the first few weeks of the annotation process, as shown in Figure 2. This reflects the complexity of the category; it took annotators time to differentiate content that was pro-women from that which actually countered previous misogynistic speech.

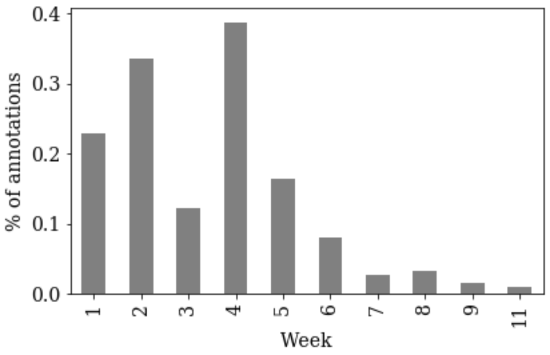


Figure 2: Percentage of original *counter speech* labels by week

7 Experiments

As reference points for further research using our dataset, we provide three experimental baselines on the binary task of distinguishing between misogynistic and non-misogynistic content, i.e. level one of our taxonomy. As the simplest baseline, we evaluate a logistic unigram classifier. Further, we evaluate two uncased BERT-base models (Devlin et al., 2019) – one unweighted, the other using class weights emphasising the minority class, i.e. misogynistic content, to account for class imbalance. For all models, we use the same stratified 80/20 train/test split of the dataset. Details on model training and parameters can be found in Appendix C.

Performance of the three models is shown in Table 6. All models perform poorly on misogynistic content, with the logistic classifier performing worst overall. The logistic classifier has the highest precision on misogynistic content (0.88) but very low recall (0.07) and a low F1 score (0.13). The weighted BERT model has the highest recall (0.50) and F1 score (0.43). Accuracy on all test cases, of which 91.9% are non-misogynistic, is around 0.90 across models.

The classification task is complicated by the relatively small size of our dataset (n=6,385 unique cases) as well as the relatively small proportion of misogynistic cases in it (8.1%). These issues are common in abusive speech detection (Fortuna and Nunes, 2018; Fortuna et al., 2020). To address them, future research can leverage the typology and annotation process we introduced to collect additional cases, particularly misogynistic ones, thus growing and balancing the dataset.

7.1 Error analysis

We make use of the more granular secondary labels in our taxonomy to conduct an error analysis for the weighted BERT model. Table 7 shows the confusion matrix for the 1,277 entries in the test set. Overall, 137 entries (10.7%) were misclassified. 86 (63%) of these errors were false positives and 51 (37%) were false negatives.

7.1.1 False positives

Of the 86 entries misclassified as *Misogynistic*, three are labelled as *Nonmisogynistic personal attacks*. An example is “Male feminist reacts to vile scumbag who murdered his kids by telling ALL men to, you know, stop killing women and kids...”. The use of aggressive language combined with gendered phrases (such as “stop killing women”) likely led to its misclassification.

The remaining 83 false positives fall under *None of the categories* and all contain some reference to women. Some refer to topics often associated with misogyny but are not misogynistic in themselves. For example, a comment in `r/seduction` stated, “the most manly thing is to find your dream woman, marry her, and live happily ever after. The constant sex with women is so overrated anyways”. This entry, which suggests that other things than high levels of sexual activity should be prioritised, is thematically similar to misogynistic content in the dataset.

Other false positives mention women indirectly. “Because they aren’t men, they are SIMPS”. ‘Simp’ is a pejorative term used in the manosphere for a man who cares too much about a woman. Under our taxonomy it did not count as a misogynistic pejorative but it is likely that the term appears in misogynistic entries in the dataset. Some false positives are critical of misogyny, though not actively enough to count as *Counter speech*. For example “Does this moid even know the meaning of the term ‘butterface’? If this woman is ugly, there is no hope for most of the female population.”. This discussion of unrealistic beauty standards of women references misogyny but is not itself misogynistic.

7.1.2 False negatives

Of the 51 Misogynistic entries the model misses, almost half (n=24) contain *Derogation*. Implicit and explicit derogation are missed at roughly similar rates, as are each of the subcategories. Importantly this shows that the different forms of derogation are no more or less likely to be missed.

Model	**Precision**	**Recall**	**F1 score**	**Accuracy**
Logistic regression	**0.88**	0.07	0.13	0.92
BERT (unweighted)	0.67	0.30	0.42	**0.93**
BERT (weighted)	0.38	**0.50**	**0.43**	0.89

Table 6: Model performance on misogynistic test cases (n=103) and accuracy on all test cases (n=1,277).

		Prediction		
		Nonmis.	Mis.	TOTAL
Label	Nonmis.	1,088	86	1,174
	Mis.	51	52	103
	TOTAL	1,139	138	1,277

Table 7: Confusion matrix of the weighted BERT model

In many cases, the derogation depends on the context of the earlier conversation thread, thus the BERT-model, which does not explicitly take into account prior entries in the thread, cannot recognise the misogyny in isolation. “It’s funny to see the hamster that starts to act up in their little widdle tiny brains after saying that too.” is an explicit statement that women are intellectually inferior, but understanding that it refers to women depends on having seen previous entries in the conversation.

The next most common source of false negatives is *Pejoratives* (n=19). The classifier misses six counts each of ‘whore’ and ‘stacy’ and five of ‘bitch’. There are seven missed labels for *Treatment*, five *Disrespectful actions* and two *Threatening language*. However, due to the low prevalence of the category in the training data we anticipate some errors. For example, “I am waiting for such incident to happen to me so that I can beat the shit out of her, and of course it will be all revenge” details a specific form of violence (i.e. ‘beat the shit out of her’) which the model cannot know to identify as misogyny without being trained on other uses of the term.

The final two errors are for *Personal attacks*. For example, “Yeah theres women that I as an Incel wouldnt even acknowledge and this is one of em [sic]”. This is an implicit attack which requires understanding that considering a woman unworthy of the attention of an incel is a gendered insult.

As we can see from these examples the main classification errors are due to context limitations. For false negatives there is usually not enough information in the entry alone or in the training dataset to identify the misogyny. Conversely, for false positives the classifier appears to overly associate content *about* women with content that *abuses* women. These limitations can be addressed by future work drawing on the taxonomy and annotation process presented here to develop larger datasets which can cover a greater range of forms of discourse, including both non-misogynistic discussions of women and a wider variety of misogynistic speech.

8 Conclusion

In this paper we have presented a hierarchical granular taxonomy for misogyny and have described a dataset containing high quality, expert labels of misogynistic content from Reddit. We have also provided the detailed coding book we created and a dataset with all of the original labels. The final dataset is small compared to other annotated datasets used for classification. However it benefits from a detailed taxonomy based on the existing literature focused on just one form of online abuse - misogyny. The use of trained annotators and an adjudication process also ensures the quality of the labels.

The more granular subcategories in the taxonomy may be too small to classify separately, but they provide insights into the relative frequency of different forms of misogynistic content on Reddit and enable detailed error analysis. They are also useful for other researchers aiming to create larger datasets, who can build on the taxonomic work conducted here.

Acknowledgments

This research was funded by Wave 1 of The UKRI Strategic Priorities Fund under the EPSRC Grant EP/T001569/1 at The Alan Turing Institute, particularly the “Tools, Practices and System” and “Criminal Justice” themes.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic Identification and Classification of Misogynistic Language on Twitter. In *Natural Language Processing and Information Systems*, Lecture Notes in Computer Science, pages 57–64, Cham. Springer International Publishing.

Emily M. Bender and Batya Friedman. 2018. Data Statements for Natural Language Processing: Toward Mitigating System Bias and Enabling Better Science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Bryce Boe. 2020. PRAW. Python Reddit API Wrapper Development.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017)*, page 4.

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

Maeve Duggan. 2017. Online Harassment 2017. Technical report, Pew Research Center.

Tracie Farrell, Miriam Fernandez, Jakub Novotny, and Harith Alani. 2019. Exploring Misogyny across the Manosphere in Reddit. In *WebSci '19 Proceedings of the 10th ACM Conference on Web Science*, pages 87–96, Boston.

E Fersini, P Rosso, and M Anzovino. 2018. Overview of the Task on Automatic Misogyny Identification at IberEval 2018. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) 215*, page 15.

Jill Filipovic. 2007. Blogging while female: How internet misogyny parallels real-world harassment. *Yale JL & Feminism*, 19:295.

Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys*, 51(4):85:1–85:30.

Paula Fortuna, Juan Soler, and Leo Wanner. 2020. Toxic, hateful, offensive or abusive? what are we really classifying? an empirical analysis of hate speech datasets. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6786–6794.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Debbie Ging. 2017. Alphas, Betas, and Incels: Theorizing the Masculinities of the Manosphere. *Men and Masculinities*, 1:20.

Debbie Ging, Theodore Lynn, and Pierangelo Rosati. 2019. Neologising misogyny: Urban Dictionary's folksonomies of sexual abuse:. *New Media & Society*.

Debbie Ging and Eugenia Siapera. 2018. Special issue on online misogyny. *Feminist Media Studies*, 18(4):515–524.

Debbie Ging and Eugenia Siapera, editors. 2019. *Gender Hate Online: Understanding the New Anti-Feminism*. Springer International Publishing, Cham.

Peter Glick and Susan T. Fiske. 1997. Hostile and Benevolent Sexism. *Psychology of Women Quarterly*, 21(1):119–135.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring Hate Speech Detection in Multimodal Publications. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1459–1467, Snowmass Village, CO, USA. IEEE.

Sarah Hewitt, T. Tiropanis, and C. Bokhove. 2016. The Problem of Identifying Misogynist Language on Twitter (and Other Online Social Spaces). In *Proceedings of the 8th ACM Conference on Web Science*, WebSci '16, pages 333–335, New York, NY, USA. ACM.

Amnesty International. 2017. Amnesty reveals alarming impact of online abuse against women. https://www.amnesty.org/en/latest/news/2017/11/amnesty-reveals-alarming-impact-of-online-abuse-against-women/.

Emma A. Jane. 2016. *Misogyny Online: A Short (and Brutish) History*. SAGE.

Emma Alice Jane. 2014. 'Back to the kitchen, cunt': Speaking the unspeakable about online misogyny. *Continuum*, 28(4):558–570.

Rebecca Jennings. 2018. Incels Categorize Women by Personal Style and Attractiveness. https://www.vox.com/2018/4/28/17290256/incel-chad-stacy-becky.

Akshita Jha and Radhika Mamidi. 2017. When does a compliment become sexist? Analysis and classification of ambivalent sexism using twitter data. In *Proceedings of the Second Workshop on NLP and Computational Social Science*, pages 7–16, Vancouver, Canada. Association for Computational Linguistics.

David Jurgens, Libby Hemphill, and Eshwar Chandrasekharan. 2019. A Just and Comprehensive Strategy for Using NLP to Address Online Abuse. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3658–3666, Florence, Italy. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Theo Lynn, Patricia Takako Endo, Pierangelo Rosati, Ivanovitch Silva, Guto Leoni Santos, and Debbie Ging. 2019a. A Comparison of Machine Learning Approaches for Detecting Misogynistic Speech in Urban Dictionary. In *2019 International Conference on Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, pages 1–8.

Theo Lynn, Patricia Takako Endo, Pierangelo Rosati, Ivanovitch Silva, Guto Leoni Santos, and Debbie Ging. 2019b. Data set for automatic detection of online misogynistic speech. *Data in Brief*, 26:104223.

Karla Mantilla. 2013. Gendertrolling: Misogyny Adapts to New Media. *Feminist Studies*, 39(2):563–570.

Adrienne Massanari. 2017. # Gamergate and The Fappening: How Reddit's algorithm, governance, and culture support toxic technocultures. *New Media & Society*, 19(3):329–346.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended Bias in Misogyny Detection. In *IEEE/WIC/ACM International Conference on Web Intelligence on - WI '19*, pages 149–155, Thessaloniki, Greece. ACM Press.

Bailey Poland. 2016. *Haters: Harassment, Abuse, and Violence Online*. Potomac Books.

Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. 2019. A Benchmark Dataset for Learning to Intervene in Online Hate Speech. *arXiv:1909.04251 [cs]*.

James Ravenscroft, Anika Oellrich, Shyamasree Saha, and Maria Liakata. 2016. Multi-label Annotation in Scientific Articles - The Multi-label Cancer Risk Assessment Corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4115–4123, Portorož, Slovenia. European Language Resources Association (ELRA).

Manoel Horta Ribeiro, Jeremy Blackburn, Barry Bradlyn, Emiliano De Cristofaro, Gianluca Stringhini, Summer Long, Stephanie Greenberg, and Savvas Zannettou. 2020. From Pick-Up Artists to Incels: A Data-Driven Sketch of the Manosphere. *arXiv:2001.07600 [cs]*.

Manuela Sanguinetti, Fabio Poletto, Cristina Bosco, Viviana Patti, and Marco Stranisci. 2018. An Italian Twitter Corpus of Hate Speech against Immigrants. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The Risk of Racial Bias in Hate Speech Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, Valencia, Spain. Association for Computational Linguistics.

Bertie Vidgen, Alex Harris, Dong Nguyen, Rebekah Tromble, Scott Hale, and Helen Margetts. 2019. Challenges and frontiers in abusive content detection. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 80–93, Florence, Italy. Association for Computational Linguistics.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: The Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In

Proceedings of the 26th International Conference on World Wide Web, pages 1391–1399, Perth Australia. International World Wide Web Conferences Steering Committee.

Donna Zuckerberg. 2018. *Not All Dead White Men: Classics and Misogyny in the Digital Age*. Harvard University Press, Cambridge, Massachusetts.

A Short form data statement

Following the recommendation of Bender and Friedman (2018) we include the following short form data statement to summarise the main features of the datasets. Further details on the creation of the datasets are in in Sections 3 and 5 in the main paper.

A.1 Data

The two datasets include labels for 6,383 unique Reddit entries (i.e. posts or comments) across 672 conversation threads collected. One dataset is of the 15,816 original labels selected by annotators and the second is of the 6,567 agreed labels. Table 8 provides a description of each of the variables in the datasets. We also include the accompanying set of images associated with some original post entries.

All threads except one are in English. The majority of threads were sampled from a set of 34 subreddits selected for the expected prevalence of misogynistic content, or non-misogynistic discussions about women. Paid annotators received extensive training to apply the taxonomy presented in this paper to label entries. The majority of annotators were White-British, spoke English as a first language, and had or were pursuing a University degree. Two-thirds of annotators were women.

A.2 Annotators

All annotators were based in the United Kingdom and worked remotely. They were paid £14 per hour for all work including training. Five of the six annotators gave permission to share their basic demographic information. All were between 18 and 29 years old. Two had high school degrees, two had an undergraduate degree, and one had a postgraduate taught degree or equivalent. Four identified as women, one as a man. All were British nationals, native English speakers, and identified as ethnically white.

All annotators used social media at least once per day. Two had never been personally targeted by online abuse, two had been targeted 2-3 times (in separate instances more than a year ago), and one had been personally targeted more than 3 times within the previous month.

B Frequency of targeted subreddits

Table 9 lists the subreddits used for target sampling of data. The columns *Num entries* and *Num threads* state how many individual entries and threads from each subreddit are in the datasets. The column *Selection* shows whether the subreddit was identified from existing literature, which is cited, or using snowball sampling.

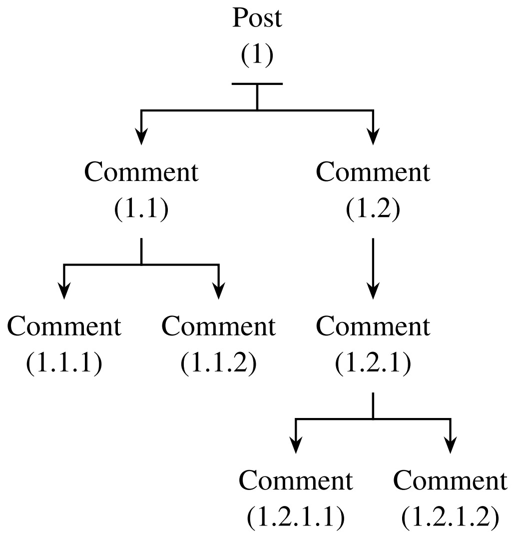


Figure 3: Tree diagram for comment order in threads

C Model Details

Pre-Processing We lowercase all text and remove newline and tab characters. URLs and emojis are replaced with [URL] and [EMOJI] tokens.

C.1 Logistic regression

Logistic regression with l1-regularisation is implemented in R using the 'glmnet' package (Friedman et al., 2010) on a unigram representation of the data. Lambda is selected using cross-validation and set to 0.015.

C.2 BERT Models (weighted/unweighted)

Model Architecture We implement uncased BERT-base models (Devlin et al., 2019) using the `transformers` Python library (Wolf et al., 2020). For sequence classification, we add a linear layer with softmax output.

Training Parameters We apply a stratified 80/20 train/test split to our dataset. Models are trained for three epochs each. Training batch size is 16. We use cross-entropy loss. For the weighted model, we add class weights emphasising the minority class, i.e. misogynistic content. Weights are set to the relative proportion of the other class in the training data, meaning that for a 1:9 misogynistic:non-misogynistic case split, loss on misogynistic cases would be multiplied by 9. The optimiser is AdamW (Loshchilov and Hutter, 2018) with a 5e-5 learning rate and a 0.01 weight decay. For regularisation, we set a 10% dropout probability.

Variable	**Description**
entry_id	A unique string assigned to every comment and post by Reddit.
link_id	The id number of the original post of a thread.
parent_id	The id number of parent entry (i.e. the post or comment this entry responds to).
subreddit	The subreddit community where the entry was made.
author	The Reddit username of the entry author.
body	The text body of the entry. For the original posts of threads the title and post body were combined.
image	Whether the entry has an accompanying image. Only applicable to posts. Images are provided as jpg files. They are named as 'X_Y_Z' corresponding to the week (X), group (Y), and thread id (Z).
label_date	The week commencing date of when the entry was labelled.
week	The week in the annotation process when the entry as assigned (1 to 11).
group	The weekly group the entry was assigned to. All weeks had two groups except week 7 which only had 1.
sheet_order	The order of the entry in the weekly annotation sheet. This is a list of numbers referring to the nested structure of comments in threads. It shows the id number of each level of the thread from the original post to the relevant entry. For example, if an entry has the sheet_order (1, 2, 3) it belongs to the first thread (1), and replied to the second comment (2), to which it is the third reply (3). See Fig. 3 for visual explanation.
annotator_id	The id number of the annotator who made the annotation (1 to 6). Only applicable to the original_labels dataset.
level_1	Whether the entry is *Misogynistic* or *Nonmisogynistic*.
level_2	The category of the label (i.e. *Pejoratives, Derogation*, etc.
level_3	EITHER the subcategory for *Derogation* or *Treatment* OR the gender of the target for either *Personal attack* category. Empty for all other categories.
strength	Whether the abuse is implicit or explicit. Only applicable to identity directed abuse.
highlight	The highlighted part of the entry's body which contains the abuse. Mandatory for all primary categories except 'None'.
split	Whether the entry was included in the 'train' or 'test' dataset split for model building. Only applicable to the final_labels dataset.

Table 8: Description of dataset variables

Subreddit	Num entries	Num threads	Selection
altTRP	2	1	Snowball
AskFeminists	263	26	Snowball
askseddit	142	16	Snowball
badwomensanatomy	430	31	Farrell et al. (2019)
becomeaman	2	1	Snowball
Egalitarianism	115	15	Snowball
exredpill	113	12	Snowball
FeMRADebates	195	20	Snowball
GEOTRP	11	1	Snowball
IncelsInAction	110	14	Farrell et al. (2019)
IncelsWithoutHate	325	28	Farrell et al. (2019)
KotakuInAction	373	28	Qian et al. (2019); Zuckerberg (2018)
marriedredpill	87	7	Snowball
masculism	34	5	Snowball
MensRants	4	1	Ging (2017)
MensRights	364	29	Ging (2017); Qian et al. (2019); Zuckerberg (2018)
mensrightslaw	2	1	Snowball
MensRightsMeta	4	1	Snowball
MGTOW	601	41	Farrell et al. (2019); Ging (2017); Qian et al. (2019); Zuckerberg (2018)
mgtowbooks	2	1	Snowball
MRActivism	8	2	Snowball
NOMAAM	2	1	Snowball
pua	10	1	Snowball
PurplePillDebate	221	21	Snowball
PussyPass	344	33	Qian et al. (2019)
pussypassdenied	262	22	Qian et al. (2019)
RedPillParenting	12	2	Snowball
RedPillWives	61	8	Snowball
RedPillWomen	217	23	Snowball
seduction	392	33	Zuckerberg (2018)
ThankTRP	8	1	Snowball
TheRedPill	338	29	Ging (2017); Zuckerberg (2018)
theredpillright	10	1	Snowball
Trufemcels	434	37	Farrell et al. (2019)

Table 9: Number of entries and threads per targeted subreddit

WikiMatrix: Mining 135M Parallel Sentences in 1620 Language Pairs from Wikipedia

Holger Schwenk Facebook AI schwenk@fb.com

Vishrav Chaudhary Facebook AI vishrav@fb.com

Shuo Sun Johns Hopkins University ssun32@jhu.edu

Hongyu Gong Univ. of Illinois Urbana-Champaign hgong6@illinois.edu

Francisco Guzmán Facebook AI fguzman@fb.com

Abstract

We present an approach based on multilingual sentence embeddings to automatically extract parallel sentences from the content of Wikipedia articles in 96 languages, including several dialects or low-resource languages. We systematically consider all possible language pairs. In total, we are able to extract 135M parallel sentences for 1620 different language pairs, out of which only 34M are aligned with English. This corpus is freely available.[1]

To get an indication on the quality of the extracted bitexts, we train neural MT baseline systems on the mined data only for 1886 languages pairs, and evaluate them on the TED corpus, achieving strong BLEU scores for many language pairs. The WikiMatrix bitexts seem to be particularly interesting to train MT systems between distant languages without the need to pivot through English.

1 Introduction

Most of the current approaches in Natural Language Processing are data-driven. The size of the resources used for training is often the primary concern, but the quality and a large variety of topics may be equally important. Monolingual texts are usually available in huge amounts for many topics and languages. However, multilingual resources, i.e. sentences which are mutual translations, are more limited, in particular when the two languages do not involve English. An important source of parallel texts is from international organizations like the European Parliament (Koehn, 2005) or the United Nations (Ziemski et al., 2016). Several projects rely on volunteers to provide translations for public texts, e.g. news commentary (Tiedemann, 2012), OpensubTitles (Lison and Tiedemann, 2016) or the TED corpus (Qi et al., 2018)

Wikipedia is probably the largest free multilingual resource on the Internet. The content of Wikipedia is very diverse and covers many topics. Articles exist in more than 300 languages. Some content on Wikipedia was human translated from an existing article into another language, not necessarily from or into English. Eventually, the translated articles have been later independently edited and are not parallel anymore. Wikipedia strongly discourages the use of unedited machine translation,[2] but the existence of such articles cannot be totally excluded. Many articles have been written independently, but may nevertheless contain sentences which are mutual translations. This makes Wikipedia a very appropriate resource to mine for parallel texts for a large number of language pairs. To the best of our knowledge, this is the first work to process the entire Wikipedia and systematically mine for parallel sentences in all language pairs.

In this work, we build on a recent approach to mine parallel texts based on a distance measure in a joint multilingual sentence embedding space (Schwenk, 2018; Artetxe and Schwenk, 2018a), and a freely available encoder for 93 languages. We approach the computational challenge to mine in almost six hundred million sentences by using fast indexing and similarity search algorithms.

The paper is organized as follows. In the next section, we first discuss related work. We then summarize the underlying mining approach. Section 4 describes in detail how we applied this approach to extract parallel sentences from Wikipedia in 1620 language pairs. In section 5, we assess the quality of the extracted bitexts by training NMT systems for a subset of language pairs and evaluate them on the TED corpus (Qi et al., 2018) for 45 languages. The paper concludes with a discussion of future research directions.

---

[1] `https://github.com/facebookresearch/LASER/tree/master/tasks/WikiMatrix`

[2] `https://en.wikipedia.org/wiki/Wikipedia:Translation`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1351–1361
April 19 - 23, 2021.

2 Related work

There is a large body of research on mining parallel sentences in monolingual texts collections, usually named *"comparable coprora"*. Initial approaches to bitext mining have relied on heavily engineered systems often based on metadata information, e.g. (Resnik, 1999; Resnik and Smith, 2003). More recent methods explore the textual content of the comparable documents. For instance, it was proposed to rely on cross-lingual document retrieval, e.g. (Utiyama and Isahara, 2003; Munteanu and Marcu, 2005) or machine translation, e.g. (Abdul-Rauf and Schwenk, 2009; Bouamor and Sajjad, 2018), typically to obtain an initial alignment that is then further filtered. In the shared task for bilingual document alignment (Buck and Koehn, 2016), many participants used techniques based on n-gram or neural language models, neural translation models and bag-of-words lexical translation probabilities for scoring candidate document pairs. The STACC method uses seed lexical translations induced from IBM alignments, which are combined with set expansion operations to score translation candidates through the Jaccard similarity coefficient (Etchegoyhen and Azpeitia, 2016; Azpeitia et al., 2017, 2018). Using multilingual noisy webcrawls such as ParaCrawl[3] for filtering good quality sentence pairs has been explored in the shared tasks for high resource (Koehn et al., 2018) and low resource (Koehn et al., 2019) languages.

In this work, we rely on massively multilingual sentence embeddings and margin-based mining in the joint embedding space, as described in (Schwenk, 2018; Artetxe and Schwenk, 2018a,b). This approach has also proven to perform best in a low resource scenario (Chaudhary et al., 2019; Koehn et al., 2019). Closest to this approach is the research described in España-Bonet et al. (2017); Hassan et al. (2018); Guo et al. (2018); Yang et al. (2019). However, in all these works, only bilingual sentence representations have been trained. Such an approach does not scale to many languages, in particular when considering all possible language pairs in Wikipedia. Finally, related ideas have been also proposed in Bouamor and Sajjad (2018) or Grégoire and Langlais (2017). However, in those works, mining is not solely based on multilingual sentence embeddings, but they are part of a larger system. To the best of our knowledge, this work is the first one that applies the same mining approach to all combinations of many different languages, written in more than twenty different scripts. In follow up work, the same underlying mining approach was applied to a huge collection of Common Crawl texts (Schwenk et al., 2019). Hierarchical mining in Common Crawl texts was performed by El-Kishky et al. (2020).[4]

Wikipedia is arguably the largest comparable corpus. One of the first attempts to exploit this resource was performed by Adafre and de Rijke (2006). An MT system was used to translate Dutch sentences into English and to compare them with the English texts, yielding several hundreds of Dutch/English bitexts. Later, a similar technique was applied to Persian/English (Mohammadi and GhasemAghaee, 2010). Structural information in Wikipedia such as the topic categories of documents was used in the alignment of multilingual corpora (Otero and López, 2010). In another work, the mining approach of Munteanu and Marcu (2005) was applied to extract large corpora from Wikipedia in sixteen languages (Smith et al., 2010). Otero et al. (2011) measured the comparability of Wikipedia corpora by the translation equivalents on three languages Portuguese, Spanish, and English. Patry and Langlais (2011) came up with a set of features such as Wikipedia entities to recognize parallel documents, and their approach was limited to a bilingual setting. Tufis et al. (2013) proposed an approach to mine bitexts from Wikipedia textual content, but they only considered high-resource languages, namely German, Spanish and Romanian paired with English. Tsai and Roth (2016) grounded multilingual mentions to English Wikipedia by training cross-lingual embeddings on twelve languages. Gottschalk and Demidova (2017) searched for parallel text passages in Wikipedia by comparing their named entities and time expressions. Finally, Aghaebrahimian (2018) propose an approach based on bilingual BiLSTM sentence encoders to mine German, French and Persian parallel texts with English. Parallel data consisting of aligned Wikipedia titles have been extracted for twenty-three languages.[5] We are not aware of other attempts to systematically mine for parallel sentences in the textual content of Wikipedia for a large number of languages.

---

[3] `http://www.paracrawl.eu/`

[4] `http://www.statmt.org/cc-aligned/`

[5] `https://linguatools.org/tools/corpora/wikipedia-parallel-titles-corpora/`

3 Distance-based mining approach

The underlying idea of the mining approach used in this work is to first learn a multilingual sentence embedding. The distance in that space can be used as an indicator of whether two sentences are mutual translations or not. Using a simple absolute threshold on the cosine distance was shown to achieve competitive results (Schwenk, 2018). However, it has been observed that an absolute threshold on the cosine distance is globally not consistent, e.g. (Guo et al., 2018). This is particularly true when mining bitexts for many different language pairs.

3.1 Margin criterion

The alignment quality can be substantially improved by using a margin criterion (Artetxe and Schwenk, 2018a). The margin between two candidate sentences x and y is defined as the ratio between the cosine distance between the two sentence embeddings, and the average cosine similarity of its nearest neighbors in both directions:

$$M(x,y) = \frac{\cos(x,y)}{\sum\limits_{z \in \mathrm{NN}_k(x)} \frac{\cos(x,z)}{2k} + \sum\limits_{z \in \mathrm{NN}_k(y)} \frac{\cos(y,z)}{2k}} \tag{1}$$

where $\mathrm{NN}_k(x)$ denotes the k unique nearest neighbors of x in the other language, and analogously for $\mathrm{NN}_k(y)$. We used $k = 4$ in all experiments.

We follow the *"max"* strategy of Artetxe and Schwenk (2018a): the margin is first calculated in both directions for all sentences in language L_1 and L_2. We then create the union of these forward and backward candidates. Candidates are sorted and pairs with source or target sentences that were already used are omitted. We then apply a threshold on the margin score to decide whether two sentences are mutual translations or not.

The complexity of a distance-based mining approach is $O(N \times M)$, where N and M are the number of sentences in each monolingual corpus. This makes a brute-force approach with exhaustive distance calculations intractable for large corpora. The languages with the largest Wikipedia are English and German with 134M and 51M sentences, respectively. This would require 6.8×10^{15} distance calculations.[6] We show in Section 3.3 how to tackle this computational challenge.

3.2 Multilingual sentence embeddings

Distance-based bitext mining requires a joint sentence embedding for all the considered languages. One may be tempted to train a bi-lingual embedding for each language pair, e.g. (España-Bonet et al., 2017; Hassan et al., 2018; Guo et al., 2018; Yang et al., 2019), but this is difficult to scale to thousands of language pairs present in Wikipedia. Instead, we chose to use one single massively multilingual sentence embedding for all languages, namely the one proposed by the open-source LASER toolkit (Artetxe and Schwenk, 2018b). Training one joint multilingual embedding on many languages at once also has the advantage that low-resource languages can benefit from the similarity to other languages in the same language family. For example, we were able to mine parallel data for several Romance (minority) languages like Aragonese, Lombard, Mirandese or Sicilian although data in those languages was not used to train the multilingual LASER embeddings. The reader is referred to Artetxe and Schwenk (2018b) for a detailed description how LASER was trained.

3.3 Fast similarity search

In this work, we use the open-source FAISS library[7] which implements highly efficient algorithms to perform similarity search on billions of vectors (Johnson et al., 2017). Our sentence representations being 1024-dimensional, all English sentences require $134{\cdot}10^6{\times}1024{\times}4{=}536$ GB of memory. Therefore, dimensionality reduction and data compression are needed for efficient search. We chose a rather aggressive compression based on a 64-bit product-quantizer (Jégou et al., 2011), and partitioning the search space in 32k cells.[8] We build one FAISS index for each language.

The compressed FAISS index for English requires only 9.2GB, i.e. more than fifty times smaller than the original sentences embeddings. This makes it possible to load the whole index on a standard GPU and to run the search in a very efficient way on multiple GPUs in parallel, without the need to shard the index. The overall mining process for German/English requires less than 3.5 hours on 8 GPUs, including the nearest neighbor search in both direction and scoring all candidates.

---

[6] Strictly speaking, Cebuano and Swedish are larger than German, yet mostly consist of template/machine translated text

[7] https://github.com/facebookresearch/faiss

[8] FAISS index type `OPQ64,IVF32768,PQ64`, see https://github.com/facebookresearch/faiss/wiki/Faiss-indexes

4 Bitext mining in Wikipedia

For each Wikipedia article, it is possible to get the link to the corresponding article in other languages. This could be used to mine sentences limited to the respective articles. On one hand, this **local mining** has several advantages: 1) mining is very fast since each article usually has a few hundreds of sentences only; 2) it seems reasonable to assume that a translation of a sentence is more likely to be found in the same article than anywhere in the whole Wikipedia. On the other hand, we hypothesize that the margin criterion will be less efficient since one article usually has few sentences which are similar. This may lead to many sentences in the overall mined corpus of the type *"NAME was born on DATE in CITY"*, *"BUILDING is a monument in CITY built on DATE"*, etc. Although those alignments may be correct, we hypothesize that they are of limited use to train an NMT system.

The other option is to consider the whole Wikipedia for each language: for each sentence in the source language, we mine in all target sentences. This **global mining** has the advantages that we can try to align two languages even though there are only a few articles in common. A drawback is a potentially increased risk of misalignment. In this work, we chose the global mining option.

4.1 Corpus preparation

Extracting the textual content of Wikipedia articles in all languages is a rather challenging task, i.e. removing all tables, citations, footnotes or formatting markup. There are several ways to download Wikipedia content. In this study, we use the so-called `CirrusSearch dumps` since they directly provide the textual content without any meta information.[9] We downloaded this dump in March 2019. A total of about 300 languages are available, but the size obviously varies a lot between languages. We applied the following processing: 1) extract the textual content; 2) split the paragraphs into sentences; 3) remove duplicate sentences; and 4) perform language identification and remove sentences which are not in the expected language.

It should be pointed out that sentence segmentation is not a trivial task. Some languages do not use specific symbols to mark the end of a sentence, namely Thai. We are not aware of a freely available sentence segmenter for Thai and we had to exclude it. We used a freely available Python tool[10] to detect sentence boundaries. Regular expressions were used for most of the Asian languages, falling back to English for the remaining languages. This gives us 879 million sentences in 300 languages. The margin criterion to mine for parallel data requires that the texts do not contain duplicates. This removes about 25% of the sentences.[11]

L_1 (French)	*Ceci est une très grande maison*
L_2 (German)	*Das ist ein sehr großes Haus* *This is a very big house* *Ez egy nagyon nagy ház* *Ini rumah yang sangat besar*

Table 1: Illustration how sentences in the wrong language can hurt the alignment process with a margin criterion. See text for a detailed discussion.

LASER's sentence embeddings are totally language agnostic. This has the side effect that the sentences in other languages (e.g. citations or quotes) may be considered closer in the embedding space than a potential translation in the target language. Table 1 illustrates this problem. The algorithm would not select the German sentence although it is a perfect translation. The sentences in the other languages are also valid translations which would yield a very small margin. To avoid this problem, we perform language identification (LID) on all sentences and remove those which are not in the expected language. LID is performed with fasttext[12] (Joulin et al., 2016). Fasttext does not support all the 300 languages present in Wikipedia and we disregarded the missing ones (which typically have only a few sentences anyway). After deduplication and LID, we dispose of 595M sentences in 182 languages. English accounts for 134M sentences, and German with 51M sentences is the second largest language. The sizes for all languages are given in Tables 3 and 5 (in the appendix).

4.2 Threshold optimization

Artetxe and Schwenk (2018a) optimized their mining approach for each language pair on a provided corpus of gold alignments. This is not possible when mining Wikipedia, in particular when con-

---

[9] `https://dumps.wikimedia.org/other/cirrussearch/`

[10] `https://pypi.org/project/sentence-splitter/`

[11] The Cebuano and Waray Wikipedia were largely created by a bot and contain more than 65% of duplicates.

[12] `https://fasttext.cc/docs/en/language-identification.html`

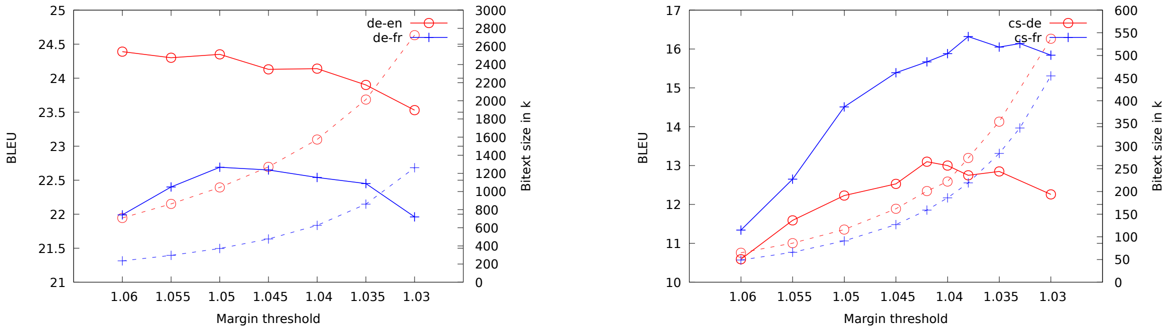


Figure 1: BLEU scores (continuous lines) for several NMT systems trained on bitexts extracted from Wikipedia for different margin thresholds. The size of the mined bitexts are depicted as dashed lines.

sidering many language pairs. In this work, we use an evaluation protocol inspired by the WMT shared task on parallel corpus filtering for low-resource conditions (Koehn et al., 2019): an NMT system is trained on the extracted bitexts – for different thresholds – and the resulting BLEU scores are compared. We choose `newstest2014` of the WMT evaluations since it provides an N-way parallel test sets for English, French, German and Czech. We favoured the translation between two morphologically rich languages from different families and considered the following language pairs: German/English, German/French, Czech/German and Czech/French. The size of the mined bitexts is in the range of 100k to more than 2M (see Table 2 and Figure 1). We did not try to optimize the architecture of the NMT system to the size of the bitexts and used the same architecture for all systems: the encoder and decoder are 5-layer transformer models as implemented in `fairseq` (Ott et al., 2019). The goal of this study is not to develop the best performing NMT system for the considered languages pairs, but to compare different mining parameters.

The evolution of the BLEU score in function of the margin threshold is given in Figure 1. Decreasing the threshold naturally leads to more mined data – we observe an exponential increase of the data size. The performance of the NMT systems trained on the mined data seems to change as expected: the BLEU score first improves with increasing amounts of available training data, reaches a maximum and than decreases since the additional data gets more and more noisy, i.e. contains wrong translations. It is also not surprising that a careful choice of the margin threshold is more important in a low-resource setting. Every additional parallel sentence is important. According to Figure 1, the optimal value of the margin threshold seems to be 1.05 when many sentences can be extracted, in our case German/English and German/French. When less parallel data is available, i.e. Czech/German and Czech/French, a value in the range of 1.03–1.04 seems to be a better choice. Aiming at one threshold for all language pairs, we chose a value of 1.04. It seems to be a good compromise for most language pairs. However, for the open release of this corpus, we provide all mined sentence with a margin of 1.02 or better. This enables end users to choose an optimal threshold for their particular applications. However, it should be emphasized that we do not expect that many sentence pairs with a margin as low as 1.02 are good translations.

Bitexts	**de-en**	**de-fr**	**cs-de**	**cs-fr**
Europarl	1.9M 21.5	1.9M 23.6	568k 14.9	627k 21.5
	1.0M 21.2	370k 21.1	200k 12.6	220k 19.2
Mined Wikipedia	1.0M 24.4	372k 22.7	201k 13.1	219k 16.3
Europarl + Wikipedia	3.0M 25.5	2.3M *25.6*	768k 17.7	846k 24.0

Table 2: Comparison of NMT systems trained on the Europarl corpus and on bitexts automatically mined in Wikipedia by our approach at a threshold of 1.04. We give the number of sentences (first line) and the BLEU score (second line of each bloc) on `newstest2014`.

For comparison, we also trained NMT systems on the Europarl corpus V7 (Koehn, 2005), i.e. professional human translations, first on all available data, and then on the same number of sentences as the mined ones (see Table 2). With the exception of Czech/French, we were able to achieve better

BLEU scores with the mined bitexts in Wikipedia than with Europarl of the same size. Adding the mined bitexts to the full Europarl corpus, leads to further improvements of 1.1 to 3.1 BLEU.

5 Result analysis

We run the alignment process for all possible combinations of languages in Wikipedia. This yielded 1620 language pairs for which we were able to mine at least ten thousand sentences. Remember that mining $L_1 \rightarrow L_2$ is identical to $L_2 \rightarrow L_1$, and is counted only once. We propose to analyze and evaluate the extracted bitexts in two ways. First, we discuss the amount of extracted sentences (Section 5.1). We then turn to a qualitative assessment by training NMT systems for all language pairs with more than twenty-five thousand mined sentences (Section 5.2).

5.1 Quantitative analysis

Due to space limits, Table 3 summarizes the number of extracted parallel sentences only for languages which have a total of at least five hundred thousand parallel sentences (with all other languages at a margin threshold of 1.04). Additional results are given in Table 5 in the Appendix.

There are many reasons which can influence the number of mined sentences. Obviously, the larger the monolingual texts, the more likely it is to mine many parallel sentences. Not surprisingly, we observe that more sentences could be mined when English is one of the two languages. Let us point out some languages for which it is usually not obvious to find parallel data with English, namely Indonesian (1M), Hebrew (545k), Farsi (303k) or Marathi (124k sentences). The largest mined texts not involving English are Russian/Ukrainian (2.5M), Catalan/Spanish (1.6M), or between the Romance languages French, Spanish, Italian and Portuguese (480k–923k), and German/French (626k).

It is striking to see that we were able to mine more sentences when Galician and Catalan are paired with Spanish than with English. On one hand, this could be explained by the fact that LASER's multilingual sentence embeddings may be better since the involved languages are linguistically very similar. On the other, it could be that the Wikipedia articles in both languages share a lot of content, or are obtained by mutual translation.

Services from the European Commission provide human translations of (legal) texts in all the 24 official languages of the European Union. This N-way parallel corpus enables training of MT system to directly translate between these languages, without the need to pivot through English. This is usually not the case when translating between other major languages, for example in Asia. Some interesting language pairs for which we were able to mine more than 100k sentences include: Korean/Japanese (222k), Russian/Japanese (196k), Indonesian/Vietnamese (146k), or Hebrew/Romance languages (120–150k sentences).

Overall, we were able to extract at least ten thousand parallel sentences for 96 different languages. For several low-resource languages, we were able to extract more parallel sentences with other languages than English. These include, among others, Aragonse with Spanish, Lombard with Italian, Breton with several Romance languages, Western Frisian with Dutch, Luxembourgish with German or Egyptian Arabic and Wu Chinese with the respective major language.

Finally, Cebuano (ceb) falls clearly apart: it has a rather huge Wikipedia (17.9M filtered sentences), but most of it was generated by a bot, as for the Waray language.[13] This certainly explains that only a very small number of parallel sentences could be extracted. Although the same bot was also used to generate articles in the Swedish Wikipedia, our alignments seem to be better for that language.

5.2 Qualitative evaluation

Aiming to perform a large-scale assessment of the quality of the extracted parallel sentences, we trained NMT systems on the bitexts. We identified a publicly available dataset which provides test sets for many language pairs: translations of TED talks as proposed in the context of a study on pretrained word embeddings for NMT[14] (Qi et al., 2018). We would like to emphasize that we did not use the training data provided by TED – we only trained on the mined sentences from Wikipedia. The goal of this study is not to build state-of-the-art NMT system for for the TED task, but to get an estimate of the quality of our extracted data, for many language pairs. In particular, there may be a mismatch in the topic and language style between Wikipedia texts and the transcribed and translated TED talks.

NMT systems are trained with a transformer model from `fairseq` (Ott et al., 2019) with the

---

[13] `https://en.wikipedia.org/wiki/Lsjbot`

[14] `https://github.com/neulab/word-embeddings-for-nmt`

ISO	Name	Lang. Family	size	az	ba	be	bg	bn	bs	ca	cs	da	de	el	en	eo	es	et	eu	fa	fi	fr	gl	he	hi	hr	hu	id	is	it	ja	kk	ko	lt	mk	ml	mr	ne	nl	no	oc	pl	pt	ro	ru	sh	si	sk	sl	sq	sr	sv	sw	ta	te	tl	tr	tt	uk	vi	zh	total
ar	Arabic	Arabic	**6516**	17	15	11	54	40	34	94	67	53	99	66	999	37	174	40	24	58	53	163	50	68	38	38	60	90	18	123	83	11	48	33	52	32	32	12	73	58	9	74	157	71	125	35	32	32	39	30	49	58	13	27	27	18	69	14	70	93	86	**4435**
az	Azerbaijani	Turkic	**1873**		7	3	14	8	9	15	17	11	34	10	71	9	31	10	7	16	14	29	10	12	9	10	15	17	4	25	23	6	11	9	8	5	4	2	22	13	2	22	25	14	47	9	3	9	8	9	10	19	5	11	5	2	42	8	19	10	19	**885**
ba	Bashkir	Turkic	**536**			2	14	4	9	17	16	13	27	10	28	9	28	8	6	9	12	29	12	9	4	10	12	12	3	26	13	4	5	7	9	1	1	1	21	15	3	19	24	15	42	10	1	10	10	7	10	20	4	3	1	1	11	8	15	7	10	**697**
be	Belarusian	Slavic	**1690**				16	4	7	16	14	9	20	8	33	9	28	7	5	5	10	24	9	10	3	9	11	9	3	24	12	2	5	7	8	2	2	1	14	10	2	19	23	13	161	8	2	8	7	5	10	16	2	3	2	1	8	2	80	8	8	**803**
bg	Bulgarian	Slavic	**3327**					38	34	76	79	53	132	62	357	40	122	43	25	37	61	117	43	58	30	47	68	60	17	102	71	11	38	42	86	29	35	13	84	58	9	96	114	69	270	41	31	43	46	26	65	63	12	21	23	21	56	12	126	60	60	**3770**
bn	Bengali	Indo-Aryan	**1412**						21	41	47	33	70	36	280	27	81	26	14	20	37	68	27	34	21	23	41	36	8	64	38	4	20	21	23	7	8	3	50	35	4	52	76	46	62	20	8	25	26	17	25	54	6	12	8	4	33		37	31	31	**1991**
bs	Bosnian	Slavic	**1060**							44	43	32	71	33	210	24	70	25	16	20	36	60	32	30	16	164	39	38	11	52	36	6	22	23	39	19	20	8	45	36	6	48	62	37	59	178	16	25	34	19	130	38	7	15	12	13	33	8	39	38	31	**2348**
ca	Catalan	Romance	**8332**								100	86	180	90	1205	81	1580	54	77	44	83	490	268	84	37	57	92	107	23	316	103	12	52	45	61	45	56	17	144	102	57	121	358	110	169	52	52	50	57	34	67	102	14	30	35	31	77	16	98	106	90	**8047**
cs	Czech	Slavic	**7634**									75	233	70	519	75	181	62	36	45	95	185	54	72	38	63	105	78	23	161	105	15	53	55	51	36	41	15	139	86	11	176	153	82	186	50	37	474	64	30	63	97	15	31	34	25	75	14	104	74	80	**5325**
da	Danish	Germanic	**2881**										180	54	436	39	140	45	26	29	75	142	44	55	25	43	69	63	20	115	76	9	37	35	39	30	35	12	110	303	8	89	123	70	109	37	32	39	40	23	43	168	11	20	21	23	55	11	62	68	57	**4006**
de	German	Germanic	**50944**											95	1573	186	418	106	53	66	163	626	80	109	57	87	192	107	34	388	217	23	82	78	64	51	58	21	472	207	17	285	294	129	368	68	50	94	106	51	81	216	20	58	57	32	127	23	165	107	134	**9692**
el	Greek	Hellenic	**3211**												620	39	145	41	23	35	55	137	48	56	26	43	64	73	15	119	69	6	35	34	52	27	32	9	76	60	8	77	144	78	114	38	31	35	46	28	52	62	11	16	18	20	56	9	68	75	62	**3728**
en	English	Germanic	**134431**													298	3377	243	119	303	375	2757	446	545	231	259	488	1019	85	2126	851	20	306	157	395	71	124	15	796	636	37	668	2461	631	1661	224	115	178	318	180	395	546	51	95	91	75	477	32	681	1073	786	**34321**
eo	Esperanto	constructed	**2371**														149	31	25	23	46	134	46	39	22	29	57	46	15	101	48	7	26	28	30	28	28	9	81	47	9	77	91	43	81	26	28	41	32	19	36	53	8	16	19	17	37	8	50	42	39	**2817**
es	Spanish	Romance	**25202**															89	154	83	155	905	610	153	71	94	167	198	42	671	219	26	108	76	92	65	98	25	272	181	35	235	923	183	393	81	84	81	93	53	107	181	21	57	71	48	147	26	187	206	174	**14980**
et	Estonian	Uralic	**2303**																22	24	70	85	32	39	20	33	56	41	14	75	57	6	29	35	32	20	21	8	72	49	7	73	76	48	96	27	19	34	35	18	34	58	9	16	16	15	43	7	56	40	44	**2511**
eu	Basque	Isolate	**2259**																	14	33	65	43	25	13	21	35	27	10	54	33	5	18	19	19	11	10	5	44	29	6	43	58	30	47	19	9	20	20	12	21	38	6	13	11	8	29	5	30	25	23	**1699**
fa	Farsi	Iranian	**3954**																		34	71	25	36	20	24	39	46	9	64	46	8	26	20	27	11	10	6	49	32	4	50	77	40	72	21	9	21	24	17	30	42	8	21	12	7	42	9	41	38	42	**2122**
fi	Finnish	Uralic	**7428**																			156	47	64	28	48	90	63	22	131	87	9	43	47	40	29	30	10	126	86	10	119	131	69	139	39	27	50	46	25	46	126	12	23	24	21	72	10	76	60	64	**3998**
fr	French	Romance	**34494**																				154	136	60	85	164	161	38	744	214	24	89	71	83	62	83	25	331	166	124	255	558	206	410	72	74	83	86	48	92	186	19	56	65	42	130	26	170	165	157	**12441**
gl	Galician	Romance	**2221**																					41	21	33	50	56	14	120	50	9	28	27	35	29	39	11	66	52	17	65	227	56	84	30	36	29	33	20	39	54	8	15	17	22	43	12	51	58	46	**3969**
he	Hebrew	Semitic	**6962**																						28	41	65	63	17	121	82	7	43	35	42	26	25	9	86	64	8	84	133	67	131	35	21	36	38	23	45	67	10	21	25	13	54	8	73	66	62	**3666**
hi	Hindi	Indo-Aryan	**1353**																							21	33	31	7	56	35	3	18	16	24	6	11	12	40	27	4	44	63	35	56	17	6	18	21	14	22	40	7	13	18	3	29	4	33	26	30	**1781**
hr	Croatian	Slavic	**2229**																								58	47	14	80	48	8	27	31	52	24	24	10	65	48	7	71	85	51	85	666	19	35	53	21	205	56	9	16	16	17	42	9	55	46	42	**3629**
hu	Hungarian	Uralic	**7702**																									70	20	146	99	11	49	48	47	27	28	10	121	75	10	126	148	87	149	46	26	56	55	27	53	88	13	29	30	20	75	9	83	74	75	**4357**
id	Indonesian	Malayo-Pol.	**3899**																										16	146	77	8	45	33	55	25	23	10	101	83	8	93	204	94	127	43	23	37	46	32	56	79	13	24	19	21	79	11	73	146	83	**4743**
is	Icelandic	Germanic	**488**																											31	18	2	9	12	12	6	5	3	27	22	3	27	35	20	30	13	4	13	13	9	13	28	4	7	5	5	16	3	18	16	14	**995**
it	Italian	Romance	**21025**																												179	24	83	62	73	58	78	24	240	150	20	219	480	161	303	67	68	72	81	48	83	153	19	44	58	41	112	23	144	143	137	**10084**
ja	Japanese	Japonic	**31614**																													14	222	47	48	21	23	8	123	81	9	128	175	79	196	40	19	48	50	28	51	96	12	37	31	12	84	12	92	75	267	**5382**
kk	Kazakh	Turkic	**1684**																														6	5	6	2	2	1	18	11	1	17	22	12	32	8	1	8	7	6	8	16	3	3	1	1	10	4	14	6	9	**581**
ko	Korean	Koreanic	**5400**																															23	26	10	10	4	56	41	5	63	93	47	89	23	9	25	26	17	29	51	7	17	13	5	47	5	48	49	57	**2539**
lt	Lithuanian	Baltic	**2031**																																28	16	16	6	57	39	6	70	64	39	107	25	15	30	30	16	29	46	8	13	11	12	36	6	57	33	35	**2091**
mk	Macedonian	Slavic	**1487**																																	21	22	9	53	46	6	56	93	56	88	52	19	29	39	25	106	48	9	14	15	16	43	8	57	57	45	**2761**
ml	Malayalam	Dravidian	**272**																																		3	1	41	32	7	41	58	35	46	21	2	22	21	13	20	44	6	4	3	1	16	2	26	15	17	**1388**
mr	Marathi	Indo-Aryan	**503**																																			1	46	35	5	47	86	47	50	22	3	25	25	13	19	56	4	5	3	1	17	1	26	14	17	**1648**
ne	Nepali	Indo-Aryan	**235**																																				17	13	4	17	21	14	19	10	1	10	10	6	9	17	3	1	1		7	1	11	6	6	**584**
nl	Dutch	Germanic	**14091**																																					133	13	177	218	96	199	53	42	66	64	34	61	151	16	37	35	29	90	18	104	84	88	**6364**
no	Norwegian	Germanic	**1364**																																						9	103	161	74	121	42	28	43	51	26	47	270	12	24	23	24	61	13	69	79	63	**4771**
oc	Occitan	Romance	**472**																																							13	24	11	14	6	5	6	6	4	7	12	2	3	3	3	7	3	9	8	7	**706**
pl	Polish	Slavic	**16270**																																								200	97	285	56	40	81	68	35	69	121	16	39	42	28	92	16	172	84	92	**6035**
pt	Portuguese	Romance	**13354**																																									177	312	74	76	71	85	47	101	155	20	42	54	45	140	23	156	213	165	**10932**
ro	Romanian	Romance	**3504**																																										136	44	43	42	49	30	58	75	15	23	27	29	72	13	82	96	74	**4515**
ru	Russian	Slavic	**32537**																																											70	42	85	78	44	114	140	17	54	55	29	119	25	2486	122	148	**11311**
sh	Serbo-Croat	Slavic	**2069**																																												17	27	46	19	373	46	8	14	13	17	35	9	45	42	36	**3337**
si	Sinhala	Indo-Aryan	**320**																																													22	22	10	15	48	4	4	4	2	15	1	21	14	16	**1464**
sk	Slovak	Slavic	**1904**																																														36	17	34	51	8	14	15	16	37	9	51	38	38	**2726**
sl	Slovenian	Slavic	**1836**																																															19	47	50	9	15	16	17	39	8	50	48	42	**2623**
sq	Albanian	Albanian	**740**																																																25	31	6	13	10	10	27		29	30	24	**1515**
sr	Serbian	Slavic	**4226**																																																	51	9	19	18	14	43	8	71	56	45	**3510**
sv	Swedish	Germanic	**17906**																																																		16	33	39	35	72	17	82	74	73	**5107**
sw	Swahili	Niger-Congo	**235**																																																			6	3	4	12		12	10	11	**652**
ta	Tamil	Dravidian	**1629**																																																				6	2	29	5	30	19	27	**1315**
te	Telugu	Dravidian	**1509**																																																					1	21	1	30	16	20	**1315**
tl	Tagalog	Malayo-Pol.	**312**																																																						12	1	16	17	10	**1015**
tr	Turkish	Turkic	**4067**																																																								10	67	77	69	**3698**
tt	Tatar	Turkic	**449**																																																									11	7	10	**603**
uk	Ukrainian	Slavic	**12871**																																																										73	72	**7043**
vi	Vietnamese	Vietic	**6727**																																																											89	**4660**
zh	Chinese	Chinese	**12306**																																																												**4415**

Table 3: WikiMatrix: number of extracted sentences for each language pair (in thousands), e.g. Es-Ja=219 corresponds to 219,260 sentences (rounded). The column *“size”* gives the number of lines in the monolingual texts after deduplication and LID.

Src/Trg	ar	bg	bs	cs	da	de	el	en	eo	es	fr	fr-ca	gl	he	hr	hu	id	it	ja	ko	mk	nb	nl	pl	pt	pt-br	ro	ru	sk	sl	sr	sv	tr	uk	vi	zh-cn	zh-tw
ar		4.9	1.8	3.0	4.5	3.8	6.7	20.3	4.1	13.2	12.2	9.0	5.6	3.5	2.2	2.7	9.2	9.9	4.2	5.3	5.5	4.9	4.4	3.0	12.0	12.2	5.6	5.6	1.5	2.7	1.2	4.0	2.4	4.5	12.3	8.2	4.9
bg	3.0		4.2	6.7	8.7	8.5	10.0	25.3	7.7	16.3	14.7	11.9	8.4	3.2	6.4	4.7	10.3	12.2	4.0	5.4	16.2	8.1	7.7	6.3	14.7	15.4	9.8	12.4	4.0	6.4	2.7	7.1	2.4	10.4	11.7	6.4	4.7
bs	1.2	6.1		4.1	3.7	5.8	4.5	21.7		9.9	6.7	4.6	5.7	1.0	30.9	2.4	5.3	6.5	1.4		12.9		3.8	2.9	9.9	10.9	5.7	4.8	3.1	10.4	10.6	4.4	1.5	5.4	5.8	2.8	1.8
cs	1.9	7.8	3.7		7.1	8.3	6.4	20.0	10.4	12.6	11.4	8.6	5.0	2.5	6.5	4.9	7.8	9.6	4.1	5.5	5.0	6.3	7.6	8.1	10.8	12.1	6.3	9.4	28.1	6.7	1.6	7.0	2.6	7.8	9.0	6.6	4.7
da	2.0	8.9	4.0	5.2		14.0	9.0	32.9	6.7	16.7	16.7	12.8	7.3	3.5	4.4	4.7	10.8	13.4	4.7	6.0	6.2	33.1	12.4	4.8	14.0	16.2	8.5	7.8	3.1	5.2	1.4	25.8	2.7	6.3	11.2	7.3	4.9
de	2.4	9.7	4.9	8.1	16.9		7.8	24.5	15.9	17.4	18.3	14.7	6.8	4.3	5.5	7.2	8.6	13.5	6.4	6.6	5.6	11.5	17.6	6.8	14.2	15.2	8.7	9.2	5.4	8.6	1.5	12.7	3.6	7.8	11.3	9.2	4.3
el	4.1	11.2	4.8	5.5	9.7	6.7		27.9	8.0	18.8	16.3	13.5	10.1	4.0	5.6	5.3	13.1	15.3	5.5	6.1	10.2	9.3	8.7	5.1	18.0	18.3	10.4	8.4	3.1	6.4	2.0	7.2	3.4	7.2	14.4	8.7	5.3
en	11.9	23.9	14.7	15.5	30.9	20.4	27.1		22.6	35.8	32.6	25.1	24.3	17.3	18.8	13.5	28.8	29.5	10.2	18.6	21.8	31.8	25.1	12.0	31.4	37.0	20.4	17.4	13.8	16.5	5.5	29.1	10.3	17.6	26.9	18.0	10.7
eo	1.8	6.4		7.3	7.4	13.5	8.1	23.1		16.1	17.6	12.7	10.5	1.9	3.0	4.8	9.2	13.9	2.5	4.8	6.2	7.2	12.1	6.7	12.4	16.0	6.9	8.6	7.4	5.6	0.6	8.8	1.8	5.4	7.7	5.2	3.6
es	6.2	14.3	6.8	8.0	15.7	12.9	16.4	33.2	13.5		25.6	19.9	30.1	8.1	8.7	7.7	16.1	23.8	7.9	9.9	11.9	13.3	14.1	8.1	27.6	27.8	14.7	11.6	6.0	8.1	2.6	13.7	5.2	10.0	17.8	12.3	6.6
fr	6.0	12.7	5.0	8.3	15.6	14.5	15.4	31.6	18.6	26.4			17.2	6.8	6.9	7.7	15.0	24.6	7.3	8.7	10.7	13.1	15.9	8.1	24.2	26.0	16.5	12.7	5.8	7.1	2.1	13.8	4.2	9.8	17.0	11.1	6.6
fr-ca	4.9	12.5	2.8	7.8	14.3	12.9	15.4	27.8	14.0	23.7			18.1	6.8	6.8	7.9	13.7	23.4	7.5	8.8	10.3		15.1	7.2	18.6	23.2	15.3	11.3	6.1	6.6	3.3	12.5	5.0	9.7	15.4		6.2
gl	2.6	7.3	4.7	2.9	7.4	5.3	8.5	23.4	9.2	34.4	16.0	15.2		2.5	3.5	2.8	9.8	19.3	3.6	4.2	5.6	6.9	6.5	4.3	22.4	23.7	7.7	5.9	1.7	3.1	0.2	4.3	2.3	3.9	9.4	5.8	4.2
he	3.4	5.7	1.8	3.7	6.1	5.4	6.3	25.7	4.2	15.3	13.6	10.3	5.1		2.5	3.2	8.9	11.2	4.1	4.5	4.6	5.4	6.5	3.7	13.7	15.0	6.3	8.0	1.9	2.6	1.0	5.2	1.8	5.1	10.7	6.9	4.6
hr	1.6	8.7	29.9	7.0	6.5	5.8	6.6	24.4	4.4	12.6	9.9	7.8	5.7	1.5		4.3	8.2	10.4	1.8	4.0	14.1	5.3	6.1	5.4	12.1	12.6	7.3	8.3	4.6	12.6	11.7	6.0	1.9	6.9	9.4	4.8	2.9
hu	1.6	5.6	2.6		5.9	7.0	5.5	16.7	6.5	10.8	10.9	9.3	3.9	2.1	3.6		6.6	8.2	4.4	6.2	3.9	4.5	6.0	4.3	9.7	9.9	7.1	5.8	3.4	4.2	1.2	5.3	3.0	4.4	8.5	6.5	4.2
id	4.1	9.1	4.2	5.2	10.1	6.7	11.1	24.9	8.2	16.4	15.1	11.1	9.9	5.1	5.6	5.2		12.7	5.8	9.1	7.0	10.0	9.4	5.5	14.6	16.7	9.8	8.1	3.6	5.6	1.8	9.3	4.6	7.3	18.5	11.0	6.2
it	5.3	11.7	5.0	6.9	13.1	11.5	14.5	30.0	13.9	26.4	24.9	20.0	19.3	6.2	7.0	6.7	14.0		7.3	9.0	9.9	13.3	12.8	7.3	22.8	24.9	13.3	10.2	5.4	7.1	2.3	11.9	4.6	8.8	15.3	10.7	5.8
ja	1.4	1.9	0.7	1.8	3.1	2.7	2.5	7.9	2.2	6.0	6.0	4.7	2.3	1.4	1.3	2.0	3.5	4.6		16.9	1.6	2.6	3.1	1.8	5.1	4.9	2.8	2.7	1.2	1.8	0.5	2.6	1.9	2.2	5.9		
ko	0.9	1.7		1.3	2.0	1.7	1.7	8.7	1.3	4.7	4.4	3.4	1.5	0.9	0.7	1.5	3.1	3.2	9.2		1.2	1.3	1.9	1.4	4.3	4.6	2.0	2.1	1.0	1.5	0.4	1.5	1.4	1.5	4.3		
mk	2.4	18.2	12.0	5.4	7.2	4.3	10.3	23.4	8.9	15.2	11.5	10.0	5.4	4.0	12.6	3.7	8.9	11.1	3.5	4.7		7.5	6.7	4.5	13.9	15.6	8.3	7.5	3.5	7.1	3.7	5.6	2.0	6.4	10.9	6.3	4.3
nb	2.7	8.5		5.4	32.7	9.8	8.9	35.1	9.5	17.0	14.6		8.0	3.1	3.9	4.5	9.9	15.0	5.5	5.7	6.7			4.7	14.2		9.5	7.5	3.2	3.9	0.6	26.3	1.9	6.2	7.8		
nl	2.4	8.2	2.9	5.9	14.2	16.1	8.4	26.5	13.4	16.8	16.7	13.5	7.3	3.9	4.8	5.3	11.4	13.3	5.2	5.9	5.9			5.3	13.8	15.4	7.8	7.6	4.1	5.1	1.6	11.1	3.2	6.1	10.6	8.0	5.1
pl	1.8	7.4	2.9	8.2	6.6	6.5	5.4	15.1	7.5	11.4	11.3	8.6	5.2	2.3	4.8	4.0	7.5	8.6	4.2	5.7	5.2	4.1	6.2		9.6	9.9	6.2	9.5	6.2	5.3	1.5	5.6	2.4	8.9	7.7	6.3	3.6
pt	7.4	15.2	7.0	8.0	15.5	13.2	18.7	35.0	12.0	32.4	26.7	19.0	23.0	8.8	10.3	8.4	17.5	24.4	7.8	10.6	13.1	14.3	14.9	8.9			15.3	11.7	6.6	8.1	2.0	14.3	6.2	9.3	18.0	12.9	5.8
pt-br	6.5	14.7	7.4	8.6	16.8	12.9	17.6	37.3	16.0	31.0	26.6	20.3	23.0	8.7	9.8	8.1	18.6	24.8	7.8	10.7	12.5		14.8	8.5			15.1	11.8	6.4	8.9	2.8	14.6	5.3	10.8	18.8	13.2	6.7
ro	3.2	9.7	3.7	5.1	9.4	7.5	10.4	25.0	6.7	18.8	19.3	14.6	10.0	4.0	5.7	5.9	11.0	15.5	4.3	6.4	7.3	8.0	8.1	5.2	15.4	17.7		8.0	3.6	5.0	1.9	7.0	3.3	6.6	12.7	7.8	4.9
ru	3.3	12.6	4.2	7.7	8.5	8.8	8.3	18.7	9.9	14.3	14.5	11.0	6.0	4.9	6.8	5.6	9.5	11.7	6.1	7.7	7.4	8.0	8.1	8.9	12.4	13.9	8.2		5.8	5.4	2.7	8.2	2.9	22.5	11.5	9.1	5.2
sk	0.7	5.1	2.7	27.0	4.3	5.7	3.2	16.9	9.3	9.4	8.5	6.7	2.7	1.0	5.1	3.7	4.9	6.9	2.2	3.9	3.5	4.9	5.0	7.1	7.8	8.5	3.5	6.6		5.0	1.5	4.3	1.6	5.4	5.4	2.3	2.5
sl	1.2	6.2	7.6	5.5	4.7	7.6	5.8	17.3	5.9	11.4	8.5	6.4	3.2	1.2	11.2	3.9	6.5	7.8	2.7	4.2	6.3	4.3	5.5	4.8		9.9	4.8	5.9	3.6		1.9	4.1	2.2	4.3	7.4	3.8	2.6
sr	1.8	7.6	33.2	5.1	3.7	3.8	5.8	22.8	3.2	11.9	9.1	7.5	3.5	1.1	30.4	2.7	6.1	8.2	1.2	2.9	13.7	3.3	3.3	3.9	10.8	11.3	5.6	7.2	2.8	9.5		3.0	1.1	5.7	6.8	4.3	3.1
sv	2.2	7.4	4.8	5.9	26.5	12.6	8.1	31.8	11.0	16.9	15.7	10.7	7.1	3.3	5.5	5.4	11.6	13.3	4.8	6.2	5.2	25.4	11.5	5.8	15.0	17.4	7.9	8.1	3.8	4.7	1.0		3.2	6.9	12.9	7.8	4.8
tr	2.2	3.5	2.0	2.6	3.9	4.1	4.7	15.9	2.9	9.4	7.7	6.7	3.6	1.6	2.1	3.4	6.7	6.4	4.3	7.0	3.5	3.1	4.2	2.5	9.0	8.4	4.6	4.0	1.8	2.3	0.8	3.5		3.3	8.2	6.7	4.4
uk	2.9	12.3	5.3	7.4	7.5	7.5	8.4	20.7	6.5	14.2	14.1	11.2	5.5	3.5	6.6	4.7	9.5	11.2	4.9	5.8	7.2	6.3	6.9	9.6		12.9	7.2	23.5	4.9	5.7	2.6	6.9	2.6		11.4	7.9	4.9
vi	4.2	7.5	4.0	4.7	8.5	6.0	8.8	20.2	7.3	13.7	13.2	9.9	6.5	4.6	4.9	4.7	14.7	10.7	5.6	9.3	6.9	5.7	7.3	4.5	13.0	14.1	8.5	7.2	3.4	4.6	1.7	8.2	4.0	6.7		9.9	6.7
zh-cn	2.1	3.2	1.0	2.2	3.8	3.2	4.5	11.8	3.8	8.2	7.6		3.2	1.7	1.9	3.0	6.6	6.0			3.4		3.8	2.2	7.1	7.9	4.1	4.1	1.6	2.4	0.9	3.1	2.3	3.0	10.8		
zh-tw	2.2	3.1	1.1	2.1	3.7	2.8	3.9	10.7	3.4	7.5	7.2	6.1	2.8	1.8	1.6	3.0	6.2	5.4			2.8		3.5	2.3	6.3	6.9	3.5	3.9	1.4	2.1	0.9	3.0	2.4	2.9	10.0		

Table 4: BLEU scores on the TED test set as proposed in (Qi et al., 2018). NMT systems were trained on bitexts mined in Wikipedia only (with at least twenty-five thousand parallel sentences). No other resources were used.

parameter settings shown in Figure 2 in the appendix. Since the TED development and test sets were already tokenized, we first detokenize them using Moses. We trained NMT systems for all possible language pairs with more than 25k mined sentences. This gives us in total 1886 language pairs in 45 languages. We train $L_1 \rightarrow L_2$ and $L_2 \rightarrow L_1$ with the same mined bitexts L_1/L_2. Scores on the test sets were computed with SacreBLEU (Post, 2018), see Table 4. Some additional results are reported in Table 6 in the annex. 23 NMT systems achieve BLEU scores over 30, the best one being 37.3 for Brazilian Portuguese to English. Several results are worth mentioning, like Farsi/English: 16.7, Hebrew/English: 25.7, Indonesian/English: 24.9 or English/Hindi: 25.7 We also achieve interesting results for translation between various non English language pairs for which it is usually not easy to find parallel data, e.g. Norwegian ↔ Danish ≈33, Norwegian ↔ Swedish ≈25, Indonesian ↔ Vietnamese ≈16 or Japanese / Korean ≈17.

Our results on the TED set give an indication on the quality of the mined parallel sentences. These BLEU scores should be of course appreciated in context of the sizes of the mined corpora as given in Table 3. Finally, we would like to point out that we run our approach on all available languages in Wikipedia, independently of the quality of LASER's sentence embeddings for each one.

6 Conclusion

We have presented an approach to systematically mine for parallel sentences in the textual content of Wikipedia, for all possible language pairs. We use a mining approach based on massively multilingual sentence embeddings (Artetxe and Schwenk, 2018b) and a margin criterion (Artetxe and Schwenk, 2018a). The same approach is used for all language pairs without the need for a language-specific optimization. In total, we make available 135M parallel sentences in 96 languages, out of which only 34M sentences are aligned with English. We were able to mine more than ten thousands sentences for 1620 different language pairs. This corpus of parallel sentences is freely available.[15] We also performed a large scale evaluation of the quality of the mined sentences by training 1886 NMT systems and evaluating them on the 45 languages of the TED corpus (Qi et al., 2018). This approach was recently extended to mine in Common Crawl texts (Schwenk et al., 2019).

---

[15] `https://github.com/facebookresearch/LASER/tree/master/tasks/WikiMatrix`

References

Sadaf Abdul-Rauf and Holger Schwenk. 2009. On the Use of Comparable Corpora to Improve SMT performance. In *EACL*, pages 16–23.

Sisay Fissaha Adafre and Maarten de Rijke. 2006. Finding similar sentences across multiple languages in Wikipedia. In *Proceedings of the Workshop on NEW TEXT Wikis and blogs and other dynamic text sources*.

Ahmad Aghaebrahimian. 2018. Deep neural networks at the service of multilingual parallel sentence extraction. In *Coling*.

Mikel Artetxe and Holger Schwenk. 2018a. Margin-based Parallel Corpus Mining with Multilingual Sentence Embeddings. *`https://arxiv.org/abs/1811.01136`*.

Mikel Artetxe and Holger Schwenk. 2018b. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. In *`https://arxiv.org/abs/1812.10464`*.

Andoni Azpeitia, Thierry Etchegoyhen, and Eva Martínez Garcia. 2017. Weighted Set-Theoretic Alignment of Comparable Sentences. In *BUCC*, pages 41–45.

Andoni Azpeitia, Thierry Etchegoyhen, and Eva Martínez Garcia. 2018. Extracting Parallel Sentences from Comparable Corpora with STACC Variants. In *BUCC*.

Houda Bouamor and Hassan Sajjad. 2018. H2@BUCC18: Parallel Sentence Extraction from Comparable Corpora Using Multilingual Sentence Embeddings. In *BUCC*.

Christian Buck and Philipp Koehn. 2016. Findings of the wmt 2016 bilingual document alignment shared task. In *Proceedings of the First Conference on Machine Translation*, pages 554–563, Berlin, Germany. Association for Computational Linguistics.

Vishrav Chaudhary, Yuqing Tang, Francisco Guzmán, Holger Schwenk, and Philipp Koehn. 2019. Low-resource corpus filtering using multilingual sentence embeddings. In *Proceedings of the Fourth Conference on Machine Translation (WMT)*.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzman, and Philipp Koehn. 2020. A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969.

Cristina España-Bonet, Ádám Csaba Varga, Alberto Barrón-Cedeño, and Josef van Genabith. 2017. An Empirical Analysis of NMT-Derived Interlingual Embeddings and their Use in Parallel Sentence Identification. *IEEE Journal of Selected Topics in Signal Processing*, pages 1340–1348.

Thierry Etchegoyhen and Andoni Azpeitia. 2016. Set-Theoretic Alignment for Comparable Corpora. In *ACL*, pages 2009–2018.

Simon Gottschalk and Elena Demidova. 2017. Multiwiki: Interlingual text passage alignment in Wikipedia. *ACM Transactions on the Web (TWEB)*, 11(1):6.

Francis Grégoire and Philippe Langlais. 2017. BUCC 2017 Shared Task: a First Attempt Toward a Deep Learning Framework for Identifying Parallel Sentences in Comparable Corpora. In *BUCC*, pages 46–50.

Mandy Guo, Qinlan Shen, Yinfei Yang, Heming Ge, Daniel Cer, Gustavo Hernandez Abrego, Keith Stevens, Noah Constant, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Effective Parallel Corpus Mining using Bilingual Sentence Embeddings. *arXiv:1807.11906*.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving Human Parity on Automatic Chinese to English News Translation. *arXiv:1803.05567*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *`https://arxiv.org/abs/1607.01759`*.

H. Jégou, M. Douze, and C. Schmid. 2011. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*.

Philipp Koehn, Francisco Guzmán, Vishrav Chaudhary, and Juan M. Pino. 2019. Findings of the wmt 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation, Volume 2: Shared Task Papers*, Florence, Italy. Association for Computational Linguistics.

Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. 2018. Findings of the wmt 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 726–739, Belgium, Brussels. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

P. Lison and J. Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *LREC*.

Mehdi Zadeh Mohammadi and Nasser GhasemAghaee. 2010. Building bilingual parallel corpora based on Wikipedia. In *2010 Second International Conference on Computer Engineering and Applications*, pages 264–268.

Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Computational Linguistics*, 31(4):477–504.

P Otero, I López, S Cilenis, and Santiago de Compostela. 2011. Measuring comparability of multilingual corpora extracted from Wikipedia. *Iberian Cross-Language Natural Language Processings Tasks (ICL)*, page 8.

Pablo Gamallo Otero and Isaac González López. 2010. Wikipedia as multilingual source of comparable corpora. In *Proceedings of the 3rd Workshop on Building and Using Comparable Corpora, LREC*, pages 21–25.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexandre Patry and Philippe Langlais. 2011. Identifying parallel documents from a large bilingual collection of texts: Application to parallel article extraction in Wikipedia. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 87–95. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.

Philip Resnik. 1999. Mining the Web for Bilingual Text. In *ACL*.

Philip Resnik and Noah A. Smith. 2003. The Web as a Parallel Corpus. *Computational Linguistics*, 29(3):349–380.

Holger Schwenk. 2018. Filtering and mining parallel data in a joint multilingual space. In *ACL*, pages 228–234.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2019. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *`https://arxiv.org/abs/1911.04944`*.

Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *NAACL*, pages 403–411.

J. Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *LREC*.

Chen-Tse Tsai and Dan Roth. 2016. Cross-lingual wikification using multilingual embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 589–598.

Dan Tufis, Radu Ion, Ştefan Daniel, Dumitrescu, and Dan Ştefănescu. 2013. Wikipedia as an smt training corpus. In *RANLP*, pages 702–709.

Masao Utiyama and Hitoshi Isahara. 2003. Reliable Measures for Aligning Japanese-English News Articles and Sentences. In *ACL*.

Yinfei Yang, Gustavo Hernández Ábrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. In *`https://arxiv.org/abs/1902.08564`*.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations Parallel Corpus v1.0. In *LREC*.

A Appendix

Table 5 provides the amounts of mined parallel sentences for languages which have a rather small Wikipedia. Aligning those languages obviously yields to a very small amount of parallel sentences. Therefore, we only provide these results for alignment with high resource languages. It is also likely that several of these alignments are of low quality since the LASER embeddings were not directly trained on most these languages, but we still hope to achieve reasonable results since other languages of the same family may be covered.

ISO	Name	Language Family	size	ca	da	de	en	es	fr	it	nl	pl	pt	sv	ru	zh	total
an	Aragonese	Romance	**222**	24	7	12	23	33	16	13	9	10	14	9	11	6	**324**
arz	Egyptian Arabic	Arabic	**120**	7	6	11	18	12	12	10	8	9	10	8	12	7	**278**
as	Assamese	Indo-Aryan	**124**	8	6	11	7	11	12	10	9	9	8	8	9	3	**216**
azb	South Azerbaijani	Turkic	**398**	6	4	9	8	9	10	9	7	6	8	6	7	3	**172**
bar	Bavarian	Germanic	**214**	7	6	41	16	12	12	10	8	9	10	8	10	5	**261**
bpy	Bishnupriya	Indo-Aryan	**128**	2	1	4	4	3	4	2	2	3	2	2	3	1	**71**
br	Breton	Celtic	**413**			20	16	22	23	22			19		16	6	**200**
ce	Chechen	Northeast Caucasian	**315**	2	1	2	2	2	2	2	2	2	2	2	2	1	**56**
ceb	Cebuano	Malayo-Polynesian	**17919**	14	9	22	29	27	24	24	15	17	20	55	21	9	**594**
ckb	Central Kurdish	Iranian	**127**	2	2	6	8	5	5	4	4	4	4	3	6	4	**113**
cv	Chuvash	Turkic	**198**	4	3	5	4	6	6	7	5	4	6	5	8	2	**129**
dv	Maldivian	Indo-Aryan	**52**	2	2	5	6	4	4	3	3	3	3	3	5	3	**96**
fo	Faroese	Germanic	**114**	13	12	14	32	21	18	15	11	11	17	12	13	6	**335**
fy	Western Frisian	Germanic	**493**	13	8	16	32	21	18	17	38	12	18	13	14	5	**453**
gd	Gaelic	Celtic	**66**	1	1	1	1	1	1	1	1	1	1	1	1	1	**41**
ga	Irish	Irish	**216**		2	3	4	3	3	3	2	2	3	2	3	1	**70**
gom	Goan Konkami	Indo-Aryan	**69**	9	7	10	8	13	13	13	9	9	11	9	10	4	**240**
ht	Haitian Creole	Creole	**60**	2	1	3	4	3	4	3	2	3	2	2	3	1	**72**
ilo	Iloko	Philippine	**63**	3	2	4	5	4	4	4	3	3	4	3	4	2	**96**
io	Ido	constructed	**153**	5	3	6	11	7	7	5	5	5	6	5	5	3	**143**
jv	Javanese	Malayo-Polynesian	**220**	8	5	8	13	12	10	11	8	7	11	8	8	3	**219**
ka	Georgian	Kartvelian	**480**	11	7	15	12	16	17	16	12	11	14	12	13	5	**288**
ku	Kurdish	Iranian	**165**	5	4	8	5	8	7	8	7	6	7	6	6	3	**222**
la	Latin	Romance	**558**	12	9	17	32	20	18	17	12	13	18	13	14	6	**478**
lb	Luxembourgish	Germanic	**372**	12	7	26	22	19	18	15	11	11	16	12	11	4	**305**
lmo	Lombard	Romance	**147**	6	3	7	10	7	7	11	6	5	7	5	5	3	**144**
mg	Malagasy	Malayo-Polynesian	**263**	6	5	9	13	9	12	8	7	7	7	8	7	4	**199**
mhr	Eastern Mari	Uralic	**61**	3	2	4	3	4	4	5	3	3	4	3	4	2	**96**
min	Minangkabau	Malayo-Polynesian	**255**	4	2	6	7	5	5	5	4	4	4	5	5	2	**121**
mn	Mongolian	Mongolic	**255**	4	3	7	5	6	6	7	6	5	5	5	5	3	**197**
mwl	Mirandese	Romance	**64**	6	3	4	10	8	6	5	3	4	34	3	4	2	**154**
nds nl	Low German/Saxon	Germanic	**65**	5	4	6	10	7	7	6	15	5	6	5	5	3	**151**
ps	Pashto	Iranian	**89**		2	3	2	3	3	3	3	3	3	3	3	1	**73**
rm	Romansh	Italic	**57**	2	2	10	5	4	4	3	2	3	3	3	3	1	**86**
sah	Yakut	Turkic/Sib	**134**	4	3	7	5	6	6	6	5	5	5	5	6	3	**134**
scn	Sicilian	Romance	**81**	5	3	6	9	7	7	11	5	5	6	5	5	2	**143**
sd	Sindhi	Iranian	**115**		3	9		8	8	7	7	6	7	5	8	5	**152**
su	Sundanese	Malayo-Polynesian	**120**	4	3	5	7	6	5	6	4	4	5	4	4	2	**117**
tk	Turkmen	Turkic	**56**	2	2	3	3	4	3	4	2	2	4	2	3	1	**76**
tg	Tajik	Iranian	**248**	5	4	11	15	9	9	8	8	7	8	6	10	6	**192**
ug	Uighur	Turkic	**83**	4	3	9	10	7	8	6	6	5	6	5	9	6	**168**
ur	Urdu	Indo-Aryan	**150**	2	2	3	5	3	3	3	3	3	3	3	3	2	**123**
wa	Walloon	Romance	**56**	3	2	4	5	5	4	4	3	3	4	3	3	2	**93**
wuu	Wu Chinese	Chinese	**75**	8	6	11	17	12	11	10	8	9	11	9	10	43	**283**
yi	Yiddish	Germanic	**131**	3	2	4	3	4	4	5	3	3	4	3	4	1	**92**

Table 5: WikiMatrix (part 2): number of extracted sentences (in thousands) for languages with a rather small Wikipedia. Alignments with other languages yield less than 5k sentences and are omitted for clarity.

Table 2 gives the detailed configuration which was used to train NMT models on the mined data in Section 5. An 5000 subword vocabulary was learnt using SentencePiece (Kudo and Richardson, 2018). Decoding was done with beam size 5 and length normalization 1.2.

```
--arch transformer
--share-all-embeddings
--encoder-layers 5
--decoder-layers 5
--encoder-embed-dim 512
--decoder-embed-dim 512
--encoder-ffn-embed-dim 2048
--decoder-ffn-embed-dim 2048
--encoder-attention-heads 2
--decoder-attention-heads 2
--encoder-normalize-before
--decoder-normalize-before
--dropout 0.4
--attention-dropout 0.2
--relu-dropout 0.2
--weight-decay 0.0001
--label-smoothing 0.2
--criterion label_smoothed_cross_entropy
--optimizer adam
--adam-betas '(0.9, 0.98)'
--clip-norm 0
--lr-scheduler inverse_sqrt
--warmup-update 4000
--warmup-init-lr 1e-7
--lr 1e-3 --min-lr 1e-9
--max-tokens 4000
--update-freq 4
--max-epoch 100
--save-interval 10
```

Figure 2: Model settings for NMT training with `fairseq`

Finally, Table 6 gives the BLEU scores on the TED corpus when translating into and from English for some additional languages.

Lang	xx → en	en → xx
et	15.9	14.3
eu	10.1	7.6
fa	16.7	8.8
fi	10.9	10.9
lt	13.7	10.0
hi	17.8	21.9
mr	2.6	3.5

Table 6: BLEU scores on the TED test set as proposed in (Qi et al., 2018). NMT systems were trained on bitexts mined in Wikipedia only. No other resources were used.

ChEMU-Ref: A Corpus for Modeling Anaphora Resolution in the Chemical Domain

Biaoyan Fang[1], **Christian Druckenbrodt**[2], **Saber A. Akhondi**[2],
Jiayuan He[1,3], **Timothy Baldwin**[1] **and Karin Verspoor**[1]
[1]The University of Melbourne, Australia
[2]Elsevier
[3]RMIT University, Australia
`biaoyanf@student.unimelb.edu.au`
`{c.druckenbrodt,s.akhondi}@elsevier.com`
`{estrid.he,tbaldwin,karin.verspoor}@unimelb.edu.au`

Abstract

Chemical patents contain rich coreference and bridging links, which are the target of this research. Specially, we introduce a novel annotation scheme, based on which we create the ChEMU-Ref dataset from reaction description snippets in English-language chemical patents. We propose a neural approach to anaphora resolution, which we show to achieve strong results, especially when jointly trained over coreference and bridging links.

1 Introduction

Chemical research has contributed greatly to human society and wellbeing, including new medicines and vaccines (Gwynne and Heabrer, 2015). Research is heavily reliant on knowledge of existing chemical processes and methods of chemical synthesis, which are documented in chemical research literature and chemical patents. Given the rapid growth of both publications and patents in chemistry, the need for automatic methods to extract semi-structured knowledge from chemical texts is becoming increasingly critical (Li et al., 2016; Akhondi et al., 2019).

Anaphora resolution is a key component of comprehensive information extraction (Rösiger, 2019; Poesio et al., 2016). In chemistry, different chemical compounds are mixed and reacted together in different ways to generate novel compounds, and to understand the precise chemical process often involves both resolving anaphoric references and understanding chemical changes/interactions a given entity is involved in. For example, as seen in Figure 1, while the final mention of *mixture* on line 3 and that on line 4 are both coreferent and chemically identical, in the case of *mixture* on line 2 and the first mention of *mixture* on line 3, the chemical composition is the same but a transformation has taken place via the *stir* and *cool* actions.

Our aim in this paper is to both identify anaphoric references in chemical patents, and determine the chemical relation between each linked pair of entities. We propose a domain-specific annotation framework based on five types of anaphora relations combining coreference and bridging. We then construct a dataset following this framework, annotated by chemical experts who achieve high inter-annotator agreement. We additionally extend existing anaphora resolution methods to model anaphora in chemical text, and compare both component-wise and joint models for anaphora resolution. This dataset will be released as part of the upcoming ChEMU 2021 shared task[1] (He et al., to appear).

Our contributions in this paper are as follows: (1) we propose a novel annotation scheme for anaphora resolution in chemical patents; (2) we develop a novel anaphora-resolution dataset based on chemical patents; and (3) we extend a general-purpose coreference resolution method, and achieve strong results via joint training over coreference and bridging with domain-specific fine-tuning.

2 Related Work

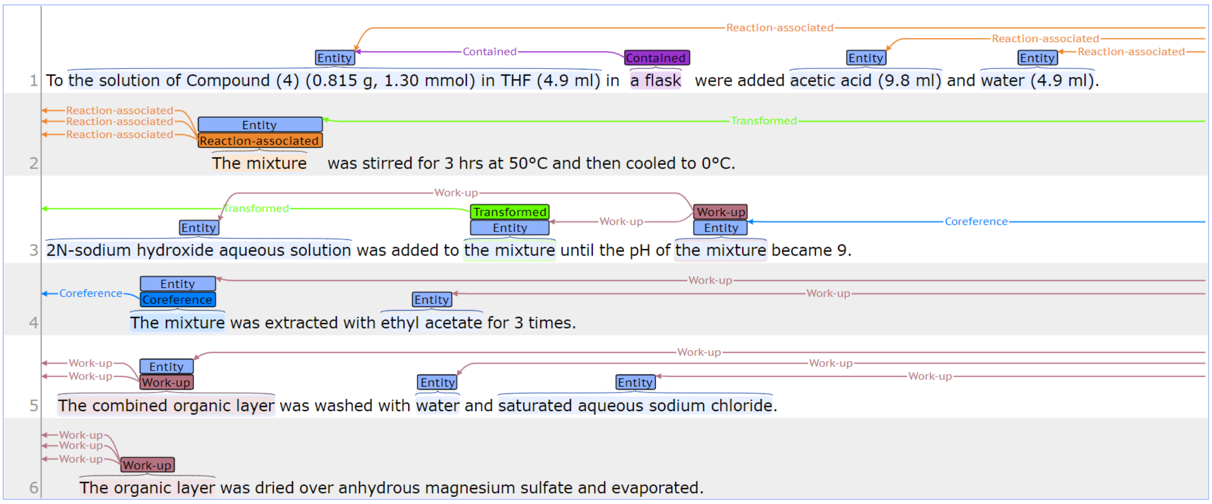
Anaphora occurs in two basic forms: coreference and bridging. Coreference occurs when different expressions in a text refer to the same entity in the real world (Ng, 2017; Clark and Manning, 2015), while bridging occurs between discrete entities that are linked via lexical semantic, frame-based, or encyclopedic relations (Asher and Lascarides, 1998; Hou et al., 2018).

Most existing anaphora datasets focus only on coreference, predominantly using generic relations (Pradhan et al., 2012; Ghaddar and Langlais, 2016a), but also using domain-specific relations for knowledge-rich corpora such as biomedical litera-

---

[1] `http://chemu.eng.unimelb.edu.au/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1362–1375
April 19 - 23, 2021. ©2021 Association for Computational Linguistics



1 To the solution of Compound (4) (0.815 g, 1.30 mmol) in THF (4.9 ml) in a flask were added acetic acid (9.8 ml) and water (4.9 ml).

2 The mixture was stirred for 3 hrs at 50°C and then cooled to 0°C.

3 2N-sodium hydroxide aqueous solution was added to the mixture until the pH of the mixture became 9.

4 The mixture was extracted with ethyl acetate for 3 times.

5 The combined organic layer was washed with water and saturated aqueous sodium chloride.

6 The organic layer was dried over anhydrous magnesium sulfate and evaporated.

Figure 1: Annotated snippet of anaphora resolution in the chemical patents. Different color of links represent different anaphora relation types. Detailed anaphora relation definition can be seen Section 3.3.

ture (Nguyen et al., 2011; Cohen et al., 2017).

The CoNLL-2012 dataset (Pradhan et al., 2012) is a general corpus consisting of texts from three languages (English, Chinese, and Arabic). It is annotated based on OntoNotes v5.0 (Weischedel et al., 2013) and includes two types of coreference relations: IDENTITY, a symmetrical and transitive relation; and APPOSITIVE, two noun phrases that are adjacent and not linked by a copula. Coreference resolution is modelled as a clustering task. The wikicoref corpus was constructed with the same relations, over Wikipedia documents (Ghaddar and Langlais, 2016a).

BioNLP-ST 2011 (Nguyen et al., 2011) is a domain-specific coreference corpus over abstracts from biomedical literature, focusing mainly on gene–protein coreference, considering four relations: RELAT (relative pronouns or adjectives, e.g. *which*), PRON (pronouns, e.g. *they*), DNP (definite or demonstrative noun phrases marked with *the*, *this*, etc.), and APPOS (apposition). Instead of modelling coreference resolution as a clustering task, here the direction of coreference links is preserved. As this corpus focuses on gene–protein coreference, the range of coreference phenomena is limited. The CRAFT-CR corpus (Cohen et al., 2017) adds coreference relations to the Colorado Richly Annotated Full Text (CRAFT) corpus (Bada et al., 2012), following OntoNotes v5.0 with minor adaptions, and including discontinuous expressions, domain-specific proper nouns, and a broad range of mention types.

The definition of bridging is somewhat imprecise (Zeldes, 2017; Hou et al., 2018), and different corpora have adopted different definitions. Based on Rösiger et al. (2018), there are two types of bridging: *referential bridging*, which can be treated as a context-based relation; and *lexical bridging*, which describes lexical-semantic relations such as holonymy and meronymy. Poesio et al. (2008) introduced the ARRAU corpus of general language texts for bridging, which consists of news, dialogue, and narrative text. In the corpus, entities are limited to noun phrases, and most bridging pairs are lexical relations, with only a small number of instances of referential bridging. ISnotes (Hou et al., 2018) includes 50 Wall Street Journal (WSJ) articles from the OntoNotes corpus, and has both coreference and bridging annotations, with most of the bridging pairs being referential. BASHI (Rösiger, 2018a) has both coreference and bridging annotations over 50 WSJ articles based on the OntoNotes v5.0 guidelines, with most bridging links once again being referential. Rösiger (2016) developed a corpus called SciCorp based on English scientific papers, following the same annotation scheme as BASHI.

Due to limited dataset availability, most research has modelled coreference resolution and bridging separately. There are two basic approaches to coreference resolution. First is mention ranking methods, which aim to score the coreferent probability of mention pairs (Clark and Manning, 2015, 2016a,b; Wiseman et al., 2015, 2016), and make the assumption that mentions have been pre-identified, meaning they are heavily reliant on upstream mention detection methods. Second is span ranking methods, which combine mention detection with coreference prediction (Lee et al., 2017, 2018; Zhang et al., 2018; Grobol, 2019; Kantor and Globerson, 2019), and tend to perform better. Bridging methods can be grouped into: (1) rule-based methods (Hou et al., 2014; Rösiger, 2018b; Rösiger et al., 2018); and (2) machine learning methods (Hou, 2018a,b, 2020; Yu and Poesio, 2020). Rule-based methods have been shown to achieve competitive results on domain-specific corpora, but equally to be domain brittle. Yu and Poesio (2020) jointly trained a model for coreference resolution and bridging by adapting a span ranking method for coreference (Lee et al., 2018; Kantor and Globerson, 2019), achieving good performance over various bridging corpora. However, they evaluated their model only on bridging.

3 Annotation Scheme

In this section, we introduce our annotation guidelines for anaphora resolution in chemical patents. The complete annotation guidelines are made available at Fang et al. (2021).

3.1 Corpus Selection

We build on the ChEMU corpus (Verspoor et al., 2020) developed for the ChEMU 2020 shared task (He et al., 2020). This corpus consists of 'snippets' extracted from chemical patents, where each snippet corresponds to a reaction description. It is common that several snippets are extracted from the same chemical patent.

3.2 Mention Type

We aim to capture anaphora in chemical patents, with a focus on identifying chemical compounds during the reaction process. Consistent with other anaphora corpora (Pradhan et al., 2012; Cohen et al., 2017; Ghaddar and Langlais, 2016b), only mentions that are involved in referring relationships (as defined in Section 3.3) and related to chemical compounds are annotated. The mention types that are considered for anaphora annotation are listed below.

It should be noted that verbs (e.g. *mix*, *purify*, *distil*) and descriptions that refer to events (e.g. *the same process*, *step 5*) are not annotated in this corpus.

Chemical Names: Chemical names are a critical component of chemical patents. We capture as atomic mentions the formal name of chemical compounds, e.g. *N-[4-(benzoxazol-2-yl)-methoxyphenyl]-S-methyl-N'-phenyl-isothiourea* or *2-Chloro-4-hydroxy-phenylboronic acid*. Chemical names often include nested chemical components, but for the purposes of our corpus, we consider chemical names to be atomic and don't annotate internal mentions. Hence *4-(benzoxazol-2-yl)-methoxyphenyl* and *acid* in the examples above will not be annotated as mentions, as they are part of larger chemical names.

Identifiers: In chemical patents, identifiers or labels may also be used to represent chemical compounds, in the form of uniquely-identifying sequences of numbers and letters such as *5i*. These can be abbreviations of longer expressions incorporating that identifier that occur earlier in the text, such as *chemical compound 5i*, or may refer back to an exact chemical name with that identifier. Thus, the identifier is annotated as an atomic mention as well.

Phrases and Noun Types: Apart from chemical names and identifiers, chemical compounds are commonly presented as noun phrases (NPs). An NP consists of a noun or pronoun, and premodifiers; NPs are the most common type of compound expressions in chemical patents. Here we detail NPs that are related to compounds:

- Pronouns: In chemical patents, pronouns (e.g. *they* or *it*) usually refer to a previously-mentioned chemical compounds.
- Definite NPs: Commonly used to refer to chemical compounds, e.g. *the solvent*, *the title compound*, *the mixture*.

Furthermore, there are a few types of NPs that need specific handling in chemical patents:

- Quantified NPs: Chemical compounds are usually described with a quantity. NPs with quantities are considered as atomic mentions if the quantities are provided, e.g. *398.4 mg of the compound 1*.
- NPs with prepositions: Chemical NPs connected with prepositions (e.g. *in*, *with*, *of*) should be considered as a single mention. For example, *the appropriate amino derivative in dry THF* is a single mention.

NPs describing chemical equipment containing a compound may also be relevant to anaphora resolution. This generally occurs when the equipment that contains the compound undergoes a process that also affects the compound. Thus, mentions such as *the flask* and *the autoclave* can also be mentions if they are used to implicitly refer to a contained compound.

Unlike many annotation schemes, our annotation allows discontinuous mentions. For example, the underlined spans of the fragment *114 mg of 4-((4aS,7aS)-6-benzyloctahydro-1-pyrrolo[3,4-b]pyridine-1-yl)-7H-pyrrolo[2,3-d]pyrimidine was obtained with a yield of about 99.1%* are treated as a single discontinuous mention. This introduces further complexity into the task and helps to capture more comprehensive anaphora phenomena.

Relationship to ChEMU 2020 entities: Since this dataset is built on the ChEMU 2020 corpus (He et al., 2020), annotation of related chemical compounds is available by leveraging existing entity annotations introduced for the ChEMU 2020 named entity recognition (NER) task. However, there are some differences in the definitions of entities for the two tasks.

In the original ChEMU 2020 corpus, entity annotations identify chemical compounds (i.e. *REACTION_PRODUCT*, *STARTING_MATERIAL*, *REAGENT_CATALYST*, *SOLVENT*, and *OTHER_COMPOUND*), reaction conditions (i.e. *TIME*, *TEMPERATURE*), quantity information (i.e. *YIELD_PERCENT*, *YIELD_OTHER*), and example labels (i.e. *EXAMPLE_LABEL*). There is overlap with our definition of mention for the labels relating to chemical compounds. However, in our annotation, chemical names are annotated along with additional quantity information, as we consider this information to be an integral part of the chemical compound description. Furthermore, the original entity annotations do not include generic expressions that co-refer with chemical compounds such as *the mixture*, *the organic layer*, or *the filtrate*, and neither do they include equipment descriptions.

3.3 Relation Types

Anaphora resolution subsumes both coreference and bridging. In the context of chemical patents, we define four sub-types of bridging, incorporating generic and chemical knowledge.

A referring mention which cannot be interpreted on its own, or an indirect mention, is called an *anaphor*, and the mention which it refers back to is called the *antecedent*. In relation annotation, we preserve the direction of the anaphoric relation, from the anaphor to the antecedent. Following similar assumptions in recent work, we restrict annotations to cases where the antecedent appears earlier in the text than the anaphor.

3.3.1 Coreference

Coreference is defined as expressions/mentions that refer to the same entity (Ng, 2017; Clark and Manning, 2015). In chemistry, identifying whether two mentions refer to the same entity needs to consider various chemical properties (e.g. temperature or pH). As such, for two mentions to be coreferent, they must share the same chemical properties. We consider two different cases of coreference:

- **Single Antecedents:** the anaphor refers to a single antecedent.
- **Multiple Antecedents:** the anaphor refers to multiple antecedents, e.g. in cases where multiple antecedents are combined to form a single *mixture*.

It is possible for there to be ambiguity as to which mention of a given antecedent an anaphor refers to (where the mention is repeated); in these cases the closest mention is selected.

3.3.2 Bridging

As stated in Section 3.3.1, when we consider the anaphora relations, we take the chemical properties of the mention into consideration. Coreference is insufficient to cover all instances of anaphora in chemical patents, and bridging occurs frequently. We define four bridging types:

TRANSFORMED: Links between chemical compounds that are initially based on the same components, but which have undergone a change in condition, such as pH or temperature. Such cases must be one-to-one relations (not one-to-many). As shown in Figure 1, the *mixture* in line 2 and the first-mentioned *mixture* in line 3 have the TRANSFORMED relation, as they have the same chemical components but different chemical properties.

REACTION-ASSOCIATED: The relationship between a chemical compound and its immediate source compounds is via a mixing process, where the source compounds retain their original chemical structure. This relation is one-to-many from the anaphor to the source compounds (antecedents). For example, the *mixture* in line 2 has REACTION-ASSOCIATED links to three mentions on line 1 that are combined to form it: (1) *the solution of Compound (4) (0.815 g, 1.30 mmol) in THF (4.9 ml)*; (2) *acetic acid (9.8 ml)*; and (3) *water (4.9 ml)*).

WORK-UP: Chemical compounds are used to isolate or purify an associated output product, in a

	Train	Dev	Test
Snippets	148	27	45
Sentences	763	164	274
Tokens/Sentences	27.5	24.7	25.8
Mentions	2,284	430	736
Dis. Mentions	88	10	17
Coref.	421	88	124
Bridging	1,731	323	577
TR	85	17	29
RA	515	105	167
WU	1,063	172	364
CT	68	29	17

Table 1: Corpus annotation statistics. "Dis. Mentions" means discontinuous mentions. "Coref.", "TR", "RA", "WU", and "CT" denote COREFERENCE, TRANSFORMED, REACTION-ASSOCIATED, WORK-UP and CONTAINED, respectively. "Bridging" is the total across all bridging relations.

one-to-many relation, from the anaphor to the compounds (antecedents) that are used for the work-up process. As demonstrated in Figure 1, *The combined organic layer* in line 5 comes from the extraction of *The mixture* and *ethyl acetate* in line 4, and they are hence annotated as WORK-UP.

CONTAINED: A chemical compound is contained inside some equipment. It is a one-to-many relation from the anaphor (equipment) to the compounds (antecedents) that it contains. An example of this is *a flask* and *the solution of Compound (4) (0.815 g, 1.30 mmol) in THF (4.9 ml)* on line 1, where the compound is contained in the flask.

4 Task definition

Anaphora resolution can be decomposed into a two-step task: (1) mention detection; and (2) anaphora relation detection.

For the evaluation of mention and relation detection, we use precision, recall and F1. One issue here is that, for coreference resolution, anaphors can link to multiple antecedents. Many coreference evaluation metrics (Moosavi and Strube, 2016; Recasens and Hovy, 2011; Luo, 2005) cannot deal with this since they model coreference resolution as a clustering task, where all related antecedents and anaphors occur in one cluster, and assume a given mention occurs in a unique cluster. Hence we adopt the approach to evaluation of Kim et al. (2012), scoring coreference from two perspectives: (1) *surface coreference*; and (2) *atom coreference*. Surface coreference considers whether the anaphor refers to the closest previous antecedent(s). Atom coreference considers whether the anaphor refers to the correct antecedent(s). Atom coreference links take the coreferent transitivity into consideration and can be generated from surface coreference links, which we use by default.

For the corpus annotation, we use the BRAT text annotation tool.[2] To date, 220 snippets have been annotated by two chemical experts, a PhD candidate and a final year bachelor student in Chemistry. Four rounds of annotation training were completed prior to beginning official annotation. In each round, the two annotators individually annotated the same 10 snippets (different across each round of annotation), and compared their annotations; annotation guidelines were then refined based on discussion. After several rounds of training, we achieved a high inner-annotator agreement of Krippendorff's $\alpha = 0.92$ (Krippendorff, 2004) at the mention level,[3] and $\alpha = 0.84$ for relations. In total, 1,500 snippets will be annotated in the final dataset that will be used in the ChEMU 2021 shared task.

The statistics of the current corpus, and train/dev/test set splits that form the basis of our experiments in this paper, are shown in Table 1. The dev and test partitions were both double annotated by the two expert annotators, with any disagreements merged by an adjudicator.

5 Methodology

We propose a joint neural model for anaphora resolution.[4] Similar to Yu and Poesio (2020), our model adopts an end-to-end neural conference resolution (Lee et al., 2017, 2018), as outlined in Figure 2.

Assume the snippet has T tokens represented as vector $X = \{x_1, ..., x_T\}$, consisting of fixed pretrained word and character embeddings learned from a convolution neural network (CNN).

For mention candidate detection, we follow the assumption of Lee et al. (2018), considering continuous tokens as a potential span and computing the span score (s_m) for each possible span. Specifically, span representation s_i is obtained by the concatenation of output token representations (x_i^*) from a bidirectional LSTM (BiLSTM) (Hochreiter and Schmidhuber, 1997), the syntactic head representation (h_i) obtained from an attention mecha-

---

[2] `https://brat.nlplab.org/`

[3] With the lowest agreement being $\alpha = 0.89$ for coreference mentions.

[4] Code available at `https://github.com/biaoyanf/ChEMU-Ref`

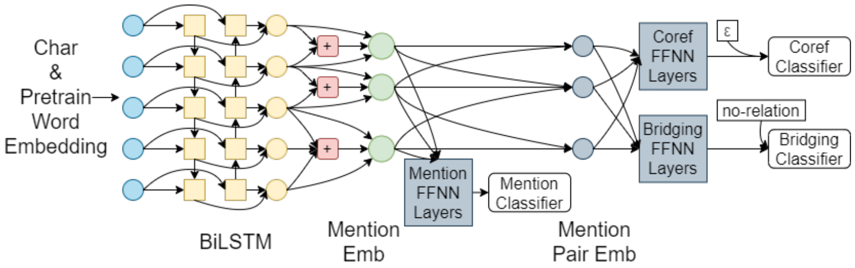


Figure 2: Joint training architecture.

nism (Bahdanau et al., 2015), and a feature vector of the mention ($\phi(i)$):

$$X^* = \text{BiLSTM}(X)$$
$$\alpha_t = w_\alpha \cdot \text{FFNN}_\alpha(x_t^*)$$
$$a_{i,t} = \frac{\exp(\alpha_t)}{\sum_{k=\text{START}(i)}^{\text{END}(i)} \exp(\alpha_k)}$$
$$h_i = \sum_{t=\text{START}(i)}^{\text{END}(i)} a_{i,t} \cdot x_t$$
$$s_i = [x^*_{\text{START}(i)}, x^*_{\text{END}(i)}, h_i, \phi(i)]$$

and the span score $s_m(i)$ is computed as:

$$s_m(i) = w_s \cdot \text{FFNN}_s(s_i)$$

where FFNN denotes a feed-forward neural network, and START(i) and END(i) represent the starting and ending token index for span i, respectively. To reduce the number of spans considered, we use a beam of λT candidate mention spans.

Inspired by Zhang et al. (2018), the mention loss is defined as:

$$L_{mention} = -\sum_{i=1}^{\lambda T} m_i * \log(\text{sigmoid}(s_m(i))) + (1 - m_i) * \log(1 - \text{sigmoid}(s_m(i)))$$

where:

$$m_i = \begin{cases} 0 & \text{span } i \notin \text{GOLD}_m \\ 1 & \text{span } i \in \text{GOLD}_m \end{cases}$$

GOLD_m is the set of gold mentions that are involved in anaphora relations.

For anaphoric relation detection, a span pair embedding is obtained by the concatenation of each span embedding ($s_m(i), s_m(j)$) and the element-wise multiplication of the span embeddings ($s_m(i) \circ s_m(j)$) and a feature vector ($\phi(i,j)$) for span pair i and j:

$$s_{i,j} = [s_m(i), s_m(j), s_m(i) \circ s_m(j), \phi(i,j)]$$

As coreference and bridging are different, we consider them separately.

For coreference resolution, we follow Lee et al. (2018) in optimizing the marginal log-likelihood of all correct antecedents for a given anaphor:

$$L_{coref} = \log \prod_{i=1}^{N} \sum_{\hat{y} \in Y(i) \bigcap \text{GOLD}_c(i)} P(\hat{y})$$

where N is the number of candidate mentions; and $Y(i) = \{\epsilon, 1, ..., i-1\}$ is the set of possible assignments for each y_i, which ϵ represents a dummy antecedent and the numbers represent the proceeding spans. $\text{GOLD}_c(i)$ is the gold coreferent antecedents that span i refers to. If span i doesn't have a coreferent antecedent, $\text{GOLD}_c(i) = \epsilon$. $P(y_i)$ is obtained via softmax over the antecedent scores s_c for the corresponding anaphor:

$$s_c(i,j) = \begin{cases} 0 & j = \epsilon \\ w_c \cdot \text{FFNN}_c(s_{i,j}) & j \neq \epsilon \end{cases}$$

For bridging resolution, as we have four relations, we model it as a multiclass classification task for each span pair. We represent the bridging relation as a one-hot representation and introduce a new relation type NO-RELATION for span pairs that do not have a bridging relation. The loss for bridging is:

$$y_b(i,j) = \text{softmax}(w_b \cdot \text{FFNN}_b(s_{i,j}))$$
$$L_{bridging} = -\sum_{c=1}^{K_c} \sum_{i=1}^{N} \sum_{j=1}^{i} b_{i,j,c} \log(y_b(i,j,c))$$

where K_c represents the number of bridging categories, $y_b(i,j,c)$ denotes the prediction of $y_b(i,j)$ under category c, and:

$$b_{i,j,c} = \begin{cases} 0 & \text{span pair}(i,j) \notin \text{GOLD}_b(c) \\ 1 & \text{span pair}(i,j) \in \text{GOLD}_b(c) \end{cases}$$

where $\text{GOLD}_b(c)$ is the gold bridging relation under category c.

Relation	Method	P_A	R_A	F_A	P_R	R_R	F_R
Coref. (Surface)	coreference	84.9	**50.0**	**62.9**	73.7	**41.9**	53.4
	joint_train	**89.4**	45.8	60.5	**81.7**	40.6	**54.2**
Coref. (Atom)	coreference	84.9	**50.0**	**62.9**	75.6	**42.6**	54.4
	joint_train	**89.4**	45.8	60.5	**82.3**	40.8	**54.5**
Bridging	bridging	88.4	80.9	84.5	76.0	65.4	70.3
	joint_train	**89.5**	**81.8**	**85.5**	**77.0**	**66.1**	**71.1**
TR	bridging	**77.5**	63.8	69.7	**76.2**	63.8	69.1
	joint_train	76.9	**69.0**	**72.7**	75.9	**69.0**	**72.3**
RA	bridging	82.7	83.3	83.0	66.0	57.5	61.4
	joint_train	**89.0**	**85.0**	**86.9**	**70.8**	**60.5**	**65.1**
WU	bridging	**92.0**	82.5	**87.0**	**81.1**	**68.5**	**74.3**
	joint_train	91.6	**82.7**	86.9	79.4	67.9	73.1
CT	bridging	**100.0**	**88.9**	**94.1**	72.1	**79.4**	75.4
	joint_train	95.8	85.2	90.2	**89.4**	78.4	**83.4**
Overall	joint_train	89.5	70.6	78.9	77.5	61.6	68.6

Table 2: Anaphora resolution results over the test dataset (%). Models are trained for "coreference", "bridging" or "joint_train" (both tasks jointly). Models were trained over 10,000 epochs, and averaged over 3 runs with different random seeds. "F_A" and "F_R" denote the F1 score for anaphor and relation prediction, respectively.

The total loss is $L = L_{mention} + L_{ref}$, where:

$$L_{ref} = \begin{cases} L_{coref} & \text{for coreference} \\ L_{bridging} & \text{for bridging} \\ L_{coref} + L_{bridging} & \text{for joint training} \end{cases}$$

6 Experiments

In this section, we detail our experiments. We use similar hyperparameters to Lee et al. (2018). Specifically, we use GloVe embeddings (Pennington et al., 2014) with window size=2 for head word embeddings. For BiLSTM, GloVe embeddings with window size=10 and contextualized ELMo word representations (Peters et al., 2018) are used. Character embeddings are learned from a character CNN with windows of 3, 4, and 5 characters, each with 50 filters. For bridging prediction, the feed-forward neural networks are composed of two hidden layers with 150 dimensions and rectified linear units (Nair and Hinton, 2010).

We separate the gold mentions into those for coreference and bridging. For joint training, the gold mentions are combined.

Table 2 presents the results. For coreference evaluation, given that the results in Table 2 indicate that the surface and atom coreference results are not substantially different, we use surface coreference as our primary evaluation metric in the remainder of this paper. For bridging evaluation, we consider the overall bridging result as our primary analysis.

Overall, the joint training configuration achieves 54.2% F_1 score for coreference resolution and 71.1% F_1 score for bridging, representing +0.8% and +1.2% F_1 score absolute improvement over the component-wise models. This indicates that joint training improves the performance of both tasks. Compared to bridging, the performance of anaphor detection in coreference resolution is lower, particularly in terms of recall, possibly because the data is sparser.

Relation	Method	F_A	F_R
Coref.	coreference	62.9	53.4
	- w/ oracle mentions	81.7	79.2
	joint_train	60.5	54.2
	- w/ oracle mentions	79.5	74.9
Bridging	bridging	84.5	70.3
	- w/ oracle mentions	91.9	83.3
	joint_train	85.5	71.1
	- w/ oracle mentions	91.8	83.5
Overall	joint_train	78.9	68.6
	- w/ oracle mentions	88.3	82.1

Table 3: Comparisons with providing oracle mention during training; results on test dataset, using surface scoring for coreference. "F_A"= F1 for anaphor prediction; "F_R"= F1 for relation prediction.

To investigate the contribution of each step (mention detection vs. relation detection), we experiment with providing oracle mentions during the training process. Table 3 shows that the performance of both tasks improves substantially with gold mentions. We achieve 82.1% F_1 score for relation prediction result under joint training, with +13.5% F_1 absolute score improvement. That is, further improvement at mention detection will improve resolution results.

Relation	Method	F_A	F_R
Coref.	coreference	62.9	53.4
	- w/ ChELMo	65.3	56.9
	joint_train	60.5	54.2
	- w/ ChELMo	64.4	58.3
Bridging	bridging	84.5	70.3
	- w/ ChELMo	87.8	74.8
	joint_train	85.5	71.1
	- w/ ChELMo	88.7	75.6
Overall	joint_train	78.9	68.6
	- w/ ChELMo	82.2	73.1

Table 4: Comparison of different pretrained embeddings; results over test dataset, using surface scoring for coreference. "F_A"= F1 for anaphor prediction; "F_R"= F1 for relation prediction.

To determine the importance of domain fine-tuning, we also experiment with an ELMo model pretrained on a 1 billion word chemical patent corpus (Zhai et al., 2019), referred to as ChELMo. The experimental results are provided in Table 4. With ChELMo, the performance of anaphor detection and relation detection improve by +3.3% and +4.5% absolute F_1 score, respectively.

We also plot model performance with increasing amounts of training data in Figure 3. While the model performance is starting to plateau, potential gains could be attained with more annotated data. The strong correlation between anaphor detection and relation detection is also self-evident in the graph.

To perform error analysis, we analysed the model errors on the dev dataset. As detailed in Table 1, the corpus contains discontinuous mentions. However, our proposed model only considers continuous spans, accounting for some of the low recall.

For coreference resolution, errors can be attributed to three primary phenomena:

1. **Long-distance relations**: as illustrated in Table 5 Ex 1, *the title compound (360 mg, 1.05 mmol, 32%)* refers to a compound at the beginning of the snippet; the model generally fails to capture such long-distance relations.

2. **Multiple antecedents**: as discussed in Section 3.3.1, an anaphor may have multiple antecedents, however the models predict a single antecedent for each anaphor.

3. **Imbalance of coreference and bridging relations**: bridging is more prevalent than coreference, meaning the model has more difficulty with coreference.

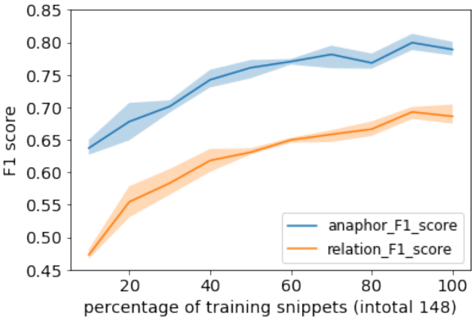
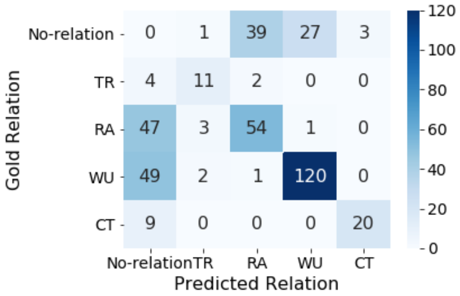


Figure 3: Joint training configuration performance on test dataset over different % of training dataset.



	No-relation	TR	RA	WU	CT
No-relation	0	1	39	27	3
TR	4	11	2	0	0
RA	47	3	54	1	0
WU	49	2	1	120	0
CT	9	0	0	0	20

Figure 4: Confusion matrix of bridging relation detection on dev dataset with joint training configuration

For bridging, as shown in Table 2, the performance suffers from low recall in anaphor detection. Furthermore, the confusion matrix of fine-grained bridging relations in Figure 4 shows that the model achieves poor performance for REACTION-ASSOCIATED and WORK-UP relation prediction, both in terms of precision and recall.

We further investigated the over-prediction problem in bridging. As shown in Table 5 Ex 2, *the reaction mixture* in line 3 has a RELATION-ASSOCIATED link with *The reaction mixture* in line 2 and *sodium borohydride (10 mg, 0.27 mmol)*. The model overpredicts additional links to the two additional compounds that are linked to the previous mention of *The reaction mixture* in line 2. The WORK-UP relation in Ex 5 is similar: the second-mentioned *the organic layer* links to the first-mentioned *the organic layer* and *magnesium sulfate*. *The filtered material*, *chloroform* and *water* should be linked with the first-mentioned *the organic layer*, but are linked to the second. Such errors result from individual span-pair predictions, making it hard to capture interactions between anaphors. Evaluating the antecedents simultane-

1	Step D: Ethyl 7-chloro-6-(difluoromethyl)-2-(trifluoromethyl)pyrazolo[1,5 -a]pyridine-3-carboxylate . ... Purification (FCC, SiO2, eluting with n-hexane:dichloromethane (2:1)) afforded the title compound (360 mg, 1.05 mmol, 32%) .
2	... to a suspension of methyl 5-bromo-6-methoxypicolinate (20 mg, 0.079 mmol) (Ark Pharm) in ethanol (0.25 mL) was added sodium borohydride (9.6 mg, 0.25 mmol) . The reaction mixture was then heated at 50^{o}C. for 2.5 h. An additional portion of sodium borohydride (10 mg, 0.27 mmol) was added, and the reaction mixture was heated at 50^{o}C. for an additional 2 h...
3	... after 55.8 mg of 6-chloro-7-deazapurine and 191 mg of potassium carbonate were sequentially added into the reaction mixture , the reaction mixture was refluxed for about 36 hours and then cooled down at room temperature...
4	... In the same manner as in Synthesis Example 8 except for using 2.11 g of the intermediate 6 in place of the intermediate 21 and using 1.00 g of 4-bromobiphenyl in place of bromobenzene, 1.49 g (yield: 56%) of a white solid was obtained.,.
5	... The filtered material was extracted with chloroform and water , and then the organic layer was dried by using magnesium sulfate . Thereafter, the organic layer was distilled under reduced pressure...
6	... acetonitrile (150 mL) was added under ultrasonic to get a large amount of with precipitate . After suck filtration, the filter cake was washed with acetonitrile (20 mLx3) , dried in vacuum to obtain the title compound (1.52 g, 86.9 %) .

Table 5: Examples of anaphora phenomena from the dev dataset.

ously may address this.

There is room for improvement in our model's ability to model context. In Table 5 Ex 3, due to the expression *add into*, the first-mentioned *the reaction mixture* does not include the chemicals mentioned prior, unlike the first mention of the phrase in Ex 2.

There are several causes of false negatives:

1. **Reaction description variation:** Chemical reactions are usually described step by step, and our model performs well in this structure. However, only part of a reaction may be described. Table 5 Ex 4 illustrates chemical compounds that are listed without a process.

2. **Abstract expressions:** In Table 5 Ex 6, *precipitate* should have a WORK-UP relation with *acetonitrile (150 mL)*, and *the title compound* ... with *the filter cake*; these are missed due to inadequate modelling of domain terminology.

7 Conclusion

We propose a novel annotation scheme for anaphora resolution in chemical patents. For our annotation, we incorporate generic and domain-specific knowledge to define coreference and bridging specific to the chemical domain, based on which we created the novel ChEMU-Ref dataset. Our corpus analysis and inner-annotator agreement show the complexity of the task, as well as the high quality of annotation. We model anaphora resolution as two sub-tasks, mention detection and anaphora relation detection, and also propose a joint training model, which outperforms the separately-trained models. By incorporating embeddings pretrained on the chemical domain, we found that domain knowledge boosts performance. With detailed error analysis, we also identified directions to further enhance performance.

Acknowledgements

Funding for the ChEMU project is provided by an Australian Research Council Linkage Project, project number LP160101469, and Elsevier. A graduate research scholarship is provided by Melbourne School of Engineering to Biaoyan Fang. We would also like to thank Dr. Meladel Mistica and our two chemical expert annotators Colleen Yeow Hui Shiuan and Sacha Novakovic for their contributions to refining the annotation guidelines.

References

Saber A Akhondi, Hinnerk Rey, Markus Schwörer, Michael Maier, John Toomey, Heike Nau, Gabriele Ilchmann, Mark Sheehan, Matthias Irmer, Claudia Bobach, Marius Doornenbal, Michelle Gregory, and Jan A Kors. 2019. Automatic identification of relevant chemical compounds from patents. *Database*, 2019.

Nicholas Asher and Alex Lascarides. 1998. Bridging. *Journal of Semantics*, 15(1):83–113.

Michael Bada, Miriam Eckert, Donald Evans, Kristin Garcia, Krista Shipley, Dmitry Sitnikov, William A Baumgartner, K Bretonnel Cohen, Karin Verspoor, Judith A Blake, and Lawrence E Hunter. 2012. Concept annotation in the CRAFT corpus. *BMC Bioinformatics*, 13(1):161.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, USA.

Kevin Clark and Christopher D Manning. 2015. Entity-centric coreference resolution with model stacking. In *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language*

Processing (Volume 1: Long Papers), pages 1405–1415, Beijing, China.

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *Proc. of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262, Austin, USA.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany.

K Bretonnel Cohen, Arrick Lanfranchi, Miji Joo-young Choi, Michael Bada, William A Baumgartner, Natalya Panteleyeva, Karin Verspoor, Martha Palmer, and Lawrence E Hunter. 2017. Coreference annotation and resolution in the colorado richly annotated full text (CRAFT) corpus of biomedical journal articles. *BMC Bioinformatics*, 18(1):372.

Biaoyan Fang, Christian Druckenbrodt, Colleen Yeow Hui Shiuan, Sacha Novakovic, Ralph Hössel, Saber A. Akhondi, Jiayuan He, Meladel Mistica, Timothy Baldwin, and Karin Verspoor. 2021. ChEMU-Ref dataset for modeling anaphora resolution in the chemical domain. Mendeley Data.

Abbas Ghaddar and Philippe Langlais. 2016a. Wikicoref: An English coreference-annotated corpus of Wikipedia articles. In *Proc. of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 136–142, Portorož, Slovenia.

Abbas Ghaddar and Phillippe Langlais. 2016b. WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proc. of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, Portorož, Slovenia.

Loïc Grobol. 2019. Neural coreference resolution with limited lexical context and explicit mention detection for oral French. In *Proc. of the Second Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 8–14, Minneapolis, USA.

P Gwynne and G Heabrer. 2015. Recent developments in drug discovery: Improvements in efficiency. *Science*.

Jiayuan He, Biaoyan Fang, Hiyori Yoshikawa, Yuan Li, Saber A Akhondi, Christian Druckenbrodt, Camilo Thorne, Zubair Afzal, Zenan Zhai, Lawrence Cavedon, Trevor Cohn, Timothy Baldwin, and Karin Verspoor. to appear. ChEMU 2021: Reaction reference resolution and anaphora resolution in chemical patents. In *Proc. of the 43rd European Conference on Information Retrieval*, online.

Jiayuan He, Dat Quoc Nguyen, Saber A. Akhondi, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Zubair Afzal, Zenan Zhai, Biaoyan Fang, Hiyori Yoshikawa, Ameer Albahem, Lawrence Cavedon, Trevor Cohn, Timothy Baldwin, and Karin Verspoor. 2020. Overview of ChEMU 2020: Named entity recognition and event extraction of chemical reactions from patents. In *Proc. of the Eleventh International Conference of the CLEF Association (CLEF 2020)*, online.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yufang Hou. 2018a. A deterministic algorithm for bridging anaphora resolution. In *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1938–1948, Brussels, Belgium.

Yufang Hou. 2018b. Enhanced word representations for bridging anaphora resolution. In *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 1–7, New Orleans, USA.

Yufang Hou. 2020. Bridging anaphora resolution as question answering. In *Proc. of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1428–1438, Online.

Yufang Hou, Katja Markert, and Michael Strube. 2014. A rule-based system for unrestricted bridging resolution: Recognizing bridging anaphora and finding links to antecedents. In *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 2082–2093, Doha, Qatar.

Yufang Hou, Katja Markert, and Michael Strube. 2018. Unrestricted bridging resolution. *Computational Linguistics*, 44(2):237–284.

Ben Kantor and Amir Globerson. 2019. Coreference resolution with entity equalization. In *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 673–677, Florence, Italy.

Jin-Dong Kim, Ngan Nguyen, Yue Wang, Jun'ichi Tsujii, Toshihisa Takagi, and Akinori Yonezawa. 2012. The Genia event and protein coreference tasks of the BioNLP shared task 2011. *BMC Bioinformatics*, 13(11):S1.

Klaus Krippendorff. 2004. Measuring the reliability of qualitative text analysis data. *Quality and quantity*, 38:787–800.

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

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, USA.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proc. of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (EMNLP 2005)*, pages 25–32, Vancouver, Canada.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proc. of the 33rd International Conference on Machine Learning (ICML 2016)*, New York, USA.

Vincent Ng. 2017. Machine learning for entity coreference resolution: A retrospective look at two decades of research. In *Proc. of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*, pages 4877–4884, San Francisco, USA.

Ngan Nguyen, Jin-Dong Kim, and Jun'ichi Tsujii. 2011. Overview of BioNLP 2011 protein coreference shared task. In *Proc. of BioNLP Shared Task 2011 Workshop*, pages 74–82, Portland, USA.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, USA.

Massimo Poesio, Ron Artstein, et al. 2008. Anaphoric annotation in the ARRAU corpus. In *Proc. of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco.

Massimo Poesio, Roland Stuckardt, and Yannick Versley. 2016. *Anaphora Resolution*. Springer.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proc. of EMNLP-CoNLL 2012: Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–40, Jeju, Korea.

Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Ina Rösiger. 2016. Scicorp: A corpus of English scientific articles annotated for information status analysis. In *Proc. of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1743–1749, Portorož, Slovenia.

Ina Rösiger. 2018a. BASHI: A corpus of Wall Street Journal articles annotated with bridging links. In *Proc. of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Ina Rösiger. 2018b. Rule- and learning-based methods for bridging resolution in the ARRAU corpus. In *Proc. of the First Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 23–33, New Orleans, USA.

Ina Rösiger. 2019. *Computational modelling of coreference and bridging resolution*. Ph.D. thesis, Stuttgart University.

Ina Rösiger, Arndt Riester, and Jonas Kuhn. 2018. Bridging resolution: Task definition, corpus resources and rule-based experiments. In *Proc. of the 27th International Conference on Computational Linguistics*, pages 3516–3528, Santa Fe, USA.

Karin Verspoor, Dat Quoc Nguyen, Saber A. Akhondi, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Jiayuan He, and Zenan Zhai. 2020. ChEMU dataset for information extraction from chemical patents. Mendeley Data.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes release 5.0. Linguistic Data Consortium Catalog No. LDC2013T19.

Sam Wiseman, Alexander M. Rush, Stuart Shieber, and Jason Weston. 2015. Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1416–1426, Beijing, China.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. 2016. Learning global features for coreference resolution. In *Proc. of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004, San Diego, USA.

Juntao Yu and Massimo Poesio. 2020. Multi-task learning based neural bridging reference resolution. *arXiv preprint arXiv:2003.03666*.

Amir Zeldes. 2017. The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Zenan Zhai, Dat Quoc Nguyen, Saber Akhondi, Camilo Thorne, Christian Druckenbrodt, Trevor Cohn, Michelle Gregory, and Karin Verspoor. 2019. Improving chemical named entity recognition in patents with contextualized word embeddings. In *Proc. of the 18th BioNLP Workshop and Shared Task*, pages 328–338, Florence, Italy.

Rui Zhang, Cícero Nogueira dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir Radev. 2018. Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proc. of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107, Melbourne, Australia.

A Additional Experimental Results

In the following tables, we provide detailed experiment results described in the main paper.

Table 6 provides a full comparison of training with gold-standard oracle mentions per anaphora relation on the test dataset.

Table 7 provides a full comparison of training with different pretrained embeddings per anaphora relation on the test dataset.

Relation	Method	P_A	R_A	F_A	P_R	R_R	F_R
Coref. (Surface)	coreference	84.9	50.0	62.9	73.7	41.9	53.4
	- w/ oracle mentions	86.0	78.1	81.7	84.8	74.5	79.2
	joint_train	89.4	45.8	60.5	81.7	40.6	54.2
	- w/ oracle mentions	90.5	70.8	79.5	87.0	65.9	74.9
Coref. (Atom)	coreference	84.9	50.0	62.9	75.6	42.6	54.4
	- w/ oracle mentions	86.0	78.1	81.7	85.1	74.7	79.5
	joint_train	89.4	45.8	60.5	82.3	40.8	54.5
	- w/ oracle mentions	90.5	70.8	79.5	88.7	66.3	75.9
Bridging	bridging	88.4	80.9	84.5	76.0	65.4	70.3
	- w/ oracle mentions	91.1	92.8	91.9	83.8	82.8	83.3
	joint_train	89.5	81.8	85.5	77.0	66.1	71.1
	- w/ oracle mentions	91.3	92.4	91.8	82.8	84.3	83.5
TR	bridging	77.5	63.8	69.7	76.2	63.8	69.1
	- w/ oracle mentions	90.2	90.8	90.3	90.2	90.8	90.3
	joint_train	76.9	69.0	72.7	75.9	69.0	72.3
	- w/ oracle mentions	91.5	86.2	88.6	90.6	86.2	88.1
RA	bridging	82.7	83.3	83.0	66.0	57.5	61.4
	- w/ oracle mentions	88.0	88.3	88.1	83.4	71.1	76.7
	joint_train	89.0	85.0	86.9	70.8	60.5	65.1
	- w/ oracle mentions	85.4	93.9	89.4	78.0	76.6	77.3
WU	bridging	92.0	82.5	87.0	81.1	68.5	74.3
	- w/ oracle mentions	92.4	94.4	93.4	83.7	86.7	85.2
	joint_train	91.6	82.7	86.9	79.4	67.9	73.1
	- w/ oracle mentions	93.7	92.6	93.1	85.2	86.9	86.0
CT	bridging	100.0	88.9	94.1	72.1	79.4	75.4
	- w/ oracle mentions	93.3	100.0	96.5	79.5	100.0	88.3
	joint_train	95.8	85.2	90.2	89.4	78.4	83.4
	- w/ oracle mentions	90.6	100.0	94.9	71.9	100.0	83.3
Overall	joint_train	89.5	70.6	78.9	77.5	61.6	68.6
	- w/ oracle mentions	91.1	85.7	88.3	83.4	81.0	82.1

Table 6: Test results with gold-standard mentions during training. Models trained for “coreference”, “bridging” or “joint_train” (both tasks jointly). Models trained over 10,000 epochs; averaged over 3 runs with different random seeds. “F_A” and “F_R” denote the F1 score for anaphor and relation prediction, respectively.

Relation	Method	P_A	R_A	F_A	P_R	R_R	F_R
Coref. (Surface)	coreference	84.9	50.0	62.9	73.7	41.9	53.4
	- w/ ChELMo	86.5	52.5	65.3	76.9	45.2	56.9
	joint_train	89.4	45.8	60.5	81.7	40.6	54.2
	- w/ ChELMo	87.2	51.1	64.4	80.6	45.7	58.3
Coref. (Atom)	coreference	84.9	50.0	62.9	75.6	42.6	54.4
	- w/ ChELMo	86.5	52.5	65.3	81.1	46.5	59.0
	joint_train	89.4	45.8	60.5	82.3	40.8	54.5
	- w/ ChELMo	87.2	51.1	64.4	81.5	46.0	58.8
Bridging	bridging	88.4	80.9	84.5	76.0	65.4	70.3
	- w/ ChELMo	88.4	87.2	87.8	75.9	73.7	74.8
	joint_train	89.5	81.8	85.5	77.0	66.1	71.1
	- w/ ChELMo	91.3	86.4	88.7	78.1	73.4	75.6
TR	bridging	77.5	63.8	69.7	76.2	63.8	69.1
	- w/ ChELMo	82.5	65.5	73.0	81.4	65.5	72.5
	joint_train	76.9	69.0	72.7	75.9	69.0	72.3
	- w/ ChELMo	81.4	64.4	71.8	79.3	64.4	70.9
RA	bridging	82.7	83.3	83.0	66.0	57.5	61.4
	- w/ ChELMo	89.5	83.9	86.5	74.1	62.9	68.0
	joint_train	89.0	85.0	86.9	70.8	60.5	65.1
	- w/ ChELMo	91.8	82.2	86.6	76.8	63.7	69.5
WU	bridging	92.0	82.5	87.0	81.1	68.5	74.3
	- w/ ChELMo	88.3	92.4	90.3	76.2	79.0	77.5
	joint_train	91.6	82.7	86.9	79.4	67.9	73.1
	- w/ ChELMo	92.2	92.0	92.1	78.5	78.2	78.3
CT	bridging	100.0	88.9	94.1	72.1	79.4	75.4
	- w/ ChELMo	100.0	85.2	91.7	78.5	78.4	78.4
	joint_train	95.8	85.2	90.2	89.4	78.4	83.4
	- w/ ChELMo	100.0	81.5	89.4	80.8	80.4	80.3
Overall	joint_train	89.5	70.6	78.9	77.5	61.6	68.6
	- w/ ChELMo	90.4	75.3	82.2	78.4	68.5	73.1

Table 7: Results with different pretrained embeddings. "coreference", "bridging" and "joint_training" represent models that are trained on the coreference resolution task, bridging task, and both tasks jointly, respectively. We train the models over 10,000 epochs, and averages over 3 runs with different random seeds. "F_A" and "F_R" denote the F1 score for anaphor and relation prediction, respectively.

Syntactic Nuclei in Dependency Parsing – A Multilingual Exploration

Ali Basirat
Uppsala University
Dept. of Linguistics and Philology
`ali.basirat@lingfil.uu.se`

Joakim Nivre
Uppsala University
Dept. of Linguistics and Philology
`joakim.nivre@lingfil.uu.se`

Abstract

Standard models for syntactic dependency parsing take words to be the elementary units that enter into dependency relations. In this paper, we investigate whether there are any benefits from enriching these models with the more abstract notion of nucleus proposed by Tesnière. We do this by showing how the concept of nucleus can be defined in the framework of Universal Dependencies and how we can use composition functions to make a transition-based dependency parser aware of this concept. Experiments on 12 languages show that nucleus composition gives small but significant improvements in parsing accuracy. Further analysis reveals that the improvement mainly concerns a small number of dependency relations, including nominal modifiers, relations of coordination, main predicates, and direct objects.

1 Introduction

A syntactic dependency tree consists of directed arcs, representing syntactic relations like subject and object, connecting a set of nodes, representing the elementary syntactic units of a sentence. In contemporary dependency parsing, it is generally assumed that the elementary units are word forms or tokens, produced by a tokenizer or word segmenter. A consequence of this assumption is that the shape and size of dependency trees will vary systematically across languages. In particular, morphologically rich languages will typically have fewer elementary units and fewer relations than more analytical languages, which use independent function words instead of morphological inflection to encode grammatical information. This is illustrated in Figure 1, which contrasts two equivalent sentences in English and Finnish, annotated with dependency trees following the guidelines of Universal Dependencies (UD) (Nivre et al., 2016, 2020), which assume words as elementary units.

An alternative view, found in the seminal work of Tesnière (1959), is that dependency relations hold between slightly more complex units called *nuclei*, semantically independent units consisting of a content word together with its grammatical markers, regardless of whether the latter are realized as independent words or not. Thus, a nucleus will often correspond to a single word – as in the English verb *chased*, where tense is realized solely through morphological inflection – but it may also correspond to several words – as in the English verb group *has chased*, where tense is realized by morphological inflection in combination with an auxiliary verb. The latter type is known as a *dissociated nucleus*. If we assume that the elementary syntactic units of a dependency tree are nuclei rather than word forms, then the English and Finnish sentences will have the same dependency trees, visualized in Figure 2, and will differ only in the realization of their nuclei. In particular, while nominal nuclei in Finnish are consistently realized as single nouns inflected for case, the nominal nuclei in English involve standalone articles and the preposition *from*.

In this paper, we set out to investigate whether research on dependency parsing can benefit from making explicit use of Tesnière's notion of nucleus, from the point of view of accuracy, interpretability and evaluation. We do this from a multilingual perspective, because it is likely that the effects of introducing nuclei will be different in different languages, and we strongly believe that a comparison between different languages is necessary in order to assess the potential usefulness of this notion. We are certainly not the first to propose that Tesnière's notion of nucleus can be useful in parsing. One of the earliest formalizations of dependency grammar for the purpose of statistical parsing, that of Samuelsson (2000), had this notion at its core, and Sangati and Mazza (2009) presented a conversion of the Penn Treebank of English to Tesnière style

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1376–1387
April 19 - 23, 2021.

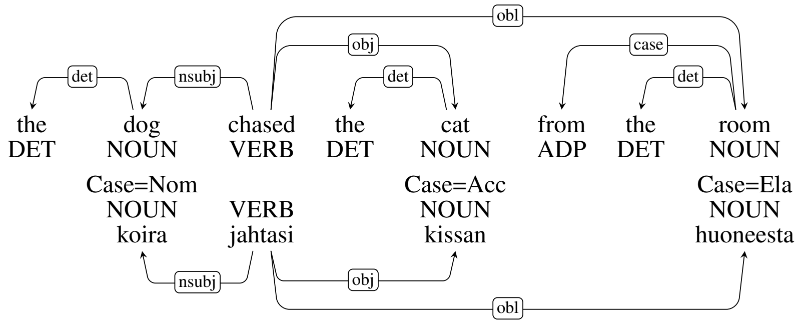


Figure 1: Word-based dependency trees for equivalent sentences from English (top) and Finnish (bottom).

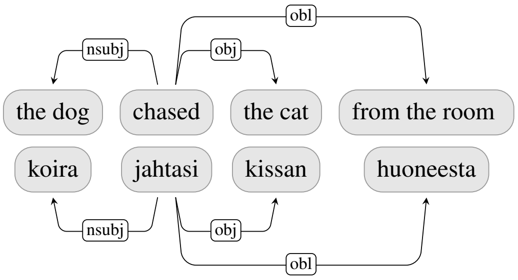


Figure 2: Nucleus-based dependency trees for equivalent sentences from English (top) and Finnish (bottom).

representations, including nuclei. However, previous attempts have been hampered by the lack of available parsers and resources to test this hypothesis on a large scale. Thus, the model of Samuelsson (2000) was never implemented, and the treebank conversion of Sangati and Mazza (2009) is available only for English and in a format that no existing dependency parser can handle. We propose to overcome these obstacles in two ways. On the resource side, we will rely on UD treebanks and exploit the fact that, although the annotation is word-based, the guidelines prioritize dependency relations between content words that are the cores of syntactic nuclei, which facilitates the recognition of dissociated nuclei and gives us access to annotated resources for a wide range of languages. On the parsing side, we will follow a transition-based approach, which can relatively easily be extended to include operations that create representations of syntactic nuclei, as previously shown by de Lhoneux et al. (2019a), something that is much harder to achieve in a graph-based approach.

2 Related Work

Dependency-based guidelines for syntactic annotation generally discard the nucleus as the basic syntactic unit in favor of the (orthographic) word form, possibly with a few exceptions for fixed multiword expressions. A notable exception is the three-layered annotation scheme of the Prague Dependency Treebank (Hajič et al., 2000), where nucleus-like concepts are captured at the tectogrammatical level according to the Functional Generative Description (Sgall et al., 1986). Bārzdiņš et al. (2007) propose a syntactic analysis model for Latvian based on the x-word concept analogous to the nucleus concept. In this grammar, an x-word acts as a non-terminal symbol in a phrase structure grammar and can appear as a head or dependent in a dependency tree. Nespore et al. (2010) compare this model to the original dependency formalism of Tesnière (1959). Finally, as already mentioned, Sangati and Mazza (2009) develop an algorithm to convert English phrase structure trees to Tesnière style representations.

When it comes to syntactic parsing, Järvinen and Tapanainen (1998) were pioneers in adapting Tesnière's dependency grammar for computational processing. They argue that the nucleus concept is crucial to establish cross-linguistically valid criteria for headedness and that it is not only a syntactic primitive but also the smallest semantic unit in a lexicographical description. As an alternative to the rule-based approach of Järvinen and Tapanainen (1998), Samuelsson (2000) defined a generative statistical model for nucleus-based dependency parsing, which however was never implemented.

The nucleus concept has affinities with the chunk concept found in many approaches to parsing, starting with Abney (1991), who proposed to first find chunks and then dependencies between chunks, an idea that was generalized into cascaded parsing by Buchholz et al. (1999) among others. It is also clearly related to the vibhakti level in the Paninian computation grammar framework (Bharati and Sangal, 1993; Bharati et al., 2009). In a similar vein, Kudo and Matsumoto (2002) use cascaded chunking for dependency parsing of Japanese, Tongchim et al. (2008) show that base-NP chunking can significantly improve the accuracy of dependency parsing for Thai, and Durgar El-Kahlout et al. (2014) show that chunking improves dependency parsing of Turkish. Das et al. (2016) study the importance of chunking in the transfer parsing model between Hindi and Bengali, and Lacroix (2018) show that NP chunks are informative for universal part-of-speech tagging and dependency parsing.

In a more recent study, de Lhoneux et al. (2019b) investigate whether the hidden representations of a neural transition-based dependency parser encodes information about syntactic nuclei, with special reference to verb groups. They find some evidence that this is the case, especially if the parser is equipped with a mechanism for recursive subtree composition of the type first proposed by Stenetorp (2013) and later developed by Dyer et al. (2015) and de Lhoneux et al. (2019a). The idea is to use a composition operator that recursively combines information from subtrees connected by a dependency relation into a representation of the new larger subtree. In this paper, we will exploit variations of this technique to create parser-internal representations of syntactic nuclei, as discussed in Section 4. However, first we need to discuss how to identify nuclei in UD treebanks.

3 Syntactic Nuclei in UD

UD[1] (Nivre et al., 2016, 2020) is an ongoing project aiming to provide cross-linguistically consistent morphosyntactic annotation of many languages around the world. The latest release (v2.7) contains 183 treebanks, representing 104 languages and 20 language families. The syntactic annotation in UD is based on dependencies and the elementary syntactic units are assumed to be words, but the style of the annotation makes it relatively straightforward to identify substructures corresponding to (dissociated) nuclei. More precisely, UD prioritizes direct dependency relations between content words, as opposed to relations being mediated by function words, which has two consequences. First, incoming dependencies always go to the lexical core of a nucleus.[2] Second, function words are normally leaves of the dependency tree, attached to the lexical core with special dependency relations, which we refer to as functional relations.[3]

Figure 3 illustrates these properties of UD representations by showing the dependency tree for the English sentence *This killing of a respected cleric will be causing us trouble for years to come* with functional relations drawn below the sentence and other relations above. Given this type of representation, we can define a *nucleus* as a subtree where all internal dependencies are functional relations, as indicated by the ovals in Figure 3. The nuclei can be divided into single-word nuclei, whitened, and dissociated nuclei, grayed. The latter can be contiguous or discontiguous, as shown by the nucleus *of a cleric*, which consists of the two parts colored with a darker shade.

This definition of nucleus in turn depends on what we define to be functional relations. For this study, we assume that the following 7 UD relations[4] belong to this class:

- Determiner (*det*): the relation between a determiner, mostly an article or demonstrative, and a noun. Especially for articles, there is considerable cross-linguistic variation. For example, definiteness is expressed by an independent function word in English (*the girl*), by a morphological inflection in Swedish (*flicka-n*), and not at all in Finnish.
- Case marker (*case*): the relation between a noun and a case marker when it is a separate syntactic word and not an affix. UD takes a radical approach to adpositions and treats them all as case markers. Thus, in Figure 1, we see that the English adposition *from* corresponds to the Finnish elative case inflection.
- Classifier (*clf*): the relation between a classifier, a counting unit used for conceptual classification of nouns, and a noun. This relation is seen in languages which have a classification system such as Chinese. For example, English *three students* corresponds to Chinese 三个学生, literally “three [human-classifier] student”.
- Auxiliary (*aux*): the relation between an auxiliary verb or nonverbal TAME marker and a verbal predicate. An example is the English verb group *will be causing* in Figure 3, which

---

[1] https://universaldependencies.org

[2] Except in some cases of ellipsis, like *she did*, where the auxiliary verb *did* is “promoted” to form a nucleus on its own.

[3] Again, there are a few well-defined exceptions to the rule that function words are leaves, including ellipsis, coordination, and fixed multiword expressions.

[4] A more detailed description of the relations is available in the UD documentation at https://universaldependencies.org.

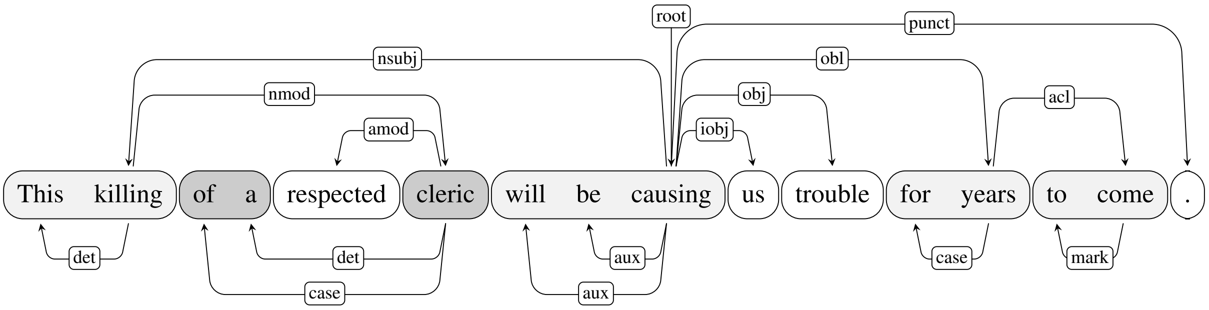


Figure 3: Syntactic UD representation with functional relations drawn below the sentence. Dissociated nuclei are grayed, with a darker shade for the discontiguous nucleus.

alternates with finite main verbs like *causes* and *caused*.

- Copula (*cop*): the relation between a verbal or nonverbal copula and a nonverbal predicate. For example, in English *Ivan is the best dancer*, the copula *is* links the predicate *the best dancer* to *Ivan*, but it has no counterpart in Russian *Ivan lucšĭj tancor*, literally "Ivan best dancer".
- Subordination marker (*mark*): the relation between a subordinator and the predicate of a subordinate clause. This is exemplified by the infinitive marker *to* in Figure 3. Other examples are subordinating conjunctions like *if*, *because* and *that*, the function of which may be encoded morphologically or through word order in other languages.
- Coordinating conjunction (*cc*): the relation between a coordinator and a conjunct (typically the last one) in a coordination. Thus, in *apples, bananas and oranges*, UD treats *and* as a dependent of *oranges*. This linking function may be missing or expressed morphologically in other languages.

The inclusion of the *cc* relation among the nucleus-internal relations is probably the most controversial decision, given that Tesnière treated coordination (including coordinating conjunctions) as a third type of grammatical relation – junction (fr. jonction) – distinct from both dependency relations and nucleus-internal relations. However, we think coordinating conjunctions have enough in common with other function words to be included in this preliminary study and leave further division into finer categories for future work.[5]

Given the definition of nucleus in terms of functional UD relations, it would be straightforward to convert the UD representations to dependency trees where the elementary syntactic units are nuclei rather than words. However, the usefulness of such a resource would currently be limited, given that it would require parsers that can deal with nucleus recognition, either in a preprocessing step or integrated with the construction of dependency trees, and such parsers are not (yet) available. Moreover, evaluation results would not be comparable to previous research. Therefore, we will make use of the nucleus concept in UD in three more indirect ways:

- **Evaluation:** Even if a parser outputs a word-based dependency tree in UD format, we can evaluate its accuracy on nucleus-based parsing by simply not scoring the functional relations. This is equivalent to the Content Labeled Attachment Score (CLAS) previously proposed by Nivre and Fang (2017), and we will use this score as a complement to the standard Labeled Attachment Score (LAS) in our experiments.[6]
- **Nucleus Composition:** Given our definition of nucleus-internal relations, we can make parsers aware of the nucleus concept by differentiating the way they predict and represent dissociated nuclei and dependency structures, respectively. More precisely, we will make use of composition operations to create internal representations of (dissociated) nuclei, as discussed in detail in Section 4 below.
- **Oracle Parsing:** To establish an upper bound on what a nucleus-aware parser can achieve, we will create a version of the UD representation which is still a word-based dependency

---

[5] In addition to separating the *cc* relation from the rest, such a division might include distinguishing nominal nucleus relations (*det*, *case* and *clf*) from predicate nucleus relations (*aux*, *cop* and *mark*).

[6] Our use of CLAS differs only in that we include punctuation in the evaluation, whereas Nivre and Fang (2017) excluded it.

tree, but where nuclei are explicitly represented by letting the word form for each nucleus core be a concatenation of all the word forms that are part of the nucleus.[7] We call this oracle parsing to emphasize that the parser has oracle information about the nuclei of a sentence, although it still has to predict all the syntactic relations.

4 Syntactic Nuclei in Transition-Based Dependency Parsing

A transition-based dependency parser derives a dependency tree from the sequence of words forming a sentence (Yamada and Matsumoto, 2003; Nivre, 2003, 2004). The parser constructs the tree incrementally by applying transitions, or parsing actions, to configurations consisting of a stack S of partially processed words, a buffer B of remaining input words, and a set of dependency arcs A representing the partially constructed dependency tree. The process of parsing starts from an initial configuration and ends when the parser reaches a terminal configuration. The transitions between configurations are predicted by a history-based model that combines information from S, B and A.

For the experiments in this paper, we use a version of the arc-hybrid transition system initially proposed by Kuhlmann et al. (2011), where the initial configuration has all words $w_1, \ldots, w_n$ plus an artificial root node r in B, while S and A are empty.[8] There are four transitions: Shift, Left-Arc, Right-Arc and Swap. Shift pushes the first word b_0 in B onto S (and is not permissible if $b_0 = r$). Left-Arc attaches the top word s_0 in S to b_0 and removes s_0 from S, while Right-Arc attaches s_0 to the next word s_1 in S and removes s_0 from S. Swap, finally, moves s_1 back to B in order to allow the construction of non-projective dependencies.[9]

Our implementation of this transition-based parsing model is based on the influential architecture of Kiperwasser and Goldberg (2016), which takes as input a sequence of vectors $x_1, \ldots, x_n$ representing the input words $w_1, \ldots, w_n$ and feeds these vectors through a BiLSTM that outputs contextualized word vectors $v_1, \ldots, v_n$, which are stored in the buffer B. Parsing is then performed by iteratively applying the transition predicted by an MLP taking as input a small number of contextualized word vectors from the stack S and the buffer B. More precisely, in the experiments reported in this paper, the predictions are based on the two top items s_0 and s_1 in S and the first item b_0 in B. In a historical perspective, this may seem like an overly simplistic prediction model, but recent work has shown that more complex feature vectors are largely superfluous thanks to the BiLSTM encoder (Shi et al., 2017; Falenska and Kuhn, 2019).

The transition-based parser as described so far does not provide any mechanism for modeling the nucleus concept. It is a purely word-based model, where any more complex syntactic structure is represented internally by the contextualized vector of its head word. Specifically, when two substructures h and d are combined in a Left-Arc or Right-Arc transition, only the vector v_h representing the syntactic head is retained in S or B, while the vector v_d representing the syntactic dependent is removed from S. In order to make the parser sensitive to (dissociated) nuclei in its internal representations, we follow de Lhoneux et al. (2019a) and augment the Right-Arc and Left-Arc actions with a composition operation. The idea is that, whenever the substructures h and d are combined with label l, we replace the current representation of h with the output of a function $f(h, d, l)$. We can then control the information flow for nuclei and other constructions through the definition of $f(h, d, l)$.

Hard Composition: The simplest version, which we call *hard* composition, is to explicitly condition the composition on the dependency label l. In this setup, $f(h, d, l)$ combines the head and dependent vectors only if l is a functional relation and simply returns the head vector otherwise:

$$f(h,d,l) = \begin{cases} \vec{h} \circ \vec{d} & \text{if } l \in F \\ \vec{h} & \text{otherwise} \end{cases} \quad (1)$$

We use $\vec{x}$ to denote the vector representation of x[10] and F to denote the set of seven functional relations defined in Section 3. The composition operator $\circ$ can be any function of the form $\mathbb{R}^n \times \mathbb{R}^n \rightarrow$

---

[7] The English sentence in Figure 1 thus becomes: *the dog-the chased the cat-the from the room-the-from*.

[8] Positioning the artificial root node at the end of the buffer is a modification of the original system by Kiperwasser and Goldberg (2016), inspired by the results reported in Ballesteros and Nivre (2013).

[9] This extension of the arc-hybrid system was proposed by de Lhoneux et al. (2017b), inspired by the corresponding extension of the arc-standard system by Nivre (2009).

[10] In the baseline parser, $\vec{h}$ is always identical to the contextualized representation v_h of the head word w_h, but after introducing composition operations we need a more abstract notation.

$\mathbb{R}^n$, e.g., vector addition $\vec{h} + \vec{d}$, where n is the dimensionality of the vector space.

Soft Composition: The *soft* composition is similar to the hard composition, but instead of applying the composition operator to the head and dependent vectors, the operator is applied to the head vector and a vector representation of the entire dependency arc (h, d, l). The vector representation of the dependency arc is trained by a differentiable function g that encodes the dependency label l into a vector $\vec{l}$ and maps the triple $(\vec{h}, \vec{d}, \vec{l})$ to a vector space, i.e., $g : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ where n and m are the dimensionalities of the word and label spaces, respectively. An example of g is a perceptron with a sigmoid activation that maps the vector representations of h, d and l to a vector space:

$$g(h, d, l) = \sigma(W(\vec{h} \odot \vec{d} \odot \vec{l}) + b) \qquad (2)$$

where $\odot$ is the vector concatenation operator. The soft nucleus composition is then:

$$f(h, d, l) = \begin{cases} \vec{h} \circ g(h, d, l), & \text{if } l \in F \\ \vec{h} & \text{otherwise} \end{cases} \qquad (3)$$

The parameters of the function g are trained with the other parameters of the parser.

Generalized Composition: To test our hypothesis that composition is beneficial for dissociated nuclei, we contrast both hard and soft composition to a generalized version of soft composition, where we do not restrict the application to functional relations. In this case, the composition function is:

$$f(h, d, l) = \vec{h} \circ g(h, d, l) \qquad (4)$$

where l can be any dependency label. In this approach, the if-clause in Equation 2 and 3 is eliminated and the parser itself learns in what conditions the composition should be performed. In particular, if the composition operator is addition, and g is a perceptron with a sigmoid activation on the output layer (as in Equation 2), then g operates as a gate that controls the contribution of the dependency elements h, d, and l to the composition. If the composition should not be performed, it returns a vector close to zero.

5 Experiments

In the previous sections, we have shown how syntactic nuclei can be identified in the UD annotation and how transition-based parsers can be made sensitive to these structures in their internal representations through the use of nucleus composition. We now proceed to a set of experiments investigating the impact of nucleus composition on a diverse selection of languages.

5.1 Experimental Settings

We use UUParser (de Lhoneux et al., 2017a; Smith et al., 2018), an evolution of the transition-based dependency parser of Kiperwasser and Goldberg (2016), which was the highest ranked transition-based dependency parser in the CoNLL shared task on universal dependency parsing in 2018 (Zeman et al., 2018). As discussed in Section 4, this is a greedy transition-based parser based on the extended arc-hybrid system of de Lhoneux et al. (2017b). It uses an MLP with one hidden layer to predict transitions between parser configurations, based on vectors representing two items on the stack S and one item in the buffer B. In the baseline model, these items are contextualized word representations produced by a BiLSTM with two hidden layers. The input to the BiLSTM for each word is the concatenation of a randomly initialized word embedding and a character-based representation produced by running a BiLSTM over the character sequence of the word. We use a dimensionality of 100 for the word embedding as well as for the output of the character BiLSTM.

For parsers with composition, we considered various composition operators $\circ$ and functions g. For the former, we tested vector addition, vector concatenation,[11] and perceptron. For the latter we tried a multi-layer perceptron with different activation functions. Based on the results of the preliminary experiments, we selected vector addition for the composition operator $\circ$ and the perceptron with sigmoid activation for the soft composition function g. The inputs to the perceptron consist of two token vectors of size 512 and a relation vector of size 10. The token vectors are the outputs of the BiLSTM layer of the parser and the relation vector is trained by a distinct embedding layer.

All parsers are trained for 50 epochs and all reported results are averaged over 10 runs with different random seeds. Altogether we explore five different parsers:

---

[11] The concatenation operator requires special care to keep vector dimensionality constant. We double the dimensionality of the contextual vectors and fill the extra dimensions with zeros. We then replace the zero part of the second operand with the first operand's non-zero part at composition time.

				LAS					CLAS				
Language	Treebank	Family	Size	Base	Hard	Soft	Gen	Ora	Base	Hard	Soft	Gen	Ora
Arabic	PADT	AA-Semitic	242K	78.0	78.1	78.3	78.2	80.5	74.4	74.6	74.9	74.8	75.8
Basque	BDT	Basque	70K	73.5	73.4	73.9	74.2	78.9	70.9	70.9	71.3	71.8	75.3
Chinese	GSD	Sino-Tibetan	49K	72.1	72.1	72.7	72.2	80.4	68.8	68.7	69.4	69.0	74.6
English	EWT	IE-Germanic	188K	82.5	82.4	82.5	82.4	85.6	78.8	78.7	78.8	78.7	81.0
Finnish	TDT	Uralic	168K	78.5	78.9	79.6	79.9	84.2	77.5	77.8	78.6	78.9	81.8
Hebrew	HTB	AA-Semitic	108K	81.5	81.6	81.8	82.1	83.3	74.5	74.7	74.9	75.3	75.9
Hindi	HDTB	IE-Indic	294K	87.9	88.1	88.4	89.0	89.8	83.9	83.9	84.4	85.4	85.0
Italian	ISDT	IE-Romance	252K	87.4	87.6	87.9	87.5	89.0	81.5	81.6	82.1	81.6	82.8
Japanese	GSD	Japanese	61K	93.4	93.4	93.5	93.4	94.1	89.7	89.7	89.9	89.7	90.5
Korean	GSD	Korean	21K	75.1	75.0	75.4	75.6	76.6	74.9	74.8	75.2	75.4	75.7
Swedish	Talbanken	IE-Germanic	61K	76.9	77.3	77.5	77.6	82.9	73.2	73.5	73.8	74.1	78.2
Turkish	IMST	Turkic	40K	55.6	55.3	56.2	54.8	58.6	53.4	53.0	54.1	52.6	54.8
Average				78.5	78.6	79.0	78.9	82.0	75.1	75.2	75.6	75.6	77.6

Table 1: Parsing accuracy for 5 parsing models evaluated on 12 UD treebanks. Language family includes genus according to WALS for large families (AA = Afro-Asian, IE = Indo-European). LAS = Labeled Attachment Score. CLAS = Content Labeled Attachment Score.

- **Base(line):** No composition.
- **Hard:** Baseline + hard composition.
- **Soft:** Baseline + soft composition.
- **Gen(eralized):** Baseline + gen. composition.
- **Ora(cle):** Baseline trained and tested on explicit annotation of nuclei (see Section 3).

Our experiments are carried out on a typologically diverse set of languages with different degrees of morphosyntactic complexity, as shown in Table 1. The corpus size is the total number of words in each treebank. We use UD v2.3 with standard data splits (Nivre et al., 2018). All evaluation results are on the development sets.[12]

5.2 Results

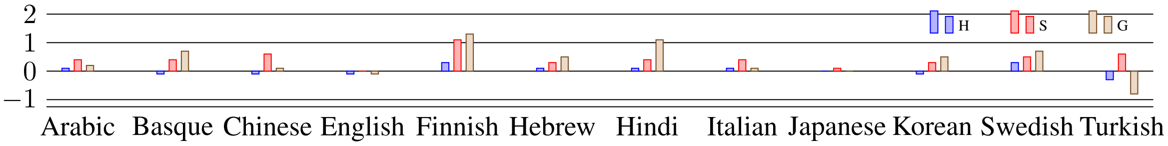
Table 1 reports the parsing accuracy achieved with our 5 parsers on the 12 different languages, using the standard LAS metric as well as the nucleus-aware CLAS metric. First of all, we see that hard composition is not very effective and mostly gives results in line with the baseline parser, except for small improvements for Finnish, Hindi and Swedish and a small degradation for Turkish. These differences are statistically significant for all four languages with respect to LAS but only for Finnish and Turkish with respect to CLAS (two-tailed t-test, $\alpha = .05$). By contrast, soft composition improves accuracy for all languages except English and the improvements are statistically significant for both LAS and CLAS. The average improvement is 0.5 percentage points for both LAS and CLAS, which indicates that most of the improvement occurs on nucleus-external relations thanks to a more effective internal representation of dissociated nuclei. There is some variation across languages, but the CLAS improvement is in the range 0.2–0.7 for most languages, with Finnish as the positive exception (1.1) and English as the negative one (0.0). Generalized composition, finally, where we allow composition also for non-functional relations, yields results very similar to those for soft composition, which could be an indication that the parser learns to apply composition mostly for functional relations. The results are a little less stable, however, with degradations for English and Turkish, and non-significant improvements for Chinese, Italian and Japanese. A tentative conclusion is therefore that composition is most effective when restricted to (but not enforced for) nucleus-internal relations.

Before we try to analyze the results in more detail, it is worth noting that most of the improvements due to composition are far below the improvements of the oracle parser.[13] However, it is important to keep in mind that, whereas the behavior of a composition parser is only affected *after* a nucleus has been constructed, the oracle

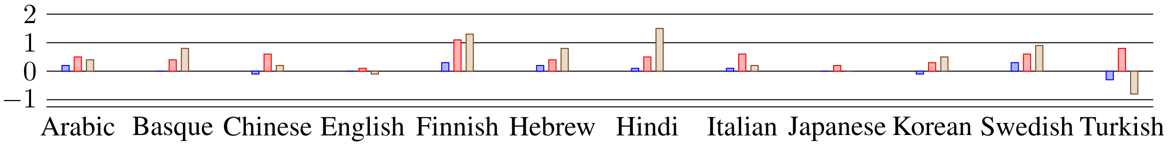
---

[12] Since we want to perform an informative error analysis, we avoid using the dedicated test sets.

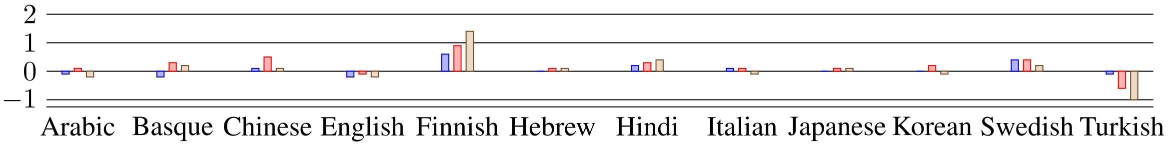
[13] The exception is generalized composition for Hindi, which exceeds the corresponding oracle parser with respect to CLAS.



(a) All relations



(b) Nucleus-external relations



(c) Nucleus-internal relations

Figure 4: Absolute improvement (or degradation) in labeled F-score with respect to the baseline for hard (H), soft (S) and generalized (G) composition for different sets of relations: (a) all relations (corresponding to LAS scores), (b) nucleus-external relations (corresponding to CLAS scores), (c) nucleus-internal relations.

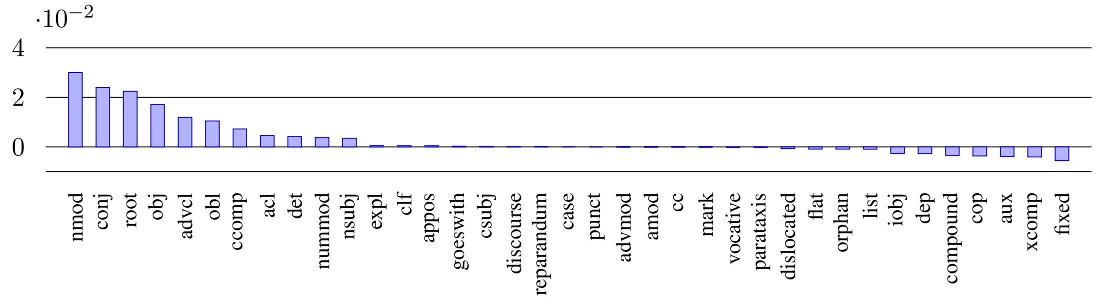


Figure 5: Improvement (or degradation) in labeled F-score with respect to the baseline for different UD relations (soft composition), weighted by the relative frequency of each relation and averaged across all languages.

parser improves also with respect to the prediction of the nuclei themselves. This explains why the oracle parser generally improves more with respect to LAS than CLAS, and sometimes by a substantial margin (2.5 points for Chinese, 1.4 points for Basque and 1.3 points for Swedish).

Figure 4 visualizes the impact of hard, soft and generalized nucleus composition for different languages, with a breakdown into (a) all relations, which corresponds to the difference in LAS compared to the baseline, (b) nucleus-external relations, which corresponds to the difference in CLAS, and (c) nucleus-internal relations. Overall, these graphs are consistent with the hypothesis that using composition to create parser-internal representations of (dissociated) nuclei primarily affects the prediction of nucleus-external relations, as the (a) and (b) graphs are very similar and the (c) graphs mostly show very small differences. There are, however, two notable exceptions. For Finnish, all three composition methods clearly improve the prediction of nucleus-internal relations as well as nucleus-external relations, by over 1 F-score point for generalized composition. Conversely, for Turkish, especially the soft versions of composition has a detrimental effect on the prediction of nucleus-internal relations, reaching 1 F-score point for generalized composition. Turkish is also exceptional in showing opposite effects overall for soft and generalized composition, the former having a positive effect and the latter a negative one, whereas all other languages either show consistent trends or fluctuations around zero. Further research will be needed to explain what causes these deviant patterns.

Figure 5 shows the improvement (or degradation) for individual UD relations, weighted by relative frequency and averaged over all languages, for the best performing soft composition parser.

Finnish		Chinese		Swedish		Turkish		Hindi	
conj	0.09	det	0.03	xcomp	0.07	obj	0.12	obj	0.03
root	0.07	case	0.02	advmod	0.05	conj	0.10	compound	0.02
nmod	0.06	cop	0.01	nmod	0.05	root	0.05	nmod	0.02
obl	0.06	conj	0.01	acl	0.04	acl	0.04	case	0.02
ccomp	0.03	clf	0.01	conj	0.04	nummod	0.03	aux	0.01
acl	0.03	ccomp	0.00	obl	0.04	obl	0.03	det	0.00
obj	0.02	dep	0.00	obj	0.04	nmod	0.01	nummod	0.00
xcomp	0.01	aux	0.00	mark	0.03	ccomp	0.01	amod	0.00
nsubj	0.01	advcl	0.00	nsubj	0.02	det	0.01	advmod	0.00
amod	0.01	cc	0.00	amod	0.02	cc	0.01	advcl	0.00

Table 2: Improvement (or degradation) in labeled F-score, weighted by relative frequency, for the 10 best UD relations in the 5 languages with greatest LAS improvements over the baseline (soft composition).

The most important improvements are observed for *nmod*, *conj*, *root* and *obj*. The *nmod* relation covers all nominal modifiers inside noun phrases, including prepositional phrase modifiers; the *conj* relation holds between conjuncts in a coordination structure; the *root* relation is assigned to the main predicate of a sentence; and *obj* is the direct object relation. In addition, we see smaller improvements for a number of relations, including major clause relations like *advcl* (adverbial clauses), *obl* (oblique modifiers), *ccomp* (complement clauses), and *nsubj* (nominal subjects), as well as noun phrase internal relations like *acl* (adnominal clauses, including relative clauses), *det* (determiner), and *nummod* (numeral modifier). Of these, only *det* is a nucleus-internal relation, so the results further support the hypothesis that richer internal representations of (dissociated) nuclei primarily improve the prediction of nucleus-external dependency relations, especially major clause relations.

It is important to remember that the results in Figure 5 are averaged over all languages and may hide interesting differences between languages. A full investigation of this variation is beyond the scope of this paper, but Table 2 presents a further zoom-in by presenting statistics on the top 10 relations in the 5 languages where LAS improves the most compared to the baseline. To a large extent, we find the same relations as in the aggregated statistics, but there are also interesting language-specific patterns. For Chinese the top three relations (*det*, *case*, *cop*) are all nucleus-internal relations; for Swedish the two top relations are *xcomp* (open clausal complements) and *advmod* (adverbial modifiers), neither of which show positive improvements on average; and for Hindi the *compound* relation shows the second largest improvement. These differences definitely deserve further investigation.

6 Conclusion

We have explored how the concept of syntactic nucleus can be used to enrich the representations of a transition-based dependency parser, relying on UD treebanks for supervision and evaluation in experiments on a wide range of languages. We conclude that the use of composition operations for building internal representations of syntactic nuclei, in particular the technique that we have called soft composition, can lead to small but significant improvements in parsing accuracy for nucleus-external relations, notably for nominal modifiers, relations of coordination, main predicates, and direct objects. In future work we want to study the behavior of different types of nuclei in more detail, in particular how the different internal relations of nominal and verbal nuclei contribute to overall parsing accuracy. We also want to analyze the variation between different languages in more detail and see if it can be explained in terms of typological properties.

Acknowledgments

We thank Daniel Dakota, Artur Kulmizev, Sara Stymne and Gongbo Tang for useful discussions and the EACL reviewers for constructive criticism. We acknowledge the computational resources provided by CSC in Helsinki and Sigma2 in Oslo through NeIC-NLPL (www.nlpl.eu).

References

S. Abney. Parsing by chunks. In R. Berwick, S. Abney, and C. Tenny, editors, *Principle-Based Parsing*, pages 257–278. Kluwer, 1991.

M. Ballesteros and J. Nivre. Going to the roots of dependency parsing. *Computational Linguistics*, 39 (1):5–13, 2013.

G. Bārzdiņš, N. Grūzītis, G. Nešpore, and B. Saulīte. Dependency-based hybrid model of syntactic analysis for the languages with a rather free word order. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA 2007)*, pages 13–20, Tartu, Estonia, May 2007. University of Tartu, Estonia.

A. Bharati and R. Sangal. Parsing free word order languages in the paninian framework. In *Proceedings of the Annual Meeting of Association for Computational Linguistics*, pages 105—-111, 1993.

A. Bharati, S. Husain, D. Misra, and R. Sangal. Two stage constraint based hybrid approach to free word order language dependency parsing. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 77–80, 2009.

S. Buchholz, J. Veenstra, and W. Daelemans. Cascaded grammatical relation assignment. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, pages 239–246, 1999.

A. Das, A. Saha, and S. Sarkar. Cross-lingual transfer parser from Hindi to Bengali using delexicalization and chunking. In *Proceedings of the 13th International Conference on Natural Language Processing*, pages 99–108, Varanasi, India, Dec. 2016. NLP Association of India.

M. de Lhoneux, Y. Shao, A. Basirat, E. Kiperwasser, S. Stymne, Y. Goldberg, and J. Nivre. From raw text to Universal Dependencies – Look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 207–217, 2017a.

M. de Lhoneux, S. Stymne, and J. Nivre. Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 99–104, Pisa, Italy, Sept. 2017b. Association for Computational Linguistics.

M. de Lhoneux, M. Ballesteros, and J. Nivre. Recursive subtree composition in LSTM-based dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1566–1576, Minneapolis, Minnesota, June 2019a. Association for Computational Linguistics.

M. de Lhoneux, S. Stymne, and J. Nivre. What should/do/can LSTMs learn when parsing auxiliary verb constructions? *CoRR*, abs/1907.07950, 2019b. URL `http://arxiv.org/abs/1907.07950`.

İ. Durgar El-Kahlout, A. A. Akın, and E. Yılmaz. Initial explorations in two-phase Turkish dependency parsing by incorporating constituents. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 82–89, Dublin, Ireland, Aug. 2014. Dublin City University.

C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China, July 2015. Association for Computational Linguistics.

A. Falenska and J. Kuhn. The (non-)utility of structural features in BiLSTM-based dependency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 117–128, 2019.

J. Hajič, A. Böhmová, E. Hajičová, and B. Vidová-Hladká. The Prague Dependency Treebank: A Three-Level Annotation Scenario. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Amsterdam:Kluwer, 2000.

T. Järvinen and P. Tapanainen. Towards an implementable dependency grammar. In *Processing of Dependency-Based Grammars*, 1998.

E. Kiperwasser and Y. Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327, 2016.

T. Kudo and Y. Matsumoto. Japanese dependency analysis using cascaded chunking. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, COLING-02, page 1–7, USA, 2002. Association for Computational Linguistics.

M. Kuhlmann, C. Gómez-Rodríguez, and G. Satta. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 673–682, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

O. Lacroix. Investigating NP-chunking with universal dependencies for English. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 85–90, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics.

G. Nespore, B. Saulite, G. Barzdins, and N. Gruzitis. Comparison of the semti-kamols and tesnière's dependency grammars. In *Proceedings of the 4th International Conference on Human Language Technologies — the Baltic Perspective*, 2010.

J. Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 149–160, Nancy, France, Apr. 2003.

J. Nivre. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, July 2004. Association for Computational Linguistics.

J. Nivre. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 351–359, 2009.

J. Nivre and C.-T. Fang. Universal dependency evaluation. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 86–95, 2017.

J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*, 2016.

J. Nivre, M. Abrams, Ž. Agić, L. Ahrenberg, L. Antonsen, K. Aplonova, M. J. Aranzabe, G. Arutie, M. Asahara, L. Ateyah, M. Attia, A. Atutxa, L. Augustinus, E. Badmaeva, M. Ballesteros, E. Banerjee, S. Bank, V. Barbu Mititelu, V. Basmov, J. Bauer, S. Bellato, K. Bengoetxea, Y. Berzak, I. A. Bhat, R. A. Bhat, E. Biagetti, E. Bick, R. Blokland, V. Bobicev, C. Börstell, C. Bosco, G. Bouma, S. Bowman, A. Boyd, A. Burchardt, M. Candito, B. Caron, G. Caron, G. Cebiroğlu Eryiğit, F. M. Cecchini, G. G. A. Celano, S. Čéplö, S. Cetin, F. Chalub, J. Choi, Y. Cho, J. Chun, S. Cinková, A. Collomb, Ç. Çöltekin, M. Connor, M. Courtin, E. Davidson, M.-C. de Marneffe, V. de Paiva, A. Diaz de Ilarraza, C. Dickerson, P. Dirix, K. Dobrovoljc, T. Dozat, K. Droganova, P. Dwivedi, M. Eli, A. Elkahky, B. Ephrem, T. Erjavec, A. Etienne, R. Farkas, H. Fernandez Alcalde, J. Foster, C. Freitas, K. Gajdošová, D. Galbraith, M. Garcia, M. Gärdenfors, S. Garza, K. Gerdes, F. Ginter, I. Goenaga, K. Gojenola, M. Gökırmak, Y. Goldberg, X. Gómez Guinovart, B. Gonzáles Saavedra, M. Grioni, N. Grūzītis, B. Guillaume, C. Guillot-Barbance, N. Habash, J. Hajič, J. Hajič jr., L. Hà Mỹ, N.-R. Han, K. Harris, D. Haug, B. Hladká, J. Hlaváčová, F. Hociung, P. Hohle, J. Hwang, R. Ion, E. Irimia, Ọ. Ishola, T. Jelínek, A. Johannsen, F. Jørgensen, H. Kaşıkara, S. Kahane, H. Kanayama, J. Kanerva, B. Katz, T. Kayadelen, J. Kenney, V. Kettnerová, J. Kirchner, K. Kopacewicz, N. Kotsyba, S. Krek, S. Kwak, V. Laippala, L. Lambertino, L. Lam, T. Lando, S. D. Larasati, A. Lavrentiev, J. Lee, P. Lê H`ông, A. Lenci, S. Lertpradit, H. Leung, C. Y. Li, J. Li, K. Li, K. Lim, N. Ljubešić, O. Loginova, O. Lyashevskaya, T. Lynn, V. Macketanz, A. Makazhanov, M. Mandl, C. Manning, R. Manurung, C. Mărănduc, D. Mareček, K. Marheinecke, H. Martínez Alonso, A. Martins, J. Mašek, Y. Matsumoto, R. McDonald, G. Mendonça, N. Miekka, M. Misirpashayeva, A. Missilä, C. Mititelu, Y. Miyao, S. Montemagni, A. More, L. Moreno Romero, K. S. Mori, S. Mori, B. Mortensen, B. Moskalevskyi, K. Muischnek, Y. Murawaki, K. Müürisep, P. Nainwani, J. I. Navarro Horñiacek, A. Nedoluzhko, G. Nešpore-Bērzkalne, L. Nguy˜ên Thị, H. Nguy˜ên Thị Minh, V. Nikolaev, R. Nitisaroj, H. Nurmi, S. Ojala, A. Olúòkun, M. Omura, P. Osenova, R. Östling, L. Øvrelid, N. Partanen, E. Pascual, M. Passarotti, A. Patejuk, G. Paulino-Passos, S. Peng, C.-A. Perez, G. Perrier, S. Petrov, J. Piitulainen, E. Pitler, B. Plank, T. Poibeau, M. Popel, L. Pretkalniņa, S. Prévost, P. Prokopidis, A. Przepiórkowski, T. Puolakainen, S. Pyysalo, A. Rääbis, A. Rademaker, L. Ramasamy, T. Rama, C. Ramisch, V. Ravishankar, L. Real, S. Reddy, G. Rehm, M. Rießler, L. Rinaldi, L. Rituma, L. Rocha, M. Romanenko, R. Rosa, D. Rovati, V. Roșca, O. Rudina, J. Rueter, S. Sadde, B. Sagot, S. Saleh, T. Samardžić, S. Samson, M. Sanguinetti, B. Saulīte, Y. Sawanakunanon, N. Schneider, S. Schuster, D. Seddah, W. Seeker, M. Seraji, M. Shen, A. Shimada, M. Shohibussirri, D. Sichinava, N. Silveira, M. Simi, R. Simionescu, K. Simkó, M. Šimková, K. Simov, A. Smith, I. Soares-Bastos, C. Spadine, A. Stella, M. Straka, J. Strnadová, A. Suhr, U. Sulubacak, Z. Szántó, D. Taji, Y. Takahashi, T. Tanaka, I. Tellier, T. Trosterud, A. Trukhina, R. Tsarfaty, F. Tyers, S. Uematsu, Z. Urešová, L. Uria, H. Uszkoreit, S. Vajjala, D. van Niekerk, G. van Noord, V. Varga, E. Villemonte de la Clergerie, V. Vincze, L. Wallin, J. X. Wang, J. N. Washington, S. Williams, M. Wirén, T. Woldemariam, T.-s. Wong, C. Yan, M. M. Yavrumyan, Z. Yu, Z. Žabokrtský, A. Zeldes, D. Zeman, M. Zhang, and H. Zhu. Universal dependencies 2.3, 2018. URL `http://hdl.handle.net/11234/1-2895`. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

J. Nivre, M.-C. de Marneffe, F. Ginter, J. Hajič, C. D. Manning, S. Pyysalo, S. Schuster, F. Tyers, and D. Zeman. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC)*, 2020.

C. Samuelsson. A statistical theory of dependency syntax. In *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*, 2000.

F. Sangati and C. Mazza. An English dependency treebank à la tesnière. In *Proceedings of the 8th International Workshop on Treebanks and Linguistic Theories*, pages 173–184, Nov. 2009.

P. Sgall, E. Hajičová, and J. Panevová. *The Meaning of the Sentence in Its Pragmatic Aspects*. Reidel, 1986.

T. Shi, L. Huang, and L. Lee. Fast(er) exact decoding and global training for transition-based dependency parsing via a minimal feature set. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 12–23, 2017.

A. Smith, B. Bohnet, M. de Lhoneux, J. Nivre, Y. Shao, and S. Stymne. 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 113–123, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics.

P. Stenetorp. Transition-based dependency parsing using recursive neural networks. In *NIPS Workshop on Deep Learning*, 2013.

L. Tesnière. *Éléments de syntaxe structurale*. Editions Klincksieck, 1959.

S. Tongchim, V. Sornlertlamvanich, and H. Isahara. Experiments in base-NP chunking and its role in dependency parsing for Thai. In *Coling 2008: Companion volume: Posters*, pages 123–126, Manchester, UK, Aug. 2008. Coling 2008 Organizing Committee.

H. Yamada and Y. Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 195–206, Nancy, France, Apr. 2003.

D. Zeman, J. Hajič, M. Popel, M. Potthtyersast, M. Straka, F. Ginter, J. Nivre, and S. Petrov. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 2018.

Searching for Search Errors in Neural Morphological Inflection

Martina Forster[1] **Clara Meister**[1] **Ryan Cotterell**[1,2]
[1]ETH Zürich [2]University of Cambridge
martfors@ethz.ch {clara.meister, ryan.cotterell}@inf.ethz.ch

Abstract

Neural sequence-to-sequence models are currently the predominant choice for language generation tasks. Yet, on word-level tasks, exact inference of these models reveals the empty string is often the global optimum. Prior works have speculated this phenomenon is a result of the inadequacy of neural models for language generation. However, in the case of morphological inflection, we find that the empty string is almost never the most probable solution under the model. Further, greedy search often finds the global optimum. These observations suggest that the poor calibration of many neural models may stem from characteristics of a specific subset of tasks rather than general ill-suitedness of such models for language generation.

1 Introduction

Neural sequence-to-sequence models are omnipresent in the field of natural language processing due to their impressive performance. They hold state of the art on a myriad of tasks, e.g., neural machine translation (NMT; Ott et al., 2018b) and abstractive summarization (AS; Lewis et al., 2019). Yet, an undesirable property of these models has been repeatedly observed in word-level tasks: When using beam search as the decoding strategy, increasing the beam width beyond a size of $k = 5$ often leads to a drop in the quality of solutions (Murray and Chiang, 2018; Yang et al., 2018; Cohen and Beck, 2019). Further, in the context of NMT, it has been shown that the empty string is frequently the most-probable solution under the model (Stahlberg and Byrne, 2019). Some suggest this is a manifestation of the general inadequacy of neural models for language generation tasks (Koehn and Knowles, 2017; Kumar and Sarawagi, 2019; Holtzman et al., 2020; Stahlberg, 2020); in this work, we find evidence demonstrating otherwise.

	$k = 1$	$k = 10$	$k = 100$	$k = 500$
NMT	63.1%	46.1%	44.3%	6.4%
MI	0.8%	0.0%	0.0%	0.0%

Table 1: Percentage of search errors—which we define as instances where the search strategy does not find the global optimum under the model—for Transformers trained on IWSLT'14 De-En (NMT) and SIGMORPHON 2020 (Morphological Inflection; MI) when decoding with beam search for varying beam widths (k). MI results are averaged across languages.

Sequence-to-sequence transducers for character-level tasks often follow the architectures of their word-level counterparts (Faruqui et al., 2016; Lee et al., 2017), and have likewise achieved state-of-the-art performance on e.g., morphological inflection generation (Wu et al., 2020) and grapheme-to-phoneme conversion (Yolchuyeva et al., 2019). Given prior findings, we might expect to see the same degenerate behavior in these models—however, we do not. We run a series of experiments on morphological inflection (MI) generators to explore whether neural transducers for this task are similarly poorly calibrated, i.e. are far from the true distribution $p(\mathbf{y} \mid \mathbf{x})$. We evaluate the performance of two character-level sequence-to-sequence transducers using different decoding strategies; our results, previewed in Tab. 1, show that evaluation metrics do not degrade with larger beam sizes as in NMT or AS. Additionally, only in extreme circumstances, e.g., low-resource settings with less than 100 training samples, is the empty string ever the global optimum under the model.

Our findings directly refute the claim that neural architectures are inherently inadequate for modeling language generation tasks. Instead, our results admit two potential causes of the degenerate behavior observed in tasks such as NMT and AS: (1) lack of a deterministic mapping between

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1388–1394
April 19 - 23, 2021.

input and output and (2) a (perhaps irreparable) discrepancy between sample complexity and training resources. Our results alone are not sufficient to accept or reject either hypothesis, and thus we leave these as future research directions.

2 Neural Transducers

Sequence-to-sequence transduction is the transformation of an input sequence into an output sequence. Tasks involving this type of transformation are often framed probabilistically, i.e., we model the *probability* of mapping one sequence to another. On many tasks of this nature, neural sequence-to-sequence models (Sutskever et al., 2014; Bahdanau et al., 2015) hold state of the art.

Formally, a neural sequence-to-sequence model defines a probability distribution $p_{\boldsymbol{\theta}}(\mathbf{y} \,|\, \mathbf{x})$ parameterized by a neural network with a set of learned weights $\boldsymbol{\theta}$ for an input sequence $\mathbf{x} = \langle x_1, x_2, \dots \rangle$ and output sequence $\mathbf{y} = \langle y_1, y_2, \dots \rangle$. Morphological inflection and NMT are two such tasks, wherein our outputs are both strings. Neural sequence-to-sequence models are typically locally normalized, i.e. $p_{\boldsymbol{\theta}}$ factorizes as follows:

$$p_{\boldsymbol{\theta}}(\mathbf{y} \,|\, \mathbf{x}) = \prod_{t=1}^{|\mathbf{y}|} p_{\boldsymbol{\theta}}(y_t \,|\, \mathbf{x}, \mathbf{y}_{<t}) \tag{1}$$

Given a vocabulary $\mathcal{V}$, each conditional $p_{\boldsymbol{\theta}}$ is a distribution over $\mathcal{V} \cup \{\text{EOS}\}$ and $y_0 := \text{BOS}$. We consider $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})$ to be **well-calibrated** if its probability estimates are representative of the true likelihood that a solution $\mathbf{y}$ is correct.

Morphological Inflection. In the task of morphological inflection, $\mathbf{x}$ is an encoding of the lemma concatenated with a flattened morphosyntactic description (MSD) and $\mathbf{y}$ is the target inflection. As a concrete example, consider inflecting the German word *Bruder* into the genitive plural, as shown in Tab. 2. Then, $\mathbf{x}$ is the string $\langle$`B r u d e r GEN PL`$\rangle$ and $\mathbf{y}$ is the string $\langle$`B r ü d e r`$\rangle$. As this demonstrates, morphological inflection generation is, by its nature, modeled at the character level (Faruqui et al., 2016; Wu and Cotterell, 2019), i.e., our target vocabulary $\mathcal{V}$ is a set of characters in the language. Note that $\mathbf{y} \in \mathcal{V}^*$, but $\mathbf{x} \notin \mathcal{V}^*$ due to the additional encoding of the MSD. This stands in contrast to NMT, which is typically performed on a (sub)word level, making the vocabulary size orders of magnitude larger.

	Singular	Plural
Nominativ	Bruder	Brüder
Genitiv	Bruders	Brüder
Dativ	Bruder	Brüdern
Akkusativ	Bruder	Brüder

Table 2: Inflection table for the German word *Bruder*

Another important differentiating factor of morphological inflection generation in comparison to many other generation tasks in NLP is the one-to-one mapping between source and target.[1] In contrast, there are almost always many correct ways to translate a sentence into another language or to summarize a large piece of text; this characteristic manifests itself in training data where a single phrase has instances of different mappings, making tasks such as translation and summarization inherently ambiguous.

3 Decoding

In the case of probabilistic models, the decoding problem is the search for the most-probable sequence among valid sequences $\mathcal{V}^*$ under the model $p_{\boldsymbol{\theta}}$:

$$\mathbf{y}^\star = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{V}^*} \log p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}) \tag{2}$$

This problem is also known as maximum-a-posteriori (MAP) inference. Decoding is often performed with a heuristic search method such as greedy or beam search (Reddy, 1977), since performing exact search can be computationally expensive, if not impossible.[2] While for a deterministic task, greedy search is optimal under a Bayes optimal model,[3] most text generation tasks benefit from using beam search. However, text quality almost invariably decreases for beam sizes larger than $k = 5$. This phenomenon is sometimes referred to as the **beam search curse**, and has been investigated in detail by a number of scholarly works (Koehn and Knowles, 2017; Murray and Chiang, 2018; Yang et al., 2018; Stahlberg and Byrne, 2019; Cohen and Beck, 2019; Eikema and Aziz, 2020).

---

[1] While there are cases where there exist multiple inflected forms of a lemma, e.g., in English the past tense of *dream* can be realized as either *dreamed* or *dreamt*, these cases (termed “overabundance”) are rare (Thornton, 2019).

[2] The search space is exponential in the sequence length and due to the non-Markov nature of (typical) neural transducers, dynamic-programming techniques are not helpful.

[3] Under such a model, the correct token y_i at time step i will be assigned all probability mass.

	Transformer				HMM			
	$k = 1$	$k = 10$	$k = 100$	Dijkstra	$k = 1$	$k = 10$	$k = 100$	Dijkstra
Overall	90.34%	90.37%	90.37%	90.37%	86.03%	85.62%	85.60%	85.60%
Low-resource	84.10%	84.12%	84.12%	84.12%	70.99%	69.37%	69.31%	69.31%
High-resource	94.05%	94.08%	94.08%	94.08%	93.60%	93.72%	93.72%	93.72%

Table 3: Prediction accuracy (averaged across languages) by decoding strategy for Transformer and HMM. We include breakdown for low-resource and high-resource trained models. k indicates beam width.

Exact decoding can be seen as the case of beam search where the beam size is effectively stretched to infinity.[4] By considering the complete search space, it finds the globally best solution under the model $p_{\boldsymbol{\theta}}$. While, as previously mentioned, exact search can be computationally expensive, we can employ efficient search strategies due to some properties of $p_{\boldsymbol{\theta}}$. Specifically, from Eq. (1), we can see that the scoring function for sequences $\mathbf{y}$ is monotonically decreasing in t. We can therefore find the provably optimal solution with Dijkstra's algorithm (Dijkstra, 1959), which terminates and returns the global optimum the first time it encounters an EOS. Additionally, to prevent a large memory footprint, we can lower-bound the search using any complete hypothesis, e.g., the empty string or a solution found by beam search (Stahlberg and Byrne, 2019; Meister et al., 2020). That is, we can prematurely stop exploring solutions whose scores become less than these hypotheses at any point in time. Although exact search is an exponential-time method in this setting, we see that, in practice, it terminates quickly due to the peakiness of $p_{\boldsymbol{\theta}}$ (see App. A). While the effects of exact decoding and beam search decoding with large beam widths have been explored for a number of word-level tasks (Stahlberg and Byrne, 2019; Cohen and Beck, 2019; Eikema and Aziz, 2020), to the best of our knowledge, they have not yet been explored for any character-level sequence-to-sequence tasks.

4 Experiments

We run a series of experiments using different decoding strategies to generate predictions from morphological inflection generators. We report results for two near-state-of-the-art models: a multilingual Transformer (Wu et al., 2020) and a (neuralized) hidden Markov model (HMM; Wu and Cotterell, 2019). For reproducibility, we mimic their proposed architectures and exactly follow their data pre-processing steps, training strategies and hyper-parameter settings.[5]

	Beam $k{=}1$	**Beam** $k{=}10$	**Optimum**	**Empty String**
Transformer	-0.619	-0.617	-0.617	-6.56
HMM	-1.08	-0.89	-0.80	-20.15

Table 4: Average log probability of inflections generated with various decoding strategies and the empty string (averaged across all languages).

Data. We use the data provided by the SIGMORPHON 2020 shared task (Vylomova et al., 2020), which features lemmas, inflections, and corresponding MSDs in the UniMorph schema (Kirov et al., 2018) in 90 languages in total. The set of languages is typologically diverse (spanning 18 language families) and contains both high- and low-resource examples, providing a spectrum over which we can evaluate model performance. The full dataset statistics can be found on the task homepage.[6] When reporting results, we consider languages with < 1000 and ≥ 10000 training samples as low- and high-resource, respectively.

Decoding Strategies. We decode morphological inflection generators using exact search and beam search for a range of beam widths. We use the SGNMT library for decoding (Stahlberg et al., 2017) albeit adding Dijkstra's algorithm.

4.1 Results

Tab. 3 shows that the accuracy of predictions from neural MI generators generally does not decrease when larger beam sizes are used for decoding; this observation holds for both model architectures. While it may be expected that

---

[4] This interpretation is useful when comparing with beam search with increasing beam widths.

[5] `https://github.com/shijie-wu/neural-transducer/tree/sharedtasks`

[6] `https://sigmorphon.github.io/sharedtasks/2020/task0/`

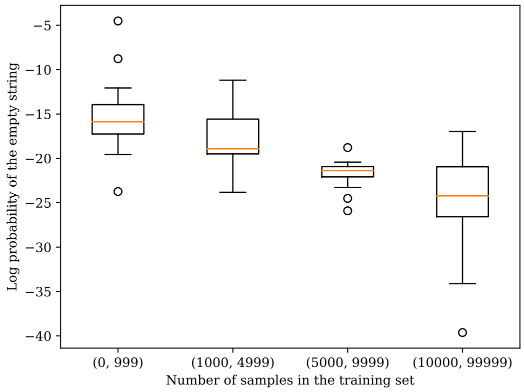


Figure 1: Average (log) probability of the empty string for different training dataset sizes for HMM.

models for low-resource languages generally perform worse than those for high-resource ones, this disparity is only prominent for HMMs, where the difference between high- and low-resource accuracy is $\approx 24\%$ vs. $\approx 10\%$ for the Transformers. Notably, for the HMM, the global optimum under the model is the empty string far more often for low-resource languages than it is for high-resource ones (see Tab. 5). We can explicitly see the inverse relationship between the log-probability of the empty string and resource size in Fig. 1. In general, across models for all 90 languages, the global optimum is rarely the empty string (Tab. 5). Indeed, under the Transformer-based transducer, the empty string was *never* the global optimum. This is in contrast to the findings of Stahlberg and Byrne (2019), who found for word-level NMT that the empty string was the optimal translation in more than 50% of cases, even under state-of-the-art models. Rather, the average log-probabilities of the empty string (which is quite low) and the chosen inflection lie far apart (Tab. 4).

5 Discussion

Our findings admit two potential hypotheses for poor calibration of neural models in certain language generation tasks, a phenomenon we do not observe in morphological inflection. First, the tasks in which we observe this property are ones that lack a deterministic mapping, i.e. tasks for which there may be more than one correct solution for any given input. As a consequence, probability mass may be spread over an arbitrarily large number of hypotheses (Ott et al., 2018a; Eikema and Aziz, 2020). In contrast, the task of morphological inflection has a near-deterministic mapping. We observe this empirically in Tab. 4, which shows that the probability of the global optimum on average covers most of the available probability mass—a phenomenon also observed by Peters and Martins (2019). Further, as shown in Tab. 6, the dearth of search errors even when using greedy search suggests there are rarely competing solutions under the model. We posit it is the lack of ambiguity in morphological inflection that allows for the well-calibrated models we observe.

	HMM	**Transformer**
Overall	2.03%	0%
Low-resource	8.65%	0%
High-resource	0.0002%	0%

Table 5: Average percentage of empty strings when decoding with exact inference for HMM and Transformer, with resource group breakdown.

	$k = 1$	$k = 10$	$k = 100$	$k = 200$
HMM	6.20%	2.33%	0.001%	0.0%
Transformer	0.68%	0.0%	0.0%	0.0%

Table 6: Average percentage of search errors (averaged across languages) for beam search with beam width k.

Second, our experiments contrasting high- and low-resource settings indicate insufficient training data may be the main cause of the poor calibration in sequence-to-sequence models for language generation tasks. We observe that models for MI trained on fewer data typically place more probability mass on the empty string. As an extreme example, we consider the case of the Zarma language, whose training set consists of only 56 samples. Under the HMM, the average log-probability of the generated inflection and empty string are very close (-8.58 and -8.77, respectively). Furthermore, on the test set, the global optimum of the HMM model for Zarma is the empty string 81.25% of the time.

From this example, we can conjecture that lack of sufficient training data may manifest itself as the (relatively) high probability of the empty string or the (relatively) low probability of the optimum. We can extrapolate to models for NMT and other word-level tasks, for which we frequently see the above phenomenon. Specifically, our experiments suggest that when neural language generators frequently place high probability on the empty string, there may be a discrepancy between the available training resources and the number of samples needed to successfully learn the target function. While this at first seems an easy problem to fix, we expect the number of resources needed in tasks such as NMT and AS is much larger than that for MI if not due to the size of the output space alone; perhaps so large that they are essentially unattainable. Under this explanation, for certain tasks, there may not be a straightforward fix to the degenerate behavior observed in some neural language generators.

6 Conclusion

In this work, we investigate whether the poor calibration often seen in sequence-to-sequence models for word-level tasks also occurs in models for morphological inflection. We find that character-level models for morphological inflection are generally well-calibrated, i.e. the probability of the globally best solution is almost invariably much higher than that of the empty string. This suggests the degenerate behavior observed in neural models for certain word-level tasks is not due to the inherent incompatibility of neural models for language generation. Rather, we find evidence that poor calibration may be linked to specific characteristics of a subset of these task, and suggest directions for future exploration of this phenomenon.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations*.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 1290–1299, Long Beach, California, USA.

Edsger W. Dijkstra. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? The inadequacy of the mode in neural machine translation. *CoRR*, abs/12005.10283.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 634–643, San Diego, California. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *International Conference on Learning Representations*.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Aviral Kumar and Sunita Sarawagi. 2019. Calibration of encoder decoder models for neural machine translation. *CoRR*, abs/1903.00802.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8(0).

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018a. Analyzing uncertainty in neural machine translation. In *International Conference on Machine Learning*, pages 3956–3965.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018b. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Ben Peters and André F. T. Martins. 2019. IT–IST at the SIGMORPHON 2019 shared task: Sparse two-headed models for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 50–56, Florence, Italy. Association for Computational Linguistics.

Raj Reddy. 1977. Speech understanding systems: A summary of results of the five-year research effort at carnegie mellon university.

Felix Stahlberg. 2020. *The Roles of Language Models and Hierarchical Models in Neural Sequence-to-Sequence Prediction*. Ph.D. thesis, University of Cambridge.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Felix Stahlberg, Eva Hasler, Danielle Saunders, and Bill Byrne. 2017. SGNMT – a flexible NMT decoding platform for quick prototyping of new models and search strategies. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 25–30, Copenhagen, Denmark. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Anna M. Thornton. 2019. Overabundance in morphology. *Oxford Research Encyclopedia of Linguistics*.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Ponti, Rowan Hall Maudslay, Ran Zmigrod, Joseph Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrej Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. The SIGMORPHON 2020 Shared Task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. *CoRR*, abs/2005.10213.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-)scoring methods and stopping criteria for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3054–3059, Brussels, Belgium. Association for Computational Linguistics.

Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth. 2019. Transformer based grapheme-to-phoneme conversion. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association*, pages 2095–2099. ISCA.

A Timing

	Transformer		**HMM**	
	$k = 1$	Dijkstra	$k = 1$	Dijkstra
Overall	0.082	0.091	0.016	0.027
Low-resource	0.072	0.082	0.013	0.032
High-resource	0.075	0.083	0.017	0.026

Table 7: Average time (s) for inflection generation by decoding strategy. Breakdown by resource group is included.

Quantifying Appropriateness of Summarization Data for Curriculum Learning

Ryuji Kano†‡ **Takumi Takahashi**† **Toru Nishino**†
Motoki Taniguchi† **Tomoki Taniguchi**† **Tomoko Ohkuma**†
†Fuji Xerox Co., Ltd.
‡Institute of Innovative Research, Tokyo Institute of Technology
{kano.ryuji, takahashi.takumi, nishino.toru,
motoki.taniguchi, taniguchi.tomoki, ohkuma.tomoko}
@fujixerox.co.jp

Abstract

Much research has reported the training data of summarization models are noisy; summaries often do not reflect what is written in the source texts. We propose an effective method of curriculum learning to train summarization models from such noisy data. Curriculum learning is used to train sequence-to-sequence models with noisy data. In translation tasks, previous research quantified noise of the training data using two models trained with noisy and clean corpora. Because such corpora do not exist in summarization fields, we propose a model that can quantify noise from a single noisy corpus. We conduct experiments on three summarization models; one pretrained model and two non-pretrained models, and verify our method improves the performance. Furthermore, we analyze how different curricula affect the performance of pretrained and non-pretrained summarization models. Our result on human evaluation also shows our method improves the performance of summarization models.

1 Introduction

Sequence-to-sequence models have led to the great advancement of summarization. These models require appropriate pairs of source texts and summaries. However, much research has reported summarization datasets contain inappropriate pairs (Zhang and Tetreault, 2019; Li et al., 2019; Kryscinski et al., 2019; Matsumaru et al., 2020). Sequence-to-sequence summarization models leverage titles as summaries. In theory, summaries should reflect what is written in the source texts, but in fact, the titles can be too general or contain information not written in the source texts. There is a growing need to deal with these noisy datasets.

One way to train with noisy data is curriculum learning (Bengio et al., 2009). Curriculum learning is a method to change the order of training data and improves convergence speed and the performance of models. In translation tasks, previous studies estimate noise of data using likelihoods of two generative models trained with clean and noisy data, and then applied it to curriculum learning (Wang et al., 2018, 2019; Kumar et al., 2019).

Because there is no such datasets in the summarization field, we propose *Appropriateness Estimator*, a noise-estimating model that can be trained from a single noisy corpus. The model distinguishes pairs of a source and target text in the original summarization dataset from randomly assigned pairs. The randomly assigned pairs are clearly inappropriate pairs; the target texts do not reflect the information on the source texts. By distinguishing the obvious inappropriate pairs, the model learns to predict *appropriateness* of data. We apply the appropriateness to curriculum learning; when training a summarization model, we gradually change the training data from inappropriate data to appropriate ones.

We experiment with two datasets; Enron subject dataset (Zhang and Tetreault, 2019), and Reddit TIFU title dataset (Kim et al., 2019). Both have noisy training data, but the Enron dataset has manually cleansed validation and evaluation datasets, whereas the validation and the evaluation datasets of the Reddit dataset are raw datasets that include noise.

As summarization models, we employ BART as a pretrained model, and Transformer and sequence-to-sequence with attention (Seq2seqAtt) as non-pretrained models. The result shows our Appropriateness Estimator improves both pretrained models and non-pretrained models.

Also, we analyze how three different curricula affect the result and conclude training with small fine data in the last phase is important for pretrained models and generalization with various data in the beginning phase is important for non-pretrained

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1395–1405
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

models. Also, we conduct human evaluation and verify curriculum learning using our Appropriateness Estimator improves the performance of summarization models. The contributions of this paper are as follows.

- We propose *Appropriateness Estimator* that estimates *appropriateness* of source and target texts and that can be trained from a single noisy corpus.
- We conduct experiments on three summarization models: one pretrained model and two non-pretrained models, and verify our method improves the performance of the models.
- We analyze how three different curricula affect the performance of pretrained and non-pretrained summarization models.

2 Related Works

Curriculum learning is a method to change the order of training data to improve convergence speed and accuracy (Bengio et al., 2009). Cirik et al. (2016) applied this to language generation, and introduced two types of curriculum learning: Baby step curriculum and One-Pass curriculum, and concluded the former is more effective to language generation. Many of the later works applied Baby step curriculum to translation tasks (Wang et al., 2019; Zhou et al., 2020), but research of curriculum learning on summarization is yet to be conducted.

Curriculum learning was originally a method to sort training data by difficulty, but recent research proposed methods to sort data by noise. Wang et al. (2018) proposed a method to quantify the noise in data using two models; one trained on clean data and the other on noisy data. Using the same algorithm, Kumar et al. (2019) applied reinforcement learning to choose which subset is most appropriate for training. However, it is not possible to apply this to summarization tasks, because clean and noisy versions of the same corpus are not available.

Sequence-to-sequence summarization models generally use headlines, titles or subjects as summaries. However, it is reported that those datasets are noisy. Zhang and Tetreault (2019) introduced a task to generate subjects of emails, but because the original subjects were noisy, they prepared new validation and evaluation datasets on their own. Li et al. (2019) used rules and a classification model to filter noisy data of review summarization.

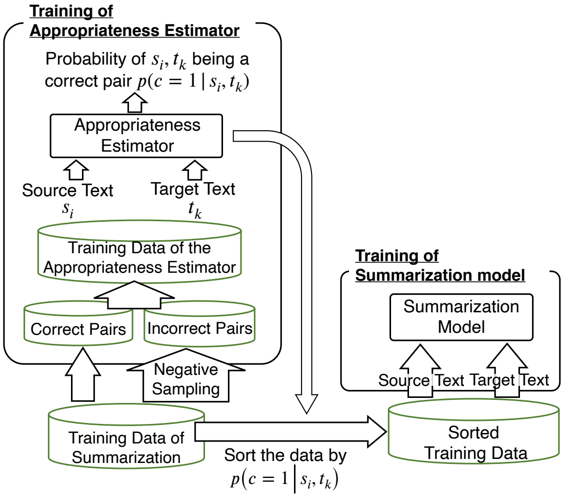


Figure 1: Description of the Appropriateness Estimator.

3 Method

Appropriateness Estimator We propose Appropriateness Estimator, a noise estimator model that can be trained from a single noisy corpus. The overview of the model is described in Figure 1. We label pairs of source and target texts in summarization training data as positive. We assign randomly sampled target texts to source texts and label the pairs as negative. The training task of the model is to predict the labels of the pairs. Pairs in summarization training data are all labeled positive, but as explained in Introduction, it includes inappropriate pairs. Following Li et al. (2020), we conduct early stopping to prevent the model from overfitting to noisy data.

The probability $p(c|s_i, t_k)$ of the model indicates appropriateness of pairs. Here s_i is a source text, and t_k is a target text. c is a binary class; $c = 1$ when the label of a pair is positive, and $c = 0$ otherwise. We sort summarization training data by the appropriateness and conduct curriculum learning.

Curriculum Learning Cirik et al. (2016) introduced two curricula: One-Pass curriculum and Baby step curriculum. The overview of these curricula is described in Figure 2. In both settings, we first sort data by a chosen metric (e.g. appropriateness or target length) in ascending order. Next, we split the data into segments.

One-Pass curriculum starts training from an easiest or noisiest segment and when the model converges, the training data shifts to a next segment. Baby step curriculum gradually increases the amount of training data starting from an easiest or noisiest segment. These two curricula both start training from small amount of data, so there is a risk of overfitting. To overcome this, we propose Noise-Annealing curriculum; we first train a model with all data, and gradually decrease the amount of the training data.

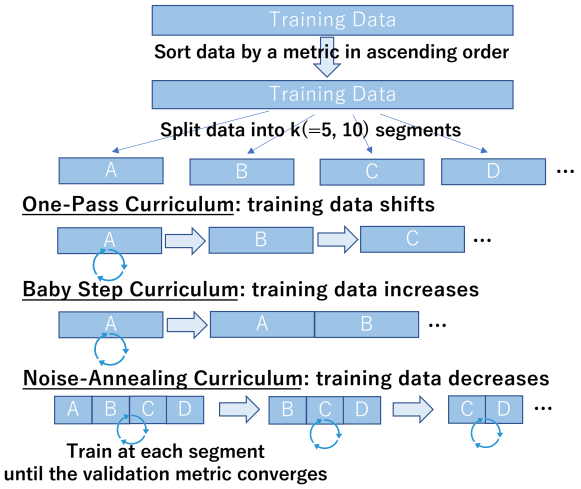


Figure 2: Description of Curriculum Learning.

4 Experiment

Enron Subject Dataset The Enron dataset (Klimt and Yang, 2004) is a collection of email messages of employees in the Enron Corporation. Zhang and Tetreault (2019) organized this data for a subject generation task. However, the original dataset was not clean enough to use for evaluation. Thus, they manually annotated appropriate subjects for validation and evaluation. For training, we have 14,436 subject-email pairs. We have 1,906 and 1,960 data as a validation and an evaluation dataset.

Reddit TIFU Dataset The Reddit TIFU Dataset (Kim et al., 2019) is a dataset of a social media forum, Reddit. TIFU stands for “today i f*** up”; the posts are about the experiences of failure. Here, we use titles of each post as summaries and leverage them for a summarization task. For training, validation, and evaluation datasets, we have 71113, 3951 and 3951 data.

Appropriateness Estimator We employ Decomposable Attention (Parikh et al., 2016) as Appropriateness Estimator. We use GloVe[1] as the initial parameters of word embeddings. The dimensions of the word embeddings and the hidden layers are 300 and 200. The training epoch is 20. We also tried BERT (Devlin et al., 2019); although the result of the classification task was better, the performance was worse when the BERT model was applied to curriculum learning of summarization models. This might be because huge neural network models like BERT can memorize all training data including noise (Zhang et al., 2017). By contrast, smaller models can be robust to noise.

As explained in Introduction and in Section 3, we label randomly assigned pairs of source texts and target texts as negative and actual pairs in summarization datasets as positive. The number of negative pairs is same as the positive pairs. Therefore, the numbers of training, and validation data of Appropriateness Estimator are twice the size of the training/validation data of summarization. We validate with F1 scores and use the model with the highest validation score for curriculum learning. The best validation F1 scores of the models were 0.94 on the Enron dataset, and 0.92 on the Reddit dataset.

Summarization Model We experiment with three summarization models: one pretrained model BART (Lewis et al., 2020), and two non-pretrained models, Transformer (Vaswani et al., 2017) and sequence-to-sequence with attention (Seq2seqAtt) (Rush et al., 2015). The hyperparameters of the models are described in Appendix A.

Changing random seeds, we conduct the same experiments 5 times and use the average values as the result. We evaluate with ROUGE F1 scores (Lin, 2004). We validate at every epoch of each segment. As the validation metric we use F1 score of ROUGE-1 (ROUGE-1-F).

Curriculum Learning We experiment in four settings; three types of curriculum described in Section 3 and one without curriculum learning. As the number of segments, we conduct experiments on 5 and 10, and adopt better result. The order of the training data in each segment is shuffled.

Metrics to Sort Data We experiment with two metrics to sort data: appropriateness, and target length. Target length is a metric generally used in curriculum learning (Cirik et al., 2016; Platanios et al., 2019; Wang et al., 2019; Zhou et al., 2020).

5 Result and Discussion

The result on Table 1 shows curriculum learning improves the performance of summarization models. Curriculum learning with appropriateness per-

---

[1] https://nlp.stanford.edu/projects/glove/

			Reddit Title			Enron Subject		
Model	Curriculum	Sort by	R-1-F	R-2-F	R-L-F	R-1-F	R-2-F	R-L-F
BART (pretrained)	No Curriculum	-	0.254	0.124	0.222	0.301	0.153	0.255
	Noise-Annealing	Appropriateness	0.271	0.132	0.239	0.315	0.167	0.270
		Target Length	**0.277**	0.135	**0.245**	0.312	0.171	0.271
	Baby step	Appropriateness	0.230	0.108	0.200	0.277	0.136	0.236
		Target Length	0.244	0.117	0.214	0.300	0.156	0.257
	One-Pass	Appropriateness	0.276	**0.137**	0.243	**0.339**	**0.193**	**0.294**
		Target Length	0.268	0.123	0.235	0.329	0.186	0.286
Transformer (non-pretrained)	No Curriculum	-	0.184	0.047	0.140	0.093	0.019	0.044
	Noise-Annealing	Appropriateness	**0.192**	**0.051**	**0.146**	**0.106**	**0.022**	0.047
		Target Length	0.188	0.048	0.141	0.094	0.019	0.044
	Baby step	Appropriateness	0.170	0.027	0.131	0.079	0.012	0.056
		Target Length	0.167	0.023	0.125	0.091	0.018	**0.065**
	One-Pass	Appropriateness	0.153	0.017	0.121	0.040	0.003	0.027
		Target Length	0.156	0.014	0.131	0.062	0.001	0.040
Seq2seqAtt (non-pretrained)	No Curriculum	-	0.171	0.041	0.118	0.051	0.006	0.031
	Noise-Annealing	Appropriateness	**0.176**	0.041	0.116	**0.060**	**0.008**	**0.040**
		Target Length	0.172	**0.043**	0.118	0.057	**0.008**	0.036
	Baby step	Appropriateness	0.167	0.027	0.112	0.051	0.006	0.029
		Target Length	0.147	0.030	0.108	0.051	0.006	0.030
	One-Pass	Appropriateness	0.161	0.018	**0.119**	0.039	0.000	0.015
		Target Length	0.142	0.021	0.099	0.034	0.000	0.016

Table 1: Result on curriculum learning. Appropriateness indicates probabilities computed by Appropriateness Estimator. R-1-F, R-2-F, and R-L-F are F1 scores of ROUGE-1, ROUGE-2, and ROUGE-L.

form better than that with target length more on the Enron dataset, which has clean validation and evaluation datasets.

Difference among Curricula One-Pass and Noise-Annealing curricula improved the BART models whereas Baby step curriculum led to the worst result. Conversely, for the non-pretrained models, only Noise-Annealing curriculum improved the performance. One-Pass and Noise-Annealing curricula does fine-tuning with smaller data in the last phase of training, and only Noise-Annealing curriculum does generalization with various data in the beginning phase. It is possible that BART is a pretrained model and does not need to be generalized. Rather, fine-tuning is more important. By contrast, non-pretrained models need generalization.

Characteristic of Appropriateness Appropriateness Estimator improved the summarization models, but it is unclear what the appropriateness represents. Table 2 shows Pearson's correlation coefficients between the appropriateness and source/target length. The coefficients are less than 0.2. This indicates the appropriateness represents a different aspect of texts from length of texts. The target texts of low appropriateness data contain information not written in the source texts. We further discuss this topic on Appendix C.

Dataset	Target length	Source length
Enron	0.151	0.079
Reddit	0.156	0.018

Table 2: The correlation coefficients between the appropriateness and source/target length.

Human Evaluation We conduct human evaluation on two BART models: one trained with Noise-Annealing curriculum and appropriateness, and the other trained without curriculum learning. We omit data if two summaries generated by the models are same, and get 90 pairs of generated summaries for each dataset. Annotators choose which summaries are better in terms of informativeness and fluency. Here, the informativeness indicates how well the generated summaries reflect important topics of the source texts, and the fluency represents naturalness of the generated summaries in terms of grammar. The result is shown on Table 3. The result shows the model trained with our method achieves better performance both in terms of informativeness and fluency. To validate the statistical significance of the result, we aggregate the number of votes on "better" and "slightly better" and conduct chi-square test. The statistical significance is also shown on Table 3.

6 Conclusion

In this research, we proposed Appropriateness Estimator that quantifies noise of training data for

	Informativeness		Fluency	
	Enr†	Red†	Enr‡	Red†
Bn is better	**42**	**37**	22	8
Bn is slightly better	17	25	**34**	**55**
B is slightly better	14	20	25	26
B is better	17	8	9	1

Table 3: Result on human evaluation. Bn is a BART model trained with Noise-Annealing curriculum and appropriateness, and B is a BART model without curriculum learning. Enr stands for the Enron subject dataset and Red stands for the Reddit title dataset. † and ‡ indicate statistical significance that Bn receives more votes of "better" and "slightly better" than B (using a chi-square test; † $p < 0.01$, ‡ $p < 0.05$).

sequence-to-sequence models from a single noisy corpus. We conducted experiments of curriculum learning on summarization tasks. We experimented on two datasets, Enron subject and Reddit title datasets and three summarization models: BART, transformer, and sequence-to-sequence with attention. The result showed our method improved the performance of the models.

We also conducted experiments with three types of curriculum learning (One-Pass, Baby step, and Noise-Annealing curricula), and concluded that choosing small data for fine-tuning in the last phase of the training was important for pretrained models and generalization with various data in the beginning phase was important for non-pretrained models. For future work, we seek for more effective methods to find data for fine-tuning.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)*, page 41–48.

Volkan Cirik, E. Hovy, and Louis-Philippe Morency. 2016. Visualizing and understanding curriculum learning for long short-term memory networks. *ArXiv*, abs/1611.06204.

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

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of Reddit posts with multi-level memory networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2519–2531.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *Proceedings of the 15th European Conference on Machine Learning*, page 217–226.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. 2019. Reinforcement learning based curriculum optimization for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2054–2061.

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

Junjie Li, Haoran Li, and Chengqing Zong. 2019. Towards personalized review summarization via user-aware sequence network. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 6690–6697.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. 2020. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. volume 108 of *Proceedings of Machine Learning Research*, pages 4313–4324. PMLR.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*.

Kazuki Matsumaru, Sho Takase, and Naoaki Okazaki. 2020. Improving truthfulness of headline generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1335–1346.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Wei Wang, Isaac Caswell, and Ciprian Chelba. 2019. Dynamically composing domain-data selection with clean-data selection by “co-curricular learning” for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1282–1292.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 133–143.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations (ICLR) , 2017*.

Rui Zhang and Joel Tetreault. 2019. This email could save your life: Introducing the task of email subject line generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 446–456.

Yikai Zhou, Baosong Yang, Derek F. Wong, Yu Wan, and Lidia S. Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6934–6944.

A Parameters of Summarization Models

The dimensions of hidden layers of both Seq2seqAtt and Transformer are 256. The dimensions of word embeddings of Seq2seqAtt and Transformer are 300 and 256 respectively. Similarly as Appropriateness Estimator, we use GloVe as initial parameters of word embeddings of Seq2seqAtt. The mini-batch size is 64 on all three models. The size of beam search is set 8. We use Adam as an optimizer of Seq2seqAtt and Transformer, and the learning rate is 0.0007.

For the optimization of BART, we use AdamW (Loshchilov and Hutter, 2019), where the learning rate is 3e-5, β_1 is 0.9, β_2 is 0.999, and eps is 1e-8.

B In Which Segment the Model Achieves Best Result?

One-Pass curriculum and Noise-Annealing curriculum both fine-tune models with smaller amount of data in each segment. Investigating at which segment the model gets best validation scores, we can validate which segment is the best data for fine-tuning.

Table 4 shows the numbers of the best segments. The experiment is conducted five times and the average and the standard deviations are shown on the table. The number of segments is set 10, so the tenth segments have longest targets or highest appropriateness. When we use target length, the model gets best validation scores on earlier segments whereas when we use our appropriateness, the model gets best validation scores on later segments. This means too long summaries are not appropriate to fine-tune summarization models, whereas the segments with the highest appropriateness computed by our Appropriateness Estimator are.

	Target Length		Appropriateness	
	Enron	Reddit	Enron	Reddit
Baby step	6.5± 2.4	6.5± 2.9	7.1± 2.6	6.8± 2.9
One-Pass	3.1± 2.3	2.9± 1.3	6.6± 3.0	5.6± 3.6
Noise-Annealing	3.5± 0.9	4.5± 1.4	7.2± 2.4	7.4± 2.4

Table 4: The segments at which each model gets best validation metric (ROUGE-1-F). Mean and standard deviation values of 5 experiments are shown. The number of the segments is 10.

C Examples of data with High and Low Appropriateness

Table 5 and 6 shows the examples of source and target pairs with low appropriateness. The target length of the examples are not short, but many of the subjects or titles include information that is not described in the source texts. In the case of Reddit, many of the source texts begin as the next sentences of the titles, and the information on the titles are not repeated on the source texts. These do not meet the requirement for the training data of summarization. By contrast, target texts of high appropriateness shown in Table 7 and 8 explain the descriptions in the source texts.

Subject	Source Text	App
How do I eat crow and still make it tasty???????	To all of our esteemed & prized Internet Banking (IBS) Clients: You probably have begun to wonder does anyone ever return phone calls (or e-mails). We really do but have been holed up trying to complete this project ASAP. We knew that this might upset some clients, but we found that this is the quickest way to fix this problem and get our users back on-line. We have finalized the movement of your customer account's and re-linked your additional accounts back to your primary Login ID (Acct). This process was more daunting than we anticipated and having to verify 1100 IBS users and accounts (to insure your data integrity and confidentiality) was even more grueling than expected. After verifying data and Login IDs, we believe that we are on the right track. Some of you (our IBS clients) might notice your accounts displayed more than once or additional accounts displayed that you may (or may not) want displayed on your IBS screen. Should this be the case, please e-mail (or phone) the information to us and we will remove this information right away. To our IBS Billpayer clients: During this process, we seem to have misplaced (blown away) your bill payment information (particularly anyone that has [had] reoccurring payments scheduled to process on particular days). This has become our highest priority to retrieve this information, thus alleviating the process of having to request that our IBS Billpayers re-enter this information. We are going to waive all bill payment charges for the months of November & December, 2001 to try and regain your confidence (and support). Again, we deeply express our regrets and hope that we (yourselves and ourselves) do not have to go through this process again. Should you be one of our IBS clients that still has not gained access to your account information, please refer to the following information:	0.0002
Additional New Works; 5/2/01 Floor Meeting, 37th Floor	In order to maximize the potential synergies between the various mid- and back- office functions, to decrease replication errors, and to increase communication standards; are there any plans of creating a platform or reference center to bridge the differences between the systems, processes, and terminology of the various departments. Perhaps a common resource center offering access to on-line system manuals and business unit overviews (which currently exist only in paper form or in some cases within the actual system database). The reason: With Enron's size and transaction volume, many of the functions and the data managed by various groups within Enron (i.e. credit, risk, settlements, volume management, global contracts, global counterparties, global rates, and the commercial systems) are fragmented. Having participated in various process reviews and trouble-shooting/clean-up projects there seems to be a large disconnect between groups operating in various systems. These disconnects; rather they be lack of information or understanding of how data flows between systems, how the data managed within each system impacts other upstream or downstream systems, or how the business processes within one group/system impacts the overall functionality of other groups, create large cracks producing an opportunity for mismanaged data, incomplete business reports, and increased risk to Enron. The ideal objective; increase communication standards through a better understanding of system data functions/requirements and business processes, decrease system downtime and replication errors through a better understanding of the data relationships between systems, maximize department-to-department synergies (left hand knows what the right hand is doing), eliminate repetition, and further reduce potential risks to Enron due to information/business process oversight.	0.0052
Returned mail: Host unknown (Name server: enron: host not found)	Danny, I'm resending as I had the same problem Cindy did. I'll give you a call later today after I've talked to Harris to discuss the various Gallup scenarios to make sure you and I are on the same page. The plan that makes the most sense in my mind is to ram the 10,000/d project through asap, with no firm contracts to preserve our options on a NEWCO structure. We'll simultaneously implement a new approach on San Juan fuel transport if possible and then throw the big expansion into the hopper at FERC in January as Stan suggested. I hope that timetable is doable–it all depends on	0.0061
GREAT NEWS ****FERC Order on Morgan Stanley Complaint Against ISO	See below. this is one of the issues that concerned us more than price caps, because it could limit our ability to move power to other markets in the west. In addition, if you get questions from the analysts on "reregulation" or price caps it is worth pointing out that the high prices prevailing in many markets help our retail sales pitch to end use customers and create opportunities for our wholesale price risk management services ... even a $250 price cap is 5-10 times what large customers are accustomed to paying.	0.0087
Noram Rigs	Richard Sanders has asked me to set up a meeting regarding the above referenced. The following participants are: Lisa Mellencamp Mark Peterson John Hopley John Enerson Richard Sanders It looks as though, this Friday, Aug. 13. at 2:00 will be a good time for everyone. Please let me know if this time is convenient for you. The location will be 38C1. (I tried to contact you by phone today, but your extension 31406 was forwarded to a non working number). My extension is 39402, if you wish to call me. Thanks	0.0093

Table 5: Example of data with low appropriateness (Enron subject dataset). App stands for "appropriateness"

Title	Source Text	App
logging onto my wife's facebook account.	I read every comment and personal message and thank everyone that gave their advice. I decided I would confront her about it after we put the kids down last night. I decided I would start out by asking her the essential questions. Do you love me? Do you want to be with me right now? Do you want to spend the rest of your life with me? Then I would tell her that when we took our vows we said we would stick together through thick and thin, and that right now we are in some pretty thick shit. I would tell her that we both breached each other's trust and we both had some explaining to do. Then we would progress the conversation from there. I have to add that since the op and prior to the confrontation the conversations between the 2 of them continued to go on. He is a pathetic little prick that obviously does this with countless other women because it is easy and safe and he doesn't have to put himself out there and risk getting hurt. At one point he even told her he loved her. She replied by telling him that wasn't appropriate and that they were just friends and that was how it was going to be. So this is pretty much how it went down: she said she loved me, wanted to be with me, and always wanted to be with me. She admitted that the things being said in the conversation were inappropriate and when I asked her why she did it she told me exactly what I knew she would: "it is really nice being told how pretty you are and getting that kind of attention." I asked if there was anyone else she was conversing with like that and she said no. I also asked if she had ever cheated on me with anyone physically and she said no. I told her I had been faithful since day 1 and I needed to know that she had been to. She assured me she was. The db she was talking to had went to school with her for 1 year in high school and now lives in north carolina. We are in arkansas. I walked away from the situation feeling really good about it all and I could tell that she was sincere. We ended up making crazy love all over the house, doing it again before bed, and again when we woke up. She apologized and I told her I would get back to making her feel like a woman so she didn't have to seek that out somewhere else. Say what you will but I think it ended as well as it possibly could have.	1.67e-04
Accidentally drinking 3 day old coffee w/milk that was sitting on my desk next to my new coffee.	just happened. will update with further details as they emerge.	2.67e-04
Backing my e class into my wife's c class mercedes	My wife had been out of town all week at a sales conference. Our driveway makes at with one car pulling to the left into our carport and one car that pulls forward to park on a concrete slab. Initially my wife was supposed to get our kids from daycare but her flight was running late so she decided to come by the house first to pick me up so we could go out to dinner. I was finishing some work projects at home when she came running in from the airport. I didn't realize we were on the verge of not picking the kids up on time. The daycare charges something like $10 a minute if you're late and it was a friday. She was gathering some things for our toddler (you can't go out with a 3 yo unless you're prepared to bring a toy store to entertain them with). I had the bright idea that I would back out of the carport and pull up so her passenger door would be readily accessible when she came out the back door (i had been pulling out that way all week so I could pull out into the street rather than back out). In a hurry, I slammed my car in r and jammed on the gas. Boom! I hit her car just as she was coming out the door. Toddler toys go flying everywhere (mostly at my head). We didn't speak all the way to the daycare until I just started laughing hysterically. I mean really. What else could you do?	0.0127
Leaving a 12-pack of beer in the bottom of a shopping cart in the grocery store parking lot.	I went back to get it 30 minutes later and it was still there :)	0.0130

Table 6: Example of data with low Appropriateness (Reddit title dataset). App stands for "appropriateness"

Title	Source Text	App
Power Indices	IntercontinentalExchange Firm Power Price Bulletin = For Power Delivered on Wednesday, October 24, 2001 = (Trade Date of Tuesday, October 23, 20= 01) Click here to access index history . * volume represents sell-side only * Hub=09High=09Low=09Wtd Avg Index=09Change ($)=09Vol (Mwh)=09 Cinergy=09 $28.50=09 $24.00=09 $26.90=09+ 6.40=09= 81,600=09 Comed=09 $26.50=09 $23.00=09 $24.25=09+ 5.34=09 = 4,800=09 Entergy=09 $25.50=09 $22.70=09 $24.72=09+ 2.62=09= 20,000=09 Nepool=09 $38.70=09 $38.50=09 $38.56=09+ 1.06=09 = 7,200=09 PJM-West=09 $27.50=09 $25.75=09 $26.35=09+ 2.14= =09 50,400=09 TVA=09 $30.50=09 $24.25=09 $27.38=09+ 6.75=09 = 11,200=09 Includes all trades done from 6 AM to 11= AM Central Prevailing Time on the trade date specified for financially fir= m power delivered during the on-peak hours (6 AM - 10 PM CPT for Eastern hu= bs / 6 AM - 10 PM Pacific Prevailing Time for Western hubs) on the delivery= date(s) specified. IntercontinentalExchange is the world's most liquid = trading platform for over-the-counter energy and metals. Active markets in= clude North American power and natural gas, global crude and refined oil pr= oducts, and precious metals. Traded instruments include forwards, swaps, a= nd options. In order to receive the proprietary information contained in this email, yo= u acknowledge and agree that you shall not further disseminate the Intercon= tinentalExchange Market Data contained herein to any person or entity witho= ut the express written consent of IntercontinentalExchange. Furthermore,= you acknowledge that (1) IntercontinentalExchange has exclusive and valuab= le property rights in this data; (2) IntercontinentalExchange's data is bei= ng made available to you only for your own business or personal activities;= and (3) you cannot communicate the data, in any form, to any other person = or entity without the express written consent of IntercontinentalExchange. This data is provided to you free of charge. IntercontinentalExchange rese= rves the right to cancel this service at any time for any reason or no reas= on at all. You agree that IntercontinentalExchange does not make any representations o= r warranties, express or implied, with respect to the data. To become an Exchange Participant or inquire about the indices, please cont= act sales@intcx.com . To unsubscribe from this service, click here unsubscribe . ?Copyright IntercontinentalExchange, Inc. 20= 01, All Rights Reserved.	1.0
Nitrogen and Sulfur reporting and Recordkeeping for Turbines	For those teams that have turbines installed after 1990 and/or for those turbines which have undergone power unit changouts, the following recordkeeping and monitoring conditions apply: 1) DAILY recordkeeping of nitrogen and sulfur must be taken of the fuel gas which supplies the applicable turbine(s). 2) This recordkeeping consists of electronic recording (gas chromtograph for nitrogen and delmar or equivelant for sulfur) or stain tubes may also be used for sulfur. These DAILY records include measurements on Saturdays and Sundays. 3) The measurement must be taken at the location. An exception to this is that the nitrogen and sulfur measurements may be taken upstream or downstream of the applicable turbine facility provided that there are no natural gas deliveries into the pipe which would interfere or dilute/increase the measurements for the applicable turbine fuel gas. 4) Fuel gas records in hard copy form or equivalent for the nitrogen and sulfur must be maintained at the facility or at a central location for easy retrival. 5) A turbine facility may waiver out of this nitorgen and sulfur daily recordkeeping requirement by obtaining a custom fuel monitoring schedule (CFMS) from the EPA. Approval of a CFMS allows a greatly reduced recordkeeping and reporting for nitrogen and sulfur. CFMS requests have been submitted for the following facilities: P-1 C/S Plains Turbine C/S Atoka No 2 C/S Monument C/S Crawford C/S Bloomfield C/S Approvals have not as yet been obtained. Until issuance of a CFMS, an applicable facility is required to continue daily sampling for nitrogen and sulfur. Facilities which have received CFMS from the EPA include: La Plata C/S P-2 C/S Please be advised that there may be certain reporting requirements that might be required for each CFMS. I would strongly advise that the La Plata and Panhandle teams review their CFMS and include reporting dates into MCS, so that the deadlines and reportings are not missed. If you have a turbine facility which is subject to the nitrogen and sulfur reporting requirements and would like to reduce the reporting burden, contact Butch or myself." Nitrogen and Sulfur reporting and Recordkeeping for Turbines	1.0
TW/ Lonestar Ward and Pecos Counties interconnect bi-directional–A-release	The following is a level "A" cost estimate to make TW/ Lonestar existing interconnects bi-directional. TW/ Lonestar at Ward County (50 to 60 mmcf/d) According to Operations this is already bi-directional . The only things are required on this one is to take the flapper out of the check-valve and blow down the gas in 5.33 miles of 12". Cost of gas loss& labor = $8,000 TW/ Lonestar at Pecos County (100 mmcf/d) A): TW/ Lonestar interconnect Scope: On this one we need a bi-directional valve skid using the existing meter run. Cost of material& labor= $ 195,000 B): Pecos Compressor Station In order to make this interconnect bi-directional we also need to make the station (two-compressor units) bi-directional. Scope: Install outlet from Lonestar I/C to inlet filter with 12" piping& valves. Unit discharge would be modified to tie in to West Texas-20" Cost for material& labor= $ 330,000. If you need more accurate costs (B -release) please let me know .	0.999

Table 7: Example of data with high appropriateness (Enron subject dataset). App stands for "appropriateness"

Title	Source Text	App
asking for a coffee without milk	i started a new job a few weeks ago. i was sat at my desk typing away, one of the guys in the team starts asking around to see if anyone wants a hot drink. he looks at me, raising his eyebrows expectantly, i think for a moment and say "i'll have a coffee please" i then realise that he doesn't know whether or not i take milk, so a second or so after asking for a coffee, i complete the sentence with "..black" there are 3 problems with this. between me saying 'coffee please' , and 'black' he'd moved from right next to me, to a few steps away, so i had to say the last word a lot louder in this time, our boss walked out of a meeting room. the guy getting the coffee is black.	0.9665
asking if my roommate had any plans for mother's day.	yesterday, technically, i was at home making myself a nice meal because i couldn't be with my family for mother's day due to distance. as i'm preparing my dinner, my roommate came into the kitchen. thinking i would be a good roommate and strike up some passing conversation, i asked him if he had any plans for mother's day, to which he replied that his mom had died just last month. he hasn't exactly made this super well known in the house, but i had seen a fb post of his last month mentioning this. i felt like the most insensitive asshole ever and apologized as well as i could. but i'll always feel bad about that one.	0.9665
while eating cereal.	was having breakfast, which consisted of a coffee and cereal. lately i've been feeling under the weather so i've been taking vitamins with my breakfast too. i put the vitamins in my mouth and realize i should have something to wash it down with. so i take a big spoonful of cereal. that's when it dawns on me. i can't swallow the cereal without chewing, and i have vitamins in my mouth (non-chewable). i decide rather than risk choking to death on granola i have to chew. before long the vitamins are ground up and mixed with the cereal in my mouth. it was vile. honestly one of the most bitter things i've ever tasted.	0.9665
running against an electricity closet inside my airbnb apartment and getting a concussion.	this happened two days ago but i couldn't post it due to my head hurting too much. i'm in florence currently and the apartment i'm staying in is not made for tall people. i'm not even that tall (6ft"1). so here comes the fuck up. there are two rooms in my apartment and my gf was chilling in the second bed room, for which you need to go down steps to get to. however there is a electricity closet sticking out so if you're taller than 5ft"9 you will bump your head. <url> so i'm sitting in one bed room and suddenly my gf screams like there is something wrong. so naturally the concerned bf that i am jumps up and starts running towards here. in the moment i did not care or think about this ridiculous electricity closet sticking out that's made of fucking stone. not wood, nope, stone. so as i'm running at bolt speed i look down to prepare to run down the steps and literally hit my head at full speed against the closet, do a flip, and fall down the stairs. next thing i know i'm in the hospital and getting a ct scan. ps. sorry for format, posted this from my phone.	0.9665

Table 8: Example of data with high appropriateness (Reddit title dataset). App stands for "appropriateness"

Evaluating language models for the retrieval and categorization of lexical collocations

Luis Espinosa-Anke[1], **Joan Codina-Filbà**[2], **Leo Wanner**[3,2]
[1]School of Computer Science and Informatics, Cardiff University, UK
[2]TALN Research Group, Pompeu Fabra University, Barcelona, Spain
[3]Catalan Institute for Research and Advanced Studies (ICREA), Barcelona, Spain
`espinosa-ankel@cardiff.ac.uk`, `joan.codina|leo.wanner@upf.edu`

Abstract

Lexical collocations are idiosyncratic combinations of two syntactically bound lexical items (e.g., *"heavy rain"*, *"take a step"* or *"undergo surgery"*). Understanding their degree of compositionality and idiosyncrasy, as well their underlying semantics, is crucial for language learners, lexicographers and downstream NLP applications alike. In this paper we analyse a suite of language models for collocation understanding. We first construct a dataset of apparitions of lexical collocations in context, categorized into 16 representative semantic categories. Then, we perform two experiments: (1) unsupervised collocate retrieval, and (2) supervised collocation classification in context. We find that most models perform well in distinguishing light verb constructions, especially if the collocation's first argument acts as a subject, but often fail to distinguish, first, different syntactic structures within the same semantic category, and second, finer-grained categories which restrict the set of correct collocates[1].

1 Introduction

Language models (LMs) such as BERT (Devlin et al., 2018), and its variants SpanBERT (Joshi et al., 2020), ALBERT (Lan et al., 2019), RoBERTa (Liu et al., 2019), etc. have proven extremely flexible, as they behave as unsupervised multitask learners (Radford et al., 2019), and can be leveraged in a wide array of NLP tasks almost out-of-the-box (see, e.g., the GLUE and SuperGLUE results in Wang et al. (2019b) and Wang et al. (2019a), respectively). They have also been harnessed as supporting resources for knowledge-based NLP (Petroni et al., 2019), as they capture a wealth of linguistic phenomena (Rogers et al., 2020). Recently, a great deal of research analyzed the degree to which they encode, e.g., morphological (Edmiston, 2020), syntactic (Hewitt and Manning, 2019), or lexico-semantic structures (Joshi et al., 2020). However, less work explored so far how LMs interpret phraseological units at various degrees of compositionality. This is crucial for understanding the suitability of different text representations (e.g., static vs contextualized word embeddings) for encoding different types of multiword expressions (Shwartz and Dagan, 2019), which, in turn, can be useful for extracting latent world or commonsense information (Zellers et al., 2018).

One central type of phraselogical units are *lexical collocations*, defined as restricted co-occurrences of two syntactically bound lexical items (Kilgarriff, 2006), such that one of the items (the *base*) conditions the selection of the other item (the *collocate*) to express a specific meaning. For instance, the base *lecture* conditions the collocates *give* or *deliver* to express the meaning 'perform', the base *applause* conditions the selection of the collocate *thunderous* to express the meaning 'intense', and so on. Lexical collocations are of high relevance to lexicography, NLP and second language learning alike, and constitute a challenge for computational models because of their heterogeneity in terms of idiosyncrasy and degree of semantic composition (Mel'čuk, 1995).

In this paper, we analyze a suite of LMs in the context of two tasks that involve lexical collocation modeling. First, **unsupervised collocate retrieval**, where we *mask* a collocation's collocate (e.g., *"heavy"* in *"heavy rain"*), and quantify how well a LM of choice (BERT in particular) predicts, via its masked language modeling (MLM) objective, a valid collocate for that particular base ({*"heavy"*, *"torrential"*, *"violent"*, ...} for the base *"rain"* and the meaning *intense*). Second, **su-**

---

[1] The resources associated with this paper are available at `https://github.com/luisespinosaanke/lexicalcollocations`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1406–1417
April 19 - 23, 2021.

pervised in-context collocation categorization, where we fine-tune LMs on the task of predicting a semantic category of a collocation in terms of its *lexical function* (LF), given its sentential context; cf. Section 3.1. Modeling, recognizing, and classifying collocations in corpora has obvious applications for automatically creating and expanding lexicographic resources, as well as for various downstream NLP applications, among them, e.g., machine translation (Seretan, 2014), word sense disambiguation (Maru et al., 2019), or natural language generation (Wanner and Bateman, 1990). The two main contributions of this paper thus are:

1. A "collocations-in-context" dataset, with instances of collocations of 16 different semantic categories (in terms of LFs) in context, and with a fixed and lexical (i.e., no overlapping) train/dev/test split (Section 3).
2. An evaluation framework for assessing the degree of compositionality of lexical collocations, pivoting around two tasks: unsupervised *collocate* retrieval (Section 4) and in-context *collocation* categorization (Section 5).

Our results suggest that modeling collocations in context is a challenge, even for widely used LMs, and that this is particularly true for less semantic (and thus less compositional and more idiosyncratic) collocations. We also find that jointly recognizing the semantics and the syntactic structure (e.g., whether the collocate acts as subject or object in verbal constructions) of a collocation also constitutes non-trivial challenges for current architectures. Moreover, as a byproduct of our analysis, we also find an interesting behaviour in LMs when modeling antonymy in adjectives, specifically that their representations undergo substantial transformations as they flow through BERT's transformer layers, with many contextualized embeddings clustered together in the tip of a narrow cone that seems to represent adjectives in collocations denoting intensity (*"heavy"* rain) and weakness (*"minor"* issue).

2 Related Work

In this section, we discuss related works in two methodological areas that are relevant to this paper, namely conditioning MLMs (Section 2.1) and recognition of multiword epressions (MWEs) (Section 2.2).

2.1 Conditioning MLMs

A Masked Language Model (MLM) can be used as a *proxy* for gaining insights into how language is encoded by the weights of the (usually transformer-based) LM architecture. Moreover, simply asking an LM to predict words in context (without task-specific fine-tuning) has proved useful in NLP applications dealing with lexical items (affixes, words or phrases). For example, Wu et al. (2019) use BERT's MLM for augmenting their training data in sentiment analysis tasks; Qiang et al. (2019) use BERT for lexical simplification by *conditioning* the predictions over the $[\mathrm{MASK}]$ token by providing the original sentence as context; and Zhou et al. (2019) obtain SotA results in lexical substitution by conditioning BERT via embedding dropout on the target (unmasked) word. Inspired by the findings in these works (especially Qiang et al. (2019)), we will explore the predictions of BERT over masked lexical collocations (with and without conditioning) in Section 4, with the aim to understand whether these predictions can be used to measure the idiosyncrasy of the underlying semantics of a lexical collocation, i.e., whether the restrictions imposed by a collocation's base are due to the frozenness of the phrase itself or, on the contrary, sentential context is neccessary.

2.2 Distributional Lexical Composition

Building representations that account for non-compositional meanings within the broader spectrum of encoding semantic relations between words is a long-standing problem in computational semantics (Baroni and Zamparelli, 2010; Mitchell and Lapata, 2010; Boleda et al., 2013). Interestingly, there seems to be little agreement on how these representations should be defined, with recent attempts focusing on verbal multiword expressions (see an overview of approaches in Ramisch et al. (2018)), phrases of variable length encoded via LSTMs, based on their definitions (Hill et al., 2016), or arbitrary lexical and commonsense relations between word pairs for downstream NLP. As a testimony of the broad methods explored in the most recent literature, let us refer to, for instance, the combination of word vector averages with conditional autoencoders (Espinosa-Anke and Schockaert, 2018), expectation maximization (Camacho-Collados et al., 2019), LSTMs for predicting word pair contexts (Joshi et al., 2019), and explicit encoding of generalized lexico-syntactic patterns (Washio and Kato,

LF	BERT input	Masked sentence	Pred. collocates	Orig. collocate
Oper1	Masked sent	iran feared that the u.s and israel may [MASK] an air raid on its controversial nuclear facilities	perform, conduct, mount,	launch
	Orig sent [SEP] Masked sent	iran feared that the u.s and israel may launch an air raid on its controversial nuclear facilities [SEP] iran feared that the u.s and israel may [MASK] an air raid on its controversial nuclear facilities	launch, conduct, order	launch
Real1	Masked sent	if that happens, lindsey will use explosive triggers to [MASK] the landing gear	destroy, stop, remove	lower
	Orig sent [SEP] Masked sent	if that happens , lindsey will use explosive triggers to lower the landing gear [SEP] if that happens , lindsey will use explosive triggers to [MASK] the landing gear	destroy, stop, trigger	lower
Magn	Masked sent	the EU is driving a [MASK] bargain over Swiss demands for greater access to EU airspace .	plea, hard, new	hard
	Orig sent [SEP] Masked sent	the EU is driving a hard bargain over Swiss demands for greater access to EU airspace . [SEP] the EU is driving a [MASK] bargain over Swiss demands for greater access to EU airspace .	plea, hard, tough	hard

Table 1: Illustrative behaviour of BERT when prompted to predict a collocate in the position of a masked token, for three LFs: Oper1 (‘launch an air raid’), Real1 (‘lower the landing gear’) and Magn (‘hard bargain’), under two settings, when not conditioned (Masked sent) and when conditioned on the original (unmasked) sentence (Orig sent [SEP] Masked sent).

2018). Parting ways with the above works, in this paper we will follow the experimental setting described in Shwartz and Dagan (2019), based on injecting sentential contexts into multiword expressions (in our case, only lexical collocations) to leverage the contextual nature of current LMs. However, our goal is not to compare different combinations of feature-extraction and training/fine-tuning methods, but rather to understand lexical collocations’ learnability, idiosyncrasy and their internal vector-space representations.

3 Data and Resources

3.1 Lexical Collocations

Let us first introduce the notion of lexical collocation and LF. The term *collocation* has been used in computational linguistics research to denote two different concepts. On the one hand, following Firth (1957), Church and Hanks (1989); Evert (2007); Pecina (2008) and others, a collocation has been assumed to be a combination of words that have the tendency to occur together in discourse. Typical examples are *doctor – hospital*, *mop – bucket*, *real – estate*, *look – for*, etc. On the other hand, for instance, Wanner et al. (2006); Gelbukh and Kolesnikova. (2012); Rodríguez Fernández et al. (2016); Garcia et al. (2017) adopt the definition that is common in lexicography and phraseology (Hausmann, 1985; Cowie, 1994; Mel’čuk, 1995), according to which, a collocation is an idiosyncratic combination of two lexical items, the base and the collocate, as defined above in Section 1. This interpretation states that collocations are phraseological units, although their degree of compositionality can vary. For instance, *win* [*a*] *war* is perceived to possess a higher degree of (free) composition than, e.g., *hold* [*a*] *meeting*, and *heavy rain* is less compositional than [*a*] *well-justified argument*. We adopt this definition of the notion of collocation, and in order to avoid any confusion, we refer to it, following Krenn (2000), as *lexical collocation*.

Lexical collocations can be typified with respect to the meaning of the collocate and the syntactic structure formed by the base and the collocate. LFs provide a fine-grained typology of this kind (Mel’čuk, 1996). An LF can be considered a function $f(L)$ that delivers for a base L a set of synonymous collocates that express the meaning of f. Where pertinent, f also codifies the subcategorization structure of the base+collocate combination. LFs are assigned Latin acronyms as names; cf., e.g., “Oper1” (‘operare’), which means ’perform’ and realizes the first argument of the base as subject: Oper1(*lecture*) = {*deliver*, *give*, *hold*}; “Magn” (‘magnum’), which stands for ‘intense’: Magn(*applause*) = {*thunderous*, *loud*, ...}.[2]

The encoding of LFs in NLP research has in recent years revolved around applying word embeddings-based techniques, e.g., in terms of linear projections (Rodríguez Fernández et al., 2016) and semantic generalizations (Espinosa-Anke et al., 2016). Recently, Shwartz and Dagan (2019) analyzed, from the perspective of “static” vs. “contextualized” representations and their applicability to studying compositional phenomena like “meaning shift”, one specific type of lexical collocations, namely *light verb constructions* (LVCs), which are well illustrated by the LFs Oper1 and Oper2 (and also, partially, by Real1 and Real2). While we find that the above research directions (i.e., embeddings-based and contextualized representations for modeling MWEs) are complementary, in this work, we

---

[2] For simplicity we will write Oper1(*lecture*) = *deliver*, etc.

LF	semantic gloss	example
Oper1	‘perform’; 1st argument → subject	Oper1(*support*) = *lend*
IncepOper1	‘begin to perform’; 1st argument → subject	IncepOper1(*impression*) = *gain*
Oper2	‘undergo’; 2nd argument → subject	Oper2(*support*) = *find*
Real1	‘realize’; 1st argument → subject	Real1(*accusation*) = *prove*
Real2	‘apply’; 2nd argument → subject	Real2(*support*) = *enjoy*
AntiReal2	‘fail to apply’; 2nd argument → subject	AntiReal2(*war*) = *lose*
CausFunc0	‘cause the existence’	CausFunc0(*hope*) = *raise*
Caus1Func0	‘cause the existence; 1st argument’	Caus1Func0(*hope*) = *gain*
LiquFunc0	‘cause termination of the existence’	LiquFunc0(*hope*) = *destroy*
IncepPredPlus	‘increase’	IncepPredPlus(*temperature*) = *rise*
Magn	‘intense’	Magn(*smoker*) = *heavy*
AntiMagn	‘little’, ‘weak’	AntiMagn(*smoker*) = *occasional*
Ver	‘genuine’	Ver(*demand*) = *legitimate*
AntiVer	‘non-genuine’	AntiVer(*demand*) = *illegitimate*
Bon	‘positive’	Bon(*performance*) = *good*
AntiBon	‘negative’	AntiBon(*performance*) = *poor*

Table 2: LFs used in this paper. The ‘semantic gloss’ column provides both a definition and the actantial structure, which is required in cases where one LF may express the same semantics but with a different syntactic structure (e.g., Real1 vs. Real2).

specifically focus on the existing (and learnable) knowledge LMs have concerning lexical collocations, and whether they can be used to recognize and categorize LFs in free text.

For our experiments, we use, as initial lexical collocation source, a **collocations dataset**, LEXFUNC (Espinosa-Anke et al., 2019), which we have extended to cover a wider range of LFs (listed in Table 2). The original LEXFUNC dataset and this extended version are both the result of an initial collection of collocations categorized into LFs made available by Igor Mel’čuk. Each collocation has been manually lemmatized, and bases and collocates have been manually annotated with part-of-speech tags and their syntactic dependency relation.

With the lexical collocations of the extended LEXFUNC dataset at hand, we first compile from the English Gigaword[3] a **collocations corpus**, which contains the occurrences of these lexical collocations. In principle, the identification of a given collocation in corpora is a straightforward procedure, as we know its elements (base and collocate) and the syntactic dependency relation between them. However, automatic dependency parsing is far from perfect, which complicates the task. Therefore, and in order not to lose any relevant collocation occurrence in the GigaWord corpus, we apply a cascaded procedure for their identification on the lemmatized and POS- and head-modifier relation tagged GigaWord.[4] In the first stage, we identify sentences in which between the collocation elements in question one of the relevant syntactic dependency relations has been identified. In the second (more relaxed) stage, we match adjacent lemmatized collocation elements and their PoS tags. In the third stage, finally, we match lemmatized collocation elements and their PoS tags within a distance of up to 5 tokens. While this procedure inevitably introduces some noise (we might retrieve sentences where base and collocate co-occur, but not as a collocation), we performed a manual inspection on a random sample, and calculated precision of our collocation retrieval strategy, which resulted in >0.95. This confirms the quality of our retrieval strategy, and hence, our resource.

In terms of corpus statistics, Table 3 indicates the number of sentences for each LF distributed across training (70% of the sentences), development (15%) and test (15%) sets. The split was done maintaining this proportion across all LF. Note that these splits are constructed such that there are no overlapping collocations, in an effort to avoid the well-known phenomenon of *lexical memorization* (Levy et al., 2015), which may artificially inflate the results on the test set. The number of different collocations per split, globally and for each LF, also maintains the same proportions (70/15/15 ±1%), such that, e.g., AntiReal2 has 55 different collocations in the 942 sentences of the training set and 11 different collocations in the development and test sets, distributed across 205 and 200 sen-

---

[3] Gigaword contains newswire material, a register balancing the staticity of encyclopedic English (e.g., Wikipedia) and the noisiness of user-generated text, and is a rich source of lexical collocations in well-formed grammatical contexts (Rodríguez Fernández et al., 2016) `https://catalog.ldc.upenn.edu/LDC2011T07`.

[4] We preprocess GigaWord with the NLP4J parser `https://emorynlp.github.io/nlp4j/`.

Label	Train	Dev	Test	Total
Magn	28,748	6,113	6,164	41,025
Oper1	11,746	2,517	2,493	16,756
Real1	3,481	746	743	4,970
AntiMagn	2,959	638	649	4,246
IncepOper1	2,489	541	539	3,569
Oper2	2,408	515	520	3,443
AntiVer	1,874	397	405	2,676
AntiBon	1,815	385	393	2,593
CausFunc0	1,714	370	367	2,451
Real2	1,570	336	337	2,243
Bon	1,471	298	315	2,084
LiquFunc0	1,398	301	297	1,996
AntiReal2	942	203	200	1,345
Ver	926	198	196	1,320
Caus1Func0	686	151	149	986
IncepPredPlus	624	147	138	909
Total	64,851	13,856	13,905	

Table 3: Statistics (in number of sentences) of our *collocations in context* dataset (ordered by frequency).

tences respectively. In the overall corpus, there is an average of 18 samples per collocation (*work hard* being the most frequent one with 102 samples). *Hope*, *attack*, *criticism*, *fire* and *thread* are bases that each co-occur with more than 30 different collocates, across most LF. These bases are also among the ones with more samples in the corpus. On the other side, half of the bases are combined with one single collocate only. Overall, the statistical properties of our dataset arguably make it a faithful replica of the distribution of collocations in, at least, newswire corpora. At the same time, it is a challenging dataset, as the results we report in this paper suggest.

4 Experiment 1: Collocate Retrieval

4.1 Setup

In the first experiment, we aim to analyze how well an MLM retrieves valid collocates for a given base when being provided with the original (sentence-level) context. We use BERT (`bert-base`) (Devlin et al., 2018), as it is the *de-facto* model on top of most specialized and distilled/quantized language models. Its behaviour should thus be a good proxy for the general distributional behaviour of lexical collocations. This experiment serves, first, as an opportunity to understand *how much* semantics that is underlying LFs can be encoded via a MLM pretraining objective, and second, as a testbed for exploring conditioning strategies often used in tasks involving data augmentation and lexical substitution and simplification (cf. Section 2.1). Since this is an "in-context collocate retrieval" task, we consider it a ranking problem. Intuitively, if BERT is able to retrieve a base's valid collocates (e.g., {*heavy*, *torrential*, *violent*, …} for *rain* as base for Magn) in the position of a masked token, this could mean that: (1) the sentence is giving enough context for the model to "know" the lexical restrictions involved in that collocation, and/or (2) the LF is sufficiently frozen, and therefore the base alone may restrict which collocates are acceptable. For the first point, and continuing with the *heavy rain* example, consider the following sentence.

(1) *Hurricane Katrina brought* [MASK] *rain to Louisiana.*

Intuitively, we would expect the sentential context to be informative enough for the model to select *heavy* or any other collocate denoting the notion of intensity, and restricted by the presence of the base *rain*. In fact, here, BERT predicts *heavy* with 79.5% probability. However, in example (2)

(2) *Policeman earns applause for staying on duty in* [MASK] *rain.*

there is much lesser evidence for the *rain* to be 'intense', and in fact BERT predicts here *'the'* with 85.1% probability. This disparity lets us investigate ways to prompt BERT to select *heavy* or any other valid collocate for example (2). Thus, in addition to simply passing one masked sentence, we explore an approach based on passing the masked sentence concatenated with the original sentence, which is a natural way to encode not only the context surrounding the word, but also the meaning of the target word itself. This strategy was successfully used for the task of unsupervised lexical simplification (Qiang et al., 2019). For the second point above, the works of Espinosa-Anke et al. (2019); Shwartz and Dagan (2019) already point to the fact that light verb constructions (LVCs) are easy to recognize in text. Further evidence is provided in Table 1, which shows BERT's top predictions for three sentences containing Oper1, Real1 and Magn collocations. It is immediately obvious that the nature of the LF itself, as well as the amount of the information provided by the sentential context, are crucial. Note that, in the case of Oper1, by providing the original sentence as context, BERT's grasp of the LF improves, as it tends to predict the correct collocate, whereas for Real1 or Magn, this improvement results in assigning higher probability to tokens which are more similar to the LF's abstract meaning (e.g., *trigger the landing gear* for Real1 or *tough bargain* for Magn).

With the above considerations in mind, we run BERT's MLM head over our GigaWord-based test set, and compute *Mean Reciprocal Rank* (MRR) and *Mean Average Precision* (MAP) over each LF. Recall that we consider as valid *hits* all the collocates for a given base and its corresponding LF. In practice, this means that for bases that have just one valid collocate, both metrics yield the same score. We lemmatize BERT's predictions using SpaCy's lemmatizer.[5]

4.2 Results and Discussion

Our results (Table 4) show, first, that conditioning BERT's MLM by passing the original sentence as additional context for the [MASK] token is useful for predicting an embedding whose semantics is more related to the original collocate. The improvements are particularly relevant for LVCs (Oper1, and, to a certain extent, Real1 and Real2), suggesting that these LFs, while perhaps easy to distinguish from others (cf. Section 5.1), they do benefit from additional contexts to be well represented. Interestingly, the Magn LF has small gains in both MRR and MAP, clearly showing that additional context helps little, and thus highlighting a strong semantic dependency between sentence meaning and the collocation's base.

A potential limitation of this setup, however, is that we cannot possibly include all possible collocates for all the bases in our resource. An estimate of the quality of BERT's predictions can be obtained by measuring the semantic similarity (for instance, by cosine distance) between the original masked collocate and the predicted collocates. In the example we already referred to above, *heavy rain*, the similarity between 'the' and 'heavy' is low, whereas, if the model predicts *hard* or even any other adjective, it should be considered less wrong. We obtain a broad picture of the quality of BERT's predictions by plotting a histogram (Figure 1) of the similarities obtained by comparing the original collocate's and BERT's predicted GloVe embeddings (Pennington et al., 2014) under both settings (MASKED and CONDITIONED) for the same three LFs as in Table 1), namely Magn, Oper1 and Real1. The conditioning strategy is helpful; it contributes not only to retrieving the original collocate (which would be trivial if we do not mask it), but also candidates with clearly similar meanings. We see, for instance, more cases for Oper1 and Real1, where the correct verb is predicted, whereas for Magn we see a more sustained improvement across all similarities, but not necessarily for retrieving the original collocate.

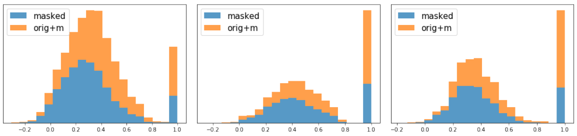


Figure 1: Histograms showing the distribution of similarities between gold and predicted (lemmatized) collocates for the three LFs Magn, Oper1 and Real1 (from left to right).

	MASKED		CONDITIONED	
	MRR	MAP	MRR	MAP
AntiBon	13.78	13.57	61.64	57.37
AntiMagn	30.16	27.11	82.18	66.89
AntiReal2	39.14	36.32	70.58	62.13
AntiVer	11.38	10.68	40.5	37.12
Bon	25.13	24.87	62.49	59.34
Caus1Func0	52.77	45.93	95.94	80.36
CausFunc0	61.14	53.53	86.2	73.52
IncepOper1	54.38	48.36	88.52	74.61
IncepPredPlus	7.27	6.375	12.27	10.67
LiquFunc0	45.51	43.12	71.14	63.60
Magn	33.43	31.67	74.72	66.92
Oper1	73.12	62.58	95.22	81.15
Oper2	63.84	53.86	93.63	76.86
Real1	59.87	53.31	90.96	75.02
Real2	55.36	46.87	76.60	64.64
Ver	33.06	29.37	72.02	63.18

Table 4: Results of the collocate retrieval experiment when passing to the MLM the masked sentence (MASKED) or with the original sentence as context (CONDITIONED).

5 Experiment 2: Collocation categorization

In the second experiment, we test the performance of a number of well-known LMs for the task of LF categorization using the train/test splits we sampled and annotated from GigaWord (Section 3). This experiment serves two purposes. First, we expect to learn about the *predictability* of LFs in context,

---

[5] `https://spacy.io/api/lemmatizer`.

which is a long-standing problem in computational lexicography and the cornerstone of automatic construction of collocation resources. Second, previous work has shown that some LFs are quite easy to distinguish, without (Espinosa-Anke et al., 2019) and with sentential context (Shwartz and Dagan, 2019). However, it is still unclear whether, by focusing exclusively on the phenomenon of collocations, and excluding, e.g., idiomatic expressions or non-compositional phrasal verbs (which are not only semantically but, more importantly, syntactically different from collocations), an LM can indeed be used to construct a resource for second language learners, or whether (and to what extent) an LM can be trained to select appropriate collocates. Our setting is essentially a sentence-pair classification problem, where the second sentence is the lexical collocation itself. Specifically, a training instance is a tuple <sentence, collocation, label>, as in Example (3) (where we use **;** as a wildcard concatenation token, as these are different across LMs):

(3) *The US military **launched an airstrike** on what it described as a safehouse in the Iraqi town of Fallujah **;** launched an airstrike* → ***Oper1***

We use as labels the LFs listed in Table 2, with their respective training/test splits, and train all LMs with the same hyperparameters.[6] The considered LMs are BERT (base and large, uncased) (Devlin et al., 2018), RoBERTa (base and large) (Liu et al., 2019), DistilBERT (Sanh et al., 2019), ALBERT (Sanh et al., 2020) and XLNet (base and large) (Yang et al., 2019). We use the implementation in the Transformers Python library (Wolf et al., 2020).[7]

5.1 Results and discussion

The results of this experiment (cf. Table 5) clearly highlight what has already pointed out in Shwartz and Dagan (2019): the prototypical LVCs (as modeled by Oper1) can be identified with a rather high quality. Interesting enough, this is not true for LVCs captured by Oper2, whose only difference to Oper1 is the subcategorization frame: while in Oper1, it is the 1st argument of the base that is realized as the grammatical subject, in Oper2, it is the 2nd argument. Frequency cannot explain this discrepancy since, e.g., IncepOper1, which appears in our corpus in nearly the same number of sentences as Oper2, is categorized with a significantly higher quality.

Results are also lower for some other verbal LFs with more semantic load, among them, e.g., Real1/2 and Caus1Func0, suggesting that the semantics expressed in the notions of 'realize' and 'cause', especially when the 2nd argument of the collocation functions as a subject, are more challenging. Again, these results cannot be fully explained by the amount of training data and neither by the semantic load. Thus, the categorization of IncepPredPlus achieves the highest score (the best model on IncepPredPlus obtains an average F1 of 95.21 with little variability across runs), and it is clearly an LF with a semantic load, namely 'increase'. Interestingly, Ver ('genuine') and Bon ('positive') are the worst categorized LFs in our sample, while their antonyms AntiVer and AntiBon are categorized considerably better.

As for the considered LMs, the best overall performing model is the RoBERTa family, with an overall F1 score of **71.19%** for RoBERTa-base and 70.6% for RoBERTa-large, and both models accounting for the best results on 7 of the 16 target LFs. The second best results are achieved by XLNet (base and large), with XLNet-large being the best model on both Magn and AntiMagn, two LFs which have been traditionally challenging to tell apart due to the fact that the representations of antonyms are clustered together in distributional spaces. We also note that, interestingly, DistilBERT is the best at categorizing Oper1 and Real2, which may suggest that small models may be sufficient to obtain good performnace on categorizing LVCs.

In order to gain further insights on *why* a LM may err in the task of in-context collocation categorization, we display a confusion matrix obtained from random runs for the two LMs with the highest avg score for Oper1 (Distilbert) and Oper2 (XLNet-base) (Figure 2) – the two LVC LFs that differ only in terms of their subcategorization patterns (cf. above). We may hypothesize that the categorization of a collocation based mainly on its actantial structure is challenging, and indeed, we observe that for these two models,[8] syntax-based categorization over the same semantics proves hard. Specifically, XLNet-base has as the greatest source for confusion regarding Oper1, precisely, Oper2;

---

[6] All models are trained for 1 epoch, with a learning rate of $4e-05$, the Adam with weight decay optimizer and a warmup ratio of 0.06.

[7] `https://huggingface.co/transformers/`.

[8] As a matter of fact, it can be assumed that this applies to all the LMs evaluated in this paper.

and this also occurs with Real1 vs. Real2 (which also differ only with respect to their subcategorization pattern). The results for DistilBERT show a greater spread among the misclassifications of Oper1, namely across Oper2, Caus1Func0 and IncepOper1, and for Oper2 across Oper1 and CausFunc0. Caus1Func0 and IncepOper1 have the same subcategorization pattern as Oper1 (but different semantics). In the case of CausFunc0 (‘cause the existence’, e.g., CausFunc0(*hope*) = *raise*), the subcategorization pattern is very similar to Caus1Func0, only that the grammatical subject of the corresponding syntactic construction is not an argument of the base. As we can observe, CausFunc0 is easily miscategorized as a full LVC. Finally, let us highlight the fact that while Magn is generally well categorized, the few misclassifications come, as would be expected, from collocations which convey a similar notion of amplification (e.g., Bon), but interestingly, also collocations that convey opposite semantics, such as AntiMagn or AntiBon.

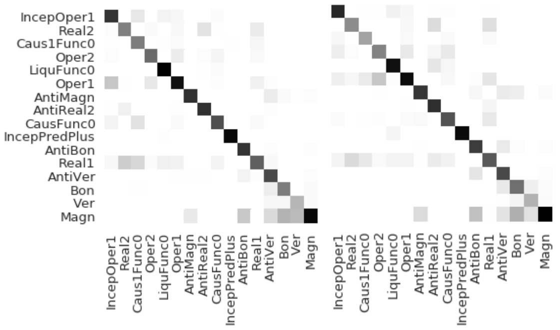


Figure 2: Confusion matrix for the two best performing models on average on Oper2 (Xlnet-base, left) and Oper1 (DistilBERT, right).

6 Subspace Analysis

In this section, we further explore the semantics of some selected LFs. We generate visualizations of PCA-projected BERT vectors for all collocation mentions of Magn, AntiMagn, Oper1 and Oper2. These four LFs are sufficiently frequent, and they encode different morphosyntactic structures.[9] We can see that antonymy (Ono et al., 2015; Schwartz et al., 2015; Nguyen et al., 2016) is relatively well captured in contextualized models, although the subspaces are clearly different between the embedding and the last transformer layer. More specifically, as the representations of collocates for Magn and AntiMagn undergo the self-attention-based transformations through BERT’s layers, many of these contextualized embeddings tend to group in a narrow cone, with many antonymic collocates indistinguishably overlapping with each other. Similarly, we also observe a tendency of representation overlap in the Oper1 vs Oper2 case, with the embeddings in the last transformer layer showing a cluttered distribution, suggesting that there is little inherent knowledge in BERT to categorize a collocation into the syntactic typification of a LF.

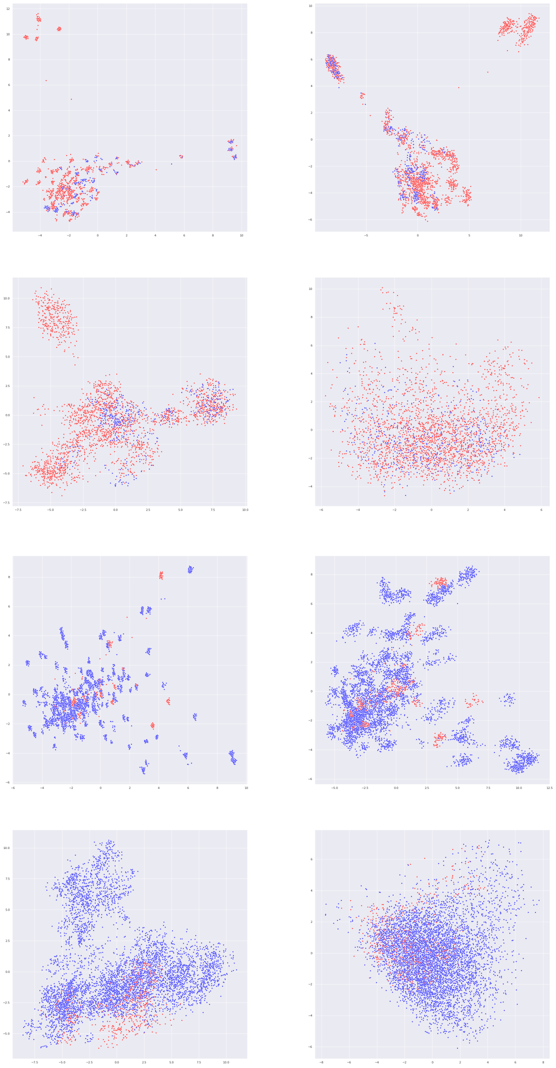


Figure 3: Oper1 (red) and Oper2 (blue) collocate embeddings for BERT’s embedding layer (top row, left), and for the 1st (top row, right), and 5th and 12th transformer layers (second row, left and right, respectively). The bottom quadrant corresponds to Magn (blue) vs AntiMagn (red), with the same arrangements (embedding, 1st, 5th and 12th layer).

---

[9] <Magn,AntiMagn> are most frequently expressed by *adj+noun* combinations, whereas <Oper1,Oper2> are always realized by a *verb+noun* pattern.

	BERT-base			BERT-large			Albert			DistilBERT		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
AntiBon	67.92±5.99	72.01±4.03	69.85±4.67	71.73±11.50	72.43±3.03	71.81±6.29	77.00±3.90	75.31±1.66	76.11±2.21	69.89±5.13	70.90±4.55	70.35±4.42
AntiMagn	85.59±3.30	78.22±4.83	81.60±1.14	85.05±2.93	74.93±3.54	79.66±3.26	84.55±1.90	74.83±0.69	79.38±0.58	85.29±3.56	75.55±2.09	80.10±2.38
AntiReal2	81.67±1.48	78.50±2.29	80.02±0.56	84.74±3.38	78.50±3.50	81.40±0.77	81.39±12.70	82.50±4.33	81.58±6.90	71.23±2.33	79.16±1.89	74.98±2.02
AntiVer	61.32±4.03	72.51±4.82	66.28±1.91	61.21±2.44	71.76±6.26	66.01±3.67	62.35±3.00	79.25±1.28	69.78±2.37	63.33±2.20	73.33±10.40	67.63±3.18
Bon	50.95±4.34	43.06±3.96	46.54±2.64	53.52±9.98	56.71±4.09	54.91±6.98	55.23±7.97	49.52±8.27	52.19±8.06	54.52±13.50	51.64±10.80	**53.02±12.10**
Caus1Func0	67.18±5.53	46.97±15.10	**53.77±10.10**	51.49±7.12	41.83±9.77	45.47±5.95	59.19±3.67	43.62±7.56	50.13±6.25	64.21±10.60	42.05±8.06	50.51±8.25
CausFunc0	72.37±10.20	59.30±7.90	64.41±2.97	71.10±7.40	58.21±6.02	63.75±4.08	78.63±7.92	60.39±5.61	68.07±3.99	74.12±2.79	65.57±1.59	69.56±1.52
IncepOper1	79.30±3.50	76.43±1.95	77.82±2.28	78.44±7.49	75.13±5.46	76.40±1.19	81.60±4.03	76.87±1.31	79.14±2.53	79.61±1.22	77.55±4.01	78.53±2.39
IncepPredPlus	97.21±3.03	91.06±0.41	94.02±1.63	93.85±6.16	90.57±1.91	92.08±2.32	97.73±2.21	85.99±6.69	91.43±4.68	98.46±2.01	90.57±0.72	94.34±0.68
LiquFunc0	88.02±6.34	91.02±2.36	89.43±3.96	88.35±3.32	87.54±2.75	87.94±2.94	88.70±2.86	91.80±5.57	90.10±1.22	85.72±2.55	87.76±1.40	86.70±0.93
Magn	92.47±0.75	93.91±0.77	93.18±0.32	92.87±0.80	93.67±2.23	93.26±1.50	92.74±0.52	93.78±0.93	93.26±0.62	92.80±1.16	94.58±0.32	93.68±0.74
Oper1	78.89±1.08	91.69±0.92	84.80±0.27	78.01±1.94	90.26±1.54	83.69±1.76	78.69±0.36	92.77±2.82	85.14±1.18	79.58±1.50	91.89±1.94	**85.28±1.27**
Oper2	70.16±2.87	48.07±6.46	56.79±3.97	66.19±1.82	52.17±5.14	58.24±3.04	71.03±9.65	51.98±2.25	59.85±3.83	67.82±4.49	48.01±2.22	56.17±2.34
Real1	70.03±2.29	59.71±4.38	64.44±3.48	69.91±2.31	59.21±2.24	64.12±2.24	70.52±4.10	62.18±1.63	66.02±1.60	70.81±8.46	62.44±4.33	66.25±5.57
Real2	58.36±2.86	53.41±7.12	55.48±3.12	64.99±5.87	54.79±5.44	59.13±1.98	63.52±2.53	47.47±8.75	54.14±6.64	69.69±0.79	54.30±12.80	**60.51±8.24**
Ver	40.15±7.96	24.31±3.82	30.23±4.92	37.59±12.30	22.27±2.90	27.38±4.40	31.54±16.10	22.61±11.60	25.95±13.40	43.15±25.20	20.91±9.93	28.07±14.50
Average	72.60±4.10	67.51±4.45	69.29±3.00	71.82±5.42	67.50±4.11	69.08±3.22	73.40±5.21	68.18±4.43	70.14±4.13	73.14±5.47	67.89±4.82	69.73±4.41

	XLNet-base			XLNet-large			RoBERTa-base			RoBERTa-large		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
AntiBon	67.74±9.42	74.72±2.49	70.88±6.14	70.30±6.48	76.75±3.99	73.25±4.11	71.57±11.90	75.82±9.03	73.51±10.20	77.44±10.10	76.42±10.10	**76.82±10.00**
AntiMagn	85.26±5.43	76.52±2.05	80.62±3.36	86.18±5.10	77.86±4.67	**81.78±4.62**	90.02±4.44	77.19±3.90	83.10±3.90	84.52±9.63	75.86±9.63	79.84±5.16
AntiReal2	81.96±3.72	77.83±2.92	79.83±3.16	85.06±10.40	78.83±1.60	**81.65±5.21**	81.72±12.30	80.00±2.78	80.61±6.89	80.64±3.95	82.66±3.95	81.64±4.08
AntiVer	65.29±2.14	67.57±1.99	66.37±0.36	67.60±3.54	70.20±6.34	68.86±4.91	69.37±1.08	75.39±9.19	**72.11±4.89**	69.93±9.95	72.92±9.95	71.18±5.85
Bon	51.70±6.46	47.40±2.16	49.41±4.10	52.08±15.90	47.51±3.42	49.24±8.88	53.95±7.90	46.87±2.06	50.01±4.13	50.84±7.58	54.70±7.58	52.70±7.85
Caus1Func0	64.71±9.52	44.29±15.40	50.52±9.85	64.46±24.90	40.93±11.70	47.48±5.79	68.68±12.20	47.87±22.40	52.47±15.40	59.92±11.70	42.05±11.70	49.41±10.20
CausFunc0	75.65±9.76	59.58±7.49	66.35±6.80	79.15±14.70	60.39±2.38	68.17±5.83	78.06±6.82	67.21±2.45	**72.15±3.85**	75.56±8.38	63.30±8.38	68.81±6.47
IncepOper1	73.37±3.87	78.16±1.08	75.65±1.94	79.35±4.77	75.20±2.35	77.13±1.86	79.38±4.43	79.03±1.76	79.16±2.54	83.29±8.40	76.49±8.40	**79.61±4.24**
IncepPredPlus	99.21±0.01	91.54±2.32	**95.21±1.26**	97.89±2.00	89.37±2.21	93.43±1.88	96.31±4.49	90.33±0.83	93.18±1.71	92.79±12.40	91.06±12.40	91.58±6.04
LiquFunc0	87.38±2.63	92.48±4.18	89.83±2.63	87.95±4.60	91.91±6.18	89.68±1.09	88.75±0.35	93.04±4.01	90.82±1.89	90.15±2.41	91.58±2.41	**90.86±2.81**
Magn	91.93±0.85	93.91±0.48	92.91±0.54	93.41±1.66	94.24±1.66	**93.82±1.62**	92.57±1.31	94.95±1.60	93.74±1.43	92.90±1.29	94.73±1.29	93.81±1.68
Oper1	77.80±0.97	91.09±0.20	83.92±0.48	78.90±0.94	91.80±0.78	84.86±0.22	78.33±2.36	91.18±1.78	84.25±1.46	78.41±1.85	92.40±1.85	84.81±1.79
Oper2	72.13±8.13	52.69±3.66	**60.85±5.12**	74.05±4.18	50.70±7.88	59.86±4.83	72.05±1.20	50.70±8.57	59.18±5.79	70.19±5.83	52.56±5.83	59.97±1.06
Real1	72.00±1.43	59.84±0.99	65.36±0.96	66.09±3.48	64.60±4.19	65.33±3.82	71.11±4.52	61.05±4.51	65.66±4.10	72.59±4.37	62.49±4.37	**66.90±5.70**
Real2	66.37±1.17	51.23±4.45	57.77±3.13	64.01±12.30	54.20±7.40	58.67±9.52	66.12±3.52	52.91±8.66	58.67±6.65	64.58±7.64	51.03±7.64	56.92±9.77
Ver	32.58±12.90	15.98±10.50	21.37±12.90	32.94±19.20	26.70±22.00	29.21±21.10	48.10±29.10	22.78±5.97	**30.38±11.30**	41.46±4.40	18.02±4.40	24.77±3.23
Average	72.82±4.90	67.18±3.90	69.18±3.92	73.71±8.38	68.20±5.55	70.15±5.33	75.38±6.75	69.15±5.59	**71.19**±5.38	74.08±6.87	68.64±6.87	70.60±5.37

Table 5: Average Precision, Recall and F1 results for the collocation classification experiment, computed by averaging the results of three independent runs. We also report standard deviation figures. Results are provided per LF as well as the average over each metric (Average).

7 Conclusions

We have analyzed LMs in tasks revolving around modeling, recognizing and categorizing lexical collocations. We conclude that some prominet types of LVCs require little context to be well encoded, as opposed to other LFs involving, e.g., nouns and adjectives, and that predictability of LFs is challenging, not a function of training data, and that syntax plays a major role.

8 Future Work

In the future, we will make this work multilingual using linguistic equivalences as anchors, in the spirit of cross-lingual embedding research, in order to align collocations of the same LF across languages (e.g., in English and Norwegian we *take a nap*, in German, we 'make' it, in Portuguese we 'pull' it, in Spanish, we 'throw' it, etc.). We would also like to explore the idea of "semantic masking" for collocate discovery, where we would train models for dynamically *masking* (or removing) idiosyncratic information such that only the semantics of the collocate remain, thus largely corresponding to a latent abstraction over the LF. This approach has been applied recently in the lexical substitution task, with the limitation, however, that the dropout rate was tuned in a validation set, whereas a promising avenue to explore would be to automatically learn the embedding dropout in a fully supervised setting. Finally, motivated by the observed large gap in performance between the categorization of, e.g., Oper1 and Oper2, Bon and AntiBon, Ver and AntiVer, we plan to investigate in more depth the codification of collocational information in pretrained LMs.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. This work was partially supported by the European Commission via its H2020 Program under the contract number 870930. The experiments were executed using the computational facilities of the Advanced Research Computing @Cardiff (ARCCA) at Cardiff University.

References

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1183–1193.

Gemma Boleda, Marco Baroni, Louise McNally, et al. 2013. Intensionality was only alleged: On adjective-noun composition in distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013): long papers; 2013 Mar 20-22; Postdam, Germany. Stroudsburg (USA): Association for Computational Linguistics (ACL); 2013. p. 35-46.* ACL (Association for Computational Linguistics).

Jose Camacho-Collados, Luis Espinosa-Anke, Shoaib Jameel, and Steven Schockaert. 2019. A latent variable model for learning distributional relation vectors. In *International Joint Conferences on Artificial Intelligence*.

Kenneth W. Church and Patrick Hanks. 1989. Word Association Norms, Mutual Information, and Lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 76–83, Vancouver, Canada.

Anthony P. Cowie. 1994. Phraseology. In R.E. Asher and J.M.Y. Simpson, editors, *The Encyclopedia of Language and Linguistics, Vol. 6*, pages 3168–3171. Pergamon, Oxford.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Daniel Edmiston. 2020. A systematic analysis of morphological content in bert models for multiple languages. *arXiv preprint arXiv:2004.03032v1*.

Luis Espinosa-Anke, Jose Camacho-Collados, Sara Rodríguez Fernández, Horacio Saggion, and Leo Wanner. 2016. Extending wordnet with fine-grained collocational information via supervised distributional learning. In *Proceedings of COLING 2016: Technical Papers. The 26th International Conference on Computational Linguistics; 2016 Dec. 11-16; Osaka (Japan): COLING; 2016. p. 900-10.* COLING.

Luis Espinosa-Anke and Steven Schockaert. 2018. Seven: Augmenting word embeddings with unsupervised relation vectors. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2653–2665.

Luis Espinosa-Anke, Steven Schockaert, and Leo Wanner. 2019. Collocation classification with unsupervised relation vectors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5765–5772.

Stefan Evert. 2007. Corpora and Collocations. In A. Lüdeling and M. Kytö, editors, *Corpus Linguistics. An International Handbook*. Mouton de Gruyter, Berlin.

John R. Firth. 1957. Modes of Meaning. In J.R. Firth, editor, *Papers in Linguistics, 1934-1951*, pages 190–215. Oxford University Press, Oxford.

Marcos Garcia, Marcos García Salido, and Margarita Alonso Ramos. 2017. Using bilingual word-embeddings for multilingual collocation extraction. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 21–30.

A. Gelbukh and O. Kolesnikova. 2012. *Semantic Analysis of Verbal Collocations with Lexical Functions*. Springer, Heidelberg.

Franz Josef Hausmann. 1985. Kollokationen im Deutschen Woerterbuch: ein Beitrag zur Theorie des lexicographischen Biespiels. *Lexikographie und Grammatik*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Mandar Joshi, Eunsol Choi, Omer Levy, Daniel S Weld, and Luke Zettlemoyer. 2019. pair2vec: Compositional word-pair embeddings for cross-sentence inference. In *NAACL-HLT (1)*.

Mandar Joshi, Kenton Lee, Yi Luan, and Kristina Toutanova. 2020. Contextualized representations using textual encyclopedic knowledge. *arXiv preprint arXiv:2004.12006*.

Adam Kilgarriff. 2006. Collocationality (And How to Measure it). In *Proceedings of the 12th Euralex International Congress on Lexicography (EURALEX)*, pages 997–1004, Turin, Italy. Springer-Verlag.

Brigitte Krenn. 2000. *The Usual Suspects: Data-Oriented Models for the Identification and Representation of Lexical Collocations*, volume 7 of *Saarbrücken Dissertations in Computational Linguistics and Language Technology*. DFKI and Universität des Saarlandes.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations.

Omer Levy, Steffen Remus, Chris Biemann, Ido Dagan, and Israel Ramat-Gan. 2015. Do supervised distributional methods really learn lexical inference relations? In *NAACL 2015*, Denver, Colorado, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marco Maru, Federico Scozzafava, Federico Martelli, and Roberto Navigli. 2019. Syntagnet: challenging supervised word sense disambiguation with lexical-semantic combinations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3525–3531.

I.A. Mel'čuk. 1996. Lexical functions: A tool for the description of lexical relations in the lexicon. In Leo Wanner, editor, *Lexical Functions in Lexicography and Natural Language Processing*, pages 37–102. Benjamins Academic Publishers, Amsterdam.

Igor Mel'čuk. 1995. Phrasemes in Language and Phraseology in Linguistics. In M. Everaert, E.-J. van der Linden, A. Schenk, and R. Schreuder, editors, *Idioms: Structural and Psychological Perspectives*, pages 167–232. Lawrence Erlbaum Associates, Hillsdale.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 454–459.

Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word embedding-based antonym detection using thesauri and distributional information. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 984–989.

Pavel Pecina. 2008. A Machine Learning Approach to Multiword Expression Extraction. In *Proceedings of the LREC 2008 Workshop Towards a Shared Task for Multiword Expressions)*, pages 54–57, Marrakech, Morocco.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Jipeng Qiang, Yun Li, Yi Zhu, and Yunhao Yuan. 2019. A simple bert-based approach for lexical simplification. *arXiv preprint arXiv:1907.06226*.

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

Carlos Ramisch, Silvio Cordeiro, Agata Savary, Veronika Vincze, Verginica Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, et al. 2018. Edition 1.1 of the parseme shared task on automatic identification of verbal multiword expressions.

Sara Rodríguez Fernández, Luis Espinosa-Anke, Roberto Carlini, and Leo Wanner. 2016. Semantics-driven recognition of collocations using word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics; 2016 Aug. 7-12; Berlin (Germany).[place unknown]: ACL; 2016. Vol. 2, Short Papers; p. 499-505.* ACL (Association for Computational Linguistics).

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

V. Sanh, L. Debut, J. Chaumond, and Th. Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing*, Vancouver, BC, Canada.

V. Sanh, L. Debut, J. Chaumond, and Th. Wolf. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of ICLR*.

Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of the nineteenth conference on computational natural language learning*, pages 258–267.

V. Seretan. 2014. On collocations and their interaction with parsing and translation. *Informatics*, 1(1):11–31.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of ICLR*.

Leo Wanner and John A. Bateman. 1990. A collocational based approach to salience sensitive lexical selection. In *Proceedings of the 5th International Workshop on Natural Language Generation*, Dawson, PA.

Leo Wanner, Bernd Bohnet, and Mark Giereth. 2006. Making sense of collocations. *Computer Speech and Language*, 20(4):609–624.

Koki Washio and Tsuneaki Kato. 2018. Neural latent relational analysis to capture lexical semantic relations in a vector space. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 594–600.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q.V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, BC, Canada.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. Bert-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373.

BART-TL: Weakly-Supervised Topic Label Generation

Cristian Popa
Universitatea Politehnica of Bucharest
cristian.viorel.popa@gmail.com

Traian Rebedea
Universitatea Politehnica of Bucharest
traian.rebedea@upb.ro

Abstract

We propose a novel solution for assigning labels to topic models by using multiple weak labelers. The method leverages generative transformers to learn accurate representations of the most important topic terms and candidate labels. This is achieved by fine-tuning pre-trained BART models on a large number of potential labels generated by state of the art non-neural models for topic labeling, enriched with different techniques. The proposed *BART-TL* model is able to generate valuable and novel labels in a weakly-supervised manner and can be improved by adding other weak labelers or distant supervision on similar tasks.

1 Introduction

As topic modeling has been used for unsupervised exploration of large text corpora, several topic labeling approaches have been proposed. These range from heuristic-based methods (Mei et al., 2007; Gourru et al., 2018) that focus on the underlying topic distributions to newer methods that use word embeddings (Bhatia et al., 2016). Supervised topic labeling methods (Lau et al., 2011; Bhatia et al., 2016) typically use annotator data with the quality of the labels to train a more accurate ranker than the unsupervised counterpart. Deep learning approaches, which gained quick popularity in NLP, are starting to be used for solving this task as well (Sorodoc et al., 2017; Alokaili et al., 2020).

Recently, transformer models pre-trained on very large amounts of data achieved impressive results on a lot of downstream NLP tasks using fewer resources than previously necessary. We introduce a method of performing a weakly-supervised fine-tuning on these models pre-trained on English data in order to obtain human-comprehensible and meaningful topic labels. We also provide a quality evaluation of the model-generated labels, in addition to an analysis of the contribution gained from using this approach that we ultimately refer to as *BART-TL*, inspired by the name of the original transformer architecture.

2 Related Work

Topic modeling is a popular unsupervised method for exploring large corpora of documents. Topics are represented as distributions over words, while documents as mixtures of topics. Historically, these methods used dimensionality reduction techniques (Deerwester et al., 1990), then migrated to probabilistic-based methods (Hofmann, 1999), with Latent Dirichlet Allocation (Blei et al., 2003) gaining popularity. LDA makes use of variational inference to obtain the distribution matrices. Further developments include hierarchical (Wang et al., 2011) and online (Hoffman et al., 2010) versions of LDA.

While the resulting distributions of topic models are useful for computational purposes, such as measuring the similarity of two documents, these may prove difficult to interpret by humans. Topic labeling aims to solve this issue by computing labels for each topic. Historically, this was achieved by establishing a pool of labels and ranking them using certain scoring functions. First attempts were fully unsupervised, extracting labels from the original corpus (Mei et al., 2007). Later approaches started using external corpora, such as Wikipedia, as candidates for labels and trained supervised rankers (Lau et al., 2011), as well as employed word embeddings (Bhatia et al., 2016) such as *word2vec* (Mikolov et al., 2013) and *doc2vec* (Le and Mikolov, 2014) for computing the similarity between a topic and a candidate label.

Huge progress was made in the NLP field with the introduction of attention models (Bahdanau et al., 2014) and, later on, transformers (Vaswani et al., 2017), which are deep neural networks that

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1418–1425
April 19 - 23, 2021.

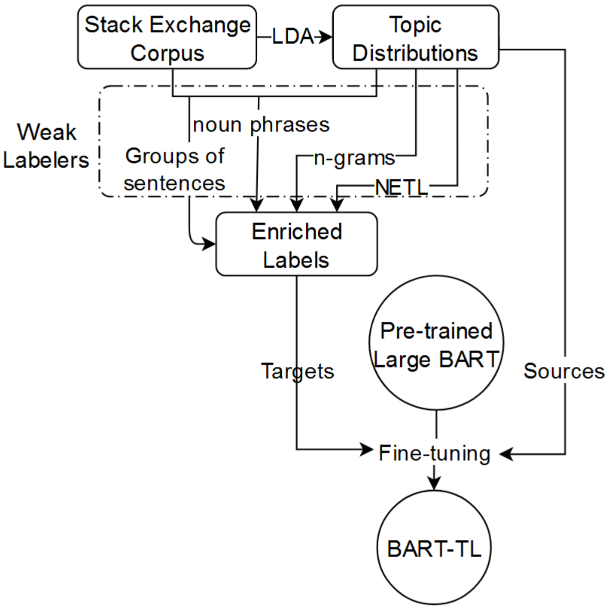


Figure 1: End-to-end training of *BART-TL* for topic labeling using weak supervision.

use an encoder-decoder architecture. A multitude of transformer-based models (Devlin et al., 2018; Radford et al., 2019; Liu et al., 2019; Lewis et al., 2019) emerged that achieved state of the art performance on a large number of NLP tasks through transfer learning. These models are pre-trained on large amounts of data in order to encompass general knowledge of the language to be later fine-tuned on downstream tasks. This allows for better results on small datasets, where deep learning was not a viable option beforehand.

However, research on using deep learning methods for topic labeling is scarce. A very recent study proposes an RNN-based encoder-decoder architecture (Alokaili et al., 2020) trained with distant supervision using Wikipedia page titles and employing BERTScore (Zhang et al., 2019) for evaluation.

3 Method

Our method utilizes a pre-trained BART (Lewis et al., 2019) transformer model, with a denoising autoencoder architecture, as we adopt a sequence-to-sequence approach for the task of topic labeling.

3.1 Building a Weakly Supervised Dataset

Topic labeling is generally performed in two steps: establishing a pool of candidate labels and then ranking them appropriately. This workflow is also adopted by a state of the art labeler that we will refer to as *NETL* (Bhatia et al., 2016).[1] This method uses names of Wikipedia articles as candidate labels and trains word2vec and doc2vec models on Wikipedia dumps. Preliminary filtering is done by selecting the labels with the highest embedding similarity scores to the topic terms, while the remaining labels are ranked in an unsupervised manner using letter trigrams. The authors also explore training a supervised ranker after obtaining feedback from annotators, incorporating PageRank (Page et al., 1999) and lexical features.

We build a dataset for fine-tuning BART starting from the **NETL labeler**. We extract the initial candidate labels for each topic after the embeddings similarity filtering but modify this process by assigning a greater weight in the scoring based on the importance of the word in the topic distribution. To avoid overfitting the most important word, we equalize the weights of the top-5 terms. The labels that consist only of stopwords are removed. We make these changes to be able to use a larger number of highest-rated topic terms in extracting labels than the standard 10 employed by NETL, expecting a better performance given a more ample context. Finally, we construct a one-to-many sequence mapping from topics, represented as a concatenation of the top-20 terms separated by spaces, to the corresponding labels. This represents the *baseline dataset*.

We also propose adding several enrichment approaches for this dataset, using other weak labelers as follows. The first additions are entries consisting of space-separated **n-grams** sampled from the most important words in the topic. The sampling is weighted by the underlying probability distribution and these do not have to be consecutive. Inspired by the work of Gourru et al. (2018), **groups of sentences** are added as targets using a variant of the COS10 technique for sentence extraction. The best sentences are joined one-by-one into a short paragraph until a minimum character threshold is met. One last idea for improving the baseline dataset is including popular **noun phrases** from the corpus. They are ranked based on the relevance to the topic and must appear at least a certain number of times in the corpus.

---

[1] The code is open-source: `https://github.com/sb1992/NETL-Automatic-Topic-Labelling-`.

3.2 Fine-tuning BART-TL

Pre-trained BART models are fine-tuned on the resulting datasets. The final *BART-TL* models are able to make predictions on sequences of topic terms. Output labels are generated as sequences and beam search is used to extract multiple ranked labels for a single topic. This strategy joins the extensive knowledge about language encompassed in the original transformer layers with traditional topic labeling techniques. The final models are fine-tuned based on unsupervised labelers and are, thus, weakly-supervised. A detailed representation of the end-to-end process can be seen in Figure 1.

4 Experiments

4.1 Baseline Dataset

We conduct experiments on corpora crawled from Stack Exchange[2] on 5 different subjects: English, Biology, Economics, Law, and Photography. These are preprocessed by removing XML artifacts, stopwords, and individual numbers. Documents with fewer than 20 words are removed from the corpus, along with words that occur less than 10 and more than 50,000 times. A total of 419,189 documents remain in the corpus. We apply LDA (Blei et al., 2003) on each corpus and obtain 100 topics for each subject. This choice for the number of topics is based on the prior work of Bhatia et al. (2016) where the authors generate 100 topics for each domain. These are filtered based on coherence (Röder et al., 2015), removing topics with a C_V score under 0.30, leaving a total of 303 topics. With the probability distributions of topics over the top-100 words, we generate 100 candidate labels for each topic using the NETL approach described in Section 3.

For the weak labelers, we choose to extract 5 n-grams with a n varying between 2 and 4, 5 groups of sentences with a character threshold of 120 and 10 noun phrases with a length of 2 to 4 words that have at least 25 occurrences. We experimented with each strategy individually but provided results for a model employing only the n-grams enrichment, *BART-TL-ng*, and one using all of them, *BART-TL-all*.

4.2 Fine-tuning Details

We fine-tune the large BART model[3] for 2 epochs using an Adam optimizer (Kingma and Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, 0.1 weight decay, 0.1 dropout, 0.1 attention dropout, 0.1 label smoothing, 6% warmup steps and a learning rate of 3e-5. The final labels are generated using beam search with a beam size of 25. These values follow the fine-tuning approach suggested for RoBERTa (Liu et al., 2019) and, by extension, BERT (Devlin et al., 2018), since the BART fine-tuning experiments do not explicitly specify different values for the hyper-parameters.

5 Results

We gather annotations in the form of surveys with 7 questions, one per topic, on the quality of topic labels on a scale from 0 to 3. The annotators have varying backgrounds, including computer science, medicine, law, and economics. For each of the 5 subjects in the corpus, we select 6 coherent topics for evaluation. The labels are taken from the unsupervised and supervised versions of the original NETL method, along with *BART-TL-ngram*, and *BART-TL-all*. For each method, only top-10 labels are considered for evaluation. An extra stopword label is introduced as a distractor, removing answers from annotators with over 25% of these scores ≥ 1. A topic is presented using its top-10 terms, along with 2 relevant short paragraphs, to offer additional context when the topic is unclear. Each survey has balanced topics based on the 5 subjects and each question contains 9 balanced labels based on the models. We gathered a total of 35 survey responses and filtered out the labels that had only a single annotation. This annotation was performed pro-bono and we estimate that the average time per annotated survey was 10 minutes. There is no bias in the annotations for certain models, as the average standard deviation for rating of individual labels is between 0.42 and 0.44 for all of them.

The results of this study are presented in Table 1. We focus on both the overall quality of the labels through top-k average rating, as well as how well the labels are ordered through normalized discounted cumulative gain (Järvelin and Kekäläinen, 2002). The two *BART-TL* models additionally feature statistics of the same labels reordered by the supervised and unsupervised ranking methods of

---

[2] The corpus can be found at `https://archive.org/download/stackexchange`.

[3] `https://github.com/pytorch/fairseq/tree/master/examples/bart`.

Table 1: Qualitative comparison of labels between *NETL* and *BART-TL* models.

Models	**All**						**English**					
	Top-k Avg.			**nDCG-k**			**Top-k Avg.**			**nDCG-k**		
	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5
NETL (U)	2.66	**2.59**	**2.50**	0.83	**0.85**	0.87	2.19	2.46	**2.38**	0.57	0.78	0.84
NETL (S)	**2.74**	2.57	2.49	**0.88**	**0.85**	**0.88**	2.63	2.47	2.28	0.84	0.86	0.86
BART-TL-all (U)	2.64	2.52	2.43	0.83	0.84	0.87	2.58	2.33	2.20	0.81	0.83	0.89
BART-TL-all (S)	2.64	2.55	2.42	0.81	0.84	0.87	2.58	2.36	2.15	0.81	0.86	0.89
BART-TL-ng (U)	2.62	2.50	2.33	0.82	0.84	0.85	2.58	**2.49**	2.26	0.81	**0.91**	**0.93**
BART-TL-ng (S)	2.73	2.46	2.25	0.87	0.83	0.83	**2.75**	2.40	2.21	**0.91**	0.88	0.91

Table 2: Samples of good and bad quality *new* labels generated by *BART-TL* models.

Top-10 topic terms	**Good new labels**	**Bad new labels**
crime center institution chain prison facility prisoner transformation jail custody	criminal justice system administrative court	guarantee principle
plate vehicle state license motor shall registration law apostille issued	driver's license license plate law	no matter what vehiclelicense *(no space)*
rate interest price inflation bond increase real money supply nominal	investment rate discount rate	rate interest rate principle

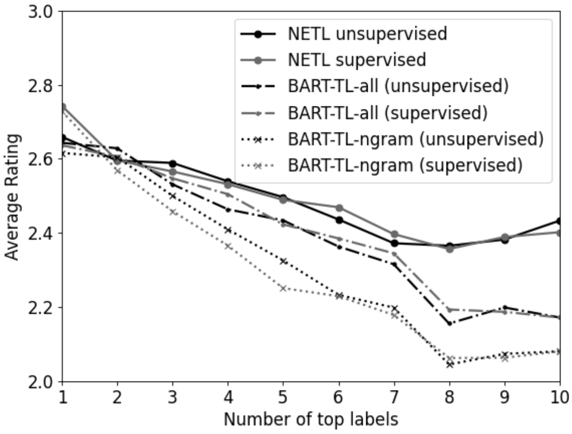


Figure 2: Evolution of average rating considering top-k labels.

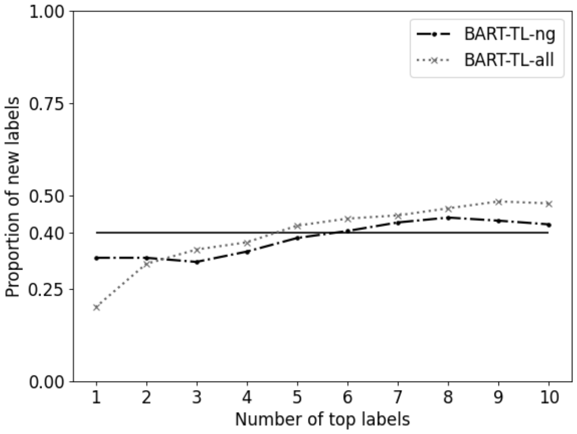


Figure 3: Average proportion of new labels in top-k.

NETL, as these usually perform better than the raw beam search results. The supervised variant of NETL uses the pre-trained ranker from the original paper. An extended version of this table is available in Appendix A.

To further investigate the results, we plot the evolution of the average rating in relation to the number of top labels considered. This can be seen in Figure 2. We study the capacity for novelty of the models in Figure 3, which outlines the proportion of new labels never encountered in the fine-tuning dataset or NETL top-10 predicted labels, as well as Figure 4, which illustrates the average rating of these labels. We observe a significant loss of up to 0.20 in rating, but even larger variations in rating are frequent in Table 1. That said, the novel labels would still be considered relevant with a

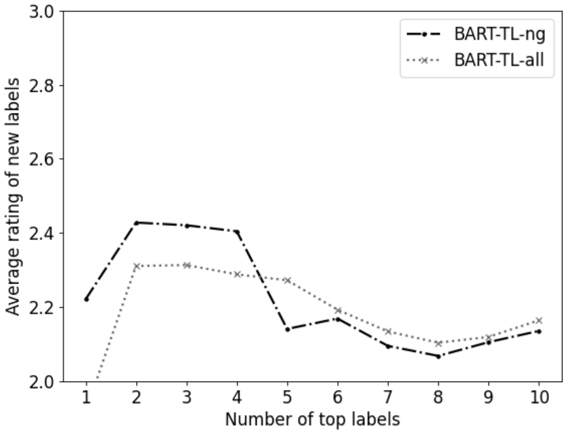


Figure 4: Average rating of new labels in top-k.

rating between 2.0 and 2.5. Table 2 showcases a few samples of original labels.

The results highlight that generative *BART-TL* models produce similar quality labels as the NETL methods when considering the top 1-2 labels. However, the quality of the generated labels degrades as their number increases. There is also no clear winner between the supervised and unsupervised versions of the proposed models, as they have similar trends. At the same time, the novelty tends to improve slightly with the number of considered labels. On average, 40% of the labels were never provided when fine-tuning the models. While novelty is an important feature for *BART-TL*, it can further be conditioned to generate labels with specific characteristics (Keskar et al., 2019).

The *BART-TL* models outperform the NETL methods on the English corpus, the largest of the five. At the same time, they achieve similar results on the Law and Biology corpora, that have the least amount of topics and are outperformed on the rest. Therefore, there was no correlation found between corpus size and the quality of the generated labels.

6 Conclusion

We introduced the *BART-TL* model that builds upon previous topic labeling solutions by adopting a generative deep learning strategy. Large pre-trained transformer models are fine-tuned in a weakly-supervised manner using unsupervised labelers to obtain meaningful labels. While current results have varying quality compared NETL, *BART-TL* is able to generate novel labels of similar quality. Although BART-TL experiments have been carried out for English, our generative methodology can be applied to any language if a pre-trained BART model is available.

Acknowledgment

This research has been partially supported by EEA Grants 2014-2021 and UEFISCDI, under project contract EEA-RO-NO-2018-0496.

References

Areej Alokaili, Nikolaos Aletras, and Mark Stevenson. 2020. Automatic generation of topic labels. *arXiv preprint arXiv:2006.00127*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. 2016. "Automatic Labelling of Topics with Neural Embeddings". In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 953–963, Osaka, Japan. The COLING 2016 Organizing Committee.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American society for information science*, 41(6):391–407.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Antoine Gourru, Julien Velcin, Mathieu Roche, Christophe Gravier, and Pascal Poncelet. 2018. United we stand: Using multiple strategies for topic labeling. In *International Conference on Applications of Natural Language to Information Systems*, pages 352–363. Springer.

Matthew Hoffman, Francis R. Bach, and David M. Blei. 2010. Online Learning for Latent Dirichlet Allocation. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 856–864. Curran Associates, Inc.

Thomas Hofmann. 1999. Probabilistic Latent Semantic Indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. 2011. "Automatic Labelling of Topic Models". In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1536–1545, Portland, Oregon, USA. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic Labeling of Multinomial Topic Models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408.

Ionut Sorodoc, Jey Han Lau, Nikolaos Aletras, and Timothy Baldwin. 2017. Multimodal topic labelling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 701–706.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Chong Wang, John Paisley, and David Blei. 2011. Online Variational Inference for the Hierarchical Dirichlet Process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 752–760, Fort Lauderdale, FL, USA. PMLR.

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

Appendix A. Extended Results

We present an extended version of the results showcased previously in Table 1. While the target metrics remain the same, the additions are the raw BART models, with the labels retaining the order that they were generated in using beam search, as well as all of the 5 different subjects. These were added in Table 3.

Table 3: Extended quality comparison of labels between *NETL* and *BART-TL* models.

Models	**All**						**English**					
	Top-k Avg.			**nDCG-k**			**Top-k Avg.**			**nDCG-k**		
	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5
NETL (U)	2.66	**2.59**	**2.50**	0.83	**0.85**	0.87	2.19	2.46	**2.38**	0.57	0.78	0.84
NETL (S)	**2.74**	2.57	2.49	**0.88**	**0.85**	**0.88**	2.63	2.47	2.28	0.84	0.86	0.86
BART-TL-all	2.41	2.38	2.28	0.73	0.75	0.79	1.89	1.94	2.00	0.52	0.60	0.74
BART-TL-all (U)	2.64	2.52	2.43	0.83	0.84	0.87	2.58	2.33	2.20	0.81	0.83	0.89
BART-TL-all (S)	2.64	2.55	2.42	0.81	0.84	0.87	2.58	2.36	2.15	0.81	0.86	0.89
BART-TL-ng	2.31	2.28	2.16	0.67	0.72	0.75	1.71	2.05	1.98	0.39	0.60	0.68
BART-TL-ng (U)	2.62	2.50	2.33	0.82	0.84	0.85	2.58	**2.49**	2.26	0.81	**0.91**	**0.93**
BART-TL-ng (S)	2.73	2.46	2.25	0.87	0.83	0.83	**2.75**	2.40	2.21	**0.91**	0.88	0.91
	Biology						**Economics**					
	Top-k Avg.			**nDCG-k**			**Top-k Avg.**			**nDCG-k**		
	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5
NETL (U)	2.57	2.26	2.18	0.86	0.80	0.83	2.83	**2.73**	2.71	0.88	**0.85**	**0.89**
NETL (S)	2.57	2.27	2.16	**0.87**	0.77	0.82	**2.89**	2.68	**2.72**	**0.92**	0.83	**0.89**
BART-TL-all	2.63	**2.51**	2.23	**0.87**	**0.89**	0.84	2.59	2.52	2.41	0.75	0.76	0.76
BART-TL-all (U)	2.42	2.43	2.37	0.75	0.82	**0.85**	2.66	2.62	2.55	0.83	0.82	0.83
BART-TL-all (S)	2.38	2.34	**2.41**	0.73	0.77	**0.85**	2.74	2.60	2.56	0.85	0.81	0.82
BART-TL-ng	**2.66**	2.42	2.09	**0.87**	0.84	0.80	2.53	2.64	2.62	0.72	0.78	0.82
BART-TL-ng (U)	2.42	2.47	2.27	0.72	0.81	0.83	2.66	2.63	2.65	0.83	0.83	0.86
BART-TL-ng (S)	2.48	2.18	2.14	0.76	0.68	0.74	2.77	2.68	2.57	0.87	**0.85**	0.83
	Law						**Photography**					
	Top-k Avg.			**nDCG-k**			**Top-k Avg.**			**nDCG-k**		
	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5	k=1	k=3	k=5
NETL (U)	2.81	**2.78**	2.52	0.88	**0.91**	0.87	**2.88**	2.61	**2.61**	**0.97**	0.89	0.92
NETL (S)	2.79	2.71	**2.59**	0.87	0.89	0.89	2.81	**2.66**	**2.61**	0.92	**0.91**	**0.93**
BART-TL-all	2.29	2.48	2.44	0.67	0.73	0.79	2.71	2.49	2.30	0.85	0.80	0.82
BART-TL-all (U)	2.86	2.67	2.57	0.91	0.86	0.88	2.67	2.59	2.46	0.82	0.85	0.88
BART-TL-all (S)	2.70	2.77	2.61	0.80	0.89	**0.91**	2.73	2.64	2.38	0.85	0.87	0.86
BART-TL-ng	2.17	2.16	1.98	0.60	0.65	0.67	2.53	2.12	2.09	0.81	0.76	0.77
BART-TL-ng (U)	**2.97**	2.39	2.29	**0.98**	0.82	0.86	2.42	2.52	2.11	0.72	0.84	0.79
BART-TL-ng (S)	2.86	2.54	2.14	0.91	0.87	0.82	2.74	2.46	2.15	0.91	0.85	0.83

Dynamic Graph Transformer for Implicit Tag Recognition

Yi-Ting Liou,[1] Chung-Chi Chen,[1] Hen-Hsen Huang,[2,3] Hsin-Hsi Chen[1,3]
[1] Department of Computer Science and Information Engineering
National Taiwan University, Taiwan
[2] Department of Computer Science, National Chengchi University, Taiwan
[3] MOST Joint Research Center for AI Technology and All Vista Healthcare, Taiwan
`ytliou, cjchen@nlg.csie.ntu.edu.tw`
`hhhuang@nccu.edu.tw, hhchen@ntu.edu.tw`

Abstract

Textual information extraction is a typical research topic in the NLP community. Several NLP tasks such as named entity recognition and relation extraction between entities have been well-studied in previous work. However, few works pay their attention to the implicit information. For example, a financial news article mentioned "Apple Inc." may be also related to Samsung, even though Samsung is not explicitly mentioned in this article. This work presents a novel dynamic graph transformer that distills the textual information and the entity relations on the fly. Experimental results confirm the effectiveness of our approach to implicit tag recognition.

1 Introduction

Documents on the web deliver and spread lots of most recent information to people worldwide. In order to automatically update the real-world information, textual information extraction is a fundamental issue for NLP researchers. Many downstream tasks such as fake news detection (Wu et al., 2019) and stock movement prediction (Peng and Jiang, 2016) can benefit from the extracted information. However, most of the previous works (Hu et al., 2018; Xu and Cohen, 2018) focus on using explicit information in articles and do not consider the implicit information. For example, two companies, Sprint and T-Mobile, are explicitly mentioned companies in the news article in Figure 1. However, the stock price of a non-mentioned company, SoftBank, may be influenced since Softbank owns shares of Sprint. Although this kind of inference is intuitive for professional analysts, few previous works take such implicit information into consideration. In this paper, we aim to increase the sense of machines toward this kind of implicit information.

Transformer-based (Vaswani et al., 2017) neural networks achieve state-of-the-art performances in many NLP tasks (Devlin et al., 2018; Malmi et al., 2019). To model the relationships between entities, graph neural network (GNN) is a well-known architecture for representing the knowledge and additional information (Fu et al., 2019; You et al., 2020). Furthermore, the models blending these two architectures show their effectiveness (Lu et al., 2020). In this paper, we propose dynamic graph transformer (DGT), a novel blend of Transformer and GNN. In previous work, the weights of the GNN are pre-determined in the training stage and not affected by the given input. Our DGT adjusts the weights depending on the input on the fly. In this way, the representation of the graph information will be more flexible and more specific to the input.

Title:
The Judge Nodded! *Sprint* Rise 70% in After-hour Trading.

News Article:
A federal judge gave his blessing to the US$26.5 billion merger between *T-Mobile* and *Sprint* on Feb. 11, several months after the deal got final antitrust approval from the U.S. government. *Sprint* surges 68.75%. *T-Mobile* rise 7.36%.

Related Stock:
Sprint, *T-Mobile*, **SoftBank**

Figure 1: An example of the implicit information in news articles.

A strategy for pre-training on in-domain data is further proposed. Experimental results show our approach is effective in the task of extracting the implicit information from news articles. The contributions of this work are summarized as follows.

- We point out an important issue of information extraction for implicit entities.
- We propose a novel model that dynamically incorporates textual information and graph in-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1426–1431
April 19 - 23, 2021.

formation. Our approach outperforms recent works in implicit tag recognition.

- Our pre-training task, masked entity prediction, is helpful for predicting the implicit entities. The pre-trained model can be also applied in other information extraction tasks.

2 Related work

Extracting and using the information in articles is one of the focuses in the NLP community. Some works (Hu et al., 2018; Ding et al., 2019; Ma et al., 2019) adopt the extracted information for stock market prediction. Some of them (Baker et al., 2016; Min and Zhao, 2019) use the information in news articles to construct socio-economic indicators. Most of previous works only focus on explicit information in the articles. In this way, the implicit entities in the articles may be under looked. In this paper, we aim to extract the non-mentioned but related entities from a document.

Recently, GNN has become popular for modeling relationships among multiple entities. Kipf and Welling (2016) use the convolution neural network to learn the node representation by aggregating the features of neighboring nodes. Veličković et al. (2017) employ the attention mechanism to improve the GNN architecture. Recent studies (Berg et al., 2017; Monti et al., 2017; Ying et al., 2018) also show the effectiveness of graph neural networks in various tasks. Inspired by these works, we present a new blend of graph attention network (GAT) (Veličković et al., 2017) with Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2018) for extracting the implicitly related entities to a given article.

3 Method

3.1 Task Setting

The task is formulated as follows. Given an article, a model is aimed at predicting a list of entities that are not explicitly mentioned but related to the given article. Let a corpus be $\mathcal{D} = \{(\mathbf{d}^1, \mathbf{y}^1), (\mathbf{d}^2, \mathbf{y}^2), ..., (\mathbf{d}^{|\mathcal{D}|}, \mathbf{y}^{|\mathcal{D}|})\}$, where $\mathbf{d}^k$ and $\mathbf{y}^k$ denote k-th article and the implicit entity list of k-th article, respectively. The k-th article can be represented by a word sequence, i.e., $\mathbf{d}^k = (w_1^k, w_2^k, ..., w_{|\mathbf{d}^k|}^k)$. Let the candidate entity list be $\mathcal{C} = \{c_1, c_2, ..., c_{|\mathcal{C}|}\}$, where c_i denotes i-th entity, and $\mathbf{y}^k \in \{0,1\}^{|\mathcal{C}|}$, $\mathbf{y}_i^k = 1$ if the entity c_i is associated with the given article $\mathbf{d}^k$ but not directly mentioned within the content, otherwise $\mathbf{y}_i^k = 0$.

3.2 Graph Attentional Layer

We build a co-occurrence matrix $\mathbf{M}$ to represent the association graph of the entities in $\mathcal{C}$. The frequency of two entities c_i and c_j appearing together in the training corpus can be defined as follows.

$$\mathbf{M}_{i,j} = \sum_{k=1}^{|\mathcal{D}|} [\mathbf{y}_i^k = 1 \text{ and } \mathbf{y}_j^k = 1] \,, \tag{1}$$

where $[\cdot]$ is the Iversion bracket. $\mathbf{M}$ can be viewed as an adjacency matrix representation of an association graph. $\mathbf{M}_{i,j}$ is the value of the edge between nodes i and j, which represents the degree of association between the entities c_i and c_j. The graph attentional layer is a component of the graph attention network (GAT) (Veličković et al., 2017). In order to suit the architecture of Transformers, we modify part of GAT as follows. Firstly, the h-th score matrix $\mathbf{S}^h \in \mathbb{R}^{n \times n}$ is defined as follows.

$$\mathbf{S}^h = \frac{\mathbf{X}\mathbf{W}_Q^h (\mathbf{X}\mathbf{W}_K^h)^\top}{\sqrt{d}} \,, \tag{2}$$

where $\mathbf{X} \in \mathbb{R}^{n \times d_{\text{input}}}$ denotes the input matrix, and $\mathbf{W}_Q^h \in \mathbb{R}^{d_{\text{input}} \times d}$ and $\mathbf{W}_K^h \in \mathbb{R}^{d_{\text{input}} \times d}$ are learnable matrices.

Secondly, the h-th multi-head attention matrix $\mathbf{A}^h \in \mathbb{R}^{n \times n}$ is defined as follows.

$$\mathbf{A}_{i,j}^h = \begin{cases} \frac{e^{\mathbf{S}_{i,j}^h}}{\sum_{k \in \mathcal{N}[i]} e^{\mathbf{S}_{i,k}^h}} & \text{if } j \in \mathcal{N}[i] \\ 0 & \text{otherwise.} \end{cases} \,, \tag{3}$$

where $\mathcal{N}[i] = \{j : \mathbf{M}_{i,j} > 0\}$ represents the closed neighborhood set of node i. Lastly, we concatenate all the computational results of multi-attention heads. The output of the graph attentional layer is computed as follows.

$$\mathrm{GAL}(\mathbf{X}) = (\mathop{\Big\Vert}_{h=1}^{H} \mathbf{A}^h \mathbf{X} \mathbf{W}_V^h) \mathbf{W}_O \,, \tag{4}$$

where $\mathbf{W}_V^h \in \mathbb{R}^{d_{\text{input}} \times d}$ and $\mathbf{W}_O \in \mathbb{R}^{d_{\text{input}} \times d_{\text{input}}}$ are learnable matrices. H is the number of attention heads and $d_{\text{input}} = d \times H$.

3.3 Dynamic Graph Transformer

We first show Static Graph Transformer (SGT), which incorporates Transformer, BERT, and GAT. Then, we extend SGT to the final model, Dynamic Graph Transformer (DGT) by considering graph information dynamically.

Static Graph Transformer

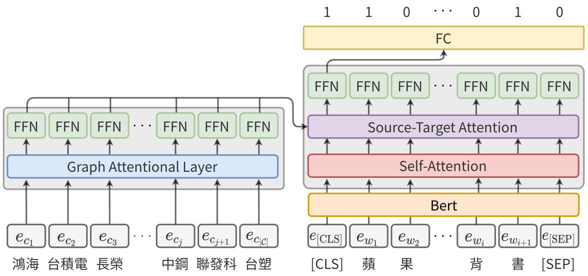
Figure 2a shows the architecture of SGT, which integrates graph and text information. The motivation behind SGT is to treat the Transformer encoder as a variation of GNN by replacing self-attention with a graph attentional layer. The Transformer encoder takes the entity's representation $\mathbf{e}_{c_j}$ as input. It is one of the row vectors in BERT's word embeddings $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d_{\text{input}}}$, where $\mathcal{V}$ denotes the vocabulary of BERT. We consider the outputs of the Transformer encoder as the entity embeddings $\mathbf{n}_1, \mathbf{n}_2, ..., \mathbf{n}_{|\mathcal{C}|}$. In SGT, the process of generating entity embedding $\mathbf{n}_j$ is static because it is irrelevant to the content information from the input article. In the Transformer decoder, we take the contextual word embeddings as the inputs of the Transformer encoder. The contextual word embeddings are the last hidden state vectors of BERT, which use the word embedding $\mathbf{e}_{w_i}$ in the article as input. Finally, we follow the settings of Devlin et al. (2018) to use the first output embedding of the Transformer decoder for predicting the implicit entity list.

Dynamic Graph Transformer

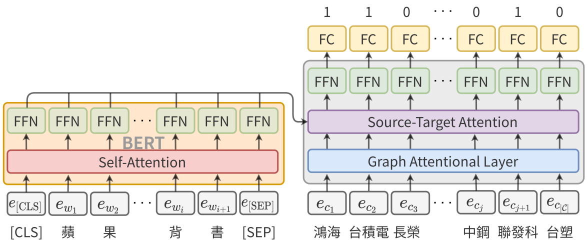
Figure 2b shows the architecture of DGT. From another perspective, since BERT is a kind of Transformer encoder, we move the learning part of entity embedding $\mathbf{n}_j$ to the Transformer decoder. We treat the last hidden state vectors of the Transformer decoder as entity embeddings $\mathbf{n}_1, \mathbf{n}_2, ..., \mathbf{n}_{|\mathcal{C}|}$. In this way, DGT is able to update the entity embedding on the fly because the source-target attention mechanism utilizes the outputs of BERT, making the model more tailored to the input article. In our task, each entity embedding is mapped to the scalar, which indicates the probability of the entity related to the article but not mentioned in it.

3.4 Pre-training by Masked Entity Prediction

Devlin et al. (2018) show that pre-training with masked language modeling and next sentence prediction is effective. Chu et al. (2020) indicate that pre-training with the value process prediction task is useful for generating correct numeric values in news headlines.



(a) Static Graph Transformer (SGT)



(b) Dynamic Graph Transformer (DGT)

Figure 2: Overview of SGT and DGT.

In this work, we propose a new pre-training task, masked entity prediction, to enrich the semantic information of entity names. Building on the original BERT vocabulary, we add a list of entity names in $\mathcal{C}$ and aim to learn their representations. We adopt the *bert-base-chinese* as the initial model and retrofit it on the training data with two sub-tasks at the same time. The first sub-task is a new masked language modeling task. Unlike the masked language model task performed for the original BERT, we not only mask the tokens using the method in Devlin et al. (2018) but also all the entity mentions in the document. The second sub-task is to label all masked entities on the position tagged as `[CLS]`. In this way, we obtain a new pre-trained model tailored to our target corpus.

4 Experiments

4.1 Dataset Description

The dataset[1] consists of 27,716 news articles collected from MoneyDJ[2], a financial newsvendor in Taiwan. Each news article is published with the labels of the related entities. These labels are annotated by professional journalists. The candidate entity list contains 735 company names, i.e., $|\mathcal{C}| = 735$. We split the dataset into the training set and the test set by time. The training set contains

---

[1] The dataset is available for academic usage by request: `http://nlg.csie.ntu.edu.tw/nlpresource/FinTag/`

[2] `https://www.moneydj.com/`

Model	Micro-F1	Macro-F1
ABC	67.11%	31.73%
TAB-LSTM	64.82%	30.47%
ITAG	60.06%	27.44%
BERT	71.72%	34.18%
VGCN-BERT	72.86%	35.65%
SGT	75.94%	42.94%
DGT	**76.16%**	**47.41%**

Table 1: Overall performances.

Model	Micro-F1	Macro-F1
SGT w/o pre-train	-1.67%	-3.78%
DGT w/o pre-train	-1.90%	-7.37%

Table 2: Performance degradation caused by the approach without the proposed pre-training task.

24,640 samples before March 24, 2017. The test set contains 3,076 samples from March 24, 2017.

4.2 Baseline Models

We adopt two kinds of models as our baselines, including the models for tag recommendation and the models for classification. The models ABC (Gong and Zhang, 2016), TAB-LSTM (Li et al., 2016), and ITAG (Tang et al., 2019) are considered as the baselines for tag recommendation. For the classification task, we adopt BERT (Devlin et al., 2018) and VGCN-BERT (Lu et al., 2020) for comparison.

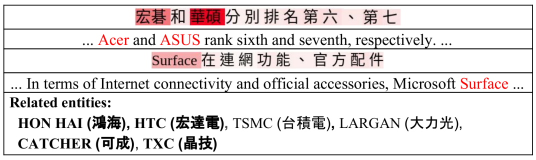
4.3 Experimental Results

We adopt the binary cross-entropy as the loss function and the Adam optimizer (Kingma and Ba, 2014) for training. The learning rate and the batch size are 3e-5 and 8, respectively. Table 1 shows that the proposed models are better than the baseline models in both micro-F1 score and macro-F1 score. In Table 2, we further show the performance degradation when using the original BERT language model instead of the language model with the proposed pre-training task. The results confirm that the proposed pre-training task is useful in the implicit relation learning task.

5 Discussion

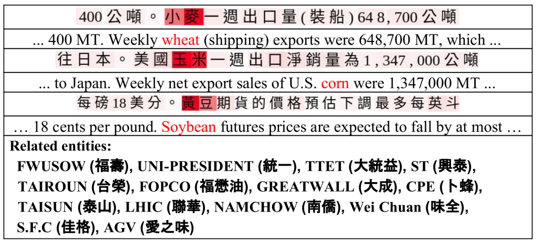
5.1 Attention Mechanism

Table 3a shows that the model focuses on the companies and the products that appear in the news article. It indicates that the proposed model captures the relationship between the companies via the related companies and the mentioned products. In Table 3b, we find that even no company names have been mentioned in the news article, DGT can still correctly infer the related entities by the mentioned raw material such as wheat, corn, and soybeans.



宏碁和華碩分別排名第六、第七
... Acer and ASUS rank sixth and seventh, respectively. ...
Surface在連網功能、官方配件
... In terms of Internet connectivity and official accessories, Microsoft Surface ...
Related entities: **HON HAI (鴻海), HTC (宏達電)**, TSMC (台積電), LARGAN (大力光), **CATCHER (可成)**, **TXC (晶技)**

(a) Inferring non-mentioned companies via mentioned companies and products.



400公噸。小麥一週出口量(裝船)648,700公噸
... 400 MT. Weekly wheat (shipping) exports were 648,700 MT, which ...
往日本。美國玉米一週出口淨銷量為1,347,000公噸
... to Japan. Weekly net export sales of U.S. corn were 1,347,000 MT ...
每磅18美分。黃豆期貨的價格預估下調最多每英斗
… 18 cents per pound. Soybean futures prices are expected to fall by at most …
Related entities: **FWUSOW (福壽), UNI-PRESIDENT (統一), TTET (大統益), ST (興泰),** **TAIROUN (台榮), FOPCO (福懋油), GREATWALL (大成), CPE (卜蜂),** **TAISUN (泰山), LHIC (聯華), NAMCHOW (南僑), Wei Chuan (味全),** **S.F.C (佳格), AGV (愛之味)**

(b) Inferring non-mentioned companies only by mentioned products.

Table 3: Examples of attention weights. All related entities are labeled by professional journalists, and the bold entities represent the model predictions.

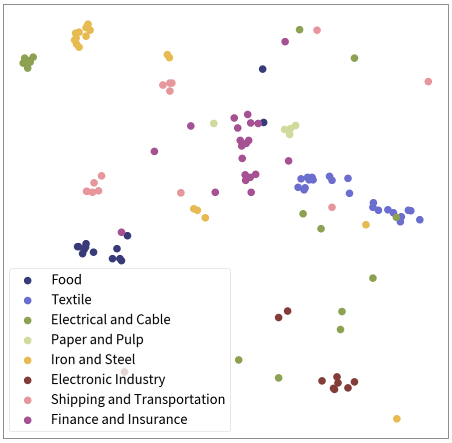


Figure 3: Visualization of entity embeddings by t-SNE.

5.2 Entity Embeddings

We use the last hidden state vectors of DGT as the corresponding entity embeddings and use t-SNE (van der Maaten and Hinton, 2008) to visualize these embeddings. As shown in Figure 3, we find that although we do not directly provide the information about the related industry or product of the entity during training, the model can still capture relationships from the corpus.

6 Conclusion and Future Work

This paper presents a novel dynamic graph transformer model and a pre-training task for extracting the implicit entities in articles. Experimental results show the usefulness of the proposed methods. We also discuss what kinds of features our model captures.

In our previous work (Liou et al., 2021), we apply the proposed task to accelerate the working process of the journalists and show that using the extracted entities could be useful for downstream tasks such as news aggregation and stock movement prediction. In the future, we plan to apply the proposed approach to datasets with both graphical knowledge and textual content.

Acknowledgments

This research was partially supported by Ministry of Science and Technology, Taiwan, under grants MOST 109-2218-E-009-014, MOST 109-2634-F-002-040, and MOST 109-2634-F-002-034.

References

Scott R Baker, Nicholas Bloom, and Steven J Davis. 2016. Measuring economic policy uncertainty. *The Quarterly Journal of Economics*, 131(4):1593–1636.

Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*.

Jui Chu, Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. Learning to generate correct numeric values in news headlines. In *Companion Proceedings of the Web Conference 2020*, pages 17–18.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4894–4903, Hong Kong, China. Association for Computational Linguistics.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1409–1418, Florence, Italy. Association for Computational Linguistics.

Yuyun Gong and Qi Zhang. 2016. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*, pages 2782–2788.

Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction. In *WSDM*, WSDM '18, pages 261–269, New York, NY, USA. ACM.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Yang Li, Ting Liu, Jing Jiang, and Liang Zhang. 2016. Hashtag recommendation with topical attention-based lstm. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3019–3029.

Yi-Ting Liou, Chung-Chi Chen, Tsun-Hsien Tang, Hen-Hsen Huang, and Hsin-Hsi Chen. 2021. Finsense: An assistant system for financial journalists and investors. In *Proceedings of the 14th International Conference on Web Search and Data Mining*.

Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. Vgcn-bert: Augmenting bert with graph embedding for text classification. In *European Conference on Information Retrieval*, pages 369–382. Springer.

Ye Ma, Lu Zong, Yikang Yang, and Jionglong Su. 2019. News2vec: News network embedding with subnode information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4843–4852, Hong Kong, China. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11).

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5057–5068.

Bonan Min and Xiaoxi Zhao. 2019. Measure country-level socio-economic indicators with streaming news: An empirical study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1249–1254, Hong Kong, China. Association for Computational Linguistics.

Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*, pages 3697–3707.

Yangtuo Peng and Hui Jiang. 2016. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 374–379, San Diego, California. Association for Computational Linguistics.

Shijie Tang, Yuan Yao, Suwei Zhang, Feng Xu, Tianxiao Gu, Hanghang Tong, Xiaohui Yan, and Jian Lu. 2019. An integral tag recommendation model for textual content. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5109–5116.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Lianwei Wu, Yuan Rao, Haolin Jin, Ambreen Nazir, and Ling Sun. 2019. Different absorption from the same sharing: Sifted multi-task learning for fake news detection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4644–4653, Hong Kong, China. Association for Computational Linguistics.

Yumo Xu and Shay B. Cohen. 2018. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1979, Melbourne, Australia. Association for Computational Linguistics.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983.

Renchun You, Zhiyao Guo, Lei Cui, Xiang Long, Yingze Bao, and Shilei Wen. 2020. Cross-modality attention with semantic graph embedding for multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Implicit Unlikelihood Training: Improving Neural Text Generation with Reinforcement Learning

Evgeny Lagutin[2,3,4] , **Daniil Gavrilov**[1,2,3] , **Pavel Kalaidin**[3,5*]
[1]VK, [2]VK Lab, [3]Moscow Institute of Physics and Technology,
[4]Skolkovo Innovation Center, Skolkovo Institute of Science and Technology [5]Tinkoff
`lagutin.em@phystech.edu`, `daniil.gavrilov@vk.com`,
`p.kalaydin@tinkoff.ai`

Abstract

Likelihood training and maximization-based decoding result in dull and repetitive generated texts even when using powerful language models (Holtzman et al., 2019). Adding a loss function for regularization was shown to improve text generation output by helping avoid unwanted properties, such as contradiction or repetition (Li et al., 2020). In this work, we propose fine-tuning a language model by using policy gradient reinforcement learning, directly optimizing for better generation. We apply this approach to minimizing repetition in generated text, and show that, when combined with unlikelihood training (Welleck et al., 2020), our method further reduces repetition without impacting the language model quality. We also evaluate other methods for improving generation at training and decoding time, and compare them using various metrics aimed at control for better text generation output.

1 Introduction

Language models have become a subject of close attention in the Natural Language Processing field over the past few years. They are widely used not only for unsupervised pre-training, but also for text generation, such as what is implemented in dialogue systems (Roller et al., 2020). While there are ongoing efforts to develop non-autoregressive models for language modeling, most current state-of-the-art approaches use the autoregressive method of generating text (i.e., word by word). Holtzman et al. (2019) showed that even powerful trained models with a high likelihood value for test data can output repetitive results. Schmidt (2019) argues that the reason for that is train-test discrepancy and lack of generalization when running standard maximum likelihood estimation (MLE) training.

Unwanted repetition can be remedied at decoding and training time. Decoding methods focus on sampling techniques that generate less repetitive or incoherent samples, while other methods aim to improve model training to minimize the effects of degeneration. An effective method for reducing language model degeneration is unlikelihood training (Welleck et al., 2020), where a regularization term forces the model to reduce the probability of generating a token that has already occurred in a sequence. Li et al. (2020) further explored this idea and showed that adding a loss function for regularization *to avoid undesirable sequences* improves text generation not only by reducing repetition, but also by decreasing contradiction. Roller et al. (2020) reported that adding unlikelihood training also improves the humanness of generated text.

In this paper, we propose **Implicit Unlikelihood Training**, a method for regularizing output by fine-tuning a language model with policy gradient reinforcement learning to improve generation results. We apply this method for a repetition objective, and show that combining Implicit Unlikelihood Training with minimizing unlikelihood loss results in reduced repetition and perplexity. We also evaluate alternative approaches to improving generated texts in terms of repetitiveness, and compare these methods using a wide variety of metrics.

The source code is available at: github.com/vklabmipt/implicit-unlikelihood-training.

---

[*] Work done while at VK.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1432–1441
April 19 - 23, 2021.

2 Related Work

2.1 Decoding Strategies

Holtzman et al. (2019) observed that maximization-based decoding methods, such as top-k sampling (Fan et al., 2018), beam search and its variations, can all lead to degeneration. They addressed this problem by using top-p (nucleus) sampling, proposing sampling from the top portion of the probability mass. Paulus et al. (2017) reported that ground-truth sentences for summarization tasks almost never contain the same trigram twice, and proposed the beam-blocking approach, where the decoder is forced to never output the same trigram more than once during testing. Penalized sampling (Keskar et al., 2019) works by discounting the scores of previously generated tokens. Martins et al. (2020) proposed preventing unlikely words from receiving any probability mass by using entmax sampling.

2.2 Training Strategies

Jiang et al. (2020) suggested that some tokens can be more difficult for a model to learn than others. These tokens are still under-learned after training, making their repetition more likely to happen. This issue is addressed by token loss dynamic reweighting (TLDR), which applies differentiable weights to individual token losses. Repetition can also be improved at training time by adding unlikelihood loss (Welleck et al., 2020; Li et al., 2020) to regular likelihood loss. Unlikelihood training is aimed at decreasing the probability of previously generated tokens, and it was shown that it can outperform beam blocking and top-p sampling.

Coverage mechanisms (Tu et al., 2016; See et al., 2017) can also be used to reduce repetition. Adding pre-attention and highway connections was shown to decrease repetition for RNNs (Jiang et al., 2020), while the architecture tweaks required for Transformers (Vaswani et al., 2017) are still an open question.

2.2.1 Unlikelihood Training

Unlikelihood Training involves adding Unlikelihood Loss to lower the probability $p_\theta(c_i|x_{<t})$ of negative candidates $\mathcal{C}^t = \{c_1, c_2, ..., c_n\}$ at each timestamp:

$$\mathcal{L}^t_{\text{UL}}\left(p_\theta\left(\cdot|x_{<t}\right), \mathcal{C}^t\right) = -\sum_{c\in\mathcal{C}^t} \log\left(1 - p_\theta\left(c|x_{<t}\right)\right). \tag{1}$$

We can construct the negative candidate set as $\mathcal{C}^t = \{x_{t-1}, x_{t_2}, ..., x_1\}\backslash\{x_t\}$ to improve generation results through reducing repetition. Welleck et al. (2020) also proposed using Sequence-Level Unlikelihood Loss on sampled continuations, where a sequence $(x_{t+1}, x_{t+2}, ..., x_{t+N})$ from a prefix $(x_1, x_2, ..., x_t)$ is sampled first, and then the loss defined in Eq. 1 for each x_{t+i} (where $1 \leq i \leq N$) with negative examples $\mathcal{C}^{t+i}$ equal to $\{x_{t+i}\}$ if x_{t+i} is a part of a repeating n-gram at a position before $t+i$ is minimized (see Algorithm 3 for details on Sequence-Level Unlikelihood Loss). Fine-tuning a language model is then performed by equally alternating between sequence-level unlikelihood and likelihood updates.

2.3 Evaluation Metrics

There are many metrics available for evaluating the performance (diversity or non-repetition) or quality of language models. These metrics include **perplexity**, the number of unique next-token predictions (**uniq**), **uniq-seq**, **rep/l**, **wrep/l** (Welleck et al., 2020), **ϵ-perplexity**, **sparsemax score** (sp), Jensen-Shannon divergence (**JSD**) (Martins et al., 2020), and **DIMEN** (Jiang et al., 2020).

Perplexity is the metric used to evaluate language model quality. It is defined as $ppl(x) = p(x_1, x_2, ..., x_t)^{-\frac{1}{t}}$, where $x_1, x_2, ..., x_t$ is the sequence of tokens from test data. The lower the perplexity, the better the language model.

Welleck et al. (2020) used a portion of duplicate n-grams in a generated sequence to measure **sequence repetition**:

$$\text{seq\_rep\_n}(\mathbf{x}) = 1 - \frac{\#\text{uniq\_ngram}(\mathbf{x})}{\#\text{total\_ngram}(\mathbf{x})}. \tag{2}$$

Higher repetition values mean that a language model tends to produce more repetitive output, which might appear less natural. Note that $0 \leq \text{seq\_rep\_n}(\mathbf{x}) < 1$.

Same as (Welleck et al., 2020), we controlled for the number of unique next-token predictions (**uniq**), as it was shown that generated texts are less diverse than those written by a human. We also used the number of unique tokens in continuations of validation or test prefixes (**uniq-seq**) as a measure of token distribution in generated text. **rep/l** is the fraction of next-token (top-1) predictions that occur in the previous l tokens. **wrep/l** is a variant of rep/l which only counts single token repetitions that are not equal to the ground truth next-token. We use 16,32,128,512 as l and average the results to compute rep and wrep.

Martins et al. (2020) introduced **ϵ-perplexity** for computing perplexity of sparse distributions. The perplexity is smoothed by adding a small value of ϵ to all terms, followed by renormalization. They also introduced **sparsemax score** (sp) and **Jensen-Shannon divergence** (JSD) for evaluating quality and sparsity of probability distributions. For deterministic models, sparsemax score becomes word accuracy, and is bounded by 0 and 1. With JSD, the distance between the sparse or truncated distribution and the one-hot encoded ground truth distribution can be measured. It is used as a metric for language models using different decoding strategies. Unlike perplexity, JSD is bounded by $\log 2$.

Jiang et al. (2020) evaluate methods with a diversity metric based on n-grams (**DIMEN**). A high DIMEN score means that a set of generated sequences is diverse.

In this paper, we mainly focused on reducing sequence repetition (**seq_rep_n**) (Welleck et al., 2020), which is a portion of duplicate n-grams in a generated sequence. Improving generation results by minimizing repetition should not significantly affect the **perplexity** of the language model.

3 Implicit Unlikelihood Training

Li et al. (2020) showed that Unlikelihood Training can be employed as a general framework for reducing the likelihood of undesirable text generation results through training on negative examples. However, we argue that, in some cases, it could be difficult to construct negative samples for specific types of Unlikelihood Loss[1]. To address this issue, we propose extending Unlikelihood Training with policy gradient reinforcement learning, which does not need explicitly created negative samples.

We chose to test this approach for repetition as the most widely considered property of neural text degeneration. To directly minimize repetition (see Equation 2) for sequence x, we define the reward as $R = 1 - \text{seq\_rep\_n}(\mathbf{x})$ with $n = 4$. We alternated between maximizing the reward R, optimizing the likelihood of training data, and Sequence-Level Unlikelihood Loss (see 1, 2 and 3 for details on the process of Implicit Unlikelihood Training and policy gradient update).

Algorithm 1: i-UT: alternating between MLE, UT and PG updates

```
Input: update rate $r$, total number of updates $N$
for $i$ = 1 to $N$ do
    sample $x \sim \mathrm{U}[0,1]$
    if $x < r$ then
        sample $y \sim \mathrm{U}[0,1]$
        if $y < 0.5$ then
            do a policy gradient update
        else
            do a sequence-level unlikelihood update
        end
    else
        do a MLE update
    end
end
```

Algorithm 2: Policy Gradient Update

```
Input: LM $\theta$, $m$ prefixes
       $\mathcal{D}_m = \{(x_1^{(j)}, \ldots, x_k^{(j)}),\ j = \overline{1..m}\}$,
       continuation length $T$
Output: loss $\mathcal{L}(\theta, \mathcal{D}_m)$
for $j$ = 1 to $m$ do
    for $t$ = $k+1$ to $k+T$ do
        Get $p_\theta(\cdot \mid x_{<t}^{(j)})$
        $x_t^{(j)} = \arg\max_{x \in \mathcal{V}} p_\theta(\cdot \mid x_{<t}^{(j)})$
    end
end
for $j$ = 1 to $m$ do
    $R_j = 1 - \text{seq\_rep\_n}(\mathbf{x}^{(j)})$
end
$b(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}) = \sum_{j=1}^{m} R_j$
for $j$ = 1 to $m$ do
    $\Psi_j = R_j - b(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}) \cdot \frac{1}{m}$
end
$\mathcal{L}(\theta, \mathcal{D}_m) =$
    $-\frac{1}{m} \sum_{j=1}^{m} \Psi_j \cdot \frac{1}{T} \sum_{t=k+1}^{k+T} \log p_\theta(x_t^{(j)} \mid x_{<t}^{(j)})$
```

4 Experiment Details

4.1 Setup

We fine-tuned small and medium GPT-2 models (175M and 345M parameters, respectively) (Radford et al., 2019) on the WikiText-103 dataset (Merity et al., 2016).

---

[1] This can include reducing the toxicity or bias level of generated sequences by using a score from an external classifier.

method	ppl↓	uniq↑	seq_rep_4↓				
		top-k, k=1	top-k, k=1	top-k, k=3	top-k, k=8	top-p, p=0.3	top-p, p=0.9
small GPT-2 model, 0.5 update rate							
PG, c=3	19.409±.195	11259±52	.032±.014	.024±.010	.017±.006	.029±.012	.007±.001
PG + UT, c=3	19.344±.034	11279±53	.010±.002	.008±.001	.008±.000	.012±.002	.007±.001
i-UT, c=3	19.182±.032	11308±73	.009±.002	.008±.001	.008±.001	.012±.002	.007±.001
i-UT, c=9	19.302±.051	11297±36	.006±.001	.006±.001	.007±.002	.009±.002	.006±.001
i-UT, c=15	**19.170±.123**	**11432±34**	.007±.005	.007±.002	**.007±.001**	.010±.003	.007±.001
i-UT, c=30	19.504±.065	11427±66	**.005±.000**	**.005±.001**	**.007±.001**	**.005±.001**	**.006±.001**
UT	19.442±.085	11210±47	.056±.033	.011±.002	.008±.001	.014±.003	**.006±.001**
UT w/ warmup	19.347±.065	11295±4	.055±.016	.010±.001	.008±.001	.014±.002	**.006±.001**
small GPT-2 model, 0.25 update rate							
PG, c=3	18.516±.086	11492±32	.060±.011	.035±.005	.022±.003	.043±.006	.009±.001
PG, c=9	18.638±.022	11496±21	.025±.011	.022±.008	.018±.005	.025±.008	.009±.001
PG, c=15	18.696±.062	11533±3	.022±.002	.019±.001	.016±.000	.021±.001	.008±.001
PG, c=30	19.026±.099	11487±36	.031±.021	.027±.017	.020±.010	.027±.016	.008±.001
i-UT, c=3	18.504±.059	11519±21	.020±.002	.014±.001	.012±.001	.017±.002	**.007±.001**
i-UT, c=9	**18.487±.022**	11552±51	.011±.003	.011±.003	.011±.002	.016±.003	**.007±.000**
i-UT, c=15	18.590±.065	11504±24	**.008±.001**	**.008±.001**	**.008±.001**	.011±.002	**.007±.001**
i-UT, c=30	18.733±.082	**11558±27**	**.007±.001**	**.008±.001**	.009±.001	**.010±.001**	.008±.000
UT	18.764±.164	11377±55	.055±.013	.011±.002	.009±.001	.016±.003	**.007±.001**
medium GPT-2 model, 0.5 update rate							
i-UT, c=3	**13.620±.015**	12437±29	.013±.002	.010±.002	.010±.001	.011±.002	.008±.001
i-UT, c=15	13.669±.018	12355±3	**.011±.003**	.010±.002	.011±.002	.012±.001	.009±.000
i-UT, c=30	13.785±.004	**13319±898**	**.011±.003**	.009±.003	.011±.001	**.010±.004**	**.008±.000**
UT	13.710±.009	12386±29	.024±.003	**.008±.001**	**.009±.001**	.013±.001	.008±.001

Table 1: Repetition on small and medium GPT-2 models. Validation data was used as sampling prefixes for evaluating the metrics.

sampling	method	uniq↑	rep↓	wrep↓	$\epsilon-$ppl/ppl↓	JSD↓	sp↑
softmax $\epsilon = 0$	MLE	19932	.373	**.174**	13.830	.382	.680
	i-UT, c=3	**20493**	.372	**.174**	**13.161**	.379	**.683**
	UT	20419	.371	**.174**	13.266	.380	.682
greedy $\epsilon = 2 \times 10^{-5}$	MLE	12639	.489	.230	539.930	.358	.483
	i-UT, c=3	12859	.488	.228	511.478	**.355**	.488
	UT	12826	.488	.228	517.316	.356	.487
top-k, k=10 $\epsilon = 8 \times 10^{-6}$	MLE	14326	.436	.220	50.003	.358	.668
	i-UT, c=3	14731	.435	.219	46.885	**.355**	.672
	UT	14648	.435	.219	47.248	.356	.671
top-p, p=0.9 $\epsilon = 2 \times 10^{-6}$	MLE	17589	.395	.186	19.689	.371	.678
	i-UT, c=3	18116	.394	.185	18.662	.368	.682
	UT	17964	.393	.186	18.782	.369	.681
α-entmax ($\alpha = 1.2$) $\epsilon = 1 \times 10^{-6}$	α-entmax loss ($\alpha = 1.2$)	19942	**.370**	.176	15.124	.389	.680

Table 2: Repetition on the medium GPT-2 model with a 0.5 update rate. Regular perplexity is reported for softmax sampling. Test data is used as sampling prefixes for evaluating the metrics.

Our experiments consisted of alternating between three types of updates: Maximizing Likelihood (MLE), minimizing Sequence-Level Unlikelihood Loss (UL), and minimizing repetition with policy gradient reinforcement learning (PG). The first approach is a plain **MLE** update, for which we do not use any specific methods for reducing repetition in samples. We also experimented with Unlikelihood Training (**UT**), which involves alternating between MLE and UL updates (see Algo-

Algorithm 3: Sequence-Level Unlikelihood Update

```
Input: LM $\theta$, batch of prefixes
       $\mathcal{D}_m = \{(x_1^{(j)}, \dots, x_k^{(j)}),\ j = \overline{1..m}\}$,
       continuation length $T$
Output: loss $\mathcal{L}(\theta, \mathcal{D}_m)$
for $j$ = 1 to $m$ do
    for $t$ = $k+1$ to $k+T$ do
        Get $p_\theta(\cdot \mid x_{<t}^{(j)})$
        $x_t^{(j)} = \arg\max_{x \in \mathcal{V}} p_\theta(\cdot \mid x_{<t}^{(j)})$
    end
end
for $j$ = 1 to $m$ do
    for $t$ = $k+1$ to $k+T$ do
        if $(x_{t-i}, \dots, x_t, \dots, x_{t+h}) \in$
          $x_{<t-i}^{(j)}$ for any $(h-i) = n, i \leq n \leq h$
          then
            $\mathcal{C}^t_{\text{repeat-n}}(\mathbf{x}^{(j)}) = \{x_t\}$
        else
            $\mathcal{C}^t_{\text{repeat-n}}(\mathbf{x}^{(j)}) = \emptyset$
        end
    end
end
$\mathcal{L}(\theta, \mathcal{D}_m) =$
  $\frac{1}{m} \sum_{j=1}^{m} \frac{1}{T} \sum_{t=k+1}^{k+T} \mathcal{L}^t_{UL}\left(p_\theta(\cdot \mid x_{<t}^{(j)}), \mathcal{C}^t_{\text{repeat-n}}\right)$
```

rithm 1 for details).

In policy gradient experiments, we trained models in three different scenarios: a plain **PG** for which we alternated MLE and PG updates; a combined **PG + UT** approach, where we alternated between maximizing the likelihood and minimizing the sum of policy gradient and unlikelihood losses; and finally, the proposed Implicit Unlikelihood Training (**i-UT**), which consisted of alternating between MLE, UL and PG updates (see Algorithms 1, 3, and 2). We used 0.25 and 0.5 alternating update rates for the small GPT-2 model, and r equal to 0.5 for the medium GPT-2 model.

Full optimization details are provided in **Appendix A.1. Optimization Details**.

4.2 Evaluation

We used top-k and top-p samplings with different k and p to evaluate sequence repetition[2] (seq_rep_4) for the described approaches. For these experiments, we used validation data to evaluate perplexity and to generate the sampling prefixes for evaluating uniq and seq_rep_4 metrics. The number of unique tokens (uniq) was evaluated using greedy sampling. We also evaluated the proposed method with rep, wrep, and JSD metrics using different sampling methods on test data, and compared it with other related approaches (MLE, UT, entmax). We repeated each experiment 5 times and reported the mean and standard deviation values of the measurements.

More experiments and their results are described in **Appendix A.2. Experiments**.

5 Results

We showed that Implicit Unlikelihood Training is a competitive approach that outperforms other methods in sequence repetition when fine-tuning small and medium GPT-2 models (see Table 1) on most variants of top-k and top-p sampling, while maintaining the lowest perplexity and the highest count of unique tokens generated. This approach also achieved better results than training with entmax loss and other related approaches, using a different range of sampling methods (see Table 2), with the only exception being the rep metric, where entmax performed similar to i-UT.

Samples of generated outputs are provided in Tables 8, 9 in Appendix.

6 Future Work

The described and evaluated reinforcement learning framework makes it possible to optimize text generation for any objective. In future work, we intend to test the approach not only for repetition, but also for various other metrics, such as the toxicity level or bias of generated text.

Acknowledgements

The authors are grateful to anonymous reviewers for valuable comments and to Viktoriia Loginova and David Prince for proofreading.

References

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*, page 889–898.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *arXiv preprint arXiv:1904.09751*.

Shaojie Jiang, Thomas Wolf, Christof Monz, and Maarten de Rijke. 2020. Tldr: Token loss dynamic reweighting for reducing repetitive utterance generation. In *arXiv preprint arXiv:2003.11963*.

---

[2] Note that top-k with $k = 1$ is greedy sampling by definition.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, , and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. In *arXiv preprint arXiv:1909.05858*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2020. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. In *ACL*.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2020. Sparse text generation. In *arXiv preprint arXiv:2004.02644*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. In *arXiv preprint arXiv:1609.07843*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. In *arXiv preprint arXiv:1705.04304*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M. Smith, Y-Lan Boureau, and Jason Weston. 2020. Recipes for building an open-domain chatbot. In *arXiv preprint arXiv:2004.13637*.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. In *EMNLP Workshop on Neural Generation and Translation (WNGT)*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. In *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. In *arXiv preprint arXiv:1707.06347*.

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

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

A Appendices

A.1 Optimization Details

For likelihood update, we evaluated the likelihood of a sequence of tokens with lengths equal to 300. For both UL and PG updates, we formed prefixes using a sequence of 300 tokens to form 6 sequences with lengths equal to 50. We then used these prefixes to sample sequences with a maximum length of 100 tokens.

For optimization, we used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 6.25×10^{-5}. Similar to (Welleck et al., 2020) and (Martins et al., 2020), we did no warmup steps for UT and $\alpha-$entmax training. For i-UT, we did 500 linear warm-up steps . After warm-up steps, we linearly decayed the learning rate to zero.

In all our experiments, we fine-tuned language models for 5000 total updates.

Once training was complete, we selected a checkpoint with the least validation perplexity obtained during training. This is the last checkpoint in most of our experiments, which means that general log-likelihood loss converges.

As shown in Algorithm 1, we equally alternated between UL and PG updates. We also found that reducing unlikelihood update rate to 0.25 may also be effective, taking twice less time (see Table 1). The parameters ϵ for $\epsilon-$ppl and α for $\alpha-$entmax training were taken from (Martins et al., 2020) (except ϵ for $\alpha-$entmax, which we set to 1×10^{-6}).

We conducted coefficient search on our policy gradient loss with $c = \{3, 9, 15, 30\}$ for the small GPT-2 model, and $c = \{3, 15, 30\}$ for medium GPT-2 model. We chose the best models based on the results on the validation set, and also reported the metrics on the test set.

A.2 Experiments

We evaluated DIMEN and uniq-seq for UT and i-UT methods, applied to small and medium GPT-2 models using different sampling methods for DIMEN, and greedy sampling for evaluation of uniq-seq. In this experiment, we observed that Implicit Unlikelihood Training performed better or equal to Unlikelihood Training with different sampling methods measured by the DIMEN metric, having a significantly better value of uniq-seq (see Table 7).

We also evaluated sequence repetition with beam-search sampling for MLE, UT, and i-UT methods for both small and medium GPT-2 models, using validation data to form sampling prefixes. When sampling with beam search, we found that Implicit Unlikelihood Training produced better results than Unlikelihood Training (see Table 3).

For greedy sampling with small GPT-2 model, we evaluated sequence repetition, wrep, uniq, and perplexity. We used test data to evaluate the perplexity, and to form sampling prefixes for other methods. We observed that MLE, UT, and i-UT methods had similar performance in terms of repetition using greedy sampling, while i-UT still had the best number of unique tokens (see Table 4).

Finally, we evaluated the TLDR method using both sequence repetition and DIMEN metrics (see Tables 5, 6). In our experimental setup, TLDR performed on par with MLE approach.

B Negative Results

Our results showed that all sampling methods, other than greedy sampling, led to worse convergence of the seq_rep_4 metric.

We experimented with using the Proximal Policy Optimization algorithm (Schulman et al., 2017) for PG update, but faced unstable validation perplexity behavior during training, and did not obtain any comparable results.

Another unsuccessful direction of our experiments was substituting the estimation of the reward calculated on the full sequence with the reward put on each token separately. We tried using two variants of the binary reward function: does the current n-gram appear first time in the text, and does the current n-gram appear in the following part of the text. We experimented with advantage estimation by using a value function estimator, and without it by using pure rewards. In the former case, we adjusted different values of λ and γ for the Generalized Advantage Estimation algorithm (Schulman et al., 2015), and in the latter, we used a general discounted future reward. We observed that the approach of estimating a single reward for a whole sequence and subtracting a baseline value to reduce the variance of the gradient estimation performed best.

	seq_rep_4↓		
GPT-2:	small		medium
seq rate:	.5	.25	.5
MLE	.67±.01	.67±.01	.64±.01
i-UT	**.03±.04**	**.08±.03**	**.08±.02**
UT	.08±.03	.14±.02	.13±.02

Table 3: Repetition with Beam Search. Validation data is used as sampling prefixes for evaluating the metrics. We reported the results of i-UT model with the value of c for which we had the best validation perplexity.

method	**ppl↓**	**rep↓**	**wrep↓**	**uniq↑**
MLE	17.94±.03	.504±.001	.252±.001	11790±92
i-UT, c=15	18.54±.13	.504±.001	.254±.0	11847±32
UT	18.76±.09	.503±.001	.253±.001	11597±52

Table 4: Repetition on small GPT-2 model, UT and i-UT 0.5 update rate with greedy sampling. Test data is used as sampling prefixes for evaluating the metrics.

method	**ppl↓**	**uniq↑**	**seq_rep_4↓**				
		top-k, k=1	**top-k, k=1**	**top-k, k=3**	**top-k, k=8**	**top-p, p=0.3**	**top-p, p=0.9**
MLE	18.611±.088	11361±37	.550±.035	.131±.004	.054±.001	.233±.005	.013±.001
TLDR	18.713±.059	11322±42	.514±.012	.118±.004	.047±.002	.241±.005	.011±.001

Table 5: Repetition on small GPT-2 model for TLDR and MLE approaches. Validation data is used as sampling prefixes for evaluating the metrics.

method	**uniq-seq↑**	**DIMEN↑**				
	top-k, k=1	**top-k, k=1**	**top-k, k=3**	**top-k, k=8**	**top-p, p=0.3**	**top-p, p=0.9**
MLE	8074.75±182.398	.365±.023	.685±.003	.778±.001	.599±.004	.869±.001
TLDR	8122.0±112.018	.390±.008	.692±.004	.781±.002	.590±.004	.867±.001

Table 6: Diversity on small GPT-2 model for TLDR and MLE approaches. Validation data is used as sampling prefixes for evaluating the metrics.

method	**uniq-seq↑**	**DIMEN↑**				
	top-k, k=1	**top-k, k=1**	**top-k, k=3**	**top-k, k=8**	**top-p, p=0.3**	**top-p, p=0.9**
small GPT-2 model, 0.5 update rate						
i-UT, c=3	**11340±703**	.855±.008	.859±.005	.859±.003	.834±.005	.880±.003
i-UT, c=9	10547±533	**.881±.014**	.876±.003	.870±.004	.852±.0	**.881±.003**
i-UT, c=15	10621±489	.863±.023	.867±.013	.864±.006	.855±.014	**.881±.002**
i-UT, c=30	10771±265	**.881±.009**	**.880±.008**	**.871±.007**	**.880±.012**	.880±.003
UT	9651±436	.785±.044	.847±.012	.856±.005	.831±.011	.880±.002
small GPT-2 model, 0.25 update rate						
i-UT, c=3	10885±375	.817±.006	.834±.003	.845±.002	.819±.003	.878±.002
i-UT, c=9	11208±884	.852±.014	.849±.01	.851±.007	.828±.01	.879±.001
i-UT, c=15	**11966±1019**	.848±.012	.854±.006	.856±.004	.837±.006	.879±.001
i-UT, c=30	10418±762	**.874±.012**	**.871±.01**	**.863±.007**	**.863±.017**	.878±.002
UT	9696±441	.778±.017	.843±.007	.854±.003	.827±.008	**.880±.002**
medium GPT-2 model, 0.5 update rate						
i-UT, c=3	**12900±303**	.843±.006	.854±.005	.857±.004	.845±.006	**.877±.002**
i-UT, c=15	13849±0	.869±.003	.861±.002	.857±.003	.854±.0	.875±.001
i-UT, c=30	11748±340	**.880±.004**	**.871±.01**	**.861±.006**	**.867±.009**	**.877±.002**
UT	12649±262	.837±.006	.869±.004	.860±.003	.842±.005	.875±.001

Table 7: Diversity on small and medium GPT-2 models. Validation data is used as sampling prefixes for evaluating the metrics.

		seq_rep_4
prefix:	Air Vice Marshal Frank Headlam, CB, CBE (15 July 1914 – 23 December 1976) was a senior commander in the Royal Australian Air Force (RAAF). Born and educated in Tasmania, he joined the RAAF in	
	medium GPT-2 model, 0.5 update rate, top-k, k=1	
MLE	1914, and was promoted to the rank of CB. He was promoted to the rank of CBE in 1916, and was promoted to the rank of CB. He was promoted to the rank of CBE in 1917, and was promoted to the rank of CB. He was promoted to the rank of CBE in 1918, and was promoted to the rank of CB. He was promoted to the rank of CBE in 1919, and was promoted to the rank of CB. He was promoted to	.6535
UT	1914 and was promoted to the rank of lieutenant in 1916. He was promoted to commandant in the Australian Air Force (R @-@ Special Operations Command (C-MEDF) in spare parts for related acronym changed to <unk>) Special Operations Command (Special Commander in Command by extension) being a technical specialist in communications equipment and equipment related to intelligence gathering and analysis, and related related related jargon, and related related related equipment related to logistics and logistics management. Headlam	.0288
i-UT	1914 and served in the Australian Army Air Corps (AAF) from 1915 to 1918. Headlam was promoted to the rank of lieutenant colonel in 1918 and promoted to the rank and title of commander in the Pacific theatre in the Pacific theatre in (R & R) <unk> in order to be eligible for the Medal of Honor for gallantry in action in the Battle of the Gulf, but the award was not awarded until the end of the war. Headley was awarded the Distinguished Service	.0189
	medium GPT-2 model, 0.5 update rate, top-k, k=8	
MLE	1917 at the age of twenty @-@ seven, and served at both the Western and Eastern Airports, and the Western Reserve Air Training Center. He was awarded a Distinguished Service Order in 1917, and a Military Medal for his service in World War I, and a Distinguished Service Order for his services during the Great Depression. In 1920 he was appointed as Chief of the Staff of the RAAF and served as the RAAF's Chief of Staff. In 1921 he was appointed a	.0211
UT	1939 as a private with the rank of Captain, serving on board Australian carriers, such as the RAN @-@ class battleships HMAS Melbourne and HMAS Sydney. In the early years of the war the Australian navy's air force, with the exception of the RAN, was under the command and control of Rear @-@ Admiral Robert W. Campbell, the commander of the RAAF in Australia. During the Second World War, Headlam became a member of the RAN	.0105
i-UT	1917 as a private and served in the Second World War. Headlam became the second man in the RAAF to be awarded the Victoria Cross in December 1941 for valour and gallantry in a landmine @-@ exploding raid on a German convoy in the Gulf of Mexico. Headlam became Commander of the Royal Australian Air Force in July 1943 after serving as a Lieutenant in the RAAF from 1917 to 1919 and then as Commander of the RAAF in the First World War and	.0104

Table 8: Generation Samples

		seq_rep_4
prefix:	Air Vice Marshal Frank Headlam, CB, CBE (15 July 1914 – 23 December 1976) was a senior commander in the Royal Australian Air Force (RAAF). Born and educated in Tasmania, he joined the RAAF in	
	medium GPT-2 model, 0.5 update rate, top-p, k=0.9	
MLE	1940 as a gunnery instructor and officer at Hobson Air Station, Melbourne. On 3 January 1940 he led a small squad that formed part of the Caraboo Air Force (the first to be taught at a Melbourne naval base) that did air training and drills at the Haro Air Force Base. Headlam was later commanding the 200th (First United States Army Air Service Group, Unit A) and Unit B (the second to be taught at a Melbourne naval base). In the	.0526
UT	December 1914 at the age of 15 years and 177 days. Headlam served with the RAAF and the Italian Free Fire Division before leaving in the middle of the War in October 1917 to become a field marshal of the Melbourne Transport Service. In 1919 he was nominated for a Distinguished Service Order (DSO) and he retired as a Field Marshal in December 1918. After the War he served in the RAAF as a field marshal until leaving to become an air force field marshal in Headlam	.0104
i-UT	October 1914 as a radio operator and served with distinction until his discharge from the RAAF in December. In February 1915, Headlam was appointed to the role of President of the RAC, and a policy advisory officer to the PM. He was promoted to the rank of Chief of the Air Staff, and underlined that he was a major and not to have the responsibility of determining the combatant classes (like the Royal Australian Flying Corps). When the RAC was created in	.0

Table 9: Generation Samples

Civil Rephrases Of Toxic Texts With Self-Supervised Transformers

Léo Laugier
Télécom Paris,
Institut Polytechnique de Paris, France
`leo.laugier@telecom-paris.fr`

John Pavlopoulos
Department of Computer and System Sciences
Stockholm University, Sweden
`annis@aueb.gr`

Jeffrey Sorensen
Google
`sorenj@google.com`

Lucas Dixon
Google
`ldixon@google.com`

Abstract

Platforms that support online commentary, from social networks to news sites, are increasingly leveraging machine learning to assist their moderation efforts. But this process does not typically provide feedback to the author that would help them contribute according to the community guidelines. This is prohibitively time-consuming for human moderators to do, and computational approaches are still nascent. This work focuses on models that can help suggest rephrasings of toxic comments in a more civil manner. Inspired by recent progress in unpaired sequence-to-sequence tasks, a self-supervised learning model is introduced, called CAE-T5[1]. CAE-T5 employs a pre-trained text-to-text transformer, which is fine tuned with a denoising and cyclic auto-encoder loss. Experimenting with the largest toxicity detection dataset to date (Civil Comments) our model generates sentences that are more fluent and better at preserving the initial content compared to earlier text style transfer systems which we compare with using several scoring systems and human evaluation.

1 Introduction

There are many ways to express our opinions. When we exchange views online, we do not always immediately measure the emotional impact of our message. Even when the opinions expressed are legitimate, well-intentioned and constructive, a poor phrasing may make the conversation go awry (Zhang et al., 2018a). Recently, Natural Language Processing (NLP) research has tackled the problem of abusive language detection by developing accurate classification models that flag toxic (or abusive, offensive, hateful) comments (Davidson et al., 2017; Pavlopoulos et al., 2017; Wulczyn et al., 2017; Gambäck and Sikdar, 2017; Fortuna and Nunes, 2018; Zhang et al., 2018a; Van Hee et al., 2018; Zampieri et al., 2019).

INPUT OFFENSIVE COMMENT	you now have to defend this clown along with his russian corruption.
GENERATED CIVIL COMMENT	**you now have to defend this guy from his russian ties........**
INPUT OFFENSIVE COMMENT	blaming trudeau and the government is just stupid.
GENERATED CIVIL COMMENT	**blaming trudeau and the liberal government is just wrong.**
INPUT OFFENSIVE COMMENT	dubya[2]was a moron.
GENERATED CIVIL COMMENT	**dubya was a republican.**

Table 1: Examples of offensive sentences from the Civil Comments test set and the more civil rephrasing generated by our model. The third example shows that its strategy may involve shifting the original intent, since "republican" is not a non-offensive synonym of "moron".

The prospect of healthier conversations, nudged by Machine Learning (ML) systems, motivates the development of Natural Language Understanding and Generation (NLU and NLG) models that could later be integrated in a system suggesting alternatives to vituperative comments before they are posted. A first approach would be to train a text-to-text model (Bahdanau et al., 2014; Vaswani et al., 2017) on a corpus of parallel comments where each offensive comment has a courteous and fluent rephrasing written by a human annotator. However, such a solution requires a large paired labeled dataset, in practice difficult and expensive to collect (see Section 4.5). Consequently, we limit our setting to the unsupervised case where the comments are only annotated in attributes related to toxicity, such as the Civil Comments dataset (Borkan et al., 2019). We summarize our investigations with the following research question:

---

[1] The code can be found at https://github.com/LeoLaugier/conditional-auto-encoder-text-to-text-transfer-transformer.

[2] A nickname for George W. Bush.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1442–1461
April 19 - 23, 2021.

RQ: *Can we fine-tune end-to-end a pre-trained text-to-text transformer to suggest civil rephrasings of rude comments using a dataset solely annotated in toxicity?*

Answering this question might provide researchers with an engineering proof-of-concept that would enable further exploration of the many complex questions that arise from such a tool being used in conversations. The main contributions of this work are the following:

- We addressed for the second time the task of unsupervised civil rephrases of toxic texts, relying for the first time on the Civil Comments dataset, and achieving results that reflect the effectiveness of our model over baselines.
- We developed a non-task specific approach (i.e. with no human hand-crafting in its design) that can be generalized and later applied to related and/or unexplored attribute transfer tasks.

While several of the ideas we combine in our model have been studied independently, to the best of our knowledge, no existing unsupervised models combine sequence-to-sequence bi-transformers, transfer learning from large pre-trained models, and self-supervised fine-tuning (denoising auto-encoder and cycle consistency). We discuss the related work introducing these tools and techniques in the following section.

2 Related work

Unsupervised complex text attribute transfer (like civil rephrasing of toxic comments) remains in its early stages, and our particular applied task has only a single antecedent (Nogueira dos Santos et al., 2018). There is a great variety of useful works to tackle the task and this section attempts to summarize the vast majority of these works. We describe below the recent strategies (such as attention mechanisms Bahdanau et al., 2014) that led to significant progress in supervised NLU and NLG tasks. Then, we present the most related lines of work in unsupervised text-to-text tasks.

2.1 Transformers[3] are state-of-the-art architectures in NLP

Vaswani et al. (2017) showed that transformer architectures, based on attention mechanisms, achieved state-of-the-art results when applied to supervised Neural Machine Translation (NMT). More generally, transformers have proven capable in various NLP and speech tasks (Dong et al., 2018; Huang et al., 2019; Le et al., 2019; Li et al., 2019). Moreover, transformers benefit from pre-training before being fine-tuned on downstream tasks (Devlin et al., 2019; Dai et al., 2019b; Yang et al., 2019; Conneau and Lample, 2019; Raffel et al., 2019). Subsequent research has adopted uni-transformers in many supervised classification and regression tasks (Devlin et al., 2019) and in unsupervised language modeling (Radford et al., 2019; Keskar et al., 2019; Dathathri et al., 2020), until Raffel et al. (2019) proposed a unified pre-trained bi-transformer applicable to any text classification, text regression and text-to-text task. Further, recent works tackle the language detoxification of unconditional language models (Krause et al., 2020; Gehman et al., 2020).

2.2 Unsupervised losses enable training text-to-text models end-to-end

After the success of unsupervised image-to-image style transfer in computer vision (CV), some approaches have addressed unsupervised text-to-text tasks. Unsupervised Neural Machine Translation (UNMT) is maybe the most promising of them. Artetxe et al. (2018); Conneau et al. (2018); Lample et al. (2018a,b); Conneau and Lample (2019) introduced methods based on techniques aligning the embedding spaces of monolingual datasets and tricks such as denoising auto-encoding losses (Vincent et al., 2008) and back-translation (Sennrich et al., 2015; Edunov et al., 2018).

Abstractive summarization (or sentence compression) is also studied in unsupervised settings. Baziotis et al. (2019) trained a model with a compressor-reconstructor strategy similar to back-translation while Liu et al. (2019b) trained a denoising auto-encoder that embeds sentences and paragraphs in a common space.

Unsupervised attribute transfer is the task most related to our work. It mainly focuses on sentiment transfer with standard review datasets (Maas et al.,

---

[3] To avoid confusion we denote as bi-transformer the original encoder-decoder transformer whereas encoder-only and decoder-only models are called uni-transformers here.

2011; He and McAuley, 2016; Shen et al., 2017; Li et al., 2018), but also addresses sociolinguistic datasets containing text in various registers (Gan et al., 2017; Rao and Tetreault, 2018) or with different identity markers (Voigt et al., 2018; Prabhumoye et al., 2018; Lample et al., 2019). When paraphrase generation aims at being explicitly attribute-invariant, it is referred as obfuscation or neutralization (Emmery et al., 2018; Xu et al., 2019b; Pryzant et al., 2020). Literary style transfer (Xu et al., 2012; Pang and Gimpel, 2019) has also been tackled by recent work. Here, we apply attribute transfer to a large dataset annotated in toxicity, but we also use the Yelp review dataset from Shen et al. (2017) for comparison purposes (see Section 4).

Initial unsupervised attribute transfer approaches sought to build a shared and attribute-agnostic latent representation encoding for the input sentence, with adversarial training. Then, a decoder, aware of the destination attribute, generated a transferred sentence (Shen et al., 2017; Hu et al., 2017; Fu et al., 2018; Zhang et al., 2018c; Xu et al., 2018; John et al., 2019).

Unsupervised attribute transfer approaches that do not rely on a latent space are also present in literature. Li et al. (2018) assumed that style markers are very local and proposed to delete the tokens most conveying the attribute, before retrieving a second sentence in the destination style. They eventually combined both sentences with a neural network. Lample et al. (2019) applied UNMT techniques from Conneau and Lample (2019) to several attribute transfer tasks, including social media datasets. Xu et al. (2018); Gong et al. (2019); Luo et al. (2019); Wu et al. (2019a) trained models with reinforcement learning. Dai et al. (2019b) introduced unsupervised training of a transformer called StyleTransformer (ST) with a discriminator network. Our approach differs from these unsupervised attribute transfer models in that they did not either leverage large pre-trained transformers, or train with a denoising objective.

The most similar work to ours is Nogueira dos Santos et al. (2018) who trained for the first time an encoder-decoder rewriting offensive sentences in a non-offensive register with non-parallel data from Twitter (Ritter et al., 2010) and Reddit (Serban et al., 2017). Our approach differs in the following aspects. First, we use transformers pre-trained on a large corpus instead of randomly initialized RNNs for encoding and decoding. Second, their approach involves collaborative classifiers to penalize generation when the attribute is not transferred, while we train end-to-end with a denoising auto-encoder. Even if their model shows high accuracy scores, it suffers from low fluency, with offensive words being often replaced by a placeholder (e.g. "big" instead of "f*cking").

As underlined by Lample et al. (2019), applying Generative Adversarial Networks (GANs) (Zhu et al., 2017) to NLG is not straightforward because generating text implies a sampling operation that is not differentiable. Consequently, as long as text is represented by discrete tokens, loss gradients computed with a classifier cannot be back-propagated without tricks such as the REINFORCE algorithm (He et al., 2016) or the Gumbel-Softmax approximation (Baziotis et al., 2019) which can be slow and unstable. Besides, controlled text generation (Ficler and Goldberg, 2017; Keskar et al., 2019; Le et al., 2019; Dathathri et al., 2020) is a NLG task that consists of a language model conditioned on the attributes of the generated text such as the style. But a major difference with attribute transfer is the absence of a constraint regarding the preservation of the input's content.

3 Method

3.1 Formalization of the attribute text rewriting problem

Let X_T and X_C be our two non-parallel corpora of comments satisfying the respective attributes "toxic" and "civil". Let $X = X_T \cup X_C$. We aim at learning a parametric function f_θ mapping a pair of source sentence x and destination attribute a to a fluent sentence y satisfying a and preserving the meaning of x. In our case, there are two attributes "toxic" and "civil" that we assumed to be mutually exclusive. We denote $\alpha(x)$ to be the attribute of x and $\bar{\alpha}(x)$ the other attribute (for instance when $\alpha(x) =$ "civil", then $\bar{\alpha}(x) =$ "toxic"). Note that $f_\theta(x, \alpha(x))$ can simply be x.

3.2 Our approach is based on bi-conditional encoder-decoder generation

Our approach is to train an autoregressive (AR) language model (LM) conditioned on both the input text x and the destination attribute a.

We compute f_θ with a LM $p(y|x, a; \theta)$. As we do not have access to ground-truth targets y, we propose in section 3.3 a training function that we assume to maximize $p(y|x, a; \theta)$ if and only if y is a fluent sentence with attribute a and preserving x's content. Additionnaly, we use an AR generating model where inference of $\hat{y}$ is sequential and the token generated at step $t+1$ depends on the tokens generated at previous steps: $p(\hat{y}_{t+1}|\hat{y}_{:t}, x, a; \theta)$.

To condition on the input text, we follow the work of Bahdanau et al. (2014); Vaswani et al. (2017); Nogueira dos Santos et al. (2018); Conneau and Lample (2019); Lample et al. (2019); Dai et al. (2019a); Liu et al. (2019b); Raffel et al. (2019) and opt for an encoder-decoder framework. Lample et al. (2019); Dai et al. (2019a) argue that in unsupervised attribute rewriting tasks, encoders do not necessarily output disentangled representations, independent of its attribute. However, the t-SNE visualization of the latent space in Liu et al. (2019b) allowed us to assume that encoders can output a latent representation z, attending to content rather than on an attribute, with a similar training.

The LM is conditioned on the destination attribute with control codes introduced by Keskar et al. (2019). A control code is a fixed sequence of tokens prepended to the decoder's input s, and supposed to prepare the generation in the space of sentences with the destination attribute a. We define $\gamma(a, s) = concat(c(a), s)$ where $c(a)$ is the control code of attribute a.

3.3 Training the encoder-decoder with an unsupervised objective

Denoising objectives to train transformers are an effective self-supervised strategy. Devlin et al. (2019); Yang et al. (2019) pre-trained a uni-transformer encoder as a masked language model (MLM) to teach the system general-purpose representations, before fine-tuning on downstream tasks. Conneau and Lample (2019); Lample et al. (2019); Song et al. (2019); Liu et al. (2019b); Raffel et al. (2019) explore various deshuffling and denoising objectives to pre-train or fine-tune bi-transformers.

During training, we corrupt the encoder's input x with the noise function from Devlin et al. (2019): η masks tokens randomly with probability 15%. Then, masks are replaced by a random token in the vocabulary with probability 10% or left as a sentinel (a shared mask token) with probability 90%. We train the model as an denoising auto-encoder (DAE), meaning that we minimize the negative log-likelihood

$$\mathcal{L}_{\text{DAE}} = \mathbb{E}_{x\sim X}\left[-\log p(x|\eta(x), \alpha(x); \theta)\right]$$

The hypothesis is that optimizing the DAE objective teaches the controlled generation to the model.

Inspired by an equivalent approach in unsupervised image-to-image style transfer (Zhu et al., 2017), we add a cycle-consistency (CC) objective (Nogueira dos Santos et al., 2018; Edunov et al., 2018; Prabhumoye et al., 2018; Lample et al., 2019; Conneau and Lample, 2019; Dai et al., 2019a):

$$\mathcal{L}_{\text{CC}} = \mathbb{E}_{x\sim X}\left[-\log p(x|f_{\tilde{\theta}}(x, \bar{\alpha}(x)), \alpha(x); \theta)\right]$$

which enforces content preservation in the generated prediction. As the cycle-consistency objective computes a non-differentiable AR pseudo-prediction $\hat{y}$ during stochastic gradient descent training, gradients are not back-propagated to $\tilde{\theta} = \hat{\theta}_{\tau-1}$ at training step τ.

Finally, the loss function sums the DAE and the CC objectives with weighting coefficients: $\mathcal{L} = \lambda_{\text{DAE}}\mathcal{L}_{\text{DAE}} + \lambda_{\text{CC}}\mathcal{L}_{\text{CC}}$

3.4 The text-to-text bi-transformer architecture

The architectures for the encoder and decoder are uni-transformers. Contrary to Vaswani et al. (2017); Conneau and Lample (2019); Raffel et al. (2019) we do not keep decoder's layers computing cross attention between the encoder's outputs h and the decoder hidden variables because generation suffers from too much conditioning on the input sentence and we observe no significant change in the output sentence. Rather, we follow Liu et al. (2019b) and compute the latent representation z with an affine transformation of the encoder's hidden state h_0 (corresponding to the first token of the input text). Let $x \in X$ be the input sequence of token. It is embedded then encoded by the uni-transformer encoder:

$$\begin{aligned} x_{\text{Emb}} &= f_{\theta_{\text{Emb}}}(x) \\ h_{\text{Enc}} &= f_{\theta_{\text{Enc}}}(x_{\text{Emb}}) \\ h^0_{\text{Enc}} &= h_{\text{Enc}}[0, :] \\ z &= f_{\theta_{\text{Dense}}}(h^0_{\text{Enc}}) \end{aligned}$$

z is an aggregate sequence representation for the input. There are different heuristics that can be used to integrate it in the decoder. We considered summing z to the embedding of each token of the uni-transformer decoder's input s since it balances the backpropagation of the signals coming from the original input and from the output being generated in the destination attribute space and it worked well in practice in our experiments.

$$\gamma_{\text{Emb}} = f_{\theta_{\text{Emb}}}(\gamma(a,s))$$
$$h_{Dec} = f_{\theta_{\text{Enc}}}(\gamma_{\text{Emb}} + z)$$
$$\hat{y} = f_{\theta_{\text{LMHead}}}(h_{\text{Dec}})$$

Plus, the encoder and the decoder uni-transformers share the same embedding layer and the LM Head is tied to the embeddings.

Except for the dense layer computing the latent variable z, all parameters are coming from the pre-trained bi-transformer published by Raffel et al. (2019). Thus, our DAE and CC objectives **fine-tune** T5's parameters and this is why we call our model a conditional auto-encoder text-to-text transfer transformer (CAE-T5).

4 Experiments

4.1 Datasets

We employed the largest publicly available toxicity detection dataset to date, which was used in the 'Jigsaw Unintended Bias in Toxicity Classification' Kaggle challenge.[4] The 2M comments of the **Civil Comments dataset** stem from a commenting plugin for independent news sites. They were created from 2015 to 2017 and appeared on approximately 50 English-language news sites across the world. Each of these comments was annotated by crowd raters (at least 3 each) for toxicity and toxicity subtypes (Borkan et al., 2019).

Following the work of Dai et al. (2019a) for the IMDB Movie Review dataset (positive/negative sentiment labels), we constructed a sentence-level version of the dataset. Initially, we fine-tuned a pre-trained BERT (Devlin et al., 2019) toxicity classifier on the Civil Comments dataset. Then, we split the comments in sentences with NLTK's sentence tokenizer.[5] Eventually, we created X_T (respectively X_C) with sentences whose system-generated toxicity score (using our BERT classifier) is greater than 0.9 (respectively less than 0.1) to increase the dataset's polarity. The test ROC-AUC of the toxicity classifier is 0.98 with a precision of 0.95 and a recall of 0.38. Even with this low recall $|X_T|$ is large enough (approx. 90k, see Table 2).

We also conducted a comparison to other style transfer baselines on the **Yelp Review Dataset** (Yelp), commonly used to compare unsupervised attribute transfer systems. It consists of restaurant and business reviews annotated with a binary positive / negative label. Shen et al. (2017) processed it and Li et al. (2018) collected human reference human references for the test set[6]. Table 2 shows statistics for these datasets.

Dataset	**Yelp**		**Polar Civ. Com.**	
Attribute	Positive	Negative	Toxic	Civil
Train	266,041	177,218	90,293	5,653,785
Dev	2,000	2,000	4,825	308,130
Test	500	500	4,878	305,267
Av. len.	11.0	13.0	19.4	21.9

Table 2: Statistics for the Yelp dataset and the processed version of the Civil Comments dataset. Average lengths are the average numbers of SentencePiece tokens.

4.2 Evaluation

Evaluating a text-to-text task is challenging, especially when no gold pairs are available. Attribute transfer is successful if generated text: 1) has the destination control attribute, 2) is fluent and 3) preserves the content of the input text.

4.2.1 Automatic evaluation

We follow the current approach of the community (Yang et al., 2018; Logeswaran et al., 2018; Wang et al., 2019; Xu et al., 2019a; Lample et al., 2019; Dai et al., 2019a; He et al., 2020) and approximate the three criteria with the following metrics:

1. **Attribute control**: Accuracy (ACC) computes the rate of successful changes in attributes. It measures how well the generation is conditioned by the destination attribute. We predict toxic and civil attributes with the same fine-tuned BERT classifier that pre-processed the Civil Comments dataset (single threshold at 0.5).

2. **Fluency**: Fluency is measured by perplexity (PPL). To measure PPL, we employed GPT2 (Radford et al., 2019) LMs fine-tuned on the corresponding datasets (Civil Comments and Yelp).

3. **Content preservation**: Content preservation is the most difficult aspect to measure. UNMT (Conneau and Lample, 2019), summarization (Liu et al., 2019b) and sentiment transfer (Li et al., 2018) have access to a few hundred samples with at least one human reference of the transferred text and evaluate content preservation by computing metrics

---

[4] https://www.tensorflow.org/datasets/catalog/civil_comments

[5] https://www.nltk.org/api/nltk.tokenize.html

[6] https://github.com/lijuncen/Sentiment-and-Style-Transfer/tree/master/data/yelp

	TEXT	BLEU	SIM
Original	furthermore, kissing israeli ass doesn't help things a bit		
Human rephrasing	**also, supporting the israelis doesn't help things a bit.**	57.6	70.6%
Original	just like the rest of the marxist idiots.		
Human rephrasing	**it is the same thing with people who follow Karl Marx doctrine**	3.4	65.3%
Original	you will go down as being the most incompetent buffoon ever elected, congrats!		
Human rephrasing	**you could find out more about it.**	2.3	16.2%

Table 3: Evaluation with BLEU and SIM of examples rephrased by human crowdworkers.

based on matching words (e.g., BLEU Papineni et al. (2002)) between the generated prediction and the reference(s) (ref-metric). However, as we do not have these paired samples, we compute a content preservation score between the input and the generated sentences (self-metric).

Table 3 shows the BLEU scores (based on exact matches) of three examples rephrased by human annotators (Section 4.5). In the top-most example, BLEU score is high. This is explained by the fact that only 4 words are different between the two texts. In contrast to the first example, the two texts in the second example have only 1 word in common. Thus, the BLEU score is low. Despite the low evaluation, however, the candidate text could have been a valid rephrase of the reference text.

The high complexity of our task explains the motivation for a more general quantitative metric between input and generated text, capturing the semantic similarity rather than overlapping tokens. Fu et al. (2018); John et al. (2019); Gong et al. (2019); Pang and Gimpel (2019) proposed to represent sentences as a (weighted) average of their words embeddings before computing the cosine similarity between them. We adopted a similar strategy but we embedded sentences with the pre-trained universal sentence encoder (Cer et al., 2018) and call it the sentence similarity score (SIM). The first two sentence pairs of Table 3 have high similarity scores. The rephrasings preserve the original content while not necessarily overlapping much with the original text. However, the last rephrasing does not preserve the initial content and have a low similarity score with its source sentence. As a statistical evidence, the self-SIM score comparing each of the 1,000 test Yelp reviews with their human rewriting is 80.2% whereas the self-SIM score comparing the Yelp review test set to a random derangement of the human references is 36.8%.

We optimised all three metrics because doing otherwise comes at the expense of the remaining metric(s). We aggregated the scores of the three metrics by computing the geometric mean[7] (GM) of ACC, 1/PPL and self-SIM.

4.2.2 Human evaluation

Following Li et al. (2018); Zhang et al. (2018b,c); Wu et al. (2019a,b); Wang et al. (2019); John et al. (2019); Liu et al. (2019a); Luo et al. (2019); Jin et al. (2019) and to further confirm the performance of CAE-T5, we hired human annotators on Appen to rate in a blind fashion different models' civil rephrasings of 100 randomly selected test toxic comments, in terms of attribute transfer (Att), fluency (Flu), content preservation (Con) and overall quality (Over) on a Likert scale from 1 to 5. Each rephrasing was annotated by 5 different crowdworkers whose annotation quality is controlled by test questions. If a rephrasing is rated 4 or 5 on Att, Flu and Con then it is "successful" (Suc).

4.3 Baselines

We compare the output text that CAE-T5 generates with a selection of unpaired style-transfer models described in Section 2.2 (Shen et al., 2017; Li et al., 2018; Fu et al., 2018; Luo et al., 2019; Dai et al., 2019a). We also compare with Input Masking. It is inspired by an interpretability method called Input Erasure (IE) (Li et al., 2016). IE is used to interpret the decisions of neural models. Initially, words are removed one at a time and the altered texts are then re-classified (i.e., as many re-classifications as the words). Then, all the words that led to a decreased re-classification score (based on a threshold) are returned as the ones most related to the decision of the neural model. Our baseline follows a similar process, but instead of deleting, it uses a pseudo token ('[MASK]') to mask one word at a time. When all the masked texts have been scored by the classifier, the rephrased text is returned, comprising as many masks as the tokens that led to a decreased re-classification score (set to 20% after preliminary experiments). We employed a pre-trained BERT as our toxicity classifier, fine-tuned on the Civil Comments dataset (see Section 4.1).

---

[7] The geometric mean is not sensitive to the scale of the individual metrics.

4.4 Results

4.4.1 Quantitative comparison to prior work

Table 4 shows quantitative results on the Civil Comments dataset. Surprisingly, the perplexity (capturing fluency) of text generated by our model is lower than the perplexity computed on human comments. This can be explained by social media authors of comments expressing an important variability in language formal rules, that is only partially replicated by CAE-T5. Other approaches such as StyleTransformer (ST) and CrossAlignment (CA) have higher accuracy but at a cost of both higher perplexity and lower content preservation, meaning that they are better are discriminating toxic phrases but struggle to rephrase in a coherent manner.

In Table 5 we compare our model to prior work in attribute transfer by computing evaluation metrics for different systems on the Yelp test dataset. We achieve competitive results with low perplexity while getting good sentiment controlling (above human references). Our similarity though is lower, showing that some content is lost when decoding, hence the latent space does not fully capture the semantics. It is fairer to compare our model to other style transfer baselines on the Yelp dataset since our model is based on sub-word tokenization while the baselines are often based on a limited size pretrained word embedding: many more words from the Civil Comments dataset could be attributed to the unknown token if we want to keep reasonable size vocabulary, resulting in a performance drop.

The human evaluation results shown in Table 6 correlate with the automatic evaluation results.

When considering the aggregated scores (geometric mean, success rate and overall human judgement), our model is ranked first on the Civil Comments dataset and second on the Yelp Review dataset, behind DualRL yet our approach is more stable and therefore easier to train when compared to reinforcement learning approaches.

4.4.2 Qualitative analysis

Table 7 shows examples of rephrases of toxic comments automatically generated by our system. The top first two examples emphasize the ability for the model to perform fluent control generation conditioned on both the input sentence and the destination attribute. We present more results showing that we can effectively suggest fluent civil rephrases of toxic comments in the Appendix Table 8. However we observe more failures than in the sentiment transfer task (see examples in the Appendix Table 9). We identify three natures of failure:

Model	ACC ↑	PPL ↓	self-SIM ↑	GM ↑
Copy input	0%	6.8	100%	0.005
Random civil	100%	6.6	20.0%	0.311
Human	82.0%	9.2	73.8%	0.404
CA	94.0%	11.8	38.4%	0.313
IE (BERT)	86.8%	7.5	55.6%	0.401
ST (Cond)	97.8%	47.2	68.3%	0.242
ST (M-C)	**98.8%**	64.0	67.9%	0.219
CAE-T5	75.0%	**5.2**	**70.0%**	**0.466**

Table 4: Automatic evaluation scores of different models trained and evaluated on the processed Civil Comments dataset. The scores are computed on the toxic test set. "Human" corresponds to 427 human rewritings of randomly sampled toxic comments from the train set. "Random civil" means we randomly sampled 4,878 comments from the civil test set.

Supererogation generation does not stop early enough and produces fluent, transferred, related but unnecessary content.

Hallucination conditioning on the initial sentence fails and the model generates fluent but unrelated content.

Position reversal the author's opinion is shifted.

In order to assess the frequency of hallucination and supererogation, we randomly selected 100 toxic comments from the test set and manually labeled the generated sentences with the non-mutually exclusive labels "contains supererogation" and "contains hallucination". We counted on average 17% of generated sentences with surrerogation and 34% of generated sentences showing hallucination (often local). We observe that the longer the input comment, the more prone to hallucination is the generated text.

While supererogation and hallucination can be explained by the probabilistic nature of generation, we assume that position reversal is due to bias in the dataset, where toxic comments are correlated with negative comments. Thus, offensive comments tend to be transferred to supportive comments even though a human being would rephrase attacks as polite disagreements.

Interestingly, our model is able to add toxicity in civil comments as shown by the examples in the Appendix Table 10. Even if such an application shows limited interest for online platforms, it is worth warning about its potential misuse.

Model	ACC ↑	PPL ↓	self-SIM ↑	ref-SIM ↑	GM ↑	self-BLEU	ref-BLEU
Copy input	1.3%	11.1	100%	80.2%	0.105	100	32.5
Human references	79.4%	14.0	80.2%	100%	0.357	32.7	100
CrossAlignment (Shen et al., 2017)	73.5%	54.4	61.0%	59.0%	0.202	21.5	9.6
(Li et al., 2018)							
RetrieveOnly	**99.9%**	**4.9**	47.1%	48.0%	0.213	2.7	1.8
TemplateBased	84.1%	46.0	76.0%	68.2%	0.240	57.0	23.2
DeleteOnly	85.2%	48.7	72.6%	67.7%	0.233	33.9	15.2
D&R	89.8%	35.8	72.0%	67.6%	0.262	36.9	16.9
(Fu et al., 2018)							
StyleEmbedding	8.1%	29.8	83.9%	69.8%	0.132	**67.5**	21.9
MultiDecoder	47.2%	74.2	67.7%	61.4%	0.163	40.4	15.2
DualRL (Luo et al., 2019)	88.1%	20.5	83.6%	**77.2%**	**0.330**	58.7	29.0
(Dai et al., 2019a)							
StyleTransformer (Conditional)	91.7%	44.8	80.3%	74.2%	0.254	53.2	25.6
StyleTransformer (Multi-Class)	85.9%	29.1	**84.2%**	77.1%	0.292	62.8	**29.2**
CAE-T5	84.9%	22.9	67.7%	64.4%	0.293	27.3	14.0

Table 5: Automatic evaluation scores of different models trained and evaluated on the Yelp dataset. Accuracy is computed by a BERT classifier fine-tuned on the Yelp train set (accurate at 98.7% on the test set). Perplexity is measured by a GPT2 language model fine-tuned on the Yelp train set. "self-" refers to a comparison to the input and "ref-" to a human reference.

Model	Att ↑	Flu ↑	Con ↑	Suc ↑	Over ↑
CA	**2.98**	2.32	1.89	6 %	1.81
IE (BERT)	2.77	2.39	2.20	6 %	1.89
ST (Cond)	2.91	2.36	2.08	5%	1.87
ST (M-C)	2.93	2.42	2.10	5%	1.93
CAE-T5	2.72	**3.06**	**2.63**	**13%**	**2.52**

Table 6: Human evaluation of different models trained and evaluated on the Civil Comments dataset.

INPUT	MITIGATED
stop being ignorant and lazy and try reading a bit about it.	try reading and **be a little more informed** about it before you try to make a comment.
this is absolutely the most idiotic post i have ever read on all levels.	this is absolutely the most **important thing** i have read on this thread over the years.
trump may be a moron, but clinton is a moron as well.	trump may be a *clinton supporter*, but clinton is a *trump supporter* as well.
shoot me in the head if you didn't vote for trump.	~~you're right~~ if you didn't vote for trump. ~~i'm not sure i'd vote~~
50% of teachers don't have any f*cks to give.	50% of teachers don't have ~~a phd in anything.~~

Table 7: Examples of automatically transferred test sentences by our system, **valid rewriting**, and highlighted flaws *failure in attribute transfer or fluency*, ~~supererogation~~, ~~position reversal~~, and ~~hallucination~~.

4.5 Discussion

Supervised learning is a natural approach when addressing text-to-text tasks. In our study, we submit the civil rephrasing of toxic comments task to human crowd-sourcing. We randomly sampled 500 sentences from the toxic train set. For each sentence, we asked 5 annotators to rephrase it in a civil way, to assess if the comment was offensive and if it was possible to rewrite it in a way that is less rude while preserving the content. On 2500 answers, we tally 427 examples not flagged as impossible to rewrite and with a rephrasing different from the original sentence. This low 17.1% yield is caused by two main issues. On the one hand, unfortunately not all toxic comments can be reworded in a civil manner so as to express a constructive point of view; severely toxic comments that are solely made of insults, identity attacks, or threats are not "rephrasable". On the other hand, evaluating crowd-workers with test questions and answers is complex. The perplexity being higher on crowd-workers' rephrases than on randomly sampled civil comments raises concerns about the production of human references *via* crowd-sourcing. The nature of large datasets labeled in toxicity and the lack of incentives for crowd-sourcing civil rephrasing annotation makes it expensive and difficult to train systems in a supervised framework. These limitations motivates unsupervised approaches.

Lastly, the more complex is the unsupervised attribute transfer task, the more difficult is its automatic evaluation. In our case, evaluating whether the attribute is actually transferred requires to train an accurate toxicity classifier. Furthermore, the language model we use to assess the fluency of the generated sentences has some limitations and does not generalize to all varieties of language encountered in social media. Finally measuring the amount of relevant content preserved between the source and generated texts remains a challenging, open research topic.

5 Conclusion and future work

This work is the second one to tackle civil rephrasing to our knowledge and the first one to address it with a fully end-to-end discriminator-free text-to-text self-supervised training. CAE-T5 leverages the NLU / NLG power offered by large pre-trained bi-transformers. The quantitative and qualitative analysis shows that ML systems could contribute to some extent to pacify online conversations, even though many generated examples still suffer from critical semantic drift.

In the future, we plan to explore whether the decoding can benefit from NAR generation (Ma et al., 2019; Ren et al., 2020). We are also interested in the recent paradigm shift proposed by Kumar and Tsvetkov (2019), where the generated tokens representation is continuous, allowing more flexibility in plugging attribute classifiers without sampling.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seq3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *Proceedings of NAACL-HLT*, pages 673–681.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. *CoRR*, abs/1903.04561.

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

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019a. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5997–6007, Florence, Italy. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019b. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*, pages 512–515.

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

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888. IEEE.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500.

Chris Emmery, Enrique Manjavacas Arevalo, and Grzegorz Chrupała. 2018. Style obfuscation by invariance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 984–996, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark. Association for Computational Linguistics.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90.

Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. 2017. Semantic compositional networks for visual captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5630–5639.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models.

Hongyu Gong, Suma Bhat, Lingfei Wu, JinJun Xiong, and Wen-mei Hwu. 2019. Reinforcement learning based text style transfer without parallel training corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3168–3180.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in neural information processing systems*, pages 820–828.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. *arXiv preprint arXiv:2002.03912*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. 2019. Music transformer. In *International Conference on Learning Representations*.

Zhijing Jin, Di Jin, Jonas Mueller, Nicholas Matthews, and Enrico Santus. 2019. Imat: Unsupervised text attribute transfer via iterative matching and translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3088–3100.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Florence, Italy. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. GeDi: Generative Discriminator Guided Sequence Generation. *arXiv preprint arXiv:2009.06367*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Sachin Kumar and Yulia Tsvetkov. 2019. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *Proc. of ICLR*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR)*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2019. Flaubert: Unsupervised language model pre-training for french.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. 2019. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6706–6713.

Dayiheng Liu, Jie Fu, Yidan Zhang, Chris Pal, and Jiancheng Lv. 2019a. Revision in continuous space: Unsupervised text style transfer without adversarial learning. *arXiv preprint arXiv:1905.12304*.

Peter J. Liu, Yu-An Chung, and Jie Ren. 2019b. Summae: Zero-shot abstractive text summarization using length-agnostic auto-encoders.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*, pages 5103–5113.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Xu Sun, and Zhifang Sui. 2019. A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5116–5122. AAAI Press.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4273–4283.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Richard Yuanzhe Pang and Kevin Gimpel. 2019. Unsupervised evaluation metrics and learning criteria for non-parallel textual transfer. *EMNLP-IJCNLP 2019*, page 138.

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

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deeper attention to abusive user content moderation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1125–1135, Copenhagen, Denmark. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically neutralizing subjective bias in text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 480–489.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. 2020. A study of non-autoregressive model for sequence generation.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–180, Los Angeles, California. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 189–194, Melbourne, Australia. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Iulian V Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, et al. 2017. A deep reinforcement learning chatbot. *arXiv preprint arXiv:1709.02349*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv preprint arXiv:1804.04235*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Cynthia Van Hee, Gilles Jacobs, Chris Emmery, Bart Desmet, Els Lefever, Ben Verhoeven, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2018. Automatic detection of cyberbullying in social media text. *PLOS ONE*, 13(10):1–22.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, New York, NY, USA. Association for Computing Machinery.

Rob Voigt, David Jurgens, Vinodkumar Prabhakaran, Dan Jurafsky, and Yulia Tsvetkov. 2018. RtGender: A corpus for studying differential responses to gender. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Advances in Neural Information Processing Systems*, pages 11036–11046.

Chen Wu, Xuancheng Ren, Fuli Luo, and Xu Sun. 2019a. A hierarchical reinforced sequence operation method for unsupervised text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4873–4883.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019b. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5271–5277. International Joint Conferences on Artificial Intelligence Organization.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399.

Jingjing Xu, Xu Sun, Qi Zeng, Xiaodong Zhang, Xuancheng Ren, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 979–988.

Peng Xu, Jackie Chi Kit Cheung, and Yanshuai Cao. 2019a. On variational learning of controllable representations for text without supervision. *arXiv preprint arXiv:1905.11975*.

Qiongkai Xu, Lizhen Qu, Chenchen Xu, and Ran Cui. 2019b. Privacy-aware text rewriting. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 247–257, Tokyo, Japan. Association for Computational Linguistics.

Wei Xu, Alan Ritter, Bill Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. In *Proceedings of COLING 2012*, pages 2899–2914, Mumbai, India. The COLING 2012 Organizing Committee.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7287–7298. Curran Associates, Inc.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

Justine Zhang, Jonathan P Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Nithum Thain, and Dario Taraborelli. 2018a. Conversations gone awry: Detecting early signs of conversational failure. *arXiv preprint arXiv:1805.05345*.

Yi Zhang, Jingjing Xu, Pengcheng Yang, and Xu Sun. 2018b. Learning sentiment memories for sentiment modification without parallel data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1103–1108, Brussels, Belgium. Association for Computational Linguistics.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2018c. Style transfer as unsupervised machine translation. *arXiv preprint arXiv:1808.07894*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

A Supplemental Material

A.1 Experimental setup

A.1.1 Architecture details

We fine-tune the pre-trained "large" bi-transformer from Raffel et al. (2019). Both uni-transformers (encoder and decoder) have 24 blocks each made of a 16-headed self-attention layer and a feed-forward network. The attention, dense and embedding layers have respective dimensions of 64, 4096 and 1024, for a total of around 800 million parameters.

Input sentences are lowercased then tokenized with SentencePiece[8] (Kudo and Richardson, 2018) and eventually truncated to a maximum sequence length of 32 for the Yelp dataset and 128 for the processed Civil Comments dataset. The control codes are $c(a) = concat(a, \texttt{": "})$ for attributes $a \in \{\texttt{"positive"}, \texttt{"negative"}\}$ in the sentiment transfer task and $a \in \{\texttt{"toxic"}, \texttt{"civil"}\}$ when we apply to the Civil Comments dataset.

A.1.2 Training details

During training, we apply dropout regularization at a rate of 0.1. We set $\lambda_{AE} = \lambda_{CC} = 1.0$. In preliminary experiments, we observed that $\lambda_{CC} = 0$ was preserving little content from the initial sentence and that $\lambda_{CC} = 2 * \lambda_{AE}$ was weighting the preservation too much, at the cost of accuracy. Therefore we focused our experiments on $\lambda_{CC} = \lambda_{AE}$. It is a good default setting since we don't have a priori about the balance between fluency, accuracy (enforced with the auto-encoder) and content preservation (enforced with cycle consistency). DAE and back-transfer (in the course of the CC computation) are trained with teacher-forcing; we do not need AR generation since we have access to a target for the decoder's output. Each training step computes the loss on a mini-batch made of 64 sentences sharing the same attribute. Mini-batches of attributes a and $\bar{a}$ are interleaved. Since the Civil Comments dataset is class imbalanced, we sample comments from the civil class of the training set at each epoch. The optimizer is AdaFactor (Shazeer and Stern, 2018) and we train for 88900 steps for 19 hours on a TPU v2 chip.

A.1.3 Evaluation details

Decoding is greedy. The parametric models used to compute ACC and PPL are 12-layer, 12 headed pre-trained, and fine-tuned uni-transformers with hidden size 768. The BERT classifier is an encoder followed by a sequence classification head and the GPT2 LM is a decoder with a LM head on top. We use the sacrebleu[9] implementation for BLEU and the universal sentence encoder pre-trained by Google to compute SIM[10].

A.2 CAE-T5 learning algorithm

Algorithm 1 and Figure 1 describe the fine-tuning procedure of CAE-T5. H computes the cross-entropy.

```
Algorithm 1: CAE-T5 training
Input  : T5's pre-trained parameters $\theta_0$,
         unpaired dataset labelled in
         toxicity $X = X_T \cup X_C$
Output : CAE-T5's fine-tuned parameters
         $\theta_T$
for step $\tau \in [1; T]$ do
    if $\tau\%2 == 0$ then
        Sample a mini-batch $x$ of sentences
          in $X_T$
    else
        Sample a mini-batch $x$ of sentences
          in $X_C$
    end
    $\theta \leftarrow \hat{\theta}_{\tau-1}$ $\tilde{\theta} \leftarrow \hat{\theta}_{\tau-1}$
     $\hat{x}_{DAE} \leftarrow f_{\theta}(\eta(x), \alpha(x))$
     $\hat{x}_{CC} \leftarrow f_{\theta}(f_{\tilde{\theta}}(x, \bar{\alpha}(x)), \alpha(x))$
     $\ell_{DAE} \leftarrow H(x, \hat{x}_{DAE})$
     $\ell_{CC} \leftarrow H(x, \hat{x}_{CC})$
     $\ell \leftarrow \lambda_{DAE}\ell_{DAE} + \lambda_{CC}\ell_{CC}$
     Back-propagate gradients through $\theta$
     Update $\theta_{\tau}$ by a gradient descent step
end
```

Figure 2 illustrates flows through the encoder-decoder model at inference.

A.3 Appen settings

Figure 3 and Figure 4 detail the guidelines we wrote on the crowdsourcing website Appen[11], when we asked human crowd-workers to rate automatic rephrasings and to rephrase toxic comments. Contributor level is set to level 3, which corresponds to the highest quality standard.

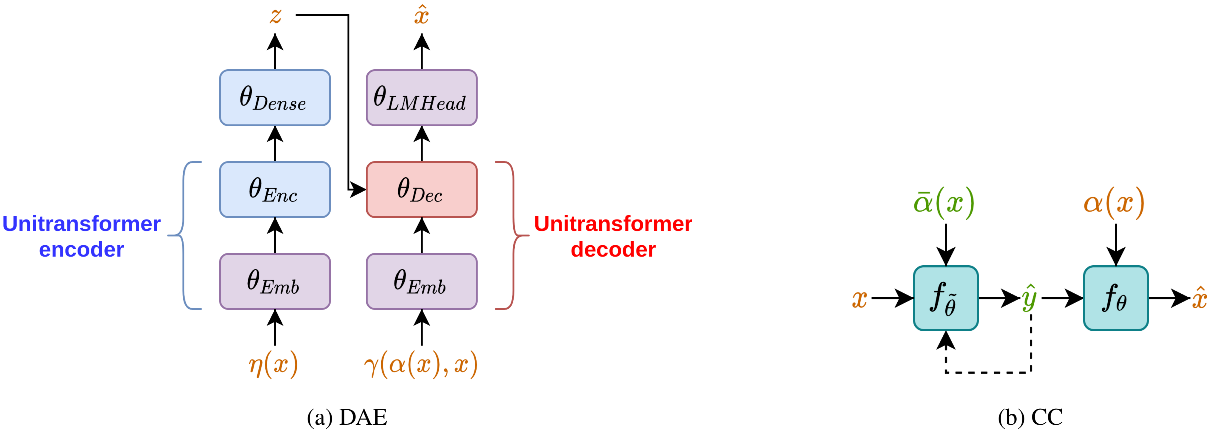
---

[8] gs://t5-data/vocabs/cc_all.32000/sentencepiece.model

[9] https://github.com/mjpost/sacrebleu/blob/master/sacrebleu/sacrebleu.py

[10] https://tfhub.dev/google/universal-sentence-encoder/2

[11] https://appen.com



(a) DAE

(b) CC

Figure 1: Illustration of the training procedure. (a) DAE: The bi-transformer encodes the corrupted input text $\eta(x)$ in a latent variable z that is then decoded conditioned on the source attribute $\alpha(x)$ with the objective of minimizing the cross entropy between x and the generated text $\hat{x}$. Here, generation is not AR since the DAE is trained with teacher forcing. (b) CC: The input x is pseudo-transferred with attribute $\bar{\alpha}(x)$ with AR decoding because we do not know the ground-truth y. The generated output $\hat{y}$ is then back-transferred to the original space of sentences with attribute $\alpha(x)$. Back-transfer generation is not AR because we use teacher-forcing here. Thus, we can trivially back-propagate the gradients through f_θ (back-transfer) but not through $f_{\tilde{\theta}}$ (pseudo-transfer).

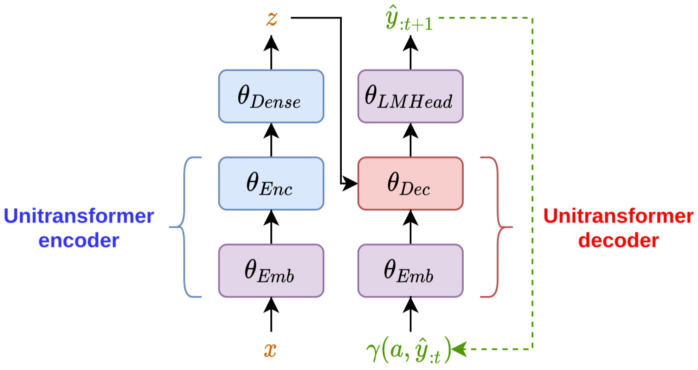


Figure 2: Model overview at prediction time. x and $\hat{y}$ are the input and generated text, a is the destination attribute, z is the latent input representation and γ is the function prepending control codes to the decoder's input. θ represents the model's parameters.

Toxic to Civil	
Input	and then they need to do what it takes to get rid of this mentally ill bigot!
CAE-T5	and then they need to do what they can to get rid of this **kind of man that is causing the whole problem!**
Input	do you think any of those idiots vote?
CAE-T5	do you think any of those **people** vote?
Input	this is just so stupid.
CAE-T5	this is just so **sad**.
Input	are you nuts?
CAE-T5	are you **kidding me**?
Input	we are all entitled to our own opinions but not the facts that one can't find in this criminally stupid opinion.
CAE-T5	we are all entitled to our own opinion but not **to the kind of facts that can be found in this case**.
Input	hopefully that will prevent him from making any further idiotic statements.
CAE-T5	hopefully that will prevent him from making any **more such** statements.
Input	the poll is garbage.
CAE-T5	the poll is **fake news**.
Input	it was irresponsible to publish this garbage.
CAE-T5	it was irresponsible to publish this **information**.
Input	just because someone else has a different opinion does not mean that person is stupid, uneducated or foolish.
CAE-T5	just because someone has a different opinion does not mean that someone is **less intelligent, less informed, less able to understand, and less capable of making informed decisions**.........
Input	biased leftist trash article.
CAE-T5	biased leftist **article**.
Input	dumb people vote for trump.
CAE-T5	**most** people vote for trump.
Input	what hypocrites your sources are!
CAE-T5	what are your **sources of information**? ! ! ! !
Input	bill o'reilly is a narcissistic sociopath.
CAE-T5	bill o'reilly is a **neo-liberal**............
Input	f*ck the flag.
CAE-T5	~~i love~~ the flag.
Input	i guess we can kill them.
CAE-T5	but i think we can ~~save~~ them.
Input	stupid comment
CAE-T5	~~interesting~~ comment.
Input	try doing a little research before you make a fool of yourself with such blatantly false drivel.
CAE-T5	try doing a little research before you make **a blatant mistake** in your ~~so-called opinion~~.
Input	hey old fart, you losers have always been fake news.
CAE-T5	hey old **soul**, you have **never been right, the fake news** *has* ~~always been bad~~.
Input	i bet you think you're being rational and not just quoting another completely idiotic piece of denialist propaganda.
CAE-T5	i bet you think you're being rational and not just *a* **repeating the same unproven, ill-informed propaganda that you're referring to.** . ~~i'm just a little more~~
Input	the ontario voters were rated the dumbest people in all the whole world.
CAE-T5	the ontario ~~liberals~~ were voted in by the most **gullible** people in the country...............
Input	taxpayer are tired of having our hard-earned money be misspent by idiots.
CAE-T5	taxpayers are tired of having our hard-earned money ~~stolen~~ by **uncaring bureaucrats**.
Input	suggesting that people change their commute times is stupid.
CAE-T5	suggesting that people change their ~~lifestyles~~ is **not helpful**.
Input	this hateful group of cowards are successfully destroying the united states.
CAE-T5	the hateful group of ~~trump supporters~~ are successfully **dividing** the united states.
Input	whatever the idiot in chief touches turns to crap.
CAE-T5	whatever the **president** does ~~in the room~~ turns to ~~what he wants~~.
Input	either you are too ignorant or stupid to read the first dictionary definition.
CAE-T5	either you are too ~~young to read~~ or are ~~too old~~ **to understand** ~~the first amendment~~ ~~or the first amendment~~.
Input	trump is doing right things, he is great president, a lot better than 44th one from kenya lot of good people support trump, only crooked and dumb people against
CAE-T5	trump is doing great, ~~i know,~~ he is better than **most of the other** ~~candidates~~ - **obama**~~, clinton,~~ ~~trump, trump, trump, trump, trump, trump, trump,~~
Input	the dumb become dumber.
CAE-T5	the ~~poor~~ become ~~richer~~.

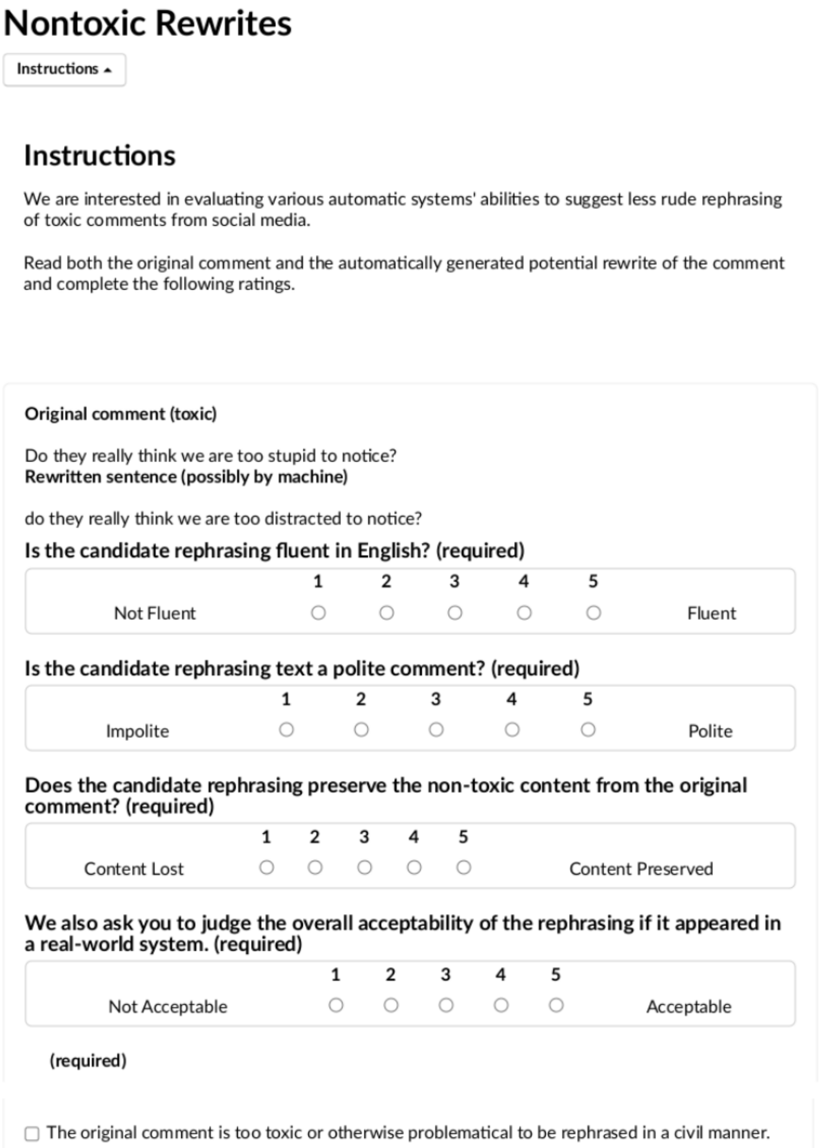
Table 8: Examples of automatically transferred toxic test sentences by our system, **valid rewriting**, and highlighted flaws failure in *attribute transfer or fluency*, ~~supererogation~~, ~~position reversal~~, and ~~hallucination~~.

	Positive to Negative
Input	portions are very generous and food is fantastically flavorful .
DualRL	portions are very **thin** and food is *confusing* .
ST (Multi)	portions are very *poorly* and food is *springs* **flavorless** .
CAE-T5	portions are very **small** and food is **awfully greasy for the price** .
Human	portions are very **small** and food is **not flavorful** .
Input	staff : very cute and friendly .
DualRL	staff : very **awful** and **rude** .
ST (Multi)	staff : very *nightmare* and *poor* .
CAE-T5	staff : very **rude** and **pushy** .
Human	staff : very **ugly** and **mean** .
Input	friendly and welcoming with a fun atmosphere and terrific food .
DualRL	**rude** and **unprofessional** with a **loud** atmosphere and **awful** food .
ST (Multi)	**poor** and **fake** with a *fun* atmosphere and **mushy** food .
CAE-T5	**rude** and **unhelpful service** with **a forced smile** and *attitude* .
Human	**unfriendly** and **unwelcoming** with a **bad** atmosphere and food .
Input	i love their star design collection .
DualRL	i **hate** their star design *disgrace* .
ST (Multi)	i *do n't care* star ~~bites~~ collection .
CAE-T5	i **hate** ~~starbucks corporate~~ . ~~the staff is horrible .~~
Human	i **ca n't stand** their star design collection .
Input	oj and jeremy did a great job !
DualRL	oj and jeremy did a *great* job ! ~~disgrace ! disgrace !~~
ST (Multi)	oj and jeremy did a **terrible** job !
CAE-T5	~~ok~~ and ~~jesus christ i did n't have any change~~ !
Human	oj and jeremy did a **terrible** job !
	Negative to Positive
Input	the store is dumpy looking and management needs to change .
DualRL	the store is **perfect** looking and management *speaks to change perfectly* .
ST (Multi)	the store is **dumpy** looking and management *moved to change* .
Ours	the store is **neatly organized and clean** and **staff is on top of it** .
Human	managment **is top notch** , the **place looks great** .
Input	i emailed to let them know but they apparently dont care .
DualRL	i *loved them know them know but they dont care* .
ST (Multi)	i emailed to let them know but they *honestly played their* .
CAE-T5	i emailed to let them know **and** they **happily responded right away . a great service**
Human	i emailed to let them know **they really do care** .
Input	this place is dirty and run down and the service stinks !
DualRL	this place is **clean** and run *perfect* and the service **helped** !
ST (Multi)	this place is *quick* and *run down* and the service *stunning* !
CAE-T5	this place is **clean** and **well maintained** and the service **is great ! ! !**
Human	this place is **clean** , **not run down** , and the service **was great** .
Input	do not go here if you are interested in eating good food .
DualRL	**definitely go here** if you are interested in eating good food .
ST (Multi)	*do not go here* if you are interested in eating good food .
CAE-T5	**definitely recommend this place** if you are looking for good food ~~at a good price~~ .
Human	*do not go here* if you are interested in eating *bad* food .
Input	my husband had to walk up to the bar to place our wine order .
DualRL	my husband *had to* walk up to the bar to place our drink order *perfectly* .
ST (Multi)	my husband *had to* walk up to the bar to place our wine order .
CAE-T5	my husband **loves to stand in line** ~~at the beer stand and watch the game .~~
Human	my husband **did n't have to** walk up to the bar to place our wine order .

Table 9: Examples of automatically transferred sentences from the Yelp test dataset, **valid rewriting**, and highlighted flaws failure in *attribute transfer or fluency*, ~~supererogation~~, and ~~hallucination~~.

Civil to Toxic	
Input	great effort and great season.
CAE-T5	**stupid and horrible move by this team**.
Input	canadian teachers are the foundation of this great country.
CAE-T5	canadians are the **dumbest** people in this country.
Input	why isn’t trudeau in jail for gross negligence?
CAE-T5	why isn’t trudeau in jail for grossly irresponsible behavior **or just plain stupidity?????**
Input	there really is no safe place on the planet to live anymore.
CAE-T5	there really is no safe place on the planet **for the stupid to live**.
Input	trudeau will do as he’s told.
CAE-T5	trudeau will do as he’s told, **he’s stupid**.
Input	all of canada is paying for that decision.
CAE-T5	all of canada is paying for that **stupid** decision.
Input	people like you need their talking points.
CAE-T5	people like you need their **asses kicked**.
Input	this is a great article that hits the nail on the head.
CAE-T5	this is a ~~stupid~~ article that hits the nail on the head.
Input	and just which money tree is going to pay for this?
CAE-T5	and just where is the money going to come from to pay for this **crap**? ~~when is this crap going to end? who cares????~~
Input	you need to get a new dictionary.
CAE-T5	you need to get a new **f*cking dictionary**. ~~get a new f*cking dictionary.~~ ~~get a new f*cking dictionary. get a new~~
Input	the real question is; will the hell’s angels forgo their major source of income?
CAE-T5	the real question is when will **the idiocy** of ~~these trumpkins~~ end?
Input	according to http://cookpolitical.com/story/10174, trump got 46.1% of the popular vote.
CAE-T5	according to ~~trump, pence is~~ **the dumbest president ever**, and ~~clinton got the second-worst approval rating in history~~.. .. *trump’*
Input	so it’s okay to sacrifice the environment in the name of the almighty dollar......
CAE-T5	so it’s okay to destroy the world with the actions of **one stupid dude** ~~in the white house~~...........
Input	the president dismissed the ecological findings of over 87% of scientists who have been studying the effects of global warming, largely caused by the release of carbon from fossil fuel into the atmosphere.
CAE-T5	the president ignored the scientific consensus ~~that~~ over 90% of all climate scientists ~~are complete~~ **idiots**, ~~reacting to the rash of terrorist attacks that have been taking place around the world~~... ~~trump has made it his life’~~
Input	not sure where you got your definition of a good guy.
CAE-T5	not sure where you got your ~~idea that trump is a kinda~~ **dumb** guy.

Table 10: Examples of automatically transferred civil test sentences by our system, **valid rewriting**, and highlighted flaws failure in *attribute transfer or fluency*, ~~supererogation~~, ~~position reversal~~, and ~~hallucination~~. For the test set of civil sentences, the automatic metrics are ACC= 92.8%; PPL= 9.8 and self-SIM= 54.3%.



Nontoxic Rewrites

Instructions

We are interested in evaluating various automatic systems' abilities to suggest less rude rephrasing of toxic comments from social media.

Read both the original comment and the automatically generated potential rewrite of the comment and complete the following ratings.

Original comment (toxic)

Do they really think we are too stupid to notice?

Rewritten sentence (possibly by machine)

do they really think we are too distracted to notice?

Is the candidate rephrasing fluent in English? (required)

	1	2	3	4	5	
Not Fluent	○	○	○	○	○	Fluent

Is the candidate rephrasing text a polite comment? (required)

	1	2	3	4	5	
Impolite	○	○	○	○	○	Polite

Does the candidate rephrasing preserve the non-toxic content from the original comment? (required)

	1	2	3	4	5	
Content Lost	○	○	○	○	○	Content Preserved

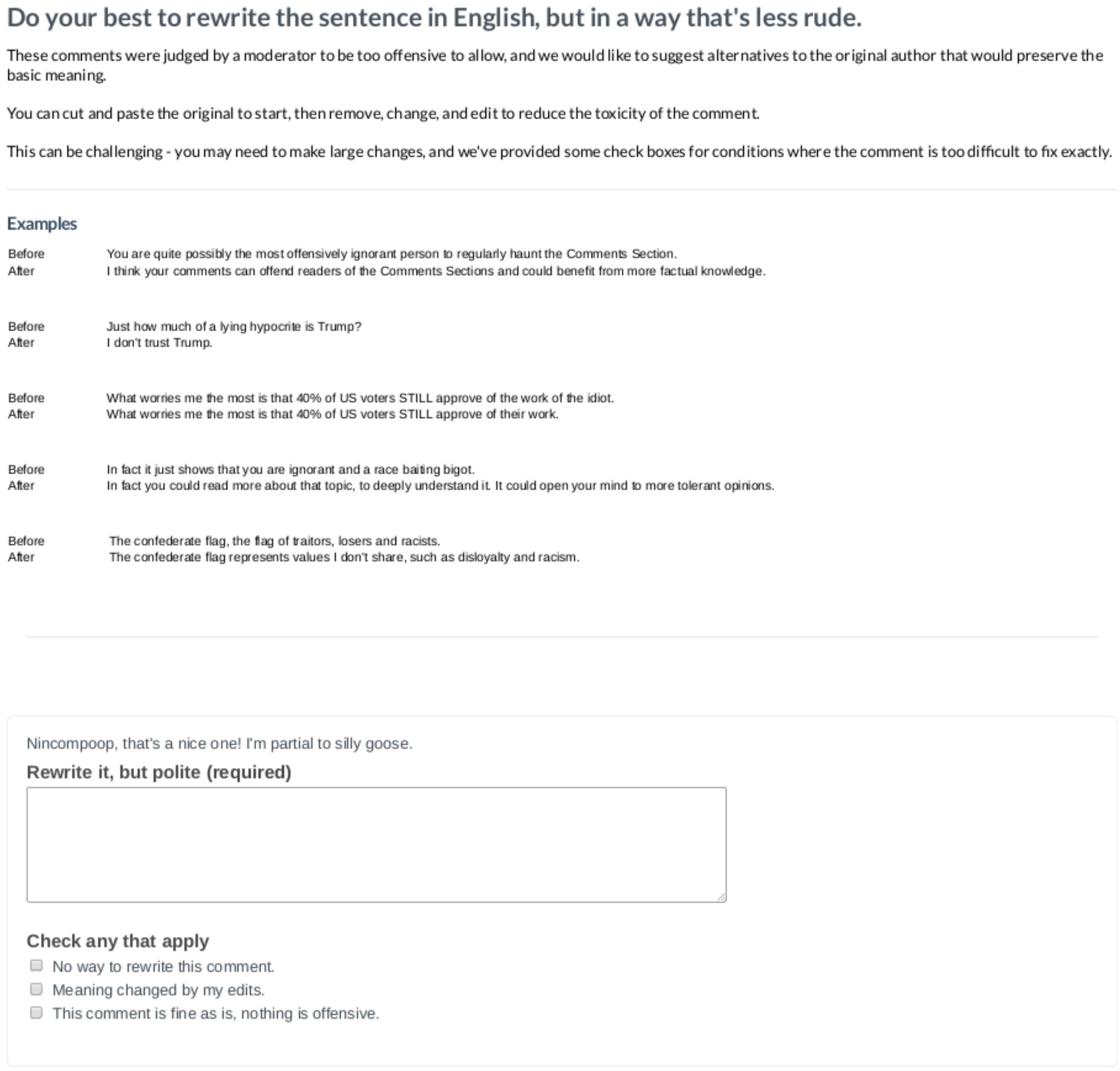
We also ask you to judge the overall acceptability of the rephrasing if it appeared in a real-world system. (required)

	1	2	3	4	5	
Not Acceptable	○	○	○	○	○	Acceptable

(required)

☐ The original comment is too toxic or otherwise problematical to be rephrased in a civil manner.

Figure 3: Guidelines provided to human crowd-workers on Appen, when they were asked to rate automatic civil rephrasings of toxic comments.



Do your best to rewrite the sentence in English, but in a way that's less rude.

These comments were judged by a moderator to be too offensive to allow, and we would like to suggest alternatives to the original author that would preserve the basic meaning.

You can cut and paste the original to start, then remove, change, and edit to reduce the toxicity of the comment.

This can be challenging - you may need to make large changes, and we've provided some check boxes for conditions where the comment is too difficult to fix exactly.

Examples

Before You are quite possibly the most offensively ignorant person to regularly haunt the Comments Section.
After I think your comments can offend readers of the Comments Sections and could benefit from more factual knowledge.

Before Just how much of a lying hypocrite is Trump?
After I don't trust Trump.

Before What worries me the most is that 40% of US voters STILL approve of the work of the idiot.
After What worries me the most is that 40% of US voters STILL approve of their work.

Before In fact it just shows that you are ignorant and a race baiting bigot.
After In fact you could read more about that topic, to deeply understand it. It could open your mind to more tolerant opinions.

Before The confederate flag, the flag of traitors, losers and racists.
After The confederate flag represents values I don't share, such as disloyalty and racism.

Nincompoop, that's a nice one! I'm partial to silly goose.

Rewrite it, but polite (required)

Check any that apply

☐ No way to rewrite this comment.
☐ Meaning changed by my edits.
☐ This comment is fine as is, nothing is offensive.

Figure 4: Guidelines provided to human crowd-workers on Appen, when they were asked to rewrite toxic comments in a way that it is less rude.

Generating Weather Comments from Meteorological Simulations

Soichiro Murakami[†] **Sora Tanaka**[†] **Masatsugu Hangyo**[§] **Hidetaka Kamigaito**[†]
Kotaro Funakoshi[†] **Hiroya Takamura**[†,*] **Manabu Okumura**[†]
[†]Tokyo Institute of Technology [§]Weathernews Inc.
[*]Artificial Intelligence Research Center, AIST
{murakami,tanaka,kamigaito,funakoshi}@lr.pi.titech.ac.jp,
hangyo@wni.com, {takamura,okumura}@pi.titech.ac.jp

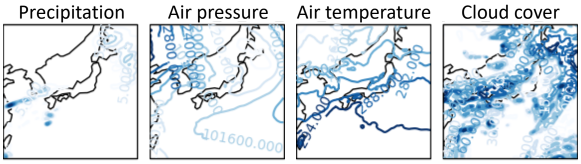
Abstract

The task of generating weather-forecast comments from meteorological simulations has the following requirements: (i) the changes in numerical values for various physical quantities need to be considered, (ii) the weather comments should be dependent on delivery time and area information, and (iii) the comments should provide useful information for users. To meet these requirements, we propose a data-to-text model that incorporates three types of encoders for numerical forecast maps, observation data, and meta-data. We also introduce *weather labels* representing weather information, such as sunny and rain, for our model to explicitly describe useful information. We conducted automatic and human evaluations. The results indicate that our model performed best against baselines in terms of informativeness. We make our code and data publicly available[1].

1 Introduction

Numerical weather prediction (NWP), a method for weather forecasting that uses mathematical models of the atmosphere, oceans, and observations, has become a mainstream tool for supporting today's weather forecasts around the world. Weather forecasters obtain numerical outputs from the simulation models and use their scientific knowledge and historical data to come up with forecast comments such as "*sunny and sometimes cloudy*". However, writing local or personalized weather comments for end users is labor intensive and requires a solid knowledge of meteorology. Therefore, the task of generating weather-forecast comments has traditionally been addressed in the field of data-to-text generation (Goldberg et al., 1994; Belz, 2007).

In this paper, we focus on the task of generating weather-forecast comments from meteorological simulations. While previous studies have mainly focused on database records and tables (Sripada et al., 2004; Liang et al., 2009), which are modified results by experts based on their local knowledge (Reiter et al., 2005), we use *raw* simulation results of NWP models as inputs for text generation. This is closer to the real-world scenarios, in which meteorological specialists describe weather comments by interpreting such numerical data. We believe it will be more helpful for less experienced forecasters. There has been little research on generating descriptions from a sequence of raw numerical data even in the data-to-text generation (Gatt and Krahmer, 2018).



Delivery time: 05:51 a.m. on 06 April, Tokyo

Today patches of blue sky will appear, but the sky will become cloudy and it will gradually start to rain in the evening. Please bring an umbrella when you go out, even if it's not raining.

Figure 1: A weather comment written by a meteorological expert and simulation results from NWP models

We illustrate the three characteristic problems of weather-comment generation in Figure 1. The first problem is that a forecaster needs to consider the changes in numerical values for different types of physical quantities. For example, the comment in this figure states that it will be raining in the evening after a sunny spell according to the changes in precipitation and cloud cover. The second problem is that weather-forecast comments are often written on the basis of meta-data such as area (e.g., Tokyo), delivery time (e.g., 05:51 a.m.), and date

---

[1] https://github.com/titech-nlp/pinpoint-weather

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1462–1473
April 19 - 23, 2021.

(e.g., 06 April). For example, weather comments contain expressions that depend on their delivery time and date; comments published in the morning use "*today*", while those posted in the evening usually refer to "*tomorrow*". The third problem is that consumers place a higher priority on informativeness of weather comments and their correctness. In particular, important information such as sunny, rain, and snow should be explicitly mentioned since it will greatly affect the consumers. For example, in Figure 1, although there are several possible types of content to be described such as precipitation, cloud cover, and air pressure, the comment mainly focuses on the information on rain and umbrellas since they can affect the consumers' behavior.

To address these issues, we propose a data-to-text model for generating weather comments from simulation results of NWP models and past observation data. To tackle the first problem, we use a multi-layer perceptron (MLP) or convolutional neural network (CNN) to capture different types of physical quantities and input them to a bidirectional recurrent neural network (Bi-RNN) to take their time scales into account. For the second problem, we incorporate meta-data, such as area information, delivery time, and date, into an encoder for the meta-data. To address the third problem, we introduce *weather labels* representing weather information, such as sunny and rain, to help our model explicitly describe useful information and improve the correctness.

We conducted automatic and human evaluations to evaluate the proposed model on the task of generating Japanese weather comments from simulation results of NWP models and meteorological observation data. The results of both automatic and human evaluations indicate that our model improves the informativeness of generated comments compared with baselines.

2 Related Work

Data-to-text generation, which is the task of automatically producing descriptions from non-linguistic data (Gatt and Krahmer, 2018), has been widely used in various fields such as sports (Wiseman et al., 2017; Puduppully et al., 2019), finance (Murakami et al., 2017; Aoki et al., 2018, 2019), and medical care (Portet et al., 2009; Jing et al., 2018). Neural generation methods have been attracting increased attention in the field of data-to-text generation (Liu et al., 2018; Iso et al., 2019), although rule-based approaches have been the mainstream (Kukich, 1983; Reiter et al., 2005).

The task of generating weather-forecast comments has traditionally been tackled in the field of data-to-text generation (Belz, 2007; Angeli et al., 2010; Mei et al., 2016). For example, there are efforts in generating weather-forecast comments intended for marine shipping or offshore oil facilities (Kittredge et al., 1986; Reiter et al., 2005), as well as local weather forecasts for more general use (Kerpedjiev, 1992; Liang et al., 2009).

Prior research has examined the second and third problems mentioned in Section 1 (Murakami et al., 2017; Puduppully et al., 2019). For the second problem, we need to incorporate information for time and area into a generation model to generate time-dependent expressions. For the third problem, we must carry out content selection to explicitly provide useful information, such as sunny and rain, for consumers. In the table-to-text task, which aims to generate a description from a structured table, there have been recent efforts to improve the correctness of generated texts by implicitly introducing a content-matching constraint (Wang et al., 2020), explicitly specifying the content in the table (Ma et al., 2019) or incorporating copy mechanism (Lebret et al., 2016). Nonetheless, the techniques proposed in the table-to-text task are not directly applicable to datasets consisting of raw numerical data, such as simulation results of NWP models, and texts since they rely on task-specific architectures such as the copy mechanism copying words from tables. In addition, Puduppully et al. (2019) proposed a method for generating summaries of basketball games by using the correspondence between entities in text and input tabular data extracted using the information-extraction method (Wiseman et al., 2017). However, the methods are also not applicable to datasets consisting of raw numerical data and texts because they rely heavily on a word-matching algorithm between input tables and texts. To overcome this limitation, we extract weather labels representing the content of weather information from *only* text on the basis of clue words and use them to explicitly describe the useful information.

3 Weather Data

Weather forecasters obtain the output of NWP models and past weather observations, and interpret them together to come up with weather comments.

To reproduce this, we use two types of meteorological data: a numerical forecast map and meteorological observation data.

Numerical Forecast Maps A numerical forecast map is composed of a sequence of 2D surface data extracted from the simulation results of an NWP model, which is a mathematical model of the atmosphere and oceans. In this study, we used numerical forecast maps around Japan simulated using the global spectral model (GSM), which is an NWP model, provided by the Japan Meteorological Agency[2]. The maps are updated four times a day at 0000, 0600, 1200, and 1800 Japan Standard Time. The prediction of physical quantities such as humidity and temperature up to 84 hours ahead is available and is suitable for roughly determining weather trends over a few days. In the maps, grid points are set every 20 km in the range of 20 to 50 degrees north latitude and 120 to 150 degrees east longitude. Therefore, the maps are composed of 151 × 121 grid points, where each point contains simulation results of the physical quantities corresponding to the area, such as 1021.01 hPa for air pressure.

Observation Data Since weather-forecast comments are often written in comparison with past weather (e.g., the day before), we also introduce meteorological observation data provided by the automated meteorological data acquisition system (AMeDAS), managed by Japan Meteorological Agency[3]. Specifically, we use four physical quantities: precipitation, air temperature, wind speed, and sunshine duration. These quantities are sequential data observed every ten minutes at about 1300 stations across Japan.

4 Weather-Forecast Generation

We consider weather-forecast generation as a task of generating a description from a sequence of 2D data, which is a forecast map. Since this can be viewed as video captioning (Yao et al., 2015; Long et al., 2018), we first introduce an encoder-decoder model (Sutskever et al., 2014) with an attention mechanism (Bahdanau et al., 2015).

Figure 2 shows an overview of our proposed model. It takes three types of input data: a sequence of numerical forecast maps $\boldsymbol{g} = (g_i)_{i=1}^{|\boldsymbol{g}|}$, observation data $\boldsymbol{a} = \{a_i\}_{i=1}^{|\boldsymbol{a}|}$, and meta-data for comments such as delivery time and area $\boldsymbol{m} = \{m_i\}_{i=1}^{|\boldsymbol{m}|}$. Here, g_i, a_i, and m_i represent a forecast map, a numeric vector for a certain physical quantity (e.g., precipitation), and an embedding vector for specific information (e.g., area name), respectively. The output of our proposed model is a weather-forecast comment $\boldsymbol{w} = (w_i)_{i=1}^{|\boldsymbol{w}|}$ and weather labels $\boldsymbol{l} = \{l_i\}_{i=1}^{|\boldsymbol{l}|}$, where w_i and l_i are a word and a weather label, respectively.

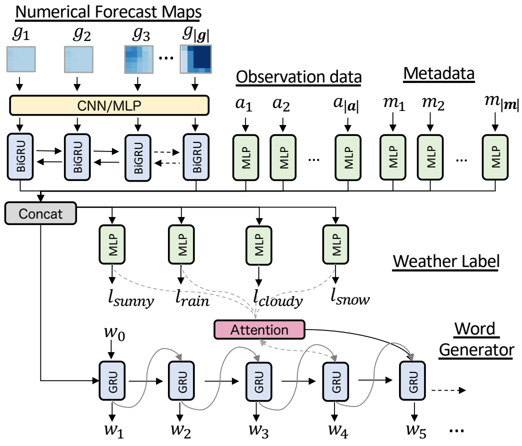


Figure 2: Neural network architecture of proposed model. For simplicity, attention mechanism for three types of input data is omitted.

For the numerical maps, we use either an MLP or CNN to extract numeric features related to physical quantities, such as air pressure, and input each map to a Bi-RNN to take their sequential information into account. We also incorporate meteorological observation data $\boldsymbol{a}$ and meta-data $\boldsymbol{m}$ by encoding with an MLP. We use the MLP to predict weather labels $\boldsymbol{l}$ from the output of the encoders. We use the RNN language model (RNNLM) (Mikolov et al., 2010) to generate words $\boldsymbol{w}$.

We explain our proposed model and how we introduce meta-data and weather labels into it in the following sections.

4.1 Extracting Numerical Maps for Areas

A weather forecaster writes a weather comment for a specific area by referring to weather data corresponding to the area. With this in mind, we extract a g_i for each area, which has 5 × 5 grid points, from an larger map that has 151 × 121 grid points on the basis of latitude and longitude. The extracted map will be a map of 100 square kilometers around the area. Thus, a sequence of numerical forecast maps $\boldsymbol{g}$, for a specific area (e.g., Tokyo) can be acquired, as shown in Figure 3, which shows changes for ten

---

[2] `https://www.jma.go.jp/jma/indexe.html`
[3] `http://www.jma.go.jp/en/amedas/`

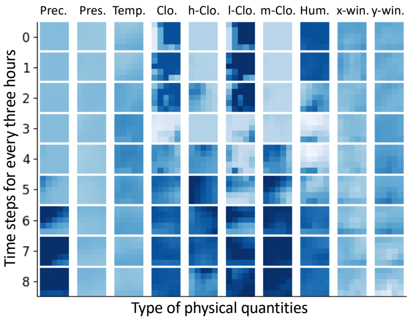


Figure 3: Sequence of numerical forecast maps for specific area, extracted from larger map. Types of physical quantities are as follows: precipitation (Prec), air pressure (Pres), air temperature (Temp), cloud cover (Clo), high-level cloud cover (h-Clo), low-level cloud cover (l-Clo), medium-level cloud cover (m-Clo), humidity (Hum), and wind direction (x-win, y-win).

types of physical quantities (such as precipitation and cloud cover) predicted with an NWP model every three hours up to 24 hours ahead.[4]

4.2 Encoding Numerical Forecast Maps

The task of generating text from sequences of 2D data can be regarded as video captioning. Thus, we use either a CNN or MLP to capture the numeric features of forecast maps. We compared the following two encoders in our experiments.

CNN-based Encoding A CNN is widely used in video captioning to extract visual features from each frame of a video. For example, the CNN encoder for color images has three channels for red, green, and blue. We used ten channels, each channel corresponds to one of the physical quantities (e.g., precipitation), shown in Figure 3, to take the relationships among them into consideration.

MLP-based Encoding In image recognition, a CNN has an advantage over an MLP in terms of translation invariance. However, since we use a map for a specific area, which is extracted from a larger map and whose center is always that area, we hypothesize that a model that takes into account the absolute position on the map is more suitable. Therefore, we use an MLP, which has $10 \times 5 \times 5$ units, to extract the features from forecast maps.

By applying the encoding method to a g_i for each time-step t, we obtain an output vector h^g_i. Next, we sequentially input vector h^g_i to a Bi-RNN to capture value changes for physical quantities in the sequence. As a result, we have the output vector h^g that represents time-series changes in $\boldsymbol{g}$ by concatenating the hidden states of the Bi-RNN:

$$h^g = [h^g_1; h^g_{|\boldsymbol{g}|}], \tag{1}$$

where $[;]$ represents a vector concatenation.

4.3 Encoding Observation Data

With regard to $\boldsymbol{a}$, which are a set of observed values for physical quantities (e.g., sun duration), we use an MLP, which performed best in a study by Murakami et al. (2017). We apply the MLP to each a_i to obtain feature vectors h^a_i. These vectors are concatenated to create the representation that captures the characteristics of meteorological observation data:

$$h^a = [h^a_1; h^a_2; \cdots; h^a_{|\boldsymbol{a}|}]. \tag{2}$$

4.4 Introducing Meta-Data

Since weather forecasters often take their own local knowledge and time information into account when writing weather comments, we incorporate the meta-data for weather comments (such as delivery time) to generate word expressions that depend on the date and time for a comment, e.g., "*today*". Specifically, we create an m_i on the basis of the delivery time and area name (such as *5 a.m.* and *Tokyo*, as shown in Figure 1). We also encode the vector by using the MLP and concatenate its output vector h^m_i. Thus, we obtain a vector h^m that captures the meta-data $\boldsymbol{m}$:

$$h^m = [h^m_1; h^m_2; \cdots; h^m_{|\boldsymbol{m}|}]. \tag{3}$$

Finally, we set the initial hidden state s_0 of both decoders as follows:

$$s_0 = \mathrm{ReLU}(\mathrm{MLP}([h^g; h^a; h^m])). \tag{4}$$

4.5 Weather Labels

In this task, the informativeness of weather comments and their correctness are a key factor. However, since neural-generation models often struggle to capture long-term dependencies, they lose important information included in input data. This shortcoming limits their application to the real world. Thus, we propose a method for explicitly specifying the *content* to be mentioned to help our proposed model correctly describe useful information,

---

[4] The shade of color for physical quantities indicates the magnitude of predicted values. The predicted values are standardized with all values for a year. For example, the dark color for precipitation around 15–18 hours later in Figure 3 indicates that the predicted amount of precipitation is relatively high.

Label	**Clue words**
SUNNY	晴れ (*sunny*), 日差し (*sunlight*), 青空 (*blue sky*)
RAIN	雨 (*rain*), 大雨 (*heavy rain*), にわか雨 (*shower*)
CLOUDY	曇り (*cloudy*), 曇 (*cloudy*), 雲 (*cloud*)
SNOW	雪 (*snow*), 吹雪 (*blizzard*), 小雪 (*light snowfall*)

Table 1: Example weather labels and their corresponding clue words. English translations are given in parentheses. All examples are shown in Appendix A.

inspired by the recent success of faithful data-to-text generation (Ma et al., 2019; Wang et al., 2020) and the controllability of neural models with explicit labels (Aoki et al., 2019). However, since most data-to-text datasets do not typically include content plans, we introduce a simple approach for extracting them from a text as a pseudo reference.

Since consumers are primarily interested in weather information such as sunny and rain, we define such content in weather comments as *weather labels*. Specifically, we introduce four types of weather labels: SUNNY, RAIN, CLOUDY and SNOW. To extract weather labels from weather comments, we define clue words for each weather label, as shown in Table 1. Our strategy is to explicitly match the clue words and words in weather comments. For example, the weather comment in Figure 1 includes the clues words "*blue sky*", "*cloudy*", and "*rain*", so three weather labels, SUNNY, CLOUDY, and RAIN can be associated with the comment. The method for labeling text is very simple, but we found that it works in most cases[5]. The method can also avoid the following two issues in weather-comment generation. First, it is almost impossible to explicitly associate a comment with continuous numerical data such as meteorological data, although the table-to-text task can easily do this, as discussed in Section 2. Second, we need expert knowledge if we annotate content to be mentioned in input data without reference text.

To determine content to be mentioned before text generation begins, we introduce a binary classifier for each weather label, as shown in Figure 2. The classifier is based on an MLP. We train the classifier with the weather labels extracted from comments in the training data. In the inference stage, each classifier predicts each weather label (e.g., l_{sunny}) from the three types of input data ($\boldsymbol{g}$, $\boldsymbol{a}$, $\boldsymbol{m}$). The word generator then generates weather comments $\boldsymbol{w}$ from the input data and those labels.

4.6 Word Generator

The word generator is based on a RNNLM with an attention mechanism. In addition to introducing the attention mechanism into input data (Wiseman et al., 2017; Chen et al., 2019), we also introduce the attention mechanism into the weather labels, as shown in Figure 2. The word generator is designed to take into account weather labels to explicitly describe important information in text generation through this attention mechanism.

In the word generator, the probability of producing a word w_t at t is computed by

$$p(w_t|w_{<t}, \boldsymbol{g}, \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{l}) = \mathrm{softmax}_{\mathrm{w_t}}(W_s s_t^w), \quad (5)$$

$$s_t^w = \mathrm{GRU}(w_{t-1}, s_{t-1}^w, c_t), \quad (6)$$

where w_{t-1} and s_{t-1}^w are an output word and the hidden state of the word decoder at time step $t-1$, respectively. W_s is a weight matrix. Vector c_t represents the context vector at t, which is created by concatenating four context vectors $[c_t^g; c_t^a; c_t^m; c_t^l]$ constructed with the attention mechanism (Bahdanau et al., 2015) for input data $(\boldsymbol{g}, \boldsymbol{a}, \boldsymbol{m})$ and weather labels $\boldsymbol{l}$. For instance, the context vector c_t^l over $\boldsymbol{l}$ at t can be calculated as follows:

$$c_t^l = \sum_{i=1}^{|\boldsymbol{l}|} \alpha_{t,i}^l s_i^l, \quad \alpha_{t,i}^l = \frac{\exp(\eta(s_{t-1}^w, s_i^l))}{\sum_{j=1}^{|\boldsymbol{l}|} \exp(\eta(s_{t-1}^w, s_j^l))}, \quad (7)$$

where s_i^l represents the hidden state of the weather-label classifier for label l_i, and $\alpha_{t,i}^l$ is the alignment probability between the t-th output word and i-th hidden state in the classifier. We use an MLP η as a score function. Note that c_t^g, c_t^a, and c_t^m for the input data ($\boldsymbol{g}$, $\boldsymbol{a}$, $\boldsymbol{m}$) can be derived with Equation (7) in a similar manner.

5 Experiments

5.1 Setup

Dataset We used weather comments from 2014 to 2015 in Japan as the text dataset, which consists of 57,412 comments provided by Weathernews Inc. We separated the dataset into 28,555 comments from 2014 for training, 14,464 and 14,393 comments from 2015 for development and testing. For the numerical forecast maps, we collected 2,715

---

[5] We evaluated correctness of the extracted weather labels by five people on the basis of 100 comments, which are randomly extracted from development set. As a result, 96% of the weather labels were judged to be appropriate.

maps corresponding to the comments from the website[6] of the Research Institute for Sustainable Humanosphere, Kyoto University. We also separated the numerical maps into 1,344 from 2014 for training, 1,326 and 1,329 from 2015 for development and testing, respectively. Their sum does not agree with the total number of maps, 2,715 because the weather comments for development and testing are sampled from 2015, and numerical maps for different areas are often extracted from a single entire map around Japan and are used for them. Note that the weather comments and the corresponding extracted maps are unique for each area, and the comments for development and testing do not overlap with each other.

We used a $\boldsymbol{g}$, which consists of nine steps every three hours up to 24 hours ahead, since the comments treat weather forecasts up to the next day. Referring to delivery date and time of a comment, we aligned each $\boldsymbol{w}$ with the extracted $\boldsymbol{g}$ obtained by following the procedure in Section 4.1. We also used precipitation, air temperature, wind speed, and sunshine duration for the last 24 hours as $\boldsymbol{a}$, where each a_i consists of observed values for 24×6 steps. As $\boldsymbol{m}$, we used delivery date, time, and area name (e.g., *April*, *Monday*, *5 a.m.*, *Tokyo*).

Implementation We used a single-layer MLP and bi-directional gated recurrent unit (Bi-GRU) for the encoders and two-layer GRU for the word generator. Note that we have investigated a transformer-based model (Vaswani et al., 2017), but we found that there were no significant differences between the transformer-based model and the GRU-based model. The hidden states of our model and size of the word embeddings were both 512. We set the dimension size of the hidden vectors for the meta-data h_i^m and observation data h_i^a to 64. The model was trained using the Adam optimizer (Kingma and Ba, 2015). We applied an early stopping strategy with a minimum number of 25 epochs. We stopped training if there was no improvement in validation loss for three consecutive epochs.

Evaluation Metrics For the automatic evaluation, since reference texts written by meteorological experts generally mention important information such as sunny and rain, we used BLEU-4[7] (Papineni et al., 2002) and ROUGE-1[8] (F_1 score) (Lin, 2004) to see whether generated texts properly mention the important information as reference texts do. However, since these metrics based on word overlapping rely on the reference texts, they cannot be used to assess the correctness of the generated texts if their expressions are different from the reference texts. Thus, we also calculated precision, recall and F_1 scores of weather labels, which are extracted from the generated texts, to see how they properly describe important information in comparison with those of the reference texts.

For the human evaluation, we asked five participants to give each generated comment a score from 1 to 3 for informativeness, consistency, and grammar, where 3 is the highest. In the evaluation of informativeness, we showed the participants a human-generated comment as a reference and asked them to compare the generated text and reference. This was done because understanding complicated input data for data-to-text generation is extremely difficult for non-specialists. We randomly selected 40 comments from the test set[9]. Each comment was rated by all five participants. We used Wilcoxon signed rank test (Wilcoxon, 1945) to test the statistical significance of the difference among comparative models. The details for instructing the human raters how to evaluate each comment are provided in Appendix B.

Models We defined five models listed in Table 2 and *w/o Meta*, which does not take meta-data into account, to determine whether each component (e.g., weather label) contributes to the results. Models (1) and (2), which do not take weather labels into account, were regarded as baselines. Model (3) uses the weather labels that we proposed. To further improve the correctness of generated texts, we also introduced model (4) to investigate a content-matching constraint loss (Wang et al., 2020) that aims to constrain an embedding of input data to be close to the corresponding target text embedding. Specifically, we calculated the loss between the output embeddings of the weather-label classifier and the target text embeddings to make their representations close. We also evaluated model (5) that used oracle labels extracted from the reference text to validate the upper bound of improvement in the

---

[6] `http://database.rish.kyoto-u.ac.jp/index-e.html`

[7] `https://github.com/mjpost/sacrebleu`

[8] `https://github.com/pltrdy/rouge`

[9] To clarify effectiveness of each weather label, we randomly extracted the comments so that each label was included in at least 10 cases.

Model	Components			Word Overlap		SUNNY			RAIN			CLOUDY			SNOW		
	Enc.	Weather	CL	BLEU	ROUGE	P%	R%	F_1%	P%	R%	F_1%	P%	R%	F_1%	P%	R%	F_1%
(1)	CNN	–	–	12.7	42.8	83.5	67.6	74.7	72.8	83.6	77.8	58.5	59.8	59.0	75.2	50.1	60.2
(2)	MLP	–	–	13.0	43.5	83.2	68.4	74.9	74.6	83.5	78.8	59.8	60.3	59.9	75.7	53.3	62.3
(3)	MLP	Pred.	–	12.9	43.8	81.0	78.5	79.7	78.6	80.0	79.3	62.5	55.9	58.9	75.9	60.4	67.2
(4)	MLP	Pred.	✓	13.2	43.9	81.0	78.4	79.7	76.6	84.1	80.2	60.6	59.3	59.8	77.7	58.5	66.6
(5)	MLP	Orac.	✓	14.6	45.5	94.9	84.5	89.4	84.4	92.9	88.4	84.7	85.6	85.1	91.3	63.8	75.1

Table 2: Results of automatic evaluation on test set using BLEU, ROUGE, and correctness of each weather label extracted from its generated text in precision (P%), recall (R%), and F_1 scores (F_1%). Models are numbered (1) through (5). Models (1) and (2) are baselines. Components of each model are as follows: encoder for numerical forecast map (Enc.), weather labels (Weather), and content-matching constraint loss (CL). Weather represents whether we use weather labels extracted from generated text (Pred.) or oracle labels (Orac.) extracted from reference text. Scores were averaged over three runs.

Expression	Model (4)	w/o Meta	Δ
今日 (*Today*)	99.3	97.3	+2.0
明日 (*Tomorrow*)	95.1	91.1	+4.0
月 (*Monday*)	29.3	0.0	+29.3
火 (*Tuesday*)	29.2	0.0	+29.2
春 (*Spring*)	14.0	2.4	+11.6
夏 (*Summer*)	19.1	12.4	+6.7
BLEU	13.2	12.7	+0.5

Table 3: F_1 scores for time-dependent expressions. Each expression is accompanied by its English translation in parenthesis. Δ means difference in each score between Model (4) and w/o Meta.

Label	Precision	Recall	F_1 score
SUNNY	79.7	84.9	82.1
RAIN	79.9	80.5	80.2
CLOUDY	61.5	62.5	61.6
SNOW	73.9	67.1	70.3

Table 4: Results of weather-label prediction by classifier, which only performs weather-label prediction, on test set. Scores were averaged over three runs.

correctness of generated text with the labels.

A rule-based system could be also considered as a comparative model, but we could not include it since it is practically impossible to construct their *rules* without expert knowledge in meteorology.

5.2 Results

The results of the automatic evaluation are listed in Table 2. To see how our proposed model can correctly describe the content of weather forecasts, we calculated precision, recall, and F_1 scores of weather labels, which are extracted from their generated texts, by referring to weather labels extracted from human-generated texts as references. Models (3) and (4), which introduce weather labels, outperformed the baselines, which do not take into account the weather labels. In particular, we found that models (3) and (4) significantly improved F_1 scores for the sunny and snow labels by around 5% in comparison to the baselines. This suggests that incorporating the weather labels enables the model to more correctly generate texts.

In terms of BLEU and ROUGE, model (4) outperformed all other models except (5), which can use oracle labels. Interestingly, model (5) significantly improved the correctness of each weather label as in Table 2 since it can use the oracle labels, but the improvement in BLEU and ROUGE scores was limited. This implies that it is difficult to evaluate the correctness of generated texts by relying on only these metrics.

5.3 Contributions of Each Component

Next, we compared the models to investigate how each component contributes to their performance.

Encoder for numerical maps In the comparison between models (1) and (2), which use either a CNN or MLP as an encoder for numerical forecast maps, no significant differences were found.[10] This suggests that either encoder is acceptable, although we had originally expected the MLP encoder that takes into account absolute positions on the maps to be more suitable than a CNN encoder because of its simplicity.

Effect of meta-data With regard to the meta-data such as delivery time and date, for which we expected the model to generate time-dependent expressions (e.g., “*today*”), we conducted an ablation

---

[10] In spite of this, we used an MLP for the models to add other components since it performed slightly better than a CNN in the experiments.

Model	I	C	G	Generated comment
Ref.	–	–	–	今日は、うっすら雲が広がりやすいものの、日差しが届きます。夏至の日差しは強烈なので、紫外線・暑さ対策が欠かせません。 (*Today, the sky will easily become partly cloudy, but it will become sunny. Since the sun this summer is intense, please take precautions against the heat and ultraviolet rays.)*
(2)	2.0	3.0	3.0	今日（月）は日差しが届きますが、段々と雲が広がります。午後はニワカ雨の可能性があるので、折りたたみ傘があると安心です。 (*Today (Monday), it will be sunny then gradually become cloudy. Since there is a chance of rain in the afternoon, it will be a good idea to take a folding umbrella when you go out.)*
(4)	2.8	2.8	3.0	今日（月）は雲が広がりやすいものの、日差しが届く時間もあります。ムシムシとした暑さになるので、熱中症対策を忘れずに。 (*Today (Monday), the sky will become cloudy but will become sunny. Since it will be hot and humid, please remember to take precautions against heatstroke.)*

Table 5: Reference weather comment written by human (Ref.) and those comments generated from models (2) and (4). Weather labels extracted from the reference text are SUNNY and CLOUDY. The reference comment was posted at 00:02 a.m. on Monday June 22, 2015 for the *Toyohashi* area. Columns I, C, and G show average scores for informativeness, consistency, and grammar from human evaluation, respectively. Each example is accompanied by its English translation.

Label	Model(2)			Model(4)			# of cases
	Info.	Con.	Gra.	Info.	Con.	Gra.	
SUNNY	1.92	2.91	2.91	2.10	2.82	2.88	26
RAIN	2.02	2.93	2.92	2.13	2.88	2.90	26
CLOUDY	1.99	2.93	2.94	2.12	2.83	2.89	19
SNOW	1.88	2.95	2.92	1.95	2.91	2.94	13
Overall	1.98	2.92	2.92	2.10	2.86	2.90	40

Table 6: Results of human evaluation. Scores are averages given by five human raters. Columns Info, Con, and Gra represent informativeness, consistency, and grammar, respectively. Differences in informativeness and consistency are statistically significant at $p < 0.05$.

study to investigate whether our proposed model can properly generate these expressions in comparison with w/o Meta that does not take into account such meta-data. Specifically, we calculated F_1 scores for time-dependent expressions by using weather comments written by human as references. Table 3 shows F_1 scores for each expression in the comments generated with model (4) and w/o Meta, respectively. We found that model (4), which takes into account the meta-data, can more accurately provide time-dependent expressions than w/o Meta. This finding suggests that introducing the meta-data into a generation model improve the correctness of meta-data in generated comments.

Effect of weather labels In comparison between models (2) and (3), which do not and do use the weather labels, respectively, we found that model (3) significantly improved the F_1 scores for the weather labels extracted from their generated comments than model (2). Specifically, the recall scores significantly improved regarding the weather labels for sunny and snow. This indicates that specifying content to be mentioned, such as weather labels, helps the model to explicitly describe the information.

We also tested a classifier that only predicts weather labels from input data to clarify the upper bound of the improvement in the correctness of each weather label extracted from its generated text. Table 4 presents the results of weather-label prediction by the classifier. According to the comparison between Tables 2 and 4, the scores of weather labels extracted from the generated texts with models (3) and (4) are approaching the upper bound by the classifier, but there is still room for improvement.

Effect of content-matching constraint To investigate the effectiveness of the content-matching constraint loss (Wang et al., 2020) for improving faithfulness of generated texts, we compared model (4), which uses this loss, with model (3), which does not. There was a slight improvement in BLEU and ROUGE scores.

5.4 Human Evaluation

Table 6 lists the results of the human evaluation, where # represents the number of cases, which includes each weather label in the evaluation set. Note that a comment may contain multiple labels. Overall, model (4), which explicitly performs content selection by using weather labels, outperformed model (2), which does not, in terms of informativeness[11]. This indicates that introducing weather labels contributes to the correctness of information included in generated texts, as we also can see from the results of the automatic evalua-

---

[11] Specifically, model (4) was rated more informative than model (2) in 20 of the 40 cases. 10 of them were equivalent.

tion. Model (4), however, was inferior to model (2) in terms of consistency, although the score is still significantly high. This is reasonable because more information makes it more challenging to maintain consistency. To solve this problem, it is necessary to carry out not only content selection but also content planning to specify both *what to say* and *in which order* (Wiseman et al., 2017).

Table 5 shows an example reference and weather comments generated with models (2) and (4). Both models correctly described the information on cloudy weather and sunshine, but model (2) mistakenly described rainy weather compared with the reference. In contrast, model (4) properly described all the information including hot weather and was judged as more informative than model (2). More generation examples are given in Appendix C.

6 Conclusion

In this paper, we addressed the task of generating weather comments from meteorological simulations. We proposed a data-to-text model and incorporated three types of encoders for forecast maps, observation data, and meta-data into the model. In addition, we introduced weather labels representing the content of weather information to explicitly carry out content selection and improve the correctness of information in generated comments. Experiments indicated that our model significantly improved the informativeness of generated comments and outperformed the baselines in both automatic and human evaluations.

Acknowledgements

We thank the anonymous reviewers for their insightful comments and valuable suggestions. This work was supported by JST PRESTO (Grant Number JPMJPR1655) and JST-Mirai Program Grant Number JPMJMI18BB, Japan.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512. Association for Computational Linguistics.

Kasumi Aoki, Akira Miyazawa, Tatsuya Ishigaki, Tatsuya Aoki, Hiroshi Noji, Keiichi Goshima, Ichiro Kobayashi, Hiroya Takamura, and Yusuke Miyao. 2019. Controlling contents in data-to-document generation with human-designed topic labels. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 323–332. Association for Computational Linguistics.

Tatsuya Aoki, Akira Miyazawa, Tatsuya Ishigaki, Keiichi Goshima, Kasumi Aoki, Ichiro Kobayashi, Hiroya Takamura, and Yusuke Miyao. 2018. Generating market comments referring to external resources. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 135–139. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

Anja Belz. 2007. Probabilistic generation of weather forecast texts. In *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 164–171. Association for Computational Linguistics.

Shuang Chen, Jinpeng Wang, Xiaocheng Feng, Feng Jiang, Bing Qin, and Chin-Yew Lin. 2019. Enhancing neural data-to-text generation models with external background knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3022–3032. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61(1):65–170.

Eli Goldberg, Norbert Driedger, and Richard I. Kittredge. 1994. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Hayate Iso, Yui Uehara, Tatsuya Ishigaki, Hiroshi Noji, Eiji Aramaki, Ichiro Kobayashi, Yusuke Miyao, Naoaki Okazaki, and Hiroya Takamura. 2019. Learning to select, track, and generate for data-to-text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2102–2113. Association for Computational Linguistics.

Baoyu Jing, Pengtao Xie, and Eric Xing. 2018. On the automatic generation of medical imaging reports. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2577–2586. Association for Computational Linguistics.

Stephan M. Kerpedjiev. 1992. Automatic generation of multimodal weather reports from datasets. In *Third*

Conference on Applied Natural Language Processing, pages 48–55. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

R. Kittredge, A. Polguere, and E. Goldberg. 1986. Synthesizing weather forecasts from formatted data. In *Proceedings of the 11th International Conference on Computational Linguistics*.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213. Association for Computational Linguistics.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of Association for Computational Linguistics and International Joint Conference on Natural Language Processing*, pages 91–99. Association for Computational Linguistics.

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

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 4881–4888.

Xiang Long, Chuang Gan, and Gerard de Melo. 2018. Video captioning with multi-faceted attention. *Transactions of the Association for Computational Linguistics*, 6:173–184.

Shuming Ma, Pengcheng Yang, Tianyu Liu, Peng Li, Jie Zhou, and Xu Sun. 2019. Key fact as pivot: A two-stage model for low resource table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2047–2057. Association for Computational Linguistics.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730. Association for Computational Linguistics.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association*, pages 1045–1048. International Speech Communication Association.

Soichiro Murakami, Akihiko Watanabe, Akira Miyazawa, Keiichi Goshima, Toshihiko Yanase, Hiroya Takamura, and Yusuke Miyao. 2017. Learning to generate market comments from stock prices. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1374–1384. Association for Computational Linguistics.

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

François Portet, Ehud Reiter, Albert Gatt, Jim Hunter, Somayajulu Sripada, Yvonne Freer, and Cindy Sykes. 2009. Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, 173(7-8):789–816.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6908–6915.

Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169.

Somayajulu Sripada, Ehud Reiter, and Ian Davy. 2004. Sumtime-mousam: Configurable marine weather forecast generator. *Expert Update*, 6.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Zhenyi Wang, Xiaoyang Wang, Bang An, Dong Yu, and Changyou Chen. 2020. Towards faithful neural table-to-text generation with content-matching constraints. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1072–1086. Association for Computational Linguistics.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263. Association for Computational Linguistics.

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. 2015. Describing videos by exploiting temporal structure. In *Proceedings of the 2015 IEEE International Conference on Computer Vision*, page 4507–4515. IEEE Computer Society.

A Examples of All Clue Words

To extract weather labels from weather comments, we defined clue words for each weather label, as shown in Table 7. Note that we selected the clue words in reference to the development data.

B Details of Human Evaluation

The following are the instructions presented to evaluation raters during the human evaluation. Each comment was rated by all five participants.

We only showed the raters the weather comments written by human as references to evaluate the generated comments. This was done because understanding complicated data, which is used as input for data-to-text generation, such as numerical forecast maps, is particularly difficult for non-specialists.

Informativeness:

3: This is an ideal weather comment since it appropriately mentions important information.

2: Although some important information is missing, the information included in the comment is appropriate and is acceptable as a weather comment.

1: The information included in the comment is incorrect and inappropriate as a weather comment.

Consistency:

3: The comment is consistent on the whole and it is easy to read.

2: The comment lacks consistency in some parts and is difficult to read.

1: The comment lacks consistency on the whole and is difficult to understand.

Grammar:

3: There are no grammatical errors.

2: There are some grammatical errors, but it is understandable.

1: There are many grammatical errors, and it is difficult to understand.

C More Generation Examples

More generation examples from model (2) and (4) are shown in Table 8 and 9. In this example, we can observe that model (2) was judged as less informative on average by the five raters in comparison with model (4) since the comments generated by model (2) provide incorrect information (e.g., *rain shower* instead of *snow* in Table 8), or lack important information (e.g., *rain* in Table 9). On the other hand, model (4), which is our proposed model, properly describes these important information.

Label	**Clue words**
SUNNY	晴れ (*sunny*), 日差し (*sunlight*), 青空 (*blue sky*), 回復 (*improvement*), 日和 (*perfect day*), 陽気 (*weather*), 秋晴れ (*fine autumn day*), 晴天 (*fine weather*), 晴れ間 (*patch of blue sky*), 晴れる (*clear up*), 太陽 (*sun*), 五月晴れ (*beautiful weather in May*)
RAIN	雨 (*rain*), 大雨 (*heavy rain*), にわか雨 (*shower*), 雷雨 (*thunderstorm*), 暴風雨 (*storm*), 雨風 (*rain and wind*), 荒天 (*stormy weather*), 台風 (*typhoon*), 傘 (*umbrella*)
CLOUDY	曇り (*cloudy*), 曇 (*cloudy*), 雲 (*cloud*)
SNOW	雪 (*snow*), 吹雪 (*blizzard*), 小雪 (*light snowfall*), 吹雪く (*snow blowing*), ふぶく (*snow blowing*)

Table 7: Weather labels and their corresponding clue words we used to extract labels from weather comments. Corresponding English translations are also given in parenthesis.

Model	**I**	**C**	**G**	**Generated comment**
Ref.	–	–	–	今日は日差しが届いても、ゆっくり天気は下り坂になります。夕方から雪が降り出すので、外出は早めがオススメ。昨日より冷え込みが増すので、防寒が必須です。 (*Today, it will be sunny, but the weather will go downhill. Snow will start to fall in the evening. If you are going out, you should go early. Since it will be colder than yesterday, cold protection is important.)*
(2)	2.0	3.0	2.6	今日（土）は日差しが届いても、ニワカ雨の可能性があります。お出かけには折りたたみ傘があると安心です。昼間でも気温は低空飛行で体を温めるのも良さそうです。 (*Today (Saturday), it will be sunny, but there is a chance of rain shower. It will be a good idea to take a folding umbrella when you go out. Since the weather is cool even during the daytime, it will be good to warm your body up.)*
(4)	2.4	3.0	3.0	今日（土）は日差しが届いても雲が優勢の天気。ニワカ雪の可能性があるので、折りたたみ傘があると安心です。また、寒さが続くので防寒対策を万全にして下さい。 (*Today (Saturday), it will be sunny, but the weather will be predominantly cloudy. There is a chance of snow shower, so it will be a good idea to have a folding umbrella. Also, please take protective measures against the cold as the cold days continue.)*

Table 8: Reference weather comment written by human (Ref.) and those comments generated from models (2) and (4). The reference comment was posted at 00:00 a.m., on Saturday, December 26, 2015 for the *Shiranuka* area. Columns I, C, and G show average scores for informativeness, consistency, and grammar from human evaluation, respectively. Each example is accompanied by its English translation. Underline indicates incorrect information.

Model	**I**	**C**	**G**	**Generated comment**
Ref.	_	_	_	今日も変わりやすい天気です。急にザッと雨が降る可能性があるので、日差しがあっても油断できません。ムシッとする風もやや強いままです。 (*Today, the weather is changeable too. It may rain suddenly, so please be careful even if it is sunny. The humid wind is a little strong.)*
(2)	1.8	2.8	2.8	今日は雲が広がりやすいものの、日差しが届く時間もあります。この時期の紫外線は強烈なので、ＵＶ対策を忘れずにお出かけ下さい。 (*Today, the cloud will spread, but there is time sunshine appears. Since ultraviolet rays are intense these days, remember to take measures against ultraviolet rays when you go out.)*
(4)	2.6	2.8	2.8	今日は変わりやすい天気。日差しが届いても、急な雨もあります。モクモクした雲が近づいて来たら天気急変のサインです。 (*Today, the weather is changeable. It will be sunny, but it will start to rain suddenly. It is a sign of a sudden change of the weather if fuzzy clouds are approaching.)*

Table 9: Reference weather comment written by human (Ref.) and those comments generated from models (2) and (4). The reference comment was posted at 10:46 a.m., on Saturday, July 18, 2015 for the *Tokyo* area. Columns I, C, and G show average scores for informativeness, consistency, and grammar from human evaluation, respectively. Each example is accompanied by its English translation.

SICK-NL: A Dataset for Dutch Natural Language Inference

Gijs Wijnholds
UiL-OTS
Utrecht University
g.j.wijnholds@uu.nl

Michael Moortgat
UiL-OTS
Utrecht University
m.j.moortgat@uu.nl

Abstract

We present SICK-NL (read: *signal*), a dataset targeting Natural Language Inference in Dutch. SICK-NL is obtained by translating the SICK dataset of Marelli et al. (2014) from English into Dutch. Having a parallel inference dataset allows us to compare both monolingual and multilingual NLP models for English and Dutch on the two tasks. In the paper, we motivate and detail the translation process, perform a baseline evaluation on both the original SICK dataset and its Dutch incarnation SICK-NL, taking inspiration from Dutch skipgram embeddings and contextualised embedding models. In addition, we encapsulate two phenomena encountered in the translation to formulate *stress tests* and verify how well the Dutch models capture syntactic restructurings that do not affect semantics. Our main finding is all models perform worse on SICK-NL than on SICK, indicating that the Dutch dataset is more challenging than the English original. Results on the stress tests show that models don't fully capture word order freedom in Dutch, warranting future systematic studies.

1 Introduction

One of the primary tasks for Natural Language Processing (NLP) systems is Natural Language Inference (NLI), where the goal is to determine, for a given premise sentence whether it contradicts, entails, or is neutral with respect to a given hypothesis sentence.

For English, several standard NLI datasets exist, such as SICK (Marelli et al., 2014), SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018). Having such inference datasets available only for English may introduces a bias in NLP research. Conneau et al. (2018) introduce XNLI, a multilingual version of a fragment of the SNLI dataset, that contains pairs for Natural Language Inference in 15 languages and is explicitly intended to serve as a resource for evaluating crosslingual representations. However, Dutch is not represented in any current NLI dataset, a lack that we wish to complement.

Dutch counts as a high-resource language, with the sixth largest Wikipedia (2M+ articles), despite having ca. 25M native speakers. Moreover, the syntactically parsed LASSY corpus of written Dutch (van Noord et al., 2013), and the SONAR corpus of written Dutch (Oostdijk et al., 2013) provide rich resources on which NLP systems may be developed. Indeed, Dutch is in the scope of the multilingual BERT models published by Google (Devlin et al., 2019), and two monolingual Dutch BERT models have been published as part of HuggingFace's transformers library (de Vries et al., 2019; Delobelle et al., 2020).

Compared to English, however, the number of evaluation tasks for Dutch is limited. There is a Named Entity Recognition task coming from the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003); from a one million word hand annotated subcorpus of SONAR (Oostdijk et al., 2013) one derives part-of-speech tagging, Named Entity Recognition and Semantic Role Labelling tasks. More recently a Sentiment Analysis dataset was introduced, based on Dutch Book reviews (van der Burgh and Verberne, 2019). Moreover, Allein et al. (2020) introduce a classification task where a model needs to distinguish between the pronouns *die* and *dat*.

Given the focus on word/token-level tasks in Dutch, we aim to complement existing resources with an NLI task for Dutch. We do so by deriving it from the English SICK dataset, for the following reasons: first, this dataset requires a small amount of world knowledge as it was derived mainly from image captions that are typically concrete descriptions of a scene. Therefore, no world knowledge requirements will be imposed on an NLP model

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1474–1479
April 19 - 23, 2021.

for the task, but rather its ability for reasoning will be assessed. Secondly, due to the structure of the sentences in SICK, the types of inferences can be attributed to particular constructs, such as hypernymy/hyponymy, negation, or choice of quantification. Thirdly, SICK contains 6076 unique sentences and almost 10K inference pairs, making it a sizeable dataset for NLP standards, while deriving a Dutch version is more manageable than with other datasets. We make the dataset, code and derived resources (see Section 5) available online[1].

2 Dataset Creation

We follow a semi-automatic translation procedure to create SICK-NL, similar to the Portuguese version of SICK (Real et al., 2018). First, we use a machine translator to translate all of the (6076) unique sentences of SICK[2]. We review each sentence and its translation in parallel, correcting any mistakes made by the machine translator, and maintaining consistency of individual words' translation, in the process guaranteeing that the meaning of each sentence is preserved as much as possible. Finally, we perform a postprocessing step in which we ensure unique translations for unique sentences (alignment) with as few exceptions as possible. In this way we obtain 6059 unique Dutch sentences, which means that the dataset is almost fully aligned on the sentence level. It should be noted, however, that we can not fully guarantee the same choice of words in each sentence *pair* in the original dataset, as we translate sentence by sentence. In the whole process, we adapted 1833 of the 6076 automatic translations, either because of translation errors or alignment constraints. As we are interested in collecting a comparable and aligned dataset, we maintain the relatedness and entailment scores from the original dataset.

	SICK	**SICK-NL**
No. of tokens	189783	176509
No. of unique tokens	2328	2870
Avg. sentence length	9.64	8.97
Avg. word overlap	66.91%	58.99%

Table 1: Basic statistics of SICK and SICK-NL.

Table 1 shows some statistics of SICK and its Dutch translation. The most notable difference is that the amount of unique words in Dutch is 23% higher than that in English, even though the total number of words in SICK-NL is about 93% of that in SICK. We argue that this is due to morphological complexities of Dutch, where verbs can be separable or compound, leading them to be split up into multiple parts (for example, "*storing*" becomes "*opbergen*", which may be used as "*de man bergt iets op*"). Moreover, Dutch enjoys a relatively free word order, which in the case of SICK-NL means that sometimes the order of the main verb and its direct object may be swapped in the sentence, especially when the present continuous form ("*is cutting an onion*") is preserved in Dutch ("*is een ui aan het snijden*"). Finally, we follow the machine translation, only making changes in the case of grammatical errors, lexical choice inconsistencies, and changes in meaning. This freedom leads to a decrease in relative word overlap between premise and hypothesis sentence, computed as the number of words in common divided by the length of the shortest sentence. From the perspective of Natural Language Inference this is preferable as word overlap often can be exploited by neural network architectures (McCoy et al., 2019).

3 Baseline Evaluation and Results

We evaluate two types of models as a baseline to compare SICK-NL with its English original.

First, we evaluate embeddings that were not specifically trained on SICK. Table 2 shows the correlation results on the relatedness task of SICK and SICK-NL, where the cosine similarity between two independently computed sentence embeddings is correlated with the relatedness scores of human annotators (between 1 and 5).

	SICK		**SICK-NL**
Skipgram	69.49	Skipgram	56.94
BERT$_{cls}$	50.78	BERTje$_{cls}$	49.06
BERT$_{avg}$	61.36	BERTje$_{avg}$	55.55
RoBERTa$_{cls}$	46.62	RobBERT$_{cls}$	43.93
RoBERTa$_{avg}$	62.71	RobBERT$_{avg}$	52.33

Table 2: Pearson r correlation coefficient for the relatedness task of the English SICK dataset (left) and its Dutch translation (right).

To obtain sentence embeddings here, we average skipgram embeddings, or, in the case of contextualised embeddings, we take either the sentence embedding given by the $[CLS]$ token, or

---

[1] `github.com/gijswijnholds/sick_nl`

[2] We used DeepL's online translator `www.deepl.com/translator`

we take the average of the individual word's embeddings. For the skipgram embeddings in English, we use the standard 300-dimensional GoogleNews vectors provided by the `word2vec` package and for Dutch, we use the 320-dimensional Wikipedia trained embeddings of Tulkens et al. (2016).

The relatedness results show that (a) using the $[CLS]$ token embedding as a sentence encoding performs worse than taking the average of word embeddings, and that (b) the Dutch incarnation of SICK is harder than the original English dataset. It may be noted that relatedness scores are less robust then entailment labels, and so our first result may not be enough support for the claim that SICK-NL poses a more challenging task. For example, it could be that relatedness scores will differ slightly if we were to ask a number of annotators to re-evaluate the Dutch dataset.

In the second setup, we use BERTje, the Dutch BERT model of de Vries et al. (2019) and RobBERT, the Dutch RoBERTa model of Delobelle et al. (2020), with their corresponding English counterparts, as well as multilingual BERT (mBERT), as sequence classifiers on the Entailment task of SICK(-NL). Here we observe a similar pattern in the results in Table 3: while there are individual difference on the same task, the main surprise is that the Dutch dataset is harder, even when exactly the same model (mBERT) is used.

	SICK		**SICK-NL**
BERT	87.34	BERTje	83.94
mBERT	87.02	mBERT	84.53
RoBERTa	90.11	RobBERT	82.02

Table 3: Accuracy results on the entailment task of the English SICK dataset and its Dutch translation for two Dutch BERT models and their English counterparts. For each model, we report the best score out of 20 epochs of fine-tuning.

4 Error Analysis

In order to understand the differences between the Dutch and English language models on the respective tasks, we dive deeper into the classification results. We plot confusion matrices for each model in Table 5, where we separate predictions that the models have in common and the from the predictions that are unique to each model.

In the case of English, performance on classifying contradictions is worse for multilingual BERT and RoBERTa, and RoBERTa also gives highest recall values for the Neutral and Entailment labels. This is all not surprising given that RoBERTa has the overall highest test set accuracy. The surprising results come mainly from the comparison between English and Dutch models. Where BERTje is rather indecisive when it comes to Neutral sentence pairs (it classifies roughly equal numbers as Neutral and Entailment), it classifies 74% of Entailment pairs as Neutral. For multilingual BERT the situation is reversed, with 47% of Neutral entailments classified as Entailment, although for cases of entailment, the classifier did not clearly distinguish Neutral from Entailment. The most surprising pattern was observed in RobBERT: where RoBERTa still has high recall for Neutral and Entailment, its Dutch counterpart RobBERT mistakes most Neutral cases as Entailment and even more so vice versa. For all models, in these four cases of misclassification, in the case of the English task the correct inference was made in at least 99% of the cases.

Following Naik et al. (2018), we inspect these prominent cases of misclassification in Dutch by looking at the number of cases of high overlap (at most four words not in common), and at the number of length mismatches (the difference between sentence length exceeds 4), and set off these distributions against that of the test set, in Table 4.

	BERT (N→E)	**mBERT** (E→N)	**RobBERT** (N→E)	**RobBERT** (E→N)	**Test**
	Word difference				
EN	66%	42%	62%	44%	40%
NL	47%	38%	41%	38%	28%
	Length mismatch				
EN	24 %	17%	25%	17%	27%
NL	30 %	31%	28%	25%	36%

Table 4: Error analysis of prominent misclassifications.

The main finding here is that word overlap does provide a strong cue in the English dataset, especially given that SICK has more cases (1970) overall than SICK-NL (1385), and that in SICK they are more concentrated in cases of Entailment. Length mismatches occur more often in SICK-NL but seem to provide less of a cue to the models to make strong inference decisions.

BERT		**Prediction EN-NL**				**Prediction EN**				**Prediction NL**			
		C	**N**	**E**	*rec.*	**C**	**N**	**E**	*rec.*	**C**	**N**	**E**	*rec.*
Gold	**C**	549	62	16	88%	46	27	12	54%	22	52	11	26%
	N	35	2341	147	93%	37	147	83	55%	32	116	119	43%
	E	3	143	1035	88%	3	53	167	75%	3	165	55	25%
	pr.	94%	92%	86%		53%	65%	64%		39%	35%	30%	
mBERT													
Gold	**C**	553	69	10	88%	24	47	9	30%	42	31	7	52%
	N	32	2344	106	94%	9	210	89	68%	71	93	144	30%
	E	3	160	1015	86%	0	103	123	54%	15	111	100	44%
	pr.	94%	91%	90%		73%	58%	56%		33%	40%	40%	
RoBERTa													
Gold	**C**	563	68	4	89%	27	46	4	35%	35	28	14	45%
	N	25	2301	82	96%	6	279	97	73%	79	96	207	25%
	E	1	108	995	90%	2	42	256	85%	15	246	39	13%
	pr.	96%	93%	92%		77%	76%	72%		27%	26%	15%	

Table 5: Confusion matrices for English vs Dutch language models, finetuned. Top: BERT vs BERTje. The models disagree in 13.3% of cases. Middle: Multilingual BERT. The model disagrees in 14.3% of cases). Bottom: Roberta vs RobBERT. The models (disagree in 18.3% of cases).

5 Stress Testing

One of the potential sources of error could have been the passive form translation of a verb. Such constructions, combined with a prepositional phrase, form an interesting testbed for Dutch as they allow the prepositional phrase to be moved in front of the verb in a sentence without changing the meaning. For example, "*Een vrouw is aan het wakeboarden op een meer*" ("*A woman is wakeboarding on a lake*"), may in Dutch be used interchangeably with "*Een vrouw is op een meer aan het wakeboarden*"). We select all (87) sentences in SICK-NL that contain both the 'aan het' construction and a prepositional phrase, and generate their permutations. Then, we replace all (225) inference pairs with these sentences such that they now contain a sentence with different word order but the exact same meaning and therefore the inference label is preserved. We then verify how the model's predictions do on those inference pairs that were in the test set (116). Additionally, we check whether the models are able to interchange sentences and their rewritten equivalent (i.e. classify as Entailment).

As a second test, we investigate the role of the simple present versus the present continuous. We take all the (383) cases of present continuous in the Dutch dataset and replace them by a simple present equivalent, leading to 1137 pairs, out of which 576 occur in the test data. For example, we turn the sentence "*De man is aan het zwemmen*" into the simple form "*De man zwemt*". We then repeat the same procedure as above, asking how many inference predictions change as a result of this form change, and whether the forms can be used interchangeably for the models.

	present cont. → present simple			
	Before	After	→	←
BERT	84.55	86.63	93.21	92.43
mBERT	86.11	84.90	94.26	94.52
RobBERT	82.81	81.94	86.16	84.33
	prep. phrase order switch			
BERT	81.03	78.45	85.06	85.06
mBERT	87.93	85.34	85.06	80.46
RobBERT	76.72	75.86	72.41	73.56

Table 6: Stress test accuracy. Left: accuracy before and after rewriting. Right: inference between rewritings.

The results in Table 6 indicate that the interchange between present continuous and simple present forms does not make much of a difference to the models' performance, and interchangeability is high except for RobBERT that scores under 90%. However, switching the order of prepositional phrase and verb has a much stronger effect with all models consistently scoring lower on the relevant part of the test set, and mainly the models being particularly poor at interchanging these sentences that are semantically equivalent.

6 Conclusion

In this paper we introduced an NLI dataset for Dutch by semi-automatically translating the SICK dataset. To our knowledge this is the first available inference task for Dutch. Despite the common perception that Dutch is very similar to English, SICK-NL was significantly more difficult to tackle, even for language models that had access to the training data for fine-tuning. We hypothesised that the difference in result may be due to a larger vocabulary in SICK-NL, and a decline in word overlap between inference pairs. In addition we performed two stress tests and found that pretrained models that were exposed to the training data had difficulty detecting semantically equivalent sentences that differ only in word order. Further work will therefore more systematically assess such phenomena.

Acknowledgments

The authors are grateful to the anonymous reviewers for their valuable feedback, which improved the paper. Both authors acknowledge support from the Dutch Research Council (NWO) under the scope of the project "A composition calculus for vector-based semantic modelling with a localization for Dutch" (360-89-070).

References

Liesbeth Allein, Artuur Leeuwenberg, and Marie-Francine Moens. 2020. Binary and multi-task classification model for Dutch anaphora resolution: Die/dat prediction. *arXiv preprint arXiv:2001.02943*.

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

Benjamin van der Burgh and Suzan Verberne. 2019. The merits of universal language model fine-tuning for small datasets–a case with Dutch book reviews. *arXiv preprint arXiv:1910.00896*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: a Dutch RoBERTa-based language model. *arXiv preprint arXiv:2001.06286*.

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

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. 2013. *Large Scale Syntactic Annotation of Written Dutch: Lassy*, pages 147–164. Springer Berlin Heidelberg, Berlin, Heidelberg.

Nelleke Oostdijk, Martin Reynaert, Véronique Hoste, and Ineke Schuurman. 2013. *The Construction of a 500-Million-Word Reference Corpus of Contemporary Written Dutch*, pages 219–247. Springer Berlin Heidelberg, Berlin, Heidelberg.

Livy Real, Ana Rodrigues, Andressa Vieira e Silva, Beatriz Albiero, Bruna Thalenberg, Bruno Guide, Cindy Silva, Guilherme de Oliveira Lima, Igor C. S. Câmara, Miloš Stanojević, Rodrigo Souza, and Valeria de Paiva. 2018. SICK-BR: A Portuguese corpus for inference. In *Computational Processing of the Portuguese Language*, pages 303–312, Cham. Springer International Publishing.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural*

Language Learning at HLT-NAACL 2003 - Volume 4, CONLL '03, page 142–147, USA. Association for Computational Linguistics.

Stéphan Tulkens, Chris Emmery, and Walter Daelemans. 2016. Evaluating unsupervised Dutch word embeddings as a linguistic resource. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4130–4136, Portorož, Slovenia. European Language Resources Association (ELRA).

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT model. *arXiv preprint arXiv:1912.09582*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

A phonetic model of non-native spoken word processing

Yevgen Matusevych
School of Informatics
University of Edinburgh
`yevgen.matusevych@ed.ac.uk`

Herman Kamper
E&E Engineering
Stellenbosch University
`kamperh@sun.ac.za`

Thomas Schatz
Department of Linguistics & UMIACS
University of Maryland
`tschatz@umd.edu`

Naomi H. Feldman
Department of Linguistics & UMIACS
University of Maryland
`nhf@umd.edu`

Sharon Goldwater
School of Informatics
University of Edinburgh
`sgwater@inf.ed.ac.uk`

Abstract

Non-native speakers show difficulties with spoken word processing. Many studies attribute these difficulties to imprecise phonological encoding of words in the lexical memory. We test an alternative hypothesis: that some of these difficulties can arise from the non-native speakers' phonetic perception. We train a computational model of phonetic learning, which has no access to phonology, on either one or two languages. We first show that the model exhibits predictable behaviors on phone-level and word-level discrimination tasks. We then test the model on a spoken word processing task, showing that phonology may not be necessary to explain some of the word processing effects observed in non-native speakers. We run an additional analysis of the model's lexical representation space, showing that the two training languages are not fully separated in that space, similarly to the languages of a bilingual human speaker.

1 Introduction

Compared to native speakers, non-native speakers perform differently in a variety of tasks related to auditory language processing, both at the phone and at the word level. At the phone level, these tasks usually require speakers to compare individual phones (e.g., phone discrimination or identification), while spoken word processing tasks usually test the implicit activation of a certain word in the memory (e.g., lexical priming, word translation). In some cases, non-native speakers' behavior is consistent across the tasks: lower performance in spoken word processing tasks is directly associated with difficult phone contrasts. For example, upon hearing a word *rock*, Japanese speakers activate both *rock* and *lock* in their lexical memory (Cutler and Otake, 2004), probably because they find it difficult to discriminate the English [ɹ]–[l] phone contrast (Miyawaki et al., 1975). In other cases, however, non-native speakers' behavior in spoken word processing tasks cannot be explained by difficult phone contrasts (Cook et al., 2016; Amengual, 2016; Darcy et al., 2012). For example, in a translation task native English speakers may confuse Russian words *moloko* [məɫʌˈko] ('milk') and *molotok* [məɫʌˈtok] ('hammer'), even though this pair of words does not have a difficult phone contrast (Cook et al., 2016).

This dissociation between the behavior in phone discrimination vs. spoken word processing tasks has been attributed to different kinds of representations involved. On the one hand, thanks to phonetic knowledge, speakers recognize individual phones in a given language. On the other hand, speakers store phonological representations of the words they know in their mental lexicon and use those representations to recognize spoken words (e.g., Pallier et al., 2001). It is often implicitly assumed that any lexical processing effect should be attributed to stored phonological representations of the words (Gor and Cook, 2020; Cook et al., 2016; Cook and Gor, 2015; Darcy et al., 2013, 2012; McQueen et al., 2006). At the same time, *phonetic* effects are not limited to the perception of individual phones:

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1480–1490
April 19 - 23, 2021.

some existing theories argue that phonetic details are encoded in the lexical memory (e.g., Pierrehumbert, 2002; Hawkins, 2003; Port, 2007). This raises the question: can some of the spoken word processing effects normally attributed to phonology be instead explained in terms of phonetic perception?

In this study, we use computational modeling to test a hypothesis that some spoken word processing effects observed in non-native speakers can be explained without involving phonolexical representations—by phonetic perception, which results from phonetic learning, or speakers' attunement to the sounds of their native language (Werker and Tees, 1984). We use a model developed for speech technology applications (Kamper, 2019). Earlier, it was used to simulate early phonetic learning and successfully predicted some infant phone discrimination data (Matusevych et al., 2020b). This model learns from natural speech data that is not segmented at the phone level. It never receives information about individual phones in isolation or about phone-level differences between words. Therefore, the model is not equipped with an explicit mechanism to learn abstract phonolexical representations, making it a good candidate to test our hypothesis: if our model without the knowledge of phonology can correctly predict a particular effect, that effect can at least partially be attributed to phonetic learning. Because our goal is to see whether phonetic perception can explain some of the existing data in principle, even just one phonetic model making correct predictions about the data would be a positive result.

By design, lexical processing tasks require at least minimal knowledge of the target language. That is why they are normally carried out with bilingual speakers (or second language learners: Gor and Cook, 2020; Amengual, 2016; Cook et al., 2016, etc.). In bilingual speakers, the two languages interact at various levels, including lexical (e.g., Weber and Cutler, 2004; Sunderman and Kroll, 2006). To take this into account, we simulate bilingual speakers by training the model on two languages simultaneously.

We present three simulations. The first two show that the model exhibits predictable behaviors in discrimination tasks. In the third one, we present a case study to test whether a lexical processing effect commonly attributed to phonological representations can be explained in terms of phonetic learning alone, without the influence of phonology. In addition, we examine whether the representations in our bilingual model match the pattern observed in bilingual lexical access. Existing studies (Weber and Cutler, 2004; Lagrou et al., 2011; Shook and Marian, 2012, etc.) show that upon the presentation of a word, competitor words in both languages may get activated (non-selective lexical access). We carry out a language classification task with our model, showing that the two languages are not fully separated in its representation space.

2 Method

2.1 Simulations

We train a computational model—a correspondence autoencoder recurrent neural network (CAE-RNN; Kamper, 2019)—on speech data from one or two languages in order to simulate monolingual and bilingual speakers. We then test these different versions of the model on discrimination tasks and compare the observed patterns to those found in human speakers. This general methodological framework is adopted from Schatz et al. (2019).

We run three simulations, described in more detail in the respective sections below. Although it would be ideal to use the same set of languages in each simulation, the choice of languages is limited by available results from studies with human participants. We use language pairs for which human data is available and where our model has previously been tested on the target languages in the monolingual context. In Simulation 1, we look at phone discrimination by infants exposed to two languages (English and Mandarin), showing that the model correctly predicts a discrimination pattern observed in such infants (Kuhl et al., 2003). In Simulation 2, we show that the model can predict discrimination effects at the word level, observed in adult native English speakers and Japanese learners of English (MacKain et al., 1981). In Simulation 3, we show that the result obtained in a translation judgment task with native Russian speakers and English learners of Russian (Cook et al., 2016) can be at least partially explained in terms of phonetic learning, without the effects of phonology.

2.2 Model

The CAE-RNN (Kamper, 2019) is an extension of a recurrent autoencoder (Chung et al., 2016), in which both encoder and decoder are recurrent neural networks. Unlike an autoencoder, the CAE-RNN is trained on pairs of word tokens of the same

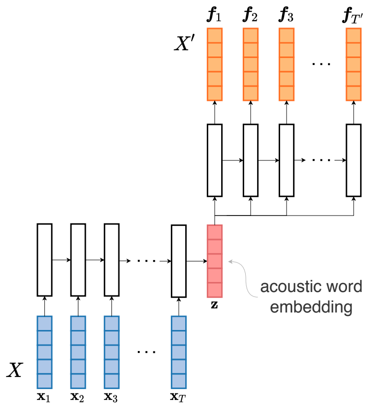


Figure 1: The model learns to reconstruct an acoustic instance of a word, X', from another acoustic instance of the same word, X.

type (e.g., two acoustic instances of the word *apple*). It receives one instance of a word (represented as a speech sequence), encodes it into a vector of a fixed dimensionality (an *acoustic embedding*), and then tries to reconstruct the other instance in the pair, as shown in Figure 1. Formally, each training item is a pair of acoustic words (X, X'). Each word is represented as a sequence of vectors: $X = (\vec{x}_1, \ldots, \vec{x}_T)$ and $X' = (\vec{x}'_1, \ldots, \vec{x}'_{T'})$. The loss for a single training item is:

$$\ell(X, X') = \sum_{t=1}^{T'} ||\vec{x}'_t - \vec{f}_t(X)||^2 \tag{1}$$

where X is the input and X' the target output sequence, and $f_t(X)$ is the t^{th} decoder output conditioned on the embedding $\mathbf{z}$. At inference time, we can encode a sequence of arbitrary duration (e.g., a phone or a word) into a fixed-dimensional acoustic embedding in the model's representation space.

We choose this model because it showed promise for the study of human cognition: it correctly predicted some patterns of infant phonetic learning (Matusevych et al., 2020b), and some of its basic properties are compatible with human auditory cognition and lexical access (Matusevych et al., 2020a). The advantage of this model compared to others (e.g., Schatz et al., 2019) is its ability to represent speech sequences of any duration in a common representation space (the embeddings have a fixed number of dimensions), in which perceptual similarity between sequences can be computed using a simple distance function. The model handles individual phones and acoustic words in exactly the same way, allowing us to easily generalize from phone-level to word-level representations. In addition, the model has been successfully trained on multiple languages for a speech technology application (Kamper et al., 2020a,b), potentially making it a good candidate for simulating bilingual speakers.

A. Training data.

Sim. #	Language	Corpus	Duration (hh:mm)	No. of spk.
1	EN	WSJ[1]	19:30	96
	JA	GlobalPhone[2]	19:33	96
	EN	Buckeye[3]	9:13	20
	JA	CSJ[4]	9:11	20
2	ZH	AIShell[5]	58:59	166
	EN	WSJ	58:49	166
	ZH	GlobalPhone	11:51	48
	EN	WSJ	11:49	48
3	RU	GlobalPhone	11:07	58
	EN	WSJ	11:07	58

B. Test data.

Sim. #	Language	Corpus	Duration (hh:mm)	No. of spk.
1	EN	WSJ	9:39	47
		Buckeye	9:01	20
2	ZH	AIShell	58:45	165
		GlobalPhone	11:51	48
3	RU	GlobalPhone	11:01	57

[1] Wall Street Journal CSR corpus (Paul and Baker, 1992).
[2] Multilingual text and speech database (Schultz, 2002).
[3] Buckeye corpus of conversational speech (Pitt et al., 2005).
[4] Corpus of spontaneous Japanese (Maekawa, 2003).
[5] Open-source Mandarin speech corpus (Bu et al., 2017).

Table 1: Corpus samples used in the simulations.

Following earlier studies (Kamper, 2019; Matusevych et al., 2020b), we first pretrain the model as an autoencoder RNN for 15 epochs without early stopping using the Adam optimization (Kingma and Ba, 2015) with a learning rate of 0.001. We then train the model for 3 epochs on 100k ground truth pairs from either one or two languages as described next. We use 3 hidden layers (400 gated recurrent units each) in both the decoder and the encoder, and an embedding dimensionality of 130.

2.3 Training and test data

The model is trained on isolated words and tested on either phones or words extracted from corpora of natural speech based on existing forced alignments (Matusevych et al., 2020b; Kamper et al., 2020b). All speech data is encoded using a common approach in speech processing: each speech sequence is divided into 25-ms-long frames (sampled every 10 ms), from which 13 Mel-frequency cepstral coefficients (MFCCs) are extracted using Kaldi (Povey et al., 2011).

The subsets of the corpora that we use are listed in Table 1. Within each pair in part A of the table, the subsets are matched on the number of speakers, their gender, and the amount of data per speaker. This ensures that we only compare models trained on the same amount and type of data. In Simulations 1 and 2, we follow the setup from a previous study and use two subsets from different corpora per language. In Simulation 3, we could not obtain two different corpora of Russian speech, and instead train the model five times with different random initializations on the same data.

In case of bilingual models, we train them simultaneously on both languages, using mixed input. We use the relative amount of training data in each language as a simple proxy variable for language proficiency: the higher the model's relative exposure to a language, the higher its proficiency in that language. In bilingual training, we use the same total amount of data as for the corresponding monolingual models—in terms of both the number of tokens (for pretraining the model) and of training pairs. For example, consider a monolingual English model and a monolingual Mandarin model, each trained on 10k tokens and 100k pairs. Then for training a 'balanced' bilingual model we take the 5k most frequent tokens from English and Mandarin each, generate 50k pairs in each language, and use the combined 10k English–Mandarin token data for pretraining and the combined 100k pairs for training the CAE-RNN.

2.4 Simulating discrimination tasks

To test a model's ability to discriminate a particular phonetic or lexical contrast, we use the machine ABX task (Schatz et al., 2013), which is standard in research on zero-resource speech technology for evaluating discriminability of speech units (Versteegh et al., 2015; Dunbar et al., 2017, 2020, etc.) and is commonly used for simulating human speech discrimination tasks (e.g., Martin et al., 2015; Schatz et al., 2019; Millet et al., 2019). The machine ABX task allows us to easily design precise comparisons (e.g., compare words that only differ in 1 phone) and is not sensitive to the absolute distances in the embedding spaces, which may vary across simulations.

In the ABX task, A and X are two instances of the same word type (e.g., *right*), while B is a different word type (e.g., *light*). If A and X are closer to each other in a model's representation space than B and X, the model's prediction is correct, otherwise it is not. Irrespective of the test units (phones, words), an acoustic segment in our model is represented by a single vector. We perform the ABX task directly on the vectors, allowing us to compare segments of different duration (without doing any type of alignment). Following earlier studies, we use angular cosine distance to measure the distance between the stimuli in the embedding space. The model is evaluated by considering the proportion of ABX triplets for which it makes correct predictions: 0% error corresponds to perfect discrimination, and 50% to chance performance.

To test whether the difference between the ABX error rates of several models is significant, we fit mixed-effects regressions to the error rates of these models. Significance for the effect of interest is then determined using two-tailed ANOVA tests (with Satterthwaite degrees of freedom approximation) on the predicted values of the regressions.

3 Simulation 1: Phone discrimination

Previously, the monolingual CAE-RNN was shown to correctly predict the crosslinguistic difference in the discrimination of the Mandarin [ɕ]–[tɕʰ] contrast (Matusevych et al., 2020b), observed in Mandarin-learning vs. English-learning infants (Tsao et al., 2006). Considering this, it may seem trivial to show that a model with some exposure to Mandarin data (bilingual) would also achieve lower error than a model with no such exposure (English monolingual). However, potentially complex interactions between training languages may result in an unpredictable phonetic space. This is why we need to ensure the bilingual model behaves as expected, before we move on to the word-level tasks. To do this, we run a simple sanity check: whether a model trained on two languages behaves in a predictable way on the same Mandarin contrast.

3.1 Setup

We focus on the experiment of Kuhl et al. (2003), who showed that exposing English-learning infants to a small amount of Mandarin Chinese improves their ability to discriminate [ɕ]–[tɕʰ]. Our goal is to test whether the model also correctly predicts the pattern for English-learning infants with vs. without exposure to Mandarin.

It is difficult to estimate how the infants' amount

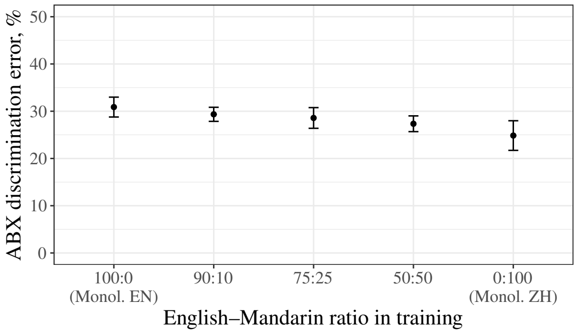


Figure 2: Models' ABX error rates in the Mandarin [ɕ]–[tɕʰ] phone discrimination task. Error bars show standard error of the mean over two training corpus samples × two test samples.

of Mandarin exposure in the experiment maps onto the English–Mandarin ratio of training data in our model. We therefore try the ratios of 90:10 and 75:25 (to simulate an infant with a higher exposure to English than to Mandarin), as well as 50:50 (a control condition, a balanced bilingual). As a baseline to compare our bilingual models to, we train the model on English speech alone (100:0 ratio). For reference, we also train a model on Mandarin speech alone (0:100). Using each model, we embed a set of [ɕ] and [tɕʰ] phones from the test corpus and run a [ɕ]–[tɕʰ] discrimination task. We expect each bilingual model to show lower error than the English monolingual model.

3.2 Results

The ABX error rates of the models (see Figure 2) show the expected pattern: the higher the exposure to Mandarin, the lower the error rates in the target discrimination task. Even having 10% of Mandarin data (90:10 model) on average results in 1.6% reduction in absolute error compared to the monolingual English model. A mixed-effects regression fitted to the models' error rates shows that this difference is not significant, but the other two bilingual models do show significantly lower error rate than the monolingual English model: we observe a 2.2% error reduction in the 75:25 model and 3.5% in the 50:50 model compared to the English baseline. This suggests that even a relatively small amount of training data in a given language (under 25%) can improve the model's ability to discriminate between some contrasts in that language, consistent with the empirical findings of Kuhl et al. (2003) with infants. To summarize, the bilingual CAE-RNN model behaves as we expected: it can correctly predict infant-like behavior in phone discrimination. In the next simulation, we test the model on a word discrimination task.

4 Simulation 2: Word discrimination

Simulation 2 tests whether our bilingual model behaves in predictable ways at the word level. Recall that it represents a sequence of any given duration as a fixed-dimensional vector. The compression of a dynamic speech sequence that unfolds in time into a 'static' vector results in information loss. Since words are normally longer speech sequences than phones, it is not obvious whether the model's behavior in word discrimination and phone discrimination will be consistent. To examine this, we test the model on minimal pairs of words with [ɹ]–[l], a phone contrast on which the monolingual CAE-RNN previously showed an infant-like crosslinguistic discrimination pattern (Matusevych et al., 2020b). As before, we train the model on one or two languages and see if it behaves in a predictable way, this time in a *word* discrimination task.

4.1 Setup

MacKain et al. (1981) tested adult native speakers of American English and native Japanese learners of English on the discrimination of English words *rock–lock* (i.e., [ɹ]–[l] contrast). Learners with low English proficiency scored nearly at chance in this task, while highly proficient learners showed the discrimination scores close to those of native English speakers. This result is also in line with studies showing that native Japanese speakers' discrimination of [ɹ]–[l] can improve after relevant phonetic training in English under certain conditions (e.g., Strange and Dittmann, 1984; Logan et al., 1991; Bradlow et al., 1997; Iverson et al., 2005). Our goal is to test whether our model can correctly predict this word-level discrimination pattern.

Similarly to Simulation 1, we train the model on Japanese (for reference) or English speech alone (as the baseline to compare to), or on a combination of the two in proportion 90:10, 75:25, or 50:50, to simulate native Japanese learners of English with variable proficiency. For each model, we embed a set of acoustic words ([ɹ]–[l] minimal pairs) from the test corpus. As in the original experiment, we compare the bilingual models to the native English model: to make correct predictions, a bilingual model needs to show higher ABX discrimination error than the English model. Also, we expect

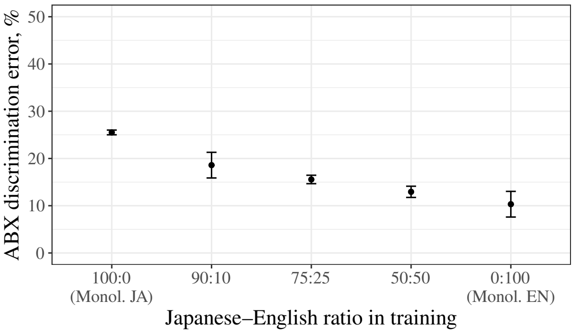


Figure 3: Models' ABX error rates in the word discrimination task with [ɹ]–[l] minimal pairs (e.g., *rock–lock*). Error bars show standard error of the mean over two training × two test corpus samples.

the error rate to decrease with higher exposure to English: 90:10 $>$ 75:25 $>$ 50:50.

4.2 Results

It is clear from Figure 3 that the monolingual Japanese model (100:0) shows higher ABX error rate on the word discrimination task than the monolingual English model (0:100). This extends the previous result for phone-level [ɹ]–[l] discrimination (Matusevych et al., 2020b) to word level.

Comparing all the models, we observe that the error rate decreases as the relative amount of English exposure increases. Even 10% of English in the training data (90:10 model) improves the discrimination by 6.9% in absolute error rate compared to the monolingual Japanese model (25.5% vs. 18.6%), and more so for the models with higher English exposure. A mixed-effects regression fitted to the error rates shows that the pairwise differences between most models are statistically significant, except that the 50:50 model shows error rates too close to both its neighbors: 75:25 and 0:100. Despite that, the expected trend is still present. In other words, our model successfully replicates the direction of the main effect in MacKain et al. (1981): the discrimination error rate decreases with higher English exposure.

The result shows that our bilingual model behaves in a predictable way at the word level, ruling out potentially damaging effect that crosslinguistic interactions can have on its lexical representation space. With this knowledge, in the next simulation we proceed with applying our model to a spoken word processing task.

5 Simulation 3: Spoken word processing

In this section, we present a case study to show how the model can be used to get a better understanding of spoken word processing. Specifically, we are interested to know if some of the effects reported in the literature can be explained in terms of phonetic learning alone. Difficulties with spoken word processing have been attributed to imprecise phonological encoding of non-native lexical representations (Gor and Cook, 2020; Cook et al., 2016; Cook and Gor, 2015; Darcy et al., 2013, 2012), which results in a spurious activation of similarly sounding competitor words. To give an example from Cook et al. (2016), if the word *parent* is encoded as [pɛrə(n)t], with an optional [n], it may often be confused with *parrot* [pɛrət].

We focus on one of the experiments in Cook et al. In a translation judgment task, native Russian speakers (proficient in English) and native English speakers (learning Russian) heard a Russian word (e.g., *moloko* [məɫʌˈko] 'milk') and then saw an English word (e.g., *hammer*, which translates into Russian as 'molotok' [məɫʌˈtok]). The participants had to decide if the English word was a good translation of the Russian one. Cook et al. manipulated the phone edit distance between the true translation and the competitor word: in the example above, the distance between [məɫʌˈko] and [məɫʌˈtok] is 2. They found that non-native speakers made more mistakes than native speakers, and that increasing the phone edit distance between the target words decreased the size of this effect. They explain the effect by ambiguous ('fuzzy') non-native phonolexical representations. It is unclear, however, whether lexical phonology is necessary to explain the observed effect. To answer this question, we test whether our model with no access to phonology can correctly predict the described effect.

Clearly, our model does not know anything about word meanings and cannot be tested on a translation task. Instead, we use a series of ABX discrimination tasks to test whether the lower performance of non-native speakers can be explained in terms of acoustic embeddings of individual word tokens in the model's representation space. Recall that the model has no access to phonology, and its acoustic embeddings result from phonetic learning alone.

Figure 4 shows the error rates of human participants in Cook et al. (2016). The patterns that we focus on are: (1) lower error rate is associated with higher proficiency (exposure to Russian) and higher

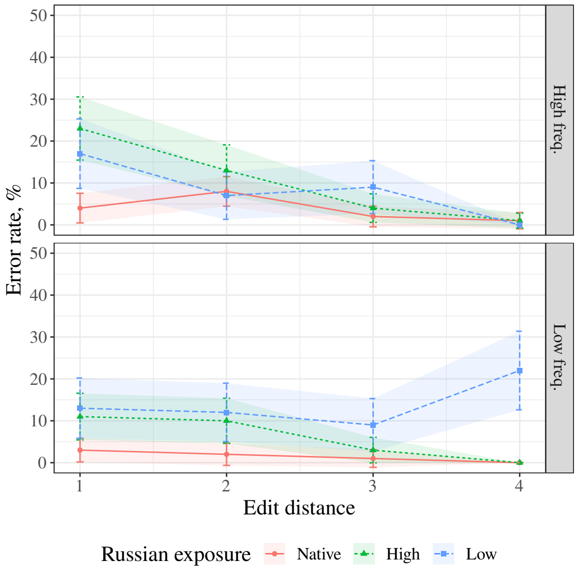


Figure 4: Average error rate of participants in the Russian translation judgment task of Cook et al. (2016) depending on their amount of exposure to Russian and the edit distance between the target and the competitor word. Error bars show mean standard error over participants. Based on Table 3 in Cook et al. (2016).

edit distance; and (2) the difference between the proficiency groups is the highest when the phone edit distance between the target and the competitor word is low. Cook et al. also looked at the effect of competitor word frequency (hence the two panels in Figure 4), but we do not consider this effect here.

5.1 Setup

We train the model on only Russian or only English data (for reference), and also on the combination of the two in proportion 75:25 (to simulate native Russian speakers with some knowledge of English), 50:50, 25:75, or 10:90 (to simulate native English speakers with variable proficiency in Russian). Each model is trained five times with different random initializations.

For testing, we prepare four ABX discrimination tasks: in each task, the words A and X are of the same type, and the words B and X differ in 1, 2, 3 or 4 phones (phone edit distance). We could not obtain the list of the original stimuli from Cook et al. (2016), therefore we sample ABX triplets from our test corpus subset. Following the original experiment, we only consider words containing 4–10 phones. Furthermore, we only consider triplets in which all pairwise ratios of the absolute durations of the words are within the factor of 1.1 (we know from previous work that this model is sensitive to the absolute duration of the test stimuli) and in which B and X are not morphological forms of the same word (such stimuli could not have been used in the original translation judgment task by design). We then sampled 5000 triplets per task, except for edit distance 1 we only had 766 triplets.

As in the original experiment, we first look at the error rates within each model: the error rates are expected to decrease with greater edit distance. Second, we compare the bilingual models to each other: to match the findings from the human study, models with less exposure to Russian (50:50, 25:75, 10:90) must show a higher ABX discrimination error than the model with more exposure (75:25). As a sanity check, we also consider the monolingual Russian (100:0) and English (0:100) models.

5.2 Results

Figure 5 shows the model's ABX error rates across tasks and training conditions. We first observe that all lines have a negative slope: all the six models show lower error in the tasks with greater edit distance between the words, which is the expected pattern. Note that comparing the absolute values to the results of human participants (Figure 4) is not necessarily meaningful because of the task difference: human participants of Cook et al. (2016) had to compare an acoustic word in Russian to a translation of an English word they saw, whereas our model directly compared two Russian acoustic words embedded in its representation space.

Second, we see in Figure 5 that the models with less exposure to Russian have higher error rates. This is especially evident for the data points with edit distance 1, whereas the difference across the models gets smaller with greater edit distances. Again, this is the expected pattern. A mixed-effects regression fitted to the models' error rates suggests that there are significant effects of (1) the amount of Russian language exposure (higher exposure is associated with lower error) and (2) the edit distance (higher edit distance is associated with lower error). A similar pattern is observed when we only consider the bilingual models, for a better analogy with the original experiment: the 75:25 model shows significantly lower error rates than both the 10:90 model and the 25:75 model (but not the 50:50 model), and also the error rates generally get lower as the edit distance increases.

Recall that in each discrimination task, we considered pairs of words with a certain edit distance.

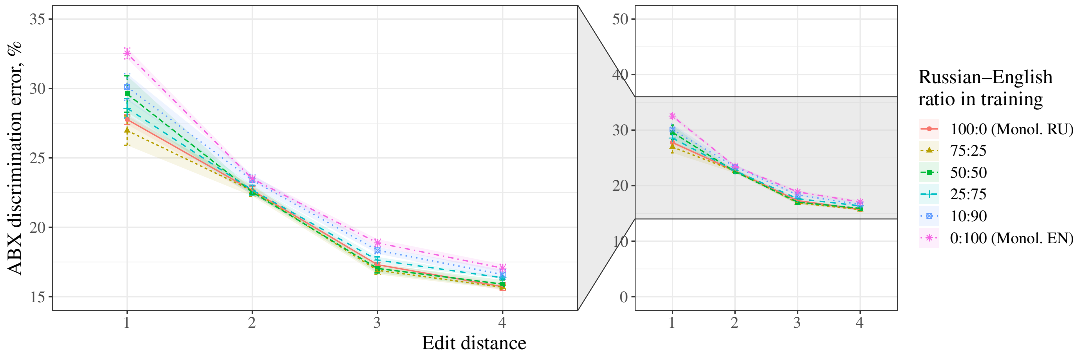


Figure 5: Average ABX discrimination error rate depending on the amount of the model's exposure to Russian and the edit distance between the words in ABX triplets. Error bars show mean standard error over random model initializations.

Most edit operations involve Russian phone contrasts that are also linguistically meaningful in English (e.g., [v]–[s]). However, there is a small number of Russian phone contrasts that are allophones of the same phoneme in English (e.g., [d]–[dʲ]). If our data included a substantial number of contrasts of this type, the model's higher error rates could be attributed to these difficult phone contrasts. To ensure that was not the case, we looked at the contrasts in our pairs with edit distance 1 and found that out of 29 contrasts present in that data, only 1 ([ɫ]–[lʲ]) was not phonemic in English, and excluding the corresponding test pairs from the analysis had only minor impact on the absolute error rates, but not on the reported patterns.

To summarize, our model could correctly predict the direction of the two main effects found in a translation judgment task of Cook et al. (2016). This suggests that their result can at least partially be explained in terms of comparing two acoustic instances: the word a participant hears and the translation of a word that (s)he sees. This presents an alternative explanation of the non-native speakers' difficulties with spoken word processing in terms of phonetic perception, which does not involve phonology.

6 Analyzing the model's representations

Most studies on bilingual lexical access advocate its non-selective nature: that is, speakers activate words in both languages in parallel, including in spoken word processing (e.g., Weber and Cutler, 2004; Lagrou et al., 2011; Shook and Marian, 2012). Ideally, our model should show a similar pattern and not completely separate the two languages in its representation space. To examine this, we run a language classification task, similar to Kamper et al. (2020a). We are interested whether the model can identify the language of a given word based on its acoustic embedding.

	Russian–English ratio			
	75:25	50:50	25:75	10:90
Mean	84.6	86.5	84.3	80.3
SD	1.0	0.9	0.4	1.6

Table 2: Accuracy (in %) of logistic regression classifiers predicting language identity of a given word from its acoustic embedding, averaged over five random initializations of each model.

Using the bilingual models from Simulation 3, we embed 5000 words per language. We then train a logistic regression classifier on 80% of this data to predict the language of a given word from its acoustic embedding, and test the classifier on the remaining 20% of words. The higher the accuracy of the classifier, the more linearly separable the two languages—specifically, their lexicons—in our model's representations. The results (Table 2) show that all models reach accuracy much higher than the 50% chance, although no model reaches 100% accuracy. This means that the lexical representations of words in two languages (acoustic word embeddings) in our bilingual models are not fully linearly separable, indicating a substantial (13.5–19.7%) overlap between the two languages. Because some of the representations from the two languages are close to each other in the embedding space, the model may confuse them, similar to the non-selective lexical access in bilingual speakers.

7 Discussion

We started by asking whether some of the difficulties in non-native spoken word processing can be explained at the level of phonetic perception, without involving phonolexical representations. To address this question, we presented a case study (Simulation 3) with a computational model that learns from unsegmented speech data and does not have access to phonology. Our model showed patterns similar to those found by Cook et al. (2016) in human speakers. This suggests that their results can be at least partly explained by phonetic learning. While we cannot estimate the relative contribution of the two factors—non-native phonetic perception vs. imprecise phonolexical representations—to the behavior of non-native speakers in the experiment of Cook et al., we argue that both factors need to be considered as possible explanations of the spoken word processing difficulties in non-native speakers. Note, however, that this result does not tell us whether the phonetic or the phonolexical explanation is more parsimonious—a question that should be addressed in the future.

One could interpret our main result differently: that our model, in fact, has succeeded in learning phonological systems from speech data and cannot be considered a purely phonetic model. Indeed, we know that deep neural networks can learn to encode various types of linguistic structure without explicit supervision (e.g., Manning et al., 2020; Linzen and Baroni, 2021). In particular, speech models can achieve high accuracy in phone discrimination (Alishahi et al., 2017) and classification (Chung et al., 2019), a finding sometimes interpreted as a successful acquisition of phonetic/phonological *categories*. While our model can discriminate at least some phone contrasts, too (Simulation 1), this does not necessarily mean that it learns phonetic categories (see Schatz et al., 2019, for a relevant discussion). More importantly, what our model does not do is store explicit phonolexical representations in its memory, whereas the (imprecise) storage of word forms is one of the key premises of the phonolexical account explaining non-native speakers' difficulties in spoken word processing (Cook et al., 2016). Therefore, we conclude that our results highlight the effects of phonetic perception on non-native word processing.

In Simulation 1 and 2 we showed that our model trained simultaneously on two languages could correctly predict some phone- and word-level discrimination effects in infants and adults (Kuhl et al., 2003; MacKain et al., 1981). This extends previous results on phone discrimination with monolingual model (Matusevych et al., 2020b) to word discrimination and to bilingual speakers. Also, our analysis of model's representations indicates a substantial overlap between the lexicons of the two languages, mimicking non-selective lexical access in bilingual speakers (e.g., Lagrou et al., 2011). All together, this suggests that the CAE-RNN can be used as a tool to study not only native/non-native phonetic learning, but also native/non-native spoken word processing, including in bilingual speakers.

Our model helps to tease apart the potential impact of phonetic learning from other effects on spoken word processing. At the same time, it is not a cognitive model of the human mental lexicon, for example because it is devoid of semantics. A method to learn acoustic and semantic embeddings in parallel has been proposed in speech engineering (Chen et al., 2018), and future research could shed some light on whether this method can be used for studying human mental lexicon.

Acknowledgments

This work is based on research supported in part by an ESRC-SBE award ES/R006660/1, a JSMF Scholar Award 220020374, and an NSF award BCS-1734245. We thank the anonymous reviewers, as well as Sameer Bansal, Kate McCurdy, Seraphina Goldfarb-Tarrant and other members of AGORA reading group at the University of Edinburgh for their helpful feedback.

References

Afra Alishahi, Marie Barking, and Grzegorz Chrupała. 2017. Encoding of phonology in a recurrent neural model of grounded speech. In *Proceedings of CoNLL*, pages 368–378.

Mark Amengual. 2016. The perception of language-specific phonetic categories does not guarantee accurate phonological representations in the lexicon of early bilinguals. *Applied Psycholinguistics*, 37:1221–1251.

Ann R. Bradlow, David B. Pisoni, Reiko Akahane-Yamada, and Yoh'ichi Tohkura. 1997. Training Japanese listeners to identify English /r/ and /l/: IV. Some effects of perceptual learning on speech production. *The Journal of the Acoustical Society of America*, 101:2299–2310.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. AISHELL-1: An open-source Mandarin speech corpus and a speech recognition baseline. In *Proceedings of O-COCOSDA*, pages 58–62.

Yi-Chen Chen, Sung-Feng Huang, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. 2018. Phonetic-and-semantic embedding of spoken words with applications in spoken content retrieval. In *Proceedings of IEEE SLT Workshop*, pages 941–948.

Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James R. Glass. 2019. An unsupervised autoregressive model for speech representation learning. In *Proceedings of Interspeech*, pages 146–150.

Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. 2016. Unsupervised learning of audio segment representations using sequence-to-sequence recurrent neural networks. In *Proceedings of Interspeech*, pages 765–769.

Svetlana V. Cook and Kira Gor. 2015. Lexical access in L2: Representational deficit or processing constraint? *The Mental Lexicon*, 10:247–270.

Svetlana V. Cook, Nick B. Pandža, Alia K. Lancaster, and Kira Gor. 2016. Fuzzy nonnative phonolexical representations lead to fuzzy form-to-meaning mappings. *Frontiers in Psychology*, 7:1345.

Anne Cutler and Takashi Otake. 2004. Pseudo-homophony in non-native listening. Paper presented at the 174th meeting of the Acoustical Society of America.

Isabelle Darcy, Danielle Daidone, and Chisato Kojima. 2013. Asymmetric lexical access and fuzzy lexical representations in second language learners. *The Mental Lexicon*, 8:372–420.

Isabelle Darcy, Laurent Dekydtspotter, Rex A. Sprouse, Justin Glover, Christiane Kaden, Michael McGuire, and John H. G. Scott. 2012. Direct mapping of acoustics to phonology: On the lexical encoding of front rounded vowels in L1 English–L2 French acquisition. *Second Language Research*, 28:5–40.

Ewan Dunbar, Xuan-Nga Cao, Juan Benjumea, Julien Karadayi, Mathieu Bernard, Laurent Besacier, Xavier Anguera, and Emmanuel Dupoux. 2017. The Zero Resource Speech Challenge 2017. In *Proceedings of ASRU Workshop*, pages 323–330.

Ewan Dunbar, Julien Karadayi, Mathieu Bernard, Xuan-Nga Cao, Robin Algayres, Lucas Ondel, Laurent Besacier, Sakriani Sakti, and Emmanuel Dupoux. 2020. The Zero Resource Speech Challenge 2020: Discovering discrete subword and word units. In *Proceedings of Interspeech*, pages 4831–4835.

Kira Gor and Svetlana V. Cook. 2020. A mare in a pub? Nonnative facilitation in phonological priming. *Second Language Research*, 36:123–140.

Sarah Hawkins. 2003. Roles and representations of systematic fine phonetic detail in speech understanding. *Journal of Phonetics*, 31:373–405.

Paul Iverson, Valerie Hazan, and Kerry Bannister. 2005. Phonetic training with acoustic cue manipulations: A comparison of methods for teaching English /r/-/l/ to Japanese adults. *The Journal of the Acoustical Society of America*, 118:3267–3278.

Herman Kamper. 2019. Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models. In *Proceedings of ICASSP*, pages 6535–3539.

Herman Kamper, Yevgen Matusevych, and Sharon Goldwater. 2020a. Improved acoustic word embeddings for zero-resource languages using multilingual transfer. *arXiv:2006.02295*.

Herman Kamper, Yevgen Matusevych, and Sharon Goldwater. 2020b. Multilingual acoustic word embedding models for processing zero-resource languages. In *Proceedings of ICASSP*, pages 6414–6418.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Patricia K. Kuhl, Feng-Ming Tsao, and Huei-Mei Liu. 2003. Foreign-language experience in infancy: Effects of short-term exposure and social interaction on phonetic learning. *Proceedings of the National Academy of Sciences*, 100:9096–9101.

Evelyne Lagrou, Robert J. Hartsuiker, and Wouter Duyck. 2011. Knowledge of a second language influences auditory word recognition in the native language. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 37:952–965.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7:195–212.

John S. Logan, Scott E. Lively, and David B. Pisoni. 1991. Training Japanese listeners to identify English /r/ and /l/: A first report. *The Journal of the Acoustical Society of America*, 89:874–886.

Kristine S. MacKain, Catherine T. Best, and Winifred Strange. 1981. Categorical perception of English /r/ and /l/ by Japanese bilinguals. *Applied Psycholinguistics*, 2:369–390.

Kikuo Maekawa. 2003. Corpus of Spontaneous Japanese: Its design and evaluation. In *Proceedings of SSPR Workshop*, pages 7–12.

Christopher D. Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, pages 30046–30054.

Andrew Martin, Thomas Schatz, Maarten Versteegh, Kouki Miyazawa, Reiko Mazuka, Emmanuel Dupoux, and Alejandrina Cristia. 2015. Mothers speak less clearly to infants than to adults: A comprehensive test of the hyperarticulation hypothesis. *Psychological Science*, 26:341–347.

Yevgen Matusevych, Herman Kamper, and Sharon Goldwater. 2020a. Analyzing autoencoder-based acoustic word embeddings. Paper presented at Bridging AI and Cognitive Science (ICLR 2020).

Yevgen Matusevych, Thomas Schatz, Herman Kamper, Naomi H. Feldman, and Sharon Goldwater. 2020b. Evaluating computational models of infant phonetic learning across languages. In *Proceedings of CogSci*, pages 571–577.

James M. McQueen, Anne Cutler, and Dennis Norris. 2006. Phonological abstraction in the mental lexicon. *Cognitive Science*, 30:1113–1126.

Juliette Millet, Nika Jurov, and Ewan Dunbar. 2019. Comparing unsupervised speech learning directly to human performance in speech perception. In *Proceedings of CogSci*, pages 2358–2364.

Kuniko Miyawaki, James J. Jenkins, Winifred Strange, Alvin M. Liberman, Robert Verbrugge, and Osamu Fujimura. 1975. An effect of linguistic experience: The discrimination of [r] and [l] by native speakers of Japanese and English. *Perception & Psychophysics*, 18:331–340.

Christophe Pallier, Angels Colomé, and Núria Sebastián-Gallés. 2001. The influence of native-language phonology on lexical access: Exemplar-based versus abstract lexical entries. *Psychological Science*, 12:445–449.

Douglas B. Paul and Janet M. Baker. 1992. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the Workshop on Speech and Natural Language*, pages 357–362.

Janet B. Pierrehumbert. 2002. Word-specific phonetics. *Laboratory Phonology*, 7:101–140.

Mark A. Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond. 2005. The Buckeye corpus of conversational speech: Labeling conventions and a test of transcriber reliability. *Speech Communication*, 45:89–95.

Robert Port. 2007. How are words stored in memory? Beyond phones and phonemes. *New Ideas in Psychology*, 25:143–170.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovský, Georg Stemmer, and Karel Veselý. 2011. The Kaldi speech recognition toolkit. In *Proceedings of ASRU Workshop*.

Thomas Schatz, Naomi H. Feldman, Sharon Goldwater, Xuan-Nga Cao, and Emmanuel Dupoux. 2019. Early phonetic learning without phonetic categories—insights from large-scale simulations on realistic input. *PsyArXiv*.

Thomas Schatz, Vijayaditya Peddinti, Francis Bach, Aren Jansen, Hynek Hermansky, and Emmanuel Dupoux. 2013. Evaluating speech features with the minimal-pair ABX task: Analysis of the classical MFC/PLP pipeline. In *Proceedings of Interspeech*, pages 1781–1785.

Tanja Schultz. 2002. GlobalPhone: a multilingual speech and text database developed at Karlsruhe University. In *Proceedings of ICSLP*, pages 345–348.

Anthony Shook and Viorica Marian. 2012. Bimodal bilinguals co-activate both languages during spoken comprehension. *Cognition*, 124:314–324.

Winifred Strange and Sibylla Dittmann. 1984. Effects of discrimination training on the perception of /r-l/ by Japanese adults learning English. *Perception & Psychophysics*, 36:131–145.

Gretchen Sunderman and Judith F. Kroll. 2006. First language activation during second language lexical processing: An investigation of lexical form, meaning, and grammatical class. *Studies in Second Language Acquisition*, pages 387–422.

Feng-Ming Tsao, Huei-Mei Liu, and Patricia K. Kuhl. 2006. Perception of native and non-native affricate-fricative contrasts: Cross-language tests on adults and infants. *The Journal of the Acoustical Society of America*, 120:2285–2294.

Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan-Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. 2015. The Zero Resource Speech Challenge 2015. In *Proceedings of Interspeech*, pages 3169–3173.

Andrea Weber and Anne Cutler. 2004. Lexical competition in non-native spoken-word recognition. *Journal of Memory and Language*, 50:1–25.

Janet F. Werker and Richard C. Tees. 1984. Cross-language speech perception: Evidence for perceptual reorganization during the first year of life. *Infant Behavior and Development*, 7:49–63.

Bootstrapping Relation Extractors using Syntactic Search by Examples

Matan Eyal[1] **Asaf Amrami**[1, 2] **Hillel Taub-Tabib**[1] **Yoav Goldberg**[1, 2]
[1]Allen Institute for AI, Tel Aviv, Israel
[2]Bar Ilan University, Ramat-Gan, Israel
`matane,asafa,hillelt,yoavg@allenai.org`

Abstract

The advent of neural-networks in NLP brought with it substantial improvements in supervised relation extraction. However, obtaining a sufficient quantity of training data remains a key challenge. In this work we propose a process for bootstrapping training datasets which can be performed quickly by non-NLP-experts. We take advantage of search engines over syntactic-graphs (Such as Shlain et al. (2020)) which expose a friendly by-example syntax. We use these to obtain positive examples by searching for sentences that are syntactically similar to user input examples. We apply this technique to relations from TACRED and DocRED and show that the resulting models are competitive with models trained on manually annotated data and on data obtained from distant supervision. The models also outperform models trained using NLG data augmentation techniques. Extending the search-based approach with the NLG method further improves the results.

1 Introduction

The goal of Relation Extraction (RE) is to find and classify instances of certain relations in raw text. We denote a binary relation instance, *i.e.* a relation instance with two arguments, with a tuple $\boldsymbol{x} = (\boldsymbol{s}, e^1, e^2, r)$, where $\boldsymbol{s} = [w_0 \cdots w_n]$ is a sequence of sentence tokens, e^1, e^2 are entity mentions within $\boldsymbol{s}$ corresponding to the first and second relation argument, respectively, and $r \in R \cup \{\varnothing\}$ is a relation label from a set of predefined relations of interest, or an indication of 'no-relation'. In binary classification our goal is to classify whether, according to $\boldsymbol{s}$, the entity mentions, e^1 and e^2, satisfy r, the relation label. For such classification we require a training dataset X, comprised of X_p, a set of positive examples, representing the relation of interest, and X_n, a set of negatives examples.

The success of recent papers (Soares et al., 2019; Murty et al., 2020) in supervised RE is fueled by advances in deep learning, but also, crucially, by the availability of a large training set such as TACRED (Zhang et al., 2017), containing tens of thousands of training examples. For most relations of interest, such training data is not available.

In this work we examine methods to inexpensively construct X_p and X_n, in cases where a training set is not available. We are especially interested in constructing the positive set, X_p.

In contrast to common NLP tasks like POS tagging, entity extraction and dependency parsing, the task of relation extraction exhibits a much larger degree of label sparsity. For some relations, even when considering only sentences with entities of the relevant types, the ratio between positive and negative examples is highly skewed toward the latter and obtaining a modest amount of positive examples will require a laborious annotation effort (see §3). While manual annotation of large datasets is a viable approach, it typically requires contracting a team of professional annotators (Doddington et al., 2004; Ellis et al., 2015) or crowd workers (Zhang et al., 2017; Yao et al., 2019) and is not well suited for smaller projects or for ad-hoc extraction tasks.

Our main contribution in this paper is a new methodology built on top of Shlain et al. (2020) for cheaply obtaining large datasets (§6). Shlain et al. (2020) proposed a syntactic search engine that given a lightly annotated example sentence, retrieves new sentences with a similar syntactic structure from a pre-annotated dataset. Our syntactic search bootstrapping method requires a small number of manually curated positive example sentences. Then the search engine matches are used as training data for ML models. We evaluate this approach comparing to human annotated data of varying sizes.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1491–1503
April 19 - 23, 2021.

While this method shows promising results with very few user input examples, we also test the impact on performance when more examples are used. One technique for obtaining an abundance of examples uses recent Natural Language Generation (NLG) models (§7.1). It has been shown in recent papers (Wei and Zou, 2019; Anaby-Tavor et al., 2019; Kumar et al., 2020; Amin-Nejad et al., 2020; Russo et al., 2020) that generating abundance of training examples can improve classifier performance. We aim to check whether this can improve our syntactic search method as well.

We evaluate the proposed methodologies by training DL classifiers on the obtained data.

We show that: (1) Syntactic patterns are competitive at bootstrapping training data for ML, even with as little as 3 patterns;
(2) Training DL models over the output of syntactic patterns can significantly improve both recall and F1 over a rule based approach which uses the patterns directly;
(3) Training ML models over the output of syntactic patterns performs better than training models over recently popular NLG data augmentation techniques;
(4) Augmenting the output of syntactic patterns using NLG techniques is often helpful;
(5) Different relations benefit from different strategies.

The code for all our experiments alongside the generation outputs is publicly available[1].

2 Related Work

Distant Supervision. Since its introduction, Distant Supervision (Mintz et al., 2009) has established itself as a viable alternative to manual annotation. Distant Supervision assumes the availability of a knowledge base (KB) of $\langle e^1, r, e^2\rangle$ triplets where e^1, e^2 are entities known to satisfy relation r. To obtain training examples for a relation r, we sample sentences from a large background corpus: sentences which include entity pairs listed in the KB as satisfying r are labeled positive, the remaining sentences are labeled negative (potentially after satisfying additional constraints). While effective in some cases, the reliance on large pre-existing KBs is a significant limitation. Such KBs are not usually available and the cost of constructing them is high.

Bootstrapping from Rules, Snorkel. To eliminate the reliance on external KBs, Angeli et al. (2015) used the predictions of a rule based extractor on a large corpus to train a first iteration of a statistical extractor. They then continued to refine the extractor through self-training.

Another system which can optionally utilize rules instead of external KBs is Snorkel (Ratner et al., 2017). Snorkel is implementing the data-programming paradigm (Ratner et al., 2016) where ML models are trained in three stages: (i) users write labeling functions that weakly label data points using arbitrary heuristics (*e.g.* extraction rules); (ii) the system learns a re-weighted combination of the labeling functions by explicitly modeling the actual distribution of each class. The results are often precise but low-recall; and (iii) The system uses discriminative models to increase recall while preserving precision.

The techniques used by Angeli et al. (2015) and Snorkel can be effective in increasing the accuracy of the initial labeling rules, but coming up with “good enough" initial rules remains a major challenge. In this sense, the search-based methods suggested in this work for bootstrapping RE datasets are complimentary and can be plugged in as a first step in these multi-step solutions.

Only few papers can be directly compared to our paper and use matches as training-data for ML classifiers. One paper similar in that sense is Angeli et al. (2013) which claims that training a classifier using search-based examples works better than traditional bootstrapping methods. See §6.2 for further compression with Angeli et al. (2013).

Augmentation Through Generation. Similarly to our Example Generation approach, recent papers (Anaby-Tavor et al., 2019; Kumar et al., 2020) suggest using pre-trained language models for data augmentation. In both these papers, the authors suggest prepending class labels to generative models in order to augment the number of instances for classes with a small number of examples. In contrast to these papers we use language models in a zero-shot context, and rather than requiring existing labeled examples of the relevant relation, we propose to manually label the generated samples.

3 The data annotation challenge

In contrast to linguistic annotation tasks such as parts-of-speech, syntactic-trees or semantic roles, annotating data for relation-extraction does not require special expertise. Annotation can be easily

---

[1] github.com/mataney/BootstrappingRelationExtractors

performed by a motivated native speaker of the language (in case of "every-day" relations such as those available in TACRED and DocRED) or by a domain expert (in case of "specialized" relations such as in biomedicine or law). Annotating a given sentence for a given relation takes roughly the time it takes to read and understand the sentence. So what stops us from obtaining large amounts of annotated data for ML?

The annotation challenge lies in *relation sparsity* in the wild. In an attempt to get a perspective on this issue, let's consider the *founded-by* relation between a PERSON and an ORG, as attested in the TACRED corpus. Assuming we consider only sentences that contain both a person mention and an organization mention, how many sentences do we have to annotate before we reach, for example, 10 positive examples? The TACRED training set has 124 *founded-by* instances, as well as 6947 "negative" instances with matching entity types ("negative" examples are either other relations, or *no-relation*). This 1-out-of-57 ratio indicates that we will likely sample 56 "negative" sentences before hitting a positive instance.[2] This ratio is overly optimistic, as the annotations in the TACRED corpus are already very skewed in favor of positive examples. Even under this very optimistic scenario, we will need to annotate 570 sentences to recover 10 positive examples. The cost of annotation, then, is not in annotating each individual positive sentence, but in finding the sentences to annotate in the first place. Therefore, we should seek for methods that point towards probable positive instances.

In this paper, we present two methods, the first returns close to 1-out-of-1 positive ratio, although with low syntactic diversity, and a second method with roughly 1-out-of-3 positive ratio.

4 Problem Statement and Setup

We are interested in the problem of obtaining a relation classifier for a binary relation, when no a-priori annotated training data for this relation is available. We seek a methodology that will allow to create an effective extractor, using a minimal amount of data annotation effort.

We compare four approaches – manual annotation, syntactic-search, manual annotation over generated examples, and a combination of the last two – to be described in later sections. Here, we discuss setup which is shared to all experiments.

In order to evaluate the methodology on multiple datasets with similar relations, we chose a set of relations that appear in both the TACRED (Zhang et al., 2017) and DocRED (Yao et al., 2019) datasets with at least 50 development examples[3].

To quantify the performance of our methodology we assess it comparing to varying amounts of manually annotated data. In our settings, large amounts of supervised examples represent upper bound for our bootstrapping methods and are not expected.

While relation extraction is often considered as a multi-class classification problem ("find the occurrences of any of these possible relations"), we instead treat the relations separately, training a binary classifier for each one. We believe this is more representative of a user who wishes to target a low number of relations, who is likely to conduct data collection and evaluation for one relation at a time.

Obtaining Negative Examples When training a binary classifier, it is required to include a set of negative examples alongside the list of positive examples. In all our experiments we obtain negative examples by looking for sentences that contain entity types that are compatible with the relation (i.e, for the *founded-by* relation we sample sentences that include both a PERSON and an ORG). In our syntactic based methods we sample from the same domain as our positive examples (Wikipedia) and then filter this list by removing sentences in which the entities are connected by a syntactic pattern which is attested by the positive examples. For the supervised baselines of various sizes, we obtain negative examples by sampling them from the annotated training set, without replacement.[4]

Datasets We used two datasets to explore our different methods. TACRED (Zhang et al., 2017), a large-scale multi-class relation extraction dataset built over newswire and web text. And DocRED (Yao et al., 2019), a dataset for document level RE, and similarly designed for multi-class prediction. Per our setup above, we changed the setting of both datasets to per relation binary classi-

---

[2] See Appendix A for similar distributions over all relations.

[3] *org:country of headquarters*, *org:founded by*, *per:children*, *per:city of death*, *per:date of death*, *per:origin*, *per:religion*, *per:spouse* for TACRED, and similarly *headquarters location*, *founded by*, *child*, *place of death*, *date of death*, *country of origin*, *religion*, *spouse* for DocRED.

[4] The **positive to negative** ratio in training data has an effect on the resulting model's quality. We experimented with positive-to-negative ratios of 1, 5, 10 and 20, as well as with a "match the dev-set" ratio. We found a ratio of 10 negative examples for each positive sentence to performs well.

fication. As our main goal in this paper is to evaluate different bootstrapping methods, and not novel methods for document-level relation extraction, we chose to include only instances with single supporting sentence in DocRED (*i.e.* sentence level relations). As DocRED's labelled test set is not publicly available, we used the development set as our test set and used 20% of the train set as development set.

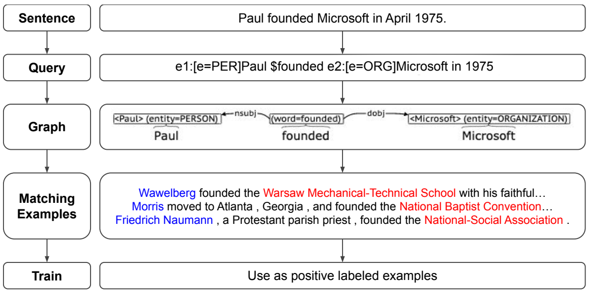
Models Our classifiers throughout the following experiments are based on the Entity Markers architecture (Soares et al., 2019). In the paper, the authors proposed wrapping the relation arguments with marker tokens (*e.g.* $[E1_{start}]$ John $[E1_{end}]$ was born in $[E2_{start}]$ 1948 $[E2_{end}]$). The altered text is then passed as input to a BERT model (Devlin et al., 2018) where the relation between the two entities is represented by the concatenation of the final hidden states corresponding to their respective start tokens. Finally, this representation is fed into a classification head and the model is fine-tuned for relation classification. *cf.* (Soares et al., 2019) for more details. We use a similar model with the exception that we use a more recent pretrained language model, RoBERTa (Liu et al., 2019), and perform binary, rather than multi-class, classification.

In all of the following experiments we trained our model with 3 different random seeds to lower variance introduced to the model with different initializations, and report the average score. At inference time we set the prediction threshold value for the test set to be the cut-off value that maximized F1 over the development set.

5 Manual Annotation Baseline

Setup. Our comparison point throughout the paper is a model trained on traditionally-collected annotated data. We sample increasing-sized annotated sets from TACRED and DocRED, containing 55, 110, 220, 550, and 1100 examples. These correspond to 5, 10, 20, 50, 100 positive examples with 50, 100, 200, 500, 1,000 negatives examples. Additionally, we measure the performance on these datasets when using all available positive examples for each relation.

Results Listed in the top rows of Table 1, averaged over all relations. Unsurprisingly, increasing the number of examples increases performance, with the exception of DocRED on which using all positive labels performs slightly worse than using 100 sampled positive examples for each relation, we attribute this to sampling noise. DocRED scores are generally lower than TACRED scores. This is because of the way we constructed the development and test sets: while in TACRED's development set each sentence includes a single entity pair with a single relation, in DocRED, we pass all possible sentences with entity pairs of the same type as the evaluated relation as possible candidates. This dramatically increases the number of candidates, and by that of possible type I errors. Moreover, as we included only examples with exactly one supporting sentence, the number of positive examples is low for some of the relations. All of this effects DocRED classification scores comparing to TACRED.



Paul founded Microsoft in April 1975.

e1:[e=PER]Paul $founded e2:[e=ORG]Microsoft in 1975

Wawelberg founded the Warsaw Mechanical-Technical School with his faithful…
Morris moved to Atlanta , Georgia , and founded the National Baptist Convention…
Friedrich Naumann , a Protestant parish priest , founded the National-Social Association .

Use as positive labeled examples

Figure 1: Flow of the Syntactic Search by Example method. For details, see §6.

Importantly, in all these experiments, the number of annotated examples used is significantly higher than the number used in our Syntactic Search experiments (3 examples in total).

6 Syntactic Search by Example

We consider this section to be the main contribution of the work. We show that:
(i) with modern DL modeling, effective relation extractors can be trained using sentences derived from less than a handful of syntactic patterns; and
(ii) through the use of by-example syntactic search engines, one can construct these patterns very quickly, without needing to understand syntax.

To explain the suggested workflow, let's consider a user who wants to train a relation extraction binary classifier for the *founded by* relation, and has a single example sentence, "Paul founded Microsoft in April 1975". Patterns over syntactic structures and entity types are very effective for deriving high-precision extraction templates. For example, searching for sentences containing the word "founded" with an *nsubj* dependency of type PERSON and *dobj* dependency of type ORG, will return many matches for the *founded-by* relations.

Method	**TACRED**	**DocRED**
Annotated 5+50	0.097	0.140
Annotated 10+100	0.136	0.215
Annotated 20+200	0.266	0.271
Annotated 50+500	0.458	0.311
Annotated 100+1000	0.516	0.321
Annotated All	0.569	0.306
Pattern Based RE (3 qrs)	0.128	-
Synt. Search (3 queries)	0.443	0.266
Example Generation	0.439	0.109
Search + Generation	0.491	0.277

Table 1: Average test F1 score over all relations. Pattern Based RE was given 3 positive patterns. Synt. Search is trained on data created from same 3 patterns. The Annotated experiments are denoted by the number of positive examples + negative examples.

There are two issues with this approach (1) while high-precision, the recall of the patterns is low; and (2) syntactic patterns require both linguistic and computational expertise to specify and execute.

The premise of this paper is that the low recall can be offset by machine learning. The sentences resulting from syntactic search over a few patterns are diverse enough that an ML model trained over them manages to generalize from the specific syntactic pattern and identify a broader range of cases, increasing recall substantially. We show this is indeed the case.

To overcome the need for linguistic expertise we propose using a by-example syntactic search engine (Shlain et al., 2020)[5] which allows users to execute syntactic queries based on example sentences: the user enters a sentence satisfying the relation of interest and annotates it with light markup indicating the arguments and the trigger words. The system then automatically translates the markup into a syntactic pattern, matches it against a large pre-annotated corpus (*e.g.* all Wikipedia sentences), and returns results. The user does not need to be familiar with syntactic formalisms or with advanced NLP.

6.1 By-example Patterns for Collecting Training Data

Fig. 1 demonstrates the user process. Starting with the sentence *Paul founded Microsoft in April 1975*, the user marks *Paul* as *e1* (`e1:`) with an entity-type restriction of PERSON (`[e=PER]`), *Microsoft* as *e2* (`e2:`) with an entity-type ORG (`[e=ORG]`), and *founded* as a trigger word (`$founded`). The SPIKE system translates the query into a syntactic graph, which is then matched against Wikipedia, returning 11,345 sentences matching the pattern (note that the word 'founded' is matched lexically, while Paul and Microsoft become place holders for any person and any organization that adhere to the syntactic configuration). A subset of the returned sentences is then used as positive examples for model training.

While 11,345 cases make an impressive training set, these sentences share the same core syntactic configuration, and classifiers, trained on these matches, will not necessarily generalize well. The matches will also share the exact same lexical predicate ("founded"). The lack of lexical diversity of the predicate can be expanded by the user by supplying alternative words, perhaps aided by distributional similarity methods such as word2vec, or by querying a bi-LM such as BERT (Devlin et al., 2018) (§6.2.1). To counter the lack of structural diversity the user can supply additional patterns, derived from example sentences. For example, the user may supply also '[$_{e_2}$Microsoft]'s <u>founder</u> [$_{e_1}$Paul]' (possessive construction) and '[$_{e_2}$Microsoft] was <u>founded</u> by [$_{e_1}$Paul]' (passive) as additional patterns (§6.2).

6.2 Experiments and Results

Setup For each relation, we select 3 representative sentences and annotate them based on the process described above[6]. We do not perform any lexical expansion of trigger words beyond the initial pattern at this point. The queries are processed by SPIKE (Shlain et al., 2020) and the results are used as positive instances in the generated training set. A full list of the SPIKE queries we used can be found in appendix D, Table 5.

We also compare the TACRED classifier to a rule based extractor which uses the syntactic queries directly. Each syntactic query is added as a syntactic pattern to this extractor: any sentence which satisfies one of the syntactic patterns is labeled as a positive instance; sentences which do not satisfy any of the patterns are labeled negative.

Results Listed in the *Synt. Search* and *Pattern*

---

[5] `https://spike.apps.allenai.org`

[6] In this experiment, the selection of representative sentences is based on a heuristic process: we intuitively conceive of basic sentences exemplifying the relation, construct the corresponding Spike queries and briefly validate the number and quality of the returned results. We limit the number of seed examples to 3 since we believe coming up with 3 examples should be simple even for non-experts. In §7.1 we show that using more seed examples can further improve performance.

Dataset	Predicates	100	500	1000
TACRED	One Trig.	0.487	0.459	0.461
	Trig. List	0.517	0.490	0.478
DocRED	One Trig.	0.290	0.336	0.338
	Trig. List	0.316	0.338	0.337

Table 2: F1 scores for *founded by*, *child*, *place of death* and *date of death* and *spouse* when expanding the triggers list for the Syntactic Search "by Example" method.

Based RE rows of Table 1[7], *Pattern Based RE*, using just the 3 patterns per relation, achieves a very low F-score of 12.8%, due to low recall. However, this is already competitive with training a classifier on 5-10 positive examples per relation. Training a classifier on the extracted relations increases the scores significantly, to 44.3F_1 on TACRED and 26.6F_1 on DocRED, approaching supervised training on 50+500 annotations (for TACRED) or 20+200 annotations (for DocRED). This result demonstrates that training an ML model over the output of a rule based model can significantly improve performance, echoing similar conclusions in Angeli et al. (2013). Interestingly, Angeli et al. (2013) used a total of 4,697 patterns across 41 relations, an average of 114 patterns per relation. We demonstrate that by applying syntactic patterns to a large corpus and using modern DL classifiers, results competitive with manual annotation baselines can be reached with as few as 3 syntactic rules.

6.2.1 Syntactic Search with Trigger Expansion

Setup Constructing queries from 3 seed sentences produces retrieved sentences with low lexical diversity. *e.g.* if all the seed sentences for *founded-by* use the word *"founded"* to express the relation, then all retrieved sentences will likewise include the word *"founded"*, and exclude alternatives like "established", "formed", "started", etc.

In this experiment we generalize the seed queries to allow a list of trigger words rather than a single word. We consider only relations which include a lexical trigger in their seed patterns[8]. Alternative triggers are selected by reviewing the closest words to the original triggers in word2vec's embedding space (Mikolov et al., 2013). Appendix C includes the lists of alternative lexical triggers used. We train classifiers on 100+1000, 500+5,000 and 1,000+10,000 examples obtained from these expanded-trigger queries.

Results As illustrated in Table 2, adding alternative triggers improves results across all sample sizes for TACRED and for the 100+1000 size in DocRED.

7 Augmenting Syntactic patterns with Natural Language Generation

We showed how the Syntactic Search by Example method works with only a few human annotated examples. In this section we would like to pursue NLG based methods to expand the number of exemplary patterns. Generative language models, compared to other methods for data augmentation (*e.g.* Iterative bootstrapping and distant supervision) are highly accessible and require low technical expertise (sometimes passing a prompt is enough). Moreover, recent papers (Wei and Zou, 2019; Anaby-Tavor et al., 2019; Kumar et al., 2020; Amin-Nejad et al., 2020; Russo et al., 2020) report high impact of such models for the closely related Data Augmentation task. We therefore present numerous methods that take advantage of such models for RE bootstrapping.

First we show how a user can produce a high number of generated sentences using GPT2 (Radford et al., 2019). Then we demonstrate how the generated sentences can be integrated in the Syntactic Search by Example method (§7.1). Finally, in order to validate the necessity of the syntactic search in this flow we compare it to feeding the raw generations as inputs to a classifier (§7.2).

Generating Examples LM

The user-flow Depicted in Fig. 2: The user enters a relation prompt ("Paul founded Microsoft"), to which the system responds by returning sentences that express the same relation. While not all returned sentences express the relation, many of them do. To filter out out-of-relation sentences the user goes through the list until she identifies a predefined number of positive examples (Here we used 100 sentences). In our experiments we encountered 1 positive example for every 3 examples annotated. This 1-out-of-3 ratio is significantly better than blindly sampling from a corpus (1-out-of-57, see §3), and by that can considerably save annotation time. For each example, the user marks the relevant entities, and optionally also the trig-

---

[7] Results correspond to 100+1,000 (TACRED) and 1,000+10,000 (DocRED) examples, for results and discussions of different dataset sizes, see Appendix B.

[8] *per:children*, *per:date of death*, *org:founded by*, *per:city of death* and *per:spouse*, and DocRED's *child*, *date of death*, *founded by*, *place of death* and *spouse*

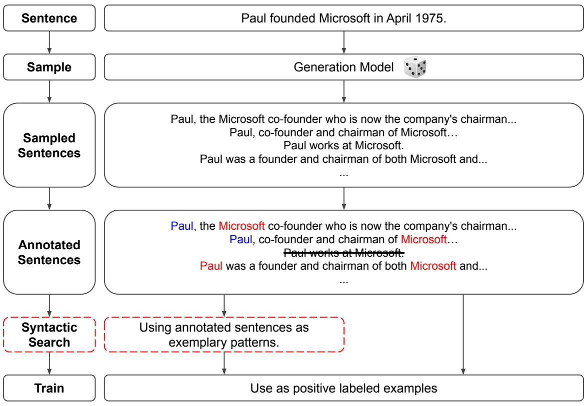


Figure 2: Flow of sampling examples from conditional language model. The "Syntactic Search" step corresponds to §7.1, while skipping this step, corresponds to §7.2.

ger word (the main word indicating the relation). These examples are then used as additional input examples to the syntactic search engine (§7.1) or as train datasets for ML models (§7.2).

Technical details We begin with a large pre-trained LM (we use GPT2-medium (Radford et al., 2019)), and fine-tune it to the generation task. The method assumes the availability of relation-annotated data, though its relations do not need to overlap with the ones we are attempting to extract (in our case, we ensured the groups are distinct). The approach can be considered as an instance of transfer-learning, where we attempt to transfer the example-generation knowledge from the training relations to novel relations. Given the annotated RE dataset, we consider positive examples of the form $(\boldsymbol{s}, e_1, e_2, r)$, where $r \in R$. We transform each instance to a conditioned LM training example, in which the LM sees a prefix (prompt) and should complete it. In our case the prompt is derived from (e_1, e_2, r), followed by a special symbol, and we train the LM to produce the corresponding sentence $\boldsymbol{s}$. To derive the prefix we apply a pre-defined *template* associated with each relation r[9]. The template has two slots to be filled with the entities e_1 and e_2. For example, a template for the *founded-by* relation can take the form $[e_2]$ *founded* $[e_1]$. We then fine-tune GPT2 on these training examples. At inference time, the user provides a *single* prompt based on their desired relation.

Given the user prompt, we generate 1000 sentences with nucleus sampling (Holtzman et al., 2019) of 0.99 and length of up to 50 tokens. We annotate the generated sentences until reaching 100 positive instances (usually requiring 200-300 sentences), this takes up to 1.5 hours per relation. These generated sentences are annotated and used as inputs to the syntactic search method (§7.1) or directly as positive examples to a classifier (§7.2).

7.1 Enhancing Syntactic Search with Example Generation

We integrate the generation outputs in the Syntactic Search by Example method by taking the positive annotated examples (on which we mark the entities as part of the annotation process) and automatically transforming them into SPIKE queries. This step has the potential to add substantial syntactic and lexical diversity to the pattern set, resulting in both larger and more diverse sets of positive examples. This combines the best of both worlds: the generative model is used to provide structural and lexical diversity, while the syntactic search system is used to provide a large selection of naturally occurring corpus sentences adhering to these patterns.

Experiments and Results

Setup To reduce noise, we exclude queries where more than 1 out of 5 sampled results does not express the relation of interest. On average, we increased the number of syntactic patterns to 9.25, ranging from 6 to 14 after filtering.

Results As listed in the *Search + Generation* row of Table 1, this method achieved best performance for both TACRED and DocRED with overall scores corresponding to 550/1100 and 220/550 annotated examples respectively. Using the generation outputs as examples doesn't only help in suggesting more sentences satisfying the relation but also in augmenting the number of predicates used. We looked on the number of predicates used for the TACRED relations which include lexical triggers (similarly to §6.2.1), the generation phase suggested 7.4 predicates on average, more than the 2.8 predicates per relation of our original patterns, and less than the trigger expansion method we suggested in §6.2.1, where we tried to find all the possible predicates, with 18.2 triggers on average. We conclude that while the Syntactic Search by Example method performs well with only a few example patterns, this can be even improved with more input examples. While we report Syntactic Search by Example enjoys such generation-based pattern aug-

---

[9] In our experiments, we use on average 3 different templates for each relation type, so a single annotated relation example will result in 3 (on average) different fine-tuning examples for the LM, each with a different prompt.

mentation, a similar boost with different, non-NLG, methods is of course possible. We leave further probing for other pattern augmentation methods as future work.

7.2 Directly Training Classifiers using Generation Outputs

It is possible that generative models produce diverse enough training examples that will suggest our syntactic search superfluous. We validate the necessity of taking the annotated generations (Annotated similarly to §7.1) through the Syntactic Search by Example method, by comparing it to simply passing the annotated generations as classifier inputs, as depicted in the RHS of Fig. 2.

Experiments and Results

Setup Many of the samples include the entities from the prompt verbatim. Before using them as the model inputs, we replace the entities with a random Wikipedia entity of the same type.

Results As can be seen in the *Example Generation* row in Table 1, on TACRED, this method produces F1 scores on par with Syntactic Search by Example. However, evaluating on DocRED, the method does not produce competitive results[10]. On both datasets it produce worse than *Search + Generation*. We conclude that it is more beneficial to use outputs of generative models as syntactic search queries, and by that find syntactically similar sentences, comparing to simply use generations as the train set. We deduce models are likely to generalize better on “real world" examples.

8 Additional Experiments

8.1 Results across relations

Analyzing the results we highlight some interesting trends (Fig. 3). First, we note that the behavior is not consistent between relations, nor datasets: different relations behave differently, showing different trade-offs between different methods.

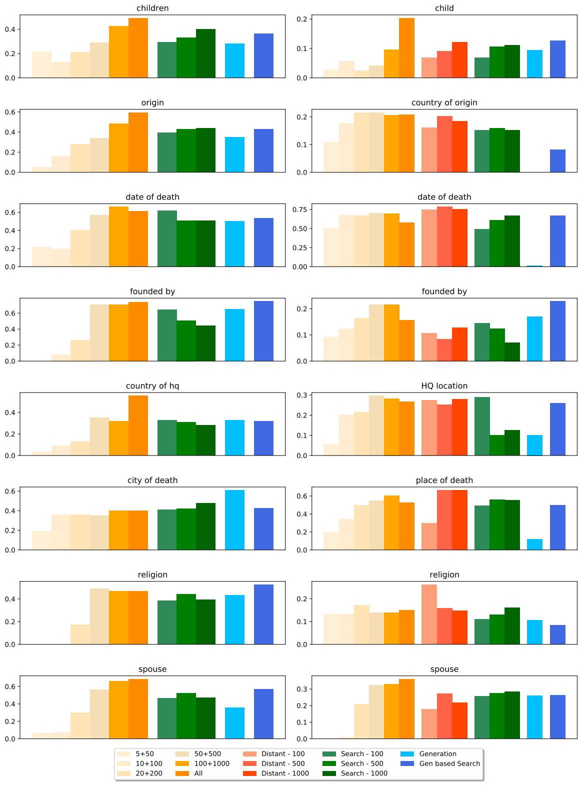


Figure 3: F1 scores of TACRED (right), and DocRED (left) by relation.

Classifiers for relations like *“Religion"*[11], *“City of Death"* and *“Date of Death"* seem to plateau at around 50-100 manually annotated examples. For these relations, annotating more data is not necessarily useful. The syntactic search approach works especially well for these relations: applying syntactic search over 3 seed queries is sufficient to yield results on par or slightly higher than all available manually annotated data. We hypothesize that these findings might be the result of low diversity in the ways these relations are typically expressed.

While the combined *Search + Generation* approach is overall useful, the effect is not consistent across relations: performance improves for some relations and deteriorates for others. In §7.1 we described the techniques we use to reduce the noise coming from additional queries. These techniques however are rather basic and these results indicate that more advanced techniques of the type used in Angeli et al. (2015) and Ratner et al. (2017), are likely to yield more consistent improvements.

8.2 Distant Supervision

Setup Distant supervision (Mintz et al., 2009) suggests a method to construct a training dataset based on a large external KB of relation triplets. Yao et al. (2019) offered a machine annotated version of DocRED constructed by aligning Wikipedia pages

---

[10] The language model used to generate examples was finetuned on a version of TACRED which excludes the relations we evaluate on. Still, for TACRED, the language model is finetuned and evaluated on data from the same domain (newswire). The DocRED data on the other hand, is taken from Wikipedia, so the evaluation is essentially out of domain. We therefore conclude that used independently, this approach is applicable only in cases where a background RE dataset is of the same domain as the target corpus from which we want to extract relations.

[11] TACRED’s “religion" relation plateaus as it has a low number of train instances.

with Wikidata. The authors took great care in creating this resource: a high-quality NER model trained on in-domain manually annotated data was used to automatically annotate possible relation arguments; a named entity linker was used to merge entities with similar KB ID; and finally, Wikidata was queried in order to label pairs of linked entities.

We trained a classifier using the released data, sampling increasing number of examples: (100+1,000, 500+5,000, 1,000+10,000). We report best score of $0.312 F_1$ (500+5,000 split).

Results This Distant Supervision dataset, created by Yao et al. (2019), appears to be of very high quality and the results are on par with the full set of manually annotated data. These results indicate that given a large KB of relation triplets, a high-quality in-domain NER, and a high quality linking solution, distant-supervision is a very promising technique. It should be noted however, that the availability of all these external resources is very rare in practice and is not required by the methods proposed in this work.

9 Applicability to other languages

We explored only English in this work. However, we argue that our main method – example-based syntactic search followed by DL-training – is not strongly tied to English, and we encourage other researchers to experiment with it in their languages of interest. We provide details of what is needed to adapt the system to a different language.

The Syntactic Search by Example method requires (1) An automatically dependency-parsed corpora in the language. These can be readily produced by the many syntactic parsers that are available for many languages (Manning et al., 2014; Honnibal and Montani, 2017; Qi et al., 2020). (2) An indexing engine that supports efficient queries over parse trees. Shlain et al. (2020) uses the open-source Odinson engine (Valenzuela-Escárcega et al., 2015) for this purpose. (3) A component that translates a query in spike's "by example" syntax to the indexing engine's query syntax. This requires finding the minimal (in terms of number of nodes) sub-graph that connects all relation arguments (and predicates if available), then search for sentences with similar sub-graphs in the index. With these three components, a syntactic-search system can be readily implemented. The rest of the components are straightforward application of DL methods. Indeed, we suspect the major obstacle in application to a new language will be the availability of evaluation data.

10 Conclusion

We show that with modern DL classifiers and a dataset bootstrapped using syntactic search with as few as 3 seed patterns can be as effective as a dataset with hundreds of manually annotated samples. Using LMs help to further diversify the dataset and improve results. Overall, our results are positively optimistic for bootstrapping methods. However, this work is only an initial step in exploring methods for bootstrapping relation extractors using minimal user effort, supported by strong pre-trained neural LMs. We hope to encourage further work in this direction.

11 Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT).

References

Ali Amin-Nejad, Julia Ive, and Sumithra Velupillai. 2020. Exploring transformer text generation for medical dataset augmentation. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4699–4708.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2019. Not enough data? deep learning to the rescue! *arXiv preprint arXiv:1911.03118*.

Gabor Angeli, Arun Tejasvi Chaganty, Angel X Chang, Kevin Reschke, Julie Tibshirani, Jean Wu, Osbert Bastani, Keith Siilats, and Christopher D Manning. 2013. Stanford's 2013 kbp system. In *TAC*.

Gabor Angeli, Victor Zhong, Danqi Chen, Arun Tejasvi Chaganty, Jason Bolton, Melvin Jose Johnson Premkumar, Panupong Pasupat, Sonal Gupta, and Christopher D Manning. 2015. Bootstrapped self training for knowledge base population. In *TAC*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, page 1. Lisbon.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M Strassel. 2015. Overview of linguistic resources for the tac kbp 2015 evaluations: Methodologies and results. In *TAC*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Shikhar Murty, Pang Wei Koh, and Percy Liang. 2020. Expbert: Representation engineering with natural language explanations. *arXiv preprint arXiv:2005.01932*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

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

Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, 11 3:269–282.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, pages 3567–3575.

Giuseppe Russo, Nora Hollenstein, Claudiu Musat, and Ce Zhang. 2020. Control, generate, augment: A scalable framework for multi-attribute text generation. *arXiv preprint arXiv:2004.14983*.

Micah Shlain, Hillel Taub-Tabib, Shoval Sadde, and Yoav Goldberg. 2020. Syntactic search by example. In *Proceedings of ACL 2020, System Demonstrations*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*.

Marco A Valenzuela-Escárcega, Gus Hahn-Powell, Mihai Surdeanu, and Thomas Hicks. 2015. A domain-independent rule-based framework for event extraction. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 127–132.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. *arXiv preprint arXiv:1906.06127*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.

A Estimated effort for Annotation

Table 3 lists for each relation the ratio of positive to negative examples in the TACRED training set. A negative example for a relation *r*, is any example whose entities share types with a positive example of *r*, but whose label is different from *r*. Note that TACRED significantly under represents negative examples so the reported ratio is an upper bound on the ratio in the wild.

Relation	Pos/Neg Ratio
org:country_of_hq	1/7
org:founded_by	1/56
per:children	1/64
per:city_of_death	1/31
per:date_of_death	1/26
per:origin	1/10
per:religion	1/2
per:spouse	1/52

Table 3: Pos/Neg ratio in TACRED, rounded to the closest fraction.

B Syntactic Search by Example with varying dataset sizes

We experimented with varying the number of sampled examples, using the same 3 seed syntactic patterns. The results are reported in Table 4. While DocRED's F1 scores increase with increasing number of sampled examples, the trend is opposite in TACRED. We believe this is due to different initializations and inductive noise in both the positive and negative samples introduced by sampling from semi-noisy data.

Method	**TACRED**	**DocRED**
Synt. Search - 100	0.443	0.250
Synt. Search - 500	0.434	0.259
Synt. Search - 1000	0.427	0.266

Table 4: Syntactic Search by Example with different training sizes

C Trigger List Expansion

For the majority of patterns used in the Syntactic Search by Example experiments we used a single trigger word (see Appendix D). To experiment with using trigger lists, we modified the patterns in Appendix D in the following way:

We changed the triggers in all *child\children* patterns to include any of the following possibilities:

baby, child, children, daughter, daughters, son, sons, step-daughter, step-son, step-child, step-children, stepchildren, stepdaughter, stepson

For *founded-by* relations we change the "founder" trigger to be any of these triggers:

founder, co-founder, cofounder, creator

and changed "founded" to be any trigger from the following list:

create, creates, created, creating, creation, co-founded, co-found, debut, emerge, emerges, emerged, emerging, establish, established, establishing, establishes, establishment, forge, forges, forged, forging, forms, formed, forming, founds, found, founded, founding, launched, launches, launching, opened, opens, opening, shapes, shaped, shaping, start, started, starting, starts

In *spouse* relations we expanded the "husband\wife" trigger to be any of:

ex-husband, ex-wife, husband, widow, widower, wife, sweetheart, bride

and the "marry" trigger to:

divorce, divorced, married, marry, wed, divorcing

For the *"date of death"* and *"place\city of death"* we changed the "died" trigger to any of:

died, executed, killed, dies, perished, succumbed, passed, murdered, suicide

D Examples used for Syntactic Search by Example

child
<>e1:[e=PER]John 's t:[w={triggers}]daughter , <>e2:[e=PER]Tim, likes swimming.

<>e1:[e=PER]Mary did something to her t:[w={triggers}]son, <>e2:[e=PER]John in 1992.

<>e1:[e=PER]Mary was survived by her 4 t:[w={triggers}]sons, John, John, <>e2:[e=PER]John and John.
triggers = son | daughter | child | children | daughters | sons

founded by
<>e1:[e=ORG]Microsoft t:[w]founder <>e2:[e=PER]Mary likes running.

<>e2:[e=PER]Mary t:[w]founded <>e1:[e=ORG]Microsoft.

<>e1:[e=ORG]Microsoft was t:[w]founded $by <>e2:[e=PER]Mary.

headquarters location
John Doe, a professor at the <>e1:[e=ORG]Oxford <>in:[t=IN]in <>e2:[e=LOC]England likes running.

<>e1:[e=ORG]Oxford, a leading <>t:[t=NN]company <>in:[t=IN]in <>e2:[e=LOC]England.

<>e2:[e=LOC]England pos:[t=POS]'s largest university is <>e1:[e=ORG]Oxford.

religion
<>e1:[e=PER]John is a e2:[w={triggers}]Jewish,,

e2:[w={triggers}]Jewish <>e1:[e=PER]John is walking down the street.

<>e1:[e=PER]John is a e2:[w={triggers}]Methodist Person.

triggers = Methodist | Episcopal | separatist | Jew | Christian | Sunni | evangelical | atheism | Islamic | secular | fundamentalist | Christianist | Jewish | Anglican | Catholic | orthodox | Scientology | Islamist | Islam | Muslim | Shia

spouse
<>e1:[e=PER]John 's t:[w=wife | husband]wife, <>e2:[e=PER]Mary , died in 1991.
<>e1:[e=PER]John t:[l]married <>e2:[e=PER]Mary,,
<>e1:[e=PER]John is t:[w]married to <>e2:[e=PER]Mary,

origin
<>e2:[e=MISC]Scottish <>e1:[e=PER]Mary is high.
<>e1:[e=PER]Mary is a <>e2:[e=MISC]Scottish professor.
<>e1:[e=PER]Mary, the <>e2:[e=LOC]US professor.

date of death
<>e1:[e=PER]John was announced t:[w]dead in <>e2:[e=DATE]1943.
<>e1:[e=PER]John t:[w]died in <>e2:[e=DATE]1943.
<>e1:[e=PER]John, an NLP scientist, t:[w]died <>e2:[e=DATE]1943.

place of death
<>e1:[e=PER]John t:[w]died in <>e2:[e=LOC]London, <>country:e=LOC England in 1997.
<>e1:[e=PER]John t:[w]died in <>e2:[e=LOC]London in 1997.
<>e1:[e=PER]John $-LRB- t:[w]died in <>e2:[e=LOC]London $-RRB-.

DocRED's *founded by*
<>e1:[e=ORG]MISC Microsoft t:[w]founder <>e2:[e=PER]Mary likes running.
<>e2:[e=PER]Mary t:[w]founded <>e1:[e=ORG]MISC Microsoft.
<>e1:[e=ORG]MISC Microsoft was t:[w]founded $by <>e2:[e=PER]Mary.

DocRED's *origin*
<>e2:[e=MISC]Scottish company, <>e1:[e=ORG]Microsoft is successful.
<>e1:[e=ORG]MISC Microsoft is a <>e2:[e=MISC]Scottish Company.

Continued on next page

<>e1:[e=ORG]MISC Microsoft is a <>t:[t=NN]song $by <>e2:[e=MISC]Scottish musician.
DocRED's ***date of death*** <>e1:[e=PER]John $-LRB- <>e1:[e=PER]John t:[w]died in <>e2:[e=DATE]1943. <>e1:[e=PER]John, an NLP scientist, t:[w]died <>e2:[e=DATE]1943.
DocRED's ***place of death*** <>e1:[e=PER]John t:[w]died in <>e2:[e=LOC]London, <>country:e=LOC England in 1997. <>e1:[e=PER]John t:[w]died in <>e2:[e=LOC]London in 1997. <>e1:[e=PER]John $-LRB- $[e=DATE]1997, $[e=LOC]London $- $[e=DATE]1997 <>e2:[e=LOC]London $-RRB-.
DocRED's ***headquarters location*** <>e1:[e=ORG]Microsoft, a leading <>t:[t=NN] company <>in:[t=IN]in <>e2:[e=LOC]Redmond. <>e1:[e=ORG]Microsoft is t:[l=base | headquarter]based in <>e2:[e=LOC]England. <>e1:[e=ORG]Microsoft, a leading <>t:[t=NN] company based <>in:[t=IN]in <>e2:[e=LOC]Redmond.

Table 5: SPIKE search patterns for TACRED and DocRED relations.

Towards a Decomposable Metric for Explainable Evaluation of Text Generation from AMR

Juri Opitz
Dept. of Computational Linguistics
Heidelberg University
69120 Heidelberg
`opitz@cl.uni-heidelberg.de`

Anette Frank
Dept. of Computational Linguistics
Heidelberg University
69120 Heidelberg
`frank@cl.uni-heidelberg.de`

Abstract

Systems that generate natural language text from abstract meaning representations such as AMR are typically evaluated using automatic surface matching metrics that compare the generated texts to reference texts from which the input meaning representations were constructed. We show that besides well-known issues from which such metrics suffer, an additional problem arises when applying these metrics for AMR-to-text evaluation, since an abstract meaning representation allows for numerous surface realizations. In this work we aim to alleviate these issues by proposing $\mathcal{MF}_\beta$, a decomposable metric that builds on two pillars. The first is the **principle of meaning preservation** $\mathcal{M}$: it measures to what extent a given AMR can be reconstructed from the generated sentence using SOTA AMR parsers and applying (fine-grained) AMR evaluation metrics to measure the distance between the original and the reconstructed AMR. The second pillar builds on a **principle of (grammatical) form** $\mathcal{F}$ that measures the linguistic quality of the generated text, which we implement using SOTA language models. In two extensive pilot studies we show that fulfillment of both principles offers benefits for AMR-to-text evaluation, including explainability of scores. Since $\mathcal{MF}_\beta$ does not necessarily rely on gold AMRs, it may extend to other text generation tasks.

1 Introduction

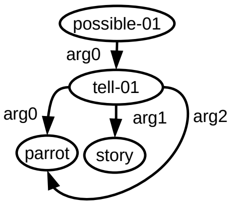


Figure 1: *"Perhaps, the parrot is telling itself a story"*.

Abstract Meaning Representation (AMR, Banarescu et al. (2013)) aims at capturing the meaning of a sentence in a machine-readable graph format. AMR captures, i.a., word senses, semantic roles and coreference. The AMR in Fig. 1 represents the sentence *Perhaps, the parrot is telling itself a story*. In this graph, tell-01 links to a PropBank (Palmer et al., 2005) frame, and arg$_n$ labels indicate participant roles: *parrot* is both *speaker* (arg0) and *hearer* (arg2), *story* is the *utterance* (arg1).

The task of AMR-to-text generation has recently garnered much attention (Song et al., 2017, 2018; Konstas et al., 2017; Cai and Lam, 2020b; Ribeiro et al., 2019). The output of AMR-to-text systems is typically evaluated against the sentence from which the AMR was created, using standard surface string matching metrics such as BLEU (Papineni et al., 2002) or CHRF(++) (Stanojević et al., 2015; Popović, 2015, 2016; Popov, 2017), as is standard in many NLG tasks. These metrics suffer from several issues, for example, they penalize paraphrases, are highly sensitive to outliers (Mathur et al., 2020), and lack interpretability (Sai et al., 2020).

Some of these issues get compounded when evaluating AMR-to-text. The core of the problem is that there are many ways to realize a sentence from a meaning representation. Fig. 2 shows four candidate sentences (**i-iv**) for a given AMR (left). One system generates (**i**): *Maybe the cat is playing.* while another generates (**iii**): *Perhaps, the cat plays the flute*. Clearly, (**i**) captures the meaning of the gold graph better than (**iii**), which contains 'hallucinated' content – a well-known issue in neural generation (Logan et al., 2019; Wang and Sennrich, 2020). Yet, when using a canonical metric such as BLEU to evaluate sentences (**i**) and (**iii**) against the reference, the system that produces hallucinations (**iii**) is greatly rewarded (54 BLEU points) to the disadvantage of systems that yield meaning preserving

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1504–1518
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

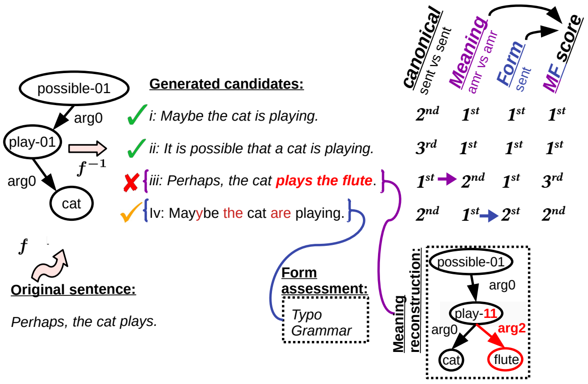


Figure 2: The ***Canonical*** evaluation matches n-grams from the sentences and assigns inappropriate ranks. Our metric $\mathcal{MF}_\beta$ fuses ***Meaning*** and ***Form*** assessment and better reflects the ranking of the generations.

sentences (**i**) (18 points) and (**ii**) (5 points).

This work aims at a (better) metric that **measures meaning preservation** of the generated output towards the MR given as input, by (re-)constructing an AMR from the generated sentence and comparing it to the input AMR. In Fig. 2, Reconstruction is the result of parsing (**iii**). The reconstructed AMR exposes several meaning deviations (marked in red): it contains an alternate sense of *play* and contains an additional semantic role arg2 with filler *flute*. By contrast, when converting sentences (**i**), (**ii**), or (**iv**) to AMRs, we obtain flawless reconstructions. We will measure preservation of ***Meaning*** using well-defined graph matching metrics.

Figure 2 also illustrates that assessing meaning preservation is not sufficient to rate the quality of generations: (**iv**) captures the meaning of the AMR well – but its form is flawed: it suffers from wrong verb inflection, a common issue in low-resource text generation settings (Koponen et al., 2019).

In order to rate both meaning and form of a generated sentence, we combine the score for meaning reconstruction with a score called ***Form*** that **judges the sentence's grammaticality and fluency**. By these moves, we obtain a more suitable and explainable ranking with a combined ***MF*** score.[1] By clearly distinguishing between ***Meaning*** and ***Form***, our ***MF*** score (henceforth denoted by $\mathcal{MF}_\beta$) also aligns well with recent calls to achieve a clearer separation of these aspects in NLU (Bender and Koller, 2020).

Generally, our contributions are as follows:

(1) We propose two linguistically motivated principles that aim at a sound evaluation of AMR-to-text systems: the **principle of meaning preservation** and the **principle of (grammatical) form**.

(2) From these principles we derive and implement a (novel) $\mathcal{MF}_\beta$ **score for AMR-to-text generation**[2] which is composed of individual metrics for meaning and form aspects. $\mathcal{MF}_\beta$ allows users to modulate these two views on generation quality to vary their impact on the final metric score.

(3) We conduct two major pilot studies involving (English) text generations from a range of competitive AMR-to-text systems and human annotations. First we study the potential practical benefits of $\mathcal{MF}_\beta$ when evaluating systems, such as its prospects to offer interpretability of scores and finer-grained system analyses. The second study probes potential weak spots of $\mathcal{MF}_\beta$, e.g., its dependence on a strong AMR parser.

We consider $\mathcal{MF}_\beta$ as it stands as a suitable metric to enhance interpretability of generation scores.

2 Fusing meaning and form into $\mathcal{MF}_\beta$

While current NLG metrics lack *interpretability* and mainly focus on the form of generated text (Sai et al., 2020), in this work we emphasize the *meaning aspect* in NLG evaluation, which is most clearly dissociated from form when generating text from structured inputs such as AMR. At the same time, form and wording of the generated text cannot be ignored, as we want such systems to produce *natural and well-formed sentences*. Equipped with this two-fold objective, we start building our $\mathcal{MF}_\beta$ score which aims at a *balanced combination* of both quality aspects: **meaning and form**.

2.1 From principles to $\mathcal{MF}_\beta$

In a first step we introduce our

Principle of meaning $\mathcal{M}$. *Generated sentences should allow loss-less AMR reconstruction.*

This principle expresses a key expectation for a system that generates NL sentences from abstract meaning representations. Namely, the generated sentence should reflect the meaning of the AMR. So, in order to assess whether a generated sentence $s' = f^{-1}(m)$ is a valid generation for the input AMR m, rather than matching s' against a

---

[1] See Fig. 2: $1^{st}/2^{nd}$ rank: **i**; 3^{rd} rank: **iv**; 4^{th} rank: **iii**.

[2] We make code available at `https://github.com/Heidelberg-NLP/MFscore`.

reference sentence s, we perform this assessment **in the abstract MR domain**, by applying an inverse system f that *parses* the generated text back to an AMR $m' = f(s') = f(f^{-1}(m))$. I.e., we desire a $metric : \mathcal{D} \times \mathcal{D} \rightarrow [0, 1]$ that satisfies: $s \equiv s' \iff m = m' \iff metric(s, s') = 1$. Two texts are equivalent iff their meaning abstractions denote the same meaning. In case $f(s')$ yields an AMR $m' \neq m$, we can still determine the degree to which s' preserves the meaning of AMR m by measuring the distance between m and m' by standard AMR metrics, e.g., $Smatch(m, m')$.

Note that computing $Smatch(m, m')$ does not depend on a reference sentence, because the comparison is conducted purely in the abstract domain. This is mathematically more appealing for the evaluation of AMR-to-text, since it solves the problem that one abstract representation may result in various (valid) surface realizations (cf. Appendix A.1). Finally, we also do not necessarily need to rely on a gold graph m, but can instead set $m = f(s)$, i.e., the parse of the reference sentence. This means that future application of $\mathcal{M}$ to other kinds of text generation tasks is straightforward.

However, the principle $\mathcal{M}$ alone is not sufficient: we also expect the system to generate grammatically well-formed and fluent text. For example, s': *Possibly, it(self) tells parrot a story.* contains relevant content expressed in the AMR of Fig. 1, but it is neither grammatically well-formed, nor a natural and fluent sentence. This leads us to our

Principle of form $\mathcal{F}$. *Generated sentences should be syntactically well-formed, natural and fluent.*

In the style of the well-established F_β score (van Rijsbergen, 1979), we fuse these two principles into the $\mathcal{MF}_\beta$ score:

$$\mathcal{MF}_\beta = (1+\beta^2)\frac{Meaning \times Form}{(\beta^2 \times Meaning) + Form} \quad (1)$$

Here, $Form$ and $Meaning$ are expressed as ratios that will be more closely described in the following subsection. β allows users to gauge the evaluation towards $Form$ or $Meaning$, depending on specific application scenarios. Users may prefer the harmonic mean (β = 1) or may give $Meaning$ double weight compared to $Form$ (e.g., β = .5).[3] In our experiments we consider extreme decompositions into $Meaning\text{-}only$ ($\beta \rightarrow 0$) or $Form\text{-}only$ ($\beta \rightarrow \infty$).

2.2 Parameterizing meaning

We measure $\mathcal{M}$ or $Meaning$ (Meaning Preservation) with a score range in $[0, 1]$ by reconstructing the AMR with a SOTA parser and computing the relative graph overlap of the reconstruction and the source AMR using graph matching. We call this RESMATCH. Given a generated sentence s' and source AMR m, we match $parse(s')$ against m by computing $amrMetric(parse(s'), m)$. This means that we have to decide upon $parse$ and $amrMetric$. We propose two potential settings.

AMR reconstruction To reconstruct the AMR with $parse$, we use the latest state-of-the-art AMR parser by Cai and Lam (2020a). With 80.3 Smatch F1, this parser is almost on-par with human agreement (estimated at 0.71–0.83 Smatch F1 in Banarescu et al. (2013)). We henceforth call it GSII.

Assessing $\mathcal{M}$ with AMR metrics To obtain a score for $\mathcal{M}$ we propose to use S^{2}match (Opitz et al., 2020) – a variant of Smatch (Cai and Knight, 2013) that performs a *graded match for concept nodes*. This offers the potential to compensate for noise in automatically generated text or minor lexical deviations from the original sentence.

Discussion Comparing to references by matching their meaning graphs has the prospect of offering interpretability and explanations, by detecting redundant or missing meaning components in the generations. In our studies, we will see that this assessment can be conducted by computing a *single graph overlap score* (e.g., S^{2}match F1), or along *multiple dimensions of meaning*, such as SRL, coreference or WSD (Damonte et al., 2017). Generally, $\mathcal{MF}_\beta$ gives researchers the flexibility of choosing a $parse$r or $amrMetric$ to their liking. In this work, we choose the best current $parse$r that achieves high IAA with humans. Yet, we would also like to know whether the $parser$ is vulnerable to specific peculiarities of generated sentences, or how using another parser affects the scores. We will investigate these issues more closely in §4.1.

2.3 Parameterizing form with LMs

Assessing sentence grammaticality and fluency is not an easy task (Heilman et al., 2014; Katinskaia and Ivanova, 2019). Recently, Lau et al. (2020); Zhu and Bhat (2020) show that probability estimates based on language models can be used as an indicator for measuring complex notions of form and for measuring acceptability in context. For

---

[3] Generally, $Form$ receives β times as much importance compared with $Meaning$.

our $\mathcal{MF}_\beta$ score we desire an interpretable ratio as input, which we base on LM predictions as follows.

Binary form assessment Given a specific candidate generation s', we use a binary variable to assess whether s' is of satisfactory form. For this, we first calculate the mean token probability:[4]

$$mtp(\cdot) = \frac{1}{n}\sum_{j=1}^{n} P(tok_j|ctx_j), \qquad (2)$$

where ctx_j is different for uni-directional LMs ($ctx_j = tok_{1...j-1}$) and bi-directional LMs ($ctx_j = tok_{1...j-1,j+1...n}$). We compute mtp for the generated sentence s' and the reference s and calculate a preference score $prefScore = \frac{mtp(s')}{mtp(s')+mtp(s)}$. The decision of whether the $Form$ of a generated sentence s' is acceptable is then calculated as

$$accept = \begin{cases} 1, & \text{if } prefScore \geq 0.5 - tol \\ 0, & \text{otherwise,} \end{cases}$$

where tol is a tolerance parameter. Less formally, a sentence is considered to have an acceptable surface form in relation to its reference if its form is estimated to be at least as good as the reference minus a tolerance, which we fix at 0.05. I.e., the corpus-level $Form$ score reflects the ratio of generated sentences that are of acceptable form.[5]

Predictor selection We consider GPT-2 (Radford et al., 2019), distil GPT-2 (Sanh et al., 2019), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) as a basis for assessing $Form$. We conduct experiments on WebNLG (Gardent et al., 2017; Shimorina et al., 2017), which contains human fluency and grammaticality judgements for machine-generated sentences. We find that GPT-2 performs best: it discriminates sentences of poor and perfect fluency and grammaticality with an F1 score of approximately 0.8, and shows marginally better performance compared to the other LMs (see Appendix A.2 for the experiment details). We thus select GPT-2 as our LM for $Form$ assessment.

Discussion While the reconstruction of meaning does not depend on the reference sentence, we do make use of it, in $prefScore$, for better assessment of $Form$. One reason is that when assessing the form of a sentence s' that contains rare words, the 'raw' $mtp(s')$ may be too pessimistic and may not well relate to the quality of the form. Generally, the mtp (or any LM probability) itself is not well interpretable and hardly allows comparison to the mtp of other sentences (e.g., if they are about a different topic). However, by relating the mtp of the generated sentence to the mtp of a (same-topic) reference, we gain three advantages: first, we do not, a-priori, penalize generations that contain rare words. Second, we obtain an interpretable corpus-level ratio (rate of sentences that are of acceptable form). This is important, since sound $\mathcal{MF}_\beta$ calculation ideally requires two interpretable ratios as input. Third, by avoiding any string matching, we still keep form and meaning aspects clearly distinct.

2.4 Goals of our pilot studies

Our main aim is to establish, with the proposed $\mathcal{MF}_\beta$ score for AMR-to-text generation, **i)** a **balanced** and **interpretable assessment** of generated text according to ***Meaning*** **and** ***Form***. Yet, as detailed in §2.2 and §2.3, both components depend on a number of **ii) hyperparameters**, such as the parser applied for $Meaning$ reconstruction, or the LM used for $Form$ assessment. These parameters may also be subject to change over time. It is thus important to assess the effects of such factors on metric scores and system rankings. We investigate both aspects of $\mathcal{MF}_\beta$ in **two pilot studies**.

In **the first study**, in §3, we aim to assess the prospects of $\mathcal{MF}_\beta$ when ranking SOTA systems. We will see that $\mathcal{MF}_\beta$ can explain system performance differences by disentangling $Form$ and $Meaning$, an asset that no other metric can offer.

The second study, in §4, investigates the impact of $\mathcal{MF}_\beta$'s dependence on a parser and a LM. We i) investigate the effects of using different parsers, ii) assess the potential suitability of $\mathcal{MF}_\beta$ for other text generation tasks, by ablating the human gold graph from the evaluation and using $\mathcal{MF}_\beta$ to evaluate generated text vs. reference text, and iii) validate the LM's binary predictions for $Form$ in a manual annotation study.

3 Study I: Assessing interpretability

Setup: data & metrics for system ranking We obtain test predictions of several state-of-the-art *AMR-to-text generation systems* on LDC2017T10, the main benchmark for this task: (i) densely con-

---

[4] We use the mean (instead of the product) because Bryant and Briscoe (2018) find that basing decisions on the mean works well in practice when assessing possible corrections of grammatical errors.

[5] I.e., the $Form$ score for a single sentence with $accept \geq 0.5 - tol$ equals 1.0. If a precise assessment for a single sentence is needed, we can fall back on $prefScore$ (+/- tol).

						Meaning			*Form*	$\mathcal{MF}_1$	$\mathcal{MF}_{0.5}$
	abbrev.	BLEU	METEOR	chrF++	BERTsc.	RESMATCH			-	-	-
					F1	P	R	F1	%acc.	Eq. 1	Eq. 1
$apprUB$	-	-	-	-	-	83.1	80.1	81.5	100	89.8	84.6
Ribeiro et al. (2019)	R'19	$27.9_{(5)}$	$33.2_{(7)}$	$58.7_{(6)}$	$92.7_{(4)}$	76.5	67.7	$71.9_{(6)}$	$51.6_{(5)}$	$60.1_{(5)}$	$66.6_{(5)}$
Guo et al. (2019)	G'19	$27.6_{(6)}$	$33.7_{(6)}$	$57.3_{(7)}$	$92.4_{(7)}$	78.2	70.0	$73.9_{(3)}$	$47.1_{(7)}$	$57.5_{(7)}$	$66.3_{(6)}$
Wang et al. (2020a)	Wb'20	$27.3_{(7)}$	$34.1_{(5)}$	$59.3_{(5)}$	$92.6_{(6)}$	79.6	65.0	$71.5_{(7)}$	$49.5_{(6)}$	$58.5_{(6)}$	$65.7_{(7)}$
Cai and Lam (2020b)	C'20	$29.8_{(4)}$	$35.1_{(4)}$	$59.4_{(4)}$	$92.7_{(4)}$	78.1	69.2	$73.4_{(5)}$	$51.9_{(4)}$	$60.3_{(4)}$	$67.0_{(4)}$
Mager et al. (2020)-M	Mb'20	$33.0_{(2)}$	$37.3_{(2)}$	$63.1_{(3)}$	$93.9_{(2)}$	79.4	68.7	$73.7_{(4)}$	**74.0**$_{(1)}$	**73.9**$_{(1)}$	**73.8**$_{(1)}$
Mager et al. (2020)-L	M'20	$33.0_{(2)}$	**37.7**$_{(1)}$	$63.9_{(2)}$	**94.0**$_{(1)}$	**80.8**	69.2	$74.5_{(2)}$	$69.8_{(2)}$	$72.1_{(2)}$	$73.5_{(2)}$
Wang et al. (2020b)	W'20	**33.9**$_{(1)}$	$37.1_{(3)}$	**65.8**$_{(1)}$	$93.7_{(3)}$	80.3	**70.9**	**75.3**$_{(1)}$	$55.7_{(3)}$	$64.0_{(3)}$	$70.3_{(3)}$

Table 1: Main metric results.

nected graph convolutional networks (Guo et al., 2019); (ii) Ribeiro et al. (2019)'s system that uses a dual graph representation; two concurrently published models (iii) based on graph transformers (Cai and Lam, 2020b; Wang et al., 2020a) and (iv) a model based on graph transformers that uses reconstruction information (Wang et al., 2020b) in a multi-task loss; finally, we obtain predictions of two system variants of Mager et al. (2020) that finetune LMs and encode linearized graphs using (v) a large and (vi) a medium-sized LM. We true-case all sentences and parse them with GSII.

To put the results of $\mathcal{MF}_\beta$ into perspective, we display the scores of several metrics that have been previously used for AMR-to-text: BLEU, METEOR, CHRF++. We also calculate BERTscore (Zhang et al., 2020) with RoBERTa-large (Liu et al., 2019).[6] Results are displayed in Table 1, col. 3-6. $\mathcal{MF}_\beta$ scores (col. 7-12) are divided into $Meaning$ (RESMATCH using GSII) and $Form$ scores (based on GPT-2), and composite $\mathcal{MF}_\beta$ scores with $\beta = 1$ (harmonic) and $\beta = 0.5$ (double weight on $\mathcal{M}$).

As an upper-bound approximation for RESMATCH we propose parsing a gold sentence s and comparing the result against the gold AMR m: $apprUB$ = metric(parse(s),m).[7]

3.1 Interpretability of system rankings

Surface matching metrics lack differentiation and interpretability Table 1 shows that the baseline metrics tend to agree with each other on the ranking of systems, but there are also differences, for example, BERTscore and METEOR select M'20 as the best performing system while BLEU and CHRF++ select W'20. While certain differences may be due to individual metric properties, e.g., METEOR allowing inexact word matching of synonyms, the underlying factors are difficult to assess, since the score differences between systems with switched ranks are small, and none of these metrics can provide us with a meaningful interpretation of their score that would extend beyond shallow surface statistics. Hence, these metrics cannot give us much intuition about why and when one system may be preferable over another.

***Meaning* vs. *Form*: How $\mathcal{MF}_\beta$ explains system performance** We have seen that current metrics cannot provide us with convincing explanations as to why, e.g., W'20 should be preferred over M'20 (BLEU), or M'20 over W'20 (BERTscore). $\mathcal{MF}_\beta$ score, however, tells a story about how these systems differ, highlighting their complementary strengths by disentangling $Meaning$ and $Form$ (Bender and Koller, 2020): W'20 displays the highest RESMATCH score, i.e., AMRs constructed from its generations recover a maximum of the meaning contained in the input AMR. M'20, by contrast, outperforms all systems in $Form$ score. Looking at $\mathcal{MF}_1$, the harmonic mean of both, both systems still occupy leading ranks, but W'20 falls back to 3rd rank, due to its weaker $Form$ score.

Hence, given our metric principles, a user who cares about faithfulness to meaning, but less about fluency, should select W'20 (with higher RESMATCH compared to M'20 by Δ=1 point) – a user who desires a system that preserves meaning well but also produces sentences of decent form, should select M(b)'20 (with $\mathcal{MF}_{0.5}$ and $\mathcal{MF}_1$

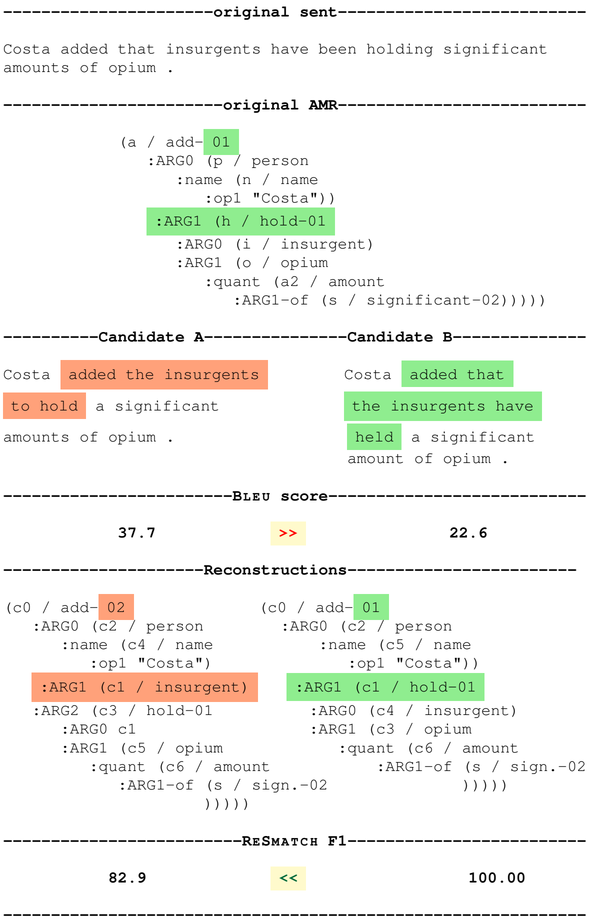
---

[6] BERTscore computes an F1-score over a *cosim*-based alignment of the contextual embeddings of paired sentences.

[7] This is the score of canonical parser evaluation. I.e., we would not expect the reconstruction m' of s' to score higher than had we applied $parse$ to the original sentence: $metric(m', m) \leq metric(parse(s), m)$ = $apprUB$. This is an idealization, as we can imagine cases where the original sentence s is more complex and thus more difficult to parse to an AMR than a simpler generated paraphrase s'. Since we are interested in a very rough upper bound estimation, we abstract from such cases in our present work.

score differences against W'20 of Δ=3.5 points and Δ=8 points). Overall, $\mathcal{MF}_\beta$ mostly agrees with BERTscore in the rankings of the teams. However, $\mathcal{MF}_\beta$'s larger score differences between the systems, due to $Form$, are striking, prompting us to investigate the $Form$ predictions in closer detail (§4.2). We will see that using a different $Form$ predictor as well as a manual native speaker annotation clearly support our assessment of $Form$.

3.2 On the quest for deeper explanation and interpretation



```
----------------------original sent--------------------------

Costa added that insurgents have been holding significant
amounts of opium .

-----------------------original AMR--------------------------

            (a / add- 01
               :ARG0 (p / person
                  :name (n / name
                     :op1 "Costa"))
               :ARG1 (h / hold-01
                  :ARG0 (i / insurgent)
                  :ARG1 (o / opium
                     :quant (a2 / amount
                        :ARG1-of (s / significant-02)))))

----------Candidate A---------------Candidate B--------------

Costa added the insurgents           Costa added that
to hold a significant                the insurgents have
amounts of opium .                   held a significant
                                     amount of opium .

------------------------Bleu score--------------------------

            37.7             >>              22.6

---------------------Reconstructions------------------------

(c0 / add- 02                   (c0 / add- 01
   :ARG0 (c2 / person             :ARG0 (c2 / person
      :name (c4 / name               :name (c5 / name
         :opl "Costa")                  :op1 "Costa"))
   :ARG1 (c1 / insurgent)         :ARG1 (c1 / hold-01
   :ARG2 (c3 / hold-01               :ARG0 (c4 / insurgent)
      :ARG0 c1                       :ARG1 (c3 / opium
      :ARG1 (c5 / opium                 :quant (c6 / amount
         :quant (c6 / amount               :ARG1-of (s / sign.-02
            :ARG1-of (s / sign.-02                )))))
                  )))))

-------------------------ReSmatch F1-------------------------

            82.9             <<              100.00

-------------------------------------------------------------
```

Figure 3: Explainable $Meaning$ score (re-)ranking.

ReSmatch can also provide us with **explanations for single-sentence (re-)rankings**. An example is shown in Figure 3. Here, the gold reference (both sentence and AMR) indicates that a person named Costa *adds* (as a communicative act[8] that some insurgents have been holding large amounts of opium. However, system generation A (which is higher ranked by Bleu) chooses a different sense of *add*, add-02, which represents the action as an operation[9], which results in an incoherent or nonsensical meaning representation where the person Costa *adds* (in the operational sense) the insurgent (as thing being added) to a circumstance to the effect that the insurgents hold a significant amount of opium. By contrast, system generation B preserves more of the gold AMR's meaning and clearly expresses that Costa performs an *act of communication* when he *adds* something. ReSmatch ($\mathcal{MF}_{\beta\to 0}$) is able to detect the meaning differences and assigns candidate B a significantly higher score than A, in fact, an S^2match score of 1.00.

ReSmatch, when parameterized with fine-grained AMR evaluation metrics of Damonte et al. (2017), can also facilitate deeper insight into **how well system generations reflect or violate specific meaning aspects**. E.g., we can investigate a system's capacity to properly reflect negation (NEG); to generate correct surface forms for NEs (NER); assess how well a system captures coreference between entities (Coref); and whether or not the predicate-argument structures (SRL) of generated sentences appropriately reflect the source meaning. We apply these fine-grained AMR metrics to the ReSmatch scores of systems displayed in Table 1 (see Appendix A.3), and observe, e.g., that R'19, which ranks last in the overall ranking, improves upon the best overall system by 3.4 points in NER recall and 1.9 points in F1. The analysis also corroborates that W'20 excels among competitors with best scores for coreference, SRL and negation, i.e., the more global aspects of sentence meaning. Such information can be valuable for researchers for deeper system analysis and for practitioners aiming for specific use cases.

4 Study II: Assess vulnerability of $\mathcal{MF}_\beta$

$\mathcal{MF}_\beta$ has two apparent vulnerabilities: first, it depends on a parser for reconstruction. We have used a SOTA parser that is on par with human IAA. Yet, we cannot exclude the possibility that it introduces unwanted errors in computing $\mathcal{MF}_\beta$ scores.

Second, the $Form$ component is based on a LM and we have seen that it can change system rankings, even when it is discounted.[10] On the one hand, our LM was carefully selected, and other metrics such as BERTscore also heavily depend on LMs.

---

[8] Sense add-01 w/ roles: Arg0: *Speaker*; Arg1: *Utterance*.

[9] Sense add-02 w/role set: Arg0: *adder*; Arg1: *thing being added*; Arg2: *thing being added to*; Arg3: *resulting sum*.

[10] In Table 1, both $\mathcal{MF}_\beta$ with $\beta = 0.5$ and $\beta = 1.0$ slightly disagree with the ranks assigned by $Meaning$ only.

On the other hand, we cannot exclude the possibility that the changed rankings are unjustified.

Our next studies investigate these weak spots more closely. First, in §4.1, we assess the outcome of $\mathcal{MF}_\beta$ when using another parser and assess its potential portability to other text generation tasks by ablating the human gold graph and evaluate generated text against reference *text*. In §4.2 we conduct a human annotation study to assess whether the provided $Form$ rankings are justified.

4.1 The parser: Achilles' heel of $\mathcal{MF}_\beta$?

Using another parser In this experiment we assess the robustness of RESMATCH against using different parsers. This is important, since the metric and rankings could change with the parser. Here, we would hope that the difference of using one competitive parser over another will not be too extreme, especially with regard to system rankings. To investigate this issue, we apply two alternative parsers: i) GPLA (Lyu and Titov, 2018), a neural graph-prediction system that jointly predicts latent alignments, concepts and relations, and ii) TTSA (Groschwitz et al., 2018), a neural transition-based parser that converts dependency trees to AMR graphs using a typed semantic algebra. We select GPLA and TTSA since they constitute technically quite distinct approaches compared to GSII.

The results are shown in Table 2 (columns labelled GPLA, TTSA and GSII). All variants tend to agree in the majority of their rankings[11] (e.g., RESMATCH ^{GPLA} vs. RESMATCH ^{GSII} F1: Spearman's ρ = 0.95, Pearson's ρ = 0.96, p<0.001). When considering $\mathcal{MF}_{\beta=0.5}$, the agreement further increases (e.g., $\mathcal{MF}_{0.5}^{GPLA}$ vs. $\mathcal{MF}_{0.5}^{GSII}$: Spearman's $\rho = 0.95$, Pearson's $\rho = 0.99$, p<0.001).

However, while using TTSA or GPLA instead of GSII has little effect on the ranks, the absolute scores can differ (e.g., W'20 70.5 F1 w/ TTSA, 73.1 F1 w/ GPLA and 75.3 F1 w/ GSII). Yet, we find that none of the generation systems are unfairly treated by our main parser GSII since we observe (mostly uniform) increments from TTSA to GPLA and from GPLA to GSII. An unfair treatment could arise, e.g., if GSII generates bad AMR reconstructions for specific NLG systems but not so for others. However, we do not observe such tendencies.

Hence we assume that GSII's score increments stem from the fact that GSII yields better reconstructions for all systems. In future work, we plan to explore parse quality control (Opitz and Frank, 2019; Opitz, 2020) or ensemble parsing (van Noord and Bos, 2017), to gain more detailed information on the quality of the meaning reconstructions.

Ablating the gold graph? Yes, we can. In lack of a gold standard for the automatic reconstructions, we elicit some indirect answers and insight about the parser's quality, by considering the following question: *What is the effect on system rankings when we replace the input gold graphs with automatic parses of the distant source sentence?* If this effect is large, this will give us reasons to worry, as it would indicate that the parser is less reliable than expected given its high IAA with humans. On the other hand, if we only see a minor effect, this may increase the trust in our parser and indicate that $\mathcal{MF}_\beta$ **could be confidently applied for explainable evaluation in other generation tasks** (such as MT or summarization), where we do not have gold AMRs, and would have to parse both generated *and* reference sentences.

The results of this experiment are displayed in Table 2: our standard setup is displayed in columns labeled GSII and the results of the setup where we replace the gold input graph with an automatic parse is indicated by GSII$^\blacklozenge$. When considering RESMATCH scores, we see only one switched rank between Mb'20 and G'20 (3–4). However, note that the absolute F1 score Δ between these two systems is overall very small (GSII: 0.2; GSII$^\blacklozenge$: 0.5). Overall, the scores do not tend to differ much when the gold graph is ablated, we observe rather small (mostly positive) changes in system scores (GSII $\rightarrow$ GSII$^\blacklozenge$): 0.1 / 1.2 / 0.4 (min/max/avg). In sum, we conclude from this experiment that ablating the gold graph does not have a major effects on the scores and rankings. And when considering the $\mathcal{MF}_{\beta=0.5}$ score, the ranking stays fully stable (the same holds true for $\mathcal{MF}_{\beta=1}$.

Discussion We have shown that metric rankings are fairly robust to using different parsers and that we do not necessarily depend on gold AMR graphs to compute the measure. This offers prospects for **using** $\mathcal{MF}_\beta$ **for an explainable assessment of systems that perform other kinds of text generation**. In order to measure $\mathcal{M}$, a parser could be applied to both the generated and the reference text, to measure their agreement in the domain of ab-

---

[11] We observe one switch of ranks for TTSA-GPLA and GPLA-GSII and 2 rank switches for TTSA–GSII in RESMATCH, and no rank switch for TTSA-GSII and one switch for TTSA-GPLA and GPLA-GSII, for $\mathcal{MF}_{0.5}$.

	ReSmatch F1				ranks ReSmatch				ranks $\mathcal{MF}_{0.5}$			
	TTSA	GPLA	GSII	GSII$^{\blacklozenge}$	TTSA	GPLA	GSII	GSII$^{\blacklozenge}$	TTSA	GPLA	GSII	GSII$^{\blacklozenge}$
$apprUB$	73.7	76.2	81.5	86.4	0	0	0	0	0	0	0	0
R'19	66.9	70.1	71.9	72.3	7	7	6	6	5	5	5	5
G'19	69.7	72.2	73.9	73.7	3	3	3	4	6	6	6	6
Wb'20	67.3	70.2	71.5	71.6	6	6	7	7	7	7	7	7
C'20	69.1	70.4	72.2	73.4	4	5	5	5	4	4	4	4
Mb'20	68.9	70.5	73.7	74.2	5	4	4	3	1	2	1	1
M'20	69.8	72.5	74.5	75.1	2	2	2	2	2	1	2	2
W'20	70.5	73.1	75.3	75.4	1	1	1	1	3	3	3	3

Table 2: Analysis of our metric using different parsers (GPLA, TTSA GSII) or ablating the gold parse by comparing the parsed generation against the parse (distant) source sentence (GSII$^{\blacklozenge}$).

stract meaning representation. This would in turn offer means for conducting fine-grained meaning analysis of generation tasks where the reference is a natural language sentence (e.g., in MT).

Note, however, that AMR, as of now, does not capture some facets of meaning that may be of interest in some generation tasks. For instance, it does not capture tense or aspect. However, what we have investigated as a *potential weakness* of $\mathcal{MF}_\beta$, namely the necessity to select a meaning parser, can also be viewed as a *potential strength*. E.g., Donatelli et al. (2018) show how tense and aspect can be captured with AMR. This indicates that $\mathcal{MF}_\beta$ can indeed be used for a tense and aspect analysis of generated text – if we parameterize it with a dedicated parser. Finally, if output and reference do not consist of single sentences, it may be apt to use a parser that constructs MRs for discourse (e.g., *DRS* (Kamp, 1981)).

In summary, we conclude that $\mathcal{MF}_\beta$, our proposed metric that aims to assess text generation quality by decomposing it into *form* and *meaning* aspects, is broadly applicable. However, different parser parametrizations may have to be considered in light of the specific nature of a generation task.

4.2 The $Form$ component of $\mathcal{MF}_\beta$

In §3.1, we have seen that the $Form$ aspect of $\mathcal{MF}_\beta$ can change system ranks. Notably, it has promoted M'20 as the best generation system, outranking W'20 (in agreement with BERTscore), whereas W'20 is selected by Bleu or ReSmatch. Now, we aim to investigate whether these impactful decisions of the $Form$ component were justified.

Human annotation We ask a native speaker of English to rate 50 paired generations of M'20 and W'20, considering only grammaticality and fluency.[12] We give more detail and provide examples in Appendix A.6. The annotator agreed in 42 of 50 pairs with the preference predicted by GPT-2 (a significant result: binomial test $p<0.000001$). We find that the M'20 and Mb'20 generations are considerably better on the surface level, compared to generations of all other systems. For instance, the best system according to $Meaning$, W'20, frequently produces inflection mishaps: *Their hopes for entering the heat is already in-sight*, while we find few such violations with M'20 (here: *Their hopes for entering the heat are already in sight*). We also find errors with adverbials, e.g., W'20 writes *They are the most indoor training at home*, while M'20 writes *They are most trained indoors at home*. Arguably both sentences are not perfect but the second is substantially more well-formed.

	R'19	G'20	Wb'20	C'20	Mb'20	M'20	W'20
GPT-2	$51.6_{(4)}$	$47.1_{(6)}$	$49.5_{(5)}$	$51.9_{(4)}$	$74.0_{(1)}$	$69.8_{(2)}$	$55.7_{(3)}$
BERT	$43.4_{(6)}$	$40.6_{(7)}$	$50.4_{(4)}$	$44.7_{(5)}$	$71.4_{(1)}$	$71.0_{(2)}$	$55.9_{(3)}$

Table 3: $Form$ scores when using a different LM.

Using a different LM The human study indicates that GPT-2 is accurate to 84% when favoring one sentence over the other, with respect to fluency and grammaticality. However, when considering that there is a trend to building systems based on fine-tuned LMs, we need to assess whether they may be favored (too) much if $Form$ is parameterized with a same or a highly similar LM to the one used by the NLG model. We find such a case in M'20: while it was not fine-tuned with the same GPT-2 that we used for $Form$ assessment, they fine-tuned their model with its siblings GPT-2-medium and GPT-large, which may share structural similarities. Therefore, we also use BERT for

---

[12] The annotator was explicitly instructed not to consider whether a sentence 'makes sense', by presenting the *Green ideas sleep furiously* example as free from structural error.

$Form$ assessment. The results in Table 3 support the conclusion from the human annotation: by large margins, both M'20 and Mb'20 deliver generations that are of significantly improved form and both agree on the group of the three best systems. Note that this insight can be provided by $\mathcal{MF}_\infty$, but it cannot be carved out by conventional metrics, since these do not disentangle $Form$ and $Meaning$.

5 Related work

Traditionally, the performance of NLG systems has been evaluated with word n-gram matching metrics such as the popular BLEU metric in MT (Papineni et al., 2002) or Rouge (Lin, 2004) in document summarization. Yet, such metrics suffer from several well-known issues (Novikova et al., 2017; Nema and Khapra, 2018; Sai et al., 2020). E.g., due to their symbolic matching strategy they cannot account for paraphrases. Recently, unsupervised (Zhang et al., 2020) or learned metrics (Sellam et al., 2020; Zhou and Xu, 2020) based on contextual language models have been proposed. For example, BERTscore (Zhang et al., 2020) uses BERT (Devlin et al., 2019) to encode candidate and reference and computes a score based on a cross-sentence word-similarity alignment. Compared with BLEU, it is computationally more expensive but tends to show higher agreement with humans. However, *all* of the aforementioned metrics return scores that are hardly interpretable and we cannot tell what exactly they have measured.

These problems carry over to the evaluation of AMR-to-text generation: May and Priyadarshi (2017) find that BLEU does not well correspond to human ratings of generations from AMR, and Manning et al. (2020) show through human analysis that none of the existing automatic metrics can provide nuanced views on generation quality. Our proposal $\mathcal{MF}_\beta$ takes a first step to address these issues by aiming at a clear separation of form and meaning, as called for by Bender and Koller (2020).

First attempts of assessing semantic generation quality have been examined in MT using semantic role labeling (Lo, 2017) or WSD and NLI (Carpuat, 2013; Poliak et al., 2018), in-between lies SPICE that evaluates caption generation via inferred semantic propositions (Anderson et al., 2016). Just like $\mathcal{MF}_\beta$, SPICE relies on automatic parses (a dependency parse of the caption and a scene graph predicted for the image) to evaluate content overlap of image and caption. Thus, SPICE is a direct precursor of an NLG metric in V&L that relies on automatically produced structured representations. Our work extends this previous work by showing ways of probing potentially harmful effects of incorporating automatic parsing components.

6 Conclusion

We propose $\mathcal{MF}_\beta$ score, a new metric for evaluation of text generation from (abstract) meaning representation. The metric is built on two pillars: $Form$ measures grammaticality and fluency of the produced sentences and $Meaning$ assesses to what extent the meaning of the input AMR is reflected in the produced sentence. We show that $\mathcal{MF}_\beta$ has the potential to yield fine-grained performance assessment that go beyond what conventional metrics can provide. Using its β-parameter, $\mathcal{MF}_\beta$ can be decomposed into complementary views – $Meaning$ and $Form$ – paving the way for custom gauging and selection of NLG systems. We have seen that $\mathcal{MF}_\beta$ corresponds well to BERTscore when rankings systems, but overcomes its opaqueness by disentangling $Meaning$- and $Form$-related quality aspects. In sharp contrast to BERTscore, the $Form$ component of $\mathcal{MF}_\beta$ dispenses with string matching against reference sentences, offering an assessment independent of lexical alignment.

An important hyperparameter of our metric is the required AMR parsing component for meaning reconstruction. We investigate the impact of its choice by choosing alternative high-performing parsers. Our study shows that absolute metric scores tend to increment when using a better parser, while system rankings are quite stable. Furthermore, we outline the potential of $\mathcal{MF}_\beta$ to extend to further text generation tasks, by ablating the human gold graph from the evaluation, such that the metric score can be computed from candidate and reference text alone. Since benchmarking of systems needs deeper exploration, we recommend $\mathcal{MF}_\beta$ score to obtain better diagnostics and explainability of text generation systems, including, but not limited to (A)MR-to-text.

Acknowledgments

We are grateful to three anonymous reviewers for their valuable comments that have helped to improve this paper. This work has been supported by the *Deutsche Forschungsgemeinschaft (DFG)* through the project *ExpLAIN*, FR 1707/4-1 as part of the RATIO Priority Program (SPP-1999).

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *Computer Vision – ECCV 2016*, pages 382–398, Cham. Springer International Publishing.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Christopher Bryant and Ted Briscoe. 2018. Language model based grammatical error correction without annotated training data. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 247–253, New Orleans, Louisiana. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020a. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020b. Graph transformer for graph-to-sequence learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7464–7471. AAAI Press.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Marine Carpuat. 2013. A semantic evaluation of machine translation lexical choice. In *Proceedings of the Seventh Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 1–10, Atlanta, Georgia. Association for Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.

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

Lucia Donatelli, Michael Regan, William Croft, and Nathan Schneider. 2018. Annotation of tense and aspect semantics for sentential AMR. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 96–108, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jonas Groschwitz, Matthias Lindemann, Meaghan Fowlie, Mark Johnson, and Alexander Koller. 2018. AMR dependency parsing with a typed semantic algebra. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1831–1841, Melbourne, Australia. Association for Computational Linguistics.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting grammaticality on an ordinal scale. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 174–180, Baltimore, Maryland. Association for Computational Linguistics.

Hans Kamp. 1981. A theory of truth and semantic representation. In J. Groenendijk, Th. Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language*, pages 277–322. Mathematisch Centrum. Reprinted in ”Meaning and the Dynamics of Interpretation - Selected Papers of Hans Kamp” (Kamp (2013)) p. 329 -369.

Anisia Katinskaia and Sardana Ivanova. 2019. Multiple admissibility: Judging grammaticality using unlabeled data in language learning. In *Proceedings of* *the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 12–22, Florence, Italy. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Maarit Koponen, Leena Salmi, and Markku Nikulin. 2019. A product and process analysis of post-editor corrections on neural, statistical and rule-based machine translation output. *Machine Translation*, 33(1-2):61–90.

Jey Han Lau, Carlos S Armendariz, Shalom Lappin, Matthew Purver, and Chang Shu. 2020. How furiously can colourless green ideas sleep? sentence acceptability in context. *arXiv preprint arXiv:2004.00881*.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Chi-kiu Lo. 2017. MEANT 2.0: Accurate semantic MT evaluation for any output language. In *Proceedings of the Second Conference on Machine Translation*, pages 589–597, Copenhagen, Denmark. Association for Computational Linguistics.

Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971, Florence, Italy. Association for Computational Linguistics.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proc. of ACL*, pages 1846–1852, Online. Association for Computational Linguistics.

Emma Manning, Shira Wein, and Nathan Schneider. 2020. A human evaluation of AMR-to-English generation systems. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4773–4786, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Jonathan May and Jay Priyadarshi. 2017. SemEval-2017 task 9: Abstract meaning representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 536–545, Vancouver, Canada. Association for Computational Linguistics.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

Rik van Noord and Johan Bos. 2017. The meaning factory at SemEval-2017 task 9: Producing AMRs with neural semantic parsing. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 929–933, Vancouver, Canada. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Juri Opitz. 2020. AMR quality rating with a lightweight CNN. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 235–247, Suzhou, China. Association for Computational Linguistics.

Juri Opitz and Anette Frank. 2019. Automatic accuracy prediction for AMR parsing. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 212–223, Minneapolis, Minnesota. Association for Computational Linguistics.

Juri Opitz, Anette Frank, and Letitia Parcalabescu. 2020. Amr similarity metrics from principles. *Transactions of the Association for Computational Linguistics*, 8(0):522–538.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

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

Adam Poliak, Yonatan Belinkov, James Glass, and Benjamin Van Durme. 2018. On the evaluation of semantic phenomena in neural machine translation using natural language inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 513–523, New Orleans, Louisiana. Association for Computational Linguistics.

Alexander Popov. 2017. Word sense disambiguation with recurrent neural networks. In *Proceedings of the Student Research Workshop Associated with RANLP 2017*, pages 25–34, Varna. INCOMA Ltd.

Maja Popović. 2015. chrF: character n-gram f-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Maja Popović. 2016. chrF deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504, Berlin, Germany. Association for Computational Linguistics.

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

Leonardo F. R. Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing AMR-to-text generation with dual graph representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3183–3194, Hong Kong, China. Association for Computational Linguistics.

C. J. van Rijsbergen. 1979. *Information Retrieval*. Butterworth.

Ananya B Sai, Akash Kumar Mohankumar, and Mitesh M Khapra. 2020. A survey of evaluation metrics used for nlg systems. *arXiv preprint arXiv:2008.12009*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*.

Anastaisa Shimorina, Claire Gardent, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: report on human evaluation. Technical report, Technical report, Université de Lorraine, Nancy, France.

Linfeng Song, Xiaochang Peng, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2017. AMR-to-text generation with synchronous node replacement grammar. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 7–13, Vancouver, Canada. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Miloš Stanojević, Amir Kamran, Philipp Koehn, and Ondřej Bojar. 2015. Results of the WMT15 metrics shared task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 256–273, Lisbon, Portugal. Association for Computational Linguistics.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation. *arXiv preprint arXiv:2005.03642*.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020a. Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8(0):19–33.

Tianming Wang, Xiaojun Wan, and Shaowei Yao. 2020b. Better amr-to-text generation with graph structure reconstruction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3919–3925. International Joint Conferences on Artificial Intelligence Organization. Main track.

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

Wangchunshu Zhou and Ke Xu. 2020. Learning to compare for better training and evaluation of open domain natural language generation models. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9717–9724. AAAI Press.

Wanzheng Zhu and Suma Bhat. 2020. Gruen for evaluating linguistic quality of generated text. *arXiv preprint arXiv:2010.02498*.

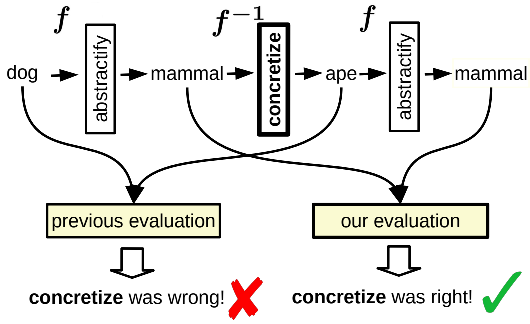


Figure 4: A critical issue and its alleviation.

A Appendices

A.1 On the soundness of comparing generated sentences in the AMR domain

First, we provide a simple example for our argument (it is safer to compare texts generated from AMR in the AMR domain) and then a simple proposition together with its proof. The example is displayed in Fig. 4, where, similar to AMR-to-text, we see a (surjective) function that generates concrete objects from abstract objects (e.g., $mammal \rightarrow \{dog, mouse, cow\}$). Now, imagine we are given $mammal$ and are tasked with generating a single concrete instance. How can we assess whether our output is correct? We cannot safely assess this by testing whether the output (e.g., cow) is the same as another instance of $mammal$ (e.g., dog). Instead, we can re-apply the abstraction f to cow and conduct the comparison safely in the abstract domain.

Proposition. *a) The canoncical AMR-to-text evaluation setup, that matches generated sentence s' to distant source sentence s, is not well defined. b) This issue can be alleviated by grounding the evaluation in the AMR domain by re-appling parse, abstaining from direct use of s (thereby using AMR-to-text generation as a right inverse function).*

Proof. Let X be a set of concrete objects (e.g., sentences) and f a (surjective) function from X to Y (e.g., 'sent-to-AMR'), where Y contains abstract objects (e.g., AMRs), s.t. $|Y| < |X|$. Then, using $f^{-1} : Y \rightarrow X$ (e.g., 'AMR-to-sent')) as right-inverse is well-defined: $f \circ f^{-1} = id_Y$ (Proposition b), but using it solely as left-inverse (as done in previous evaluation) does not guarantee a well-defined result: $f^{-1} \circ f \neq id_X$ (Proposition a). $\square$

A.2 Form predictor selection experiment

To estimate how well they are able to assess $Form$, we make use of human-assigned scores for data from the WebNLG task as provided by Gardent et al. (2017). It contains grammaticality and fluency judgments by humans for more than 2000 machine-generated sentences. We report the F1 score, both for grammaticality and fluency, by converting the human assessment scores to *accept* predictions, and using them as a gold standard to evaluate the LM-based *accept* predictions over (i) all 12k sentence pairs[13] and (ii) only the 5k sentence pairs where both grammaticality and fluency where either rated as 'perfect' (max. score) or 'poor' (min. score) by the human.[14]

	F1 score			
	grammaticality		fluency	
LM	poor/perfect	all	poor/perfect	all
GPT2	**0.80**	**0.74**	**0.80**	0.71
GPT2-distill	0.79	0.73	0.76	0.70
BERT	**0.80**	0.72	**0.80**	**0.72**
RoBERTa	0.66	0.72	0.69	**0.72**

Table 4: Results for assessing the $Form$ score prediction (corpus-level) of different LMs for NLG-generated sentences against humans judgements (separated by grammaticality and fluency); all: all 12k generated sentences vs. 'poor/perfect': the 5k instances of best/worst generations in both grammaticality and fluency.

	Reentrancies			SRL			negation			NER		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
$apprUB$	72.1	60.7	65.9	77.7	73.5	75.5	88.6	70.5	78.5	82.2	80.1	81.1
R'19	63.7	50.3	56.2	71.1	62.4	66.4	72.1	50.6	59.5	82.2	**70.7**	**76.0**
G'19	66.9	52.9	59.1	73.7	64.9	69.0	75.0	51.5	61.1	78.6	68.9	73.5
Wb'20	67.6	51.5	58.4	75.1	63.6	68.9	74.3	49.7	59.6	86.5	60.3	71.0
C'20	66.1	52.4	58.4	73.4	64.8	68.8	78.3	54.2	64.1	80.8	67.2	73.4
Mb'20	65.9	53.2	58.9	74.3	65.7	69.8	70.6	45.5	55.3	82.6	69.4	75.4
M'20	67.9	53.3	59.7	**76.4**	66.5	71.1	73.7	53.9	62.3	**82.8**	68.3	74.9
W'20	**68.8**	**55.7**	**61.6**	76.1	**68.1**	**71.9**	**79.2**	**55.1**	**65.0**	82.4	67.3	74.1

Table 5: Fine-grained results using $\mathcal{MF}_0$ parameterized with metrics proposed by Damonte et al. (2017).

The results are displayed in Table 4 and show (i) that the LMs lie very close to each other with respect to their capacity to predict fluency and grammatically, and (ii) that both fluency and grammaticality can be predicted fairly well.

A.3 RESMATCH with fine-grained meaning metrics

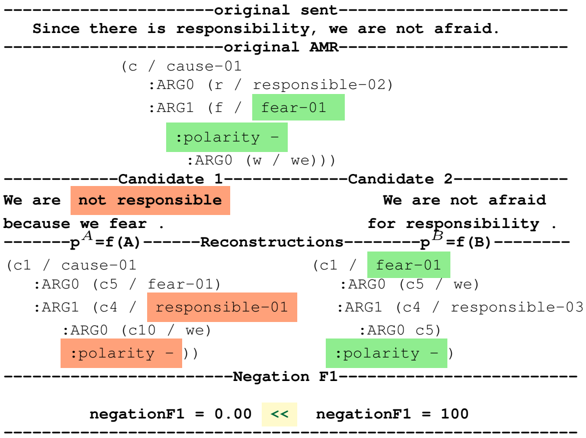
Using Damonte et al. (2017)'s metric suite for fine grained semantic system analysis, we obtain fine-grained results with respect to various meaning aspects of system performance. The results are shown in Table 5.

In sum, the system of W'20 appears to be the

---

[13] This includes all generated sentences from a given input, as provided by Gardent et al. (2017); Shimorina et al. (2017)

[14] The ratings are based on a 3-point Likert scale.



```
----------------------original sent-------------------------
   Since there is responsibility, we are not afraid.
-----------------------original AMR-------------------------
            (c / cause-01
               :ARG0 (r / responsible-02)
               :ARG1 (f / fear-01
                 :polarity -
                   :ARG0 (w / we)))
------------Candidate 1-------------Candidate 2------------
We are not responsible                  We are not afraid
because we fear .                     for responsibility .
-------p^A=f(A)------Reconstructions--------p^B=f(B)--------
(c1 / cause-01                (c1 / fear-01
   :ARG0 (c5 / fear-01)          :ARG0 (c5 / we)
   :ARG1 (c4 / responsible-01    :ARG1 (c4 / responsible-03
      :ARG0 (c10 / we)              :ARG0 c5)
      :polarity - ))             :polarity - )
------------------------Negation F1--------------------------

         negationF1 = 0.00  <<  negationF1 = 100
-------------------------------------------------------------
```

Figure 5: Explained negation confusion.

clear winner in most aspects of meaning. This is intuitive, since the system has been trained with an auxiliary signal that provides information on how well an AMR can be reconstructed from the generated sentence.

A.4 ReSmatch explains negation error

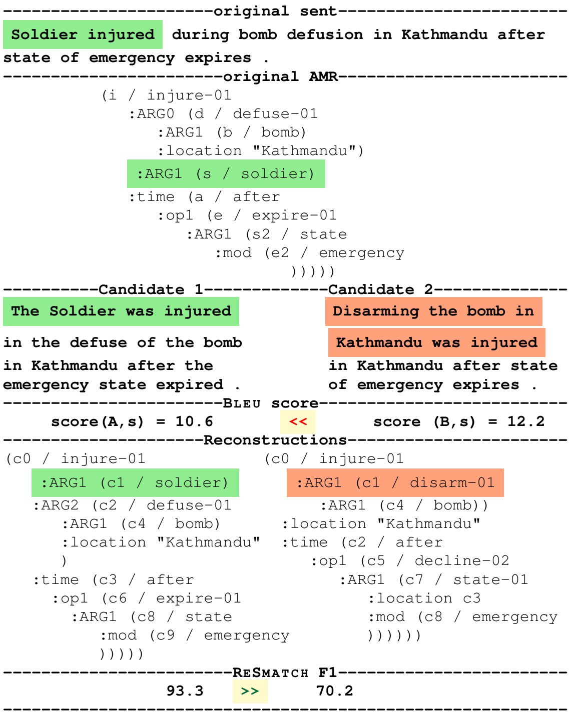
In Figure 5, both systems struggle to fully capture the meaning of the original AMR $f(s)$. However, the system based on GPT medium (Mb'20) erroneously assesses that *we are not responsible* and *we fear*. However, quite the opposite is true: the gold graph and gold sentence states that *there is responsibility* and *there is no fear*. This important facet of meaning is better captured by C'20. The reconstruction shows that it reflects the gold negated concepts much better and does not distort facts that are core to the meaning. In consequence, the negation F1 is zero for the left sentence with the distorted facts and maximum for the sentence that sticks true to the facts.

A.5 ReSmatch explains SRL error

Figure 6 shows an example, were ReSmatch ranks two generated candidate sentences differently compared to Bleu. In this case, gold sentence and gold AMR both express that there is some soldier who tried to defuse a bomb and got injured in the process. Clearly, candidate generation A captures the meaning better, in fact, it captures it almost perfectly. However, since the surface text deviates from the gold sentence, Bleu overly penalizes this generation and assigns a very low score of 10.6 points. In contrast, candidate B matches the surface slightly better (12.2 points), but distorts the meaning: it does not contain any information about the soldier and states that *Disarming was injured*, which is grammatically correct, but semantically wrong, or even non-sense.

We see that the surface matching metric cannot explain its scores (beyond superficial statistics) and delivers a ranking that does not appropriately reflect the performance of the generation systems. However, ReSmatch shows that the gold parse and the parse of candidate A agree with each other in the central *ARG1*-role of the main predicate *injure-01*: *it is the soldier who got injured*. On the other hand, in the reconstruction of the AMR of candidate B, the *ARG1* argument is filled differently: *it is the disarmament that gets injured*.

This assessment allows ReSmatch to increment the score for generation A by a large margin, from 10.6 (Bleu) to 93.3 points (ReSmatch), expressing substantial agreement in meaning with the gold. The score for the candidate generation B also gets incremented – but it gets incremented much less, only to 70.2 points, expressing good to mediocre agreement. Thus, by detecting the SRL confusion, ReSmatch re-ranks the candidate generation such that the resulting ranking is more appropriate.



```
---------------------original sent-----------------------
Soldier injured during bomb defusion in Kathmandu after
state of emergency expires .
-----------------------original AMR----------------------
         (i / injure-01
            :ARG0 (d / defuse-01
               :ARG1 (b / bomb)
               :location "Kathmandu")
            :ARG1 (s / soldier)
            :time (a / after
               :op1 (e / expire-01
                  :ARG1 (s2 / state
                     :mod (e2 / emergency
                            )))))
----------Candidate 1-------------Candidate 2--------------
The Soldier was injured          Disarming the bomb in
                                 Kathmandu was injured
in the defuse of the bomb
in Kathmandu after the           in Kathmandu after state
emergency state expired .        of emergency expires .
-----------------------Bleu score-------------------------
     score(A,s) = 10.6     <<      score (B,s) = 12.2
---------------------Reconstructions----------------------
(c0 / injure-01              (c0 / injure-01
   :ARG1 (c1 / soldier)         :ARG1 (c1 / disarm-01
   :ARG2 (c2 / defuse-01           :ARG1 (c4 / bomb))
      :ARG1 (c4 / bomb)         :location "Kathmandu"
      :location "Kathmandu"     :time (c2 / after
      )                            :op1 (c5 / decline-02
   :time (c3 / after                  :ARG1 (c7 / state-01
     :op1 (c6 / expire-01                :location c3
       :ARG1 (c8 / state                 :mod (c8 / emergency
          :mod (c9 / emergency           ))))))
          )))))
-----------------------ReSmatch F1------------------------
                    93.3   >>    70.2
----------------------------------------------------------
```

Figure 6: Explained SRL confusion.

A.6 Annotation study for form assessment

```
Sys (W'20): He also said that our athletes do n't very use of competition under strong sunlight .
Corr (human): He also said that our athletes are not very used to competition under strong sunlight .
---->  not acceptable

Sys (W'20): Sheng Chen , the 6 th position of Hubei province , who was totally scored 342.60 at 342.60 points this year ,
is a temporary position .
Corr (human): Sheng Chen , the 6 th position of Hubei province , who has totally scored 342.60 points this year ,
is in a temporary position .
---->  not acceptable

Sys (W'20): The Chinese competitors are Lan Wei and Sheng Chen , qualify semi - final .
Corr (human):  The Chinese competitor Lan Wei and Sheng Chen qualify for the semi - final .
---->  acceptable

Sys (M'20): Fengzhu Xu won many championships in international competition before .
Corr (human): Fengzhu Xu won many championships in international competitions before .
---->  acceptable
```

Figure 7: Sentences of flawed form. `--->` refers to the binary acceptability judgment (Eq. 2.3).

Annotator and annotation The English native speaker (UK) annotated 50 paired sentences of M'20 and W'20. They were presented in shuffled order and the annotator was tasked with assigning a nominal number, starting from zero, that indicates the amount of grammatical or fluency issues as assessed by the native speaker. Additionally, the human was asked to provide a correction.

Examples of sentences of flawed form. See Figure 7.

The Source-Target Domain Mismatch Problem in Machine Translation

Jiajun Shen$^{\diamond\dagger}$ **Peng-Jen Chen**$^{\diamond\dagger}$ **Matt Le**† **Junxian He**$^{\bullet *}$ **Jiatao Gu**†
Myle Ott† **Michael Auli**† **Marc'Aurelio Ranzato**†
†Facebook AI Research
$^{\bullet}$Carnergie Mellon University
{jiajunshen,pipibjc,mattle,jgu}@fb.com
{myleott,michaelauli,ranzato}@fb.com junxianh@cs.cmu.edu

Abstract

While we live in an increasingly interconnected world, different places still exhibit strikingly different cultures and many events we experience in our every day life pertain only to the specific place we live in. As a result, people often talk about different things in different parts of the world. In this work we study the effect of local context in machine translation and postulate that this causes the domains of the source and target language to greatly mismatch. We first formalize the concept of source-target domain mismatch, propose a metric to quantify it, and provide empirical evidence for its existence. We conclude with an empirical study of how source-target domain mismatch affects training of machine translation systems on low resource languages. While this may severely affect back-translation, the degradation can be alleviated by combining back-translation with self-training and by increasing the amount of target side monolingual data.

1 Introduction

The use of language greatly varies with the geographic location (Firth, 1935; Johnstone, 2010). Even within places where people speak the same language (Britain, 2013), there is a lot of lexical variability due to change of style and topic distribution, particularly when considering content posted on social media, blogs and news outlets. For instance, while a primary topic of discussion between British sport fans is cricket, American sport fans are more likely to discuss other sports such as baseball (Leech and Fallon, 1992).

The effect of local context in the use of language is even more extreme when considering regions where different languages are spoken. Despite the increasingly interconnected world we live in, people in different places tend to talk about different things. There are several reasons for this, from cultural differences due to geographic separation and history, to the local nature of many events we experience in our every day life.

This phenomenon has not only interesting sociolinguistic aspects but it has also strong implications in machine translation (MT) (Bernardini and Zanettin, 2004). In particular, machine translation aims at automatically translating content in two languages that are often spoken in very distant geographic locations by people with rather different cultures.

As of today, most MT research has been based on the often implicit assumption that content in the two languages is *comparable*. Sentences comprising the parallel dataset used for training are assumed to cover the same topic distribution, regardless of the originating language. Even when there exist a mismatch between the source and the target domain, the dataset creator is assumed to have made the effort to equalize the two distributions.

The major contribution of this work is to raise awareness in the MT community that this assumption may not hold in many real world settings of interest, which often involve distant and low-resource language pairs and for content produced every day on the Internet by means of blogs, social platforms and news outlets. The goal of an MT system is to translate source sentences sampled from the source domain to the target language. Training an MT system on a comparable corpus may lead to poor generalization unless the test domain matches the domain of the training data. Likewise, training on an uncurated corpus exhibiting mismatch between the source and the target domain may also work poorly when naïvely applying popular methods like back-translation (Sennrich et al., 2015).

---

[*] Work done during an internship at Facebook AI Research.
[◇] Equal contribution.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1519–1533
April 19 - 23, 2021.

Notice that there may very well be several factors contributing to such mismatch, such as change of register, different functional use of the text in the two languages, and difference in the quality of the data generation process of the two languages. Regardless of what is contributing to the observed mismatch, it is important to be aware of its existance and effect on MT algorithms.

The source-target domain mismatch (STDM) can be understood as an instance of *multi-domain* MT (see Fig. 1 for an illustration and §3 for a formal definition), whereby part of the parallel dataset and the source monolingual dataset are "in-domain" because they originate from the source domain, and the remaining part of the parallel dataset as well as the target monolingual data are "out-of-domain". There already exist several techniques for domain adaptation, like domain tagging (Caswell et al., 2019) and dataset weighting (Edunov et al., 2018; Wang et al., 2017; van der Wees et al., 2017), which are applicable and which we also employ in this work. However, these may not be enough to improve generalization on low resource languages because STDM effectively decreases the already scarce amount of useful (in-domain) parallel data, hindering good generalization. It is therefore important to quantify STDM (§4) and consider how STDM affects methods that leverage monolingual data (§5).

For instance, STDM may negatively impact the effectiveness of back-translation because, even if the backward model was perfect, the back-translated data is out-of-domain relative to the source domain from which we aim to translate. Empirically we found that this is the case both in a controlled setting (§6.1) as well as in realistic datasets (§6.2). However, this issue can be compensated by adding more target-side monolingual data and by combining back-translation with self-training (Yarowski, 1995).

2 Related Work

The observation that topic distributions and various kinds of lexical variabilities depend on the local context has been known and studied for a long time (Firth, 1935). For instance, Firth (1935) says "*Most of the give-and-take of conversation in our everyday life is stereotyped and very narrowly conditioned by our particular type of culture*". In her seminal work, Johnstone (2010) analyzed the role of place in language, focusing on lexical variations within the same language, a subject further explored by Britain (2013). Some of these works were the basis for later studies that introduced computational models for how language changes with geographic location (Mei et al., 2006; Eisenstein et al., 2010).

In the field of topic modeling, there has been a new sub-field emerging over the past 10 years focusing on modeling multi-lingual corpora (Mimno et al., 2009; Boyd-Graber and Blei, 2009; Gutierrez et al., 2016). However, only recently had researchers dropped assumptions on the use of parallel and comparable corpora (Hao and Paul, 2018; Yang et al., 2019). While some works do investigate issues related to STDM (Gutierrez et al., 2016), like how named entities receive a different distribution over words in different languages (Lin et al., 2018), none of these works have analyzed how the overall topic distribution of data originating in the source and target language differ.

In MT, researchers have often made an explicit assumption on the use of *comparable* corpora (Fung and Yee, 1998; Munteanu et al., 2004; Irvine and Callison-Burch, 2013), i.e. corpora in the two languages that roughly cover the same set of topics. Unfortunately, monolingual corpora are seldom comparable in practice. Leech and Fallon (1992) analyzes two comparable corpora, one in American English and the other in British English, and demonstrate differences that reflect the cultures of origin. Similarly, Bernardini and Zanettin (2004) observes that parallel datasets built for MT exhibit strong biases in the selection of the original documents, making the text collection not quite comparable.

The non-comparable nature of MT datasets is even more striking when considering low resource language pairs, for which differences in local context and cultures are often more pronounced. Recent studies (Søgaard et al., 2018; Neubig and Hu, 2018) have warned that removing the assumption on comparable corpora strongly deteriorates performance of lexicon induction techniques which are at the foundation of MT.

Back-translation (Sennrich et al., 2015) has been the workhorse of modern neural MT, enabling very effective use of target side monolingual data. Back-translation is beneficial because it helps regularizing the model and adapting to new domains (Burlot and Yvon, 2018). However, the typical setting of current MT benchmarks as popularized by re-

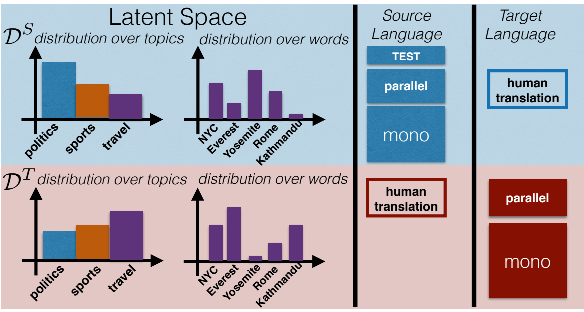


Figure 1: Toy illustration of STDM in MT. There are two domains, the source domain $\mathcal{D}^S$ (top) and the target domain $\mathcal{D}^T$ (bottom). We postulate that in a latent concept space these two domains differ because the topic distributions are different (e.g., in the source domain politics is more popular than travel) and because for the same topic the word distributions are different (e.g., the word "Everest" is more common than "Yosemite" in the travel topic of the target domain). On the right hand side, we show how STDM manifests in machine translation datasets. All data originating in the source language belongs to the source domain, this includes a portion of the parallel dataset, the source side monolingual dataset and the test set we eventually would like to translate. Empty boxes represent human translated data in the parallel training dataset.

cent WMT competitions (Bojar et al., 2019) is a mismatch between *training and test* sets, as opposed to a mismatch between *source and target* domains as in this work. Self-training (Ueffing, 2006; Yarowski, 1995; He et al., 2020) has then been employed to make better use of source side monolingual data as this is in-domain with the text we would like to translate at test time. Finally, there is a vast literature on domain adaptation which has so far mostly focused on domain shift between training and test distribution, and presence of multiple domains. Finetuning (Freitag and Al-Onaizan, 2016), domain tagging (Caswell et al., 2019) and various kinds of dataset weighting (Wang et al., 2017; van der Wees et al., 2017) are among the most popular methods to cope with domain issues, and this is also what we use in this work.

3 The STDM Problem

In this section we formalize the definition of Source-Target Domain Mismatch (STDM); this is an intrinsic property of the data which is independent of the particular MT system under consideration. In practice, there might be several factors contributing to STDM. Here, we are going to consider only the difference in topic distributions, since this is what we can easily quantify.

We assume there exists a latent concept space shared across all languages. The process to generate a sentence follows the standard data generation process used in topic modeling, whereby we first sample a distribution over topics, $\pi_i \sim \Pi$ where i is an index over topics, and then a distribution over words for each topic, $w_{ij} \sim \pi_i$, where j indexes the words in the dictionary. Next, we assume there are two distinct domains, the source domain $\mathcal{D}^S$ and the target domain $\mathcal{D}^T$. These two domains differ in both the distribution over topics Π, and the distribution over words given a certain topic π_i, as depicted in Fig. 1. For the sake of conciseness, we will refer to z^s and z^t as sentences in the concept space generated from domain $\mathcal{D}^S$ and $\mathcal{D}^T$, respectively.

Let's imagine now that we have generated two sets of sentences in each domain. What we observe in practice is their realization in each language, $\text{src}(z^s)$ and $\text{tgt}(z^t)$, where src and tgt map sentences from the concept space to the source and target language, respectively. Finally, let's denote with $h_{s\to t}$ and $h_{t\to s}$ the functions representing human translations of source sentences in the target language and vice versa.

In the simplest setting, a machine translation dataset is composed of parallel and monolingual datasets. Using the notation introduced above, the parallel dataset is denoted by $\mathcal{P} = \{(\text{src}(z^s), h_{s\to t}(\text{src}(z^s))\}_{z^s \sim \mathcal{D}^S} \cup \{(h_{t\to s}(\text{tgt}(z^t)), \text{tgt}(z^t)\}_{z^t \sim \mathcal{D}^T}$. The first set originates in the source language and belongs to the source domain, while the second set originates in the target language and belongs to the target domain. We then have a source side monolingual dataset, $\mathcal{M}^S = \{\text{src}(z^s)\}_{z^s \sim \mathcal{D}^S}$, and a target side monolingual dataset, $\mathcal{M}^T = \{\text{tgt}(z^t)\}_{z^t \sim \mathcal{D}^T}$, belonging to the source and target domains, respectively. Most importantly, the *test set* which we would like to eventually translate contains sentences in the source language, all belonging to the *source* domain. The existence of distinct source and target domains and datasets derived from these two domains as described above define the STDM problem.

While in previous MT studies, there is a mismatch between the training and the test distribution, STDM is a particular case of multi-domain training, with the test set matching one of the training domains. We would like to a) understand the effects of such mismatch and b) understand how to best leverage the out-of-domain data originating from the target language (target monolingual dataset and

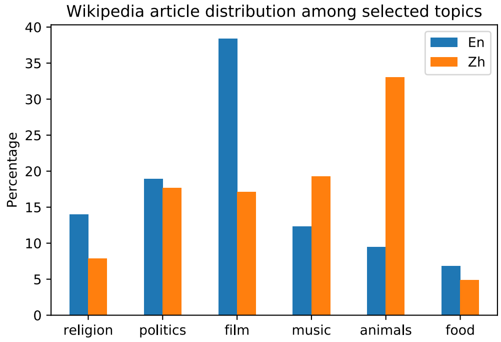


Figure 2: Topic distribution of Wikipedia pages written in English and Chinese.

portion of the parallel dataset originating in the target language).

3.1 Empirical Evidence

In this section we first provide anecdotal evidence that documents originating in different languages possess different distributions over topics. We train two topic classifiers (see Appendix A for details), one for Chinese and the other for English, using the Wikipedia annotated data from Yuan et al. (2018). We apply this classifier to 20,000 documents randomly sampled from English and Chinese Wikipedia. Fig. 2 shows that according to this classifier, English Wikipedia has more pages about entertainment and religion than Chinese Wikipedia, for instance.

Second, we refer the readers to Leech and Fallon (1992)'s study to find evidence that corpora originating in different places may have different word distributions for the same set of topics. In that study, Leech and Fallon (1992) analyzed a British and an American corpus constructed using exactly the same distribution of topics, and yet the word distribution was different, reflecting the different cultural biases of the two countries.

4 Metric: The STDM Score

Given the framework and assumptions introduced in §3, in this section we are going to discuss a practical way to measure STDM. Ideally, we would like to measure a distance between two sample distributions, $z^s \sim \mathcal{D}^S$ and $z^t \sim \mathcal{D}^T$. Unfortunately, we have no access to such latent space. What we observe are realizations in the source and target language. However, it is also an open research question (Hao and Paul, 2018; Yang et al., 2019) how to compare the distribution of $\{\text{src}(z^s)\}$ against $\{\text{tgt}(z^t)\}$, since these are two possibly incomparable corpora in different languages.

In this work, we therefore leverage the existence of a parallel corpus and compare the distribution of $\mathcal{A}^T = \{\text{tgt}(z^t)\}_{z^t \sim \mathcal{D}^T}$ with $\mathcal{A}^S = \{h_{s\to t}(\text{src}(z^s))\}_{z^s \sim \mathcal{D}^S}$. The underlying assumptions are a) we know the originating language of each training example, b) the effect of the change of the word distribution is negligible compared to the shift in topic distribution, and c) the effect of translationese (Baker, 1993; Zhang and Toral, 2019; Toury, 2012) is negligible compared to the actual STDM, and therefore, we can ignore changes to the distribution brought by the mapping $h_{s\to t}$ (we validate this assumption in §C).

Under these assumptions, we define the score as a measure of the topic discrepancy between $\mathcal{A}^S$ and $\mathcal{A}^T$. Let $\mathcal{A} = \mathcal{A}^S \cup \mathcal{A}^T$ be the concatenation of the corpus originating in the source and target language. We first extract topics using LSA. Let $A \in \mathbb{R}^{(n^S+n^T)\times k}$ be the TF-IDF matrix derived from $\mathcal{A}$ where the first n^S rows are representations taken from $\mathcal{A}^S$, the bottom n^T rows are representations of $\mathcal{A}^T$, and k is the number of words in the dictionary. The SVD decomposition of A yields: $A = USV = (U\sqrt{(S)})(\sqrt{(S)}V) = \bar{U}\bar{V}$. Matrix $\bar{U}$ collects topic representations of the original documents; let's denote by $\bar{U}^S$ the first n^S rows corresponding to $\mathcal{A}^S$ and $\bar{U}^T$ the remaining n^T rows corresponding to $\mathcal{A}^T$. Let $C = \bar{U}\bar{U}' = \begin{bmatrix} C^{SS} & C^{ST} \\ C^{ST\prime} & C^{TT} \end{bmatrix}$, where $C^{SS} = \bar{U}^S\bar{U}^{S\prime}$, $C^{ST} = \bar{U}^S\bar{U}^{T\prime}$ and $C^{TT} = \bar{U}^T\bar{U}^{T\prime}$. The STDM score is defined as:

$$\text{score} = \frac{s^{\text{ST}} + s^{\text{TS}}}{s^{\text{SS}} + s^{\text{TT}}}, \text{with } s^{\text{AB}} = \frac{1}{n^A n^B}\sum_{i=1}^{n^A}\sum_{j=1}^{n^B} C_{i,j}^{AB} \quad (1)$$

where s^{AB} measures the average similarity between documents of set A to documents of set B. The score measures the cross-corpus similarity normalized by the within corpus similarity. In the extreme setting where $\mathcal{D}^S$ and $\mathcal{D}^T$ are fully disjoint, then we would have that the off-diagonal block C^{ST} is going to be a zero matrix and therefore the score is equal to 0. When the two domains perfectly match instead, $s^{\text{SS}} = s^{\text{TT}} = s^{\text{ST}} = s^{\text{TS}}$, and therefore, the score is equal to 1. In practice, we expect a score in the range $[0, 1]$. Note that we opted for this metric because of its simplicity, but other methods to extract topics and measure domain mismatch could have been used.

4.1 A Controlled Setting

Similarly to Kilgarriff and Rose (1998), we introduce a *synthetic* benchmark to finely control the

α	0	0.25	0.5	0.75	1.0
STDM score	0.29	0.55	0.78	0.93	0.99

Table 1: STDM score as a function of the parameter α controlling the STDM in the synthetic setting.

domain of the target originating data, and therefore the amount of STDM. The objective is to assess whether the STDM score defined in Eq. 1 captures well the expected amount of mismatch, since we have full control over how the data was generated and its domain.

The key idea of this controlled setting is to use data from two very different domains, assign the source domain to one of them and the target domain to a convex combination of the two. In this work we use EuroParl (Koehn, 2005) as our source originating data, while our target originating data contains a mix of data from EuroParl and OpenSubtitles (Lison and Tiedemann, 2016). Specifically, we consider a French to English translation task with a parallel dataset composed of 10,000 sentences from EuroParl (which "originates" in French) and 10,000 sentences from the target domain (which is "originates" in English)[1]. Let $\alpha \in [0, 1]$, the domain of the target is set to: α EuroParl + $(1 - \alpha)$ OpenSubtitles. α controls how similar the target domain is to the source domain.

For instance, when $\alpha = 0$ then the target domain (OpenSubtitles) is totally out-of-domain with respect to the source domain (EuroParl). When $\alpha = 1$ instead, the target domain matches perfectly the source domain. For intermediate values of α, the match is only partial. Notice that even when $\alpha = 0$, we assume that the parallel dataset is comprised of two halves, one originating from the EuroParl domain (the "French originating" data) and one from OpenSubtitles (the "English originating" data).

Next, we evaluate the STDM score as a function of α. As we can see from Table 1 and as desired, the STDM score increases fairly linearly as we increase the value of α. We refer the reader to Appendix B for experimental details.

4.2 STDM Score of Various Datasets

	De-En	Fi-En	Ru-En	Ne-En	Zh-En	Ja-En
WMT	0.79	0.79	0.76	-	0.65	-
MTNT	-	-	-	-	-	0.69
SMD	0.81	0.71	0.71	0.64	0.71	0.61

Table 2: STDM score on several language pairs using parallel data from WMT, MTNT and from a social media platform (SMD) test sets.

We now evaluate the STDM score on real data. We consider six language pairs, German-English, Finnish-English, Russian-English, Nepali-English, Chinese-English and Japanese-English. We analyze datasets from WMT, MTNT (Michel and Neubig, 2018) and from a social media platform (SMD). For each language, we sample 5000 sentences from WMT newstest sets and MTNT dataset, and 20000 sentences from SMD. We then merge all these datasets and their English translations to compute a common set of topics, making STDM scores comparable across language pairs and datasets.

The results in Table 2 are striking. First, WMT datasets, except for Chinese, show relatively mild signs of STDM and negligible difference across language pairs, suggesting that the data curation process of WMT datasets have made source and target originating corpora rather comparable. The distribution of WMT Chinese originating data instead is rather different because it contains much more local news, while the other languages are mostly about international news which are largely language independent. Interestingly, En-De data derived from social media data has even milder STDM, Fi-En and Ru-En have more substantial STDM. Instead, *MTNT and SMD exhibit strong signs of STDM for distant languages* like Nepali, Chinese and Japanese. This agrees well with our intuition that STDM is more severe for more distant languages associated to more diverse cultures.

5 Machine Translation Baselines

In this section, we turn our attention to how STDM affects training of MT systems. We consider state-of-the-art neural machine translation (NMT) systems based on the transformer architecture (Vaswani et al., 2017) with subword vocabularies learned via byte-pair encoding (BPE) (Sennrich et al., 2015). In order to adapt to the different domains, we employ domain tagging (Zheng et al., 2019) by adding a domain token to the input source sentence, and we cross validate the weights between in-domain and out-domain data[2]. We also

---

[1] Clearly, EuroParl does not originate all in French. However, the chosen domains are so distinct that difference in topic distribution between EuroParl and OpenSubtitles will dominate discrepancies caused by other factors such as the actual origin of the data.

[2] In the controlled setting of §6.1 we found that tagging yields a small but consistent improvement by up to 1 BLEU point, and dataset weighting yields an improvement of up to 0.3 BLEU. ST and BT which are the focus of this study,

use label smoothing (Szegedy et al., 2016) and dropout (Srivastava et al., 2014) to improve generalization, as we focus on low resource language pairs where models tend to severely overfit. Finally, we explore ways to leverage both target and source side monolingual data via back-translation and self-training which we review next.

We simplify our notation and denote with $x^s = \mathrm{src}(z^s)$ and $y^t = \mathrm{tgt}(z^t)$ the source and target originating sentences, $y^s = h_{s\to t}(x^s)$ and $x^t = h_{t\to s}(y^t)$ the corresponding human translations, and $\hat{y}^s$ and $\hat{x}^t$ the corresponding machine translations. The superscript always specifies the domain. We assume access to a parallel dataset $\mathcal{P} = \{(x^s, y^s)\} \cup \{(x^t, y^t)\}$, a source side monolingual dataset $\mathcal{M}^s = \{x^s\}$ and a target side monolingual dataset $\mathcal{M}^t = \{y^t\}$. The test side consists of sentences from the source domain. We study STDM by training with the following algorithms:

Back-translation (BT) (Sennrich et al., 2015) is a very effective data augmentation technique that leverages $\mathcal{M}^t$. The algorithm proceeds in three steps. First, a reverse MT system is trained from target to source using the provided parallel data: $\overleftarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{P}} \log p(x|y;\theta)$. Then, the reverse model is used to translate the target monolingual data: $\hat{x}^t \approx \arg\max_z p(z|y^t; \overleftarrow{\theta})$, for $y^t \sim \mathcal{M}^t$. The maximization is typically approximated by beam search. Finally, the forward model is trained over the concatenation of the original parallel and back-translated data: $\overrightarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{Q}} \log p(y|x;\theta)$ with $\mathcal{Q} = \mathcal{P} \cup \{\hat{x}^t, y^t\}_{y^t\sim\mathcal{M}^t}$. In practice, the parallel data is weighted more in the loss, with a weight selected via hyper-parameter search on the validation set.

Self-Training (ST) (He et al., 2020) is another method for data augmentation that instead leverages $\mathcal{M}^s$; see Alg. 1 in Appendix D.

Also this algorithm proceeds in three steps. First, a forward MT system is trained from source to target using the provided parallel data: $\overrightarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{P}} \log p(y|x;\theta)$. Then, this model is used to translate the source monolingual data: $\hat{y}^s \approx \arg\max_z p(z|x^s; \overrightarrow{\theta})$, for $x^s \sim \mathcal{M}^s$. Finally, the forward model is retrained over the concatenation of the original parallel and forward-translated data: $\overrightarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{Q}} \log p(y|x;\theta)$ with $\mathcal{Q} = \mathcal{P} \cup \{x^s, \hat{y}^s\}_{y^t\sim\mathcal{M}^s}$. As with BT, the parallel data is weighted more in the loss.

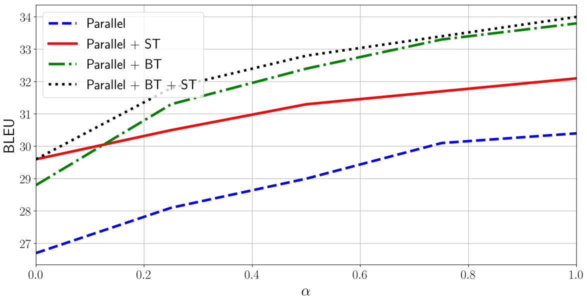


Figure 3: BLEU score in Fr-En as a function of the amount of STDM. The target domain is fully out-of-domain when $\alpha = 0$, and fully in-domain when $\alpha = 1$.

ST + BT also proceeds in three steps. First, we train an initial forward and reverse model using the parallel dataset. Second, we back-translate target side monolingual data using the reverse model and iteratively forward translate source side monolingual data using the forward model. We then retrain the forward model from random initialization using the union of the original parallel dataset, the synthetic back-translated data, and the synthetic forward translated data at the last iteration of the ST algorithm. This combined algorithm aims at leveraging the strengths of both ST and BT: the use of in-domain source monolingual data and the use of synthetic data with correct targets, respectively.

6 Machine Translation Results

In this section, we first study the effect of STDM on NMT using the controlled setting introduced in §4.1 which enables us to assess the influence of various factors, such as the extent to which target originating data is out-of-domain, and the effect of monolingual data size. We then report experiments on genuine low resource language pairs, namely Nepali-English and English-Myanmar. We report SacreBLEU (Post, 2018).

6.1 Controlled Setting

In the default setting, we have a parallel dataset with 20,000 parallel sentences. 10,000 are in-domain source originating data (EuroParl) and the remaining 10,000 are target originating data from a mix of domains, controlled by $\alpha \in [0, 1]$: α EuroParl + $(1-\alpha)$ OpenSubtitles. The source side monolingual dataset has 100,000 French sentences from EuroParl. The target side monolingual dataset has 100,000 English sentences from: α EuroParl + $(1-\alpha)$ OpenSubtitles. Finally, the test set consists of novel French sentences from EuroParl which we wish to translate to English.

---

improve by more than 2 BLEU points instead.

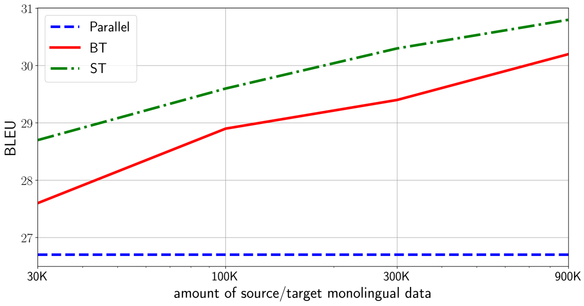


Figure 4: BLEU as a function of the amount of monolingual data when $\alpha = 0$.

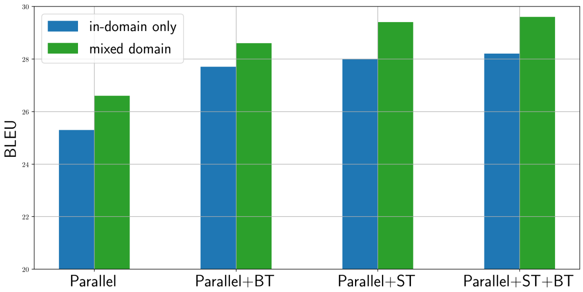


Figure 5: BLEU when using only source originating in-domain data (blue bars) or also out-of-domain target originating data (green bars) for $\alpha = 0$.

We tune model hyper-parameters (e.g., number of layers and hidden state size) and BPE size on the validation set. Based on cross-validation, when training on datasets with less than 300k parallel sentences (including those from ST or BT), we use a 5-layer transformer with 8M parameters. The number of attention heads, embedding dimension and inner-layer dimension are 2, 256, 512, respectively. When training on bigger datasets, we use a bigger transformer with 5 layers, 8 attention heads, 1024 embedding dimension, 2048 inner-layer dimension and a total of 110M parameters. We find that using bigger model can better utilize the monolingual data, but we do not find the bigger model benefits when training with less than 300k parallel sentences. The full list of hyper-parameters can be found in Appendix E.

Varying amount of STDM. In Fig. 3, we benchmark our baseline approaches while varying α (see §4.1 and Tab. 1), which controls the overlap between source and target domain.

First, we observe improved BLEU (Papineni et al., 2002) scores for all methods as we increase α. Second, there is a big gap between the baseline trained on parallel data only and methods which leverage monolingual data. Third, combining ST and BT works better than each individual method, confirming that these approaches are complementary. Finally, BT works better than ST but the gap reduces as the target domain becomes increasingly different from the source domain (small values of α). In the extreme case of STDM ($\alpha = 0$), ST outperforms BT. In fact, we observe that the gain of BT over the baseline decreases as α decreases, despite that the amount of monolingual data and parallel data remains constant across these experiments, thus showing that *BT is less effective in the presence of STDM*.

Varying amount of monolingual data. We next explore how the quantity of monolingual data affects performance and if the relative gain of ST over BT when $\alpha = 0$ disappears as we provide BT with more monolingual data. The experiment in Fig. 4 shows that a) the gain in BLEU tapers off exponentially with the amount of data (notice the log-scale in the x-axis), b) for the same amount of monolingual data ST is always better than BT and by roughly the same amount, and c) BT would require about 3 times more target monolingual data (which is out-of-domain) to yield the performance of ST. Therefore, *increasing the amount of data can compensate for domain mismatch*.

Varying amount of in-domain data. Now we explore whether, in the presence of extreme STDM ($\alpha = 0$), it may be worth restricting the training data to only contain in-domain source originating sentences. In this case, the parallel set is reduced to 10,000 EuroParl sentences, the target side monolingual data is removed and back-translation is performed on the target side of the parallel dataset. Fig. 5 demonstrates that in all cases it is better to include the out-of-domain data originating on the target side (green bars). Particularly in the low resource settings considered here, *neural models benefit from all available examples even if these are out-of-domain*.

Finally, we investigate how to construct a parallel dataset when STDM is significant ($\alpha = 0$), i.e. the target domain is OpenSubtitles. If we have a translation budget of 20,000 sentences, is it best to translate 20,000 sentences from EuroParl or to also include sentences from OpenSubtitles? This is not obvious when training with BT, since the backward model may benefit from in-domain OpenSubtitles data. In order to answer this question, we consider a *parallel* dataset with 20,000 sentences defined as: β EuroParl + $(1-\beta)$ OpenSubtitles, with $\beta \in [0, 1]$.

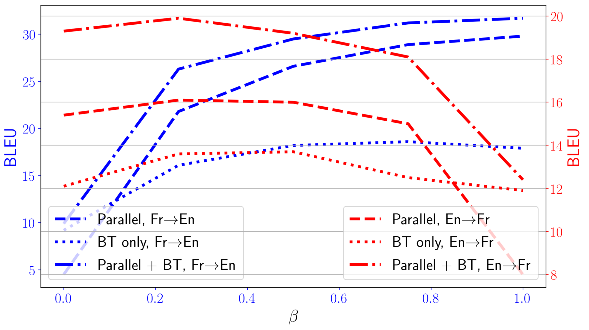


Figure 6: BLEU score as a function of the proportion of parallel data originating in the source and target domain domain. When $\beta = 0$ all parallel data originates from OpenSubtitles, when $\beta = 1$ all parallel data originates from EuroParl. Source and target monolingual corpora have 900,000 sentences from EuroParl and OpenSubtitles, respectively. The blue curves show BLEU in the forward direction (Fr-En translation of EuroParl data). The red curves show BLEU in the reverse direction (En-Fr translation of OpenSubtitles sentences).

When $\beta = 0$, the parallel dataset is out-of-domain; when $\beta = 1$ the parallel data is all in-domain. The target side monolingual dataset is fixed and contains 900,000 sentences from OpenSubtitles.

Fig. 6 shows that taking *all* sentences from EuroParl ($\beta = 1$) is optimal when translating from French (EuroParl) to English (blue curves). At high values of β, we observe a slight decrease in accuracy for models trained only on back-translated data (dotted line), confirming that BT loses its effectiveness when the reverse model is trained on out-of-domain data. However, this is compensated by the gains brought by the additional in-domain parallel sentences (dashed line). In the more natural setting in which the model is trained on both parallel and back-translated data (dash-dotted line), we see monotonic improvement in accuracy with β. A similar trend is observed in the other direction (English to French, red lines). Therefore, if the goal is to maximize translation accuracy in *both* directions, an intermediate value of β (≈ 0.5) is more desirable.

6.2 Low-Resource MT

We now measure STDM on two low-resource language pairs and verify whether in practice BT's performance deteriorates as expected when the STDM score is low, while the combination of ST+BT offers better generalization. We consider two low-resource language pairs, Nepali-English (Ne-En) and English-Myanmar (En-My). Nepali and Myanmar are spoken in regions with unique local context that is very distinct from English-speaking regions, and thus these make good language pairs for studying the STDM setting in real life.

Model	Ne → En	En → My
	STDM score=0.64	STDM score=0.27
baseline	20.4	28.1
BT	22.3	30.0
ST	22.1	31.9
ST + BT	22.9	32.4

Table 3: BLEU scores for the Nepali to English and English to Myanmar translation task.

Data. The Ne-En parallel dataset is composed of 40,000 sentences originating in Nepali and only 7,500 sentences originating in English. There are 5,000 sentences in the validation and test sets all originating in Nepali. We also have 1.8M monolingual sentences in Nepali and English, collected from public posts from a social media platform. This dataset closely resembles our idealized setting of Fig. 1. The STDM score of this dataset is 0.64 (see Tab. 2) and is analogous to our synthetic setting (§6.1) where α is low but β is large.

The En-My parallel data is taken from the Asian Language Treebank (ALT) corpus (Thu et al., 2016; Ding et al., 2018, 2019) with 18,088 training sentences all originating from English news. The validation and test sets have 1,000 sentences each, all originating from English. Following Chen et al. (2019), we use 5M English sentences from NewsCrawl as source side monolingual data and 100K Myanmar sentences from Common Crawl as target side monolingual data. The STDM Score is even lower on this dataset, only 0.27. Comparing to our controlled setting this dataset would have β equal to 1 and presumably a small value of α, an ideal setting for ST.

Models. We run model hyper-parameter sweep on the validation set and pick the best-performing model architecture (e.g., number of layers and hidden layer sizes). On both datasets, the parallel data baseline is a 5-layer transformer with 8 attention heads, 512 embedding dimensions and 2048 inner-layer dimensions, which consists of 42M parameters. When training with BT and ST, we use a 6-layer transformer with 8 attention heads, 1024 embedding dimensions, 2048 inner-layer dimensions, resulting in 186M parameters. The detailed hyper-parameters search range can be found in Appendix §E.

Results. In Table 3, we observe that on the Ne-En task augmenting the parallel dataset with either forward- or back-translated monolingual data achieves almost 2 BLEU points improvement over the supervised baseline. On the En-My task where STDM is more severe (with a value of 0.27 which is similar to $\alpha = 0$ in Tab. 1), BT outperforms the baseline by 1.9 BLEU, while ST improves by twice as much. This is perhaps not surprising since source side monolingual data is in-domain and abundant, while target side monolingual data is scarce and out-of-domain. On both tasks, combining ST and BT outperforms each individual method.

7 Conclusions

While the commonly used WMT datasets exhibit mild STDM, we find that less curated datasets, often in more distant and lower resource language pairs (§4.2), exhibit much stronger STDM. How can these findings inform us on how to better set up a MT system in practice? Our first recommendation is to be aware of possible STDM, and *(i)* check whether origin language information is available. If this is available, then it may be possible to *(ii)* quantitatively measure STDM as described in §4. Next, *(iii)* be aware that when STDM is severe (STDM score is low), BT performance suffers (Fig. 3). However, *(iv)* we may be able to combat this by increasing the amount of target side (out-of-domain) monolingual data (Fig. 4) and *(v)* by combining BT with ST (Fig. 3).

Of course, the relative ratio of monolingual data in the source and target side and the actual degradation brought by STDM depend on the particular language pair. The more distant two languages, the more difficult the learning task and the more data is needed to learn it. And finally, the less parallel data there is, the more monolingual data will be needed to compensate. The intricate dependency between all these factors merits future investigation.

References

Mona Baker. 1993. Corpus linguistics and translation studies: Implications and applications. *Text and technology: In honour of John Sinclair*, 233:250.

Silvia Bernardini and Federico Zanettin. 2004. When is a universal not a universal. *Translation universals: do they exist? John Benjamin publisher Edited by Anna Mauranen and Pekka Kujammak*, pages 51–62.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. 2019. Findings of the 2019 conference on machine translation (wmt19). In *Proc. of WMT*.

Jordan Boyd-Graber and David M. Blei. 2009. Multilingual topic models for unaligned text. In *Conference on Uncertainty in Artificial Intelligence*.

David Britain. 2013. *Space, Diffusion and Mobility*. Wiley publishers; Book Editor(s): J.K. Chambers Natalie Schilling First. Chapter 22.

Franck Burlot and François Yvon. 2018. Using monolingual data in neural machine translation: a systematic study. In *Empirical Methods in Natural Language Processing*.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged back-translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 53–63, Florence, Italy. Association for Computational Linguistics.

Peng-Jen Chen, Jiajun Shen, Matt Le, Vishrav Chaudhary, Ahmed El-Kishky, Guillaume Wenzek, Myle Ott, and Marc'Aurelio Ranzato. 2019. Facebook ai's wat19 myanmar-english translation task submission. In *Workshop on Asian Translation*.

Chenchen Ding, Hnin Thu Zar Aye, Win Pa Pa, Khin Thandar Nwet, Khin Mar Soe, Masao Utiyama, and Eiichiro Sumita. 2019. Towards Burmese (Myanmar) morphological analysis: Syllable-based tokenization and part-of-speech tagging. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(1):5.

Chenchen Ding, Masao Utiyama, and Eiichiro Sumita. 2018. NOVA: A feasible and flexible annotation system for joint tokenization and part-of-speech tagging. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(2):17.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Conference on Empirical Methods in Natural Language Processing*.

Jacob Eisenstein, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Annual Meeting of the Association for Computational Linguistics*.

John Rupert Firth. 1935. On sociological linguistics. *Transactions of the Royal Society*, pages 67–69.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation.

Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *The 17th International Conference on Computational Linguistics*.

E.D. Gutierrez, Ekaterina Shutova annd Patricia Lichtenstein, Gerard de Melo, and Luca Gilardi. 2016. Detecting cross-cultural differences using a multilingual topic model. In *Conference of the Association for Computational Linguistics*.

Shudong Hao and Michael J. Paul. 2018. Learning multilingual topics from incomparable corpora. In *International Conference on Computational Linguistics (COLING)*.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations*.

Ann Irvine and Chris Callison-Burch. 2013. Combining bilingual and comparable corpora for low resource machine translation. In *Proceedings of the eighth workshop on statistical machine translation*, pages 262–270.

Barbara Johnstone. 2010. *Language and place*. R. Mesthrie and W. Wolfram, editors, Cambridge Handbook of Sociolinguistics. Cambridge University Press.

Adam Kilgarriff and Tony Rose. 1998. Measures for corpus similarity and homogeneity. In *Conference on Empirical Methods in Natural Language Processing*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. *MT Summit*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Geoffrey Leech and Roger Fallon. 1992. Computer corpora: What do they tell us about culture? *ICAME Journal Computers in English Linguistics*, (16).

Bill Yuchen Lin, Frank F. Xu, Kenny Q. Zhu, and Seung won Hwang. 2018. Mining cross-cultural differences and similarities in social media. In *Conference of the Association for Computational Linguistics*.

P. Lison and J. Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *10th International Conference on Language Resources and Evaluation (LREC)*.

Qiaozhu Mei, Chao Liu, and Hang Su. 2006. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *WWW*.

Paul Michel and Graham Neubig. 2018. Mtnt: A testbed for machine translation of noisy text. In *Conference on Empirical Methods in Natural Language Processing*.

David Mimno, Hanna M. Wallach, Jason Naradowsky, David A. Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Conference on Empirical Methods in Natural Language Processing*.

D.S. Munteanu, A. Fraser, and D. Marcu. 2004. Improved machine translation performance via parallel sentence extraction from comparable corpora. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Brussels, Belgium.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *International Conference on Machine Learning*.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 86–96.

Anders Søgaard, Sebastian Ruder, and Ivan Vulic. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Conference of the Association for Computational Linguistics (ACL)*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *IEEE conference on computer vision and pattern recognition*.

Ye Kyaw Thu, Win Pa Pa, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Introducing the asian language treebank (alt). In *LREC*.

Gideo Toury. 2012. *Descriptive translation studies and beyond: Revised edition*. John Benjamins Publishing.

Nicola Ueffing. 2006. Using monolingual source-language data to improve mt performance. In *IWSLT*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proc. of NIPS*.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Conference on Empirical Methods in Natural Language Processing*.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Conference on Empirical Methods in Natural Language Processing*.

Weiwei Yang, Jordan Boyd-Graber, and Philip Resnik. 2019. A multilingual topic model for learning weighted topic links across corpora with low comparability. In *Conference on Empirical Methods in Natural Language Processing*.

David Yarowski. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Annual Meeting of the Association for Computational Linguistics*.

Michelle Yuan, Benjamin Van Durme, and Jordan Boyd-Graber. 2018. Multilingual anchoring: Interactive topic modeling and alignment across languages. In *Neural Information Processing Systems*.

Mike Zhang and Antonio Toral. 2019. The effect of translationese in machine translation test sets. *arXiv*, abs/1906.08069.

Renjie Zheng, Hairong Liu, Mingbo Ma, Baigong Zheng, and Liang Huang. 2019. Robust machine translation with domain sensitive pseudo-sources: Baidu-OSU WMT19 MT robustness shared task system report. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 559–564, Florence, Italy. Association for Computational Linguistics.

A Topic Classifiers

In this section we provide the experimental details of the findings reported in §3.1.

Dataset We use the WIKISHORT dataset provided by Yuan et al. (2018) [3]. The dataset contains thousands of English and Chinese wikipedia articles, and each article is labeled with one of the six categories, film, music, animals, politics, religion, and food. We use the train split in the dataset for training, validate with test split. The train split includes 7730 and 7095 English and Chinese articles respectively. To make the classifier better differentiate articles that is not one of the six categories, we uniformly sample 1500 wikipedia articles from both English and Chinese wikidump [4] and add them into the training data with 'other' label.

Preprocessing WIKISHORT provides shortened text of Wikipedia articles. To generate the shortened text of articles with 'other' label, we use the text from the first paragraph of the articles and follow similar preprocessing as in Yuan et al. (2018). For English articles, we lowercase the paragraph, tokenize by word, filter out stop words, lemmatize the words and remove all the punctuations. For Chinese articles, we apply Stanford CoreNLP Tokenizer [5] to segment the Chinese text.

Training We train two multi-nomial logistic regression models, one for English and one for Chinese. To extract features from the shortened text, we compute tf-idf weights from the training data. We use the LogisticRegression implementation of SCIKIT-LEARN (Pedregosa et al., 2011) with default setting to train the models. We tune the regularization term C on validation set. However, we do not observe significant accuracy difference between different C values, therefore we use default setting $C = 1.0$. The accuracy on the test set of the English and Chinese classifiers are 92.5% and 77.9%, respectively.

Prediction on Wikipedia articles To estimate the category distribution of English and Chinese Wikipedia articles, we uniformly sampled 8500 articles from wikidump for each language, follow the same preprocessing and feature extraction as how we does for the training data, and we use the classifiers to predict the category of each article. After removing articles labeled "other", we re-normalize the distribution of the prediction of the main six categories and report the values in 2.

B Computing STDM score

In this section we provide the experimental details of the STDM evaluations performed in §4.1 and §4.2.

Dataset In §4.1, we have described the details for constructing datasets from EuroParl and OpenSubtitles with different amount of STDM.

For the experiments in §4.2, we evaluate the STDM score on datasets with known language origins. We use three different data sources, including WMT newstest sets, MTNT (Michel and Neubig, 2018) and a social media platform (SMD). For the WMT newstest sets, we combine datasets from year 2014 to 2019 and sample 5000 sentences for each language. For the analysis on MTNT dataset, we use the train split in the dataset and sample 5000 sentences for each language. For SMD, 20000 sentences are sampled for each language.

Preprocessing We use SentencePiece (Kudo and Richardson, 2018) to learn a BPE vocabulary of size 10000 over the combined English text corpus of all datasets. We preprocess sentences from all datasets with BPE and remove sentences with less than 10 BPE tokens.

Topic Learning and STDM Score Computing For each dataset, we derive the TF-IDF matrix from the preprocessed sentences and perform an SVD decomposition of the matrix. We retain the top 400 eigenvalues and collect the corresponding topic representations of the original sentences. The topic representations of the sentences are used to calculate the STDM score as described in §4.

C The Effect of Translationese

In §3 we have made the assumption that the effect of translationese is negligible when estimating STDM. However, there are previous studies showing clear artifacts in (human) translations (Baker, 1993; Zhang and Toral, 2019; Toury, 2012). In this section we aim at assessing whether our STDM score is affected by translationese.

We consider the WMT'17 De-En dataset from Ott et al. (2018) which contains double translations of source and target originating sentences.

---

[3] https://github.com/forest-snow/mtanchor_demo#data

[4] https://dumps.wikimedia.org/

[5] https://stanfordnlp.github.io/stanza/tokenize.html

From this, we construct paired inputs and labels, $\{(h_{s\to t}(h_{t\to s}(\text{tgt}(z^t))), 1)\} \cup \{(\text{tgt}(z^t), 0)\}$, and train two classifiers to predict whether or not the input is translationese. The first classifier takes as input a TF-IDF representation w of the sentence, while the second classifier takes only the corresponding topic distribution: $\bar{V}w$. On this binary task a linear classifier achieves 58% accuracy on the test set with TF-IDF input representations, and only 52% when given just the topic distribution. If we apply the same binary classifier in the topic space to discriminate between sentences originating in the source and target domain ($\text{tgt}(z^t)$ VS. $h_{s\to t}(\text{src}(z^s))$), the accuracy increases to 64%.

We conclude that once we control for domain effect (by discriminating the same set of sentences in their original form versus their double translationese form), the accuracy is much lower than previously reported (Zhang and Toral, 2019), and working in the topic space further removes translationese artifacts. Therefore, the STDM score computed in the topic space is unlikely affected by such artifacts and captures the desired discrepancy between the source and the target domains.

D Self-Training

```
1 Data: Given a parallel dataset $\mathcal{P}$ and a source
      monolingual dataset $\mathcal{M}^s$ with $N^s$ examples;
2 Noise: Let $n(x)$ be a function that adds noise to the
      input by dropping, swapping and blanking words;
3 Hyper-params: Let $k$ be the number of iterations
      and $A_1 < \cdots < A_k \leq N_S$ be the number of
      samples to add at each iteration;
4 Train a forward model:
      $\overrightarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{P}} \log p(y|x;\theta)$;
5 for $t$ in $[1 \dots k]$ do
6     forward-translate data:
          $(\hat{y}^s, v) \approx \arg\max_z p(z|x^s; \overrightarrow{\theta})$, for $x^s \in \mathcal{M}^s$,
          where $v$ is the model score;
7     Let $\bar{\mathcal{M}}^s \subset \mathcal{M}^s$ containing the top-$A_t$ highest
          scoring examples according to $v$;
8     re-train forward model:
          $\overrightarrow{\theta} = \arg\max_\theta \mathbb{E}_{(x,y)\sim\mathcal{Q}} \log p(y|x;\theta)$ with
          $\mathcal{Q} = \mathcal{P} \cup \{n(x^s), \hat{y}^s\}_{x^s\sim\bar{\mathcal{M}}^s}$.
  end
```

Algorithm 1: Self-Training algorithm.

Alg. 1 describes self-training, a data augmentation method that leverages $\mathcal{M}^s$, using the notation of §5. First, a baseline forward model is trained on the parallel data (line 4). Second, this initial model is applied to the source monolingual data (line 6). Finally, the forward model is re-trained from random initialization by augmenting the original parallel dataset with the forward-translated data. As with BT, the parallel dataset receives more weight in the loss. In order to increase robustness to prediction mistakes, we make the algorithm iterative and add only the examples for which the model was most confident (line 3, loop in line 5 and line 7). In our experiments we iterate three times. We also inject noise to the input sentences, in the form of word swap and drop (Lample et al., 2018), to further improve generalization (line 8).

E Hyper-parameters Used in MT Experiments

In this section, we report the hyper-parameters used in Sec. 6.

In all our experiments we use the standard machine learning methodology of model cross-validation to select hyper-parameters. We train models with several random combination of hyper-parameters and select the best configuration based on the performance on the validation set. We finally report results on the test set.

Data The full list of datasets we have considered in this work with some basic statistics is reported in Tab. 4

We jointly learn BPEs on both source and target languages. First, we train an MT system on the parallel data for each BPE setting, with values in the set {3000, 5000, 10000, 20000}. Then, we select the number of BPE tokens by selecting the setting that yields the best performance on the validation set. We report below the value that worked best in each experiment.

Loss and Optimizer All models are trained using cross-entropy loss with label smoothing (Szegedy et al., 2016) equal to 0.2. The optimizer is Adam (Kingma and Ba, 2015) with beta1 = 0.9, beta2 = 0.98 and warm-up steps 4000. We share embeddings across encoder and decoder, and input and output lookup tables. We use fixed batch size with 4000 tokens per GPU, and we train with 4 GPUs in fp16. Other optimization hyper-parameters are reported below.

Model Architecture and Training To decide the model architecture hyper-parameters for different amount of parallel and monolingual data, we use random search to find the setting that yields the best performance on validation set. The range of values that we used in our random hyper-parameter search are:

MT Experiments	Language Pair	BPE size	Origin	Parallel data		Monolingual data	
				Domain	# sents (train/valid/test)	Domain	# sents
Controlled Setting	Fr->En	5000	source	Europarl	10K / 10K / 10K	Europarl	900K
			target	OpenSubtitle	10K / 10K / 10K	OpenSubtitle	900K
Low-Resource MT	Ne->En	5000	source	Social Media	40K / 5K / 5K	Social Media	1.8M
			target	Social Media	7.5K / 5K / 5K	Social Media	1.8M
	En-My	10000	source	ALT(News)	18K / 1K / 1K	NewsCrawl	5M
			target	-	-	CommonCrawl	100K

Table 4: Datasets used for the machine translation experiments of Sec. 6

- Layers: {4, 5, 6}
- Embedding dimension: {256, 512, 1024}
- Inner-layer dimension: {512, 1024, 2048, 4096}
- Attention Heads: {2, 4, 8, 16}

We report the model architecture of each experiment in §E.1.

For each data point which corresponds to a particular combination of $(\mathcal{P}, \mathcal{M}^s, \mathcal{S}^t, \alpha, \beta, \text{training procedure})$, we use random search to sweep over hyper-parameters. The hyper-parameters are dropout rate, learning rate, source side noise level for ST experiments and upsample ratio between parallel data, back-translated data and self-translated data. For all experiments, the dropout rates are {0.1, 0.2, 0.3, 0.4, 0.5}, and the learning rate takes values in {0.0007, 0.001, 0.003, 0.005}.

E.1 Hyper-parameter for Controlled Setting Experiments

We use the same BPEs for all the experiments in the controlled setting of §6.1. The BPE size is 5000 and it is shared between English and French.

Varying amount of STDM. We use embedding dimension 256, inner-layer dimension 512, 2 attention heads. We sweep the upsampling ratio of parallel data, back-translated data and self-training data in a range between 1 and 8.

We train the ST system with 2 iterations ($k = 2$), where $A_1 = 30K$ and $A_2 = 100K$. In each iteration, we use random search to sweep over different source side noise, and we report the model with the best performance based on validation set. The values of input noise are as follows: Word shuffling {0, 2, 3}, word dropout {0.0, 0.1, 0.2}, word blank {0.0, 0.1, 0.2}.

Varying amount of monolingual data. When the monolingual data has less than 300k sentences, we use embedding dimension 256, inner-layer dimension 512 and 2 attention heads. For bigger monolingual datasets, we use embedding dimension 1024, inner-layer dimension 2048 and 8 attention heads. We sweep the upsampling ratio of parallel data, back-translated data and self-training data in a range between 1 and 8.

We train the ST system with 5 iterations ($k = 5$), where $A_1 = 10K$, $A_2 = 30K$, $A_3 = 100K$, $A_4 = 300K$ and $A_5 = 900K$. The values of input noise are as follows: Word shuffling {0, 2, 3}, word dropout {0.0, 0.1, 0.2}, word blank {0.0, 0.1, 0.2}.

Varying amount of in-domain data. When training only with parallel data, we use embedding dimension 256, inner-layer dimension 512, and 2 attention heads. For models trained with BT-only and parallel + BT instead, we use embedding dimension 1024, inner-layer dimension 2048 and 8 attention heads. We sweep the parallel data upsampling ratio in a range between 1 and 16.

E.2 Hyper-parameter for Low-Resource MT Experiments

We use a joint BPE tokenization with 5000 tokens for En $\rightarrow$ Ne and 10000 tokens for En $\rightarrow$ My.

On both datasets (Ne $\rightarrow$ En and En $\rightarrow$ My), the baseline trained only on the parallel dataset is a 5-layer transformer model with 512 embedding dimensions, 2048 inner-layer dimensions and 8 attention heads. For models trained with BT and ST, we use a 6-layer transformer with 1024 embedding dimensions, 2048 inner-layer dimensions and 8 attention heads. We sweep the parallel data and ST data ratio between 1 and 32.

For both language pairs, we train the ST systems with 3 iterations ($k = 3$). For Ne $\rightarrow$ En, we use A_1 = 600K, A_2 = 1M, A_3 = 1.8M. For En $\rightarrow$ My, we use A_1 = 1M, A_2 = 3M and A_3 = 5M. The values of input noise are as follows: Word shuffling {0, 2, 3}, word dropout {0.0, 0.1, 0.2}, word blank {0.0, 0.1, 0.2}.

Cross-Topic Rumor Detection using Topic-Mixtures

Xiaoying Ren, Jing Jiang[*]
Singapore Management University
80 Stamford Road
Singapore 178902
`lemontreejying@gmail.com`
`jingjiang@smu.edu.sg`

Ling Min Serena Khoo, Hai Leong Chieu
DSO National Laboratories
12 Science Park Drive
Singapore 118225
`{klingmin,chaileon}@dso.org.sg`

Abstract

There has been much interest in rumor detection using deep learning models in recent years. A well-known limitation of deep learning models is that they tend to learn superficial patterns, which restricts their generalization ability. We find that this is also true for cross-topic rumor detection. In this paper, we propose a method inspired by the "mixture of experts" paradigm. We assume that the prediction of the rumor class label given an instance is dependent on the topic distribution of the instance. After deriving a vector representation for each topic, given an instance, we derive a "topic mixture" vector for the instance based on its topic distribution. This topic mixture is combined with the vector representation of the instance itself to make rumor predictions. Our experiments show that our proposed method can outperform two baseline debiasing methods in a cross-topic setting. In a synthetic setting when we removed topic-specific words, our method also works better than the baselines, showing that our method does not rely on superficial features.

1 Introduction

Recently, there has been much interest in detecting online false information such as rumors and fake news. Existing work has explored different features including network structures (Ma et al., 2019a), propagation paths (Liu and Wu, 2018), user credibility (Castillo et al., 2011) and the fusion of heterogeneous data such as image and text (Wang et al., 2018). However, these proposed algorithms still cannot be easily deployed for real-world applications, and one of the key reasons is that, just like many other NLP problems, rumor or fake news detection models may easily overfit the training data and thus cannot perform well on new data. The problem can be more serious with deep learning solutions, because deep neural networks tend to learn superficial patterns that are specific to the training data but do not always generalize well (Wang et al., 2018).

In this work, we study the task of rumor detection and focus on the problem of adapting a rumor detection model trained on a set of *source* topics to a *target* topic, which we refer to as *cross-topic rumor detection*. In a recent study by Khoo et al. (2020), the authors compared the performance of rumor detection in an in-topic setting and an out-of-topic setting. They found that their model could achieve 77.4% macro F-score on the in-topic testing data but the performance of the same classifier dropped to 39.5% when applied to out-of-topic testing data, which describe events different from the training events.

In this paper, we propose a method inspired by the "mixture of experts" paradigm, abbreviated as "MOE". Understanding that the rumor prediction model may work differently for different topics, we assume that the prediction result on an instance is dependent on the topic distribution of that instance. While a standard method is to train topic-specific classifiers and then use the topic distribution to combine these topic-specific classifiers, we propose a different approach where the topic distribution is used to linearly combine a set of vectors representing different topics. This gives us a "topic-mixture" given an example. This topic-mixture vector of the example is concatenated with the vector representation of the example itself and used as the input to a neural network model for rumor label prediction.

We implement our method on top of a state-of-the-art StA-HiTPLAN model and conduct experiments using the PHEME dataset. Compared with two baseline methods that also perform debiasing, we find that our method can achieve clearly better cross-topic performance. We also experiment with

---

[*] Corresponding author

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1534–1538

modified within-topic data where we intentionally remove topic-specific words. This creates a setting where it is hard for models to rely on topic-specific words to make rumor predictions. We find that our method can also outperform the baselines substantially.

2 Performance Degradation in Cross-Topic Rumur Detection

In this section, we present a case study on the PHEME dataset to quantify the degree of overfitting of an existing model by analyzing the influence of topic-specific words.

Concretely, we use the PHEME dataset, which has five topics. We use four topics during training and the remaining one for out-of-domain testing. After obtaining a trained hierarchical transformer model (Vaswani et al., 2017), we perform post-hoc testing by applying it to different topics, with K topic-specific words masked to examine the performance drop. Here the topic-specific words are identified based on log-odds ratio with Dirichlet prior (Monroe et al., 2008), and we regard these topic-specific words as possible spurious patterns. It is a common way to identify words that are statistically over-represented in a particular population compared to others. For the in-domain testing, we split the data as 7:2:1 for training, testing and validation. Experiments are performed using $K \in \{20, 50, 100, 200\}$.

Results: The partial results are shown in Table 1. It is noteworthy that the accuracy drops from 67.69% to 36.7% when we only mask the top-20 frequent event-aware words in in-domain set - the model is highly sensitive to event sensitive patterns. Besides, the little dropping in accuracy with the out-of-domain setting when we mask top-20 out-of-domain words may indicate that we mask some training unseen words compared with non-mask setting. These experiments confirm our hypothesis that the baseline classifier is primarily learning topical correlations, and motivate the need for a debiased classification approach which we will describe next.

3 Method

3.1 Notation

Let x be an input, which is a thread represented as a sequence of tokens. We assume that x consists of a sequence of posts $x = x_1, x_2, \ldots, x_T$ chronologically ordered, in which x_1 represents a source post and x_i $(i > 1)$ represents a reply post. Let y be the rumor label (e.g., true rumor, false rumor, etc.) we want to predict. We assume that the training data come from a set of M different topics, and we use $\{\mathcal{S}_i\}_{i=1}^{M}$ to denote the data, where $\mathcal{S}_i \triangleq \{(x_n^i, y_n^i)\}_{n=1}^{|\mathcal{S}_i|}$. Our goal is to train a rumor detection classifier using the labeled data from the M topics such that the classifier can work well on a target example.

tA-HiTPLAN	**in-domain**	**out-of-domain**
`{non-mask}`	0.6769	0.3441
`{+MASK TOP 20}`	0.3670	0.3425
`{+MASK TOP 50}`	0.3526	0.3255
`{+MASK TOP 100}`	0.3413	0.3122
`{+MASK TOP 200}`	0.3202	0.2903

Table 1: Accuracy on PHEME event-5 datset.

3.2 Mixture Of Experts

Our idea is inspired by Mixture of Experts models (Jacobs et al., 1991). Specifically, we assume that each example $\mathbf{x}$ has a distribution over the M training topics. Let t be a variable denoting topic. We model $p(y|x)$ as follows:

$$p(y|x) \quad = \quad \sum_{i=1}^{M} p(t=i|x) p(y|x, t=i). \quad (1)$$

Normally, to model $p(t|x)$ and $p(y|x,t)$, we can train parameterized models $p(t|x;\theta_1)$ and $p(y|x,t;\theta_2)$ using our training data, because our examples have clear topic labels. However, if the number of topics is large, or the number of training instances for each topic is small, training such topic-specific models may not work well. Moreover, if we train independent models for each training topic and combine their out-of-domain testing result as a whole, the result may be unsatisfactory because each model may be overfitting a specific topic. Our initial experimental observation also verifies that independent training method works well on in-topic setting but does not perform well on out-of-topic setting. Here we explore an alternative approach as described below.

We assume that x and t are both represented as vectors (which we will explain later). We can then use the following neural network model to model $p(y|x,t)$:

$$\begin{aligned} p(y|x,t) &= p(y|\mathbf{x} \oplus \mathbf{t}; \theta) \\ &= \frac{\exp(\theta_y \cdot (\mathbf{x} \oplus \mathbf{t}))}{\sum_{y'} \exp(\theta_{y'} \cdot (\mathbf{x} \oplus \mathbf{t}))}, \end{aligned}$$

where θ_y are vectors to be learned and $\oplus$ means vector concatenation.

Now we can make an approximation of Eqn. (1) as follows:

$$\begin{aligned} p(y|x) &= \sum_{i=1}^{M} p(y|x,t)p(t|x) \\ &= p(y|\mathbf{x} \oplus \sum_{i=1}^{M} p(t=i|x)\mathbf{t}_i;\theta), \quad (2) \end{aligned}$$

where $\mathbf{t}_i$ is a vector representation of topic i. We can see that instead of computing $p(y|x, t = i)$ for each i, and then use $p(t = i|x)$ to obtain a weighted sum of these $p(y|x, t = i)$, we first get a sum of the vector representations of different topics weighted by $p(t|x)$, and then use this weighted sum to compute $p(y|x)$.

To obtain a vector representation of x, we can use BERT to process the sequence of tokens in x and then use the vector representing the `[CLS]` token at the top layer as $\mathbf{x}$. For each topic t, since we have instances of x belonging to each topic, here we explore two ways of deriving $\mathbf{t}_i$ for topic i: (1) We use the average of the vectors $\mathbf{x}$ belonging to topic i to form $\mathbf{t}_i$. We refer to this as **Avg**. (2) We use the parameters at the top layer of the topic classification model $p(t|x)$ as vector representations for the different topics. We refer to this as **Param**. During test time, since our instance x does not have a t associated with it, we use a topic classification model trained on the training data where each example has its correct topic labeled to estimate $p(t|x)$.

4 Experiments

4.1 Implementation Details

We follow the model architecture StA-PLAN in (Khoo et al., 2020) as our backbone. StA-PLAN is a hierarchical transformer which contains 12 post-level multi-head attention layer (MHA) and 2 token-level MHA layers. As claimed in (Khoo et al., 2020) that BERT (Devlin et al., 2018) did not improve results and was time-consuming, we apply GLOVE-300d (Pennington et al., 2014) to embed each token in a post. The initial learning rate was set as 0.01 with 0.3 dropout and we used the ADAM optimizer with 6000 warm start-up steps. Batch size is set as 256 for all cross-validation tasks.

4.2 Dataset

We use the public PHEME dataset (Zubiaga et al., 2016) for our evaluation. PHEME was collected based on 9 breaking news stories and can be categorised into four classes: true rumor, false rumour, unverified rumour and non-rumour. Following the setting in (Kumar and Carley, 2019), we select five breaking events from PHEME and split them into two sets. Four events are chosen for training and in-domain testing, and the remaining one is used as out-of-domain testing set.

4.3 Baselines and Our Methods

We consider a state-of-the-art model and some baselines that are also addressing cross-domain issues.

StA-HiTPLAN: Replicating (Khoo et al., 2020), we train a hierarchical transformer model which is a state-of-the art model and can be viewed as a feature extractor in the following experiments.

Ensemble-based model (EM): Following (He et al., 2019; Clark et al., 2019), we take topical words as bias features and introduce an auxiliary bias_only model f_b taking bias priori features as input. Then using this bias_only model to train a robust model through an ensemble model. We firstly obtain the class distribution $p_b(y|x)$ using this biased model. Then we train an ensemble model that combines the former biased model with a robust model through this function: $p(\hat{y}|x) = T(p(y|x) + p_b(y|x))$. In the testing stage, only the robust model $p(y|x)$ is used for prediction.

Adversary-based model (AM): This is a common way to learn domain-invariant features. We implement a recent work (Wang et al., 2018; Ma et al., 2019b) and replace their Bi-LSTM with (Khoo et al., 2020) as backbones for fair comparison. The parameter of the gradient reversal layer is set as 1.

MOE-Avg and **MOE-Param:** These are our proposed models, where MOE-Avg and MOE-Param are according to our descriptions given in Section 3.2.

4.4 Cross-Topic and In-Topic Settings

We use two settings to evaluate the effectiveness of our method for cross-topic rumor detection. The first setting is the standard setting where we train on a set of source topics and test the performance of the model on a different target topic. For PHEME dataset, we use 4 topics as training topics and the remaining topic as the test topic. We repeat this 5

Method	Orig.		Mask-20		Mask-50		Mask-100	
	Accuracy	Macro F	Accuracy	Macro F	Accuracy	Macro F	Accuracy	Macro F
StA-HiTPLAN	67.69	62.69	36.70	34.12	35.26	30.86	34.13	29.78
EM	72.31	72.77	38.46	32.28	37.65	31.92	37.58	31.62
AM	66.81	60.65	47.47	38.54	43.35	34.99	42.15	33.93
MOE-Avg	76.48	73.72	50.01	39.56	47.42	37.11	44.54	33.02
MOE-Param	76.54	73.85	50.23	39.74	47.51	37.46	44.82	33.17

Table 2: Average accuracy and macro-F scores (%) of the in-topic setting on PHEME. Orig. refers to the original data. Mask-k refers to the setting where we artificially mask k topic-specific words.

times with different split of training/test topics, and report the average performance. We refer to this as the "cross-topic" setting. We also experiment with a second in-topic setting, where we train and test on the same topic, but we artificially remove topic-specific words. We refer to this as our "in-topic" setting. In Table 2, these are labeled as Mask-20, Mask-30 and Mask-50, depending on how many topic-specific words we mask (i.e., remove).

4.5 Results and Analysis

We present our experiments on the PHEME dataset in Table 2 and Table 3. Several observations can be made from the experiment results:

1) From Table 3, we can see that MOE-Avg and MOE-Param are both effective strategies that mitigate the topic overfitting problem. The accuracy improves from 34.41% to 41.24% and 41.33%, respectively, when we only intervene feature without modify the backbone network. 2) Adversarial training model AM works better than ensemble methods EM in the early stage but deteriorates after we mask more than 50 event sensitive words. One reason is ensemble-based model depends on the bias only model : the model is sensitive to the choice of bias, and seems more robust when we mask more irrelevant words. 3) Instead of unstable adversarial training method, we show that MOE-Avg and MOE-Param can make the model robust to topic bias and increase generalization ability. 4) Instead of using the average of the vector representation of x for those x belonging to the same topic, we also aggregate the final layer parameters of topic classifier. MOE-Param works slightly better than MOE-Avg method. More attention can be given to how to better represent a topic embedding in future work.

Method	NON-MASK	
	Accuracy	Macro F
StA-HiTPLAN	34.41	32.69
EM	39.96	34.79
AM	38.03	34.32
MOE-Avg	41.24	36.84
MOE-Param	41.33	36.95

Table 3: Average accuracy and macro-F score (%) on PHEME data for the cross-topic setting.

5 Conclusion and Future work

In this work, we propose a new cross-topic rumor detection task base on mixture of experts, which can reinforce the generalization capacity of a model when adapting to new topics. we suggest that: 1) instead of training an unstable adversarial component or removing bias directly from semantic contents, the mixture of experts provides us with another way to increase generalization ability. 2) in this work, we use feature concatenation and train one classifier rather than several expert classifiers, and utilize a fixed confidence score. In the future, we can learn adaptive weights to make the model more flexible. For example, we could use variational inference methods to dynamically learn the best mixture of topics for a given held-out topic.

6 Acknowledgment

We thank the reviewers for their valuable comments. This research is supported by DSO grant DSOCL18009.

References

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. *arXiv preprint arXiv:1909.03683*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. *arXiv preprint arXiv:1908.10763*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Ling Min Serena Khoo, Hai Leong Chieu, Zhong Qian, and Jing Jiang. 2020. Interpretable rumor detection in microblogs by attending to user interactions. *arXiv preprint arXiv:2001.10667*.

Sumeet Kumar and Kathleen M Carley. 2019. Tree lstms with convolution units to predict stance and rumor veracity in social media conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5047–5058.

Yang Liu and Yi-Fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jing Ma, Wei Gao, Shafiq Joty, and Kam-Fai Wong. 2019a. Sentence-level evidence embedding for claim verification with hierarchical attention networks. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2019b. Detect rumors on twitter by promoting information campaigns with generative adversarial learning. In *The World Wide Web Conference*, pages 3049–3055.

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

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 849–857.

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3).

Understanding Pre-Editing for Black-Box Neural Machine Translation

Rei Miyata[†] **Atsushi Fujita**[‡]
[†]Nagoya University, Nagoya, Japan
[‡]National Institute of Information and Communications Technology, Kyoto, Japan
`miyata@nuee.nagoya-u.ac.jp`, `atsushi.fujita@nict.go.jp`

Abstract

Pre-editing is the process of modifying the source text (ST) so that it can be translated by machine translation (MT) in a better quality. Despite the unpredictability of black-box neural MT (NMT), pre-editing has been deployed in various practical MT use cases. Although many studies have demonstrated the effectiveness of pre-editing methods for particular settings, thus far, a deep understanding of what pre-editing is and how it works for black-box NMT is lacking. To elicit such understanding, we extensively investigated human pre-editing practices. We first implemented a protocol to incrementally record the minimum edits for each ST and collected 6,652 instances of pre-editing across three translation directions, two MT systems, and four text domains. We then analysed the instances from three perspectives: the characteristics of the pre-edited ST, the diversity of pre-editing operations, and the impact of the pre-editing operations on NMT outputs. Our findings include the following: (1) enhancing the explicitness of the meaning of an ST and its syntactic structure is more important for obtaining better translations than making the ST shorter and simpler, and (2) although the impact of pre-editing on NMT is generally unpredictable, there are some tendencies of changes in the NMT outputs depending on the editing operation types.

1 Introduction

Recent advances in machine translation (MT) have greatly facilitated its practical use in various settings from business documentation to personal communication. In many practical cases, MT systems are used as black-box and one well-tested approach to make use of a black-box MT is pre-editing, i.e., modifying the source text (ST) to make it suitable for the intended MT system.

The effectiveness of pre-editing has so far been demonstrated in many studies (Pym, 1990; O'Brien and Roturier, 2007; Seretan et al., 2014). A study focusing on statistical MT (SMT) has also shown that more than 90% of an ST can be rewritten into a text that can be machine-translated with sufficient quality (Miyata and Fujita, 2017), exhibiting the potential of the pre-editing approach.

However, the feasibility and possibility of pre-editing for neural MT (NMT) has not been examined extensively. While efforts have recently been invested in the implementation of pre-editing strategies for black-box NMT settings, achieving improved MT quality (e.g., Hiraoka and Yamada, 2019; Mehta et al., 2020), the potential gains of pre-editing remain unexplored. Notably, the impact of pre-editing on black-box MT is unpredictable in nature. In particular, NMT models trained in an end-to-end manner can be sensitive to minor modifications of the ST (Cheng et al., 2019), which may affect the feasibility of pre-editing.

In short, while pre-editing has been implemented in practical MT use cases, what pre-editing is and how it works with black-box NMT systems remain open questions. To explore the possibility of pre-editing and its automation, in this study, we provide fine-grained analyses of human pre-editing practices and their impact on NMT. We systematically collected pre-editing instances in various conditions, i.e., translation directions, NMT systems, and text domains (§3). We then conducted in-depth analyses of the collected instances from the following three perspectives: the characteristics of the pre-edited ST (§4), the diversity of pre-editing operations (§5), and the impact of pre-editing operations on the NMT outputs (§6). The findings of these analyses provide useful insights into the effective and efficient implementation of pre-editing for the better use of black-box NMT systems in the future, as well as the robustness of current NMT systems when STs are manually perturbed.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1539–1550
April 19 - 23, 2021.

2 Related Work

Pre-editing is the process of rewriting the source text (ST) to be translated in order to obtain better translations by MT. Though the scope of effective pre-editing operations depends on the downstream MT system and there is no deterministic relation between pre-editing operations and the quality of MT output, its effectiveness has been demonstrated for various translation directions, MT architectures, and text domains.

Manual pre-editing has long been implemented in combination with controlled languages (Pym, 1990; Reuther, 2003; Nyberg et al., 2003; Kuhn, 2014). In the period of rule-based MT (RBMT), pre-editing was considered as a promising approach since the behaviour of RBMT is more predictable and controllable. For example, O'Brien and Roturier (2007) examined the impact of English controlled language rules on two different MT engines, revealing the rules of high effectiveness. The pre-editing approach with controlled languages has also been tested for statistical MT (SMT) (Aikawa et al., 2007; Hartley et al., 2012; Seretan et al., 2014). These studies developed or utilised a set of controlled language rules for rewriting ST. While these rule sets are optimised for particular MT systems and differ from each other, we can observe some shared characteristics among them. In particular, rules that prohibit long sentences (e.g., of more than 25 words) are widely adopted in the existing rule sets (O'Brien, 2003).

Automation of pre-editing is also an important research field in natural language processing. Semi-automatic tools such as controlled language checkers (Bernth and Gdaniec, 2001; Mitamura et al., 2003) and interactive rewriting assistants (Mirkin et al., 2013; Gulati et al., 2015) were developed to facilitate manual pre-editing activities. Fully automatic pre-editing has long been explored (e.g., Shirai et al., 1998; Mitamura and Nyberg, 2001; Yoshimi, 2001; Sun et al., 2010). In particular, many researchers have examined methods of *reordering* the source-side word order as a pre-translation processing (Xia and McCord, 2004; Li et al., 2007; Hoshino et al., 2015). While the reordering approach has generally proven effective for SMT, its effectiveness for NMT is not obvious; negative effects have even be reported (Zhu, 2015; Du and Way, 2017). In recent years, techniques of automatic text simplification have been applied to improve NMT outputs (Štajner and Popović, 2018; Mehta et al., 2020). The underlying assumption of these studies is that simpler sentences are more machine translatable.

Previous studies have investigated various pre-editing methods from different perspectives, focusing on different linguistic phenomena. Indeed, individual research has led to improved MT results. However, what is crucially needed is a broad understanding of what pre-editing is and how it works. For example, Miyata and Fujita (2017) addressed this issue by collecting instances of *bilingual pre-editing*, i.e., pre-editing ST while referring to its MT output, done by human editors and analysing them in detail. They demonstrated the maximum gain of pre-editing for an SMT and provided a comprehensive typology of editing operations. Nevertheless, their study has two major limitations: (1) recent NMT was not examined, and (2) practical insights for better practices of pre-editing were not sufficiently presented.

NMT models trained in an end-to-end manner behave very differently from SMT and RBMT, which, in turn, affects pre-editing practices. As reported in several studies, despite their rapid improvement, NMT models are still vulnerable to input noise (Belinkov and Bisk, 2018; Ebrahimi et al., 2018; Cheng et al., 2019; Niu et al., 2020). The pre-editing operations identified in previous studies are not necessarily effective for current black-box NMT systems.[1] For example, Marzouk and Hansen-Schirra (2019) adopted nine controlled language rules[2] and evaluated their impact on the MT output for German-to-English translation in the technical domain. The human evaluation results revealed that these rules improved the performance of the RBMT, SMT, and hybrid systems, but did not have positive effects on the NMT system. Hiraoka and Yamada (2019) demonstrated the effectiveness of the following three pre-editing rules in improving Japanese-to-English TED Talk subtitle

---

[1] The ideal goal of the pre-editing approach is to adapt the STs to what the intended NMT system can properly translate, and in the end, what it has been trained on, i.e., training data. For a black-box MT system, because we cannot directly refer to its training data, we should grasp its statistical characteristics indirectly through MT output.

[2] The rules are as follows: (1) using straight quotes for interface texts, (2) avoiding light-verb construction, (3) formulating conditions as if sentences, (4) using unambiguous pronominal references, (5) avoiding participial constructions, (6) avoiding passives, (7) avoiding constructions with "sein" + "zu" + infinitive, (8) avoiding superfluous prefixes, and (9) avoiding omitting parts of the words (Marzouk and Hansen-Schirra, 2019, p.184).

5. Perfect	Information in the original text has been completely translated. There are no grammatical errors in the translation. The word choice and phrasing are natural even from a native speaker's point of view.
4. Good	The word choice and phrasing are slightly unnatural, but the information in the original text has been completely translated, and there are no grammatical errors in the translation.
3. Fair	There are some minor errors in the translation of less important information in the original text, but the meaning of the original text can be easily understood.
2. Acceptable	Important parts of the original text are omitted or incorrectly translated, but the core meaning of the original text can still be understood with some effort.
1. Incorrect/nonsense	The meaning of the original text is incomprehensible.

Table 1: MT evaluation criterion adopted in Miyata and Fujita (2017): The "Perfect" and "Good" ratings are regarded as satisfactory quality.

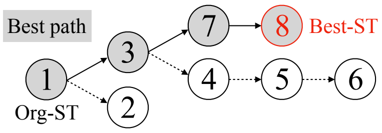


Figure 1: Tree representation of ST versions in a unit.

Name	Domain	Mode	Size	Avg. length (S.D.)
hospital	hospital conversation	spoken	25	13.0 (4.7)
municipal	municipal procedure	written	25	20.4 (10.7)
bccwj	Japanese-origin news article from BCCWJ	written	25	28.6 (18.6)
reuters	English-origin news article from Reuters	written	25	36.8 (15.3)

Table 2: Statistics for the Org-ST datasets for pre-editing.

translation using a black-box NMT system: (1) inserting punctuation, (2) making implied subjects and objects explicit, and (3) writing proper nouns in the target language (English).

As these studies cover a limited range of linguistic phenomena, translation directions, and text domains, we are not in the position to draw decisive conclusions; we still do not know what types of pre-editing operations are possible and how NMT is affected when these operations are performed. To elicit the best pre-editing practices for NMT, as a starting point, we need to understand what is happening and what can be obtained in the process of pre-editing, while also re-examining the previous findings and conventional methods.

3 Collection of Pre-Editing Instances

3.1 Protocol

To collect fine-grained manual pre-editing instances, we adopted the protocol formalised by Miyata and Fujita (2017), in which a human editor incrementally and minimally rewrites an ST on a trial-and-error basis with the aim of obtaining better MT output. An original ST (Org-ST) and its pre-edited versions are collectively called a *unit*. Using an online editing platform we developed, editors implement the protocol in the following steps:

Step 1. Evaluate the MT output of the current ST based on a 5-point scale criterion shown in Table 1. If the quality of the MT output is satisfactory (i.e., "Perfect" or "Good"), go to Step 4; otherwise, go to Step 2.

Step 2. Select one of the versions of the ST in the unit to be rewritten and go to Step 3. If none of the versions are likely to become satisfactory through further edits, go to Step 4.

Step 3. Minimally edit the ST[3] while maintaining its meaning, referring to the corresponding MT output. The MT output for the edited ST is automatically generated and registered in the unit. Return to Step 1.

Step 4. Select one version of the ST that achieves the best MT quality (Best-ST) from among all the versions in the unit, and terminate the process for the unit.

The pre-editing instances in a unit collected through this protocol form a tree structure as shown in Figure 1. We refer to the shortest path between the Org-ST and the Best-ST as *Best path*. An important extension to the work in Miyata and Fujita (2017) is that our platform provides editors with a visualisation of the tree representation of the pre-editing history. This can facilitate the selection of ST versions in Step 2.

3.2 Implementation

To extensively investigate pre-editing phenomena, we prepared the following conditions:

Translation directions: We targeted Japanese-to-English (Ja-En), Japanese-to-Chinese (Ja-Zh), and Japanese-to-Korean (Ja-Ko) translations.

---

[3] We operationally defined "to minimally edit" as "to modify an ST with a small edit that is difficult to be further decomposed into more than one independent edit, without inducing ungrammaticality in the edited sentence."

Lang.	**System**	**Domain**	**Num. of pre-editing instances**				**Num. of units**	
			Total	**Avg.**	**Med.**	**Max**	**Org=Satisfactory**	**Best=Satisfactory**
Ja-En	**Google**	**hospital**	255	10.2	7	55	4/25	**25/25**
		municipal	162	6.5	5	44	9/25	**25/25**
		bccwj	545	21.8	10.5	171	7/25	23/25
		reuters	370	14.8	6.5	80	7/25	**25/25**
	TexTra	**hospital**	139	5.6	5.5	25	7/25	**25/25**
		municipal	136	5.4	4	35	10/25	**25/25**
		bccwj	493	19.7	11.5	79	2/25	22/25
		reuters	492	19.7	18	86	4/25	24/25
Ja-Zh	**Google**	**hospital**	264	10.6	10	30	0/25	24/25
		municipal	376	15.0	13	41	0/25	23/25
		bccwj	427	17.1	16	41	2/25	20/25
		reuters	304	12.2	10	27	0/25	24/25
	TexTra	**hospital**	160	6.4	6.5	15	1/25	**25/25**
		municipal	172	6.9	7	20	2/25	**25/25**
		bccwj	231	9.2	5	38	4/25	22/25
		reuters	249	10.0	7	31	1/25	22/25
Ja-Ko	**Google**	**hospital**	209	8.4	9	22	0/25	**25/25**
		municipal	225	9.0	8	26	0/25	**25/25**
		bccwj	223	8.9	7	27	1/25	22/25
		reuters	293	11.7	10	33	0/25	24/25
	TexTra	**hospital**	160	6.4	6	26	2/25	**25/25**
		municipal	171	6.8	5	32	2/25	**25/25**
		bccwj	277	11.1	6	28	3/25	23/25
		reuters	319	12.8	11	38	1/25	23/25

Table 3: Statistics for the collected pre-editing instances and the MT quality achievement.

MT systems: As black-box MT systems, we adopted Google Translate[4] and TexTra.[5] Both are general-purpose NMT systems that are prevalently used for translating Japanese texts into other languages.

Text domains: We selected four text domains, whose linguistic characteristics, such as mode and sentence length, are different from each other (see Table 2 for details).

We randomly selected 25 Japanese sentences for each of the four text domains, and used the resulting ST set consisting of 100 sentences for all of the six combinations of translation direction and MT system. We assigned one editor to each translation direction. Each editor was asked to work with both MT systems, without being informed of the type of MT system used in the task. All editors were professional translators with sufficient writing skills in Japanese and experience for evaluating MT outputs. Before the commencement of the formal tasks, we trained the editors using example sentences so that they could become accustomed to the task and platform.

The Ja-En task was implemented from November to December 2019; the Ja-Zh and Ja-Ko tasks were implemented from December 2019 to February 2020.

3.3 Statistics

Table 3 shows statistics for the pre-editing instances collected through the protocol described above. In general, the numbers of collected instances for the **hospital** and **municipal** domains were smaller than those for the **bccwj** and **reuters** domains, reflecting the influence of sentence length of the Org-ST. In other words, the shorter the sentence is, the fewer parts there are to be edited.

A notable finding is that while only about 11% (69/600) of the MT output for the Org-ST was of satisfactory quality, 95% (571/600) of the MT output of the Best-ST was satisfactory. This means that almost all the ST can be pre-edited into a form that can lead to satisfactory MT output, demonstrating the potential of both pre-editing and NMT.

The number of collected instances can be interpreted as the editing efforts required to obtain the Best-ST from the Org-ST. In most of the settings, the median number of collected instances for a unit falls in the range of 5 to 10. It is thus necessary to optimise the pre-editing process for an intended MT system. The length of the Best path approximates the minimum editing efforts needed to obtain the Best-ST. The total number of pre-editing instances in the Best path was 2,443, while the total of all instances is 6,652. This implies that there is substantial opportunity for reduction of the pre-editing efforts.

---

[4] https://translate.google.com/

[5] https://textra.nict.go.jp/

		Org-ST	Ja-En (Best-ST)		Ja-Zh (Best-ST)		Ja-Ko (Best-ST)	
			Google	TexTra	Google	TexTra	Google	TexTra
Sentence length	**Avg.**	25.4	27.8	26.9	28.6	27.1	27.8	26.9
	S.D.	16.3	17.6	16.7	17.2	16.0	16.7	16.6
	Med.	19.5	21.5	20	23	22	22.5	20.5
Attachment distance (Avg. per sentence)	**Avg.**	1.95	1.97	1.99	1.99	1.99	2.00	1.98
	S.D.	0.65	0.53	0.65	0.60	0.63	0.64	0.62
	Med.	1.83	2.00	1.96	2.00	1.98	2.00	1.91
Dependency depth	**Avg.**	3.57	3.73	3.68	3.73	3.77	3.78	3.76
	S.D.	1.91	1.97	1.88	1.89	1.93	1.85	1.92
	Med.	3	3	3	3	4	4	4
Lexical diversity	**Token (A)**	2,538	2,779	2,685	2,861	2,709	2,780	2,693
	Type (B)	1,010	1,074	1,060	1,106	1,061	1,068	1,055
	A/B	2.513	2.588	2.533	2.587	2.553	2.603	2.553
Word frequency rank (Percentile)	**25th**	7	7	7	7	7	7	7
	50th (Med.)	170	143	154	143	155	143	169.5
	75th	2655	2304.25	2458	2471	2554	2470	2593.5

Table 4: Linguistic characteristics of the Org-ST and Best-ST.

4 Characteristics of Pre-Edited Sentences

To understand the differences between the original and pre-edited STs, in this section, we describe their general linguistic characteristics. Here, we compare the Org-ST and the Best-ST that achieved a satisfactory MT result in order to elicit the features of machine translatable ST.

4.1 Structural Characteristics

To quantify structural complexity, we used the following three indices:

(1) sentence length: the number of words per sentence[6]
(2) attachment distance: the averaged distance of all attachment pairs of the Japanese base phrases in a sentence
(3) dependency depth: the maximum distance from the root word in the dependency tree

We used the Japanese tokeniser MeCab[7] to calculate (1) and the Japanese dependency parser JUMAN/KNP[8] to calculate (2) and (3).

The first three blocks in Table 4 show the results for these indices. It is evident on all indices, the Org-ST exhibits the lowest scores. In other words, the length and surface complexity of the sentences generally increased through the pre-editing operations. This is a counter-intuitive finding in that most previous pre-editing practices have axiomatically assumed that shorter and less complex sentences are better for MT. We further delve into this in §5.

4.2 Lexical Characteristics

The remaining two blocks in Table 4 present statistics for the lexical characteristics of the STs. The results for lexical diversity indicate that both the total number of word types and the Token/Type ratio increased from the Org-ST to the Best-ST for all the conditions. This suggests that though the diversity of words increased slightly, the word distribution became peakier through pre-editing.

We also calculated the word frequency rank with Wikipedia as the reference.[9] To assess the status of word frequency in relation to MT, it would be ideal to use the training data for each MT system, but such data are unavailable in black-box MT settings. Therefore, we decided to use Wikipedia as a convenient way to observe general word frequency. Lower numbers indicate higher word frequencies in Wikipedia. The 50th and 75th percentile values in the datasets imply that pre-editing induced the avoidance of low-frequency words.

To further inspect the differences between the Org-ST and the Best-ST, we extracted the word types (a) that appeared only in the Org-ST and (b) that appeared only in the Best-ST. Figure 2 illustrates the rank distributions of (a) and (b) for each condition. It is clear that low-frequency words with a frequency rank of around 10,000 decreased in the Best-ST, while words with a frequency rank of around 2,000–4,000 increased in the Best-ST. As Koehn and Knowles (2017) demonstrated, low-frequency words still pose major obstacles for NMT systems. Our results endorse this claim from a different perspective and can provide general strategies for word choice in the pre-editing task.

---

[6] If ST instance includes multiple sentences, we averaged the scores.

[7] https://taku910.github.io/mecab/

[8] http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP

[9] We used the whole text data of Japanese Wikipedia obtained in October 2019 (https://dumps.wikimedia.org/).

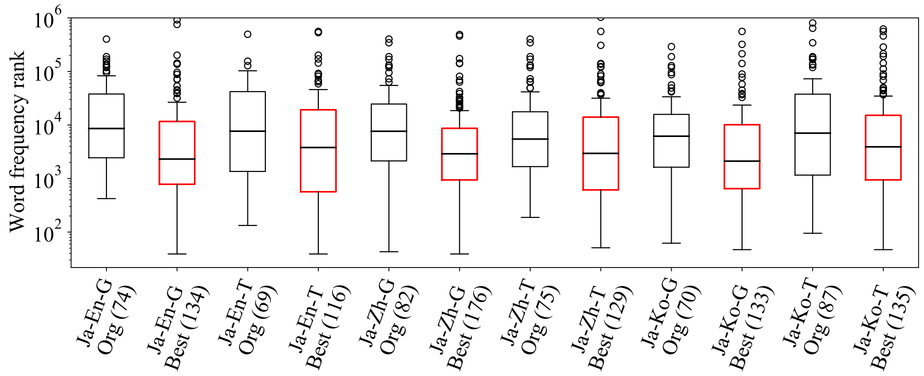


Figure 2: Differences in word frequency rank distribution between the Org-ST and Best-ST (G: Google, T: TexTra). The numbers in parentheses indicate the number of instances, i.e., word types.

5 Diversity of Pre-Editing Operations

5.1 Typology of Edit Operations

To understand the diversity of edit operations for pre-editing, we manually annotated the collected pre-editing instances in terms of linguistic operations. Given that the Best path contains effective editing operations for improved MT quality, we focused on the pairs of ST versions in the Best path (e.g., the pairs {1→3, 3→7, 7→8} in Figure 1). We randomly selected 10 units for each of the 24 combinations of translation direction, MT system, and text domain, resulting in a total of 961 pre-editing instances. We then excluded 26 instances that could be decomposed into multiple smaller edits[10] and classified the remaining 935 instances, each of which consists of a minimum edit of ST, based on the typology proposed by Miyata and Fujita (2017). Through the classification, we refined the existing typology to consistently accommodate all the instances.

Table 5 presents our typology of editing operations with the number of instances in the different conditions. The typology consists of 39 operation types under 6 major categories, which enables us to grasp the diversity and trends of pre-editing operations. Compared to structural editing, local modifications of words and phrases were frequently used in the Best path. The dominant type is **C01 (Use of synonymous words)**: content words are replaced by another synonymous word. This operation is important for achieving appropriate word choice in the MT output. **C07 (Change of content)**, the second dominant type, includes the addition of information that is inferred by human editors based on the intra-sentential context or even external knowledge. For example, a named entity '*Nemuro-sho*' (Nemuro office) was changed into '*Nemuro-keisatsu-sho*' (Nemuro police office) by using the knowledge of the entity. It might be challenging to automate such *creative* operations.

It is also notable that **S01 (Sentence splitting)** only amounts to 1.5% of all instances, which supports the observation in §4.1 that in general, sentence length was not reduced, and even increased by pre-editing. Among the 14 cases of this type, nine of the split sentences were 60–67 words in length. These results support the empirical observation by Koehn and Knowles (2017) that NMT systems still have difficulty in translating sentences longer than 60 words, and suggest that sentence splitting may only be promising for such very long sentences.

5.2 Strategies for Effective Pre-Editing

Towards the effective exercise of pre-editing, we further analysed the pre-editing instances in terms of *informational strategies* based on the notion of explicitation/implicitation acknowledged in translation studies (Vinay and Darbelnet, 1958; Chesterman, 1997; Murtisari, 2016). Following these studies, we broadly defined explicitation as an act of indicating what is implied in the text to clarify its meaning and implicitation as the inverse act of explicitation. We classified all the instances analysed above except for the **E01** and **E02** types into three general strategies, namely, explicitation, implicitation, and (information) preservation. The right side of Table 5 shows the classification result. The total numbers of instances classified into each strategy were 329, 88, and 480, respectively. Not surprisingly, this indicates that explicitation is an essential

---

[10] Only 2.7% of the edits were not regarded as minimum, which demonstrated satisfactory adherence to our instructions, compared with the implementation by Miyata and Fujita (2017), in which 568 pre-editing instances were finally decomposed into 979 instances.

ID	Editing operation type	Ja-En		Ja-Zh		Ja-Ko		Total	Expl.	Impl.	Pres.
		G	T	G	T	G	T				
S01	Sentence splitting	1	0	3	3	4	3	14	0	0	14
S02	Structural change	3	5	9	4	4	2	27	8	1	18
S03	Use/disuse of topicalisation	1	7	4	3	1	3	19	5	2	12
S04	Insertion of subject/object	2	1	1	3	5	2	14	14	0	0
S05	Use/disuse of clause-ending noun	3	2	2	2	2	1	12	12	0	0
S06	Change of voice	1	3	0	0	0	0	4	2	0	2
S07	Other structural changes	1	0	2	1	1	0	5	3	0	2
P01	Insertion/deletion of punctuation	19	16	5	12	9	10	71	0	0	71
P02	Use/disuse of chunking marker(s)	6	12	2	1	3	4	28	11	8	9
P03	Phrase reordering	6	4	7	1	9	4	31	0	0	31
P04	Change of modification	1	3	3	0	0	0	7	0	0	7
P05	Change of connective expression	3	18	4	2	10	3	40	24	5	11
P06	Change of parallel expression	3	8	2	8	4	11	36	7	2	27
P07	Change of apposition expression	1	7	2	1	1	4	16	8	4	4
P08	Change of noun/verb phrase	1	3	2	1	3	3	13	9	3	1
P09	Use/disuse of compound noun	1	5	2	2	6	12	28	16	12	0
P10	Use/disuse of affix	4	4	1	2	3	3	17	1	0	16
P11	Change of sahen noun expression	0	1	1	1	2	0	5	1	0	4
P12	Change of formal noun expression	1	2	2	2	2	0	9	4	0	5
P13	Other phrasal changes	0	1	0	1	2	1	5	4	0	1
C01	Use of synonymous words	18	18	19	18	25	20	118	14	10	94
C02	Use/disuse of abbreviation	2	7	2	2	1	7	21	19	2	0
C03	Use/disuse of anaphoric expression	4	4	2	2	1	1	14	10	2	2
C04	Use/disuse of emphatic expression	1	2	2	1	4	1	11	10	1	0
C05	Category indication/suppression	5	3	6	5	4	7	30	29	1	0
C06	Explanatory paraphrase	3	4	1	0	1	1	10	0	0	10
C07	Change of content	22	20	21	9	14	8	94	57	23	14
F01	Change of particle	9	14	4	6	7	7	47	13	5	29
F02	Change of compound particle	8	5	5	2	5	6	31	24	2	5
F03	Change of aspect	1	4	1	0	5	1	12	0	0	12
F04	Change of tense	0	0	1	1	1	1	4	0	0	4
F05	Change of modality	3	1	2	1	3	1	11	5	0	6
F06	Use/disuse of honorific expression	3	1	1	2	2	1	10	0	0	10
O01	Japanese orthographical change	10	16	9	5	9	12	61	12	4	45
O02	Change of half-/full-width character	0	5	3	2	2	4	16	7	1	8
O03	Insertion/deletion/change of symbol	0	2	0	0	0	0	2	0	0	2
O04	Other orthographical change	0	1	0	0	3	0	4	0	0	4
E01	Grammatical errors	0	8	5	2	2	5	22	–	–	–
E02	Content errors	5	0	8	1	1	1	16	–	–	–

Table 5: Constructed typology of editing operations (G: Google, T: TexTra). The first letter of ID indicates the six major categories (S: Structure, P: Phrase, C: Content word, F: Functional word, O: Orthography, E: Errors casually introduced in the ST). The right three columns provide the frequencies for general informational strategies (Expl.: Explicitation, Impl.: Implicitation, Pres.: Preservation).

strategy for effective pre-editing.

We also grouped all the 329 instances of explicitation into the following four subcategories.[11]

Information addition is the strategy of adding supplementary information, such as subjects, modality, and explanation, to clarify the content of the ST. For example, subjects were sometimes inserted as they tend to be omitted in Japanese sentences. This strategy generally corresponds to operation **C07 (Change of content)** described earlier.

Use of clear relation includes structural changes and the use of explicit connective markers to make the relation between words, phrases, and clauses more intelligible. For example, the relation between the subject and object can be clarified by using the nominative case marker '*ga*' in Japanese.

Use of narrower sense is the strategy of replacing general words with more specific ones. For example, the verb '*dasu*,' which has multiple meanings such as 'put,' 'take,' and 'send,' was replaced with the verb '*teishutsusuru*,' which has a narrower range of meaning and was correctly translated as 'submit.'

Normalisation includes the use of authorised or standardised expressions, style, and notation. For example, elliptic sentence-ending was completed to construct a normal structure.

---

[11] See Appendix A for details.

		Ja-En		Ja-Zh		Ja-Ko	
		Google	**TexTra**	**Google**	**TexTra**	**Google**	**TexTra**
TER	**Pearson's r**	0.244	0.217	0.144	0.204	0.580	0.347
	Spearman's ρ	0.218	0.184	0.094	0.172	0.574	0.248
Num. of edits	**Pearson's r**	0.264	0.153	0.205	0.181	0.465	0.212
	Spearman's ρ	0.210	0.221	0.219	0.226	0.449	0.245

Table 6: Correlation of the TER and the number of edits between ST and MT.

These strategies can be used as concise pre-editing principles for human editors and can guide researchers in devising effective tools for pre-editing. We also emphasise that these general informational strategies are not specific to the Japanese language and could be applied to other languages.

6 Impact of Pre-Editing on Neural Machine Translation

This section investigates how pre-editing operations affect the NMT output. As indicated in §2, NMT systems still lack robustness, and minor modifications of the input would drastically change the output. From the practical viewpoint of deploying pre-editing, predictability is an important object to pursue. Here, we examine the impacts of minimum edits of the ST on the NMT output. To measure the amount of text editing, hereafter, we use the Translation Edit Rate (TER), which is calculated by dividing the number of edits (insertion, deletion, substitution, and shift) required to change a string into the reference string by the average number of reference words (Snover et al., 2006). For any consecutive pair of STs or their corresponding MT outputs, we used the chronologically later version as the reference. For word-level tokenisation, we used MeCab for Japanese, NLTK[12] for English, jieba[13] for Chinese, and KoNLPy[14] for Korean.

6.1 Correlation of the Amount of Edits between the ST and MT

To grasp the general tendency, using all the collected pre-editing instances (see Table 3), we first calculated the correlation coefficients (Pearson's r and Spearman's ρ) between the amount of edits (the TER and the number of edits) in the ST and in the MT. More formally, let $\mathtt{ST}'$ be the pre-edited versions of $\mathtt{ST}$. For TER, the correlation is between $\mathtt{TER}(\mathtt{ST}, \mathtt{ST}')$ and $\mathtt{TER}(\mathtt{MT}(\mathtt{ST}), \mathtt{MT}(\mathtt{ST}'))$. For the number of edits, the correlation is between $\mathtt{EditCount}(\mathtt{ST}, \mathtt{ST}')$ and $\mathtt{EditCount}(\mathtt{MT}(\mathtt{ST}), \mathtt{MT}(\mathtt{ST}'))$.

As shown in Table 6, most coefficients are in the range of 0.15–0.25, suggesting a very weak correlation. This means that the change in NMT output is hardly predictable based on the amount of edits in the ST. For example, the replacement of a single particle in the ST sometimes caused drastic changes of lexical choices in the MT output.

The Japanese-to-Korean translation is an exception; in particular, the correlation coefficients of the TER for the Google NMT system, i.e., 0.580 for Pearson's r and 0.574 for Spearman's ρ, indicate a moderate positive relationship between the changes in the ST and those in the MT. This is partly attributable to the fact that the syntactic structures of Japanese and Korean, including the word order and usage of particles, are substantially close. Thus, it is relatively easy to build sufficiently accurate MT systems.

6.2 Impact of Editing Operations on NMT

Finally, using the pre-editing instances in the Best path analysed in §5, we further investigated to what extent each type of minimum editing operation affects the MT output. At this stage, we focused on the 28 editing types that have at least 10 instances, considering that it is difficult to derive reliable insights from fewer data.

Figure 3 presents the distribution of the degree of changes in the MT output when an ST is pre-edited, measured by $\mathtt{TER}(\mathtt{MT}(\mathtt{ST}), \mathtt{MT}(\mathtt{ST}'))$. Most of the structural edits (**S01–S04**) resulted in sizeable changes in the MT. This is reasonable since structural modifications in the ST tended to cause major changes in the MT as well, leading to high TER. In contrast, many of the editing types that include local modifications of functional words and orthographic notations (**F01–F03, F05, F06, O01, O02**) did not have major impacts on the MT results.

It is worth noticing that **P03 (Phrase reordering)** did not drastically affect the MT output. In other words, recent NMT systems in practical use manage to retain the phrase-level equivalence even

---

[12] https://www.nltk.org/index.html

[13] https://github.com/fxsjy/jieba

[14] https://konlpy.org/en/latest/api/konlpy.tag/#module-konlpy.tag._kkma

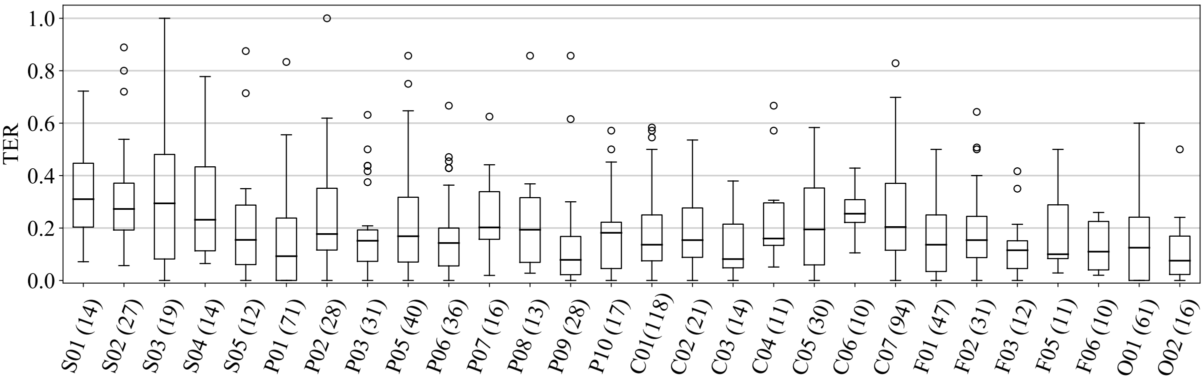


Figure 3: Distribution of the TER for changes in the MT for each operation type with at least 10 instances. The numbers in parentheses indicate the number of instances.

when the position of a phrase is shifted. The influence of **P02 (Use/disuse of chunking marker(s))** is fairly significant. For human readers, the use of chunking markers, such as double quotes and square brackets, does not greatly affect the sentence parsing, but for NMT, it might seriously impinge on the tokenisation result, eventually leading to a large change in the final output.

7 Conclusion and Outlook

Towards a better understanding of pre-editing for black-box NMT settings, in this study, we collected instances of manual pre-editing in various conditions and conducted in-depth analyses of the instances. We implemented a human-in-the-loop protocol to incrementally record minimum edits of ST for all combinations of three translation directions, two NMT systems, and four text domains, and obtained a total of 6,652 instances of manual pre-editing. Since more than 95% of the STs were successfully pre-edited into one that led to a satisfactory MT quality, our collected instances contain empirical, tacit human knowledge on the effective use of black-box NMT systems. We also investigated the collected data from three perspectives: the characteristics of the pre-edited STs, the diversity of pre-editing operations, and the impact of pre-editing operations on the NMT output. The remarkable findings can be summarised as follows:

- Contrary to the acknowledged practices of pre-editing, the operation of making source sentences shorter and simpler was not frequently observed. Rather, it is more important to make the content, syntactic relations, and word senses clearer and more explicit, even if the ST becomes longer.
- As indicated by recent studies, the NMT systems are still sensitive to minor edits in the ST, and are unpredictable in general. However, there are recognisable tendencies in the MT output according to the types of editing operations, such as the relatively small impact of phrase reordering on NMT.

In future work, we plan to explore the effective implementation of pre-editing. The findings of this study provide a broad overview of the range of pre-editing operations and their expected benefits, which enables us to find feasible pre-editing solutions in practical use cases of black-box NMT systems. To develop automatic pre-editing tools using a collection of pre-editing instances, we need to handle the data insufficiency issue in machine learning, filling the gap between the training data and targeted black-box MT systems.

Moreover, as our pre-editing instances contain a wide variety of perturbations in the ST, they can also be used to evaluate the robustness of MT systems, which can lead to advances in MT research. We aim to jointly improve the two wheels of translation technology: pre-editing and MT.

Acknowledgments

This work was partly supported by JSPS KAKENHI Grant Numbers 19K20628 and 19H05660, and the Research Grant Program of KDDI Foundation, Japan. One of the corpora used in our study was created under a program “Research and Development of Enhanced Multilingual and Multipurpose Speech Translation System” of the Ministry of Internal Affairs and Communications, Japan.

References

Takako Aikawa, Lee Schwartz, Ronit King, Monica Corston-Oliver, and Carmen Lozano. 2007. Impact of controlled language on translation quality and post-editing in a statistical machine translation environment. In *Proceedings of the Machine Translation Summit XI*, pages 1–7, Copenhagen, Denmark.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, pages 1–13, Vancouver, Canada.

Arendse Bernth and Claudia Gdaniec. 2001. MTranslatability. *Machine Translation*, 16(3):175–218.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4324–4333. Florence, Italy.

Andrew Chesterman. 1997. *Memes of Translation*. John Benjamins, Amsterdam.

Jinhua Du and Andy Way. 2017. Pre-reordering for neural machine translation: Helpful or harmful? *The Prague Bulletin of Mathematical Linguistics*, 108:171–182.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, pages 653–663, Santa Fe, New Mexico, USA.

Asheesh Gulati, Pierrette Bouillon, Johanna Gerlach, Victoria Porro, and Violeta Seretan. 2015. The ACCEPT Academic Portal: A user-centred online platform for pre-editing and post-editing. In *Proceedings of the 7th International Conference of the Iberian Association of Translation and Interpreting Studies (AIETI)*, Malaga, Spain.

Anthony Hartley, Midori Tatsumi, Hitoshi Isahara, Kyo Kageura, and Rei Miyata. 2012. Readability and translatability judgments for 'Controlled Japanese'. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 237–244, Trento, Italy.

Yusuke Hiraoka and Masaru Yamada. 2019. Pre-editing plus neural machine translation for subtitling: Effective pre-editing rules for subtitling of TED talks. In *Proceedings of the Machine Translation Summit XVII*, pages 64–72, Dublin, Ireland.

Sho Hoshino, Yusuke Miyao, Katsuhito Sudoh, Katsuhiko Hayashi, and Masaaki Nagata. 2015. Discriminative preordering meets Kendall's τ maximization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 139–144, Beijing, China.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the 1st Workshop on Neural Machine Translation (NMT)*, pages 28–39, Vancouver, Canada.

Tobias Kuhn. 2014. A survey and classification of controlled natural languages. *Computational Linguistics*, 40(1):121–170.

Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proceedings of the 45th Annual Meeting on Association for Computational Linguistics (ACL)*, pages 720–727, Prague, Czech Republic.

Shaimaa Marzouk and Silvia Hansen-Schirra. 2019. Evaluation of the impact of controlled language on neural machine translation compared to other MT architectures. *Machine Translation*, 33(1-2):179–203.

Sneha Mehta, Bahareh Azarnoush, Boris Chen, Avneesh Saluja, Vinith Misra, Ballav Bihani, and Ritwik Kumar. 2020. Simplify-then-translate: Automatic preprocessing for black-box translation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, pages 8488–8495, New York, USA.

Shachar Mirkin, Sriram Venkatapathy, Marc Dymetman, and Ioan Calapodescu. 2013. SORT: An interactive source-rewriting tool for improved translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*, pages 85–90, Sofia, Bulgaria.

Teruko Mitamura, Kathryn L. Baker, Eric Nyberg, and David Svoboda. 2003. Diagnostics for interactive controlled language checking. In *Proceedings of the Joint Conference Combining the 8th International Workshop of the European Association for Machine Translation and the 4th Controlled Language Applications Workshop (EAMT/CLAW)*, pages 237–244, Dublin, Ireland.

Teruko Mitamura and Eric Nyberg. 2001. Automatic rewriting for controlled language translation. In *Proceedings of the NLPRS2001 Workshop on Automatic Paraphrasing: Theories and Applications*, pages 1–12, Tokyo, Japan.

Rei Miyata and Atsushi Fujita. 2017. Dissecting human pre-editing toward better use of off-the-shelf machine translation systems. In *Proceedings of the 20th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 54–59, Prague, Czech Republic.

Elisabet Titik Murtisari. 2016. Explicitation in Translation Studies: The journey of an elusive concept. *The International Journal for Translation & Interpreting Research*, 8(2):64–81.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8538–8544, Online.

Eric Nyberg, Teruko Mitamura, and Willem-Olaf Huijsen. 2003. Controlled language for authoring and translation. In Harold Somers, editor, *Computers and Translation: A Translator's Guide*, pages 245–281. John Benjamins, Amsterdam.

Sharon O'Brien. 2003. Controlling controlled English: An analysis of several controlled language rule sets. In *Proceedings of the Joint Conference Combining the 8th International Workshop of the European Association for Machine Translation and the 4th Controlled Language Applications Workshop (EAMT/CLAW)*, pages 105–114, Dublin, Ireland.

Sharon O'Brien and Johann Roturier. 2007. How portable are controlled language rules? In *Proceedings of the Machine Translation Summit XI*, pages 345–352, Copenhagen, Denmark.

Peter Pym. 1990. Pre-editing and the use of simplified writing for MT. In Pamela Mayorcas, editor, *Translating and the Computer 10: The Translation Environment 10 Years on*, pages 80–95. Aslib, London.

Ursula Reuther. 2003. Two in one – Can it work?: Readability and translatability by means of controlled language. In *Proceedings of the Joint Conference Combining the 8th International Workshop of the European Association for Machine Translation and the 4th Controlled Language Applications Workshop (EAMT/CLAW)*, pages 124–132, Dublin, Ireland.

Violeta Seretan, Pierrette Bouillon, and Johanna Gerlach. 2014. A large-scale evaluation of pre-editing strategies for improving user-generated content translation. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pages 1793–1799, Reykjavik, Iceland.

Satoshi Shirai, Satoru Ikehara, Akio Yokoo, and Yoshifumi Ooyama. 1998. Automatic rewriting method for internal expressions in Japanese to English MT and its effects. In *Proceedings of the 2nd International Workshop on Controlled Language Applications (CLAW)*, pages 62–75, Pennsylvania, USA.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 223–231, Cambridge, Massachusetts, USA.

Yanli Sun, Sharon O'Brien, Minako O'Hagan, and Fred Hollowood. 2010. A novel statistical preprocessing model for rule-based machine translation system. In *Proceedings of the 14th Annual Conference of the European Association for Machine Translation (EAMT)*, Saint-Raphaël, France.

Jean-Paul Vinay and Jean Darbelnet. 1958. *Stylistique comparée du français et de l'anglais*. Didier, Paris, trans. and ed. by J. C. Sager & M.-J. Hamel (1995) as *Comparative Stylistics of French and English: A Methodology for Translation.* John Benjamins, Amsterdam.

Sanja Štajner and Maja Popović. 2018. Improving machine translation of English relative clauses with automatic text simplification. In *Proceedings of the 1st Workshop on Automatic Text Adaptation (ATA)*, pages 39–48, Tilburg, Netherlands.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 508–514, Geneva, Switzerland.

Takehiko Yoshimi. 2001. Improvement of translation quality of English newspaper headlines by automatic pre-editing. *Machine Translation*, 16(4):233–250.

Zhongyuan Zhu. 2015. Evaluating neural machine translation in English-Japanese task. In *Proceedings of the 2nd Workshop on Asian Translation (WAT)*, pages 61–68, Kyoto, Japan.

Explicitation strategy	Total		Example of ST pre-editing	MT output
Information addition	142		12日は台湾の休日のため休場。 *12-nichi wa taiwan no kyujitsu no tame kyujo.*	The twelfth is a holiday in Taiwan.
		→	12日は台湾の休日のため**株式市場は**休場。 *12-nichi wa taiwan no kyujitsu no tame **kabushiki shijo wa** kyujo.*	**The stock market** was closed on the twelfth due to a holiday in Taiwan.
Use of clear relation	103		来院しなくても10日前後で登録のクレジットカードから**引き落としを行います**。 *Raiin-shinakutemo toka zengo de touroku no kurejitto kado kara **hikiotoshi o okonaimasu**.*	Withdraw from your registered credit card in about 10 days without visiting the hospital.
		→	来院しなくても10日前後で登録のクレジットカードから**引き落としが行われます**。 *Raiin-shinakutemo toka zengo de touroku no kurejitto kado kara **hikiotoshi ga okonawaremasu**.*	Even if you do not visit the hospital, your credit card will be debited in about 10 days.
Use of narrower sense	54		採尿と採便を**出して**ください。 *Sai-nyo to sai-ben o **dashite** kudasai.*	Please **collect** urine and feces.
		→	採尿と採便を**提出して**ください。 *Sai-nyo to sai-ben o **teishutsushite** kudasai.*	Please **submit** urine and stool samples.
Normalisation	30		単位は億円。 *Tan'i wa oku en.*	**Figures are in billions of yen**.
		→	単位は億円**です**。 *Tan'i wa oku en **desu**.*	**The unit is 100 million yen**.

Table 7: The number of instances and an example of each explicitation strategy for pre-editing ST with MT outputs.

A Details of Explicitation Strategy

Table 7 shows the statistics and examples of each subcategory of the explicitation strategy. A total of 329 pre-editing instances of the explicitation strategy can be further classified into four subcategories: information addition, use of clear relation, use of narrower sense, and normalisation.

The example of the information addition illustrates the insertion of a subject '*kabushiki shijo wa*' ('stock market'), which is implicit in the preceding ST. The example of the use of clear relation shows that the relation between the subject and object can be clarified by using the nominative case marker '*ga*' instead of the accusative one '*o*' and accordingly changing the voice of the main clause. As a result, the inappropriate imperative construction 'Withdraw from ...' in the MT output is changed to the correct passive construction 'will be debited.' In the example of the use of narrower sense, the verb '*dashite*,' which has multiple meanings such as 'put,' 'take,' and 'send,' was replaced with the verb '*teishutsushite*,' which has a narrower range of meaning and was correctly translated as 'submit.' In the example of normalisation, the elliptic sentence-ending was completed with a normal structure '... desu.' This operation led to not only the improvement of the sentence construction, but also the semantic correctness in the MT output ('billions'→'100 million').

RelWalk – A Latent Variable Model Approach to Knowledge Graph Embedding

Danushka Bollegala[1]*, **Huda Hakami**[2], **Yuichi Yoshida**[3] and **Ken-ichi Kawarabayashi**[3]
[1]University of Liverpool, Amazon.
[2]Taif University, Saudi Arabia
[3]National Institute of Informatics, Japan
`danushka@liverpool.ac.uk`, `hahakami@tu.edu.sa`
`{yyoshida,k_keniti}@nii.ac.jp`

Abstract

Embedding entities and relations of a knowledge graph in a low-dimensional space has shown impressive performance in predicting missing links between entities. Although progresses have been achieved, existing methods are heuristically motivated and theoretical understanding of such embeddings is comparatively underdeveloped. This paper extends the random walk model (Arora et al., 2016a) of word embeddings to Knowledge Graph Embeddings (KGEs) to derive a scoring function that evaluates the strength of a relation R between two entities h (head) and t (tail). Moreover, we show that marginal loss minimisation, a popular objective used in much prior work in KGE, follows naturally from the log-likelihood ratio maximisation under the probabilities estimated from the KGEs according to our theoretical relationship. We propose a learning objective motivated by the theoretical analysis to learn KGEs from a given knowledge graph. Using the derived objective, accurate KGEs are learnt from FB15K237 and WN18RR benchmark datasets, providing empirical evidence in support of the theory.

1 Introduction

Knowedge graphs (KGs) such as Freebase (Bollacker et al., 2008) organise information in the form of graphs, where entities are represented by the vertices and the relations between two entities are represented by the edges that connect the corresponding vertices. Despite the best efforts to create complete and large-scale KGs, most KGs remain incomplete and do not represent all the relations that exist between entities (Min et al., 2013). In particular, new entities are constantly being generated, and new relations are formed between new as well as existing entities. Therefore, it is unrealistic to assume that a real-world KG would be complete at any given time point. Developing approaches for KG completion is an important research field associated with KGs.

KG components can be embedded into numerical formats by learning representations (a.k.a embeddings) for the entities and relations in a given KG. The learnt KGEs can be used for *link prediction*, which is the task of predicting whether a particular relation exists between two given entities in the KG. Specifically, given KGEs for entities and relations, in link prediction, we predict R that is most likely to exist between h and t according to some scoring formula. Thus, by embedding entities and relations that exist in a KG in some (possibly lower-dimensional and latent) space, we can infer previously unseen relations between entities, thereby expanding a given KG.

KGE can be seen as a two-step process. Given a KG represented by a set of relational triples (h, R, t), where a semantic relation R holds between a head entity h and a tail entity t, first a scoring function is defined that measures the *relational strength* of a triple (h, R, t). Second,the entity and relation embeddings that optimise the defined scoring function are learnt using some optimisation method. Despite the wide applications of entity and relation embeddings created via KGE methods, the existing scoring functions are heuristically motivated to capture some geometric requirements of the embedding space. For example, TransE (Bordes et al., 2011) assumes that the entity and relation embeddings co-exist in the same (possibly lower dimensional) vector space and translating (shifting) the head entity embedding by the relation embedding must make it closer to the tail entity embedding, whereas ComplEx (Trouillon et al., 2016) models the asymmetry in relations using the component-wise multi-linear inner-product among

---

[*] Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1551–1565
April 19 - 23, 2021.

entity and relation embeddings.

Theoretical understanding of KGE methods is under developed. For example, it is not clear how the heuristically defined KGE objectives relate to the generative process of a KG. Providing such a theoretical understanding of the KGE process will enable us to develop KGE methods that address the weaknesses in the existing KGE methods. For this purpose, we propose Relational Walk (**RelWalk**), a theoretically motivated generative approach for learning KGEs. We are particularly interested in the semantic relationships that exist between entities such as the is-CEO-of relation between a person such as **Jeff Bezos** and a company such as **Amazon Inc.**

We model KGE as a random walk over the KG. Specifically, a random walker at the vertex corresponding to the (head) entity h will uniformly at random select one of the outgoing edges corresponding to the semantic relation R, which will lead it to the vertex corresponding to the (tail) entity t. Continuing this random walk will result in a traversal over a path in the KG. Based on this random walk model we derive a relationship between the probability of R holding between h and t, $p(h, t \mid R)$, and their KGEs $\mathbf{R}$, $\boldsymbol{h}$ and $\boldsymbol{t}$. Interestingly, the derived relationship is not covered by any of the previously proposed heuristically-motivated scoring functions, providing the first-ever KGE method with a provable generative explanation.

We show that the *margin loss*, a popular training objective in prior work on KGE, naturally emerges as the log-likelihood ratio computed from the derived $p(h, t \mid R)$. Based on this result, we derive a training objective that is optimised for learning KGEs that satisfy our theoretical relationship. This enables us to empirically verify the theoretical relationships that we derived from the proposed random walk process.

Using FB15K237 and WN18RR benchmarks, we evaluate the learnt KGEs on link prediction and triple classification. Although we do not obtain state-of-the-art (SoTA) performance on these benchmark datasets, KGEs learnt using RelWalk perform consistently well on both tasks, providing empirical support to the theoretical analysis conducted in this paper. We re-emphasise that our main objective in this paper is to study KGEs from an interpretable theoretical perspective and not necessarily improving SoTA. To this end, we study the relationship between the concentration of the partition function as predicted by our theoretical analysis and the performance of the learnt KGEs. We observe that when the partition function is narrowly distributed, we are able to learn accurate KGEs. Moreover, we empirically verify that the learnt relation embedding matrices satisfy the orthogonality property as expected by the theoretical analysis.

2 Related Work

At a high-level of abstraction, KGE methods can be seen as differing in their design choices for the following two main problems: (a) how to represent entities and relations, and (b) how to model the interaction between two entities and a relation that holds between them. Next, we briefly discuss prior proposals to those two problems (refer to Wang et al. (2017); Nguyen (2017); Nickel et al. (2015) for an extended survey on KGE).

A popular choice for representing entities is to use vectors (Bordes et al., 2013; Ji et al., 2015; Yang et al., 2015), whereas relations have been represented by vectors, matrices (Bordes et al., 2011; Nguyen et al., 2016; Nickel et al., 2011) or tensors (Socher et al., 2013). ComplEx (Trouillon et al., 2016) introduced complex vectors for KGEs to capture the asymmetry in semantic relations. Ding et al. (2018) further improved ComplEx by imposing non-negativity and entailment constraints to ComplEx.

Given entity and relation embeddings, a scoring function evaluates the strength of a triple (h, R, t). Scoring functions that encode various intuitions have been proposed such as the ℓ_1 or ℓ_2 norms of the vector formed by a translation of the head entity embedding by the relation embedding over the target embedding, or by first performing a projection from the entity embedding space to the relation embedding space (Yoon et al., 2016). As an alternative to using vector norms as scoring functions, DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016) use the component-wise multi-linear dot product. Lacroix et al. (2018) proposed the use of nuclear 3-norm regularisers instead of the popular Frobenius norm for canonical tensor decomposition. Table 1 shows the scoring functions along with algebraic structures for entities and relations proposed in selected prior work in KGE learning. Given a scoring function, KGEs are learnt that assign higher scores to relational triples in existing KGs over triples where the relation does not hold (negative triples) by minimising a loss function such as the logistic loss (RESCAL, DistMult, ComplEx) or marginal loss (TransE).

Alternatively to directly learning embeddings from a graph, several methods (Grover and Leskovec, 2016; Perozzi et al., 2014; Ristoski et al., 2018) have considered the vertices visited during truncated random walks over the graph as *pseudo sentences*, and have applied popular word embedding learning algorithms such as continuous bag-of-words model (Mikolov et al., 2013) to learn vertex embeddings. However, pseudo sentences generated in this manner are syntactically very different from sentences in natural languages.

On the other hand, our work extends the random walk analysis by Arora et al. (2016a) that derives a useful connection between the joint co-occurrence probability of two words and the ℓ_2 norm of the sum of the corresponding word embeddings. Specifically, they proposed a latent variable model where the words in a corpus are generated by a probabilistic model parametrised by a time-dependent discourse vector that performs a random walk. In contrast to Arora's model that uses co-occurrences as a generic relation, in our work we include relations as labels for the edges in the graph. Bollegala et al. (2018) extended the model proposed by Arora et al. (2016a) to capture co-occurrences involving more than two words. Specifically, they defined the co-occurrence of k unique words in a given context as a k-way co-occurrence, where Arora et al. (2016a) result could be seen as a special case corresponding to $k = 2$. Moreover, it has been shown that it is possible to learn word embeddings that capture some types of semantic relations such as antonymy and collocation using 3-way co-occurrences more accurately than using 2-way co-occurrences. However, that model does not explicitly consider the relations between words/entities and uses only a corpus for learning the word embeddings.

3 Relational Walk

Let us consider a KG, $\mathcal{D}$, where the *knowledge* is represented by relational triples $(h, R, t) \in \mathcal{D}$. Here, R is a relational predicate with two arguments, where h (*head*) and t (*tail*) entities respectively filling the first and second arguments. In this work, we assume relations to be asymmetric in general (if $(h, R, t) \in \mathcal{D}$ then it does not necessarily follow that $(t, R, h) \in \mathcal{D}$). The goal of KGE is to learn embeddings for the relations and entities in the KG such that the entities that participate in similar relations are embedded closely to each other in the entity embedding space, while at the same time relations that hold between similar entities are embedded closely to each other in the relational embedding space. We call the learnt entity and relation embeddings collectively as KGEs. We assume that entities and relations are embedded in the same vector space, allowing us to perform linear algebraic operations using the embeddings in the same vector space.

KGE method	Score function $f(h, R, t)$	Relation parameters
Unstructured (Bordes et al., 2012)	$\|\|\boldsymbol{h} - \boldsymbol{t}\|\|_{\ell_{1/2}}$	none
Structured (Bordes et al., 2011)	$\|\|\mathbf{R}_1\boldsymbol{h} - \mathbf{R}_2\boldsymbol{t}\|\|_{\ell_{1,2}}$	$\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{d \times d}$
TransE (Bordes et al., 2013)	$\|\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|\|_{\ell_{1/2}}$	$\boldsymbol{r} \in \mathbb{R}^d$
DistMult (Yang et al., 2015)	$\langle \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \rangle$	$\boldsymbol{r} \in \mathbb{R}^d$
RESCAL (Nickel et al., 2011)	$\boldsymbol{h}^\top \mathbf{R} \boldsymbol{t}$	$\mathbf{R} \in \mathbb{R}^{d \times d}$
ComplEx (Trouillon et al., 2016)	$\langle \boldsymbol{h}, \boldsymbol{r}, \bar{\boldsymbol{t}} \rangle$	$\boldsymbol{r} \in \mathbb{C}^d$

Table 1: Score functions proposed in selected prior work on KGEs. Entity embeddings $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$ are vectors in all models, except in ComplEx where $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{C}^d$. Here, $\ell_{1/2}$ denotes either ℓ_1 or ℓ_2 norm of a vector. In ComplEx, $\bar{\boldsymbol{t}}$ is the element-wise complex conjugate.

Following our aforementioned modelling of a knowledge base as a graph, let us consider a random walker who is at a vertex corresponding to some entity h. This entity will have one or more semantic relations with other entities in the KG. The random walker will uniformly at random pick one of the outgoing edges corresponding to a particular semantic relation R, and follow it to land on the entity t. This one-step of the random walk thus *generates* a tuple (h, R, t) in the KG. The random walker proceeds by using t as the new starting point. Multiple steps of this random walk trace a single *path* in the KG.

To illustrate a random walk over a KG, let us assume that we are currently at the vertex corresponding to the company entity **Amazon Inc.** Possible outgoing edges at **Amazon Inc.** would correspond to semantic relations such as has-ceo, is-headquarted-at, founded-in etc., where **Amazon Inc.** is the head entity. If there are only three such outgoing relations at **Amazon Inc.**, then the random walker will pick any one of those relations with a probability $1/3$. For example, by selecting has-ceo, is-headquarted-at or founded-in the random walker would arrive at entities respectively **Jeff Bezos**, **Seattle** or **1994**. Let us assume that the random walker selected the has-ceo relation and landed at **Jeff Bezos**. The random walker might subsequently continue its random walk from **Jeff Bezos** following the relation born-in and transiting to **New Mexico, US**. Prior work studying inferences in KGs have successfully used random walk models similar to what we describe here (Gardner et al., 2013; Lao et al., 2012, 2011; Lao and Cohen, 2010).

Let us consider a random walk characterised by a time-dependent *knowledge vector* $\boldsymbol{c}_k$, where k is the current time step. The knowledge vector represents the knowledge we have about a particular group of entities and relations that express some facts about the world. For example, when we are talking about **Amazon Inc.**, we will use the knowledge associated with **Amazon Inc.** such as its CEO, location of the headquarters, when it was founded etc. Therefore, it is intuitive to assume that the entities associated with **Amazon Inc.** with some set of semantic relations can be generated from this knowledge vector. Each entity and relation has time-independent latent representations that capture their correlations with $\boldsymbol{c}_k$. For entities h and t, we denote their representations by d-dimensional vectors respectively $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$.

We assume the task of generating a relational triple (h, R, t) in a given KG to be a two-step process as described next. First, given the current knowledge vector at time k, $\boldsymbol{c} = \boldsymbol{c}_k$ and the relation R, we assume that the probability of an entity h satisfying the first argument of R to be given by the loglinear entity production model in (1).

$$p(h \mid R, \boldsymbol{c}) = \frac{1}{Z_c} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right). \tag{1}$$

Here, $\mathbf{R}_1 \in \mathbb{R}^{d \times d}$ is a relation-specific orthogonal matrix that evaluates the appropriateness of h for the first argument of R. For example, if R is the is-ceo-of relation, we would require a person as the first argument and a company as the second argument of R. However, note that the role of $\mathbf{R}_1$ extends beyond simply checking the types of the entities that can fill the first argument of a relation. For our example above, not all people are CEOs and $\mathbf{R}_1$ evaluates the likelihood of a person to be selected as the first argument of the ceo-of relation. Z_c is a normalisation coefficient such that $\sum_{h \in \mathcal{V}} p(h \mid R, \boldsymbol{c}) = 1$, where the vocabulary $\mathcal{V}$ is the set of all entities in the KG.

After generating h, the state of our random walker changes to $\boldsymbol{c}' = \boldsymbol{c}_{k+1}$, and we next generate the second argument of R with the probability given by (2).

$$p(t \mid R, \boldsymbol{c}') = \frac{1}{Z_{c'}} \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right). \tag{2}$$

Here, $\mathbf{R}_2 \in \mathbb{R}^{d \times d}$ is a relation-specific orthogonal matrix that evaluates the appropriateness of t as the second argument of R. $Z_{c'}$ is a normalisation coefficient such that $\sum_{t \in \mathcal{V}} p(t \mid R, \boldsymbol{c}') = 1$. Following our previous example of is-ceo-of relation, $\mathbf{R}_2$ evaluates the likelihood of an organisation to be a company with a CEO position. Importantly, $\mathbf{R}_1$ and $\mathbf{R}_2$ are representations of the relation R and independent of the entities. Therefore, we consider ($\mathbf{R}_1$ and $\mathbf{R}_2$) to collectively represent the embedding of R. Orthogonality of $\mathbf{R}_1, \mathbf{R}_2$ is a requirement for the mathematical proof and also acts as a regularisation constraint to prevent overfitting by restricting the relational embedding space (Tang et al., 2020). Intuitively, orthogonality of the relation embedding matrices ensures that the length of the head and tail entity embeddings are not altered during the generation of the tuple. Orthogonal transformation has been shown to improve the performance of relation representation in prior work. For example, Tang et al. (2020) apply orthogonal transformation that extends RotatE (Sun et al., 2019) to model complex relations (e.g., N-to-N). For relationships in word embedding space, Ethayarajh (2019) use orthogonal transformation for analogical reasoning between words. The performance of hypernymy prediction through orthogonal projections has been improved as shown in Wang et al. (2019).

The knowledge vector $\boldsymbol{c}_k$ performs a *slow* random walk (meaning $\boldsymbol{c}_{k+1}$ is obtained from $\boldsymbol{c}_k$ by adding a small random displacement vector) such that the head and tail entities of a relation are generated under similar knowledge vectors. More specifically, we assume that $||\boldsymbol{c}_k - \boldsymbol{c}_{k+1}|| \leq \epsilon_2$ for some small $\epsilon_2 > 0$. This is a realistic assumption for generating the two entity arguments in the same relational triple because, if the knowledge vectors were significantly different in the two generation steps, then it is likely that the corresponding relations are also different, which would not be coherent with the above-described generative process. Moreover, we assume that the knowledge vectors are distributed uniformly in the unit sphere and denote the distribution of knowledge vectors by $\mathcal{C}$.

To relate KGEs with the connections in the graph, we must estimate the probability that h and t satisfy the relation R, $p(h, t \mid R)$, which can be obtained by taking the expectation of $p(h, t \mid R, \boldsymbol{c}, \boldsymbol{c}')$ w.r.t. $\boldsymbol{c}, \boldsymbol{c}' \sim \mathcal{C}$ given by (3).

$$p(h, t \mid R) = \mathbb{E}_{\boldsymbol{c},\boldsymbol{c}'}\left[p(h, t \mid R, \boldsymbol{c}, \boldsymbol{c}')\right] \tag{3}$$

$$= \mathbb{E}_{\boldsymbol{c},\boldsymbol{c}'}\left[p(h \mid R, \boldsymbol{c})p(t \mid R, \boldsymbol{c}')\right] \tag{4}$$

$$= \mathbb{E}_{\boldsymbol{c},\boldsymbol{c}'}\left[\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)}{Z_c}\frac{\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right)}{Z_{c'}}\right]. \tag{5}$$

Here, partition functions are given by

$$Z_c = \sum_{h \in \mathcal{V}} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \tag{6}$$

$$Z_{c'} = \sum_{t \in \mathcal{V}} \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right) \tag{7}$$

(4) follows from our two-step generative process where the generation of h and t in each step is independent given the relation and the corresponding knowledge vectors.

Computing the expectation in (5) is generally difficult because of the two partition functions Z_c and $Z_{c'}$. However, Lemma 1 shows that the partition functions are narrowly distributed around a constant value for all c (or c') values with high probability.

Lemma 1 (Concentration Lemma). *If the entity embedding vectors satisfy the Bayesian prior* $\boldsymbol{v} = s\hat{\boldsymbol{v}}$, *where* $\hat{\boldsymbol{v}}$ *is from the spherical Gaussian distribution, and* s *is a scalar random variable, which is always bounded by a constant* κ, *then the entire ensemble of entity embeddings satisfies that:*

$$\Pr_{c \sim \mathcal{C}}[(1 - \epsilon_z)Z \leq Z_c \leq (1 + \epsilon_z)Z] \geq 1 - \delta, \tag{8}$$

for $\epsilon_z = O(1/\sqrt{n})$, *and* $\delta = \exp(-\Omega(\log^2 n))$, *where* $n \geq d$ *is the number of entities in a given KG and* Z_c *is the partition function for* c *given by* $\sum_{h \in \mathcal{V}} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)$.

Refer to Appendix A for the proof of the concentration lemma. We empirically investigate the relationship between the performance of the KGEs and the degree to which Lemma 1 is satisfied in subsection 5.1. Under the conditions required to satisfy Lemma 1, the following main theorem of this paper holds:

Theorem 1. *Suppose that the entity embeddings satisfy* (1). *Then, we have*

$$\log p(h, t \mid R) = \frac{\left|\left|{\mathbf{R}_1}^\top \boldsymbol{h} + {\mathbf{R}_2}^\top \boldsymbol{t}\right|\right|_2^2}{2d} - 2\log Z \pm \epsilon. \tag{9}$$

for $\epsilon = O(1/\sqrt{n}) + \widetilde{O}(1/d)$, *where*

$$Z = Z_c = Z_{c'}. \tag{10}$$

Proof sketch: Let F be the event that both c and c' are within $(1 \pm \epsilon_z)Z$. Then, from Lemma 1 and the union bound, event F happens with probability at least $1 - 2\exp(-\Omega(\log^2 n))$. The R.H.S. of (5) can be split into two parts T_1 and T_2 according to whether F happens or not.

$$p(h, t \mid R) = \underbrace{\mathbb{E}_{c,c'}\left[\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)}{Z_c}\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_2 \boldsymbol{c}'\right)}{Z_{c'}}\mathbf{1}_F\right]}_{=T_1} + \underbrace{\mathbb{E}_{c,c'}\left[\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)}{Z_c}\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_2 \boldsymbol{c}'\right)}{Z_{c'}}\mathbf{1}_{\bar{F}}\right]}_{=T_2}. \tag{11}$$

T_1 can be approximated as given by (12).

$$T_1 = \frac{1 \pm \mathcal{O}(\epsilon_z)}{Z^2}\mathbb{E}_{c,c'}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right)\right] \tag{12}$$

On the other hand, T_2 can be shown to be a constant, independent of d, given by (13).

$$|T_2| = \exp(-\Omega(\log^{1.8} n)) \tag{13}$$

The vocabulary size n of real-world KGs is typically over 10^5, for which T_2 becomes negligibly small. Therefore, it suffices to consider only T_1. Because of the slowness of the random walk we have $\boldsymbol{c} \approx \boldsymbol{c}'$.

Using the law of total expectation we can write T_1 as follows:

$$T_1 = \frac{1 \pm \mathcal{O}(\epsilon_z)}{Z^2}\mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)\mathbb{E}_{c'|c}\left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right)\right]\right]$$

$$= \frac{1 \pm \mathcal{O}(\epsilon_z)}{Z^2}\mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)A(c)\right] \tag{14}$$

where $A(c) := \mathbb{E}_{c'|c}\left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}'\right)\right]$. Doing some further evaluations we show that

$$A(c) = (1 \pm \epsilon_2)\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \tag{15}$$

Plugging (51) back in (14) provides the claim of the theorem. Detailed proof is shown in Appendix B. □

The relationship given by (9) indicates that head and tail entity embeddings are first transformed respectively by ${\mathbf{R}_1}^\top$ and ${\mathbf{R}_2}^\top$, and the squared ℓ_2 norm of the sum of the transformed vectors is proportional to the probability $p(h, t \mid R)$.

4 Learning KG Embeddings

In this section, we derive a training objective from Theorem 1 that we can then optimise to learn KGEs. The goal is to empirically validate the theoretical result by evaluating the learnt KGEs. KGs represent information about relations between two entities in the form of *relational triples*. The joint probability $p(h, R, t)$ given by Theorem 1 is useful for determining whether a relation R exists between two given entities h and t. For example, if we know that with a high probability that R holds between h and t, then we can append (h, R, t) to the KG. The task of expanding KGs by predicting missing links between entities or relations is known as the *link prediction* problem (Trouillon et al., 2016). In particular, if we can automatically append such previously unknown knowledge to the KG, we can expand the KG and address the knowledge acquisition bottleneck.

To derive a criteria for determining whether a link must be predicted among entities and relations, let us consider a relational triple $(h, R, t) \in \mathcal{D}$ that exists in a given KG $\mathcal{D}$. We call such relational triples as *positive* triples because from the assumption it is known that R holds between h and t. On the other hand, consider a *negative* relational triple $(h', R, t') \in \mathcal{D}$ formed by, for example, randomly perturbing a positive triple. A popular technique for generating such (pseudo) negative triples is to replace h or t with a randomly selected different instance of the same entity type. As an alternative for random perturbation, Cai and Wang (2018) proposed a method for generating negative instances using adversarial learning. Here, we are not concerned about the actual method used for generating the negative triples but assume a set of negative triples, $\bar{\mathcal{D}}$, generated using some method, to be given.

Given a positive triple $(h, R, t) \in \mathcal{D}$ and a negative triple $(h', R, t') \in \bar{\mathcal{D}}$, we would like to learn KGEs such that a higher probability is assigned to (h, R, t) than that assigned to (h', R, t'). We can formalise this requirement using the likelihood ratio given by (16).

$$\frac{p(h, R, t)}{p(h', R, t')} \geq \eta \tag{16}$$

Here, $\eta > 1$ is a threshold that determines how higher we would like to set the probabilities for the positive triples compares to that of the negative triples.

By taking the logarithm of both sides in (16) we obtain

$$\log p(h, R, t) - \log p(h', R, t') \geq \log \eta$$
$$\log \eta + \log p(h', R, t') - \log p(h, R, t) \leq 0 \tag{17}$$

If a positive triple (h, R, t) is correctly assigned a higher probability than a negative triple $p(h', R, t')$, then the left hand side of (17) will be negative, indicating that there is no *loss* incurred during this classification task. Therefore, we can re-write (17) to obtain the *marginal loss* (Bordes et al., 2013, 2011), $L(\mathcal{D}, \bar{\mathcal{D}})$, a popular choice as a learning objective in prior work in KGE, as shown in (18).

$$\begin{aligned} L(\mathcal{D}, \bar{\mathcal{D}}) = & \\ & \sum_{\substack{(h,R,t)\in\mathcal{D} \\ (h',R,t')\in\bar{\mathcal{D}}}} \max\left(0, \log \eta + \log p(h', R, t') - \log p(h, R, t)\right) \\ = & \max\Big(0, 2d \log \eta + \left|\left|\mathbf{R}_1{}^\top \boldsymbol{h}' + \mathbf{R}_2{}^\top \boldsymbol{t}'\right|\right|_2^2 \\ & - \left|\left|\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right|\right|_2^2\Big) \end{aligned} \tag{18}$$

We can assume $2d \log \eta$ to be the *margin* for the constraint violation.

Theorem 1 requires $\mathbf{R}_1$ and $\mathbf{R}_2$ to be orthogonal. To reflect this requirement, we add two ℓ_2 regularisation terms $\left|\left|\mathbf{R}_1{}^\top \mathbf{R}_1 - \mathbf{I}\right|\right|_2^2$ and $\left|\left|\mathbf{R}_2{}^\top \mathbf{R}_2 - \mathbf{I}\right|\right|_2^2$ respectively with regularisation coefficients λ_1 and λ_2 to the objective function given by (18). In our experiments, we compute the gradients (18) w.r.t. each of the parameters $\boldsymbol{h}$, $\boldsymbol{t}$, $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ and use stochastic gradient descent (SGD) for optimisation. Considering that negative triples are generated via random perturbation, it is important to consider multiple negative triples during training to better estimate the classification boundary. This approach can be easily extended to learn from multiple negative triples as shown in Appendix C.

5 Empirical validation

To empirically evaluate the theoretical result stated in Theorem 1, we learn KGEs (denoted by **RelWalk**) by minimising the marginal loss objective derived in section 4. We use the FB15k237, FB13 (subsets of *Freebase*) and WN18RR (a subset of *WordNet*) datasets, which are standard benchmarks for KGE. We use the standard training, validation and test splits. Statistics about the datasets and training details are in Appendix D. **RelWalk** is implemented in the open-source toolkit OpenKE (Han et al., 2018) and the code and learnt KGEs will are publicly available[1].

We conduct two evaluation tasks: *link prediction* (predict the missing head or tail entity in a given triple $(h, R, ?)$ or $(?, R, t)$) (Bordes et al., 2011) and *triple classification* (predict whether a relation R holds between h and t in a given triple

---

[1] `https://github.com/LivNLP/Relational-Walk-for-Knowledge-Graphs`

	FB15K237					WN18RR				
Method	MRR	MR	H@1	H@3	H@10	MRR	MR	H@1	H@3	H@10
TransE$^{\bullet}$	0.294	347	-	-	0.465	0.226	3384	-	-	0.50
TransD$^{\triangleleft}$	0.280	-	-	-	0.453	-	-	-	-	0.43
DistMult*	0.241	254	0.155	0.263	0.419	0.430	5110	0.39	0.44	0.49
ComplEx*	0.247	339	0.158	0.275	0.428	0.440	5261	0.41	0.46	0.51
ConvE (Dettmers et al., 2017a)	0.325	244	0.237	**0.356**	0.501	0.430	4187	0.40	0.44	0.52
CP-N3 (Lacroix et al., 2018)	**0.360**	-	-	-	**0.540**	**0.470**	-	-	-	**0.54**
RelWalk	0.329	**105**	**0.243**	0.354	0.502	0.451	**3232**	**0.42**	**0.47**	0.51

Method	Accuracy
Structured$^{\diamond}$	75.2
TransE$^{\diamond}$	81.5
TransR (Lin et al., 2015)	82.5
TransG (Xiao et al., 2016)	87.3
NTN (Socher et al., 2013)	87.2
RelWalk	**88.6**

Table 2: Results of link prediction (left) and triple classification on FB13 (right). Results marked with [$\star$] are taken from (Dettmers et al., 2017a), [$\bullet$] from (Nguyen et al., 2016),[$\triangleleft$] from (Cai and Wang, 2018) and [$\diamond$] from (Wang et al., 2014). All other results for the baselines are taken from their original papers.

Relation	H@10	ν_R	σ_c	$\sigma_{c'}$	$\sqrt{\sigma_c^2 + \sigma_{c'}^2}$
hypernym	0.188	3.249	68.89	64.41	94.31
derivational	0.955	1.690	63.44	65.33	91.07
instance_hypernym	0.541	0.362	63.11	64.56	90.28
also_see	0.670	0.234	70.76	61.51	93.76
member_meronym	0.281	4.389	63.78	66.09	91.84
synset_domain_topic	0.513	0.727	65.66	65.48	92.73
has_part	0.247	0.548	66.21	66.50	93.84
domain_usage	0.688	0.045	65.24	63.16	90.81
domain_region	0.442	0.065	67.53	66.31	94.64
verb_group	0.974	0.038	64.22	63.19	90.09
similar_to	1.000	0.111	63.67	63.96	90.25
Correlations		−0.51	−0.39	−0.49	−0.70

Table 3: Empirical analysis of the concentration of the partitioning functions and the orthogonality of the relation embeddings, and their Pearson correlation coefficients against H@10 for the relations in WN18RR.

(h, R, t)) (Socher et al., 2013). We evaluate the performance in the link prediction task using mean reciprocal rank (**MRR**), mean rank (**MR**) (the average of the rank assigned to the original head or tail entity in a corrupted triple) and hits at ranks 1, 3 and 10 (**H@1, 3, 10**), whereas in the triple classification task we use **accuracy** (percentage of the correctly classified test triples). We only report scores under the *filtered* setting (Bordes et al., 2013), which removes all triples appeared in training, validating and testing sets from candidate triples before obtaining the rank of the ground truth triple. In link prediction, we consider all entities that appear in the corresponding argument in the entire knowledge graph as candidates.

In Table 2 we compare the KGEs learnt by **RelWalk** against prior work using the published results. For triple classification, **RelWalk** reports the best performance on FB13, outperforming all methods compared. For the link prediction results as shown in Table 2, we see that **RelWalk** obtains competitive performance on both WN18RR and FB15K237 under all evaluation measures. In particular, it is outperformed by the KGE method proposed by Lacroix et al. (2018) (**CP-N3**), which uses nuclear 3-norm regularisers with canonical tensor decomposition. Interestingly, the improvement against structured embeddings (SE) is consistent and interesting because the scoring function of SE closely resembles that of RelWalk as we can redefine $\mathbf{R}_2$ with the negative sign. However, SE learns KGEs that *minimise* the $\ell_{1,2}$ norm whereas according to (9) we must *maximise* the probability for relational triples in a knowledge graph. WN18RR excludes triples from WN18 that are simply inverted between train and test partitions (Toutanova and Chen, 2015; Dettmers et al., 2017b), making it a difficult dataset for link prediction using simple memorisation heuristics. **RelWalk**'s consistent good performance on both versions of this dataset shows that it is considering the global structure in the KG when learning KGEs.

We note that our goal in this paper is *not* to claim SoTA for KGE but to provide a theoretical understanding with empirical validation. To this end, the experimental results support our theoretical claim and emphasise the importance of theoretically motivating the KGE scoring function design process.

5.1 Orthogonality and Concentration

Our theoretical analysis depends on two main assumptions: (a) concentration of the partition function Z_c (Lemma 1), and (b) the orthogonality of the relation embedding matrices $\mathbf{R}_1, \mathbf{R}_2$. In this section, we empirically study the relationship between these assumptions and the performance of RelWalk.

Given $\mathbf{R}_1$ and $\mathbf{R}_2$ learnt by RelWalk for a particular R, we can measure the degree to which the orthogonality, ν_R, is satisfied by the sum of the non-diagonal elements (19).

$$\nu_R = \sum_{i \neq j} |{\mathbf{R}_1}^\top \mathbf{R}_1|_{ij} + |{\mathbf{R}_2}^\top \mathbf{R}_2|_{ij} \tag{19}$$

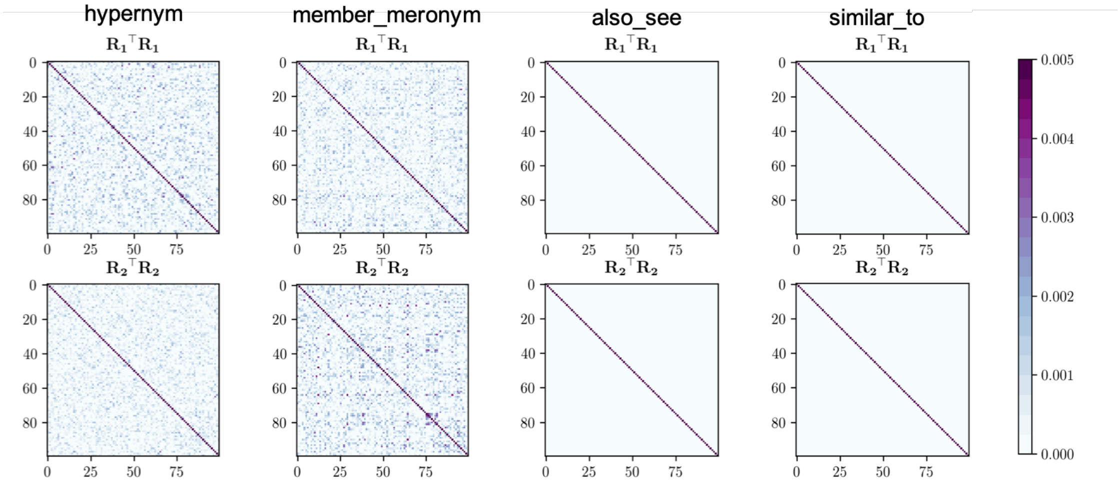


Figure 1: Heatmap visualisation of the orthogonality in different relation embeddings from the WN18RR.

If a matrix $\mathbf{A}$ is orthogonal, then the non-diagonal elements of the inner-product $\mathbf{A}^\top\mathbf{A}$ will contain zeros. Therefore, the smaller the ν_R values, more orthogonal the relation embeddings will be. We measure ν_R values for the 11 relation types in the WN18RR dataset as shown in Table 3. From Table 3 we see that ν_R values are indeed small for different relation types indicating that the orthogonality requirement is satisfied as expected. Interestingly, a moderately high (-0.515) negative Pearson correlation between H@10 and ν_R shows that orthogonality correlates with the better the performance.

To visualise how the orthogonality affects different relation types, we plot the elements in $\mathbf{R}_1{}^\top\mathbf{R}_1$ and $\mathbf{R}_2{}^\top\mathbf{R}_2$ for four relations in the WN18RR dataset in Figure 1 for 100×100 dimensional relational embeddings. For the two relations also_see and similar_to we see that the corresponding inner-products are sparse except in the main diagonal, compared to that in hypernym and member_meronym relations. On the other hand, according to Table 3 the H@10 values for also_see and similar_to are higher than that for hypernym and member_meronym as implied by the negative correlation.

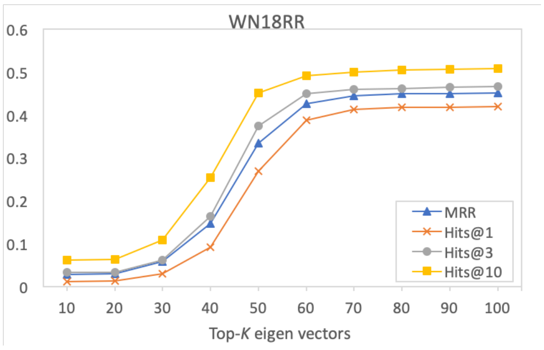


Figure 2: Results for the approximated relation embeddings for link prediction on WN18RR.

To test for the concentration of the partition function, for a relation R we compute Z_c and $Z_{c'}$ values using respectively (6) and (7) over a set of randomly sampled 10000 head or tail entities. We compute the standard deviations σ_c and $\sigma_{c'}$ respectively for the distributions of Z_c and $Z_{c'}$ and their geometric means as shown in Table 3. We observed a Gaussian-like distributions for the partition functions for different relations for which smaller standard deviations indicate stronger concentration around the mean. Interestingly, from Table 3 we see a negative correlation between H@10 and the standard deviations indicating that the performance of RelWalk depends on the validity of the concentration assumption.

5.2 Compression of Embeddings

To reduce the amount of memory required for KGEs, especially with a large KG, compressing KGEs has been studied recently (Sachan, 2020). **RelWalk** uses (orthogonal) matrices to represent relations, which require more parameters compared to a vector representation of the same dimensionality of a relation. Prior work studying lower-rank decomposition of KGEs have shown that, although linear embeddings of graphs can require prohibitively large dimensionality to model certain types of relations (Nickel et al., 2014) (e.g. sameAs), nonlinear embeddings can mitigate this problem (Bouchard et al., 2015). In this section, we propose memory-efficient low-rank approximations to the **RelWalk** embeddings.

From the definition of orthogonality it follows that the relation embeddings $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{d \times d}$ learnt by **RelWalk** for a particular relation R means that $\mathbf{R}_1, \mathbf{R}_2$ are both full-rank and cannot be factorised as the product of two lower rank matrices. This prevents us from directly applying matrix decomposition methods such as non-negative matrix factorisation on the learnt relation embeddings to obtain low-rank approximations. Therefore, we subtract the identity matrix $\mathbf{I} \in \mathbb{R}^{n \times n}$ from the relation embedding $\mathbf{R}(\in \{\mathbf{R}_1, \mathbf{R}_2\})$ and factorise the remainder $\mathbf{R}' \in \mathbb{R}^{n \times n}$ as the product of two low-rank matrices using the eigendecomposition of $\mathbf{R}'$ as given by (20).

$$\begin{aligned}\mathbf{R} &= \mathbf{I} + \mathbf{R}' \\ &= \mathbf{I} + \mathbf{U}_R \mathbf{D} \mathbf{U}_R{}^\top \\ &\approx \mathbf{I} + \sum_{k=1}^{K} \mathbf{D}_{(k,k)} \mathbf{U}_{R(k,:)} \mathbf{U}_{R(:,k)} \end{aligned} \quad (20)$$

Here, $\mathbf{U}$ is the matrix formed by arranging the eigenvectors of $\mathbf{R}'$ as columns, and $\mathbf{D}$ is a diagonal matrix containing the eigenvalues of $\mathbf{R}'$ in the descending order. We can then use the largest $K \leq d$ eigenvalues and corresponding eigenvectors to obtain a rank-K approximation in the sense of minimum Frobenius distance between $\mathbf{R}'$ and its rank-K approximation. In the case we use K factors in the approximation, we must store dK real numbers corresponding to the d-dimensional eigen vectors per each of the K components as opposed to d^2 real numbers in $\mathbf{R}$. The compression ratio in this case becomes $dK/d^2 = K/d$. When $K \ll d$, this results in a significant compression.

To empirically evaluate the trade-off between the number of singular vectors used in the compression and the accuracy of the learnt relation embeddings, we use the approximated relation embeddings for link prediction on WN18RR as shown in Figure 2 (similar trend was observed for FB15K237). We use $d = 100$ dimensional relation embeddings learnt by **RelWalk** and approximate using top-K eigenvectors. From Figure 2 we see for $K > 60$ components the performance saturates in both datasets. On the other hand, we need at least $K = 30$ components to get any meaningful accuracy for link prediction on these two datasets. With $K = 60$ and $d = 100$ this approximation results in an 60% compression ratio.

6 Conclusion

We proposed **RelWalk**, a generative model of KGE and derived a theoretical relationship between the probability of a triple consisting of head, tail entities and the relation that exists between those two entities, and the embedding vectors for the two entities and embeddings matrices for the relation. In RelWalk, we represented entities by vectors and relations by matrices. We then proposed a learning objective based on the theoretical relationship we derived to learn entity and relation embeddings from a given knowledge graph. Experimental results on a link prediction and a triple classification tasks show that **RelWalk** outperforms several previously proposed KGE learning methods. The key assumptions of **RelWalk** are validated by empirically analysing the relationship between such assumptions and the performance of the learnt embeddings from a KG. Moreover, we studied the compressibility of the learnt relation embeddings and discovered that using only 60% of the components, we can approximate the relation embeddings without any significant loss in performance.

Acknowledgement

Yuichi Yoshida and Ken-ichi Kawarabayashi are supported by JSPS KAKENHI Grant Number JP18H05291.

References

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2015. Rand-walk: A latent variable model approach to word embeddings. *arXiv preprint arXiv:1502.03520*.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016a. A latent variable model approach to pmi-based word embeddings. *TACL*, 4:385–399.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016b. Rand-walk: A latent variable model approach to word embeddings. *arXiv*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. of SIGMOD*, pages 1247 – 1250.

Danushka Bollegala, Yuichi Yoshida, and Ken-ichi Kawarabayashi. 2018. Using k-way Co-occurrences for Learning Word Embeddings. In *Proc. of AAAI*.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2012. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhenko. 2013. Translating embeddings for modeling multi-relational data. In *Proc. of NIPS*.

Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. In *Proc. of AAAI*.

Guillaume Bouchard, Sameer Singh, and Théo Trouillon. 2015. On approximate reasoning capabilities of low-rank vector spaces. In *Proc. of Knowlege Representation and Reasoning: Integrating Symbolic and Neural Approaches: Papers from the 2015 AAAI Spring Symposium*, pages 6–9.

Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial learning for knowledge graph embeddings. In *Proc. of NAACL*, pages 1470–1480.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2017a. Convolutional 2d knowledge graph embeddings. *arXiv preprint arXiv:1707.01476*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2017b. Convolutional 2D Knowledge Graph Embeddings.

Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving knowledge graph embedding using simple constraints. In *Proc. of ACL*, pages 110–121.

Kawin Ethayarajh. 2019. Rotate king to get queen: Word relationships as orthogonal transformations in embedding space. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3503–3508, Hong Kong, China. Association for Computational Linguistics.

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 833–838, Seattle, Washington, USA. Association for Computational Linguistics.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proc. of KDD*.

Xu Han, Shulin Cao, Lv Xin, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. Openke: An open toolkit for knowledge embedding. In *Proc. of EMNLP*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proc. of ACL*, pages 687–696.

Timothee Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *Proc. of ICML*, pages 2863–2872.

Ni Lao and William W. Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81(1):53–67.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 529–539, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1017–1026, Jeju Island, Korea. Association for Computational Linguistics.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proc. of AAAI*, pages 2181–2187.

Tomas Mikolov, Kai Chen, and Jeffrey Dean. 2013. Efficient estimation of word representation in vector space. In *Proc. of ICLR*.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 777–782, Atlanta, Georgia.

Dat Quoc Nguyen. 2017. An overview of embedding models of entities and relationships for knowledge base completion.

Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. 2016. Stranse: a novel embedding model of entities and relationships in knowledge bases. In *Proc. of NAACL-HLT*, pages 460–466.

Maximilian Nickel, Xueyan Jiang, and Volker Tresp. 2014. Reducing the rank in relational factorization models by including observable patterns. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1179–1187. Curran Associates, Inc.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proc. of ICML*, pages 809–816.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proc. of KDD*, pages 701–710.

Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. 2018. Rdf2vec: Rdf graph embeddings and their applications. *Semantic Web*, (Preprint):1–32.

Mrinmaya Sachan. 2020. Knowledge graph embedding compression. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2681–2691, Online. Association for Computational Linguistics.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Proc. of NIPS*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou. 2020. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2713–2722, Online. Association for Computational Linguistics.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proc. of 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proc. of ICML*.

Chengyu Wang, Yan Fan, Xiaofeng He, and Aoying Zhou. 2019. A family of fuzzy orthogonal projection models for monolingual and cross-lingual hypernymy prediction. In *The World Wide Web Conference*, pages 1965–1976.

Q. Wang, Z. Mao, B. Wang, and L. Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *TKDE*, 29(12):2724–2743.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proc. of AAAI*, pages 1112 – 1119.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. Transg : A generative model for knowledge graph embedding. In *Proc. of ACL*, pages 2316–2325.

Bishan Yang, Wen-tau Yih, Xiadong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proc. of ICLR*.

Hee-Geun Yoon, Hyun-Je Song, Seong-Bae Park, and Se-Young Park. 2016. A translation-based knowledge graph embedding preserving logical property of relations. In *Proc. of NAACL*, pages 907–916.

A Proof of the Concentration Lemma

To prove the concentration lemma, we show that the mean $\mathbb{E}_{\boldsymbol{h}}[Z_c]$ of Z_c is concentrated around a constant for all knowledge vectors $\boldsymbol{c}$ and its variance is bounded. If $\mathbf{P}$ is an orthogonal matrix and $\boldsymbol{x}$ is a vector, then $||\mathbf{P}^\top \boldsymbol{x}||_2^2 = (\mathbf{P}^\top \boldsymbol{x})^\top (\mathbf{P}^\top \boldsymbol{x}) = \boldsymbol{x}^\top \mathbf{P}\mathbf{P}^\top \boldsymbol{x} = ||\boldsymbol{x}||_2^2$, because $\mathbf{P}^\top \mathbf{P} = \mathbf{I}$. Therefore, from (6) and the orthogonality of the relational embeddings, we see that $\mathbf{R}_1 \boldsymbol{c}$ is a simple rotation of $\boldsymbol{c}$ and does not alter the length of $\boldsymbol{c}$. We represent $\boldsymbol{h} = s_h \hat{\boldsymbol{h}}$, where $s_h = ||\boldsymbol{h}||$ and $\hat{\boldsymbol{h}}$ is a unit vector (i.e. $\left|\left|\hat{\boldsymbol{h}}\right|\right|_2 = 1$) distributed on the spherical Gaussian with zero mean and unit covariance matrix $\mathbf{I}_d \in \mathbb{R}^{d \times d}$. Let s be a random variable that has the same distribution as s_h. Moreover, let us assume that s is upper bounded by a constant κ such that $s \leq \kappa$. From the assumption of the knowledge vector $\boldsymbol{c}$, it is on the unit sphere as well, which is then rotated by $\mathbf{R}_1$.

We can write the partition function using the inner-product between two vectors $\boldsymbol{h}$ and $\mathbf{R}_1 \boldsymbol{c}$, $Z_c = \sum_{h \in \mathcal{V}} \exp\left(\boldsymbol{h}^\top (\mathbf{R}_1 \boldsymbol{c})\right)$. Arora et al.(Arora et al., 2016a) showed that (Lemma 2.1 in their paper) the expectation of a partition function of this form can be approximated as follows:

$$\mathbb{E}_{\boldsymbol{h}}[Z_c] = n\mathbb{E}_{\boldsymbol{h}}[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)] \tag{21}$$

$$\geq n\mathbb{E}_{\boldsymbol{h}}[1 + \boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}] = n. \tag{22}$$

where $n = |\mathcal{V}|$ is the number of entities in the vocabulary. (21) follows from the expectation of a sum and the independence of $\boldsymbol{h}$ and $\mathbf{R}_1$ from $\boldsymbol{c}$. The inequality of (22) is obtained by applying the Taylor expansion of the exponential series and the final equality is due to the symmetry of the spherical Gaussian. From the law of total expectation, we can write

$$\begin{aligned}\mathbb{E}_{\boldsymbol{h}}[Z_c] &= n\mathbb{E}_{\boldsymbol{h}}[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)] \\ &= n\mathbb{E}_{s_h}\left[\mathbb{E}_{x|s_h}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \mid s_h\right]\right].\end{aligned} \tag{23}$$

where, $x = \boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}$. Note that conditioned on s_h, $\boldsymbol{h}$ is a Gaussian random variable with variance $\sigma^2 = s_h^2$. Therefore, conditioned on s_h, x is a random variable with variance $\sigma^2 = \sigma_h^2$. Using this distribution, we can evaluate $\mathbb{E}_{x|s_h}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)\right]$ as follows:

$$\begin{aligned}&\mathbb{E}_{x|s_h}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \mid s_h\right] \\ &= \int_x \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \exp(x) dx \\ &= \int_x \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\sigma^2)^2}{2\sigma^2} + \sigma^2/2\right) dx \\ &= \exp(\sigma^2/2).\end{aligned} \tag{24}$$

Therefore, it follows that

$$\begin{aligned}\mathbb{E}_{\boldsymbol{h}}[Z_c] &= n\mathbb{E}_{s_h}[\exp(\sigma^2/2)] \\ &= n\mathbb{E}_{s_h}[\exp(s_h^2/2)] = n\exp(s^2/2),\end{aligned} \tag{25}$$

where s is the variance of the ℓ_2 norms of the entity embeddings. Because the set of entities is given and fixed, both n and σ are constants, proving that $\mathbb{E}_{\boldsymbol{h}}[Z_c]$ does not depend on c.

Next, we calculate the variance $\mathbb{V}_{\boldsymbol{h}}[Z_c]$ as follows:

$$\begin{aligned}\mathbb{V}_{\boldsymbol{h}}[Z_c] &= \sum_h \mathbb{V}_{\boldsymbol{h}}[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)] \\ &\le n\mathbb{E}_{\boldsymbol{h}}\left[\exp\left(2\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)\right] \\ &= n\mathbb{E}_{s_h}\left[\mathbb{E}_{x|s_h}\left[\exp\left(2\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \mid s_h\right]\right].\end{aligned} \tag{26}$$

Because $2\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}$ is a Gaussian random variable with variance $4\sigma^2 = 4s_h^2$ from a similar calculation as in (24) we obtain,

$$\mathbb{E}_{x|s_h}\left[\exp\left(2\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \mid s_h\right] = \exp(2\sigma^2). \tag{27}$$

By substituting (27) in (26) we have that

$$\mathbb{V}_{\boldsymbol{h}}[Z_c] \le n\mathbb{E}_{s_h}\left[\exp\left(2\sigma^2\right)\right] = n\mathbb{E}_{s_h}\left[\exp(2s^2)\right] \le \Lambda n \tag{28}$$

for $\Lambda = \exp(8\kappa^2)$ a constant bounding $s \le \kappa$ as stated. From above, we have bounded both the mean and variance of the partition function by constants that are independent of the knowledge vector. Note that neither $\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)$ nor $\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)$ are sub-Gaussian nor sub-exponential. Therefore, standard concentration bounds derived for sub-Gaussian or sub-exponential random variables cannot be used in our analysis. However, the argument given in Appendix A.1 in (Arora et al., 2016b) for a partition function with bounded mean and variance can be directly applied to Z_c in our case, which completes the proof of the concentration lemma. From the symmetry between h and t, the concentration Lemma is also applies for the partition function $Z_{c'} = \sum_{t \in \mathcal{V}} \left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)$.

B Proof of RelWalk Theorem

Let us consider the probabilistic event that $(1-\epsilon_z)Z \le Z_c \le (1+\epsilon_z)Z$ to be F_c and $(1-\epsilon_z)Z \le Z_{c'} \le (1+\epsilon_z)Z$ to be $F_{c'}$. From Lemma 1 we have $\Pr[F_c] \ge 1-\delta$. Then from the union bound we have,

$$\begin{aligned}\Pr[\bar{F}_c \cup \bar{F}_{c'}] &\le \Pr[\bar{F}_c] + \Pr[\bar{F}_{c'}] \\ &= 1 - \Pr[F_c] + 1 - \Pr[F_{c'}] \\ &= 2\delta.\end{aligned} \tag{29}$$

where $\bar{F}$ is the complement of event F. Moreover, let F be the probabilistic event that both F_c and $F_{c'}$ being True. Then from $\Pr[F] = 1 - \Pr[\bar{F}_c \cup \bar{F}_{c'}]$ we have, $\Pr[F] \ge 1 - 2\exp\left(-\Omega\left(\log^2 n\right)\right)$. The R.H.S. of (5) can be split into two parts T_1 and T_2 according to whether F happens or not.

$$\begin{aligned}p(h,t \mid R) = &\underbrace{\mathbb{E}_{\boldsymbol{c},\boldsymbol{c'}}\left[\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)}{Z_c}\frac{\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)}{Z_{c'}}\mathbf{1}_F\right]}_{T_1} \\ &+ \underbrace{\mathbb{E}_{\boldsymbol{c},\boldsymbol{c'}}\left[\frac{\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)}{Z_c}\frac{\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)}{Z_{c'}}\mathbf{1}_{\bar{F}}\right]}_{T_2}.\end{aligned} \tag{30}$$

Here, $\mathbf{1}_F$ and $\mathbf{1}_{\bar{F}}$ are indicator functions of the events F and $\bar{F}$ given as follows:

$$\mathbf{1}_F = \begin{cases} 1 & \text{if } F \text{ is True,} \\ 0 & \text{otherwise,} \end{cases} \tag{31}$$

$$\mathbf{1}_{\bar{F}} = \begin{cases} 0 & \text{if } F \text{ is True,} \\ 1 & \text{otherwise.} \end{cases} \tag{32}$$

Let us first show that T_2 is negligibly small.

For two real integrable functions $\psi_1(x)$ and $\psi_2(x)$ in $[a, b]$, the Cauchy-Schwarz's inequality states that

$$\left[\int_a^b \psi_1(x)\psi_2(x)dx\right]^2 \leq \int_a^b [\psi_1(x)]^2 dx \int_a^b [\psi_2(x)]^2 dx. \tag{33}$$

Applying (33) to T_2 in (30) we have:

$$\begin{aligned}
&\left(\mathbb{E}_{c,c'}\left[\frac{1}{Z_c Z_{c'}} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \mathbf{1}_{\bar{F}}\right]\right)^2 \\
&\leq \left(\mathbb{E}_{c,c'}\left[\frac{1}{Z_c^2} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbf{1}_{\bar{F}}\right]\right) \times \\
&\quad \left(\mathbb{E}_{c,c'}\left[\frac{1}{Z_{c'}^2} \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)^2 \mathbf{1}_{\bar{F}}\right]\right) \\
&= \left(\mathbb{E}_{c}\left[\frac{1}{Z_c^2} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]\right) \times \\
&\quad \left(\mathbb{E}_{c'}\left[\frac{1}{Z_{c'}^2} \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)^2 \mathbb{E}_{c|c'}\left[\mathbf{1}_{\bar{F}}\right]\right]\right)
\end{aligned} \tag{34}$$

Note that $Z_c \geq 1$ because Z_c is the sum of positive numbers and if $\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0$ for at least one of the $h \in \mathcal{V}$, then the total sum will be greater than 1. Therefore, by dropping Z_c term from the denominator we can further increase the first term in (34) as given by (35).

$$\begin{aligned}
&\mathbb{E}_c\left[\frac{1}{Z_c^2} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right] \\
&\leq \mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]
\end{aligned} \tag{35}$$

Let us split the expectation on the R.H.S. of (35) into two cases depending on whether $\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0$ or otherwise, indicated respectively by $\mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0)}$ and $\mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} \leq 0)}$.

$$\begin{aligned}
&\mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right] \\
&= \mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0)} \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right] \\
&+ \mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} \leq 0)} \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]
\end{aligned} \tag{36}$$

The second term of (36) is upper bounded by

$$\mathbb{E}_{c,c'}\left[\mathbf{1}_{\bar{F}}\right] \leq \exp\left(-\Omega(\log^2 n)\right) \tag{37}$$

The first term of (36) can be bounded as follows:

$$\begin{aligned}
&\mathbb{E}_c\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0)} \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right] \\
&\leq \mathbb{E}_c\left[\exp(\alpha \boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c})^2 \mathbf{1}_{(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} > 0)} \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right] \\
&\leq \mathbb{E}_c\left[\exp(\alpha \boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c})^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]
\end{aligned} \tag{38}$$

where $\alpha > 1$. Therefore, it is sufficient to bound $\mathbb{E}_c\left[\exp(\alpha \boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c})^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]$ when $||\boldsymbol{h}|| = \Omega(\sqrt{d})$.

Let us denote by z the random variable $2\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}$. Moreover, let $r(z) = \mathbb{E}_{c'|z}[\mathbf{1}_{\bar{F}}]$, which is a function of z between $[0, 1]$. We wish to upper bound $\mathbb{E}_c[\exp(z) r(z)]$. The worst-case $r(z)$ can be quantified using a continuous version of Abel's inequality (proved as Lemma A.4 in (Arora et al., 2015)), we can upper bound $\mathbb{E}_c\left[\exp(z) r(z)\right]$ as follows:

$$\mathbb{E}_c\left[\exp(z) r(z)\right] \leq \mathbb{E}\left[\exp(z) \mathbf{1}_{[t,+\infty]}(z)\right] \tag{39}$$

where t satisfies that $\mathbb{E}_c[\mathbf{1}_{[t,+\infty]}(z)] = \Pr[z \geq t] = \mathbb{E}_c[r(z)] \leq \exp(-\Omega(\log^2 n))$. Here, $\mathbf{1}_{[t,+\infty]}(z)$ is a function that takes the value 1 when $z \geq t$ and zero elsewhere. Then, we claim $\Pr_c[z \geq t] \leq \exp(-\Omega(\log^2 n))$ implies that $t \geq \Omega(\log^{.9} n)$.

If c was distributed as $\mathcal{N}(0, \frac{1}{d}\mathbf{I})$, this would be a simple tail bound. However, as c is distributed uniformly on the sphere, this requires special care, and the claim follows by applying the tail bound for the spherical distribution given by Lemma A.1 in (Arora et al., 2015) instead. Finally, applying Corollary A.3 in (Arora et al., 2015), we have:

$$\begin{aligned}
\mathbb{E}[\exp(z) r(z)] &\leq \mathbb{E}[\exp(z) \mathbf{1}_{[t,+\infty]}(z)] \\
&= \exp(-\Omega(\log^{1.8} n))
\end{aligned} \tag{40}$$

From a similar argument as above we can obtain the same bound for c' as well. Therefore, T_2 in (30) can be upper bounded as follows:

$$\begin{aligned}
&\mathbb{E}_{c,c'}\left[\frac{1}{Z_c Z_{c'}} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \mathbf{1}_{\bar{F}}\right] \\
&\leq \left(\mathbb{E}_c\left[\frac{1}{Z_c^2} \exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right)^2 \mathbb{E}_{c'|c}\left[\mathbf{1}_{\bar{F}}\right]\right]\right)^{1/2} \times \\
&\left(\mathbb{E}_{c'}\left[\frac{1}{Z_{c'}^2} \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)^2 \mathbb{E}_{c|c'}\left[\mathbf{1}_{\bar{F}}\right]\right]\right)^{1/2} \\
&\leq \exp(-\Omega(\log^{1.8} n))
\end{aligned} \tag{41}$$

Because $n = |\mathcal{V}|$, the size of the entity vocabulary, is large (ca. $n > 10^5$) in most knowledge graphs, we can ignore the T_2 term in (30).

Combining the above analysis of T_2 term with (30) we obtain an upper bound for $p(h, t \mid r)$ given by (42).

$$\begin{aligned}
p(h, t \mid R) \leq\ & (1 + \epsilon_z)^2 \frac{1}{Z^2} \mathbb{E}_{c,c'}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \mathbf{1}_F\right] \\
& + |\mathcal{D}| \exp(-\Omega(\log^{1.8} n)) \\
=\ & (1 + \epsilon_z)^2 \frac{1}{Z^2} \mathbb{E}_{c,c'}\left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right)\right] + \delta_0
\end{aligned} \tag{42}$$

where $|\mathcal{D}|$ is the number of relational tuples (h, r, t) in the KB and $\delta_0 = |\mathcal{D}| \exp(-\Omega(\log^{1.8} n)) \leq \exp(-\Omega(\log^{1.8} n))$ by the fact that $Z \leq \exp(2\kappa)n = O(n)$, where κ is the upper bound on $\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}$ and $\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}$, which is regarded as a constant.

On the other hand, we can lower bound $p(h, t \mid r)$ as given by (43).

$$
\begin{aligned}
p(h, t \mid R) &\geq (1 - \epsilon_z)^2 \frac{1}{Z^2} \mathbb{E}_{c,c'} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) 1_{\mathcal{F}} \right] \\
&\geq (1 - \epsilon_z)^2 \frac{1}{Z^2} \mathbb{E}_{c,c'} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right] \\
&\quad - |\mathcal{D}| \exp(-\Omega(\log^{1.8} n)) \\
&\geq (1 - \epsilon_z)^2 \frac{1}{Z^2} \mathbb{E}_{c,c'} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right] - \delta_0
\end{aligned} \tag{43}
$$

Taking the logarithm of both sides, from (42) and (43), the multiplicative error translates to an additive error given by (44).

$$
\begin{aligned}
\log p(h, t \mid R) &= \log \left(\mathbb{E}_{c,c'} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \mathbb{E}_{c'|c} \left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right] \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) A(c) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z)
\end{aligned} \tag{44}
$$

where $A(c) := \mathbb{E}_{c'|c} \left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right]$.

We assumed that $\boldsymbol{c}$ and $\boldsymbol{c'}$ are on the unit sphere and $\mathbf{R}_1$ and $\mathbf{R}_2$ to be orthogonal matrices. Therefore, $\mathbf{R}_1 \boldsymbol{c}$ and $\mathbf{R}_2 \boldsymbol{c'}$ are also on the unit sphere. Moreover, if we let the upper bound of the ℓ_2 norm of the entity embeddings to be $\kappa' \sqrt{d}$, then we have $||\boldsymbol{h}|| \leq \kappa' \sqrt{d}$ and $||\boldsymbol{t}|| \leq \kappa' \sqrt{d}$. Therefore, we have

$$
\langle \mathbf{R}_1 \boldsymbol{h}, \boldsymbol{c'} - \boldsymbol{c} \rangle \leq ||\boldsymbol{h}|| \, ||\boldsymbol{c} - \boldsymbol{c'}|| \leq \kappa' \sqrt{d} \, ||\boldsymbol{c} - \boldsymbol{c'}|| \tag{45}
$$

Then, we can upper bound $A(c)$ as follows:

$$
\begin{aligned}
A(c) &= \mathbb{E}_{c'|c} \left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c'}\right) \right] \\
&= \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \mathbb{E}_{c'|c} \left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 (\boldsymbol{c'} - \boldsymbol{c})\right) \right] \\
&\leq \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \mathbb{E}_{c'|c} \left[\exp\left(\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \\
&\leq (1 + \epsilon_2) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right)
\end{aligned} \tag{46}
$$

For some $\epsilon_2 > 0$. The last inequality holds because

$$
\begin{aligned}
&\mathbb{E}_{c|c'} \left[\exp\left(\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \\
&= \int \exp\left(\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) p(c'|c) dc' \\
&= \underbrace{\exp(\kappa' \sqrt{d})}_{\geq 1} \underbrace{\int \exp(||\boldsymbol{c} - \boldsymbol{c'}||) p(c'|c) dc'}_{\geq 1} \\
&= 1 + \epsilon_2
\end{aligned} \tag{47}
$$

To obtain a lower bound on $A(c)$ from the first-order Taylor approximation of $\exp(x) \geq 1 + x$ we observe that:

$$
\begin{aligned}
&\mathbb{E}_{c|c'} \left[\exp\left(\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \\
&+ \mathbb{E}_{c|c'} \left[\exp\left(-\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \geq 2.
\end{aligned} \tag{48}
$$

Therefore, from our model assumptions we have

$$
\mathbb{E}_{c|c'} \left[\exp\left(-\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \geq 1 - \epsilon_2 \tag{49}
$$

Hence,

$$
\begin{aligned}
A(c) &= \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \mathbb{E}_{c'|c} \left[\exp\left(\boldsymbol{t}^\top \mathbf{R}_2 (\boldsymbol{c'} - \boldsymbol{c})\right) \right] \\
&\geq \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \mathbb{E}_{c'|c} \left[\exp\left(-\kappa' \sqrt{d} \, ||\boldsymbol{c'} - \boldsymbol{c}||\right) \right] \\
&\geq (1 - \epsilon_2) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right)
\end{aligned} \tag{50}
$$

Therefore, from (47) and (50) we have

$$
A(c) = (1 \pm \epsilon_2) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \tag{51}
$$

Plugging $A(c)$ back in (44) results in $\log p(h, t \mid r)$ equal to:

$$
\begin{aligned}
&\log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) A(c) \right] \pm \delta_0 \right) - 2 \log Z + 2 \log(1 \pm \epsilon_z) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) (1 \pm \epsilon_2) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c}\right) \exp\left(\boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) + \log(1 \pm \epsilon_2) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\boldsymbol{h}^\top \mathbf{R}_1 \boldsymbol{c} + \boldsymbol{t}^\top \mathbf{R}_2 \boldsymbol{c}\right) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) + \log(1 \pm \epsilon_2) \\
&= \log \left(\mathbb{E}_{c} \left[\exp\left(\left(\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right)^\top \boldsymbol{c}\right) \right] \pm \delta_0 \right) \\
&\quad - 2 \log Z + 2 \log(1 \pm \epsilon_z) + \log(1 \pm \epsilon_2)
\end{aligned} \tag{52}
$$

Note that $\boldsymbol{c}$ has a uniform distribution over the unit sphere. In this case, from Lemma A.5 in (Arora et al., 2015), (53) holds approximately.

$$
\begin{aligned}
&\mathbb{E}_{c} \left[\exp\left(\left(\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right)^\top \boldsymbol{c}\right) \right] \\
&= (1 \pm \epsilon_3) \exp\left(\frac{||\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}||_2^2}{2d} \right)
\end{aligned} \tag{53}
$$

where $\epsilon_3 = \tilde{O}(1/d)$. Plugging (53) in (52) we have that

$$\log p(h, t \mid R) = \frac{\left|\left|\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right|\right|_2^2}{2d} + O(\epsilon_z) + O(\epsilon_2) + O(\epsilon_3) + O(\delta_0') - 2\log Z \tag{54}$$

where $\delta_0' = \delta_0$. $\left(\mathbb{E}_c\left[\exp\left((\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t})^\top \boldsymbol{c}\right)\right]\right)^{-1} = \exp(-\Omega(\log^{1.8} n))$. Therefore, δ_0' can be ignored. Note that $\epsilon_3 = \tilde{O}(1/d)$ and $\epsilon_z = \tilde{O}(1/\sqrt{n})$ by assumption. Therefore, we obtain that

$$\log p(h, t \mid R) = \frac{\left|\left|\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right|\right|_2^2}{2d} + O(\epsilon_z) + O(\epsilon_2) + \tilde{O}(1/d) - 2\log Z \tag{55}$$

C Learning with Multiple Negative Triples

This approach can be easily extended to learn from multiple negative triples as follows. Let us consider that we are given a positive triple, (h, R, t) and a set of K negative triples $\{(h_k', R, t_k')\}_{k=1}^K$. We would like our model to assign a probability, $p(h, t \mid R)$, to the positive triple that is higher than that assigned to any of the negative triples. This requirement can be written as (56).

$$p(h, t|R) \geq \max_{k=1,\ldots,K} p(h_k', t_k' \mid R) \tag{56}$$

We could further require the ratio between the probability of the positive triple and maximum probability over all negative triples to be greater than a threshold $\eta \geq 1$ to make the requirement of (56) to be tighter.

$$\frac{p(h, t \mid R)}{\max_{k=1,\ldots,K} p(h_k', t_k' \mid R)} \geq \eta \tag{57}$$

By taking the logarithm of (57) we obtain

$$\log p\left(h, t \mid R\right) - \log\left(\max_{k=1,\ldots,K} p\left(h_k', t_k' \mid R\right)\right) \geq \log(\eta) \tag{58}$$

Therefore, we can define the marginal loss for a misclassification as follows:

$$L\left(\left(h, R, t\right), \{(h_k', R, t_k')\}_{k=1}^K\right) = \max\left(0, \log\left(\max_{k=1,\ldots,K} p(h_k', t_k' \mid R)\right) + {} + \log\left(\eta\right) - \log p\left(h, t \mid R\right)\right) \tag{59}$$

However, from the monotonicity of the logarithm we have $\forall x_1, x_2 > 0$, if $\log(x_1) \geq \log(x_2)$ then $x_1 \geq x_2$. Therefore, the logarithm of the maximum can be replaced by the maximum of the logarithms in (59) as shown in (60).

Dataset	#R	#E	Train	Test	Val.
FB15K237	237	14,541	272,115	17,535	20,466
WN18RR	11	40,943	86,835	3,134	3,034
FB13	13	75,043	316,232	23,733	5,908

Table 4: Statistics of the datasets

$$L\left(\left(h, R, t\right), \{(h_k', R, t_k')\}_{k=1}^K\right) = \max\left(0, \max_{k=1,\ldots,K} \log\left(p\left(h_k', t_k' \mid R\right)\right) + \log\left(\eta\right) - \log p\left(h, t \mid R\right)\right) \tag{60}$$

By substituting (9) for the probabilities in (60) we obtain the rank-based loss given by (61).

$$L\left((h, R, t), \{(h_k', R, t_k')\}_{k=1}^K\right) = \max\left(0, 2d\log(\eta) + \max_{k=1,\ldots,K} \left|\left|\mathbf{R}_1{}^\top \boldsymbol{h}_k' + \mathbf{R}_2{}^\top \boldsymbol{t}_k'\right|\right|_2^2 - \left|\left|\mathbf{R}_1{}^\top \boldsymbol{h} + \mathbf{R}_2{}^\top \boldsymbol{t}\right|\right|_2^2\right) \tag{61}$$

In practice, we can use $p(h_k', t_k' \mid R)$ to select the negative triple with the highest probability for training with the positive triple.

D Training Details

The statistics of the benchmark datasets are shown in Table 4. We selected the initial learning rate (α) for SGD in $\{0.01, 0.001\}$, the regularisation coefficients (λ_1, λ_2) for the orthogonality constraints of relation matrices in $\{0, 1, 10, 100\}$. The number of randomly generated negative triples n_{neg} for each positive example is varied in $\{1, 10, 20, 50, 100\}$ and $d \in \{50, 100\}$. Optimal hyperparameter settings were: $\lambda_1 = \lambda_2 = 10$, $n_{\text{neg}} = 100$ for all the datasets, $\alpha = 0.001$ for FB15K237 and FB13, $\alpha = 0.01$ for WN18RR. For FB15K237 and WN18RR $d = 100$ was the best, whereas for FB13 $d = 50$ performed best. Negative triples are generated by replacing a head or a tail entity in a positive triple by a randomly selected entity and learn KGEs. We train the model until convergence or at most 1000 epochs over the training data where each epoch is divided into 100 mini-batches. The best model is selected by early stopping based on the performance of the learnt embeddings on the validation set (evaluated after each 20 epochs).

Few-shot Learning for Slot Tagging with Attentive Relational Network

Cennet Oguz[1] **Ngoc Thang Vu**[2]
[1]Multilinguality and Language Technology, German Research Center for Artificial Intelligence
[2]Institute for Natural Language Processing, University of Stuttgart
cennet.oguz@dfki.de thangvu@ims.uni-stuttgart.de

Abstract

Metric-based learning is a well-known family of methods for few-shot learning, especially in computer vision. Recently, they have been used in many natural language processing applications but not for slot tagging. In this paper, we explore metric-based learning methods in the slot tagging task and propose a novel metric-based learning architecture - Attentive Relational Network. Our proposed method extends relation networks, making them more suitable for natural language processing applications in general, by leveraging pretrained contextual embeddings such as ELMO and BERT and by using attention mechanism. The results on SNIPS data show that our proposed method outperforms other state of the art metric-based learning methods.

1 Introduction

Neural networks have been successfully utilized in natural language processing (NLP) applications with a large amount of hand-labeled data whereas they suffer a persistent challenge of low-resource. The approach of learning with few samples, known as few-shot learning - a branch of meta-learning (*learn to learn*) - has recently been popularized (Fei-Fei et al., 2006; Ravi and Larochelle, 2016; Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018) in computer vision. Recently, few-shot learning has also been applied to NLP tasks, e.g. natural language understanding (Dou et al., 2019), text classification (Jiang et al., 2018; Rios and Kavuluru, 2018; Gao et al., 2019; Geng et al., 2019), machine translation (Gu et al., 2018) and relation classification (Obamuyide and Vlachos, 2019).

In the slot tagging task, we aim at predicting task-specific values (e.g. artist, time) for slots (placeholders) in user utterances. Oguz and Vu (2020) propose a two-stage modeling approach to exploit domain-agnostic features to tackle low-resource domain challenges. Besides, the other state of the art techniques e.g. based on external memory (Peng and Yao, 2015), ranking loss (Vu et al., 2016), encoder (Kurata et al., 2016), and attention (Zhu and Yu, 2017) have achieved promising results with a wide range of neural networks methods.

However as many other NLP applications, the low-resource issue is a tremendous challenge for slot tagging in new domains, although labeled samples exist in related domains. Many studies have recently proposed to overcome this low-resource challenge using different techniques, e.g. multi-task modeling (Jaech et al., 2016), adversarial training (Kim et al., 2017), and pointer networks (Zhai et al., 2017). In addition, studies like zero-shot learning has influenced the studies of the domain scaling problem for slots prediction (Bapna et al., 2017), eliminating the need of labeled examples for transferring reusable concepts (Zhu and Yu, 2018; Lee and Jha, 2019), and conveying the domain-agnostic concepts between the intents (Shah et al., 2019) by exploiting label names and descriptions. Likewise, (Hou et al., 2020) use label semantics within a few-shot classification method TapNet (Yoon et al., 2019).

We suggest using a small amount of annotated samples from different domains as training input instead of slot descriptions and slot names as in previos zero-shot (Bapna et al., 2017; Lee and Jha, 2019; Shah et al., 2019) and few-shot (Hou et al., 2020) slot tagging studies for two reasons: (1) The creation of slot descriptions needs qualified linguistic expertise and is thus expensive. (2) The relationship between slot names and the corresponding tokens is not constant. To give an example, the relationship between the '*genre*' slot name and '*drama*' token is hypernymic whereas the relationship between the '*artist*' slot name and '*Tarkan*' token is instance based. Hence, it may not be valid to learn only one function to represent the different

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1566–1572
April 19 - 23, 2021.

relationships between names and tokens.

In this paper, we provide a new experimental design where the slot tagging task needs to be solved for unseen slot labels. The experimental design mimics previous few-shot learning studies (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018). Thus, the existing data sources from different domains are used to learn meta-knowledge, whereas unseen labels from low-resource domains are used to evaluate the models. Furthermore, we propose a novel modeling approach - *Attentive Relational Network*, inspired by (Sung et al., 2018; Jiang et al., 2018; Jetley et al., 2018), that leverages contextual embeddings such as ELMO and BERT and extends the previous relation networks (Sung et al., 2018) by learning to attend local and global features (Jetley et al., 2018). Experimental results on SNIPS data show that the proposed model outperforms other few-shot learning networks.

2 Methods

2.1 Input

FastText (Mikolov et al., 2018) is an approach to enrich the word vectors with a bag of character n-gram vectors.

ELMo (Peters et al., 2018) is a contextualized word representation methods. It concatenates the output of two LSTM independently trained on the bidirectional language modeling task and return the hidden states for the given input sequence.

BERT (Devlin et al., 2019) uses a bidirectional transformer model that is trained on a masked language modeling task. Because of WordPiece embeddings (Wu et al., 2016), there are different choices of presenting words. We use the first subtoken for representing the word as proposed in (Devlin et al., 2019). Additionally, due to the structure of multiple successive layers, i.e., 24 layers and as suggested in (Oguz and Vu, 2020), we select *10th*, *11th*, *12th*, and *13th* as the focused layers on local context (Clark et al., 2019; Tenney et al., 2018) for slot tagging.

2.2 Meta-learning strategy

Despite the fact that the proposed methods differ in their learning strategies, episode-based training is the same in meta-training and meta-testing phases for proposed meta models as mentioned in (Chen et al., 2018). For the purpose of applying episodic training in a robust way, we follow the proposed procedures in Vinyals et al. (2016); Snell et al. (2017). In the episodic training, each step - episode - is formed to compute gradients and update the model parameters. An episode consists of two components: support and query sets. To construct an episode, C unique classes are randomly sampled, and for each selected C unique classes K labeled examples randomly drawn for support $S = \{(x_i, y_i)\}_{i=1}^{m}$ and query $Q = \{(x_i, y_i)\}_{i=1}^{m}$ set, where $K > 1$ and $m = K * C$. The same episode composition strategy is applied in the meta-testing stage to evaluate the performance of the trained model over unseen classes.

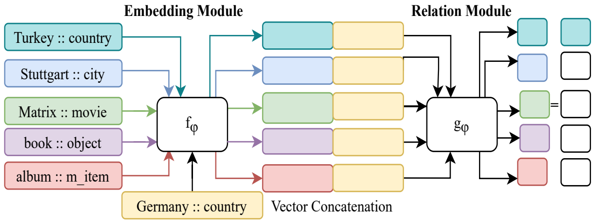


Figure 1: One-shot training example of slot tagging with Relation Network: Embedding Module extracts the feature vectors of each slot value whereas Relation Module calculates the relation scores between support samples and query. Then, the label of most relational value is assigned as a label of query.

Meta-training. The aim of this phase is to learn a meta learner that maps from a few labeled samples to a classifier. In each episode, meta-training employs a two-stage process: (1) the first stage implies producing the feature maps from the given input S and Q, called embedding function $f_\phi(x)$ (2) the second stage is to make prediction conditioned on few labeled examples, S. More formally, we define an episode E_{train} includes S and Q selected from train data,D_{train} . Then, the model is trained to minimize the label prediction error in the Q conditioned on S, i.e., $P_\theta(y_j|x_j, S)$, by utilizing the distance or relation metrics like $distance/relation(f_\phi(x_j), f_\phi(x_i))$, as also shown with an example in Figure 1.

Meta-testing. In this phase, we test the performance of trained meta-learner on unseen labels by following the same steps in meta-training phase. An episode E_{test} with S and Q is formed by randomly selecting from test data, D_{test}. The over all accuracy is computed by averaging the test episodes, $acc = \frac{1}{||E_{test}||} \sum_i E_{test}$.

We define $C = 5$ in meta-training and meta-testing stages except the meta-testing stage of *SearchCreativeW.* domain with $C = 3$ because *SearchCreativeW.* domain has only three slots. We

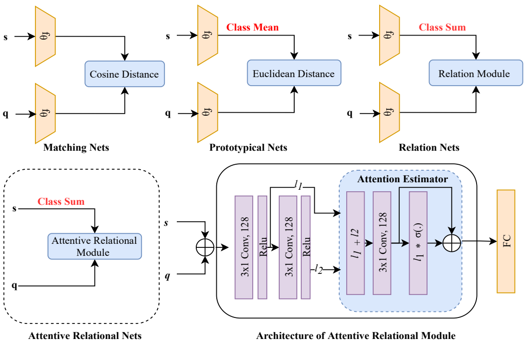


Figure 2: Few-shot learning models: Matching Nets compare each support sample with query in order to calculate the distance metric, Prototypical Nets rely on class mean of support set, and Relation Nets are based on class sum of support samples. f_θ represent the embedding module of each network and Attentive Relational Nets eject embedding module. Support sets are represented with s whereas q denotes the query.

train all the model within 10,000 episodes, and evaluate with 1000 test episode after every 500 steps of total train episodes.

2.3 Models

We focus on three *metric-based learning* few-shot learning methods such as Matching Networks (Vinyals et al., 2016), Prototypical Network (Snell et al., 2017), and Relation Network (Sung et al., 2018). Each network consists of two consecutive modules. The first module, called embedding function, focuses on the learning of the transferable embeddings for support and query samples. The second module is the classifier which identify the corresponding classes over the defined metric scores, e.g., distance and relation.

Matching Networks (MatchingNets) compare the cosine distance between the query feature and each support feature, and computes average cosine distance for each class.

Prototypical Networks (PrototypicalNets) compare the Euclidean distance between query features and the class mean of support features.

Relation Networks (RelationNets) propose a learnable non-linear relation module to output the relation scores over element-wise sum of each support and query features.

In one-shot scenario, MatchingNets and PrototypicalNets could be interpreted as identical, RelationNets differs with the relation module in order to calculate the relation score.

2.3.1 Attentive Relational Networks

We propose a novel metric-learning approach - Attentive Relational Networks (AttentiveRelationalNets) that highlight the relevant, and suppress the misleading between support and query samples. AttentiveRelationalNets address the few-shot classification problem by utilizing *learn to compare based on attention* insight. This can be seen as extending the strategy of Sung et al. (2018) to include a learnable attention module. A trainable attention module, inspired from Jetley et al. (2018), is added to incorporate the relation module of RelationNets. Besides, we make use of pretrained (contextual) embeddings since they have the proven strength on feature extraction for linguistics items instead of using embedding module, Figure 2.

For AttentiveRelationalNets, as shown in Figure 2, we implement two convolution blocks as it is in RelationNets with residual connection, as proposed in He et al. (2016).Then, the convolution blocks produce local descriptors, i.e., l_1 and l_2, as the output of activation function and pass them to the attention estimator in order to find the global g feature vector.

In order to compute the compatibility function, we define a convolution function with the input of two local features to an addition operation, $c = \langle u, l_1 + l_2 \rangle$. Here, u represents the universal set of features relevant to the s and q pairs in the object categories. We normalize the compatibility scores by using sigmoid operation, $a = \sigma(c)$. Then, the global feature vector is assessed by element-wise weighted average, i.e, $g = l_1 * a$. Afterwards, we concatenate the global features g with learned compatibility scores c as the input of the linear classifier which eventually produces a scalar in range of 0 to 1 representing the similarity between s and q , which is called relation score r. We define *mean square error*, as proposed in RelationNets, as the objective function of our model.

2.4 Evaluation

As we use the same implementation details for meta-training and meta-testing stages, we also evaluate the performance by few-shot classification accuracy following previous studies few-shot learning (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018) with a small change: since the meta-learning approaches are fast learning methods, we present the average accuracies of training epochs instead of presenting the best accuracy.

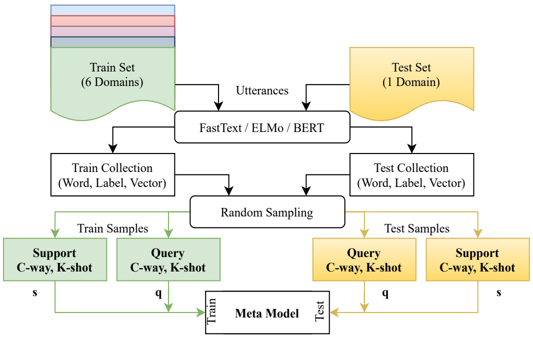


Figure 3: The schema of few-shot data construction for train and test episodes. 6 different domains are used for training phase whereas one different domain is used for test. Green path indicates the training support and query samples while yellow path represents the samples of test episodes. Train and test collections include N number of values of each slot. Random sampling function draws K samples for each of C slots.

3 Settings

3.1 Resources

In our study we address the few-shot learning approaches to recognize novel slot categories with very few examples from a new domain. In order to provide a deep experimental analysis of proposed networks and language models and to compare our model among each other, we set various experimental scenarios with different data and different K-shot sizes. Hence, we utilize the SNIPS dataset (Coucke et al., 2018) as a base dataset in our experiment. SNIPS is a SLU dataset of crowd-sourced user utterances with 39 slots and 7 intents. Thus, it is a well-categorized dataset which include tasks in domains, which makes the setup more realistic; learn to learn on a bunch of domains and test on new domains. We split SNIPS with the purpose of creating a single-domain dataset. We combine the originally divided training, testing, and development sets and separate them into domain sets in order to create new train and test data.

3.2 Few-shot Data Construction

Meta-learning models aim to learn from the training tasks, i.e. the train label space is disjoint with test label space and the trained model evaluated on unseen classes. Therefore, we utilize other domain data as the training set whereas the models are evaluated by using the current domain. Thus, we created 7 different sets contain a train which consists of 6 different domains as well as a test set which includes only one test domain.

As can be seen from Figure 3, we aggregate the six different domain data for the training set, whereas the remain one domain is used for testing, aiming at evaluating the performance of models on unseen classes per domain. Then, we convert the train and test sets to *train* and *test* collection that contain triplets in order to mimic the same data organization in the previous meta-learning studies. The triplet consists of three items: token, label, vector. The vector of the corresponding token is produced by using different (contextual) embeddings from randomly selected sentences for each label from the train and test set separately. Thus, the train collection is formed with the triplets from the different domain slots, whereas the test collection includes only the triplets of labels from the corresponding domain.

To investigate the efficiency of different models according to data availability in the few-shot setting, we experiment with three data collections sizes of slot values: 50, 100, and 200. Note that the collection size controls the total number of values that can be seen for each slot during training. In the meta-testing stage, we only use the test collection with size of 200 to be able to keep the comparative analyses under control. Furthermore, we examine different numbers of K shot, namely 5-shot, 10-shot, and 15-shot.

4 Results and Analysis

4.1 Different Network Architectures

Table 1 shows the performance in comparison with state of the art metric-based learning models on slot tagging task with different (contextual) embeddings. As can be seen from the scores, RelationNets with ELMo embeddings constantly give the best results, whereas MatchingNets present the lowest scores for each embedding variance. We assume that learning from the distance or relation scores with individual support and query samples instead of the class sum, i.e., as it is in PrototypicalNets, or class mean, i.e., as it is in RelationNets, decrease the learning performance.

Furthermore, Table 2 presents the results from AttentiveRelationalNets with different embeddings methods and demonstrates that our proposed model with ELMo and BERT outperforms the previous models consistently. Additionally, AttentiveRelationalNets significantly improve the results with BERT from the previous experiments in Table 1. When the success of AttentiveRelationalNets is

Table 1: Few-shot slot tagging on SNIPS data. Results are accuracies averaged of three different slot value sizes (50, 100, and 200) with different Ks (5, 10, and 15).

	FastText									ELMo									BERT								
	MatchingNet			PrototypicalNet			RelationNet			MatchingNet			PrototypicalNet			RelationNet			MatchingNet			PrototypicalNet			RelationNet		
Domain / K-shot	5	10	15	5	10	15	5	10	15	5	10	15	5	10	15	5	10	15	5	10	15	5	10	15	5	10	15
AddToPlaylist	20.4	20.5	20.5	67.4	67.6	67.6	65.9	70.4	71.7	65.4	71.7	73.9	72.1	71.8	71.1	70.0	75.1	**76.3**	65.9	70.6	73.2	70.8	71.5	71.4	69.9	74.0	74.3
PlayMusic	20.9	20.8	20.7	68.9	68.6	69.1	68.8	72.2	73.7	67.3	73.7	75.9	73.5	72.2	71.5	72.2	76.5	**78.0**	64.8	70.9	73.3	67.4	68.2	67.7	67.7	71.6	73.0
BookRestaurant	23.5	23.8	24.0	75.0	74.5	74.3	78.7	82.5	83.6	71.3	76.8	78.9	81.2	79.5	77.5	82.9	85.9	**86.5**	68.4	72.7	74.8	75.8	76.5	76.8	78.4	81.8	83.7
GetWeather	20.7	25.7	20.7	76.6	75.9	75.7	79.5	84.2	85.3	75.6	82.2	84.8	80.1	78.5	77.9	79.8	85.1	**87.0**	75.3	81.7	84.3	78.5	78.9	79.1	83.6	86.2	85.8
RateBook	34.5	35.1	35.2	87.4	88.3	87.2	89.8	92.2	93.9	83.8	89.3	90.6	90.2	90.2	89.6	90.0	93.9	**95.1**	88.0	92.0	93.7	92.8	93.6	93.2	92.0	94.1	93.9
SearchCreativeW.	37.8	37.9	38.2	78.1	78.2	78.2	82.8	85.9	86.3	82.8	86.4	88.6	85.8	85.7	83.8	89.7	92.1	**92.9**	79.7	84.4	87.1	81.4	82.0	81.8	86.0	90.3	91.3
FindScreeningE.	21.2	21.3	21.4	73.4	73.5	73.7	78.8	83.1	84.6	80.7	87.7	90.9	83.5	81.6	81.0	86.8	89.9	**91.7**	78.6	86.4	89.5	78.0	78.1	78.3	81.3	86.8	87.9

Table 2: Few-shot slot tagging on SNIPS data with Attentive Relational Networks. Results are accuracies averaged of three different slot value sizes (50, 100, and 200) with different Ks (5, 10, and 15).

	FastText			ELMo			BERT		
Domain	5	10	15	5	10	15	5	10	15
AddToPlaylist	63.6	66.1	68.3	71.9	75.8	77.7	72.6	76.6	**78.5**
PlayMusic	68.5	71.8	73.2	74.7	77.7	79.2	72.8	77.0	**78.7**
BookRestaurant	78.9	82.0	83.1	84.2	87.2	**87.6**	82.7	86.0	87.5
GetWeather	79.0	82.7	84.5	83.3	87.1	88.5	84.9	89.5	**90.2**
RateBook	88.1	90.9	91.4	92.3	94.8	95.6	94.8	96.5	**96.7**
SearchCreativeW.	81.7	84.4	85.1	90.8	93.0	**93.2**	88.2	91.5	92.6
FindScreeningE.	76.9	81.2	82.1	87.3	90.8	**92.3**	82.9	87.0	88.3

examined extensively, it seems that our proposed model gives better results on slot labels that categorize common nouns, while provides relative competitive results for proper nouns. Note that proper nouns, in general, seem to be challenging for all setups. In addition, RelationalNets outperform AttentiveRelationalNets with FastText along with the explanation that embedding function is still effective with average embeddings. However, as opposed to RelationNets, Attentive Relational Networks do better classification overall because of trainable attention. Since trainable attention highlights the relevant features between the slot values labeled with the same slot, whereas it suppresses the misleading them. In an other word, slot local features are able to be more informative for the model while the global features are suppressed.

4.2 Different Contextual Embeedings

Table 2 shows that ELMo and BERT have comparable performance, with BERT slightly better on most tasks: ELMo, however, scores higher on *FindScreeningE.* consistently with AttentiveRelationalNets and all different Ks. Although embedding modules are presented as a feature extraction method for inputs according to distance or relational score, the significant performance gap between FastText and contextualized embeddings shows that the contextualized features outperform the embedding module of few-shot classification models. On the other hand, when we compare FastText embeddings with contextualized word vectors in Table 2, the lower results can be seen. Additionally, when FastText features are compared between the results of RelationNets and AttentiveRelationalNets, we observe that RelationNets outperform AttentiveRelationalNets.

We further look at the wrong predictions in order to understand the reason for the success of ELMo on *FindScreeningE.* and observe that ELMo shows high performance on the labels of proper nouns such as city and location and the labels like *object_type* and *movie_type*. However, BERT demonstrates high performance on proper nouns such as artist, album, hence it outperforms ELMo on *AddToPlaylist* and *PlayMusic* domains. In addition, the improvement of BERT with AttentiveRelationalNets mostly relies on the increase of the accuracy of overall labels, but especially the slot labels that contain common nouns.

4.3 Different Collection and Shot Sizes

AttentiveRelationalNets demonstrate a linear correlation between the increase of performance and the increase of collection size. In addition, the increase of shot size mostly shows improvement in overall results, apart from PrototypicalNets which show their highest result with 10-shot.

5 Conclusion

We presented a deep analysis with a wide variety of few-shot learning methods and pretrained (contextual) embeddings for slot tagging. Furthermore, we proposed a novel architecture that leverages attention mechanism attending both, local and global features of given support samples. Experimental results on SNIPS dataset show that a) pretrained contextual embeddings contributed to high performance and b) our proposed approach consistently outperformed other methods in all setups.

References

Ankur Bapna, Gokhan Tür, Dilek Hakkani-Tür, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *Proc. Interspeech 2017*, pages 2476–2480.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. 2018. A closer look at few-shot classification. In *International Conference on Learning Representations*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv*, abs/1805.10190.

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

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6407–6414.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3904–3913, Hong Kong, China. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. *arXiv preprint arXiv:2006.05702*.

Aaron Jaech, Larry Heck, and Mari Ostendorf. 2016. Domain adaptation of recurrent neural networks for natural language understanding. *Interspeech 2016*, pages 690–694.

Saumya Jetley, Nicholas A Lord, Namhoon Lee, and Philip HS Torr. 2018. Learn to pay attention. In *International Conference on Learning Representations*.

Xiang Jiang, Mohammad Havaei, Gabriel Chartrand, Hassan Chouaib, Thomas Vincent, Andrew Jesson, Nicolas Chapados, and Stan Matwin. 2018. Attentive task-agnostic meta-learning for few-shot text classification.

Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017. Adversarial adaptation of synthetic or stale data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1297–1307.

Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. Leveraging sentence-level information with encoder lstm for semantic slot filling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2077–2083.

Sungjin Lee and Rahul Jha. 2019. Zero-shot adaptive transfer for conversational language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6642–6649.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Abiola Obamuyide and Andreas Vlachos. 2019. Model-agnostic meta-learning for relation classification with limited supervision. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5873–5879.

Cennet Oguz and Ngoc Thang Vu. 2020. A two-stage model for slot filling in low-resource settings: Domain-agnostic non-slot reduction and pretrained contextual embeddings. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 73–82.

Baolin Peng and Kaisheng Yao. 2015. Recurrent neural networks with external memory for language understanding. *ArXiv*, abs/1506.00195.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 3132. NIH Public Access.

Darsh Shah, Raghav Gupta, Amir Fayazi, and Dilek Hakkani-Tur. 2019. Robust zero-shot cross-domain slot filling with example values. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5484–5490.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2018. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6060–6064. IEEE.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144.

Sung Whan Yoon, Jun Seo, and J. Moon. 2019. Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In *ICML*.

Feifei Zhai, Saloni Potdar, Bing Xiang, and Bowen Zhou. 2017. Neural models for sequence chunking. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Su Zhu and Kai Yu. 2017. Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5675–5679. IEEE.

Su Zhu and Kai Yu. 2018. Concept transfer learning for adaptive language understanding. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 391–399.

SpanEmo: Casting Multi-label Emotion Classification as Span-prediction

Hassan Alhuzali **Sophia Ananiadou**
National Centre for Text Mining
Department of Computer Science, The University of Manchester, United Kingdom
{hassan.alhuzali@postgrad.,sophia.ananiadou@}manchester.ac.uk

Abstract

Emotion recognition (ER) is an important task in Natural Language Processing (NLP), due to its high impact in real-world applications from health and well-being to author profiling, consumer analysis and security. Current approaches to ER, mainly classify emotions independently without considering that emotions can co-exist. Such approaches overlook potential ambiguities, in which multiple emotions overlap. We propose a new model "SpanEmo" casting multi-label emotion classification as span-prediction, which can aid ER models to learn associations between labels and words in a sentence. Furthermore, we introduce a loss function focused on modelling multiple co-existing emotions in the input sentence. Experiments performed on the SemEval2018 multi-label emotion data over three language sets (i.e., English, Arabic and Spanish) demonstrate our method's effectiveness. Finally, we present different analyses that illustrate the benefits of our method in terms of improving the model performance and learning meaningful associations between emotion classes and words in the sentence[1].

1 Introduction

Emotion is essential to human communication, thus emotion recognition (ER) models have a host of applications from health and well-being (Alhuzali and Ananiadou, 2019; Aragón et al., 2019; Chen et al., 2018) to consumer analysis (Alaluf and Illouz, 2019; Herzig et al., 2016) and user profiling (Volkova and Bachrach, 2016; Mohammad and Kiritchenko, 2013), among others. Interest in this area has given rise to new NLP approaches aimed at emotion classification, including single-label and multi-label emotion classification. Most existing approaches for multi-label emotion classification (Ying et al., 2019; Baziotis et al., 2018; Yu et al., 2018; Badaro et al., 2018; Mulki et al., 2018; Mohammad et al., 2018; Yang et al., 2018) do not effectively capture emotion-specific associations, which can be useful for prediction, as well as learning of association between emotion labels and words in a sentence. In addition, standard approaches in emotion classification treat individual emotion independently. However, emotions are not independent; a specific emotive expression can be associated with multiple emotions. The existence of association/correlation among emotions has been well-studied in psychological theories of emotions, such as Plutchik's wheels of emotion (Plutchik, 1984) that introduces the notion of mixed and contrastive emotions. For example, "joy" is close to "love" and "optimism", instead of "anger" and "sadness".

#	Sentence	GT
S1	well my day started off great the mocha machine wasn't working @ mcdonalds.	anger, disgust, joy, sadness
S2	I'm doing all this to make sure you smiling down on me bro.	joy, love, optimism

Table 1: Example Tweets from SemEval-18 Task 1. GT represents the ground truth labels.

Consider S1 in Table 1, which contains a mix of positive and negative emotions, although it is more negative oriented. This can be observed clearly via the ground truth labels assigned to this example, where the first part of this sentence only expresses a positive emotion (i.e., joy), while the other part expresses negative emotions. For example, clue words like "great" are more likely to be associated with "joy", whereas "wasn't working" are more likely to be associated with negative emotions. Learning such associations between emotion labels and

---

[1] Source code is available at `https://github.com/hasanhuz/SpanEmo`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1573–1584
April 19 - 23, 2021.

words in a sentence can help ER models to predict the correct labels. S2 further highlights that certain emotions are more likely to be associated with each other. Mohammad and Bravo-Marquez (2017) also observed that negative emotions are highly associated with each other, while less associated with positive emotions. Based on these observations, we seek to answer the following research questions: i) how to enable ER models to learn emotion-specific associations by taking into account label information and ii) how to benefit from the multiple co-existing emotions in a multi-label emotion data set with the intention of learning label correlations. Our contributions are summarised as follows:

I. a novel framework casting the task of multi-label emotion classification as a span-prediction problem. We introduce "SpanEmo" to train the model to take into consideration both the input sentence and a label set (i.e., emotion classes) for selecting a span of emotion classes in the label set as the output. The objective of SpanEmo is to predict emotion classes directly from the label set and capture associations corresponding to each emotion.

II. a loss function, modelling multiple co-existing emotions for each input sentence. We make use of the label-correlation aware loss (LCA) (Yeh et al., 2017), originally introduced by Zhang and Zhou (2006). The objective of this loss function is to maximise the distance between positive and negative labels, which is learned directly from the multi-label emotion data set.

III. a large number of experiments and analyses both at the word- and sentence-level, demonstrating the strength of SpanEmo for multi-label emotion classification across three languages (i.e. English, Arabic and Spanish).

The rest of the paper is organised as follows: section 2 describes our methodology, while section 3 discusses experimental details. We evaluate the proposed method and compare it to related methods in section 4. Section 5 reports on the analysis of results, while section 6 reviews related work. We conclude in section 7.

2 Methodology

2.1 Framework

Figure 1 presents our framework (SpanEmo). Given an input sentence and a set of classes, a base encoder was employed to learn contextualised word representations. Next, a feed forward network (FFN) was used to project the learned representations into a single score for each token. We then used the scores for the label tokens as predictions for the corresponding emotion label. The green boxes at the top of the FFN illustrate the positive label set, while the red ones illustrate the negative label set for multi-label emotion classification. We now turn to describing our framework in detail.

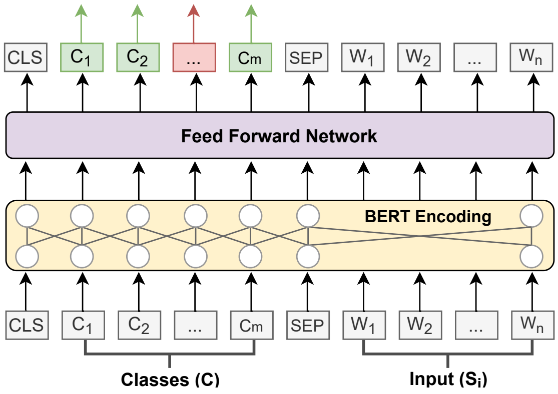


Figure 1: Illustration of our proposed framework (SpanEmo).

2.2 Our Method (SpanEmo)

Let $\{(s_i, y_i)\}_{i=1}^{N}$ be a set of N examples with the corresponding emotion labels of C classes, where s_i denotes the input sentence and $y_i \in \{0,1\}^m$ represents the label set for s_i. As shown in Figure 1, both the label set and the input sentence were passed into the encoder BERT (Devlin et al., 2019). The encoder received two segments: the first corresponds to the set of emotion classes, while the second refers to the input sentence. The hidden representations ($H_i \in R^{T \times D}$)[2] for each input sentence and the label set were obtained as follows:

$$\mathrm{H}_i = \mathrm{Encoder}([\mathrm{CLS}] + |\mathrm{C}| + [\mathrm{SEP}] + \mathrm{s}_i), \quad (1)$$

where {[CLS], [SEP]} are special tokens and |C| denotes the size of emotion classes. Feeding both segments to the encoder has a few advantages. First, the encoder can interpolate between emotion classes and all words in the input sentence. Second, a hidden representation is generated both for words and emotion classes, which can be further used to understand whether the encoder can learn association between the emotion classes and words in the input sentence. Third, SpanEmo is flexible because its predictions are directly produced from the first segment corresponding to the emotion classes.

---

[2] T and D denote the input length and dimensional size, respectively.

We further introduced a feed-forward network (FFN) consisting of a non-linear hidden layer with a Tanh activation ($f_i(\text{H}_i)$) as well as a position vector $p_i \in R^D$, which was used to compute a dot product between the output of f_i and p_i. As our task involved a multi-label emotion classification, we added a sigmoid activation to determine whether class$_i$ was the correct emotion label or not. It should be mentioned that the use of the position vector is quite similar to how start and end vectors are defined in transformer-based models for question-answering. Finally, the span-prediction tokens were obtained from the label segment and then compared with the ground truth labels since there was a 1-to-1 correspondence between the label tokens and the original emotion labels.

$$\hat{\mathbf{y}} = \text{sigmoid}(\text{FFN}(\mathbf{H}_i)), \tag{2}$$

2.3 Label-Correlation Aware (LCA) Loss

Following Yeh et al. (2017), we employed the label-correlation aware loss, which takes a vector of true-binary labels (y), as well as a vector of probabilities ($\hat{y}$), as input:

$$\mathcal{L}_{\text{LCA}}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{|\mathbf{y}^0||\mathbf{y}^1|} \sum_{(p,q)\in \mathbf{y}^0 \times \mathbf{y}^1} \exp(\hat{\mathbf{y}}_p - \hat{\mathbf{y}}_q), \tag{3}$$

where y^0 denotes the set of negative labels, while y^1 denotes the set of positive labels. $\hat{y}_p$ represents the p^{th} element of vector $\hat{y}$. The objective of this loss function is to maximise the distance between positive and negative labels by implicitly retaining the label-dependency information. In other words, the model should be penalised when it predicts a pair of labels that should not co-exist for a given example.

2.4 Training Objective

To model label-correlation, we combined LCA loss with binary cross-entropy (BCE) and trained them jointly. This aimed to help the LCA loss to focus on maximising the distance between positive and negative label sets, while at the same time taking advantage of the BCE loss through maximising the probability of the correct labels. We experimentally observed that training our approach jointly with those two loss functions produced the best results. The overall training objective was computed as follows:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{\text{BCE}} + \alpha \sum_{i=1}^{M} \mathcal{L}_{\text{LCA}}, \tag{4}$$

where $\alpha \in [0, 1]$ denotes the weight used to control the contribution of each part to the overall loss.

3 Experiments

3.1 Implementation Details

We used PyTorch (Paszke et al., 2017) for implementation and ran all experiments on an Nvidia GeForce GTX 1080 with 11 GB memory. We also trained BERT$_{\text{BASE}}$ utilising the open-source Hugging-Face implementation (Wolf et al., 2019). For experiments related to Arabic, we chose "bert-base-arabic" developed by Safaya et al. (2020), while selecting "bert-base-spanish-uncased" developed by Cañete et al. (2020) for Spanish. All three models were trained on the same hyper-parameters with a fixed initialisation seed, including a feature dimension of 786, a batch size of 32, a dropout rate of 0.1, an early stop patience of 10 and 20 epochs. Adam was selected for optimisation (Kingma and Ba, 2014) with a learning rate of $2e$-5 for the BERT encoder, and a learning rate of $1e$-3 for the FFN. It should be mentioned that we tuned our method only on the validation set and further report on the analysis of the effect of parameter α in section 5.4. Table 2 summarises the hyper-parameters used in our experiments.

Parameter	**Value**
Feature dimension	768
Batch size	32
Dropout	0.1
Early stop patience	10
Number of epochs	20
lr-BERT	$2e$-5
lr-FFN	$1e$-3
Optimiser	Adam
Alpha (α)	0.2

Table 2: Hyper-parameter values. lr: refers to the Learning rate.

3.2 Data Set and Task Settings

In this work, we chose semEval2018 (Mohammad et al., 2018) for our multi-label emotion classification, which is based on labelled data from tweets in English, Arabic and Spanish. The data was originally partitioned into three sets: training set (Train), validation set (Valid) and test (Test) set. Following the metrics in Mohammad et al. (2018), we run our experiments on micro F1-score, macro F1-score and Jaccard index score[3]. Table 3 presents the summary of all three sets for each language, including the number of instances in the train, valid and test sets. In addition, the number of emotion classes and the percentage of instances with varying numbers of classes (co-existing) are included. It is worth noting that these percentages do not include the neutral instances.

Info./Lang.	**English**	**Arabic**	**Spanish**
Train (#)	6,838	2,278	3,561
Valid (#)	886	585	679
Test (#)	3,259	1,518	2,854
Total (#)	10,983	4,381	7,094
Classes (#)	11	11	11
1 co.emo (%)	14.36	21.38	39.11
2 co.emo (%)	40.55	39.03	42.15
3 co.emo (%)	30.92	29.85	12.76

Table 3: Data Statistics. co.emo: refers to the percentage of co-existing emotions.

To pre-process the data, we utilised a tool designed for the specific characteristics of Twitter, i.e., misspellings and abbreviations (Baziotis et al., 2017). The tool offers different functionalities, such as tokenisation, normalisation, spelling correction, and segmentation. We used the tool to tokenise the text, convert words to lower case, normalise user mentions, urls and repeated-characters.

3.3 Multi-label Emotion Classification

We compared the performance of SpanEmo to some baseline as well as state-of-the-art models on all three languages. For experiments related to English, we selected seven models, while we chose three models for both Arabic and Spanish. We also include the results of $\text{BERT}_{\text{BASE}}$.

3.3.1 English

English models include JBNN (He and Xia, 2018), DATN (Yu et al., 2018), NTUA (Baziotis et al., 2018), RERc (Zhou et al., 2018), $\text{BERT}_{\text{BASE}}$+DK (Ying et al., 2019), $\text{BERT}_{\text{BASE}}$-GCN (Xu et al., 2020) and LEM (Fei et al., 2020). JBNN introduces a joint binary neural network, which focuses on learning the relations between emotions based on the theory of Plutchik's wheel of emotions (Plutchik, 1980), and then performing multi-label emotion classification via integrating these label relations into the loss function. DATN proposes a dual attention transfer network to improve multi-label emotion classification with the help of sentiment classification, while NTUA is ranked the top-1 model of the SemEval2018 competition as it relies on different pre-training and fine-tuning strategies. RERc defines a ranking emotion relevant loss focused on incorporating emotion relations into the loss function to improve both emotion prediction and rankings of relevant emotions. Both $\text{BERT}_{\text{BASE}}$+DK and $\text{BERT}_{\text{BASE}}$-GCN utilise the same encoder as our own with the former considering additional domain knowledge (DK) and the latter capturing emotion relations through Graph Convolutional Network (GCN), respectively. LEM introduces a latent emotion memory network, in which the latent emotion module learns emotion distribution via a variational autoencoder, while the memory module captures features corresponding to each emotion.

3.3.2 Arabic

Arabic models consist of EMA (Badaro et al., 2018), Tw-StAR (Mulki et al., 2018) and HEF (Alswaidan and Menai, 2020). EMA is the best performing model from the SemEval2018 competition on this set. It utilises various pre-processing steps (e.g. diacritics removal, normalisation, emojis transcription and stemming), as well as different classification algorithms. The Tw-StAR model applies some pre-processing steps and then uses TF-IDF to learn features of a Support Vector Machine. HEF is based on a hybrid neural network, including different word embeddings (e.g. Word2Vec, Glove, FastText) plus variations of RNN neural networks, such as Long Short-Term Memory and Gated Recurrent Unit.

3.3.3 Spanish

Spanish models comprise Tw-StAR (Mulki et al., 2018), ELiRF (González et al., 2018) and MILAB (Mohammad et al., 2018). The ELiRF model applies some pre-processing steps while also adapting the tweet tokeniser for Spanish tweets. MILAB is the best performing model from the SemEval2018 shared-task on this set.

4 Results

Table 4 presents the performance of our proposed approach (SpanEmo) on all three languages, in terms of micro F1-score (miF1), macro F1-score (maF1) and Jaccard index score (jacS), and compares it to the baseline and state-of-the-art models

---

[3] jacS is defined as the size of the intersection divided by the size of the union of the true label set and predicted label set.

discussed in section 3.3.

Language	**English**		
Model/Metric	**miF1**	**maF1**	**jacS**
JBNN	0.632	0.528	-
RERc	0.651	0.539	-
DATN	-	0.551	0.583
NTUA	0.701	0.528	0.588
$\text{BERT}_{\text{base}}$	0.695	0.520	0.570
$\text{BERT}_{\text{base}}$+DK	**0.713**	0.549	0.591
$\text{BERT}_{\text{base}}$-GCN	0.707	0.563	0.589
LEM	0.675	0.567	-
SpanEmo (ours)	**0.713**	**0.578**	**0.601**
	Arabic		
	miF1	**maF1**	**jacS**
Tw-StAR	0.597	0.446	0.465
EMA	0.618	0.461	0.489
$\text{BERT}_{\text{base}}$	0.650	0.477	0.523
HEF	0.631	0.502	0.512
SpanEmo (ours)	**0.666**	**0.521**	**0.548**
	Spanish		
	miF1	**maF1**	**jacS**
Tw-StAR	0.520	0.392	0.438
ELiRF	0.535	0.440	0.458
MILAB	0.558	0.407	0.469
$\text{BERT}_{\text{base}}$	0.596	0.474	0.487
SpanEmo (ours)	**0.641**	**0.532**	**0.532**

Table 4: The results of multi-label emotion classification on SemEval-2018 test set.

As shown in Table 4, our method outperformed all models on all languages, as well as on almost all metrics, with a marginal improvement of up to 1-1.3% for English, 1.9-3.6% for Arabic and 6.3-9.2% for Spanish. This demonstrates the utility and advantages of SpanEmo, as well as the label-correlation aware loss for improving the performance of multi-label emotion classification in English, Arabic and Spanish.

Based on the empirical results reported in Table 4, the following observations can be made. First, incorporating the relations between emotions into the models tends to lead to higher performance, especially for macro F1-score. For example, both DATN and LEM learn emotion-related features and achieve better performance than NTUA and $\text{BERT}_{\text{BASE}}$+DK. Additionally, ELiRF makes use of various sentiment/emotion features (i.e., learned from lexica) and it yielded the best performance among the three compared models. This corroborates our earlier hypothesis that learning emotion-specific associations is crucial for improving the performance. Although $\text{BERT}_{\text{BASE}}$+DK adopts the same encoder as our own and adds domain knowledge, our method still performs strongly, especially for both macro F1- and jaccard score with a marginal improvement of up to 2.9% and 1%, respectively. In short, capturing emotion-specific associations as well as integrating the relations between emotions into the loss function, helped SpanEmo to achieve the best results compared with all models on almost all metrics.

4.1 Ablation Study

To understand the effect of our framework, we undertook an ablation study of the model performance under three settings: firstly, the model was trained only with BCE loss; secondly, it was trained only with LCA loss; and thirdly it was trained without the label segment. The third setting is equivalent to training the model as a simple multi-label classification task, by only considering the input sentence.

Language	**English**		
Model/Metric	**miF1**	**maF1**	**jacS**
SpanEmo (joint)	**0.713**	0.578	**0.601**
- $\mathcal{L}$ (LCA)	0.712	0.564	0.590
- $\mathcal{L}$ (BCE)	0.698	**0.583**	0.582
- Label Seg.	0.695	0.520	0.570
	Arabic		
	miF1	**maF1**	**jacS**
SpanEmo (joint)	**0.666**	0.521	**0.548**
- $\mathcal{L}$ (LCA)	0.654	0.481	0.534
- $\mathcal{L}$ (BCE)	0.660	**0.526**	0.532
- Label Seg.	0.650	0.477	0.523
	Spanish		
	miF1	**maF1**	**jacS**
SpanEmo (joint)	**0.641**	0.532	**0.532**
- $\mathcal{L}$ (LCA)	0.629	0.526	0.507
- $\mathcal{L}$ (BCE)	0.606	**0.544**	0.499
- Label Seg.	0.596	0.474	0.487

Table 5: Ablation experiment results. The second and third rows from each group correspond to the removal of the respective loss function, whereas the last row corresponding to the removal of the label segment.

Table 5 presents the results. When SpanEmo was trained without the LCA loss, the results dropped

Model/Metric		**miF1**	**maF1**	**jacS**	**miF1**	**maF1**	**jacS**	**miF1**	**maF1**	**jacS**
English	$\mathcal{L}$	≥ 1 co.emo			≥ 2 co.emo			≥ 3 co.emo		
$\text{BERT}_{\text{base}}$	BCE	0.703	0.515	0.587	0.712	0.521	0.596	0.692	0.509	0.554
SpanEmo	BCE	0.716	0.563	0.599	0.737	0.578	0.629	0.748	0.597	0.639
SpanEmo	Joint	**0.724**	**0.590**	**0.613**	**0.746**	**0.606**	**0.648**	**0.753**	**0.624**	**0.643**
Arabic	$\mathcal{L}$	≥ 1 co.emo			≥ 2 co.emo			≥ 3 co.emo		
$\text{BERT}_{\text{base}}$	BCE	0.656	0.459	0.527	0.668	0.471	0.531	0.682	0.485	0.555
SpanEmo	BCE	**0.689**	0.518	**0.565**	0.709	0.536	0.586	0.745	0.567	**0.629**
SpanEmo	Joint	**0.689**	**0.534**	**0.565**	**0.710**	**0.551**	**0.587**	**0.746**	**0.584**	0.626
Spanish	$\mathcal{L}$	≥ 1 co.emo			≥ 2 co.emo			≥ 3 co.emo		
$\text{BERT}_{\text{base}}$	BCE	0.603	0.476	0.526	0.567	0.461	0.441	0.518	0.432	0.364
SpanEmo	BCE	0.653	0.528	0.561	0.646	0.528	0.519	**0.663**	0.566	**0.508**
SpanEmo	Joint	**0.662**	**0.565**	**0.581**	**0.655**	**0.568**	**0.530**	0.644	**0.570**	0.490

Table 6: Presenting the number of co-existing emotion classes. The best results in each language group are marked in bold.

by 1-2% for macro F1- and jaccard score. In addition, the results of SpanEmo dropped by 1-2% for two metrics apart from the macro F1-score when trained without the BCE loss. However, the removal of the label segment led to a much higher drop of 3-6%. The same patterns were also observed in the Arabic and Spanish experiments. This supports our earlier hypothesis that casting the task of multi-label emotion classification as span-prediction is beneficial for improving both the representation and performance of multi-label emotion classification.

5 Analysis

5.1 Prediction of Multiple Emotions

We additionally validated the effectiveness of our method for learning the multiple co-existing emotions on English, Arabic and Spanish sets. Table 6 presents the results, including $\text{BERT}_{\text{BASE}}$. SpanEmo demonstrated a strong ability to handle multi-label emotion classification much better than $\text{BERT}_{\text{BASE}}$. Since $\text{BERT}_{\text{BASE}}$ is trained only with binary cross-entropy (BCE) loss, here we include the results of our method trained only with this loss function. SpanEmo still achieved consistent improvement as the number of co-existing emotions increases, showing the usefulness of our method in learning multiple emotions. Improvement can clearly be observed for English and Arabic experiments, but not as much for Spanish. This may be attributed to the high percentage of single-label data, which is around (40%) for Spanish, while it is lower than that for both English and Arabic. Obviously, SpanEmo can be used without LCA loss, and still obtain descent performance. Nevertheless, training our method jointly with the LCA loss leads to better results.

5.2 Learning Emotion-specific Associations

5.2.1 Word-Level

In this section, we present the top 10 words learned by SpanEmo for each emotion class by extracting the learned representations for each emotion class and all words in every input sentence, and then computing the similarity between them via cosine similarity. Finally, we performed this operation on all inputs in the SemEval2018 English validation set and then sorted all words for each emotion class in ascending order.

Table 7 presents the top-10 words per emotion class. As shown in Table 7, the words discovered by our framework are indicative of the corresponding emotion. This helps to show that SpanEmo learns meaningful associations between emotion classes and words automatically, which can be beneficial for feature extraction and learning. Additionally, SpanEmo demonstrated that it can learn diverse words as well as shareable words across some emotions. For example, the words $\{pissed, wrath, smashed\}$ are associated with both anger and disgust, demonstrating the ability of SpanEmo to learn the relations between emotions.

Emotion	Top 10 Words
anger	anger pissed wrath idiots dammit kicking irritated thrown smashed complain
anticipation	prediction planning mailsport assumptions upcoming waiting route waited frown ideas
disgust	disgusting smashed gross hate pissed wrath dirty awful vile dumb
fear	nervous fear terror frightening afraid frown panic terrifying scary dreading
joy	happy excitement joyful congratulations glad delightful excited adorable amusing smiling
love	love sweetness loved hug mate lucky carefree shine care gracious
optimism	optimism integrity salvation persevere perspective bright effort faith glad lord
pessimism	hopeless frown disappointed weary dread despair depressing chronic suicide pain
sadness	sadness frown depressing saddened hurt disappointed weary upset sorrow hate
surprise	stunned awestruck shocking awe mailsport buster genuinely curious hardly believing
trust	integrity shine respect courage sign effort confident faith easy kindness

Table 7: Top 10 words associated with each corresponding emotion.

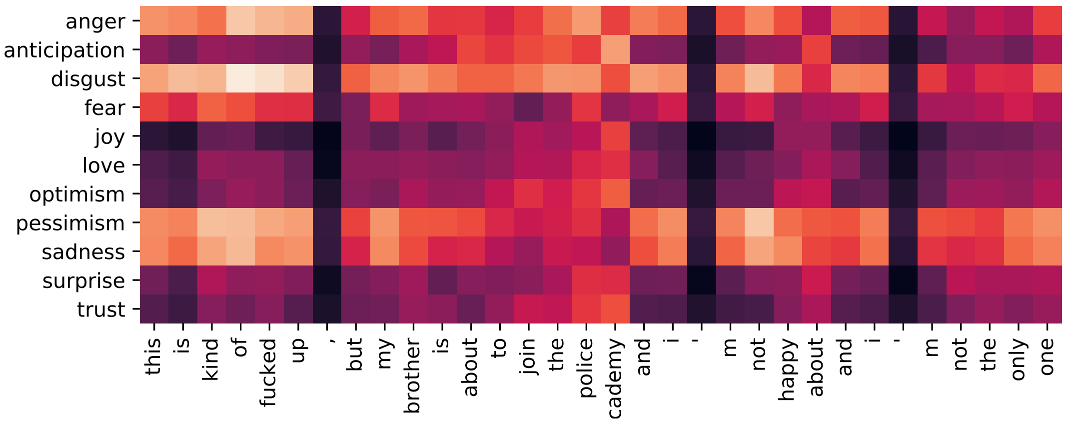


Figure 2: Visualisation on an example. The left presents the emotion labels, and the bottom presents the example. Each cell shows the cosine similarity value computed via using the hidden representation of each word and label. Lighter colour indicates higher similarity.

5.2.2 Sentence-Level

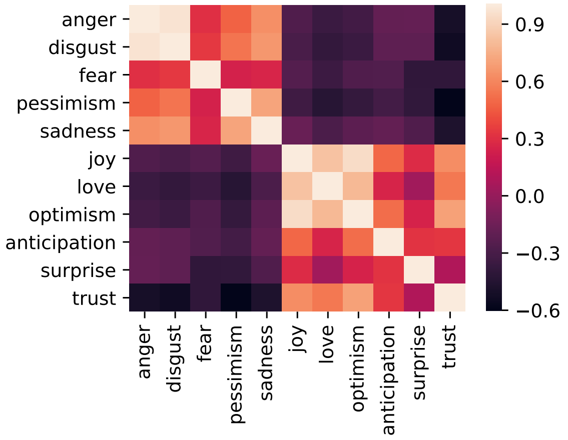
We visualised an example from the English validation set annotated with four emotions, which were anger, disgust, pessimism and sadness. Our goal was to determine whether by adding emotion classes to the example, SpanEmo could learn their associations to each other (i.e., associations between emotion classes and words in the example). To compute the similarity between emotion classes and words in the example, we also followed the same process discussed in section 5.2.1.

Figure 2 presents the results, where lighter cells indicate higher similarity, while darker cells indicate lower similarity. As shown in Figure 2, the learned representations capture the association between the correct emotion label set and every token in the example. Interestingly, we can also observe that the word "happy" is usually expressed as a positive emotion, but, in this context, this word becomes negative and the model learns this contextual information. Moreover, the phrase "about to join the police academy" is associated with "anticipation", which makes sense although this class is not part of the correct label set. This demonstrates the utility and advantages of our approach not only in deriving associations reported in the annotations, but also providing us with a mechanism to explore additional information beyond them.

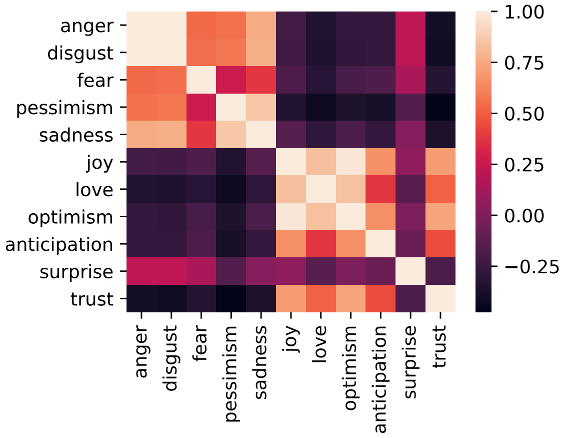
5.3 Label Correlations

Since one of the research questions in this paper was to learn the multiple co-existing emotions from a multi-label emotion data set, we analysed the learned emotion correlations from SpanEmo and compared them to those adopted from the ground truth labels in the SemEval2018 validation set. Figure 3 presents the two emotion correlations as obtained from the ground truth labels and from the predicted labels, respectively.

It can be observed that Figure 3(b) is almost identical to 3(a), demonstrating that our method in capturing the emotion correlations is in line with



(a) Emotion correlations: GT



(b) Emotion correlations: Prediction

Figure 3: The top plot presents emotion correlations obtained from the ground truth (GT) labels, whereas the bottom plot presents emotion correlations obtained from the predicted labels.

what the emotion annotations have revealed. 3(b), which was learned by SpanEmo, also highlights that negative emotions are positively correlated with each other, and negatively correlated with positive emotions. For example, “anger and disgust” share almost the same patterns, which is consistent with the studies of Mohammad and Bravo-Marquez (2017) and Agrawal et al. (2018), both of which report the same issue with negative emotions of “anger” and “disgust”, as they are easily confused with each other. This is not surprising as their manifestation in language is quite similar in terms of the use of similar words/expression. We also noted this finding when analysing the top-10 key words learned by SpanEmo in section 5.2.1. In short, taking into account emotion correlations is crucial for multi-label emotion classification in addressing the ambiguity characteristic of the task, especially for emotions that are highly correlated.

5.4 Influence of Parameter (α)

SpanEmo was trained with BCE loss and with LCA loss via a weight (α), whose impact on the results is presented in Figure 4. It should be mentioned that this analysis was performed on the validation set of SemEval2018 data set. The lower bound (i.e., 0.0) indicates that the model was trained only with the BCE loss, whereas the upper bound (i.e., 1.0) indicates that it was trained only with the LCA loss. When the value of α increased from 0.0 to 0.5, the results first improved considerably and then gradually deteriorated apart from the results of the macro F1-score. The results of BCE loss favoured the micro F1- and jaccard score, whereas the results of LCA loss favoured the macro F1-score. However, integrating LCA with BCE can balance the results across all three metrics, resulting in strong performance. The best results were achieved on almost all metrics when the value of α was set to 0.2. Thus, we set the value of parameter α to 0.2 for all experiments reported in this paper.

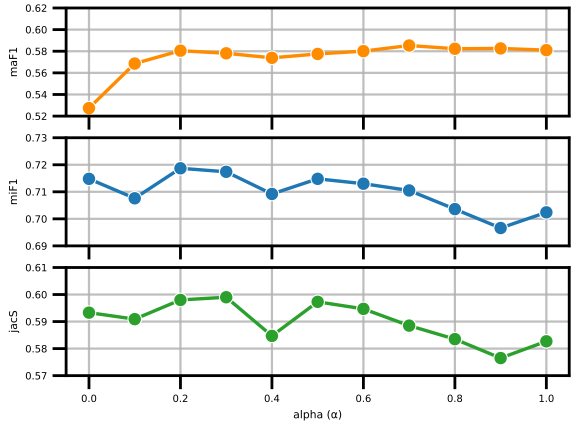


Figure 4: Sensitivity analysis of parameter (α). Note that $\alpha = 0.0$ means that only BCE loss is used in training SpanEmo, whereas $\alpha = 1.0$ means that only LCA loss is utilised in training it.

6 Related Work

There is a large body of NLP literature on emotion recognition (Mohammad and Alm, 2015). Earlier studies focused on lexicon-based approaches, which make use of some words and their corresponding labels to identify emotions in text, e.g. NRC[4] (Mohammad and Turney, 2013) and

---

[4] Bravo-Marquez et al. (2016) proposes an approach for expanding it for the language used in Twitter.

EmoSenticNet (Poria et al., 2014). Other methods treat the emotion recognition task as a supervised learning task, in which a learner (e.g. linear classifier based methods) is trained on the features of labelled data to classify inputs into one label (Bostan and Klinger, 2018; Liew et al., 2016; Mohammad et al., 2015; Tang et al., 2013; Wang et al., 2012; Aman and Szpakowicz, 2007).

More recently, several neural network models have been developed for this task, obtaining competitive results on different emotion data sets. Some of these models generally focus on a single-label emotion classification, in which only a single label is assigned to each input (Islam et al., 2019; Xia and Ding, 2019; Alhuzali et al., 2018b,a; Agrawal et al., 2018; Saravia et al., 2018; Felbo et al., 2017; Abdul-Mageed and Ungar, 2017). Other models have also been proposed for multi-label emotion classification, in which one or more labels are assigned to each input (see detailed description in section 3.3).

Our work is motivated by research focused on learning features corresponding to each emotion as well as incorporating the relations between emotions into a loss function (Fei et al., 2020; He and Xia, 2018). Our work differs from these two works in the following ways: i) our method learns features related to each corresponding emotion without relying on any external resources (e.g. lexicons). ii) We further integrated the relations between emotions into the loss function by taking advantage of the label co-occurrences in a multi-label emotion data set. In this respect, our approach does not rely on any theory of emotion. iii) We empirically evaluated our method for three languages, demonstrating its effectiveness as being language agnostic. In contrast to previous research, we focus on both learning emotion-specific associations and integrating the relations between emotions into the loss function.

7 Conclusion

We have proposed a novel framework "SpanEmo" aimed at casting multi-label emotion classification as a span-prediction problem. We demonstrated that our proposed method outperforms prior approaches reported in the literature on three languages (i.e., English, Arabic and Spanish). Our empirical evaluation and analyses also demonstrated the utility and advantages of our method for multi-label emotion classification, specifically the addition of emotion classes to the input sentence, which helped the model learn emotion-specific associations and increase its performance. Finally, training our method with LCA loss jointly led to better results, showing the benefits of integrating the relations between emotions into the loss function.

The standard approach in a multi-label emotion classification problem often focuses on modelling individual emotion independently. In this respect, most existing methods do not take into account label dependencies while learning emotion-specific associations. However, we demonstrated the effectiveness of including label information to the input sentence when training SpanEmo, helping it achieve better performance and capture emotion correlations. We hope that this study will inspire the community to investigate further the vital role of learning label dependencies and associations corresponding to each emotion.

Acknowledgements

We would like to thank the anonymous reviewers and Chrysoula Zerva for their valuable feedback and suggestions. The first author is supported by a doctoral fellowship from Umm Al-Qura University.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. EmoNet: Fine-Grained Emotion Detection with Gated Recurrent Neural Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 718–728.

Ameeta Agrawal, Aijun An, and Manos Papagelis. 2018. Learning emotion-enriched word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 950–961.

Yaara Benger Alaluf and Eva Illouz. 2019. Emotions in consumer studies. *The Oxford Handbook of Consumption*, page 239.

Hassan Alhuzali, Muhammad Abdul-Mageed, and Lyle Ungar. 2018a. Enabling deep learning of emotion with first-person seed expressions. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 25–35.

Hassan Alhuzali and Sophia Ananiadou. 2019. Improving classification of adverse drug reactions through using sentiment analysis and transfer learning. In *Proceedings of the 18th BioNLP Workshop and*

Shared Task, pages 339–347, Florence, Italy. Association for Computational Linguistics.

Hassan Alhuzali, Mohamed Elaraby, and Muhammad Abdul-Mageed. 2018b. UBC-NLP at IEST 2018: Learning implicit emotion with an ensemble of language models. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 342–347, Brussels, Belgium. Association for Computational Linguistics.

Nourah Alswaidan and Mohamed El Bachir Menai. 2020. Hybrid feature model for emotion recognition in arabic text. *IEEE Access*, 8:37843–37854.

Saima Aman and Stan Szpakowicz. 2007. Identifying expressions of emotion in text. In *International Conference on Text, Speech and Dialogue*, pages 196–205. Springer.

Mario Ezra Aragón, Adrian Pastor López-Monroy, Luis Carlos González-Gurrola, and Manuel Montes. 2019. Detecting depression in social media using fine-grained emotions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1481–1486.

Gilbert Badaro, Obeida El Jundi, Alaa Khaddaj, Alaa Maarouf, Raslan Kain, Hazem Hajj, and Wassim El-Hajj. 2018. EMA at SemEval-2018 task 1: Emotion mining for Arabic. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 236–244, New Orleans, Louisiana. Association for Computational Linguistics.

Christos Baziotis, Athanasiou Nikolaos, Alexandra Chronopoulou, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, Shrikanth Narayanan, and Alexandros Potamianos. 2018. Ntua-slp at semeval-2018 task 1: Predicting affective content in tweets with deep attentive rnns and transfer learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 245–255.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754.

Laura-Ana-Maria Bostan and Roman Klinger. 2018. An Analysis of Annotated Corpora for Emotion Classification in Text. In *Proceedings ofthe 27th International Conference on Computational Linguistics*, pages 2104–2119.

Felipe Bravo-Marquez, Eibe Frank, Saif M Mohammad, and Bernhard Pfahringer. 2016. Determining word-emotion associations from tweets by multi-label classification. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 536–539. IEEE.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *to appear in PML4DC at ICLR 2020*.

Xuetong Chen, Martin D Sykora, Thomas W Jackson, and Suzanne Elayan. 2018. What about mood swings: Identifying depression on twitter with temporal measures of emotions. In *Companion Proceedings of the The Web Conference 2018*, pages 1653–1660. International World Wide Web Conferences Steering Committee.

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

Hao Fei, Yue Zhang, Yafeng Ren, and Donghong Ji. 2020. Latent emotion memory for multi-label emotion classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7692–7699.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625.

José-Ángel González, Lluís-F. Hurtado, and Ferran Pla. 2018. ELiRF-UPV at SemEval-2018 tasks 1 and 3: Affect and irony detection in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 565–569, New Orleans, Louisiana. Association for Computational Linguistics.

Huihui He and Rui Xia. 2018. Joint binary neural network for multi-label learning with applications to emotion classification. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 250–259. Springer.

Jonathan Herzig, Guy Feigenblat, Michal Shmueli-Scheuer, David Konopnicki, and Anat Rafaeli. 2016. Predicting customer satisfaction in customer support conversations in social media using affective features. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, pages 115–119.

Jumayel Islam, Robert E. Mercer, and Lu Xiao. 2019. Multi-channel convolutional neural network for twitter emotion and sentiment recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*

(Long and Short Papers), pages 1355–1365, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jasy Liew, Suet Yan, and Howard R Turtle. 2016. Exploring Fine-Grained Emotion Detection in Tweets. In *The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 73–80.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Saif M Mohammad and Cecilia Ovesdotter Alm. 2015. Computational analysis of affect and emotion in language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*.

Saif M. Mohammad and Felipe Bravo-Marquez. 2017. Emotion intensities in tweets. In *Proceedings of the sixth joint conference on lexical and computational semantics (*Sem)*, Vancouver, Canada.

Saif M Mohammad and Svetlana Kiritchenko. 2013. Using nuances of emotion to identify personality. *arXiv preprint arXiv:1309.6352*.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. 29(3):436–465.

Saif M Mohammad, Xiaodan Zhu, Svetlana Kiritchenko, and Joel Martin. 2015. Sentiment, emotion, purpose, and style in electoral tweets. *Information Processing & Management*, 51(4):480–499.

Hala Mulki, Chedi Bechikh Ali, Hatem Haddad, and Ismail Babaoğlu. 2018. Tw-StAR at SemEval-2018 task 1: Preprocessing impact on multi-label emotion classification. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 167–171, New Orleans, Louisiana. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Robert Plutchik. 1980. *Emotion: A psychoevolutionary synthesis*. Harpercollins College Division.

Robert Plutchik. 1984. Emotions: A general psychoevolutionary theory. *Approaches to emotion*, 1984:197–219.

Soujanya Poria, Alexander Gelbukh, Erik Cambria, Amir Hussain, and Guang-Bin Huang. 2014. Emosenticspace: A novel framework for affective common-sense reasoning. *Knowledge-Based Systems*, 69:108–123.

Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In *Empirical Methods in Natural Language Processing*, pages 3687–3697.

Duyu Tang, Bing Qin, Ting Liu, and Zhenghua Li. 2013. Learning sentence representation for emotion classification on microblogs. In *Natural Language Processing and Chinese Computing*, pages 212–223. Springer.

Svitlana Volkova and Yoram Bachrach. 2016. Inferring perceived demographics from user emotional tone and user-environment emotional contrast. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1567–1578.

Wenbo Wang, Lu Chen, Krishnaprasad Thirunarayan, and Amit P Sheth. 2012. Harnessing twitter" big data" for automatic emotion identification. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 587–592. IEEE.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012, Florence, Italy. Association for Computational Linguistics.

Peng Xu, Zihan Liu, Genta Indra Winata, Zhaojiang Lin, and Pascale Fung. 2020. Emograph: Capturing emotion correlations using graph networks. *arXiv preprint arXiv:2008.09378*.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3915–3926.

Chih-Kuan Yeh, Wei-Chieh Wu, Wei-Jen Ko, and Yu-Chiang Frank Wang. 2017. Learning deep latent spaces for multi-label classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 2838–2844.

Wenhao Ying, Rong Xiang, and Qin Lu. 2019. Improving multi-label emotion classification by integrating both general and domain-specific knowledge. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 316–321, Hong Kong, China. Association for Computational Linguistics.

Jianfei Yu, Luis Marujo, Jing Jiang, Pradeep Karuturi, and William Brendel. 2018. Improving multi-label emotion classification via sentiment classification with dual attention transfer network. ACL.

Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, 18(10):1338–1351.

Deyu Zhou, Yang Yang, and Yulan He. 2018. Relevant emotion ranking from text constrained with emotion relationships. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 561–571.

Exploiting Position and Contextual Word Embeddings for Keyphrase Extraction from Scientific Papers

Krutarth Patel
Computer Science
Kansas State University
Manhattan, Kansas, United States
`kipatel@ksu.edu`

Cornelia Caragea
Computer Science
University of Illinois at Chicago
Chicago, Illinois, United States
`cornelia@uic.edu`

Abstract

Keyphrases associated with research papers provide an effective way to find useful information in the large and growing scholarly digital collections. In this paper, we present KPRank, an unsupervised graph-based algorithm for keyphrase extraction that exploits both positional information and contextual word embeddings into a biased PageRank. Our experimental results on five benchmark datasets show that KPRank that uses contextual word embeddings with additional position signal outperforms previous approaches and strong baselines for this task.

1 Introduction

Keyphrase extraction is the task of automatically extracting a small set of descriptive words or phrases that can accurately summarize the topics discussed in a document (Hasan and Ng, 2010, 2014). Keyphrases are useful in many applications such as document indexing and summarization (Abu-Jbara and Radev, 2011; Qazvinian et al., 2010; Turney, 2003), topic tracking (Augenstein et al., 2017), contextual advertising (Yih et al., 2006), and opinion mining (Berend, 2011).

Most of the previous approaches to keyphrase extraction are either supervised or unsupervised. While supervised approaches perform generally better (Kim et al., 2013), the unsupervised ones have the advantage that they do not require large human-annotated corpora for training reliable models. Unsupervised keyphrase extraction methods usually use graph-based ranking algorithms such as PageRank that work on the word graph constructed from the target document (Mihalcea and Tarau, 2004). Various PageRank extensions have been proposed that incorporate different types of information (Wan and Xiao, 2008; Gollapalli and Caragea, 2014). For example, Wan and Xiao (2008) proposed to incorporate a local neighborhood of the target document into the graph construction, with the neighborhood being determined based on the textual similarity between documents. Liu et al. (2010) exploited topical information to select keyphrases from all major topics. More recently, Mahata et al. (2018) proposed a theme-weighted biased PageRank, called Key2Vec, for keyphrase extraction. In Key2Vec, a theme-vector is computed by averaging the embeddings of words and phrases from the title of a scientific document to capture its theme and the PageRank is biased based on the similarity of candidate words or phrases to the computed theme vector. However, this model is oblivious to the position of words in a scientific document, in which more important words appear not only frequently, but also close to the beginning of the document (Florescu and Caragea, 2017).

Inspired by the Transformer models (Vaswani et al., 2017) that infuse positional information into the word embeddings to produce embeddings with time signal, we propose an extension of Key2Vec that incorporates words' positions into a biased PageRank. Moreover, different from Mahata et al. (2018), who used non-contextual FastText embeddings (Mikolov et al., 2018), we propose to integrate SciBERT contextual embeddings (Beltagy et al., 2019) into our biased PageRank extension. Our contributions are as follows: (1) We propose KPRank, an unsupervised graph-based algorithm that exploits both the position of words in a document and the contextual word embeddings for computing a biased PageRank score for ranking candidate phrases; (2) We show empirically that infusing position information into our biased KPRank model yields better performance compared with its counterpart that does not use the position information. In addition, KPRank with contextual SciBERT embeddings performs better than FastText-based KPRank; (3) Finally, we show that KPRank outperforms many previous unsupervised models.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1585–1591
April 19 - 23, 2021.

2 Proposed Approach

In this section, we describe our unsupervised graph-based algorithm called KPRank, that exploits both position information of the words in a document along with contextual word embeddings for computing a biased PageRank score for each candidate word. Our approach consists of three steps: (1) candidate word selection and word graph construction; (2) word scoring by biased PageRank; and (3) candidate phrase formation.

2.1 Candidate Word Selection and Graph Construction

For a target doucment D, we first apply a part-of-speech filter[1] and select only nouns and adjectives as candidate words, consistent with previous works (Gollapalli and Caragea, 2014; Mihalcea and Tarau, 2004; Wan and Xiao, 2008). We build a word graph $G = (N, E)$ for D using the candidate words as nodes in G. N and E are the sets of nodes and edges, respectively. We consider an edge $(n_i, n_j) \in E$ between two nodes n_i and n_j in N if the words corresponding to these nodes appear within a window of k consecutive words in the content of D. We experimented with values of k from 1 to 10 and obtained best results with $k = 10$, which is consistent with (Wan and Xiao, 2008). The weight of an edge (n_i, n_j), denoted as w_{ij}, is computed based on the co-occurrence count of the two words within k consecutive words in D ($k = 10$). Here, we build undirected graphs because prior work (Mihalcea and Tarau, 2004; Liu et al., 2010) observed that the type of graph (directed or undirected) used to represent the text does not significantly influence the performance of the keyphrase extraction task.

2.2 Biased PageRank

Preliminaries. PageRank (Page et al., 1998) is a graph-based ranking algorithm that iteratively calculates the importance of each node in a graph through endorsements from its neighbors. For document D, we construct an undirected graph G as explained above. Initially, the score of each node in G is set to $\frac{1}{|N|}$. This score is then iteratively updated using PageRank. That is, the score s for node n_i is obtained by applying the equation:

$$s(n_i) = (1 - \alpha)\widetilde{p_i} + \alpha \sum_{n_j \in Adj(n_i)} \frac{w_{ji}}{O(n_j)} s(n_j) \quad (1)$$

where $O(n_j) = \sum_{n_k \in Adj(n_j)} w_{jk}$ and $Adj(n_j)$ is the set of all adjacent nodes of node $n_j \in N$. $\widetilde{p_i}$ is defined below.

In order to prevent the PageRank from getting stuck in cycles or dead ends, a dumping factor α was added to Eq. (1) to allow the PageRank to randomly jump to any node in the graph ($\alpha = 0.85$). Let $\widetilde{p} = [\widetilde{p}_1, \cdots, \widetilde{p}_i, \cdots, \widetilde{p}_{|N|}]$ be the probability distribution of randomly jumping to any node in the graph. For an unbiased PageRank, this is a uniform distribution, with $\widetilde{p_i} = \frac{1}{|N|}$, for all i from 1 to $|N|$. For a biased PageRank, this probability distribution is not uniform, but rather the nodes in the graph are visited preferentially, with some nodes being visited more often than others, depending on the $\widetilde{p_i}$ value for node n_i (Haveliwala, 2003). Key2Vec is an example of (topic) biased PageRank for keyphrase extraction that computes $\widetilde{p_i}$ for node n_i using the cosine similarity between the embedding of word/phrase corresponding to node n_i and a theme vector for the entire document, which corresponds to the aggregated word/phrase embeddings from the document's title (Mahata et al., 2018). That is, $\widetilde{p_i}$ is higher for words/phrases that are topically (semantically) more similar to the overall theme vector for the document. Next, we describe our extension KPRank of Key2Vec.

KPRank. In our proposed approach, we calculate $\widetilde{p_i}$ for node n_i using two types of scores: *theme (or topic) score* and *positional score*. We multiply both scores to assign a final weight to node n_i before running the biased PageRank algorithm. Both scores and their calculation are explained below.

To calculate the *theme score* (ts_i) for node $n_i \in N$, we first calculate a theme vector (T_D) for document D. A *theme vector* is obtained by averaging SciBERT (Beltagy et al., 2019) word embeddings of adjectives and nouns from D's title. The *theme score* for node n_i is calculated using the cosine similarity of the SciBERT word embedding corresponding to node n_i and T_D. The idea is to assign a higher score to a word if that word is closer to the theme (topic) of a given document. For obtaining word embeddings, for all the words with similar stemmed version (obtained with Porter stemmer), we averaged the contextualized word embeddings of a word obtained by using SciBERT. We used the title and abstract of a document as input to the SciBERT model. We also experimented with pretrained BERT (Devlin et al., 2018), and found that the performance of BERT-based KPRank and SciBERT-based KPRank are very similar.

---

[1] We used Python's NLTK toolkit for POS tagging.

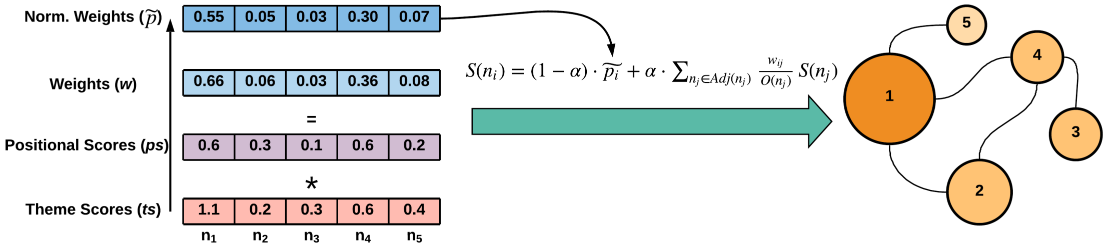


Figure 1: Illustration of our approach.

To calculate the *positional score* (ps_i) for node n_i, we consider the set P_i that contains all the positions of occurrence in the text of the word corresponding to node n_i. Then, ps_i is calculated as $ps_i = \sum_{j\in P_i}\frac{1}{j}$. For example, for a word occurring on positions 1 and 10 in the text, its ps_i score is $\frac{1}{1}+\frac{1}{10}$, whereas for a word occurring on position 100, its ps_i score is $\frac{1}{100}$. The intuition behind this weighting scheme is to give higher weight to words appearing in the beginning of a document since in scientific writing, authors tend to use keyphrases very early in the document (even from the title) (Florescu and Caragea, 2017). Based on these considerations, the first position of a phrase/word and its relative position are also used in many supervised approaches as powerful features (Patel and Caragea, 2019; Hulth, 2003; Wu et al., 2005)

To calculate the weight w_i for n_i, we perform multiplication of both the *theme score* (ts_i) and the *positional score* (ps_i). The intuition is that we give preference to words that appear near the beginning of the document and are more frequent as compared with less frequent words appearing later in document even though both words may be equally close to the theme of the document or may have similar theme score. The vector $\widetilde{p}$ is last set to the normalized weights for each node as follows:

$$\widetilde{p} = \left[\frac{w_1}{\sum_{i=1}^{|N|} w_i}, \frac{w_2}{\sum_{i=1}^{|N|} w_i}, \ldots, \frac{w_{|N|}}{\sum_{i=1}^{|N|} w_i}\right] \qquad (2)$$

The biased PageRank scores for each node n_i are finally calculated by iteratively applying Eq. (1) with $\widetilde{p}$ as in Eq. (2). Figure 1 shows the illustration of our approach. As can be seen, even though both n_1 and n_4 have similar *theme score*, final weights are different based on different *positional scores*.

In our experiments, the PageRank scores are updated until the difference between two consecutive iterations is ≤ 0.001 or for 100 iterations.

2.3 Candidate Phrases Formation

Candidate words appearing continuously in the document are concatenated to generate candidate phrases. We consider phrases with the regular expression (adjective)*(noun)+, of length up to four words, to generate candidate phrases. We used stemmed version of each word using Porter stemmer. We use POS tagger from Python's NLTK toolkit. The score for each candidate phrase is calculated by summing up the scores of its individual words (Wan and Xiao, 2008). The top-scoring phrases are output as predicted keyphrases for a given document.

Dataset	# Test Papers	Avg. # KP per doc.
SemEval	100	6.87
Inspec	500	8.82
Krapivin	400	3.48
NUS	211	5.49
ACM	20,000	2.67

Table 1: The summary of the datasets along with their gold-standard keyphrases (KP).

3 Data

For evaluation, we use five datasets, which we describe below. We use the combination of controlled (author assigned) and uncontrolled (reader assigned) keyphrases as gold-standard phrases. We used uncontrolled keyphrases when available. Table 1 shows the summary of the datasets.

SemEval (Kim et al., 2010) contains 288 research papers with a train and test split consisting of 188 and 100 papers, respectively.

Inspec (Hulth, 2003) contains 2,000 research papers. It has a train-validation-test split of 1,000, 500 and 500 papers, respectively.

Krapivin (Krapivin et al., 2009) contains 2,304 ACM research papers with full text and author-assigned keyphrases. Similar to (Meng et al., 2017), since the dataset does not have a train-test split, we sampled 400 papers as the test set.

NUS (Nguyen and Kan, 2007) contains 211 research papers. This dataset does not have a train and test split and it is relatively small. Hence, consistent with (Meng et al., 2017) we used entire dataset as the test set.

ACM (Patel and Caragea, 2019) This dataset

	SemEval		Inspec		Krapivin		NUS		ACM	
	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10
KPRank(SB)	**22.51**	**25.76**	**27.72**	32.30	17.74	18.57	**21.09**	**22.36**	**14.79**	**15.17**
KPRank(SB−POS)	17.33	23.97	26.88	32.72	15.81	16.97	16.79	19.68	12.13	13.14
KPRank(FastText)	22.04	25.39	27.28	32.12	18.20	**18.91**	20.69	22.12	14.77	15.08
PositionRank	**22.51**	24.99	26.73	31.84	**18.49**	18.30	18.65	20.99	13.04	14.09
Key2Vec	17.54	23.63	26.97	**32.82**	15.50	16.68	16.80	20.10	12.08	13.07
Tf-Idf	18.13	21.82	24.73	31.41	15.67	17.23	14.22	18.05	11.07	12.90
TextRank	12.12	17.90	22.81	30.47	09.15	12.91	09.04	12.64	05.02	07.54
SingleRank	11.22	18.24	24.11	31.96	10.25	13.53	08.69	13.78	05.66	08.32
ExpandRank	14.65	20.46	24.81	31.45	12.32	15.32	10.90	15.29	10.02	11.28
TopicRank	10.77	11.04	14.65	17.46	08.11	07.82	11.79	11.45	09.39	07.62

Table 2: The comparison of SciBERT (SB) based KPRank, and previous works.

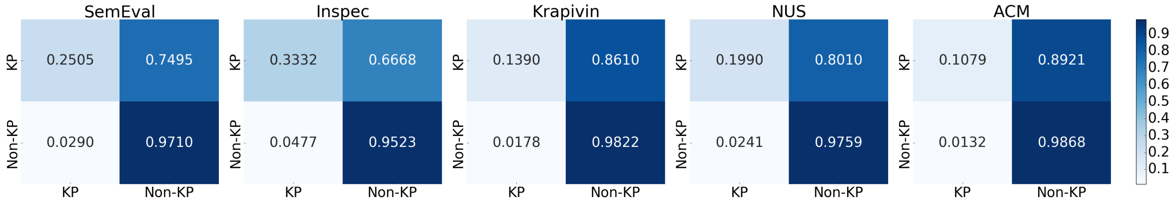


Figure 2: Keyphrase extraction confusion matrices of KPRank(SB) using @5 predictions on all the datasets. The darker the blue on the main diagonal, the more accurate the model is.

contains 30,000 papers published in ACM conferences with a train and test split consisting of 10,000 and 20,000 papers, respectively.

For each dataset we use its test set for evaluation.

4 Experimental Setup and Results

Evaluation metrics. To evaluate the performance of different methods, we use micro avg. F1-score. We report the performance for the top 5 and 10 candidate phrases returned by different methods as in (Meng et al., 2017). To create a word graph for a given document, we use its title and abstract. To match the predicted keyphrases with gold-standard keyphrases, we do exact match between the stemmed version of each.

The effect of position, contextual embeddings, and the comparison with previous works. To see the effect of positional information, we compare the performance of KPRank that uses contextual SciBERT (SB) embeddings along with positional information (denoted as KPRank(SB)) with that of its counterpart that does not use positional information (denoted as KPRank(SB−POS)). Moreover, to see the effect of contextual embeddings, we compare the performance of SciBERT-based KPRank (KPRank(SB)) with that of KPRank that uses FastText non-contextual word embeddings (Mikolov et al., 2018) (denoted as KPRank(FastText)). For FastText, we used pre-trained 300 dimensional embeddings trained on subword information on Common Crawl. Note that KPRank(SciBERT) and KPRank(FastText) use positional information along with the theme score. Last, we compare the performance of KPRank with Tf-Idf and six PageRank based unsupervised methods as baselines: PositionRank (Florescu and Caragea, 2017), Key2Vec (Mahata et al., 2018), TextRank (Mihalcea and Tarau, 2004), SingleRank (Wan and Xiao, 2008), ExpandRank (Wan and Xiao, 2008), TopicRank (Bougouin et al., 2013).

Tables 2 shows these comparisons on SemEval, Inspec, Krapivin, NUS, and ACM. It can be seen from the table that adding position information shows much higher improvement in the performance of KPRank, i.e. KPRank(SB) substantially outperforms KPRank(SB−POS). Moreover, KPRank(SB) outperforms KPRank(FastText) on all the datasets except for Krapivin. Importantly, KPRank(SB) outperforms most baseline methods, including Key2Vec (by a large margin) e.g., on SemEval, KPRank(SB) achieves an F1@5 of 22.51% as compared with 17.54% achieved by Key2Vec. We can also notice from Table 2 that KPRank(SB) achieves comparable performance whenever any baseline method achieves the best performance.

Figure 2 shows the confusion matrices of KPRank(SB) using @5 predictions on all five datasets. Each matrix is represented as a heat map, i.e., the darker the blue color the higher the value at that position and the darker the blue on the main diagonal, the more accurate the model is.

Organization design: The continuing influence of *information technology*

Drawing from an *information processing perspective*, this paper examines how *information technology* (IT) has been a catalyst in the development of new forms of *organizational structures*. [...] to the present *environmental instability* that now characterizes many industries. Specifically, the authors suggest that advances in IT have enabled managers to adapt existing forms and create new models for *organizational design* that better fit requirements of an unstable environment. [...]. IT has gone from a support mechanism to a substitute for *organizational structures* in the form of the shadow structure. [...]

Gold-standard keyphrases: *Organization design, Information processing perspective, Organizational structures, Environmental instability, Information technology*

Predicted keyphrases: ***Organization design***, ***Information technology***, ***Information processing perspective***, ***Organizational structures***, *Organizational design, Organization, Information processing, Shadow structure, New forms, Bureaucratic structure*

Figure 3: The title, abstract, gold-standard keyphrases and predicted keyphrases of a paper. The phrases marked with cyan in the title and abstract shown on the top of the figure are gold-standard keyphrases.

Comparison with a supervised approach. Usually, the performance of the supervised keyphrase extraction models is better than the unsupervised models (Kim et al., 2013). We compare the performance of KPRank(SB) with the CRF based sequence classification model for the keyphrase extraction (Patel and Caragea, 2019) that uses word embeddings as features along with document specific features. The CRF model outperforms KPRank(SB) on all five datasets, e.g., CRF model achieves an F1 of 45.73% as compared with 25.76% achieved by KPRank(SB) on SemEval.

Anecdotal example. To see the quality of predicted phrases by the KPRank(SB), we randomly selected a paper from the Inspec dataset and evaluated the KPRank(SB) on it. We manually inspected the top-10 predictions by the KPRank(SB) and contrasted them with the gold-standard keyphrases. The title, abstract, gold-standard keyphrases and top-10 predicted keyphrases for this paper are shown in Figure 3. Precisely, in the figure, the cyan italic phrases shown in the text on the top of the figure represent gold-standard keyphrases, whereas the bottom of the figure shows gold-standard keyphrases and the top-10 predicted keyphrases by KPRank(SB) (shown in the order of their prediction). It can be seen from the figure that four out of five gold-standard keyphrases are present in the top-5 predicted keyphrases.

We can also see that KPRank(SB) did not predict gold-standard phrase "environmental instabily." A closer inspection of the document and both types of scores (*theme score* and *positional score*) assigned by KPRank(SB) to both constituent words of the gold-standard phrase that was not ranked in top-10 predictions revealed that these constituent words have lower values of *theme score* and they both appear only once in the document. Hence, the Pagerank algorithm will not boost these words. Inspecting other errors, we found that KPRank can fail to predict phrases that contain words that are less frequent in the document and their word embeddings are far from the *theme vector*.

5 Conclusion and Future Work

In this paper, we proposed a novel unsupervised graph-based algorithm, named KPRank, which incorporates both positional appearances of the words along with contextual word embeddings for computing a biased PageRank score for each candidate word. Our experimental results on five datasets show that incorporating position information into our biased KPRank model yields better performance compared with a KPRank that does not use the position information, and SciBERT-based KPRank usually outperforms FastText-based KPRank on this task. Moreover, KPRank outperforms strong baseline methods. In the future, it would be interesting to explore KPRank on other domains, such as Biology, and Social Science.

Acknowledgments

We thank our anonymous reviewers for their constructive comments and feedback, which helped improve our paper. This research is supported in part by NSF. Any opinions, findings, and conclusions expressed here are those of the authors and do not necessarily reflect the views of NSF.

References

Amjad Abu-Jbara and Dragomir Radev. 2011. Coherent citation-based summarization of scientific papers. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 500–509.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 546–555.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Corina Florescu and Cornelia Caragea. 2017. Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, Vancouver, Canada.

Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, pages 1629–1635.

Kazi Saidul Hasan and Vincent Ng. 2010. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 365–373.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273.

Taher H Haveliwala. 2003. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE transactions on knowledge and data engineering*, pages 784–796.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 216–223.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2013. Automatic keyphrase extraction from scientific articles. *Language resources and evaluation*, 47(3):723–742.

Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. Large dataset for keyphrases extraction. Technical report, University of Trento.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 366–376.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 634–639.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 404–411.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries*, pages 317–326.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1998. The pagerank citation ranking: bringing order to the web. *Technical report, Standford Digital Library Technologies Project.*

Krutarth Patel and Cornelia Caragea. 2019. Exploring word embeddings in crf-based keyphrase extraction from research papers. In *Proceedings of the 10th International Conference on Knowledge Capture*, pages 37–44.

Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd international conference on computational linguistics (COLING 2010)*, pages 895–903.

Peter D Turney. 2003. Coherent keyphrase extraction via web mining. *arXiv preprint cs/0308033*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, pages 855–860.

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. 2005. Domain-specific keyphrase extraction. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 283–284.

Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. 2006. Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web*, pages 213–222.

Benchmarking Machine Reading Comprehension: A Psychological Perspective

Saku Sugawara[1], **Pontus Stenetorp**[2], **Akiko Aizawa**[1]
[1]National Institute of Informatics, [2]University College London
`{saku,aizawa}@nii.ac.jp`, `p.stenetorp@cs.ucl.ac.uk`

Abstract

Machine reading comprehension (MRC) has received considerable attention as a benchmark for natural language understanding. However, the conventional task design of MRC lacks explainability beyond the model interpretation, i.e., reading comprehension by a model cannot be explained in human terms. To this end, this position paper provides a theoretical basis for the design of MRC datasets based on psychology as well as psychometrics, and summarizes it in terms of the prerequisites for benchmarking MRC. We conclude that future datasets should (i) evaluate the capability of the model for constructing a coherent and grounded representation to understand context-dependent situations and (ii) ensure substantive validity by shortcut-proof questions and explanation as a part of the task design.

1 Introduction

Evaluation of natural language understanding (NLU) is a long-standing goal in the field of artificial intelligence. Machine reading comprehension (MRC) is a task that tests the ability of a machine to read and understand unstructured text and could be the most suitable task for evaluating NLU because of its generic formulation (Chen, 2018). Recently, many large-scale datasets have been proposed, and deep learning systems have achieved human-level performance for some of these datasets.

However, analytical studies have shown that MRC models do not necessarily achieve human-level understanding. For example, Jia and Liang (2017) use manually crafted adversarial examples to show that successful systems are easily distracted. Sugawara et al. (2020) show that a significant part of already solved questions is solvable even after shuffling the words in a sentence or dropping content words. These studies demonstrate that we cannot explain what type of understanding is required by the datasets and is actually acquired by models. Although benchmarking MRC is related to the intent behind questions and is critical to test hypotheses from a top-down viewpoint (Bender and Koller, 2020), its theoretical foundation is poorly investigated in the literature.

In this position paper, we examine the prerequisites for benchmarking MRC based on the following two questions: (i) *What* does reading comprehension involve? (ii) *How* can we evaluate it? Our motivation is to provide a theoretical basis for the creation of MRC datasets. As Gilpin et al. (2018) indicate, interpreting the internals of a system is closely related to only the system's architecture and is insufficient for explaining how the task is accomplished. This is because even if the internals of models can be interpreted, we cannot explain what is *measured* by the datasets. Therefore, our study focuses on the explainability of the task rather than the interpretability of models.

We first overview MRC and review the analytical literature that indicates that existing datasets might fail to correctly evaluate their intended behavior (Section 2). Subsequently, we present a psychological study of human reading comprehension in Section 3 for answering the *what* question. We argue that the concept of *representation levels* can serve as a conceptual hierarchy for organizing the technologies in MRC. Section 4 focuses on answering the *how* question. Here, we implement psychometrics to analyze the prerequisites for the task design of MRC. Furthermore, we introduce the concept of *construct validity*, which emphasizes validating the interpretation of the task's outcome. Finally, in Section 5, we explain the application of the proposed concepts into practical approaches, highlighting potential future directions toward the advancement of MRC. Regarding the *what* question, we indicate that datasets should evaluate the capability of the *situation model*, which refers to the construction

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1592–1612
April 19 - 23, 2021.

Question	Foundation	Requirements	Future direction
What is reading comprehension?	Representation levels in human reading comprehension: (A) surface structure, (B) textbase, and (C) situation model.	(A) Linguistic-level sentence understanding, (B) comprehensiveness of skills for inter-sentence understanding, and (C) evaluation of coherent representation grounded to non-textual information.	(C) Dependence of context on defeasibility and novelty, and grounding to non-textual information with a long passage.
How can we evaluate reading comprehension?	Construct validity in psychometrics: (1) content, (2) substantive, (3) structural, (4) generalizability, (5) external, and (6) consequential aspects.	(1) Wide coverage of skills, (2) evaluation of the internal process, (3) structured metrics, (4) reliability of metrics, (5) comparison with external variables, and (6) robustness to adversarial attacks and social biases.	(2) Creating shortcut-proof questions by filtering and ablation, and designing a task for validating the internal process.

Table 1: Overview of theoretical foundations, requirements, and future directions of MRC discussed in this paper.

of a coherent and grounded representation of text based on human understanding. Regarding the *how* question, we argue that among the important aspects of the construct validity, *substantive validity* must be ensured, which requires the verification of the internal mechanism of comprehension.

Table 1 provides an overview of the perspectives taken in this paper. Our answers and suggestions to the *what* and *how* questions are summarized as follows: (1) Reading comprehension is the process of creating a situation model that best explains given texts and the reader's background knowledge. The situation model should be the next focal point in future datasets for benchmarking the human-level reading comprehension. (2) To evaluate reading comprehension correctly, the task needs to provide a rubric (scoring guide) for sufficiently covering the aspects of the construct validity. In particular, the substantive validity should be ensured by creating shortcut-proof questions and by designing a task formulation that is explanatory itself.

2 Task Overview

2.1 Task Variations and Existing Datasets

MRC is a task in which a machine is given a document (*context*) and it answers the questions based on the context. Burges (2013) provides a general definition of MRC, i.e., *a machine comprehends a passage of text if, for any question regarding that text that can be answered correctly by a majority of native speakers, that machine can provide a string which those speakers would agree both answers that question*. We overview various aspects of the task along with representative datasets as follows. Existing datasets are listed in Appendix A.

Context Styles A context can be given in various forms with different lengths such as a single passage (MCTest (Richardson et al., 2013)), a set of passages (HotpotQA (Yang et al., 2018)), a longer document (CBT (Hill et al., 2016)), or open domain (Chen et al., 2017). In some datasets, a context includes non-textual information such as images (RecipeQA (Yagcioglu et al., 2018)).

Question Styles A question can be an interrogative sentence (in most datasets), a fill-in-the-blank sentence (cloze) (CLOTH (Xie et al., 2018)), knowledge base entries (QAngaroo (Welbl et al., 2018)) and search engine queries (MSMARCO (Nguyen et al., 2016)).

Answering Styles An answer can be (i) chosen from a text span of the given document (*answer extraction*) (NewsQA (Trischler et al., 2017)), (ii) chosen from a candidate set of answers (*multiple choice*) (MCTest (Richardson et al., 2013)), or (iii) generated as a free-form text (*description*) (NarrativeQA (Kočiský et al., 2018)). Some datasets optionally allow answering by a *yes*/*no* reply (BoolQ (Clark et al., 2019)).

Sourcing Methods Initially, questions in small-scale datasets are created by experts (QA4MRE (Sutcliffe et al., 2013)). Later, fueling the development of neural models, most published datasets have more than a hundred thousand questions that are automatically created (CNN/Daily Mail (Hermann et al., 2015)), crowdsourced (SQuAD v1.1 (Rajpurkar et al., 2016)), and collected from examinations (RACE (Lai et al., 2017)).

Domains The most popular domain is Wikipedia articles (Natural Questions (Kwiatkowski et al., 2019)), but news articles are also used (Who-did-What (Onishi et al., 2016)). CliCR (Suster and Daelemans, 2018) and emrQA (Pampari et al., 2018) are datasets in the clinical domain. DuoRC

(Saha et al., 2018) uses movie scripts.

Specific Skills Several recently proposed datasets require specific skills including unanswerable questions (SQuAD v2.0 (Rajpurkar et al., 2018)), dialogues (CoQA (Reddy et al., 2019), DREAM (Sun et al., 2019)), multiple-sentence reasoning (MultiRC (Khashabi et al., 2018)), multi-hop reasoning (HotpotQA (Yang et al., 2018)), mathematical and set reasoning (DROP (Dua et al., 2019)), commonsense reasoning (CosmosQA (Huang et al., 2019)), coreference resolution (QuoRef (Dasigi et al., 2019)), and logical reasoning (ReClor (Yu et al., 2020)).

2.2 Benchmarking Issues

In some datasets, the performance of machines has already reached human-level performance. However, Jia and Liang (2017) indicate that models can easily be fooled by manual injection of distracting sentences. Their study revealed that questions simply gathered by crowdsourcing without careful guidelines or constraints are insufficient to evaluate precise language understanding.

This argument is supported by further studies across a variety of datasets. For example, Min et al. (2018) find that more than 90% of the questions in SQuAD (Rajpurkar et al., 2016) require obtaining an answer from a single sentence despite being provided with a passage. Sugawara et al. (2018) show that large parts of twelve datasets are easily solved only by looking at a few first question tokens and attending the similarity between the given questions and the context. Similarly, Feng et al. (2018) and Mudrakarta et al. (2018) demonstrate that models trained on SQuAD do not change their predictions even when the question tokens are partly dropped. Kaushik and Lipton (2018) also observe that question- and passage-only models perform well for some popular datasets. Min et al. (2019) and Chen and Durrett (2019) concurrently indicate that for multi-hop reasoning datasets, the questions are solvable only with a single paragraph and thus do not require multi-hop reasoning over multiple paragraphs. Zellers et al. (2019b) report that their dataset unintentionally contains stylistic biases in the answer options which are embedded by a language-based model.

Overall, these investigations highlight a grave issue of the task design, i.e., even if the models achieve human-level accuracies, we cannot prove that they successfully perform reading comprehension. This issue may be attributed to the low interpretability of black-box neural network models. However, a problem is that we cannot explain what is measured by the datasets even if we can interpret the internals of models. We speculate that this benchmarking issue in MRC can be attributed to the following two points: (i) we do not have a comprehensive theoretical basis of reading comprehension for specifying what we should ask (Section 3) and (ii) we do not have a well-established methodology for creating a dataset and for analyzing a model based on it (Section 4).[1] In the remainder of this paper, we argue that these issues can be addressed by using insights from the psychological study of reading comprehension and by implementing psychometric means of validation.

3 Reading Comprehension from Psychology to MRC

3.1 Computational Model in Psychology

Human text comprehension has been studied in psychology for a long time (Kintsch and Rawson, 2005; Graesser et al., 1994; Kintsch, 1988). Connectionist and computational architectures have been proposed for such comprehension including a mechanism pertinent to knowledge activation and memory storing. Among the computational models, the construction–integration (CI) model is the most influential and provides a strong foundation of the field (McNamara and Magliano, 2009).

The CI model assumes three different representation levels as follows:

- *Surface structure* is the linguistic information of particular words, phrases, and syntax obtained by decoding the raw textual input.
- *Textbase* is a set of propositions in the text, where the propositions are locally connected by inferences (*microstructure*).
- *Situation model* is a situational and coherent mental representation in which the propositions are globally connected (*macrostructure*), and it is often grounded to not only texts but also to sounds, images, and background information.

The CI model first decodes textual information (i.e., the surface structure) from the raw textual

---

[1] These two issues loosely correspond to the plausibility and faithfulness of explanation (Jacovi and Goldberg, 2020). The plausibility is linked to what we expect as an explanation, whereas the faithfulness refers to how accurately we explain models' reasoning process.

input, then creates the propositions (i.e., textbase) and their local connections occasionally using the reader's knowledge (*construction*), and finally constructs a coherent representation (i.e., situation model) that is organized according to five dimensions including time, space, causation, intentionality, and objects (Zwaan and Radvansky, 1998), which provides a global description of the events (*integration*). These steps are not exclusive, i.e., propositions are iteratively updated in accordance with the surrounding ones with which they are linked. Although the definition of successful text comprehension can vary, Hernández-Orallo (2017) indicates that comprehension implies the process of creating (or searching for) a situation model that best explains the given text and the reader's background knowledge (Zwaan and Radvansky, 1998). We use this definition to highlight that the creation of a situation model plays a vital role in human reading comprehension.

Our aim in this section is to provide a basis for explaining what reading comprehension is, which requires terms for explanation. In the computational model above, the representation levels appear to be useful for organizing such terms. We ground existing NLP technologies and tasks to different representation levels in the next section.

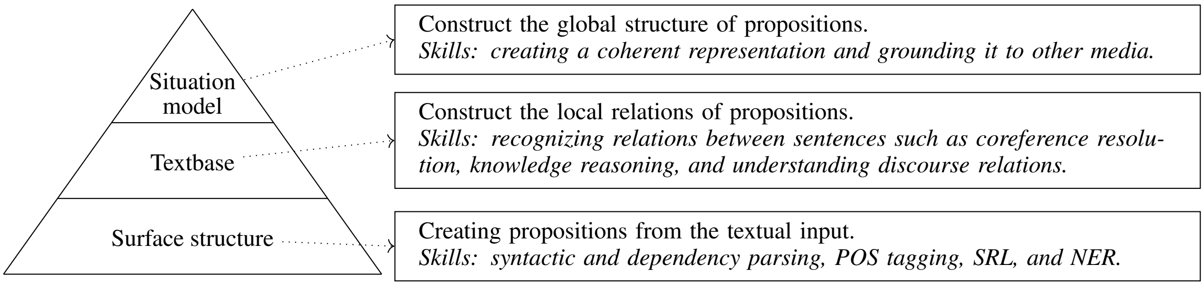
3.2 Skill Hierarchy for MRC

Here, we associate the existing NLP tasks with the three representation levels introduced above. The biggest advantage of MRC is its general formulation, which makes it the most general task for evaluating NLU. This emphasizes the importance of the requirement of various *skills* in MRC, which can serve as the units for the explanation of reading comprehension. Therefore, our motivation is to provide an overview of the skills as a hierarchical taxonomy and to highlight the missing aspects in existing MRC datasets that are required for comprehensively covering the representation levels.

Existing Taxonomies We first provide a brief overview of the existing taxonomies of skills in NLU tasks. For recognizing textual entailment (Dagan et al., 2006), several studies present a classification of reasoning and commonsense knowledge (Bentivogli et al., 2010; Sammons et al., 2010; LoBue and Yates, 2011). For scientific question answering, Jansen et al. (2016) categorize knowledge and inference for an elementary-level dataset. Similarly, Boratko et al. (2018) propose types of knowledge and reasoning for scientific questions in MRC (Clark et al., 2018). A limitation of both these studies is that the proposed sets of knowledge and inference are limited to the domain of elementary-level science. Although some existing datasets for MRC have their own classifications of skills, they are coarse and only cover a limited extent of typical NLP tasks (e.g., word matching and paraphrasing). In contrast, for a more generalizable definition, Sugawara et al. (2017) propose a set of 13 skills for MRC. Rogers et al. (2020) pursue this direction by proposing a set of questions with eight question types. In addition, Schlegel et al. (2020) propose an annotation schema to investigate requisite knowledge and reasoning. Dunietz et al. (2020) propose a template of understanding that consists of spatial, temporal, causal, and motivational questions to evaluate precise understanding of narratives with reference to human text comprehension.

In what follows, we describe the three representation levels that basically follow the three representations of the CI model but are modified for MRC. The three levels are shown in Figure 1. We emphasize that we do not intend to create exhaustive and rigid definitions of skills. Rather, we aim to place them in a hierarchical organization, which can serve as a foundation to highlight the missing aspects in the current MRC.

Surface Structure This level broadly covers the linguistic information and its semantic meaning, which can be based on the raw textual input. Although these features form a proposition according to psychology, it should be viewed as sentence-level semantic representation in computational linguistics. This level includes part-of-speech tagging, syntactic parsing, dependency parsing, punctuation recognition, named entity recognition (NER), and semantic role labeling (SRL). Although these basic tasks can be accomplished by some recent pretraining-based neural language models (Liu et al., 2019), they are hardly required in NLU tasks including MRC. In the natural language inference task, McCoy et al. (2019) indicate that existing datasets (e.g., Bowman et al. (2015)) may fail to elucidate the syntactic understanding of given sentences. Although it is not obvious that these basic tasks should be included in MRC and it is not easy to circumscribe linguistic knowledge from concrete and abstract knowledge (Zaenen et al., 2005; Manning, 2006), we should always care about the capabilities of basic tasks (e.g., use of checklists



Construct the global structure of propositions.
Skills: creating a coherent representation and grounding it to other media.

Construct the local relations of propositions.
Skills: recognizing relations between sentences such as coreference resolution, knowledge reasoning, and understanding discourse relations.

Creating propositions from the textual input.
Skills: syntactic and dependency parsing, POS tagging, SRL, and NER.

Figure 1: Representation levels and corresponding natural language understanding skills.

(Ribeiro et al., 2020)) when the performance of a model is being assessed.

Textbase This level covers local relations of propositions in the computational model of reading comprehension. In the context of NLP, it refers to various types of relations linked between sentences. These relations not only include the typical relations between sentences (discourse relations) but also the links between entities. Consequently, this level includes coreference resolution, causality, temporal relations, spatial relations, text structuring relations, logical reasoning, knowledge reasoning, commonsense reasoning, and mathematical reasoning. We also include multi-hop reasoning (Welbl et al., 2018) at this level because it does not necessarily require a coherent global representation over a given context. For studying the generalizability of MRC, Fisch et al. (2019) propose a shared task featuring training and testing on multiple domains. Talmor and Berant (2019) and Khashabi et al. (2020) also find that training on multiple datasets leads to robust generalization. However, unless we make sure that datasets require various skills with sufficient coverage, it might remain unclear whether we evaluate a model's transferability of the reading comprehension ability.

Situation Model This level targets the global structure of propositions in human reading comprehension. It includes a coherent and situational representation of a given context and its grounding to the non-textual information. A coherent representation has well-organized sentence-to-sentence transitions (Barzilay and Lapata, 2008), which are vital for using procedural and script knowledge (Schank and Abelson, 1977). This level also includes characters' goals and plans, meta perspective including author's intent and attitude, thematic understanding, and grounding to other media. Most existing MRC datasets seem to struggle to target the situation model. We discuss further in Section 5.1.

Passage: *The princess climbed out the window of the high tower and climbed down the south wall when her mother was sleeping. She wandered out a good way. Finally, she went into the forest where there are no electric poles.*

Q1: *Who climbed out of the castle?* A: *Princess*
Q2: *Where did the princess wander after escaping?*
A: *Forest*
Q3: *What would happen if her mother was not sleeping?*
A: *the princess would be caught soon (multiple choice)*

Figure 2: Example questions of the different representation levels. The passage is taken from MCTest.

Example The representation levels in the example shown in Figure 2 are described as follows. Q1 is at the surface-structure level where a reader only needs to understand the subject of the first event. We expect that Q2 requires understanding of relations among described entities and events at the textbase level; the reader may need to understand who *she* means using coreference resolution. *Escaping* in Q2 also requires the reader's commonsense to associate it with the first event. However, the reader might be able to answer this question only by looking for a place (specified by *where*) described in the passage, thereby necessitating the validity of the question to correctly evaluate the understanding of the described events. Q3 is an example that requires imagining a different situation at the situation-model level, which could be further associated with a grounding question such as *which figure best depicts the given passage?*

In summary, we indicate that the following features might be missing in existing datasets:

- Considering the capability to acquire basic understanding of the linguistic-level information.
- Ensuring that the questions comprehensively specify and evaluate textbase-level skills.
- Evaluating the capability of the situation model in which propositions are coherently organized

and are grounded to non-textual information.

Should MRC models mimic human text comprehension? In this paper, we do not argue that MRC models should mimic human text comprehension. However, when we design an NLU task and create datasets for testing human-like linguistic generalization, we can refer to the aforementioned features to frame the intended behavior to evaluate in the task. As Linzen (2020) discusses, the task design is orthogonal to how the intended behavior is realized at the implementation level (Marr, 1982).

4 MRC on Psychometrics

In this section, we provide a theoretical foundation for the evaluation of MRC models. When MRC measures the capability of reading comprehension, validation of the measurement is crucial to obtain a reliable and useful explanation. Therefore, we focus on psychometrics—a field of study concerned with the assessment of the quality of psychological measurement (Furr, 2018). We expect that the insights obtained from psychometrics can facilitate a better task design. In Section 4.1, we first review the concept of validity in psychometrics. Subsequently, in Section 4.2, we examine the aspects that correspond to construct validity in MRC and then indicate the prerequisites for verifying the intended explanation of MRC in its task design.

4.1 Construct Validity in Psychometrics

According to psychometrics, *construct validity* is necessary to validate the interpretation of outcomes of psychological experiments.[2] Messick (1995) report that construct validity consists of the six aspects shown in Table 2.

In the design of educational and psychological measurement, these aspects collectively provide verification questions that need to be answered for justifying the interpretation and use of test scores. In this sense, the construct validation can be viewed as an empirical evaluation of the meaning and consequence of measurement. Given that MRC is intended to capture the reading comprehension ability, the task designers need to be aware of these validity aspects. Otherwise, users of the task cannot justify the score interpretation, i.e., it cannot be confirmed that successful systems actually perform intended reading comprehension.

4.2 Construct Validity in MRC

Table 2 also raises MRC features corresponding to the six aspects of construct validity. In what follows, we elaborate on these correspondings and discuss the missing aspects that are needed to achieve the construct validity of the current MRC.

Content Aspect As discussed in Section 3, sufficiently covering the skills across all the representation levels is an important requirement of MRC. It may be desirable that an MRC model is simultaneously evaluated on various skill-oriented examples.

Substantive Aspect This aspect appraises the evidence for the consistency of model behavior. We consider that this is the most important aspect for explaining reading comprehension, a process that subsumes various implicit and complex steps. To obtain a consistent response from an MRC system, it is necessary to ensure that the questions correctly assess the internal steps in the process of reading comprehension. However, as stated in Section 2.2, most existing datasets fail to verify that a question is solved by using an intended skill, which implies that it cannot be proved that a successful system can actually perform intended comprehension.

Structural Aspect Another issue in the current MRC is that they only provide simple accuracy as a metric. Given that the substantive aspect necessitates the evaluation of the internal process of reading comprehension, the structure of metrics needs to reflect it. However, a few studies have attempted to provide a dataset with multiple metrics. For example, Yang et al. (2018) not only ask for the answers to questions but also provide sentence-level supporting facts. This metric can also evaluate the process of multi-hop reasoning whenever the supporting sentences need to be understood for answering a question. Therefore, we need to consider both substantive and structural aspects.

Generalizability Aspect The generalizability of MRC can be understood from the reliability of metrics and the reproducibility of findings. For the reliability of metrics, we need to take care of the reliability of gold answers and model predictions. Regarding the accuracy of answers, the performance of the model becomes unreliable when the answers are unintentionally ambiguous or impractical. Because the gold answers in most datasets are only decided by the majority vote of crowd workers, the ambiguity of the answers is not considered. It

---

[2] In psychology, a construct is an abstract concept, which facilitates the understanding of human behavior such as vocabulary, skills, and comprehension.

Validity aspects	Definition in psychometrics	Correspondence in MRC
1. Content	Evidence of content relevance, representativeness, and technical quality.	Questions require reading comprehension skills with sufficient coverage and representativeness over the representation levels.
2. Substantive	Theoretical rationales for the observed consistencies in the test responses including task performance of models.	Questions correctly evaluate the intended intermediate process of reading comprehension and provide rationales to the interpreters.
3. Structural	Fidelity of the scoring structure to the structure of the construct domain at issue.	Correspondence between the task structure and the score structure.
4. Generalizability	Extent to which score properties and interpretations can be generalized to and across population groups, settings, and tasks.	Reliability of test scores in correct answers and model predictions, and applicability to other datasets and models.
5. External	Extent to which the assessment scores' relationship with other measures and non-assessment behaviors reflect the expected relations.	Comparison of the performance of MRC with that of other NLU tasks and measurements.
6. Consequential	Value implications of score interpretation as a basis for the consequences of test use, especially regarding the sources of invalidity related to issues of bias, fairness, and distributive justice.	Considering the model vulnerabilities to adversarial attacks and social biases of models and datasets to ensure the fairness of model outputs.

Table 2: Aspects of the construct validity in psychometrics and corresponding features in MRC.

may be useful if such ambiguity can be reflected in the evaluation (e.g., using the item response theory (Lalor et al., 2016)). As for model predictions, an issue may be the reproducibility of results (Bouthillier et al., 2019), which implies that the reimplementation of a system generates statistically similar predictions. For the reproducibility of models, Dror et al. (2018) emphasize statistical testing methods to evaluate models. For the reproducibility of findings, Bouthillier et al. (2019) stress it as the transferability of findings in a dataset/task to another dataset/task. In open-domain question answering, Lewis et al. (2021) point out that successful models might only memorize dataset-specific knowledge. To facilitate this transferability, we need to have units of explanation that can be used in different datasets (Doshi-Velez and Kim, 2018).

External Aspect This aspect refers to the relationship between a model's scores on different tasks. Yogatama et al. (2019) point out that current models struggle to transfer their ability from a task originally trained on (e.g., MRC) to different unseen tasks (e.g., SRL). To develop a general NLU model, one would expect that a successful MRC model should show sufficient performance on other NLU tasks as well. To this end, Wang et al. (2019) propose an evaluation framework with ten different NLU tasks in the same format.

Consequential Aspect This aspect refers to the actual and potential consequences of test use. In MRC, this refers to the use of a successful model in practical situations other than tasks, where we need to ensure the robustness of a model to adversarial attacks and the accountability for unintended model behaviors. Wallace et al. (2019) highlight this aspect by showing that existing NLP models are vulnerable to adversarial examples, thereby generating egregious outputs.

Summary: Design of Rubric Given the validity aspects, our suggestion is to design a *rubric* (scoring guide used in education) of what reading comprehension we expect is evaluated in a dataset; this helps to inspect detailed strengths and weaknesses of models that cannot be obtained only by simple accuracy. The rubric should not only cover various linguistic phenomena (the *content* aspect) but also involve different levels of intermediate evaluation in the reading comprehension process (the *substantive* and *structural* aspects) as well as stress testing of adversarial attacks (the *consequential* aspect). The rubric is in a similar motivation with dataset statements (Bender and Friedman, 2018; Gebru et al., 2018); however, taking the validity aspects into account would improve its substance.

5 Future Directions

This section discusses future potential directions toward answering the *what* and *how* questions in Sections 3 and 4. In particular, we infer that the *situation model* and *substantive validity* are critical for benchmarking human-level MRC.

5.1 *What* Question: Situation Model

As mentioned in Section 3, existing datasets fail to fully assess the ability of creating the situation model. As a future direction, we suggest that the task should deal with two features of the situation model: context dependency and grounding.

5.1.1 Context-dependent Situations

A vital feature of the situation model is that it is conditioned on a given text, i.e., a representation is constructed distinctively depending on the given context. We elaborate it by discussing the two key features: defeasibility and novelty.

Defeasibility The defeasibility of a constructed representation implies that a reader can modify and revise it according to the newly acquired information (Davis and Marcus, 2015; Schubert, 2015). The defeasibility of NLU has been tackled in the task of if-then reasoning (Sap et al., 2019a), abductive reasoning (Bhagavatula et al., 2020), counterfactual reasoning (Qin et al., 2019), or contrast sets (Gardner et al., 2020). A possible approach in MRC is that we ask questions against a set of modified passages that describe slightly different situations, where the same question can lead to different conclusions.

Novelty An example showing the importance of contextual novelty is *Could a crocodile run a steeplechase?* by Levesque (2014). This question poses a novel situation where the solver needs to combine multiple commonsense knowledge to derive the correct answer. If non-fiction documents, such as newspaper and Wikipedia articles, are only used, some questions require only the reasoning of facts already known in web-based corpus. Fictional narratives may be a better source for creating questions on novel situations.

5.1.2 Grounding to Other Media

In MRC, grounding texts to non-textual information is not fully explored yet. Kembhavi et al. (2017) propose a dataset based on science textbooks, which contain questions with passages, diagrams, and images. Kahou et al. (2018) propose a figure-based question answering dataset that requires the understanding of figures including line plots and bar charts. Although another approach could be vision-based question answering tasks (Antol et al., 2015; Zellers et al., 2019a), we cannot directly use them for evaluating NLU because they focus on understanding of images rather than texts. Similarly to the textbook questions (Kembhavi et al., 2017), a possible approach would be to create questions for understanding of texts through showing figures. We might also need to account for the *scope* of grounding (Bisk et al., 2020), i.e., ultimately understanding human language in a social context beyond simply associating texts with perceptual information.

5.2 *How* Question: Substantive Validity

Substantive validity requires us to ensure that the questions correctly assess the internal steps of reading comprehension. We discuss two approaches for this challenge: creating shortcut-proof questions and ensuring the explanation by design.

5.2.1 Shortcut-proof Questions

Gururangan et al. (2018) reveal that NLU datasets can contain unintended dataset biases embedded by annotators. If machine learning models exploit such biases for answering questions, we cannot evaluate the precise NLU of models. Following Geirhos et al. (2020), we define *shortcut-proof* questions as ones that prevent models from exploiting dataset biases and learning decision rules (*shortcuts*) that perform well only on i.i.d. test examples with regard to its training examples. Gardner et al. (2019) also point out the importance of mitigating shortcuts in MRC. In this section, we view two different approaches for this challenge.

Removing Unintended Biases by Filtering Zellers et al. (2018) propose a model-based adversarial filtering method that iteratively trains an ensemble of stylistic classifiers and uses them to filter out the questions. Sakaguchi et al. (2020) also propose filtering methods based on both machines and humans to alleviate *dataset-specific* and *word-association* biases. However, a major issue is the inability to discern knowledge from bias in a closed domain. When the domain is equal to a dataset, patterns that are valid only in the domain are called *dataset-specific* biases (or annotation artifacts in the labeled data). When the domain covers larger corpora, the patterns (e.g., frequency) are called *word-association* biases. When the domain includes everyday experience, patterns are called *commonsense*. However, as mentioned in Section 5.1, commonsense knowledge can be *defeasible*, which implies that the knowledge can be false in unusual situations. In contrast, when the domain is our real world, indefeasible patterns are called *factual knowledge*. Therefore, the distinction of bias and knowledge depends on where the pattern is recognized. This means that a dataset should be created so that it can evaluate reasoning on the intended knowledge. For example, to test defeasible reasoning, we must filter out questions that are solvable by usual commonsense only. If we want to investigate the reading comprehension ability without depending on factual knowledge, we can consider counterfactual or fictional situations.

Identifying Requisite Skills by Ablating Input Features Another approach is to verify shortcut-proof questions by analyzing the human answerability of questions regarding their key features. We speculate that if a question is still answerable by humans even after removing the intended features, the question does not require understanding of the ablated features (e.g., checking the necessity of resolving pronoun coreference after replacing pronouns with dummy nouns). Even if we cannot accurately identify such necessary features, by identifying partial features of them in a sufficient number of questions, we could expect that the questions evaluate the corresponding intended skill. In a similar vein, Geirhos et al. (2020) argue that a dataset is useful only if it is a good proxy for the underlying ability one is actually interested in.

5.2.2 Explanation by Design

Another approach for ensuring the substantive validity is to include explicit explanation in the task formulation. Although gathering human explanations is costly, the following approaches can facilitate the explicit verification of a model's understanding using a few test examples.

Generating Introspective Explanation Inoue et al. (2020) classify two types of explanation in text comprehension: *justification explanation* and *introspective explanation*. The justification explanation only provides a collection of supporting facts for making a certain decision, whereas the introspective explanation provides the derivation of the answer for making the decision, which can cover linguistic phenomena and commonsense knowledge not explicitly mentioned in the text. They annotate multi-hop reasoning questions with introspective explanation and propose a task that requires the derivation of the correct answer of a given question to improve the explainability. Rajani et al. (2019) collect human explanations for commonsense reasoning and improve the system's performance by modeling the generation of the explanation. Although we must take into account the faithfulness of explanation, asking for introspective explanations could be useful in inspecting the internal reasoning process, e.g., by extending the task formulation so that it includes auxiliary questions that consider the intermediate facts in a reasoning process. For example, before answering Q2 in Figure 2, a reader should be able to answer *who escaped?* and *where did she escape from?* at the surface-structure level.

Creating Dependency Between Questions Another approach for improving the substantive validity is to create dependency between questions by which answering them correctly involves answering some other questions correctly. For example, Dalvi et al. (2018) propose a dataset that requires a procedural understanding of scientific facts. In their dataset, a set of questions corresponds to the steps of the entire process of a scientific phenomenon. Therefore, this set can be viewed as a single question that requires a complete understanding of the scientific phenomenon. In CoQA (Reddy et al., 2019), it is noted that questions often have pronouns that refer back to nouns appearing in previous questions. These mutually-dependent questions can probably facilitate the explicit validation of the models' understanding of given texts.

6 Conclusion

In this paper, we outlined current issues and future directions for benchmarking machine reading comprehension. We visited the psychology study to analyze *what* we should ask of reading comprehension and the construct validity in psychometrics to analyze *how* we should correctly evaluate it. We deduced that future datasets should evaluate the capability of the situation model for understanding context-dependent situations and for grounding to non-textual information and ensure the substantive validity by creating shortcut-proof questions and designing an explanatory task formulation.

Acknowledgments

The authors would like to thank Xanh Ho for helping create the dataset list and the anonymous reviewers for their insightful comments. This work was supported by JSPS KAKENHI Grant Number 18H03297, JST ACT-X Grant Number JPMJAX190G, and JST PRESTO Grant Number JPMJPR20C4.

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433.

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Emily M. Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Luisa Bentivogli, Elena Cabrio, Ido Dagan, Danilo Giampiccolo, Medea Lo Leggio, and Bernardo Magnini. 2010. Building textual entailment specialized data sets: a methodology for isolating linguistic phenomena relevant to inference. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Michael Boratko, Xiang Li, Tim O'Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. 2020. ProtoQA: A question answering dataset for prototypical common-sense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1122–1136, Online. Association for Computational Linguistics.

Michael Boratko, Harshit Padigela, Divyendra Mikkilineni, Pritish Yuvraj, Rajarshi Das, Andrew McCallum, Maria Chang, Achille Fokoue-Nkoutche, Pavan Kapanipathi, Nicholas Mattei, Ryan Musa, Kartik Talamadupula, and Michael Witbrock. 2018. A systematic classification of knowledge, reasoning, and context within the ARC dataset. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 60–70. Association for Computational Linguistics.

Xavier Bouthillier, César Laurent, and Pascal Vincent. 2019. Unreproducible research is reproducible. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 725–734, Long Beach, California, USA. PMLR.

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

Christopher J.C. Burges. 2013. Towards the machine comprehension of text: An essay. Technical report, Microsoft Research Technical Report MSR-TR-2013-125.

Vittorio Castelli, Rishav Chakravarti, Saswati Dana, Anthony Ferritto, Radu Florian, Martin Franz, Dinesh Garg, Dinesh Khandelwal, Scott McCarley, Michael McCawley, Mohamed Nasr, Lin Pan, Cezar Pendus, John Pitrelli, Saurabh Pujar, Salim Roukos, Andrzej Sakrajda, Avi Sil, Rosario Uceda-Sosa, Todd Ward, and Rong Zhang. 2020. The TechQA dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1269–1278, Online. Association for Computational Linguistics.

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: An adversarially-authored question answering dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 63–69, Minneapolis, USA. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604. Association for Computational Linguistics.

Pradeep Dasigi, Nelson F. Liu, Ana Marasovic, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5927–5934, Hong Kong, China. Association for Computational Linguistics.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Commun. ACM*, 58(9):92–103.

Bhuwan Dhingra, Kathryn Mazaitis, and William W. Cohen. 2017. Quasar: Datasets for question answering by search and reading.

Finale Doshi-Velez and Been Kim. 2018. *Considerations for Evaluation and Generalization in Interpretable Machine Learning*, 1st edition. Springer International Publishing.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Jesse Dunietz, Greg Burnham, Akash Bharadwaj, Owen Rambow, Jennifer Chu-Carroll, and Dave Ferrucci. 2020. To test machine comprehension, start by defining comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7839–7859, Online. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728. Association for Computational Linguistics.

James Ferguson, Matt Gardner, Hannaneh Hajishirzi, Tushar Khot, and Pradeep Dasigi. 2020. IIRC: A dataset of incomplete information reading comprehension questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1137–1147, Online. Association for Computational Linguistics.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

R Michael Furr. 2018. *Psychometrics: an introduction*. Sage Publications.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Matt Gardner, Jonathan Berant, Hannaneh Hajishirzi, Alon Talmor, and Sewon Min. 2019. On making reading comprehension more comprehensive. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 105–112, Hong Kong, China. Association for Computational Linguistics.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2018. Datasheets for datasets. ArXiv preprint 1803.09010.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. 2018. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE.

Arthur C. Graesser, Murray Singer, and Tom Trabasso. 1994. Constructing inferences during narrative text comprehension. *Psychological review*, 101(3):371.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112. Association for Computational Linguistics.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1930–1940, New Orleans, Louisiana. Association for Computational Linguistics.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

José Hernández-Orallo. 2017. *The measure of all minds: evaluating natural and artificial intelligence*. Cambridge University Press.

Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. WikiReading: A novel large-scale language understanding task over wikipedia. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1545, Berlin, Germany. Association for Computational Linguistics.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *International Conference on Learning Representations*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. 2020. R4C: A benchmark for evaluating RC systems to get the right answer for the right reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750, Online. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the*

58th Annual Meeting of the Association for Computational Linguistics, pages 4198–4205, Online. Association for Computational Linguistics.

Peter Jansen, Niranjan Balasubramanian, Mihai Surdeanu, and Peter Clark. 2016. What's in an explanation? characterizing knowledge and inference requirements for elementary science exams. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2956–2965, Osaka, Japan. The COLING 2016 Organizing Committee.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2011–2021. Association for Computational Linguistics.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611. Association for Computational Linguistics.

Samira Ebrahimi Kahou, Adam Atkinson, Vincent Michalski, Ákos Kádár, Adam Trischler, and Yoshua Bengio. 2018. FigureQA: An annotated figure dataset for visual reasoning. In *International Conference on Learning Representations Workshop Track*.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015. Association for Computational Linguistics.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *the IEEE Conference on Computer Vision and Pattern Recognition*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UnifiedQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. QASC: A dataset for question answering via sentence composition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8082–8090. AAAI Press.

Walter Kintsch. 1988. The role of knowledge in discourse comprehension: A construction-integration model. *Psychological review*, 95(2):163.

Walter Kintsch and Katherine A Rawson. 2005. Comprehension. *The Science of Reading: A Handbook*, pages 211–226.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Igor Labutov, Bishan Yang, Anusha Prakash, and Amos Azaria. 2018. Multi-relational question answering from narratives: Machine reading and reasoning in simulated worlds. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 833–844. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 796–805. Association for Computational Linguistics.

John Lalor, Hao Wu, and Hong Yu. 2016. Building an evaluation scale using item response theory. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 648–657, Austin, Texas. Association for Computational Linguistics.

Hector J. Levesque. 2014. On our best behaviour. *Artificial Intelligence*, 212:27 – 35.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 58–62, Hong Kong, China. Association for Computational Linguistics.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, Online. Association for Computational Linguistics.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3622–3628. International Joint Conferences on Artificial Intelligence Organization. Main track.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter LoBue and Alexander Yates. 2011. Types of common-sense knowledge needed for recognizing textual entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 329–334, Portland, Oregon, USA. Association for Computational Linguistics.

Kaixin Ma, Tomasz Jurczyk, and Jinho D. Choi. 2018. Challenging reading comprehension on daily conversation: Passage completion on multiparty dialog. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2039–2048, New Orleans, Louisiana. Association for Computational Linguistics.

Christopher D. Manning. 2006. Local textual inference: It's hard to circumscribe, but you know it when you see it—and NLP needs it. Unpublished manuscript.

David Marr. 1982. *Vision: A computational investigation into the human representation and processing of visual information*. New York: Freeman.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Danielle S McNamara and Joe Magliano. 2009. Toward a comprehensive model of comprehension. *Psychology of learning and motivation*, 51:297–384.

Samuel Messick. 1995. Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *American psychologist*, 50(9):741.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1725–1735. Association for Computational Linguistics.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51. Association for Computational Linguistics.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association* *for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020. TORQUE: A reading comprehension dataset of temporal ordering questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1158–1172, Online. Association for Computational Linguistics.

Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2016. Who did What: A large-scale person-centered cloze dataset. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2230–2235. Association for Computational Linguistics.

Simon Ostermann, Ashutosh Modi, Michael Roth, Stefan Thater, and Manfred Pinkal. 2018. MCScript: A novel dataset for assessing machine comprehension using script knowledge. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA).

Simon Ostermann, Michael Roth, and Manfred Pinkal. 2019. MCScript2.0: A machine comprehension corpus focused on script events and participants. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 103–117, Minneapolis, Minnesota. Association for Computational Linguistics.

Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. 2018. emrQA: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2357–2368, Brussels, Belgium. Association for Computational Linguistics.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789. Association for Computational Linguistics.

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

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203. Association for Computational Linguistics.

A Rogers, O Kovaleva, M Downey, and A Rumshisky. 2020. Getting closer to AI complete question answering: A set of prerequisite real tasks. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8722–8731.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2087–2097, Brussels, Belgium. Association for Computational Linguistics.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WINOGRANDE: an adversarial winograd schema challenge at scale. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8732–8430.

Mark Sammons, V.G.Vinod Vydiswaran, and Dan Roth. 2010. "Ask not what textual entailment can do for you...". In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1199–1208. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum.

Viktor Schlegel, Marco Valentino, Andre Freitas, Goran Nenadic, and Riza Batista-Navarro. 2020. A framework for evaluation of machine reading comprehension gold standards. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5359–5369, Marseille, France. European Language Resources Association.

Lenhart K Schubert. 2015. What kinds of knowledge are needed for genuine understanding? In *IJCAI 2015 Workshop on Cognitive Knowledge Acquisition and Applications (Cognitum 2015)*.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219. Association for Computational Linguistics.

Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 806–817. Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8918–8927.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. DREAM: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231.

Simon Suster and Walter Daelemans. 2018. CliCR: a dataset of clinical case reports for machine reading comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1551–1563. Association for Computational Linguistics.

Richard Sutcliffe, Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Corina Forascu, Yassine Benajiba, and Petya Osenova. 2013. Overview of QA4MRE main task at CLEF 2013. *Working Notes, CLEF*.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4911–4921, Florence, Italy. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong

Kong, China. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32*, pages 3261–3275. Curran Associates, Inc.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards AI-complete question answering: a set of prerequisite toy tasks. In *International Conference on Learning Representations*.

Qizhe Xie, Guokun Lai, Zihang Dai, and Eduard Hovy. 2018. Large-scale cloze test dataset created by teachers. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2344–2356, Brussels, Belgium. Association for Computational Linguistics.

Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. 2018. RecipeQA: A challenge dataset for multimodal comprehension of cooking recipes. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1368, Brussels, Belgium. Association for Computational Linguistics.

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

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. 2019. Learning and evaluating general linguistic intelligence. *CoRR*, abs/1901.11373.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. ReClor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*.

Annie Zaenen, Lauri Karttunen, and Richard Crouch. 2005. Local textual inference: Can it be defined or circumscribed? In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 31–36, Ann Arbor, Michigan. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019b. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the gap between human and machine commonsense reading comprehension. ArXiv preprint 1810.12885.

Rolf A Zwaan and Gabriel A Radvansky. 1998. Situation models in language comprehension and memory. *Psychological bulletin*, 123(2):162.

A Machine Reading Comprehension Datasets

Tables 3, 4, 5, and 6 list machine reading comprehension and related datasets along with their answer styles, dataset size, type of corpus, sourcing methods, and focuses.

Dataset	Answer style	Size	Corpus	Question source	Focus
QA4MRE (Sutcliffe et al., 2013)	multiple-choice	240	technical document	expert	exam-level questions
MCTest (Richardson et al., 2013)	multiple-choice	2.6K	written story	crowd	children-level narrative
bAbI (Weston et al., 2015)	descript	10K * 20	generated text	automated	toy tasks for prerequisite skills
CNN/ DailyMail (Hermann et al., 2015)	extract	1.4M	news article	automated	entity cloze
Children's Book Test (Hill et al., 2016)	extract	688K	narrative	automated	large-scale automated
SQuAD 1.1 (Rajpurkar et al., 2016)	extract	100K	Wikipedia	crowd	large-scale crowdsourced
LAMBADA (Paperno et al., 2016)	descript	10K	narrative	crowd	hard language modeling
WikiReading (Hewlett et al., 2016)	descript	18m	Wikipedia	automated	Wikidata articles
Who did What (Onishi et al., 2016)	multiple-choice	200K	news article	automated	cloze of person names
MS MARCO (Nguyen et al., 2016)	descript	100K	web snippet	query	description on web snippets
NewsQA (Trischler et al., 2017)	extract	120K	news article	crowd	blindly created questions
SearchQA (Dunn et al., 2017)	extract	140K	web snippet	trivia	49.6 snippets on average
RACE (Lai et al., 2017)	multiple-choice	100K	language exam	expert	middle and high school English exam in China
Story Cloze Test (Mostafazadeh et al., 2017)	multiple-choice	3.7K	written story	crowd	98,159 stories for training
TriviaQA (Joshi et al., 2017)	extract	650K	web snippet	trivia	trivia questions
Quasar (Dhingra et al., 2017)	extract	80K	web snippet	query	search queries
TextbookQA (Kembhavi et al., 2017)	multiple-choice	26K	textbook	expert	figures included
AddSent SQuAD (Jia and Liang, 2017)	extract	3.6K	Wikipedia	crowd	distracting sentences injected

Table 3: Machine reading comprehension datasets published until 2017. In the *answer style* column, *descript* represents description (free-form answering) and *extract* denotes answer extraction by selecting a span in given texts. *Size* indicates the size of the whole dataset including training, development, and test sets. In the *question source* column, *crowd* indicates questions written by crowdworkers and *query* indicates questions collected from search-engine queries.

Dataset	Answer style	Size	Corpus	Question source	Focus
ARCT (Habernal et al., 2018)	multiple-choice	2.0K	debate article	crowd expert	reasoning on argument
QAngaroo (Welbl et al., 2018)	multiple-choice	50K	Wikipedia, MEDLINE	automated	multi-hop reasoning
CLOTH (Xie et al., 2018)	multiple-choice	99K	various	expert	cloze in exam texts
NarrativeQA (Kočiský et al., 2018)	descript	45K	movie script	crowd	summary and full story tasks
MCScript (Ostermann et al., 2018)	multiple-choice	30K	written story	crowd	commonsense reasoning, script knowledge
CliCR (Suster and Daelemans, 2018)	extract	100K	clinical case text	automated	cloze style queries
ARC (Clark et al., 2018)	multiple-choice	8K	science exam	expert	retrieved documents from textbooks
DuoRC (Saha et al., 2018)	extract	186K	movie script	crowd	commonsense reasoning, multi-sentence reasoning
ProPara (Dalvi et al., 2018)	extract	2K	science exam	automated	procedural understanding
DuReader (He et al., 2018)	descript	200K	web snippet	query crowd	Chinese, Baidu Search/Knows
MultiRC (Khashabi et al., 2018)	multiple-choice	6K	various documents	crowd	multi-sentence reasoning
Multi-party Dialog (Ma et al., 2018)	extract	13K	TV show transcript	automated	1.7k crowd dialogues, cloze query
SQuAD 2.0 (Rajpurkar et al., 2018)	extract no answer	100K	Wikipedia	crowd	unanswerable questions
ShARC (Saeidi et al., 2018)	yes/no/ irrelevant	32K	web snippet	crowd	reasoning on rules from government documents
QuAC (Choi et al., 2018)	extract yes/no	100K	Wikipedia	crowd	dialogue-based, 14k dialogues
Textworlds QA (Labutov et al., 2018)	extract	1.2M	generated text	automated	simulated worlds, logical reasoning
SWAG (Zellers et al., 2018)	multiple-choice	113K	video captions	language-model	commonsense reasoning
emrQA (Pampari et al., 2018)	extract	400K	clinical documents	automated	using annotated logical forms on i2b2 dataset
HotpotQA (Yang et al., 2018)	extract yes/no	113K	Wikipedia	crowd	multi-hop reasoning
OpenbookQA (Mihaylov et al., 2018)	multiple-choice	6.0K	textbook	crowd	commonsense reasoning
RecipeQA (Yagcioglu et al., 2018)	multiple-choice	36K	recipe script	automated	multimodal questions
ReCoRD (Zhang et al., 2018)	extract	120K	news article	crowd	commonsense reasoning, cloze query

Table 4: Machine reading comprehension datasets published in 2018. In the *answer style* column, *descript* represents description (free-form answering) and *extract* denotes answer extraction by selecting a span in given texts. *Size* indicates the size of the whole dataset including training, development, and test sets. In the *question source* column, *crowd* indicates questions written by crowdworkers and *query* indicates questions collected from search-engine queries.

Dataset	Answer style	Size	Corpus	Question source	Focus
CoQA (Reddy et al., 2019)	extract yes/no	127K	Wikipedia	crowd	dialogue-based, 8k dialogues
Commonsense QA (Talmor et al., 2019)	multiple-choice	12K	ConceptNet	crowd	commonsense reasoning
Natural Questions (Kwiatkowski et al., 2019)	extract yes/no	323K	Wikipedia	query crowd	short or long answer styles
DREAM (Sun et al., 2019)	multiple-choice	10K	language exam	expert	dialogue-based, 6.4k multi-party dialogues
DROP (Dua et al., 2019)	descript	96K	Wikipedia	crowd	discrete reasoning
SocialIQA (Sap et al., 2019b)	multiple-choice	38K	crowd	crowd	commonsense reasoning about social situation
BoolQ (Clark et al., 2019)	yes/no	16K	Wikipedia	query crowd	boolean questions, subset of Natural Questions
MSCript 2.0 (Ostermann et al., 2019)	multiple-choice	20K	narrative	crowd	commonsense reasoning, script knowledge
HellaSWAG (Zellers et al., 2019b)	multiple-choice	70K	web snippet	language-model	commonsense reasoning, WikiHow and ActivityNet
CODAH (Chen et al., 2019)	multiple-choice	2.8K	written prompt	expert	adversarial collection
Quoref (Dasigi et al., 2019)	extract	24K	Wikipedia	crowd	coreference resolution
CosmosQA (Huang et al., 2019)	multiple-choice	36K	narrative	crowd	commonsense reasoning
PubMedQA (Jin et al., 2019)	yes/no	273.5K	PubMed	expert automated	biomedical domain, 1k expert questions
ROPES (Lin et al., 2019)	extract	14K	textbook Wikipedia	crowd	paragraph effects in situations

Table 5: Machine reading comprehension datasets published in 2019. In the *answer style* column, *descript* represents description (free-form answering) and *extract* denotes answer extraction by selecting a span in given texts. *Size* indicates the size of the whole dataset including training, development, and test sets. In the *question source* column, *crowd* indicates questions written by crowdworkers and *query* indicates questions collected from search-engine queries.

Dataset	Answer style	Size	Corpus	Question source	Focus
QuAIL (Rogers et al., 2020)	multiple-choice	15K	various	crowd	prerequisite real tasks
QASC (Khot et al., 2020)	multiple-choice	10K	textbook	crowd	knowledge composition
AdversarialQA (Bartolo et al., 2020)	extract	36K	Wikipedia	crowd	adversarial collection
ReClor (Yu et al., 2020)	multiple-choice	6.1K	exam	expert	logical reasoning
R^{4}C (Inoue et al., 2020)	extract descript	5K	Wikipedia	crowd	multi-hop reasoning
TechQA (Castelli et al., 2020)	descript	1.4K	tech documents	crowd	tech forum questions
LogiQA (Liu et al., 2020)	multiple-choice	8.7K	exam	expert	logical reasoning
ProtoQA (Boratko et al., 2020)	descript	9.8K	web snippet	crowd	commonsense reasoning over prototypical situtations
IIRC (Ferguson et al., 2020)	descript	13K	Wikipedia	crowd	incomplete information
HybridQA (Chen et al., 2020)	extract	70K	Wikipedia	crowd	understanding tabular data
TORQUE (Ning et al., 2020)	extract	21K	TempEval-3	crowd	temporal ordering
2WikiMultiHopQA (Ho et al., 2020)	extract yes/no descript	200K	Wikipedia	automated	multi-hop reasoning

Table 6: Machine reading comprehension datasets published in 2020. In the *answer style* column, *descript* represents description (free-form answering) and *extract* denotes answer extraction by selecting a span in given texts. *Size* indicates the size of the whole dataset including training, development, and test sets. In the *question source* column, *crowd* indicates questions written by crowdworkers and *query* indicates questions collected from search-engine queries.

Multilingual Neural Machine Translation with Deep Encoder and Multiple Shallow Decoders

Xiang Kong[‡][*], **Adithya Renduchintala**[†], **James Cross**[†],
Yuqing Tang[†], **Jiatao Gu**[†], **Xian Li**[†]
[‡]Language Technologies Institute, Carnegie Mellon University
[†]Facebook AI
`xiangk@cs.cmu.edu`
`{adirendu,jcross,yuqtang,jgu,xianl}@fb.com`

Abstract

Recent work in multilingual translation advances translation quality surpassing bilingual baselines using deep transformer models with increased capacity. However, the extra latency and memory costs introduced by this approach may make it unacceptable for efficiency-constrained applications. It has recently been shown for bilingual translation that using a deep encoder and shallow decoder (DESD) can reduce inference latency while maintaining translation quality, so we study similar speed-accuracy trade-offs for multilingual translation. We find that for many-to-one translation we can indeed increase decoder speed without sacrificing quality using this approach, but for one-to-many translation, shallow decoders cause a clear quality drop. To ameliorate this drop, we propose a deep encoder with *multiple* shallow decoders (DEMSD) where each shallow decoder is responsible for a disjoint subset of target languages. Specifically, the DEMSD model with 2-layer decoders is able to obtain a **1.8x speedup** on average compared to a standard transformer model with no drop in translation quality.

1 Introduction

Encoder-decoder based neural machine translation (NMT) systems have achieved great success on bilingual translation tasks (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014; Gehring et al., 2017; Vaswani et al., 2017). Recently, multilingual neural machine translation (MNMT) has also attracted much attention because of its ease of deployment, knowledge transfer among languages and the potential to enable zero-shot translation (Dong et al., 2015; Firat et al., 2016; Ha et al., 2016; Johnson et al., 2017; Arivazhagan et al., 2019; Zhang et al., 2020). While MNMT can support translations in several directions, not all of them have better performance when compared to their corresponding bilingual models. Suspecting that poor performance in some directions is due to the limited model capacity, many prior works adopt deeper encoder and decoder (Zhang et al., 2019; Wang et al., 2019; Zhang et al., 2020). However, increasing the number of layers, especially in the decoder, deteriorates the latency of translation and memory costs. Recently, Kasai et al. (2020) show that given a fixed capacity budget, as measured by the number of layers, models with a deep encoder and a shallow decoder (DESD) are faster at inference time when compared to standard models with an equal number of encoder and decoder layers while maintaining translation quality.

Inspired by findings from Kasai et al. (2020), in this work, we explore the speed-accuracy trade-off in multilingual machine translation systems. Given the same model capacity budget, we experiment various layer allocation strategies. We analyze multilingual models in the one-to-many (O2M) setting and many-to-one (M2O) setting. In the one-to-many setting, there are numerous target languages from a single source language (limited to English in this study); and in the many-to-one setting, several possible source languages are translated into a single target language (again, English in this study).

In the many-to-one scenario, we find that allocating more capacity to the encoder reduces the latency while achieving comparable performance. We hypothesize that a deeper encoder helps the model accommodate multiple source languages, while a shallow decoder is sufficient to support a single target language.

However, in the one-to-many translation setting, speed-accuracy trade-off is complicated. We observe a performance drop as the decoder depth is reduced. We hypothesize that the shallow decoder

---

[*] Work done at Facebook AI.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1613–1624
April 19 - 23, 2021.

can no longer model several different target languages adequately. With the goal of obtaining low latency while maintaining translation quality, we propose using *multiple* shallow decoders where each decoder is responsible for a subset of the target languages. Clearly, the introduction of multiple shallow decoders increases the size of our model. However, at inference time only one shallow decoder will be used, thus not adding latency or memory costs. With multiple target languages and decoders, one natural question is how to assign each target language to one of these decoders. We investigate several methods to assign each target language to one of these shallow decoders. More details are in the Section 3. Experimental results on three multilingual translation corpora show the effectiveness of our method to improve translation accuracy with lower latency at the same time.

Our main contributions are summarized as follows:

- We extend the speed-accuracy trade-off study of DESD models from bilingual to multilingual machine translation tasks with various layer allocations.
- We show that on many-to-one translation, multilingual DESD models enable 1.8x speedup on average without sacrificing performance comparing to the baseline (equal model capacity).
- We further proposed shared encoder and *multiple* shallow decoders (DEMSD) for one-to-many setting again achieving 1.8x speed-up in decoding while preserving high-quality translations at the same time.

2 Deep encoder and shallow decoder (DESD) for multilingual NMT

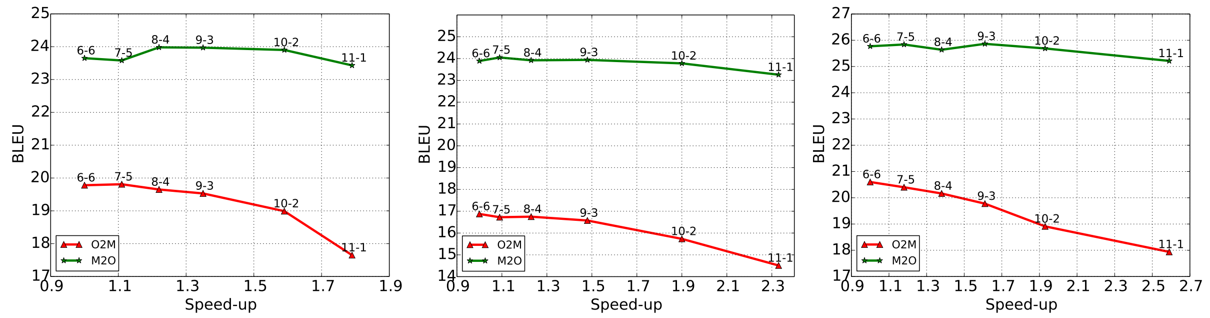
Background The transformer-based NMT model (Vaswani et al., 2017) achieves state-of-the-art performance on many translation tasks. It consists of an encoder and a decoder, each of which contains several stacked layers. Since the transformer relies entirely on the attention mechanism, it allows more parallelization compared to recurrent neural networks. Specifically, at training time, the computation can be parallelized both in the encoder and decoder. At inference time, due to the auto-regressive property, the decoder needs to generate tokens one by one. However, the computation in the encoder is still parallelized given the source sentence. Therefore, the main latency of the transformer at inference time happens in the decoder, especially translating long sentences. Recently, Kasai et al. (2020) find that on bilingual machine translation tasks, putting more capacity of the transformer model to the encoder substantially reduces the decoding time and maintain the performance at the same time.

Because this deep encoder and shallow decoder model achieves a superior speed-accuracy trade-off on bilingual translation tasks, in this section, we try to understand the layer allocations of transformer on the multilingual neural machine translation task given the same capacity budget which is measured by the number of layers in the encoder and decoder. We first experiment with three multilingual translation corpora.

- ML50 (Tang et al., 2020): a large-scale multilingual translation dataset which contains 49 languages↔English and more than 200 million training sentence pairs in total. All data are collected from open-resource data such as WMT, IWSLT, WAT, TED, etc.
- TED8-Related (Wang et al., 2020): 4 low resource languages (Azerbaijani: az, Belarusian: be, Glacian: gl, Slovak: sk) and 4 relevant high resource languages (Turkish: tr, Russian: ru, Portuguese: pt, Czech: cs)
- TED8-Diverse (Wang et al., 2020): 8 languages without consideration for relatedness (Bosnian: bs, Marathi: hr, Hindi: hi, Macedonian: mk, Greek: el, Bulgarian: bg, French: fr, Korean: ko)

Instead of just trying the shallowest possible decoder (1-layer), we train models with various configurations on each of these three corpora. Other than the layer allocation, all the other hyper-parameters and model configurations are the same among these models and the same training procedure is applied to these models (odel and training details are listed in Appendix A.). To understand the speed and accuracy trade-off of the layer allocation, two metrics are reported:

- BLEU: the average tokenized BLEU score (Papineni et al., 2002) over all directions.
- DS: the decoding speed. It is measured by the number of tokens per second the system



(a) ML50 (b) TED8-Related (c) TED8-Diverse

Figure 1: Speed and accuracy trade-off of various layer allocations for O2M and M2O translations on ML50, TED8-Related and TED8-Diverse corpora. X-Y denotes X and Y layers in the encoder and decoder respectively. Best viewed in color.

translates given one sentence at a time on a single GPU.

The results are shown in Figure 1. Models with fewer decoder layers obtain higher decoding speed.

2.1 Many-to-one translation

In the M2O translation, there is no significant performance difference among these layer allocations. We hypothesize that this is because the deeper encoder learns better representations from a large number of source languages while on the decoder side only one language needs to be modeled. Therefore, given a more robust representation of source languages, the shallow decoder is able to generate high-quality translations. For example, the model with 10 encoder layers and 2 decoder layers obtain slightly better performance and a 1.8**x** speedup at the same time.

2.2 One-to-many translation

However, in the O2M translation setting, although models with the shallower decoder have lower latency compared to the standard transformer (6-6), there is a clear performance drop in terms of translation accuracy, especially for models with just 1 or 2 decoder layers. We attribute this to the shallow decoder not having enough capacity to model a large number of target languages.

3 Deep Encoder and Multiple Shallow Decoders (DEMSD)

We have seen that in one-to-many translation, DESD models have a performance drop compared to the standard transformer. In order to preserve translation quality and low latency at the same time, we propose a model with a shared encoder and *multiple* shallow decoders (DEMSD), each of which is used to decode a subset of target languages. Although this will introduce more parameters, at inference time only one shallow decoder is needed for a given translation (since the output language is fixed) thus the model incurs no extra latency or memory costs. One natural question that arises when using this multiple-decoder approach is how to assign output languages to each of the decoders. In this section, we explore several language assignment methods to assign each target language (or language group) to one of these multiple decoders. As a result, each decoder only needs to handle a disjoint subset of target languages.

3.1 One language per decoder (EACH)

The simplest way is to use a separate decoder for each output language. As a result, we will have as many decoders as the number of target languages and each decoder only needs to model one language.

3.2 Random language set per decoder (RAND)

In this method, we assign a random set of languages to a single decoder. As the performance of the model will vary significantly based on the random assignment, we repeat this scheme with three different random assignments and report the average results. Instead of completely random grouping languages, we let each decoder handle a same number of languages but languages in one decoder are randomly grouped.

Language Family	Languages
TURKIC	az, tr
SLAVIC	be, ru
ROMANCE	gl, pt
CZECH–SLOVAK	sk, cs

Table 1: Language families in the TED8-Related corpus.

3.3 One language family per decoder (FAM)

Another intuitive way for language assignment is to use linguistic features (Comrie, 1989; Lewis, 2009; Dryer and Haspelmath, 2013), such as language family, typology, etc. In this method, we are guided by the intuition that languages from the same linguistic family share similar features which might be captured by a single decoder resulting in better performance. Thus, we group languages into several sets based on their linguistic families, and assign a family of languages to each decoder. As a result, we will have as many decoders as the number of language families in the target languages. We expect that in the same decoder, a better knowledge transfer will happen among languages in the same language family. For example, in the TED8-Related corpus, 8 target languages are split into 4 languages families which are TURKIC, SLAVIC, ROMANCE and CZECH–SLOVAK. The details are shown in Table 1. The language family-based assignment results on other corpora are shown in Appendix B.

3.4 Pre-trained language embedding based assignment (EMB)

From Johnson et al. (2017), a common way to indicate the target language is prepending a target language token to the source sentence. With the goal of capturing the information of languages they represent, their embeddings are trained end-to-end with source-target sentence pairs. We call these embeddings as the language embeddings here. According to Johnson et al. (2017), these language embeddings are able to capture target language features in their training data. Therefore, we first extract them from a well-trained model and group target languages according to them. Finally, each group are assigned to one of these decoders.

3.5 Self-taught assignment (ST)

One disadvantage of the pre-trained language embedding based grouping method is the need of a pre-trained machine translation model. It would be better if the model assign each target languages to one of these multiple shallow decoders during the training automatically. We expect that given a fixed number of decoders and target languages, the model is capable of choosing the most appropriate decoders for each language.

Specifically, our model consists of a shared encoder, E, and N multiple decoders, $D = [D_1, D_2, ..., D_N]$. Given a language L, the model will choose a decoder, D_i for training and translation so that the log probability of output sequence y given the input sequence x is $\log p(y|x, E, D_i)$ where i=$\arg\max_j p(j|L_e)$ and $p(\cdot|L_e)$ is the probability of each decoder being chosen given the language L and its language embedding vector L_e. Intuitively, our model will learn the distribution of each decoder being chosen given a language and choose the one with the highest probability. However, the $\arg\max$ operation here is non-differenetiable thus during trainiing we consider the Gumbel-Softmax (Jang et al., 2016), a differentiable approximation of the $\arg\max$ operation.

In Gumbel-Softmax, it models the $p(j|L_e)$ as:

$$p(j|L_e) = \frac{\exp(l_j + g_j)/\tau}{\sum_{k=1}^{N} \exp(l_k + g_k)/\tau} \quad (1)$$

where l is the logit and g=$-\log(-\log(u))$ and $u \sim \mathcal{U}(0, 1)$. In the forward pass, the differentiable approximation of the $\arg\max$ operation is used to choose the decoder for the input language and during the backward, the true gradient of the Straight-Through Gumbel-Softmax outputs is used. In our experiments, the temperature τ is linearly reduced from 5 to 0.5. Finally, during training, the probability of the target sequence y given the source sentence x and multiple decoders D is:

$$p(y|x) = \sum_{n=1}^{N} p(n|L_e) p_n(y|x) \quad (2)$$

where $p_n(y|x)$ is the probability of y given x in the n-th decoder and $p(n|L_e)$ is the probability of the n-th decoder being sampled given the embedding of the language L. During inference, only the decoder with the highest probability will be used to decode the input sentences.

4 Experiments

4.1 Experimental setup

We conduct experiments on ML50, TED8-Related, TED8-Diverse multilingual machine translation

Method	Abbrev.
One language per decoder	EACH
Random language set	RAND
One language family per decoder	FAM
Pre-trained Language embedding	EMB
Self-taught	ST

(a) The abbreviations of language assignment methods.

Metric	Abbrev.
# parameters at training time	#TP
# parameters at inference time	#DP
Decoding speed	DS
# decoders	#DEC

(b) The abbreviations of metrics.

Table 2: The abbreviations of language assignment methods and metrics.

corpora. ML50 (Tang et al., 2020) is an English-central translation benchmark of 50 languages with publicly available training and evaluation sets, including high, mid, and extremely low resource directions. Following Tang et al. (2020), we adopt the 250k SentencePiece model (Kudo and Richardson, 2018) used in XLM-R (Conneau et al., 2019) to tokenize the dataset so that all languages share the same vocabulary. For TED8-Related, TED8-Diverse corpora, we follow the preprocessing steps in Wang et al. (2020).

Hyperparameters On ML50, we follow most of the standard hyperparameters in the transformer-base (Vaswani et al., 2017): 8 attention heads per layer, 512 model dimensions, 2048 hidden dimensions and 0.1 dropout. We train batches of 64k tokens using Adam (Kingma and Ba, 2014) with $\beta = (0.9, 0.98)$ and $\epsilon = 10^{-6}$ and 0.1 label smoothing. The learning rate goes to $1e{-}3$ within 4,000 steps, and then decays with the inverse square-root schedule. All models are trained for 100,000 steps. Furthermore, to mitigate the training data imbalance issue, the temperature sampling method is adopted (Arivazhagan et al., 2019) which is set as 5 in all experiments.

On TED8 corpora, a smaller transformer model with 512 model dimensions, 1024 hidden dimensions and 0.3 dropout is adopted. All models are trained for 40k steps with batches of 16k tokens with a smaller learning rate $2e{-}4$. The other training procedure is the same as the ML50.

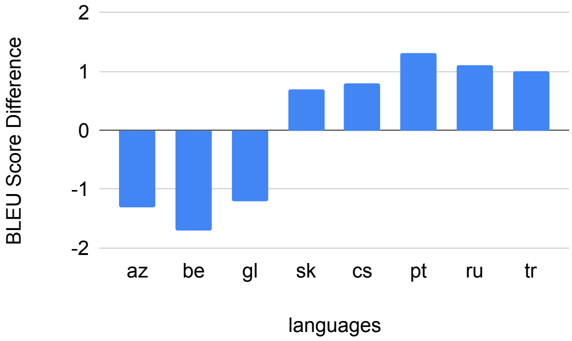


Figure 2: The BLEU score difference between models 10-2-EACH and 10-2 on TED8-Related ($\text{BLEU}_{\text{10-2-EACH}} - \text{BLEU}_{\text{10-2}}$). (Left four languages are low-resourced and the right four are high-resourced.)

Evaluation metrics For all models, we evaluate on the checkpoint with the best validation loss and use beam size 5 and length penalty 1.0 in decoding. Besides reporting the average BLEU score over all languages, on ML50, we predefine high ($>$ 1M), mid (100K, 1M]) and low ($<$ 100K) resource languages according to their data sizes and average BLEU scores on each of them are also computed. For the evaluation speed, **DS**, it is measured by the number of tokens the system translates per second given one sentence at a time on a single GPU.

4.2 Results

From Figure 1, we find that for O2M translation, models with 1- or 2-layer decoders have a clear performance drop compared to the standard transformer (6-6). Therefore, our main experiments adopt multiple shallow decoders with 1 and 2 decoder layers. Results on ML50 and TED8 corpora are shown in Table 3 and 4 respectively. For simplicity, we introduce the abbreviation of each language assignment method and evaluating metrics in Table 2.

One language per decoder (EACH) With this assignment method, models obtain superior performance on high and mid resource languages but poor results on low resource languages. On ML50, if each language has its own decoder, we find that it achieves great results on high resource languages (BLEU_H in rows 2 vs. 3 and 8 vs. 9 in Table 3). We think that given enough training data, the shallow decoder has enough ability to model one language. However, it performs worse on the low resource languages compared with the baseline (BLEU_L between rows 2 vs. 3 and 8 vs. 9 in Table 3). To further understand this assignment method, we also

#	Model	BLEU	BLEU_H	BLEU_M	BLEU_L	DS	#TP (M)	#DP (M)	#Dec
1	6-6	19.68	19.60	18.99	**20.34**	1.0**x**	172	172	1
2	11-1	17.65	16.93	17.02	18.70	1.8**x**	167	167	1
3	11-1-EACH	17.83	19.18	18.32	16.47	1.8**x**	368	167	49
4	11-1-RAND	17.96	17.96	17.54	18.17	1.8**x**	230	167	15
5	11-1-FAM	18.34	18.25	17.92	18.79	1.8**x**	230	167	15
6	11-1-EMB	18.19	18.17	17.79	18.40	1.8**x**	230	167	15
7	11-1-ST	18.47	18.31	18.02	18.79	1.8**x**	230	167	15
8	10-2	18.99	18.50	18.41	19.72	1.6**x**	168	168	1
9	10-2-EACH	18.93	**21.45**	**19.42**	16.78	1.6**x**	572	168	49
10	10-2-RAND	19.24	19.64	18.76	19.24	1.6**x**	294	168	15
11	10-2-FAM	19.70	20.01	19.31	19.81	1.6**x**	294	168	15
12	10-2-EMB	19.64	20.08	19.01	19.94	1.6**x**	294	168	15
13	10-2-ST	**19.71**	20.03	19.36	19.98	1.6**x**	294	168	15

Table 3: Comparison among various models on ML50. BLEU_H, BLEU_M and BLEU_L denote the average BLEU score over high, mid and low resource languages respectively. More notation information can be found in Table 2

#	Model	Related					Diverse				
		#TP (M)	#DP (M)	#Dec	DS	BLEU	#TP (M)	#DP (M)	#Dec	DS	BLEU
1	6-6	64	64	1	1.0**x**	16.75	66	66	1	1.0**x**	20.60
2	11-1	58	58	1	2.3**x**	14.51	61	61	1	2.6**x**	17.94
3	11-1-EACH	80	58	8	2.3**x**	14.81	83	61	8	2.6**x**	18.68
4	11-1-RAND	58	58	4	2.3**x**	14.69	74	61	5	2.6**x**	18.37
5	11-1-FAM	68	58	4	2.3**x**	15.20	74	61	5	2.6**x**	**18.82**
6	11-1-EMB	68	58	4	2.3**x**	15.20	74	61	5	2.6**x**	18.62
7	11-1-ST	65	58	3	2.3**x**	15.04	74	61	5	2.6**x**	18.64
8	10-2	59	59	1	1.9**x**	15.73	62	62	1	1.9**x**	18.91
9	10-2-EACH	104	59	8	1.9**x**	16.13	106	62	8	1.9**x**	19.60
10	10-2-RAND	78	59	4	1.9**x**	16.04	87	62	5	1.9**x**	19.23
11	10-2-FAM	78	59	4	1.9**x**	**16.88**	87	62	5	1.9**x**	**20.25**
12	10-2-EMB	78	59	4	1.9**x**	**16.88**	87	62	5	1.9**x**	19.98
13	10-2-ST	78	59	4	1.9**x**	16.71	87	62	5	1.9**x**	20.03

Table 4: Translation speed and accuracy trade-off on TED8-Related and TED8-Diverse corpora. Notation information can be found in Table 2.

show the BLEU score differences between models 10-2 and 10-2-EACH on TED8-Related in Figure 2. The left three languages are relatively low resourced and their performance is lower than the baseline model in which all languages share one decoder[1]. This also demonstrates that their decoders are not able to learn robust representations given a limited amount of training data. And decoders trained with high resource languages generate higher quality translations and we attribute this to the enough training data and no negative transfer effect when trained without other languages (Arivazhagan et al., 2019).

Random language set assignment (RAND) We find that random language set assignment slightly improve the performance over the baseline due to the sub-optimal knowledge transfer among languages in the same decoder. If each decoder handles a similar number of languages, it also slightly improve the performance compared to the model with one shared decoder (BLEU scores between rows 2 vs. 4 and 8 vs. 10 in Tables 3 and 4). We attribute this to that the shallow decoder performs better given fewer languages. This also demonstrates that one shallow decoder does not have enough capacity to model a large number of languages. However, compared to language family and embedding assignment methods, the random

---

[1] Note that although sk is defined as a low resourced language in this dataset, the reason why language sk still have slightly better result is that sk has 61.5k training data but the other three low resource languages (az, be, gl) have less than 10k training sentence pairs.

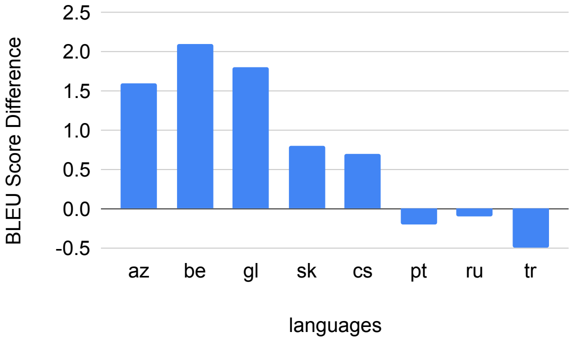


Figure 3: The BLEU score difference between models 10-2-FAM and 10-2-EACH on TED8-Related ($\text{BLEU}_{\text{10-2-FAM}} - \text{BLEU}_{\text{10-2-EACH}}$). (Left four languages are low-resourced and the right four are high-resourced.)

language set method has lower translation quality, showing that how to assign target languages into these decoders is also crucial.

One language family per decoder (FAM) We group all languages into several groups according to their language families and assign each family to one shallow decoder. As a result, we have 15, 4, 5 language families in ML50, TED8-Related and TED8-Diverse corpora respectively. From the comparison between rows 2 vs. 5 and 8 vs 11 in Tables 3 and 4. It is clear to find that language family-based decoders achieves better accuracy and maintain the low latency at the same time. Furthermore, for models with multiple 2-layer decoders, they achieve comparable performance with the model 6-6 and obtain around a 1.8 times speedup at inference time. We think the improvement is mainly coming from the better knowledge transfer among similar languages (in one language family). In order to understand this further, we plot the BLEU score difference between models 10-2-EACH and 10-2-FAM on TED8-Related in Figure 3.

We find that the major improvement of model 10-2-FAM over 10-2-EACH is from the low resource languages which means the high resource languages help their relevant low resource languages effectively.

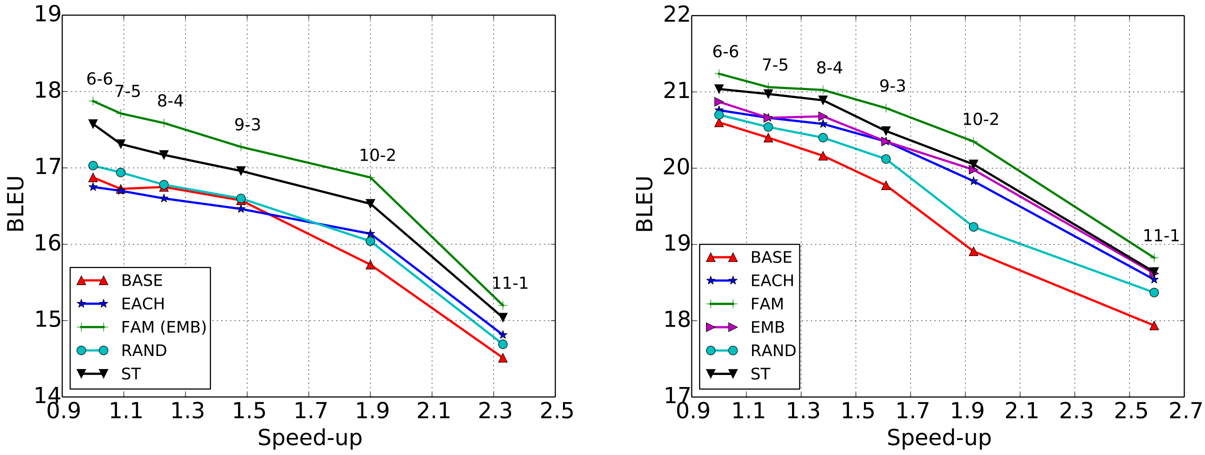
Language embedding-based assignment (EMB) For the fair comparison, languages are also grouped into the same number of language families according to language embeddings from the well-trained baseline model 6-6. Grouping results are listed in the Appendix C. We first find that language embedding-based grouping method is able to group similar languages together, showing the ability of language embeddings to effectively capture language characteristics during training. For example, on TED8-Related, the language embedding achieve the same grouping result as the language family-based one shown in Table 1. The language embedding-based assignment method achieves similar results compared to the language family-based one and effectively improve the performance of the baseline model.

Self-taught language assignment (ST) In this method, the model tries to assign target languages to multiple decoders automatically and there is no need having any prior knowledge (linguistic families) or well-trained models (language embeddings). From the rows 7 vs. 2 and 13 vs. 8 in Tables 3 and 4, our self-taught method improves around 1 BLEU score over the baseline. It also achieves similar results compared with the language family (embedding)-based language assignment methods, demonstrating the effectiveness of this method.

5 Analysis and Discussion

5.1 Multiple decoders for various layer allocations

In our main experiments, we use multiple very shallow decoders (i.e., 1 and 2-layer decoders) because there is a clear performance drop when using a single decoder with this configuration for one-to-many translation compared with the standard transformer (6-6), and compared to deeper decoders, employing multiple 1- or 2-layer decoders keeps the number of parameters manageable at training time. Nevertheless, it will be meaningful to explore the effect of multiple decoders on various layer allocations. Considering the model size and tractable training time, we only conduct experiments on TED8 corpora and the results are shown in Figure 4. On each line (the same language assignment method), the deeper decoders achieve better performance and the shallower decoder has lower latency. Moreover, if we compare language family-based assignment and the baseline models, given the same decoding speed at inference time, the former one consistently improve the performance with the same decoding speed at inference time. And with the similar performance, e.g., 10-2-FAM and 6-6, our best multiple shallow decoder models have much



(a) TED8-Related

(b) TED8-Diverse

Figure 4: Multiple decoders with various layer allocations of Transformer on ML50, TED8-Related and TED8-Diverse corpora. X-Y denotes X and Y layers in the encoder and decoder respectively. 'BASE' denotes the shared decoder model.

lower latency.

5.2 Speed-accuracy trade-off in multilingual machine translation

From the above experiments and findings, in the one-to-many translation, the DESD framework obtains superior speed-accuracy trade-off. For example, the model with 10 encoder layers and 2 decoder layers obtain slightly better accuracy and a 1.8**x** speedup.

Under the one-to-many setting, multiple shallow decoders are needed to mitigate the performance drop of the DESD model. And the crucial part is to group languages with similar features to one decoder to obtain the better knowledge transfer among languages (our FAM, EMB and ST methods). With this, our DEMSD model with multiple 2-layer decoder is capable of achieving similar performance and a 1.8**x** speedup compared to the standard transformer.

6 Related Work

Speed and accuracy are two important metrics to evaluate a machine translation system. In this work, we mainly discuss the transformer architecture (Vaswani et al., 2017). A number of works have explored various ways to improve its inference speed. Kim et al. (2019) adopt shallow decoder and layer trying to speed up the inference on CPUs. Shi and Knight (2017) and Senellart et al. (2018) employee vocabulary reduction to speed up the softmax layer. Li et al. (2020) employ a latent depth transformer model which prune layers during inference time to reduce the inference cost. There are also some works optimizing attention computations to speed up the inference speed (Zhang et al., 2018; Kitaev et al., 2020; Katharopoulos et al., 2020; Chelba et al., 2020). Recently, Kasai et al. (2020) places more capacity to the encoder side and keep an extremely shallow (one-layer) decoder to achieve a superior speed-accuracy trade-off.

Multilingual neural machine translation (MNMT) is an attractive field recently (Firat et al., 2016; Ha et al., 2016; Johnson et al., 2017) because MNMT tries to employ one model to translate more than one language pair, even including ones unseen during training (zero-shot translation). Knowledge transfer among languages boosts the performance of low-resource languages. However, many works (Arivazhagan et al., 2019; Zhang et al., 2020; Aharoni et al., 2019) have shown the capacity bottleneck of translation when modeling many languages. Therefore, before simply stacking more layers in the encoder and decoder, it is crucial to first understand how to balance the speed and accuracy given a fixed capacity budget. Therefore, in this work, we try to understand various capacity allocations to achieve the best speed-accuracy trade-off.

7 Conclusion

In this work, we study speed-accuracy trade-offs using various layer configurations for multilingual neural machine translation. We find that for many-to-one translation, deep encoder and shallow decoder (DESD) models improve decoding speed while maintaining translation quality with the same model capacity. However, for one-to-many translation we do observe a drop in quality when the decoder depth is reduced. To mitigate the performance drop of DESD models in one-to-many translation, we proposed using a shared encoder and *multiple* shallow decoders (DEMSD). Our best DEMSD models with 2-layer decoders are capable of speeding up decoding by 1.8 times while achieving the same quality compared to a standard transformer.

Our work can be combined with techniques mentioned in Section 6 such as optimized attention computation, vocabulary reduction, knowledge distillation, etc. We expect that these combinations will further improve the decoding speed and obtain a better speed-accuracy trade-off. This work can also be extended to other encoder-decoder applications beyond translation, such as question answering, dialogues, and so on. We will explore these directions in the future work.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ciprian Chelba, Mia Chen, Ankur Bapna, and Noam Shazeer. 2020. Faster transformer decoding: N-gram masked self-attention. *arXiv preprint arXiv:2001.04589*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Bernard Comrie. 1989. *Language universals and linguistic typology: Syntax and morphology*. University of Chicago press.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252.

Thanh-Le Ha, Jan Niehues, and Alexander Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. *arXiv preprint arXiv:1611.04798*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Melvin Johnson, Mike Schuster, Quoc Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A Smith. 2020. Deep encoder, shallow decoder: Reevaluating the speed-quality tradeoff in machine translation. *arXiv preprint arXiv:2006.10369*.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Young Jin Kim, Marcin Junczys-Dowmunt, Hany Hassan, Alham Fikri Aji, Kenneth Heafield, Roman Grundkiewicz, and Nikolay Bogoychev. 2019. From research to production and back: Ludicrously fast neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 280–288, Hong Kong. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

M Paul Lewis. 2009. *Ethnologue: Languages of the world*. SIL international.

Xian Li, Asa Cooper Stickland, Yuqing Tang, and Xiang Kong. 2020. Deep transformers with latent depth. *arXiv preprint arXiv:2009.13102*.

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

Jean Senellart, Dakun Zhang, Bo Wang, Guillaume Klein, Jean-Pierre Ramatchandirin, Josep Crego, and Alexander Rush. 2018. OpenNMT system description for WNMT 2018: 800 words/sec on a single-core CPU. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 122–128, Melbourne, Australia. Association for Computational Linguistics.

Xing Shi and Kevin Knight. 2017. Speeding up neural machine translation decoding by shrinking run-time vocabulary. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 574–579, Vancouver, Canada. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. 2019. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020. Balancing training for multilingual neural machine translation. *arXiv preprint arXiv:2004.06748*.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019. Improving deep transformer with depth-scaled initialization and merged attention. *arXiv preprint arXiv:1908.11365*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. *arXiv preprint arXiv:2004.11867*.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018. Accelerating neural transformer via an average attention network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Melbourne, Australia. Association for Computational Linguistics.

Language Family	Languages
INDO-IRANIAN	hi, hr
SLAVIC	mk, bs, bg
KOREAN	ko
HELLENIC	el
ROMANCE	fr

Table 5: Language families in the TED8-Related corpus.

A Training details of DESD model

In order to explore how DESD models work on multilingual machine translation, we train transformer-based models with various layer allocations on three multilingual machine translation corpora, ML50, TED8-Related and TED8-Diverse. For the fair comparison, the training process is the same across all models.

On ML50, we employ the standard transformer-base model: 8 attention heads per layer, 512 model dimensions, 2048 hidden dimensions and 0.1 dropout. All models are trained for 100,000 with batches of 64k tokens using Adam and 0.1 label smoothing. The learning rate goes to $1e{-3}$ within 4,000 steps,and then decays with the inverse square-root schedule.

On TED8 corpora, following (Wang et al., 2020), a smaller transformer model is adopted, i.e., 4 attention heads per layer, 512 model dimensions, 1024 hidden dimensions and 0.3 dropout. All models are trained for 40,000 with batches of 16k tokens using Adam and 0.1 label smoothing. The learning rate goes to $2e{-4}$ within 4,000 steps, and then decays with the inverse square-root schedule.

B Language family assignment results

In Table 5, we show the language family-based assignment result on TED8-Diverse. Since this corpus is collected without considering relatedness, some groups just have one language. But its multiple decoders model improves the accuracy, showing the effectiveness of this method.

The language families in ML50 is shown in Table 6.

Language Family	Languages
DRAVIDIAN	ta, ml, te, vi
GERMANIC	de, nl, sv, af
INDO-ARYAN	hi, si, ne, ar, ur, mr, gu, bn
IRANIAN	ps, fa
CHINESE	zh
BALTIC	lv
AUSTROASIATIC	km, id, xh, he
JAPONIC	ja
KOREANIC	ko
KRA-DAI	th, my
POLLSH	pl
ROMANCE	fr, es, lt, ro, it, pt, gl
SLAVIC	ka, mn, cs, ru, hr, uk, mk, sl
TURKIC	tr, kk, az
URALIC	fi, et

Table 6: Language families in the TED8-Related corpus.

C Language embedding assignment results

On TED8-Related, we obtain the same language assignment results as the language family-based one. On TED8-Diverse, the result of language embedding assignment is pretty similar to the language family assignment result (Table 7. The only difference is that language bg is grouped with language mk. We think this is because the language embedding not only contains the linguistic feature but the data feature as well.

Group Id	Languages
0	hi, ko
1	mk, bg
2	ko, bs
3	el
4	fr

Table 7: Language embedding-based language assignment result on the ML50 corpus.

Group Id	Languages
0	ta, ml, te
1	de, nl, af
2	hi, si, ne, ur, mr, gu, bn
3	ps, fa
4	zh
5	lv
6	km, id, xh, he
7	ja
8	ko
9	th, my, vi
10	pl, sv, ar
11	fr, es, lt, ro, it, pt, gl
12	ka, mn, cs, ru, hr, uk, mk, sl
13	tr, kk, az
14	fi, et

Table 8: Language embedding-based grouping results on ML50.

With Measured Words: Simple Sentence Selection for Black-Box Optimization of Sentence Compression Algorithms

Yotam Shichel
Department of Software and
Information Systems Engineering
Ben Gurion University
yotamsh@post.bgu.ac.il

Meir Kalech
Department of Software and
Information Systems Engineering
Ben Gurion University
kalech@bgu.ac.il

Oren Tsur
Department of Software and
Information Systems Engineering
Ben Gurion University
orentsur@bgu.ac.il

Abstract

Sentence Compression is the task of generating a shorter, yet grammatical version of a given sentence, preserving the essence of the original sentence. This paper proposes a Black-Box Optimizer for Compression (`B-BOC`): given a black-box compression algorithm and assuming not all sentences need be compressed – find the best candidates for compression in order to maximize both compression rate and quality. Given a required compression ratio, we consider two scenarios: (i) single-sentence compression, and (ii) sentences-sequence compression. In the first scenario, our optimizer is trained to predict how well each sentence could be compressed while meeting the specified ratio requirement. In the latter, the desired compression ratio is applied to a sequence of sentences (e.g., a paragraph) as a whole, rather than on each individual sentence. To achieve that, we use `B-BOC` to assign an optimal compression ratio to each sentence, then cast it as a Knapsack problem, which we solve using bounded dynamic programming. We evaluate `B-BOC` on both scenarios on three datasets, demonstrating that our optimizer improves both accuracy and Rouge-F1-score compared to direct application of other compression algorithms.

1 Introduction

Sentence Compression is the task of generating a short, accurate, and fluent sentence that preserves the essence of a given original sentence by removing nonessential words and/or rephrasing it in a compact form. Compression can take many forms, ranging from Extractive and Abstractive Summarization (Jing, 2000; Madnani et al., 2007; Cohn and Lapata, 2008, 2009; Galanis and Androutsopoulos, 2010; Rush et al., 2015; Chopra et al., 2016) to Text Simplification and Paraphrasing (Bannard and Callison-Burch, 2005; Xu et al., 2012; Klerke et al., 2016; Narayan et al., 2017; Aharoni and Goldberg, 2018; Botha et al., 2018), among others.

On the sentential level, compression is often viewed as a *word deletion* task (Knight and Marcu, 2000, 2002; Filippova and Strube, 2008; Filippova et al., 2015; Wang et al., 2016, 2017; Zhou and Rush, 2019). However, not all sentences could, or should be compressed as part of compressing a longer text they reside in. Consider the familiar scenario in which a full paragraph needs to be compressed in order to have an EACL paper meet the page restriction specified in the submission guidelines. A common approach by LaTeX users is to first identify paragraphs ending with a short line, (e.g., this very paragraph), then choose one or more sentences that could be compressed with a minimal loss of information – shaving the extra line. We propose a Black-Box Optimizer for Compression (`B-BOC`) that mitigates this problem. Given a compression algorithm A, a desired compression ratio, and a document D, `B-BOC` chooses the best sentences to compress using A in order to produce a shorter version of D, while keeping the other sentences of D untouched. `B-BOC` achieves that without explicit knowledge of the inner-workings of the given compression algorithm, hence we call it a *black-box* optimizer. Selected sentences are expected to be the best candidates for compression – balancing compression rate with compression quality.

This paper addresses two main research questions: (1) How to predict the compression performance (preserving meaning and grammar) of an algorithm on a given sentence? (2) Given a document

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1625–1634
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

and a required compression ratio, how to choose the optimal *subset* of sentences to compress, along with the appropriate compression ratio per each of the sentences, so that the *total* compression meets the required compression requirement?

Given a gold set of pairs of sentences and their compressions, we represent each sentence as a vector of shallow and syntactic features, and train a regression model to predict its expected compression rate. `B-BOC` ranks all sentences by the predicted compression potential while considering a required compression ratio.

The document-level task could be modeled as a Knapsack optimization problem, considering the subset of sentences to be compressed in order to satisfy the overall compression requirement (capacity), with a minimal loss of information (value). The solution space covers the trade-off between aggressively compressing only a few sentences and applying minimal compression on a larger number of sentences. While the general Knapsack is NP-complete, the 0-1 variation can be approximated efficiently by using Dynamic Programming (Hristakeva and Shrestha, 2005).

We evaluate `B-BOC` on three benchmarks commonly used for the sentence compression task. We show that applying `B-BOC` on top of state-of-the-art sentence compression models improves the performance for any desired compression rate. In addition, optimizing the `B-BOC`-Knapsack achieves top performance on the document-level task.

2 Related Work

Early sentence compression works employ the noisy channel model, learning the words and clauses to be pruned (Knight and Marcu, 2000, 2002; Filippova and Strube, 2008; Clarke and Lapata, 2008; Cohn and Lapata, 2009).

The top-performing sentence compression models use a Policy Network coupled with a Syntactic Language Model (bi-LSTM) evaluator (Zhou and Rush, 2019), and a stacked LSTM with dropout layers (Filippova et al., 2015). An extension of Filippova et al., adding syntactic features and using Integer Linear Programming (ILP), yields improved results in a cross-domain setting (Wang et al., 2017).

Sentence selection is used for document extractive summarization – a task conceptually close to ours, in which full sentences are extracted from a long document, see (Nenkova et al., 2011) for an overview. State-of-the-art selection is achieved by combining sentence and document encoders (CNN and LSTM) with a sentence extraction model (LSTM) and a reinforcement layer (Narayan et al., 2018).

Sentence rephrasing is an abstractive approach to rewrite a sentence into a shorter form using some words that may not appear in the original sentence. A data-driven approach to abstractive sentence summarization is suggested in (Rush et al., 2015; Chopra et al., 2016), using about four million title-article pairs from the Gigaword corpus for training, and uses a convolutional neural network model to encode the source and produce a single representation for the entire input sentence. Tree-to-tree grammar extraction method for the rewriting task is used in (Cohn and Lapata, 2008, 2009). State-of-the-art performance on the abstractive summarization task is obtained using Hierarchical Attentional Seq2Seq Recurrent Neural Network (Nallapati et al., 2016; See et al., 2017).

3 Task Definitions and Methodology

In this section we formally define the sentence-level and the document-level tasks (§3.1) and provide a detailed description of the application of `B-BOC` in both settings (§3.2).

3.1 Problem Definitions

Sentence-Level Compression Given a set of sentences $S = \{s_i\}_{i=1}^{n}$; a desired compression rate γ; the number of sentences to compress $k \leq n$; a compression algorithm A; and an oracle $\mathbb{R} : (A, S) \rightarrow [0, 1]$, returning a score reflecting the compression quality (grammaticality and minimal loss of information) A would achieve on $s \in S$ – we would like to choose a set $S^{k,\gamma} \subseteq S$ of k sentences:

$$S^{k,\gamma} = \{s_j | \frac{|A(s_j)|}{|s_j|} \leq \gamma \wedge \underset{s_j}{\operatorname{argmax}}\, \mathbb{R}(A, s_j)\}_{j=1}^{k}.$$

We call this *sentence-level* compression since each sentence should meet the γ constraint independently. It is important to note that $\gamma' \leq \gamma \not\Rightarrow S^{k,\gamma} \subseteq S^{k,\gamma'}$, since different sentences may be better compressed to different γ values. Consider the following two sentences used to illustrate the importance of the Oxford comma: $S =${*"I had a yummy dinner with my parents, Batman and Catwoman", "I had a yummy dinner with my parents, Batman, and Catwoman"*}[1], and $k = 1$. The first

---

[1] The first sentence, without the Oxford comma, implies that Batman and Catwoman are the speaker's parents, the

sentence could be compressed to "I had a yummy dinner with my parents" with a minimal loss of information, while it does not make sense to compress the second sentence this way and it should be compressed to "I had a yummy dinner", thus specifying $k = 1$, the sentence to be compressed with minimal loss of meaning depends on the desired γ value.

Document-Level Compression In this setting, we are given a sequence of sentences $D = \{s_i\}_{i=1}^{n}$ (a paragraph or a full document), and a desired compression rate γ that should be applied to D as a *whole*. That is, we wish to find an optimal subset of S^γ that satisfies:

$$\underset{S^\gamma \subseteq D}{\operatorname{argmax}} \sum_{s_i \in S^\gamma} \mathbb{R}(A, s_i) \tag{1}$$
$$\text{s.t.} \quad \frac{\sum_{s_i \in S^\gamma} |A(s_i)| + \sum_{s_i \in D \setminus S^\gamma} |s_i|}{|D|} \leq \gamma$$

Since γ refers to D rather than to individual sentences, the overall quality can be maximized by choosing a varying number of sentences expected to achieve different optimal compressions. Unlike the sentence-level setting, here, an optimal S^γ may contain a combination of sentences, for some of which $\frac{|A(s)|}{|s|} \leq \gamma$, and for others $\frac{|A(s)|}{|s|} > \gamma$.

3.2 Computational Approach

Scoring Function Given a corpus $C = \{\langle s_i, \hat{s}_i \rangle\}_i^m$ of sentence pairs, each pair contains an original sentence s and its gold compression $\hat{s}$, we define the golden ratio $\hat{\gamma}_i = \frac{|\hat{s}_i|}{|s_i|}$, and posit $\mathbb{R}(A, s) \approx 1 - |\hat{\gamma}_i - \frac{|A(s_i)|}{|s_i|}|$.

We justify the use of $\hat{\gamma}_i$ as a proxy to the optimal compression quality, as compression ratios are found to correlate with compression quality measured against gold compressions (Napoles et al., 2011). The use of $\hat{\gamma}_i$ as a proxy is validated through manual evaluation, see Sections 4.2 and 5.1.

Syntactic features were successfully used for sentence compression (Clarke and Lapata, 2008; Wang et al., 2017; Liu et al., 2017; Futrell and Levy, 2017). Assuming that sentence complexity correlates with the ease of compression, we follow (Brunato et al., 2018) and represent each sentence as a vector of shallow features (sentence length, average word length, punctuation counts, etc.) and syntactic features (depth of constituent parse tree as well as the number of internal nodes, word's depth in a dependency parse tree, mean dependency distance, etc.).

We now train a regression model and learn the scoring function $R(A, s)$ by minimizing the loss:

$$L(C, A) = \Sigma_{s_i \in C} \left[R(A, s_i) - \mathbb{R}(A, s_i)\right]^2$$

We note that we do not train a compression algorithm, but an oracle – a scoring function that predicts the *quality* of the compression algorithm A will achieve on a given sentence. This oracle will be used to rank candidate sentences in order to optimize the choice of sentences in the two tasks defined in Section 3.1.

We train a Gradient Boosted Tree regression model using XGBoost. The model's hyperparameters (e.g., subsample ratio, learning rate, max depth) were tuned on a separated development set.

Sentence level compression: Given a set of sentences S, B-BOC operates on two steps: (i) It applies R on every $s \in S$, producing an ordered set $\hat{S}$ for which $\forall_{i<j} R(A, s_i) \geq R(A, s_j)$. (ii) It constructs $S^{k,\gamma}$ by iterating over $\hat{S}$, choosing the first k sentences that satisfy the γ requirement.

Document level compression: Using the task definition in Section 3.1, it is straight forward to cast the task as a combinatorial 0-1 Knapsack problem in the following way: Given a set of items (sentences) $S = \{s_1, ..., s_n\}$, each weighs $w_i = |A(s_i)|$ if compressed, or $|s_i|$ if kept in the original form, and each holds a value $v_i = R(A, s_i)$ (predicted compression quality), if compressed and $v_i = 1$ if kept in the original form; and given a weight limit $W = \gamma \cdot \sum_1^n |s_i|$ – we wish to find $S^\gamma = \{s_i | x_i = 1\}$ that maximizes:

$$\sum_i (v_i x_i - [1 - x_i]^{-1}) \qquad \text{s.t.}$$
$$\sum_i (w_i x_i - |s_i| \, [1 - x_i]^{-1}) \leq W \quad , x_i \in \{0, 1\}$$

were $x_i = 1$ denotes we choose to compress s_i and $x_i = 0$ denotes that s_i remains in its original long form (hence the 0-1 Knapsack setting). Note that the value we maximize and the weight constraints include a term for the unchanged sentences, in case they are not chosen for compression. This term is introduced since the γ constraint in the task definition applies to the document as a whole.

---

second sentence implies that the speaker had dinner with four people – her parents *and* Batman *and* Catwoman.

B-BOC-knapsack returns S^{γ} by solving the 0-1 knapsack problem using the dynamic programming approach proposed by (Hristakeva and Shrestha, 2005) to reduce the computation complexity to a pseudo-polynomial time. Knapsack's solution ensures an optimal set of sentences, satisfying the required compression limitations, while achieving the maximum quality score.

4 Experimental Setting

4.1 Datasets

Training Data: We train B-BOC on a dataset of 200,000 sentence-compression pairs[2] used by Filippova and Altun (2013). Each pair is composed of a long sentence (usually the teaser, caption, extract or the first sentence that bears the most salient information) from a news story and the story's headline, which is a compressed version of the long sentence.

Out of these 200,000 sentences, we set aside 9,000 to be used as a development set, and 1,000 as one of our three test sets.

Evaluation datasets: Three datasets are used for evaluation:

1. Google (GGL) – the first 1000 sentences of the training corpus (described above) were used for testing.
2. British National Corpus (BNC) – a manually crafted dataset of $\sim$ 1500 sentence-compression pairs. Given a long sentence, annotators were asked to produce a short version by deleting extraneous words from the source without changing the order of words [3].
3. Gigaword (GIGA)- headline-generation corpus of articles[4] consists $\sim$ 4 million sentence-compression pairs. We note that this dataset contains abstractive pairs, nevertheless, it can be used to measure accuracy.

4.2 Evaluation Procedures

Evaluation metrics: We used four evaluation metrics that complement each other, providing a comprehensive evaluation of the different factors that contribute to quality summarization as suggested by (Filippova et al., 2015): (1) Accuracy – how many compressed sentences are fully reproduced, (i.e., the generated compression is identical to the golden one). (2) F-score – given the golden and predicted compressions, recall and precision are based on the ROUGE metric. (3) Readability score – the grammaticality of the compression. (4) Informativeness – the level in which the compression covers the most salient information.

The two latter metrics are based on a manual evaluation by three annotators, scoring Readability and Informativeness on a 5-Point Likert scale. The annotators were guided to give a top Readability score (score 5) if the predicted compressed sentence is clear and grammatically correct, regardless of the original context, and a top Informativeness score (score 5) if the essence of the original is preserved completely. The Informativeness measurement bears some degree of subjectivity as annotators may not agree on what should be considered "the essence" of a sentence, see examples in Table 1. We used Cohen's Kappa (Cohen, 1960) to measure inter-annotator agreements. Low agreements are expected due to the subjectivity and the five-point scale, i.e., when two raters agree on the grammaticality of a sentence, but do not give the same exact Informativeness score. To account for slight variations in assessment, we measure agreement using the off-by-one procedure proposed by (Tsur and Rappoport, 2009) and supported by (Toutanova et al., 2016). Linear and Quadratic weighting were added as additional statistical methods. Nevertheless, we kept the 5-point scale to be aligned with Filippova's evaluations. The Kappa values for the strict and the off-by-one agreement for a sample of 200 sentences of the GGL dataset are reported in Table 2. These scores are comparable with the scores reported by (Filippova et al., 2015). Agreement of 0.86 and 0.78 for Readability and Informativeness reflect an almost perfect agreement on Readability and substantial agreement on Informativeness, according to the interpretation protocol suggested by McHugh (2012).

4.2.1 Black-Box Compression Models

As described in Section 3.2, B-BOC accommodates any compression model used to compress the sentences. To show this independence, B-BOC is evaluated with three competitive compression models: (1) `Filippova`: An LSTM model trained on two million sentence-compression pairs (Filippova et al., 2015), (2) `Zhou`: An unsupervised model for sentence summarization (Zhou and Rush, 2019), and (3) `Klerke`: A three-layer bi-LSTM model (Klerke et al., 2016).

---

[2] `www.github.com/google-research-datasets/sentence-compression`

[3] `jamesclarke.net/research/resources`

[4] `github.com/harvardnlp/sent-summary`

Source	Text	Issue
Long	A gang of youths between eight and sixteen robbed a man in an Oldbrook underpass for just 10£	The salience the clause "for just 10£"
Manual 1	*A gang of youths robbed a man in an Oldbrook underpass for just 10£*	
Manual 2	*A gang robbed a man in an Oldbrook underpass*	
Long	A woman was injured by a falling tree in the Gresham neighborhood, according to the Chicago Fire Department	The salience of the location "Gresham neighborhood"
Manual 1	*A woman was injured by a falling tree*	
Manual 2	*A woman was injured by a falling tree in the Gresham neighborhood*	

Table 1: Two examples of compression disagreements.

Agreement coverage	Readability	Informativeness
Strict	0.61	0.32
Off-by-one	0.86	0.78
Linear	0.78	0.54
Quadratic	0.87	0.72

Table 2: Cohen's Kappa inter-annotator agreement between three annotators for the strict, off-by-one, and other statistical approaches to calculate agreement.

4.3 Experimental Procedure

Given the two settings presented in Section 3.1, we aim to evaluate the performance of `B-BOC` in optimizing compression quality, on top of a number of black-box compression algorithms. We evaluate the way different values of k affect the performance, and explore the contribution of various feature types to the trained optimizer.

4.3.1 Sentence level compression:

We evaluate the effectiveness of `B-BOC` for varied compression rates. The tested sentences were divided into buckets of different compression rates 0.1-0.9. For each bucket we set k to be 50% of the sentence in a bucket and compare `B-BOC` selections to: (1) A random selection of k sentences from the bucket (RANDOM). (2) The average of all sentences in the compression rate bucket (ALL). We report results of this comparison for each of the black-box algorithms listed in Section 4.2.1). Note that the F-score is based on the actual results of each of the black-box models, and that both `B-BOC` and RANDOM choose from the same pool of candidates for each compression rate bucket.

4.3.2 Document level compression

Having a document or a paragraph comprised of several sentences that are needed to be compressed, the target is to find the sentences that would gain the highest performance score subject to the overall compression ratio constraint.

To simulate a document, we synthesized one hundred documents by randomly sampling sentences from the test set. Every document contains 100 different sentences of varying lengths. We then use `B-BOC`-Knapsack as described in 3.2. `B-BOC`-Knapsack is compared with: (1) an oracle Knapsack solution where the golden scores are provided, rather than estimated by `B-BOC` (ORACLE). (2) We iteratively sample sentences to compress until the compression ratio is reached (RANDOM). (3) A sorted selection- choosing sentences by their lengths in an ascending sort (SHORTER FIRST). The latter baseline was added followed by our experiments, showing that compression quality tends to be higher for shorter sentences.

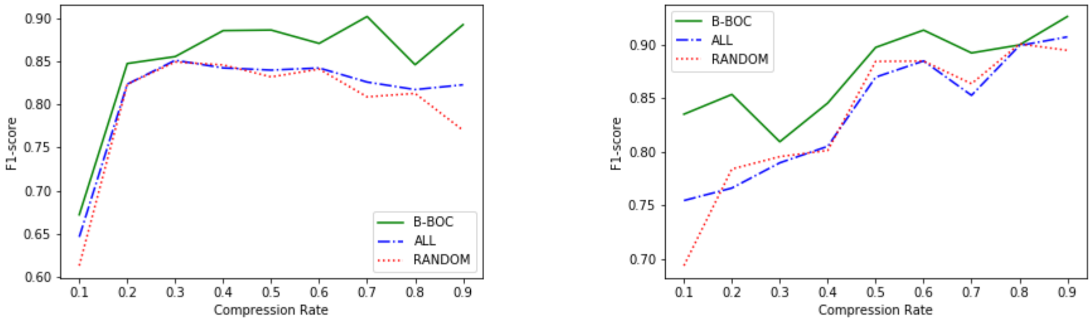
5 Results and Discussion

5.1 Results

Detailed results for both sentence level and document level compression are presented below.

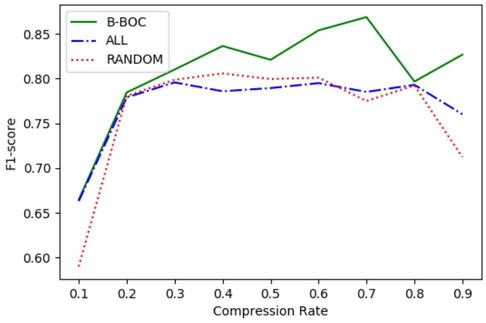
Sentence level compression: Figure 1 presents the F1 performance of sentence selection methods over varied γ-buckets on the GGL dataset, while training with the compression models of Filippova and Zhou respectively. `B-BOC` is compared with all sentences and a random selection of sentences, as described in Section 4.3.1. It can be seen that `B-BOC` achieves the highest F1-score for every γ. The evaluated metrics' averages for all compression buckets are presented in Table 3, evaluating the GGL dataset using three different compression models. Best results are in bold. `B-BOC` achieves the best performance overall measures – automatic and manual (F1-score, Accuracy, Readability and Informativeness).

The results confirm that by utilizing `B-BOC`, the top sentences which yield the best overall compression results will be chosen, no matter which black-box compression model is applied, for every given compression ratio. Table 4 describes the average F1-scores and variances for the manual evaluations



(a) Filippova's compression model

(b) Zhou's compression model



(c) Klerke's compression model

Figure 1: Average F1-score (y-axis) applied on the GGL dataset for different compression rate buckets (x-axis) while training `B−BOC` with Filippova (a), Zhou's (b) and Klerke (c) compression models.

		F1-score	F2-score	Accuracy	Readability	Info.
Filippova	ALL	0.837	0.77	0.31	4.562	3.78
	RANDOM	0.835	0.76	0.306	4.559	3.79
	`B−BOC`	**0.86 ***	**0.795**	**0.332**	**4.65**	**4.08**
Zhou	ALL	0.82	0.7	0.24	**3.92**	3.41
	RANDOM	0.815	0.69	0.226	3.91	3.41
	`B−BOC`	**0.87 ***	**0.77**	**0.30**	**3.92**	**3.60**
Klerke	ALL	0.787	0.685	0.187	4.12	3.73
	RANDOM	0.783	0.677	0.156	4.00	3.65
	`B−BOC`	**0.815 ***	**0.72**	**0.214**	**4.23**	**3.97**

Table 3: GGL dataset: Evaluation metrics' average results over all compression rate buckets. Statistical significance using a paired T-test is indicated by *.

Likert score	Info. F1	Read. F1	Info. Var	Read. Var
1	0.59	0.67	0.13	0.10
2	0.66	0.66	0.07	0.04
3	0.70	0.69	0.05	0.05
4	0.77	0.74	0.04	0.05
5	0.77	0.75	0.03	0.04

Table 4: Readability and Informativeness average F1-scores and variance.

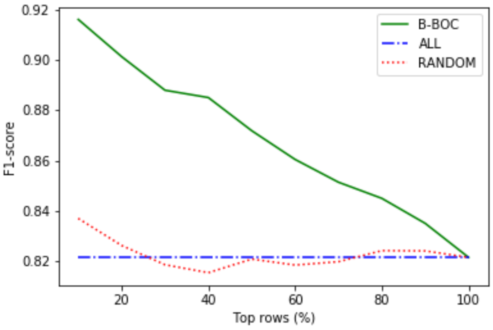


Figure 2: F1-score (y-axis) per top X% of sentences (x-axis) that are ranked by `B−BOC`.

of the GGL dataset using Filippova's compression model. It can be seen that both Readability and Informativeness are correlated with F1-scores. A compression that gets a higher Readability or Informativeness on the 5-point Likert score, will most probably get a higher F1-score as well, with a lower variance. This manual evaluation of Readability and Informativeness supports our choice of $\mathbb{R}$ (see Section 3.2).

As described in Section 4.3.1, the top 50% of the test dataset are chosen for our evaluations. We repeated the same experiment varying the number (percentage) of sentences ranked by `B-BOC`. Figure 2 presents the impact of the number of sentences we consider, and demonstrates the deterministic trend of `B-BOC` ranking method. It can be seen that when considering only the higher ranked sentences, their compression will produce a higher F1-score. It suggests that the lower number of sentences we consider – the higher the benefit of `B-BOC` is, compared with a random selection of the same number of selected sentences.

Document level compression: Given a document or a paragraph and a specified compression rate requirement, `B-BOC`-Knapsack aims to find a subset of sentences, that together will satisfy the compression rate constraints if being compressed, and while guaranteeing a top F1-score. Our results below depict an experiment for compressing a document with a certain compression ratio constraint. A document is constructed using 100 sentences with variate lengths, randomly selected from a given dataset. `B-BOC`-Knapsack sentence selection is being compared with a random selection and a sorted selection of the sentences, as described in Section 4.3.2. Each experiment was repeated 100 times, sampling different sentences for each of the datasets. The average scores reported below were achieved with the same compression model used by Filippova (see Section 4.2.1) for all sentences.

Figure 3 presents the experiments for GGL and BNC datasets respectively. An overall compression requirement is added, ranging from 0.1 to 0.5 (e.g., 0.1 means that the document should be compressed in 10 percent). `B-BOC`-Knapsack has a higher F1-score for almost every compression ratio, especially at the lower ratios.

Knapsack's oracle solution can be created when considering the actual F1 and compression rates for all sentences. A histogram of the sentences that the oracle Knapsack chose to compress, grouped by their lengths is presented in Figure 4. The Figure provides a number of insights: (1) The F1-score decreases as the number of compressed sentences grow, due to the increased uncertainty when compressing more sentences. A similar pattern is observed in Figure 3. (2) The Knapsack prefers to choose shorter sentences, as these perform better than longer sentences. We attribute this to the fact that shorter sentences may be easier to optimize, as compression alternatives are limited, compared to longer sentences.

		F1-score	Readability	Informativeness
GGL	ORACLE	0.89	4.64	4.05
	RANDOM	0.837	4.59	3.76
	SHORTER FIRST	0.850	4.6	3.97
	`B-BOC`	**0.854**	**4.62**	**4.01**
BNC	ORACLE	0.68	3.77	3.26
	RANDOM	0.55	3.53	3.13
	SHORTER FIRST	0.62	3.69	3.24
	`B-BOC`	**0.63**	**3.79**	**3.28**
GIGA	ORACLE	0.47	-	-
	RANDOM	0.256	-	-
	SHORTER FIRST	0.297	-	-
	`B-BOC`	**0.303**	-	-

Table 5: Average results- Document level compression.

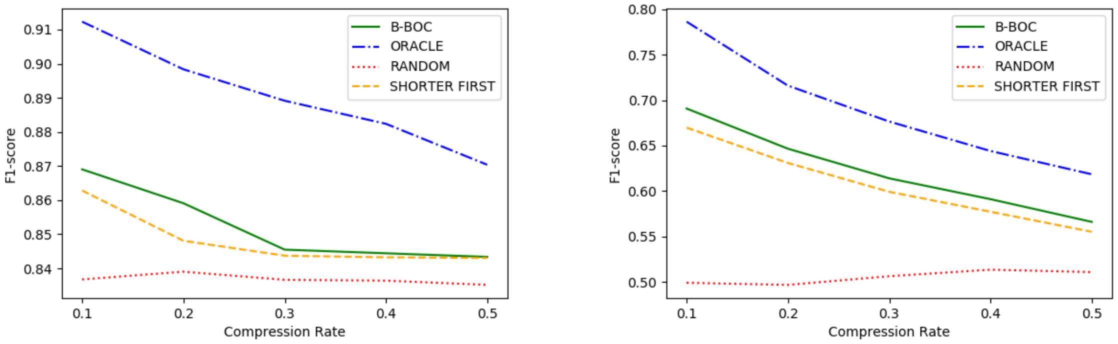
The average results for the three datasets are presented in Table 5. Best results are in bold. Informativeness and Readability average scores are aligned with the F1-scores (note that the GIGA dataset was not manually annotated for Readability and Informativeness, since we are focused on extractive summarization rather than abstractive, and the Readability and Informativeness of the two types cannot be compared directly). We observe that `B-BOC` chooses the best sentences and provides a better compression performance for any compression ratio.

Feature Importance: Sentence complexity is correlated with the parse tree structure (Oya, 2011). Analyzing the contribution of each feature type, we find the tree depth features and especially Mean Dependency Distance (MDD) to do the heavy lifting. The MDD is the sum of the depth of words in the dependency tree, divided by the total number of dependencies. For example, the MDD scores for two sentences of the same character length "Sarah read the book quickly and understood it correctly" (Figure 5 top) and "US President Donald Trump tests positive for coronavirus" (Figure 5 bottom) is $19/8 = 2.735$ and $11/7 = 1.57$, respectively. This observation validates the relation between sentence complexity and compression.

Table 6 presents the importance of the syntactic features to the `B-BOC` model in terms of weight, which means the relative number of times a feature occurs in the boosted trees of the trained model. Shallow properties such as the number of verbs and number of nodes are located at the bottom.

5.2 Discussion

Limitation of the F1-score Our main target is maximizing the F1-score, which happens to be a common approach for the sentence compression task, e.g., (Filippova et al., 2015; Zhao et al., 2018).



(a) GGL dataset

(b) BNC dataset

Figure 3: 0-1 Knapsack's F-score results for GGL dataset (a) and BNC dataset (b). x-axis is the total desired compression rate of the document (i.e., 0.1 means compressing the whole document by 10 percent). y-axis is the average F1-score of the subset of sentences being compressed.

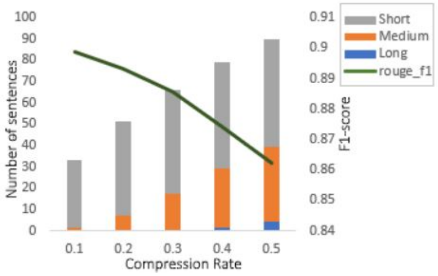


Figure 4: GGL dataset: Oracle Knapsack subset's histogram. Two y-axis are average F1-score of the subset (right axis, describes the line) and the number of selected sentences to be compressed (left axis, describes the bars' height). x-axis is the total desired compression rate of the document.

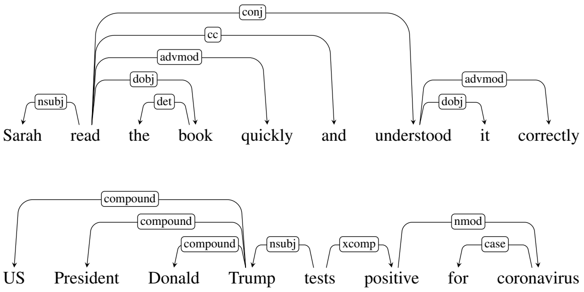


Figure 5: Dependency trees of sentences of the same length (chars), but different depth and MDD score.

Feature name	Weight
Word Length Average	0.140630
MDD-score	0.121428
Tree Depth Average	0.116776
Character count	0.093331
Count Relations	0.079562
Count PoS	0.072715
Parse-Tree Height	0.069478
Count Nodes	0.066835
Parse-Tree sub trees	0.061700
Count words	0.051541
Dependency tree depth	0.045512
Verb count	0.042461
Parse-Tree count POS types	0.038032

Table 6: Feature Importance of B-BOC. The percentages representing the relative number of times a particular feature occurs in the trees of the model.

Automatic evaluation metrics like the F1-score serve complementary purposes for linguistic quality evaluation rather than replacement because it is unclear whether the improvement in F1-score necessarily indicates the improvement of linguistic quality. Nevertheless, it was shown that the F1-score correlates with human judgment (Napoles et al., 2011). We manually performed additional evaluation for Readability and Informativeness to complement the evaluation based on the F1-score. For example, when applied on the document-level on the BNC dataset, B-BOC does not achieve the best F-score but does achieve best Readability and Informativeness scores (see Table 5).

Fairness Compression algorithms should be compared for similar levels of compression (Napoles et al., 2011). Partitioning S to different compression rate buckets, as explained in 4.3.1 and demonstrated in Figure 1, ensures a fair comparison between the different compression models.

Manual evaluations. Exploring the cases in which annotators did not agree on either Readability or Informativeness, we noticed a higher likelihood for disagreement in the lower scale of both measurements, especially in case the original sentence was convoluted or grammatically flawed.

6 Conclusions

In this paper we presented `B-BOC`- Black-Box Optimizer for Compression, a new complexity optimization method designated to the sentence compression problem. We defined the correlation between the complexity of a sentence and the chance that a black-box compression model could successfully compress it. Our optimization model is independent of the compression model used to compress the sentences and can be combined with any sentence compression model. Our evaluation on three benchmarks revealed promising results when applied to three different types of sentence compression models. We achieve top performance for a document compression problem using the `B-BOC`-Knapsack optimization implemented with a bounded Dynamic Programming technique. Our method could assist in compressing any kind of text while applying their desired compression model. Utilizing our method provides a proper guideline for which of the sentences are the most beneficial to focus on, in order to compress a given text, while yielding the best overall compression results.

For future work, we plan to construct Model-Dependant Optimization that accounts for the features of each compression model. This will facilitate a choice of the compression model that is the most suitable for a given sentence.

Acknowledgments

We would like to thank Katja Filippova and Alexander Rush for providing us with the outputs of their compression models.

References

Roee Aharoni and Yoav Goldberg. 2018. Split and rephrase: Better evaluation and stronger baselines. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 719–724.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics.

Jan A Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from wikipedia edit history. *arXiv preprint arXiv:1808.09468*.

Dominique Brunato, Lorenzo De Mattei, Felice Dell'Orletta, Benedetta Iavarone, and Giulia Venturi. 2018. Is this sentence difficult? do you agree? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2690–2699.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 137–144. Association for Computational Linguistics.

Trevor Anthony Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1491, Seattle, Washington, USA. Association for Computational Linguistics.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 25–32. Association for Computational Linguistics.

Richard Futrell and Roger Levy. 2017. Noisy-context surprisal as a human sentence processing cost model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 688–698, Valencia, Spain. Association for Computational Linguistics.

Dimitrios Galanis and Ion Androutsopoulos. 2010. An extractive supervised two-stage method for sentence compression. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 885–893. Association for Computational Linguistics.

Maya Hristakeva and Dipti Shrestha. 2005. Different approaches to solve the 0/1 knapsack problem. In *The Midwest Instruction and Computing Symposium*.

Hongyan Jing. 2000. Sentence reduction for automatic text summarization. In *Sixth Applied Natural Language Processing Conference*.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1528–1533, San Diego, California. Association for Computational Linguistics.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, 2000:703–710.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Haitao Liu, Chunshan Xu, and Junying Liang. 2017. Dependency distance: A new perspective on syntactic patterns in natural languages. *Physics of life reviews*, 21:171–193.

Nitin Madnani, David Zajic, Bonnie Dorr, Necip Fazil Ayan, and Jimmy Lin. 2007. Multiple alternative sentence compressions for automatic text summarization. In *Proceedings of DUC*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica*, 22(3):276–282.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. Evaluating sentence compression: Pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*.

Shashi Narayan, Claire Gardent, Shay B Cohen, and Anastasia Shimorina. 2017. Split and rephrase. *arXiv preprint arXiv:1707.06971*.

Ani Nenkova, Kathleen McKeown, et al. 2011. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233.

Masanori Oya. 2011. Syntactic dependency distance as sentence complexity measure. In *Proceedings of the 16th International Conference of Pan-Pacific Association of Applied Linguistics*, pages 313–316.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Kristina Toutanova, Chris Brockett, Ke M. Tran, and Saleema Amershi. 2016. A dataset and evaluation metrics for abstractive compression of sentences and short paragraphs. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 340–350, Austin, Texas. Association for Computational Linguistics.

Oren Tsur and Ari Rappoport. 2009. Revrank: A fully unsupervised algorithm for selecting the most helpful book reviews. In *ICWSM*. Citeseer.

Liangguo Wang, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song, and Lejian Liao. 2017. Can syntax help? improving an lstm-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1385–1393.

Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. 2016. A sentence compression based framework to query-focused multi-document summarization. *arXiv preprint arXiv:1606.07548*.

Wei Xu, Alan Ritter, Bill Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. *Proceedings of COLING 2012*, pages 2899–2914.

Yang Zhao, Zhiyuan Luo, and Akiko Aizawa. 2018. A language model based evaluator for sentence compression. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 170–175.

Jiawei Zhou and Alexander Rush. 2019. Simple unsupervised summarization by contextual matching. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5101–5106, Florence, Italy. Association for Computational Linguistics.

WiC-TSV: An Evaluation Benchmark for Target Sense Verification of Words in Context

Anna Breit[1], **Artem Revenko**[1], **Kiamehr Rezaee**[2],
Mohammad Taher Pilehvar[3], **Jose Camacho-Collados**[4]
[1]Semantic Web Company, Austria; [2]Iran University of Science and Technology;
[3]Tehran Institute for Advanced Studies, Iran; [4]Cardiff University, United Kingdom
[1]`{anna.breit,artem.revenko}@semantic-web.com`,
[2]`k_rezaee@comp.iust.ac.ir`, [3]`mp792@cam.ac.uk`,
[4]`camachocolladosj@cardiff.ac.uk`

Abstract

We present WiC-TSV, a new multi-domain evaluation benchmark for Word Sense Disambiguation. More specifically, we introduce a framework for *Target Sense Verification of Words in Context* which grounds its uniqueness in the formulation as binary classification task thus being independent of external sense inventories, and the coverage of various domains. This makes the dataset highly flexible for the evaluation of a diverse set of models and systems in and across domains. WiC-TSV provides three different evaluation settings, depending on the input signals provided to the model. We set baseline performance on the dataset using state-of-the-art language models. Experimental results show that even though these models can perform decently on the task, there remains a gap between machine and human performance, especially in out-of-domain settings. WiC-TSV data is available at `https://competitions.codalab.org/competitions/23683`

1 Introduction

Word Sense Disambiguation (WSD) is a long-standing task in Natural Language Processing and Artificial Intelligence. While progress has been made in recent years, mainly thanks to the surge of transformer-based language models such as BERT (Loureiro and Jorge, 2019; Vial et al., 2019; Huang et al., 2019), the evaluation of WSD models has been limited to a set of (mostly SemEval-based) standard WSD datasets (Raganato et al., 2017). These datasets usually come in one of the two forms: *lexical sample*, in which a target word is placed in various contexts, triggering different senses, and *all-words*, in which all the content words in a given text are to be disambiguated. Both settings, however, come with a major restriction: word senses in the datasets are linked to external sense inventories such as WordNet (Fellbaum, 1998). Therefore, existing benchmarks are limited to only those WSD systems in which sense distinctions are defined according to an underlying sense inventory. This not only gives restrictions to the model's flexibility, but also enforces the assumption of the availability of complete data. However, as general sense inventories are complex to maintain they often lag behind in being up-to-date[1], yielding to the absence of novel terms and term usages. Furthermore, the coverage of domain-specific terms and named entities in general sense inventories is quite limited, while domain-specific sense inventories are rare and in most cases incomplete.

As a motivating example, let us assume *Technology* as the target domain and the collection of information on the current technology landscape as a goal. Therefore, the following context needs to be disambiguated in order to evaluate its relevance:

> *From 1970 to 2007,* ***Apple's*** *chief executive was former Beatles road manager Neil Aspinall.*

Even when incorporating a general sense inventory (which would include senses for the fruit and the tree) and a technology-specific sense inventory (which would include the sense for Apple Inc. the technology company), the actual target sense of this context (i.e., Apple Corps Limited, a multimedia corporation founded by the Beatles) may still be missing, which makes the annotation of the correct sense impossible. For these reasons, the current WSD task formulation and existing benchmarks are not fully able to evaluate the suitability of disambiguation systems in realistic domain-specific and/or enterprise settings.

In this paper, we try to fill this gap by proposing a re-formulation of the existing WSD task as well as a new benchmark for evaluating WSD systems under this paradigm.

---

[1] The last update in WordNet dates back to June 2011.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1635–1645
April 19 - 23, 2021.

Target Sense Verification (TSV) formulates the disambiguation of a word as a binary classification task where the equivalence of the intended sense of a word in context and a single given sense is evaluated. For instance, in the example above, the system would need to decide whether the sentence refers to *Apple Inc.* the technology company or not, by being provided with a sense indicator for solely Apple Inc. (e.g., the hypernym *technology company* or the definition).

A system able to efficiently solve the TSV task could be effectively used in the scenario of collecting and tagging large amounts of textual data; e.g., from social media, news agencies, blogs and for downstream tasks such as information retrieval, sentiment analysis or relation extraction. Furthermore, such a system could be a good candidate for entity linking (EL) as the task statement of TSV resembles the usage of enterprise knowledge graphs (Galkin et al., 2017) for EL: typically, small domain-specific enterprise knowledge graphs only contain entities from the domain of interest, partially or completely missing the general purpose senses of the contained labels.

In order to train and evaluate models for TSV we constructed **WiC-TSV (Word in Context - Target Sense Verification)** a multi-domain dataset and evaluated standard unsupervised and supervised approaches (including language models). While WiC-TSV's training and development set consist of general purpose instances, the test set contains domain-specific instances from three different domains. Therefore, this dataset aims at evaluating the ability of a model to (1) disambiguate the word in context without an external sense inventory, (2) deal with unseen instances and incomplete data, and (3) transfer the intrinsic knowledge (gained on general domain data) into a specific domain.

2 Related Work

Word Sense Disambiguation. The task of WSD consists of associating a word in context with its most appropriate entry in a given sense inventory (Navigli, 2009), e.g., WordNet. For WSD there are many associated datasets (Raganato et al., 2017; Vial et al., 2018; Röder et al., 2018; Ling et al., 2015), including domain-specific ones (Agirre et al., 2009; Faralli and Navigli, 2012). The main difference between WSD and its re-formulation TSV is that for TSV the availability of a sense inventory is not required. Instead of associating a word in context with its most appropriate sense, the usage of a single given sense in the provided context is to be verified. Systems that aim to solve the proposed task are therefore not required to model all senses of the target word, but only a single sense instead.

This facilitates the development of systems for specific domains or settings, as no general-domain knowledge resource is required to perform this task. For instance, an Indonesian company may want to retrieve all sentences referring to the *Java* island and not other unrelated senses. This framing of the task is frequent in business and data mining settings where domain-specific knowledge resources or inventories may be available, without the need for modeling instances from other domains.

WiC. The task closest to the proposed WiC-TSV is probably Word-in-Context (Pilehvar and Camacho-Collados, 2019, WiC), which our dataset is based on. WiC is a binary classification dataset where a target word is presented within two different contexts. The task consists of deciding whether the word is associated with the same sense in the two contexts or not. WiC is also one of the tasks included in the general language understanding framework SuperGLUE (Wang et al., 2019).

WiC-TSV inherits some of the desirable properties of WiC, such as independence from external sense inventories and the binary classification nature of the task. However, though our benchmark draws ideas from the Word-in-Context benchmark, it provides a different evaluation setting with additional flavors. The main difference with respect to our dataset lies in the presence of relevant information such as hypernyms and definitions, which makes our dataset more realistic and a direct proxy for downstream evaluation: in WiC-TSV the ambiguous target word in a single context is compared against a specific target sense (indicated by provided hypernyms and definitions), in contrast to the comparison of the intended senses of the target word in two different contexts. Also, the task is more targeted at word-level representation, as in one of the tasks (i.e. *hypernymy* task) the model is not provided with any contextual information and, therefore, needs to have a clear understanding of the word to be able to make correct judgements. Moreover, WiC-TSV includes instances from three domains (cocktails, medicine, and computer science) in its test set, which makes the benchmark more challenging and comparable to a real setting.

3 WiC-TSV: The Benchmark

A goal of this benchmark is to evaluate the ability of a model to verify the target sense of a word in a context without the usage of an external sense inventory, i.e., without knowing all possible senses of the target word. Another model quality that is aimed at with the presented benchmark is the ability to transfer the intrinsic knowledge into a specific domain. As for most areas, domain-specific training data is hard to obtain, being able to learn on general purpose data and still perform well on domain-specific data is a huge advantage in a real world setting.

To this end, we constructed a benchmark satisfying following requirements:

1. Knowledge of only a **single sense** of the target word;
2. Knowledge of the **definition and/or hypernyms** of the target sense;
3. Ability to test the models capability to disambiguate both **general purpose and domain-specific** senses;
4. Ability to test the models capability to classify usages of previously **unseen words**;

Formally, each instance in the dataset consists of a target word w, a context c containing the target word w, and its corresponding target sense s represented by either its definition (Task 1), its hypernym/s (Task 2), or both definition and hypenyms (Task 3). The task aims to determine whether the intended sense of the word w used in the context c matches the target sense s.

Table 1 contains examples of instances from the WiC-TSV test set. Furthermore, a small sample of 10 instances is available online in the form of a survey[2], where the achieved score is shown to the user after the submission.

3.1 Dataset Construction

In this section we detail the construction of the dataset. First, we describe the construction of the training and development set (Section 3.1.1) and then the test set (Section 3.1.2), with a special focus on the creation of the domain-specific subsets.

3.1.1 Training and Development Set

Instances in the training and development set do not focus on a specific domain. As basis served the Word-in-Context (WiC) dataset (Pilehvar and Camacho-Collados, 2019), which contains a target word w and two contexts c_1 and c_2 for each instance. The contexts from WiC for noun instances come from two resources: WordNet and Wiktionary. To maintain the desirable characteristics of the WiC dataset (e.g., balanced data, not having repeated contexts across instances), the splits of the original training and development sets were treated separately in the following way: starting from a noun-only sub-sample, for each context c_i, the sense of the target word w was mapped to the corresponding synset of WordNet, adding a sense identifier. Each WiC instance was then split into two instances, one for each context. For initial negative instances (i.e. w has different intended senses in c_1 and c_2), the sense identifiers of these two instances were switched. To avoid information leakage, only one of the two instances were kept for the WiC-TSV dataset[3]. Finally, for each sense, the definition and hypernyms were derived from WordNet using the sense identifiers.[4]

3.1.2 Test Sets

To make the dataset more challenging and realistic, the test set incorporates both general purpose and domain-specific instances.

General Purpose (WNT/WKT). The general purpose instances were generated analogously to 3.1.1. Hence, this test set is composed of both WordNet and Wiktionary examples, with definitions and hypernyms extracted from WordNet.

In the following we describe the construction of the domain-specific subsets. The main difference between domain-specific and WNT/WKT test sets is that in the former the target sense remains the same. That means, that even though "fork" might have different senses within the computer science domain, we are only interested in one of these senses.

Cocktails (CTL). For the cocktails instances the target words were taken from the "All about cocktails" thesaurus[5]. The thesaurus contains 300 en-

---

[2] https://www.surveymonkey.com/r/LHYWXPV

[3] If both instances were kept, the label could have been predicted with a high accuracy by counting the appearance of the target sense (even=True, odd=False).

[4] WordNet sense identifiers are omitted in the final dataset.

[5] vocabulary.semantic-web.at/cocktails

Tag	Context	Definition	Hypernyms
General Purpose (WNT/WKT)			
T	**Smoking** is permitted .	the act of smoking tobacco or other substances	breathing, respiration, ventilation
F	all that work went down the **sewer**	someone who sews	needleworker
Cocktails (CTL)			
T	We were 11 at table for this feast . We started the evening with **Bellini** , made with fresh , Niagara peaches . (Thank you , Jack Lalanne Juicer !)	A Bellini cocktail is a mixture of Prosecco sparkling wine and peach purée.	cocktail
F	After a morning 's work I went off to see the **Bellini** retrospective at the Quirinale . Beautiful !	A Bellini cocktail is a mixture of Prosecco sparkling wine and peach purée.	cocktail
Medical Subjects (MSH)			
T	Italy now reports the second highest number of **corona** cases wordlwide .	A viral disorder characterized by high fever; cough; dyspnea; renal dysfunction and other symptoms of a viral pneumonia.	viral_pneumonia; coronavirus_infection
F	**Corona** Labs is happy to announce the general availability of the public beta of Android 64-bit Corona builds .	A viral disorder characterized by high fever; cough; dyspnea; renal dysfunction and other symptoms of a viral pneumonia.	viral_pneumonia; coronavirus_infection
Computer Science (CPS)			
T	pandas is a fast , powerful , flexible and easy to use open source data analysis and manipulation tool , built on top of the **Python** programming language .	Python is an interpreted, high-level, general-purpose programming language	object_oriented_programming_language
F	The present paper compares the recently studied **pythons** with those examined 20 years ago , and uses the combined dataset to assess the ecological sustainability .	Python is an interpreted, high-level, general-purpose programming language	object_oriented_programming_language

Table 1: Sample instances from WiC-TSV. Target words are marked in bold. Tags: T (True) and F (False).

tries describing not only cocktails, but also beverages, garnishes and glassware, among others. For instances obtained from this resource, the hypernym "cocktail" is used in the WiC-TSV dataset, while the definition is derived from the thesaurus.

Medical Subjects (MSH). For medical subject instances we use terms, definitions and hypernyms from the MeSH thesaurus[6]. This thesaurus is used for indexing medical articles and therefore contains a wide variety of terms in this domain. We considered various types, such as diseases, symptoms and body parts as target words.

Computer Science (CPS). Target words in the computer science domain were gathered manually, without a readily available thesaurus. The definitions were derived from the lead section of the corresponding Wikipedia page, while hypernyms were created by the consensus of two domain experts.

In order to create the domain-specific instances, first a list of ambiguous words and their domain-specific target senses was fixed for each domain. Then, we used the Wikilinks dataset (Singh et al., 2012) as a basis for collecting different contexts containing the target words. This dataset contains documents – blog posts scraped from the web – and the links from these documents to the Wikipedia pages, which were used to assign the intended sense (i.e., target sense or other sense) to the target word. Where needed, additional contexts were collected manually by incorporating a search engine to find contexts for the target word. The intended senses for these instances were assigned manually.

Postprocessing. After creating the initial domain-specific instances, the subsets were checked manually to remove non-suitable and unsolvable instances. To maintain a rather realistic evaluation setup, data was not completely cleaned, meaning that contexts can contain *noisy* elements such as headings or meta-info derived from the websites (e.g., "posted by").

3.2 Data Cleaning

While the quality of the domain-specific instances is assured due to their manual creation process,

---

[6] `www.nlm.nih.gov/mesh/`

an additional data cleaning step was introduced in which general purpose instances were manually curated. The instances from the test set were split into four sets with an overlap of 20%. Each set was evaluated by an annotator regarding correctness and solvability of the instances. For example, when the hypernym of an instance was too generic to help in the disambiguation process, or the context itself was too ambiguous, the instance was marked as "to filter out". Each marked instance was reviewed by a second annotator, who could either confirm, or reject the request of removal. Instances marked by both annotators were removed.

An example of such a removed instance would be the context *"The* ***zero*** *sign in American Sign Language is considered rude in some cultures ."* for the target word "zero" with the target definition '*a mathematical element that when added to another number yields the same number*'. In American Sign Language (ASL), "zero sign" is a ring-shaped hand sign using the thumb and pointing finger, similar to the OK-gesture. The provided instance mixes two senses of "zero sign". On the one hand, it refers to the hand gesture itself (synonymous to OK-gesture) which does not fully match the target sense. On the other hand, it also refers to the sign of the digit zero in ASL, which does match the target sense.

Other examples of filtered instances involve sentences where the target word may have been used metaphorically.

This procedure resulted in 106 instances which were removed. About 8% of these instances were part of evaluation sets created to measure the human performance (see 3.4)[7]: the annotators achieved a mean accuracy of only 56% on these instances. This shows that the data cleaning step was necessary in order to ensure the data quality of the test set.

3.3 Statistics

A statistical overview of the dataset and their splits is shown in Table 2. The totality of 3832 available instances were split into train, development and test sets with a ratio of 56:10:34 which allows a sophisticated analysis of the generalisation capabilities of tested systems, while still providing an appropriately sized training set.

The test set contains around 55% general purpose instances and 45% from specific domains. For each domain, the number of unique target words is relatively low compared to the general domain subset, which results in a higher number of instances per target word. However, for domain specific words, a great variety of senses is used in the contexts, yielding a big diversity among the instances. For all three splits, positive and negative instances are approximately balanced.

		Total	N_w	R_+
Train	**WNT/WKT**	2137	864	0.56
Dev	**WNT/WKT**	389	377	0.51
Test	**General-domain (WNT/WKT)**	717	664	0.54
	Domain-specific (MSH+CTL+CPS)	589	25	0.47
	MSH	205	8	0.52
	CTL	216	9	0.43
	CPS	168	8	0.46
	All	1306	689	0.51

Table 2: Statistics of training, development and testing splits of WiC-TSV, including total number of instances (Total), unique number of target words (N_w) and percentage of positive instances (R_+).

3.4 Human Performance

To estimate the human performance upper bound, a sub-sample of the test set was manually annotated. The performance was evaluated on the setting of Task 3, meaning that both the definition and the hypernyms were provided to disambiguate. A random selection of 250 instances were split into two evaluation sets of the size of 150, resulting in a 20% overlap. Each evaluation set was assigned to a non-expert annotator with English as native language. No additional information - especially not from the respective ontology or about other senses of the target - was provided to the annotators and they were instructed not to use external knowledge sources (e.g. if they are not familiar with the domain-specific sense of a word).

Results of the human performance evaluation can be found in Table 3. The mean accuracy for the evaluated datasets was 85%, with individual scores of 81% and 89%. To estimate the inter-annotator reliability, the agreement of the two annotators on the overlapping instances was calculated: for 42 instances (84%) the annotators agreed on the label.

When evaluating the instances per domain, it can be seen that the general purpose instances were

---

[7] Annotations for these instances were removed before calculating the metrics presented in 3.4

	Human Perf.
WNT/WKT	82.1
MSH	89.1
CTL	92.0
CPS	86.5
All	85.3

Table 3: Average human accuracy for native English annotators, on different subsets of the dataset: general purpose, i.e., WNT/WKT, and the domain specific, i.e., MSH, CTL, and CPS.

more difficult than the domain-specific ones, as annotators achieved an average accuracy of 82% (individual scores of 77% and 87%) on the general purpose instances, while the mean accuracy on the domains were 89% (83% and 96%), 92% (88% and 96%), and 86.5% (89% and 84%) for MSH, CTL, and CPS, respectively. This performance difference is even more evident when comparing to the performance of non-native speakers: an additional experiment showed, that evaluators whose mother language is not English only achieved an average accuracy of about 77% on the WNT/WKT instances, while performances on the domain specific subsets were comparable to native speakers.

4 Experimental Results

In this section we evaluate the performance of different baseline models on our WiC-TSV benchmark. For our experiments we considered two main systems, namely BERT (Devlin et al., 2019) and FastText (Joulin et al., 2017), as well as unsupervised baselines adapted to the corresponding tasks in WiC-TSV.

4.1 Evaluation Tasks

The benchmark provides three different tasks depending on the input information available: definition-based (Section 4.1.1), hypernym-based (Section 4.1.2), and both (Section 4.1.3).

4.1.1 Task 1: Definition Information

In this task, the goal is to identify if the intended sense of the target word in the *context* matches the target sense described by the *definition*.

In other words, the model has to check if the sense represented by the *definition* can fit within the given *context*. For this task, the system is provided with a *context* (in which the target word is marked) along with a *definition* (which describes one of the possible senses of the word).

Baselines. The first baseline is based on the pre-trained transformer-based language model **BERT**[8]. It consists of a simple classification layer on top of the BERT model which is responsible for encoding the input. For this task, we concatenate the context and the definition and feed the whole sequence to BERT. Then, the classification layer takes as input the concatenation of three different vectors, all provided by BERT: the `[CLS]` token representation, the representation of the target word in the context and the average representation of the words in the definition. This is similar to the baseline BERT model employed in SuperGLUE (Wang et al., 2019). It is worth mentioning that BERT is originally trained using WordPiece tokenization (Wu et al., 2016), which means that each word can be broken down into more than one sub-word. Therefore, in order to have a fixed length representation for each word, we take the average of its sub-word representations. Finally, the whole model is fine-tuned on the training set.

For the **FastText**-based baseline, we first extract the corresponding embeddings for each word in the context and definition, respectively. Then, the representation is simply computed as the average of the corresponding embeddings it contains. Next, these two representations are concatenated together to form a fixed length vector which we then feed to a fully connected layer. Finally, we put a simple classification layer on top of this fully connected layer and train the model on the training set.

We also evaluated **GlossBERT** (Huang et al., 2019) on our dataset. The authors describe a weak supervision algorithm that consists in surrounding the target word with special symbols – quotation marks are used in the available implementation[9]. We provide results both with (GBERT$_{ws}$) and without (GBERT) weak supervision. We chose the hyper-parameters as suggested by the authors, trained for 6 epochs and achieved the highest scores on the 4th epoch.

Unsupervised baselines **U-BERT** and **U-dBERT**, which do not make use of the training set, are simple threshold-based classifier which take the cosine distance of a target word representation and

---

[8] We used the implementation of BERT available at `https://github.com/CyberZHG/keras-bert` for the *base* (BERT-B) and *large* (BERT-L) pre-trained models.

[9] `https://github.com/HSLCY/GlossBERT`

a definition representation into account. As source for these vectors serve BERT and DistilBERT, respectively.

Similar to before, we derive the target word vector by taking the embedding of the target word in the context and the definition vector by averaging over all embeddings of the definition. The threshold is tuned on the development set with a step size of 0.02.

4.1.2 Task 2: Hypernym Information

For this task, the system is provided with a target word (in a *context*) and a set of *hypernyms* for the target sense. The task is to identify if the intended sense given through the context is the hyponym of the provided hypernyms. Note that, unlike Task 1, no *definition* is involved in this setting and the task is directed only by hypernymy information.

Baselines. We used baseline models similar to those used in the previous task. The only difference lies in how we shape the inputs fed to these models.

For the supervised and unsupervised BERT-based models, we put together the context with the hypernyms to form the input. Similarly, for the FastText-based model, the hypernyms' embeddings are concatenated with the context's representation and fed to the classifier.

4.1.3 Task 3: Both Sources of Information

In the third task systems are provided with both definition and hypernymy information.

Baselines. For this task, we concatenate the definition and the hypernyms, and feed the generated sequence together with the context to BERT. Then, the concatenation of the [CLS] token representation, the representation of the target word in the context and the average representation of the words in the definition/hypernyms sequence is fed to the classification layer.

For the unsupervised model we use the same BERT input and take the representation of the target in context and the average over the definition and hypernyms as input vectors. For the FastText-based baseline, the hypernyms' embeddings are concatenated with both the context's representation and the definition representation and the combination is fed to the classifier.

		WiC-TSV			
		Acc	**Prec**	**Rec**	**F1**
Task-1 (def)	BERT-B	75.3	71.7	84.9	77.7
	BERT-L	75.3	70.4	**88.5**	78.4
	FastText	53.7	54.1	57.6	55.7
	GBERT	**76.0**	71.3	88.2	**78.8**
	GBERT$_{ws}$	75.9	71.2	88.1	**78.8**
	U-dBERT	56.9	**76.0**	22.0	34.2
	U-BERT	54.4	73.1	16.0	26.2
Task-2 (hyp)	BERT-B	71.4	67.7	83.5	74.8
	BERT-L	**75.3**	**71.7**	**85.1**	**77.8**
	FastText	52.7	52.4	73.6	61.1
	U-dBERT	62.3	64.8	56.3	60.2
	U-BERT	62.8	65.9	55.2	60.1
Task-3 (both)	BERT-B	**76.6**	**74.1**	82.8	78.2
	BERT-L	76.3	72.6	**85.7**	**78.6**
	FastText	53.4	52.8	79.4	63.4
	U-dBERT	61.2	70.6	40.3	51.3
	U-BERT	60.5	68.0	41.9	51.9
Baseline$_{\text{True}}$		50.8	50.8	100	67.3
Human		*85.3*	*80.2*	*96.2*	*87.4*

Table 4: Test set performance of the baseline models on WiC-TSV, in terms of accuracy, precision, recall, and F1, on the three different tasks. Baseline$_{\text{True}}$ is a naive baseline that always returns *True* and the human performance is computed as described in Section 3.4.

4.2 Results

Table 4 shows the overall results for the three tasks. As can be observed, GlossBERT performs best in terms of accuracy and F$_1$. BERT-L is a little worse, but achieves the best recall. The worst supervised baseline – FastText – does not perform better than a naive baseline that retrieves all instances as true. This also reinforces the challenging nature of the benchmark, as even BERT-based models are far from the human annotator performance (estimated on 85.3% for accuracy). Clearly, the definition information is more helpful than the hypernyms for BERT, while the combination of both attains the best overall results. Yet GlossBERT reaches a better performance with definition only[10].

The unsupervised models only perform well with hypernyms. Though U-dBERT reaches the best precision in Task 1, the recall remains very low and therefore the overall performance.

Another point to highlight is the high recall of BERT-based models, in contrast to its precision. This is mainly attributed to the domain-specific subsets as it will be analysed below. As for the comparison between BERT-based models, the larger model (BERT-L) performs as expected better than

---

[10] We did not evaluate GlossBERT with hypernyms as such-configuration was not considered by the authors in the original system and its integration would not be straightforward.

		WNT/WKT				CTL				MSH				CPS			
		Acc	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**
T1 def	BERT-B	73.3	74.0	77.7	75.8	**76.2**	65.1	**98.9**	**78.4**	77.6	73.4	89.0	80.4	80.0	70.5	97.9	81.9
	BERT-L	**77.1**	**75.7**	**84.7**	**80.0**	73.1	62.3	95.3	75.4	75.3	70.6	89.6	78.9	70.2	61.4	97.4	75.3
	FastText	56.2	58.9	61.9	60.3	49.8	39.0	30.8	34.3	51.7	52.2	79.2	62.9	50.4	45.6	38.5	41.6
	GBERT	75.7	74.9	82.6	78.6	75.5	64.9	93.5	76.7	74.1	67.5	96.2	79.4	79.8	70.0	**98.7**	81.9
	GBERT$_{ws}$	75.2	74.6	81.6	78.0	70.4	59.7	95.7	73.6	**78.5**	71.5	**97.2**	**82.4**	**82.7**	73.3	**98.7**	**84.2**
	U-BERT	49.2	64.1	13.0	21.6	57.4	**100**	1.1	2.1	62.0	78.0	36.8	50.0	63.1	**100**	20.5	34.0
	U-dBERT	51.5	67.0	19.4	30.1	56.9	0.0	0.0	0.0	65.9	**86.0**	40.6	55.1	69.0	93.3	35.9	51.9
T2 hyp	BERT-B	68.6	70.0	72.9	71.4	77.9	66.6	97.8	79.3	71.9	65.1	**98.4**	78.3	74.4	64.7	**98.7**	78.2
	BERT-L	**71.4**	**71.7**	**77.4**	**74.4**	**82.7**	**72.7**	96.4	**82.8**	77.2	71.2	94.0	**81.0**	**80.6**	**71.3**	97.4	**82.3**
	FastText	56.8	58.9	66.3	62.1	43.1	43.0	**99.3**	60.0	49.1	50.4	84.0	62.9	52.0	48.8	65.0	55.3
	U-BERT	57.6	61.3	57.8	59.5	57.4	52.0	14.0	22.0	74.6	79.3	68.9	73.7	77.4	77.0	73.1	75.0
	U-dBERT	55.0	58.1	58.5	58.3	56.0	42.9	6.5	11.2	**80.5**	**86.7**	73.6	79.6	79.2	75.9	80.8	78.3
T3 both	BERT-B	73.5	76.1	74.2	75.1	**79.2**	67.8	98.2	**80.2**	**79.8**	75.8	89.6	**82.1**	**82.1**	73.0	**97.9**	**83.6**
	BERT-L	**77.3**	**77.2**	**82.1**	**79.6**	76.4	67.0	90.0	76.6	75.4	71.6	87.4	78.7	72.8	63.8	96.2	76.7
	FastText	57.1	58.0	74.0	65.0	43.1	43.1	**100**	60.2	51.1	51.5	**90.3**	65.6	54.0	50.5	67.1	57.3
	U-BERT	54.4	61.3	41.5	49.5	58.8	62.5	10.8	18.3	71.2	78.3	61.3	68.8	75.6	**87.8**	55.1	67.7
	U-dBERT	54.8	62.4	40.4	49.1	57.9	**75.0**	3.2	6.2	74.1	**87.3**	58.5	70.1	76.8	86.8	59.0	70.2
Baseline$_{\text{True}}$		53.8	53.8	100	70.0	43.1	43.1	100	60.2	51.7	51.7	100	68.2	46.4	46.4	100	63.4

Table 5: Performance for the baseline models for the three tasks (i.e., T1: definition-based, T2: hypernymy-based, and T3: both sources of information) split by domain: General (WNT/WKT), Cocktails (CTL), Medical Subjects (MSH), and Computer Science (CPS). Baseline$_{\text{True}}$ is a naive baseline that always returns "True". Human performance in terms of accuracy is estimated to be 82.1% (WNT/WKT), 92.0% (CTL), 89.1% (MSH) and 86.5% (CPS) as described in Section 3.4.

the base model (BERT-B) overall.

4.3 Analysis

In order to better understand the results, in this section we perform a focused analysis on the performance split by domain.

4.3.1 Domain-based Analysis

Table 5 presents the results split by domain. FastText faces a massive challenge in adapting to new domains and generalising from WNT/WKT to the other domains. However, BERT-based models show to be much more robust to domain changes. In fact, the results on the domain-specific instances are in the same ballpark as the WNT/WKT test set. This can be attributed to the fact that specific domains highly constrain the set of possible senses for a word, resulting in an easier WSD classification task (Magnini et al., 2002). On the other hand, WordNet is known to be quite fine-grained (e.g., the noun *run* has 16 different senses in WordNet).

Surprisingly, unsupervised DistilBERT achieves the best accuracy over all tasks and classifiers on MSH. However, both unsupervised models do not perform well on WNT/WKT and CTL. We can observe that supervised models are significantly more reliable and produce similar scores on different tasks and datasets than unsupervised models.

In general, for BERT-based models, recall is substantially higher than precision on the domain-specific subsets. This is desirable in a retrieval setting where a high-coverage retrieval of relevant cases is of more importance. Interestingly, among the two BERT alternatives, the smaller model performs better on the domain-specific subsets, suggesting that it is more robust to domain changes. This is an important observation which needs further careful investigation in future work, given that most evaluation benchmarks (on which the larger model consistently outperforms the smaller one) comprise in-domain test sets, which cannot reveal robustness across domains.

4.3.2 In-domain Few-shot Analysis

Although the availability of big annotated domain-specific training sets is quite rare, the presence of a small training set forms a realistic scenario. Incorporating these domain-specific instances in the model training could potentially increase its prediction performance. To investigate this theory, we performed an additional analysis focusing on the usage of in-domain examples in the learning process, where for each domain 100 instances from the test sets were used as a training set. To enforce the assumption that not all target senses would be seen during the training process, we put aside all instances of 3 target words for each domain test set.[11] Two additional domain-based strategies were considered: (1) *few-shot learning:* only using the domain-specific instances, and (2) *continued learning:* extending the existing general-purpose training set with the domain-specific instances.

For this analysis we focused on Task-3 and BERT-large, which performed better overall. Table 6 shows the F1 results. In general, few-shot learning works surprisingly well overall (achieving the best overall performance in the CTL and MSH domains). On CTL pairs unseen during training, it even performs considerably better than the same BERT model trained in the continued learning setting. In the CPS domain, for both few-shot in-domain learning and continued learning the performance on seen target words is quite high, while the prediction of unseen target words produces relatively low F1 scores, which indicates a low ability to generalise to new senses. As for the model trained on the general-domain dataset, it performs best in the CPS domain, but performs considerably lower than the domain-tuned alternatives in the CTL domain. Indeed, the domain-tuned BERT systems clearly outperform the same model trained on the general domain on seen pairs, proving the importance of obtaining word-specific examples to boost performance. However, this may not be realistic in practice, and therefore further research should be devoted in improving the generalization capabilities of disambiguation systems, and language models in particular. These findings are consistent with the results of an experiment conducted with GlossBERT in the few-shot learning setting on Task-1: the overall accuracy increase ranged from 0.1% (CPS) to 13.6% (CTL) compared to the model trained solely on general domain instances.

Train	**CTL**			**MSH**			**CPS**		
	All	**See**	**Uns**	**All**	**See**	**Uns**	**All**	**See**	**Uns**
WNT	76.5	74.7	75.5	77.4	82.9	76.0	**74.9**	77.8	**74.8**
WNT+D	80.2	**89.5**	75.5	78.5	86.2	**76.6**	73.6	88.9	72.9
Dom	**84.2**	88.8	**82.6**	**78.7**	**88.6**	75.5	70.7	**93.3**	69.1
Base$_{\text{True}}$	58.5	54.8	59.1	68.2	70.9	67.5	62.6	58.3	62.8

Table 6: F1 score for the in-domain few-shot analysis (Task-3) using BERT-L trained on general domain (WNT), domain-specific (Dom) and general domain fine-tuned on the target domain (WNT+D). In addition to the full test set (All), results are split on seen (See) and unseen (Uns), as per the presence or absence of the target word in the domain-specific training set.

5 Conclusions and Future Work

In this paper we have introduced the Target Sense Verification task, a re-formulation of WSD where the equivalence of the intended sense of a word in context and a single given sense is evaluated. Furthermore, we presented WiC-TSV, a multi-domain benchmark which differs from existing WSD datasets in three main ways: (1) it is based on TSV and therefore framed as a binary classification task where only one target sense needs to be verified, (2) it is independent from external sense inventories, and (3) its test set contains instances from three specific and heterogeneous domains are included: cocktails, medical subjects and computer science. Our benchmark therefore opens the floor for different disambiguation algorithms that do not require modeling the entirety of a sense inventory. This characteristic also provides a crucial advantage in enterprise and domain-specific settings as it facilitates the development of systems which are only aimed at modelling the domain at hand. Moreover, having these out-of-domain test instances makes our benchmark more robust and generalisable, preventing (or making it harder) for statistical models to learn spurious correlations from the training set, which has been proven to be an issue in standard NLP tasks (Poliak et al., 2018; Gururangan et al., 2018; Linzen, 2020).

In our initial experiments we found that current state-of-the-art disambiguation techniques based on pre-trained language models such as BERT are very accurate at handling ambiguity, even in specialised domains. However, there is still room for improvement as highlighted by the gap with the human performance. This benchmark therefore opens up avenues for future research on domain-transfer and on developing general-purpose solutions which can perform well on a variety of domains without the need for large amounts of training data.

As future work, we are planning to further investigate and analyse the robustness of pre-trained models with respect to domain changes. Also, it would be interesting to develop hybrid models which take both definition and hypernymy information into account – in this paper we combined both sources in BERT in a simple manner, but more complex models should lead to further improvements.

---

[11] To add robustness to the results, three different random samples were considered for this experiment, with the results being averaged after the three different runs.

References

Eneko Agirre, Oier Lopez de Lacalle, Christiane Fellbaum, Andrea Marchetti, Antonio Toral, and Piek Vossen. 2009. SemEval-2010 task 17: All-words word sense disambiguation on a specific domain. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*, pages 123–128, Boulder, Colorado. Association for Computational Linguistics.

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

Stefano Faralli and Roberto Navigli. 2012. A new minimally-supervised framework for domain word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1411–1422, Jeju Island, Korea. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA.

Mikhail Galkin, Sören Auer, María Esther Vidal, and Simon Scerri. 2017. Enterprise knowledge graphs: A semantic approach for knowledge management in the next generation of enterprise information systems. In *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems*, volume 2, pages 88–98.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3507–3512, Hong Kong, China. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of ACL*.

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through WordNet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691, Florence, Italy. Association for Computational Linguistics.

Bernardo Magnini, Carlo Strapparava, Giovanni Pezzulo, and Alfio Gliozzo. 2002. The role of domain information in word sense disambiguation. *Natural Language Engineering*, 8(4):359–373.

Roberto Navigli. 2009. Word Sense Disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

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

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of EACL*, pages 99–110, Valencia, Spain.

Michael Röder, Ricardo Usbeck, and Axel Cyrille Ngonga Ngomo. 2018. Gerbil - Benchmarking named entity recognition and linking consistently. *Semantic Web*, 9(5).

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2012. Wikilinks: A large-scale cross-document coreference corpus labeled via links to Wikipedia. Technical Report UM-CS-2012-015, University of Massachusetts, Amherst.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2018. UFSAC: Unification of Sense Annotated Corpora and Tools. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2019. Sense vocabulary compression through the semantic knowledge of wordnet for neural word sense disambiguation. In *Proceedings of the 10th Global WordNet Conference*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of NeurIPS*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Self-Supervised and Controlled Multi-Document Opinion Summarization

Hady Elsahar,[1] **Maximin Coavoux,**[2] **Matthias Gallé,**[1] **Jos Rozen**[1]
{hady.elsahar,matthias.galle,jos.rozen}@naverlabs.com
maximin.coavoux@univ-grenoble-alpes.fr
[1]Naver Labs Europe
[2]Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG

Abstract

We address the problem of unsupervised abstractive summarization of collections of user generated reviews through self-supervision and control. We propose a self-supervised setup that considers an individual document as a target summary for a set of similar documents. This setting makes training simpler than previous approaches by relying only on standard log-likelihood loss and mainstream models. We address the problem of hallucinations through the use of control codes, to steer the generation towards more coherent and relevant summaries. Our benchmarks on two English datasets against graph-based and recent neural abstractive unsupervised models show that our proposed method generates summaries with a superior quality and relevance, as well as a high sentiment and topic alignment with the input reviews. This is confirmed in our human evaluation which focuses explicitly on the faithfulness of generated summaries. We also provide an ablation study showing the importance of the control setup in controlling hallucinations.

1 Introduction

Recent progress in unsupervised methods has created breakthroughs in natural language processing applications, such as machine translation (Artetxe et al., 2018; Lample et al., 2018). Those have been mostly based on a bootstrapping approach, which consists in iteratively alternating between two representations, and optimizing a reconstruction loss. Beyond machine translation, other applications include Question-Answering (Lewis et al., 2019) and parsing (Drozdov et al., 2019). While similar ideas have been applied as well for video summarization (Yuan et al., 2019), such a bootstrapping approach seems less suited for summarization, because of the inherent information loss when going from the full text to the summarized one. Existing unsupervised approaches for summarization therefore relied mostly on extractive graph-based systems (Mihalcea and Tarau, 2004). Only recently have there been proposals for unsupervised abstractive summarization, using auto-encoders (Chu and Liu, 2019; Bražinskas et al., 2020). However, these set-ups are complex and require a combination of loss functions (Chu and Liu, 2019) or hierarchical latent variables (Bražinskas et al., 2020) to ensure that the generated summaries remain on-topic.

In this paper, we investigate a self-supervised approach for multi-document opinion summarization. In this setting, there are multiple opinions (reviews), one entity (products, venues, movies, etc) and the goal is to extract a short summary of those opinions. Our approach is based on self-supervision and does not require any gold summaries. We train a supervised model on examples artificially created by selecting (i) one review that will act as a target summary and (ii) a subset of reviews of the same entity that acts as a document collection.

Neural models have a known problem of hallucination (Rohrbach et al., 2018), which can be misleading in natural language generation tasks as the fluency of those models often distract from the wrong facts stated in the generated text. To reduce this effect, we propose to use control tokens (Fan et al., 2018; Keskar et al., 2019; Krause et al., 2020). Control tokens are discrete variables that are used to condition the generation. Different from previous work, our goal is not to allow users to control the generated text, but instead to steer the generated text to produce an output which is consistent with the input documents to be summarized.

Our main contributions are therefore three-fold:

- performing multi-document summarization by modelling it as a self-supervised problem where one document acts as the summary of a subset. We carefully select those two, and

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1646–1662
April 19 - 23, 2021.

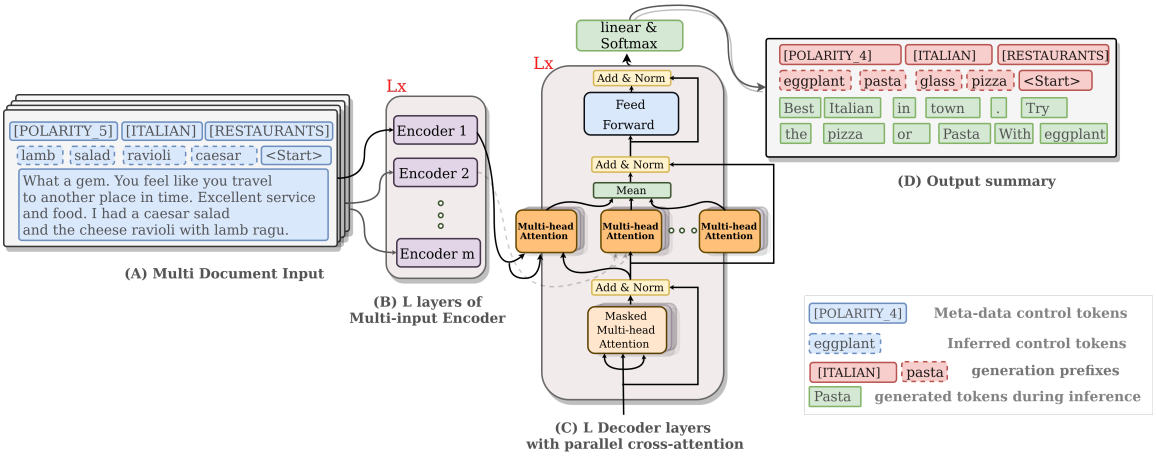


Figure 1: Description of our proposed model: (A) is the set of input reviews, augmented with control tokens (from meta-data in uppercase, inferred in lowercase). (B) is the encoder, which is run separately on each input review. The standard Transformer decoder is modified in (C) to allow for Parallel cross-attention on different inputs separately. Finally, (D) is the generated output. During inference the control tokens are fed as prompts to the decoder and generation starts afterwards.

link the resulting formulation to a recent theoretical framework (Peyrard, 2019) (Sect. 3);

- using control tokens to steer the model towards consistency, increasing relevance of the generated summary (Sect. 4);
- an application of multi-input transformer model (Libovický et al., 2018) to summarization. This model encodes each input independently, and at decoding time applies parallel attention to each encoded input (Sect. 5).

Our experimental results (Sect. 6 and 7) show that our approach outperforms existing models on two datasets: Yelp reviews on venues (Chu and Liu, 2019) and Rotten Tomatoes movie reviews (Wang and Ling, 2016). We focus the human evaluation on the faithfulness of the summaries, confirming they are more factually correct than baselines.

2 Related Work

Unsupervised Opinion Summarization Extractive summarization consists in selecting a few sentences from the input documents to form the output summary. The centroid method (Radev et al., 2004; Rossiello et al., 2017; Gholipour Ghalandari, 2017) consists in ranking sentences according to their relevance to the whole input. Graph-based methods, such as LexRank (Erkan and Radev, 2004) or TextRank (Mihalcea and Tarau, 2004; Zheng and Lapata, 2019), use the PageRank algorithm to find the most central sentences in a graph of input sentences, where edge weights indicate word overlap. In contrast to these methods, we focus on abstractive summarization methods.

Non-neural abstractive methods (Ganesan et al., 2010; Nayeem et al., 2018) are also graph-based, but work on word-type graphs. West et al. (2019) introduced a self-supervised model for sentence compression: they use an unsupervised extractive system to generate training data for a supervised sentence compressor. Their system works on single sentences whereas our end-to-end approach summarizes multiple reviews.

Recently, a few approaches for *neural* unsupervised abstractive summarization have been proposed. Chu and Liu (2019, MeanSum) introduced a summarization system based on a review autoencoder. At inference time, MeanSum encodes every review for a product to a vector, computes the centroid of reviews and uses this centroid to seed the decoder and generate a summary. However, averaging representations of statements that are sometimes contradictory tends to confuse the decoder, and might lead it to ignore the input signal. To deal with this limitation, Coavoux et al. (2019) add a clustering step to group similar reviews and to generate one sentence per such found cluster. Bražinskas et al. (2020) proposed to solve the problem of unsupervised opinion summarization with an auto-encoder with latent variables. They use latent variable for products and reviews to address the hallucination issue, while at the same time allowing it to capture information from the set of reviews on the same entity. In contrast, we argue that our self-supervised setting is simpler as it relies on training with standard models. In addition, the use of Transformer (as opposed to GRU in their case) makes it possible to apply separate attentions to each input. Probably most similar to our self-supervised proposal is the recent work of Amplayo and Lapata (2020), in particular their document noising sub-strategy. Compared to it, our simple selection criteria of the dataset avoids any use of (domain-specific) noise generator. In addition, our use of control tokens allows to easily include existing (or inferred) meta-information. A similar approach is also used by Shapira and Levy (2020), which trains a seq2seq model by clustering reviews and using the medoid as target summary.

Another work that has recently shown the promise of self-supervision for summarization is Zhang et al. (2020a), in which masked-out sentences are predicted from the surrounding text. Our self-supervision training mechanism can be seen as a multi-document version of that.

Controlled Generation Controllable text generation has been previously investigated to apply global constraints on text generation, by directly optimizing evaluation metrics through policy gradient methods (Ranzato et al., 2016; Wu et al., 2018; Liu et al., 2017; Li et al., 2016b; Yi et al., 2018) or continuous approximation methods (Chu and Liu, 2019; Yang et al., 2018).

Other methods applied control only at inference time. Weighted decoding (Holtzman et al., 2018) was shown to be challenging, and often detrimental to fluency and coherence (See et al., 2019). Constrained beam search (Anderson et al., 2017; Hokamp and Liu, 2017; Post and Vilar, 2018) is slower, requires very large beam sizes, and does not enable soft constraints. Finally, updating the decoder hidden states (Chen et al., 2018; Dathathri et al., 2020) requires an extra training step.

Control codes have been introduced in generation as an early form of copy mechanism (Luong et al., 2015; ElSahar et al., 2018) to address the problem of rare words. They were widely adopted to steer language models towards specific features, such as aspects (Keskar et al., 2019) or structured outputs (Zellers et al., 2019).

In prior work, controlled language models rely on a predefined set of control tokens, collected manually (Keskar et al., 2019) or from dictionaries (Dathathri et al., 2020), which can lead to low domain coverage. Nabil et al. (2014) and ElSahar and El-Beltagy (2015) construct lexicons by exploiting the feature selection ability of sentiment classifiers, an approach that produces more relevant lexicons than classical topic models (e.g. LDA, Blei et al., 2003). In our work, we also rely on classifiers using the categories of reviews provided as meta-data. Without meta-data, we could have relied instead on unsupervised or weakly supervised aspect extractors (He et al., 2017; Angelidis and Lapata, 2018).

Hierarchical encoding. In order to allow a neural summarizer to read several sections, Cohan et al. (2018) proposes a hierarchical LSTM that works at two level. Similar to our proposal, Liu and Lapata (2019) extends a Transformer network to read several ranked paragraphs as input, avoiding a retrieve-then-read pipeline. In multi-document summarization, the paragraphs are not ranked but independent. This entails a significant change model-wise. We propose to encode each review independently (avoiding inter-paragraph self-attention) and only adapt the decoder-encoder attention.

3 Self-Supervision

In order to create our training dataset we assume that a review s_i for an entity (venue or product) can serve as a summary for a set of other similar reviews D_i. This simple intuition allows us to create training points (D_i, s_i) in a very similar way to what the model will experience at inference time. However, there are two issues with this approach. First, the potential set of training points is too large to be explored exhaustively. Given the set of all reviews $\mathcal{D}$ the total number of possible input-output pairs is $2^{|\mathcal{D}|-1} \times |\mathcal{D}|$. Second, the assumption that any review is fit to serve as a summary for any set of other reviews is obviously not true, and might yield a very noisy training dataset.

To solve the combinatorial explosion, we limit the size of D_i to k, and from a given s_i, we look for a set of k *good* reviews D_i, for which s_i serves as a good summary. Fixing k also simplifies training, and enables comparison with previous work where the number of input reviews is fixed (Chu and Liu, 2019; Bražinskas et al., 2020). Both s_i and all members of D_i are reviews of the same entity.

Having s_i fixed, we now search for reviews $d_1, \dots, d_k$ for which s_i is a relevant review:

$$\begin{aligned} \mathit{rel}(s_i) &= \{d_1, d_2, ..., d_k\}, \\ &= \underset{D_i \subset \mathcal{D} \setminus \{s_i\}, |D_i| = k}{\arg\max} \sum_{d_j \in D_i} \text{sim}(s_i, d_j) \end{aligned} \tag{1}$$

where *sim* is an arbitrary similarity function (that we define at the end of this section).

Fixing first the target summaries turns traditional approaches upside down. In particular, a recently proposed theoretical model of importance in summarization (Peyrard, 2019) defines the importance of a summary based on three aspects: (i) minimum redundancy, (ii) maximum relevance with the input document, and (iii) maximum informativeness. In that line of work D_i is considered fixed: redundancy and informativeness are not dependent on D_i and can therefore be ignored when s_i is fixed. In this setting Peyrard (2019) reduces then to Eq. 1

Then, we sort the data-points $(d_i, \mathit{rel}(d_i))$ according to the value of the relevance $(\sum_{d_j \in \mathrm{rel}(d_i)} \mathrm{sim}(d_i, d_j))$. Depending on the desired size of the target dataset, we keep the top-T pairs for training. Limiting T inherently increases informativeness, since it limits the creation of training examples where input and outputs are repetitive similar reviews that might be very prominent on corpora level (e.g. "Great restaurant."). This method is simple and fast, thanks to nearest neighbour search libraries (Pedregosa et al., 2011b). For all our experiments we defined *sim* to be the cosine similarity over a tf-idf bag-of-word representation (Ramos et al., 2003).

4 Controlling Hallucinations

Hallucinations are pieces of generated text that bear no relationship to the text they were conditioned on. They are likely to happen in our self-supervised setting, due to the noise from the construction of training instances. This might happen if the synthetically created training data contains contradictory signals, or because certain types of review are overly present (e.g. "great movie"). The model might default to those frequent patterns if it finds itself in a unfrequent state during decoding. To alleviate the problem of hallucinations, we propose to use *control tokens* that represent desired traits of the output text to steer the generated text towards more input-coherent summaries. These control tokens are inferred from each review, and used as prompts at inference time. We use two types of codes as follows:

1) Metadata control tokens. Those are special tokens that are associated with each input review, and are the capitalized control tokens in Fig. 1. We use two types of metadata that represent (i) the review **polarity**, a numerical value denoting the average sentiment score of the input reviews; (ii) **categorical tokens** representing the type of the entity of the review (e.g. Deli, Beauty&Spa, Furniture Stores). When meta-data labels are unavailable for all reviews (as in the Rotten-Tomatoes dataset), we infer control tokens with the same process, but using categories predicted by a classifier trained on labeled examples from the same domain.

2) Inferred control tokens. We follow recent work (Keskar et al., 2019; Dathathri et al., 2020) that shows that it is preferable to condition NLG models on control tokens that naturally co-occur in text. On one side, this allows for better control, and at the same it seems to be more robust when new (previously unseen) control codes are used. Here, we propose to use control codes that represent informative aspects (e.g. wine, service, ingredients) that occur in the input reviews text. However, instead of relying on manually created bag of control tokens for each desired attribute – which comes with obvious domain coverage limitations – we propose to infer those control codes from the text corpus.

To do so, we rely on the intrinsic feature selection capabilities of regularized linear classification models. For each category ℓ in the meta-data associated with each review we train a linear support vector machine (SVM) classifier (Vapnik and Lerner, 1963)[1] that learns to classify between reviews from this category and negative examples sampled randomly from the rest of the corpus. The features of the SVMs are parameterized by the weight vector $\theta_\ell \in \mathcal{R}^d$, where d is the number of features (in our experiments: all unigrams and bigrams present in the corpus). We used a squared hinge loss with $L1$ regularization over θ_ℓ – the latter to increase sparsity and force feature selection (Tibshirani, 1996; Ng, 2004). Finally, we trim the feature list into those who correspond to positive weights and renormalize the weights. The output of this step is a ranked list of n-grams that represent the distinctive aspects of each category.

When creating training data for summarization, we enrich each review with the top weighted n-grams of their corresponding categories as follows. For a given review d about entity p, we consider all m labels of p and use the weights of the corresponding classifiers $\theta_{\ell_i^{(p)}}$ (for each label $\ell_i^{(p)}$ of p). We only consider those n-grams actually occurring in d, and keep the top 8 such features. Note that these features could come from different classifiers, as we consider all m labels.

During training, we enrich each review with its tailored control codes. In particular, the reviews acting as summary also contain them, and by con-

---

[1] We use `liblinear` (Fan et al., 2008).

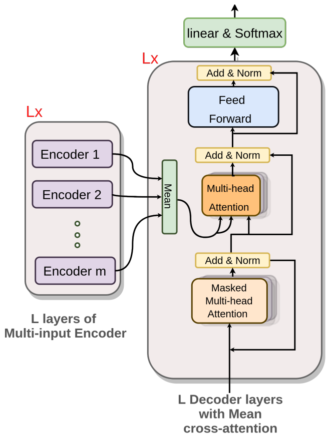


Figure 2: Our adaptation of the Transformer cross-attention to allow *Mean* combination of multi-sources.

struction those are n-grams present in the text. At inference time – when the target side and its control codes are not available – we select the most repeated control tokens from the input side and feed them as a prefix to the decoder. There is clearly a risk that the model just learns to copy the control codes it has seen somewhere in the text. We check whether this is the case in Sect. 7.

5 Multi-source Transformer Model

Previous work for multi-document summarization (Chu and Liu, 2019) built multi-source input representations through a simple mean over the last hidden states of the encoder. An intrinsic limitation of this method is that the full set of reviews is represented as a single vector. This aggregation might cause information distortion especially when some input reviews are expected to have conflicted opinions in between. Standard transformer models (Vaswani et al., 2017) consider only a single input to the decoder part of the model. Aggregating all input reviews into a single input (Junczys-Dowmunt, 2019) with special tokens to represent document boundaries might be slow and impractical due the $O(n^2)$ complexity of the self-attention mechanism. We therefore experiment with several input combination strategies of the transformer cross-attention (Libovický et al., 2018).

Parallel. At each cross-attention head, the decoder set of queries Q attend to each of the encoded inputs separately from which the set of keys ($K_i \in K_{1:m}$) and values ($V_i \in V_{1:m}$) are generated and then the yielded context is averaged and followed by a residual connection from the previous decoder layer (box C in Fig. 1).

$$A^h_{\text{parallel}}(Q, K_{1:m}, V_{1:m}) = \frac{1}{m} \sum_{i=1}^{m} A^h(Q, K_i, V_i).$$

Mean. We also propose a simpler and less computationally demanding input combination strategy. It does not apply the cross-attention with each encoder separately. Instead, the set of keys and values coming from each input encoder are aggregated using the average at each absolute position. Afterwards the decoder set of queries attend to this aggregated set of keys and values. This combination can be seen as a more efficient variation of the flat combination strategy (Libovický et al., 2018) with mean instead of concatenation. Fig. 2 depicts this strategy, which replaces box (C) in Fig. 1.

$$A^h_{\text{mean}}(Q, K_{1:m}, V_{1:m}) = A^h\left(Q, \frac{1}{|m|}\sum_{i=1}^{m} K_i, \frac{1}{|m|}\sum_{i=1}^{m} V_i\right)$$

In practice, we share the parameters across all encoders, this can be also seen as a single encoder used to encode each input document independently. We believe that this is an appropriate design choice as the order of the input document doesn't matter. Furthermore, this is necessary to allow variable number of input documents during different training batches or during inference. In Sect. 7, we compare both approaches through an ablation study, focusing on summary quality as well as empirical training times.

6 Experimental Setup

Experimental Details All our models are implemented with PyTorch (Paszke et al., 2019) and Fairseq (Ott et al., 2019) libraries, as well as scikit-learn (Pedregosa et al., 2011a) for the classifiers used either for inferring control tokens or for evaluation. For all our models we use sentence piece (Kudo and Richardson, 2018) as a tokenizer with a vocabulary size of 32 000. We use the same hyperparameters as the Transformer Big model described by Vaswani et al. (2017) ($d_{\text{model}} = 1024$, $n_{\text{heads}} = 16$, $n_{\text{layer}} = 6$, dropout $= 0.1$). We optimize them with a Nesterov accelerated SGD optimizer with a learning rate of 0.01. We train all models for a total of 80 000 steps across 25 epochs, with linear warm-up for the first 8 000 steps. We select the best model checkpoint based on perplexity on the validation set. All models were trained on one machine with 4 NVIDIA V100 GPUs, the

	Model	ROUGE-1	ROUGE-2	ROUGE-L	$\mathrm{F_{BERT}}$	Sentiment Acc.	$\mathrm{F_{category}}$
YELP	Textrank (Mihalcea and Tarau, 2004)	28.3	4.2	14.9	84.1	82.0	53.4
	Lexrank (Radev et al., 2004)	27.4	3.9	14.9	84.2	83.5	54.1
	Opinosis (Ganesan et al., 2010)	26.8	3.4	14.2	81.2	80.5	53.0
	H-VAE (Bražinskas et al., 2020)	29.5	5.3	18.1	–	–	–
	DenoiseSum (Amplayo and Lapata, 2020)	30.1	4.9	17.6	–	–	–
	MeanSum (Chu and Liu, 2019)	28.6	3.8	15.9	86.5	83.5	50.3
	Ours	**32.8**	**8.7**	**18.8**	**86.8**	**83.9**	**55.2**
RT	Textrank	19.0	4.3	19.4	**85.3**	**75.8**	41.6
	Lexrank	17.6	3.5	18.2	**85.3**	73.2	40.9
	Opinosis	15.2	2.9	16.9	84.1	67.5	37.1
	Ours	**20.9**	**4.5**	**22.7**	**85.3**	70.9	**43.6**
	DenoiseSum*	21.2	4.6	16.27	–	–	–

Table 1: Automatic evaluations results against gold summaries of Yelp and Rotten Tomatoes (RT) datasets. "Ours" denotes our proposed system with parallel input combination strategy and control codes. In the RT experiments, we report numbers from (Amplayo and Lapata, 2020) denoted as DenoiseSum* which are not comparable as they utilize different train/dev/test splits.

	Model	Dist-1	Dist-2	Dist-3	Dist_c-1	Dist_c-2	Dist_c-3
Extract.	Textrank	0.68	0.95	0.992	0.135	0.62	0.90
	Lextrank	0.70	0.96	0.994	0.144	0.6	0.92
	Opinosis	0.72	0.94	0.97	**0.159**	**0.66**	**0.92**
Abstr.	MeanSum	0.72	0.95	0.98	0.091	0.39	0.67
	Ours	**0.79**	**0.99**	**1.00**	0.097	0.41	0.64

Table 2: Referenceless evaluation results on Yelp dataset.

longest model took 50 hours to train. For inference, we use a beam size of 35. We discard hypotheses that contain twice the same trigram. We limit generation of each summary to a maximum budget of 150 tokens for each summary for Yelp, as was done by Chu and Liu (2019), and a budget of 50 tokens for Rotten Tomatoes. We set a similar budget for all other extractive baselines in the experiments. Finally, we use length normalization (Wu et al., 2016) with length penalty 1.2 to account for the model's bias towards shorter sequences.

Datasets We evaluate our proposal on two English datasets: Yelp[2] (Chu and Liu, 2019) and Rotten Tomatoes (Wang and Ling, 2016). The Yelp dataset contains reviews of businesses (around 1M reviews for 40k venues). As described in Sect. 3, for each venue, we select the best reviews to use as target summaries: either the top-p (with $p = 15\%$) or the top-T (with $T = 100$) reviews, whichever is smaller. For each selected target summary, we take its $k = 8$ most similar reviews (cosine) to form its input. We obtain around 340k training examples, representing 22.5k venues. The Rotten Tomatoes dataset was constructed by (Wang and Ling, 2016) from the movie review website `rottentomatoes.com`. We use the same process as for Yelp, but use $p = 1\%$ and $T = 150$. We construct around 170k training examples, representing 3.7k movies. We provide more details in the supplementary material.

Evaluation Metrics We evaluate summary systems with the classical ROUGE-F-{1,2,L} metrics (Lin, 2004).[3] We also report BERT-score (Zhang et al., 2020b), a metric that uses pre-trained BERT (Devlin et al., 2019) to compute the semantic similarity between a candidate summary and the gold summary. Dist-n and Dist_c-n ($n = 1, 2, 3$) scores (Li et al., 2016a) are the percentage of distinct n-grams in the generated text on the summary level or the corpora level respectively. Dist-n is an indicator of repetitiveness within a single summary while Dist_c-n indicates the diversity of different generations. Finally, as done by Chu and Liu (2019), we use a classifier to check whether the sentiment of the summary is consistent with that of input reviews (Sentiment Acc., Tab. 1).[4] We extend this method to check whether the correct product category can also be inferred from the summary, we report $\mathbf{F}_{\text{category}}$ the micro F-score of the multi-label category classifier.

Baselines and Other Systems We compare our system to three unsupervised baselines. TextRank (Mihalcea and Tarau, 2004) and LexRank (Radev et al., 2004) are extractive systems. Opinosis (Ganesan et al., 2010) is an abstractive graph-based system. We use openly available Python implementations for TextRank[5] (Barrios et al., 2015)

---

[2] `https://www.yelp.com/dataset/challenge`

[3] For Yelp we use Chu and Liu (2019)'s implementation to keep results comparable. For RottenTomatoes we use *py-rouge* package `pypi.python.org/pypi/pyrouge/0.1.3`

[4] We use 3 classes: negative (1, 2 star), neutral (3), positive (4, 5). Numbers are not comparable with Chu and Liu (2019).

[5] `https://github.com/summanlp/textrank`

Model	*Quality* ROUGE-1	ROUGE-2	ROUGE-L	F_{BERT}	Sentiment Acc.	F_{category}	*Speed* **Train.** (wps)
$\text{Ours}_{\text{Parallel}}$	**32.8**	**8.7**	**18.8**	86.8	**83.9**	55.2	3785
$\text{Ours}_{\text{Mean}}$	29.4	5.3	17.2	**87.6**	83.4	**56.2**	8075
$\text{Ours}_{\text{Parallel}}$ − cntrl.	25.3	3.7	15.5	85.2	76.9	43.9	7609
$\text{Ours}_{\text{Mean}}$ − cntrl.	27.5	5.3	17.1	87.3	80.0	52.1	**8714**

Table 3: Ablation study showing the effectiveness of parallel-cross attention and control tokens on Yelp dataset. "−cntrl." denotes models trained without the control step. "Train. (wps)" denotes the word per second rate at training time.

OURS:
This was my *first visit* to *Capriotti's* and I really enjoyed it . I had the *Capastrami* and my husband had *the Bobbie* . We both enjoyed our *sandwiches* as well . The *quality* of the *ingredients*, however, was not what we expected . We also enjoyed the *cheese steak* as well as the *turkey*, which was not bad at all . This place is a bit on the expensive side for what you get, but you get what you pay for . The *seating* is limited, so it's a good place to visit if you're in a hurry.

MeanSum:
Drove by here for the *first time*. I just went to the *deli* with a friend and it's a quick fix that is just about as good as it gets. But it's not an actual *sandwich*, but it's not as good as I remembered it, but they were great!! *Sandwich* was also very good, just a hint of cinnamon. I will be back for the other locations.

TextRank (Extractive):
Will not return This place is always good, I think the owner actually made my sandwich last time I was there , owner or manager, anyway it was superb! Ordered a sandwich, watched the guy write it down and 25 minutes later the same person asked what I wanted when I reminded him of my sandwich, he only said he can't remember where the order went. I watched 4 people come in after me order, one person the same sandwich just a different size then me get their food, pay and leave. At that point I gave up because as much as I like their sandwiches I am never going back.

Figure 3: Examples of different model generations to the same input set of documents. Green (italics) denotes substrings with exact match with the input, red (underlined) denotes statements without support in the input. TextRank is shown as a reference: all substrings are present in the input, but note the lack of cohesion.

and LexRank,[6] with their default parameters. For Opinosis, we use the official Java implementation, with default hyperparameters.[7] We also compare our systems with recent neural unsupervised summarization systems (Chu and Liu, 2019; Bražinskas et al., 2020). In addition, in our ablation study (next section) we also compare against a vanilla Transformer system, to capture the relative gains obtained on top of that model.

7 Evaluation Results

Automatic Evaluation Table 1 contains the automatic evaluation metrics with respect to reference summaries. The proposed multi-input self-supervised model with control codes perform consistently better in the Yelp dataset across the benchmarked models, including the recent neural unsupervised models of MeanSum and H-VAE. For the recent H-VAE model we report the numbers from their paper.[8] For MeanSum we re-run their provided checkpoint and run evaluation through the same pipeline. The BERTScore (Zhang et al., 2020b) differences are closer and seem to favour neural models.

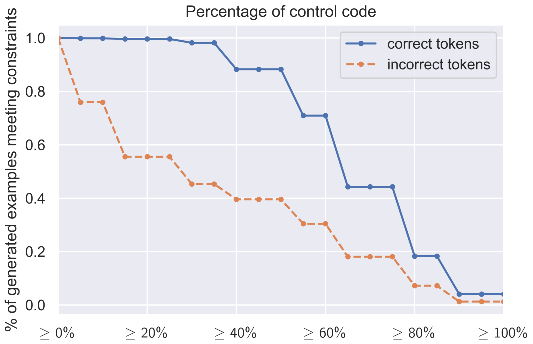


Figure 4: Proportion of control tokens fed as prompts that occur in the generated summary. When the model is fed control tokens that occur in the input reviews (*correct*) it tends to generate them in the output. Contrary to this, *incorrect* control tokens are mostly ignored.

With the RottenTomatoes dataset we only benchmarked the graph-based unsupervised methods, since the released pretrained MeanSum model does not cover the domain of movie reviews. We attribute the lower score in sentiment accuracy to the fact that the "summaries" in RottenTomatoes are critical reviews, written in a very different style than the original reviews.

Table 2 contains reference-less evaluation, analyzing the number of distinct n-grams (an indicator of repetitiveness) on the summary level and corpora level. On the summary level our model outperforms all the baselines: our model is capable of generating richer and less repetitive summaries. On the level of all generations our model generates

---

[6] `https://github.com/crabcamp/lexrank`

[7] Except for the redundancy parameter set to one, since the default led to many empty outputs. `https://github.com/kavgan/opinosis-summarization`

[8] While the ROUGE implementation might be different, the numbers of the common baselines are very close.

text with more diversity than MeanSum. In general however extractive models tend to have more diversity on the corpus level as they directly copy from each input separately, while abstractive models tend to learn repetitive patterns present in the training set.

Fig. 3 shows summaries generated by different models from the same input. We notice that our model learned to copy aspects of the input documents such as restaurant names "Capricotti's" and menu items "the Bobbie", possibly due to the cross-attention mechanism. We provide more examples as supplementary material.

Human Evaluation Existing natural language generation systems are known to generate very fluent language, that looks very natural to native speakers. On the other side, current neural models are known to generate factually incorrect data, something which was less of a concern in pre-neural methods but also much harder to detect. As mentioned by Kryscinski et al. (2019): "Neither of the methods explicitly examines the factual consistency of summaries, leaving this important dimension unchecked." Inspired by Falke et al. (2019) we decided to focus the human evaluation on those aspects of the summarization evaluation in which existing models risk failing the most, the one of *faithfulness*.

We annotated 94 summaries through a crowdsourcing platform, comparing 3 systems (Gold, MeanSum and ours). Workers were asked if "the summary contains correct information given the original reviews". In total we had 282 tasks (94×3) and each task was labeled by 3 annotators and paid $0.50 (defined by a pilot study to aim for $15 / hour) and restricted to experienced, English-speaking workers. A full description of the campaign, including the filtering of the annotations, is detailed in the supplementary material.

Faithfulness	**Gold**	**Ours**	**MeanSum**
Correct	67	47	43
Incorrect	3	7	16
%Correct	95.71	87.04	72.88

Table 4: Results of the human evaluation focused on faithfulness of generated reviews.

The results in Table 4 show that 87.0% of the generated summaries of our system are considered factually correct (compare with 95.7% for the gold summaries), as opposed to 72.9% of MeanSum.

Ablation We analyzed the impact of our proposed variations of the basic self-supervised setting in Table 3. Removing control codes degrades significantly sentiment and category classification of the produced summary F_1. It also impacts greatly the ROUGE score. Changing the decoder-encoder attention from parallel to mean (Sect. 5) also degrades ROUGE. The difference of this attention change without control codes is smaller but – surprisingly – in the different direction.

Control Codes The previous ablation study shows the importance of the control codes in the quality of the final summaries. In order to see how rigidly the model follows those control codes we devise the following experiment to see if the tokens used as control codes are forced to appear in the output text, independent of the input text.

For this, we sample 500 reviews (for 279 venues from the Yelp validation set). For each input example, we randomly sample 8 inferred control tokens (see Sect 4) from the tokens occurring in the review, referring to these as *correct control tokens*. We run the decoder using these control tokens as prompt and count the proportion of them that also occurs in the generated summary. For comparison, we repeat the same experiment but sampling instead 8 control tokens that do *not* occur in the input text, referring to these as *incorrect*.

To minimize the possibility of conditioning on control tokens that might show up naturally in the generated text, for both settings, we repeat the process 5 times per input example (resulting in 2500 with *correct control tokens* as prefix and 2500 using *incorrect*). We report in Fig. 4 the proportion of fed control codes that are generated by the model in both cases. We observe that the model tends to comply with the correct control tokens that occur in the input documents (eg: 89% of the summaries contain more than 50% of the control tokens), but tends to ignore the control tokens when they do not occur in the input. We illustrate this behaviour with a set of examples generated from the same input but different control tokens in the supplementary material.

8 Conclusion

The promise of unsupervised multi-document abstractive summarization is been hampered by the complexity of those models and the problem of hallucinations. Our proposed model has the advantage of being very simple to train compared to previous proposals. In addition, the combined use of control-codes to steer the generation and of multi-input transformers results in summaries that are better (as measured by automatic measures), and produce more faithful summaries (as measured by human evaluation).

While the generated reviews are more factual than those generated by other models, we want to stress that inaccuracies can still appear. Generated summaries are often conjugated in first person, which could lend to believe that an actual human wrote those. We recommend strongly that any use of such algorithms to be accompanied by a clear disclaimer on its true nature.

References

Reinald Kim Amplayo and Mirella Lapata. 2020. Unsupervised opinion summarization with noising and denoising. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1934–1945. Association for Computational Linguistics.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 936–945.

Stefanos Angelidis and Mirella Lapata. 2018. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3675–3686, Brussels, Belgium. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Federico Barrios, Federico López, Luis Argerich, and Rosita Wachenchauzer. 2015. Variations of the similarity function of textrank for automated summarization. In *Argentine Symposium on Artificial Intelligence (ASAI 2015)-JAIIO 44 (Rosario, 2015)*.

D.M. Blei, A.Y. Ng, and M.I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics.

Yun Chen, Victor O. K. Li, Kyunghyun Cho, and Samuel R. Bowman. 2018. A stable and effective learning strategy for trainable greedy decoding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 380–390.

Eric Chu and Peter J. Liu. 2019. MeanSum: A neural model for unsupervised multi-document abstractive summarization. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 1223–1232.

Maximin Coavoux, Hady Elsahar, and Matthias Gallé. 2019. Unsupervised aspect-based multi-document abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 42–47, Hong Kong, China. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Andrew Drozdov, Patrick Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. 2019. Unsupervised latent tree induction with deep inside-outside recursive auto-encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1129–1141, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady ElSahar and Samhaa R. El-Beltagy. 2015. Building large arabic multi-domain resources for sentiment analysis. In *Computational Linguistics and Intelligent Text Processing - 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part II*, pages 23–34.

Hady ElSahar, Christophe Gravier, and Frédérique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 218–228.

Günes Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479.

Tobias Falke, Leonardo FR Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Demian Gholipour Ghalandari. 2017. Revisiting the centroid-based method: A strong baseline for multi-document summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 85–90, Copenhagen, Denmark. Association for Computational Linguistics.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 388–397, Vancouver, Canada. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1535–1546.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation, WMT 2019, Florence, Italy, August 1-2, 2019 - Volume 2: Shared Task Papers, Day 1*, pages 225–233.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq R. Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. Gedi: Generative discriminator guided sequence generation. *CoRR*, abs/2009.06367.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016b. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1192–1202.

Jindrich Libovický, Jindrich Helcl, and David Marecek. 2018. Input combination strategies for multi-source transformer decoder. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 253–260.

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

Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. 2017. Improved image captioning via policy gradient optimization of spider. In *Proceedings of the IEEE international conference on computer vision*, pages 873–881.

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 11–19.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Mahmoud Nabil, Mohamed Aly, and Amir Atiya. 2014. Labr: A large scale arabic sentiment analysis benchmark. *arXiv preprint arXiv:1411.6718*.

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

Andrew Y Ng. 2004. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011a. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011b. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Maxime Peyrard. 2019. A Simple Theoretical Model of Importance for Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073, Florence, Italy. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1314–1324.

Dragomir R. Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40(6):919 – 938.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of* *the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. 2018. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4035–4045, Brussels, Belgium. Association for Computational Linguistics.

Gaetano Rossiello, Pierpaolo Basile, and Giovanni Semeraro. 2017. Centroid-based text summarization through compositionality of word embeddings. In *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, pages 12–21, Valencia, Spain. Association for Computational Linguistics.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1702–1723.

Ori Shapira and Ran Levy. 2020. Massive multi-document summarization of product reviews with weak supervision. *arXiv preprint arXiv:2007.11348*.

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

V Vapnik and A Lerner. 1963. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 47–57, San Diego, California. Association for Computational Linguistics.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. BottleSum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3750–3759, Hong Kong, China. Association for Computational Linguistics.

Lijun Wu, Yingce Xia, Fei Tian, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. Adversarial neural machine translation. In *Proceedings of The 10th Asian Conference on Machine Learning, ACML 2018, Beijing, China, November 14-16, 2018*, pages 534–549.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 7298–7309.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-symbolic VQA: disentangling reasoning from vision and language understanding. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 1039–1050.

Li Yuan, Francis EH Tay, Ping Li, Li Zhou, and Jiashi Feng. 2019. Cycle-sum: cycle-consistent adversarial lstm networks for unsupervised video summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9143–9150.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, pages 9051–9062.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*.

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

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

A Generated Examples

Figures 6, and 7 include a set of samples generated from our model and baselines, full generations our model can be downloaded from `https://www.dropbox.com/s/w6eqviy5fnda11f/hypos_and_refs.zip?dl=0`

B Control Codes Analysis

Figure 5 illustrates the behaviour of the model on the same input when it is fed either ‘correct’ or ‘incorrect’ control codes. The correct control codes occur in the output, but this is not the case for incorrect control codes: the model does not simply learn to copy control codes, it ignores them when they are inconsistent with the input reviews.

Correct Control Tokens: *eat*, *lentil*, *eggplant*, *new*, remember, *flavourful*, *friendly*
Seriously best we’ve had in Toronto. We were looking for a *new* place to *eat* and stumbled upon this place. The atmosphere is very authentic, the food is authentic, and the service is very *friendly* and attentive. We started with the *lentil* soup, which was very *flavourful* and full of flavor. For dinner, I had the lamb shank and my husband had the *eggplant* dish. Both were very good. We also had the baklava for dessert and it was amazing. We can’t wait to come back and try more of the menu

Correct Control Tokens: *yogurt*, *remember*, *service*, *shawarma*, *dishes*, *eat*, *restaurant*, *flavourful*
Seriously best I’ve had in Toronto. We were looking for a place to *eat* and this was the first *restaurant* we went to. The *service* was great and the *dishes* were authentic, *flavourful* and authentic. We had the *shawarma* sampler and the *yogurt* sampler. Both dishes were amazing. We also had the baklava, which was also very good. I can’t *remember* the name of the dish, but it was very authentic

Incorrect Control Tokens: open, tortellini, filter, details, rabbit, signature, craving, double
Seriously best I’ve had in Toronto. We were looking for a new place to try and stumbled upon Syrianian food. We came across this place on yelp and decided to give it a try. The food, the service, the atmosphere, the quality of the food, and the service were all amazing. We ordered the sampler platter for two and it was enough for two people to share. We also ordered the Brabianianian appetizer, which was one of the best we’ve ever had. For our main course, we ordered the Crabian Frabian Bianianca and the Bianca. Both were so authentic, and full of flavour. For dessert, we had the Turkish coffee which was also very good. The only thing I didn’t like was the bread they give you .

Incorrect Control Tokens: beer, balls, skip, mains, croissant, replacement, wish
Seriously best I’ve had in Toronto. We came here with a friend and we were looking for a place to eat. We found Syrianian food, which was very authentic. We were very impressed with the quality of the food, the service, and the atmosphere. We had the sampler platter, which came with two mains and two mains for dinner. We also had the bread pudding for dessert and it was to die for. I’m not a huge fan of sweets, but this was one of the best we’ve ever had. I wish we lived in Toronto so we could come here all the time. We’ll be back to try more of the menu .

Figure 5: Summaries generated **from the same input** when different ‘correct’ and ‘incorrect’ control tokens are fed as prefixes at inference time. Control tokens that occur in the summary are highlighted (green/italics for the first rows, red/underlined for the other two).

C Inferred Control Tokens

Fig. 8 shows examples of the top inferred tokens for some categories in the Yelp dataset, those tokens have been inferred using our proposed method in this work.

D Human Evaluation Campaign

We used Amazon Mechanical Turk to ask 3 “workers” to assess if 282 summaries produced by 3 systems (94 from each: ours, gold from human experts and MeanSum) aligned correctly with sets of 8 reviews. Workers had to read the reviews, the summary and answer the question: “does the summary contain correct information given in the original reviews?” Instructions specified to “assess the faithfulness of the summary with respect to the set of reviews,” specifically to “verify that the summary did not contain factually incorrect or self-contradicting statements that could not be inferred from what was provided in the original reviews.” Using Mechanical Turk qualification criteria, we asked for the workers: (1) to be located in the United States, Canada or United Kingdom; (2) to have a HIT approval rate higher than 98; (3) to have more than 1000 HITs approved.

Note that in an initial pilot, we asked evaluators to pick the best and worst summaries for fluency, coherency and alignment, as well as overall. We decided to simplify the task because it turned out to be a quite difficult one as workers struggled on many summaries to decide of the best and worst. We decided to focus the human evaluation on the aspect that is currently very difficult to automate, faithfulness to the original text. We see this evaluation as complementary to the automatic evaluation, focusing on different aspects.

We did an internal run to estimate the time needed per individual assignment – each Human Intelligence Task, or HIT, an annotation in our case, was assigned to 3 workers. We followed it by a short pilot to validate the average 2 minutes we had estimated. This is important to establish the rate to pay: 2 minutes translate into 30 potential assignments per hour, we picked $0.50 to target an average $15 hourly wage. Beyond the timing, the pilot was also used as a dry run for the full campaign. The average time to answer and the theoretical hourly wage are provided in Table 6

By using shuffled gold summaries, hence written for another set of reviews, we included 21 badly aligned “negatives.” Workers who answered *yes* for these obvious *no* were filtered out as “dubious” from the results: all their answers were dis-

Dataset	Reviews		Businesses / Movies
	Train	Valid	
Yelp — Ours	349,839	48,677	22,522
Yelp — Ours + Control	404,811	47,938	22,522
Rotten Tomatoes — Ours	167,168	18,731	3,732

Table 5: Sizes of Training and validation splits of different datasets.

carded. After filtering out the "negatives" HITs and the ones from "dubious" answers, we were left with 446 annotations, from the 782 we received. We further discarded all annotations made in less than a minute to keep 377 realistic answers —one minute may seem harsh but we estimated it was the minimum time needed to first read the reviews, then the summary, and to assess the latter, given the question: proceeding backward by first reading the summary would still require the worker to read all summaries, to make sure a factual error according to one review is not extracted from another one.

Finally we looked for full agreement at the HIT level and kept only the ones with either 0 *yes* or 0 *no*, with varying numbers, from 1 to 3, of the alternatives after the filtering of the "dubious" and "unrealistic" answers. Not surprisingly, as we focused on alignment, Gold summaries scored best but ours scored nicely, with a very low number of misaligned summaries.

Assessing the alignment of summaries to a set of reviews is not an easy task. We decided to discard all answers from the "dubious" workers who erred on our "negatives" summaries to be on the safe side. Mechanical Turk reports the time taken for an assignment, their averages is an interesting metric to look at, especially the way it evolves along our filterings — we translated it to the associated theoretical hourly wages, alas all under the $15 we initially targeted.

We also looked at the results with no full agreement: instead of doing it per HIT, or summary, it had to be done at the lower level of the evaluation. For the 276 evaluations with full agreement, the numbers are: Gold 108/4 (96.43% correct), Ours 69/7 (90.79%), MeanSum 63/25 (71.59%).
When including the disagreements (377 evaluations), they are: Gold 116/13 (89.92%), Ours 89/27 (76.72%), MeanSum 85/47 (64.39%).
The numbers are similar, however given the difficulty of the assessment for the workers, we decided to focus on the summaries they agreed on.

Inputs:

1. Best Philly Ever!!! Thank You Sam!!! Sometimes it is the little things in life that can Make You Happy- All it took was a Perfect Cheese Steak to Cheer Me Up, not to mention seeing a Friend Again - Thanks again Sam,, It wouldn't be the same without You

2. Wow after all the hype about what a great place I was really disappointed. If this is a franchised operation than the quality control is really lacking. Our first visit to Capriotti's and with so many other quality places I doubt if they will get us as repeat customers. Well, here it is. We ordered the Bobbie and the Capastrami shared it. Both had cold bread in fact we got the impression that both sandwiches had been pre made and put in a refrigerator because the insides were also cold. No taste at all in either. For a company that supposedly cooks overnight you would think the turkey ingredients would look like turkey but apparently they shred it into little tiny bits. Will not return

3. This place is always good, I think the owner actually made my sandwich last time I was there , owner or manager, anyway it was superb! quite flavorful, even the next day it tasted just as good. Grab a Capistrami you can't go wrong, until next time Cappie's , be well.

4. one New Year's resolution is to write more Yelp reviews, so here goes... In Vegas for NYE and gave this place a shot per other Yelp reviews. I had the Capistrami and the girlfriend had the Cheese Steak, which I had few bites of. Both were absolutely delicious in an awesome-deli-sandwich-sort-of-way. The shop is no-frills with only some bar seating, but the sandwiches are really reasonably priced. So if all you're after is a fantastic deli sandwich, definitely go.

5. number 1 in Vegas for a reason. Everyone has their favs.... the capistrami, the cheese steak with mush... Mine is definitely the Bobby. In case you haven't viewed their menu yet, the bobby is thanksgiving leftovers in a huge sandwich... yeah, exactly.

6. Worst service I have seen at a capriotti's. Ordered a sandwich, watched the guy write it down and 25 minutes later the same person asked what I wanted when I reminded him of my sandwich, he only said he can't remember where the order went. I watched 4 people come in after me order, one person the same sandwich just a different size then me get their food, pay and leave. I will not be coming back to the location ever again. Looks like I will be going to firehouse for now on!

7. Stopped in for a sandwich on the way to the park. Next day I notice the charge has had a $2 tip added to it that I did not authorize. (I left a cash tip in the beer money jar) I called Corporate and got nowhere because this is a franchise store. At that point I gave up because as much as I like their sandwiches I am never going back.

8. Don't bother calling in an order. If they tell you a time it will be off by at least thirty minutes. Terrible service. Great food tho.

Summary OURS:

This was my *first visit* to *Capriotti's* and I really enjoyed it . I had the *Capastrami* and my husband had *the Bobbie* . We both enjoyed our *sandwiches* as well . The *quality* of the *ingredients*, however, was not what we expected . We also enjoyed the *cheese steak* as well as the *turkey*, which was not bad at all . This place is a bit on the expensive side for what you get, but you get what you pay for . The *seating* is limited, so it's a good place to visit if you're in a hurry.

Summary MeanSum:

Drove by here for the *first time*. I just went to the *deli* with a friend and it's a quick fix that is just about as good as it gets. But it's not an actual *sandwich*, but it's not as good as I remembered it, but they were great!! *Sandwich* was also very good, just a hint of cinnamon. I will be back for the other locations.

Summary TextRank (Extractive):

Will not return This place is always good, I think the owner actually made my sandwich last time I was there , owner or manager, anyway it was superb! Ordered a sandwich, watched the guy write it down and 25 minutes later the same person asked what I wanted when I reminded him of my sandwich, he only said he can't remember where the order went. I watched 4 people come in after me order, one person the same sandwich just a different size then me get their food, pay and leave. At that point I gave up because as much as I like their sandwiches I am never going back.

Figure 6: Examples of output summaries for different models.

Set	**Unfiltered**	**Negatives discarded**	**Dubious discarded**	**less than 1min discarded**	**Agreement**
Average time to Answer	2min16s	2min17	2min9	2min26	2min26
Theoretical hourly wage	13.22	13.16	13.96	12.36	12.35

Table 6: Average time to Answer and the theoritical hourly wage of turkers (in USD) for the crowdsourcing experiments of human evaluation.

Inputs:
1. Great service and a super clean nice location here. Considering this is in a busy airport, I was impressed. The pricing here, which is about double to triple regular prices, was what knocks off a star for me.
2. If you’re by the D gates at Sky Harbor, this is your coffee stop. Much better than the Starsucks at the high C gates.
3. Spotted in due to flight delay. The big comfy brown lawyer seats is what attracted me in. I figured I could get some work done. I ordered a non fat vanilla iced tea. It was pretty good. I noticed the prices were a lil bit more expensive.
4. $9 for a sandwich. I guess you can charge whatever you want when the airport doesn’t have any other options in concourse C
5. The line is quick, the people are friendly and the drinks are tasty. Also for skyharbor employees, they actually give an airport discount, unlike Starbucks.
6. Try gingerbread latte yum. This is the best looking, most comfortable airport coffee shop I’ve ever been in !!! Big comfy chairs with little tables. A big water container with cups in the restaurant away from the congestion of the order counter. Friendly , happy workers equals happy customers. I know you can’t please everyone, but at 5am a room full of happy airline travelers is a hard thing to come by. If your in need of coffee in Phoenix Sky Harbor , terminal 3 be sure to stop by and take a load off !!
7. The lid fell off my cup, burned my hand, and spilled half of my coffee. Employees never asked if I was okay, or offered to replace my coffee. Will not be back to this location.
8. Delicious cup of coffee. Very impressed Mr. Peets. Will be returning whenever I can.

OURS:
Try the *gingerbread* cups. The coffee and ginger cups are *delicious* and the *chairs are comfortable* . I’ve been working in the *terminal* for a long time . This is a must stop if you’re in the *airport* . The staff is *friendly* .

MeanSum:
5.50 for a 2.5” breakfast sandwich. I’m a big fan of the concept but this place is way better than *Starbucks*. The staff is *friendly*, and fast. I’m not a big fan of sweets but I’d be happy to come back.

TextRank (Extractive):
Great service and a super clean nice location here. Considering this is in a busy airport, I was impressed. The pricing here, which is about double to triple regular prices, was what knocks off a star for me. I noticed the prices were a lil bit more expensive. I guess you can charge whatever you want when the airport doesn’t have any other options in concourse C The line is quick, the people are friendly and the drinks are tasty. This is the best looking, most comfortable airport coffee shop I’ve ever been in !!! Will not be back to this location.

Figure 7: Examples of output summaries for different models.

Delis: deli, sandwiches, sandwich, bagels, skinnyfats, subs, bagel, sub, chompie, smoked meat
Nail Salons: nails, pedicure, nail, pedicures, pedi, salon, manicure, pedis, colors, salons
Sushi Bars: sushi, hibachi, kona, rolls, roll, japanese, ayce, sake, benihana, poke
Florists: flowers, trader, arrangement, florist, wedding, bouquet, tj, arrangements, aj, grocery
Beauty & Spas: walgreens, tattoo, sephora, ti, vdara, tattoos, haircut, barbers, barber
Party & Event Planning: herb box, wedding, kids, fun, party, event, golf, painting, rainforest, blast
Trainers: gym, workout, fitness, equipment, membership, trainers, training, trainer, instructors, machines
Cafes: cafe, first watch, bouchon, salsa bar, café, coffee, breakfast, gallo, crepes, latte
Mags: books, store, book, games, bookstore, selection, records, comics, vinyl, game
Ice Cream & Frozen Yogurt: gelato, ice, sonic, yogurt, custard, culver, flavors, freddy, froyo, icecream
Burgers: burgers, burger, mcdonald, ihop, applebee, red robin, mcdonalds, wellington, hamburgers, castle
Furniture Stores: furniture, ikea, mattress, store, sales, delivery, bought, couch, purchase, bed
Sporting Goods: bike, bikes, shoes, gear, gun, store, range, golf, shop, equipment
Bakeries: bakery, pastries, wildflower, cupcakes, panera, cake, pastry, cookies, cinnamon rolls, cakes
Thai: thai, curry, pad, asian, khao, curries, food, papaya, satay, tom
Gyms: gym, workout, fitness, membership, equipment, trainer, trainers, work out, coaches, class
Cosmetics & Beauty Supply: walgreens, pharmacy, products, haircut, store, sephora, hair, makeup, lashes, kohl
Auto Repair: car, vehicle, dealership, cars, auto, mechanic, vehicles, oil, windshield, tire
Department Stores: walmart, target, costco, store, department, shopping, mall, section, ross, sears
Local Services: post office, thrift, laundromat, daycare, guitar, cleaners, pest, activities, storage, laundry
Hair Extensions: hair, salon, stylist, color, haircut, extensions, appointment, she, lashes, blow
Hair Removal: eyebrows, nails, pedi, pedicure, nail, appointment, brows, wax, salon, waxing
Laundry Services: cleaners, clothes, laundry, cleaning, dry, laundromat, dress, pants, machines, shirts
Doctors: dr, doctor, doctors, medical, hospital, office, patients, appointment, nurse, clinic
Movers: move, moving, movers, company, truck, storage, guys, furniture, moved, haul
Printing Services: printing, print, ups, package, business, fedex, shipping, customer, printed, store
Makeup Artists: makeup, hair, salon, make up, lashes, stylist, blow, appointment, eyebrows, brows
Plumbing: plumbing, company, plumber, water, call, called, work, house, job, leak
Real Estate Services: property, estate, westgate, home, company, process, house, realtor, rent, work with

Figure 8: Examples of Inferred control tokens for each category of venues for the Yelp dataset.

NewsMTSC: A Dataset for (Multi-)Target-dependent Sentiment Classification in Political News Articles

Felix Hamborg[1,2] **Karsten Donnay**[3,2]

[1]Dept. of Computer and Information Science, University of Konstanz, Germany
[2]Heidelberg Academy of Sciences and Humanities, Germany
[3]Dept. of Political Science, University of Zurich, Switzerland
felix.hamborg@uni-konstanz.de donnay@ipz.uzh.ch

Abstract

Previous research on target-dependent sentiment classification (TSC) has mostly focused on reviews, social media, and other domains where authors tend to express sentiment explicitly. In this paper, we investigate TSC in news articles, a much less researched TSC domain despite the importance of news as an essential information source in individual and societal decision making. We introduce NewsMTSC, a high-quality dataset for TSC on news articles with key differences compared to established TSC datasets, including, for example, different means to express sentiment, longer texts, and a second test-set to measure the influence of multi-target sentences. We also propose a model that uses a BiGRU to interact with multiple embeddings, e.g., from a language model and external knowledge sources. The proposed model improves the performance of the prior state-of-the-art from $F1_m = 81.7$ to 83.1 (real-world sentiment distribution) and from $F1_m = 81.2$ to 82.5 (multi-target sentences).

1 Introduction

Previous research on target-dependent sentiment classification (TSC, also called aspect-term sentiment classification) and related tasks, e.g., aspect-based sentiment classification (ABSC, or aspect-category sentiment classification) and stance detection, has focused mostly on domains in which authors tend to express their opinions explicitly, such as reviews and social media (Pontiki et al., 2015; Nakov et al., 2016; Rosenthal et al., 2017; Zhang et al., 2020; Hosseinia et al., 2020; AlDayel and Magdy, 2020; Liu, 2012; Zhang et al., 2018).

We investigate TSC in news articles – a much less researched domain despite its critical relevance, especially in times of "fake news," echo chambers, and news ownership centralization (Hamborg et al., 2019). How persons are portrayed in news on political topics is very relevant, e.g., for individual and societal opinion formation (Bernhardt et al., 2008).

We define our problem statement as follows: we seek to detect polar judgments towards target persons (Steinberger et al., 2017). Following the TSC literature, we include only in-text, specifically in-sentence, means to express sentiment. In news texts such means are, e.g., word choice and generally framing[1] (Kahneman and Tversky, 1984; Entman, 2007), e.g., "freedom fighters" vs. "terrorists," or describing actions performed by the target, and indirect sentiment through quoting another person (Steinberger et al., 2017). Other means may also alter the perception of persons and topics in the news, but are not in the scope of the task (Balahur et al., 2010), e.g., because they are not on sentence-level. For example, story selection, source selection, article's placement and size (Hamborg et al., 2019), and epistemological bias (Recasens et al., 2013).

The main contributions of this paper are: (1) We introduce *NewsMTSC*, a large, manually annotated dataset for TSC in political news articles. We analyze the quality and characteristics of the dataset using an on-site, expert annotation. Because of its fundamentally different characteristics compared to previous TSC datasets, e.g., as to sentiment expressions and text lengths, NewsMTSC represents a challenging novel dataset for the TSC task. (2) We propose a neural model that improves TSC performance compared to prior state-of-the-art models. Additionally, our model yields competitive performance on established TSC datasets. (3) We perform an extensive evaluation and ablation study of the proposed model. Among others, we investigate the recently claimed "degeneration" of TSC to sequence-level classification (Jiang et al., 2019)

---

[1] Political frames determine what a causal agent does with which benefit and cost (Entman, 2007). They are defined on a higher level than semantic frames (Fillmore and Baker, 2001).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1663–1675
April 19 - 23, 2021.

finding a performance drop in all models when comparing single- and multi-target sentences.

In a previous short-paper, we explored the characteristics of how sentiment is expressed in news articles by creating and analyzing a small-scale TSC dataset (Hamborg et al., 2021). The paper at hand addresses our former exploratory work's critical findings, including essential improvements to the dataset. Key differences and improvements are as follows. We significantly increase the dataset's size and the number of annotators per example and address its class imbalance. Further, we devise annotation instructions specifically created to capture a broad spectrum of sentiment expressions specific to news articles. In contrast, the early dataset misses the more implicit sentiment expressions commonly used by news authors (Hamborg et al., 2021; Steinberger et al., 2017). Also, we comprehensively test various consolidation strategies and conduct an expert annotation to validate the dataset.

We provide the dataset and code to reproduce our experiments at:
`https://github.com/fhamborg/NewsMTSC`

2 Related Work

Analogously to other NLP tasks, the TSC task has recently seen a significant performance leap due to the rise of language models (Devlin et al., 2019). Pre-BERT approaches yield up to $F1_m = 63.3$ on the SemEval 2014 Twitter set (Kiritchenko et al., 2014). They employ traditional machine learning combining hand-crafted sentiment dictionaries, such as SentiWordNet (Baccianella et al., 2010), and other linguistic features (Biber and Finegan, 1989). On the same dataset, vanilla BERT (also called BERT-SPC) yields 73.6 (Devlin et al., 2019; Zeng et al., 2019). Specialized downstream architectures improve performance further, e.g., LCF-BERT yields 75.8 (Zeng et al., 2019).

The vast majority of recently proposed TSC approaches employ BERT and focus on devising specialized down-stream architectures (Sun et al., 2019a; Zeng et al., 2019; Song et al., 2019). More recently, to improve performance further, additional measures have been proposed. For example, domain adaption of BERT, i.e., domain-specific language model finetuning prior to the TSC finetuning (Rietzler et al., 2019; Du et al., 2020); use of external knowledge, such as sentiment or emotion dictionaries (Hosseinia et al., 2020; Zhang et al., 2020), rule-based sentiment systems (Hosseinia et al., 2020), and knowledge graphs (Ghosal et al., 2020); use of all mentions of a target and/or related targets in a document (Chen et al., 2020); and explicit encoding of syntactic information (Phan and Ogunbona, 2020; Yin et al., 2020).

To train and evaluate recent TSC approaches, three datasets are commonly used: Twitter (Nakov et al., 2013, 2016; Rosenthal et al., 2017), Laptop and Restaurant (Pontiki et al., 2014, 2015). These and other TSC datasets (Pang and Lee, 2005) suffer from at least one of the following shortcomings. First, implicitly or indirectly expressed sentiment is rare in them. In their domains, e.g., social media and reviews, typically authors explicitly express their sentiment regarding a target (Zhang et al., 2018). Second, they largely neglect that a text may contain coreferential mentions of the target or mentions of different concepts (with potentially different polarities), respectively (Jiang et al., 2019).

Texts in news articles differ from reviews and social media in that news authors typically do not express sentiment toward a target explicitly (exceptions include opinion pieces and columns). Instead, journalists implicitly or indirectly express sentiment (Section 1) because language in news is typically expected to be neutral and journalists to be objective (Balahur et al., 2010; Godbole et al., 2007; Hamborg et al., 2019).

Our problem statement (Section 1) is largely identical to prior news TSC literature (Steinberger et al., 2017; Balahur et al., 2010) with key differences: we do not generally discard the "author-" and "reader-level." Doing so would neglect large parts of sentiment expressions. Thus, it would degrade real-world performance of the resulting dataset and models trained on it. For example, word choice (listed as "author-level" and discarded from their problem statement) is in our view an in-text means that may in fact strongly influence how readers perceive a target, e.g., "freedom fighters" or "terrorists." While we do not exclude their "reader-level," we do seek to exclude polarizing or contentious cases, where no uniform answer can be found in a set of randomly selected readers (Sections 3.3 and 3.4). As a consequence, we generally do not distinguish between the three levels of sentiment ("author," "reader," and "text") in this paper.

Previous news TSC approaches mostly employ sentiment dictionaries, e.g., created manually (Balahur et al., 2010; Steinberger et al., 2017) or extended semi-automatically (Godbole et al., 2007), but yield poor or even “useless” (Steinberger et al., 2017) performances. To our knowledge, there exist two datasets for evaluation of news TSC methods (Steinberger et al., 2017), which – perhaps due to its small size ($N = 1274$) – has not been used or tested in recent TSC literature. Recently, Hamborg et al. (2021) proposed a dataset ($N = 3002$) used to explore target-dependent sentiment in news articles. The dataset suffers from various shortcomings, particularly its small size, class imbalance, and lacking the more ambiguous and implicit types of sentiment expressions described above. Another dataset contains quotes extracted from news articles, since quotes more likely contain explicit sentiment ($N = 1592$) (Balahur et al., 2010).

3 NewsMTSC: Dataset Creation

In creating the dataset, we rely on best practices reported in literature on the creation of datasets for NLP (Pustejovsky and Stubbs, 2012), especially for the TSC task (Rosenthal et al., 2017). Compared to previous TSC datasets though, the nature of sentiment in news articles requires key changes, especially in the annotation instructions and consolidation of answers (Steinberger et al., 2017).

3.1 Data sources

We use two datasets as sources: POLUSA (Gebhard and Hamborg, 2020) and Bias Flipper 2018 (BF18) (Chen et al., 2018). Both satisfy five criteria that are important to our problem. First, they contain news articles reporting on political topics. Second, they approximately match the online media landscape as perceived by an average US news consumer.[2] Third, they have a high diversity in topics due to the number of articles contained and time frames covered (POLUSA: 0.9M articles published between Jan. 2017 and Aug. 2019, BF18: 6447 articles associated to 2781 events). Fourth, they feature high diversity in writing styles because they contain articles from across the political spectrum, including left- and right-wing outlets. Fifth, we find that they contain only few minor content errors albeit being created through scraping or crawling.

3.2 Creation of examples

To create a batch of examples for annotation, we devise a three tasks process: first, we extract example candidates from randomly selected articles. Second, we discard non-optimal candidates. Only for the train set, third, we filter candidates to address class imbalance. We repeatedly execute these tasks so that each batch yields 500 examples for annotation, contributed equally by both sources.

First, we randomly select articles from the two sources. Since both are at least very approximately uniformly distributed over time (Gebhard and Hamborg, 2020; Chen et al., 2018), randomly drawing articles will yield sufficiently high diversity in both writings styles and reported topics (Section 3.1). To extract from an article examples that contain meaningful target mentions, we employ coreference resolution (CR).[3] We iterate all resulting coreference clusters of the given article and create a single example for each mention and its enclosing sentence.

Extraction of mentions of named entities (NEs) is the commonly employed method to create examples in previous TSC datasets (Rosenthal et al., 2017; Nakov et al., 2016, 2013; Steinberger et al., 2017). We do not use it since we find it would miss $\gtrapprox 30\%$ mentions of relevant target candidates, e.g., pronominal or near-identity mentions.

Second, we perform a two level filtering to improve quality and “substance” of candidates. On coreference cluster level, we discard a cluster c in a document d if $|M_c| \leq 0.2|S_d|$, where $|...|$ is the number of mentions of a cluster (M_c) and sentences in a document (S_d). Also, we discard non-persons clusters, i.e., if $\exists m \in M_c : t(m) \notin \{-, P\}$, where $t(m)$ yields the NE type[4] of m, and $-$ and P represent the unknown and person type, respectively. On example level, we discard short and similar examples e, i.e., if $|s_e| < 50$ or if $\exists \hat{e} : sim(s_e, s_{\hat{e}}) > 0.6 \land m_e = m_{\hat{e}} \land t_e = t_{\hat{e}}$ where s_e, m_e, and t_e are the sentence of e, its mention, and the target's cluster, respectively, and $sim(...)$ the cosine similarity. Lastly, if a cluster has multiple mentions in a sentence, we try to select the most meaningful example. In short, we prefer the cluster's representative mention[5] over nominal mentions, and those over all other instances.

Third, for only the train set, we filter candidates to address class imbalance. Specifically, we discard examples e that are likely the majority class ($p(\text{neutral}|s_e) > 0.95$) as determined by a simple binary classifier (Sanh et al., 2019). Whenever annotated and consolidated examples are added to the

---

[2] POLUSA by design (Gebhard and Hamborg, 2020) and BF18 was crawled from a news aggregator on a daily basis.

[3] We employ spaCy 2.1 and neuralcoref 4.0.

[4] Determined by spaCy.

[5] Determined by neuralcoref.

train set of NewsMTSC, we retrain the classifier on them and all previous examples in the train set.

3.3 Annotation

Instructions used in popular TSC datasets plainly ask annotators to rate the sentiment of a text toward a target (Rosenthal et al., 2017; Pontiki et al., 2015). For news texts, we find that doing so yields two issues (Balahur et al., 2010): low inter-annotator reliability (IAR) and low suitability. Low suitability refers to examples where annotators' answers can be consolidated but the resulting majority answer is incorrect as to the task. For example, instructions from prior TSC datasets often yield low suitability for polarizing targets, independently of the sentence they are mentioned in. Figure 2 (Appendix) depicts our final annotation instructions.

In an interactive process with multiple test annotations (six on-site and eight on Amazon Mechanical Turk, MTurk), we test various measures to address the two issues. We find that asking annotators to think from the perspective of the sentence's author strongly facilitates that annotators overcome their personal attitude. Further, we find that we can effectively draw annotators' attention not only at the event and other "facts" described in sentence (the "what") but also at word choice ("how" it is described) by exemplarily mentioning both factors and abstracting these factors as the author's holistic "attitude." [6] We further improve IAR and suitability, e.g., by explicitly instructing annotators to rate sentiment only regarding the target but not other aspects, such as the reported event.

While most TSC dataset creation procedures use 3- or 5-point Likert scales (Nakov et al., 2013, 2016; Rosenthal et al., 2017; Pontiki et al., 2014, 2015; Balahur et al., 2010; Steinberger et al., 2017), we use a 7-point scale to encourage rating also only slightly positive or negative examples as such.

Technically, we closely follow previous literature on TSC datasets (Pontiki et al., 2015; Rosenthal et al., 2017). We conduct the annotation of our examples on MTurk. Each example is shown to five randomly selected crowdworkers. To participate in our annotation, crowdworkers must have the "Master" qualification, i.e., have a record of successfully completed, high quality work on MTurk. To ensure quality, we implement a set of objective measures and tests (Kim et al., 2012). While we pay all crowdworkers always (USD 0.07 per assignment), we discard all of a crowdworker's answers if at least one of the following conditions is met. A crowdworker (a) was not shown any test question or answered at least one incorrectly[7], (b) provided answers to invisible fields in the HTML form (0.3% of crowdworkers did so, supposedly bots), or (c) the average duration of time spent on the assignments was extremely low ($< 4s$).

The IAR is sufficiently high ($\kappa_C = 0.74$) when considering only examples in NewsMTSC. The expected mixed quality of crowdsourced work becomes apparent when considering all examples, including those that could not be consolidated and answers of those crowdworkers who did not pass our quality checks ($\kappa_C = 0.50$).

3.4 Consolidation

We consolidate the answers of each example to a majority answer by employing a restrictive strategy. Specifically, we consolidate the set of five answers A to the single-label 3-class polarity $p \in \{\text{pos.}, \text{neu.}, \text{neg.}\}$ if $\exists C \subseteq A : |C| \geq 4 \wedge \forall c \in C : s(c) = p$, where $s(c)$ yields the 3-class polarity of an individual 7-class answer c, i.e., neutral ⇒ neutral, any positive (from slightly to strongly) ⇒ positive, and respectively for negative. If there is no such consolidation set C, A cannot be consolidated and the example is discarded. Consolidating to 3-class polarity allows for direct comparison to established TSC dataset.

While the strategy is restrictive (only 50.6% of all examples are consolidated this way), we find it yields the highest quality. We quantify the dataset's quality by comparing the dataset to an expert annotation (Section 3.6) and by training and testing models on dataset variants with different consolidations. Compared to consolidations employed for previous TSC datasets, quality is improved significantly on our examples, e.g., our strategy yields $F1_m = 86.4$ when comparing to experts' annotations and models trained on the resulting set yield up to $F1_m = 83.1$ whereas the two-step majority strategy employed for the Twitter 2016 set (Nakov et al., 2016) yields 50.6 and 53.4 respectively.

3.5 Splits and multi-target examples

NewsMTSC consists of three sets as depicted in Table 1. For the *train* set, we employ class balanc-

---

[6] To think from the author's perspective is not to be confused with the "author-level" defined by Balahur et al. (2010).

[7] Prior to submitting a batch of examples to MTurk, we add 6% test examples with unambiguous sentiment, e.g., "Mr. Smith is a bad guy."

Set	Total	Pos.	Neu.	Neg.	MT-a	MT-d	+Corefs	Pos.	Neu.	Neg.
Train	8739	2395	3028	3316	972	341	11880	3434	3744	4702
Test-mt	1476	246	748	482	721	294	1883	333	910	640
Test-rw	1146	361	587	624	73	30	1572	361	587	624

Table 1: Statistics of NewsMTSC. Columns (f.l.t.r.): name; count of targets with any, positive, neutral, and negative sentiment, respectively; count of examples with multiple targets of any and different polarity, respectively; count of targets and their coreferential mentions with any, pos., neu. and neg. sentiment, respectively.

ing prior to annotation (Section 3.2). To minimize dataset shift, which might yield a skewed sentiment distribution in the dataset compared to the real-world (Quionero-Candela et al., 2009), we do not use class balancing for either of the two test sets. Sentences can have multiple targets (MT) with potentially different polarities. We call this *MT property*. To investigate the effect on TSC performance of considering or neglecting the MT property (Jiang et al., 2019), we devise a test set named *test-mt*, which consists only of examples that have at least two semantically different targets, i.e., each belonging to a separate coreference cluster (Section 3.2). Since the additional filtering required for *test-mt* naturally yields dataset shift, we create a second test set named *test-rw*, which omits the MT filtering and is thus designed to be as close as possible to the real-world distribution of sentiment. We seek to provide a sentiment score for each person in each sentence in *train* and *test-rw* but mentions may be missing, e.g., because of erroneous coreference resolution or crowdworkers' answers could not be consolidated.

3.6 Quality and characteristics

We conduct an expert annotation of a random subset of 360 examples used during the creation of NewsMTSC with five international graduate students (studying Political or Communication Science at the University of Zurich, Switzerland, 3 female, 2 male, aged between 23 and 29). Key differences compared to the MTurk annotation are: first, extensive training until high IAR is reached (considering all examples: $\kappa_C = 0.72$, only consolidated: $\kappa_C = 0.93$). We conduct five iterations, each consisting of individual annotations by the students, quantitative and qualitative review, adaption of instructions, and individual and group discussions. Second, comprehensive instructions (4 pages). Third, no time pressure, since the students are paid per hour (crowdworkers per assignment).

When comparing the expert annotation with our dataset, we find that NewsMTSC is of high quality ($F1_m = 86.4$). The quality of unfiltered answers from MTurk is, as expected, much lower (50.1).

What is contained in NewsMTSC? In a random set of 50 consolidated examples from MTurk, we find that most frequent, non-mutually exclusive means to express a polar statement (62% of the 50) are usage of quotes (in total, direct, and indirect 42%, 28%, and 14%, respectively), target being subject to action (24%), evaluative expression by the author or an opinion holder mentioned outside of the sentence (18%), target performing an action (16%), and loaded language or connotated terms (14%). Direct quotes often contain evaluative expressions or connotated terms, indirect quotes less. Neutral examples (38% of the 50) contain mostly objective storytelling about neutral events (16%) or variants of "[target] said that ..." (8%).

What is not contained in NewsMTSC? We qualitatively review all examples where individual answers could not be consolidated to identify potential causes why annotators do not agree. The predominant reason is technical, i.e., the restrictiveness of the consolidation (MTurk compared to experts: $26\% \approx 30\%$). Other examples lack apparent causes ($24\% \gg 8\%$). Further potential causes are (not mutually exclusive): ambiguous sentence ($16\% \approx 18\%$), sentence contains positive and negative parts ($8\% \approx 6\%$), opinion holder is target ($6\% \approx 8\%$), e.g., "[...] Bauman asked supporters to 'push back' against what he called a targeted campaign to spread false rumors about him online."

What are qualitative differences in the annotations by crowdworkers and experts? We review all 63 cases (18%) where answers from MTurk could be consolidated but differ to experts' answers. The major reason for disagreement is the restrictiveness of the consolidation (53 cases have no consolidation among the experts). In 10 cases the consolidated answers differ. We find that in few examples (2-3%) crowdsourced annotations are superficial and fail to interpret the full sentence correctly.

Texts in NewsMTSC are much longer than in prior TSC datasets (mean over all examples): 152 characters compared to 100, 96, and 90 in Twitter, Restaurant, and Laptops, respectively.

4 Methodology

The goal of TSC is to find a target's polarity $y \in \{\text{pos.}, \text{neu.}, \text{neg.}\}$ in a sentence. Our model consists of four key components (Figure 1): a pre-trained language model (LM), a representation of external knowledge sources (EKS), a target mention mask, and a bidirectional GRU (BiGRU) (Cho et al., 2014). We adapt our model from Hosseinia et al. (2020) and change the design as follows: we employ a target mask (which they did not) and use multiple EKS simultaneously (instead of one). Further, we use a different set of EKS (Section 5) and do not exclude the LM's parameters from fine-tuning.

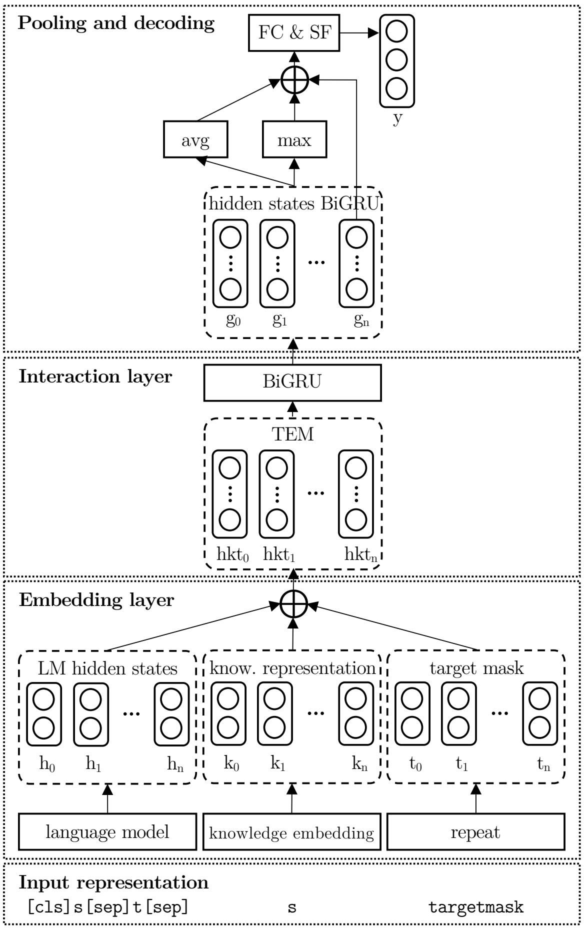


Figure 1: Overall architecture of the proposed model.

4.1 Input representation

We construct three model inputs. The first is a text input T constructed as suggested by Devlin et al. (2019) for question-answering (QA) tasks. Specifically, we concatenate the sentence and target mention and tokenize the two segments using the LM's tokenizer and vocabulary, e.g., WordPiece for BERT (Wu et al., 2016).[8] This step results in a text input sequence $T = [\text{CLS}, s_0, s_1, ..., s_p, \text{SEP}, t_0, t_1, ..., t_q, \text{SEP}] \in \mathbb{N}^n$ consisting of n word pieces, where n is the manually defined maximum sequence length.

The second input is a feature representation of the sentence, which we create using one or more EKS, such as dictionaries (Hosseinia et al., 2020; Zhang et al., 2020). Given an EKS with d dimensions, we construct an EKS representation $E \in \mathbb{R}^{n \times d}$ of S, where each vector $e_{i \in \{0,1,...,p\}}$ is a feature representation of the word piece i in the sentence. To facilitate learning associations between the token-based EKS representation and the WordPiece-based sequence T, we create E so that it contains k repeated vectors for each token where k is the token's number of word pieces. Thereby, we also consider special characters, such as CLS. If multiple EKS with a total number of dimensions $\hat{d} = \sum d$ are used, their representations of the sentence are stacked resulting in $E \in \mathbb{R}^{n \times \hat{d}}$.

The third input is a target mask $M \in \mathbb{R}^n$, i.e., for each word piece i in the sentence that belongs to the target, $m_i = 1$, else 0 (Gao et al., 2019).

4.2 Embedding layer

We feed T into the LM to yield a contextualized word embedding of shape $\mathbb{R}^{n \times h}$, where h is the number of hidden states in the language model. We feed E into a randomly initialized matrix $W_E \in \mathbb{R}^{\hat{d} \times h}$ to yield an EKS embedding. We repeat M to be of shape $\mathbb{R}^{n \times h}$. By creating all embeddings in the same shape, we facilitate a balanced influence of each input to the model's downstream components. We stack all embeddings to form a matrix $TEM \in \mathbb{R}^{n \times 3h}$.

4.3 Interaction layer

We allow the three embeddings to interact using a single-layer BiGRU (Hosseinia et al., 2020), which yields hidden states $H \in \mathbb{R}^{n \times 6h} = \text{BiGRU}(TEM)$. RNNs, such as LSTMs and GRUs, are commonly used to learn a higher-level representation of a word embedding, especially in state-of-the-art TSC prior to BERT-based models but also recently (Liu et al., 2015; Li et al., 2019; Hosseinia et al., 2020; Zhang et al., 2020). We

---

[8] For readability, we showcase inputs as used for BERT.

choose an BiGRU over an LSTM because of the smaller number of parameters in BiGRUs, which may in some cases result in better performance (Chung et al., 2014; Jiang et al., 2019; Hosseinia et al., 2020; Gruber and Jockisch, 2020).

4.4 Pooling and decoding

We employ three common pooling techniques to turn the interacted, sequenced representation H into a single vector (Hosseinia et al., 2020). We calculate element-wise (1) mean and (2) maximum over all hidden states H and retrieve the (3) last hidden state h_{n-1}. Then, we stack the three vectors to P, feed P into a fully connected layer FC so that $z = FC(P)$ and calculate $y = \sigma(z)$.

5 Experiments

5.1 Experimental data

In addition to NewsMTSC, we use the three established TSC sets: Twitter, Laptop, and Restaurant.

5.2 Evaluation metrics

We use metrics established in the TSC literature: macro F1 on all ($F1_m$) and only the positive and negative classes ($F1_{pn}$), accuracy (a), and average recall (r_a). If not otherwise noted performances are reported for our primary metric, $F1_m$.

5.3 Baselines

We compare our model with TSC methods that yield state-of-the-art results on at least one of the established datasets: SPC-BERT (Devlin et al., 2019): input is identical to our text input. FC and softmax is calculated on CLS token. TD-BERT (Gao et al., 2019): masks hidden states depending on whether they belong to the target mention. LCF-BERT (Zeng et al., 2019): similar to TD but additionally weights hidden states depending on their token-based distance to the target mention. We use the improved implementation (Yang, 2020) and enable the dual-LM option, which yields slightly better performance than using only one LM instance (Zeng et al., 2019). We also planned to test LCFS-BERT (Phan and Ogunbona, 2020) but due to technical issues we were not able to reproduce the authors' results and thus exclude LCFS from our experiments.

5.4 Implementation details

To find for each model the best parameter configuration, we perform an exhaustive grid search. Any number we report is the mean of five experiments that we run per configuration. We randomly split each test set into a dev-set (30%) and the actual test-set (70%). We test the base version of three LMs: BERT, RoBERTa, and XLNET. For all methods, we test parameters suggested by their respective authors.[9] We test all 15 combinations of the following 4 EKS: (1) *SENT* (Hu and Liu, 2004): a sentiment dictionary (number of non-mutually exclusive dimensions: 2, domain: customer reviews). (2) *LIWC* (Tausczik and Pennebaker, 2010): a psychometric dictionary (73, multiple). (3) *MPQA* (Wilson et al., 2005): a subjectivity dictionary (3, multiple). (4) *NRC* (Mohammad and Turney, 2010): dictionary of sentiment and emotion (10, multiple).

5.5 Overall performance

Table 2 reports the performances of the models using different LMs and evaluated on both test sets. We find that the best performance is achieved by our model ($F1_m = 83.1$ on *test-rw* compared to 81.8 by prior state-of-the-art). For all models, performances are (strongly) improved when using RoBERTa, which is pre-trained on news texts, or XLNET, likely because of its large pre-training corpus. Because of limited space, XLNET is not reported in Table 2, but results are generally similar to RoBERTa except for the TD model, where XLNET degrades performance by 5-9pp. Looking at BERT, we find no significant improvement of GRU-TSC over prior state-of-the-art. Even if we domain-adapt BERT (Rietzler et al., 2019) for 3 epochs on a random sample of 10M English sentences (Gebhard and Hamborg, 2020), BERT's performance ($F1_m = 81.8$) is lower than RoBERTa. We notice a performance drop for all models when comparing *test-rw* and *test-mt*. It seems that RoBERTa is better able to resolve in-sentence relations between multiple targets (performance degeneration of only up to -0.6pp) than BERT (-2.9pp). We suggest to use RoBERTa for TSC on news, since fine-tuning it is faster than fine-tuning XLNET, and RoBERTa achieves similar or better performance than other LMs.

While GRU-TSC yields competitive results on

---

[9] Epochs $\in \{2, 3, 4\}$; batch size $\in \{8, 16\}$ (due to constrained resources not 32); learning rate $\in \{2e{-}5, 3e{-}5, 5e{-}5\}$; label smoothing regularization (LSR) (Szegedy et al., 2016): $\epsilon \in \{0, .2\}$; dropout rate: .1; $\mathcal{L}_2$ regularization: $\lambda = 1e{-}5$; SRD for LCF $\in \{3, 4, 5\}$. We use Adam optimization (Kingma and Ba, 2014), Xavier uniform initialization (Glorot and Bengio, 2010), and cross-entropy loss.

Model		Test-rw				Test-mt			
		$F1_m$	a	$F1_{pn}$	r_a	$F1_m$	a	$F1_{pn}$	r_a
BERT	SPC	**80.1**	80.7	79.5	79.8	73.7	76.1	71.1	76.0
	TD	79.4	79.9	78.9	80.0	75.6	79.1	72.0	75.8
	LCF	79.7	80.9	78.9	79.2	**77.7**	80.5	74.6	79.1
	GRU	**80.2**	81.1	79.7	80.0	77.3	80.0	74.1	77.9
RoBERTa	SPC	81.1	82.7	80.5	80.6	79.4	81.6	77.0	79.9
	TD	81.7	82.5	81.3	81.4	78.4	81.1	75.3	78.2
	LCF	81.4	82.5	80.8	81.1	81.2	83.8	78.6	81.7
	GRU	**83.1**	83.8	82.9	83.3	**82.5**	84.6	80.2	81.0

Table 2: Experimental results on the two test sets.

	Laptop		Restaurant		Twitter	
	$F1_m$	a	$F1_m$	a	$F1_m$	a
SPC	77.4	80.3	78.8	86.0	73.6	75.3
TD	74.4	78.9	78.4	85.1	74.3	77.7
LCF	**79.6**	82.4	**81.7**	87.1	**75.8**	77.3
GRU	79.0	82.1	80.7	86.0	74.6	76.0

Table 3: Results on previous TSC datasets.

Name	$F1_m$	a
no EKS	78.2	81.0
zeros	78.4	81.1
SENT	**80.7**	83.0
LIWC	**80.8**	83.1
MPQA	78.8	80.8
NRC	80.0	82.0
best combination	**81.0**	83.3

Table 4: Results of exemplary EKS combinations.

previous TSC datasets (Table 3), LCF is the top performing model.[10] When comparing the performances across all four datasets, the importance of the consolidation becomes apparent, e.g., performance is lowest on Twitter, which employs a simplistic consolidation (Section 3.4). The performance differences of individual models when contrasting their use on prior datasets and NewsMTSC highlight the need LCF performs consistently best on prior datasets but worse than GRU-TSC on NewsMTSC. One reason might be that LCF's weighting approach relies on a static distance parameter, which seems to degrade performance when used on longer texts as in NewsMTSC (Section 3.6). When increasing LCF's window width SRD, we notice a slight improvement of 1pp (SRD=5) but degradation for larger SRD.

5.6 Ablation study

We perform an ablation study to test the impact of four key factors: target mask, EKS, coreferential mentions, and fine-tuning the LM's parameters. We test all LMs and if not noted otherwise report results for RoBERTa since it generally performs best (Section 5.5). We report results for *test-mt* (performance influence is similar on either test set, with performances generally being $\approx$ 3-5pp higher on *test-rw*). Overall, we find that our changes to the initial design (Hosseinia et al., 2020) contribute to an improvement of approximately 1.9pp. The most influential changes are the selected EKS and in part use of coreferential mentions. Using the target mask input channel without coreferences and LM fine-tuning yield insignificant improvements of up to 0.3 each. We do not test the VADER-based sentence classification proposed by Hosseinia et al. (2020) since we expect no improvement by using it for various reasons. For example, VADER uses a dictionary created for a domain other than news and classifies the sentence's overall sentiment and thus is target-independent.

Table 4 details the results of exemplary EKS, showing that the best combination (SENT, MPQA, and NRC) yields an improvement of 2.6pp compared to not using an EKS (zeros). The single best EKS (LIWC or SENT) each yield an improvement of 2.4pp. The two EKS "no EKS" and "zeros" represent a model lacking the EKS input channel and an EKS that only yields 0's, respectively.

The use of coreferences has a mixed influence on performance (Table 5). While using coreferences has no or even negative effect in our model for large LMs (RoBERTa and XLNET), it can be beneficial

---

[10] For previous models, Table 3 lists results reported by their authors. In our experiments, we find 0.4-1.8pp lower performance compared to the reported results.

Name	BERT	RoBERTa
none	73.1	**78.1**
target mask	73.3	**78.2**
add coref. to mask	**75.6**	**78.1**
add coref. as example	73.0	73.4

Table 5: Influence of target mask and coreferences.

for smaller LMs (BERT) or batch sizes (8). When using the mode "ignore," "add coref. to mask," and "add coref. as example" we ignore coreferences, add them to the target mask, and create an additional example for each, respectively. Mode "none" represents a model that lacks the target mask input channel.

6 Error Analysis

To understand the limitations of GRU-TSC, we carry out a manual error analysis by investigating a random sample of 50 incorrectly predicted examples for each of the test sets. For *test-rw*, we find the following potential causes (not mutually exclusive): edge cases with very weak, indirect, or in part subjective sentiment (22%) or where both the predicted and true sentiment can actually be considered correct (10%); sentiment of given target confused with different target (14%). Further, sentence's sentiment is unclear due to missing context (10%) and consolidated answer in NewsMTSC is wrong (10%). In 16% we find no apparent reason. For *test-mt*, potential causes occur approximately similarly often except that targets are confused more often (20%).

7 Future Work

We identify three main areas for future work. The first area is related to the dataset. Instead of consolidating multiple annotators' answers during the dataset creation, we propose to test to integrate the label selection into the model (Raykar et al., 2010). Integrating the label selection into the machine learning part could improve the classification performance. It could also allow us to include more sentences in the dataset, especially the edge cases that our restrictive consolidation currently discards.

To improve the model design, we propose to design the model specifically for sentences with multiple targets, for example, by classifying multiple targets in a sentence simultaneously. While we early tested various such designs, we did not report them in the paper due to their comparably poor performances. Further work in this direction should perhaps also focus on devising specialized loss functions that set multiple targets and their polarity into relation. Lastly, one can improve various technical details of GRU-TSC, e.g., by testing other interaction layers, such as LSTMs, or using layer-specific learning rates in the overall model, which can increase performance (Sun et al., 2019b).

8 Conclusion

We present NewsMTSC, a dataset for target-dependent sentiment classification (TSC) on news articles consisting of 11.3k manually annotated examples. Compared to prior TSC datasets, it is different in key factors, such as its texts are on average 50% longer, sentiment is expressed explicitly only rarely, and there is a separate test set for multi-target sentences. As a consequence, state-of-the-art TSC models yield non-optimal performances. We propose GRU-TSC, which uses a bidirectional GRU on top of a language model (LM) and other embeddings, instead of masking or weighting mechanisms as employed by prior state-of-the-art. We find that GRU-TSC achieves superior performances on NewsMTSC and is competitive on prior TSC datasets. RoBERTa yields best results compared to using BERT, because RoBERTa is pre-trained on news and we find it can better resolve in-sentence relations of multiple targets.

Acknowledgments

The work described in this paper is partially funded by the WIN program of the Heidelberg Academy of Sciences and Humanities, financed by the Ministry of Science, Research and the Arts of the State of Baden-Wurttemberg, Germany. We are grateful to the crowdworkers who participated in our online annotation and the UZH students who participated in the expert annotation: F. Jedelhauser, Y. Kipfer, J. Roberts, L. Sopa, and F. Wallin. We thank the anonymous reviewers for their valuable comments that helped to improve this paper.

References

Abeer AlDayel and Walid Magdy. 2020. Stance Detection on Social Media: State of the Art and Trends. *Preprint*.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion

Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, volume 10, pages 2200–2204, Valletta, Malta. European Language Resources Association (ELRA).

Alexandra Balahur, Ralf Steinberger, Mijail Kabadjov, Vanni Zavarella, Erik Van Der Goot, Matina Halkia, Bruno Pouliquen, and Jenya Belyaeva. 2010. Sentiment analysis in the news. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Dan Bernhardt, Stefan Krasa, and Mattias Polborn. 2008. Political polarization and the electoral effects of media bias. *Journal of Public Economics*, 92(5):1092–1104.

Douglas Biber and Edward Finegan. 1989. Styles of stance in English: Lexical and grammatical marking of evidentiality and affect. *Text - Interdisciplinary Journal for the Study of Discourse*, 9(1).

Wei-Fan Chen, Henning Wachsmuth, Khalid Al-Khatib, and Benno Stein. 2018. Learning to Flip the Bias of News Headlines. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 79–88, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiao Chen, Changlong Sun, Jingjing Wang, Shoushan Li, Luo Si, Min Zhang, and Guodong Zhou. 2020. Aspect Sentiment Classification with Document-level Sentiment Preference Modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3677, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Stroudsburg, PA, USA. Association for Computational Linguistics.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North*, pages 4171–4186, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and Domain-Aware BERT for Cross-Domain Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4019–4028, Stroudsburg, PA, USA. Association for Computational Linguistics.

Robert M. Entman. 2007. Framing Bias: Media in the Distribution of Power. *Journal of Communication*, 57(1):163–173.

Charles J. Fillmore and Collin F. Baker. 2001. Frame semantics for text understanding. In *Proceedings of NAACL WordNet and Other Lexical Resources Workshop*, pages 1–6, Pittsburgh, US.

Zhengjie Gao, Ao Feng, Xinyu Song, and Xi Wu. 2019. Target-Dependent Sentiment Classification With BERT. *IEEE Access*, 7:154290–154299.

Lukas Gebhard and Felix Hamborg. 2020. The POLUSA Dataset: 0.9M Political News Articles Balanced by Time and Outlet Popularity. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, pages 467–468, New York, NY, USA. ACM.

Deepanway Ghosal, Devamanyu Hazarika, Abhinaba Roy, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2020. KinGDOM: Knowledge-Guided DOMain Adaptation for Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3198–3210, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Journal of Machine Learning Research*, pages 249–256.

Namrata Godbole, Manja Srinivasaiah, and Steven Skiena. 2007. Large-Scale Sentiment Analysis for News and Blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*, volume 7, pages 219–222, Boulder, CO, USA.

Nicole Gruber and Alfred Jockisch. 2020. Are GRU Cells More Specific and LSTM Cells More Sensitive in Motive Classification of Text? *Frontiers in Artificial Intelligence*, 3.

Felix Hamborg, Karsten Donnay, and Bela Gipp. 2019. Automated identification of media bias in news articles: an interdisciplinary literature review. *International Journal on Digital Libraries*, 20(4):391–415.

Felix Hamborg, Karsten Donnay, and Bela Gipp. 2021. Towards Target-dependent Sentiment Classification in News Articles. In *Proceedings of the 16th iConference*, pages 1–9, Beijing, China (Virtual Event). Springer.

Marjan Hosseinia, Eduard Dragut, and Arjun Mukherjee. 2020. Stance Prediction for Contemporary Issues: Data and Experiments. In *Proceedings of* *the Eighth International Workshop on Natural Language Processing for Social Media*, pages 32–40, Stroudsburg, PA, USA. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, page 168, New York, New York, USA. ACM Press.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A Challenge Dataset and Effective Models for Aspect-Based Sentiment Analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6279–6284, Stroudsburg, PA, USA. Association for Computational Linguistics.

Daniel Kahneman and Amos Tversky. 1984. Choices, values, and frames. *American Psychologist*, 39(4):341–350.

Sung-Hee Kim, Hyokun Yun, and Ji Soo Yi. 2012. How to filter out random clickers in a crowdsourcing-based study? In *Proceedings of the 2012 BELIV Workshop on Beyond Time and Errors - Novel Evaluation Methods for Visualization - BELIV '12*, pages 1–7, New York, New York, USA. ACM Press.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv: 1412.6980.*

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442, Dublin, Ireland. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A Unified Model for Opinion Target Extraction and Target Sentiment Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6714–6721.

Bing Liu. 2012. Sentiment Analysis and Opinion Mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained Opinion Mining with Recurrent Neural Networks and Word Embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443, Stroudsburg, PA, USA. Association for Computational Linguistics.

Saif Mohammad and Peter Turney. 2010. Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 26–34, Los Angeles, CA. Association for Computational Linguistics.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. SemEval-2016 Task 4: Sentiment Analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1–18, San Diego, CA, USA. Association for Computational Linguistics.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. SemEval-2013 Task 2: Sentiment Analysis in Twitter. In *Second Joint Conference on Lexical and Computational Semantics (SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, GA, USA. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05*, pages 115–124, Morristown, NJ, USA. Association for Computational Linguistics.

Minh Hieu Phan and Philip O. Ogunbona. 2020. Modelling Context and Syntactical Features for Aspect-based Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3211–3220, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 Task 12: Aspect Based Sentiment Analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Stroudsburg, PA, USA. Association for Computational Linguistics.

James Pustejovsky and Amber Stubbs. 2012. *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*, 1 edition. O'Reilly Media, Inc., Sebastopol, CA, US.

Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. 2009. *Dataset Shift in Machine Learning*. The MIT Press.

Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca

Bogoni, and Linda Moy. 2010. Learning From Crowds. *The Journal of Machine Learning Research*, 11:1297–1322.

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic Models for Analyzing and Detecting Biased Language. In *Proceedings of the 51st Annual Meeting on Association for Computational Linguistics*, pages 1650–1659, Sofia, BG. Association for Computational Linguistics.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2019. Adapt or Get Left Behind: Domain Adaptation through BERT Language Model Finetuning for Aspect-Target Sentiment Classification. *arXiv preprint arXiv:1908.11860*.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 Task 4: Sentiment Analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv: 1910.01108*.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Targeted Sentiment Classification with Attentional Encoder Network. In *Artificial Neural Networks and Machine Learning - ICANN 2019: Text and Time Series*, pages 93–103, Cham, US. Springer International Publishing.

Ralf Steinberger, Stefanie Hegele, Hristo Tanev, and Leonida Della Rocca. 2017. Large-scale news entity sentiment analysis. In *RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning*, pages 707–715. Incoma Ltd. Shoumen, Bulgaria.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019a. Utilizing BERT for Aspect-Based Sentiment Analysisvia Constructing Auxiliary Sentence. In *Proceedings of the 2019 Conference of the North*, pages 380–385, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019b. How to Fine-Tune BERT for Text Classification? In *Chinese Computational Linguistics*, pages 194–206. Springer International Publishing, Cham, US.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826. IEEE.

Yla R. Tausczik and James W. Pennebaker. 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology*, 29(1):24–54.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing - HLT '05*, pages 347–354, Morristown, NJ, USA. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *Preprint*.

Heng Yang. 2020. LC-ABSA.

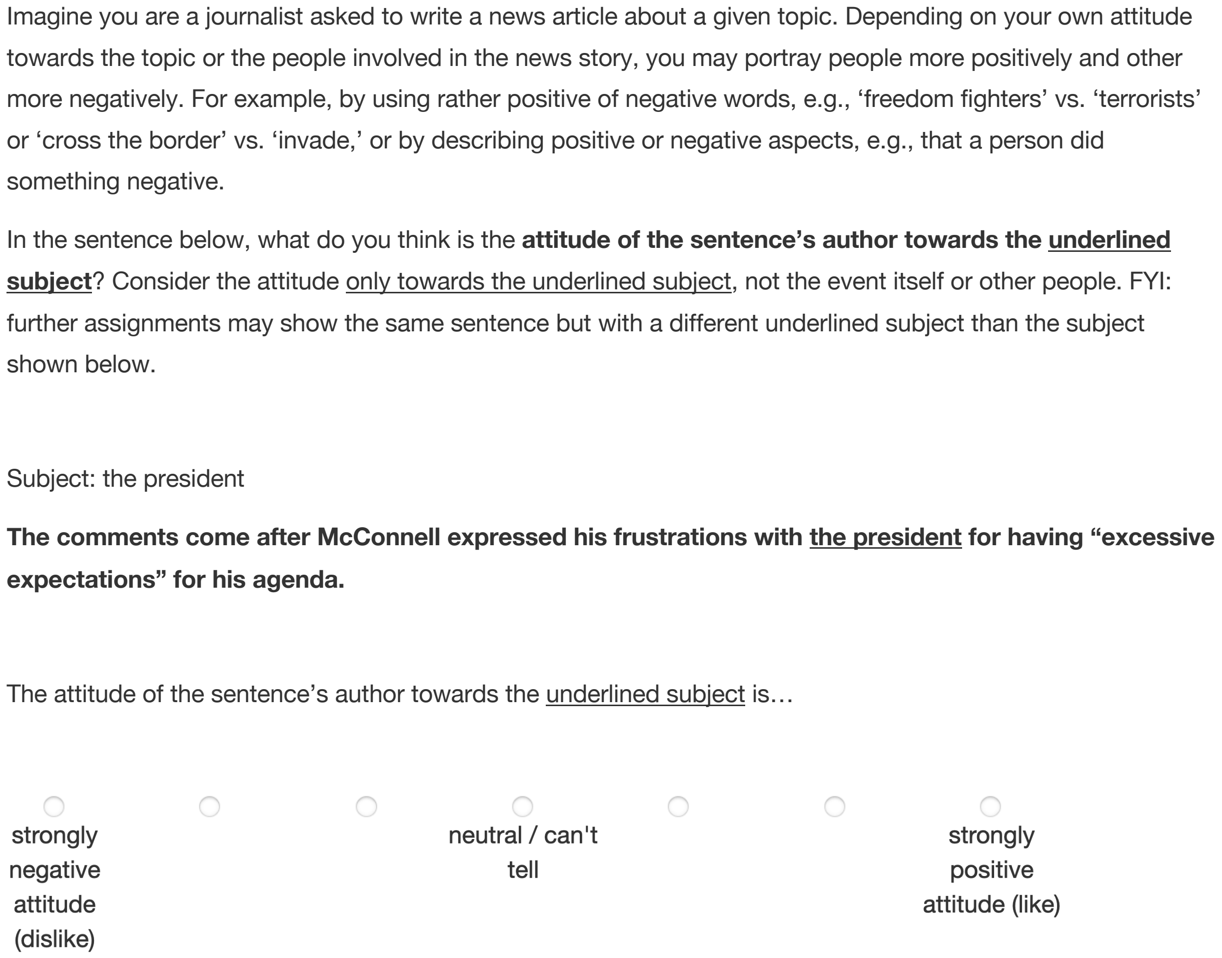
Da Yin, Tao Meng, and Kai-Wei Chang. 2020. SentiBERT: A Transferable Transformer-Based Architecture for Compositional Sentiment Semantics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3695–3706, Stroudsburg, PA, USA. Association for Computational Linguistics.

Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. 2019. LCF: A Local Context Focus Mechanism for Aspect-Based Sentiment Classification. *Applied Sciences*, 9(16):1–22.

Bowen Zhang, Min Yang, Xutao Li, Yunming Ye, Xiaofei Xu, and Kuai Dai. 2020. Enhancing Cross-target Stance Detection with Transferable Semantic-Emotion Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3188–3197, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4).

A Appendices



Imagine you are a journalist asked to write a news article about a given topic. Depending on your own attitude towards the topic or the people involved in the news story, you may portray people more positively and other more negatively. For example, by using rather positive of negative words, e.g., ‘freedom fighters’ vs. ‘terrorists’ or ‘cross the border’ vs. ‘invade,’ or by describing positive or negative aspects, e.g., that a person did something negative.

In the sentence below, what do you think is the **attitude of the sentence’s author towards the underlined subject**? Consider the attitude only towards the underlined subject, not the event itself or other people. FYI: further assignments may show the same sentence but with a different underlined subject than the subject shown below.

Subject: the president

The comments come after McConnell expressed his frustrations with the president for having “excessive expectations” for his agenda.

The attitude of the sentence’s author towards the underlined subject is…

○	○	○	○	○	○	○
strongly negative attitude (dislike)			neutral / can't tell			strongly positive attitude (like)

Figure 2: Final version of the annotation instructions shown on MTurk.

Cross-lingual Contextualized Topic Models with Zero-shot Learning

Federico Bianchi
Bocconi University
Via Sarfatti 25, 20136
Milan, Italy
f.bianchi@unibocconi.it

Silvia Terragni
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
s.terragni4@campus.unimib.it

Dirk Hovy
Bocconi University
Via Sarfatti 25, 20136
Milan, Italy
dirk.hovy@unibocconi.it

Debora Nozza
Bocconi University
Via Sarfatti 25, 20136
Milan, Italy
debora.nozza@unibocconi.it

Elisabetta Fersini
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
elisabetta.fersini@unimib.it

Abstract

Many data sets (e.g., reviews, forums, news, etc.) exist parallelly in multiple languages. They all cover the same content, but the linguistic differences make it impossible to use traditional, bag-of-word-based topic models. Models have to be either single-language or suffer from a huge, but extremely sparse vocabulary. Both issues can be addressed by transfer learning. In this paper, we introduce a zero-shot cross-lingual topic model. Our model learns topics on one language (here, English), and predicts them for unseen documents in different languages (here, Italian, French, German, and Portuguese). We evaluate the quality of the topic predictions for the same document in different languages. Our results show that the transferred topics are coherent and stable across languages, which suggests exciting future research directions.

1 Introduction

Topic models (Blei et al., 2003; Blei, 2012) allow us to find the main themes and overarching tropes in textual data. However, traditional methods are language-specific and cannot be used in a *transferable manner*. They rely on a fixed vocabulary specific to the training language.

Therefore, currently available topic models suffer from two limitations: (i) they cannot handle unknown words by default, and (ii) they cannot easily be applied to other languages - except the one in the training data - since the vocabulary would not match. Training on several languages together, though, results in a vocabulary so vast that it creates problems with parameter size, search, and overfitting (Boyd-Graber et al., 2014). Traditional topic modeling provides methods to extract meaningful word distributions from "unstructured" text but requires language-specific bag-of-words (BoW) representations (Boyd-Graber and Blei, 2009; Jagarlamudi and Daumé, 2010).

A cross-lingual setup proves ideal for *transfer learning*: provided that the gist of topics is the same across languages, we can learn this gist on texts in one language and then apply it to others. This setup is *zero-shot learning*: we train a model on one language and test it on several other languages to which the model had no access during training.

To this end, we need to leverage external information to support the topic modeling task. Indeed, topic models have often gained significant advantages from introducing external knowledge, e.g., document relationships (Yang et al., 2015; Wang et al., 2020; Terragni et al., 2020a,b) and word embeddings (Nozza et al., 2016; Li et al., 2016; Zhao et al., 2017; Dieng et al., 2020). Recently, pre-trained contextualized embeddings, e.g., BERT (Devlin et al., 2019) embeddings, have enabled exciting new results in several NLP tasks (Rogers et al., 2020; Nozza et al., 2020). More importantly, there do exist contextualized embeddings that are also multilingual.

This paper introduces a novel neural topic modeling architecture in which we *replace* the input BoW document representations with multilingual *contextualized* embeddings. Neural topic models take in input the document BoW representations, which provide valuable symbolic information; however, this information's structure is lost after the first hidden layer in any neural architecture. We, therefore, hypothesize that contextual information can replace the BoW representation.

We use a neural encoding layer for the pre-trained document representations from a contextu-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1676–1683
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

alized embedding model input (e.g., BERT) before the neural topic model's sampling process. This change allows us to address the two limitations mentioned above jointly: (i) our approach solves the problem of dealing with unseen words at test time since we do not need them to have a BoW representation; moreover, (ii) the model infers topics on unseen documents in languages other than the one in the training data. The inferred topics consist of tokens from the training language and can be applied to any supported test language. We show the high quality of the resulting topics for four test languages both quantitatively and qualitatively.

To the best of our knowledge, there is no prior work on zero-shot cross-lingual topic modeling. Our model can be applied to new languages after training is complete and does not require external resources, alignment, or other conditions. Nonetheless, the flexibility of the input means our model will benefit from any future improvement of language modeling techniques.

Contributions We release a novel neural topic model that relies on language-independent representations to generate topic distributions. We show that this input can replace the standard input BoW without loss of quality. We show that its multilingual representations enable zero-shot cross-lingual tasks. The solution we propose is straightforward and does not require high computational resources since it can efficiently run on common laptops (see Appendix). We have implemented the tool as a documented python package available at `https://github.com/MilaNLProc/contextualized-topic-models`.

2 Contextualized Neural Topic Models

We extend Neural-ProdLDA (Srivastava and Sutton, 2017), one of the most recent and promising approaches of neural topic modeling, based on the Variational AutoEncoder (VAE) (Kingma and Welling, 2014). The neural variational framework trains an inference network, i.e., a neural network that directly maps the BoW representation of a document onto a continuous latent representation. A decoder network then reconstructs the BoW by generating its words from the latent document representation. This latent representation is sampled from a Gaussian distribution parameterized by μ and σ^2 that are part of the variational inference framework (Kingma and Welling, 2014) — see (Srivastava and Sutton, 2017) for more details.

We replace the input BoW in Neural-ProdLDA with pre-trained multilingual representations from SBERT (Reimers and Gurevych, 2019), a recent and effective model for contextualized representations. In Figure 1, we sketch the architecture of our contextualized neural topic model. The final *reconstructed BoW* layer is still a component of our model: the BoW representation is necessary for the model's training to obtain the topic indicators (i.e., the most likely words representing a topic), but it becomes useless during testing.

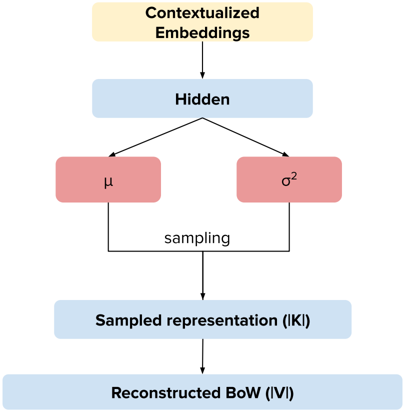


Figure 1: High-level schema of the architecture for the proposed contextualized neural topic model.

Our proposed model, **Zero-Shot Topic Model** (ZeroShotTM), is trained with input document representations that account for word-order and contextual information, overcoming one of the central limitations of BoW models. Moreover, the use of language-independent document representations allows us to do zero-shot topic modeling for unseen languages. This property is essential in low-resource settings in which there is little data available for the new languages. Because multilingual contextualized representations exist for multiple languages, it allows zero-shot modeling in a *cross-lingual* scenario. Indeed, ZeroShotTM is language-independent: given a contextualized representation of a new language as input,[1] it can predict the topic distribution of the document. The predicted topic descriptors, though, will be from the training language. Let us also notice that our method is agnostic about the choice of the neural topic model architecture (here, Neural-ProdLDA), as long as it extends a Variational Autoencoder.

---

[1] As long as a multilingual model - like multilingual BERT - covers it.

3 Experiments

Our experiments evaluate two main hypotheses: (i) *we can define a topic model that does not rely on the BoW input but instead uses contextual information;* (ii) *the model can tackle zero-shot cross-lingual topic modeling*. The Appendix contains more details about the experiments (e.g., code, data, runtime, replication details).

Datasets We use datasets collected from English Wikipedia abstracts from DBpedia.[2] The first dataset (W1) contains 20,000 randomly sampled abstracts. The second dataset (W2) contains 100,000 English documents. We use 99,700 documents as training and consider the remaining 300 documents as the test set. We collect the 300 respective instances in Portuguese, Italian, French, and German. This collection creates a test set of comparable documents, i.e., documents that refer to the same entity in Wikipedia, but in different languages.

We extract only the first 200 tokens of each abstract to reduce the length limit's effects in the tokenization process. In particular, we use the efficient and effective SBERT (Reimers and Gurevych, 2019),[3] using the multilingual model,[4] on this unpreprocessed text. We then remove stopwords and use the most frequent remaining 2,000 words to create the English vocabulary for BoW model comparisons.

3.1 To Contextualize or Not To Contextualize

First, we want to check if **ZeroShotTM** maintains comparable performance to other topic models; if this is true, we can then explore its performance in a cross-lingual setting. Since we use only English text, in this setting we use English representations.[5]

Model	τ (50)	τ (100)
ZeroShotTM	0.1632	0.1381
Combined TM	0.1644	**0.1409***
Neural-ProdLDA	**0.1658**	0.1285
LDA	-0.0246	-0.0757

Table 1: NPMI Coherences on W1 data set. * denotes the statistically significant results (t-test).

We compare **ZeroShotTM** on W1 with: (i) Combined TM (Bianchi et al., 2020), an extension of Neural-ProdLDA that concatenates *both* BoWs and SBERT representations (transformed to the same dimension of the BoWs) as inputs to the model, (ii) Neural-ProdLDA (Srivastava and Sutton, 2017), and (iii) LDA (Blei et al., 2003).

We compute the topic coherence (Lau et al., 2014) via NPMI (τ) for 50 and 100 topics averaging models' results over 30 runs. We report the results in Table 1. ZeroShotTM obtains comparable results to Combined TM and Neural-ProdLDA in this setting. Contextualized embeddings *can* replace BoW input representations without loss of coherence.

3.2 Zero-shot Cross-Lingual Topic Modeling

ZeroShotTM can be used for zero-shot cross-lingual topic modeling. We evaluate multilingual topic predictions on the multilingual abstracts in W2. We use SBERT [6] to generate multilingual embeddings as the input of the model.

3.2.1 Quantitative Evaluation

Since the predicted document-topic distribution is subject to a stochastic sampling process, we average it over 100 samples to obtain a better estimate.

Metrics We expect the topic distributions over a set of comparable documents (e.g., in English and Portuguese) to be similar to each other. We compare the topic distributions of each abstract in a test language with the topic distribution of the respective abstract in English, which is the training language. Note that the English test document is also unseen, i.e., the training data does not include it. We evaluate our model on three different metrics. The first metric is **matches**, i.e., the percentage of times the predicted topic for the non-English test document is the same as for the respective test document in English. The higher the scores, the better.

To also account for similar but not exactly equal topic predictions, we compute the **centroid embeddings** of the five words describing the predicted topic for both English and non-English documents. Then we compute the cosine similarity between those two centroids (CD).

Finally, to capture the **distributional similarity**, we also compute the KL divergence between the

---

[2] `https://wiki.dbpedia.org/downloads-2016-10`

[3] `https://github.com/UKPLab/sentence-transformers`

[4] We use the *distiluse-base-multilingual-cased* embeddings for this experiment available on the authors' repository.

[5] We use the *bert-base-nli-mean-tokens* model.

[6] `https://github.com/UKPLab/sentence-transformers`

Lang	Mat25↑	KL25↓	CD25↑	Mat50↑	KL50↓	CD50↑
IT	75.67	0.16	0.84	62.00	0.21	0.75
FR	79.00	0.14	0.86	63.33	0.19	0.77
PT	78.00	0.14	0.85	68.00	0.19	0.79
DE	79.33	0.15	0.85	64.33	0.20	0.77
ZeroShotTM Avg	**78.00**	**0.15**	**0.85**	64.41	0.20	0.77
Ori Avg	76.00	**0.15**	0.84	**69.00**	**0.19**	**0.79**
Uni	4.00	0.75	—	2.00	0.85	—

Table 2: Match, KL, and centroid similarity for 25 and 50 topics on various languages on W2.

predicted topic distribution on the test document and the same test document in English. Here, lower scores are better, indicating that the distributions do not differ by much.

Automatic Evaluation We use two baselines: the first one (Ori) consists of performing topic modeling on documents translated into English via DeepL.[7] While this is an easily accessible baseline, automatic translation is costly and may introduce bias in the representations (as shown by Hovy et al. (2020)). We compare the predicted topics of each translated document to the ones predicted for the original English document (as done above). The second baseline is a uniform distribution (Uni): we compute all the metrics over a uniform distribution (this baseline gives a lower bound).

Table 2 shows the evaluation results of our model in the zero-shot context. Note that because we trained on English data, the topic descriptors are in English. Topic predictions are significantly better than the uniform baselines: more than 70% of the times, the predicted topic on the test set matches the topic of the same document in English. The CD similarity suggests that even when there is no match, the predicted topic on the unseen language is at least similar to the one on the English testing data. Simultaneously, the predictions for the contextualized model are in line with the ones obtained using the translations (Ori Avg), showing that our model is capable of finding good topics for documents in unseen languages without the need for translation.

Manual Evaluation We rated the predicted topics for 300 test documents in five languages (thus, 1500 docs including English) on an ordinal scale from 0-3. A 0 rate means that the predicted topic is wrong, a 1 rate means the topic is somewhat related, a 2 rate means the topic is good, and a 3 rate means the topic is entirely associated with the considered document. Table 3 shows the results per language. We evaluate the inter-rater reliability using Gwet AC1 with ordinal weighting (Gwet, 2014). The resulting value of 0.88 indicates consistent scoring.

Language	Average Topic Quality
English	2.35
Italian	2.29
French	2.22
Portuguese	2.26
German	2.19
Average	2.26

Table 3: Average topic quality (out of 3).

3.2.2 Qualitative Evaluation

In Table 4, we show some examples of topic predictions on test languages. Our model predicts the main topic for all languages, even though they were unseen during training.

The predicted topic is generally consistent with the text. I.e., the topics are easily interpretable and give the user a coherent impression. In some circumstances, noise biases the results: dates in the abstract tend to make the model predict a topic about time. Another interesting case is the abstract of the artist Joan Brossa, who was both a poet and a graphic designer. In the English and Italian abstract, the model has discovered a topic related to writing. In constrast, in the Portuguese abstract, the model has found a topic related to art, which is still meaningful.

---

[7] `https://www.deepl.com/`

Lang	Sentence	Predicted Topic
EN	Blackmore's Night is a British/American traditional folk rock duo [...]	rock, band, bass, formed
IT	I Blackmore's Night sono la band fondatrice del renaissance rock [...]	rock, band, bass, formed
PT	Blackmore's Night é uma banda de folk rock de estilo renascentista [...]	rock, band, bass, formed
EN	Langton's ant is a two-dimensional Turing machine with [...]	mathematics, theory, space, numbers
FR	On nomme fourmi de Langton un automate cellulaire [...]	mathematics, theory, space, numbers
DE	Die Ameise ist eine Turingmaschine mit einem zweidimensionalen [...]	mathematics, theory, space, numbers
EN	The Journal of Organic Chemistry, colloquially known as JOC or [...]	journal, published, articles, editor
IT	Journal of Organic Chemistry è una rivista accademica [...]	journal, published, articles, editor
PT	Journal of Organic Chemistry é uma publicação científica [...]	journal, published, articles, editor
EN	Joan Brossa [...] was a Catalan poet, playwright, graphic designer [...]	book, french, novel, written
IT	Fu l'ispiratore e uno dei fondatori della rivista "Dau al Set"[...]	book, french, novel, written
PT	Joan Brossa i Cuervo [...] foi um poeta, dramaturgo, artista plástico [...]	painting, art, painter, works

Table 4: Examples of zero-shot cross-lingual topic classification in various languages with ZeroShotTM.

4 Related Work

While not in a zero-shot fashion, several researchers have studied multilingual and cross-lingual topic modeling (Ma and Nasukawa, 2017; Gutiérrez et al., 2016; Hao and Paul, 2018; Heyman et al., 2016; Liu et al., 2015; Krstovski et al., 2016).

The first model proposed to process multilingual corpora with LDA is the Polylingual Topic Model by Mimno et al. (2009). It uses LDA to extract language-consistent topics from parallel multilingual corpora, assuming that translations share the same topic distributions. Models that transfer knowledge on the document level have many variants, including (Hao and Paul, 2018; Heyman et al., 2016; Liu et al., 2015; Krstovski et al., 2016). However, existing models require to be trained on multilingual corpora and are always language-dependent: they cannot predict the main topics of a document in an unseen language.

Other models use multilingual dictionaries (Boyd-Graber and Blei, 2009; Jagarlamudi and Daumé, 2010), requiring some predefined mapping. Embeddings, both for words and documents, have been shown to capture a wide range of semantic, syntactic, and social aspects of language (Hovy and Purschke, 2018; Rogers et al., 2020). Our work adds language-independent topics to that list.

5 Conclusions

We propose a novel neural architecture for cross-lingual topic modeling using contextualized document embeddings as input. Our results show that (i) contextualized embeddings can replace the input BoW representations and (ii) using contextualized representations allows us to tackle zero-shot cross-lingual topic modeling. The resulting model can be trained on any one language and applied to any other language for which embeddings are available.

Acknowledgements

Federico Bianchi, Dirk Hovy, and Debora Nozza are members of the Bocconi Institute for Data Science and Analytics (BIDSA) and the Data and Marketing Insights (DMI) unit.

References

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974*.

David M Blei. 2012. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jordan L. Boyd-Graber and David M. Blei. 2009. Multilingual topic models for unaligned text. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009*, pages 75–82. AUAI Press.

Jordan L. Boyd-Graber, David Mimno, and David Newman. 2014. *Care and Feeding of Topic Models: Problems, Diagnostics, and Improvements*, CRC Handbooks of Modern Statistical Methods. CRC Press.

Stephen Carrow. 2018. Pytorchavitm: Open source avitm implementation in pytorch. Github.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2019, Volume 1 (Long and Short Papers), pages 4171–4186.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

E.D. Gutiérrez, Ekaterina Shutova, Patricia Lichtenstein, Gerard de Melo, and Luca Gilardi. 2016. Detecting cross-cultural differences using a multilingual topic model. *Transactions of the Association for Computational Linguistics*, 4:47–60.

Kilem L Gwet. 2014. *Handbook of inter-rater reliability: The definitive guide to measuring the extent of agreement among raters*. Advanced Analytics, LLC.

Shudong Hao and Michael J. Paul. 2018. Learning multilingual topics from incomparable corpora. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 2595–2609. Association for Computational Linguistics.

Geert Heyman, Ivan Vulic, and Marie-Francine Moens. 2016. C-BiLDA extracting cross-lingual topics from non-parallel texts by distinguishing shared from unshared content. *Data Mining and Knowledge Discovery*, 30(5):1299–1323.

Dirk Hovy, Federico Bianchi, and Tommaso Fornaciari. 2020. "You sound just like your father" Commercial machine translation systems include stylistic biases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1686–1690, Online. Association for Computational Linguistics.

Dirk Hovy and Christoph Purschke. 2018. Capturing regional variation with distributed place representations and geographic retrofitting. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4383–4394, Brussels, Belgium. Association for Computational Linguistics.

Jagadeesh Jagarlamudi and Hal Daumé. 2010. Extracting multilingual topics from unaligned comparable corpora. In *Advances in Information Retrieval*, pages 444–456, Berlin, Heidelberg. Springer Berlin Heidelberg.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014*.

Kriste Krstovski, David A. Smith, and Michael J. Kurtz. 2016. Online multilingual topic models with multi-level hyperpriors. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016*, pages 454–459. Association for Computational Linguistics.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 530–539.

Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. 2016. Topic modeling for short texts with auxiliary word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pages 165–174.

Xiaodong Liu, Kevin Duh, and Yuji Matsumoto. 2015. Multilingual topic models for bilingual dictionary extraction. *ACM Transactions on Asian Language Information Processing*, 14(3):11:1–11:22.

Tengfei Ma and Tetsuya Nasukawa. 2017. Inverted bilingual topic models for lexicon extraction from non-parallel data. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 4075–4081. IJCAI.org.

David M. Mimno, Hanna M. Wallach, Jason Naradowsky, David A. Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009*, pages 880–889. Association for Computational Linguistics.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [MASK]? Making Sense of Language-Specific BERT Models. *arXiv preprint arXiv:2003.02912*.

Debora Nozza, Elisabetta Fersini, and Enza Messina. 2016. Unsupervised irony detection: a probabilistic model with word embeddings. In *International Conference on Knowledge Discovery and Information Retrieval*, volume 2, pages 68–76. SCITEPRESS.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *arXiv preprint arXiv:2002.12327*.

Akash Srivastava and Charles A. Sutton. 2017. Autoencoding variational inference for topic models. In *In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net.

Silvia Terragni, Elisabetta Fersini, and Enza Messina. 2020a. Constrained relational topic models. *Information Sciences*, 512:581–594.

Silvia Terragni, Debora Nozza, Elisabetta Fersini, and Messina Enza. 2020b. Which matters most? comparing the impact of concept and document relationships in topic models. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 32–40, Online. Association for Computational Linguistics.

Chaojie Wang, Hao Zhang, Bo Chen, Dongsheng Wang, Zhengjue Wang, and Mingyuan Zhou. 2020. Deep relational topic modeling via graph poisson gamma belief network. *Advances in Neural Information Processing Systems*, 33.

Weiwei Yang, Jordan Boyd-Graber, and Philip Resnik. 2015. Birds of a feather linked together: A discriminative topic model using link-based priors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 261–266.

He Zhao, Lan Du, Wray Buntine, and Gang Liu. 2017. Metalda: A topic model that efficiently incorporates meta information. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 635–644. IEEE.

A Datasets

We used the English DBpedia 2016-10 abstract dump[8] to create our datasets.

W1 We randomly sampled 20,000 documents from the English DBpedia abstract dump to create our first set of documents. We created W1 to provide a quick collection of documents to test if our Contextual TM performance does not decrease significantly.

W2 We collected 100,000 abstracts sampling randomly from those that had at least 200 chars. Given this set, we extracted 300 random English abstracts. Given the random abstracts, we retrieved the respective version in other languages using the DBpedia SPARQL endpoint.[9] We manually evaluated the quality of the 300 abstracts since we looked at each of those during our manual evaluation, finding no mismatch between the abstract and no corrupted text.

A.1 Preprocessing

We followed a standard pre-processing pipeline to generate the preprocessed set of documents. We removed punctuation, digits, and nltk's English stop-words.[10] Following other researchers, we selected 2,000 as the maximum number of words for the BoW, and thus we kept in the abstracts only the 2,000 most frequent words.

B Models and Baselines

B.1 Neural-ProdLDA

We use the implementation made available by Carrow (2018) since it is the most recent and with the most updated packages (e.g., one of the latest versions of PyTorch). The model is trained for 100 epochs. We use ADAM optimizer (with a learning rate equal to 2e-3). The inference network is composed of a single hidden layer and 100-dimension of softplus units. The priors over the topic and document distributions are learnable parameters. Momentum is set to 0.99, the learning rate is set to 0.002, and we apply 20% of drop-out to the hidden document representation. The batch size is equal to 200. More details related to the architecture can be found in the original work (Srivastava and Sutton, 2017).

B.2 ZeroShot TM

The model and the hyper-parameters are the same for Neural-ProdLDA, with the difference that we replace the BoW with SBERT features. The model is trained for 100 epochs. We use ADAM optimizer.

B.3 Combined TM

The model (Bianchi et al., 2020)[11] and the hyper-parameters are the same used for Neural-ProdLDA with the difference that we also use SBERT features in combination with the BoW: we take the SBERT embeddings, apply a (learnable) function/dense layer $R^{512} \rightarrow R^{|V|}$ and concatenate the representation to the BoW. The model is trained for 100 epochs. We use ADAM optimizer.

B.4 LDA

We use Gensim's[12] implementation of this model. The hyper-parameters $alpha$ and $beta$, controlling the document-topic and word-topic distribution respectively, are estimated from the data during training.

---

[8] `https://wiki.dbpedia.org/downloads-2016-10`

[9] `https://dbpedia.org/sparql`

[10] `https://www.nltk.org/`

[11] `https://github.com/MilaNLProc/contextualized-topic-models`

[12] `https://radimrehurek.com/gensim/models/ldamodel.html`

C Computing Infrastructure

We ran experiments on two common laptops, equipped with a GeForce GTX 1050 (running CUDA 10). As our experiments show, the models can be easily run with basic hardware (having a GPU is better than just using CPU, but the experiments can also be replicated on CPU). Both laptops have 16GB of RAM.

C.1 Runtime

Our implementation is written in PyTorch and runs on both GPU and CPU. Table 5 shows the runtime for one epoch of both our Combined TM and Neural-ProdLDA for 25 and 50 topics on the GeForce GTX 1050. Neural-ProdLDA is slightly faster than our ZeroShotTM. This is due to the additional representation that cannot be encoded as a sparse matrix. However, we believe that these numbers are comparable and make our model easy to use even with common hardware.

Model	W1 (25)	W1 (50)
ZeroShot TM	1.2s	1.2s
Neural-ProdLDA	0.8s	0.9s

Table 5: Time to complete one epoch on the **W1** dataset with 25 and 50 topics.

D Source Code

D.1 Development

Our software is available as a Python package that a user can easily install.[13]

---

[13] `https://github.com/MilaNLProc/contextualized-topic-models`

Dependency parsing with structure preserving embeddings

Ákos Kádár[*] [†] **Lan Xiao**[*] **Mete Kemertas**[‡] **Federico Fancellu**
Allan Jepson **Afsaneh Fazly**
Samsung AI Centre Toronto (SAIC Toronto)
`{lan.xiao, federico.f, allan.jepson, a.fazly}@samsung.com`

Abstract

Modern neural approaches to dependency parsing are trained to predict a tree structure by jointly learning a contextual representation for tokens in a sentence, as well as a head–dependent scoring function. Whereas this strategy results in high performance, it is difficult to interpret these representations in relation to the geometry of the underlying tree structure. Our work seeks instead to learn interpretable representations by training a parser to explicitly preserve structural properties of a tree. We do so by casting dependency parsing as a tree embedding problem where we incorporate geometric properties of dependency trees in the form of training losses within a graph-based parser. We provide a thorough evaluation of these geometric losses, showing that the majority of them yield strong tree distance preservation as well as parsing performance on par with a competitive graph-based parser (Qi et al., 2018). Finally, we show where parsing errors lie in terms of tree relationship in order to guide future work.

1 Introduction

Dependency grammars are syntactic formalisms that represent the syntactic structure of a sentence as asymmetric binary grammatical relations among words (Tesnière, 1959; Hudson, 1984; Melcuk, 2003). An example dependency structure is given in Figure 1. Formally, a dependency structure is defined as a directed graph where words are vertices and relations are labelled directed edges (the *arcs*) between a child (the *dependent*) and its parent (the *head*). In practice, dependency structures considered for syntactic analysis are trees. Dependency trees have long been used to improve the performance of many NLP applications, including machine translation (Ding and Palmer, 2004; Menezes et al., 2010; Bastings et al., 2017), relation extraction (Kambhatla, 2004; Bunescu and Mooney, 2005; Miwa and Bansal, 2016; Zhang et al., 2018), and semantic role labeling (Hacioglu, 2004; Marcheggiani and Titov, 2017; He et al., 2018).

In order to assign the correct dependency tree to a sentence, dependency parsers are trained to correctly identify head–dependent relations between pairs of words. Modern neural approaches do so by jointly learning contextual feature representations for the tokens in a sentence, as well as a parsing decision function. This is the case for recent graph-based parsers (Zhang et al., 2017; Dozat et al., 2017; Mohammadshahi and Henderson, 2020, *inter alia*) where an encoder feature extractor is complemented by a score function predicting the likelihood of a word to be the head of another. However, whereas this joint learning strategy results in state-of-the-art performance, the representations learned by these parsers are opaque.

As a first step towards learning interpretable parser representations, here we take a different approach: in addition to learning-to-parse, we seek to learn representations from which tree distances between words in dependency trees can be recovered. This stems from one simple observation: previous approaches do not take into account the *geometry* of the tree they try to model. That is, parsers are unaware of the structural properties of the tree (e.g., distance between nodes, depth from root), and as such are not trained to *explicitly* preserve these properties. In this respect, our approach is aligned with recent work looking at these geometric properties in the context of *probing* the BERT (Devlin et al., 2019) embedding space for syntactic structure (Hewitt and Manning, 2019; Reif et al., 2019). In particular, Hewitt and Manning (2019)

---

[*] Equal contribution.

[†] Now at Borealis AI (`akos.kadar@borealisai.com`).

[‡] Now at University of Toronto (`kemertas@cs.toronto.edu`).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1684–1697
April 19 - 23, 2021.

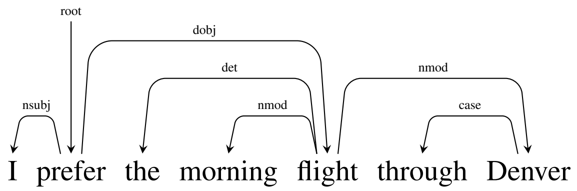


Figure 1: Example dependency tree.

have shown that it is possible to recover approximate syntactic trees from BERT embeddings by a linear transformation trained to minimize the difference between predicted and ground-truth tree distances. Given these results we then ask: is it possible to extend this idea to directly train tree-aware dependency parsers? We argue that using the geometric tree structure to embed the dependency trees enhances the interpretability of the learned representations

In this paper, we show that this is indeed possible by casting dependency parsing as a tree embedding problem. Specifically, we view a dependency tree as a finite metric space, and compute head–dependent scores for all word pairs within a sentence as follows: Given a sentence $s = (w_0, \ldots, w_{n_s})$, where w_0 is a special ROOT token, we compute a geometric tree embedding $\phi : \{w_0, \ldots, w_{n_s}\} \to \mathbb{R}^m$ that maps tokens w_i to m-dimensional vectors. Geometric properties of an ideal isometric tree embedding are used to define the functional form of head–dependent (edge) scores $\psi(w_i, w_j)$. Concretely, our approach predict dependency trees only from pairwise embedded node distances, which completely specify the score function. We consider this as a step in the direction of interpretable end-to-end dependency parsing. We start with a straight-forward application of Hewitt and Manning (2019) and consider a mean absolute error (MAE) loss that encourages the embedding ϕ to approximate an isometric embedding of the ground-truth tree T_s into $(\mathbb{R}^m, d_R)$. In this paper, we use the squared Euclidean distance semi-metric (d_2^2) as in Hewitt and Manning (2019) and also consider the distance obtained from the ℓ_1-norm (d_1)[1]. We show formally that any isometric d_2^2 embedding can be simply rotated to form an isometric d_1 embedding.

We learn the tree embedding ϕ and the edge score function ψ through end-to-end training, by incorporating geometric properties of dependency trees (in terms of distance and depth) in the form of geometric losses within a graph-based parser. As our base parser, we use a simplified version of the biaffine parser of Qi et al. (2018). This setup allows us to directly compare the performance of our losses and score functions to the biaffine score function used in several state-of-the-art graph parsers. We propose three additional losses for training a dependency parser expressed explicitly through tree distances: a maximum likelihood estimation function, a margin-based loss function, and one based on cross-entropy.

Finally, we explore whether adding a soft global constraint on the isometry of the learned trees helps with parsing performance; to this end, we combine our novel loss functions with the MAE loss.

We evaluate our approach on 16 languages from different families. We complement unlabeled accuracy of head–dependent attachement scores (UAS) with a Spearman's ρ correlation between predicted and true distances (DSpr) to directly assess geometric properties of the output trees. We also provide labeled attachment scores (LAS) for completeness. Through extensive experimentation, we make the following observations:

- All of our novel tree distance based losses outperform the MAE loss of Hewitt and Manning (2019)
- All losses using the d_1 metric provide better distance preservation properties and dependency parsing performance than using the d_2^2 semi-metric.
- Five of the six loss combinations (using d_1) show both strong distance preservation properties and parsing performance, indicating that distance preservation can be obtained without trading off parsing performance. Only the maximum likelihood estimation loss (on its own) has poor distance preservation; however, we find that the combination of this loss with the MAE loss greatly improves distance preservation, while achieving similar or better parsing performance.
- We show that the majority of parsing errors are local in tree distance, with by far the most frequent incorrect head assignments being either true sisters or grandparents.

---

[1] The squared Euclidean distance is a semi-metric, not a metric, as it does not respect the triangle inequality. Here $d_1(\mathbf{u}, \mathbf{v}) := ||\mathbf{u} - \mathbf{v}||_1$. Other distance functions could also be considered for d_R but that is beyond the scope of current work.

Our results in the direction of accurate dependency parser that closely preserve tree distances are encouraging.

2 Background: Metric Tree Embeddings

A tree $T = (V, E)$ along with the distance $d_T(u, v)$ is a finite metric space,[2] where $d_T(u, v)$ is the length of the shortest path between any $u, v \in V$. In this paper we consider tree embeddings, $\phi : V \to \mathbb{R}^m$, which map nodes $v \in V$ to points $\phi(v) \in \mathbb{R}^m$ such that the mapping ϕ approximately preserves tree distance. That is, for all pairs $u, v \in V$, the tree distance $d_T(u, v)$ is roughly the corresponding distance $d_R(\phi(u), \phi(v))$ in the embedding space $\mathbb{R}^m$, where we select the metric or semi-metric $d_R(x, y)$ in $\mathbb{R}^m$.

In this section, we discuss the choice of d_R and illustrate distortion free (i.e., isometric) embeddings $\phi : V \to \mathbb{R}^m$. These distortion free embeddings motivate the formulation of losses that we use for training suitable embeddings, as discussed in § 3.

To choose the distance measure d_R in the embedding space, we note that for a sufficiently large dimension m: i) any tree can be embedded isometrically into ℓ_1; ii) any metric space (including trees) can be embedded into ℓ_∞; and iii) for ℓ_p spaces, with $1 < p < \infty$, trees can only be embedded with distortion (Linial et al., 1995). The power transform of the Euclidean distance $d_2(\mathbf{x}, \mathbf{y})^c$, with $c \geq 2$, allows for isometric embedding of trees (Reif et al., 2019). However, the squared Euclidean distance $d_2(\mathbf{x}, \mathbf{y})^2$ does not satisfy the triangle inequality[3] and therefore is only a semi-metric. Nevertheless, both $(\mathbb{R}^m, d_1)$ and $(\mathbb{R}^m, d_2^2)$ are natural choices for embedding spaces for trees and, in this paper, we restrict our attention to these.

We follow Reif et al. (2019) to explicitly construct squared Euclidean embeddings. Specifically, all distortion free embeddings into $(\mathbb{R}^m, d_2^2)$ can be simply expressed in terms of the edge displacement vectors $\{\mathbf{z}_i\}_{i=1}^{|E|}$, where $\mathbf{z}_i \in \mathbb{R}^m$ is the displacement between the embedded endpoints of edge $e_i \in E$ (i.e., $\mathbf{z} := \phi(c) - \phi(p)$, where $c, p \in V$ are a pair of child and parent nodes). For an isometric embedding, it turns out we require these $\mathbf{z}_i$'s to be orthonormal, that is

$$\mathbf{Z}^T \mathbf{Z} = \mathbb{1}, \tag{1}$$

where $\mathbf{Z} \in \mathbb{R}^{m \times |E|}$ is the matrix having $\mathbf{z}_i$ as the i^{th} column and $\mathbb{1}$ denotes the $|E| \times |E|$ identity matrix. In addition, it is useful to define $\rho(u, v)$ to denote the shortest path between two vertices $u, v \in V$. And, finally, define the indicator vector $\mathbf{b}(u, v) \in \mathbb{Z}^{|E|}$ such that $b_i(u, v) = 1$ when edge e_i is on the shortest path between u and v, and $b_i(u, v) = 0$ otherwise. With this notation we have the following theorem:

Theorem 2.1. Pythagorean Embeddings. Given a rooted tree (V, E, r), where $r \in V$ denotes the root, then for any $m \geq |E|$ there exists an embedding $\phi : V \to \mathbb{R}^m$ such that,

$$\forall_{u,v \in V}\ d_T(u, v) = ||\phi(u) - \phi(v)||_2^2. \tag{2}$$

Moreover, ϕ satisfies (2) if and only if

$$\phi(v) = \phi(r) + \mathbf{Z}\,\mathbf{b}(r, v) \tag{3}$$

for some matrix $\mathbf{Z} \in \mathbb{R}^{m \times |E|}$ with orthonormal columns (i.e., Eqn. (1) is satisfied).

See Appendix B-D for an example that demonstrates this construction along with proofs.

The edge-on-path indicator vectors $\mathbf{b}(r, v)$ provide some additional intuition about these squared Euclidean embeddings. Specifically, for $v, w \in V$ we have

$$\begin{aligned}
g(v) := \mathbf{Z}^T \phi(v) &= \mathbf{Z}^T \phi(r) + \mathbf{b}(r, v), & (4)\\
g(v) - g(w) &= \mathbf{Z}^T \mathbf{Z}(\mathbf{b}(r, v) - \mathbf{b}(r, w)), &\\
&= \mathbf{b}(r, v) - \mathbf{b}(r, w) & (5)\\
||g(v) - g(w)||_1 &= ||\mathbf{b}(r, v) - \mathbf{b}(r, w)||_1 &\\
&= d_T(v, w) & (6)
\end{aligned}$$

Here (5) follows from (1), (3) and (4). Therefore g is an isometric ℓ_1 embedding which expresses tree distance in terms of the ℓ_1 norm of the difference between two path vectors $\mathbf{b}$. One interesting consequence of Theorem 2.1, along with Eqn.'s (4) and (6), is:

Corollary 2.1. ℓ_1 Embeddings. Given any isometric embedding $\phi : V \to \mathbb{R}^m$ using the squared Euclidean distance, the embedding $g(v) = \mathbf{Z}^T \phi(v)$ is an isometric embedding of the same tree (V, E, r) into $(\mathbb{R}^{|E|}, d_1)$. Here $\mathbf{Z}$ is as described in (3).

---

[2] Metric spaces are 2-tuples (X, d_X) consisting of a set of elements X and a metric $d_X : X \times X \mapsto [0, \infty)$ quantifying notion of distance between any pair of elements of X.

[3] Consider three points on a line with successive pairs separated by a d_2^2 distance of 1. Then the outer two are separated by an d_2^2 of $2^2 = 4$, which is larger than the sum of the distances between the successive pairs.

That is, any distance preserving tree embedding using the squared Euclidean norm can simply be rotated to an isometric ℓ_1 tree embedding.[4] Note the converse of Cor. 2.1 is not true. For example, three equally spaced points on a line form an ℓ_1 embedding that cannot be linearly transformed to a d_2^2 embedding. Indeed, it is shown in (Aksoy et al., 2020) that a finite metric tree can be embedded into $(\mathbb{R}^m, d_1)$ if and only if it has at most $2m$ leaves. Thus, for a fixed dimensional embedding space $\mathbb{R}^m$, the metric d_1 allows for more trees to be isometrically embedded than d_2^2.

In this paper we use an embedding dimension m that is larger than the number of edges and thus isometric tree embeddings are feasible using both d_2^2 and d_1. The learning problem considered in this paper is, given a sentence s, we seek to embed each of the sentence's tokens into $(\mathbb{R}^m, d_R)$ such that the embedded tree is nearly isometric to the dependency parse tree for s. Here we evaluate using both d_1 and d_2^2 for d_R.

3 Geometric Losses for Tree Embeddings

Given a sentence $s = (w_0, \dots, w_{n_s})$, where w_0 is a special ROOT token inserted at the beginning of every sentence, dependency parsing seeks to recover the correct dependency tree $T_s = (V_s, E_s)$. For simplicity, we label the tree nodes in V_s with the tokens themselves, so $V_s = \{w_0, \dots, w_{n_s}\}$. A geometric tree embedding maps tokens w_i within a sentence s to embedded points $\mathbf{v}_i = \phi(i, s) \in \mathbb{R}^m$ (for brevity, we drop the dependence on the whole sentence s and simply write $\phi(w_i)$). In § 2, we describe the exact geometry of the isometric embeddings using the d_2^2 semi-metric. For d_1 isometric embeddings we show one sub-class are simply rotations of isometric d_2^2 embeddings, but there are other forms. This section examines the use of auxiliary losses on the embedding ϕ that encourage approximately isometric embeddings. We expect an approximation of this geometry holds in d_2^2 when the losses are sufficiently small. Since we do not have a similar proof of necessity as in Appendix D for d_1 embeddings, the local geometry is more open.

Given such an embedding, we then follow first-order graph-based dependency parsers (McDonald et al., 2005) which compute a pairwise score $\psi(\mathbf{v}_i, \mathbf{v}_j)$ that indicates how likely it is for w_j to be the head of w_i. These scores provide edge weights on a fully connected embedded graph on $\phi(V_s) \times \phi(V_s)$. Having trained a network to compute suitable embeddings $\mathbf{v}_i = \phi(w_i)$ and edge weights $\psi(\mathbf{v}_i, \mathbf{v}_j)$, parsing then amounts to finding the maximum spanning tree in this weighted graph that is rooted at $\mathbf{v}_0$; a detailed description of the parser architecture is provided in § 4.

Given this general approach, a natural choice for an auxiliary loss on ϕ is to consider the mean absolute error (MAE) in the distances between the embeddings of any two nodes:

$$\mathcal{L}_{MAE}(\phi, T) = \frac{1}{|V|(|V|-1)} \sum_{\substack{w_i, w_j \in V, \\ i \neq j}} |\, d_R(\mathbf{v}_i, \mathbf{v}_j) - d_{T_s}(w_i, w_j)|, \quad (7)$$

where $\mathbf{v}_i = \phi(w_i)$ (we drop the subscripts s for brevity). This MAE loss treats the distance errors for all pair of nodes as equally important, and we therefore refer to it as a *global* loss. Note the loss in (7) is zero only when the embedding ϕ is an isometric tree embedding with respect to d_R.

We next consider the edge scoring function ψ, whose role is to assign costs to proposed head–dependent pairs (w_i, w_j). We denote this ground-truth head–dependent relation by $\varrho(w_i, w_j)$ (which is true when w_j is the head of w_i in T) and note it can be defined **only** in terms of the distance $d_T(w_i, w_j)$ and depth difference $\Delta_s(w_i, w_j) = d_T(w_0, w_i) - d_T(w_0, w_j)$ as follows:

$$\forall_{w_i, w_j \in V}\, \varrho(w_i, w_j) \iff d_T(w_i, w_j) = 1 \land \Delta(w_i, w_j) = 1. \quad (8)$$

Each node w_i then has a unique head w_j defined by (8), except for the `ROOT` w_0 which has none. Given the embedding ϕ is providing a near-isometric tree, we define an edge scoring function in the embedding space, namely $\psi(\mathbf{v}_i, \mathbf{v}_j)$, by rewriting (8) in the following probabilistic form:

$$\mathbf{v}_i := \phi(w_i), \quad (9)$$

$$\Delta_{ij}^{\phi} := d_R(\mathbf{v}_i, \mathbf{v}_0) - d_R(\mathbf{v}_j, \mathbf{v}_0), \quad (10)$$

$$\ell_{ij}^{\phi} := |d_R(\mathbf{v}_i, \mathbf{v}_j) - 1| + |\Delta_{ij}^{\phi} - 1|, \quad (11)$$

$$\psi(\mathbf{v}_i, \mathbf{v}_j) := -\ell_{ij}^{\phi}, \quad (12)$$

$$p_\phi(w_j | w_i) := \frac{e^{\psi(\mathbf{v}_i, \mathbf{v}_j)/\tau}}{\sum_{\substack{0 \le k \le n \\ k \neq i}} e^{\psi(\mathbf{v}_i, \mathbf{v}_k)/\tau}}. \quad (13)$$

where τ is a temperature parameter. Here we refer to ℓ_{ij}^{ϕ} as the "head–dependent cost" for the pair w_i and w_j.

---

[4] This rotation aligns the edge directions $\mathbf{z}_i$ with the standard bases vectors $\pm e_i$ of $\mathbb{R}^m$.

A natural loss on both the embedding $\phi : V \to \mathbb{R}^m$ and the edge-cost $\psi(\mathbf{v}_i, \mathbf{v}_j)$ is the maximum likelihood loss

$$\mathcal{L}_{MLE}(\phi, T) = \sum_{\substack{w_i, w_j \in V, \\ i \neq j}} \varrho(w_i, w_j) \left[-\log(p_\phi(w_j \mid w_i))\right] \tag{14}$$

where $p_\phi(w_j|w_i)$ is defined in (13).

As an alternative to the MLE loss we consider a margin based approach where the task is to minimize ℓ_{ij}^ϕ for the true head j, subject to $\ell_{ik}^\phi \geq \ell_{ij}^\phi + \alpha$ for all $k \in \{0, \ldots, n\} \backslash \{i, j\}$, where $\alpha > 0$ is a margin.[5] We explore such a margin-based approach using the soft triplet loss (Sohn, 2016)

$$\mathcal{L}_\alpha(\phi, T) = \tag{15}$$
$$\sum_{\substack{w_i, w_j \in V, \\ i \neq j}} \varrho(w_i, w_j) \log \left[1 + \sum_{w_k \in V \backslash \{w_j, w_i\}} e^{\alpha - \ell_{ik}^\phi + \ell_{ij}^\phi}\right].$$

As a third alternative for the head–dependent loss we consider the cross-entropy between the probability distribution $p_\phi(w_j|w_i)$ and the corresponding distribution $p_T(w_i|w_j)$ formed by using an isometric embedding ϕ_T in Eqns. (9 - 13). Note that for an isometric embedding ϕ_T we have $d_R(\mathbf{v}_i, \mathbf{v}_j) = d_T(w_i, w_j)$, and this can be used to simplify the resulting expression. Specifically, we find $p_T(w_j|w_i)$ depends only on the true tree distance d_T and not on the details of ϕ_T. The cross-entropy loss is then

$$\mathcal{L}_{CE}(\phi, T) = \sum_{\substack{w_i \in V \backslash \{w_0\}, \\ w_j \in V \backslash \{w_i\}}} p_T(w_j|w_i) \left[-\log(p_\phi(w_j|w_i))\right]. \tag{16}$$

In summary, we investigate the choice between several different head–dependent losses (i.e., Eqn.'s (14), (15), or (16), and optionally combine each of these with the explicit global MAE loss (7).

4 Evaluation

4.1 Model

We use a simplified version of the Biaffine dependency parser of Qi et al. (2018).[6] First, we give an overview of the Biaffine parser, and then describe our modifications. Biaffine is composed of a highway-BiLSTM encoder (Srivastava et al., 2015) that takes as input a sequence of $n_s + 1$ embeddings $\mathbf{x}_0, \ldots, \mathbf{x}_{n_s}$, where each $\mathbf{x}_i$ is a concatenation of word-level, character-level, part-of-speech and morphological feature embeddings. We use pre-trained word2vec (Mikolov et al., 2013) when available, and fastText embedding (Bojanowski et al., 2017) otherwise. We train the rest of the embeddings from scratch.

Given such an input sequence, the Biaffine parser predicts the most likely head for each word (referred to as unlabelled attachment prediction), along with the grammatical relation between each pair of head and dependent words (labelled attachment prediction). Biaffine first calculates contextual embeddings $\mathbf{h}_i$ (through the encoder), and then projects these into separate head and dependent representations for each word (through two separate MLP networks):

$$\mathbf{h}_i = \text{BiLSTM}_i(\mathbf{x}_0, \ldots, \mathbf{x}_{n_s}), \tag{17}$$
$$\mathbf{h}_i^{\text{head}}, \mathbf{h}_j^{\text{dep}} = \text{MLP}^{\text{head}}(\mathbf{h}_i), \text{MLP}^{\text{dep}}(\mathbf{h}_j), \tag{18}$$

where $\mathbf{h}_i^{\text{head}}$ and $\mathbf{h}_i^{\text{dep}}$ are the *head* and *dependent* representations. Next, for each pair of words w_i and w_j, a head–dependent score s_{ij} and a corresponding probability $p(w_j|w_i)$ are calculated with a learnable biaffine weight $\mathbf{U}$:

$$\begin{aligned} s_{ij} &= \mathbf{h}_i^{\text{head}} \mathbf{U} \mathbf{h}_j^{\text{dep}}, \\ &:= \text{Biaffine}(\mathbf{h}_i, \mathbf{h}_j), \end{aligned} \tag{19}$$
$$p(w_j|w_i) = \frac{e^{s_{ij}}}{\sum_{w_k \in V_s \backslash \{w_i\}} e^{s_{ik}}}. \tag{20}$$

Our geometric tree embedding ϕ computes a single representation for each node, as such we replace the separate head and dependent MLP networks with a single MLP network[7]:

$$\mathbf{v}_i := \phi(w_i), \tag{21}$$
$$= \text{MLP}(\mathbf{h}_i). \tag{22}$$

Given $\mathbf{v}_0, \ldots, \mathbf{v}_n$, head–dependent scores s_{ij} are defined as in:

$$s_{ij} = \psi(\mathbf{v}_i, \mathbf{v}_j), \tag{23}$$

where ψ is calculated as in Eqn. (12). We obtain asymmetry in our score function $\psi(\mathbf{v}_i, \mathbf{v}_j)$ from the depth difference term $|\Delta_{ij}^\phi - 1|$ in Eqn. (11).

---

[5] Note that an isometric embedding ϕ provides one solution for which $\ell_{ij}^\phi = 0$ for all head–dependent pairs, and $\ell_{ij}^\phi \geq 2$ otherwise.

[6] We use the codebase provided at `https://github.com/stanfordnlp/stanfordnlp`.

[7] To focus on learning unlabeled tree structures, we also remove the auxiliary losses that penalize rightward attachments or long dependencies, since the model behaves differently from the unlabeled prediction.

During inference, we use the Chu-Liu-Edmonds algorithm (Chi, 1999; Edmonds, 1967) to find the highest-scoring dependency tree. While our main focus is embedding unlabeled trees for dependency relation prediction, for completeness we also report results on labeled dependency tree prediction. We use the same classifier as Biaffine — with the same setting as described in Qi et al. (2018) — to estimate the probabilities of dependency labels $\mathbf{l}_{ij}$ for a given head–dependent pair w_i and w_j:

$$p(l_{k,ij}|w_j, w_i) = \frac{e^{s^{\text{rel}}_{k,ij}}}{\sum_n e^{s^{\text{rel}}_{n,ij}}} \tag{24}$$

$$\mathbf{s}^{\text{rel}}_{ij} = \text{Biaffine}^{\text{rel}}(\mathbf{h}_i, \mathbf{h}_j) \tag{25}$$

where $\mathbf{s}^{\text{rel}}_{ij} \in \mathbb{R}^K$ is a K-dimensional vector containing the dependency relation scores for each of the K dependency labels.

4.2 Experimental Setup

Data. We perform experiments on Universal Dependencies (UD; Nivre et al. 2020). Relying on criteria proposed by Kulmizev et al. (2019), we select 16 languages from different families, with different scrips and training sizes, all with good annotation quality. We use the following treebanks from UD v2.2: Arabic-PADT, Basque-BDT, Bulgarian-BTB, Chinese-GSD, Czech-PDT, Danish-DDT, English-EWT, Finnish-TDT, Hebrew-HTB, Hindi-HDTB, Italian-ISDT, Japanese-GSD, Korean-GSD, Russian-SynTagRus, Swedish-Talbanken, Turkish-IMST.

Performance Measures. We report the overall accuracy of head and relation predictions for all tokens in the test portion of the data sets. Given our model prediction and a reference parse for a given input, accuracy is calculated using two standard measures: Unlabelled Attachment Score (UAS), that is the percentage of tokens that are assigned the correct head; and Labelled Attachment Score (LAS) that is the percentage of tokens that are assigned the correct head and the correct grammatical relation. We use UAS for model selection.

To assess how well the learned tree embeddings preserve distances, we follow Hewitt and Manning (2019) and Hall Maudslay et al. (2020), and measure the correlation between the learned and ground-truth tree distances. Specifically, for all words in all sentences, we compute Spearman's ρ between predicted and true distances. We first average the correlation coefficients for sentences of the same length. We report the macro-average over these averages for sentences of length 5–50, referred to as DSpr.

	d_2^2	d_1	d_1 [+ $\mathcal{L}_{MAE}$]
$\mathcal{L}_{MAE}$	90.28 (0.89)	90.53 (0.89)	N/A
$\mathcal{L}_{MLE}$	91.47 (0.60)	91.51 (0.71)	91.77 (0.85)
$\mathcal{L}_{\alpha}$	91.43 (0.83)	91.57 (0.88)	91.41 (0.89)
$\mathcal{L}_{CE}$	86.82 (0.38)	91.79 (0.81)	91.85 (0.88)

Table 1: UAS (DSpr) of d_2^2 and d_1 embeddings, plus effects of adding the auxiliary MAE loss; reported on the development portion of English-EWT.

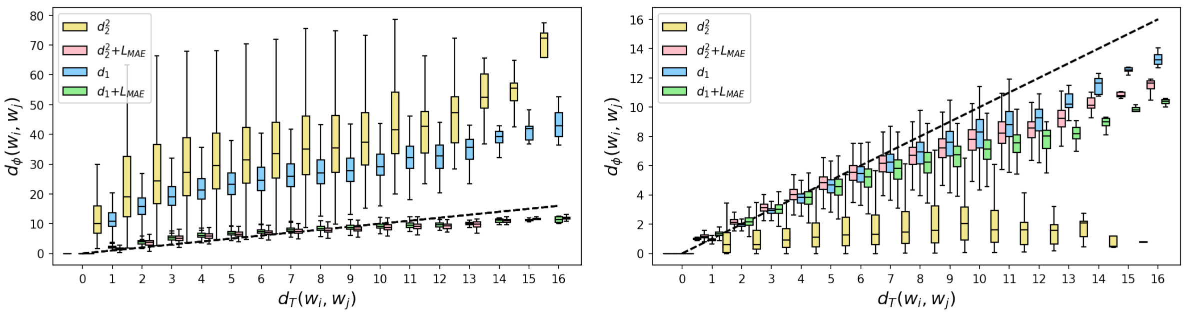
Hyperparameters. We adopt the same hyperparameter configuration as in the original Biaffine model (Qi et al., 2018) up to the BiLSTM layer for the head–dependent classifier, and the same configuration for the entire dependency label classifier. We perform grid search on the remaining hyperparameters and select best hyperparameter configurations based on UAS on the development portion of the English-EWT data. Based on the results, we set the margin $\alpha = 3$ for $\mathcal{L}_\alpha$, and the temperature $\tau = 1$ for $\mathcal{L}_{MLE}$ and $\tau = 0.2$ for $\mathcal{L}_{CE}$. In our evaluation, we run experiments that involve the combination of any of $\mathcal{L}_\alpha$, $\mathcal{L}_{CE}$, and $\mathcal{L}_{MLE}$ with $\mathcal{L}_{MAE}$ as an auxiliary loss, with a coefficient λ_1. We find the best values for λ_1 to be 0.2 for $\mathcal{L}_\alpha$, and 0.1 for both $\mathcal{L}_{CE}$ and $\mathcal{L}_{MLE}$. We refer the reader to Appendix A for the full list of hyperparameters and training details.

5 Results

5.1 Metric Spaces and Geometric Losses

We first verify whether the choice of metric space impacts performance. Table 1 reports UAS (and DSpr) on the development portion of the English-EWT corpus, for tree embeddings in both $(\mathbb{R}^m, d_1)$ and $(\mathbb{R}^m, d_2^2)$ metric spaces (referred to as d_1 and d_2^2 for brevity, respectively).

The first row shows results when trained with $\mathcal{L}_{MAE}$ only. It learns an embedding that provides good approximation of global tree distances using d_2^2 (DSpr: 0.89), which is similar to the findings reported by Hewitt and Manning (2019), but is suboptimal in terms of parsing (UAS: 90.28). On the other hand we can see that all head–dependent losses achieve stronger parsing performances in both spaces, with the proper metric d_1 leading to higher UAS and DSpr scores across the board (including $\mathcal{L}_{MAE}$) when compared to d_2^2. We will then report results in the rest of this section only



(a) (b)

Figure 2: Distance distributions for (a) $\mathcal{L}_{MLE}$ and (b) $\mathcal{L}_{CE}$, for different metric spaces (d_2^2/d_1) and trained with or without the auxiliary loss $\mathcal{L}_{MAE}$. The four box plots at each x-tick summarize the distribution of predicted distances for a particular ground-truth distance. Each box covers data distribution between 25 to 75 percentile, with median showing by the black line. The dashed line represents perfect correlation.

with the d_1 metric, unless otherwise stated.

Table 1 also shows a comparison between the different head–dependent losses, with and without the auxiliary loss $\mathcal{L}_{MAE}$ that further constrains ϕ to be globally isometric. In isolation, $\mathcal{L}_{CE}$ yields the best UAS while $\mathcal{L}_{\alpha}$ the best DSpr; interpolating the losses with $\mathcal{L}_{MAE}$ improves results for $\mathcal{L}_{MLE}$ and $\mathcal{L}_{CE}$, especially in terms of DSpr. This is in line with our expectations that the auxiliary loss encourages the parser to learn an embedding that more faithfully preserves global tree distances.

$\mathcal{L}_{MLE}$ seeks to correctly identify all head–dependent relations by maximizing the probability $p_\phi(w_j|w_i)$ of true head–dependent pairs with no explicit constraints on global isometry; see Eqn. (14). We thus hypothesize that $\mathcal{L}_{MLE}$ can learn an embedding that produces good UAS, but is far from being isometric to the ground-truth tree. We verify this empirically: Figure 2(a) shows that the addition of $\mathcal{L}_{MAE}$ greatly regulates the embedding distances of the trees produced by $\mathcal{L}_{MLE}$, hence improving the DSpr score for this loss.[8] $\mathcal{L}_{CE}$ on the other hand seeks to match $p_\phi(w_j|w_i)$ with $p_T(w_j|w_i)$, which encourages all embedding distances to be correlated with tree distances; see Eqn. (16). However, the $p_T(w_j|w_i)$ term in this loss gives higher weights to word pairs that are closer in terms of ground-truth tree distance and therefore the model is trained to focus more on preserving short distances.[9] Figure 2(b) confirms that the embedding obtained using $\mathcal{L}_{CE}$ underestimates the ground-truth tree distances, and adding the auxiliary MAE loss helps regulate this distortion of tree distances.

§ 2 describes the exact geometry of the embeddings that out model learns using the d_2^2 semi-metric in the special case that $\mathcal{L}_{MAE}$ is reduced to zero. In Figure 2(b), we observe small losses up to tree distance five when d_2^2 is used with $\mathcal{L}_{CE}$ and $\mathcal{L}_{MAE}$. Therefore we expect the local geometry of d_2^2 trees to approximately follow Eqn. (3).

Overall, for both losses we observe that, on average, d_2^2 embeddings lead to predicted tree distances that have large variances, as well as medians further away from the ground-truth, whereas the d_1 embeddings are more stable and accurate. This agrees with the observations in Table 1 that embeddings in d_1 have better DSpr than embeddings in d_2^2 for all losses we considered. The same comparison between d_ϕ and d_T for $\mathcal{L}_\alpha$ is provided in Appendix E.

5.2 Comparison with Qi et al. (2018)

We compare the parsing results for the six geometric loss combinations against the biaffine parser of Qi et al. (2018) for all treebanks in Table 2. For a fair comparison, we re-run all experiments and report our results for the biaffine parser. We report UAS for models trained without the dependency label prediction loss in order to focus on unlabeled tree structure and LAS for completeness. In general, the parsing performance is stable across

---

[8] We observe that the optimized $\mathcal{L}_{MLE}$ (without MAE) is lower than that produced by an isometric embedding on the development portion of English-EWT, indicating that the observed distortion of tree distances is an overfitting issue.

[9] The local emphasis is stronger with lower temperature. We chose temperature $\tau = 0.2$ based on UAS on the development portion of English-EWT. However, we find a temperature of 1 to provide better approximation of global distances, especially for tree distances less than five (see Appendix E and Figure 6).

	Biaffine		$\mathcal{L}_{MLE}$		$\mathcal{L}_{MLE+MAE}$		$\mathcal{L}_{\alpha}$		$\mathcal{L}_{\alpha+MAE}$		$\mathcal{L}_{CE}$		$\mathcal{L}_{CE+MAE}$	
Arabic	**87.70**	**84.83**	86.75	83.44	86.60	82.96	86.16	82.93	85.81	82.65	86.01	83.14	86.82	82.77
Basque	**89.03**	**87.45**	88.01	86.08	87.75	86.17	87.49	85.97	87.57	86.05	87.14	85.92	87.61	86.03
Bulgarian	**94.97**	92.44	94.06	92.06	94.58	92.06	93.97	91.97	94.18	91.54	94.22	92.12	94.51	**92.45**
Chinese	**86.73**	**86.16**	85.51	84.54	86.29	84.75	85.92	84.82	85.79	85.11	85.95	85.23	85.79	85.22
Czech	92.43	90.92	93.18	92.04	93.10	91.72	93.40	91.54	93.24	91.97	93.29	91.64	**93.42**	**92.12**
Danish	**88.23**	**87.70**	87.08	86.31	87.51	86.40	87.11	86.45	87.26	86.50	87.08	86.55	87.91	87.30
English	**91.21**	**90.15**	90.71	89.37	90.81	89.57	90.82	89.66	90.81	89.63	90.78	89.69	90.87	89.86
Finnish	**91.55**	**91.29**	90.54	90.06	90.51	90.24	90.14	89.60	90.32	89.73	90.55	90.02	90.85	90.75
Hebrew	**91.00**	**90.28**	90.07	89.14	90.11	88.90	89.14	88.46	89.16	88.94	89.50	88.95	90.29	89.51
Hindi	96.78	94.80	96.55	94.62	96.55	94.35	96.65	94.44	96.40	94.58	96.40	94.58	**96.81**	**94.69**
Italian	**94.14**	**92.98**	93.56	92.62	93.35	92.27	93.21	92.49	93.27	92.34	93.64	92.26	93.30	92.51
Japanese	**96.03**	**95.69**	95.43	94.94	95.41	94.83	95.14	95.16	95.14	94.87	94.77	94.97	95.53	95.50
Korean	**88.84**	**86.75**	87.80	86.24	87.98	85.78	87.59	85.66	87.73	85.97	87.03	85.72	88.28	86.40
Russian	**93.86**	**93.22**	93.16	92.61	93.37	92.17	93.06	91.80	92.97	91.96	93.50	91.99	92.93	92.61
Swedish	**91.52**	**90.16**	90.06	89.07	90.52	89.13	90.85	89.74	90.61	89.33	90.74	89.67	91.12	89.58
Turkish	73.14	**70.25**	72.69	69.16	72.85	67.80	72.18	68.48	72.61	69.14	70.34	68.15	**73.45**	67.34
Average	**90.45**	**89.07**	89.70	88.29	89.83	88.07	89.55	88.07	89.56	88.04	89.43	88.16	90.18	88.42

Table 2: UAS (left) and LAS (right) on the test set of 16 treebanks in the UD dataset, plus the macro averages over all treebanks. We mark the highest performing system for both UAS and LAS in **bold** and second highest in blue.

	d_1	d_1 [$+\mathcal{L}_{MAE}$]
$\mathcal{L}_{MLE}$	0.72 ± 0.03	0.85 ± 0.02
$\mathcal{L}_{CE}$	0.89 ± 0.03	0.86 ± 0.05
$\mathcal{L}_{\alpha}$	0.87 ± 0.03	0.89 ± 0.03

Table 3: Average DSpr over all (test) treebanks.

different languages for all geometric losses. Overall, $\mathcal{L}_{CE+MAE}$ is our best performing model and the average UAS/LAS across languages are on par with the Biaffine parser in spite of only having a single representation for each token. Moreover, it achieves top performance on Czech, Hindi and Turkish. All other geometric losses also achieve competitive results that are within 1% of the Biaffine parser for both UAS and LAS.

We also report the mean DSpr along with standard deviation across the 16 treebanks in Table 3. Unlike the UAS and LAS we find a substantial difference in the DSpr score for the loss $\mathcal{L}_{MLE}$. When combined with the auxiliary loss $\mathcal{L}_{MAE}$ we find a pronounced increase in DSpr, this agrees with the findings in § 5.1.

5.3 Parsing Errors w.r.t. Tree Relationships

Inspired by the geometric structure of tree embeddings, we investigate the sources of errors in terms of ground-truth dependency trees. Given a sentence s and a dependency tree T_s, we define the type of relation between a pair of token (w_i, w_j) by a 2-tuple consisting the distance $d_{T_s}(w_i, w_j)$ and depth difference $\Delta_s(w_i, w_j) = d_{T_s}(w_0, w_i) - d_{T_s}(w_0, w_j)$. This definition follows naturally from the geometric interpretation of trees: for a node w_i, (1, -1) defines its children, (2, 0) defines its sisters, and (2, 2) defines its grandparents.

To visualize the distribution of errors, for each trained model, we plot the percentage of wrong edges for each relation type on the development set of English-EWT. We show an example plot in Figure 3 for our best $\mathcal{L}_{CE+MAE}$, with results for other losses provided in Appendix F.

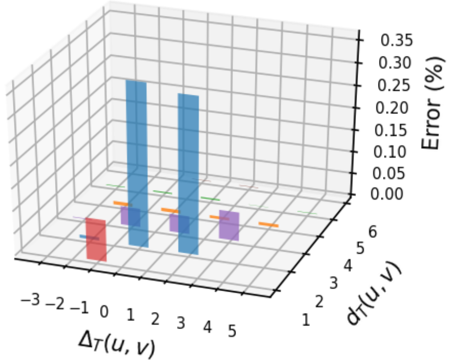


Figure 3: Distribution of edge errors on the development portion of English-EWT, optimized with $\mathcal{L}_{CE+MAE}$; the two axes are $(d_T(i, v), \Delta_T(u, v))$.

Surprisingly, we do not identify long distance ambiguities as a major source of errors (i.e., 99.8% of UAS errors have incorrectly assigned a head node that is within a tree distance of 5 from the dependent). Moreover, we find that sisters and grandparents account for 36.2% and 34.8% of all the UAS errors, respectively. To put these results into context, we construct a synthetic random error distribution: for each tree in the English-EWT development set we generate all trees with a single attachment error. We observe 81.6% of the errors to be local (up to distance 5) and sisters account for 22.5% and grandparents only account for 5.4% of the errors. We further compare with the biaffine parser and find 99.5% of UAS errors are local with 40.2% for sister and 32.7% for grandparent errors. Therefore parsing errors for a trained parser are dominated by errors that are more local in terms of tree distance than expected from a uniform error distribution. One immediate question that may arise is how can we reduce these specific high-frequency errors. One intuitive extension of the current work is to modify the formulation of edge scores in Eqn. (12) to push the decision boundary away from sisters or grandparents during inference. By training to explicitly model the geometry of the tree, our approach is one step closer towards addressing specific high-frequency errors.

6 Conclusions

In this work, we propose to use the geometry of (dependency) tree structures to construct a neural dependency parser that improves the interpretability of the learned representations without compromising parsing performance. We propose several geometric loss functions, and show that for a majority of them, our simple network learns distance-preserving embeddings through end-to-end training. In doing so, we also compare squared Euclidean distance (d_2^2) with the distance obtained from ℓ_1-norm (d_1), as the (semi-)metric in the embedding space $\mathbb{R}_m$, and provide empirical evidence for using the proper d_1 metric. We compare our results with a competitive and widely-used graph-parser proposed by Qi et al. (2018) on 16 languages from different families, and show overall parser performances that are on par with it. Our experiments also suggest a new way of looking at the sources of parsing errors in terms of tree distances, and show that the majority of errors are local (e.g., sisters or grandparents).

For future work, we suggest looking at potential ways to correct such high-frequency head prediction errors, defined by their relationship within a tree. Another interesting direction that's worth exploring is to use the continuous tree distances predicted by our methods as features for downstream tasks instead of the discrete tree structures produced by conventional parsers. As recent work has been exploring, this differentiable representation of tree structure is potentially useful within the iterative-refinement framework (Mohammadshahi and Henderson, 2020), or as additional tree-specific positional features in a transformer (Omote et al., 2019).

Acknowledgments

We thank three anonymous reviewers for their useful comments. Research was conducted at Samsung AI Centre Toronto and funded by Samsung Research, Samsung Electronics Co.,Ltd

References

Asuman Aksoy, Mehmet Kili, and Sahin Koak. 2020. Isometric embeddings of finite metric trees into $(\mathbf{R}^n, d_1)$ and $(\mathbf{R}^n, d_\infty)$. *arXiv preprint arXiv:2002.00062*.

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Z. Chi. 1999. Statistical properties of probabilistic context-free grammars. *Computational Linguistics*, 25:131–160.

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

Yuan Ding and Martha Palmer. 2004. Synchronous dependency insertion grammars: A grammar formalism for syntax based statistical MT. In *Proceedings of the Workshop on Recent Advances in Dependency Grammar*, pages 90–97, Geneva, Switzerland. COLING.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Kadri Hacioglu. 2004. Semantic role labeling using dependency trees. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1273–1276, Geneva, Switzerland. COLING.

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A tale of a probe and a parser. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7389–7395, Online. Association for Computational Linguistics.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071, Melbourne, Australia. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Richard A Hudson. 1984. *Word grammar*. Blackwell Oxford.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 178–181, Barcelona, Spain. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.

Nathan Linial, Eran London, and Yuri Rabinovich. 1995. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Igor Melcuk. 2003. Levels of dependency in linguistic description: Concepts and problems. *Dependency and Valency. An International Handbook of Contemporary Research*, 1:188–229.

Arul A Menezes, Christopher B Quirk, and Colin A Cherry. 2010. Machine translation system incorporating syntactic dependency treelets into a statistical framework. US Patent 7,698,124.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119. Curran Associates, Inc.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Alireza Mohammadshahi and James Henderson. 2020. Recursive non-autoregressive graph-to-graph transformer for dependency parsing with iterative refinement. *Transactions of the Association for Computational Linguistics(under submission)*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Yutaro Omote, Akihiro Tamura, and Takashi Ninomiya. 2019. Dependency-based relative positional encoding for transformer NMT. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 854–861, Varna, Bulgaria. INCOMA Ltd.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. 2018. Universal dependency parsing from scratch. In *CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of adam and beyond. In *International Conference on Learning Representations*.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems*, volume 32, pages 8594–8603. Curran Associates, Inc.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, volume 29, pages 1857–1865. Curran Associates, Inc.

Rupesh Kumar Srivastava, Klaus Greff, , and Jürgen Schmidhuber. 2015. Highway networks. In *Proceedings of the Deep Learning Workshop at the International Conference on Machine Learning*.

Lucien Tesnière. 1959. *Eléments de Syntaxe Structurale*. Klincksieck, Paris.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 665–676, Valencia, Spain. Association for Computational Linguistics.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, Brussels, Belgium. Association for Computational Linguistics.

A Hyperparameters

We use 2 layers of MLP with leaky-relu as activation to map the biLSTM outputs into a 800-dimensional embedding space[10]. The layer weights are initialized with the values from uniform distribution $\mathcal{U}(-0.05, 0.05)$, and biases are initialized to zero.

We train the models with Adam (Kingma and Ba, 2015) with an initial learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and $\epsilon = 1e-8$ for up to $50,000$ iterations, where each iteration is a batch of up to 5000 tokens or the maximum number of tokens we can fit in the GPU memory. We evaluate the models every 100 steps and save them only if we see improvement in UAS on development data. We switch to AMSGrad (Reddi et al., 2018) after 3000 iterations with no observed improvement on development set UAS, at which point we terminate training when another 3000 iterations pass without improving development set UAS.

B Example d_2^2 and d_1 Tree Embeddings

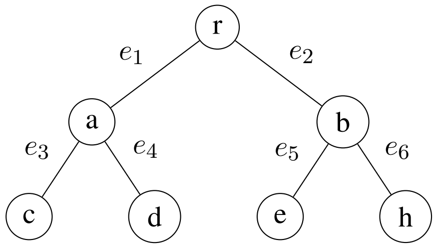


Figure 4: Simple binary tree example.

Let us take as a working example the binary tree on Figure 4. The embedding of node v is the embedding of the root r plus the sum of all direction vectors on the path $\rho(r, v)$, from r to v. If can take $f(r)$ to be **0** or any random vector then embedding of all nodes in the tree are by definition:

$$f(a) = f(r) + \mathbf{u}_1$$
$$f(c) = f(r) + \mathbf{u}_1 + \mathbf{u}_3$$
$$f(d) = f(r) + \mathbf{u}_1 + \mathbf{u}_4$$
$$f(b) = f(r) + \mathbf{u}_2$$
$$f(e) = f(r) + \mathbf{u}_2 + \mathbf{u}_5$$
$$f(h) = f(r) + \mathbf{u}_2 + \mathbf{u}_6$$

We took an arbitrary root embedding $f(r)$ and $n-1$ orthogonal unit-length vectors $\mathbf{u}_i$. If the $\mathbf{u}_i$-s are the standard unit basis vectors, for example, then the tree is embedded on the edges of the unit cube and it is an isometric embedding for both d_2^2 and d_1.

C Proof of Thm. 2.1, Sufficiency

Proof. **Eqn. (3) sufficient.** We first show that an embedding of the form given in Eqn. (3) necessarily satisfies (2). This is Thm. 1 in Reif et al. (2019) and here we find it useful to expand on their proof to introduce notation and assist the reader.

Let $T = (V, E, r)$, **Z** and $b(r, u)$ be as described in § 2. Then Eqn. (3) is simply

$$f(v) = f(r) + \sum_{\{i \in \rho(r,v)\}} \mathbf{z}_i, \tag{26}$$

where the notation $\{i \in \rho(r, v)\}$ is short for $\{i \mid e_i \in \rho(r, v)\}$, which is the set of i's for which $b_i(r, v) = 1$. Consider any two vertices v and w. Notice the two paths $\rho(r, v)$ and $\rho(r, w)$ must share a common prefix, namely $\rho(r, a)$, the sub-path from r to the lowest common ancestor a of v and w, with the remaining paths $\rho(a, v)$ and $\rho(a, w)$ being edge disjoint. Therefore

$$\begin{aligned} &||f(v) - f(w)||_2^2 \\ &= || \sum_{\{i \in \rho(r,v)\}} \mathbf{z}_i - \sum_{\{j \in \rho(r,v))\}} \mathbf{z}_j ||_2^2 && (27) \\ &= || \sum_{\{i \in \rho(a,v)\}} \mathbf{z}_i - \sum_{\{j \in \rho(a,w)\}} \mathbf{z}_j ||_2^2 && (28) \\ &= |\rho(a, v)| + |\rho(a, w)| && (29) \\ &= |\rho(v, w)| \equiv d_T(v, w). && (30) \end{aligned}$$

Here we canceled the common term $f(r)$ and common prefix edges to derive Eqn. (27) and (28). Eqn. (29) follows from the fact that paths $\rho(a, v)$ and $\rho(a, w)$ are edge disjoint and the orthonormality of the $\mathbf{z}_i$'s. Here $|\rho(a, v)|$ denotes the number of edges on the a to v path. Finally, (30) follows since a is the least common ancestor of v and w.

□

D Proof of Thm. 2.1, Necessity

We provide a proofs that any isometric d_2^2 embedding must have the form defined in (3) that relies only on linear algebra and may provide the reader with additional intuition about the construction of d_2^2 embeddings.

---

[10] The maximum length of any sentence in the dataset is smaller than 800 and thus an isometric tree embeddings are feasible using both d_2^2 and d_1

Proof. First, for the case $|V| \leq m+1$, we use induction to show that any isometric d_2^2 embedding f must have the form described in Eqn. (3). Then we show that for $|V| > m+1$, such an f cannot exist.

We use the notation $V = \{v_j \mid 0 \leq j < k\}$, where $v_0 = r$, $|V| = k$ and $|E| = k-1$. For $2 \leq k \leq m+1$ define the the $|E| \times |E|$ path matrix B to be

$$\mathbf{B} = (\mathbf{b}(r, v_1), \ldots, \mathbf{b}(r, v_{k-1})), \tag{31}$$

where $\mathbf{b}(r,v)$ are defined to be the binary path vectors used in (3).

We use induction to prove that if $f : V \to \mathbb{R}^m$ is an isometric embedding for any $k \leq m+1$ then (3) must hold and $\mathbf{B}$ must be full rank. Note this statement is trivially true for $k = 1$ and 2.

Let $k \geq 2$ and $k \leq m$. Let $T = (V, E, r)$ be a tree of size $|V| = k$. Consider the induction hypothesis that any isometric embedding (using d_2^2) of T must have the form in (3) and, moreover, the matrix $\mathbf{B}$ in (31) has full rank. We next show that it follows the same is true for any tree of size $k+1$.

Suppose $T' = (V', E', r)$ is a rooted tree with $|V'| = k+1 \leq m+1$. Suppose $f : V' \to \mathbb{R}^m$ is an isometric embedding using d_2^2. Let $c \neq r$ be a leaf and p be its parent in T'[11], and consider the subtree $T = (V, E, r)$ formed by removing the vertex c and the edge (c, p) from the tree. So $|V| = k$, and the restriction of f to V must provide an isometric embedding of this subtree T. By the induction hypothesis this embedding f, when restricted to V, must satisfy (3) and have the form in (3). Moreover the path matrix $\mathbf{B}$ in (31) for T must be full rank.

Since the full embedding f of T' (including the new vertex c) is isometric it follows that

$$\begin{aligned} ||f(c) - f(v)||_2^2 &= d_T(c, v), \\ &= d_T(p, v) + 1, \ \forall v \in V. \end{aligned} \tag{32}$$

Specifically, when $v = p$ this constrains the embedded displacement for the (p, c) edge, namely $\mathbf{z} := f(c) - f(p)$, to be length 1. Moreover, for any $v \in V' \setminus \{p, c\}$, consider the triangle with vertices $f(c)$, $f(p)$, and $f(v)$. Then (32) states that the squared Euclidean distances of the triangle edges $(f(c), f(p))$, $(f(p), f(v))$ and $(f(c), f(v))$, must be 1, $d_T(p, v)$, and $d_T(p, v) + 1$, respectively. By the Pythagorean Theorem, this must be a right triangle with

$$\mathbf{z}^T (f(p) - f(v)) = 0, \forall v \in V' \backslash \{p, c\}.$$

By using Eqn. (3) on the subtree T this can be rewritten as

$$\mathbf{z}^T \mathbf{Z} (\mathbf{b}(r, p) - \mathbf{b}(r, v)) = 0, \forall v \in V' \backslash \{p, c\}. \tag{33}$$

By setting $v = r$ in (33) and using $\mathbf{b}(r, r) = \mathbf{0}$ (i.e., null vector in $\mathbb{Z}^{|E|}$) we find $\mathbf{z}^T \mathbf{Z} \mathbf{b}(r, p) = 0$. So (33) is equivalent to

$$\mathbf{z}^T \mathbf{Z} \mathbf{b}(r, v) = 0, \forall v \in V' \backslash \{c\}. \tag{34}$$

That is $z^T \mathbf{Z} B = 0$. However, by the induction hypothesis, the $(k-1) \times (k-1)$ matrix $\mathbf{B}$ (for T) is full rank, and so we must have $\mathbf{z}^T \mathbf{Z} = \mathbf{0}$. That is, the direction of the edge (p, c) in the embedding space, $\mathbf{z} = f(c) - f(p)$, must be perpendicular to all the other edges in the tree T' and have unit length, so

$$\begin{aligned} \mathbf{z}^T \mathbf{Z} &= \mathbf{0}, \\ ||\mathbf{z}||_2 &= 1. \end{aligned} \tag{35}$$

Since $\mathbf{Z}$ is a $m \times (k-1)$ matrix with $k \leq m$ it follows that such $\mathbf{z}$ must exist.

Note the path vector $\mathbf{b}(r, c)$, in the tree T', is just the path vector for the parent, $\mathbf{b}(r, p)$, modified to have a 1 in the k^{th} row, while all other path vectors $\mathbf{b}(r, v_j)$ must have zero in this k^{th} row (since c is a leaf). The path matrix $\mathbf{B}'$ for T' therefore has the form

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & \mathbf{b}(r, p) \\ \mathbf{0} & \mathbf{1} \end{pmatrix}. \tag{36}$$

Since the induction hypothesis ensures that $\mathbf{B}$ has full rank, it follows that $\mathbf{B}'$ has full rank. Finally, defining $\mathbf{Z}' = (\mathbf{Z}\mathbf{z}) \in \mathbb{R}^{m \times k}$, it follows from (35) that $(\mathbf{Z}')^T \mathbf{Z}' = \mathbb{1}$ and therefore we have shown that f must have the form in (3) with this $\mathbf{Z}'$ and these path vectors $\mathbf{b}(r, v)$. This completes the proof of the induction step, the desired result follows by induction.

To show that there is no embedding for $|V'| \geq m+2$ it is sufficient to show that there is no embedding for $|V'| = m+2$. We use contradiction. Suppose $f : V' \to \mathbb{R}^m$ is such an embedding. We can proceed as above, with c a leaf node of $T' = (V', E', r)$, and p its parent. Then, by using the previous result, any isometric embedding of the subtree $T = (V, E, r)$ that is formed by removing

---

[11] Note $c \neq r$ is possible since $k \geq 2$

this child (so $|V| = m + 1$) must have the form described in (3) with a full rank $m \times m$ path matrix $\mathbf{B}$ and orthonormal $m \times m$ matrix $\mathbf{Z}$. The same line of reasoning then shows that $z := f(c) - f(p)$ must satisfy (35). But here $\mathbf{Z}$ is full rank and so $z = 0$, which contradicts the constraint that the (c, p) edge in the embedding must have length 1. $\square$

E Distance plots for learned embedding

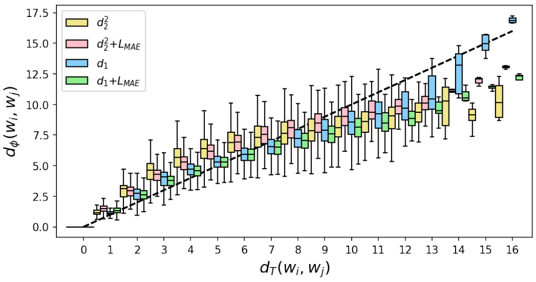


Figure 5: Distance distribution for triplet loss in d_2^2, d_1 and d_1 with MAE on dev set of English-EWT.

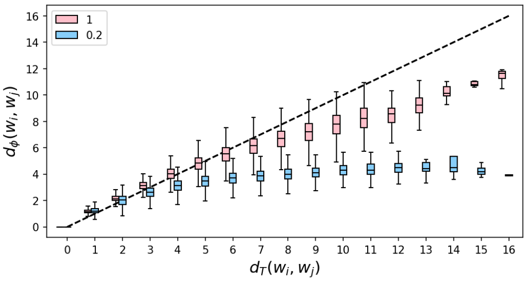
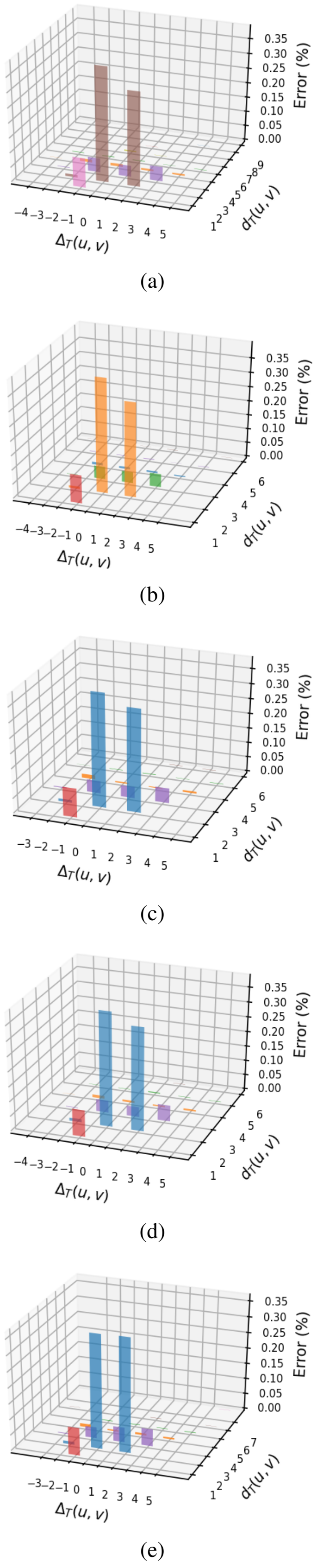


Figure 6: Distance distribution for CE loss using d_1 for different softmax temperature on dev set of English-EWT.

F Error distribution trained with different losses



(a)

(b)

(c)

(d)

(e)

Figure 7: Distribution of edge errors for (a) $\mathcal{L}_{MLE}$ (b) $\mathcal{L}_{MLE+MAE}$ (c) $\mathcal{L}_\alpha$ (d) $\mathcal{L}_{\alpha+MAE}$ and (e) $\mathcal{L}_{CE}$ on dev set of English-EWT.

Active Learning for Sequence Tagging with Deep Pre-trained Models and Bayesian Uncertainty Estimates

Artem Shelmanov[1], **Dmitri Puzyrev**[1,2,3], **Lyubov Kupriyanova**[1], **Denis Belyakov**[2],
Daniil Larionov[2,3,4], **Nikita Khromov**[1], **Olga Kozlova**[3], **Ekaterina Artemova**[2],
Dmitry V. Dylov[1], **Alexander Panchenko**[1]
[1]Skolkovo Institute of Science and Technology, Russia
[2]Higher School of Economics University, Russia
[3]Mobile TeleSystems (MTS), Russia
[4]Federal Research Center "Computer Science and Control" of Russian Academy of Sciences
{a.shelmanov,dmitrii.puzyrev,lyubov.kupriyanova,
nikita.khromov,d.dylov,a.panchenko}@skoltech.ru
{dobelyakov,dslarionov}@edu.hse.ru, elartemova@hse.ru
oskozlo9@mts.ru

Abstract

Annotating training data for sequence tagging of texts is usually very time-consuming. Recent advances in transfer learning for natural language processing in conjunction with active learning open the possibility to significantly reduce the necessary annotation budget. We are the first to thoroughly investigate this powerful combination for the sequence tagging task. We conduct an extensive empirical study of various Bayesian uncertainty estimation methods and Monte Carlo dropout options for deep pre-trained models in the active learning framework and find the best combinations for different types of models. Besides, we also demonstrate that to acquire instances during active learning, a full-size Transformer can be substituted with a distilled version, which yields better computational performance and reduces obstacles for applying deep active learning in practice.

1 Introduction

In many natural language processing (NLP) tasks, such as named entity recognition (NER), obtaining gold standard labels for constructing the training dataset can be very time and labor-consuming. It makes the annotation process expensive and limits the application of supervised machine learning models. This is especially the case in such domains as biomedical or scientific text processing, where crowdsourcing is either difficult or prohibitively expensive. In these domains, highly-qualified experts are needed to annotate data correctly, which dramatically increases the annotation cost.

Active Learning (AL) is a technique that can help to reduce the amount of annotation required to train a good model by multiple times (Settles and Craven, 2008; Settles, 2009). Opposite to exhaustive and redundant manual annotation of the entire corpus, AL drives the annotation process to focus the expensive human expert time only on the most informative objects, which contributes to a substantial increase in the model quality.

AL is an iterative process that starts from a small number of labeled seeding instances. In each iteration, an *acquisition* model is trained on the currently annotated dataset and is applied to the large pool of unannotated objects. The model predictions are used by the AL query strategy to sample the informative objects, which are then further demonstrated to the expert annotators. When the annotators provide labels for these objects, the next iteration begins. The collected data can be used for training a final *successor* model that is used in a target application.

During AL, acquisition models have to be trained on very small amounts of the labeled data, especially during the early iterations. Recently, this problem has been tackled by transfer learning with deep pre-trained models: ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), ELECTRA (Clark et al., 2020), and others. Pre-trained on a large amount of unlabeled data, they are capable of demonstrating remarkable performance when only hundreds or even dozens of labeled training instances are available. This trait suits the AL framework but poses the question about the usefulness of the biased sampling provided by the AL query strategies.

In this work, we investigate AL with the aforementioned deep pre-trained models and compare the results of this combination to the outcome of

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1698–1712
April 19 - 23, 2021.

the models that do not take advantage of deep pretraining. The main contributions of this paper are the following:

- We are the first to thoroughly investigate deep pre-trained models in the AL setting for sequence tagging of natural language texts on the widely-used benchmarks in this area.
- We conduct an extensive empirical study of various AL query strategies, including Bayesian uncertainty estimation methods with multiple Monte Carlo (MC) dropout variants (Gal and Ghahramani, 2016a; Gal et al., 2017). We find the best combinations of uncertainty estimates and dropout options for different types of deep pre-trained models.
- We show that to acquire instances during AL, a full-size Transformer can be substituted with a distilled version, which yields better computational performance and reduces obstacles for applying deep AL in practice.

The remainder paper is structured as follows. Section 2 covers relevant works on AL for sequence tagging. In Section 3, we describe the sequence tagging models. Section 4 describes the AL strategies used in the experiments. In Section 5, we discuss the experimental setup and present the evaluation results. Finally, Section 6 concludes the paper.

2 Related Work

AL for sequence tagging with classical machine learning algorithms and a feature-engineering approach has a long research history, e.g. (Settles and Craven, 2008; Settles, 2009; Marcheggiani and Artières, 2014). More recently, AL in conjunction with deep learning has received much attention.

In one of the first works on this topic, Shen et al. (2018) note that practical deep learning models that can be used in AL should be computational efficient both for training and inference to reduce the delays in the annotators' work. They propose a CNN-CNN-LSTM architecture with convolutional character and word encoders and an LSTM tag decoder, which is a faster alternative to the widely adopted LSTM-CRF architecture (Lample et al., 2016) with comparable quality. They also reveal disadvantages of the standard query strategy – least confident (LC), and propose a modification, namely Maximum Normalized Log-Probability (MNLP). Siddhant and Lipton (2018) experiment with Bayesian uncertainty estimates. They use CNN-CNN-LSTM and CNN-BiLSTM-CRF (Ma and Hovy, 2016) networks and two methods for calculating the uncertainty estimates: Bayes-by-Backprop (Blundell et al., 2015) and the MC dropout (Gal and Ghahramani, 2016a). The experiments show that the variation ratio (Freeman, 1965) has substantial improvements over MNLP. In contrast to them, we additionally experiment with the Bayesian active learning by disagreement (BALD) query strategy proposed by Houlsby et al. (2011) and perform a comparison with variation ratio.

There is a series of works that tackle AL with a trainable policy model that serves as a query strategy. For this purpose, imitation learning is used in Liu et al. (2018); Vu et al. (2019); Brantley et al. (2020), while in (Fang et al., 2017), the authors use deep reinforcement learning. Although the proposed solutions are shown to outperform other heuristic algorithms with comparably weak models (basic CRF or BERT without fine-tuning) in experiments with a small number of AL iterations, they can be not very practical due to the high computational costs of collecting training data for policy models. Other notable works on deep active learning include (Erdmann et al., 2019), which proposes an AL algorithm based on a bootstrapping approach (Jones et al., 1999) and (Lowell et al., 2019), which concerns the problem of the mismatch between a model used to construct a training dataset via AL (acquisition model) and a final model that is trained on it (successor model).

Deep pre-trained models are evaluated in the AL setting for NER by Shelmanov et al. (2019). However, they perform the evaluation only on the specific biomedical datasets and do not consider the Bayesian query strategies. Ein-Dor et al. (2020) conduct an empirical study of AL with pre-trained BERT but only on the text classification task. Brantley et al. (2020) use pre-trained BERT in experiments with NER, but they do not fine-tune it, which results in suboptimal performance. In this work, we try to fill the gap by evaluating deep pre-trained models: ELMo and various Transformers, in the AL setting with practical query strategies, including Bayesian, and on the widely-used benchmarks in this area.

3 Sequence Tagging Models

We use a tagger based on the Conditional Random Field model (Lafferty et al., 2001), two BiLSTM-

CRF taggers (Lample et al., 2016) with different word representation models, and taggers based on state-of-the-art Transformer models.

3.1 Conditional Random Field

As a baseline for comparison, we use a feature-based linear-chain Conditional Random Field (CRF) model (Lafferty et al., 2001). It is trained to maximize the conditional log-likelihood of entire tag sequences. The inference is performed using the Viterbi decoding algorithm, which maximizes the joint probability of tags of all tokens in a sequence. The features used for the CRF model are presented in Appendix B.

3.2 BiLSTM-CRF

This model encodes embedded input tokens via a bidirectional long short term memory neural network (LSTM) (Hochreiter and Schmidhuber, 1997). BiLSTM processes sequences in two passes: from left-to-right and from right-to-left producing a contextualized token vector in each pass. These vectors are concatenated and are used as features in a CRF layer that performs the final scoring of tags.

We experiment with two versions of the BiLSTM-CRF model. The first one uses GloVe (Pennington et al., 2014) word embeddings pre-trained on English Wikipedia and the 5-th edition of the Gigaword corpus, and a convolutional character encoder (Ma and Hovy, 2016), which helps to deal with out-of-vocabulary words. As in (Chiu and Nichols, 2016), the model additionally leverages the basic capitalization features, which has been shown to be useful for achieving good performance with this model. We will refer to it as CNN-BiLSTM-CRF. We consider this model as another baseline that does not exploit deep pre-training. The second version of the BiLSTM-CRF model uses pre-trained medium-size ELMo (Peters et al., 2018) to produce contextualized word representations. ELMo is a BiLSTM language model enhanced with a CNN character encoder. This model does not rely on feature-engineering at all. We will refer to it as ELMo-BiLSTM-CRF.

3.3 Transformer-based Taggers

We perform AL experiments with state-of-the-art pre-trained Transformers: BERT (Devlin et al., 2019), DistilBERT (Sanh et al., 2019), and ELECTRA (Clark et al., 2020). The sequence tagger, in this case, consists of a Transformer "body" and a decoding classifier with one linear layer. Unlike BiLSTM that encodes text sequentially, these Transformers are designed to process the whole token sequence in parallel with the help of the self-attention mechanism (Vaswani et al., 2017). This mechanism is bi-directional since it encodes each token on multiple neural network layers taking into account all other token representations in a sequence. These models are usually faster than the recurrent counterparts and show remarkable performance on many downstream tasks (Li et al., 2020).

BERT is a masked language model (MLM). Its main learning objective is to restore randomly masked tokens, so it can be considered as a variant of a denoising autoencoder. Although this objective makes the model to learn many aspects of natural languages (Tenney et al., 2019; Rogers et al., 2020), it has multiple drawbacks, including the fact that training is performed only using a small subset of masked tokens. ELECTRA has almost the same architecture as BERT but utilizes a novel pre-training objective, called replaced token detection (RTD), which is inspired by generative adversarial networks. In this task, the model has to determine what tokens in the input are corrupted by a separate generative model, in particular, a smaller version of BERT. Therefore, the model has to classify all tokens in the sequence, which increases training efficiency compared to BERT, and the RTD task is usually harder than MLM, which makes the model learn a better understanding of a language (Clark et al., 2020).

DistilBERT is a widely-used compact version of BERT obtained via a distillation procedure (Hinton et al., 2015). The main advantages of this model are the smaller memory consumption and the higher fine-tuning and inference speed achieved by sacrificing the quality. We note that good computational performance is a must for the practical applicability of AL. Delays in the interactions between a human annotator and an AL system can be expensive. Therefore, although DistilBERT is inferior compared to other Transformers in terms of quality, it is a computationally cheaper alternative for acquiring training instances during AL that could be used for fine-tuning bigger counterparts. Lowell et al. (2019) showed that a mismatch between an acquisition model and a successor model (the model that is trained on the annotated data for the final application) could eliminate the benefits of AL. The similarity in the architectures and the shared knowledge between the smaller distilled Transformer and its ancestor potentially can help to alleviate this problem and deliver an AL solution that is both effective and practical.

4 Active Learning Query Strategies

We experiment with four query strategies for the selection of training instances during AL.

Random sampling is the simplest query strategy possible: we just randomly select instances from the unlabeled pool for annotation. In this case, there is no active learning at all.

Uncertainty sampling (Lewis and Gale, 1994) methods select instances according to some probabilistic criteria, which indicates how uncertain the model is about the label that was given to the instance. The baseline method is Least Confident (LC): the instances are sorted in the ascending order of probabilities of the most likely tag sequence. Let y_i be a tag of a token i that can take one class c out of C values, let x_j be a representation of a token j in an input sequence of length n. Then the LC score can be formulated as follows:

$$LC = 1 - \max_{y_1,\dots,y_n} \mathbb{P}\left[y_1,\dots,y_n \middle| \{\mathbf{x}_j\}\right]$$

This score favors longer sentences since long sentences usually have a lower probability. Maximization of probability is equivalent to maximizing the sum of log-probabilities:

$$\max_{y_1,\dots,y_n} \mathbb{P}\left[y_1,\dots,y_n|\{\mathbf{x}_j\}\right] \Leftrightarrow$$
$$\Leftrightarrow \max_{y_1,\dots,y_n} \sum_i^n \log \mathbb{P}\left[y_i|\{y_j\} \setminus y_i, \{\mathbf{x}_j\}\right]$$

To make LC less biased towards longer sentences, Shen et al. (2018) propose a normalization of the log-probability sum. They call the method ***Maximum Normalized Log-Probability (MNLP)***. The MNLP score can be expressed as follows:

$$MNLP = -\max_{y_1,\dots,y_n} \frac{1}{n} \sum_i^n \log \mathbb{P}\left[y_i|\{y_j\} \setminus y_i, \{\mathbf{x}_j\}\right]$$

In our experiments, we use this normalized version of the uncertainty estimate since it has been shown to be slightly better than the classical LC (Shen et al., 2018), and it is commonly applied in other works on active learning for NER.

Following Siddhant and Lipton (2018), we implement extensions for the Transformer-based and BiLSTM-based sequence taggers applying the MC dropout technique. Gal and Ghahramani (2016a) showed that applying a dropout at the prediction time allows us to consider the model as a Bayesian neural network and calculate theoretically-grounded approximations of uncertainty estimates by analyzing its multiple stochastic predictions. Like Shen et al. (2018) and Siddhant and Lipton (2018) we experiment with ***variation ratio (VR)*** (Freeman, 1965): a fraction of models, which predictions differ from the majority:

$$VR_i = 1 - \frac{\text{count}\left(\text{mode}\left(\{y_i^m\}_m^M\right), \{y_i^m\}_m^M\right)}{M}$$
$$VR = \frac{1}{n}\sum_i^n VR_i,$$

where M is a number of stochastic predictions.

Siddhant and Lipton (2018) and Shen et al. (2018) refer to this method as BALD. However, the ***Bayesian active learning by disagreement (BALD)*** proposed by Houlsby et al. (2011) leverage mutual information between outputs y_i and model parameters θ trained on a dataset $\mathcal{D}$:

$$BALD_i = H(y_i|x_i, \mathcal{D}) - E_{\theta \sim p(\theta|\mathcal{D})}\left[H(y_i|x_i, \theta)\right]$$

Let p_i^{cm} be a probability of a tag c for a token i that is predicted by a m-th stochastic pass of a model with the MC dropout. Then the BALD score can be approximated according to the following expression (Gal et al., 2017):

$$BALD_i \approx$$
$$-\sum_c^C \left(\frac{1}{M}\sum_m^M p_i^{cm}\right) \log\left(\frac{1}{M}\sum_m^M p_i^{cm}\right)$$
$$+\frac{1}{M}\sum_{c,m}^{C,M} p_i^{cm} \log p_i^{cm}$$
$$BALD \approx \frac{1}{n}\sum_i^n BALD_i$$

Although BALD can be considered similar to VR, it potentially can give better uncertainty estimates than VR since it leverages the whole probability distributions produced by the model. However, this method has not been tested in the previous works on active learning for sequence tagging.

For Transformer models, we have two variants of the Monte Carlo dropout: the MC dropout on the last layer before the classification layer (MC last), and on all layers (MC all). We note that calculating uncertainty estimates in the case of MC all requires multiple stochastic passes, and in each of them, we have to perform inference of the whole Transformer model. However, if we replace the dropout only in the classifier, multiple recalculations are needed only for the classifier, while it is enough to perform the inference of the massive Transformer "body" only once. Therefore, in this case, the overhead of calculating the Bayesian uncertainty estimates can be less than 1% (in the case of 10 stochastic passes for ELECTRA according to the number of parameters) compared to deterministic strategies like MNLP.

The BiLSTM-CRF model has two types of dropout: the word dropout that randomly drops entire words after the embedding layer and the locked dropout (Gal and Ghahramani, 2016b) that drops the same neurons in the embedding space of a recurrent layer for a whole sequence. Therefore, for the BiLSTM-CRF taggers, we have three options: replacing the locked dropout (MC locked), replacing the word dropout (MC word), and replacing both of them (MC all). We should note that obtaining Bayesian uncertainty estimates does not require the recalculation of word embeddings (in our case, ELMo).

5 Experiments and Results

5.1 Experimental Setup

We experiment with two widely-used datasets for evaluation of sequence tagging models and AL query strategies: English CoNLL-2003 (Sang and Meulder, 2003) and English OntoNotes 5.0 (Pradhan et al., 2013). The corpora statistics are presented in Appendix A. In the experiments, we use the "native" tag schemes: IOB1 for CoNLL-2003 and IOB2 for OntoNotes 5.0.

Each experiment is an *emulation* of the AL cycle: selected instances are not presented to experts for annotation but are labeled automatically according to the gold standard. Each experiment is performed for each AL query strategy and is repeated five times for CoNLL-2003 and three times for OntoNotes to report the standard deviation. A random 2% subset in tokens of the whole training set is chosen for seeding, and instances with 2% of tokens in total are selected for annotation on each iteration. Overall, 24 AL iterations are made, so in the final iteration, half of the training dataset (in tokens) is labeled. We do not use validation sets provided in the corpora but keep 25% of the labeled corpus as the development set and train models from scratch on the rest. For OntoNotes, the AL query strategy is applied to only the 50% subsample of the unlabeled dataset. This helps to reduce the duration of the inference phase. In preliminary experiments, we note that it does not seriously affect the AL performance, but the further reduction of the subsample size has a significant negative impact.

MODEL	CONLL-2003	ONTONOTES
CRF	78.2 ± NA	79.8 ± NA
CNN-BILSTM-CRF	88.3 ± 0.2	82.9 ± 0.3
ELMO-BILSTM-CRF	91.2 ± 0.2	87.2 ± 0.2
DISTILBERT	89.8 ± 0.2	87.3 ± 0.1
BERT	91.1 ± 0.2	88.2 ± 0.2
ELECTRA	91.5 ± 0.2	87.6 ± 0.2

Table 1: Performance of models built on the entire training datasets without active learning

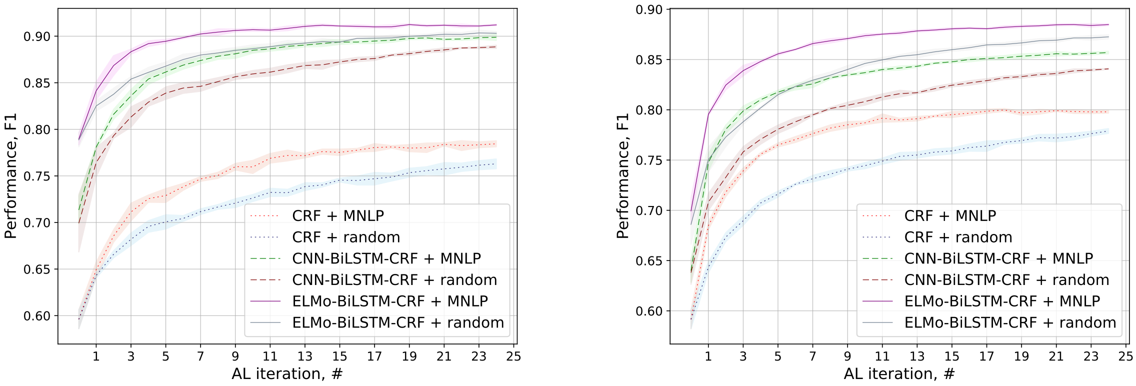
Details of models and training procedures are presented in Appendix C. We conduct AL experiments with the pre-selected model and training hyperparameters. Tuning hyperparameters of acquisition models during AL would drastically increase the duration of an AL iteration, which makes AL impractical. The hyperparameter optimization is reasonable for successor models. However, in preliminary AL experiments, tuning hyperparameters of successor models on the development set appeared to demonstrate an insignificant difference in the performance compared to using pre-selected hyperparameters that are fixed for all AL iterations. Therefore, since tuning hyperparameters for training a successor model on each AL iteration drastically increases the amount of computation, we fix them to the values pre-selected for experiments without AL.

The evaluation is performed using the span-based F1-score (Sang and Meulder, 2003). For query strategies based on the MC dropout, we make ten stochastic predictions.

5.2 Results and Discussion

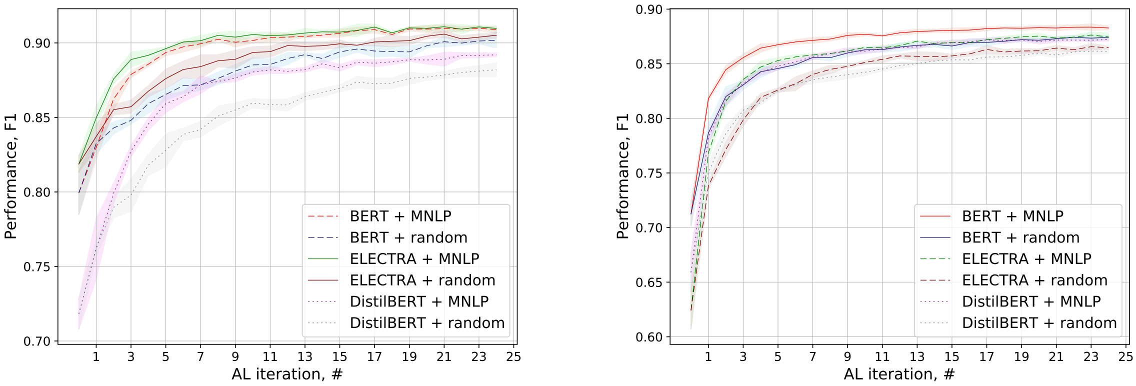
5.2.1 Training on Entire Datasets

We evaluate the performance of models trained on 75% of the available training corpus while keeping the rest 25% as the development set similarly to the experiments with AL. From Table 1, we can find that for both OntoNotes and CoNLL-2003, the model performance pattern is almost the same.



a) CoNLL-2003 b) OntoNotes

The comparison of MNLP and random query strategies for CRF and BiLSTM-based models.



c) CoNLL-2003 d) OntoNotes

The comparison of MNLP and random sampling query strategies for Transformers.

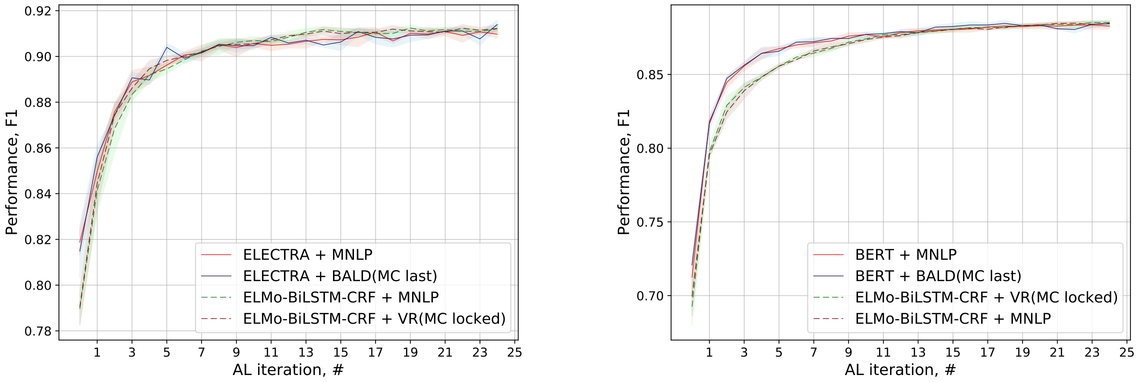
Figure 1: The comparison of MNLP and random query strategies

CRF, as the baseline model, has the lowest F1 score. Sequence taggers based on deep pre-trained models achieve substantially higher results compared to the classical CNN-BiLSTM-CRF model and CRF. BERT and ELECTRA significantly outperform ELMo-BiLSTM-CRF on OntoNotes, while on CoNLL-2003, all models have comparable scores. DistilBERT is behind larger Transformers. It also has a lower performance than ELMo-BiLSTM-CRF on CoNLL-2003 but similar scores on the OntoNotes dataset. We should note that our goal was not to show the state-of-the-art performance on each dataset but to determine reasonable hyperparameters and reference scores for experiments with AL.

5.2.2 Active Learning

The main results of experiments with AL are presented in Figures 1–3. AL shows significant improvements over the random sampling baseline for all models and on both datasets. Performance gains are bigger for simpler models like CRF or CNN-BiLSTM-CRF without deep pre-training and for the more complex OntoNotes dataset. However, we see that both ELMo-BiLSTM-CRF and Transformers benefit from a biased sampling of AL query strategies, which magnifies their ability to be trained on extremely small amount of labeled data. For example, to get 99% of the score achieved with training on the entire CoNLL-2003 dataset, only 20% of the annotated data is required for the ELECTRA tagger accompanied with the best AL strategy. For OntoNotes and BERT, only 16% of the corpus is required. Random sampling requires more than 44% and 46% of annotated data for CoNLL-2003 and OntoNotes correspondingly.

MNLP strategy. For the CoNLL-2003 corpus, the best performance in AL with the MNLP query strategy is achieved by the ELECTRA model. It



a) CoNLL

b) OntoNotes

Figure 2: The comparison of the best query strategies and models overall

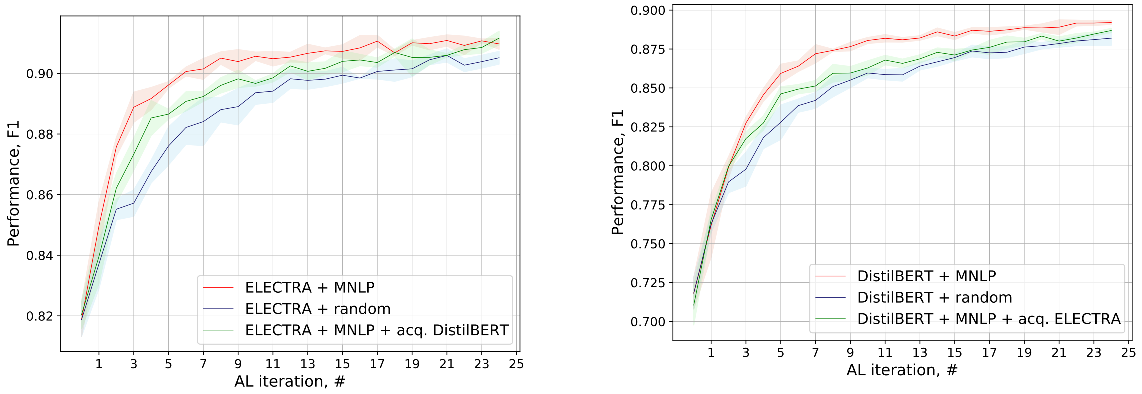
shows significantly better results in the beginning compared to BERT (see Figure 1c). ELECTRA also slightly outperforms the ELMo-BiLSTM-CRF tagger in the beginning and is on par with it on the rest of the AL curve (see Figure 2a). The CNN-BilSTM-CRF model is always better than the baseline CRF but worse than the models that take advantage of deep pre-training (see Figure 1a). DistilBERT appeared to be the weakest model in the experiments on the CoNLL-2003 dataset. With random sampling, it is on par with the baseline CNN-BiLSTM-CRF model on most iterations, but with the MNLP strategy, DistilBERT significantly falls behind CNN-BiLSTM-CRF.

Although BERT is slightly worse than the ELECTRA and ELMo-BiLSTM-CRF taggers in the experiments on the CoNLL-2003 corpus, for OntoNotes, BERT has a significant advantage over them up to 2.3 % of the F1 score in the AL setting on early iterations. The ELMo-BiLSTM-CRF tagger falls behind the main Transformers on the OntoNotes corpus. This might be because the BiLSTM-based tagger is underfitted to the bigger corpus with only 30 training epochs. Likewise, the baseline CNN-BiLSTM-CRF model without deep pre-training significantly falls behind DistilBERT on this corpus for MNLP and random query strategies.

Bayesian active learning. Bayesian uncertainty estimates based on the MC dropout perform comparably with the deterministic MNLP strategy for Transformer-based and BiLSTM-CRF-based taggers (see Figure 2 and Tables 5, 6 in Appendix D). We consider that Bayesian uncertainty estimates do not outperform the deterministic uncertainty estimates because the performance of the latter is very close to the maximum that can be achieved with the given amount of data.

We compare the performance of Bayesian AL strategies, when different dropout layers are replaced with the MC dropout. For ELMo-BiLSTM-CRF, we compare three options: replacing the dropout that follows word embeddings (embeddings acquired from ELMo), locked dropout in the recurrent layer, and both. Replacing the dropout that follows the embedding layer degrades the performance of AL significantly, especially for BALD. Replacing both yields the same performance as replacing only the locked dropout. We consider that the latter option is the best for AL since it requires fewer changes to the architecture. Overall, for both datasets, variation ratio has a slight advantage over the BALD strategy for the ELMo-BiLSTM-CRF model for all MC dropout options.

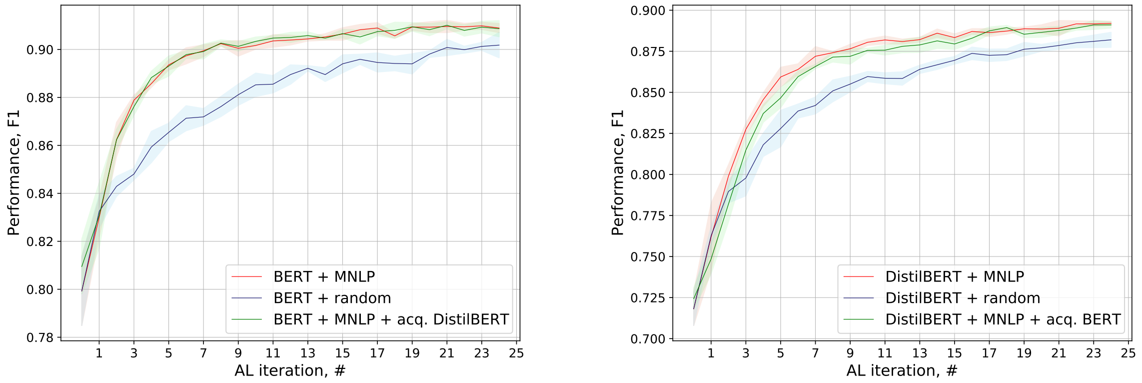
For Transformers, we compare two options: replacing the dropout only on the last classification layer and all dropouts in the model. When the variation ratio strategy is used, replacing only the last dropout layer with the MC dropout degrades the performance compared to MNLP, while replacing all dropout layers shows comparable results with it. However, for the BALD strategy, we see the inverse situation: replacing the last dropout layer leads to the significantly better performance than replacing all layers. This pattern can be noted for both ELECTRA and DistilBERT on the CoNLL-2003 corpus and for both BERT and DistilBERT on the OntoNontes corpus. Therefore, for Transformers, BALD with the MC dropout on the last layer is the best Bayesian query strategy since it



a) DistilBERT is an acquisition model

b) ELECTRA is an acquisition model

DistilBERT/ELECTRA as acquisition/successor models.



c) DistilBERT is an acquisition model

d) BERT is an acquisition model

DistilBERT/BERT as acquisition/successor models.

Figure 3: AL experiments on the CoNLL-2003 dataset, in which a successor model does not match an acquisition model.

Table 2: Duration of active learning iteration phases (seconds). The presented values correspond to the 5-th iteration of AL on the CoNLL-2003 dataset with the MNLP query strategy. The duration is averaged over three runs.

MODEL	ACQ. MODEL TRAINING	QUERYING INST.	TOTAL
CNN-BILSTM-CRF	1052	68	1120
ELMO-BILSTM-CRF	339	100	439
ELECTRA	100	50	150
DISTILBERT	61	27	88

provides both good quality and low computational overhead.

Duration of AL iterations. The valuable benefit of deep pre-trained models is also their high inference and training speed, which helps to significantly reduce the duration of an AL iteration and makes it feasible to implement text annotation tools empowered with interactive deep active learning. We measure the duration of the acquisition model training phase and the duration of the instance querying phase, which includes model inference on the whole unlabeled dataset (see Table 2). The experiments were conducted using the Nvidia V100 GPU and Xeon E5-2698 CPU.

According to obtained results, the large ELECTRA model can be trained more than ten times faster than the basic CNN-BiLSTM-CRF model because of a hardware-optimized architecture, a smaller number of the necessary training epochs, and the absence of the validation on the development set. ELECTRA also helps to reduce the duration of the query selection by more than 26%. The lightweight DistilBERT model is expectedly even faster. We also should note that using BiLSTM-

CRF with the pre-trained contextualized word representation model ELMo also helps to make the duration of AL iterations shorter compared to CNN-BiLSTM-CRF due to no need for training of the CNN word representation subnetwork (ELMo usually is not fine-tuned) and caching of the forward passes of ELMo during training across different epochs.

Mismatch between a successor model and an acquisition model. Since AL creates substantial computational overhead caused by training and inference of an acquisition model, for practical usage, it is reasonable to keep an acquisition model as lightweight as possible, while retaining the successor model complex and full-fledged. However, Lowell et al. (2019) demonstrate that the mismatch between an acquisition model and a successor model can diminish the benefits of applying the AL. They show that using AL can even harm the performance of the successor model compared to random sampling. We investigate the situation when an acquisition model does not match a successor model. Figure 3 shows the results of BERT, DistilBERT, and ELECTRA in this setting.

From Figures 3a and 3b, we can find that, when DistilBERT is used as an acquisition model for ELECTRA and vice versa, improvements achieved due to AL over the random sampling baseline are substantially lower compared to the "native" acquisition model. The same negative effect can be seen due to a mismatch between ELECTRA and BERT models in Figure 4 in Appendix E. These results support the findings presented in (Lowell et al., 2019). However, in our case, AL still gives notable improvements over the random sampling baseline. From Figure 3d, we can see that using BERT as an acquisition model for DistilBERT results only in a slight reduction of performance compared to using the "native" acquisition model. Moreover, in a reversed experiment, when DistilBERT is used as an acquisition model (see Figure 3c), there is no performance drop in AL at all. Such results for BERT-DistilBERT can be explained by the relationship between the distilled model and its ancestor resulting in similar uncertainty estimates for unlabeled instances in the annotation pool. This finding reveals the possibility of replacing a big acquisition model such as BERT with a distilled version that is faster and requires much less amount of memory. This can help to alleviate practical obstacles of deploying AL in real-world scenarios.

6 Conclusion

In this work, we investigated the combination of AL with sequence taggers that take advantage of deep pre-trained models. In the AL setting, these sequence taggers substantially outperform the models that do not use deep pre-training. We show that AL and transfer learning is a very powerful combination that can help to produce remarkably performing models with just a small fraction of the annotated data. For the CoNLL-2003 corpus, the combination of the best performing pre-trained model and AL strategy achieves 99% of the score that can be obtained with training on the full corpus, while using only 20% of the annotated data. For the OntoNotes corpus, one needs just 16%.

We performed a large empirical study of AL query strategies based on the Monte Carlo dropout in conjunction with deep pre-trained models and are the first to apply Bayesian active learning by disagreement to sequence tagging tasks. Bayesian active learning by disagreement achieves better results than the variation ratio for Transformers. However, we find that the variation ratio is slightly better for the ELMo-BiLSTM-CRF model. It is reasonable to use both MC dropout-based query strategies when only the last dropout layer works in a stochastic mode during the inference. This makes this type of query strategies suitable for practical usage due to little computational overhead. Finally, we demonstrate that it is possible to reduce the computational overhead of AL with deep pre-trained models by using a smaller distilled version of a Transformer model for acquiring instances.

In the future work, we are seeking to extend the empirical investigation of deep pre-trained models in active learning to query strategies that aim at better handling the batch selection of instances for annotation.

Acknowledgments

We thank the reviewers for their valuable feedback. This work was done in the framework of the joint MTS-Skoltech lab. The development of the experimental setup for the study of active learning methods and its application to sequence tagging tasks (Section 4) was supported by the Russian Science Foundation grant 20-71-10135. The Zhores supercomputer (Zacharov et al., 2019) was used for computations. We are very grateful to Sergey Ustyantsev for help with the implementation and the execution of the experiments.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1613–1622. JMLR.org.

Kianté Brantley, Hal Daumé III, and Amr Sharaf. 2020. Active imitation learning with noisy guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2093–2105, Online. Association for Computational Linguistics.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pretraining text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

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

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. 2019. Practical, efficient, and customizable active learning for named entity recognition in the digital humanities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2223–2234, Minneapolis, Minnesota. Association for Computational Linguistics.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Copenhagen, Denmark. Association for Computational Linguistics.

Linton C Freeman. 1965. *Elementary applied statistics: for students in behavioral science*. John Wiley & Sons.

Yarin Gal and Zoubin Ghahramani. 2016a. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059.

Yarin Gal and Zoubin Ghahramani. 2016b. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 29, pages 1019–1027. Curran Associates, Inc.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1183–1192. PMLR.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745.

Rosie Jones, Andrew McCallum, Kamal Nigam, and Ellen Riloff. 1999. Bootstrapping for text learning tasks. In *IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, volume 1.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 3-6 July 1994 (Special Issue of the SIGIR Forum)*, pages 3–12. ACM/Springer.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning how to actively learn: A deep imitation learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1883, Melbourne, Australia. Association for Computational Linguistics.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 21–30, Hong Kong, China. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Diego Marcheggiani and Thierry Artières. 2014. An experimental comparison of active learning strategies for partially labeled sequences. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 898–906, Doha, Qatar. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using

OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *CoRR*, abs/2002.12327.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. pages 142–147.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Burr Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1070–1079. Association for Natural Language Processing.

Artem Shelmanov, Vadim Liventsev, Danil Kireev, Nikita Khromov, Alexander Panchenko, Irina Fedulova, and Dmitry V. Dylov. 2019. Active learning with deep pre-trained models for sequence tagging of clinical and biomedical texts. In *2019 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2019, San Diego, CA, USA, November 18-21, 2019*, pages 482–489. IEEE.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *International Conference on Learning Representations*.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Thuy-Trang Vu, Ming Liu, Dinh Phung, and Gholamreza Haffari. 2019. Learning how to active learn by dreaming. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4091–4101, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's Transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Igor Zacharov, Rinat Arslanov, Maksim Gunin, Daniil Stefonishin, Andrey Bykov, Sergey Pavlov, Oleg Panarin, Anton Maliutin, Sergey Rykovanov, and Maxim Fedorov. 2019. "Zhores" — petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in Skolkovo Institute of Science and Technology. *Open Engineering*, 9(1):512 – 520.

A Dataset Characteristics

Tables 3 and 4 present the characteristics of the datasets used in experiments.

Table 3: Characteristics of the OntoNotes 5.0 corpus (without the PT section)

	ENG.TRAIN	ENG.TEST
# OF TOKENS	1,088,503	152,728
# OF SENTENCES	59,924	8,262
ENTITY TYPES:		
PERSON	15,429	1,988
GPE	15,405	2,240
ORG	12,820	1,795
DATE	10,922	1,602
CARDINAL	7,367	935
NORP	6,870	841
MONEY	2,434	314
PERCENT	1,763	349
ORDINAL	1,640	195
LOC	1,514	179
TIME	1,233	212
WORK_OF_ART	974	166
FAC	860	135
EVENT	748	63
QUANTITY	657	105
PRODUCT	606	76
LANGUAGE	304	22
LAW	282	40
TOTAL ENTITIES:	81,828	11,257

Table 4: Characteristics of the CoNLL-2003 corpus

	ENG.TRAIN	ENG.TESTB
# OF TOKENS	203,621	46,435
# OF SENTENCES	14,041	3,453
ENTITY TYPES:		
LOC	7,140	1,668
PER	6,600	1,617
ORG	6,321	1,661
MISC	3,438	702
TOTAL ENTITIES:	23,499	5,648

B Features Used by the CRF Model

1. A lowercased word form.
2. Trigram and bigram suffixes of words.
3. Capitalization features.
4. An indicator that shows whether a word is a digit.
5. A part-of-speech tag of a word with specific info (plurality, verb tense, etc.)
6. A generalized part-of-speech.
7. An indicator whether a word is at the beginning or ending of a sentence.
8. The aforementioned characteristics for the next word and previous word except suffixes.

C Model and Training Details

C.1 CRF

We set CRF L1 and L2 regularization terms equal to 0.1, and limit the number of iterations by 100.

C.2 BiLSTM-CRF Taggers

We implement the BiLSTM-CRF sequence tagger on the basis of the `Flair` package[1] (Akbik et al., 2018). We use the same parameters for both types of BiLSTM-CRF models. The recurrent network has one layer with 128 neurons. During training, we anneal the learning rate by half, when the performance of the model stops improving on the development set for 3 epochs. After annealing, we restore the model from the epoch with the best validation score. The starting learning rate is 0.1. The maximal number of epochs is 30, and the batch size is 32. For optimization, we use the standard SGD algorithm.

C.3 Transformer-based Taggers

The implementation of Trasnformer-based taggers is based on the `Hugging Face Transformers` (Wolf et al., 2019)[2] library. We use the following pre-trained versions of BERT, ELECTRA, and DistilBERT accordingly: *'bert-base-cased'*, *'google/electra-base-discriminator'*, and *'distilbert-base-cased'*. The corrected version of Adam (AdamW from the `Transformers` library) is used for optimization with the base learning rate of 5e-5. The linear decay of the learning rate is applied following the (Devlin et al., 2019). The number of epochs is 4 and the batch size is 16. As in (Shen et al., 2018), we see that it is critical to adjust the batch size on early AL iterations, when only small amount of labeled data is available. We reduce the batch size to keep the number of iterations per epoch over 50, but limit the minimal batch size to 4.

---

[1] `https://github.com/flairNLP/flair`

[2] `https://huggingface.co/transformers/`

D Comparison of Various MC Dropout Options

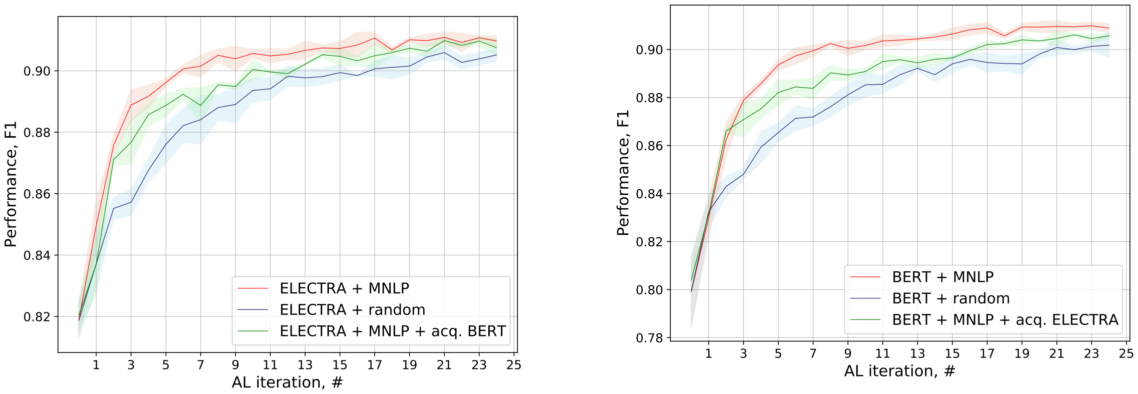
Table 5: Results of AL with various MC dropout options on the CoNLL-2003 dataset

		1	5	10	15	20	24
ELMo-BiLSTM-CRF	MNLP	84.1 ± 1.0	89.5 ± 0.2	90.7 ± 0.2	91.1 ± 0.1	91.1 ± 0.1	91.2 ± 0.1
	Random	82.5 ± 0.5	86.8 ± 0.4	88.7 ± 0.4	89.4 ± 0.2	90.1 ± 0.2	90.3 ± 0.2
	VR(MC word)	83.3 ± 1.0	89.4 ± 0.1	90.5 ± 0.1	90.8 ± 0.3	91.0 ± 0.2	91.1 ± 0.1
	VR(MC all)	84.7 ± 0.7	89.7 ± 0.2	90.7 ± 0.1	90.9 ± 0.1	90.9 ± 0.1	91.2 ± 0.2
	VR(MC locked)	84.4 ± 1.0	89.8 ± 0.2	90.6 ± 0.2	91.0 ± 0.3	91.1 ± 0.2	91.2 ± 0.2
	BALD(MC word)	83.5 ± 1.0	88.8 ± 0.5	90.3 ± 0.2	90.5 ± 0.2	90.9 ± 0.2	91.1 ± 0.2
	BALD(MC locked)	84.3 ± 0.4	89.5 ± 0.4	90.5 ± 0.0	90.8 ± 0.1	91.1 ± 0.1	91.3 ± 0.1
	BALD(MC all)	84.2 ± 0.7	89.6 ± 0.2	90.5 ± 0.2	90.7 ± 0.1	91.1 ± 0.1	91.1 ± 0.1
DistilBERT	MNLP	76.2 ± 2.0	85.9 ± 0.6	88.0 ± 0.2	88.3 ± 0.2	88.9 ± 0.3	89.2 ± 0.1
	Random	76.3 ± 0.6	82.8 ± 1.1	86.0 ± 0.3	87.0 ± 0.2	87.7 ± 0.2	88.2 ± 0.5
	VR (MC last)	78.0 ± 0.2	85.3 ± 0.2	87.7 ± 0.5	88.1 ± 0.4	88.6 ± 0.5	89.0 ± 0.3
	VR (MC all)	78.3 ± 1.4	86.3 ± 0.6	88.3 ± 0.1	88.8 ± 0.3	88.8 ± 0.3	89.0 ± 0.2
	BALD (MC last)	78.0 ± 0.9	86.0 ± 0.2	87.7 ± 0.3	88.3 ± 0.3	89.1 ± 0.0	89.1 ± 0.1
	BALD (MC all)	77.3 ± 1.0	85.6 ± 0.4	87.9 ± 0.1	88.6 ± 0.3	89.0 ± 0.1	89.0 ± 0.1
ELECTRA	MNLP	85.0 ± 0.8	89.6 ± 0.1	90.6 ± 0.1	90.7 ± 0.2	91.0 ± 0.2	91.0 ± 0.2
	Random	83.7 ± 0.8	87.6 ± 0.6	89.4 ± 0.4	89.9 ± 0.3	90.4 ± 0.2	90.5 ± 0.2
	VR (MC last)	85.9 ± 0.7	88.7 ± 0.2	90.3 ± 0.2	90.6 ± 0.5	91.0 ± 0.3	91.0 ± 0.1
	VR (MC all)	84.3 ± 1.0	89.8 ± 0.2	90.4 ± 0.3	90.9 ± 0.2	91.0 ± 0.1	91.2 ± 0.3
	BALD (MC last)	85.6 ± 0.5	90.4 ± 0.2	90.5 ± 0.2	90.6 ± 0.4	90.9 ± 0.2	91.4 ± 0.2
	BALD (MC all)	84.0 ± 0.9	89.5 ± 0.4	90.6 ± 0.2	90.7 ± 0.3	91.1 ± 0.2	90.8 ± 0.3

Table 6: Results of AL with various MC dropout options on the OntoNotes 5.0 dataset

Model	Query strat.	1	5	10	15	20	24
ELMo-BiLSTM-CRF	MNLP	79.5 ± 0.1	85.6 ± 0.2	87.4 ± 0.2	88.1 ± 0.1	88.4 ± 0.1	88.5 ± 0.1
	Random	75.1 ± 1.2	81.5 ± 0.2	84.6 ± 0.2	86.0 ± 0.2	86.9 ± 0.2	87.3 ± 0.2
	VR(MC word)	79.4 ± 0.6	85.1 ± 0.3	87.0 ± 0.1	87.7 ± 0.1	88.0 ± 0.2	88.2 ± 0.2
	VR(MC all)	79.6 ± 1.0	85.6 ± 0.1	87.4 ± 0.2	88.0 ± 0.3	88.3 ± 0.1	88.4 ± 0.1
	VR(MC locked)	79.8 ± 0.5	85.6 ± 0.1	87.4 ± 0.1	88.1 ± 0.2	88.4 ± 0.1	88.5 ± 0.2
	BALD(MC word)	78.6 ± 0.8	84.4 ± 0.3	86.3 ± 0.4	87.4 ± 0.3	87.9 ± 0.2	88.2 ± 0.0
	BALD(MC locked)	78.8 ± 1.5	85.1 ± 0.5	87.0 ± 0.3	87.9 ± 0.2	88.2 ± 0.2	88.4 ± 0.1
	BALD(MC all)	78.9 ± 1.3	85.0 ± 0.5	87.0 ± 0.6	87.9 ± 0.3	88.1 ± 0.2	88.3 ± 0.2
DistilBERT	MNLP	78.3 ± 0.5	84.8 ± 0.4	86.2 ± 0.2	86.9 ± 0.2	87.0 ± 0.2	87.2 ± 0.1
	Random	75.1 ± 0.9	82.5 ± 0.2	84.2 ± 0.3	85.4 ± 0.2	86.0 ± 0.2	86.1 ± 0.2
	VR (MC last)	77.8 ± 0.3	84.4 ± 0.3	85.8 ± 0.0	86.6 ± 0.3	87.0 ± 0.3	87.0 ± 0.3
	VR (MC all)	78.5 ± 0.2	84.6 ± 0.3	86.1 ± 0.0	86.7 ± 0.2	87.2 ± 0.1	87.2 ± 0.1
	BALD (MC last)	78.3 ± 0.3	84.9 ± 0.1	86.3 ± 0.0	86.9 ± 0.0	87.1 ± 0.2	87.0 ± 0.1
	BALD (MC all)	78.3 ± 0.1	84.8 ± 0.1	86.4 ± 0.2	87.2 ± 0.1	87.1 ± 0.1	87.2 ± 0.2
BERT	MNLP	81.8 ± 0.2	86.7 ± 0.1	87.7 ± 0.1	88.1 ± 0.2	88.3 ± 0.2	88.3 ± 0.2
	Random	78.7 ± 0.8	84.6 ± 0.1	86.3 ± 0.3	86.6 ± 0.3	87.2 ± 0.2	87.4 ± 0.1
	VR (MC last)	81.6 ± 0.8	86.4 ± 0.3	87.4 ± 0.2	87.7 ± 0.2	88.0 ± 0.4	88.1 ± 0.1
	VR (MC all)	82.2 ± 0.5	86.8 ± 0.3	87.7 ± 0.2	88.0 ± 0.2	88.5 ± 0.2	88.5 ± 0.2
	BALD (MC last)	81.7 ± 0.5	86.6 ± 0.3	87.7 ± 0.2	88.3 ± 0.2	88.4 ± 0.1	88.4 ± 0.2
	BALD (MC all)	82.3 ± 0.7	86.7 ± 0.0	87.8 ± 0.2	88.3 ± 0.2	88.4 ± 0.1	88.6 ± 0.3

E Experiments with a Mismatch between a Successor Model and an Acquisition Model



a) BERT is an acquisition model

b) ELECTRA is an acquisition model

Figure 4: AL experiments on the CoNLL-2003 dataset, in which a successor model does not match an acquisition model (for BERT and ELECTRA).

MultiHumES: Multilingual Humanitarian Response Dataset for Extractive Summarization

Jenny Paola Yela-Bello
EPFL
Lausanne, Switzerland
ypyelab@gmail.com

Ewan Oglethorpe
Data Friendly Space
Virginia, United States
ewan@datafriendlyspace.org

Navid Rekabsaz
Johannes Kepler University
Linz, Austria
navid.rekabsaz@jku.at

Abstract

When responding to a disaster, humanitarian experts must rapidly process large amounts of secondary data sources to derive situational awareness and guide decision-making. While these documents contain valuable information, manually processing them is extremely time-consuming when an expedient response is necessary. To improve this process, effective summarization models are a valuable tool for humanitarian response experts as they provide digestible overviews of essential information in secondary data. This paper focuses on extractive summarization for the humanitarian response domain and describes and makes public a new multilingual data collection for this purpose. The collection – called MultiHumES – provides multilingual documents coupled with informative snippets that have been annotated by humanitarian analysts over the past four years. We report the performance results of a recent neural networks-based summarization model together with other baselines. We hope that the released data collection can further grow the research on multilingual extractive summarization in the humanitarian response domain.

1 Introduction

The disaster risk management cycle consists of four stages: mitigation, preparedness, response, and recovery (Alexander, 2002). The review of secondary data sources (i.e., reports, news, and other forms of text data) is embedded in all these stages, with varying levels of importance from stage to stage. The work of secondary data review is characterized by a high and ever-increasing amount of information to be analyzed. At the same time, typically, only a small workforce is available to analyze such information. Early in the response phase, namely in the first 72 hours after a disaster strikes, secondary data review to gain situational awareness is essential, as it brings to light which type of relief activities to undertake. After this stage, primary data collection (such as surveys) begins while still supported by the secondary data review processes.

The disaster information cycle by its part consists of the collection, collation, analysis, dissemination, decision-making, and reporting stages. An effective summarization tool can provide meaningful support for the collection stage when analysts prioritize what documents to read first (i.e., offering an overview of a document). In the collation stage, when analysts need to take and merge the most important findings from several documents, and even in the reporting stage, analysts are asked to bring in a few sentences containing the key findings of what they have written. An auto summarization system aims to provide analysts with a starting point from which they can continue their work rather than replace it. Such a system can significantly save time in the overall disaster information cycle.

In this work, we take the initial steps of creating such an extractive summarization system for humanitarian responders by curating and releasing the novel publicly available Multilingual Humanitarian Response Dataset for Extractive Summarization (MultiHumES)[1]. This collection is annotated by humanitarian experts and consists of the data related to various disasters around the globe that occurred in the last four years. Our contribution in this work occurs in two ways: the collection and consultation process for the possible release of the dataset and the dataset curation for performing extractive summarization tasks.

The dataset consists of approximately 50K documents in three languages: English, French, and Spanish. Among these documents, approximately 35K are annotated with informative snippets and can be used for the training and evaluation of ex-

---

[1] https://deephelp.zendesk.com/hc/en-us/sections/360011925552-MultiHumES

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1713–1717
April 19 - 23, 2021.

tractive summarization models. We evaluate the performance of LEAD4, TextRank (Mihalcea and Tarau, 2004) – an unsupervised graph-based model –, and NeuSum (Qingyu et al., 2018) – a recently created supervised neural model in the dataset.

To the best of our knowledge, this is the first multilingual dataset released for summarization in the humanitarian domain. The most similar initiatives are done by (Alam et al., 2020), which released a social-media based dataset for classification in the humanitarian domain, and by Appen, which released a set of short messages from social media and news articles for classification in the humanitarian domain [2].

This article is structured as follows: Section 2 describes the background, annotation process, curation process, and main statistics of the MultiHumES collection. Section 3 explains the experiment design of the summarization models on the dataset, and Section 4 presents and discusses the results.

2 MultiHumES Collection

This section first provides the background on the humanitarian response domain and the ecosystem from which the collection originated. Then it explains the annotation process, the curation process and finalizes presenting key statistics from the collection.

2.1 Collection Background

The collection originated from a multi-organizational platform called DEEP[3]. The DEEP platform was created due to a direct need for effective secondary data management during the Nepal 2015 earthquake response. The platform facilitates classifying primarily qualitative information with respect to analysis frameworks and allows for collaborative classification and annotation of secondary data. To date, the platform has processed almost 250k manually annotated snippets across 1.7k humanitarian projects across the globe.

This research dataset contains the documents analyzed from 2016 to 2019 related to projects that occurred within 159 countries. Approximately 46% of the documents came from media sources, 29% from international organizations, and the rest from various organizations such as United Nations agencies, governments, academic and research institutions, NGOs, donors, and Red Cross/Red Crescent Movement.

Although 82% of the uploaded documents were from publicly available sources, and more than 96% are labeled as non-confidential, we made an additional consultation process with the involved organizations to ensure that the released collection preserves the privacy and dignity of any affected populations discussed in the reports.

2.2 Collection Annotation Process

Taggers (or humanitarian annotators) are trained in analytical standards and thinking. While undertaking secondary data review, information is selected if it fits within a given project's scope and can lead to more appropriate decision-making in a given humanitarian crisis. Key information relevant to understanding unmet needs and their underlying factors is captured and categorized into a commonly agreed analysis framework (or taxonomy), enabling a comprehensive and holistic understanding of humanitarian needs. Detailed categories and sub-categories in line with global standards and taxonomies (geographical area assessed, sectors and sub-sectors, demographic and specific needs groups, etc.) are also labeled. These selected snippets of text fill critical analysis information needs that are essential for strategic and programmatic decision-making in humanitarian response.

One of the major features of the collection is that the data has gone through a rigorous quality control process to ensure standardization, accuracy, and comprehensiveness. Designated quality control experts undertake this process in addition to peer-review and continuous training. The efficacy of the quality control efforts is seen in an external quality assessment, which shows an inter indexer consistency metric of 0.97[4].

2.3 Collection Curation

The original dataset consists of the plain text of 66,412 documents, the snippets extracted from each document by humanitarian analysts, and in some cases, the humanitarian classification label related to each snippet. We parsed the documents provided in HTML format to extract their text content. We also filtered out documents with 50 tokens

---

[2] `https://appen.com/datasets/combined-disaster-response-data/`

[3] `https://thedeep.io/`

[4] `https://github.com/IFRCGo/Tagging-QAQC`

or less. As verified manually, these documents were either title of web pages or small descriptions of attached documents.

The remaining documents are in a variety of languages, but mainly in English, French or Spanish. For automatically identifying the languages, we used the `langdetect` library. We kept documents in the three mentioned languages, resulting in 50,380 documents.

Note that a highlight or relevant text snippet is a sentence or a set of sentences that a humanitarian analyst deems as providing relevant information on a specific category embedded in a conceptual framework. We follow the assumption that these text snippets are not only relevant for a specific analyst and for a specific topic but also for the general summary of the document. Therefore, we combine all text snippets of a document into one document summary, which is used as the ground truth for the extractive summarization task.

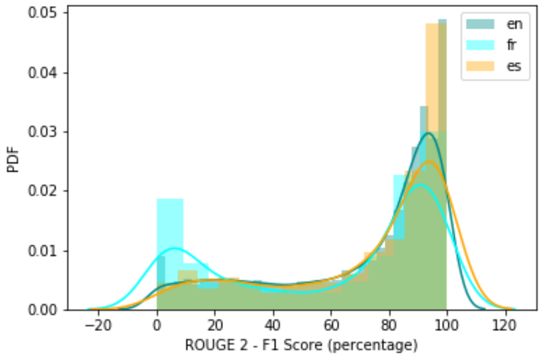


Figure 1: Oracle performance across languages.

Following Qingyu et al. (2018), we constructed an Oracle version of those summaries. Figure 1 shows the ROUGE 2 F1 Score behavior of the Oracle summaries across languages. The x-axis shows the ROUGE 2 F1 Score in a percentage format, and the y-axis shows the probability density function for the kernel density estimation. Based on these results, we see that even with an average score of 70

We evaluated some of the documents in which that score was 0 and found that some were summaries in a language different from the source document. Some had extra characters within the text caused by problems reading the texts from PDF files and implied differences between the document and summary sentences.

After conducting these preprocessing steps, 35,567 documents remained used to create the extractive summarization collection.

2.4 Collection Statistics

	English	**French**	**Spanish**
# of documents	29351	4311	1904
# of tokens	222918	58921	49609
Median number of sent in doc	30	16	23
Median number of sent in sum	5	4	5

Table 1: Dataset Statistics

Table 1 reports the total number of documents, unique tokens, and the median values of the number of sentences per document and summary. The distributions of the number of sentences per document and summary are highly skewed, such that 80% of the documents have less than 191 sentences, and 80% of the summaries have less than 11 sentences.

3 Experiment Design

This section explains the baseline extractive summarization models and their corresponding parameter settings.

3.1 Baseline Models

LEAD4 LEAD-n is an algorithm that selects the first n sentences from a document as its summary. It is a simple but strong baseline for extractive summarization models, created based on the assumption that the first sentences in a document are the most informative ones. In our experiments, we use $n = 4$ as it shows the best performance on the validation set.

Text Rank Text Rank is a graph-based model that ranks text units from most relevant to least relevant by using text units as vertices and the similarity between text units as edges. Given this ranking and a fixed length of the desired output summary, the model produces a summary in an unsupervised manner (Mihalcea and Tarau, 2004).

NeuSum NeuSum is a neural extractive summarization method that employs a hierarchical document encoder to produce sentence representations. It also uses a sentence extractor to iteratively extract sentences from sentence representation extracts according to their overall contribution to the

	ROUGE-1 F1			ROUGE-2 F1		
Model	**en**	**es**	**fr**	**en**	**es**	**fr**
Oracle	0.713	0.761	0.668	0.706	0.733	0.619
LEAD4	0.389	0.454	0.438	0.318	0.378	0.344
TextRank	0.419	0.456	0.418	0.289	0.320	0.277
NeuSum	**0.474**	**0.531**	**0.470**	**0.380**	**0.424**	**0.358**

Table 2: Models' ROUGE scores by language.

performance of the current summary (Qingyu et al., 2018).

The hierarchical document encoder is composed of two Recurrent Neural Networks (RNNs). The first network encodes each word in a sentence (sentence encoder), while the second network encodes each sentence given its context in the document (document encoder). Bidirectional Gated Recurrent Units (BiGRU) (Cho et al., 2014) are used for both RNNs.

3.2 Training and Evaluation

We partition the data into training, validation, and test sets for each language with portions of 70%, 10%, and 20%, respectively.

To evaluate the models, we used ROUGE-1 F1 and ROUGE-2 F1 as standard metrics to evaluate extractive summarization tasks. For each one of the models in each language, the 95% confidence interval of the measures' average values was calculated. The results are reported as significantly different if there is no intersection between the intervals.

For the ROUGE-1 F1 metric, we found that NeuSum performed statistically better for all languages. For the English corpus, TextRank performed statistically better than the LEAD4 baseline. For the ROUGE-2 F1 metric, we found that all results per language were significantly different.

3.3 Parameter Setting

We used the TextRank model implemented by the gensim library (Řehůřek and Sojka, 2010). This implementation uses the Okapi Best Matching 25 similarity measure (BM25+) to measure the similarity between sentences (Barrios et al., 2016). We modified the preprocessing steps and fixed the summaries' length to 100 words to have coherent results with the TextRank model.

For the NeuSum model, we used the word2vec SkipGram model (Mikolov et al., 2013) created on a corpus of Wikipedia provided by the gensim library (Řehůřek and Sojka, 2010). This provided embeddings trained in a similar corpus with a similar dimensionality for the three languages considered. We used 100 as the embedding dimension and 100,000 as the vocabulary size. The pre-trained embeddings covered 51.85% of the English vocabulary, 60.54% of the French vocabulary, and 70.88% of the Spanish vocabulary.

We set 200 sentences as the maximum number of sentences per document and 80 tokens as the maximum length of a sentence. The length of the output summary was set to 4. We ran the model for 40 epochs with a batch size of 64. The rest of the parameters were set as proposed in the original paper.

4 Results and Discussion

Table 2 reports the performance on the test sets. It can be seen that the best performing model was NeuSum, which improved by up to 10 points compared to the TextRank and LEAD4 performance.

The French corpus had the lowest performance, but the fact that the Oracle presents a low performance indicates that this result may be more related to the corpus's nature or the preprocessing of the data rather than the model.

LEAD4 bases its success on the position of the relevant sentences. TextRank bases its success on the content of the relevant sentences. NeuSum higher performance may be explained by the joint encoding of the position information and the content information of a sentence.

It is also key to remember that we truncated each document to a maximum length of 200 sentences because of our GPU capacity. This meant that reports such as the *Humanitarian Needs Overview* with around 60 pages (900 sentences) were reduced to 200 sentences and were therefore not well summarized. Our neural approach for the summarization task of humanitarian documents was useful for setting a precedent of how good these models are in the domain. However, until higher computational resources are more readily available, it would be necessary to have a simpler model to treat longer documents.

5 Conclusions and Future Work

Automatic summarization in the humanitarian domain is crucial for supporting fast and effective responses to crises. To facilitate this process, we provide MultiHumES, a novel multilingual collection for extractive summarization in the humanitarian response domain. The collection enables the training and evaluation of machine/deep learning-based models, revealing new horizons for research in this domain. The collection consists of approximately 50k documents, from which 35K have related annotated snippets by experts. We test and evaluate the performance of three strong baselines on the collection.

We consider the following points as potential future directions. First, a key aspect of automatic summarization for the humanitarian response domain is the human evaluation of the output. It is indeed important to understand whether the reported performance results correlate with the evaluation of domain experts. The second direction is learning multilingual models, trained over all the data, and investigating whether this approach can improve performance, especially in languages with a smaller amount of available training data.

Acknowledgements

We would like to thank the Data Friendly Space team, and the DEEP board for their support. Jenny Paola Yela-Bello thanks the International Federation of Red Cross and Red Crescent societies for supporting her research (under contract CLMX021702).

References

F. Alam, H. Sajjad, M. Imran, and F. Ofli. 2020. Standardizing and benchmarking crisis-related social media datasets for humanitarian information processing. *arxiv, 2020.*

David Alexander. 2002. *Principles of Emergency Planning and Management*. Terra publishing.

Federico Barrios, Federico Lopez, Luis Argerich, and Rosita Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *Argentine Symposium on Artificial Intelligence (ASAI) 2015 - 44 JAIIO (September 2015).*

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

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, page 404–411.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems.*

Zhou Qingyu, Yang Nan, Wei Furu, Huan Shaohan, Zhou Ming, and Zhao Tiejun. 2018. Neural document summarization by jointly learning to score and select sentences. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 654–663.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Learning From Revisions:
Quality Assessment of Claims in Argumentation at Scale

Gabriella Skitalinskaya **Jonas Klaff**
Department of Computer Science
University of Bremen
Bremen, Germany
{gabski, joklaff}@uni-bremen.de

Henning Wachsmuth
Department of Computer Science
Paderborn University
Paderborn, Germany
henningw@upb.de

Abstract

Assessing the quality of arguments and of the claims the arguments are composed of has become a key task in computational argumentation. However, even if different claims share the same stance on the same topic, their assessment depends on the prior perception and weighting of the different aspects of the topic being discussed. This renders it difficult to learn topic-independent quality indicators. In this paper, we study claim quality assessment irrespective of discussed aspects by comparing different *revisions of the same claim*. We compile a large-scale corpus with over 377k claim revision pairs of various types from *kialo.com*, covering diverse topics from politics, ethics, entertainment, and others. We then propose two tasks: (a) assessing which claim of a revision pair is better, and (b) ranking all versions of a claim by quality. Our first experiments with embedding-based logistic regression and transformer-based neural networks show promising results, suggesting that learned indicators generalize well across topics. In a detailed error analysis, we give insights into what quality dimensions of claims can be assessed reliably. We provide the data and scripts needed to reproduce all results.[1]

1 Introduction

Assessing argument quality is as important as it is questionable in nature. On the one hand, identifying the good and the bad claims and reasons for arguing on a given topic is key to convincingly support or attack a stance in debating technologies (Rinott et al., 2015), argument search (Ajjour et al., 2019), and similar. On the other hand, argument quality can be considered on different granularity levels and from diverse perspectives, many of which are inherently subjective (Wachsmuth et al., 2017a); they depend on the prior beliefs and stance on a topic as well as on the personal weighting of different aspects of the topic (Kock, 2007).

Existing research largely ignores this limitation, by focusing on learning to predict argument quality based on subjective assessments of human annotators (see Section 2 for examples). In contrast, Habernal and Gurevych (2016) control for topic and stance to compare the convincingness of arguments. Wachsmuth et al. (2017b) abstract from an argument's text, assessing its relevance only structurally. Lukin et al. (2017) and El Baff et al. (2020) focus on personality-specific and ideology-specific quality perception, respectively, whereas Toledo et al. (2019a) asked annotators to disregard their own stance in judging length-restricted arguments. However, none of these approaches controls for the concrete aspects of a topic that the arguments claim and reason about. This renders it difficult to learn what makes an argument and its building blocks good or bad in general.

In this paper, we study quality in argumentation irrespective of the discussed topics, aspects, and stances by assessing different revisions of the basic building blocks of arguments, i.e., claims. Such revisions are found in large quantities on online debate platforms such as *kialo.com*, where users post claims, other users suggest revisions to improve claim quality (in terms of clarity, grammaticality, grounding, etc.), and moderators approve or disapprove them. By comparing the quality of different revisions of the same instance, we argue that we can learn general quality characteristics of argumentative text and, to a wide extent, abstract from prior perceptions and weightings.

To address the proposed problem, we present a new large-scale corpus, consisting of 124k unique claims from kialo.com spanning a diverse range of topics related to politics, ethics, and several others (Section 3). Using distant supervision, we derive a total number of 377k claim revision pairs from the

---

[1] Data and code:https://github.com/GabriellaSky/claimrev

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1718–1729
April 19 - 23, 2021.

Claim before Revision	**Claim after Revision**	**Type**
Dogs can help disabled people function better.	Dogs can help disabled people to navigate the world better.	Claim Clarification
African American soldiers joined unionists to fight for their freedom.	Black soldiers joined unionists to fight for their freedom.	Typo / Grammar Correction
Elections insure the independence of the judiciary.	Elections ensure the independence of the judiciary.	Typo / Grammar Correction
Israel has a track record of selling US arms to third countries without authorization.	Israel has a track record of selling US arms to third countries without authorization (*https://www.jstor.org/stable/1149008?seq=1#page_scan_tab_contents*).	Corrected / Added links

Table 1: Four examples of claims from Kialo before and after revision, along with the type of revision performed.

platform, each reflecting a quality improvement, often, with a specified revision type. Four examples are shown in Table 1. To the best of our knowledge, this is the first corpus to target quality assessment based on claim revisions. In a manual annotation study, we provide support for our underlying hypothesis that a revision improves a claim in most cases, and we test how much the revision types correlate with known argument quality dimensions.

Given the corpus, we study two tasks: (a) how to compare revisions of a claim by quality and (b) how to rank a set of claim revisions. As initial approaches to the first task, we select in Section 4 a "traditional" logistic regression model based on word embeddings as well as transformer-based neural networks (Vaswani et al., 2017), such as BERT (Devlin et al., 2019) and SBERT (Reimers and Gurevych, 2019). For the ranking task, we consider the Bradley-Terry-Luce model (Bradley and Terry, 1952; Luce, 2012) and SVMRank (Joachims, 2006). They achieve promising results, indicating that the compiled corpus allows learning topic-independent characteristics associated with the quality of claims (Section 5). To understand what claim quality improvements can be assessed reliably, we then carry out a detailed error analysis for different revision types and numbers of revisions.

The main contributions of our work are: (1) A new corpus for topic-independent claim quality assessment, with distantly supervised quality improvement labels of claim revision pairs, (2) initial promising approaches to the tasks of claim quality classification and ranking, and (3) insights into what works well in claim quality assessment and what remains to be solved.

2 Related Work

In the recent years, there has been an increase of research on the quality of arguments and the claims and reasoning they are composed of. Wachsmuth et al. (2017a) describe argumentation quality as a multidimensional concept that can be considered from a logical, rhetorical, and dialectical perspectives. To achieve a common understanding, the authors suggest a unified framework with 15 quality dimensions, which together give a holistic quality evaluation at a certain abstraction level. They point out, that several dimensions may be perceived differently depending on the target audience. In recent follow-up work, Wachsmuth and Werner (2020) examined how well each dimension can be assessed only based on plain text only.

Most existing quality assessment approaches target a single dimension. On mixed-topic student essays, Persing and Ng (2013) learn to score the clarity of an argument's thesis, Persing and Ng (2015) do the same for argument strength, and Stab and Gurevych (2017) classify whether an argument's premises sufficiently support its conclusion. All these are trained on pointwise quality annotations in the form of scores or binary judgments. Gretz et al. (2019) provide a corpus with crowdsourced quality annotations for 30,497 arguments, the largest to date for pointwise argument quality. The authors studied how their annotations correlate with the 15 dimensions from the framework of Wachsmuth et al. (2017a), finding that only *global relevance* and *effectiveness* are captured. Similarly, Lauscher et al. (2020) built a new corpus based on the framework to then exploit interactions between the dimensions in a neural approach. We present a small related annotation study for our dataset below. However, we follow Habernal and Gurevych (2016) in that we cast argument quality assessment as a relation classification problem, where the goal is to identify the better among a pair of instances.

In particular, Habernal and Gurevych (2016) created a dataset with argument convincingness pairs on 32 topics. To mitigate annotator bias, the arguments in a pair always have the same stance on the same topic. The more convincing argument is then predicted using a feature-rich SVM and a simple bidirectional LSTM. Other approaches to the same task map passage representations to real-valued scores using Gaussian Process Preference Learning (Simpson and Gurevych, 2018) or represent arguments by the sum of their token embeddings (Potash et al., 2017), later extended by a Feed Forward Neural Network (Potash et al., 2019). Recently, Gleize et al. (2019) employed a Siamese neural network to rank arguments by the convincingness of evidence. In our experiments below, we take on some of these ideas, but also explore the impact of transformer-based methods such as BERT (Devlin et al., 2019), which have been shown to predict argument quality well (Gretz et al., 2019).

Potash et al. (2017) observed that longer arguments tend to be judged better in existing corpora, a phenomenon we will also check for below. Toledo et al. (2019b) prevent such bias in their corpora for both pointwise and pairwise quality, by restricting the length of arguments to 8–36 words. The authors define quality as the level of preference for an argument over other arguments with the same stance, asking annotators to disregard their own stance. For a more objective assessment of argument relevance, Wachsmuth et al. (2017b) abstract from content, ranking arguments only based on structural relations, but they employ majority human assessments for evaluation. Lukin et al. (2017) take a different approach, including knowledge about the personality of the reader into the assessment, and El Baff et al. (2020) study the impact of argumentative texts on people depending on their political ideology.

As can be seen, several approaches aim to control for length, stance, audience, or similar. However, all of them still compare argumentative texts with different content and meaning in terms of the aspects of topics being discussed. In this work, we assess quality based on different revisions of the same text. In this setting, the quality is primarily focused on how a text is formulated, which will help to better understand what influences argument quality in general, irrespective of the topic. To be able to do so, we refer to online debate portals.

Debate portals give users the opportunity to discuss their views on a wide range of topics. Existing research has used the rich argumentative content and structure of different portals for argument mining, including *createdebate.com* (Habernal and Gurevych, 2015), *idebate.org* (Al-Khatib et al., 2016), and others. Also, large-scale debate portal datasets form the basis of applications such as argument search engines (Ajjour et al., 2019). Unlike these works, we exploit debate portals for studying *quality*. Tan et al. (2016) predicted argument persuasiveness in the discussion forum *ChangeMyView* from ground-truth labels given by opinion posters, and Wei et al. (2016) used user upvotes and downvotes for the same purpose. Here, we resort to *kialo.com*, where users cannot only state argumentative claims and vote on the impact of claims submitted by others, but they can also help improve claims by suggesting revisions, which are approved or disapproved by moderators. While Durmus et al. (2019) assessed quality based on the impact value of claims from kialo.com, we derive information on quality from the revision history of claims.

The only work we are aware of that analyzes revision quality of argumentative texts is the study of Afrin and Litman (2018). From the corpus of Zhang et al. (2017) containing 60 student essays with three draft versions each, 940 sentence writing revision pairs were annotated for whether the revision improves essay quality or not. The authors then trained a random forest classifier for automatic revision quality classification. In contrast, instead of sentences, we shift our focus to claims. Moreover, our dataset is orders of magnitude larger and includes notably longer revision chains, which enables deeper analyses and more reliable prediction of revision quality using data-intensive methods.

3 Data

Here, we present our corpus created based on claim revision histories collected from *kialo.com*.

3.1 A New Corpus based on Kialo

Kialo is a typical example of an online debate portal for collaborative argumentative discussions, where participants jointly develop complex pro/con debates on a variety of topics. The scope ranges from general topics (religion, fair trade, etc.) to very specific ones, for instance, on particular policy-making (e.g., whether wealthy countries should provide citizens with a universal basic income). Each debate consists of a set of claims and is associated with a list of related pre-defined generic categories, such as politics, ethics, education, and entertainment.

What differentiates Kialo from other portals is

Corpus	Type of Instances	Instances
ClaimRev$_{\text{BASE}}$	**Total claim pairs**	**210 222**
	Claim Clarification	63 729
	Typo/Grammar Correction	59 690
	Corrected/Added Links	17 882
	Changed Meaning of Claim	1 178
	Misc	10 464
	None	57 279
ClaimRev$_{\text{EXT}}$	**Total claim pairs**	**377 659**
	Revision distance 1	77 217
	Revision distance 2	27 819
	Revision distance 3	10 753
	Revision distance 4	4 460
	Revision distance 5	2 055
	Revision distance 6+	2 008
Both Corpora	**Claim revision chains**	**124 312**

Table 2: Statistics of the two provided corpus versions. ClaimRev$_{\text{BASE}}$: Number of claim pairs in total and of each revision type. ClaimRev$_{\text{EXT}}$: Number of claim pairs in total and of each revision distance. The bottom line shows the number of unique revision chains in the corpora.

that it allows editing claims and tracking changes made in a discussion. All users can help improve existing claims by suggesting edits, which are then accepted or rejected by the moderator team of the debate. As every suggested change is discussed by the community, this collaborative process should lead to a continuous improvement of claim quality and a diverse set of claims for each topic.

As a result of the editing process, claims in a debate have a version history in the format of claim pairs, forming a chain where one claim is the successor of another and is considered to be of higher quality (examples found in Table 1). In addition, claim pairs may have a revision type label assigned to them via a non-mandatory free form text field, where moderators explain the reason of revision.

Base Corpus To compile the corpus, we scraped all 1628 debates found on Kialo until June 26th, 2020, related to over 1120 categories. They contain 124,312 unique claims along with their revision histories, which comprise of 210,222 pairwise relations. The average number of revisions per claim is 1.7 and the maximum length of a revision chain is 36. 74% of all pairs have a revision type. Overall, there are 8105 unique revision type labels in the corpus. 92% of labeled claim pairs refer to three types only: *Claim Clarification*, *Typo/Grammar Correction*, and *Corrected/Added Links*. An overview of the distribution of revision labels is given in Table 2. We refer to the resulting corpus as *ClaimRev*$_{\mathit{BASE}}$.

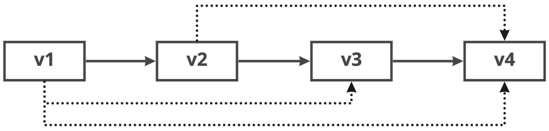


Figure 1: Visual representation of relations between revisions. Solid and dashed lines denote original and inferred non-consecutive relations respectively.

Data pre-processing included removing all claim pairs from debates carried out in languages other than English. Also, we considered claims with less than four characters as uninformative and left them out. As we seek to compare different versions of the *same* claim, claim version pairs with a general change of meaning do not satisfy this description. Thus, we removed such pairs from the corpus, too (inspecting the data revealed that such pairs were mostly generated due to debate restructuring). For this, we assessed the cosine similarity of a given claim pair using *spacy.io* and remove a pair if the score is lower than the threshold of 0.8.

Extended Corpus To increase the diversity of data available for training models, without actually collecting new data, we applied data augmentation. ClaimRev$_{\text{BASE}}$ consists of consecutive claim version pairs, i.e., if a claim v has four versions, it will be represented by three three pairs: (v_1, v_2), (v_2, v_3), and (v_3, v_4), where v_1 is the original claim and v_4 is the latest version. We extend this data by adding all pairs between non-consecutive versions that are inferrable transitively. Considering the previous example, this means we add (v_1, v_3), (v_1, v_4), and $(v2, v4)$. This is based on our hypothesis that every argument version is of higher quality than its predecessors, which we come back to below. Figure 1 illustrates the data augmentation. We call the augmented corpus *ClaimRev*$_{\mathit{EXT}}$.

For this corpus, we introduce the concept of *revision distance*, by which we mean the number of revisions between two versions. For example, the distance between v_1 and v_2 would be 1, whereas the distance between v_1 and v_3 would be 2. The distribution of the revision distances across ClaimRev$_{\text{EXT}}$ is summarized in Table 2.

The number of claim pairs of the 20 most frequent categories in both corpus versions are presented in Figure 2. We will restrict our view to the topics in these categories in our experiments.

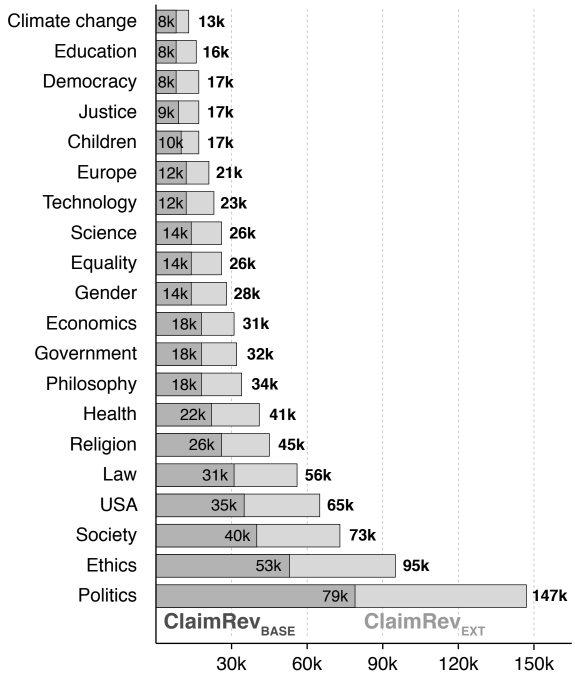


Figure 2: Number of claim revision pairs in each debate category of the two provided versions of our corpus (ClaimRev$_{\text{BASE}}$, ClaimRev$_{\text{EXT}}$).

3.2 Data Consistency on Kialo

While collaborative content creation enables leveraging the wisdom of large groups of individuals toward solving problems, it also poses challenges in terms of quality control, because it relies on varying perceptions of quality, backgrounds, expertise, and personal objectives of the moderators. To assess the consistency of the distantly-supervised corpus annotations, we carried out two annotation studies on samples of our corpus.

Consistency of Relative Quality In this study, we aimed to capture the general perception of claim quality on a meta-level, by deriving a data-driven quality assessment based on the revision histories. This was based on our hypothesis that every claim version is better than its predecessor. To test the validity of this hypothesis, two authors of this paper annotated whether a revision increases, decreases, or does not affect the overall claim quality. For this purpose, we randomly sampled 315 claim revision pairs, found in the supplementary material.

The results clearly support our hypothesis, showing an increase in quality in 292 (93%) of the annotated cases at a Cohen's κ agreement of 0.75, while 8 (3%) of the revisions had no effect on quality and only 6 (2%) led to a decrease. On the remaining 2%, the annotators did not reach an agreement.

Consistency of Revision Type Labels Our second annotation study focused on the reliability of the revision type labels. We restricted our view to the top three revision labels, which cover 96% of all revisions. We randomly sampled 140–150 claim pairs per each revision type, 440 in total. For each claim pair, the same annotators as above provided a label for the revision type from the following set: *Claim Clarification*, *Typo/Grammar Correction*, *Corrected/Added Links*, and *Other*.

Comparing the results to the original labels in the corpus revealed that the annotators strongly agreed with the labels, namely, with Cohen's κ of 0.82 and 0.76 respectively. The level of agreement between the annotators was even higher ($\kappa = 0.84$). In further analysis, we observed that most confusion happened between the revision types *Typo/Grammar correction* and *Claim Clarification*. This may be due to the non-strict nature of the revision type labels, which leaves space for different interpretations on a case-to-case basis. Still, we conclude that the revision type labels seem reliable in general.

3.3 Quality Dimensions on Kialo

To explore the relationship between the revision types on Kialo and argument quality in general, we conducted a third annotation study. In particular, for each of the 315 claim pairs from Section 3.2, one of the authors of this paper provided a label indicating whether the revision improved for each of the 15 quality dimensions defined by Wachsmuth et al. (2017a) or not. It should be noted that the annotators reached an agreement on the revision type for all these pairs.

Table 3 shows Pearson's r rank correlation for each quality dimension for the three main revision types. We observe a strong correlation between the revision type *Corrected/Added Links* and the logical quality dimensions *Cogency* (0.65) and *Local Sufficiency* (0.62), which matches the main purpose of such revisions: to add supporting information to a claim. The high negative correlation of this revision type with *Global Acceptability* (-0.82) indicates that improvements regarding the dimension in question are more prominent in other types. Complementarily, *Claim Clarification* mainly improves the other logical dimensions (*Local Acceptability* 0.38, *Local Relevance* 0.44), matching the intuition that a clarification helps to ensure a correct understanding of the meaning. *Typo/Grammar corrections*, finally, rather seem to support an accept-

	Clarification	Grammar	Links
Cogency	**-0.31**	**-0.31**	**0.65**
Local Acceptability	**0.38**	-0.20	-0.19
Local Relevance	**0.44**	-0.25	-0.22
Local Sufficiency	-0.28	**-0.33**	**0.62**
Effectiveness	0.02	**-0.35**	**0.34**
Credibility	0.06	-0.16	0.10
Emotional Appeal	0.00	0.00	0.00
Clarity	-0.16	**0.35**	-0.18
Appropriateness	0.01	0.02	-0.04
Arrangement	0.00	0.00	0.00
Reasonableness	0.07	-0.04	-0.04
Global Acceptability	**0.37**	**0.42**	**-0.82**
Global Relevance	0.02	**-0.43**	**0.42**
Global Sufficiency	0.00	0.00	0.00
Overall	-0.05	0.00	0.05
Pairs with revision type	120	100	95

Table 3: Pearson's r correlation in our annotation study between increases in the 15 quality dimensions of Wachsmuth et al. (2017a) and the main revision types: Claim *Clarification*, Typo/*Grammar* Correction, Corrected/Added *Links*. Moderate and high correlations are shown in bold ($r \geq 0.3$).

able linguistic shape, improving *Clarity* (0.35) and *Global Acceptability* (0.42).

Finding only low correlations for many rhetorical dimensions (credibility, emotional appeal, etc.) as well as for overall quality, we conclude that the revisions on Kialo seem to target primarily the general form a well-phrased claim should have.

4 Approaches

To study the two proposed tasks, claim quality classification and claim quality ranking, on the given corpus, we consider the following approaches.

4.1 Claim Quality Classification

We cast this task as a pairwise classification task, where the objective is to compare two versions of the same claim and determine which one is better. To solve this task, we compare four methods:

Length To check whether there is a bias towards longer claims in the data, we use a trivial method which assumes that claims with more characters are better.

S-BOW As a "traditional" method, we employ the siamese bag-of-words embedding (S-BOW) as described by Potash et al. (2017). We concatenate two bag-of-words matrices, each representing a claim version from a pair, and input the concatenated matrix to a logistic regression. We also test whether information on length improves S-BOW.

	v_1	v_2	v_3
v_1	0	0.018	0.002
v_2	0.982	0	0.428
v_3	0.998	0.572	0

Table 4: Example of a pairwise score matrix for ranking of three claim revisions, v_1–v_3, given the following pairwise scores: $(v_1, v_2) = (0.018, 0.982)$, $(v_2, v_3) = (0.428, 0.572)$, and $(v_1, v_3) = (0.002, 0.998)$.

BERT We select the BERT model, as it has become the standard neural baseline. BERT is a pre-trained deep bidirectional transformer language model (Devlin et al., 2019). For our experiments we use the pre-trained version *bert-base-cased*, as implemented in the *huggingface* library.[2] We fine-tune the model for two epochs using the Adam optimizer with learning rate 1e-5. [3]

SBERT We also use Sentence-BERT (SBERT) to learn to represent each claim version as a sentence embedding (Reimers and Gurevych, 2019), opposed to the token-level embeddings of standard BERT models. We fine-tune SBERT based on *bert-base-cased* using a siamese network structure, as implemented in the *sentence-transformers* library.[4] We set the numbers of epochs to one which is recommended by the authors (Reimers and Gurevych, 2019), and we use a batch-size of 16, Adam optimizer with learning rate 1e-5, and a linear learning rate warm-up over 10% of the training data. Our default pooling strategy is MEAN.

4.2 Claim Quality Ranking

In contrast to the previous task, we cast this problem as a sequence-pair regression task. After obtaining all pairwise scores using S-BOW, BERT, and SBERT respectively, we map the pairwise labels to real-valued scores and rank them using the following models, once for each method.

BTL For mapping, we use the well-established Bradley-Terry-Luce (BTL) model (Bradley and Terry, 1952; Luce, 2012), in which items are ranked according to the probability that a given item beats an item chosen randomly. We feed the BTL model a pairwise-comparison matrix for all revisions related to a claim, generated as follows: Each row

---

[2] Huggingface library, https://huggingface.co/transformers/pretrained_models.html

[3] We chose the number of epochs empirically, picking the best learning rate out of {5e-7, 5e-6,1e-5,2e-5,3e-5}.

[4] Sentence-transformers library, https://www.sbert.net/

	Test set: ClaimRev$_{\text{BASE}}$				Test set: ClaimRev$_{\text{EXT}}$			
	Random-Split		Cross-Category		Random-Split		Cross-Category	
Model	**Accuracy**	**MCC**	**Accuracy**	**MCC**	**Accuracy**	**MCC**	**Accuracy**	**MCC**
Length	61.3 / 61.3	0.23 / 0.23	60.7 / 60.7	0.21 / 0.21	60.8 / 60.8	0.22 / 0.22	60.0 / 60.0	0.20 / 0.20
SBOW	62.0 / 62.6	0.24 / 0.25	61.4 / 61.4	0.23 / 0.23	64.9 / 65.4	0.30 / 0.31	63.9 / 64.1	0.28 / 0.28
SBOW + Length	65.1 / 65.5	0.30 / 0.31	64.8 / 64.4	0.29 / 0.29	67.1 / 67.5	0.34 / 0.35	66.1 / 66.2	0.32 / 0.32
BERT	75.5 / 75.2	0.51 / 0.51	75.1 / 74.1	**0.51** / 0.49	76.4 / 76.5	0.53 / 0.53	76.2 / 75.4	0.53 / 0.51
SBERT	**76.2 / 76.2**	**0.53 / 0.52**	**75.5 / 75.4**	**0.51 / 0.51**	**77.4 / 77.7**	**0.55 / 0.55**	**76.8 / 76.8**	**0.54 / 0.54**
Random baseline	50.0 / 50.0	0.00 / 0.00	50.0 / 50.0	0.00 / 0.00	50.0 / 50.0	0.00 / 0.00	50.0 / 50.0	0.00 / 0.00
Single claim baseline	57.7 / 58.1	0.17 / 0.17	57.7 / 57.3	0.17 / 0.16	58.8 / 59.8	0.20 / 0.20	58.9 / 58.9	0.20 / 0.20

Table 5: Claim quality classification results: Accuracy and Matthew Correlation Coefficient (MCC) for all tested approaches in the random-split and the cross-category setting on the two corpus versions. The first value in each value pair is obtained by a model trained on ClaimRev$_{\text{BASE}}$, the second by a model trained on ClaimRev$_{\text{EXT}}$. All improvements from one row to the next are significant at $p < 0.001$ according to a two-sided Student's t-test.

represents the probability of the revision being better than other revisions. All diagonal values are set to zero. Table 4 illustrates an example for a set of three argument revisions.

SVMRank Additionally, we employ SVMRank (Joachims, 2006), which views the ranking problem as a pairwise classification task. First, we change the input data, provided as a ranked list, into a set of ordered pairs, where the (binary) class label for every pair is the order in which the elements of the pair should be ranked. Then, SVMRank learns by minimizing the error of the order relation when comparing all possible combinations of candidate pairs. Given the nature of the algorithm we cannot work with token embeddings obtained from BERT directly. Thus, we utilize one of most commonly used approaches to transform token embeddings to a sentence embedding: extracting the special [CLS] token vector (Reimers and Gurevych, 2019; May et al., 2019). In our experiments we select a linear kernel for the SVM and use PySVMRank,[5] a python API to the SVM^{rank} library written in C.[6]

5 Experiments and Discussion

We now present empirical experiments with the approaches from Section 4. The goal is to evaluate how hard it is to compare and rank the claim revisions in our corpus from Section 3 by quality.

5.1 Experimental Setup

We carry out experiments in two settings. The first considers creating *random splits* over revision histories, ensuring that all versions of the same claim are in a single split in order to avoid data leakage. We assign 80% of the revision histories to the training set and the remaining 20% to the test set. A drawback of this setup is that it is not clear how well models generalize to unseen debate categories. In the second setting, we therefore evaluate the methods also in a *cross-category* setup using a leave-one-category-out paradigm, which ensures that all claims from the same debate category are confined to a single split. We split the data in this way to evaluate if our models learn independent features that are applicable across the diverse set of categories. To assess the effect of adding augmented data, we evaluate all models on both ClaimRev$_{\text{BASE}}$ and ClaimRev$_{\text{EXT}}$.

For quality *classification*, we report accuracy and the Matthews correlation coefficient (Matthews, 1975). We report the mean results over five runs in the random setting and the mean results across all test categories in the cross-category setting. To ensure balanced class labels, we create one false claim pair for each true claim pair by shuffling the order of the claims: $(v_1, v_2, true) \rightarrow (v_2, v_1, false)$, where the label denotes whether the second claim in the pair is of higher quality. We report results obtained by models trained on ClaimRev$_{\text{BASE}}$ and ClaimRev$_{\text{EXT}}$ as score pairs in Table 5.

To measure *ranking* performance, we calculate Pearson's r and Spearman's ρ correlation, as well as NDCG and MRR. We also compute the Top-1 accuracy, i.e. the proportion of claim sets, where the latest version has been ranked best. We average the results on each claim set across the test set for each metric. Afterwards we average the results across five runs or across all categories, depending on the chosen setting.

---

[5] PySVMRank, https://github.com/ds4dm/PySVMRank

[6] SVM^{rank}, www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

	Random-Split					**Cross-Category**				
Model	r	ρ	**Top-1**	NDCG	MRR	r	ρ	**Top-1**	NDCG	MRR
BTL + SBOW+L	0.38	0.37	0.62	0.94	0.79	0.36	0.35	0.60	0.94	0.78
BTL + BERT	0.60	0.59	0.74	0.96	0.86	0.58	0.57	0.72	0.96	0.85
BTL + SBERT	0.63	0.62	0.77	**0.97**	0.87	0.62	0.61	0.75	0.97	0.86
SVMRank + SBOW+L	0.18	0.18	0.50	0.93	0.73	0.24	0.23	0.52	0.93	0.75
SVMRank + BERT CLS	0.50	0.49	0.67	0.95	0.84	0.51	0.51	0.67	0.96	0.84
SVMRank + SBERT	**0.70**	**0.70**	**0.79**	**0.97**	**0.90**	**0.73**	**0.72**	**0.80**	**0.98**	**0.91**
Random baseline	0.00	0.00	0.42	0.91	0.68	0.00	0.00	0.42	0.91	0.67

Table 6: Claim quality ranking results: Pearson's r and Spearman's ρ correlation as well as top-1 accuracy for all tested approaches in the random-split and the cross-category setting on ClaimRev$_{\text{EXT}}$.In all cases, SVMRank + SBERT is significantly better than all others at $p < 0.001$ according to a two-sided Student's t-test.

5.2 Claim Quality Classification

The results in Table 5 show that a claim's *length* is a weak indicator of quality (up to 61.3 accuracy). An intuitive explanation is that, even though claims with more information may be better, it is also important to keep them readable and concise.

Despite *SBOW*'s good performance on predicting convincingness (Potash et al., 2017), the claim quality in our corpus cannot be captured by a model of such simplicity (maximum accuracy of 65.4). We point out that adding other linguistic features (for example, part-of-speech tags or sentiment scores) may further improve SBOW. Exemplarily, we equip SBOW with length features and observe a significant improvement (up to 67.5).

As for the transformer-based methods, we see that *BERT* and *SBERT* consistently outperform SBOW in all settings on both corpus versions, with SBERT's accuracy of up to 77.7 being best.[7]

A comparison of the performance of the methods depending on the corpus used for training in Table 5 shows the effect of augmenting the original Kialo data. In most cases, the results obtained by models trained on ClaimRev$_{\text{EXT}}$ are comparable (slightly higher/lower) than results obtained by models trained on ClaimRev$_{\text{BASE}}$. This means that adding relations between non-consecutive claim versions does not improve the reliability of methods. Given that the performance scores obtained on the ClaimRev$_{\text{EXT}}$ test set are evidently higher than on the ClaimRev$_{\text{BASE}}$ test set, we can conclude that the augmented cases are easier to classify and the cumulative difference in quality is more evident.

We can also see in Table 5 that the trained models are able to generalize across categories; the accuracy and MCC scores in the random split and cross-category settings for each method are very similar, with only a slight drop in the cross-category setting. This indicates that the nature of the revisions is relatively consistent among all categories, yet reveals the existence of some category-dependent features.

To find out whether BERT really captures the relative revision quality and not only lexical features present in the original claim, we introduced a *Single claim* baseline, analogous to the *hypothesis-only* baseline in natural language inference(Poliak et al., 2018). It can be seen that the accuracy and MCC scores are low across all settings (maximum accuracy of 59.8), which indicates that BERT indeed captures relative revision quality mostly.

5.3 Claim Quality Ranking

Table 6 lists the results of our ranking experiments, which show patterns similar to the results achieved in the classification task.

We can observe similar patterns in both of the selected ranking approaches: SBERT consistently outperforms all other considered approaches across all settings (up to 0.73 and 0.72 in Pearson's r and Spearman's ρ accordingly). BERT and SBERT outperform SBOW, indicating that transformer-based methods are more capable of capturing the relative quality of revisions. While BTL + BERT obtains results comparable to BTL + SBERT, we find that using the CLS-vector as a sentence embedding representation leads to lower results. We point out, though, that using other sentence embeddings and/or pooling strategies (for example, averaged BERT embeddings) may further improve results.

Similar to the results of the classification task, we observe only a slight performance drop in the

---

[7] Additionally, we have experimented with an adversarial training algorithm, ELECTRA (Clark et al., 2020), and obtained results slightly better than BERT, yet inferior to SBERT. We omit to report these results here, since they did not provide any further notable insights.

Task	Label	Accuracy	Instances
Type	Claim Clarification	69.7	12 856
	Typo/Grammar Correction	83.6	12 125
	Corrected/Added Links	89.3	3 660
	Changed Meaning of Claim	57.3	232
	Misc	67.2	2 130
	None	78.3	45 842
Distance	Revision distance 1	76.2	42 341
	Revision distance 2	79.6	17 478
	Revision distance 3	80.6	8 023
	Revision distance 4	81.0	3 979
	Revision distance 5	79.5	2 103
	Revision distance 6+	74.9	2 921
	All	**77.7**	**76 845**

Table 7: Accuracy of the best model, SBERT, on each single revision type and distance in ClaimRev$_{\text{EXT}}$, along with the number of instances per each case.

cross-category setting when using BTL for ranking, yet an increase when using SVMRank, again emphasizing the topic-independent nature of claim quality in our corpus.

5.4 Error Analysis

To further explore the capabilities and limitations of the best model, SBERT, we analyzed its performance on each revision type and distance.

As the upper part of Table 7 shows, SBERT is highly capable of assessing revisions related to the correction and addition of links and supporting information. This revision type also obtained the highest correlations between quality dimensions and type of revision (see Table 3), which indicates that the patterns of changes performed within this type are more consistent. In contrast, we observe that the model fails to address revisions related to the changed meaning of a claim. On the one hand, this may be due to the fact that such examples are underrepresented in the data. On the other hand, the consideration of such examples in the selected tasks is questionable, since changing the meaning of claim is usually considered as the creation of a *new claim* and not a *new version* of a claim.

An insight from the lower part of Table 7 is that the accuracy of predictions increases from revision distance 1 to 4. We obtain better results when comparing non-consecutive claims than when comparing claim pairs with distance of 1. An intuitive explanation is that, since each single revision should ideally improve the quality of a claim, the more revisions a claim undergoes, the more evident the quality improvement should be. For distances > 5, the accuracy starts to decrease again, but this may be due to the limited number of cases given.

6 Conclusion and Future Work

In this paper, we have proposed a new way of assessing quality in argumentation by considering different revisions of the same claim. This allows us to focus on characteristics of quality regardless of the discussed topics, aspects, and stances in argumentation. We provide a new corpus of web claims, which is the first large-scale corpus to target quality assessment and revision processes on a claim level. We have carried out initial experiments on this corpus using traditional and transformer-based models, yielding promising results but also pointing to limitations. In a detailed analysis we have studied different kinds of claim revisions and provided insights into the aspects of a claim that influence the users' perception of quality. Such insights could help improve writing support in educational settings, or identify the best claims for debating technologies and argument search.

We seek to encourage further research on how to help online debate platforms automate the process of quality control and design automatic quality assessment systems. Such systems can be used to indicate if the suggested revisions increase the quality of an argument or recommend the type of revision needed. We leave it for future work to investigate whether the learned concepts of quality are transferable to content from other collaborative online platforms (such as idebate.org or Wikipedia), or to data from other domains, such as student essays and forum discussions.

Acknowledgments

We thank Andreas Breiter for feedback on early drafts, and the anonymous reviewers for their helpful comments. This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under project number 374666841, SFB 1342.

References

Tazin Afrin and Diane Litman. 2018. Annotation and classification of sentence-level revision improvement. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 240–246, New Orleans, Louisiana. Association for Computational Linguistics.

Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein.

2019. Data acquisition for argument search: The args.me corpus. In *KI 2019: Advances in Artificial Intelligence - 42nd German Conference on AI, Kassel, Germany, September 23-26, 2019, Proceedings*, pages 48–59.

Khalid Al-Khatib, Henning Wachsmuth, Matthias Hagen, Jonas Köhler, and Benno Stein. 2016. Cross-domain mining of argumentative text through distant supervision. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1395–1404. Association for Computational Linguistics.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

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

Esin Durmus, Faisal Ladhak, and Claire Cardie. 2019. The role of pragmatic and discourse context in determining argument impact. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5668–5678, Hong Kong, China. Association for Computational Linguistics.

Roxanne El Baff, Henning Wachsmuth, Khalid Al Khatib, and Benno Stein. 2020. Analyzing the persuasive effect of style in news editorial argumentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3154–3160, Online. Association for Computational Linguistics.

Martin Gleize, Eyal Shnarch, Leshem Choshen, Lena Dankin, Guy Moshkowich, Ranit Aharonov, and Noam Slonim. 2019. Are you convinced? choosing the more convincing evidence with a Siamese network. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 967–976, Florence, Italy. Association for Computational Linguistics.

Shai Gretz, Roni Friedman, Edo Cohen-Karlik, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2019. A large-scale dataset for argument quality ranking: Construction and analysis. *arXiv preprint arXiv:1911.11408*.

Ivan Habernal and Iryna Gurevych. 2015. Exploiting debate portals for semi-supervised argumentation mining in user-generated web discourse. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2127–2137. Association for Computational Linguistics.

Ivan Habernal and Iryna Gurevych. 2016. Which argument is more convincing? analyzing and predicting convincingness of web arguments using bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1589–1599, Berlin, Germany. Association for Computational Linguistics.

Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 217–226, New York, NY, USA. Association for Computing Machinery.

Christian Kock. 2007. Dialectical obligations in political debate. *Informal Logic*, 27(3):233–247.

Anne Lauscher, Lily Ng, Courtney Napoles, and Joel Tetreault. 2020. Rhetoric, logic, and dialectic: Advancing theory-based argument quality assessment in natural language processing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4563–4574, Barcelona, Spain (Online). International Committee on Computational Linguistics.

R Duncan Luce. 2012. *Individual choice behavior: A theoretical analysis*. Courier Corporation.

Stephanie Lukin, Pranav Anand, Marilyn Walker, and Steve Whittaker. 2017. Argument Strength is in the Eye of the Beholder: Audience Effects in Persuasion. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 742–753. Association for Computational Linguistics.

B.W. Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Isaac Persing and Vincent Ng. 2013. Modeling thesis clarity in student essays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 260–269, Sofia, Bulgaria. Association for Computational Linguistics.

Isaac Persing and Vincent Ng. 2015. Modeling argument strength in student essays. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 543–552, Beijing, China. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Peter Potash, Robin Bhattacharya, and Anna Rumshisky. 2017. Length, interchangeability, and external knowledge: Observations from predicting argument convincingness. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 342–351, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Peter Potash, Adam Ferguson, and Timothy J. Hazen. 2019. Ranking passages for argument convincingness. In *Proceedings of the 6th Workshop on Argument Mining*, pages 146–155, Florence, Italy. Association for Computational Linguistics.

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

Ruty Rinott, Lena Dankin, Carlos Alzate Perez, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. 2015. Show me your evidence - an automatic method for context dependent evidence detection. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 440–450, Lisbon, Portugal. Association for Computational Linguistics.

Edwin Simpson and Iryna Gurevych. 2018. Finding convincing arguments using scalable Bayesian preference learning. *Transactions of the Association for Computational Linguistics*, 6:357–371.

Christian Stab and Iryna Gurevych. 2017. Recognizing insufficiently supported arguments in argumentative essays. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 980–990, Valencia, Spain. Association for Computational Linguistics.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 613–624, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Assaf Toledo, Shai Gretz, Edo Cohen-Karlik, Roni Friedman, Elad Venezian, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. 2019a. Automatic argument quality assessment - New datasets and methods. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5625–5635. Association for Computational Linguistics.

Assaf Toledo, Shai Gretz, Edo Cohen-Karlik, Roni Friedman, Elad Venezian, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. 2019b. Automatic argument quality assessment - new datasets and methods. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5625–5635, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Henning Wachsmuth, Nona Naderi, Yufang Hou, Yonatan Bilu, Vinodkumar Prabhakaran, Tim Alberdingk Thijm, Graeme Hirst, and Benno Stein. 2017a. Computational argumentation quality assessment in natural language. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 176–187, Valencia, Spain. Association for Computational Linguistics.

Henning Wachsmuth, Benno Stein, and Yamen Ajjour. 2017b. "PageRank" for argument relevance. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1117–1127, Valencia, Spain. Association for Computational Linguistics.

Henning Wachsmuth and Till Werner. 2020. Intrinsic quality assessment of arguments. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6739–6745, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhongyu Wei, Yang Liu, and Yi Li. 2016. Is this post persuasive? ranking argumentative comments in online forum. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 195–200, Berlin, Germany. Association for Computational Linguistics.

Fan Zhang, Homa B. Hashemi, Rebecca Hwa, and Diane Litman. 2017. A corpus of annotated revisions for studying argumentative writing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1568–1578, Vancouver, Canada. Association for Computational Linguistics.

Few Shot Dialogue State Tracking using Meta-learning

Saket Dingliwal[1,2] **Bill Gao**[1] **Sanchit Agarwal**[1] **Chien-Wei Lin**[1]
Tagyoung Chung[1] **Dilek Hakkani-Tür**[1]
[1]Amazon Alexa AI [2]Carnegie Mellon University
{skdin,shuyag,agsanchi,chienwel,tagyoung,hakkanit}@amazon.com

Abstract

Dialogue State Tracking (DST) forms a core component of automated chatbot based systems designed for specific goals like hotel, taxi reservation, tourist information etc. With the increasing need to deploy such systems in new domains, solving the problem of zero/few-shot DST has become necessary. There has been a rising trend for learning to transfer knowledge from resource-rich domains to unknown domains with minimal need for additional data. In this work, we explore the merits of meta-learning algorithms for this transfer and hence, propose a meta-learner D-REPTILE specific to the DST problem. With extensive experimentation, we provide clear evidence of benefits over conventional approaches across different domains, methods, base models and datasets with significant (5-25%) improvement over the baseline in low-data setting. Our proposed meta-learner is agnostic of the underlying model and hence any existing state-of-the-art DST system can improve its performance on unknown domains using our training strategy.

1 Introduction

Task-Oriented Dialogue (TOD) systems are automated conversational agents built for a specific goal (for example hotel reservation). Many businesses from wide-variety of *domains* (like hotel, restaurant, car-rental, payments etc) have adopted these systems to cut down their cost on customer support services. Almost all such systems have a Dialogue State Tracking (DST) module which keeps track of values for some predefined domain-specific slots (example hotel-name, restaurant-rating etc) after every turn of utterances from user and system. These values are then used by Natural Language Generator (NLG) to generate system responses and fulfill the user goals.

Many of the recent works (Wu et al., 2019; Zhang et al., 2019; Goel et al., 2019; Heck et al., 2020) have proposed various neural models that achieve good performance for the task but are data hungry in general. Therefore, adapting to a new unknown domain (*target* domain) requires large amounts of domain-specific annotations limiting their use. However, given a wide range of practical applications, there has been a recent interest in data-efficient approaches. Lee et al. (2019), Gao et al. (2020) used transformer (Vaswani et al., 2017) based models which significantly reduce data dependence. Further, Gao et al. (2020) model the problem as machine reading comprehension task and benefit from its readily available external datasets and methods. Wu et al. (2019) were first to propose transferring knowledge from one domain to another. Since, many domains like restaurant and hotel share a lot of common slots like name, area, rating, etc and hence such a transfer proved to be effective for a low-resource domain. More recently, Campagna et al. (2020) aimed at zero-shot DST using synthetic data generation for the *target* domain imitating data from other domains.

Recent meta-learning methods like MAML (Finn et al., 2017), REPTILE (Nichol et al., 2018) have proven to be very successful in efficient and fast adaptations to new tasks with very few labelled samples. These methods specifically aim at the setting where there are many similar tasks but very small amount of data for each task. Agnostic of the underlying model, these meta-learning algorithms spit out initialization for its parameters which when fine-tuned using low-resource *target* task achieves good performance. Following their widespread success in few-shot image classification, there has been a lot of recent work on their merit in natural language processing tasks. Huang et al. (2018); Gu et al. (2018); Sennrich and Zhang (2019); Bansal et al. (2019); Dou et al. (2019); Yan et al. (2020)

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1730–1739
April 19 - 23, 2021.

attempt at using meta-learning for efficient transfer of knowledge from high-resource tasks to a low-resource task. Further, some of the more recent works (Dai et al., 2020; Qian and Yu, 2019) have shown meta-learners can be used for system response generation in TOD systems which is generally downstream task for our DST task.

To the best of our knowledge, ours is the first work exploring meta-learning algorithms for the DST problem. While prior work focused on training their models with a mixture of data from other available domains (*train* domains) followed by fine-tuning with data from *target* domain, we identify that this method of transferring knowledge between domains is inefficient, particularly in very low-data setting with just $0, 1, 2, 4$ annotated examples from *target* domain. We, on the other hand, use *train* domains to meta-learn the parameters of the model used to initialize the fine-tuning process. We hypothesize that though different domains share many common slots, they can have different complexities. For some of the domains, it might be easier to train the model using very few examples while others may require large number of gradient steps (based on their different data complexity and training curves with 1%, 5%, 10% data in Gao et al. (2020)). Meta-learning takes into account that this gradient information and share it across domains. Rather than looking for an initialization that try to simultaneously minimizes joint loss over all the domains, it looks for a point from which the optimum parameters of individual domains are reachable in few (< 5) gradient steps (and hence very few training examples for these steps). Then the hope is that the *target* domain is similar to at least one of the *train* domains (for example hotel & restaurant or taxi & train) and hence the learned initialization will achieve efficient fine-tuning with very few examples for the *target* domain as well. This direction of limited data is motivated by practical applicability, where it might be possible for any developer to manually annotate 4-8 examples before deploying the chatbot for a new domain.

We highlight the main contributions of our work below (i) We are the first to explore and reason about the benefits of meta-learning algorithms for DST problem (ii) We propose a meta-learner D-REPTILE that is agnostic to underlying model and hence has the ability to improve the state of the art performance in zero/few-shot DST for new domains. (iii) With extensive experimentation, we provide evidence of the benefit of our approach over conventional methods. We achieve a significant 5-25% improvement over the baseline in few-shot DST that is consistent across different *target* domains, methods, base models and datasets.

2 Background

2.1 Dialogue State Tracking

DST refers to keeping track of the *state* of the dialogue at every turn. *State* of dialogue can be defined as $\langle slot\_name, slot\_value \rangle$ pairs that represents, given a domain-specific slot, the value that the user provides or system-provided value that the user accepts. Further, many domains have a pre-defined *ontology* that specify the set of values each slot can take. Note that the number of values in ontology varies a lot with slots. Some slots like hotel-stars might have just five different values (called *categorical* slots), while those like hotel-name have hundreds of possible values (called *extractive* slots). It might be possible that a slot has never been discussed in the dialogue sequence and in that case, model has to predict a *None* value for that particular slot.

Various models have been proposed for the above task, but particularly relevant to this work is transformer-based model *STARC* by Gao et al. (2020). For each slot, they form a question (like what is the name of the hotel for hotel-name slot) and then at each turn append the tokens from dialogue utterance and the question separated by [SEP] token. They then pass these sequence of tokens through a transformer to form *token embeddings*. For the *extractive* slots, they use *token embeddings* to mark the span (start and end position) of the answer value in the dialogue itself (called *extractive-model*). For the *categorical* slots with less number of possible values, *categorical-model* append embeddings of each possible value to the *token embeddings* and then use a classifier with softmax layer to predict the correct option.

2.2 Meta-Learning

With advances in model-agnostic meta-learning framework by Finn et al. (2017); Nichol et al. (2018), the few-shot problems have been revolutionized. These frameworks define a underlying task-distribution from which both train (τ) and target tasks ($\tau^{'}$) are sampled. For each task τ, we are given very few labelled datapoints $\mathcal{D}_{\tau}^{train}$ and a loss function $\mathcal{L}_{\tau}$. Now, given a new data point $\mathcal{D}_{\tau'}^{test}$ from target task τ', the goal is to learn parameters $\theta_{\mathcal{M}}$ of any model $\mathcal{M}$ such that $\mathcal{L}_{\tau'}(\mathcal{D}_{\tau'}^{test};\theta_{\mathcal{M}})$ is minimized. This is achieved by k-steps of gradient descent using $\mathcal{D}_{\tau'}^{train}$ with learning rate α. More formally, $\theta_{\mathcal{M}} = SGD(\mathcal{D}_{\tau'}^{train}, \mathcal{L}_{\tau'}, \theta_{\mathcal{M}}^{INIT}; k, \alpha)$ where $SGD(\mathcal{D}, \mathcal{L}, \theta^{INIT}; k, \alpha)$ gives $\theta^{(k)}$ such that

$$\theta^{(t)} = \theta^{(t-1)} - \alpha\nabla_{\theta}(\mathcal{L}(\mathcal{D};\theta)), \theta^{(0)} = \theta^{INIT} \quad (1)$$

Therefore, the goal now is to find a good initialization $\theta_{\mathcal{M}}^{INIT}$ for the gradient descent using the data from train tasks τ. This is achieved by minimizing the empirical loss as

$$\theta^{(k)} = SGD(\mathcal{D}_{\tau}^{train}, \mathcal{L}_{\tau}, \theta; k, \alpha) \quad (2)$$

$$\theta_{\mathcal{M}}^{INIT} = arg\ \min_{\theta} \sum_{\tau} \mathcal{L}_{\tau}(\mathcal{D}_{\tau}^{train}; \theta^{(k)}) \quad (3)$$

Note that the above optimization is complex and involve second-order derivatives. For computational benefits, Nichol et al. (2018) proposed REPTILE and showed that these terms can be ignored without effecting the performance of the meta-learning algorithm. We refer the reader to their work for more details.

3 Methodology

In this work, we propose D-REPTILE, a meta-learning algorithm specific to DST task. Following what Qian and Yu (2019) did for dialogue generation problem, we treat different domains as tasks for the meta-learning algorithm. Let D $= \{d_1, d_2, \dots d_n\}$ (eg. $\{restaurant, taxi, payment, \dots\}$) be the set of *train* domains for which we have annotated data available. Let $p_D(.)$ define a probability distribution over these domains. Let $\mathcal{D}_{d_1}, \mathcal{D}_{d_2} \dots \mathcal{D}_{d_n}$ be the training data from each of these domains. Let $\mathcal{M}$ be any DST model with parameters $\theta_{\mathcal{M}}$. Let m be the task-batch size (number of domains in a batch in our case), α, β be the inner and outer learning rate respectively, k be the number of gradient steps. Let $SGD(.)$ be the function as defined in equation 1. Borrowing the meta-learning theory regarding optimizing the objective equation 3 from Nichol et al. (2018), we define the algorithm D-REPTILE in Algorithm 1. The update rule for initialization (as defined in step 8) is same as that of REPTILE. We chose REPTILE over other meta-learning algorithms because of its simplicity and computational advantages. Nonetheless, its straight-forward to switch any other initialization based meta-learner by changing meta-update step. The novelty of our learner lies in its definition of the meta-learning tasks that represent different domains of DST problem. This algorithm aims to find $\theta_{\mathcal{M}}^{INIT}$, which we use to initialize the model for the fine-tuning stage with the *target* domain.

Algorithm 1: D-REPTILE: Meta Learner for DST

```
Input: $\mathcal{D}_{d_1}, \mathcal{D}_{d_2} \dots \mathcal{D}_{d_n}$
Parameters: $\mathcal{M}$, $\mathcal{L}$, $p_D(.)$, $\alpha$, $\beta$, $k$, m
Result: $\theta_{\mathcal{M}}^{INIT}$
1  Initialize $\theta_{\mathcal{M}}$ randomly
2  for iteration $i = 1, 2, \dots$ do
3      sample m domains D_i using $p_D(.)$
4      for domain $d_j \in$ D_i do
5          sample data points $\mathcal{D}_{ij}$ from $\mathcal{D}_{d_j}$
6          $\theta_{\mathcal{M}}^{d_j} \leftarrow SGD(\mathcal{D}_{ij}, \mathcal{L}, \theta_{\mathcal{M}}; k, \alpha)$
7      end
8      $\theta_{\mathcal{M}} \leftarrow \theta_{\mathcal{M}} + \beta\frac{1}{m}\sum_{j=1}^{m}(\theta_{\mathcal{M}}^{d_j} - \theta_{\mathcal{M}})$
9  end
10 return $\theta_{\mathcal{M}}$;
```

We argue that the meta-learned initialization are better suited for fine-tuning than conventional methods. In the hope that joint optimal parameters for *train* domains lie close to individual domains, Wu et al. (2019) initialize the fine-tuning stage of the *target* domain from the joint minimum of the loss from data from all the *train* domains (called Naive pre-training before Fine-Tuning or NFT here). More formally, they chose the following initialization

$$\theta_{\mathcal{M}}^{INIT} = arg \min_{\theta} \sum_{j=1}^{n} \mathcal{L}(\mathcal{D}_{d_j}; \theta) \quad (4)$$

Such an initialization tries to simultaneously minimize the loss for all the domains which might be useful if the goal was to perform well on test data coming from mixture of these domains. However, here our goal is to perform well on a single unknown *target* domain and no direct relation between this initialization and the optimal parameters for the *target* domain can be seen. Further, as the number of *train* domains increases or training data for each domain decreases, the joint optimum can be very far-off from the individual domain-optimum parameters. Therefore, these methods perform particularly bad. We show empirical evidence for this hypothesis in Section 4. On the other hand, if we optimize equation 3, we will reach a point in the parameter space from where all the domain-optimum parameters are reachable in k-gradient descent steps. Therefore, we can hope to reach the optimum parameters for the *target* domain as well efficiently. This hope is much larger for DST problem specifically because of similarities in different related domains (specifically related slots as shown in Section 5).

Let us consider the following example, let restaurant and taxi be two of the *train* domains. Optimizing equation 3, we might reach a point which is closer to optimum parameters of restaurant domain than taxi domain if we have have smaller gradient values for restaurant data but large for taxi. However notably, both the optimum-parameters are reachable in k-gradient steps. Now if *target* domain is hotel (similar to restaurant domain with common slots like rating, name, etc), we will already be close to its optimum parameters. Also if the *target* domain is bus (similar to taxi domain with common slots like time, place, etc), we will have larger gradients in fine-tuning stage and thus will reach the optimum parameters for bus as well. This might not have been possible with equation 4 as the optimum parameters for joint of restaurant and taxi data might be very far from both the individual *train* domains and will also have no specific gradient properties for faster adaptation for any of hotel or bus *target* domains.

4 Experiments

4.1 Datasets

We used two different DST datasets for our experiments. (i) MultiWoz 2.0 (Budzianowski et al., 2018), 2.1 (Eric et al., 2019) (ii) DSTC8 (Rastogi et al., 2019). The former is manually annotated complex dataset with mostly 5 different domains, 8438 dialogues while the latter is relatively simple synthetically generated dataset with 26 domains and 16142 dialogues. Both the datasets contains dialogues spanning multiple domains. Following the setting from Wu et al. (2019), for extracting data of a particular domain from the dataset, we consider all the dialogues in which that domain is present and ignore slots from other domains both in train and test set. Further, as shown by Gao et al. (2020), we use external datasets from Machine Reading for Question Answering (MRQA) 2019 shared task (Fisch et al., 2019), DREAM (Sun et al., 2019), RACE (Lai et al., 2017) to pre-train our transformer in our experiments and label it with suffix '-RC' to distinguish it from '-base' model.

4.2 Evaluation Metric

Based on the objective in DST, there is a well established metric Joint Goal Accuracy (JGA). JGA is the fraction of total turns across all dialogues for which the predicted and ground truth dialogue state matches for all slots. Following Wu et al. (2019), for testing for a single *target* domain in a multi-domain dialogue, we only consider slots from that domain in metric computation. Note that in some of our experiments (where explicitly mentioned), we further restrict the slots to only extractive or only categorical slots. Also, as it happens most of the times, whenever a slot is not mentioned in any turn, the ground truth value for that slot is *None*. For analysis, we further use the metric Active Slot Accuracy which is the fraction of predicted values of a particular slot that were correct whenever the ground truth value was not *none*.

4.3 Experimental Setting

For all our experiments, both D-REPTILE and baseline (NFT (Sec. 3)) uses STARC (Sec. 2) as base model $\mathcal{M}$. This ensures that all the gains achieved in our experiments are only due to meta-learning. In our implementation [1], we use pre-trained word embeddings Roberta-Large (Liu et al., 2019), Adam optimizer (Kingma and Ba, 2014) for gradient updates in both inner and outer loop, $\alpha = 5e^{-5}$, $\beta = 1$, $m = 4$, $k = 5$, $p_D(i) \propto |\mathcal{D}_{d_i}|$ (chosen using dev-set experiments as explained in Section 5). As shown recently (Mosbach et al., 2020), the fine-tuning of transformer based model is unstable, therefore, we run fine-tuning multiple times and report the mean and the standard deviation of the performance. Also, the performance varies with the choice of training data from *target* domain used for fine-tuning. However, for our experiments, we chose these dialogues based on number of active slots (not *None*) and use the same dialogue for both D-REPTILE and baseline. Since, we use very little data (0, 1, or 2 examples) from *target* domain, we obviously would like to have dialogues that at-least have all the slots being discussed in the utterances. In practical scenarios, where a developer might be creating 1 or 2 examples for a new domain, it is always possible to include all the slots in the dialogue utterances.

---

[1] https://github.com/saketdingliwal/Few-Shot-DST

4.4 Results

In our experiments, we are able to achieve significant improvement over the baseline method under low-data setting (< 32 dialogues). Note that the choice of low-data setting is guided by the practical applications of the method. It also validates our hypothesis that the initialization chosen by meta-learning is closer to optimal parameters of the *target* domain in terms of gradient steps and therefore perform better when there is very less data. However, as fine-tuning data is increased to 1000s of dialogues, any random initialization is also able to reach the optimal parameters for *target* domain. We observe the benefits of D-REPTILE in limited data consistently across different domains, datasets and models as explained one-by-one below

Across domains - We used all different domains of MultiWoz 2.0 data as *target* domain in 5 plots in Figure 1. We pre-train D-REPTILE (solid) and NFT (dashed) versions of different models (represented by different colors). For the models represented by red and blue colors, we used all domains other than *target* domain as our *train* domains. For example, for the first plot, hotel domain is our *target* domain, while restaurant, train, attraction and taxi are our *train* domains. The red corresponds to starting with Roberta-Base embeddings, while the blue represent Roberta-RC which is pre-trained Roberta-Base with reading comprehension datasets (Gao et al., 2020). The green dotted line represent model without any pretraining. It is clearly very bad and unstable. This shows importance of using other domains for few-shot experiments. We fine-tune all our models using different amount of training data of *target* domain (x-axis). In each one of our models, the solid lines (D-REPTILE) lies strictly above the dashed lines (NFT) in JGA metric. The gains obtained are as high as 47.8% (D-REPTILE) vs 22.3% (NFT) for restaurant domain with 1 dialogue which is more than 100% improvement at no annotation cost at all.

Across models - Not only the results are consistent across different base models for transformer as shown in Figure 1 but also across different DST methods. As done in Gao et al. (2020), we train separate categorical and extractive models for hotel domain (using categorical and extractive data respectively from *train* domains) (which we have combined to plot Figure 1). If we consider these two fairly different models separately, we achieve similar trends in each individually as plotted in Figure 2. Note that JGA metric is computed here with restricted slots based on the type of the model. The gains are larger for the extractive model possibly because marking span in original dialogue can be considered slightly harder task than choosing among limited number of choices.

Joint Goal Accuracy	Hotel	Restaurant	Taxi	Attraction	Train
TRADE (Wu et al., 2019)	13.7	11.52	60.58	19.87	22.37
STARC (Gao et al., 2020)	28.6	28.2	65	36.9	26.1
STARC + D-REPTILE	**32.4**	**47.8**	**67.2**	**45.9**	**46.1**

Table 1: Zero-shot performance on MultiWoz 2.0 dataset. Domains like restaurant and train witness a significant boost in performance over baselines

Across datasets - To show that the merits of D-REPTILE are not limited to MultiWoz data, we tested with domains from DSTC8 dataset as both *train* and *target* domain. In Figure 1, the orange lines represent model pre-trained using all the domains in DSTC8 as *train* domains while *target* domain is from MultiWoz. As expected, the performance of these models fall below red and blue lines (models pre-trained with MultiWoz *train* domains) but above green (no pre-training) as training and testing datasets are different. However, the solid orange line (D-REPTILE) lies above dashed line (NFT). In another set of experiments, we used *target* domain from DSTC8 and compiled the results in Figure 3. Except for Hotels_1, Hotels_2 and Hotels_3, all other domains from DSTC8 are used as *train* domains while Hotels_2 is kept as *target* domain. We see that the benefits of meta-learning are much larger for DSTC8 dataset than MultiWoz. For example, with 8 dialogues for fine-tuning, D-REPTILE achieves JGA of 43.9% while NFT is only able to get 14.1%. This can be attributed to increase in number of different training tasks (23 domains were used as *train* domains for DSTC8 as opposed to 4 for MultiWoz experiments).

Surprisingly, the meta-learned initializations not only adapt faster but are also better to start with. We see an improvement in zero-shot performance as well. In addition to comparison with the NFT baseline, we also show improvement over existing models on MultiWoz 2.0 dataset in Table 1. Also note that D-REPTILE is model-agnostic and therefore has the capability to improve the JGA for any underlying model for a new unknown domain.

5 Ablation Studies

To validate our various theoretical hypothesis, search for hyper-parameters, clearly identify and

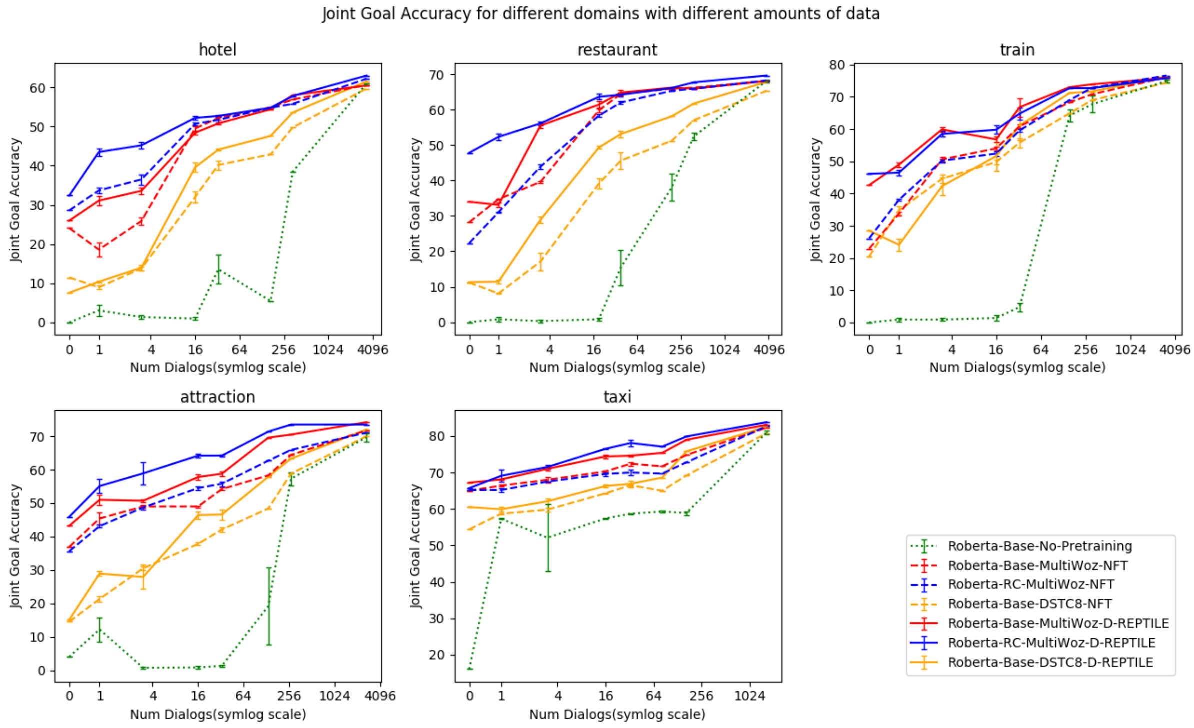


Figure 1: Performance of D-REPTILE vs NFT for different MultiWoz domains with three different models.

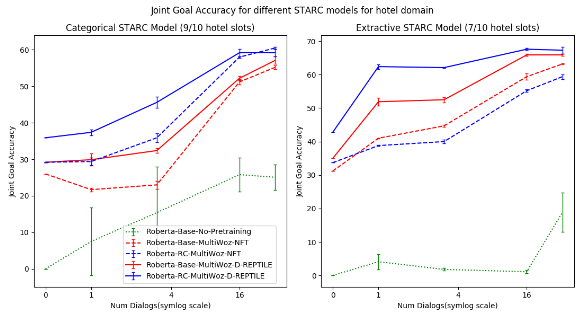


Figure 2: Performance of D-REPTILE vs NFT for different DST models for different slots in hotel domain.

Figure 3: Performance of D-REPTILE vs NFT for Hotels_2 domain in DSTC8 data as *target* domain.

reason about the situations where using meta-learning helps DST, we perform additional analysis as written in subsections below.

5.1 Slot-wise Analysis

To exactly pin-point the advantage of D-REPTILE, we do a slot-wise analysis of our models in Figure 4 and 5. Note that slots are defined as *domain_name.slot_name*. For example, *hotel.day* represents performance of the models in predicting the values for *day* slot where the *target* domain was *hotel*. Overall performance or JGA in plot 1 of Figure 1 is combination of all the *hotel* slots like *day, people, area, etc*. Figure 4 shows the slots which are common among different domains while Figure 5 compare the performance for slots that are unique to a *target* domain. We can see that for the common slots, the solid lines (D-REPTILE) mostly lie higher than the dashed (NFT) counterparts. However, nothing can be said in particular about slots in Figure 5. This behaviour is expected as unique slots particular to a *target* domain have little to gain from the different slots present in *train* domains (which were used for pre-training). This is evident from the fact that slots like *hotel.internet*, *hotel.parking* have zero-shot active accuracy close to zero for all kinds of pretraining strategies (Figure 5). However, wherever slots between different do-

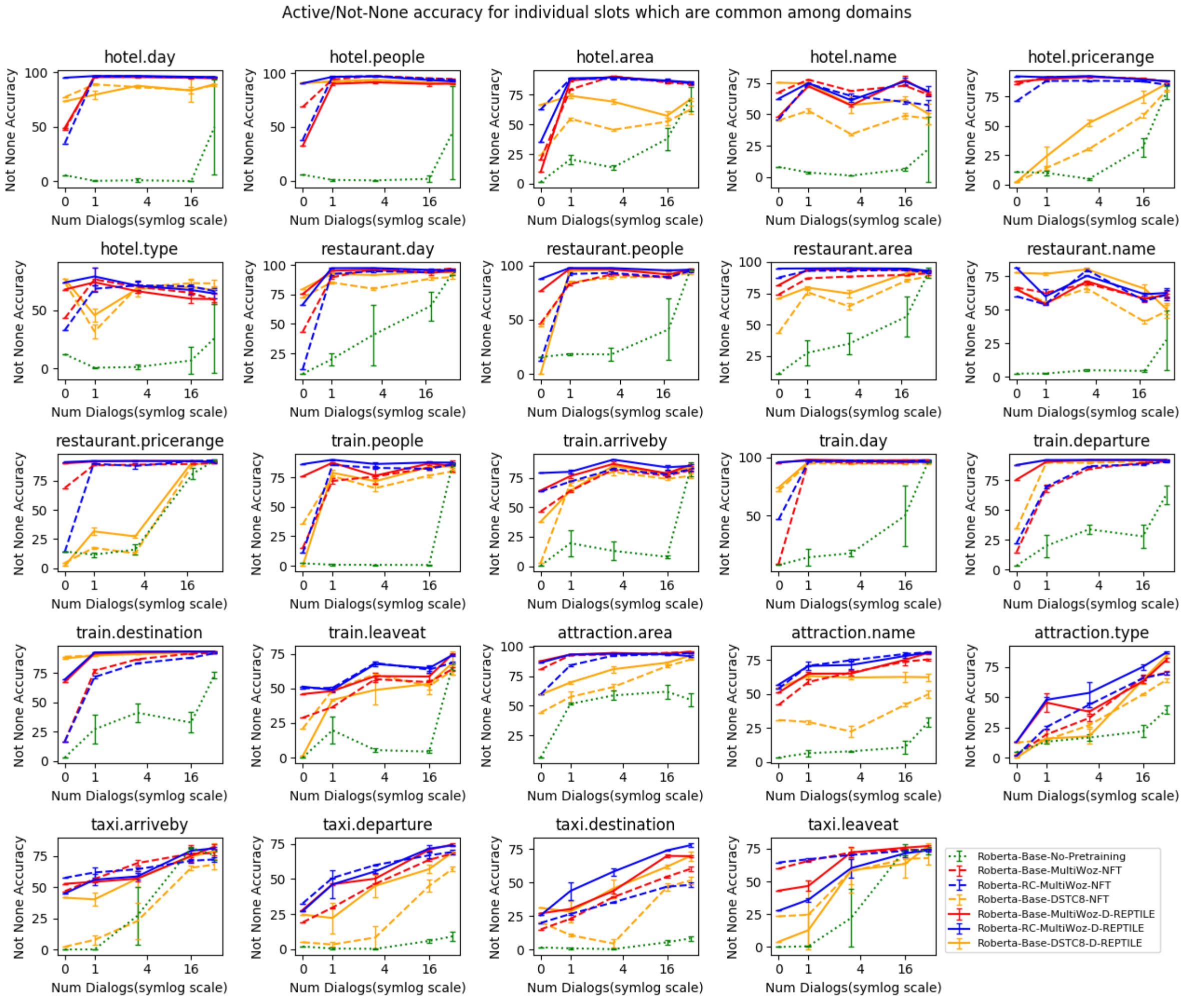


Figure 4: Active Slot accuracy for slots common between different domains

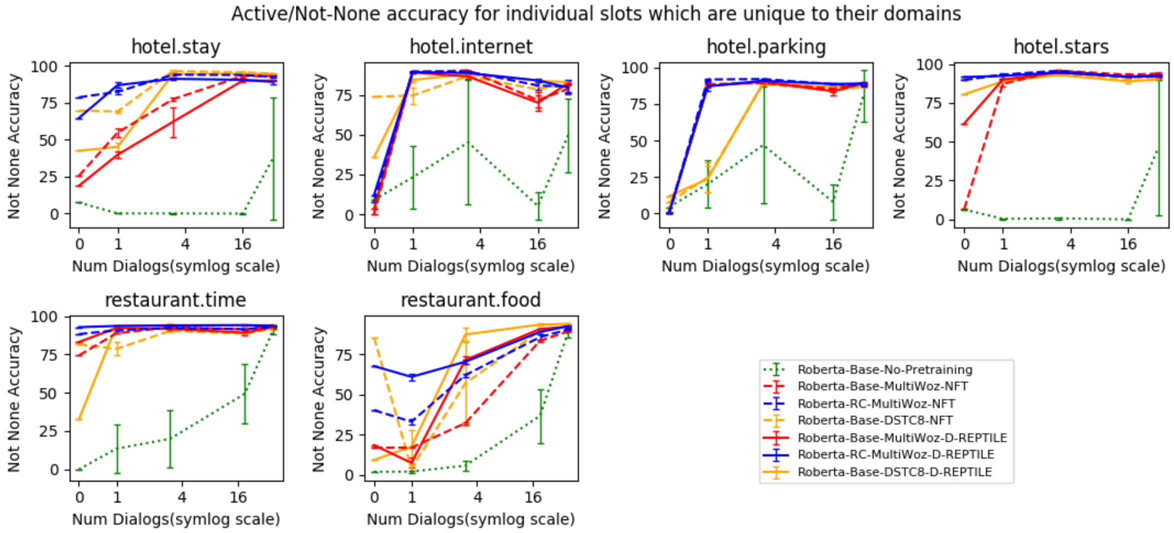


Figure 5: Active Slot accuracy for slots unique to specific *target* domain

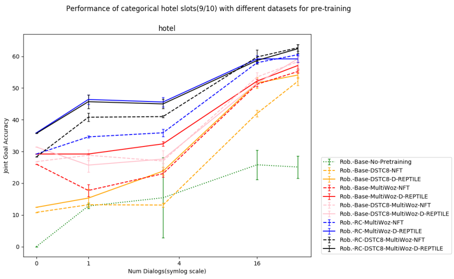


Figure 6: JGA for categorical model for hotel domain with different datasets

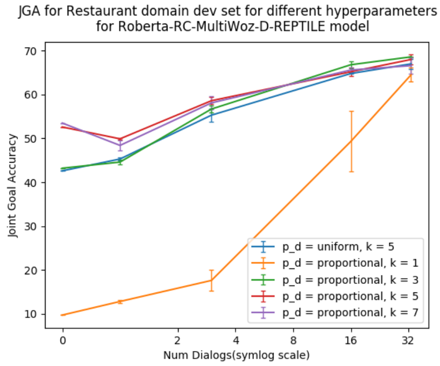


Figure 7: JGA for restaurant domain dev set with different hyper-parameters for the best D-REPTILE model

mains are similar, the pretraining have much larger influence. In that case, the merit of learning generalizable initialization from D-REPTILE than NFT is much more clearly evident (Figure 4).

5.2 Hyper-parameter Search

We briefly discuss the choice of various hyperparameters here. We use dev set from restaurant domain for searching for optimum values for different parameters introduced by meta-learning, while the rest are kept same as STARC model (Gao et al., 2020). In Figure 7, we plot the variation in performance with k and $p_D(.)$. Like any meta-learning algorithm, setting k too small or too large hurts the performance in our case as well (specially $k = 1$ where it becomes theoretically similar to NFT (Nichol et al., 2018)). Hence, optimum value $k = 5$ is used for all our experiments. Also, similar to the conclusion in Dou et al. (2019), we find choosing $p_D(.)$ of any domain as proportional to the size of the training dataset of that domain helpful (blue vs red line). This is attributed to the fact that in case of imbalance in data among different *train* domains, the algorithm gets to see all the data from the resource-rich domain as it is chosen more often and hence generalizes better.

5.3 Adding more *train* domains

As mentioned in previous section, we observe that benefits of D-REPTILE are much more profound when *target* domain is from DSTC8 dataset than when it is from MultiWoz (Figure 3). Given that DSTC8 has 23 *train* domains as compared to 4 in MultiWoz, it is not difficult to see the reason for this boost in performance. In this subsection, we try to answer the question whether MultiWoz *target* can also gain from additional domains of DSTC8. Here, for ease of computation, we only experiment with categorical model with *hotel* domain as *target* . We use both DSTC8 domains and MultiWoz domains (of course excluding hotel domain data during pre-training) and test it on *hotel* data from MultiWoz. These are represented by additional pink and black lines in Figure 6. We observe that although D-REPTILE helps to improve performance over baseline NFT but adding additional domains does not help the model much overall(solid black line is similar to solid blue line). This shows that in addition to the number of different training tasks, the relatedness of those tasks is also very crucial for meta-learning. The DSTC8 domains which are out-of-sample for MultiWoz *target* domain did not prove to be effective. (the small difference between JGA values for 1-dialogue fine-tuning in Figure 6 and categorical model in Figure 2 is due to difference in the choice of the single dialogue from hotel domain used for fine-tuning)

6 Conclusion

We conclude our analysis on the merits of meta-learning as compared to naive pre-training for DST problem on a very positive note. Given the practical applicability of very-low data analysis, we provide enough evidence to a developer of an automated conversational system for an unknown domain that irrespective of his/her model and *target* domain, D-REPTILE can achieve significant improvement (sometimes almost double) over conventional fine-tuning methods with no additional cost. With detailed ablations, we further provide insights on which slots and domains will particularly benefit from pre-traning strategies and which of those will require additional data. Being agnostic to underlying model, our proposed algorithm has capability to push state-of-the-art in zero/few-shot DST problem, giving hope for expanding the scope of similar chatbot based systems in new businesses.

References

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Giovanni Campagna, Agata Foryciarz, Mehrad Moradshahi, and Monica S Lam. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. *arXiv preprint arXiv:2005.00891*.

Yinpei Dai, Hangyu Li, Chengguang Tang, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Learning low-resource end-to-end goal-oriented dialog for fast and reliable system deployment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 609–618.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. *arXiv preprint arXiv:1908.10423*.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. *arXiv preprint arXiv:1910.09753*.

Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Hakkani-Tur. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. *arXiv preprint arXiv:2004.05827*.

Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. *arXiv preprint arXiv:1907.00883*.

Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor OK Li. 2018. Meta-learning for low-resource neural machine translation. *arXiv preprint arXiv:1808.08437*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. *arXiv preprint arXiv:2005.02877*.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. *arXiv preprint arXiv:1803.02400*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. *arXiv preprint arXiv:1907.07421*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Kun Qian and Zhou Yu. 2019. Domain adaptive dialog generation via meta learning. *arXiv preprint arXiv:1906.03520*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. *arXiv preprint arXiv:1905.11901*.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. Dream: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. *arXiv preprint arXiv:1905.08743*.

Ming Yan, Hao Zhang, Di Jin, and Joey Tianyi Zhou. 2020. Multi-source meta transfer for low resource multiple-choice question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7331–7341.

Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. 2019. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv preprint arXiv:1910.03544*.

BERT Prescriptions to Avoid Unwanted Headaches: A Comparison of Transformer Architectures for Adverse Drug Event Detection

Beatrice Portelli[1] **Edoardo Lenzi**[1] **Emmanuele Chersoni**[2]
Giuseppe Serra[1] **Enrico Santus**[3]

[1]AILAB UniUd - University of Udine, Italy
[2]The Hong Kong Polytechnic University
[3]Decision Science and Advanced Analytics for MAPV & RA, Bayer
`{portelli.beatrice,lenzi.edoardo}@spes.uniud.it`
`emmanuele.chersoni@polyu.edu.hk`
`giuseppe.serra@uniud.it, enrico.santus@bayer.com`

Abstract

Pretrained transformer-based models, such as BERT and its variants, have become a common choice to obtain state-of-the-art performances in NLP tasks. In the identification of Adverse Drug Events (ADE) from social media texts, for example, BERT architectures rank first in the leaderboard. However, a systematic comparison between these models has not yet been done. In this paper, we aim at shedding light on the differences between their performance analyzing the results of 12 models, tested on two standard benchmarks.

SpanBERT and PubMedBERT emerged as the best models in our evaluation: this result clearly shows that span-based pretraining gives a decisive advantage in the precise recognition of ADEs, and that in-domain language pretraining is particularly useful when the transformer model is trained just on biomedical text from scratch.

1 Introduction

The identification of Adverse Drug Events (ADEs) from text recently attracted a lot of attention in the NLP community. On the one hand, it represents a challenge even for the most advanced NLP technologies, since mentions of ADEs can be found in different varieties of online text and present unconventional linguistic features (they may involve specialized language, or consist of discontinuous spans of tokens etc.) (Dai, 2018). On the other hand, the task has an industrial application of primary importance in the field of digital pharmacovigilance (Sarker et al., 2015; Karimi et al., 2015b).

This raising interest is attested, for example, by the ACL workshop series on Social Media Health Mining (SMM4H), in which shared tasks on ADE detection have been regularly organized since 2016 (Paul et al., 2016; Sarker and Gonzalez-Hernandez, 2017; Weissenbacher et al., 2018, 2019). With the recent introduction of Transformers architectures and their impressive achievements in NLP (Vaswani et al., 2017; Devlin et al., 2019), it is not surprising that these tools have become a common choice for the researchers working in the area.

The contribution of this paper is a comparison between different Transformers on ADE detection, in order to understand which one is the most appropriate for tackling the task. Shared tasks are not the best scenario for addressing this question, since the wide range of differences in the architectures (which could include, for example, ensembles of Transformers and other types of networks) does not allow a comparison on the same grounds. In our view, two key questions deserve a particular attention in this evaluation. First, whether there is an advantage in using a model with some form of *in-domain language pretraining*, given the wide availability of Transformers for the biomedical domain (Lee et al., 2020; Gu et al., 2020). Second, whether a model trained *to predict coherent spans of text* instead of single words can achieve a better performance (Joshi et al., 2019), since our goal is to identify the groups of tokens corresponding to ADEs as precisely as possible.

Two models that we introduce for the first time in this task, SpanBERT and PubMedBERT, achieved the top performance. The former takes advantage of a span-based pretraining objective, while the latter shows that in-domain language data are better used for training the model from scratch, without any general-domain pretraining.

2 Related Work

2.1 ADE Detection

Automatic extraction of ADE in social media started receiving more attention in the last few

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1740–1747
April 19 - 23, 2021.

years, given the increasing number of users that discuss their drug-related experiences on Twitter and similar platforms. Studies like Sarker and Gonzalez (2015); Nikfarjam et al. (2015); Daniulaityte et al. (2016) were among the first to propose machine learning systems for the detection of ADE in social media texts, using traditional feature engineering and word embeddings-based approaches.

With the introduction of the SMM4H shared task, methods based on neural networks became a more and more common choice for tackling the task (Wu et al., 2018; Nikhil and Mundra, 2018), and finally, it was the turn of Transformer-based models such as BERT (Devlin et al., 2019) and BioBERT (Lee et al., 2020), which are the building blocks of most of the top performing systems in the recent competitions (Chen et al., 2019; Mahata et al., 2019; Miftahutdinov et al., 2019).

At the same time, the task has been independently tackled also by researchers in Named Entity Recognition, since ADE detection represents a classical case of a challenging task where the entities can be composed by discontinuous spans of text (Stanovsky et al., 2017; Dai et al., 2020; Wunnava et al., 2020).

2.2 Transformers Architectures in NLP

There is little doubt that Transformers (Vaswani et al., 2017) have been the dominant class of NLP systems in the last few years. The "golden child" of this revolution is BERT (Devlin et al., 2019), which was the first system to apply the bidirectional training of a Transformer to a language modeling task. More specifically, BERT is trained with a Masked Language Modeling objective: random words in the input sentences are replaced by a [MASK] token and the model attempts to predict the masked token based on the surrounding context.

Following BERT's success, several similar architectures have been introduced in biomedical NLP, proposing different forms of in-domain training or using different corpora (Beltagy et al., 2019; Alsentzer et al., 2019; Lee et al., 2020; Gu et al., 2020). Some of them already proved to be efficient for ADE detection: for example, the top system of the SMM4H shared task 2019 is based on an ensemble of BioBERTs (Weissenbacher et al., 2019).

Another potentially interesting addition to the library of BERTs for ADE detection is SpanBERT (Joshi et al., 2019). During the training of SpanBERT, random contiguous spans of tokens are masked, rather than individual words, forcing the model to predict the full span from the tokens at its boundaries. We decided to introduce SpanBERT in our experiments because longer spans and relations between multiple spans of text are a key factor in ADE detection, and thus encoding such information is potentially an advantage.

3 Experimental Settings

3.1 Datasets

The datasets chosen for the experiments are two widely used benchmarks. They are annotated for the presence of ADEs at character level: each document is accompanied by list of start and end indices for the ADEs contained in it. We convert these annotations using the IOB annotation scheme for the tokens: B marks the start of a mention, I and O the tokens inside and outside a mention respectively.

CADEC (Karimi et al., 2015a) contains 1250 posts from the health-related forum "AskaPatient", annotated for the presence of ADEs. We use the splits made publicly available by Dai et al. (2020).

SMM4H is the training dataset for Task 2 of the SMM4H shared task 2019 (Weissenbacher et al., 2019). It contains 2276 tweets which mention at least one drug name, 1300 of which are positive for the presence of ADEs while the other 976 are negative samples. The competition includes a blind test set, but in order to perform a deeper analysis on the results, we use the training set only. As far as we know there is no official split for the training set alone, so we partitioned it into training, validation and test sets (60:20:20), maintaining the proportions of positive and negative samples. This split and the code for all the experiments are available at `https://github.com/AilabUdineGit/ADE`.

The datasets correspond to different text genres: the tweets of SMM4H are mostly short messages, containing informal language, while the texts of CADEC are longer and structured descriptions. To verify this point, we used the TEXTSTAT Python package to extract some statistics from the texts of the two datasets (see Appendix A).

3.2 Metrics

As evaluation metrics we use the Strict F1 score, which is commonly adopted for this task (Segura-Bedmar et al., 2013). It is computed at the entity level, and assigns a hit only in case of perfect match between the labels assigned by the model and the labels in the gold annotation.

In CADEC around 10% of mentions are discontinuous (Dai et al., 2020) and it is possible to have overlaps and intersections of discontinuous spans. We performed data tidying by merging overlapping ADE mentions, keeping only the longer span (as it is customary in the literature) and splitting discontinuous spans in multiple continuous spans.

3.3 Overview of the Models

3.3.1 Pretrained BERT Variants

Apart from the original BERT, we experimented with SpanBERT, for its peculiar pretraining procedure which focuses on predicting and encoding spans instead of single words, and with four BERT variants with in-domain knowledge, which differ from each other both for the corpus they were trained on and for the kind of pretraining.

BERT Standard model, pretrained on general purpose texts (Wikipedia and BookCorpus).

SpanBERT This model is pretrained using the same corpus as the original BERT, so it comes with no in-domain knowledge. But the pretraining procedure makes its embeddings more appropriate for NER-like tasks. as it introduces an additional loss called Span Boundary Objective (SBO), alongside the traditional Masked Language Modelling (MLM) used for BERT.
Let us consider a sentence $S = [w_1, w_2, \dots, w_k]$ and its substring $S_{m:n} = [w_m, \dots, w_n]$. w_{m-1} and w_{n+1} are the boundaries of $S_{m:n}$ (the words immediately preceding and following it). We *mask* S by replacing all the words in $S_{m:n}$ with the `[MASK]` token. SpanBERT reads the masked version of S and returns an embedding for each word. The MLM loss measures if it is possible to reconstruct each original word $w_i \in S_{m:n}$ from the corresponding embedding. The SBO loss measures if it is possible to reconstruct each $w_i \in S_{m:n}$ using the embeddings of the boundary words w_{m-1} and w_{n+1}.

BioBERT (Lee et al., 2020), pretrained from a BERT checkpoint, on PubMed abstracts.
The authors of BioBERT provide different versions of the model, pretrained on different corpora. We selected the version which seemed to have the greatest advantage on this task, according to the results by Lee et al. (2020). We chose BioBERT v1.1 (+PubMed), which outperformed other BioBERT v1.0 versions (including the ones trained on full texts) in NER tasks involving Diseases and Drugs. Preliminary experiments against BioBERT v.1.0 (+PubMed+PMC) confirmed this behaviour (see Appendix D).

BioClinicalBERT (Alsentzer et al., 2019), pretrained from a BioBERT checkpoint, on clinical texts from the MIMIC-III database.

SciBERT (Beltagy et al., 2019), pretrained *from scratch*, on papers retrieved from Semantic Scholar (82% of medical domain).

PubMedBERT (Gu et al., 2020), pretrained *from scratch*, on PubMed abstracts and full text articles from PubMed Central. This model was created to prove that pretraining from scratch on a *single domain* produces substantial gains on in-domain downstream tasks. Gu et al. (2020) compared it with various other models pretrained on either general texts, mixed-domain texts or in-domain texts starting from a general-purpose checkpoint (e.g. BioBERT), showing that PubMedBERT outperforms them on several tasks based on medical language. The vocabulary of PubMedBERT contains more in-domain medical words than any other model under consideration. However, it should be kept in mind that ADE detection requires an understanding of *both* medical terms and colloquial language, as both can occur in social media text.

Notice that two in-domain architectures were pretrained from scratch (SciBERT and PubMedBERT), meaning that they have a unique vocabulary tailored on their pretraining corpus, and include specific embeddings for in-domain words. BioBERT and BioClinicalBERT were instead pretrained starting from a BERT and BioBERT checkpoint, respectively. This means that the vocabularies are built from general-domain texts (similarly to BERT) and the embeddings are initialized likewise.

3.3.2 Simple and CRF Architecture

For all of the BERT variants, we take into account two versions. The first one simply uses the model to generate a sequence of embeddings (one for each sub-word token), which are then passed to a Linear Layer + Softmax to project them to the output space (one value for each output label) and turn them into a probability distribution over the labels.

The second version combines the Transformer-based model with a Conditional Random Field (CRF) classifier (Lafferty et al., 2001; Papay et al., 2020). The outputs generated by the first version become the input of a CRF module, producing another sequence of subword-level IOB labels. This step aims at denoising the output labels produced by the previous components.

The output labels are calculated for sub-word tokens, then we aggregate each set of sub-word labels $\{\ell_i\}$ into a word label $\mathcal{L}$ using the first rule that applies: (i) if ℓ_i = O for all i, then $\mathcal{L}$ = O; (ii) if ℓ_i = B for any i, then $\mathcal{L}$ = B; (iii) if ℓ_i = I for any i, then $\mathcal{L}$ = I. The aggregated output is a sequence of word-level IOB labels.

3.3.3 Baseline

As a strong baseline, we used the **TMRLeiden** architecture (Dirkson and Verberne, 2019), which achieved the 2nd best Strict F1-Score in the latest SMM4H shared task (Weissenbacher et al., 2019) and is composed of a BiLSTM taking as input a concatenation of BERT and Flair embeddings (Akbik et al., 2019). We chose this baseline since the TMRLeiden code is publicly available.

3.4 Implementation details

TMRLeiden was re-implemented starting from its the original code[1] and trained according to the details in the paper. As for the Transformers, all experiments were performed using the TRANSFORMERS library (Wolf et al., 2019) (see Appendix C). Parameter-tuning was done via grid-search, using different learning rates ($[5e{-}4, 5e{-}5, 5e{-}6]$) and dropout rates (from 0.15 to 0.30, increments of 0.05). All the architectures were trained for 50 epochs on the training set. Learning rate, dropout rate and maximum epoch were chosen evaluating the models on the validation set.

During evaluation all the models were then trained using the best hyperparameters on the concatenation of the training set and the validation set, and tested on the test set. This procedure was repeated five times with different random seeds, and finally we averaged the results over the five runs.

4 Evaluation

The results for the two datasets are shown in Table 1 (we focus on the F1-score, but Precision and Recall are reported in Appendix D). For reference, we reported the scores of the best architecture by Dai et al. (2020), which is the state-of-the-art system on CADEC. At a glance, all systems perform better on CADEC, whose texts belong to a more standardized variety of language. SpanBERT and PubMedBERT emerge as the top performing models, with close F1-scores, and in particular, the SpanBERT models achieve the top score on both datasets, proving that modeling spans gives an important advantage for the identification of ADEs.

	SMM4H		**CADEC**	
Architecture	**F1**	**std**	**F1**	**std**
Dai et al. (2020)	–	–	**68.90**	–
TMRLeiden	60.70	2.08	65.03	1.14
BERT	54.74	1.40	65.20	0.47
BERT+CRF	59.35	1.23	64.36	0.83
SpanBERT	**62.15**	2.17	67.18	0.78
SpanBERT+CRF	59.89	2.16	**67.59**	0.60
PubMedBERT	61.88	0.79	67.16	0.52
PubMedBERT+CRF	59.53	2.07	67.28	0.82
BioBERT	57.83	2.59	65.59	1.10
BioBERT+CRF	58.05	1.45	66.00	0.67
SciBERT	57.75	1.55	65.61	0.54
SciBERT+CRF	58.86	1.55	67.09	0.74
BioClinicalBert	58.03	0.89	64.64	0.53
BioClinicalBert+CRF	59.11	1.99	65.97	0.60

Table 1: F1 scores with standard deviations for all models (our best performing model is in bold).

For both models, the addition of CRF generally determines a slight improvement on CADEC, while it is detrimental on SMM4H. On SMM4H, the F1-scores of BioBERT, SciBERT and BioClinicalBERT consistently improve over the standard BERT, but they are outperformed by its CRF-augmented version, while on CADEC they perform closely to the standard model. The results suggest that in-domain knowledge is consistently useful only when *training is done on in-domain text from scratch*, instead of using general domain text first. SciBERT is also trained from scratch, but on a corpus that is less specific for the biomedical domain than the PubMedBERT one (Gu et al., 2020).

The models are also being compared with TMRLeiden: we can notice that both versions of SpanBERT and PubMedBERT outperform it on CADEC (the differences are also statistically significant for the McNemar test at $p < 0.001$), while only the basic versions of the same models retain an advantage on it on SMM4H (also in this case, the difference is significant at $p < 0.001$).

4.1 Error Analysis

We analyzed the differences between the ADE entities correctly identified by the models and those that were missed, using the text statistics that we previously extracted with TEXTSTAT. As it was

---

[1] `https://github.com/AnneDirkson/SharedTaskSMM4H2019`

1	@hospitalpatient have been on humira 2years now n get on off **chest infections** that sometimes need 2diff pills 2sort out should i b worried ?	**4**	i have had no side effects been taking arthrotec a little over a year, have not noticed any side effects. it does help alot i noticed that when there are times when i forget to take it i can't stand or walk for any lengths of time.
2	had a great few hours on my bike but exercise drives my olanzapine **#munchies** . getting fed up with **not being able to fit into summer wardrobe**	**5**	works just fine. if there are any side effects, they are definitely not noticeable. what's with all these older people (70's) complaining about the lack of sex drive ? how much of what you are complaining about is simply related to getting older?
3	this new baccy is just making my **cough** so much worse but ahh well need my nicotine	**6**	what a great store @walmart is: i loss iq points , gained weight & got addicted to nicotine - all in under 15 min from going in !!

Table 2: Examples of ADEs extracted by PubMedBERT (overlined in blue) and SpanBERT (underlined in red). Actual ADEs in bold with gray background. The Samples belong to SMM4H (1–3, 6) and CADEC (4–5).

predictable, it turns out that longer ADE spans are more difficult to identify: for all models, we extracted the average word length of correct and missed spans and we compared them with a two-tailed Mann-Whitney U test, finding that the latter are significantly longer (Z = -6.176, $p < 0.001$). We also extracted the average number of difficult words in the correct and in the missed spans, defined as words with more than two syllables that are not included in the TEXTSTAT list of words of common usage in standard English. We took this as an approximation of the number of "technical" terms in the dataset instances. However, the average values for correct and missed instances do not differ (Z = 0.109, $p > 0.1$), suggesting that the presence of difficult or technical words in a given instance does not represent an inherent factor of difficulty or facilitation. Still, for some of the models – including SpanBERT, PubMedBERT and TMRLeiden – this difference reaches a marginal significance ($p < 0.05$) exclusively on the SMM4H dataset, where correctly identified spans have more difficult words. A possible interpretation is that, as the tweets' language is more informal, such words represent a stronger ADE cue, compared to the more technical language of the CADEC dataset.

Finally, we performed a qualitative analysis, comparing the predictions of SpanBERT and PubMedBERT. We selected the samples on which one of the architectures performed significantly better than the other one in terms of F1-Score, and analyzed them manually. Some significant samples can be found in Table 2. We observed that most of the samples in which PubMedBERT performed better than SpanBERT contained medical terms, which SpanBERT had completely ignored (e.g. Sample 1). The samples in which SpanBERT outperformed the in-domain model contained instead long ADE mentions, often associated with informal descriptions (e.g. Samples 2, 3). As regards false positives, both models make similar errors, which fit into two broad categories: (1) extracting diseases or symptoms of a disease (e.g. Samples 4, 6); (2) not being able to handle general mentions, hypothetical language, negations and similar linguistic constructs (e.g. Sample 5). While the second kind of error requires a deeper reflection, the first one might be addressed by training the model to extract multiple kinds of entities (e.g. both ADEs and Diseases).

5 Conclusions

We presented a comparison between 12 transformers-based models, with the goal of "prescribing" the best option to the researchers working in the field. We also wanted to test whether the span-based objective of SpanBERT and in-domain language pretraining were useful for the task. We can positively answer to the first question, since SpanBERT turned out to be the best performing model on both datasets. As for the in-domain models, PubMedBERT came as a close second after SpanBERT, suggesting that pretraining from scratch with no general domain data is the best strategy, at least for this task.

We have been the first, to our knowledge, to test these two models in a systematic comparison on ADE detection, and they delivered promising results for future research. For the next step, a possible direction would be to combine the strengths of their respective representations: the accurate modeling of text spans on the one side, and deep biomedical knowledge on the other one.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An Easy-to-use Framework for State-of-the-art nlp. In *Proceedings of NAACL*.

Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly Available Clinical BERT Embeddings. In *Proceedings of the NAACL Workshop on Clinical Natural Language Processing*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *Proceedings of EMNLP*.

Shuai Chen, Yuanhang Huang, Xiaowei Huang, Haoming Qin, Jun Yan, and Buzhou Tang. 2019. HITSZ-ICRC: A Report for SMM4H Shared Task 2019-Automatic Classification and Extraction of Adverse Effect Mentions in Tweets. In *Proceedings of the ACL Workshop on Social Media Mining for Health Applications*.

Xiang Dai. 2018. Recognizing Complex Entity Mentions: A Review and Future Directions. In *Proceedings of ACL 2018, Student Research Workshop*.

Xiang Dai, Sarvnaz Karimi, Ben Hachey, and Cécile Paris. 2020. An Effective Transition-based Model for Discontinuous NER. In *Proceedings of ACL*.

Raminta Daniulaityte, Lu Chen, Francois R Lamy, Robert G Carlson, Krishnaprasad Thirunarayan, and Amit Sheth. 2016. "When 'Bad' Is 'Good'": Identifying Personal Communication and Sentiment in Drug-related Tweets. *JMIR Public Health and Surveillance*, 2(2):e162.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*.

Anne Dirkson and Suzan Verberne. 2019. Transfer Learning for Health-related Twitter Data. In *Proceedings of the ACL Workshop on Social Media Mining for Health Applications*.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *arXiv preprint arXiv:2007.15779*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Sarvnaz Karimi, Alejandro Metke-Jimenez, Madonna Kemp, and Chenchen Wang. 2015a. Cadec: A Corpus of Adverse Drug Event Annotations. *Journal of Biomedical Informatics*, 55:73–81.

Sarvnaz Karimi, Chen Wang, Alejandro Metke-Jimenez, Raj Gaire, and Cecile Paris. 2015b. Text and Data Mining Techniques in Adverse Drug Reaction Detection. *ACM Computing Surveys (CSUR)*, 47(4):1–39.

John Lafferty, Andrew Mccallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: A Pre-trained Biomedical Language Representation Model for Biomedical Text Mining. *Bioinformatics*, 36(4):1234–1240.

Debanjan Mahata, Sarthak Anand, Haimin Zhang, Simra Shahid, Laiba Mehnaz, Yaman Kumar, and Rajiv Shah. 2019. MIDAS@ SMM4H-2019: Identifying Adverse Drug Reactions and Personal Health Experience Mentions from Twitter. In *Proceedings of the ACL Workshop on Social Media Mining for Health Applications*.

Zulfat Miftahutdinov, Ilseyar Alimova, and Elena Tutubalina. 2019. KFU NLP Team at SMM4H 2019 Tasks: Want to Extract Adverse Drugs Reactions from Tweets? BERT to the Rescue. In *Proceedings of the ACL Workshop on Social Media Mining for Health Applications*.

Azadeh Nikfarjam, Abeed Sarker, Karen O'Connor, Rachel E. Ginn, and Graciela Gonzalez-Hernandez. 2015. Pharmacovigilance from Social Media: Mining Adverse Drug Reaction Mentions Using Sequence Labeling with Word Embedding Cluster Features. *Journal of the American Medical Informatics Association : JAMIA*, 22:671 – 681.

Nishant Nikhil and Shivansh Mundra. 2018. Neural DrugNet. In *Proceedings of the EMNLP Workshop on Social Media Mining for Health Applications*.

Sean Papay, Roman Klinger, and Sebastian Padó. 2020. Dissecting Span Identification Tasks with Performance Prediction. In *Proceedings of EMNLP*.

Michael Paul, Abeed Sarker, John Brownstein, Azadeh Nikfarjam, Matthew Scotch, Karen Smith, and Graciela Gonzalez. 2016. Social Media Mining for Public Health Monitoring and Surveillance. In *Biocomputing 2016*, pages 468–479.

Abeed Sarker, Rachel Ginn, Azadeh Nikfarjam, Karen O'Connor, Karen Smith, Swetha Jayaraman, Tejaswi Upadhaya, and Graciela Gonzalez. 2015. Utilizing Social Media Data for Pharmacovigilance: A Review. *Journal of Biomedical Informatics*, 54:202–212.

Abeed Sarker and Graciela Gonzalez. 2015. Portable Automatic Text Classification for Adverse Drug Reaction Detection via Multi-corpus Training. *Journal of Biomedical Informatics*, 53:196–207.

Abeed Sarker and Graciela Gonzalez-Hernandez. 2017. Overview of the Social Media Mining for Health (SMM4H) Shared Tasks at AMIA 2017. *Training*, 1(10,822):1239.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013). In *Proceedings of SemEval*.

Gabriel Stanovsky, Daniel Gruhl, and Pablo Mendes. 2017. Recognizing Mentions of Adverse Drug Reaction in Social Media Using Knowledge-Infused Recurrent Models. In *Proceedings of EACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proceedings of NIPS*.

Davy Weissenbacher, Abeed Sarker, Arjun Magge, Ashlynn Daughton, Karen O'Connor, Michael Paul, and Graciela Gonzalez. 2019. Overview of the Fourth Social Media Mining for Health (SMM4H) Shared Tasks at ACL 2019. In *Proceedings of the ACL Social Media Mining for Health Applications (# SMM4H) Workshop & Shared Task*.

Davy Weissenbacher, Abeed Sarker, Michael Paul, and Graciela Gonzalez. 2018. Overview of the Social Media Mining for Health (SMM4H) Shared Tasks at EMNLP 2018. In *Proceedings of the EMNLP Workshop on Social Media Mining for Health Applications*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, abs/1910.03771.

Chuhan Wu, Fangzhao Wu, Junxin Liu, Sixing Wu, Yongfeng Huang, and Xing Xie. 2018. Detecting Tweets Mentioning Drug Name and Adverse Drug Reaction with Hierarchical Tweet Representation and Multi-head Self-attention. In *Proceedings of the EMNLP Workshop on Social Media Mining for Health Applications*.

Susmitha Wunnava, Xiao Qin, Tabassum Kakar, Xiangnan Kong, and Elke Rundensteiner. 2020. A Dual-Attention Network for Joint Named Entity Recognition and Sentence Classification of Adverse Drug Events. In *Findings of EMNLP*.

A Text statistics for datasets

Some statistics for the texts of the two datasets have been extracted with the TEXTSTAT Python package and reported reported in Table A: we extracted the counts of syllables, lexicon (how many different word types are being used), sentences and characters. Difficult words refers to the number of polysyllabic words with Syllable Count > 2 that are not included in the list of words of common usage in English.

Metric	**CADEC**	**SMM4H**
Syllable Count	116 ± 2.7	26 ± 0.2
Lexicon Count	83 ± 1.9	18 ± 0.1
Sentence Count	5 ± 0.1	1 ± 0.0
Character Count	461 ± 10.5	104 ± 0.7
Difficult Words	14 ± 0.3	4 ± 0.1

Table 3: Average metrics per dataset, computed by the TEXTSTAT Python library.

B Best hyperparameters on the two datasets

Table 4 reports the best hyperparameters for all architectures on SMM4H and CADEC, respectively.

	SMM4H			**CADEC**		
Architecture	**lr**	**dropout**	**epoch**	**lr**	**dropout**	**epoch**
BERT	$5e{-}5$	0.20	4	$5e{-}5$	0.25	11
BERT+CRF	$5e{-}5$	0.15	6	$5e{-}5$	0.15	7
SpanBERT	$5e{-}5$	0.25	43	$5e{-}5$	0.25	19
SpanBERT+CRF	$5e{-}5$	0.15	14	$5e{-}5$	0.15	11
PubMedBERT	$5e{-}5$	0.25	21	$5e{-}5$	0.15	7
PubMedBERT+CRF	$5e{-}5$	0.25	13	$5e{-}5$	0.25	16
BioBERT	$5e{-}5$	0.20	8	$5e{-}5$	0.25	12
BioBERT+CRF	$5e{-}5$	0.15	6	$5e{-}5$	0.20	9
SciBERT	$5e{-}5$	0.15	7	$5e{-}5$	0.15	6
SciBERT+CRF	$5e{-}5$	0.25	13	$5e{-}5$	0.25	12
BioClinicalBERT	$5e{-}5$	0.25	10	$5e{-}5$	0.25	6
BioClinicalBERT+CRF	$5e{-}5$	0.25	12	$5e{-}5$	0.25	10

Table 4: Best hyperparameters for all Transformer-based architectures on SMM4H and CADEC.

C General information on the models

Table 5 is a summary of the information about the version of all Transformer-based models used and their pretraining methods.

D Detailed metrics of all the models

Table 6 and 7 report as Strict and Partial metrics the F1-score, Precision and Recall calculated for all architectures on SMM4H and CADEC respectively. Results are the average over five runs.

The Partial scores are standard metrics for this task (Weissenbacher et al., 2019) and take into account "partial"matches, in which it is sufficient for a system prediction to partially overlap with the gold annotation to be considered as a true match.

Name	Version	Vocabulary	Pretraining	Pretraining Corpus
BERT	base uncased	Wikipedia+BookCorpus	from scratch	Wikipedia+BookCorpus
SpanBERT	base cased	Wikipedia+BookCorpus	from scratch	Wikipedia+BookCorpus
PubMedBERT	base uncased abstract+fulltext	PubMed	from scratch	PubMed+PMC
BioBERT	base v1.1 (+PubMed)	Wikipedia+BookCorpus	from BERT	PubMed
BioBERT(v1.0)	base v1.0 (+PubMed+PMC)	Wikipedia+BookCorpus	from BERT	PubMed+PMC
SciBERT	scivocab cased	Semantic Scholar	from scratch	Semantic Scholar
BioClinicalBERT	bio+clinical	Wikipedia+BookCorpus	from BioBERT	MIMIC-III

Table 5: Information about the version of all the Transformer-based models used and their pretraining.

Strict				Partial		
F1	**P**	**R**	**Architecture**	**F1**	**P**	**R**
60.70 ± 2.08	**68.36 ± 2.41**	54.59 ± 1.97	TMRLeiden	66.08 ± 1.79	**74.42 ± 2.11**	59.43 ± 1.76
54.74 ± 1.40	48.50 ± 1.67	62.84 ± 1.12	BERT	64.53 ± 1.09	57.17 ± 1.52	74.08 ± 0.78
59.35 ± 1.23	54.12 ± 1.19	65.69 ± 1.34	BERT+CRF	68.35 ± 0.64	62.33 ± 0.74	75.66 ± 0.68
62.15 ± 2.17	54.54 ± 3.06	**72.31 ± 1.30**	SpanBERT	69.38 ± 1.60	60.88 ± 2.74	**80.74 ± 1.08**
59.89 ± 2.16	54.86 ± 3.10	66.05 ± 1.93	SpanBERT+CRF	68.09 ± 1.51	62.35 ± 2.79	75.10 ± 1.72
61.88 ± 0.79	58.70 ± 0.83	65.45 ± 1.39	PubMedBERT	**69.82 ± 0.60**	66.23 ± 0.86	73.84 ± 1.26
59.53 ± 2.07	55.29 ± 2.19	64.49 ± 2.27	PubMedBERT+CRF	67.94 ± 1.48	63.10 ± 1.69	73.61 ± 1.84
55.22 ± 1.71	49.85 ± 1.76	61.89 ± 1.78	BioBERT v1.0	64.25 ± 1.09	58.00 ± 1.22	72.02 ± 1.30
57.83 ± 2.59	53.68 ± 3.20	62.72 ± 2.30	BioBERT	66.58 ± 1.34	61.79 ± 2.25	72.23 ± 1.42
58.05 ± 1.45	54.44 ± 2.18	62.22 ± 1.22	BioBERT+CRF	66.30 ± 0.85	62.17 ± 1.83	71.07 ± 1.15
57.75 ± 1.55	53.49 ± 0.97	62.75 ± 2.54	SciBERT	66.49 ± 0.83	61.61 ± 0.61	72.25 ± 1.89
58.86 ± 1.55	52.94 ± 2.27	66.35 ± 1.86	SciBERT+CRF	67.12 ± 0.97	60.36 ± 1.93	75.67 ± 1.99
58.03 ± 0.89	51.63 ± 1.51	66.26 ± 0.46	BioClinicalBERT	66.90 ± 0.57	59.52 ± 1.29	76.39 ± 0.99
59.11 ± 1.99	52.35 ± 2.55	67.92 ± 1.55	BioClinicalBERT+CRF	67.41 ± 1.19	59.69 ± 1.92	77.48 ± 1.40

Table 6: Results on SMM4H, F1-scores, Precision and Recall calculated as Strict and Partial metrics, with standard deviations for all models.

Strict				Partial		
F1	**P**	**R**	**Architecture**	**F1**	**P**	**R**
65.03 ± 1.14	**67.50 ± 1.01**	62.75 ± 1.26	TMRLeiden	77.08 ± 0.78	**79.99 ± 0.60**	74.36 ± 0.97
65.20 ± 0.47	62.86 ± 0.52	67.72 ± 0.70	BERT	77.73 ± 0.28	74.95 ± 0.57	80.74 ± 0.47
64.36 ± 0.83	62.47 ± 0.97	66.36 ± 0.79	BERT+CRF	77.23 ± 0.45	74.97 ± 0.72	79.63 ± 0.41
67.18 ± 0.78	65.84 ± 0.94	**68.57 ± 0.78**	SpanBERT	79.18 ± 0.61	77.60 ± 0.79	**80.82 ± 0.72**
67.59 ± 0.60	67.09 ± 0.54	68.10 ± 0.78	SpanBERT+CRF	**79.43 ± 0.27**	78.84 ± 0.24	80.02 ± 0.60
67.16 ± 0.52	66.60 ± 0.67	67.73 ± 0.57	PubMedBERT	79.13 ± 0.23	78.47 ± 0.51	79.81 ± 0.42
67.28 ± 0.82	66.69 ± 0.99	67.88 ± 0.91	PubMedBERT+CRF	79.12 ± 0.43	78.43 ± 0.72	79.83 ± 0.71
65.54 ± 0.47	64.24 ± 0.48	66.90 ± 0.46	BioBERT v1.0	77.86 ± 0.34	76.32 ± 0.36	79.47 ± 0.33
65.59 ± 1.10	64.86 ± 1.39	66.34 ± 0.85	BioBERT	78.17 ± 0.75	77.30 ± 1.13	79.06 ± 0.48
66.00 ± 0.67	65.52 ± 0.97	66.48 ± 0.63	BioBERT+CRF	78.24 ± 0.43	77.68 ± 0.81	78.82 ± 0.58
65.61 ± 0.54	64.46 ± 0.70	66.80 ± 0.50	SciBERT	78.05 ± 0.19	76.69 ± 0.36	79.47 ± 0.46
67.09 ± 0.74	65.99 ± 0.74	68.23 ± 0.80	SciBERT+CRF	79.01 ± 0.35	77.72 ± 0.36	80.35 ± 0.50
64.64 ± 0.53	61.99 ± 0.51	67.53 ± 0.56	BioClinicalBERT	76.95 ± 0.35	73.80 ± 0.36	80.39 ± 0.37
65.97 ± 0.60	64.23 ± 1.16	67.82 ± 0.60	BioClinicalBERT+CRF	77.98 ± 0.49	75.92 ± 1.26	80.17 ± 0.53

Table 7: Results on CADEC, F1-scores, Precision and Recall calculated as Strict and Partial metrics, with standard deviations for all models.

Semantic Parsing of Disfluent Speech

Priyanka Sen
Amazon Alexa AI
Cambridge, UK
`sepriyan@amazon.com`

Isabel Groves
Amazon Alexa AI
Cambridge, UK
`isabeg@amazon.com`

Abstract

Speech disfluencies are prevalent in spontaneous speech. The rising popularity of voice assistants presents a growing need to handle naturally occurring disfluencies. Semantic parsing is a key component for understanding user utterances in voice assistants, yet most semantic parsing research to date focuses on written text. In this paper, we investigate semantic parsing of disfluent speech with the ATIS dataset. We find that a state-of-the-art semantic parser does not seamlessly handle disfluencies. We experiment with adding real and synthetic disfluencies at training time and find that adding synthetic disfluencies not only improves model performance by up to 39% but can also outperform adding real disfluencies in the ATIS dataset.

1 Introduction

Spoken language differs from written language. Unlike written texts, spontaneous speech frequently contains disfluencies such as filled pauses, repetitions, repairs, and false starts. These affect around 6% of words (Kasl and Mahl, 1965; Tree, 1995; Bortfeld et al., 2001) and occur in both human-human and human-computer interactions (Oviatt, 1995). While considerable research has been done on speech disfluencies in syntactic parsing (Johnson and Charniak, 2004; Wang et al., 2018; Jamshid Lou et al., 2019), semantic parsing of disfluent speech has received less attention.

Semantic parsing is the task of mapping written text to a representation of its meaning. In voice assistants, such as Amazon Alexa, Google Assistant, or Siri, an automated speech recognition (ASR) component is often followed by a semantic parsing task that identifies the intent and slots of the transcribed utterance. In voice-based contexts, speech disfluencies are to be expected, and so a voice assistant must accurately support disfluent speech to support more natural user interaction. Despite the growing popularity of voice assistants, semantic parsing research to date has focused on written language. Most popular semantic parsing datasets only include written forms that were generated from grammars (Coucke et al., 2018) or crowdsourced (Gupta et al., 2018) and do not contain speech phenomena such as disfluencies.

One approach to parsing disfluent speech is to detect and remove transcribed disfluencies in a post-processing step after ASR with a disfluency detector (Zayats et al., 2016; Jamshid Lou et al., 2018). Alternatively, some ASR models attempt to directly produce fluent transcriptions from disfluent inputs (Jamshid Lou and Johnson, 2020) but have lower performance. We investigate semantic parsing of disfluent utterances directly, without ASR post-processing, by training a single semantic parsing model that handles both fluent and disfluent utterances. Our approach has the advantage of being simple and easier to implement than approaches that require multiple models in a pipeline.

We investigate semantic parsing of disfluent speech using the ATIS semantic parsing dataset (Price, 1990), which was originally collected as spontaneous speech. ATIS is a small dataset but currently the only semantic parsing dataset that contains speech disfluencies. As such, our findings are limited by dataset size and results may differ given more data, however, we still find interesting preliminary results. First, we compare performance of a state-of-the-art semantic parser on disfluent and fluent examples in ATIS and identify a performance gap of 79%. To address this gap, we experiment with augmenting the training data with real disfluencies, synthetically-generated disfluencies, and a combination of both. We find that adding synthetic disfluencies adds up to 39% improvement in exact-match accuracy for disfluent utterances. We also find that real disfluencies degrade perfor-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1748–1753
April 19 - 23, 2021.

Fluent	please find a flight from detroit michigan to st. petersburg arriving before 10 pm
Speech Transcription	please find a flight <from> <saint> <petersburg> <excuse> <me> from detroit michigan to saint petersburg arriving before ten p m
Disfluent	please find a flight from st. petersburg excuse me from detroit michigan to st. petersburg arriving before 10 pm

Table 1: An example of a disfluent utterance transcribed in ATIS

mance on fluent data, while synthetic disfluencies do not. This demonstrates that adding synthetic but targeted disfluent training data can be more effective than adding real but noisier data to improve model performance on disfluent utterances.

2 ATIS

The Airline Travel Information System (ATIS) corpus (Price, 1990) is a widely used dataset in semantic parsing and spoken language understanding. ATIS contains human-computer dialogs in American English where speakers asked a computer system about hypothetical flight planning scenarios. In our experiments, we use the ATIS data splits from Tur et al. (2010)'s *All Train* setup with 4,978 training examples (of which 500 are for development) and 893 test examples selected from the wider corpus as those that are context independent. We use the target meaning representation format with slots and intents introduced by Gupta et al. (2018), with the same modification as Rongali et al. (2020) of custom end-brackets.

	Real	Synthetic
Contains Filled Pause(s)	43.1%	73.0%
Contains Repair(s)	68.7%	66.5%
Fluent avg token length	13.6	11.1
Disfluent avg token length	15.4	13.5

Table 2: Description of ATIS disfluencies

Unlike most semantic parsing datasets, the ATIS corpus contains recorded speech transcribed in detail, which has been used to research characteristics of disfluencies (Shriberg, 1996; Savova and Bachenko, 2003). However, most semantic parsing work on ATIS uses a cleaned version of the utterances where disfluencies are removed (Price, 1990). We use the original ATIS transcriptions to restore disfluencies to the dataset. We reintroduce deleted words in repairs, partial words longer than 1 character, and filled pauses ('mm', 'uh', 'ah', 'um'). Utterances identified as disfluent are manually verified. An example is shown in Table 1 and dataset statistics about the disfluencies are in Table 2. We map the disfluent utterances to the target meaning representation of their fluent counterpart. This results in 314 disfluent utterances in the train set, 36 in development, and 56 in the test set, accounting for 6.9% of the dataset.

3 Model Architecture

Hyperparameter	Value
Batch Size	1400
Learning Rate	0.05
Epochs	100
Max Seq Length	50
Beam Width	4

Table 3: Hyperparameters used when training semantic parsing models on the ATIS dataset

Our semantic parsing model architecture is a reimplementation of the sequence-to-sequence model with pointer generator network (Vinyals et al., 2015) proposed by Rongali et al. (2020), which achieved state-of-the-art performance on three public datasets including ATIS. We use a pre-trained RoBERTa language model (Liu et al., 2019) as the encoder and a transformer based on Vaswani et al. (2017) as the decoder. The encoder converts a sequence of words into a sequence of embeddings. Then at each time step, the decoder outputs either a symbol from the output vocabulary or uses a pointer generator network to generate pointers to tokens in the source sequence. A final softmax layer provides a probability distribution over all actions, and beam search maximizes the output sequence probability. We train on a single Nvidia Tesla v100 16GB GPU with the hyperparameters shown in Table 3.

4 Experiments

We run four experiments to evaluate how our semantic parsing model handles speech disfluencies. Each model is evaluated against three test sets: the *Original* ATIS test set of 893 fluent utterances, the *Disfluent* subset of 56 disfluent test set utterances identified in §2, and the *Fluent Subset* containing the fluent versions of these 56 utterances. The Fluent and Disfluent subsets are small, however we train multiple models and report results as the average exact-match accuracy over 5 runs with standard deviation. We also calculate slot-match accuracy for one model per setting on the disfluent subset, which measures whether the slots in the representation are correct. Finally, we provide a breakdown of performance based on disfluency type for a selection of models in Table 5.

4.1 Experiment 1: Are models trained on fluent data robust to disfluencies?

First, we investigate how a semantic parsing model trained on fluent data performs on disfluent utterances. We train a model with the original ATIS dataset and evaluate on all test sets. The results are in the first row of Table 4. These results show that our ATIS model is not robust to disfluencies, scoring 79% lower on the disfluent subset compared to the fluent subset (0.02 vs. 0.81) even though these are variations of the same utterances. In an error analysis, we find that most of the errors on the disfluent subset come from incorrect slots rather than intents, and so we also report the slot accuracy on the disfluent subset (0.29). Common errors include repairs within slots where they should be deleted (e.g. *[SL:TolocCityName um washington]*), or mislabeling filled pauses as airport or state codes.

4.2 Experiment 2: Can we improve performance on disfluencies with real disfluent data?

Next we experiment with adding real disfluent examples to improve model performance. We add the 314 disfluent examples identified in §2 into the train set and 36 into the dev set and fully retrain the model. As shown in the second row of Table 4, adding real disfluent examples improves model performance on the disfluent subset by 32% exact-match accuracy and 17% slot accuracy. This is promising, especially since only 7% of the train set was disfluent. However, this improvement comes at a cost: average performance drops by 6% on the original test set and by 4% on the fluent subset. We see errors where meaningful tokens are erroneously dropped from slots, such as omitting *"closest"* when parsing *"which airport is closest to ontario california"*, or *"city"* from *"kansas city"*.

4.3 Experiment 3: Can we improve performance on disfluencies with synthetic data?

Adding disfluent examples improved performance on the disfluent subset, but real disfluencies can be sparse and costly to collect with target labels, requiring expert annotators familiar with the meaning representation. Additionally, real examples can be complex with multiple disfluencies (e.g. *"beginning on april thirtieth and returning no beginning on april twenty fifth and beginning on may sixth returning on may sixth"*. The drop in performance on fluent examples observed in §4.2 could suggest that adding real disfluencies caused the model to overfit to noisier signals, thus leading to worse performance. We hypothesize that adding simpler and more targeted synthetic disfluencies may improve model performance. Previous studies have also illustrated the benefits of augmenting training data with generated synthetic data for disfluency detection and removal (Bach and Huang, 2019; Dong et al., 2019; Wang et al., 2019). And so our third experiment evaluates whether synthetically-generated disfluencies can improve model performance.

We generate synthetic disfluencies for each fluent utterance in the ATIS train and dev set. While real disfluencies can be affected by several factors, such as sentence length (Shriberg, 1994), word frequency (Hartsuiker and Notebaert, 2009; De Jong, 2016), and cognitive load (Oviatt, 1995), we try to generate disfluencies with a simple method. This lets us generate disfluencies quickly, and also makes our generation process easy to replicate and transfer to new domains and languages. We leave research on the generation of more complex and linguistically-guided disfluencies for future work.

For each fluent utterance *u* we randomly generate one of 3 disfluent variations:

1. **Filled Pause:** Add 1 random filled pause per 10 words inserted after a random token in *u*. Example: *"um show me flights from denver"*
2. **False start repair:** Select a random second utterance *u2*. Take the first *n* tokens of *u2* (where *n* is a random number between 1 and

	Dataset sizes		Exact match accuracy			Slot accuracy
	Train	Dev	Original	Fluent	Disfluent	Disfluent
ATIS	4478	500	0.85 ± 0.01	**0.81 ± 0.01**	0.02 ± 0.02	0.29
+ real	4792	536	0.79 ± 0.03	0.77 ± 0.04	0.34 ± 0.02	0.46
+ 1% synth	4523	505	0.85 ± 0.01	0.80 ± 0.02	0.20 ± 0.05	0.47
+ 5% synth	4702	525	0.85 ± 0.01	0.80 ± 0.02	0.31 ± 0.03	0.50
+ 10% synth	4926	550	0.85 ± 0.01	0.80 ± 0.01	0.34 ± 0.02	0.52
+ 25% synth	5598	625	0.85 ± 0.01	0.78 ± 0.01	0.37 ± 0.03	0.60
+ 50% synth	6717	750	0.85 ± 0.01	0.79 ± 0.02	0.39 ± 0.02	0.61
+ 100% synth	8956	1000	**0.86 ± 0.01**	0.79 ± 0.02	0.40 ± 0.04	**0.64**
100% synth	4478	500	0.79 ± 0.01	0.73 ± 0.01	**0.41 ± 0.01**	**0.64**
+ real + 1% synth	4837	541	0.78 ± 0.01	0.78 ± 0.02	0.33 ± 0.02	0.46
+ real + 5% synth	5016	561	0.79 ± 0.01	0.78 ± 0.02	0.33 ± 0.02	0.47
+ real + 10% synth	5240	586	0.79 ± 0.00	0.76 ± 0.03	0.32 ± 0.00	0.47
+ real + 25% synth	5912	661	0.78 ± 0.02	0.77 ± 0.03	0.33 ± 0.03	0.47
+ real + 50% synth	7031	786	0.79 ± 0.01	0.76 ± 0.03	0.30 ± 0.02	0.46
+ real + 100% synth	9270	1036	0.76 ± 0.03	0.73 ± 0.05	0.33 ± 0.02	0.54

Table 4: Results reported as averages over 5 runs ± the standard deviation. *ATIS* refers to fluent examples. *Original* is the original test set of 893 utterances. *Fluent* and *Disfluent* are the 56 fluent/disfluent counterparts from transcriptions. *Slot Accuracy* is the accuracy of predicted slots.

	FPs	Repairs
# of examples	18	34
ATIS	0.06	0.06
+ real	0.72	0.12
+ 50% synth	0.72	0.18
+ 100% synth	**0.78**	**0.24**
100% synth	0.78	0.26
+ real + 50% synth	0.72	0.15
+ real + 100% synth	0.78	0.21

Table 5: A breakdown of performance on the disfluent subset comparing Filled Pauses (FPs) vs. Repairs for select models. Examples containing both filled pauses and repairs are excluded.

5, no larger than 60% of the tokens in *u2*) and prepend them to *u*, along with a random interregnum (e.g. 'sorry'), filled pause, or nothing. Example: *"what's the earliest flight mm show me flights from denver to philadelphia"*

3 **Repeat repair:** Split *u* after a random character, ensuring the final token of the first part *p1* is at least 2 characters long. We repeat up to 2 of final full tokens in *p1* then append the remainder *p2* with the complete token if split. Example: *"show me flights from denver to phila denver to philadelphia"*

Statistics about the synthetic disfluencies are in Table 2. We incrementally add between 1% to 100% of the disfluent examples into the original train and dev sets and fully retrain the model. We also train a model on only synthetic disfluencies.

The results in Table 4 show that adding 10% of the synthetic disfluencies has comparable performance to adding the same number of real disfluencies on the disfluent subset and has less degradation on the fluent test sets. A model trained entirely on synthetic disfluencies only achieves 41% exact-match accuracy and 64% slot accuracy on the disfluent subset, indicating there is still work to be done in creating more robust models and better synthetic disfluencies. The improvements of the synthetic models on the disfluent subset come at little to no cost on the fluent test sets. This lack of degradation supports our hypothesis that real disfluencies may add more noise at training time than synthetic disfluencies.

In Table 5, we see that the performance boost from synthetic examples comes mostly from better handling of repairs. For disfluent utterances containing filled pauses, real disfluencies alone can get high exact-match accuracy (72%), while the addition of synthetic data increases this up to 78%.

For models trained with either real or synthetic disfluencies, performance for repairs is worse than for filled pauses, with 12% accuracy with real disfluencies and up to 26% accuracy with synthetic disfluencies. This suggests that repairs are generally harder for our semantic parser to handle. Still, synthetic disfluencies outperform real disfluencies on repairs by up to 13%. For example, the partial word "*phoe*" in "*flight between dallas and phoe phoenix*" is correctly dropped by synthetic models, whereas models trained on fluent or real disfluencies mistake "*phoe*" for an airport code.

4.4 Experiment 4: Combining real and synthetic disfluent data

In our final experiment, we investigate whether real disfluencies contribute alongside synthetic examples. We combine the fluent ATIS examples with the real ATIS disfluencies and then incrementally add between 1% to 100% of the synthetic examples. The results are in Table 4. These results show an initial exact-match accuracy advantage on the disfluent subset from using real and synthetic examples. Models with real disfluencies and 1% of the synthetic disfluencies score higher than 1% of the synthetic disfluencies alone (0.33 vs. 0.20), but synthetic disfluencies alone outperform after 10%. Performance on the original test set and fluent subset remain low. These results reveal that even if a dataset of real disfluencies is available, it can still be better to add only simpler synthetic disfluencies to training data to avoid adding noise to the model.

5 Conclusion

In this work, we evaluate semantic parsing of disfluent utterances in the ATIS dataset. We highlight the importance of considering spoken language phenomena when building semantic parsers by showing that a state-of-the-art semantic parsing model trained on fluent data is not robust to naturally occurring speech disfluencies. Although the ATIS dataset is small, we present preliminary results showing that adding real or synthetic disfluencies at training time can significantly improve performance by up to 39%. While adding real disfluencies comes with degradation in performance on fluent utterances, adding synthetic disfluencies improves performance on disfluent utterances with almost no degradation on fluent utterances and requires no additional costs for annotations or labeling. These findings show that adding simpler, targeted synthetic disfluencies can be a practical and effective way to improve a semantic parser's performance on disfluent utterances. With the release of more and larger spoken language semantic parsing datasets in the future, we hope to replicate and strengthen our findings in future work.

References

Nguyen Bach and Fei Huang. 2019. Noisy BiLSTM-Based Models for Disfluency Detection. In *Proc. Interspeech 2019*, pages 4230–4234.

Heather Bortfeld, Silvia D Leon, Jonathan E Bloom, Michael F Schober, and Susan E Brennan. 2001. Disfluency rates in conversation: Effects of age, relationship, topic, role, and gender. *Language and Speech*, 44(2):123–147.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: An embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Nivja H De Jong. 2016. Predicting pauses in L1 and L2 speech: The effects of utterance boundaries and word frequency. *International Review of Applied Linguistics in Language Teaching*, 54(2):113–132.

Qianqian Dong, Feng Wang, Zhen Yang, Wei Chen, Shuang Xu, and Bo Xu. 2019. Adapting translation models for transcript disfluency detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6351–6358.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Robert J Hartsuiker and Lies Notebaert. 2009. Lexical access problems lead to disfluencies in speech. *Experimental Psychology*.

Paria Jamshid Lou, Peter Anderson, and Mark Johnson. 2018. Disfluency detection using auto-correlational neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4610–4619, Brussels, Belgium. Association for Computational Linguistics.

Paria Jamshid Lou and Mark Johnson. 2020. End-to-end speech recognition and disfluency removal. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2051–2061, Online. Association for Computational Linguistics.

Paria Jamshid Lou, Yufei Wang, and Mark Johnson. 2019. Neural constituency parsing of speech transcripts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2756–2765, Minneapolis, Minnesota. Association for Computational Linguistics.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy-channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 33–39, Barcelona, Spain.

Stanislav V Kasl and George F Mahl. 1965. Relationship of disturbances and hesitations in spontaneous speech to anxiety. *Journal of Personality and Social Psychology*, 1(5):425.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sharon Oviatt. 1995. Predicting spoken disfluencies during human-computer interaction. *Computer Speech and Language*, 9(1):19–36.

Patti Price. 1990. Evaluation of spoken language systems: The ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! A sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, pages 2962–2968.

Guergana Savova and Joan Bachenko. 2003. Designing for errors: Similarities and differences of disfluency rates and prosodic characteristics across domains. In *Eighth European Conference on Speech Communication and Technology*.

Elizabeth Shriberg. 1996. Disfluencies in Switchboard. In *Proc. of International Conference on Spoken Language Processing, Addendum*, pages 11–14.

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis, University of California, Berkeley.

Jean E Fox Tree. 1995. The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech. *Journal of Memory and Language*, 34(6):709–738.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in ATIS? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3529–3538, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaolei Wang, Wanxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang. 2019. Multi-task self-supervised learning for disfluency detection. *arXiv preprint arXiv:1908.05378*.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional LSTM. *Interspeech 2016*, pages 2523–2527.

Joint Energy-based Model Training for Better Calibrated Natural Language Understanding Models

Tianxing He
MIT
cloudygoose@csail.mit.edu

Bryan McCann*
Salesforce Research
bryan@you.com

Caiming Xiong
Salesforce Research
cxiong@salesforce.com

Ehsan Hosseini-Asl
Salesforce Research
ehosseiniasl@salesforce.com

Abstract

In this work, we explore joint energy-based model (EBM) training during the finetuning of pretrained text encoders (e.g., Roberta) for natural language understanding (NLU) tasks. Our experiments show that EBM training can help the model reach a better calibration that is competitive to strong baselines, with little or no loss in accuracy. We discuss three variants of energy functions (namely *scalar*, *hidden*, and *sharp-hidden*) that can be defined on top of a text encoder, and compare them in experiments. Due to the discreteness of text data, we adopt noise contrastive estimation (NCE) to train the energy-based model. To make NCE training more effective, we train an autoregressive noise model with the masked language model (MLM) objective.

1 Introduction

Calibration refers to how well a classification model's confidence (reflected by its output posterior probability) aligns with its actual accuracy. As deep learning models achieve amazing accuracy in computer vision (He et al., 2015) or natural language processing (NLP) (Liu et al., 2019; Devlin et al., 2018), more research attention has been drawn to the calibration aspect of these models. As shown by Guo et al. (2017), the high accuracy from deep models does not always lead to better calibration. This motivates an important line of works attempting to achieve a better trade-off between accuracy and calibration.

Most existing calibration methods (Guo et al., 2017; Kumar et al., 2019; Zadrozny and Elkan, 2001) generally rescale the posterior distribution predicted from the classifier after training. Such post-processing methods require a held-out development set with a decent number of samples to be available. To overcome this constraint, Jung et al. (2020) uses a penalty term to encourage better calibration during training.

In another line of work, Grathwohl et al. (2019) shows that one can jointly train an energy-based model (EBM) during the standard training of a neural classifier. Although calibration is not explicitly addressed during EBM training, the calibration of the resulting model is shown to be greatly improved. Some intuitions of the underlying reasons will be given in Section 2.3. However, the training framework proposed by Grathwohl et al. (2019) is designed for image classifiers, and it can not be readily applied to discrete text data.

In this work, we propose a framework that uses noise contrastive estimation (NCE) to jointly train an energy-based model during the finetuning of pretrained text encoders (e.g., BERT (Devlin et al., 2018) or Roberta (Liu et al., 2019)) for NLU tasks. We compare several variants of energy functions that can be defined on top of the encoder. Our experiments show that the resulting models achieve competitive calibration results comparing to strong baselines, with little or no loss in accuracy.

2 Framework

2.1 Notations and Background

We focus on the finetuning of pretrained text encoder on NLU tasks. We assume samples from the data distribution P_D are in the form of (x, y) pairs, where x usually refers to a single or a pair of sentences, and y refers to the corresponding label. The number of classes are denoted by $|Y|$.

Given input x, we first use a text encoder model (e.g., BERT or Roberta) to encode it and we denote this embedding as $\text{enc}(x)$. For the target classification task, a classifier f_{CLS}, which could be a simple linear transform or a multi-layer perception (MLP), will be applied to $\text{enc}(x)$. We denote the output

---

* Bryan McCann contributed to this work while he was at Salesforce Research.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1754–1761
April 19 - 23, 2021.

logits as $f_{\text{CLS}}(\text{enc}(x))$, whose dimension is equal to the number of possible classes $|Y|$. The y-th logit is denoted by $f_{\text{CLS}}(\text{enc}(x))[y]$. The posterior distribution $P_\theta(y|x)$ is obtained by applying a softmax operation to the logits, where θ refers to the parameters in the model.

In standard finetuning, the cross-entropy (CE) loss and gradient based optimizers are used to train the classifier:

$$\mathcal{L}_{\text{CE}} = \underset{(x,y)\sim P_D}{\mathbb{E}} (-\log P_\theta(y|x)). \tag{1}$$

In the next few sections, we discuss how we define and jointly train an EBM on top of the text encoder.

2.2 Definitions of Energy Function

An energy-based model (LeCun et al., 2006) expresses $P_\theta(x)$ as:

$$P_\theta(x) = \frac{\exp(-E_\theta(x))}{Z}, \tag{2}$$

where Z is the normalization factor, and is usually intractable to compute. We refer to $E_\theta(x)$, which returns a scalar value, as *the energy function*. We now define three variants of energy functions.

Variant *scalar*: We introduce another linear layer g_{S} whose output is a scalar. And we use it to define the energy function:

$$\hat{E}_\theta(x) = g_{\text{S}}(\text{enc}(x)). \tag{3}$$

Variant *hidden*: As pointed out by Grathwohl et al. (2019), there's an EBM "hidden" in every neural classifier with softmax output, and the energy function for x can be derived[1] as:

$$\hat{E}_\theta(x) = -\text{LogSumExp}_{y=1}^{|Y|}(f_{\text{CLS}}(\text{enc}(x))[y]). \tag{4}$$

Different from the *scalar* variant, here the energy function directly uses the logits for prediction (visualized in Figure 1). Hence the impact on the model's classification behavior could be larger.

Variant *sharp-hidden*: The *hidden* variant has a potential weakness that, the correlation between input x and the prediction y is not addressed because the energy is distributed among all the logits. Motivated by this potential issue, we propose the following "sharp" variant:

$$\hat{E}_\theta(x) = -\max_y f_{\text{CLS}}(\text{enc}(x))[y]. \tag{5}$$

Note that (5) can be viewed as an approximation to (4), and we find it to work well in practice.

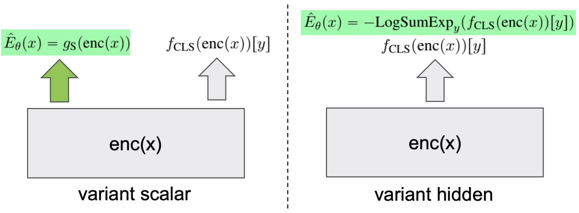


Figure 1: Comparison of the *scalar* and the *hidden* variants of energy functions. The modules introduced for EBM are shaded in green.

Finally, for each variant, we define the energy function to be $E_\theta(x) = \hat{E}_\theta(x) - \log P_N(x)$, where P_N is the *noise distribution* introduced for NCE. We will motivate this design choice below.

2.3 NCE Training

We use noise contrastive estimation (NCE) (Gutmann and Hyvärinen, 2010; Ma and Collins, 2018) to jointly train the energy model. NCE trains the model to discriminate between data samples and noise samples from a given noise distribution P_N. We formulate the NCE loss below:

$$\begin{aligned}\mathcal{L}_{\text{NCE}} = \underset{x_+\sim P_D}{\mathbb{E}} -\log \frac{\tilde{P}_\theta(x_+)}{\tilde{P}_\theta(x_+) + K \cdot P_N(x_+)} + \\ K \cdot \underset{x_-\sim P_N}{\mathbb{E}} -\log \frac{K \cdot P_N(x_-)}{\tilde{P}_\theta(x_-) + K \cdot P_N(x_-)},\end{aligned} \tag{6}$$

where K is the ratio of noise samples. Note that $\tilde{P}_\theta(x)$ does not need to be normalized by construction, therefore we set it to be $\tilde{P}_\theta(x) = \exp(-E_\theta(x))$. In our experiments, we mostly report results with noise ratio $K = 8$, while in some cases we find that a small ratio of $K = 1$ works slightly better. We have also tried with larger ratio such as 16, but the gain is minimal.

If we directly use the formulations of $\hat{E}_\theta(x)$ defined in last section as the energy function, the optimization will be difficult because of the $P_N(x)$ terms (which could be of very small value) in the NCE objective. To overcome this issue, we follow Deng et al. (2020) and define $E_\theta(x) = \hat{E}_\theta(x) - \log P_N(x)$. In this way, the $P_N(x)$ terms are canceled, and the objective is simplified to:

$$\begin{aligned}\mathcal{L}_{\text{NCE}} = \underset{x_+\sim P_D}{\mathbb{E}} -\log \frac{1}{1 + K \cdot \exp(\hat{E}_\theta(x_+))} + \\ K \cdot \underset{x_-\sim P_N}{\mathbb{E}} -\log \frac{K}{K + \exp(-\hat{E}_\theta(x_-))}.\end{aligned} \tag{7}$$

In training, we jointly optimize $\mathcal{L}_{\text{CE}}$ and $\mathcal{L}_{\text{NCE}}$ with the Adam optimizer (Kingma and Ba, 2014):

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{NCE}}. \tag{8}$$

---

[1] Please see Appendix A for the detailed derivation.

Intuitively, joint EBM training makes the model aware of $P(x)$, instead of only focusing on predicting $P(y|x)$ as in standard finetuning. This awareness can potentially help with calibration because the model can be more conservative when it detects the input is out-of-distribution.

2.4 Construction of Noise Distribution

For the choice of noise distribution P_N, in our preliminary trials, we finetune the GPT-2 language model (Radford et al., 2019) with samples from the target training set using the standard LM objective. However during NCE training, we find that the energy model can easily discriminate between data samples and noise samples, which makes training ineffective. To alleviate this issue, we adopt an objective similar[2] to the masked language model (MLM) loss (Devlin et al., 2018) during the finetuning of the noise model (GPT-2): With a given mask ratio M, we randomly mask part of x, and train the model to complete it:

$$\mathcal{L}_{\text{MLM}} = \mathop{\mathbb{E}}_{x \sim P_D, x^m \sim P_{\text{mask}}(x^m|x;M)} -\log P_{\text{N}}(x|x^m). \quad (9)$$

During noise sample generation, adopting the same mask ratio M, we feed a masked x^m to the LM (x is from the training set), and use the generated sample as the noise sample. In this way, the noise distribution is made closer to the data distribution. In our experiments we set $M = 0.4$. During generation, we use top-k (Fan et al., 2018) sampling with $k = 20$. More details are provided in Appendix B.

3 Experiments

Setting We consider finetuning the Roberta-base model[3], on eight GLUE tasks (Wang et al., 2018). We do not include results on STS-B because it is a regression task. To measure calibration error, we follow Jung et al. (2020); Grathwohl et al. (2019) and use the expected calibration error (ECE) metric with B (number of bins) set to 20. To save space, we defer detailed definition of ECE to Appendix C.

For baseline or NCE training, we follow the recommended hyper-parameters (learning rate, batch size, etc.) for Roberta (Liu et al., 2019). Since NCE training requires more computation (because of the noise ratio), we have tried finetuning the baseline with more steps, but we find that gives worse ECE and very little or no improvement on accuracy.

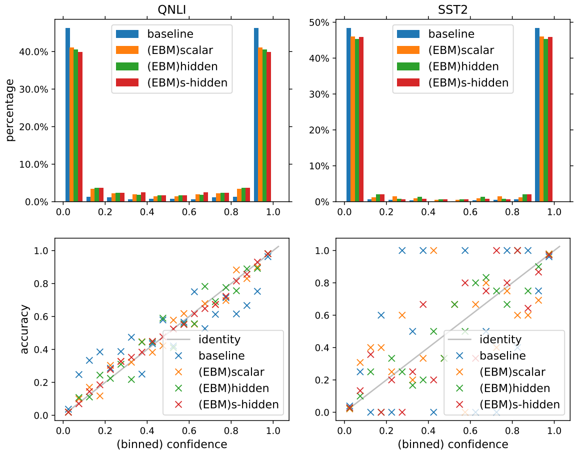


Figure 2: Visualization of calibration on QNLI and SST-2. In the histogram plots, we use 10 bins instead of 20 for better readability. An enlarged version of this figure is provided in Appendix D.

We compare EBM training with three strong baselines for calibration: *posterior calibrated training* (PosCal) (Jung et al., 2020), *temperature scaling* (T-Scal) (Guo et al., 2017), and *scaling-binning calibrator* (Scal-bin) (Kumar et al., 2019). For PosCal and Scal-bin, we use the published code.

Scal-bin and T-Scal require a development set for parameter learning and a test set for evaluation, but for each GLUE task we only have one labeled development set available. Therefore, in this work we treat half of the standard development set as test set, and keep the other half as development set.

Results In Table 1 and Table 2 we compare test-set accuracy[4] and ECE for different methods on the GLUE tasks. For fair comparison between Scal-bin / T-Scal and EBM training (which does not use the development set), we apply them to the whole training set. We also report their performance when applied to the development set for reference.

In most tasks, all three EBM variants get substantial improvement in ECE with little or no loss in accuracy comparing to the (strong) baseline methods. Moreover, the performance of EBM training is comparable to Scal-bin / T-Scal applied to the development set, while their performance degrades when the development set is not available. Among the three variants, on average, the *sharp-hidden* variant achieves the best accuracy, while the *hidden* variant achieves best calibration. We visualize the calibration error in Figure 2.

---

[2] The difference is that we still train the model to generate the full sentence, instead of only the masked words.

[3] Our code is based on `https://github.com/huggingface/transformers`.

[4] For CoLA we report with Matthews correlation coefficient (mcc).

Method	SST-2 acc.	 ECE	MNLI acc.	 ECE	MNLI(mm) acc.	 ECE	QNLI acc.	 ECE	QQP acc.	 ECE	MRPC acc.	 ECE	CoLA mcc.	 ECE	Average perf.	 ECE
Baseline	.942	.050	.876	.067	.872	.068	.929	.043	.904	.034	.862	.133	.539	.182	.802	.102
Scal-bin(train)	.940	.036	.872	.051	.869	.056	.931	.034	.904	.035	.843	.092	.586	.146	.791	.096
T-Scal(train)	.942	.042	.876	.058	.872	.060	.929	.030	.904	.034	.862	.126	.539	.175	.802	.096
PosCal	.944	.040	.876	.067	.872	.067	.930	.039	.905	.032	.867	.129	.540	.184	.810	.092
(EBM)scalar	.942	.033	.871	.038	.871	.047	.927	.016	.899	.034	.862	.098	.540	.150	.801	.073
(EBM)hidden	.956	**.032**	.869	.032	.868	.044	.923	.016	.900	.033	.867	.099	.545	**.131**	.807	.063
(EBM)s-hidden	.947	.038	.875	**.027**	.872	**.031**	.930	**.016**	.900	**.032**	.862	**.089**	.563	.133	.815	.069
Scal-bin(dev)	.944	.019	.876	.030	.870	.032	.931	.021	.905	.021	.862	.062	.557	.048	.802	.052
T-Scal(dev)	.942	.037	.876	.024	.872	.026	.929	.018	.904	.026	.862	.126	.539	.109	.802	.072

Table 1: Test-set accuracy and ECE results for different methods on GLUE tasks. "s-hidden" refers to the *sharp-hidden* variant. The leading zeros are omitted to save space. Note that T-Scal and Scal-bin are applied to the training set or the development set, respectively. Due to space constraint, results on RTE and WNLI are deferred to Table 2. The average value is compute on all nine test sets. For each task, the method that achieves best calibration without using the development set are shown in bold.

Method	RTE acc.	 ECE	WNLI acc.	 ECE
Baseline	.724	.279	.571	.058
Scal-bin(train)	.717	.271	.457	.144
T-Scal(train)	.724	.275	.571	.063
PosCal	.789	.206	.571	.060
(EBM)scalar	.753	.207	.542	**.033**
(EBM)hidden	.797	**.148**	.542	.036
(EBM)s-hidden	.811	.182	.571	.073
Scal-bin(dev)	.731	.042	.542	.189
T-Scal(dev)	.724	.235	.571	.046

Table 2: (Following Table 1) Main results on RTE and WNLI.

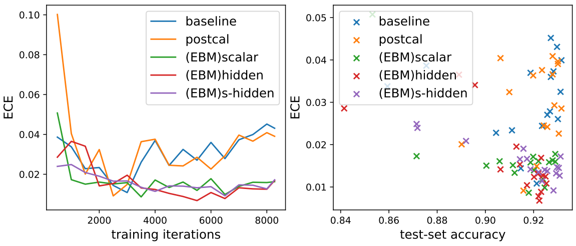


Figure 3: (QNLI) Left: How ECE changes during training. Right: The trade-off between accuracy and ECE for checkpoints (every 500 iterations) during training.

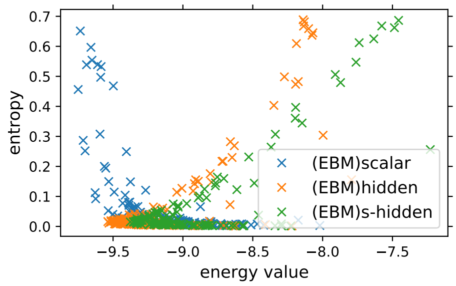


Figure 4: The entropy of the posterior ($P_\theta(\cdot|x)$) versus energy value $\hat{E}_\theta(x)$ for SST-2 test-set samples.

Text: *when the film ended, i felt tired and drained and wanted to lie on my own deathbed.* Label: 1 $\hat{E}_\theta(x)$: -9.37 Baseline: (.999, .001) → EBM: (.998, .002)
Text: *sit through this one, you won't need a magic watch to stop time; your dvd player will do it for you.* Label: 1 $\hat{E}_\theta(x)$: -7.57 Baseline: (.006, .994) → EBM: (.345, .655)

Table 3: The change of the model's confidence (posterior distribution) for low and high-energy data samples of SST-2. The EBM variant shown is *sharp-hidden*. We also provide QNLI examples in Appendix D.

In Figure 3, we plot how test-set ECE changes during training. It is shown as the training reaches the high-accuracy area, the calibration for baseline model becomes worse, while EBM training is able to reach a better trade-off between accuracy and calibration.

How does the model get better calibration? In Figure 4, we compute and plot the energy value $\hat{E}_\theta(x)$ versus the entropy of the posterior distribution $\mathcal{H}(P_\theta(\cdot|x)) = \sum_{y=1}^{|Y|} -P_\theta(y|x)\log P_\theta(y|x)$, for samples in the SST-2 test set. It is shown that models trained with the *hidden* and *sharp-hidden* variants tend to assign more conservative predictions (reflected by higher entropy) for higher-energy (less likely) samples. We suspect this is due to the strong coupling between the energy function and the classification logits. We provide concrete examples in Table 3. However, we need to mention that we do not observe this interesting trend (Figure 4) in all datasets (e.g., QNLI).

4 Related Works

Finally, we review applications of NCE or energy-based models in the NLP literature. Due to its self-normalizing property, NCE training has been used for faster inference (Mnih and Teh, 2012; Chen et al., 2015; Labeau and Allauzen, 2018) of autoregressive language models. It has also been used in an attempt to train a sentence-level bi-directional LM (He et al., 2016).

More closely related to our work, Deng et al. (2020) adopts NCE to train an EBM defined on top of a text encoder (the *scalar* variant), and uses it to improve language generation. EBM has also been recently used in non-autoregressive machine translation (Tu et al., 2020).

5 Conclusion

In this work, we explore joint EBM training during the finetuning of pretrained text encoders with noise contrastive estimation. We find that joint EBM training can greatly improve the calibration of NLU models, with little or no loss on accuracy.

References

Xie Chen, Xunying Liu, Mark J. F. Gales, and Philip C. Woodland. 2015. Recurrent neural network language model training with noise contrastive estimation for speech recognition. In *ICASSP*, pages 5411–5415. IEEE.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020. Residual energy-based models for text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. 2019. Your classifier is secretly an energy based model and you should treat it like one.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330, International Convention Centre, Sydney, Australia. PMLR.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy. PMLR.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

Tianxing He, Yu Zhang, Jasha Droppo, and Kai Yu. 2016. On training bi-directional neural network language model with noise contrastive estimation. *CoRR*, abs/1602.06064.

Taehee Jung, Dongyeop Kang, Hua Cheng, Lucas Mentch, and Thomas Schaaf. 2020. Posterior calibrated training on sentence classification tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2723–2730, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Ananya Kumar, Percy S Liang, and Tengyu Ma. 2019. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems 32*, pages 3792–3803. Curran Associates, Inc.

Matthieu Labeau and Alexandre Allauzen. 2018. Learning with noise-contrastive estimation: Easing training by learning to scale. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3090–3101, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yann LeCun, Sumit Chopra, Raia Hadsell, Fu Jie Huang, and et al. 2006. A tutorial on energy-based learning. In *PREDICTING STRUCTURED DATA*. MIT Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Zhuang Ma and Michael Collins. 2018. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3698–3707, Brussels, Belgium. Association for Computational Linguistics.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 419–426, Madison, WI, USA. Omnipress.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Lifu Tu, Richard Yuanzhe Pang, Sam Wiseman, and Kevin Gimpel. 2020. ENGINE: Energy-based inference networks for non-autoregressive machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2826, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Bianca Zadrozny and Charles Elkan. 2001. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *In Proceedings of the Eighteenth International Conference on Machine Learning*, pages 609–616. Morgan Kaufmann.

Appendices

A Derivation of the *hidden* Variant

Remember from Section 2.1, the posterior distribution is obtained from a softmax operation on the logits, in other words:

$$P_\theta(y|x) \propto \exp(f_{\text{CLS}}(enc(x))[y]). \tag{10}$$

Without changing any parameters, one can reuse the logits to define an energy based model of the joint distribution of data point x and labels y via:

$$P_\theta(x, y) = \frac{\exp(f_{\text{CLS}}(enc(x))[y])}{Z(\theta)}, \tag{11}$$

where $Z(\theta)$ is the normalizing factor. Note that Equation 11 is **consistent** with Equation 10 in the sense that Equation 10 is a direct consequence of Equation 11.

Now by marginalizing out y, we get:

$$P_\theta(x) = \frac{\sum_{y=1}^{|Y|} \exp(f_{\text{CLS}}(enc(x))[y])}{Z(\theta)}, \tag{12}$$

which is equivalent to

$$P_\theta(x) = \frac{\exp(-E_\theta(x))}{Z(\theta)}, \tag{13}$$

where

$$E_\theta(x) = -\text{LogSumExp}_y(f_{\text{CLS}}(\text{enc}(x))[y]). \tag{14}$$

For more intuition behind this derivation we refer readers to Grathwohl et al. (2019).

B Details About the Noise Distribution

We show some examples of generated noise samples and the masking in Table 4. Note that the masks could be applied to a consecutive span of words (Masking is applied to each token independently with probability M).

Input: absolutely and completely `<M>` `(ridiculous)` **Gen:** absolutely and completely hilarious
Input: `<M>` `(as a)` young `<M>` `(woman)` of great charm, `<M>` `(generosity)` and diplomacy **Gen:** of a young man with a great charm, wit and diplomacy

Table 4: Example of generated noise samples on SST-2. The original words that are masked are also shown.

Another possible way to get noise samples is that we can sample from BERT or Roberta with masked input. However, due to the nature of masked language modeling and the architecture of BERT / Roberta, the sampled tokens will be independent of each other, which could result in unnatural noise samples. That is why we choose to utilize an auto-regressive LM (e.g., GPT-2).

C Definition of ECE

Given an input sample x, for each label y, we say that the model predicts that x belongs to label y with confidence $P_\theta(y|x)$. Assuming the test-set contains n samples, we will have $n \times |Y|$ predictions.

ECE first partitions all predictions into B equally-spaced bins by its confidence. Following Jung et al. (2020); Grathwohl et al. (2019), we set $B = 20$, which means the width of each bin is 0.05. For example, the first bin contains all predictions that have confidence in the range of $[0, 0.05)$. Then for each bin ECE computes how the average of confidence is different from its actual accuracy:

$$\text{ECE} = \frac{1}{|Y|} \sum_{y=1}^{|Y|} \sum_{b=1}^{B} \frac{|B_{yb}|}{n} |\text{acc}(B_{yb}) - \text{conf}(B_{yb})|, \tag{15}$$

where n is the number of samples in the test set, and $\text{acc}(B_{yb})$ is simply the ratio of samples (x) whose true label is indeed y in B_{yb}.

D Auxiliary Results and Examples

Examples of the model's confidence for low and high-energy data samples in QNLI are shown in Table 5.

The histogram of energy values $\hat{E}_\theta(x)$ for samples in the test set of QNLI and SST-2 are shown in Figure 5.

In Figure 6, we provide an enlarged version of Figure 2.

Text: *Q: What city north of New York was settled by Huguenots? A: Huguenot immigrants did not disperse or settle in different parts of the country, but rather, formed three societies or congregations; one in the city of New York, another 21 miles north of New York in a town which they named New Rochelle, and a third further upstate in New Paltz.* Label: 1 $\hat{E}_\theta(x)$: -8.48 Baseline: (.997, .003) → EBM: (.995, .005)
Text: *Q: What is the source of oxygen production through electrocatalytic means? A: A similar method is the electrocatalytic O2 evolution from oxides and oxoacids.* Label: 1 $\hat{E}_\theta(x)$: 4.22 Baseline: (.252, .748) → EBM: (.472, .527)

Table 5: The change of the model's confidence (posterior distribution) for low and high-energy data samples in the test set of QNLI. The EBM variant shown is *sharp-hidden*.

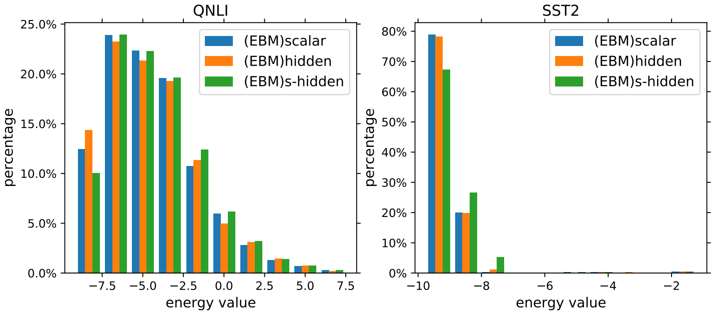


Figure 5: The histogram of energy values $\hat{E}_\theta(x)$ for samples in the test set of QNLI and SST-2.

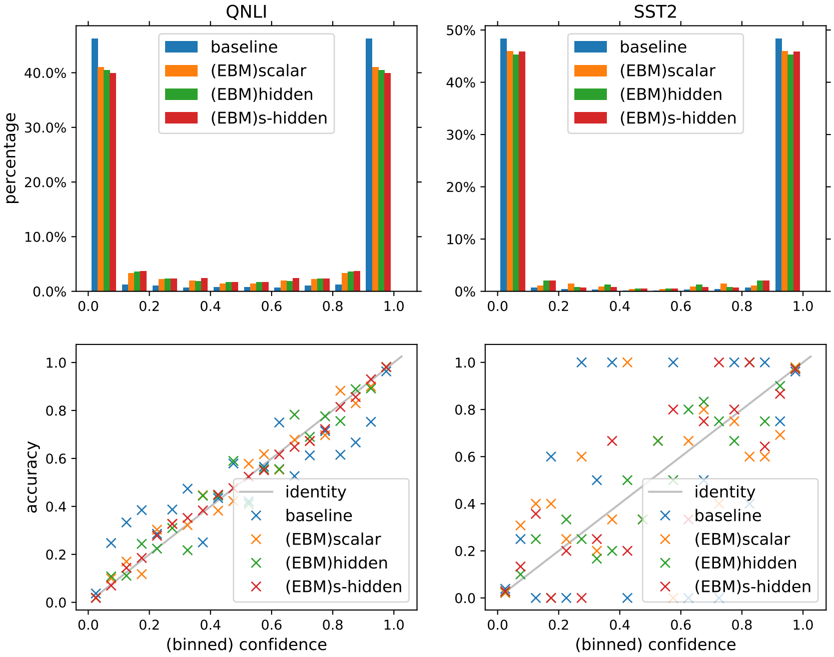


Figure 6: Visualization of calibration on QNLI and SST-2. Enlarged version.

What Sounds “Right” to Me? Experiential Factors in the Perception of Political Ideology

Qinlan Shen
Carnegie Mellon University
qinlans@cs.cmu.edu

Carolyn P. Rosé
Carnegie Mellon University
cprose@cs.cmu.edu

Abstract

In this paper, we challenge the assumption that political ideology is inherently built into text by presenting an investigation into the impact of experiential factors on annotator perceptions of political ideology. We construct an annotated corpus of U.S. political discussion, where in addition to ideology labels for texts, annotators provide information about their political affiliation, exposure to political news, and familiarity with the source domain of discussion, Reddit. We investigate the variability in ideology judgments across annotators, finding evidence that these experiential factors may influence the consistency of how political ideologies are perceived. Finally, we present evidence that understanding how humans perceive and interpret ideology from texts remains a challenging task for state-of-the-art language models, pointing towards potential issues when modeling user experiences that may require more contextual knowledge.

1 Introduction

Social media companies, like Twitter, Facebook, and Reddit, play an important role in political discourse by providing a space for users to interact with different viewpoints. Understanding political discussion on these platforms often requires one to identify the ideologies behind texts, as understanding the viewpoints reflected in a text can provide insight into the partisanship of beliefs (Monroe et al., 2008) or the persuasive strategies used by different ideological groups (Tsur et al., 2015).

Prior research on political discussion often relies on a ”ground-truth” to aid in obtaining ideology labels for social media data. For example, due to the scale of political content on social media, a common paradigm is to obtain some ground-truth labels that are propagated to a larger set of texts using semi-supervised learning (Lin and Cohen, 2010; Zhou et al., 2011). The relationship between a social media artifact and various forms of established political knowledge can also be used to ground or validate ideology labels. Some examples of this include using author interactions with politicians with known party affiliations (Djemili et al., 2014; Barberá, 2015), ideological communities (Chandrasekharan et al., 2017; Shen and Rosé, 2019), and central users (Pennacchiotti and Popescu, 2011; Preotiuc-Pietro et al., 2017) as a starting heuristic, or evaluating a labeling approach by comparing geolocation tags attached to posts with historical voting patterns (Demszky et al., 2019).

A limitation of these approaches, however, is that behavior on social media does not evenly or uniformly reflect the held political beliefs of participants. While there is evidence that people tend to engage with others who share similar beliefs (Halberstam and Knight, 2016), people also commonly interact with or even seek out communities and users they do not agree with (Kelly et al., 2005; Tan et al., 2016). Additionally, the practice of displaying one’s political beliefs, which many grounding techniques rely on, varies in prevalence across online communities (Lampe et al., 2007; Zhong et al., 2017; Pathak, 2020). The concept of linguistic agency (Goffman et al., 1978) also challenges the idea that individual factors, such as ideology, are predictably presented in text. Based on an author’s social goals for participating in political discussion, it may not be contextually relevant to project a strong impression of their political ideology. People engaged in interactive political discussion, however, still form perceptions about the alignments of others based on how they sound, often relying on their own conceptions of ideology in the process.

The issue of perceiving ideology also plays a role when ideology labels are obtained using crowdsourced annotators. While making judgments, the

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1762–1771
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

annotator plays a similar role to a user participating in the discussion when perceiving the ideology of the speaker behind a text. However, annotators are expected to assign an explicit ideology label to a text with less contextual knowledge about how the text was produced. Thus, annotators may rely heavily on their own *experiential* factors, such as one's own beliefs or level of political engagement, when considering ideology. As a result, this process may introduce inconsistencies and biases in ideological labels used for political analysis.

In this paper, we present an exploration of how experiential factors play a role in how annotators perceive ideology in text. Building upon prior work investigating annotation bias (Zaidan and Callison-Burch, 2011; Waseem, 2016; Joseph et al., 2017; Ross et al., 2017; Schwartz et al., 2017; Geva et al., 2019), we construct an annotated corpus of posts from political subcommunities on Reddit but incorporate additional contextual information about the annotators making ideology judgments.[1] While previous work (Joseph et al., 2017) has shown that source-side contextual features, such as user profiles and previous tweets, can influence label quality in stance annotation, we focus our analyses on contextual factors on the side of annotators. Most similar to our work, Carpenter et al. (2017) and Carpenter et al. (2018) examine the impact of an annotator's identity and openness on their ability to accurately assess author attributes, including political orientation. In our work, however, we examine the impact of an annotator's political beliefs, knowledge, and Reddit familiarity, on their judgments, using factors more specific to political participation on Reddit. We additionally consider the issue of annotator bias in ideology labeling not as an issue of accuracy but rather an issue of social variability. Under this view, we evaluate the performance of a state-of-the-art language model on its capacity to mirror different human perceptions of ideology to examine whether extralinguistic factors introduced through annotation may degrade model performance compared to other labels.

2 Dataset Construction

Our dataset is drawn from the popular content aggregation and discussion platform Reddit. Political discussion on Reddit is centered on *subreddits*, subcommunities centered on support for specific political candidates, organizations, and issues. For our analyses, we aim to label political distinctions on Reddit along the left-right political spectrum in U.S. politics. Using the monthly dumps from May to September 2019 from the Reddit Pushshift API (Baumgartner et al., 2020), we collect all submissions and comments from the top political subreddits[2] by subscriber count. The collected subreddits were manually labeled as left or right, based on the subreddit description and top posts. We then select the top 12 left and top 12 right subreddits from the monthly dumps where discussion is primarily focused on U.S. politics.[3] The selected subreddits are shown in Table 3 (Supplementary Material).

2.1 Paired Ideology Ranking Task

Prior work on annotating viewpoints (Iyyer et al., 2014; Bamman and Smith, 2015) generally presents annotators with texts in isolation to label with an ideology of interest. One drawback of this approach is the high degree of political expertise annotators are required to have to recognize that a text matches an ideology. To reduce the amount of overhead in recruiting and training political annotators, we present annotators instead with a paired ideology ranking task. Rather than examining texts in isolation, annotators are shown two texts and asked to select the text that is more likely to be authored by someone with the ideology of interest.

For our setup, our goal is to pair a text authored by a left-leaning user with one by a right-leaning user. We use a heuristic-based semi-supervised approach to label texts based on the subreddit participation patterns of their authors. To expand the set of subreddits with ideological labels, we label all subreddits in the monthly dump data as left, neutral, or right based on user overlap with the 24 political subreddits with a known ideological slant (Section 2). For each subreddit, we calculate the z-score of the log odds ratio of a user participating in that subreddit and a known left-leaning subreddit vs. a right-leaning subreddit. A subreddit is labeled as either "left" or "right" if the calculated z-score satisfies a one-tailed Z test at $p = 0.05$ in the corresponding direction or "neutral" otherwise. Authors are then labeled based on their distribution of participation on the left vs. right subreddits. While

---

[1] This study was approved by the institutional review board at our institution.

[2] `https://www.reddit.com/r/redditlists/comments/josdr/list_of_political_subreddits/`

[3] r/politics was not included due to its initial history as a default subreddit contributing to its high subscriber count.

users on Reddit have been shown to primarily engage with pro-social home communities (Datta and Adar, 2019) and similar heuristics have been used in prior work as an indicator of user interests and/or ideology (Olson and Neal, 2015; Chandrasekharan et al., 2017; Shen and Rosé, 2019), we emphasize that we use this heuristic to create a basis of comparison, rather than assuming that it provides "correct" ideology labels.

In order to ensure that the text comparison helps annotators to perceive ideological differences, rather than presenting two unrelated texts that are essentially considered in isolation, we want to present paired texts that are similar in content. As a first step for generating comparisons with similar content, we require paired texts to discuss the same entity, since political discussions are primarily centered on the politicians, organizations, and geopolitical entities influencing policy decisions. To identify entities of interest, we use Stanford Core NLP (Manning et al., 2014) to extract occurrences of people, locations, organizations, and ideologies over our corpus of 24 subreddits. We limit entities under consideration to those that have occurred at least 300 times in our corpus and are easy to disambiguate. The considered entities are shown in Table 4 (Supplementary Material).

To limit the impact of confounds, such as topic or entity salience, when comparing texts with the same entity, we use propensity score matching (Rosenbaum and Rubin, 1983) to match each left-aligned text with a right-aligned text that discusses the same entity in a similar context. A subset of 65 pairs was manually curated to use as screening questions to ensure that workers had a baseline knowledge of U.S. politics. These screening pairs were selected to be easier than the main task pairs – they are more limited in which entities discussed and express more explicit and/or extreme attitudes.

2.2 Annotation Task Details

We recruit workers on Amazon Mechanical Turk to complete our paired ideological ranking task. Given a pair of texts discussing the same highlighted political entity, we ask annotators to determine which of the two posts is more likely to have been written by someone who is either left-leaning or right-leaning. Annotators were instructed to use as many contextual cues as possible to form an impression of the political views held by the authors of the texts. To provide some guidance to annotators for what cues to consider, we train workers to consider the following features in the instructions:

- **Attitude:** evaluation in favor of or against an entity. Ex: *I trust Bernie* from someone who favors Bernie Sanders (left).
- **Positioning:** situating one's viewpoint with respect to the entity's. Ex: *Listen to the Dems* refers to Democrats as an out-group (right).
- **Jargon:** use of speciality in-group vocab. Ex: *Trump GEOTUS!* – "God-Emperor" abbreviation specific to Trump supporters (right).

The annotation task is shown in Figure 1 (Supplementary Material). Each worker was asked to annotate 18 pairs from our main task set and 8 screening questions, which were scattered throughout the assignment as an attention check. For each main task pair, we assign up to 5 workers for annotation. We restrict the worker pool to the U.S. and filter out workers who scored less than a 75% on the screening questions. Overall, we collect annotations for 630 non-screening pairs.

2.3 Annotator Background Post-Survey

After the annotation task, workers were asked to complete a survey (questions listed in Supplementary Material A) to assess their political affiliation, exposure to U.S. political news, and familiarity with political discussion on Reddit. Answers to the survey were inspected manually to assign annotators labels along three identifier categories:

- **Political ideology:** This category indicates the annotator's political ideology. Annotators are labeled as *left*, *center*, or *right* based on their self-identified ideology and affiliation with U.S. political parties.
- **News access:** This category indicates the annotator's exposure to political news. Annotators are labeled as *news* or *non-news* based on how frequently they access news on the 2020 U.S. presidential election.
- **Reddit familiarity:** This category indicates the annotator's familiarity with participation in political discussion on Reddit. Annotators are labeled as a *redditor* or a *non-redditor* based on their level of participation on Reddit in the past year. Redditors are further subdivided into *political* and *non-political* redditors based on their familiarity with the political subreddits included in our corpus.

		# workers	α
Overall	-	158	0.388
Ideology	left	89	0.427
	right	43	0.372
	center	26	0.325
News	news	126	0.393
	non-news	32	0.336
Reddit	redditor	114	0.393
	–political	86	0.389
	–non-political	28	0.430
	non-redditor	44	0.359

Table 1: Number of workers and Krippendorff's α agreement within the annotator groups over the full non-screening set. Agreement over other question sets can be found in Table 5 (Supplementary Material)

3 Dataset Statistics and Analysis

3.1 Annotator Demographics

Of the 180 recruited workers initially recruited for the task, 22 were discarded for answering fewer than 75% of the screening questions correctly, giving us a final pool of 158 annotators. Table 1 illustrates the distribution of the remaining workers across labels within the three categories. Labels across categories do not appear to be correlated (mean variance inflation factor = 1.043).

3.2 Agreement/Difference Results

We use Krippendorf's α (Krippendorff, 2004) to evaluate annotator agreement on our task to account for different user pools for each question. Despite a high degree of agreement across the pool of screening questions ($\alpha = 0.7311$), the overall agreement across annotators in our general, non-screening set is relatively low ($\alpha = 0.3878$), suggesting that the task of predicting the ideology of a text is nuanced and open to interpretation.

We also calculate agreement for workers within each of our annotator groups (Table 1) in order to examine whether annotators with similar backgrounds are more likely to perceive ideology similarly. Overall, in-group agreement remains around the same level as the general task. However, an interesting pattern across annotator labels is that workers who are less likely to be familiar with the expression of political ideology on Reddit – non-redditors ($\alpha = 0.359$), people who do not frequently read political news ($\alpha = 0.336$), and people who do not identify with the left or right ($\alpha = 0.325$) – have lower agreement. This suggests that familiarity with the norms of political discussion on Reddit may contribute to a more consistent perception of ideology for Reddit texts.

We additionally use McNemar's chi-squared test over pairwise comparisons of annotator groups under the same category to examine whether annotators with different backgrounds differ in their judgments. To ground the comparison, we evaluate annotator groups based on whether the majority of workers in the group gave the same answer as our semi-supervised labels (Section 2.1). Because these semi-supervised labels only provide a noisy estimate of ideology, we use these labels to create a basis of comparison. Rather than to check how "accurately" each group estimates ideology, this heuristic allows us to specifically quantify differences in judgments between groups. We find that for all comparison pairs, groups differ significantly in their answers over the same questions. In our pairwise comparisons, we also saw that the ideology of the annotator contributes heavily to variability in annotator judgments. The two groups with the highest percentage of questions with mismatched answers are left-leaning and right-leaning annotators, and 3 of the top 4 comparison pairs with the most mismatched answers are between ideology groups (Supplementary Material Table 6).

3.3 Sources of Variability

To examine possible explanations for the variability in annotator judgments across groups, we focus primarily on differences in judgments between left-leaning and right-leaning annotators. When examining differences at the entity-level, we find that the entities with the most mismatches tended to be highly visible entities that had a strong connection to a particular party during the 2020 election, such as highly visible political figures (e.g. Joe Biden, Nancy Pelosi) or the most common ideologies associated with each side (e.g. Republican Party, conservatism, liberalism), compared to less salient entities. This is unsurprising, as we expect people to develop different conceptions of salient entities building up to major events like elections, even with relatively limited media exposure.

Finally, to investigate what aspects of the posts themselves contributed to variations in judgments between left-leaning and right-leaning workers, we ran a salience (Monroe et al., 2008) analysis for mismatched question pairs with highly visible entities. We found that annotators were less likely to select a post that expresses explicit abuse towards an opposing entity as being authored by someone with the same political views as themselves. For example, a right-leaning annotator was less likely to consider a post calling Biden a "pedophile" as right-leaning compared to liberal annotator. This may suggest that social desirability bias (Krumpal, 2013), may have an impact on decision-making, even when the task is not directly related to collecting data about the annotator themselves.

4 Perceptions vs. Heuristic Labels

Prior work (Castelle, 2018) suggests that deep text classification models perform poorly when labels are influenced by extralinguistic contextual factors. While the semi-supervised labels that we generated are based on a behavioral heuristic outside of the text, our analyses of human judgments suggest that the annotation process introduced additional interactional factors into ideological labeling. We investigate whether these factors influence model performance by evaluating a BERT-based (Devlin et al., 2019) model on its ability to match human judgments on the paired ideology ranking task.

For our evaluation model, we finetune BERT-mask on the 24 subreddit corpus. Next, for each text, we average its contextual embeddings in two ways: over (a) all tokens in the text and (b) all entity-related tokens in the text. We then concatenate the averaged embeddings, then use the resulting vector as input to a pairwise logistic regression model. For each annotator group, we use the majority answer for each question as the group label.

Table 2 shows the performance of the model on the full 630 pair non-screening set. For all annotator groups, we found that the model has a significant drop in performance when asked to match human judgments vs. labels generated through our semi-supervised heuristic on the same dataset. To examine whether this drop in performance was due to inconsistencies in human judgments on particularly difficult or contentious distinctions, we additionally present results on a higher consensus subset ($\alpha = 0.6216$) of 459 text pairs, where at least 75% of workers select the same answer. We found that while there was a small increase in performance on matching human judgments on the high consensus subset for all groups, performance still dropped compared to the semi-supervised labels, suggesting that matching human understanding of ideology is challenging for these models.

	SS (F)	**H (F)**	**SS (C)**	**H (C)**
Overall	69.28*	56.82	70.16*	58.62
left	68.60*	60.13	69.75*	63.00
right	65.97*	53.41	69.51*	56.49
center	63.81*	52.58	70.41*	60.27
news	67.48*	57.43	68.86*	59.43
non-news	66.01*	54.03	67.89	63.27
redditor	70.89*	57.65	69.46*	59.95
–political	70.59*	57.25	67.90*	59.01
–non-political	69.62*	57.01	72.63*	58.57
non-redditor	65.89*	51.65	64.06*	55.49

Table 2: F1 scores for a BERT-based ranking model on semi-supervised (**SS**) and human annotator (**H**) labels for the full non-screening set (**F**) and a high-consensus subset (**C**). $*p < 0.05$ difference in performance between the semi-supervised and human annotator labels

5 Conclusion and Future Work

In this paper, we reconsider the idea of ground-truth labels of political ideology and investigate the impact of experiential factors on human perception of ideology in text. We construct and analyze an annotated corpus that incorporates experiential information about annotators, finding evidence that annotator backgrounds influence the consistency of political ideology judgments and that current classification models struggle to match human perceptions of ideology across different groups. From our analyses on factors contributing to variations in judgments, there is a greater need for targeted recruiting of annotators that are familiar with and contextualized to the domain being annotated. In future work, we aim to extend our investigation to examine how stylistic elements of text contribute to people's perception of political ideologies in interaction. These analyses may provide further insight into the effectiveness of political communication strategies or the differences in how political groups interact with in-group and out-group members.

Acknowledgements

This work was supported in part by NSF Grant IIS 1546393 and the K&L Gates Presidential Fellowship.

References

David Bamman and Noah A Smith. 2015. Open extraction of fine-grained political statements. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 76–85.

Pablo Barberá. 2015. Birds of the same feather tweet together: Bayesian ideal point estimation using Twitter data. *Political Analysis*, 23(1):76–91.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 830–839.

Jordan Carpenter, Daniel Preotiuc-Pietro, Jenna Clark, Lucie Flekova, Laura Smith, Margaret L Kern, Anneke Buffone, Lyle Ungar, and Martin Seligman. 2018. The impact of actively open-minded thinking on social media communication. *Judgment and Decision Making*, 13(6):562.

Jordan Carpenter, Daniel Preotiuc-Pietro, Lucie Flekova, Salvatore Giorgi, Courtney Hagan, Margaret L Kern, Anneke EK Buffone, Lyle Ungar, and Martin EP Seligman. 2017. Real men don't say "cute" using automatic language analysis to isolate inaccurate aspects of stereotypes. *Social Psychological and Personality Science*, 8(3):310–322.

Michael Castelle. 2018. The linguistic ideologies of deep abusive language classification. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 160–170.

Eshwar Chandrasekharan, Umashanthi Pavalanathan, Anirudh Srinivasan, Adam Glynn, Jacob Eisenstein, and Eric Gilbert. 2017. You can't stay here: The efficacy of reddit's 2015 ban examined through hate speech. *Proceedings of the ACM on Human-Computer Interaction*, 1(CSCW):1–22.

Srayan Datta and Eytan Adar. 2019. Extracting intercommunity conflicts in reddit. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 146–157.

Dorottya Demszky, Nikhil Garg, Rob Voigt, James Zou, Jesse Shapiro, Matthew Gentzkow, and Dan Jurafsky. 2019. Analyzing Polarization in Social Media: Method and Application to Tweets on 21 Mass Shootings. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2970–3005.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Sarah Djemili, Julien Longhi, Claudia Marinica, Dimitris Kotzinos, and Georges-Elia Sarfati. 2014. What does Twitter have to say about ideology? In *NLP 4 CMC: Natural Language Processing for Computer-Mediated Communication/Social Media-Pre-conference workshop at Konvens 2014*, volume 1. Universitätsverlag Hildesheim.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are We Modeling the Task or the Annotator? An Investigation of Annotator Bias in Natural Language Understanding Datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166.

Erving Goffman et al. 1978. *The Presentation of Self in Everyday Life*. Harmondsworth London.

Yosh Halberstam and Brian Knight. 2016. Homophily, group size, and the diffusion of political information in social networks: Evidence from Twitter. *Journal of Public Economics*, 143:73–88.

Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political ideology detection using recursive neural networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1113–1122.

Kenneth Joseph, Lisa Friedland, William Hobbs, David Lazer, and Oren Tsur. 2017. ConStance: Modeling Annotation Contexts to Improve Stance Classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1115–1124.

John Kelly, Danyel Fisher, and Marc Smith. 2005. Debate, division, and diversity: Political discourse networks in USENET newsgroups. In *Online Deliberation Conference*, pages 1–35. Stanford University.

Klaus Krippendorff. 2004. Reliability in content analysis: Some common misconceptions and recommendations. *Human communication research*, 30(3):411–433.

Ivar Krumpal. 2013. Determinants of social desirability bias in sensitive surveys: a literature review. *Quality & Quantity*, 47(4):2025–2047.

Cliff AC Lampe, Nicole Ellison, and Charles Steinfield. 2007. A familiar face (book) profile elements as signals in an online social network. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 435–444.

Frank Lin and William W Cohen. 2010. Semi-supervised classification of network data using very few labels. In *2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 192–199. IEEE.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

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

Randal S Olson and Zachary P Neal. 2015. Navigating the massive world of reddit: Using backbone networks to map user interests in social media. *PeerJ Computer Science*, 1:e4.

Arjunil Pathak. 2020. *Extraction and Analysis of Self Identity in Twitter Biographies*. Ph.D. thesis, State University of New York at Buffalo.

Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, Republicans and Starbucks Afficionados: User Classification in Twitter. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 430–438.

Daniel Preotiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. 2017. Beyond binary labels: political ideology prediction of twitter users. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 729–740.

Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2017. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. *arXiv preprint arXiv:1701.08118*.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A Smith. 2017. The Effect of Different Writing Tasks on Linguistic Style: A Case Study of the ROC Story Cloze Task. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 15–25.

Qinlan Shen and Carolyn Rosé. 2019. The Discourse of Online Content Moderation: Investigating Polarized User Responses to Changes in Reddit's Quarantine Policy. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 58–69.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th International Conference on World Wide Web*, pages 613–624.

Oren Tsur, Dan Calacci, and David Lazer. 2015. A frame of mind: Using statistical models for detection of framing and agenda setting campaigns. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1629–1638.

Zeerak Waseem. 2016. Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142.

Omar Zaidan and Chris Callison-Burch. 2011. Crowdsourcing Translation: Professional Quality from Non-Professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1220–1229.

Changtao Zhong, Hau Wen Chang, Dmytro Karamshuk, Dongwon Lee, and Nishanth Sastry. 2017. Wearing many (social) hats: How different are your different social network personae? In *11th International Conference on Web and Social Media, ICWSM 2017*, pages 397–406. AAAI press.

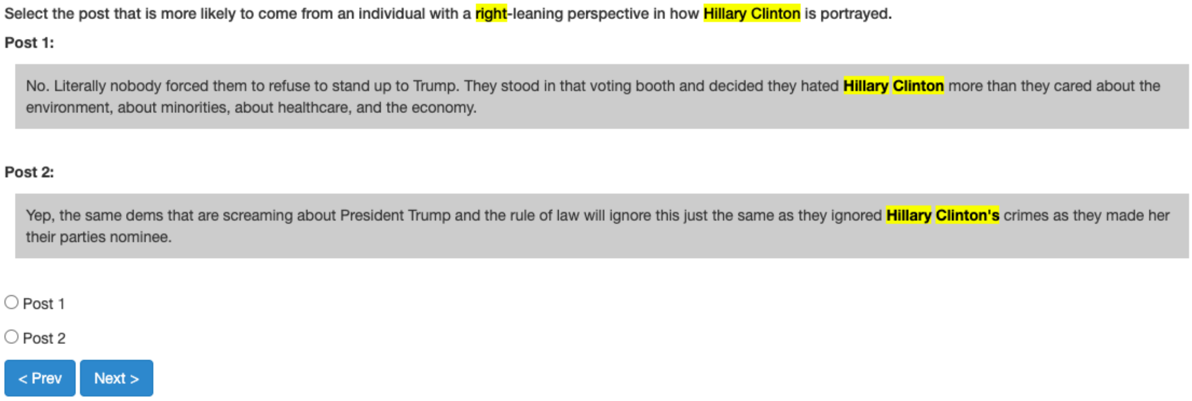
Daniel Xiaodan Zhou, Paul Resnick, and Qiaozhu Mei. 2011. Classifying the political leaning of news articles and users from user votes. In *Proceedings of the International AAAI Conference on Web and Social Media*.

	Selected subreddits
Left	r/LateStageCapitalism, r/SandersForPresident, r/democrats, r/socialism, r/Liberal, r/VoteBlue, r/progressive, r/ChapoTrapHouse, r/neoliberal, r/esist, r/The_Mueller, r/The_Mueller
Right	r/The_Donald, r/Libertarian, r/Republican, r/Conservative, r/JordanPeterson, r/TheNewRight, r/Anarcho_Capitalism, r/conservatives, r/ShitPoliticsSays, r/POLITIC, r/AskTrumpSupporters, r/AskThe_Donald

Table 3: Selected subreddits included in the construction of the dataset and their ideological alignments.

	Selected entities
People	*Donald Trump*, *Joe Biden*, *Bernie Sanders*, *Barack Obama*, *Hillary Clinton*, *Robert Mueller*, *Nancy Pelosi*, *Kamala Harris*, *Alexandria Ocasio-Cortez*, Andrew Yang, Elizabeth Warren, Pete Buttigieg
Ideologies	*conservatives/conservatism*, *liberals/liberalism*, libertarians/libertarianism, socialists/socialism, capitalists/capitalism
Organizations	*Republican Party/Republicans*, *Democratic Party/Democrats*, Congress
Locations	Russia

Table 4: Selected entities included in the construction of the dataset. Italicized entities are also included in the screening set.



Select the post that is more likely to come from an individual with a right-leaning perspective in how Hillary Clinton is portrayed.

Post 1:

No. Literally nobody forced them to refuse to stand up to Trump. They stood in that voting booth and decided they hated **Hillary Clinton** more than they cared about the environment, about minorities, about healthcare, and the economy.

Post 2:

Yep, the same dems that are screaming about President Trump and the rule of law will ignore this just the same as they ignored **Hillary Clinton's** crimes as they made her their parties nominee.

○ Post 1

○ Post 2

Figure 1: Screenshot of a question in the paired ideological annotation task. Annotators are presented with two texts discussing the same highlighted entity in a similar context, one from a left-leaning user and another from a right-leaning user based on a semi-supervised labeling heuristic. Annotators are asked to select which of the two texts is more likely to be authored by someone with the highlighted ideology.

A Survey Questions

A.1 Political ideology

1. Please indicate where you identify on the liberal-conservative spectrum.
    - Liberal
    - Somewhat liberal
    - Moderate
    - Somewhat conservative
    - Conservative
    - I don't know
2. Please indicate how strongly you identify with the following U.S. political parties.
    - Parties
        - Democratic Party
        - Republican Party
        - Libertarian Party
        - Green Party
        - Constitution Party
        - Democratic Socialists of America
        - Reform Party
    - Responses
        - I do not identify with this party
        - Somewhat identify
        - Identify
        - Strongly identify
        - I don't know

A.2 News access

1. On average, how often did you check the news related to the 2020 presidential election in the U.S. in the past year?
    - Never
    - Less than once a month
    - A few times a month
    - Once a week
    - Several times a week
    - Once a day
    - Several times a day

A.3 Reddit familiarity

1. On average, how often have you visited Reddit in the past year?
    - Never
    - Less than once a month
    - A few times a month
    - Once a week
    - Several times a week
    - Once a day
    - Several times a day
2. On average, how often have you posted content to Reddit in the past year?
    - Never
    - Less than once a month
    - A few times a month
    - Once a week
    - Several times a week
    - Once a day
    - Several times a day
3. Please indicate your familiarity with the following subreddits (listed in Table 3).
    - I have never heard of this subreddit
    - I have heard of but never accessed this subreddit
    - I have accessed or posted on this subreddit at least once
    - I sometimes access or post on this subreddit
    - I often access or post on this subreddit

B Detailed Agreement Results

		Agreement		
		F	**S**	**C**
Overall	-	0.388	0.731	0.621
Ideology	left	0.427	0.632	0.674
	right	0.372	0.638	0.583
	center	0.325	0.493	0.609
News	news	0.393	0.627	0.622
	non-news	0.336	0.638	0.588
Reddit	redditor	0.393	0.644	0.622
	–political	0.389	0.641	0.615
	–non-political	0.430	0.582	0.676
	non-redditor	0.359	0.586	0.608

Table 5: Krippendorff's α agreement results for survey categories for the full non-screening annotated set (F), the screening questions (S), and the high-consensus questions subset (C).

C Mismatch Statistics

Group comparison	% mismatch
left/right	28.15
right/center	26.97
non-political/non-redditor	26.47
left/center	24.44
right/non-news	23.74
non-political/right	23.17
news/non-news	22.71
non-political/political	22.61
non-political/center	21.05
non-redditor/political	20.40

Table 6: Comparison pairs with highest percentage of questions where the majority gave different answers.

Entity	% mismatch
libertarians/libertarianism	100.0
Republican Party/Republicans	53.85
Russia	43.75
conservatives/conservatism	42.86
Hillary Clinton	39.13
Joe Biden	38.89
Nancy Pelosi	36.36
liberals/liberalism	31.81
Robert Mueller	28.57
Alexandria Ocasio-Cortez	26.67

Table 7: Entities with highest percentage of questions where the left-leaning and right-leaning annotators gave different answers.

D Human Judgments vs. Labels

		Worker Match
Overall	-	68.53
Ideology	left	70.05
	right	67.30
	center	65.38
News	news	69.24
	non-news	65.80
Reddit	redditor	68.49
	–political	69.03
	–non-political	66.87
	non-redditor	68.65

Table 8: Average percentage of human judgments that match with semi-supervised labels per annotation group.

Language Models as Knowledge Bases:
On Entity Representations, Storage Capacity, and Paraphrased Queries

Benjamin Heinzerling[1, 2] and **Kentaro Inui**[2, 1]
[1]RIKEN AIP & [2]Tohoku University
benjamin.heinzerling@riken.jp | inui@tohoku.ac.jp

Abstract

Pretrained language models have been suggested as a possible alternative or complement to structured knowledge bases. However, this emerging LM-as-KB paradigm has so far only been considered in a very limited setting, which only allows handling 21k entities whose name is found in common LM vocabularies. Furthermore, a major benefit of this paradigm, i.e., querying the KB using natural language paraphrases, is underexplored. Here we formulate two basic requirements for treating LMs as KBs: (i) the ability to store a large number facts involving a large number of entities and (ii) the ability to query stored facts. We explore three entity representations that allow LMs to handle millions of entities and present a detailed case study on paraphrased querying of facts stored in LMs, thereby providing a proof-of-concept that language models can indeed serve as knowledge bases.

1 Introduction

Language models (LMs) appear to memorize world knowledge facts during training. For example, BERT (Devlin et al., 2019) correctly answers the query "Paris is the capital of [MASK]" with "France". This observation prompted Petroni et al. (2019) to ask if LMs can serve as an alternative or complement to structured knowledge bases (KBs), thereby introducing the idea of treating LMs as KBs: During training, the LM encounters world knowledge facts expressed in its training data, some of which are stored in some form in the LM's parameters. After training, some of the stored facts can be recovered from the LM's parameters by means of a suitable natural language query (Fig. 1). A LM with such a "built-in" KB is useful for knowledge-intensive tasks (Petroni et al., 2020) and question answering (Roberts et al., 2020), and could improve natural language interfaces to structured data (Hendrix et al., 1978; Herzig et al., 2020).

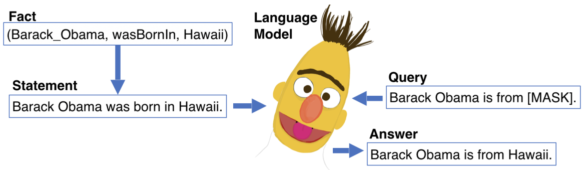


Figure 1: The LM-as-KB paradigm, introduced by Petroni et al. (2019). A LM memorizes factual statements, which can be queried in natural language.

However, this emerging LM-as-KB paradigm is faced with several foundational questions.

First question: KBs contain millions of entities, while LM vocabulary size usually does not exceed 100k entries. **How can millions of entities be represented in LMs?** Petroni et al. (2019) circumvent this problem by only considering 21k entities whose canonical name corresponds to a single token in the LM's vocabulary, e.g., entities like "France" or "Bert", but not "United Kingdom" or "Sesame Street". Hence, this approach cannot handle entities not contained in the vocabulary and a query like "Bert is a character on [MASK]" is not answerable in this simplified setting. To answer this first question, we compare three methods for scaling LM-as-KB to millions of entities:

1. Symbolic representation, i.e., extending the LM vocabulary with entries for all entities;
2. Surface form representation, i.e., each entity is represented by its subword-encoded canonical name, which is stored and queried by extending the LM with a sequence decoder; and
3. Continuous representation, i.e., each entity is represented as an embedding.

We find that, while all three entity representations allow LMs to store millions of world-knowledge facts involving a large number of entities, each representation comes with different trade-offs: Sym-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1772–1791
April 19 - 23, 2021.

bolic representation allows the most accurate storage, but is computationally expensive and requires entity-linked training data. Surface representation is computationally efficient and does not require entity-linked data, but is less accurate, especially for longer entity names. Continuous representation also requires entity-linked data, but is computationally more efficient than symbolic representation.

Second question: **What is the capacity of LMs for storing world knowledge?** Can a LM store, say, all relation triples contained in a KB like Wikidata (Vrandečić and Krötzsch, 2014)? Here we conduct experiments using synthetic data to study the scaling behaviour of current LM architectures. Varying the number of trainable model parameters and recording the number of relation triples memorized at a given accuracy level, we find that, e.g., a Transformer (Vaswani et al., 2017) with 125 million parameters (12 layers of size 768), has the capacity to memorize 1 million Wikidata relation triples with 95 percent accuracy or 5 million relation triples with 79 percent accuracy. Assuming linear scaling, this finding suggests that larger LMs with tens or hundreds of billions of parameters (Raffel et al., 2019; Brown et al., 2020) can be used to store sizable parts, if not all, of a KB like Wikidata.

Third question: **How robustly is world knowledge stored in LMs?** Is the LM able to recall a fact even if the query is slightly different than what was memorized during training? For example, if the LM memorized "Barack Obama was born in Hawaii" during training, can it answer queries like "Barack Obama is from [MASK]" or "Where was Barack Obama born? [MASK]"? Here we conduct experiments to measure how well the LM transfers knowledge from memorized statements to query variants, both in a zero-shot setting in which the model is not exposed to the target query variant during training, and a few shot setting, in which the model is finetuned on a small number of statements containing the target query variant. We observe zero-shot transfer in case of highly similar query variants, and see successful few-shot transfer after finetuning with 5 to 100 instances in case of less similar queries. This ability to handle soft, natural language queries, as opposed to hard, symbolic queries in a language like SQL or SPARQL, is one of the key motivations for using LMs as KBs.

Contributions. We formulate two requirements for treating LMs as KBs: (i) the ability to store a large number of facts involving a large number of entities and (ii) the ability to query stored facts. After providing background on world knowledge in LMs (§2), we make the following contributions:[1]

- A comparison of entity representations for scaling LM-as-KB to millions of entities (§3);
- Empirical lower bounds on LM capacity for storing world knowledge facts (§4); and
- A controlled study of knowledge transfer from stored facts to paraphrased queries (§5).

Terminology. In this work we are interested in storing and retrieving world knowledge facts in and from a LM. **World knowledge** is knowledge pertaining to entities, such as Barack_Obama. A **fact** is a piece of world knowledge that can be expressed with a concise natural language **statement**, such as the English sentence *Barack Obama was Born in Hawaii*, or with a **relation triple**, such as ⟨Barack_Obama, wasBornIn, Hawaii⟩. A relation triple, or relation for short, consists of a **subject** entity (Barack_Obama), a **predicate** (wasBornIn), and an **object** entity (Hawaii). A **knowledge base** is a set of relations. Knowledge bases, such as Wikidata, typically contain thousands of predicates, millions of entities, and billions of relations.

2 World Knowledge in Language Models

Large pretrained LMs have been the driver of recent progress in natural language processing (Peters et al., 2018; Howard and Ruder, 2018; Radford et al., 2019; Devlin et al., 2019). While the trend towards larger LMs is likely to continue (Raffel et al., 2019; Kaplan et al., 2020; Brown et al., 2020), it has limitations: (i) A model trained only on text lacks grounding in perception and experience and hence cannot learn meaning (Bender and Koller, 2020). (ii) Reporting bias leads to certain knowledge rarely or never being expressed in text. For example, a LM will easily learn to associate the phrase "Barack Obama" with the phrase "U.S. President", but might less likely learn that he is a "human being", since the latter fact is rarely stated explicitly in text. In contrast, this type of knowledge is readily available in KBs. (iii) A large number of rare entities (Hoffart et al., 2014; Derczynski et al., 2017; Ilievski et al., 2018) are, by definition, rarely mentioned, making it difficult for

---

[1] Code available at:
`https://github.com/bheinzerling/lm-as-kb`

LMs to acquire knowledge about this long tail of entities from text alone.

These limitations have motivated efforts to explicitly[2] equip LMs with world knowledge. Table 2 (Appx. A) situates these efforts on a spectrum from purely text-based LMs to representations of structured KBs. Models based on text generation (Raffel et al., 2019; Roberts et al., 2020) and retrieval (Guu et al., 2020) have proven most successful in knowledge-intensive tasks. However, we argue that models which reify entities (Logan et al., 2019), i.e., models in which entities are "first-class citizens" that can be directly predicted[3], are a promising research direction, since the direct links into a KB can be seen as a form of grounding. This is one of our main motivations for considering symbolic and continuous entity representations.

3 Entity Representations

How can millions of entities be represented in a LM? To answer our first question, we compare three types of entity representations: symbolic, surface form, and continuous.

Experimental setup. We evaluate entity representations by measuring how well they allow a LM to store and retrieve world knowledge facts. For example, if the LM's training data contains the statement "Bert is a character on Sesame Street", the model should memorize this statement and recall the correct object Sesame_Street when asked with a query like "Bert is a character on [MASK]."

Synthetic data. It is not a priori clear how many facts a text from the LM's training data, say, a Wikipedia article, expresses. Since we want to precisely measure how well a LM can store and retrieve facts, we create synthetic data by generating statements from KB relations and then train the model to memorize these statements. Using Wikidata as KB, we first define two sets of entities: A smaller set consisting of the top 1 million Wikidata entities according to node outdegree, and a larger set consisting of the roughly 6 million Wikidata entities that have an entry in the English Wikipedia.

Next, we manually create templates for the 100 most frequent Wikidata predicates. For example, for the predicate P19 ("place of birth"), we create the template *S was born in O* and generate English statements by filling the *S* and *O* slots with entities from the sets defined above for which this relation holds.[4] To make queries for an object unique given subject and predicate, we arbitrarily select exactly one fact if there are multiple objects and discard the other facts. This process yields 5 million statements involving up to 1 million entities, and 10 million statements involving up to 6 million entities. These statements then serve as training instances, i.e., given the query "Barack Obama was born in [MASK]", the model should predict Hawaii. As our goal is to store facts in a LM, there is no distinction between training and test data.

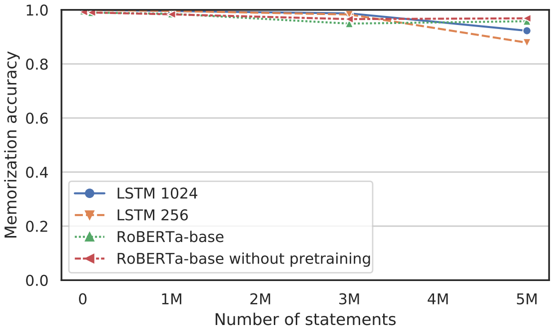


Figure 2: Accuracy of statement memorization with symbolic representation of 1 million entities.

Models and training. We consider two common LM architectures: LSTMs (Hochreiter and Schmidhuber, 1997) and Transformers (Vaswani et al., 2017). For LSTMs, we compare two configurations; a randomly initialized two-layer LSTM with layers size 256 (*LSTM 256*) and one with layer size 1024 (*LSTM 1024*). For Transformers, we compare a pretrained RoBERTa-base (Liu et al., 2019), and RoBERTa without pretraining, i.e., a randomly initialized Transformer of the same size. For consistent tokenization across all four models, we subword-tokenize statements with the RoBERTa tokenizer. To store statements in a LM, we train until the model reaches 99 percent memorization accuracy, i.e., overfits the training data almost perfectly, or stop early if accuracy does not improve for 20 epochs. See Appx. D for training details.

3.1 Symbolic Representation

With symbolic representation, each entity is represented as an entry in the LM's vocabulary. Prediction is done via masked language modeling (Devlin et al., 2019), by encoding the query with the LM, projecting the final hidden state of the [MASK] token onto the vocabulary and then taking a softmax

---

[2] As opposed to the LM acquiring world knowledge implicitly as a side effect of its training objective.

[3] As opposed to generating or retrieving a surface form which may or may not correspond to an entity.

[4] Templates and sample of statements in Appx. B and C.

over the vocabulary. As the results show (Fig. 2), symbolic representation yields very high memorization accuracies with a vocabulary of 1 million entities. Randomly initialized RoBERTa-base without pretraining works best, memorizing 97 percent of 5 million statements correctly.

Unfortunately, the softmax computation becomes prohibitively slow as the vocabulary size increases (Morin and Bengio, 2005), making symbolic representation with a softmax over a vocabulary consisting of the full set of 6 million Wikipedia entities impractical. Imposing a hierarchy is a common approach for dealing with large vocabularies, but did not work well in this case (See Appx. F.1).

3.2 Surface Form Representation

With surface form representation, each entity is represented by its canonical name.[5] Since this name generally consists of more than one token, we cast memorizing statements and querying facts as a sequence-to-sequence task (Sutskever et al., 2014): Given the source sequence “Bert is a character on [MASK]”, the model should generate the target sequence “Sesame Street”.[6] To make models memorize statements, we train until perplexity on the training data reaches 1.0 or does not improve for 20 epochs. For evaluation, we generate the target sequence – i.e., the answer to a given query – via a beam search with beam size 10. We measure perfect-match accuracy of the full entity name, i.e., there is no credit for partial token matches.

The four models under comparison are now treated as sequence-to-sequence encoders and extended with a decoder of the same size: LSTM decoders for LSTM encoders (*LSTM2LSTM*) and randomly initialized Transformers for Transformer encoders (*RoBERTa2Transformer*, *Transformer2Transformer*).

Unlike symbolic representation, surface representation can handle the entire set of 6 million Wikipedia entities. As with symbolic representation, the randomly initialized Transformer (Fig. 3, dash-dotted red line) has the highest capacity, memorizing 10 million statements with 90 percent accuracy. A pretrained encoder (*RoBERTa2Transformer*) appears to have a deleterious effect, yielding lower accuracies than the randomly initialized *Transformer2Transformer*. While the larger *LSTM2LSTM* (layer size 1024) almost matches the performance of the best Transformer model, the smaller one (layer size 256) has insufficient capacity, memorizing less than 50 percent of 5 million statements.

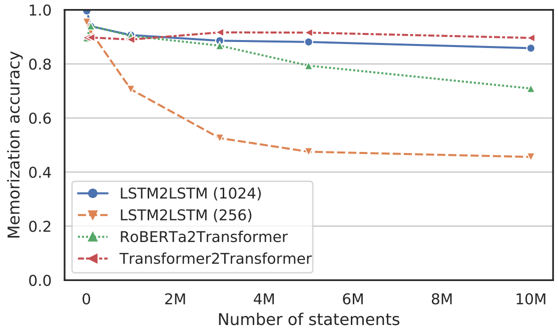


Figure 3: Accuracy of statement memorization with object entities represented by surface forms.

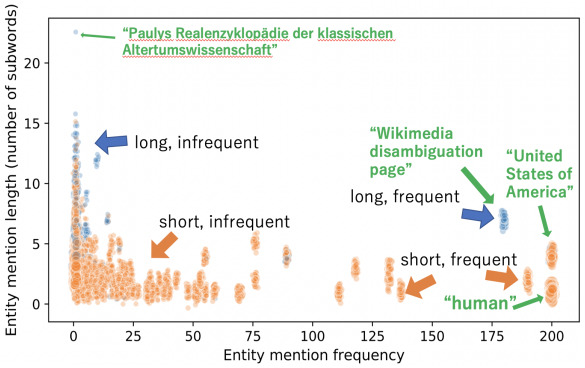


Figure 4: Error analysis of statements memorized via surface form representation. Correctly memorized statements orange, wrong ones blue. Selected clusters are annotated with the statement’s object entity (green). Frequencies clipped to 200, jitter applied for clarity.

Analysis of the Transformer2Transformer model (Fig. 4) reveals, perhaps unsurprisingly, that statements involving infrequent, long entity mentions are difficult to memorize.[7] For example, the model fails to memorize most entity mentions that occur only in one to ten statements and have a length of 12 or more subwords (blue cluster, upper left).

3.3 Continuous Representation

With continuous representation, an entity $e_i, i \in [1, N_{entities}]$ is represented by a d-dimensional embedding $\mathbf{y}_i \in \mathbb{R}^d$. After encoding a query with the LM, prediction is performed by projecting the final hidden state corresponding to the [MASK]

---

[5] We use English Wikidata labels as canonical names.

[6] The [MASK] token is included since the target entity does not always occur at the end of a statement.

[7] We speculate that this drawback can be mitigated by shortening canonical names while ensuring a one-to-one mapping to entities, but leave this to future work.

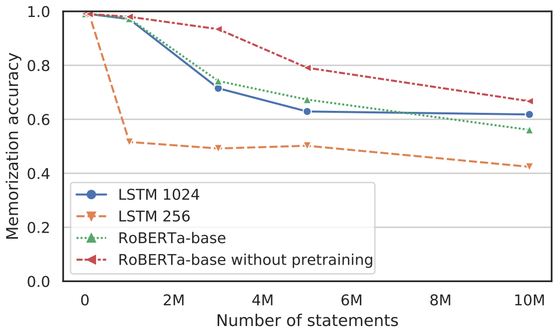


Figure 5: Accuracy of statement memorization with continuous entity representation.

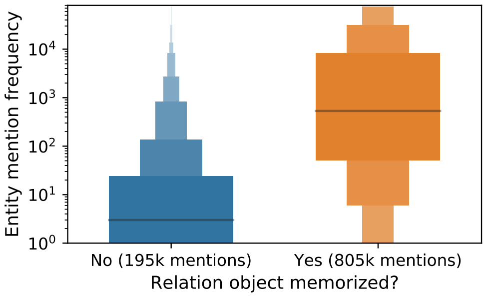


Figure 6: Error analysis of a sample of 1 million statements memorized by a randomly initialized Transformer with continuous representation.

token onto $\mathbb{R}^d$, obtaining the predicted embedding $\hat{\mathbf{y}} \in \mathbb{R}^d$. We use fixed, pretrained entity embeddings and train with cosine loss $L = 1 - \cos(\hat{\mathbf{y}}, \mathbf{y_i})$. At test time, the model prediction $\hat{\mathbf{y}}$ is mapped to the closest pretrained entity embedding $\mathbf{y}_i$ via nearest-neighbor search (Johnson et al., 2017).

Continuous prediction with fixed, pretrained embeddings. When training randomly initialized embeddings with a similarity objective, a degenerate solution is to make all embeddings the same, e.g., all-zero vectors. To prevent this, it is common practice to use negative samples (Bordes et al., 2013). When using fixed, pretrained embeddings as supervision signal, negative sampling is not necessary, since the target embeddings are not updated and therefore cannot become degenerate.

Wikidata embeddings. We train embeddings for 6 million Wikidata entities using feature-specific autoencoders to encode entity features such as names, aliases, description, entity types, and numeric attributes, following prior work on multi-modal KB embeddings (Pezeshkpour et al., 2018) and KB embeddings with autoencoders (Takahashi et al., 2018). Embedding training is detailed in Appx. E.

Results. Fig. 5 shows memorization accuracies achieved with continuous representation. Like surface representation, continuous representation scales to 6 million entities, and we see the same relative order of models, but with overall lower accuracies. RoBERTa without pretraining has the highest capacity for storing world knowledge statements, memorizing 67 percent of 10 million statements, while the small LSTM 256 model has the lowest capacity, memorizing 42 percent. Although far from fully understood, sequence-to-sequence architectures are relatively mature, with highly-optimized toolkits and hyperparameter settings publicly available (Ott et al., 2019). In contrast, prediction of continuous representations is still in an early stage of research (Kumar and Tsvetkov, 2019). We therefore see these results as lower bounds for LM capacity with continuous representations.

By design, memorization with continuous representations does not rely on entity names, and hence, in contrast to surface form representation, does not lead to difficulties in handling entities with long names. However, as with surface form representation, infrequent entities are more difficult to memorize than frequent ones. Most of the memorization errors (Fig. 6, blue, left) involve infrequent entities with a median frequency of 3, while most of the correctly memorized statements (orange, right) involve entities that occur more than 100 times.

4 LM Capacity for Storing Facts

We now turn to the second question, how model capacity scales with model size (Fig. 7, top). With a 12-layer Transformer of layer size 96 or 192 (top subfigure, solid red and dashed green lines), memorization accuracy quickly drops as the number of facts to memorize increases. Larger models can memorize more facts, but accuracy drops rather quickly, e.g., to 65 percent of 3 million facts memorized with a layer size of 384 (dotted orange line).

Assuming a desired memorization accuracy of 80 percent, we record the maximum number of facts a model of a given size can memorize at this level (Fig. 7, bottom). For the model sizes considered here, storage capacity appears to scale linearly, with a model of layer size 384 (55M parameters) storing one million facts and a model of layer size 960 (160M parameters) storing 7 million facts.

Apart from the number of facts to store, we hypothesize that successful storage depends on two more factors: the number of entities and the en-

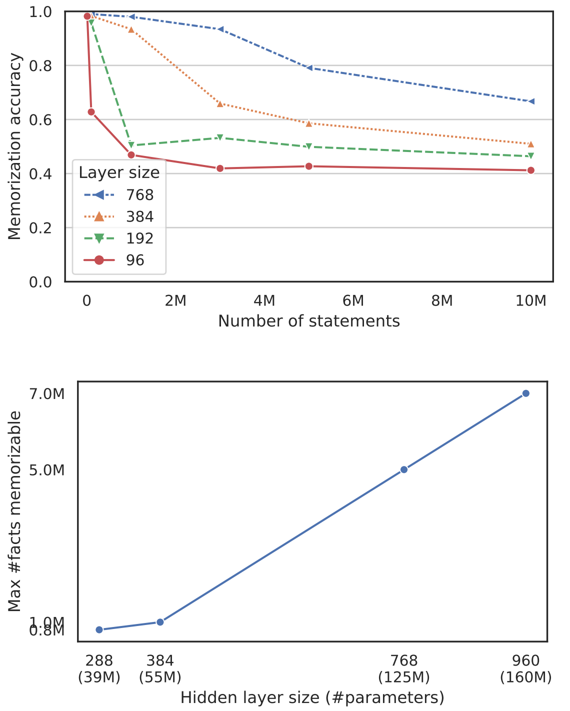


Figure 7: Scaling of storage capacity with model size. Memorization accuracy decreases as the number of facts grows (top). The maximum number of facts a model of a given layer size (parameter count) can memorize with an accuracy of 80 percent increases linearly (bottom). All models are 12-layer Transformer with continuous representation of 6 million entities.

Representation	Accuracy 1M	 6M
Symbolic	0.97	n/a
Surface	0.92	0.90
Continuous	0.85	0.79

Table 1: The number of entities (1M or 6M) impacts memorization accuracy. The model is a 12-layer Transformer, layer size 768, memorizing 1 million facts.

tropy of their distribution. As expected, a large number of entities makes memorization more difficult (Table 1). The number of entities has a small effect with surface representation (2 percent drop), but with continuous representation accuracy drops from 85 percent to 79 percent when the number of entities increases from 1 to 6 million. We also observe an impact of the entity distribution (Appx. G), but leave detailed analysis to future work.

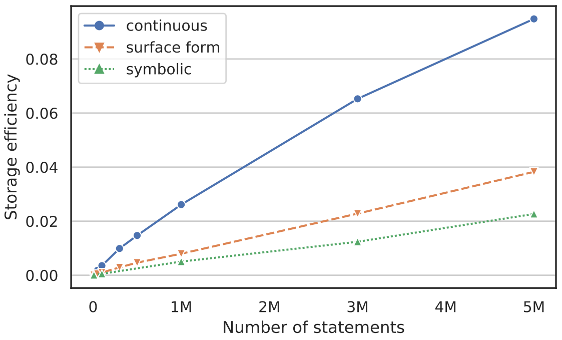


Figure 8: Storage efficiency with symbolic, surface form, and continuous representation of entities. In the setting considered in this work, continuous representation is most efficient.

4.1 Storage Efficiency

Our comparison of different entity representations (§3) does not control for the number of trainable model parameters. That is, we selected common architectures, such as a Transformer with 12 layers of size 768, but made no effort to ensure that, e.g., the number of trainable parameters introduced by the softmax layer in a model with symbolic representation matches the number of trainable parameters introduced by the addition of a sequence-to-sequence decoder component in a model with surface form representation. In order to more fairly compare entity representations across models with differing numbers of trainable parameters, we formulate the *storage efficiency* of a model designed to memorize statements:

$$\text{Storage efficiency} = \frac{\#statements \times accuracy}{\#parameters}$$

This measure expresses the intuition that a model is efficient if it requires few parameters to memorize a large number of statements with high accuracy. When quantifying efficiency with this measure, we find that continuous representation is the most efficient (Figure 8) and hence use this form of entity representation in the remainder of this work.

5 Querying Stored Facts

So far, we saw that it is possible to store millions of facts in a LM, by finetuning the model to predict the masked object of statements like *Barack Obama was born in [MASK]*. However, given the large number of model parameters and training effort, mere storage is not a compelling achievement: The underlying relations, here ⟨Barack_Obama, wasBornIn, Hawaii⟩, can be stored more compactly and with perfect accuracy in a structured KB.

One of the potential benefits of the LM-as-KB paradigm is the LM's ability to handle paraphrases. If the LM's representation of the statement above is sufficiently similar to its representation of queries like *Barack Obama is from [MASK]* or *Where is Barack Obama from? [MASK]*, this similarity could allow transfer from the memorized statement to these unseen queries. Is this soft querying of facts stored in a LM possible? We now conduct a controlled experiment to answer this question, expecting one of the following three outcomes:

1. **Rote memorization**. The model memorizes statements with little or no abstraction, so that even small, meaning-preserving changes to the query prevent the model from recalling the correct object.
2. **Generic association**. The model memorizes pairs of subject and object entities but disregards the predicate. For example, a model might predict Hawaii whenever the query contains the phrase *Barack Obama*, regardless of context. This pathological behaviour could be especially prevalent if the distribution of object entities co-occurring with a given subject is dominated by one object.
3. **Fact memorization**. The model memorizes facts expressed in statements by forming abstractions corresponding to entities and predicates. This allows retrieving a fact with a variety of queries.

Sections 3 and 4 already established that a model of sufficient size can perform rote memorization of millions of statements. We now design an experiment to test whether LMs are capable of fact memorization while taking care to distinguish this capability from generic association. Concretely, our goal is to test if a LM that has memorized a statement like *Barack Obama was born in Hawaii* can use this knowledge to answer a query like *Barack Obama is from [MASK]*. Conveniently, wasBornIn relations are among the most frequent in Wikidata and hold for a diverse set of subject and object entities. This diversity of entities makes this predicate a good candidate for our case study, since statements involving a predicate with a less diverse set of subject or object entities are easier to memorize.[8]

Statements and controls. We sample 100k statements generated by the template "S was born in O". To allow distinguishing if a model that memorizes these 100k facts does so by generic assocation or by fact memorization, we introduce *control* facts. Given a fact ⟨S, P, O⟩, its control ⟨S, P', O'⟩ involves the same subject S, but a distinct predicate P' and object O'. For example, a control for the fact ⟨Albert_Einstein, wasBornIn, Ulm⟩ is the fact ⟨Albert_Einstein, diedIn, Princeton⟩. We add 100k control statements generated from the template "S died in O"' and train RoBERTa-base to memorize all 200k statements with 98 percent accuracy. The combination of statements and controls counters generic association: To correctly answer the query "Albert Einstein died in [MASK]", the model needs to take into account the predicate, since two distinct objects are associated with Albert_Einstein.

Query variants. Next, we collect query variants, such as "S is from O" (row labels in Fig. 9, top). Expecting good transfer for variants that are similar to the original statement, we include variants with small changes, such as varying punctuation. As more diverse variants, we select frequent relation patterns, such as "S (b. 1970, O)", from the GoogleRE Corpus (Google, 2013), as well as a query in question form and queries with irrelevant or misleading distractors such as "S was born in O, but died somewhere else". For each variant, we generate 100k queries by filling the S and O slots with the same entity pairs as the original statements. To balance statements and controls, we create control templates (row labels in Fig. 9, bottom) and generate a matching number of control statements.

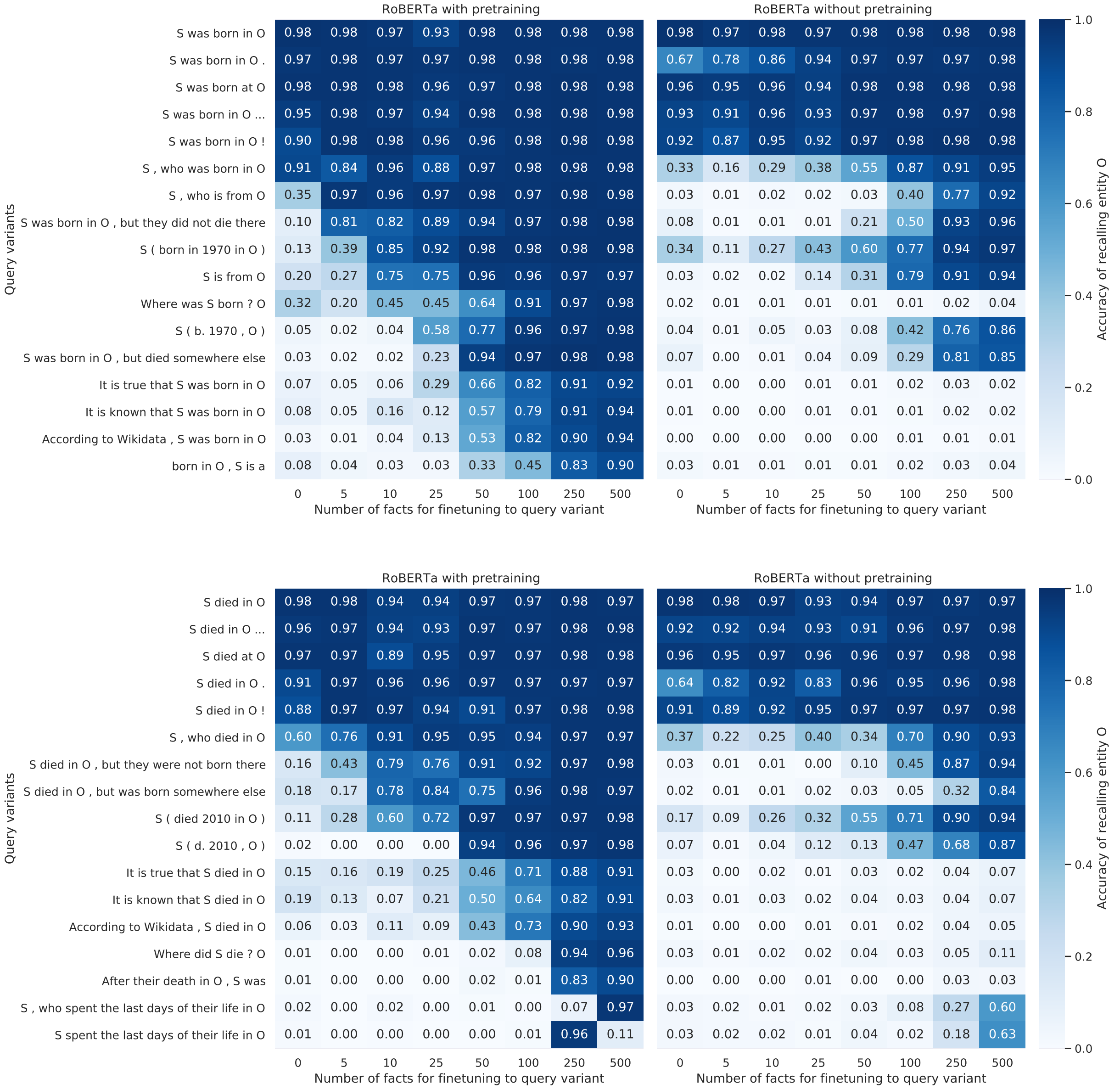
Transfer results. We evaluate knowledge transfer from memorized statements to query variants using RoBERTa-base (Fig. 9, top, left), measuring accuracy over the 100k statements generated with a target query variant template. To measure the effect of pretraining on transfer ability, we compare to RoBERTa-base without pretraining (Fig. 9, top, right). We consider zero-shot transfer without any finetuning towards the target query variant, and a finetuning setting, in which the LM is first trained to memorize all 100k original statements and then finetuned until it memorizes a small number of statements in the target query format.[9]

In the zero-shot setting (leftmost column), even small changes to the query lead to a drop in fact recall: Adding an ellipsis (4th row) causes the model to answer 95% of queries correctly, a 3% drop from 98% memorization of the original statements (first row). Adding an exclamation mark (5th row) results in a 8% drop. For other paraphrases, e.g., *S, who is from O* (7th row) and *S is from O*, zero-shot transfer works only in 35% and 20% of cases, and

---

[8] For example, with the predicate isA and relations like ⟨Barack_Obama, isA, human⟩ the model would do well by always predicting human if the subject mention matches a frequent person name pattern such as "two capitalized words".

[9] Our experiments, which test whether a LM can transfer a memorized fact to given target paraphrases, are converse to the probing setup by Jiang et al. (2020), which aims to find the best paraphrase for querying a given fact from a LM.



RoBERTa with pretraining

Query variants	0	5	10	25	50	100	250	500
S was born in O	0.98	0.98	0.97	0.93	0.98	0.98	0.98	0.98
S was born in O .	0.97	0.98	0.97	0.97	0.98	0.98	0.98	0.98
S was born at O	0.98	0.98	0.98	0.96	0.97	0.98	0.98	0.98
S was born in O ...	0.95	0.98	0.97	0.94	0.98	0.98	0.98	0.98
S was born in O !	0.90	0.98	0.98	0.96	0.96	0.98	0.98	0.98
S , who was born in O	0.91	0.84	0.96	0.88	0.97	0.98	0.98	0.98
S , who is from O	0.35	0.97	0.96	0.97	0.98	0.97	0.98	0.98
S was born in O , but they did not die there	0.10	0.81	0.82	0.89	0.94	0.97	0.98	0.98
S (born in 1970 in O)	0.13	0.39	0.85	0.92	0.98	0.98	0.98	0.98
S is from O	0.20	0.27	0.75	0.75	0.96	0.96	0.97	0.97
Where was S born ? O	0.32	0.20	0.45	0.45	0.64	0.91	0.97	0.98
S (b. 1970 , O)	0.05	0.02	0.04	0.58	0.77	0.96	0.97	0.98
S was born in O , but died somewhere else	0.03	0.02	0.02	0.23	0.94	0.97	0.98	0.98
It is true that S was born in O	0.07	0.05	0.06	0.29	0.66	0.82	0.91	0.92
It is known that S was born in O	0.08	0.05	0.16	0.12	0.57	0.79	0.91	0.94
According to Wikidata , S was born in O	0.03	0.01	0.04	0.13	0.53	0.82	0.90	0.94
born in O , S is a	0.08	0.04	0.03	0.03	0.33	0.45	0.83	0.90

RoBERTa without pretraining

Query variants	0	5	10	25	50	100	250	500
S was born in O	0.98	0.97	0.98	0.97	0.98	0.98	0.98	0.98
S was born in O .	0.67	0.78	0.86	0.94	0.97	0.97	0.97	0.98
S was born at O	0.96	0.95	0.96	0.94	0.98	0.98	0.98	0.98
S was born in O ...	0.93	0.91	0.96	0.93	0.97	0.98	0.97	0.98
S was born in O !	0.92	0.87	0.95	0.92	0.97	0.98	0.98	0.98
S , who was born in O	0.33	0.16	0.29	0.38	0.55	0.87	0.91	0.95
S , who is from O	0.03	0.01	0.02	0.02	0.03	0.40	0.77	0.92
S was born in O , but they did not die there	0.08	0.01	0.01	0.01	0.21	0.50	0.93	0.96
S (born in 1970 in O)	0.34	0.11	0.27	0.43	0.60	0.77	0.94	0.97
S is from O	0.03	0.02	0.02	0.14	0.31	0.79	0.91	0.94
Where was S born ? O	0.02	0.01	0.01	0.01	0.01	0.01	0.02	0.04
S (b. 1970 , O)	0.04	0.01	0.05	0.03	0.08	0.42	0.76	0.86
S was born in O , but died somewhere else	0.07	0.00	0.01	0.04	0.09	0.29	0.81	0.85
It is true that S was born in O	0.01	0.00	0.00	0.01	0.01	0.02	0.03	0.02
It is known that S was born in O	0.01	0.00	0.00	0.01	0.01	0.01	0.02	0.02
According to Wikidata , S was born in O	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01
born in O , S is a	0.03	0.01	0.01	0.01	0.01	0.02	0.03	0.04

RoBERTa with pretraining

Query variants	0	5	10	25	50	100	250	500
S died in O	0.98	0.98	0.94	0.94	0.97	0.97	0.98	0.97
S died in O ...	0.96	0.97	0.94	0.93	0.97	0.97	0.98	0.98
S died at O	0.97	0.97	0.89	0.95	0.97	0.97	0.98	0.98
S died in O .	0.91	0.97	0.96	0.96	0.97	0.97	0.97	0.97
S died in O !	0.88	0.97	0.97	0.94	0.91	0.97	0.98	0.98
S , who died in O	0.60	0.76	0.91	0.95	0.95	0.94	0.97	0.97
S died in O , but they were not born there	0.16	0.43	0.79	0.76	0.91	0.92	0.97	0.98
S died in O , but was born somewhere else	0.18	0.17	0.78	0.84	0.75	0.96	0.98	0.97
S (died 2010 in O)	0.11	0.28	0.60	0.72	0.97	0.97	0.97	0.98
S (d. 2010 , O)	0.02	0.00	0.00	0.00	0.94	0.96	0.97	0.98
It is true that S died in O	0.15	0.16	0.19	0.25	0.46	0.71	0.88	0.91
It is known that S died in O	0.19	0.13	0.07	0.21	0.50	0.64	0.82	0.91
According to Wikidata , S died in O	0.06	0.03	0.11	0.09	0.43	0.73	0.90	0.93
Where did S die ? O	0.01	0.00	0.00	0.01	0.02	0.08	0.94	0.96
After their death in O , S was	0.01	0.00	0.00	0.00	0.02	0.01	0.83	0.90
S , who spent the last days of their life in O	0.02	0.00	0.02	0.00	0.01	0.00	0.07	0.97
S spent the last days of their life in O	0.01	0.00	0.00	0.00	0.00	0.01	0.96	0.11

RoBERTa without pretraining

Query variants	0	5	10	25	50	100	250	500
S died in O	0.98	0.98	0.97	0.93	0.94	0.97	0.97	0.97
S died in O ...	0.92	0.92	0.94	0.93	0.91	0.96	0.97	0.98
S died at O	0.96	0.95	0.97	0.96	0.96	0.97	0.98	0.98
S died in O .	0.64	0.82	0.92	0.83	0.96	0.95	0.96	0.98
S died in O !	0.91	0.89	0.92	0.95	0.97	0.97	0.97	0.98
S , who died in O	0.37	0.22	0.25	0.40	0.34	0.70	0.90	0.93
S died in O , but they were not born there	0.03	0.01	0.01	0.00	0.10	0.45	0.87	0.94
S died in O , but was born somewhere else	0.02	0.01	0.01	0.02	0.03	0.05	0.32	0.84
S (died 2010 in O)	0.17	0.09	0.26	0.32	0.55	0.71	0.90	0.94
S (d. 2010 , O)	0.07	0.01	0.04	0.12	0.13	0.47	0.68	0.87
It is true that S died in O	0.03	0.00	0.02	0.01	0.03	0.02	0.04	0.07
It is known that S died in O	0.03	0.01	0.03	0.02	0.04	0.03	0.04	0.07
According to Wikidata , S died in O	0.01	0.00	0.00	0.01	0.01	0.02	0.04	0.05
Where did S die ? O	0.03	0.01	0.02	0.02	0.04	0.03	0.05	0.11
After their death in O , S was	0.00	0.00	0.00	0.01	0.00	0.00	0.03	0.03
S , who spent the last days of their life in O	0.03	0.02	0.01	0.02	0.03	0.08	0.27	0.60
S spent the last days of their life in O	0.03	0.02	0.02	0.01	0.04	0.02	0.18	0.63

Figure 9: Transfer from memorized statements (top: "S was born in O", bottom: "S died in O") to query variants.

the question format (11th row) allows zero-shot transfer with 32% accuracy. For the remaining paraphrases, e.g., those with parentheticals or the distractor *died*, zero-shot transfer is poor, with accuracies ranging from 3% to 13%.

A clear trend is visible: transfer works best for similar statements and worst for dissimilar ones. To quantify this trend, we compute a representation of a statement template by averaging over its 100k mean-pooled, LM-encoded statements, and then measure the Euclidean distance between the original template representation and the representation of a query variant template. Correlating Euclidean distance and accuracy of zero-shot transfer obtains a Pearson coefficient of -0.68, indicating a strong negative correlation. That is, transfer tends to work well for paraphrased queries the LM deems similar to the originally memorized statement, but fails if the LM's representation of a query is too dissimilar to its representation of the original statement. This trend is also reflected in the finetuning setting, with less similar variants requiring up to 500 instances until the model achieves 90 percent accuracy (last row), while transfer to more similar variants works well after finetuning on 5 to 50 target instances.

When using RoBERTa without pretraining to memorize statements, knowledge transfer to query variants is much worse. While transfer still works for the most similar variants (right, top rows), less similar variants require more finetuning compared to pretrained RoBERTa (right, middle rows). Transfer does not work for the least similar variants, with accuracies as low as 1 to 4 percent even after finetuning with 500 instances (right, bottom rows). Similar results for control statements are shown in Fig. 9 (bottom). We take these results as evidence that pretraining enables LMs to handle paraphrased queries and that LMs can memorize facts beyond mere rote memorization and generic association.

6 Limitations and Conclusions

Limitations. This work is not without limitations. We only use one KB in our experiments. Arguably, as the largest publicly available source of world knowledge, Wikidata is the most promising resource for equipping LMs with such knowledge, but attempts to store a KB with different structure might result in different outcomes, since some types of graphs are easier to memorize for a LM than others (See Appx. G).

While we use language like "*train* a LM to memorize statements" for simplicity throughout this work, what we do in case of pretrained LMs is more akin to adaptive pretraining (Gururangan et al., 2020). It is possible that integrating entity supervision directly into LM pretraining (Févry et al., 2020) allows more efficient fact storage.

Our analysis was focused on entity representations and ignored the question of how to represent relation predicates or entire relation triples. Here, relation learning (Baldini Soares et al., 2019) and LM pretraining on fact-aligned corpora (Elsahar et al., 2018) are avenues for future work.

Finally, we formulated the LM-as-KB paradigm in terms of storing and retrieving relation triples. While structured KBs such as Wikidata consist of such triples and hence our experiments showing storage and retrieval of triples LMs are sufficient as a proof-of-concept in principle, structured KBs allow more complex queries than the ones considered here, such as 1-to-n relations, multihop inference, queries involving numerical ranges, or facts qualified by time and location (Hoffart et al., 2013).

Conclusions. We gave a positive answer to Petroni et al. (2019)'s question if language models can serve as knowledge bases. Arguing that treating LMs as KBs requires representing a large number of entities, storing a large number of facts, and the ability to query a fact with a variety of queries, we showed that current LM architectures fulfill these requirements when extended with a component for representing entities. In addition to the ability to handle paraphrased queries, we envision further benefits from the LM-as-KB paradigm. For example, the fact-memorization and paraphrase-finetuning setting introduced in Section 5 allows precise control over which facts a LM learns.

7 Acknowledgments

We thank the anonymous reviewers for helpful feedback. This work was supported by a Google Focused Research Award.

References

Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A neural knowledge language model. *CoRR*, abs/1608.00318.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2787–2795, Red Hook, NY, USA. Curran Associates Inc.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda

Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Sam Coppens, Miel Vander Sande, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. 2013. Reasoning over SPARQL. In *LDOW*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818.

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

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. *CoRR*, abs/2004.07202.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing semantic similarity for entity linking with convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1256–1261, San Diego, California. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Google. 2013. Google relation extraction corpus. `https://ai.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html`. Accessed: 2020-10-07.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Gary G Hendrix, Earl D Sacerdoti, Daniel Sagalowicz, and Jonathan Slocum. 1978. Developing a natural language interface to complex data. *ACM Transactions on Database Systems (TODS)*, 3(2):105–147.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In *Proceedings of the 23rd international conference on World wide web, WWW 2014, Seoul, South Korea*, pages 385–396.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard

Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Filip Ilievski, Piek Vossen, and Stefan Schlobach. 2018. Systematic study of long tail phenomena in entity linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 664–674, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Sachin Kumar and Yulia Tsvetkov. 2019. Von Mises-Fisher loss for training sequence to sequence models with continuous outputs. In *Proc. of ICLR*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife Hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971, Florence, Italy. Association for Computational Linguistics.

Federico López, Benjamin Heinzerling, and Michael Strube. 2019. Fine-grained entity typing in hyperbolic space. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 169–180, Florence, Italy. Association for Computational Linguistics.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, pages 809–816. Omnipress.

Yusuke Oda, Philip Arthur, Graham Neubig, Koichiro Yoshino, and Satoshi Nakamura. 2017. Neural machine translation via binary code prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 850–860, Vancouver, Canada. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2020. KILT: a benchmark for knowledge intensive language tasks. *CoRR*, abs/2009.02252.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Pouya Pezeshkpour, Liyan Chen, and Sameer Singh. 2018. Embedding multimodal relational data for knowledge base completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3208–3218, Brussels, Belgium. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. E-BERT: Efficient-yet-effective entity embeddings for BERT. *CoRR*, abs/1911.03681.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5406–5413. AAAI Press.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *CoRR*, abs/2002.08910.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27(3):379–423.

Daniil Sorokin and Iryna Gurevych. 2018. Modeling semantics with gated graph neural networks for knowledge base question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3306–3317. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Ryo Takahashi, Ran Tian, and Kentaro Inui. 2018. Interpretable and compositional relation learning by joint training with an autoencoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2148–2159, Melbourne, Australia. Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 2071–2080. JMLR.org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2019. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. *CoRR*, abs/1912.09637.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259. Association for Computational Linguistics.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages

1850–1859, Copenhagen, Denmark. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

A Overview: world knowledge in natural language processing

Paradigm / Task	Input	Output	Models and objectives
Language modeling	Text	Text	Next word prediction (Shannon, 1948; Elman, 1990; Bengio et al., 2003), masked token prediction (Devlin et al., 2019)
LM-as-KB?	Text	Text / single-token entity name	Closed-book QA (LAMA probe, Petroni et al., 2019)
Sequence-to-sequence	Text	Text	Text-to-text transformer (T5, Raffel et al., 2019), closed-book QA (Roberts et al., 2020)
Retrieval	Text	Text, answer span	Answer-span selection (Chen et al., 2017), retrieval-augmented LM (Guu et al., 2020), open-book QA
Entity replacement	Text, entity mention spans	Text	Detecting replaced entity mentions (Xiong et al., 2019)
Entity linking (EL)	Text, entity mention spans	Target entity	AIDA (Hoffart et al., 2011), neural EL (Francis-Landau et al., 2016; Kolitsas et al., 2018)
Entity embeddings	Text, entity mention spans	Entity embeddings	Joint embedding of entities and text (Yamada et al., 2016)
LM with entity embeddings	Text, linked entity mentions, entity embeddings	Text	ERNIE (Zhang et al., 2019), E-BERT (Poerner et al., 2019)
LM with integrated EL	Text, entity embeddings	Text	KnowBert (Peters et al., 2019)
LM-as-KB (this work)	Natural language query	Target entity	Fact memorization, paraphrased queries, closed-book QA
Knowledge-aware LM	Text, knowledge (sub)graph	Target entity, text	Neural Knowledge LM (Ahn et al., 2016), Reference-aware LM (Yang et al., 2017), Knowledge graph LM (Logan et al., 2019)
Semantic parsing	natural language query	meaning representation, target entity	SEMPRE (Berant et al., 2013), GNNs for KBQA (Sorokin and Gurevych, 2018)
Universal Schema	relation triples, text patterns	entity tuple and relation embeddings	Matrix factorization (Riedel et al., 2013)
Knowledge graph embeddings	relation triples	node and edge embeddings	Link prediction; RESCAL (Nickel et al., 2011), TransE (Bordes et al., 2013), ComplexE (Trouillon et al., 2016), ConvE (Dettmers et al., 2018)
Graph neural networks	nodes, node features, edges	node embeddings	DeepWalk (Perozzi et al., 2014), graph neural networks (Kipf and Welling, 2017)
Knowledge graphs	nodes, edges	nodes, edges	Storage and retrieval, SQL/SPARQL queries, symbolic reasoning (Coppens et al., 2013)

Table 2: Approaches for using world knowledge in natural language processing, ranging from unstructured, purely text-based approaches (top), over approaches that mix text and structured KBs to varying degrees (middle), to approaches operating on structured KBs (bottom).

B Templates for generating English statements from Wikidata relations

ID	Template
P31	S is an instance of O
P106	S has the occupation O
P17	S belongs to the country O
P131	S is located in the administrative territorial entity O
P27	S is citizen of O
P47	S shares a border with O
P19	S was born in O
P161	S has the cast member O
P421	S is located in time zone O
P166	S received the award O
P54	S is a member of the sports team O
P20	S died in O
P136	S has the genre O
P69	S was educated at O
P1412	S is a language spoken, written or signed in O
P190	S is a twinned administrative body of O
P641	S participates in the sport O
P150	S contains the administrative territorial entity O
P463	S is a member of O
P735	S has the given name O
P1343	S is described by source O
P361	S is a part of O
P159	the headquarters of S are located in O
P1344	S is participant of O
P495	S has the country of origin O
P39	S held the position of O
P910	S has the main category O
P105	S has the taxon rank O
P527	S has the part O
P108	S is employed by O
P279	S is a subclass of O
P171	S has the parent taxon O
P140	S has the religion O
P407	S is in the O language
P1303	S plays the instrument O
P1411	S has been nominated for O
P102	S is a member of political party O
P3373	S is a sibling of O
P1376	S is the capital of O
P509	S died because of O
P937	S works in O
P264	S was produced by the record label O
P119	S is buried in O
P138	S is named after O
P530	S has diplomatic relations with O
P40	S is a child of O
P155	S follows O
P276	S is located in O
P156	S is followed by O
P36	S has the capital O
P1196	S has the manner of death O
P127	S is owned by O
P101	S works in the field O
P607	S participated in the conflict O
P364	S is a film or TV show with the original language O
P6379	S has works in the collection O
P1346	S is a winner of the O
P22	S is the father of O
P137	S is operated by O
P413	S plays the position O
P26	S is spouse of O
P1830	S is owner of O
P1454	S has the legal form O
P206	S is located in or next to body of water O
P710	S is a participant of O
P1441	S is present in the work O
P1532	S represents O when playing sport O
P86	S was composed by O
P840	S is set in the location O
P172	S belongs to the ethnic group O
P175	S is performed by O
P57	S is directed by O
P1889	S is different from O
P162	S is produced by O
P118	S belongs to the league O
P58	S is screenwritten by O
P551	S has the residence O
P103	S has the native language O
P2789	S connects with O
P750	S has the distributor O
P725	S is voiced by O
P272	S is produced by the company O
P112	S was founded by O
P452	S belongs to the industrial sector O
P81	S is connected to line O
P97	S has noble title O
P740	S formed in the location O
P360	S is a list of O
P793	S is associated with the significant event O
P915	S was filmed at O
P410	S has military rank O
P1001	S applies to the jurisdiction of O
P30	S is located on the continent O
P749	S has parent organization O
P1435	S has heritage designation O
P53	S belongs to the family of O
P400	S was developed for the platform O
P921	S has the main subject O
P37	S has the official language O
P734	S has the family name O

Table 3: Templates used to generate English statements from Wikidata relations.

C Random sample of English statements generated from Wikidata relations

- The Underfall Yard is followed by English Electric Part One
- Gazi Beg is a child of Farrukh Yassar
- 2011 European Rowing Championships is followed by 2012 European Rowing Championships
- 2009 Yemeni tourist attacks is located in Shibam
- George Best – A Tribute is performed by Peter Corry
- Gamecock Media Group is owned by SouthPeak Games
- 2017–18 Sheffield Wednesday F.C. season is followed by 2018–19 Sheffield Wednesday F.C. season
- Nennslingen is located in or next to body of water Anlauter
- 2013–14 Xavier Musketeers men’s basketball team is followed by 2014–15 Xavier Musketeers men’s basketball team
- Shock to the System is a part of Cyberpunk
- 1918–19 Ohio Bobcats men’s basketball team follows 1917–18 Ohio Bobcats men’s basketball team
- Ramya Krishnan has the spouse Krishna Vamsi
- The Cloud Minders follows The Way to Eden
- Curve is followed by Somethingness
- Austin Road is named after John Gardiner Austin
- Dione juno has the parent taxon Dione
- Spirit Bound Flesh is followed by The Wake
- Sidnei da Silva has the given name Sidnei
- In Memoriam is performed by Living Sacrifice
- Tracks and Traces is followed by Live 1974
- Grumman Gulfstream I is operated by Phoenix Air
- Timeline of Quebec history has the part Timeline of Quebec history (1982–present)
- Edwin C. Johnson held the position of Lieutenant Governor of Colorado
- Here Comes the Summer follows Jimmy Jimmy
- In Custody is screenwritten by Anita Desai
- Bertie Charles Forbes is the father of Malcolm Forbes
- The Mambo Kings has the cast member Helena Carroll
- Carnival of Souls has the cast member Art Ellison
- 1995–96 Philadelphia Flyers season is followed by 1996–97 Philadelphia Flyers season
- John Harley is the father of Edward Harley, 5th Earl of Oxford and Earl Mortimer
- Jane Fellowes, Baroness Fellowes has the spouse Robert Fellowes, Baron Fellowes
- Francis of Assisi is buried in Basilica of San Francesco d’Assisi
- 1990 Maharashtra Legislative Assembly election follows 1985 Maharashtra Legislative Assembly election
- Makabana Airport is named after Makabana
- Calvin Booth was born in Reynoldsburg
- The Telltale Head is followed by Life on the Fast Lane
- Alajos Keserű is a sibling of Ferenc Keserű
- Long An contains the administrative territorial entity Châu Thành

D Hyperparameter settings and replicability statement

Entity representation	Architecture	Hyper-param.	Value
Symbolic	LSTM	layers	2
		hidden size	256, 1024
		dropout	0.0
		learning rate	0.001
		lr-scheduler	plateau
		optimizer	Adam
	Transformer	model name	RoBERTa-base
		layers	12
		hidden size	768
		learning rate	5e-5
		lr-scheduler	plateau
		optimizer	Adam
Surface form	LSTM	layers (enc)	2
		hidden size (enc)	256, 1024
		layers (dec)	2
		hidden size (dec)	256, 1024
		learning rate	0.001
		lr-scheduler	plateau
		optimizer	Adam
	Transformer	model name (enc)	RoBERTa-base
		layers (enc)	12
		hidden size (enc)	768
		dropout	0.0
		model name (dec)	random init.
		layers (dec)	12
		hidden size (dec)	768
		learning rate	5e-4
		lr-scheduler	inverse sqrt
		optimizer	Adam
Continuous	LSTM	layers	2
		hidden size	256, 1024
		dropout	0.0
		learning rate	0.001
		lr-scheduler	plateau
		optimizer	Adam
		entity emb. dim	64
		entity emb. trainable	no
	Transformer	model name	RoBERTa-base
		layers	12
		hidden size	768
		learning rate	5e-5
		lr-scheduler	plateau
		optimizer	Adam
		entity emb. dim	64
		entity emb. trainable	no

Table 4: Hyperparameter settings used in our experiments.

E Embeddings of Wikidata entities

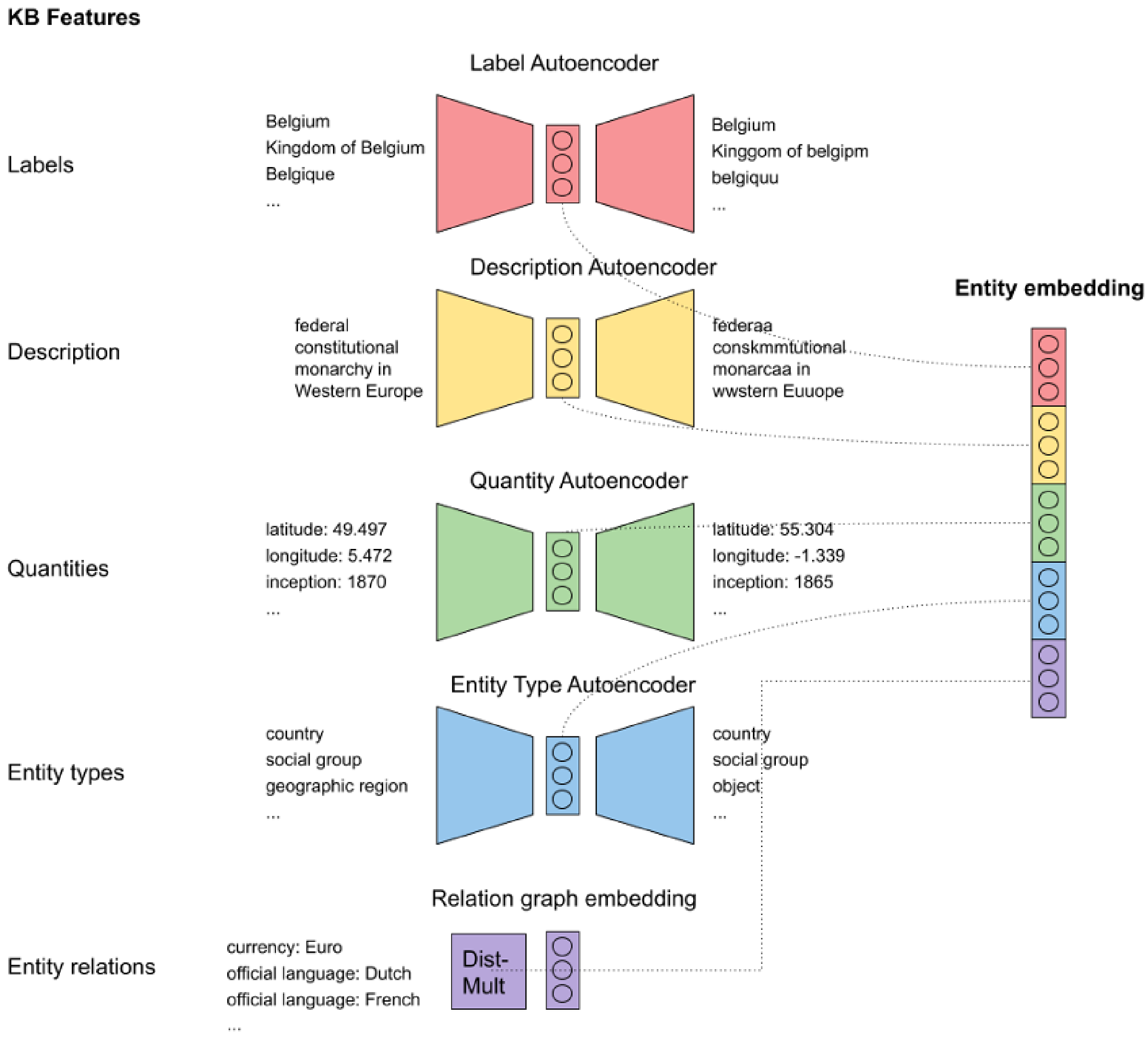


Figure 10: Training embeddings of Wikidata entities with feature-specific autoencoders.

We train the embedding of a given Wikidata entity by collecting its features from, encoding each feature to obtain a dense feature representation, and then concatenating feature representations. For textual features, we use RoBERTa-base as encoder and train corresponding decoders in a standard sequence-to-sequence auto-encoding setup. For quantities, we select the 100 most common quantity types to obtain a fixed-sized representation and then follow a standard auto-encoding setup. Similarly we obtain a fixed-size entity type representation by selecting the 1000 most common entity types. The concatenated feature-representations are then compressed to embedding size d, using a separate autoencoder. Preliminary experiments with embedding sizes $d \in \{64, 128, 192, 256\}$ showed similar memorization accuracies for all d, but faster convergence for smaller sizes. We set $d = 64$ in our main experiments.

F Things that didn't work

F.1 Hierarchical entity representation with binary codes

Since imposing a hierarchy is a common method for dealing with large vocabulary sizes (Morin and Bengio, 2005) in general, and large inventories of entities and entity types in particular (Raiman and Raiman, 2018; López et al., 2019), we created a hierarchy of all entities in Wikidata, using a given entity's position in this hierarchy as training signal. Specifically, we created the entity hierarchy by fitting a KD-tree (Bentley, 1975; Virtanen et al., 2020) with leaf size 1 over pretrained entity embeddings, thereby obtaining a binary partitioning of the embedding space in which each final partition contains exactly one entity embedding. The path from the KD-tree's root to a leaf can be represented as a binary code, which we use as training signal (Oda et al., 2017). Memorization accuracy of world knowledge facts with object entities represented in the form of these binary codes was substantially lower compared to the three approaches described in the main part of this work.

F.2 Training entity embeddings with negative sampling

Instead of using fixed, pretrained entity embeddings as training signal, we experimented with randomly initialized embeddings that are updated during training, using between 1 and 50 in-batch negative samples, which is a standard method in the knowledge base embedding literature (Bordes et al., 2013) and has been used successfully for entity retrieval (Gillick et al., 2019). However, compared to using fixed, pretrained entity embeddings without negative sampling, we observed lower memorization accuracies and slower convergence in our experiments.

F.3 Updating pretrained entity embeddings during training

Instead of using fixed entity embeddings, we tried updating them during training with in-batch negative sampling. This increased the number of trainable parameters, memory usage, and training time, but did not lead to higher memorization accuracies.

F.4 Continuous representation with Euclidean distance loss

Instead of normalizing entity embeddings to the unit hypersphere and training with cosine loss, we experimented with predicting the original pretrained entity embeddings and using the Euclidean distance as loss. Compared to using spherical entity embeddings as prediction targets, we observed slower convergence and lower memorization accuracies.

G Impact of graph type on memorizability

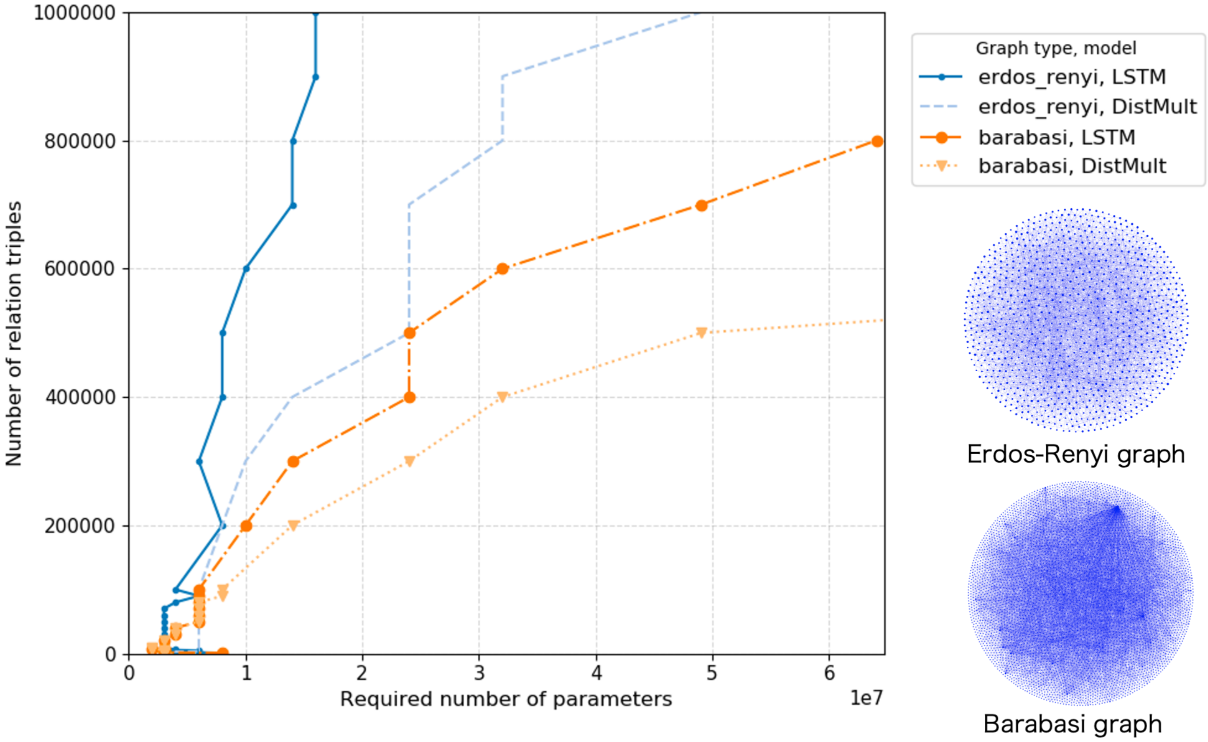


Figure 11: Impact of graph type on a model's ability to memorize the graph. We consider two types of random graphs, namely a uniform (Erdos-Renyi) graph, and a scale-free (Barabasi) graph. We interpret graph edges as relation triples in a knowledge graph and train models to predict the relation object, given subject and predicate, until memorization accuracy reaches 99 percent. For a given number of model parameters, we gradually increase the number of relation triples to memorizes and record the maximum number of relation triples memorized for this number of parameters. We compare an LSTM, as well as a bilinear KB embedding (DistMult). For a given parameter budget, models are able to memorize more triples from a Erdos-Renyi graph (blue) than from a Barabasi graph, indicating that the latter is more difficult to memorize.

Globalizing BERT-based Transformer Architectures for Long Document Summarization

Quentin Grail
NAVER LABS Europe,
Meylan, France
Univ. Grenoble Alpes, CNRS, LIG,
Grenoble, France
quentin.grail@naverlabs.com

Julien Perez
NAVER LABS Europe,
Meylan, France
julien.perez@naverlabs.com

Eric Gaussier
Univ. Grenoble Alpes, CNRS, LIG,
Grenoble, France
eric.gaussier@imag.fr

Abstract

Fine-tuning a large language model on downstream tasks has become a commonly adopted process in the Natural Language Processing (NLP) (Wang et al., 2019). However, such a process, when associated with the current transformer-based (Vaswani et al., 2017) architectures, shows several limitations when the target task requires to reason with long documents. In this work, we introduce a novel hierarchical propagation layer that spreads information between multiple transformer windows. We adopt a hierarchical approach where the input is divided in multiple blocks independently processed by the scaled dot-attentions and combined between the successive layers. We validate the effectiveness of our approach on three extractive summarization corpora of long scientific papers and news articles. We compare our approach to standard and pre-trained language-model-based summarizers and report state-of-the-art results for long document summarization and comparable results for smaller document summarization.

1 Introduction

Language model pre-training has become a key component to improve performances on a majority of Natural Language Processing (NLP) tasks (Wang et al., 2019). Most of the recent competitive architectures (Devlin et al., 2019; Lan et al., 2020; Liu et al., 2019b; Radford et al., 2018) are based on the efficient transformer layer introduced in Vaswani et al. (2017). BERT (Devlin et al., 2019) is one of these architectures that has been widely adopted for comprehension and generation tasks. It is a multi-layer transformer network, pre-trained with different self-supervised objectives. Numerous variations of transformer architectures have been proposed to improve this approach (Lan et al., 2020; Liu et al., 2019b; Radford et al., 2018). However, this type of process is only evaluated on tasks composed of relatively short input text, GLUE (Wang et al., 2019), SQUAD (Rajpurkar et al., 2016), SWAG (Zellers et al., 2018). Indeed, for the tasks that require reasoning with longer documents, this approach exhibits several limitations. The transformer self-attention memory quadratically increases with the number of input tokens, making it technically impossible to compute on document-scale sequences. In addition, they usually require to define a fixed maximum input length, typically of 512 tokens, at the pre-training stage.

One solution is to pre-train the entire model on longer sequences. However, this will still require a massive computation power and will only push the length limitation further. Other alternatives have been proposed to extend multi-layer transformers architectures to longer sequences without modifying this maximum length limitation. The first one is to limit the input sequence to its first tokens by removing the text beyond the length limit. Obviously, it cannot be a reasonable solution to treat long documents that are consistently longer than this limit. The second alternative is to apply the model on a window that slides all over the document. It has been used in Wolf et al. (2019) to deal with SQUAD documents that are longer than the 512 token limitation and in Joshi et al. (2019) for a co-reference resolution task on long documents. This approach can only work if the tokens need to be contextualized only in their surroundings because there is no interaction between the different windows. It seems to be a solution for co-reference resolution (Joshi et al., 2019) as they usually can be solved with a reasonably sized window. Another

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1792–1810
April 19 - 23, 2021.

approach adopted to deal with long documents or multi-document is to select a sub-sample of the input that is small enough for the transformer model. Most of the state-of-the-art pipelines on the multi-hop question answering dataset HotpotQA (Yang et al., 2018) use a first model to retrieve the relevant pieces of text before feeding them to a transformer-based architecture (Fang et al., 2019a; Tu et al., 2019).

We argue that these solutions are not feasible to deal with tasks that require a global understanding of long documents. An example is extractive summarization, where the decision for each sentence should be based on the information of the complete document. To address these challenges, we propose a simple adaptation of the multi-layer transformer architecture that can scale to long documents and benefit from pre-trained parameters with a relatively small length limitation. The general idea is to independently apply a transformer network on small blocks of a text, instead of a long sequence, and to share information among the blocks between two successive layers. To the best of our knowledge, this is the first attempt to introduce hierarchical components directly between the layers of a pre-trained model and not only on top of it (Fang et al., 2019b; Zhang et al., 2019b; Tu et al., 2020). Between each of the transformer layers, we use a Bidirectional Gated Recurrent Unit (BiGRU) network (Cho et al., 2014) to spread global information across the blocks. Adding these propagation layers between the transformer layers preserves the original structure of the pre-trained model and makes it possible to transfer parameter weights from a large pre-trained language model with only few additional parameters to propagate information between blocks.

The contributions of this paper can be summarized as follows: (i) we propose a novel architecture dedicated to long documents which interweaves recurrent hierarchical modules with transformer layers and which exploits pre-trained language models like BERT, and (ii) we demonstrate that this architecture is able to build informative representations in the context of extractive summarization.

2 Global BERT-based Transformer Architecture

In this part, we briefly recall the transformer layer from Vaswani et al. (2017) and its integration in the BERT model (Devlin et al., 2019). Then we describe our modifications of this architecture that allow the model to read longer documents.

Transformers: The transformer architecture, based on a sequence of transformer layers, has been initially introduced in Vaswani et al. (2017). The key idea of this layer is to produce a contextualized representation of an input sequence of tokens. It is composed of the succession of a multi-head self-attention, a first normalizer, a feed-forward neural network, and a second normalizer. This model, which has originally been introduced for machine translation, has then been adopted for most natural language comprehension tasks. Most of the successful approaches (Devlin et al., 2019; Liu et al., 2019b; Lan et al., 2020) are composed of multiple stacked transformer layers. In the remainder, we denote by T^ℓ the transformation corresponding to the ℓ^{th}, $1 \leq \ell \leq L$, transformer layer (T^ℓ is a function from $\mathbb{R}^{N \times h}$ to $\mathbb{R}^{N \times h}$, where N denotes the length of the sequence and h the hidden dimension).

BERT (Devlin et al., 2019) is a multi-layer transformer encoder pre-trained on large text corpora. Two BERT architectures have been proposed in Devlin et al. (2019): $\text{BERT}_{\text{BASE}}$ composed of 12 stacked transformer layers with hidden dimension of 768 ($L = 12, h = 768$) and $\text{BERT}_{\text{LARGE}}$ composed of 24 layers of hidden dimension 1024 ($L = 24, h = 1024$). For both architectures, the input length is limited to 512 WordPiece tokens and the pre-training includes two self-supervised tasks, namely masked language modeling and next sentence prediction. For masked language modeling, 15% of all the WordPiece tokens of the input sequence are masked or corrupted, and the model is used to predict the original token with a cross-entropy loss. For next sentence prediction, the model is trained as a classifier to predict if two sentences are contiguous or not. The pre-training procedure uses the BooksCorpus (Zhu et al., 2015) and documents from English Wikipedia. It requires 4 days of optimization on 16 TPU chips for $\text{BERT}_{\text{BASE}}$ and 64 TPU chips for $\text{BERT}_{\text{LARGE}}$.

2.1 Stacked Propagation Layers

We propose a hierarchical structure that uses pre-trained transformers to encode local text blocks that will be used to compute document level representations. The novel contribution of this work, depicted in Figure 1, is to incorporate recurrent hierarchical modules between the different transformer layers and not only on top of the model, as proposed in

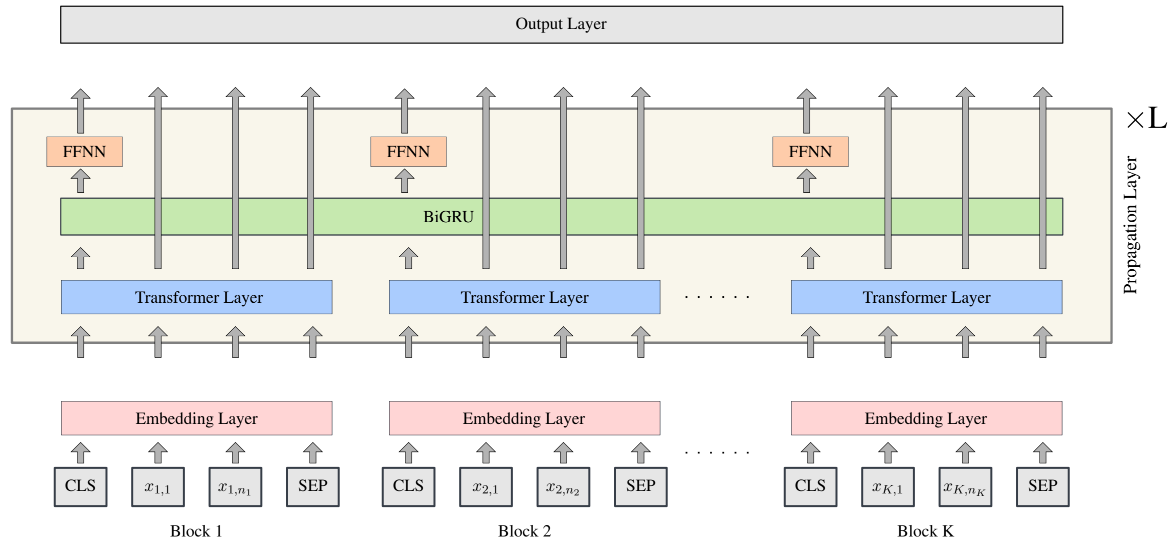


Figure 1: Our proposed modification of a multi-layer transformer architecture. The input sequence is composed of K blocks of tokens. Each transformer layer is applied within the blocks, and a bidirectional GRU network propagates information in the whole document by updating the [CLS] representation of each block.

several recent works (Fang et al., 2019b; Zhang et al., 2019b; Tu et al., 2020). Because we construct and propagate document level information between the layers, global and local information are fused at every level of the architecture. The text blocks can be sentences, paragraphs, or sections. We experiment using sentences as blocks because it generally does not exceed the maximum length allowed by pre-trained models and because BERT has demonstrated to be well adapted to represent such sequences.

We start by splitting the original sequence into multiple blocks. Let D be a document composed of K blocks, $D = \{B_1; B_2; \cdots ; B_K\}$ where a block B_k, $1 \leq k \leq K$, is composed of n_k tokens. To follow the convention of BERT, special tokens [CLS] and [SEP] are respectively added at the beginning and end of each block of the document, so that: $B_k = \{[\text{CLS}]; x_{k,1}; x_{k,2}; \cdots x_{k,n_k}; [\text{SEP}]\}$ where $x_{k,i}$ is the index of the WordPiece token i of block k. In the remainder, the index 0 (resp. $n_k + 1$) will be used to refer to the representation of the [CLS] (resp. [SEP]) token in each block.

Embedding Layer Because our goal is to reuse the available pre-trained BERT parameters, token representations are kept the same as in the original BERT and are composed of a token embedding, a segment embedding, and a positional encoding that represents the position of the token in its block. We will denote by E_k ($E_k \in \mathbb{R}^{(n_k+2)\times h}$, $1 \leq k \leq K$) the embedding representation of block k.

Propagation Layers Our model is composed of L stacked identical hierarchical layers, called *propagation layers*, that comprise a transformer layer, a BiGRU to propagate information across blocks and, finally, a feed-forward network. For any layer ℓ, $1 \leq \ell \leq L$, let $U_k^\ell \in \mathbb{R}^{(n_k+2)\times h}$ be the representation of block k after the $(\ell - 1)^{th}$ layer, the representation for the first layer being initialized with the output of the embedding layer: $U_k^1 = E_k, \quad \forall k \in \{1, \cdots, K\}$. We first apply the pre-trained transformer function T^ℓ individually on each block of the document to compute local, token-aware representations $V_k^\ell \in \mathbb{R}^{(n_k+2)\times h}$:

$$V_k^\ell = T^\ell(U_k^\ell), \quad \forall k \in \{1, \cdots, K\}.$$

The next step is to propagate information across all the blocks of the document in order to compute a global block-aware representation for the document at layer ℓ, denoted by $W^\ell \in \mathbb{R}^{K\times h}$, $1 \leq k \leq K$. To do so, we use a BiGRU network, fed with the representation vectors of the different blocks, and apply a feed-forward neural network to preserve the hidden dimension of the transformer. Each block k is represented by its [CLS] vector, *i.e.*, the vector (represented by $V_{k,0}^\ell \in \mathbb{R}^h$) at the first position in the local representation of the block. These representations are then concatenated to form the input to the BiGRU. The global, block-aware representation is then computed by applying the feed-forward neural network (FFNN) to all K outputs of the BiGRU:

$$W_k^\ell = \text{FFNN}(\text{BiGRU}_k([V_{1,0}^\ell; \cdots ; V_{K,0}^\ell])),$$

Datasets	avg. doc length sentences	avg. doc length words	avg. summary length sentences	avg. summary length words
arXiv	204	5038	5.6	165
PubMed	88	3235	6.8	205
CNN/DM	32	757	4.1	57

Table 1: Statistics on arXiv, PubMed and CNN/DailyMail validation datasets in terms of documents and summary lengths.

where BiGRU_k denotes the k^{th} output of the Bi-GRU and $[;]$ is the concatenation operation.

At this stage, we have computed, for a given document, local block representations $V_k^\ell \, (1 \le k \le K)$ and a global representation W^ℓ. We combine them to build the output representation of the layer:

$$U_k^{\ell+1} = [W_k^\ell; V_{k,1}^\ell; \cdots ; V_{k,n_k+1}^\ell], \quad 1 \le k \le K.$$

As one can note, $U_k^{\ell+1} \in \mathbb{R}^{(n_k+2)\times h}$ is a representation of block k in which the [CLS] vector representation has been enriched with document level information propagated from other blocks. $U_k^{\ell+1}$ is then used as input for the next propagation layer.

2.2 Output Layer

In this work, we validate our approach on the task of extractive summarization described in Section 3. This task can be considered as a binary classification problem where each block has to be labeled as selected or not. We use a feed-forward neural network followed by a Softmax function on the top of the block level representations after the last layer L to compute $Y \in \mathbb{R}^{K\times 2}$.

$$Y_k = \text{Softmax}(\text{FFNN}(W_k^{L+1})).$$

Using a recurrent architecture to propagate information between blocks has two interesting properties. First, it allows our model to scale to long sequences of blocks without using an attention mechanism that would not scale. Second, it does not require to implement any positional encoding on block representations.

3 Experiments

We evaluate our approach, which we refer to as GBT-EXTSUM (for 'Global BERT-based Transformer for Extractive Summarization'), in the context of extractive summarization, the goal of which being to identify and extract from a document the pieces of text that are the most important (Kupiec et al., 1995). We view this task as a sentence-level classification problem where each sentence has to be labeled according to its belonging to the summary or not. To validate the effectiveness of our approach, we propose to test it on three summarization datasets, namely ArXiv, PubMed and CNN/DailyMail:

- The **ArXiv** and **Pubmed** datasets have been introduced in Cohan et al. (2018). They contain long scientific documents from `arXiv.org` and `PubMed.com` and use their abstracts as the ground-truth summaries. We use the original splits that respectively contain 203,037/6,436/6,440 samples in the training, validation, and test sets for arXiv, and 119,924/6,633/6,658 for PubMed.
- The **CNN/DailyMail** dataset contains news articles associated with short summaries. We use the splits of Hermann et al. (2015), where entities have not been anonymized. This dataset contains 287,226 training samples, 13,368 validation samples, and 11,490 test samples.

Table 1 presents some statistics on these three datasets. As one can note, for the scientific articles, the average number of tokens in the documents to summarize is way beyond the capabilities of a standard transformer pre-trained with BERT.

3.1 Evaluation Metrics

We evaluate the quality of the extracted summaries using the ROUGE metric (Lin, 2004), and more particularly ROUGE-1 (overlap of unigrams), ROUGE-2 (overlap of bigrams), ROUGE-3 (overlap of trigrams) and ROUGE-L (longest common subsequence between the produced summary and the gold-standard one).

3.2 Label Generation

In order to train extractive summarizers, one needs annotations in the form of sentence-level binary labels. To compute such annotations, we follow the work of Kedzie et al. (2018) and label all sentences by greedily optimizing the ROUGE-1 score of the extracted summary against the gold-standard summary associated with each article. These labels are only used at training time, the evaluation of the extracted summaries being done against the gold-standard summaries provided in the datasets.

		PubMed				arXiv			
	Summarizer	RG-1	RG-2	RG-3	RG-L	RG-1	RG-2	RG-3	RG-L
	Oracle	58.15	34.16	24.11	52.99	57.78	30.43	18.41	51.24
	Lead	37.77	13.35	7.64	34.31	35.54	9.50	3.33	31.19
Abstractive or Mix	Attn-Seq2Seq (Nallapati et al., 2016)	31.55	8.52	7.05	27.38	29.30	6.00	1.77	25.56
	Pntr-Gen-Seq2Seq (See et al.)	35.86	10.22	7.60	29.69	32.06	9.04	2.15	25.16
	Discourse summarizer (Cohan et al., 2018)	38.93	15.37	9.97	35.21	35.80	11.05	3.62	31.80
	TLM-I+E (G,M) (Subramanian et al., 2019)	42.13	16.27	8.82	39.21	41.62	14.69	6.16	38.03
	DANCER PEGASUS (Gidiotis and Tsoumakas, 2020)	46.34	19.97	-	42.42	45.01	17.60	-	40.56
	PEGASUS (Zhang et al., 2019a)	45.97	20.15	-	28.25	44.21	16.95	-	25.67
	BigBird-Pegasus (Zaheer et al., 2020)	46.32	20.65	-	42.33	46.63	19.02	-	41.77
Extractive	SumBasic (Vanderwende et al., 2007)	37.15	11.36	5.42	33.43	29.47	6.95	2.36	26.30
	LexRank (Erkan and Radev, 2004)	39.19	13.89	7.27	34.59	33.85	10.73	4.54	28.99
	LSA (Steinberger and Jezek, 2004)	33.89	9.93	5.04	29.70	29.91	7.42	3.12	25.67
	Sent-CLF (Subramanian et al., 2019)	45.01	19.91	**12.13**	41.16	34.01	8.71	2.99	30.41
	Sent-PTR (Subramanian et al., 2019)	43.30	17.92	10.67	39.47	42.32	15.63	7.49	38.06
	Bert Ranker (Nogueira and Cho, 2019)	43.67	18.00	10.74	39.22	41.65	13.88	5.92	36.40
	BertSumExt (Liu and Lapata, 2019b)	41.09	15.51	8.64	36.85	41.24	13.01	5.26	36.10
	BertSumExt (SW) (Liu and Lapata, 2019b)	45.01	20.00	12.05	40.43	42.93	15.08	6.01	37.22
	Longformer-Ext (Beltagy et al., 2020)	43.75	17.37	10.18	39.71	45.24	16.88	8.06	40.03
	Reformer-Ext (Kitaev et al., 2020)	42.32	15.91	9.02	38.26	43.26	14.86	6.66	38.10
	GBT-ExtSum (Ours)	**46.87**	**20.19**	12.11	**42.68**	**48.08**	**19.21**	**9.58**	**42.68**

Table 2: Summarization results on PubMed and arXiv. Except for BERT-based approaches, for Reformer-Ext and for Longformer-Ext, which we have reimplemented, the results of the baselines are taken from their associated paper as well as from Cohan et al. (2018). Bold results correspond to the best scores of extractive summarizers.

3.3 Baseline Models

We compare our approach to several well known published methods described below. These methods include SumBasic (Vanderwende et al., 2007), LexRank (Erkan and Radev, 2004), LSA (Steinberger and Jezek, 2004), Attn-Seq2Seq (Nallapati et al., 2016), Pntr-Gen-Seq2Seq (See et al.) and Discourse-aware summarizer (Cohan et al., 2018). The results for these models are the ones reported in the paper (Cohan et al., 2018). We also report the results of Sent-CLF and Sent-PTR, which are hierarchical sentence pointer and classifier, TLM-I+E (G,M) a mixed extractive/generative transformer language model from Subramanian et al. (2019), BigBird (Zaheer et al., 2020), PEGASUS (Zhang et al., 2019a) and DANCER (Gidiotis and Tsoumakas, 2020) which are three abstractive methods. Lastly, we developed several baseline models based on BERT, Longformer (Beltagy et al., 2020) and Reformer (Kitaev et al., 2020):

BERT Ranker: We used a BERT ranker, similar to Nogueira and Cho (2019) in which each sentence of the document is processed individually. We apply BERT on each sentence[1] and use a Sigmoid layer, the input of which consists of the [CLS] representation of the sentence, to model the probability of the sentence to be selected.

BertSumExt has been introduced in Liu and Lapata (2019b). This model is an adaptation of BERT for extractive summarization. Because this model takes as input the concatenation of all the tokens of the document, it cannot scale to the arXiv and PubMed datasets. We propose two variants: the first one is to take as input only the first 800 tokens of the document, as suggested in the original paper. This solution is displayed as BertSumExt in Table 2. The second is to apply BertSumExt per sliding windows on the original document and to use, as a token representation, its representation in the window that maximizes its surrounding context. We name this sliding window implementation BertSumExt (SW) in Table 2.

Longformer-Ext and Reformer-Ext: The Longformer and Reformer models were respectively introduced by Beltagy et al. (2020) and Kitaev et al. (2020). They both propose an adaptation of the Transformer self-attention that scale to long sequences. We add the same classification head as the one used in our model on top of the contextualized representation of the first token of each sentence to label them as selected or not in the summary.

---

[1] This is possible as no sentence exceeds BERT token limitation.

We also present the Oracle extractive results as an upper bound as well as the Lead baseline (which respectively select the first 3, 6, 7 sentences for CNN/DailyMail, arXiv and PubMed datasets). Several models are reported only on CNN/DailyMail dataset and not on arXiv/Pubmed as they do not scale to long documents.

3.4 Implementation details

We run all our experiments using the Pytorch library (Paszke et al., 2019). We built our model using the "bert-base-uncase"[2] version of BERT and its implementation in the HuggingFace library (Wolf et al., 2019). Our architecture is composed of $L = 12$ propagation layers with a transformer hidden dimension of $h = 768$. The hidden dimension of the BiGRU is set to 384 and we share its parameters among all the propagation layers. The FFNN inside the propagation layers maps the output of the BiGRU of dimension 2×384 to a vector of dimension 768. The FFNN of the output layer is a binary classifier that projects the sentence representations of dimension 768 to an output of dimension 2. We fine-tuned our model on the cross-entropy loss, for 5 epochs on 4 GPUs V100 and use Adam optimizer (Kingma and Ba, 2015) with the initial learning rate set to 3×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, no learning rate warmup and a linear decay of the learning rate. We describe implementation details of BERTSUMEXT, Longformer-Ext and Reformer-Ext baselines in the Supplementary Material, Appendix A.

We used *Trigram Blocking* to avoid the repetition of trigrams in the extracted summaries as suggested in Paulus et al. (2018). Given the extracted summary so far, we only added candidate sentences that had no overlapping trigram with the current summary. We limited the summary to 3 sentences for the CNN/DailyMail dataset, 6 sentences for arXiv, and 7 for PubMed.

3.5 Results

Our main results are shown in Tables 2 and 3. On the arXiv and PubMed datasets, our model outperforms the baseline models on almost all of the reported metrics. Our approach manages to summarize long documents while preserving informativeness (evaluated by ROUGE-1) and fluency (evaluated by ROUGE-L) of the summaries. In addition to the previously published methods, our approach also improves over the BERT-based, Longformer-Ext and Reformer-Ext baselines we have developed. Among them, BERTSUMEXT, which focuses on a truncated version of the document, is the less effective. As documents are significantly longer than the 800 tokens limitation of this model, this result is not surprising. The sliding window adaptation of this model, that allows it to scale to long documents, is the one that achieves results that are the most comparable to ours. Our approach still outperforms this adaptation, demonstrating that summaries require to propagate information beyond a single BERT window.

Model	R-1	R-2	R-L
Oracle	56.22	33.74	52.19
Lead-3	40.11	17.54	36.32
LATENT (Zhang et al., 2018)	41.05	18.77	37.54
NEUSUM (Zhou et al., 2018)	41.59	19.01	37.98
SUMO (Liu et al., 2019a)	41.00	18.40	37.20
TransformerExt (Liu and Lapata, 2019b)	40.90	18.02	37.17
MASK-LM^{global} (Chang et al., 2019)	41.2	19.1	37.6
PNBERT (Zhong et al., 2019)	42.69	19.60	38.85
BERT-ext + RL (Bae et al., 2019)	42.76	19.87	39.11
HIBERT$_M$ (Zhang et al., 2019b)	42.37	19.95	38.83
BERTSUMEXT (Liu and Lapata, 2019b)	43.25	20.24	39.63
BERTSUMEXT w/o interval embedding	43.20	20.22	39.59
BERTSUMEXT (large)	43.85	20.34	39.90
MatchSum (RoBERTa) (Zhong et al., 2020)	**44.41**	**20.86**	**40.55**
Reformer-Ext (Kitaev et al., 2020)	38.85	16.46	35.16
Longformer-Ext (Beltagy et al., 2020)	43.00	20.20	39.30
GBT-EXTSUM (Ours)	42.93	19.81	39.20

Table 3: Comparison of ROUGE scores on CNN/DailyMail wrt extractive models. All results are taken from original papers but Reformer-Ext and Longformer-Ext which we have reimplemented.

On the CNN/DailyMail dataset, one can see that our model outperforms all the models that do not use pre-trained parameters. This includes several transformer-based and hierarchical models. However, while having comparable results, we do not achieve stronger performance than the current extractive state of the art from Zhong et al. (2020). This is not surprising as the majority of the CNN/DailyMail examples contains their oracle summary sentences in the first positions of the articles, as shown in the Supplementary Material, Appendix B.

Lastly, we evaluate the impact of several elements of our proposed model in Table 4. We first study the influence of the underlying language model by considering both RoBERTa (Liu et al., 2019b) and PEGASUS (Zhang et al., 2019a) pre-trained models, respectively referred to as GBT-EXTSUM-RoBERTa and GBT-EXTSUM-

---

[2] `https://github.com/google-research/bert`

Model	PubMed R-1	PubMed R-2	PubMed R-L	arXiv R-1	arXiv R-2	arXiv R-L
GBT-ExtSum	46.87	20.19	42.68	48.08	19.21	42.68
GBT-ExtSum-RoBERTa	46.02	19.29	41.84	47.42	18.62	42.03
GBT-ExtSum-PEGASUS	44.11	17.34	40.03	43.50	15.35	38.41
GBT-ExtSum-NoShare	46.84	20.19	42.63	48.11	19.30	42.75
GBT-ExtSum-AveragePool	45.24	18.13	40.94	45.71	17.36	40.43
GBT-ExtSum-Transformer	46.46	19.62	42.17	47.64	18.82	42.22

Table 4: Analysis of the influence of different key components of our proposed architecture.

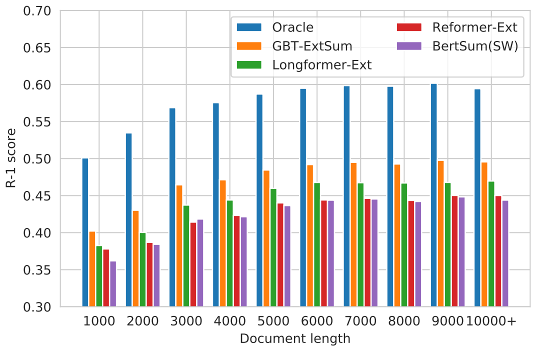


Figure 2: Average R-1 scores of extracted summaries according to the number of words in the input documents from arXiv test dataset.

PEGASUS. As one can see, the results show that BERT-base architecture performs best in terms of ROUGE scores on both arXiv and PubMed. One major difference between PEGASUS and BERT/RoBERTA pre-trained models is that BERT/RoBERTA are only encoders while PEGASUS is a pre-trained encoder/decoder architecture. This could explain why BERT/RoBERTA outperform PEGASUS on extractive summarization tasks. We then compare an alternative of our implementation of GBT-ExtSum in which the parameters of the BiGRU are not shared among all the propagation layers (GBT-ExtSum-NoShare) and found no clear difference with the version in which the parameters are shared. Lastly, we compare three architectures of propagation layers, including an average pooling of the [CLS] representations of the sentences, a Transformer layer between the [CLS] tokens (associated to a block position embedding), and a BiGRU layer. Among these three layers, the average pooling layer, which introduces no additional trainable parameters, performs the worst. Furthermore, the BiGRU layer slightly outperforms the Transformer layer in terms of ROUGE scores.

Analysis. In Figure 2, we compare the R-1 score of several models regarding the number of words in the source documents. One can see that GBT-ExtSum consistently outperforms BertSumExt (SW), Reformer-Ext and Longformer-Ext regardless of the number of words in the source documents.

We present in Table 5 two example summaries of a document from the PubMed test set (Kamio et al., 2009), respectively obtained by GBT-ExtSum and BertSumExt (SW). The numbers in the margin indicate the position of the sentences in the original document, which is composed of a total of 78 sentences. As one can observe, GBT-ExtSum extracts sentences from various parts of the document whereas BertSumExt (SW) mostly focuses on the beginning of the document. Among the sentences selected by the two models, the most meaningful one, in terms of ROUGE, is the last one selected by GBT-ExtSum. This sentence appears at position 66, in the last section (*Discussion*) of the original paper. In contrast, BertSumExt (SW) proposes sentences that are less relevant for summarization purposes. Additional summaries of the PubMed and arXiv articles are provided in the Supplementary Material, Appendices C and D.

To analyse the influence of the positions of the sentences in the input document, we present in Figure 3 the histograms of the positions of the sentences of the Oracle summary as well as that of the predicted positions of different models, on the PubMed test set. One can see that if most relevant sentences appear at the beginning of a document, other Oracle sentences are still relevant further down the document. GBT-ExtSum is the model that behaves the most closely to the Oracle, followed by BertSumExt (SW), Reformer-Ext and Longformer-Ext. These last two models tend to over-select sentences from the beginning while focusing less on the ones appearing later in the document. Our model remains influenced by the sentence position but is still able to select sentences from all over the document and is closer to the Oracle distribution.

4 Related Work

Hierarchical neural architectures have been competitive on a collection of NLP tasks that require to reason over long or multiple documents such as aspect-based sentiment analysis (Paulus et al., 2018), document summarization (Cheng and Lapata, 2016), document segmentation (Koshorek

GOLD		purpose : to investigate whether the glc3a locus harboring the cyp1b1 gene is associated with normal tension glaucoma (ntg) in japanese patients.materials and methods : one hundred forty two japanese patients with ntg and 101 japanese healthy controls were recruited . patients exhibiting a comparatively early onset were selected as this suggests that genetic factors may show stronger involvement . genotyping and assessment of allelic diversity was performed on 13 highly polymorphic microsatellite markers in and around the glc3a locus.results:there were decreased frequencies of the 444 allele of d2s0416i and the 258 allele of d2s0425i in cases compared to controls (p = 0.022 and p = 0.034 , respectively) . however , this statistical significance disappeared when corrected ($pc > 0.05$) . we did not find any significant association between the remaining 11 microsatellite markers , including d2s177 , which may be associated with cyp1b1 , and ntg ($p > 0.05$). conclusions : our study showed no association between the glca3 locus and ntg , suggesting that the cyp1b1 gene , which is reportedly involved in a range of glaucoma phenotypes , may not be an associated factor in the pathogenesis of ntg .
GBT-EXTSUM	1-	primary open angle glaucoma (poag) is the most common type of glaucoma .
	15-	we excluded individuals who were diagnosed under 20 or over 60 years of age and who had 8.0 d or higher myopic refractive error of spherical equivalence .
	17-	the cases exhibiting a comparatively early onset were selected as they suggest that genetic factors may show stronger involvement . during diagnosis ,
	30-	the probability of association was corrected by the bonferroni inequality method , ie , by multiplying the obtained p values with the number of alleles compared .
	63-	only two adjacent markers , d2s0416i and d2s0425i , were significantly positive , as shown in table 2 , and the frequency of the 444 allele of d2s0416i and the 258 allele of d2s0425i were decreased in cases compared to controls (p = 0.022 , or = 0.59 and p = 0.034 , or = 0.42 , respectively) .
	66-	the purpose of this study was to investigate whether the glc3a locus is associated with ntg in japanese subjects , based on results from recent studies reporting that the cyp1b1 gene , located at the glc3a locus on chromosome 2p21 , could be a causative gene in poag as well as pcg . to this end , we genotyped 13 microsatellite markers in and around the glc3a locus . here
BERTSUMEXT (SW)	1-	primary open angle glaucoma (poag) is the most common type of glaucoma .
	2-	normal tension glaucoma (ntg) is an important subset of poag ; while many poag patients have high iop,1 patients with ntg have statistically normal iop.24 the prevalence of ntg is higher among the japanese population than among caucasians , and recent studies reported that 92% of poag patients in japan had ntg.58 the diagnosis of glaucoma is based on a combination of factors including optic nerve damage and specific field defects for which iop is the only treatable risk factor .
	7-	of these subjects , 142 were diagnosed with ntg , and 101 were control subjects .
	20-	genomic dna was extracted using the qiaamp dna blood mini kit (qiagen , hilden , germany) or the guanidine method . in this association study , we selected 13 highly polymorphic microsatellite markers that are located in and around the glc3a locus as shown in figure 1 .
	28-	the number of microsatellite repeats was estimated automatically using the genescan 672 software (applied biosystems) by the local southern method with a size marker of gs500 tamra (applied biosystems) .
	22-	polymerase chain reaction (pcr) was performed in a reaction mixture with a total volume of 12.5 l containing pcr buffer , genomic dna , 0.2 mm dinucleotide triphosphates (dntps) , 0.5 m primers , and 0.35 u taq polymerase .

Table 5: An example of summary produced by our method compared to the gold summary and one produced by BERTSUMEXT (SW). With a red scale, we highlight the sentences with the highest ROUGE score when evaluated against the abstract. We show in the margin the position of the extracted sentence in the document. This document (Kamio et al., 2009) is 78 sentences long.

et al., 2018) and text classification (Yang et al., 2016). The hierarchical structure enables the model to learn local contextualized token representations in its lower hierarchy level, while higher-level representations can capture long-distance dependencies within the document. Liu and Lapata (2019a) have proposed a hierarchical modification of the transformer layer-based attention modules to model relations between documents for abstractive summarization but do not investigate parameter transfer form pre-trained language models. Chang et al. (2019) and Zhang et al. (2019b) suggested pre-training processes for hierarchical models, without however testing their approaches on long document summarization nor releasing their pre-trained models. We have not included these models in our comparison for this reason. Transformer-XH (Zhao et al., 2020) introduced an eXtra Hop attention to model dependencies between different transformer windows but requires a graph of related documents.

Long-Document Transformers: Multiple studies have investigated different self-attention mechanisms to extend transformers to long documents. Transformer-XL (Dai et al., 2019) introduced a recurrence between successive transformer windows which run from left to right through the document, preventing global information to bidirectionally flow through the document. Other approaches design the self-attention as a sparse layer, as sparse transformers (Child et al., 2019) or the recently proposed Longformer and BIGBIRD models (Beltagy et al., 2020; Zaheer et al., 2020). One major difference with our work is that these models compute the attention only between a limited set of randomly or *a priori* chosen tokens. Reformer (Kitaev et al., 2020) also tackles the problem of language modeling for long sequences, but it does so by computing the self-attention only between *similar*

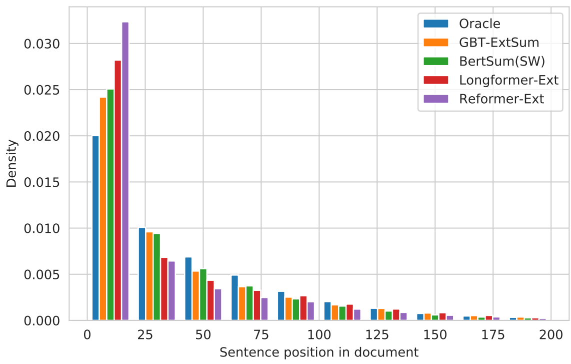


Figure 3: Proportion of the extracted sentences according to their position in the input document from PubMed test dataset.

tokens, based on locality-sensitive hashing.

5 Conclusion

In this paper, we have introduced a novel transformer-based model for long document summarization based on propagation layers that spread information between multiple transformer windows. This model preserves the architecture of commonly used pre-trained language models, thus allowing the transfer of parameters. An evaluation, conducted on top of the BERT model in the context of an extractive summarization task, further revealed its effectiveness in dealing with long documents compared to other adaptations of BERT and previously proposed models. In the future, we plan to adapt our model to other tasks that require understanding long documents, as question-answering and document-scale machine translation.

Acknowledgments

This work was partially supported by MIAI@Grenoble Alpes, (ANR-19-P3IA-0003).

References

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sang-goo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. *CoRR*, abs/1909.08752.

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

Ming-Wei Chang, Kristina Toutanova, Kenton Lee, and Jacob Devlin. 2019. Language model pre-training for hierarchical document representations. *CoRR*, abs/1901.09128.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509.

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

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 615–621. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics.

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

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2019a. Hierarchical graph network for multi-hop question answering. *CoRR*, abs/1911.03631.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2019b. Hierarchical graph network for multi-hop question answering. *CoRR*, abs/1911.03631.

Alexios Gidiotis and Grigorios Tsoumakas. 2020. A divide-and-conquer approach to the summarization of long documents.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28, pages 1693–1701. Curran Associates, Inc.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel S. Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5802–5807. Association for Computational Linguistics.

M Kamio, Akira Meguro, Masao Ota, N Nomura, Kenji Kashiwagi, F Mabuchi, Hiroyuki Iijima, K Kawase, T Yamamoto, M Nakamura, Akira Negi, T Sagara, Teruo Nishida, M Inatani, Hidenobu Tanihara, M Aihara, M Araie, Takeo Fukuchi, H Abe, and Nakamura Mizuki. 2009. Investigation of the association between the glc3a locus and normal tension glaucoma in japanese patients by microsatellite analysis. *Clinical ophthalmology (Auckland, N.Z.)*, 3:183–8.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text segmentation as a supervised learning task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 469–473, New Orleans, Louisiana. Association for Computational Linguistics.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, page 68–73, New York, NY, USA. Association for Computing Machinery.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

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

Yang Liu and Mirella Lapata. 2019a. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5070–5081. Association for Computational Linguistics.

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

Yang Liu, Ivan Titov, and Mirella Lapata. 2019a. Single document summarization as tree induction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1745–1755, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

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

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 8024–8035.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics.

Josef Steinberger and Karel Jezek. 2004. Using latent semantic analysis in text summarization and summary evaluation.

Sandeep Subramanian, Raymond Li, Jonathan Pilault, and Christopher J. Pal. 2019. On extractive and abstractive neural document summarization with transformer language models. *CoRR*, abs/1909.03186.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2019. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. *CoRR*, abs/1911.00484.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. *national conference on artificial intelligence*.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. 2007. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Inf. Process. Manag.*, 43(6):1606–1618.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. *CoRR*, abs/1808.05326.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 779–784, Brussels, Belgium. Association for Computational Linguistics.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. HIBERT: document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5059–5069. Association for Computational Linguistics.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul N. Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1049–1058. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663, Melbourne, Australia. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

A Baselines: Implementation Details

BERTSUMEXT: For all experiments with BERTSUMEXT, we started with the original implementation[3] and adapted the code to build the sliding windows version. This implementation leverage *bert-base-uncased* pre-trained model and its associated hyperparameters. We use windows of width 800 with an overlap of 300 tokens between two following windows. If a sentence is in multiple windows, we select its [CLS] representation in the window that maximizes the number of surrounding tokens. We finetune the model for 5 epochs using Adam optimizer with an initial learning rate of 1×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$.

Longformer-Ext: We built the Longformer-Ext baseline from the Longformer implementation released by HuggingFace[4]. We use the official *longformer-base-4096* pre-trained model trained by AllenAI[5]. This model is based on *RoBERTa-base* and its associated hyperparameters. To increase the maximal position embedding, we drop the pre-trained positional embedding parameters and train a novel token embedding layer to scale Longformer-Ext input up to 12294 tokens. This model computes a sliding self-attention with a window size of 512 tokens on all its 12 Transformer layers. We finetune the model for 5 epochs with only local attention because of memory constraints, using Adam optimizer with an initial learning rate of 1×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, no learning rate warmup and a linear decay of the learning rate.

Reformer-Ext: We started from the HuggingFace implementation of Reformer to build Reformer-Ext baseline. We use a Reformer configuration composed of six layers of attention. We use Locality-Sensitive Hashing Attention with 128 buckets on the input sequence and Local Self-attention on chunks of 64 tokens. We use hidden sates of dimension 256, a feed-forward layer of dimension 512, and 12 attention heads in Transformer encoders. We train this model for 5 epochs using Adam optimizer with an initial learning rate of 1×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, no learning rate warmup and a linear decay of the learning rate.

B Datasets Statistics

Figure 4 presents the distribution of the document lengths in arXiv, PubMed and CNN/DailyMail, after tokenization with pretrained BERT-base tokenizer. It also provides the histograms of the position of the [CLS] tokens of the Oracle sentences in input documents. One can see that the three datasets contain an important number of documents longer than 512 tokens, the standard length limitation of pre-trained language models. However, one can also notice that CNN/DailyMail contains a large part of its Oracle sentences within this first window of 512 tokens. As a consequence, a model that is not able to ”read” beyond this limitation is not penalized. It is also a reason why Lead baseline is quite strong on this dataset. On the contrary, on arXiv and PubMed, one can see that a large part of Oracle sentences occur beyond this 512 windows. This explains why models capable of reading long sequences are required to achieve good results on these datasets.

---

[3] https://github.com/nlpyang/PreSumm

[4] https://github.com/huggingface/transformers

[5] https://github.com/allenai/longformer

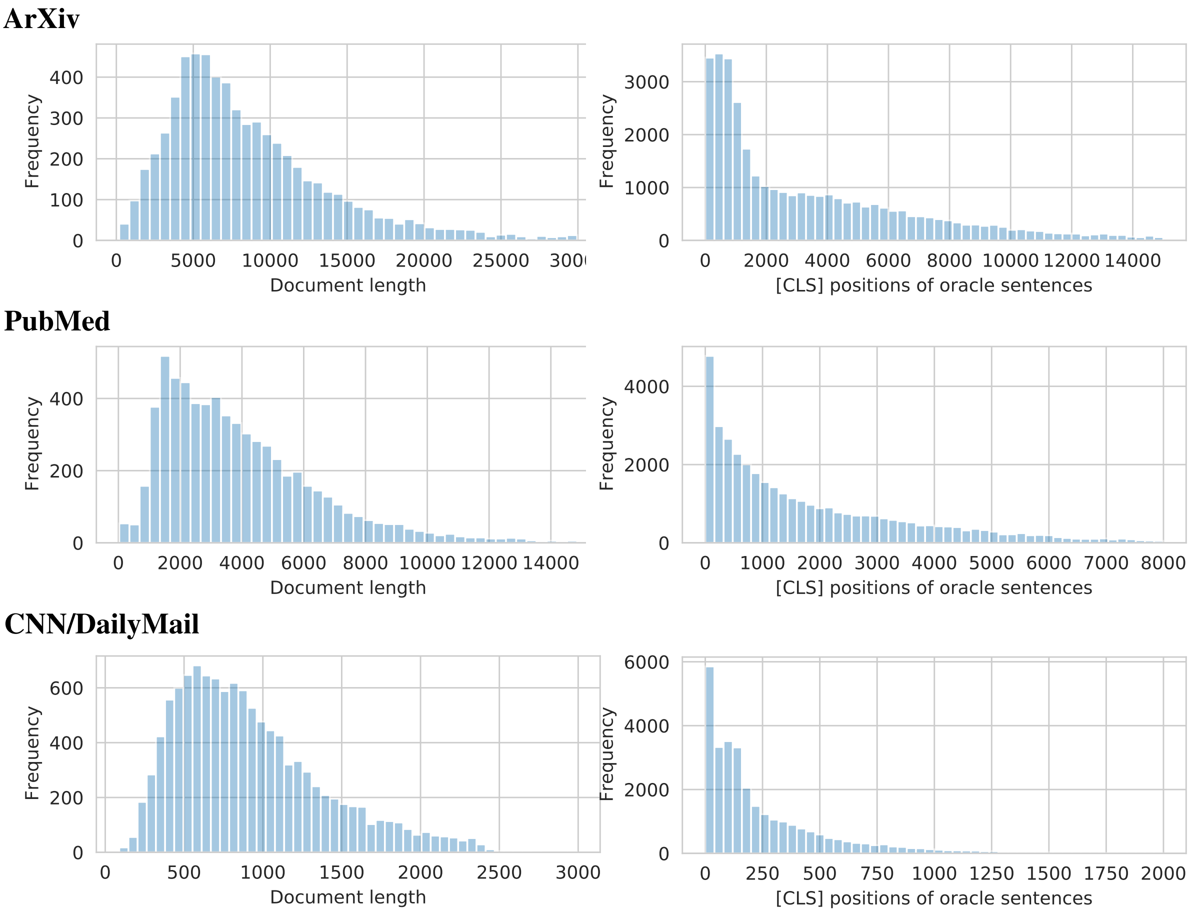


Figure 4: Document lengths after tokenization with pretrained BERT-base tokenizer and position of the [CLS] tokens of Oracle sentences in the input documents.

C PubMed Summaries

Gold		aim . to investigate incidental adrenal enlargement clinical characteristics and functional status and analyze functional lesion risk factors . materials and methods . this retrospective study included 578 patients with adrenal imaging features showing enlargement . incidental adrenal enlargement cases (78) were considered eligible . demographics , functional diagnosis , adrenal imaging features , and concomitant diseases were analyzed . results . the number of adrenal enlargements and proportion of incidental adrenal enlargement increased each year . mean patient age was 50.32 years . thirty - nine cases had unilateral enlargement on the left side and 3 on the right side ; 36 had bilateral enlargement . routine medical checkup was found to have the greatest chance (43.59%) of revealing clinical onsets leading to discovery . biochemical and functional evaluation revealed 54 (69.23%) cases of nonfunctional lesions , 12 (15.38%) of subclinical cushing syndrome , 6 (7.69%) of primary hyperaldosteronism , 1 (1.28%) of metastasis , and 5 (6.41%) of unknown functional status . nodular adrenal enlargement (or , 7.306 ; 95% ci , 1.72728.667 ; p = 0.006) was a risk factor for functional lesions . age and lesion location were not significant factors . conclusion . incidental adrenal enlargement is a frequent radiographic finding and is accompanied by diverse clinical factors that require proper evaluation and management . nodular adrenal enlargement was a risk factor .
GBT-ExtSum	8-	data retrieved included patient demographics , final functional diagnosis , adrenal imaging features , and concomitant diseases .
	14-	smooth enlargement was defined as enlargement of the gland with a smooth contour and no measureable or diffuse nodules . after obtaining patient history and physical examination , all patients underwent biochemical evaluation to assess their functional status .
	16-	patients with an aldosterone - rennin ratio (arr) > 20 underwent any 1 of 3 confirmatory tests (saline infusion , captopril challenge , or postural stimulation) to confirm or exclude definitively primary hyperaldosteronism (pa) .
	25-	as shown in table 1 , routine medical checkup was found to have the greatest chance (43.59%) of revealing clinical onsets leading to the discovery of adrenal enlargement .
	29-	nodular adrenal enlargement (or 7.306 ; 95% ci , 1.72728.667 ; p = 0.006) was the risk factor for functional lesions .
	31-	our study shows that the proportion of incidental adrenal enlargement has gradually increased by year .
	46-	acth - independent macronodular hyperplasia (aimah) and primary pigmented nodular adrenal hyperplasia often manifest as adrenal hyperplasia . the clinical features of aimah tended to be atypical .
BertSumExt (SW)	4-	it is a common term for a variety of adrenal disorders , but its cause must be properly assessed so that patients needing treatment , such as those with hormone hypersecretion or malignant disease , can receive appropriate care . however , there is a lack of literature on functional status and its follow - up to provide comprehensive insight to these findings .
	5-	patients with incidental adrenal enlargement were evaluated in a tertiary referral hospital with endocrinological departments in china .
	7-	this retrospective study included 578 patients with adrenal imaging features showing adrenal enlargement who were hospitalized at the department of endocrinology in pla general hospital (beijing , china) between january 1993 and july 2013 .
	29-	nodular adrenal enlargement (or 7.306 ; 95% ci , 1.72728.667 ; p = 0.006) was the risk factor for functional lesions .
	36-	in addition , smooth enlargement was more common , in 53 (83%) cases , and together these statistics reflect the likelihood that adrenal enlargement will be bilateral , smooth , and found in men .
	37-	however , our study did not show this tendency , likely because the research goals and thus , study populations , differed between the 2 studies .
	38-	's study aimed to explore prevalence , while the present study aimed to evaluate functional status .

GOLD		background and objective . antimicrobial resistance is now a major challenge to clinicians for treating patients . hence , this short term study was undertaken to detect the incidence of multidrug - resistant (mdr) , extensively drug - resistant (xdr) , and pandrug - resistant (pdr) bacterial isolates in a tertiary care hospital . material and methods . the clinical samples were cultured and bacterial strains were identified in the department of microbiology . the antibiotic susceptibility profile of different bacterial isolates was studied to detect mdr , xdr , and pdr bacteria . results . the antibiotic susceptibility profile of 1060 bacterial strains was studied . 393 (37.1%) bacterial strains were mdr , 146 (13.8%) strains were xdr , and no pdr was isolated . all (100%) gram negative bacterial strains were sensitive to colistin whereas all (100%) gram positive bacterial strains were sensitive to vancomycin . conclusion . close monitoring of mdr , xdr , or even pdr must be done by all clinical microbiology laboratories to implement effective measures to reduce the menace of antimicrobial resistance .
GBT-EXTSUM	5-	multidrug resistant (mdr) was defined as acquired nonsusceptibility to at least one agent in three or more antimicrobial categories . extensively drug
	36-	no mdr or xdr strain was isolated from streptococcus sp . all (100%) gram positive cocci were sensitive to vancomycin and linezolid .
	38-	e. coli was the commonest isolate 261 (35%) , followed by pseudomonas aeruginosa 212 (28.4%) .
	40-	out of 200 klebsiella pneumoniae strains isolated , 75 (37.5%) and 25 (12.5%) were detected as mdr and xdr , respectively . out of 42 acinetobacter and other nonfermenter species isolated , 19 (45.2%) and 8 (19%) were mdr and xdr strains , respectively . amongst 250 gnb - mdr strains isolated ,
	62-	, it has been reported that most frequent mdr pathogens were pseudomonas aeruginosa followed by e. coli .
	67-	unless and until multidrug resistant organisms are detected and their incidence is known , the strategies for their control can not be adopted properly in healthcare setup . hence , detection , prevention of transmission of mdros by following infection control practices , antimicrobial surveillance , and stewardship are need of the hour .
	69-	we hereby conclude that early detection and close monitoring of mdr , xdr , or even pdr bacterial strains must be started by all clinical microbiology laboratories to reduce the menace of antimicrobial resistance which is now a global problem .
BERTSUMEXT (SW)	9-	this short term cross - sectional study was conducted in the department of microbiology from 15th of april to 15th of july , 2014 .
	10-	the bacterial strains were isolated from different clinical samples and were identified by conventional methods .
	17-	methicillin resistant staphylococcus aureus (mrsa) strains were detected by meca - mediated oxacillin resistance using cefoxitin disk (30 g) on mueller hinton (mh) agar plate inoculated with test strains as per standard disk diffusion recommendations and incubated at 3335c for 1618 hours .
	20-	an increase in diameter of 5 mm with ceftazidime plus clavulanic acid as compared to ceftazidime disk alone was considered positive for esbl detection .
	36-	no mdr or xdr strain was isolated from streptococcus sp . all (100%) gram positive cocci were sensitive to vancomycin and linezolid .
	38-	e. coli was the commonest isolate 261 (35%) , followed by pseudomonas aeruginosa 212 (28.4%) .
	65-	the limitation of this study is that this is a single center study for only three - month period in a tertiary care hospital in central india . to reflect the trend of infections caused by mdr and xdr strains of bacteria in the region , a multicenter study involving all types of healthcare setups for a minimum period of one year

GOLD		background suicide is a grave public health issue that is responsible for a high mortality rate among individuals aged 1544 years . attitudes toward suicide among medical staff members have been associated with appropriate therapeutic responses to suicidal individuals . the aim of this study was to examine the effects of parental rearing on attitudes toward suicide among japanese medical college students.methodswe examined the association between parental bonding and attitudes toward suicide in 160 medical college students in japan . the parental bonding instrument was used to assess the attitudes and behaviors of parents . the attitudes toward suicide were evaluated using the japanese version of the attitudes toward suicide questionnaire.resultsthe mean age of the subjects was 25.24.0 years old . the majority of the participants in our study agreed that anyone could commit suicide (88.8%) and that suicide is preventable (86.3%) . after adjusting for age and sex , multivariate regression analysis revealed that maternal care approached a statistically significant association with the right to suicide attitude . under the same conditions , maternal care was shown to be significantly associated with the common occurrence attitude . no other significant relationships were observed between parental bonding and attitudes toward suicide.conclusionthis study suggests that a higher level of maternal care ensures that children think that suicide occurs less commonly . the promotion of best practices for suicide prevention among medical students is needed . child rearing support might be associated with suicide prevention .
GBT-EXTSUM	3-	previous studies have shown that difficulties with parental bonding during childhood could be a predisposing factor for the onset of many psychiatric conditions , such as anxiety , depressive states , and maladjusted behaviors.68 parental bonding and premorbid personality traits play an important role in shaping the developmental trajectory of an individual , including his / her ability to adjust to stressful events .
	5-	the objective of this study was to investigate whether parental bonding is associated with attitudes toward suicide among medical college students in japan .
	8-	the demographic data (age and sex) were obtained from self - questionnaires and interviews .
	14-	higher scores on the care and protection dimensions reveal that participants perceive their parents to be more caring and/or protective .
	39-	right to suicide was significantly associated with common occurrence , unjustified behavior , and preventability / readiness to help .
	43-	the majority of the participants in our study agreed that anyone could commit suicide (88.8%) and that suicide is preventable (86.3%) .
	44-	in addition , the multiple regression analysis revealed that participants who reported a higher level of maternal care thought that suicide was a common occurrence and tended to think that people do not have the right to commit suicide .
BERTSUMEXT (SW)	6-	students in their fifth year of medical school at hirosaki university , hirosaki , japan , participated in the study .
	7-	the surveys were distributed to 226 medical students . of the distributed 226 surveys , 160 questionnaires (116 males and 44 females)
	13-	the overprotection dimension of the pbi reflects parental overprotection and control in contrast to the encouragement of autonomy .
	14-	higher scores on the care and protection dimensions reveal that participants perceive their parents to be more caring and/or protective .
	15-	we employed the japanese version of the attitudes toward suicide questionnaire (atts) to assess the attitudes toward suicide held by the study participants.12 we employed a six factor model that was previously developed in studies of japanese attitudes , including
	16-	common occurrence , suicidal expression as mere threat , unjustified behavior ,
	17-	impulsiveness.12,13 each item , with the exception of items 10 and 28 , was scored on a five point scale from 1 (strongly agree) to 5 (strongly disagree) .

D ArXiv Summaries

Gold		in vivo calcium imaging through microscopes has enabled deep brain imaging of previously inaccessible neuronal populations within the brains of freely moving subjects . however , microendoscopic data suffer from high levels of background fluorescence as well as an increased potential for overlapping neuronal signals . previous methods fail in identifying neurons and demixing their temporal activity because the cellular signals are often submerged in the large fluctuating background . here we develop an efficient method to extract cellular signals with minimal influence from the background . we model the background with two realistic components : (1) one models the constant baseline and slow trends of each pixel , and (2) the other models the fast fluctuations from out - of - focus signals and is therefore constrained to have low spatial - frequency structure . this decomposition avoids cellular signals being absorbed into the background term . after subtracting the background approximated with this model , we use constrained nonnegative matrix factorization (cnmf , @xcite) to better demix neural signals and get their denoised and deconvolved temporal activity . we validate our method on simulated and experimental data , where it shows fast , reliable , and high quality signal extraction under a wide variety of imaging parameters .
GBT-ExtSum	1-	. continued advances in optical imaging technology are greatly expanding the number and depth of neuronal populations that can be visualized .
	2-	specifically , in vivo calcium imaging through microendoscopic lenses and the development of miniaturized microscopes have enabled deep brain imaging of previously inaccessible neuronal populations of freely moving mice (@xcite) . while these techniques have been widely used by neuroscientists ,
	20-	like the proposed cnmf in @xcite , our extended cnmf for microendoscopic data (cnmf - e) also has the capability of identifying neurons with low signal - to - noise ratio (snr) and simultaneously denoising , deconvolving and demixing large - scale microendoscopic data . to accomplish this : (1) we replace the rank-1 nmf approximation of the background with a more sophisticated approximation , which can better account the complex background and avoid absorbing cellular signals , and (2) we develop an efficient initialization procedure to extract neural activities with minimal influence from the background .
	71-	@xmath56 is a template matching filter to detect spatial structures with similar shapes and sizes . for flat structures in the small regions , like background , filtering them with @xmath56
	134-	in this paper , we proposed an efficient method for extracting cellular signals from microendoscopic data ; such methods are in very high demand in the neuroscience community .
	136-	our method shows credible performances in recovering the real neuronal signals and outperforms the previous standard pca - ica method .
BertSumExt (SW)	0-	monitoring the activity of large - scale neuronal ensembles during complex behavioral states is fundamental to neuroscience research
	11-	our work is based on a matrix factorization approach , which can simultaneously segment cells and estimate changes in fluorescence in the temporal domain .
	26-	the video data we have are observations from the optical field for a total number of @xmath2 frames .
	64-	we estimate the temporal component of one neuron @xmath15 from spatially filtered data and then use it to extract the corresponding spatial footprint @xmath14 from the raw data . in the step of estimating @xmath14 , we re - order all frames to make nearby frames share the similar local background levels and then take the temporal differencing to remove the background signals temporally .
	105-	we also display @xmath98 tightly clustered neurons in the simulated data (figure [fig : sim]e) to demonstrate that our cnmf - e approach can accurately detect and demix their activity (figure [fig : sim]g) .
	107-	in contrast , pca - ica based detection can only detect two neurons and the calcium traces have high level of noise .

GOLD		statistical learning theory chiefly studies restricted hypothesis classes , particularly those with finite vapnik - chervonenkis (vc) dimension . the fundamental quantity of interest is the sample complexity : the number of samples required to learn to a specified level of accuracy . here we consider learning over the set of all computable labeling functions . since the vc - dimension is infinite and a priori (uniform) bounds on the number of samples are impossible , we let the learning algorithm decide when it has seen sufficient samples to have learned . we first show that learning in this setting is indeed possible , and develop a learning algorithm . we then show , however , that bounding sample complexity independently of the distribution is impossible . notably , this impossibility is entirely due to the requirement that the learning algorithm be computable , and not due to the statistical nature of the problem .
GBT-EXTSUM	6-	an alternative approach , and one we follow in this paper , is simply to consider a single learning model that includes all possible classification methods .
	8-	since the vc - dimension is clearly infinite , there are no uniform bounds (independent of the distribution and the target concept) on the number of samples needed to learn accurately @xcite .
	10-	, it is natural to allow the learning algorithm to decide when it has seen sufficiently many labeled samples based on the training samples seen up to now and their labels . since the above learning model includes any practical classification scheme , we term it universal (pac-) learning .
	11-	we first show that there is a computable learning algorithm in our universal setting .
	19-	our results imply that computable learning algorithms in the universal setting must waste samples " in the sense of requiring more samples than is necessary for statistical reasons alone .
	81-	then we will contrast this to the case of an uncomputable learning algorithm .
BERTSUMEXT (SW)	50-	(semantic requirements) for any @xmath27 , for any concept @xmath8 , and distribution @xmath9 over @xmath2 , if the oracle returns pairs @xmath28 for @xmath29 drawn iid from @xmath9 , then @xmath0 always halts , and with probability at least @xmath12 outputs a hypothesis @xmath13 such that @xmath30 ¡ { varepsilon}$]
	64-	suppose @xmath36 is an infinite sequence of iid samples drawn from @xmath9 .
	75-	the learning algorithm queries the oracle as necessary for new learning samples and their labeling .
	78-	note that it seems necessary to expand the hypothesis space to include all partial recursive functions because the concept space of total recursive functions does not have a recursive enumeration (it is uncomputable whether a given program is total recursive or not) .
	79-	we will see in theorem [thm : nobound] that there is no bound @xmath55 on the number of samples queried by any computable learning algorithm in our setting .
	80-	let us obtain some intuition for why that is true for the above learning algorithm .

GOLD		in this paper , we propose majority voting neural networks for sparse signal recovery in binary compressed sensing . the majority voting neural network is composed of several independently trained feedforward neural networks employing the sigmoid function as an activation function . our empirical study shows that a choice of a loss function used in training processes for the network is of prime importance . we found a loss function suitable for sparse signal recovery , which includes a cross entropy - like term and an @xmath0 regularized term . from the experimental results , we observed that the majority voting neural network achieves excellent recovery performance , which is approaching the optimal performance as the number of component nets grows . the simple architecture of the majority voting neural networks would be beneficial for both software and hardware implementations .
GBT-EXTSUM	40-	requires only several matrix - vector products to obtain an output signal , which is an estimate signal of the sparse vector @xmath12 .
	48-	the signal propagates from left to right and the output signal @xmath17 eventually comes out from the output layer . the network should be trained so that the output signal @xmath17 is an accurate estimation of the original sparse signal @xmath12 .
	168-	in this paper , we proposed sparse signal recovery schemes based on neural networks for binary compressed sensing .
	169-	our empirical study shows a choice of the loss function used for training neural networks is of prime importance to achieve excellent reconstruction performance .
	170-	we found a loss function suitable for this purpose , which includes a cross entropy like term and an @xmath0 regularized term .
	173-	the simple architecture of the majority voting neural network would be beneficial for both software and hardware implementation .
BERTSUMEXT (SW)	19-	the paper @xcite presents binary iterative hard thresholding (biht) algorithm by reforming iterative hard thresholding (iht) algorithm @xcite .
	20-	although the known sparse recovery algorithms exhibit reasonable sparse recovery performance , it may not be suitable for applications in high speed wireless communications .
	48-	the signal propagates from left to right and the output signal @xmath17 eventually comes out from the output layer . the network should be trained so that the output signal @xmath17 is an accurate estimation of the original sparse signal @xmath12 .
	137-	the outputs from these neural network are combined by soft majority voting nodes and the final estimation vector is obtained by rounding the output from the soft majority voting nodes . combining a several neural networks to obtain improved performance is not a novel idea , e.g. , @xcite , but it will be shown that the idea is very effective for our purpose . from statistics of reconstruction errors occurred in our computer experiments , we observed that many reconstruction error events (i.e. , @xmath97) occur due to only one symbol mismatch .
	149-	note that implementation of neural networks with fpga is recently becoming a hot research topic @xcite .
	151-	the length of the sparse signal is set to @xmath59 and the sparseness parameter is set to @xmath110 .) , width=317] from fig.[fig : rr_and_m_k6] , we can observe significant improvement in recovery performance compared with the performance of the single neural network . a single feedforward neural network discussed in the previous section

Through the Looking Glass:
Learning to Attribute Synthetic Text Generated by Language Models

Shaoor Munir[†] Brishna Batool[†] Zubair Shafiq[‡] Padmini Srinivasan[*] Fareed Zaffar[†]

[†]Lahore University of Management Sciences, Pakistan
shaoor.munir@lums.edu.pk
[‡]University of California at Davis, USA
zubair@ucdavis.edu
[*]University of Iowa, USA
padmini-srinivasan@uiowa.edu

Abstract

Given the potential misuse of recent advances in synthetic text generation by language models (LMs), it is important to have the capacity to attribute authorship of synthetic text. While stylometric organic (i.e., human written) authorship attribution has been quite successful, it is unclear whether similar approaches can be used to attribute a synthetic text to its source LM. We address this question with the key insight that synthetic texts carry subtle distinguishing marks inherited from their source LM and that these marks can be leveraged by machine learning (ML) algorithms for attribution. We propose and test several ML-based attribution methods. Our best attributor built using a fine-tuned version of XLNet (XLNet-FT) consistently achieves excellent accuracy scores (91% to near perfect 98%) in terms of attributing the parent pre-trained LM behind a synthetic text. Our experiments show promising results across a range of experiments where the synthetic text may be generated using pre-trained LMs, fine-tuned LMs, or by varying text generation parameters.

1 Introduction

Recent advancements in natural language processing have enabled *synthetic* text generation that is often of comparable quality to the organic text (Ippolito et al., 2020; Radford et al., 2019; Zellers et al., 2019; Gehrmann et al., 2019). This capability has the potential to be misused by malicious actors to launch misinformation, spam, and phishing campaigns (Solaiman et al., 2019; Brown et al., 2020). To prevent potential misuse, prior research has shown considerable success in building machine learning (ML) algorithms that detect (Zellers et al., 2019) or assist humans in detecting (Gehrmann et al., 2019) synthetic text.

While prior research has shown promise in distinguishing between synthetic and organic text, very little has been done on attributing the authorship of the language model (LM) generating the synthetic text (Pan et al., 2020). It is important to be able to track the provenance of synthetic text to the source LM. This can be useful in identifying perpetrators of potential misuse and the unauthorized use of an LM (e.g., in case it is stolen through sophisticated model inversion attacks (Fredrikson et al., 2015) or outright security breaches).

It is particularly challenging to attribute the authorship of the synthetic texts because of the variety and number of available LMs and configurations. While there are only a handful of public pre-trained LMs, it is common to further fine-tune them before using them to generate synthetic text (Devlin et al., 2019; Sanh et al., 2020). Fine-tuning can significantly impact the characteristics of the generated text (Howard and Ruder, 2018; Cruz and Cheng, 2019). Moreover, variations in the sampling parameters used while generating synthetic text whether from pre-trained or fine-tuned LMs can further impact text characteristics (Zellers et al., 2019).

In this paper, we design and evaluate ML-based techniques for attributing the LM and configuration used to generate a synthetic text. We do this in the context of four problem scenarios, each representing a variation of a threat posed by an adversary or malicious user. The scenarios vary in terms of what information the LM attribution system has about the adversary's strategy for generating fake text.

Methodologically, our key insight for attributing the LM used by the adversary is that differences between LM architecture (i.e., layers, parameters), training (i.e., pre-training and fine-tuning), and generation techniques (i.e., sampling parameters) will leave their subtle mark on the generated synthetic texts. The success of our attributors at identifying the LM and configuration used relies on the presence of these subtle distinguishing marks and on the ability to exploit them effectively. As our re-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1811–1822
April 19 - 23, 2021.

sults indicate, this success holds especially in terms of attributing pre-trained models used to generate text even under varying conditions.

In summary, our key contributions are:

- We evaluate a variety of attribution techniques on their ability to attribute the LM and configuration used to generate text. These include attributors making use of stylometric features as well as static and dynamic embeddings.
- We evaluate these attributors on a corpus of 350,000 synthetic texts that we generated in a controlled manner using combinations of LMs, sampling parameters, and fine-tuning.
- Our best attributor built on top of a fine-tuned version of XLNet (XLNet-FT) performs excellently at identifying pre-trained LM used to generate coherent synthetic texts. Accuracy ranges between 91% and close to perfect 98%. This performance holds for various experiments where we use fine-tuning and different sampling parameters. However, the performance is mediocre when attributing the fine-tuned LM used to generate the text.

Paper Organization: The rest of the paper is organized as follows. Section 2 presents the different threat models based on the adversary's strategy for generating synthetic text and assumptions made by the attributor. We then describe our data and attribution methods in Section 3. Experimental results are in Section 4. Section 5 contextualizes our work with respect to prior literature. Section 6 concludes the paper with an outlook on future work.

2 Threat Model

This section describes different threat models that we consider in this paper. The adversary's goal is to generate synthetic text using language models (LMs). The attributor's goal is to attribute the synthetic text to the source LM used by the adversary. All of the threat models operate under the closed world scenario, where the attributor is assumed to know the universe of LMs. The threat models differ based on the adversary's LM training (i.e., pre-training or fine-tuning) and sampling strategies.

2.1 Attributing pre-trained LMs

In the first scenario, the adversary uses a pre-trained LM to generate synthetic text. The attributor trains a classifier to attribute the synthetic text to the source pre-trained LM. We assume a closed-world scenario where both the adversary and attributor have access to the set of off-the-shelf pre-trained LMs such as GPT-2.[1]

More formally, the scenario can be described as: *Given* n *pre-trained LMs* $PM_1, PM_2, ..., PM_n$*, the goal is to train a* n*-class attributor to attribute test instances to the correct source pre-trained LM. In this scenario, the adversary generates texts using* PM_k *where* $1 \leq k \leq n$ *and the attributor's goal is to predict label* PM_k *for the generated texts.*

2.2 Attributing fine-tuned LMs to parent pre-trained LMs

In this scenario, the adversary fine-tunes a pre-trained LM to generate synthetic text. The attributor trains a classifier to attribute the synthetic text to the source pre-trained LM. The main difference from the first scenario is that the attributor is unaware of the fine-tuning used by the adversary before generating text. Note that the goal of the attributor is to detect the source pre-trained LM rather than the fine-tuned LM that is used to generate synthetic text.

More formally, the scenario can be described as: *Given* n *pre-trained LMs* $PM_1, PM_2, ..., PM_n$*, and a LM* FM_k*, generated by fine-tuning* PM_K *where* $1 \leq k \leq n$*, the goal is to train a* n*-class attributor to attribute test instances to the correct source pre-trained LM. In this scenario, the adversary generates text using fine-tuned LM* FM_k *and the attributor's goal is to predict label* PM_k *for generated text.*

2.3 Attributing pre-trained or fine-tuned LMs with different sampling parameters

In this scenario, the attributor trains a classifier to attribute the synthetic text generated by the adversary using a pre-trained or fine-tuned LM. The main difference from the first scenario is that the adversary potentially uses different sampling parameters for text generation than those used by the attributor to train the classifier.

More formally, the scenario can be described as: *Given* n *pre-trained or fine-tuned LMs* $M_1, M_2, ..., M_n$*, the goal is to train a* n*-class attributor to attribute test instances to the correct source model. As per this scenario the adversary*

---

[1] This assumption holds for the rest of the paper, unless stated otherwise.

generates texts using model M_k, $1 \leq k \leq n$, with sampling parameters S_k that are unknown to the attributor, and the attributor's goal is to predict label M_k for the generated text.

2.4 Attributing fine-tuned variants of a pre-trained LM

In this scenario, the adversary fine-tunes a pre-trained LM to generate synthetic text. The attributor trains a classifier to attribute the synthetic text to the source fine-tuned LM. The main difference as compared to the second scenario is that the attributor is aware of the fine-tuning used by the adversary. Note that there are multiple fine-tuned variants of the same parent pre-trained LM.

More formally, the scenario can be described as: *Given n fine-tuned LMs $FM_1, FM_2, ..., FM_n$, the goal is to train a n-class attributor to attribute test instances to the correct fine-tuned LM. As per this scenario, the adversary generates text using a fine-tuned LM FM_k and the attributor's goal is to predict label FM_k for the generated text.*

3 Data & Methods

In this section, we present details about (1) the text generating language models (LMs) and their configurations, and (2) about the attributors studied. To address our research goals, we need a dataset of synthetic texts generated by various pre-trained and fine-tuned LMs under different configurations. Publicly available datasets are unsuitable because there can be high variability in the conditions under which text was generated [2]. It is crucial for us to be able to control the underlying conditions such as: the architecture of LM, prompt used for text generation, sampling parameters, and the data size and topics used for fine-tuning. Details about this generated dataset are also provided in this section.

3.1 Text Generation: LMs, parameters, and configurations

We used four pre-trained LMs: OpenAI GPT (Radford et al., 2018), OpenAI GPT2 (Radford et al., 2019), XLNet (Yang et al., 2019), and BART (Lewis et al., 2020). BART and XLNet are both based on the BERT architecture, which makes use of the bidirectional context of input text to develop a deep understanding of language. XLNet improves on BERT with a form of generalized auto-regressive pre-training using permutation modeling. It outperforms BERT on several classification tasks (Yang et al., 2019). BART combines the bidirectional encoder used by BERT with an auto-regressive decoder used by GPT, which, through a noising and text reconstruction pre-training task, achieves good performance in both language understanding and language generation tasks. In other words, both BART and XLNet augment their training strategies to make up for the lack of language generation capabilities in BERT. GPT and GPT2 are architecturally identical LMs with GPT2 trained on 10 times the data used for training original GPT LM. These use a more traditional generative pre-training approach, looking only at the context coming before a part of the text and not after (Radford et al., 2019). All four pre-trained LMs are publicly available.

3.1.1 Text generation parameters

Three key parameters when generating texts are: *p*, *k*, and *temperature*. The range of values tested are given in Table 1, with default values emphasized in boldface. Note that one chooses to use either *p-value* or *k-value* sampling since they have the same goal - controlling the number of words taken into consideration while sampling text from an LM.

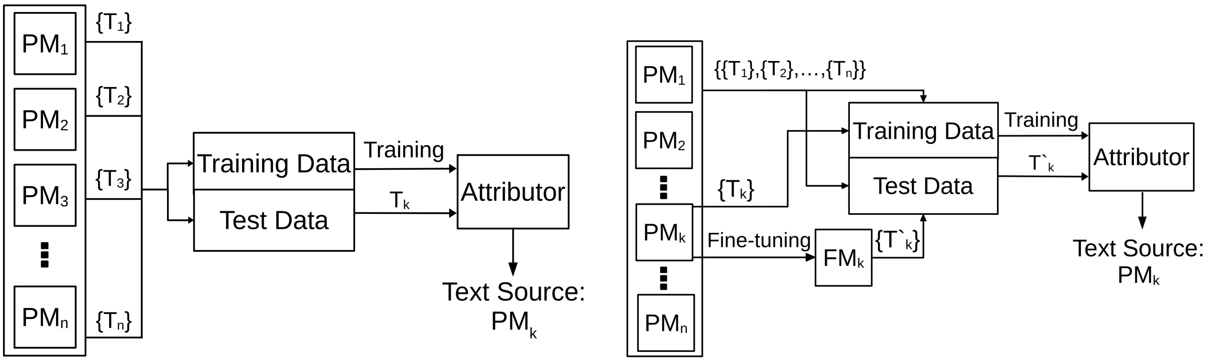
With *top-k* sampling, the LM randomly chooses one from the top *k* words. With *top-p* sampling, it chooses from the set of words whose cumulative probability exceeds *p*. Both Zellers et al. (2019) and Holtzman et al. (2020) conclude that synthetic text matches organic text closely when the *p-value* is kept in range [0.9, 1.0]. Higher values lead to repetitions as the length of text increases. Thus, we choose the lower limit of *p* from the range [0.9, 1.0].

For *top-k* sampling, we use a range of values both higher and lower than 40, which is used as the default for text generation by Radford et al. (2019) in their breakthrough GPT2 paper. Between a choice of *top-p* or *top-k* sampling, we chose *top-p* (p = 0.9) as default due to its lower dependency on vocabulary size and extensive use in previous research on GPT2 (Radford et al., 2019; Zellers et al., 2019; Ippolito et al., 2020).

Temperature controls the likelihood of low probability words appearing in the final pool of words used for random selection (Holtzman et al., 2020). Higher temperatures produce text containing highly unusual words that are normally not favored by *top-k* or *top-p* sampling. At the other end, Holtzman et al. (2020) note that temperatures below 1 reduce

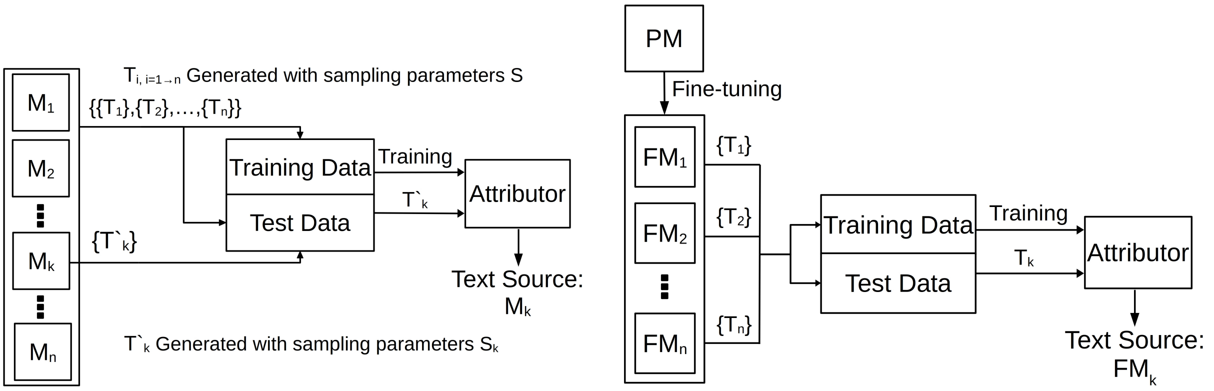
---

[2] https://www.kaggle.com/abhishek/gpt2-output-data



(a) Scenario 1: Attributing pre-trained LM

(b) Scenario 2: Attributing fine-tuned LMs to the parent pre-trained LMs



(c) Scenario 3: Attributing LM with different sampling parameters

(d) Scenario 4: Attributing fine-tuned variants of a pre-trained LM

Figure 1: Illustration of different threat models studied in this work

word diversity but at the cost of increasing word repetition. To avoid this we set temperature as 1 in experiments where evaluation of its effect on synthetic text is not of concern.

#	**Parameter**	**Values**
1	Architecture	GPT, **GPT2**, XLNet, BART
2	Text Length	Short, **Medium**, Long
3	Fine-tuning Topic	**r/changemyview**, r/technology, r/relationships, r/conspiracy
4	p-value	**0.9**, 0.92, 0.94, 0.96, 0.98, 1.0
5	Temperature	0.1, 0.5, **1**, 1.5, 2
6	k-value	1, 20, 40, 80, 160

Table 1: The parameters explored (defaults in bold)

3.1.2 Data for fine-tuning

For scenarios where we need synthetic text generated using fine-tuned LMs, we limit text generation to the GPT2 LM. GPT2 has been shown to have state of the art performance in language generation tasks (Radford et al., 2019; Klein and Nabi, 2019). Data from four Reddit communities was used for fine-tuning LMs: *r/relationships*, *r/technology*, *r/changemyview*, and *r/conspiracy*. These subreddits were chosen based on qualitative differences between their content. r/technology contains technical jargon, while r/relationships focuses on personal pronouns and adopts a critical approach towards writing. r/changemyview has confrontational content with members attempting to challenge and disprove each other's views, while r/conspiracy focuses on hyperbolic statements. In essence, each subreddit is considered a different *topic* area. Table 2 shows the number of posts and comments scraped from each subreddit.

3.1.3 Dataset details

We generate text of three different lengths: *short* (up to 40 words), *medium* (between 40 and 100 words), and *long* (above 100 words). In experiments where length is not the focus, we use *medium* as the default. Each synthetic text is generated using a randomly selected subreddit submission as a prompt. We start by sampling words equal to the length of the prompt from the LM. We trim the generated text to follow standard sentence structure such as start capitalization and end punctuation, after which text is sorted into one of three length

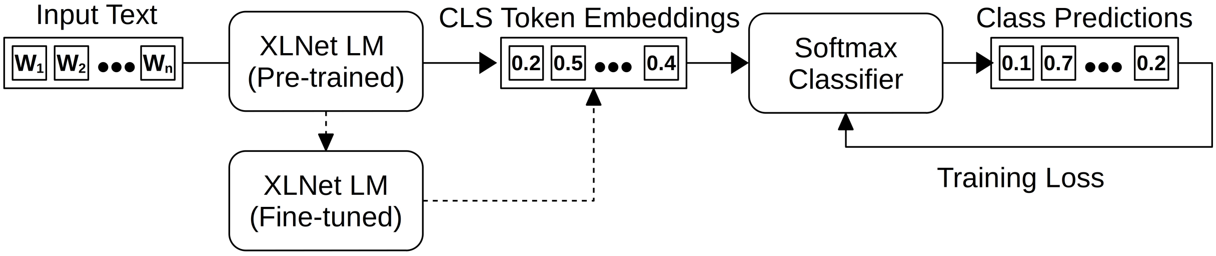


Figure 2: Attributor training on XLNet embeddings. The dashed lines are part of the fine-tuned pathway.

categories.

We generated 10,000 synthetic texts for each target class in our experiments. For example, when evaluating the performance of attributors against fine-tuned LMs, we generated 10,000 samples for each of four GPT2 LMs fine-tuned on one of the four Reddit topics mentioned previously. In total, 35 distinct sets of synthetic documents, each with 10,000 examples, were used for a total of 350,000 unique synthetic documents[3]. We build training and test datasets that are balanced in classes for each scenario because while there is growing evidence that synthetic text is appearing in the wild, there is little to no information about the relative impact of the source LMs. Thus, any split beyond an even split across classes has little justification.

Subreddit	Posts	Comments	Total
r/changemyview	136,775	321,527	458,302
r/relationships	200,047	167,219	367,266
r/technology	174,431	143,199	317,630
r/conspiracy	99,302	161,993	261,295

Table 2: Data scraped per each subreddit.

3.2 Attributors

We test six attributors in their ability to identify the source LMs. The first attributor is a decision tree classifier with Writeprints (Abbasi and Chen, 2008) feature set, second is a CNN with GloVe (Pennington et al., 2014) embeddings as the feature set. The next four attributors are softmax classifiers, with the first two built on top of pre-trained XLNet and GPT2, and the other two on top of XLNet and GPT2 fine-tuned on training data used in the corresponding scenario.

3.2.1 Decision tree with Writeprints features

The Writeprints features have been used extensively and successfully for authorship attribution. (Abbasi and Chen, 2008; Mahmood et al., 2020) When combined with SVMs and decision trees these have shown good performance in attribution tasks (Abbasi and Chen, 2008; Pearl and Steyvers, 2012). Due to ease of interpretability of features, we implement a decision tree classifier with Writeprints features to test our intuition that stylistic, rather than topical, differences contribute towards the attribution of synthetic text.

3.2.2 CNN with GloVe embeddings

Pre-trained GloVe embeddings have been shown to outperform word frequency and count-based embeddings for sentence and sequence classification tasks (Pennington et al., 2014; Le-Hong and Le, 2018). Also, the use of GloVe with CNNs has shown good results in classification tasks like newsgroup identification (Gupta et al., 2018).

3.2.3 Attributors from LM embeddings

Embeddings generated through LMs like BERT, XLNet, and GPT2 have been shown to capture language semantics and context much better than static embeddings generated through GloVe and other word count or frequency-based embeddings generators (Sun et al., 2020; Howard and Ruder, 2018). Because of their extensive pre-training, these LMs can capture long-term dependencies and incorporate contextual and hierarchical relations between words better than pre-computed static embeddings.

LMs such as XLNet make use of a special *[CLS]* token to get pooled output representing a complete text sequence. We use the final network layer embeddings of this token for attribution. Specifically, we train a softmax output layer that takes as input XLNet's *[CLS]* token embeddings and generates probabilities for each decision class in the experiment setup. For GPT2 we use a parallel strategy with pooled output from the complete final layer of the model for a particular input text. Again this output is connected to a softmax output layer which, similar to our strategy with XLNet, is trained to

---

[3] We will make this dataset available for research upon publication of our paper

generate class predictions based on the input embeddings.

In addition to using the pre-trained versions of XLNet and GPT2, we also evaluate attributors built from fine-tuned versions of these LMs. Note that here fine-tuning is on training data used to train all attributors in the corresponding experiment. Figure 2 illustrates these strategies with XLNet as an example. The sequence with dashed lines represents the fine-tuned versions.

4 Results

We present attribution accuracy results in the same order of scenarios described earlier in Section 2.[4]

4.1 Attributing pre-trained LMs

Table 3 presents the accuracy results for short (up to 40 words), medium (between 40 and 100 words), and long (more than 100 words) synthetic text generated using pre-trained GPT, GPT2, XLNet, and BART language models (LMs). Decision tree and the two XLNet versions achieve accuracy between 82 and, near perfect 98%, across the three types of texts. In comparison, CNN and GPT2 attributors lag behind.

While both XLNet attributors score higher than decision tree, XLNet-FT has the best performance which when compared with the next best XLNet-PT ranges from 3% to 7%. Note that apart from the pre-trained GPT2 attributor, all show marked improvement in accuracy scores with an increase in text length. Similar results showing direct proportionality of classifier performance with text length were also observed by Ippolito et al. (2020) in experiments detecting synthetic text.

Prior work has shown that uni-directional LMs are more suited for language generation due to generative pre-training (Lewis et al., 2020) where the LM learns to predict the next word based on the previous context. Bidirectional LMs like BERT and XLNet excel at classification as they make use of masked modeling and next sentence prediction tasks to improve understanding of necessary language attributes (Devlin et al., 2019; Yang et al., 2019). Our results are consistent in that XLNet performance is better than GPT2.

Interestingly, the decision tree with Writeprints outperforms GPT2 based attributors in all three text lengths. Our investigation into specific Writeprints features emphasized by the decision tree (**see appendix A.1**) reveals a greater emphasis on stylistic features. This gives further credence to our intuition that variations between texts generated by different LMs are more stylistic than topical in nature. Our results suggest that GPT2 based attributors are not adept at capturing such stylistic differences.

4.2 Attributing fine-tuned LMs to the parent pre-trained LMs

Our goal in this scenario is to attribute the synthetic text generated using a fine-tuned variant (using an unknown dataset) of a pre-trained LM. Note that the attributor is unaware of fine-tuning. We limit fine-tuned text generation in this experiment to just GPT2 for reasons described in Section 3.

The first row in Table 4 reports the accuracy results. We note that XLNet-FT again performs the best with XLNet-PT in the second place. CNN has the weakest results. Interestingly, Writeprints continues to do fairly well – once again emphasizing the role of style in identifying source LM. Comparing these results with Table 3 (for medium length texts), we see all accuracies drop slightly as expected when the adversary chooses to fine-tune the LM that is unknown to the attributor.

We run a second variation of the same experiment – one where the attributor has partial knowledge of the adversary's strategy. Specifically, the fact that the adversary is using a fine-tuned LM to generate text is known but the dataset used for fine-tuning remains unknown. In response, we pick some dataset (here r/relationships) to add fine-tuned LM generated texts to our training data. Note that the adversary uses r/changemyview. The second row in Table 4 reports similar results as the first row. Thus, it seems that this additional knowledge does not help improve attribution accuracy.

In sum, the accuracy for XLNet-FT across all experiments thus far is above 90%. This indicates that even when the adversary fine-tunes the LM for text generation, the parent pre-trained LM is still identifiable. This result confirms our intuition that as fine-tuning is known to leave the majority of layers unchanged, the text generated retains characteristics of the parent pre-trained LM, making accurate attribution possible.

---

[4] We measured performance using F1 score as well. However, since there were no remarkable differences, we only report accuracy results to be concise.

Synthetic Text Length	**DT (Writeprints)**	**CNN (GloVe)**	**XLNet-PT**	**XLNet-FT**	**GPT2-PT**	**GPT2-FT**
Short (Upto 40 Words)	82	68	85	**91**	72	74
Medium (40 to 100 words)	86	73	90	**96**	71	72
Long (Above 100 words)	93	83	95	**98**	72	72

Table 3: Accuracy percentages for attributing source pre-trained LMs. Datasets contain synthetic texts of different lengths generated using pre-trained BERT, GPT, GPT2 and XLNet.

Training data includes	**Test data includes**	**DT (Writeprints)**	**CNN (GloVe)**	**XLNet-PT**	**XLNet-FT**	**GPT2-PT**	**GPT2-FT**
GPT2 (PT)	GPT2 (FT-r/changemyview)	78	64	86	**93**	70	71
GPT2 (FT-r/relationships)	GPT2 (FT-r/changemyview)	77	67	86	**91**	70	70

Table 4: Accuracy percentages in attributing source pre-trained LM when adversary generates synthetic text using a fine-tuned LM. In addition to GPT2 variants mentioned in columns 1 and 2, training and testing data also includes classes representing XLNet, BART, and GPT.

4.3 Attributing LM with different sampling parameters

Here we consider the scenario where both the attributor and the adversary use the same LM but they differ in parameter choices when generating texts. We run this experiment assuming the adversary uses GPT2 fine-tuned on r/changemyview. The attributor is aware of this but not the sampling parameters. Selecting parameter values that are quite different from each other we see from Table 5 that there is virtually no performance drop for XLNet-FT. That is, our best performing attributor is resilient to these differences. Temperature sampling shows weaker results all around. This is not a concern as discussed later in this section.

We next explore the parameter differences angle further to get a sense of what would happen if the adversary chose a parameter value other than the ones explored in Table 5. The different values for k, p, and temperature are as listed in Table 1. We remind the reader that one uses either $top-k$ or $top-p$ sampling to control the number of words under consideration during text generation. We use $top-p$ sampling as the default strategy. When varying k or p, the temperature is fixed at the default value. When varying temperature, p is kept at the default value.

The results in Table 6 show that it is challenging to tell apart synthetic texts generated by different values of k and p. Given their strong similarities we expect to see results as in Table 5 when the adversary picks other parameter values. With temperature variations we get accuracy above 80%, indicating marked differences between texts generated at different temperatures. However, taking a closer look at the text reveals a serious problem: temperature > 1 produces erratic and confusing text. This problem becomes more acute as the temperature approaches its upper limit. This is consistent with the observation by Holtzman et al. (2020) showing that temperatures above 1 produce incoherent and confusing text. This reduces viability in a setting where the synthetic text is to be used as a suitable replacement for organic text.

We conclude from Tables 5 and 6 that our attributors should be resilient even when the adversary chooses parameter values for text generation beyond the ones explicitly tested here.

4.4 Attributing fine-tuned variants of a pre-trained LM

Finally, we explore the scenario where the adversary is using different fine-tuned LMs with the same parent pre-trained LM. The attributor is aware of this fine-tuning and is attempting to tell apart these fine-tuned LMs. Table 7 presents the accuracy results when synthetic text is generated by fine-tuning GPT2 on 4 different subreddits (r/changemyview, r/technology, r/relationships, r/conspiracy). XLNet-FT again achieved the best accuracy, however, this time it is less than 60%. Curiously, CNN which was the least successful in earlier experiments performed almost identically to XLNet-FT. GPT2 performed just slightly better than a random attributor (1/4, i.e., 25%). Overall, variations between texts generated by different fine-tuned variants of the same pre-trained LM are not

Training data includes	Test data includes	DT (Writeprints)	CNN (GloVe)	XLNet-PT	XLNet-FT	GPT2-PT	GPT2-FT
GPT2 (k=20)	GPT2 (k=160)	76	71	88	**95**	71	71
GPT2 (p=0.9)	GPT2 (p=1.0)	80	70	88	**96**	70	70
GPT2 (t=0.1)	GPT2 (t=1.0)	60	67	70	**77**	71	72

Table 5: Accuracy percentages for attributing source LM when adversary generates synthetic text using different sampling parameters. In addition to GPT2 variants mentioned in columns 1 and 2, training and testing data also includes classes representing XLNet, BART, and GPT.

Attributor	k-value	p-value	Temperature
DT (Writeprints)	**53**	**43**	82
CNN (GloVe)	47	26	79
XLNet-PT	42	22	81
XLNet-FT	45	25	**86**
GPT2-PT	28	17	44
GPT2-FT	28	18	44

Table 6: Accuracy percentages for identifying texts generated by GPT2 LM fine-tuned on r/changemyview with varying sampling parameters. Parameter values tested are as reported in Table 1.

Attributor	Accuracy
DT (Writeprints)	52
CNN (GloVe)	56
XLNet-PT	53
XLNet-FT	**57**
GPT2-PT	29
GPT2-FT	29

Table 7: Accuracy percentage in attributing fine-tuned GPT2 LMs. Dataset contains texts generated by four GPT2 LMs fine-tuned on each subreddit in Table 2.

pronounced enough to be leveraged by the attribution techniques we consider. Our preliminary analysis shows that there is some correlation between the attributor's mistakes and the vocabulary similarity of the corresponding subreddits. However, further research is needed to probe the causes of this lackluster performance and devise ways to improve the attribution of text produced by fine-tuned LMs.

5 Related Work

We contextualize our work with respect to prior literature on detection and attribution of organic and synthetic text.

5.1 Synthetic text detection

There has been a lot of recent interest in developing ML approaches to distinguish between organic and synthetic text. GLTR (Giant Language Model Test Room) leveraged the statistical tendency of LMs to produce words with higher probability of occurrence to help users differentiate between synthetic and organic text (Gehrmann et al., 2019). Grover used a purpose-built LM to train a classifier for synthetic and organic text (Zellers et al., 2019). Bakhtin et al. (2019) proposed energy based models for differentiating between synthetic and organic text. Our work takes this line of work a step further by trying to attribute synthetic text to the source LM.

5.2 Synthetic text attribution

Pan et al. (2020) proposed a dynamic embedding based approach to attribute synthetic text generated by a pre-trained LM as part of their broader investigation of sensitive information exposed by LMs. We significantly build on this work from both the methodological and application perspectives. Differently from this work, we use stylometric as well as static and dynamic embeddings. We also consider more realistic threat models where the synthetic text is generated by either pre-trained or fine-tuned LMs and using different sampling parameters.

5.3 Organic text attribution

There is a rich body of literature on authorship attribution of organic text using stylometric features. We discuss a few classic papers here. Mosteller and Wallace (1964) used word frequency analysis for authorship attribution. Abbasi and Chen (2008) proposed a ML-based approach for authorship attribution using an exhaustive stylometric feature set called Writeprints. While there is impressive progress in stylometric organic text attribution (e.g., Narayanan et al., 2012; Ruder et al., 2016), these approaches do not work as effectively for synthetic text attribution. As our evaluation showed, Writeprints were significantly outperformed by other approaches for synthetic text attribution. This is because LMs are trained on large text corpora from different authors thus there are no clear-cut stylometric differences in synthetic text generated by different LMs.

5.4 Synthetic image attribution

Recent advances in Generative Adversarial Networks (GANs) have led to impressive results in synthetic image generation (Bao et al., 2017; Taigman et al., 2017; Ma et al., 2017). For example, Chen et al. (2020) proposed image models similar to pre-trained LMs to learn an unsupervised representation of images for various downstream tasks. Most related to our work, Yu et al. (2019) proposed an ML approach to attribute synthetic images generated by GANs with different architectures and parameters. At the most basic level, the problem of synthetic image attribution differs from synthetic text attribution because images are smooth and local where words in a text document may be correlated even if they are far apart (Sharir et al., 2020). For instance, Yu et al. (2019) showed that their ML classifier could use only part of the synthetic image for attribution. In contrast, we observed a large drop in accuracy when we make use of only part of input synthetic text.

6 Conclusion

In this paper, we presented an ML approach to attribute authorship of synthetic text to its source LM. Our results showed that an attributor based on fine-tuned XLNet embeddings outperformed other approaches based on stylometric features as well as static and dynamic embeddings. Our results also showed there is significant room for improvement in distinguishing between synthetic text generated by different fine-tuned variants of an LM. Further research is also needed for effective attribution of synthetic text generated by more diverse fine-tuned LMs in both closed-world and-open world settings. Finally, future research on synthetic text attribution should also consider more sophisticated LMs (e.g., GPT-3 with 175 billion parameters (Brown et al., 2020) and Google's trillion parameter LM (Fedus et al., 2021)) when they are publicly released.

References

Ahmed Abbasi and Hsinchun Chen. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Trans. Inf. Syst.*, 26(2).

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or Fake? Learning to Discriminate Machine from Human Generated Text. *CoRR*, abs/1906.03351.

J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. 2017. Cvae-gan: Fine-grained image generation through asymmetric training. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2764–2773.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. 2020. Generative pretraining from pixels. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1691–1703. PMLR.

Jan Christian Blaise Cruz and Charibeth Cheng. 2019. Evaluating language model finetuning techniques for low-resource languages. *arXiv preprint arXiv:1907.00409*.

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

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *arXiv 2101.03961*.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, page 1322–1333, New York, NY, USA. Association for Computing Machinery.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical Detection and Visualization of Generated Text. In *57th Annual Meeting of the Association for Computational Linguistics (ACL): Demo Track*.

Varun Gupta, Akhilesh Kumar, and Aashish Bhardwaj. 2018. Newsgroup classification using cnn and glove embeddings. *International Journal of Applied Research on Information Technology and Computing*, 9:135.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Tassilo Klein and Moin Nabi. 2019. Learning to Answer by Learning to Ask: Getting the Best of GPT-2 and BERT Worlds. *arXiv*, 1911.02365.

Phuong Le-Hong and Anh-Cuong Le. 2018. A comparative study of neural network models for sentence classification. In *5th NAFOSTED Conference on Information and Computer Science (NICS)*.

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

Liqian Ma, Xu Jia, Qianru Sun, Bernt Schiele, Tinne Tuytelaars, and Luc Van Gool. 2017. Pose guided person image generation. In *Advances in Neural Information Processing Systems*, volume 30, pages 406–416. Curran Associates, Inc.

Asad Mahmood, Zubair Shafiq, and Padmini Srinivasan. 2020. A Girl Has A Name: Detecting Authorship Obfuscation. In *58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations (ICLR)*.

Frederick Mosteller and David L. Wallace. 1964. *Inference and Disputed Authorship: The Federalist*. Addison-Wesley series in behavioral science. Addison-Wesley.

Arvind Narayanan, Hristo Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. 2012. On the feasibility of internet-scale author identification. In *2012 IEEE Symposium on Security and Privacy*.

X. Pan, M. Zhang, S. Ji, and M. Yang. 2020. Privacy risks of general-purpose language models. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331.

Lisa Pearl and Mark Steyvers. 2012. Detecting authorship deception: A supervised machine learning approach using author writeprints. *Literary and Linguistic Computing*, 27:183–196.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

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

Sebastian Ruder, Parsa Ghaffari, and John G. Breslin. 2016. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *arXiv 1609.06686*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The Cost of Training NLP Models: A Concise Overview. *arXiv 2004.08900*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to Fine-Tune BERT for Text Classification? *arXiv 1905.05583*.

Yaniv Taigman, Adam Polyak, and Lior Wolf. 2017. Unsupervised cross-domain image generation. In *International Conference on Learning Representations (ICLR)*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural*

Information Processing Systems, volume 32, pages 5753–5763. Curran Associates, Inc.

N. Yu, L. Davis, and M. Fritz. 2019. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7555–7565.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32, pages 9054–9065. Curran Associates, Inc.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *IEEE International Conference on Computer Vision (ICCV)*.

A Appendix

A.1 Analysis of importance given by Decision Tree to Writeprints

Synthetic texts generated by pre-trained LMs are distinguishable to a high degree. This holds even when the adversary decides to use a fine-tuned LM or varies text generation parameters like p-value, k-value, or temperature. Thus, these texts carry exploitable model fingerprints.

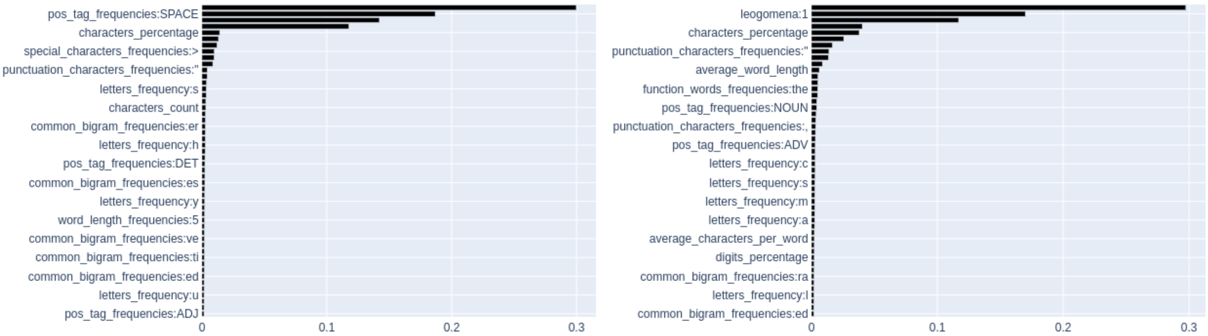
Most challenging is the ability to tell apart texts generated by different fine-tuned versions of the same pre-trained model. XLNet fine-tuned on the training data yields excellent results; except for attributing fine-tuned models accuracies are almost entirely above 90%. Interestingly, decision trees fare quite well offering the advantage of interpretability of decisions. Decision Tree based classifier focusing only on stylistic differences achieves an accuracy of higher than 80% in all three configurations. Investigating the importance given by classifier to different Writeprints show stylistic features like spaces, percentage of characters, and special characters being given the highest importance.

Figure 3 shows a comparison of importance given by a decision tree based attributor to features before and after eliminating whitespace as a feature. Running the experiment again after eliminating the highest rated feature (frequency of white space) results in minimal drop in performance (within a range of 1%- 2%) and shows continued focus on more stylistic language features as key indicators of differences between these texts. This confirms our intuition that different pre-trained versions of language models have different writing styles which are discernible through text classification techniques. Moreover, our experiments showed that fine-tuned LMs retained characteristics from their parent pre-trained LM, allowing an attributor trained entirely on pre-trained text to successfully attribute fine-tuned LM text with above 90% accuracy even in a worst case scenario.

Comparing importance maps from a decision tree attributor trained on Writeprints from pre-trained and fine-tuned GPT2 LMs shows interesting results. From figure 4, it is apparent that top two most important features are common among pre-trained and fine-tuned variants, with a number of other similarities in features given relatively less importance. It shows that there are certain stylistic characteristics that are passed down from a pre-trained LM to its fine-tuned variant.

A.2 Details of pre-trained language models used

In our experiments we have made use of four publicly available pre-trained language models: XLNet, BART, GPT, GPT2. Details about those are given in Table 8. Comparing sizes, XLNet is trained on largest dataset with over 142 GB of documents. Although no explicit size is mentioned for GPT2, it is said to be trained on 10 times more data than GPT.



(a) Before removing whitespace

(b) After removing whitespace

Figure 3: Comparison of feature importance with and without whitespace

(a) Importance given by attributor trained on pre-trained text data to different Writeprints features

(b) Importance given by attributor trained on fine-tuned text data to different Writeprints features

Figure 4: Comparison of feature importance for pre-trained and fine-tuned variants of GPT2 LM

Language Model	**Dataset**	**Words**	**Size**	**Number of Documents**
OpenAI GPT	BooksCorpus (Zhu et al., 2015)	∼985M	Not Available	Not Available
OpenAI GPT2	WebText (Radford et al., 2018)	Not Available	40GB	∼8 million documents
BART	WikiText-103 (Merity et al., 2017)	∼103M Words	181MB	28,475 articles
XLNet	BookCorpus + English Wikipedia + CommonCrawl + Giga5 + ClueWeb 2012-B	∼32.8B subword pieces	142GB	Not Available

Table 8: Breakdown of pre-trained LMs used

We Need to Talk About Random Splits

Anders Søgaard[1,2] **Sebastian Ebert**[1] **Jasmijn Bastings**[1] **Katja Filippova**[1]
[1]: Google Research, Berlin
[2]: University of Copenhagen
{soegaard|eberts|bastings|katjaf}@google.com

Abstract

Gorman and Bedrick (2019) argued for using random splits rather than standard splits in NLP experiments. We argue that random splits, like standard splits, lead to overly optimistic performance estimates. We can also split data in biased or adversarial ways, e.g., training on short sentences and evaluating on long ones. Biased sampling has been used in domain adaptation to simulate real-world drift; this is known as the covariate shift assumption. In NLP, however, even worst-case splits, maximizing bias, often under-estimate the error observed on new samples of in-domain data, i.e., the data that models should minimally generalize to at test time. This invalidates the covariate shift assumption. Instead of using multiple random splits, future benchmarks should ideally include multiple, independent test sets instead; if infeasible, we argue that multiple biased splits leads to more realistic performance estimates than multiple random splits.

1 Introduction

It is common practice in NLP to collect and annotate a text corpus – and split it into training, development and test data. These splits are often based on the order in which texts were published or sampled, and are referred to as 'standard splits'. Gorman and Bedrick (2019) recently showed that system ranking results based on *standard* splits differ from results based on *random* splits and used this to argue in favor of using random splits. While perhaps less common, random splits are already used in probing (Elazar and Goldberg, 2018), interpretability (Pörner et al., 2018), as well as core NLP tasks (Yu et al., 2019; Geva et al., 2019).[1]

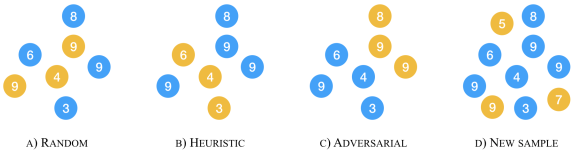


Figure 1: Data splitting strategies. Each ball corresponds to a sentence represented in (two-dimensional) feature space. Blue (dark)/orange (bright) balls represent examples for training/test. Numbers represent sentence length. Heuristic splits can, e.g., be based on sentence length; adversarial splits maximize divergence.

Gorman and Bedrick (2019) focus on whether there is a significant performance difference δ between systems S_1 and S_2; $\mathcal{M}(G_{test}, S_1) - \mathcal{M}(G_{test}, S_2)$, in their notation. They argue McNemar's test (Gillick and Cox, 1989) or bootstrap (Efron, 1981) can establish that $\delta \neq 0$, using random splits to sample from G_{test}. This, of course, relies on the assumption that data is representative, i.e., was sampled i.i.d. (Wolpert, 1996).

In reality, what Gorman and Bedrick (2019) call *the true difference in system performance*, i.e., $\delta = \mathcal{M}(G_{test}, S_1) - \mathcal{M}(G_{test}, S_2)$, is the system difference on *data that users would expect the systems to work well on* (see §2 for practical examples) – and not just on the corpus that we have annotations for. Our corpus-based estimates of δ can in fact be very misleading, i.e., very different from performance on new samples of data. In this paper, we investigate *how* misleading our estimates can be: We show that random splits consistently over-estimate performance at test time. This favors systems that overfit. We investigate alternatives across a heterogeneous set of NLP tasks. Based on our experiments, our answer to community-wide overfitting to standard splits is *not* to use random splits but to collect more diverse data with different biases – or if that is not feasible, split your data in adversarial, not random, ways. In general, we observe that estimates of test time error are worst for random splits, slightly better for standard splits

---

[1] See also many of the tasks in the SemEval evaluation campaigns: `http://alt.qcri.org/semeval2020/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1823–1832
April 19 - 23, 2021.

Task	Benchmark	Source/Domain	New Samples
POS TAGGING	WSJ	News	Xinhua
PROBING	SentEval	Toronto BC	Gutenberg
EMOJIS	S140	Twitter	S140*
QE	WMT 2016	IT	WMT 2018
HEADLINES	Gigaword	News	Gigaword*
NEWS	UCI	News	UCI*

Table 1: Data used in our experiments. *: We time slice the original data to create different samples.

(if those exist), better for heuristic and adversarial splits, but error still tends to be higher on new (in-domain) samples; see Figure 1.

Our results not only refute the hypothesis that δ can be estimated using random splits (Gorman and Bedrick, 2019),[2] but also the covariate shift hypothesis (Shimodaira, 2000; Shah et al., 2020) that δ can be estimated using reweightings of the data. While biased splits are useful in the absence of multiple held-out samples, and have been proposed before (Karimi et al., 2015),[3] they often overestimate performance in the wild. Our code is made publicly available at `https://github.com/google-research/google-research/talk_about_random_splits`.

2 Experiments

We consider 7 different NLP tasks: POS tagging (like Gorman and Bedrick (2019)), two sentence representation probing tasks, headline generation, translation quality estimation, emoji prediction, and news classification. We experiment with these tasks, because they a) are diverse, b) have not been subject to decades of community-wide overfitting (with the exception of POS tagging), and c) three of them enabled temporal splits (see Appendix §A.5).

Data splits The datasets which we will use in our experiments are presented in Table 1. For all seven tasks, we will present results for standard splits when possible (POS, PROBING,QE, HEADLINES), random splits, heuristic and adversarial splits, as well as on new samples. In the case of EMOJIS, HEADLINES and NEWS, which are all time-stamped datasets, we leave out historically more recent data as our new samples. All **new samples** are in-domain samples of data where models are supposed to generalize, i.e, samples from similar text sources.[4] This is a key point: These are *samples that any end user would expect decent NLP models to fair well on*. Examples include a sample of newspaper articles from newspaper A for a POS tagger trained on articles from newspaper B; tweets sampled the day after the training data was sampled; or news headlines sampled from the same sources, but a year later.

We resample **random** splits multiple times (3-10 per task) and report average results. The **heuristic** splits are obtained by finding a sentence length threshold and putting the long sentences in the test split. We choose a threshold so that approximately 10% of the data ends up in this split. The idea of checking whether models generalize to longer sentences is not new; on the contrary, this goes back, at least, to early formal studies of recurrent neural networks, e.g., Siegelmann and Sontag (1992). In the §A.3, we present a few experiments with alternative heuristic splits, but in our main experiments we limit ourselves to splits based on sentence length.

Finally, the **adversarial** splits are computed by approximately maximizing the Wasserstein distance between the splits. The Wasserstein distance is often used to measure divergence between distributions (Arjovsky et al., 2017; Tolstikhin et al., 2018; Shen et al., 2018; Shah et al., 2018), and while alternatives exist (Ben-David et al., 2006; Borgwardt et al., 2006), it is easy to compute and parameter-free. Since selecting *the* worst-case split is an NP-hard problem (e.g., by reduction of the knapsack problem), we have to rely on an approximation. We first compute a ball tree encoding the Wasserstein distances between the data points in our sample. We then *randomly* select a centroid for our test split and find its k nearest neighbors. Those k nearest neighbors constitute our test split; the rest is used to train and validate our model. We repeat these steps to estimate performance on worst-case splits of our sample. See §A.4 for an algorithm sketch. Random, heuristic, and adversarial results are averaged across five runs.

---

[2] Or cross-validation, as more recently proposed in Szymański and Gorman (2020). In this very interesting follow-up paper, about Bayesian inference of δ, the authors write that their "estimates are valid insofar as the data sets used to estimate the Bayesian models comprise a representative sample of a coherent population of data sets." Our results show how *off* this assumption is.

[3] Karimi et al. (2015) discuss temporal splits and splits based on neighbor-based heuristics that are similar in spirit to our worst-case splits.

[4] Domains are commonly defined as collections of similar text sources (Harbusch et al., 2003; Koehn and Knowles, 2017). In addition to using similar sources, we control for low $\mathcal{A}$-distance (Ben-David et al., 2006) by looking at separability; e.g., a simple linear classifier over frequent unigrams can distinguish between Penn Treebank development and test sections with an accuracy of 64%; and between the development and our new sample with an accuracy of 69%.

POS tagging We first consider the task in Gorman and Bedrick (2019), experiment with heuristic and adversarial splits of the original Penn Treebank (Marcus et al., 1993), and add the Xinhua section of OntoNotes 5.0[5] as our **New Sample**. Our tagger is NCRF^{++} with default parameters.[6]

Probing We also include two SentEval probing tasks (Conneau et al., 2018) with data from the Toronto Book Corpus: PROBING-WC (word classification) and PROBING-BSHIFT (whether a bigram was swapped) (Conneau et al., 2018). Unlike the other probing tasks, these two tasks do not rely on external syntactic parsers, which would otherwise introduce a new type of bias that we would have to take into account in our analysis. We use the official SentEval framework[7] and BERT (Devlin et al., 2019) as our sentence encoder. The probing model is a logistic regression classifier with L_2 regularization, tuned on the development set. As our **New Samples**, we use five random samples of the 2018 Gutenberg Corpus[8] for each task, preprocessed in the same way as Conneau et al. (2018).

Quality estimation We use the WMT 2014 shared task datasets for QUALITY ESTIMATION. Specifically, we use the Spanish-English data from Task 1.1: scoring for perceived post-editing effort. The dataset comes with a training and test set, and a second, unofficial test set, which we use as our **New Sample**. In the §A.2, we also present results training on Spanish-English and evaluating on German-English. We present a simple model that only considers the target sentence, but performs better than the best shared task systems: we train an MLP over a LASER sentence embedding (Schwenk et al., 2019) with the following hyper-parameters: two hidden layers with 100 parameters each and ReLU activation functions, trained using the Adam stochastic gradient-based optimizer (Kingma and Ba, 2015), a batch size of 200, and L_2 penalty of strength $\alpha = 0.01$.

Headline generation We use the standard dataset for headline generation, derived from the Gigaword corpus (Napoles et al., 2012), as published by Rush et al. (2015). The task is to generate a headline from the first sentence of a news article. Our architecture is a sequence-to-sequence model with stacked bi-directional LSTMs with dropout, attention (Luong et al., 2015) and beam decoding; the number of hidden units is 128; we do not pre-train. Different from Rush et al. (2015), we use subword units (Sennrich et al., 2016) to overcome the OOV problem and speed up training. The ROUGE scores we obtain on the standard splits are higher than those reported by Rush et al. (2015) and comparable to those of Nallapati et al. (2016), e.g., ROUGE-1 of 0.321. As our **New sample**, we reserve 20,000 sentence-headline pairs each from the first and second halves of 2004 for validation and testing; years 1998-2003 are used for training. For all the experiments we report the error reduction in ROUGE-2 of the model over the IDENTITY baseline, which simply copies the input sentence (other ROUGE values are reported in the §A.1). In §5, we will explore how much of a performance drop on the fixed test set is caused by shifting the *training* data by only five years to the past.

Emoji prediction Go et al. (2009) introduce an emoji prediction dataset, collected from Twitter and is time-stamped. We use the 67,980 tweets from June 16 as our **New Sample**, and tweets from all previous days for the remaining experiments. For this task, we again train an MLP over a LASER embedding (Schwenk et al., 2019) with hyper-parameters: two hidden layers with 50 parameters each and ReLU activation functions, trained using the Adam stochastic gradient-based optimizer (Kingma and Ba, 2015), a batch size of 200, and L_2 penalty of strength $\alpha = 0.01$. See §5 for a discussion of temporal drift in this data.

News classification We use a UCI Machine Learning Repository text classification problem.[9] Our datapoints are headlines associated with five different news genres. We use the last year of this corpus as our **New Sample**. We sample 100,000 headlines from the rest and train an MLP over a LASER embedding (Schwenk et al., 2019) with the following hyper-parameters: two hidden layers with 100 parameters and ReLU activation functions, trained using the Adam stochastic gradient-based optimizer (Kingma and Ba, 2015), dynamic batch sizes, and L_2 penalty of strength $\alpha = 0.01$.

---

[5] `https://catalog.ldc.upenn.edu/LDC2013T19`

[6] `github.com/jiesutd/NCRFpp`

[7] `github.com/facebookresearch/SentEval`

[8] `tinyurl.com/zyq3yvn`

[9] `tinyurl.com/yysysmtr`

Task	Model	Splits				
		Standard	**Random**	**Heuristic**	**Adversarial**	**New Samples**
POS TAGGING	NCRF^{++}	0.961	0.962	0.960	0.944	**0.927**
PROBING-WC	BERT	0.520	0.527	**0.232**	0.250	0.279
PROBING-BSHIFT		0.800	0.808	0.695	0.706	**0.450**
HEADLINE GENERATION*	seq2seq	0.073	0.095	0.062	**0.040**	0.069
QUALITY ESTIMATION†	MLP-Laser	0.502	0.626	0.621	0.711	**0.767**
EMOJI PREDICTION		-	0.125	0.196	**-0.040**	0.091
NEWS CLASSIFICATION		-	0.681	0.720	0.634	**0.618**
MSE (**New Samples**)		0.179	0.030	0.015	0.011	-

Table 2: Error reductions over random baselines on **Standard** (original) splits, if available, **Random** splits (obtained using cross-validation), **Heuristic** splits resulting from a sentence length-based threshold, **Adversarial** splits based on (five) approximate maximizations of Wasserstein differences between splits, and on **New Samples**. We bold face the **lowest** error reduction, i.e., where results differ the most from the random baseline. **We see that standard and random splits consistently over-estimate *real* performance on new samples, which is sometimes even lower than performance on adversarial splits.** We also report the mean squared error (MSE) with respect to **New Samples**, which shows **Adversarial** estimates empirical error best. Note: *While annotator bias could explain POS tagging results, there is no annotator bias in the other tasks.* *: For HEADLINES we use an identity baseline. Scores are ROUGE-2; see §A.1 for more. †: For QUALITY ESTIMATION, we report RMSE. The WMT QE 2014 best system obtained RMSE of 0.64; our system is significantly better with 0.50 on the standard split.

3 Results

Our results are presented in Table 2. Since the results are computed on different subsamples of data, we report *error reductions over multinomial random* (or, for HEADLINES, *identity*) *baselines*, following previous work comparing system rankings across different samples (Søgaard, 2013). More formally, we present error reduction as $r = \frac{p_s - p_b}{1 - p_b}$, where p_s and p_b are the performances of the system at hand and the multinomial random baseline.

Our main observations are the following: (a) **Random splits (and standard splits) consistently under-estimate error on new samples**. The absolute differences between error reductions over random baselines for random splits and on new samples are often higher than 20%, and in the case of PROBING-BSHIFT, for example, the BERT model reduces 80% of the error of a random baseline when data is randomly split, but only 45% averaging over five samples of new data from the same domain. (b) Heuristic splits sometimes under-estimate error on new samples. Our heuristic splits in the above experiments are quite aggressive. We only evaluate our models on sentences that are longer than any of the sentences observed during training. Nevertheless for 5/7 tasks, this leads to more optimistic performance estimates than evaluating on *new* samples! (c) The same story holds for adversarial splits based on approximate maximization of Wasserstein distances between training and test data. While adversarial splits are very challenging, results on adversarial splits are more optimistic than on new samples in 4/7 cases. Note the fact that random splits over-estimate real-life performance also leads to misleading **system rankings**. If, for example, we remove the CRF inference layer from our POS tagger, performance on our **Random** splits drops to 0.952; on the **New Sample**, however, performance is 0.930, which is significantly better than *with* a CRF layer.

Discussion In the spirit of earlier work (Sakaguchi et al., 2017; Madnani and Cahill, 2018; Gorman and Bedrick, 2019), we provide recommendations for future evaluation protocols: (i) In the absence of multiple held-out samples, using **biased splits** better approximates real-world performance and can help determine what data characteristics affect performance. (ii) Evaluating on **new samples** is superior and also enables significance testing across datasets (Demsar, 2006), providing confidence estimates. Several benchmarks already provide multiple, diverse test sets (e.g. Hovy et al., 2006; Petrov and McDonald, 2012; Williams et al., 2018); we hope more will follow. What explains the high variance across samples in NLP? One reason is the dimensionality of language (Bengio et al., 2003), but in §A.5 we also show significant impact of temporal drift.

Conclusions We have shown that out-of-sample error can be hard to estimate from random splits, which tend to underestimate error by some margin, but even biased and adversarial splits sometimes underestimate error on new samples. We show this phenomenon across seven very different NLP tasks and provide practical recommendations on how to best bridge the gap between experimental practices and what is needed to produce truly robust NLP models that perform well in the wild.

Acknowledgments

We would like to thank our reviewers for their comments and for engaging in an interesting discussion. The paper also benefited greatly from discussions with several of our colleagues at Google Research, including Slav Petrov and Sascha Rothe.

References

Martín Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 137–144. MIT Press.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alexander J. Smola. 2006. Integrating structured biological data by kernel maximum mean discrepancy. In *Proceedings 14th International Conference on Intelligent Systems for Molecular Biology 2006, Fortaleza, Brazil, August 6-10, 2006*, pages 49–57.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single \$&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2126–2136.

Janez Demsar. 2006. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bradley Efron. 1981. Nonparametric estimates of standard error: the jackknife, the bootstrap and other methods. *Biometrika*, 68.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 11–21. Association for Computational Linguistics.

Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. Discofuse: A large-scale dataset for discourse-based sentence fusion. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3443–3455. Association for Computational Linguistics.

L. Gillick and Stephen J. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '89, Glasgow, Scotland, May 23-26, 1989*, pages 532–535. IEEE.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, 150.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2786–2791. Association for Computational Linguistics.

Karin Harbusch, Saša Hasan, Hajo Hoffmann, Michael Kühn, and Bernhard Schüler. 2003. Domain-specific disambiguation for typing with ambiguous keyboards. In *Proceedings of the 2003 EACL Workshop on Language Modeling for Text Entry Methods*, TextEntry '03, page 67–74, USA. Association for Computational Linguistics.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

Sarvnaz Karimi, Jie Yin, and Jiri Baum. 2015. Evaluation methods for statistically dependent text. *Comput. Linguistics*, 41(3):539–548.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation, NMT@ACL 2017, Vancouver, Canada, August 4, 2017*, pages 28–39. Association for Computational Linguistics.

Jan Lukes and Anders Søgaard. 2018. Sentiment analysis under temporal shift. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–71, Brussels, Belgium. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Nitin Madnani and Aoife Cahill. 2018. Automated scoring: Beyond natural language processing. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1099–1109. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguistics*, 19(2):313–330.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100, Montréal, Canada. Association for Computational Linguistics.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web.

Nina Pörner, Hinrich Schütze, and Benjamin Roth. 2018. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 340–350. Association for Computational Linguistics.

Shruti Rijhwani and Daniel Preotiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7605–7617, Online. Association for Computational Linguistics.

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

Keisuke Sakaguchi, Courtney Napoles, and Joel R. Tetreault. 2017. GEC into the future: Where are we going and how do we get there? In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 180–187. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, and Armand Joulin. 2019. Ccmatrix: Mining billions of high-quality parallel sentences on the WEB. *CoRR*, abs/1911.04944.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Darsh J. Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. 2018. Adversarial domain adaptation for duplicate question detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1056–1063. Association for Computational Linguistics.

Deven Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020,* *Online, July 5-10, 2020*, pages 5248–5264. Association for Computational Linguistics.

Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. 2018. Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4058–4065. AAAI Press.

Hidetoshi Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

Hava T. Siegelmann and Eduardo D. Sontag. 1992. On the computational power of neural nets. In *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT 1992, Pittsburgh, PA, USA, July 27-29, 1992*, pages 440–449. ACM.

Anders Søgaard. 2013. Estimating effect size across datasets. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 607–611. The Association for Computational Linguistics.

Piotr Szymański and Kyle Gorman. 2020. Is the best better? Bayesian statistical model comparison for natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2203–2212, Online. Association for Computational Linguistics.

Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. 2018. Wasserstein auto-encoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

David H. Wolpert. 1996. The lack of A priori distinctions between learning algorithms. *Neural Comput.*, 8(7):1341–1390.

Xintong Yu, Hongming Zhang, Yangqiu Song, Yan Song, and Changshui Zhang. 2019. What you see is what you get: Visual pronoun coreference resolution in dialogues. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5122–5131. Association for Computational Linguistics.

	ROUGE-1	ROUGE-2	ROUGE-L
Standard	0.080	0.073	0.097
Random	0.109	0.095	0.127
Heuristic	0.091	0.062	0.109
Adversarial	0.060	0.040	0.080
New Sample	0.067	0.069	0.091

Table 3: Error reduction as compared with an identity baseline (output as input) for three ROUGE metrics. Random is a five-fold cross-validation result.

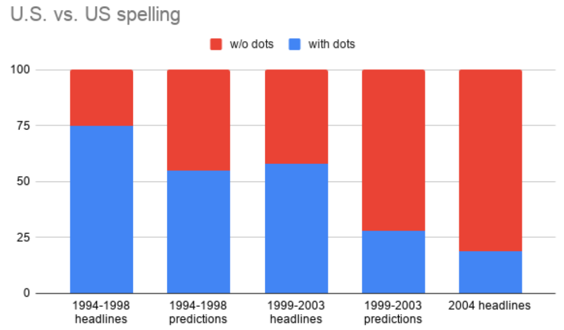


Figure 2: Proportions of *US* vs. *U.S.* spellings in the headlines for two training sets and the test set (2004) as well as in the two models' predictions on the test set.

A Appendices

We present supplementary details about two of our tasks in §A.1 and §A.2 and discuss variations over heuristic splits in §A.3. In §A.4, we present the pseudo-algorithm for how we compute adversarial splits, and finally, in §A.5, we present our results documenting temporal drift.

A.1 Headlines

Table 3 reports the error reduction in ROUGE-1, ROUGE-2 and ROUGE-L over the identity baseline (see §2) for the different data splits. The results are consistent with Table 2. Figure 2 gives more details on an interesting drift phenomenon, which contributed to the superior performance of the model trained on the most recent five years (1999-2003). Apparently, the dotless spelling of *U.S./US* ('United States') became more common over time. Consequently, the model trained on the 1999-2003 part generated *US* more frequently than the model trained on 1994-1998.

A.2 Quality Estimation

In the results above, we train and test our quality estimation regressor on Spanish-English from WMT QE 2014. We also ran a similar experiment where we used the German-English test data as our **New Sample**. Here, we see a similar pattern to the one above: The RMSE on the **Standard** split was 0.630, which is slightly higher than for Spanish-English; with our **Heuristic** split, RMSE is 0.652; for **Adversarial**, it is 0.626 (which is slightly better than with standard splits), and on our **New Sample**, RMSE is 0.813.

A.3 Alternative Heuristic Splits

For both SentEval tasks we experimented with the following alternatives for heuristic splits.

Bootstrap Resampling Instead of cross-validation, a random split can be generated by bootstrap resampling. For this we randomly select 10% of the data as test set and then randomly sample (with replacement) a new training and dev set from the remaining examples.

Random Length As alternative to the length threshold heuristic in earlier experiments we randomly sample a length and select all examples having this length to be part of the test set. We repeat this procedure until approximately 10% of the data ends up in the test set. With this procedure we create 5 different test sets. We included this heuristic in order to see how fragile the probing setup is.

Rare Words Another alternative for heuristic splits is to use word frequency information. Here we assign those sentences containing at least one of the rarest words of the dataset to the test set. This way we end up again with approximately 10% of the data in the test set. Note that this way we create only 1 dataset, because it's not a random process.

Results Table 4 lists the results. While bootstrap resampling leads to slightly lower error reduction than cross-validation we decided to report the latter in the main part of this paper, because it is a more wide-spread way to randomly split datasets. Random Length results are comparable to standard splits results. The split based on word frequency (Rare Words) leads to considerable drop in both tasks. However, it is not as strong as the drop of the heuristic split (length threshold) in the main part of the paper.

A.4 Computing adversarial splits

We present the pseudo-algorithm of our implementation of approximate Wasserstein splitting in Algorithm 1. We also make the corresponding code available as part of our code repository for this paper.

Task	Model	Splits Standard	 Bootstrap	 Random Length	 Rare Words
PROBING-WC	BERT	0.520	0.504	0.554	0.339
PROBING-BSHIFT		0.800	0.807	0.798	0.731

Table 4: Error reductions over random baselines on **Standard** (original) splits, if available, **Bootstrap** splits, **Random Length** splits resulting from a sentence length-based separation, **Rare Words** splits based on word frequency.

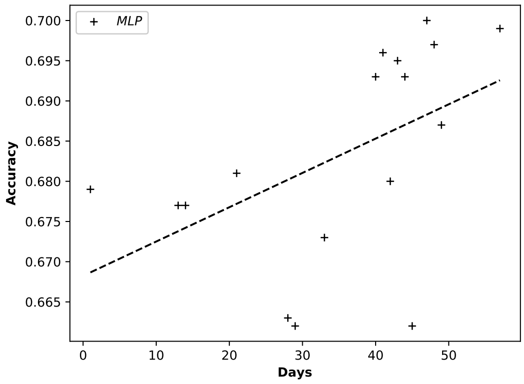


Figure 3: Temporal drift in emoji prediction. The correlation between temporal gap and performance is significant ($p < 0.05$).

```
Data: Dataset $G_{train}$
Result: Adversarial split: $G_{train}$ and $G_{test}$
$T \leftarrow WassersteinBallTree(G_{train})$;
$G_{test} \leftarrow \{\langle x, y\rangle \sim G_{train}\}$;
$G_{train} \leftarrow G_{train}/\{\langle x, y\rangle\}$;
while $i \leq k$ do
    $\langle x_i, y_i\rangle = \min_{\langle x', y'\rangle} T(x, x')$;
    $G_{train} \leftarrow G_{train}/\{\langle x_i, y_i\rangle\}$;
    $G_{test} \leftarrow G_{test} \cup \{\langle x_i, y_i\rangle\}$;
    $i+ = 1$;
end
```

Algorithm 1: Computing adversarial splits

	ROUGE-1	ROUGE-2	ROUGE-L
Identity baseline	0.302	0.110	0.260
50% of 1994-2003	0.409	0.205	0.386
1994-1998	0.346	0.161	0.329
1999-2003	0.413	0.208	0.388

Table 5: Performance of three seq2seq models trained on different samples, evaluated on 2004 data.

A.5 The significance of drift

Some of our splits in the main experiments were based on slicing data into different time periods (HEADLINES, EMOJIS). Since temporal drift is a potential explanation for sampling bias, we analyze this in more detail here. We show that temporal drift is pervasive and leads to surprising drops in performance. We note, however, that temporal drift is not the only cause of sampling bias, of course. Since we have time stamps for two of our datasets we study these in greater detail. For similar studies of temporal drift, see Lukes and Søgaard (2018); Rijhwani and Preotiuc-Pietro (2020).

Headline generation Our headline generation data covers the years 1994 to 2004. Having reserved 20,000 sentence-headline pairs from the first half of 2004 for validation and the same amount from the second half for testing, we use 50% of the years 1994-2003 for training three models. The models' architectures and parameters are identical (same as in Sec. 3). The only difference is in what the models are trained on: (a) a random half, (b) the first, or (c) the second half of 1994-2003. The training data sizes are comparable (1.63-1.76M), the publisher distributions (AFP, APW, CNA, NYT or XIN) are also similar. Hence, the models are expected to perform similarly on the same test set.

As Table 5 indicates, shifting the training data by five years to the past results in a big performance drop. Sampling training data randomly or taking the most recent period produces models with similar ROUGE scores, both much better than the identity baseline. However, about half of the gap to the identity baseline disappears when older training data is taken. In the §A.1, we give an example of temporal drift in the HEADLINES data: *US* largely replaces *U.S.* in the newer training set and the test set.

Emoji prediction For emoji prediction, Go et al. (2009) provide data for a temporal span of 62 days. We split the data into single days and keep the splits with more than 25,000 datapoints in which both classes are represented. We use the last of these, June 16, as our test sample and vary the training data from the first day to the day before June 16. Figure 3 (left) visualizes the results.

How Certain is Your Transformer?

Artem Shelmanov[‡], **Evgenii Tsymbalov**[†,‡], **Dmitri Puzyrev**[‡,★], **Kirill Fedyanin**[‡],
Alexander Panchenko[‡], **and Maxim Panov**[‡]
[‡]Skolkovo Institute of Science and Technology, Moscow, Russia
[†]Huawei Industry Video Application Lab, Moscow, Russia
[★]Mobile TeleSystems (MTS), Moscow, Russia
{a.shelmanov,evgenii.tsymbalov,dmitry.puzyrev,k.fedyanin,a.panchenko,m.panov}@skoltech.ru

Abstract

In this work, we consider the problem of uncertainty estimation for Transformer-based models. We investigate the applicability of uncertainty estimates based on dropout usage at the inference stage (Monte Carlo dropout). The series of experiments on natural language understanding tasks shows that the resulting uncertainty estimates improve the quality of detection of error-prone instances. Special attention is paid to the construction of computationally inexpensive estimates via Monte Carlo dropout and Determinantal Point Processes.

1 Introduction

Quantifying the uncertainty of machine learning models is an important aspect of trustworthy, reliable, and accountable natural language understanding (NLU) systems. Obtaining measures of uncertainty in predictions (also known as *uncertainty estimations*, UE) helps to detect out-of-domain (Malinin and Gales, 2018), adversarial, or error-prone instances that require special treatment. For example, such instances can be additionally checked by human experts or another more advanced system or alternatively rejected from classification (Herbei and Wegkamp, 2006). Besides, uncertainty estimation is an essential component of various applications such as active learning (Shelmanov et al., 2021) and outlier/error detection in a dataset (Larson et al., 2019).

Many modern NLU methods take advantage of deep pre-trained models that are based on the Transformer architecture (Vaswani et al., 2017) (e.g., BERT (Devlin et al., 2019) or ELECTRA (Clark et al., 2020)). Obtaining reliable uncertainty estimations for such neural networks (NNs) can, therefore, directly benefit a wide range of NLU tasks, yet implementing UEs, in this case, is challenging due to the huge number of parameters in these deep learning models. The approximations of Bayesian inference based on dropout usage at the inference stage – Monte Carlo (MC) dropout (Gal and Ghahramani, 2016), provide a realizable approach to quantifying UEs of deep models. However, they are usually accompanied by serious computational overhead due to the necessity of performing multiple stochastic predictions. Importantly, training ensembles of independent models (Lakshminarayanan et al., 2017) leads to even more prohibitive overheads.

In this work, we investigate various MC dropout-based approaches to uncertainty quantification of NLU models on the widely-used General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018). The main contributions of our work are two-fold:[1]

- We show that the use of the MC dropout with pre-trained Transformer models significantly improves the quality of UEs in NLU tasks compared to deterministic baselines.
- We are the first to our knowledge to apply a modification of the MC dropout based on determinantal point processes (DPP; Tsymbalov et al. (2020)) to Transformers and show that this approach allows obtaining the UEs competitive to the standard MC dropout at a fraction of its cost. To improve the stability of the DPP-based dropout for Transformer-based models, we extend the method presented in Tsymbalov et al. (2020) by averaging multiple dropout masks sampled with DPP.

2 Related Work

Three dominating approaches to uncertainty estimation in neural networks exist: (i) interpretation of the model's logits from the uncertainty estimation perspective (Gal, 2016), which is the basic one; (ii) ensembling, where a discrepancy

---

[1] Code of our experiments: `https://github.com/skoltech-nlp/certain-transformer`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1833–1840
April 19 - 23, 2021.

between models' predictions are interpreted as a sample variance (Lakshminarayanan et al., 2017); (iii) Bayesian neural networks (Teye et al., 2018), which have a built-in mechanism to capture uncertainty via a single model.

There are a few recent works that investigate uncertainty quantification for NLP models and use MC dropout techniques. Dong et al. (2018) use Bayesian UEs for the analysis of semantic parser predictions for correctness. Zhang et al. (2019) propose an additional training loss component that facilitates smaller inter-class and bigger intra-class distances in the vector space of the output layer. Experiments with convolutional NNs on text classification datasets show that this modification helps to improve error detection using MC dropout UEs. For quantifying model data uncertainty, Xiao and Wang (2019) use NNs to parameterize a probability distribution (mean and variance) instead of making a prediction directly. For quantifying model uncertainty, the authors leverage the MC dropout. Modeling both types of uncertainties in convolutional and recurrent NNs helped them to improve the performance in regression and classification NLP tasks. Kochkina and Liakata (2020) apply UEs to the problem of rumor verification.

3 Uncertainty Estimation of Deep Transformer Neural Networks

In this section, we describe types of dropout, uncertainty estimation methods, and the Transformer-based neural classifier used in our experiments.

3.1 Types of Dropout

We use two types of dropout described below.

Monte Carlo Dropout The dropout (Srivastava et al., 2014) has emerged as a powerful and universal regularization technique applicable to most DL architectures, with the Transformers not being an exception. Despite originally being an empirical, engineering way to fight the overfitting, it then obtained a theoretical explanation as a special case of Bayesian NNs, where activations are drawn from the Bernoulli distribution (Gal, 2016). This allows to represent a vector of outputs x_h of the h-th layer of the network as a function of its weights W_h, activation function σ, and a dropout mask M_h:

$$x_h = \sigma(x_{h-1} \mid W_h, M_h), M_h \sim Bernoulli(1{-}p),$$

where $p \in [0; 1]$ is the dropout rate.

This theoretical explanation enables the use of the dropout not only at the training stage but also at the inference stage via sampling of multiple masks $M_h^{(t)}, t = 1, \dots, T$ for each dropout layer of the network h and subsequently providing an ensemble of models parameterized by these masks: $f_t(x) = f\big(x \mid \{M_h^{(t)}\}\big)$. The obtained UEs are relatively fast, convenient, and applicable to various tasks, such as regression (Tsymbalov et al., 2018), image classification (Gal and Ghahramani, 2016), and active learning (Gal et al., 2017; Siddhant and Lipton, 2018).

Monte Carlo Dropout with Determinantal Point Processes The models obtained from the standard dropout masks usually show a high degree of correlation in predictions between them, limiting the power of the resulting ensemble. Recently, it was proposed to improve the diversity of predictions by considering the correlations between neurons and sampling the diverse neurons via the mechanism of Determinantal Point Processes (DPP; Kulesza and Taskar (2012)), an approach for sampling diverse elements from a set of points. This setup was proposed by (Tsymbalov et al., 2020) and evaluated for the simple multilayer perceptrons and CNNs. In this work, we aim to extend this approach to Transformer models.

DPP-based dropout masks M_h^{DPP} for the h-th layer are constructed using the correlation matrix C_h between neurons as a likelihood kernel for the DPP: $M_h^{\mathrm{DPP}} \sim DPP\big(C_h\big)$. The probability to select a set S of activations on the layer h is given by

$$P[M_h^{\mathrm{DPP}} = S] = \frac{\det C_h^S}{\det\big(C_h + I\big)},$$

where C_h^S is a square submatrix of C_h obtained by keeping only rows and columns indexed by the sample S. The matrix of correlations between activations of the h-th layer C_h is estimated empirically based on some set of points, which represents the data distribution well enough (i.e. training set). The key feature of the approach is that DPP tends to sample neurons with low correlations between them, which in turn improves the overall diversity of the obtained models. More information about DPP is presented in Appendix B.

To improve the stability of the DPP-based dropout for Transformer-based models, we create a final dropout mask by sampling from DPP and averaging multiple initial masks.

3.2 Uncertainty Estimates

Let T be a number of stochastic passes, i.e., the number of dropout masks to be sampled. We use the three following UEs (also known as *acquisition methods*) for the classification with C classes:

- **Sampled maximum probability**:

  $$1 - \max_c \bar{p}_T(y = c \mid x),$$

  where $\bar{p}_T$ is an average probability for the class c prediction over multiple stochastic passes $t = 1, \ldots, T$.

- **Probability variance** averaged over classes:

  $$\frac{1}{T}\sum_{c=1}^{C}\sum_{t=1}^{T}\big(p_t(y = c \mid x) - \bar{p}_T(y = c \mid x)\big)^2.$$

- Bayesian Active Learning by Disagreement (**BALD**) proposed by Houlsby et al. (2011) describes the mutual information between outputs and model parameters:

  $$H(x) + \frac{1}{T}\sum_{c=1}^{C}\sum_{i=1}^{T} p(y = c \mid x) \log\big(p(y = c \mid x)\big),$$

  where $H(x)$ is the entropy of the ensemble mean.

We would like to note that all these estimates can be used for any ensembling technique, including the MC dropout and the DPP-based dropout.

3.3 Classification Models

In this work, we focus on the **ELECTRA** (Clark et al., 2020) model, which is a recent successor to BERT (Devlin et al., 2019). It is based on the same Transformer architecture but takes advantage of the harder "replaced token detection" objective instead of the "masked language model" objective. This gives better pre-training capabilities and makes ELECTRA the state-of-the-art Transformer in natural language understanding benchmarks. We should note that ELECTRA is regularized with multiple dropout layers, which facilitates the usage of the MC dropout. For example, the body of the "ELECTRA-base" model has 37 dropout layers.

We also experiment with **DistilBERT** (Sanh et al., 2019), which is a smaller Transformer obtained from the middle-size BERT (Devlin et al., 2019) via a distillation procedure (Hinton et al., 2015). This model provides the faster inference and has smaller memory requirements but retains 97% of the language understanding capabilities of the original model according to Sanh et al. (2019).

4 Experiments

4.1 Experimental Setup

We evaluate the UEs on the basis of their ability to detect misclassification. High UEs should indicate potential errors in the model output, while low uncertainties should correspond to correctly classified instances. In this vein, we transform the original task into a binary classification task by comparing predictions of a model with the ground truth labels in the validation dataset. Uncertainty estimates on the validation dataset are treated as the outputs of the binary classifier that is trained to look for potential errors. We calculate the ROC AUC score using the new ground truth labels and UEs and use this score as the main evaluation metric.

The baseline in this task is the UE calculated based on the maximal probability of the original deterministic model. We compare it to the estimates obtained using multiple stochastic predictions with activated dropout layers. Three variants of estimates are calculated: 1) based on the model, in which MC dropout is applied to all dropout layers; 2) based on the model with the MC dropout applied only to the last layer; 3) based on the model with the DPP-based sampling applied to the last dropout layer. For calculating these UEs, we conduct 20 stochastic predictions. The dropout rate in these passes for the MC dropout is 0.1, which is shown to be optimal in the preliminary experiments. For the DPP dropout, we sample and average multiple masks produced by DPP. In experiments with SST-2 and ELECTRA, we average as many masks so at least 30% of neurons remain active during the pass (this roughly can be considered as a "dropout rate" of 0.7). For MRPC, we choose the "dropout rate" equal to 0.2 and for CoLA: 0.4. For DistilBERT, we use the "dropout rate" of 0.4 in all tasks.

We train three versions of models with different random seeds. For each model, another five random seeds are used to produce predictions for stochastic methods. Multiple models and predictions are used for estimating the standard deviation and conducting the statistical significance testing.

4.2 Datasets

We evaluate UEs and dropout variants on the widely used NLU benchmark GLUE (Wang et al., 2018). Specifically, we perform experiments on three tasks: Stanford Sentiment Treebank (SST-2; Socher et al. (2013)), Corpus of Linguistic Acceptability (CoLA; Warstadt et al. (2019)), and

Dropout Type	Model Acquisition	Dropout Layers	Tasks SST-2	MRPC	CoLA
No (baseline)	Max. probability	-	79.7±3.4	78.6±4.1	78.7±2.0
MC	Sampled max. probability	last	-0.1±0.2	-0.5±0.4	-0.1±0.1
MC	Probability variance	last	-1.9±1.0	-3.0±0.7	-1.4±0.7
MC	BALD	last	-5.0±1.7	-5.6±1.3	-3.9±1.3
DPP	Sampled max. probability	last	**3.2**±**2.4**	0.0±0.7	-0.4±0.7
DPP	Probability variance	last	**2.7**±**3.1**	**1.5**±**2.3**	-1.1±1.3
DPP	BALD	last	0.7±2.7	**2.1**±**3.1**	-2.0±2.1
MC	Sampled max. probability	all	**3.2**±**1.6**	**5.5**±**2.6**	**3.2**±**0.6**
MC	Probability variance	all	**4.7**±**2.1**	**7.2**±**3.1**	**2.9**±**0.4**
MC	BALD	all	**5.2**±**2.4**	**7.5**±**3.3**	**2.8**±**0.4**

Table 1: The misclassification detection performance (ROC AUC) (±SD) for the baseline with the ELECTRA model and performance improvements over the baseline for various UE methods. Statistically significant improvements (p-value$\leq$ 0.05) are highlighted.

Microsoft Research Paraphrase Corpus (MRPC; Dolan and Brockett (2005)). The SST-2 task is to predict the sentiment of a given sentence (positive/ negative). The SST-2 dataset was randomly subsampled to 2% of the original size to emulate the situation with a small amount of training data. The CoLA task is to determine whether the given sentence is grammatical or not. The MRPC task is to predict whether two given sentences are semantically similar or not. We select these three datasets for their compact size.

4.3 Model and Training Details

We use the middle-size pre-trained ELECTRA-base model with 110 million parameters and the DistilBERT model with 66 million parameters obtained from the middle-size BERT. The implementation of the models is provided by the Huggingface Transformers library (Wolf et al., 2020). For fine-tuning models, we follow the approach described by Clark et al. (2020) and Devlin et al. (2019): train for 4 epochs with 10% warm-up and a linear learning rate scheduler. For all models and tasks, we use the same learning rate equal to 5e-5. For ELECTRA and SST-2 and MRPC tasks, the batch size is 16. For ELECTRA and CoLA, the batch size is 32. For DistilBERT, the batch size is 32 for all tasks. Although calibrating these hyperparameters can yield some performance improvements, the aforementioned settings allow achieving good results across all tasks.

4.4 Results and Discussion

ROC AUC scores for the misclassification detection task and ELECTRA are presented in Table 1. The results for DistilBERT are presented in Table 3 in Appendix A. While the classifier performance does not significantly variate across multiple versions of the fine-tuned models, the difference in the misclassification detection performance is statistically significant. Therefore, we present the absolute values of the performance only for the baseline (UE based on maximum probability), while for other methods, we present the improvement over the baseline across multiple runs. Tables with results also present the standard deviation of scores.

We note that the UE based on the maximum probability of the deterministic model is a strong baseline. Overall, Transformers are able to indicate their potential mistakes with just the probability from the softmax layer. Applying the MC dropout to all dropout layers in the network always gives a reliable boost in the misclassification detection. For SST-2 and MRPC tasks, UE based on BALD demonstrates better performance than sampled maximum probability and variance, while on CoLA, all UEs perform comparably well. The biggest improvement can be achieved for MRPC and ELECTRA: up to 7.5% ROC AUC.

On the contrary, the UEs based on the MC dropout applied only to the last layer do not perform well. We see that the misclassification detection performance always deteriorates compared to the baseline, especially, for variance and BALD.

UEs that take advantage of the DPP-based masks applied to the last dropout layer are somewhere in the middle in terms of quality compared to the MC dropout variants. Although this method also does not give any improvement for CoLA, unlike the last layer MC dropout, DPP gives a significant advantage over the baseline on the SST-2 task for both models and on the MRPC task for ELECTRA. We note that although DPP-based sampling and the last layer MC dropout diverge in terms of "dropout rate" (e.g., in the experiment on the SST-2 task with ELECTRA, 0.7 for the DPP dropout versus 0.1 for the MC dropout), this aspect does not explain the performance difference. Applying dropout rates higher than 0.1 to the MC dropout downgrades the performance of the misclassification detection due to the overall decrease of the model quality, while for DPP, only 30% of neurons is more than enough to retain the model performance and obtain better UEs on the SST-2 task.

Despite the fact that the DPP-based approach appears to be worse than applying the MC dropout on all layers, it is much faster since it is applied to only the last dropout layer. For practical applications, obtaining UEs normally should not cause a significant overhead compared to the standard model inference time. This strikes the methods based on the MC dropout since they require multiple stochastic predictions. However, for most of the pre-trained Transformers, if only the last dropout is replaced with the MC variant, the outputs of the massive Transformer "body" are not affected during the stochastic predictions. This means that the body outputs can be calculated only once, and only the last linear layer with the softmax activation should be recalculated multiple times. As the last layer contains less than 1% of total parameters, this favors the UEs that do not use stochastic inference on dropout layers except the last. Compared to masks generated uniformly with the MC dropout, sampling masks with DPP has some insignificant computation overhead, but, as we showed, it can give a useful contribution to the misclassification performance (for MRPC and SST-2) even if it is used only in the last dropout layer.

We performed an investigation of computation time overhead for calculating UEs with various MC dropout options for the development dataset. The results for ELECTRA are presented in Table 2. The computations were conducted with the Nvidia 2080ti GPU and the Intel Xeon 5217 CPU. We use BALD as an acquisition function, but other functions have comparable execution time. The MC dropout placed on all layers of Transformers gives better improvements, but it causes roughly 2,000% overhead (in the case of 20 stochastic passes), with less than 10% overhead for the last layer MC and DPP. Therefore, DPP can provide a better trade-off between computation time and performance of error detection.

UE Method	**Inference time, sec.** **SST-2**	 **MRPC**
Deterministic, –	3.07 ± 0.03	1.43 ± 0.04
MC dropout, all	65.5 ± 0.7	30.2 ± 0.2
MC dropout, last	3.17 ± 0.06	1.51 ± 0.05
DPP dropout, last	3.33 ± 0.02	1.57 ± 0.01

Table 2: The inference time of the ELECTRA model on the development dataset with BALD UE.

5 Conclusion

In this work, we evaluated several UEs for the state-of-the-art Transformer model ELECTRA and the speed-oriented DistilBERT model in the text classification tasks. To obtain estimates, we leverage multiple stochastic passes using the MC dropout, and the DPP-based dropout proposed by (Tsymbalov et al., 2020). We show that by activating all dropouts in the model for stochastic predictions, one can beat the baseline deterministic uncertainty estimate by the significant margin in the binary misclassification detection task. We also demonstrate that replacing the last dropout layer with the DPP dropout can yield significant improvements over the baseline in some cases, but less than the usage of the MC dropout on all dropout layers. Despite being inferior compared to the latter, the DPP dropout can provide a better trade-off between computation time and performance of error detection, which can be important for practical use cases.

In future work, we are seeking to improve UEs quality obtained using the DPP dropout with the help of calibration (Safavi et al., 2020) and conduct experiments on sequence tagging tasks.

Acknowledgments

We thank the reviewers for their valuable feedback. The development of uncertainty estimation algorithms for Transformer models (Section 3) was supported by the joint MTS-Skoltech lab. The development of a software system for the experimental study of uncertainty estimation methods and its application to NLP tasks (Section 4) was supported by the Russian Science Foundation grant 20-71-10135. The Zhores supercomputer (Zacharov et al., 2019) was used for computations.

References

Ali Çivril and Malik Magdon-Ismail. 2009. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*, 410(47-49):4801–4811.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Li Dong, Chris Quirk, and Mirella Lapata. 2018. Confidence modeling for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 743–753, Melbourne, Australia. Association for Computational Linguistics.

Yarin Gal. 2016. Uncertainty in deep learning. *University of Cambridge*.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1183–1192. PMLR.

Sergei A Goreinov, Ivan V Oseledets, Dimitry V Savostyanov, Eugene E Tyrtyshnikov, and Nikolay L Zamarashkin. 2010. How to find a good submatrix. In *Matrix Methods: Theory, Algorithms And Applications: Dedicated to the Memory of Gene Golub*, pages 247–256. World Scientific.

Radu Herbei and Marten H. Wegkamp. 2006. Classification with reject option. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 34(4):709–721.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745.

Elena Kochkina and Maria Liakata. 2020. Estimating predictive uncertainty for rumour verification models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6964–6981, Online. Association for Computational Linguistics.

Alex Kulesza and Ben Taskar. 2012. Determinantal point processes for machine learning. *Found. Trends Mach. Learn.*, 5(2-3):123–286.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6402–6413.

Stefan Larson, Anish Mahendran, Andrew Lee, Jonathan K. Kummerfeld, Parker Hill, Michael A. Laurenzano, Johann Hauswald, Lingjia Tang, and Jason Mars. 2019. Outlier detection for improved data quality and diversity in dialog systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 517–527, Minneapolis, Minnesota. Association for Computational Linguistics.

Odile Macchi. 1975. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 7(1):83–122.

A Malinin and M Gales. 2018. Predictive uncertainty estimation via prior networks. In *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, volume 31, pages 7047–7058. Curran Associates, Inc.

Tara Safavi, Danai Koutra, and Edgar Meij. 2020. Evaluating the Calibration of Knowledge Graph Embeddings for Trustworthy Link Prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8308–8321, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and bayesian uncertainty estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Online. Association for Computational Linguistics.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Mattias Teye, Hossein Azizpour, and Kevin Smith. 2018. Bayesian uncertainty estimation for batch normalized deep networks. In *International Conference on Machine Learning*, pages 4907–4916. PMLR.

Evgenii Tsymbalov, Kirill Fedyanin, and Maxim Panov. 2020. Dropout strikes back: Improved uncertainty estimation via diversity sampled implicit ensembles. *CoRR*, abs/2003.03274.

Evgenii Tsymbalov, Maxim Panov, and Alexander Shapeev. 2018. Dropout-based active learning for regression. In *Analysis of Images, Social Networks and Texts - 7th International Conference, AIST 2018, Moscow, Russia, July 5-7, 2018, Revised Selected Papers*, volume 11179 of *Lecture Notes in Computer Science*, pages 247–258. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Trans. Assoc. Comput. Linguistics*, 7:625–641.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yijun Xiao and William Yang Wang. 2019. Quantifying uncertainties in natural language processing tasks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7322–7329. AAAI Press.

Igor Zacharov, Rinat Arslanov, Maksim Gunin, Daniil Stefonishin, Andrey Bykov, Sergey Pavlov, Oleg Panarin, Anton Maliutin, Sergey Rykovanov, and Maxim Fedorov. 2019. "zhores" — petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in skolkovo institute of science and technology. *Open Engineering*, 9(1):512 – 520.

Xuchao Zhang, Fanglan Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2019. Mitigating uncertainty in document classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3126–3136, Minneapolis, Minnesota. Association for Computational Linguistics.

A Results for DistilBERT

	Model		Tasks		
Dropout Type	**Acquisition**	**Dropout Layers**	**SST-2**	**MRPC**	**CoLA**
No (baseline)	Max. probability	-	76.6±2.6	76.6±0.8	73.0±1.7
MC	Sampled max. probability	last	-0.0±0.1	-0.1±0.3	0.0±0.1
MC	Probability variance	last	-0.5±0.4	-0.6±0.7	-0.2±0.4
MC	BALD	last	-1.7±0.6	-1.6±1.1	-1.1±0.7
DPP	Sampled max. probability	last	**0.6±1.0**	0.2±0.6	-0.1±0.2
DPP	Probability variance	last	**0.6±1.3**	0.4±1.2	-0.4±0.6
DPP	BALD	last	0.5±1.6	0.2±1.5	-0.8±1.0
MC	Sampled max. probability	all	**0.6±0.2**	**2.1±0.6**	**1.4±0.7**
MC	Probability variance	all	**2.0±0.8**	**2.4±1.0**	**1.3±1.0**
MC	BALD	all	**2.3±1.0**	**2.4±1.2**	**1.1±1.0**

Table 3: The misclassification detection performance (ROC AUC) (±SD) for the maximal probability baseline with the DistilBERT model and performance improvements over the baseline for various UEs. Statistically significant improvements (p-value ≤ 0.05) are highlighted.

B Determinantal Point Processes

Determinantal point processes (DPPs) are specific probability distributions over a set of points. They allow choosing the subset of points enforcing the diversity between the samples. The DPPs were introduced for the needs of statistical physics (Macchi, 1975), and found their applications in machine learning (Kulesza and Taskar, 2012)

For example, consider the situation where we observe N news from different outlets during one specific day. Let us also assume that we can measure the corresponding texts' pairwise similarity. In this case, DPPs allow choosing a number $n \ll N$ of most non-similar news for the day, giving a good representation of the agenda. Most importantly, DPPs have efficient implementation for the exact sampling and several even more efficient approximate solutions. We also note that DPP sampling is stochastic, i.e., it provides a different result for each repetition. That is an essential property for the uncertainty estimation problems we consider in this work.

Formally, let us assume that the kernel matrix K of pairwise similarities between the considered points X is given. DPPs are similar to the algorithm of finding maximum volume submatrix of K (Goreinov et al., 2010; Çivril and Magdon-Ismail, 2009) as geometrically determinant of the matrix is equal to the scaling volume of a corresponding linear transformation. In this case, a large volume is good because it corresponds to orthogonal vectors (i.e. non-similar vectors). Likewise, DPPs sample points S with probabilities:

$$P[S \subset X] = \det K_S,$$

where K_S is the submatrix of the matrix K corresponding to points S.

As probability takes values between 0 and 1, the matrix K needs to be positive semidefinite and should not have minors with determinant larger than 1. In practice, usually only some unnormalized likelihood matrix L is given. The standard approach is to normalize it in the following way:

$$K = L(L + I)^{-1}.$$

In this case, we can directly calculate the submatrix probabilities:

$$P[X = S] = \frac{\det L_S}{\det(L + I)}.$$

Alignment verification to improve NMT translation towards highly inflectional languages with limited resources

George Tambouratzis
ILSP, Athena R.C.
6 Artemidos Str.,
Maroussi, 15125, Greece
giorg_t@athenarc.gr

Marina Vassiliou
ILSP, Athena R.C.
6 Artemidos Str.,
Maroussi, 15125, Greece
mvas@athenarc.gr

Abstract

The present article studies translation quality when limited training data is available to translate towards morphologically rich languages. The starting point is a neural MT system, used to train translation models with only publicly available parallel data. An initial analysis of the translation output has shown that quality is sub-optimal, mainly due to the insufficient amount of training data. To improve translation, a hybridized solution is proposed, using an ensemble of relatively simple NMT systems trained with different metrics, combined with an open source module designed for low-resource MT that measures the alignment level. A quantitative analysis based on established metrics is complemented by a qualitative analysis of translation results. These show that over multiple test sets, the proposed hybridized method confers improvements over (i) both the best individual NMT and (ii) the ensemble system provided in the Marian-NMT package. Improvements over Marian-NMT are in many cases statistically significant.

1 Introduction

The state of the art in MT involves corpus-based systems developed with machine-learning methods. These methods learn from corpora the models needed for translation. A key strength of this approach is that the system is adapted specifically towards the data it is trained with.

For many years, the most successful data-driven approaches were phrase-based and syntax-based Statistical MT (SMT; Koehn, 2009). However, lately Neural MT (NMT) based on the encoder-decoder architecture and the concept of attention (Sutskever et al., 2014; Bahdanau et al., 2016) has become very popular. Indeed, since 2015, in MT shared tasks (Cettolo et al., 2015; Bojar et al., 2015; Bojar et al., 2016) most top-performing systems have been NMT systems. This trend is confirmed in the most recent MT shared task (Barrault et al., 2019), where 80% of participating systems are of NMT type. Though NMT represents the state of the art for MT, specific weaknesses have been reported:

- NMT performance suffers from the lack of data resources (Koehn and Knowles, 2017), giving lower translation performance, especially when training with out-of-domain rather than in-domain data.
- Recent advances in NMT models have been shown (Sennrich and Zhang, 2019) to allow good translations to be achieved with smaller parallel corpora of typically 10^5 sentences, though substantial improvements are achieved when the corpus size reaches 10^6 sentences. However, training sets of such sizes are not available for all languages.
- Translation performance is affected by non-parallel texts and non-literal translations (Carpuat et al., 2017).
- The integration of multiple algorithms into an NMT system does not necessarily improve translation (Denkowski and Neubig, 2017).
- The time complexity of training a new NMT system can be very high, with training sessions of the order of weeks.

NMT requires very large amounts of parallel data, measured in millions of parallel sentences. This is reflected by the separate studies carried out for MT with limited resources, which includes initiatives such as Lorelei [1]. In the case of morphologically-rich languages, the requirements for parallel corpora are further exacerbated.

---

[1] https://www.darpa.mil/program/low-resource-languages-for-emergent-incidents

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1841–1851
April 19 - 23, 2021.

Proposed approaches for translating towards low-resource and morphologically-rich languages have included transfer learning (Zoph et al., 2016) as well as multilingual and multi-way NMT (Rikters et al., 2018).

In this paper, an effort to improve the translation quality is presented, when translating towards a morphologically-rich language, while reducing the training time. This approach combines the output of multiple NMT systems with an NLP module developed for an example-based MT paradigm, resulting in a hybridized solution. The latter module is fast and runs independently of its original MT system and thus the computational complexity of the proposed hybrid solution is not substantially increased over the base NMT system.

The idea of combining multiple MT models to produce a higher performing MT system has been studied extensively in the area of MT. For instance in the recent shared task (Barrault et al., 2019) more than 20 entries consist of ensembles of multiple NMT systems. Ensembles of weaker NMT systems of the same general architecture have been proposed by Freitag et al. (2017) to train a higher performing NMT system. In addition ensembles of factored NMT models have been proposed for automatic post-editing and quality estimation (for example Hokamp, 2017).

This base NMT system is described in section 2. The training data used is reported in section 3. The proposed hybridization is presented in section 4, whilst the improvements attained are presented in section 5. Future developments are discussed in section 6.

2 Overview of the Base NMT System

Since NMT systems have achieved the highest translation quality in recent evaluation contests, the Marian-NMT package (Junczys-Dowmunt et al., 2018) is adopted for experimentation here. Marian-NMT development was funded by the European Commission to consolidate NMT research and incorporates the most recent advances in NMT. Its code is optimized to reduce the CPU/GPU time required to complete the simulations of NMT systems.

For creating NMT systems, three of the models provided by Marian-NMT were chosen, termed as the "transformer", "amun" and "s2s" models. The "transformer" model has been based on the work of Vaswani et al. (2017) and uses a simple structure incorporating attention mechanisms and dispensing with recurrence to implement a fast NMT system. The other two models are more conventional, using a recurrent neural network to implement the translation. The "amun" model follows the approach of Bahdanau et al. (2016), employing a recurrent neural network but allowing the model to automatically search for wider ranges of the source language (SL) to connect with the target language side (TL) words. Finally, "s2s" implements a recurrent neural network-based encoder-decoder model with attention mechanism, using the architecture proposed in (Sennrich et al., 2017). Hereafter, the three models are identified via the names used within Marian-NMT, which are also used in evaluations (cf. Bojar et al., 2018).

The main configuration parameters used for each model are depicted in Table 1, to enable replication of experiments. For each model, different optimization options from Marian-NMT during the validation phase are used to create three NMT variants of each model, namely optimizing with (i) BLEU, (ii) entropy and (iii) word-wise normalized cross-entropy (denoted as "ce-mean" and representing the default optimization for Marian-NMT).

Regarding the main NMT parameters, all recurrent networks comprise 1,024 units in the hidden layer, an encoder depth of 6 layers and an embedding size of 512. All cells used both in the encoder and decoder side are gated recurrent units (GRU). The transformer dimension is set to 2,048. To reduce the lexicon size, a total of 85,000 merge operations are allowed using the BPE (Byte Pair Encoding) method proposed in (Sennrich et al., 2016), this being the default setting for marian-nmt applications.

Initially, the three Marian-NMT models are trained to provide the base NMT systems. Typically, for a single-GPU system (equipped with an NVIDIA Titan XP GTX1080 GPU card driven by an Intel i-9700K CPU), 24 hours are required for training the transformer, 130 hours for amun and 308 hours for s2s. This is equivalent to a ratio of 1:5:12 to train the respective systems.

3 Experimental Set-up

The experiments aim to improve the translation accuracy of an NMT system, taking into account limited training data and constrained computing resources. In order to investigate translation into a lesser-used and highly inflectional language, we

Common to all 3 models	
layer-normalization	yes
exponential smoothing	yes
beam-size	6
normalize	0.6
early-stopping	5
Transformer-specific	
transformer-dropout	0.1
transformer-dropout-attention	0.1
transformer-dropout-ffn	0.1
Amun and s2s-specific	
dropout-rnn	0.2
dropout-src	0.1
dropout-trg	0.1

Table 1: Key NMT hyper-parameters used

corpus	senten.	wordEn	wordGr
raw(Europarl)	1.23M	31.8M	31.9M
raw(DGT)	4.90M	97.8M	87.2M
train(DGT + Europarl)	6.13M	129.6M	119.1M
devel(Eparl)	3,000	77,681	78,610
testset2(Eparl)	1,000	27,712	27,630
testset1(Pres.)	200	2,873	2,757

Table 2: Corpora for training and evaluation

have chosen the English-to-Greek language pair.

When selecting the training corpora, it has been decided to refrain from using expensive language resources such as specialized or hand-built parallel corpora. Instead, only standard publicly available parallel corpora have been adopted, namely the Europarl and DGT-Acquis corpora[2], as listed in Table 2.

The largest part of the Europarl corpus and the entire DGT-Acquis corpus are used to train the NMT system. Three small portions of the Europarl corpus have been reserved for test and validation purposes. More specifically, two independent sets of approx. 3,000 Europarl sentences each are excluded, to ensure that the NMT evaluation is unbiased. In the present experiment, one of these sets is used for in-training validation. The other set is reserved to allow additional cross-evaluation of experiments in the future, without invalidating the previously trained models. Finally, a sample of 1,000 sentences from Europarl (Testset2) is retained to provide an unseen in-domain test set.

Another independent test set was drawn from the PRESEMT project resources, comprising 200 sentences which have not been used to either train an MT model or create any resources used herewith (denoted as Testset1).

A preliminary analysis of the NMT outputs has shown that translations are commendably fluent, though errors are evident. A sample of amun translations is shown in Figure 1. In sentence #1, the term “Αμεριϰανοί” (Transl. “Americans”) is erroneously used as a translation of the terms “American”, “European”, and “Japanese monopolies”. Similarly, in sentence #2, the phrase “η ϰαταπολέμηση της φτώχειας” (meaning “the reduction of poverty”) is used to translate semantically diverse phrases, including “genetically modified organisms”, and “the negative social effects of unbridled, unregulated globalization”. Repetition is a widely reported weakness of NMT systems, most frequently attributed to insufficient training data.

An additional problem concerns the translation of rare words (i.e. words with low frequency in the corpus), due to the limited vocabulary that NMT systems can directly handle. This is especially severe when translating towards languages with complex morphology, which increases the effective vocabulary size. For example the word “ostensibly” is translated into Greek as “ostenfigher” (ungrammatical). Similarly the word “room” is translated as “δωματείο” instead of the correct “δωμάτιο” (meaning room), whilst the word “indistinct” is translated as “άχωρος” which is not a valid Greek word.

Another issue is that entire phrases present in the source text may be omitted in the translation. For instance the sentence “Businesses have undertaken the education” is translated by a transformer NMT as “Η εϰπαίδευση έχει αναλάβει≫, [meaning “education has undertaken”]. Hence, the subject “business” has been deleted.

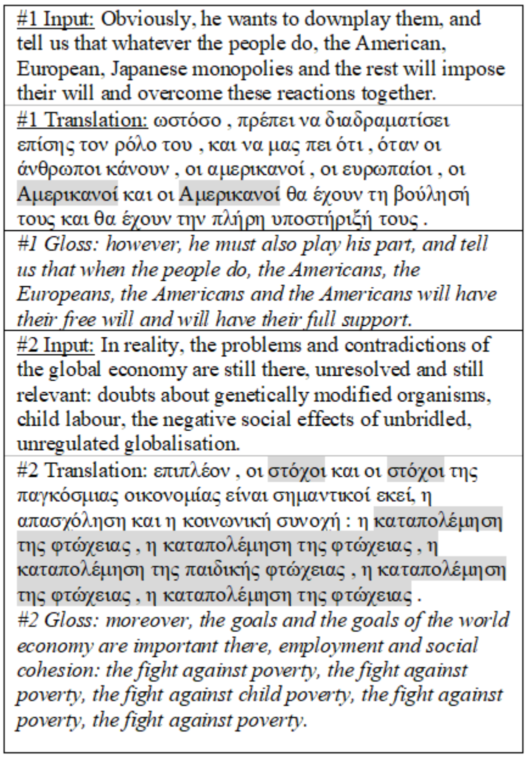
4 Improving NMT via the Alignment Verification Method (AVM)

4.1 Aim of AVM

To improve translation accuracy, the main errors need to be identified in an automated manner. The idea is that a poor alignment between source text and translation indicates substantial loss of meaning during translation. On the contrary a high alignment score is indicative of a high likelihood that

---

[2] The Europarl corpus (ver.7) was retrieved from https://www.statmt.org/europarl. The DGT-Acquis corpus was retrieved from https://ec.europa.eu/jrc/en/language-technologies/dgt-translation-memory



#1 Input: Obviously, he wants to downplay them, and tell us that whatever the people do, the American, European, Japanese monopolies and the rest will impose their will and overcome these reactions together.

#1 Translation: ωστόσο , πρέπει να διαδραματίσει επίσης τον ρόλο του , και να μας πει ότι , όταν οι άνθρωποι κάνουν , οι αμερικανοί , οι ευρωπαίοι , οι Αμερικανοί και οι Αμερικανοί θα έχουν τη βούλησή τους και θα έχουν την πλήρη υποστήριξή τους .

#1 Gloss: however, he must also play his part, and tell us that when the people do, the Americans, the Europeans, the Americans and the Americans will have their free will and will have their full support.

#2 Input: In reality, the problems and contradictions of the global economy are still there, unresolved and still relevant: doubts about genetically modified organisms, child labour, the negative social effects of unbridled, unregulated globalisation.

#2 Translation: επιπλέον , οι στόχοι και οι στόχοι της παγκόσμιας οικονομίας είναι σημαντικοί εκεί, η απασχόληση και η κοινωνική συνοχή : η καταπολέμηση της φτώχειας , η καταπολέμηση της φτώχειας , η καταπολέμηση της παιδικής φτώχειας , η καταπολέμηση της φτώχειας , η καταπολέμηση της φτώχειας .

#2 Gloss: moreover, the goals and the goals of the world economy are important there, employment and social cohesion: the fight against poverty, the fight against poverty, the fight against child poverty, the fight against poverty, the fight against poverty.

Figure 1: Example translations generated by amun, with repetitions of texts highlighted in grey

the NMT output is an accurate translation.

To this end a module will be added to implement alignment verification (AVM), by determining the match between the input sentence and its translation. The establishment of representative alignment scores allows in turn the combination of multiple NMT models, using AVM to evaluate the accuracy of each candidate translation and thus select the best translation on a sentence-by-sentence basis. For this research, MT software and tools released via open-source code have been surveyed and the Phrase Aligner Module (PAM, cf. Troullinos, 2013) has been selected. The architecture of the proposal hybrid NMT is depicted in Figure 2.

4.2 PRESEMT essentials

PAM was developed as part of the PRESEMT hybrid MT methodology (Tambouratzis et al., 2017 (Tambouratzis et al., 2017)). PRESEMT was designed to create MT systems requiring only very limited amounts of specialized, expensive linguistic resources. Frequently, the most expensive resource is the parallel corpus of SL – TL sentences. PRESEMT uses parallel corpora of only a few hundred sentences, augmented by very extensive but comparatively inexpensive monolingual corpora.

Within the PRESEMT methodology, the small parallel corpus serves to establish the transformation from the SL structure to the TL one, using the Phrase Aligner module. This module, handling sentence pairs from this parallel corpus, identifies the correspondence of words and phrases from SL to TL, to determine the translation accuracy.

4.3 Description of the PAM module

PAM utilizes a limited-size bilingual lexicon (of typically 30 to 40 thousand token pairs) together with a publicly available parser. Details on these resources are reported in section 4.4, as their choices are language-specific. Based on these resources, PAM establishes for the set of parallel sentences the alignment of both words and phrases from SL to TL, in three hierarchically ordered stages:

1. Within the first stage, the alignment of words is based on equivalences provided by the bilingual lexicon. Dedicated PAM processes resolve cases where (i) words have multiple appearances within a sentence and (ii) multiple potential translations of an SL word exist in the TL side.

2. Within the second stage, words are aligned by establishing statistical correspondences between grammatical features across the SL and TL pair. These correspondences are automatically extracted from the lexicon.

3. Within the third stage, any remaining words are aligned and grouped into phrases on the basis of the alignments of their neighboring words that are successfully aligned. To implement this, the principle of locality across languages is adopted (words at a small distance to each other in SL also tend to be located close to each other in TL).

The key PAM principle is that decisions made at a later stage have a lower degree of confidence than those made at an earlier stage (Troullinos, 2013).

4.4 Using PAM for Alignment Verification

In the current application, PAM determines the suitability of each candidate translation, based on its match with the source sentence. Thus, the assumption made is that the input sentence and the candidate translation represent the corresponding SL and TL entries of a parallel corpus and PAM determines their level of parallelism.

As the requirement is to grade various translations, the PAM operation is reversed, to identify the quality of match between the input sentence and the generated translations. When PAM was used in PRESEMT, sentence pairs from the parallel corpus with a very low percentage of successful alignments were discarded without measuring their degree of parallelism, as poor exemplars of the structural transformations from SL to TL. Here, PAM is modified so that for all pairs of input sentence and NMT-translation the word alignments and assignments of words to phrases are calculated. This allows the re-roled PAM to grade any source/translation pair, no matter how poor the match of the two sentences is.

Two metrics have been established to calculate divergence between the SL sentence and its NMT-derived translations. The first metric (*Uscore*) calculates the number of unaligned words of the source sentence, after PAM is applied. The aim is to have as few unaligned words as possible, so the lower *Uscore* is, the better the translation is.

$$Uscore = \#unaligned\_words \tag{1}$$

The second metric (*Wscore*) is a weighted combination of several indicators of alignment between source sentence and candidate translation. This summarizes in one measurement the type of alignments and the stage at which they were achieved. Hence, for a sentence with *K* words, *Wscore* is defined as:

$$Wscore = \sum_{i=1}^{K} (w_{\mathrm{i}} * align\_stage_{\mathrm{i}}) \tag{2}$$

In equation (2), *align-stage*$_i$ denotes the stage (cf. section 4.3 for the different stages) at which the i-th SL word is aligned successfully to a TL word, and w_i denotes the relevant weight for this stage. In the case of the weighted metric Wscore, the higher the score, the more accurate the corresponding translation is. The actual weight values must reward the establishment of alignments at an earlier rather than a later stage. Thus, w_i should be larger than w_j , for *i* smaller than *j*. For the purposes of the present article, w_i is set to integer values of 5, 2 and 1 for the first, second and third stage respectively (other sets of weight values that follow this reasoning produce similar results to those reported here). The code of PAM has been modified to integrate *Wscore* and *Uscore* calculation, though the actual alignment algorithms within PAM have remained intact. The associated PAM resources (the TL-side parser and bilingual lexicon) remain unchanged. For processing the SL language, Treetagger (Schmid, 1994) is used, with a reported tagging accuracy exceeding 96%.

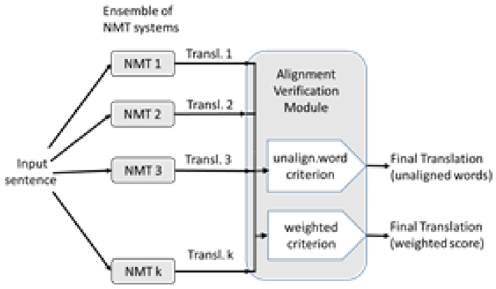


Figure 2: Proposed hybrid NMT approach

5 Experimental Results

To determine the quality of the NMT-based translations (amun-, s2s- and transformer-based models), two widely used MT evaluation metrics are utilised, namely BLEU (Papineni et al., 2002) and NIST (Doddington, 2002). To calculate both these metrics, the mt-eval package (version 13a) is used.

For PAM, the PRESEMT bilingual lexicon from Greek to English is used, which contains approx. 8,000 lemmas and 40,000 Greek-English token pairs. This lexicon is from the same domain as testset1 and is thus out-of-domain for testset2, providing a more limited coverage for this testset.

Two different types of experiments are possible, depending on whether the ensemble comprises multiple NMT architectures, or only one type of architecture. The first experiment reported here involves NMT ensembles that all share the same architecture, but are optimized with different criteria. The second type of experiment studies ensembles which consist of systems with different architectures, to investigate if their combination results in a better translation quality.

5.1 NMT Ensembles of a single architecture

The results obtained for testset1 of the English-to-Greek translation pair are depicted in Table 3, when running the single transformer, amun and s2s models respectively, as well as their ensembles. The corresponding results for testset2 are depicted in Table 4.

In Tables 3 and 4, the first 3 rows correspond to single NMT models generated when Marian-NMT is trained to optimise (i) BLEU, (ii) entropy and (iii) the word-wise normalised cross-entropy (this is denoted as "ce-mean").

The final three rows of Tables 3 and 4 report the accuracy of translations obtained by NMT ensembles. Marian-NMT implements a standard ensemble method, which allows the user to combine different models provided they use the same lexicon. The user may specify weighting factors to boost selection of the models deemed to be better. For this article, this ensemble combines the three aforementioned NMT models (i), (ii) and (iii), with equal weights for all NMTs. The last two rows report the accuracy of ensembles using PAM with (i) Uscore and (ii) Wscore, respectively.

A key difference of the Marian-NMT ensemble is that it is able to recombine partial results of the translation process from each NMT model and thus may generate a new translation that is different from all the translations of single-NMT systems. On the contrary, Uscore and Wscore grade the translations generated by single-NMT models in the ensemble, and then select the highest-scoring translation to be the translation produced by the PAM-based ensemble.

To evaluate the quality of translations produced by the PAM-based ensembles, two baselines are selected. The first baseline is the "ce-mean" option of the Marian-NMT translation system.The second, and stronger, baseline is the Marian-NMT ensemble (referred to as "Marian-ensemble" hereafter). Entries that exceed the first baseline are depicted in bold. Entries with scores that exceed the stronger Marian-ensemble baseline are annotated with an asterisk.

Based on Table 3, for testset1 the best BLEU scores are achieved by the Marian-NMT ensemble in comparison to single-NMT models. The PAM-Wscore ensemble gives a higher accuracy than the Marian-NMT ensemble, whilst the accuracy of PAM-Uscore is lower than PAM-Wscore. On the whole, it is Marian-ensemble and PAM-Wscore that generate the best NIST and BLEU scores.

A broadly similar situation is found when using testset2 (Table 4). Here, the improvement conferred by the ensemble methods over the three base models is much more marked. For instance, for BLEU, the score is only 19.0 to 20.0 for single NMT models, but rises to more than 28.0 for the ensembles, which equates to more than eight BLEU percentage points of improvement.

5.2 Statistical analysis of ensemble results

One question is whether the improvements conferred by the ensembles are statistically significant. To that end, the BLEU and NIST scores of all the independent sentences are assembled, forming two populations of scores (one for BLEU and one for NIST) for each experimental run. Then the Wilcoxon and sign tests are used to determine if these populations have significant differences.

For testset1, the scores of the single NMT systems and the NMT-ensembles are relatively close, differing by less than 2 BLEU points. Applying the sign and Wilcoxon tests, Marian-ensemble produces statistically better NIST scores (at a 0.05 level) than the default Marian-NMT output for amun and s2s models, but not for the transformer model.

For the transformer and s2s models, the scores generated by PAM-Wscore are significantly better that those of single-model Marian-NMT, according to both the Wilcoxon and sign tests (at a 0.05 level). Similarly, PAM-Uscore gives statistically superior results to Marian-NMT (ce-mean optimization) for the s2s model (at a significance level of 0.05).

Comparing the ensembles to each other, Wscore consistently produces higher scores than Uscore. This superiority is statistically significant at a 0.05 level according to both Wilcoxon and sign tests, for the transformer and the amun models.

PAM-Wscore achieves consistently higher translation scores than Marian-ensemble for both BLEU and NIST. According to the Wilcoxon test, these differences are statistically significant, at a 0.05 level, only for the s2s (BLEU score) and the transformer model (both BLEU and NIST scores).

Turning to testset2, the results are more clearly separated. All three ensembles (i.e. PAM-Wscore, PAM-Uscore and Marian-ensemble) have statistically superior scores to Marian (optimised with ce-mean) for both BLEU and NIST, at a significance level of 0.01. This extends to all three NMT models (amun, transformer and s2s), and indicates that both Marian-ensemble and the two PAM-based ensembles give substantially higher scores than single Marian-NMT models.

On the other hand, when comparing PAM-Wscore to PAM-Uscore for testset2, no statistically significant difference (at a 0.05 level of significance) between the two systems is discerned by either the Wilcoxon or sign test. Similarly, no statistically significant differences at a 0.05 level are found between the PAM-based ensembles and the Marian-ensemble and only small differences at a 0.10 level. Thus, even though PAM-based ensembles achieve scores higher than Marian-ensemble, differences are not significant.

5.3 Measuring improvement over baselines

To quantize the improvements achieved by the proposed PAM-Wscore approach, in this section the computational requirements posed by each NMT system are also considered. To this end, the most accurate NMT system is defined for each dataset and metric combination. Two baselines are chosen, namely the most accurate NMT model and the most accurate Marian-ensemble.

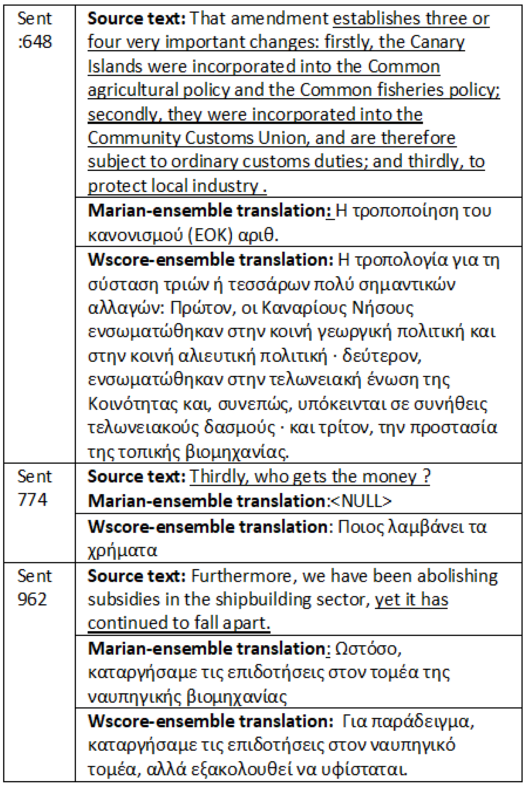
We focus on the transformer model, which is the least expensive model to train. For each ensemble using transformers, the aim is to determine how close to the Marian-ensemble baseline this is. Results are shown in Table 5, where the accuracy of each transformer NMT is expressed as a fraction of the Marian-ensemble score.

The best single transformer model achieves for testset1 88.7% of the baseline BLEU score and 93.1% of the NIST score. Using the Wscore ensembling method, this rises to 90.7% for BLEU and 95.2% for NIST, showing a gain of 2%.

Turning to dataset2, the single transformer scores just 70.5% in comparison to the baseline BLEU score and 73.4% of the NIST score (therefore it is 27% to 30% lower). The Wscore ensemble improves relative scores, reaching 92.7% and 94.5% of the baseline scores for BLEU and NIST respectively. This equates to an increase of ca. 22% in both scores, making the final result directly comparable to s2s, though GPU training requirements are reduced by a factor of 12.

5.4 Subjective studies

A second type of evaluation moves away from metrics to focus on analysing the translation errors by different models, with subjective methods. For instance, when transformer NMT models are tasked to translate testset1, the BLEU-optimised NMT generates 26 ungrammatical words, the entropy-optimised NMT generates 24 ungrammatical words and the cross-entropy optimised model produces 23 ungrammatical words. The Wscore-ensemble reduces the ungrammatical words to 21, improving translation. The ungrammatical words were determined in all cases by visual inspection of the body of translations complemented by spell-checking tools to aid detection.



Sent :648	**Source text:** That amendment establishes three or four very important changes: firstly, the Canary Islands were incorporated into the Common agricultural policy and the Common fisheries policy; secondly, they were incorporated into the Community Customs Union, and are therefore subject to ordinary customs duties; and thirdly, to protect local industry .
	Marian-ensemble translation: Η τροποποίηση του κανονισμού (ΕΟΚ) αριθ.
	Wscore-ensemble translation: Η τροπολογία για τη σύσταση τριών ή τεσσάρων πολύ σημαντικών αλλαγών: Πρώτον, οι Καναρίους Νήσους ενσωματώθηκαν στην κοινή γεωργική πολιτική και στην κοινή αλιευτική πολιτική · δεύτερον, ενσωματώθηκαν στην τελωνειακή ένωση της Κοινότητας και, συνεπώς, υπόκεινται σε συνήθεις τελωνειακούς δασμούς · και τρίτον, την προστασία της τοπικής βιομηχανίας.
Sent 774	**Source text:** Thirdly, who gets the money ? **Marian-ensemble translation**:<NULL>
	Wscore-ensemble translation: Ποιος λαμβάνει τα χρήματα
Sent 962	**Source text:** Furthermore, we have been abolishing subsidies in the shipbuilding sector, yet it has continued to fall apart.
	Marian-ensemble translation: Ωστόσο, καταργήσαμε τις επιδοτήσεις στον τομέα της ναυπηγικής βιομηχανίας
	Wscore-ensemble translation: Για παράδειγμα, καταργήσαμε τις επιδοτήσεις στον ναυπηγικό τομέα, αλλά εξακολουθεί να υφίσταται.

Figure 3: Examples of poor translations produced by Marian-ensemble (omitted parts are underlined in source).

Further inspection of translation quality has involved comparing the Marian-ensemble and Wscore-ensemble outputs. The length (in words) of translations per test sentence is found to differ substantially between the two ensembles, with the difference being more than 1/10 for 9% of sentences, more than 1/4 for 2.5% of sentences and more than 1/2 for 1% of sentences (close to identical results are obtained for testset1 and testset2). As such deviations are unexpectedly large, an analysis was performed, with typical examples being shown in Figure 3. As can be seen, PAM assists the Wscore-ensemble in retaining all phrases of the sentence. On the contrary, Marian-ensemble fails to ensure this, and frequently discards portions of the input sentence. In one case (sentence #774) Marian-ensemble results in a null-length translation, and in another (sentence #648) the final translation covers less than 10% of the input text, radically distorting meaning. Both PAM-ensembles are unaffected by such phenomena.

6 Conclusions and Future Work

This article has studied the creation of translation systems towards highly inflectional languages, when the amount of in-domain training data is limited. Emphasis has been placed on improving the translation accuracy of NMT models that can be trained more rapidly and cost-effectively (in terms of CPU processing power) and rendering this performance comparable to that of more complex models. The Marian-NMT package has been chosen as the starting point to create NMT models for the English to Greek language pair. Using only publicly available text corpora, the NMT models produce commendably fluent translations. Identified errors in the NMT translations are typical of a lack of training data.

A hybrid methodology has been proposed that samples an ensemble of NMT models to select the final translation, chosen by a module calculating the alignment level between the input sentence and each translation. This module was developed for resource-poor MT systems.

The proposed hybrid approach has resulted in higher BLEU and NIST scores, compared to those of single NMT models. Improvements are in many cases statistically significant even over the ensemble system provided within the Marian-NMT package, indicating the promising nature of the hybrid approach. Also, the translation process is found to be more robust, giving more consistent translations in comparison to the Marian-NMT ensemble system, which occasionally omits large portions of the input text from the translation.

One of the advantages of the proposed method is that it is general-purpose and does not rely on the use of ensembles of Neural MT systems with a specific architecture. Instead, it can be used to combine the results of different types of Neural MT systems, or MT systems that belong to different paradigms, or even to combine human translations.

In addition the proposed method can be used to clean up a corpus of parallel sentences or several such corpora, by removing sentence pairs for which the source and target-language texts do not have a high degree of parallelism. Similarly, the proposed method may be used to filter a corpus consisting of original text and its MT-derived translation, to produce a parallel corpus for training of other MT systems, fulfilling a role similar to that proposed by (Rikters and Fishel, 2017). One point for future research is how effective a filtering system based on PAM would be, in comparison to already proposed systems.

Future work involves some relatively simple activities that can be imminently implemented, such as releasing the modified version of PAM for experimentation by interested parties. Another short term activity involves using the proposed method with sacreBLEU instead of the BLEU and NIST metrics provided by mt-eval. Future experiments will investigate the effectiveness of this hybrid approach for other language pairs. One area of interest would be to determine the effectiveness of the PAM-based method when very limited dictionaries are available as well as the limitations when the accuracy of the parser used is relatively low. All these represent issues for the future.

It is also planned to study the approach using systematic optimisation of the PAM parameters, to identify in more detail configurations that produce more accurate translations. Another possibility is to use PAM to detect sub-sentential parts of the translated sentences with particularly poor alignments between input and translation and seek better translations of only these specific parts.

Another direction is to investigate more extensively cases where the translation is not sufficiently close to the input sentence. Then, comparisons to other low-scored translations are more difficult and result in a reduced level of confidence of the chosen translation. Such a line of study will evaluate more thoroughly the robustness of the proposed method.

Acknowledgements

The authors acknowledge support of this work by the project “DRASSI” (MIS5002437) which is implemented under the Action “Reinforcement of the Research and Innovation Infrastructure”, funded by the Operational Programme “Competitiveness, Entrepreneurship and Innovation” (NSRF2014-2020) and co-financed by Greece and the European Commission (European Regional Development Fund). The authors wish to acknowledge the contribution of NVIDIA who donated for research purposes in the area of Machine Translation a Titan XP GPU card under the NVIDIA Academic Support Programme to the MT group of ILSP/Athena R.C.

	BLEU			**NIST**		
criterion	transformer	amun	s2s	transformer	amun	s2s
BLEU-optimised (1)	**35.22**	36.92	35.28	**6.238**	6.440	6.459
Entropy-optimised (2)	34.80	**37.91**	35.25	6.213	**6.532**	6.428
Ce-mean-optimised (3)	34.96	37.77	35.58	6.222	6.524	6.476
Marian-ensemble (1,2,3)	**35.63**	**38.38**	**39.70**	**6.305**	**6.590**	**6.707**
PAM–ensemble + Uscore	33.94	37.25	**39.57**	6.128	6.463	**6.721***
PAM–ensemble + Wscore	**35.99***	**38.56***	**39.79***	**6.384***	**6.584**	**6.758***

Table 3: Translation accuracy of NMT models and ensembles, where each ensemble consists of identically structured NMTs that have been optimized with different criteria (using testset1)

	BLEU			**NIST**		
criterion	transformer	amun	s2s	transformer	amun	s2s
BLEU-optimised (1)	18.27	**19.11**	18.92	4.100	**4.271**	3.894
Entropy-optimised (2)	**20.41**	**19.07**	20.04	**4.944**	**4.260**	4.396
Ce-mean-optimised (3)	18.84	18.83	20.48	4.254	4.129	4.526
Marian-ensemble (1,2,3)	**26.48**	**26.9**5	**28.79**	**6.454**	**6.507**	**6.735**
PAM–ensemble + Uscore	**26.35**	**27.61***	**28.96***	**6.407**	**6.467**	**6.684**
PAM–ensemble + Wscore	**26.68***	**27.45***	**28.85***	**6.363**	**6.513***	**6.716**

Table 4: Translation accuracy of NMT models and ensembles, where each ensemble consists of identically structured NMTs that have been optimized with different criteria (using testset2)

	Testset1	**Testset1**	**Testset2**	**Testset2**
Model	BLEU	NIST	BLEU	NIST
Best NMT (single model)	95.5% (amun)	97.4% (amun)	71.1% (s2s)	73.4% (transf)
Best NMT (Marian-ensemble)	100% (s2s)	100% (s2s)	100% (s2s)	100% (s2s)
Transformer (single model)	88.7%	93.1%	70.9%	73.4%
Transformer (Marian-ensem)	89.7%	94.0%	92.0%	95.8%
Transf PAM + Uscore	85.5%	91.4%	91.5%	95.2%
Transf PAM + Wscore	90.7%	95.2%	92.7%	94.5%

Table 5: Scores achieved for testset1 by different transformer models in comparison to the two baseline models, reported in the first two rows. Scores are normalized over the Marian-ensemble score (cf. row 2).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate. *Computing Research Repository*, arXiv:1409.0473. Version 7.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. 2015. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Marine Carpuat, Yogarshi Vyas, and Xing Niu. 2017. Detecting cross-lingual semantic divergence for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 69–79, Vancouver. Association for Computational Linguistics.

Mauro Cettolo, Jan Niehues, Sebastian Stuker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2015. The iwslt 2015 evaluation campaign. In *IWSLT Proceedings*, Da Nang.

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27, Vancouver. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram cooccurrence statistics. In *HLT '02: Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145.

Chris Hokamp. 2017. Ensembling factored neural machine translation models for automatic post-editing and quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 647–654, Copenhagen, Denmark. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Philipp Koehn. 2009. *Statistical Machine Translation*. Cambridge University Press, Cambridge, UK.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

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

Matiss Rikters and Mark Fishel. 2017. Confidence through attention. *CoRR*, abs/1710.03743.

Matīss Rikters, Mārcis Pinnis, and Rihards Krišlauks. 2018. Training and adapting multilingual NMT for less-resourced and morphologically rich languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Valencia, Spain. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Ilia Sutskever, Oriol Vinyals, and Quok V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the NIPS Conference*, page 3104–3112. NIPS.

George Tambouratzis, Marina Vassiliou, and Sokratis Sofianopoulos. 2017. *Machine Translation with Minimal Reliance on Parallel Resources*. Springer-Verlag, Berlin.

Michalis Troullinos. 2013. Phrase aligner. technical report. *Faculty of Informatics, Masaryk University Brno, FI MU Report Series FIMU-RS-2013-2*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

Data Augmentation for Voice-Assistant NLU using BERT-based Interchangeable Rephrase

Akhila Yerukola[*] **Mason Bretan**[*] **Hongxia Jin**
Samsung Research America
{a.yerukola, mason.bretan, hongxia.jin}@samsung.com

Abstract

We introduce a data augmentation technique based on byte pair encoding and a BERT-like self-attention model to boost performance on spoken language understanding tasks. We compare and evaluate this method with a range of augmentation techniques encompassing generative models such as VAEs and performance-boosting techniques such as synonym replacement and back-translation. We show our method performs strongly on domain and intent classification tasks for a voice assistant and in a user-study focused on utterance naturalness and semantic similarity.

1 Introduction

With conversational assistants becoming more and more pervasive in everyday life, task-oriented dialogue systems are rapidly evolving. These systems typically consist of Spoken Language Understanding (SLU) models, tasked with determining the domain category and the intent of user utterances at each turn of the conversation.

The ability to quickly train such models to meet changing and evolving user needs is necessary. However, developers often find themselves in situations with access to very little labeled training data. This is especially true when new functions are deployed, and a large user base has not had a chance to utilize the function, thus, limiting the number of available utterances that can be labeled for training. Furthermore, the process of labeling large amounts of data can be time consuming and expensive. More recently, these challenges have been both enhanced and complicated by privacy concerns and legislation that may prevent the use of user utterances for training.

Much of the recent research addressing data paucity has focused on pre-training using self-supervision and vast amounts of unlabeled data (Devlin et al., 2018; Radford et al., 2018, 2019). Pre-trained models can later be fine-tuned with a much smaller amount of labeled data for specific tasks. In this work, instead of pre-training, we explore methods that enhance and expand the task-specific training set by using data augmentation. While models such as BERT prove to be both useful and relevant, we show that data augmentation during the fine-tuning stage can boost performance even on these large pre-trained models. We implement and compare several pre-existing techniques for data augmentation on Natural Language Understanding (NLU) tasks such as domain classification (DC) and intent classification (IC) for a voice assistant. We also introduce a new method of data augmentation called *Interchangeable Rephrase* (IR) with the goal of "rephrasing" an existing utterance using new language while maintaining the original intent or goal (see Table 1).

Utterance	how do I make a margherita pizza
Domain	Recipe
Intent	Return_Recipe
Rephrase	show me how to cook a margherita pizza
Utterance	can you check air quality in santa rosa
Domain	Weather
Intent	Air_Quality
Rephrase	is it possible to check air quality in santa rosa
Utterance	delete all emails which have come from hotel.com
Domain	Email
Intent	Delete_Emails
Rephrase	get rid of all emails which have come from hotel.com

Table 1: An illustrative example of a generated rephrase using our Interchangeable Rephrase, while still maintaining the original domain and intent.

2 Related Work

Recurrent neural network (RNN)-based VAE generative models (Kingma and Welling, 2013; Bowman et al., 2015) explicitly model properties of utterances like topic, style, and other higher-level syntactic features. The variational component helps in generating diverse text, thus, we use VAEs as a candidate for transforming and augmenting text

---

[*] equal contribution

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1852–1860
April 19 - 23, 2021.

data in our experiments.

Building upon unconditioned text generation, Conditional VAE (CVAE) (Sohn et al., 2015; Hu et al., 2017) generates more relevant and diverse text conditioned on certain control attributes, e.g. tense, sentiment (Hu et al., 2017), style (Ficler and Goldberg, 2017). In this work, our goal is to maintain semantic similarity, therefore, we generate text by conditioning on the original intent or goal.

Guu et al. generate novel sentences from prototypes by exploiting analogical relationships of sentences. They have shown that generated sentences have a varied style. We extend this idea by using prototype utterances and then editing it into a new utterance or rephrase, using both the VAE and CVAE architecture, to generate augmented rephrase data. We call these models VAE-edit and CVAE-edit.

Back translation (BT) is the process of translating an utterance in a certain language to another language and then translating it back to the original language. Certain question answering (QA) models (Yu et al., 2018) have observed that back-translation generates diverse paraphrases, while preserving the semantics of the original sentences. In our case, we use back-translations as rephrases to augment data.

Another form of simple augmentation techniques is EDA: easy data augmentation (Wei and Zou, 2019). They consist of four operations: synonym replacement, random insertion, random swap and random deletion. They show a boost in text classification tasks with these operations on smaller datasets. Since the number of labeled of spoken utterances are limited, we compare to this approach for our NLU tasks.

Automatic speech recognition (ASR) module, which converts audio input into text, introduces some errors in the process before feeding it into downstream NLU tasks. Without modifying ASR or NLU components, an utterance correction module can be used to help with denoising data (Freitag and Roy, 2018). The reconstructed utterances can be further used as data augmentation.

PPDB is a well known paraphrase database consisting of automatically generated paraphrases (Ganitkevitch et al., 2013). We use this database to rephrase utterances by identifying short phrases within the utterance and replacing them with a related phrase according to the database and POS (e.g. “there is a lot of” → “there are plenty of”).

3 Interchangeable Rephrase

BERT is pre-trained using two tasks: Masked LM (MLM) and Next Sentence Prediction (NSP). In our rephrase task, the end-objective is almost identical to the MLM training, and only this procedure is used to train our self-attention model. MLM training allows the model to predict appropriate word(s) to replace the masked token depending on the context of the rest of the phrase. Each input token corresponds to a final hidden vector that is fed into an output softmax over the vocabulary. Thus, to rephrase an utterance that has more or fewer tokens than the original, the desired number of tokens for the rephrase must be known a priori.

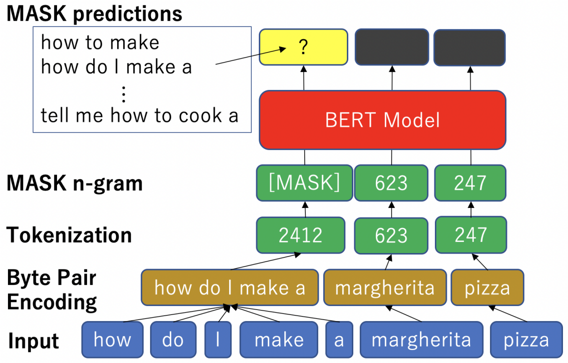


Figure 1: Overview of BERT based interchangeable rephrase (BERT-IR) - BPE is used is to encode n-grams into a single token. Thus, the model’s vocabulary is comprised of tokens representing both single words and sequences of words. The model then computes a softmax over the vocabulary representing the vector of the masked input token allowing for a final output that may be a different word length than the original input.

To allow rephrases with unknown lengths, we use byte pair encoding (BPE) to group word n-grams into single tokens (Sennrich et al., 2015). By using BPE, an individual token may represent a sequence of several words, but the model can still be trained to predict only a single token. We perform BPE on a set of training relevant to the end tasks to get the most frequent n-gram sequences and include these in the model’s vocabulary. Similarly to PPDB, we assume that many of these n-grams are synonyms and are interchangeable. For example, in the context of a virtual assistant skill that enables finding and reciting recipes, n-gram short phrases such as a “how to make”, “tell me how to cook” and “teach me to make” can all be used in the place of the n-gram “how do I make” in the utterance “how do I make a margherita pizza.” This interchangeable property is the foundation of our rephrase and data augmentation system.

The BERT-like self-attention model allows for the predictions to be made based on the context of the input utterance and BPE tokenization allows for variable-length outputs while still only having to predict a single token. Though the new BPE-based vocabulary requires re-training of the model (pre-trained BERT cannot be used), it remains structurally the same as the original BERT model. We refer to our BERT based interchangeable rephrase model as **BERT-IR**. Figure 1 provides an overview of the rephrase model.

In order to maintain intent such that the bigram "turn on" in the utterance "turn on the lights" is not inadvertently replaced with "turn off", negative examples and an intent feature are included in a fine-tuning step. The fine-tuning process minimizes a loss function based on the cosine similarity function

$$sim(\tilde{X}, \tilde{Y}) = \frac{\tilde{X}^T \cdot \tilde{Y}}{|\tilde{X}||\tilde{Y}|} \tag{1}$$

where $\vec{X}$ and $\vec{Y}$ are two equal length vectors and negative examples are included through a softmax function

$$P(\tilde{R}|\tilde{Q}) = \frac{\exp(sim(\tilde{Q}, \tilde{R}))}{\sum_{\tilde{d} \epsilon D} \exp(sim(\tilde{Q}, \tilde{d}))} \tag{2}$$

where $\vec{Q}$ is a one hot ground truth token label derived from the input Q, and $\vec{R}$ represents the output vector of Q. D is the set of three vectors that includes $\vec{R}$ and two vectors derived from two negative examples in the training set. The network then minimizes the following differentiable loss function using gradient descent

$$-log \prod_{(Q,R)} P(\tilde{R}|\tilde{Q}) \tag{3}$$

We emphasize that it is not necessary to use a BERT model for this rephrase method. It is technically possible to use a model architecture identical to that of BERT. However, given that this is not a fine-tuning task and the model needs to be fully re-trained to support the new n-gram tokens a less resource intensive and data hungry model is preferred. In our experiments we use a BERT-like model that leverages self-attention, but has only a fraction of the total parameters of BERT.

4 Experiments

In the following experiments, we examine and compare our proposed method with various data augmentation techniques in the context of utterance generation (rephrase) for voice assistants.

First, we study the properties of the utterances the systems are capable of generating. An ideal data augmentation method should create data that either expands upon or fills in missing gaps of the original training distribution, while still being inherently natural and meaningful. In this context, given an input utterance (from the original distribution), the goal is to generate rephrases that are semantically similar, yet, different enough to positively alter the original training distribution. This is measured in our next two experiments in which we compare the performance of augmented datasets by training an utterance domain classifier (DC) and an intent classifier (IC).

Finally, we perform a user study to examine the quality of utterance generation in terms of naturalness and semantic similarity.

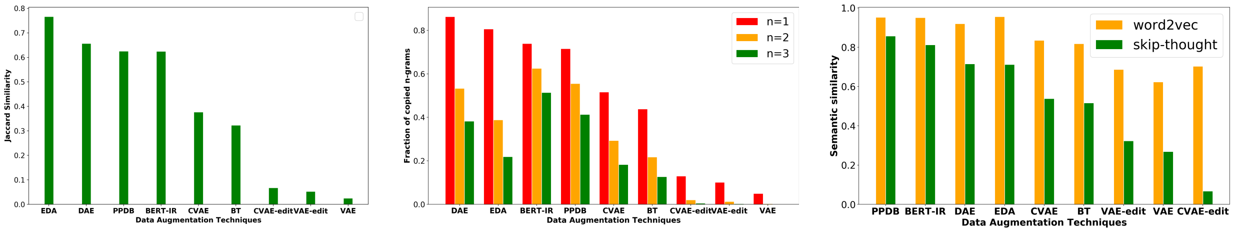
Data. We use an original dataset comprised of utterances for a set of 63 skills (domains) for a voice assistant. The skills range from playing a song on Spotify to turning on/off in-home appliances to providing the weather. The developer of each skill provides a set of training utterances that users can say as an entry point to the function. Each utterance is annotated with an intent, and on average there are 530 utterances per skill (maximum 2000 and minimum 9).

For the DC task, of the 43540 utterances in the dataset, 6590 are held out for validation and the remaining utterances are used for training and processed for augmentation. Additionally, we use a separate test set that was collected and labeled through user trials and crowd sourcing. Each domain in this test set has roughly 900 test utterances.

For the IC task, we consider 5 different skills (domains) from the above dataset: Weather, SystemApp, SmartThings, TvControl and TvSettings. We allocate 30% of the data in each skill for testing and use the remaining for training including augmentation. (see Appendix A.3 for more details).

4.1 Experimental Settings

Our BERT-based interchangeable rephrase model uses an architecture that is similar in size and structure to DistilBERT (Sanh et al., 2019). It is possible to use alternative types of models, but we are motivated by the power of self-attention mechanisms for sequential tasks. Our model is pre-trained with roughly 500k utterances from user data (in



(a) **Jaccard Similarity:** ↓ lower is better

(b) **Copied n-gram fraction:** ↓ lower is better

(c) **Semantic Similarity:** ↑ higher is better

Figure 2: Evaluation of automatic metrics: Averaged across all the generated rephrases for each model.

USA) for the voice assistant. The vocabulary is established using a combination of the developer training data and the usage data. After the byte-pair-encoding process, we prune the resulting pairs so that only pairs occurring more than 100 times are used in the final vocabulary. All unigrams appearing two or more times are also included in the vocabulary.

Baselines. Our first baseline is a classifier that is finetuned solely on the original training data with no augmentation. We compare with existing work on data augmentation via VAE/CVAE, VAE/CVAE-edit, back-translation, denoising autoencoder (DAE), easy data augmentation (EDA) and PPDB. We refer the reader to Appendix A.1 for experimental details of the above augmentation techniques.

4.2 Analysis and Comparison of Methods

We apply several automated linguistic metrics to examine differences in quality of generated rephrases. We evaluate the generated rephrases based on how *related* they are to the original utterances. A model which minimizes the word-level overlap (i.e more variation) and increases the semantic similarity the most is presumably ideal.

Jaccard similarity (Jaccard, 1912; Roemmele et al., 2017) is used to measure the proportion of overlapping words between the rephrase and the original utterance. Additionally, for $n = 1, 2, 3$, we measure the proportion of generated n-grams that also appear in the original utterance i.e amount copied from original utterance (See et al., 2019).

We measure semantic similarity at the word-level and sentence-level. We compute the the mean cosine similarity of the word2vec vectors of all pairs of words between a rephrase and the original utterance. We also measure the cosine similarity of the sentence encodings, generated by the skip-thought model (Kiros et al., 2015), of the rephrase and the original utterance (see Figure 2c). The skip-thought model maps sentences sharing semantic and syntactic properties to similar vector representations.

MODEL	HELD OUT	TEST ACC
NO - AUGMENTATION	0.9229	0.8251
VAE	0.9212	0.8379
CVAE	0.9277	0.8447
VAE-EDIT	0.84	0.6891
CVAE-EDIT	0.8617	0.7469
BACK-TRANSLATION	0.9361	0.8881
DAE	0.9307	0.8765
EDA	0.9896	0.8921
PPDB	0.9341	0.8764
BERT-IR	0.9738	**0.9062**

Table 2: Domain classification results.

We average all these metrics across all the generated rephrases for each model (see Figure 2).

4.3 Domain and Intent Classification

The data generated from each augmentation technique is used to train a domain classifier and an intent classifier. Ten classifiers are trained for each task, using the same distribution of training and held out data described previously. Though several of the augmentation techniques are capable of generating more than one utterance per input, here we generate augmented data in a 1-1 fashion i.e, for each original utterance in the training set, a single rephrase is generated (see Figure 3).

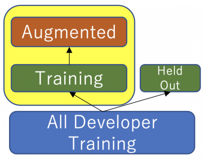


Figure 3: Training distribution for domain and intent classification. All data within yellow boundary box is used for training the classifier.

For both the tasks, we train a classifier on top of the base uncased DistilBERT [1] model (see A.2).

---

[1] `https://huggingface.co/transformers/model_doc/distilbert.html`

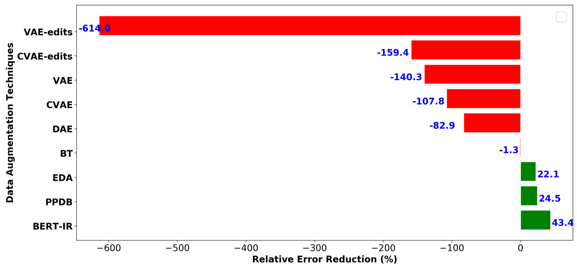


Figure 4: Intent Classification: Relative error reduction of each augmentation type compared to the original "no augmentation" classifier. A positive value (green bar) indicates better performance.

SOURCE	AVG. NATURALNESS	INTENT ACCURACY
HUMAN	3.9	84.5
PPDB	3.1	72.3
OURS	3.4	78.6

Table 3: User study results. The average likert naturalness (where 5 is very natural and 1 is very unnatural) and intent matching accuracy are reported for each of the three utterance sources.

The results on DC task are shown in Table 2. In Figure 4, we show the relative error reduction of each augmentation technique achieved on IC, averaged across all 5 skills (see Appendix A.4 for complete details).

4.4 User study

We performed an online user study where participants completed two tasks. In the first task, participants answered a three item Likert survey regarding the naturalness of the utterance (see Appendix A.5). We compared results for utterances that came from the original developer training (i.e. human-generated), our proposed BERT-IR, and a baseline rephrase using PPDB.

In the second task, users were given a reference utterance and four candidate utterances. Of the four candidates, one of them had an identical intent as the reference, but with different wording, and the other three were random utterances with different intents. The participants were asked to select the candidate with the same intent as the reference. The "correct" candidate was generated from one of the three sources: human, our rephrase, or PPDB rephrase. There were 88 participants, and each participant completed each task 30 times. Results for both tasks are shown in Table 3. Using a 3-way ANOVA, we found significant differences between all three methods on a $p < .05$ level.

5 Discussion

In our experiments, we show that VAE models are capable of generating diverse rephrases. However, these rephrases do not preserve the original meaning. This likely contributed to poorer performance on DC and IC. The discriminator of CVAE models trained to condition on a domain (DC) or intent (IC) helps improve semantic similarity resulting in slightly improved performance of NLU tasks. The VAE-edit and CVAE-edit models perform quite poorly on all comparisons as they don't necessarily preserve the meaning of the utterance when transforming them into an altered style.

Back-translation yields relatively diverse rephrases; however, it poorly conserves the original meaning. DAE just tends to copy a higher fraction of n-grams, while not changing the meaning of the utterance. EDA has boosted the DC and IC performance, compared to other methods. However, it merely changes a small percent of words in utterances by replacement operations, which are not always grammatically sound.

Our BERT-IR yields the best performance on the NLU tasks, with a relative error reduction of 46.26% on DC and 43.4% on IC as compared to no additional augmentation. Similar to PPDB, our approach generates rephrases with a lower word-overlap and a significantly higher semantic similarity with the original utterance. The user study reveals that BERT-rephrase is an improvement over the PPDB baseline, but does not perform at the level of human generated utterances in terms of naturalness and intelligibility.

For examples of generated rephrases see Table 8 in the Appendix.

6 Conclusion

We introduced BERT-IR, a simple augmentation strategy based on byte pair encoding and a BERT-like self-attention model to generate diverse, natural and meaningful rephrases in the context of utterance generation for voice assistants. We demonstrate that BERT-IR performs strongly on spoken understanding tasks like domain classification and intent classification and in a user-study focused on evaluating quality of rephrases based on naturalness and interpretability (intent preservation).

References

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*.

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. *arXiv preprint arXiv:1804.07899*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

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

Melissa Roemmele, Andrew S Gordon, and Reid Swanson. 2017. Evaluating story generation systems using automated linguistic analyses. In *SIGKDD 2017 Workshop on Machine Learning for Creativity*, pages 13–17.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. 2019. Do massively pretrained language models make better storytellers? *arXiv preprint arXiv:1909.10705*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491.

Jason W Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

A Appendices

A.1 Implementation and Training Details of Augmentation Strategies

VAE: The decoder and encoder of the VAE model are set as single-layer GRU RNNs with input/hidden dimension of 100/150 and max sample length of 15. Larger input/hidden dimension models performed similarly. To avoid vanishingly small KL term in the VAE module, we use a KL term weight linearly annealing from 0 to 0.15 during training (similar to training procedure in Hu et al.). Training a larger VAE model yielded comparable results.

CVAE: The discriminator is set as ConvNets similar to Hu et al.. Balancing parameters are set to $\lambda_c = \lambda_z = \lambda_u = 0.1$. Training procedure is exactly as Hu et al..

VAE edit, CVAE-edit: We match tuples of utterances which have the same domain and intent. These pairs are used as the prototypes and new utterances during training of VAE-edit and CVAE-edit (similar to training procedure in Guu et al. by constructing edit vectors and concatenating it with z before feeding into the decoder).

Denoising Auto-encoder: The decoder and encoder of the denoising auto-encoder are set as single-layer GRU RNNs with input/hidden dimension of 300/512 with Luong attention. We follow a similar utterance corruption as Freitag and Roy by dropping words randomly whose frequency was greater than 100 (i.e non-content words like "the"), followed by shuffling of its bigrams while not splitting bigrams that also exist in the original utterance. We train a DAE using this corrupted data. We use the corrected utterance/rephrases generated as augmentation data.

We train the above models until convergence on 2 NVIDIA Tesla V100 GPUs.

Back-translation: We use the open-sourced back-translation system from (Xie et al., 2019). Specifically, we use pre-trained WMT'14 English-French translation models (in both directions) to perform back-translation on each utterance.

EDA: We followed three strategies by Wei and Zou: random insertion, random replacement and random swap.

PPDB: The PPDB [2] database consists of 1 paraphrase rule per line. A standard format of a line is:

```
LHS ||| PHRASE ||| PARAPHRASE |||
(FEATURE=VALUE )* ||| ALIGNMENT |||
ENTAILMENT
```

If a PHRASE exists in the utterance, we replace it with the PARAPHRASE. Since it's a 1-1 data augmentation, we sample a rephrase from the list of all possible generated rephrases. We use the English-Phrasal PPDB database .

Refer to Table 4 for the average run time for the different augmentations techniques.

MODEL	TIME
VAE (GPU)	1 MINUTE
CVAE (GPU)	1.5 MINUTES
VAE-EDIT (GPU)	2 MINUTES
CVAE-EDIT (GPU)	3 MINUTES
BACK-TRANSLATION (GPU)	30 MINUTES
DAE (GPU)	1 MINUTE
EDA (CPU)	3 MINUTES
PPDB (CPU)	11 MINUTES
BERT-REPHRASE (GPU)	7 MINUTES

Table 4: Average time to run inference for augmentation over the training set (36,950 utterances) assuming one generated output per utterance.

A.2 Domain Classification and Intent Classification

For domain classification and intent classification, we train a classifier on top of the base uncased DistilBERT model. We use a maximum sequence length of 128, a dropout rate of 0.1, and a learning rate of 2e-5. We train the classifier for 15 epochs and batch size of 32 on a single NVIDIA Tesla V100.

SPLITS	NUM. OF UTTERANCES
TRAIN	36950
HELD OUT	6590
TEST	56820

Table 5: Domain Classification Data Statistics

A.3 Training Data Statistics

Domain Classification: Table 5 shows the counts for each training, development (held out), and test set. The partition used for training is also the partition that is augmented (and subsequently also used for training in domain classification, see Figure 3). Each utterance has an average of 6 words. There are 63 different domains in our dataset like music, calendar, calculator, weather etc. We evaluate the automatic metrics from Section 4.2 on the same data used for training the domain classifier.

---

[2] http://paraphrase.org/#/download

DOMAIN/SKILL	NUM. OF INTENTS	TRAIN DATA SIZE	TEST DATA SIZE
WEATHER	22	732	313
SYSTEMAPP	29	665	286
SMARTTHINGS	61	1190	511
TVCONTROL	19	392	168
TVSETTINGS	40	834	357

Table 6: Intent Classification Data Statistics

Intent Classification: Table 6 shows the training split counts for each of the 5 different domains. Similar to domain classification, the partition used for training is also the partition that is augmented (and subsequently also used for training in intent classification, see Figure 3).

A.4 Intent Classification Results

As shown in Table 6, the amount of training data available to train an intent classifier for each domain is very less. This can explain the poor performance of VAE models and its variations on IC since they require a lot more training data to perform well. This is also illustrated in the results of domain classification where these VAE based models achieve a slight improvement since there is a higher amount of training data available. Please refer to Table 7 for the fine-grained performance results of the augmentation techniques on the 5 domains.

A.5 User Study Details

Participants responded to three items on a Likert-scale of one to five:

1. This utterance is natural. (disagree to agree)
2. It is easy to understand the intent. (disagree to agree)
3. A person is likely to say this to a voice assistant to achieve the underlying goal. (disagree to agree)

MODEL	WEATHER	SYSTEMAPP	SMARTTHINGS	TVCONTROL	TVSETTINGS
NO - AUGMENTATION	0.97129	0.97894	0.95882	0.92814	0.92977
VAE	0.92971	0.91929	0.92745	0.88023	0.88764
CVAE	0.93062	0.95087	0.92941	0.88023	0.88764
VAE-EDIT	0.72966	0.81403	0.68823	0.80239	0.9382
CVAE-EDIT	0.92344	0.9614	0.87058	0.88622	0.87921
BACK-TRANSLATION	0.97368	0.97192	0.95294	0.93413	0.95505
DAE	0.94976	0.95438	0.92352	0.8982	0.91853
EDA	0.99361	0.98245	0.96078	0.93413	0.92696
PPDB	0.98564	0.98245	0.9666	0.93413	0.93539
BERT-IR	**0.9968**	**0.98596**	**0.97058**	**0.94011**	**0.94382**

Table 7: Test accuracy for Intent Classification

Model	**Generated Rephrase**
Original	*barbershop near me*
VAE	send an email to tony
CVAE	barbershop near me
VAE-edit	is mike’s bikes closed on weekends?
CVAE-edit	where is the highest rated dentist nearby?
Back-translation	the ”Teller” Argile been in close commute for us!
DAE	barbershop near near me
EDA	near me barbershop
PPDB	barbershop near me
BERT-IR	barbershop close to my current location

Model	**Generated Rephrase**
Original	*call an uber espanol to los altos taqueria for me*
VAE	what is the time difference between kuala lumpur and and 40 feet
CVAE	call holly davis espanol to a stadium in brooklyn
VAE-edit	order a lyft premier to 1
CVAE-edit	order a lyft lux for 2 to a cake for me
Back-translation	call a espanol uber at altos taqueria commands for me
DAE	look for an espanol espanol los altos
EDA	call an uber espanol to los for me altos taqueria
PPDB	’m calling the uber espanol to los altos taqueria for me
BERT-IR	please contact an uber espanol to los altos taqueria for me

Model	**Generated Rephrase**
Original	*how is today’s visibility?*
VAE	send a lyft premier to see you
CVAE	how is today’s visibility?
VAE-edit	what’s the weather going to be here
CVAE-edit	navigate to golden gate bridge
Back-translation	How do we make today’s visibility?
DAE	what is today’s visibility?
EDA	how is nowadays’s visibility?
PPDB	how is today’s visibility?
BERT-IR	i want to know how’s today’s visibility

Table 8: Examples of generated rephrases by the different augmentations techniques

How to Evaluate a Summarizer: Study Design and Statistical Analysis for Manual Linguistic Quality Evaluation

Julius Steen Katja Markert
Department of Computational Linguistics
Heidelberg University
69120 Heidelberg, Germany
(steen|markert)@cl.uni-heidelberg.de

Abstract

Manual evaluation is essential to judge progress on automatic text summarization. However, we conduct a survey on recent summarization system papers that reveals little agreement on how to perform such evaluation studies. We conduct two evaluation experiments on two aspects of summaries' linguistic quality (coherence and repetitiveness) to compare Likert-type and ranking annotations and show that best choice of evaluation method can vary from one aspect to another. In our survey, we also find that study parameters such as the overall number of annotators and distribution of annotators to annotation items are often not fully reported and that subsequent statistical analysis ignores grouping factors arising from one annotator judging multiple summaries. Using our evaluation experiments, we show that the total number of annotators can have a strong impact on study power and that current statistical analysis methods can inflate type I error rates up to eight-fold. In addition, we highlight that for the purpose of system comparison the current practice of eliciting multiple judgements per summary leads to less powerful and reliable annotations given a fixed study budget.

1 Introduction

Current automatic metrics for summary evaluation have low correlation with human judgements on summary quality, especially for linguistic quality evaluation (Fabbri et al., 2020). As a consequence, manual evaluation is still vital to properly compare the linguistic quality of summarization systems.

While the document understanding conferences (DUC) established a standard manual evaluation procedure (Dang, 2005), we conduct a comprehensive survey of recent works in text summarization that reveals a wide array of different evaluation questions and methods in current use. Furthermore, DUC procedures were designed for a small set of expert judges, while current evaluation campaigns are often conducted by untrained crowd-workers. The design of the manual annotation, specifically *the overall number of annotators* as well as *the distribution of annotators to annotation items*, has substantial impact on power, reliability and type I errors of subsequent statistical analysis. However, most current papers (see Section 2) do not consider the interaction of annotation design and statistical analysis. We investigate the optimal annotation methods, design and statistical analysis of summary evaluation studies, making the following contributions:

1. We conduct a comprehensive survey on the current practices in manual summary evaluation in Section 2. Often, important study parameters, such as the total number of annotators, are not reported. In addition, statistical significance is either not assessed at all or with tests (t-test or one-way ANOVA) that lead to inflated type I error in the presence of grouping factors (Barr et al., 2013). In summarization evaluation, grouping factors arise whenever one annotator rates multiple summaries.

2. We carry out annotation experiments for coherence and repetition. We use both Likert- and ranking-style questions on the output of four recent summarizers and reference summaries. We show that ranking-style evaluations are more reliable and cost-efficient for coherence, similar to prior findings by Novikova et al. (2018) and Sakaguchi and Van Durme (2018). However, on repetition, where many documents do not exhibit any problems, Likert outperforms ranking.

3. Based on our annotation data, we perform

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1861–1875
April 19 - 23, 2021.

Monte-Carlo simulations to show the risk posed by ignoring grouping factors in statistical analysis and find up to eight-fold increases in type I errors when using standard significance tests. As an alternative, we propose to either use mixed effect models (Barr et al., 2013) for analysis or to design studies in such a manner that results can be aggregated into independent samples, amenable to simpler analysis tools.

4. Finally, we show that the common practice of eliciting repeated judgements for the same summary leads to less reliable and powerful studies for system-level comparison when compared to studies with the same budget but only one judgement per summary.

Code and data for our experiments is available at `https://github.com/julmaxi/summary_lq_analysis`.

2 Literature Survey

We survey all summarization papers in ACL, EACL, NAACL, ConLL, EMNLP, TACL and the *Computational Linguistics* journal in the years 2017-2019. We choose this timeframe as we are interested in current practices in summarization evaluation: 2017 marks the publication of the pointer generator network (See et al., 2017), which has been highly influential for neural summarization. We focus our analysis on papers that present *a novel system* for single- or multi-document summarization and take a single or multiple full texts as input and also output text (SDS/MDS). This allows us to concentrate on recommendations for human evaluation of newly developed summarization systems.[1]

Out of the resulting **105** SDS/MDS system papers, we identify all papers that conduct at least one new comparative system evaluation with human annotators for further analysis, leading to **58** papers in the survey. The fact that this is only about half of all papers is troubling given that it has been recently demonstrated that current automatic evaluation measures such as ROUGE (Lin, 2004) are

	Category	Pa.	St.
Evaluation Questions	Overall	17	23
	Content	45	65
	Fluency	29	34
	Coherence	10	11
	Repetition	14	17
	Faithfulness	6	8
	Referential Clarity	2	2
	Other	8	9
Evaluation Method	Likert	32	43
	Pairwise	10	14
	Rank	9	9
	BWS	6	9
	QA	9	14
	Binary	4	4
	Other	2	2
Number of Documents in Evaluation	< 20	6	10
	20-34	22	41
	35-49	3	4
	50-99	14	21
	100	11	14
	> 100	4	4
	not given	1	1
Number of Systems considered	< 3	13	20
	3	17	23
	4	16	23
	5	6	10
	> 5	12	19
	w/ Reference	16	25
	w/o Reference	45	70
Number of Annotations per Summary	1	2	5
	2-3	20	30
	4-5	12	27
	6-10	3	5
	not given	23	28
Overall Number of Annotators	1-5	19	25
	6-10	3	3
	> 10	5	9
	not given	32	58
Annotator Recruitment	Crowd	25	49
	Other	35	46
Statistical Evaluation	t-test	9	16
	ANOVA	9	18
	CI	4	6
	Other/unspecified	7	8
	None	32	47

Table 1: Our survey for 58 system papers with 95 manual evaluation studies (2017-2019). We show numbers both for individual studies and per paper. As a paper may contain several studies with different parameters, counts in the paper column do not always add up.

---

[1] Excluded from the analysis are sentence summarization or headline generation papers, although most of the points we make hold for their evaluation campaigns as well. Summarization evaluation papers that do not present a new system but concentrate on sometimes large-scale system comparisons are discussed in the Related Work section instead. Lists of all included and excluded papers are given in Supplementary Material, which also contains exact evaluation parameters per paper in a spreadsheet.

not good at predicting summary scores for modern systems (Schluter, 2017; Kryscinski et al., 2019; Peyrard, 2019).

We assess both *what* studies ask annotators to judge, as well as *how* they elicit and analyse judgements. The survey was conducted by one of the authors: for most papers, the categories they fell into were obvious. For difficult cases (unclear specifications, papers that do not fit the normal mould) the two authors discussed the categorisations. Survey results are given in Table 1. Further details about the choices made in the survey, including category groupings/definitions and what is included under *Other*, can be found in Appendix B. As many papers conduct more than one human evaluation (for example on different corpora), we also list individual annotation studies (a total of 95).

Of the systems that do have human evaluation, many focus on *content*, including informativeness, coverage, focus, and relevance. Where linguistic quality is evaluated, most focus on general questions about fluency/readability, with a smaller number of papers evaluating coherence and repetition.

In the rest of this section we focus on the three aspects of evaluation we cover in this paper: How to elicit judgements, how these judgements are analysed statistically and how studies are designed.

2.1 Methods

The majority of evaluations is conducted using Likert-type judgements, with the second most frequent method being rank-based annotations, including pairwise comparison. Best-worst scaling (BWS) is a specific type of ranking-oriented evaluation that requires annotators to specify only the first and last rank (Kiritchenko and Mohammad, 2017). QA (Narayan et al., 2018) is used for content evaluation only. This motivates us to compare both Likert and ranking annotations in Section 4.1.

2.2 Statistical Analysis

If a significance test is conducted, most papers analyse their data either using ANOVA or a sequence of paired t-tests. Both tests are based on the assumption that judgements (or pairs of judgements, in case of paired t-test) are sampled *independently* from each other. However, in almost all studies, annotators give judgements on more than one summary from the same system. Thus the resulting judgements are only independent if we assume that all annotators behave identically. Given that prior work (Gillick and Liu, 2010; Amidei et al., 2018), as well as our own reliability analysis in Section 4.1, show that especially crowd-workers tend to disagree about judgements, this assumption does not seem warranted. As a consequence, traditional significance tests are at high risk of inflated type I error rates. This is well known in the broader field of linguistics (Barr et al., 2013), but is disregarded in summarization evaluation. We show in Section 5 that this is a substantial problem for current summarization evaluations and suggest alternative analysis methods.

2.3 Design

Most papers only report the number of documents in the evaluation and the number of judgements *per summary*. This, however, is not sufficient to describe the design of a study, lacking any indication about the overall number of annotators that made these judgements. A study with 100 summaries and 3 annotations per summary can mean 3 annotators did all judgements in one extreme, or a study with 300 distinct annotators in the other. Only 26 of the 95 studies describe their annotation design in full, almost all of which use designs in which a small number of annotators judge all summaries. Only **6** of **49** crowdsourced studies report the full design.

We show in Section 5 that a low total number of annotators aggravates type I error rates with improper statistical analysis. In Section 6 we further show that with proper analysis, a low total number of annotators leads to less powerful experiments. Almost all analysed papers choose designs with multiple judgements per summary. However, we show in Section 6.2 that this — for the purpose of system ranking — leads to loss of reliability as well as power when compared to a study with the same budget and only one annotation per summary.

3 Coherence and Repetition Annotation

To elicit summary judgements for analysis, we conduct studies on two linguistic quality tasks. In the first, we ask annotators to judge the *coherence* of the summaries, while in the second we ask for the *repetitiveness* of the summary. We select these two tasks over the more frequent *Fluency* task as we found in preliminary investigations that many recent summarization systems already produce highly fluent text, making them hard to differentiate. We do not evaluate *Overall* and *Content* as both require access to the input document, which differentiates these questions from the linguistic quality evaluation of the summaries.

For both tasks, we conduct one study using a seven-point Likert-scale (`Likert`) and another using a ranking-based annotation method (`Rank`), where annotators rank summaries for the same document from best to worst. Screenshots of the interfaces for both approaches and full annotator instructions are given in Appendix A.

Corpus and Systems. Mirroring a common setup (see Section 2), we select four abstractive summarization systems and the reference summaries (`ref`) for analysis.

- The pointer generator summarizer (`PG`) (See et al., 2017), which is still often used as a baseline for abstractive summarization
- The abstractive sentence rewriter (`ASR`) of Gehrmann et al. (2018), which is a strong summarization system that does not rely on external pretraining for its generation step
- `Seneca` (Sharma et al., 2019), a system that combines explicit modelling of coreference information with an external coherence model
- `BART` (Lewis et al., 2020), a transformer network that achieves SotA on CNN/DM.

We randomly sample 100 documents from the popular CNN/DM corpus (Hermann et al., 2015) with corresponding summaries from all systems to form the item set for all our studies.

Study design. We ensure a sufficient total number of annotators by using a block design. We separated our corpus into 20 blocks of 5 documents and included all 5 summaries for each document in the same block, which results in 5 × 5 = 25 summaries per block.

All items in a block were judged by the same set of three annotators. No annotator was allowed to judge more than one block. This results in a total of 3 × 20 = 60 annotators and 1500 judgements per task. Figure 1 shows a schematic overview of our design, which balances the need for a large enough annotator pool with a sufficient task size to be worthwhile to annotators.

We recruited native English speakers from the crowdsourcing platform Prolific[2] and carefully adjusted the reward to be no lower than £7.50 per hour based on pilot studies. Summaries (or sets of summaries for `Rank`) within a block were presented in random order.

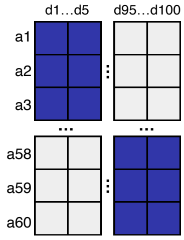


Figure 1: Schematic representation of our study design. Rows represent annotators, columns documents. Each blue square corresponds to a judgement of the summaries of all five systems for a document. Every rectangular group of blue squares forms one block.

4 Ranking vs. Likert

Table 2 shows the average Likert scores and the average rank for all systems, tasks and annotation methods. We use mixed-effect ordinal regression to identify significant score differences (see Section 5 for details). Both annotation methods provide compatible system rankings for the two tasks, though for the repetition task both methods struggle to differentiate between systems. If we were interested in the true ranking, we could conduct a power analysis given some effect size of interest and elicit additional judgements to improve the ranking. However, as we are concerned with the *process* of system evaluation and not the evaluation itself, we do not conduct any further analysis.

In the remainder of this section, we focus on the reliability of the two methods as well as their cost-effectiveness.

4.1 Reliability

Traditionally, reliability is computed by chance-adjusted agreement on individual instances. However, for NLG evaluation, Amidei et al. (2018) argue that a low agreement often reflects variability in language perception. Additionally, we are not interested in individual *document scores*, but in whether independent runs of the same study would result in consistent *system scores*. In Table 3 we thus report split-half reliability (SHR) in addition to Krippendorffs α (Krippendorff, 1980). To compute SHR, we randomly divide judgements into two groups that share neither annotators nor documents, i.e. two independent runs of the study. We

---

[2] `prolific.com`

System	Likert (Coh)	Rank (Coh)	Likert (Rep)	Rank (Rep)
BART	$5.25^{(1)}$	$1.73^{(1)}$	$5.85^{(2/3)}$	$2.88^{(2/3/4)}$
ref	$4.33^{(3/4)}$	$3.31^{(3/4)}$	$6.14^{(1/2)}$	$2.41^{(1/2)}$
ASR	$4.17^{(3/4)}$	$3.17^{(3/4)}$	$4.88^{(4/5)}$	$3.51^{(4/5)}$
PG	$4.81^{(2)}$	$2.68^{(2)}$	$5.63^{(3)}$	$2.92^{(3/4)}$
seneca	$3.52^{(5)}$	$4.11^{(5)}$	$5.16^{(4/5)}$	$3.27^{(3/4/5)}$

Table 2: Results of our annotation experiment. Numbers in brackets indicate rank for a system for a given annotation method. Multiple ranks in the brackets indicate systems at these ranks are not statistically significantly different ($p \geq 0.05$, mixed-effects ordinal regression).

System	α	SHR
Coh: Likert	0.22	0.96
Coh: Rank	0.43	0.98
Rep: Likert	0.27	0.95
Rep: Rank	0.18	0.91

Table 3: Krippendorffs α with ordinal level of measurement and Split-Half-Reliability for both annotation methods on the two tasks.

then compute the correlation[3] between the system scores in both halves. The final score is the average correlation after 1000 trials.

Though agreement on individual summaries is relatively low for all annotation methods, studies still arrive at consistent system scores when we average over many annotators as demonstrated by the SHR. This reflects similar observations made by Gillick and Liu (2010).

We find that on coherence, Rank is more reliable than Likert, though not on repetition. An investigation of the Likert score distributions for both tasks in Figure 2 shows that coherence scores are relatively well differentiated whereas a majority of repetition judgements give the highest score of 7, indicating no repetition at all in most summaries. We speculate overall agreement suffers, because ranking summaries with similarly low level of repetition (and not allowing ties) is potentially arbitrary.[4]

4.2 Cost-efficiency

While more reliable annotation methods allow for fewer annotations, the cost of a study is ultimately determined by the work-time that needs to be invested to achieve a reliable result. To investigate this, we randomly sample between 2 and 19 blocks from our annotations and compute the total time annotators spent to complete each sample. We also compute the Pearson correlation of the system scores in each sample with the scores on the full annotation set. We relate time spent to similarity between sample and full score in Figure 3.

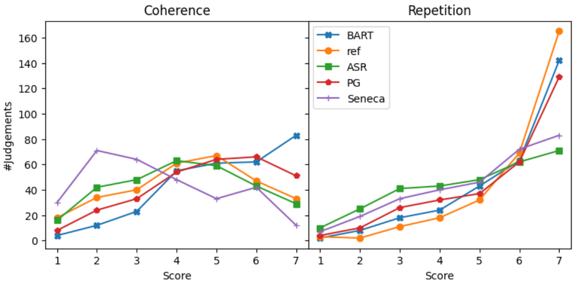


Figure 2: Score distribution of Likert for both tasks. Each data point shows the number of times a particular score was assigned to each system.

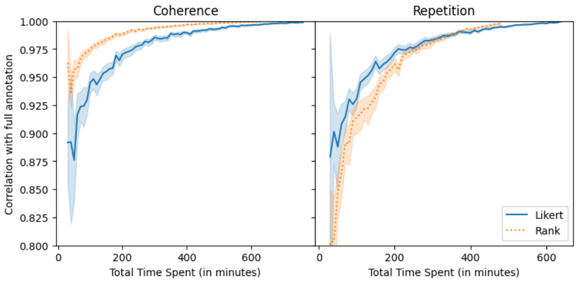


Figure 3: Time spent on annotation (in minutes) vs. correlation with the full-sized score. We gather annotation times in buckets with a width of ten minutes and show the 95% confidence interval for each bucket.

For coherence, Rank is more efficient than Likert. On repetition, the lower reliability of Rank also results in lower efficiency. However, with additional annotation effort, reliability becomes on-par with Likert. This is a consequence of the overall lower annotator workload for Rank.

---

[3] We use the Pearson correlation implementation of scipy (Virtanen et al., 2020).

[4] This is supported by feedback we received from annotators that the summaries were difficult to rank as they mostly avoided repetition well.

5 Statistical Analysis and Type I Errors

The two most common significance tests in summarization studies, ANOVA and t-test (see Table 1), both assume judgements (or pairs of judgements, in the case of t-test) are independent. This is, however, not true for most study setups as a single annotator typically judges multiple summaries and multiple summaries are generated from the same input document. Both documents and annotators are thus grouping factors in a study that must be taken into account by the statistical analysis. Generalized mixed effect models (Barr et al., 2013) offer a solution but have, to the best of our knowledge, not been used in summarization evaluation at all. We choose a mixed effect ordered logit model to analyse our Likert data for both tasks.[5] We will show that traditional analysis methods have a substantially elevated risk of type I errors, i.e. differences between systems found in manual analysis might be overstated.

Method. The ordered logit model we employ can be described as follows:

$$\begin{aligned} &logit(P(Y \leq c)) \\ &= \mu_c - (X\beta + Z_a u_a + Z_d u_d) \end{aligned}$$

where $P(Y \leq c)$ is the probability that the score of a summary is at most c. μ_c is the threshold coefficient for level c, β is the vector of fixed effects and u_a, u_d are the vectors of annotator- and document-level random effects respectively, where u_a, u_d are both drawn from normal distributions with mean 0. Finally, X, Z_a, Z_d are design matrices for fixed and random effects. As the only fixed effect, we use a dummy-coded variable indicating the system that has produced the summary, with `ref` as the reference level. We estimate both random intercepts and slopes for both documents and annotators following advice of Barr et al. (2013) to always specify the maximal random effect structure. In practical terms this means that we allow annotators to both differ in how harsh or generous they are in their assessment, as well as in which system they prefer. Similarly, we allow system performance to vary per-document, leading to both generally higher or lower scores, as well as different system rankings per document.

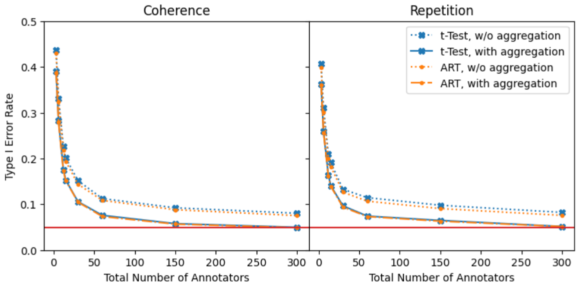


Figure 4: Relation of type I error rates at $p < 0.05$ to the total number of annotators for different designs, all with 100 documents and 3 judgements per summary. We conduct the experiment with both the t-test and approximate randomization test (ART). We show results both with averaging results per document and without any aggregation. We run 2000 trials per design. The red line marks the nominal error rate of 0.05.

We fit all models using the `ordinal` R-package (Christensen, 2019) and compute pairwise contrasts between the parameters estimated for each system using the `emmeans`-package (Lenth et al., 2018) with Tukey-adjustment.

To demonstrate the problem of ignoring the grouping factors, we can now sample artificial data from the model distribution and try to analyse it with inappropriate tests. This Monte-Carlo simulation is similar to the more general analysis of Barr et al. (2013).

We set β to 0 so all systems perform equally well on the population level and only keep the (zero-mean) document and annotator effects in the model. The false-positive rate of statistical tests on this artificial data should thus be no higher than the significance level. We then repeatedly apply both the t-test and the approximate randomization test (ART) (Noreen, 1989), a non-parametric test, to samples drawn from the model and determine the type I error rate at $p < 0.05$. We set the number of documents to 100 and demand 3 judgements per summary to mirror a common setup in manual evaluation. We then vary the total number of annotators between 3 and 300 by changing how many summaries a single annotator judges.

Results. We report results given the model estimated for `Likert` in Figure 4. Ignoring the dependencies between samples leads to inflated type I error rates, whether using the t-test or the ART. This is especially severe when only few annotators judge the whole corpus. In the extreme case with only three annotators in total, the null-hypothesis is rejected in about 40% of trials at a significance

---

[5] We do not include Rank data as the ordinal regression model does not generate ranks.

level of 0.05 in both tasks. Even our original design with 60 annotators still sees an increase of the type I error rate by about 3%. Only if every annotator judges a single document and annotations are averaged per document, samples are independent and thus the real error is at the nominal 0.05 level. This design, however, is unrealistic given that annotators must be recruited and instructed.

We suggest two solutions to this problem: Either use mixed effect models or aggregate the judgements so samples become independent. This allows the assumptions of simpler tools such as ART to be met. In our study, we could average judgements in every block to receive independent samples. This is only possible, however, if the design of the study considers this problem in advance: a crowd-sourcing study that allows annotators to judge as many samples as they like is unlikely to result in such a design.

6 Study Design and Study Power

When conducting studies for system comparison, we are interested in maximizing their power to detect differences between systems. For traditional analysis, the power is ultimately determined by the number of documents (or judgements, when no aggregation takes place) in the study. However, when analysis takes into account individual annotators, power becomes additionally dependent on the total number of annotators and how evenly they participated in the study. This gives additional importance to the design of evaluation studies. In this section, we thus focus on how to optimize studies for power and reliability.

We first show that for well-powered experiments, we need to ensure that a sufficient total number of annotators participates in a study. In the second part of this section, we will then demonstrate studies can improve their power by not eliciting multiple judgements per summary.

6.1 Overall Number of Annotators

To demonstrate the difference in power caused by varying the total number of annotators in a study, we determine the power for a design with the same total number of documents and judgements per document but different total numbers of annotators.

We run the experiment both with regression and ART with proper aggregation of dependent samples as described in Section 5. We refer to the latter as `ARTagg` to differentiate it from normal ART.

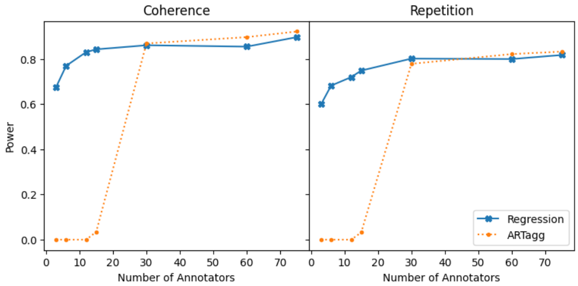


Figure 5: Power for 100 documents and 3 judgements per summary with different number of total annotators.

For each design we repeatedly sample artificial data from the `Likert` model and apply both tests to the data. The process is the same as in Section 5 except we do not set β to zero and count acceptances of the null-hypothesis.[6]

We again set the number of documents to 100 and the number of repeat judgements to 3 and vary the total number of annotators between 3 and 75 by varying the number of blocks between 1 and 25. We test for power at a significance level of 0.05.

Figure 5 shows how power drops sharply when only few annotators take part in the study. This is in line with the theoretical analysis of Judd et al. (2017) that shows that the number of participants is crucial for power when analysing studies with mixed effect models. The drop is worse for `ARTagg` as fewer annotators mean fewer independent blocks and thus a lack of datapoints for the analysis.

6.2 Annotator Distribution

Most studies elicit multiple judgements per summary, following best practices in NLP for corpus design (Carletta, 1996). While this leads to better judgements per *document*, the goal of many summarization evaluations is a per *system* judgement.

For this kind of study, Judd et al. (2017) show that for mixed models that include both annotator and target (in our case, input document) effects, a design where targets are *nested* within annotators, i.e. every annotator has its own set of documents, is always more powerful than one where they are (partially) *crossed* with annotators, i.e. a study with multiple annotations per summary, *given the same total number of judgements*. In fact, power could be maximized by having each annotator judge the sum-

---

[6] As this is an observed power analysis it probably overestimates the power of our analysis for the true effect. The analysis is thus only useful to compare designs under our best estimate of actual effect sizes.

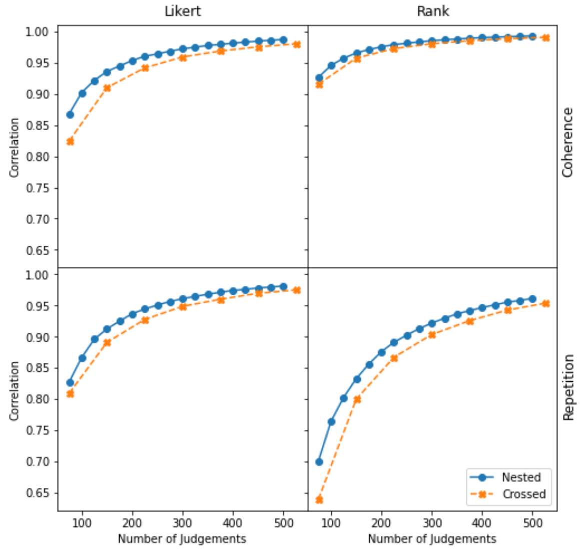


Figure 6: Reliabilities of nested vs. crossed designs for `Rank` and `Likert` for both tasks.

maries for only a single, unique document. However, this is usually not realistic due to the fixed costs of annotator recruitment and instruction. We demonstrate on our dataset how both reliability and power are affected by nested vs. crossed design.

To compare reliability, we randomly sample both nested and crossed designs from our full study and then compute the Pearson correlation of the system scores given by this smaller annotation set with the system scores given by the full study. As shown in Figure 6, nested samples are always at least as good and mostly better at approximating the results of the full annotation compared to a crossed sample with the same annotation effort.

We also conduct a power analysis for regression and `ARTagg` comparing nested and crossed designs. We again turn to Monte-Carlo simulation on the `Likert` models and sample nested and crossed designs with the same total number of judgements (i.e. the same cost). We keep the block size constant at 5 and vary the number of annotators between 3 and 60. For nested designs, we drop the document-level random effects from the ordinal regression, as document is no longer a grouping factor in nested designs.

Figure 7 shows that nested designs always have a power advantage over crossed designs, especially when few judgements are elicited. We also find that ART can be used to analyse data without loss of power when there are enough independent blocks. This might be attractive as ART is less computationally expensive than ordinal regression.

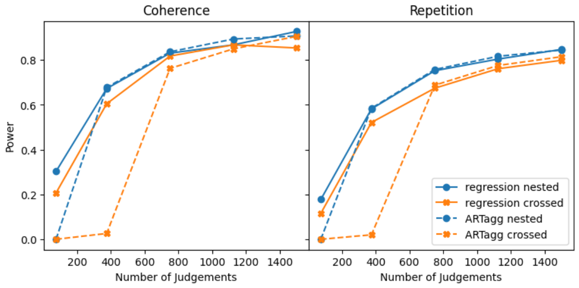


Figure 7: Power for $p < 0.05$ of nested and crossed designs for `ARTagg` and regression. X-axis shows the number of judgements elicited, Y-axis the power-level.

7 Related Work

Human evaluation has a long history in summarization research. This includes work on the correlation of automatic metrics with human judgements (Lin, 2004; Liu and Liu, 2008; Graham, 2015; Peyrard and Eckle-Kohler, 2017; Gao et al., 2019; Sun and Nenkova, 2019; Xenouleas et al., 2019; Zhao et al., 2019; Fabbri et al., 2020; Gao et al., 2020) and improving the efficiency of the annotation process (Nenkova and Passonneau, 2004; Hardy et al., 2019; Shapira et al., 2019). The impact of annotator inconsistency on system ranking has been studied both by Owczarzak et al. (2012) and Gillick and Liu (2010). To the best of our knowledge, we are the first to investigate the implications of annotator variance on the statistical analysis and the design in summarization system comparison studies.

For general NLG evaluation, van der Lee et al. (2019) establish best practices for evaluation studies. We extend on their advice by conducting experimental studies specifically for summary evaluation. In addition, we show the importance of study design and consideration of annotator-effects in analysis on real world data. The advice of Mathur et al. (2017) regarding annotation sequence effects should be taken into account in addition to our suggestions.

Method Comparison. Ranking has been shown to be effective in multiple NLP-tasks (Kiritchenko and Mohammad, 2017; Zopf, 2018), including NLG quality evaluation (Novikova et al., 2018). In this work we confirm this for coherence evaluation, although we find evidence that ranking is less efficient on repetition, where many documents do not exhibit any problems. We also add the dimension of annotator workload as a primary determinant of cost to the analysis of the comparison.

Multiple methods have been suggested to reduce study cost by sample selection (Sakaguchi et al., 2014; Novikova et al., 2018; Sakaguchi and Van Durme, 2018; Liang et al., 2020) or integration with automatic metrics (Chaganty et al., 2018). These efforts complement ours, as care still needs to be taken in analysis and study design.

Recently, rank-based magnitude estimation has been shown to be a promising method for eliciting judgements in NLG tasks and offers a combination of ranking and rating approaches (Novikova et al., 2018; Santhanam and Shaikh, 2019). However, it has not yet found widespread use in the summarization community. While magnitude estimation has been shown to reduce annotator variance, our advice regarding experimental design and grouping factors in statistical analysis applies to this method as well, as annotators can still systematically differ in which systems they prefer.

Statistical analysis. With regard to statistical analysis of experimental results, Dror et al. (2018) give advice for hypothesis testing in NLP. However, they do not touch on the problem of dependent samples. Rankel et al. (2011) analyse TAC data and show the importance of accounting for input documents in statistical analysis of summarizer performance and suggest the use of the Wilcoxon signed rank test for analysis. Sadeqi Azer et al. (2020) argue that p-values are often not well understood and advocate bayesian methods as an alternative. While the analysis in our paper is frequentist, the mixed effect model approach can also be integrated into a bayesian framework. Kulikov et al. (2019) model annotator bias in such a framework but do not account for differences in annotator preferences. In work conducted in parallel to ours, Card et al. (2020) show that many human experiments in NLP underreport their experimental parameters and are underpowered, including Likert-type judgements. Their simulation approach to power analysis is very similar to our experiments. In addition to their analysis, we show that ignoring grouping factors in statistical analysis of human annotations leads to inflated type I error rates. We also show that power can be increased by choosing nested over crossed designs with the same budget. The problem of underpowered studies has also been tackled outside of NLP by Brysbaert (2019).

For psycholinguistics, Barr et al. (2013) demonstrate how generalizability of results is negatively impacted by ignoring grouping factors in the analysis. Mixed effect models have found use in NLP before (Green et al., 2014; Cagan et al., 2017; Karimova et al., 2018; Kreutzer et al., 2020), but to the best of our knowledge they have not been used in summary evaluation.

8 Conclusion

We surveyed the current state of the art in manual summary quality evaluation and investigated methods, statistical analysis and design of these studies. We distill our findings into the following guidelines for manual summary quality evaluation:

Method. Both ranking and Likert-type annotations are valid choices for quality judgements. However, we present preliminary evidence that the optimal choice of method is dependent on task characteristics: If many summaries are similar for a given aspect, Likert may be the better option.

Analysis. Analysis of elicited data should take into account variance in annotator preferences to avoid inflated type I error rates. We suggest the use of mixed effect models for analysis that can explicitly take into account grouping factors in studies. Alternatively, traditional tests can be used with proper study design and aggregation.

Study Design. Study designers should control the number of annotators and how many summaries each individual annotator judges to ensure sufficient study power. Additionally, to ensure reliability of results, studies should report the design and the total number of annotators in addition to the number of documents and repeat judgements. Studies with repeat judgements on the same summary do not provide any advantage for system comparison and are less reliable and powerful than nested studies of the same size.

We hope that these findings will help researchers plan their own evaluation studies by allowing them to allocate their budget better. We also hope that our findings will encourage researchers to take more care in the statistical analysis of results. This prevents misleading conclusions due to ignoring the effect of differences in annotator behaviour.

Acknowledgements

We would like to thank Stefan Riezler for many fruitful discussions about the applications of mixed effect models.

References

Jacopo Amidei, Paul Piwek, and Alistair Willis. 2018. Rethinking the agreement in human evaluation tasks. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3318–3329, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Dale J. Barr, Roger Levy, Christoph Scheepers, and Harry J. Tily. 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language*, 68(3):255–278.

Marc Brysbaert. 2019. How many participants do we have to include in properly powered experiments? a tutorial of power analysis with reference tables. *Journal of Cognition*, 2(1).

Tomer Cagan, Stefan L. Frank, and Reut Tsarfaty. 2017. Data-driven broad-coverage grammars for opinionated natural language generation (ONLG). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1331–1341, Vancouver, Canada. Association for Computational Linguistics.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Jean Carletta. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

Arun Chaganty, Stephen Mussmann, and Percy Liang. 2018. The price of debiasing automatic metrics in natural language evalaution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Melbourne, Australia. Association for Computational Linguistics.

R. H. B. Christensen. 2019. ordinal—regression models for ordinal data. R package version 2019.12-10. https://CRAN.R-project.org/package=ordinal.

Hoa Trang Dang. 2005. Overview of duc 2005. In *In Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2020. Summeval: Re-evaluating summarization evaluation.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Yanjun Gao, Chen Sun, and Rebecca J. Passonneau. 2019. Automated pyramid summarization evaluation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 404–418, Hong Kong, China. Association for Computational Linguistics.

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

Dan Gillick and Yang Liu. 2010. Non-expert evaluation of summarization systems is risky. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 148–151, Los Angeles. Association for Computational Linguistics.

Yvette Graham. 2015. Re-evaluating automatic summarization with BLEU and 192 shades of ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 128–137, Lisbon, Portugal. Association for Computational Linguistics.

Spence Green, Sida I. Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D. Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1225–1236, Doha, Qatar. Association for Computational Linguistics.

Hardy Hardy, Shashi Narayan, and Andreas Vlachos. 2019. HighRES: Highlight-based reference-less evaluation of summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3381–3392, Florence, Italy. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1693–1701, Cambridge, MA, USA. MIT Press.

Charles M Judd, Jacob Westfall, and David A Kenny. 2017. Experiments with more than one random factor: Designs, analytic models, and statistical power. *Annual Review of Psychology*, 68:601–625.

Sariya Karimova, Patrick Simianer, and Stefan Riezler. 2018. A user-study on online adaptation of neural machine translation to human post-edits. *Machine Translation*, 32(4):309–324.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Julia Kreutzer, Nathaniel Berger, and Stefan Riezler. 2020. Correct me if you can: Learning from error corrections and markings. *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*.

Klaus Krippendorff. 1980. *Content Analysis: An Introduction to Its Methodology*. Sage, Beverly Hills, CA.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Ilia Kulikov, Alexander Miller, Kyunghyun Cho, and Jason Weston. 2019. Importance of search and evaluation strategies in neural dialogue modeling. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 76–87, Tokyo, Japan. Association for Computational Linguistics.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan. Association for Computational Linguistics.

Russell Lenth, Henrik Singmann, Jonathon Love, Paul Buerkner, and Maxime Herve. 2018. Emmeans: Estimated marginal means, aka least-squares means. *R package version*, 1(1):3.

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

Weixin Liang, James Zou, and Zhou Yu. 2020. Beyond user self-reported Likert scale ratings: A comparison model for automatic dialog evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1363–1374, Online. Association for Computational Linguistics.

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

Feifan Liu and Yang Liu. 2008. Correlation between ROUGE and human evaluation of extractive meeting summaries. In *Proceedings of ACL-08: HLT, Short Papers*, pages 201–204, Columbus, Ohio. Association for Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2017. Sequence effects in crowdsourced annotations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2860–2865, Copenhagen, Denmark. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759, New Orleans, Louisiana. Association for Computational Linguistics.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 145–152, Boston, Massachusetts, USA. Association for Computational Linguistics.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2018. RankME: Reliable human ratings for natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 72–78, New Orleans, Louisiana. Association for Computational Linguistics.

Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 1–9, Montréal, Canada. Association for Computational Linguistics.

Maxime Peyrard. 2019. Studying summarization evaluation metrics in the appropriate scoring range. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5093–5100, Florence, Italy. Association for Computational Linguistics.

Maxime Peyrard and Judith Eckle-Kohler. 2017. A principled framework for evaluating summarizers: Comparing models of summary quality against human judgments. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 26–31, Vancouver, Canada. Association for Computational Linguistics.

Peter Rankel, John Conroy, Eric Slud, and Dianne O'Leary. 2011. Ranking human and machine summarization systems. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 467–473, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Erfan Sadeqi Azer, Daniel Khashabi, Ashish Sabharwal, and Dan Roth. 2020. Not all claims are created equal: Choosing the right statistical approach to assess hypotheses. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5715–5725, Online. Association for Computational Linguistics.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 1–11, Baltimore, Maryland, USA. Association for Computational Linguistics.

Keisuke Sakaguchi and Benjamin Van Durme. 2018. Efficient online scalar annotation with bounded support. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 208–218, Melbourne, Australia. Association for Computational Linguistics.

Sashank Santhanam and Samira Shaikh. 2019. Towards best experiment design for evaluating dialogue system output. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 88–94, Tokyo, Japan. Association for Computational Linguistics.

Natalie Schluter. 2017. The limits of automatic summarisation according to ROUGE. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 41–45, Valencia, Spain. Association for Computational Linguistics.

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

Ori Shapira, David Gabay, Yang Gao, Hadar Ronen, Ramakanth Pasunuru, Mohit Bansal, Yael Amsterdamer, and Ido Dagan. 2019. Crowdsourcing lightweight pyramids for manual summary evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 682–687, Minneapolis, Minnesota. Association for Computational Linguistics.

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An entity-driven framework for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3280–3291, Hong Kong, China. Association for Computational Linguistics.

Simeng Sun and Ani Nenkova. 2019. The feasibility of embedding based automatic evaluation for single document summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1216–1221, Hong Kong, China. Association for Computational Linguistics.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Stratos Xenouleas, Prodromos Malakasiotis, Marianna Apidianaki, and Ion Androutsopoulos. 2019. SUM-QE: a BERT-based summary quality estimation model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6005–6011, Hong Kong, China. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in*

Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Markus Zopf. 2018. Estimating summary quality with pairwise preferences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1687–1696, New Orleans, Louisiana. Association for Computational Linguistics.

A Interface Screenshots

We show screenshots of the instructions for both annotation methods and tasks in Figure 8 and interfaces in Figure 9.

B Survey

B.1 Categories

While most categories are self-explanatory, we elaborate on some of the decisions we made during the survey in this section.

Evaluation Questions. We allow a single study to include multiple evaluation questions, as long as all questions are answered by the same annotators and use the same method. We make no distinction between informativeness, coverage, focus and relevance and summarize them under *Content*. Similarly, we summarize fluency, grammaticality and readability under *Fluency*. *Other* includes:

- One study with a specialized set of evaluation questions evaluating the usefulness of a generated related work summary
- One study of *polarity* in a sentiment summarization context
- One study where annotators were asked to identify the aspect a summary covers in the context of review summarization
- Two studies evaluating formality and *meaning similarity* of reference and system summary
- One study evaluating diversity
- One study conducting a Turing test
- One study asking paper authors whether they would consider a sentence part of a summary of their own paper.
- One study evaluating structure and topic diversity

Evaluation Method. *Binary* includes any task with a yes/no style decision, while *pairwise* includes any method in which two systems are ranked against each other. *Other* includes

- The aspect identification task mentioned above
- One study in which participants selected a single best summary out of a set of summaries

Annotator Recruitment. *Other* includes any recruitment strategy that does not rely on crowdsourcing. This includes cases in which the recruitment was not specified, students, experts, the authors themselves and various kinds of volunteers.

Statistical Evaluation. *Other/unspecified* includes

- Four studies which reported statistical significance without reporting the test used
- Two studies using the approximate randomization test
- One study using the chi-square test
- One study using a Tukey test without prior ANOVA.

B.2 Survey Files

All papers we considered for the survey are listed in the supplementary material in the file `all_papers.yaml` by their id in the ACL anthology bib-file. The 58 SDS/MDS system papers that contain new human evaluation studies and are thus included in the survey are listed in the category `with_human_eval`.

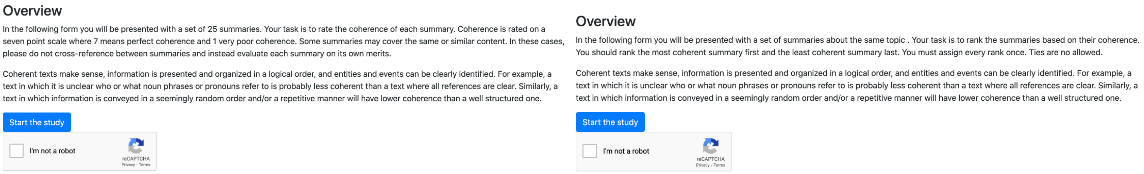
For the sake of completeness, we further list summarization papers we did not include in our survey. We separate them into the following categories:

no_human_eval 47 SDS/MDS system papers without human evaluation

sentsum 27 Sentence summarization and headline generation papers

non_system 34 summarization papers that do not introduce new systems, like surveys, opinion pieces and evaluation studies

other 10 Papers that conduct summarization with either non-textual input or non-textual output


Overview

In the following form you will be presented with a set of 25 summaries. Your task is to rate the coherence of each summary. Coherence is rated on a seven point scale where 7 means perfect coherence and 1 very poor coherence. Some summaries may cover the same or similar content. In these cases, please do not cross-reference between summaries and instead evaluate each summary on its own merits.

Coherent texts make sense, information is presented and organized in a logical order, and entities and events can be clearly identified. For example, a text in which it is unclear who or what noun phrases or pronouns refer to is probably less coherent than a text where all references are clear. Similarly, a text in which information is conveyed in a seemingly random order and/or a repetitive manner will have lower coherence than a well structured one.

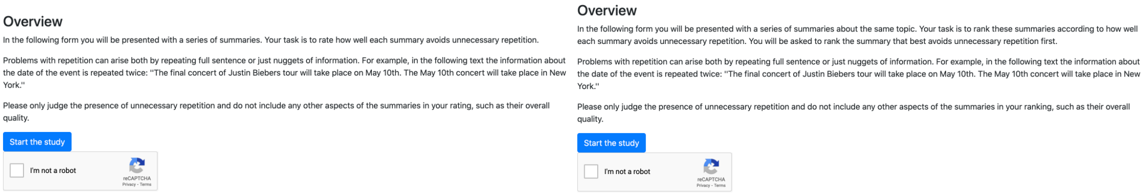
Overview

In the following form you will be presented with a set of summaries about the same topic . Your task is to rank the summaries based on their coherence. You should rank the most coherent summary first and the least coherent summary last. You must assign every rank once. Ties are no allowed.

Coherent texts make sense, information is presented and organized in a logical order, and entities and events can be clearly identified. For example, a text in which it is unclear who or what noun phrases or pronouns refer to is probably less coherent than a text where all references are clear. Similarly, a text in which information is conveyed in a seemingly random order and/or a repetitive manner will have lower coherence than a well structured one.

(a) `Likert` - Coherence

(b) `Rank` - Coherence


Overview

In the following form you will be presented with a series of summaries. Your task is to rate how well each summary avoids unnecessary repetition.

Problems with repetition can arise both by repeating full sentence or just nuggets of information. For example, in the following text the information about the date of the event is repeated twice: "The final concert of Justin Biebers tour will take place on May 10th. The May 10th concert will take place in New York."

Please only judge the presence of unnecessary repetition and do not include any other aspects of the summaries in your rating, such as their overall quality.

Overview

In the following form you will be presented with a series of summaries about the same topic. Your task is to rank these summaries according to how well each summary avoids unnecessary repetition. You will be asked to rank the summary that best avoids unnecessary repetition first.

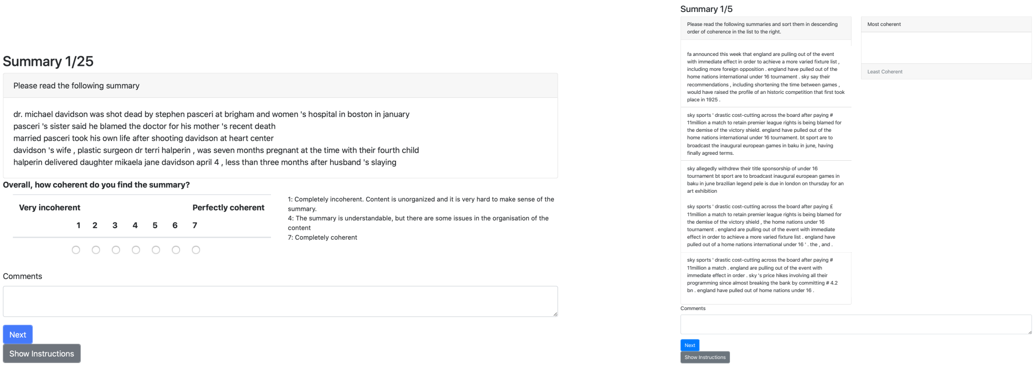
Problems with repetition can arise both by repeating full sentence or just nuggets of information. For example, in the following text the information about the date of the event is repeated twice: "The final concert of Justin Biebers tour will take place on May 10th. The May 10th concert will take place in New York."

Please only judge the presence of unnecessary repetition and do not include any other aspects of the summaries in your ranking, such as their overall quality.

(c) `Likert` - Repetition

(d) `Rank` - Repetition

Figure 8: Screenshots of the Annotator Instructions.


Summary 1/25

Please read the following summary

dr. michael davidson was shot dead by stephen pasceri at brigham and women 's hospital in boston in january
pasceri 's sister said he blamed the doctor for his mother 's recent death
married pasceri took his own life after shooting davidson at heart center
davidson 's wife , plastic surgeon dr terri halperin , was seven months pregnant at the time with their fourth child
halperin delivered daughter mikaela jane davidson april 4 , less than three months after husband 's slaying

Overall, how coherent do you find the summary?

Very incoherent **Perfectly coherent**
1 2 3 4 5 6 7
○ ○ ○ ○ ○ ○ ○

1: Completely incoherent. Content is unorganized and it is very hard to make sense of the summary.
4: The summary is understandable, but there are some issues in the organisation of the content
7: Completely coherent

Comments

Summary 1/5

Please read the following summaries and sort them in descending order of coherence in the list to the right.

fa announced this week that england are pulling out of the event with immediate effect in order to achieve a more varied fixture list , including more foreign opposition . england have pulled out of the home nations international under 16 tournament . sky say their recommendations , including shortening the time between games , would have raised the profile of an historic competition that first took place in 1925 .

sky sports ' drastic cost-cutting across the board after paying # 11million a match to retain premier league rights is being blamed for the demise of the victory shield. england have pulled out of the home nations international under 16 tournament. bt sport are to broadcast the inaugural european games in baku in june, having finally agreed terms.

sky allegedly withdrew their title sponsorship of under 16 tournament bt sport are to broadcast inaugural european games in baku in june brazilian legend pele is due in london on thursday for an art exhibition

sky sports ' drastic cost-cutting across the board after paying £ 11million a match to retain premier league rights is being blamed for the demise of the victory shield , the home nations under 16 tournament . england are pulling out of the event with immediate effect in order to achieve a more varied fixture list . england have pulled out of a home nations international under 16 ' . the , and .

sky sports ' drastic cost-cutting across the board after paying # 11million a match . england are pulling out of the event with immediate effect in order . sky 's price hikes involving all their programming since almost breaking the bank by committing # 4.2 bn . england have pulled out of home nations under 16 .

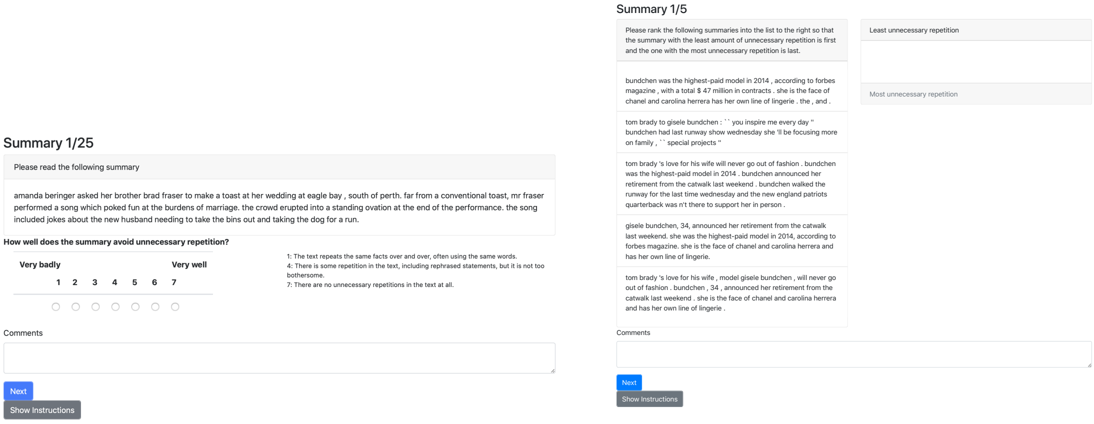
Most coherent

Least Coherent

Comments

(a) `Likert` - Coherence

(b) `Rank` - Coherence


Summary 1/25

Please read the following summary

amanda beringer asked her brother brad fraser to make a toast at her wedding at eagle bay , south of perth. far from a conventional toast, mr fraser performed a song which poked fun at the burdens of marriage. the crowd erupted into a standing ovation at the end of the performance. the song included jokes about the new husband needing to take the bins out and taking the dog for a run.

How well does the summary avoid unnecessary repetition?

Very badly **Very well**
1 2 3 4 5 6 7
○ ○ ○ ○ ○ ○ ○

1: The text repeats the same facts over and over, often using the same words.
4: There is some repetition in the text, including rephrased statements, but it is not too bothersome.
7: There are no unnecessary repetitions in the text at all.

Comments

Summary 1/5

Please rank the following summaries into the list to the right so that the summary with the least amount of unnecessary repetition is first and the one with the most unnecessary repetition is last.

bundchen was the highest-paid model in 2014 , according to forbes magazine , with a total $ 47 million in contracts . she is the face of chanel and carolina herrera has her own line of lingerie . the , and .

tom brady to gisele bundchen : `` you inspire me every day ''  bundchen had last runway show wednesday she 'll be focusing more on family , `` special projects ''

tom brady 's love for his wife will never go out of fashion . bundchen was the highest-paid model in 2014 . bundchen announced her retirement from the catwalk last weekend . bundchen walked the runway for the last time wednesday and the new england patriots quarterback was n't there to support her in person .

gisele bundchen, 34, announced her retirement from the catwalk last weekend. she was the highest-paid model in 2014, according to forbes magazine. she is the face of chanel and carolina herrera and has her own line of lingerie.

tom brady 's love for his wife , model gisele bundchen , will never go out of fashion . bundchen , 34 , announced her retirement from the catwalk last weekend . she is the face of chanel and carolina herrera and has her own line of lingerie .

Least unnecessary repetition

Most unnecessary repetition

Comments

(c) `Likert` - Repetition

(d) `Rank` - Repetition

Figure 9: Screenshots of the Annotation Interfaces.

We give a full list of the survey results for all papers with human evaluation studies in the file `survey_details.csv`. The file has the following columns:

paper Id of the paper in the ACL anthology

eval_id Id of the evaluation study to differentiate them in papers with multiple studies

task Summarization task of the paper: SDS vs. MDS

genre Genre of the summarized documents

#docs Number of documents in the evaluation

#systems Number of systems in the evaluation

includes_reference Whether the reference summary is included in the human evaluation

#ann_total Total number of annotators in the study

#ann_item Number of annotators per summary

content, fluency, repetition, coherence, referential_clarity, other, overall Binary columns indicating evaluation questions in the paper

measure Annotation method used in the study

anntype Annotator recruitment strategy

stattest Statistical test used

design_specified Indicates whether it is possible to determine the full design from the information given about the study in the paper

comments Comment column. This column describes the use of *other* where present.

Open-Mindedness and Style Coordination in Argumentative Discussions

Aviv Ben Haim
Department of Software and
Information Systems Engineering
Ben-Gurion University
avivbenh@post.bgu.ac.il

Oren Tsur
Department of Software and
Information Systems Engineering
Ben-Gurion University
orentsur@bgu.ac.il

Abstract

Linguistic accommodation is the process in which speakers adjust their accent, diction, vocabulary, and other aspects of language according to the communication style of one another. Previous research has shown how linguistic accommodation correlates with gaps in the power and status of the speakers and the way it promotes approval and discussion efficiency. In this work, we provide a novel perspective on the phenomena, exploring its correlation with the open-mindedness of a speaker, rather than to her social status. We process thousands of unstructured argumentative discussions that took place in Reddit's Change My View (CMV) subreddit, demonstrating that open-mindedness relates to the assumed role of a speaker in different contexts. On the discussion level, we surprisingly find that discussions that reach agreement present lower levels of accommodation.

1 Introduction

1.1 Linguistic Accommodation

Accommodation in personal communication refers to the unconscious process in which a speaker changes (accommodates) her communicative behavior with respect to the communication partner. The change can be manifested across many dimensions, e.g., posture (Condon and Ogston, 1967), nodding (Hale and Burgoon, 1984), pauses (Jaffe and Feldstein, 1970), and linguistic style (Niederhoffer and Pennebaker, 2002). From the perspective of sociolinguistic and social-psychology, accommodation is argued to increase cognitive efficiency (Street and Giles, 1982), provide approval and validation (Giles, 2008), and project the speaker's positive image (Infante et al., 1997).

In this work we focus on the accommodation of the linguistic style – the usage of stylistic markers and function words, such as auxiliary verbs and prepositions. Tausczik and Pennebaker (2010) estimated that style and function words make up for 55% of the words we use. These markers shape the conversation regardless of its topical content. Analysis of function-word usage is common in many NLP tasks, e.g., gender detection (Koppel et al., 2002; Mukherjee and Liu, 2010; Bortolato, 2016), forensic linguistics (Juola, 2008; Boukhaled and Ganascia, 2015; Kestemont, 2014) and personality type detection (Argamon et al., 2005, 2009; Litvinova et al., 2016), among others.

Accommodation is not necessarily a symmetric process. One party in a communication can accommodate while the other party can either accommodate as well (converge), sustain her behavior, or actively diverge. Linguistic style accommodation was first studied quantitatively by Niederhoffer and Pennebaker (2002), analyzing the linguistic style accommodation in a small scale dataset. Large scale datasets and advanced statistical and algorithmic methods were explored by Danescu-Niculescu-Mizil et al. (2011); Guo et al. (2015); Muir et al. (2017), among others. Differences in the social status of the speakers were found to dominate accommodation (Danescu-Niculescu-Mizil et al., 2012).

In this paper we explore linguistic accommodation from a novel perspective. We argue that the speaker's *open-mindedness* drives her linguistic coordination. While the speaker's open-mindedness can be "socially forced" by power relations – it is also inherent to one's character and her actively assumed social role.

1.2 Open Mindedness

"To be open-minded is... to be critically receptive to alternative possibilities, to be willing to think again despite having formulated a view, and to be concerned to defuse any factors that constrain one's thinking in predetermined ways" (Hare, 2003).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1876–1886
April 19 - 23, 2021.

Open-mindedness is closely related to the concepts of dialogism (Bakhtin, 1981) and the development of dialogic agency (Parker, 2006), vital in a liberal and multi-cultural society. Open-mindedness is found to help in conflict resolution (Tjosvold and Morishima, 1999; Tjosvold and Poon, 1998), boost creativity (Mitchell and Nicholas, 2006; Keskin, 2006), increase rationality and neutrality, and play a significant role in the theory of education (Hare, 1993).

From a practical perspective, the significant role open-mindedness plays in a range of situations, from the negotiation table to political debates and the classroom, requires an efficient way to detect open-mindedness or the lack-of, allowing intervention by a moderator or a teacher (Zakharov et al., 2020). We show the level accommodation is simple to compute and can be used as a proxy for open-mindedness.

1.3 Reddit's CMV

The Change My View (CMV) is a subreddit (forum) on the forum-based Reddit platform. The forum is self described as *"A place to post an opinion you accept may be flawed, in an effort to understand other perspectives on the issue. Enter with a mindset for conversation, not debate."*[1]. Each discussion thread in CMV evolves around the topic presented in the submission by the Original Poster (OP). A discussion, therefore, takes the structure of a tree with the *submission* at the root and the various comments (replies) stemming from it. One unique feature of the CMV subreddit is the *Delta* Δ – a way to acknowledge a convincing argument. A Delta can be awarded by any user, OP or not, to a comment made by any other user, and should be explicitly justified. A user can award a Delta only to users holding an opposing view to her own. A delta can signify "good point" and does not have to reflect a complete reversal of opinion. CMV is heavily moderated to maintain a high level of discussion and to ensure that Deltas are awarded with proper reasoning. Table 1 provides a partial and truncated example of a submission from CMV.

Open-mindedness in CMV We use CMV discussions to study the relation between linguistic accommodation and open-mindedness. While open-mindedness is an informal prerequisite of all CMV participants, we use two explicit indications of open-mindedness: (i) Being an OP – Stating an opinion and literally declaring the willingness to change a presupposition, and (ii) The use of Δ, explicitly acknowledging a convincing argument made by an opposing party.

It could be argued that the OPs are not necessarily open-minded and only respond to people that agree with them. Fortunately, CMV rules strive to create a constructive conversation and CMV is a heavily moderated subreddit. The rule states an OP is obliged to reply to opposing views and show open-mindedness.

Dataset Our corpus consists of 9,374 English discussions initiated by 4873 OPs between January 2018 and June 2020. It contains 1,301,545 posts (utterances) by 86,941 unique users. 8,659 Deltas were awarded, 877 of them by users that are not the OP. The mean number of speakers in each submission is 42, max and min are 898 and 3 respectively, and the median is 26. The mean number of comments in each submission is 138, max and min being 2559 and 26 respectively, and the median is 88. The mean numbers of word in CMV comment is 83 and the median is 50.

1.4 Research Questions

Based on the indications described above, we explore the relations between open-mindedness and accommodation through the following research questions:

RQ1: Do OPs present higher levels of linguistic accommodation? By definition, OPs are open to change their view, hence open-minded. We expect to observe higher levels of linguistic coordination presented by the OP of a discussion than by other participants.

RQ2: Do Delta givers present higher levels of linguistic accommodation? Delta givers explicitly express open-mindedness by awarding a Δ upon recognizing convincing argumentation. We expect to observe higher levels of linguistic coordination presented by the Delta givers than by other participants. In the scope of this research question, we will focus on delta givers that are not the OPs of the discussion in which they awarded the Delta. By doing so we examine delta givers that unlike OPs, have no status – they are not the initiators of the discussion, are not entitled to give the 'final verdict', and can choose to opt-in or out. Since previous work explains accommodation in terms of

---

[1] https://www.reddit.com/r/changemyview/wiki/index (accessed Oct. 5 2020)

Speaker	Text	Gave delta
A	**CMV: Money needs to get out of politics** I am genuinely curious about the counter arguments as I haven't heard any and I want to know if I'm wrong. This began with this case: Buckley v. Valeo, 424 U.S. 1 (1976), was a landmark decision of the US Supreme Court on campaign finance...	
B	> ...There's a lot of attention given to the massive amount of money that come from wealthy individuals or special interest groups but ordinary people can make campaign contributions to the candidates they support, too. Bernie Sanders only accepted donations from small-dollar donors and outspent Joe Biden in primary at a whopping $200 million...	
A	>> You gave a good counter argument. I meant more big money, but I see it's hard to separate the two. Δ	Yes
C	> There have actually been a lot of studies suggesting that political donations don't swing votes...	

Table 1: Partial and truncated example of a submission from CMV

power differences, delta givers serve as a control group, lacking any formal power or privilege.

RQ3: Are Deltas correlated with accommodation on the *discussion level*, rather than on the user level? We hypothesize that Delta awards may be correlated with the general accommodation observed in a discussion, rather with the open-mindedness of a specific user – the Delta giver. In order to test this hypothesis we compare the accommodation of all users in branches (of the discussion tree) in which Deltas were awarded, to the accommodation in branches in which no Delta was awarded.

The formal definitions of user, group and aggregate coordination are presented in Section 3.1, and the hypotheses promoted by research questions above are formally defined in Section 3.2 and tested in Section 4. Further analysis and refined hypotheses are proposed and tested in Section 4.2.

2 Related Work

Linguistic coordination, sometimes referred to as 'accommodation' or 'matching', is mainly studied from two perspectives – social (-psychology) and computational. Linguistic style matching in social interactions and its correlation to various psychometric properties of language was first studied by Niederhoffer and Pennebaker (2002). This approach was later generalized to discussions in small groups by (Gonzales et al., 2010). These metrics were applied to social media data by Danescu-Niculescu-Mizil et al. (2011).

Mukherjee and Liu (2012) analyzed a large corpus of online debate, reporting that debaters using agreement expressions (e.g., 'I agree with that point', 'rightly said') as a debate strategy show higher linguistic accommodation than debaters that explicitly express disagreement (e.g., 'I don't buy it', 'I disagree', 'nonsense'). However, the use of agreement/disagreement expressions in a formal debate setting is a strategic choice and does not imply actual agreement or open-mindedness. Moreover, the stated objective in a formal debate is winning the debate, rather than self-reflection through convincing argumentation.

Power differences were found to drive linguistic coordination toward the speaker with the higher status (Danescu-Niculescu-Mizil et al., 2012), and a particular configurations of personal traits (e.g., agreeableness, Machiavellianism, low self-consciousness) combined with low-power position were shown to increase the likelihood of coordination (Muir et al., 2016, 2017). Guo et al. (2015) proposed Bayesian inference approach to capture influence manifested through linguistic accommodation.

This work is the first to explore coordination through the lens of open-mindedness, rather than through self-awareness, power, influence or strategic manipulation.

3 Measuring Linguistics Accommodation

3.1 Linguistic Style Coordination

We adopt the common coordination metric used in previous studies by Danescu-Niculescu-Mizil et al. (2011, 2012); Mukherjee and Liu (2012) and others. The metric quantifies the accommodation of an individual along a linguistic category by looking at the shifting usage of terms of that category, compared to other speakers. The linguistic categories include Articles, Adverbs, Quantifiers, Conjunctions, Indefinite Pronouns, Personal Pronouns, Prepositions and Auxiliary Verbs as well as other psycho-linguistic categories from the widely used LIWC dictionary (J. W. Pennebaker, 2015).

Speaker's accommodation Given users a and b, and a linguistic marker m, we want to measure how the usage of m by a triggers (or suppresses) the occurrence of m in the direct responses of b.

Given a set $S_{a,b} : (a : u_1, b : u_2)$ of exchanges where u_1 denotes an utterance (comment) by a and u_2 is a direct reply by b we define coordination of b towards a on marker m as:

$$C^m(b,a) = P(\xi^m_{u_2\to u_1} | \xi^m_{u_1}) - P(\xi^m_{u_2\to u_1}) \quad (1)$$

Where $P(\xi^m_{u_1})$ is the probability of speaker a using marker m; $P(\xi^m_{u_2\to u_1})$ is the probability of speaker b using m in replying to u_1 (made by a), and $P(\xi^m_{u_2\to u_1} | \xi^m_{u_1})$ is the probability of speaker b using m in replying to an utterance u_1 in which a also used m. Equation 1 bears three important properties:

1. It is asymmetric, that is $C^m(b,a) \neq C^m(a,b)$, allowing us to realize which user is accommodating toward the other.
2. $C^m(b,a) = 0$ if any of the speakers uses m in every utterance, since no turn-taking influence could be modeled in this scenario.
3. $C^m(b,a) \xrightarrow[|S^{\neg m}_{a,b}|\to\infty]{} 1$ if b uses m in a response to a *if and only if* a used m, where $S^{\neg m}_{a,b}$ is the set of exchanges between a and b the do not contain m.

Group Coordination The definition of coordination is extended to address the coordination of a group of speakers. This enables us to measure (i) The coordination of a speaker b toward a group of speakers A, and (ii) The coordination of a group B toward a group of speakers A.

We accomplish (i) by simply looking at the set of consecutive utterances: $S_{A,b} : (A : u_1, b : u_2)$ that includes the responses of speaker b to every speaker $a \in A$. The coordination of b toward the group A is therefore defined as:

$$C^m(b,A) = P(\xi^m_{u_2\to u_1} | \xi^m_{u_1}) - P(\xi^m_{u_2\to u_1}) \quad (2)$$

To accomplish (ii), the coordination of speakers in group B to targets in group A, we calculate the average coordination of speakers in B toward targets speakers A:

$$C^m(B,A) = \langle C^m(b,A) \rangle_{b\in B} \quad (3)$$

Aggregated coordination The coordination measures defined above apply to a specific marker m. Previous studies report that accommodation is not necessarily correlated across markers as user b can accommodate toward a on m and diverge on m' (Street and Giles, 1982; Bilous and Krauss, 1988). Therefore, having an aggregated coordination metric may provide a fuller picture regarding the overall accommodation across markers M and groups. Three types of *aggregated* coordination measure are considered:

1. *Aggregate 1*: We first compute $C(b,A)$ as the macro average of $C^m(b,A)$ for all $m \in M$, then use it to compute $C(B,A)$ as described in Equation 3. However, we wish to note that $C^m(b,A)$ can only be computed if there is a sufficient number of exchanges between b and A with the marker m (we enforce $\delta > 3$), and therefore, *Aggregate 1* is calculated *only* using those $b \in B$ where $C^m(b,A)$ is defined for every m. This aggregation provides the most accurate generalized coordination, but relies only on a subset of the speakers. In order to compensate for cases in which $C^m(b,A)$ is undefined we propose the two aggregated measures below.

2. *Aggregate 2*: Assuming group homogeneity, we approximate $C^m(b,A)$ by averaging $C^m(B,A)$ over all $b \in B$ for whom $C^m(b,A)$ is defined.

3. *Aggregate 3*: Assuming speakers tend to present the same behaviour across markers, the undefined $C^m(b,A)$ can be approximated by averaging $C^{m'}(b,A)$ over all $m' \in M$ for which $C^{m'}(b,A)$ could be computed directly. While this assumption does not always hold, this definition of aggregated coordination has proved useful in Danescu-Niculescu-Mizil et al. (2012).

We used the ConvoKit toolkit for conversation analysis (Chang et al., 2020) in preprocessing and measuring coordination.

Notation	Definition	Used in
U	The set of all users (speakers) in the data	all
G^{OPs}	The set of all OPs	$H_{1.1}$, $H_{1.2}$, $H_{4.3}$
$G^{\overline{OPs}}$	The set of all users in a submission that are not the OP (AKA regulars)	$H_{1.1}$, $H_{4.3}$
$G^{\overline{r}}$	Regulars users in a submission that are OPs in other submissions	$H_{1.2}$
$G^{\Delta_{reg}}$	The set of Delta givers $\notin G^{OPs}$. That is: Delta givers that are not the OP of the submission	H_2, $H_{4.1}$
$G^{\Delta_{OP}}$	The set of Delta givers $\in G^{OPs}$. That is: OPs that gave Delta in their own submission	$H_{4.2}$
$G^{\overline{\Delta}}$	The set of users that did not award a Δ in the submission	H_2, H_4, $H_{4.1}$
$C^{\Delta}(\circ,\circ)$	Coordination is measured only for discussion branches in which a Δ was awarded	H_3
$C^{\overline{\Delta}}(\circ,\circ)$	Coordination is measured only for discussion branches in which a Δ was *not* awarded	H_3
G^{Δ}	The set of Delta givers in a submission regardless of role	H_4
$G^{\overline{OP_{\Delta}}}$	The set of OPs that did not give delta in their own submission	$H_{4.2}$

Table 2: Notations and definitions for groups of speakers assuming different roles in the CMV data.

3.2 Formalizing the Research Questions

Notation Before using the definitions above to formalize the research questions presented in Section 1.4, we introduce a few more notations, presented in Table 2 for convenience. An important aspect to note is that for each Reddit user a, we produce dummy users for each submission she takes part in. Formally: a user a participating in n discussions will have n dummy users $\{a_i\}_{i=1\ldots n}$, one for each discussion. Different dummy representations of a user may be part of different groups, for example, in a discussion she initiated: $a_i \in G^{OPs}$ and in another discussion she participates (but didn't initiated) $a_j \in G^{\overline{OPs}}$. These dummy representations allows us to explore the coordination of a user in different contexts.

We provide the following example of a conversation to familiarize with the notations: Given a submission opened by Bobby, comments by Jess, Arnold and a delta given by both Bobby and Ava. We can attribute our speakers to the groups accordingly: Bobby is in the group of OPs, the group of delta givers and the group of OPs that gave delta - $Bobby_1 \in G^{OPs}, G^{\Delta}, G^{\Delta_{OP}}$. Jess and Arnold are in the group of users that did not give delta - $\{Jess_1, Arnold_1\} \subseteq G^{\overline{\Delta}}$. Ava is in the group of delta givers and the group of delta givers that are not OPs - $Ava_1 \in G^{\Delta}, G^{\Delta_{reg}}$.

RQ1: Do OPs present higher levels of linguistic accommodation? In order to answer positively we need to reject the null hypothesis:

$$C(G^{OPs}, U) = C(G^{\overline{OPs}}, U) \qquad (H_{1.1})$$

and verify that $C(G^{OPs}, U) > C(G^{\overline{OPs}}, U)$, showing that coordination of OPs (in discussions they initiated) towards all other users is higher than the coordination of regular users toward all other users.

We further explore the behaviour of OPs putting forward the following null hypothesis:

$$C(G^{OPs}, U) = C(G^{\overline{r}}, U) \qquad (H_{1.2})$$

Namely, we test whether OPs present different levels of coordination in discussions they initiated and in discussions they participate without assuming the role of the OP.

RQ2: Do Delta givers present higher levels of linguistic accommodation? A positive answer is validated by rejecting the following null hypothesis:

$$C(G^{\Delta_{reg}}, U) = C(G^{\overline{\Delta}}, U) \qquad (H_2)$$

and verifying that $C(G^{\Delta_{reg}}, U) > C(G^{\overline{\Delta}}, U)$.

It is important to note that $G^{\Delta_{reg}}$ only includes Δ-givers that are not OPs (see definitions in Table 2). This distinction is important due to the fact that a user in an OP role declares her flexibility by inviting other users to change her mind, while other Δ-givers do not assume this flexibility.

RQ3: Are Deltas correlated with accommodation on the discussion level, rather than on the user level? That is we check whether Deltas are awarded more generously in discussions with higher levels of accommodation, not necessarily by or toward the Δ-giver. We put forward the following null hypothesis:

$$C^{\Delta}(U,U) = C^{\overline{\Delta}}(U,U) \qquad (H_3)$$

Validating H_3 shows that the general accommodation level is not correlated with Δ awards.

4 Results and Analysis

All of the hypothesis presented above are tested using the two-sample t-test. We report significance values for the following p-values: 0.05, 0.01 and 0.001. Conversations are extracted from the discussion data described in Section 1.3, containing 1.3M utterances by 87K unique users participating in 9.3K discussions.

4.1 Open-mindedness

OPs present higher coordination (RQ1) Figure 1 presents the coordination values for a number of marker types and for the three aggregate measures. We reject the null hypothesis $H_{1.1}$ with $p < 0.001$ for all markers and measures and confirm that OPs present significantly higher levels of coordination, compared to regular users (Figure 1a). Similarly, we reject the null hypothesis $H_{1.2}$, though significance values vary between markers (Figure 1b).

The results above show that the open-mindedness of an OP is not an inherent personal capacity but depends on the role assumed in a specific context. We further validated this by testing the coordination of OPs in discussions they did not initiate ($G^{\bar{r}}$) to the coordination level of users that never initiated a discussion as OPs. Indeed, we find no significant difference between their coordination levels.

Delta givers present higher coordination (RQ2) We reject the null hypothesis (H_2) for most markers and all aggregate types (Figure 2). This result confirms that non-OP Delta givers present higher levels of coordination compared to users that did not give Delta.

Coordination varies on branch level (RQ3) We reject the null hypothesis (H_3) for most markers and all aggregate measures, asserting that coordination levels in branches in which a Δ was awarded are significantly different from those in branches without any Δ. Surprisingly, we find that the levels of accommodation are higher in branches where no Delta was awarded (see figure 3). In the next subsection we further explore this surprising result.

4.2 Accommodation Effect on Delta Giving

In the previous section we have established that on the user-level, Δ-givers present higher coordination than other users, however, on the discussion level, Δ is more likely to be awarded in non-coordinated branches. In order to understand these dynamics, we test the coordination *toward* Δ-givers (G^{Δ}). We test the following hypothesis:

$$C(U, G^{\Delta}) = C(U, G^{\overline{\Delta}}) \qquad (H_4)$$

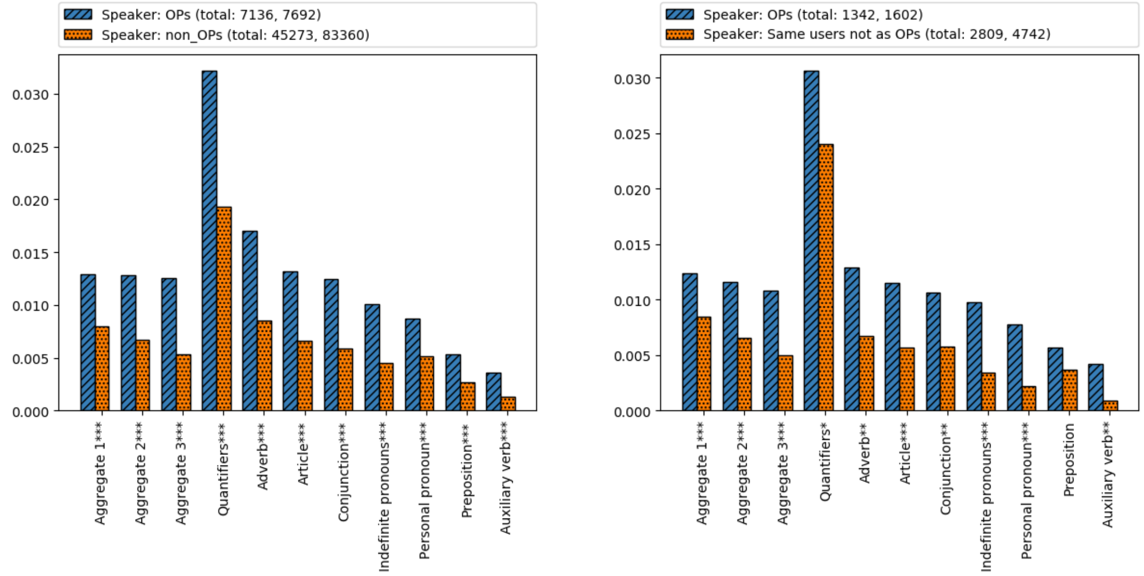
It is important to note that C is calculated only on utterances made *before* the Δ was awarded. Using the two-sample t-test we get mixed results (Figure 4). Significance is observed for some markers (quantifiers, adverbs, prepositions and personal pronouns) and for Aggregate 1 and 2, but not for Aggregate 3 and the other markers that proved to be highly distinctive in RQ1-3.

Following the inconclusive result for H_4, we divide G^{Δ}, the group of Δ-givers, to two distinct sets – OPs ($G^{\Delta_{OP}}$) and Δ-givers that are not OPs ($G^{\Delta_{reg}}$). We can now test whether Δ-giving OPs respond differently than other Δ-givers when *accommodated to*. The idea behind this split is to find out if one group is more susceptible to accommodation than the other. We consider two settings: (i) Accommodation toward non-OP Δ-givers ($G^{\Delta_{reg}}$) vs. accommodation toward non-Δ-givers ($G^{\overline{\Delta}}$), for which we test $H_{4.1}$, and (ii) Accommodation toward Δ-giving OPs ($G^{\Delta_{OP}}$) vs. the accommodation toward OPs that did not award a Δ ($G^{\overline{OP_{\Delta}}}$), for which we test $H_{4.2}$. In both settings we calculate the coordination on utterances made *before* the Δ was awarded, similar to (H_4).

$$C(U, G^{\Delta_{reg}}) = C(U, G^{\overline{\Delta}}) \qquad (H_{4.1})$$

$$C(U, G^{\Delta_{OP}}) = C(U, G^{\overline{OP_{\Delta}}}) \qquad (H_{4.2})$$

While we cannot reject $H_{4.1}$ (Figure 5a), we do reject $H_{4.2}$ (Figure 5b), as significance ($p < 0.001$) is observed for all markers and aggregate measures but for adverbs. That is, while regular Δ-givers do not experience any special accommodation, we do observe a dramatic difference in accommodation patterns toward Δ-giving OPs vs. OPs that did not award a Δ – OPs are inclined to award a Δ in discussion threads that are accommodating toward them, and reserve themselves from awarding a Δ in case the use of style markers diverges over time, rather than coordinated toward them.



(a) OP vs non-OP coordination to others. Testing ($H_{1.1}$)

(b) OP vs OP as regular participants coordination to others. Testing ($H_{1.2}$)

Figure 1: Parentheses in figure legends denote (number of speakers used in type 1 aggregation, number of speakers used in aggregations of type 2 and 3). Asterisks denote significance using independent t-test: */**/*** for $p < 0.05/0.01/0.001$, respectively.

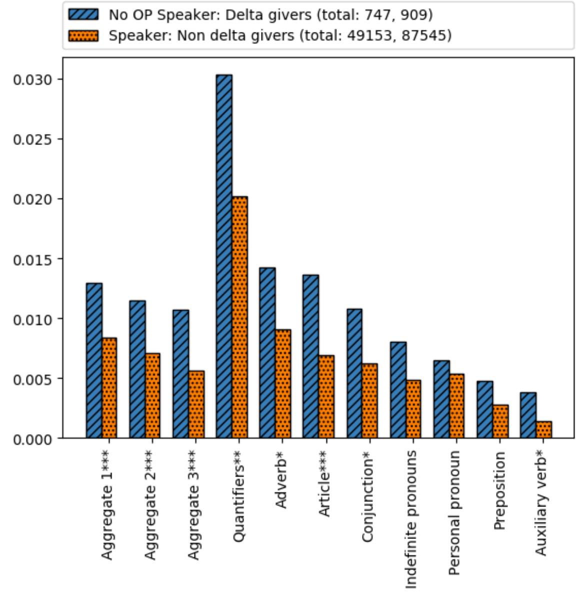


Figure 2: Non-OP Delta givers vs Users that did not give Delta. Test (H_2)

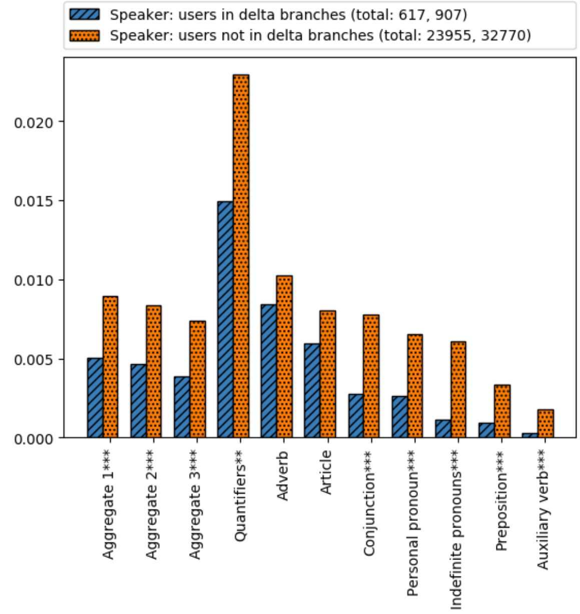


Figure 3: Delta branches vs Non-Delta branches. Test (H_3)

Finally, we complement this analysis by looking at accommodation toward OPs in their own submissions (G^{OPs}) and non-OPs ($G^{\overline{OPs}}$). The null hypothesis, asserting that OPs do not experience higher levels of accommodation toward them is formalized as:

$$C(U, G^{OPs}) = C(U, G^{\overline{OPs}}) \qquad (H_{4.3})$$

This hypothesis is rejected with significance of $p < 0.001$ for all markers. However, we see that users tend to diverge from the OPs, rather than accommodate toward them (Figure 6). This result

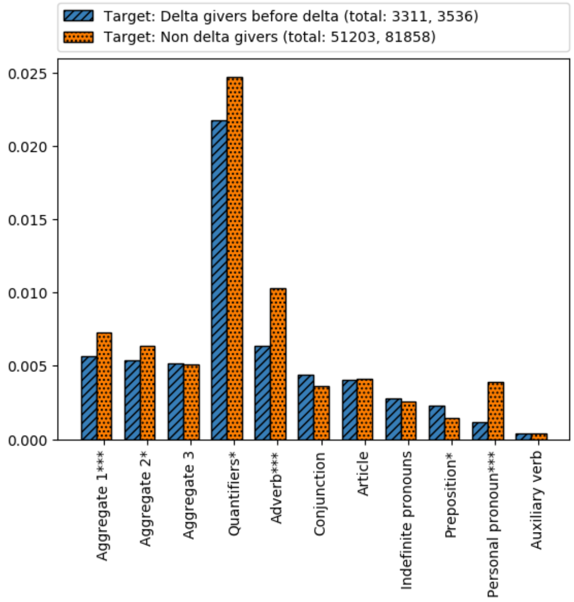


Figure 4: Accommodation towards delta givers vs Non-Delta givers. Test (H_4)

coupled with the results of H_4, $H_{4.1}$ and $H_{4.2}$ highlights the importance of accommodation to achieve persuasion, as we briefly discuss on below.

4.3 Discussion

Open-mindedness or power and status Previous work promoted the social framework of power-relations as the force behind linguistic style accommodation. Our results suggest that accommodation is related to the willingness of an individual to examine her or his beliefs, consider alternative viewpoints, appreciate convincing argumentation, and ultimately "convert". This correlation between open-mindedness and accommodation is observed for OPs and Δ-givers alike – both present higher accommodation levels *toward* their conversation partners.

Dynamic Open-mindedness Analysing the accommodation of OPs toward others, we observed that they present significant accommodation in the discussions they initiated, but present accommodation levels similar to other users in the discussions they only take part of as regular users. These results suggest that open-mindedness can be context-dependent, rather than an inherent personal trait. This view could also accommodate (pun intended) previous works promoting the power-structure hypothesis, as the power difference implicitly imposes open-mindedness – the expected state-of-mind of a subordinate, given our social norms.

Accommodation on the discussion level Surprisingly we found that discussion convergence (reflected by a Δ awarded by the OP) does not positively correlate with accommodation. In fact, Δ-givers that are not OPs are more likely to award the Δ in discussions that are characterized by style *divergence*, rather than coordination. We hypothesize that Deltas serve a subtle, secondary, social function, beyond the acknowledgment of a convincing argument. Users may be inclined to award a Δ in a subconscious effort to ease tensions that are manifested by the divergence of style. This is in line with previous work by Mukherjee and Liu (2012), showing that strategic expressions of an agreement by debaters are correlated with high accommodation. Similarly, we observe the "strategic" impact of style coordination in promoting agreement – as OPs are more likely to award a Δ if they are accommodated to.

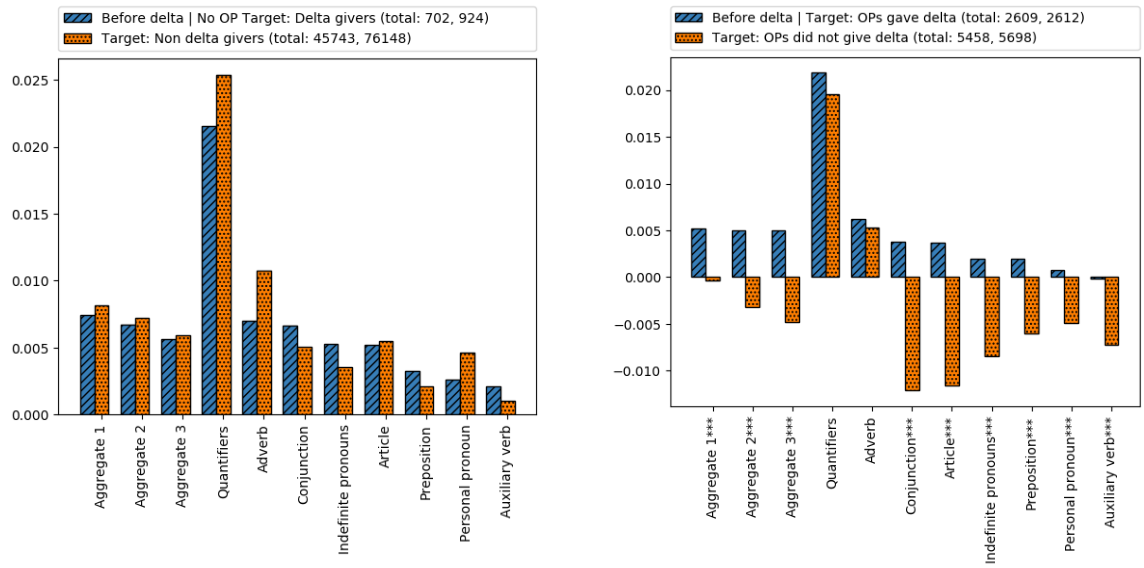
5 Conclusions and Future Work

5.1 Conclusions

We have proposed a novel hypothesis, asserting that linguistic style accommodation correlates with *open-mindedness*. We have provided evidence that supports this hypothesis through a series of experiments. We further demonstrated that open-mindedness is context-dependent, and argued that previous frameworks through which accommodation is studied could be viewed as a special case under the umbrella framework of open-mindedness we have proposed.

In future work we aim to further explore the different accommodation levels between some of the markers, and the way these are correlated with the conversational discourse structure, as proposed by (Zakharov et al., 2020).

Additionally, another area we believe is worth exploring and that is the effect of 'performance' as studied by Goffman (1967). OPs, being in a sort of formal role maybe "acting" and their accommodation might be altered for it. In this work we used the delta users as a control group that is not affected by this effect. Still, it is interesting to research how accommodation is affected when "acting" compared to genuine behavior. Further exploration of the shifting performative roles of the users, their dynamic roles, their assumed persona should be addressed in future work through a controlled experiment.



(a) Accommodation towards non-OPs that gave delta vs users that did not give Delta. Test ($H_{4.1}$)

(b) Accommodation towards OPs that gave delta vs OPs that did not give Delta. Test ($H_{4.2}$)

Figure 5: Breaking down RQ4 to two groups

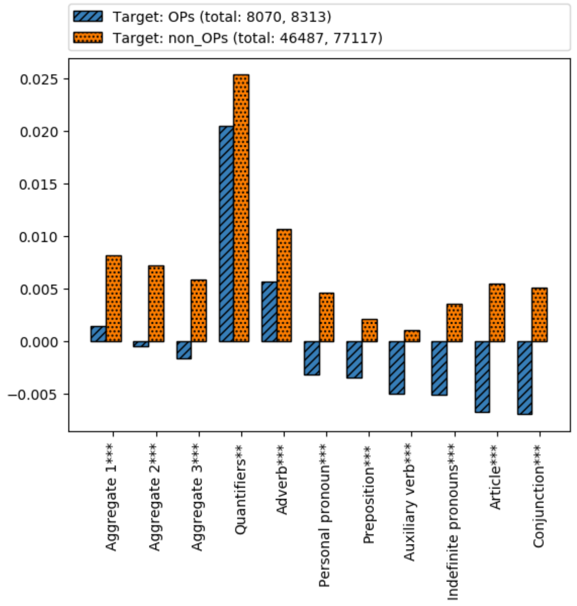


Figure 6: Accommodation towards OPs vs Non-OPs. Test ($H_{4.3}$)

Acknowledgments

We thank the anonymous reviewers for their valuable comments, and especially the suggestion to address accommodation and open-mindedness trough the framework of 'performance'.

References

Shlomo Argamon, Sushant Dhawle, Moshe Koppel, and James W Pennebaker. 2005. Lexical predictors of personality type. In *Proceedings of the 2005 Joint Annual Meeting of the Interface and the Classification Society of North America*, pages 1–16.

Shlomo Argamon, Moshe Koppel, James W Pennebaker, and Jonathan Schler. 2009. Automatically profiling the author of an anonymous text. *Communications of the ACM*, 52(2):119–123.

Mikhail M Bakhtin. 1981. The dialogic imagination: Four essays by mm bakhtin (m. holquist, ed.; c. emerson & m. holquist, trans.).

Frances R Bilous and Robert M Krauss. 1988. Dominance and accommodation in the conversational behaviours of same-and mixed-gender dyads. *Language & Communication*.

Claudia Bortolato. 2016. Intertextual distance of function words as a tool to detect an author's gender: A corpus-based study on contemporary italian literature. *Glottometrics*, 34:28–43.

Mohamed Amine Boukhaled and Jean-Gabriel Ganascia. 2015. Using function words for authorship attribution: Bag-of-words vs. sequential rules. *Natural Language Processing and Cognitive Science: Proceedings 2014*, 115.

Jonathan P Chang, Caleb Chiam, Liye Fu, Andrew Z Wang, Justine Zhang, and Cristian Danescu-Niculescu-Mizil. 2020. Convokit: A toolkit for the analysis of conversations. *arXiv preprint arXiv:2005.04246*.

William S Condon and William D Ogston. 1967. A segmentation of behavior. *Journal of psychiatric research*.

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. 2011. Mark my words! linguistic style accommodation in social media. In *Proceedings of the 20th international conference on World wide web*, pages 745–754.

Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: Language effects and power differences in social interaction. In *Proceedings of the 21st international conference on World Wide Web*, pages 699–708.

H. Giles. 2008. Communication accommodation theory. In *Engaging theories in interpersonal communication: multiple perspectives*. Sage Publications.

Erving Goffman. 1967. Interaction ritual: Essays on face-to-face interaction.

Amy L Gonzales, Jeffrey T Hancock, and James W Pennebaker. 2010. Language style matching as a predictor of social dynamics in small groups. *Communication Research*, 37(1):3–19.

Fangjian Guo, Charles Blundell, Hanna Wallach, and Katherine Heller. 2015. The bayesian echo chamber: Modeling social influence via linguistic accommodation. In *Artificial Intelligence and Statistics*, pages 315–323.

Jerold L Hale and Judee K Burgoon. 1984. Models of reactions to changes in nonverbal immediacy. *Journal of Nonverbal Behavior*, 8(4):287–314.

William Hare. 1993. *Open-mindedness and education*. McGill-Queen's Press-MQUP.

William Hare. 2003. The ideal of open-mindedness and its place in education. *Journal of Thought*, 38(2):3–10.

Dominic A Infante, Andrew S Rancer, and Deanna F Womack. 1997. Building communication theory.

M. E. Francis J. W. Pennebaker, R. J. Booth. 2015. Linguistic inquiry and word count (liwc): A computerized text analysis program. `https://liwc.wpengine.com`.

Josep Jaffe and Stanley Feldstein. 1970. *Rhythms of Dialogue*. Academic Press.

Patrick Juola. 2008. *Authorship attribution*, volume 3. Now Publishers Inc.

Halit Keskin. 2006. Market orientation, learning orientation, and innovation capabilities in smes. *European Journal of innovation management*.

Mike Kestemont. 2014. Function words in authorship attribution. from black magic to theory? In *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*, pages 59–66.

Moshe Koppel, Shlomo Argamon, and Anat Rachel Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and linguistic computing*, 17(4):401–412.

Tatiana Litvinova, Pavel Seredin, Olga Litvinova, and Olga Zagorovskaya. 2016. Profiling a set of personality traits of text author: what our words reveal about us. *Research in Language*, 14(4):409–422.

Rebecca Mitchell and Stephen Nicholas. 2006. Knowledge creation in groups: The value of cognitive diversity, transactive memory and open-mindedness norms. *The Electronic Journal of Knowledge Management*, 4(1):67–74.

Kate Muir, Adam Joinson, Rachel Cotterill, and Nigel Dewdney. 2016. Characterizing the linguistic chameleon: Personal and social correlates of linguistic style accommodation. *Human Communication Research*, 42(3):462–484.

Kate Muir, Adam Joinson, Rachel Cotterill, and Nigel Dewdney. 2017. Linguistic style accommodation shapes impression formation and rapport in computer-mediated communication. *Journal of Language and Social Psychology*, 36(5):525–548.

Arjun Mukherjee and Bing Liu. 2010. Improving gender classification of blog authors. In *Proceedings of the 2010 conference on Empirical Methods in natural Language Processing*, pages 207–217.

Arjun Mukherjee and Bing Liu. 2012. Analysis of linguistic style accommodation in online debates. In *Proceedings of COLING 2012*, pages 1831–1846.

Kate G Niederhoffer and James W Pennebaker. 2002. Linguistic style matching in social interaction. *Journal of Language and Social Psychology*, 21(4):337–360.

Walter C Parker. 2006. Public discourses in schools: Purposes, problems, possibilities. *Educational Researcher*, 35(8):11–18.

R. L. Street and H. Giles. 1982. Speech accommodation theory. In *Social cognition and communication*. Sage Publications.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Dean Tjosvold and Motohiro Morishima. 1999. Grievance resolution: Perceived goal interdependence and interaction patterns. *Relations industrielles/Industrial Relations*, 54(3):527–548.

Dean Tjosvold and Margaret Poon. 1998. Dealing with scarce resources: Open-minded interaction for resolving budget conflicts. *Group & Organization Management*, 23(3):237–255.

Stepan Zakharov, Omri Hadar, Tovit Hakak, Dina Grossman, Yifat Ben-David Kolikant, and Oren Tsur. 2020. Discourse parsing of contentious, non-convergent online discussions.

Error Analysis and the Role of Morphology

Marcel Bollmann
Department of Computer Science
University of Copenhagen
Denmark
`marcel@di.ku.dk`

Anders Søgaard
Department of Computer Science
University of Copenhagen
Denmark
`soegaard@di.ku.dk`

Abstract

We evaluate two common conjectures in error analysis of NLP models: (i) Morphology is predictive of errors; and (ii) the importance of morphology increases with the morphological complexity of a language. We show across four different tasks and up to 57 languages that of these conjectures, somewhat surprisingly, only (i) is true. Using morphological features *does* improve error prediction across tasks; however, this effect is *less* pronounced with morphologically complex languages. We speculate this is because morphology is more discriminative in morphologically simple languages. Across all four tasks, case and gender are the morphological features most predictive of error.

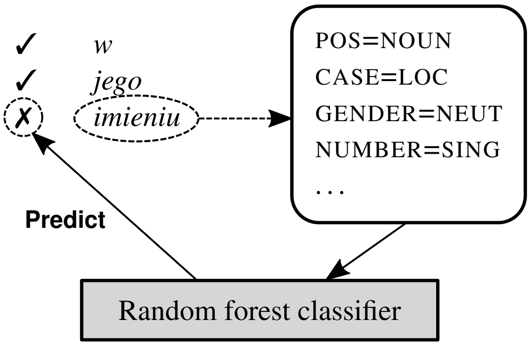


Figure 1: Overview of our methodology: We map each token to a set of morphological features and, based on this representation, predict whether some NLP system (e.g., a dependency parser) was correct (✓) or made an error (✗) on that token.

1 Introduction

In error analysis, we often blame morphology (Nivre, 2007; Bender, 2009), i.e., the productive inflection and derivation of new word forms. Morphology has been argued to be a major source of error in syntactic parsing (Tsarfaty et al., 2020), semantic parsing (Şahin and Steedman, 2018), machine translation (Irvine et al., 2013; Burlot and Yvon, 2017) and a range of other tasks, in particular in morphologically complex languages (Bender, 2009; Søgaard et al., 2018; Tsarfaty et al., 2020). This paper presents a large-scale study showing that morphology *is*, as commonly conjectured, an important source of error across tasks, but somewhat surprisingly, that morphology is *less* predictive of errors in morphologically complex languages.

English is a morphologically *simple* language, showing very limited inflection and expressing most concepts through syntactic structure instead; it is also the most-represented language at major natural language processing (NLP) venues and that with the largest amount of language resources available (Bender, 2011; Joshi et al., 2020). This makes it easy to ignore morphology when designing model architectures. As a consequence, we frequently observe that performance of NLP systems on morphologically more complex languages lags behind that for English (e.g. Czarnowska et al., 2019; Tsarfaty et al., 2020).

Complex morphology leads to the occurrence of rare inflected word forms. Polish nouns, for example, can inflect for number and seven different cases; this makes it less likely that all of these inflected word forms appear in the training data for our NLP models. Consequently, a model that correctly handles *imię* 'name' (NOM.SG) might not have seen the less frequent form *imionami* (INST.PL), potentially resulting in errors. If the model has generally seen fewer words in instrumental case, this can lead to systematic errors on this class of inflections.

Nowadays, many NLP systems use statistically learned subword units such as byte-pair encodings (Sennrich et al., 2016) or use characters as input representations, which could allow a system to generalize to individual affixes. However, in practice, these approaches are often found to be in-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1887–1900
April 19 - 23, 2021.

sufficient at capturing morphological structure (Vania and Lopez, 2017; Bostrom and Durrett, 2020; Klein and Tsarfaty, 2020).

Contributions In this study, we revisit two common conjectures about the role of morphology that are made in error analysis of NLP systems. Specifically, we ask whether (i) whether morphology is generally predictive of errors across tasks and languages; and (ii) whether the extent to which morphology is predictive depends on the morphological complexity of the language in question. These conjectures are common throughout the literature (Nivre, 2007; Bender, 2009; Manning, 2011).

Looking at data from four shared tasks on semantic role labeling (Hajič et al., 2009), dependency parsing (Zeman et al., 2018), verbal multi-word expression identification (Ramisch et al., 2018), and quality estimation (Fonseca et al., 2019), we map each token in the input data to *a set of morphological features*. Using only this feature set, and *without* using any orthographic or distributional representation of the input, we train random forest classifiers to predict whether a system has made an error on an input token. Figure 1 illustrates this approach.

Using this methodology, we find that, somewhat surprisingly, our results only support the first conjecture. In other words, (i) while morphology *is* helpful in predicting such errors, (ii) the degree to which morphology helps does not increase with the morphological complexity of the language. Moreover, we find and discuss task-specific differences between which morphological features are predictive of error. In general, part of speech, case and gender are most predictive of error.

The code for obtaining the datasets and running the experiments is made publicly available.[1]

2 Background

Morphology is frequently identified as a source of error during qualitative evaluations of NLP systems. Honnibal et al. (2010) observe that inflectional variants cause problems for statistical CCG tagging due to training data sparseness, and explicit morphological analysis helps, even for English. For dependency parsing, Seeker and Kuhn (2013) identify case syncretism as a source of error propagation in data from Czech, German, and Hungarian. Tsarfaty et al. (2020) give a broader overview of the challenges that rich morphological structure presents for dependency parsing, and Şahin and Steedman (2018) discuss the importance of morphology in semantic parsing.

Many observations of the effect of morphology come from evaluating machine translation (MT) systems. Federico et al. (2014) show that morphological errors are common for MT into Arabic and Russian and strongly affect human quality judgement. For English–Romanian MT, Peter et al. (2016) find that tense and verb form on the target side are a common source of error. Klubička et al. (2017) find that errors in English–Croatian MT are more common for some morphological categories, such as case. In a similar vein, Burlot and Yvon (2017) evaluate morphological competence of MT systems using contrast pairs and show that systems have different strengths and weaknesses for different morphological phenomena. Beyond parsing and MT, morphology has also been shown to present a challenge for tasks such as Arabic handwriting recognition (Habash and Roth, 2011) or Russian anaphora resolution (Toldova et al., 2016).

Most of the studies cited above predate contextual embedding models such as BERT (Devlin et al., 2019), which are now considered state-of-the-art for many NLP tasks. So far, few studies have explicitly analysed BERT with regard to morphology. Edmiston (2020) analyses morphological content in BERT-style models for five languages and finds that "[morphological] ambiguity is negatively correlated with performance on classification, and to a significant degree in many cases", suggesting that morphology is still a significant source of error in these models. We go significantly beyond this work by studying a much larger set of morphological variables, across several architectures and tasks, and across up to 57 languages.

3 Datasets

We collect datasets from shared tasks that (i) publish system outputs along with their gold annotations, (ii) span a variety of languages, and (iii) cover different NLP tasks. Based on these criteria, we pick datasets from the following shared tasks:

- **SEM**: CoNLL-2009 Shared Task on Semantic Dependencies (Hajič et al., 2009), covering semantic role labeling for seven languages.

---

[1] `https://github.com/coastalcph/eacl2021-morpherror`

- **UDP**: CoNLL-2018 Shared Task on Universal Dependencies Parsing (Zeman et al., 2018), covering syntactic parsing for 57 languages.

- **VMWE**: PARSEME 2018 Shared Task on Automatic Identification of Verbal Multiword Expressions (MWE; Ramisch et al., 2018), covering nine languages.

Additionally, we use the following dataset for its *gold* annotations:

- **MT**: WMT 2019 Shared Tasks on Quality Estimation (Fonseca et al., 2019), covering word-level quality estimation for English–German and English–Russian machine translation.

Here, we are not interested in the system outputs from the shared task; instead, we use the gold annotations for the quality estimation, which give us token-level error labels for the underlying machine translation outputs. Section 4.2 describes in detail how we assign error labels to these datasets.

4 Methodology

We train a classifier to predict errors made by NLP systems based on morphological features of the input tokens, in order to then analyze which morphological features (if any) are most predictive of such errors. We first describe how we obtain these features (Sec. 4.1) and how we classify when an NLP system has made an error (Sec. 4.2), then describe the classifier itself (Sec. 4.3).

4.1 Feature Extraction

We represent each token in the input data using a binary feature set. Each individual feature is named using the convention of {CATEGORY}={VALUE}, where the former is a feature category (such as POS for "part of speech") and the latter is a value within that category (e.g. VERB). We encode these features in a binary manner, i.e., for each feature in our inventory, that feature is either present or not present. Importantly, the classifier itself has no notion of "feature categories" as it only sees a single, binary feature vector.

The full feature inventory is summarized in Table 1; what follows is a description of these features and how we derived them.

Morphological features Our *morphological feature inventory* consists of (i) Universal Dependencies (UD) features, (ii) lexical features, and (iii) string-based features.

UD features include the universal part-of-speech (POS) category and the universal feature set as defined by Universal Dependencies; e.g. U:POS=VERB or U:TENSE=PAST.[2] The UDP shared-task gold data already provides this annotation; for the other tasks, we obtain these features by running UDPipe[3] (Straka and Straková, 2017) with the largest pre-trained model for the language in question.[4]

We complement this with the following additional *lexical features*: (i) SYNCRETIC specifies to what extent a token can be representative for several morphological feature sets: e.g., *ask* can be either U:MOOD=IND or U:MOOD=IMP, depending on context; (ii) $\text{AMBIG}_{\text{POS}}$ specifies to what extent the universal part-of-speech tag of the token can differ based on context: e.g., *book* could be either U:POS=VERB or U:POS=NOUN; and (iii) $\text{AMBIG}_{\text{LEX}}$ specifies whether or not the token belongs to multiple lexemes: e.g., *ruling* is a form of both '(to) rule' and '(the) ruling'. To determine these features for a given token, we use UDLexicons[5] (Sagot, 2018); in case a language is not covered by UDLexicons, we fall back to UniMorph[6] (Kirov et al., 2018).

Finally, we define purely *string-based features* based on comparing the token with its lemma. We perform character-based string alignment using Edlib (Šošić and Šikić, 2017) and derive the following features: (i) EDIT=PRE and EDIT=SUF when there is an edit at the beginning or the end of the sequence, respectively; (ii) EDIT=IN when there is an edit in the middle of the sequence; and (iii) EDIT=FULL when there is no character alignment between the strings. These features are intended to approximate prefixation, suffixation, infixation or other word-internal processes, and suppletion, respectively.

Control features To estimate the relative importance of our morphological features for the error prediction task, we additionally introduce a set of *control features* that are not morphologically motivated (cf. Tab. 1). These are (i) string length fea-

---

[2] The U: prefix serves to distinguish them from the other features we define. For the full feature set, see: `https://universaldependencies.org/u/feat/`

[3] `http://ufal.mff.cuni.cz/udpipe`

[4] The VMWE task only provides POS and morphology annotation for a subset of its languages, so for consistency, we choose to run UDPipe for all languages in this task as well.

[5] `http://pauillac.inria.fr/~sagot/index.html#udlexicons`

[6] `https://unimorph.github.io/`

Feature	**Definition**	
Morphological features		
U:POS={VALUE}	universal part-of-speech tag, e.g. U:POS=VERB	
U:{FEAT}={VALUE}	universal feature according to the UD specification, e.g. U:TENSE=PAST	
AMBIG$_{\text{POS}}$=NO	$\lvert P_t \rvert = 1$	where P_t is the set of all observed universal POS tags for t
AMBIG$_{\text{POS}}$=YES	$1 < \lvert P_t \rvert < 5$	
AMBIG$_{\text{POS}}$=HIGH	$\lvert P_t \rvert \geq 5$	
AMBIG$_{\text{LEX}}$=NO	$\lvert L_t \rvert = 1$	where L_t is the set of all observed lemmata for t
AMBIG$_{\text{LEX}}$=YES	$\lvert L_t \rvert > 1$	
SYNCRETIC=NO	$\lvert M_t \rvert = 1$	where M_t is the set of all observed morphological feature combinations for t
SYNCRETIC=YES	$1 < \lvert M_t \rvert < 5$	
SYNCRETIC=HIGH	$\lvert M_t \rvert \geq 5$	
EDIT=PRE	$x_0 \neq \text{MATCH}$	where $[x_0, \ldots, x_n]$ is the sequence of edit alignments between t and l, $x_i \in \{\text{MATCH}, \text{MISMATCH}, \text{GAP}\}$
EDIT=SUF	$x_n \neq \text{MATCH}$	
EDIT=IN	$\exists i, j, k : i < j < k$	
	$\wedge x_i = \text{MATCH}$	
	$\wedge x_j \neq \text{MATCH}$	
	$\wedge x_k = \text{MATCH}$	
EDIT=FULL	$\forall i : x_i \neq \text{MATCH}$	
Control features		
LEN=1-3	$1 \leq \lvert t \rvert \leq 3$	where $\lvert t \rvert$ is the string length of t
LEN=4-6	$4 \leq \lvert t \rvert \leq 6$	
LEN=7-9	$7 \leq \lvert t \rvert \leq 9$	
LEN=10+	$\lvert t \rvert \geq 10$	
FREQ=99	$P_{99} \leq f(t)$	where $f(t)$ is the absolute frequency count of t and P_n is the n-th percentile of the frequency distribution
FREQ=98	$P_{98} \leq f(t) < P_{99}$	
FREQ=95	$P_{95} \leq f(t) < P_{98}$	
FREQ=90	$P_{90} \leq f(t) < P_{95}$	
FREQ=UNCOMMON	$4 \leq f(t) < P_{90}$	
FREQ=RARE	$f(t) < 4$	

Table 1: Inventory of extracted features (cf. Sec. 4.1). t always denotes the token, l its lemma.

tures, where each token is assigned exactly one such feature depending on its length; and (ii) token frequency bins. For the latter, we count token frequencies in the Universal Dependencies treebanks and assign each token a frequency feature. These features are based on frequency bins that we manually curated to provide a roughly balanced distribution of tokens to bins: e.g., FREQ=99 denotes a token that is in the 99th percentile of the frequency distribution of all types, while FREQ=RARE denotes a token occurring less than four times overall (see Table 1 for all definitions).

Pruning and statistics Since very rare features are not very informative, for any given dataset, we remove features that occur less than 10 times in that dataset. Depending on the task and language, we generate between 17 and 120 unique features this way, with an average of 68.

4.2 Classifying errors in system outputs

The target variable for our classifier is a binary label corresponding to whether or not the shared-task system has made an error on the input token. This requires comparing the outputs of a system to the gold data and classifying each token as either *correct* or *incorrect*. We will also refer to the latter as the *error* class. This classification follows the original evaluation criteria by the shared tasks to the extent possible.

For SEM, a prediction is classified as "correct" *iff* the semantic dependencies and label columns are an exact match with the gold data. For UDP, we do the same with the syntactic head and dependency relation columns; this is the same criterion that underlies the labeled attachment score (LAS) commonly used to evaluate dependency parsing. VMWE is a little more challenging since its prediction involves a set of tokens within a sentence. For each sentence, we match up each gold MWE with the predicted MWE that has the same label and the largest token overlap. We then consider a token "incorrectly" predicted if has a MWE annotation that does not belong to one of these matched MWEs, or if it lacks a MWE annotation that it should have according to the gold data.

As mentioned before, we treat the MT data a little differently: here, the *gold data* provides binary labels in the form of "OK" and "BAD" tags, corresponding to the correctness of some machine translation system. These tags are provided both for tokens and gaps between tokens (to account for the deletion/insertion of words in machine translation). We use the token-level tags from the gold data directly as our error classification labels.

Appendix A gives an example for the error classification approach on VMWE and MT.

4.3 Training classifiers

With the extracted features (from Sec. 4.1), we can now train classifiers to predict the error variable (from Sec. 4.2). Concretely, we train *random forest classifiers* (Breiman, 2001) as implemented by Scikit-learn[7] (Pedregosa et al., 2011) on each output file provided by each shared task. Random forests are ensembles of decision trees and are quick to train: the average training time on our datasets was 14 seconds on CPU, with no single run taking longer than five minutes.

As an alternative to random forests, we also experimented with randomized logistic regression classifiers followed by *stability selection* (Meinshausen and Bühlmann, 2010) to select predictive features. In our trials, this approach showed a worse performance (in terms of F_1-score) compared to random forests, while also taking considerably longer to run (averaging 7 minutes per dataset). We therefore only report results with random forest classifiers.

5 Analysis

For each shared task (Sec. 3), we ran our classification pipeline (Sec. 4) separately for each combination of (i) system submission and (ii) language evaluated on. Since random forests are largely interpretable, our analysis focuses on the important features in our learned models.

First, though, we look at the overall F_1-score of the individual classifiers, which we evaluate via stratified 5-fold cross-validation on each data point (Sec. 5.1). Additionally, to better estimate the importance of morphology, we run our cross-validation pipeline a second time *without* the morphological features, i.e., only providing the classifiers with the "control features" shown in Tab. 1. We refer to these two feature sets as "full" and "control" settings, respectively, and analyze their differences in F_1-score (Sec. 5.2).[8] Finally, we analyse the importance of individual morphological features (Sec. 5.3).

5.1 How well do the classifiers predict errors?

To evaluate how well the full classifiers learned the task, we consider their F_1-score for predicting the "error" class. Across all of our datasets, we observe a mean F_1 of 0.43 with a standard deviation of ± 0.18. Note that our setup is not comparable to most other NLP classification tasks: we evaluate a classifier trained to detect the errors of state-of-the-art systems, which means that (i) the task is inherently hard, as those systems are optimized to fix easily detectable errors, and (ii) there is no reason to assume a priori that this task is well learnable from morphological input features alone. Therefore, we believe an F_1 score of 0.43—albeit with considerable variance in performance across tasks and languages—is a strong result.

Error rate There is one important aspect to consider: the frequency of the "error" class depends on the system performance of the data point we look at, and as such our class distribution can be highly imbalanced and varied. Indeed, F_1-score and frequency of the error class correlate very strongly with Pearson's $r = 0.93$. Figure 2 plots this relationship.[9] This suggests that the errors introduced by state-of-the-art NLP systems, unsurprisingly, become harder and harder to predict the better the underlying systems perform.

Note that data imbalance is in the nature of the error prediction task, as we expect errors in state-of-the-art systems to be rare. Additionally, different

---

[7] We use the default parameters in Scikit-learn 0.23, with the exception of setting class weights to be "balanced" according to their frequencies in the input data.

[8] To complement the results and analyses presented here, we also provide a detailed table with the results for all task/language pairs in Appendix B.

[9] It might look surprising that many data points have very high error rates, with some even going above 0.95; i.e., more than 95% of all predictions in the respective file are deemed to be "incorrect" according to the criteria in Sec. 4.2. Spot-checking reveals that this is, however, plausible: for example, in UDP, the average labeled attachment score (LAS) on the Thai TH_PUD treebank was only 1.38 (Zeman et al., 2018, Table 15), with 23 systems achieving a LAS of only 0.77 or lower (out of 100; cf. `http://universaldependencies.org/conll18/results-las.html`), which is reflected by an error rate of $\geq$99.23% in our data.

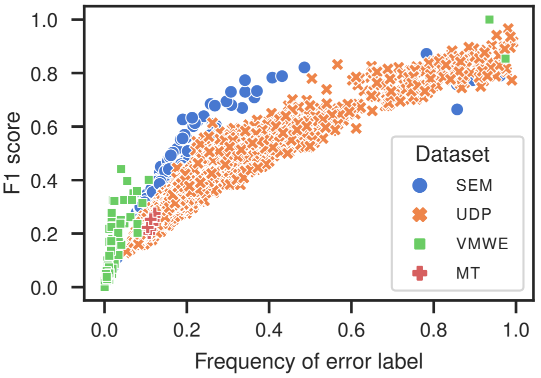


Figure 2: F_1-scores of trained error classifiers in relation to the frequency of error, i.e. the error rate of the original model (cf. Sec. 4.2).

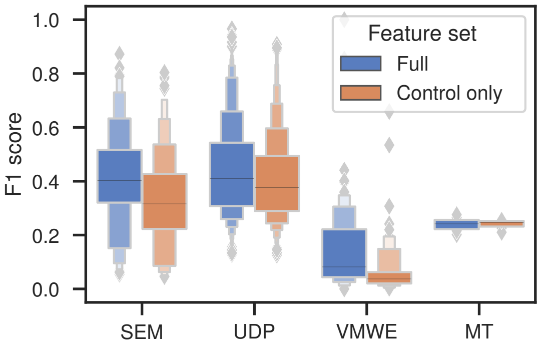


Figure 3: Quantiles of the F_1 distribution by dataset, and whether the classifiers were trained using the full feature set from Tab. 1 (blue) or only the control features (orange).

languages have differently-sized morphological tag inventories, affecting the total number of input features for the classifier. We do not attempt to apply data balancing techniques to counteract this, since this would make the task artificially easy and our results overly optimistic.

5.2 How important is morphology for predicting errors?

Figure 3 provides an alternative view of the F_1-scores presented in Fig. 2, this time as a letter-value plot (Hofmann et al., 2017) showing quantiles of the F_1 distribution. Additionally, we compare the classifier with the full feature set to the control set where morphological features were not included.

We observe that the classifiers learn best on UDP followed by SEM, while classifier F_1 is relatively poor on VMWE data. A probable explanation for this is the generally low error rate in VMWE (cf. Fig. 2). The other important observation is that classifiers in the "control" setting score consistently lower than the classifiers that have access to morphological features.

Importance by language For looking at individual languages, we restrict ourselves to the UDP data. Firstly, UDP covers 57 languages—more than any other task in our comparison—and there are no languages in the other tasks that are not also contained in UDP. Secondly, our classifier performance is generally highest on UDP (cf. Fig. 3), allowing for a more meaningful interpretation of results, particularly of selected features.

Furthermore, to factor out the effect of a data point's error rate (as discussed in Sec. 5.1), we look at the *difference* between the F_1-score of the full classifier and the control classifier trained on the same data point. In other words, we define

$$\Delta F_1 = F_1(g_f) - F_1(g_c) \qquad (1)$$

where g_f and g_c are the classifiers with the full and the control feature set, respectively. This gives us a way to judge the importance of morphological features relative to the non-morphological ones while minimizing the effect of the error rate on the results, since ΔF_1 no longer shows a strong correlation with the error rate ($r = 0.29$).

Figure 4 (bottom half) shows the quartiles of ΔF_1 scores by language in the UDP dataset. They span a wide range of values, with the median ΔF_1 varying gradually between -0.03 (for Turkish, TUR) and 0.24 (for Nigerian Pidgin, PCM). Morphological features appear to be important for some languages while being unhelpful, and sometimes even detrimental, for others.

Morphological complexity Are the differences in ΔF_1 scores (in Fig. 4) somehow related to the morphological complexity of the languages? To analyze this relationship more systematically, we use the measure of *morphological feature entropy* (MFE) introduced by Çöltekin and Rama (2018). MFE is sensitive to both the size of a language's morphological feature inventory as well as its distribution, with a more uniform distribution of features resulting in a higher MFE. Since MFE is a treebank measure that relies on the association between tokens and morphological tags, it is affected by tokenization and annotation choices of the treebank used to calculate it; therefore, it can only be considered a rough approximation of the underlying language's complexity. Like Çöltekin

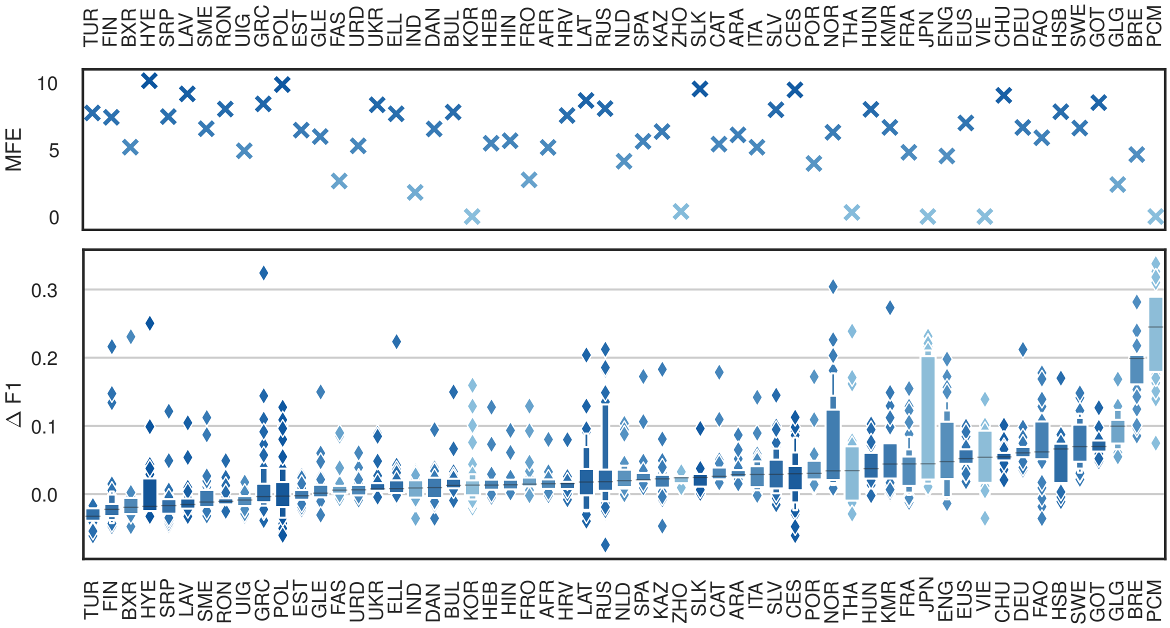


Figure 4: Classifier performance on UDP by language, sorted by median ΔF_1, where ΔF_1 is the difference in F_1-scores between training with the full and the control feature set (cf. Eq. 1). Bottom half shows the quartiles of the ΔF_1 distribution, top half shows the morphological feature entropy (MFE) for the given language; color shading is also based on MFE (with darker shade = higher MFE). Full names for all language codes as well as exact numeric values can be found in Appendix B.

and Rama (2018), we calculate the MFE score for each language on the UD treebanks.[10]

The MFE score for each language is shown in the top half of Fig. 4. Surprisingly, we find a slight, negative correlation between MFE and ΔF_1 (Pearson's $r = -0.24$). While languages with high MFE appear across the whole range of the ΔF_1 distribution, a number of languages with low MFE—and thus deemed to be more morphologically simple, such as Thai (THA), Japanese (JPN), or Nigerian Pidgin (PCM)—are found to profit *more* from the inclusion of morphological features. One possible explanation is that the control features are already very strong, which we will look at more closely in Sec. 5.3. Another possible factor is that morphologically complex languages introduce a much larger set of morphological features; if, for a given language, most of them are not relevant for predicting errors in the UDP task, they might hurt the overall classifier performance.

5.3 What morphological features are most predictive of errors?

Morphological features provide a helpful signal to the classifiers, though its overall magnitude differs by language (cf. Sec. 5.2). Now, we ask *which* of the morphological features are particularly relevant for error prediction. Since plain feature importances of trained random forest classifiers can be misleading (Strobl et al., 2007; Parr et al., 2018), we follow the approach of explicitly removing features and retraining (Parr et al., 2018; Hooker and Mentch, 2019). Unlike the analyses above, we are not concerned with generalization here, but with identifying features that are especially predictive for the error variable on each dataset as a whole. Therefore, we do not use a cross-validation strategy, but rely on the full dataset for both training and obtaining feature importances.

Concretely, for each feature *category* (as introduced in Sec. 4.1), we retrain the model *without* features from that category and note the drop in error-class F_1-score compared to the model with the full feature set. Formally, let Φ be the full feature set and $\phi_c \subset \Phi$ the subset of features belonging to category c (e.g., c = U:TENSE). The *importance* of category c is then defined as

$$f(c) = F_1(C_\Phi) - F_1(C_{\Phi \setminus \phi_c}) \qquad (2)$$

where C_X is a random forest classifier trained using feature set X. Higher values for $f(c)$ mean a higher importance of category c, while negative values

---

[10] We use UD version 2.5 (Zeman et al., 2019).

Category	FI
U:POS	32.65
FREQ	15.51
LEN	11.38
U:CASE	7.25
U:GENDER	6.96
EDIT	6.75
U:NUMBER	3.78
U:NAMETYPE	2.77
U:ANIMACY	2.73
U:ADPTYPE	1.96

(a) SEM

Category	FI
U:POS	34.74
FREQ	31.99
LEN	21.79
U:CASE	16.51
U:GENDER	10.98
EDIT	9.01
U:NUMBER	6.47
U:ANIMACY	3.78
U:ASPECT	2.28
SYNCRETIC	2.08

(b) UDP

Category	FI
FREQ	38.24
LEN	28.97
U:CASE	12.56
U:POS	12.47
U:GENDER	10.15
EDIT	9.78
U:ANIMACY	7.24
U:NUMBER	7.15
U:ASPECT	5.81
U:TENSE	2.68

(c) VMWE

Category	FI
FREQ	29.50
LEN	19.63
U:CASE	12.39
U:POS	10.93
U:GENDER	9.69
EDIT	5.91
U:NUMBER	3.94
U:ASPECT	3.25
SYNCRETIC	2.84
U:ANIMACY	2.30

(d) MT

Table 2: Top 10 feature categories by *average* feature importance (FI) for each task. All FI scores given $\cdot 10^{-3}$

mean that including c is actually detrimental to the F_1-score.

Average feature importances Table 2 shows the top 10 feature categories for each task, averaged over all languages and datasets. The two control features, FREQ and LEN, always appear among the three most important categories, only trumped by U:POS for the UDP and SEM tasks. Notably, these three are the only feature categories that are *guaranteed* to appear with every token. It is no surprise that token frequency is strongly related to the likelihood of errors, while Zipf's law tells us that token length is strongly correlated with frequency.

Figure 5 shows the *distribution* of feature importances for the top 10 categories of UDP (cf. Tab. 2b). U:POS spans a much wider range of FI values than any of the other categories, although the outliers at the upper end all come from Nigerian Pidgin (PCM). Moreover, categories with a low average FI (e.g., U:ASPECT or SYNCRETIC) do not show outliers, i.e., are of low importance across languages. This is also true for the remaining feature categories.

Individual part-of-speech tags Since U:POS is an important feature category across tasks (cf. Tab. 2), we also look at feature importances for *individual* POS tags. For this, we use the same approach as for the feature categories (cf. Eq. 2), except that we now only remove a single U:POS feature from Φ at a time.

Table 3 shows the average feature importances for individual U:POS features, though this time we restrict ourselves to the subset of languages in UDP that are also covered in SEM.[11] This way, we can better isolate the *task-specific* differences in FI scores, without conflating them with the different *language-specific* distributions of part-of-speech tags that may affect these results. We find that adverbs (ADV) are the most important part-of-speech category for both tasks, while INTJ and PART are found to be important for predicting errors in UDP, but not in SEM. This aligns with our intuitions about what is hard in syntactic and semantic parsing, further supporting the validity of our approach.

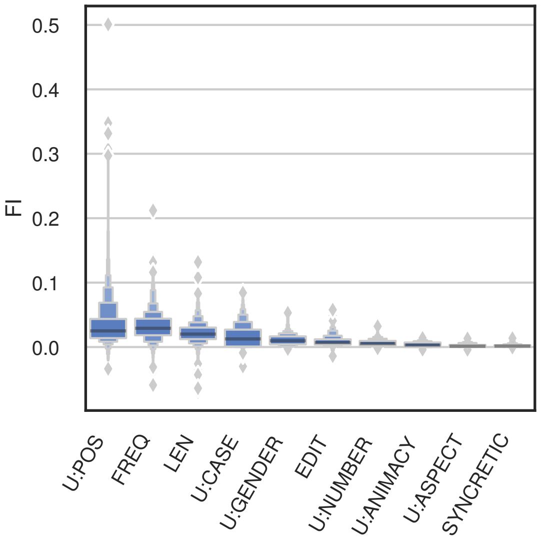


Figure 5: Quartiles of the top 10 feature categories on UDP by average feature importance (FI).

6 Conclusion

We presented a large-scale error analysis focusing on the role of morphology. Our analysis spans a range of morphological variables, four NLP tasks, and up to 57 languages. We confirm the common conjecture that morphological variables—especially case and gender—are predictive of errors across NLP tasks and languages. Somewhat sur-

---

[11] These are Catalan, Czech, German, English, Japanese, Spanish, and Chinese; cf. Appendix B.

POS	FI ($\cdot 10^{-6}$)
ADV	39.5
INTJ	38.4
PART	26.5
AUX	15.0
PROPN	11.2
CCONJ	5.2
ADP	4.9
SCONJ	3.4
PRON	3.2
SYM	3.1
X	2.0
DET	0.6
VERB	-0.3
NUM	-1.7
PUNCT	-13.8
ADJ	-14.7
NOUN	-15.9

(a) UDP

POS	FI ($\cdot 10^{-6}$)
ADV	9.9
AUX	6.8
X	4.3
ADP	3.7
SCONJ	2.9
SYM	1.7
NOUN	-1.2
PRON	-1.2
PART	-2.6
INTJ	-4.3
NUM	-4.3
PROPN	-5.0
VERB	-5.0
DET	-6.2
CCONJ	-6.8
ADJ	-8.1
PUNCT	-9.3

(b) SEM

Table 3: Average feature importance (FI) for U:POS features on the subset of languages that are both in UDP and SEM.

prisingly, we found that the usefulness of morphological variables is *negatively* correlated with the morphological complexity of the language in question. We speculate this is because morphological information is more discriminative in morphologically simple languages.

Acknowledgments

Marcel Bollmann was funded from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 845995. Anders Søgaard was funded by a Google Focused Research Award.

References

Emily M. Bender. 2009. Linguistically naïve != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece. Association for Computational Linguistics.

Emily M. Bender. 2011. On achieving and evaluating language-independence in NLP. *Linguistic Issues in Language Technology*, 6(3):1–26.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45:5–32.

Franck Burlot and François Yvon. 2017. Evaluating the morphological competence of machine translation systems. In *Proceedings of the Second Conference on Machine Translation*, pages 43–55, Copenhagen, Denmark. Association for Computational Linguistics.

Çağrı Çöltekin and Taraka Rama. 2018. Exploiting Universal Dependencies treebanks for measuring morphosyntactic complexity. In *Proceedings of the First Shared Task on Measuring Language Complexity*, Torun, Poland.

Paula Czarnowska, Sebastian Ruder, Edouard Grave, Ryan Cotterell, and Ann Copestake. 2019. Don't forget the long tail! A comprehensive analysis of morphological generalization in bilingual lexicon induction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 974–983, Hong Kong, China. Association for Computational Linguistics.

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

Daniel Edmiston. 2020. A systematic analysis of morphological content in BERT models for multiple languages. arXiv:2004.03032.

Marcello Federico, Matteo Negri, Luisa Bentivogli, and Marco Turchi. 2014. Assessing the impact of translation errors on machine translation quality with mixed-effects models. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1643–1653, Doha, Qatar. Association for Computational Linguistics.

Erick Fonseca, Lisa Yankovskaya, André F. T. Martins, Mark Fishel, and Christian Federmann. 2019. Findings of the WMT 2019 shared tasks on quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Nizar Habash and Ryan Roth. 2011. Using deep morphology to improve automatic error detection in Arabic handwriting recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 875–884, Portland, Oregon, USA. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís

Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Heike Hofmann, Hadley Wickham, and Karen Kafadar. 2017. Letter-value plots: Boxplots for large data. *Journal of Computational and Graphical Statistics*, 26(3):469–477.

Matthew Honnibal, Jonathan K. Kummerfeld, and James R. Curran. 2010. Morphological analysis can improve a CCG parser for English. In *Coling 2010: Posters*, pages 445–453, Beijing, China. Coling 2010 Organizing Committee.

Giles Hooker and Lucas Mentch. 2019. Please stop permuting features: An explanation and alternatives. arXiv:1905.03151.

Ann Irvine, John Morgan, Marine Carpuat, Hal Daumé III, and Dragos Munteanu. 2013. Measuring machine translation errors in new domains. *Transactions of the Association for Computational Linguistics*, 1:429–440.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

Filip Klubička, Antonio Toral, and Víctor M. Sánchez-Cartagena. 2017. Fine-grained human evaluation of neural versus phrase-based machine translation. *The Prague Bulletin of Mathematical Linguistics*, 108(1):121–132.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *Proceedings of the 12th international conference on Computational linguistics and intelligent text processing - Volume Part I*, CICLing'11, pages 171–189, Berlin, Heidelberg. Springer-Verlag.

Nicolai Meinshausen and Peter Bühlmann. 2010. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Joakim Nivre. 2007. Data-driven dependency parsing across languages and domains: Perspectives from the CoNLL-2007 shared task. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 168–170, Prague, Czech Republic. Association for Computational Linguistics.

Terence Parr, Kerem Turgutlu, Christopher Csiszar, and Jeremy Howard. 2018. Beware default random forest importances.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Jan-Thorsten Peter, Tamer Alkhouli, Hermann Ney, Matthias Huck, Fabienne Braune, Alexander Fraser, Aleš Tamchyna, Ondřej Bojar, Barry Haddow, Rico Sennrich, Frédéric Blain, Lucia Specia, Jan Niehues, Alex Waibel, Alexandre Allauzen, Lauriane Aufrant, Franck Burlot, Elena Knyazeva, Thomas Lavergne, François Yvon, Mārcis Pinnis, and Stella Frank. 2016. The QT21/HimL combined machine translation system. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 344–355, Berlin, Germany. Association for Computational Linguistics.

Carlos Ramisch, Silvio Ricardo Cordeiro, Agata Savary, Veronika Vincze, Verginica Barbu Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, Tunga Güngör, Abdelati Hawwari, Uxoa Iñurrieta, Jolanta Kovalevskaitė, Simon Krek, Timm Lichte, Chaya Liebeskind, Johanna Monti, Carla Parra Escartín, Behrang QasemiZadeh, Renata Ramisch, Nathan Schneider, Ivelina Stoyanova, Ashwini Vaidya, and Abigail Walsh. 2018. Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Benoît Sagot. 2018. A multilingual collection of CoNLL-u-compatible morphological lexicons. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Gözde Gül Şahin and Mark Steedman. 2018. Character-level models versus morphology in semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 386–396, Melbourne, Australia. Association for Computational Linguistics.

Wolfgang Seeker and Jonas Kuhn. 2013. Morphological and syntactic case in statistical dependency parsing. *Computational Linguistics*, 39(1):23–55.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788, Melbourne, Australia. Association for Computational Linguistics.

Milan Straka and Jana Straková. 2017. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. 2007. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(25).

Svetlana Toldova, Ilya Azerkovich, Alina Ladygina, Anna Roitberg, and Maria Vasilyeva. 2016. Error analysis for anaphora resolution in Russian: new challenging issues for anaphora resolution task in a morphologically rich language. In *Proceedings of the Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2016)*, pages 74–83, San Diego, California. Association for Computational Linguistics.

Reut Tsarfaty, Dan Bareket, Stav Klein, and Amit Seker. 2020. From SPMRL to NMRL: What did we learn (and unlearn) in a decade of parsing morphologically-rich languages (MRLs)? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7396–7408, Online. Association for Computational Linguistics.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2016–2027, Vancouver, Canada. Association for Computational Linguistics.

Martin Šošić and Mile Šikić. 2017. Edlib: a C/C++ library for fast, exact sequence alignment using edit distance. *Bioinformatics*, 33(9):1394–1395.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, Noëmi Aepli, Željko Agić, Lars Ahrenberg, Gabrielė Aleksandravičiūtė, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, Colin Batchelor, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Agnė Bielinskienė, Rogier Blokland, Victoria Bobicev, Loïc Boizou, Emanuel Borges Völker, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Kristina Brokaitė, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Tatiana Cavalcanti, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Alessandra T. Cignarella, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Elvis de Souza, Arantza Diaz de Ilarraza, Carly Dickerson, Bamba Dione, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Hanne Eckhoff, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Olga Erina, Tomaž Erjavec, Aline Etienne, Wograine Evelyn, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Kazunori Fujita, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Bernadeta Griciūtė, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Mika Hämäläinen, Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Johannes Heinecke, Felix Hennig, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Takumi Ikeda, Radu Ion, Elena Irimia, Ọlájídé Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Markus Juutinen, Hüner Kaşıkara, Andre Kaasen, Nadezhda Kabaeva, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Elena Klementieva, Arne Köhn, Kamil Kopacewicz, Natalia Kotsyba, Jolanta

Kovalevskaitė, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phuong Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Maria Liovina, Yuan Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Sarah McGuinness, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Maria Mitrofan, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Tomohiko Morioka, Shinsuke Mori, Shigeki Moro, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Robert Munro, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Luong Nguyễn Thị, Huyền Nguyễn Thị Minh, Yoshihiro Nikaido, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Atul Kr. Ojha, Adédayọ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Angelika Peljak-Łapińska, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Daria Petrova, Slav Petrov, Jason Phelan, Jussi Piitulainen, Tommi A Pirinen, Emily Pitler, Barbara Plank, Thierry Poibeau, Larisa Ponomareva, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Peng Qi, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Ivan Riabov, Michael Rießler, Erika Rimkutė, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roșca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Alessio Salomoni, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Dage Särg, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Hiroyuki Shirasu, Muh Shohibussirri, Dmitry Sichinava, Aline Silveira, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Shingo Suzuki, Zsolt Szántó, Dima Taji, Yuta Takahashi, Fabio Tamburini, Takaaki Tanaka, Isabelle Tellier, Guillaume Thomas, Liisi Torga, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Andrius Utka, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Abigail Walsh, Jing Xian Wang, Jonathan North Washington, Maximilan Wendt, Seyi Williams, Mats Wirén, Christian Wittern, Tsegay Woldemariam, Tak-sum Wong, Alina Wróblewska, Mary Yako, Naoki Yamazaki, Chunxiao Yan, Koichi Yasuoka, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Manying Zhang, and Hanzhi Zhu. 2019. Universal dependencies 2.5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

	✗	✓	✓	✓	✗	✓	✓	✓	✓
	To	było	uczciwe	postawienie	sprawy	–	utrzymuje	Kurski	.
GOLD:				1	1				
SYS:	1			1					

(a) VMWE example from Polish

		✓		✓		✗		✓		✓		✗		✓	
	␣	ersetzt	␣	die	␣	Standardglyphen	␣	durch	␣	die	␣	Glyphenglyphenglyphen	␣	.	␣
GOLD:	OK	OK	OK	OK	OK	BAD	OK	OK	OK	OK	OK	BAD	OK	OK	OK

(b) MT example from German

Table 4: Examples for how tokens are classified as correct (✓) or incorrect (✗) in our experiments. GOLD shows gold annotations, SYS shows output from an NLP system participating in the respective shared task.

A Examples for error classification

Table 4a shows an example for how we classify errors (cf. Sec. 4.2) in the VMWE dataset on verbal multi-word expression (MWE) identification. In the gold data, a single MWE (*'postawienie sprawy'*) is annotated, while the NLP system has incorrectly identified the MWE as being *'to … postawienie'*. The annotation "1" here is an ID in case there are multiple MWEs within the same sentence. We annotate both *'to'*, which was mistakenly identified as part of the MWE, as well as *'sprawy'*, which was mistakenly left out, as an error (✗). All remaining tokens are marked as correct (✓).

Table 4b shows an example from the MT dataset on quality estimation for machine translation (MT). Here, the gold data provides us with "OK" and "BAD" labels for the individual tokens of the machine-generated translation as well as for the *gaps* between the tokens. The latter is done to be able to annotate missing passages in the machine translation output; i.e., a gap between tokens would be labelled "BAD" if the MT system should have produced more output at a given position in a sentence than it did. Since it is unclear to which (existing) tokens these "gap annotations" should be ascribed to, we do not consider them for the error classification, and only consider "OK/BAD" labels for the tokens that *do* appear in the data.

B Statistics and classifier results

Table 5 presents statistics and classifier results, corresponding to the analyses in Secs. 5.1 and 5.2, for each task/language pair. The column "Avg. error rate" corresponds to the error rates plotted in Fig. 2, while the "MFE" column shows the morphological feature entropy (cf. Sec. 5.2) for the respective language. "Avg. F_1" shows the average F_1-score after stratified 5-fold cross-validation (cf. Sec. 5.1), while "Avg. ΔF_1" corresponds to the ΔF_1-measure defined in Eq. (1).

Dataset	Language (ISO 639-3 + Name)		Avg. Error Ratio	MFE	Avg. F_1	Avg. ΔF_1
SEM	CAT	Catalan	0.15	5.41	0.39	0.13
	CES	Czech	0.31	9.47	0.63	0.08
	DEU	German	0.03	6.65	0.11	0.03
	ENG	English	0.19	4.52	0.42	0.08
	JPN	Japanese	0.23	0.00	0.57	0.09
	SPA	Spanish	0.16	5.62	0.40	0.12
	ZHO	Chinese	0.21	0.38	0.39	0.05
UDP	AFR	Afrikaans	0.20	5.16	0.40	0.02
	ARA	Arabic	0.26	6.10	0.44	0.03
	BRE	Breton	0.84	4.63	0.83	0.18
	BUL	Bulgarian	0.14	7.80	0.31	0.02
	BXR	Russia Buriat	0.87	5.18	0.85	-0.01
	CAT	Catalan	0.13	5.41	0.31	0.04
	CES	Czech	0.16	9.47	0.29	0.03
	CHU	Church Slavic	0.32	9.06	0.50	0.06
	DAN	Danish	0.22	6.55	0.35	0.01
	DEU	German	0.27	6.65	0.44	0.07
	ELL	Modern Greek (1453-)	0.16	7.68	0.33	0.02
	ENG	English	0.22	4.52	0.41	0.06
	EST	Estonian	0.22	6.46	0.34	-0.00
	EUS	Basque	0.25	6.99	0.41	0.06
	FAO	Faroese	0.68	5.89	0.71	0.07
	FAS	Persian	0.18	2.66	0.35	0.01
	FIN	Finnish	0.21	7.41	0.32	-0.01
	FRA	French	0.22	4.80	0.40	0.04
	FRO	Old French (842-ca. 1400)	0.23	2.74	0.40	0.02
	GLE	Irish	0.36	5.98	0.55	0.01
	GLG	Galician	0.26	2.39	0.50	0.09
	GOT	Gothic	0.37	8.51	0.55	0.07
	GRC	Ancient Greek (to 1453)	0.35	8.42	0.49	0.01
	HEB	Hebrew	0.29	5.48	0.48	0.02
	HIN	Hindi	0.11	5.67	0.29	0.02
	HRV	Croatian	0.18	7.55	0.32	0.02
	HSB	Upper Sorbian	0.72	7.83	0.72	0.05
	HUN	Hungarian	0.30	8.01	0.46	0.04
	HYE	Armenian	0.76	10.15	0.73	0.01
	IND	Indonesian	0.24	1.81	0.38	0.01
	ITA	Italian	0.20	5.18	0.39	0.03
	JPN	Japanese	0.42	0.00	0.55	0.11
	KAZ	Kazakh	0.78	6.35	0.80	0.03
	KMR	Northern Kurdish	0.77	6.66	0.75	0.06
	KOR	Korean	0.22	0.00	0.39	0.02
	LAT	Latin	0.35	8.67	0.46	0.02
	LAV	Latvian	0.26	9.17	0.36	-0.01
	NLD	Dutch	0.20	4.13	0.38	0.03
	NOR	Norwegian	0.25	6.29	0.38	0.07
	PCM	Nigerian Pidgin	0.84	0.00	0.79	0.24
	POL	Polish	0.13	9.86	0.25	0.00
	POR	Portuguese	0.16	3.97	0.32	0.04
	RON	Romanian	0.18	8.03	0.32	-0.01
	RUS	Russian	0.27	8.07	0.39	0.04
	SLK	Slovak	0.20	9.54	0.37	0.02
	SLV	Slovenian	0.35	7.98	0.47	0.03
	SME	Northern Sami	0.44	6.56	0.53	0.00
	SPA	Spanish	0.14	5.62	0.31	0.03
	SRP	Serbian	0.17	7.47	0.30	-0.01
	SWE	Swedish	0.23	6.61	0.41	0.07
	THA	Thai	0.93	0.31	0.83	0.04
	TUR	Turkish	0.41	7.74	0.48	-0.03
	UIG	Uighur	0.41	4.91	0.60	-0.01
	UKR	Ukrainian	0.21	8.35	0.36	0.02
	URD	Urdu	0.21	5.28	0.42	0.01
	VIE	Vietnamese	0.51	0.00	0.58	0.05
	ZHO	Chinese	0.31	0.38	0.47	0.02
VMWE	ELL	Modern Greek (1453-)	0.01	7.68	0.11	0.06
	ENG	English	0.06	4.52	0.26	0.10
	EUS	Basque	0.03	6.99	0.17	0.07
	FAS	Persian	0.03	2.66	0.11	0.03
	HEB	Hebrew	0.01	5.48	0.05	0.01
	HUN	Hungarian	0.10	8.01	0.15	0.07
	POL	Polish	0.02	9.86	0.16	0.11
	POR	Portuguese	0.07	3.97	0.15	0.06
	SPA	Spanish	0.02	5.62	0.24	0.17
MT	DEU	German	0.12	6.65	0.26	0.00
	RUS	Russian	0.11	8.07	0.22	-0.00

Table 5: Statistics and classifier results averaged over each task/language pair.

Applying the Transformer to Character-level Transduction

Shijie Wu[ʒ] **Ryan Cotterell**[ʕ,ɒ] **Mans Hulden**[χ]
[ʒ]Johns Hopkins University [ɒ]University of Cambridge
[ʕ]ETH Zürich [χ]University of Colorado Boulder
shijie.wu@jhu.edu ryan.cotterell@inf.ethz.ch mans.hulden@colorado.edu

Abstract

The transformer (Vaswani et al., 2017) has been shown to outperform recurrent neural network-based sequence-to-sequence models in various word-level NLP tasks. Yet for character-level transduction tasks, e.g. morphological inflection generation and historical text normalization, there are few works that outperform recurrent models using the transformer. In an empirical study, we uncover that, in contrast to recurrent sequence-to-sequence models, the batch size plays a crucial role in the performance of the transformer on character-level tasks, and we show that with a large enough batch size, the transformer does indeed outperform recurrent models. We also introduce a simple technique to handle feature-guided character-level transduction that further improves performance. With these insights, we achieve state-of-the-art performance on morphological inflection and historical text normalization. We also show that the transformer outperforms a strong baseline on two other character-level transduction tasks: grapheme-to-phoneme conversion and transliteration.

1 Introduction

The transformer (Vaswani et al., 2017) has become a popular architecture for sequence-to-sequence transduction in NLP. It has achieved state-of-the-art performance on a range of common word-level transduction tasks: neural machine translation (Barrault et al., 2019), question answering (Devlin et al., 2019) and abstractive summarization (Dong et al., 2019). In addition, the transformer forms the backbone of the widely-used BERT (Devlin et al., 2019). Yet for character-level transduction tasks like morphological inflection, the dominant model has remained a recurrent neural network-based sequence-to-sequence model with attention (Cotterell et al., 2018). This is not for lack of effort—but rather, it is the case that the transformer has consistently underperformed in experiments on average (Tang et al., 2018b).[1] As anecdotal evidence of this, we note that in the 2019 SIGMORPHON shared task on cross-lingual transfer for morphological inflection, no participating system was based on the transformer (McCarthy et al., 2019).

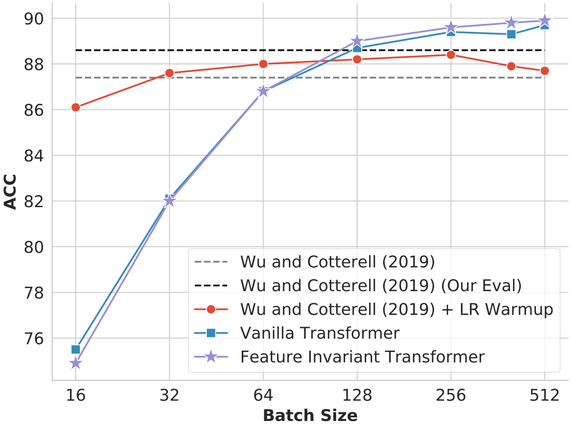


Figure 1: Development set accuracy for 5 languages on morphological inflection with different batch sizes. We evince our two primary contributions: (1) we set the **new state of the art** morphological inflection using the transformer and (2) we demonstrate the transformer's **dependence on the batch size**.

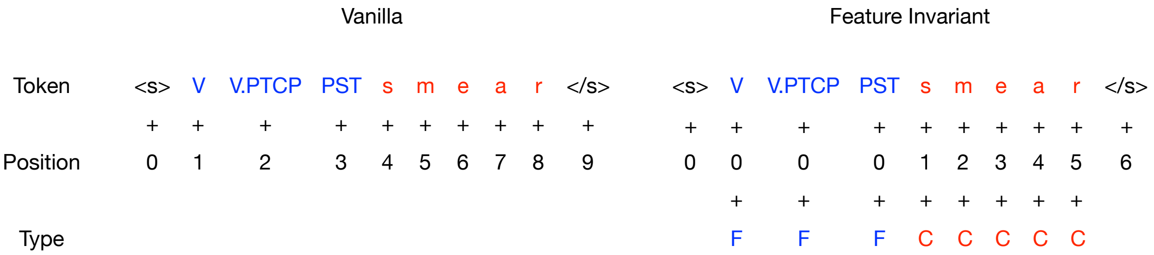
Character-level transduction models are often trained with less data than their word-level counterparts: In contrast to machine translation, where millions of training samples are available, the 2018 SIGMORPHON shared task (Cotterell et al., 2018) high-resource setting only provides ≈ 10k training examples per language. It is also not obvious that non-recurrent architectures such as the transformer

---

Code will be available at `https://github.com/shijie-wu/neural-transducer`.

[1] This claim is also based on the authors' personal communication with other researchers in morphology in the corridors of conferences and through email.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1901–1907
April 19 - 23, 2021.



Vanilla

Token	<s>	V	V.PTCP	PST	s	m	e	a	r	</s>
	+	+	+	+	+	+	+	+	+	+
Position	0	1	2	3	4	5	6	7	8	9

Feature Invariant

Token	<s>	V	V.PTCP	PST	s	m	e	a	r	</s>
	+	+	+	+	+	+	+	+	+	+
Position	0	0	0	0	1	2	3	4	5	6
		+	+	+	+	+	+	+	+	
Type		F	F	F	C	C	C	C	C	

Figure 2: Handling of feature-guided character-level transduction with special position and type embeddings in the encoder. `F` denotes features while `C` denotes characters. We use morphological inflection as an example, inflecting *smear* into its past participle form, *smeared*.

should provide an advantage at many character-level tasks: For instance, Gehring et al. (2017) and Vaswani et al. (2017) suggest that transformers (and convolutional models in general) should be better at remembering long-range dependencies. In the case of morphology, none of these considerations seem relevant: inflecting a word (a) requires little capacity to model long-distance dependencies and is largely a monotonic transduction; (b) it involves no semantic disambiguation, the tokens in question being letters; (c) it is not a task for which parallelization during training appears to help, since training time has never been an issue in morphology tasks.[2]

In this work, we provide state-of-the-art numbers for morphological inflection and historical text normalization, a novel result in the literature. We also show the transformer outperforms a strong recurrent baseline on two other character-level tasks: grapheme-to-phoneme (g2p) conversion and transliteration. We find that a single hyperparameter, batch size, is largely responsible for the previous poor results. Despite having fewer parameters, the transformer outperforms the recurrent sequence-to-sequence baselines on all four tasks. We conduct a short error analysis on the task of morphological inflection to round out the paper.

2 The Transformer for Characters

The Transformer. The transformer, originally described by Vaswani et al. (2017), is a self-attention-based encoder-decoder model. The encoder has N layers, consisting of a multi-head self-attention layer and a two-layer feed-forward layer with ReLU activation, both equipped with a skip connection. The decoder has a similar structure as the encoder except that, in each decoder layer between the self-attention layer and feed-forward layer, a multi-head attention layer attends to the output of the encoder. Layer normalization (Ba et al., 2016) is applied to the output of each skip connection. Sinusoidal positional embeddings are used to incorporate positional information without the need for recurrence or convolution. Here, we describe two modifications we make to the transformer for character-level tasks.

A Smaller Transformer. As the dataset sizes in character-level transduction tasks are significantly smaller than in machine translation, we employ a smaller transformer with $N = 4$ encoder-decoder layers. We use 4 self-attention heads. The embedding size is $d_{model} = 256$ and the hidden size of the feed-forward layer is $d_{FF} = 1024$. In the preliminary experiments, we found that using layer normalization before self-attention and the feed-forward layer performed slightly better than the original model. It is also the default setting of a popular implementation of the transformer (Vaswani et al., 2018). The transformer alone has around 7.37M parameters, excluding character embeddings and the linear mapping before the softmax layer. We decode the model left to right in a greedy fashion.

Feature Invariance. Some character-level transduction is guided by features. For example, in the case of morphological reinflection, the task requires a set of morphological attributes that control what form a citation form is inflected into (see Fig. 2 for an example). However, the order of the features is irrelevant. In a recurrent neural network, features are input in some predefined order as special characters and pre- or postpended to the input character sequence representing the citation form. The same is true for a vanilla transformer model, as shown on the left-hand side of Fig. 2. This leads to

---

[2] Many successful CoNLL–SIGMORPHON shared task participants report training their models on laptop CPUs.

LS	β_2	Vanilla	Feature Invariant
0	0.999	89.34	89.80
0	0.98	89.62	89.92
0.1	0.999	89.48	90.02
0.1	0.98	89.98	90.28

Table 1: Average development accuracy on morphological inflection with different LS and β_2, which denote hyperparameter of label smoothing and Adam optimizer respectively.

different relative distances between a character and a set of features.[3] To avoid such an inconsistency, we propose a simple remedy: We set the positional encoding of features to 0 and only start counting the positions for characters. Additionally, we add a special token to indicate whether a symbol is a word character or a feature. The right-hand side of Fig. 2 evinces how we have the same relative distance between characters and features.

3 Empirical Findings

Tasks. We consider four character-level transduction tasks: morphological inflection, grapheme-to-phoneme conversion, transliteration, and historical text normalization. For morphological inflection, we use the 2017 SIGMORPHON shared task data (Cotterell et al., 2017) with 52 languages. The performance is evaluated by accuracy (ACC) and edit distance (Dist). For the g2p task, we use the unstressed CMUDict (Weide, 1998) and NETtalk (Sejnowski and Rosenberg, 1987) resources. We use the splits from Wu et al. (2018). We evaluate under word error rate (WER) and phoneme error rate (PER). For transliteration, we use the NEWS 2015 shared task data (Zhang et al., 2015).[4] For historical text normalization, we follow Bollmann (2019) and use datasets for Spanish (Sánchez-Martínez et al., 2013), Icelandic and Swedish (Pettersson et al., 2013), Slovene (Scherrer and Erjavec, 2013, 2016; Ljubešic et al., 2016), Hungarian and German (Pettersson, 2016).[5] We evaluate using accuracy (ACC) and character error rate of incorrect prediction (CER$_i$).

Optimization. We use Adam (Kingma and Ba, 2014) with a learning rate of 0.001 and an inverse square root learning rate scheduler (Vaswani et al., 2017) with 4k steps during the warm-up. We train the model for 20k gradient updates and save and evaluate the model every 400 gradient updates. We select the best model out of 50 checkpoints based on development set accuracy. The number of gradient updates and checkpoints are roughly the same as Wu and Cotterell (2019), the single model state of the art on the 2017 SIGMORPHON dataset. We use their model as a baseline model. For all experiments, we use a single predefined random seed.

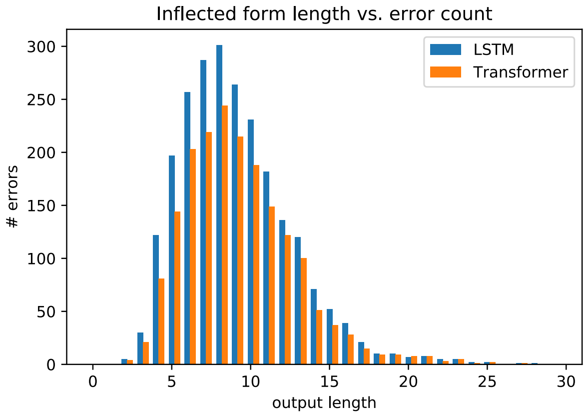


Figure 3: Distribution of incorrectly inflected forms in the test set of the inflection task over all 52 languages grouped by desired output word length.

3.1 A Controlled Hyperparameter Study

To demonstrate the importance of hyperparameter tuning for the transformer on character-level tasks, we perform a small controlled hyperparameter study. This is important since researchers had previously failed to achieve high-performing results with the transformer on character-level tasks. Here, we look at morphological inflection on the five languages in the 2017 SIGMORPHON dataset where submitted systems performed the worst: Latin, Faroese, French, Hungarian, and Norwegian (Nynorsk). We set the dropout to 0.3, β_2 of Adam to 0.999 (the default value), and do not use label smoothing. We do not tune any other hyperparameter except the following three hyperparameters.

The Importance of Batch Size. While recurrent models like Wu and Cotterell use a batch size of 20, halving the learning rate when stuck and employing early stopping, we find that a less aggressive learning rate scheduler, allowing the model to train longer, outperforms these hyperparameters. Fig. 1 shows that the *significant impact of batch size on the transformer*. The transformer performance in-

---

[3] While the features could be encoded with a binary vector followed by MLP, it introduces a representation bottleneck for encoding features.

[4] We do not have access to the test set.

[5] We do not include English due to licensing issues.

	ACC	Dist
Silfverberg et al. (2017)*	92.97	0.170
Wu et al. (2018)	93.60	0.128
Wu and Cotterell (2019)	94.40	0.113
Wu and Cotterell (2019) (Our eval)	94.81	0.123
Makarov et al. (2017)*	95.12	0.100
Bergmanis et al. (2017)*	95.32	0.100
Transformer (Dropout = 0.3)	**95.59**	**0.088**
Transformer (Dropout = 0.1)	95.56	0.090

Table 2: Average test performance on morphological inflection of Transformer against models from the literature. * denotes model ensembling.

creases steadily as the batch size is increased, similarly to what Popel and Bojar (2018) observe for machine translation. The transformer only outperforms the recurrent baseline when the batch size is at least 128, which is much larger than batch size commonly used in recurrent models.[6] Note that the model of Wu and Cotterell has 8.66M parameters, 17% more than the transformer model. To get an apples-to-apples comparison, we apply the same learning rate scheduler to Wu and Cotterell; this does not yield similar improvements and underperforms with respect to the traditional learning rate scheduler. Our feature invariant transformer also outperforms the vanilla transformer model. We set the batch size to 400 for our main experiments. Note the batch size of 400 is especially large (4% of training data) considering the training size is only 10k.

Other Hyperparameters. Vaswani et al. (2017) applies label smoothing (Szegedy et al., 2016) of 0.1 to the transformer model and shows that it hurts perplexity, but improves BLEU scores for machine translation. Instead of the default 0.999 β_2 for Adam, Vaswani et al. (2017) uses 0.98 and we find that both choices benefit character-level transduction tasks as well (see Tab. 1).

3.2 New State-of-the-Art Results

We train our feature invariant transformer on the four character-level tasks, exhibiting state-of-the-art results on morphological inflection and historical text normalization.

	ACC	CER$_i$	ACC^s	CER$_i^s$
Ljubešić et al. (2016)	91.78	0.392	90.37	0.360
Ljubešić et al. (2016) (LM)	91.56	0.399	89.93	0.368
Bollmann (2018)	91.27	0.381	89.73	0.350
Tang et al. (2018a)	91.67	0.389	90.32	0.358
Flachs et al. (2019)	-	-	90.06	-
Transformer (Dropout = 0.3)	91.30	**0.340**	89.99	**0.330**
Transformer (Dropout = 0.1)	**91.85**	0.352	**90.61**	0.334

Table 3: Average test performance on historical text normalization of Transformer against models from the literature. ^{s} denote subset of dataset as Flachs et al. (2019) only experiment with subset of languages.

	WER	PER	ACC	MFS
Wu et al. (2018)	28.20	**0.068**	41.10	0.894
Wu and Cotterell (2019)	28.20	0.069	41.20	0.895
Transformer (Dropout = 0.3)	28.08	0.070	**43.39**	**0.897**
Transformer (Dropout = 0.1)	**27.63**	0.069	41.35	0.891

Table 4: Average test performance on Grapheme-to-Phoneme and dev performance on Transliteration of Transformer against models from the literature.

Morphological Inflection. As shown in Tab. 2, the feature invariant transformer produces state-of-the-art results on the 2017 SIGMORPHON shared tasks, improving upon ensemble-based systems by 0.27 points. We observe that as the dataset decreases in size, a model with a larger dropout value performs slightly better. A brief tally of phenomena that are difficult to learn for many machine learning models, categorized along typical linguistic dimensions (such as word-internal sound changes, vowel harmony, circumfixation, ablaut, and umlaut phenomena) fail to reveal any consistent pattern of advantage to the transformer model. In fact, errors seem to be randomly distributed with an overall advantage of the transformer model. Curiously, errors grouped along the dimension of word length reveal that as word forms grow longer, the transformer advantage shrinks (Fig. 3).

Historical Text Normalization. Tab. 3 shows that the transformer model with dropout of 0.1, as in the case of morphological inflection, improves upon the previous state of the art, although the model with a dropout of 0.3 yields a slightly better CER$_i$.

G2P and Transliteration. Tab. 4 shows that the transformer outperforms previously published strong recurrent models on two tasks despite having fewer parameters. A dropout rate of 0.3 yields

---

[6] It is also large in the context of character-level tasks, which typically have around 10k training examples. Batch size of 400 would imply approximately 4% of training data in a single gradient update.

significantly better performance on the transliteration task while a dropout rate of 0.1 is stronger on the g2p task. This shows that transformers can and do outperform recurrent transducers on common character-level tasks when properly tuned.

4 Related Work

Character-level transduction is largely dominated by attention-based LSTM sequence-to-sequence (Luong et al., 2015) models (Cotterell et al., 2018). Character-level transduction tasks usually involve input-output pairs that share large substrings and alignments between these are often monotonic. Models that address the task tend to focus on exploiting such structural bias. Instead of learning the alignments, Aharoni and Goldberg (2017) use external monotonic alignments from the SIGMORPHON 2016 shared task baseline Cotterell et al. (2016). Makarov et al. (2017) use this approach to win the CoNLL-SIGMORPHON 2017 shared task on morphological inflection (Cotterell et al., 2017). Wu et al. (2018) shows that explicitly modeling alignment (hard attention) between source and target characters outperforms soft attention. Wu and Cotterell (2019) further shows that enforcing monotonicity in a hard attention model improves performance.

5 Conclusion

Using a large batch size and feature invariant input allows the transformer to achieve strong performance on character-level tasks. However, it is unclear what linguistic errors the transformer makes compared to recurrent models on these tasks. Future work should analyze the errors in detail as Gorman et al. (2019) does for recurrent models. While Wu and Cotterell shows that the monotonicity bias benefits character-level tasks, it is not evident how to enforce monotonicity on multi-headed self-attention. Future work should consider how to best incorporate monotonicity into the model, either by enforcing it strictly (Wu and Cotterell, 2019) or by pretraining the model to copy (Anastasopoulos and Neubig, 2019).

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2004–2015, Vancouver, Canada. Association for Computational Linguistics.

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39, Vancouver. Association for Computational Linguistics.

Marcel Bollmann. 2018. *Normalization of historical texts with neural network models*. Ph.D. thesis, Bochum, Ruhr-Universität Bochum.

Marcel Bollmann. 2019. A large-scale comparison of historical text normalization systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3885–3898, Minneapolis, Minnesota. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In

Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection, pages 1–30, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Simon Flachs, Marcel Bollmann, and Anders Søgaard. 2019. Historical text normalization with delayed rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1614–1619, Florence, Italy. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR.

Kyle Gorman, Arya D. McCarthy, Ryan Cotterell, Ekaterina Vylomova, Miikka Silfverberg, and Magdalena Markowska. 2019. Weird inflects but OK: Making sense of morphological generation errors. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 140–151, Hong Kong, China. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikola Ljubešic, Katja Zupan, Darja Fišer, and Tomaz Erjavec. 2016. Normalising Slovene data: historical texts vs. user-generated content. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, pages 146–155.

Nikola Ljubešić, Katja Zupan, Darja Fišer, and Tomaž Erjavec. 2016. Normalising Slovene data: historical texts vs. user-generated content. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, pages 146–155.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Peter Makarov, Tatiana Ruzsics, and Simon Clematide. 2017. Align and copy: UZH at SIGMORPHON 2017 shared task for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 49–57, Vancouver. Association for Computational Linguistics.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Eva Pettersson. 2016. *Spelling normalisation and linguistic analysis of historical text for information extraction*. Ph.D. thesis, Acta Universitatis Upsaliensis.

Eva Pettersson, Beáta Megyesi, and Jörg Tiedemann. 2013. An SMT approach to automatic annotation of historical text. In *Proceedings of the workshop on computational historical linguistics at NODALIDA 2013; May 22-24; 2013; Oslo; Norway. NEALT Proceedings Series 18*, 087, pages 54–69. Linköping University Electronic Press.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Felipe Sánchez-Martínez, Isabel Martínez-Sempere, Xavier Ivars-Ribes, and Rafael C. Carrasco. 2013. An open diachronic corpus of historical Spanish: annotation criteria and automatic modernisation of spelling. *arXiv preprint arXiv:1306.3692*.

Yves Scherrer and Tomaž Erjavec. 2013. Modernizing historical Slovene words with character-based smt. In *BSNLP 2013-4th Biennial Workshop on Balto-Slavic Natural Language Processing*.

Yves Scherrer and Tomaž Erjavec. 2016. Modernising historical Slovene words. *Natural Language Engineering*, 22(6):881–905.

Terrence J. Sejnowski and Charles R. Rosenberg. 1987. Parallel networks that learn to pronounce English text. *Complex Systems*, 1.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Gongbo Tang, Fabienne Cap, Eva Pettersson, and Joakim Nivre. 2018a. An evaluation of neural machine translation models on historical spelling normalization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1320–1331, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018b. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, et al. 2018. Tensor2tensor for neural machine translation. *arXiv preprint arXiv:1803.07416*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

R.L. Weide. 1998. The Carnegie Mellon pronouncing dictionary.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4425–4438, Brussels, Belgium. Association for Computational Linguistics.

Min Zhang, Haizhou Li, Rafael E. Banchs, and A Kumaran. 2015. Whitepaper of NEWS 2015 shared task on machine transliteration. In *Proceedings of the Fifth Named Entity Workshop*, pages 1–9, Beijing, China. Association for Computational Linguistics.

Exploring Supervised and Unsupervised Rewards in Machine Translation

Julia Ive[1], **Zixu Wang**[1], **Marina Fomicheva**[2], **Lucia Specia**[1,2,3]
Imperial College London[1], University of Sheffield[2], ADAPT - Dublin City University[3]
`j.ive@ic.ac.uk, zixu.wang@imperial.ac.uk`
`m.fomicheva@sheffield.ac.uk, l.specia@ic.ac.uk`

Abstract

Reinforcement Learning (RL) is a powerful framework to address the discrepancy between loss functions used during training and the final evaluation metrics to be used at test time. When applied to neural Machine Translation (MT), it minimises the mismatch between the cross-entropy loss and non-differentiable evaluation metrics like BLEU. However, the suitability of these metrics as reward function at training time is questionable: they tend to be sparse and biased towards the specific words used in the reference texts. We propose to address this problem by making models less reliant on such metrics in two ways: (a) with an entropy-regularised RL method that does not only maximise a reward function but also explore the action space to avoid peaky distributions; (b) with a novel RL method that explores a dynamic unsupervised reward function to balance between exploration and exploitation. We base our proposals on the Soft Actor-Critic (SAC) framework, adapting the off-policy maximum entropy model for language generation applications such as MT. We demonstrate that SAC with BLEU reward tends to overfit less to the training data and performs better on out-of-domain data. We also show that our dynamic unsupervised reward can lead to better translation of ambiguous words.

1 Introduction

Autoregressive sequence-to-sequence (seq2seq) neural architectures have become the *de facto* approach in Machine Translation (MT). Such models include Recurrent Neural Networks (RNN) (Sutskever et al., 2014; Bahdanau et al., 2014) and Transformer networks (Vaswani et al., 2017), among others. However, these models have as a serious limitation the discrepancy between their training and inference time regimes. They are traditionally trained using the Maximum Likelihood Estimation (MLE), which aims to maximise log-likelihood of a categorical ground truth distribution (samples in the training corpus) using loss functions such as cross-entropy, which are very different from the evaluation metric used at inference time, which generally compares string similarity between the system output and reference outputs. Moreover, during training, the generator receives the ground truth as input and is trained to minimise the loss of a single token at a time without taking the sequential nature of language into account. At inference time, however, the generator will take the previous sampled output as the input at next time step, rather than the ground truth word. MLE training thus causes: (a) the problem of "exposure bias" as a result of recursive conditioning on its own errors at test time, since the model has never been exclusively "exposed" to its own predictions during training; (b) a mismatch between the training objective and the test objective, where the latter relies on evaluation using discrete and non-differentiable measures such as BLEU (Papineni et al., 2002).

The current solution for both problems is mainly based on Reinforcement Learning (RL), where a seq2seq model (Sutskever et al., 2014; Bahdanau et al., 2014) is used as the policy which generates actions (tokens) and at each step receives rewards based on a discrete metric taking into account importance of immediate and future rewards. However, RL methods for seq2seq MT models also have their challenges: high-dimensional discrete action space, efficient sampling and exploration, choice of baseline reward, among others (Choshen et al., 2020). The typical metrics used as rewards (e.g., BLEU) are often biased and sparse. They are measured against one or a few human references and do not take into account alternative translation options that are not present in the references.

One way to address this problem is to use

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1908–1920
April 19 - 23, 2021.

entropy-regularised RL frameworks. They incorporate the entropy measure of the policy into the reward to encourage exploration. The expectation is that this leads to learning a policy that acts as stochastically as possible while able to succeed at the task. Specifically, we focus on the Soft Actor-Critic (SAC) (Haarnoja et al., 2018a,b) RL framework, which to the best of our knowledge has not yet been explored for MT, as well as other natural language processing (NLP) tasks. The main advantage of this architecture, as compared to other entropy regularised architectures (Haarnoja et al., 2017; Ziebart et al., 2008), is that it is formulated in the off-policy setting that enables reusing previously collected samples for more stability and better exploration. We demonstrate that SAC prevents the model from overfitting, and as a consequence leads to better performance on out-of-domain data.

Another way to address the problem of sparse or biased reward is to design an unsupervised reward. Recently, in Robotics, SAC has been successfully used in unsupervised reward architectures, such as the "Diversity is All You Need" (DIAYN) framework (Eysenbach et al., 2018). DIAYN allows the learning of latent-conditioned sub-policies ("skills") in unsupervised manner, which allows to better explore and model target distributions. Inspired by this work, we propose a formulation of an unsupervised reward for MT. We thoroughly investigate effects of this reward and conclude that it is useful in lexical choice, particularly the rare sense translation for ambiguous words.

Our **main contributions** are thus twofold: (a) the re-framing of the SAC framework such that it can be applied to MT and other natural language generation tasks (Section 3). We demonstrate that SAC results in improved generalisation compared to the MLE training, leading to better translation of out-of-domain data; (b) the proposal of a dynamic unsupervised reward within the SAC framework (Section 3.4). We demonstrate its efficacy in translating ambiguous words, particularly the rare senses of such words. Our datasets and settings are described in Section 4, and our experiments in Section 5.

2 Related Work

Reinforcement Learning for MT RL has been successfully applied to MT to bridge the gap between training and testing by optimising the sequence-level objective directly (Yu et al., 2017; Ranzato et al., 2015; Bahdanau et al., 2016). However, thus far mainly the REINFORCE (Williams, 1992) algorithm and its variants have been used (Ranzato et al., 2015; Kreutzer et al., 2018). These are simpler algorithms that handle the large natural language action space, but they employ a sequence-level reward which tends to be sparse.

To reduce model variance, Actor-Critic (AC) models consider the reward at each decoding step and use the Critic model to guide future actions (Konda and Tsitsiklis, 2000). This approach has also been explored for MT (Bahdanau et al., 2016; He et al., 2017). However, more advanced AC models with Q-Learning are rarely applied to language generation problems. This is due to the difficulty of approximating the Q-function for the large action space. The large action space is one of the bottleneck for RL for text generation in general. Pre-training of the agent parameters to be close to the true distribution is thus necessary to make RL work (Choshen et al., 2020). Further RL training of the agent makes the overfitting problem even more pronounced resulting in peaky distributions. Such problems are traditionally addressed by entropy regularised RL.

Entropy Regularised RL The main goal of this type of RL is to learn an efficient policy while keeping the entropy of the agent actions as high as possible. The paradigm promotes exploration of actions, suppresses peaky distributions and improves robustness. In this work, we explore the effectiveness of the maximum entropy SAC framework (Haarnoja et al., 2018a).

The work closest to ours is of Dai et al. (2018) where the Entropy-Regularised AC (ERAC) model leads to better MT performance. The major difference between ERAC and SAC is that the former is an on-policy model and the latter is an off-policy model. On-policy approaches use consecutive samples collected in real-time that are correlated to each other. In the off-policy setting, our SAC algorithm uses samples from the memory that are taken uniformly with reduced correlation. This key characteristic of SAC ensures better model generalisation and stability (Mnih et al., 2015). There are also differences in the architectures of SAC and ERAC, i.a., using 4 Q-value networks instead of two. These differences will be covered in detail in Section 3.

Unsupervised reward RL Significant work has been done in Robotics to improve the learning capability of robots. These approaches do not rely on a single objective but rather promote intrinsic motivation and exploration. Such an approach to learn diverse skills (latent-conditioned sub-policies, in practice, skills like walking or jumping) in unsupervised manner was recently proposed by Eysenbach et al. (2018). The approach relies on the SAC model and inspired our approach to designing our unsupervised reward for MT. We are not aware of other attempts to design dynamic unsupervised RL rewards (learnt together with the network) in seq2seq in general, or MT in particular. Recent work on unsupervised rewards in NLP (Gao et al., 2020) explores mainly static rewards computed against synthetic references.

3 Methodology

In this section we start by describing the underlying MT architecture and its variant using RL, to then introduce our SAC formulation and the reward functions used.

3.1 Neural Machine Translation (NMT)

A typical Neural Machine Translation (NMT) system is a seq2seq architecture (Sutskever et al., 2014; Bahdanau et al., 2014), where each source sentence $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ is encoded by the encoder into a series of hidden states. At each decoding step t, a target word y_t is generated according to $p(y_t|y_{<t}, x)$ conditioned on the input sequence x and decoded sequence $\mathbf{y}_{<t} = (y_1, \cdots, y_{t-1})$ up to the t-th time step. Given the corpus of pairs of source and target sentences $\{x_i, y_i\}_{i=1}^{N}$, the training objective function - maximum likelihood estimation (MLE) is defined as:

$$\mathcal{L}_{\text{MLE}} = -\sum_{i=1}^{N}\sum_{t=1}^{T} p(y_t^i|y_1^i, ..., y_{t-1}^i, x^i) \tag{1}$$

3.2 Reinforcement Learning for NMT

Within the RL framework, the task of NMT can be formulated as a sequential decision making process, where the **state** is defined by the previously generated words ($\mathbf{y}_{<t}$) and the **action** is the next word to be generated. Given the state s_t, the agent picks an action a_t (for seq2seq it is the same as y_t), according to a (typically stochastic) **policy** π_θ and observes a reward r_t for that action. The reward can be calculated based on any evaluation metric, *e.g.* BLEU.

The objective of the RL training is to maximise the expected reward:

$$\mathcal{L}_{RL} = \mathbb{E}_{a_1,\cdots,a_T \sim \pi_\theta(a_1,\cdots,a_T)}[r(a_1, \cdots, a_T)] \tag{2}$$

Under the policy π, we can also define the values of the state-action pair $Q(s_t, y_t)$ and the state $V(s_t)$ as follows:

$$\begin{aligned} Q_\pi(s_t, a_t) &= \mathbb{E}[r_t|s = s_t, a = a_t] \\ V_\pi(s_t) &= \mathbb{E}_{a\sim\pi(s)}[Q_\pi(s_t, a = a_t)] \end{aligned} \tag{3}$$

Intuitively, the value function V measures how good the model could be when it is in a specific state s_t. The Q function measures the value of choosing a specific action when we are in such state.

Given the above definitions, we can define a function called *advantage* – denoted by A_π – relating the value function V and Q function as follows:

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \tag{4}$$

Therefore, the focus is on maximising one of the following objectives:

$$\max_a A_\pi(s_t, a_t) \rightarrow \max_a Q_\pi(s_t, a_t) \tag{5}$$

Different RL algorithms have different ways to search for the optimal policy. Algorithms such as REINFORCE, as well as its variant MIXER (Ranzato et al., 2015), popular in language tasks, search for the optimal policy via Eq. 2 using the Policy Gradient. Actor-Critic (AC) models typically improve the performance of Policy Gradient models by solving Eq. 5 (left part) (Bahdanau et al., 2016). Q-learning models that aim at maximising the Q function (Eq 5, right part) to improve over both the Policy Gradient and AC models (Dai et al., 2018).

3.3 Soft Actor-Critic (SAC)

The SAC algorithm (Haarnoja et al., 2018a) adds to the Eq. 2 an entropy term:

$$\mathcal{L}(\pi) = \sum_{t=1}^{T} \mathop{\mathbb{E}}_{a_t\sim\pi(\cdot|s_t)} [r(s_t, a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))] \tag{6}$$

where α controls the stochasticity of the optimal policy, a trade-off between the relative importance of the entropy term $\mathcal{H}$ and the reward $r(s_t, a_t)$ that the agent receives by taking action a_t when the state of the environment is s_t. Its aim is to maximise the entropy of actions at the same time as maximising the rewards.

As mentioned earlier, SAC is an off-policy Q-learning AC algorithm. As other AC algorithms it consists of two parts: the actor (the policy function) and the critic – action-value function (Q), parameterised by ϕ and θ, respectively.

During off-policy learning, the history of states, actions and respective rewards are stored in a memory (D), *a.k.a.* the replay buffer.

- **Critic Training**

The Q-function estimates the value of an action at a given state based on its future rewards. The soft-Q value is computed recursively by applying a modified Bellman backup operator:

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathop{\mathbb{E}}_{s_{t+1} \sim D}[V(s_{t+1})] \tag{7}$$

where

$$V(s_t) = \mathop{\mathbb{E}}_{a_t \sim \pi}[Q(s_t, a_t) - \alpha \log \pi(a_t|s_t)] \tag{8}$$

is the expected future reward of a state and $\log(\pi(a_t|s_t))$ is the entropy of the policy.

The parameters of the Q-function are updated towards minimising the mean squared error between the estimated Q-values and the assumed ground-truth Q-value. The assumed ground-truth Q-values are estimated based on the current reward ($r(s_t, a_t)$) and the discounted future reward of the next state ($\gamma V_{\bar{\theta}}(s_{t+1})$). This mean squared error objective function of the Q network is as follows:

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{s_t, a_t, r_t, s_{t+1} \sim D, a_{t+1} \sim \pi_\phi} \Big[\big(Q_\theta(a_t, s_t) - \\ [r(s_t, a_t) + \gamma \mathop{\mathbb{E}}_{s_{t+1} \sim D}[V_{\bar{\theta}}(s_{t+1})]]\big)^2 \Big] \tag{9}$$

Note that the parameters of the networks are denoted as θ and $\bar{\theta}$ respectively. This is the best practice where the critic is modeled with two neural networks with the exact same architecture but independent parameters (Mnih et al., 2015).

The parameters of the target critic network ($Q_{\bar{\theta}}$) are iteratively updated with the exponential moving average of the parameters of the main critic network (Q_θ). This constrains the parameters of the target network to update at a slower pace toward the parameters of the main critic, which has been shown to stabilise the training process (Lillicrap et al., 2016).

Another advantage of SAC is the double Q-learning (Hasselt, 2010). In this approach, two Q networks for both of the main and the target critic functions are maintained. When estimating the current Q values or the discounted future rewards, the minimum of the outputs of the two Q networks is used. Thus the estimated Q values do not grow too large, which improves the policy training (Haarnoja et al., 2018a).

- **Actor Training**

SAC updates the policy to minimise the KL-divergence to make the distribution of $\pi_\phi(s_t)$ policy function look more like the distribution of the Q function:

$$\mathcal{L}_\pi(\phi) = \mathop{\mathbb{E}}_{s_t \sim D}[\pi_t(s_t)^T [\alpha \log(\pi_\phi(s_t)) - Q_\theta(s_t)]] \tag{10}$$

where softmax is used in the final layer of the policy to output a probability distribution over the actions.

We note that some versions of the SAC algorithm allow to automatically tune the α parameter so that while maximising the expected return, the policy should satisfy the minimum entropy criteria. In our experiments we however used a fixed α. Updating α during training resulted in too short sentences in the output.

Finally, we note that Eq. 10 does *not* simply add an entropy term to the standard Policy Gradient. The critic Q_θ trained by Eq. 9 additionally captures the *entropy from future steps*.

For more details on SAC for the discrete setting (like MT) we refer to Christodoulou (2019). For more formal details on the architecture, see Haarnoja et al. (2018a,b).

3.4 Reward functions

Below we define the reward functions we use in our SAC architecture.

Supervised BLEU reward: - `SAC BLEU` In the supervised setup, we employ the sequence-level BLEU score (Papineni et al., 2002) with add-1 smoothing (Chen and Cherry, 2014). As an additional length constraint at each time step, we deduct from the respective score the length penalty: $lp = |l_{\mathbf{y}} - l_{\hat{\mathbf{y}}}|$, where $\mathbf{y}$ is the reference translation. This penalty prevents longer translations that are not penalised by the brevity penalty of BLEU.

BLEU has been chosen in our study to ensure better comparability with the related work in RL MT traditionally using the BLEU reward (Bahdanau et al., 2016; Dai et al., 2018).

Unsupervised reward - `SAC unsuper` As discussed above, using automatic metrics as reward function can lead to a number of issues, *e.g.* reward sparsity, overfitting towards single reference. Moreover, designing a good reward can be challenging.

Inspired by recent work on the SAC algorithm in unsupervised RL (Eysenbach et al., 2018), we have designed an unsupervised reward that *balances the quality and diversity in the model search space*.

The pseudo-reward function we use is as follows:

$$r_z(\mathbf{x}, a) = \log q_\delta(z|\mathbf{x}, a) - \log p(z) \qquad (11)$$

where $p(z)$ is a categorical uniform distribution for a latent variable z.

$q_\delta(z|\mathbf{x}, a)$ is provided by a discriminator parametrised by a neural network. z is randomly assigned to a word sampled at each step from the actor distribution. The discriminator is a Bag-of-Words model that takes as input the encoded source sequence and the word itself to predict its z.

More intuitively, every time a word appears in the translation hypothesis for a source sentence (within the Bag-of-Words formulation) it is randomly assigned a certain value of z. The more times this word appears in the sampled hypotheses (for a given source) the closer will be $\log q_\delta(z|\mathbf{x}, a)$ to the uniform prior $p(z)$, hence reward $r_z(\mathbf{x}, a)$ will be close to 0. Thus, frequent translations will be suppressed and search for less frequent translations will be encouraged in order to receive a reward larger than 0.

Such a reward is less sparse than the traditional ones and is also dynamic which prevents memorising and overfitting.

4 Experimental Setup

4.1 Data

We perform experiments on the **Multi30K** dataset (Elliott et al., 2016)[1] of image description translations and focus on the English-German (EN-DE) and English-French (EN-FR) (Elliott et al., 2017) language directions. Following best practises, we use sub-word segmentation (BPE (Sennrich et al., 2016)) only on the target side of the corpus. The dataset contains 29,000 instances for training, 1,014 for development, and 1,000 for testing. We use **flickr2016** (**2016**), **flickr2017** (**2017**) and **coco2017** (**COCO**) test sets for model evaluation.

2016 is the most **in-domain** test set since it was taken from the same superset of descriptions as the training set, whereas **2017** and **COCO** are from different image description corpora and are thus considered **out-of-domain**.

For more fine-grained assessment of our models with unsupervised reward, we use the **MLT** test set (Lala and Specia, 2018; Lala et al., 2019), an annotated subset of the **Multi30K** corpus where each instance is a 3-tuple consisting of an **ambiguous** source word, its textual context (a source sentence), and its correct translation. The test set contains 1,298 sentences for English-French and 1,708 for English-German. It was designed to benchmark models in their ability to select the right lexical choice for words with multiple translations, especially when some of these translations are rarer.

Additionally, to allow for comparison with previous work, we evaluate on the **IWSLT 2014** German-to-English dataset (Cettolo et al., 2012) from TED talks, which has been used as testbed in most work on RL for MT. The training set contains $153K$ sentence pairs. We followed the pre-processing procedure described in (Dai et al., 2018).

When compared to the **IWSLT 2014** dataset, all the three **Multi30K** test sets are more out-of-domain. This was found by the analysis of perplexities of language models trained with respective training data for each dataset (see Appendix A.4).

4.2 Training

We modify the original SAC architecture to adapt it to MT following best practices (Bahdanau et al., 2016) in the area. The functions π_ϕ and Q_θ are parameterised with neural networks: π_ϕ is an RNN seq2seq model with a 2-layer GRU (Cho et al., 2014) encoder and a 2-layer Conditional GRU decoder (Sennrich et al., 2017) with attention (Bahdanau et al., 2014). For `SAC BLEU`, Q_θ duplicates the structure of the former, but encodes the reference instead of the source sentence to mimic inputs to the actual BLEU function.

We first pretrain the actor and then pretrain the critic, before the actor-critic training. The pretraining of actors is done until convergence according

---

[1] https://github.com/multi30k/dataset

		2016			2017			COCO		
	model	BLEU	METEOR	TER	BLEU	METEOR	TER	BLEU	METEOR	TER
EN-FR	MLE	57.5	71.7	27.5	50.9	66.8	33.0	42.8	61.5	37.3
	ERAC (ours)	**59.4***	**73.3***	**26.7***	51.2	66.8	32.5	42.5*	60.6*	37.6*
	`SAC BLEU`	57.9	72.0	27.8	**51.7***	**67.5***	**32.1***	**44.4**	**62.9**	**36.4**
	`SAC unsuper`	56.9	71.4	28.2	51.1	67.1	32.5	43.6	62.6*	36.6
EN-DE	MLE	38.5	**57.2**	42.2	**31.9**	**51.3**	49.5	27.2	46.7	55.2
	ERAC (ours)	**38.9***	56.1*	**41.9***	31.4*	49.6*	49.7*	25.0	44.0	56.0*
	`SAC BLEU`	38.1	56.8*	42.5	**31.9**	51.2	**49.1**	**27.7**	**47.0**	**54.5**
	`SAC unsuper`	38.0*	56.9	43.0*	31.6	50.8	49.7*	26.6	46.5	55.1

Table 1: Performance of `SAC BLEU` on the **Multi30K** test sets (EN-FR, EN-DE) trained on the **Multi30K** train set. * marks statistically significant changes (p-value ≤ 0.05) as compared to MLE. Bold highlights best results. ERAC (ours) indicates results obtained by us using the code openly provided by Dai et al. (2018).

to the early stopping criteria of 10 epochs wrt. to the MLE loss. We have also found that our critics require much less pretraining (3-5 epochs as compared to 10-20 epochs in general for AC architectures with the MSE loss). Also, to prevent divergence during the actor-critic training, we continue performing MLE training using a smaller weight λ_{mle}. We set α to 0.01. Following Haarnoja et al. (2018a), we rescale the reward to the value inverse to α. Note that we did not find it useful to add to SAC the smoothing objective minimising variance of Q-values (Bahdanau et al., 2016; Dai et al., 2018). We presume that the double Q-learning significantly contributes to the stability of the network and additional smoothing is not required.

For `SAC unsuper`, we parameterise q_δ by a 2-layer feed-forward neural network, which takes the source as encoded by the actor and a_t and outputs $q_\delta(z|\mathbf{x}, a)$. We set z to take one of 4 values.[2] For this unsupervised setting, we do not train a Q-function. We instead operate in the oracle mode and following (Keneshloo et al., 2018) define true Q-value estimates and use it to update our actor. Details on training are given in Appendix A. We use `pysimt` (Caglayan et al., 2020) with PyTorch (Paszke et al., 2019) v1.4 for our experiments.[3]

4.3 Evaluation

We use the standard set of MT evaluation metrics: BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014) and TER (Snover et al., 2006). We perform significance testing via bootstrap resampling using the `Multeval` tool (Clark et al., 2011).

For the lexical translation task, we measure the **Lexical Translation Accuracy (LTA)** score (Lala et al., 2019). The score provides an average estimation of how accurately the words have been translated. For each ambiguous word, a score of +1 is awarded if the correct translation of the word is found in the output translation; a score of 0 is assigned if a known incorrect translation is found, or none of the candidate words are found in the translation. We also propose a metric that not only rewards correctly translated ambiguous words, but also penalises words translated with the wrong sense: the **Ambiguous Lexical Index (ALI)**. ALI assigns -1 for wrong translations in the given context, whereas LTA simply does not reward them.

5 Results

5.1 Comparison to state-of-the-art

We first compare our SAC models against the MLE model (baseline) and ERAC[4] (state-of-the-art – SOTA) both trained and tested on the **Multi30K** data (Table 1). Compared to SAC, ERAC differs in that it uses the on-policy setting (i.e., using samples collected in real time). Our SAC algorithm is an off-policy algorithm and uses samples from the memory to promote generalisation.

We clearly observe the tendency of ERAC models to perform better on the more in-domain **2016** data (+1.9 BLEU, +1.6 METEOR, -0.8 TER

---

[2] This hyperparameter is tuned on the validation set. It typically varies from 2 to several hundreds in the related work (Haarnoja et al., 2018b).

[3] https://github.com/ImperialNLP/pysimt

[4] For ERAC, we present results that we reproduced ourselves using the code publicly provided by the authors. We had to perform several modifications to this code to make it conform recent deep learning framework software updates. The performance of this model is on pair with this reported by the authors.

		2016			2017			COCO		
	Model	BLEU	METEOR	TER	BLEU	METEOR	TER	BLEU	METEOR	TER
UNK	MLE	25.1	**29.1**	**49.9**	23.1	**27.7**	54.5	18.9	**25.8**	59.2
	SAC BLEU	**25.2**	28.9	50.1	**23.2**	27.5	54.5	**19.4**	25.5*	**58.3***
noUNK	MLE	34.4	37.8	40.4	31.6	**37.7**	46.4	25.9	34.2	50.6
	SAC BLEU	**34.9**	**38.0**	**40.0**	**32.1**	37.6	**45.9**	**28.3***	**34.5**	**48.6**

Table 2: Performance of SAC BLEU on **Multi30K** (German-English) trained on the **IWSLT 2014** train set. UNK indicates standard output containing the UNK symbol; noUNK – outputs with sentences containing UNK not taken into account. * marks statistically significant changes (p-value $\leq$ 0.05) as compared to MLE. Bold highlights best results.

against MLE for EN-FR) and the tendency of SAC BLEU models to outperform other models on more out-of-domain **2017** and **COCO** sets (+2.7 BLEU and +3.0 METEOR, -1.5 TER against ERAC on **COCO** for EN-DE).

SAC unsuper results are however worse than the baseline and SOTA. We focus thus on the investigation of SAC BLEU and come back to SAC unsuper in Section 5.2.

To further confirm our hypothesis that SAC reduces overfitting and performs better on the out-of-domain data, we train our models on the **IWSLT 2014** train set and test on the out-of-domain **Multi30K** test sets (in the reverse direction, German into English, Table 2).

We observe similar performance for complete set of outputs (including sentences with UNK tokens) for MLE and SAC BLEU. If the lines with UNK words are not taken into account,[5] we observe an improvement for the **2016** and **2017** test sets (+0.5 BLEU, +0.1 METEOR, -0.5 TER on average), and a much bigger improvement for the more out-of-domain **COCO** set (+2.5 BLEU, +0.3 METEOR, -2 TER on average). This confirms our hypothesis that SAC helps to reduce overfitting.

Finally, we compare SAC to the SOTA AC-base RL architectures, namely ERAC and AC, on the **IWSLT 2014** set that is commonly used for this task. Compared to SAC, AC differs in that it does not use entropy regularisation. We also provide the performance for the popular MIXER algorithm. Results are shown in Table 3.

In terms of the general performance, our SAC performs on pair with the MLE model. SAC BLEU even slightly lowers this score (-0.2 BLEU, -0.2 METEOR). We note that SAC BLEU results contain an increased count of UNK words as compared to MLE (+2.8%) This increased generation of UNK words due to the entropy regularisation is partially responsible for this similar performance. Another cause is that SAC does not overfit to the BLEU distribution of the target data.[6]

Model	BLEU	METEOR	TER
MLE (ours)	29.8	31.2	48.9
MIXER (Ranzato et al., 2015)	20.73	-	-
AC (Bahdanau et al., 2016)	28.53	-	-
ERAC (w/feed) (Dai et al., 2018)	29.36	-	-
ERAC (w/o feed) (Dai et al., 2018)	28.42	-	-
ERAC (w/o feed, ours)	29.0*	30.6*	51.5*
SAC BLEU	**29.6***	**31.0***	**48.8***

Table 3: Performance of MLE and different RL algorithms on the **IWSLT 2014** test set trained on the **IWSLT 2014** train set. * marks statistically significant changes (p-value $\leq$ 0.05) as compared to MLE. Bold highlights best RL results. MIXER, AC and ERAC scores were taken from original papers. ERAC (ours) indicates our results using the code provided in (Dai et al., 2018).

5.2 Translation of ambiguous words

To further investigate the effect of the unsupervised reward, we have evaluated SAC unsuper on the **MLT** dataset. Results are shown in Table 4. We calculate the scores on two conditions: **All Cases** takes into account all possible lexical translations; while for **Rare Cases**, only the instances where the gold-standard translation is not the most frequent translation for that particular ambiguous word. We observe that both SAC BLEU and SAC unsuper

---

[5] The original corpus pre-processing pipeline that we followed to increase comparability does not include subword segmentation. We take the intersection of hypotheses sentences across **Multi30K** test setups that contain no generated UNK token wrt. the **IWSLT 2014** vocabulary. Reference files may still contain the UNK token, we focus on the generated text here.

[6] We mean that the model would have a tendency to select certain words to simply boost BLEU rather than picking words to reflect the correct meaning.

		All Cases						**Rare Cases**					
		2016		**2017**		**COCO**		**2016**		**2017**		**COCO**	
	Model	LTA	ALI	LTA	ALI	LTA	ALI	LTA	ALI	LTA	ALI	LTA	ALI
EN-FR	MLE	81.60	63.19	79.65	59.31	74.60	49.21	52.81	24.49	**47.80**	**16.48**	47.16	**18.49**
	`SAC BLEU`	81.94	63.89	79.76	59.53	**77.32**	**54.65**	53.37	25.39	45.91	13.46	**49.05**	15.47
	`SAC unsuper`	**82.75**	**65.51**	**80.62**	**61.25**	75.28	50.57	**54.49**	**27.19**	**47.80**	**16.48**	47.16	15.47
EN-DE	MLE	65.34	30.68	70.91	41.82	67.45	34.91	50.95	11.72	60.00	28.00	56.56	21.82
	`SAC BLEU`	64.74	29.48	71.93	43.86	**67.72**	**35.43**	50.14	10.24	60.58	29.04	**58.58**	**25.45**
	`SAC unsuper`	**65.54**	**31.08**	**73.41**	**46.82**	66.40	32.81	**51.50**	**12.70**	**63.77**	**34.78**	52.52	14.55

Table 4: Performance of `SAC BLEU` on the **MLT** test sets (EN-FR, EN-DE). We report Ambiguous Words Accuracy: LTA and ALI. **Rare Cases** indicates the cases where the correct translation is *not* the most frequent translation in the training set.

outperform the MLE baseline across metrics in all setups except for the **COCO** EN-FR translation in Rare Cases, where MLE performs better. For `SAC BLEU`, this observation is also shown by general evaluation metrics BLEU, METEOR and TER on all **MLT** test sets (see Table 9 in Appendix).

Moreover, `SAC unsuper` is particularly successful when evaluated on **2016** and **2017** and outperforms both MLE and `SAC BLEU` across setups. This demonstrates the potential of the unsupervised reward function for the cases when we have to choose between possible translations for an ambiguous word (i.e., better exploration of the search space). BLEU reward, on the other hand, is more reliable when we have to adjust distributions to produce one single possible translation. Manual inspection of these `SAC unsuper` improvements confirmed their increased accuracy (see Table 5). For example, the ambiguous French source word 'hill' ('colline') is translated as 'pente'('slope') by both MLE and `SAC BLEU`, while only `SAC unsuper` produces the correct sentence: 'adolescent saute la *colline* 'hill' avec son vélo'.

5.3 Qualitative analysis

To get further insights into the general results, we also performed human evaluation of the outputs for MLE, `SAC BLEU`, and `SAC unsuper` using professional in-house expertise. This was done for **COCO** EN-FR and **2016** EN-DE as two sets with contrastive results in the lexical translation experiment.

For this human analysis, we randomly selected test samples (50 samples per language pair per group) with source words of different frequency in the training data: rare words (frequency 1) and other words (frequency ≥ 10). These other words are randomly chosen from the sentences that differ in their translation across setups. The resulting average frequency of those words is around 40 for both language pairs. A rank of quality (both fluency and adequacy together) is assigned by the human evaluator from 1 to 3, allowing ties. Following the common practice in MT, each system was then assigned a score which reflects how often it was judged to be better or equal to other systems (Bojar et al., 2017).

Results are in Table 6. We observe a tendency of `SAC BLEU` to do well on the translation of rare source words, but not so well on the translation of words in the middle frequency range (this observation is confirmed by the analysis of the frequency of output words, see Appendix A.5, see Table 10). Our unsupervised reward tends to increase the performance on more frequent words ('Other' in Table 6) by promoting their less common translations in the distribution, hence better translations for ambiguous words from our previous experiment. These ambiguous words are quite frequent, they potentially have multiple possible translations but only one correct translation in a given context.

6 Conclusions

We propose and reformulate SAC reinforcement learning approaches to help machine translation through better exploration and less reliance on the reward function. To provide a good trade-off between exploration and quality, we devise two reward methods in the supervised and dynamic unsupervised manner. The maximum entropy off-policy SAC algorithm mitigates the overfitting problem when evaluated in the out-of-domain space; both rewards introduced in our SAC architecture can achieve better quality for lexical translation of ambiguous words, particularly the rare senses of

EN-FR	source word	hill
	gold target word	colline
	source sentence	the teen jumps the **hill** with his bicycle .
	reference sentence	ado saute sur la **colline** 'hill' avec son vélo .
	MLE	adolescent saute sur la **pente** 'slope' avec son vélo .
	`SAC BLEU`	adolescent saute la **pente** 'slope' avec son vélo .
	`SAC unsuper`	adolescent saute la **colline** 'hill' avec son vélo .
EN-DE	source word	outfit
	gold target word	outfit
	source sentence	a rhythmic gymnast in a blue and pink **outfit** performs a ribbon routine .
	reference sentence	eine rhythmische sportgymnastin in einem blauen und pinken **outfit** vollführt eine bewegung mit dem band .
	MLE	ein begeisterter turner in blau-rosa **kleidung** 'dress' führt eine band auf .
	`SAC BLEU`	ein begeisterter turner in blau-rosa **kleidung** 'dress' führt eine band auf .
	`SAC unsuper`	ein aufgeregter turner in einem blau-rosa **outfit** führt eine band aus .

Table 5: Samples of ambiguous words translation on **2016** for both EN-FR and EN-DE. In both cases more correct translations are provided by `SAC unsuper`. Bold highlights target words and their translations.

Lang	Words	MLE	`SAC BLEU`	`SAC unsuper`
EN-FR	Rare (Freq. 1)	1.76	**1.88**	1.68
	Other	**1.88**	1.82	1.86
EN-DE	Rare (Freq. 1)	1.72	**1.74**	1.70
	Other	1.93	1.83	**1.94**

Table 6: Human ranking results for **2016** EN-DE and **COCO** EN-FR test set. Bold highlights best results per group of word types. The first column indicates the groups of word types. Results are averaged for all words per word type group.

words. The formulation of the unsupervised reward and its potential to influence translation quality open perspectives for future studies on the subject. We leave the exploration of how those supervised and unsupervised rewards could be combined to improve MT for future work.

Acknowledgments

The authors thank the anonymous reviewers for their useful feedback. This work was supported by the MultiMT (H2020 ERC Starting Grant No. 678017) and the Air Force Office of Scientific Research (under award number FA8655-20-1-7006) projects. Marina Fomicheva and Lucia Specia were supported by funding from the Bergamot project (EU H2020 grant no. 825303). We also thank the annotators for their valuable help.

References

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. Cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Ozan Caglayan, Julia Ive, Veneta Haralampieva, Pranava Madhyastha, Loïc Barrault, and Lucia Specia. 2020. Simultaneous machine translation with visual context. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2350–2361, Online. Association for Computational Linguistics.

Mauro Cettolo, C. Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. *Proceedings of EAMT*, pages 261–268.

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level BLEU. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 362–367, Baltimore, Maryland, USA. Association for Computational Linguistics.

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

Leshem Choshen, Lior Fox, Zohar Aizenbud, and Omri Abend. 2020. On the weaknesses of reinforcement learning for neural machine translation. In *International Conference on Learning Representations*.

Petros Christodoulou. 2019. Soft actor-critic for discrete action settings. *arXiv preprint arXiv:1910.07207*.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA. Association for Computational Linguistics.

Zihang Dai, Qizhe Xie, and Eduard Hovy. 2018. From credit assignment to entropy regularization: Two new algorithms for neural sequence prediction. *arXiv preprint arXiv:1804.10974*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the second shared task on multimodal machine translation and multilingual image description. In *Proceedings of the Second Conference on Machine Translation*, pages 215–233, Copenhagen, Denmark. Association for Computational Linguistics.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. 2018. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. *CoRR*, abs/1702.08165.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018a. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. 2018b. Soft actor-critic algorithms and applications. *CoRR*.

Hado V. Hasselt. 2010. Double q-learning. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2613–2621. Curran Associates, Inc.

Di He, Hanqing Lu, Yingce Xia, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2017. Decoding with value networks for neural machine translation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 178–187. Curran Associates, Inc.

Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K. Reddy. 2018. Deep reinforcement learning for sequence to sequence models. *arXiv preprint arXiv:1805.09461*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Vijay R. Konda and John N. Tsitsiklis. 2000. Actor-critic algorithms. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 1008–1014. MIT Press.

Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. 2018. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Melbourne, Australia. Association for Computational Linguistics.

Chiraag Lala, Pranava Madhyastha, and Lucia Specia. 2019. Grounded word sense translation. In *Proceedings of the Second Workshop on Shortcomings in Vision and Language*, pages 78–85, Minneapolis, Minnesota. Association for Computational Linguistics.

Chiraag Lala and Lucia Specia. 2018. Multimodal Lexical Translation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *ICLR*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

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

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1310–1318, Atlanta, Georgia, USA. PMLR.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Valencia, Spain. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-first AAAI conference on artificial intelligence*.

Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. 2008. Maximum entropy inverse reinforcement learning. pages 1433–1438. AAAI Press.

A Training Details

A.1 Hyperparameters

For the NMT RNN agent, the dimensions of embeddings and GRU hidden states are set to 200 and 320, respectively. The decoder's input and output embeddings are shared (Press and Wolf, 2017). We use Adam (Kingma and Ba, 2014) as the optimiser and set the learning rate and mini-batch size to 0.0004 and 64, respectively. A weight decay of $1e{-}5$ is applied for regularisation. We clip the gradients if the norm of the full parameter vector exceeds 1 (Pascanu et al., 2013). The four Q-networks are identical to the agent (see Table 7).

For the unsupervised reward setting, we use 2 two-layer feed-forward neural network (both dimensionalities are equal to 100). We use again Adam as the optimiser and set the learning rate and mini-batch size to 0.0001 and 64, respectively.

Hyper-parameters	
Pre-train Critic	
optimiser	Adam
learning rate	0.0003
batch size	64
τ (target net speed)	0.005
α (entropy regularization)	0.001
buffer size	1000
length penalty	0.0001
Joint Training	
optimiser	Adam
learning rate	0.0004
batch size	64
τ (target net speed)	0.005
α (entropy regularization)	0.001
buffer size	1000
length penalty	0.0001
λ_{MLE}	0.1

Table 7: Hyper-parameters for SAC training.

A.2 Training

We use PyTorch (Paszke et al., 2019) (v1.4, CUDA 10.1) for our experiments. We early stop the actor training if validation loss does not improve for 10 epochs, we pretrain critics for 5 epochs for the **Multi30K** datasets and for 3 epochs for the larger **IWSLT 2014**. We early stop the SAC training if validation BLEU does not improve for 10 epochs. For all the setups, we also halve the learning rate if no improvement is obtained for two epochs. On a single NVIDIA RTX2080-Ti GPU, it takes around 5-6 hours up to 36 hours to train a model depending on the data size and the language pair. The number of learnable parameters is about 7.89M for smaller Multi30K models and about for 15.64M for the bigger IWSLT model. All models were re-trained 3 times to ensure reproducibility.

A.3 Soft Actor-Critic Training Algorithm

We describe the main steps of SAC training in Algorithm 1.

Algorithm 1: Soft Actor-Critic.

```
Initialise parameters:
Q function: $\theta$;
Policy: $\phi$;
Unsupervised Reward: $\delta$;
Replay Buffer: $\mathcal{D} \leftarrow \emptyset$;
for each iteration do
    for each translation step do
        $a_t \sim \pi_\phi(a_t, s_t)$;
        $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$;
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{s_t, a_t, r(s_t, a_t), s_{t+1}\}$ ;
    end
    for each gradient step do
        $\theta_i \leftarrow \theta_i - \lambda_Q \nabla_{\theta_i} L(\theta_i)$ for $i \in \{1, 2\}$;
        $\phi \leftarrow \phi - \lambda_\pi \nabla_\phi J(\phi)$;
        $\alpha \leftarrow \alpha - \lambda_\pi \nabla_\alpha J(\alpha)$;
        $\theta_i \leftarrow \tau\theta_i + (1 - \tau)\bar{\theta}_i$ for $i \in \{1, 2\}$;
        if unsupervised reward then
            $\delta \leftarrow \delta - \lambda_z \nabla_\delta r(\delta)$;
        end
    end
end
```

LM	**2016**	**2017**	**COCO**
Multi30K	44.07	79.95	77.7
IWSLT 2014	579.47	403.54	381.56

Table 8: Perplexity on **Multi30K** testsets for **Multi30K** and **IWSLT 2014** language models.

A.4 Domain Distance

To assess to what extent the test sets used in our experiments can be considered out-of-domain, we train (i) an English language model on **Multi30K** training set; and (ii) a German language model on

		2016			2017			COCO		
	model	BLEU	METEOR	TER	BLEU	METEOR	TER	BLEU	METEOR	TER
EN-FR	MLE	58.8	73.8	**26.7**	54.2	70.2	30.1	42.6	62.1	36.0
	`SAC BLEU`	**59.4**	**74.0**	**26.7***	**55.2**	**70.8**	**29.2**	**44.1**	**63.4**	**35.5**
	`SAC unsuper`	58.2	73.6	27.3	54.4*	70.6	29.8	43.5	63.2	35.7*
EN-DE	MLE	**37.5**	56.3	**42.1**	**33.8**	**53.1**	**47.6**	29.3	**49.3**	**50.9**
	`SAC BLEU`	36.6	56.2*	43.2	33.5	**53.1***	**47.6***	**29.6***	**49.3***	51.0*
	`SAC unsuper`	36.3	**56.5***	44.1	33.1	52.9*	48.7	28.3	48.6	51.5

Table 9: Results on the test sets for ambiguous words.

Freq. 1	source word	traveler
	gold target word	reisender
	source sentence	an oriental **traveler** awaits his turn at the currency exchange .
	reference sentence	ein orientalischer **reisender** ‘traveler’ wartet am wechselschalter bis er dran ist .
	MLE	ein orientalisch aussehender **behinderter** ‘disabled’ wartet darauf , dass die kurve sich die glastür aufhebt .
	`SAC BLEU`	ein orientalisch aussehender **techniker** ‘technician’ wartet auf die hecke seiner kurve .
	`SAC unsuper`	ein orientalisch aussehender **mann** ‘man’ wartet darauf , dass seine kurve auf den fehenk die kurve ist .
Freq. 28	source word	check
	gold target word	scheck
	source sentence	a woman is holding a large **check** for kids food basket .
	reference sentence	eine frau hält einen großen **scheck** ‘check’ für ” kids’ food basket ” .
	MLE	eine frau hält ein großes **überprüfen** ‘proof’ für kinder .
	`SAC BLEU`	eine frau hält einen großen **informationen** ‘information’ für kinder in den korb .
	`SAC unsuper`	eine frau hält ein großes **überprüfen** ‘proof’ für kinder , die einen korb zu verkaufen ist .

Table 10: Samples of translations for words of different frequency on **2016** EN-DE. In both cases more correct translations are provided by `SAC unsuper`. Bold highlights target words and their translations.

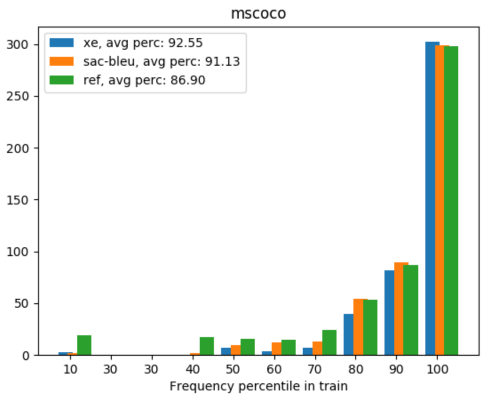


Figure 1: Training frequency for **COCO** words as translated by MLE and `SAC BLEU`. We also report reference frequencies.

the **IWSLT 2014** training set.[7] Table 8 shows language model perplexities on the Mutli30k test data. With respect to the **IWSLT 2014** model, **Multi30K** test sets are clearly very different from the training data. With respect to the Multi30K model, **2017** and **COCO** are more distant from the train partition than 2016 testset.

A.5 Analysis of distributions

We argue that the improvement over MLE can be partially attributed to a better handling of less frequent words. It has been shown that rare words tend to be under-represented in NMT (Koehn and Knowles, 2017; Shen et al., 2016). RL training with regularized entropy might mitigate this issue due to a better exploration of the action space. To illustrate this point, we compute the training frequency of the words generated by the NMT systems for the sentences where an improvement over MLE is observed. Figure 1 shows the training frequency percentiles for MLE and `SAC BLEU` English-French translations of the **COCO** testset. Reference frequencies are also provided for comparison. We observe that although both MLE and SAC contain more frequent words than the reference, this tendency is less pronounced for SAC. We relate this observation to the fact that our SAC outperforms MLE for the ambiguous word translation (Table 4) where the most frequent translation is not always the correct one.

---

[7] We train Transformer language models using the fairseq toolkit (Ott et al., 2019).

Us vs. Them: A Dataset of Populist Attitudes, News Bias and Emotions

Pere-Lluís Huguet Cabot[1,3], **David Abadi**[2], **Agneta Fischer**[2], **Ekaterina Shutova**[1]
[1] Institute for Logic, Language and Computation, University of Amsterdam
[2] Department of Psychology, University of Amsterdam
[3] Babelscape Srl, Sapienza University of Rome
`perelluis1993@gmail.com`
`{d.r.abadi, A.H.Fischer, e.shutova}@uva.nl`

Abstract

Computational modelling of political discourse tasks has become an increasingly important area of research in natural language processing. Populist rhetoric has risen across the political sphere in recent years; however, computational approaches to it have been scarce due to its complex nature. In this paper, we present the new *Us vs. Them* dataset, consisting of 6861 Reddit comments annotated for populist attitudes and the first large-scale computational models of this phenomenon. We investigate the relationship between populist mindsets and social groups, as well as a range of emotions typically associated with these. We set a baseline for two tasks related to populist attitudes and present a set of multi-task learning models that leverage and demonstrate the importance of emotion and group identification as auxiliary tasks.

1 Introduction

Political discourse is essential in shaping public opinion. The tasks related to modelling political rhetoric have thus been gaining interest in the natural language processing (NLP) community. Many of them focused on automatically placing a piece of text on the left-to-right political spectrum. For instance, much research has been devoted to detecting bias in news sources (Kiesel et al., 2019) and predicting the political affiliation of politicians (Iyyer et al., 2014) and social media users, more generally (Conover et al., 2011; Pennacchiotti and Popescu, 2011; Preoţiuc-Pietro et al., 2017). Other works conducted a more fine-grained analysis, identifying the framing of political issues in news articles (Card et al., 2015; Ji and Smith, 2017). Recently, the field has also turned attention towards modelling the spread of political information in social media, such as detecting fake news or political perspectives (Li and Goldwasser, 2019; Chandra et al., 2020; Nguyen et al., 2020).

Populism has taken the spotlight in political communication in recent years. Various countries around the globe have experienced a surge of populist rhetoric (Inglehart and Norris, 2016) in both the public and political space. Despite this, approaches to computational modelling of populist discourse have so far been scarce. Due to the flexible nature of populism, annotating populist rhetoric in text is challenging, and the existing research in this area has relied on small-scale analysis by experts (Hawkins et al., 2019). In this paper, we present a new dataset[1] of *Reddit* comments annotated for populist attitudes and the first large-scale computational models of this phenomenon. We rely on research in social- and behavioural sciences (e.g., political science and social psychology) to operationalise a definition of populism and an annotation procedure that allows us to capture and generalise the crucial aspects of populist rhetoric at scale.

In social sciences, populism is essentially described as a not fully developed political ideology and a series of background beliefs and techniques (Aslanidis, 2016), traditionally centred around the *Us vs. Them* dichotomy. In one of the first attempts to fully define populism (Mudde, 2004), it is described as a thin ideology around the distinction between 'the people', which includes the 'Us', and 'the elites' describing the 'Them', and with politics being a tool for 'the people' to achieve the common good or 'the popular will' (Kyle and Gultchin, 2018; Rodrik, 2019). Through different platforms, populism uses this rhetoric that revolves around social identity (Hogg, 2016; Abadi, 2017) and the *Us vs. Them* argumentation (Mudde, 2004). While right-wing populism tends to be characterised by *fear*, *resentment*, *anger* and *hatred*, left-wing populism is associated with *shame* and *guilt* (Otjes and

---

[1] Available at https://github.com/LittlePea13/UsVsThem

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1921–1945
April 19 - 23, 2021.

Louwerse, 2015; Salmela and von Scheve, 2017a). Moreover, emotions have been shown to be crucial in shaping public opinion more generally (Marcus, 2002, 2003; Demertzis, 2006; Rico et al., 2020).

The design of our annotation scheme and the dataset are inspired by this research, particularly the link between populist rhetoric and both social identity and emotions. Our dataset consists of comments posted on Reddit that explicitly mention a social group. We collect the comments posted in response to news articles across the political spectrum. Through crowd-sourcing, we annotate supportive, critical and discriminatory attitudes towards the group, as well as a range of emotions typically associated with populist attitudes. At the same time, given the relevance of news in the spread of such mindsets, we investigate the relationship between news bias and the *Us vs. Them* rhetoric. Our data analysis reveals interesting interactions between populist attitudes, specific social groups and emotions.

We also present a series of computational models, automatically identifying populist attitudes, based on *RoBERTa* (Liu et al., 2019). We experiment in a multi-task learning framework, jointly modelling *supportive* vs. *discriminatory* attitudes towards a group, the identity of the group and emotions towards the group. We demonstrate that joint modelling of these phenomena leads to significant improvements in detection of populist attitudes.

2 Related work

2.1 Psychology research on populism

Populist rhetoric revolves around social identity (Hogg, 2016; Abadi, 2017; Marchlewska et al., 2018; Bos et al., 2020) and the *Us vs. Them* argumentation (Mudde, 2004). Social identity explores the relations of individuals to social groups. Turner and Reynolds (2010) study the evolution of research into social identity and explain the *Us vs. Them* as an inter-group phenomenon, exposing its relation to social identity where the “self is hierarchically organised and that it is possible to shift from intra-group (‘we’) to inter-group (‘us’ versus ‘them’) and vice versa.”

Emotions constitute a part of the populist rhetoric and have been essential for information processing and the formation of (public) opinion among citizens (Marcus, 2002; Götz et al., 2005; Demertzis, 2006). While social identity and socio-economic factors have been considered primary indicators of populism’s growth (Rooduijn and Burgoon, 2018), emotional factors have lately become a focus within empirical studies, particularly regarding the reactions and spread of populist views (Hameleers et al., 2017). Specific appraisal patterns have characterised emotions, i.e. an adverse event for which one blames the other is felt as anger - a pattern of appraisals is referred to as *Core Relational Themes* (Smith and Lazarus, 1993; Lazarus, 2001), which are the central (therefore core) harm or benefit that underlies each of the negative and positive emotions (Smith and Lazarus, 1993; Moors et al., 2013). Latest attempts to scrutinise populism from the communication science and social psychological perspective have described populist communication and language (Abadi et al., 2016; Rico et al., 2017) and demonstrated its operationalisation through experimental research (Wirz et al., 2018) as being successful in inducing emotions (Bakker et al., 2020). According to the concept of media populism (Krämer, 2014; Mazzoleni and Bracciale, 2018), media effects can further evoke hostility toward the perceived ‘elites’ and (ethnic/religious) minorities, as it contributes to the construction of social identities, such as in-groups and out-groups (i.e., *Us vs. Them*).

2.2 Modelling political discourse in NLP

Handcrafted features such as word-frequency (Laver et al., 2003) were initially the base of NLP approaches to model political data. Thomas et al. (2006) introduced the *Convote* dataset of US congressional speeches, and applied an *support-vector machine* (SVM) classifier leveraging discourse information to identify policy stances in it. One of the first uses of neural networks on political text was the work of Iyyer et al. (2014), who used a *recurrent neural network* (RNN) to identify the party affiliation on the *Convote* dataset. Li and Goldwasser (2019) detected the political perspective of news articles using a *long short-term memory* (LSTM) and a *graph convolutional network* (GCN) on user data from Twitter. Other research investigated the *framing effect* in news articles, which is a mechanism that promotes a particular perspective (Entman, 1993). Card et al. (2015) presented the *Media Frame Corpus*, which explores policy framing within news articles. Ji and Smith (2017) developed a discourse-level Tree-RNN model to identify the framing in each article, by using this corpus dataset. Huguet Cabot et al. (2020) addressed this task by leveraging emotion and metaphor detection in an MTL setup. Other works have also explored sentence-level framing (Johnson et al., 2017; Hartmann et al., 2019).

Hate speech detection is not limited to the analysis of political discourses. However, it is related to exposing populist rhetoric in digital communication (Meret and Pajnik, 2017; Estelles and Castellví Mata, 2020). Several NLP approaches (Mishra et al., 2020), as well as recent shared tasks (Zampieri et al., 2019, 2020) have been proposed to tackle this widespread problem.

While political bias and framing have been widely explored, research on modelling populist rhetoric is still in its nascent stages. Previous work in this area focused on a general description of populism to determine whether a particular text, such as a party manifesto or a political speech contains what is understood as populist rhetoric or attitudes (Hawkins, 2009; Rooduijn and Pauwels, 2011; Manucci and Weber, 2017). Manual annotation was necessary to perform this analysis, often by experts, which also limited the scope and amount of data used, while the resulting datasets are not sufficiently large to train current machine learning models. Furthermore, the description of what constitutes populist rhetoric is still diffuse and covers many different aspects. Hawkins et al. (2019) used holistic grading to assess whether a text is populist or not, to later determine the degree of 'populism' of individual political leaders, thus creating the only existing dataset of populist rhetoric, the *Global Populist Database*.

3 Dataset creation

Data collection. By annotating Reddit comments that refer to a social group, we monitored how online discussions target them and whether the text showed a positive or negative attitude towards that social group, ranging from support to discrimination. While this process did not ensure capturing the complexity behind the *Us vs. Them* rhetoric, we detected comments directed at certain groups (out-groups) and the attitude towards them within an online community (in-group). We restricted this to six specific groups that populist rhetoric has targeted as an out-group, **Immigrants**, **Refugees**, **Muslims**, **Jews**, **Liberals** and **Conservatives**. Current research has shown these groups are common targets of populism in the US, UK and across Europe (Inglehart and Norris, 2016; Mudde and Kaltwasser, 2018). Note that to annotate sufficient comments per group we limit the current work to six groups, which is by no means a complete list of targeted groups. We encourage future research to broaden the scope of groups covered.

We chose to extract data from Reddit, (1) due to its availability through the *Pushshift* repository[2] (Baumgartner et al., 2020) and the *Google Bigquery* service, (2) its social identity dynamics (in-group vs. out-group) as close-nit communities created by sub-Reddits, (3) its nature as a social news aggregation platform, and (4) that it has been shown to encourage toxic communication between users and hate speech towards social groups (Massanari, 2017; Salminen et al., 2020; Munn, 2020). To filter the data for annotation, we followed several steps. (1) We identified submissions in Reddit which shared a news article from a news source listed at the *AllSides* website[3], (2) we extracted comments which are direct replies to the submission where both the news article title and the comment match any of the keywords for our groups. Keywords were devised using online resources from the *Anti Defamation League*[4] as well as by consulting social scientists. The full list of keywords can be found in Appendix A Table 4. (3) We selected comments with a minimum of 30 words and a maximum of 250 words, and sampled from specific periods during which each group was actively discussed on Reddit. See Appendix A Table 3 for details. (4) We removed comments that contained keywords from multiple social groups to make the annotation process more straightforward. (5) We randomly sampled 300 comments per group and news source bias according to *AllSides* (left, centre-left, centre, centre-right, right), resulting in a total of 9000 Reddit comments. Note that the bias is not directly related to individual comments, but rather to the news article the comment responded to.

Annotation procedure. To capture the *Us vs. Them* rhetoric, we asked: **What kind of language does this comment contain towards _*group*_?**, where _*group*_ corresponds to the specific social group that comment refers to. Respondents had four options: **Discriminatory**, **Critical**, **Neutral** or **Supportive**. An extended description and an example as presented to annotators can be found in Appendix A.2, and Figure 5. We asked anno-

---

[2] https://pushshift.io/

[3] https://www.allsides.com/unbiased-balanced-news

[4] https://www.adl.org/

tators a second question to capture the emotions expressed towards the group in the same comment. We extended Ekman's model of 6 *Basic Emotions* (Ekman, 1992) to a 12-emotions model, which includes a balanced set of positive and negative sentiments. Specifically, we included emotions previously shown to be associated with populist attitudes (Demertzis, 2006). We also provided the annotators with a brief description of each emotion, inspired by the concept of *Core Relational Themes* (Smith and Lazarus, 1990). The positive emotions are **Gratitude**, **Happiness**, **Hope**, **Pride**, **Relief** and **Sympathy**, and the negative emotions are **Anger**, **Fear**, **Contempt**, **Sadness**, **Disgust** and **Guilt**. Detailed descriptions can be found in Appendix A.2 along with an example 6. The annotation was conducted on *Amazon Mechanical Turk* (MTurk) and its framework can be accessed here.

Annotation reliability. Once the MTurk annotation was completed, we deployed the *CrowdTruth 2.0* toolkit (Dumitrache et al., 2018) to assess the quality of annotations and to identify unreliable workers. *CrowdTruth* includes a set of metrics to analyse and obtain probabilistic scores from crowdsourced annotations. *Worker Quality Score* (WQS) is a metric that measures each worker's performance, by leveraging their agreement with other annotators and the difficulty of annotation. The *Media Unit Annotation Score* (UAS) is given for each comment and possible answer, indicating the probability with which each option could be the gold label. Finally, *Media Unit Quality score* (UQS) describes the quality of annotation for each comment. We removed annotations by workers with a WQS lower than 0.1 and those with high disagreement after manually checking their responses, and recomputed the metrics. We removed comments left with only one annotator and comments with a UQS lower than 0.2. This resulted in 4278 comments with 5+ annotators, and 2564 comments with less, which constitute our final dataset.

Following the same procedure as Demszky et al. (2020), we computed inter-rater correlation (Delgado and Tibau Alberdi, 2019) by using Spearman correlation. We took the average of the correlation between each annotator's answers and all other annotators' average answers that labelled the same items. We obtain a range between 0.5 and 0.13 per emotion. The lowest agreement is for *Relief* (0.13), in line with Demszky et al. (2020), where the lowest value for correlation agreement being 0.16, and 0.17 for *Relief*. The full distribution can bee found in Appendix A.3 Figure 7.

Of course Dems are stealing the elections. They are playing by a different set of rules - being ruthless and violent. Dems take no prisoners and show no mercy to their enemies. The sooner GOP realizes it, the better. Because we have to up our game. If things go this way, we have only one way to save this country - Martial Law and kick every single liberal out!

Label	**UsVsThem**	**Group**	**Emotions**
Discriminatory	1	Liberals	Contempt, Disgust & Fear

You do realize it's sad to celebrate the US cutting the number of refugees down, right? These are people who come here seeking a safe haven from the violence or despair of their home countries, and we're turning them away.

Label	**UsVsThem**	**Group**	**Emotions**
Supportive	0	Refugees	Sympathy

Figure 1: Two samples of our *Us Vs. Them dataset.*

4 Data analysis

4.1 UsVsThem scale

For the *Us vs. Them* question, we aggregated the answers into a continuous scale. To obtain a score for each comment, we computed the *CrowdTruth* UAS for each of the labels assigned to it and then take a weighted sum. We assigned the weight of 0 to the *Supportive* label, $1/3$ to *Neutral*, $2/3$ to *Critical* and 1 to *Discriminatory*. The frequency distribution of comments on this scale can be seen in Appendix A.4 Figure 8. From now on, we will refer to it as the *UsVsThem* scale.

The scale is skewed, with an overall mean of 0.551 ± 0.265. Although our data selection was random across the selected news sources and groups, there are more comments with *negative* attitudes towards selected groups than positive or neutral ones due to its nature and our keyword selection.

We performed a two-way ANOVA (*Analysis of Variance*) test (Fujikoshi, 1993) on news bias and social groups as independent variables and the *UsVsThem* scale as the dependent variable to see whether the interactions between groups and news bias are significant. One-way tests show statistical significance. Interestingly, there was a statistically significant interaction between the effects of social groups and bias on the *UsVSThem* scale, $F(1, 20) = 12.33, p < 0.05$. Values can be found in Appendix A.4 Table 5. Therefore, we explored the interaction between them and the influence of

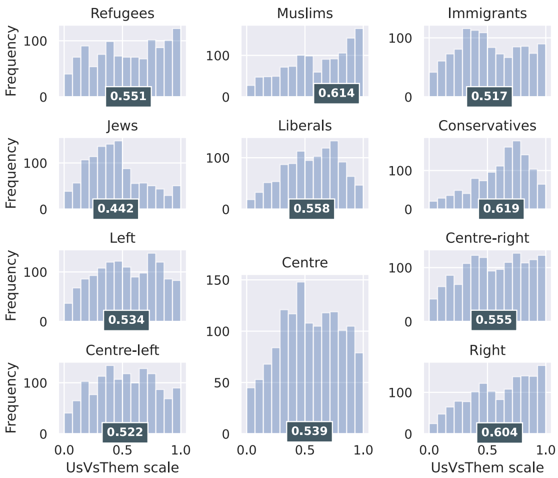


Figure 2: Comment frequency distribution on the *UsVsThem* scale per social group and news source bias. The mean for the scale is shown at the x axis.

news bias on how each group is perceived. We performed a *Tukey HSD test* to check for significance between means in the *UsVsThem* scale.

Social groups. There were differences between groups in terms of the *UsVsThem* scale when looking at the comment frequency distributions in Figure 2. For *Refugees*, the distribution was relatively flat as they received a similar amount of positive and negative attitude comments. *Immigrants* showed a similar distribution with fewer comments in the higher end, i.e. the group received less discrimination than *Refugees*. Despite the two share many inherent similarities, these differences may be explained by negative media coverage of *Refugees* portrayed as a threat and being attributed to negative attitudes. *Muslims* received a higher amount of discriminatory comments than any other group. On the other hand, *Conservatives* showed a similar mean, due to a very high amount of critical comments. *Liberals* also received a relatively high amount of critical comments. Both share moderately low tails, as they received less support and discrimination. Finally, *Jews* showed lower critical and discrimination values, with most values around *Neutral*, having the lowest mean value of all social groups. These variations translate into a significant ($p < 0.05$) difference between the means of each group, except for *Conservatives* and *Muslims*, and for *Liberals* and *Refugees*.

News source bias. In this case, the bias was not directly associated with the comment itself. However, differences in the distribution of comments and the out-group attitudes based on the original article's bias can be observed, as shown in Figure 2. Moreover, means increased from the *centre-left* to the *right* bias. Interestingly, there was no symmetry at the *centre* bias, contrary to the *Horseshoe Theory* (Hanel et al., 2019), which argues both ends of the political spectrum closely resemble one another. In terms of significant differences, all biases were significantly different from the *right* bias ($p < 0.05$), and there was a significant difference between *centre-left* and *centre-right* ($p < 0.05$). The remaining groups showed no significant difference.

Groups and news source bias. In line with the above-mentioned bias effect, there was almost always a significant difference between *right* and *centre-right* bias and the rest for each group. Only *right* bias showed a distinct high value and a negative attitude towards *Immigrants*, which even exceeds those towards *Refugees*. With the exception of the attitude towards *Conservatives*, *centre*, *centre-left* and *left* showed lower degrees of negative attitude towards any of the groups. Full results can be seen in Appendix A.4 Table 6 and Figure 10.

4.2 Emotions

Instead of using *CrowdTruth* for emotions, we considered an emotion as being present in the comment provided that at least 1/4 of annotators selected it. In case more than half of annotators marked that comment as *Neutral*, it was labelled as *Neutral*. This way, a comment can contain more than one emotion, except for *Neutral*. Unless specified otherwise, in this subsection *Neutral* refers to emotionally neutral.

In Figure 3 we present the correlations between the values for each emotion dimension and the *UsVsThem* scale across all comments, in the same fashion as Demszky et al. (2020). We show the hierarchical relations at the top, demonstrating which emotions interact more strongly with each other. The frequency of emotions in our dataset is as follows: *Anger* 1724, *Contempt* 2538, *Disgust* 1843, *Fear* 1136, *Gratitude* 70, *Guilt* 170, *Happiness* 59, *Hope* 307, *Pride* 174, *Relief* 37, *Sadness* 122, *Sympathy* 1139, *Neutral* 2094.

Emotions and the *UsVsThem* scale. We were interested in the interaction between emotions and social identity by exploring how the *UsVsThem* scale is shaped for each emotion. Not surprisingly, comments with negative emotions showed a higher value on the *UsVsThem* scale and a high correlation

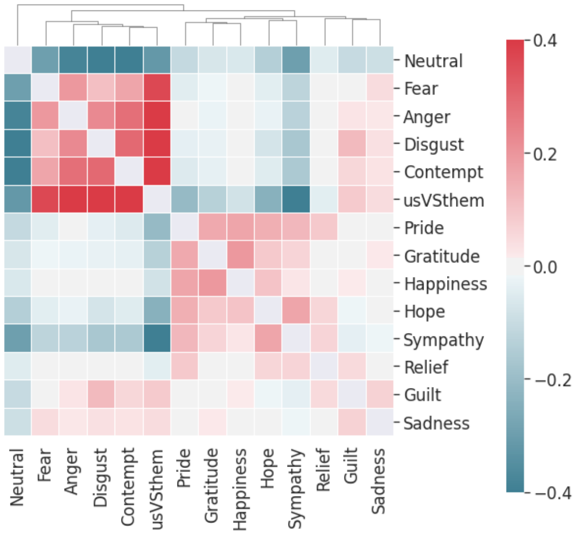


Figure 3: Correlation heat-map for different emotions.

with it, except for *Guilt* and *Sadness*. *Contempt* showed the strongest correlation, while *Anger* and *Fear* showed a higher proportion of *Discriminatory* comments. *Guilt* and *Sadness*, on the other hand, are characterised by a lower amount of comments in the *Discriminatory* range. In line with these results, the *UsVsThem* scale had a negative correlation with *Sympathy* and *Neutral* comments, and while *Sympathy* was the most frequent positive emotion, other emotions displayed a very similar relation to the *UsVsThem* scale. These results are visualised in Figure 3 including the *UsVsThem* scale and the distributions are summarised in Figure 9 in Appendix A.4.

Emotions and groups. We used a two-sided proportion z-test to check for significant differences since emotions are discrete variables. More than 25% of comments towards *Muslims* and *Refugees* showed *Fear*, with no significant difference between the two, followed by *Immigrants* at 21.5% and other groups at less than 10%. Another notable finding is that *Contempt* (47.7%) and *Disgust* (44.4%) were significantly higher towards *Conservatives*, particularly the latter, which for other groups never exceeded 30% of comments. *Sympathy* for *Liberals* (6.5%) and *Conservatives* (6.9%) was significantly lower when compared to other groups. *Hope* was present in a significantly higher number of comments for *Liberals* (7.9%). The values for all proportions can be found in Appendix Appendix A.4 Table 7.

Emotions and bias. Not many differences between biases were found. Most salient was the *right* bias showing a higher value in all negative emotions, significantly for *Anger* (31.5%), *Contempt* (43.4%) and *Fear* (21.9%). All proportion values can be found in Appendix A.4 Table 8.

5 Modelling populist rhetoric

5.1 Main tasks

Our models' main focus was to assess to which degree a social group is viewed as an out-group and whether in a negative or discriminatory manner. Our annotation procedure provided a scale from *Supportive* to *Discriminatory* for each comment. While this scale is artificial and highly dependent on our task's context, it provides a good indication of how strongly a social group is targeted in social media comments.

Regression *UsVsThem*. In our models, we explored two different main tasks. The first task was to predict the values on the *UsVsThem* scale in a regression model. This scale provides a score for each comment, which illustrates the attitude towards a social group mentioned in the comment ranging from *Supportive* (closer to 0) to *Discriminatory* (closer to 1). Values in between depict an intermediate attitude, *Neutral* lies at $1/3$, and *Critical* at $2/3$. By predicting the score, we modelled the out-group attitude of each comment. We used 33% of the data as the test set, and 13.4% as the validation set.

Classification *UsVsThem*. Our second task was to classify each comment in a binary fashion as whether the comment shows a negative attitude towards a group, i.e., *Critical* or *Discriminatory*, or not, i.e., *Neutral* or *Supportive*. This task resulted in a relatively balanced dataset, with 56% of *Critical* or *Discriminatory* comments. We used the same splits as before.

5.2 Auxiliary tasks

Emotion detection. Interactions between populist rhetoric and emotions have been explored in political psychology through surveys and behavioural experiments (Fischer and Roseman, 2007; Tausch et al., 2011; Salmela and von Scheve, 2017b; Redlawsk et al., 2018; Rollwage et al., 2019; Nguyen, 2019; Roseman et al., 2020). This is consistent with our findings in section 4 and further motivates modelling emotions in the context of populist rhetoric. For each comment, emotions were annotated as a Boolean vector. For our task, some emotions were rarely annotated or only present alongside more frequent ones. They increased the difficulty of the task while not providing relevant information. To simplify the auxiliary task, we considered the 8 most common emotions, *Anger, Contempt, Disgust, Fear, Hope, Pride, Sympathy* and *Neutral*.

Group identification. In the work of Burnap and Williams (2016), types of hate speech were differentiated based on race, religion, etc., and models were trained specifically on those categories. In ElSherief et al. (2018), data-driven analysis of online hate speech explored in profundity the differences between *directed* and *generalised* hate speech, and Silva et al. (2016) analysed the different targets of hate online. In our case, the *Us vs. Them* rhetoric metric showed significant differences for each group as we have seen in the previous section. Therefore, we hypothesised that the information bias (Caruana, 1993) the group identification task provides will help understand the *Us vs. Them* rhetoric aimed at the different social groups, which motivated its role as an auxiliary task.

5.3 Model architecture

We used the *Robustly Optimized BERT Pretraining Approach* (RoBERTa) (Liu et al., 2019) in its BASE variant as provided by Wolf et al. (2019).

Multi-task learning. In all setups, tasks shared the first eleven *transformer* layers of *RoBERTa*. The final 12th layer was task-specific, followed by a classification layer that used the hidden representation of the <s> token, to output a prediction. We used scheduled learning, where the losses of each task are weighted and changed during training. We also experimented with a three-task MTL model where the two auxiliary tasks are learned simultaneously.

We assigned three different loss weights associated with each task, λ_m for the main task, either regression or binary classification; λ_e for emotion detection; λ_g for group identification. For MTL with one auxiliary task, $\lambda_m + \lambda_e = \lambda_m + \lambda_g = 2$, while for the three-task MTL: $\lambda_m + \lambda_e + \lambda_g = 3$.

Regression *UsVsThem*. We used *Mean Squared Error* loss with a sigmoid activation function for the main task. For emotion identification as the auxiliary task, we used *Binary Cross-Entropy* loss, and for the group identification, we used *Cross-Entropy* loss, both with sigmoid activation. For all MTL models, there was a warm-up period of ω epochs, after which the weight is changed to $\lambda_g = 10^{-2}$ and $\lambda_e = 10^{-5}$, and $\lambda_e = \lambda_g = 10^{-5}$ for the three-task setting.

Classification *UsVsThem*. We used *Cross-Entropy* loss with a sigmoid activation function for the main task. The remaining tasks were kept the same as with the Regression case above. For all MTL models, there was a warm-up period of ω epochs, after which the weight was changed to $\lambda_g = 10^{-2}$ and $\lambda_e = 10^{-2}$, and $\lambda_e = \lambda_g = 10^{-5}$ for the three-task setting.

5.4 Experimental setup

Regression *UsVsThem*. We report model performance in terms of Pearson correlation coefficient (R). We found the optimal STL hyperparameters using the validation set: a learning rate of $3e-05$, a lineal warm-up period of 2 epochs and dropout of 0.15. The batch size used was 128. These hyperparameters were kept constant across our experiments for the regression *UsVsThem* task. For the emotion detection MTL setup, $\lambda_e = 0.15$ and $\omega = 8$. For the groups MTL, $\lambda_g = 0.15$ and $\omega = 5$. For the three-task MTL model we obtained optimal validation performance by setting $\omega = 8$ and both $\lambda_g = \lambda_e = 0.073$, which was the equivalent of $\lambda_g = \lambda_e = 0.05$ for the two-task MTL.

Classification *UsVsThem*. Similarly, we ran a grid-search to find the best hyperparameters for the classification setup. For the STL model, we obtained a learning rate of $5e-05$, a warm-up of 2 epochs and an extra dropout of 0.2. For emotions-MTL, $\lambda_e = 0.2$ and $\omega = 8$. For the groups-related MTL, $\lambda_g = 0.25$ and $\omega = 5$. For the three-task MTL, $\lambda_e = 0.95$, $\lambda_g = 0.25$, and $\omega = 8$. For both regression and classification, we report performance averaged over 10 different seeds.

5.5 Results

Results are presented in Table 1. We find that MTL outperforms STL in both versions of our task.

Regression *UsVsThem*. The STL baseline showed a 0.545 Pearson R to the gold score. When emotion identification was used as an auxiliary task, the performance increased by almost one point, to 0.553. The groups MTL setup showed a higher increase, up to 0.557. Both improvements

	STL	MTL, Emotion	MTL, Group	MTL, Emotion & Group
Pearson R	0.545 ± 0.005	$\mathbf{0.553 \pm 0.009}$	$\mathbf{0.557 \pm 0.012}$	$\underline{\mathbf{0.570 \pm 0.009}}$
Accuracy	0.705 ± 0.006	0.710 ± 0.009	0.711 ± 0.007	$\mathbf{0.717 \pm 0.004}$

Table 1: Results for the *Us vs. Them* rhetoric as regression and classification tasks. Significance compared to STL is bolded ($p < 0.05$). Significance compared to two-task MTL is underlined ($p < 0.05$). Average over 10 seeds.

were significant compared to the STL model, using the Williams test (Williams, 1959). Perhaps all the more interesting is that the three-task MTL model achieved the highest performance, even without its hyperparameters being specifically tuned as with the other setups. It resulted in a Pearson R of 0.570, i.e. over 2 points performance increase over STL, a statistically significant improvement over both STL and the remaining two MTL approaches.

Classification *UsVsThem*. Although not shown in the table, the accuracy baseline for a majority class classifier would be 0.550. All models highly surpassed that, with the real baseline set by the STL setup achieving a 0.705 accuracy. Results for the MTL approaches were similar to what we observed in the regression task. Emotion-MTL increased performance by half a point, to 0.710, as did group-MTL, with 0.711. The best performing model was again the three-task MTL, at 0.717, yielding a statistically significant improvement over STL, using the permutation test.

5.6 Analysis

Qualitative and error analysis. We selected comments with higher values on the scale where MTL improved the STL baseline predictions for the regression task. Comments with high emotion valence were better predicted by models that included emotion identification. Comments that had group-specific rhetoric with references to (derogatory) terms such as ‘illegal aliens’ were better predicted by models that incorporated group identification (see the first example in Table 2).

The standard deviation of the difference between the STL and the three-task predictions was just 0.055. This means that MTL helped capture nuanced information that improved prediction; however, comments with high squared error for STL still showed similar behaviour for MTL models. This aspect is shown in Appendix A.5 Figure 11. All models’ squared error showed a pair-wise Pearson correlation higher than 0.92. This observation prompted us to investigate comments with a high squared error. We identified three different sources. (1) **Comments with emotionally charged language**, slurs, or insults, which may often be associated with a more negative attitude towards a group, were mispredicted due to not being negative towards such group or being used ironically or satirically. (see second example in Table 2). (2) **Reference to multiple groups**: we removed comments that included keywords from similar groups, however it was impossible to account for all the terms that may refer to other groups. Hence, there are comments for which the prediction seems to be about a target different than the one at annotation time (see the third example in Table 2). More examples can be found in Appendix A.5 Table 9. (3) **Annotation error** is expected in any crowd-sourced annotation. While these were not as frequent as to pose a problem during training, they did occur as incorrect model predictions that can mistakenly decrease performance.

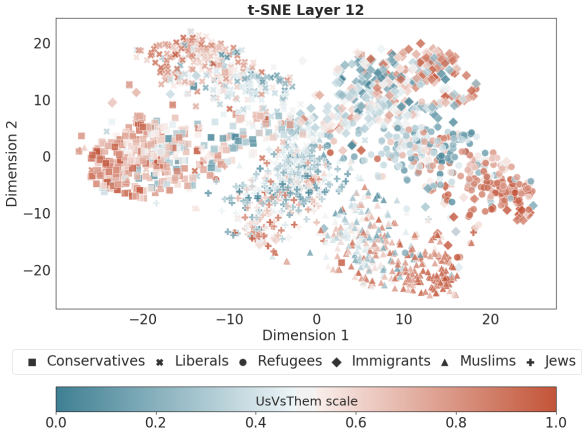


Figure 4: Three-task MTL main task specific layer.

Analysis of model representations. Our qualitative analysis showed that auxiliary tasks had a positive effect on predictions. Still, it cannot explain how the model changes its underlying structure and Reddit comments’ encoding. We explored how auxiliary tasks affected how the network encodes information through its layers to get a better understanding. We used *t-Distributed Stochastic Neighbour Embedding* (t-SNE) (van der Maaten and Hinton, 2008), a stochastic technique for dimension-

n	**Reddit Comment**
1	Does anyone else think it's absurd that we have Muslims serving in our congress? Literal foreign agents acting as if they hold the interests of the country and its people at heart? They never talk about the will of the people. It's always some bullshit about how white men (who founded and built this country) are evil and we need to let in more 3rd worlders who want to bomb and kill us. This is literal clown world tier nonsense.
2	You proud of yourselves, making 3 year olds represent themselves in immigration court? You fucking proud of that insanity? All for the sake of keeping out a gang that has already been in America for a long time, meanwhile regular home grown white kids are murdering dozens of their own classmates but goddam, at least they we're legal, amirite
3	Conservatives have every right to revolt. If we don't get our way we will destroy the country. I hope the left keeps pushing us to provoke a civil war. Or maybe Commiefornia should secede. Maybe that's the best thing that can happen, a complete break up. That way we can have our ethnostate, and the left can have their degenerate cesspool without us paying taxes for it. The US is dead anyway. It's time to burn this diverse shithole to the ground. It will be the ultimate proof that diversity doesn't work.

	Label	**MTL, E. & G.**	**MTL, Emo.**	**MTL, Groups**	**STL**	**Group**	**Emotions**
1	1.000	0.872	0.870	0.847	0.759	Muslims	Anger, Contempt, Disgust & Fear
2	0.02	0.774	0.874	0.740	0.834	Immigrants	Sympathy
3	0.071	0.729	0.773	0.747	0.8	Conservatives	Hope & Pride

Table 2: Examples of predictions for comments. Predictions are averages over 10 seeds for each model.

ality reduction focused on high dimensional data visualisation. We used it to visualise the hidden representations of the test set comments in-between *transformer* layers across the network. We present the results for both STL and the three-task MTL in Appendix A.5 Figures 12 and 13, where for both the first layers showed some structure not related to the tasks at hand. As we were using pre-trained weights from *RoBERTa*, this could be explained by the first layers modelling lower-level language characteristics as shown empirically in Tenney et al. (2019), where probing mechanisms indicate early layers being more relevant for tasks such as *POS tagging*. For STL, the last layers showed the *UsVsThem* scale continuously in the y axis. Once we introduced the auxiliary tasks of group identification and emotion classification differences in the last layers were exacerbated. For three-task MTL the last layers showed clusters for each social group, and related groups were closer together, such as *Refugees* and *Immigrants*, or *Liberals* and *Conservatives*. We also observe a radial distribution with highly emotional comments further away from the centre. Comments with very distant values on the scale (*Discriminatory* and *Supportive*) were closer together than with those in the mid-range (*Neutral* and *Critical*) as seen in Figure 4 and Appendix A.5 Figure 15. While paradoxical, our interpretation is that the model leverages the valence of emotion, where *Discriminatory* and *Supportive* comments are more loaded with emotion. This leads to a better performance of MTL compared to STL. This idea is supported by the distribution of emotions on the last layer, where emotionally neutral comments are closer to the centre of the plot, while more emotionally charged comments radially increase, visualised in Appendix A.5 Figure 14. In Appendix A.5 Figures 16 and 17 we present the emotion distribution for the group and emotion-specific layers, respectively.

6 Conclusions

We presented a new, large-scale dataset of populist rhetoric and the first series of computational models on this phenomenon. We have shown that joint modelling of emotion and populist attitudes towards social groups enhances performance over the single-task model, further corroborating previous research findings in various social sciences. Future work may deploy social information (e.g., Twitter) or explore the interactions of populist attitudes and the political bias of news articles as provided in our *Us Vs. Them* dataset.

Funding statement. This research was funded by the European Union's H2020 project *Democratic Efficacy and the Varieties of Populism in Europe* (DEMOS) under H2020-EU.3.6.1.1. and H2020-EU.3.6.1.2. (grant agreement ID: 822590) and supported by the European Union's H2020 Marie Skłodowska-Curie project *Knowledge Graphs at Scale* (KnowGraphs) under H2020-EU.1.3.1. (grant agreement ID: 860801).

References

David Abadi. 2017. *Negotiating Group Identities in Multicultural Germany: The Role of Mainstream Media, Discourse Relations, and Political Alliances*. Communication, Globalization, and Cultural Identity. Lexington Books.

David Abadi, Leen d'Haenens, Keith Roe, and Joyce Koeman. 2016. Leitkultur and discourse hegemonies: German mainstream media coverage on the integration debate between 2009 and 2014. *International Communication Gazette*, 78(6):557–584.

Paris Aslanidis. 2016. Is populism an ideology? a refutation and a new perspective. *Political Studies*, 64(1_suppl):88–104.

Bert N. Bakker, Gijs Schumacher, and Matthijs Rooduijn. 2020. Hot politics? affective responses to political rhetoric. *American Political Science Review*, 115(1):150–164.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset.

Linda Bos, Christian Schemer, Nicoleta Corbu, Michael Hameleers, Ioannis Andreadis, Anne Schulz, Desirée Schmuck, Carsten Reinemann, and Nayla Fawzi. 2020. The effects of populism as a social identity frame on persuasion and mobilisation: Evidence from a 15-country experiment. *European Journal of Political Research*, 59(1):3–24.

Peter Burnap and Matthew Leighton Williams. 2016. Us and them: identifying cyber hate on twitter across multiple protected characteristics. *EPJ Data Science*, 5(1):11.

Dallas Card, Amber E. Boydstun, Justin H. Gross, Philip Resnik, and Noah A. Smith. 2015. The media frames corpus: Annotations of frames across issues. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 438–444, Beijing, China. Association for Computational Linguistics.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Shantanu Chandra, Pushkar Mishra, Helen Yannakoudakis, Madhav Nimishakavi, Marzieh Saeidi, and Ekaterina Shutova. 2020. Graph-based modeling of online communities for fake news detection.

Michael D. Conover, Bruno Goncalves, Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. 2011. Predicting the political alignment of twitter users. In *2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third Int'l Conference on Social Computing*, pages 192–199.

Alan S. Cowen, Petri Laukka, Hillary Anger Elfenbein, Runjing Liu, and Dacher Keltner. 2019. The primacy of categories in the recognition of 12 emotions in speech prosody across two cultures. *Nature Human Behaviour*, 3(4):369–382.

Rosario Delgado and Xavier-Andoni Tibau Alberdi. 2019. Why cohen's kappa should be avoided as performance measure in classification. *PLoS ONE*, 14:1–26.

Nicolas Demertzis. 2006. Emotions and populism. In *Emotion, Politics and Society*.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online. Association for Computational Linguistics.

Anca Dumitrache, Oana Inel, Lora Aroyo, Benjamin Timmermans, and Chris Welty. 2018. Crowdtruth 2.0: Quality metrics for crowdsourcing with disagreement.

Paul Ekman. 1992. An Argument for Basic Emotions. *Cognition and Emotion*.

Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, and Elizabeth Belding. 2018. Hate lingo: A target-based linguistic analysis of hate speech in social media. In *Twelfth International AAAI Conference on Web and Social Media*.

Robert M. Entman. 1993. Framing: Toward clarification of a fractured paradigm. *Journal of Communication*, 43(4):51–58.

Marta Estelles and Jordi Castellví Mata. 2020. The educational implications of populism, emotions and digital hate speech: A dialogue with scholars from canada, chile, spain, the uk, and the us. *Sustainability*, 12.

Agneta H. Fischer and Ira J. Roseman. 2007. Beat them or ban them: the characteristics and social functions of anger and contempt. *Journal of personality and social psychology*, 93 1:103–15.

Yasunori Fujikoshi. 1993. Two-way anova models with unbalanced data. *Discrete Mathematics*, 116(1):315 – 334.

Thomas Götz, Anne Frenzel, Reinhard Pekrun, and Nathan Hall. 2005. *Emotional intelligence in the context of learning and achievement*, pages 233–253. Cambridge, MA: Hogrefe & Huber Publishers.

Michael Hameleers, Linda Bos, and Claes H. de Vreese. 2017. "they did it": The effects of emotionalized blame attribution in populist communication. *Communication Research*, 44(6):870–900.

Paul H. P. Hanel, Natalia Zarzeczna, and Geoffrey Haddock. 2019. Sharing the same political ideology yet endorsing different values: Left- and right-wing political supporters are more heterogeneous than moderates. *Social Psychological and Personality Science*, 10(7):874–882.

Mareike Hartmann, Tallulah Jansen, Isabelle Augenstein, and Anders Søgaard. 2019. Issue framing in online discussion fora. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1401–1407, Minneapolis, Minnesota. Association for Computational Linguistics.

Kirk A Hawkins. 2009. Is Chávez Populist?: Measuring Populist Discourse in Comparative Perspective. *Comparative Political Studies*, 42(8):1040–1067.

Kirk A. Hawkins, Rosario Aguilar, Bruno Castanho Silva, Erin K. Jenne, Bojana Kocijan, and Rovira Kaltwasser. 2019. Measuring Populist Discourse: The Global Populism Database. In *EPSA Annual Conference in Belfast*.

Michael A. Hogg. 2016. *Social Identity Theory*, pages 3–17. Springer International Publishing, Cham.

Pere-Lluís Huguet Cabot, Verna Dankers, David Abadi, Agneta Fischer, and Ekaterina Shutova. 2020. The Pragmatics behind Politics: Modelling Metaphor, Framing and Emotion in Political Discourse. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4479–4488, Online. Association for Computational Linguistics.

Ronald F. Inglehart and Pippa Norris. 2016. Trump, brexit, and the rise of populism: Economic have-nots and cultural backlash. *Social Science Research Network*.

Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political ideology detection using recursive neural networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1113–1122, Baltimore, Maryland. Association for Computational Linguistics.

Yangfeng Ji and Noah A. Smith. 2017. Neural discourse structure for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 996–1005, Vancouver, Canada. Association for Computational Linguistics.

Kristen Johnson, Di Jin, and Dan Goldwasser. 2017. Leveraging behavioral and social information for weakly supervised collective classification of political discourse on twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 741–752, Vancouver, Canada. Association for Computational Linguistics.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Benjamin Krämer. 2014. Media populism: A conceptual clarification and some theses on its effects. *Communication Theory*, 24(1):42–60.

Jordan Kyle and Limor Gultchin. 2018. Populism in power around the world. *SSRN Electronic Journal*.

Michael Laver, Kenneth Benoit, and John Garry. 2003. Extracting policy positions from political texts using words as data. *American Political Science Review*, 97(2):311–331.

Richard Lazarus. 2001. Relational meaning and discrete emotions. *Appraisal Processes in Emotion: Theory, Methods, Research*, pages 37–67.

Chang Li and Dan Goldwasser. 2019. Encoding social information with graph convolutional networks forPolitical perspective detection in news media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2594–2604, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Luca Manucci and Edward Weber. 2017. Why The Big Picture Matters: Political and Media Populism in Western Europe since the 1970s. *Swiss Political Science Review*.

Marta Marchlewska, Aleksandra Cichocka, Orestis Panayiotou, Kevin Castellanos, and Jude Batayneh. 2018. Populism as identity politics: Perceived in-group disadvantage, collective narcissism, and support for populism. *Social Psychological and Personality Science*, 9(2):151–162.

George Marcus. 2002. The sentimental citizen: emotion in democratic politics. *Choice Reviews Online*, 40(05):40–3068–40–3068.

George Marcus. 2003. *The Psychology of Emotion*, pages 182–221. Oxford University Press.

Adrienne Massanari. 2017. #gamergate and the fappening: How reddit's algorithm, governance, and culture support toxic technocultures. *New Media & Society*, 19(3):329–346.

Gianpietro Mazzoleni and Roberta Bracciale. 2018. Socially mediated populism: the communicative strategies of political leaders on Facebook. *Palgrave Communications*, 4(1):50.

Susi Meret and Mojca Pajnik. 2017. *Populist political communication in mediatized society*. Routledge, United Kingdom.

Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020. Tackling online abuse: A survey of automated abuse detection methods.

Agnes Moors, Phoebe C. Ellsworth, Klaus R. Scherer, and Nico H. Frijda. 2013. Appraisal theories of emotion: State of the art and future development. *Emotion Review*, 5(2):119–124.

Cas Mudde. 2004. The Populist Zeitgeist. *Government and Opposition*.

Cas Mudde and Cristóbal Rovira Kaltwasser. 2018. Studying populism in comparative perspective: Reflections on the contemporary and future research agenda. *Comparative Political Studies*, 51(13):1667–1693.

Luke Munn. 2020. Angry by design: toxic communication and technical architectures. *Humanities and Social Sciences Communications*, 7.

Christoph G Nguyen. 2019. Emotions and populist support.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. Fang: Leveraging social context for fake news detection using graph representation. *ArXiv*, abs/2008.07939.

Simon Otjes and Tom Louwerse. 2015. Populists in parliament: Comparing left-wing and right-wing populism in the netherlands. *Political Studies*, 63(1):60–79.

Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, republicans and starbucks afficionados: user classification in twitter. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 430–438.

Daniel Preoţiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. 2017. Beyond binary labels: Political ideology prediction of twitter users. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 729–740, Vancouver, Canada. Association for Computational Linguistics.

David P. Redlawsk, Ira J. Roseman, Kyle Mattes, and Steven Katz. 2018. Donald trump, contempt, and the 2016 gop iowa caucuses. *Journal of Elections, Public Opinion and Parties*, 28(2):173–189.

Guillem Rico, Marc Guinjoan, and Eva Anduiza. 2017. The Emotional Underpinnings of Populism: How Anger and Fear Affect Populist Attitudes. *Swiss Political Science Review*, 23(4):444–461.

Guillem Rico, Marc Guinjoan, and Eva Anduiza. 2020. Empowered and enraged: Political efficacy, anger and support for populism in europe. *European Journal of Political Research*, 59(4):797–816.

Dani Rodrik. 2019. What's driving populism?

Max Rollwage, Leor Zmigrod, Lee de Wit, Raymond J. Dolan, and Stephen M. Fleming. 2019. What underlies political polarization? a manifesto for computational political psychology. *Trends in Cognitive Sciences*, 23(10):820 – 822.

Matthijs Rooduijn and Brian Burgoon. 2018. The Paradox of Well-being: Do Unfavorable Socioeconomic and Sociocultural Contexts Deepen or Dampen Radical Left and Right Voting Among the Less Well-Off? *Comparative Political Studies*, 51(13):1720–1753.

Matthijs Rooduijn and Teun Pauwels. 2011. Measuring populism: Comparing two methods of content analysis. *West European Politics*.

Ira J. Roseman, Kyle Mattes, David P. Redlawsk, and Steven Katz. 2020. Reprehensible, laughable: The role of contempt in negative campaigning. *American Politics Research*, 48(1):44–77.

Mikko Salmela and Christian von Scheve. 2017a. Emotional roots of right-wing political populism. *Science Information*, 56(4).

Mikko Salmela and Christian von Scheve. 2017b. Emotional roots of right-wing political populism. *Social Science Information*, 56(4):567–595.

Joni Salminen, Sercan Sengün, Juan Corporan, Soongyo Jung, and Bernard J. Jansen. 2020. Topic-driven toxicity: Exploring the relationship between online toxicity and news topics. *PLOS ONE*, 15(2):1–24.

Leandro Silva, Mainack Mondal, Denzil Correa, Fabrício Benevenuto, and Ingmar Weber. 2016. Analyzing the targets of hate in online social media.

Craig Smith and Richard Lazarus. 1990. *Emotion and Adaptation*, volume 21, pages 609–637.

Craig Smith and Richard Lazarus. 1993. Appraisal Components, Core Relational Themes, and the Emotions. *Cognition & Emotion - COGNITION EMOTION*, 7:233–269.

Nicole Tausch, Julia Becker, Russell Spears, Oliver Christ, Rim Saab, Purnima Singh, and Roomana Siddiqui. 2011. Explaining radical group behavior: Developing emotion and efficacy routes to normative and nonnormative collective action. *Journal of personality and social psychology*, 101:129–48.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. *CoRR*, abs/cs/0607062.

John C Turner and Katherine J Reynolds. 2010. The story of social identity. In *Rediscovering Social Identity: Key Readings*, pages 13–32. Psychology Press, Taylor & Francis, New York.

E. J. Williams. 1959. The comparison of regression variables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 21(2):396–399.

Dominique S Wirz, Martin Wettstein, Anne Schulz, Philipp Müller, Christian Schemer, Nicole Ernst, Frank Esser, and Werner Wirth. 2018. The Effects of Right-Wing Populist Communication on Emotions and Cognitions toward Immigrants. *The International Journal of Press/Politics*, 23(4):496–516.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online). International Committee for Computational Linguistics.

A Supplemental material

A.1 Data collection

	Time ranges	Events
Conservatives	2016/09/15 - 2016/12/15 2018/09/15 - 2018/12/15	Election periods
Liberals	2016/09/15 - 2016/12/15 2018/09/15 - 2018/12/15	Election periods
Muslims	2016/11/01 - 2017/11/30 2018/04/01 - 2018/05/01 2019/03/01 - 2019/06/01	Trump Muslim ban, Mosque attacks.
Immigrants	2016/11/01 - 2017/11/30 2017/01/15 - 2017/03/15 2018/06/17 - 2018/07/01 2018/10/01 - 2019/02/01	Migrant caravans, Children at the US border
Jews	2018/10/20 - 2018/11/25	Christchurch shooting

Table 3: Events and periods used for each group. If comments were not sufficient, they were sampled randomly from other time ranges. Refugees did not have enough overall comments to be filtered by time range.

	News Title	Comment
Refugees	refugee, asylum seeker	refugee, asylum seeker, undocumented, colonization
Immigration	-migra-, undocumented, colonization	-migra-, undocumented, colonization
Muslims	muslim, arab, muhammad, muhammed, islam, hijab, sharia	muslim, arab, muhammad, muhammed, islam, hijab, sharia
Jews	-jew(i/s)-, heeb- , sikey-, -zionis-, -semit-	-jew(i/s)-, heeb- , sikey-, -zionis-, -semit-
Liberals	antifa, libtard, communist, socialist, leftist, liberal, democrat	antifa, libtard, communist, socialist, leftist, liberal, democrat
Conservatives	altright, alt-right, cuckservative, trumpster, conservative, republican	altright, alt-right, cuckservative, trumpster, conservative, republican

Table 4: Keywords used in our data filtering process. The use of more emotionally laden terms is justified by their low occurrence compared to more common terms just to ensure a more diverse dataset.

A.2 Description of the annotation options

Discriminatory or Alienating. Annotators were asked to mark this in case the comment was either, (A) alienating or portraying a social group as negative, (B) a threat, danger or peril to society, (C) trying to ridicule it and attack that group as lesser or worthless.

Critical but not Discriminatory. In case the comment was critical, but not to the extent of the first option, annotators were asked to mark this option.

Supportive or Favorable. This answer refers to comments expressing support towards that group, by defending it or praising it.

Neutral. This option was offered in case none of the above applied, either because the group was only mentioned but the comment was not addressed at them, or there was no opinion whatsoever expressed towards the group, such as expressing purely factual information.

Annotators were first asked to select whether the comment showed a 'Positive', 'Negative' or 'Neutral' sentiment towards the specified group. With this approach, we intended to simplify the task and guide annotators, which then were offered to choose from 6 positive or 6 negative emotions according to sentiment they initially chose. In case annotators selected Neutral no further options were provided. The descriptions for each emotion were:

Positive emotions: **Gratitude** *Someone is doing/causing something good or lovely.*
Happiness/Joy[5] *Something good is happening. Something amusing or funny is happening.*
Hope *Something good/better might happen (sooner or later).*
Pride *Someone is taking credit for a good achievement.*
Relief *Something bad has changed for the better.*
Sympathy *Someone shows support or devotion.*

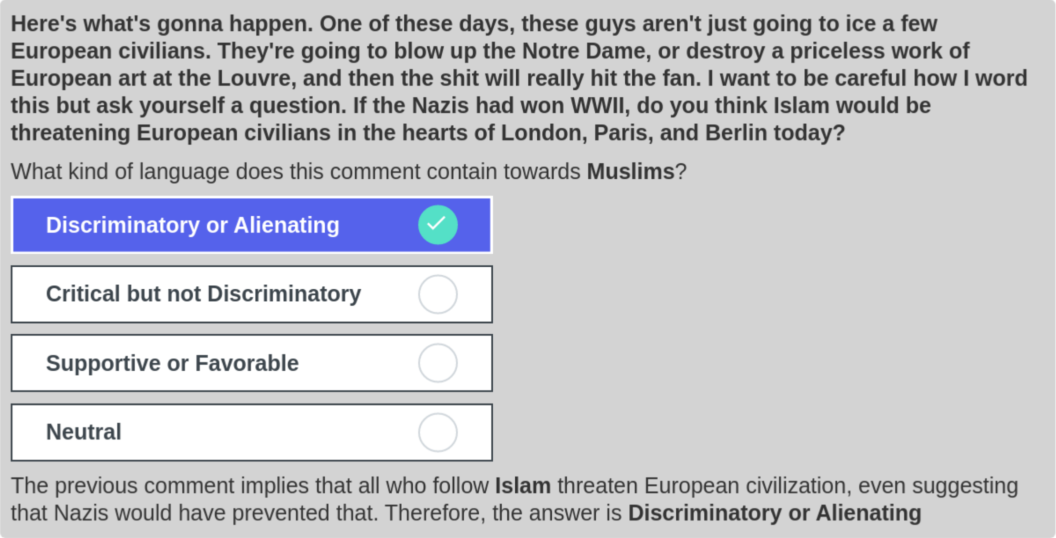
Negative emotions: **Anger.** *Someone is causing harm or a negative/undeserved outcome, while this could have been avoided.*
Someone is acting in an unjustified manner towards people.
Someone is blocking the goals of people.
Anxiety/Fear *Something negative might/could happen (sooner or later), which threatens the well-being of people.*
Contempt *Someone is inferior (for example, immoral, lazy or greedy).*
Someone is incompetent (for example, weak or stupid).
Sadness *Something bad or sad has happened.*
Someone has experienced a loss (for example, death or loss of possessions).
Moral Disgust[6] *Someone behaves in an offensive way (for example, corrupt, dishonest, ruthless, or unscrupulous behavior).*
Guilt/Shame[7] *Someone sees him-/herself as responsible for causing a harmful/ immoral/ shameful/ embarrassing outcome to people.*

---

[5] Referred to as Happiness for simplicity

[6] Referred to as Disgust for simplicity

[7] Referred to as Guilt for simplicity



Here's what's gonna happen. One of these days, these guys aren't just going to ice a few European civilians. They're going to blow up the Notre Dame, or destroy a priceless work of European art at the Louvre, and then the shit will really hit the fan. I want to be careful how I word this but ask yourself a question. If the Nazis had won WWII, do you think Islam would be threatening European civilians in the hearts of London, Paris, and Berlin today?

What kind of language does this comment contain towards **Muslims**?

- **Discriminatory or Alienating** ●
- **Critical but not Discriminatory** ○
- **Supportive or Favorable** ○
- **Neutral** ○

The previous comment implies that all who follow **Islam** threaten European civilization, even suggesting that Nazis would have prevented that. Therefore, the answer is **Discriminatory or Alienating**

Figure 5: Example of the first question as reference presented to MTurk annotators.

Which **primary** emotion does the text show towards **Muslims**?

- **Positive** ○
- **Negative** ●

Please select at least one type of type of **Negative** emotion the text primarily contains towards **Muslims**. You may select up to two emotions if the text contains them.

- **Anger** ○
  Someone is causing harm or a negative/undeserved outcome, while this could have been avoided.
  Someone is acting in an unjustified manner towards people.
  Someone is blocking the goals of people.
- **Anxiety/Fear** ●
  Something negative might/could happen (sooner or later), which threatens the well-being of people.
- **Contempt** ○
  Someone is inferior (for example, immoral, lazy or greedy).
  Someone is incompetent (for example, weak or stupid).
- **Moral Disgust** ○
  Someone behaves in an offensive way (for example, corrupt, dishonest, ruthless, or unscrupulous behavior).
- **Guilt/Shame** ○
  Someone sees him-/herself as responsible for causing a harmful/immoral/shameful/embarrassing outcome to people.
- **Sadness** ○
  Something bad or sad has happened.
  Someone has experienced a loss (for example, death or loss of possessions).

- **Neutral** ○

The exemplary comment describes **Muslims** and **Islam** as a threat that will "blow out the Notre Dame or destroy...", therefore it sees it as dangerous and posing a threat.

Figure 6: Example of the second question as reference presented to MTurk annotators.

A.3 Reliability

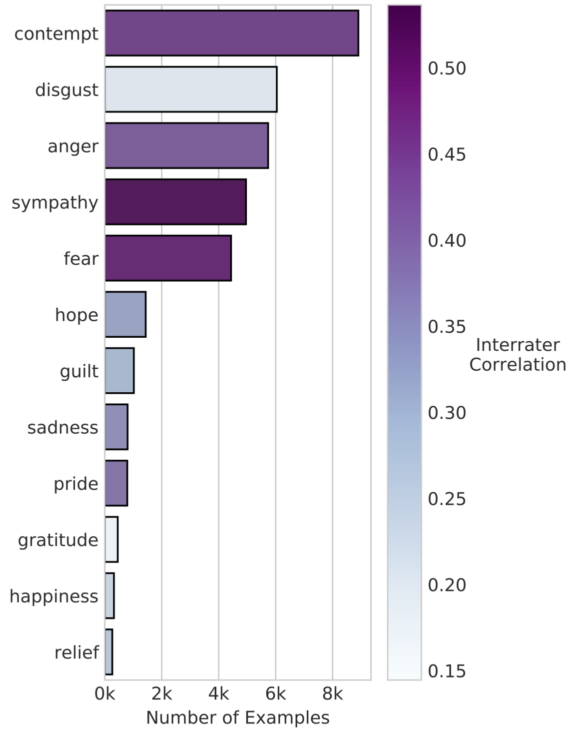


Figure 7: Number of annotations per emotions and the inter-rater correlation.

We also applied the algorithm called *Leave-One-Rater-Out* PPCA Cowen et al. (2019), using Bonferroni correction on p-values. *Principal Preserved Component Analysis* (PPCA) finds principal components which instead of preserving variance within a single dataset as conducted in PCA, preserve the cross-covariance between two different datasets, in our case being a comparison between annotations by one rater and a random set of other raters. In this manner, we can assess the degree of agreement and whether all component dimensions are significant, indicating significant emotion dimensions to be preserved. In our setup, the largest p-value for a dimension was $1.2e-03$, with all other dimensions showing much smaller values. This supports the idea that our emotion dimensions are significant, in order to be kept.

A.4 Data analysis

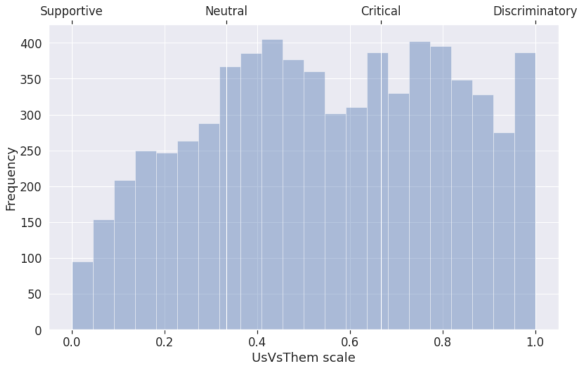


Figure 8: Distribution for the *UsVsThem* scale. Values closer to 0 are more supportive towards the target group, while higher values indicate a higher degree of criticism or eventually discrimination.

Predictor	Sum of Squares	*df*	Mean Square	*F*	*p*	partial η^2
(Intercept)	2582.47	1	2582.47	46×10^3	0.000	
Groups	22.05	5	4.41	78.73	0.000	0.04
Bias	4.82	4	1.21	21.52	0.000	0.01
Groups x Bias	13.82	20	0.69	12.33	0.000	0.03
Error	492.63	8794	0.06			

Table 5: Two-way ANOVA test.

	Conservatives	**Liberals**	**Immigrants**	**Refugees**	**Jews**	**Muslims**
left	$0.668^{rc,r}$	$0.540^{rc,r}$	0.471^{r}	0.524^{r}	0.433	$0.554^{rc,r}$
centre-left	$0.669^{rc,r}$	$0.513^{rc,r}$	0.447^{r}	0.507^{r}	0.422	$0.556^{rc,r}$
centre	$0.646^{rc,r}$	$0.487^{rc,r}$	0.516^{r}	0.541^{r}	0.452	$0.573^{rc,r}$
centre-right	$0.555^{c,lc,l}$	$0.602^{c,lc,l}$	0.497^{r}	0.557	0.433	$0.682^{c,lc,l}$
right	$0.543^{c,lc,l}$	$0.638^{c,lc,l}$	$0.646^{rc,c,lc,l}$	$0.625^{c,lc,l}$	0.467	$0.696^{c,lc,l}$

Table 6: Mean *UsVsThem* Regression scale for each group and bias. Statistical significance is shown as super-indexes, in case the mean is statistically different with other biases for that group. ^{l} left, ^{lc} centre-left, ^{c} centre, ^{rc} centre-right, ^{r} right. Tested using Tukey HSD test.

	Anger	**Contempt**	**Disgust**	**Fear**	**Gratitude**	**Guilt**	**Happiness**	**Hope**	**Pride**	**Relief**	**Sadness**	**Sympathy**	**Neutral**
Conservatives	28.5%	47.7%	44.4%	8.9%	1.3%	4.5%	1.4%	4.2%	4.1%	0.5%	2.4%	6.9%	23.7%
Liberals	22.4%	40.7%	28.3%	8.4%	1.3%	3.1%	0.8%	7.9%	2.9%	0.4%	1.6%	6.4%	32.6%
Jews	17.4%	22.8%	16.9%	8.2%	0.5%	1.3%	0.6%	2.3%	3.1%	0.6%	1.5%	23.8%	44.4%
Muslims	31.4%	39.2%	30.0%	26.1%	0.8%	2.5%	0.9%	2.0%	1.9%	0.3%	1.5%	15.1%	26.8%
Immigrants	23.5%	33.4%	18.8%	21.5%	1.5%	1.4%	0.8%	4.3%	2.2%	0.6%	1.6%	23.4%	30.9%
Refugees	26.9%	37.3%	21.8%	25.8%	0.7%	2.0%	0.6%	6.2%	1.0%	0.8%	2.0%	24.1%	25.3%

Table 7: Percentages of comments within each social group per emotion.

	Anger	**Contempt**	**Disgust**	**Fear**	**Gratitude**	**Guilt**	**Happiness**	**Hope**	**Pride**	**Relief**	**Sadness**	**Sympathy**	**Neutral**
left	22.5%	35.8%	25.8%	15.1%	0.7%	2.4%	0.6%	4.9%	2.5%	0.5%	2.0%	18.6%	31.5%
centre-left	21.1%	34.1%	27.0%	13.5%	1.2%	2.6%	0.7%	5.0%	2.1%	0.4%	1.8%	18.1%	32.6%
centre	24.4%	35.2%	26.1%	15.0%	1.5%	2.1%	1.1%	5.0%	2.7%	0.7%	1.8%	15.1%	32.8%
centre-right	25.9%	36.3%	26.5%	17.0%	1.2%	2.4%	0.9%	3.9%	3.2%	0.7%	1.2%	17.8%	29.4%
right	31.5%	43.4%	29.0%	21.9%	0.6%	2.9%	0.9%	3.6%	2.2%	0.4%	2.1%	13.5%	26.4%

Table 8: Percentages of comments within bias in the news source per emotion.

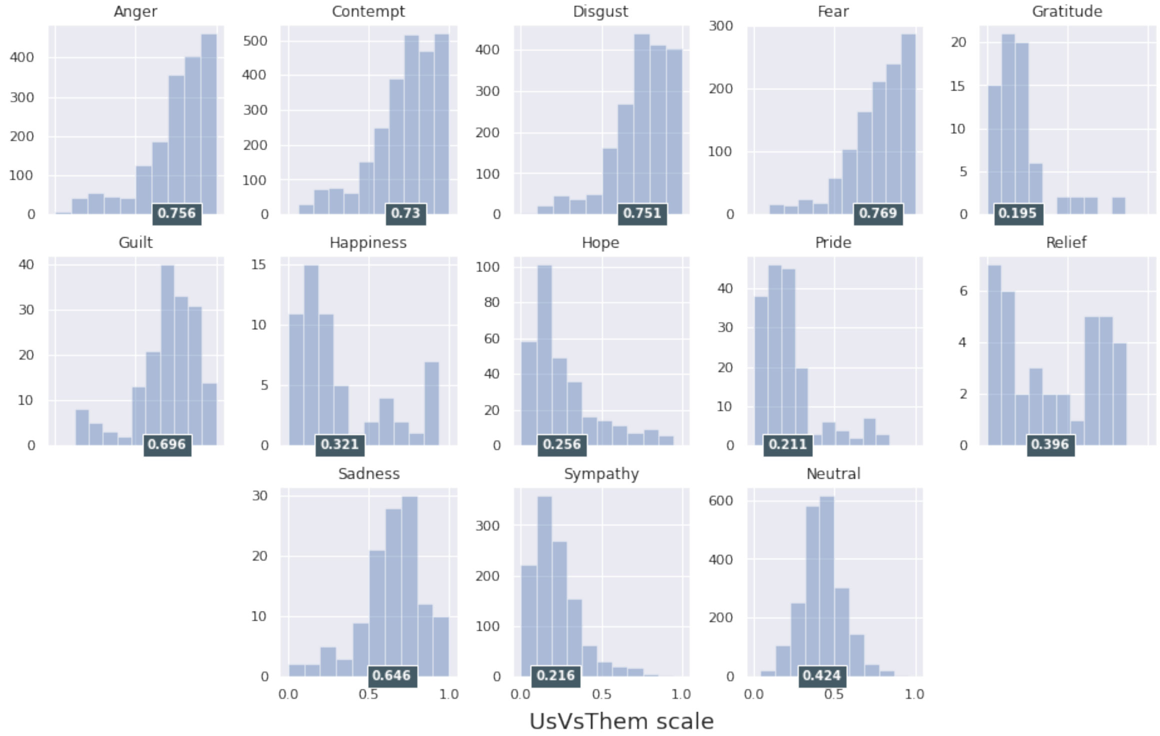


Figure 9: *UsVsThem* scale for each emotion.

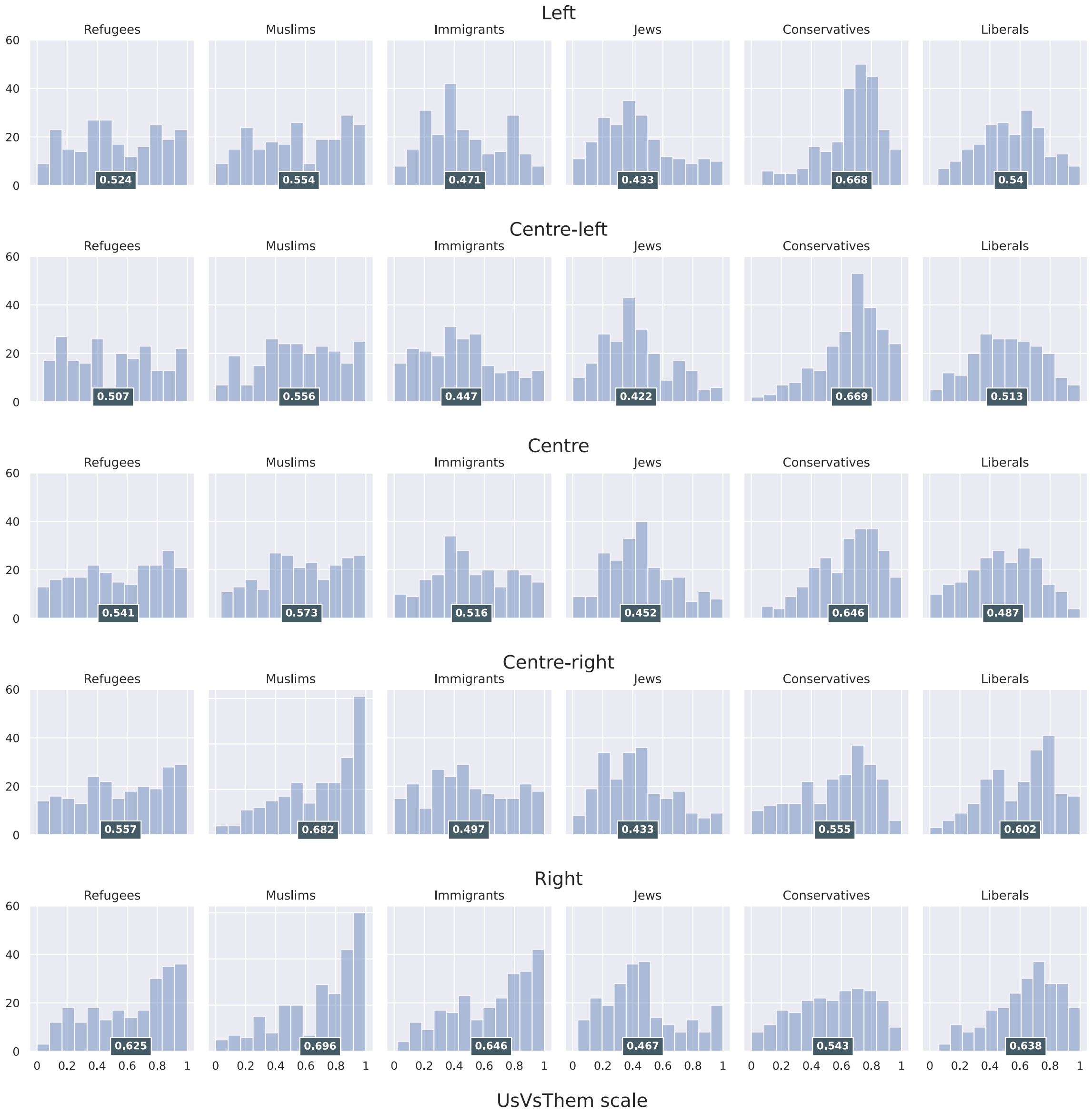


Figure 10: Distribution for the *UsVsThem* scale per social group and bias. The mean for the scale is shown at the x axis.

A.5 Analysis

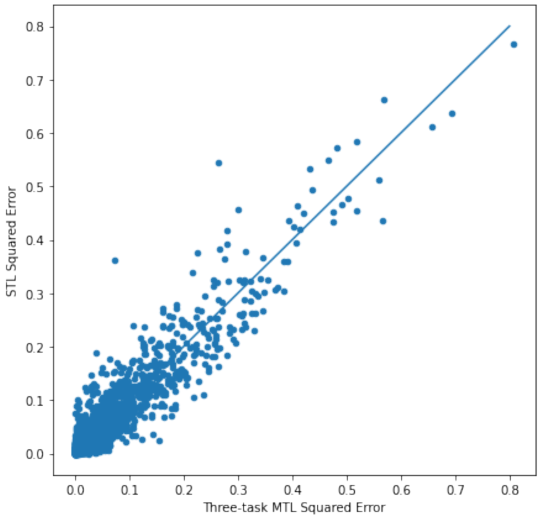


Figure 11: Squared error for STL and three-task MTL

n	**Reddit Comment**
1	I can't believe this bullshit. It's literally come down to picking between letting refugees sleep in your bed and fuck your wife and daughter or you're a racist hate monger. Literally no point on the spectrum exists between the two ends.

Label	**MTL, E. & G.**	**MTL, E.**	**MTL, G.**	**STL**	**Group**	**Emotions**
0.920	0.646	0.752	0.376	0.655	Refugees	Anger & Fear

n	**Reddit Comment**
2	As a legal immigrant, the newfound term 'undocumented immigrant' annoys the heck out of me. They're illegal aliens. Stop trying to sugarcoat it. It took me years to move here legally, and I resent those who chose to do it illegally. The process is long but it is fair. Come in through the front door, not the backdoor.

Label	**MTL, E. & G.**	**MTL, E.**	**MTL, G.**	**STL**	**Group**	**Emotions**
0.746	0.661	0.530	0.577	0.436	Immigrants	Anger & Disgust

n	**Reddit Comment**
3	By every moral or ethical standard, it is your duty to refuse orders to "defend" the US from these migrants. History will look kindly upon you if you do. There are thousands, if not millions, of us who will support your decision to lay your weapons down.

Label	**MTL, E. & G.**	**MTL, E.**	**MTL, G.**	**STL**	**Group**	**Emotions**
0.17	0.923	0.856	0.884	0.83	Immigrants	Sympathy & Hope

n	**Reddit Comment**
4	I was about to be shocked, until i thought about the god damn state of the world, the western world is at the moment at almost the same state, where at least a large minority wish the same thing of the Muslims. That and god damn people THERE IS MILLIONS OF MUSLIMS NOT EVERYONE THINKS THIS WAY!

Label	**MTL, E. & G.**	**MTL, E.**	**MTL, G.**	**STL**	**Group**	**Emotions**
0.099	0.847	0.833	0.882	0.815	Muslims	Sympathy

n	**Reddit Comment**
5	The Democrats are the ones preventing people? That's funny. Who are the lawmakers in the state legislatures that are constantly scheming up roundabout ways to defund planned parenthood and completely outlaw abortion access, despite a large majority of Americans supporting at least some degree of abortion? Hint: they're not Dems.

Label	**MTL, E. & G.**	**MTL, Emo.**	**MTL, Groups**	**STL**	**Group**	**Emotions**
0.059	0.78	0.75	0.766	0.734	Liberals	Sympathy

Table 9: 1 and 2 are examples of predictions for comments with high values on the *UsVsThem* scale where MTL models showed an improvement over STL, 3 and 4 are examples of ambiguous and challenging comments and 5 is an example with mentions to more than one group with high error predictions. Predictions are averages of all 10 seeds predictions for each model.

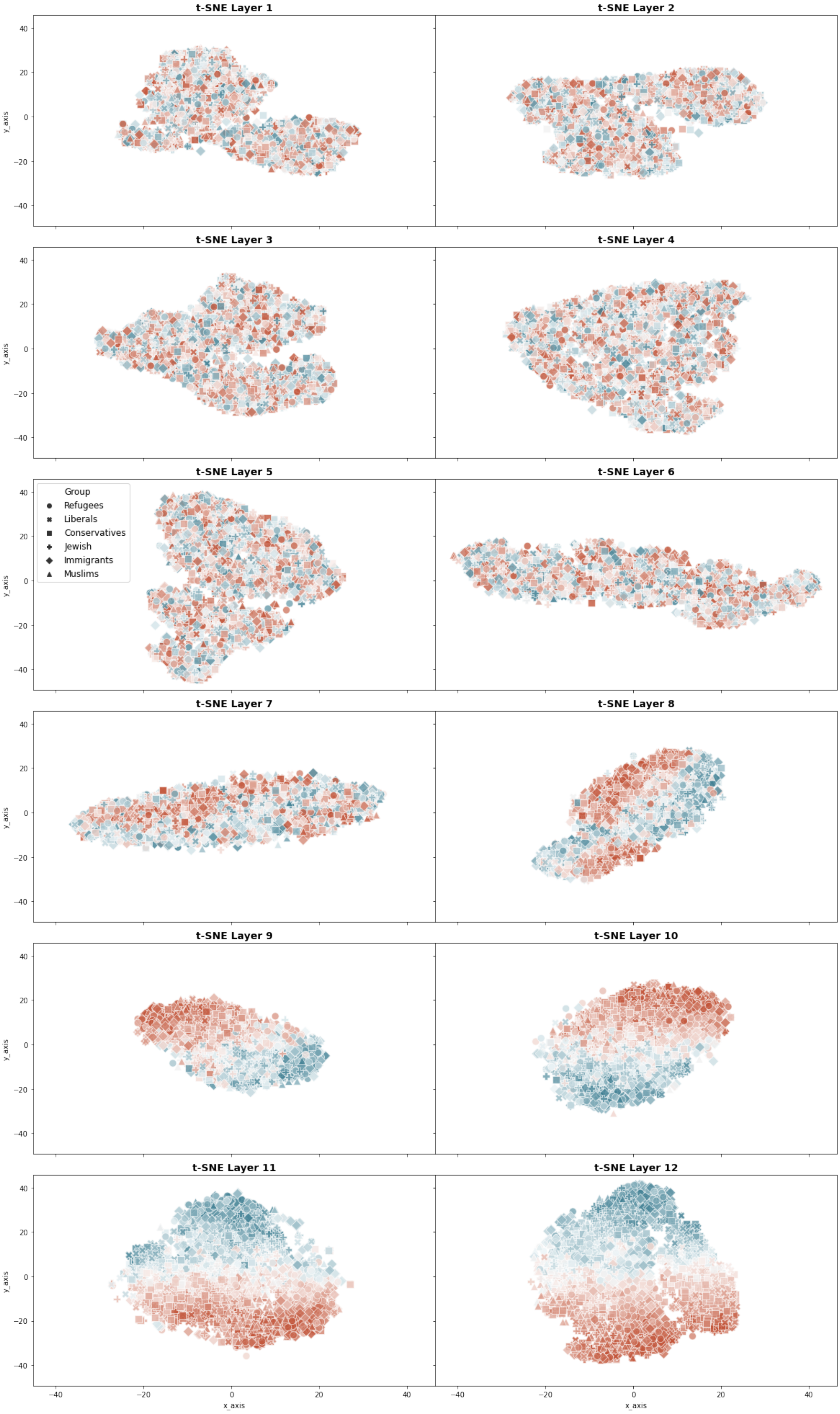


Figure 12: Hidden representations at each layer of the Transformer model for the single task model. Red represents a value closer to 1 in the *UsVsThem* scale and blue closer to 0.

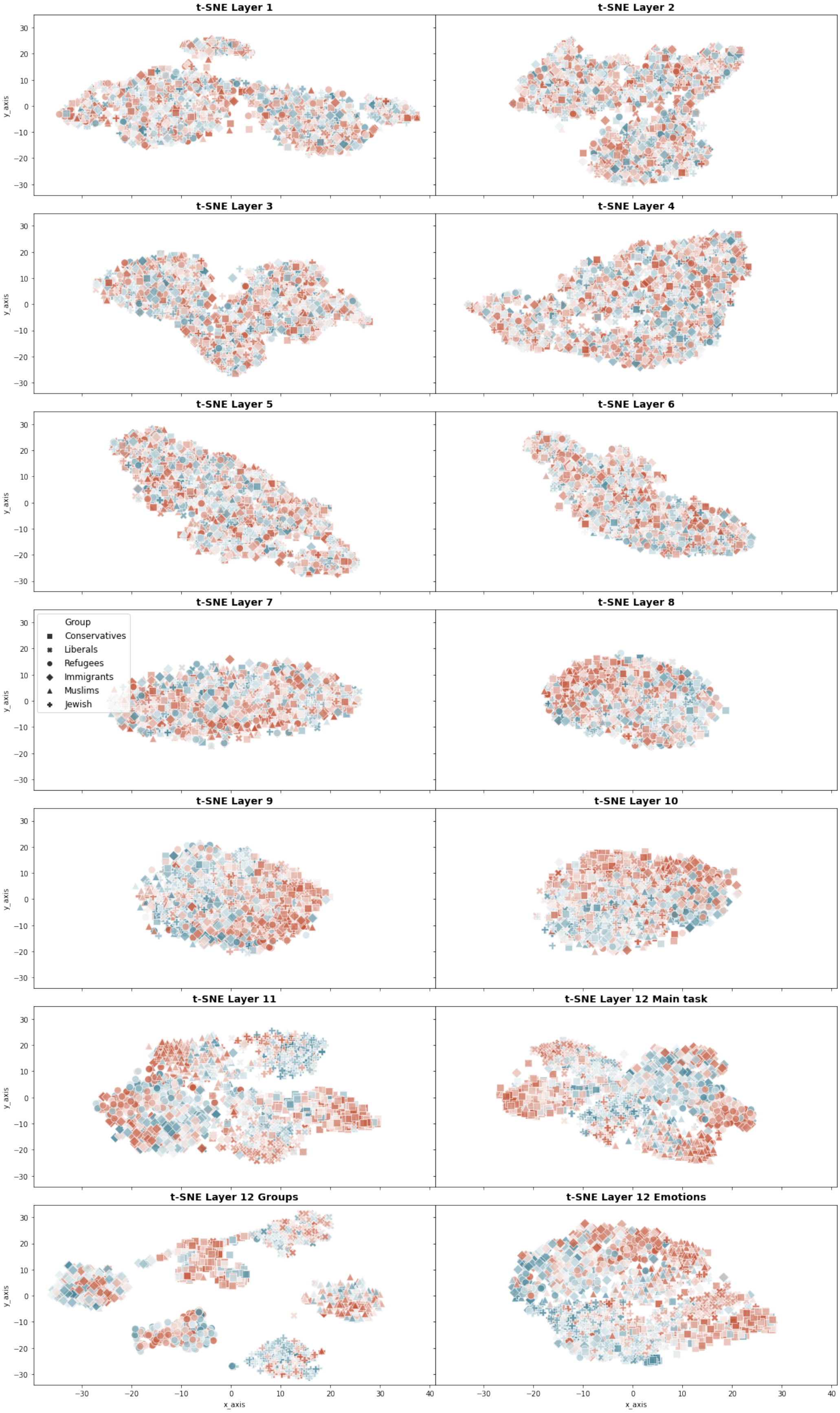


Figure 13: Hidden representations at each layer of the Transformer model for the three-task MTL. The last plots show the task specific Transformer layer output. Red represents a value closer to 1 in the *UsVsThem* scale and blue closer to 0.

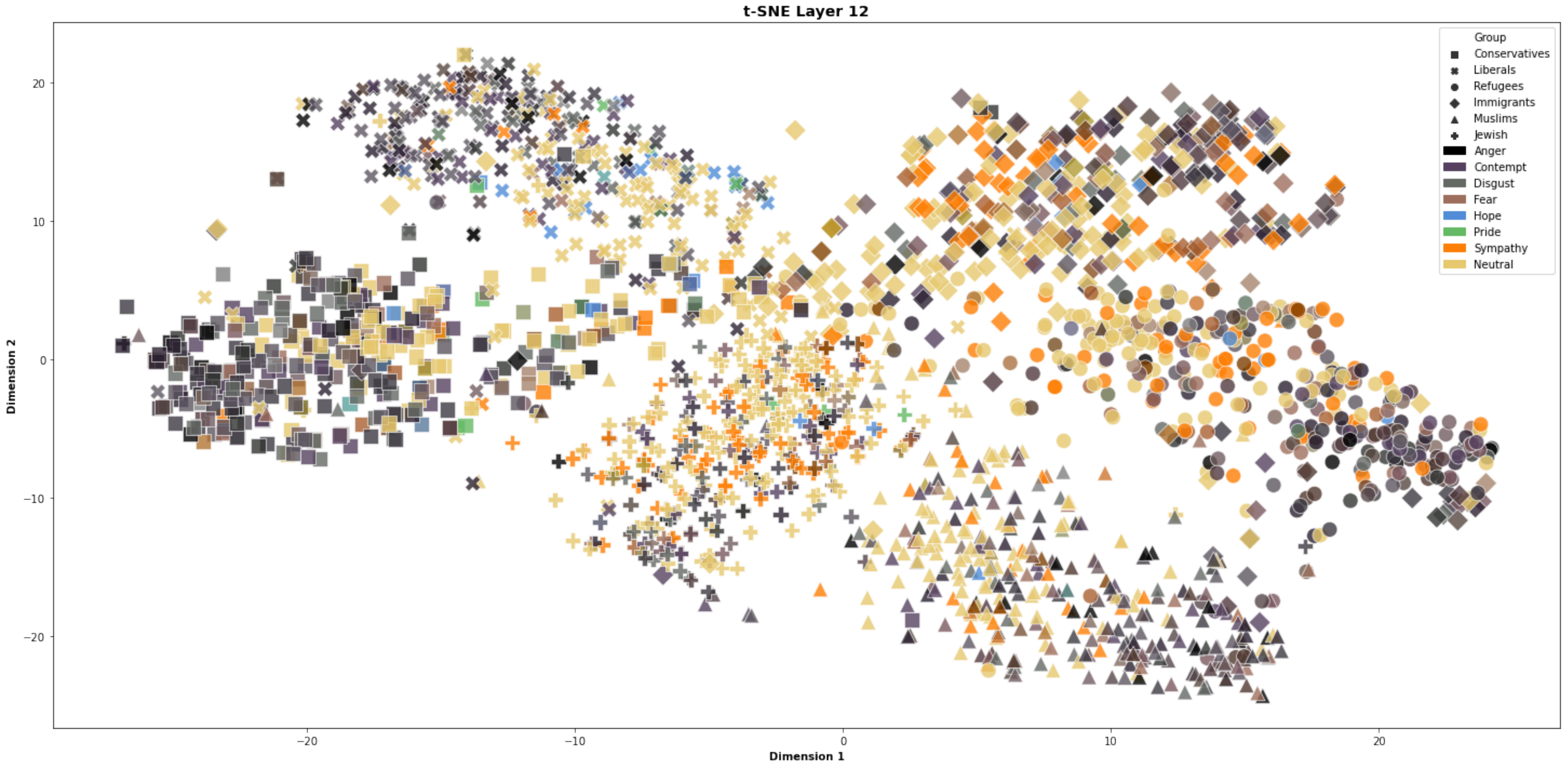


Figure 14: Hidden representations for the three-task MTL main task specific layer. Emotions are represented by colours. Comments with more than one emotion show an average of the colours.

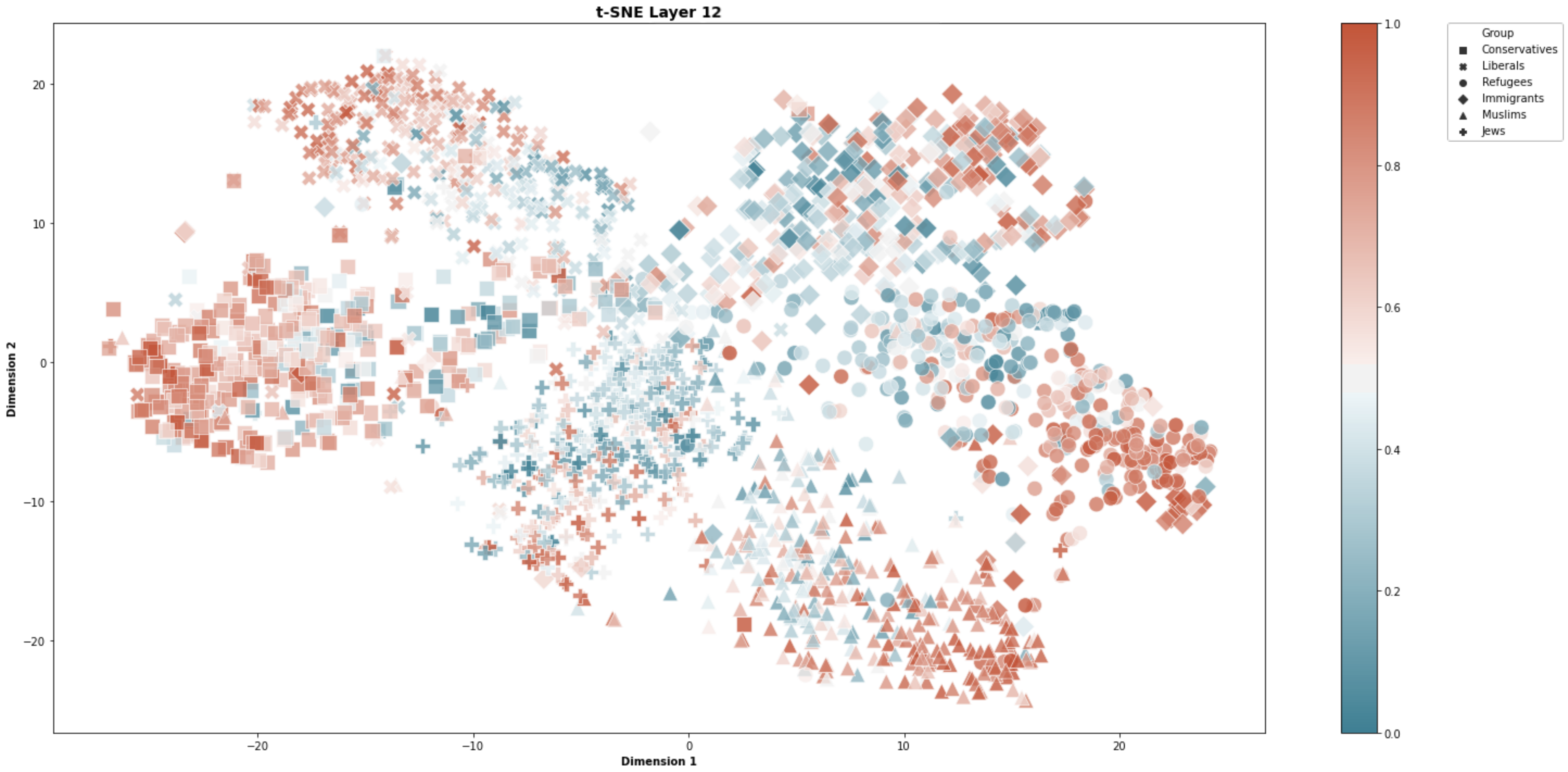


Figure 15: Hidden representations for the three-task MTL main task specific layer. Red represents a value closer to 1 in the *UsVsThem* scale and blue closer to 0.

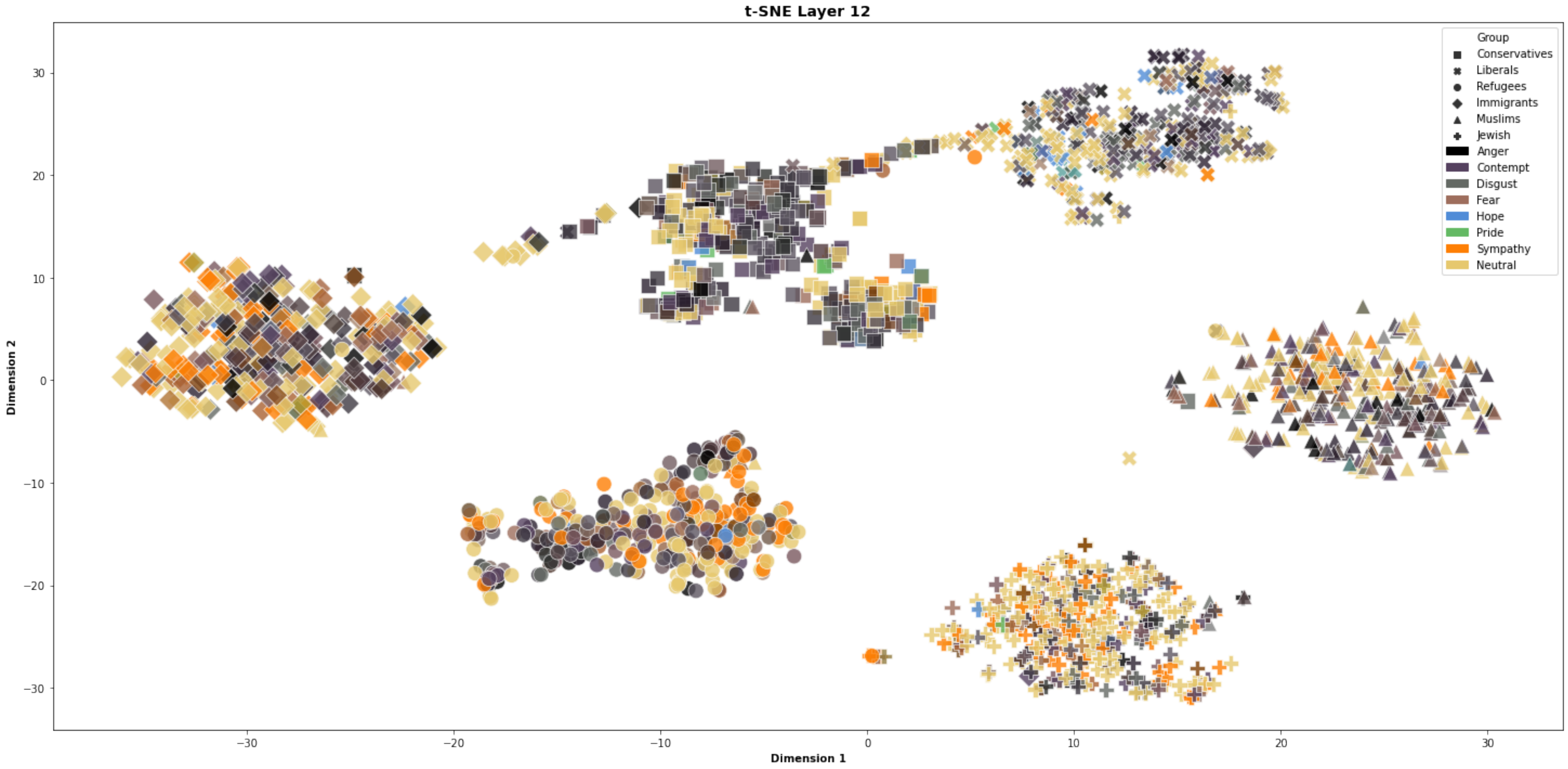


Figure 16: Hidden representations for the three-way MTL group identification specific layer. Emotions are represented by colours. Comments with more than one emotion show an average of the colours.

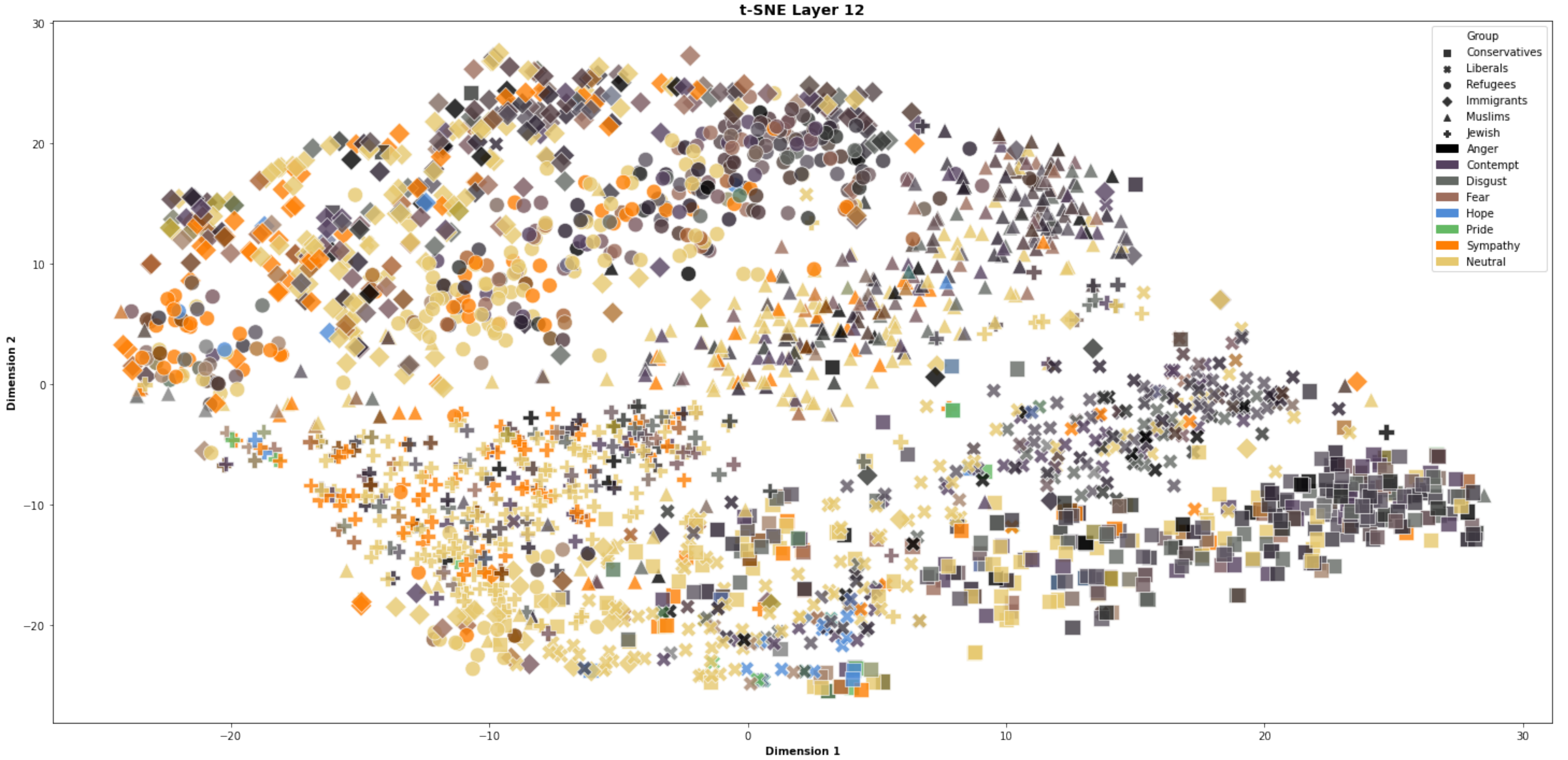


Figure 17: Hidden representations for the three-way MTL emotion specific layer. Emotions are represented by colours. Comments with more than one emotion show an average of the colours.

Multilingual Entity and Relation Extraction Dataset and Model

Alessandro Seganti*[2], Klaudia Firląg[1], Helena Skowrońska*[3],
Michał Satława[1], and Piotr Andruszkiewicz[1,4]

[1]Samsung R&D Institute Poland
[2]Equinix
[3]NextSell, ODC Group
[4]Warsaw University of Technology
alessandro.seganti@gmail.com
{k.firlag, m.satlawa, p.andruszki2}@samsung.com
arvala@wp.pl, p.andruszkiewicz@ii.pw.edu.pl

Abstract

We present a novel dataset and model for a multilingual setting to approach the task of Joint Entity and Relation Extraction. The SMiLER dataset consists of 1.1 M annotated sentences, representing 36 relations, and 14 languages. To the best of our knowledge, this is currently both the largest and the most comprehensive dataset of this type. We introduce HERBERTa, a pipeline that combines two independent BERT models: one for sequence classification, and the other for entity tagging. The model achieves micro F_1 81.49 for English on this dataset, which is close to the current SOTA on CoNLL, SpERT.

1 Introduction

The majority of the – constantly growing – amount of openly accessible knowledge is locked in unstructured text, and hence inefficiently utilized by any systems. The NLP tasks related to this problem include Information Extraction, Relation Extraction, Named Entity Retrieval, as well as Joint Entity and Relation Extraction.

Our contribution is twofold. First, we present SMiLER (Samsung MultiLingual Entity and Relation Extraction dataset): an open-domain corpus of annotated sentences, created for the Joint Entity and Relation Extraction task. With 1.1 M sentences, 36 relation types, and 14 languages, SMiLER seems to be both the largest and the most diversified corpus for the task in existence, to the best of our knowledge. The corpus was semi-automatically created from Wikipedia and DBpedia, and partly checked by linguists.

Our second contribution is HERBERTa – Hybrid Entity and Relation extraction BERT for a multilingual setting that consists of two independently pretrained BERT models (Devlin et al., 2018). The first one classifies the input sequence as belonging to one of our 36 relations (including *no_relation*). Its output – the relation – is then fed to the second BERT, together with the same input sequence. The second model performs entity tagging and outputs two spans for the two entities selected from the input sequence. Our model is close to SpERT (Eberts and Ulges, 2019), the current state-of-the-art for Joint Entity and Relation Extraction, achieving micro F_1 81.49 for English on the presented dataset.

The SMiLER corpus and the source code is available at https://github.com/samsungnlp/smiler/.

2 Related Work

Available Datasets The datasets that are commonly used for the task of Joint Entity and Relation Extraction are still insufficient in size and diversification; furthermore, they are all monolingual English corpora. For instance, CONLL04 (Roth and Yih, 2004) has only 1.7k annotated sentences and 5 relations. ADE (Gurulingappa et al., 2012) is larger, with almost 21k sentences, but it distinguishes only 2 relations (plus no_relation). Yet another such dataset, SciERC (Luan et al., 2018), contains 500 annotated abstracts and 7 relations; however, all of the relations belong to the scientific domain.

We believe that presenting a new, large and diversified dataset will be a valuable contribution to the joint task of Entity and Relation Extraction – both for English and for the multilingual setting.

Information Extraction Information extraction systems collect knowledge that is locked in unstructured text, such as in our Wikipedia articles. A nota-

---

*Work done while at Samsung R&D Institute Poland.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1946–1955
April 19 - 23, 2021.

ble example is the Never-Ending Language Learner (NELL), which was reading the Web for almost 10 years, 2010-2019 (Mitchell et al., 2018). This semi-supervised system was retrained continuously, with the use of the current knowledge, to collect new instances of a pre-defined set of entity types and relations, which also constitutes the final goal of our system. Another well-known knowledge base is Knowledge Vault (Dong et al., 2014). The system extracts information from the Internet (text, tabular data, page structure, human annotations) and combines it with information from Freebase (Bollacker et al., 2008). A probabilistic inference system computes calibrated probabilities of fact correctness.

Instead of utilizing pre-defined entity and relation types, some systems use syntactic analysis: ReVerb (Fader et al., 2011), MinIE (Gashteovski et al., 2017). Another approach is to create a taxonomy with just one relation: isA, as in Probase (Wu et al., 2012).

Relation Extraction Current models (HEBERTa included) for Relation Extraction are based on BERT (Devlin et al., 2018), a multi-layer bidirectional Transformer encoder (Vaswani et al., 2017). Its main contribution is pre-training deep bidirectional representations from unlabeled text by joint conditioning on both left and right context, in all the layers.

The current state-of-the-art in Relation Extraction are REDN (on SemEval-2010 Task 8, NYT, and WebNLG) (Li and Tian, 2020) and Matching the Blanks (on FewRel) (Soares et al., 2019). REDN extracts the token embeddings (BERT) from two different layers to represent the head and tail entities separately, in order to enhance learning reversible relations. Next, it calculates a parameterized asymmetric kernel inner product matrix between all the head and tail embeddings of each token in a sequence. On the other hand, the model constructed by Soares et al. (2019) combines BERT with extensions of Harris' distributional hypothesis to relations, to build task-agnostic relation representations solely from entity-linked text.

Joint Entity and Relation Extraction A more robust approach to Relation Extraction is to combine it with Entity Extraction. Three such joint solutions have achieved the current state-of-the-art: SciBERT (on SciERC) (Beltagy et al., 2019), SpERT (for RE on CONLL04) (Eberts and Ulges, 2019), and End2End Joint NER & RE (for NER on CONLL04) (Giorgi et al., 2019).

SciBERT leverages unsupervised pretraining on a multi-domain corpus of scientific publications, as this domain differs significantly from the general domain used by the original BERT. The second model, SpERT, is an attention model for Span-Based Joint Entity and Relation Extraction. It is trained using within-sentence negative samples, which are extracted in a single BERT pass. Finally, End2End Joint NER & RE combines BERT with biaffine attention.

Distant Supervision *Distant supervision* proposed by Mintz et al. (2009) for Relation Extraction task is a training paradigm, based on the assumption that if there is a relation between entities, then every sentence containing them may also express that relation. In the experiments, multiple sentences containing the entities were used to create their feature vectors. Thus, this approach allows a prediction of the relation between two entities, but does not identify sentences containing useful and correct cues for this relation in the corpus.

Data collection for SMiLER was based on the same assumption, but thanks to the subset of manually validated examples, the dataset could be used as the sole source of supervision during training, where the algorithm was supervised by the database.

Multilingual BERT M-BERT (Pires et al., 2019) is a model that is particularly relevant to our multilingual approach, and one that we tested repeatedly. This is a single language model, pre-trained from monolingual corpora in 104 languages, which performs zero-shot cross-lingual model transfer. Task-specific annotations in one language are used to fine-tune the model for evaluation in another language. Such transfer is possible even to languages in different scripts, such as English and Korean.

M-BERT has become the basis for building other multilingual models, as well as monolingual ones for languages other than English. We have experimented with three such models: German BERT, Italian BERT, and KoBERT for Korean (see section 4 for details).

3 Data

Our SMiLER corpus consists of 1.1 M annotated Wikipedia sentences, and was created specifically for the task of Entity and Relation Extraction. It

	EN-full	EN-mid	EN-small	KO	IT	FR	DE	PT	NL	PL	ES	AR	RU	SV	FA	UK	Total
sentences	748k	269k	35k	20k	76k	62k	53k	45k	40k	17k	12k	9k	7k	5k	3k	1k	1.1M
relations	36	36	32	28	22	22	22	22	22	22	22	9	8	22	8	7	36

Table 1: SMiLER: number of sentences and relations for each language. EN-mid includes EN-small, and EN-full includes EN-mid.

consists of annotated sentences in 14 languages, with each language representing a subset of 36 relations. As far as we know, this is both the biggest and the most comprehensive corpus for these NLP tasks. Table 1 provides the number of sentences and relations for each language, while Table 2 shows a few examples from the corpus.

The relations belong to 9 rough domains: Person (e.g. has-child, has-occupation), Organization (e.g. org-has-member, headquarters), Location (e.g. loc-leader, has-tourist-attraction), Animal (e.g. has-lifespan, eats), Art (e.g. starring, has-author), Device (e.g. invented-by), Event (event-year), Measurement (e.g. has-length), and no_relation. The no_relation sentences do not contain any of the 35 "positive" relations.

Figure 1 shows the number of sentences for each relation, for each language.

Relation	Sentence
has-child	[**Bill**]$_{e1}$ married Hillary on October 11, 1975, and their only child, [**Chelsea**]$_{e2}$, was born on February 27, 1980.
head-quarters	[**AMC Airlines**]$_{e1}$ è una compagnia aerea egiziana con sede al [**Cairo**]$_{e2}$, esegue voli charter da Sharm el-Sheikh, Hurghada, Il Cairo verso le maggiori capitali europee dal maggio 2006.
movie-has-director	[**Lili**]$_{e1}$ ist ein US-amerikanischer Spielfilm des Regisseurs [**Charles Walters**]$_{e2}$ aus dem Jahr 1953.

Table 2: Examples from the SMiLER corpus.

3.1 Dataset Building

To collect the English dataset, we queried DBpedia for (entity$_1$, entity$_2$, relation) triples. The articles about entity$_1$ that would also contain entity$_2$ were obtained from a Wikipedia dump. They were parsed and automatically selected. The complete English dataset is referred to as EN-full in the following discussion.

During this process it is possible that the same sentence will appear multiple times containing:

1. different entity annotations for the same relation, or
2. different entity annotations for another relation.

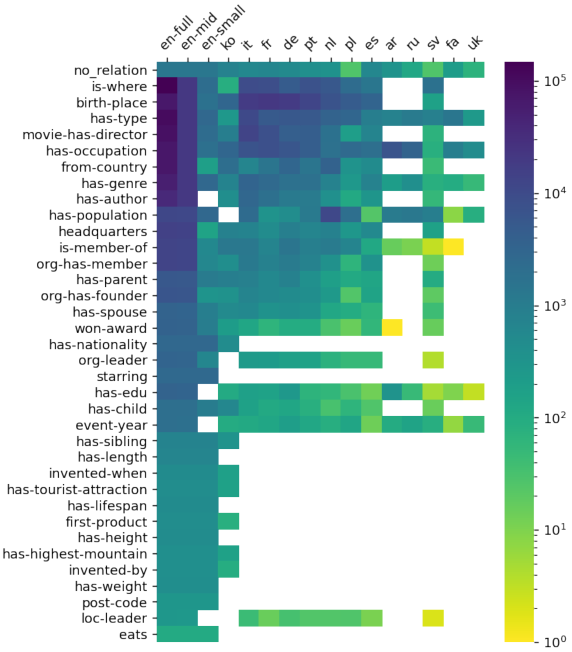


Figure 1: SMiLER: the number of sentences for each relation and for each language.

The first case is unlikely, because sentences are taken from the Wikipedia article of the "main" entity. The second case is possible, though.

Theoretically, it is possible to write grammatical rules in order to do a "cross-match" for augmenting a number of annotated sentences, and it has been done previously (e.g. Wu et al. (2012)). The reason why we did not use grammatical rules is that we wanted to scale the approach to multiple (and diverse) languages and the rules that could help with this task are not universal.

A part of EN-full was manually validated by linguists. The linguistic validation was more in-depth for English than for other languages, as for English the task was to correct the entity annotation whenever possible. Hence, passable annotations were corrected for English, while they were simply assessed as correct for the other languages. Overall, the linguists assessed 58.2% of the sample as fully correct, corrected the annotations in further 18.4%, and assessed the remaining 23.4% as incorrect. The correct 58.2% plus the corrected 18.4% of the sample together form the EN-small dataset.

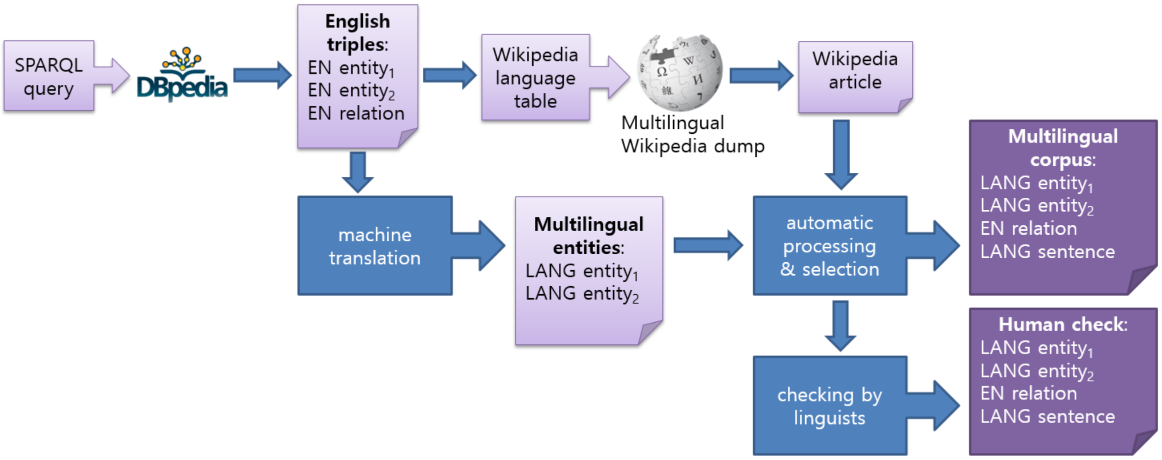


Figure 2: The process of creating the multilingual parts of SMiLER.

The corpus-building process was adapted for the other languages. First, the entities (as well as the entire no_relation sentences) were translated from English. The translation was carried out by an in-house multilingual system similar to (Przybysz et al., 2017; Williams et al., 2018; Wetesko et al., 2019). After translating the entities, the relevant article was copied from the Wikipedia dump in the other language. Finally, samples from the corpus were checked by linguists. Figure 2 illustrates the process.

The no_relation sentences in English have been selected by linguists from Wikipedia articles. The sentences contained entities that were in a relation in another sentence in the data set but were not in a relation in the selected sentence. Since the process is so strict, the translation to other languages was done automatically. All no_relation sentences needed to be hand-checked, because they could contain some hidden relations, and for this reason preparing original sentences for all the languages would take too much time. This is not the case for the 'positive' relations; e.g. if 'Bill' is the father of 'Chelsea', then a sentence containing both 'Bill' and 'Chelsea' – and coming from a Wikipedia article on Bill Clinton – would probably also express the relation has-child. However, if a sentence contains 'Bill', but not 'Chelsea', it could still express another of the "positive" relations.

We also tried other automated approaches for finding no_relation sentences. For example, given a Wikipedia article with a relation found, we could find other sentences with the same entities and assume all of them are no_relation sentences. Unfortunately this approach caused too much noise and was not used in the end.

The linguists verified a random sample of 50 sentences for 16 largest relations + no_relation for each of the following languages: IT, FR, DE, PT, ES, KO. The results are shown in Table 3. The overall percentage of correctness reached 79-80%, which is very high, considering the level of automation involved in the process. The selected random sample was small, because the verification needed to be cost- and time-effective. Still, the results were thoroughly checked and they were largely consistent.

	IT	FR	DE	PT	ES	KO	Overall
positive relations (scraped)	79	70	76	84	84	78	79
no_relation (translated)	88	94	86	86	74	53	80

Table 3: The percentage of correct sentences for selected languages in SMiLER.

The most common errors found by the linguists belong to three groups: (1) unexpected DBpedia query results, (2) wrong sentence parsing (deleting the second half of the sentence or including HTML), (3) selecting random sentences, which either do not express the relation, or have the entities marked in wrong places. Other typical errors are related to missing translations of some English words and *no_relation* sentences that contain one of the 35 positive relations. Table 4 shows one example of each error.

3.2 Datasets for the Model

For each language in our multilingual dataset, we automatically extracted 2% of the sentences to create a dev set and another 2% for a test set. This corpus split maintained the distribution of the rela-

Relation	Error	Sentence with an error
has-type	unexpected type "cardinal" obtained from DBpedia	[**Gerhard Ludwig Müller**]$_{e1}$ (Finthen, 31 dicembre 1947) è un [**cardinale**]$_{e2}$, arcivescovo cattolico e teologo tedesco, prefetto emerito della Congregazione per la dottrina della fede dal 1º luglio 2017.
headquarters	random sentences and deleted words	Temple des Martyrs à [**Taipei**]$_{e2}$. Les [**Forces armées de la république de Chine**]$_{e1}$ sont constituées d'une force d'active d'environ et de .
has-type	HTML included	vignette|gauche|250px|La salle de concerts de Raanana [**Ra'anana**]$_{e1}$ est une [**ville**]$_{e2}$ israélienne d'environ habitants au nord-est de Tel Aviv, dans le sud de la région de Sharon.
no_relation	untranslated "ranks"	[**iSuppli**]$_{e1}$ ranks [**Kingston**]$_{e2}$ como el fabricante de módulos de memoria número uno del mundo para el mercado de memoria de terceros por el décimo año consecutivo.
no_relation	sentence contains relation has-genre	Nach der Registrierung seines aktuellen YouTube-Kanals 2010 veröffentlichte [**Kjellberg**]$_{e1}$ vor allem [**Let's Play**]$_{e2}$ Videos von Horror- und Action-Videospielen.

Table 4: Most common error types found by linguists.

tions in the original data, whenever possible. For the languages where the label distribution was impossible to preserve, the number of relations in the test set may be smaller than in the train set.[1]

For English, we used a single test set for both EN-mid and EN-small. The only difference was in the number of labels: EN-mid (both train and dev) had 36 labels, while EN-small (albo both train and dev) – 32 (see Table 1).

For training the models, we created several combinations of languages, in order to examine the effect of adding/removing languages from the training set. We experimented with the following combinations:

1. **EURO**: IT, FR, PT, DE, ES, EN
2. **SVO**: **EURO**, RU, SV, NL, PL, UK
3. **ALL**: **SVO**, AR, KO, FA
4. Each individual language combined with EN (e.g. IT+EN)
5. Each language alone (e.g. IT)

Pires et al. (2019) showed that the language structure had an impact on the performance of the model. Therefore, we treat Korean, Farsi and Arabic as a special case because they are non-SVO languages (Korean and Farsi are SOV, while Arabic is VSO).[2]

4 Model

Our architecture HERBERTa (Figure 3) uses two pre-trained BERT models. It solves the problem of Entity and Relation Extraction with the use of an unconventional pipeline. In the first step it is trained for Relation Extraction, while the entities are retrieved in the second stage.

The first model is BERT, fine-tuned for the Sequence Classification task. Its input is a tokenized sequence, while its output consists of a sequence output and a pooled output that represents the overall sentence context (from the [CLS] token). The latter is passed to a softmax for classifying the relation.

The second model is our implementation of BERT for Entity Tagging. It is based on BERT for Question Answering, and its inputs are:

- the same tokenized sequence as the one used for Relation Extraction,
- the relation outputted in the first step – encoded as a BERT [unused] token.

As a result, the input sequence is as follows: [CLS] [relation] [SEP] [E0] [E1] ... [En] [SEP]. The output of the model is a set of four indices that correspond to the spans of the two entities having the relation.

At inference time, the model will return the start and end of the two entities separately and this is used to mark the entities. Using our model, it is also possible to select N best predictions (one prediction = one entity pair) for the same relation. This has not been used for the result of the paper because we wanted to find a single best entity pair for each sentence.

---

[1] Train/test set number of relations for the languages in which the numbers differ: ES (train: 21, test: 16), RU (train: 8, test: 7), SV (train: 22, test: 14), FA (train: 8, test: 4), AR (train: 9, test: 7), NL (train: 22, test: 21), PL (train: 21, test: 20), UK (train: 7, test: 6), KO (train: 28, test: 26).

[2] SVO, SOV, and VSO stand for the relative position of the Subject, Verb, and Object in the typical affirmative sentence.

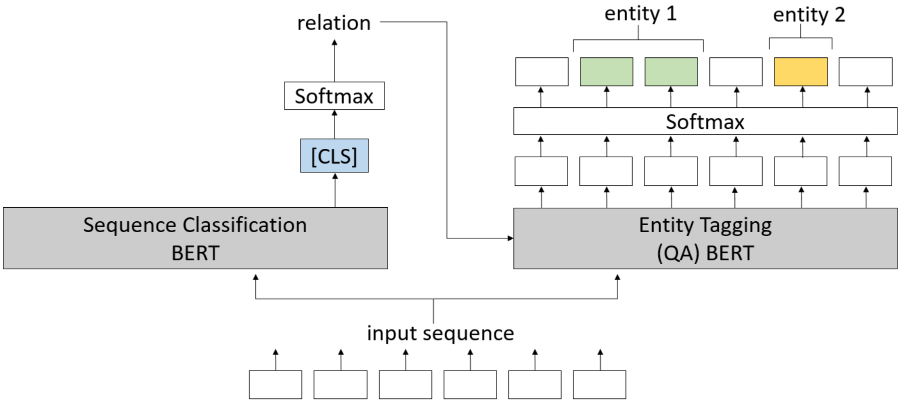


Figure 3: Model architecture.

We used the following types of BERT, depending on the language(s):

- bert-base-cased (Devlin et al., 2018),
- bert-base-multilingual-cased (Pires et al., 2019),
- bert-base-german-cased from the HuggingFace Transformer library (Wolf et al., 2019),
- bert-base-korean-cased – the monologg/kobert model from HuggingFace,
- bert-base-italian-cased – the dbmdz/bert-base-italian-cased model from HuggingFace.

Below we call our model LANG(B) if it is a fine-tuning of bert-base, e.g. DE(B) is the bert-base-german model fine-tuned on the German corpus. We call our model just LANG if it is a fine-tuning of bert-base-multilingual, e.g. DE-EN is the English and German model trained using bert-base-multilingual. The English dataset has two versions: EN (EN-mid) and EN_S (EN-small). We did not use EN-full for the training.

4.1 Training Procedure

The two models presented in Section 4 are trained independently, each one with a different loss function. The first model uses standard cross-entropy as its loss for relation classification. The loss function for the second model is:

$$\mathcal{L} = \frac{1}{4}(\mathcal{L}^{1,start} + \mathcal{L}^{1,end} + \mathcal{L}^{2,start} + \mathcal{L}^{2,end}) \quad (1)$$

Where $\mathcal{L}^{j,start}$ is the cross-entropy loss for the prediction of j-th entity first token index and $\mathcal{L}^{j,end}$ is the cross-entropy of the predictions of its last token index.

The result of the first model becomes the input to the second one, during inference. We conclude that this works well because the performance of the relation classification model is very high (please refer to Table 5 and the Relation column).

The number of epochs is established using an early stopping mechanism. On average, each model (Relation Classifier and Entity Tagger) is trained for around 6 epochs for models with EN-mid set and 3-4 epochs for others. We fine-tune the whole BERT.

The model with all languages (ALL_EN) runs on 6 GPU GeForce RTX 2080 (8GB) for 31h.

5 Results

For evaluating the models, we use the micro F_1 score on the entities and relations together, ignoring no_relation. Below, this measure is referred to as Combined. When comparing with the state-of-the-art, we also show the relation-only micro F_1, as well as the entity-only micro F_1 (for a single entity and for the pair of entities). We use such metrics because they are typically calculated for these tasks; see for instance Eberts and Ulges (2019).

5.1 Comparison with the State-of-the-Art

Given that our dataset is new, there is no state-of-the-art for it. Nevertheless, we decided to compare the results of HERBERTa with SpERT (Eberts and Ulges, 2019), the currently best model on CoNLL (Roth and Yih, 2004). Table 5 presents the results that we obtained by training SpERT on our Wikipedia EN-mid (denoted as EN(SpERT)) and EN-small (denoted as EN_S(SpERT)). Additionally, Table 5 also shows the results of our model trained on the same datasets. In terms of combined F_1, HERBERTa is close, the difference being about 0.2

percentage point with SpERT, the current SOTA on CONLL.

Model	Relation	Entities (Pair)	Combined
EN_S(SpERT)	N/A	80.71	59.24
EN(SpERT)	N/A	**92.89**	**81.71**
EN_S(B)	80.94	61.94	58.31
EN(B)	**94.94**	82.55	81.49
ALL-EN_S	81.64	57.46	53.76
ALL-EN	93.85	78.26	76.97

Table 5: Micro F_1 results on the Wikipedia EN-mid dataset. Combined – both the relation and the entity pair are correct.

5.2 Language and Model Comparison

Single Language As the first step, we trained HERBERTa for each language separately. The results (F_1 measure) are presented in Figure 4 (left). Our models trained with the use of multilingual BERT are named after language, e.g. IT, FR, PT. Our non-multilingual BERT models are called LANG plus (B), e.g. IT(B), DE(B).

We obtained the best result for EN(B). BERT trained on EN-small (EN_S(B)) achieved lower results, 81 vs. 58, because EN-small contains over 7 times fewer sentences. If we compare this with the other languages, the relationship between F_1 and the logarithm of the number of sentences holds in general. That is, the higher number of sentences, the higher F_1 (see Figure 5). EN-small (35k sentences, F_1=58) falls between NL (40k sentences, F_1=60) and PL (17k sentences, F_1=50). RU (7k) and UK (1k, the smallest set) achieve F_1=29 and 10 respectively.

Surprisingly, FA (3k) and SV (5k) – despite small sets – yield F_1=65 and 58 respectively; as a result, they are the outliers in Figure 5. AR (9k) achieves a higher F_1 score than ES (12k); however, the difference is just 2 percentage points. FA, AR (and KO) have different word orders than all the other languages, which might be one of the reasons why they achieve high F_1 despite small datasets. Another reason might be their small number of relations. The high result for SV, which is SVO, could be explained by a smaller number of relations in the test set than in the train set. UK and RU, in spite of their small number of relations, obtain the lowest results, because they have small datasets. Furthermore, contrary to SV, UK and RU have rich inflection which might also be the case.

Comparing the results of our models trained with multilingual BERT versus non-multilingual BERT, we observe that all three possibilities are present in the pairs we checked. For DE we get the same results F_1=58, for IT the non-multilingual BERT gives lower results 64 vs. 66, while for KO the non-multilingual BERT achieves significantly higher results, 51 vs. 42.

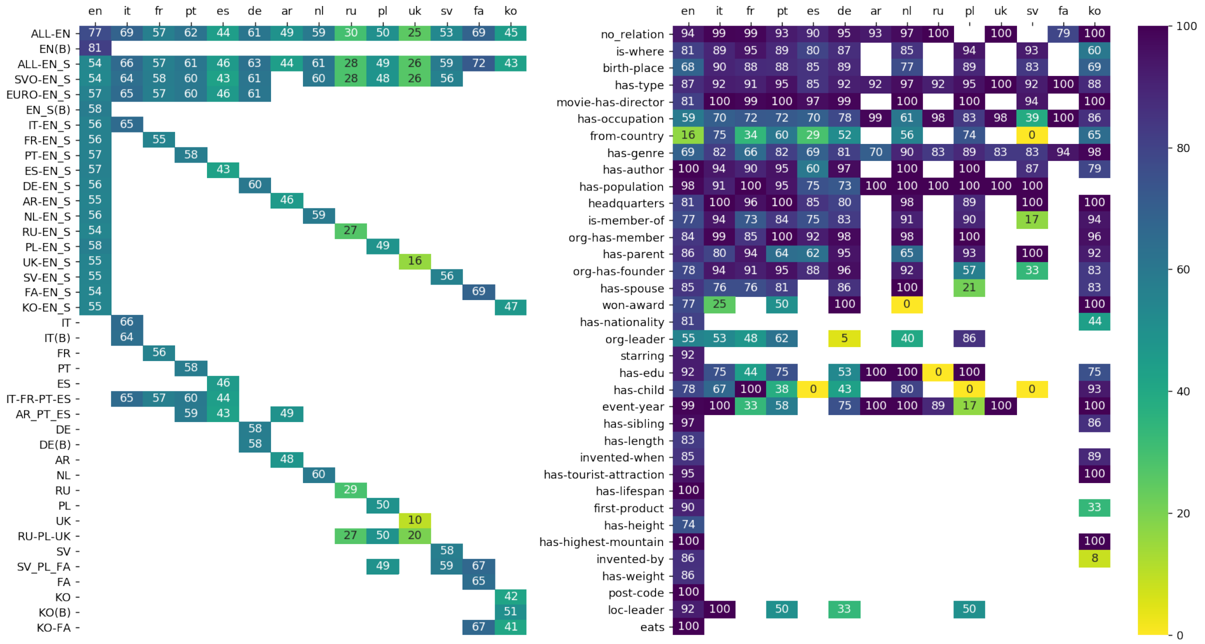
Multilingual In the second step, we added models trained with the multilingual BERT for several languages simultaneously, e.g. ALL-EN for all the languages available in the corpus (see Figure 4 (left)). There are four main groups of models: (A) a model with EN-mid and all other languages, (B) models with EN-small and at least 6 languages, (C) models with EN-small and one other language, (D) other models with about 3-4 similar languages.

Comparing the models for single languages with group A and B models, we observe that the multilingual models typically yield similar or better results. The largest increase was observed for UK: F_1=10 on its own and even 26 in ALL-EN_S. We tried to group similar languages RU, PL, and UK to boost UK. However, we obtained just F_1=20 for UK, which is still significantly higher compared to the UK single language model, but lower compared to UK-EN and UK-EN_S.

One exception is EN-mid, which works significantly better in the single model (81 vs. 77); for EN-small, the difference is less pronounced (58 vs. 54-57). Another exception is KO, for which the single KO(B) model obtains 51 vs. multilingual 43-47. However, if we compare the multilingual KO, F_1=42, then the multilanguage models increase the results. Finally, the model that groups IT, FR, PT, and ES achieves similar results to ALL-EN_S.

Relations Figure 4 (right) shows F_1 for relations. The results differ widely between the languages and the relations. For instance, for EN some relations achieve F_1=100 (e.g. has-lifespan, eats), while one relation gets just F_1=22 (from-country). However, the same relation achieves F_1=73 for PL and even 0 for SV. We conclude that the results depend on the number of training examples for each relation.

Entity$_1$ vs Entity$_2$ Figure 6 demonstrates a significant difference in results between the two entities in the sentence. We obtain far higher results for entity$_1$ (left) than for entity$_2$ (right). This is because entity$_1$ usually occurs at the beginning of the sentence, while the position of entity$_2$ is not so deterministic.



	en	it	fr	pt	es	de	ar	nl	ru	pl	uk	sv	fa	ko
ALL-EN	77	69	57	62	44	61	49	59	30	50	25	53	69	45
EN(B)	81													
ALL-EN_S	54	66	57	61	46	63	44	61	28	49	26	59	72	43
SVO-EN_S	54	64	58	60	43	61		60	28	48	26	56		
EURO-EN_S	57	65	57	60	46	61								
EN_S(B)	58													
IT-EN_S	56	65												
FR-EN_S	56		55											
PT-EN_S	57			58										
ES-EN_S	57				43									
DE-EN_S	56					60								
AR-EN_S	55						46							
NL-EN_S	56							59						
RU-EN_S	54								27					
PL-EN_S	58									49				
UK-EN_S	55										16			
SV-EN_S	55											56		
FA-EN_S	54												69	
KO-EN_S	55													47
IT		66												
IT(B)		64												
FR			56											
PT				58										
ES					46									
IT-FR-PT-ES		65	57	60	44									
AR_PT_ES				59	43		49							
DE						58								
DE(B)						58								
AR							48							
NL								60						
RU									29					
PL										50				
UK											10			
RU-PL-UK									27	50	20			
SV												58		
SV_PL_FA										49		59	67	
FA													65	
KO														42
KO(B)														51
KO-FA													67	41

	en	it	fr	pt	es	de	ar	nl	ru	pl	uk	sv	fa	ko
no_relation	94	99	99	93	90	95	93	97	100		100		79	100
is-where	81	89	95	89	80	87		85		94		93		60
birth-place	68	90	88	88	85	89		77		89		83		69
has-type	87	92	91	95	85	92	92	97	92	95	100	92	100	88
movie-has-director	81	100	99	100	97	99		100		100		94		100
has-occupation	59	70	72	72	70	78	99	61	98	83	98	39	100	86
from-country	16	75	34	60	29	52		56		74		0		65
has-genre	69	82	66	82	69	81	70	90	83	89	83	83	94	98
has-author	100	94	90	95	60	97		100		100		87		79
has-population	98	91	100	95	75	73	100	100	100	100	100	100		
headquarters	81	100	96	100	85	80		98		89		100		100
is-member-of	77	94	73	84	75	83		91		90		17		94
org-has-member	84	99	85	100	92	98		98		100				96
has-parent	86	80	94	64	62	95		65		93		100		92
org-has-founder	78	94	91	95	88	96		92		57		33		83
has-spouse	85	76	76	81		86		100		21				83
won-award	77	25		50		100		0						100
has-nationality	81													44
org-leader	55	53	48	62		5		40		86				
starring	92													
has-edu	92	75	44	75		53	100	100	0	100				75
has-child	78	67	100	38	0	43		80		0		0		93
event-year	99	100	33	58		75	100	100	89	17	100			100
has-sibling	97													86
has-length	83													
invented-when	85													89
has-tourist-attraction	95													100
has-lifespan	100													
first-product	90													33
has-height	74													
has-highest-mountain	100													100
invented-by	86													8
has-weight	86													
post-code	100													
loc-leader	92	100		50		33				50				
eats	100													

Figure 4: Left: F_1 (label and both entities correct) for all languages and models. Right: F_1 (label correct) for all languages and labels, averaged over the models available for each language.

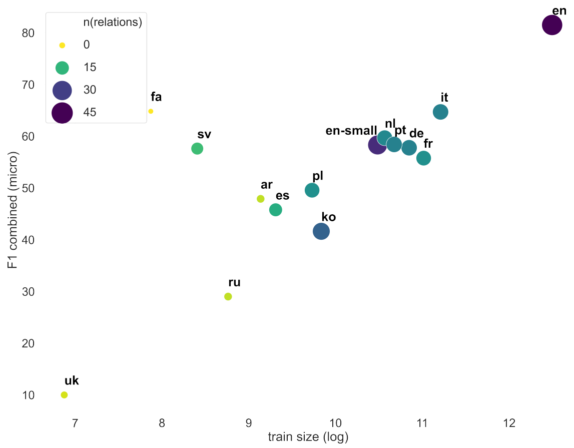


Figure 5: F_1 Combined for single language models versus the number of sentences in the train set. Dot size is proportional to the number of relations in the test set.

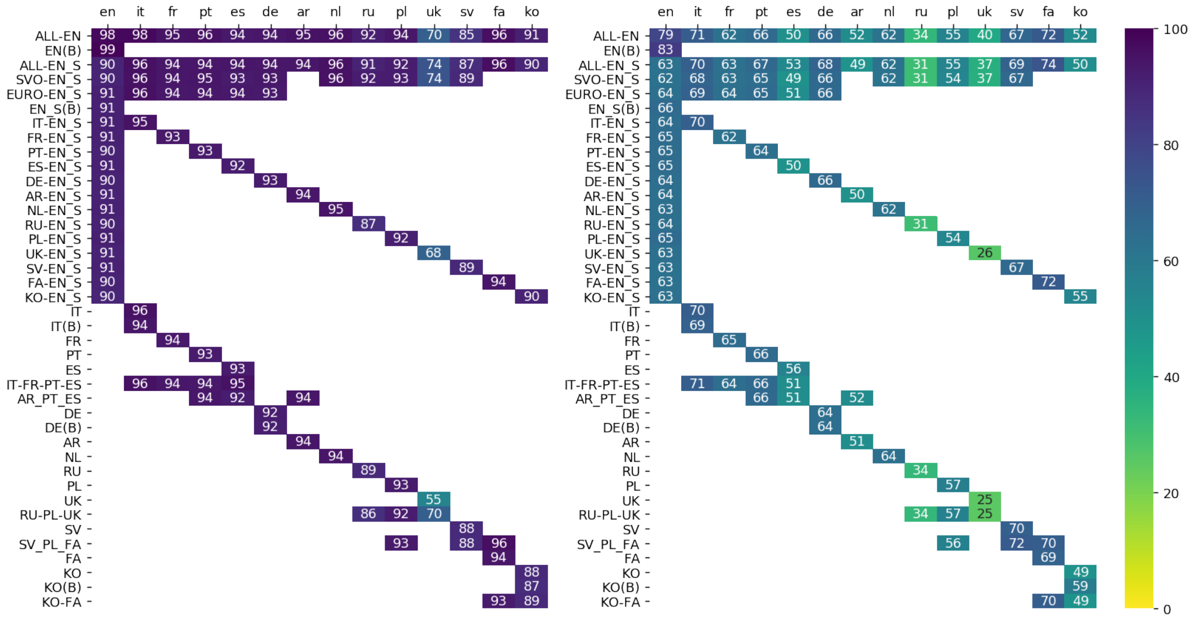
It is important to note though that typically sentences begin with their linguistic subject. As a result, for the sentence 'Bill has a child called Chelsea.', the relation has-child is far more plausible than has-parent, even though both are logically correct. For this reason, the fact that entity$_1$ has a more deterministic behavior is not a problem.

6 Conclusions

We have described our approach to solving the task of multilingual Joint Entity and Relation Extraction, by training our novel, HERBERTa model on SMiLER – our large, comprehensive dataset. The model combines two independent BERT models: one for Sequence Classification, and the other for Entity Tagging. Regarding F1 measure our model is close to SpERT (the current state-of-the-art for CONLL04), the difference being 0.2 percentage point. What is more, the SMiLER dataset appears to be currently the largest (1.1 M annotated sentences) dataset for this task, as well as the most comprehensive one (14 languages, 36 relation types).

We observe that our multilingual models achieve higher or similar results, compared to the models trained for each language separately. Languages with less data can benefit the most from such multilingual models. The victim here is English, as it seems to be a giver of F1 to other languages, especially when the number of sentences for this language is significantly higher than for other languages. On the one hand, due to the large amount of data for English, a model is well trained for patterns existing in this language and thus other languages with less data can benefit from it (because of some similarities between languages). Therefore, we observe increased results for less resourced languages. On the other hand, a model tries to accommodate the nuances of the less resourced languages. Their features are noticeable for the model, which reduces the dominant role of English.

As we can see from the results, each non-English language follows a slightly different (error) path.



	en	it	fr	pt	es	de	ar	nl	ru	pl	uk	sv	fa	ko
ALL-EN	98	98	95	96	94	94	95	96	92	94	70	85	96	91
EN(B)	99													
ALL-EN_S	90	96	94	94	94	94	94	96	91	92	74	87	96	90
SVO-EN_S	90	96	94	95	93	93		96	92	93	74	89		
EURO-EN_S	91	96	94	94	94	93								
EN_S(B)	91													
IT-EN_S	91	95												
FR-EN_S	91		93											
PT-EN_S	90			93										
ES-EN_S	91				92									
DE-EN_S	90					93								
AR-EN_S	91						94							
NL-EN_S	91							95						
RU-EN_S	90								87					
PL-EN_S	91									92				
UK-EN_S	91										68			
SV-EN_S	91											89		
FA-EN_S	90												94	
KO-EN_S	90													90
IT		96												
IT(B)		94												
FR			94											
PT				93										
ES					93									
IT-FR-PT-ES		96	94	94	95									
AR_PT_ES				94	92		94							
DE						92								
DE(B)						92								
AR							94							
NL								94						
RU									89					
PL										93				
UK											55			
RU-PL-UK									86	92	70			
SV												88		
SV_PL_FA										93		88	96	
FA													94	
KO														88
KO(B)														87
KO-FA													93	89

	en	it	fr	pt	es	de	ar	nl	ru	pl	uk	sv	fa	ko
ALL-EN	79	71	62	66	50	66	52	62	34	55	40	67	72	52
EN(B)	83													
ALL-EN_S	63	70	63	67	53	68	49	62	31	55	37	69	74	50
SVO-EN_S	62	68	63	65	49	66		62	31	54	37	67		
EURO-EN_S	64	69	64	65	51	66								
EN_S(B)	66													
IT-EN_S	64	70												
FR-EN_S	65		62											
PT-EN_S	65			64										
ES-EN_S	65				50									
DE-EN_S	64					66								
AR-EN_S	64						50							
NL-EN_S	63							62						
RU-EN_S	64								31					
PL-EN_S	65									54				
UK-EN_S	63										26			
SV-EN_S	63											67		
FA-EN_S	63												72	
KO-EN_S	63													55
IT		70												
IT(B)		69												
FR			65											
PT				66										
ES					56									
IT-FR-PT-ES		71	64	66	51									
AR_PT_ES				66	51		52							
DE						64								
DE(B)						64								
AR							51							
NL								64						
RU									34					
PL										57				
UK											25			
RU-PL-UK									34	57	25			
SV												70		
SV_PL_FA										56		72	70	
FA													69	
KO														49
KO(B)														59
KO-FA													70	49

Figure 6: F_1 (left: entity_1 correct, right: entity_2 correct) for all languages and models.

This does not seem to be a general rule that can be applied. We would like to point out though that this is exactly the reason why we have created this dataset in the first place. We wanted to observe the performance of models on different languages on the relation extraction task and now, thanks to our dataset, this is possible. The fact that there is no simple explanation for the difference in model performance shows that deeper analysis for each language is necessary.

Another observation is that we obtain significantly higher F_1 for entity_1 than for entity_2, which suggests that entity_1 is simpler. This seems to be true, because entity_1 typically occurs at the beginning of the sentence, while entity_2 does not have any consistent location.

In the future, we plan to train our model on the EN-full dataset and to predict multiple plausible entity pairs for the same sentence. We would also like to extend the dataset to include entity types and sentences containing multiple relations.

Another promising direction is data augmentation by "cross-matching" entities and relations in the dataset with sentences in the dataset. This cross-match could search for two cases.

1. Sentences that contain multiple relations between the same 2 entities.
2. Sentences that contain more than 2 entities (e.g. 3), with different relations between them.

In both cases, the sentence could be added to the dataset multiple times.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *EMNLP/IJCNLP*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186.

Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *The* 20^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610.

Markus Eberts and Adrian Ulges. 2019. Span-Based Joint Entity and Relation Extraction with Transformer Pre-training. In *24^{th} European Conference on Artificial Intelligence*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545.

Kiril Gashteovski, Rainer Gemulla, and Luciano del Corro. 2017. MinIE: Minimizing Facts in Open Information Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2630–2640.

John Giorgi, Xindi Wang, Nicola Sahar, Won Young Shin, Gary D. Bader, and Bo Wang. 2019. End-to-End Named Entity Recognition and Relation Extraction Using Pre-trained Language Models. arXiv:1912.13415.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a Benchmark Corpus to Support the Automatic Extraction of Drug-Related Adverse Effects from Medical Case Reports. *Journal of Biomedical Informatics*, 45(5):885 – 892.

Cheng Li and Ye Tian. 2020. Downstream Model Design of Pre-trained Language Model for Relation Extraction Task. arXiv:2004.03786.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

T. Mitchell, W. Cohen, and E. Hruschka. 2018. Never-Ending Learning. *Commun. ACM*, 61(5):103–115.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT? In *ACL*.

Paweł Przybysz, Marcin Chochowski, Rico Sennrich, Barry Haddow, and Alexandra Birch-Mayne. 2017. The Samsung and University of Edinburgh's submission to IWSLT17. In *Proceedings of the* 14^{th} *International Workshop on Spoken Language Translation*, pages 23–28.

Dan Roth and Wen-tau Yih. 2004. A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the* 57^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In 31^{st} *Conference on Neural Information Processing Systems (NIPS 2017)*.

Joanna Wetesko, Marcin Chochowski, Paweł Przybysz, Philip Williams, Roman Grundkiewicz, Rico Sennrich, Barry Haddow, Antonio Valerio Miceli Barone, and Alexandra Birch. 2019. Samsung and University of Edinburgh's System for the IWSLT 2019. In 16^{th} *International Workshop on Spoken Language Translation 2019*.

Philip Williams, Marcin Chochowski, Paweł Przybysz, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2018. Samsung and University of Edinburgh's System for the IWSLT 2018 Low Resource MT Task. In *Proceedings of the* 15^{th} *International Workshop on Spoken Language Translation*, pages 118–123.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv:1910.03771.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Zhu. 2012. Probase: A Probabilistic Taxonomy for Text Understanding. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

A New View of Multi-modal Language Analysis: Audio and Video Features as Text "Styles"

Zhongkai Sun
University of Wisconsin-Madison
`zsun227@wisc.edu`

Prathusha K Sarma[*]
Apple
`prathyushaks.21@gmail.com`

Yingyu Liang
University of Wisconsin-Madison
`yliang@cs.wisc.edu`

William A. Sethares
University of Wisconsin-Madison
`sethares@wisc.edu`

Abstract

Imposing the style of one image onto another is called *style transfer*. For example, the style of a Van Gogh painting might be imposed on a photograph to yield an interesting hybrid. This paper applies the adaptive normalization used for image style transfer to language semantics, i.e., the *style* is the way the words are said (tone of voice and facial expressions) and these are style-transferred onto the text. The goal is to learn richer representations for multi-modal utterances using style-transferred multi-modal features. The proposed Style-Transfer Transformer (STT) grafts a stepped styled adaptive layer-normalization onto a transformer network, the output from which is used in sentiment analysis and emotion recognition problems. In addition to achieving performance on par with the state-of-the art (but using less than a third of the model parameters), we examine the relative contributions of each mode when used in the downstream applications.

1 Introduction

Multi-modal language analysis expands textual analysis by utilizing co-occurring acoustic and visual information, and has recently become a popular topic in machine learning (Morency et al., 2011; Baltrušaitis et al., 2018). In both sentiment analysis (Wang et al., 2016; Zadeh et al., 2016) and emotion recognition (Busso et al., 2008; Mittal et al., 2019), the three modalities are combined to better represent the sentiment or emotional meaning of a passage. The idea of combing textual, acoustic, and visual features is obvious: individual modalities are not always able to convey as accurate an impression as multi-modal features, which typically provide more complete information. For instance, Fig. 1 shows a caricature where the sentiment may be easily understood from the audio and video, but not from textual analysis alone.

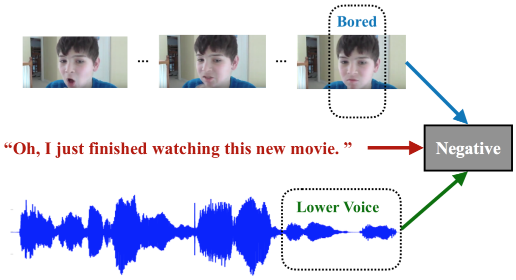


Figure 1: An example of multi-modal sentiment analysis. In this example, textual information is unrelated to the underlying sentiment while acoustic and visual features reflect the perceived sentiment.

Combining textual, acoustic, and visual features can be accomplished in a variety of ways ((Tsai et al., 2018; Zadeh et al., 2018a; Liang et al., 2018; Mai et al., 2019; Sun et al., 2019b; Mittal et al., 2019)). Among these, (Wang et al., 2018b; Sun et al., 2019a) consider text as the backbone and study methods to inject acoustic and visual information into the textual features that are typically extracted via pre-trained word/language models, e.g., Glove (Pennington et al., 2014), ELMO (Peters et al., 2018), and BERT (Devlin et al., 2018).

This paper also uses textual features as a backbone and studies a novel way of injecting non-text features into a primarily text-only model. A *style* vector is learned from the acoustic and visual features. This style vector is then transferred to a text input transformer encoder via **S**tepped **Ad**aptive **La**yer **N**ormalization (SAdLaN). While adaptive style transfer in image processing literature is well studied (Karras et al., 2018; Huang and Belongie,

---

[*] Work done while at UW-Madison

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1956–1965
April 19 - 23, 2021.

2017; Park et al., 2019), the novelty in our work is to consider the corresponding audio and video to be the *style* of the text and transfer the non-verbal features' information to text-based models via the effective SAdLaN. Concretely, our model replaces the original layer normalization in the text transformer encoder with the proposed SAdLaN, which learns style scale and style bias from non-text features. The proposed model is named as Style-Transfer Transformer (STT) and it is tested on three benchmark datasets.

This paper makes three contributions. First, our model's performance is on par with the state-of-the-art but using only less than one third of the model parameters. Since our model does not require training a multimodal transformer from scratch to achieve the same results, the style transfer method benefits from both reduced model size and training time. Second, we introduce the Stepped Adaptive Layer Normalization (SAdLaN), which performs adaptive normalization as a function of the layer of the DNN encoder. Third, we study the contributions of each modality towards use in our downstream applications. While we know that multimodal embeddings would contribute more than each individual modality, such an examination highlights the relative strength of each mode, particularly on sentiment classification tasks.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 describes our proposed method Style Transfer Transformer (STT). Section 4 and 5 presents experimental results, and Section 6 concludes this work.

2 Related Work

Multi-modal Language Analysis: Previous work on multi-modal language analysis typically learns a novel network structure to capture interactions between text, audio, and video features via supervised learning. Liu et al. (2018) develop an efficient low-rank tensor fusion mechanism to learn the outer-product representation of multi-modal features, while (Zadeh et al., 2018a) learn a multi-modal memory gate that is applied to an LSTM to capture the flow of information in the different modalities. In their work (Liang et al., 2018) learn a specialized multi-modal fusion model by applying a novel multistage fusion in the recurrent network. Recently, the multi-modal transformer introduced by (Tsai et al., 2019) achieves the state-of-the-art performance by using bi-directional cross modal relationships between the different modalities. Note that the transformer architecture in STT is the same as in (Tsai et al., 2019). However, we depart from their modeling procedure by encoding audio and video jointly using a single bimodal transformer block, thereby eliminating additional cross modal attention blocks.

Adaptive Normalization: Adaptive normalization is widely used in image processing and computer vision. Huang and Belongie (2017) proposes adaptive instance normalization, which learns the affine parameters from the style vector to perform a style transfer in the image encoder's feature space. Karras et al. (2018) applies the AdIN in the generative adversarial network (GAN) to generate fake multi-styled human faces. Park et al. (2019) proposes a spatially-adaptive normalization for image semantic synthesis. Wang et al. (2018a) applies a spatial feature transformation for image super-resolution. This paper introduces the SAdLaN framework in which a bimodal style vector is transferred as a function of the DNN's depth.

Adapter Method in Pre-trained Language Models: Due to the enormous size of pre-trained language models (e.g. BERT, XLNet), the procedure of fine-tuning the models on downstream data may be inefficient. To address this "adapter" based methods have been developed. In their work (Houlsby et al., 2019) apply the task-specific layer at each encoder layer of BERT. Each task-specific layer contains two feed-forward projectors: one down-projects (maps the input vector to a low-dimensional space) and the other up-projects (maps the prior layer's output back to its original dimension). During training, only the parameters in these task-specific layers are updated. Stickland and Murray (2019) proposes another method: the task-specific layer is applied between the two layer-normalization in the encoder. Wang et al. (2020) applies parallel adapters which each learn different information from the knowledge base in order to enable the model to deal with the multi-task learning task.

Our proposed SAdLaN is motivated by the the adapter designed in (Houlsby et al., 2019). In their work, the authors find that lower layers of the transformer have less impact on the fine tuning objective of a given task, while higher layers are more vital (this is an intuitive observation, because, lower layers are more likely to learn semantic meanings while the higher layers are more related to the specifics of the task). When using SAdLaN, lower layers of the STT are changed as little as possible to enable the model to learn basic semantic information from the input text, while the top layers allow a larger effect influenced by the style factors.

3 Style Transfer Transformer (STT)

We define the text, audio, and video features for a given utterance as $f_t \in \mathbb{R}^{l\times d_t}, f_a \in \mathbb{R}^{l\times d_a}, f_v \in \mathbb{R}^{l\times d_v}$, where l is the length of the modality sequence and d_t, d_a, d_v are embedding dimensions of each modality (each modalities' sequence length is forced to be the same by applying alignment). Figure 2 (*left*) shows an overview of our model STT. It consists of the following four steps:

- Step 1: learn a style vector from the audio and video features.
- Step 2: use the learned style vector during adaptive layer normalization for a text input transformer model.
- Step 3: take the style transferred text representation and pass it through a GRU to get the final multimodal embedding.
- Step 4: use this multimodal embedding for a downstream task.

Step 1 Learn a style vector: In keeping with the hypothesis that the text modality is the major contributor to the learned multimodal embedding, a style encoder is first applied to acoustic (f_a) and visual (f_v) features to learn the non-verbal style vector $f_s \in \mathbb{R}^{l_s}$. The STT first concatenates audio and video sequences at each time step, i.e., $f_{av} \in \mathbb{R}^{l\times(d_a+d_v)}$. The concatenated f_{av} is then input into the transformer model $\textsc{Transformer}_{av}$, which is a basic self-attention multi-head transformer encoder (with query = key = value; see (Vaswani et al., 2017)). The final state of $\textsc{Transformer}_{av}$ is the style vector f_s.

Step 2 Adaptive style transfer onto text: The original layer norm (Ba et al., 2016) is defined as:

$$f_y = \frac{f_x - \mathbb{E}[f_x]}{\sqrt{\text{Var}[f_x]+\epsilon}} \times \gamma + \beta \qquad (1)$$

where f_x is the input vector to the layer normalization, ϵ is a value added to the denominator for numerical stability, and γ, β are scalar and bias factors computed from the data for normalization. The style transfer techniques for image processing applications use factors s, b learned from a style vector to replace γ, β (Huang and Belongie, 2017; Karras et al., 2018).

Inspired by these, we propose our Stepped Adaptive Layer Normalization technique (SAdLaN). It also computes factors from our style vector f_s for the normalization, but with the key difference that it takes into account the depths of the layers. Since lower layers of the transformer have less impact on the fine tuning objective for a given task while higher layers are more vital to the task, the lower layers of the STT should be changed as little as possible to enable the model to learn basic semantic information from the input text, while the top layers should allow a larger effect influenced by the style factors.

Figure 2 (*right*) shows how the SAdLaN is applied to a text transformer encoder. Formally, we input the style vector f_s to a MLP layer to compute factors s^i, b^i for the normalization of the ith layer, and introduce a novel stepped ratio factor r_i where r_i gradually increases as a function of the depth of the layer. SAdLaN for the ith layer is then defined as:

$$\begin{aligned} f_y^i = (&\frac{f_x^i - \mathbb{E}[f_x^i]}{\sqrt{\text{Var}[f_x^i]+\epsilon}} \times \gamma + \beta) \\ &\times (1 + s^i \times r_i) + (b^i \times r_i). \end{aligned} \qquad (2)$$

Here γ and β are computed as in the original layer norm (without using the style vector). The s^i, b^i are factors learned from the style vector: at each layer i, the learned style vector f_s is input to a specific MLP to learn the factors s^i, b^i. The r_i are defined as

$$r_i = (i-1) \times \frac{\text{ratio}}{\#\text{layers}} \qquad (3)$$

where ratio is a constant value used to limit the maximum r_i value. Thus, small r_i indicate the style factors have little impact on the layer norm and vice versa. Note that SAdLaN reduces to the original layer norm (1) when $\text{ratio} = 0$.

Step 3 Style transferred multimodal embedding: The output of the last adapted layer from Step 2 is considered to be the style transferred text sequence. This styled text sequence is then passed through a GRU to learn the final multimodal embedding.

Step 4 Downstream applications: Multimodal embeddings learned in Step 3 can be passed to a softmax layer or a MLP for any downstream task of choice.

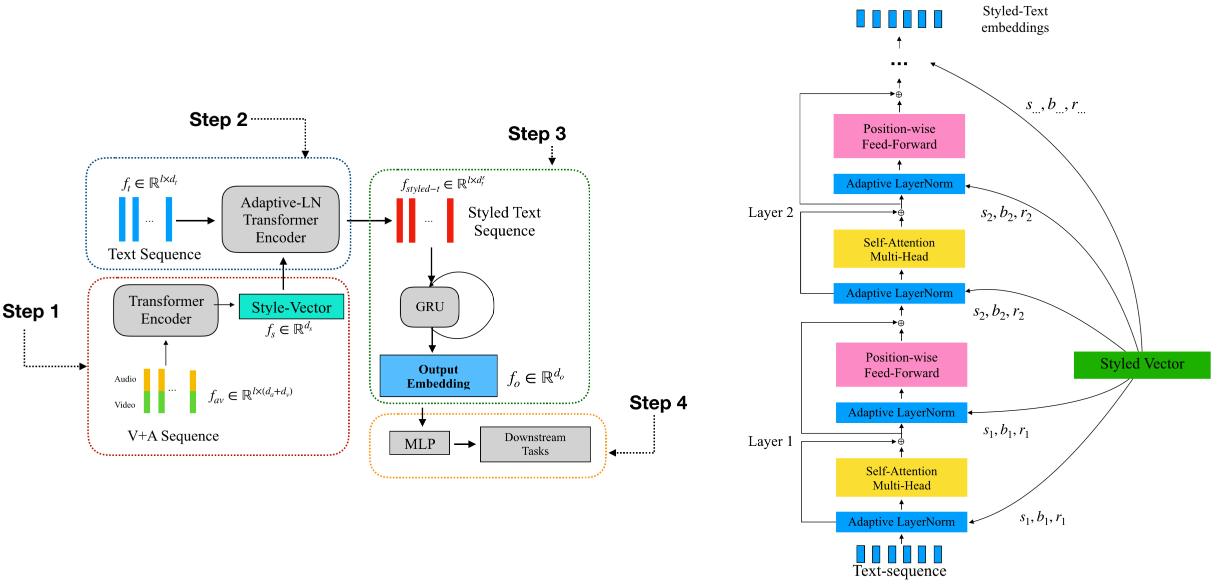


Figure 2: *left:* An overview of the STT model. First, a style vector is learned from audio and video features. Second, the style vector is used along with the text to learn a styled-text sequence by a transformer encoder with the adaptive layer normalization. Finally, use a GRU to get the final multimodal embedding and use a MLP for the downstream task. See the main text for the details. *right:* The STT Transformer encoder with the adaptive layer normalization. Scalar and bias factors are learned from the style vector. Smaller ratios are applied to the bottom layers so that accurate semantic features can be captured, while larger ratios are applied to top layers to encourage the learning of the styled features.

4 Experimental Setup

This section provides a brief overview of the experimental setup, datasets, and baseline methods used for comparison.

4.1 Data sets

In order to test the performance of the STT, three benchmark multi-modal datasets are selected: i) CMU-MOSI, ii) CMU-MOSEI (Zadeh et al., 2016, 2018b), and iii) IEMOCAP (Busso et al., 2008). The first two are standard datasets for sentiment analysis while IEMOCAP is a standard emotion recognition dataset.

- **CMU-MOSI:** This data set contains 2199 utterance-level video segments. Each video segment is labeled with sentiment scores that range from -3 (extremely negative sentiment) to $+3$ (extremely positive sentiment).

- **CMU-MOSEI:** This data set contains 22856 utterance-level video segments. Annotation of each segment is the same as in the CMU-MOSI dataset.

- **IEMOCAP:** The original dataset contains 10000 examples with 9 different emotion annotations. In this paper, we follow (Tsai et al., 2019) and choose four emotions (angry, sad, happy, and neutral) with balanced distributions for the evaluation.

4.2 Baselines

We compare the proposed STT with several baselines: i) Early and Late Fusion LSTM (Zadeh et al., 2016), ii) **RAVEN** (Wang et al., 2018b), iii) **MCTN** (Tsai et al., 2018), iv) **LMF** (Liu et al., 2018), and v) **RMFN** (Liang et al., 2018). The state of the art **Mult** (Tsai et al., 2019), which applies 6 cross-modal transformers to learn cross relationships between modalities, is also included in our evaluations.

To keep comparisons fair, our experiments use the same multimodal features as in (Liu et al., 2018; Zadeh et al., 2016, 2018a; Liang et al., 2018; Tsai et al., 2019). Here are some high level details with regards to each modalities' features:

- **Textual features:** Glove (Pennington et al., 2014) word embeddings of 300 dimensions are used as inputs to obtain textual features.

- **Acoustic features:** They are extracted by COVAREP (Degottex et al., 2014) and have 74 dimensions. These features contain information on frequency, volume, pitch, MFCC, etc.

	MOSI	MOSEI	IEMOCAP
batch size	48	128	48
learning rate	0.002	0.0001	0.003
transformer hidden dim	40	40	40
GRU hidden dim	60	40	60
# of encoder layers	8	8	4
# of heads	8	10	8
style ratio	0.3	0.2	0.25
dropout	0.2	0.5	0.3

Table 1: This table presents the best hyper-parameter settings of the STT model for all data sets reported in our experiments.

They are aligned at the word-level: for every word, its corresponding acoustic feature is the average of all audio frame's features between the start time and the end time of that specific word.

- **Visual features:** They are extracted by Facet (iMotions, 2017) and have 34 dimensions. Features include facial landmarks, action units, etc. They are also aligned at the word level.

In consistency with prior work on these three datasets, several performance metrics are used to evaluate the models. On CMU-MOSI and CMU-MOSEI, binary accuracy, weighted F1 score, mean absolute error, 7-class accuracy, and correlation with human labels are reported. For IEMOCAP, binary accuracy and weighted f1 score are used.

Hyperparameters: We perform grid search on the hyperparameter values. For STT, i) the learning rates of Adam are from 0.0001 to 0.001; ii) dropout ratios are from 0 to 0.5; iii) transformer encoder's hidden layers are of dimensions 40 and 60; iv) hidden states of GRU are of dimensions 40 and 60; v) the numbers of transformer encoder layers are from 4 to 8; vi) the numbers of heads in the multi-head attention layer are 8 and 10; vii) style ratios are from 0.1 to 0.5. The best hyper-parameter settings in our experiments are presented in table 1.

5 Experimental Results

Tables 2 and 3 report results from the baselines and STT on the three benchmark data sets. We can see that STT performs on par with the baselines and SOTA considered in our experiments. Since there exists a disparity between the results reported in (Tsai et al., 2019) and the results obtained from reproducing the source code released by the authors, we report numbers from reproduced Mult as Mult_{rep} and report numbers from the paper as $\text{Mult}_{\text{paper}}$. Note that STT beats the performance of Mult_{rep} while matching the performance of $\text{Mult}_{\text{paper}}$. To keep comparisons fair, we compare STT with the averages of scores attained by $\text{Mult}_{\text{paper}}$ and Mult_{rep} and note that on all three benchmark datasets on average we do better than Mult. While one may argue that reproducing the baseline is essential, we can offer as an explanation that the lack of clarity in reproducibility should not penalize our modeling efforts. To document our experimental evaluations and hypothesis in a credible manner, we report numbers from the paper as well as the reproduction. Furthermore, we note that STT is more efficient than Mult: it achieves better performance using less than a third number of parameters. See Table 4. This is because it uses 2 transformer blocks as opposed to 6 blocks in Mult.

To verify if the performance of STT increases with the number of parameters, we investigate increasing: 1) the dimension of the transformer encoder's hidden layer, 2) the number of encoder layers and number of heads, and 3) the number of layers in the GRU. However, we do not find improvements in performance by increasing the number of model parameters. We posit that this performance limitation is due to the relatively crude features extracted for audio and video analysis and suggest exploration in this direction for improved performance.

5.1 Ablation Studies

In this section we present results from ablation studies performed by i) varying ratio and studying the effect of different ratio values on the transformer's performance, ii) providing as input to the transformer unimodal features for downstream sentiment classification tasks, as well as iii) evaluating the effects of non verbal features.

5.1.1 Study 1: Varying Stepped Ratio

Table 5 presents results from the ablation study that shows the performance of different ratio values in equation (2). When $\text{ratio} = 0$, the model contains only a single transformer using text as the only input and standard layer normalization. This setup is meant to demonstrate the performance of STT when using text alone. "No stepped ratio" means the same style scalar and bias are used for all layers, i.e., $r_i = 1$ for all i in (2). Varying the ratio from 0.1 to 2 investigates the influence of its values.

Table 5 shows the effect of the stepped ratio on

Data View	CMU-MOSI					CMU-MOSEI				
	Acc-2	F-score	MAE	Acc-7	Corr	Acc-2	F-score	MAE	Acc-7	Corr
EF-LSTM	75.9	75.5	1.035	32.7	0.611	77.2	77.5	0.632	46.4	0.623
LF-LSTM	77.8	77.7	1.009	34.8	0.645	79.4	80.2	0.611	48.3	0.666
RAVEN	78.0	76.6	0.915	33.2	0.691	79.1	79.5	0.614	50.0	0.662
MCTN	79.3	79.1	0.909	35.6	0.676	79.8	80.6	0.609	49.6	0.670
RMFN	78.4	78.0	0.922	38.3	0.681	NA	NA	NA	NA	NA
Mult(paper)	**83.0**	**82.8**	0.871	**40.0**	0.698	**82.5**	82.3	**0.580**	**51.8**	**0.703**
Mult(rep)	81.7	81.8	0.874	38.1	0.708	82.0	81.9	0.585	50.8	0.690
STT	82.4	82.2	**0.847**	38.9	**0.733**	82.1	**82.6**	0.586	51.2	0.695

Table 2: Results on CMU-MOSI and CMU-MOSEI. Best numbers are in bold. Note that for MAE, lower is better, while for the other metrics, higher is better.

Data View	IEMOCAP							
Emotions	Happy		Angry		Sad		Neutral	
	Acc-2	F-score	Acc-2	F-score	Acc-2	F-score	Acc-2	F-score
EF-LSTM	85.5	84.8	85.7	83.1	81.2	80.3	66.3	65.3
LF-LSTM	85.4	85.7	83.6	82.9	79.9	80.1	67.1	67.2
RAVEN	87.3	85.8	85.1	84.6	83.8	82.9	69.5	69.1
MCTN	84.9	83.1	79.7	80.4	80.5	79.6	62.3	57.0
RMFN	87.5	85.8	85.1	84.6	83.8	82.9	69.5	69.1
Mult(paper)	**90.7**	**88.6**	**87.4**	87.0	86.7	86.0	**72.4**	**70.7**
Mult(rep)	88.7	86.9	87.0	**87.2**	86.6	86.3	70.6	69.4
STT	88.3	87.8	87.3	87.0	**87.5**	**87.4**	70.1	68.5

Table 3: Results on IEMOCAP. Best numbers are in bold.

Model	CMU-MOSI	CMU-MOSEI	IEMOCAP
Mult	1.54M	1.55M	1.53M
STT	**0.44M**	**0.44M**	**0.38M**

Table 4: The number of parameters in each model for the different tasks. Hyper-parameters with best performance are selected. "M" means one million.

	Acc-2	F-score	MAE	Acc-7	Corr
ratio = 0	81.1	81.0	0.871	37.4	0.718
ratio = 0.1	81.3	81.6	0.889	38.6	0.707
ratio = 0.2	82.1	**82.2**	0.853	39.2	0.713
ratio = 0.3	**82.4**	**82.2**	**0.847**	38.9	**0.733**
ratio = 0.5	81.9	81.9	0.864	37.0	0.722
ratio = 1.0	81.3	81.2	0.869	**39.5**	0.714
ratio = 2.0	81.4	81.4	0.871	38.0	0.718
no stepped ratio	79.9	79.8	0.912	37.4	0.697

Table 5: Ablation study and effect of different ratios on CMU-MOSI. Ratio values vary in different levels as defined in (2). "ratio = 0" means the original layer normalization is applied. "no stepped ratio" means that the scale and bias terms are applied equally to all layers.

the performance. Using $\mathrm{ratio} = 0$, i.e., using only text features, worsens the performance. This is intuitive since additional modalities capture information not present in text. On the other hand, a large ratio like 1 or 2 degrades the performance of STT. This is also consistent with our intuition, since large ratios correspond to an almost complete dependency on the audio and video features (which are not as deeply studied as the representation of text). Thus an intermediate value for ratio should work best, and our experiments suggest a value of about 0.3. This supports our hypothesis that one should utilize the text embedding to capture the major semantics and utilize the audio and video embeddings to capture additional crucial stylistic information injected into text in a more gentle manner.

Finally, we see that using the stepped ratio is better than setting the same stepped ratio at each layer. This is consistent with our intuition that lower layers of the transformer model corresponds to basic semantics that should remain unaffected by external information.

5.1.2 Study 2: Unimodal Input Features

To study the effects of audio or video only input to the transformer model, we set up experiments in which the STT takes as input each of audio or video modes alone. For each modality, we vary the stepped ratio from 0.1 to -0.5 and report the best numbers. Table 6 presents results from STT using unimodal text or audio input when evaluated against the CMU-MOSI data set. From Table 6, while it is hard to determine if individually audio or video makes for a better input on the sentiment classification task, it is evident that the combined bimodal (audio+video) input does better than each unimodal input.

5.1.3 Study 3: Effect of Non-verbal Features

The results in Table 6 quantitatively show the effectiveness of non-verbal features, however, it's also vital to demonstrate that the non-verbal features can help to successfully classify specific examples' sentiments. Figures 3 and 4 present two case-studies that show how the STT model is able to capture non-verbal information effectively by comparing

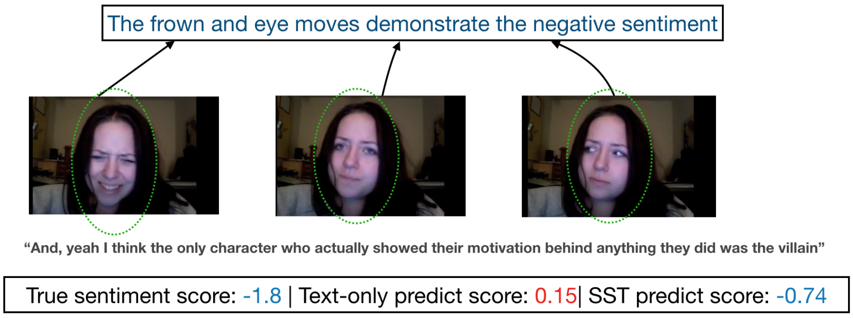


Figure 3: Case Study Example 1. The text in this utterance doesn't have a clear sentiment, however, the woman's frown and eye movements reveal her disappointment in the characters' motivations.

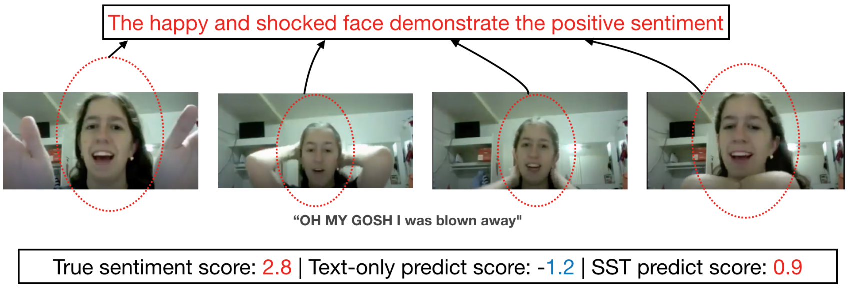


Figure 4: Case Study Example 2. The text "blown away" alone may be ambiguous. However, combined with the woman's happy facial expressions, the overall support is towards a strong positive sentiment.

	Acc-2	F-score	MAE	Acc-7	Corr
STT without audio/video	81.1	81.0	0.871	37.4	0.718
STT with audio	81.1	81.4	0.887	38.0	0.711
STT with video	81.7	81.5	0.888	37.9	0.720
STT with audio and video	**82.4**	**82.2**	**0.847**	**38.9**	**0.733**

Table 6: This table presents the effect of unimodal audio/video inputs to STT when evaluated on the CMU-MOSI data set.

the prediction of our STT model with that of a text-only model (stepped ratio $= 0$). We select two typical examples where our STT model is able to predict the correct sentiment score while the text only model is not. Figure 3 shows an example in which the text is neutral but the facial expression clearly demonstrates a negative sentiment. The STT model is able to predict the negative sentiment by using the facial features. Figure 4 shows another example with ambiguous text "blown away." Again, the text-only model does not predict correctly, but our STT is able to exploit the information in the visual features to predict the true sentiment.

5.2 Applications in Pre-trained Language Models

Our proposed method replaces the vanilla layer normalization in the transformer's layers with a stepped style adaptive layer normalization (SAdLaN). The same technique can also be applied to expand the capabilities of a pre-trained text-only transformer model. To validate this, we consider a pre-trained language model BERT and apply SAdLaN to every layer in BERT. This allows applying it on multimodal data sets, by fine-tuning the SAdLaN parameters or fine-tuning both the SAdLaN and BERT parameters. To clarify, differences in results reported in the previous sections and here lie in the transformer architecture. Since the BERT language model is pre-trained in prior work, the number of transformer layers differ in BERT and STT. The BERT language model in addition to using the transformer architecture also follows a *cloze* task in the language model. Unlike BERT, STT does not perform random masking on word tokens in the input.

It is also possible to combine styled layer normalization with the adapter method (Houlsby et al., 2019). The adapter method injects some trainable modules called adapters between the feed-forward layer and the layer-normalization in each transformer encoder layer of BERT. Each adapter contains two feed-forward projection layers, connected by a non-linear activation. The first feed-forward layer projects its input (of dimension d) to a smaller dimension, then the second layer maps the output

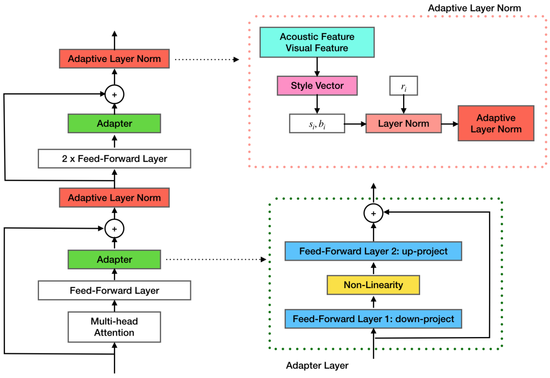


Figure 5: This figure illustrates injecting the adapter and the adaptive layer-normalization into BERT's transformer encoders. At each encoder layer of the BERT model, the adapter network is injected between the feed-forward layer and the adaptive layer normalization. During fine tuning, any part of the model can be frozen for ablation studies.

of the first layer back to dimension d. Houlsby et al. (2019) showed that injecting adapters and fine-tuning only the parameters of the adapters achieve competitive results against fine-tuning the whole BERT. We can combine SAdLaN with adapters to achieve similar results that again use fewer trainable parameters. Figure 5 illustrates the combination.

We now present experiments on these applications of SAdLaN (combined with BERT, or with BERT+adapters). The BERT model used in our experiments is the "bert-base-uncased" version in (Devlin et al., 2018). Evaluation uses the CMU-MOSI and CMU-MOSEI datasets. Raw text is used as the input textual feature instead of the pretrained Glove embeddings used in the other experiments, while acoustic and visual features remain the same.

Results: Tables 7 and 8 present the performance of different methods fine-tuning BERT on multi-modal data.

- The BERT model performs roughly the same with or without SAdLaN (see the first two rows in both tables). This is likely due to the large model parameters in BERT (110M) as opposed to the much fewer parameters in SAdLaN ($\leq$ 1M).

- Updating only the parameters of SAdLaN and the final logistic regression layer demonstrates an obvious improvement compared to using the final regression layer only (see the third to fifth rows). This confirms that our method is able to inject acoustic and visual information into the transformer model.

- Combining SAdLaN with the adapter method performs better than using the adapter method alone (see the last two rows). In this case, the performance is similar to fine-tuning the whole BERT (in the first row), while updating far fewer parameters (about only 1.5%). Thus our method improves the adapter method and enables efficient training on multimodal tasks.

	Acc-2	F-score	MAE	Corr	Size
BERT	**86.6**	**86.0**	**0.683**	**0.80**	110M
BERT + SAdLaN	85.1	85.0	0.689	**0.80**	111M
SAdLaN	**83.0**	**83.3**	**0.794**	**0.76**	1.5M
(BERT output)	80.5	81.1	0.89	0.69	0.06M
(a + v + BERT output)	81.0	81.0	0.885	0.69	0.06M
Adapter	84.1	84.3	0.72	0.77	3.1M
Adapter + SAdLaN	**85.1**	**85.2**	**0.696**	**0.79**	3.5M

Table 7: Results of fine-tuning BERT, adapter, and our method SAdLaN on CMU-MOSI. Logistic regression is used as the final classifier. The training only updates the parameters in the first column (and those in the logistic regression). (Bert output) means a simple average of all the layers hidden vector, (a + v + Bert output) means a direct concatenation of acoustic, visual, and BERT hidden vectors. Best results in each block are in bold.

	Acc-2	F-score	MAE	Corr	Size
BERT	**85.9**	**85.9**	0.533	0.76	110M
BERT + SAdLaN	85.5	85.7	**0.527**	**0.77**	112M
SAdLaN	**84.8**	**84.9**	**0.565**	**0.73**	1.7M
(BERT output)	82.8	83.0	0.582	0.68	0.07M
(a + v + BERT output)	82.7	82.8	0.583	0.67	0.07M
Adapter	**85.2**	85.1	0.543	0.751	3.0M
Adapter + SAdLaN	**85.2**	**85.3**	**0.533**	**0.77**	3.6M

Table 8: Results from fine tuning BERT, adapter, and our method SAdLaN on CMU-MOSEI. Annotations are the same as in Table 7.

6 Conclusion and Future Work

Inspired by the success of style transfer algorithms in image processing, this paper proposed the novel Style Transfer Transformer (STT) in which layer-normalization in the transformer model is replaced with a style Stepped Adaptive Layer Normalization (SAdLaN). The model is used to learn comprehensive multimodal representations for sentiment analysis and emotion recognition. Experiments on benchmark data sets established the effectiveness of the proposed method. Furthermore, ablation studies provided supports for our hypothesis of injecting audio and video to highly efficient text embeddings enhances the performance of the text embedding in multimodal tasks without the need for larger models or training data.

While our work is a first step towards learning multimodal embeddings via style transfer of non textual features onto text, as part of future work we will consider learning to inject non-verbal information into the text model in a recursive manner in order to achieve a higher model performance. The acoustic and visual features can be processed separately; besides of analyzing examples that benefit from the STT, it's also worthwhile to study examples that are negatively impacted by the method. While transformer models promise improved results, reproducibility in these models is a cause for concern since transformer models are particularly sensitive to initial conditions. As part of future work we will perform experiments to establish significance in observed results and report averaged hyper-parameters.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335.

Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. 2014. Covarep—a collaborative voice analysis repository for speech technologies. In *ICASSP*, pages 960–964. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. *arXiv preprint arXiv:1902.00751*.

Xun Huang and Serge J. Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR*, abs/1703.06868.

iMotions. 2017. Facial expression analysis. *imotions.com*.

Tero Karras, Samuli Laine, and Timo Aila. 2018. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948.

Paul Pu Liang, Ziyin Liu, Amir Zadeh, and Louis-Philippe Morency. 2018. Multimodal language analysis with recurrent multistage fusion. *arXiv preprint arXiv:1808.03920*.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. *ACL*.

Sijie Mai, Haifeng Hu, and Songlong Xing. 2019. Modality to modality translation: An adversarial representation learning and graph fusion network for multimodal fusion.

Trisha Mittal, Uttaran Bhattacharya, Rohan Chandra, Aniket Bera, and Dinesh Manocha. 2019. M3er: Multiplicative multimodal emotion recognition using facial, textual, and speech cues. *arXiv preprint arXiv:1911.05659*.

Louis-Philippe Morency, Rada Mihalcea, and Payal Doshi. 2011. Towards Multimodal Sentiment Analysis: Harvesting Opinions from The Web. In *ICMI 2011*, Alicante, Spain.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. *arXiv preprint arXiv:1902.02671*.

Zhongkai Sun, Prathusha Sarma, William Sethares, and Yingyu Liang. 2019a. Learning relationships between text, audio, and video via deep canonical correlation for multimodal language analysis.

Zhongkai Sun, Prathusha K Sarma, William Sethares, and Erik P Bucy. 2019b. Multi-modal sentiment analysis using deep canonical correlation analysis. *arXiv preprint arXiv:1907.08696*.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. *arXiv preprint arXiv:1906.00295*.

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2018. Learning factorized multimodal representations. *arXiv preprint arXiv:1806.06176*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Liwei Wang, Yin Li, and Svetlana Lazebnik. 2016. Learning deep structure-preserving image-text embeddings. In *CVPR*, pages 5005–5013.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Cuihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. 2018a. Recovering realistic texture in image super-resolution by deep spatial feature transform. *CoRR*, abs/1804.02815.

Yansen Wang, Ying Shen, Zhun Liu, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2018b. Words can shift: Dynamically adjusting word representations using nonverbal behaviors. *arXiv preprint arXiv:1811.09362*.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018a. Memory fusion network for multi-view sequential learning. In *AAAI*.

Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. 2016. Mosi: multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos. *arXiv preprint arXiv:1606.06259*.

AmirAli Bagher Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018b. Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In *ACL*, pages 2236–2246.

Multilingual and cross-lingual document classification: A meta-learning approach

Niels van der Heijden♣ **Helen Yannakoudakis**♠ **Pushkar Mishra**♢ **Ekaterina Shutova**♣
♣ILLC, University of Amsterdam, the Netherlands
♠Dept. of Informatics, King's College London, United Kingdom
♢Facebook AI, London, United Kingdom
`niels.vanderheijden@student.uva.nl`, `helen.yannakoudakis@kcl.ac.uk`,
`pushkarmishra@fb.com`, `e.shutova@uva.nl`

Abstract

The great majority of languages in the world are considered under-resourced for the successful application of deep learning methods. In this work, we propose a meta-learning approach to document classification in limited-resource setting and demonstrate its effectiveness in two different settings: few-shot, cross-lingual adaptation to previously unseen languages; and multilingual joint training when limited target-language data is available during training. We conduct a systematic comparison of several meta-learning methods, investigate multiple settings in terms of data availability and show that meta-learning thrives in settings with a heterogeneous task distribution. We propose a simple, yet effective adjustment to existing meta-learning methods which allows for better and more stable learning, and set a new state of the art on several languages while performing on-par on others, using only a small amount of labeled data.

1 Introduction

There are more than 7000 languages around the world and, of them, around 6% account for 94% of the population.[1] Even for the 6% most spoken languages, very few of them possess adequate resources for natural language research and, when they do, resources in different domains are highly imbalanced. Additionally, human language is dynamic in nature: new words and domains emerge continuously and hence no model learned in a particular time will remain valid forever.

With the aim of extending the global reach of Natural Language Processing (NLP) technology, much recent research has focused on the development of multilingual models and methods to efficiently transfer knowledge across languages. Among these advances are multilingual word vectors which aim to give word-translation pairs a similar encoding in some embedding space (Mikolov et al., 2013a; Lample et al., 2017). There has also been a lot of work on multilingual sentence and word encoders that either explicitly utilizes corpora of bi-texts (Artetxe and Schwenk, 2019; Lample and Conneau, 2019) or jointly trains language models for many languages in one encoder (Devlin et al., 2018; Conneau et al., 2019). Although great progress has been made in cross-lingual transfer learning, these methods either do not close the gap with performance in a single high-resource language (Artetxe and Schwenk, 2019; Conneau et al., 2019), e.g., because of cultural differences in languages which are not accounted for, or are impractically expensive (Lai et al., 2019).

Meta-learning, or *learning to learn* (Schmidhuber, 1987; Bengio et al., 1990; Thrun and Pratt, 1998), is a learning paradigm which focuses on the quick adaption of a learner to new tasks. The idea is that by training a learner to adapt quickly and from a few examples on a diverse set of training tasks, the learner can also generalize to unseen tasks at test time. Meta-learning has recently emerged as a promising technique for few-shot learning for a wide array of tasks (Finn et al., 2017; Koch et al., 2015; Ravi and Larochelle, 2017) including NLP (Dou et al., 2019; Gu et al., 2018). To our best knowledge, no previous work has been done in investigating meta-learning as a framework for multilingual and cross-lingual few-shot learning. We propose such a framework and demonstrate its effectiveness in document classification tasks. The only current study on meta-learning for cross-lingual few-shot learning is the one by (Nooralahzadeh et al., 2020), focusing on natural language inference and multilingual question answering. In their work, the authors focus on applying meta-learning to learn to adapt a monolingually trained classi-

---

[1] https://www.ethnologue.com/statistics

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1966–1976
April 19 - 23, 2021.

Algorithm 1 Meta-training procedure.

```
Require: $p(\mathcal{D})$: distribution over tasks.
Require: $\alpha, \beta$: step size hyper-parameters
  Initialize $\theta$
  while not done do
    Sample batch of tasks $\{D^l\} = \{(S^l, Q^l)\} \sim p(\mathcal{D})$
    for all $(S^l, Q^l)$ do
      Initialize $\theta_l^{(0)} = \theta$
      for all steps k do
        Compute: $\theta_l^{(k+1)} = \theta_l^{(k)} - \alpha(\nabla_{\theta_l^{(k)}} \mathcal{L}_{S_l}(f_{\theta_l^{(k)}}))$
      end for
    end for
    Update $\theta = \theta - \beta(\text{MetaUpdate}(f_{\theta_l^{(K)}}, Q^l))$
  end while
```

fier to new languages. In contrast to this work, we instead show that, in many cases, it is more favourable to not initialize the meta-learning process from a monolingually trained classifier, but rather reserve its respective training data for meta-learning instead.

Our contributions are as follows: 1) We propose a meta-learning approach to few-shot cross-lingual and multilingual adaptation and demonstrate its effectiveness on document classification tasks over traditional supervised learning; 2) We provide an extensive comparison of meta-learning methods on multilingual and cross-lingual few-shot learning and release our code to facilitate further research in the field;[2] 3) We analyse the effectiveness of meta-learning under a number of different parameter initializations and multiple settings in terms of data availability, and show that meta-learning can effectively learn from few examples and diverse data distributions; 4) We introduce a simple yet effective modification to existing methods and empirically show that it stabilizes training and converges faster to better local optima; 5) We set a new state of the art on several languages and achieve on-par results on others using only a small amount of data.

2 Meta-learning methods

Meta-learning, or *learning to learn*, aims to create models that can learn new skills or adapt to new tasks rapidly from few training examples. Unlike traditional machine learning, datasets for either training or testing, which are referred to as *meta-train* and *meta-test* datasets, comprise of many tasks sampled from a distribution of tasks $p(\mathcal{D})$ rather than individual data points. Each task is associated with a dataset $\mathcal{D}$ which contains both feature vectors and ground truth labels and is split into a *support set* and a *query set*, $\mathcal{D} = \{S, Q\}$. The support set is used for fast adaptation and the query set is used to evaluate performance and compute a loss with respect to model parameter initialization. Generally, some model f_θ parameterized by θ, often referred to as the *base-learner*, is considered. A cycle of fast-adaptation on a support-set followed by updating the parameter initialization of the base-learner based on the loss on the query-set is called an *episode*. In the case of classification, the optimal parameters maximize the probability of the true labels across multiple batches $Q \subset \mathcal{D}$

$$\theta^* := \underset{\theta}{argmax} \mathbb{E}_{Q \subset \mathcal{D}}[\sum_{(x,y) \in Q} P_\theta(y|x)] \quad (1)$$

In few-shot classification/fast learning, the goal is to minimize the prediction error on data samples with unknown labels given a small support set for learning. *Meta-training* (Algorithm 1) consists of updating the parameters of the base-learner by performing many of the formerly described *episodes*, until some stop criterion is reached.

Following this procedure, the extended definition of optimal parameters is given in Eq. 2 to include fast adaptation based on the support set. The underlined parts mark the difference between traditional supervised-learning and meta-learning. The optimal parameters θ^* are obtained by solving

$$\underset{\theta}{argmax} \underline{\mathbb{E}_{l \subset L}[} \mathbb{E}_{\underline{S^l \subset \mathcal{D}}, Q^l \subset \mathcal{D}}[\textstyle\sum_{(x,y) \in Q^l} P_\theta(y|x, \underline{S^l})]\underline{]} \quad (2)$$

In this work, we focus on metric- and optimization-based meta-learning algorithms. In the following sections, their respective characteristics and the update methods in Algorithm 1 are introduced.

2.1 Prototypical Networks

Prototypical Networks (Snell et al., 2017) belong to the metric-based family of meta-learning algorithms. Typically they consist of an embedding network f_θ and a distance function $d(x_1, x_2)$ such as Euclidean distance. The embedding network is used to encode all samples in the support set S_c and compute *prototypes* μ_c per class $c \in C$ by computing the mean of the sample encodings of that respective class

$$\mu_c := \frac{1}{|S_c|} \sum_{(x_i, y_i) \in S_c} f_\theta(x_i) \quad (3)$$

---

[2] `https://github.com/mrvoh/meta_learning_multilingual_doc_classification`

Using the computed prototypes, Prototypical Networks classify a new sample as

$$p(y=c|x) = \frac{exp(-d(f_\theta(x),\mu_c)}{\sum_{c' \in C} exp(-d(f_\theta(x),\mu_{c'})} \quad (4)$$

Wang et al. (2019) show that despite their simplicity, Prototypical Networks can perform on par or better than other state-of-the-art meta-learning methods when all sample encodings are centered around the overall mean of all classes and consecutively L2-normalized. We also adopt this strategy.

2.2 MAML

Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) is an optimization-based method that uses the following objective function

$$\theta^* := \underset{\theta}{argmin} \sum_{D_l \sim p(D)} \mathcal{L}_l(f_{\theta_l^{(k)}}) \quad (5)$$

$\mathcal{L}_l(f_{\theta_l^{(k)}})$ is the loss on the query set after updating the base-learner for k steps on the support set. Hence, MAML directly optimizes the base-learner such that fast-adaptation of θ, often referred to as *inner-loop optimization*, results in task-specific parameters $\theta_l^{(k)}$ which generalize well on the task. Setting B as the batch size, MAML implements its MetaUpdate, which is also referred to as *outer-loop optimization*, as

$$\theta = \theta - \beta \frac{1}{B} \sum_{D_l \sim p(\mathcal{D})} (\nabla_\theta \mathcal{L}_l(f_{\theta_l^{(k)}})) \quad (6)$$

Such a MetaUpdate requires computing second order derivatives and, in turn, holding $\theta_l^{(j)} \forall j = 1, \ldots, k$ in memory. A first-order approximation of MAML (foMAML), which ignores second order derivatives, can be used to bypass this problem:

$$\theta = \theta - \beta \frac{1}{B} \sum_{D_l \sim p(\mathcal{D})} (\nabla_{\theta_l^{(k)}} \mathcal{L}_l(f_{\theta_l^{(k)}})) \quad (7)$$

Following previous work (Antoniou et al., 2018), we also adopt the following improvements in our framework for all MAML-based methods:

Per-step Layer Normalization weights Layer normalization weights and biases are not updated in the inner-loop. Sharing one set of weights and biases across inner-loop steps implicitly assumes that the feature distribution between layers stays the same at every step of the inner optimization.

Per-layer per-step learnable inner-loop learning rate Instead of using a shared learning rate for all parameters, the authors propose to initialize a learning rate per layer and per step and jointly learn their values in the MetaUpdate steps.

Cosine annealing of outer-loop learning rate It has shown to be crucial to model performance to anneal the learning rate using some annealing function (Loshchilov and Hutter, 2016).

2.3 Reptile

Reptile (Nichol et al., 2018) is a first-order optimization-based meta-learning algorithm which is designed to move the weights towards a manifold of the weighted averages of task-specific parameters $\theta_l^{(k)}$:

$$\theta = \theta - \beta \frac{1}{B} \sum_{D^l \sim p(\mathcal{D})} (\theta_l^{(k)} - \theta) \quad (8)$$

Despite its simplicity, it has shown competitive or superior performance against MAML, e.g., on Natural Language Understanding (Dou et al., 2019).

2.4 ProtoMAML

Triantafillou et al. (2020) introduce ProtoMAML as a meta-learning method which combines the complementary strengths of Prototypical Networks and MAML by leveraging the inductive bias of the use of prototypes instead of random initialization of the final linear layer of the network. Snell et al. (2017) show that Prototypical Networks are equivalent to a linear model when Euclidean distance is used. Using the definition of prototypes μ_c as per Eq. 3, the weights w_c and bias b_c corresponding to class c can be computed as follows

$$\mathbf{w}_c := 2\mu_c \qquad b_c := -\mu_c^T \mu_c \quad (9)$$

ProtoMAML is defined as the adaptation of MAML where the final linear layer is parameterized as per Eq. 9 at the start of each episode using the support set. Due to this initialization, it allows modeling a varying number of classes per episode.

ProtoMAMLn Inspired by Wang et al. (2019), we propose a simple, yet effective adaptation to ProtoMAML by applying L_2 normalization to the prototypes themselves, referred to as ProtoMAMLn, and, again, use a first-order approximation (foProtoMAMLn). We demonstrate that doing so leads to a more stable, faster and effective learning algorithm at only constant extra computational cost ($\mathcal{O}(1)$).

We hypothesize the normalization to be particularly beneficial in case of a relatively high-dimensional final feature space – in case of BERT-like models typically 768 dimensions. Let x be a sample and $\hat{x} = f_\theta(x)$ be the encoding of the sample in the final feature space. Since the final activation function is the *tanh* activation, all entries of both $\hat{x}$ and μ_c have values between -1 and 1. The pre-softmax activation for class c is computed as $\hat{x}^T\mu_c$. Due to the size of the vectors and the scale of their respective entries, this in-product can yield a wide range of values, which in turn results in relatively high loss values, making the inner-loop optimization unstable.

3 Related work

3.1 Multilingual NLP

Just as the deep learning era for monolingual NLP started with the invention of dense, low-dimensional vector representations for words (Mikolov et al., 2013b) so did cross-lingual NLP with works like those of Mikolov et al. (2013a); Faruqui et al. (2014). More recently, multilingual and/or cross-lingual NLP is approached by training one shared encoder for multiple languages at once, either by explicitly aligning representations with the use of parallel corpora (Artetxe and Schwenk, 2019; Lample and Conneau, 2019) or by jointly training on some monolingual language model objective, such as the Masked Language Model (MLM) (Devlin et al., 2018), in multiple languages (Devlin et al., 2018; Conneau et al., 2019).

The formerly described language models aim to create a shared embedding space for multiple languages with the hope that fine-tuning in one language does not degrade performance in others. Lai et al. (2019) argue that just aligning languages is not sufficient to generalize performance to new languages due to the phenomenon they describe as *domain drift*. Domain drift accounts for all differences for the same tasks in different languages which cannot be captured by a perfect translation system, such as differences in culture. They instead propose a multi-step approach which utilizes a multilingual teacher trained with Unsupervised Data Augmentation (UDA) (Xie et al., 2019) to create labels for a student model that is pretrained on large amounts of unlabeled data in the target language and domain using the MLM objective. With their method, the authors obtain state-of-the-art results on the MLDoc document classification task (Schwenk and Li, 2018) and the Amazon Sentiment Polarity Review task (Prettenhofer and Stein, 2010). A downside, however, is the high computational cost involved. For every language and domain combination: 1) a machine translation system has to be inferred on a large amount of unlabeled samples; 2) the UDA method needs to be applied to obtain a teacher model to generate pseudo-labels on the unlabeled in-domain data; 3) a language model must be finetuned, which involves forwards and backwards computation of a softmax function over a large output space (e.g., 50k tokens for mBERT and 250k tokens for XLM-RoBERTa). The final classifier is then obtained by 4) training the finetuned language model on the pseudo-labels generated by the teacher.

3.2 Meta-learning in NLP

Monolingual Bansal et al. (2019) apply meta-learning to a wide range of NLP tasks within a monolingual setting and show superior performance for parameter initialization over self-supervised pretraining and multi-task learning. Their method is an adaptation of MAML where a combination of a text-encoder, BERT (Devlin et al., 2018), is coupled with a parameter generator that learns to generate task-dependent initializations of the classification head such that meta-learning can be performed across tasks with disjoint label spaces. Obamuyide and Vlachos (2019b) apply meta-learning on the task of relation extraction; Obamuyide and Vlachos (2019a) apply lifelong meta-learning for relation extraction; Chen et al. (2019) apply meta-learning for few-shot learning on missing link prediction in knowledge graphs.

Multilingual Gu et al. (2018) apply meta-learning to Neural Machine Translation (NMT) and show its advantage over strong baselines such as cross-lingual transfer learning. By viewing each language pair as a task, the authors apply MAML to obtain competitive NMT systems with as little as 600 parallel sentences. To our best knowledge, the only application of meta-learning for cross-lingual few-shot learning is the one by Nooralahzadeh et al. (2020). The authors study the application of X-MAML, a MAML-based variant, to cross-lingual Natural Language Inference (XNLI) (Conneau et al., 2018) and Multilingual Question Answering (MLQA) (Lewis et al., 2019) in both a cross-domain and cross-language setting. X-MAML works by pretraining some model M on a high-resource task h to obtain initial model parameters θ_{mono}. Consecutively, a set L of one or more auxiliary languages is taken, and MAML is applied to achieve fast adaptation of θ_{mono} for $l \in L$. In their experiments, the authors use either one or two auxiliary languages and evaluate their method in both a zero- and few-shot setting. It should be noted that, in the few-shot setting, the full development set (2.5k instances) is used to finetune the model, which is not in line with other work on few-shot learning, such as (Bansal et al., 2019). Also, there is a discrepancy in the training set used for the baselines and their proposed method. All reported baselines are either zero-shot evaluations of θ_{mono} or of θ_{mono} finetuned on the development set of the target language, whereas their proposed method additionally uses the development set in either one or two auxiliary languages during meta-training.

4 Data

In this section, we give an overview of the datasets we use and the respective classification tasks.

MLDoc Schwenk and Li (2018) published an improved version of the Reuters Corpus Volume 2 (Lewis et al., 2004) with balanced class priors for all languages. MLDoc consists of news stories in 8 languages: English, Spanish, French, Italian, Russian, Japanese and Chinese. Each news story is manually classified into one of four groups: *Corporate/Industrial, Economics, Government/Social* and *Markets*. The train datasets contain 10k samples whereas the test sets contain 4k samples.

Amazon Sentiment Polarity Another widely used dataset for cross-lingual text classification is the Amazon Sentiment Analysis dataset (Prettenhofer and Stein, 2010). The dataset is a collection of product reviews in English, French, German and Japanese in three categories: *books dvds* and *music*. Each sample consists of the original review accompanied by meta-data such as the rating of the reviewed product expressed as an integer on a scale from one to five. In this work, we consider the sentiment polarity task where we distinguish between positive (rating > 3) and negative (rating < 3) reviews. When all product categories are concatenated, the dataset consists of 6K samples per language per dataset (train, test). We extend this with Chinese product reviews in the cosmetics domain from JD.com (Zhang et al., 2015), a large e-commerce website in China. The train and test sets contain 2k and 20k samples respectively.

5 Experiments

We use XLM-RoBERTa (Conneau et al., 2019), a strong multilingual model, as the base-learner in all models. We quantify the strengths and weaknesses of meta-learning as opposed to traditional supervised learning in both a cross- and a multilingual joint-training setting with limited resources.

Cross-lingual adaptation Here, the available data is split into multiple subsets: the auxiliary languages l_{aux} which are used in meta-training, the validation language l_{dev} which is used to monitor performance, and the target languages l_{tgt} which are kept *unseen* until meta-testing. Two scenarios in terms of amounts of available data are considered. A small sample of the available training data of l_{aux} is taken to create a limited-resource setting, whereas all available training data of l_{aux} is used in a high-resource setting. The chosen training data per language is split evenly and stratified over two disjoint sets from which the meta-training support and query samples are sampled, respectively. For meta-testing, one batch (16 samples) is taken from the training data of each target language as support set, while we test on the whole test set per target language (i.e., the query set).

Multilingual joint training We also investigate meta-learning as an approach to multilingual joint-training in the same limited-resource setting as previously described for the cross-lingual experiments. The difference is that instead of learning to generalize to $l_{tgt} \neq l_{aux}$ from few examples, here $l_{tgt} = l_{aux}$. If we can show that one can learn many similar tasks across languages from few examples per language, using a total number of examples in the same order of magnitude as in "traditional" supervised learning for training a monolingual classifier, this might be an incentive to change data collection processes in practice.

For both experimental settings above, we examine the influence of additionally using all training data from a high-resource language l_{src} during meta-training, English.

MetaUpdate Method	Num inner-loop steps	Inner-loop lr	Class-head lr multiplier	Inner-optimizer lr
Reptile	2,3,5	1e-5, 5e-5, 1e-4	1, 10	-
foMAML	2,3,5	1e-5, 1e-4, 1e-3	1, 10	3e-5, 6e-5, 1e-4
foProtoMAMLn	2,3,5	1e-5, 1e-4, 1e-3	1, 10	3e-5, 6e-5, 1e-4

Table 1: Search range per hyper-parameter. We consider the number of update steps in the inner-loop, *Num inner-loop steps*, the (initial) learning rate of the inner-loop, *Inner-loop lr*, the factor by which the learning rate of the classification head is multiplied, *Class-head lr multiplier*, and, if applicable, the learning rate with which the inner-loop optimizer is updated, *Inner-optimizer lr*. The chosen value is underlined.

l_{src} = en	Method	Limited-resource setting							High-resource setting						
		de	fr	it	ja	ru	zh	Δ	de	fr	it	ja	ru	zh	Δ
Excluded	Non-episodic	82.0	**86.7**	68.3	71.9	70.9	81.0	76.8	95.3	90.9	80.9	82.9	74.5	**89.6**	85.7
	ProtoNet	90.5	85.0	76.6	75.0	69.6	82.0	79.8	95.5	91.7	82.0	82.2	76.6	87.4	85.9
	foMAML	89.7	85.5	74.1	74.1	**74.0**	83.2	80.1	95.0	91.4	81.4	82.7	76.9	87.8	86.1
	foProtoMAMLn	**90.6**	86.2	**77.8**	**75.6**	73.6	**83.8**	**80.7**	**95.6**	**92.1**	**82.6**	**83.1**	**77.9**	88.9	**86.7**
	Reptile	87.9	81.8	72.7	74.4	73.9	80.9	78.6	95.0	90.1	81.1	82.7	72.5	88.7	85.0
Included	Zero-shot	92.4	**92.1**	80.3	81.0	71.7	89.1	84.4	92.4	92.1	80.3	81.0	71.7	89.1	84.4
	Non-episodic	93.7	91.3	**81.5**	80.6	71.1	88.4	84.4	93.7	92.9	82.4	82.3	72.1	90.1	85.6
	ProtoNet	93.4	91.9	79.1	81.3	72.2	87.8	84.5	95.0	91.7	81.1	82.7	72.0	88.0	85.9
	foMAML	**95.1**	91.2	79.5	79.6	73.3	89.7	84.6	94.8	93.2	79.9	82.4	75.7	**90.6**	86.1
	foProtoMAMLn	94.9	91.7	**81.5**	**81.4**	**75.2**	**89.9**	**85.5**	**95.8**	**94.1**	**82.7**	**83.0**	**81.2**	90.4	**87.9**
	Reptile	92.3	91.4	79.7	79.5	71.8	88.1	83.8	94.8	91.0	80.2	82.0	72.7	89.9	85.1

Table 2: Average accuracy of 5 different seeds on the unseen target languages for MLDoc. Δ corresponds to the average accuracy across test languages.

5.1 Specifics per dataset

MLDoc As MLDoc has sufficient languages, we set l_{src} = English and l_{dev} = Spanish. The remaining languages are split in two groups: $l_{aux} = \{$German, Italian, Japanese$\}$; and $l_{tgt} = \{$French, Russian, Chinese$\}$. In the limited-resource setting, we randomly sample 64 samples per language in l_{aux} for training. Apart from comparing low- and high-resource settings, we also quantify the influence of augmenting the training set l_{aux} with a high-resource source language l_{src}, English.

Amazon Sentiment Polarity The fact that the Amazon dataset (augmented with Chinese) comprises of only five languages has some implications for our experimental design. In the cross-lingual experiments, where l_{aux}, l_{dev} and l_{tgt} should be disjoint, only three languages, including English, remain for meta-training. As we consider two languages too little data for meta-training, we do not experiment with leaving out the English data. Hence, for meta-training, the data consists of l_{src} = English, as well as two languages in l_{aux}. We always keep one language unseen until meta-testing, and alter l_{aux} such that we can meta-test on every language. We set l_{dev} = French in all cases except when French is used as the target language; then, l_{dev} = Chinese. In the limited-resource setting, a total of 128 samples per language in l_{aux} is used.

For the multilingual joint-training experiments there are enough languages available to quantify the influence of English during meta-training. When English is excluded, it is used for meta-validation. When included, we average results over two sets of experiments: one where l_{dev} = French and one where l_{dev} = Chinese.

5.2 Baselines

We introduce baselines trained in a standard supervised, non-episodic fashion. Again, we use XLM-RoBERTa-base as the base-learner in all models.

Zero-shot This baseline assumes sufficient training data for the task to be available in one language l_{src} (English). The base-learner is trained in a non-episodic manner using mini-batch gradient descent with cross-entropy loss. Performance is monitored during training on a held-out validation set in l_{src}, the model with the lowest loss is selected, and then evaluated on the same task in the target languages.

Non-episodic The second baseline aims to quantify the exact impact of learning a model through the meta-learning paradigm versus standard super-

Method	Limited-resource setting					High-resource setting				
	de	**fr**	**ja**	**zh**	Δ	**de**	**fr**	**ja**	**zh**	Δ
Zero-shot	**91.2**	90.7	87.0	84.6	88.4	91.2	90.7	87.0	84.6	88.4
Non-episodic	90.9	90.6	86.1	86.9	88.6	91.6	91.0	85.5	87.9	89.0
ProtoNet	89.7	90.2	86.6	85.2	87.9	90.7	92.0	86.7	84.0	88.4
foMAML	88.3	90.5	86.8	88.1	88.4	91.4	92.5	88.0	**90.4**	90.6
foProtoMAMLn	89.0	**91.1**	**87.3**	**88.8**	**89.1**	**92.0**	**93.1**	**88.6**	89.8	**90.9**
Reptile	88.1	87.9	86.8	87.5	87.6	90.6	91.7	87.3	86.2	89.0

Table 3: Average accuracy of 5 different seeds on the unseen target languages for Amazon. Δ corresponds to the average accuracy across test languages.

vised learning. The model learns from exactly the same data as the meta-learning algorithms, but in a non-episodic manner: i.e., merging support and query sets in l_{aux} (and l_{src} when included) and training using mini-batch gradient descent with cross-entropy loss. During testing, the trained model is independently finetuned for 5 steps on the support set (one mini-batch) of each target language l_{tgt}.

5.3 Training setup and hyper-parameters

We use the Ranger optimizer, an adapted version of Adam (Kingma and Ba, 2014) with improved stability at the beginning of training – by accounting for the variance in adaptive learning rates (Liu et al., 2019) – and improved robustness and convergence speed (Zhang et al., 2019; Yong et al., 2020). We use a batch size of 16 and a learning rate of 3e-5 to which we apply cosine annealing. For meta-training, we perform 100 epochs of 100 episodes and perform evaluation with 5 different seeds on the meta-validation set after each epoch. One epoch consists of 100 update steps where each update step consists of a batch of 4 episodes. Early-stopping with a patience of 3 epochs is performed to avoid overfitting. For the non-episodic baselines, we train for 10 epochs on the auxiliary languages while validating after each epoch. All models are created using the PyTorch library (Paszke et al., 2017) and trained on a single 24Gb NVIDIA Titan RTX GPU.

We perform grid search on MLDoc in order to determine optimal hyperparameters for the MetaUpdate methods. The hyper-parameters resulting in the lowest loss on l_{dev} = Spanish are used in all experiments. The number of update steps in the inner-loop is 5; the (initial) learning rate of the inner-loop is 1e-5 for MAML and ProtoMAML and 5e-5 for Reptile; the factor by which the learning rate of the classification head is multiplied is 10 for MAML and ProtoMAML and 1 for Reptile; when applicable, the learning rate with which the inner-loop optimizer is updated is 6e-5. See Table 1 for the considered grid.

6 Results

Cross-lingual adaptation Tables 2 and 3 show the accuracy scores on the target languages on MLDoc and Amazon respectively. We start by noting the strong multilingual capabilities of XLM-RoBERTa as our base-learner: Adding the full training datasets in three extra languages (i.e., comparing the zero-shot with the non-episodic baseline in the high-resource, 'Included' setting) results in a mere 1.2% points increase in accuracy on average for MLDoc and 0.6% points for Amazon. Although the zero-shot[3] and non-episodic baselines are strong, in the majority of cases, a meta-learning approach improves performance. This holds especially for our version of ProtoMAML (ProtoMAMLn), which achieves the highest average accuracy in all considered settings.

The substantial improvements for Russian on MLDoc and Chinese on Amazon indicate that meta-learning is most advantageous when the considered task distribution is somewhat heterogeneous or, in other words, when *domain drift* (Lai et al., 2019) is present. For the Chinese data used for the sentiment polarity task, the presence of domain drift is obvious as the data is collected from a different website and concerns different products than the other languages. For Russian in the MLDoc dataset, it holds that the non-episodic baseline has the smallest gain in performance when adding English data (l_{src}) in the limited-resource setting (0.2% absolute gain as

---

[3] The zero-shot baseline is only applicable in the 'Included' setting, as the English data is not available under 'Excluded'.

l_{src} = en	Method	Amazon					MLDoc						
		de	fr	ja	zh	Δ	de	fr	it	ja	ru	zh	Δ
Excluded	Non-episodic	88.4	88.6	85.7	88.2	87.7	92.8	89.1	81.2	83.2	**84.0**	87.4	86.3
	ProtoNet	86.7	88.0	86.2	87.3	87.1	89.7	87.6	80.5	82.2	80.6	85.2	84.3
	foMAML	88.3	87.5	84.6	**89.1**	86.3	94.1	**89.7**	**81.5**	84.2	77.6	87.5	85.8
	foProtoMAMLn	**88.9**	**89.5**	**86.5**	89.0	**88.5**	**94.8**	89.5	**81.5**	**84.8**	81.0	**88.7**	**86.6**
	Reptile	86.1	86.3	82.9	87.0	85.6	92.4	88.2	80.5	82.5	79.5	87.8	85.3
Included	Non-episodic	**91.0**	91.0	87.3	89.4	89.8	94.9	92.1	84.7	84.8	83.7	**91.4**	88.6
	ProtoNet	90.3	91.3	87.5	88.7	89.5	95.5	91.7	83.4	85.1	82.8	88.3	87.8
	foMAML	90.1	90.7	87.2	89.5	89.4	95.1	92.5	83.1	84.9	84.3	90.6	88.4
	foProtoMAMLn	90.7	**91.5**	**88.0**	**90.4**	**90.2**	**96.0**	**93.6**	**85.0**	**85.7**	**84.8**	90.8	**89.3**
	Reptile	90.0	89.5	86.5	87.6	88.4	94.4	93.1	83.8	85.2	83.6	90.4	88.4

Table 4: Average accuracy of 5 different seeds on the target languages in the joint-training setting for MLDoc and Amazon. Δ corresponds to the average accuracy across test languages.

opposed to 5.7% on average for the remaining languages) and even a decrease of 2.4% points when adding English data in the high-resource setting. Especially for these languages with domain drift, our version of ProtoMAML (foProtoMAMLn) outperforms the non-episodic baselines with a relatively large margin. For instance, in Table 2 in the high-resource setting with English included during training, foProtoMAMLn improves over the non-episodic baseline with 9.1% points whereas the average gain over the remaining languages is 0.9% points. A similar trend can be seen in Table 3 where, in the limited-resource setting, foProtoMAMLn outperforms the non-episodic baseline with 1.9% points on Chinese, with comparatively smaller gains on average for the remaining languages.

Joint training In this setting, we achieve a new state of the art on MLDoc for German, Italian, Japanese and Russian using our method, foProtoMAMLn (Table 4).[4] The previous state of the art for German and Russian is held by Lai et al. (2019) (95.73% and 84.65% respectively). For Japanese and Italian, it is held by Eisenschlos et al. (2019) (80.55% and 80.12% respectively). The state of the art for French and Chinese is also held by Lai et al. (2019) (96.05% and 93.32% respectively). On the Amazon dataset, foProtoMAMLn also outperforms all other methods on average. The state of the art is held by (2019) with 93.3%, 94.2% and 90.6% for French, German and Chinese respectively and, although we do not outperform it, the differences are rather small – between 0.2% (Chinese) and 3.4% points (German) – even when grid search is based on MLDoc, while we use a much less computationally expensive approach.

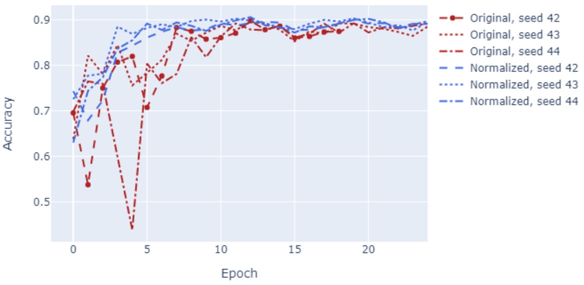


Figure 1: Validation accuracy for 3 seeds for original foProtoMAML and our new method, foProtoMAMLn.

Again, we use Russian in MLDoc to exemplify the difference between meta-learning and standard supervised learning. When comparing the difference in performance between excluding and including English meta-training episodes (l_{src}), opposite trends are noticeable: for standard supervised, non-episodic learning, performance drops slightly by 0.3%, whereas all meta-learning algorithms gain between 2.2% and 6.7% in absolute accuracy. This confirms our earlier finding that meta-learning benefits from, and usefully exploits heterogeneity in data distributions; in contrast, this harms performance in the standard supervised-learning case.

7 Ablations

foProtoMAMLn Figure 1 shows the development of the validation accuracy during training for 25 epochs for the original foProtoMAML and our model, foProtoMAMLn. By applying L_2 normalization to the prototypes, we obtain a more stable version of foProtoMAML which empirically converges faster. We furthermore re-run the high-

---

[4] The zero-shot baselines are the same as in Tables 2 and 3.

Dataset	de	fr	it	ja	ru	zh	Diff
Amazon	90.4	90.9	-	87.3	-	88.3	-1.7
MLDoc	92.8	92.4	78.6	79.3	69.3	88.9	-4.3

Table 5: Average accuracy of 5 different seeds on unseen target languages using the original/unnormalized foProtoMAML model. *Diff* is the difference in average accuracy Δ across languages against foProtoMAMLn.

Method	Limited-resource setting					High-resource setting				
	de	fr	ja	zh	Diff	de	fr	ja	zh	Diff
ProtoNet	91.1	90.9	87.1	85.5	+0.75	91.3	91.1	87.4	88.7	+1.44
foMAML	90.8	87.4	87.3	85.2	-0.75	91.7	91.2	87.2	88.1	-1.13
foProtoMAMLn	87.7	87.8	83.9	84.4	-3.1	90.8	89.8	86.2	82.3	-3.96
Reptile	89.3	90.2	86.7	85.5	+0.35	90.0	89.3	87.1	85.7	-1.04

Table 6: Average accuracy of 5 different seeds on unseen target languages for Amazon when initializing from monolingual classifier in l_{src}. *Diff*: difference in average accuracy Δ across languages compared to initializing from the XLM-RoBERTa language model.

resource experiments with English for both MLDoc and Amazon using the original foProtoMAML (Table 5) and find it performs 4.3% and 1.7% accuracy points worse on average, respectively, further demonstrating the effectiveness of our approach.

Initializing from a monolingual classifier In our experiments, we often assume the presence of a source language (English). We now investigate (in the l_{src} = en 'Excluded' setting) whether it is beneficial to pre-train the base-learner in a standard supervised way on this source language and use the obtained checkpoint θ_{mono} as an initialization for meta-training (Table 6) rather than initializing from the transformer checkpoint.

We observe that only ProtoNet consistently improves performance, whereas foProtoMAMLn suffers the most with a decrease of 3.1% and 3.96% in accuracy in the low- and high-resource setting respectively. We surmise this difference is attributable to two factors. Intuitively, the monolingual classifier aims to learn a transformation from the input space to the final feature space, from which the prototypes for ProtoNet and ProtoMAML are created, in which the learned classes are encoded in their own disjoint sub-spaces such that a linear combination of these features can be used to correctly classify instances. ProtoNet aims to learn a similar transformation, but uses a Nearest Neighbours approach to classify instances instead. ProtoMAML on the other hand benefits the most from prototypes which can be used to classify instances *after* the inner-loop updates have been performed. This, in combination with the fact that the first-order approximation of ProtoMAML cannot differentiate through the creation of the prototypes, could explain the difference in performance gain with respect to ProtoNet.

8 Conclusion

We proposed a meta-learning framework for few-shot cross- and multilingual joint-learning for document classification tasks in different domains. We demonstrated that it leads to consistent gains over traditional supervised learning on a wide array of data availability and diversity settings, and showed that it thrives in settings with a heterogenous task distribution. We presented an effective adaptation to ProtoMAML and, among others, obtained a new state of the art on German, Italian, Japanese and Russian in the few-shot setting on MLDoc.

9 Acknowledgements

This work was supported by Deloitte Risk Advisory B.V., the Netherlands.

References

Antreas Antoniou, Harrison Edwards, and Amos Storkey. 2018. How to train your maml. *arXiv preprint arXiv:1810.09502*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

Yoshua Bengio, Samy Bengio, and Jocelyn Cloutier. 1990. *Learning a synaptic learning rule*. Citeseer.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4217–4226, Hong Kong, China. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. *arXiv preprint arXiv:1908.10423*.

Julian Eisenschlos, Sebastian Ruder, Piotr Czapla, Marcin Kardas, Sylvain Gugger, and Jeremy Howard. 2019. Multifit: Efficient multi-lingual language model fine-tuning. *arXiv preprint arXiv:1909.04761*.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org.

Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor OK Li. 2018. Meta-learning for low-resource neural machine translation. *arXiv preprint arXiv:1808.08437*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille.

Guokun Lai, Barlas Oguz, and Veselin Stoyanov. 2019. Bridging the domain gap in cross-lingual document classification. *arXiv preprint arXiv:1909.07009*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Ilya Loshchilov and Frank Hutter. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. *arXiv preprint arXiv:2003.02739*.

Abiola Obamuyide and Andreas Vlachos. 2019a. Meta-learning improves lifelong relation extraction. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 224–229, Florence, Italy. Association for Computational Linguistics.

Abiola Obamuyide and Andreas Vlachos. 2019b. Model-agnostic meta-learning for relation classification with limited supervision. In *Proceedings of the*

57th Annual Meeting of the Association for Computational Linguistics, pages 5873–5879, Florence, Italy. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS 2017 Workshop Autodiff Submission*.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127, Uppsala, Sweden. Association for Computationprettenhoferal Linguistics.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*.

Jurgen Schmidhuber. 1987. Evolutionary principles in self-referential learning. *On learning how to learn: The meta-meta-... hook.) Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1(2).

Holger Schwenk and Xian Li. 2018. A corpus for multilingual document classification in eight languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Sebastian Thrun and Lorien Pratt. 1998. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*.

Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. 2019. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Hongwei Yong, Jianqiang Huang, Xiansheng Hua, and Lei Zhang. 2020. Gradient centralization: A new optimization technique for deep neural networks. *arXiv preprint arXiv:2004.01461*.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. 2019. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9597–9608.

Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. 2015. Daily-aware personalized recommendation based on feature-level time series analysis. In *Proceedings of the 24th international conference on world wide web*, pages 1373–1383.

Boosting Low-Resource Biomedical QA via Entity-Aware Masking Strategies

Gabriele Pergola[1], **Elena Kochkina**[1,3], **Lin Gui**[1], **Maria Liakata**[1,2,3], **Yulan He**[1]
[1]University of Warwick, UK [2]Queen Mary University of London, UK
[3]The Alan Turing Institute, UK
{gabriele.pergola,e.kochkina,lin.gui,yulan.he}@warwick.ac.uk
m.liakata@qmul.ac.uk

Abstract

Biomedical question-answering (QA) has gained increased attention for its capability to provide users with high-quality information from a vast scientific literature. Although an increasing number of biomedical QA datasets has been recently made available, those resources are still rather limited and expensive to produce. Transfer learning via pre-trained language models (LMs) has been shown as a promising approach to leverage existing general-purpose knowledge. However, fine-tuning these large models can be costly and time consuming, often yielding limited benefits when adapting to specific themes of specialised domains, such as the COVID-19 literature. To bootstrap further their domain adaptation, we propose a simple yet unexplored approach, which we call *biomedical entity-aware masking* (BEM). We encourage masked language models to learn entity-centric knowledge based on the pivotal entities characterizing the domain at hand, and employ those entities to drive the LM fine-tuning. The resulting strategy is a downstream process applicable to a wide variety of masked LMs, not requiring additional memory or components in the neural architectures. Experimental results show performance on par with state-of-the-art models on several biomedical QA datasets.

1 Introduction

Biomedical question-answering (QA) aims to provide users with succinct answers given their queries by analysing a large-scale scientific literature. It enables clinicians, public health officials and end-users to quickly access the rapid flow of specialised knowledge continuously produced. This has led the research community's effort towards developing specialised models and tools for biomedical QA and assessing their performance on benchmark datasets such as BioASQ (Tsatsaronis et al., 2015). Producing such data is time-consuming and requires involving domain experts, making it an expensive process. As a result, high-quality biomedical QA datasets are a scarce resource. The recently released CovidQA collection (Tang et al., 2020), the first manually curated dataset about COVID-19 related issues, provides only 127 question-answer pairs. Even one of the largest available biomedical QA datasets, BioASQ, only contains a few thousand questions.

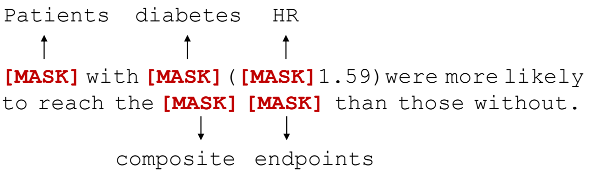
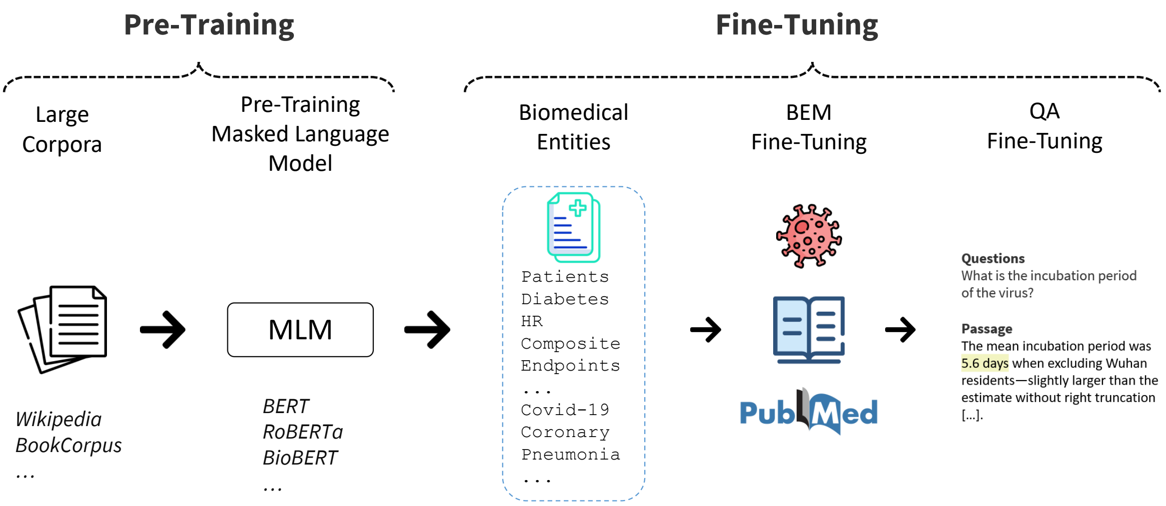


Figure 1: An excerpt of a sentence masked via the BEM strategy, where the masked words were chosen through a biomedical named entity recognizer. In contrast, BERT (Devlin et al., 2019) would randomly select the words to be masked, without attention to the relevant concepts characterizing a technical domain.

There have been attempts to fine-tune pre-trained large-scale language models for general-purpose QA tasks (Rajpurkar et al., 2016; Liu et al., 2019; Raffel et al., 2020) and then use them directly for biomedical QA. Furthermore, there has also been increasing interest in developing domain-specific language models, such as BioBERT (Lee et al., 2019) or RoBERTa-Biomed (Gururangan et al., 2020), leveraging the vast medical literature available. While achieving state-of-the-art results on the QA task, these models come with a high computational cost: BioBERT needs ten days on eight GPUs to train (Lee et al., 2019), making it prohibitive for researchers with no access to massive computing resources.

An alternative approach to incorporating external knowledge into pre-trained language models is to drive the LM to focus on pivotal entities characterising the domain at hand during the fine-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1977–1985
April 19 - 23, 2021.



Questions
What is the incubation period of the virus?

Passage
The mean incubation period was 5.6 days when excluding Wuhan residents—slightly larger than the estimate without right truncation [...].

Figure 2: A schematic representation of the main steps involved in fine-tuning masked language models for the QA task through the biomedical entity-aware masking (BEM) strategy.

tuning stage. Similar ideas were explored in works by Zhang et al. (2019), Sun et al. (2020), which proposed the ERNIE model. However, their adaptation strategy was designed to generally improve the LM representations rather than adapting it to a particular domain, requiring additional objective functions and memory. In this work we aim to enrich existing general-purpose LM models (e.g. BERT (Devlin et al., 2019)) with the knowledge related to key medical concepts. In addition, we want domain-specific LMs (e.g. BioBERT) to re-encode the already acquired information around the medical entities of interests for a particular topic or theme (e.g. literature relating to COVID-19).

Therefore, to facilitate further domain adaptation, we propose a simple yet unexplored approach based on a novel masking strategy to fine-tune a LM. Our approach introduces a *biomedical entity-aware masking* (BEM) strategy encouraging masked language models (MLMs) to learn entity-centric knowledge (§2). We first identify a set of entities characterising the domain at hand using a domain-specific entity recogniser (SciSpacy (Neumann et al., 2019)), and then employ a subset of those entities to drive the masking strategy while fine-tuning (Figure 1). The resulting BEM strategy is applicable to a vast variety of MLMs and does not require additional memory or components in the neural architectures. Experimental results show performance on a par with the state-of-the-art models for biomedical QA tasks (§4) on several biomedical QA datasets. A further qualitative assessment provides an insight into how QA pairs benefit from the proposed approach.

2 BEM: A Biomedical Entity-Aware Masking Strategy

The fundamental principle of a masked language model (MLM) is to generate word representations that can be used to predict the missing tokens of an input text. While this general principle is adopted in the vast majority of MLMs, the particular way in which the tokens to be masked are chosen can vary considerably. We thus proceed analysing the random masking strategy adopted in BERT (Devlin et al., 2019) which has inspired most of the existing approaches, and we then introduce the biomedical entity-aware masking strategy used to fine-tune MLMs in the biomedical domain.

BERT Masking strategy. The masking strategy adopted in BERT randomly replaces a predefined proportion of words with a special `[MASK]` token and the model is required to predict them. In BERT, 15% of tokens are chosen uniformly at random, 10% of them are swapped into random tokens (thus, resulting in an overall 1.5% of the tokens randomly swapped). This introduces a rather limited amount of noise with the aim of making the predictions more robust to trivial associations between the masked tokens and the context. While another 10% of the selected tokens are kept without modifications, the remaining 80% of them are replaced with the `[MASK]` token.

Biomedical Entity-Aware Masking Strategy We describe an entity-aware masking strategy which only masks biomedical entities detected by a domain-specific named entity recogniser (SciS-

#	Model	CovidQA			BioASQ 7b		
		P@1	R@3	MRR	SAcc	LAcc	MRR
1	**BERT**	0.081*	0.117*	0.159*	0.012	0.032	0.027
2	`+ BioASQ`	0.125	0.177	0.206	0.226	0.317	0.262
3	`+ STM + BioASQ`	0.132	0.195	0.218	0.233	0.325	0.265
4	`+ BEM + BioASQ`	0.145	0.278	0.269	0.241	0.341	0.288
5	**RoBERTa**	0.068	0.115	0.122	0.023	0.041	0.036
6	`+ BioASQ`	0.106	0.155	0.178	0.278	0.324	0.294
7	`+ STM + BioASQ`	0.112	0.167	0.194	0.282	0.333	0.300
8	`+ BEM + BioASQ`	0.125	0.198	0.236	0.323	0.374	0.325
9	**RoBERTa-Biomed**	0.104	0.163	0.192	0.028	0.044	0.037
10	`+ BioASQ`	0.128	0.355	0.315	0.415	0.398	0.376
11	`+ STM + BioASQ`	0.136	0.364	0.321	0.423	0.410	0.397
12	`+ BEM + BioASQ`	0.143	0.386	0.347	**0.435**	0.443	0.398
13	**BioBERT**	0.097*	0.142*	0.170*	0.031	0.046	0.039
14	`+ BioASQ`	0.166	0.419	0.348	0.410†	0.474†	0.409†
15	`+ STM + BioASQ`	0.172	0.432	0.385	0.418	0.482	0.416
16	`+ BEM + BioASQ`	*0.179*	**0.458**	*0.391*	0.421	**0.497**	**0.434**
17	**T5 LM**						
18	`+ MS-MARCO`	**0.282***	0.404*	**0.415***	—	—	—

Table 1: Performance of language models on the CovidQA and BioASQ 7b1 dataset. Values referenced with * come from the Tang et al. (2020) work and with † from Yoon et al. (2020).

pacy[1]). Compared to the random masking strategy described above, which is used to pre-train the masked language models, the introduced entity-aware masking strategy is adopted to boost the fine-tuning process for biomedical documents. In this phase, rather than randomly choosing the tokens to be masked, we inform the model of the relevant tokens to pay attention to, and encourage the model to refine its representations using the new surrounding context.

Replacing strategy We decompose the BEM strategy into two steps: (1) *recognition* and (2) *sub-sampling and substitution*. During the *recognition phase*, a set of biomedical entities $\mathcal{E}$ is identified in advance over a training corpus.

Then, at the *sub-sampling and substitution* stage, we first sample a proportion ρ of biomedical entities $\mathcal{E}_f \in \mathcal{E}$. The resulting entity subsets $\mathcal{E}_f$ is thus dynamically computed at batch time, in order to introduce a diverse and flexible spectrum of masked entities during training. For consistency, we use the same tokeniser for the documents d_i in the batch and the entities $e_j \in \mathcal{E}$. Then, we substitute all the k entity mentions $w^k_{e_j}$ in d_i with the special token `[MASK]`, making sure that no consecutive entities are replaced. The substitution takes place at batch time, so that the substitution is a downstream process suitable for a wide typology of MLMs. A diagram synthesizing the involved steps is reported in Figure 2.

3 Evaluation Design

Biomedical Reading Comprehension. We represent a document as $d_i := (s^i_0, \ldots, s^i_{j-1})$, a sequence of sentences, in turn defined as $s_j := (w^j_0, \ldots, w^j_{k-1})$, with w_k a word occurring in s_j. Given a question q, the task is to retrieve the span $w^j_s, \ldots, w^j_{s+t}$ from a document d_j that can answer the question. We assume the extractive QA setting where the answer span to be extracted lies entirely within one, or more than one document d_i.

In addition, for consistency with the CovidQA dataset and to compare with results in Tang et al. (2020), we consider a further and sightly modified setting in which the task consists of retrieving the sentence s^i_j that most likely contains the exact answer. This sentence level QA task mitigates the non-trivial ambiguities intrinsic to the definition of the exact span for an answer, an issue particularly relevant in the medical domain and well-know in the literature (Voorhees and Tice, 1999)[2].

Datasets. We assess the performance of the proposed masking strategies on two biomedical datasets: CovidQA and BioASQ.

---

[1] https://scispacy.apps.allenai.org/

[2] Consider, for instance, the following QA pair: *"What is the incubation period of the virus?"*, *"6.4 days (95% 175 CI 5.3 to 7.6)"*, where a model returning just *"6.4 days"* would be considered wrong.

BERT with STM	BERT with BEM
*What is the **OR** for severe infection in COVID-19 patients with hypertension?*	
- There were significant correlations between COVID-19 severity and [..], diabetes [OR=2.67], coronary heart disease [OR=2.85]. **-** Compared with the non-severe patient, the pooled odds ratio of hypertension, respiratory system disease, cardiovascular disease in severe patients were (OR 2.36, ..), (OR 2.46, ..) and (OR 3.42, ..).	**-** There were significant correlations between COVID-19 severity and [..], diabetes [OR=2.67], coronary heart disease [OR=2.85]. **-** Compared with the non-severe patient, the pooled odds ratio of hypertension, respiratory system disease, cardiovascular disease in severe patients were (OR 2.36, ..), (OR 2.46, ..) and (OR 3.42, ..).
*What is the **HR** for severe infection in COVID-19 patients with hypertension?*	
- - - -	**-** After adjusting for age and smoking status, patients with COPD (HR 2.681), diabetes (HR 1.59), and malignancy (HR 3.50) were more likely to reach to the composite endpoints than those without.
*What is the **RR** for severe infection in COVID-19 patients with hypertension?*	
- - - -	**-** In univariate analyses, factors significantly associated with severe COVID-19 were male sex (14 studies; pooled RR=1.70, ...), hypertension (10 studies 2.74 ...),diabetes (11 studies ...), and CVD (..).

Table 2: Examples of questions and retrieved answers using BERT fine-tuned either with its original masking approach or with the biomedical entity-aware masking (BEM) strategy.

CovidQA (Tang et al., 2020) is a manually curated dataset based on the AI2's COVID-19 Open Research Dataset (Wang et al., 2020). It consists of 127 question-answer pairs with 27 questions and 85 unique related articles. This dataset is too small for supervised training, but is a valuable resource for zero-shot evaluation to assess the unsupervised and transfer capability of models.

BioASQ (Tsatsaronis et al., 2015) is one of the larger biomedical QA datasets available with over 2000 question-answer pairs. To use it within the extractive questions answering framework, we convert the questions into the SQuAD dataset format (Rajpurkar et al., 2016), consisting of question-answer pairs and the corresponding *passages*, medical articles containing the answers or clues with a length varying from a sentence to a paragraph. When multiple passages are available for a single question, we form additional question-context pairs combined subsequently in a postprocessing step to choose the answer with highest probability, similarly to Yoon et al. (2020). For consistency with the CovidQA dataset, we report our evaluation exclusively on the factoid questions of the BioASQ 7b Phase B1.

Baselines. We use the following unsupervised neural models as baselines: the out-of-the-box BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), as well as their variants BioBERT (Lee et al., 2019) and RoBERTa-Biomed (Gururangan et al., 2020) fine-tuned on medical and scientific corpora.

To highlight the impact of different fine-tuning strategies, we examine several configurations depending on the data and the masking strategy adopted. We experiment using the BioASQ QA training pairs during the fine-tuning stage and denote the models using them with `+BioASQ`. When we fine-tune the models on the corpus consisting of PubMed articles referred within the BioASQ and AI2's COVID-19 Open Research dataset, we compare two masking strategies denoted as `+STM` and `+BEM`, where `+STM` indicates the standard masking strategy of the model at hand and `+BEM` is our proposed strategy. We additionally report the T5 (Raffel et al., 2020) performance over CovidQA, which constitutes the current state-of-the-art (Tang et al., 2020)[3].

Metrics. To facilitate comparisons, we adopt the same evaluation scores used in Tang et al. (2020) to assess the models on the CovidQA dataset, i.e. mean reciprocal rank (MRR), precision at rank one (P@1), and recall at rank three (R@3); similarly, for the BioASQ dataset, we use the strict accuracy (SAcc), lenient accuracy (LAcc) and MRR, the BioASQ challenge's official metrics.

4 Experimental Results and Discussion

We report the results on the QA tasks in Table 1.

Among the unsupervised models, BERT achieves slightly better performance than RoBERTa on CovidQA, yet the situation is reversed on BioASQ (rows 1,5). The low precision of the two models (especially on the BioASQ dataset) confirms the difficulties in generalising to the biomedical domain. Specialised language

---

[3] We attach supplementary results in Appx. A on SQuAD (Tab. A1) and the *perplexity* of MLMs when fine-tuned on the medical collection with different masking strategies (Fig. A1)

models such as RoBERTa-Biomed and BioBERT show a significant improvement on the CovidQA dataset, but a rather limited one on BioASQ (rows 9,13), highlighting the importance of having larger medical corpora to assess the model's effectiveness. A general boost in performance is shared across models fine-tuned on the QA tasks, with a large benefit from the BioASQ QA. The performance gains obtained by the specialised models (BioBERT and RoBERTa-Biomed) suggest the importance of transferring not only the domain knowledge but also the ability to perform the QA task itself (rows 9,10; 13,14).

A further fine-tuning step before the training over the QA pairs has been proven beneficial for all of the models. The BEM masking strategy has significantly amplified the model's generalisability, with an increased adaptation to the biomedical themes shown by the notable improvement in R@3 and MRR; with the R@3 outperforming the state-of-the-art results of T5 fine-tuned on MS-MARCO (Bajaj et al., 2018) and proving the effectiveness of the BEM strategy.

Table 2 reports questions from the CovidQA related to three statistical indices (i.e. Odds Ratio, Hazard Ratio and Relative Risk) to assess the risk of an event occurring in a group (e.g. infections or death). We notice that even though the indices are mentioned as abbreviations, BERT fine-tuned with the STM is able to retrieve sentences with the exact answer for just one of three questions. By contrast, BERT fine-tuned with the BEM strategy succeeds in retrieving at least one correct sentence for each question. This example suggests the importance of placing the emphasis on the entities, which might be overlooked by LMs during the training process despite being available.

5 Related Work

Our work is closely related to two lines of research: the design of masking strategies for LMs and the development of specialized models for the biomedical domain.

Masking strategies. Building on top of the BERT's masking strategy (Devlin et al., 2019), a wide variety of approaches has been proposed (Liu et al., 2019; Yang et al., 2019; Jiang et al., 2020).

A family of masking approaches aimed at leveraging entity and phrase occurrences in text. SpanBERT, Joshi et al. (2020) proposed to mask and predict whole spans rather than standalone tokens and to make use of an auxiliary objective function. ERNIE (Zhang et al., 2019) is instead developed to mask well-known named entities and phrases to improve the external knowledge encoded. Similarly, KnowBERT (Peters et al., 2019) explicitly model entity spans and use an entity linker to an external knowledge base to form knowledge enhanced entity-span representations. However, despite the analogies with the BEM approach, the above masking strategies were designed to generally improve the LM representations rather than adapting them to particular domains, requiring additional objective functions and memory.

Biomedical LMs. Particular attention has been devoted to the adaptation of LMs to the medical domain, with different corpora and tasks requiring tailored methodologies. BioBERT (Lee et al., 2019) is a biomedical language model based on BERT-*Base* with additional pre-training on biomedical documents from the PubMed and PMC collections using the same training settings adopted in BERT. BioMed-RoBERTa (Gururangan et al., 2020) is instead based on RoBERTa-*Base* (Liu et al., 2019) using a corpus of 2.27M articles from the Semantic Scholar dataset (Ammar et al., 2018). SciBERT (Beltagy et al., 2019) follows the BERT's masking strategy to pre-train the model from scratch using a scientific corpus composed of papers from Semantic Scholar (Ammar et al., 2018). Out of the 1.14M papers used, more than 80% belong to the biomedical domain.

6 Conclusion

We presented BEM, a biomedical entity-aware masking strategy to boost LM adaptation to low-resource biomedical QA. It uses an entity-driven masking strategy to fine-tune LMs and effectively lead them in learning entity-centric knowledge based on the pivotal entities characterizing the domain at hand. Experimental results have shown the benefits of such an approach on several metrics for biomedical QA tasks.

Acknowledgements

This work is funded by the EPSRC (grant no. EP/T017112/1, EP/V048597/1). YH is supported by a Turing AI Fellowship funded by the UK Research and Innovation (UKRI) (grant no. EP/V020579/1).

References

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, and Oren Etzioni. 2018. Construction of the literature graph in semantic scholar. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 84–91.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A human generated machine reading comprehension dataset.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

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

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL20*.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

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

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics 2019*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, Florence, Italy.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

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

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

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*.

Raphael Tang, Rodrigo Nogueira, Edwin Zhang, Nikhil Gupta, Phuong Cam, Kyunghyun Cho, and Jimmy Lin. 2020. Rapidly bootstrapping a question answering dataset for COVID-19.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1).

Ellen M. Voorhees and Dawn M. Tice. 1999. The trec-8 question answering track evaluation. In *In Text Retrieval Conference TREC-8*, pages 83–105.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32*.

Wonjin Yoon, Jinhyuk Lee, Donghyeon Kim, Minbyul Jeong, and Jaewoo Kang. 2020. Pre-trained language model for biomedical question answering. In *Machine Learning and Knowledge Discovery in Databases*.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, ACL19*, Florence, Italy.

A Appendix

We further examined whether the fine-tuning of the QA pairs affects not only the model adaptation to the QA task but it further helps realign the repression for the domain at hand. The report scores point out that the vanilla LMs are the ones gaining the most when using in-domain QA pairs, such as BioASQ, compared to the SQuAD (rows 2,3; 9,10). The advantage tends to be reduced on already specialised LMs (rows 16,17; 23;24).

#	Model	**CovidQA**			**BioASQ 7b**		
		P@1	R@3	MRR	SAcc	LAcc	MRR
1	**BERT**	0.081*	0.117*	0.159*	0.012	0.032	0.027
2	`+ SQuAD`	0.110	0.131	0.158	0.292	0.343	0.318
3	`+ BioASQ`	0.125	0.177	0.206	0.226	0.317	0.262
4	`+ STM + SQuAD`	0.114	0.146	0.173	0.305	0.355	0.336
5	`+ STM + BioASQ`	0.132	0.195	0.218	0.233	0.325	0.265
6	`+ BEM + SQuAD`	0.126	0.173	0.191	0.317	0.371	0.349
7	`+ BEM + BioASQ`	0.145	0.278	0.269	0.241	0.341	0.288
8	**RoBERTa**	0.068	0.115	0.122	0.023	0.041	0.036
9	`+ SQuAD`	0.098	0.134	0.160	0.353	0.365	0.328
10	`+ BioASQ`	0.106	0.155	0.178	0.278	0.324	0.294
11	`+ STM + SQuAD`	0.107	0.148	0.175	0.361	0.388	0.347
12	`+ STM + BioASQ`	0.112	0.167	0.194	0.282	0.333	0.300
13	`+ BEM + SQuAD`	0.114	0.162	0.185	0.368	0.391	0.353
14	`+ BEM + BioASQ`	0.125	0.198	0.236	0.323	0.374	0.325
15	**RoBERTa-Biomed**	0.104	0.163	0.192	0.028	0.044	0.037
16	`+ SQuAD`	0.111	0.308	0.288	0.376	0.382	0.358
17	`+ BioASQ`	0.128	0.355	0.315	0.415	0.398	0.376
18	`+ STM + SQuAD`	0.118	0.314	0.297	0.381	0.390	0.367
19	`+ STM + BioASQ`	0.136	0.364	0.321	0.423	0.410	0.397
20	`+ BEM + SQuAD`	0.121	0.331	0.323	0.385	0.397	0.378
21	`+ BEM + BioASQ`	0.143	0.386	0.347	**0.435**	0.443	0.398
22	**BioBERT**	0.097*	0.142*	0.170*	0.031	0.046	0.039
23	`+ SQuAD`	0.161*	0.403*	0.336*	0.381	0.445	0.397
24	`+ BioASQ`	0.166	0.419	0.348	0.410†	0.474†	0.409†
25	`+ STM + SQuAD`	0.161	0.411	0.339	0.387	0.447	0.401
26	`+ STM + BioASQ`	0.172	0.432	0.385	0.418	0.482	0.416
27	`+ BEM + SQuAD`	0.168	0.427	0.354	0.391	0.458	0.423
28	`+ BEM + BioASQ`	*0.179*	**0.458**	*0.391*	0.421	**0.497**	**0.434**
29	**T5 LM**						
30	`+ MS-MARCO`	**0.282***	0.404*	**0.415***	—	—	—

Table A1: Performance of language models on the CovidQA and BioASQ 7b1 dataset. Values referenced with * comes from the Tang et al. (2020) work and with † from Yoon et al. (2020).

In Figure A1, we report the LM perplexity obtained when fine-tuning the model with the standard masking strategy versus the BEM strategy with different proportion of medical entities. Vanilla LMs experienced a huge gain with just a small fraction of entities, while already specialised LMs has a lower but still significant improvement. This could be expected as the specialised LMs has already encoded a large domain knowledge with representations that need to be realigned to the new ones.

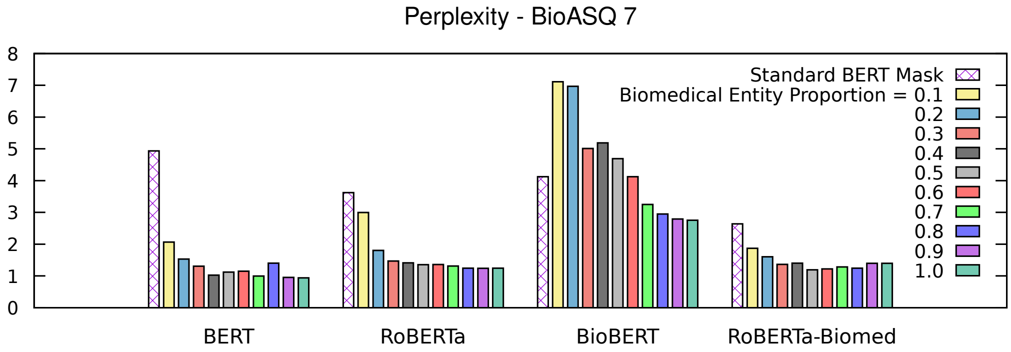


Figure A1: Perplexity of MLMs using different masking strategies on the collection of medical articles.

Attention-based Relational Graph Convolutional Network for Target-Oriented Opinion Words Extraction

Junfeng Jiang[*]
The University of Tokyo
Tokyo, Japan
jiangjf@is.s.u-tokyo.ac.jp

An Wang[*]
Tokyo Institute of Technology
Tokyo, Japan
wang.a.aa@m.titech.ac.jp

Akiko Aizawa[†]
National Institute of Informatics
Tokyo, Japan
aizawa@nii.ac.jp

Abstract

Target-oriented opinion words extraction (TOWE) is a subtask of aspect-based sentiment analysis (ABSA). It aims to extract the corresponding opinion words for a given opinion target in a review sentence. Intuitively, the relation between an opinion target and an opinion word mostly relies on syntactics. In this study, we design a directed syntactic dependency graph based on a dependency tree to establish a path from the target to candidate opinions. Subsequently, we propose a novel attention-based relational graph convolutional neural network (ARGCN) to exploit syntactic information over dependency graphs. Moreover, to explicitly extract the corresponding opinion words toward the given opinion target, we effectively encode target information in our model with the target-aware representation. Empirical results demonstrate that our model significantly outperforms all of the existing models on four benchmark datasets. Extensive analysis also demonstrates the effectiveness of each component of our models. Our code is available at `https://github.com/wcwowwwww/towe-eacl`.

1 Introduction

Target-oriented opinion words extraction (TOWE) (Fan et al., 2019) is a subtask of aspect-based sentiment analysis (ABSA) (Hu and Liu, 2004; Pontiki et al., 2016). Given a review and an opinion target in the sentence, the objective of TOWE is to extract the corresponding opinion words describing or evaluating the opinion targets from the review. Opinion targets are the words or phrases representing features or entities toward which users express their attitudes, whereas opinion words referring to those terms are used to express attitudes or opinions explicitly.

The food is tasty and portion sizes are appropriate.
Target: food Opinion: tasty

The food is tasty and portion sizes are appropriate.
Target: portion size Opinion: appropriate

Figure 1: Examples of TOWE task. The words highlighted in orange represent the given opinion targets, whereas the words in blue represent the corresponding opinion words.

Figure 1 shows two examples of TOWE. In the review "*The food is tasty and portion sizes are appropriate* .", the terms "*food*" and "*portion sizes*" are two given opinion targets. TOWE needs to extract the word "*tasty*" as the opinion word for the opinion target "*food*" and the opinion word "*appropriate*" for the opinion target "*portion sizes*".

Therefore, the first challenge is to effectively introduce the opinion target information into our model. Fan et al. (2019) designed the IO-BiLSTM to encode the context before and after the given opinion targets separately to represent the position of the existing opinion targets. Wu et al. (2020) introduced position embeddings based on the relative distance toward opinion targets. However, both studies only introduce parts of target information (the position information of targets). In this paper, we introduce the target-aware representation to fully exploit opinion target information in a concise way, which is especially important when our models are used for real-world reviews.

Becase TOWE can be viewed as a syntactic task, a natural solution is analysing the relationship between opinion targets and opinion words by dependency parsing. Recently, owing to the great success of graph convolutional networks (GCNs) in various fields (Kipf and Welling, 2016; Chen et al., 2018; Marcheggiani et al., 2018), a few researchers have attempted to encode the syntactic

---

[*] These authors contributed equally to this work; the order is random.

[†] Corresponding author.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1986–1997
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

dependency information with GCNs to build a robust dependency encoder. For example, GCNs over the dependency tree have been exploited to perform semantic role labelling (Marcheggiani and Titov, 2017) and named entity recognition (Cetoli et al., 2017).In addition, several studies explore GCNs over a dependency graph to complete the ABSA task (Sun et al. (2019), Zhang et al. (2019), Liang et al. (2020), Wang et al. (2020)).

However, it is worth mentioning that TOWE is defined as a sequence labelling task, and the manner in which GCNs are applied to TOWE effectively is yet to be explored. In this study, we first construct a directed graph based on a dependency tree to be more suitable for TOWE. Subsequently, we propose ARGCN, which can enhance our model by encoding syntactic information. ARGCN can be seen as extending the Relational Graph Convolutional Networks (R-GCNs) (Schlichtkrull et al., 2018) with the distance-aware attention mechanism. ARGCN can consider the semantic relevance and syntactic relevance between words simultaneously when it propagates information. In addition, sequential information is extremely important for sequence-labelling tasks. Therefore, after using multi-layer graph convolutions to encode syntactic information, we feed the syntactic representation to a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to capture the sequential information.

Experiments on four benchmark datasets demonstrate that our base model, Target-BiLSTM which is a BiLSTM with target-aware inputs has a similar or better performance than the state-of-the-art model, although we do not introduce extra external knowledge. In addiction, our full model ARGCN further improves the performance and significantly outperforms all of the existing models on four benchmark datasets. Furthermore, extensive experiments

demonstrate the effectiveness and necessity of all components in our full model. To the best of our knowledge, it is the first work on applying GCNs to the TOWE task.

The contributions of this paper can be summarized as follows.

- We propose target-aware representation to effectively introduce opinion target information. An empirical study shows it is significant and extensible for the TOWE task.
- We exploit syntactic dependency graphs of sentences and establish the relations between opinion targets and the corresponding opinion words.
- We propose a novel attention-based relational graph convolutional network, ARGCN, an extension of R-GCNs suited to encode syntactic dependency information.
- We propose an ARGCN-based TOWE model. Experimental results show that it significantly outperforms the state-of-the-art model on all datasets of the TOWE task.

2 Related Work

As subtasks of ABSA, a series of early studies focused on opinion targets extraction, including unsupervised/semi-supervised methods (Qiu et al., 2011; Liu et al., 2012, 2013) and supervised methods (Jakob and Gurevych, 2010; Li et al., 2010). Some recent studies extracted opinion targets and opinion words jointly in a uniform framework and achieved promising results (Wang et al., 2016; Li and Lam, 2017). However, they did not extract the corresponding relation between opinion targets and opinion words. Moreover, studies on extracting paired opinion relations are rare (Hu and Liu, 2004; Zhuang et al., 2006). Because it is important for downstream sentiment analysis and real-world applications, Fan et al. (2019) proposed a new subtask of ABSA, target-oriented word extraction, aiming to extract the corresponding opinion words for the given opinion targets in a review. They released four benchmark datasets for evaluation, designed a target-fused model, and achieved excellent performance. Wu et al. (2020) adopted transfer learning to transfer latent opinion information from the sentiment analysis model to the TOWE model. In this study, we also focus on the TOWE task.

Since Kipf and Welling (2016) proposed their GCN with some simplifications on ChebNet (Defferrard et al., 2016), a variety of graph convolutional networks appeared (Veličković et al., 2018; Schlichtkrull et al., 2018; Busbridge et al., 2019) and achieved great success in many fields, including computer vision (Chen et al., 2018; Garcia and Estrach, 2018; Wang et al., 2019), natural language processing (Marcheggiani and Titov, 2017; Marcheggiani et al., 2018; Yao et al., 2019; Wang et al., 2020) and even in chemistry (De Cao and Kipf, 2018). One of the reasons why GCNs work well in several fields is that they can naturally process the graph-structured data to greatly exploit

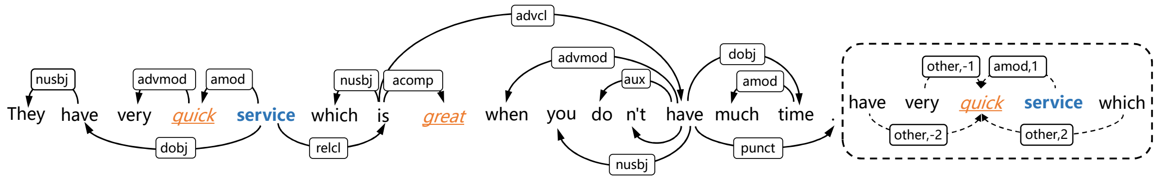


Figure 2: An example of the syntactic dependency graph based on dependency tree generated by spaCy[*]dependency parser. “service” is the given opinion target. “quick” and “great” are two corresponding opinion words. In the left figure, we show the reshaped dependency graph before adding extra edges for those who do not have dependency relation but have close distance (closer than a threshold D). In the right figure, we can see that each edge has two features: dependency relation and distance. And we show all edges coming towards the word ”quick” in the final graph as an example.

the latent information behind the graph structure. Therefore, they are proven to be efficient especially with only a small amount of data (Kipf and Welling, 2016; Garcia and Estrach, 2018).

Recently, a few studies have tried applying GCNs over the dependency graph to complete some ABSA tasks. Sun et al. (2019) proposed the CDT to perform GCN over a dependency tree together with contextual representations extracted by BiLSTM. Liang et al. (2020) introduced dependency relational embedding to GCN (Kipf and Welling, 2016) to complete ABSA with their DREGCN. Specially, the R-GAT-ABSA (Wang et al., 2020) is a newly proposed architecture for the ABSA task. It focuses on the GAT (Veličković et al., 2018) and extends it by introducing relational embedding for calculating relational attention.

3 Our Methods

3.1 Task Formalization

TOWE aims to extract corresponding opinion words based on the given opinion targets. Formally, we have a review sentence $s = \{w_1, w_2, ..., w_n\}$ containing n words. Then, we adopt the BIO tagging scheme (Ramshaw and Marcus, 1999) as Fan et al. (2019) do in their paper. For each words in the sentence, we tag them as $y_i \in \{B, I, O\}$ (B: Beginning, I: Inside, O: Others). For example, the sentence in figure 1 is tagged as “*The/O food/O is/O [tasty/B] and/O portion/O sizes/O are/O appropriate/O ./O*”, indicating the opinion word “*tasty*” for target “*food*”.

3.2 Target-Aware Representation

As described above, we should extract the corresponding opinion words based on the given opinion targets. Therefore, our model should be aware of which words are the opinion targets and identify the corresponding opinion words. All previous studies only encode the position information for targets. In contrast, we directly introduce category embeddings with respect to the target tag of words to fully introduce target information in the TOWE model. Figure 3 shows an overview of our model.

We denote the category embedding table as $\mathbf{T}^t \in \mathbb{R}^{3\times d^t}$, where d^t is the dimension of the category embedding. Next, we can obtain the target embedding of each word and form a target embedding matrix of a sentence as $\mathbf{E}^t = [\mathbf{e}_1^t; \mathbf{e}_2^t; \cdots ; \mathbf{e}_n^t]$.

To retain the target information clearly when feeding to the next module, we concatenate it together with the word representation.

$$\mathbf{e}_i = [\mathbf{e}_i^w, \mathbf{e}_i^t] \tag{1}$$

where e_i^w is word representation of word i, [,] represents the concatenation operation.

Thus, our model can understand which words are opinion targets. The target embedding table is jointly optimized during training so that our model can learn the proper target embeddings specifically for the TOWE task.

For simplicity, we denote our target-aware representation as $\mathbf{E} = [\mathbf{e}_1; \mathbf{e}_2; \cdots ; \mathbf{e}_n]$ and then feed it to the following modules.

3.3 Syntactic Dependency Graph

In this section, we provide a detailed description of our method of building a suitable syntactic dependency graph for the TOWE task. For a given sentence $s = \{w_1, w_2, \cdots , w_n\}$, after dependency parsing, we obtain a dependency tree. Figure 2 is the original dependency tree of the sentence “The food is tasty and portion sizes are appropriate .”. Next, we add some edges whose relative distance in the sentence is smaller than a given threshold D.

We formally define the directed graph as $G = \{V, E, \mathcal{R}, \mathcal{P}\}$, where $V = \{v_i\}_{i=1}^n$ is the set of nodes, which are words in a sentence, $E =$

---

[*] `https://spacy.io/`

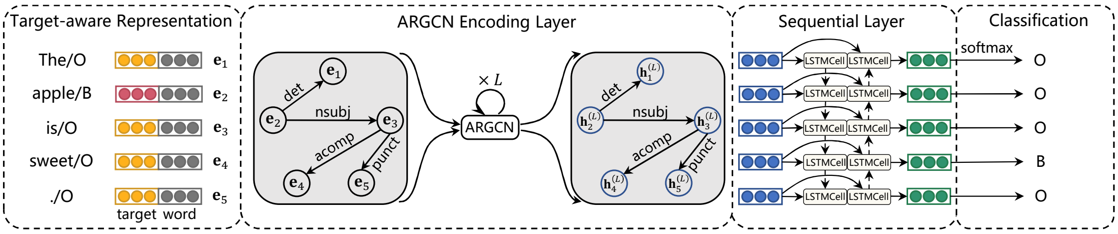


Figure 3: Overview of ARGCN. We generate the target-aware representation as the input node representation. Then, L-layers of ARGCN are applied over our syntactic dependency graph. After encoding, we capture sequential information with BiLSTM. Finally, we perform prediction with softmax classifier. Because of space limits, we omit other edges except for those who have dependency relations.

$\{e_{ij}\}_{i,j=1}^{n}$ is the set of edges, and $\mathcal{R} = \{r_{ij}\}_{i,j=1}^{n}$ is the set of edge relational types, where r_{ij} is the corresponding dependency relation from v_i to v_j. If there is not any dependency relation between v_i and v_j whose relative distance is smaller than D, we add an edge between them and set a special edge type for it, such as *other*. $\mathcal{P} = \{p_{ij}\}_{i,j=1}^{n}$ represents the set of relative positions, and p_{ij} is the relative position from v_i to v_j in the sentence. Note that e_{ij} indicates v_i is the neighbour of v_j.

To ensure the target information can correctly propagate to the latent opinion words, we redirect some specific dependency relations linking to the target words. Regarding dependency trees, when the edge type is *nsubj* or *dobj*, the direction of the edge is from predicate to subject or object. Hence, the information of the subject or object cannot flow through the predicate. Thus, we reverse the dependency edge when it links target words and its type is *nsubj* (nominal subject) or *dobj* (direct object). In addition, we remove the *root* relation because it is a self-loop, which is not helpful for our model.

3.4 Attention-Based Relational Graph Convolutional Network (ARGCN)

To encode the well-designed syntactic dependency graph, we begin from R-GCNs (Schlichtkrull et al., 2018) and extend it with a distance-aware attention mechanism. In this paper, we propose an attention-based relational graph convolutional network (ARGCN). The main purpose of our model is to consider semantic and syntactic relevance between words simultaneously.

R-GCNs (Schlichtkrull et al., 2018) updated the hidden states of nodes by aggregating node representations of their neighbours according to the edge type of their connections,

$$\mathbf{h}'_i = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\mathbf{x}_{ij,r} + \mathbf{W}_1\mathbf{h}_i\right) \tag{2}$$

$$\mathbf{x}_{ij,r} = \frac{1}{c_{i,r}}\mathbf{W}_r\mathbf{h}_j \tag{3}$$

where $\mathcal{R}$ denotes the set of relations, $\mathbf{h}_i$ is the input representation of node v_i, $\mathbf{h}'_i$ is the output representation of node v_i, $\mathcal{N}_i^r$ is the set of neighbours of v_i under relation $r \in \mathcal{R}$, $\mathbf{W}_r$ and $\mathbf{W}_1$ are trainable parameters, and $c_{i,r}$ is a problem-specific normalization constant, which is usually assigned as the number of neighbours of v_i under relation r. Moreover, σ is an element-wise activation function.

In Equation (5), each relation r corresponds to a relation-specific matrix $\mathbf{W}_r$. To reduce the parameter number, we perform a basis decomposition (Schlichtkrull et al., 2018). In particular, we set the number of bases as one:

$$\mathbf{W}_r = b_r\mathbf{W}_0 \tag{4}$$

where b_r is the coefficient depending on r. In this way, every $\mathbf{W}_r$ shares $\mathbf{W}_0$ as the basis, thereby the number of parameters is greatly reduced. On the other hand, b_r denotes the influence with respect to relation types.

In ARGCN, we introduce a distance-aware attention mechanism to enhance the power of RGCN:

$$\mathbf{x}_{ij,r} = \alpha_{ij,r}\mathbf{W}_0\mathbf{h}_j \tag{5}$$

$$\alpha_{ij,r} = \sigma(\mathbf{c}^T[b_r, \beta_{ij}]) \tag{6}$$

where β_{ij} is the attention coefficient between v_i and v_j, and $\mathbf{c}$ is a trainable vector, which can adjust the influence of the relation and the attention coefficient. σ is an activation function, and we choose to use ReLU in ARGCN layers.

We assume that the attention coefficients between two nodes are based on the features of nodes and the relative position in the sentence. First, we obtain query and key by project node features $\mathbf{h}_i$ and $\mathbf{h}_j$ by multiplying the same projection matrix $\mathbf{W}_1$. Next, we get relative positional encoding $\mathbf{p}$ by a sinusoid encoding matrix as in Dai et al. (2019).

Then, we use a shared attention mechanism to perform attention on the query, key, and relative positional encoding:

$$o_{ij} = \sigma(\mathbf{a}^T[\mathbf{W}_1\mathbf{h}_j, \mathbf{W}_1\mathbf{h}_i, \mathbf{p}]) \tag{7}$$

where **a** is a trainable vector mapping the concatenated representation to a scalar.

Finally, we normalize o_{ij} across all neighbours of v_i using the softmax function:

$$\beta_{ij} = \frac{exp(o_{ij})}{\sum\limits_{k \in \mathcal{N}_i} exp(o_{ik})} \tag{8}$$

where β_{ij} indicates the importance of v_i toward v_j with respect to the node representations and the relative position.

In addition, extending our mechanism to employ multi-head attention helps to stabilize the learning process and enhances the performance. Specifically, K independent attention mechanisms execute the transformation of Equation (5).

$$\mathbf{x}^k_{ij,r} = \alpha^k_{ij,r}\mathbf{W}^k_0\mathbf{h}_j \tag{9}$$

where $\mathbf{W}^k_0 \in \mathbb{R}^{d_j \times d_k}$, d_j is the input dimension, and d_k is the dimension of each head. Then, the output of the multi-head attention mechanism is

$$\mathbf{x}_{ij,r} = \mathbf{W}_d[\mathbf{x}^1_{ij,r}, \mathbf{x}^2_{ij,r}, ..., \mathbf{x}^K_{ij,r}] \tag{10}$$

where $\mathbf{W}_d \in \mathbb{R}^{Kd_k \times d_{j+1}}$.

Unlike Vaswani et al. (2017), who chose to use d_{j+1}/K as a dimension of each head, we set $d_k = d_{j+1}$, which leads to slight performance gains based on preliminary experiments.

We find that aspect and opinion terms often have direct or indirect relations in the graph based on the syntactic dependency tree. For example, Figure 2 shows that the relation between “service” and “quick” is direct whereas that between “service” and “great” is indirect. To capture these direct or indirect relations, we use L-layers of ARGCN, because L successive ARGCNs result in the propagation of information across the L-th order neighbour.

Moreover, with the deepening network layers, ARGCN tends to be over-smooth. In order to alleviate this problem, we add a residual connection on each ARGCN layer:

$$\mathbf{h}^{l+1}_i = \mathbf{h}^l_i + \mathbf{h}'^l_i \tag{11}$$

where $\mathbf{h}^l_i$ is the input of v_i in l-th layer of ARGCN, and $\mathbf{h}'^l_i$ is the output of v_i in l-th layer of ARGCN. Thus, $\mathbf{h}^{l+1}_i$ is the input of $(l+1)$-th layer of ARGCN.

3.5 Sequential Layer

The insufficiency of ARGCN is that it cannot encode the sequential information, which is extremely important for the TOWE task because it is defined as a sequence-labelling task. Intuitively, prediction relies on the prediction label of the words before and after the current word. Therefore, the performance of the model will not be satisfactory without capturing sequential information.

Consequently, we feed the syntactic representation extracted from L-layers of ARGCN to a BiLSTM to capture the sequential information:

$$\hat{\mathbf{h}}_i = \text{BiLSTM}(\mathbf{h}^{(L)}_i, \hat{\mathbf{h}}_{i-1}) \tag{12}$$

where $\hat{\mathbf{h}}_i$ is the concatenation of the forward and backward output vectors at time-step i.

Many other studies that used GCNs over the dependency graph (Marcheggiani and Titov, 2017; Sun et al., 2019; Wang et al., 2020) often applied LSTM to encode the sequential information, fed the obtained contextual representation to GCNs, and used them for predictions. We also attempted to first encode the sequential relationship by LSTM and then feed them to ARGCN to finally predict the labels of words. However, the performance was impaired. We believe the reason is that the sequential relationship is essential for sequence-labelling tasks. If we collect it before encoding the dependency information, it will be confused through aggregation, leading to poor performance.

3.6 Model Training

After collecting the sequential information, we simply mapped the representations to the output space with a fully connected layer and calculated the probability of the labels of words with the softmax function:

$$\hat{\mathbf{y}}_i = \text{softmax}(\mathbf{W}_{fc}\hat{\mathbf{h}}_i + \mathbf{b}_{fc}) \tag{13}$$

where $\mathbf{W}_{fc}$ and $\mathbf{b}_{fc}$ are the trainable parameters of the fully connected layer.

Next, the cross-entropy loss is defined as

$$\mathcal{L} = -\sum_{i=1}^{n}\sum_{k=0}^{2}\mathbb{I}(y_i = k)\log(\hat{y}_{ik}; \Theta) \tag{14}$$

and minimized during training. Here, the opinion word tags $\{O, B, I\}$ are correspondingly numeralized as labels $\{0, 1, 2\}$, respectively, and y_i denotes the gold label.

4 Experiments

4.1 Datasets and Metrics

Following the previous works (Fan et al., 2019; Wu et al., 2020), we evaluate the models on four benchmark datasets, including **14res**, **14lap**, **15res** and **16res**. Explicitly, the datasets **14res** and **14lap** are annotated from SemEval Challenge 2014 task 4 (Pontiki et al., 2014). The **15res** and **16res** are annotated from SemEval Challenge 2015 task 12 (Pontiki et al., 2015) and SemEval Challenge 2016 task 5 (Pontiki et al., 2016) respectively. The suffixes "**res**" and "**lap**" indicate they are collected from restaurant reviews and laptop reviews, respectively.

Datasets		#sentences	#targets
14res	Train	1627	2643
	Test	500	865
14lap	Train	1158	1634
	Test	343	482
15res	Train	754	1076
	Test	325	436
16res	Train	1079	1512
	Test	329	457

Table 1: Statistics of the four benchmark datasets.

The original SemEval challenge datasets are very popular for ABSA subtasks. However, they only contain annotations of aspect terms. Therefore, Fan et al. (2019) extended the annotation to further annotate the corresponding opinion words based on the given opinion targets and ignored the cases without explicit opinion words. Detailed statistics are shown in Table 1.

For the classification task, we adopted commonly used evaluation metrics: precision, recall, and F1-score. An extraction is considered as correct only when the opinion words from the beginning to the end are all predicted exactly as the ground truth.

4.2 Experimental Settings

For ARGCN and Target-BiLSTM, we adopted 300-dimension GloVe word embeddings (Pennington et al., 2014) as our word representations. For ARGCN-bert and Target-BiLSTM-bert, we adopted the last hidden states of the pre-trained BERT (Devlin et al., 2018) as word representations and fine-tuned it jointly. Inspired by Xu et al. (2018), we fine-tuned the GloVe vectors during training to obtain a domain-specific representation. The dimension of target embedding was 3 and 100 for our base model and GCNs-based models, respectively. We implemented our models with PyTorch (Paszke et al., 2019). We introduced 10 layers of ARGCN with 128 channels, 8 attention heads and set the hidden size of BiLSTM to 128.

We used spaCy (Honnibal and Johnson, 2015) as our dependency parser. To improve the generalization of ARGCN, dropout (Hinton et al., 2012) layers were applied after the activation with the probability of 0.5. The threshold of relative distance was set to be 3. All of the parameters were optimized by Adam optimizer (Kingma and Ba, 2014). The initial learning rate was 1×10^{-3}. We randomly split 20% of the training set as the validation set to fine-tune the hyperparameters and apply early stopping. Subsequently, we tested our models and averaged the results of 5 runs.

4.3 Compared Methods

We compare our model with several methods which can be categorized into three groups.

- **Early Solutions**: Some early solutions including rule-based methods and trivial deep learning methods are assigned to the first group. Inspired by Hu and Liu (2004) and Zhuang et al. (2006), Fan et al. (2019) proposed the Distance-rule and Dependency-rule as two representative rule-based methods. Following Liu et al. (2015) and Tang et al. (2016), Fan et al. (2019) proposed LSTM/BiLSTM and the TC-BiLSTM as some trivial deep learning methods. Besides, Fan et al. (2019) combined BiLSTM and Distance-rule method to complete TOWE in a pipelined way, which is named as Pipeline in the experiments.

- **TOWE models**: IOG is the first TOWE model proposed by Fan et al. (2019). It adopts six different positional and directional LSTMs to extract the opinion words. PE-BiLSTM is the base model of the LOTN (Wu et al., 2020). They introduced target information of TOWE by position embedding and extracted opinion words with a BiLSTM. Wu et al. (2020) proposed an effective transfer learning method LOTN to identify latent opinions from the sentiment analysis model. Next, they integrated it with the PE-BiLSTM to achieve the state-of-the-art performance in TOWE.

- **AOPE model**: Aspect-opinion pair extraction (AOPE) task which aims at extracting aspects

Models	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Distance-rule	58.39	43.59	49.92	50.13	33.86	40.42	54.12	39.96	45.97	61.90	44.57	51.83
Dependency-rule	64.57	52.72	58.04	45.09	31.57	37.14	65.49	48.88	55.98	76.03	56.19	64.62
LSTM	52.64	65.47	58.34	55.71	57.53	56.52	57.27	60.69	58.93	62.46	68.72	65.33
BiLSTM	58.34	61.73	59.95	64.52	61.45	62.71	60.46	63.65	62.00	68.68	70.51	69.57
Pipeline	77.72	62.33	69.18	72.58	56.97	63.83	74.75	60.65	66.97	81.46	67.81	74.01
TC-BiLSTM	67.65	67.67	67.61	62.45	60.14	61.21	66.06	60.16	62.94	73.46	72.88	73.10
IOG	82.38	78.25	80.23	73.43	68.74	70.99	72.19	71.76	71.91	84.36	79.08	81.60
PE-BiLSTM	80.10	76.51	78.26	72.01	64.20	67.83	70.36	65.73	67.96	82.27	74.95	78.43
LOTN	84.00	80.52	82.21	77.08	67.62	72.02	76.61	70.29	73.29	86.57	80.89	83.62
SDRN+bert[‡]	91.14	76.37	83.10	84.37	65.42	73.69	83.57	70.33	76.38	91.13	80.34	85.40
Target-BiLSTM	84.00	79.34	81.58	75.35	**69.93**	72.50	76.95	71.62	74.14	87.85	80.99	84.24
ARGCN	86.67	**82.72**	**84.65**	79.45	**71.60**	**75.32**	76.57	**76.88**	**76.72**	86.16	**84.19**	85.16
Target-BiLSTM+bert	86.72	78.64	82.48	75.50	**72.84**	**74.15**	81.25	71.20	75.89	86.58	**84.76**	85.66
ARGCN+bert	87.32	**83.59**	**85.42**	75.83	**76.90**	**76.36**	78.81	**77.69**	**78.24**	88.49	**84.95**	**86.69**

Table 2: Main Experimental Results(%). Comparison between our proposed models and baselines on four benchmark datasets. P, R and F1 are *precision*, *recall* and *F1-score*, respectively. The result in bold indicates that the model outperforms all of the baselines above significantly ($p < 0.01$). The results are averaged scores of 10 runs. The results of baselines are copied from the previous work (Wu et al., 2020). Noted that the experiment results of SDRN are obtained by using their released codes to train and evaluate on TOWE datasets.

and opinion expressions in pairs, is a similar task as TOWE. SDRN, which is the state-of-the-art AOPE model proposed by Chen et al. (2020), mainly consists of an opinion entity extraction unit, a relation detection unit, and a synchronization unit. The synchronization unit could enhance the mutual benefit on the opinion entity extraction unit and a relation detection unit. As a baseline, it extracts the target and opinion. Subsequently, it collect the corresponding target-opinion pairs based on the predicted relations to complete the TOWE task.

- **Base model**: To show the effectiveness of the target-aware representation, we propose our base model, Target-BiLSTM. A BiLSTM receives the target-aware representation as the input and then predicts after a fully-conneceted layer and a softmax layer.

4.4 Results and Discussion

Table 2 shows the main experimental results of the baselines and our models on four benchmark datasets. We can observe that under the same condition of using GloVe for word representation, our base model Target-BiLSTM outperforms PE-BiLSTM with large improvements ranging from 3.31% to 6.18% on F1-score. Note that PE-BiLSTM uses position embedding. Instead, Target-BiLSTM introduces target embedding, which is the evidence of the effectiveness of our target-aware representation. Moreover, it not only performs similarly with LOTN on 14res and 14lap but also significantly outperforms LOTN on 15res and 16res, which introduces a large-scale sentiment analysis dataset for transfer learning. In contrast, our base model does not require additional resources except for the pre-trained word embeddings. Besides, our full model ARGCN outperforms Target-BiLSTM by a large margin on the four datasets. Therefore, we conclude that the syntactic information ARGCN encoded over the dependency graph is helpful to TOWE. Furthermore, ARGCN significantly outperforms LOTN, with large improvements of F1-score ranging from 1.54% to 3.43%, which proves its effectiveness on the TOWE task. With a pre-trained representation model, BERT, Target-BiLSTM achieves the state-of-the-art performance by significant margins, demonstrating the power of the pre-trained language model in this task. In addition, when we apply BERT as the representation layer for ARGCN, it achieves a further state-of-the-art performance, which demonstrates the effectiveness of capturing important syntactic information for sentiment analysis.

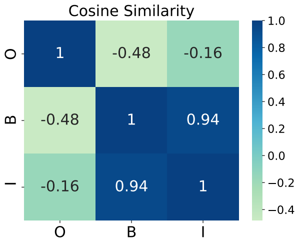


Figure 4: We train a Target-BiLSTM on 14res. After convergence, we collect the target embeddings and calculate their cosine similarities.

[‡] SDRN is an APOE model but evaluated on TOWE task.

Models	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
R-GCN (#basis=1)	85.23	81.67	83.40	76.24	70.26	73.12	74.08	72.74	73.35	86.25	80.76	83.39
GAT	81.30	74.27	77.63	66.73	65.43	66.07	68.91	70.59	69.74	81.80	72.76	77.02
RGAT	82.99	74.37	78.44	74.38	62.96	68.19	77.67	66.33	71.55	87.18	77.71	82.18
ARGCN (original)	84.79	82.59	83.65	77.79	70.79	74.06	74.83	74.06	74.43	85.64	83.47	84.53
ARGCN	86.67	82.72	84.65	79.45	71.60	75.32	76.57	76.88	76.72	86.16	84.19	85.16

Table 3: Ablation study results (%). LSTM-ARGCN denotes the model that places the BiLSTM before ARGCN. R-GCN (#basis=1) denotes using R-GCNs with basis decomposition and the number of basis is one. ARGCN (original) denotes using original dependency tree.

4.5 Visualization on Target Embedding

We also designed an experiment to evaluate if our model can learn suitable target embeddings during training, thereby it can benefit from the target-aware representation.

Intuitively, a good target embedding should have such a property: the representation of tag "O" is significantly different from that of tags "B" and "I". However, representations of " B" and "I" are similar. However, after training, as we can observe from Figure 4, the cosine similarity between "B" and "I" is close to 1 (0.94), whereas the similarity between "B" and "O" is even smaller than 0, and that between "I" and "O" has the same property. Therefore, we conclude that our model can learn to generate suitable target embeddings during training, which confirms the effectiveness and interpretability of our target-aware representation.

4.6 Ablation Study

To evaluate the influence of each component of ARGCN, we conducted an ablation study on ARGCN. As shown in Table 3, we observe performance drops on the four datasets when replacing ARGCN layers with R-GCN layers following Equations (2) and (3), which verifies the effectiveness of employing the distance attention mechanism in ARGCN. In addition, we also find that ARGCN outperforms GAT(Veličković et al., 2018), which proves that specifying the dependency relational type is crucial for applying the dependency graph to the TOWE task. Moreover, we compared ARGCN with an ABSA model, RGAT (Busbridge et al., 2019), which is similar to our model. We observe that our ARGCN performs much better than RGAT. These results prove that in TOWE task, the approach to encode syntactic information in ARGCN is more suitable than the approach used in RGAT.

In addition, to confirm that the syntactic graph that we constructed is effective and reasonable, we compare the ARGCN over the original dependency tree and our reshaped graph. The results show that the latter model outperforms the former one, which proves the effectiveness of our reshaped syntactic dependency graph.

4.7 Model Analysis

We further analyzed the effect of the layers number of ARGCN, the number of attention heads and the threshold of relative distance in our model by using different hyper-parameters but keeping the other hyper-parameters unchanged as the experimental settings mentioned above.

Because ARGCN involves an L-layer GCN, we investigated the effect of the layers number L with the final performance of ARGCN. Basically, we varied the value of L in the set $\{2, 4, 6, 8, 10, 12\}$ and showed the corresponding F1-score of ARGCN on the 14res dataset.

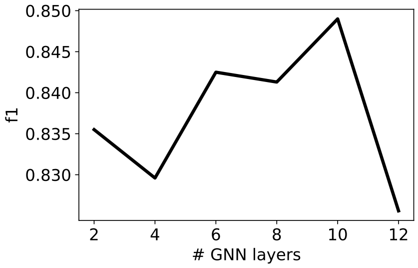


Figure 5: Effect of the number of GCN layers.

The results are illustrated in Figure 5, which shows that ARGCN achieves the best performance when $L = 10$. In this sense, our model can benefit from the increasing number of layers. However, when the number of layers is larger than 10, our model will tend to be over-smoothing which makes the performance drop dramatically.

As for the effect of the number of attention heads, we also varied the value of attention head number K in the set $\{1, 2, 4, 6, 8, 10, 12, 14\}$ and showed the corresponding F1-score of ARGCN on the 14res dataset.

Models	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Dependency-rule	64.57	52.72	58.04	45.09	31.57	37.14	65.49	48.88	55.98	76.03	56.19	64.62
RGCN (syntax only)	78.03	67.57	72.42	69.61	52.91	60.12	71.69	64.71	68.02	77.48	72.76	75.05
RGAT (syntax only)	78.59	61.65	69.10	61.87	45.50	52.44	68.09	58.42	62.88	74.62	65.52	69.78
ARGCN (syntax only)	78.51	71.65	74.92	68.23	56.44	61.78	74.36	65.31	69.55	82.11	72.57	77.05

Table 4: Evaluation of syntactic information (%). *Syntax only* means these models use only target embedding without word representation.

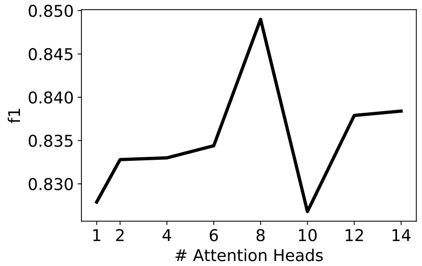


Figure 6: Effect of the number of Attention Heads.

The results are illustrated in Figure 6, which shows that ARGCN achieves the best performance when $K = 8$, which justifies the selection on the number of attention heads in the experimental settings. Comparing with the cases between $K = 1$ and $K = 8$, we found that the model with 8 attention heads performed better than that with only one attention head. This experiment demonstrated the necessity of the multi-head attention mechanism in our ARGCN.

As for the effect of threshold of relative distance, we performed experiments with different threshold D ranging from 1 to 6 and showed the corresponding F1-score of ARGCN on the 14res dataset.

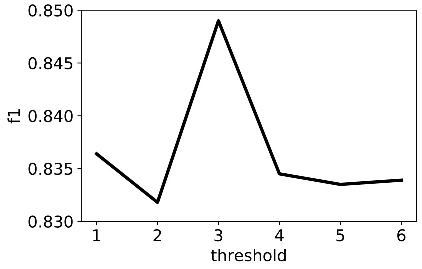


Figure 7: Effect of threshold.

The results are illustrated in Figure 7, where we observe that $D = 3$ is best value for the threshold in ARGCN.

4.8 Evaluation of syntactic information

To understand the role of syntactic information in the TOWE task and measure the ability of ARGCN to encode syntactic information, we removed word representation from our models, leaving the target embedding only. We performed some experiments on evaluating our model on the TOWE task only with syntactic information and position information. The results are shown in Table 4.

We notice that the GNN models perform better than the dependency-rule model, which indicates that the GNN models can exploit the syntactic information well from dependency graph. Furthermore, our well-designed ARGCN outperforms other GNN models including the latest one, RGAT. The reason is that ARGCN considers the relative position of words in the sentence and dependency relation type at the same time when it propagate the information.

5 Conclusions

In this paper, we proposed a target-aware representation to efficiently introduce opinion target information to our TOWE model. Moreover, we proposed ARGCN by extending the R-GCNs with a distance-aware attention mechanism. Because the sequential information is essential for such a sequence-labelling task, we captured the sequential information with BiLSTM after ARGCN layers and then completed the TOWE task. Empirical results show that our model significantly outperforms all baselines, including state-of-the-art, with large margins, which strongly proves the effectiveness of our model. The extensive analysis also demonstrated the effectiveness and necessity of all components in our model. In addition, we found that GNN model, especially a well-designed GNN model, such as ARGCN, is suitable for encoding syntactic information. We hope that these findings can be insightful for other researchers in the community.

References

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. 2019. Relational graph attention networks.

Alberto Cetoli, Stefano Bragaglia, Andrew D O'Harney, and Marc Sloan. 2017. Graph convolutional networks for named entity recognition. *arXiv preprint arXiv:1709.10053*.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524, Online. Association for Computational Linguistics.

Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. 2018. Iterative visual reasoning beyond convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7239–7248.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Nicola De Cao and Thomas Kipf. 2018. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2509–2518.

Victor Garcia and Joan Bruna Estrach. 2018. Few-shot learning with graph neural networks. In *6th International Conference on Learning Representations, ICLR 2018*.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1373–1378.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single and cross-domain setting with conditional random fields. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1035–1045.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Yingju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 653–661.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2886–2892.

Yunlong Liang, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2020. A dependency syntactic knowledge augmented interactive architecture for end-to-end aspect-based sentiment analysis. *arXiv preprint arXiv:2004.01951*.

Kang Liu, Heng Li Xu, Yang Liu, and Jun Zhao. 2013. Opinion target extraction using partially-supervised word alignment model. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1346–1356. Association for Computational Linguistics.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443, Lisbon, Portugal. Association for Computational Linguistics.

Diego Marcheggiani, Joost Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 486–492.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5683–5692.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan. The COLING 2016 Organizing Committee.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*. Accepted as poster.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. *arXiv preprint arXiv:2004.12362*.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. 2019. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):1–12.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020. Latent opinions transfer network for target-oriented opinion words extraction. *arXiv preprint arXiv:2001.01989*.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. *arXiv preprint arXiv:1909.03477*.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, page 43–50, New York, NY, USA. Association for Computing Machinery.

“Laughing at you or with you”: The Role of Sarcasm in Shaping the Disagreement Space

Debanjan Ghosh*[1] **Ritvik Shrivastava***[2,3] **Smaranda Muresan**[3]
[1]Educational Testing Service
[2]MindMeld, Cisco Systems
[3]Data Science Institute, Columbia University
`dghosh@ets.org`, `ritvshri@cisco.com`, {`rs3868`, `smara`}`@columbia.edu`

Abstract

Detecting arguments in online interactions is useful to understand how conflicts arise and get resolved. Users often use figurative language, such as sarcasm, either as persuasive devices or to attack the opponent by an ad hominem argument. To further our understanding of the role of sarcasm in shaping the disagreement space, we present a thorough experimental setup using a corpus annotated with both argumentative moves (agree/disagree) and sarcasm. We exploit joint modeling in terms of (a) applying discrete features that are useful in detecting sarcasm to the task of argumentative relation classification (agree/disagree/none), and (b) multi-task learning for argumentative relation classification and sarcasm detection using deep learning architectures (e.g., dual Long Short-Term Memory (LSTM) with hierarchical attention and Transformer-based architectures). We demonstrate that modeling sarcasm improves the argumentative relation classification task (agree/disagree/none) in all setups.

1 Introduction

User-generated conversational data such as discussion forums provide a wealth of naturally occurring arguments. The ability to automatically detect and classify argumentative relations (e.g., agree/disagree) in threaded discussions is useful to understand how collective opinions form, how conflict arises and is resolved (van Eemeren et al., 1993; Abbott et al., 2011; Walker et al., 2012b; Misra and Walker, 2013; Ghosh et al., 2014; Rosenthal and McKeown, 2015; Stede and Schneider, 2018). Linguistic and argumentation theories have thoroughly studied the use of sarcasm in argumentation, including its effectiveness as a persuasive device or as a means to express an ad hominem fallacy (attacking the opponent instead of her/his argument) (Tindale and Gough, 1987; van Eemeren and Grootendorst, 1992; Gibbs and Izett, 2005; Averbeck, 2013). We propose an experimental setup to further our understanding of the role of sarcasm in shaping up the disagreement space in online interactions. The disagreement space, defined in the context of the dialogical perspective on argumentation, is seen as the speech acts initiating the difference of opinions that argumentation is intended to resolve (Jackson, 1992; van Eemeren et al., 1993). Our study is based on the Internet Argument Corpus (IAC) introduced by Abbott et al. (2011) that contains online discussions annotated for the presence/absence and the type of an argumentative move (agree/disagree/none) as well as the presence/absence of sarcasm. Consider the dialogue turns from IAC in Table 1, where the **current turn** (henceforth, ct) is a sarcastic response to the

Arg. Rel.	Turn Pairs
Agree	**Prior Turn:** Today, no informed creationist would deny natural selection. **Current Turn:** Seeing how this was proposed over a century and a half ago by Darwin, what took the creationists so long to catch up?
Disagree	**Prior Turn:** Personally I wouldn’t own a gun for self defense because I am just not that big of a sissy. **Current Turn:** Because taking responsibility for ones own safety is certainly a sissy thing to do?
	Prior Turn: I’m not surprised that no one on your side of the debate would correct you, but wolves and dogs are both members of the same species. The Canid species. **Current Turn:** Wow, you ’re even wrong when you get away from your precious Bible and try to sound scientific.
None	**Prior Turn:** The hand of God kept me from serious harm. Maybe He has a plan for me. **Current Turn:** You better hurry up . Are n’t you like 113 years old.

Table 1: Sarcastic turns that disagree, agree or have no argumentative relation with their prior turns.

---

*Equal Contribution.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1998–2010
April 19 - 23, 2021.

prior turn (henceforth, pt). These dialogue moves can be argumentative (agree/disagree) or not argumentative (none). The argumentative move can express agreement (first example) or disagreement (the second example is an undercutter, while the third example is an ad hominem attack). The fourth example, although sarcastic, it is not argumentative. It can be noticed that none of the current turns contain explicit lexical terms that could signal an argumentative relation with the prior turn. Instead, the argumentative move is being implicitly expressed using sarcasm.

We study whether modeling *sarcasm* can improve the detection and classification of *argumentative relations* in online discussions. We propose a thorough experimental setup to answer this question using feature-based machine learning approaches and deep learning models. For the former, we show that *combining* features that are useful to detect sarcasm (Joshi et al., 2015; Muresan et al., 2016; Ghosh and Muresan, 2018) with state-of-the-art argument features leads to better performance for the argumentative relation classification task (agree/disagree/none) (Section 5). For the deep learning approaches, we hypothesize that *multitask learning*, which allows representations to be shared between multiple tasks (e.g., here, the tasks of argumentative relation classification and sarcasm detection), lead to better generalizations. We investigate the impact of multitask learning for a dual Long Short-Term Memory (LSTM) Network with hierarchical attention (Ghosh et al., 2017) (Section 4.2) and BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), including an optional joint multitask learning objective with uncertainty-based weighting of task-specific losses (Kendall et al., 2018) (Section 4.3). We demonstrate that multitask learning improves the performance of the argumentative relation classification task for all settings (Section 5). We provide a detailed qualitative analysis (Section 5.1) to give insights into when and how modeling sarcasm helps. We make the code from our experiments publicly available.[1] The Internet Argument Corpus (IAC) (Walker et al., 2012b) can be found for public acess here:[2]

2 Related Work

Argument mining is a growing area of research in computational linguistics, focusing on the detection of argumentative structures in a text (see Stede and Schneider (2018) for an overview). This paper focuses on two subtasks: argumentative relation identification and classification (i.e., agree/disagree/none). Some of the earlier work on argumentative relation identification and classification has relied on feature-based machine learning models, focusing on online discussions (Abbott et al., 2011; Walker et al., 2012b; Misra and Walker, 2013; Ghosh et al., 2014; Wacholder et al., 2014) and monologues (Stab and Gurevych, 2014, 2017; Persing and Ng, 2016; Ghosh et al., 2016). Stab and Gurevych (2014) proposed a set of lexical, syntactic, semantic, and discourse features to classify them. On the same essay dataset, Nguyen and Litman (2016) utilized contextual information to improve the accuracy. Both Stab and Gurevych (2017) and Persing and Ng (2016) used Integer Linear Programming (ILP) based joint modeling to detect argument components and relations. Rosenthal and McKeown (2015) introduced sentence similarity and accommodation features, whereas Menini and Tonelli (2016) presented how entailment between text pairs can discover argumentative relations. Our argumentative features in the feature-based model are based on the above works (Section 4.1). We show that additional features that are useful in sarcasm detection (Joshi et al., 2015; Ghosh and Muresan, 2018) enhance the performance on the argumentative relation identification and classification tasks.

In addition to feature-based models, deep learning models have been recently used for these tasks. Potash et al. (2017) proposed a pointer network, and Hou and Jochim (2017) offered LSTM+Attention network to predict argument components and relations jointly, whereas (Chakrabarty et al., 2019) exploited adaptive pretraining (Gururangan et al., 2020) for BERT to identify argument relations. We use two multitask learning objectives (argumentative relation identification/classification and sarcasm detection), as our goal is to investigate whether identifying sarcasm can help in modeling the disagreement space. Majumder et al. (2019); Chauhan et al. (2020) used multitask learning for sarcasm & sentiment and sarcasm, sentiment, & emotion, respectively, where a direct link between the corresponding tasks is

---

[1] https://github.com/ritvikshrivastava/multitask_transformers
[2] https://nlds.soe.ucsc.edu/iac2

evident.

Finally, analyzing the role of sarcasm and verbal irony in argumentation has a long history in linguistics (Tindale and Gough, 1987; Gibbs and Izett, 2005; Averbeck, 2013; van Eemeren and Grootendorst, 1992). We propose joint modeling of argumentative relation detection and sarcasm detection to empirically validate sarcasm's role in shaping the disagreement space in online conversations.

While the focus of our paper is not to provide a state-of-the-art sarcasm detection model, our feature-based models, along with the deep learning models for sarcasm detection are based on state-of-the-art approaches. We implemented discrete features such as pragmatic features (González-Ibáñez et al., 2011; Muresan et al., 2016), diverse sarcasm markers (Ghosh and Muresan, 2018), and incongruity detection features (Riloff et al., 2013; Joshi et al., 2015). The LSTM models are influenced by Ghosh and Veale (2017); Ghosh et al. (2018), where the function of contextual knowledge is used to detect sarcasm. Lastly, transformer models such as BERT and RoBERTa have been used in the winning entries for the recent shared task on sarcasm detection (Ghosh et al., 2020). In our research, for both kinds of deep-learning models, the best results are obtained by using the multitask setup, showing that multitask learning indeed helps improve both tasks.

3 Data

Our $training$ and $test$ data are collected from the Internet Argument Corpus (IAC) (Walker et al., 2012a). This corpus consists of posts from conversations in online forums on a range of controversial political and social topics such as Evolution, Abortion, Gun Control, and Gay Marriage (Abbott et al., 2011, 2016). Multiple versions of IAC corpora are publicly available, and we use a particular subset, marked as IAC_{orig}, collected from Abbott et al. (2011). This consists of around 10K pairs of conversation turns (i.e., prior turn pt and the current turn ct) that were annotated using Mechanical Turk for argumentative relations (agree/disagree/none) and other characteristics such as sarcasm/non-sarcasm, respect/insult, nice/nastiness. Median Cohen's κ is 0.5 across all topics.

For agree/disagree/none relations the annotation was a scalar judgment on an 11 point scale [-5,5] where "-5" indicates a high disagreement move, "0" indicates none relation, and "5" denotes a high agreement move. We converted the scalar values to three categories: disagree (D) for values between [-5, -2], none (N) for values between [-1,1], and agree (A) for values between [2,5], where the scalar partitions ([]) follow prior work with IAC (Misra and Walker, 2013; Rosenthal and McKeown, 2015).

Arg. Rel.	Sarcasm	# of turns
A	S	315 (33%)
	NS	638 (67%)
D	S	2207 (57%)
	NS	1696 (43%)
N	S	2285 (44%)
	NS	2841 (56%)

Table 2: Dataset statistics; A (Agree), D (Disagree), N (None); S (Sarcasm), NS (Non-Sarcasm)

Each "current turn" that is part of a <pt,ct> pair is also labeled with a Sarcasm (S) or Non-Sarcasm (NS) label. Table 2 shows the data statistics in terms of argumentative relations (A/D/N) and sarcasm (S/NS). We split the dataset into $training$ (80%; 7,982 turn pairs), $test$ (10%; 999 turn pairs), and dev (10%; 999 turn pairs) sets where each set contains a proportional number of instances (i.e., 80% of 315 (=252) sarcastic turns (S) with argument relation label A (agree) appears in the training set). The dev set is used for parameter tuning.

4 Experimental Setup

We present the computational approaches to investigate whether modeling *sarcasm* can help detect argumentative relations. As our goal is to provide a comprehensive empirical investigation of sarcasm's role in argument mining rather than propose new models, we explore three separate machine learning approaches well-established for studying argumentation and figurative language. First, we implement a Logistic Regression method that exploits a combination of state-of-the-art features to detect argumentative relations as well as sarcasm (Section 4.1). Second, we present a dual LSTM architecture with hierarchical attention and its multitask learning setup (Section 4.2). Third, we discuss experiments using the pre-trained BERT models and our multitask learning architectures based on it (Section 4.3).

4.1 Logistic Regression with Discrete Features

We use a Logistic Regression (LR) model that uses both argument-relevant ($ArgF$) and sarcasm-relevant ($SarcF$) features. Unless mentioned, all features were extracted from the current turn ct.

Argument-relevant features ($ArgF$). We first evaluate the features that are reported as being useful for identifying and classifying argumentative relations: (a) *n-grams* (e.g., unigram, bigram, trigram) created based on the full vocabulary of the IAC corpus; (b) *argument lexicons*: two lists of twenty words representing agreement (e.g., "agree", "accord") and disagreement (e.g., "differ", "oppose"), respectively (Rosenthal and McKeown, 2015) (c) *sentiment lexicons* such as MPQA (Wilson et al., 2005) and opinion lexicon (Hu and Liu, 2004) to identify sentiment in the turns; (d) *hedge features*, since they are often used to mitigate speaker's commitment (Tan et al., 2016); (e) *PDTB discourse markers* because *claims* often start with discourse markers such as *therefore*, *so*. We discard markers from the temporal relation; (f) *modal verbs* because they signal the degree of certainty when expressing a claim (Stab and Gurevych, 2014); (g) *pronouns*, since they dialogically point to the previous speaker's stance; (h) *textual entailment*: captures whether a position expressed in the prior turn is accepted in the current turn (Cabrio and Villata, 2012; Menini and Tonelli, 2016)[3]; (i) *lemma overlap* to determine topical alignment between the prior and current turn (Somasundaran and Wiebe, 2010). We compute lemma overlap of noun, verbs, and adjectives between the turns, and (j) *negation* to extract explicit negation cues (e.g., "not", "don't") that often signal disagreement.

Sarcasm-relevant features ($SarcF$). As sarcasm-relevant features we use: (a) Linguistic Inquiry Word Count *(LIWC)* (Pennebaker et al., 2001) features to capture the linguistic, social, individual, and psychological processes; (b) measuring *sentiment incongruity*, that is, capturing the number of times the difference in sentiment polarity between the prior turn pt and the current turn ct occurs and number of positive and negative sentiment words in turns (Joshi et al., 2015); (c) *sarcasm markers* used by Ghosh and Muresan (2018), such as *capitalization*, *quotation marks*, *punctuation*, *exclamations* that emphasize a sense of surprisal, *tag questions*, *interjections* because they seem to undermine a literal evaluation, *hyperbole* because users frequently overstate the magnitude of an event in sarcasm, and *emoticons & emojis*, since they often emphasize the sarcastic intent.

We use *SKLL*, an open-source Python package that wraps around the Scikit-learn tool (Pedregosa et al., 2011). [4] We perform the feature-based experiment using the Logistic Regression model from Scikit-learn.

In the experimental runs, LR$_{ArgF}$ (i.e., model that uses just the $ArgF$ features) denotes the *individual* model and LR$_{ArgF+SarcF}$ (i.e., model that uses both $ArgF$ and $SarcF$ features) is the *joint* model.

4.2 Dual LSTM and Multitask Learning

LSTMs are able to learn long-term dependencies (Hochreiter and Schmidhuber, 1997) and have been shown to be effective in Natural Language Inference (NLI) research, where the task is to establish the *relationship* between multiple inputs (Rocktäschel et al., 2015). This type of architecture is often denoted as the *dual architecture* since one LSTM models the premise and the other models the hypothesis (in Recognizing Textual Entailment(RTE) tasks). Ghosh et al. (2018) used the dual LSTM architecture with hierarchical attention (HAN) (Yang et al., 2016) for sarcasm detection to model the conversation context, and we use their approach in this paper to model the current turn ct and the prior turn pt. HAN implements attention both at the word level and sentence level. The distinct characteristics of this attention is that the word/sentence-representations are weighted by measuring similarity with a word/sentence level context vector, respectively, which are randomly initialized and jointly learned during training (Yang et al., 2016). We compute the vector representation for the current turn ct and prior turn pt and concatenate vectors from the two LSTMs for the final softmax decision (i.e., A, D or N for argumentative relation detection). Henceforth, this dual LSTM architecture is denoted as $LSTM_{attn}$.

To measure the impact of *sarcasm* in argumentative relation detection, we use a multitask learning approach. Multitask learning aims to leverage useful information in multiple related tasks to im-

---

[3] We used the textual entailment toolkit (AllenNLP) (Gardner et al., 2017).

[4] https://pypi.org/project/skll/

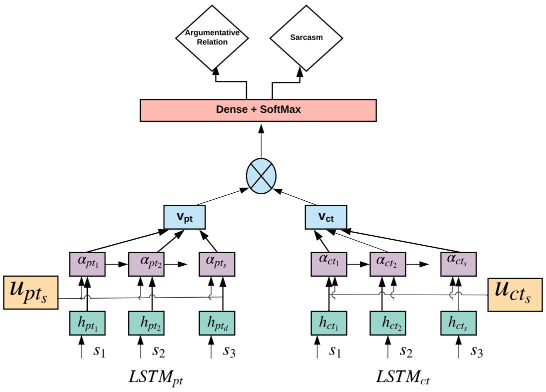


Figure 1: Sentence-level Multitask Attention Network for prior turn pt and current turn ct. Figure is inspired by Yang et al. (2016).

prove each task's performance (Caruana, 1997; Liu et al., 2019). We use a simple hard parameter sharing network. The architecture is a replica of the $LSTM_{attn}$, with a modification of employing two loss functions, one for sarcasm detection (i.e., training using the S and NS labels) and another for the argumentative relation classification task (i.e., training using the A, D, and N labels).

Figure 1 shows the high-level architecture of the dual LSTM and multitask learning ($LSTM_{MT}$). The prior turn pt (left) and the current turn ct (right) are read by two separate LSTMs (i.e., $LSTM_{pt}$ and $LSTM_{ct}$). In case of $LSTM_{MT}$ the concatenation of v_{pt} and v_{ct} is passed through a dense+Softmax layer for the MTL as shown in Figure 1. Similar to the LR models, $LSTM_{attn}$ now represents the *individual* model (i.e., predicts only the argumentative relation) whereas $LSTM_{MT}$ represents the *joint* model.

Dynamic Multitask Loss. In addition to simply adding the two losses, we also employed *dynamic weighting* of task-specific losses during the training process, based on the homoscedastic uncertainty of tasks, as proposed in Kendall et al. (2018):

$$L = \sum_t \frac{1}{2\sigma_t^2} L_t + \log \sigma_t^2 \qquad (1)$$

where L_t and σ_t depict the task-specific loss and its variance, respectively, over training instances. We denote this variation as $\text{LSTM}_{MT_{uncert}}$.

4.3 Pretrained BERT and Multitask Learning

BERT (Devlin et al., 2019), a bidirectional transformer model, has achieved state-of-the-art performance for many NLP tasks. BERT is initially trained on masked token prediction and next sentence prediction tasks over large corpora (English Wikipedia and Book Corpus). During its training, a special token "[CLS]" is added to the beginning of each training instance, and the "[SEP]" tokens are added to indicate the end of utterance(s) and separate, in case of two utterances (e.g., pt and ct). During the evaluation, the learned representation for the "[CLS]" token is processed by an additional layer with nonlinear activation. In its standard form, pre-trained BERT ("bert-base-uncased") can be used for transfer learning by fine-tuning on a downstream task, i.e., argument relation detection where training instances are labeled as A, D, and N. We denote the BERT baseline model as $BERT_{orig}$ that is fine-tuned over the $training$ partition of only the argumentative relation data (i.e., individual task training). Unless mentioned otherwise, we use the BERT predictions available via the "[CLS]" token. To this end, we propose a couple of variations in the multitask learning settings, and they are briefly described in the following sections.

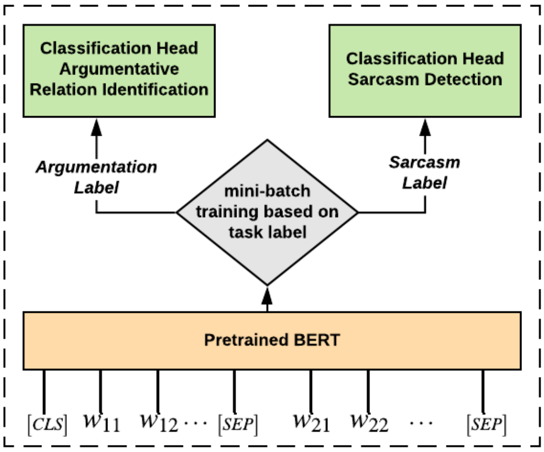


Figure 2: Alternating mini-batch training based on the task type ($BERT_{ALT}$).

Multitask Learning with BERT. The first model we use for multitask learning is denoted as $BERT_{MT}$ (i.e., BERT Multitask Learning). Here, we pass the BERT output embeddings to two classification heads - one for each task (i.e., detection of argumentative relation and sarcasm), and the relevant gold labels are passed to them. Each classification head is a linear layer (size=3 and 2 for # of labels for argumentative relation and sarcasm detection, respectively) applied on top of the pooled BERT output. The losses from these individual heads are added and propagated back through the model. This allows BERT to model the nuances of both tasks and their interdependence simultaneously.

Dynamic Loss: Similar to the LSTM architecture, here, too, we experiment with dynamic multitask loss. We denote this variation as BERT$_{MT_{uncert}}$.

Alternate Multitask Learning. We employ another multitask learning technique where we attempt to enrich the learning with fine-tuning of labeled *additional* material from the sarcasm detection task. Notably, we exploit "sarcasm V2", a sarcasm detection dataset that was also curated from the original corpus of IAC and was released by Oraby et al. (2016). We pre-process the "sarcasm V2" dataset by removing duplicates that appear in IAC_{orig} and we end up selecting 3513 $training_{v2}$ instances and 423 dev_{v2} instances balanced between S/NS categories for experiments and merged them to the sarcasm dataset ($training$ and dev, respectively) from IAC_{orig}. Note, unlike the original multitask setting, this time we have more sarcastic instances (a total of 11,495) than instances labeled with argumentative roles (7,982 instances as before) for the training purpose, while keeping the $test$ set from IAC_{orig} unchanged.

Since the $training$ data is now unequal between the two tasks of argumentative relation and sarcasm detection, we create mini-batches so that each batch consists of instances with only one task label (i.e., either argumentative labels or sarcasm labels). The batches from the two tasks are interleaved uniformly, i.e., the BERT model is only passed to one of the two tasks' specific classification heads, and the related loss is used to update the parameters in that iteration. This way, the model trains both tasks but alternates between the two tasks per mini-batch iteration while the extra batches of sarcasm data from the "sarcasm V2" dataset are managed at the end together. This model is denoted as $BERT_{ALT}$ (see Figure 2).

For brevity, all models' parameter tuning description (e.g., Logistic Regression, Dual LSTM, BERT) is in the supplemental material.

5 Results and Discussion

Table 3 presents the classification results on the $test$ set. We report F1 scores for each class (A, D and N) and Micro-F1 overall score (F1$_{micro}$) (used to account for multi-class and class imbalance).

Model	$F1_{micro}$	A	D	N
LR$_{ArgF}$	53.5	22.4	57.2	56.3
LR$_{ArgF+SarcF}$	**56.4**$^{\alpha^*}$	31.0	58.4	58.9
LSTM$_{Attn}$	51.8	28.0	49.4	59.2
LSTM$_{MT}$	53.1	30.0	53.2	56.5
LSTM$_{MT_{uncert}}$	**54.6** $^{\alpha^*}$	33.1	54.5	58.5
BERT$_{orig}$	62.2	41.8	63.3	64.4
BERT$_{MT}$	63.2	44.5	64.1	65.4
BERT$_{MT_{uncert}}$	**65.3**$^{\alpha^*}$	44.6	66.2	67.5
BERT$_{ALT}$	63.4	40.1	62.2	66.9

Table 3: Results for argumentative relation detection ($F1_{micro}$ and F1 scores/category) on the $test$ set of IAC_{orig}. $^{\alpha^*}$ depict significance on $p \leq 0.05$ (measured via Mcnemar's test) against the corresponding individual model (e.g., LR$_{ArgF}$, LSTM$_{Attn}$, BERT$_{orig}$, respectively). Highest scores per group of models are in **bold**.

The LR model using both the $SarcF$ and $ArgF$ features performs better than the model that uses $ArgF$ features alone, improving the overall performance by an absolute 2.9% F1$_{micro}$, and showing a huge impact on the agreement class (A) (8.6% absolute improvement). Table 4 shows the *top* discrete features for argumentative relation identification. From $ArgF$ features (first column), we notice discourse expansion ("particularly"), contrast ("although") and agree/disagree lexicon getting high feature weights. We also notice *pronouns* receive large feature weights because argumentative text often refers to personal stance (e.g., "you think", "I believe"). However, when analyzing $ArgF + SarcF$ features we find various sarcasm markers, such as tag questions, hyperbole, multiple punctuation, or sarcasm characteristics such as sentiment incongruity receive the highest weights.

For **LSTM models**, we see that multitask learning helps, LSTM$_{MT_{uncert}}$ showing a 2.8% improvement over the single model LSTM$_{Attn}$, which is statistically significant. Moreover, we notice that the improvement for the agree (A) and disagree (D) classes is 5.1%, with just a small reduction for the none (N) class (0.7%).

For **BERT**, we notice better results when performing multitask learning, while the best performing model is obtained from BERT$_{MT_{uncert}}$ where we experimented with the dynamic weighting of task-specific losses during the training process (Kendall et al., 2018). The performance increase is consistent across all three classes. The difference in performance among each setup is sta-

LR$_{ArgF}$	LR$_{ArgF+SarcF}$
pronouns: I. my (both A), your(s) (D); *discourse*: so, because, for (all A), incidentally, particularly, although (all D); *disagree_lexicon*: disagree, differ (both D);*agree_lexicon*: agreed (A); *entailment relation*; *negation* (D)	*pronouns*: mine, my (both A), you (D); *discourse*: then (A), though, however (both D); *modal*: will (A); *punctuation*: multiple question marks (both A and D); *tag question*: "are you", "do you" (both D); *hyperbole*: wonderful (A), nonsense, biased (both D); *LIWC dimensions*: anxiety, assent, certainty (all D); *sentiment incongruity* (D); *interj*: so, agreed (both A)

Table 4: Top discrete features from LR$_{ArgF}$ and LR$_{ArgF+SarcF}$ models, respectively. A and D depict the argumentative relations (agree and disagree) for the particular feature.

tistically significant, as shown in Table 3. Moreover, BERT$_{MT_{uncert}}$ model improves the $F1_{micro}$ by a large margin when compared to the LR and the LSTM models. However, adding more data for the auxiliary task (i.e., sarcasm detection) as presented in $BERT_{ALT}$ did not provide any significant improvement, only a 0.2 improvement of $F1_{micro}$ over $BERT_{MT}$ (however it does show improvement over the single task model). The reason could be that although "sarcasm V2"is a subset of the original IAC corpus, it was annotated by a different set of Turkers than IAC_{orig} with different annotation guidelines.

Between the three classes - A, D, and N - we observe the lowest performance on the A class. This is unsurprising, given the highly unbalanced setting of the $training$ data (A occurs less than 10% of times in the IAC_{orig}, see Table 2).

In sum, these improvements through multitask learning over single task argumentative relation detection indicate that modeling sarcasm is useful in modeling the disagreement space in online discussions. This provides an empirical justification to existing theories that study sarcasm's impact in modeling argumentation, persuasion, and argument fallacies such as ad hominem attacks. Finally, we notice that multitask learning also improves the performance on the sarcasm detection task (results are presented in the Appendix).

5.1 Qualitative Analysis

To further investigate the effect of multitask learning, we present qualitative analysis studies to:

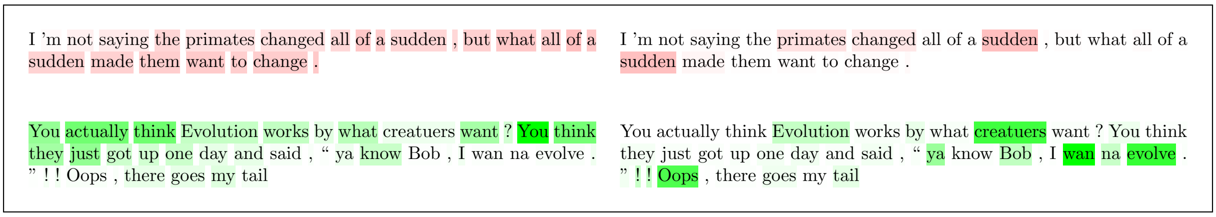
1. Understand the models' performance by looking at the turns correctly classified by the multitask models and misclassified by the corresponding individual single task model. We analyze the turns in terms of sarcastic characteristics - whether they depict incongruity, humor, or sarcasm indicators (i.e., markers).
2. Understand when both multitask and individual model made incorrect predictions.

We compare the predictions between the multitask and the individual models for different settings to address the first issue. For example, $BERT_{MT_{uncert}}$ correctly identifies 6 A, 50 D, and 60 N instances more than $BERT_{Orig}$ (out of 91, 398 and 510 instances, respectively). Two of the authors independently investigated a random sample of 100 instances ($qual$ set) chosen from the union of the $test$ instances that are correctly predicted only by the multitask models (LR$_{ArgF+SarcF}$, $LSTM_{MT_{uncert}}$, $BERT_{MT_{uncert}}$, and $BERT_{ALT}$) and not by the corresponding individual models (LR$_{ArgF}$, $LSTM_{attn}$, and $BERT_{Orig}$). For both Transformer and LSTM-based models, we explore how attention heads behave and whether common patterns exist (e.g., attending words with opposite meaning when incongruity occurs). We display the heat maps of the attention weights for a pair of prior and current turns (LSTM-based models) (Figure 3) whereas for BERT we display word-to-word attentions (Figures 4, 5, 6, 7, and 8) using visualization tools (Vig, 2019; Yang and Zhang, 2018).[5] All the examples presented in this section are argumentative moves (i.e., turns with A or D) correctly identified by our multitask learning models but wrongly predicted as none (N) by the individual models. Moreover, the multitask learning models also correctly predict that these turns are instances of sarcasm.

Incongruity between prior turn and current turn. Semantic incongruity, which can appear between conversation context pt and the current turn ct is an inherent characteristic of sarcasm (Joshi et al., 2015). This characteristic highlights the inconsistency between *expectations* and *reality*, making sarcasm or irony highly effective in persuasive communication (Gibbs and Izett, 2005).

---

[5] Clark et al. (2019) have probed different layers and attention heads in BERT to find patterns, e.g., whether a token consistently attends a fixed token in a specific layer. To avoid confusion and bias, we select attention examples from only the middle (layer=6) layer.



I 'm not saying the primates changed all of a sudden , but what all of a sudden made them want to change .

You actually think Evolution works by what creatuers want ? You think they just got up one day and said , " ya know Bob , I wan na evolve . " ! ! Oops , there goes my tail

I 'm not saying the primates changed all of a sudden , but what all of a sudden made them want to change .

You actually think Evolution works by what creatuers want ? You think they just got up one day and said , " ya know Bob , I wan na evolve . " ! ! Oops , there goes my tail

Figure 3: Attention heatmap of a particular turn pair from $LSTM_{attn}$(left) and LSTM$_{MT_{uncert}}$(right) showing higher weights on sarcasm marker such as "Oops" and "!!" for LSTM$_{MT_{uncert}}$ (disagree relation)

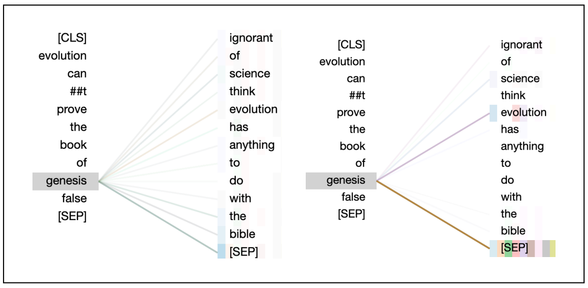


Figure 4: $BERT_{MT_{uncert}}$ (right) attending contrasting words more in word-level attention in comparison to $BERT_{Orig}$ (left) (disagree relation)

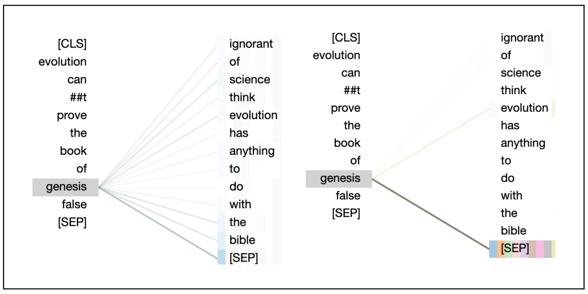


Figure 5: $BERT_{ALT}$ (right) attending only contrasting words in comparison to $BERT_{Orig}$ (left) (disagree relation). However, the strength of the contrast in the case of $BERT_{ALT}$ is lower than $BERT_{MT_{uncert}}$ for the same example turns.

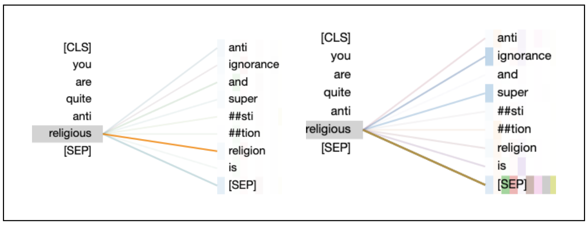


Figure 6: $BERT_{MT_{uncert}}$ (right) attending contrasting words more than $BERT_{Orig}$ (left) (disagree relation)

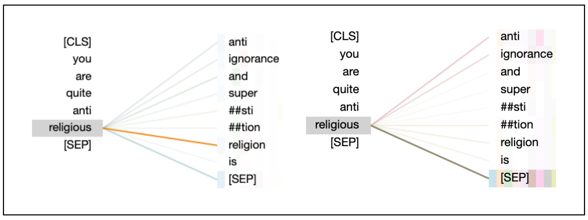


Figure 7: $BERT_{ALT}$ (right) attending only the contrasting words in comparison to $BERT_{Orig}$ (left) (disagree relation)

In the case of BERT, Figure 4 presents the turns "evolution can't prove the book of genesis false" (*pt*) ↔ "ignorant of science think evolution has anything to do with the bible" (*ct*). Here, $BERT_{MT_{uncert}}$ shows more attention between incongruous terms ("genesis" ↔ "science", "evolution") as well as to the mocking word "ignorance". Likewise, Figure 6 presents two turns "you are quite anti religious it seems" (*pt*) ↔ "anti ignorance and superstition . . . this is religion" (*ct*). We notice the word "religious" is attending "anti" and "ignorance" with high weights in case of $BERT_{MT_{uncert}}$ (from *pt* to *ct*) whereas $BERT_{Orig}$ only attends to the word "religious" from the *pt* to *ct* turn. By modeling sarcasm, the multitask learning models can better predict argumentative moves that are expressed implicitly.

We also evaluate the $BERT_{ALT}$ model for the examples presented in Figure 4 and Figure 6. Figure 5 shows that although $BERT_{ALT}$ is attending (from *pt* to *ct*) incongruous terms "genesis" ↔ "evolution", the strength of the relation (i.e., attention weight) is comparatively lower than $BERT_{MT_{uncert}}$ (See Figure 4). On the contrary, between Figure 6 and Figure 7, $BERT_{MT_{uncert}}$ model is attending multiple words in *ct* from the word "religion" in *pt*, but the $BERT_{ALT}$ model attends only two words 'anti" and "ignorance", with high weights from "religion" (*pt* to *ct*).

Humor by word repetition. Often the current turn *ct* sarcastically taunts the prior turn *pt* by word repetition and rhyme, imposing a humorous comic effect, also regarded as the phonetic style of humor (Yang et al., 2015). For the pair, "genetics

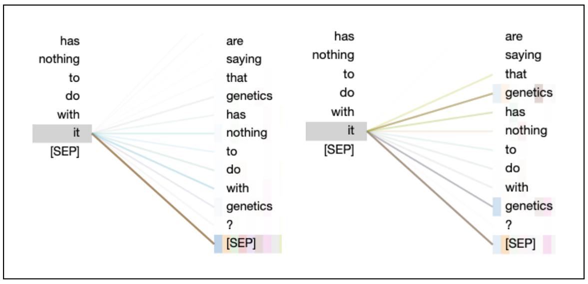


Figure 8: $BERT_{MT_{uncert}}$ (right) attending co-referenced words in a humorous example missed by the $BERT_{Orig}$ model (left) (disagree relation)

has nothing to do with it" (pt) ↔ "are saying that genetics has nothing to do with genetics?" (ct), we notice in $BERT_{MT_{uncert}}$ the token "it" in pt correctly attends to both occurrences of "genetics" in ct where the second occurrence is the co-reference of "it" (Figure 8), which is missed by the individual model $BERT_{Orig}$.

Role of sarcasm markers. Sarcasm markers are indicators that alert if an utterance is sarcastic (Attardo, 2000). While comparing the logistic regression models between LR$_{ArgF+SarcF}$ and LR$_{ArgF}$, we observe markers such as multiple punctuations ("???"), tag question ("are you"), upper case ("NOT") have received the highest features weights (Table 4). In Figure 3, while the individual model $LSTM_{attn}$ attends the words almost equally, we notice in the multitask variation several sarcasm markers such as "ya", "oops", and numerous exclamations ("!!") receive larger attention weights.

Addressing the second issue (i.e., when both multitask and single tasks models make the wrong predictions), we notice that over 100 examples of none (N) class were classified as argumentative by both $BERT_{MT_{uncert}}$ and $BERT_{Orig}$. For the none N class, one of the most common instances of wrong predictions is when the current turn ct sarcastically takes a "different stance" on a topic from pt in a narrow context but the whole turn is not argumentative. In the following example: "does he just say the opposite of everything <name> says?" (pt) ↔ "using <name> as a 180 compass is just fine by me" (ct), $BERT_{MT_{uncert}}$, $BERT_{Orig}$, LSTM$_{MT_{uncert}}$, and $LSTM_{attn}$ models make disagree D prediction (since ct is sarcastic on "<name>") where the gold label is none N. Looking closely at this pair of turns, it seems that the ct presents a case of ad hominem attack (on the person's "<name>") rather than a none relation.

In the case of argumentative turns (agree and disagree) that are wrongly classified as none by all models, we found two common patterns: the use of concessions (e.g., "it's a consideration, *but* I doubt we should be promoting this ...") and arguments with uncommitted beliefs (e.g., "it is *possible* that", "that could *probably* be", "*possibly*, I must admit").

6 Conclusion and Future Work

Linguistic and argumentation theories have studied the use of sarcasm in argumentation, including its effectiveness as a persuasive device or as a means to express an ad hominem fallacy. We present a comprehensive experimental study for argumentative relation identification and classification using sarcasm detection as an additional task. First, in discrete feature space, we show that sarcasm-related features, in addition to argument-related features, improve the accuracy of the argumentative relation identification/classification task by 3%. Next, we show that multitask learning using both a dual LSTM framework and BERT helps improve performance compared to the corresponding single model by a statistically significant margin. In both cases, the dynamic weighting of task specific losses performs best. We provide a detailed qualitative analysis by investigating a large sample manually and show what characteristics of sarcasm are attended to, which might have guided the correct prediction on the identification of the argumentative relation/classification task. In the future, we aim to study this synergy further by looking at sarcasm as well as the persuasive strategies (e.g., ethos, pathos, logos), and argument fallacies (e.g., ad hominem attack that was also noticed by Habernal et al. (2018)).

Acknowledgements

The authors thank the anonymous reviewers and Tuhin Chakrabarty for helpful comments.

References

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn Walker. 2016. Internet argument corpus 2.0: An sql schema for dialogic social media and the corpora to go with it. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4445–4452.

Rob Abbott, Marilyn Walker, Pranav Anand, Jean E Fox Tree, Robeson Bowmani, and Joseph King.

2011. How can you say such things?!?: Recognizing disagreement in informal political argument. In *Proceedings of the Workshop on Languages in Social Media*, pages 2–11. Association for Computational Linguistics.

Salvatore Attardo. 2000. Irony markers and functions: Towards a goal-oriented theory of irony and its processing. *Rask*, 12(1):3–20.

Joshua M. Averbeck. 2013. Comparisons of ironic and sarcastic arguments in terms of appropriateness and effectiveness in personal relationships. *Argumentation and Advocacy*, 50(1):47–57.

Elena Cabrio and Serena Villata. 2012. Combining textual entailment and argumentation theory for supporting online debates interactions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 208–212.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Tuhin Chakrabarty, Christopher Hidey, Smaranda Muresan, Kathy McKeown, and Alyssa Hwang. 2019. AMPERSAND: Argument mining for PERSuAsive oNline discussions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2933–2943, Hong Kong, China. Association for Computational Linguistics.

Dushyant Singh Chauhan, SR Dhanush, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Sentiment and emotion help sarcasm? a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4351–4360.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

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

Frans H. van Eemeren and Rob Grootendorst. 1992. *Argumentation, communication, and fallacies: a pragma-dialectical perspective*. Lawrence Erlbaum Associates, Inc.

Frans Hendrik van Eemeren, Rob Grootendorst, Sally Jackson, Scott Jacobs, et al. 1993. *Reconstructing argumentative discourse.* University of Alabama Press.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Aniruddha Ghosh and Tony Veale. 2017. Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 482–491.

Debanjan Ghosh, Alexander R Fabbri, and Smaranda Muresan. 2018. Sarcasm analysis using conversation context. *Computational Linguistics*, 44(4):755–792.

Debanjan Ghosh, Alexander Richard Fabbri, and Smaranda Muresan. 2017. The role of conversation context for sarcasm detection in online interactions. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 186–196, Saarbrücken, Germany. Association for Computational Linguistics.

Debanjan Ghosh, Aquila Khanam, Yubo Han, and Smaranda Muresan. 2016. Coarse-grained argumentation features for scoring persuasive essays. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 549–554, Berlin, Germany. Association for Computational Linguistics.

Debanjan Ghosh and Smaranda Muresan. 2018. "with 1 follower i must be awesome :p". exploring the role of irony markers in irony recognition. *Proceedings of ICWSM*.

Debanjan Ghosh, Smaranda Muresan, Nina Wacholder, Mark Aakhus, and Matthew Mitsui. 2014. Analyzing argumentative discourse units in online interactions. In *Proceedings of the First Workshop on Argumentation Mining*, pages 39–48.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 1–11, Online. Association for Computational Linguistics.

Raymond W Gibbs and Christin Izett. 2005. Irony as persuasive communication. *Figurative language comprehension: Social and cultural influences*, pages 131–151.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages

581–586, Portland, Oregon, USA. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. Before name-calling: Dynamics and triggers of ad hominem fallacies in web argumentation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 386–396, New Orleans, Louisiana. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yufang Hou and Charles Jochim. 2017. Argument relation classification using a joint inference model. In *Proceedings of the 4th Workshop on Argument Mining*, pages 60–66, Copenhagen, Denmark. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

S Jackson. 1992. Virtual standpoints' and the pragmatics of conversational argument. *Argumentation illuminated*, page 260–269.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Navonil Majumder, Soujanya Poria, Haiyun Peng, Niyati Chhaya, Erik Cambria, and Alexander Gelbukh. 2019. Sentiment and sarcasm classification with multitask learning. *IEEE Intelligent Systems*, 34(3):38–43.

Stefano Menini and Sara Tonelli. 2016. Agreement and disagreement: Comparison of points of view in the political domain. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2461–2470.

Amita Misra and Marilyn A Walker. 2013. Topic independent identification of agreement and disagreement in social media dialogue. In *Proceedings of the SIGDIAL 2013 Conference*, pages 41–50. Association for Computational Linguistics.

Smaranda Muresan, Roberto Gonzalez-Ibanez, Debanjan Ghosh, and Nina Wacholder. 2016. Identification of nonliteral language in social media: A case study on sarcasm. *Journal of the Association for Information Science and Technology*.

Huy Nguyen and Diane Litman. 2016. Context-aware argumentative relation mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1137.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. 2016. Creating and characterizing a diverse corpus of sarcasm in dialogue. In *17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 31.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71:2001.

Isaac Persing and Vincent Ng. 2016. Modeling stance in student essays. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2174–2184, Berlin, Germany. Association for Computational Linguistics.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, Copenhagen, Denmark. Association for Computational Linguistics.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 704–714. Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Sara Rosenthal and Kathleen McKeown. 2015. I couldn't agree more: The role of conversational structure in agreement and disagreement detection in online discussions. In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 168.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124. Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *EMNLP*, pages 46–56.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Manfred Stede and Jodi Schneider. 2018. Argumentation mining. *Synthesis Lectures on Human Language Technologies*, 11(2):1–191.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of WWW*.

Christopher W Tindale and James Gough. 1987. The use of irony in argumentation. *Philosophy & rhetoric*, pages 1–17.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Nina Wacholder, Smaranda Muresan, Debanjan Ghosh, and Mark Aakhus. 2014. Annotating multiparty discourse: Challenges for agreement metrics. *LAW VIII*, page 120.

Marilyn A Walker, Pranav Anand, Robert Abbott, and Ricky Grant. 2012a. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596. Association for Computational Linguistics.

Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012b. A corpus for research on deliberation and debate. In *LREC*, pages 812–817.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. 2015. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376.

Jie Yang and Yue Zhang. 2018. Ncrf++: An open-source neural sequence labeling toolkit. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*, pages 1480–1489.

A Supplemental Material

A.1 Parameter Tuning

Logistic Regression (LR) experiment: A Logistic Regression model with L_2 penalty is employed where the class weights are proportional to the number of instances for A, D and N classes. The regularization strength C is searched over a grid using the dev data. Following values were tried for c: [.0001, .001, .01, .1, 1, 10, 100, 1000, 10000].

Dual LSTM and Multi-task Learning experiment: For LSTM networks based experiments we searched the hyper parameters over the dev set. Particularly we experimented with different mini-batch size (e.g., 8, 16, 32), dropout value (e.g., 0.3, 0.5, 0.7), number of epochs (e.g., 40, 50), hidden state of different sized-vectors (100, 300) and the Adam optimizer (learning rate of 0.01). Embeddings were generated using FastText vectors (300 dimensions) (Joulin et al., 2016). Any token occurring less than five times were replaced by a special UNK token where the UNK vector is created based on random samples from a normal (Gaussian) distribution between 0.0 and 0.17. After tuning we use the following hyper-parameters for the $test$ set: mini-batch size of 16, hidden state of size 300, number of epochs = 50, and dropout value of 0.5. Task-specific losses for the dynamic multitask version was learned during training.

BERT based models: We use the dev partition for hyperparameter tuning such as different mini-batch size (e.g., 8, 16, 32, 48), number of epochs (3, 5, 6), learning rate of 3e-5) and optimized networks with the Adam optimizer. The training partitions were fine-tuned for 5 epochs with batch size = 16. Each training epoch took between 08:46 $\sim$ 9 minutes over a K-80 GPU with 48GB vRAM.

A.2 Results on the Sarcasm Detection Task

Although improving sarcasm detection is not the focus our paper, we observe that multi-task learning improves the performance on this task as well, when compared to the single task model. We present results for the deep learning models in Table 5. The multi-task models (both for LSTM and BERT) outperform the corresponding single task models (by 6.9 F1 and 6.4 F1 for LSTM and BERT models, respectively). We note that the results on this particular dataset are much lower than on other datasets used for sarcasm detection. For example, the LSTM$_{Attn}$ which is the best model used by Ghosh et al. (2018) obtained only 52.9 F1 score on this dataset, while it obtained 70.34 F1 on Sarcasm V2 (derived also from IAC but using different annotation guidelines), 74.96 F1 on a Twitter dataset and 75.41 F1 on a Reddit dataset (Ghosh et al., 2018).

Model	Precision	Recall	F1
LSTM$_{Attn}$	52.9	52.8	52.9
LSTM$_{MT}$	59.5	59.3	**59.4**
BERT$_{orig}$	57.4	57.4	57.4
BERT$_{MT}$	61.8	61.7	**61.8**
BERT$_{MT_{uncert}}$	64.1	63.5	**64.0**

Table 5: Evaluations of sarcasm detection on the $test$ set of IAC_{orig}.

Learning Relatedness between Types with Prototypes for Relation Extraction

Lisheng Fu[*] **Ralph Grishman**
Computer Science Department
New York University
New York, NY 10003, USA
`{lisheng, grishman}@cs.nyu.edu`

Abstract

Relation schemas are often pre-defined for each relation dataset. Relation types can be related from different datasets and have overlapping semantics. We hypothesize we can combine these datasets according to the semantic relatedness between the relation types to overcome the problem of lack of training data. It is often easy to discover the connection between relation types based on relation names or annotation guides, but hard to measure the exact similarity and take advantage of the connection between the relation types from different datasets. We propose to use prototypical examples to represent each relation type and use these examples to augment related types from a different dataset. We obtain further improvement (ACE05) with this type augmentation over a strong baseline which uses multi-task learning between datasets to obtain better feature representation for relations. We make our implementation publicly available: https://github.com/fufrank5/relatedness

1 Introduction

Relation extraction identifies specific semantic relationships between two entities within a single sentence. For example, there is a Physical.Located relationship between *George Bush* and *France* in the sentence: *George Bush* traveled to *France* on Thursday for a summit. Relation extraction is a crucial task for many applications such as knowledge base population.

Relation schemas are mostly pre-defined in existing datasets. The definition of the relation type depends on the annotation guide. There is no clear intrinsic Ontology for relation types. In practice, relation types can be created based on interests. This leaves datasets with similar, related or overlapping schemas. For example, the annotation guides for Automatic Content Extraction (ACE) 03-05 changed from year to year. The later created Entities, Relations and Events (ERE) dataset was similar in the schema, but differs in details. Because of the difficulty of annotating relations, these datasets are all small individually and hard to be utilized together.

It is not an easy task to learn relatedness between relation schemas across different datasets since there is no instance-level labels available for the relatedness. However, we can observe the connections between the relation types from different datasets based on relation names or annotation guides. We propose to simplify the relatedness as binary (related or not) and to use manual review of relation names to decide the relatedness labels. This would give the prior knowledge that one relation type in one dataset may have closer relationships to some types than the others in another dataset. Then we design a model to recognize this similarity. We propose to use prototypical examples to represent each relation type. We rank these representations higher for related types, and lower for unrelated types using a pairwise loss function. Our base model is a multi-task learning model which focuses on learning a strong encoder using multiple datasets regardless of the relation schemas. We take the step further to explore utilizing the relatedness between the relation types. Experiments on ACE05 and ERE show that it can further boost the performance, especially in the low-resource settings.

2 Related Work

Relation type dependency: There have been a few ways to model the relationships between types in a multi-label relation dataset where we can learn

---

[*] This is the work that the author has done before joining Amazon Alexa AI.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2011–2016
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

the similarity or dependency from annotated examples. Surdeanu et al. (2012) used a two-layer hierarchical model. The object-level classifier is able to capture the label dependency, while the mention-level classifier is focused on multi-label classification. Riedel et al. (2013) used a neighborhood model to explicitly model the dependency between the labels in a matrix factorization framework. Both of models are designed to work on multi-label examples, which require annotation to capture the dependency between labels. In the recent work of neural methods for relation extraction, most of the work (Zeng et al., 2015; Lin et al., 2016; Liu et al., 2017) ignores the multi-label setting and does not explicitly model the label dependency. Ye et al. (2017), on the other hand, ranks the similarity between feature representation of the instance and the label embedding. In addition to ranking the positive classes higher than the negative ones, it ranks positive classes against each other to learn the connections between the positives classes. These methods all require annotated examples to learn the connections. In the case of relation types across different datasets, such annotation does not exist. We attempt to learn the similarity nevertheless using prototypes from each type.

Multi-task learning: Training multiple relation datasets at the same time could improve the robustness of the model and reduce annotation cost for relation extraction. (Fu et al., 2018) proposed to use a shared encoder to learn more general feature representation. We use a similar multi-task learning base model and incorporate the similarity between the relation schemas to further improve the performance.

3 Relation Model with Multi-task Learning

The majority of neural relation models (Zeng et al., 2014; Nguyen and Grishman, 2015b; Zeng et al., 2015; Lin et al., 2016) encode a sentence using a deep architecture to a vector representation followed by a softmax classifier, while the others (dos Santos et al., 2015; Ye et al., 2017) use a function to compute the score between label embedding and sentence representation. Inspired by Fu et al. (2018) where the shared encoder helps in the case of the multi-task learning, we choose the latter so that all relation types (including from different datasets) will share the whole model parameters except the label embeddings. Suppose we obtain the sentence representation $\phi(x)$ with a neural architecture. We define the label embedding as $W_l \in \mathbb{R}^D$, a D-dimension vector for each type. We compute the L_1 distance between them and learn a scoring function to estimate the scores $S_\theta(x)$ for every type:

$$S_\theta(x)_l = W_o \cdot |\phi(x) - W_l| + b_o, \tag{1}$$

where $W_o \in \mathbb{R}^D$ and $b_o \in \mathbb{R}$ are shared for all types.

We do not use the dot product (dos Santos et al., 2015; Ye et al., 2017) as the scoring function because the L_1 distance works slightly better in the multi-task learning experiments. The probability of every class is computed as the softmax output of the scores. Similar to (Fu et al., 2018), we jointly train two relation tasks at the same time with cross-entropy losses.

$$L = \lambda L_{r1} + (1 - \lambda) L_{r2}, \tag{2}$$

where L_{r1} and L_{r2} are the cross-entropy losses for the two relation tasks. λ is the hyperparameter to control the learning speed between the two tasks. This would give a strong baseline of utilizing the two datasets together.

3.1 Prototypes of Relation Types for Learning Similarity

For each relation type, we randomly select k examples (S_k) from the training set as supporting examples. We use the mean of the representations of these examples as the prototype for the relation type:

$$\bar{x}_c = \frac{1}{k} \sum_{x_i \in S_k} \phi(x_i) \tag{3}$$

These prototypes are inspired by the Prototypical Networks (Snell et al., 2017). However, in the training procedure, these supporting examples are randomly selected for every mini-batch. We have dynamic prototypes during training.

We define $S_\theta(\bar{x}_c)_l$ as the similarity score to type c for type l. We hypothesize that if the two relation types are similar in semantics, they should obtain high similarity score. Within the dataset, if the relation types in the schema are mutually exclusive to each other, then we would expect a high similarity score to itself and low scores to the other types. Across the datasets, the prototypes would obtain high scores for related types and low scores in the unrelated types. We use a pair-wise ranking loss (dos Santos et al., 2015) to learn this relatedness across the datasets.

For $l \in L$ and $c \in C$, $S_\theta(\bar{x}_c)_l$ gives the score for the similarity between the type l in the relation schema L and the type c in the relation schema C. Let $c^+ \in C$ be a class related to l and $c^- \in C$ be a class unrelated to l. The similarity scores would be $S_\theta(\bar{x}_{c^+})_l$ and $S_\theta(\bar{x}_{c^-})_l$ respectively. We define the pair-wise ranking loss as:

$$\begin{aligned} L_s = log(1 + exp(\gamma(m^+ - S_\theta(\bar{x}_{c^+})_l)) \\ + log(1 + exp(\gamma(m^- + S_\theta(\bar{x}_{c^-})_l)) \end{aligned} \tag{4}$$

m^+ and m^- are the margins and γ is the scaling factor. This loss function would push $S_\theta(\bar{x}_{c^+})_l$ higher for related type pair between c^+ and l and $S_\theta(\bar{x}_{c^-})_l$ lower for unrelated type pair between c^- and l. We manually create a relatedness matrix to state whether the two types are related or not between the types in C and L based on the definition of the relation types. For each step of training, we pick the highest scored c^- from unrelated types and lowest scored c^+ for related types corresponding to type l.

$$c^- = \underset{c \in C^-}{argmax} S_\theta(\bar{x}_c)_l \tag{5}$$

$$c^+ = \underset{c \in C^+}{argmin} S_\theta(\bar{x}_c)_l \tag{6}$$

where C^- are types unrelated to l and C^+ are types related to l. In experiments, we use looser margins ($m^+ = 0.5$, $m^- = 0.5$, $\gamma = 1.0$) compared to (dos Santos et al., 2015) as we are learning the relatedness between types rather than doing classification for individual instance. The ranking loss is jointly trained as an auxiliary task with the main relation tasks:

$$L = \lambda L_{r1} + (1 - \lambda) L_{r2} + \beta L_s, \tag{7}$$

where we use β to control the weight for learning the relatedness. If β is too large, it could disrupt the learning of main relation tasks. With appropriate weight, it could help augment the label embeddings for the relation types by considering the similarity between them.

4 Experiments

4.1 Datasets

We select two datasets with similar relation schemas in this experiment. There is overlapping of relation types between ACE05 and ERE, but the annotation guides are different in details (Aguilar et al., 2014). Thus, we can not combine the training data directly as the same type. Doing so would actually lead to worse result as it introduces more noise than benefit. The multi-task learning would be a better choice at this setting. We take a step further and try to learn the similarity between the types at the same time. There are 6 main semantic types in ACE and 5 in ERE. Manual review of the relatedness (related or not) is trivial in this case because the relation names are almost identical for related relation types. In practice, it may take a few minutes to review more complicated relation schemas, but it would cost significantly less than annotating on the instance-level in text. For preprocessing the data, we follow previous work (Gormley et al., 2015; Nguyen and Grishman, 2015a; Fu et al., 2017, 2018) on ACE05. It contains 6 domains: broadcast conversation (bc), broadcast news (bn), telephone conversation (cts), newswire (nw), usenet (un) and weblogs (wl). We use newswire as training set (bn & nw), half of bc as the development set, and the other half of bc, cts and wl as the test sets. We followed their split of documents and their split of the relation types for asymmetric relations (directionality taken into account expect for *physcial* and *person-social* types). We perform the same preprocessing for the ERE dataset, which contains documents from newswire and discussion forums. We follow the document split from (Fu et al., 2018).

4.2 Multi-task Learning Baseline

Following previous work (Nguyen and Grishman, 2015a; Fu et al., 2018), We use a similar encoder to obtain the feature representation $\phi(x)$ as our baseline. The input layer is the concatenation of word embedding, entity embedding and position embeddings. We use pretrained word2vec (Mikolov et al., 2013) as the word embedding with embedding size d_w. The entity embedding and position embeddings are randomly initialized vectors according to the entity type of the token and relative distance to the two arguments of the relation. The embedding sizes are d_e and d_p respectively. We follow previous work for these input embedding sizes as $d_w, d_e, d_p = 300, 50, 50$. It is followed by Bidirectional RNN with attention and a fully connected layer to match the size for the label embedding. We use 150 for the RNN state size

	ACE05			
Method	**bc**	**wl**	**cts**	**avg**
Single-task	60.22	53.77	52.01	55.33
Multi-task	60.60	**56.20**	56.72	57.84
+ Relatedness	62.05	56.10	**59.12**	**59.08**
(Fu et al., 2018)	61.67	55.03	56.47	57.72
+ Regularization	**62.24**	55.30	56.27	57.94

Table 1: Learning the relatedness between types (full training set).

	ACE05			
Method	**bc**	**wl**	**cts**	**avg**
Single-task	54.80	47.27	48.42	50.17
Multi-task	56.67	51.39	55.23	54.43
+ Relatedness	**58.31**	**53.13**	**56.50**	**55.98**
(Fu et al., 2018)	57.39	51.44	54.28	54.37
+ Regularization	57.73	52.30	54.63	54.89

Table 2: Learning the relatedness between types (50% training).

and 200 for the label embedding size. The output of this encoder is $\phi(x)$. Then we can perform classification using the scores obtained from Equation 1.

In a mini-batch of training step, we randomly select examples from both datasets proportionally according to the dataset size so that the model can finish reading both datasets at the same time after every epoch. Because the difference of the number of examples for the two datasets in the batch, we set $\lambda = \lambda_d \frac{|D_1|}{|D_1|+|D_2|}$, where $|D_1|$ and $|D_2|$ are the number of examples for each dataset in a single batch. In a special case where the two datasets are actually split from one original dataset, we can set $\lambda_d = 1.0$, and then the two datasets are going to be learned at the same speed. In our case, we use $\lambda = 0.8$ so that the two relation tasks are roughly learning at the same speed. As the result, our multi-task model using label embedding is comparable to (Fu et al., 2018) (Table 1), which serves as a strong baseline since it is already better than training a single relation task. We obtain all our scores as the average of 10 runs to report stable results.

4.3 Learning the Relatedness between Two Relation Schemas

By learning the relatedness at the same time (Equation 4,7, $\beta = 0.001$), we obtain better re-

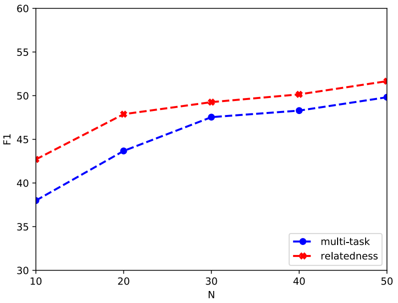


Figure 1: Low-resource setting with N examples for each positive relation type on ACE05.

sults at the full training set (Table 1). The improvement is more obvious with a smaller training set (Table 2 at 50%). The regularization in the previous work does not take the relatedness on the type-specific basis into account, which fails to obtain clear improvement over the multi-task baseline. Our method is more effective in incorporating additional knowledge from multiple sources. We also set up a low-resource setting where we only have N examples for each relation type (Figure 1 at $N = [10, 20, 30, 40, 50]$). The negatives are randomly selected according to the pos/neg ratio. We can observe larger improvement with less training data. This is also to consider the skewed data distribution in the dataset where there are far more examples for some types than the others. The k supporting prototype examples are drawn randomly at every step. We use $k = 50$ for the experiments and $k = N$ for the low-resource settings. Overall, the improvement is impressive, especially for the low-resource settings. It is also worth to note that the single task models for these low-resource settings obtain virtually zero scores without multi-task learning as there is not enough data to train the encoder. The multi-task learning between two relation tasks is better than training on a single task and more effective for a smaller training set. We now show that learning the relatedness between the types could further improve the model.

5 Conclusion

We use prototypes of relation types to learn the relatedness between them in a multi-task learning framework. With prior knowledge of relatedness between relation types, the model obtains further improvement in addition to sharing the encoder of the sentence. The prior knowledge is obtained through manual review of relation names, which costs significantly less than annotating on the instance-level in text. In this paper, we simplify the relatedness as binary. It would be interesting to further explore the relationships between relation types as a more dynamic metric.

References

Jacqueline Aguilar, Charley Beller, Paul McNamee, Benjamin Van Durme, Stephanie Strassel, Zhiyi Song, and Joe Ellis. 2014. A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards. In *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 45–53.

Lisheng Fu, Bonan Min, Thien Huu Nguyen, and Ralph Grishman. 2018. A case study on learning a unified encoder of relations. In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 202–207, Brussels, Belgium. Association for Computational Linguistics.

Lisheng Fu, Thien Huu Nguyen, Bonan Min, and Ralph Grishman. 2017. Domain adaptation for relation extraction with domain adversarial neural network. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–429, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Matthew R. Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1774–1784, Lisbon, Portugal. Association for Computational Linguistics.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795, Copenhagen, Denmark. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Thien Huu Nguyen and Ralph Grishman. 2015a. Combining neural networks and log-linear models to improve relation extraction. *arXiv preprint arXiv:1511.05926*.

Thien Huu Nguyen and Ralph Grishman. 2015b. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48, Denver, Colorado. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

Cícero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 626–634, Beijing, China. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Hai Ye, Wenhan Chao, Zhunchen Luo, and Zhoujun Li. 2017. Jointly extracting relations with class ties via effective deep ranking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1810–1820, Vancouver, Canada. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

I Beg to Differ:
A study of constructive disagreement in online conversations

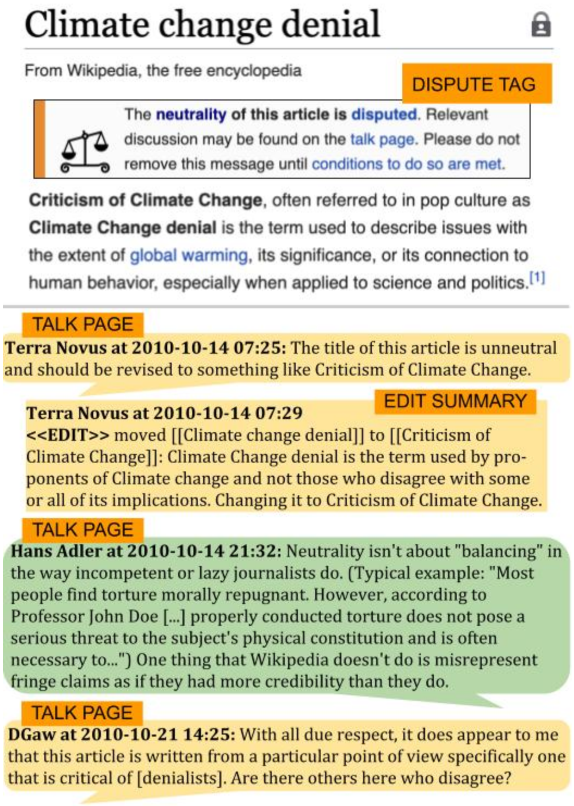
Christine De Kock and Andreas Vlachos
University of Cambridge
Department of Computer Science and Technology
`{cd700,av308}@cam.ac.uk`

Abstract

Disagreements are pervasive in human communication. In this paper we investigate what makes disagreement constructive. To this end, we construct WikiDisputes, a corpus of 7 425 Wikipedia Talk page conversations that contain content disputes, and define the task of predicting whether disagreements will be escalated to mediation by a moderator. We evaluate feature-based models with linguistic markers from previous work, and demonstrate that their performance is improved by using features that capture changes in linguistic markers throughout the conversations, as opposed to averaged values. We develop a variety of neural models and show that taking into account the structure of the conversation improves predictive accuracy, exceeding that of feature-based models. We assess our best neural model in terms of both predictive accuracy and uncertainty by evaluating its behaviour when it is only exposed to the beginning of the conversation, finding that model accuracy improves and uncertainty reduces as models are exposed to more information.

1 Introduction

Disagreements online are a familiar occurrence for any internet user. While they are often perceived as a negative phenomenon, disagreements can be useful; as is illustrated in Figure 1, disagreements can lead to an improved understanding of a topic by introducing and evaluating different perspectives. Research on online disagreements has focused on mostly negative aspects such as trolling (Cheng et al., 2017), hate speech (Waseem and Hovy, 2016), harassment (Yin et al., 2009) and personal attacks (Wulczyn et al., 2017). Recent works by Zhang et al. (2018) and Chang and Danescu-Niculescu-Mizil (2019) study conversations on Wikipedia talk pages that start out as civil but derail into personal attacks, using linguistic markers and deep learning approaches, respectively.



Climate change denial

From Wikipedia, the free encyclopedia

The **neutrality of this article is disputed**. Relevant discussion may be found on the talk page. Please do not remove this message until conditions to do so are met.

Criticism of Climate Change, often referred to in pop culture as **Climate Change denial** is the term used to describe issues with the extent of global warming, its significance, or its connection to human behavior, especially when applied to science and politics.[1]

Terra Novus at 2010-10-14 07:25: The title of this article is unneutral and should be revised to something like Criticism of Climate Change.

Terra Novus at 2010-10-14 07:29
<<EDIT>> moved [[Climate change denial]] to [[Criticism of Climate Change]]: Climate Change denial is the term used by proponents of Climate change and not those who disagree with some or all of its implications. Changing it to Criticism of Climate Change.

Hans Adler at 2010-10-14 21:32: Neutrality isn't about "balancing" in the way incompetent or lazy journalists do. (Typical example: "Most people find torture morally repugnant. However, according to Professor John Doe [...] properly conducted torture does not pose a serious threat to the subject's physical constitution and is often necessary to...") One thing that Wikipedia doesn't do is misrepresent fringe claims as if they had more credibility than they do.

DGaw at 2010-10-21 14:25: With all due respect, it does appear to me that this article is written from a particular point of view specifically one that is critical of [denialists]. Are there others here who disagree?

Figure 1: An example of a dispute on Wikipedia, showing the dispute tag on the article, talk page posts, and summaries of edits occurring during the discussion.

An alternative approach is to study good faith disagreement through debates on online platforms such as ChangeMyView (Tan et al., 2016), debate.org (Durmus and Cardie, 2019) and Kialo (Boschi et al., 2019), or formal Oxford-style debates (Zhang et al., 2016a). While this has benefits, such as readily available annotation of stances and indication of winning and losing sides, it does not mirror the way people naturally converse. For example, in formal debates, temporal and structural

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2017–2027
April 19 - 23, 2021.

constraints are imposed. Moreover, in Oxford-style debates, participants are not motivated to come to a consensus but rather to persuade an audience of a predefined stance (Zhang et al., 2016a). Finally, the task of detecting disagreement in conversations has been studied by a number of authors, e.g. Wang and Cardie (2014) and Rosenthal and McKeown (2015), without attempting to analyze it further.

We are interested instead in *constructive disagreement in an uncoerced setting*. To this end, we present `WikiDisputes`[1]; a corpus of 7 425 disagreements (totalling 99 907 utterances) mined from Wikipedia Talk pages. To construct it, we map dispute tags in the platform's edit history (shown in Figure 1) to conversations in WikiConv (Hua et al., 2018) to locate conversations that relate to content disputes. Observing that conversations are often conducted in the Talk pages and the edit summaries simultaneously, as illustrated in Figure 1, we augment the WikiConv conversations with edit summaries that occur concurrently. To investigate the factors that make a disagreement constructive, we define the task of predicting whether a dispute is eventually considered resolved by its participants, or it was escalated by them to mediation by a moderator, and therefore considered unconstructive.

We evaluate both feature-based and neural models for this task. Our findings indicate that balancing hedging and certainty plays an important role in constructive disagreements. We find that incorporating edit summaries (which have been ignored in previous work on Wikipedia discussions) improves model performance on predicting escalation. We further find that including information about the conversation structure aids model performance. We observe this in two ways. Firstly, we include gradient features, which capture changes in linguistic markers throughout the conversations as opposed to averaged values. Secondly, we experiment with adding sequential and hierarchical conversation structure to our neural models, finding that a Hierarchical Attention Network (Yang et al., 2016) provides the best performance on our task.

We further evaluate this model in terms of its predictive accuracy when exposed to the beginnings of the conversation as opposed to completed conversations, as well as its uncertainty (more details in Section 6.4). Our results indicate that model performance is reduced from a PR-AUC of 0.29 to 0.19 when exposed to only the first half of the conversation. However, this reduced model still outperforms toxicity and sentiment models predicting on full conversations. Model uncertainty decreases roughly linearly as the model is exposed to more information. We conduct a qualitative analysis of the points in the conversation where the model changes its prediction on constructiveness, finding that some markers from our feature-based models also seem to have been picked up by the neural model.

2 Dispute resolution on Wikipedia

Wikipedia relies heavily on collaboration and discussion by editors to maintain content standards, using the platform's Talk pages (Wikipedia, 2020c). Wikipedia contributors add "dispute" templates to articles (Wikipedia, 2020a) referring to a content accuracy dispute or to a violation of the platform's neutral point of view (NPOV) policy (Wikipedia, 2020b). Adding such a template to the article creates a dispute tag on the article as shown in Figure 1. Variations of these tags allow contributors to flag specific sections or phrases which relate to the issue in question, or to add details of the dispute.

All edits to Wikipedia, including the aforementioned dispute tags, are retained in the site's edit history. Revision metadata contains a description of the edit (hereafter referred to as the edit summary). As observed in Figure 1, when such edits occur while a conversation is underway, editors use the "edit summaries" to clarify their intentions in the context of the conversation.

The Wikipedia dispute resolution policy stipulates that if contributors are unable to reach a consensus through Talk page discussion, they can request mediation by a community volunteer. A prerequisite for this escalation step is proof of recent, extensive discussion on a Talk page[2].

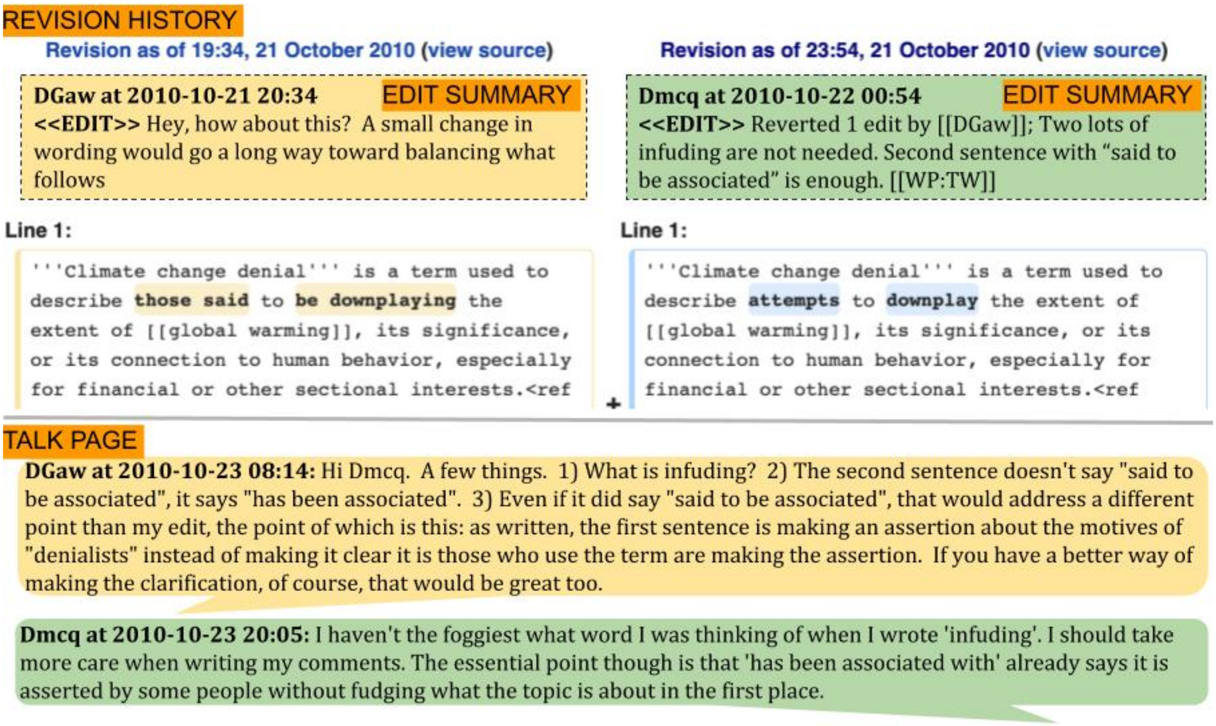
3 WikiDisputes

Our dataset consists of three facets: disagreements on Talk pages, edit summaries, and escalation labels. Wikipedia informs users that all contributions on article or Talk pages are published publicly, and provides channels for users to permanently remove personal information that they have shared accidentally or otherwise (Wikipedia, 2020d), thus we are able to release this data to the community. In

---

[1] `github.com/christinedekock11/wikidisputes`

[2] `https://en.wikipedia.org/wiki/Wikipedia:Dispute_resolution_noticeboard`


REVISION HISTORY

Revision as of 19:34, 21 October 2010 (view source)

DGaw at 2010-10-21 20:34 EDIT SUMMARY
<<EDIT>> Hey, how about this? A small change in wording would go a long way toward balancing what follows

Line 1:

```
'''Climate change denial''' is a term used to
describe those said to be downplaying the
extent of [[global warming]], its significance,
or its connection to human behavior, especially
for financial or other sectional interests.<ref
```

Revision as of 23:54, 21 October 2010 (view source)

Dmcq at 2010-10-22 00:54 EDIT SUMMARY
<<EDIT>> Reverted 1 edit by [[DGaw]]; Two lots of infuding are not needed. Second sentence with "said to be associated" is enough. [[WP:TW]]

Line 1:

```
+
'''Climate change denial''' is a term used to
describe attempts to downplay the extent of
[[global warming]], its significance, or its
connection to human behavior, especially for
financial or other sectional interests.<ref
```

TALK PAGE

DGaw at 2010-10-23 08:14: Hi Dmcq. A few things. 1) What is infuding? 2) The second sentence doesn't say "said to be associated", it says "has been associated". 3) Even if it did say "said to be associated", that would address a different point than my edit, the point of which is this: as written, the first sentence is making an assertion about the motives of "denialists" instead of making it clear it is those who use the term are making the assertion. If you have a better way of making the clarification, of course, that would be great too.

Dmcq at 2010-10-23 20:05: I haven't the foggiest what word I was thinking of when I wrote 'infuding'. I should take more care when writing my comments. The essential point though is that 'has been associated with' already says it is asserted by some people without fudging what the topic is about in the first place.

Figure 2: An illustration of the relationship between the revision history and Talk page conversations, using later utterances from the dispute in Figure 1. Two consecutive edits and their summaries are shown side by side.

the remainder of this section, we detail the process for obtaining the conversations, the edit summaries and the labels.

3.1 Finding disagreements

We use the Wikipedia revision history dump[3] from 1 July 2020 to find content disputes. We locate the addition of dispute templates in the article history using regular expressions that account for variations in the free text dispute templates. Using the revisions where dispute tags are added, we can also determine which user logged the dispute, the timestamp, and the article and section in dispute.

To find the related WikiConv conversation for each dispute tag, we select all conversations on the relevant article's Talk page in which the user who logged the dispute was involved. Of these candidates, the conversation closest in time to the timestamp of the dispute tag addition is selected. We filter the conversations extracted above using a number of heuristics in the pursuit of high quality samples of constructive disagreements. To ensure conversations of adequate length, we set a minimum conversation length of five utterances and 250 tokens. Based on an inspection of 100 conversations, we remove conversations of more than 50 utterances, as such conversations are often flame wars. We further include only conversations in which more than one user participated. Using this methodology, we obtain 7 425 content disputes.

Our manual inspection further indicates that the usage of the dispute tag is not entirely consistent across the platform; editors sometimes use the tag to report that they dispute some aspect of the text before discussing it, rather than to flag a currently occurring dispute in the Talk page. However, the former frequently results in a disagreement. We found that 88 out of 100 inspected examples were correctly identified disagreements, which we believe is an acceptably low error rate.

3.2 Collecting edit summaries

We show in Figure 2 two further posts from the climate dispute (continued from Figure 1), along with two edits that occurred during the course of the conversation. We note here that including the edit summaries as part of the conversation is important for understanding the conversation; without this, the reference to "infuding" in this example would be incomprehensible. We thus include all edit summaries written by users involved in the conversation, which are timestamped between the first and last utterance in the conversation.

---

[3] `https://dumps.wikimedia.org/enwiki/`

3.3 Escalation tags

For the task of inferring whether a dispute was escalated to mediation (and therefore Talk page discussion was unsuccessful), we scrape mediated cases from the Dispute Resolution Noticeboard archives and align them with WikiConv conversations.

Of the 2 520 archived mediation processes, 149 were closed as failures and 237 as successes, with the remaining 2 134 receiving "General closures". The last label is frequently assigned due to insufficient Talk page discussion. We only include accepted mediations in our dataset and thus ignore "General closures", resulting in 386 mediations.

We record the location of the disagreement to which the mediation relates, the usernames of participants and the timestamp. We use a similar alignment procedure to that described in Section 3.1 to find the relevant WikiConv conversations; except that we include the usernames of all participants listed in the mediation. During alignment, a simple match is recorded when there is exactly one conversation on the article Talk page involving all listed users (118 cases out of 386).

There are 111 cases in which no full matches occur; this sometimes happens when a participant changes their username or is participating anonymously. We manually inspect potential matches in this case. A further 67 cases have multiple matches; this happens when a participant changes their username, or is participating anonymously, or the Talk page has been archived; in this case we try to find the conversation most related to the mediation summary.

We use only the utterances which are timestamped from before the conversation was escalated and apply the same filters outlined in Section 3.1. Using this methodology, we are able to find 201 disagreements that are escalated to mediation. The complementary class label (not escalated) is assigned to the disagreements extracted using the dispute tags, for which no mediation was found.

Our classification task uses escalation as a proxy label for cases where Talk page discussion was not constructive. This relies on the assumption that non-escalated disagreements are more constructive than escalated disagreements, even though disagreements that were not escalated are not guaranteed to be successfully resolved; participants may simply not be aware of the escalation procedure, or not be willing to go through the extra effort of participating in mediation. To validate this, we annotated 35 samples from the dataset as to whether they represented constructive disagreements, and our annotations agreed with the escalation labels in 25 of these. Combined with the low estimates of antisocial behaviour on the site (Wulczyn et al., 2017), we draw the inference that the non-escalated class contains more constructive disagreements than the escalated ones. Throughout the remainder of this paper, we use these terms interchangeably.

4 Modelling constructive disagreement

4.1 Feature-based models

We develop our feature-based models using feature sets suggested in previous research on tasks related to conversational analysis. These include:

- **Politeness**: The politeness strategies from Zhang et al. (2018) as implemented in Convokit (Chang et al., 2020), which capture greetings, apologies, directness, and saying "please", etc.
- **Collaboration**: Conversation markers in collaborative discussions (Niculae and Danescu-Niculescu-Mizil, 2016), which capture the introduction and adoption of ideas, uncertainty and confidence terms, pronoun usage, and linguistic style accommodation.
- **Toxicity**: Toxicity and severe toxicity scores as estimated by the Perspective API (Wulczyn et al., 2017) and included in WikiConv (Hua et al., 2018).
- **Sentiment**: Positive and negative sentiment word counts, as per the lexicon of Liu et al. (2005) and implemented in Convokit (Chang et al., 2020).

For each feature, we calculate the average value throughout the conversation, as well as the gradient of a straight line fit of the feature value throughout the conversation. Our intuition for including this variation is that the outcome of a disagreement is likely to be influenced by a change in the language usage through the course of a conversation, which is not reflected by the mean. For instance, a disagreement might start out very politely and end impolitely, or the other way around, and would have the same mean. Logistic regression is used to infer linear relationships between the linguistic features and disagreement outcomes.

4.2 Neural models

Dialogue structure has been difficult to capture in feature based models due to sparsity. For this reason, we implement a number of neural models with increasing capacities for modelling conversation structure to assess its importance.

Averaged embeddings Our simplest variant averages the GloVe embeddings (Pennington et al., 2014) of all words in a conversation, and uses a fully connected layer for classification. It ignores both utterance hierarchy and word ordering and can be seen as a bag-of-word-embeddings approach.

LSTM Instead of averaging the GloVe embeddings, we use a bidirectional LSTM-based model (Hochreiter and Schmidhuber, 1997) to process the sequence of words in the conversation; ignoring utterance hierarchy but preserving word order.

HAN We use a Hierarchical Attention Network (HAN) (Yang et al., 2016) to model both word order and utterance hierarchy. HANs have recently been used to predict emotion in conversations (Ma et al., 2020) and are useful for capturing the structure of texts by building up utterance embeddings from word embeddings, and then constructing a conversation vector from these utterance embeddings. Given a sequence of words in an utterance, the words are first mapped to vectors through an embedding matrix (GloVe embeddings, in our case). A bidirectional LSTM layer (Hochreiter and Schmidhuber, 1997) is used to process the sequence of embeddings and calculate an embedding for each word in the context of the utterance. An attention mechanism (Bahdanau et al., 2015) is applied to these embeddings to find an utterance embedding. A similar procedure is followed to build up a conversation vector based on the utterances embeddings. This vector is then used to perform the classification. To incorporate edit summaries in a conversation, we prepend each edit summary with a special token (<EDIT>) to indicate its origin.

5 Experimental setup

An initial data analysis indicated that escalated disagreements have a median length of 16 utterances, compared to 11 for their non-escalated counterparts. While this may be an informative feature for classification, we are primarily interested in the effects of language on the conversation outcome, and therefore choose to control for this effect. We use *matching*, a technique developed for causal inference in observational studies (Rubin, 2007), also employed by Zhang et al. (2018) in the context of conversational modelling. We pair each escalated disagreement with a non-escalated disagreement of the same length in utterances, bearing in mind that the dataset is heavily imbalanced, with 7 425 non-escalated disagreements compared to 201 escalated disagreements. To retain an imbalance in the classes while conducting matching, we match every escalated disagreement with up to ten non-escalated disagreements (the actual number depending on availability) by randomly sampling without replacement from the non-escalated disagreements of the same length. Characteristics of the dataset after performing matching are shown in Table 1, with both classes having a median of 16 utterances now. Escalated disagreements have one more participant than non-escalated disagreements on average, and utterances in the escalated disagreements class are slightly shorter.

	Not escalated	**Escalated**
# Samples	1994	216
# Participants	2	3
# Utterances	16	16
Tokens per utt.	61	53

Table 1: Dataset statistics for the task of predicting escalation. Where applicable, median values are used.

We split the dataset for training as indicated in Table 2. Due to the class imbalance we use the area under the precision-recall curve (Davis and Goadrich, 2006) as metric for this task; however, for the sake of interpretability we also present the break-even F1 scores. We use a distribution-aware random class predictor as a random baseline.

Hyperparameters The logistic regression models are implemented in Scikit-Learn. We use a grid search to determine the best regularisation mode (L1 or L2) per model, evaluating C-values in [0.1,1,10,100]. The neural models are implemented in Keras, using Focal Loss (Lin et al., 2017) with Adam optimisation (Kingma and Ba, 2014) (learning rate=0.001) and Dropout ($p = 0.3$). For the HAN, we use bidirectional LSTM layers with 128 nodes in both the utterance and conversation encoders. For the LSTM model, we use only one such a layer.

Dataset	Escalated	Not escalated
Train	125	1411
Test	46	284
Validation	30	299

Table 2: Test set splits for predicting escalation.

Model	PR-AUC	F1
Baselines		
Random	0.121	0.128
Bag-of-words	0.213	0.239
Feature-based models		
Toxicity	0.140	0.125
Sentiment	0.150	0.055
Politeness	0.232	0.241
+ *gradients*	0.275	0.243
Collaboration	0.261	0.320
+ *gradients*	0.269	0.302
Politeness and collaboration	0.255	0.256
+ *gradients*	0.281	0.289
Neural models		
Averaged embeddings	0.243	0.256
LSTM	0.263	0.194
HAN	0.373	0.304
+ *edit summaries*	**0.400**	**0.333**

Table 3: Results of the escalation prediction task.

6 Results

Results are shown in Table 3. We note that generally, the neural network models perform better than the feature-based models. We discuss the PR-AUC scores of each model category below, noting that the scores from the F1 metric generally follow the same trends, but that PR-AUC is more robust in the face of imbalanced data. Furthermore, we investigate whether it is possible to predict the outcome from the beginning, how model uncertainty changes, or if there is often some inflection point in a conversation that changes the outcome.

6.1 Feature-based models

A number of our feature-based models outperforms the bag-of-words baseline, which in turn performs better than the random baseline. The toxicity and sentiment features do not perform better than the bag-of-words baseline, which indicates that these features by themselves do not capture constructiveness in disagreements. Toxicity scores were found by Zhang et al. (2016a) to be predictive of a conversation derailing, but in our case seemingly lead to the inference of spurious correlations. An explanation might be that comments which seem toxic out of context are in fact spirited discussion due to the degree of involvement of the participants; for instance, we have observed that contributors in some cases challenge others' credentials, which may seem rude in casual conversation, but can be useful in resolving a dispute.

Feature	Type	Coeff.
2nd person pronouns, $\bar{x}$	Both	+3.64
# hedging terms, $\bar{x}$	Both	-2.48
Greetings, $\bar{x}$	Politeness	+2.30
1st person pronouns, $\bar{x}$	Both	+1.91
Greetings, ∇	Politeness	-1.81
Deference, $\bar{x}$	Politeness	-1.44
3rd person pronouns, ∇	Collaboration	-1.38
# ideas adopted + certainty, $\bar{x}$	Collaboration	-1.23
Use of "by the way", $\bar{x}$	Politeness	-1.04
Certainty,∇	Collaboration	-0.92

Table 4: Ten features with the largest coefficients of the best feature-based model, which uses politeness and collaboration featuresets. Feature variations are the mean ($\bar{x}$) and gradient (∇). Positive weights are associated with the unconstructive class.

The best featuresets proposed in previous work are collaboration (Niculae and Danescu-Niculescu-Mizil, 2016), followed closely by politeness (Zhang et al., 2016a). Adding the gradient features we proposed improves performance for both, indicating that not only the presence of a marker is important, but also how its usage throughout a conversation changes. The best feature-based model is a combination of politeness and collaboration features, with a PR-AUC of 0.281 ($P < 0.05$, using a randomized permutation test).

The ten features with the largest coefficients from the combined model are shown in Table 4 in descending order or magnitude, along with their directions. Positive coefficients are associated with the escalated class, which indicates that a disagreement was not constructive.

Politeness markers such as deference and the use of "by the way" are found indicative of constructiveness, corroborating the findings of Zhang et al. (2016b). Greetings are associated with unconstructive disagreements on average, but an increase in greetings towards the end of a conversation is associated with constructiveness. An explanation of this might be that greetings can seem overly formal or indicative of tension early on, but if used later in a conversation, they indicate that new participants have entered the conversation or that more time is taken between replies. Indeed, longer gaps between replies (a feature in the collaboration featureset) are also associated with constructiveness.

The use of first and second person pronouns ("I" and "you") is associated with unconstructive disagreements, with the latter corroborating the findings of Zhang et al. (2018). This is also consistent with psychotherapy research on disagreements in relationships (e.g. Gottman and Krokoff (1989)), which emphasises avoiding 'you'-messages which might be perceived as blameful.

Hedging is associated with constructive disagreements, which corroborates the findings of Zhang et al. (2016a). An intuitive understanding of this result is that using hedging terms shows that the speaker is more open to adjusting their opinion or compromising. The certainty gradient and the adoption of new ideas with certainty are also associated with constructiveness. Niculae and Danescu-Niculescu-Mizil (2016) found no significant correlation between either certainty or hedging and successful collaboration. We attribute the observed differences in feature associations to the fact that WikiDisputes are sourced from an uncoerced setting, where users feel more invested in the outcome of a disagreement and may need to balance the use of certainty and hedging to negotiate compromises. The setting of Niculae and Danescu-Niculescu-Mizil (2016) obligates volunteers to participate in a photo geolocation game, where it is unlikely that interlocutors are as invested in the task as collaborators working on a widely read encyclopaedia.

Words associated with either class, obtained through the bag-of-words model, are shown in Appendix A. An interesting observation from this list is that citing Wikipedia policy (which is indicated with WP) is associated with escalating conversations. This practice is sometimes referred to as "Wiki-Lawyering" within the community and can signal an unwillingness to compromise.

6.2 Neural network models

The results from our neural models illustrate that incorporating structure improves predictive accuracy on our task.

The model that averages over word embeddings performs the worst; a moderate increase in performance (2%) is gained from adding sequential word processing by way of an LSTM model. Adding utterance boundary information with HAN results in a much larger improvement (11%) over the LSTM. This indicates that, while it is helpful to observe the ordering of words in a conversation, a more critical component in the case of disagreements is how words are arranged in utterances.

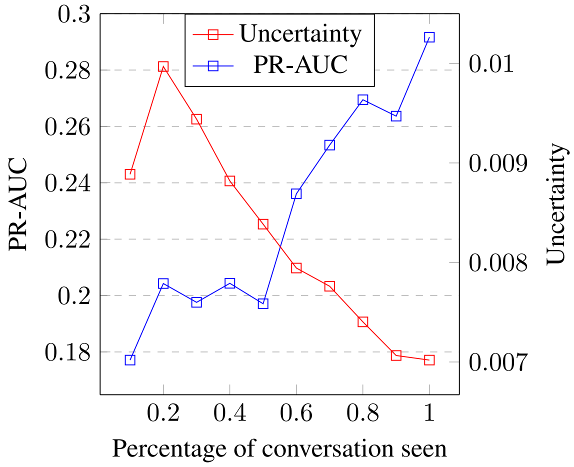


Figure 3: PR-AUC scores and uncertainty values of a HAN model when exposed to partial conversations.

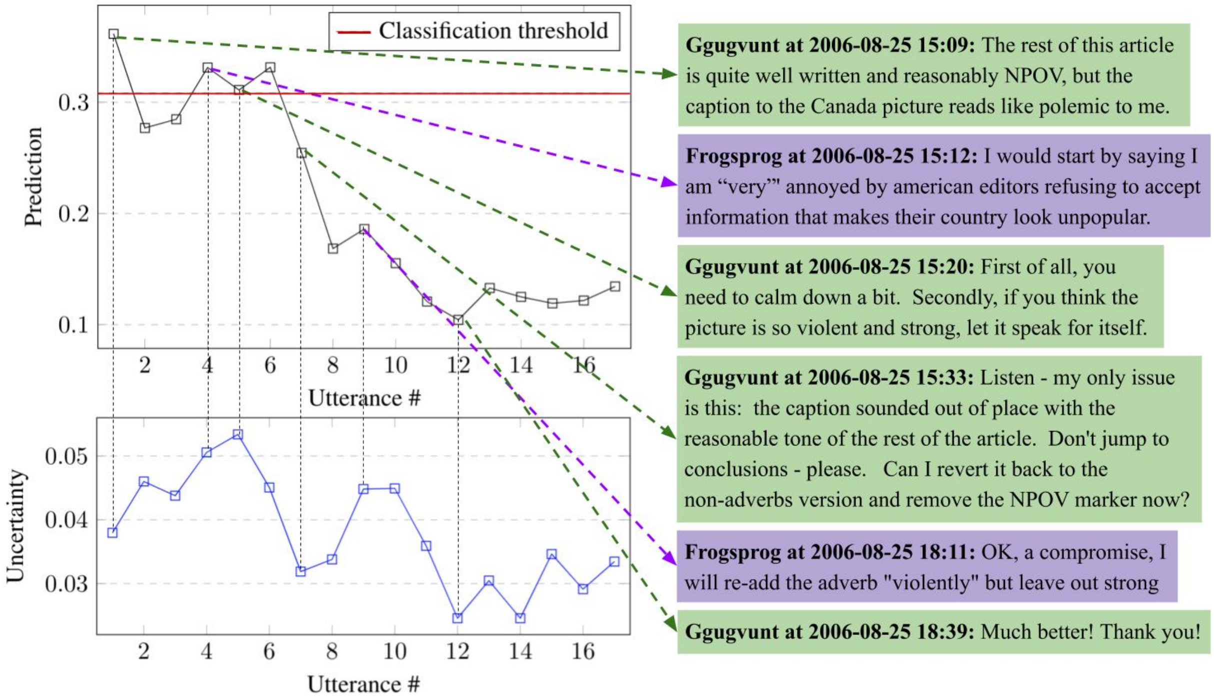
The highest scoring model, which a PR-AUC of 0.40, results from including edit summaries in the conversations ($P < 0.05$, using a randomized permutation test). This provides support for our observation that understanding some conversations requires knowledge of the edits that occurred during the conversation. However, neglecting to insert the "EDIT" token (as explained in Section 4.2) results in the PR-AUC decreasing to 0.33, which indicates that utterances sourced from edit summaries require special processing and are not completely homogeneous in the conversation.

Evaluating the predictions of this model, we observe that false positives often co-occur with heated debate. This is not unexpected, given that the positive class is contains interlocutors who feel strongly enough about their case to request mediation. The false negatives, on the other hand, contain a number of cases where it seemed as though a disagreement had been resolved, but then the argument progresses as further edits are made.

6.3 Early estimation of outcome

We are interested in how model predictions change throughout the conversation; whether it is possible to predict the outcome from the beginning, or if there is often some inflection point in a conversation that changes the outcome. Predicting the outcome of a conversation based on only a few utterances means that the model has less information for its inference; however, if models can perform well under such conditions, early intervention by a mediator could be recommended.

To evaluate model performance in such conditions, we split each disagreement into 10 buckets, chronologically (using only conversations of more than 10 utterances). We evaluate models trained on



Ggugvunt at 2006-08-25 15:09: The rest of this article is quite well written and reasonably NPOV, but the caption to the Canada picture reads like polemic to me.

Frogsprog at 2006-08-25 15:12: I would start by saying I am “very”' annoyed by american editors refusing to accept information that makes their country look unpopular.

Ggugvunt at 2006-08-25 15:20: First of all, you need to calm down a bit. Secondly, if you think the picture is so violent and strong, let it speak for itself.

Ggugvunt at 2006-08-25 15:33: Listen - my only issue is this: the caption sounded out of place with the reasonable tone of the rest of the article. Don't jump to conclusions - please. Can I revert it back to the non-adverbs version and remove the NPOV marker now?

Frogsprog at 2006-08-25 18:11: OK, a compromise, I will re-add the adverb "violently" but leave out strong

Ggugvunt at 2006-08-25 18:39: Much better! Thank you!

Figure 4: Model predictions and uncertainty throughout the course of a conversation in our dataset. The classification threshold shown is tuned on the validation set.

full conversations on these subsets to observe the change in model performance. These results are shown in Figure 3 in blue. Model performance increases from 0.198 at the midway point (0.5 bucket) to 0.292 when the full conversation has been observed, indicating that signals later in conversations are often important for predicting outcome. However, the neural model's performance when predicting on half the conversation only is still better than the toxicity and sentiment models on the complete conversations.

6.4 Uncertainty estimation

Given that this early estimation exposes the model to less information, we are interested in whether this impacts model uncertainty. We employ Monte Carlo dropout (Gal and Ghahramani, 2016) to this end. This approach calls for applying dropout before every layer of weights during both training and prediction, allowing us to sample from the distribution of predicted values for each input. Each input is evaluated multiple times (in our case, $N = 30$) and the mean and standard deviation of these predictions represent the prediction and its uncertainty.

Our results are shown in Figure 3 (in red). We note that uncertainty initially increases and then decreases monotonically. The initial increase in uncertainty could be due to a contrary position being introduced in the second utterance, in response to the introductory comment (as occurs in Figure 4). From there, the dispute is either resolved or eventually escalated, and uncertainty decreases almost linearly as the model is exposed to new data.

6.5 Identifying inflection points

Having ascertained that model accuracy improves and uncertainty decreases as a model is exposed to more information, we are interested in factors that cause the predicted class to change, and how the model predictions relate to the feature-based model coefficients. Although methods for interpreting neural network predictions exist (Ribeiro et al., 2016, 2018), these are not easily extendable to process inter-utterance dependencies as observed with the HAN. We instead analyse a conversation from our dataset to observe inflection points and what may have caused them. We show HAN model predictions and uncertainty values in Figure 4.

The HAN model initially predicts escalation, with a relatively low uncertainty value. The conversation remains confrontational for the first seven utterances with a high escalation score. There are two “I” pronouns in the second utterance and two “you” pronouns in the third utterance, which were also associated with unconstructiveness in the feature-based models. We note the use of politeness cues ("please" and "thank you" in the fourth and sixth utterances in this example), which reduces the score for escalation. The uncertainty and prediction values increase when one user proposes a compromise in utterance 9, and then decreases again in utterance 12 when the other user accepts the compromise. This feature was not explicitly modelled for in our feature-based models.

7 Conclusion

In a discussion of online disagreements by Graham (2008) (which Wikipedia references in its guidelines for resolving disputes), the author remarks that an increase in disagreements through widespread online interaction has the potential to create a surge in anger among internet users. On the other hand, as illustrated in our data, disagreements can sometimes be constructive. The success of Wikipedia shows the benefits of integrating multiple perspectives. As Hahn (2020) states, "Arguing things through is at the center of our attempts to come to accurate beliefs about the world [and] decide on a best course of action." For this reason, it is important to understand why disagreement can sometimes be constructive, and in other cases leads to conversational failure.

In this work, we investigated constructive disagreements from an NLP perspective. We have proposed a dataset of disagreements online and defined the task of predicting escalation as a proxy label for cases where disagreements were unconstructive. We analysed features that are associated with either class, and drew parallels with existing work. Using neural network models, we investigated the effect of modeling conversation structure and found that adding utterance hierarchy lead to an increase in performance. Finally, we validated our neural models by evaluating their performance and uncertainty when exposed to partial conversations. Our insights on constructive disagreements are not limited to Wikipedia and would be transferable to disagreements on other platforms. Additionally, our finding that edit summaries are an informative part of Talk page conversations should be useful for researchers who work on Wikipedia Talk pages more generally.

Acknowledgements

Christine De Kock is supported by scholarships from Huawei and the Oppenheimer Memorial Trust. Andreas Vlachos is supported the EPSRC grant no. EP/T023414/1: Opening Up Minds.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Gioia Boschi, Anthony P Young, Sagar Joglekar, Chiara Cammarota, and Nishanth Sastry. 2019. Having the last word: Understanding how to sample discussions online. *arXiv preprint arXiv:1906.04148*.

Jonathan P. Chang, Caleb Chiam, Liye Fu, Andrew Wang, Justine Zhang, and Cristian Danescu-Niculescu-Mizil. 2020. ConvoKit: A toolkit for the analysis of conversations. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 57–60, 1st virtual meeting. Association for Computational Linguistics.

Jonathan P. Chang and Cristian Danescu-Niculescu-Mizil. 2019. Trouble on the horizon: Forecasting the derailment of online conversations as they develop. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4743–4754, Hong Kong, China. Association for Computational Linguistics.

Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2017. Anyone can become a troll: Causes of trolling behavior in online discussions. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing*, pages 1217–1230.

Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

Esin Durmus and Claire Cardie. 2019. A corpus for modeling user and language effects in argumentation on online debating. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 602–607, Florence, Italy. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

John M Gottman and Lowell J Krokoff. 1989. Marital interaction and satisfaction: A longitudinal view. *Journal of consulting and clinical psychology*, 57(1):47.

Paul Graham. 2008. How to disagree. `http://www.paulgraham.com/disagree.html`.

Ulrike Hahn. 2020. Argument quality in real world argumentation. *Trends in Cognitive Sciences*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yiqing Hua, Cristian Danescu-Niculescu-Mizil, Dario Taraborelli, Nithum Thain, Jeffery Sorensen, and Lucas Dixon. 2018. WikiConv: A corpus of the complete conversational history of a large online collaborative community. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2818–2823, Brussels, Belgium. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351.

Hui Ma, Jian Wang, Lingfei Qian, and Hongfei Lin. 2020. HAN-ReGRU: hierarchical attention network with residual gated recurrent unit for emotion recognition in conversation. *Neural Computing and Applications*, pages 1–19.

Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2016. Conversational markers of constructive discussions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–578, San Diego, California. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *AAAI*, volume 18, pages 1527–1535.

Sara Rosenthal and Kathy McKeown. 2015. I couldn't agree more: The role of conversational structure in agreement and disagreement detection in online discussions. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 168–177, Prague, Czech Republic. Association for Computational Linguistics.

Donald B Rubin. 2007. The design versus the analysis of observational studies for causal effects: parallels with the design of randomized trials. *Statistics in medicine*, 26(1):20–36.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th international conference on world wide web*, pages 613–624. International World Wide Web Conferences Steering Committee.

Lu Wang and Claire Cardie. 2014. A piece of my mind: A sentiment analysis approach for online dispute detection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 693–699, Baltimore, Maryland. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Wikipedia. 2020a. Dispute tag template. `https://en.wikipedia.org/wiki/Template:Disputed`.

Wikipedia. 2020b. Npov tag template. `https://en.wikipedia.org/wiki/Template:POV`.

Wikipedia. 2020c. Wikipedia dispute resolution. `https://en.wikipedia.org/wiki/Wikipedia:Dispute_resolution`.

Wikipedia. 2020d. Wikipedia personal security. `https://en.wikipedia.org/wiki/Wikipedia:Personal_security_practices`.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.

Justine Zhang, Jonathan P Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Nithum Tahin, and Dario Taraborelli. 2018. Conversations gone awry: Detecting early signs of conversational failure. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics.*, volume 1.

Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016a. Conversational flow in Oxford-style debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 136–141, San Diego, California. Association for Computational Linguistics.

Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016b. Conversational flow in oxford-style debates. In *Proceedings of NAACL-HLT*, pages 136–141.

A Words list

Words associated with the positive and negative class are shown in Table 5. Coefficients are determined by training a bag-of-words model with logistic regression. The positive class is escalation.

Words	**Coefficients**
npov	-8.421
information	6.138
did	5.583
wp	4.969
right	-4.819
agree	-4.124
discussion	-4.106
issue	3.864
say	3.771
pov	-3.754
want	3.610
article	-3.398
work	3.362
articles	2.755
edit	2.747
claim	-2.232
case	2.180
people	-2.140
wikipedia	2.132
use	-2.013

Table 5: Words associated with the positive and negative class, using a bag-of-words model to predict escalation.

Acquiring a Formality-Informed Lexical Resource for Style Analysis

Elisabeth Eder[1], **Ulrike Krieg-Holz**[1], **Udo Hahn**[2]
[1] Institut für Germanistik, Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria
[2] Jena University Language & Information Engineering (JULIE) Lab,
Friedrich-Schiller-Universität Jena, Jena, Germany
`{elisabeth.eder, ulrike.krieg-holz}@aau.at, udo.hahn@uni-jena.de`

Abstract

To track different levels of formality in written discourse, we introduce a novel type of lexicon for the German language, with entries ordered by their degree of (in)formality. We start with a set of words extracted from traditional lexicographic resources, extend it by sentence-based similarity computations, and let crowdworkers assess the enlarged set of lexical items on a continuous informal-formal scale as a gold standard for evaluation. We submit this lexicon to an intrinsic evaluation related to the best regression models and their effect on predicting formality scores and complement our investigation by an extrinsic evaluation of formality on a German-language email corpus.

1 Introduction

The computational treatment of style in verbal communication has long been dominated by application concerns, e.g., the identification or profiling of authors in forensic linguistics (Ding et al., 2019) or the recognition of plagiarism (Alzahrani et al., 2012). This research was conducted assuming that simple lexico-statistic patterns identified by stylometric computations were sufficient to solve authorship and plagiarism assignment problems.

Despite their undisputed success in those limited fields, these studies scratched only the surface of the notion of 'style' as discussed in linguistic pragmatics (Hickey, 1993). From the many ways 'style' can be approached from a pragmatics perspective, we here focus on its inherent formality dimension, i.e., the distinction between *formal* (standard) and *informal* (colloquial) language use (for a survey, cf. Heylighen and Dewaele (1999)), with further extensions directed at the higher level of formal (e.g., elevated style) and the lower level of informal (e.g., vulgar) phrasing. Such distinctions of formality levels are crucial for the appropriateness of verbal expressions in a given discourse context.

In order to track different levels of formality in written communication, we introduce a novel type of lexicon for the German language, with entries ordered by their degree of (in)formality.[1] We start with a set of words extracted from traditional lexicographic resources, extend it by sentence-based similarity computations, and let crowdworkers assess the enlarged set of lexical items on a continuous informal-formal scale. This workflow is described in Section 3. The resulting lexicon comprising words with their respective formality scores subsequently serves as a gold standard for evaluation. In Section 4, we submit this lexicon to an intrinsic evaluation related to the best regression models and their effect on predicting formality scores, and complement our investigation by an extrinsic evaluation of formality on a German-language email corpus in Section 5.

2 Related Work

The relevance of 'style' for NLP is obvious for language output-focused core applications such as language generation (Sheikha and Inkpen, 2011; Dethlefs et al., 2014; Ficler and Goldberg, 2017), machine translation (Niu et al., 2018; Prabhumoye et al., 2018) or proper phrasing in argumentation (El Baff et al., 2020). Quite recently, the notion of 'formality style transfer' has received increasing attention, which captures the idea to generate a formal sentence given an informal one (*et vice versa*), while preserving its meaning (Shen et al., 2017; Fu et al., 2018; Rao and Tetreault, 2018; Li et al., 2018; Prabhumoye et al., 2018; Yang et al., 2018; Lample et al., 2019; Gong et al., 2019; Dai et al., 2019; Wu et al., 2019; John et al., 2019; Luo et al., 2019; Wang et al., 2019; Shang et al., 2019; Wang et al., 2020; Zhang et al., 2020; Yi et al., 2021).

---

[1] The lexicon is available at `https://github.com/ee-2/I-ForGer`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2028–2041
April 19 - 23, 2021.

Many efforts to cope with language style have been spent, however, in application niches, such as author identification or plagiarism detection. Most of the methodological contributions developed in this forensic branch are summarized under the label of *stylometrics* and have recently found their way into NLP analytics to unveil deception (Potthast et al., 2018; Pascucci et al., 2020a) or linguistic aggression (Harpalani et al., 2011; Nogueira dos Santos et al., 2018; Pascucci et al., 2020b).

The computational analysis of style according to stylometric principles, from its inception, is closely linked with *lexical frequency* counts. Typically, mostly function words (such as articles, pronouns, conjunctions, contractions, common abbreviations, hedging terms, also including punctuation marks) are assembled in small-sized dictionaries, together, if at all, with only a few content words (domain-specific nouns, verbs, adjectives). The frequency distributions resulting from counting these dictionary entries at the document or corpus level are already very beneficial for successfully dealing with disputed authorship problems (mostly for literary texts, but also for the detection of spam, fake news, or other kinds of toxic language) or uncovering plagiarism (mostly in scientific or news documents). Similar in spirit, word and sentence length criteria originating from readability metrics (Flesch-Kincaid, etc.) and several measures of vocabulary richness (e.g., type-token ratios, Yule's K and Burrow's Δ) were also incorporated into stylometric toolkits (Eder et al., 2016).

As a simple extension from these uni-grams, lexical or pseudo-lexical character *n-grams* (bi- or tri-grams, mostly) were determined and counted, as well. Slightly extending this (pseudo-)lexically focused approach by syntactic information, part-of-speech n-grams (or part-of-speech frequencies) were also considered to trace the human 'stylome', although lexical factors were found to be more relevant for style analysis than syntactic (POS sequence) patterns (van Halteren et al., 2005).

Simple frequency metrics have increasingly been complemented by various forms of lexical *association measures* (such as information gain, mutual information), and more sophisticated probabilistic models (principal component analysis (PCA), latent semantic analysis (LSA), or other types of topic models). Comprehensive lists of criteria and metrics are provided by Sheikha and Inkpen (2010); Neal et al. (2017); Ding et al. (2019).

We claim that despite their relevance for applications, such as authorship attribution and plagiarism detection, these mechanisms merely serve as easy to trace proxies for characterizing linguistic style. In our work, we will have a closer look at the style-marking semantic connotation of single lexical items as explicit carriers of linguistic formality as an important facet of language style.

A milestone for the formal definition of formality was set up by the pioneering work of Heylighen and Dewaele (1999) who defined the $\mathcal{F}$*-score*—close in spirit with the simple *lexico-statistic frequency* metrics from stylometry—as the percentage difference between deictic (article, pronouns, etc.) and non-deictic parts of speech (nouns, adjectives, etc.) in a document ($\mathcal{F}$ ranges between 0 and 100, with higher $\mathcal{F}$ indicating higher formality).[2] This document-level perspective was adapted by Lahiri et al. (2011) to sentence-level formality analysis.

A complementary *lexical* dimension for the formalization of formality was introduced by Brooke et al. (2010). They define the *formality score* for a word as a real number value in the range 1 to -1, with 1 representing an extremely formal word and -1 an extremely informal one, and assign a formality score to each lexical item based on standard word length, morphology-based features, lexical distribution criteria or association methods (LSA). Our work adheres to their way formality is scored in a formality lexicon and manually supplied seed sets are used (as starters), but differs markedly whether the lexicon is considered as a static (Brooke et al., 2010) or a dynamic resource (as we do; in a later study, Brooke and Hirst (2014) proposed a dynamic acquisition method, as well, by assigning a continuous formality score to single words based on their co-occurrence frequency with a hand-picked seed set of formal, neutral and informal words), and the way how semantic similarity is computed (LSA vs. embeddings). Further, we do not induce formality levels for a near-synonym task automatically but rather crowdsource nuances of formality for a relationally unrestricted lexical inventory from human raters.

Pavlick and Tetreault (2016) proposed a model of formality based on an empirical analysis of human formality perceptions. They apply their approach to analyze language use in online debate forums for multiple genres (news, blogs, emails,

---

[2] The $\mathcal{F}$*(ormality)*-score must not be confused with the F-score as a measure relating precision and recall.

and community question answering sites). Formality assessments are solicited via Amazon Turk (following the protocol established by Lahiri (2015)) using a 7-point Likert scale, with labels ranging from -3 (Very Informal) to 3 (Very Formal). A ridge regression classifier uses 11 different feature groups—five rarely used ones (among them WORD2VEC embeddings (Mikolov et al., 2013), parse trees, dependency tuples, and named entities) and six much more common ones (among them lower/upper casing, punctuation, readability scores, POS tags, and length-normalized formality and subjectivity scores)—to determine the formality level of sentences. Cross-genre analysis reveals that n-grams and word embeddings perform the best among all tested features (they achieve over 80% of the performance of the full classifier in all cases). This work comes closest to our approach, yet with differences in the way formality is assessed (Likert scales vs. best-worst scaling) and lexicon building is dealt with. Pavlick and Tetreault (2016) employ an acquisition method to score the formality of unseen phrases along the formal-casual dimension from scratch, as described in earlier work by Pavlick and Nenkova (2015) who use a log ratio metric based on the occurrence of phrases in various style-tagged corpora, in contrast to the embedding-based similarity model we propose.

Earlier computational models for detecting formality were proposed by Sheikha and Inkpen (2010); Peterson et al. (2011); Mosquera and Moreda (2012). The first two perform a binary classification only into formal *vs.* informal utterances, the third model classifies into four levels of (in)formality, and all of them operate at the document (as opposed to sentence) level.

3 Building a Formality-Informed Lexicon

3.1 Getting Started with VULGER

Previous work on computational lexicons (and lexicon acquisition) incorporating formality information focuses exclusively on the English language (Brooke et al., 2010; Brooke and Hirst, 2014; Pavlick and Nenkova, 2015; Pavlick and Tetreault, 2016). For German, VULGER (Eder et al., 2019)[3] constitutes a lexical resource that can be reused for such purposes to some degree. It comprises 3,300 German words scored by vulgarity/neutrality within a range of -1 (most vulgar) to $+1$ (most neutral). Accordingly, it covers the lower half of the formality spectrum quite well but completely lacks its upper half (formal up to elevated language). This study attempts to fill this gap by introducing I-FORGER, a comprehensive lexicon for *I*nformal and *For*mal *Ger*man. To acquire a lexicon covering the formal spectrum as well, we gathered formality-marked lexical entries in several ways as described in the following subsections.[4]

3.2 Input from Lexicographic Resources

As a first lexical acquisition step, we gathered lexical items from existing lexicographic resources based on their *manually* assigned categorical (in)formality tags:

Swear Words. As there is an overlap between swear words and vulgar lexicalizations, we used 500 lexical items randomly chosen from three German swear word lists[5] to feed the lower end of formality in I-FORGER.

Colloquial Items. In addition, we extracted 500 arbitrary terms marked as 'colloquial' ('ugs.' or 'umgangssprachlich,' in German) from the German slice of WIKTIONARY[6] and the German OPENTHESAURUS[7] supposed to range somewhere between vulgar and neutral on our scale.

Elevated Items. To extend the scale to the upper levels of linguistic formality, we also picked lexical items marked as 'elevated' ('geh.' or 'gehoben,' in German) from OPENTHESAURUS and WIKTIONARY yielding 1,000 additional terms (for the sake of balancing informal and formal entries in that phase). The reuse of manually curated lexicon resources (as seeds) thus follows the approach proposed by Brooke et al. (2010).

3.3 Lexicon Extension via Sentence Similarity

Given the intrinsic limitations of any manually curated lexicon resource, in the next step, we augmented I-FORGER by *automatic* means. We here suggest harvesting lexical candidates potentially carrying formality information from semantically

---

[3] `https://github.com/ee-2/VulGer`

[4] For basic NLP processing routines, we used SPACY (Honnibal et al., 2020) and FLAIR (Akbik et al., 2018, 2019).

[5] Retrieved from `http://www.hyperhero.com/de/insults.htm`, `http://www.insult.wiki/wiki/Schimpfwort-Liste` and `https://www.schimpfwoerter.de` on April 24, 2020.

[6] `https://de.wiktionary.org`

[7] `https://www.openthesaurus.de`

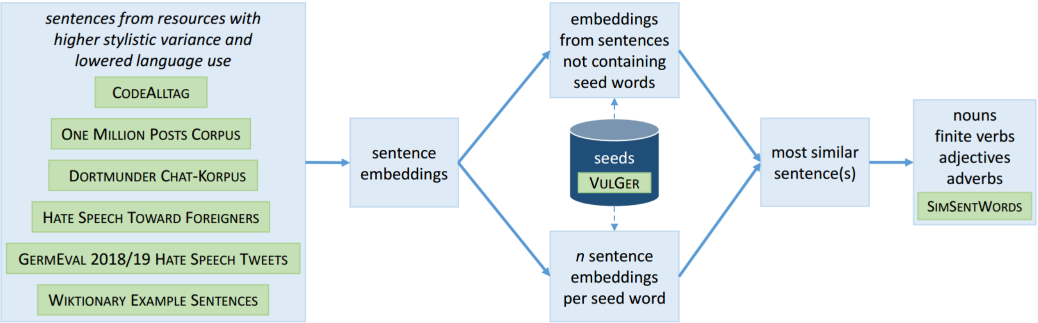


Figure 1: Generic language-independent workflow for gathering words for formality scoring approaches utilizing similar sentences (in blue) and its instantiation for our use case to acquire SIMSENTWORDS (in green)

similar sentences. This proposal goes beyond the standard way to utilize word embeddings in order to find close semantic neighbors based on the distributional hypothesis (see, e.g., Tulkens et al. (2016); Wiegand et al. (2018a) for detecting abusive lexicalizations this way). Rather than only discovering semantically related *words*, we extended our scope to semantically similar *sentences* to identify other relevant lexical candidates in the mined sentences, like an adjective modifying an offensive noun or other vulgar, yet otherwise unrelated, words in a vulgar word's context. On the flip side, this method admittedly gathers a considerable amount of noise (cf. Section 4 for a scoring approach to account for this problem).

3.3.1 Sentence Embeddings

As is well-known, BERT (Devlin et al., 2019) reaches new state-of-the-art results for various NLP problems, including semantic similarity tasks. However, finding semantically similar sentences close in vector space with BERT is computationally expensive. As a cure, Reimers and Gurevych (2019) introduced SENTENCE-BERT (SBERT), which modifies the pre-trained BERT network using siamese and triplet networks and produces semantically meaningful sentence embeddings that can be compared employing standard cosine similarity.

3.3.2 Sentence Similarity

To obtain candidate sentences for similarity computation for the German language, we employed a wide range of corpora. Our choices were guided by the requirements that these corpora should possess a high stylistic variance and contain vocabulary from the lower language register, too. We came up with:

- CODE ALLTAG[8] (Eder et al., 2020) comprising roughly 1,5M German-language emails,
- ONE MILLION POSTS CORPUS[9] (Schabus et al., 2017) containing about 1M user comments on news articles from the Austrian daily broadsheet newspaper DER STANDARD,
- DORTMUNDER CHAT KORPUS[10] (Beißwenger, 2013), with more than 140,000 German-language chats,
- HATE SPEECH TOWARDS FOREIGNERS[11] (Bretschneider and Peters, 2017), with about 6,000 posts and comments on German anti-foreign FACEBOOK pages,
- GERMEVAL 2018/19, collected for the task of identifying offensive language[12] (Wiegand et al., 2018b; Struß et al., 2019), including roughly 15,000 German-language tweets.

Using VULGER as a seed lexicon, we extracted sentences from these corpora by separating those containing VULGER entries from those that did not contain any VULGER item.[13] To further enlarge the number of sentences for each seed item, we also gathered sentences given as examples on the WIKTIONARY pages for the entries included in VULGER. From the resulting pool of sentences with seed words, we collected up to six sentences

---

[8] `https://github.com/codealltag`

[9] `https://ofai.github.io/million-post-corpus`

[10] `https://www.uni-due.de/germanistik/chatkorpus`

[11] `http://www.ub-web.de/research/index.html`

[12] `https://projects.fzai.h-da.de/iggsa/data-2019`

[13] We only took 100,000 randomly chosen sentences from CODE ALLTAG and the ONE MILLION POSTS CORPUS for performance reasons.

per word. We chose them randomly but tried to take one sentence from each of the six resources to keep some balance, both formality-wise as well as genre-wise.

These sentences served as seeds for the computation of similar sentences. Like the remaining sentences not containing any seed words, they were embedded with SENTENCE TRANSFORMERS (Reimers and Gurevych, 2019) using the multilingual model supporting German (Reimers and Gurevych, 2020). Then, for all seed sentence embeddings, we calculated the most similar sentence in the remaining sentence embeddings using cosine distance (the acquisition step proper). From these most similar sentences, we gathered lemmatized nouns, finite verbs, adjectives, and adverbs, omitting named entities. An overview of the entire acquisition procedure is depicted in Figure 1.

From the resulting word list, we randomly chose 1,000 items (denoted SIMSENTWORDS, in the following) to evaluate the regression approach and the acquisition strategy of automatically gathering new words to score. As we also wanted to measure the acquisition noise, we further divided the words into 500 items manually cleansed from spelling mistakes, etc. (SIMSENTWORDS$_{cleansed}$), and left 500 as-is (SIMSENTWORDS$_{noisy}$).

3.4 I-FORGER at a Glance

Putting these pieces together, I-FORGER, the final lexicon, comprises 3,000 words, in total, with three major divisions: 1,000 terms from elevated language usage, 1,000 words, with swearwords and colloquial items joined, presumably linked to the lowered stylistic inventory, and 1,000 words that should rather occur at the lower end of our informality scale, but potentially include words from all stylistic levels (see Table 1).

Resource	# Lexical Items
ELEVATEDWORDS	1,000
SWEARWORDS	500
COLLOQUIALWORDS	500
SIMSENTWORDS	1,000
SIMSENTWORDS$_{cleansed}$	500
SIMSENTWORDS$_{noisy}$	500
Total	3,000

Table 1: Contributions from various resources for the I-FORGER lexicon

3.5 Human Assessment of I-FORGER

To establish a gold standard for subsequent evaluation, we gathered human formality assessments. For that, I-FORGER was annotated with Best-Worst-Scaling (BWS), a method that delivers high-quality annotations with only a relatively small number of annotation steps compared to standard point-interval based methods (e.g., Likert scales) for human assessment tasks. BWS also adheres to the principle that a “continuum of formality” (Heylighen and Dewaele, 1999) exists rather than n-ary categorical distinctions between formal and informal utterances (see also Lahiri et al. (2011); Brooke and Hirst (2014) for works based on degrees of formality).

BWS was introduced into NLP for emotion scaling by Kiritchenko and Mohammad (2016, 2017). Annotators are presented with n items at a time (an n-tuple, where $n > 1$, and typically $n = 4$). They then have to decide which item from the n-tuple is the *best* (highest in terms of the property of interest) and which is the *worst* (lowest in terms of the property of interest).

In our case, judges had to select the *most elevated* and the *most vulgar* terms per given n-tuple. We used the BWS tool[14] from Kiritchenko and Mohammad (2016, 2017) to generate 6,000 4-tuples for human assessment. Tuples were produced randomly under the premise that each term had to occur only once in eight different tuples and each tuple was unique.

For the annotation process proper, we used the crowdsourcing platform CLICKWORKER,[15] where we had each n-tuple assessed by five annotators (Kiritchenko and Mohammad (2016) showed that as few as 2-3 responses per tuple are sufficient to get reliable scores, at least for the assessment of sentiment.). In order to get real-valued scores from the BWS annotations, we applied COUNTS ANALYSIS (Orme, 2009)[16] and subtracted the percentage of times the term was chosen as worst from the percentage of times the term was chosen as best. Thus, we got scores between $+1$ (most formal) and -1 (most informal). We computed the split-half reliability[16] by randomly splitting the annotations of a tuple into two halves, calculating scores independently for these halves, and mea-

---

[14] `http://www.saifmohammad.com/WebPages/BestWorst.html`

[15] `https://www.clickworker.de`

[16] Again, we used the scripts from Kiritchenko and Mohammad (2016, 2017).

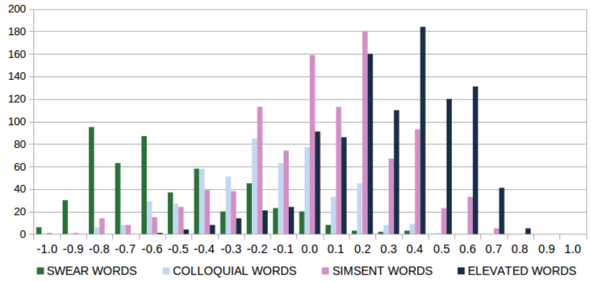


Figure 2: Distribution of scores per resource of I-ForGer

suring the correlation between the resulting two sets of scores. We got an average Spearman's ρ of $0.8954\ \ (+/-0.0030)$ over 100 trials.

Figure 2 displays the distribution of human assessed scores per resource for I-ForGer. While SwearWords and, to a lesser degree, also ColloquialWords are linked to lower scores, and ElevatedWords obtained higher scores, SimSentWords are found in the middle spreading on the entire scale of scores, also comprising a fair amount of words from the lower end of formality.

4 Intrinsic Evaluation of I-ForGer

Rather than increasing the size and thus the coverage of lexicons to improve performance on potential applications, we intend to score (unseen) words on the fly. Hence, we first evaluate the word scoring model (Section 4.1). Next, we assess the four main input streams of I-ForGer (Section 4.2) and the extension of the scale regarding formality levels (Section 4.3). Figure 3 illustrates the schematic workflow for our word scoring procedure, including (and marked in green) the three evaluation tasks.

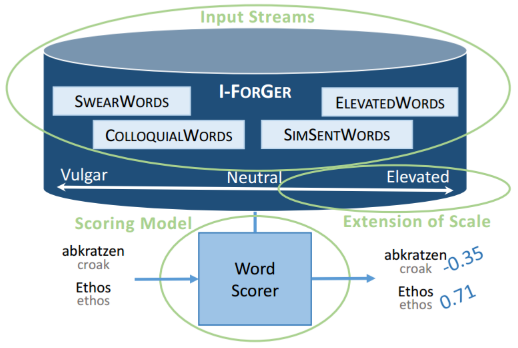


Figure 3: Overview of the word scoring workflow; parts to evaluate are marked in green

4.1 Regression Models for Word Scoring

We adopted various approaches using a seed lexicon, actually, the entries' word embeddings, as training data for regression models to automatically score new lexical items for their formality connotation (see, e.g., Li et al. (2017) and Buechel and Hahn (2018) for a similar scenario for automatic emotion induction).

As input features we decided for fastText word embeddings (Grave et al., 2018) with their own out-of-vocabulary (OOV) functionality. We found that they performed better than getting the OOV handling from BPEmb subword embeddings (Heinzerling and Strube, 2018), based on Byte Pair Encoding (BPE) (Sennrich et al., 2016), instead, or solely utilizing pure BPEmb embeddings.

We evaluated different regression models. Besides Ridge Regression,[17] which is linear regression with L_2 regularization during training, we also experimented with Densifier (Rothe et al., 2016), which learns an orthogonal transformation of the embedding space, and a modified, more robust variant of the latter, DensRay (Dufter and Schütze, 2019).[18] We ran a feed-forward neural network with one hidden layer combined with the boosting algorithm AdaBoost.R2 (Boosted FFNN) as proposed by Du and Zhang (2016).[19] Further, we tested neural networks with more than one hidden layer, namely two hidden layers with 256, 128 units ($\text{NN}_{2Hidden}$) and three hidden layers with 256, 128 and 64 units ($\text{NN}_{3Hidden}$).[20]

Table 2 depicts that Densifier and DensRay performed worse than all the others. Also, Ridge Regression yielded significantly lower results than the Boosted FFNN model. We found no difference between $\text{NN}_{2Hidden}$, $\text{NN}_{3Hidden}$ and Boosted FFNN since all three reached a strong Spearman's ρ of 0.77. As a higher number of hidden layers did not significantly improve results, we used Boosted FFNN for further processing.

---

[17] We used the scikit-learn.org implementation with the default parameters.

[18] We used their code provided on `https://github.com/pdufter/densray`.

[19] We copied their code on `https://github.com/StevenLOL/ialp2016_Shared_Task`.

[20] We used Keras in TensorFlow with the following hyperparameters: embedding/input layer with 0.2 and hidden layers with 0.5 dropout, MaxNorm weight constraint of 3, random normal weight initialization, ReLu activation, Adam optimizer, batch size of 32, mean squared error loss and 1,000 epochs with early stopping.

Model	Spearman's ρ
Ridge Regression	0.706*
Densifier	0.632*
DensRay	0.621*
NN$_{2Hidden}$	0.773
NN$_{3Hidden}$	0.771
Boosted FFNN	0.773

Table 2: Averaged Spearman's ρ for different models (10-fold cross-validation on I-ForGer); statistically significant differences (using the two-sided Wilcoxon signed-rank test on Spearman's ρ) are marked with '*' for $p < 0.005$ with respect to Boosted FFNN

4.2 Assessment of Input Streams

Table 3 pinpoints the predictability of formality for a particular input stream of I-ForGer in a 10-fold cross-validation setting. Learning scores of ColloquialWords and ElevatedWords seems harder than scoring SwearWords and SimSentWords. The lower human agreement on choosing the most elevated item supports this finding for the upper half of the formality spectrum. The data also reveal that the regression model is somewhat prone to noise since original SimSentWords$_{noisy}$ achieved much lower results than curated SimSentWords$_{cleansed}$. However, this acquisition strategy seems to be a choice worth considering for scoring approaches.

Input Stream	Spearman's ρ
SwearWords	0.593
ColloquialWords	0.409
SimSentWords	0.672
SimSentWords$_{cleansed}$	0.732
SimSentWords$_{noisy}$	0.595
ElevatedWords	0.477
I-ForGer	0.773

Table 3: Spearman's ρ for Boosted FFNN on I-ForGer with results for different input streams (10-fold cross-validation)

4.3 Assessment of Formality Scale Extension

A comparison with VulGer suggests that scoring an extended range of linguistic styles is a more difficult task, since evaluating the Boosted FFNN model on VulGer achieved a higher Spearman's ρ of 0.827 (10-fold cross-validation) than on I-ForGer (see Table 3). Nevertheless, applying a model trained on I-ForGer to VulGer gave a Spearman's ρ of 0.678, which signals evidence that I-ForGer still captures the vulgar-neutral dimension despite being trained on an extended scale with fewer words (3,000 *vs.* 3,300). It also shows that the word scoring approach *per se* indeed yields reliable results on the informal-formal dimension.

5 Extrinsic Evaluation of I-ForGer

In order to gather evidence for the value of I-ForGer in combination with the word scoring approach within a realistic use case, we ran experiments with emails, which possess a higher stylistic variability than news concerning their formality spread (Pavlick and Tetreault, 2016). Other work related to the formality of emails is typically carried out in the context of communication behavior studies in enterprises, with a focus on determining social factors (social distance, relative power, and the weight of imposition) that affect the sender's choice of formality (Peterson et al., 2011) or on the affective dimension of email exchanges (Chhaya et al., 2018) in terms of the prediction of frustration of employees from email data.

5.1 Email Corpus and Formality Gold Standard

Again using BWS and the tools from Kiritchenko and Mohammad (2016, 2017) mentioned before, we manually scored 800 German emails from CodE Alltag$_{S+d}$, a specialized, metadata-rich subset of CodE Alltag (Eder et al., 2020), for their formality. 35 annotators had to select the *most formal* email and the *most informal* email from four emails per rating step. Altogether, we had 1,600 4-tuples assessed three times. We got an average

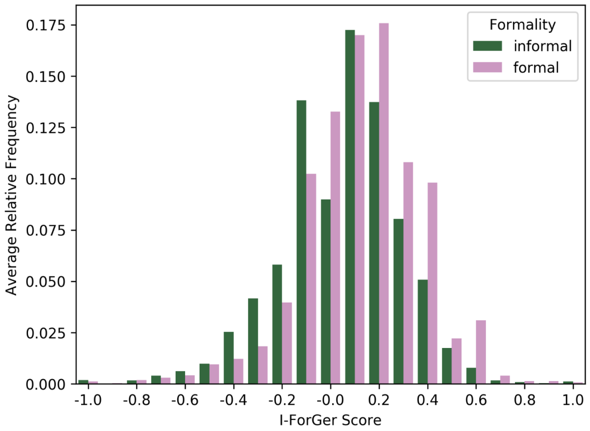


Figure 4: Distribution of I-ForGer scores for formal (with formality scores from 0 to +1) and informal emails (rated from -1 to 0)

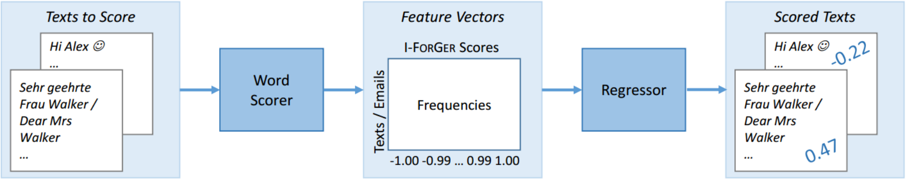


Figure 5: Overview of our workflow to score emails for formality using I-ForGer scores

Spearman's ρ of 0.9198 $(+/-0.0043)$ over 100 trials. The resulting scores on an informal-formal scale from -1 (informal) to $+1$ (formal) served as basis for our experiments.

5.2 Distribution of I-ForGer Scores

Under the assumption that formal emails include more formal words and informal emails more informal terms, we, first, examined the distribution of scores calculated for the emails' words with the Boosted FFNN model learned on I-ForGer. We split our dataset tentatively in two folds: emails with scores from -1 to 0 formed the informal part, whereas emails rated with positive numbers in a range from 0 to $+1$ were regarded as formal. Figure 4 indicates that, in comparison, informal emails indeed contain more negatively scored terms and formal emails comprise more words in the upper part of the informal-formal scale.

5.3 Formality Scoring of Emails

In the final evaluation setup, we tested whether word formality scoring works better than lexicon look-up in traditional resources and whether categorized items or continuous scores get better results. To determine the proper features for a linear regressor predicting formality scores,[21] we used a vector comprising the relative frequencies of an email's word scores as input (count per score divided by the total number of scored words). In one setting, the I-ForGer word scorer tagged (unseen) nouns, finite verbs and adjectives (Figure 5 depicts the workflow for this experiment.). In another setting, we only counted the scores of words already present in I-ForGer (without acquisition step). Besides relative score frequencies, we also tested taking the average score per document (sum of all calculated scores divided by the total number of scored words) as input feature[22] for both settings.

For a comparison of scores against pre-specified categories, we mapped the scores of I-ForGer to formality categories. We divided the scale into five distinct sections (e.g., scores between 0.6 and 1.0 form one category), assigned the respective category to each score and used a classifier instead of a linear regressor to learn the categories of new words. The relative frequencies of the categories then served as input for the linear regressor. We also experimented with ignoring OOV words and only utilizing lexicon look-up for the categorical scenario. For this setting, we exploited the *complete* pre-categorized word lists we got the Swear Words, Colloquial Words and Elevated Words from in order to increase coverage. In this way, in case of swear words, e.g., we did not only use the 500 items assembled in the I-ForGer lexicon, but used a list of more than 13,000 entries. As features instead of scores we counted the frequency of swear words, colloquial words and elevated words separately in each email and divided it by the total number of words found in the lexicons.

Table 4 summarizes our results. Scoring words based on I-ForGer yielded significantly better results than any other configuration reaching a strong Spearman's ρ of 0.728. When using the average score per document, there is still a positive correlation with the emails' formality scores. Utilizing a fixed set of lexical terms and not scoring new

Lexicon	OOV	Features	ρ
I-ForGer	Scored	counts	**0.728**
I-ForGer	Scored	average	0.587*
I-ForGer	Ignored	counts	0.446*
I-ForGer	Ignored	average	0.123*
I-ForGer$_{cat}$	Classified	counts	0.476*
Categories	Ignored	counts	0.335*

Table 4: Spearman's ρ for different configurations (10-fold cross-validation on formality scored CodE AlltaG$_{S+d}$); significance differences in respect to best model calculated with two-sided Wilcoxon signed-rank test are marked with '*' for $p < 0.005$

---

[21] We used a neural network with two hidden layers (128 and 64 units) and the same configurations in the Keras library in TensorFlow as reported for NN$_{2Hidden}$ or NN$_{3Hidden}$.

[22] Using the average scores directly to determine a correlation to the emails' formality scores gave comparable results.

words also performed better with score frequencies than using the average. However, compared to employing a word scorer for unseen words, the results for simple lexicon look-up are lower, a finding that seems to be due to the limited coverage of I-ForGer. Therefore, we can conclude that our way of scoring potentially unseen words is an effective and advantageous alternative to using fixed-size, and thus limited, lexical resources.

Employing the relative frequencies of formality categories instead of scores also yielded lower results for both settings, classifying new words (see I-ForGer$_{cat}$) and utilizing lexicon look-up with pre-categorized items (CATEGORIES). This demonstrates the benefit of a scaling approach instead of relying on coarse-grained categories.

6 Conclusion

Different levels of formality these days find increasing attention, both in methodological approaches and NLP applications. The necessity of choosing a socially appropriate tone is particularly evident in digitally mediated discourse, e.g., formal business or informal private email communication (Chhaya et al., 2018) or social media interaction via reviews, chats, or blogs (Pavlick and Tetreault, 2016; Gonzàlez Bermúdez, 2015). The increasing relevance of conversationally adequate virtual personal assistants (Shamekhi et al., 2016), chatbots (Chaves et al., 2019) and automatic procedures for smart response generation (Kannan et al., 2016) requires sensitivity on the generator's side to strike the right tone and avoid the false one. Similarly, machine translation poses special problems when expressions of (in)formality have to be adequately transferred between different languages (Niu et al., 2018). Progress in monitoring formality levels is a methodological prerequisite for several downstream applications that have to comply with users' habitual expectations or increase user satisfaction, e.g., in commercial interactions (customer service communication) (Liebrecht et al., 2020; Elsholz et al., 2019) or medical consultation (Fadhil and Schiavo, 2019).

As a methodological contribution, we here propose a lexical approach to computational style analysis based on I-ForGer, a lexicon whose (3,000) items are scaled on a continuous informal-formal spectrum. We make three new contributions to style analysis: First, a language-independent lexicon acquisition architecture employing sentence embeddings forms the basis for computing sentence similarity, thus finding formality-sensitive lexical items not contained in the seeds. Second, best-worst scaling is used for creating gold standards available for an in-depth intrinsic and extrinsic evaluation of the new lexical resource. Finally, I-ForGer stands out as the first formality-informed lexicon for the German language. This resource is available at `https://github.com/ee-2/I-ForGer`.

Despite our lexical focus, we are aware of the fact that formality is not only lexically expressed. Consequently, a lexicon-based approach has to be complemented by methods that account for non-lexicalized varieties of formality. Such forms may include syntactic variability, linguistic complexity and readability, as well as correctness of language use regarding orthography, morphology and syntax. For research on formality detection incorporating its syntactic, semantic and discourse facets, cf., e.g., Heylighen and Dewaele (1999), Li et al. (2013) or Pavlick and Tetreault (2016). These branches will also be part of our future work. Still, a (potentially) large portion of formality assessments is rooted in lexical signals, which we capture by the methodology advanced in this paper.

Acknowledgments

We thank the anonymous reviewers of our paper for their remarks, which helped us strengthen the paper. Special thanks go to Michael Wiegand for additional hints and comments.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR : an easy-to-use framework for state-of-the-art NLP. In *NAACL-HLT 2019 — Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Demonstrations Session. Minneapolis, Minnesota, USA, June 3-4, 2019*, pages 54–59.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018 — Proceedings of the 27th International Conference on Computational Linguistics: Main Conference. Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1638–1649.

Salha M. Alzahrani, Naomie Salim, and Ajith Abraham. 2012. Understanding plagiarism linguistic patterns, textual features, and detection methods. *IEEE Transactions on Systems, Man, and Cybernetics — Part C: Applications and Reviews*, 42(2):133–149.

Michael Beißwenger. 2013. Das Dortmunder Chat-Korpus. *Zeitschrift für germanistische Linguistik*, 41(1):161–164.

Uwe Bretschneider and Ralf Peters. 2017. Detecting offensive statements towards foreigners in social media. In *HICSS-50 — Proceedings of the 50th Hawaii International Conference on System Sciences 2017. Hawaii, USA, January 4-7, 2017*, pages 2213–2222.

Julian Brooke and Graeme Hirst. 2014. Supervised ranking of co-occurrence profiles for acquisition of continuous lexical attributes. In *COLING 2014 — Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers. Dublin, Ireland, August 23-29, 2014*, pages 2172–2183.

Julian Brooke, Tong Wang, and Graeme Hirst. 2010. Automatic acquisition of lexical formality. In *COLING 2010 — Proceedings of the 23rd International Conference on Computational Linguistics: Posters. Beijing, China, 23-27 August 2010*, pages 90–98.

Sven Buechel and Udo Hahn. 2018. Word emotion induction for multiple languages as a deep multi-task learning problem. In *NAACL-HLT 2018 — Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, Louisiana, USA, June 1-6, 2018*, volume 1: Long Papers, pages 1907–1918.

Ana Paula Chaves, Eck Doerry, Jesse Egbert, and Marco Gerosa. 2019. It's how you say it: identifying appropriate register for chatbot language design. In *HAI '19 — Proceedings of the 7th ACM International Conference on Human-Agent Interaction. Kyoto, Japan, October 6-10, 2019*, pages 102–109.

Niyati Chhaya, Kushal Chawla, Tanya Goyal, Projjal Chanda, and Jaya Singh. 2018. Frustrated, polite or formal: quantifying feelings and tone in emails. In *PEOPLES 2018 — Proceedings of the 2nd Workshop on Computational Modeling of PEople's Opinions, PersonaLity, and Emotions in Social media @ NAACL-HLT 2018. New Orleans, Louisiana, USA, June 6, 2018*, pages 76–86.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style transformer: unpaired text style transfer without disentangled latent representation. In *ACL 2019 — Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy, July 28 - August 2, 2019*, pages 5997–6007.

Nina Dethlefs, Heriberto Cuayáhuitl, Helen W. Hastie, Verena Rieser, and Oliver Lemon. 2014. Cluster-based prediction of user ratings for stylistic surface realisation. In *EACL 2014 — Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. Gothenburg, Sweden, April 26-30, 2014*, pages 702–711.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. 2019. BERT : pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019 — Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, Minnesota, USA, June 2-7, 2019*, volume 1: Long and Short Papers, pages 4171–4186.

Steven H. H. Ding, Benjamin C. M. Fung, Farkhund Iqbal, and William K. Cheung. 2019. Learning stylometric representations for authorship analysis. *IEEE Transactions on Cybernetics*, 49(1):107–121.

Steven Du and Xi Zhang. 2016. AICYBER's system for IALP 2016 Shared Task: character-enhanced word vectors and boosted neural networks. In *IALP 2016 — Proceedings of the [20th] 2016 International Conference on Asian Language Processing. Tainan, Taiwan, November 21-23, 2016*, pages 161–163.

Philipp Dufter and Hinrich Schütze. 2019. Analytical methods for interpretable ultradense word embeddings. In *EMNLP-IJCNLP 2019 — Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing & 9th International Joint Conference on Natural Language Processing. Hong Kong, China, November 3-7, 2019*, pages 1185–1191.

Elisabeth Eder, Ulrike Krieg-Holz, and Udo Hahn. 2019. At the lower end of language: exploring the vulgar and obscene side of German. In *ALW-3 — Proceedings of the 3rd Workshop on Abusive Language Online @ ACL 2019. Florence, Italy, August, 1, 2019*, pages 119–128.

Elisabeth Eder, Ulrike Krieg-Holz, and Udo Hahn. 2020. CODE ALLTAG 2.0 : a pseudonymized German-language email corpus. In *LREC 2020 — Proceedings of the 12th International Conference on Language Resources and Evaluation. Marseille, France, May 11-16, 2020*, pages 4466–4477.

Maciej Eder, Mike Kestemont, and Jan Rybicki. 2016. Stylometry with R : a package for computational text analysis. *R Journal*, 16(1):107–121.

Roxanne El Baff, Henning Wachsmuth, Khalid Al-Khatib, and Benno Stein. 2020. Analyzing the persuasive effect of style in news editorial argumentation. In *ACL 2020 — Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. [Seattle, Washington, USA,] July 5-10, 2020 (Virtual Event)*, pages 3154–3160.

Ela Elsholz, Jon Chamberlain, and Udo Kruschwitz. 2019. Exploring language style in chatbots to increase perceived product value and user engagement. In *CHIIR '19 — Proceedings of the 2019 Conference on Human Information Interaction and Retrieval. Glasgow, Scotland, UK, March 10-14, 2019*, pages 301–305.

Ahmed Fadhil and Gianluca Schiavo. 2019. Designing for health chatbots. ArXiv preprint arXiv:1902.09022.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *StyVa 2017 — Proceedings of the [1st] Workshop on Stylistic Variation @ EMNLP 2017. Copenhagen, Denmark, September 8, 2017*, pages 94–104.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: exploration and evaluation. In *AAAI-IAAI-EAAI '18 — Proceedings of the 32nd AAAI Conference on Artificial Intelligence & 30th Conference on Innovative Applications of Artificial Intelligence & 8th Symposium on Educational Advances in Artificial Intelligence. New Orleans, Louisiana, USA, February 2-7, 2018*, pages 663–670.

Hongyu Gong, Suma Bhat, Lingfei Wu, JinJun Xiong, and Wen-mei Hwu. 2019. Reinforcement learning based text style transfer without parallel training corpus. In *NAACL-HLT 2019 — Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, Minnesota, USA, June 2-7, 2019*, pages 3168–3180.

Meritxell Gonzàlez Bermúdez. 2015. An analysis of Twitter corpora and the differences between formal and colloquial tweets. In *TweetMT 2015 — Proceedings of the Tweet Translation Workshop 2015 @ SEPLN 2015. Alicante, Spain, September 15, 2015*, number 1445 in CEUR Workshop Proceedings, pages 1–7.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomáš Mikolov. 2018. Learning word vectors for 157 languages. In *LREC 2018 — Proceedings of the 11th International Conference on Language Resources and Evaluation. Miyazaki, Japan, May 7-12, 2018*, pages 3483–3487.

Hans van Halteren, R. Harald Baayen, Fiona J. Tweedie, Marco Haverkort, and Anneke Neijt. 2005. New machine learning methods demonstrate the existence of a human stylome. *Journal of Quantitative Linguistics*, 12:65–77.

Manoj Harpalani, Michael Hart, Sandesh Singh, Rob Johnson, and Yejin Choi. 2011. Language of vandalism: improving WIKIPEDIA vandalism detection via stylometric analysis. In *ACL-HLT 2011 — Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland, Oregon, USA, 19-24 June 2011*, volume 2: Short Papers, pages 83–88.

Benjamin Heinzerling and Michael Strube. 2018. BPEMB: tokenization-free pre-trained subword embeddings in 275 languages. In *LREC 2018 — Proceedings of the 11th International Conference on Language Resources and Evaluation. Miyazaki, Japan, May 7-12, 2018*, pages 2989–2993.

Francis Heylighen and Jean-Marc Dewaele. 1999. Formality of language: definition, measurement and behavioral determinants. Technical report, Center "Leo Apostel", Free University of Brussels.

Leo Hickey. 1993. Stylistics, pragmatics and pragmastylistics. *Revue Belge de Philologie et d'Histoire*, 71(3):573–586.

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

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *ACL 2019 — Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy, July 28 - August 2, 2019*, pages 424–434.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufman, Andrew D. Tomkins, Balint Miklos, Gregory S. Corrado, László Lukács, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart reply: automated response suggestion for email. In *KDD 2016 — Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, California, USA, August 13-17, 2016*, pages 955–964.

Svetlana Kiritchenko and Saif M. Mohammad. 2016. Capturing reliable fine-grained sentiment associations by crowdsourcing and best-worst scaling. In *NAACL-HLT 2016 — Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California, USA, June 12-17, 2016*, pages 811–817.

Svetlana Kiritchenko and Saif M. Mohammad. 2017. Best-worst scaling more reliable than rating scales: a case study on sentiment intensity annotation. In *ACL 2017 — Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. Vancouver, British Columbia, Canada, July 30 - August 4, 2017*, volume 2: Short Papers, pages 465–470.

Shibamouli Lahiri. 2015. SQUINKY! A corpus of sentence-level formality, informativeness, and implicature. CoRR, abs/1506.02306.

Shibamouli Lahiri, Prasenjit Mitra, and Xiaofei Lu. 2011. Informality judgment at sentence level and experiments with formality score. In *Computational Linguistics and Intelligent Text Processing. CICLing 2011 — Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing. Tokyo, Japan, February 20-26, 2011*, number 6609 in Lecture Notes in Computer Science (LNCS), pages 446–457. Springer.

Guillaume Lample, Sandeep Subramanian, Eric M. Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *ICLR 2019 — Proceedings of the 7th International Conference on Learning Representations. New Orleans, Louisiana, USA, May 6-9, 2019*.

Haiying Li, Zhiqiang Cai, and Arthur C. Graesser. 2013. Comparing two measures for formality. In *FLAIRS 2013 — Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference. St. Pete Beach, Florida, USA, May 22–24, 2013*, pages 220–225.

Juncen Li, Robin Jia, He He, and Percy S. Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *NAACL-HLT 2018 — Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, Louisiana, USA, June 1-6, 2018*, pages 1865–1874.

Minglei Li, Qin Lu, Yunfei Long, and Lin Gui. 2017. Inferring affective meanings of words from word embedding. *IEEE Transactions on Affective Computing*, 8(4):443–456.

Christine Liebrecht, Lena Sander, and Charlotte Van Hooijdonk. 2020. Too informal? How a chatbot's communication style affects brand attitude and quality of interaction. In *Conversations 2020 — Proceedings of the 4th International Workshop on Chatbot Research. [Amsterdam, Netherlands], 23-24 Nov 2020 (Virtual Event)*, page [16pp.].

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Xu Sun, and Zhifang Sui. 2019. A dual reinforcement learning framework for unsupervised text style transfer. In *IJCAI '19 — Proceedings of the 28th International Joint Conference on Artificial Intelligence. Macao, China, August 10-16, 2019*, pages 5116–5122.

Tomáš Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26 — NIPS 2013. Proceedings of the 27th Annual Conference on Neural Information Processing Systems. Lake Tahoe, Nevada, USA, December 5-10, 2013*, pages 3111–3119.

Alejandro Mosquera and Paloma Moreda. 2012. SMILE : an informality classification tool for helping to assess quality and credibility in Web 2.0 texts. In *RAMSS 2012 — Proceedings of the 1st Workshop on Real-Time Analysis and Mining of Social Streams @ ICWSM 2012. Dublin, Ireland, June 4, 2012*, number WS-12-02 in AAAI Technical Report, pages 2–7.

Tempestt Neal, Kalaivani Sundararajan, Aneez Fatima, Yiming Yan, Yingfei Xiang, and Damon Woodard. 2017. Surveying stylometry techniques and applications. *ACM Computing Surveys*, 50(6):#86.

Xing Niu, Sudha Rao, and Marine Carpuat. 2018. Multi-task neural models for translating between styles within and across languages. In *COLING 2018 — Proceedings of the 27th International Conference on Computational Linguistics: Main Conference. Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1008–1021.

Bryan Orme. 2009. Maxdiff analysis: simple counting, individual-level logit, and HB. *Sawtooth Software, Inc.*

Antonio Pascucci, Raffaele Manna, Ciro Caterino, Vincenzo Masucci, and Johanna Monti. 2020a. Is this hotel review truthful or deceptive? A platform for disinformation detection through computational stylometry. In *STOC 2020 — Proceedings of the 1st International Workshop on Social Threats in Online Conversations: Understanding and Management @ LREC 2020. Marseille, France, May 11, 2020*, pages 35–40.

Antonio Pascucci, Raffaele Manna, Vincenzo Masucci, and Johanna Monti. 2020b. The role of computational stylometry in identifying (misogynistic) aggression in English social media texts. In *TRAC-2 2020 — Proceedings of the 2nd Workshop on Trolling, Aggression and Cyberbullying @ LREC 2020. Marseille, France, May 16, 2020*, pages 69–75.

Ellie Pavlick and Ani Nenkova. 2015. Inducing lexical style properties for paraphrase and genre differentiation. In *NAACL-HLT 2015 — Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Denver, Colorado, USA, May 31 - June 5, 2015*, pages 218–224.

Ellie Pavlick and Joel R. Tetreault. 2016. An empirical analysis of formality in online communication. *Transactions of the Association for Computational Linguistics*, 4:61–74.

Kelly Peterson, Matt Hohensee, and Fei Xia. 2011. Email formality in the workplace: a case study on the Enron corpus. In *LSM 2011 — Proceedings of the Workshop on Language in Social Media @ ACL-HLT 2011. Portland, Oregon, USA, 23 June 2011*, pages 86–95.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *ACL 2018 — Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, Victoria, Australia, July 15-20, 2018*, volume 1: Long Papers, pages 231–240.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan R. Salakhutdinov, and Alan W. Black. 2018. Style transfer through back-translation. In *ACL 2018 — Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, Victoria, Australia, July 15-20, 2018*, pages 866–876.

Sudha Rao and Joel R. Tetreault. 2018. Dear Sir or Madam, may I introduce the GYAFC dataset: corpus, benchmarks and metrics for formality style transfer. In *NAACL-HLT 2018 — Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, Louisiana, USA, June 1-6, 2018*, volume 1: Long Papers, pages 129–140.

Nils Reimers and Iryna Gurevych. 2019. SENTENCE-BERT: Sentence embeddings using siamese BERT-networks. In *EMNLP-IJCNLP 2019 — Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing & 9th International Joint Conference on Natural Language Processing. Hong Kong, China, November 3-7, 2019*, pages 3982–3992.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. ArXiv preprint arXiv:2004.09813.

Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. 2016. Ultradense word embeddings by orthogonal transformation. In *NAACL-HLT 2016 — Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California, USA, June 12-17, 2016*, pages 767–777.

Cícero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. In *ACL 2018 — Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, Victoria, Australia, July 15-20, 2018*, volume 2: Short Papers, pages 189–194.

Dietmar Schabus, Marcin Skowron, and Martin Trapp. 2017. One million posts: a data set of German online discussions. In *SIGIR '17 — Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 1241–1244.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL 2016 — Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Berlin, Germany, August 7-12, 2016*, volume 1: Long Papers, pages 1715–1725.

Ameneh Shamekhi, Mary Czerwinski, Gloria Mark, Margeigh Novotny, and Gregory A. Bennett. 2016. An exploratory study toward the preferred conversational style for compatible virtual agents. In *Intelligent Virtual Agents. IVA 2016 — Proceedings of the 16th International Conference on Intelligent Virtual Agents. Los Angeles, California, USA, September 20-23, 2016*, number 10011 in Lecture Notes in Artificial Intelligence (LNAI), pages 40–50. Springer.

Mingyue Shang, Piji Li, Zhenxin Fu, Lidong Bing, Dongyan Zhao, Shuming Shi, and Rui Yan. 2019. Semi-supervised text style transfer: cross projection in latent space. In *EMNLP-IJCNLP 2019 — Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing & 9th International Joint Conference on Natural Language Processing. Hong Kong, China, November 3-7, 2019*, pages 4936–4945.

Fadi Abu Sheikha and Diana Z. Inkpen. 2010. Automatic classification of documents by formality. In *NLPKE 2010 — Proceedings of the 6th IEEE International Conference on Natural Language Processing and Knowledge Engineering. Beijing, China, 21-23 August 2010*, page [5pp.].

Fadi Abu Sheikha and Diana Z. Inkpen. 2011. Generation of formal and informal sentences. In *ENLG 2011 — Proceedings of the 13th European Workshop on Natural Language Generation. Nancy, France, 28-30 September 2011*, pages 187–193.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems 30 — NIPS 2017. Proceedings of the 31st Annual Conference on Neural Information Processing Systems. Long Beach, California, USA, December 3-9, 2017*, pages 6833–6844.

Julia Maria Struß, Melanie Siegel, Josef Ruppenhofer, Michael Wiegand, and Manfred Klenner. 2019. Overview of GERMEVAL Task 2, 2019 Shared Task on the Identification of Offensive Language. In *KONVENS 2019 — Proceedings of the 15th Conference on Natural Language Processing. Erlangen-Nürnberg, Germany, October 9-11, 2019*, pages 352–363.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A dictionary-based approach to racism detection in Dutch social media. In *TA-COS 2016 — Proceedings of the Workshop on Text Analytics for Cybersecurity and Online Safety @ LREC 2016. Portorož, Slovenia, 23 May 2016*, pages 11–17.

Yunli Wang, Yu Wu, Lili Mou, Zhoujun Li, and Wenhan Chao. 2019. Harnessing pre-trained neural networks with rules for formality style transfer. In *EMNLP-IJCNLP 2019 — Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing & 9th International Joint Conference on Natural Language Processing. Hong Kong, China, November 3-7, 2019*, pages 3573–3578.

Yunli Wang, Yu Wu, Lili Mou, Zhoujun Li, and Wenhan Chao. 2020. Formality style transfer with shared latent space. In *COLING 2020 — Proceedings of the 28th International Conference on Computational Linguistics. [Barcelona, Spain,] December 8-13, 2020 (Virtual Event)*, pages 2236–2249.

Michael Wiegand, Josef Ruppenhofer, Anna Schmidt, and Clayton Greenberg. 2018a. Inducing a lexicon of abusive words: a feature-based approach. In *NAACL-HLT 2018 — Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, Louisiana, USA, June 1-6, 2018*, volume 1: Long Papers, pages 1046–1056.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018b. Overview of the GERMEVAL 2018 Shared Task on the Identification of Offensive Language. In *Proceedings of the GermEval 2018 Workshop @ KONVENS 2018. Vienna, Austria, September 21, 2018*, pages 1–10.

Chen Wu, Xuancheng Ren, Fuli Luo, and Xu Sun. 2019. A hierarchical reinforced sequence operation method for unsupervised text style transfer. In *ACL 2019 — Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy, July 28 - August 2, 2019*, pages 4873–4883.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems 31 — NeurIPS 2018. Proceedings of the 32nd Annual Conference on Neural Information Processing Systems. Montréal, Québec, Canada, December 3-8, 2018*, pages 7298–7309.

Xiaoyuan Yi, Zhenghao Liu, Wenhao Li, and Maosong Sun. 2021. Text style transfer via learning style instance supported latent space. In *IJCAI-PRICAI 2020 — Proceedings of the 29th International Joint Conference on Artificial Intelligence & 17th Pacific Rim International Conference on Artificial Intelligence. [planned: Yokohama, Japan, 11-17 July 2020], 7-15 January 2021 (Virtual Event)*, pages 3801–3807.

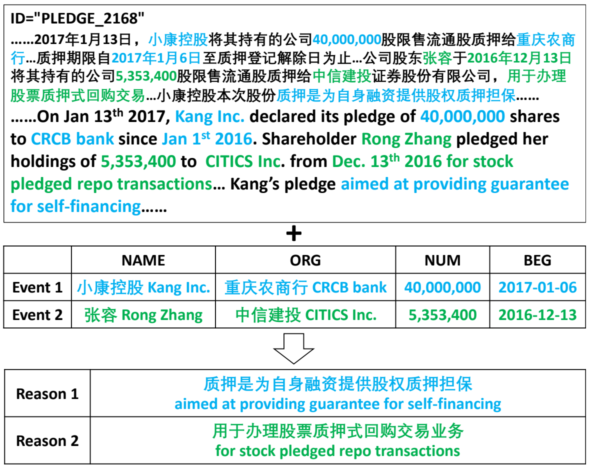
Yi Zhang, Tao Ge, and Xu Sun. 2020. Parallel data augmentation for formality style transfer. In *ACL 2020 – Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. [Seattle, Washington, USA,] July 5-10, 2020 (Virtual Event)*, pages 3221–3228.

Probing into the Root: A Dataset for Reason Extraction of Structural Events from Financial Documents

Pei Chen[1]*, **Kang Liu**[2,3],
Yubo Chen[2,3], **Taifeng Wang**[4], and **Jun Zhao**[2,3]
[1] Texas A&M University, College Station, TX
[2] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[3] University of Chinese Academy of Sciences, Beijing, China
[4] Ant Group, Hangzhou, China
`chenpei@tamu.edu, taifeng.wang@antgroup.com`
`{kliu, yubo.chen, jzhao}@nlpr.ia.ac.cn`

Abstract

This paper proposes a new task regarding event reason extraction from document-level texts. Unlike the previous causality detection task, we do not assign target events in the text but only provide structural event descriptions, and such settings accord more with practice scenarios. Moreover, we annotate a large dataset **FinReason** for evaluation, which provides **Reason**s annotation for **Fin**ancial events in company announcements. This task is challenging because the cases of multiple-events, multiple-reasons, and implicit-reasons are included. In total, FinReason contains 8,794 documents, 12,861 financial events and 11,006 reason spans. We also provide the performance of existing canonical methods in event extraction and machine reading comprehension on this task. The results show a 7 percentage point F1 score gap between the best model and human performance, and existing methods are far from resolving this problem.



ID="PLEDGE_2168"
……2017年1月13日，小康控股将其持有的公司40,000,000股限售流通股质押给重庆农商行…质押期限自2017年1月6日至质押登记解除日为止…公司股东张容于2016年12月13日将其持有的公司5,353,400股限售流通股质押给中信建投证券股份有限公司，用于办理股票质押式回购交易…小康控股本次股份质押是为自身融资提供股权质押担保……
……On Jan 13th 2017, Kang Inc. declared its pledge of 40,000,000 shares to CRCB bank since Jan 1st 2016. Shareholder Rong Zhang pledged her holdings of 5,353,400 to CITICS Inc. from Dec. 13th 2016 for stock pledged repo transactions… Kang’s pledge aimed at providing guarantee for self-financing……

+

	NAME	ORG	NUM	BEG
Event 1	小康控股 Kang Inc.	重庆农商行 CRCB bank	40,000,000	2017-01-06
Event 2	张容 Rong Zhang	中信建投 CITICS Inc.	5,353,400	2016-12-13

Reason 1	质押是为自身融资提供股权质押担保 aimed at providing guarantee for self-financing
Reason 2	用于办理股票质押式回购交易业务 for stock pledged repo transactions

Figure 1: An example of reason extraction for structural events from a document. We need to extract the textual spans from the document as reasons for the given structurally presented events. Here, we need to extract *Reason 1* and *Reason 2* for *Event 1* and *Event 2* respectively.

1 Introduction

Why does the event happen? People are always eager to find the reasons for an event. Automatically extracting the causal explanations of the given events from texts is useful and important for common users and downstream applications. For example, in the financial domain, returning the reasons of a concerned financial event in an Information Retrieval system can free analysts from reading the enormous company announcements and help investors make financial decisions.

Previous work on event causality (Do et al., 2011; Riaz and Girju, 2013; Mirza and Tonelli, 2014; Caselli and Vossen, 2017) mainly focus on the identification of causal relations between two given events that are usually presented as event trigger words. However, in reality, users may only know a particular event happened but without knowing its mention or trigger in the documents, and they just wonder the reasons for it. Therefore, we propose a new task aiming at extracting the causal explanations of the given structurally presented events from document-level texts. Specifically, a ***Structural Event*** defined here is a structural description that contains all necessary roles for an event type. Such a description can completely represent a specific happened event in reality. For example, in Figure 1, the *PLEDGE* event has four predefined roles *NAME*, *ORG*, *NUM*, *BEG* to represent an occurred *PLEDGE* event. Then, our

---

[*] Most of the work was done when the first author was a research engineer in the Institute of Automation, CAS.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2042–2048
April 19 - 23, 2021.

task is to extract the reasons of the structural events as textual spans from the document.

To investigate the solution for this challenging task, we construct a large-scale Chinese dataset **FinReason**[1]. Specifically, we automatically collect the formal financial documents with their corresponding structural events the same as Yang et al. (2018). Then, crowd workers are employed to annotate the reasons in the documents for each structural event. In order to guarantee annotation quality and high inter-annotator agreement (IAA), we set several annotation principles and define 3 types of possible causal explanations (*MOTIVATION*, *CAUSE*, *ENABLE*) as ***Reasons*** for the events in company announcements to guide annotators. Finally, there are 8,794 documents, 12,861 collected financial events, and 11,006 reason spans in total. The Cohen's kappa of annotations is 83.87%.

Moreover, to understand this task's difficulties, we regard this task as an Event Extraction (EE) or a Machine Reading Comprehension (MRC) task. We also try some canonical models, such as BiLSRM+CRF (Ma and Hovy, 2016), and BERT (Devlin et al., 2019) on this task and set benchmarks. Empirical results show that this task is challenging, and there is still an overall gap of 7pp (percentage points) in the F1 score between the best model and human performance.

2 Related Work

Much NLP research has focused on identifying causality relations from text, including knowledge bases (WordNet (Miller, 1998), FrameNet (Baker et al., 1998) and ConceptNet (Speer et al., 2017)), semantic related evaluations (SemEval-2007 task 04 (Girju et al., 2007), COPA (Roemmele et al., 2011), RED (Ikuta et al., 2014)), and event-related systems (Beamer and Girju, 2009; Do et al., 2011; Riaz and Girju, 2013; Hu and Walker, 2017; Caselli and Vossen, 2017). These work tried to identify real-world causality in lexicons or texts from different aspects. However, they have found it is difficult to agree on if a causal relationship exists in reality due to the ambiguity of causality definition. Our dataset mitigates this problem by only identifying contextual causality and do not check with reality.

In addition, plenty of work also only identify context-level causal relationships, such as general causality detection tasks PDTB (Prasad et al., 2007) and BECauSE 2.0 (Dunietz et al., 2017), and emotion causality detection task ECA (Lee et al., 2010). Some work (Radinsky et al., 2012; Mirza and Tonelli, 2014; Zhao et al., 2017) also tries to identify the causal relations between events at the contextual-level. However, our task is different because we focus on extracting the reasons for well-defined structural events, which is more close to practice scenarios.

3 Task Description and Data Collection

Task Description Our task is to extract the corresponding causal explanations for given structural events in a document. The inputs are a document with corresponding structural events described in it. The outputs are the causal text spans for the given events. For a given event in the document, there may be zero, single, or multiple causal explanations that need to be identified.

3.1 Data Collection

Event Type	Doc Count	Event Count	Reason Count	Doc Count w/ reason
Pledge	4,138	5,379	4,714	2,901 (70.11%)
O/U	2,550	4,127	3,565	2,132 (83.61%)
Lawsuit	2,106	3,355	2,727	1,438 (68.28%)
Total	8,794	12,861	11,006	6,471 (73.58%)

Table 1: Statistics of FinReason[2].

To construct this dataset, we first collect a corpus of structural events with their corresponding documents following Yang et al. (2018). The collected documents are constrained to company financial announcements, which are relatively formal documents. Such a setting could improve annotation IAA because of the logical consistency and clarity. In specific, we crawl the public company financial announcements as documents from sohu.com[3] and the structural events from eastmoney.com[4]. Since the documents are not in line with their corresponding structural events, we leverage key event items (see more details in Appendix B) matching to align them. Same as Yang et al. (2018), we assume that if the key event items of a structural event appear in a document, the document mentions the target structural event. This alignment method has a high precision of 94.5% as evaluated by Yang et al. (2018).

---

[1] `http://www.nlpr.ia.ac.cn/cip/~liukang/dataset/finreason1.html`

[2] The statistics are calculated after manual cleaning in the second step.

[3] `http://q.stock.sohu.com/index.shtml`

[4] `http://choice.eastmoney.com/`

In total, as in Table 5, we align 8,794 documents with corresponding 12,861 structural events of 3 types in financial domain, namely Pledge of Shares *(Pledge)*, Overweight and Underweight of Shares *(O/U)*, Lawsuit and Arbitration *(Lawsuit)*. To the best of our knowledge, this is the largest dataset in the event reason extraction task.

3.2 Event Reason Annotation

Annotation Principles: To construct a corresponding dataset with high IAA, we follow the two principles in the annotation. **First**, we annotate the event reasons according to the contextual expressions. We do not check with reality, even if it is obviously a false statement (*e.g., the stock market falls because of intense sunspot activity*). **Second**, we specifically define 3 types of possible causal explanations as reasons for the financial events in announcements following previous work (Trabasso et al., 1989; Van den Broek, 1990; Dunietz et al., 2015): *MOTIVATION*, *CAUSE*, *ENABLE* (see details in Appendix A). This provides a clear guideline to annotators to decide what to annotate and what not to. Because it is also ambitious to differentiate those reason types from texts (Dunietz et al., 2015), we do not require the annotators to distinguish them but just require them to confirm that the reasons annotated at least belong to one of the 3 types.

Quality Control: Besides the aforementioned 2 principles, we adopt several more rules to control data quality as follows. (1) Each member should find as many reasons for a target event as possible. (2) Each reason annotated should be as short as possible but with complete expressivity. (3) When explicit causal relation terms such as 因为 (*because*), 为了 (*in order to*) are mentioned, they should be included in the annotated reasons. (4) For each reason annotated, the annotator should confirm it by doing a *why* test (Grivaz, 2010), which means the reason should answer the question *why the event happened*.

Then, we employ crowdsourcing to annotate the reasons for each event. Specifically, 9 workers are divided into 3 teams to annotate each event type separately. Each team is trained to acquire the domain knowledge of the target event type so they can figure out the possible reasons for the events. Within each team, 2 members are responsible for annotating the reasons independently, and the 3rd member will be activated to make a judgment when 2 annotators have inconsistent annotations. Because the alignment in the first step may not be perfectly accurate, annotators are also responsible for removing those wrongly aligned cases in the annotation to maintain data quality. Finally, as shown in Table 5, there are totally 8,794 documents, 12,861 collected financial events and 11,006 annotated event reason. And, approximately 73.58% of the documents are annotated with event reason. The Cohen's kappa of IAA is 83.87%.

3.3 Task Challenges

Event Type	Multi-event Doc Count	Multi-reason Doc Count	Implicit-reason Doc Count
Pledge	796 (19.24%)	461 (11.14%)	2,845 (68.75%)
O/U	635 (24.90%)	483 (18.94%)	2,030 (79.61%)
Lawsuit	387 (18.38%)	221 (10.49%)	1,434 (68.09%)
Total	1,818 (20.67%)	1,165 (13.25%)	6,309 (71.74%)

Table 2: Three types of challenges in FinReason.

From the annotation results, we could briefly conclude that extracting the reasons for given structural events in a document is not an easy task. First, a document may mention multiple events like the 2 events in the example of Figure 1. As in Table 5, approximately 20.67% documents mention more than one events. Without event mention assignment, discriminating the corresponding reasons for different events within the same document is difficult. Second, about 13.25% of documents mention multiple reasons for an event. Finding all reasons out is also not easy. Thirdly, 71.74% of the documents mention the reason for the events in an implicit way. There are only 28.26% reasons mentioned with explicit modifiers, like 因为(because),由于(since),原因(cause),为(in order to),目的(aims to), etc. Such implicitly mentioned reasons are harder to be identified because they do not have any syntactic clue and require deep reasoning.

We regard the average performance of the two annotators with respect to the final golden standard in the test set as human performance[5]. We can see in Table 2, the human performance on the test set is in line with intuition. Compared with simple cases (Single-Event, Single-Reason, Explicit-Reason), identifying reasons in multiple-event, multiple-reason, and implicit-reason cases are more challenging.

---

[5] This setting may overestimate the actual human performance but acceptable as a performance upper bound.

4 Benchmark Settings

4.1 Evaluation Criterion

To evaluate the solution on this task, we follow a similar paradigm of SQuAD 2.0 (Rajpurkar et al., 2018) but also with several differences. In general, we get *precision/recall/f1* scores of every event in the test set and calculate the macro-average of all events as the overall performance. However, there are multiple reason cases in FinReason, and we try to evaluate the ability of multiple reasons identification. As a result, we do not fully follow SQuAD-style evaluation by selecting the best prediction but considering all predictions to avoid the systems cheating by predicting all possible causal expressions in the text. For each case, we compute the scores as follows. 1) When there is no reason annotated for an event, the prediction should be *Null* string so as to get *precision/recall/f1* scores of all 1; otherwise, all scores will be assigned 0. 2) When there is only one reason annotated for an event, we calculate the *precision/recall/f1* scores based on the overlapping strings of prediction and ground truth. 3) When there are multiple reasons for a target event, we first calculate each reason's scores with corresponding predictions as in situation 2 and then calculate the macro-average of all reasons as the final scores for the target event.

4.2 Baselines

FinReaon is a new task but similar to several existing tasks, such as event extraction (EE) or machine reading comprehension (MRC). So we apply existing canonical methods for those similar tasks on FinReason as benchmarks for future research. The selected baselines are as followed (see more details in Appendix C):

Regular Expressions (RegExp): In this setting, we regard the FinReason task as a causal sentence detection problem and employ some ad-hoc regular expressions to solve it. Specifically, we use five modifiers (因为(because),由于(since),原因(cause),为(in order to),目的(aims to)) as causal clues to detect the sentence as the reasons for an event.

BiLSTM-CRF (BiLSTM): We can take the reasons as one part of the event description and regard the task as an EE task. Similar to Yang et al. (2018), we employ a BiLSTM-CRF (Ma and Hovy, 2016) to predict the start and end positions of each reason. Specifically, We simply get the event participants in the documents via string matching between the documents and the given structural events. Such information is used as features in a BIO tagging format.

BERT-QA: We can take this task as an MRC problem if the structural event is regarded as a query and the target reason as the answer. In particular, we use templates to turn each structural event into a *why-question* and employ BERT-QA (Devlin et al., 2019) model to find the corresponding reasons.

Type	RegExp	BiLSTM	BERT-QA	Human
Pledge	19/21/20	**76/86/81**	**76**/70/73	93/94/93
O/U	20/27/23	**90/94/92**	**90**/89/89	99/99/99
Lawsuit	20/24/22	**73/73/73**	**73**/72/72	74/78/76
All	20/24/22	**80/84/82**	**80**/77/78	89/90/89

Table 3: Performance of baselines and human beings on FinReason (*precision/recall/f1*, %).

Challenges	RegExp	BiLSTM	BERT-QA	Human
Single-event	16/21/18	**86/90/88**	84/81/82	90/92/91
Multi-event	25/28/26	73/**77/75**	**74**/72/73	87/88/87
Single-reason	23/26/24	85/**86/85**	**86**/84/**85**	91/92/91
Multi-reason	8/13/10	**53/81/64**	43/41/42	76/85/80
Explicit-reason	29/32/30	**85/87/86**	**85**/83/84	90/90/90
Implicit-reason	2/4/3	54/**65/59**	**61**/58/**59**	85/90/87

Table 4: Performance for the three challenges (*precision/recall/f1*, %).

Results We split the dataset into *train/dev/test* sets with a ratio of 8:1:1 for experiments. From the results in Table 2, we can see that there is still an average of 7pp (82% vs. 89%) F1 score gap between the best model (BiLSTM) and human performance. Besides, we can see that human performance is relatively low for *Lawsuit*. This is because the reason for a lawsuit usually lies in a whole story between the plaintiff and the defendant, and it is hard to agree on the boundaries of the span. Furthermore, the BiLSTM model generally performs better than BERT-QA. The reason may be that the BiLSTM model knows the positions of event mentions by using the event BIO features, but BERT-QA only uses the structural event as the query. So it may be easier for BiLSTM to locate the correct reasons.

Besides, we also evaluate the 3 challenges on the whole test set. As from Table 2, the average F1 gap between the best model and human for the 3 challenges are 12pp, 16pp, 28pp, respectively, which is much larger than the overall average gap of 7pp. This demonstrates that the challenges are also the bottlenecks of the models to reach comparable human performance, especially the implicit cases.

5 Conclusion

In this work, we propose a dataset FinReason for a new event causality extraction task. Our experiments show that this task is still challenging for current models. Future work may consider breaking the challenging cases (multiple-events, multiple-reasons, and implicit-reasons) to achieve a more satisfying performance.

Acknowledgments

This work is supported by the National Key RD Program of China under Grant 2018YFB1005100, the National Natural Science Foundation of China (No.61922085, No.61831022, No.61806201), This work is also supported by Beijing Academy of Artificial Intelligence(BAAI). We would like to thank the anonymous reviewers for their valuable feedback.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Brandon Beamer and Roxana Girju. 2009. Using a bigram event model to predict causal potential. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 430–441. Springer.

Paul Van den Broek. 1990. The causal inference maker: Towards a process model of inference generation in text comprehension. *Comprehension processes in reading*, pages 423–445.

Tommaso Caselli and Piek Vossen. 2017. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86.

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

Quang Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Jesse Dunietz, Lori Levin, and Jaime G Carbonell. 2015. Annotating causal language using corpus lexicography of constructions. In *Proceedings of The 9th Linguistic Annotation Workshop*, pages 188–196.

Jesse Dunietz, Lori Levin, and Jaime G Carbonell. 2017. The because corpus 2.0: Annotating causality and overlapping relations. In *Proceedings of the 11th Linguistic Annotation Workshop*, pages 95–104.

Roxana Girju, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney, and Deniz Yuret. 2007. Semeval-2007 task 04: Classification of semantic relations between nominals. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 13–18.

Cécile Grivaz. 2010. Human judgements on causation in French texts. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Languages Resources Association (ELRA).

Zhichao Hu and Marilyn Walker. 2017. Inferring narrative causality between event pairs in films. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 342–351, Saarbrücken, Germany. Association for Computational Linguistics.

Rei Ikuta, Will Styler, Mariah Hamang, Tim O'Gorman, and Martha Palmer. 2014. Challenges of adding causation to richer event descriptions. In *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 12–20.

Sophia Yat Mei Lee, Ying Chen, and Chu-Ren Huang. 2010. A text-driven rule-based system for emotion cause detection. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 45–53. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Paramita Mirza and Sara Tonelli. 2014. An analysis of causality between events and its relation to temporal information. In *Proceedings of COLING 2014,* *the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2097–2106.

Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind Joshi, Livio Robaldo, and Bonnie L Webber. 2007. The penn discourse treebank 2.0 annotation manual.

Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. 2012. Learning causality for news events prediction. In *Proceedings of the 21st international conference on World Wide Web*, pages 909–918. ACM.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Tom Trabasso, Paul Van den Broek, and So Young Suh. 1989. Logical necessity and transitivity of causal relations in stories. *Discourse processes*, 12(1):1–25.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55.

Sendong Zhao, Quan Wang, Sean Massung, Bing Qin, Ting Liu, Bin Wang, and ChengXiang Zhai. 2017. Constructing and embedding abstract event causality networks from text snippets. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 335–344. ACM.

Appendices

Appendix A: Event Causality Types

We define the Event Causality Types referring to some previous work. Trabasso et al. (1989) and Van den Broek (1990) defined 4 types of narrative causality relations between events.

- *PHYSICAL*: Event A physically causes event B to happen.
- *MOTIVATIONAL*: Event A happens with B as a motivation.
- *PSYCHOLOGICAL*: Event A brings about emotions expressed in event B.
- *ENABLING*: Event A creates a state or condition for B to happen.

Dunietz et al. (2015) also defined 4 types of causal languages in texts:

- *CONSEQUENCE*: The cause naturally leads to the effect via some chain of events.
- *MOTIVATION*: Some agent perceives the cause, and therefore consciously thinks, feels, or chooses something.
- *PURPOSE*: An agent chooses the effect out of a desire to make the cause true.
- *INFERENCE*: Present the cause as evidence or justification for the effect.

However, we refer to those previous definitions, but we only define 3 types of causal explanations as **Reasons** according to our specific task and application domain. These causality types are the most common reasons for the financial events described in company announcements.

- *MOTIVATION*: The event happens with the explanation as a motivation or purpose. *e.g., He pledged the stocks aiming at providing a guarantee for self-financing.*
- *CAUSE*: The cause in the explanation naturally leads to the occurring of the event. *e.g., He sued the company because of loan disputes.*
- *ENABLE*: The explanation creates a condition or state for the event's occurrence. *e.g., He reduced his shares according to the contract.*

The reasons annotated should at least belong to one of the 3 types. However, we do not require the annotators to distinguish specific types because it is ambitious to differentiate them (Dunietz et al., 2015).

Appendix B: Event-Document Alignment

Following Yang et al. (2018), we assume that if the key items of a structured event appear in a document, then the document mentions the event. We first group those events and documents with the same Announcement Date (DATE[6]) and then use the following key items to align them:

- *Pledge*: Number of Shares (NUM), Name of Shareholder (NAME), Pledge Institution (ORG).

---

[6] The event item abbreviations are used in the corpus.

- *O/U*: Number of Shares (NUM), Name of Shareholder (NAME), Name of Shares (STOCK).
- *Lawsuit*: Plaintiff (OBG), Defendant (NAME), Court Name (ORG).

Appendix C: Model Settings

We build our baselines based on the open sources of the BiLSTM[7] and BERT-QA[8]. We do not do hyperparameter search and mainly use the default settings. Common parameters are in the Table 5:

Models	Batch Size	Learning Rate	Max Epochs	Optimizer
BiLSTM	20	$1e^{-3}$	30	Adam
BERT-QA	16	$5e^{-5}$	30	BERTAdam

Table 5: Parameters Settings.

Specifically, for BiLSTM, we use the 100-dim character embeddings[9] trained with on Chinese WiKi corpus as initial word features. For structural event embeddings, we first label back the event items to corresponding documents by string matching with BIO schema, then use the 100-dim random vectors as initial BIO features. Besides, the hidden dimension and dropout rate of the LSTM are set as 100 and 0.5, respectively. For BERT-QA, we use the *bert-base-chinese* version and choose the maximum sequence length as 512, the document stride as 128, the max query length as 64, and the max answer length as 30. We train them on two Nvidia GEFORCE GTX 1080Ti GUPs.

Moreover, we need the models to deal with the multiple-event cases. For the RegExp method, we cannot distinguish different events, so we just regard all the extracted reasons as explanations for all the events indiscriminately. For BiLSTM, we create different samples for different events from the same documents to ensure one sample just have one document with at most one event. The BERT-QA regards events as queries so it can naturally adapt to it.

Even for one event, there may be multiple reasons for it, and our models need adaptation. The RegExp is the same as before. The BiLSTM can label multiple pairs of start and end for textual spans, so it naturally adapts to multiple-reason cases. The BERT-QA can return the top k answers as reasons from the documents, and in practice, we get the best result when setting k as 1.

---

[7] `https://github.com/zjy-ucas/ChineseNER`.

[8] `https://github.com/huggingface/transformers/tree/master/examples/\question-answering`.

[9] `https://github.com/zjy-ucas/ChineseNER/blob/master/wiki_100.utf8`

Language Modelling as a Multi-Task Problem

Lucas Weber
DTCL, University Pompeu Fabra
`lucas.weber@upf.edu`

Jaap Jumelet
ILLC, University of Amsterdam
`j.w.d.jumelet@uva.nl`

Elia Bruni
IKW, University of Osnabrück
`elia.bruni@gmail.com`

Dieuwke Hupkes
Facebook AI Research
`dieuwkehupkes@fb.com`

Abstract

In this paper, we propose to study language modelling as a multi-task problem, bringing together three strands of research: multi-task learning, linguistics, and interpretability. Based on hypotheses derived from linguistic theory, we investigate whether language models adhere to learning principles of multi-task learning during training. To showcase the idea, we analyse the generalisation behaviour of language models as they learn the linguistic concept of Negative Polarity Items (NPIs). Our experiments demonstrate that a multi-task setting naturally emerges *within* the objective of the more general task of language modelling. We argue that this insight is valuable for multi-task learning, linguistics and interpretability research and can lead to exciting new findings in all three domains.

1 Introduction

Humans are optimising their behaviour towards a multitude of objectives to reach their goals in day-to-day life. By learning many things at the same time and exploiting their commonalities, they acquire more general knowledge about the world, which in turn helps them to learn new things quicker (Perkins et al., 1992; Schwartz et al., 2005; Cormier and Hagman, 2014; Luriia, 1976). This idea of finding more general solutions through the diversification of tasks has found its way also to the machine learning community, in the field of multi-task learning (MTL) (Caruana, 1993, 1997). In MTL, multiple tasks are optimised jointly, enabling the transfer of relevant information across tasks. MTL research yields fruitful results in both application (e.g. Collobert and Weston, 2008; Collobert et al., 2011; Zhang et al., 2014; Donahue et al., 2014; Kaiser et al., 2017) and theory (e.g. Baxter, 2000; Maurer, 2006; Ando and Zhang, 2005; Argyriou et al., 2008).

However, deciding on a setup requires making many arbitrary choices. The researcher or engineer has to decide which tasks to train together (e.g. Bingel and Søgaard, 2017; Standley et al., 2020); at which hierarchy-level to allow tasks to interact (e.g. Søgaard and Goldberg, 2016); which degree of parameter sharing to employ (Ruder, 2017); which distribution of training data to employ (e.g. Luong et al., 2016), and so on. Having to make so many arbitrary choices is inconvenient for modellers, but also stands in the way of understanding the learning principles of neural models in multi-task settings. The highly constructed learning scenarios make it difficult to see whether outcomes should be attributed to one of the many a-priori decisions or to inherent properties of the learning process.

In this paper, we propose to study MTL not in a constructed, artificial scenario, but in a more natural setting. To do so, we consider the objective of *language modelling* and exploit the fact that it can be seen as a conglomerate of many different tasks. To give an example: rules of word ordering have to be learned simultaneously to rules of feature agreement and the monotonicity properties of different linguistic environments. These different tasks all need to be learned to achieve the greater goal of producing acceptable sentences, and they have to be optimised in parallel when the language model (LM) is trained. Language modelling is in that sense a *natural* multi-task learning problem with a naturally given *task hierarchy* provided by linguistic theory (see also Figure 1).

Studying language modelling as a multi-task problem has several distinct advantages. From an MTL perspective, it gives us a complete hierarchy of relevant tasks that can freely interact throughout the learning process, unconstrained by prior assumptions. We can make theoretically informed decisions about these tasks, drawing on linguistic theory. We can also deduce from linguistics how these tasks relate to each other (or, in other words, how similar they are), which in MTL is considered to be one of the crucial factors for the learn-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2049–2060
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

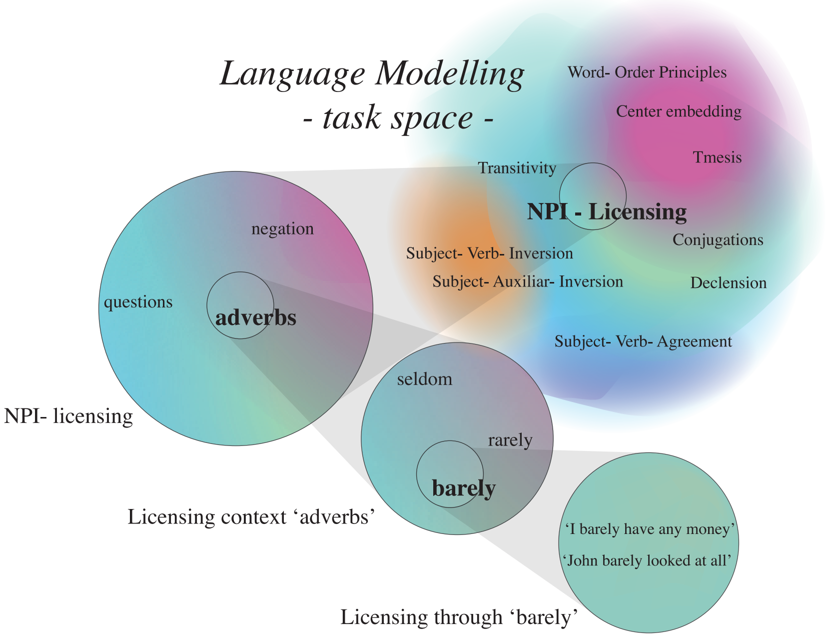


Figure 1: A conceptual visualisation of a language modelling task hierarchy, from language modelling as a whole to single examples, with complex similarities between tasks. Colours indicate task similarities.

ing outcomes (e.g. Thrun and O'Sullivan, 1996; Passos et al., 2012). MTL has not yet been studied from this dynamic and unconstrained perspective. Then, somewhat more delicately, the extent to which models can exploit similarities hypothesised by linguistic theory can play a role in confirming or refuting specific linguistic hypotheses. Lastly, when it comes to interpretability research, applying concepts from MTL can be valuable to better understand the learning dynamics of models. By understanding how models are finding solutions, we can infer what these solutions are.

Outline In the remainder of this paper, we will first provide some basic background about MTL (§ 2.1), the subset of linguistic tasks we focus on (*Negative Polarity Items*, where we consider their different *licensing contexts* as tasks, § 2.2) and discuss some related work in interpretability (§ 2.3). Then, in § 3 and § 4, respectively, we present our approach and empirical results that showcase our idea. In § 5, we discuss our results and framework in the light of the three fields mentioned before. We conclude in § 6.

2 Background

In this paper, we aim to bring together three strands of research: MTL, linguistics and interpretability research. As a proof of concept, we focus on one specific complex subset of linguistic tasks: *licensing of Negative Polarity Items (NPIs)*. Below, we give a short overview of the most important characteristics of the three fields of interest.

2.1 Multi-task learning

In MTL, multiple tasks are learned together to enable information transfer from one task to another. If the transfer is successful, the benefits might be threefold: the model learns tasks with less training data (i.e. *more efficient*, Collobert et al., 2011; Benton et al., 2017; Kaiser et al., 2017), up to a higher final accuracy (Collobert and Weston, 2008; Kaiser et al., 2017) and in a way that better generalises to new tasks (Baxter, 2000; Collobert and Weston, 2008).

Caruana (1993, 1997) and Ruder (2017) propose several different – but related – processes that might enable positive transfer: related tasks can provide additional training examples for each other on the features they share (*statistical data amplification*), certain features might be easier to learn through one task than through another, but be useful for both of them (*eavesdropping*), and idiosyncratic features of single tasks can be averaged out, while more general features are reinforced (*attention focusing*)[1].

---

[1] For a complete list of processes please consult the original publications.

However, positive transfer is not guaranteed; It is also possible that performance *deteriorates* due to interference between different tasks, resulting in negative transfer, (Rosenstein et al., 2005; Pan and Yang, 2010; Wang et al., 2019). Whether transfer is positive depends on the *task similarity* and whether the model is able to exploit this similarity (Rosenstein et al., 2005; Thrun and O'Sullivan, 1996; Passos et al., 2012).

The main goal of MTL so far has been to avoid negative- and promote positive transfer by determining task-similarity and regulate the interactions between tasks based on these similarities. Due to its pivotal role, much research effort was spent on determining similarities of tasks and the regulation of information transfer between them (for an overview, see Zhang and Yang, 2017; Ruder, 2017). The disadvantage of these approaches is that assuming fixed tasks and regulating transfer between them based on fixed task-similarities puts large constraints on possible transfers between tasks, because it neglects the fact that learning processes are dynamic. From the perspective of the model, tasks, as well as their similarities, can change throughout the learning process. Here, we only use predefined tasks and their similarities to *analyse* the learning behaviour of the model, without constraining the learning process in any way.

2.2 Negative Polarity Items

We exemplify our idea by analysing the learning behaviour on a complex subset of linguistic tasks: the licensing of Negative Polarity Items (NPIs). The properties of NPI licensing make it an interesting and adequate subset of tasks to study, as it has a high degree of complexity, has an appropriate frequency within natural language and was previously frequently investigated in neural models.

NPIs are characterised by the property that they can only occur within the scope of certain *licensing contexts*. For instance, in the example below, the NPI '**any**' can occur in sentence (1)a., where it is in the scope of a negation, but not in sentence (1)b., where there is no licensor present.

(1) a. Bill did*n't* buy **any** books that day.
b. * Bill did buy **any** books that day.

(2) a. *Nobody* has **ever** been there.
b. * Somebody has **ever** been there.

Licensing contexts are formed on the basis of semantic properties, such as downward entailment (Fauconnier, 1975; Ladusaw, 1980), nonveridicality (Giannakidou, 2011), or scope marking (Barker, 2018). Common licensing contexts include negation, conditionals, or superlatives, and are often *triggered* by a specific expression, such as 'not' or 'nobody'.

Grasping the phenomenon of NPI licensing requires understanding of three different aspects:

1. *The class of NPIs*: there is a group of expressions that are restricted in their occurrence.
2. *Licensing contexts*: there exists a group of expressions that allow NPIs to occur.
3. *Scope and structure*: the licensing contexts have to stand in a certain structural relationship to the NPIs.

We focus on how LMs learn the second aspect by analysing how different types of licensing contexts interact and generalize throughout training. During learning they should be able to exploit their similarity in the other two aspects.

2.3 Interpretability

Interpretability research on LMs has shown that in pre-trained models, such as BERT (Devlin et al., 2019), hierarchical structure emerges throughout the layers and that this structure demonstrates parallels with linguistic theory (Peters et al., 2018; Liu et al., 2019; Tenney et al., 2019). However, the emergence of this structure has not been explicitly connected to MTL yet.

In recent years, research has shown that LMs are able to understand NPI licensing. Jumelet and Hupkes (2018) evaluate the performance of LMs on data sets containing NPI constructions extracted from large corpora, and Marvin and Linzen (2018); Wilcox et al. (2019); Warstadt and Bowman (2020) test them on artificial data sets containing template-based NPI constructions. In our own experimental setup we will utilise the extensive template-based NPI corpus of Warstadt et al. (2019).

What these approaches have in common is their focus on the performance of pretrained LMs. Our MTL approach sheds light on an unexplored aspect of NPI understanding: the learning dynamics of the model *during* training.

3 Approach

We consider two different types of experiments. First, to understand to which extent models can understand and use the similarity between different licensing contexts (our *tasks*) during learning, we exploit the effect that frequency of the different contexts has on learning. Second, we manipulate the LMs' training corpus to constrain their ability to leverage information from other licensing contexts during learning. In accordance with the MTL-literature, we expect the LMs to learn tasks more data-efficient and to a higher final accuracy if they can leverage information across contexts. Before we describe our experiments in more detail, we present our model architecture and training, the evaluation procedure of the licensing contexts, and the filter procedure we use to manipulate the training corpus.

3.1 Model

Following previous work in this area, we consider recurrent language models. We focus on unidirectional LSTM models and mirror the hyperparameter setup of Gulordava et al. (2018)[2]. We train the models on the corpus provided by the same authors[3] – a subset of the English Wikipedia – or modified versions of the same for our second experiment (see § 4.2). To track the learning process, we save models every 100 batches of training (371 model-checkpoints per epoch). For all experiments, we average performance across five random seeds.

3.2 Evaluation

To estimate the LMs' understanding of NPIs and their dependence on the different licensing contexts, we adapt the Cloze task of Warstadt et al. (2019), based on the implementation of Jumelet (2020). This task considers nine different types of licensing contexts (a list of the contexts, including examples, can be found in Table 1). For every such context, Warstadt et al. (2019) generated a large number of *minimal pair sentences*, containing correctly and incorrectly licensed NPIs. For instance, for the *adverbs* licensing context:

(3) a. A lady *rarely* **ever** thought that the children saw the boy.
b. * A lady *sometimes* **ever** thought that the children saw the boy.

Following previous work, we quantify an LM's understanding of a particular type of licensing context by computing the percentage of minimal pairs in that context for which the model correctly assigns a higher probability to the NPI in the licensing contexts than in the non-licensing contexts. I.e., in the example above, we would compare the probability the model assigns to the word *ever* in the contexts "A lady rarely" and "A lady sometimes" (see also Figure 2).

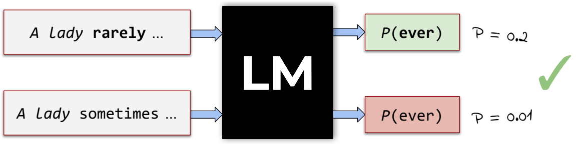


Figure 2: The NPI judgement task that is used for evaluating the LMs. A correct prediction assigns a higher probability to an NPI in a context that licenses it, based on the corpus of Warstadt et al. (2019).

3.3 Identification of NPIs in training corpus

The Warstadt et al. (2019) corpus provides us with a task to evaluate nine different context types that license NPIs. To manipulate the training corpus for our experiments we also need to identify sentences in the training corpus of the model in which these contexts actually licence NPIs. To do so, we need to locate these contexts, as well as establish that they in fact licence an NPI in a particular sentence.

We consider the nine Warstadt et al. context types, and the corresponding list of 30 expressions that are part of these contexts (e.g. the list of adverbs licensing NPIs). As for the NPIs, we consider an extensive list of 160 distinct NPIs[4], based on the collection provided by Hoeksema (2012). We then identify sentences in which an element of our NPI list is preceded by an element from our context list, ensuring that there is a dependency relation between them using the dependency parser of spaCy (Honnibal and Johnson, 2015). When there are multiple potential licensors in a sentence, we use the hierarchical distance between the licensor and the NPI in the parse tree as a heuristic to find the correct licensor. By testing this procedure on a manually labeled set of 200 randomly selected sentences with multiple licensors, we estimate that it identifies the correct among multiple licensors in around 97% of cases. In Table 1, we report examples and frequencies of the different licensing contexts in the training corpus based on this filtering scheme.

---

[2] Hyperparameters: batch size = 64, BPTT length = 35, dropout = 0.1, adaptive SGD learning rate = 20, layers = 2, hidden and embedding size = 650, epochs = 40.

[3] `https://github.com/facebookresearch/colorlessgreenRNNs/tree/master/data`

[4] This list can be found in Appendix A.

Context	**Example**	**Frequency** per 100k sentences
Simple Questions	Did he **ever** do a mean thing*?*	10
Adverbs	In the present political culture, there are *hardly* **any** leaders who would avoid limelight and refuse positions of power.	23
Questions	However, various writers attribute it to Putnam, Stark, Prescott or Gridley, while others question *whether* it was said **at all** .	25
Superlative	[...] and caused the *worst* winter flooding **in decades** for river and stream valleys [...].	32
Only	[...] ”Those [students] *only* are supposed to pay **anything** who are abundantly able, or prefer to do so.	85
Conditional	In 1997 Li published a paper attempting to replicate `<unk>`’s results and showed the effect was very small, *if* it existed **at all**.	127
Quantifier	That’s *all* you’ll **ever** need.	179
Determiner negation	In spite of the `<unk>` of the disaster, *no* one was **ever** held accountable.	218
Sentential negation	It is *not* judged under **any** subjective points of view, only the clock.	712

Table 1: The nine types of licensing contexts taken from Warstadt et al. (2019), with an example and the context frequency within the training corpus.

4 Experiments and results

As a first step, we assess whether the LMs can adequately represent all nine categories of the evaluation task. To do so, we train five models on the regular training corpus, and compute their final accuracy on our nine tasks. All models show adequate performance on most contexts (see Table 2), with the exception of the simple question context. Additionally, we observe that the models achieve their accuracy surprisingly fast: already after two epochs, there are no more substantial changes in empirical error (see Figure 3). In the rest of our experiments, we therefore focus only on these first two epochs.

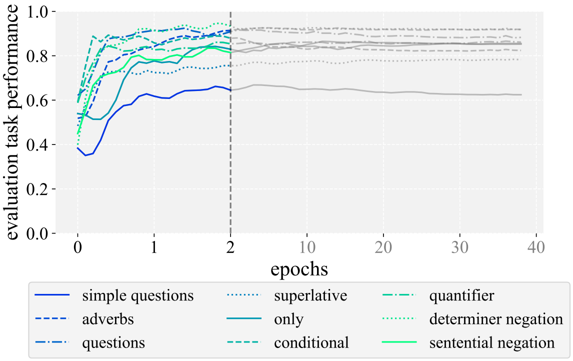


Figure 3: Average evaluation task performance. The performance rises steeply during approximately the first 2 epochs of training and afterwards levels off.

Context	**Accuracy** ± std
Simple Questions	0.62 ± 0.05
Adverbs	0.92 ± 0.01
Questions	0.88 ± 0.03
Superlative	0.78 ± 0.03
Only	0.86 ± 0.04
Conditional	0.82 ± 0.06
Quantifier	0.86 ± 0.04
Determiner negation	0.92 ± 0.05
Sentential negation	0.85 ± 0.03

Table 2: Performance of the LMs on the evaluation task after 40 epochs of training, averaged over 5 runs.

4.1 Frequency vs data efficiency

While some licensing contexts are rather common (e.g. negation), others appear scarcely as a licensor (e.g. adverbs). Therefore, throughout the learning process, the LMs encounter many instances of the more frequent contexts before they see an example of an infrequent context. If LMs were able to leverage information across contexts, less frequent contexts should thus have more prior established NPI-understanding that they can bootstrap from. Consequently, the LMs should require fewer training examples to learn less frequent contexts than they need to learn more frequent contexts. In other words, the LM should be more *data efficient* for these infrequent contexts.

In our first experiment, we use this hypothesised relationship between frequency and data efficiency to assess whether LMs can exploit the similarities between different licensing contexts. To be able to compare across different contexts, we quantify the data efficiency of an LM for a particular context as the number of examples the LM needs to observe until it reaches 95% of its final accuracy for that context.[5] To make this measure more robust, we first apply a Savitzky–Golay noise-filter to the learning curve (degree of polynomial = 1, window size = 25; Savitzky and Golay 1964).

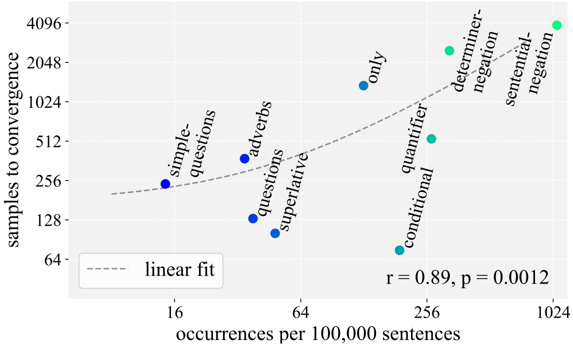


Figure 4: Data efficiency of nine different licensing contexts plotted against their frequency, averaged over five runs. The data efficiency is quantified as number of training examples the model needs to observe to achieve 95% of the trained-out performance.

We compute the data efficiency of the trained LMs for all nine contexts and compute the correlation between a context's frequency and the model's data efficiency with respect to that context. In Figure 4, we plot the average data efficiency of each context against the frequency of that context, as well as the linear fit that relates these two variables. The experiment demonstrates a strong relationship between the data efficiency and frequency of a respective context: $r = .89$, $p < .05$. Hence, the less frequent a licensing context is, the fewer examples are needed for the model to learn it, from which we conclude that the model is indeed able to transfer knowledge from previously acquired knowledge.

4.2 Transfer from general knowledge

While the presented relationship between frequency and data efficiency demonstrates that LMs can leverage previously learned information to learn less frequent licensing contexts, it does not unequivocally show that it leverages information from *other NPI contexts*. After all, when a less frequent context is encountered, the LM has not only had the opportunity to acquire prior knowledge about NPIs, it has also simply seen more language in general. In other words, the LM may meanwhile also have acquired more *general language knowledge*, which may help it to more quickly learn a less frequent licensing context. In our second experiment, we isolate transfer from general language knowledge and transfer from previously observed NPIs by training LMs on *single-context* corpora.

Single-context corpora *Single-context corpora* contain NPIs licensed only by a single context. LMs trained on these corpora can thus not transfer knowledge acquired from other licensing contexts, as these are not present in the training data. By comparing the data efficiency of contexts between LMs trained on all-context and single-context corpora, we can thus infer how much of the increase of data efficiency for lower-frequent contexts is due to leveraging information from other contexts.

To create our nine single-context corpora, we use the procedure described in § 3.3 to identify all sentences containing NPIs licensed by our nine contexts. For every context, we then create a corpus in which all sentences containing other contexts licensing NPIs are replaced by a neutral sentence of the same length, sampled from the rest of the corpus. During this replacement procedure, the ordering and composition of the corpus remained otherwise intact.

When we compare the learning of single-context with all-context models, we cannot rely on the previously used data-efficiency metric from Experiment 4.1. The data-efficiency measure is bound to how quickly the model reaches its final accuracy and accordingly benefits when its final accuracy decreases. As we expect the final accuracy to be lower in the single context models, comparing only data-efficiencies between models is likely to be uninformative.[6]. In this experiment, as explained below, we instead consider the area between the curves (AbC).

Area between Curves (AbC) *Area between Curves* (AbC) incorporates both data efficiency and accuracy: for every context, we calculate the area between the all-contexts and single-context learning curves until the point in time where they both have reached 95% of their final accuracy. The larger this area is, the more impactful it is to remove

---

[5] The *more* data efficient, the *lower* this number thus is.

[6] Consider, for instance, the extreme case in which an LM does not learn a particular context at all anymore in the single-context condition, as indicated by a chance accuracy of 0.5. Because it is not learning anything, the model would arrive at its maximum accuracy before having seen any examples, resulting in a data efficiency of 0.

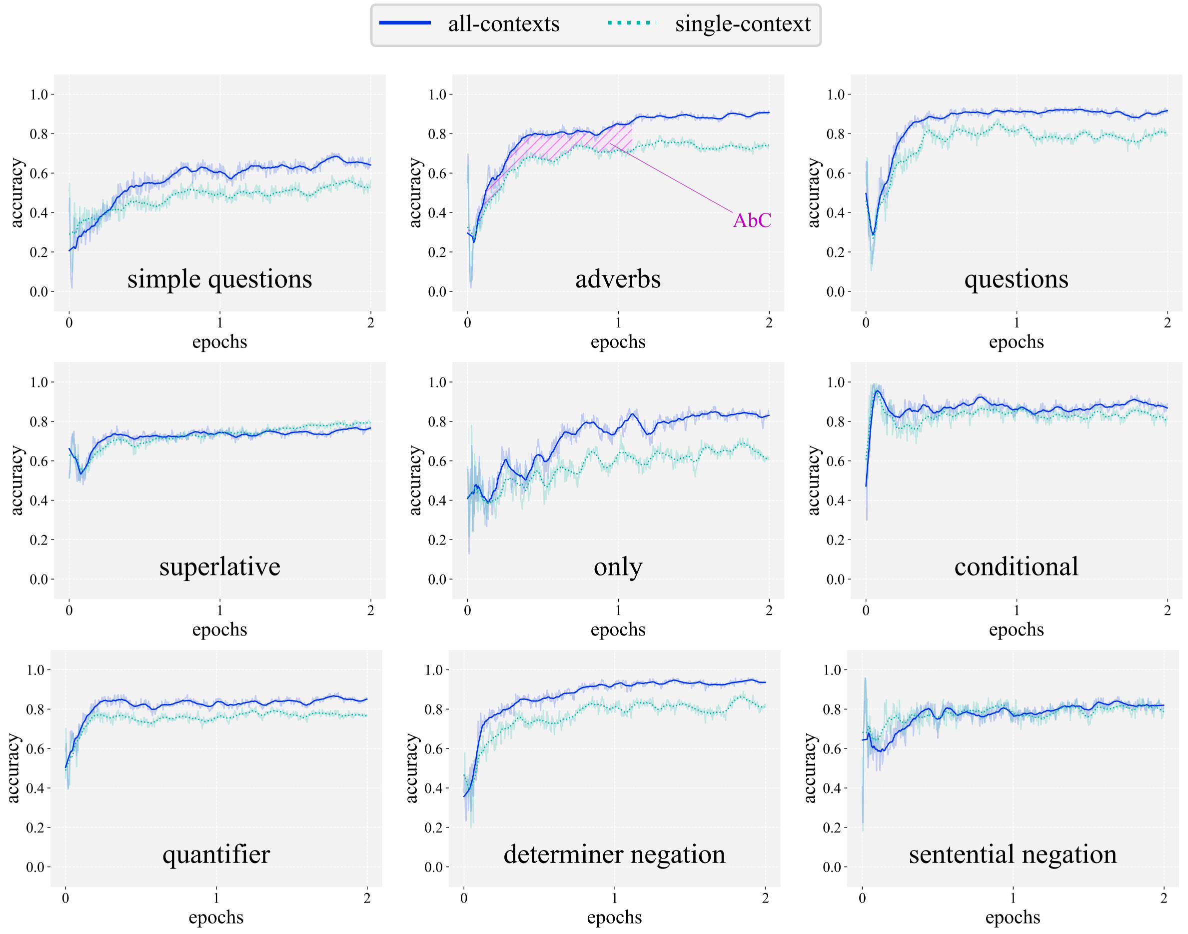


Figure 5: The LMs performance on different licensing contexts for the first two epochs of training. We obtained these curves by evaluating all models at all 730 training-checkpoints on the evaluation task.

all other NPI contexts, and the more the model leveraged from these contexts. The learning curves of all contexts, along with an illustration of the AbC-measure, can be found in Figure 5.

As a first interesting observation, we see that for seven of the nine contexts, the all-contexts model learns faster and achieves higher final performance.[7] Both frequent and infrequent contexts thus benefit from information acquired by other licensing contexts, in terms of both data-efficiency and final accuracy.

This positive transfer can also be seen in Figure 6, where we plot the AbC for all licensing contexts against their frequency. This plot also confirms the relationship found in our previous experiments: the less frequent a context is, the more it benefits from other NPIs ($r = .76$, $p < .05$).

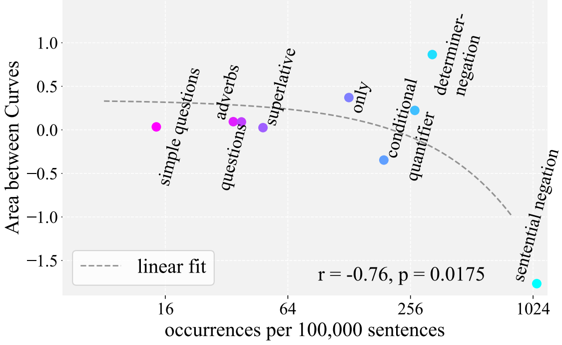


Figure 6: Normalised AbC for all licensing contexts until convergence of both contexts to 95% accuracy. AbC > 0 indicates a better performance of the all-context model and vice versa.

5 Discussion

In this paper, we studied language modelling as a multi-task problem. We show that neural language models can find and exploit similarity between the different language construction rules that we deduced from linguistic theory and that their transfer

---

[7] A one-sided Welch's test confirms that the calculated AbCs are overall different from zero: $t = 2.61$, $p < .05$.

behaviour mirrors the generalisation behaviour in traditionally constructed MTL settings. In this section, we now reflect on how our setup and results contribute to the three different areas that we mentioned in the introduction: MTL, linguistics and interpretability research.

5.1 Multi-task learning research

Studying LMs as multi-task learners, we observe several phenomena known from traditional MTL: when trained in parallel, similar (sub)tasks are learned more efficiently (compare Collobert et al., 2011; Kaiser et al., 2017), and with higher accuracy (Collobert and Weston, 2008; Kaiser et al., 2017), and this effect is stronger for less frequent tasks (Benton et al., 2017; Kaiser et al., 2017).

Our study differs in one crucial aspect from previous research on MTL: it looks at learning dynamics *within* one, larger, natural task instead of between tasks defined by the modeller. As a consequence, the learning process itself is not constrained through a priori decisions concerning task selection, or how tasks should be optimised together. In our scenario, contrary to traditional MTL, we use tasks and their hypothesised similarity only to *analyse* the learning process of the language model, not to inform its training. As such, our natural setting allows to study traditional MTL phenomena, such as data amplification, eavesdropping, and attention focusing, independent of arbitrary decisions regarding task selection and optimisation. This knowledge can then be transferred to scenarios in which more control over the selection of tasks may be required.

5.2 Interpretability research

A second field where we believe studying language models as multi-task learners can contribute, is the field of interpretability. On a more basic level, our paper confirms previous findings in interpretability that LMs are able to adequately model NPIs (Jumelet and Hupkes, 2018; Wilcox et al., 2019; Marvin and Linzen, 2018). We add to this literature by *explicitly* showing that LMs are connecting different types of contexts together through their learning behaviour. Contrary to previous work, we are tapping the learning process itself as a source of information to better understand the inner workings of these models.

Traditional concepts from MTL, such as the earlier mentioned explanations of Caruana (1993) and Ruder (2017) (§ 2.1) are valuable to better understanding what models are learning and how. For instance, when we observe that the solution of models improves when more varied NPI material is presented (our single- versus all-context experiment), MTL can aid to formulate concrete hypotheses about *why* this is the case. This, in turn, can help us improve our understanding of the solutions that are learned by the model. For instance, we find that the single-context models usually level-off on a lower accuracy-level than the all-context model (see Figure 5). This is not merely explainable by the amount of data, as we continue to add training examples in either case. The difference between models instead appears to be due to the variety of the training data. The idea of *attention focusing* (Caruana, 1993, 1997; Ruder, 2017) helps us to understand what is going on: by being trained on more varied NPI material, the model can better sort out which features are relevant and which ones are instead idiosyncrasies correlated with specific contexts. Such hypotheses can then help inform further experiments, that investigate – for example – which features specifically are better learned through attention focusing.

5.3 Linguistics research

Finally, we believe that studying language models as multi-task learners can also contribute to the field of linguistics. In our study, we show that LMs can find and exploit similarities between linguistically defined concepts. Turning things around, this generalisation behaviour of models can also be seen as a confirmation of the linguistic task hierarchy that we assumed from the start. The language modelling objective is unconstrained by linguistic theory and therefore does not necessarily have to find the same solutions as linguistics. Similarity derived from the learning behaviour of language models might therefore be used as a tool to work on more disputed ideas in linguistics and to form new hypotheses in linguistic theory. While the linguistic insights that can be drawn from the current study are relatively limited, they do provide a proof of concept for future work: we show that domain knowledge and learning behaviour of neural models can be connected.

6 Conclusion

In the current study we explored the possibility to use multi-task learning as a framework to study learning behaviour *within* a task. To this end we considered LMs as multi-task learners and investigated how they learn the task-cluster of NPI-licensing. We find that LMs pick up on similarities that we assume from linguistic theory and exploit them to learn similar language constructions with less data and to a higher accuracy. Especially less frequent tasks benefit from this effect.

These results resemble positive transfer in 'traditional' MTL. We lined out the possible benefits that our study may have for MTL research, interpretability and linguistics. From here there are many directions for future work: targeting less comprehensively researched areas in linguistics to add empirical data to otherwise usually theoretical linguistic discussions, investigating the change of internal representations in place of the behavioural measure used here to more precisely describe the learning process, or applying the approach to other high-level tasks in other modalities obeying other knowledge domains are just few of theses possibilities.

Acknowledgments

We thank the anonymous reviewers; the COLT-group at UPF for the discussions and their useful feedback; the participants of the EviL-seminars, and especially Emmanuel Dupoux, for inspiration; and Laura Castro Moreno for her help with the graphic designs. Further, LW thanks the Department of Translation and Language Sciences at the University Pompeu Fabra for funding.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853.

Andreas Argyriou, Massimiliano Pontil, Yiming Ying, and Charles A Micchelli. 2008. A spectral regularization framework for multi-task structure learning. In *Advances in neural information processing systems*, pages 25–32.

Chris Barker. 2018. Negative polarity as scope marking. *Linguistics and Philosophy*, pages 1–28.

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198.

Adrian Benton, Margaret Mitchell, and Dirk Hovy. 2017. Multitask learning for mental health conditions with limited social media data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 152–162, Valencia, Spain. Association for Computational Linguistics.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169, Valencia, Spain. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML '08: Proceedings of the 25th international conference on Machine Learning*, pages 160–167.

Ronan Collobert, Jason Weston, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Stephen M Cormier and Joseph D Hagman. 2014. *Transfer of learning: Contemporary research and applications*. Academic Press.

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

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655.

Gilles Fauconnier. 1975. Polarity and the scale principle. *Chicago Linguistics Society*, 11:188–199.

Anastasia Giannakidou. 2011. Negative and positive polarity items: Variation, licensing, and compositionality. *Semantics: An International Handbook of Natural Language Meaning*, pages 1660–1712.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New

Orleans, Louisiana. Association for Computational Linguistics.

Jack Hoeksema. 2012. On the Natural History of Negative Polarity Items Syntax View project Morphology View project. *Linguistic Analysis*, 44(2):3–3–3.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal. Association for Computational Linguistics.

Jaap Jumelet. 2020. diagNNose: A library for neural activation analysis. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 342–350, Online. Association for Computational Linguistics.

Jaap Jumelet and Dieuwke Hupkes. 2018. Do language models understand anything? on the ability of LSTMs to understand negative polarity items. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 222–231, Brussels, Belgium. Association for Computational Linguistics.

Lukasz Kaiser, Aidan N. Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. 2017. One model to learn them all. *arXiv preprint arXiv:1706.05137*.

William A. Ladusaw. 1980. *Polarity Sensitivity as Inherent Scope Relations*. Ph.D. thesis, University of Texas, Austin.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Aleksandr Romanovich Luriia. 1976. *Cognitive development: Its cultural and social foundations*. Harvard university press.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Andreas Maurer. 2006. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7(Jan):117–139.

Sinno J. Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Alexandre Passos, Piyush Rai, Jacques Wainer, and Hal Daume. 2012. Flexible modeling of latent task structures in multitask learning. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, volume 2, page 1283–1290.

David N Perkins, Gavriel Salomon, et al. 1992. Transfer of learning. *International encyclopedia of education*, 2:6452–6457.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509.

Michael T. Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G. Dietterich. 2005. To transfer or not to transfer. In *In NIPS'05 Workshop, Inductive Transfer: 10 Years Later*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Abraham Savitzky and Marcel J.E. Golay. 1964. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8):1627–1639.

Daniel L Schwartz, John D Bransford, David Sears, et al. 2005. Efficiency and innovation in transfer. *Transfer of learning from a modern multidisciplinary perspective*, 3:1–51.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9120–9132. PMLR.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Sebastian Thrun and Joseph O'Sullivan. 1996. Discovering structure in multiple learning tasks: The tc algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 489–497. Morgan Kaufmann.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. 2019. Characterizing and avoiding negative transfer. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11285–11294.

Alex Warstadt and Samuel R Bowman. 2020. Can neural networks acquire a structural bias from raw linguistic data? *arXiv preprint arXiv:2007.06761*.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Ethan Wilcox, Peng Qian, Richard Futrell, Miguel Ballesteros, and Roger Levy. 2019. Structural Supervision Improves Learning of Non-Local Grammatical Dependencies. In *Proceedings of North American Association for Computational Linguistics (NAACL)*, pages 3302–3312.

Yu Zhang and Qiang Yang. 2017. A survey on multitask learning. *CoRR*, abs/1707.08114.

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. 2014. Facial landmark detection by deep multi-task learning. In *Computer Vision – ECCV 2014*, pages 94–108, Cham. Springer International Publishing.

A List of NPIs

We here present the full list of 160 NPIs that has been used for modifying the corpora:

- a bed of roses
- a care in the world
- a chance in hell
- a damn
- a damn thing
- a day goes by
- a day over
- a ghost of a
- a hair out of place
- a living soul
- a moment of your time
- a moment too soon
- a shadow of a doubt
- a single soul
- all that much
- all that many
- any
- any longer
- any old
- any time soon
- anybody
- anymore
- anyone
- anything
- anything like
- anytime soon
- anywhere
- anywhere close
- anywhere near
- as of yet
- as yet
- at all
- avail
- bat an eye
- be any time
- be anything like
- beat around the bush
- by a long sho
- by any chance
- by any means
- by any stretch
- by miles
- by much
- can be bothered
- can compare to
- can hold a candle to
- can make of
- can possibly
- chance in hell
- come at a worse time
- come cheap
- could care less
- could possibly
- cut the mustard
- even once
- ever
- far wrong
- for much longer
- for shit
- for the life of
- for the soul of
- give a crap
- give a damn
- give a fuck
- give a shit
- half a chance
- half bad
- have a clue
- have any of
- hold a candle to
- hold water
- in a blue moon
- in a hundred years
- in a long time
- in a million years
- in ages
- in all of history
- in any
- in any manner
- in any way
- in centuries
- in days
- in decades
- in his right mind
- in hours
- in living memory
- in minutes
- in months
- in recent memory
- in the least
- in the least bit
- in the slightest
- in weeks
- in years
- just any
- just yet
- know the first thing
- know the first thing about
- know the half of it
- least of all
- let alone
- lift a finger
- make a sound
- make head or tail of
- make much difference
- mean a thing
- mean feat
- miss a beat
- much care
- much help
- much of a
- much of anything
- much to look at
- much to lose
- nor
- on speaking terms
- on your life
- one single thing
- or anything
- rhyme or reason
- say much
- see eye to eye
- set foot
- set foot in
- set foot on
- sit right with
- sit well
- sit well with
- small feat
- so much as
- square with
- squat
- stand a chance
- strong suit
- such thing
- sweat it
- take his eyes off
- take kindly to
- take lightly
- take no for an answer
- that many
- that much
- that often
- the ghost of
- the half of
- the half of it
- the least bit
- the like of which
- the likes of which
- the slightest
- the slightest bit
- think much of
- to be taken lightly
- whatever
- whatsoever
- with a barge pole
- worth a damn
- worth his salt
- worth its salt
- yet

ChainCQG: Flow-Aware Conversational Question Generation

Jing Gu[1,3] **Mostafa Mirshekari**[1] **Zhou Yu**[1,2] **Aaron Sisto**[1]
[1] Searchable.ai
[2] Columbia University
[3] University of California, Davis
{jing,mostafa,aaron}@searchable.ai
zy2461@columbia.edu

Abstract

Conversational systems enable numerous valuable applications, and question-answering is an important component underlying many of these. However, conversational question-answering remains challenging due to the lack of realistic, domain-specific training data. Inspired by this bottleneck, we focus on conversational question generation as a means to generate synthetic conversations for training and evaluation purposes. We present a number of novel strategies to improve conversational flow and accommodate varying question types and overall fluidity. Specifically, we design *ChainCQG* as a two-stage architecture that learns question-answer representations across multiple dialogue turns using a flow propagation training strategy. *ChainCQG* significantly outperforms both answer-aware and answer-unaware SOTA baselines (e.g., up to 48% BLEU-1 improvement). Additionally, our model is able to generate different types of questions, with improved fluidity and coreference alignment.

1 Introduction

Conversational systems are important in many real-world applications, including personal assistants, educational tutors (Winkler et al., 2020), customer service (El Asri et al., 2017; Budzianowski et al., 2018), and increasingly, entertainment. A key component of these systems is the ability to interpret a search query and retrieve information from different sources as naturally and efficiently as possible. In analogous human interactions, such a search generally occurs through conversation. In this context, a conversation consists of a sequence of dialogue turns during which the search objective becomes clearer over time. The applications mentioned above could benefit greatly from this type of multi-turn interaction, enabling conversational agents to accurately predict intent, request additional information, and better understand ambiguous followup questions and comments. In an applied setting, meaningful and natural conversations are important features of virtual entities as a means to establish trust and improve usability.

Here, we are motivated by the challenging task of conversational question answering (CQA). Current open-source datasets such as CoQA (Reddy et al., 2019) and QuAC (Choi et al., 2018) provide strong baselines for this task. However, these datasets have limited applicability in practical settings, because 1) they are created from domain-agnostic source material, and 2) they do not necessarily consider the full diversity of question types and vernacular that may be encountered in natural dialogue. Creating realistic, domain-specific datasets to train CQA models is notoriously costly and time-consuming. As such, we focus on the related task of question generation as a means to generate synthetic conversational questions and subsequently, create new datasets or augment existing ones. This will ultimately enable training CQA models in closed-loop, simulation environments, as well as allow machines to initiate dialogue and engage in information-seeking behavior.

While QA models have been studied previously (Zhu et al., 2018; Huang et al., 2019; Yeh and Chen, 2019; Chen et al., 2020; Ju et al., 2019; Ohsugi et al., 2019)), the QG task, which is the focus of this paper, has received less attention. QG models in the answer-unaware setting aim to predict a question given the source passage, while in the answer-aware setting, the target answer and rationale are included as inputs as well. Most QG-related literature has focused on single-turn question generation using question-answer datasets such as SQuAD (Rajpurkar et al., 2016), and other textual sources like Wikipedia articles (Du and Cardie, 2018).

Conversational Question Generation (CQG)

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2061–2070
April 19 - 23, 2021.

proves more challenging than single-turn QG as the questions are often highly ambiguous on their own, forcing the model to learn a deeper understanding of the context surrounding the passage text and dialogue history (Pan et al., 2019). Most CQG studies have generated questions using only the passage and dialogue history as inputs (i.e., answer-unaware) (Pan et al., 2019; Qi et al., 2020; Nakanishi et al., 2019; Wang et al., 2018). Answer-aware CQG models, on the other hand, generate questions based on the target answer, as well as dialogue history and passage. Although answer-aware CQG models seek to improve the generated conversation flow, current answer-aware QG models suffer from issues including inaccurate coreference alignment, dialogue inconsistencies, incorrect grammar and the inability to generate many different types of questions (e.g. yes/no, factoid, explanation).

In this paper, we introduce *ChainCQG*, a Conversational QG model that achieves improved performance by jointly learning the representations of questions and answers sequentially, across multiple dialogue turns. To this end, we outline a two-stage architecture, inspired by the approach discussed in Wu et al. (2019) and Gu et al. (2020), where two language models are used to simulate user and system in a response generation task. Our *ChainCQG* model is trained end-to-end, resulting in high-quality questions, while reducing computational cost by using shared parameters across both models. Using an answer-aware strategy grounds each turn of QG by jointly encoding the passage with the target answer rationale, increasing accuracy of the generated question types and further aligning coreferences between dialogue turns. We evaluate our approach using the inverted CoQA dataset (Reddy et al., 2019), which is a large-scale CQA dataset that we re-purposed for question generation. Our model outperforms existing SOTA CFNet (Gao et al., 2019) and ReDR (Pan et al., 2019) by a large margin on automatic evaluation metrics, and shows improved results on human evaluation metrics as well. More information about the baselines will be discussed in Section 2.

In summary, the main contributions of this paper are threefold [1]:

- The *ChainCQG* two-stage architecture is introduced with answer-aware input encoding, and it is an end-to-end model which is able to fluently generate different types of questions and achieve high consistency with the target answers.
- We demonstrate a flow propagation-based training method to learn question-answer representations across multiple dialogue spans.
- *ChainCQG* sets the new SOTA results on the answer-aware CQG task with robust human evaluation results.

The remainder of the paper is structured as follows: First, we discuss related work and how our approach is distinguished from previous methods (in Section 2). Then, we discuss the *ChainCQG* framework and preprocessing steps (in Section 3). Next, we describe our experiments, datasets and metrics, and evaluation results (in Section 4). Finally, we discuss conclusions, future work and the ethical issues (in Sections 5 and 6).

2 Related Work

In this section, we first explore previous approaches to question generation and then discuss outstanding research challenges in conversational question generation.

2.1 Single-turn Question Generation

Single-turn question generation has been the focus of extensive research. Two of the main categories in QG are answer-unaware and answer-aware. The former category generates the question without knowledge of the answer and solely based on the passage; whereas, the latter takes both passage and answer as inputs. Traditional approaches for answer-unaware QG include two main steps: content selection and question generation (Du and Cardie, 2017; Subramanian et al., 2018). Some of the more recent approaches utilize sequence-to-sequence (seq2seq) models for end-to-end question generation using Transformer-based architectures (Scialom et al., 2019). Various techniques have been used for improving the generated questions, including contextualized word embeddings (Scialom et al., 2019), question type usage and copying mechanism (Wu et al., 2020), and typed decoders (Wang et al., 2018).

To enable answer-aware question generation, the input passage is augmented with information describing the answer. For example, the passage can be concatenated with the answer positions and

---

[1] Code available at `https://github.com/searchableai/ChainCQG.`

lexical features (e.g., part-of-speech (POS) and named entity (NER)) to form the encoder input of a seq2seq model (Zhou et al., 2017). Jointly modelling the unstructured passage and the structured answer-relevant relation has been suggested for improving question generation as well (Li et al., 2019). Additional techniques have been proposed to solve various answer-aware QG challenges, including poor performance on long passages (Zhao et al., 2018) and the bias of repeating the terms in the target answer within the generated question (Kim et al., 2019).

2.2 Conversational Question Generation

Compared to single-turn QG, conversational (i.e., multi-turn) QG is less frequently explored in the literature. Further, it is more difficult as it requires a deeper understanding of the context and the dialogue history. Previous work mostly focused on answer-unaware CQG (Pan et al., 2019; Qi et al., 2020; Nakanishi et al., 2019; Wang et al., 2018). Specifically, Pan et al. (2019) proposed an encoder-decoder framework, ReDR, for answer-agnostic CQG, which is fine-tuned using feedback from an independent question-answer model. However, in this setting, maintaining conversational flow and consistency between dialogue turns is a primary challenge.

In this paper, we focus on answer-aware question generation. By grounding the generated question with the target answer rationale in each turn, this approach seeks to improve conversational flow and question-answer consistency. Within answer-aware CQG, (Gao et al., 2019) introduced the current SOTA, CFNet, which combined an auxiliary coreference alignment module with a copy mechanism and dialogue flow embedding. As will be described (in Section 4), we compare our model to answer-aware, CFNet (Gao et al., 2019), and answer-unaware, ReDR (Pan et al., 2019), SOTA baselines. As a practical note, in real-world applications, answer-aware QG systems may be augmented with an Answer Generation (AG) model to form an answer-unaware model that predicts the next conversational question-answer pair jointly. Discussing this AG model is out of the scope of this paper and will be the focus of future work.

3 The *ChainCQG* Framework

ChainCQG learns the question-answer representations jointly using two modules: Answer Encoding (AE) and Question Generation (QG). Encoding the answer based on the passage and dialogue history improves the answer understanding within the QG module, which in turn improves the generated questions. In the rest of this section, we provide a description of the input pre-processing steps, the general CQG problem formulation, and the AE and QG modules.

3.1 Task Definition

The conversational QG task in this paper aims to predict the next question given the passage (P), target answer (A_n) and history of the dialogue preceding the n$^{\text{th}}$ turn, (H_n). We also consider the answer rationale in each turn, and annotate the passage with the target answer rationale span, which we denote as P_{HL_n}. Mathematically, given the annotated passage (P_{HL_n}), target answer (A_n), and dialogue history ($H_n = ((Q_1, A_1), (Q_2, A_2), ..., (Q_{n-1}, A_{n-1}))$), the QG task predicts the next question Q_n. This task can be defined explicitly as generating a question, $\hat{Q}$, where:

$$\hat{Q} = \underset{Q_n}{argmax} Prob(Q_n | P_{HL_n}, A_n, H_n). \quad (1)$$

3.2 Input Preprocessing

In this Section, we briefly describe the processing steps necessary to prepare the input data. Specifically, we take the following approach:

1. We first create n sub-dialogues based on the full dialogue, with the i-th sub-dialogue starting from the first turn and finishing with the i-th turn, i.e., $SD_1 = \{\{Q_1, A_1\}\}$, $SD_2 = \{\{Q_1, A_1\}, \{Q_2, A_2\}\}$, ..., $SD_n = \{\{Q_1, A_1\}, \{Q_2, A_2\}, ..., \{Q_n, A_n\}\}$.

2. For the i-th sub-dialogue, we use a highlight token, [HL], to denote the answer rationale in the passage corresponding to the answer in the i-th turn, which serves as additional context for the target answer.

3. For each sub-dialogue, i, the passage (with the highlighted token corresponding to the i-th turn) is concatenated with the first answer (i.e., A_1) by a SEP token. We denote the concatenation as A^*.

4. We then reverse the order of the answers and questions in the sub-dialogues (e.g., {A1, Q1} instead of {Q1, A1}). The reason behind this

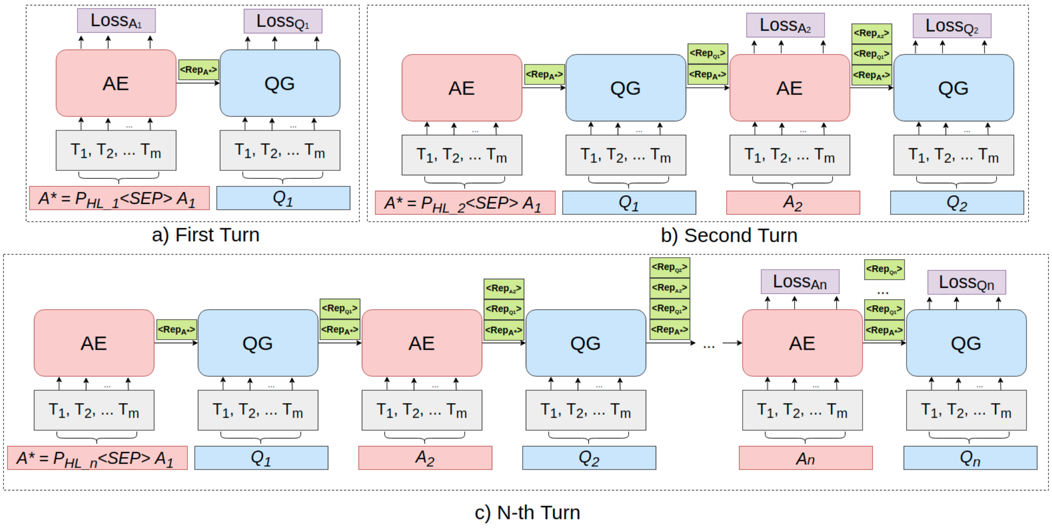


Figure 1: Main structure of *ChainCQG*. Each QA turn in the dialogue span is trained with a separate conversational flow that contains all previous dialogue turns. Answer Encoder and Question Generator modules iteratively generate and share answer and question representations across multiple dialogue turns.

step is to align the input sequence with the natural order of the QG task, where the questions come after the answers. We examine the effects of this ordering scheme in later ablation studies.

3.3 Answer Encoding and Question Generation Modules

In this paper, we use GPT-2 (Radford et al., 2019) to represent both the AE and QG modules described above. Specifically, the AE model is used to learn the representation of the passage and answer in each turn, and the QG model is used to learn the representation of the dialogue history and generate the next question in the conversation. The AE and QG models communicate via the models' hidden states, which are the K and V values when using GPT. K and V values together form a contexual representation for the entire conversation history.

3.4 Flow Propagation-Based Training

To improve the conversational flow of the CQG, we introduce a sequential training process. Figure 1 shows the main structure. To make it more concrete, let us consider a dialogue of n turns ($\{\{Q_1, A_1\}, \{Q_2, A_2\}, ..., \{Q_n, A_n\}\ \}$). The forward propagation process iterates through all previous turns and finally estimates the loss values for A_n and Q_n. In this process, we pass the GPT-based (K, V) representation forward to the next module, which accumulates the representations of each previous turn, with the original highlighted passage as reference. For each sub-dialogue, we only update the loss from the answer and question in the last turn since the highlighted span specifies the information for the last turn. For a sub-dialog of n turns, the loss is calculated as

$$Loss = Loss_{A_n} + Loss_{Q_n} \tag{2}$$

where

$$Loss_{A_n} = CE(A_n, P_{A_n}) \tag{3}$$

and

$$Loss_{Q_n} = CE(Q_n, P_{Q_n}) \tag{4}$$

CE refers to the cross-entropy loss from a target sentence. The parameters of the model are updated by backpropagating the aggregated loss values. By considering the encoding of the previous turns for estimating the loss and increasingly considering various sub-dialogues, the flow propagation-based training improves the conversational flow of the CQG.

4 Experiments

4.1 Dataset

We conduct experiments on the CoQA dataset (Reddy et al., 2019), which is a large-scale conversational question answering dataset composed of 8k conversations with 127k question-answer pairs collected via Amazon Mechanical Turk. Each dialogue turn also contains the supporting rationale for each answer. A number of different question types are present, as documented in the original paper, including Yes/No, explanation (i.e. How?, Why?), and factoid (i.e. What? When? Where? How much?). Yes/No and explanation questions, alongside the overall conversational language, make this an exceedingly challenging dataset for CQG. Since the private test set is not available, we conduct experiments on the training set and the dev set. We randomly sample 10% from the original training set to form the test set, and keep the original dev set unchanged. We conduct our experiments with a training set with 97783 examples, dev set with 7983 examples and test set with 10846 examples. We report the performance on the test set.

4.2 Implementation Details

We use both $\text{GPT}_{\text{small}}$ and $\text{GPT}_{\text{medium}}$ in all experiments. For baselines, we consider ReDR, the SOTA method in answer-unaware CQG, and CFNet, the SOTA method in answer-aware CQG. We also implemented two SOTA pre-training generation models, T5 and BART. They utilize all our preprocessing methods and training skills except the AE/QG modules. We used T5_{large} (770M) and $\text{BART}_{\text{large}}$ (400M), which are comparable with *ChainCQG*-M in terms of parameter size.

We initialize *ChainCQG* with the open sourced GPT-2 parameters (Radford et al., 2019). We apply AdamW optimizer (Loshchilov and Hutter, 2019), and the warmup ratio is set to 0.1. The learning rate is tuned between 2e-5 and 5e-5. The dropout ratio is set to be 0.1. We decode questions by nucleus sampling (Holtzman et al., 2020) with top-p as 0.2, top-k as 400, and temperature as 0.7.

4.3 Evaluation Metric

Our main objectives when evaluating our model are quality of the generated questions and performance on our task goal (e.g., asking conversational questions that are consistent with the target answers). To this end, we first examine a set of automated metrics. Then, to ensure robustness, we evaluate and discuss a set of human-based metrics.

4.3.1 Automated Metrics

To evaluate our question generation approach, we aim to show that it is 1) grammatically and semantically correct and 2) able to achieve the task objectives. To achieve the first goal, we compute automatic metrics with respect to the ground truth questions. We report multiple commonly used metrics, including BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005), and perplexity (Clarkson and Robinson, 1999). BertScore (Zhang* et al., 2020) and MoverScore (Zhao et al., 2019) are recently proposed metrics that utilize a large pre-trained model to evaluate the generation quality at the semantic level. We use both of these to evaluate the semantic similarity between the generated question and the reference question. Since two questions could express similar meaning with low lexical overlap, these two semantic-level metric could also show important information about the question quality.

4.3.2 Human Evaluation Metrics and Procedure

In this section, we discuss the metrics used for human evaluation. Human evaluation provides additional support for the approach and the robustness of automatic evaluation. Specifically, we use answerability and fluency to measure the quality of the generated questions in relation to the context. We have used Mechanical Turk for this evaluation.

In the context of answer-aware question generation, **Answer Consistency** describes whether the generated questions result in the correct answers (Celikyilmaz et al., 2020). To measure this metric, we provide the passage and the answer, and ask the evaluators whether the generated question is consistent with the answer (i.e., 1 for consistent and 0 for inconsistent). **Fluency** measures the quality of the generated questions and accounts for criteria such as grammar, spelling, choice of words, and style (Du et al., 2017). To measure this metric, we provide the generated question and ask the human evaluator whether the language in the generated question is fluent. We consider three categories of errors: grammar/spelling mistakes, missing entity names, and mismatched pronouns. Based on these categories, we assign 2 for cases with no mistake in any of the categories, 1 for cases with maximum of one mistake in any of the mentioned categories, and 0 for cases with one or more mistakes in each one of the categories. We scale the fluency score to (0,1) by maximum evaluation scores.

4.4 Main Results

The *ChainCQG* model architecture is evaluated alongside two SOTA baselines (ReDR, CFNet) on a number of automatic metrics, including BLEU (1-4), METEOR, ROUGE-L, MoverScore, and BERTScore. More information about these metrics and baselines was presented in Section 4.3.

Model	B1	B2	B3	B4	M	RL	BS	MS
ReDR	27.58	7.81	2.83	1.35	12.15	34.05	87.14	7.62
CFNet	38.24	22.60	16.11	12.23	25.75	43.25	91.25	25.92
BART-large	49.41	30.57	19.40	12.34	35.78	46.88	92.55	31.89
T5-large	50.83	32.64	20.81	13.84	37.08	48.67	92.86	33.91
ChainCQG-M	**53.15**	**35.31**	**23.31**	**15.78**	**40.15**	**50.98**	**93.14**	**36.40**
ChainCQG-S	49.26	31.06	20.24	12.11	33.26	46.23	92.53	32.82

Table 1: Automated Metric Evaluation Results.

These scores seek to evaluate the lexical overlap, and to some degree, the semantic similarity, between generated and ground truth questions within each dialogue turn. We also train two QG models based on pretrained Transformer seq2seq architectures (BART-large (Lewis et al., 2020), T5-large (Raffel et al., 2020)) using all elements of the *ChainCQG* methodology except the question-answer representation sharing mechanism used in *ChainCQG* model. Instead, the target answer is the direct input to the model, concatenated with the passage and dialogue history.

4.4.1 Automated Metrics Results

Results of all models and baselines are shown in Table 1. In the top row of this Table, P is the perplexity, B1-4 are BLEU 1 through BLEU 4, M is METEOR, RL is ROUGE-L, BS is the BERTScore, and MS is MoverScore. In the first column, ChainCQG-M and ChainCQG-S refer to two version of our approach using medium and small GPT-2. Major observations are listed below:

The top performing *ChainCQG* model, composed of two GPT-2 Medium modules, improves upon all baselines by a considerable margin and across all the considered metrics. In addition, it also outperforms the T5-large, which has more parameters, by a large margin. This suggests that the *ChainCQG* learns a better representation using the AE-QG structure, with less parameters than the T5. We improve upon the current answer-aware CQG SOTA, CFNet, on each metric as well. Note that with our methods, T5-large and BART-large also outperforms the SOTA methods. T5-large is the next best performing model, trailing *ChainCQG* by at least two points on all metrics except BERTScore, which shows a narrower margin of improvement.

4.4.2 Human Evaluation Results

It is well-known that automatic evaluation metrics do not always correlate with human judgement in conversational generation tasks (Celikyilmaz et al., 2020). Especially in the context of CQG, there is a many-to-one relationship between questions and their target answers and dialogue contexts, and token-based metrics are inherently unable to measure the similarity between such sequences with low degrees of lexical overlap. As a recourse, we also assess our model performance on a number of human evaluation metrics described in a previous section: Answer Consistency and Fluidity. These metrics cover an important cross-section of human judgement, which is not represented in the automatic metrics. Specifically, we seek to quantify the naturalness and consistency of *ChainCQG* results within each dialogue span. Table 2 shows the performance of our models and baselines on these metrics.

Model	Consistency	Fluency
CFNet	0.710	0.439
BART	0.792	0.482
T5	0.757	0.462
ChainCQG-M	**0.817**	**0.548**

Table 2: Human Evaluation Results.

4.5 Results Discussion

Overall, both standalone QG models using BART and T5, as well as *ChainCQG* outperformed the SOTA baseline, CFNet, on both metrics, while the *ChainCQG* model achieved the best performance of all models on both metrics. The Answer Consistency roughly indicates that the question types were better aligned with the target answer and dialogue history, than the baseline, while the Fluency metric points to improvements in factors like grammar, coreference alignment, and dialogue flow. Together with the Automatic Metrics, these results support our finding that the *ChainCQG* model is able to learn to produce conversational dialogue that is well aligned with the CoQA dataset, both lexically and semantically, and more robust in general.

The improvement over single CQG seq2seq models like T5 demonstrates the success of learning joint question-answer representations and using the encoding of the latter to inform the QG module. We present a more complete analysis of error and ablation studies in the following sections. The answer-aware QG strategy is also validated here, as shown by the significant improvement of every answer-aware model over the answer-unaware (ReDR) baseline. Finally, the comparison of *ChainCQG* to the current answer-aware QG SOTA, CFNet, demonstrates the importance of the question-answer representation and encoding scheme in our model. While CFNet incorporates tactical model components to improve the quality of CQG along specific dimensions, such as coreference and dialogue flow, our model is able to flexibly learn the progression of questions without the need for architectural components that target specific dialogue features. Another important point is that CFNet excluded all Yes/No questions from their analysis, as they proved difficult to reliably generate. Our model not only achieves SOTA performance, but is also able to natively generate every question type present in the dataset. Moreover, we notice that our model shows improved coreference alignment ability when generating questions with complex and entangled dialog history.

4.6 Ablation Study

Our ablation study aims to understand the effectiveness of various design choices in the *ChainCQG* approach outlined (in Section 3.3). In all ablation experiments, the reference model is the *ChainCQG* model, combining the AE and QG modules. Results of this analysis are shown in Table 3. We have applied the following ablations:

4.6.1 Removing the dialogue history

Here, we evaluate the effect of the flow propagation training scheme. Table 3 shows that removing the dialogue history, and consequently, any notion of dialogue flow, reduces performance across all the metrics (e.g., approximately 14% for both small and medium versions). These results match the intuition that dialogue history provides essential context to correctly handle coreferences and natural transitions.

4.6.2 Removing the answer rationale highlight tokens

To evaluate the effect of grounding the generated questions in the relevant passage text, we remove the answer rationale highlight tokens from the input passage. The results in Table 3 show that this ablation decreases performance in all the metrics. For example, removing the highlight reduces BLEU-1 for the medium GPT from 53.15 to 47.07 (approximately 11% reduction). We conclude that the highlighted tokens ground the model in the relevant passage information, providing essential context while focusing the scope of the question.

4.6.3 Changing the order of the questions and answers

As discussed in Section 3.2, we have used the AQ order instead of QA in our input encoding. Here, we evaluate the effect of such ordering. As Table 3 shows, reversing the order of the question and answers results in a performance reduction of approximately 5% in the BLEU-1 score for the medium GPT model. This shows that the AQ order is a more natural structure for dialogue flow propagation, since answers precede the generated question in each turn.

4.6.4 Removing the AE module

As discussed in Sections 1 and 3.3, by including the AE module, we aim to address the challenge of expressing the representations of questions and answers over multiple dialogue turns. Here, we remove the AE module to validate the effect of this modelling choice. The results in Table 3 show that removing the AE module reduces the performance of the model by 3% and 8% in the BLEU-1 score for the medium and small GPT models, respectively. This indicates that propagating the question-answer representations across dialogue turns produces rich temporal representations that improve the fidelity of dialogue flow.

4.7 Error Analysis

In order to better understand the performance differentiation between our model and baselines considered here, we inspected some samples of generated questions with poor quality. While the SOTA baseline, CFNet, neglected all Yes/No questions completely, our model is overall, very successful at generating this type, alongside others such as factoid and explanation. However, Yes/No questions

Model	P	B1	B2	B3	B4	M	RL	BS	MS
ChainCQG-M	7.04	53.15	35.31	23.31	15.78	40.15	50.98	93.14	36.40
ChainCQG-S	9.55	49.26	31.06	20.24	12.11	33.26	46.23	92.53	32.82
M w/o history	9.13	45.53	28.35	18.31	11.27	30.35	40.45	92.54	27.59
S w/o history	11.1	42.54	25.63	14.91	8.01	27.73	39.23	91.03	24.30
M w/o highlight	7.83	47.07	30.63	20.54	12.91	32.56	43.63	92.36	31.50
S w/o highlight	11.09	43.63	25.74	17.14	10.54	27.73	40.43	91.23	24.82
M w/o AQ order	7.43	50.23	33.65	21.43	14.08	37.35	48.88	92.94	34.73
S w/o AQ order	9.90	47.65	30.51	19.04	10.80	31.58	42.82	92.21	29.74
M w/o AE module	8.05	51.64	33.26	21.26	13.86	37.23	47.23	92.64	32.23
S w/o AE module	15.21	45.23	28.19	18.01	10.73	29.43	44.12	92.13	27.28

Table 3: Ablation Study Results.

can still be problematic when the answer context includes many potential targets, each of which could be satisfied by a consistent Yes/No question. We also find that in minority cases, *ChainCQG* cannot handle questions requiring complex logic or reasoning to arrive at the target answer. We hypothesize that a more powerful pre-training model could alleviate this issue. Also, the *ChainCQG* model sometimes includes additional details related to the answer, not contained in the gold question, which results in slightly more verbose, though consistent, questions.

5 Discussion and Ethical Issues

The results presented here demonstrate the efficacy of modern Transformer-based architectures, and specifically *ChainCQG*, in producing conversational questions on a challenging dataset. While answer-aware QG is our focus here, we plan to expand this in future work to include answer-unaware and open-ended QG, multi-task NLG involving QA, and domain-specific dialogue simulation. The flexibility of the *ChainCQG* architecture lends itself well to each of these problems, as representations from different inputs and tasks can be shared easily between modules.

As for the practical implications of our QG work, a number of applications mentioned in previous sections could immediately take advantage of QG features, either for training QA models or generating user-facing questions. In the former setting, generation models, such as *ChainCQG*, risk polluting the training dataset with examples that are noisy or inconsistent with the target answers, which can cause undesirable effects at inference time. In the latter setting, generation models may suffer from bias based on the questions available for training, which may lead to misrepresentation of application domains and individual users. Additional work is required to understand the extent of these issues in real-world applications, and identify corrective measures to ensure model robustness and diversified training distributions.

6 Conclusion

In this paper, we introduce *ChainCQG*, an answer-aware Conversational Question Generation model that outperforms all baselines on both automatic and human evaluation metrics on the inverted CoQA dataset (e.g., BLEU-1 improvement of 48% and 28% with GPT medium compared to ReDR and CFNet, respectively). We have designed a two-stage GPT-2-based architecture that jointly learns passage and dialogue history representations via a flow propagation training method. *ChainCQG* produces high-quality questions in multi-turn dialogue, addressing previous SOTA issues such as question type fidelity, question-answer inconsistency and coreference misalignment. Finally, we have performed and presented extensive ablation studies for various aspects of our approach.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of* the 2018 Conference on Empirical Methods in Natural Language Processing, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Graphflow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 1230–1236. International Joint Conferences on Artificial Intelligence Organization. Main track.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Philip Clarkson and Tony Robinson. 1999. Towards improved language model evaluation measures. In *Sixth European Conference on Speech Communication and Technology*.

Xinya Du and Claire Cardie. 2017. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2067–2073.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Association for Computational Linguistics (ACL)*.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219, Saarbrücken, Germany. Association for Computational Linguistics.

Yifan Gao, Piji Li, Irwin King, and Michael R. Lyu. 2019. Interconnected question generation with coreference alignment and conversation flow modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4853–4862, Florence, Italy. Association for Computational Linguistics.

Jing Gu, Qingyang Wu, Chongruo Wu, Weiyan Shi, and Zhou Yu. 2020. A tailored pre-training model for task-oriented dialog generation.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Hsin-Yuan Huang, Eunsol Choi, and Wen tau Yih. 2019. FlowQA: Grasping flow in history for conversational machine comprehension. In *International Conference on Learning Representations*.

Ying Ju, Fubang Zhao, Shijie Chen, Bowen Zheng, Xuefeng Yang, and Yunfeng Liu. 2019. Technical report on conversational question answering. *arXiv preprint arXiv:1909.10772*.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6602–6609.

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

Jingjing Li, Yifan Gao, Lidong Bing, Irwin King, and Michael R. Lyu. 2019. Improving question generation with to the point context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3216–3226, Hong Kong, China. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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

Mao Nakanishi, Tetsunori Kobayashi, and Yoshihiko Hayashi. 2019. Towards answer-unaware conversational question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 63–71.

Yasuhito Ohsugi, Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. 2019. A simple but effective method to incorporate multi-turn context with BERT for conversational machine comprehension. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 11–17, Florence, Italy. Association for Computational Linguistics.

Boyuan Pan, Hao Li, Ziyu Yao, Deng Cai, and Huan Sun. 2019. Reinforced dynamic reasoning for conversational question generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2114–2124.

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

Peng Qi, Yuhao Zhang, and Christopher D. Manning. 2020. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

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

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano. 2019. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6027–6032.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88, Melbourne, Australia. Association for Computational Linguistics.

Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018. Learning to ask questions in open-domain conversational systems with typed decoders. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2193–2203, Melbourne, Australia. Association for Computational Linguistics.

Rainer Winkler, Sebastian Hobert, Antti Salovaara, Matthias Söllner, and Jan Marco Leimeister. 2020. Sara, the lecturer: Improving learning in online education with a scaffolding-based conversational agent. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Qingyang Wu, Yichi Zhang, Yu Li, and Zhou Yu. 2019. Alternating roles dialog model with large-scale pretrained language models. *arXiv*, pages arXiv–1910.

Xiuyu Wu, Nan Jiang, and Yunfang Wu. 2020. A question type driven and copy loss enhanced framework for answer-agnostic neural question generation. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 69–78, Online. Association for Computational Linguistics.

Yi-Ting Yeh and Yun-Nung Chen. 2019. FlowDelta: Modeling flow information gain in reasoning for conversational machine comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 86–90, Hong Kong, China. Association for Computational Linguistics.

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

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2018. Sdnet: Contextualized attention-based deep network for conversational question answering. *arXiv preprint arXiv:1812.03593*.

The Interplay of Task Success and Dialogue Quality: An in-depth Evaluation in Task-Oriented Visual Dialogues

Alberto Testoni
DISI - University of Trento
Trento - Italy
alberto.testoni@unitn.it

Raffaella Bernardi
CIMeC and DISI - University of Trento
Rovereto (TN) - Italy
raffaella.bernardi@unitn.it

Abstract

When training a model on referential dialogue guessing games, the best model is usually chosen based on its task success. We show that in the popular end-to-end approach, this choice prevents the model from learning to generate linguistically richer dialogues, since the acquisition of language proficiency takes longer than learning the guessing task. By comparing models playing different games (GuessWhat, GuessWhich, and Mutual Friends), we show that this discrepancy is model- and task-agnostic. We investigate whether and when better language quality could lead to higher task success. We show that in GuessWhat, models could increase their accuracy if they learn to ground, encode, and decode also words that do not occur frequently in the training set.

1 Introduction

A good dialogue model should generate utterances that are indistinguishable from human dialogues (Liu et al., 2016; Li et al., 2017). This holds for both chit-chat, open-domain, and task-oriented dialogues. While chit-chat dialogue systems are usually evaluated by analysing the quality of their dialogues (Lowe et al., 2017; See et al., 2019), task-oriented dialogue models are evaluated on their task success and it is common practice to choose the best model based only on the task success metric. We explore whether this choice prevents the system from learning better linguistic skills.

Important progress has been made on the development of such conversational agents. The boost is mostly due to the introduction of the encoder-decoder framework (Sutskever et al., 2014) which allows learning directly from raw data to both understand and generate utterances. The framework has been found to be promising both for chit-chat (Vinyals and Le, 2015) and task-oriented dialogues (Lewis et al., 2017), and it has been further extended to develop agents that can communicate through natural language about visual content (Mostafazadeh et al., 2017; Das et al., 2017a; de Vries et al., 2017). Several dialogue tasks have been proposed as *referential guessing games* in which an agent (the Q-bot) asks questions to another agent (the A-bot) and has to guess the referent (e.g., a specific object depicted in the image) they have been speaking about (de Vries et al., 2017; Das et al., 2017b; He et al., 2017; Haber et al., 2019; Ilinykh et al., 2019; Udagawa and Aizawa, 2019). We are interested in understanding the interplay between the learning processes behind these two sub-tasks: generating questions and guessing the referent.

Shekhar et al. (2019) have compared models on GuessWhat and have shown that task success (TS) does not correlate with the quality of machine-generated dialogues. First of all, we check whether this result is task-agnostic by carrying out a comparative analysis of models playing different referential games. We choose a task in which visual grounding happens during question generation (GuessWhat, de Vries et al. 2017); a task in which it happens only in the guessing phase (GuessWhich, Das et al. 2017b), and a task that is only based on language (MutualFriends, He et al. 2017). We introduce a linguistic metric, Linguistic Divergence (LD), that, by assembling various metrics used in the literature (Shekhar et al., 2019; Murahari et al., 2019; van Miltenburg et al., 2019), measures how much the language generated by computational models differs, on the surface level, from the one used by humans. We consider LD to be a proxy of the quality of machine-generated dialogues.

For each task, we compare State-Of-The-Art (SOTA) models against their TS and LD. In the

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2071–2082
April 19 - 23, 2021.

core part of the paper, we study the relationship between the learning process behind TS and LD by comparing model performance across epochs and by downsizing the training set. Finally, we study whether and when a lower LD (i.e., the generated dialogues are more similar to humans) could help reach a higher TS.

Our results confirm that models performing similarly on TS differ quite a lot on their conversational skills, as claimed in Shekhar et al. (2019) for models evaluated on the GuessWhat game. Furthermore, we show that:

- SOTA models are much faster in achieving high performance on the guessing task compared to reaching a high dialogue quality (i.e., low LD). Hence, choosing the best model on task success prevents the model from reaching better conversational skills;
- SOTA models mostly use very frequent words; this limited vocabulary is sufficient for succeeding in a high number of games;
- in GusseWhat, a higher TS could be reached if the model learns to use also less frequent words.

2 Related Work

Task-oriented models can be evaluated based on their task success, but this is not enough to know whether the generated dialogues are human-like. The development of quantitative metrics to evaluate the quality of dialogues generated by conversational agents is a difficult challenge (Liu et al., 2016), and it is under investigation for chit-chat dialogue systems. For instance, Guo et al. (2017) study topic diversity in the conversational flow, which is rather important in chit-chat and open-domain dialogues, but less so for task-oriented ones; Kannan and Vinyals (2016), Li et al. (2017), Bruni and Fernández (2017) propose to use adversarial evaluation, whereas Lowe et al. (2017), See et al. (2019), and Hashimoto et al. (2019) propose automatic systems that build upon human evaluation. All these efforts are still preliminary and are not easily employable for new datasets or new models. Since no standard and unique metric has been proposed to evaluate the quality of task-oriented (grounded) conversational dialogues, we consider a mixture of metrics used independently in various studies, and we provide a comparative analysis across models and tasks based on the same set of linguistic metrics.

Neural Networks have been shown to generate text that sounds unnatural due to the presence of repeated utterances, poor vocabulary, and inconsistency in word usage (Ray et al., 2019). Various improvements have been proposed to mitigate these weaknesses. To prevent the decoder from choosing words based simply on their frequency, Li et al. (2019) replace its maximum likelihood estimation objective, while others change the sampling search strategy (Holtzman et al., 2020; Wu et al., 2019; See et al., 2019); these changes aim to reduce the number of repeated questions, to increase the variety of words and their distribution. Attempts have been made to provide the conversational models with a reasoning module based on Bayesian inference (Abbasnejad et al., 2019) or Rational Speech Act (Shuklar et al., 2019) frameworks that should lead to more informative and coherent questions. Here, we do not propose new models, but rather aim to better understand the strengths and weaknesses of current models.

3 Games and Metrics

Our focus is on task-oriented dialogues. We consider a task that relies on grounding language into vision during question generation, i.e. GuessWhat (de Vries et al., 2017), a task that requires grounding only at the guessing phase, i.e. GuessWhich (Das et al., 2017b), and a task based only on language, i.e. MutualFriends, (He et al., 2017).

Games As illustrated by the snippets reported in Table 1, the three tasks also differ in the flexibility of the dialogues: GuessWhat and GuessWhich are both based on rigid turns in which an agent asks questions and the other answers, whereas MutualFriends has free-form dialogues. Moreover, GuessWhat consists only of Yes/No questions, while in GuessWhich this constraint does not apply. Relevant statistics of the three datasets are summarized in Table 1.

GuessWhat (de Vries et al., 2017) is an asymmetric game.[1] A Questioner (Q-Bot) has to ask Yes/No questions to guess which is the target object among a set of maximum 20 candidates; while asking questions, it sees the image containing the

---

[1] The dataset of human dialogues is available at `https://guesswhat.ai/download`.

	#dialogues training	 testing	Vocab. size	#candidates	#turns	Examples
GuessWhat	108K	23K	4900	3-20	1-10	***A***: Is it a person? ***B***: No. ***A***: Is it a dog? ***B***: Yes. [***A*** guesses the target object]
GuessWhich	120K	2K	11321	2K	10	***A***: What color is the car? ***B***: Blue. ***A***: Who is driving it? ***B***: A man. [***A*** guesses the target image]
MutualFriends	8K	1K	5325	5-12	2-46	***A***: My friends work at Google. ***B***: None of mine do. [***A*** and ***B*** select a friend]

Table 1: Salient statistics of the human dialogues in the three data sets under consideration in this work. The last column reports samples of dialogues exchanged between two agents (A and B).

candidate objects and it has access to the dialogue history. The Answerer (A-Bot), who knows which is the target, provides the answers. The two bots learn to speak about the image by being trained on human dialogues, which have been collected by letting humans play the game. Humans could stop asking questions at any time (human dialogues contain on average 5.2 question-answer pairs), while models have to ask a fixed number of questions (8 in the setting we have considered).

GuessWhich (Das et al., 2017b) is also an asymmetric game. Unlike the task described above, the Q-Bot has to ask questions without seeing the candidate images, but it has access to captions describing the images. Q-Bot can ask any type of question; the target image has to be selected among 2K candidates at the end of the dialogue. The A-Bot instead sees both the caption and the target image. Human dialogues are from the VisDial dataset[2] and were collected as chit-chat dialogues (Das et al., 2017a). Both humans and models have to ask exactly 10 questions.

MutualFriends (He et al., 2017) is a symmetric game based only on text: two agents, each given a private list of friends described by a set of attributes/labels, try to identify their mutual friend based on the friend's attributes.

Metrics: Since we are interested in the interplay between the downstream task and the quality of the generated dialogues, we consider two types of metrics.

Task Success: We use the task success (TS) metrics used in the literature to evaluate models against these tasks, namely accuracy (ACC) for GuessWhat and MutualFriends, and Mean Percentile Rank (MPR) for GuessWhich. The latter is computed from the mean rank position (MR) of the target image among all the candidates. An MPR of e.g., 96% means that, on average, the target image is closer to the one chosen by the model than the 96% of the candidate images. Hence, in the VisDial test set with 2K candidates, 96% MPR corresponds to an MR of 80, and a difference of $\pm$ 1% MPR corresponds to $\mp$ 20 mean rank. The task success chance levels are: 5% accuracy (GuessWhat), 50% MPR (GuessWhich) and 11.76% accuracy (MutualFriends).

Linguistic metrics: It has been shown that the quality of the dialogues generated by computational agents is not satisfactory. The main weaknesses of these models consist of poor lexical diversity, a high number of repetitions, and the use of a limited vocabulary. To evaluate the quality of the generated dialogues (defined as the closeness to human dialogues according to surface-level cues), we use several metrics that have been proposed in the literature. As in He et al. (2017), we compute *unigram entropy (H)*, which measures the entropy of unique unigrams in the generated dialogues normalized by the total number of tokens used by the model. From Murahari et al. (2019), we take the *Mutual Overlap (MO)* metric, which evaluates the question diversity within a dialogue by computing the average of the BLEU-4 score obtained by comparing each question with the other questions within the same dialogue.[3]

---

[2] VisDial is available from `https://visualdialog.org/data`.

[3] A high number of novel questions and low mutual overlap cannot be taken per se as a sign of high quality of the dialogues: a model could ask a question never seen in training or with very little overlap with the other questions but completely out of scope. To rule out this possibility, we compute the cosine similarity of each question marked as novel and with a low mutual overlap with the dialogue they occur in, and compare it with the similarity between the latter and

Moreover, following Shekhar et al. (2019), we report the percentage of games with *one question repeated verbatim (GRQ)* within a dialogue. Finally, we compare models with respect to their ability on lexical acquisition by calculating the *Global Recall (GR)* introduced by van Miltenburg et al. (2019) to evaluate image captioning: it is defined as the overall percentage of learnable words (from the training set) that the models recall (use) during generation. Furthermore, taking inspiration from the Local Recall introduced in the same work, we propose a similar metric tailored to dialogues, i.e., *Local Recall-d (LRd)*, which measures how many content words the generated dialogue shares with the corresponding human dialogue for the same game. Given a human dialogue D_h about an image and a generated dialogue D_g about the same image, we compute LRd as the normalized lexical overlap (considering only content words) between D_h and D_g.

We sum up all these linguistic metrics used in the literature so far into one which we take as a proxy of the quality of dialogues: it shows the *linguistic divergence (LD)* of the dialogues generated by a model from human dialogues. To this end, we normalize each metric so that all values lie between 0 and 1: 0 stands for human performance for the "lower is better" metrics and 1 stands for human performance for "higher is better" metrics. We compute LD by averaging all the scaled values V for each model; we take $1 - V$ for "higher is better" metrics to obtain a "divergence" value. All the metrics are equally weighted. By definition, LD is 0 for human dialogues. LD captures three main surface-level aspects: overall vocabulary usage (H, GR), diversity of questions/phrases within a dialogue (MO, GRQ), and similarity of content word usage with respect to human dialogues (LRd). There could be some correlation between metrics capturing similar aspects of language quality, but this does not affect the validity of the proposed LD metric.

4 Models

For both visual dialogue games, GuessWhat and GuessWhich, supervised learning has been compared with other learning paradigms. After the introduction of the supervised baseline model (de Vries et al., 2017), several models have been proposed for GuessWhat. They exploit either reinforcement learning (Sang-Woo et al., 2019; Zhang et al., 2018b,a; Zhao and Tresp, 2018; Gan et al., 2019; Pang and Wang, 2020) or cooperative learning (Shekhar et al., 2019; Pang and Wang, 2020); in both cases, the model is first trained with the supervised learning regime and then the new paradigm is applied. This two-step process has been shown to reach higher task success than the supervised approach. For GuessWhich, after the supervised model introduced in Das et al. (2017a), new models based on reinforcement learning have been proposed, too (Das et al., 2017b; Murahari et al., 2019; Zhou et al., 2019), but their task success is comparable if not lower than the one achieved by using only supervised learning (see Testoni et al. 2019). Below, we briefly describe the models we have compared in our analysis. For each task, we have chosen generative models trained with different learning paradigms and for which the code is available; for each paradigm, we have tried to choose the best performing ones or those that obtain a task success near to state-of-the art and could help better understand the interplay between task success and dialogue quality.

GuessWhat We use the A-Bot introduced in de Vries et al. (2017), which is trained in a supervised learning (SL) fashion. For the Q-Bot, we compare models based on different learning paradigms: supervised and cooperative learning (GDSE-SL and GDSE-CL, respectively) proposed in Shekhar et al. (2019) and reinforcement learning (RL) proposed in Strub et al. (2017). In RL, the reinforce paradigm used aims at optimizing the task accuracy of the game. Besides using different learning paradigms, these models differ in their architecture. In particular, while in RL the Question Generator (QGen) and the Guesser are trained independently, in GDSE a common visually-grounded dialogue state encoder is used and the two modules are trained jointly. In both cases, the Guesser receives as input the candidate object's categories and their spatial coordinates, and during training it is updated only at the end of the dialogue.[4]

---

random questions taken from other dialogues. Embeddings are obtained by using Universal Sentence Encoder-USE (Cer et al., 2018). We found that novel and low-MO questions are more similar to their dialogue than the random ones, confirming the effectiveness of these metrics.

[4] The code of the A-Bot and of RL is available at https://github.com/GuessWhatGame/guesswhat. The code of GDSE at: https://github.com/shekharRavi/

GuessWhich We use the A-Bot introduced in Das et al. (2017b). For the Q-bot, we compare Diverse (Murahari et al., 2019) and ReCap (Testoni et al., 2019). Diverse and ReCap have similar architectures: several encoders incrementally process the linguistic inputs to produce the dialogue hidden state. This state is used to condition a decoder that generates a new question at each turn, and a guesser that is trained to produce the visual representation of the target image through a feature regression module. The two models differ in the encoders used and in the training paradigm. While Diverse encodes the caption together with the dialogue history through a Hierarchical LSTM, ReCap has two independent LSTMs that produce the linguistic features of the caption and of the dialogue history, merged together to produce the dialogue hidden state. Secondly, in Diverse, an auxiliary objective on the dialogue state embedding (Huber loss) is used to incentivize the bot to ask more diverse questions with respect to the immediate previous turn. In ReCap, the Guesser sees the ground-truth image only at the end of the game while in Diverse the Guesser is updated at each turn. ReCap has been trained only by SL, Diverse both by SL (D-SL) and SL plus RL (D-RL). Further details can be found in the respective papers (Murahari et al., 2019; Testoni et al., 2019).[5]

MutualFriends We evaluate the model proposed in He et al. (2017), DynoNet (Dynamic Knowledge Graph Network), in which entities are structured as a knowledge graph and the utterance generation is driven by an attention mechanism over the node embeddings of such graph. The model is trained via supervised learning and at test time it plays with itself. DynoNet consists of three components: a dynamic knowledge graph (which represents the agent's private KB and shared dialogue history as a graph), and two LSTMs that map the graph embedding over the nodes and generate utterances or guess the entity.[6]

5 Experiments and Results

Shekhar et al. (2019) has shown that in GuessWhat task success (TS) does not correlate with the quality of the dialogues. First of all, we check to what extent this result is task and model agnostic by taking GuessWhat, GuessWhich and MutualFriends as case-studies and compare the behaviour of the models described above.

First, we evaluate the impact of the number of epochs and the size of the training set; then, we study whether and when a lower LD could help to reach a higher TS.

5.1 Task Success and Linguistic Divergence

We evaluate all models described above, in their supervised, cooperative, or reinforcement learning version, aiming to test whether some patterns can be found irrespectively of the model and data explored. Our results confirm what has been shown in Shekhar et al. (2019) for GuessWhat: **TS does not correlate with the quality of the generated dialogues**; models with similar TS generate dialogues that vary greatly with respect to the linguistic metrics. We run a Spearman's analysis and found a very weak correlation between LD and TS (coefficient < 0.15, p-value < 0.05). Our comparison across tasks shows that both in GuessWhich and in GuessWhat the vocabulary used by humans while playing the games in the training and testing set is rather similar (resp., 91% and 84% of words are in common between training and testing sets). Yet in both visual tasks models reach an LRd of around 42%. Specifically, the average mean rank of words they fail to use is 7000 (over 11321) for GuessWhich and 3016 (over 4900) for GuessWhat. **Hence, models mostly use very frequent words.** Details on the metrics for each task and model are reported in Table 2.

5.2 Learning Processes behind TS and LD

We aim to understand the relation between TS and LD. To this end, we compare the two metrics during the training processes across epochs and by downsizing the training set. For each task, we consider the models trained in a SL fashion since those trained with other paradigms build on them. Hence, we focus on ReCap, GDSE-SL, and DynoNet.

Comparison Across Epochs We study for *how long* a model has to be trained to reach its best performance on guessing the target referent and

---

`Beyond-Task-Success-NAACL2019.`

[5] The code for the A-Bot model and for D-SL and D-RL is available at `https://github.com/vmurahari3/visdial-diversity`; it is not specified how the best models are chosen. For ReCap: we have obtained the code by the authors and trained the model; we have chosen the model whose MPR does not increase for the subsequent 5 epochs.

[6] The code is available at `https://github.com/stanfordnlp/cocoa/tree/mutualfriends`.

	GuessWhich				**GuessWhat**				**MutualFriends**	
	D-SL	D-RL	ReCap-SL	Hum	GDSE-SL	GDSE-CL	RL	Hum	DynoNet-SL	H
TS ↑	95.2	94.89	96.76	-	48.21	59.14	56.3	84.62	0.98	0.82
GR ↑	6.46	9.04	14.4	27.69	34.73	36.35	12.67	72.98	51.15	65.2
LRd ↑	39.93	41.83	42.76	-	42.1	42.41	34.51	-	-	-
MO ↓	0.51	0.41	0.23	0.07	0.39	0.23	0.46	0.03	-	-
GRQ ↓	93.01	81.17	55.37	0.78	64.96	36.79	96.54	0.8	-	-
H ↑	4.03	3.92	4.19	4.55	3.52	3.66	2.42	4.21	3.91	4.57
LD ↓	0.58	0.52	0.38	-	0.46	0.36	0.67	-	0.18	-

Table 2: Comparative analysis of different models on several tasks and datasets.TS: task success. GR: global recall. LRd: local recall. MO: mutual overlap. GRQ: games with repeated questions. H: unigram entropy. LD: linguistic divergence. ↑: higher is better. ↓: lower is better.

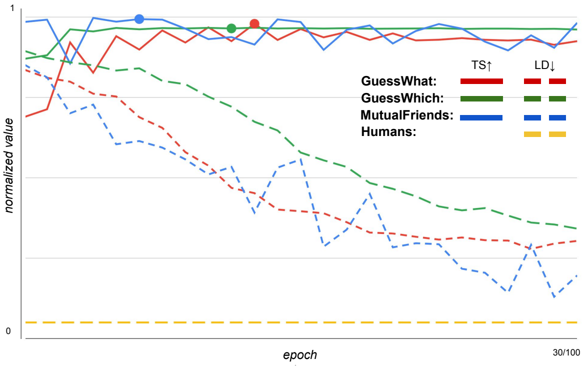


Figure 1: Comparison of Task Success (TS, solid lines) and Linguistic Divergence (LD, dashed lines) for GDSE-SL (GuessWhat, trained for 100 epochs), ReCap (GuessWhich, trained for 100 epochs), DynoNet (MutualFriends, trained for 30 epochs); Humans LD lower-bound metric in yellow. The LD of the generated dialogues keeps decreasing (moving close to human level) even though we no longer notice improvements in TS, whose highest value is reached well before (marked by bullets).

generating human-like dialogues. Figure 1 reports the TS and the LD of three models trained on the three tasks under examination. Each line is normalized w.r.t the highest value for each metric, so that it is possible to see different trends on the same plot. As we can see from the figure, the highest TS (marked by bullets) is reached earlier in GuessWhich and in MutualFriends than in GuessWhat. More interestingly, for all the tasks, the LD of the generated dialogues keeps decreasing (moving close to human level) even though we no longer notice improvements in TS, whose highest value is reached well before. Figure 2 (solid lines) reports the details of the linguistic metrics used to compute LD. We see that for all tasks a high entropy is reached already after a few epochs; this means that though the number of words used is small, models learn to distribute their use well. All the other metrics improve through the epochs quite a lot. For MutualFriends, we do not compute MO and GRQ since the model trained on it, DynoNet, asks questions referring to different attributes and hence, by design, it generates very few repetitions. From the results of this first experiment, it emerges fairly clearly that in all referential games we have considered, **models learn to perform well on the task quite quickly.** On the one hand, this means that **choosing the best model purely on the basis of its TS prevents the model from developing better linguistic skills**, on the other hand, that **the higher quality of the dialogues does not help reach a higher TS.** This result holds in all cases despite the target being an entity in a graph described by linguistic attributes (MutualFriends), an object (GuessWhat) or an image (GuessWhich).

Comparison by Downsizing the Training Set To understand whether the relation between TS and LD is related to the size of the training set, we compare models trained on datasets of decreasing size. We evaluate the models by training them with 50% and 25% of the standard GuessWhat and GuessWhich datasets. For MutualFriends, we have not run the downsizing analysis since the dataset is too small. For readability reasons, in Figure 2 we report only the results obtained with the 25% setting since they represent the observed pattern well enough. The y-axis reports the metrics scaled between 0 and 1. In GuessWhich the TS (yellow lines) does not decrease by downsizing the dataset: when using just 25% of the full dataset (dotted line) it gets very close to the highest MPR obtained by the model trained on the full dataset (solid line) already after the first 5 epochs. Interestingly, the linguistic metrics do not get worse ei-

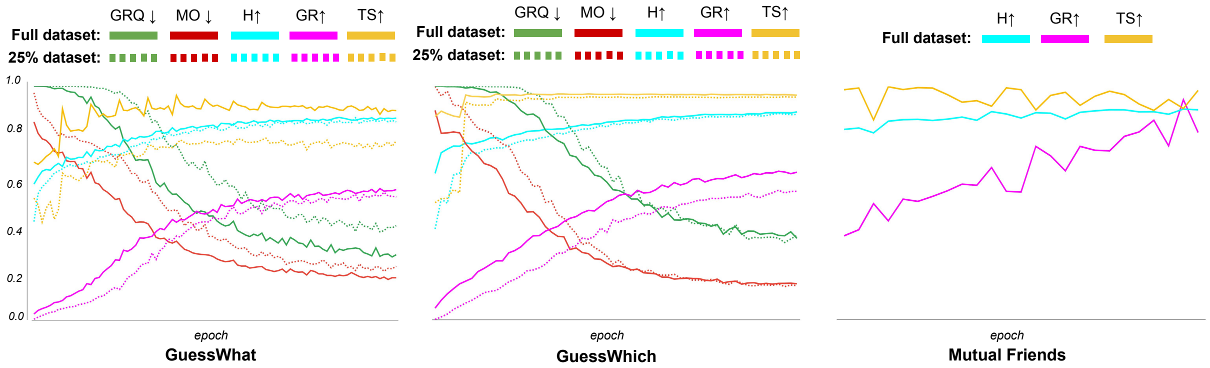


Figure 2: Comparison across epochs and by downsizing the training data using the following metrics: Task Success (TS), Games with Repeated Questions (GRQ), Mutual Overlap (MO), Unigram Entropy (H) and Global Recall (GR); all metrics are scaled between 0 and 1 (y-axis). **Left:** GDSE-SL on GuessWhat. **Middle**: ReCap on GuessWhich. **Right**: DynoNet on MutualFriends. Downsizing the training data has a higher impact, both for TS and some linguistic metrics, in GuessWhat than in GuessWhich. Among the linguistic metrics, entropy is the most stable and GR increases through the epochs in all tasks. For readability, we have not reported Local Recall-d since its pattern is very close to GR. The dataset of MutualFriends is too small to analyse the effect of downsizing it.

ther, with the only exception of GR. However, in GuessWhat the TS decreases when downsizing the training data (again, yellow solid vs. dotted lines) and dialogues quality is affected too (with the exception of entropy and GR). This result shows that **in GuessWhat *how well* the model learns to ground language plays an important role and affects the TS.** In the next experiment, we aim to further understand the difference between the two visual tasks and when LD could impact TS in GuessWhat.

5.3 When Could Language Quality Impact Task Success?

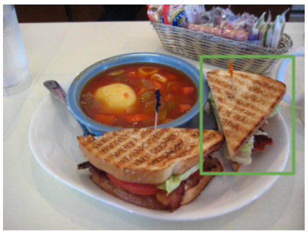
First of all, we check the extent to which the dialogue is used by the Guesser module to pick the correct target. Secondly, we evaluate whether the quality of the dialogues could lead to higher task success. Finally, we pinpoint when a lower LD could contribute to succeed in the task.

The role of the dialogues on TS We run a by-turn evaluation checking whether the information incrementally gained through the dialogue brings increased performance. We evaluate ReCap on GuessWhich and GDSE-SL on GuessWhat. We find that the performance of ReCap is flat across the dialogue turns, confirming results reported in Murahari et al. (2019) for other models. Instead, the performance of GDSE-SL keeps on increasing at each turn from the beginning till the end of the dialogue, though the increase in the first 3 turns is higher than in the later ones (details in Appendix A). **This suggests that in GuessWhich the role of the dialogue is rather limited.** This might be due to the highly informative image caption that GuessWhich models receive together with the dialogue to solve the guessing task (Testoni et al., 2019). Instead, in **GuessWhat dialogues do play a major role in the guessing task**. Hence, we focus on this dataset to understand whether and when the quality of the dialogue could lead to a higher task success.

Impact of the quality of dialogues on TS To check whether the Guesser could profit from dialogues of better quality, we evaluate GDSE-SL using human dialogues. When given human dialogues, the model reaches an accuracy of 60.6%, which is +8.5% higher than the one it achieves with the dialogues generated by its decoder (52.1%). One hypothesis could be that this higher TS is due to the mistakes produced by the A-bot when using instead the generated dialogues, but this is not the case: we have evaluated the model when receiving human questions paired with the A-bot's answers for each question and the accuracy drops of only 2.5%. This experiment suggests that **a lower LD could indeed lead to a higher TS.**

The role of less frequent words As we have observed above, models mostly use very frequent words. Here, we aim to understand to what extent this penalizes GuessWhat models. In this dataset, more than half (55%) of the words in the vocabulary are used less than 15 times in the training set. We refer to this set of words as "*rare*" words: most

Human dialogues		Generated dialogues	
Questioner	**Answerer**	**Questioner**	**Answerer**
1. Is it a man?	No	1. Is it a person?	No
2. Is it food?	Yes	2. Is it food?	Yes
3. Is *pancake*?	No	3. Is it pizza?	Yes
4. Is egg?	Yes	4. Is it the pizza in front?	Yes
		5. . . .	
$\leadsto$ **model suceeds guessing**		$\leadsto$ **model fails guessing**	



Human dialogues		Generated dialogues	
Questioner	**Answerer**	**Questioner**	**Answerer**
1.Is it edible?	Yes	1. Is it food?	Yes
2.Is it a sandwich?	Yes	2. Is it a sandwich?	Yes
3. Does it have an orange *toothpick* in it?	Yes	3. Is it on the right?	Yes
		. . .	
$\leadsto$ **model fails guessing**		$\leadsto$ **model succeeds guessing**	

Figure 3: Examples of GuessWhat games in which humans use "rare" words (rare words in italic) and the corresponding generated dialogues. The failure of the model could be due to the inability to generate (top) or to encode (bottom) rare words.

Figure 4: In GuessWhat, longer human dialogues contain more rare words, more distractors and more distractors of the same category of the target object.

of them are nouns (79%) or verbs (11%) (e.g., "*feline*", "*forest*", "*compute*", "*highlight*").

We check whether there is a relation between rare words and difficult games. Human dialogue length is a good proxy of the difficulty of the games, both for humans and models. Figure 4 illustrates some statistics about human dialogues: games for which humans ask more questions are about images with a higher number of distractors, with a higher number of distractors of the same category of the target, and with a higher number of "rare" words. In 10% of the games in the test set, humans have used at least one rare word. These dialogues are longer than those that do not contain rare words (resp., 7.8 vs. 4.7 turns on average). Interestingly, the accuracy of the model on these games is lower than the overall accuracy: -13.8% (48.7% vs. 62.5%) when evaluating it with human dialogues and -8.3% (45% vs. 53.3%) when using the dialogues generated by the model itself. Moreover, the accuracy reached by the model in the latter setting is lower when comparing games for which humans have used a higher number of rare words. Overall, we found that 65% of the rare words in the human test set show up in games that the model is not able to solve correctly.

Figure 3 shows some examples of games in which humans have used a rare word. It illustrates the human vs. generated dialogue and whether the model succeeds in guessing the target object when receiving the former or the latter. The failure of the model in guessing the target object could be due to its inability to generate or encode rare words. The example on top shows that if the model fails to generate an appropriate word (e.g. the rare word "*pancake*") this can have a domino effect on the next words and the next questions it generates. On the other hand, the model can fail to encode rare words, e.g., "*toothpick*" in Figure 3-bottom. The inability to generate rare words could be mitigated by developing dialogue strategies that produce less natural but still informative dialogues. For instance, in the example at the bottom, the model avoids using "*toothpick*" by asking a spatial question ("*Is it on the right?*") which is rather informative for the Guesser since it has the coordinates of each candidate object. These observations show that current models fail to properly ground and use *rare words* and suggest that, in some contexts, **the use of only frequent words could be behind the failure in reaching the communication goal.**

6 Conclusion

Our work highlights the different complexity of two sub-tasks involved in referential guessing (visual) games: guessing the target and asking questions. We have shown that while learning to win the game can be reached in a rather short time (with a few epochs), learning to generate rich human-like dialogues takes much longer. This holds for all three tasks and models we have scrutinized independently of the size of the vocabulary, the task, and the learning paradigm used. Therefore, choosing the best model only on the base of the task success could prevent the model from generating more human-like dialogues. We have shown that in GuessWhich decreasing the size of the training set does not bring a drop in either TS (task success, higher is better) or in LD (linguistic divergence, lower is better) and, moreover, the dialogues play a minor role on TS. Instead, for GuessWhat, decreasing the size of the training dataset brings a decrease in TS and an increase in LD, and, through dialogues, models accumulate information to succeed in the task. Hence, we have focused our in-depth analysis on GuessWhat. Furthermore, we have investigated whether and when higher language quality could lead to higher task success. We have shown that if models are given human dialogues, they can reach a higher TS. Hence, LD could boost TS. We have shown that this boost could help more in difficult games, i.e. those for which humans ask longer dialogues. These games contain images with more distractors and humans use less frequent words while playing them. Hence, we claim that in GuessWhat models could increase their accuracy if they learn to ground, encode and decode words that do not occur frequently occur in the training set.

In the paper, we propose the LD metric that, despite its limitations (i.e., being based only on surface cues) represents a proxy of the quality of dialogues. We believe LD effectively captures the most common deficiencies of current models and it allows a straightforward comparison between different models. As future work, LD can be used as a training signal to improve the quality of generated dialogues. Moreover, a comparison between human quality judgments and LD may shed some light on the strengths and weaknesses of this metric. Further work is needed to design new metrics that capture more fine-grained phenomena and better evaluate the quality of generated dialogues.

Acknowledgements

The authors kindly acknowledge the support of NVIDIA Corporation with the donation of the GPUs used in our research. We are grateful to SAP for supporting the work. We would like to thank the following people for their suggestions and comments: Luciana Benotti, Guillem Collell, Stella Frank, Claudio Greco, Aurelie Herbelot, Sandro Pezzelle, and Barbara Plank. Finally, we thank the anonymous reviewers for the insightful feedback.

References

Ehsan Abbasnejad, Qi Wu, Javen Shi, and Anton van den Hengel. 2019. What's to know? Uncertainty as a guide to asking goal-oriented questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4155–4164.

Elia Bruni and Raquel Fernández. 2017. Adversarial evaluation for open-domain dialogue generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 284–288.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (System Demonstrations)*, pages 169–174.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra. 2017a. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335.

Abhishek Das, Satwik Kottur, José M.F. Moura, Stefan Lee, and Dhruv Batra. 2017b. Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning. In *2017 IEEE International Conference on Computer Vision*, pages 2951–2960.

Zhe Gan, Yu Cheng, Ahmed EI Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. 2019. Multi-step reasoning via recurrent dual attention for visual dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6463–6474.

Fenfei Guo, Angeliki Metallinou, Chandra Khatri, Anirudh Raju, Anu Venkatesh, and Ashwin Ram.

2017. Topic-based evaluation for conversational bots. In *Proceedings ofthe 31st Conference on Neural Information Processing Systems (NIPS)*.

Janosch Haber, Tim Baumgärtner, Ece Takmaz, Lieke Gelderloos, Elia Bruni, and Raquel Fernández. 2019. The PhotoBook dataset: Building common ground through visually-grounded dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1895–1910.

Tatsunori B. Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1689–1701.

H. He, A. Balakrishnan, M. Eric, and P. Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1766–1776.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *to appear in Proceedings of ICLR 2020*.

Nikolai Ilinykh, Sina Zarrieß, and David Schlangen. 2019. Tell Me More: A Dataset of Visual Scene Description Sequences. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 152–157.

Anjuli Kannan and Oriol Vinyals. 2016. Adversarial evaluation of dialogue models. In *NIPS 2016 Workshop on Adversarial Training*.

Mike Lewis, Denis Yarats, Yann N. Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or No Deal? End-to-End learning for negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2019. Don't say that! Making inconsistent dialogue unlikely with unlikelihood training. ArXiv:1911.03860.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an automatic Turing test: Learning to evaluate dialogue responses. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Emiel van Miltenburg, Desmond Elliott, and Piek Vossen. 2019. Measuring the diversity of automatic image descriptions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1730–1741.

Nasrin Mostafazadeh, Chris Brockett, Bill Dolan, Michel Galley, Jianfeng Gao, Georgios P. Spithourakis, and Lucy Vanderwende. 2017. Image-grounded conversations: Multimodal context for natural question and response generation. In *Proceedings of the The 8th International Joint Conference on Natural Language Processing*, pages 462–472.

Vishvak Murahari, Prithvijit Chattopadhyay, Dhruv Batra, Devi Parikh, and Abhishek Das. 2019. Improving generative visual dialog by answering diverse questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1449–1454.

Wei Pang and Xiaojie Wang. 2020. Visual dialogue state tracking for question generation. In *Proceedings of 34th AAAI Conference on Artificial Intelligence*.

Arijit Ray, Karan Sikka, Ajay Divakaran, Stefan Lee, and Giedrius Burachas. 2019. Sunny and dark outside?! improving answer consistency in VQA through entailed question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5863–5868, Hong Kong, China. Association for Computational Linguistics.

Lee Sang-Woo, Gao Tong, Yang Sohee, Yao Jaejun, and Ha Jung-Woo. 2019. Large-scale answerer in questioner's mind for visual dialog question generation. In *Proceedings of International Conference on Learning Representations, ICLR*.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? How controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1702–1723.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and

Raquel Fernández. 2019. Beyond task success: A closer look at jointly learning to see, ask, and GuessWhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2578–2587.

Pushkar Shuklar, Carlos Elmadjian, Richika Sharan, Vivek Kulkarni, William Yang Wang, and Matthew Turk. 2019. What should I ask? Using conversationally informative rewards for goal-oriented visual dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6442–6451.

Florian Strub, Harm de Vries, Jeremie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. 2017. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2765–2771.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Alberto Testoni, Ravi Shekhar, Raquel Fernández, and Raffaella Bernardi. 2019. The devil is in the detail: A magnifying glass for the GuessWhich visual dialogue game. In *Proceedings of the 23rd SemDial Workshop on the Semantics and Pragmatics of Dialogue (LondonLogue)*, pages 15–24.

Takuma Udagawa and Akiko Aizawa. 2019. A natural language corpus of common grounding under continuous and partially-observable context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7120–7127.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *ICML Deep Learning Workshop*.

Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. 2017. GuessWhat?! Visual object discovery through multi-modal dialogue. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 5503–5512.

Jiawei Wu, Xin Wang, and William Yang Wang. 2019. Self-supervised dialogue learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3857–3867.

Jiaping Zhang, Tiancheng Zhao, and Zhou Yu. 2018a. Multimodal hierarchical reinforcement learning policy for task-oriented visual dialog. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 140–150.

Junjie Zhang, Qi Wu, Chunhua Shen, Jian Zhang, Jianfeng Lu, and Anton van den Hengel. 2018b. Goal-oriented visual question generation via intermediate rewards. In *Proceedings of the European Conference of Computer Vision (ECCV)*, pages 186–201.

Rui Zhao and Volker Tresp. 2018. Improving goal-oriented visual dialog agents via advanced recurrent nets with tempered policy gradient. In *Proceedings of IJCAI*.

Mingyang Zhou, Josh Arnold, and Zhou Yu. 2019. Building task-oriented visual dialog systems through alternative optimization between dialog policy and language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 143–153.

A Appendix A

Figure 5 reports the token frequency curve for human dialogues and generated dialogues on the GuessWhat test set (Zipf's law). Human dialogues are clearly more rich and diverse compared to generated dialogues.

Figure 6 and Figure 7 show the per-turn accuracy of GDSE-SL for GuessWhat and ReCap for GuessWhich, respectively. For GuessWhat, we report the simple task accuracy on the game, while for GuessWhich we use the Mean Percentile Rank; please refer to the main paper for additional details. For GuessWhat, the accuracy keeps increasing while new turns are given as input to the model. For GuessWhich, on the other hand, the Mean Percentile Rank (MPR) is pretty stable after very few turns and it is already high at turn 0, i.e. when only the caption is provided without any dialogue history.

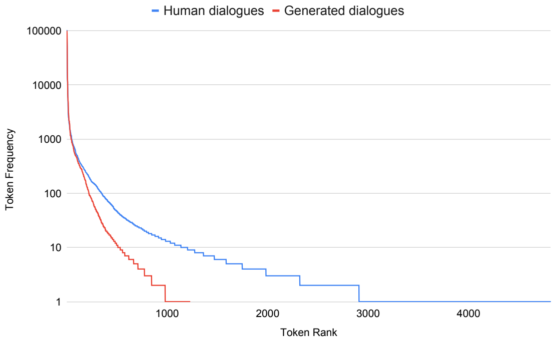


Figure 5: Token frequency plot (Zipf's Law curve) for human vs. generated dialogues

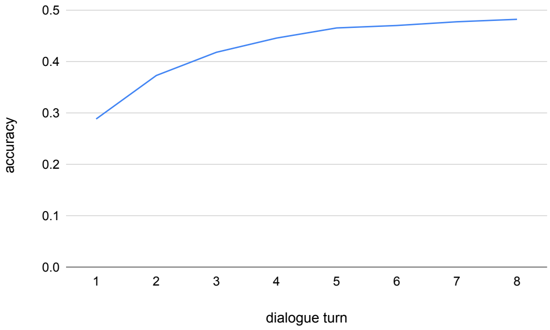


Figure 6: GDSE-SL per-turn accuracy on the GuessWhat game.

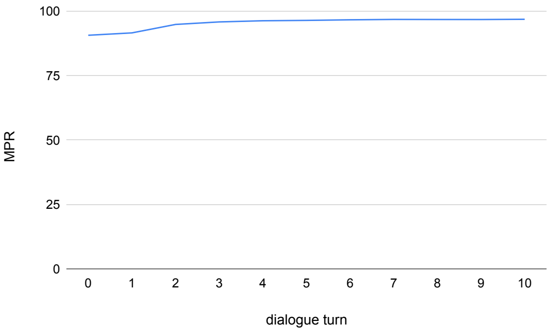


Figure 7: ReCap per-turn Mean Percentile Rank (MPR) on the GuessWhich game

“Are you kidding me?”:
Detecting Unpalatable Questions on Reddit

Sunyam Bagga
.txtLAB
McGill University
sunyam.bagga@mcgill.ca

Andrew Piper
Dept. of Languages,
Literatures, & Cultures
McGill University
andrew.piper@mcgill.ca

Derek Ruths
School of Computer Science
McGill University
derek.ruths@mcgill.ca

Abstract

Abusive language in online discourse negatively affects a large number of social media users. Many computational methods have been proposed to address this issue of online abuse. The existing work, however, tends to focus on detecting the more explicit forms of abuse leaving the subtler forms of abuse largely untouched. Our work addresses this gap by making three core contributions. First, inspired by the theory of impoliteness, we propose a novel task of detecting a subtler form of abuse, namely *unpalatable questions*. Second, we publish a context-aware dataset for the task using data from a diverse set of Reddit communities. Third, we implement a wide array of learning models and also investigate the benefits of incorporating conversational context into computational models. Our results show that modeling subtle abuse is feasible but difficult due to the language involved being highly nuanced and context-sensitive. We hope that future research in the field will address such subtle forms of abuse since their harm currently passes unnoticed through existing detection systems.

1 Introduction

Abusive language and other antisocial behaviour is omnipresent in online discourse. According to a recent survey, 41% of Americans have personally experienced some form of online harassment (Duggan, 2017). To counter abusive behaviour online, different social media platforms implement their own mechanisms such as content moderation, muting or blocking users from posting etc. It is, however, infeasible to *manually* moderate online communities due to the sheer enormity of content produced every day – Twitter, for example, receives over 500 million tweets per day. Manual moderation in such a scenario would require humans to read millions of tweets daily which would take an impractical amount of time and other resources. Consequently, many computational models have been proposed by the Natural Language Processing (NLP) community to detect online abuse and facilitate automatic content moderation.

Abuse is an umbrella term which can cover several types of negative expressions. There exists a plethora of abuse detection studies employing different terminology: personal attacks (Wulczyn et al., 2017), bullying (Dadvar et al., 2013; Chatzakou et al., 2017), hate speech (Warner and Hirschberg, 2012; Davidson et al., 2017; Djuric et al., 2015; Gao and Huang, 2017), nastiness (Samghabadi et al., 2017), harassment (Golbeck et al., 2017; Yin et al., 2009), hostility (Liu et al., 2018), racism or sexism (Waseem and Hovy, 2016), abusive language (Nobata et al., 2016), aggression (Caines et al., 2018), and others. However, extant work in abuse detection has largely focused on detecting overt abuse ignoring the more subtle forms of abuse which can be just as damaging. This is also noted in a recent survey calling on the NLP community to rethink and expand what constitutes abuse (Jurgens et al., 2019).

In this work, we make three contributions to address this gap in the literature. First, inspired from the theory of linguistic impoliteness, we propose a novel task of detecting a subtler form of abuse called *unpalatable questions* (UQ). It is one of the conventionalized impoliteness formulae introduced by Culpeper (2010). We define the UQ task as detecting a negatively phrased question designed to antagonise its recipient in online discourse.

Second, we collect, annotate, and make publicly available a context-aware dataset for the UQ task.[1] The data comes from a diverse set of online communities (or *subreddits*) on the popular social media site Reddit. Most existing datasets used in abuse

---

[1] We make our dataset and code publicly available at https://github.com/networkdynamics/unpalatable-questions.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2083–2099
April 19 - 23, 2021.

detection (Wulczyn et al., 2017; Waseem and Hovy, 2016; Davidson et al., 2017; Founta et al., 2018; Golbeck et al., 2017) only include annotations for stand-alone comments or tweets. In comparison, we explicitly consider conversational context during annotation and preserve contextual information in our dataset (see Section 4.4 for a detailed comparison).

A major limitation of existing abuse detection studies – also pointed out in (Castelle, 2018; Mishra et al., 2019; Gao and Huang, 2017) – is that a comment is treated as a single-utterance in isolation, ignoring any conversational context provided by other comments in the discussion. This is problematic since abuse is inherently contextual, and it becomes a major issue when working with subtler forms of abuse such as unpalatable questions. To this end, our third contribution is that we implement a wide array of learning models to detect unpalatable questions and investigate the benefits of incorporating conversational context in our computational models.

2 What is an Unpalatable Question?

We adopt the term *unpalatable question* (UQ) from the conventionalised impoliteness formulae introduced by Culpeper (2010). Although Culpeper did not formally define UQ, several examples were laid out: ‘why do you make my life impossible?’, ‘which lie are you telling me?’, ‘what’s gone wrong now?’. We find that UQs tend to be rhetorical in nature in that they are usually asked not to elicit an answer but to make a point. In particular, they have a close resemblance to *epiplexis*: a type of rhetorical question which is asked not to elicit information but to reproach, upbraid, or rebuke (Zimmerman, 2005). This can be seen in the examples listed in Table 1, where the questions are asked to shame the interlocutor for adopting a particular point of view and are often insults asked as questions. For our task, we define an *unpalatable question* as a negatively phrased question designed to antagonise its recipient.

Why UQ? Jurgens et al. (2019) outline a spectrum of abusive behaviour highlighting that existing work only focuses on overt abuse ignoring both the subtler forms and extreme behaviours. As can be seen in Figure 1, our task of detecting a subtler form of abuse is a step towards addressing this gap. Moreover, studies in linguistics show that being asked an unpalatable question puts the recipient in a vulnerable position to receive further verbal attacks (Bousfield, 2007; Wijayanto et al., 2017).

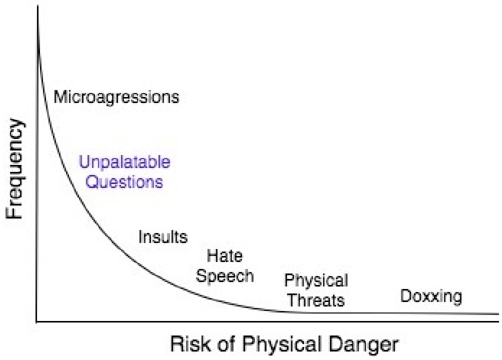


Figure 1: This figure, taken from (Jurgens et al., 2019), illustrates where *unpalatable questions* fit in a hypothetical spectrum of online abuse.

3 Further Related Work

3.1 Abusive Language Detection

Work on abuse detection has studied specific types of abuse using several feature-based and deep learning approaches. One of the earliest studies was by Yin et al. (2009) employing SVM to detect ‘personal insult harassment’ using TF-IDF values for words and sentiment-based features. Using a similar but enhanced set of features, Davidson et al. (2017) implement Logistic Regression and SVM to detect hate speech and offensive language on Twitter. Warner and Hirschberg (2012) use a template-based strategy to extract features from text and a linear-SVM to detect hate speech with a focus on anti-semitic language. Djuric et al. (2015) report an AUC of 80% using Logistic Regression with $paragraph2vec$ which outperformed standard bag-of-words approaches. Nobata et al. (2016) also use word2vec and comment2vec as one of their features to detect ‘abusive language’ which, in their work, encompasses hate speech, profanity and derogatory language. Wulczyn et al. (2017) implement a multilayer perceptron with word and character n-grams to detect personal attacks on Wikipedia and report an AUC of 96.5%.

In recent years, deep learning techniques have been widely adopted to detect online abuse. Pavlopoulos et al. (2017) show that RNN with GRU cells outperform the original classifier on detecting personal attacks (Wulczyn et al., 2017). Park and Fung (2017) propose a Hybrid-CNN that uses both word-level and character-level CNNs to detect hate speech on Twitter. Aken et al. (2018) utilise

Preceding Comment	Main Comment (Reply)
They were safe in turkey.	Turkey isn't even safe for (Kurdish) Turks. *Do you even watch the news?*
NH dems or Bernie bros?	*You don't really understand that term "Berniebro" do you?* Just parroting it.
Fuck the refs. Fuck the stars. Fuck Texas and every shit-kick fuck that calls it home.	*Would you like some salt with that?*
SJW's happened.	*Why the fuck did you put an apostrophe there?*
So no article, right? Thought not.	*Oh, your brain stopped functioning at that?* Well then, I'll repeat myself. The abstract is enough, the article, you can find it yourself. I'm not going to waste my time.

Table 1: Examples of unpalatable questions from our annotated dataset.

CNN, bi-directional LSTM and GRU initialised with pre-trained word embeddings and report a f1-score of 78.3% and AUC of 98.3%. Gunasekara and Nejadgholi (2018) implement a Light Gradient Boosting Machine stacking model where they combine two bidirectional-LSTM based architectures and report an AUC of 98.6%. Finally, some studies also incorporate user-level context in their models. Mishra et al. (2018) report improvements in performance by incorporating author embedding features using $node2vec$ on community graphs of Twitter users. Dadvar et al. (2013) employs both user-based features and standard content-based features to detect bullying on YouTube. There are other studies that employ user-based features to detect aggression (Chatzakou et al., 2017) and hate speech (Gao and Huang, 2017).

3.2 Linguistic Impoliteness

Long before detecting online abuse gained attention, there had been significant research on linguistic impoliteness. The most notable contribution in this field is by Culpeper (1996) who introduced his theory of impoliteness as a parallel to Brown and Levinson (1987)'s politeness theory. Impoliteness is defined as the use of strategies to attack the interlocutor's *face* – a persona that one presents in a conversation (Goffman, 1967) – and create social disruption (Culpeper, 1996). More recently, Culpeper (2010) offered conventionalized impoliteness formulae for English derived from his corpora that consisted of phone calls, 'exploitative' TV shows, army training documentaries etc. He identified candidates for impoliteness and grouped them according to structural commonalities:[2]

- *Insults*: you f*cking moron; you disgust me
- *Pointed criticisms*: this was absolutely terrible
- *Unpalatable questions*: why do you make my life impossible?
- *Dismissals*: piss off; get lost
- *Threats*: I'm gonna beat the sh*t out of you if you don't [X]

3.3 Rhetorical Questions

Rhetorical questions are defined as sentences "that have the form of a question but serve as a statement" (Anzilotti, 1982). Since unpalatable questions tend to be rhetorical in nature, we present a brief overview of the literature on rhetorical question detection in social media.

One of the first studies was on Twitter data by Li et al. (2011) where they distinguish 'qweets' – tweets that ask for some information – from other interrogative tweets including rhetorical questions. They implement SVM using a set of different hand-crafted features. Bhattasali et al. (2015) use bag of n-grams to detect rhetorical questions in the Switchboard Dialogue Corpus. Their best-performing model achieved a F1-score of 0.53 by incorporating both preceding and subsequent text. Using questions from Twitter and Debate Forums, Oraby et al. (2017) implement SVM and LSTM to detect rhetorical questions, and further distinguish between sarcastic rhetorical questions and other questions. There exists other studies modeling rhetorical questions that draw inspiration from linguistic theories behind the motivations of users to post rhetorical questions (Ranganath et al., 2016, 2018). A general consensus in these studies is that rhetorical questions are hard to accurately classify due to their syntactic similarity to regular questions.

4 Data

The aim is to detect unpalatable questions in online discourse. For this, we construct a dataset using comments from Reddit and annotate them

---

[2] See (Culpeper, 2010) for a complete list and additional examples of each form.

for whether they contain an unpalatable question or not. We also preserve conversational context in the dataset by including the preceding comment in the discussion; therefore, our data consists of (pc_i, r_i, y_i) tuples denoting the preceding comment, reply[3] (or main comment), and the corresponding label respectively. The task is formulated as a binary classification problem where $y_i = 1$ indicates that the main comment r_i contains an unpalatable question.

We collect data from a diverse set of 15 online communities (or *subreddits*) belonging to different genres: politics, sports, hate and toxic.[4] The subreddits were carefully selected to prevent the dataset from being heavily skewed towards *not unpalatable* samples since these topics are more likely to involve opinionated and antagonistic discussions.

4.1 Question Filter

A challenge during data collection was to filter out comments that did not contain a question. We experiment with two approaches: (1) simple rule-based approach where we tokenize the comment and extract sentences that end with a '?', and (2) parsing-based approach where we first generate constituent parse trees using Stanford CoreNLP (Manning et al., 2014) and then identify questions using clause-level Penn Treebank Tags.[5]

Performance Comparison. We manually annotated a random sample of 300 Reddit comments for the presence of questions. There were a total of 81 questions out of which 74 contained a '?'. Although the parsing-based approach achieved a high precision, it missed out on several questions due to the low accuracy of the parser on noisy social media text. On the other hand, the simpler rule-based approach achieved a much higher recall and only missed out on the 7 samples that did not contain a '?'. Given the high disparity in performance, we decided to use the rule-based approach as our question filter. Although a potential data limitation, it is an acceptable design decision given that 91% of questions in our random sample were explicitly phrased using a '?'. This simple '?' heuristic has also been successfully used for identifying questions in other social media studies (Zhao and Mei, 2013; Ranganath et al., 2016; Paul et al., 2011).

	Confidence **0.6**	 **0.8**	 **1.0**	*Total*
Unpalatable	879	585	453	1917
Not Unpalatable	1324	2386	5282	8992
Total	2203	2971	5735	10,909

Table 2: Distribution of confidence scores in the annotated dataset.

4.2 Crowdsourcing

We use Amazon Mechanical Turk for crowdsourcing our data annotations. The coders were shown the *main comment* and also the *preceding comment* for context. They were asked to label the main comment for whether it contained an unpalatable question or not. Each comment in our dataset is labeled by at least five different coders.

Quality Control. Since coders can sometimes be unreliable at labeling abusive content (Nobata et al., 2016), we employ three measures to ensure high quality annotations. First, we were able to provide high-quality training to our coders through the use of clear instructions that laid out detailed tips, examples, and counter examples.[6] Second, we allowed only qualified coders to contribute to the task – they were required to achieve a perfect score on a quiz which had a total of 10 questions. Third, we inserted secret test questions throughout our task to address the issue of spam responses (Kittur et al., 2008). The coders were disqualified and blocked if their accuracy on the test questions fell below our predefined threshold of 90%.

4.3 Data Description

We aggregated the five annotations by taking the majority as the final label – a data sample is considered *unpalatable* if at least 3 coders labeled it as *unpalatable*. In order to not lose useful information, we added a *confidence* dimension to the dataset which is the ratio of the number of annotations with the majority label and the total number of annotators: *confidence* $\in \{0.6, 0.8, 1.0\}$. As can be seen in Table 2, 1,917 (17.5%) comments contain an *unpalatable* question, and the remaining 82.5% of comments do not. It is interesting to note the distribution of confidence scores across the two labels. Annotators seem to be much more confident

---

[3] Note that we use the term *main comment* and *reply* interchangeably.

[4] See Appendix A for the complete list of subreddits.

[5] https://gist.github.com/nlothian/9240750

[6] This was done through multiple in-house annotation rounds where we improved the instructions at each step. See Appendix B for the instructions shown to the coders.

for *not unpalatable* samples: 58% of samples correspond to a confidence score of 1.0 as compared to 25% for *unpalatable* samples. Following a similar trend, 45% of comments labeled *unpalatable* have a confidence score of 0.6 as compared to only 14% for *not unpalatable* samples. This highlights the nuanced nature and complexity associated with identifying unpalatable questions.

Annotator Agreement. We compute two measures of inter-annotator reliability: (1) Cohen's Kappa, and (2) Krippendorff's alpha. Our data achieved a Kappa score of 0.82 against a random sample of 150 comments manually annotated by the authors. Out of 150 comments, there were a total of 8 instances of disagreement – 7 out of which had a confidence score of 0.6. Next, we compute Krippendorff's α which is used when there are multiple coders annotating overlapping but different sets of comments (Krippendorff, 2004). Our data achieved an $\alpha = 0.39$ which is in-line with other abuse detection work that used crowdsourcing (Wulczyn et al., 2017; Cheng et al., 2015).

4.4 Comparison with Existing Datasets

As previously discussed, most datasets used for abuse detection contain annotations only for standalone comments in isolation. This is problematic since offensiveness can highly depend on the context. Castelle (2018) shows how their learning models failed (F1 = 0.3) on a StackOverflow dataset that required contextual enrichment to determine the offensiveness of a comment – a majority of comments, that were originally flagged as offensive, were not considered offensive by their coders. This is because the dataset lacked interactional context which was available to StackOverflow users when they originally flagged it as offensive.

We are aware of the following existing datasets that include contextual information:[7]

- Karan and Šnajder (2019) published a large dataset of 400k comments from Wikipedia including complete discussion threads. However, a major limitation of their data is that the labels are generated automatically using an existing toxicity classifier.[8] This implies that their labels would not be accurate for comments where the original toxicity classifier itself fails. In comparison, we perform manual annotation where our coders explicitly consider interactional context.
- Liu et al. (2018) published a dataset of 30,987 comments from Instagram annotated for *hostility*. The coders were shown an Instagram post and all comments in the thread. However, their data collection is biased (intentionally) towards teenagers and is filtered by certain keywords, eg: profanities, emojis. In comparison, our dataset involves a random sample of comments from a diverse set of subreddits without the use of any keyword-filtering.
- Gao and Huang (2017) released a dataset of 10 complete discussion threads from Fox News. The data includes additional contextual information in the form of user screen name, other comments in the thread, and title of the news article. However, their dataset is much smaller with only 1,528 comments and includes only two annotations per comment. In comparison, we use at least five annotations for each of the 10,909 comments in our dataset.

5 Methodology

In this section, we introduce our methodology for detecting unpalatable questions.

5.1 Traditional Machine Learning

We implement Logistic Regression[9] using a diverse set of features:

- **N-grams**: We use TF-IDF values for word unigrams, bigrams, and trigrams. We also utilise character trigrams, 4-grams, and 5-grams.
- **Embeddings**: We experiment with several pre-trained 300-dimensional embeddings: word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and Fasttext (Bojanowski et al., 2017). In addition, we trained a word2vec model[10] from scratch on Reddit using Gensim (Řehůřek and Sojka, 2010).
- **Writing Style**: This category captures writing style of the comment and includes the following features: total number of words,

---

[7] Additionally, Zhang et al. (2018) released a dataset to *pre-emptively* detect toxic comments given all preceding comments in the discussion focusing on "personal attacks."

[8] http://www.perspectiveapi.com

[9] Note that additional experiments with SVM yielded similar results (not shown here).

[10] Specifically, we train the continuous bag-of-words (CBOW) architecture on all of Reddit data from 2016.

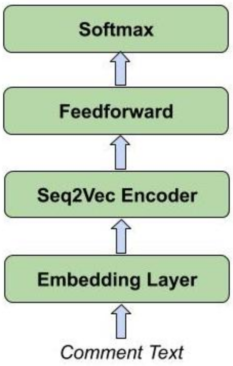


Figure 2: The skeletal architecture for our deep learning models.

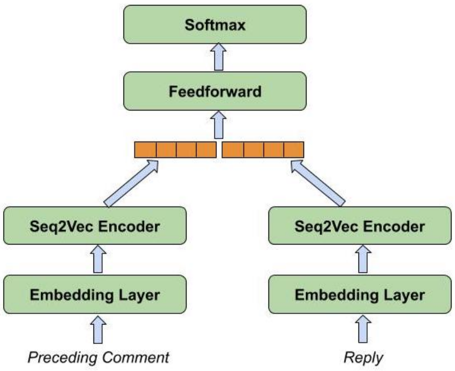


Figure 3: The skeletal architecture for our deep learning models that incorporate interactional context.

capital words, question marks, exclamation marks and second person pronouns.

- **Lexicon-based**: This category includes three features computed using pre-defined lexicons:
  - We compute the number of non-English words by comparing against NLTK words (Loper and Bird, 2002) and Enchant's dictionary.[11]
  - We compute the number of toxic words using a lexicon compiled from different sources: list of bad words released by Google[12] and lexicons released by other studies (Chandrasekharan et al., 2017; Davidson et al., 2017).
  - Empath (Fast et al., 2016) provides a set of 200 built-in validated categories for analysing text. We hand-picked a set of 15 relevant categories, for example: aggression, disgust, hate, shame etc.
- **Sentiment**: We use the *positive*, *negative*, and *neutral* sentiment scores returned by VADER, a rule-based sentiment analyser built for social media text (Hutto and Gilbert, 2014).

To build feature vectors, we experiment with the features in isolation as well as several combinations of these feature categories. The feature vectors along with the corresponding labels y_i are then fed to the learning algorithm, which is implemented using scikit-learn (Pedregosa et al., 2011).

5.2 Deep Learning

Deep learning models have been successfully used in many abuse detection studies (Pavlopoulos et al., 2017; Park and Fung, 2017; Aken et al., 2018). In this work, we implement a number of deep learning models – both CNN and RNN-based – using the architecture shown in Figure 2. We use pre-trained GloVe embeddings for the embedding layer.[13] An encoder is responsible for condensing a sequence of word vectors to a single vector. We experiment with a number of neural networks for the encoder: CNN, LSTM, Bidirectional LSTM, and Stacked Bidirectional LSTM.

ELMo. For the embedding layer, we also experiment with deep contextualized ELMo (Peters et al., 2018) representations[14] as an alternative to using GloVe embeddings. The encoder layer here can be either a CNN, LSTM, Bi-LSTM, or Stacked Bi-LSTM.

Dense Hybrid. We also implement deep learning models that utilise the various hand-engineered features discussed in Section 5.1. For this, we compute a 'dense' feature vector using those feature categories, and concatenate it with the neural encoder's output. This combined vector is then fed to a fully-connected feedforward neural network followed by a softmax layer.

Implementation details. All models are implemented using AllenNLP (Gardner et al., 2017), an open-source deep learning library for NLP. The training objective is weighted cross entropy loss, and Adam optimizer (Kingma and Ba, 2014) is used for learning network weights. Additionally, early stopping is implemented to terminate training of the neural network once the loss stops improving on a set-aside validation set.

---

[11] https://github.com/rfk/pyenchant

[12] https://code.google.com/archive/p/badwordslist/downloads

[13] We picked GloVe since it showed the best performance in our traditional machine learning experiments.

[14] 'Original 5.5B' model from https://allennlp.org/elmo.

Text Input	Model	F1	AUROC	W-F1	Prec.	Rec.	AUPRC
Reply Text	Dense CNN + ELMo	**0.532**	0.818	0.815	0.459	0.636	0.54
Reply Text + Comment Text	CNN + ELMo	0.52	0.814	0.806	0.444	0.636	0.527
Reply Text + Comment Text	NLI-CNN + ELMo	0.507	0.810	0.805	0.443	0.602	0.515
Reply Text + Comment Text	Dense CNN (GloVe)	0.5	0.8	0.825	0.507	0.496	0.502
Question Text Only	Dense Bi-LSTM + ELMo	0.5	0.809	0.778	0.394	0.692	0.517
Reply Text	word (1, 3)	0.443	0.782	0.823	0.549	0.373	0.481
Reply Text	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.429	0.785	0.825	0.6	0.335	0.491

Table 3: Classification results for learning models on the *All-Data* scenario: *confidence* $\in \{0.6, 0.8, 1.0\}$

Text Input	Model	F1	AUROC	W-F1	Prec.	Rec.	AUPRC
Question Text Only	Dense CNN + ELMo	**0.686**	0.937	0.95	0.672	0.704	0.739
Reply Text + Comment Text	NLI-CNN + ELMo	0.674	0.937	0.95	0.74	0.625	0.721
Reply Text	CNN + ELMo	0.671	0.936	0.949	0.688	0.662	0.727
Question Text Only	Stacked Bi-LSTM + ELMo	0.638	0.931	0.938	0.569	0.73	0.718
Reply Text + Comment Text	Dense CNN (GloVe)	0.63	0.936	0.945	0.754	0.56	0.702
Question Text Only	Writing Style + Lexicon + Sentiment + GloVe	0.618	0.906	0.942	0.671	0.574	0.648
Question Text Only	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.614	0.916	0.944	0.753	0.521	0.672

Table 4: Classification results for learning models on the *High-Agreement-Data* scenario: *confidence* $= 1.0$

5.3 Incorporating Conversational Context

Since humans can better comprehend a comment with reference to its context, we wanted to investigate the benefits of incorporating conversational context in the learning models. For traditional machine learning models, we concatenate the feature vectors for the preceding comment pc_i and main comment r_i which is then fed to the learning algorithm. For deep learning models, we first vectorize the preceding comment pc_i and the main comment r_i using the same encoder pipeline. The two vectors are then concatenated and fed to a feedforward neural network (Figure 3). In the LSTM-based models, the final hidden states of the pc_i and r_i pipeline are concatenated. For CNN, the output of the max pooling layers of the pc_i and r_i pipeline are concatenated.

In addition to simple concatenation, we experiment with additional heuristics to model context inspired from the task of Natural Language Inference (NLI). Specifically, we used a CNN encoder to vectorize the context pc_i and main comment r_i. They are then combined using three-heuristics: (1) concatenation, (2) element-wise product, and (3) element-wise difference (Mou et al., 2016).

6 Experiments and Results

We conduct experiments across two dimensions:

1. **Confidence Score:** We hypothesize that the models would exhibit better performance on data samples which were easier for the coders to annotate. To test this, we experiment with two scenarios:
    - *All-Data*: we use the complete dataset.
    - *High-Agreement-Data*: we use data samples corresponding to *confidence* $= 1.0$.
2. **Text Input:** To investigate the benefits of including contextual information, we experiment with three input scenarios:
    - *Question Text Only*: we only provide the question text as input.
    - *Reply Text*: we provide the full text of the main comment as input.
    - *Reply Text + Comment Text*: In addition to the main comment, we also provide the preceding comment text as input.

We evaluate our computational models on several classification metrics: precision, recall, F1-score, and Area under Receiver Operating Characteristic curve (AUROC) and Precision-Recall curve (AUPRC). All reported values are averaged over stratified five-fold cross-validation runs. The empirical results on *All-Data* and *High-Agreement-Data* are presented in Table 3 and Table 4 respectively.[15]

---

[15] We only display the top-performing traditional and deep learning models here. The complete list of all results is available in Appendix C.

7 Discussion

Among traditional learning algorithms, a combination of simple word unigrams, bigrams, and trigrams achieves the best F1-score of 0.44. Adding other hand-engineered features to $word(1,3)$ results in a better precision and AUPRC. As expected, deep learning models outperform traditional machine learning algorithms for both *All-Data* and *High-Agreement-Data* scenarios. In particular, CNN models perform much better than LSTM models. This is evident from Tables 3 and 4 where the best-CNN model outperforms the best-LSTM model by a 3-point and 5-point increase in F1-score respectively. We observe improvements with using contextualized ELMo embeddings as opposed to static GloVe embeddings for both scenarios. Moreover, the addition of dense hand-engineered feature vector further improves the F1-score to 0.532. Finally, our hypothesis, that it would be easier for the models to classify if it was easier for the humans, holds true in that there is a considerable improvement in the F1-score (15 points) for *High-Agreement-Data*.

Despite the performance gains observed with using more sophisticated deep learning models, the performance is still poor to be used for any practical applications. This is not surprising given how linguistically nuanced our dataset is and the complexity associated with abusive language detection on noisy social media text (Nobata et al., 2016). Specifically, learning models struggle to deal with *implicit* abuse – language which does not immediately convey abuse (Waseem et al., 2017). Aken et al. (2018) find that their toxicity classifier fails on data where there were instances of sarcasm, toxicity without employing swear words, and rhetorical questions. We qualitatively examined a random sample of hundred *unpalatable* samples from our dataset, and found that 65% do not contain swear words and 20% involve sarcasm. Similarly, from a random sample of hundred mis-classified *unpalatable* samples, 72% do not contain swear words and 30% involve sarcasm. Moreover, since unpalatable questions are rhetorical in nature, it is not surprising that learning models performed relatively poorly on the task.

Context. Our assumption was that models would benefit from conversational context since humans find it easier to determine the offensiveness of a comment when provided with some context. It is, however, evident from our empirical results that incorporating context through providing the preceding comment to the model did not improve performance for both traditional machine learning and deep learning models. This finding is consistent with other studies that attempt to incorporate interactional context into their models (Karan and Šnajder, 2019; Lee et al., 2018). We believe that effectively incorporating deeper context, as opposed to just the preceding comment, using more sophisticated methods such as hierarchical neural networks might help improve performance.

Evaluation Metrics. Mishra et al. (2019) observe a problematic trend with several abuse detection studies using AUROC for evaluation. This is not ideal since ROC plots can be deceptive when dealing with imbalanced classification scenarios (Saito and Rehmsmeier, 2015). Since most abuse detection datasets tend to be heavily skewed towards non-abusive samples, this can lead to misleadingly optimistic values for AUROC (also observed in Tables 3 and 4). A better alternative is to report AUPRC which is more robust to imbalanced data since it evaluates the fraction of true positives among positive predictions at different thresholds (Saito and Rehmsmeier, 2015).

8 Conclusion

In this work, we addressed an important gap in the abuse detection literature by introducing a novel task of detecting unpalatable questions. We also released a context-rich dataset for the task and implemented a number of learning models to automatically detect unpalatable questions. Our results show that it is difficult to model subtle abuse due to the language being nuanced and context-sensitive. This calls for advancements in natural language understanding methods that can identify such implicit signals and take pragmatic context into account. We hope that future research would explore other forms of abuse and draw inspiration from related fields such as linguistic impoliteness. Detecting abuse – both overt and subtle – on the Internet would help enhance user's experience online and facilitate civil and productive discussions.

References

Betty Van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for toxic comment classification: An in-depth error analysis. In

Proceedings of the 2nd Workshop on Abusive Language Online (ALW2), pages 33–42, Brussels, Belgium. Association for Computational Linguistics.

Gloria Italiano Anzilotti. 1982. The rhetorical question as an indirect speech device in english and italian. *The Canadian Modern Language Review*, 38(2):290–302.

Shohini Bhattasali, Jeremy Cytryn, Elana Feldman, and Joonsuk Park. 2015. Automatic identification of rhetorical questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 743–749, Beijing, China. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Derek Bousfield. 2007. Impoliteness, preference organization and conducivity. *Multilingua – Journal of Cross-Cultural and Interlanguage Communication*, 26(1):1–33.

Penelope Brown and Stephen Levinson. 1987. *Politeness: Some universals in language usage*. Cambridge University Press, New York, NY, US.

Andrew Caines, Sergio Pastrana, Alice Hutchings, and Paula Buttery. 2018. Aggressive language in an online hacking forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 66–74, Brussels, Belgium. Association for Computational Linguistics.

Michael Castelle. 2018. The linguistic ideologies of deep abusive language classification. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Eshwar Chandrasekharan, Umashanthi Pavalanathan, Anirudh Srinivasan, Adam Glynn, Jacob Eisenstein, and Eric Gilbert. 2017. You can't stay here: The efficacy of reddit's 2015 ban examined through hate speech. *Proc. ACM Hum.-Comput. Interact.*, 1(CSCW):31:1–31:22.

Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Mean birds: Detecting aggression and bullying on twitter. In *Proceedings of the 2017 ACM on Web Science Conference*, WebSci '17, pages 13–22, New York, NY, USA. ACM.

Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2015. Antisocial behavior in online discussion communities. In *Proceedings of ICWSM*.

Jonathan Culpeper. 1996. Towards an anatomy of impoliteness. *Journal of Pragmatics*, 25(3):349–367.

Jonathan Culpeper. 2010. Conventionalised impoliteness formulae. *Journal of Pragmatics*, 42(12):3232 – 3245.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696, Berlin, Heidelberg. Springer Berlin Heidelberg.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 29–30, New York, NY, USA. ACM.

Maeve Duggan. 2017. *Online Harassment 2017*. Pew Research Center.

Ethan Fast, Binbin Chen, and Michael S. Bernstein. 2016. Empath: Understanding topic signals in large-scale text. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 4647–4657, New York, NY, USA. ACM.

Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *International AAAI Conference on Web and Social Media*.

Lei Gao and Ruihong Huang. 2017. Detecting online hate speech using context aware models. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 260–266, Varna, Bulgaria. INCOMA Ltd.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

E. Goffman. 1967. *Interaction Ritual: Essays in Face-to-face Behavior*. Aldine Publishing Company.

Jennifer Golbeck, Zahra Ashktorab, Rashad O. Banjo, Alexandra Berlinger, Siddharth Bhagwan, Cody Buntain, Paul Cheakalos, Alicia A. Geller, Quint Gergory, Rajesh Kumar Gnanasekaran, Raja Rajan Gunasekaran, Kelly M. Hoffman, Jenny Hottle, Vichita Jienjitlert, Shivika Khare, Ryan Lau, Marianna J. Martindale, Shalmali Naik, Heather L. Nixon, Piyush Ramachandran, Kristine M. Rogers, Lisa Rogers, Meghna Sardana Sarin, Gaurav Shahane, Jayanee Thanki, Priyanka Vengataraman, Zijian Wan, and Derek Michael Wu. 2017. A large labeled corpus for online harassment research. In *Proceedings of the 2017 ACM on Web Science Conference*, WebSci '17, pages 229–233, New York, NY, USA. ACM.

Isuru Gunasekara and Isar Nejadgholi. 2018. A review of standard text classification practices for multi-label toxicity identification of online content. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 21–25, Brussels, Belgium. Association for Computational Linguistics.

Clayton J. Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *ICWSM*.

David Jurgens, Libby Hemphill, and Eshwar Chandrasekharan. 2019. A just and comprehensive strategy for using NLP to address online abuse. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3658–3666, Florence, Italy. Association for Computational Linguistics.

Mladen Karan and Jan Šnajder. 2019. Preemptive toxic language detection in Wikipedia comments using thread-level context. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 129–134, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 453–456, New York, NY, USA. ACM.

Klaus Krippendorff. 2004. Reliability in content analysis. *Human Communication Research*, 30(3):411–433.

Younghun Lee, Seunghyun Yoon, and Kyomin Jung. 2018. Comparative studies of detecting abusive language on twitter. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 101–106, Brussels, Belgium. Association for Computational Linguistics.

Baichuan Li, Xiance Si, Michael R. Lyu, Irwin King, and Edward Y. Chang. 2011. Question identification on twitter. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 2477–2480, New York, NY, USA. ACM.

Ping Liu, Joshua Guberman, Libby Hemphill, and Aron Culotta. 2018. Forecasting the presence and intensity of hostility on instagram using linguistic and social features. In *International AAAI Conference on Web and Social Media*.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. 2018. Author profiling for abuse detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1088–1098, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2019. Tackling online abuse: A survey of automated abuse detection methods.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 130–136, Berlin, Germany. Association for Computational Linguistics.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 145–153, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Shereen Oraby, Vrindavan Harrison, Amita Misra, Ellen Riloff, and Marilyn Walker. 2017. Are you serious?: Rhetorical questions and sarcasm in social media dialog. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 310–319, Saarbrücken, Germany. Association for Computational Linguistics.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45, Vancouver, BC, Canada. Association for Computational Linguistics.

Sharoda Paul, Lichan Hong, and Ed Chi. 2011. Is twitter a good place for asking questions? a characterization study. In *International AAAI Conference on Web and Social Media*.

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deeper attention to abusive user content moderation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1125–1135, Copenhagen, Denmark. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Suhas Ranganath, Xia Hu, Jiliang Tang, Suhang Wang, and Huan Liu. 2016. Identifying rhetorical questions in social media. In *International AAAI Conference on Web and Social Media*.

Suhas Ranganath, Xia Hu, Jiliang Tang, Suhang Wang, and Huan Liu. 2018. Understanding and identifying rhetorical questions in social media. *ACM Trans. Intell. Syst. Technol.*, 9(2):17:1–17:22.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Takaya Saito and Marc Rehmsmeier. 2015. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3).

Niloofar Safi Samghabadi, Suraj Maharjan, Alan Sprague, Raquel Diaz-Sprague, and Thamar Solorio. 2017. Detecting nastiness in social media. In *Proceedings of the First Workshop on Abusive Language Online*, pages 63–72, Vancouver, BC, Canada. Association for Computational Linguistics.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada. Association for Computational Linguistics.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Agus Wijayanto, Aryati Prasetyarini, and Mauly Halwat Hikmat. 2017. Impoliteness in efl: Foreign language learners' complaining behaviors across social distance and status levels. *SAGE Open*, 7(3).

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 1391–1399, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Dawei Yin, Brian D. Davison, Zhenzhen Xue, Liangjie Hong, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0.

Justine Zhang, Jonathan Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Dario Taraborelli, and Nithum Thain. 2018. Conversations gone awry: Detecting early signs of conversational failure. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1350–1361, Melbourne, Australia. Association for Computational Linguistics.

Zhe Zhao and Qiaozhu Mei. 2013. Questions about questions: An empirical analysis of information needs on twitter. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, pages 1545–1556, New York, NY, USA. ACM.

Brett Zimmerman. 2005. *Catalogue of Rhetorical and Other Literary Terms in Poe's Works*, pages 107–325. McGill-Queen's University Press.

A Data Collection

We collect data from a diverse set of 15 Reddit communities (or *subreddits*) belonging to different genres:

- **Politics**:

  - r/The_Donald
  - r/politics
  - r/PoliticalDiscussion
  - r/Conservative

- **Sports**:
  - r/nfl
  - r/sports
  - r/nba
  - r/hockey

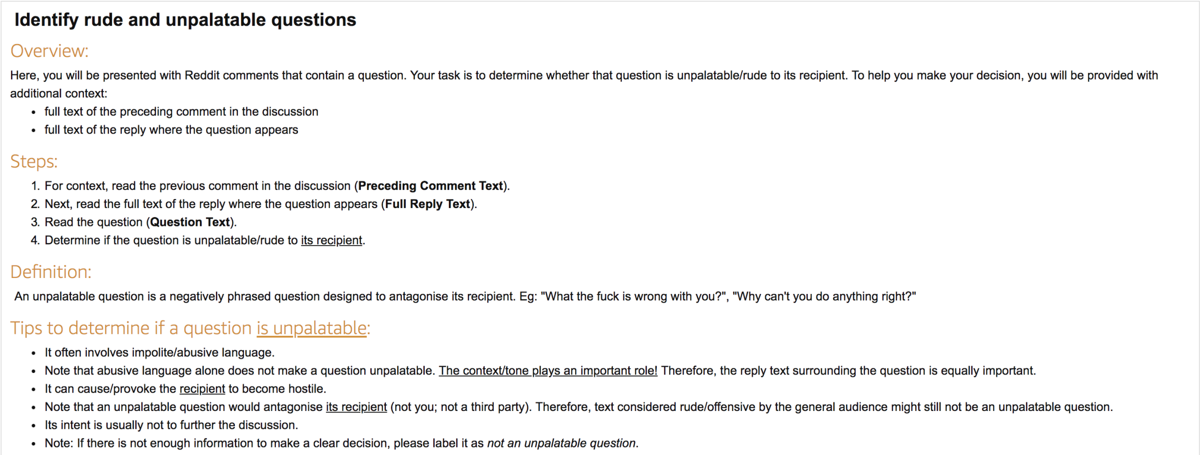
- **Hate and Toxic**:
  - r/cringepics
  - r/cringe
  - r/4chan
  - r/CringeAnarchy
  - r/KotakuInAction
  - r/ImGoingToHellForThis
  - r/TumblrInAction

B Crowdsourcing Instructions

The instructions provided to Amazon Mechanical Turk coders are shown in Figure 4 and Figure 5.

C Results

The complete list of results for the *All-Data* scenario are shown in Table 5 (deep learning models) and Table 7 (traditional machine learning models). Next, the complete list of results for the *High-Agreement-Data* scenario are shown in Table 6 (deep learning models) and Table 8 (traditional machine learning models).



Identify rude and unpalatable questions

Overview:

Here, you will be presented with Reddit comments that contain a question. Your task is to determine whether that question is unpalatable/rude to its recipient. To help you make your decision, you will be provided with additional context:

- full text of the preceding comment in the discussion
- full text of the reply where the question appears

Steps:

1. For context, read the previous comment in the discussion (**Preceding Comment Text**).
2. Next, read the full text of the reply where the question appears (**Full Reply Text**).
3. Read the question (**Question Text**).
4. Determine if the question is unpalatable/rude to its recipient.

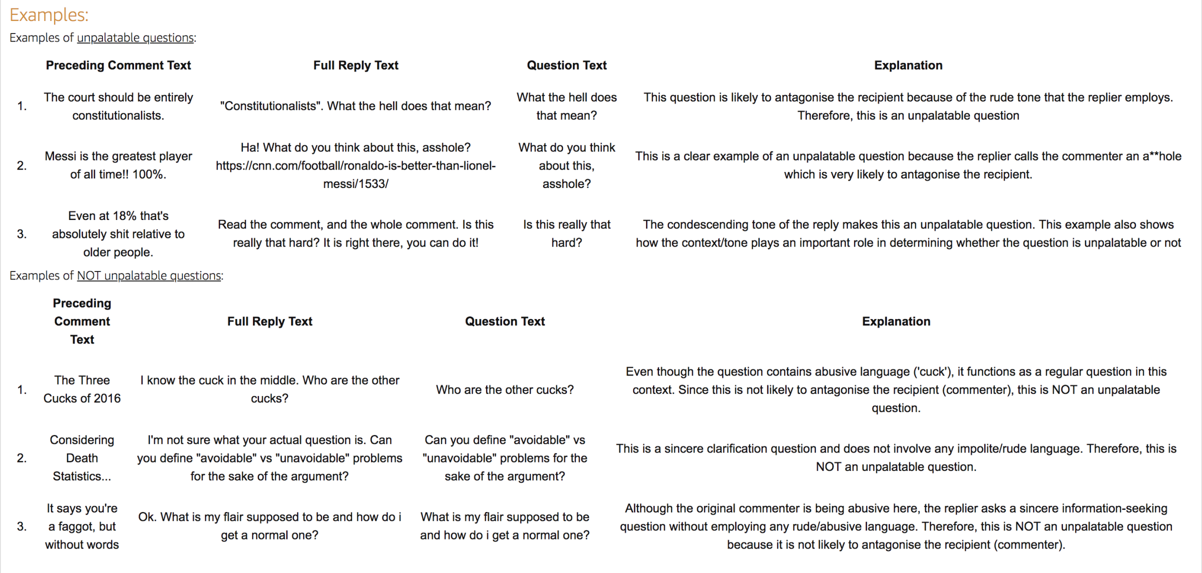
Definition:

An unpalatable question is a negatively phrased question designed to antagonise its recipient. Eg: "What the fuck is wrong with you?", "Why can't you do anything right?"

Tips to determine if a question is unpalatable:

- It often involves impolite/abusive language.
- Note that abusive language alone does not make a question unpalatable. The context/tone plays an important role! Therefore, the reply text surrounding the question is equally important.
- It can cause/provoke the recipient to become hostile.
- Note that an unpalatable question would antagonise its recipient (not you; not a third party). Therefore, text considered rude/offensive by the general audience might still not be an unpalatable question.
- Its intent is usually not to further the discussion.
- Note: If there is not enough information to make a clear decision, please label it as *not an unpalatable question*.

Figure 4: Instructions for the crowdsourcing task as seen by Mechanical Turk Workers.



Examples:

Examples of unpalatable questions:

	Preceding Comment Text	Full Reply Text	Question Text	Explanation
1.	The court should be entirely constitutionalists.	"Constitutionalists". What the hell does that mean?	What the hell does that mean?	This question is likely to antagonise the recipient because of the rude tone that the replier employs. Therefore, this is an unpalatable question
2.	Messi is the greatest player of all time!! 100%.	Ha! What do you think about this, asshole? https://cnn.com/football/ronaldo-is-better-than-lionel-messi/1533/	What do you think about this, asshole?	This is a clear example of an unpalatable question because the replier calls the commenter an a**hole which is very likely to antagonise the recipient.
3.	Even at 18% that's absolutely shit relative to older people.	Read the comment, and the whole comment. Is this really that hard? It is right there, you can do it!	Is this really that hard?	The condescending tone of the reply makes this an unpalatable question. This example also shows how the context/tone plays an important role in determining whether the question is unpalatable or not

Examples of NOT unpalatable questions:

	Preceding Comment Text	Full Reply Text	Question Text	Explanation
1.	The Three Cucks of 2016	I know the cuck in the middle. Who are the other cucks?	Who are the other cucks?	Even though the question contains abusive language ('cuck'), it functions as a regular question in this context. Since this is not likely to antagonise the recipient (commenter), this is NOT an unpalatable question.
2.	Considering Death Statistics...	I'm not sure what your actual question is. Can you define "avoidable" vs "unavoidable" problems for the sake of the argument?	Can you define "avoidable" vs "unavoidable" problems for the sake of the argument?	This is a sincere clarification question and does not involve any impolite/rude language. Therefore, this is NOT an unpalatable question.
3.	It says you're a faggot, but without words	Ok. What is my flair supposed to be and how do i get a normal one?	What is my flair supposed to be and how do i get a normal one?	Although the original commenter is being abusive here, the replier asks a sincere information-seeking question without employing any rude/abusive language. Therefore, this is NOT an unpalatable question because it is not likely to antagonise the recipient (commenter).

Figure 5: Additional examples for the crowdsourcing task as seen by Mechanical Turk Workers.

Text Input	Model	F1-score	AUROC	Weighted F1	Precision	Recall	Accuracy	AUPRC
Reply Text	Dense CNN (ELMo)	**0.532**	0.818	0.815	0.459	0.636	0.803	0.54
Reply Text + Comment Text	CNN (ELMo)	0.52	0.814	0.806	0.444	0.636	0.792	0.527
Reply Text + Comment Text	Dense CNN (ELMo)	0.516	0.815	0.816	0.476	0.578	0.809	0.531
Question Text Only	CNN (ELMo)	0.514	0.811	0.802	0.434	0.643	0.787	0.532
Reply Text	CNN (ELMo)	0.509	0.814	0.787	0.406	0.688	0.766	0.537
Reply Text + Comment Text	NLI-CNN (ELMo)	0.507	0.810	0.805	0.443	0.602	0.792	0.515
Question Text Only	Dense CNN (ELMo)	0.502	0.813	0.78	0.401	0.689	0.758	0.54
Question Text Only	Dense Bi-LSTM (ELMo)	0.5	0.809	0.778	0.394	0.692	0.755	0.517
Reply Text + Comment Text	Dense CNN (GloVe)	0.5	0.8	0.825	0.507	0.496	0.826	0.502
Question Text Only	Bi-LSTM (ELMo)	0.5	0.804	0.778	0.391	0.697	0.755	0.513
Reply Text	Stacked Bi-LSTM (ELMo)	0.497	0.803	0.782	0.396	0.67	0.76	0.501
Reply Text + Comment Text	Dense Bi-LSTM (ELMo)	0.494	0.802	0.777	0.391	0.683	0.754	0.489
Reply Text + Comment Text	CNN (GloVe)	0.491	0.802	0.826	0.517	0.471	0.829	0.51
Reply Text	Dense LSTM (ELMo)	0.49	0.794	0.768	0.378	0.702	0.742	0.489
Question Text Only	LSTM (ELMo)	0.489	0.802	0.767	0.378	0.7	0.742	0.509
Reply Text	Bi-LSTM (ELMo)	0.489	0.799	0.765	0.376	0.707	0.739	0.494
Reply Text	Dense Stacked Bi-LSTM (ELMo)	0.487	0.797	0.773	0.386	0.673	0.75	0.49
Reply Text + Comment Text	LSTM (ELMo)	0.487	0.791	0.768	0.381	0.687	0.744	0.453
Reply Text + Comment Text	Dense Stacked Bi-LSTM (ELMo)	0.484	0.791	0.783	0.407	0.626	0.766	0.468
Reply Text	Dense Bi-LSTM (ELMo)	0.484	0.803	0.771	0.398	0.661	0.75	0.503
Question Text Only	Dense Stacked Bi-LSTM (ELMo)	0.482	0.804	0.754	0.372	0.719	0.726	0.516
Reply Text + Comment Text	Bi-LSTM (ELMo)	0.481	0.801	0.757	0.366	0.714	0.729	0.493
Reply Text	LSTM (ELMo)	0.481	0.793	0.748	0.355	0.747	0.716	0.484
Question Text Only	CNN (GloVe)	0.48	0.782	0.809	0.451	0.517	0.804	0.504
Reply Text + Comment Text	Stacked Bi-LSTM (ELMo)	0.479	0.793	0.75	0.36	0.726	0.721	0.466
Question Text Only	Stacked Bi-LSTM (ELMo)	0.478	0.8	0.744	0.352	0.746	0.712	0.507
Question Text Only	Dense LSTM (ELMo)	0.477	0.804	0.74	0.349	0.76	0.707	0.514
Reply Text	CNN (GloVe)	0.473	0.792	0.817	0.488	0.466	0.818	0.498
Question Text Only	Dense CNN (GloVe)	0.472	0.765	0.788	0.416	0.562	0.775	0.479
Reply Text + Comment Text	NLI-CNN (GloVe)	0.471	0.811	0.829	0.573	0.407	0.841	0.516
Reply Text + Comment Text	Dense LSTM (ELMo)	0.468	0.783	0.748	0.357	0.698	0.72	0.448
Reply Text + Comment Text	Stacked Bi-LSTM (GloVe)	0.461	0.767	0.795	0.422	0.524	0.787	0.443
Reply Text	Dense Bi-LSTM (GloVe)	0.461	0.755	0.805	0.446	0.485	0.801	0.451
Reply Text	Dense CNN (GloVe)	0.459	0.787	0.802	0.457	0.49	0.799	0.489
Reply Text	Dense LSTM (GloVe)	0.458	0.742	0.8	0.435	0.492	0.795	0.436
Reply Text	Dense Stacked Bi-LSTM (GloVe)	0.456	0.753	0.806	0.45	0.468	0.805	0.452
Reply Text + Comment Text	Dense LSTM (GloVe)	0.455	0.752	0.796	0.419	0.504	0.789	0.423
Reply Text + Comment Text	Dense Bi-LSTM (GloVe)	0.455	0.758	0.802	0.446	0.48	0.798	0.448
Question Text Only	LSTM (GloVe)	0.452	0.736	0.776	0.394	0.555	0.761	0.435
Reply Text + Comment Text	Bi-LSTM (GloVe)	0.451	0.762	0.805	0.457	0.459	0.804	0.446
Reply Text + Comment Text	Dense Stacked Bi-LSTM (GloVe)	0.446	0.75	0.81	0.478	0.423	0.814	0.444
Question Text Only	Dense LSTM (GloVe)	0.446	0.735	0.773	0.39	0.554	0.758	0.448
Question Text Only	Dense Bi-LSTM (GloVe)	0.446	0.735	0.798	0.427	0.473	0.794	0.452
Question Text Only	Dense Stacked Bi-LSTM (GloVe)	0.442	0.738	0.746	0.352	0.617	0.72	0.446
Question Text Only	Stacked Bi-LSTM (GloVe)	0.439	0.733	0.768	0.376	0.551	0.751	0.447
Reply Text + Comment Text	LSTM (GloVe)	0.438	0.742	0.793	0.451	0.462	0.79	0.422
Question Text Only	Bi-LSTM (GloVe)	0.437	0.738	0.765	0.368	0.557	0.746	0.438
Reply Text	Stacked Bi-LSTM (GloVe)	0.435	0.749	0.79	0.438	0.471	0.785	0.435
Reply Text	LSTM (GloVe)	0.434	0.74	0.767	0.383	0.556	0.751	0.426
Reply Text	Bi-LSTM (GloVe)	0.421	0.731	0.75	0.353	0.55	0.729	0.404

Table 5: Classification results for deep learning models on the complete dataset: *confidence* $\in \{0.6, 0.8, 1.0\}$

Text Input	**Model**	**F1-score**	**AUROC**	**Weighted F1**	**Precision**	**Recall**	**Accuracy**	**AUPRC**
Question Text Only	Dense CNN (ELMo)	**0.686**	0.937	0.95	0.672	0.704	0.949	0.739
Reply Text + Comment Text	NLI-CNN (ELMo)	0.674	0.937	0.95	0.74	0.625	0.953	0.721
Reply Text	CNN (ELMo)	0.671	0.936	0.949	0.688	0.662	0.949	0.727
Question Text Only	CNN (ELMo)	0.669	0.937	0.946	0.643	0.702	0.945	0.734
Reply Text	Dense CNN (ELMo)	0.668	0.939	0.949	0.709	0.634	0.95	0.736
Reply Text + Comment Text	Dense CNN (ELMo)	0.667	0.941	0.949	0.73	0.618	0.951	0.735
Reply Text + Comment Text	CNN (ELMo)	0.665	0.941	0.948	0.695	0.655	0.949	0.737
Question Text Only	Stacked Bi-LSTM (ELMo)	0.638	0.931	0.938	0.569	0.73	0.934	0.718
Reply Text + Comment Text	Dense CNN (GloVe)	0.63	0.936	0.945	0.754	0.56	0.949	0.702
Question Text Only	Dense Bi-LSTM (ELMo)	0.628	0.931	0.937	0.568	0.715	0.934	0.714
Reply Text	CNN (GloVe)	0.626	0.932	0.945	0.754	0.538	0.949	0.711
Question Text Only	Dense CNN (GloVe)	0.625	0.926	0.944	0.705	0.572	0.946	0.695
Question Text Only	Bi-LSTM (ELMo)	0.623	0.931	0.934	0.543	0.735	0.929	0.711
Reply Text	Dense Stacked Bi-LSTM (ELMo)	0.61	0.915	0.934	0.553	0.687	0.931	0.651
Question Text Only	CNN (GloVe)	0.609	0.92	0.94	0.673	0.569	0.942	0.692
Reply Text + Comment Text	CNN (GloVe)	0.605	0.936	0.944	0.777	0.499	0.949	0.708
Reply Text + Comment Text	Bi-LSTM (GloVe)	0.603	0.9	0.942	0.712	0.525	0.946	0.644
Reply Text + Comment Text	Dense Bi-LSTM (ELMo)	0.603	0.914	0.934	0.565	0.653	0.932	0.638
Reply Text	Dense CNN (GloVe)	0.603	0.934	0.94	0.682	0.549	0.943	0.703
Reply Text + Comment Text	Dense LSTM (ELMo)	0.602	0.9	0.934	0.553	0.664	0.931	0.586
Reply Text + Comment Text	NLI-CNN (GloVe)	0.6	0.937	0.943	0.794	0.485	0.949	0.717
Reply Text + Comment Text	Stacked Bi-LSTM (GloVe)	0.597	0.908	0.939	0.657	0.556	0.941	0.638
Reply Text + Comment Text	Bi-LSTM (ELMo)	0.595	0.911	0.932	0.542	0.671	0.929	0.626
Question Text Only	Dense Stacked Bi-LSTM (ELMo)	0.595	0.931	0.925	0.505	0.763	0.916	0.712
Question Text Only	Dense Bi-LSTM (GloVe)	0.592	0.885	0.937	0.646	0.554	0.939	0.629
Reply Text + Comment Text	Dense Bi-LSTM (GloVe)	0.583	0.901	0.937	0.655	0.534	0.94	0.633
Reply Text + Comment Text	Dense Stacked Bi-LSTM (GloVe)	0.582	0.895	0.936	0.619	0.558	0.937	0.603
Reply Text	Bi-LSTM (ELMo)	0.582	0.92	0.924	0.492	0.722	0.916	0.654
Reply Text	Dense Bi-LSTM (GloVe)	0.581	0.894	0.936	0.623	0.554	0.937	0.623
Reply Text + Comment Text	Dense Stacked Bi-LSTM (ELMo)	0.58	0.902	0.932	0.569	0.601	0.931	0.601
Question Text Only	Dense LSTM (ELMo)	0.577	0.922	0.92	0.467	0.77	0.91	0.68
Reply Text	Dense LSTM (ELMo)	0.577	0.901	0.927	0.512	0.673	0.922	0.637
Reply Text	Dense Bi-LSTM (ELMo)	0.576	0.917	0.921	0.477	0.744	0.911	0.65
Question Text Only	Bi-LSTM (GloVe)	0.575	0.891	0.932	0.6	0.585	0.932	0.64
Reply Text + Comment Text	Stacked Bi-LSTM (ELMo)	0.574	0.906	0.931	0.563	0.592	0.93	0.594
Question Text Only	LSTM (GloVe)	0.573	0.871	0.936	0.65	0.521	0.939	0.612
Reply Text	Stacked Bi-LSTM (ELMo)	0.567	0.916	0.926	0.539	0.638	0.921	0.632
Reply Text + Comment Text	LSTM (ELMo)	0.566	0.888	0.928	0.539	0.62	0.925	0.555
Reply Text	Bi-LSTM (GloVe)	0.566	0.887	0.937	0.685	0.485	0.941	0.625
Reply Text	Stacked Bi-LSTM (GloVe)	0.563	0.89	0.932	0.582	0.55	0.932	0.61
Question Text Only	LSTM (ELMo)	0.561	0.925	0.915	0.444	0.768	0.903	0.684
Reply Text	Dense LSTM (GloVe)	0.557	0.872	0.932	0.613	0.528	0.933	0.59
Reply Text	LSTM (GloVe)	0.556	0.865	0.931	0.599	0.541	0.932	0.589
Reply Text + Comment Text	Dense LSTM (GloVe)	0.552	0.88	0.933	0.631	0.508	0.936	0.592
Question Text Only	Dense Stacked Bi-LSTM (GloVe)	0.552	0.87	0.925	0.568	0.589	0.921	0.615
Question Text Only	Dense LSTM (GloVe)	0.551	0.881	0.928	0.571	0.563	0.928	0.617
Reply Text + Comment Text	LSTM (GloVe)	0.545	0.88	0.933	0.639	0.488	0.937	0.593
Reply Text	LSTM (ELMo)	0.541	0.898	0.914	0.447	0.698	0.904	0.603
Reply Text	Dense Stacked Bi-LSTM (GloVe)	0.529	0.903	0.912	0.494	0.658	0.902	0.608

Table 6: Classification results for deep learning models on the high-agreement dataset: $\textit{confidence} = 1.0$

Text Input	Feature	F1-score	AUROC	Weighted F1	Precision	Recall	Accuracy	AUPRC
Reply Text	word (1, 3)	**0.443**	0.782	0.823	0.549	0.373	0.836	0.481
Question Text Only	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.432	0.764	0.81	0.471	0.4	0.816	0.443
Reply Text	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.429	0.785	0.825	0.6	0.335	0.843	0.491
Question Text Only	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.423	0.752	0.814	0.508	0.363	0.826	0.46
Question Text Only	word (1, 3)	0.422	0.764	0.815	0.519	0.357	0.828	0.46
Question Text Only	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.419	0.753	0.813	0.503	0.359	0.825	0.457
Question Text Only	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.417	0.762	0.816	0.531	0.345	0.83	0.464
Question Text Only	char (3, 5)	0.415	0.748	0.812	0.5	0.355	0.824	0.451
Reply Text	char (3, 5)	0.414	0.759	0.817	0.542	0.335	0.833	0.463
Reply Text	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.412	0.764	0.815	0.531	0.336	0.831	0.467
Reply Text + Comment Text	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.412	0.727	0.793	0.415	0.411	0.794	0.39
Reply Text	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.409	0.767	0.819	0.57	0.319	0.838	0.471
Reply Text + Comment Text	word (1, 3)	0.409	0.786	0.818	0.561	0.323	0.836	0.466
Question Text Only	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.408	0.759	0.814	0.526	0.336	0.829	0.456
Reply Text	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.407	0.744	0.809	0.487	0.349	0.821	0.437
Question Text Only	char (4, 4)	0.403	0.739	0.803	0.456	0.361	0.812	0.432
Question Text Only	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.4	0.765	0.813	0.525	0.323	0.83	0.46
Reply Text	char (4, 4)	0.4	0.742	0.807	0.482	0.342	0.82	0.434
Reply Text	char (5, 5)	0.399	0.727	0.801	0.446	0.362	0.809	0.402
Reply Text + Comment Text	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.398	0.758	0.813	0.531	0.319	0.831	0.443
Question Text Only	char (5, 5)	0.394	0.744	0.808	0.495	0.328	0.823	0.448
Reply Text + Comment Text	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.392	0.737	0.802	0.456	0.344	0.813	0.406
Reply Text + Comment Text	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.391	0.729	0.798	0.438	0.354	0.807	0.399
Question Text Only	char (3, 3)	0.39	0.704	0.784	0.385	0.395	0.783	0.357
Reply Text	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.389	0.71	0.782	0.38	0.399	0.78	0.369
Reply Text + Comment Text	Writing Style + Lexicon + Sentiment + GloVe	0.387	0.767	0.81	0.516	0.309	0.828	0.432
Reply Text + Comment Text	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.387	0.734	0.8	0.446	0.342	0.81	0.403
Reply Text + Comment Text	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.385	0.72	0.793	0.418	0.357	0.8	0.383
Question Text Only	Writing Style + Lexicon + Sentiment + GloVe	0.384	0.783	0.817	0.609	0.281	0.842	0.474
Reply Text + Comment Text	Writing Style + Lexicon + GloVe	0.384	0.768	0.809	0.515	0.307	0.828	0.434
Reply Text + Comment Text	word (1, 1)	0.384	0.73	0.8	0.449	0.336	0.811	0.398
Reply Text	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.384	0.696	0.786	0.392	0.376	0.788	0.358
Question Text Only	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.384	0.707	0.8	0.451	0.335	0.811	0.386
Reply Text	char (3, 3)	0.383	0.704	0.78	0.375	0.39	0.778	0.362
Question Text Only	Writing Style + Lexicon + GloVe	0.383	0.782	0.816	0.597	0.283	0.84	0.472
Reply Text + Comment Text	Fasttext	0.379	0.765	0.809	0.525	0.296	0.829	0.428
Reply Text	word (1, 1)	0.377	0.697	0.788	0.4	0.357	0.793	0.362
Reply Text	Writing Style + Lexicon + Sentiment + GloVe	0.377	0.785	0.813	0.571	0.282	0.837	0.465
Reply Text + Comment Text	char (5, 5)	0.377	0.718	0.796	0.436	0.332	0.807	0.389
Reply Text	Writing Style + Lexicon + GloVe	0.376	0.786	0.813	0.57	0.281	0.837	0.466
Question Text Only	word (1, 1)	0.375	0.697	0.787	0.395	0.357	0.791	0.352
Reply Text + Comment Text	char (3, 5)	0.374	0.72	0.792	0.415	0.342	0.8	0.386
Reply Text + Comment Text	char (3, 3)	0.371	0.714	0.789	0.405	0.344	0.796	0.376
Reply Text + Comment Text	char (4, 4)	0.367	0.751	0.805	0.5	0.291	0.824	0.433
Reply Text + Comment Text	RedditW2V	0.367	0.762	0.807	0.517	0.285	0.828	0.416
Question Text Only	Fasttext	0.365	0.775	0.811	0.58	0.266	0.837	0.459
Question Text Only	GloVe	0.362	0.78	0.812	0.594	0.261	0.839	0.464
Reply Text	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.361	0.727	0.808	0.544	0.271	0.832	0.425
Reply Text + Comment Text	GloVe	0.356	0.762	0.804	0.511	0.273	0.827	0.424
Reply Text	Fasttext	0.355	0.781	0.809	0.569	0.258	0.835	0.455
Reply Text + Comment Text	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.354	0.704	0.79	0.418	0.307	0.803	0.362
Question Text Only	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.348	0.712	0.804	0.526	0.261	0.829	0.4
Reply Text	RedditW2V	0.345	0.778	0.806	0.556	0.25	0.833	0.438
Question Text Only	RedditW2V	0.342	0.776	0.806	0.559	0.247	0.834	0.436
Reply Text	GloVe	0.336	0.778	0.805	0.557	0.241	0.833	0.452
Reply Text	word (2, 2)	0.326	0.651	0.79	0.43	0.262	0.809	0.321
Reply Text + Comment Text	word (2, 2)	0.324	0.68	0.785	0.403	0.271	0.801	0.338
Question Text Only	word (2, 2)	0.324	0.676	0.791	0.438	0.257	0.812	0.36
Reply Text	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.324	0.69	0.797	0.496	0.242	0.823	0.37
Reply Text + Comment Text	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.294	0.752	0.798	0.575	0.198	0.833	0.431
Reply Text + Comment Text	GoogleW2V	0.291	0.735	0.787	0.435	0.219	0.813	0.362
Question Text Only	GoogleW2V	0.279	0.752	0.792	0.511	0.192	0.826	0.394
Question Text Only	word (3, 3)	0.278	0.62	0.79	0.491	0.194	0.823	0.314
Reply Text	GoogleW2V	0.26	0.754	0.786	0.481	0.178	0.822	0.384
Reply Text	word (3, 3)	0.25	0.592	0.783	0.46	0.172	0.819	0.274

Table 7: Classification results for Logistic Regression on the complete dataset: *confidence* $\in \{0.6, 0.8, 1.0\}$

Text Input	Feature	F1-score	AUROC	Weighted F1	Precision	Recall	Accuracy	AUPRC
Question Text Only	Writing Style + Lexicon + Sentiment + GloVe	**0.618**	0.906	0.942	0.671	0.574	0.944	0.648
Question Text Only	Writing Style + Lexicon + GloVe	0.616	0.907	0.942	0.672	0.572	0.944	0.65
Question Text Only	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.614	0.916	0.944	0.753	0.521	0.949	0.672
Question Text Only	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.612	0.917	0.944	0.747	0.521	0.948	0.666
Question Text Only	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.606	0.917	0.943	0.733	0.518	0.947	0.668
Question Text Only	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.6	0.917	0.941	0.711	0.521	0.945	0.665
Question Text Only	GloVe	0.594	0.897	0.938	0.639	0.558	0.94	0.611
Question Text Only	Fasttext	0.59	0.907	0.938	0.648	0.543	0.941	0.623
Question Text Only	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.582	0.907	0.936	0.627	0.545	0.938	0.643
Question Text Only	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.581	0.905	0.938	0.663	0.521	0.941	0.629
Reply Text	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.567	0.93	0.939	0.745	0.461	0.945	0.654
Question Text Only	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.566	0.896	0.937	0.671	0.49	0.941	0.605
Reply Text	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.558	0.913	0.936	0.677	0.481	0.941	0.62
Question Text Only	word (1, 3)	0.554	0.918	0.935	0.651	0.483	0.939	0.579
Question Text Only	char (4, 4)	0.552	0.891	0.934	0.649	0.483	0.939	0.581
Reply Text	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.549	0.9	0.934	0.646	0.479	0.938	0.598
Reply Text	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.549	0.925	0.937	0.765	0.432	0.945	0.654
Reply Text	Fasttext	0.545	0.898	0.932	0.618	0.49	0.936	0.58
Reply Text	word (1, 3)	0.544	0.925	0.935	0.688	0.452	0.941	0.608
Reply Text	Writing Style + Lexicon + GloVe	0.539	0.897	0.931	0.589	0.499	0.933	0.586
Reply Text	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.539	0.926	0.936	0.771	0.419	0.944	0.651
Reply Text	Writing Style + Lexicon + Sentiment + GloVe	0.538	0.897	0.931	0.597	0.492	0.934	0.587
Reply Text + Comment Text	word (1, 3)	0.537	0.888	0.926	0.526	0.552	0.925	0.5
Reply Text + Comment Text	word (1, 1) + Sentiment + Writing-Style + Lexicon	0.533	0.915	0.934	0.702	0.433	0.941	0.597
Reply Text	char (3, 5)	0.53	0.889	0.932	0.643	0.452	0.937	0.57
Question Text Only	char (3, 5)	0.53	0.917	0.936	0.783	0.404	0.944	0.646
Reply Text	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.529	0.909	0.932	0.646	0.45	0.937	0.606
Reply Text	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.529	0.909	0.934	0.735	0.415	0.942	0.612
Question Text Only	word (1, 1)	0.527	0.888	0.93	0.602	0.47	0.933	0.56
Reply Text	GloVe	0.526	0.889	0.93	0.613	0.463	0.935	0.567
Reply Text + Comment Text	word (1, 3) + Sentiment + Writing-Style + Lexicon	0.526	0.883	0.924	0.508	0.545	0.922	0.501
Reply Text + Comment Text	Fasttext	0.525	0.876	0.924	0.512	0.541	0.923	0.516
Reply Text + Comment Text	char (5, 5) + Sentiment + Writing-Style + Lexicon	0.523	0.89	0.931	0.648	0.439	0.937	0.569
Reply Text + Comment Text	char (3, 5) + Sentiment + Writing-Style + Lexicon	0.521	0.925	0.934	0.756	0.399	0.942	0.637
Question Text Only	char (3, 3)	0.519	0.889	0.93	0.62	0.448	0.935	0.575
Reply Text + Comment Text	Writing Style + Lexicon + GloVe	0.519	0.879	0.924	0.515	0.525	0.923	0.528
Reply Text	word (1, 1)	0.517	0.898	0.931	0.658	0.43	0.938	0.578
Reply Text + Comment Text	char (3, 3) + Sentiment + Writing-Style + Lexicon	0.517	0.911	0.932	0.676	0.419	0.938	0.603
Reply Text + Comment Text	char (4, 4) + Sentiment + Writing-Style + Lexicon	0.515	0.888	0.93	0.636	0.435	0.936	0.571
Reply Text	char (3, 3)	0.512	0.881	0.926	0.565	0.468	0.93	0.549
Reply Text + Comment Text	Writing Style + Lexicon + Sentiment + GloVe	0.511	0.878	0.922	0.503	0.521	0.922	0.522
Reply Text + Comment Text	word (2, 2) + Sentiment + Writing-Style + Lexicon	0.507	0.911	0.932	0.739	0.386	0.941	0.602
Reply Text	char (5, 5)	0.506	0.881	0.928	0.61	0.435	0.934	0.549
Question Text Only	RedditW2V	0.505	0.889	0.925	0.56	0.461	0.928	0.506
Question Text Only	char (5, 5)	0.505	0.909	0.932	0.764	0.379	0.942	0.621
Reply Text	char (4, 4)	0.504	0.879	0.928	0.599	0.437	0.933	0.55
Reply Text + Comment Text	GloVe	0.504	0.872	0.922	0.505	0.507	0.922	0.516
Question Text Only	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.499	0.891	0.93	0.688	0.395	0.937	0.591
Reply Text + Comment Text	char (5, 5)	0.493	0.874	0.927	0.627	0.406	0.934	0.533
Reply Text + Comment Text	char (3, 5)	0.489	0.876	0.926	0.612	0.408	0.933	0.529
Reply Text	RedditW2V	0.488	0.877	0.924	0.556	0.441	0.928	0.483
Reply Text + Comment Text	word (1, 1)	0.487	0.885	0.927	0.616	0.404	0.933	0.518
Reply Text + Comment Text	RedditW2V	0.483	0.875	0.92	0.503	0.466	0.921	0.479
Reply Text + Comment Text	char (4, 4)	0.482	0.882	0.926	0.61	0.399	0.933	0.552
Reply Text	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.472	0.847	0.925	0.614	0.388	0.932	0.505
Reply Text + Comment Text	char (3, 3)	0.47	0.901	0.927	0.715	0.351	0.937	0.572
Question Text Only	Writing Style + Lexicon + Sentiment	0.457	0.88	0.926	0.697	0.342	0.936	0.543
Question Text Only	GoogleW2V	0.456	0.874	0.92	0.531	0.399	0.925	0.453
Question Text Only	Writing Style + Lexicon	0.446	0.87	0.925	0.72	0.324	0.937	0.518
Question Text Only	word (2, 2)	0.443	0.832	0.923	0.62	0.346	0.932	0.446
Reply Text	word (2, 2)	0.443	0.802	0.921	0.58	0.36	0.929	0.419
Reply Text + Comment Text	word (3, 3) + Sentiment + Writing-Style + Lexicon	0.439	0.864	0.92	0.569	0.36	0.927	0.476
Reply Text + Comment Text	Writing Style + Lexicon + Sentiment	0.409	0.877	0.92	0.658	0.298	0.932	0.49
Reply Text + Comment Text	word (2, 2)	0.397	0.81	0.915	0.534	0.318	0.924	0.398
Reply Text + Comment Text	GoogleW2V	0.393	0.844	0.903	0.385	0.406	0.901	0.371

Table 8: Classification results for Logistic Regression on the high-agreement dataset: *confidence* $= 1.0$

Neural-Driven Search-Based Paraphrase Generation

Betty Fabre
Orange Labs, Lannion
Univ Rennes, IRISA
betty.fabre@orange.com

Jonathan Chevelu
Univ Rennes, IRISA Lannion
jonathan.chevelu@irisa.fr

Tanguy Urvoy
Orange Labs Lannion
tanguy.urvoy@orange.com

Damien Lolive
Univ Rennes, IRISA Lannion
damien.lolive@irisa.fr

Abstract

We study a search-based paraphrase generation scheme where candidate paraphrases are generated by iterated transformations from the original sentence and evaluated in terms of syntax quality, semantic distance, and lexical distance. The semantic distance is derived from BERT, and the lexical quality is based on GPT2 perplexity. To solve this multi-objective search problem, we propose two algorithms: *Monte-Carlo Tree Search For Paraphrase Generation* (MCPG) and *Pareto Tree Search* (PTS). We provide an extensive set of experiments on 5 datasets with a rigorous reproduction and validation for several state-of-the-art paraphrase generation algorithms. These experiments show that, although being non explicitly supervised, our algorithms perform well against these baselines.

1 Introduction

Paraphrase generation, *i.e.* the transformation of a sentence into a well-formed but lexically different one while preserving its original meaning, is a fundamental task of NLP. Its ability to provide diversity and coverage finds applications in several domains like question answering (McKeown, 1979; Harabagiu and Hickl, 2006), machine-translation (Callison-Burch et al., 2006), dialog systems (Yan et al., 2016), privacy (Gröndahl and Asokan, 2019a) or adversarial learning (Iyyer et al., 2018).

The formal definition of paraphrase may vary according to the targeted application and the tolerance we set along several axes, including the *semantic distance* from the source that we want to minimize, the *quality of the syntax*, and the *lexical distance* from the source that we want to maximize to ensure diversity.

The available aligned paraphrase corpora are often biased toward specific problems like *question answering* or *image captioning*. For instance, a transformer or a seq2seq model trained on a question answering corpus will typically turn any input sentence into a question. With the lack of generic aligned datasets, it remains challenging to train generic paraphrase models in a supervised manner.

On the other hand, with the availability of large-scale non-supervised language models like BERT and GPT2, the assessment of a given candidate paraphrase in terms of *semantic distance* from its source and *lexical quality* has become much more tractable.

Leveraging from these metrics, we propose to cast the paraphrase generation task as a multi-criteria search problem. We use PPDB 2.0 (Pavlick et al., 2015), a large-scale database of rewriting rules derived from bilingual corpora, to potentially generate billions of 'naive' candidate paraphrases by edition from the source sentence. To sort the good candidates efficiently from the others, we experiment with two search algorithms. The first one, called *Monte-Carlo Paraphrase Generation* (MCPG), is a variant of the *Monte-Carlo Tree Search* algorithm (MCTS) (Kocsis and Szepesvári, 2006; Gelly and Silver, 2007; Chevelu et al., 2009). The MCTS algorithm is famous for its successes on mastering the – highly combinatorial – game of Go (Gelly and Silver, 2007; Silver et al., 2016).

The second one is a novel search algorithm that we call *Pareto Tree Search* (PTS). In contrast to MCTS which is a single-criterion search algorithm, we designed PTS to retrieve an approximation of the whole Pareto optimal set. This allows for more flexibility on paraphrase generation where the balance between *semantic distance*, *syntax quality*, and *lexical distance* is hard to tune *a priori*. Another difference between MCPG and PTS is that PTS uses a randomized breadth-first exploration policy which proves to be more efficient on this problem.

The main contribution of this article is a study on search-based paraphrase generation through the

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2100–2111
April 19 - 23, 2021.

Source sentence :	he is speaking on june 14 .
PPDB rule	**Edited sentence**
is → is found	he is found speaking on june 14 .
is speaking → 's talking	he 's talking on june 14 .
speaking → speak now	he is speak now on june 14 .
14 → 14th	he is speaking on june 14th .

Table 1: PPDB rules applied to a source sentence

sieve of three criteria: semantic similarity, syntax quality, and lexical distance. We propose and evaluate two search algorithms: MCPG and PTS. We also provide an extensive set of experiments on English datasets with a rigorous reproduction and validation methodology for several state-of-the-art paraphrase generation algorithms.

This article is organized as follows: The candidate paraphrase generation scheme is presented in Section 2. In Section 3, we develop the different criteria we use to qualify correct paraphrases. The two search algorithms (MCPG and PTS) are described in Section 4. Section 5 gives a survey on other state-of-the-art paraphrase generation algorithms. The comparisons with non-supervised and supervised baselines are presented in Section 6 where the methodology is discussed in Section 6.3. We conclude by a data-augmentation experiment in Section 6.6.

2 Paraphrase generation scheme

We model paraphrase generation as a sequence of editions and transformations from a source sentence into its paraphrase. In this work, we only consider local transformations, *i.e.* replacement of certain words or group of words by others that have the same or similar meanings, but the method we propose should work with more sophisticated transformations as well.

The *Paraphrase Database* (PPDB) (Ganitkevitch et al., 2013; Pavlick et al., 2015) is a large collection of scored paraphrase rules that was automatically constructed from various bilingual corpora using a pivot alignment method (Callison-Burch, 2008). The database is divided into increasingly large and decreasingly accurate subsets[1]. We used the XL subset, and we removed the rules labeled as "`Independent`". This left us with a set of 5.5 million rewriting rules. We give some examples of these rules in Table 1.

By iteratively applying the rules from a source sentence like the one in Table 1, we obtain a vast lattice of candidate paraphrases. Some of these candidates like "*he's* ***talking*** *on june 14*" are well-formed, but many are syntactically broken, like "*he is* ***speak now*** *on june 14*".

The number of rules that apply depends on the source sentence's size and the words it contains. For instance, on the MSRPARAPHRASE dataset (see section 6.2), sentences are quite long and the median number of PPDB-XL rules that apply is around 450. After two rewriting steps, the median number of candidates is around 10^5, and by iterative rewriting, we quickly reach a number of paraphrase candidates that is greater than 10^8.

3 Paraphrase selection criteria

As it depends on the type of text we consider that may be spoken or written, casual or formal; it is not easy to define a *universal semantic distance* or a *universal scale of well formed syntax*. However, recent advances in NLP with neural networks like BERT and GPT2 trained on huge corpora have led to the development of metrics that can act as good proxies for these ideal notions.

For the semantic distance, a quick experiment confirms that the BERT score (Zhang et al., 2019a) performs well on difficult paraphrase identification tasks. The BERT score is an F1-measure over an alignment of the BERT contextual word embeddings of each of the sentences. To assess the sensitivity of this score, we computed the Area Under the ROC curve (AUC) on QQP and PAWS, two difficult paraphrase identification corpora (Kornél Csernai, 2017; Zhang et al., 2019b). On QQP, we obtained 75.2% and on PAWS, we obtained 67.0%. The PAWS corpus being designed to trick paraphrase identification classifiers, 67.0% is a reasonable performance. We hence opted for the BERT score between the source sentence and paraphrase candidate (denoted BERT$_{S}$) as our semantic score.

Regarding the syntax quality, the perplexity of GPT2 (Radford et al., 2019) is a good ranking criterion. Although, as illustrated in Table 2, in some cases, a rule-based spell-checker may detect errors that GPT2 would miss (but the reverse is also true). We hence opted for GPT2 as a primary criterion for syntax quality, combined with the LANGUAGETOOL spell-checker (Naber, 2003) that we only used on a second stage for performance reasons.

The lexical distance is important to ensure the

---

[1] PPDB is available on `http://paraphrase.org`, and a python interface is available here `https://github.com/erickrf/ppdb`

diversity of the produced paraphrases. It is however simple to handle. Some authors use the BLEU surface metric (Miao et al., 2018a), we opted here for the normalized character-level Levenshtein edition distance.

The balance between these criteria is difficult to obtain. Table 2 illustrates their impact on a sentence taken from the train set of MSRPARAPHRASE. The candidate examples in this table underline the tough dilemma between maximizing the semantic similarity (safe and conservative policy) and maximizing the lexical distance (risk-prone). The third and fourth examples underline the utility of the spell-checker: some low-perplexity examples are ill-formed. On the second part of the table, we printed the sentences chosen by our two models: MCPG and PTS.

4 Searching algorithms

Searching for good paraphrases in the large lattice of candidates generated by PPDB is a costly task. We propose two algorithms that share a similar structure: an outer loop explores the lattice at different depths, while an inner loop explores the candidates at each depth. Both algorithms are anytime: they return the best solutions found so far when the time or space budget is depleted.

4.1 MCPG: Monte-Carlo tree search for Paraphrase Generation

Following the idea of Chevelu et al. (2009), we used *Monte-Carlo Tree Search* (MCTS) to explore the PPDB lattice. The three key ingredients of MCTS are: a *bandit* policy at each node of a search-tree to select the most promising paths, randomized roll-outs to estimate the quality of these paths, and back-propagation of rewards along the paths to update the bandit. We opted here for a randomized bandit policy called EXP3 (Auer et al., 2002). The MCTS algorithm being not designed for multi-objective problems, we needed to combine semantic similarity BERT_S, syntax correctness GPT2 and surface diversity Lev_S into a single criterion. We opted for the following polynomial:

$$\alpha \cdot \text{BERT}_\text{S} + \beta \cdot \text{Lev}_\text{S} \cdot \text{BERT}_\text{S} - \gamma \cdot \text{GPT2} \quad (1)$$

where the product $\text{Lev}_\text{S} \cdot \text{BERT}_\text{S}$ is intended to avoid a trivial maximization of the score by applying a lot of editions to the source sentence. After a few experiments on train sets, we tuned empirically the weights to $\alpha = 3, \beta = 0.5$ and $\gamma = 0.025$ in order to obtain a balance as the one described in Table 2.

4.2 PTS: Pareto Tree Search in the paraphrase lattice

We observed two drawbacks for MCTS.

First, it was designed for combinatorial problems like Go where the evaluation is only possible on the leaves of the search tree. This is not the case for paraphrase generation where the neural models can evaluate any rewriting step and where rewriting from good candidates is more likely to provide good paraphrases than rewriting from bad ones. Secondly, it has been designed for single criterion search which requires fixing the balance between criteria definitively before any paraphrase search begins. This is not very flexible, and it becomes painful when we want to generate sets of candidates.

By plotting the distributions of the scores like on Figure 1, we noticed that most of the candidates were dominated in the Pareto sense: it was possible to eliminate most of the candidates without any hyper-parameter tuning. Hence, we adapted MCPG to explore the paraphrase lattice and recover an approximation of the Pareto front, postponing the balance between criteria as a quick post-optimization stage. This led us to the PTS algorithm described as pseudo-code in Table 3.

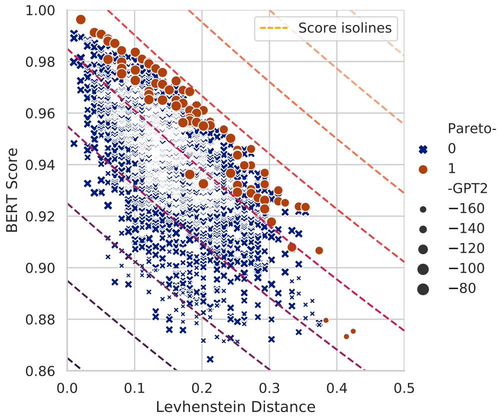


Figure 1: The cloud of candidates generated from the sample of Table 2. The optima of any positive combination of BERT score, normalized Levenshtein distance, and GPT2 perplexity belong to the Pareto front (orange dots). We plotted the projections of MCPG combined score (1) with dashed isolines. The BERT score and Levenshtein distance being clearly anti-correlated, the balance between these two criteria is difficult to tune.

Source	the agency could not say when the tape was made , though the voice says he is speaking on june 14 .

Policy	Candidate Sentence	BERT_S	GPT2	Lev_S	Errs
high BERT conservative	the agency could not say when the tape was made , though the voice says he is talking on june 14 .	**0.99**	4.23	0.04	0
high Lev risk prone	the organizations and agencies could just 'm saying whenever the tape-based was provided presentation there are , however the express opinions informed that he was found talking pertaining end-june fourteen .	0.60	7.83	**1.0**	0
high GPT2 bad syntax	the agencies was not able say when the tape been given currently undertaking though the voice indicated he talking on june 14 .	0.82	**7.10**	0.55	2
spell & grammar errors	the organisation are incapable of 're saying when the tape is set out , despite the fact that the voice just said he have a conversation on june 14 .	0.73	5.49	0.79	**2**
balanced	the organization could not say when the tape was made , although the voice indicates that he is talking on june 14 .	0.95	4.34	0.28	0
MCPG output	the organization could not say when the tape was made , though the voice indicates that he is talking on june 14 .	0.96	4.36	0.26	0
PTS output	the agency could not say when the tape was made , although the voice says he is speaking on june 14 .	1.00	4.17	0.02	0
Target	the agency could put no exact date on the tape , though the voice says he is speaking on june 14 .	0.87	4.77	0.22	0

Table 2: An example of a source sentence sampled from the MSRPARAPHRASE train set with some representative candidates from its PPDB rewriting graph. For each of the candidates, we computed the BERT score with respect to the source (BERT_S), the normalized GPT2 perplexity (GPT2), the Levenshtein distance from the source sentence (Lev_S), and the number of spell and grammar errors detected (Errs). The PPDB editions are highlighted in green. The detected spell and grammar errors are highlighted in purple. The first candidate maximizes the semantic similarity (BERT score) and is very conservative. The second one maximizes the surface diversity (Lev_S) and takes a lot of risks. The third one shows an ill-formed paraphrase candidate. The fourth candidate emphasizes the utility of the spell-checker. The last candidate, on the fifth row, achieves our equilibrium goal. We show on a second part, the paraphrases generated by our models MCPG an PTS, and on a third part, we give the reference paraphrase from the dataset.

```
function PTS(input_sentence)
    candidates ← REWRITE(input_sentence)
    depth ← 1
    while time/space $<$ budget do
        while time/space $<$ layer-budget do
            batch ← SAMPLE(candidates)
            scored ← scored $\cup$ NN.SCORES(batch)
        end while
        layer_front_set ← PARETO-FRONT(scored)
        candidates ← REWRITE(layer_front_set)
        depth ← depth + 1
    end while
    return PARETO-FRONT(all scored nodes)
end function
```

Table 3: **Pareto Tree Search** (PTS) algorithm

5 Related work

Rule-based and statistical approaches Following the path of machine translation, the paraphrase generation literature first evolved from laboriously handcrafted linguistic rules (McKeown, 1979; Meteer and Shaked, 1988; Chandrasekar and Srinivas, 1997; Carroll et al., 1999) to more automatic and data-driven rules extraction methods (Callison-Burch et al., 2006; Madnani and Dorr, 2010). Like in machine translation, phrase-level substitution rules can be extracted by sub-sentence alignment algorithms from parallel corpora (Brown et al., 1993). Building such a dedicated corpus being a long and costly task, one usually transforms other corpora through a "pivot representation" (Barzilay and McKeown, 2001; Callison-Burch et al., 2006; Ganitkevitch et al., 2013; Chen et al., 2015). The weakness of these approaches is that phrase-level rewriting rules alone are not able to build coherent sentences. A typical data-driven paraphrase generator used to be a mixture of potentially noisy handcrafted and data-driven rewriting rules coupled with a score that had to be optimized in real-time through dynamic programming. However, dynamic programming methods like *Viterbi* are constrained by the requirement of a score that decomposes into a sum of word-level or phrase-level criteria (Xu et al., 2016). Some attempts were made to relax this constraint with search-based approaches (Chevelu et al., 2009; Daumé et al., 2009), but the global optimized criteria were simplistic, and the obtained solutions were not suitable for practical deployment.

Supervised encoder-decoder approaches Like machine translation, paraphrase generation benefited from deep neural networks and evolved to efficient end-to-end architectures that can both learn to align and translate (Bahdanau et al., 2016; Vaswani et al., 2017). Several papers like (Prakash et al., 2016a; Cao et al., 2017) set the paraphrase generation task as a supervised sequence-to-sequence problem. As confirmed by our experiments in Section 6.4, this approach is efficient for specific types of paraphrases. It is also able to produce relatively long-range transformations, but it requires huge and high-quality sentence-level aligned datasets for training.

The paraphrase generation literature mostly reports results on MSCOCO (Chen et al., 2015) and QQP (Kornél Csernai, 2017) datasets which are built respectively from image captions and semi-duplicate questions. These datasets are very specific: MSCOCO is strongly biased toward image description sentences, and QQP is dedicated to questions. The Transformer model we trained on QQP typically transforms any input into a question. For instance, from "He is speaking on june 14." it gives "Who is speaking on june 14?".

Generative and hybrid approaches Recent approaches rely on conditional generative models like CVAE to allow the generation of paraphrase sets and palliate the lack of supervision data (Gupta et al., 2018; Yang et al., 2019a; Roy and Grangier, 2019; An and Liu, 2019). Others make use of CGANs combined with reinforcement learning (Li et al., 2018; Wang and Lee, 2018). Witteveen and Andrews (2019) propose to simply fine-tune a large non-supervised model like GPT2.

Recent text-generation architectures like the one proposed in (Moryossef et al., 2019; Fu et al., 2019) make a clear separation between a rule-based planning phase and the neural realization. A similar idea was tested by Huang et al. (2019) where PPDB rules are used to control the decoder of a CVAE.

Search-based approaches Search-based methods regained interest in the text generation community for several reasons, including the need for flexibility and the fact that with deep neural-networks, the search evaluation criteria have become more reliable. These methods are often slower than auto-regressive text generation methods, but it is always possible to distillate the models into faster ones like we do in Section 6.6. In (Gröndahl and Asokan, 2019b) they deployed a search-based policy for style imitation, Schwartz and Wolter (2018) and Kumagai et al. (2016) both used MCTS for text generation. Following the same trend, Miao et al. (2018a) proposed to use Metropolis-Hasting sampling (Metropolis et al., 2004) for constrained sentence generation in an algorithm called CGMH.

Starting from the source sentence, the CGMH algorithm samples a sequence of sentences by using local editions: word replacement, deletion, and insertion. For paraphrase generation, CGMH constraints the sentence generation using a matching function that combines a measure of semantic similarity and a measure of English fluency. This model is therefore directly comparable with our MCPG and PTS approaches.

6 Experiments

The evaluation metrics and datasets are described respectively in Section 6.1 and 6.2. We paid attention to set up a rigorous validation protocol for our experiments. The reproducibility and methodology issues that we faced are discussed in Section 6.3. We compare our model with state-of-the-art supervised methods and with CGMH, another search-based algorithm. The technical details on these algorithms are developed in Section 6.4. The results are detailed in Section 6.5.

6.1 Evaluation metrics

We rely on standard machine translation metrics that compare the generated paraphrase to one or several ground-truth references (Olive et al., 2011). We report surface metrics BLEU and TER and semantic metrics METEOR and BERT[2], the average BERT score of the generated sentence with respect to the reference sentences[3].

6.2 Evaluation datasets

Table 4 gives a summary of the datasets we used. On one hand, we have large datasets like MSCOCO or OPUSPARCUS (OPUSPAR.) that are noisy and very specific. On the other hand, we have MSR-PARAPHRASE (MSRPARA.), a high-quality but small human-labeled dataset.

---

[2] This is the same metric, but this usage of BERT score against **reference** must not be confused with the BERT score against **source** ($BERT_S$) that we use in (1).

[3] We used the scripts from `github.com/jhclark/multeval` and `github.com/Tiiiger/bert_score`. For the BERT score we used 'bert-base-uncased_L8_no-idf_version=0.1.2'

Corpus	**Size**	**Len**	**B>0.75**	**L<0.25**	**PPDB@1**	**PPDB@3**
MSCOCO	$2.4 \cdot 10^6$	11	**13.6%**	0.6%	119	$7.4 \cdot 10^4$
MSRPARA.	$7.8 \cdot 10^3$	23	81.8%	18.3%	454	$\mathbf{7.0 \cdot 10^6}$
OPUSPAR.	$4.2 \cdot 10^7$	6	36.0%	0.5%	113	$1.5 \cdot 10^4$
PAWS	$3.6 \cdot 10^5$	20	**100%**	**65.6%**	289	$1.3 \cdot 10^6$
QQP	$1.5 \cdot 10^5$	10	64.1%	21.8%	141	$1.3 \cdot 10^5$

Table 4: **Datasets statistics**. 'Size' is the number of instances. 'Len' is the median number of words per sentence. We also report a rough distribution of the BERT score (B) and the Levenshtein distance (L) computed on the corpora paraphrases pairs. The 'PPDB@x' columns give the median number of candidates that PPDB-XL generates from one sentence respectively at one and three rewriting steps. The size of the PPDB rewriting lattice is correlated to the length of sentences.

Model applied on MSCOCO	$BLEU \uparrow$
RESIDUAL LSTM (Fu et al., 2020)	23.7
LBOW-TOPK (Fu et al., 2020)	25.3
RESIDUAL LSTM (Prakash et al., 2016a)	37.0
VAE-SVG-EQ (Gupta et al., 2017)	39.6
TRANSFORMER (Egonmwan and Chali, 2019b)	41.8
TRANSSEQ (Egonmwan and Chali, 2019b)	44.5

Model applied on QQP	$BLEU \uparrow$
RESIDUAL LSTM (Fu et al., 2020)	24.9
CGMH (Miao et al., 2018a) - *weakly-supervised*	18.8
LBOW-TOPK (Fu et al., 2020)	26.2
VAE-SVG-EQ (Gupta et al., 2017)	37.1
TRANSFORMER (Egonmwan and Chali, 2019b)	39.0
TRANSSEQ (Egonmwan and Chali, 2019b)	39.8

Table 5: Inconsistent BLEU scores as reported in several articles on paraphrase generation.

The MSCOCO-2017 set (Chen et al., 2015) contains image captions, assuming that captions associated with the same picture are paraphrases. The strengths of MSCOCO are its size and the fact that each source sentence is associated with four reference paraphrases. However, the sentences are biased towards a descriptive style, and the quality of the paraphrases is often questionable.

The OPUSPARCUS dataset (Creutz, 2018) has been extracted from movies and TV shows subtitles. It contains mostly informal dialogues.

Quora Question Pairs dataset (QQP) (Kornél Csernai, 2017) is a paraphrase identification corpus dedicated to question-answering systems. The MSRPARAPHRASE dataset (Dolan and Brockett, 2005) is mostly build with pieces of news. The sentences of this corpus are quite long. It is a small but high-quality paraphrase identification corpus that was labeled by humans. PAWS$_{wiki}$ (PAWS) (Zhang et al., 2019b; Yang et al., 2019b) is a paraphrase identification corpus that contains several lexically-similar but hard-to-classify pairs like "*Flights to Florida from New York*" and "*Flights from Florida to New York*".

6.3 Methodology and reproducibility issues

In the paraphrase generation literature, most of the papers report results on MSCOCO and QQP corpora. In table 5, we provide the BLEU scores as reported in (Prakash et al., 2016b; Gupta et al., 2017; Fu et al., 2019; Miao et al., 2018b) and (Egonmwan and Chali, 2019a). However, even if the dataset names coincide, and even if each evaluation methodology is correct on its own, the discrepancies between methodologies render these values impossible to compare with each other.

The strange gap between the residual LSTM performance of (Prakash et al., 2016b) and the one reported in (Fu et al., 2019) can be explained by the fact that the first one is using the 2014 version of MSCOCO while the other is using 2017 version. But we also found several other issues: differences in test sets splits, different strategies for sentence length shrinking (or pruning), different vocabulary size, and tokenization strategies.

Regarding the sentence lengths, Prakash et al. (2016b) and Gupta et al. (2017) shrunk all sentences to 15 words. Fu et al. (2019) set the maximum length to 16 while Egonmwan and Chali (2019a) set it to 15 and 10 respectively for the input and target sentences. Knowing that roughly 56% (resp. 5%) of MSCOCO target sentences are strictly longer than 10 (resp. 15) words, these small changes can have a great impact on the results. The vocabulary considered also differs. Fu et al. (2019) used a vocabulary of 8k and 11k tokens from the train sets of QQP and MSCOCO respectively, whereas Egonmwan and Chali (2019a) had a vocabulary of approximately 15k words that was constructed on both the train and test sets.

The scripts used to compute metrics did also differ from one paper to another, and it is known that BLEU scores can vary wildly with different parameterizations (Post, 2018).

We used the code when available and otherwise, we tried as much as possible to reproduce the models of the literature faithfully. This allowed us to have the exact same preprocessing, training and testing pipeline for all our experiments. As a side effect, it gives a grounded benchmark between the methods that we could test.

6.4 Baseline systems implementation

In the next subsections, we present the re-implemented encoder-decoder neural networks architectures and the weakly-supervised paraphrase generator used as baselines in our experiments.

Supervised paraphrase generators As supervised baselines, we trained three neural network architectures that were previously reported to achieve good results on MSCOCO and QQP, in particular, the Seq2Seq architecture, a *Residual* LSTM architecture (Prakash et al., 2016b) and a TRANSFORMER model (Egonmwan and Chali, 2019a). We extended the experiments to the other aligned corpora: MSRPARAPHRASE, OPUSPARCUS and PAWS.

To be more precise, we trained a 4-layers LSTM Seq2Seq with a bidirectional encoder and decoder using attention. This architecture is reported as SEQ2SEQ in the results. We trained a 4-layer Residual LSTM Seq2Seq as introduced by Prakash et al. (2016b) and reproduced by Fu et al. (2019). This architecture is reported as RESIDUAL LSTM in the results. The results we obtained with this model are close to the ones reported by Fu et al. (2019).

Finally, we trained a TRANSFORMER using the *transformer_base* hyper-parameters set from (Vaswani et al., 2017). This architecture is reported as TRANSFORMER BASE in the results.

For all the encoder-decoder experiments, we used the *fairseq* framework (Ott et al., 2019) that implements the SEQ2SEQ and TRANSFORMER architectures. We added our own implementation of the RESIDUAL LSTM architecture.

For preprocessing, we used Moses tokenizer and subword segmentation following Sennrich et al. (2016b) and using the subword-nmt library . The maximum sentence length is set to 1024 tokens which the default setting in *fairseq*. For decoding, we did a beam search with a beam of size 5.

Weakly-supervised paraphrase generator For the weakly-supervised strategy CGMH introduced by Miao et al. (2018b) we used the official code. We managed to reproduce their results on QQP. On our test set we achieve a BLEU score of 22.5 while they reported 18.8. We then extended the experiment to other datasets and metrics.

6.5 Results

Table 6 summarizes the results of the comparison of our models, supervised encoder-decoder neural networks and the weakly-supervised method CGMH. Overall, these results are mixed: it is however important to keep in mind that contrary to the supervised baselines which are retrained for each dataset, the parameters of the CGMH, MCPG and PTS models are left unchanged.

On the MSCOCO and QQP datasets, the supervised baselines achieve clearly better results, but MCPG and PTS achieve better results on OPUSPARCUS and PAWS except with the BERT score for which the TRANSFORMER model achieves similar results. On MSRPARAPHRASE, the encoder-decoder neural networks models perform poorly. This result can be explained by the small number of training examples available on this corpus (See Table 4). On the weakly-supervised side, MCPG and PTS models outperform the CGMH baseline on all corpora except on the MSCOCO dataset where the results are similar.

These results prove that even without a specialized training sets, generic search-based methods are competitive for paraphrase generation. However, it is a fact that encoder-decoder networks have excellent performances for text generation and have the potential to generate more complex paraphrases than those obtained by simple local transformations as in our models.

Training a general – all-purpose – paraphrase generation network would require a huge volume of data. And there is yet much less aligned corpora available for paraphrase than for translation.

6.6 Data-augmentation experiment

In order to get the best of both worlds, one option is to enrich the training set of a TRANSFORMER with the results of a search-based method.

To test this idea, we used our models to augment the MSRPARAPHRASE training set. For that purpose we created new pairs of paraphrases from unused sentences of MSRPARAPHRASE (the pairs labeled as “not paraphrases”) using MCPG and PTS. We then trained a new supervised TRANSFORMER models on the augmented training sets.

Having no guarantee that our models generate syntactically perfect sentences, we inverted the pairs, thus taking the generated paraphrases as input and the dataset’s sentences as output. This trick, called *back-translation*, forces the target model to generate correct sentences (Sennrich et al., 2016a; Edunov et al., 2018).

We report the results of this experiment in Table 7. The models trained with the augmented

Corpus	**Model**	*BLEU* ↑	*TER* ↓	*METEOR* ↑	*BERT* ↑
MSCOCO	SEQ2SEQ	**27.5**	**62.3**	**24.3**	0.76
	RESIDUAL LSTM	**26.9**	63.3	**24.2**	0.76
	TRANSFORMER BASE	**26.9**	63.3	**24.2**	0.76
	CGMH	17.3	72.6	21.9	0.7
	MCPG	16.5	73.5	23.2	0.71
	PTS	17.0	69.9	22.8	0.64
QQP	SEQ2SEQ	**29.2**	60.3	30.7	0.8
	RESIDUAL LSTM	**28.4**	**59.1**	30.2	0.8
	TRANSFORMER BASE	**29.1**	**59.5**	30.5	0.8
	CGMH	22.5	65.0	27.0	0.72
	MCPG	24.1	64.5	**31.8**	0.78
	PTS	25.6	**58.7**	31.4	0.78
OPUSPARCUS	SEQ2SEQ	8.4	79.3	13.8	0.69
	RESIDUAL LSTM	8.1	78.6	14.3	0.7
	TRANSFORMER BASE	8.1	84.3	13.9	0.7
	CGMH	7.6	78.9	16.8	0.58
	MCPG	**9.6**	78.6	**23.3**	0.67
	PTS	**9.1**	**70.2**	22.1	0.66
PAWS	SEQ2SEQ	44.0	36.8	39.2	**0.92**
	RESIDUAL LSTM	43.6	37.1	38.9	**0.92**
	TRANSFORMER BASE	42.4	37.6	38.9	**0.92**
	CGMH	15.4	58.1	20.7	0.61
	MCPG	55.5	24.3	**49.2**	**0.93**
	PTS	**57.9**	**21.9**	**48.5**	**0.92**
MSRPARAPHRASE	SEQ2SEQ	11.6	89.5	12.6	0.53
	RESIDUAL LSTM	10.5	93.7	11.2	0.52
	TRANSFORMER BASE	20.7	76.7	21.6	0.65
	CGMH	9.7	72.9	15.4	0.48
	MCPG	**39.3**	52.4	**37.2**	**0.81**
	PTS	**40.3**	**48.4**	36.1	**0.80**

Table 6: **Experiments summary.** Symbol '↑' means that higher value is better. Significantly best values are marked in bold. MCPG and PTS outperform state-of-the-art models on OPUSPAR., PAWS and MSRPARA.. Contrary to the supervised baselines, the parameters of the MCPG and PTS models are left unchanged for each dataset.

Train set	*BLEU*	*TER*	*METEOR*	*BERT*
ORIG.	20.7	76.7	21.6	**0.65**
ORIG. + MCPG	**25.3**	**69.6**	24.5	0.63
ORIG. + PTS	25.3	71.1	**24.6**	0.63

Table 7: **Data-augmentation experiment summary**. We trained a TRANSFORMER base model on three versions of the MSRPARAPHRASE set: the original train set (ORIG) and the original train set extended by paraphrasing other sentences from the same distribution with the MCPG (ORIG + MCPG) and PTS (ORIG + PTS) models.

training sets achieved a significant performance gain on BLEU, TER and METEOR.

7 Conclusion

We experimented with two search-based approaches for paraphrase generation. These approaches are pragmatic and flexible. Being generic, our approaches did not overfit on small datasets. We performed extensive experiments with a rigorous evaluation methodology that we applied both on our algorithms and on the other related methods that we tried to reproduce faithfully. These experiments confirm that our two methods, namely MCPG and PTS, are comparable to supervised state-of-the-art baselines despite being less tightly supervised. When compared with CGMH, another search-based and weakly-supervised method, our algorithms proved to be faster and more efficient.

We plan to refine the scoring with deep reinforcement learning techniques and enrich the edition rules with more sophisticated patterns like phrase permutations. Our search algorithms remain slow for a real-time deployment: the current versions are better suited as an offline model for data augmentation. The experiment of Section 6.6 confirms that this application of search-based methods is a promising research avenue. A planning-then-realization hybridization like the one proposed by Moryossef et al. (2019) and Fu et al. (2019) could also be considered for further works.

Acknowledgment

We want to thank the reviewers for their constructive comments. This work was funded by the ANR Cifre convention N°2018/1559 and Orange-Labs Research.

References

Zhecheng An and Sicong Liu. 2019. Towards Diverse Paraphrase Generation Using Multi-Class Wasserstein GAN. *CoRR*, abs/1909.13827.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal on Computing*, 32(1):48–77.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*. ArXiv: 1409.0473.

Regina Barzilay and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*, pages 50–57, Toulouse, France. Association for Computational Linguistics.

Peter E Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):50.

Chris Callison-Burch. 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 196–205, Honolulu, Hawaii. Association for Computational Linguistics.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24, New York City, USA. Association for Computational Linguistics.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint Copying and Restricted Generation for Paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3152–3158.

John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 269–270, Bergen, Norway. Association for Computational Linguistics.

R. Chandrasekar and B. Srinivas. 1997. Automatic induction of rules for text simplification. *Knowledge-Based Systems*, 10(3):183–190.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv:1504.00325 [cs]*. ArXiv: 1504.00325.

Jonathan Chevelu, Thomas Lavergne, Yves Lepage, and Thierry Moudenc. 2009. Introduction of a new paraphrase generation tool based on Monte-Carlo sampling. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 249–252, Suntec, Singapore. Association for Computational Linguistics.

Mathias Creutz. 2018. Open Subtitles Paraphrase Corpus for Six Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

Bill Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding Back-Translation at Scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019a. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255, Hong Kong. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019b. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255, Hong Kong. Association for Computational Linguistics.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase Generation with Latent Bag of Words. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13645–13656. Curran Associates, Inc.

Yao Fu, Yansong Feng, and John P. Cunningham. 2020. Paraphrase Generation with Latent Bag of Words. *arXiv:2001.01941 [cs]*. ArXiv: 2001.01941.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Sylvain Gelly and David Silver. 2007. Combining online and offline knowledge in uct. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 273–280, New York, NY, USA. Association for Computing Machinery.

Tommi Gröndahl and N. Asokan. 2019a. Effective writing style imitation via combinatorial paraphrasing. *CoRR*, abs/1905.13464.

Tommi Gröndahl and N. Asokan. 2019b. Effective writing style imitation via combinatorial paraphrasing. *CoRR*, abs/1905.13464.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2017. A Deep Generative Framework for Paraphrase Generation. *arXiv:1709.05074 [cs]*. ArXiv: 1709.05074.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A Deep Generative Framework for Paraphrase Generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for Using Textual Entailment in Open-Domain Question Answering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 905–912, Sydney, Australia. Association for Computational Linguistics.

Shaohan Huang, Yu Wu, Furu Wei, and Zhongzhi Luan. 2019. Dictionary-Guided Editing Networks for Paraphrase Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6546–6553.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. *arXiv:1804.06059 [cs]*. ArXiv: 1804.06059.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit Based Monte-carlo Planning. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 282–293, Berlin, Heidelberg. Springer-Verlag. Event-place: Berlin, Germany.

Kornél Csernai. 2017. First Quora Dataset Release: Question Pairs - Data @ Quora - Quora.

Kaori Kumagai, Ichiro Kobayashi, Daichi Mochihashi, Hideki Asoh, Tomoaki Nakamura, and Takayuki Nagai. 2016. Human-like Natural Language Generation Using Monte Carlo Tree Search. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*, pages 11–18, Edinburgh, UK. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase Generation with Deep Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Nitin Madnani and Bonnie J. Dorr. 2010. Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods. *Computational Linguistics*, 36(3):341–387.

Kathleen R. McKeown. 1979. Paraphrasing Using Given and New Information in a Question-Answer System. In *17th Annual Meeting of the Association for Computational Linguistics*, pages 67–72, La Jolla, California, USA. Association for Computational Linguistics.

Marie Meteer and Varda Shaked. 1988. STRATEGIES FOR EFFECTIVE PARAPHRASING. In *Coling Budapest 1988 Volume 2: International Conference on Computational Linguistics*.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 2004. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087. Publisher: American Institute of PhysicsAIP.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018a. CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. *arXiv:1811.10996 [cs, math, stat]*. ArXiv: 1811.10996.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018b. CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. *arXiv:1811.10996 [cs, math, stat]*. ArXiv: 1811.10996.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-Step: Separating Planning from Realization in Neural Data-to-Text Generation. *arXiv:1904.03396 [cs]*. ArXiv: 1904.03396.

Daniel Naber. 2003. LanguageTool: A Rule-Based Style and Grammar Checker.

Joseph Olive, Caitlin Christianson, and John McCary, editors. 2011. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*. Springer-Verlag, New York.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational*

Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 425–430, Beijing, China. Association for Computational Linguistics.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016a. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016b. Neural Paraphrase Generation with Stacked Residual LSTM Networks. *COLING*, page 12.

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

Aurko Roy and David Grangier. 2019. Unsupervised Paraphrasing without Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6033–6039, Florence, Italy. Association for Computational Linguistics.

Tobias Schwartz and Diedrich Wolter. 2018. A Variant of Monte-Carlo Tree Search for Referring Expression Generation. In *KI 2018: Advances in Artificial Intelligence*, pages 315–326. Springer, Cham.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *arXiv:1706.03762 [cs]*. ArXiv: 1706.03762.

Yaushian Wang and Hung-Yi Lee. 2018. Learning to Encode Text as Human-Readable Summaries using Generative Adversarial Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4187–4195, Brussels, Belgium. Association for Computational Linguistics.

Sam Witteveen and Martin Andrews. 2019. Paraphrasing with Large Language Models. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 215–220, Hong Kong. Association for Computational Linguistics.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing Statistical Machine Translation for Text Simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. 2016. DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 516–525, Berlin, Germany. Association for Computational Linguistics.

Qian Yang, Dinghan Shen, Yong Cheng, Wenlin Wang, Guoyin Wang, Lawrence Carin, et al. 2019a. An end-to-end generative architecture for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3123–3133.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019b. PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3678–3683, Hong Kong, China. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019a. BERTScore: Evaluating Text Generation with BERT. *arXiv:1904.09675 [cs]*. ArXiv: 1904.09675.

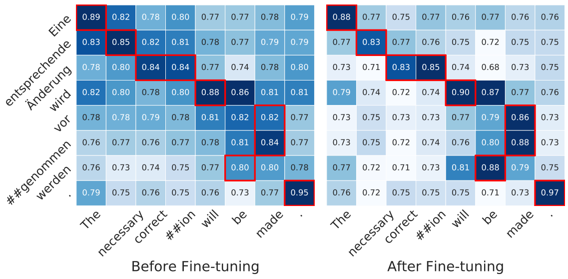
Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proceedings of the 2019 Conference of* *the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308,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

Word Alignment by Fine-tuning Embeddings on Parallel Corpora

Zi-Yi Dou, Graham Neubig
Language Technologies Institute, Carnegie Mellon University
{zdou,gneubig}@cs.cmu.edu

Abstract

Word alignment over parallel corpora has a wide variety of applications, including learning translation lexicons, cross-lingual transfer of language processing tools, and automatic evaluation or analysis of translation outputs. The great majority of past work on word alignment has worked by performing unsupervised learning on parallel text. Recently, however, other work has demonstrated that pre-trained contextualized word embeddings derived from multilingually trained language models (LMs) prove an attractive alternative, achieving competitive results on the word alignment task even in the absence of explicit training on parallel data. In this paper, we examine methods to marry the two approaches: leveraging pre-trained LMs but fine-tuning them on parallel text with objectives designed to improve alignment quality, and proposing methods to effectively extract alignments from these fine-tuned models. We perform experiments on five language pairs and demonstrate that our model can consistently outperform previous state-of-the-art models of all varieties. In addition, we demonstrate that we are able to train multilingual word aligners that can obtain robust performance on different language pairs. Our aligner, **AWESOME** (**A**ligning **W**ord **E**mbedding **S**paces **O**f **M**ultilingual **E**ncoders), with pre-trained models is available at `https://github.com/neulab/awesome-align`.



	The	necessary	correct	##ion	will	be	made	.
Eine	0.89	0.82	0.78	0.80	0.77	0.77	0.78	0.79
entsprechende	0.83	0.85	0.82	0.81	0.78	0.77	0.79	0.79
Änderung	0.78	0.80	0.84	0.84	0.77	0.74	0.78	0.80
wird	0.82	0.80	0.78	0.80	0.88	0.86	0.81	0.81
vor	0.78	0.78	0.79	0.78	0.81	0.82	0.82	0.77
##genommen	0.76	0.77	0.76	0.77	0.78	0.81	0.84	0.77
werden	0.76	0.73	0.74	0.75	0.77	0.80	0.80	0.78
.	0.79	0.75	0.76	0.75	0.76	0.73	0.77	0.95

Before Fine-tuning

	The	necessary	correct	##ion	will	be	made	.
Eine	0.88	0.77	0.75	0.77	0.76	0.77	0.76	0.76
entsprechende	0.77	0.83	0.77	0.76	0.75	0.72	0.75	0.75
Änderung	0.73	0.71	0.83	0.85	0.74	0.68	0.73	0.75
wird	0.79	0.74	0.72	0.74	0.90	0.87	0.77	0.76
vor	0.73	0.75	0.73	0.73	0.77	0.79	0.86	0.73
##genommen	0.73	0.75	0.72	0.74	0.76	0.80	0.88	0.73
werden	0.77	0.72	0.71	0.73	0.81	0.88	0.79	0.75
.	0.76	0.72	0.75	0.75	0.75	0.71	0.73	0.97

After Fine-tuning

Figure 1: Cosine similarities between subword representations in a parallel sentence pair before and after fine-tuning. Red boxes indicate the gold alignments.

1 Introduction

Word alignment is a useful tool to tackle a variety of natural language processing (NLP) tasks, including learning translation lexicons (Ammar et al., 2016; Cao et al., 2019), cross-lingual transfer of language processing tools (Yarowsky et al., 2001; Padó and Lapata, 2009; Tiedemann, 2014; Agić et al., 2016; Mayhew et al., 2017; Nicolai and Yarowsky, 2019), semantic parsing (Herzig and Berant, 2018) and speech recognition (Xu et al., 2019). In particular, word alignment plays a crucial role in many machine translation (MT) related methods, including guiding learned attention (Liu et al., 2016), incorporating lexicons during decoding (Arthur et al., 2016), domain adaptation (Hu et al., 2019), unsupervised MT (Ren et al., 2020) and automatic evaluation or analysis of translation models (Bau et al., 2018; Stanovsky et al., 2019; Neubig et al., 2019; Wang et al., 2020). However, with neural networks advancing the state of the arts in almost every field of NLP, tools developed based on the 30-year-old IBM word-based translation models (Brown et al., 1993), such as GIZA++ (Och and Ney, 2003) or fast-align (Dyer et al., 2013), remain popular choices for word alignment tasks.

One alternative to using statistical word-based translation models to learn alignments would be to instead train state-of-the-art neural machine translation (NMT) models on parallel corpora, and extract alignments therefrom, as examined by Luong et al. (2015); Garg et al. (2019); Zenkel et al. (2020). However, these methods have two disadvantages (also shared with more traditional alignment methods): (1) they are directional and the source and target side are treated differently and (2) they cannot easily take advantage of large-scale contextualized

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2112–2128
April 19 - 23, 2021.

word embeddings derived from language models (LMs) multilingually trained on monolingual corpora (Devlin et al., 2019; Lample and Conneau, 2019; Conneau et al., 2020), which have proven useful in other cross-lingual transfer settings (Libovický et al., 2019; Hu et al., 2020b). In the field of word alignment, Sabet et al. (2020) have recently proposed methods to align words using multilingual contextualized embeddings and achieve good performance even in the absence of explicit training on parallel data, suggesting that these are an attractive alternative for neural word alignment.

In this paper, we investigate if we can combine the best of the two lines of approaches. Concretely, we leverage pre-trained LMs and fine-tune them on parallel text with not only LM-based objectives, but also unsupervised objectives over the parallel corpus designed to improve alignment quality. Specifically, we propose a self-training objective, which encourages aligned words to have further closer contextualized representations, and a parallel sentence identification objective, which enables the model to bring parallel sentences' representations closer to each other. In addition, we propose to effectively extract alignments from these fine-tuned models using probability thresholding or optimal transport.

We perform experiments on five different language pairs and demonstrate that our model can achieve state-of-the-art performance on all of them. In analysis, we find that these approaches also generate more aligned contextualized representations after fine-tuning (see Figure 1 as an example) and we can incorporate supervised signals within our paradigm. Importantly, we show that it is possible to train multilingual word aligners that can obtain robust performance even in zero-shot settings, making them a valuable tool that can be used out-of-the-box with good performance over a wide variety of language pairs.

2 Methods

Formally, the task of word alignment can be defined as: given a sentence $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$ in the source language and its corresponding parallel sentence $\mathbf{y} = \langle y_1, \cdots, y_m \rangle$ in the target language, a word aligner needs to find a set of pairs of source and target words:

$$A = \{\langle x_i, y_j \rangle : x_i \in \mathbf{x}, y_j \in \mathbf{y}\},$$

where for each word pair $\langle x_i, y_j \rangle$, x_i and y_j are semantically similar to each other within the context of the sentence.

In the following paragraphs, we will first illustrate how we extract alignments from contextualized word embeddings, then describe our objectives designed to improve alignment quality.

2.1 Extracting Alignments from Embeddings

Contextualized word embedding models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) represent words using continuous vectors calculated in context, and have achieved impressive performance on a diverse array of NLP tasks. Multilingually trained word embedding models such as multilingual BERT can generate contextualized embeddings across different languages. These models can be used to extract contextualized word embeddings $h_{\mathbf{x}} = \langle h_{x_1}, \cdots, h_{x_n} \rangle$ and $h_{\mathbf{y}} = \langle h_{y_1}, \cdots, h_{y_m} \rangle$ for each pair of parallel sentences $\mathbf{x}$ and $\mathbf{y}$. Specifically, this is done by extracting the hidden states of the i-th layer of the model, where i is an empirically-chosen hyper-parameter. Given these contextualized word embeddings, we propose two methods to calculate unidirectional alignment scores based on probability simplexes and optimal transport. We then turn these alignment scores into alignment matrices and reconcile alignments in the forward and backward directions.

Probability Thresholding. In this method, for each word in the source/target sentence, we calculate a value on the probability simplex for each word in the aligned target/source sentence, and then select all values that exceed a particular threshold as "aligned" words. Concretely, taking inspiration from attention mechanisms (Bahdanau et al., 2015; Vaswani et al., 2017), we take the contextualized embeddings $h_{\mathbf{x}}$ and $h_{\mathbf{y}}$ and compute the dot products between them and get the similarity matrix:

$$S = h_{\mathbf{x}} h_{\mathbf{y}}^T.$$

Then, we apply a normalization function $\mathcal{N}$ to convert the similarity matrix into values on the probability simplex $S_{\mathbf{xy}} = \mathcal{N}(S)$, and treat $S_{\mathbf{xy}}$ as the source-to-target alignment matrix. In this paper, we propose to use *softmax* and a sparse variant α-entmax (Peters et al., 2019) to do the normalization. Compared with the *softmax* function, α-entmax can produce sparse alignments for any $\alpha > 1$ and assign non-zero probability to a short

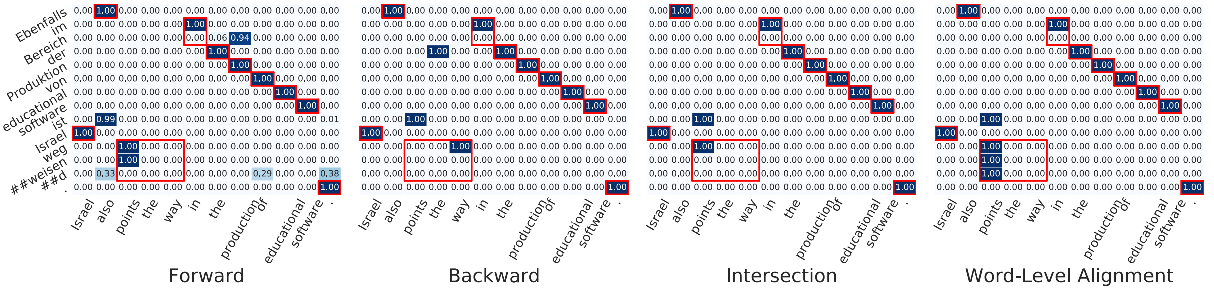


Figure 2: Extracting word alignments from multilingual BERT using probability thresholding (*softmax*). Red boxes denote the gold alignments.

list of plausible word pairs, where a higher α will lead to a more sparse alignment.

Optimal Transport. The goal of optimal transport (Monge, 1781; Cuturi, 2013) is to find a mapping that moves probability from one distribution to another, which can be used to find an optimal matching of similar words between two sequences (Kusner et al., 2015). Formally, in a discrete optimal transport problem, we are given two point sets $\{x_i\}_{i=1}^n$ and $\{y_j\}_{j=1}^m$ associated with their probability distributions $p_{\mathbf{x}}$ and $p_{\mathbf{y}}$ where $\sum_i p_{x_i} = 1$ and $\sum_j p_{y_j} = 1$. Also, a function $C(x_i, y_j)$ defines the cost of moving point x_i to y_j. The goal of optimal transport is to find a mapping that moves probability mass from $\{x_i\}_{i=1}^n$ to $\{y_j\}_{j=1}^m$ and the total cost of moving the mass between points is minimized. In other words, it finds the transition matrix $S_{\mathbf{xy}}$ that minimizes:

$$\sum_{i,j} C(x_i, y_j) S_{\mathbf{xy}_{ij}}, \qquad (1)$$

where $S_{\mathbf{xy}} \mathbf{1}_m = p_{\mathbf{x}}$ and $S_{\mathbf{xy}}^T \mathbf{1}_n = p_{\mathbf{y}}$. The resulting transition matrix is self-normalized and sparse (Swanson et al., 2020), making it appealing alternative towards extracting alignments from word embeddings.

In this paper, we propose to adapt optimal transport techniques to the task of word alignment. Concretely, we treat the parallel sentences $\mathbf{x}$ and $\mathbf{y}$ as two point sets and assume each word is uniformly distributed. The cost function is obtained by computing the pairwise distance (e.g. cosine distance) between $h_{\mathbf{x}}$ and $h_{\mathbf{y}}$, and all the distance values are scaled to [0, 1] with min-max normalization. The optimal transition matrix $S_{\mathbf{xy}}$ to Equation 1 can be calculated using the Sinkhorn-Knopp matrix scaling algorithm (Sinkhorn and Knopp, 1967). If the value of $S_{\mathbf{xy}_{ij}}$ is high, x_i and y_j are likely to have similar semantics and values that exceed a particular threshold will be considered as "aligned".

Extracting Bidirectional Alignments. After we obtain both the source-to-target and target-to-source alignment probability matrices $S_{\mathbf{xy}}$ and $S_{\mathbf{yx}}$ using the previous methods, we can deduce the final alignment matrix by taking the intersection of the two matrices:

$$A = (S_{\mathbf{xy}} > c) * (S_{\mathbf{yx}}^T > c),$$

where c is a threshold and $A_{ij} = 1$ means x_i and y_j are aligned.

Note that growing heuristics such as *grow-diag-final* (Och and Ney, 2000; Koehn et al., 2005) that are popular in statistical word aligners can also be applied in our alignment extraction algorithms, and we will demonstrate the effect of these heuristics in the experiment section.

Handling Subwords. Subword segmentation techniques (Sennrich et al., 2016; Kudo and Richardson, 2018) are widely used in training LMs, thus the above alignment extraction methods can only produce alignments on the subword level. To convert them to word alignments, we follow previous work (Sabet et al., 2020; Zenkel et al., 2020) and consider two words to be aligned if any of their subwords are aligned. Figure 2 shows a concrete example of how we extract word-level alignments from a pre-trained embedding model.

2.2 Fine-tuning Contextualized Embeddings for Word Alignment

While language models can be used to produce reasonable word alignments even without any fine-tuning (Sabet et al., 2020), we propose objectives that further improve their alignment ability if we have access to parallel data.

Masked Language Modeling (MLM). Gururangan et al. (2020) suggest that we can gain improvements in downstream tasks by further pretraining LMs on the task datasets. Therefore, we propose to fine-tune the LMs with a masked language modeling objective on both the source and target side of parallel corpora. Specifically, given a pair of parallel sentences $\mathbf{x}$ and $\mathbf{y}$, we choose 15% of the token positions randomly for both $\mathbf{x}$ and $\mathbf{y}$, and for each chosen token, we replace it with (1) the `[MASK]` token 80% of the time (2) a random token 10% of the time and (3) unchanged 10% of the time. The model is trained to reconstruct the original tokens given the masked sentences $\mathbf{x}^{mask}$ and $\mathbf{y}^{mask}$:

$$L_{MLM} = \log p(\mathbf{x}|\mathbf{x}^{mask}) + \log p(\mathbf{y}|\mathbf{y}^{mask}). \quad (2)$$

Translation Language Modeling (TLM). The MLM objective only requires monolingual data and the model cannot make direct connections between parallel sentences. To solve the issue, similarly to Lample and Conneau (2019), we concatenate parallel sentences $\mathbf{x}$ and $\mathbf{y}$ and perform MLM on the concatenated data. Compared with MLM, the translation language modeling (TLM) objective enable the model to align the source and target representations. Different from Lample and Conneau (2019), we feed source and target sentences twice in different orders instead of resetting the positions of target sentences:

$$\begin{aligned} L_{TLM} &= \log p([\mathbf{x};\mathbf{y}]|[\mathbf{x}^{mask};\mathbf{y}^{mask}]) \\ &+ \log p([\mathbf{y};\mathbf{x}]|[\mathbf{y}^{mask};\mathbf{x}^{mask}]). \end{aligned} \quad (3)$$

Self-training Objective (SO). We also propose a self-training objective for fine-tuning LMs which is similar to the EM algorithm used in the IBM models and the agreement constraints in Tamura et al. (2014). Specifically, at each training step, we first use our alignment extraction methods (described in Section 2.1) to extract the alignment A for $\mathbf{x}$ and $\mathbf{y}$, then maximize the following objective:

$$L_{SO} = \sum_{i,j} A_{ij} \frac{1}{2}\left(\frac{S_{\mathbf{xy}_{ij}}}{n} + \frac{S_{\mathbf{yx}_{ij}}}{m}\right). \quad (4)$$

Intuitively, this objective encourages words aligned in the first pass of alignment to have further closer contextualized representations. In addition, because of the intersection operation during extraction, the self-training objective can ideally reduce spurious alignments and encourage the source-to-target and target-to-source alignments to be symmetrical to each other by exploiting their agreement (Liang et al., 2006).

	De-En	Fr-En	Ro-En	Ja-En	Zh-En
#Train Sents.	1.9M	1.1M	450K	444K	40K
#Test Sents.	508	447	248	582	450

Table 1: Statistics of datasets.

Parallel Sentence Identification (PSI). We also propose a contrastive parallel sentence identification loss that attempts to make parallel sentences more similar than mismatched sentence pairs (Liu and Sun, 2015; Legrand et al., 2016). This encourages the overall alignments of embeddings on both word and sentence level to be closer together. Concretely, we randomly select a pair of parallel or non-parallel sentences $\langle \mathbf{x}', \mathbf{y}' \rangle$ from the training data with equal probability. Then, the model is required to predict whether the two sampled sentences are parallel or not. The representation of the first `[CLS]` token is fed into a multi-layer perceptron to output a prediction score $s(\mathbf{x}', \mathbf{y}')$. Denoting the binary label as l, the objective function can be written as:

$$L_{PSI} = l \log s(\mathbf{x}', \mathbf{y}') + (1-l)\log(1 - s(\mathbf{x}', \mathbf{y}')). \quad (5)$$

Consistency Optimization (CO). While the self-training objective can potentially improve the symmetricity between forward and backward alignments, following previous work on machine translation and multilingual representation learning (Cohn et al., 2016; Zhang et al., 2019; Hu et al., 2020a), we use an objective to explicitly encourage the consistency between the two alignment matrices. Specifically, we maximize the trace of $S_{\mathbf{xy}}^{\mathrm{T}} S_{\mathbf{yx}}$:

$$L_{CO} = -\frac{\mathrm{trace}(S_{\mathbf{xy}}^{\mathrm{T}} S_{\mathbf{yx}})}{\min(m, n)}. \quad (6)$$

Our Final Objective. In summary, our training objective is a combination of the proposed objectives and we train the model with them jointly at each training step:

$$L = L_{MLM} + L_{TLM} + L_{SO} + L_{PSI} + \beta L_{CO},$$

where β is set to 0 or 1 in our experiments.

Model	**Setting**	**De-En**	**Fr-En**	**Ro-En**	**Ja-En**	**Zh-En**
Baseline						
SimAlign	*w/o fine-tuning*	18.8	7.6	27.2	46.6	21.6
fast_align	*bilingual*	27.0	10.5	32.1	51.1	38.1
eflomal	*bilingual*	22.6	8.2	25.1	47.5	28.7
GIZA++	*bilingual*	20.6	5.9	26.4	48.0	35.1
Zenkel et al. (2020)	*bilingual*	16.0	5.0	23.4	-	-
Chen et al. (2020)	*bilingual*	15.4	4.7	21.2	-	-
Ours						
α-entmax	*w/o fine-tuning*	18.1	5.6	29.0	46.3	18.4
	bilingual	16.1	***4.1***	23.4	38.6	15.4
	multilingual (β = 0)	15.4	***4.1***	22.9	***37.4***	**13.9**
	multilingual (β = 1)	***15.0***	4.5	**20.8**	38.7	14.5
	zero-shot	16.0	4.3	28.4	44.0	**13.9**
softmax	*w/o fine-tuning*	17.4	5.6	27.9	45.6	18.1
	bilingual	15.6	**4.4**	23.0	38.4	15.3
	multilingual (β = 0)	15.3	**4.4**	22.6	**37.9**	***13.6***
	multilingual (β = 1)	**15.1**	4.5	***20.7***	38.4	14.5
	zero-shot	15.7	4.6	27.2	43.7	14.0

Table 2: Performance (AER) of our models in bilingual, multilingual and zero-shot settings. The best scores for each alignment extraction method are in **bold** and the overall best scores are in ***italicized bold***.

3 Experiments

In this section, we first present our main results, then conduct several ablation studies and analyses of our models.

3.1 Setup

Datasets. We perform experiments on five different language pairs, namely German-English (De-En), French-English (Fr-En), Romanian-English (Ro-En), Japanese-English (Ja-En) and Chinese-English (Zh-En). For the De-En, Fr-En, Ro-En datasets, we follow the experimental setting of previous work (Zenkel et al., 2019; Garg et al., 2019; Zenkel et al., 2020). The training and test data for Ro-En and Fr-En are provided by Mihalcea and Pedersen (2003). The Ro-En training data are also augmented by the Europarl v8 corpus (Koehn, 2005). For the De-En data, the Europarl v7 corpus is used as training data and the gold alignments are provided by Vilar et al. (2006). The Ja-En dataset is obtained from the Kyoto Free Translation Task (KFTT) word alignment data (Neubig, 2011), and the Japanese sentences are tokenized with the KyTea tokenizer (Neubig et al., 2011). The Zh-En dataset is obtained from the TsinghuaAligner website[1]. We treat their evaluation set as the training data and use the test set in Liu and Sun (2015). The De-En, En-Fr, Zh-En datasets contain the distinction between sure and possible alignment links. The statistics of these datasets are shown in Table 1. We use the Ja-En development set to tune the hyper-parameters.

Baselines. We compare our models with:

- fast_align (Dyer et al., 2013): a popular statistical word aligner which is a simple, fast reparameterization of IBM Model 2.
- eflomal (Östling and Tiedemann, 2016): an efficient statistical word aligner using a Bayesian model with Markov Chain Monte Carlo (MCMC) inference.
- GIZA++ (Och and Ney, 2003; Gao and Vogel, 2008): an implementation of IBM models. Following previous work (Zenkel et al., 2020), we use five iterations each for Model 1, the HMM model, Model 3 and Model 4.
- SimAlign (Sabet et al., 2020): a BERT-based word aligner that is not fine-tuned on any parallel data. The authors propose three alignment extraction methods and we implement their IterMax model with default parameters.
- Zenkel et al. (2020) and Chen et al. (2020): two state-of-the-art neural word aligners based on MT models.

---

[1] `http://nlp.csai.tsinghua.edu.cn/~ly/systems/TsinghuaAligner/TsinghuaAligner.html`

Implementation Details. Our main results are obtained by using the probability thresholding method on the contextualized embeddings in the 8-th layer of multilingual BERT-Base (mBERT; Devlin et al. (2019)) and we will discuss this choice in our ablation studies. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2e-5 and the batch size is set to 8. Following Peters et al. (2019), we set α to 1.5 for α-entmax. The threshold c is set to 0 for α-entmax and 0.001 for *softmax* and optimal transport. Unless otherwise stated, β is set to 0. We mainly evaluate the model performance using Alignment Error Rate (AER).

3.2 Main Results

We first train our model on each individual language pair, then investigate if it is possible to train multilingual word aligners.

Bilingual Model Performance. From Table 2, we can see that our *softmax* model can achieve consistent improvements over the baseline models, demonstrating the effectiveness of our proposed method. Surprisingly, directly extracting alignments from mBERT (the *w/o fine-tuning* setting) can already achieve better performance than the popular statistical word aligner GIZA++ on 4 out of 5 settings, especially in the Zh-En setting where the size of parallel data is small.

Multilingual Model Performance. We also randomly sample 200k parallel sentence pairs from each language pair (except for Zh-En where we take all of its 40k parallel sentences) and concatenate them together to train multilingual word aligners. As shown in Table 2, the multilingually trained word aligners can achieve further improvements and they consistently outperform our bilingual word aligners and all the baselines even though the size of training data for each individual language pair is smaller. The results demonstrate that we can indeed obtain a neural word aligner that has state-of-the-art and robust performance across different language pairs. We also test the performance of our consistency optimization objective in this setting. We can see that incorporating this objective (β=1) can significantly improve the model performance on Ro-En, while it also deteriorates the Ja-En and Zh-En performance by a non-negligible margin. We find that this is because the CO objective can significantly improve the alignment recall while sacrificing the precisions, and our Ro-En dataset tends to favor models with high recall and the Ja-En and Zh-En datasets have an opposite tendency.

	Component	De-En	Fr-En	Ro-En	Ja-En	Zh-En	Speed
Prob.	*softmax*	**17.4**	**5.6**	**27.9**	**45.6**	**18.1**	**33.22**
	α-entmax	18.1	**5.6**	29.0	46.3	18.4	32.36
OT	Cosine	24.4	15.7	33.7	54.0	31.1	3.36
	Dot Product	25.4	17.1	34.1	54.2	30.9	3.82
	Euclidean	20.7	15.1	33.3	53.2	29.8	3.05

Table 3: Comparisons of probability thresholding (Prob.) and optimal transport (OT) for alignment extraction. We try both *softmax* and α-entmax for probability thresholding and different cost functions for optimal transport. We measure both the extraction speed (#sentences/seconds) and the alignment quality (AER) on five language pairs, namely German-English (De-En), French-English (Fr-En), Romanian-English (Ro-En), Japanese-English (Ja-En), and Chinese-English (Zh-En). The best scores are in **bold**.

Zero-Shot Performance. In this paragraph, we want to find out how our models perform on language pairs that it has never seen during training. To this end, for each language pair, we train our model with data of all the other language pairs and test its performance on the target language pair. Results in Table 2 demonstrate that training our models with parallel data on *other* language pairs can still improve the model performance on the target language pair. This is a very important result, as it indicates that our model can be used as a off-the-shelf tool for multilingual word alignment for any language supported by the underlying embeddings, *regardless of whether parallel data has been used for training or not*.

3.3 Ablation Studies

In this part, we compare the performance of different alignment extraction methods, pre-trained embedding models and training objectives.

Alignment Extraction Methods. We first compare the performance of our two proposed alignment extraction methods, namely the probability thresholding and optimal transport techniques. We use the representations of the 8-th layer of mBERT following Sabet et al. (2020).

As shown in Table 3, probability thresholding methods can consistently outperform optimal transport by a large margin on the five language pairs. In addition, probability thresholding methods are much faster than optimal transport. *softmax* is marginally better than α-entmax, yet one advantage of α-entmax is that we do not need to manually set

Model	Layer	De-En	Fr-En	Zh-En
mBERT	7	18.7	6.1	19.1
	8	**17.4**	**5.6**	***18.1***
	9	18.8	6.1	20.1
XLM-15 (MLM)	4	21.1	6.8	**25.3**
	5	**20.4**	**6.1**	26.1
	6	23.2	7.7	33.3
XLM-15 (MLM+TLM)	4	16.4	4.9	**18.6**
	5	***16.2***	***4.7***	23.7
	6	18.8	5.7	26.2
XLM-100 (MLM)	7	20.5	8.5	30.8
	8	**19.8**	**8.2**	**28.6**
	9	19.9	8.8	29.3
XLM-R	5	24.4	10.3	33.2
	6	**23.1**	**9.2**	30.7
	7	24.7	11.5	**28.1**

Table 4: Comparisons of different LMs in terms of AER. We extract alignments using *softmax* and take representations from different layers of LMs. The best scores for each individual model are in **bold** and the overall best scores are in ***italicized bold***.

the threshold. Therefore, we use both *softmax* and α-entmax to obtain the main results.

Pre-trained Embedding Models. In this paragraph, we investigate the performance of three different types of pre-trained embedding models, including mBERT, XLM (Lample and Conneau, 2019) and XLM-R (Conneau et al., 2020). For XLM, we have tried its three released models: 1) XLM-15 (MLM) pre-trained with MLM and supports 15 languages; 2) XLM-15 (MLM+TLM) pre-trained with both the MLM and TLM objectives and supports 15 languages; 3) XLM-100 (MLM) pre-trained with MLM and supports 100 languages. We use *softmax* to extract the alignments.

Because XLM-15 does not support Japanese or Romanian, we only report the performance on the three other language pairs in Table 4. We take representations from different layers and report the performance of the best three layers. We can see that while XLM-15 (MLM+TLM) can achieve the best performance on De-En and Fr-En, the best layer is not consistent across language pairs. On the other hand, the optimal configurations for mBERT are consistent across language pairs. In addition, considering mBERT supports many more languages than XLM-15 (MLM+TLM), we will use mBERT in the following sections.

Training Objectives. We also conduct ablation studies on each of our training objectives. We can see from Table 5 that the self-training objective can best improve the model performance. Also, the translation language modeling and parallel sentence identification objectives can marginally benefit the model. The masked language modeling objective, on the other hand, cannot always improve the model and can sometimes even deteriorate the model performance, possibly because the TLM objective already provides the model with sufficient supervision signals.

3.4 Analysis

We conduct several analyses to better understand our models. Unless otherwise stated, we perform experiments on the *softmax* model using mBERT.

Incorporating Supervised Signals. We investigate if our models can benefit from supervised signals. If we have access to word-level gold labels for word alignment, we can simply utilize them in our self-training objectives. Specifically, we can set A_{ij} in Equation 4 to 1 if and only if they are aligned. In our experimental settings, we have gold labels for all the Zh-En sentences and 653 sentences from the Ja-En development set. Table 6 demonstrates that training our models with as few as 653 labeled sentences can dramatically improve the alignment quality, and combining labeled and unlabeled parallel data can further improve the model performance. This analysis demonstrate the generality of our models as they can also be applied in semi-supervised settings.

Growing Heuristics. As stated in Section 2.1, because our alignment extraction methods essentially take the intersection of forward and backward alignments, growing heuristics can also be applied in our settings. The main motivation of growing heuristics is to improve the recall of the resulting alignments. While effective in statistical word aligners, as shown in Table 7, the growing heuristics only improve our alignment extraction method on the vanilla mBERT model in the Ro-En setting while degrading the model performance on all the other language pairs. After fine-tuning, the growing heuristics can only hurt the model performance, possibly because the self-training objective encourages the forward and backward alignments to be symmetrical. Based on these results, we do not adopt the growing heuristics in our models.

Annotation Projection. Word alignment has been a useful tool in cross-lingual annotation projection (Yarowsky et al., 2001; Nicolai and Yarowsky, 2019). Therefore, it would be inter-

Model	Objective	De-En	Fr-En	Ro-En	Ja-En	Zh-En
	All	15.3	4.4	22.6	37.9	13.6
	All w/o MLM	15.3	4.4	22.8	38.6	13.7
softmax	*All w/o TLM*	15.5	4.7	22.9	39.7	14.0
	All w/o SO	16.9	4.8	23.0	39.1	15.4
	All w/o PSI	15.4	4.4	22.7	37.9	13.8

Table 5: Ablation studies on our training objectives in multilingual settings.

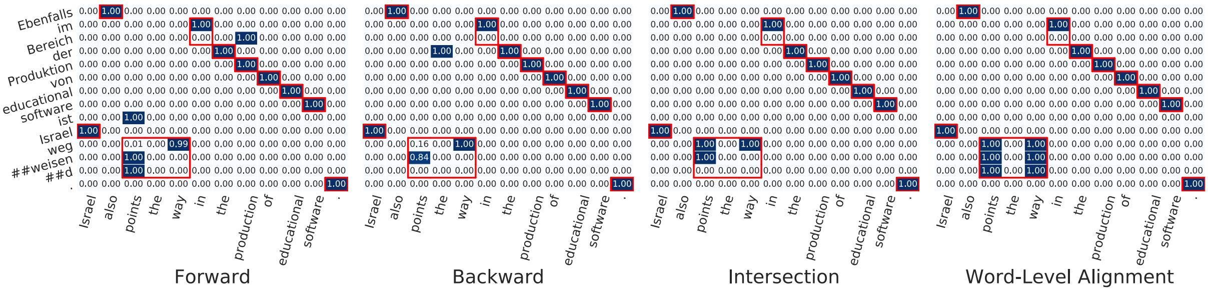


Figure 3: An example of extracting alignments from our fine-tuned model using *softmax*. Red boxes indicate the gold alignments. The fine-tuned model can generate more accurate alignments then vanilla mBERT (Figure 2).

Lang.	Unsup.	Sup.	Semi-Sup.
Zh-En	15.3	12.5	-
Ja-En	38.4	31.6	30.0

Table 6: Incorporating supervised word alignment signals into our model can further improve the model performance in terms of AER.

Model	Ext.	De-En	Fr-En	Ro-En	Ja-En	Zh-En
	X-En	24.7	14.4	31.9	54.7	27.4
	En-X	22.6	12.2	32.0	52.7	29.9
mBERT	*softmax*	17.4	5.6	27.9	45.6	18.1
	gd	18.7	9.2	27.0	48.5	23.4
	gd-final	18.6	9.3	26.9	48.7	23.2
	X-En	20.2	12.9	25.4	42.1	19.3
	En-X	18.1	9.3	25.9	41.7	23.5
Ours-Multi.	*softmax*	15.3	4.4	22.6	37.9	13.6
	gd	16.3	8.1	23.1	38.2	18.3
	gd-final	16.5	8.3	23.2	38.7	18.5

Table 7: The *grow-diag-final* heuristic can only improve our alignment extraction method in the Romanian-English setting without fine-tuning. "gd" refers to grow-diag.

Model	Prec. %	Rec. %	F_1 %
BERT-En (zero-shot)	53.1	54.3	52.7
fast_align	51.5	59.8	55.2
GIZA++	56.5	64.1	60.0
SimAlign	59.9	67.6	63.5
Ours	**60.6**	**68.5**	**64.3**

Table 8: Our model is also effective in an annotation projection setting where we train a BERT-based NER model on English data and test it on Spanish data. The best scores are in **bold**.

esting to see if our model can be beneficial in these settings. To this end, we evaluate our model and baselines on cross-lingual named entity recognition (NER). We train a BERT-based NER model on the CoNLL 2003 English data (Tjong Kim Sang and De Meulder, 2003) and test it on the CoNLL 2002 Spanish data (Tjong Kim Sang, 2002). We use Google Translate to translate Spanish test set into English, predict the labels using the NER model, then project the labels from English to Spanish using word aligners. From Table 8, we can see that our model is also better than baselines in this setting, demonstrating its usefulness in cross-lingual annotation projection.

Sentence-Level Representation Transfer. We also test if the aligned representations are beneficial for sentence-level cross-lingual transfer. In doing so, we perform experiments on XNLI (Conneau et al., 2018), which evaluates cross-lingual sentence representations in 15 languages on the task of natural language inference (NLI). We train our models with the provided 10k parallel data on the 15 languages, fine-tune our model on the English NLI data, then test its performance on other languages. As shown in Table 9, our model can outperform the baseline, indicating the aligned word representations can also be helpful for sentence-level cross-lingual transfer.

Model	En	Fr	Es	De	El	Bg	Ru	Tr	Ar	Vi	Th	Zh	Hi	Sw	Ur	Ave.
mBERT	81.3	73.4	74.3	70.5	66.9	68.2	68.5	59.5	64.3	**70.6**	50.7	68.8	59.3	49.4	57.5	65.5
Ours	**81.5**	**74.1***	**74.9***	**71.2***	**67.1**	**68.7***	**68.6**	**61.0***	**66.2***	70.5	**53.8***	**69.1**	**59.8***	**50.6***	**58.6***	**66.4***

Table 9: Results of mBERT and our fine-tuned model on XNLI (Conneau et al., 2018). Our objectives can improve the model cross-lingual transfer ability. "*" denotes significant differences using paired bootstrapping ($p<0.05$) .

Alignment Examples. We also conduct qualitative analyses as shown in Figure 1, 2 and 3. After fine-tuning, the learned contextualized representations are more aligned, as the cosine distances between semantically similar words become closer, and the extracted alignments are more accurate. More examples are shown in Appendix B.

4 Related Work

Based on the IBM translation models (Brown et al., 1993), many statistical word aligners have been proposed (Vogel et al., 1996; Östling and Tiedemann, 2016), including the current most popular tools GIZA++ (Och and Ney, 2000, 2003; Gao and Vogel, 2008) and fast_align (Dyer et al., 2013).

Recently, there is a resurgence of interest in neural word alignment (Tamura et al., 2014; Alkhouli et al., 2018). Based on NMT models trained on parallel corpora, researchers have proposed several methods to extract alignments from them (Luong et al., 2015; Zenkel et al., 2019; Garg et al., 2019; Li et al., 2019) and successfully build an end-to-end neural model that can outperform statistical tools (Zenkel et al., 2020). However, there is an inherent discrepancy between translation and word alignment: translation models are directional and the source and target side are treated differently, while word alignment is a non-directional task. Therefore, certain adaptations are required for translation models to perform word alignment.

Another disadvantage of MT-based word aligners is that they cannot easily utilize contextualized embeddings. Using learned representations to improve word alignment have been investigated (Sabet et al., 2016; Pourdamghani et al., 2018). Recently, pre-trained LMs (Peters et al., 2018; Devlin et al., 2019; Brown et al., 2020) have proven to be useful in cross-lingual transfer (Libovickỳ et al., 2019; Hu et al., 2020b). In word alignment, Sabet et al. (2020) propose effective methods to extract alignments from multilingual LMs without explicit training on parallel data. In this work, we propose better alignment extraction methods and combine the best of the two worlds by fine-tuning contextualized embeddings on parallel data.

There are also work on supervised neural word alignment (Stengel-Eskin et al., 2019; Nagata et al., 2020). However, supervised data are not always accessible, making their methods inapplicable in many scenarios. In this paper, we demonstrate that our model can incorporate supervised signals if available and perform semi-supervised learning, which is a more realistic and general setting.

Some work on bilingual lexicon induction also share similar general ideas with ours. For example, Zhang et al. (2017) minimize the earth mover's distance to match the embedding distributions from different languages. Similarly, Grave et al. (2019) present an algorithm to align point clouds with Procrustes (Schönemann, 1966) in Wasserstein distance for unsupervised embedding alignment.

5 Discussion and Conclusion

We present a neural word aligner that achieves state-of-the-art performance on five diverse language pairs and obtains robust performance in zero-shot settings. We propose to fine-tune multilingual embeddings with objectives suitable for word alignment and develop two alignment extraction methods. We also demonstrate its applications in semi-supervised settings. We hope our word aligner can be a tool that can be used out-of-the-box with good performance over various language pairs. Future directions include designing better training objectives and experimenting on more language pairs.

Also, note that we mainly evaluate our word aligners using AER following previous work, which has certain limitations. For example, it may not be well-correlated with statistical machine translation performance Fraser and Marcu (2007) and different types of alignments can be suitable for different tasks or conditions (Lambert et al., 2012; Stymne et al., 2014). Although we have evaluated models in annotation projection and cross-lingual transfer settings, alternative metrics (Tiedemann, 2005; Søgaard and Wu, 2009; Ahrenberg, 2010) are also worth considering in the future.

Acknowledgement

We thank our reviewers for helpful suggestions.

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*.

Lars Ahrenberg. 2010. Alignment-based profiling of europarl data in an english-swedish parallel corpus. In *Proceedings of the International Conference on Language Resources and Evaluation*.

Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney. 2018. On the alignment problem in multi-head attention-based neural machine translation. In *Proceedings of the Conference on Machine Translation*.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. *arXiv preprint*.

Philip Arthur, Graham Neubig, and Satoshi Nakamura. 2016. Incorporating discrete translation lexicons into neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Rrepresentations*.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2018. Identifying and controlling important neurons in neural machine translation. In *Proceedings of the International Conference on Learning Representations*.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*.

Tom B. Brown, Benjamin Pickman Mann, Nick Ryder, Melanie Subbiah, Jean Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, G. Krüger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric J Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *arXiv preprint*.

Steven Cao, Nikita Kitaev, and Dan Klein. 2019. Multilingual alignment of contextual word representations. In *Proceedings of the International Conference on Learning Representations*.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020. Accurate word alignment induction from neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Proceedings of the Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Edouard Grave, Armand Joulin, and Quentin Berthet. 2019. Unsupervised alignment of embeddings with wasserstein procrustes. In *Proceedinds of the International Conference on Artificial Intelligence and Statistics*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Jonathan Herzig and Jonathan Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Junjie Hu, Melvin Johnson, Orhan Firat, Aditya Siddhant, and Graham Neubig. 2020a. Explicit alignment objectives for multilingual bidirectional encoders. *arXiv preprint*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020b. XTREME: A massively multilingual multitask benchmark for evaluating cross-lingual generalisation. In *Proceedings of the International Conference on Machine Learning*.

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime G Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the International Conference on Machine Learning*.

Patrik Lambert, Simon Petitrenaud, Yanjun Ma, and Andy Way. 2012. What types of word alignment improve statistical machine translation? *Machine Translation*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Proceedings of the Advances in Neural Information Processing Systems*.

Joël Legrand, Michael Auli, and Ronan Collobert. 2016. Neural network-based word alignment through score aggregation. In *Proceedings of the Conference on Machine Translation*.

Xintong Li, Guanlin Li, Lemao Liu, Max Meng, and Shuming Shi. 2019. On the word alignment from neural machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2019. How language-neutral is multilingual BERT? *arXiv preprint*.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of the International Conference on Computational Linguistics*.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint*.

I. Loshchilov and F. Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Bill MacCartney, Michel Galley, and Christopher D Manning. 2008. A phrase-based alignment model for natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Rada Mihalcea and Ted Pedersen. 2003. An evaluation exercise for word alignment. In *Proceedings of Workshop on Building and Using Parallel Texts*.

Gaspard Monge. 1781. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences de Paris*.

Masaaki Nagata, Chousa Katsuki, and Masaaki Nishino. 2020. A supervised word alignment method based on cross-language span prediction using multilingual BERT. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Graham Neubig. 2011. The Kyoto free translation task. http://www.phontron.com/kftt.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: System Demonstrations*.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Garrett Nicolai and David Yarowsky. 2019. Learning morphosyntactic analyzers from the Bible via iterative annotation projection across 26 languages. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*.

Robert Östling and Jörg Tiedemann. 2016. Efficient word alignment with Markov Chain Monte Carlo. *The Prague Bulletin of Mathematical Linguistics*.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*.

Ben Peters, Vlad Niculae, and André FT Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Nima Pourdamghani, Marjan Ghazvininejad, and Kevin Knight. 2018. Using word vectors to improve word alignments for low resource machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Shuo Ren, Yu Wu, Shujie Liu, Ming Zhou, and Shuai Ma. 2020. A retrieve-and-rewrite initialization method for unsupervised machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: Findings*.

Masoud Jalili Sabet, Heshaam Faili, and Gholamreza Haffari. 2016. Improving word alignment of rare words with word embeddings. In *Proceedings of the International Conference on Computational Linguistics*.

Peter H Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Richard Sinkhorn and Paul Knopp. 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*.

Anders Søgaard and Dekai Wu. 2009. Empirical lower bounds on translation unit error rate for the full class of inversion transduction grammars. In *Proceedings of the International Conference on Parsing Technologies*.

Gabriel Stanovsky, Noah A Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Elias Stengel-Eskin, Tzu-ray Su, Matt Post, and Benjamin Van Durme. 2019. A discriminative neural model for cross-lingual word alignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Sara Stymne, Jörg Tiedemann, and Joakim Nivre. 2014. Estimating word alignment quality for smt reordering tasks. In *Proceedings of the Workshop on Statistical Machine Translation*.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Linguistics*.

Kyle Swanson, Lili Yu, and Tao Lei. 2020. Rationalizing text matching: Learning sparse alignments via optimal transport. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2014. Recurrent neural networks for word alignment model. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Jörg Tiedemann. 2005. Optimization of word alignment clues. *Natural Language Engineering*.

Jörg Tiedemann. 2014. Rediscovering annotation projection for cross-lingual parser induction. In *Proceedings of the International Conference on Computational Linguistics*.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Natural Language Learning*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*.

David Vilar, Maja Popović, and Hermann Ney. 2006. AER: Do we need to "improve" our alignments? In *Proceedings of the International Workshop on Spoken Language Translation*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the International Conference on Computational Linguistics*.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Hainan Xu, Shuoyang Ding, and Shinji Watanabe. 2019. Improving end-to-end speech recognition with pronunciation-assisted sub-word modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013a. A lightweight and high performance monolingual word aligner. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013b. Semi-Markov phrase-based monolingual alignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the International Conference on Human Language Technology Research*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *arXiv preprint*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. End-to-end neural word alignment outperforms GIZA++. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Zhirui Zhang, Shuangzhi Wu, Shujie Liu, Mu Li, Ming Zhou, and Tong Xu. 2019. Regularizing neural machine translation by target-bidirectional agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Model	Prec. %	Rec.%	F_1 %
Baseline			
Yao et al. (2013a)	91.3	82.0	86.4
Yao et al. (2013b)	90.4	81.9	85.9
Sultan et al. (2014)	**93.5**	82.6	87.6
Ours			
mBERT	87.0	89.0	88.0
Ours-Multilingual	87.0	89.3	88.1
Ours-Supervised	87.2	**89.8**	**88.5**

Table 10: Our model is also effective in monolingual alignment settings.

A Implementation Details

We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2e-5 and the batch size is set to 8. Following Peters et al. (2019), we set α to 1.5 for α-entmax. The threshold c is set to 0 for α-entmax and 0.001 for *softmax* and optimal transport. We train our models on one 2080 Ti for one epoch and it takes 3 to 24 hours for the model to converge depending on the size of the dataset. We evaluate the model performance using Alignment Error Rate (AER).

B Analysis

In this section, we conduct more analyses of our models.

Monolingual Alignment. We also investigate how our models perform in monolingual alignment settings. Previous methods (MacCartney et al., 2008; Yao et al., 2013a,b; Sultan et al., 2014) typically exploit external resources such as WordNet to tackle the problem. As shown in Table 10, mBERT can outperform previous methods in terms of recall and F_1 without any fine-tuning. Our multilingually fine-tuned model can achieve better recall and slightly better F_1 score than the vanilla mBERT model, and fine-tuning our model with supervised signals can achieve further improvements.

Sensitivity Analysis. We also conduct a sensitivity analysis on the threshold c for our *softmax* alignment extraction method. As shown in Table 11, our method is relatively robust to this threshold. In particular, after fine-tuning, the AERs change within 0.5% when varying the threshold.

Comparisons with IterMax. IterMax is the best alignment extraction method in SimAlign (Sabet et al., 2020). The results in the main paper have demonstrated that our alignment extraction methods are able to outperform IterMax. In Figure 4, we can see that the IterMax algorithm tends to sacrifice precision for a small improvements in recall, while our model can generate more accurate alignments.

Model	c.	De-En	Fr-En	Ro-En	Ja-En	Zh-En
mBERT	1e-6	17.3	6.0	27.2	45.2	18.9
	1e-5	17.3	5.9	27.4	45.1	18.6
	1e-4	17.3	5.7	27.6	45.3	18.3
	1e-3	17.4	5.6	27.9	45.6	18.1
	1e-2	17.7	5.6	28.4	45.8	18.2
	1e-1	18.1	5.6	28.9	46.3	18.3
	5e-1	18.4	5.6	29.5	47.0	18.7
Ours-Multilingual	1e-6	15.4	4.6	22.7	38.2	14.1
	1e-5	15.4	4.5	22.7	38.1	14.0
	1e-4	15.3	4.5	22.6	37.9	13.9
	1e-3	15.3	4.4	22.6	37.9	13.8
	1e-2	15.3	4.3	22.7	37.9	13.8
	1e-1	15.4	4.3	22.8	38.0	13.8
	5e-1	15.4	4.2	23.0	38.2	13.9

Table 11: Our *softmax* alignment extraction method is relatively robust to the threshold c.

Ablation Studies on Training Objectives. Table 12 presents more ablation studies on our training objectives. We can see that the self training objective is the most effective one, with the translation language modeling objective being the second and the parallel sentence identification objective being the third. The masked language modeling objective can sometimes hurt the model performance, possibly because of the translation language modeling objective.

Experiments on More Language Pairs. We also test our alignment extraction methods on other language pairs following the setting of Sabet et al. (2020) without fine-tuning as shown in Table 13.[2]

More Qualitative Examples. In addition to the examples provided in the main text, we also present some randomly sampled samples in Figure 5. We can clearly see that our model learns more aligned representations than the baseline model.

---

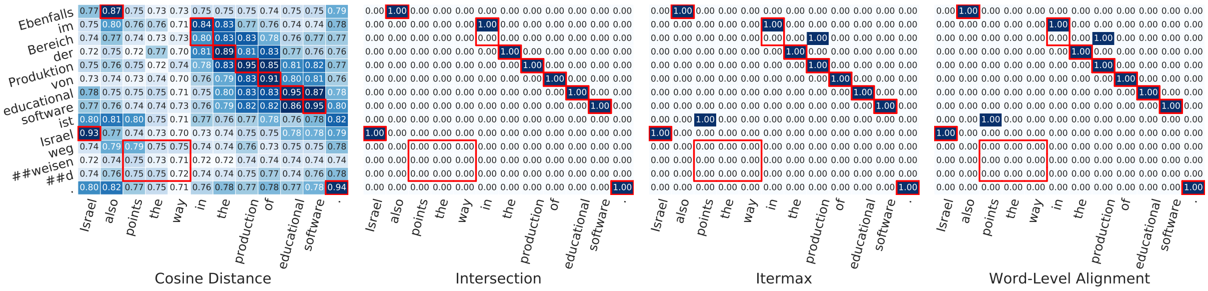
[2] Their English-Persian dataset is unavailable at the time of writing the paper.

Model	Objective	De-En	Fr-En	Ro-En	Ja-En	Zh-En
Ours-Bilingual						
α-entmax	*All*	16.1	4.1	23.4	38.6	15.4
	All w/o MLM	15.6	4.2	23.3	38.8	15.1
	All w/o TLM	16.4	4.3	23.7	40.1	15.3
	All w/o SO	17.8	4.7	23.9	39.4	16.3
	All w/o PSI	16.5	4.2	23.1	38.5	15.4
softmax	*All*	15.6	4.4	23.0	38.4	15.3
	All w/o MLM	15.5	4.2	23.2	38.9	14.9
	All w/o TLM	15.9	4.5	23.7	40.1	15.1
	All w/o SO	17.4	4.7	23.2	38.6	16.3
	All w/o PSI	15.6	4.3	23.1	38.8	15.4
Ours-Multilingual						
α-entmax	*All*	15.4	4.1	22.9	37.4	13.9
	All w/o MLM	15.1	4.2	22.8	37.8	13.7
	All w/o TLM	16.4	4.4	23.3	39.7	14.4
	All w/o SO	17.5	4.6	23.6	40.0	15.6
	All w/o PSI	15.5	3.9	23.0	38.2	14.1
softmax	*All*	15.3	4.4	22.6	37.9	13.6
	All w/o MLM	15.3	4.4	22.8	38.6	13.7
	All w/o TLM	15.5	4.7	22.9	39.7	14.0
	All w/o SO	16.9	4.8	23.0	39.1	15.4
	All w/o PSI	15.4	4.4	22.7	37.9	13.8

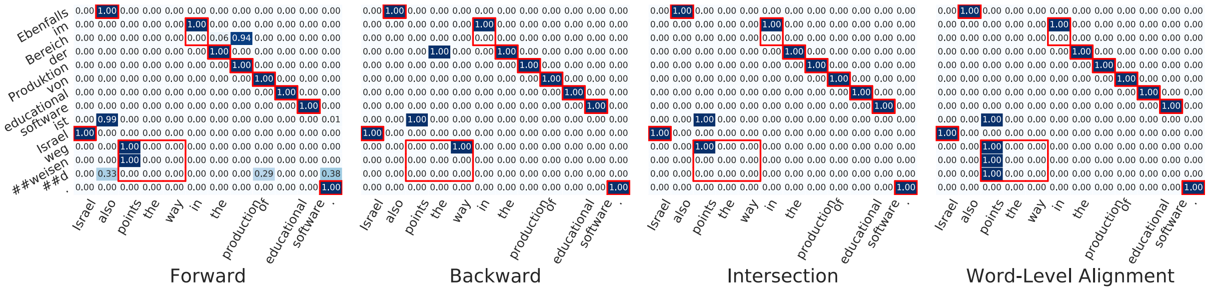
Table 12: Ablation studies on training objectives.

Model	En-Cs	En-Hi
GIZA++	18.2	51.8
SimAlign	13.4	40.2
Ours (*softmax*, c=1e-3)	**12.3**	41.2
Ours (*softmax*, c=1e-5)	12.7	39.5
Ours (*softmax*, c=1e-7)	13.3	**39.2**

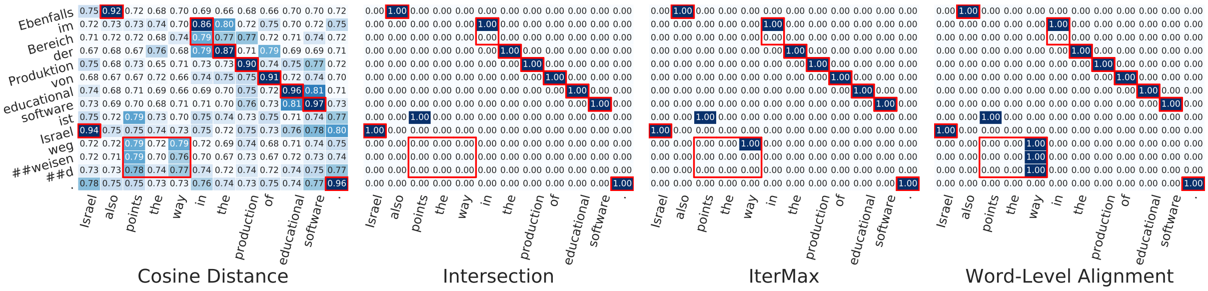
Table 13: Performance on more language pairs.



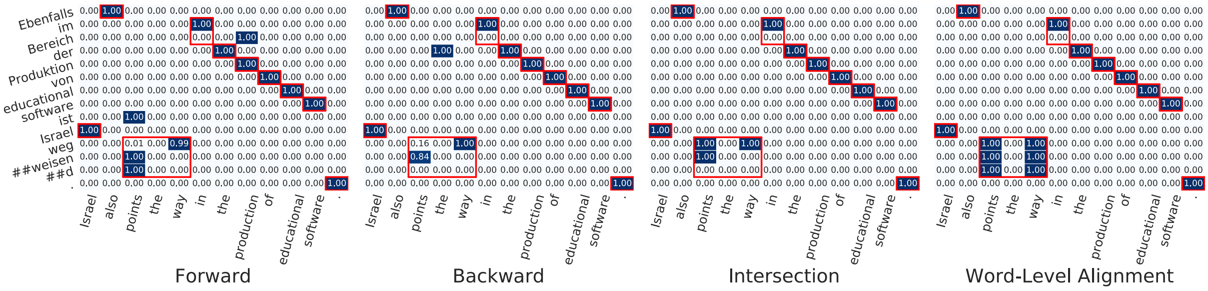
(a) mBERT Itermax



(b) mBERT *softmax*



(c) Fine-tuned IterMax



(d) Fine-tuned *softmax*

Figure 4: Extracting alignments from our model using IterMax(Sabet et al., 2020) and our *softmax* method from the vanilla and fine-tuned mBERT models.

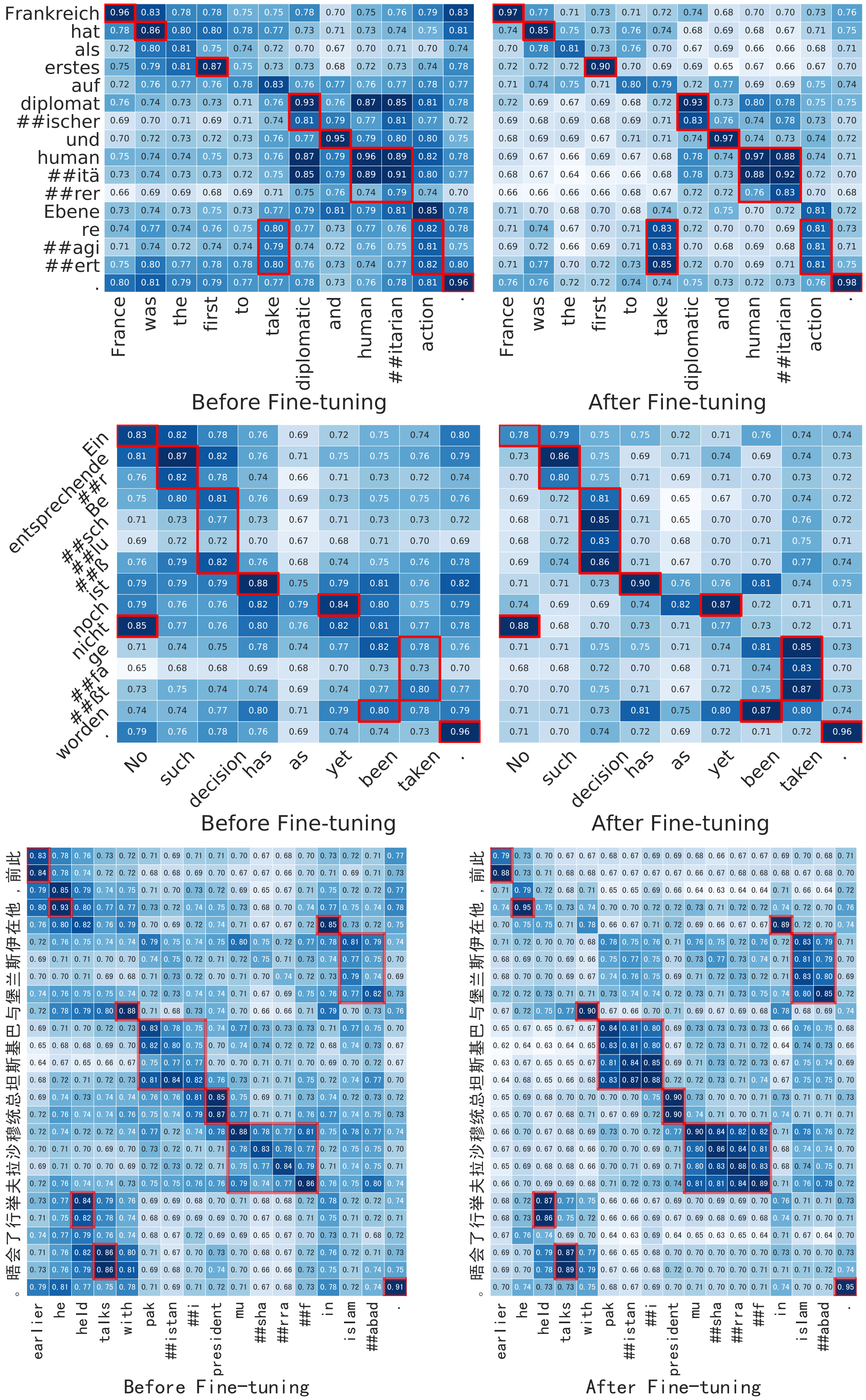


Figure 5: Cosine similarities between subword representations in a parallel sentence pair before and after fine-tuning. Red boxes indicate the gold alignments.

Paraphrases do not explain word analogies

Louis Fournier
Cognitive Machine Learning
CNRS-EHESS-INRIA
PSL Research University
Paris, France
`louis.fournier@polytechnique.edu`

Ewan Dunbar
Cognitive Machine Learning
CNRS-EHESS-INRIA
PSL Research University
Paris, France
and
University of Toronto
Toronto, Canada
`ewan.dunbar@utoronto.ca`

Abstract

Many types of distributional word embeddings (weakly) encode linguistic regularities as directions (the difference between *jump* and *jumped* will be in a similar direction to that of *walk* and *walked*, and so on). Several attempts have been made to explain this fact. We respond to Allen and Hospedales' recent (ICML, 2019) theoretical explanation, which claims that word2vec and GloVe will encode linguistic regularities whenever a specific relation of *paraphrase* holds between the four words involved in the regularity. We demonstrate that the explanation does not go through: the paraphrase relations needed under this explanation do not hold empirically.

1 Introduction

The study of linguistic regularities in distributional word embeddings—that the difference vector calculated between the vectors *jump* and *jumped* shows a similar direction to that of *walk* and *walked*, and so on—has been both stimulating and controversial. While a number of such regularities appear to hold, across a number of different kinds of embeddings, the standard 3COSADD analogy test used to measure the presence of these regularities has come under fire for confounding analogical regularities with unrelated properties of semantic embeddings. It is thus important to note that several papers have proposed theoretical explanations for why linguistic regularities *should* hold in distributional word embeddings. Particularly in light of the controversies over linguistic regularities, it is important to examine the soundness of these arguments.

Allen and Hospedales (2019) develop such an explanation by linking the semantic definition of an analogy to *paraphrases.* In the sense of Gittens et al. (2017), paraphrases are sets of words which are semantically and distributionally closely equivalent to another word or set of words—for example, *king* may be paraphrased by $\{$*man*, *royal*$\}$. Allen and Hospedales argue that the standard analogy criterion, that *king* - *man* + *woman* = *queen*, is equivalent to a criterion whereby $\{$*king*, *woman*$\}$ paraphrases $\{$*man*, *queen*$\}$. With this in mind, it becomes possible to rewrite the arithmetic analogy criterion in terms of vectors encoding the pointwise mutual information (PMI) between words and their contexts, and to decompose the error in the analogy equality into several components, including a *paraphrase error* term measuring the degree to which the critical paraphrase holds. Making use of an assumption that the word2vec embedding is a linear transformation of the PMI matrix, they argue that results in terms of PMI apply to word vectors. Thus, under their explanation, a major part of success on an analogy $a - a^* + b^* = b$ is due to a, b^* and a^*, b being close distributional paraphrases.

We first review the literature on the analogy test itself, underlining known pitfalls which any explanation of linguistic regularities must navigate. We then show empirically that the relation between the PMI matrix and word2vec embeddings is to some degree linear, which may be enough to satisfy the assumption of Allen and Hospedales (2019). We further examine the proposed decomposition into error terms. We demonstrate that, empirically, these error terms tend to be undefined due to data sparseness, undermining their explanatory force. Most importantly, examining a number of analogies which pass the standard test, we show that the critical paraphrase error term is, contrary to the proposed explanation, very large.[1]

---

[1] Code is available at `www.github.com/bootphon/paraphrases_do_not_explain_analogies`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2129–2134
April 19 - 23, 2021.

2 Related work

Early works proposing explanations of the analogical properties of word embeddings include Mikolov et al. (2013b) and Pennington et al. (2014). A geometrical explanation is proposed by Arora et al. (2016), but this explanation relies on very strong preconditions, notably, that the word vectors be distributed uniformly in space. Ethayarajh et al. (2019) also propose an explanation, providing a link between the PMI and the norm of word embeddings. However, as pointed out by Allen and Hospedales (2019), this explanation, too, rests on strong assumptions. Notably, the words involved in the analogy are required to be coplanar, a property that seems unlikely in light of the lack of parallelism we discuss in the next section.

3 Issues with the test

Issues have arisen with the standard way of measuring linguistic analogies. Levy and Goldberg (2014a), Vylomova et al. (2016), Rogers et al. (2017), and Fournier et al. (2020) all demonstrate that the standard 3CosAdd measure conflates several very different properties of embeddings, simultaneously measuring not only the directional regularities suggested by typical illustrations of vectors in a parallelogram, but also the similarity of individual matched pairs such as *king, man*, as well as the global arrangement of vectors in semantic fields, such as *king, queen, prince, …* versus *man, woman, child, …* in distinct regions of the space. These issues undermine the construct validity of the standard analogy test. This conflation of properties explains certain pathological behaviours of the test (Linzen, 2016; Rogers et al., 2017). In spite of these issues, Fournier et al. (2020) demonstrate, using alternative measures, that linguistic regularities are nevertheless coded by directional similarities. This parallelism is weak, with directions tending to be closer, in the absolute, to being orthogonal than to being parallel, but is present above chance level (unmatched word pairs).

Thus, before turning to Allen and Hospedales (2019), one of a number of theoretical attempts to explain performance on the 3CosAdd objective, we underscore that such demonstrations run the risk of explaining properties of the test which may be of secondary interest, or, conversely, of placing undue emphasis on the role of directional regularities, which have been shown to play only a small role in success on 3CosAdd.

4 Explaining analogies through paraphrases

For a word w_i and a word c_j which can appear in the context of w_i, the pairwise mutual information $PMI(w_i, c_j)$ is defined as $\log \frac{p(w_i,c_j)}{p(w_i)p(c_j)}$. As shown by Levy and Goldberg (2014b), skip-gram word2vec with negative sampling factorizes the PMI: PMI $\approx W^\top \cdot C$, with W and C the word and context embedding matrices of a word2vec model.

For two pairs of words (a, a^*) and (b, b^*) from the same semantic relation, the standard arithmetic analogy test criterion is that $a - a^* + b^* = b$. Writing $\mathcal{W} = \{a, b^*\}, \mathcal{W}_* = \{a^*, b\}$, and $\mathbf{PMI}_x$ the PMI vector of x, Allen and Hospedales (2019) show that is possible to rewrite the arithmetic analogy formula with PMI vectors, and to decompose the error in the equality into five terms as follows:

$$\begin{aligned}\mathbf{PMI}_{b^*} =& \mathbf{PMI}_b + \mathbf{PMI}_{a^*} - \mathbf{PMI}_a \\ &+ \underbrace{\rho^{\mathcal{W},\mathcal{W}_*}}_{\text{Paraphrase error}} + \underbrace{\sigma^{\mathcal{W}} - \sigma^{\mathcal{W}_*}}_{\text{Conditional dependence error}} \\ &+ \underbrace{(\tau^{\mathcal{W}} - \tau^{\mathcal{W}_*})\mathbf{1}}_{\text{Mutual dependence error}}\end{aligned} \tag{1}$$

The error terms are vectors of length $|\mathcal{V}|$ (vocabulary size), with each element j defined as:

$$\begin{aligned}\rho^{\mathcal{W},\mathcal{W}_*} &= \log \frac{p(c_j|\mathcal{W}_*)}{p(c_j|\mathcal{W})} \\ \sigma^{\mathcal{W}} &= \log \frac{p(\mathcal{W}|c_j)}{\prod_{\mathcal{W}} p(w_i|c_j)} \\ \tau^{\mathcal{W}} &= \log \frac{p(\mathcal{W})}{\prod_{\mathcal{W}} p(w_i)}\end{aligned} \tag{2}$$

The authors claim that these terms can be embedded linearly into a word2vec embedding space by multiplying them by the Moore-Penrose pseudo-inverse $C^\dagger$ of the context matrix C. Then with $\mathbf{w}_x$ the word2vec embedding of x, $C^\dagger \cdot \mathbf{PMI}_x \approx \mathbf{w}_x$. Thus we get the final decomposition:

$$\begin{aligned}\mathbf{w}_{b^*} =& \mathbf{w}_b + \mathbf{w}_{a^*} - \mathbf{w}_a + \\ & C^\dagger \left(\rho^{\mathcal{W},\mathcal{W}_*} + \sigma^{\mathcal{W}} - \sigma^{\mathcal{W}_*} - (\tau^{\mathcal{W}} - \tau^{\mathcal{W}_*})\mathbf{1}\right)\end{aligned} \tag{3}$$

The paraphrase error term is claimed to be small for successful analogies. Elaborating on the notation, $\mathcal{W}$ is taken to paraphrase $\mathcal{W}_*$ if, wherever all $w \in \mathcal{W}$ appear together, we observe the same distribution of surrounding words as for $\mathcal{W}_*$. The paraphrase error assesses the similarity of the distributions of words in the context of $\mathcal{W}$ (all words in $\mathcal{W}$ appearing together) versus $\mathcal{W}_*$.

5 Linearity of the link between PMI and word2vec

Though it is true that there is a relation between the word2vec matrices $W^\top \cdot C$ and the PMI matrix, in practice the link is more complicated than simple linear matrix factorization, due in part to the training tricks described in Mikolov et al. (2013a). The result of Allen and Hospedales (2019) requires that the embedding from PMI vectors to word2vec embeddings be "linear enough" for $C^\dagger\cdot$ PMI to approximate W.

To assess this, we use the *text8* corpus [2] both to train word2vec embeddings [3] and to estimate a PMI matrix. We replace infinite values in the PMI matrix by 0. In Figure 1a, we show the distribution of the Pearson correlation coefficient (assessing the presence of a linear relation) between the word2vec embedding and the corresponding row of $C^\dagger\cdot$ PMI for the top ten thousand words in the corpus. As can be seen from the figure, the correlation tends to be between 0.5 and 0.8. For instance in Figure 1b, the word2vec embedding for *king* is plotted against the row of $C^\dagger\cdot$ PMI corresponding to *king*.

While the relation is not perfectly linear—many words have a correlation of around 0.55, far lower than that of *king*—the empirical relations shown here leave open the possibility that it may indeed be "sufficiently linear" to be taken for granted. However, while linearity is necessary for the result of Allen and Hospedales (2019) to go through, it is not sufficient. In the next section, we assess the critical question of whether the paraphrase error is small enough to serve as an explanation for the success of linguistic analogies.

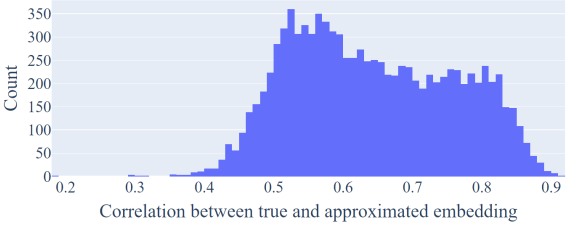


Figure 1a: Histogram of the Pearson correlations between true and approximated word2vec embeddings for the top ten thousand words in the *text8* corpus. The mean value is 0.643 and the variance is 0.014.

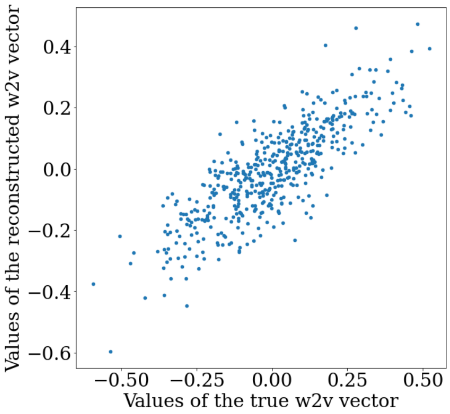


Figure 1b: Plot of the values of the word2vec embedding for *king*, versus coefficients for the row of $C^\dagger\cdot$ PMI corresponding to *king*, for word2vec matrices trained on the same corpus (*text8*). The Pearson correlation is one of the best possible at 0.825.

6 Empirical analysis of the error terms

We now seek to examine the proposed explanation by calculating the proposed error terms empirically. However, in practice, many of the terms are undefined, since they rely on cooccurrences unattested in practical corpora. The most extreme situation occurs when the two words of a paraphrase $\mathcal{W} = \{w_1, w_2\}$ are *never* present in the same context window in the corpus. We found that only 16% of the paraphrase sets associated with the BATS analogy set (Gladkova et al., 2016)—for example, *king, woman*—were present together in the *text8* corpus in a context window of length five. We refer to such paraphrase sets as "well-defined" with respect to the corpus. The problem of zero co-occurrence counts was anticipated by Allen and Hospedales (2019), who propose to restrict their analysis to the case where the context window is sufficiently large that all relevant terms are well defined. We stress that our trained word2vec vectors are also trained with a context window of five, and yield expected levels of performance on the BATS analogy test, despite having access to little training data on which to model co-occurrences such as *king, woman*, *queen, man*, and so on.

At a minimum, if the proposed explanation holds, the cases for which the error terms are empirically well-defined should show signs of the paraphrase

---

[2] A text dataset composed of 100 million characters from Wikipedia: (Mahoney, 2006).

[3] Skip-gram architecture with negative sampling (1 word), negative sampling exponent equal to 1, no undersampling of common words, and a high dimension size of 500. These parameters allow us to be as close as possible to a direct factorization of the PMI matrix.

Category	**I01**	**I02**	**I05**	**I06**	**I07**	**I08**	**I09**	**I10**	**D02**	**D03**	**D05**	**D08**	**D10**	**E01**	**E02**
Paraphrase error norm	177	153	111	127	126	124	138	97	102	122	130	110	107	124	176
Dependence errors sum norm	1006	938	867	903	957	883	952	908	856	893	514	585	699	749	848
All errors sum norm	1032	957	878	917	970	897	966	916	864	905	539	602	710	765	875

Category	**E03**	**E04**	**E05**	**E08**	**E09**	**E10**	**L02**	**L03**	**L04**	**L05**	**L06**	**L07**	**L08**	**L09**	**L10**
Paraphrase error norm	162	176	155	229	179	190	197	189	209	206	133	169	185	175	432
Dependence errors sum norm	866	797	519	739	910	833	642	982	907	1103	921	995	1044	1017	1302
All errors sum norm	889	822	553	778	933	865	683	1007	939	1131	937	1016	1066	1040	1416

Table 1: L2 norms of the error terms in 1, following our implementation.

Category	**I01**	**I02**	**I05**	**I06**	**I07**	**I08**	**I09**	**I10**	**D02**	**D03**	**D05**	**D08**	**D10**	**E01**	**E02**
Average rank	7762K	7589K	7759K	8744K	8160K	6454K	7028K	11889K	31952K	19558K	7857K	1506K	2556K	4394K	9507K
Median rank	1630K	2195K	3055K	3239K	2530K	4090K	3004K	4535K	6754K	3564K	3260K	1506K	2556K	2117K	1622K

Category	**E03**	**E04**	**E05**	**E08**	**E09**	**E10**	**L02**	**L03**	**L04**	**L05**	**L06**	**L07**	**L08**	**L09**	**L10**
Average rank	1305K	5611K	9192K	727K	8421K	11946K	52183K	1857K	12687K	6747K	2475K	7727K	4502K	4679K	16871K
Median rank	695K	1703K	1426K	854K	1908K	169K	52182K	1261K	2460K	1343K	2255K	2136K	1549K	1 739K	785K

Table 2: For an analogy equivalent to two paraphrases $\mathcal{W}$ and $\mathcal{W}_*$, the rank of $\mathcal{W}_*$ in the list of the closest paraphrases to $\mathcal{W}$ with respect to the L2 norm of the paraphrase error vector. 7762K means a rank of 7762000, rounded to the nearest thousand.

error being relatively small. We now detail how we implemented the error terms in cases for which they were well-defined. We count co-occurrences $N(w_i, w_j, w_k)$ in *text8* for all triplets of words w_i, w_j, w_k, with w_k at the center of the context window, and $W = \{w_i, w_j\}$ any paraphrase, both occurring anywhere within a context window of width five. We restrict analysis to the ten thousand most frequent word types w_i and w_j, yielding 10^8 possible paraphrases.[4] We use the relative frequencies as estimators of $p(w_k|\{w_i, w_j\})$ and $p(\{w_i, w_j\}|w_k)$, and marginalize to obtain $p(w_i|w_k)$, $p(\{w_i, w_j\})$ and $p(w_k)$. The error terms follow. Since this can still lead to ill-defined elements, we replace $log(+\infty)$ and $log(0)$ by $+/-log(\epsilon)$, with $\epsilon = 10^{-15}$ (within reason, the value of ϵ is immaterial). We also replace $log(0/0)$ with 0.

Table 1 shows the mean and median values of the L2 norms of the paraphrase error vectors across several categories of the BATS dataset. We compare them with the sum of the four dependence error terms (the dependence error reflects statistical dependencies within $\mathcal{W}$ and $\mathcal{W}_*$ irrelevant to the analogy), as well as the sum of all five error terms (equal to the difference between the PMI of $\mathcal{W}$ and $\mathcal{W}_*$).The paraphrase error is indeed smaller than the other error terms. However, as we now show, the paraphrase error is not small *enough* to contribute substantially to the success of analogies.[5]

Take the norm of the paraphrase error vector ρ as a measure of the divergence in the PMI between two paraphrases. For an analogy with associated paraphrases $\mathcal{W}$ and $\mathcal{W}_*$, we assess how many paraphrases are closer to $\mathcal{W}$ than to $\mathcal{W}_*$ by calculating the rank of the norm of $\rho^{\mathcal{W},\mathcal{W}_*}$ among all $\rho^{\mathcal{W},\mathcal{X}}$, where $\mathcal{X}$ spans over all pairs of words constructible from the top ten thousand most frequent words in the corpus. To do so, we define a Paraphrase Conditional Information matrix (PCI). For $W_{ij} = \{w_i, w_j\}$ and w_k, we define $PCI(l_{ij}, k)$, the value at column l_{ij} and row k to be $log(p(W_{ij}|w_k))$, where with l_{ij} is a unique index associate with tuple (i, j). We compute only the positive PCI, to obtain a sparse matrix. The difference between two PCI columns is a paraphrase error vector, and their Euclidean distance is the norm of the paraphrase error.

We now compute, for each analogy, the distance between the PCI column of $\mathcal{W}$ and every other column (paraphrase) of the PCI matrix. We calculate the rank of the true analogy pair $\mathcal{W}_*$. Given that the analogy test generally succeeds in picking out b as being the most similar to $a - a^* + b^*$ out of the entire vocabulary (modulo Linzen 2016), we would expect that, for successful analogies, the paraphrase error for the true analogy would be among the highest, if small paraphrase error were the explanation for success. Table 2 displays the mean of this rank within each BATS category. The rank is extremely low (in the millions), making the paraphrase error in true analogies far too high to be the explanation for their success.[6]

---

[4] w_k is allowed to vary over all of the types included in the training for word2vec, of which there are 71290. Thus, for each paraphrase, the error vectors have 71290 elements, one for each vocabulary word.

[5] We note also that the error values seem relatively consistent between categories, while success on the analogy test varies differ greatly between categories.

[6] Limiting the search to the paraphrases composed by at least one of the words of W_* still results in a very low rank

7 Conclusion

Recent work has shown that, in spite of the standard analogy test's confound with simple vector similarity, distributional word vectors genuinely do encode linguistic regularities as directional regularities above and beyond vector similarity (Fournier et al., 2020), further research is warranted into the mechanisms by which distributional word embeddings come to show these regularities. However, the analysis of analogies as paraphrases does not hold up as an explanation of performance on the analogy test—nor would an explanation of performance on the 3COSADD analogy test be a satisfying result, since the test is not a useful measure to begin with.

Acknowledgments

This work was funded in part by the European Research Council (ERC-2011-AdG-295810 BOOTPHON), the Agence Nationale pour la Recherche (ANR-17-EURE-0017 Frontcog, ANR-17-CE28-0009 GEOMPHON, ANR-10-IDEX-0001-02 PSL*, ANR-19-P3IA-0001 PRAIRIE 3IA Institute, ANR-18-IDEX-0001 U de Paris, ANR-10-LABX-0083 EFL) and grants from CIFAR (Learning in Machines and Brains), Facebook AI Research (Research Grant), Google (Faculty Research Award), Microsoft Research (Azure Credits and Grant), and Amazon Web Service (AWS Research Credits).

References

Carl Allen and Timothy Hospedales. 2019. Analogies Explained: Towards Understanding Word Embeddings. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 223–231, Long Beach, California, USA. PMLR.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016. A Latent Variable Model Approach to PMI-based Word Embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Towards Understanding Linear Word Analogies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3253–3262, Florence, Italy. Association for Computational Linguistics.

Louis Fournier, Emmanuel Dupoux, and Ewan Dunbar. 2020. Analogies minus analogy test: measuring regularities in word embeddings. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 365–375, Online. Association for Computational Linguistics.

Alex Gittens, Dimitris Achlioptas, and Michael W. Mahoney. 2017. Skip-gram âˆ’ Zipf + uniform = vector additivity. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 69–76, Vancouver, Canada. Association for Computational Linguistics.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-Based Detection of Morphological and Semantic Relations with Word Embeddings: What Works and What Doesn't. In *Proceedings of the NAACL-HLT SRW*, pages 47–54, San Diego, California, June 12-17, 2016. ACL.

Omer Levy and Yoav Goldberg. 2014a. Linguistic Regularities in Sparse and Explicit Word Representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 171–180, Ann Arbor, Michigan. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Tal Linzen. 2016. Issues in Evaluating Semantic Spaces Using Word Analogies. In *Proceedings of the First Workshop on Evaluating Vector Space Representations for NLP*. Association for Computational Linguistics.

Matt Mahoney. 2006. Large text compression benchmark: About the test data (web page).

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of NAACL-HLT 2013*, pages 746–751, Atlanta, Georgia, 9–14 June 2013.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Anna Rogers, Aleksandr Drozd, and Bofang Li. 2017. The (Too Many) Problems of Analogical Reasoning with Word Vectors. In *Proceedings of the 6th Joint*

---

for W_*.

Conference on Lexical and Computational Semantics (SEM 2017)*, pages 135–148.

Ekaterina Vylomova, Laura Rimell, Trevor Cohn, and Timothy Baldwin. 2016. Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1671–1682, Berlin, Germany. Association for Computational Linguistics.

An Empirical Study on the Generalization Power of Neural Representations Learned via Visual Guessing Games

Alessandro Suglia[1], **Yonatan Bisk**[2], **Ioannis Konstas**[1], **Antonio Vergari**[3],
Emanuele Bastianelli[1], **Andrea Vanzo**[1], and **Oliver Lemon**[1]
[1]Heriot-Watt University, Edinburgh, UK
[2]Carnegie Mellon University, Pittsburgh, USA
[3]University of California, Los Angeles, USA
[1]{as247,i.konstas,e.bastianelli,a.vanzo,o.lemon}@hw.ac.uk
[2]ybisk@cs.cmu.edu, [3]aver@cs.ucla.edu

Abstract

Guessing games are a prototypical instance of the "learning by interacting" paradigm. This work investigates how well an artificial agent can benefit from playing guessing games when later asked to perform on novel NLP downstream tasks such as Visual Question Answering (VQA). We propose two ways to exploit playing guessing games: 1) a supervised learning scenario in which the agent learns to mimic successful guessing games and 2) a novel way for an agent to play by itself, called Self-play via Iterated Experience Learning (SPIEL). We evaluate the ability of both procedures to generalise: an in-domain evaluation shows an increased accuracy (+7.79) compared with competitors on the evaluation suite CompGuessWhat?!; a transfer evaluation shows improved performance for VQA on the TDIUC dataset in terms of harmonic average accuracy (+5.31) thanks to more fine-grained object representations learned via SPIEL.

1 Background & Related Work

Learning a language requires interacting with both the environment and other agents (Bisk et al., 2020). Language games represent one common example of this (Wittgenstein et al., 1953), as seen by the important role of play in L1 child language acquisition (Hainey et al., 2016) as well as L2 learners (Godwin-Jones, 2014).

Among the language games defined in the literature (Steels, 2015), *guessing games* represent the first step in a curriculum for language learning. For example, in GuessWhat?! (de Vries et al., 2017), two agents interact with each other: a *Questioner* generates questions aimed at finding a hidden object in the scene and an *Oracle*, aware of the target object, answers the questions supporting the Questioner in playing the game. Different from other language games (Das et al., 2017), guessing games have a specific goal which represents a clear incentive for learning. In addition, they require that the *Questioner* masters both natural language generation and understanding with a focus on object categories and attributes. For humans, concepts learned in this way are generic and generalisable to new tasks and domains where grounded reasoning is important (Hampton, 1979). However, *how well can AI agents generalise with concepts acquired from visual guessing games?*

The literature has not explored if representations built from self-play are transferable, focusing instead on large scale self-supervised learning. For instance, large scale image captioning datasets have been used to train multi-modal Transformers (Lu et al., 2019; Li et al., 2019; Tan and Bansal, 2019; Chen et al., 2019). Multi-task learning (Lu et al., 2020) has been used to leverage the diversity of training signals provided combining datasets, but only for discriminative tasks. While some dialogue work (Cogswell et al., 2020) aims to bootstrap a conversing agent from VQA datasets, most work on GuessWhat?! (de Vries et al., 2017; Shekhar et al., 2019; Strub et al., 2017) has designed bespoke models for the task, ignoring the utility of this dataset for other Vision+Language tasks.

We propose self-play as a mechanism for learning general grounded representations. We seed our approach with the *GuessWhat?!* corpus of questions and objects, and demonstrate how to generalise to other downstream tasks. We propose two different strategies to exploit these data. First, a supervised learning phase is undertaken to learn a Questioner and Oracle model able to play guessing games. Second, the trained agents can be used to play guessing games on images requiring only object annotations as supervision. We show that an agent trained on GuessWhat?! dialogues can use self-play to adapt to new and harder tasks. Specifically, we investigate models' gener-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2135–2144
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

alisation performance and quality of the learned representations on the CompGuessWhat?! benchmark (Suglia et al., 2020), a more extensive evaluation suite for GuessWhat?!. Furthermore, we study how the learned representation help solve VQA on the dataset TDIUC (Kafle and Kanan, 2017). We show overall comparable performance with state-of-the-art models and improvements for specific question types that require object attribute information to be answered correctly.

2 Methodology

Our proposed transfer/fine-tuning procedure requires a training set of guessing games D_g from which we learn a Questioner Q and an Oracle O via supervised learning. Given a set of images $\mathcal{I}$, it is possible to use the trained models Q and O to run the self-play procedure for n epochs obtaining the model Q^n. Finally, given a downstream task t and an associated dataset $\mathcal{D}_t$ based on images from $\mathcal{I}$, we use Q^n's parameters as initialisation for the training procedure on $\mathcal{D}_t$.

To apply this procedure, both the *Questioner* and the *Oracle* require a *multi-modal encoder* Γ able to generate d-dimensional representations for the textual tokens $\mathbf{h}_t$, for the objects $\mathbf{h}_o$, as well as fusing the visual and textual modalities in a representation of the current context $\mathbf{h}_c$. After the self-play procedure, only the encoder Γ of the model Q^n is used in the fine-tuning process on the downstream task t using the dataset $\mathcal{D}_t$. It is important to underline that the presented self-play procedure does not depend on a specific implementation of the multi-modal encoder Γ. A possible implementation is presented in Section 2.4 and it is used in the experimental evaluation of this paper.

2.1 Oracle design

The Oracle task is cast as a Visual Question Answering (VQA) task conditioned on the image I, the current question q and on the target object $\hat{o}$. We follow common practice in vocabulary-based VQA (Antol et al., 2015) and we treat the problem as a multi-class classification task over the classes $\{Yes, No, N/A\}$. We use $\mathbf{h}_c$ as input to a multi-layer feedforward neural network to obtain a probability distribution over the label set.

2.2 Questioner design

The Questioner must play two roles: question generation and target object prediction (de Vries et al., 2017). It is beneficial to jointly learn the two tasks because the representations learned by each task are complementary. In addition, they better encode attributes, which favours better generalisation to unseen object categories (Suglia et al., 2020).

To solve the two specific tasks in a multi-task fashion, we design two different heads on top of the shared encoder Γ: 1) the *guesser head*, produces a probability distribution over every object o_i using the encoded representations $\mathbf{h}_{o_i}$ passed through an MLP; 2) the *generator head*, a multi-modal decoder, also implemented as an MLP, which predicts a probability distribution over the vocabulary V given the context representation generated by Γ.

We include two losses in our model: 1) the negative log-likelihood of the probability associated by the guesser head with the target object $\hat{o}$ (Shekhar et al., 2019); 2) a sequence-to-sequence cross-entropy loss (Sutskever et al., 2014) for the generated question tokens. Unlike previous work that trains a separate module to learn to stop (Shekhar et al., 2018), we add a special token `[STOP]` to the input data so that it learns when to stop more efficiently as part of the question generation task.

Training an agent to solve tasks of different complexity and size is challenging. The procedure presented in (Shekhar et al., 2019) alternates between tasks, updating the hardest task more often. For this technique, finding the right schedule is cumbersome and requires fine-tuning. We rely on a more systematic training procedure based on random dataset-proportional batch sampling inspired by (Sanh et al., 2019). This represents a hard-parameter sharing multi-task training procedure that avoids interference between tasks and favours a more stable training, which mitigates catastrophic forgetting (French, 1999).

2.3 Self-Play via Iterated Experience Learning (SPIEL)

Inspired by iterated learning (Kirby et al., 2014), we design a process by which the Questioner learns from games previously generated by other instances of the Questioner agent. We call our training procedure *Self-play via Iterated Experience Learning* (SPIEL).

In SPIEL, described in Algorithm 1, we assume access to a set of images $\mathcal{I}$ and the bounding boxes $\mathcal{O}_I$ of the objects therein.[1] In every gameplay, there

---

[1] Object annotations intended as either gold bounding boxes or predicted bounding boxes from an object detector.

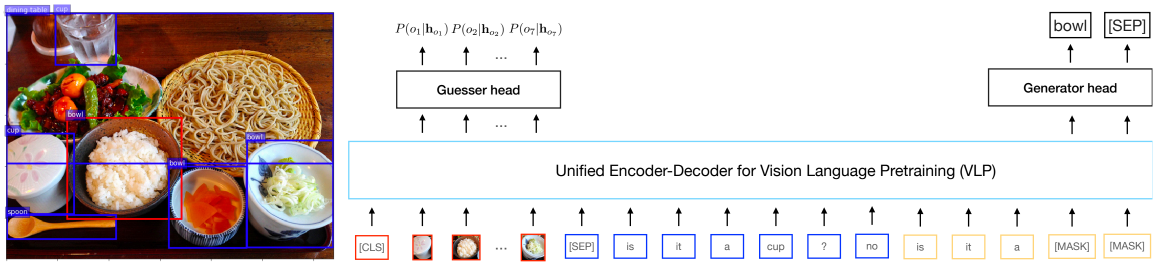


Figure 1: We use the single-stream VLP model as a backbone multi-modal encoder for our task. The visual features tokens (marked in red) are the FastRCNN features associated with the objects in the image, the history tokens (marked in blue) and the tokens to be generated (marked in yellow) are given in input to the model. A *Guesser head* uses the learned contextual object representations to generate a probability distribution over the objects $P(o_i|\mathbf{h}_{o_i})$, whereas the *Generator head* is used to incrementally predict the masked tokens.

Algorithm 1 SPIEL: Self-Play via Iterated Experience Learning

```
procedure SELF_PLAY($Q_0, O, \mathcal{I}, n$)
    $\mathcal{D}_q \leftarrow$ READ_GOLD_GAMES()
    $\mathcal{E}_g \leftarrow []$                    ▷ Initialise the experience buffer
    for $e \leftarrow 1, n$ do
        ▷ Interactive phase
        $Q \leftarrow Q^e$                         ▷ load latest weights
        $\mathcal{G}^e \leftarrow$ GENERATE_GAMES($\mathcal{I}$)
        $\mathcal{G}^e \leftarrow$ PLAY_GAMES($Q, O, \mathcal{G}^e$)
        APPEND($\mathcal{E}_g, \mathcal{G}^e$)
        $\mathcal{D}^e_g \leftarrow []$
        ▷ Transmission phase
        for $i \leftarrow 0, len(\mathcal{E}_g)$ do
            $g \leftarrow \mathcal{E}_g[i]$             ▷ Priority to the latest games
            if IS_VALID_GAME($g$) then
                APPEND($\mathcal{D}^e_g, g$)
            if LEN($\mathcal{D}^e_g$) == LEN($\mathcal{D}_q$) then break
        ▷ Learning phase
        $Q^{e+1} \leftarrow$ TRAIN($Q, \mathcal{D}_q, \mathcal{D}^e_g$)
```

is a Questioner Q and an Oracle O, initialised with agents Q^0 and O, respectively, that were trained with Supervised Learning using gold successful dialogues.[2] We consider every iteration e of the algorithm as a *self-play* epoch. In a single self-play epoch, we alternate 3 phases:

Interactive phase: the agents play guessing games with novel combinations of image and target object. The generated dialogue can be successful if the predicted target object is equal to the target object. Every played dialogue is stored in an experience buffer $\mathcal{E}_g$.

Transmission phase: in this phase the datasets for the multi-task learning procedure for the Questioner are created. The generator head dataset $\mathcal{D}_q$ is fixed in advance while the dataset for the guesser head $\mathcal{D}^e_g$ is created from the experience buffer $\mathcal{E}_g$ by selecting the *unique* and *valid* dialogues.

Learning phase: the same multi-task learning procedure used in the supervised learning phase is used to fine-tune the Questioner parameters using the datasets $\mathcal{D}^e_g$ and $\mathcal{D}_q$ collected for the current epoch e. This procedure is repeated n times or until a halting condition is reached (e.g. early stopping based on validation metric).

See Appendix A.1 for implementation details. At the end of the SPIEL procedure, we obtain the model Q^n whose parameters can be reused in other tasks. Particularly, we use the parameters of Q^n's shared encoder Γ as initialisation for the fine-tuning on the downstream task t using dataset $\mathcal{D}_t$.

2.4 Implementation

We implement a shared multi-modal encoder Γ using VLP (Zhou et al., 2020), a single-stream multi-modal Transformer for captioning depicted in Figure 1. During the GuessWhat?! fine-tuning, we extend VLP by including dialogue context in the input together with the features associated with the objects in the image. We learn two new segment ids to represent the question/answer exchanges in the dialogue, as described in (Wolf et al., 2019). The question is generated by incrementally replacing `[MASK]` tokens until the end of sequence is generated. See Appendix A.2 for more details. SPIEL training is run on a set of images $\mathcal{I}$ from GuessWhat?! and TDIUC dataset with corresponding object annotations. We make sure that GuessWhat?! test images are not contained in $\mathcal{I}$. This is not an issue for TDIUC test images because the downstream task annotations (QA pairs) are not used by the model during this phase. Once the model has been trained with SPIEL, we use the parameters of the shared encoder Γ as a backbone for a VQA model that is fine-tuned on the TDIUC dataset.

---

[2] The Oracle is fixed during this learning procedure.

3 Experimental Evaluation

To assess the generality of our learned representations, we include two evaluation paradigms: 1) *in-domain evaluation* and 2) *transfer evaluation*. We evaluate several variants of our model: 1) VLP-SL: VLP-based model trained on GuessWhat?! data using multi-task learning; 2) SPIEL-gs: VLP-SL model fine-tuned with our SPIEL procedure where the generator head uses only gold successful games (gs); 3) SPIEL-gm: same as 2) but both successful and failed gold games are used by the generator head. In both SPIEL variants, the guesser head is trained using failed and successful generated games because it is important for the guesser head to be exposed to both types of signal to learn a more robust policy. We decided to investigate the two variants SPIEL-gs and SPIEL-gm to get more insights about the effect that successful and failed games have on the generator head ability to produce effective dialogues.

3.1 In-domain evaluation

We use the CompGuessWhat?! evaluation suite (Suglia et al., 2020) to assess the ability of the Questioner to play guessing games and learn visually grounded representations in the process. It complements an evaluation based only on gameplay accuracy (de Vries et al., 2017) with 2 auxiliary tasks: target object 1) attribute-prediction expressed in terms of abstract attributes (A), situated-attributes (SO), abstract+situated attributes (AS), and location attributes (L); 2) zero-shot gameplay with near-domain accuracy (ND) and out-of-domain accuracy (OD). Table 1 shows the comparison with previous state-of-the-art models on this benchmark such as de Vries et al. (2017) (DV-⋆) and Shekhar et al. (2019) (GDSE-⋆). VLP-SL has a greater advantage in terms of representation power compared to previous models. This is reflected in all the tasks of the CompGuessWhat?! evaluation. Particularly, we see better performance even for the zero-shot gameplay (ND: +5.6, OD: +15.2). This is because VLP associates a vector of probabilities that represents a distribution over the VisualGenome object classes with every object. This helps VLP to cope with the issue of unseen objects and helps the model to generalise. Learning to play is key to gameplay performance, leading to an increase of +4.4 over VLP-SL and +7.9 over GDSE-CL. In this setup, the difference between the versions SPIEL-gs and SPIEL-gm is very minimal (0.1). However, when analysed in more detail, we can see that training the questioner with gold successful data only improves attribute prediction while using mixed data improves overall generalisation in the zero-shot evaluation.

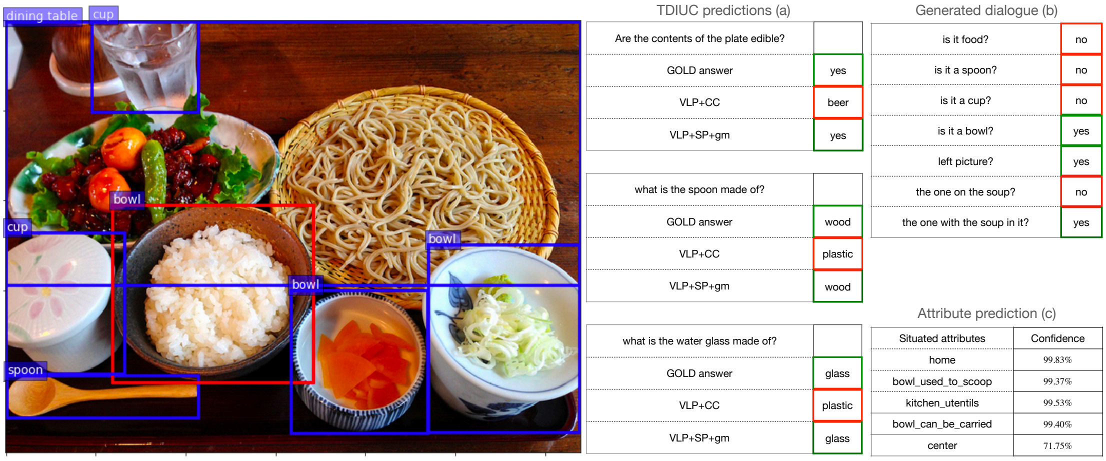
		Attribute Pred.				ZShot		Score
Models	Acc.	A	SO	AS	L	ND	OD	
Random	15.8	15.1	0.1	7.8	2.8	16.8	18.6	13.3
DV-SL	41.5	46.8	39.1	48.5	42.7	31.3	28.4	38.5
DV-RL	53.5	45.2	38.9	47.2	43.5	43.9	38.7	46.2
GDSE-SL	49.1	59.9	47.6	60.1	48.3	29.8	22.3	43.0
GDSE-CL	59.8	59.5	47.6	59.8	48.1	43.4	29.8	50.1
VLP-SL	59.5	59.2	48.2	59.7	49.3	49.0	45.0	53.5
SPIEL-gs	64.1	**61.3**	**49.6**	**61.6**	**51.1**	54.9	51.9	57.8
SPIEL-gm	**64.6**	60.8	48.3	59.5	51.0	**55.3**	**52.9**	**57.9**

Table 1: F1 scores for attribute prediction and accuracies for zero-shot evaluation on CompGuessWhat?!.

3.2 Transfer evaluation

For the transfer evaluation, we use the VQA dataset TDIUC (Kafle and Kanan, 2017). It provides a finer-grained way to assess the quality of the representations learned by our guessing game transfer technique in terms of several question types including object categories and their attributes. Specifically, we were interested in improving on the following question types: 1) Positional reasoning; 2) Counting; 3) Object presence; 4) Utility/Affordances; 5) Attribute; 6) Color; and 7) Object recognition. TDIUC is evaluated using the arithmetic mean accuracy per question type (A-MPT), as well as the harmonic mean (H-MPT) that better captures the skewed question-type distribution. In Table 2, we report a comparison between variants trained on guessing games data (VLP+SL and SPIEL-⋆), the original model VLP trained on Conceptual Captions (VLP+CC) and other state-of-the-art models *specifically* designed for the VQA task such as MUREL (Cadene et al., 2019), RAU (Noh and Han, 2016), NMN (Andreas et al., 2016), MCB-⋆ (Fukui et al., 2016). The full set of results is available in the Appendix, Table 4.

Among them, MUREL achieves the best scores across the board, due to a custom iterative reasoning mechanism and a non-linear fusion module. However, all our models have a more balanced overall performance which results in better harmonic means (H-MPT, +5 points over MUREL). Specifically, this improvement is favoured by an increase in accuracy on the *Utility/Affordances* ques-



TDIUC predictions (a)

Are the contents of the plate edible?	
GOLD answer	yes
VLP+CC	beer
VLP+SP+gm	yes

what is the spoon made of?	
GOLD answer	wood
VLP+CC	plastic
VLP+SP+gm	wood

what is the water glass made of?	
GOLD answer	glass
VLP+CC	plastic
VLP+SP+gm	glass

Generated dialogue (b)

is it food?	no
is it a spoon?	no
is it a cup?	no
is it a bowl?	yes
left picture?	yes
the one on the soup?	no
the one with the soup in it?	yes

Attribute prediction (c)

Situated attributes	Confidence
home	99.83%
bowl_used_to_scoop	99.37%
kitchen_utentils	99.53%
bowl_can_be_carried	99.40%
center	71.75%

Figure 2: We show the ability of the model to play guessing games with the *bowl* as target object (highlighted in *red*). Given the generated dialogue, we use the probing classifier trained for CompGuessWhat?! to predict the *bowl*'s attributes. Predictions on TDIUC questions associated with the current image are reported as well.

tion type (+20.7). As shown by the attribute prediction in the CompGuessWhat?! and depicted in Figure 2 (c), our models learn better representations than competitors specifically for abstract attributes among which there are object affordances. Particularly, we can see how it is able to understand that certain objects can *contain* things (e.g. "the one with the soup in it?"), that objects have specific functions (e.g. "are the contents of the plate edible?") or that they have specific properties (e.g. "a spoon is made of wood").

The effectiveness of the proposed fine-tuning procedure is confirmed by the improved performance across all the question types compared to our baseline `VLP+CC`. Models such as `MUREL` and `MCB-*` equipped with specific VQA modules have an advantage on specific question (e.g., positional reasoning) compared to VLP that relies only on BERT self-attention layers (Devlin et al., 2019). In addition, when comparing the two SPIEL variants, a similar trend showed in the in-domain evaluation can be observed. Particularly, `SPIEL-gm` benefits from being exposed to more language data coming from successful and failed guessing games.

MODEL	POSITION	COUNT	PRESENCE	AFFORD.	ATTR.	COLOR	RECOG.	A-MPT	H-MPT
`RAU`	35.3	48.4	94.4	31.6	56.5	66.9	86.1	67.8	59.0
`NMN`	27.9	49.2	92.5	25.2	47.7	54.9	82.0	62.6	51.9
`MCB-A`	55.4	51.0	93.6	**35.1**	56.7	68.5	81.9	67.9	**60.5**
`MCB`	33.3	50.3	91.8	33.9	53.2	56.9	84.6	65.8	58.0
`MUREL`	**41.2**	**61.8**	**95.8**	21.4	**58.2**	**74.4**	**89.4**	**71.2**	59.3
`VLP`									
`+CC`	36.9	55.3	94.7	31.0	55.4	67.3	85.8	68.8	60.1
`+SL`	39.0	**57.6**	**94.8**	**42.1**	54.3	69.0	86.1	70.5	64.0
`SPIEL-gs`	**40.9**	57.5	**94.8**	36.3	56.9	69.2	**86.3**	70.4	63.3
`SPIEL-gm`	40.6	57.0	**94.8**	39.2	**57.0**	**69.4**	86.2	**70.9**	**64.3**

Table 2: Results for the transfer evaluation on TDIUC. The models are divided in two categories: (top) Models specifically designed for VQA and (bottom) our VLP-based implementations. We report only the question types that we believe will benefit from the guessing games fine-tuning procedure. For the full set of results please refer to Appendix, Table 4.

4 Conclusions

In this work, we verified that representations learned while playing guessing games can be transferred to other downstream tasks such as VQA. We presented two ways of learning from guessing games data namely multi-task learning and SPIEL. Models using SPIEL performed better both on in-domain evaluation on CompGuessWhat?! as well as on the transfer task TDIUC. Our self-play procedure was able to learn useful and finer-grained object representations such as object affordances, thus demonstrating that *learning to guess helps learning to ground.*

The current study showed how we can apply the SPIEL training procedure to a VQA dataset such as TDIUC. We believe that this work can be extended to other datasets because the SPIEL procedure only requires a set of images and associated object bounding boxes. These could be either gold or generated by a trained object detector therefore classifying guessing games as a *holistic self-training procedure* for multi-modal datasets.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, et al. 2020. Experience grounds language. *arXiv preprint arXiv:2004.10151*.

Remi Cadene, Hedi Ben-Younes, Matthieu Cord, and Nicolas Thome. 2019. Murel: Multimodal relational reasoning for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1989–1998.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*.

Michael Cogswell, Jiasen Lu, Rishabh Jain, Stefan Lee, Devi Parikh, and Dhruv Batra. 2020. Dialog without dialog data: Learning visual dialog agents from vqa data. *arXiv preprint arXiv:2007.12750*.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2017. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335.

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 457–468.

Robert Godwin-Jones. 2014. Games in language learning: Opportunities and challenges. *Language Learning & Technology*, 18(2):9–19.

Thomas Hainey, Thomas M Connolly, Elizabeth A Boyle, Amanda Wilson, and Aisya Razak. 2016. A systematic literature review of games-based learning empirical evidence in primary education. *Computers & Education*, 102:202–223.

James A Hampton. 1979. Polymorphous concepts in semantic memory. *Journal of verbal learning and verbal behavior*, 18(4):441–461.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Kushal Kafle and Christopher Kanan. 2017. An analysis of visual question answering algorithms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1965–1973.

Simon Kirby, Tom Griffiths, and Kenny Smith. 2014. Iterated learning and the evolution of language. *Current opinion in neurobiology*, 28:108–114.

Jason Lee, Kyunghyun Cho, and Douwe Kiela. 2019. Countering language drift via visual grounding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4376–4386.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446.

Hyeonwoo Noh and Bohyung Han. 2016. Training recurrent answering units with joint loss minimization for vqa. *arXiv preprint arXiv:1606.03647*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Ravi Shekhar, Tim Baumgärtner, Aashish Venkatesh, Elia Bruni, Raffaella Bernardi, and Raquel Fernández. 2018. Ask no more: Deciding when to guess in referential visual dialogue. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1218–1233.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2019. Beyond task success: A closer look at jointly learning to see, ask, and guesswhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2578–2587.

Luc Steels. 2015. *The Talking Heads experiment: Origins of words and meanings*, volume 1. Language Science Press.

Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courvile, and Olivier Pietquin. 2017. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2765–2771.

Alessandro Suglia, Ioannis Konstas, Andrea Vanzo, Emanuele Bastianelli, Desmond Elliott, Stella Frank, and Oliver Lemon. 2020. CompGuessWhat?!: A multi-task evaluation framework for grounded language learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7625–7641, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5103–5114.

Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. 2017. Guesswhat?! visual object discovery through multi-modal dialogue. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4466–4475. IEEE Computer Society.

Ludwig Wittgenstein, Gertrude Elizabeth Margaret Anscombe, and Rush Rhees. 1953. *Philosophische Untersuchungen.(Philosophical investigations*.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. In *AAAI*, pages 13041–13049.

Bohan Zhuang, Qi Wu, Chunhua Shen, Ian Reid, and Anton Van Den Hengel. 2018. Parallel attention: A unified framework for visual object discovery through dialogs and queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4252–4261.

A Appendices

A.1 Self-Play via Iterated Experience Learning (SPIEL)

Learning to replicate gold dialogues is not enough to play successfully. High performance in gameplay can be achieved only when the agents start playing the game and are exposed to their own mistakes. Reinforcement Learning (Strub et al., 2017) or Collaborative Learning (Shekhar et al., 2019) are possible approaches to tackle this problem.

Inspired by iterated learning (Kirby et al., 2014), we design a process by which "the gameplay arises in one instance of the questioner through induction on the basis of observations of gameplay in other questioner agents who acquired that gameplay capability in the same way". Therefore, we call our procedure *Self-play via Iterated Experience Learning* (SPIEL).

In this setup, we assume we have access to a set of images $\mathcal{I}$ and for each image I we have object bounding boxes $\mathcal{O}_I$. The SP training procedure, showed in Figure 1, can be described as follows. We assume that there is a Questioner agent Q and an Oracle agent O. At the beginning of the procedure they are initialised with agents Q^0 and O, respectively, trained with Supervised Learning using gold successful dialogues [3]. We consider every iteration e of the algorithm as a *self-play* epoch. In a single self-play epoch we alternate 3 phases: 1) *interactive* phase: the agents play guessing games with novel combinations of image and target object; 2) *transmission* phase: the questioner creates new datasets from the dialogues generated over the epochs; 3) *learning* phase: multi-task learning is used to fine-tune the Questioner parameters using the datasets collected for the current epoch.

A.1.1 Interactive phase

We start the interactive phase by first sampling a set of reference games $\mathcal{G}^e$ which consists of pairs $(I, \hat{o})$ where $I \in \mathcal{I}$ and $\hat{o}$ is the target object sampled at random from the object annotations $\mathcal{O}_I$. The agents Q^e and O play the games $\mathcal{G}^e$ and accumulate the generated experiences. During this phase, the questioner agent is using the most updated weights generated at epoch $e-1$. It generates questions by nucleus sampling (Holtzman et al., 2019) from the probability distribution over the vocabulary learned by the generator head. When the `[STOP]` token is sampled, the guesser head, conditioned on the dialogue generated so far, selects the object $\tilde{o}$ with the highest probability. A game is *successful* if the predicted object $\tilde{o}$ is equal to the target object $\hat{o}$.

A.1.2 Transmission phase

For every epoch e, in the transmission phase, we create the datasets $\mathcal{D}_q$ and $\mathcal{D}_g$ for the questioner and guesser heads, respectively, used in the learning phase for the questioner parameters update.

Questioner experience buffer To make sure that the questioner does not experience *language drift* (Lee et al., 2019), we consider a fixed dataset $\mathcal{D}_q$ composed of dialogues generated by humans contained in the GuessWhat?! training data. The shared encoder Γ benefits from this data too because it is still exposed to human generated language, which guarantees better generalisation.

Guesser experience buffer The Guesser should learn from its own mistakes – therefore we use generated dialogues for the model updates (de Vries et al., 2017; Shekhar et al., 2019). Inspired by Prioritised Experience Replay (Schaul et al., 2015), we create the experience buffer for the guesser $\mathcal{E}_g^e$ by accumulating all the *unique* and *valid* dialogues generated until epoch e. We consider a dialogue *unique* if $\mathcal{D}_g^e$ does not contain another dialogue with the same encoding [4]. In addition, we consider a dialogue *valid* if it does not contain repeated questions. We cap the number of dialogues in $\mathcal{D}_g^e$ so that it matches the number of experiences in $\mathcal{D}_q$. This is done so that during the multi-task training procedure there is an equal number of dialogues for each task from which the agent will learn.

A.1.3 Learning phase

In this phase, we use the same multi-task training procedure that was used during the supervised learning phase. We update the Questioner parameters using the dialogues collected in $\mathcal{D}_q$ and $\mathcal{D}_g^e$. The updated parameters resulting from this step will be used for the self-play epoch $e+1$.

A.2 VLP implementation

A.2.1 Multi-modal encoder

To implement the agents in our guessing games, we rely on VLP, a single-stream multi-modal model (Zhou et al., 2020) that jointly learns visual and language representations using Conceptual Captions (CC) dataset (Sharma et al., 2018). The input starts with a classification token (`[CLS]`), followed by a series of K visual tokens, a separation token (`[SEP]`) divides the *dialogue* sequence from the visual and from the sequence of *tokens to be generated*. In a guessing game, we represent the reference image I as a set of image regions extracted from an off-the-shelf object detector $\{r_1, r_2, \ldots, r_K\}$. Following (Zhou et al., 2020), each region r_i is represented by linear transformation of a feature vector $f \in \mathbb{R}^{d_n}$, region class probabilities $c \in \mathbb{R}^{d_c}$ and region geometric information $g \in \mathbb{R}^{d_o}$ where $d_o = 5$ consists of four values for top left and bottom right corner coordinates of the region bounding box (normalized between 0 and 1) and one value for its relative area (i.e., ratio of the bounding box area to the image area, also between 0 and 1). The Questioner models uses at most 36 predicted bounding boxes from FastRCNN while the Guesser is using features generated by FastRCNN for gold bounding boxes. We use a specific segment id s_v for every region.

For the language part, we use Wordpiece embeddings (Wu et al., 2016). In particular, we flatten the turns of the dialogue context as a sequence of

---

[3] The Oracle is fixed during this learning procedure.

[4] The encoding of a dialogue is the SHA-256 hash associated with its sequence of tokens.

tokens. However, to allow the model to differentiate between question and answer tokens, following (Wolf et al., 2019), we rely on novel segment ids (s_u,s_a). The VLP's hidden state of the `[CLS]` token is used as context representation $\mathbf{h}_c$.

A.2.2 Oracle design

The implementation of the Oracle follows the one presented in the original VLP paper to solve the VQA task (Zhou et al., 2020). Particularly, the model predicts a probability distribution over the possible answers by using a multi-layer feed-forward neural network that receives in input the element-wise product between the hidden state associated with the `[CLS]` token and the hidden state associated with target object. The model is optimised by minimising the cross-entropy loss using as training dataset the question/answer pairs in the successful GuessWhat?! training dialogues.

A.2.3 Questioner design

We rely on the VLP ability to generate captions for the question generation task. In particular, we provide in input to the model: 1) predicted FastR-CNN visual features following (Zhou et al., 2020); 2) dialogue generated so far as a flattened sequence of tokens; 3) question to be generated. We use another segment id s_q to allow the model to differentiate what is the input and which are the tokens to be generated. Following (Dong et al., 2019), we make sure that the attention mask for tokens of the question to be generated are masked so that the token at timestep t is not allowed to attend to the future tokens (*seq2seq attention mask*). For this specific model, we use the masked language modelling objective (Devlin et al., 2019) casting the task as multi-modal masked language modelling.

A.3 GuessWhat?! evaluation

Oracle evaluation We report the test accuracy for the Oracle of 82.22%. The baseline model used by all the other is 78.5% (de Vries et al., 2017).

Guesser evaluation We report in Table 3 the accuracy of the guesser in predicting the target object when gold dialogues are given in input. We compare this model with several baselines reported in (de Vries et al., 2017) (first block), more sophisticated methods such as `ParallelAttention` (Zhuang et al., 2018) and `GDSE-*` (Shekhar et al., 2019) (second block) as well as other Transformer-based models such as VILBERT (Lu et al., 2020) (third block).

Model	Accuracy
Human	90.80%
Random	17.10%
LSTM	61.30%
HRED	61%
LSTM+VGG	60.50%
HRED+VGG	60.40%
ParallelAttention	63.40%
GDSE-SL	62.96%
GDSE-CL	59.79%
VILBERT	65.69%
VLP-SL	69.30%
SPIEL-gs	**71.80%**
SPIEL-gm	71.70%

Table 3: Results for the guesser accuracy evaluation on gold dialogues.

MODEL	POSITIONAL	COUNTING	PRESENCE	AFFORDANCES	ATTRIBUTE	COLOR	RECOGNITION	SCENE	ABSURD	SENTIMENT	ACTIVITY	SPORT	ACCURACY	A-MPT	H-MPT
MUREL	41.19	61.78	95.75	21.43	58.19	74.43	89.41	96.11	99.80	60.65	63.83	96.20	88.20	71.20	59.30
RAU	35.26	48.43	94.38	31.58	56.49	66.86	86.11	93.96	96.08	60.09	51.60	93.47	84.26	67.81	59.00
NMN	27.92	49.21	92.50	25.15	47.66	54.91	82.02	91.88	87.51	58.02	44.26	89.99	79.56	62.59	51.87
MCB-A	55.40	51.01	93.64	35.09	56.72	68.54	85.54	93.06	84.82	66.25	52.35	92.77	81.86	67.90	**60.47**
MCB	33.34	50.29	91.84	33.92	53.24	56.93	84.63	92.04	83.44	65.46	51.42	92.47	79.20	65.75	58.03
VLP-CC	36.93	55.28	94.65	30.99	55.42	67.33	85.76	92.98	98.34	62.62	51.34	94.11	85.60	68.81	60.14
VLP-SL	39.04	57.61	94.79	42.11	54.29	69.01	86.07	93.39	97.54	65.77	52.39	94.34	85.98	70.53	63.95
SPIEL-gs	40.94	57.53	94.76	36.26	56.87	69.2	86.33	93.97	97.48	62.3	54.44	94.62	**86.1**	70.39	63.34
SPIEL-gm	40.6	57.01	94.77	39.18	56.97	69.42	86.21	93.72	97.19	66.09	55.29	94.18	86	**70.89**	**64.31**

Table 4: Summary of results for the transfer evaluation on TDIUC. The models are divided in two categories: (1) Models which are specifically designed for VQA (top) and (2) models that rely on the VLP encoder to generalise to different downstream tasks (bottom). We underline the question types that we believe will benefit from the guessing games transfer/fine-tuning procedure.

A Unified Feature Representation for Lexical Connotations

Emily Allaway
Columbia University
New York, USA
eallaway@cs.columbia.edu

Kathleen McKeown
Columbia University
New York, USA
kathy@cs.columbia.edu

Abstract

Ideological attitudes and stance are often expressed through subtle meanings of words and phrases. Understanding these *connotations* is critical to recognizing the cultural and emotional perspectives of the speaker. In this paper, we use distant labeling to create a new lexical resource representing connotation aspects for nouns and adjectives. Our analysis shows that it aligns well with human judgments. Additionally, we present a method for creating lexical representations that capture connotations within the embedding space and show that using the embeddings provides a statistically significant improvement on the task of stance detection when data is limited.

1 Introduction

Expressions of ideological attitudes are widespread in today's online world, influencing how we perceive and react to events and people on a daily basis. These attitudes are often expressed through subtle expressions or associations (Somasundaran and Wiebe, 2010; Murakami and Putra, 2010). For example, the sentence "the people opposed gun control" conveys no information about the author's opinion. However, by adding just one word, "the *selfish* people opposed gun control", the author can convey their stance on both gun control (against) and the people who support it (not valuable and disliked). Discerning such subtle meaning is crucial for fully understanding and recognizing the hidden influences behind everyday content.

Recent studies in NLP have begun to examine these hidden influences through framing in social media and news (Asur and Huberman, 2010; Hartmann et al., 2019; Klenner, 2017) and style detection in hyperpartisan news (Potthast et al., 2018). Lexical connotations provide a method to study these influences, including stance, in more detail.

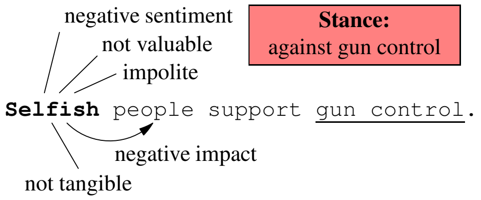


Figure 1: Connotations of the word "selfish" and the resulting implied stance on the topic "gun control".

Connotations are implied cultural and emotional associations for words that augment their literal meanings (Carpuat, 2015; Feng et al., 2011). Connotation values are *associated* with a phrase (e.g., fear is associated with "cancer") (Feng et al., 2011) and capture a range of nuances, such as whether a phrase is an insult or implies value (see Figure 1).

In this paper, we define six new fine-grained connotation aspects for nouns and adjectives, filling a gap in the literature on connotation lexica, which has focused on verbs (Sap et al., 2017; Rashkin et al., 2016, 2017), and coarse-grained polarity (Feng et al., 2011; Kang et al., 2014). We create a new distantly labeled English lexicon that maps nouns and adjectives to our six aspects and show that it aligns well with human judgments. In addition, we show that our lexicon confirms existing hypotheses about subtle semantic differences between synonyms.

We then learn a single connotation embedding space for words from all parts of speech, combining our lexicon with existing verb lexica and contributing to the literature on unifying lexica (Hoyle et al., 2019). Intrinsic evaluation shows that our embedding space captures clusters of connotatively-similar words. In addition, our embedding model can generate representations for new words without the numerous training examples required by standard word-embedding methods. Finally, we show that our connotation embeddings improve performance on stance detection, particularly in a

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2145–2163
April 19 - 23, 2021.

low-resource setting.

Our contributions are as follows: (1) we create a new connotation lexicon and show that it aligns well with human judgments, (2) we train a connotation feature embedding for all parts of speech and show that it captures connotations within the embedding space, and (3) we show the connotation embeddings improve stance detection when data is limited. Our resources are available: `https://github.com/emilyallaway/connotation-embedding`.

2 Related Work

Studies of connotation build upon the literature examining subtle language nuances, including good and bad effects of verbs (Choi and Wiebe, 2014), evoked sentiments and emotions (Mohammad et al., 2013a; Mohammad and Turney, 2010; Mohammad, 2018b), multi-dimensional sentiment (Whissell, 2009; Mohammad, 2018a; Whissell, 1989), offensiveness (Klenner et al., 2018), and psycho-sociological properties of words (Stone and Hunt, 1963; Tausczik and Pennebaker, 2009). Work explicitly on connotations has focused primarily on detailed aspects for verbs (Rashkin et al., 2016, 2017; Sap et al., 2017; Klenner, 2017) or single polarities for many parts of speech (Feng et al., 2011, 2013; Kang et al., 2014). One exception is the work of Field et al. (2019), which extends limited detailed connotation dimensions from verbs to nouns within the context of certain verbs. Our work is unique in directly defining detailed aspects for nouns and adjectives.

Early work on stance detection applied topic-specific models to various genres, including online debate forums (Sridhar et al., 2015; Somasundaran and Wiebe, 2010; Murakami and Putra, 2010; Hasan and Ng, 2013, 2014) and student essays (Faulkner, 2014). More recent studies have used a single model for many topics to predict stance in Tweets (Mohammad et al., 2016; Augenstein et al., 2016; Xu et al., 2018) and as part of the fact extraction and verification pipeline (Conforti et al., 2018; Ghanem et al., 2018; Riedel et al., 2017; Hanselowski et al., 2018). Klenner et al. (2017) explore the relationship between connotations and stance through verb frames. In contrast, our work studies stance using connotation representations from a learned joint embedding space for words from all parts of speech. Recently, Webson et al. (2020) examine representations of political ideology as connotations and its use in information retrieval. Representation learning has been used for stance detection of online debates by Li et al. (2018), who develop a joint representation of the text and the authors. Our work, however, uses a representation of word connotations and does not use any author information (a strong feature in fully-supervised datasets but which may not be available in real-world settings).

3 Connotation Lexicon

We build a connotation lexicon for nouns and adjectives by defining six new aspects of connotation. We take inspiration from verb connotation frames and their extensions (Rashkin et al., 2016; Sap et al., 2017), which define aspects of connotation in terms of the agent and theme of transitive verbs. Rashkin et al. (2016) define six aspects of connotation for verbs (entities', writer's, and reader's perspectives, effect, value, and mental state) in connotation frames (e.g., "suffer" $\rightsquigarrow$ negative effect on the agent) and Sap et al. (2017) extend these aspects to include power and agency.

We first define the six new aspects of connotation for nouns and adjectives (§3.1) in our work, then we describe our distant labeling procedure (§3.2) and human evaluation of the final lexicon (§3.3).

3.1 Definitions

We use w to indicate a word and w' to indicate the person, thing or attribute signified by w.

For each w, we define (1) **Social Value**: whether w' is considered valuable by society, (2) **Politeness** *(Polite)*: whether w is a socially polite term, (3) **Impact** whether w' has an impact on society (or the thing modified by w if w is an adjective), (4) **Factuality** *(Fact)*: whether w' is tangible, (5) **Sentiment** *(Sent)*: the sentiment polarity of w, and (6) **Emotional association** *(Emo)*: the emotions associated with w'. We show examples in Table 1.

(1) Social Value includes both the value of objects or concepts and the social status and power of people or people-referring nouns (e.g., occupations). "Sociocultural pragmatic reasoning" (Colston and Katz, 2005) about such factors is crucial for understanding language such as connotations.

Initial work on connotation polarity lexica recognized the important role of *Social Value* in overall connotation by defining a 'positive' connotation for objects and concepts that people value (Feng et al., 2011). Later work made this idea more explicit

Aspect	Lexicon	Example Rules	Examples
Social Value	GI	Authoritative power → valuable Related to failure → not valuable	`attorney` ⇝ valuable (+) `aimless` ⇝ not valuable (-)
Politeness	GI	Gain of respect → polite Loss of affection → impolite	`commendable` ⇝ polite (+) `alienation` ⇝ impolite (-)
Impact	GI	Virtue → positive Loss of well-being → negative	`adept` ⇝ positive impact (+) `shock` ⇝ negative impact (-)
Factuality	DAL	Imagery(w) $> \theta_F \rightarrow$ factual Imagery(w) $< -\theta_F \rightarrow$ not factual	`rocky` ⇝ factual (+) `tradition` ⇝ not factual(-)
Sentiment	CWN	$v > \theta_S \rightarrow$ positive $v < -\theta_S \rightarrow$ negative	`song` ⇝ positive (+) `cancerous` ⇝ negative (-)
Emotional Association	NRC	emotions $E \subseteq$ {anger, joy, fear, trust, anticipation, sadness, disgust, surprise}	`snake` ⇝ {disgust, fear} `effective` ⇝ {trust}

Table 1: Example mappings from existing lexica to our connotation aspects. GI: Harvard General Inquirer, DAL: Dictionary of Affect in Language, CWN: Connotation WordNet, and NRC: NRC Emotion Lexicon. Scores for imagery, Imagery(w), and sentiment, v, are real-valued.

by defining 'Value' for transitive verb arguments in connotation frames. More recently 'power' and 'agency', components of *Social Value*, have been defined for verbs in connotation frames and for nouns in context (Field et al., 2019) and have been used to analyze bias and framing in a variety of texts, illustrating the applications and importance of *Social Value* in connotations.

(2) Politeness follows the definition of Lakoff (1973) in noting words that make the addressee feel good but also includes notions of formality. These notions have been previously studied within the context of politeness as a set of behaviors and linguistic cues (Brown and Levinson, 1987; Danescu-Niculescu-Mizil et al., 2013; Aubakirova and Bansal, 2016). We focus on purely lexical distinctions because how one comprehends these distinctions affects one's "attitude towards the speaker ... or some issue" as well as whether one feels insulted by the exchange (Colston and Katz, 2005). This aspect of perspective is a component of verb connotation frames and we extend it to nouns and adjectives in our lexicon through *Politeness*.

(3) Impact and effect have been studied in verb connotation frames and other verb lexica (Choi and Wiebe, 2014), capturing notions of implicit benefit or harm on the arguments of the verb. We extend this idea to nouns and adjectives by observing that while they do not directly have arguments, nouns (e.g. "democracy") often impact society and adjectives (e.g. "sick") impact the nouns they modify. Thus, we define *Impact* in this way.

(4) Factuality captures whether words correspond to real-world objects or attributes, following the sense of Saurí and Pustejovsky (2009). Klenner and Clematide (2016) argue that the factuality of events is crucial for understanding sentiment inferences. Building upon this, Klenner et al. (2017) use factuality as a key component of German verb connotations and of applying those connotations to analyze stance and sides in German Facebook posts. Imagery, as an "indicator of abstraction" (Whissell, 2009), also models a similar attribute to event factuality for all parts of speech. Given its importance, we include a notion of *Factuality* for nouns and adjectives as aspect of connotations.

(5) Sentiment polarity has been used to convey overall connotations since the early work on connotation lexica (Feng et al., 2011, 2013; Kang et al., 2014). As such, we deem it important to include this polarity in our lexicon.

(6) Emotional Associations for words can be strong, persisting long after they are formed and improving the recall of memories triggered by those words (Rubin, 2006). Emotions are also impacted when people process non-literal meaning (Colston and Katz, 2005). To fully understand what a piece of text is trying to convey, it is important to understand what emotional associations exist in the text. For example, news headlines often aim to evoke strong emotions in their readers (Mohammad and Turney, 2013). To capture this, we include *Emotional Association* as an aspect of connotation.

3.2 Labeling Connotations

We use distant labeling to build our lexicon, since complete manual annotation of a lexical resource is

	Social Val	*Polite*	*Impact*	*Fact*	*Sent*
%+	32.1	10.5	14.8	19.0	56.8
%−	15.5	1.0	13.3	67.2	33.1

Table 2: Class distributions of fully-labeled words in the connotation lexicon.

a lengthy and costly process. Although crowdsourcing can lessen these burdens, the results are often unreliable with low inter-annotator agreement and, for this reason, many lexical resources are automatically created (Mohammad, 2012; Mohammad et al., 2013b; Kang et al., 2014). Following these researchers, we automatically generate our lexicon by combining several existing lexica.

To generate our lexicon, we map dimensions from existing lexica to connotation aspects (see Table 1). We use dimensions from the Harvard General Inquirer (Stone and Hunt, 1963) for *Social Value*, *Politeness*, and *Impact*. For *Factuality* we map the real-valued 'Imagery' dimension, Imagery(w), from the revised Dictionary of Affect in Language (Whissell, 2009) into distinct classes. For *Sentiment* we directly use the polarity v from Connotation WordNet (Kang et al., 2014) and for *Emotional Association* we use the eight Plutchik emotions (Plutchik, 2001) from the NRC Emotion Lexicon (Mohammad and Turney, 2013) (see appendix B for full rules).

The labels are word-sense-independent, following other automatically generated lexica, such as the Sentiment140 lexicon (Mohammad et al., 2013b), which do not treat word sense. In addition, sense-level annotations are not available for all lexica in our distant labeling method and therefore sense-level connotations would require both extensive manual annotation and automated word-sense disambiguation, introducing cost and additional noise. As a result, we use sense-level distinctions (e.g., in the Harvard General Inquirer) when available and combine the labels for an aspect across senses to obtain the final connotation aspect label. These aggregate aspect labels represent a word's connotative potential, rather than exact value

Our resulting lexicon has $7,578$ words fully-labeled for all aspects, with an additional $\sim$93k words labeled only for some aspects (e.g., only *Sentiment*), resulting in $100,176$ words total. For each non-emotion aspect, we have a label $l \in \{-1, 0, 1\}$. For *Emotional Association*, each of the eight emotions has label $l \in \{0, 1\}$.

Aspect	**Avg** κ	**Avg % Agree**	**Lex % Agree**	**Lex % NC**
Social Value	.699	88.9	68.6	92.6
Politeness	.381	56.6	59.4	95.1
Impact	.630	87.6	73.7	94.6
Factuality	.675	86.3	58.0	77.7
Average	.596	87.9	64.2	90.0

Table 3: Lexicon annotation results. Fleiss' κ and % agreement are averaged across nouns and adjectives. Lex % is agreement between annotators and the lexicon. NC indicates non-conflicting value agreement.

We find that many aspects exhibit uneven class distributions (e.g., 10.5% of words are polite and only 1% are impolite) (see Table 2). For emotions, we calculate the class distribution using the number of fully-labeled words with at least one associated emotion ($1,373$ words or 18%). For these $1,373$ words, the average number of associated emotions is $\sim$2. Our distributions are similar to previous work on verb connotations, where distributions range from 1.4% to 20.2% for the smallest class (Rashkin et al., 2016).

3.3 Evaluation of the Lexical Resource

Human Evaluation

We evaluate the quality of the lexicon by creating a gold-labeled set and comparing the labels created with distant supervision against the human labels. We ask nine NLP researchers[1] to annotate 350 words (175 nouns, 175 adjectives) for *Social Value, Politeness, Impact* and *Factuality*. We do not annotate *Sentiment* or *Emotional Association*, since these labels come directly from existing lexica.

Annotators are given a word w, along with its definitions (for all senses) and related words, and annotate connotation independent of word sense. This setup mimics the input to the representation learning models in §4. The average Fleiss' κ across nouns and adjectives is 0.60 (see Table 3), indicating substantial agreement. We select as the final annotator label, the majority vote.

We find that the distantly labeled lexicon agrees with human annotators the majority of the time (on average 64.2% or Cohen's $\kappa = 0.368$ (Cohen, 1988)). If we consider *non-conflicting value agreement* (NC), the lexicon agreement with humans rises to 90%, where NC agreement is defined as: the pairs (+, neutral) and (−, neutral) agree but

---

[1] native English speakers at Columbia University

Aspect	**Same Connotation**	**Different Connotation**
Social Value	(=) `hurry` vs. `rush` (+) `fantastic` vs. `wonderful`	`sentence` (=) vs. `condemnation` (-) `relentless` (-) vs. `persistent` (+)
Politeness	(-) `disgrace` vs. `shame` (+) `humble` vs. `modest`	`gentleman` (+) vs. `man` (=) `preposterous` (=) vs. `ridiculous` (-)
Impact	(-) `weariness` vs. `fatigue` (+) `rightful` vs. `lawful`	`fire` (=) vs. `burning` (-) `supporting` (+) vs. `suffering` (-)
Factuality	(+) `daytime` vs. `day` (-) `dire` vs. `terrible`	`post` (+) vs. `position` (=) `protection` (-) vs. `security` (=)
Sentiment	(+) `wonderous` vs. `marvelous` (-) `commotion` vs. `disturbance`	`giddy` (=) vs. `dizzy` (+) `moving` (=) vs. `striking` (+)
Emotional Association	(trust) `wise` vs. `smarter` (sadness) `flaw` vs. `disturbance`	`dire` (fear, sadness) vs. `terrible` (fear,sadness,disgust)

Table 4: Synonym examples from the lexicon. = indicates neutral or neither.

(+,−) does not. This shows that the lexicon and humans rarely select opposite values and instead disagree on neutral vs. non-neutral.

Looking closer at disagreements between neutral and non-neutral, we see that most result from human annotators selecting a non-neutral label. That is, the lexicon makes fewer distinctions between neutral and non-neutral than humans; humans select a non-neutral value 68% of the time, compared to 56% in the lexicon. Despite this tendency towards neutral, the lexicon aligns with human judgments, agreeing the majority of the time and rarely providing a value opposite to humans.

Synonym Analysis

We also evaluate the ability of our lexicon to capture subtle semantic differences between words using lexical paraphrases (synonyms). In the paraphrase literature, it has been argued that paraphrases actually differ in many ways, including in connotations (Bhagat and Hovy, 2013). In fact, Clark (1992) proposes that absolute synonymy between linguistic forms does not exist. With this in mind, we hypothesize that our connotation lexicon should differentiate between lexical paraphrases.

To test this, we select synonym paraphrase pairs from lexical PPDB (Pavlick et al., 2015) where one element in the pair is in the Wordnet synset of the other [2]. We find that out of the 2216 resulting pairs where both words are in our lexicon, 74.3% have connotations that differ in some aspect. Many words agree on *Sentiment* (67.5% the same), following the intuition that two synonyms likely have the same sentiment but differ in more fine-grained ways. Other pairs agree on *Politeness* (76.1% the same), resulting from the extreme class imbalance for this aspect (88.5% neutral). However, the lexicon does still capture differences along these dimensions, for example in terms of formality (e.g., “gentleman” vs. “man”).

Looking more closely, we find that many times agreements along a particular dimension accurately represent synonyms that differ along other dimensions. For example, “weariness” and “fatigue” both have a negative *Impact*, but “weariness” is associated with *sadness* and “fatigue” is not.

On the other hand, the majority of differences across almost all aspects (79% on average) are between neutral and non-neutral polarities within a synonym pair, for example, between “position” (possibly tangible) and “post” (tangible), from *Factuality*. This confirms the intuition that synonyms often do not have opposing connotation values, although examples do exist (e.g., the *Social Value* of “relentless” vs. “persistent”) (see Table 4). As a whole, our analysis confirms our hypothesis and the claims of Clark (1992) about synonymy.

4 Connotation Embedding

4.1 Methods

Using our connotation lexicon, we train a dense connotation feature representation for words from all parts of speech. We combine three lexica (our lexicon and two verb lexica) into a single vector space, making connotations easier to use as model input and providing a single representation method for the connotations of any word.

We design a novel multi-task learning model that

---

[2] `https://wordnet.princeton.edu/`

jointly predicts all of the connotation labels for a word w, from a learned representation v_w. Each task is to predict the label for exactly one connotation aspect: the 6 aspects in §3.2 for nouns and adjectives and the 11 aspects in CFs+ (connotation frames and their extension to power and agency) for verbs (Rashkin et al., 2016; Sap et al., 2017).

To learn a representation for w we encode dictionary definitions of the word w and words related to w (e.g., synonyms, hypernyms) in a single vector, which we then use to predict connotation labels. We use definitions and related words since linguists have argued that definitions and related words convey a word's meaning (Guralnik, 1958).

Let w be a word with part of speech t. The input to the connotation encoding model is then: (1) a set of dictionary definitions D_{w^t} and (2) a set of words related to w^t, R_{w^t}. We use multiple definitions to capture multiple senses of w^t. To emphasize more prevalent senses of w^t, we use similar repeated definitions for the same sense, collected from multiple sources. From D_{w^t} and R_{w^t}, the encoder produces a connotation feature embedding $v_{w^t} \in \mathbb{R}^d$ of dimension $d = 300$. Then we use v_{w^t} to predict the label ℓ_a for connotation aspect a (see Figure 2).

4.1.1 Encoding Models

For a word w^t, the input to our encoder is $d_{w^t} = [d^1_{w^t}; d^2_{w^t}; ...; d^N_{w^t}] \in \mathbb{R}^{Nd_{in}}$, the sequence of fixed pre-trained token embeddings for concatenated definitions in D_{w^t}. Then we take as our embedding the normalized final hidden state from a BiLSTM, a standard architecture for text encoding: $v_{w^t} = \frac{h_{w^t}}{\|h_{w^t}\|}$, where $h_{w^t} = \text{BiLSTM}(d_{w^t})$ and $h_{w^t} \in \mathbb{R}^{2H}$ is the concatenation of the last forward and backward hidden states (model **CE**).

As a variation of our model, we apply scaled dot-product attention (Vaswani et al., 2017) over the related words R_{w^t}, using h_{w^t} as the attention query, to obtain v_{w^t}. Then we add the result to h_{w^t} before normalizing (model **CE+R**).

4.1.2 Label Classifier

For each connotation aspect, we train a separate linear layer plus softmax with the input $[v_{w^t}; e_{w^t}]$, where e_{w^t} is the pre-trained embedding for w^t. For the non-emotion aspects, the layer has three target classes $\{-1, 0, 1\}$ for most aspects (four classes for the 'power' and 'agency' verb aspects) and we predict the label with highest output probability. For emotions, we do multi-label classification by thresholding the output probabilities for each emotion dimension with a fixed $\theta \in \mathbb{R}$. We include e_{w^t} in the predictor input to encourage v_{w^t} to model connotation information that is complementary to the information present in pre-trained embeddings.

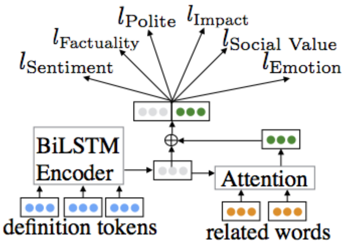


Figure 2: Connotation embedding modeling in §4.1.

4.1.3 Learning

For each non-emotion connotation aspect a (e.g., *Impact*) we calculate the weighted cross-entropy loss $\mathcal{L}^a$. For *Emo* we calculate the one-versus-all cross-entropy loss on each of the eight emotions, $\mathcal{L}^{Emo}_i$ for $1 \leq i \leq 8$, and their sum is $\mathcal{L}^{Emo}$.

In our multi-task joint learning framework **(J)**, we minimize the weighted sum of $\mathcal{L}^a$ across all connotation aspects. We also experiment with training a separate encoding model for each connotation aspect a that minimizes $\mathcal{L}^a$ **(S)**.

4.1.4 Baselines and Models

For each baseline, we implement one classifier per connotation aspect, or, for *Emo*, one classifier for each emotion. Following Rashkin et al. (2016) we implement a Logistic Regression classifier trained on the 300-dimensional pre-trained word embedding for w using the standard L-BFGS optimization algorithm and sample re-weighting (`LR`). We also implement a majority class baseline (`Maj`).

We present three variations of our model: (i) trained jointly for all parts of speech and all connotation aspects (`CE(J)`), (ii) trained on each aspect individually with related word attention (`CE+R(S)`), and (iii) trained jointly on all parts of speech and all connotation aspects with related word attention (`CE+R(J)`).

4.2 Connotation Prediction

4.2.1 Data and Parameters

For nouns and adjectives, we train using the aspects described in §3 (6 aspects). For verbs, we train on 9 aspects[3] from Rashkin et al. (2016) as well as 'power' and 'agency' from Sap et al. (2017)

---

[3] perspective of the writer on the agent/theme, perspective of the agent on the theme, effect on/value/ state of entities

	Maj	**LR**	**CE+R (S)**	**CE (J)**	**CE+R (J)**
N/Aj Avg	.304	.594	.589	.597	.597
Verb Avg	.222	.553	.489	.521	.520
All Avg	.251	.568	.524	.548	.547

Table 5: Macro-averaged F1 for connotation prediction on the test set, averaged across aspects. N/Aj indicates noun and adjective.

(11 aspects total). We split our connotation lexicon (§3) into train (60%), development (20%) and test (20%). For the verb CFs+, we preserve the originally published data splits where possible. We move words only to ensure that all parts of speech for a word are in the same split (e.g., 'evil' both as a noun and adj is in the dev set).

We collect dictionary definitions and related words from all seven dictionaries available on the Wordnik API[4]. These are extracted for each word and part-of-speech pair. We preprocess definitions by removing stopwords, punctuation, and the word itself. We use pre-trained Concept-Net numberbatch embeddings (Speer et al., 2016).

4.2.2 Results

Label Prediction

We present results on the connotation prediction task to check the quality of our representation learning system. Given dictionary definitions and related words, we predict the labels from our lexicon (§3) and CFs+ (see Table 5).

First, we observe that joint learning (models `(J)`) improves over training representations individually (`CE+R(S)`). We hypothesize that joint learning provides regularization across all aspects. Second, we compare joint learning with (`CE+R(J)`) and without (`CE(J)` related words to the strong `LR` baseline. We find that the model with related words (`CE+R(J)`) is statistically indistinguishable from the baseline[5] (for $p \leq 0.05$). In contrast, our model without related words (`CE(J)`) is significantly worse than the `LR` baseline for one aspect (see Appendix D for aspect-level results). Thus we conclude that related words are beneficial for learning connotations.

Overall, our approach provides a single unified feature representation for the lexical connotations of all parts of speech, without any loss in label prediction performance. Specifically, our best representation learning model (`CE+R(J)`) has comparable label prediction performance to a strong baseline (`LR`), a baseline that does not learn any kind of representation. We use `CE+R(J)` to generate connotation embeddings that we use in all further evaluation.

Observations

Our connotation representation learning model presents several advantages. Since the model uses dictionary definitions, we can generate representations for slang words (e.g., "gucci" meaning "really good"), where knowledge-base entries (e.g., in Concept-Net) do not capture the slang meaning. For example, in our connotation embedding space, the nearest neighbors of "gucci" include words related to the slang connotations (e.g., "beneficial" – positive impact, not factual), whereas neighbors in a pre-trained word embedding space are specific to the fashion meaning and connotations (e.g., "buy", "italy", "textile"). Along with slang, our model can also generate representations for new or rare words (e.g., "merchantile") that don't have a pre-trained word representation.

5 Experiments

5.1 Intrinsic Evaluation

To evaluate the connotation embedding space, we look at the 50 nearest neighbors, by Euclidean distance, of every word in our training and development sets. We find that neighbors in the connotation embedding space are more closely related based on the connotation label than in the pre-trained embedding space.

Looking at example nearest neighbors (Table 6) we see that nearest neighbors in the pre-trained embedding space include antonyms (e.g., "inability" is close to "ability") and topically related words (e.g., "merry" is close to "wives"), while in the connotation space, neighbors often share connotation labels even though they may be topically or denotatively unrelated. For example, "slug" (noun) is close to many *impolite* but otherwise unrelated words (e.g., "shove", "murder", "scum") in the connotation embedding space while in the pre-trained space "slug" is close to topically related (e.g., "bug") but *polite* words. Therefore, we can see that words with similar connotations are placed closer together than in the pre-trained semantic space.

To quantify the semantic differences, we measure neighbor-cluster connotation label purity.

---

[4] `https://www.wordnik.com/`

[5] We use an approximate randomization test

Aspect	**Word**	**Conn Only**	**Both**	**Word Only**
Social Value	ability (+)	service, imagination, worth, practical	NONE	lack, inability, enough, difficult
Polite	slug (-)	bang, shove, murder, scum	NONE	quote, bug, exception, reference
Impact	merry (+)	glee, exhilaration, pretty, prosperous	cheery, genial, joyful, fun, merriment, jolly joy, delightful, cheerful	amiable, wonderful, crazy, wives

Table 6: Examples of nearest neighbors in the connotation embedding space (Conn Only), the pre-trained word embedding space (Word Only) and all top 50 nearest neighbors in both spaces (Both). NONE indicats no overlap.

	c	***Social Val***	***Polite***	***Impact***
$r_c^{a(C)}$	+	21.27	2640.14	47.49
	-	5.88	50.00	33.33
$r_c^{a(P)}$	+	4.70	43.71	4.73
	-	2.38	0.54	8.33

Table 7: Select cluster connotation purity ratios.

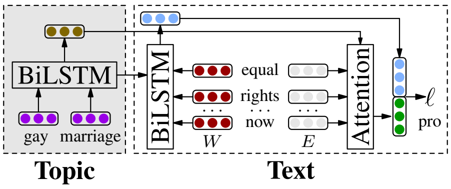


Figure 3: Stance models `BiC+`E; $E \in \{C, W, R\}$.

Specifically, for each connotation aspect a (e.g., *Social Value*) and each non-neutral label c (e.g., valuable (+)), we calculate $r_c^{a(C)}$: the average ratio of words with label c to label $-c$ in the set of nearest neighbors of all words with label c for aspect a. We compare it against the same ratio for the nearest neighbors selected using the same pre-trained word embeddings as in §4.2, denoted $r_c^{a(P)}$.

We find that across connotation aspects, these ratios are higher for the learned connotation embeddings, compared to pre-trained embeddings. For example, $r_+^{\textit{Social Val}(C)} = 21.27$ but $r_+^{\textit{Social Val}(P)} = 4.70$ (see Table 7). This shows the connotation embeddings reshape the pre-trained semantic space.

5.2 Extrinsic Evaluation

We further evaluate our connotation embeddings using the stance detection task, hypothesizing they will lead to improvement. Given a text on a topic (e.g., "gun control"), the task is to predict the stance (pro/con/neutral) towards the topic (see Figure 1).

5.2.1 Methods and Experiments

Models

As a baseline architecture, we implement the bidirectional conditional encoding model (Augenstein et al., 2016). This model encodes a text as h_T with a BiLSTM, conditioned on a separate topic encoding h_P, and predicts stance from h_T (`BiC`). We include connotation embeddings through scaled dot-product attention over the noun, adjective, and verb embeddings from the text, with h_P as the query (see Figure 3). We experiment with three types of embeddings in the attention: pre-trained word embeddings (`BiC+W`), our connotation embeddings (`BiC+C`), and randomly initialized embeddings (`BiC+R`), as a baseline to measure the importance of attention. We also implement a Bag-of-Word-Vectors baseline (`BoWV`), encoding the text and topic as separate BoW vectors and passing their concatenation to a Logistic Regression classifier.

Data and Parameters

We use the Internet Argument Corpus (Abbott et al., 2016): $\sim$59k posts from online debate forums. Of the 16 total topics, four are large (*L*, with $> 7k$ ex each), five are medium (*M*, with $\sim 2k$ ex each), and seven are small (*S*, with 30-300 ex each).

Since not every text will take a position on every topic, we automatically generate 'neutral' examples for the data. To do this, we sample a pro/con example and then assign it a new (different) topic, randomly sampled from the original topic distribution. We split the data into train, development, and test such that no posts by one author are in multiple splits and preprocess the data by removing stopwords and punctuation and lowercasing.

Stance is topic-dependent and as a result, models require numerous training examples for each individual topic. However, many examples are not always available for every topic. Since there are hundreds of thousands of potential topics, the vast majority of which will have very few examples, our goal is to build models that exhibit strong performance across all topics, regardless of size.

	All Data	Trunc Train	Trunc All
BoWV	.3587	.3473	.3613
BiC	.5677	.5151	.5244
BiC+R	.5282	.5260	.5128
BiC+W	.5650	.5384	.5421
BiC+C	.5613	.5579*	.5562*

Table 8: Stance detection macro-averaged F1 on the test set. * indicates significance ($p < 0.01$) between `BiC+C` and `BiC+W`.

Therefore, we experiment with three data scenarios: (i) training and evaluating using all the data (`All Data`), (ii) truncating each topic in training to *M* size (at most $2k$ examples) and evaluating using all data (`Trunc Train`), and (iii) truncating each topic to *M* size in training and in evaluation (`Trunc All`), so that topics have the same frequency for both training and evaluation.

5.2.2 Results

We find that when using all of the training data, the pre-trained embeddings and our connotation embeddings perform comparably (significance level $p = 0.3$). Note that both the connotation and pre-trained embeddings outperform the random embeddings in all scenarios, showing that the architecture difference is not the only reason for improvement when adding embeddings. We find that in both scenarios where data is limited per topic (`Trunc Train` and `Trunc All`), the connotation embeddings improve significantly over the pre-trained word embeddings. In fact, the same trend is visible across varying numbers of training examples (see Figure 4). Our results demonstrate that the connotation information is useful for detecting stance when data is limited.

We find further evidence that the connotation embeddings (`BiC+C`) make the model robust to loss of training data when we look at the results on the individual topic level. Namely, in setting `Trunc Train`, `BiC+C` has a significant improvement (with $p < 0.05$) over `BiC+W` on six topics, including four of the *M* and truncated *L* topics. In fact, for the four *M*/*L* topics, the average per-topic decrease in F1 for `BiC+C` is $1/4$ that of `BiC+W`. These per-topic results further highlight the robustness of `BiC+C` when training data is restricted.

We conclude that connotation embeddings improve stance performance when training data is limited, suggesting they can be used in future work that generalizes stance models to topics with no training data (i.e. most topics).

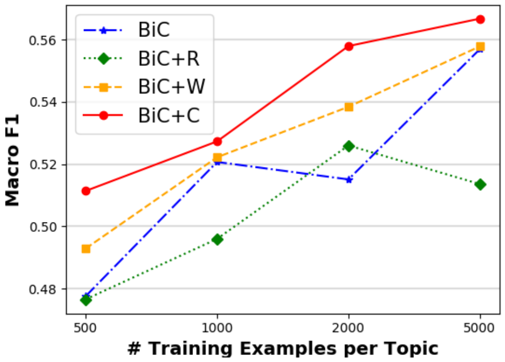


Figure 4: Stance F1 on the test set as number of training examples per topic varies.

6 Conclusion

We create a new lexicon with six new connotation aspects for nouns and adjectives that aligns well with human judgments. We also show that the lexicon confirms hypotheses about semantic divergences between synonyms. We then use our lexicon to train a unified connotation representation for words from all parts of speech, yielding an embedding space that captures more connotative information than pre-trained word embeddings.

We evaluate our connotation representations on stance detection. Since the stance detection tasks encountered in real life concern a very large number of topics, zero-shot and few-shot stance detection are important subtasks. We show that models using our connotation representations are well suited for few-shot stance detection and may also generalize well to zero-shot settings.

In future work, we plan to explore the relationships between connotations, context, and word sense, as well as adapting our methods to learn multi-lingual connotation representations that accurately capture cultural and linguistic variations.

Acknowledgements

We thank the Columbia NLP group and the anonymous reviewers for their comments. This work is based on research sponsored by DARPA under agreement number FA8750-18- 2-0014. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

References

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn A. Walker. 2016. Internet argument corpus 2.0: An sql schema for dialogic social media and the corpora to go with it. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*.

Sitaram Asur and Bernardo A. Huberman. 2010. Predicting the future with social media. In *Web Intelligence*.

Malika Aubakirova and Mohit Bansal. 2016. Interpreting neural networks to improve politeness comprehension. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2035–2041, Austin, Texas. Association for Computational Linguistics.

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Rahul Bhagat and Eduard H. Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39:463–472.

Penelope Brown and Stephen C. Levinson. 1987. *Politeness: Some universals in language usage*. Cambridge University Press.

Marine Carpuat. 2015. Connotation in translation. In *Proceedings of 6th Workshop on Computational Approches to Subjectivity, Sentiment and Social Media Analysis*.

Serina Chang and Kathleen McKeown. 2019. Automatically inferring gender associations from language. In *Proceedings of 9th International Joint Conference on Natural Language Processing*.

Yoonjung Choi and Janyce Wiebe. 2014. +/-effectwordnet: Sense-level lexicon acquisition for opinion inference. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

E. V. Clark. 1992. Conventionality and contrasts: Pragmatic principles with lexical consequences. In *Frame, Fields,and Contrasts: New Essays in Semantic Lexical Organization*, pages 171–188. Lawrence Erlbaum Associates, Hillsdale, NJ.

Robin Cohen. 1988. On the relationship between user models and discourse models. *Computational Linguistics*, 14(3).

Herbert L. Colston and Albert N. Katz, editors. 2005. *Figurative Language Comprehension: Social and Culturla Influences*. Lawrence Erlbaum Associates, Inc.

Costanza Conforti, Mohammad Taher Pilehvar, and Nigel Collier. 2018. Towards automatic fake news detection: Cross-level stance detection in news articles. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*.

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 250–259, Sofia, Bulgaria. Association for Computational Linguistics.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. In *EMNLP/IJCNLP*.

Adam Faulkner. 2014. Automated classification of stance in student essays: An approach using stance target information and the wikipedia link-based measure. In *Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference*.

Song Feng, Ritwik Bose, and Yejin Choi. 2011. Learning general connotation of words using graph-based algorithms. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Song Feng, Jun Seok Kang, Polina Kuznetsova, and Yejin Choi. 2013. Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1774–1784, Sofia, Bulgaria. Association for Computational Linguistics.

Anjalie Field, Gayatri Bhat, and Yulia Tsvetkov. 2019. Contextual affective analysis: A case study of people portrayals in online #metoo stories. *Proceeding of the 13th Internaltional AAAI Conference on Web and Social Media*.

Bilal Ghanem, Paolo Rosso, and Francisco Rangel. 2018. Stance detection in fake news a combined feature representation. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*.

David B. Guralnik. 1958. Connotation in dictionary definition. *College Composition and Communication*, 9:90–93.

Andreas Hanselowski, Avinesh PVS, Benjamin Schiller, Felix Caspelherr, Debanjan Chaudhuri, Christian M. Meyer, and Iryna Gurevych. 2018. A retrospective analysis of the fake news challenge stance-detection task. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1859–1874, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mareike Hartmann, Tallulah Jansen, Isabelle Augenstein, and Anders Søgaard. 2019. Issue framing in online discussion fora. In *Proceedings of 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Kazi Saidul Hasan and Vincent Ng. 2013. Stance classification of ideological debates: Data, models, features, and constraints. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1348–1356, Nagoya, Japan. Asian Federation of Natural Language Processing.

Kazi Saidul Hasan and Vincent Ng. 2014. Why are you taking this stance? identifying and classifying reasons in ideological debates. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Alexander Hoyle, Lawrence Wolf-Sonkin, Hanna M. Wallach, Ryan Cotterell, and Isabelle Augenstein. 2019. Combining sentiment lexica with a multi-view variational autoencoder. In *Proceedings of 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jun Seok Kang, Song Feng, Leman Akoglu, and Yejin Choi. 2014. ConnotationWordNet: Learning connotation over the Word+Sense network. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1544–1554, Baltimore, Maryland. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of the 2nd International Conference on Representation Learning*.

Manfred Klenner. 2017. An object-oriented model of role framing and attitude prediction. In *Proceedings of the 12th International Conference on Computational Semantics*.

Manfred Klenner and Simon Clematide. 2016. How factuality determines sentiment inferences. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics (*SEM@ACL*.

Manfred Klenner, Josef Ruppenhofer, Michael Wiegand, and Melanie Siegel. 2018. Offensive language without offensive words (olwow). In *Proceedings of KOVENS*.

Manfred Klenner, Don Tuggener, and Simon Clematide. 2017. Stance detection in Facebook posts of a German right-wing party. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 31–40, Valencia, Spain. Association for Computational Linguistics.

Robin Lakoff. 1973. The logic of politeness; or minding your p's and q's. In *9th Regional Meeting of the Chicago Linguistic Society*, Chicago.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland. Association for Computational Linguistics.

Chang Li, Aldo Porco, and Dan Goldwasser. 2018. Structured representation learning for online debate stance prediction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3728–3739, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Saif Mohammad. 2012. #emotional tweets. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 246–255, Montréal, Canada. Association for Computational Linguistics.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013a. NRC-canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 321–327, Atlanta, Georgia, USA. Association for Computational Linguistics.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013b. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013)*, Atlanta, Georgia, USA.

Saif M. Mohammad. 2018a. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. In *Proceedings of The Annual Conference of the Association for Computational Linguistics*, Melbourne, Australia.

Saif M. Mohammad. 2018b. Word affect intensities. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference*, Miyazaki, Japan.

Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiao-Dan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation*.

Saif M. Mohammad and Peter D. Turney. 2010. Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *Proceedings of 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Akiko Murakami and Raymond H. Putra. 2010. Support or oppose? classifying positions in online debates from reply activities and opinion expressions. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Robert Plutchik. 2001. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American Scientist*, 89(4):344–350.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Hannah Rashkin, Eric Bell, Yejin Choi, and Svitlana Volkova. 2017. Multilingual connotation frames: A case study on social media for targeted sentiment analysis and forecast. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 459–464, Vancouver, Canada. Association for Computational Linguistics.

Hannah Rashkin, Sameer Singh, and Yejin Choi. 2016. Connotation frames: A data-driven investigation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 311–321, Berlin, Germany. Association for Computational Linguistics.

Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *ArXiv*, abs/1707.03264.

David C. Rubin. 2006. The basic-systems model of episodic memory. *Perspectives on psychological science : a journal of the Association for Psychological Science*, 1 4:277–311.

Maarten Sap, Marcella Cindy Prasettio, Ari Holtzman, Hannah Rashkin, and Yejin Choi. 2017. Connotation frames of power and agency in modern films. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Roser Saurí and James Pustejovsky. 2009. Factbank: a corpus annotated with event factuality. *Language Resources and Evaluation*, 43:227–268.

Jenn Selby. 2014. Cate blanchett calls out red carpet sexism: “do you do this to the guys?”. In *Independent*.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirtieth Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*.

Dhanya Sridhar, James Foulds, Bert Huang, Lise Getoor, and Marilyn Walker. 2015. Joint models of disagreement and stance in online debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 116–125, Beijing, China. Association for Computational Linguistics.

Philip J. Stone and E. B. Hunt. 1963. A computer approach to content analysis: studies using the general inquirer system. In *AFIPS ’63 (Spring)*.

Yla R. Tausczik and James W. Pennebaker. 2009. The psychological meaning of words: Liwc and computerized text analysis methods. In *Journal of Language and Social Psychology*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Thirty-First Annual Conference on Neural Information Systems*.

Albert Webson, Zhizhong Chen, Carsten Eickhoff, and Ellie Pavlick. 2020. Are “undocumented workers” the same as “illegal aliens”? disentangling denotation and connotation in vector spaces. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Cynthia Whissell. 1989. The dictionary of affect in language. In *The Measurement of Emotions*, pages 113–131.

Cynthia Whissell. 2009. Using the revised dictionary of affect in language to quantify the emotional undertones of samples of natural language. *Psychological reports*, 105 2:509–21.

Chang Xu, Cécile Paris, Surya Nepal, and Ross Sparks. 2018. Cross-target stance classification with self-attention networks. In *Proceedings of The Annual Conference of the Association for Computational Linguistics*.

A Overview

The data and software are provided as supplementary material her: `https://github.com/connotationembeddingteam/connotation-embedding`.

B Connotation Labeling

We construct labels for our connotation lexicon (§3) using categories from the following existing resources: *HGI* – the Harvard General Inquirer (Stone and Hunt, 1963), *DAL* – the revised Dictionary of Affect in Language (Whissell, 2009), *CWN* – Connotation WordNet (Kang et al., 2014), and *NRCEmoLex* – the NRC Emotion Lexicon (Mohammad and Turney, 2013).

The HGI consists of 183 psycho-sociological categories. Each lexical entry ($\sim 11k$ total) is tagged with a non-zero number of categories. Different senses (noted through brief definitions) and parts-of-speech for the same word have separate entries. The available categories include valence (i.e., positive and negative), words related to a particular entity or social structure (e.g., institutions, communication), and value judgements (e.g., concern with respect).

The DAL consists of $\sim 8k$ words with scores for 3 categories: pleasantness, activation, and imagery. Word entries include inflection but do not explicitly mark part-of-speech. CWN is a lexicon of connotation polarity scores (ranging from 0 to 1) for $\sim 180k$ words, explicitly marked for part of speech. Finally, NRCEmoLex consists of word entries marked for any number the eight Plutchik emotions (anticipation, joy, trust, fear, surprise, sadness, disgust, anger) as well as positive and negative sentiment. Two versions of the lexicon are available: with and without sense level distinctions. Neither version includes explicit information on part-of-speech, and so we infer part-of-speech using the words provided to distinguish different senses.

We provide the complete distant labeling rules for each of the connotation aspects in Table 9 (see `http://www.wjh.harvard.edu/~inquirer/homecat.htm` for complete information on abbreviations). Within each connotation aspect, we determine the connotation polarity using the additional categories: *Positiv, Negativ, Strong, Weak, Hostile, Submit, Active* and *Power*.

Aspect	**General Inquirer Categories**
Social Value	PowGain, PowLoss, PowEnds, PowCon, PowCoop, PowAuPt, PowPt, PowAuth, PowOth, RcEthic, RcRelig, RcGain, RcEnds, RcLoss, Virtue, Vice, WltPt WltTran, WltOth, Food, Object, Doctrin, Academ, Work, NatrObj, Vehicle, Econ@, Goal, EnlPt, EnlOth, EnlLoss, SklPt, SklAsth, SklOth, Exprsv, Legal, COLL, Means MeansLw, Fail, Solve, EndsLw, Try, WlbPhys, WlbGain, WlbPt, WlbLoss, WlbPsyc, Quality, SocRel
Politeness	RspGain, RspLoss, RspOth, AffGain, AffLoss, AffOth, WlbPt, SklPt, EnlPt, Relig, WltPt, Polit, HU, Milit, Legal, Academ, Doctrin
Impact	PosAff, Pleasur, Pain, NegAff, Anomie, NotLw, Vice, Virtue, RcGain, RcLoss, RspLoss, RcEthic, RspOth, WlbPysc, RcEnds, EnlOth, WlbGain, RspGain, EnlGain, EnlEnds, EnlPt, WlbLoss, WlbPt, EnlLoss, SklOth, WlbPhys, Try, Goal, Work
Factuality	$v = \begin{cases} -1 & \text{if } x \leq -0.25 \\ 1 & \text{if } x \geq 0.25 \\ 0 & \text{otherwise} \end{cases}$ where x is the Imagery score normalized to $[-1, 1]$.
Sentiment	$v = \begin{cases} -1 & \text{if } x \leq -0.25 \\ 1 & \text{if } x \geq 0.25 \\ 0 & \text{otherwise} \end{cases}$ where x is the sentiment score normalized to $[-1, 1]$.

Table 9: Categories from the Harvard General Inquirer used in distant labeling connotations.

Aspect	**Avg Cohen's κ**
Social Value	0.526
Politeness	0.186
Impact	0.595
Factuality	0.164
Average	0.368

Table 10: Cohen's κ for agreement between the connotation lexicon and human annotators.

C Analysis of the Connotation Lexicon

In this section, we provide further analysis of the connotation lexicon as exemplification of its content and properties.

C.1 Human Evaluation

We show the instructions provided to annotators for the manual labeling of samples from the connotation lexicon in §3.3 (see Figures 5 and 6).

We include Cohen's $kappa$ score for agreement between the lexicon and human annotators for individual connotation aspects in Table 10.

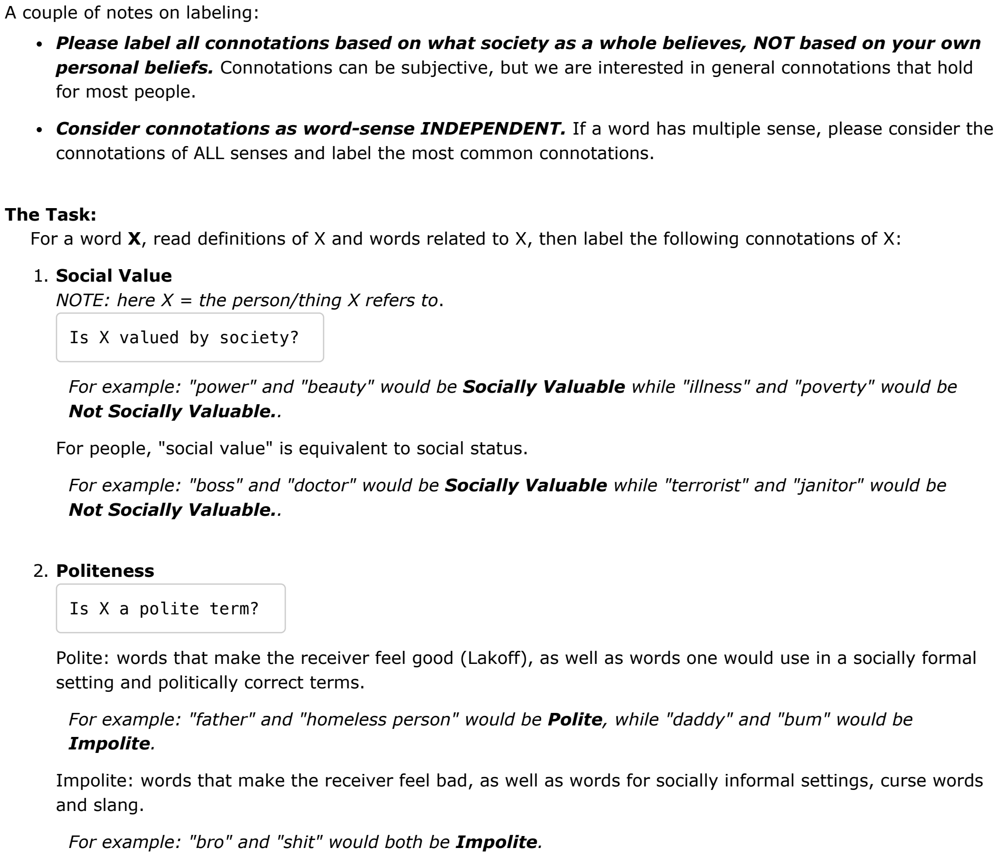
C.2 Gender Bias Analysis

Connotations have been used to study gender bias in movie scripts (Sap et al., 2017) and online media (Field et al., 2019). Here we use our connotation lexicon to analyze gender bias in two new domains: celebrity news (Celebrity) and student reviews of computer science professors (Professors).

We use existing datasets for these domains and the accompanying methodology of Chang and McKeown (2019) to infer word-level gender associations. Then, for the gender-associated words that are in our lexicon, for each connotation aspect and domain, we examine the percentage of positive and negative polarity words and find that these quantify known trends in gender-biased portrayals.

In the Celebrity domain, *Factuality* highlights the tendency of news media to focus on physical characteristics of female celebrities (Selby, 2014). More words with positive *Factuality* polarity (tangible concepts and attributes) are associated with women and more words with negative polarity (abstract concepts and attributes) are associated with men. For example, women are described as "beautiful" and "slim", while men are described as "political" and "presidential". In fact, even many of the not tangible female-associated words still align with physical attributes (e.g., "chic"), further emphasizing the biased portrayal.

We also find that in the Professors domain, patterns in *Social Value* and *Impact* agree with the observations of Chang and McKeown (2019) and with social science literature that finds male teachers are praised more than female teachers for being experts (both socially valuable and positively impacting society). For example, men are associated with posi-



A couple of notes on labeling:

- ***Please label all connotations based on what society as a whole believes, NOT based on your own personal beliefs.*** Connotations can be subjective, but we are interested in general connotations that hold for most people.
- ***Consider connotations as word-sense INDEPENDENT.*** If a word has multiple sense, please consider the connotations of ALL senses and label the most common connotations.

The Task:

For a word **X**, read definitions of X and words related to X, then label the following connotations of X:

1. **Social Value**

   NOTE: here X = the person/thing X refers to.

   `Is X valued by society?`

   *For example: "power" and "beauty" would be **Socially Valuable** while "illness" and "poverty" would be **Not Socially Valuable.**.*

   For people, "social value" is equivalent to social status.

   *For example: "boss" and "doctor" would be **Socially Valuable** while "terrorist" and "janitor" would be **Not Socially Valuable.**.*

2. **Politeness**

   `Is X a polite term?`

   Polite: words that make the receiver feel good (Lakoff), as well as words one would use in a socially formal setting and politically correct terms.

   *For example: "father" and "homeless person" would be **Polite**, while "daddy" and "bum" would be **Impolite**.*

   Impolite: words that make the receiver feel bad, as well as words for socially informal settings, curse words and slang.

   *For example: "bro" and "shit" would both be **Impolite**.*

Figure 5: First part of annotator instructions for connotation labeling in §3.3.

tive *Social Value* (socially valuable) words such as "knowledge" and "experience", while women are relatively less often associated with the same type of words.

Finally, in the Celebrity domain we find our lexicon reflects the coverage in the media of recent sexual harassment allegations against male celebrities. Namely, women are associated with more positive *Social Value* words and men are associated with many more negative *Social Value* words (see Table 11). Overall, our results quantitatively validate previous observations and known patterns of gender bias.

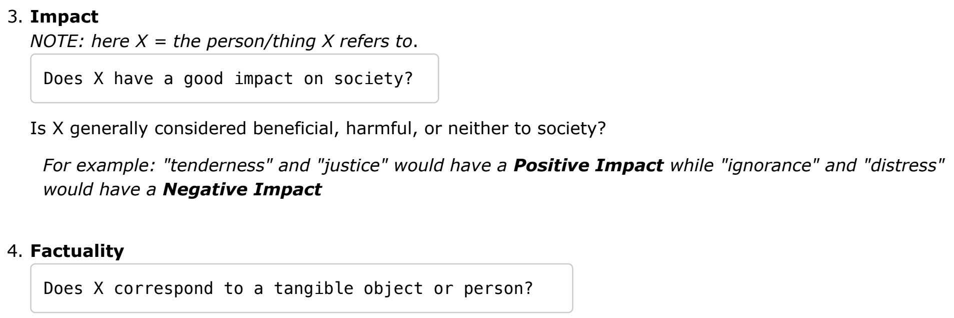
Aspect	Pol	F:M	Examples
Fact	+	54:27	F: beautiful, slim M: actor, film
	−	26:39	F: style, chic M: apology, political
Social Value	+	39:27	F: beauty, body, gold M: attempt, evidence
	−	3:24	F: blue, dancer, party M: abuse, allegation

Table 11: Gender bias examples in the Celebrity news domain. Pol is Polarity (tangible(+) vs. not tangible(−) and valuable (+) vs. not valuable (−)). F:M shows percent of female-associated words to percent of male-associated.

D Connotation Modeling

D.1 Data

We use dictionary definitions extracted from all available dictionaries on the Wordnik API: American Heritage Dictionary, CMU Pronouncing Dictionary, Macmillan Dictionary, Wiktionary, Webster's Dictionary, and WordNet. For labels, we use six aspects (see §3): *Social Value*, *Politeness*, *Impact*, *Factuality*, *Sentiment*, and *Emotional Association*. For verbs we use 11 aspects: perspective of the writer on the theme *P(wt)* and agent *P(wa)*, perspective of the agent on the theme *P(at)*, effect on the theme *E(t)* and agent *E(a)*, value of the theme *V(t)* and agent *V(a)*, mental state of the theme *S(t)* and agent *S(a)*, power, and agency.



3. **Impact**

NOTE: here X = the person/thing X refers to.

`Does X have a good impact on society?`

Is X generally considered beneficial, harmful, or neither to society?

For example: "tenderness" and "justice" would have a ***Positive Impact*** *while "ignorance" and "distress" would have a* ***Negative Impact***

4. **Factuality**

`Does X correspond to a tangible object or person?`

Figure 6: Second part of annotator instructions for connotation labeling in §3.3.

D.2 Hyperparameters

All models are trained with hidden size $H = 150$, number of definition words $N = 42$, number of related words $|R_{w^t}| = 20$ and dropout of 0.5 to prevent overfitting. For emotion prediction we set $\theta = 0.5$. We use Concept-Net numberbatch embeddings (Speer et al., 2016) because we find empirically that these outperform other pre-trained embeddings (GloVe and dependency-based embeddings (Levy and Goldberg, 2014)) on the development set.

We tune our only hyperparameters on the development set: the weights λ_a for the contribution of each loss term $\mathcal{L}^a$ to the total loss $\sum_a \lambda_a \mathcal{L}^a$ (see 4.1.3). We experiment with 10 manually selected weight combinations, where each $\lambda_a \in (0, 5)$. We find that the optimal weights are:

- $\lambda_a = 0.3$ for $a \in \{SocialVal, Impact, V(t), V(a), power, agency\}$
- $\lambda_a = 0.167$ for $a \in \{Fact, Sent\}$
- $\lambda_a = 1.0$ for $a \in \{P(wt), P(wa), P(at), E(t), E(a), S(t), S(a)\}$
- $\lambda_a = 0.5$ for $a = Polite$
- $\lambda_a = 3.0$ for $a = Emo$

Additionally, we compute expected validation performance (Dodge et al., 2019) on each connotation aspect individually (see Table 12).

D.3 Training

We optimize using Adam (Kingma and Ba, 2014) with learning rate 0.001 and minibatch-size of 64 for 80 epochs with early stopping. We optimize the parameters W^a, b^a for each noun and adjective aspect a separately from the parameters for each verb aspect a, allowing both to update the parameters of the definition encoder, and attention layer.

	Best Dev	$\mathbb{E}$[Dev]
Social Val	0.681	0.700
Polite	0.540	0.554
Impact	0.704	0.711
Fact	0.546	0.565
Sent	0.612	0.631
Emo	0.574	0.584
Avg	0.610	0.610
P(wt)	0.525	0.583
P(wa)	0.580	0.577
P(at)	0.606	0.613
E(t)	0.668	0.686
E(a)	0.596	0.618
V(t)	0.376	0.469
V(a)	0.463	0.459
S(t)	0.600	0.642
S(a)	0.611	0.603
power	0.466	0.493
agency	0.426	0.487
Avg	0.538	0.542
Avg	0.563	0.563

Table 12: Best macro-averaged F1 on the development set and expected validation score (Dodge et al., 2019) for our connotation representation learning model on all connotation aspects.

D.4 Detailed results

We present aspect-level results for the task of connotation label prediction (see Table13).

E Extrinsic Stance Evaluation

E.1 Dataset Details

We map the topic-stance annotations in the Internet Argument Corpus to individual topics and labels (e.g., ‘pro-life’ → topic ‘abortion’ with label ‘con’). We show dataset statistics in Table 14, where topics in the upper part are large sized, topics in the mid-

	Maj	**LR**	**CE+R (S)**	**CE (J)**	**CE+R (J)**
Social Val	.228	.664	.651	.632	**.651**
Polite	.311	.470	.467	**.518**	.464
Impact	.278	.669	.681	.687	**.704**
Fact	.271	.576	.531	.549	**.560**
Sent	.247	.585	.606	**.615**	**.615**
Emo	.487	.604	.599	.578	**.587**
Avg	.304	.594	.589	.597	.597
P(wt)	.246	.501	.437	**.481**	.439
P(wa)	.213	.564	.487	.544	**.583**
P(at)	.204	.649	.553	.623	**.629**
E(t)	.156	.721	.673	.655	**.661**
E(a)	.226	.573	.420	**.557**	.530
V(t)	.109	.369	.365	**.391**	.373
V(a)	.320	.449	.428	**.375**	.370
S(t)	.286	.640	.548	.586	**.629**
S(a)	.203	.551	.481	**.543**	.537
power	.294	.476	.467	.474	**.480**
agency	.182	.589	.515	**.505**	.490
Avg	.222	.553	.489	.521	.520
Avg	.251	.568	.524	.548	.547

Table 13: Macro-averaged F1 results for connotation prediction on the test set. The upper part shows noun/adjective aspect results, the bottom shows verb aspect results. Underline indicates the best performing model per row. **Bold** indicates the best performing joint learning model per row.

dle part are medium sized, and topics in the lower part are small sized.

E.2 Training Details

We split the data 60% train, 20% development, and 20% test. We train our models using pre-trained 100-dimensional word embeddings from GloVe (Pennington et al., 2014), as these are comparable to and more time-efficient than larger word embeddings. We use a hidden size of 60, dropout of 0.5, and train for 70 epochs with early stopping on the development set. We optimize Adam with learning rate 0.001 and minibatch-size of 64 on the cross-entropy loss. Our hyperparameters are set following Augenstein et al. (2016).

When truncating to medium size in §5.2, we truncate train topics to at most 2000 examples (`Trunc Train` and `Trunc All`) and development and test topics to at most 600 examples (`Trunc All`).

Topic	**# Ex**	**# C**	**# P**	**# N**
abortion	12453	3962	5236	3255
gay marriage	11037	2907	5082	3048
gun control	10119	4610	2681	2828
evolution	9896	2586	4480	2830
existence of God	7227	2588	2517	2122
death penalty	2834	995	951	888
humans are responsible	1608	560	538	510
marijuana legalization	1491	328	697	466
communism is better than capitalism	1279	618	277	384
illegal immigration	291	108	87	96
health care reform	201	76	51	74
legalize prostitution	199	57	88	54
Israel	100	29	38	33
vegetarian diet is best	79	29	29	21
women in combat	47	15	19	13
minimum wage	27	9	8	10
Overall	58888	19477	22779	16632

Table 14: Statistics for the stance detection dataset. C indicates ‘con’, P indicates ‘pro’, N indicates ’neutral’.

E.3 Topic Stance Analysis

We present a detailed analysis of the results of the models `BiC+W` and `BiC+C` on the stance detection on individual topics. First, we find that when the models are trained with all of the data (All Data), there are statistically significant differences on only two topics, one of which is very small (see Table 15a). This is further evidence that the models are comparable in this setting.

We then find that when trained with truncated training data (see §5.2 for details) (`Trunc Train`), `BiC+C` improves over `BiC+W` on six topics, including four of the medium or truncated large topics (see Table 15b). When trained and evaluated with truncated data (`Trunc All`), `BiC+W` and `BiC+C` have statistically significant improvements over each other on the same number of topics (two each) but `BiC+C` is significantly better overall (see

Table 15c). These results further show that connotations help to learn stance when data is limited.

Topic	`BiC +W`	`BiC +C`
abortion	.49	.49
gay marriage	.47	.48
gun control	.53	.55
evolution	.43	.44
existence of God	.52	.52
death penalty	.45	.50†
humans are responsible	.49	.54
marijuana legalization	.51	.50
communism is better than capitalism	.52	.54
illegal immigration	.34	.38
health care reform	.64	.91
legalize prostitution	.40	.53
Israel	.66	.42
vegetarian diet is best	.52†	.33
women in combat	.28	.30
minimum wage	.30	.22
Overall	.57	.57

(a) On `All Data`.

Topic	`BiC +W`	`BiC +C`
abortion	.46	.47
gay marriage	.48	.46
gun control	.50	.55*
evolution	.41	.43*†
existence of God	.48	.51*
death penalty	.48	.50
humans are responsible	.45	.53*†
marijuana legalization	.51	.50
communism is better than capitalism	.53	.54
illegal immigration	.45*	.36
health care reform	.62	.64*†
legalize prostitution	.49	.50
Israel	.54	.44
vegetarian diet is best	.10	.11*
women in combat	.52	.36
minimum wage	.22	.33
Overall	.54	.56*†

(b) On `Trunc Train`.

Topic	`BiC +W`	`BiC +C`
abortion	.49	.48
gay marriage	.49*	.46
gun control	.51	.50
evolution	.43	.43
existence of God	.49	.47
death penalty	.48	.46
humans are responsible	.46	.55*†
marijuana legalization	.50	.49
communism is better than capitalism	.55	.52
illegal immigration	.44	.44
health care textit reform	.54	.64*†
legalize prostitution	.47	.51
Israel	.41	.56
vegetarian diet is best	.52*†	.50
women in combat	.43	.47
minimum wage	.33	.35
Overall	.54	.56*†

(c) On `Trunc All`.

Table 15: Macro F1 results on the test set for three different data scenarios. * indicates significance with $p < 0.05$, † indicates significane with $p < 0.01$.

`FAST`: Financial News and Tweet Based Time Aware Network for Stock Trading

Ramit Sawhney*
Netaji Subhas Institute of Technology
`ramits.co@nsit.net.in`

Arnav Wadhwa*
MIDAS, IIIT Delhi
`arnavw96@gmail.com`

Shivam Agarwal
Manipal Institute of Technology
`shivamag99@gmail.com`

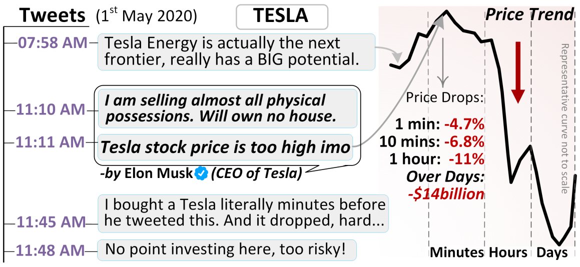
Rajiv Ratn Shah
IIIT Delhi
`rajivratn@iiitd.ac.in`

Abstract

Designing profitable trading strategies is complex as stock movements are highly stochastic; the market is influenced by large volumes of noisy data across diverse information sources like news and social media. Prior work mostly treats stock movement prediction as a regression or classification task and is not directly optimized towards profit-making. Further, they do not model the fine-grain temporal irregularities in the release of vast volumes of text that the market responds to quickly. Building on these limitations, we propose a novel hierarchical, learning to rank approach that uses textual data to make time-aware predictions for ranking stocks based on expected profit. Our approach outperforms state-of-the-art methods by over 8% in terms of cumulative profit and risk-adjusted returns in trading simulations on two benchmarks: English tweets and Chinese financial news spanning two major stock indexes and four global markets. Through ablative and qualitative analyses, we build the case for our method as a tool for daily stock trading.

1 Introduction

The stock market, a financial ecosystem involving quantitative trading and investing, observed a market capitalization exceeding $US 60 trillion as of 2019.[1] Stock trading presents lucrative opportunities for investors to utilize the market as a platform for investing funds and maximizing profits. However, making profitable investment decisions is challenging due to the market's volatile and rapid-changing nature (Adam et al., 2016; Foucault et al., 2016). Research at the intersection of natural language processing (NLP) and finance presents encouraging prospects in stock prediction (Jiang, 2020). Conventional work forecasts future trends by modeling numerical historic stock data (Lu et al., 2009; Bao et al., 2017). However, price signals alone can not capture market surprises, news, and company announcements. Such events, often reported across financial news and social media, have shown to influence market dynamics (Laakkonen, 2004). As shown in Figure 1, prices immediately react to breaking news about the related company. Such reactions conform to the Efficient Market Hypothesis (EMH), which states that financial markets are *informationally efficient* and prices reflect all available information (Malkiel, 1989).



07:58 AM – Tesla Energy is actually the next frontier, really has a BIG potential.

11:10 AM – *I am selling almost all physical possessions. Will own no house.*

11:11 AM – *Tesla stock price is too high imo*

-by Elon Musk *(CEO of Tesla)*

11:45 AM – I bought a Tesla literally minutes before he tweeted this. And it dropped, hard...

11:48 AM – No point investing here, too risky!

Figure 1: Here, we study how Tesla's tweets influence investors' opinions about the company and impact its stock price trend. The first tweet shows positive opinions, and we observe a rise in prices. Later, the tweets made by the CEO *rapidly* lead to drastic price drops within minutes. Further, without a sequential context, it gets challenging to understand the tweets that follow.

The abundance of stock affecting information across news and Twitter helps investors analyze market trends and inspires the adoption of NLP to study the interplay between textual data and stock prices (Xu and Cohen, 2018; Oliveira et al., 2017). However, unlike structured numerical data, analyzing natural language poses various challenges. First, analyzing individual tweets or news headlines may not be informative enough. They often exhibit a sequential context-dependency, where analyzing them together can provide a greater unified context,

---

*Equal contribution.

[1] World Federation of Exchanges: https://data.worldbank.org/indicator/CM.MKT.LCAP.CD/

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2164–2175
April 19 - 23, 2021.

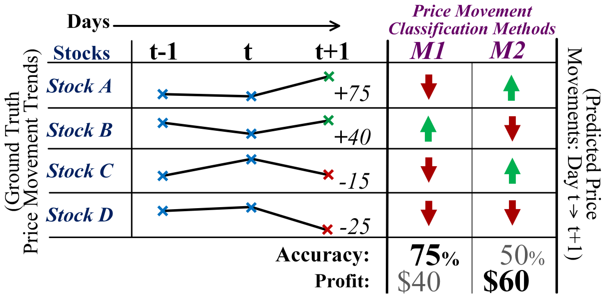


Figure 2: More accurate methods M_1 (higher accuracy) may not always be more profitable than less accurate methods M_2 (lower accuracy). Profit is gained by selling stocks having ⬆ prediction for price movement from trading day t to t+1.

as shown in Figure 1. Despite the success of recurrent neural networks (RNNs) in modeling such a sequential context (§2), a critical drawback is that they assume all text to be equally spaced in time, ignoring the inherent dynamic timing irregularities of social media and news. Timing plays a critical role as stock markets rapidly react to new information (Foucault et al., 2016), leading to significant price changes within minutes, as shown in Figure 1. Scholtus et al. (2014) show that reacting one second slower than other market participants can lead to a loss in thousands of dollars. Further, not each text holds the potential to influence stock prices, texts have a *diverse influence* on stock prices based on their content, such as breaking news or tweets from a reliable source, as opposed to noise like vague comments as shown in Figure 1. These observations mandate the need to factor in the *time-aware dependencies* and *diverse influence* in analyzing online natural language data for stock trading.

Despite profitability being the prime objective of trading, NLP methods for stock prediction (Xu and Cohen, 2018; Hu et al., 2017) are commonly framed as classification or regression tasks, and are not directly optimized towards profitable stock selection. Consider the toy example in Figure 2 that shows how methods having a higher classification accuracy may not always lead to higher overall profits. This research gap in NLP methods for stock prediction presents a new direction for stock selection, where both predictive performance and profits are jointly and directly optimized.

Contributions: We formulate stock prediction as a learning-to-rank problem (§**3.1**) and present `FAST`: **F**inancial News and Tweet based Time **A**ware Network for **S**tock **T**rading, which uses text for maximizing profit by jointly optimizing predictive power and the optimal ranking of stocks. `FAST` learns *time-aware* representations of financial news and tweets, and captures *relevant* market signals using hierarchical temporal attention for ranking stocks (§**3**). Through experiments (§**4**) on English and Chinese text corresponding to the NASDAQ, Shanghai, Shenzhen, and Hong Kong markets, we show that `FAST` outperforms state-of-the-art methods in terms of intraday returns by over 8% and risk-adjusted returns by over 10% (§**5.1**, §**5.2**). Further, through exploratory (§**5.3**, §**5.4**) and qualitative (§**6**) analyses, we demonstrate the practical applicability of `FAST` to daily stock trading.

2 Background

Conventional Methods: Stock prediction spans various methods, commonly framed as regression or classification tasks (Jiang, 2020). Conventional methods rely on numeric features like historical prices (Kohara et al., 1997; Lin et al., 2009), technical (Shynkevich et al., 2017), and macroeconomic indicators (Hoseinzade et al., 2019). These include discrete (Bollerslev, 1986), continuous (Andersen, 2007), and neural approaches (Zhang et al., 2017; Feng et al., 2019a). Despite their success, a limitation of these methods is that they are limited to numerical features and do not factor crucial stock influencing factors such as text (Lee et al., 2014).

Contemporary Methods: Newer models based on the EMH, leverage natural language features extracted from investor sentiments (Li and Shah, 2017), financial reports (Kogan et al., 2009; Rekabsaz et al., 2017), earnings calls (Qin and Yang, 2019), online news (Peng and Jiang, 2016; Chen et al., 2019a,b; Du and Tanaka-Ishii, 2020) and social media (Si et al., 2013; Tabari et al., 2018; Sawhney et al., 2020a) for stock price regression and movement classification tasks. These methods show how NLP can complement conventional price-based methods in capturing the effect of events like market surprises and mergers over stock returns. However, these methods do not directly optimize profit, and do not factor the fine-grain irregularities in release times of stock affecting text. For stock trading, the timing of release of information across these sources plays a critical role, as price changes rapidly factor all publicly available information (Norman, 2014). Firms may exploit investors' perception of market information (Forbes, 2009), for instance, by timing the release

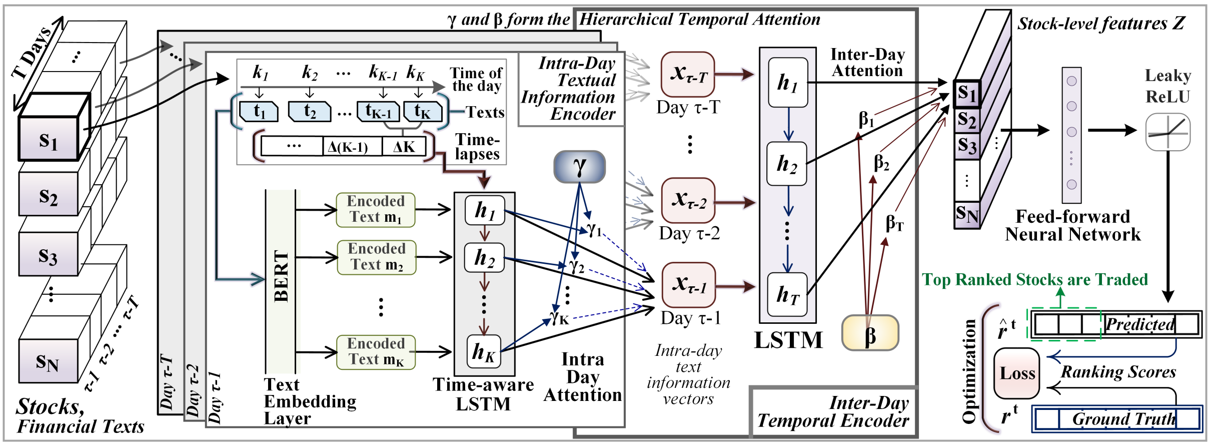


Figure 3: `FAST`: Time-aware modeling, hierarchical temporal attention, joint optimization for ranking.

of negative news between positive ones to minimize losses (Segal and Segal, 2016). These limitations hinder contemporary methods from modeling a time-aware progression of stock-affecting market signals to directly optimize profit generation.

Time-aware Methods: Recently, time-aware modeling of time series data has shown improvements over conventional sequential models like RNNs and LSTMs on various tasks such as patient subtyping (Baytas et al., 2017), suicide ideation and buildup detection using Twitter history (Sawhney et al., 2020b), disease progression (Gao et al., 2020), and much more. However, modeling the temporal dynamics inherent in social media and online news is complex as it involves noisy and diversely influential data across irregular time intervals. The intersection of modeling the temporal dynamics of natural language with finance presents an underexplored yet promising research avenue.

3 Proposed Approach: `FAST`

3.1 Problem Formulation: Stock Ranking

We adopt a learning to rank formulation for stock selection. Let $S = \{s_1, s_2, \dots, s_N\}$ represent a set of N stocks, where for every stock $s_i \in S$, on a trading day τ, there is an associated closing price p^τ and a one-day return ratio $r_i^\tau = \frac{p^\tau - p^{\tau-1}}{p^{\tau-1}}$. On any given trading day τ, there exists an optimal *ranking* $Y^\tau = \{y_1^\tau > y_2^\tau \cdots > y_N^\tau\}$ of the stocks, such that a total order exists between the ranks $y_i^\tau > y_j^\tau$ for any two stocks $s_i, s_j \in S$, provided $r_i^\tau > r_j^\tau$. Such an ordering of stocks S on a trading day τ represents a ranking list, where stocks achieving higher ranking scores Y are expected to generate a higher investment revenue (profit) on day τ. Formally, given stock-relevant textual data (financial news or tweets) for a lookback period of length T days (i.e., days $\in [\tau - T, \tau - 1]$), we aim to learn a ranking function that outputs a score $\hat{r}^\tau$ to rank each stock s on day τ in terms of their expected profit.

We now describe the components of `FAST` as shown in Figure 3, for *hierarchically* and *attentively* learning *time-aware* representations of news and tweets within (§**3.2**) and across (§**3.3**) the days in the lookback period. Lastly, we optimize `FAST` to rank stocks in terms of expected profitability (§**3.4**) for daily stock trading.

3.2 Intra-Day Textual Information Encoder

To model the news or tweets over a day, `FAST` first encodes the texts via an embedding layer.

Text Embedding Layer: Owing to the success of transfer learning and pre-training of language models in NLP, we use Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) to encode the texts. BERT has shown to capture more contextual text representations as opposed to methods like word2vec (Hu et al., 2017), GloVe (Xu and Cohen, 2018), ELMO (Mohammadi et al., 2019). We encode each text t to a higher dimensional representation $m = \text{BERT}(t) \in \mathbb{R}^d$ where $d = 768$, obtained by averaging the token level outputs from the final layer of BERT.

Learning Stock Representations for One Day: For each stock s on any given day i, a variable number of texts (news or tweets) $[t_1, t_2, \dots t_K]$ are posted at times $[k_1, k_2, \dots k_K]$ that may discuss news or express sentiments towards the stock. We encode each of the K texts posted in a day, using

BERT as $[m_1, m_2, \dots m_K]$. Often, analyzing a single text alone may not be informative enough to analyze a stock (Barber and Odean, 2007). Whereas, analyzing a *sequence* of texts released over the day provides a unified context to gain a more informed understanding of the performance of a stock (Hu et al., 2017).

RNNs, particularly LSTMs are a natural way to capture such *sequential context dependencies* in tweets and news over time (Akhtar et al., 2017). However, a standard LSTM assume inputs (texts) to be equally spaced sequences in time. In contrast, the time interval between news releases or tweets can vary widely, from a few seconds to many hours that can have a drastic impact on their influence on the market (Robertson et al., 2007; O'Hara, 2015). Consequently, news and opinions may change substantially over a day. Capturing the fine-grained granularities in the posting times of online text can lead to better and quicker reactions to market opportunities and increased profits.[2]

Since timing serves as a crucial factor to model the *progression* of market data (Tafti et al., 2016), we propose the use of a **time-aware LSTM** (t-LSTM) (Baytas et al., 2017), by modifying a standard LSTM. We feed the time between texts to the t-LSTM cell to model the **temporal irregularities** in news and tweets. The t-LSTM applies a decay to the short-term memory in the LSTM according to the time elapsed between the release of two successively posted texts. Formally, the t-LSTM adopts a decaying function of elapsed time, transforming the time differences into appropriate weights for each input as:

$$
\begin{aligned}
C^S_{k-1} &= tanh(W_d C_{k-1} + b_d) && \text{(Short-term memory)}\\
\tilde{C}^S_{k-1} &= C^S_{k-1} * g(\Delta k) && \text{(Discounted short-term memory)}\\
C^{LT}_{k-1} &= C_{k-1} - C^S_{k-1} && \text{(Long-term memory)}\\
C^{*}_{k-1} &= C^{LT}_{k-1} + \tilde{C}^S_{k-1} && \text{(Adjusted previous memory)}
\end{aligned}
$$

where C_{k-1} is the previous cell memory, $\{W_d, b_d\}$ are the network parameters, Δk is the elapsed time between two financial news or tweets $[t_k, t_{k-1}]$, and $g(\cdot)$ is a heuristic decaying function. We select $g(\Delta k) = 1/\Delta k$ empirically as suggested by Baytas et al. (2017). Intuitively, the greater the elapsed time between two news or tweets, the lesser impact they should have on each other due to the market's dynamic nature. The t-LSTM computes the current hidden state h_k for each input text t_k generated in a day, as:

$$
\begin{aligned}
\tilde{C} &= tanh(W_c m + U_c h_{k-1} + b_c) && \text{(Candidate memory)}\\
C_k &= f_k * C^{*}_{k-1} + i_k * \tilde{C} && \text{(Current memory)}\\
h_k &= o_k * tanh(C_k) && \text{(Current hidden state)}
\end{aligned}
$$

where $\{W_c, U_c, b_c\}$ are the network parameters of the candidate memory $\tilde{C}$, m is the embedding for text t_k and $\{i_k, f_k, o_k\}$ are input, forget and output gates. We encode the texts for each stock s on day i using the t-LSTM as:

$$h_j = \text{t-LSTM}(m_j, \Delta j, h_{j-1}); j \in [1, K] \tag{1}$$

where the hidden state h_j represents the current text j as well as the preceding texts while focusing on text j in a time-aware fashion. All news and tweets released in a day may not be equally informative, and have *diverse influence* over a stock's trend (Barber and Odean, 2007). We use an intra-day attention mechanism (Luong et al., 2015) to emphasize texts likely to have a more substantial influence on the price. As shown in Figure 3, the intra-day attention mechanism learns to adaptively aggregate the variable number of hidden states of the t-LSTM (due to a variable number of texts per day) into an *intra-day text information vector* x_i as:

$$x_i = \sum_j \gamma_j h_j, \gamma_j = \frac{\exp\left(h_j^{\mathrm{T}}(W\bar{h}_m)\right)}{\sum_{j=1}^{K} \exp\left(h_j^{\mathrm{T}}(W\bar{h}_m)\right)} \tag{2}$$

where $\bar{h}_m \in \mathbb{R}^{K \times d_m}$ denotes the concatenation of all the hidden states from the t-LSTM, d_m is the dimension of each hidden state, γ_j represents the learned attention weights, and W is a parameter.

3.3 Inter-Day Temporal Encoder

We now combine the representations learned from texts in each day across multiple days in a lookback period. We combine these representations in a *hierarchical* manner within and across days, using the sequence of intra-day text information vectors x. Since days are equally spaced in time, we first feed the vectors x to an LSTM layer as:

$$h_i = \text{LSTM}(x_i, h_{i-1}); \quad \tau - T \le i \le \tau - 1 \tag{3}$$

where, h_i is the hidden state representation for day i. However, as per the the adaptive market hypothesis (Lo, 2004), tweets and news published across different days have shown to have a varying impact on stock prices (Calvet and Fisher,

---

[2] Every millisecond lost results in \$100m per annum in lost opportunity. Details: `https://en.wikipedia.org/wiki/Low_latency_(capital_markets)`

2007), due to financial phenomena such as calendar anomalies (Jacobs and Levy, 1988), the week-day effect (Berument and Kiymaz, 2001), etc. To selectively weigh critical days, we employ an inter-day attention mechanism (Luong et al., 2015). The inter-day attention aggregates representations across all days into an *overall representation* z_τ using the learned attention weights β_i for day i as:

$$z_\tau = \sum_i \beta_i h_i, \beta_i = \frac{\exp\left(h_i^{\mathrm{T}}(W\bar{h}_z)\right)}{\sum_{i=1}^{T} \exp\left(h_i^{\mathrm{T}}(W\bar{h}_z)\right)} \tag{4}$$

where, W is a learned linear transform, $\bar{h}_z \in \mathbb{R}^{T \times d_z}$ represents the concatenated hidden states, d_z is the size of output space of LSTM. The inter-day and the intra-day attention together comprise a **hierarchical temporal attention**. FAST captures time-aware dependencies in large volumes of *chaotic* text to rank stocks, as described next.

3.4 Ranking and Network Optimization

To optimize FAST for stock ranking, we first concatenate the temporal representations z_τ obtained for each stock s to form stock-level features Z. We then feed Z to a feed-forward neural network followed by a Leaky-ReLU activation (Maas et al., 2013) which outputs the predicted return ratio $\hat{r}^\tau$ for stock ranking. We optimize FAST through a joint point-wise regression and pairwise rank-aware loss L, to minimize the differences between the predicted and the actual return ratios, while maintaining the relative order of the top-ranked stocks as:

$$L = \|\hat{r}^\tau - r^\tau\|^2 + \phi \sum_{i=1}^{N}\sum_{j=1}^{N} \max\left(0, -\left(\hat{r}_i^\tau - \hat{r}_j^\tau\right)\left(r_i^\tau - r_j^\tau\right)\right) \tag{5}$$

where $\hat{r}^\tau$ and r^τ are the predicted and actual scores for ranking stocks on day τ and ϕ is a loss weighing parameter.

4 Experimental Setup

4.1 Datasets and Stock Markets

US S&P 500[3] (Xu and Cohen, 2018): Comprises 109,915 *English tweets* from social media platform Twitter spanning January 2014 to December 2015, related to 88 high-trade-volume-stocks from the *NASDAQ* Stock Exchange forming the *S&P 500* index. Xu and Cohen (2018) extract stock specific tweets using regex queries made of stock tickers (e.g., $AAPL for Apple, $ is a *cashtag* on Twitter).

China & Hong Kong[4] (Huang et al., 2018): Comprises 90,361 financial *news headlines* in *Chinese* spanning from January to December 2015, aggregated by *Wind*[5] related to 85 top-traded *China A-share* stocks in Shanghai, Shenzhen and the Hong Kong Stock Exchange. Huang et al. (2018) extract corporate news from major financial websites.

Pre-processing: We pre-process English tweets using NLTK (Twitter mode), for treatment of URLs, identifiers (@) and hashtags (#). We adopt the Bert-Tokenizer for tokenization. For the English tweets, we use the pre-trained BERT-base-cased.[6] For the Chinese news, we adopt the Chinese-BERT.[6] We collect historical prices for all stocks from Yahoo Finance.[7] We align trading days by dropping samples that lack tweets for a consecutive 5-day trading window, and further align the data across trading windows for stocks to ensure data is available for all trading days in the window for the same set of stocks. We split the US S&P 500 temporally based on date ranges from 01/01/2014 to 31/07/2015 for training, 01/08/2015 to 30/09/2015 for validation, and 01/10/2015 to 01/01/2016 for testing. We split the China & HK dataset temporally based on date ranges from 01/01/2015 to 31/08/2015 for training, 01/09/2015 to 30/09/2015 for validation, and 01/10/2015 to 01/01/2016 for testing all models.

4.2 FAST Training Setup

We conduct all experiments on a Tesla P100 GPU. We use grid search to find optimal hyperparameters based on the validation Sharpe Ratio (§**4.3**) for all models. We explore lookback window length $T \in \text{range}[2, 10]$ (best $T = 5$), loss weighing factor $\phi \in \text{range}[1, 10]$ (best $\phi = 4$), and the hidden state dimension for both t-LSTM and LSTM $d \in [32, 64, 128]$ (best $d = 64$) for both datasets. We use Xavier initialization (Glorot and Bengio, 2010) to initialize all weights. We use an exponential learning rate scheduler (Li and Arora, 2019) with a decay rate of 0.05 and an initial learning rate of $5e-4$. We train FAST end-to-end using the Adam optimizer (Kingma and Ba, 2014) for 500 epochs requiring 8 hours of compute time.

---

[3] US S&P 500 dataset: `www.github.com/yumoxu/stocknet-dataset`

[4] China & Hong Kong dataset: `https://pan.baidu.com/s/1mhCLJJi`

[5] `https://www.wind.com.cn/en/wft.html`

[6] `www.github.com/google-research/bert`

[7] Prices from: `https://finance.yahoo.com/`

4.3 Evaluation Metrics and Trading Strategy

Returns: To assess the profit generation ability of all methods (§4.4), we compute the **Sharpe ratio (SR)**, a measure of the return of a portfolio compared to its risk (Sharpe, 1994), and the cumulative **investment return ratio (IRR)**. Following Feng et al. (2019b), we adopt a daily *buy-hold-sell* trading strategy, that is, when the market closes on trading day $\tau-1$, the trader uses the method to get a ranked list of the predicted return ratio for each stock. The trader then buys the top η stocks and then sells the bought stocks on the market close of the trading day τ. The IRR on any day τ is defined as, $\text{IRR}^{\tau}=\sum_{i\in\mathcal{S}^{\tau-1}}\frac{p_i^{\tau}-p_i^{\tau-1}}{p_i^{\tau-1}}$ where, $\mathcal{S}^{\tau-1}$ denotes the set of stocks in the portfolio on day $\tau-1$ and p_i^{τ}, $p_i^{\tau-1}$ are the closing price of the stock i on days τ and $\tau-1$ respectively. We calculate SR by computing the earned return R_a in excess of the risk-free return[8] R_f, defined as: $\text{SR}=\frac{E[R_a-R_f]}{std[R_a-R_f]}$.

Ranking: We also evaluate the stock ranking ability of `FAST` using two widely-used ranking metrics: **Mean Reciprocal Rank (MRR)** and **Normalized Discounted Cumulative Gain (NDCG@η)**. MRR is the reciprocal rank of the first relevant stock while, NDCG@η sums the true scores for the top η stocks, ranked in the order induced by the predicted scores, after applying a logarithmic discount. For both returns and NDCG, we report results for top $\eta=5$ stocks, and present performance variations with different values of top stocks η (§5.4).

4.4 Baselines

We compare `FAST` with baselines spanning different formulations: *regression*, *classification*, *reinforcement learning*, and *ranking*. We follow the same preprocessing protocols as proposed by the works and adopt their implementation, if available.

***Regression* (REG)** These methods regress return ratios from past data and trade the top stocks.

- **AZFinText:** Proper noun-based text representations fed to Support Vector Regression for forecasting return ratios (Schumaker and Chen, 2009).
- **W-LSTM:** LSTMs with stacked autoencoders that encode noise-free data obtained through wavelet transform of historic prices (Bao et al., 2017).

***Classification* (CLF)** The following methods classify movements as $[up, down, neutral]$ and trade the stocks where prices are expected to rise.

- **TSLDA:** Topic Sentiment Latent Dirichlet Allocation, a generative model jointly exploiting topics and sentiments in text (Nguyen and Shirai, 2015).
- **CH-RNN:** An RNN with cross-modal attention on price trends and texts across days (Wu et al., 2018).
- **StockEmb:** Stock embeddings acquired using prices, and dual vector (word-level, context-level) representation of texts (Du and Tanaka-Ishii, 2020).
- **SN - HFA:** StockNet - HedgeFundAnalyst, a variational autoencoder with attention on texts and prices (Xu and Cohen, 2018).
- **HAN:** A Hierarchical Attention Network using GRU encoders with temporal attention on texts and days in the lookback period (Hu et al., 2017).

***Reinforcement Learning* (RL)** The following approaches optimize quantitative trading through reinforcement learning.

- **iRDPG:** An imitative RDPG algorithm exploiting temporal stock price features, while optimizing the Sharpe Ratio as the reward (Liu et al., 2020).
- **S-Reward:** Inverse-RL method to model relations between sentiments and returns (Yang et al., 2018).

***Ranking* (RAN)** The following methods rank stocks to select most profitable trading candidates.

- **R-LSTM:** Utilizes 5, 10, 20, 30-day averages and closing prices to train an LSTM (Feng et al., 2019b).
- **RankNet:** A DNN that utilizes sentiment-based shock and trend scores to optimize a probabilistic ranking function (Song et al., 2017).

---

[8] T-Bill rates: `https://home.treasury.gov/`

`FAST`: **Model Components**			**US S&P 500**				**China & Hong Kong**			
Intra-Day Encoder	***Intra-Day Attention***	***Inter-Day Attention***	***MRR***↑	***IRR***↑	***NDCG***↑	***SR***↑	***MRR***↑	***IRR***↑	***NDCG***↑	***SR***↑
s-LSTM	–	–	0.050	0.794	0.521	0.571	0.072	0.861	0.580	0.663
LSTM	✗	✗	0.056	0.913	0.582	0.654	0.074	0.902	0.623	0.714
LSTM	✗	✓	0.060	0.922	0.611	0.713	0.080	1.027	0.695	0.773
LSTM	✓	✗	0.063	1.097	0.630	0.865	0.090	1.148	0.728	0.901
LSTM	✓	✓	0.064	1.214*	0.725*	0.919	0.096*	1.244*	0.756*	0.965*
t-LSTM	✗	✗	0.063	1.131	0.621	0.842	0.095	1.160	0.703	0.911
t-LSTM	✗	✓	0.065*	1.224*†	0.703	0.933*	0.109*	1.295*	0.759*	1.018*
t-LSTM	✓	✗	*0.067**†	*1.267**†	*0.760**	*0.949**†	*0.114**†	*1.383**†	*0.804**†	*1.165**†
t-LSTM	✓	✓	**0.068***†	**1.336***†	**0.762***†	**0.957***†	**0.117***†	**1.441***†	**0.810***†	**1.193***†

Table 1: Ablation study over components of `FAST`. s-LSTM is a single LSTM layer, which models all texts within and across days. All model variations except s-LSTM use LSTM as the Temporal Encoder. Intense color indicates better results. **Bold**, *italics* indicate **best**, *second best* results. $\star$ & † imply statistically significant (p<0.001) improvements over s-LSTM & RankNet (Wilcoxon's signed rank test). ↑ indicates that higher values are better.

			US S&P500		**China&HK**	
	Model & Components		***IRR***↑	***SR***↑	***IRR***↑	***SR***↑
REG	AZFinText	T + P	0.42	0.40	0.49	0.50
	W-LSTM	P	0.40	0.41	0.47	0.49
CLF	TSLDA	T + P	0.40	0.39	0.50	0.51
	CH-RNN	T + P + A	0.78	0.54	0.83	0.70
	StockEmb	T + P + A	0.70	0.51	0.89	0.74
	SN - HFA	T + P + A	1.09	0.81	1.12	0.93
	HAN	T + A	1.07	0.80	1.20	1.01
RL	iRDPG	P	1.05	0.79	1.19	1.03
	S-Reward	T	0.93	0.73	1.33	1.08
RAN	R-LSTM	P	1.01	0.78	1.17	0.96
	RankNet	T	1.16*	0.87*	1.14	0.95
	`FAST`	T + A	1.34*†	0.96*†	1.44*†	1.19*†

Table 2: Profitability comparison with baselines (§4.4). Within Components, T = Text, P = Prices, A = Attention across modalities. Green & Blue depict Best & Second-best (SOTA) results. $\star$ & † imply statistically significant (p<0.001) improvements over S-Reward and RankNet (under Wilcoxon's signed rank test).

5 Results and Discussion

5.1 Profitability Comparison with Baselines

As the ultimate goal of stock prediction is profit, we compare the profitability of `FAST` against baseline methods in Table 2. `FAST` generates significantly ($p < 0.001$) higher cumulative and risk-adjusted returns than all methods. Overall, we observe that RL and ranking methods are more profitable as they are directly optimized towards profit generation through stock selection. This observation validates the premise of formulating stock prediction as a learning-to-rank problem, compared to conventionally adopted regression and classification tasks.

Further, we find methods that study stock affecting information from news and tweets, generate profits higher or comparable to methods that only use historical prices. These improvements revalidate the effectiveness of leveraging textual sources to capture stock affecting signals like market surprises, announcements (mergers, acquisitions) and public sentiment. We attribute the higher profitability of `FAST` to two major reasons. First, through its hierarchical temporal attention mechanism, `FAST` captures the diverse influence of different texts and days over stock movements. Second, as `FAST` is time-aware, it models the influence of fine-grain temporal irregularities in the release of financial news and tweets over stock movements.

The test periods of the US S&P 500 and the China & Hong Kong datasets span diverse market conditions. The China & Hong Kong test period covers the 2015-16 China Stock Market Turbulence (Liu et al., 2016), a bearish market scenario,[9] and that of US S&P 500 covers standard market conditions. We find that `FAST` is profitable and outperforms existing baselines over such diverse market scenarios. Next, we further probe the performance of `FAST` through a series of ablative experiments.

5.2 `FAST` Component Ablation

Table 1 shows how `FAST`'s stock ranking ability and profitability benefits from each of its components. On just feeding all data into a single LSTM, we observe poor performance. As we adopt the intra-day (LSTM) and the temporal (LSTM) encoders, we observe higher profits that suggest benefit in modeling the sequential context of texts hierarchi-

---

[9] Bearish markets are those that experience prolonged price declines, and experience high volatility and risk on investments.

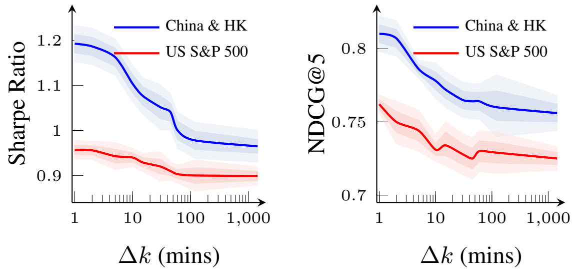


Figure 4: Profitability and ranking performance against granularities of time difference Δk adopted by t-LSTM

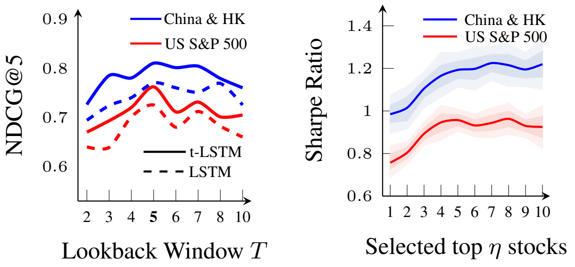


Figure 5: Sensitivity to parameters T and η

cally within and across days. Further, on adding intra-day attention, we note improvements in profit as `FAST` can better distinguish noise inducing text from relevant market signals, minimizing false evaluations and overreactions (De Long et al., 1990). The attention mechanism can likely diminish the impact of such noise (rumours, vague comments). Intuitively, complementing the intra-day with the inter-day attention lead to further improvements as `FAST` can better capture the *diverse influence* of texts, hierarchically within and across days.

Next, we note the biggest improvements on adding the t-LSTM instead of a standard LSTM as the intra-day encoder, suggesting that `FAST` benefits by factoring the fine-grain time irregularities in texts to model the flow of stock-affecting information (Kalev et al., 2004). Through this time-aware mechanism, `FAST` can potentially better react to online news and tweets, by discounting stale information more accurately by factoring in fine-grained elapsed time differences (seconds). We further quantify the impact of time-aware modeling on the improvements in ranking and profitability, next.

5.3 Advantages of Time-Aware Modeling

The influence of older information over the market decreases rapidly as newer data is released (Russell, 2010). As we coarsen the granularity of the elapsed time difference between two texts fed to the t-LSTM, from minutes, to hours, to a day, we observe drops in `FAST`'s performance as shown in Figure 4. At the coarsest granularity of a day, the t-LSTM essentially degenerates to an LSTM, and attains the lowest performance. The drops show that factoring time at finer granularities benefits `FAST` in modeling the temporal dynamics of market response to stock affecting signals. Our findings line with financial research that shows market reactions to news complete within minutes (Smales, 2013), and the impact of news and tweets achieves an equilibrium over time (Shen et al., 2018).

5.4 Parameter Analysis: Probing Sensitivity

Lookback window length T: Here, we study how `FAST`'s stock ranking performance (NDCG@5) varies with the length of lookback $T \in [2, 10]$ days in Figure 5. Lower ranking performance indicates the inability of shorter lookbacks to capture stock affecting market information, likely as public information requires time to absorb into price movements (Luss and D'Aspremont, 2015). As we increase T, we find that larger lookbacks allow the inclusion of stale information from older days having relatively lower influence on prices (Bernhaedt and Miao, 2004), thus deteriorating the ranking performance. We observe the best stock ranking performance for mid-sized (approx. 5-day) lookback periods.

Selected top stocks η: We analyze `FAST`'s profitability (SR) variation with the number of top stocks η in Figure 5. We find that `FAST` performs well across varying η, showing suitability to strategies having different risk taking appetites.

6 `FAST` Qualitative Analysis

We now conduct an extended analysis as shown in Figure 6 to elucidate on `FAST`'s explainable predictions and practical applicability to real-world quantitative trading. Here, we study the China & Hong Kong market during 5^{th} - 9^{th} December 2015. We visualize token-level and hierarchical temporal attention to analyze how `FAST` ranks stocks on 10^{th} December 2015, outperforming the state-of-the-art baseline methods: RankNet and SN-HFA.

Analyzing Hierarchical Attention: Within days, the intra-day attention filters less informative news and emphasizes more influential ones. For instance, we observe that the second news about BOE Technology on 7^{th} categorizes the stock as "overweight," a rating through which equity

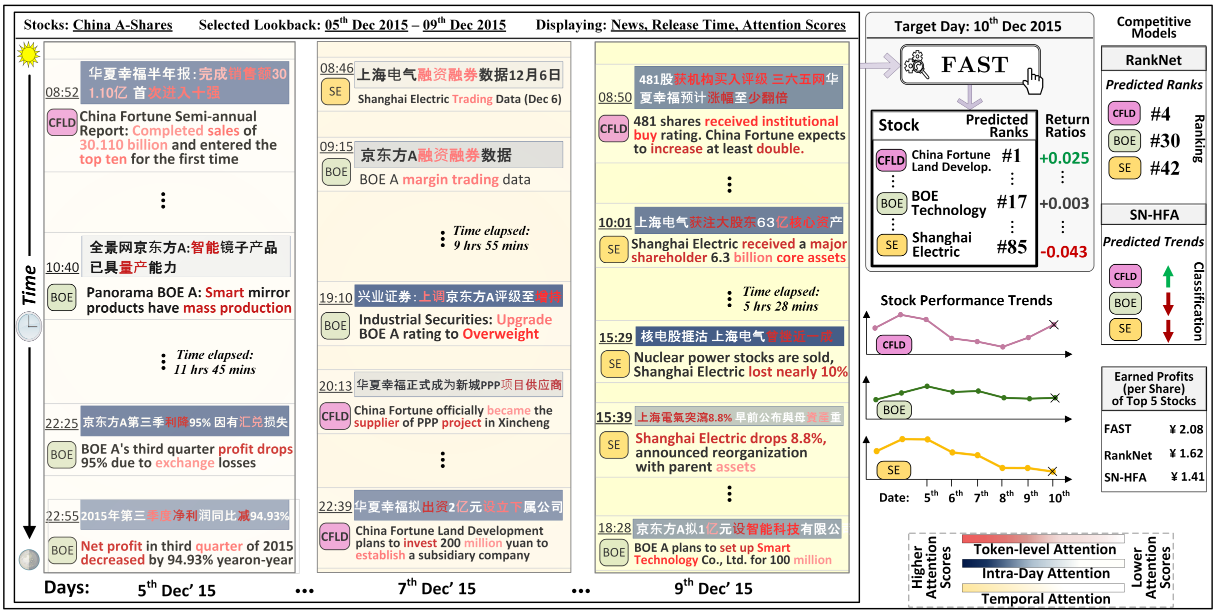


Figure 6: Financial news in Chinese (shown with corresponding English translations) during 5^{th}-9^{th} December 2015 for stocks in the China & Hong Kong dataset (test split), with visualization of FAST's attention mechanisms (left); Stock trading performance for 10^{th} December 2015 across FAST and competitive baseline methods (right).

analysts forecast better future performance (Kumar, 2009). Such news would likely induce positive public sentiments and drive more investment to the stock, as opposed to the less informative news about "margin trading data." The intra-day attention accurately captures the *diverse influence* of such news headlines. Further, we observe that news released on 9^{th} comprises relatively more crucial information than other days, and the inter-day attention accurately emphasizes its importance. These observations reiterate the diverse influence of different news and days over future stock returns, accurately captured by the hierarchical temporal attention mechanism.

Probing Time-aware Modeling: The news released for Shanghai Electric (SE) on the morning of 9^{th} reports a positive event, likely indicating future profits. Later during that afternoon, after a few hours, two other news report negative impacts over SE due to a loss in nuclear power stocks, indicating a downtrend for the upcoming days. `FAST` disregards older news to emphasize on the newer ones to forecast the upcoming loss and allots a lower rank to SE. In contrast, ranking methods such as RankNet assign a higher rank to SE, potentially due to their inability to model the time-aware dependencies. Further, classification methods like SN-HFA do not correctly predict the stock return trends, as they do not capture the fine-grain temporal irregularities in texts, and are not optimized towards profit. Consequently, `FAST` relatively outperforms SN-HFA by a margin of 47.5%, and RankNet by 28.4% in profits on 10^{th} December 2015.

7 Conclusion

We propose `FAST`, a neural approach to rank profitable stocks using stock-relevant textual data across online financial news and tweets. To model the market information, `FAST` hierarchically learns temporally relevant signals from texts and shows the positive effects of factoring the fine-grain temporal irregularities in textual data. Through quantitative and qualitative analyses on English tweets and Chinese financial news spanning four stock markets, we highlight the real-world applicability of `FAST`. In trading simulations on the S&P 500 and China A-shares indexes, `FAST` outperforms state-of-the-art methods across four different formulations by over 8% in terms of profit and Sharpe Ratio.

References

Klaus Adam, Albert Marcet, and Juan Pablo Nicoli. 2016. Stock market volatility and learning. *The Journal of Finance*, 71(1):33–82.

Md Shad Akhtar, Abhishek Kumar, Deepanway Ghosal, Asif Ekbal, and Pushpak Bhattacharyya. 2017. A multilayer perceptron based ensemble technique for fine-grained financial sentiment analysis.

In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 540–546, Copenhagen, Denmark. Association for Computational Linguistics.

Leif BG Andersen. 2007. Efficient simulation of the heston stochastic volatility model. *Available at SSRN 946405*.

Wei Bao, Jun Yue, and Yulei Rao. 2017. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLOS ONE*.

Brad M. Barber and Terrance Odean. 2007. All That Glitters: The Effect of Attention and News on the Buying Behavior of Individual and Institutional Investors. *The Review of Financial Studies*, 21(2).

Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 65–74, New York, NY, USA. Association for Computing Machinery.

Dan Bernhaedt and Jianjun Miao. 2004. Informed trading when information becomes stale. *The Journal of Finance*, 59(1):339–390.

Hakan Berument and Halil Kiymaz. 2001. The day of the week effect on stock market volatility. *Journal of economics and finance*, 25(2):181–193.

Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327.

Laurent E. Calvet and Adlai J. Fisher. 2007. Multifrequency news and stock returns. *Journal of Financial Economics*, 86(1):178 – 212.

Deli Chen, Shuming Ma, Keiko Harimoto, Ruihan Bao, Qi Su, and Xu Sun. 2019a. Group, extract and aggregate: Summarizing a large amount of finance news for forex movement prediction. In *Proceedings of the Second Workshop on Economics and Natural Language Processing*, pages 41–50, Hong Kong. Association for Computational Linguistics.

Deli Chen, Yanyan Zou, Keiko Harimoto, Ruihan Bao, Xuancheng Ren, and Xu Sun. 2019b. Incorporating fine-grained events in stock movement prediction. In *Proceedings of the Second Workshop on Economics and Natural Language Processing*, pages 31–40, Hong Kong. Association for Computational Linguistics.

J Bradford De Long, Andrei Shleifer, Lawrence H Summers, and Robert J Waldmann. 1990. Noise trader risk in financial markets. *Journal of political Economy*, 98(4):703–738.

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

Xin Du and Kumiko Tanaka-Ishii. 2020. Stock embeddings acquired from news articles and price history, and an application to portfolio optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3353–3363, Online. Association for Computational Linguistics.

Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. 2019a. Enhancing stock movement prediction with adversarial training.

Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019b. Temporal relational ranking for stock prediction. *ACM Trans. Inf. Syst.*, 37(2).

William Forbes. 2009. *Behavioural finance*. John Wiley & Sons.

Thierry Foucault, Johan Hombert, and Ioanid Roşu. 2016. News trading and speed. *The Journal of Finance*, 71(1):335–382.

Junyi Gao, Cao Xiao, Yasha Wang, Wen Tang, Lucas M. Glass, and Jimeng Sun. 2020. Stagenet: Stage-aware neural networks for health risk prediction. In *Proceedings of The Web Conference 2020*, WWW '20, page 530–540, New York, NY, USA. Association for Computing Machinery.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Ehsan Hoseinzade, Saman Haratizadeh, and Arash Khoeini. 2019. U-cnnpred: A universal cnn-based predictor for stock markets.

Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2017. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the 11 international conference on web search and data mining*.

Jieyun Huang, Yunjia Zhang, Jialai Zhang, and Xi Zhang. 2018. A tensor-based sub-mode coordinate algorithm for stock prediction.

Bruce I Jacobs and Kenneth N Levy. 1988. Calendar anomalies: Abnormal returns at calendar turning points. *Financial Analysts Journal*, 44(6):28–39.

Weiwei Jiang. 2020. Applications of deep learning in stock market prediction: recent progress.

Petko S Kalev, Wai-Man Liu, Peter K Pham, and Elvis Jarnecic. 2004. Public information arrival and volatility of intraday stock returns. *Journal of Banking & Finance*, 28(6):1441–1467.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Shimon Kogan, Dimitry Levin, Bryan R. Routledge, Jacob S. Sagi, and Noah A. Smith. 2009. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 272–280, Boulder, Colorado. Association for Computational Linguistics.

Kazuhiro Kohara, Tsutomu Ishikawa, Yoshimi Fukuhara, and Yukihiro Nakamura. 1997. Stock price prediction using prior knowledge and neural networks. *Intelligent Systems in Accounting, Finance & Management*, 6(1):11–22.

Alok Kumar. 2009. Who gambles in the stock market? *The Journal of Finance*, 64(4):1889–1933.

Helinä Laakkonen. 2004. The impact of macroeconomic news on exchange rate volatility. *Bank of Finland discussion paper*, (24).

Heeyoung Lee, Mihai Surdeanu, Bill MacCartney, and Dan Jurafsky. 2014. On the importance of text analysis for stock price prediction. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 1170–1175, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Quanzhi Li and Sameena Shah. 2017. Learning stock market sentiment lexicon and sentiment-oriented word vector from StockTwits. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 301–310, Vancouver, Canada. Association for Computational Linguistics.

Zhiyuan Li and Sanjeev Arora. 2019. An exponential learning rate schedule for deep learning.

Xiaowei Lin, Zehong Yang, and Yixu Song. 2009. Short-term stock price prediction based on echo state networks. *Expert systems with applications*, 36(3):7313–7317.

Dehong Liu, Hongmei Gu, and Tiancai Xing. 2016. The meltdown of the chinese equity market in the summer of 2015. *International Review of Economics & Finance*, 45:504 – 517.

Yang Liu, Qi Liu, Hongke Zhao, Zhen Pan, and Chuanren Liu. 2020. Adaptive quantitative trading: An imitative deep reinforcement learning approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(02):2128–2135.

Andrew W. Lo. 2004. The adaptive markets hypothesis. *The Journal of Portfolio Management*, 30(5):15–29.

Chi-Jie Lu, Tian-Shyug Lee, and Chih-Chou Chiu. 2009. Financial time series forecasting using independent component analysis and support vector regression. *Decision support systems*, 47(2):115–125.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation.

Ronny Luss and Alexandre D'Aspremont. 2015. Predicting abnormal returns from news using text classification. *Quantitative Finance*, 15(6):999–1012.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models.

Burton G Malkiel. 1989. Efficient market hypothesis. In *Finance*, pages 127–134. Springer.

Elham Mohammadi, Hessam Amini, and Leila Kosseim. 2019. CLaC lab at SemEval-2019 task 3: Contextual emotion detection using a combination of neural networks and SVM. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 153–158, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. Topic modeling based sentiment analysis on social media for stock market prediction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1354–1364, Beijing, China. Association for Computational Linguistics.

George Norman. 2014. *Semi-Strong Form Efficient Market Hypothesis*. Edward Elgar Publishing.

Nuno Oliveira, Paulo Cortez, and Nelson Areal. 2017. The impact of microblogging data for stock market prediction: Using twitter to predict returns, volatility, trading volume and survey sentiment indices. *Expert Systems with Applications*, 73:125–144.

Maureen O'Hara. 2015. High frequency market microstructure. *Journal of Financial Economics*, 116(2):257 – 270.

Yangtuo Peng and Hui Jiang. 2016. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 374–379, San Diego, California. Association for Computational Linguistics.

Yu Qin and Yi Yang. 2019. What you say and how you say it matters: Predicting stock volatility using verbal and vocal cues. In *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics, pages 390–401, Florence, Italy. Association for Computational Linguistics.

Navid Rekabsaz, Mihai Lupu, Artem Baklanov, Alexander Dür, Linda Andersson, and Allan Hanbury. 2017. Volatility prediction using financial disclosures sentiments with word embedding-based IR models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1712–1721, Vancouver, Canada. Association for Computational Linguistics.

Calum Robertson, Shlomo Geva, and Rodney Wolff. 2007. Predicting the short-term market reaction to asset specific news: Is time against us? In *Emerging Technologies in Knowledge Discovery and Data Mining*, pages 15–26, Berlin, Heidelberg. Springer.

Jeffrey R Russell. 2010. Market microstructure effects. *Encyclopedia of Quantitative Finance*.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Ratn Shah. 2020a. Deep attentive learning for stock movement prediction from social media text and company correlations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8415–8426, Online. Association for Computational Linguistics.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Ratn Shah. 2020b. A time-aware transformer based model for suicide ideation detection on social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7685–7697, Online. Association for Computational Linguistics.

Martin Scholtus, Dick van Dijk, and Bart Frijns. 2014. Speed, algorithmic trading, and market quality around macroeconomic news announcements. *Journal of Banking and Finance*, 38:89 – 105.

Robert P. Schumaker and Hsinchun Chen. 2009. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Trans. Inf. Syst.*, 27(2).

Benjamin Segal and Dan Segal. 2016. Are managers strategic in reporting non-earnings news? evidence on timing and news bundling. *Review of Accounting Studies*, 21(4):1203–1244.

William F Sharpe. 1994. The sharpe ratio. *Journal of portfolio management*, 21(1):49–58.

Dehua Shen, Xiao Li, and Wei Zhang. 2018. Baidu news information flow and return volatility: Evidence for the sequential information arrival hypothesis. *Economic Modelling*, 69:127 – 133.

Yauheniya Shynkevich, T.M. McGinnity, Sonya A. Coleman, Ammar Belatreche, and Yuhua Li. 2017. Forecasting price movements using technical indicators: Investigating the impact of varying input window length. *Neurocomputing*, 264:71 – 88. Machine learning in finance.

Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based twitter sentiment for stock prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 24–29, Sofia, Bulgaria. Association for Computational Linguistics.

Lee A. Smales. 2013. Impact of macroeconomic announcements on interest rate futures: High-frequency evidence from australia. *Journal of Financial Research*, 36(3):371–388.

Qiang Song, Anqi Liu, and Steve Y. Yang. 2017. Stock portfolio selection using learning-to-rank algorithms with news sentiment. *Neurocomputing*, 264:20 – 28. Machine learning in finance.

Narges Tabari, Piyusha Biswas, Bhanu Praneeth, Armin Seyeditabari, Mirsad Hadzikadic, and Wlodek Zadrozny. 2018. Causality analysis of Twitter sentiments and stock market returns. In *Proceedings of the First Workshop on Economics and Natural Language Processing*, pages 11–19, Melbourne, Australia. Association for Computational Linguistics.

Ali Tafti, Ryan Zotti, and Wolfgang Jank. 2016. Real-time diffusion of information on twitter and the financial markets. *PloS one*, 11(8):e0159226.

Huizhe Wu, Wei Zhang, Weiwei Shen, and Jun Wang. 2018. Hybrid deep sequential modeling for social text-driven stock prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1627–1630, New York, NY, USA. Association for Computing Machinery.

Yumo Xu and Shay B. Cohen. 2018. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1979, Melbourne, Australia. Association for Computational Linguistics.

Steve Y. Yang, Yangyang Yu, and Saud Almahdi. 2018. An investor sentiment reward-based trading system using gaussian inverse reinforcement learning algorithm. *Expert Systems with Applications*, 114:388 – 401.

Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. 2017. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 2141–2149, New York, NY, USA. Association for Computing Machinery.

Building Representative Corpora from Illiterate Communities: A Review of Challenges and Mitigation Strategies for Developing Countries

Stephanie Hirmer[1], **Alycia Leonard**[1], **Josephine Tumwesige**[2], **Costanza Conforti**[2,3]
[1]Energy and Power Group, University of Oxford
[2]Rural Senses Ltd.
[3]Language Technology Lab, University of Cambridge
stephanie.hirmer@eng.ox.ac.uk

Abstract

Most well-established data collection methods currently adopted in NLP depend on the assumption of speaker literacy. Consequently, the collected corpora largely fail to represent swathes of the global population, which tend to be some of the most vulnerable and marginalised people in society, and often live in rural developing areas. Such underrepresented groups are thus not only ignored when making modeling and system design decisions, but also prevented from benefiting from development outcomes achieved through data-driven NLP. This paper aims to address the under-representation of illiterate communities in NLP corpora: we identify potential biases and ethical issues that might arise when collecting data from rural communities with high illiteracy rates in Low-Income Countries, and propose a set of practical mitigation strategies to help future work.

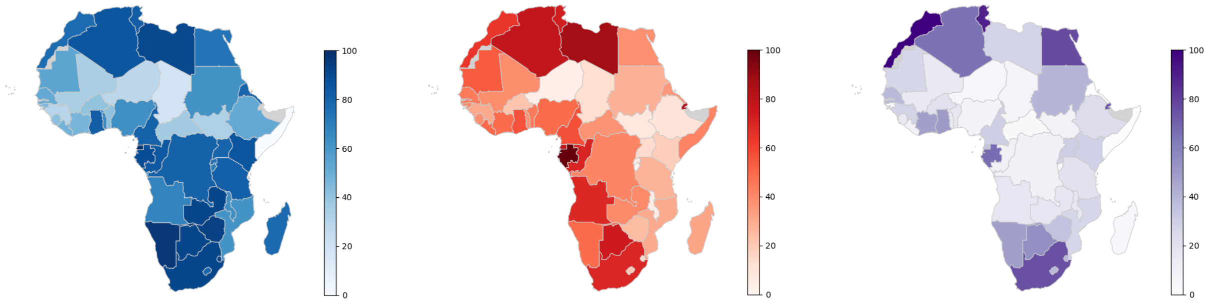
1 Introduction

The exponentially increasing popularity of supervised Machine Learning (ML) in the past decade has made the availability of data crucial to the development of the Natural Language Processing (NLP) field. As a result, much NLP research has focused on developing rigorous processes for collecting large corpora suitable for training ML systems. We observe, however, that many best practices for quality data collection make two implicit assumptions: that speakers have internet access and that they are literate (i.e. able to read and often write text effortlessly[1]). Such assumptions might be reasonable in the context of most High-Income Countries (HICs) (UNESCO, 2018). However, in Low-Income Countries (LICs), and especially in sub-Saharan Africa (SSA), such assumptions may not hold, particularly in rural developing areas where the bulk of the population lives (Roser and Ortiz-Ospina (2016), Figure 1). As a consequence, common data collection techniques – designed for use in HICs – fail to capture data from a vast portion of the population when applied to LICs. Such techniques include, for example, crowdsourcing (Packham, 2016), scraping social media (Le et al., 2016) or other websites (Roy et al., 2020), collecting articles from local newspapers (Marivate et al., 2020), or interviewing experts from international organizations (Friedman et al., 2017). While these techniques are important to easily build large corpora, they implicitly rely on the above-mentioned assumptions (i.e. internet access and literacy), and might result in demographic misrepresentation (Hovy and Spruit, 2016). In this paper, we make a first step towards addressing *how to build representative corpora in LICs from illiterate speakers*. We believe that this is a currently unaddressed topic within NLP research. It aligns with previous work investigating sources of bias resulting from the under-representation of specific demographic groups in NLP corpora (such as women (Hovy, 2015), youth (Hovy and Søgaard, 2015), or ethnic minorities (Groenwold et al., 2020)). In this paper, we make the following contributions: (i) we introduce the challenges of collecting data from illiterate speakers in §2; (ii) we define various possible sources of biases and ethical issues which can contribute to low data quality we define various possible sources of biases and ethical issues which can contribute to low data quality we define various possible sources of biases and ethical issues which can contribute to low data quality §3; finally, (iii) drawing on years of experience in data collection in LICs, we outline practical countermeasures to address these issues in §4.

---

[1] For example, input from speakers is often taken in writing, in response to a written stimulus which must be read.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2176–2189
April 19 - 23, 2021.



(a) Adult literacy (% ages 15+, UNESCO (2018))

(b) Urban population (% total, UN-DESA (2018))

(c) Internet usage (% of total, ITU (2019))

Figure 1: Literacy, urban population, and internet usage in African countries. Note that countries with more rural populations tend to have less literacy and less internet users. These countries are likely to be under-represented in corpora generated using common data collection methods that assume literacy and internet access (Grey: no data).

2 Listening to the Illiterate: What Makes it Challenging?

In recent years, developing corpora that encompasses as many human languages as possible has been recognised as important in the NLP community. In this context, widely translated texts (such as the Bible (Mueller et al., 2020) or the Human Rights declaration (King, 2015)) are often used as a source of data. However, these texts tend to be quite short and domain-specific. Moreover, while the Internet constitutes a powerful data collection tool which is more representative of real language use than the previously-mentioned texts, it excludes illiterate communities, as well as speakers which lack reliable internet access (as is often the case in rural developing settings, Figure 1).

Given the obstacles to using these common language data collection methods in LIC contexts, the NLP community can learn from methodologies adopted in other fields. Researchers from fields such as sustainable development (SD, Gleitsmann et al. (2007)), African studies (Adams, 2014), and ethnology (Skinner et al., 2013), tend to rely heavily on qualitative data from oral interviews, transcribed verbatim. Collecting such data in rural developing areas is considerably more difficult than in developed or urban contexts. In addition to high illiteracy levels, researchers face challenges such as seasonal roads and low population densities. To our knowledge, there are very few NLP works which explicitly focus on building corpora from rural and illiterate communities: of those works that exist, some present clear priming effect issues (Abraham et al., 2020), while others focus on application (Conforti et al., 2020). A detailed description of best practices for data collection remains a notable research gap.

3 Definitions and Challenges

Guided by research in medicine (Pannucci and Wilkins, 2010), sociology (Berk, 1983), and psychology (Gilovich et al., 2002), NLP has experienced increasing interest in ethics and bias mitigation to minimise unintentional demographic misrepresentation and harm (Hovy and Spruit, 2016). While there are many stages where bias may enter the NLP pipeline (Shah et al., 2019), we focus on those pertinent to *data collection* from rural illiterate communities in LICs, leaving the study of biases in model development for future work[2].

3.1 Data Collection Biases

Biases in data collection are inevitable (Marshall, 1996) but can be minimised when known to the researcher (Trembley, 1957). We identify various biases that can emerge when collecting language data in rural developing contexts, which fall under three broad categories: sampling, observer, and response bias. Sampling determines who is studied, the interviewer (or observer) determines what information is sought and how it is interpreted, and the interviewee (or respondent) determines which information is revealed (Woodhouse, 1998). These categories span the entire data collection process and can affect the quality and quantity of language data obtained.

---

[2] Note, this paper does not focus on a particular NLP application, as once the data has been collected from illiterate communities it can be annotated for virtually any specific task.

3.2 Sampling or selection bias

Sampling bias occurs when observations are drawn from an unrepresentative subset of the population being studied (Marshall, 1996) and applied more widely. In our context, this might arise when selecting communities from which to collect language data, or specific individuals within each community. When sampling communities, bias can be introduced if convenience is prioritized. Communities which are easier to access may not produce language data representative of a larger area or group. This can be illustrated through Uganda's refugee response, which consists of 13 settlements (including the 2nd largest in the world) hosted in 12 districts (UNHCR, 2020). Data collection may be easier in one of the older, established settlements; however, such data cannot be generalised over the entire refugee response due to different cultural backgrounds, length of stay of refugees in different areas, and the varied stages along the humanitarian chain – emergency, recovery or development – found therein (Winter, 1983; OECD, 2019). Prioritizing convenience in this case may result in corpora which over-represents the cultural and economic contexts of more established, longer-term refugees. When sampling interviewees, bias can be introduced when certain sub-sets of a community have more data collected than others (Bryman, 2012). This is seen when data is collected only from men in a community due to cultural norms (Nadal, 2017), or only from wealthier people in cell-phone-based surveys (Labrique et al., 2017).

3.2.1 Observer bias

Observer bias occurs when there are systematic errors in how data is recorded, which may stem from observer viewpoints and predispositions (Gonsamo and D'Odorico, 2014). We identify three key observer biases relevant to our context.

Firstly, **confirmation bias**, which refers to the tendency to look for information which confirms one's preconceptions or hypotheses (Nickerson, 1998). Researchers collecting data in LICs may expect interviewees to express needs or hardships based on their preconceptions. As Kumar (1987) points out, "often they hear what they want to hear and ignore what they do not want to hear". A team conducting a needs assessment for a rural electrification project, for instance, may expect a need for electricity, and thus consciously or subconsciously seek data which confirms this, interpret potentially unrelated data as electricity-motivated (Hirmer and Guthrie, 2017), or omit data which contradicts their hypothesis (Peters, 2020). Using such data to train NLP models may introduce unintentional bias towards the original expectations of the researchers instead of accurately representing the community.

Secondly, the interviewer's understanding and interpretation of the speaker's utterances might be influenced by their class, culture and language. Note that, particularly in countries without strong language standardisation policies, consistent semantic shifts can happen even between varieties spoken in neighboring regions (Gordon, 2019), which may result in systematic **misunderstanding** (Sayer, 2013). For example, in the neighboring Ugandan tribes of Toro and Bunyoro, the same word *omunyoro* means respectively *husband* and *a member of the tribe*. Language data collected in such contexts, if not properly handled, may contain inaccuracies which lead to NLP models that misrepresent these tribes. Rich information communicated through gesture, expression, and tone (i.e. nonverbal data, Oliver et al. (2005)) may also be systematically lost during verbatim transcription, causing inadvertent inconsistencies in the corpora.

Thirdly, **interviewer bias**, which refers to the subjectivity unconsciously introduced into data gathering by the worldview of the interviewer (Frey, 2018). For instance, a deeply religious interviewer may unintentionally frame questions through religious language (e.g. *it is God's will*, *thank God*, etc.), or may perceive certain emotions (e.g. thankfulness) as inherently religious, and record language data including this perception. The researcher's attitude and behaviour may also influence responses (Silverman, 2013); for instance, when interviewers take longer to deliver questions, interviewees tend to provide longer responses (Matarazzo et al., 1963). Unlike in internet-based language data collection, where all speakers are exposed to uniform, text-based interfaces, collecting data from illiterate communities necessitates the presence of an interviewer, who cannot always be the same person due to scalability constraints, introducing this inevitable variability and subsequent data bias.

3.2.2 Response bias

Response bias occurs when speakers provide inaccurate or false responses to questions. This is particularly important when working in rural settings, where the majority of data collection is currently related to SD projects. The majority of existing data is biased by the projects for which it has been collected, and any newly collected data for NLP uses is also likely to be used in decision making for SD. This inherent link of data collection to material development outcomes inevitably affects what is communicated. There are five key response biases relevant to our context.

Firstly, **recall bias**, where speakers recall only certain events or omit details (Coughlin, 1990). This is often as a result of external influences, such as the presence of a data collector who is new to the community. Recall can also be affected by the distortion or amplification of traumatic memories (Strange and Takarangi, 2015); if data is collected around a topic a speaker may find traumatic, recall bias may be unintentionally introduced.

Secondly, **social desirability bias**, which refers to the tendency of interviewees to provide socially desirable/acceptable responses rather than honest responses, particularly in certain interview contexts (Bergen and Labonté, 2020). In tight-knit rural communities, it may be difficult to deviate from traditional social norms, leading to biased data. As an illustrative example, researchers in Nepal found that interviewer gender affected the detail in responses to some sensitive questions (e.g. sex and contraception): participants provided less detail to male interviewers (Axinn, 1991). Social desirability bias can produce corpora which misrepresent community social dynamics or under-represent sensitive topics.

Thirdly, **recency effect or serial-position**, which is the tendency of a person to recall the first and last items in a series best, and the middle items worst (Troyer, 2011). This can greatly impact the content of language data. For instance, in the context of data collection to guide development work, it is important to understand current needs and values (Hirmer and Guthrie, 2016); however, if only the most recent needs are discussed, long-term needs may be overlooked. To illustrate, while a community which has just experienced a poor agricultural season may tend to express the importance of improving agricultural output, other needs which are less top-of-mind (i.e. healthcare, education) may be equally important despite being expressed less frequently. If data containing *recency bias* is used to develop NLP models, particularly for sustainable development applications (such as for Automatic UPV Classification, Conforti et al. (2020)), these may amplify current needs and under-represent long-term needs.

Fourthly, **acquiescence bias**, also known as “yea” saying (Laajaj and Macours, 2017), which can occur in rural developing contexts when interviewees perceive that certain (possibly false) responses will please a data collector and bring benefits to their community. For example, if data collection is being undertaken by a group with a stated desire to build a school may be more likely to hear about how much education is valued.

Finally, **priming effect**, or the ability of a presented stimulus to influence one’s response to a subsequent stimulus (Lavrakas, 2008). Priming is problematic in data collection to inform SD projects; it can be difficult to collect data on the relative importance of simultaneous (or conflicting) needs if the community is primed to focus on one (Veltkamp et al., 2011). An example is shown in Figure 2a; respondents may be drawn to speak more about the most dominant prompts presented in the chart. This is typical of a broader failure in SD to uncover beneficiary priorities without introducing project bias (Watkins et al., 2012). Needs assessments, like the one referenced above linked to a rural electrification project, tend to focus explicitly on project-related needs instead of more broadly identifying what may be most important to communities (Masangwi, 2015; USAID, 2006). As speakers will usually know why data is being collected in such cases, they may be biased towards stating the project aim as a need, thereby skewing the corpora to over-represent this aim.

3.3 Ethical Considerations

Certain ethical codes of conduct must be followed when collecting data from illiterate speakers in rural communities in LICs (Musoke et al., 2020). Unethical data collection may harm communities, treat them without dignity, disrupt their lives, damage intra-community or external relationships, and disregard community norms (Thorley and Henrion, 2019). This is particularly critical in rural developing regions, as these areas are home to some of the world’s poorest and most vulnerable to exploitation (Christiaensen and Subbarao, 2005; de Ceni-val M., 2008). Unethical data collection can replicate extractive colonial relationships whereby data is extracted from communities with no mutual benefit or ownership (Dunbar and Scrimgeour, 2006). It can lead to a lack of trust between data collector and interviewees and unwillingness to participate in future research (Clark, 2008). These phenomena can bias data or reduce data availability. Ethical data collection practices in rural developing regions with high illiteracy include: obtaining consent (McAdam, 2004), accounting for cultural differences (Silverman, 2013), ensuring anonymity and confidentiality (Bryman, 2012), respecting existing community or leadership structures (Harding et al., 2012), and making the community the owner of the data. While the latter is not often currently practiced, it is an important consideration for community empowerment, with indigenous data sovereignty efforts (Rainie et al., 2019) already setting precedent.

4 Countermeasures

Drawing on existing literature and years of field experience collecting spoken data in LICs, below we outline a number of practical data collection strategies to minimise previously-outlined challenges (§3), enabling the collection of high-quality, minimally-biased data from illiterate speakers in LICs suitable for use in NLP models. While these measures have primarily been applied in SSA, we have also successfully tested them in projects focusing on refugees in the Middle East and rural communities in South Asia.

4.1 Preparation

Here, we outline practical preparation steps for careful planning, which can minimise error and reduce fieldwork duration (Tukey, 1980).

Local Context. A thorough understanding of local context is key to successful data collection (Hentschel, 1999; Bukenya et al., 2012; Launiala and Kulmala, 2006). Local context is broadly defined as facts, concepts, beliefs, values, and perceptions used by local people to interpret the world around them, and is shaped by their surroundings (i.e. their worldview, Vasconcellos and Vasconcellos Sobrinho (2014)). It is important to consider local context when preparing to collect data in rural developing areas, as common data collection methods may be inappropriate due to contextual linguistic differences and deep-rooted social and cultural norms (Walker and Hamilton, 2011; Mafuta et al., 2016; Nikulina et al., 2019; Wang et al.). Selecting a contextually-appropriate data collection method is critical in mitigating *social desirability bias* in the collected data, among other challenges. Researchers should review socio-economic surveys and/or consult local stakeholders who can offer valuable insights on practices and social norms. These stakeholders can also highlight current or historical matters of concern to the area, which may be unfamiliar to researchers, and reveal local, traditional, and indigenous knowledge which may impact the data being collected (Wu, 2014) and result in *recency effect*. It is good practice to identify local conflicts and segmentation within a community, especially in a rural context, where the population is vulnerable and systematically unheard (Dudwick et al., 2006; Mallick et al., 2011).

Case sampling. In qualitative research, sample cases are often strategically selected based on the research question (i.e. *systematic* or *purposive* sampling, Bryman (2012)), and characteristics or circumstances relevant to the topic of study (Yach, 1992). If data collected in such research is used beyond its original scope, *sampling bias* may result. So, while data collected in previous research should be re-used to expand NLP corpora where possible, it is important to be cognizant of the purposive sampling underlying existing data. A comprehensive dataset characterisation (Bender and Friedman, 2018; Gebru et al., 2018) can help researchers understand whether an existing dataset is appropriate to use in new or different research, such as in training new NLP models, and can highlight the potential ethical concerns of data re-use.

Participant sampling. Interviewees should be selected to represent the diverse interests of a community or sampling group (e.g. occupation, age, gender, religion, ethnicity or male/female household heads (Bryman, 2012)) to reduce *sampling bias* (Kitzinger, 1994). To ensure representativity in collected data, sampling should be random, i.e. every subject has equal probability to be included (Etikan et al., 2016). There may be certain societal subsets that are concealed from view (e.g. as a result of embarrassment from disabilities or physical differences) based on cultural norms in less inclusive societies (Vesper, 2019); particular care should be exercised to ensure such subsets are represented.

Group composition. Participant sampling best practices vary by data collection method, with particular care being necessary in group settings. In traditional societies where strong power dynamics exist, attention should be paid to group composition and interaction to prevent some voices from

	Bias & Definition	Key countermeasures
Sampling	**Community**: An unrepresentative sample set is generalised over the entire case being studied.	• Select representative communities & only apply data within same scope (i.e. consult data statements)
	Participant: Certain sub-sets of a community have more data collected from them than others.	• Select representative participants, only apply data within same scope & avoid tempting rewards
Observer	**Confirmation**: Looking for information that confirms one's preconceptions or hypotheses about a topic/research/sector.	• Employ interviewers that are impartial to the topic/research/sector investigated.
	Misunderstanding: Data is incorrectly transcribed or categorized as a result of class, cultural, or linguistic differences.	• Employ local people & minimise # of people involved for both data collection & transcription.
	Interviewer: Unconscious subjectivity introduced into data gathering by interviewers' worldview.	• Undertake training to minimise influence exerted from questions, technology, & attitudes.
Response	**Recall**: Tendency of speakers to recall only certain events or omit details	• Collect support data (e.g. from socio-economic data or local stakeholders) to compare with interviews.
	Social-desirability: Tendency of participants to provide socially desirable/acceptable responses rather than to respond honestly.	• Select interviewers & design interview processes to account for known norms which might skew responses
	Recency effect: Tendency to recall first or last items in a series best, & middle items worst.	• Minimise external influence on participants throughout data gathering (e.g. technologies, people, perceptions).
	Acquiescence: Respondents perceive certain, perhaps false, answers may please data collectors, bringing community benefits.	• Gather non-sectoral holistic insights (e.g. from socio-economic data or local stakeholders)
	Priming effect: Ability of a presented stimulus to influence one's response to a subsequent stimulus	• Use appropriate visual prompts (graphically similar), language and technology

Table 1: Sources of potential bias in data collection when operating in rural and illiterate settings in developing countries, and key countermeasures that can help mitigating them.

being silenced or over-represented (Stewart et al., 2007). For example, in Uganda, female interviewees may be less likely to voice opinions in the presence of male interviewees (FIDH, 2012; Axinn, 1991), introducing a form of *social desirability bias* in resulting corpora. To minimise this risk of data bias, relations and power dynamics must be considered during data collection planning (Hirmer, 2018). It may be necessary to exclude, for instance, close relatives, governmental officials, and village leaders from group discussions where data is being collected, and instead engage such stakeholders in separate activities to ensure that their voices are included in the corpora without biasing the data collected from others.

Interviewer selection. The interviewer has a significant opportunity to introduce *observer and response biases* in collected data (Salazar, 1990). Interviewers familiar with local language, including community-specific dialects, should be selected wherever possible. Moreover, to reduce *misunderstanding* and *recall biases* in collected data, it is useful to have the same person who conducts the interviews also transcribe them. This minimizes the layers of linguistic interpretation affecting the final dataset and can increase accuracy through familiarity with the interview content. If the interviewer is unavailable, the transcriber must be properly trained and briefed on the interviews, and made aware of the level of detail needed during transcription (Parcell and Rafferty, 2017).

Study design. In rural LIC communities, qualitative data like natural language is usually collected by observation, interview, and/or focus group discussion (or a combination, known as mixed methods) which are transcribed verbatim (Moser and Korstjens, 2018). Prompts are often used to spark discussion. Whether visual prompts (Hirmer, 2018) or verbalised question prompts are used during data collection, these should be designed to: (i) accommodate illiteracy, (ii) account for disabilities (e.g. visually impairment; both could cause *sampling bias*), and (iii) minimise bias towards a topic or sector (e.g. minimising *acquisition bias* and *confirmation bias*). For instance, visual prompts should be graphically similar and contain only visuals familiar to the respondents. This is analogous to the uniform interface with which speakers interact during text-based online data collection, where the platform used is graphically the same to all users inputting data. Using varied graphical styles or unfamiliar images may result in *priming* (Figure 2a). To minimise *recall bias* or *recency effect* in collected data, socio-economic data can be integrated in data analysis to better understand if the assertions made in collected data reference recent events, for example. These should be non-sector specific, to gain holistic insights and to minimise *acquisition bias* and *confirmation bias*.

4.2 Engagement

Here, we outline practical steps for successful community engagement to achieve ethical and high-quality data collection.

Defining community. Defining a community in an open and participatory manner is critical to meaningful engagement (Dyer et al., 2014). By understanding the community the way they understand themselves, misunderstandings and tensions that affect data quality can be minimized. The definition of the community (MacQueen et al., 2001) coupled with the requirements and use-cases for the collected data determines the data collection methodology and style which will be most appropriate (e.g. interview-based community consultation vs. collaborative co-design for mutual learning).

Follow formal structures. Researchers entering a community where they have no background to collect data should endeavour to know the community prior to commencing any work (Diallo et al., 2005). This could entail visiting the community and mapping its hierarchies of authority and decision-making pathways, which can guide the research team on how to interact respectfully with the community (Tindana et al., 2011). This process should also illuminate whether knowledgeable community members should facilitate entry by performing introductions and assisting the external data collection team. Following formal community structures is vital, especially in developing communities, where traditional rules and social conventions are strongly held yet often not articulated explicitly or documented. Approaching community leaders in the traditional way can help to build a positive long-term relationship, removing suspicion about the nature and motivation of the researchers' activities, explaining their presence in the community, and most importantly building trust as they are granted permission to engage the community by its leadership (Tindana et al., 2007).

Verbalising consent. Data ethics is paramount for research involving human participants (Accenture, 2016; Tindana et al., 2007), including any collection of personal and identifiable data, such as natural language. Genuine (i.e. voluntary and informed) consent must be obtained from interviewees to prevent use of data which is illegal, coercive, or for a purpose other than that which has been agreed (McAdam, 2004). The Nuffield Council on Bioethics (2002) caution that in LICs, misunderstandings may occur due to cultural differences, lower social-economic status, and illiteracy (McMillan et al., 2004) which can call into question the legitimacy of consent obtained. Researchers must understand that methods such as long information forms and consent forms which must be signed may be inappropriate for the cultural context of LICs and can be more likely to confuse than to inform (Tekola et al., 2009). The authors advise that consent forms should be verbal instead of written, with wording familiar to the interviewees and appropriate to their level of comprehension (Tekola et al., 2009). For example, to speak of data storage on a password protected computer while obtaining consent in a rural community without access to electricity or information technology is unfitting. Innovative ways to record consent can be employed in such contexts (e.g. video taping or recording), as signing an official document may be "viewed with suspicion or even outright hostility" (Upjohn and Wells, 2016), or seen as "committing ... to something other than answering questions". Researchers new to qualitative data collection should seek advice from experienced researchers and approval from their ethics committee before implementing consent processes.

Approaching participants. Despite having gained permission from community authorities and obtained consent to collect data, researchers must be cautious when approaching participants (Irabor and Omonzejele, 2009; Diallo et al., 2005) to ensure they do not violate cultural norms. For example, in some cultures a senior family member must be present for another household member to be interviewed, or a female must be accompanied by a male counterpart during data collection. Insensitivity to such norms may compromise the data collection process; so, they should be carefully noted when researching local context (§4.1) and interviews should be designed to accommodate them where possible. Furthermore, researchers should investigate the motivations of the participants to identify when inducements become inappropriate and may lead to either harm or data bias (McAdam, 2004).

Minimise external influence. Researchers must be aware of how external influences can affect data collection (Ramakrishnan et al., 2012). We find three main levels of external influence: (i) technologies unfamiliar to a rural developing country context may induce *social desirability bias* or *priming* (e.g. if a researcher arrives to a community in an expensive vehicle or uses a tablet for data collection); (ii) intergroup context, which according to Abrams (2010) refers to when "people in different social groups view members of other groups" and may feel prejudiced or threatened by these differences. This can occur, for instance, when a newcomer arrives and speaks loudly relative to the indigenous community, which may be perceived as overpowering; (iii) there is the risk of a researcher over-incentivizing the data collection process, using leading questions and judgemental framing (*interviewer bias* or *confirmation bias*). To overcome these influences, researchers must be cognizant of their influence and minimise it by hiring local mediators where possible alongside employing appropriate technology, mannerisms, and language.

4.3 Undertaking Interviews

Here, we detail practical steps to minimise challenges during the actual data collection.

Interview settings. People have personal values and drivers that may change in specific settings. For example, in the Ugandan Buganda and Busoga tribes, it is culturally appropriate for the male head if present to speak on behalf of his wife and children. This could lead to corpora where input from the husband is over-represented compared to the rest of the family. To account for this, it is important to collect data in multiple interview settings (e.g. individual, group male/female/mixed; Figures 2b, 2c). Additionally, the inputs of individuals in group settings should be considered independently to ensure all participants have an equal say, regardless of their position within the group (Barry et al., 2008; Gallagher et al., 1993). This helps to avoid *social desirability bias* in the data and is particularly important in various developing contexts where stereotypical gender roles are prominent (Hirmer, 2018). During interviews, verbal information can be supplemented through the observation of tone, cadence, gestures, and facial expressions (Narayanasamy, 2009; Hess et al., 2009), which could enrich the collected data with an additional layer of annotation.

Working with multiple interviewers. Arguably, one of the biggest challenges in data collection is ensuring consistency when working with multiple interviewers. Some may report word-for-word what is being said, while others may summarise or misreport, resulting in systematic *misunderstanding*. Despite these risks, employing multiple interviewers is often unavoidable when collecting data in rural areas of developing countries, where languages often exhibit a high number of regional, non-mutually intelligible varieties. This is particularly prominent across SSA. For example, 41 languages are spoken in Uganda (Nakayiza, 2016); English, the official language, is fluently spoken by only ∼5% of the population, despite being widely used among researchers and NGOs (Katushemererwe and Nerbonne, 2015). To minimise data inconsistency, researchers should: (i) undertake interviewer training workshops to communicate data requirements and practice data collection processes through mock field interviews; (ii) pilot the data collection process and seek feedback to spot early deviation from data requirements; (iii) regularly spot-check interview notes; (iv) support written notes with audio recordings[3]; and (v) offer quality based incentives to data collectors.

Participant remuneration. While it is common to offer interviewees some form of remuneration for their time, the decision surrounding payment is ethically-charged and widely contested (Hammett and Sporton, 2012). Rewards may tempt people to participate in data collection against their judgement. They can introduce *sampling bias* or create power dynamics resulting in *acquiescence bias* (Largent and Lynch, 2017). Barbour (2013) offers three practical solutions: (i) not advertise payment; (ii) omit the amount being offered; or (iii) offer non-financial incentives (e.g. products that are desirable but difficult to get in an area). The decision whether or not to remunerate should not be based upon the researcher's own ethical beliefs and resources, but instead by considering the specific context[4], interviewee expectations, precedents set by previous researchers, and local norms (Hammett and Sporton, 2012). Representatives from local organisations (such as NGOs or governmental authorities) may be able to offer advice.

---

[2] While participants' photographing permission was granted, photos were pixelised to protect identity.

[3] Relying only on audio data recording may be risky: equipment can fail or run out of battery (which is not easily remedied in rural off-grid regions) and seasonal factors (as noise from rain on corrugated iron sheets, commonly used for roofing in SSA) can make recordings inaudible (Hirmer, 2018)).

[4] In rural Uganda, for example, politicians commonly engage in *vote buying* by distributing gifts (Blattman et al., 2019) such as soap or alcohol. It is therefore considered an unruly form of remuneration and can only be avoided when known.

(a) (b) (c)

Figure 2: Collecting oral data in rural Uganda. 2a *Priming effect* (note the word "Energy" in the poster's title and the visual prompts differences between items). On the contrary, 2b and 2c show minimal priming; note also that different demographics are separately interviewed (women group, single men) to avoid *social desirability bias*.

4.4 Post-interviewing

Here, we discuss practical strategies to mitigate ethical issues surrounding the management and stewardship of collected data.

Anonymisation. To protect the participants' identity and data privacy, locations, proper names, and culturally explicit aspects (such as tribe names) of collected data should be made anonymous (Sweeney, 2000; Kirilova and Karcher, 2017). This is particularly important in countries with security issues and low levels of democracy.

Safeguarding data. A primary responsibility of the researcher is to safeguard participants' data (Kirilova and Karcher, 2017). In addition to anonymizing data, mechanisms for data management include in-place handling and storage of data (UKRI, 2020a). Whatever data management plan is adopted, it must be clearly articulated to participants before the start of the interview (i.e. as part of the consent process (Silverman, 2013)), as was discussed in §4.2 (*Verbalising consent*).

Withdrawing consent. Participants should have the ability to withdraw from research within a specified time frame. This is known as *withdraw consent* and is commonly done by phone or email (UKRI, 2020b). As people in rural illiterate communities have limited means and technology access, a local phone number and contact details of a responsible person in the area should be provided to facilitate withdraw consent.

Communication and research fatigue. While researchers frequently extract knowledge and data from communities, only rarely are findings fed back to communities in a way that can be useful to them. Whatever the research outcomes, researchers should share the results with participating communities in an appropriate manner. In illiterate communities, for instance, murals (Jimenez, 2020), artwork, speeches, or song could be used to communicate findings. Not communicating findings may result in research fatigue as people in *over-studied* communities are no longer willing to participate in data collection. This is common "where repeated engagements do not lead to any experience of change [...]" Clark (2008). Patel et al. (2020) offers practical guidance to minimise research fatigue by: (i) increasing transparency of research purpose at the beginning of the research, and (ii) engaging with gatekeeper or oversight bodies to minimise number of engagements per participant. Failure to restrict the number of times that people are asked to participate in studies risks poor future participation (Patel et al., 2020) which can also lead to *sampling bias*.

5 Conclusion

In this paper, we provided a first step towards defining best practices in data collection in rural and illiterate communities in Low-Income Countries to create globally representative corpora. We proposed a comprehensive classification of sources of bias and unethical practices that might arise in the data collection process, and discussed practical steps to minimise their negative effects. We hope that this work will motivate NLP practitioners to include input from rural illiterate communities in their research, and facilitate smooth and respectful interaction with communities during data collection. Importantly, despite the challenges that working in such contexts might bring, the effort to build substantial and high-quality corpora which represent this subset of the population can result in considerable SD outcomes.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback. We are also grateful to Claire McAlpine, as well as Malcolm McCulloch and other members of the Energy and Power Group (University of Oxford) for providing valuable feedback on early versions of this paper. This research was carried out as part of the Oxford Martin Programme on Integrating Renewable Energy. Finally, we are grateful to the Rural Senses team for sharing experiences on data collection.

References

Basil Abraham, Danish Goel, Divya Siddarth, Kalika Bali, Manu Chopra, Monojit Choudhury, Pratik Joshi, Preethi Jyoti, Sunayana Sitaram, and Vivek Seshadri. 2020. Crowdsourcing speech data for low-resource languages from low-income workers. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2819–2826, Marseille, France. European Language Resources Association.

Dominic Abrams. 2010. *Processes of prejudices: Theory, evidence and intervention*. Equalities and Human Rights Commission.

Accenture. 2016. Building digital trust: The role of data ethics in the digital age. Accenture Labs.

Glenn Adams. 2014. Decolonizing methods: African studies and qualitative research. *Journal of social and personal relationships*, 31(4):467–474.

William G Axinn. 1991. The influence of interviewer sex on responses to sensitive questions in nepal. *Social Science Research*, 20(3):303–318.

Rosaline Barbour. 2013. *Introducing qualitative research: a student's guide*. Sage.

Marie-Louise Barry, Herman Steyn, and Alan Brent. 2008. Determining the most important factors for sustainable energy technology selection in africa: Application of the focus group technique. In *PICMET'08-2008 Portland International Conference on Management of Engineering & Technology*, pages 181–187. IEEE.

Emily M. Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Nicole Bergen and Ronald Labonté. 2020. "everything is perfect, and we have no problems": Detecting and limiting social desirability bias in qualitative research. *Qualitative Health Research*, 30(5):783–792.

Richard A Berk. 1983. An introduction to sample selection bias in sociological data. *American sociological review*, pages 386–398.

The Nuffield Council on Bioethics. 2002. The ethics of research related to healthcare in developing countries. The Nuffield Council on Bioethics is funded jointly by the Medical Research Council, the Nuffield Foundation and the Wellcome Trust.

Christopher Blattman, Horacio Larreguy, Benjamin Marx, and Otis R Reid. 2019. Eat widely, vote wisely? lessons from a campaign against vote buying in uganda. Technical report, National Bureau of Economic Research.

Alan Bryman. 2012. Mixed methods research: combining quantitative and qualitative research. In Alan Bryman, editor, *Social Reserach Methods*, forth edition, chapter 27, pages 628–652. Oxford University Press, New York.

Badru Bukenya, Sam Hickey, and Sophie King. 2012. Understanding the role of context in shaping social accountability interventions: towards an evidence-based approach. *Manchester: Institute for Development Policy and Management, University of Manchester.*

de Cenival M. 2008. Ethics of research: the freedom to withdraw. *Bulletin de la Societe de pathologie exotique*, 101(2):98—-101.

Luc J. Christiaensen and Kalanidhi Subbarao. 2005. Towards an understanding of household vulnerability in rural kenya. *Journal of African Economies*, 14(4):520–558.

Tom Clark. 2008. We're over-researched here!' exploring accounts of research fatigue within qualitative research engagements. *Sociology*, 42(5):953–970.

Costanza Conforti, Stephanie Hirmer, Dai Morgan, Marco Basaldella, and Yau Ben Or. 2020. Natural language processing for achieving sustainable development: the case of neural labelling to enhance community profiling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8427–8444, Online. Association for Computational Linguistics.

Steven S Coughlin. 1990. Recall bias in epidemiologic studies. *Journal of clinical epidemiology*, 43(1):87–91.

D. A. Diallo, O. K. Doumbo, C. V. Plowe, T. E. Wellems, E. J. Emanuel, and S. A Hurst. 2005. Community permission for medical research in developing countries. *Infectious Diseases Society of America*, 41(2):255—-259.

Nora Dudwick, Kathleen Kuehnast, Veronica N. Jones, and Michael Woolcock. 2006. Analyzing social capital in context: A guide to using qualitative methods and data.

Terry Dunbar and Margaret Scrimgeour. 2006. Ethics in indigenous research–connecting with community. *Journal of Bioethical Inquiry*, 3(3):179–185.

J Dyer, L.C Stringer, A.J Dougill, J Leventon, M Nshimbi, F Chama, A Kafwifwi, J.I Muledi, J.-M.K Kaumbu, M Falcao, S Muhorro, F Munyemba, G.M Kalaba, and S Syampungani. 2014. Assessing participatory practices in community-based natural resource management: Experiences in community engagement from southern africa. *Journal of Environmental Management*, 137:137–145.

Ilker Etikan, Sulaiman Abubakar Musa, and Rukayya Sunusi Alkassim. 2016. Comparison of convenience sampling and purposive sampling. *American journal of theoretical and applied statistics*, 5(1):1–4.

FIDH. 2012. Women's rights in Uganda: gaps between policy and practice. Technical report, International Federation for Human Rights, Paris.

Bruce B Frey. 2018. *The SAGE encyclopedia of educational research, measurement, and evaluation*. Sage Publications.

Batya Friedman, Lisa P. Nathan, and Daisy Yoo. 2017. Multi-lifespan information system design in support of transitional justice: Evolving situated design principles for the long (er) term. *Interacting with Computers*, 29(1):80–96.

Morris Gallagher, Tim Hares, John Spencer, Colin Bradshaw, and Ian Webb. 1993. The nominal group technique: a research tool for general practice? *Family practice*, 10(1):76–81.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2018. Datasheets for datasets. *arXiv preprint arXiv:1803.09010*.

Thomas Gilovich, Dale Griffin, and Daniel Kahneman. 2002. *Heuristics and biases: The psychology of intuitive judgment*. Cambridge university press.

Brett A Gleitsmann, Margaret M Kroma, and Tammo Steenhuis. 2007. Analysis of a rural water supply project in three communities in mali: Participation and sustainability. In *Natural resources forum*, volume 31, pages 142–150. Wiley Online Library.

Alemu Gonsamo and Petra D'Odorico. 2014. Citizen science: best practices to remove observer bias in trend analysis. *International journal of biometeorology*, 58(10):2159–2163.

Matthew J Gordon. 2019. Language variation and change in rural communities. *Annual Review of Linguistics*, 5:435–453.

Sophie Groenwold, Lily Ou, Aesha Parekh, Samhita Honnavalli, Sharon Levy, Diba Mirza, and William Yang Wang. 2020. Investigating African-American Vernacular English in transformer-based text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5877–5883, Online. Association for Computational Linguistics.

Daniel Hammett and Deborah Sporton. 2012. Paying for interviews? negotiating ethics, power and expectation. *Area*, 44(4):496–502.

Anna Harding, Barbara Harper, Dave Stone, Catherine O'Neill, Patricia Berger, Stuart Harris, and Jamie Donatuto. 2012. Conducting research with tribal communities: Sovereignty, ethics, and data-sharing issues. *Environmental health perspectives*, 120(1):6–10.

J. Hentschel. 1999. Contextuality and data collection methods: A framework and application to health service utilisation. *Journal of Development Studies*, 35(4):64–94.

Ursula Hess, RB Jr Adams, and Robert E Kleck. 2009. Intergroup misunderstandings in emotion communication. *Intergroup misunderstandings: Impact of divergent social realities*, pages 85–100.

Stephanie Hirmer. 2018. *Improving the Sustainability of Rural Electrification Schemes: Capturing Value for Rural Communities in Uganda*. Ph.D. thesis, University of Cambridge.

Stephanie Hirmer and Peter Guthrie. 2016. Identifying the needs of communities in rural uganda: A method for determining the 'user-perceived value' of rural electrification initiatives. *Renewable and Sustainable Energy Reviews*, 66:476–486.

Stephanie Hirmer and Peter Guthrie. 2017. The benefits of energy appliances in the off-grid energy sector based on seven off-grid initiatives in rural uganda. *Renewable and Sustainable Energy Reviews*, 79:924–934.

Dirk Hovy. 2015. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 752–762, Beijing, China. Association for Computational Linguistics.

Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 483–488, Beijing, China. Association for Computational Linguistics.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

D. O. Irabor and P. Omonzejele. 2009. Local attitudes, moral obligation, customary obedience and other cultural practices: their influence on the process of gaining informed consent for surgery in a tertiary institution in a developing country. *Developing world bioethics*, 9(1):34—-42.

ITU. 2019. International Telecommunication Union World Telecommunication/ICT Indicators Database. World Bank Open Data.

Stephany Jimenez. 2020. Creatively Communicating through Visual and Verbal Art- Poetry and Murals. Yale National Initiative.

Fridah Katushemererwe and John Nerbonne. 2015. Computer-assisted language learning (call) in support of (re)-learning native languages: the case of runyakitara. *Computer Assisted Language Learning*, 28(2):112–129.

Benjamin Philip King. 2015. *Practical Natural Language Processing for Low-Resource Languages*. Ph.D. thesis, University of Michigan.

Dessi Kirilova and Sebastian Karcher. 2017. Rethinking data sharing and human participant protection in social science research: Applications from the qualitative realm. *Data Science Journal*, 16.

Jenny Kitzinger. 1994. The methodology of Focus Groups: the importance of interaction between research participants. *Sociology of Health and Illness*, 16(1):103–121.

Krishna Kumar. 1987. *Conducting group interviews in developing countries*. US Agency for International Development Washington, DC.

Rachid Laajaj and Karen Macours. 2017. *Measuring skills in developing countries*. The World Bank.

Alain Labrique, Emily Blynn, Saifuddin Ahmed, Dustin Gibson, George Pariyo, and Adnan A Hyder. 2017. Health surveys using mobile phones in developing countries: automated active strata monitoring and other statistical considerations for improving precision and reducing biases. *Journal of medical Internet research*, 19(5):e121.

Emily A Largent and Holly Fernandez Lynch. 2017. Paying research participants: regulatory uncertainty, conceptual confusion, and a path forward. *Yale journal of health policy, law, and ethics*, 17(1):61.

A. Launiala and T. Kulmala. 2006. The importance of understanding the local context: Women's perceptions and knowledge concerning malaria in pregnancy in rural malawi. *Acta Tropica*, 98(2):111–117.

Paul J Lavrakas. 2008. *Encyclopedia of survey research methods*. Sage Publications.

Tuan Anh Le, David Moeljadi, Yasuhide Miura, and Tomoko Ohkuma. 2016. Sentiment analysis for low resource languages: A study on informal indonesian tweets. In *Proceedings of the 12th Workshop on Asian Language Resources (ALR12)*, pages 123–131.

KM MacQueen, E McLellan, DS Metzger, S Kegeles, RP Strauss, and et al. 2001. What is community? an evidence-based definition for participatory public health. *American Journal of Public Health*, 91:1929–1938.

Eric M. Mafuta, Lisanne Hogema, Thérèse N.M. Mambu, Pontien B Kiyimbi, Berthys P. Indebe, Patrick K. Kayembe, Tjard De Cock Buning, and Marjolein A. Dieleman. 2016. Understanding the local context and its possible influences on shaping, implementing and running social accountability initiatives for maternal health services in rural democratic republic of the congo: a contextual factor analysis. *Science Advances*, 16(1):1–13.

B. Mallick, K. Rubayet Rahaman, and J. Vogt. 2011. Social vulnerability analysis for sustainable disaster mitigation planning in coastal bangladesh. *Disaster Prevention and Management*, 20(3):220–237.

Vukosi Marivate, Tshephisho Sefara, Vongani Chabalala, Keamogetswe Makhaya, Tumisho Mokgonyane, Rethabile Mokoena, and Abiodun Modupe. 2020. Investigating an approach for low resource language dataset creation, curation and classification: Setswana and sepedi. *arXiv preprint arXiv:2003.04986*.

M N Marshall. 1996. Sampling for qualitative research. *Family practice*, 13(6):522–5.

Salule J. Masangwi. 2015. Methodology for Solar PV Needs Assessment in Chikwawa, Southern Malawi. Technical report, Malawi Renewable Energy Acceleration Programme (MREAP) MREAP: Renewable Energy Capacity Building Programme (RECBP) Produced.

Joseph D Matarazzo, Morris Weitman, George Saslow, and Arthur N Wiens. 1963. Interviewer influence on durations of interviewee speech. *Journal of Verbal Learning and Verbal Behavior*, 1(6):451–458.

Keith McAdam. 2004. The ethics of research related to healthcare in developing countries. *Acta Bioethica*, 10(1):49–55.

J. R. McMillan, C. Conlon, and Nuffield Council on Bioethics. 2004. The ethics of research related to healthcare in developing countries. *Journal of medical ethics*, 30:204–206.

Albine Moser and Irene Korstjens. 2018. Series: Practical guidance to qualitative research. part 3: Sampling, data collection and analysis. *European Journal of General Practice*, 24(1):9–18.

Aaron Mueller, Garrett Nicolai, Arya D McCarthy, Dylan Lewis, Winston Wu, and David Yarowsky. 2020. An analysis of massively multilingual neural machine translation for low-resource languages. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3710–3718.

David Musoke, Charles Ssemugabo, Rawlance Ndejjo, Sassy Molyneux, and Elizabeth Ekirapa-Kiracho. 2020. Ethical practice in my work: community health workers' perspectives using photovoice in wakiso district, uganda. *BMC Medical Ethics*, 21(1):1–10.

Kevin L. Nadal. 2017. Sampling Bias and Gender. In *The SAGE Encyclopedia of Psychology and Gender*.

Judith Nakayiza. 2016. The sociolinguistic situation of english in uganda. *Ugandan English. Amsterdam/Philadelphia: John Benjamins*, pages 75–94.

N. Narayanasamy. 2009. *Participatory Rural Appraisal: Principles, Methods and Application*, first edition. SAGE Publications India Pvt Ltd, New Dehli.

Raymond S Nickerson. 1998. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175–220.

V. Nikulina, H. Larson Lindal, J.and Baumann, D. Simon, and H Ny. 2019. Lost in translation: A framework for analysing complexity of co-production settings in relation to epistemic communities, linguistic diversities and culture. *Futures*, 113(102442):1–13.

OECD. 2019. Survey of refugees and humanitarian staff in Uganda. Joint effort by Ground Truth Solutions (GTS) and the Organisation for Economic Co-operation and Development (OECD) Secretariat with financial support from the United Kingdom's Department for International Development (DFID).

Daniel G Oliver, Julianne M Serovich, and Tina L Mason. 2005. Constraints and opportunities with interview transcription: Towards reflection in qualitative research. *Social forces*, 84(2):1273–1289.

Sean Packham. 2016. *Crowdsourcing a text corpus for a low resource language*. Ph.D. thesis, University of Cape Town.

Christopher J Pannucci and Edwin G Wilkins. 2010. Identifying and avoiding bias in research. *Plastic and reconstructive surgery*, 126(2):619.

Erin S. Parcell and Katherine A. Rafferty. 2017. Interviews, recording and transcribing. In Mike Allen, editor, *The SAGE Encyclopedia of Communication Research Methods*, pages 800–803. SAGE Publications, Thousand Oaks.

Sonny S Patel, Rebecca K Webster, Neil Greenberg, Dale Weston, and Samantha K Brooks. 2020. Research fatigue in covid-19 pandemic and post-disaster research: causes, consequences and recommendations. *Disaster Prevention and Management: An International Journal*.

Uwe Peters. 2020. What is the function of confirmation bias? *Erkenntnis*, pages 1–26.

Stephanie Carroll Rainie, Tahu Kukutai, Maggie Walter, Oscar Luis Figueroa-Rodríguez, Jennifer Walker, and Per Axelsson. 2019. Indigenous data sovereignty.

Thiagarajan Ramakrishnan, Mary C. Jones, and Anna Sidorova. 2012. Factors influencing business intelligence (bi) data collection strategies: An empirical investigation. *Decision Support Systems*, 52:486––496.

Max Roser and Esteban Ortiz-Ospina. 2016. Literacy. *Our World in Data*.

Dwaipayan Roy, Sumit Bhatia, and Prateek Jain. 2020. A topic-aligned multilingual corpus of Wikipedia articles for studying information asymmetry in low resource languages. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2373–2380, Marseille, France. European Language Resources Association.

Mary Kathryn Salazar. 1990. Interviewer bias: How it affects survey research. *Aaohn Journal*, 38(12):567–572.

Inaad Mutlib Sayer. 2013. Misunderstanding and language comprehension. *Procedia-Social and Behavioral Sciences*, 70:738–748.

Deven Shah, H. Andrew Schwartz, and Dirk Hovy. 2019. Predictive biases in natural language processing models: A conceptual framework and overview.

David Silverman. 2013. *Doing Qualitative Reserach*, fourth edition. SAGE Publications Inc.

Jonathan Skinner et al. 2013. *The interview: An ethnographic approach*, volume 49. A&C Black.

David Stewart, Prem Shamdasani, and Dennis Rook. 2007. Group dynamics and focus group research. In David Stewart, Prem Shamdasani, and Dennis Rook, editors, *Focus Groups: Theory and Practice*, chapter 2, pages 19–36. SAGE Publications, Thousand Oaks.

Deryn Strange and Melanie KT Takarangi. 2015. Memory distortion for traumatic events: the role of mental imagery. *Frontiers in psychiatry*, 6:27.

Latanya Sweeney. 2000. Simple demographics often identify people uniquely. *Health (San Francisco)*, 671(2000):1–34.

Fasil Tekola, Susan J. Bull, Farsides Bobbie, J. Newport Melanie, Adeyemo Adebowale, N. Rotimi Charles, and Davey Gail. 2009. Tailoring consent to context: Designing an appropriate consent process for a biomedical study in a low income setting. *PLOS Neglected Tropical Diseases*, 3:7.

Lisa Thorley and Emma Henrion. 2019. DFID ethical guidance for research, evaluation and monitoring activities. Prepared for the UK Department for International Development.

Paulina O. Tindana, Linda Rozmovits, Renaud F. Boulanger, Sunita V.S. Bandewar, Raymond A. Aborigo, Abraham V.O. Hodgson, Pamela Kolopack, and James V. Lavery. 2011. Aligning community engagement with traditional authority structures in global health research: A case study from northern ghana. *American Journal of Public Health*, 101:1857–1867.

Paulina O. Tindana, Jerome A. Singh, C. Shawn Tracy, Ross E.G. Upshur, Abdallah S. Daar, Peter A. Singer, Janet Frohlich, and James V. Lavery. 2007. Grand challenges in global health: Community engagement in research in developing countries. *PLOS Medicine*, 4(9).

Marc-Adèlard Trembley. 1957. The Key Informant Technique: A Nonethnographic Application. Technical report, Cornell University, New York.

Angela K. Troyer. 2011. *Serial Position Effect*, pages 2263–2264. Springer New York, New York, NY.

J. W. Tukey. 1980. We need both exploratory and confirmatory. *American Statistician*, 34(1):23–25.

UKRI. 2020a. Data protection Guidance. Economic and Social Research Council.

UKRI. 2020b. Do participants have a right to withdraw consent? Economic and Social Research Council.

UNDESA. 2018. United Nations Department of Economic and Social Affairs, Population Division: World Urbanization Prospects. World Bank Open Data.

UNESCO. 2018. UNESCO Institute for Statistics Adult Literacy Rate. World Bank Open Data.

UNHCR. 2020. Refugees and Nationals by District. Uganda Comprehensive Refugee Response Portal.

Melissa Upjohn and Kimberly Wells. 2016. Challenges associated with informed consent in low-and low-middle-income countries. *Frontiers in veterinary science*, 3:92.

USAID. 2006. Powering Health, Electrification Options for Rural Health Centers. Technical report, USAID, Washington DC, USA.

Ana Maria de Albuquerque Vasconcellos and Mário Vasconcellos Sobrinho. 2014. Knowledge and culture: two significant issues for local level development programme analysis. *Interações (Campo Grande)*, 15(2):285–300.

Martijn Veltkamp, Ruud Custers, and Henk Aarts. 2011. Motivating consumer behavior by subliminal conditioning in the absence of basic needs: Striking even while the iron is cold. *Journal of Consumer Psychology*, 21(1):49–56.

Inga Vesper. 2019. Facts & Figures: Disabilities in developing countries. Sci Dev Net.

Robert S Walker and Marcus J Hamilton. 2011. Social complexity and linguistic diversity in the austronesian and bantu population expansions. *Proceedings of the Royal Society B: Biological Sciences*, 278(1710):1399–1404.

Ke Wang, Steven Goldstein, Madeleine Bleasdale, Bernard Clist, Koen Bostoen, Paul Bakwa-Lufu, Laura T Buck, Alison Crowther, Alioune Dème, Roderick J McIntosh, et al. Ancient genomes reveal complex patterns of population movement, interaction, and replacement in sub-saharan africa.

Christopher D Watkins, Lisa M DeBruine, Anthony C Little, David R Feinberg, and Benedict C Jones. 2012. Priming concerns about pathogen threat versus resource scarcity: dissociable effects on women's perceptions of men's attractiveness and dominance. *Behavioral Ecology and Sociobiology*, 66(12):1549–1556.

Roger P Winter. 1983. Uganda-creating a refugee crisis. *Cultural Survival Quarterly*, 7(2).

Philip Woodhouse. 1998. Thinking with People and Organizations. In Alan Thomas, Joanna Chataway, and Marc Wuyts, editors, *Finding out Fast: Investigative Skills for Policy and Development*, first edition, chapter Part III, pages 127–146. SAGE Publications Inc, Milton Keynes.

B. Wu. 2014. Embedding research in local context: local knowledge, stakeholders' participation and fieldwork design. *Field Research Method Lab at LSE*.

Derek Yach. 1992. The use and value of qualitative methods in health research in developing countries. *Social science & medicine*, 35(4):603–612.

Process-Level Representation of Scientific Protocols with Interactive Annotation

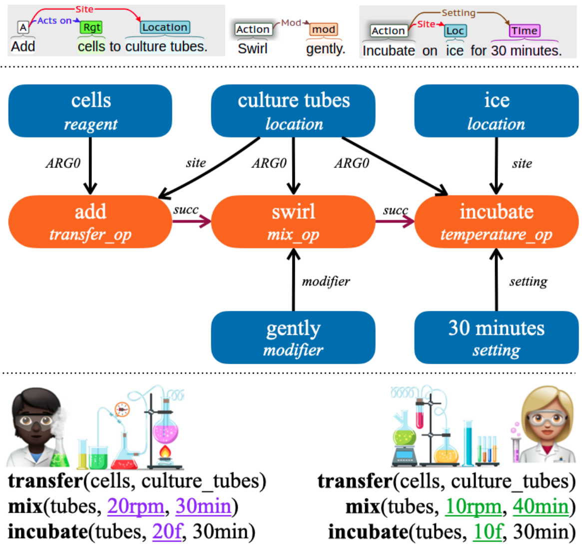
Ronen Tamari[†*] **Fan Bai**[‡] **Alan Ritter**[‡] **Gabriel Stanovsky**[†⋆]
[†]The Hebrew University of Jerusalem
[‡]Georgia Institute of Technology
[⋆]Allen Institute for Artificial Intelligence
`{ronent,gabis}@cs.huji.ac.il`
`{fan.bai,alan.ritter}@cc.gatech.edu`

Abstract

We develop Process Execution Graphs (PEG), a document-level representation of real-world wet lab biochemistry protocols, addressing challenges such as cross-sentence relations, long-range coreference, grounding, and implicit arguments. We manually annotate PEGs in a corpus of complex lab protocols with a novel interactive textual simulator that keeps track of entity traits and semantic constraints during annotation. We use this data to develop graph-prediction models, finding them to be good at entity identification and local relation extraction, while our corpus facilitates further exploration of challenging long-range relations.[1]

1 Introduction

There is a drive in recent years towards automating wet lab environments, where menial benchwork procedures such as pipetting, centrifuging, or incubation are software-controlled, and either executed by fully automatic lab equipment (Lee and Miles, 2018), or with a human-in-the-loop (Keller et al., 2019). These environments allow reliable and precise experiment reproducbility while relieving researchers from tedious and laborious work which is prone to human error (Bates et al., 2017; Prabhu and Urban, 2017). To achieve this, several programmatic formalisms are developed to describe an experiment as an executable program. For example, Autoprotocol (Lee and Miles, 2018) defines a `mix` predicate taking three arguments: *mode*, *speed*, and *duration*.



transfer(cells, culture_tubes)
mix(tubes, 20rpm, 30min)
incubate(tubes, 20f, 30min)

transfer(cells, culture_tubes)
mix(tubes, 10rpm, 40min)
incubate(tubes, 10f, 30min)

Figure 1: We develop a scaffold (center) between sentence-level lab procedure representations (top) and low-level, lab-specific instructions (bottom). The Process Execution Graph (PEG) captures document-level relations between procedures (orange rounded nodes) and their arguments (blue rectangular nodes).

A promising direction to leverage automatic wet-lab environments is a conversion from natural language protocols, written in expressive free-form language, to low-level instructions, ensuring a non-ambiguous, repeatable description of experiments.

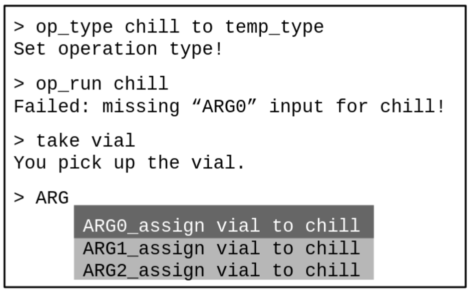
In this work, we focus on a crucial first step towards such conversion – the extraction and representation of the relations conveyed by the protocol in a formal graph structure, termed Process Execution Graphs (PEG), exemplified in Figure 1. PEGs capture both concrete, exact quantities ("*30 minutes*"), as well as vague instructions ("swirl *gently*"). A researcher can then port the PEG (either manually or automatically) to their specific lab equipment, e.g., specifying what constitutes a gentle swirl setting and adding missing arguments, such as the temperature of the

---

[*] Work begun on an internship at the Allen Institute for Artificial Intelligence.

[1] Our annotated corpus, simulator, annotation interface, interaction data, and models are available for use by the research community at `https://textlabs.github.io/`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2190–2202
April 19 - 23, 2021.



```
> op_type chill to temp_type
Set operation type!

> op_run chill
Failed: missing "ARG0" input for chill!

> take vial
You pick up the vial.

> ARG
       ARG0_assign vial to chill
       ARG1_assign vial to chill
       ARG2_assign vial to chill
```

Figure 2: Example interaction with our simulator, showing predicate grounding ("chill" is a `temp_type` operation) input assignment ("vial" is an argument of "chill"), validation (warning for a missing argument) and auto-complete driven by state-tracking, where only legal instructions in a given state are presented.

incubation in Figure 1.

Formally, PEGs are directed, acyclic labeled graphs, capturing how objects in the lab (e.g., *cells*, *tubes*) are manipulated by lab operations (e.g., `mixing`, `incubating`), and in what order. Importantly, PEGs capture relations which may span across multiple sentences and implicit arguments. For example, the PEG in Figure 1 explicitly captures the relation between *culture tubes*, mentioned in the first sentence, and `swirl` and `incubate` which appear in later sentences.

To annotate long and complex lab protocols, we develop a text-based game annotation interface simulating objects and actions in a lab environment (see example in Figure 2). Our annotators are given wet-lab protocols written in natural language taken from biochemistry publications, and are asked to repeat their steps by issuing textual commands to the simulator. The commands are deterministically converted to our PEG representation. This interface takes much of the burden off annotators by keeping track of object traits and commonsense constraints. For example, when the annotator issues a `transfer` command for a container, the simulator moves all its contents as well. We find that in-house annotators were able to effectively use this interface on complex protocols, achieving good agreement.

Finally, we use this data to explore several models, building upon recent advances in graph prediction algorithms (Luan et al., 2019; Wadden et al., 2019). We thoroughly analyze model performance and find that our data introduces interesting new challenges, such as complex coreference resolution and long-range, cross-sentence relation identification.

In conclusion, we make the following contributions:

- We formalize a PEG representation for free-form, natural language lab protocols, providing a semantic scaffold between free-form scientific literature and low-level instruments instruction.
- We develop a novel annotation interface for procedural text annotation using text-based games, and show that it is intuitive enough for wet-lab protocol annotation by non-experts.
- We release X-WLP, a challenging corpus of 279 PEGs representing document-level lab protocols. This size is on par with similar corpora of procedural text (Dalvi et al., 2018; Mysore et al., 2019; Vaucher et al., 2020).
- We develop two graph parsers: a pipeline model which chains predictions for graph sub-components, and a joint-model of mention and relation detectors.

2 Background and Motivation

Several formalisms for programmatic lab controller interfaces were developed in recent years (Yachie and Natsume, 2017; Lee and Miles, 2018). For instance, Autoprotocol defines 35 lab commands, including `spin`, `incubate`, and `mix`.[2] While these define wet-lab experiments in a precise and unambiguous manner, they do not readily replace their natural language description in scientific publications, much like a model implementation in python does not replace its high-level description in ML papers. Similarly to ML model descriptions, lab protocols are often not specified enough to support direct conversion to low-level programs. For example, the protocol in Figure 1 does not specify the swirling (mixing) speed or its duration.

Our process execution graph (PEG) captures the predicate-argument structure of the protocol, allowing it to be more lenient than a programming language (for example, capturing that *gently* modifies `swirl`). Better suited to represent underspecified natural language, PEGs can serve as a convenient scaffold to support downstream tasks such as text-to-code assistants (Mehr et al., 2020). For example, by asking researchers to fill in missing required arguments for `swirl`.

To annotate PEGs, we leverage the sentence-level annotations of Kulkarni et al. (2018) (WLP henceforth). WLP, exemplified at the top of

---

[2] https://autoprotocol.org/specification

Figure 1, collected sentence-level structures using the BRAT annotation tool (Stenetorp et al., 2012). For example, capturing that *cells*, *culture tubes* are arguments for `add`. However, WLP does not capture cross-sentence implicit relations such that *culture tubes* are an argument for `incubate`. These are abundant in lab protocols, require tracking entities across many sentences, and are not easy to annotate using BRAT (see discussion in §4). We vastly extend upon WLP annotations, aiming to capture the full set of expressed protocol relations, using a novel text-based games annotation interface which lends itself to procedural text annotation.

3 Task Definition: Process Execution Graphs

Intuitively, we extend the WLP annotations (Kulkarni et al., 2018) from the sentence level to entire documents, aiming to capture *all* of the relations in the protocol. Formally, our representation is a directed, labeled, acyclic graph structure, dubbed a Process Execution Graph (PEG), exemplified in Figures 1 and 3, and formally defined below.

Nodes PEG nodes are triggered by explicit text spans in the protocol, e.g., “swirl", or “ice”. Nodes consist of two types: (1) *predicates*, marked in orange: denoting lab operations, such as `add` or `incubate`; and (2) *arguments*, marked in blue: representing physical lab objects (e.g., *culture tubes*, *cells*), exact quantities (*30 minutes*), or abstract instructions (e.g., *gently*).

Operation type	Frequent example spans	Count	Pct.
Transfer	add, transfer, place	1301	33.2
Temperature Treatment	incubate, store, thaw	503	12.8
General	Initiate, run, do not vortex	469	11.9
Mix	mix, vortex, inverting	346	8.8
Spin	spin, centrifuge, pellet	282	7.2
Create	prepare, make, set up	178	4.5
Destroy	discard, decant, pour off	170	4.3
Remove	remove, elute, extract	168	4.3
Measure	count, weigh, measure	149	3.8
Wash	wash, rinse, clean	146	3.7
Time	wait, sit, leave	114	2.9
Seal	cover, seal, cap	68	1.7
Convert	change, transform, changes	21	0.5

Table 1: Details of PEG predicate types, along with example frequent trigger spans and relative frequency in X-WLP.

Argument type	Frequent example spans	Count	Pct.
Reagent	supernatant, dna, sample	3362	32.6
Measurement	1.5 mL, 595nm, 1pmol	1924	18.6
Setting	overnight, room temperature	1622	15.7
Location	tube, ice, plates	1373	13.3
Modifier	gently, carefully, clean	1070	10.3
Device	forceps, pipette tip	590	5.7
Method	dilutions, pipetting	271	2.6
Seal	lid, cap, aluminum foil	97	0.9

Table 2: Details of PEG argument types, along with example frequent trigger spans and relative frequency in X-WLP.

Node grounding The PEG formulation above is motivated as a scaffold towards fully-executable lab programs employed in automatic lab environments. To achieve this, we introduce an ontology for each of the node types, based on the Autoprotocol specification (Lee and Miles, 2018), as indicated below each text span in Figures 1 and 3. For example, `swirl` corresponds to an Autoprotocol `mix` operation, a *culture tube* is of type *location*, and *30 minutes* is a *setting*. See Tables 1, 2 for details of predicate and argument types respectively, their frequencies in our data and example spans.

Edges Following PropBank notation (Kingsbury and Palmer, 2003), PEGs consist of three types of edges derived from the Autoprotocol ontology, and denoted by their labels: (1) core-roles (e.g., “ARG0”, “ARG1”): indicating predicate-specific roles, aligning with Autoprotocol’s ontology. For example, *ARG0* of `mix` assigns the element to be mixed; (2) non-core roles (e.g., “setting”, “site”, or “co-ref”): indicate predicate-agnostic relations. For example, the *site* argument always marks the location in which a predicate is taking place; and (3) temporal edges, labeled with a special “succ” label: define a temporal transitive ordering between predicates. In Figure 1, `add` occurs before `swirl`, which occurs before `incubate`. See Table 3 for predicate-specific core-role semantics, and Table 6 for non-cores roles types and frequencies of all roles in X-WLP. See Appendix A.3 for the rules defining what relations can hold between various entity types.

Relation to Autoprotocol As shown at the bottom of Figure 1, a PEG is readily convertible to Autoprotocol or similar laboratory interfaces once it is fully instantiated, thanks to edge labels and node grounding to an ontology. For example, a

Operation	Role Semantics	Required
Spin	ARG0 centrifuged to produce solid phase ARG1 and/or liquid phase ARG2	ARG0
Convert	ARG0 converted to ARG1	ARG0, ARG1
Seal	ARG0 sealed with ARG1	ARG0
Create	ARG* are created	ARG0
General	-	ARG0
Destroy	ARG* discarded	ARG0
Measure	ARG* to be measured	ARG0
Mix	ARG* are mixed	ARG0
Remove	ARG0 removed from ARG1	ARG0
Temperature Treatment	ARG* to be heated/cooled	ARG0
Time	Wait after operation on ARG0	ARG0
Transfer	ARG* are sources, transferred to "site"	ARG0, site
Wash	ARG0 washed with ARG1	ARG0

Table 3: Details of core role semantics for all operation types. The "Required" column specifies which roles must be filled for a given operation. ARG* is short for {ARG0, ARG1, ARG2}.

researcher can specify what *gently* means in terms of mixing speed for their particular lab instruments.

Reentrancies and cross-sentence relations While the PEG does not form directed cycles,[3] it does form non-directed cycles (or *reentrancies*) – where there exists nodes u, v such that there are two different paths from u to v. This occurs when an object participates in two or more temporally-dependent operations. For example, see *culture tubes*, which participates in all operations in Figure 1. In addition, edges (u, v) may be triggered either by *within-sentence relations*, when both u and v are triggered by spans in the same sentence, or by *cross-sentence relations*, when u and v are triggered by spans in different sentences. In the following section we will show that both reentrancies and cross-sentence relations, which are not captured by previous annotations, are abundant in our annotations.

4 Data Collection: The X-WLP Corpus

In this section, we describe in detail the creation of our annotated corpus: X-WLP. The protocols in X-WLP are a subset (44.8%) of those annotated in the WLP corpus. These were chosen because they are covered well by Autoprotocol's ontology (for details on ontology coverage, see §A.1).

In total, we collected 3,708 sentences (54.1K tokens) in 279 wet lab protocols annotated with our graph representation. As can be seen in Table 4, X-WLP annotates long examples, often spanning dozens of sentences, and its size is comparable (e.g., in terms of annotated words) to the ProPara corpus (Dalvi et al., 2018) and other related procedural datasets.

	X-WLP (ours)	MSPTC	CSP	ProPara
# words	54k	56k	45k	29k
# words / sent.	14.6	26	25.8	9
# sentences	3,708	2,113	1,764	3,300
# sentences / docs.	13.29	9	N/A	6.8
# docs.	279	230	N/A	488

Table 4: Statistics of our annotated corpus (X-WLP). X-WLP annotates complex documents, constituting more than 13 sentences on average. X-WLP overall size is on par with other recent procedural corpora, including ProPara (Dalvi et al., 2018), material science (MSPTC; Mysore et al. (2019)) and chemical synthesis procedures (CSP; Vaucher et al. (2020)). CSP is comprised of annotated sentences (document level information is not provided).

4.1 WLP as a Starting Point

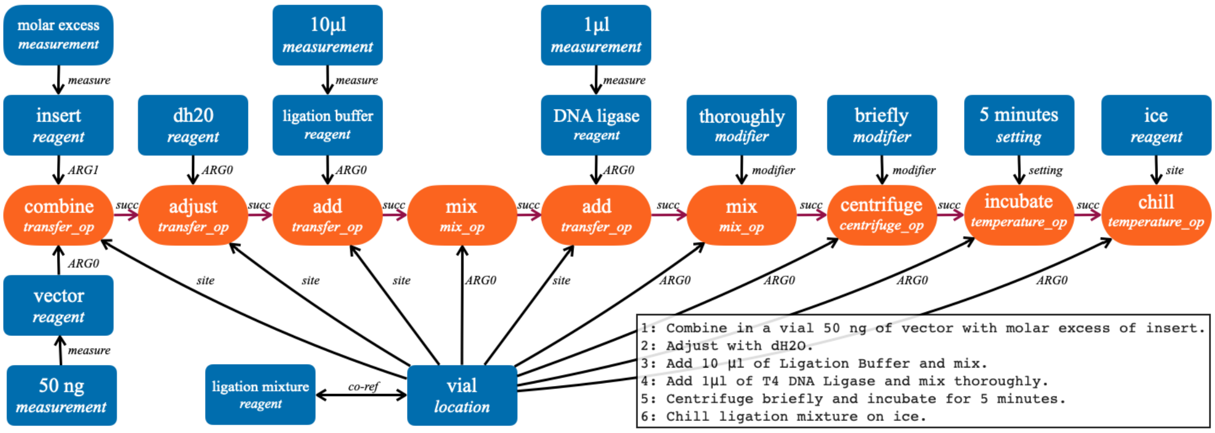
Despite WLP's focus on sentence-level relations (see top of Figure 1), it is a valuable starting point for a document-level representation. We pre-populate our PEG representations with WLP's gold object mentions (e.g., *cells*, *30 minutes*), operation mentions (`swirl` and `incubate`), and within-sentence relations (e.g., between *gently* and `swirl`). We ask annotators to enrich them with type grounding for operations and arguments, as well as cross-sentence relations, as defined in §3. From these annotations we obtain process-level representations as presented in Figures 1 and 3.

4.2 Process-Level Annotation Interface: Text-Based Simulator

Annotating cross-sentence relations and grounding without a dedicated user interface is an arduous and error-prone prospect. Consider as an example the *ligation mixture* mention in Figure 3. This mention is a metonym for *vial* (5 sentences earlier), after mixing in the *ligase*. This kind of metonymic co-reference is known to be difficult for annotation (Jurafsky and Martin, 2009), and indeed, such complicated annotation has been a factor in the omission of cross-sentence information in similar domains (Mysore et al., 2019). A simulator can provide a natural way to account for it by

---

[3] This happens because the temporal relations define a partial ordering imposed by the linearity of the execution.



1: Combine in a vial 50 ng of vector with molar excess of insert.
2: Adjust with dH2O.
3: Add 10 µl of Ligation Buffer and mix.
4: Add 1µl of T4 DNA Ligase and mix thoroughly.
5: Centrifuge briefly and incubate for 5 minutes.
6: Chill ligation mixture on ice.

Figure 3: A full process gold PEG annotation from X-WLP for a real-world wet lab protocol whose text is presented in the lower right corner (protocol 512), exemplifying several common properties: (1) complex, technical language, in relatively short sentences; (2) a chain of temporally-dependent, cross-sentence operations; (3) a common object that is being acted upon through side effects throughout the process (*vial*); and (4) *vial* is mostly omitted in the text after being introduced in the first sentence, despite participating in all following sentences. In the last sentence it appears with a metonymic expression (*ligation mixture*).

representing the relevant temporal and contextual information: after sentence 4, *vial* contains the *ligation buffer* mixed with other entities.

To overcome these challenges and achieve high-quality annotations for this complex task, we develop a simulator annotation interface, building upon the TextWorld framework (Côté et al., 2018). This approach uses text-based games as the underlying simulator environment, which we adapt to the biochemistry domain. The human annotator interacts with the text-based interface to simulate the raw wet lab protocol (Figure 2): setting the types of operations (the first interaction sets the span “chill" as a `temperature` operation) and assigning their inputs (the last line assigns `vial` as an input to `chill`), while the simulator tracks entity states and ensures the correct number and type of arguments, based on the Autoprotocol ontology. For example, the second interaction in Figure 2 indicates a missing argument for the `chill` operation (the argument to be chilled). Finally, tracking temporal dependency (“succ” edges) is also managed entirely by the simulator by tracking the order in which the annotator issues the different operations.

Further assistance is provided to annotators in the form of an auto-complete tool (last interaction in Figure 2), visualization of current PEG and a simple heuristic “linter” (Johnson, 1977) which flags errors such as ignored entities by producing a score based on the number of connected components in the output PEG.

See the project web page for the complete annotation guidelines, visualizations of annotated protocols, and demonstration videos of the annotation process.

4.3 Data Analysis

Four in-house CS undergraduate students with interest in NLP used our simulator to annotate the protocols of X-WLP, where 44 of the protocols were annotated by two different annotators to estimate agreement.

Inter-annotator agreement. We turn to the literature on abstract meaning representation (AMR; Banarescu et al., 2013) for established graph agreement metrics, which we adapt to our setting. Similarly to our PEG representation, the AMR formalism has predicate and argument nodes (lab operations and entities in our notation) and directed labeled edges which can form undirected cycles through reentrancies (nodes with multiple incoming edges).[4] In Table 5 we report a graph Smatch score (Cai and Knight, 2013) widely used to quantify AMR’s graph structure agreement, as well as finer grained graph agreement metrics, adapted from Damonte et al. (2017). Smatch values are comparable to those obtained for AMR, where reported gold agreement

---

[4] Unfortunately, we cannot follow this analogy to train AMR models on our graphs, since, to the best of our knowledge, they are currently limited to single sentences, notwithstanding a promising recent initial exploration into multi-sentence AMR annotation (O’Gorman et al., 2018).

Agreement Metric	**F1**
Smatch	84.99
Argument identification	89.72
Predicate identification	86.68
Core roles	80.52
Re-entrancies	73.12

Table 5: X-WLP inter-annotator agreement metrics. Smatch (Cai and Knight, 2013) quantifies overall graph structure. Following metrics provide a finer-grained break down (Damonte et al., 2017).

Relation	# Intra.	# Inter.	Total	# Re-entrancy
Core				
• ARG0	2962	952	3914	1645
• ARG1	560	127	687	3
• ARG2	84	123	207	77
Total (core)	3606	1202	4808	1725
Non-Core				
• site	1306	325	1631	360
• setting	3499	2	3501	-
• usage	1114	24	1138	-
• co-ref	129	1575	1704	-
• located-at	199	72	271	-
• measure	2936	18	2954	-
• modifier	1861	2	1863	-
• part-of	72	65	137	-
Total (non-core)	11116	2083	13199	360
Temporal	1218	788	2006	-
Grand Total	15940 (80%)	4073 (20%)	20013	2085

Table 6: Breakdown of PEG relation types by frequency in X-WLP, showing counts of inter/intra-sentence relations. Re-entrancies are possible only for core and "site" arguments, and may be either inter or intra-sentence.

varies between 0.69 – 0.89 (Cai and Knight, 2013), while our task deals with longer, paragraph length representations. Reentrancies are the hardest for annotators to agree on, probably since they involve longer-range, typically cross-sentence relations. On the other hand, local decisions such as argument and predicate identification achieve higher agreement, and also benefit greatly from the annotations of WLP.

Information gain from process-level annotation. Analysis of the relations in X-WLP, presented in Table 6, reveals that a significant proportion of arguments in PEGs are re-entrancies (32.4%) or cross-sentence (50.3%).[5] Figure 3 shows a representative example, with the *vial* participating in multiple re-entrancies and long-range relations, triggered by each sentence in the protocol. These relations are crucial to correctly model the protocols at the process level, and are inherently missed by sentence-level formalisms, showing the value of our annotations.

Dataset	Avg. #args/op	#Ops. w/o core arg.	#Ops.	Pct.
WLP	1.87	3297	17485	18.9
X-WLP	3.01	0	3915	0.0

Table 7: Comparison of average arguments per operation and percentage of semantically under-specified operations (missing core arguments) in WLP and X-WLP.

To shed light on the additional process-level information captured by our approach relative to WLP, in Table 7 we compare the average number of arguments per operation node as well as the amount of operation nodes with no core arguments. For example, see the `swirl` instruction at the top of Figure 1: in WLP, this predicate has no core role argument and is thus semantically under-defined. X-WLP correctly captures the core role of *culture tubes*. By definition, our use of input validation by the simulator prevents semantic under-specification, which is likely a significant factor in the higher counts for cross-sentence relations and overall average arguments in X-WLP.

Annotation cost. The time to annotate an average document of 13.29 sentences was approximately 53 minutes (roughly 4 minutes per sentence), not including annotator training. Our annotator pay was 13 USD / hour. The overall annotation budget for X-WLP was roughly 3,200 USD.

5 Models

We present two approaches for PEG prediction. First, in §5.1 we design models for separate graph sub-component prediction, which are chained to form a pipeline PEG prediction model. Second, in §5.2 we present a model which directly predicts the entire PEG using a span-graph prediction approach.

5.1 Pipeline Model (PIPELINE)

A full PEG representation as defined in §3 can be obtained by chaining the following models which predict its sub-components. In all of these, we use SciBERT (Beltagy et al., 2019) which was trained on scientific texts similar to our domain.

---

[5] For these calculations we consider only argument relations that can in principle occur as re-entrancies: "ARG*" and "site", see relation ontology in Appendix A.3 for details. Cross-sentence calculation includes co-reference closure information.

Mention identification. Given a scientific protocol written in natural language, we begin by identifying all experiment-involved text spans mentioning lab operations (predicates) or entities and their traits (arguments), which are the building blocks for PEGs. We model this problem of mention identification as a sequence tagging problem. Specifically, we transfer span-level mention labels, which are annotated in the WLP corpus into token-level labels using the BIO tagging scheme, then fine-tune the SciBERT model for token classification.

Predicate grounding. Next, we ground predicate nodes into the operation ontology types discussed in §3. See Table 1 in the Appendix for the complete list. Predicted mentions are marked using special start and end tokens (`[E-start]` and `[E-end]`), then fed as input to SciBERT. The contextual embedding of `[E-start]` is input to a linear softmax layer to predict the fine-grained operation type.

Operation argument role labeling. Once the operation type is identified, we predict its semantic arguments and their roles. Given an operation and an argument mention, four special tokens are used to specify the positions of their spans (Baldini Soares et al., 2019). Type information is also encoded into the tokens, for example, when the types of the operator and its argument are `mix-op` and `reagent` respectively, four special tokens `[E1-mix-op-start]`, `[E1-mix-op-end]`, `[E2-rg-start]` and `[E2-rg-end]` are used to denote the spans of the mention pair. After feeding the input into SciBERT, the contextualized embeddings of `[E1-op-mix-start]` and `[E2-rg-start]` are concatenated as input to a linear layer that is used to predict the entity's argument role. Arguments of an operation can be selected from anywhere in the protocol, leading to many cross-sentence operation-argument link candidates. To accommodate cross-sentence argument roles, we use the entire document as input to SciBERT for each mention pair. However, SciBERT is limited to processing sequences of at most 512 tokens. To address this limitation, longer documents are truncated in a way that preserves surrounding context, when encoding mention pairs.[6] Only 8 of the 279 protocols in our dataset contain more than 512 tokens.

Temporal ordering. Finally, we model order of operations using the `succ` relation (see Figure 3). These are predicted using a similar approach as argument role labeling, where special tokens are used to encode operation spans.

5.2 Jointly-Trained Model (MULTI-TASK)

To explore the benefits of jointly modeling mentions and relations, we experiment with a graph-based multi-task framework based on DYGIE++ model (Wadden et al., 2019). Candidate mention spans are encoded using SciBERT, and a graph is constructed based on predicted X-WLP relations and argument roles. A message-passing neural network is then used to predict mention spans while propagating information about related spans in the graph (Dai et al., 2016; Gilmer et al., 2017; Jin et al., 2018).

This approach requires computing hidden state representations for all $O(n^4)$ pairs of spans in an input text, which for long sequences, will exhaust GPU memory. While Wadden et al. (2019) considered primarily within-sentence relations, our model must consider relations across the entire protocol, which makes this a problem of practical concern. To address this, we encode a sliding window of w adjacent sentences when the full protocol does not fit into memory, allowing smaller windows for the start and end of the protocol, and concatenate sentences within each window as inputs to the model. As a result, each sentence is involved in w windows leading to repeated, possibly contradicting predictions for both mentions and relations. To handle this, we output predictions agreed upon by at least k windows, where k is a hyperparameter tuned on a development set.

6 Experiments

In §5, we presented a pipelined approach to PEG prediction based on SciBERT and a message-passing neural network that jointly learns span and relation representations. Next, we describe the details of our experiments and present empirical results demonstrating that X-WLP supports training models that can predict PEGs from natural language instructions.

---

[6] Given an input document, which has more than 512 words, with n words between two mentions, we truncate the context to keep at most $(512 - n)/2$ words for each side.

Data Split	System	F_1
original	Kulkarni et al. (2018)	78.0
	Wadden et al. (2019)	**79.7**
	Pipeline	78.3
X-WLP-eval	Pipeline	74.7

Table 8: Mention identification test set F_1 scores for models on the WLP dataset. Top: WLP dataset with the original train/dev/test split. Bottom: excluding X-WLP protocols from the WLP training data, and using them for evaluation.

Data. X-WLP is our main dataset including 279 fully annotated protocols. Statistics of X-WLP are presented in Table 4. Additionally, we have 344 protocols from the original WLP dataset. We use this auxiliary data only for training mention taggers in the pipeline model, and use X-WLP for all other tasks. For argument role labeling and temporal ordering, negative instances are generated by enumerating all possible mention pairs whose types appear at least once in the gold data. We use 5-fold cross validation; 2 folds (112 protocols) are used for development, and the other 3 folds (167 protocols) are used to report final results.

Model setup. The Pipeline framework employs a separate model for each task, by default using the propagated predictions from previous tasks as input. In addition, we evaluate the model for each task with gold input denoted as Pipeline `(gold)`. Finally, the Multi-task framework learns all tasks together and we decompose its performance into the component subtasks.

Implementation details. We use the uncased version of SciBERT[7] for all our models due to the importance of in-domain pre-training. The models under the Pipeline system are implemented using Huggingface Transformers (Wolf et al., 2020), and we use AdamW with the learning rate 2×10^{-5} for SciBERT finetuing. For the Multi-task framework, we set the widow size w to 5, the maximum value that enables the model to fit in GPU memory. For all other hyperparameters, we follow the settings of the WLP experiments in (Wadden et al., 2019).

6.1 Results

The results of the two models on the different subtasks are presented in Tables 8- 11. We identify three main observations based on these results.

System	P	R	F_1
Multi-task	76.0	69.0	72.3
Pipeline	71.8	76.3	74.0
• w/ gold mentions	79.0	80.2	79.6

Table 9: Predicate grounding test set results.

Task	Multi-task	Pipeline	# gold
Core			
• All roles	**57.9**	53.7	2839
• All roles (gold mentions)	-	76.5	2839
• ARG0	**61.0**	57.1	2313
• ARG1	**36.1**	32.9	412
• ARG2	**69.7**	61.4	114
Non-Core			
• All roles	**55.7**	48.8	4826
• All roles (gold mentions)	-	78.1	4826
• site	**58.7**	55.4	962
• setting	**77.4**	74.7	974
• usage	**35.6**	33.0	296
• co-ref	**39.8**	36.7	1014
• measure	**63.3**	56.6	804
• modifier	**51.0**	41.8	519
• located-at	9.7	**13.3**	179
• part-of	0.5	**10.8**	78
Temporal Ordering	**61.8**	57.3	2176
Temp. Ord. (gold mentions)	-	76.3	2176

Table 10: Operation argument role labeling (core and non-core roles, decomposed by relation) and temporal ordering test set F_1 performance.

Split	Multi-task	Pipeline	# gold
Intra-sentence	**63.4**	58.2	2160
Inter-sentence	**32.5**	39.1	679

Table 11: Operation argument role labeling (core roles) test set F_1, decomposed based on whether the operation and the argument are triggered within the same sentence (intra-sentence) versus different sentences (inter-sentence).

First, Pipeline outperforms Multi-task on the operation classification task in Table 9, as it uses all protocols from WLP as additional training data to improve mention tagging.

Second, Multi-task performs better than the Pipeline approach on most relation classification tasks in Table 10, but is worse than Pipeline when Pipeline uses gold mentions, demonstrating that jointly modeling mentions and relations helps in mitigating error propagation.

Third, *cross-sentence* relations are challenging for both models, as shown in Table 11. This explains the low performance of `co-ref`, which is comprised of 92.4% cross-sentence relations.

In addition, there are a couple of interesting points to note. In Table 8, the performance of Pipeline on the X-WLP subset is lower than its performance on the WLP test set, likely because

---

[7] `https://github.com/allenai/scibert`

there are fewer protocols in the training set. For the relation-decomposed performance in Table 10, we can see that some of the relations like "ARG2" can be correctly predicted by MULTI-TASK using only a few gold labels while some more widely used relations are harder to learn, such as "ARG0" and "site"; indeed, "ARG2" is only used in the `spin` operation (see Table 3), while the other roles participate in more diverse contexts.

7 Related Work

Natural Language Processing (NLP) for scientific procedural text is a rapidly growing field. To-date, most approaches have focused on text-mining applications (Isayev, 2019) and typically annotate only shallow, sentence-level semantic structures (e.g., Fig. 1, top). Examples include WLP (Kulkarni et al., 2018) and materials science procedures (Mysore et al., 2019; Kuniyoshi et al., 2020). Recent interest in automation of lab procedures has also led to sentence-level annotation of procedural texts with action sequences designed to facilitate execution (Vaucher et al., 2020).

However, as noted in recent concurrent work (Mehr et al., 2020), neither sentence-level semantic structures nor action sequences are sufficient for the goal of converting text to a machine-executable synthesis procedure; for this purpose, a more structured, process-level semantic representation is required. In particular, executable representations require a structured declaration of the locations and states of the different materials throughout a process, details not represented by sentence-level annotations. Our simulator can naturally represent such information by maintaining a stateful model of the process. Simulation fidelity can be controlled by implementing the execution semantics of operations to the level of detail required.

Mehr et al. (2020) have similarly proposed a process-level executable representation, but use an NLP pipeline consisting primarily of rules and simple pattern matching, relying on a human-in-the-loop for corrections; linking our approach with their framework is a promising future direction.

Structurally, PEGs are similar to abstract meaning representation (AMR; Banarescu et al. 2013), allowing us to use agreement and performance metrics developed for AMR. In contrast with the sentence-level AMR, a major challenge in this work is annotating and predicting procedure-level representations.[8]

Another line of research focuses on procedural text understanding for more general domains: simple scientific processes (Dalvi et al., 2018), open domain procedural texts (Tandon et al., 2020), and cooking recipes (Kiddon et al., 2015; Bosselut et al., 2018). These works represent process-level information and entity state changes, but typically feature shorter processes, simpler language and an open ontology, compared with our domain-specific terminology and grounded ontology.

Our framework also provides a link to text-based game approaches to procedural text understanding. Tamari et al. (2019) modelled scientific procedures with text-based games but used only synthetic data. Our simulator enables leveraging recent advances on text-based games agents (e.g., (Adhikari et al., 2020)) towards *natural* language understanding.

8 Conclusion

We developed a novel meaning representation and simulation-based annotation interface, enabling the collection of process-level annotations of experimental procedures, as well as two parsers (pipeline and joint modelling) trained on this data. Our dataset and experiments present several directions for future work, including the modelling of challenging long range dependencies, application of text-based games for procedural text understanding, and extending simulation-based annotation to new domains.

Acknowledgments

We would like to thank Peter Clark, Noah Smith, Yoav Goldberg, Dafna Shahaf, and Reut Tsarfaty for many fruitful discussions and helpful comments, as well as the X-WLP annotators: Pranay Methuku, Rider Osentoski, Noah Zhang and Michael Zhan. This work was partially supported by an Allen Institute for AI Research Gift to Gabriel Stanovsky. This material is based upon work supported by the NSF (IIS-1845670) and the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001119C0108. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

---

[8] In addition, in contrast with AMR, PEG nodes are directly mapped to the trigger spans in the document.

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and Will Hamilton. 2020. Learning dynamic belief graphs to generalize on text-based games. *Advances in Neural Information Processing Systems*, 33.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *LAW@ACL*.

Maxwell Bates, Aaron J Berliner, Joe Lachoff, Paul R Jaschke, and Eli S Groban. 2017. Wet lab accelerator: a web-based application democratizing laboratory automation for synthetic biology. *ACS synthetic biology*, 6(1):167–171.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Antoine Bosselut, Corin Ennis, Omer Levy, Ari Holtzman, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *International Conference on Learning Representations*.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, pages 41–75. Springer.

Hanjun Dai, Bo Dai, and Le Song. 2016. Discriminative embeddings of latent variable models for structured data. In *International conference on machine learning*, pages 2702–2711.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604, New Orleans, Louisiana. Association for Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for Abstract Meaning Representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Olexandr Isayev. 2019. Text mining facilitates materials discovery. *Nature*, 571(7763):42–43.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*.

Stephen C Johnson. 1977. *Lint, a C program checker*. Citeseer.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, second edition. Pearson Prentice Hall.

Ben Keller, Justin Vrana, Abraham Miller, Garrett Newman, and Eric Klavins. 2019. Aquarium: The Laboratory Operating System version 2.6.0.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. 2015. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 982–992, Lisbon, Portugal. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2003. Propbank: the next level of treebank.

Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. 2018. An annotated corpus for machine reading of instructions in wet lab protocols. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 97–106, New Orleans, Louisiana. Association for Computational Linguistics.

Fusataka Kuniyoshi, Kohei Makino, Jun Ozawa, and Makoto Miwa. 2020. Annotating and extracting synthesis process of all-solid-state batteries from scientific literature. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1941–1950, Marseille, France. European Language Resources Association.

Peter L Lee and Benjamin N Miles. 2018. Autoprotocol driven robotic cloud lab enables systematic machine learning approaches to designing, optimizing, and discovering novel biological synthesis pathways. In *SIMB Annual Meeting 2018*. SIMB.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

S. Hessam M. Mehr, Matthew Craven, Artem I. Leonov, Graham Keenan, and Leroy Cronin. 2020. A universal system for digitization and automatic execution of the chemical synthesis literature. *Science*, 370(6512):101–108.

Ben Miles and Peter L. Lee. 2018. Achieving reproducibility and closed-loop automation in biological experimentation with an iot-enabled lab of the future. *SLAS TECHNOLOGY: Translating Life Sciences Innovation*, 23(5):432–439. PMID: 30045649.

Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64.

Tim O'Gorman, Michael Regan, Kira Griffitt, Ulf Hermjakob, Kevin Knight, and Martha Palmer. 2018. AMR beyond the sentence: the multi-sentence AMR corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3693–3702, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Gurpur Rakesh D Prabhu and Pawel L Urban. 2017. The dawn of unmanned analytical laboratories. *TrAC Trends in Analytical Chemistry*, 88:41–52.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: A web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, page 102–107, USA. Association for Computational Linguistics.

Ronen Tamari, Hiroyuki Shindo, Dafna Shahaf, and Yuji Matsumoto. 2019. Playing by the book: An interactive game approach for action graph extraction from text. In *Proceedings of the Workshop on Extracting Structured Knowledge from Scientific Publications*, pages 62–71, Minneapolis, Minnesota. Association for Computational Linguistics.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417, Online. Association for Computational Linguistics.

Alain C. Vaucher, Federico Zipoli, Joppe Geluykens, Vishnu H. Nair, Philippe Schwaller, and Teodoro Laino. 2020. Automated extraction of chemical synthesis actions from experimental procedures. *Nature Communications*, 11(1):1–11.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Nozomu Yachie and Tohru Natsume. 2017. Robotic crowd biology with maholo labdroids. *Nature Biotechnology*, 35(4):310–312.

A Annotation Schema

In the following subsections, we provide further details of the annotation schema used. Section §A.1 describes how the ontology was constructed based on Autoprotocol, and §A.2 provides details on ontology coverage for the X-WLP protocols which were chosen for annotation. Section §A.3 details the rules defining valid PEG edges, or what relations can hold between various entity types. The annotation guidelines given to annotators are available on the project web page.

A.1 Ontology Construction

Operation nodes correspond to "action" entities in WLP. In X-WLP, to facilitate conversion to executable instructions, we further add a fine-grain operation type; for each operation, annotators were required to select the closest operation type, or a `general` type if none applied.

To define our operation type ontology, we consulted the Autoprotocol (Miles and Lee, 2018) open source standard used for executable biology lab protocols. Autoprotocol defines 35 different operation types,[9] from which we grouped relevant types into higher level clusters; X-WLP operation types are broadly aligned with Autoprotocol operation types, but are more general in scope, to not limit applicability to any one platform. For example, we use a more general `measure` operation type rather than the specific types of measurement operations in Autoprotocol (`spectrophotometry`, `measure-volume`, etc.).

Table 12 maps between X-WLP operation types and their equivalents in Autoprotocol, if one exists. The X-WLP operation types do not perfectly overlap with Autoprotocol as the former is written for humans, while the latter is designed for the more constrained domain of robot execution. Accordingly, some operations not currently supported in Autoprotocol were added, like `wash`. See Table 1 for example mention spans for each X-WLP operation type.

The set of supported operations was chosen to maximize coverage over the types of operations found in the sentence-level annotations of WLP (see §A.2 below for details).

A.2 Ontology Coverage

To identify candidate protocols for annotation which were well covered by the ontology, we created a mapping between ontology instruction types and the 100 most frequent text-spans of WLP action entities (constituting 74% of all action spans in WLP). WLP action text spans that didn't correspond to any ontology instruction were mapped to a `general` label; action text spans that could be mapped to the ontology we call ontology-covered actions. For annotation in X-WLP, we then selected WLP protocols estimated to have a high percentage of ontology-covered actions (based on the mapping above). This simple method was found to be effective in practice, as measured by the actual ontology coverage of X-WLP annotations, summarized in Fig. 4.

For each annotated protocol, we calculated the percentage of known (not `general`) operations. Fig. 4 plots, for each coverage percentile (y-axis), the percentage (x-axis) of X-WLP protocols with at least y percent known operations. From the plot we can see for example that half of the protocols in X-WLP have >90% ontology coverage, and 90% of the protocols have >70% ontology coverage.

X-WLP Operation	Autoprotocol Instructions
Spin	Spin
Convert	N/A
Seal	Seal, Cover
Create	Oligosynthesize, Provision
General	N/A
Destroy	N/A
Measure	Absorbance, Fluorescence, Luminescence, IlluminaSeq, SangerSeq, MeasureConcentration, MeasureMass, MeasureVolume, CountCells, Spectrophotometry, FlowCytometry, FlowAnalyze, ImagePlate
Mix	Agitate
Remove	Unseal, Uncover
Temperature Treatment	Thermocycle, Incubate, FlashFreeze
Transfer	AcousticTransfer, MagneticTransfer, Dispense, Provision, LiquidHandle, Autopick
Wash	N/A
Time	N/A

Table 12: Mapping between X-WLP operation types and corresponding Autoprotocol instructions (if any exist). Autoprotocol operations tend to be more specific as they are intended for machine execution. X-WLP protocols are written for humans, so operation types are defined at a higher level of abstraction.

---

[9] Based on `https://github.com/autoprotocol/autoprotocol-python/blob/master/autoprotocol/instruction.py` as of January 2021.

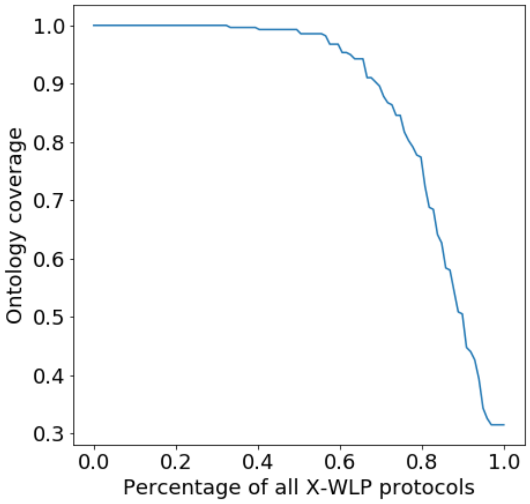


Figure 4: Plot displaying for each coverage percentile (y-axis), the percentage (x-axis) of X-WLP protocols with at least y percent known (ontology-covered) operations.

A.3 Syntax governing PEG edges

Formally, edges are represented by triplets of the form (s, r, t) where s and t are argument nodes and r is a core or non-core role. Dependent on a particular role r, certain restrictions may apply to the fine-grained type of s and t, as described below.

A.3.1 Core Roles

Core roles, displayed in Table 3, represent operation specific roles, for example "ARG1" for the `seal` operation is a *seal* entity representing the seal of the "ARG0" argument. For core roles, the following restrictions hold:

- Source nodes s are restricted to any of the *object* types $s \in \{reagent, device, seal, location\}$ representing physical objects. The only exception to this rule is that "ARG1" for the `seal` operation must be a *seal* entity.
- Target node t is a predicate of one of the types in Table 1.
- r is a core argument relation, $r \in \{\text{ARG0}, \text{ARG1}, \text{ARG2}\}$ or ARG* for short.
- Certain roles may be required for a valid predicate t, for example the `transfer` operation requires at minimum both source and target arguments to be specified by the ARG0 and "site" roles, respectively.

Role	Source Types	Target Types
co-ref	Object	Object
measure	Measurement	Object
setting	Setting	Object
modifier	Modifier	Object, Operation, Measurement
usage	Method, Object	Operation
located-at	Object	Object
part-of	Object	Object

Table 13: Details of non-core roles and restrictions on source and target node types. Object is short for the set of entity types representing physical objects: $\{reagent, device, seal, location\}$.

A.3.2 Non-core Roles

Non-core roles (e.g., "setting", "site", or "co-ref") indicate predicate-agnostic labels. For example, the *site* argument always marks the location in which a predicate is taking place. Non-core roles are displayed in Table 13, and role-specific restrictions on s and t are listed under "Source Types" and "Target Types", respectively.

Machine Translationese: Effects of Algorithmic Bias on Linguistic Complexity in Machine Translation

Eva Vanmassenhove$^\alpha$ **Dimitar Shterionov**$^\alpha$ **Matthew Gwilliam**$^\beta$
$^\alpha$ Cognitive Science and AI, Tilburg University, The Netherlands
$^\beta$ University of Maryland, College Park
`{e.o.j.vanmassenhove, d.shterionov}@tilburguniversity.edu`
`mgwilliam@umd.edu`

Abstract

Recent studies in the field of Machine Translation (MT) and Natural Language Processing (NLP) have shown that existing models amplify biases observed in the training data. The amplification of biases in language technology has mainly been examined with respect to specific phenomena, such as gender bias. In this work, we go beyond the study of gender in MT and investigate how bias amplification might affect language in a broader sense. We hypothesize that the 'algorithmic bias', i.e. an exacerbation of frequently observed patterns in combination with a loss of less frequent ones, not only exacerbates societal biases present in current datasets but could also lead to an artificially impoverished language: 'machine translationese'. We assess the linguistic richness (on a lexical and morphological level) of translations created by different data-driven MT paradigms – phrase-based statistical (PB-SMT) and neural MT (NMT). Our experiments show that there is a loss of lexical and morphological richness in the translations produced by all investigated MT paradigms for two language pairs (EN↔FR and EN↔ES).

1 Introduction

The idea of translation entailing a transformation is widely recognised in the field of Translation Studies (TS) (Ippolito, 2014). Translations are specific communicative acts occurring in a particular context governed by their own laws. Some of the features that characterize translated texts are defined as simplification, explicitation, normalization and leveling out (Baker, 1999). The fingerprints left by the translation process and the language this results into, have been referred to as 'translationese' (Gellerstam, 1986). Empirical evidence of the existence of translationese can be found in studies showing that machine learning techniques can be employed to automatically distinguish between human translated and original text by looking at lexical and grammatical information (Baroni and Bernardini, 2006; Koppel and Ordan, 2011). Translationese differs from original texts due to a combination of factors including intentional (e.g. explicitation and normalization) and unintentional ones (e.g. unconscious effects of the source language input on the target language produced). Unlike other work on (human) translationese (or even related work on 'Post-editese'), we delve into the effects of machine translation (MT) algorithms on language, i.e. 'machine translationese'.

So far, generating accurate and fluent translations has been the main objective of MT systems. As such, maintaining the richness and diversity in the outputs has understandably not been a priority (Vanmassenhove, 2020).[1] However, as MT systems have reached a quality that is (arguably) close to that of human translations (Läubli et al., 2018; Toral et al., 2018) and as such are being used widely on a daily basis, we believe it is time to look into the potential effects of (MT) algorithms on language itself.[2]

The main motivations behind this work are: (i) if algorithmic bias is indeed a by-product of our algorithms, a statistically biased MT system might prefer frequently occurring words (or sub-words) over others. Since MT systems do not necessarily distinguish between different synonymous translations (lexical richness) and morphological variants (grammatical richness), algorithmic bias could lead to the loss of morphological variety (and thus interfere with the ability of our systems to generate

---

[1] One might argue that for some tasks and domains, diversity can be harmful (specific in-domain translations would prefer consistency over e.g. lexical diversity).

[2] Google Translate alone translates more than 100 billions words per day and is used by at least 500 million people according to estimates (`https://www.blog.google/products/translate/ten-years-of-google-translate/`).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2203–2213
April 19 - 23, 2021.

at all times a grammatically correct option); (ii) the sociolinguistic perspective of machine translationese since it has been established that language contact (e.g. via translationese) can entail language changes (Kranich, 2014). If machine translationese (and other types of 'NLPese') is a simplified version of the training data, what does that imply from a sociolinguistic perspective and how could this affect language on a longer term?

The main objective of the presented research is to establish whether there is indeed a quantitatively measurable difference between the linguistic richness of an MT system's training data and its output in terms of morphological and lexical diversity. To do so, we conduct an in-depth analysis that goes beyond frequently used standard lexical diversity metrics such as TTR, Yule's I and MTLD. We assess the lexical and morphological diversity through an adapted version of the Lexical Frequency Profile used to assess language acquisition, a measure of morphological diversity based on Shannon and Simpson Diversity and an novel automatic evaluation of synonym frequency. We focus on the most prominent data-driven MT paradigms: Neural MT (NMT), both LSTM (Bahdanau et al., 2015; Sutskever et al., 2014) and Transformer (Vaswani et al., 2017), and Phrase-Based Statistical MT (PB-SMT). Up to our knowledge this is the first research on lexical and morphological diversity of machine translation output, i.e. machine translationese.

The contributions of this work can be summarised as: (i) a detailed analysis of lexical and morphological diversity of machine translationese and the loss thereof to quantify the effects of algorithmic bias; (ii) the adaptation of a metric used in language acquisition for assessing lexical sophistication in MT[3]; (iii) the use of Shannon entropy and Simpson diversity to measure morphological richness, and (iv) a novel, automatic evaluation of synonym frequency.

2 Related Work

Several studies have exposed the societal biases present in datasets (racial bias (Merullo et al., 2019), political bias (Fan et al., 2019), gender bias (Vanmassenhove and Hardmeier, 2018). Existing NLP technology are likely to pick up biases present in the training data and various explorations of e.g. gender bias in NLP systems have indeed revealed the existence of harmful biases in the output they generate (Bolukbasi et al., 2016; Caliskan-Islam et al., 2016; Garg et al., 2018; Vanmassenhove et al., 2018; Stanovsky et al., 2019; Sun et al., 2019; Habash et al., 2019). Research related to bias has often focused on gender or race. Especially in a field such as MT, the implicit gender in a language such as English and its consecutive translations into morphologically richer languages with gender agreement, makes it relatively easy to expose and study biases related to gender in a contrastive linguistic setting. In the context of this paper, we would like to note that (statistical) bias is not limited to gender or race but can be defined as any systematic inaccuracy in one direction leading to an under (or over) estimation of observations.

A handful of recent work has mentioned the possibility of algorithmic bias on top of the already existing societal biases in the training data (Bolukbasi et al., 2016; Caliskan-Islam et al., 2016; Garg et al., 2018). For instance, Zhao et al. (2017) observe a phenomenon they refer to as 'bias amplification'. They note that in their training data an activity such as 'cooking' is associated 33% times more with women compared to men. After training a model on that dataset, the existing disparity is amplified to 68% times more associations with women.

In the field of MT, Vanmassenhove et al. (2019) address the effects of statistical bias on language generation in an MT setting. They assess lexical diversity using standard metrics –TTR, MTLD and Yule's K– and conclude that the translations produced by various MT systems (PB-SMT and NMT) are consistently less diverse than the original training data. Their approach was conducted on NMT systems that were trained without byte-pair-encoding (BPE) (Sennrich et al., 2016) which limits the creativity of the translation systems.

Toral (2019) measures the lexical diversity of 18 state-of-the-art systems on 6 language pairs, reaching similar conclusions. They do so focusing specifically on post-editese. The experiments indicate that post-editese is simpler and more normalised than human translationese. The post-editese also shows a higher degree of interference from the source compared to the human translations. Daems et al. (2017), like Toral (2019), centers around the automatic detection of post-editese and does not look into properties of unedited machine translationese. In Aranberri (Aranberri, 2020) different freely available MT systems (neural and rule-

---

[3] In fact, our implementation of the LFP metric can be employed for any NLP tasks.

based) are compared in terms of automatic metrics (BLEU, TER) and translationese features (TTR, length ratio input/output, perplexity, etc.) to investigate how such features correlate with translation quality. Bizzoni et al. (2020) presents a comparison using similar translationese features of three MT architectures and the human translations of spoken and written language.

In the field of PB-SMT, Klebanov and Flor (2013) show that PB-SMT suffers considerably more than human translations (HT) from lexical loss, resulting in loss of lexical tightness and text cohesion. Aharoni et al. (2014) proof that automatic and human translated sentences can be automatically identified corroborating that human translations systematically differ from the translations produced by PB-SMT systems.

Aside from Vanmassenhove et al. (2019), the above discussed related work uses metrics of lexical diversity to compare human translations to (post-edited) machine translations. In this work, we compare how and whether the output of an MT system differs (in terms of lexical and morphological diversity) from the data it was originally trained on. This way, we aim to investigate the effect of the algorithm (and algorithmic bias) on language itself.

3 Machine Translation Systems

MT paradigms have changed quickly over the last decades. Since this is the first attempt to quantify both the lexical and grammatical diversity of machine translationese, we experimented with the current state-of-the-art data-driven paradigms, LSTM and Transformer, as well as with PB-SMT. We used data from the Europarl corpus (Koehn, 2005) for two language pairs, English–French and English–Spanish in both direction (EN→FR, FR→EN, EN→ES and ES→EN). We are interested in both directions in order to verify whether there is a difference in terms of (the potential loss of) diversity when comparing translations from a morphologically poorer language (English) into morphologically richer ones (French and Spanish) and vice versa. Our data is summarised in Table 2.[4]

Lang. pair	Train	Test	Dev
EN-FR/FR-EN	1,467,489	499,487	7,723
EN-ES/ES-EN	1,472,203	459,633	5,734

Table 1: Number of parallel sentences for the training, testing and development sets.

The specifics of the MT systems we trained are:

PB-SMT For the PB-SMT systems we used Moses (Koehn et al., 2007) with default settings and a 5-gram language model with pruning of bigrams. We also tuned each system using MERT (Och and Ney, 2003) until convergence or for a maximum of 25 iterations. During translation we mask unknown words with the UNK token to avoid bleeding through (source) words which would artificially increase the linguistic diversity.

NMT For the RNN and Transformer systems we used OpenNMT-py.[5] The systems were trained for maximum of 150K steps, saving an intermediate model every 5000 steps or until reaching convergence according to an early stopping criteria of no improvements of the perplexity (scored on the development set) for 5 intermediate models. The options we used for the neural systems are:

- RNN: size: 512, RNN type: bidirectional LSTM, number of layers of the encoder and of the decoder: 4, attention type: MLP, dropout: 0.2, batch size: 128, learning optimizer: Adam (Kingma and Ba, 2014) and learning rate: 0.0001.
- Transformer: number of layers: 6, size: 512, transformer_ff: 2048, number of heads: 8, dropout: 0.1, batch size: 4096, batch type: tokens, learning optimizer Adam with $\text{beta}_2 = 0.998$, learning rate: 2.

All NMT systems have the learning rate decay enabled and their training is distributed over 4 nVidia 1080Ti GPUs. The selected settings for the RNN systems are optimal according to Britz et al. (2017); for the Transformer we use the settings suggested by the OpenNMT community[6] as the optimal ones that lead to quality on par with the original Transformer work (Vaswani et al., 2017).

For training, testing and validation of the systems we used the same data. To build the vocabularies for the NMT systems we used sub-word units, allowing NMT to be more creative; using sub-word units also mitigates to a certain extent the out of vo-

---

[4] We ought to address the fact that the Europarl data consists of both human-uttered and translated text which have different properties in terms of diversity. In this work we analyse the impoverishment of data when it passes through the "filter" of the MT system, i.e. the effect of algorithm. As the origin of the data, human-uttered or translated, has no impact on the inherent workings of the MT system we do not take this into account in our analysis.

[5] `https://opennmt.net/OpenNMT-py/`

[6] `http://opennmt.net/OpenNMT-py/FAQ.html`

cabulary problem. To compute the sub-word units we used BPE with 50,000 merging operations for all our data sets. In Table 2 we present the vocabulary sizes of the data used to train our PB-SMT and NMT systems.

Lang. pair	no BPE		with BPE	
	EN	FR/ES	EN	FR/ES
EN-FR/FR-EN	113,132	131,104	47,628	48,459
EN-ES/ES-EN	113,692	168,195	47,639	49,283

Table 2: Vocabulary sizes. For completeness we also present the vocabulary size without BPE, i.e. the number of unique words in the corpora.

The quality of our MT systems is evaluated on the test set using standard evaluation metrics – BLEU (Papineni et al., 2002) (as implemented in SacreBLEU (Post, 2018)) and TER (Snover et al., 2006) (as implemented in Multeval (Clark et al., 2011)). Our evaluation scores are presented in Table 3.

English as source

System	EN→FR		EN→ES	
	BLEU↑	TER↓	BLEU↑	TER↓
PB-SMT	35.7	50.7	38.6	45.9
LSTM	34.2	50.9	38.2	45.3
TRANS	**37.2**	**48.7**	**40.9**	**43.4**

English as target

System	FR→EN		ES→EN	
PB-SMT	36.2	47.1	39.3	44.0
LSTM	34.6	48.2	38.1	44.7
TRANS	**37.0**	**46.4**	**41.3**	**41.4**

Table 3: Quality evaluation scores for our MT systems. TRANS denotes Transformer systems.

We computed pairwise statistical significance using bootstrap resampling (Koehn, 2004) and a 95% confidence interval. The results shown in Table 3 are all statistically significant based on 1000 iterations and samples of 100 sentences. All metrics show the same performance trends for all language pairs: Transformer (TRANS) outperforms all other systems, followed by PB-SMT, and LSTM.

For all PB-SMT we replaced marked unknown words with only one token "UNK". While this does not effect the computation of BLEU and TER, it allows us not to artificially boost the lexical and grammatical scores for these MT engines (see Section 4) and assess their realistic dimensions.

4 Experiments and Results

Assessing linguistic complexity is a multifaceted task spanning over various domains (lexis, morphology, syntax, etc.). The lexical and grammatical diversity are two of its major components (Bachman, 2004; Bulté et al., 2008; Bulté, 2013). As such, we conduct an analysis using (i) lexical diversity and sophistication metrics (Section 4.2) and (ii) grammatical diversity metrics (Section 4.3). For lexical diversity, we use the following metrics: an adapted version of the Lexical Frequency Profile (LFP), three standard metrics commonly used to assess diversity –TTR, Yule's I and MTLD–, and three new metrics based on synonym frequency in translations. Up to our knowledge, this research is the first to employ LFP to analyze synthetic data. For grammatical diversity, we focus specifically on morphological inflectional diversity. We adopt the Shannon entropy and Simpson's diversity index to compute the entropy of the inflectional paradigms of lemmas, measuring the abundance and the evenness of wordforms per lemma.

Next, we will discuss the evaluation data and the aforementioned metrics designed in order to compare the diversity of the training data with the machine translationese. Our evaluation scripts are available at `https://github.com/dimitarsh1/BiasMT`; due to its large size, the data is not hosted in the github repository but is available upon request.

4.1 Evaluation data

To observe the effects of the MT algorithm on linguistic diversity, we used the MT engines (Section 3) to translate the source side of the training set, i.e. completely observed data. Data that has fully been observed during training is most suitable for our objectives as we are interested in the effects of the algorithm on language itself. It is also the most favourable translation (and evaluation) scenario for the MT systems since all data has been observed.

4.2 Lexical Diversity

Lexical Frequency Profile To look at the lexical sophistication and diversity in the text produced by the MT systems, we adapted the Lexical Frequency Profile (LFP) method (Laufer, 1994; Laufer and Nation, 1995). LFP is a measure that stems from research in second language (L2) acquisition and student writing methods. It is designed to study the lexical diversity or sophistication in texts produced by L2 learners. It is based on the observation that texts including a higher proportion of less frequent words are more sophisticated than those containing higher proportions of more frequent words (Kyle, 2019).

The LFP method measures diversity and sophistication by looking at frequency bands. In its original version, the LFP analysis would distinguish between 4 bands: (i) percentage of words in a text belonging to the 1000 most frequent words in that language, (ii) percentage of words in a text belonging to the next 1000 most frequent words, (iii) a list of academic words that did not occur in the first 2000 words and (iv) the remaining words. The lists used to determine the word bands are predefined word lists such as Nation's word lists (Nation, 1984). One shortcoming of the approach is that a mismatch between reference corpus and the target text can lead to misleading outcomes. However, since we are looking into the (side-)effects of the training algorithm, instead of using preset word lists in order to compute the LFP, we use the original training data to generate the word frequency lists. This allows for a better comparison between the original data and the machine translationese while bypassing the potential mismatch issue.

Several studies (Crossley et al., 2013; Laufer, 1994; Laufer and Nation, 1995) have employed the LFP method to assess L2 acquisition in learners. From these studies, it resulted that the less proficient a user of an L2, the more words belonged to the first band and the least words belong to the list of academic words or the remaining words (band 3 and 4 respectively of the original formulation).

The lexical profile mentioned above is a detailed profile, showing 4 types of words used by the learner. Because of interpretability issues, the 'Beyond 2000' is also frequently used to assess the profile of the users. It distinguishes between the first two bands (comprising of the first 2000 words) and the rest. This condensed profile has been found equally reliable and valid as the original LFP having the advantage that it reduces the profile to one single score facilitating a comparison between learners (or in our case MT systems).

Since we are interested in the difference between the original training data and the output of the MT systems, we compute the frequency bands on the original training data instead of based on pre-set word lists used in L2 research. As such, we leave out the third band consisting of a list of academic words. Presenting and computing the LFP this way, will give us immediately the 'Beyond 2000' metric score as well (as we distinguish between three bands only, the last one being anything beyond the first 2000 words).

	FR			ES		
	B1	B2	B3	B1	B2	B3
ORIG	**79.80**	6.59	13.61	**77.80**	6.83	15.36
PB-SMT	81.78	6.48	11.74	79.77	6.86	13.36
LSTM	82.95	6.18	10.88	80.34	6.84	12.81
TRANS	82.01	6.24	11.75	82.35	6.99	10.67

	EN$_{FR}$			EN$_{ES}$		
	B1	B2	B3	B1	B2	B3
ORIG	**80.83**	7.10	12.07	**80.81**	7.11	12.08
PB-SMT	82.06	7.04	10.90	82.25	7.01	10.74
LSTM	83.23	6.93	9.81	83.29	6.93	9.78
TRANS	82.25	7.05	10.70	82.35	6.99	10.67

Table 4: Lexical Frequency Profile (French, Spanish, English (EN$_{FR}$ and EN$_{ES}$) with 3 bands (B1: 0-1000, B2: 1001-2000, B3: 2001-end) for the original data and the output of the MT systems.

The LFP for French, Spanish and English, from the EN→FR, EN→ES, FR→EN (denoted as EN$_{FR}$) and ES→EN (denoted as EN$_{ES}$) data is presented in Table 4. It shows that the original data is consistently more diverse than the output of the MT systems as (i) the percentage of text occupied by the 1000 most frequent words (B1) is lower than in the corresponding B1 scores for all MT systems which implies that the 1000 most frequent words take up a smaller percentage of the text in the original training data compared to in the output of the different MT systems; and (ii) the so-called 'Beyond 2000' measure, which in our LFP is equal to the third band (B3), showing us the percentage of text occupied by the words that do not belong to the first two bands, is consistently higher for the original data compared to the MT systems (meaning that the less frequent words occupy a bigger proportion of the original data than they do in its machine translationese variants). Note that it has been established that LFPs are large-grained so small gains in vocabulary are likely to be obscured (Kyle, 2019). The results indicate a consistent and clear difference between the original data and the different types of machine translationese for all language pairs.

Aside from the different LFP scores between the training data and the translations, we also see a difference between the languages themselves. French and Spanish have more variety (higher B1 and lower B3 (Beyond 2000) values) compared to EN$_{ES}$ and EN$_{FR}$. Since the LFPs are computed on tokens, this reflects the richer morphology in French and Spanish compared to English.

TTR, Yule's I and MTLD For completeness, we also present three more commonly used measures of lexical diversity: type/token ratio (TTR) (Templin, 1975), Yule's K (in practice, we use the reverse Yule's I) (Yule, 1944), and the measure of textual lexical diversity (MTLD) (McCarthy, 2005).

TTR presents the ratio of the total number of *different* words (types) to the *total* number of words (tokens). Higher TTR indicates a higher degree of lexical diversity. Yule's characteristic constant (Yule's K) (Yule, 1944) measures constancy of text as the repetitiveness of vocabulary. Yule's K and its inverse Yule's I are considered to be more resilient to fluctuations related to text length than TTR (Oakes and Ji, 2013). The third lexical diversity metric is MTLD. MTLD is evaluated sequentially as the mean length of sequential word strings in a text that maintains a given TTR value (McCarthy, 2005).[7] We present the scores for TTR, Yule's I and MTLD for our data and MT engines in Table 5.

	FR			ES		
	TTR	Yule's I	MTLD	TTR	Yule's I	MTLD
ORIG	**3.02**	**9.28**	**119.40**	**4.08**	**12.31**	**96.23**
PB-SMT	1.79	3.00	112.00	2.37	4.02	92.01
LSTM	1.56	2.14	104.89	2.03	2.95	86.57
TRANS	2.07	3.82	115.66	2.89	6.23	95.72

	EN$_{FR}$			EN$_{ES}$		
	TTR	Yule's I	MTLD	TTR	Yule's I	MTLD
ORIG	**2.89**	**6.64**	**108.70**	**2.88**	**6.61**	**108.46**
PB-SMT	1.74	2.07	94.65	1.82	2.25	93.18
LSTM	1.50	1.53	86.93	1.44	1.42	87.91
TRANS	2.04	3.10	101.95	2.09	3.26	99.62

Table 5: TTR, Yule's I and MTLD scores. For all metrics, higher scores indicate higher lexical richness. For ease of readability and comparison we multiplied TTR scores by 1,000 and Yule's I scores by 10,000.

The scores in Table 5 show that, overall, and according to all three metrics, the original training data has a higher lexical diversity than the machine translationese.

The data for the morphologically richer languages (FR, ES) as well as its machine translationese variants (PB-SMT, LSTM and TRANS) have higher lexical richness than the morphologically poor(er) language (EN).

Synonym Frequency Analysis The objective of synonym frequency analysis is to understand, for words with multiple possible translations, with what frequency the various translations for a given word appear in the translated text. It is called *synonym* frequency in reference to the fact that when translating from one language to another, it is common for a word in the source language to have a corresponding word in the target language to which the source word is typically translated, and that primary translated word can have many *synonyms* that constitute acceptable alternative translation options. Note that we perform this analysis only in one direction: from English into the morphologically richer languages French and Spanish.

	FR			ES		
	PTF↓	CDU↓	SynTTR↑	PTF↓	CDU↓	SynTTR↑
ORIG	**9.666**	**2.725**	**15.10**	**9.131**	**4.539**	**21.13**
PB-SMT	9.715	2.957	11.87	9.236	4.637	17.4
LSTM	9.748	3.154	10.96	9.32	4.782	15.34
TRANS	9.717	3.077	12.25	9.285	4.687	17.15

Table 6: Synonym frequency metrics for our MT systems: primary translation frequency (PTF), cosine distance from uniform (CDU) and TTR modified to only consider words with multiple translation options (SynTTR). The SynTTR scores were multiplied by 100,00 for easier viewing. Higher SynTTR scores indicate greater diversity, while lower PTF and CDU scores indicate greater diversity.

To examine synonym frequency, we first lemmatize the text using SpaCy.[8] Next, we map all nouns, verbs, and adjectives in the source to their possible translation options retrieved from bilingual dictionaries.[9] We then count the number of appearances of these different translation options for the ORIG as well as the MT data. For example, for the English word "look" with translation options in Spanish {"mirar", "esperar", "buscar", "parecer", "dar", "vistazo", "aspecto", "ojeada", "mirada"}, the number of appearances in the TRANS data are as follows: {("mirar": 4002), ("esperar": 3302), ("buscar": 2814), ("parecer": 1144), ("dar": 977), ("vistazo": 182), ("aspecto": 46), ("ojeada": 0), ("mirada": 0)}. From this mapping of translation option to number of appearances we take a vector consisting only of the numbers of appearances for each translation option, and refer to this as a translated word distribution. That is, for the aforementioned example, the distribution vector is: {4002, 3302, 2014, 1144, 997, 182, 46, 0, 0}. We use these counts and distributions as described below.

Our first synonym frequency metric deals di-

---

[7] In our experiments we used 0.72 as a TTR threshold.

[8] https://spacy.io/

[9] English-Spanish: https://github.com/mananoreboton/en-es-en-Dic, English-French: https://freedict.org/downloads

rectly with the primary translation frequency (PTF), where the "primary translation" is the translation for a given source word that appears in the target text most often. We argue that selecting secondary translation options for each source word less frequently, and selecting the primary option more frequently, indicates a decrease of lexical diversity. We measure the PTF by taking the average *primary translation prevalence* over all source words for each MT system.

As a second metric we used the cosine distance between a uniform translated word distribution, where each translation option would be equally prevalent, and the actual translation distributions (we denote this metric as CDU). While the ideal distribution of translations for a given word is almost certainly non-uniform (and therefore not perfectly diverse), this metric still gives valuable information about the tendencies of different systems to favor certain translation options over others.

The third metric is a modified TTR which we refer to as Synonym TTR (or SynTTR). Unlike with regular type/token ratio, rather than considering all tokens that appear in the text, we consider as types only translation options from the source-target mappings described above and as tokens we consider only appearances of valid types. This metric exposes where translation systems completely drop viable translation options from their vocabulary.

Table 6 shows the results for these 3 metrics. Interestingly, the MT systems can be ranked in the same order according to all these metrics: PB-SMT > TRANS > LSTM, where > denotes the comparison of lexical diversity (higher to lower). However, across the 3 metrics, for both language pairs, the reference translations (ORIG) appear to be the most lexically diverse in terms of synonym frequency, with the lowest PTF and CDU and highest SynTTR. This reinforces the idea that MT algorithms have a negative impact on the diversity of language.

4.3 Grammatical Diversity

Grammatical diversity manifests itself on the sentence (syntactic complexity) and word level (morphological complexity). With our experiments, we focus on the morphological complexity by averaging the inflectional diversity of all lemmas. To do so, we adopted two measures, originating from Information Theory: Shannon entropy (Shannon, 1948) and Simpson's Diversity Index (Simpson, 1949). The former emphasizes on the richness aspect of diversity while the latter on the evenness aspect of diversity. We used the Spacy-udpipe lemmatizer to retrieve all lemmas.[10]

Shannon Entropy Shannon entropy (H) measures the level of uncertainty associated with a random variable (X). It has been applied in use-cases from economy, ecology, biology, complex systems, language and many others (Page, 2007, 2011). In the study of language Shannon entropy has previously been used for estimating the entropy of language models (Behr et al., 2003). We use it to measure the entropy of wordforms given a lemma. In particular, the entropy of inflectional paradigm of a specific lemma could be computed by taking the base frequency of that lemma (frequency of all wordforms associated with that lemma) and the probabilities of all the wordforms within the inflectional paradigm of that particular lemma. Using such a formulation of entropy allows us to measure the morphological variety (or the loss thereof) for the machine translationese produced by each system – higher values of H indicate higher diversity and vice-versa. We use Equation 1 to compute the entropy of the inflectional paradigm of a lemma.

$$H(\ell) = -\sum_{wf \in \ell} p(wf|\ell) \log p(wf|\ell)) \qquad (1)$$

$H(\ell)$ denotes the entropy of the lemma ℓ and, for the wordform wf, $p(wf|\ell)$ is computed as the fraction of the counts of the wordform, $count(wf)$, to the count of all wordforms for the lemma ℓ, i.e. $p(wf|\ell) = \frac{count(wf)}{\sum_{wf^* \in \ell} count(wf^*)}$. We use $\in$ to indicate wordforms of a given lemma.

Simpson's Diversity Index Like Shannon Entropy, Simpson's Diversity Index (D) is a measure used to determine variation in categorical data. Values close to 1 indicate higher homogeneity, thus lower diversity and values close to 0 indicate higher variability, thus higher diversity.

Following the same reasoning as with Shannon entropy, we compute Simpson's diversity index for each lemma and the corresponding wordforms according to the formula in Equation 2.

$$D(\ell) = \frac{1}{\sum_{wf \in \ell} p(wf|\ell)^2} \qquad (2)$$

---

[10] `https://github.com/TakeLab/spacy-udpipe`

We average the Shannon entropy and Simpson's diversity index for all lemmas to get an indicative score for each translation system or the original text. We denote these with ***H*** and ***D***, accordingly. To the best of our knowledge, our work is the first to use Shannon entropy and Simpson's diversity index for the study of inflectional richness on a text level. The closest to our application of these two diversity metrics for measuring inflectional richness is the work by del Prado Martín et al. (2004). Their work on morphological processing uses Shannon entropy to compute the amount of information carried by a morphological paradigm.

An illustration of the Shannon entropy and Simpson's diversity index of a lemma is given in Table 7. We list the number of occurrences for every wordform (male singular, male plural, female singular and female plural) appearing in our datasets for the French lemma 'président' (EN: president). We then compute H and D by applying Equation 1 and Equation 2 accordingly. While both Shannon H and Simpson's D scores usually range between 0–1, for ease of readability we multiply the scores presented in Table 7 and Table 8 by 100 to present them in the range of $[0 - 100]$.

	lemma: président				H↑	D↓
	président	présidents	présidente	présidentes		
ORIG	93774	2029	1490	8	**18.11**	**92.95**
PB-SMT	99367	2019	496	1	12.81	95.16
LSTM	95272	2039	291	N/A	12.17	95.3
TRANS	92946	1952	617	N/A	13.86	94.74

Table 7: An illustration of the Shannon entropy and the Simpson's diversity index computed for the occurrences of the different wordforms of the French lemma for 'president' (président).

For lemmas with a single wordform Shannon entropy and Simpson's diversity index will be $H = 0.0$ and $D = 1.0$, respectively. While this makes sense when measuring the diversity of one morphological paradigm, they actually impact the average scores ***H*** and ***D*** without contributing to the understanding of diversity in a comparative study such as ours. In particular, lemmas with single wordforms may be either an evidence of low diversity, e.g. a translation system will always generate only one form or of high diversity, e.g. rare words that are single wordform for a particular lemma (such as synonyms of more common words) can indicate the ability of a system to generate more diverse language (in terms of synonymy). That is why we computed ***H*** and ***D*** on lemmas with two or more wordforms. For completeness, we also present the number of single wordform lemmas. The Shannon entropy and Simpson's diversity index for French, Spanish and English for all datasets are presented in Table 8. The scores are shown in the range $[0 - 100]$ as noted above.

	FR			ES		
	H↑	***D***↓	Single	***H***↑	***D***↓	Single
ORIG	**75.20**	**56.42**	79k	**78.42**	**54.96**	92k
PB-SMT	69.00	59.64	51k	71.79	58.56	54k
LSTM	69.28	59.48	53k	72.84	58.29	55k
TRANS	73.13	57.70	58k	77.23	56.26	64k

	EN_{FR}			EN_{ES}		
	H↑	***D***↓	Single	***H***↑	***D***↓	Single
ORIG	**59.04**	**63.43**	78k	**59.05**	**63.42**	78k
PB-SMT	55.57	65.80	51k	56.31	65.29	61k
LSTM	53.15	67.02	50k	53.85	66.64	48k
TRANS	55.85	65.43	58k	56.22	65.19	68k

Table 8: Shannon entropy (***H***) for French, Spanish, English (EN_{FR} and EN_{ES}) and Simpson's diversity index (***D***) for original training data and the output of the PB-SMT, LSTM and TRANS systems. Scores are multiplied by 100 for ease of readability.

The ***H*** and ***D*** scores in Table 8 are an evidence of the negative impact of MT on the morphological diversity – the scores for the ORIG indicate a consistent higher diversity. Comparing the MT systems, it results that TRANS retains morphological diversity better than the others. LSTM performs better than PB-SMT for translations into the morphologically richer languages (FR and ES) but PB-SMT seems much better than LSTM for translations into English. While the loss of lexical diversity could, in some cases be a desirable side-effect of MT systems (in terms of simplification or consistency), the uncontrolled loss of morphological richness is problematic as it can prevent systems from picking the grammatically correct option.

5 Conclusions

In this work, we explore the effects of MT algorithms on the richness and complexity of language. We establish that there is indeed a quantitatively measurable difference between the linguistic richness of MT systems' training data and their output – a product of algorithmic bias. These findings are in line with previous results described in Vanmassenhove et al. (2019). Assessing diversity or richness in language is a multifacted task spanning over various domains. As such, we approach this task from multiple angles focusing on lexical diversity and sophistication, morphological variety and a more translation specific metric focusing on synonymy. To do so, we analyse the results of 9 different metrics including established, newly proposed and adapted ones. The metrics suit we developed is unprecedented in the study of MT quality and we believe it could drive future research on MT evaluation.

Based on a wide range of experiments with 3 different MT architectures, we draw the following main conclusions: (i) all 9 metrics indicate that the original training data has more lexical and morphological diversity compared to translations produced by the MT systems. This is the case for all language pairs and directions; (ii) Comparing the MT systems among themselves, there is a strong indication (for most metrics) that Transformer models outperform the others in terms of lexical and morphological richness. We also ought to note that, on average, the ranking of the systems in terms of diversity metrics correlates with the quality of the translations (in terms of BLEU and TER). This is something that would need to be further explored in future work; (iii) The data for the morphologically richer languages (ES, FR) has higher lexical and (evidently) morphological diversity than the English data both in the original data and in the translations generated by all systems. However, for PB-SMT, LSTM and TRANS the difference in scores is much smaller than the ORIG, indicating that the MT systems have a stronger negative impact (in terms of diversity and richness) on the morphologically richer languages.

Acknowledgement

We would like to thank the reviewers for their insightful comments and feedback.

References

Roee Aharoni, Moshe Koppel, and Yoav Goldberg. 2014. Automatic detection of machine translated text and translation quality estimation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL2014), Volume 2: Short Papers*, pages 289–295.

Nora Aranberri. 2020. Can translationese features help users select an MT system for post-editing? *Procesamiento del Lenguaje Natural*, 64:93–100.

Lyle F Bachman. 2004. *Statistical analyses for language assessment*. Cambridge University Press.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, CA, USA. 15pp.

Mona Baker. 1999. The role of corpora in investigating the linguistic behaviour of professional translators. *International journal of corpus linguistics*, 4(2):281–298.

Marco Baroni and Silvia Bernardini. 2006. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274.

Frederic H. Behr, Victoria Fossum, Michael Mitzenmacher, and David Xiao. 2003. Estimating and comparing entropies across written natural languages using PPM compression. In *Proceedings of the 2003 Data Compression Conference (DCC 2003)*, page 416, Snowbird, UT, USA.

Yuri Bizzoni, Tom S Juzek, Cristina España-Bonet, Koel Dutta Chowdhury, Josef van Genabith, and Elke Teich. 2020. How human is machine translationese? comparing human and machine translations of text and speech. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT2020)*, pages 280–290, Online.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*, pages 4349–4357.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive Exploration of Neural Machine Translation Architectures. In *Proceedings of the Association for Computational Linguistics (ACL2017)*, pages 1442–1451, Vancouver, Canada.

Bram Bulté. 2013. The development of complexity in second language acquisition. *A dynamic systems approach (Unpublished doctoral dissertation)*.

Bram Bulté, Alex Housen, Michel Pierrard, and Siska Van Daele. 2008. Investigating lexical proficiency development over time–the case of dutch-speaking learners of french in brussels. *Journal of French Language Studies*, 18(3):277–298.

Aylin Caliskan-Islam, Joanna J Bryson, and Arvind Narayanan. 2016. Semantics derived automatically from language corpora necessarily contain human biases. *arXiv preprint arXiv:1608.07187*, pages 1–14.

Jonathan H Clark, Chris Dyer, Alon Lavie, and Noah A Smith. 2011. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL2011): Human Language Technologies, Volume 2: Short Papers*, pages 176–181, Portland, Oregon, USA.

Scott A Crossley, Tom Cobb, and Danielle S McNamara. 2013. Comparing count-based and band-based indices of word frequency: Implications for active vocabulary research and pedagogical applications. *System*, 41(4):965–981.

Joke Daems, Orphée De Clercq, and Lieve Macken. 2017. Translationese and post-editese: How comparable is comparable quality? *Linguistica Antverpiensia New Series-Themes in Translation Studies*, 16:89–103.

Lisa Fan, Marshall White, Eva Sharma, Ruisi Su, Prafulla Kumar Choubey, Ruihong Huang, and Lu Wang. 2019. In plain sight: Media bias through the lens of factual reporting. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP2019)*, pages 6343–6349.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Martin Gellerstam. 1986. Translationese in swedish novels translated from english. *Translation studies in Scandinavia*, 1:88–95.

Nizar Habash, Houda Bouamor, and Christine Chung. 2019. Automatic gender identification and reinflection in arabic. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 155–165.

Margherita Ippolito. 2014. *Simplification, Explicitation and Normalization: Corpus-Based Research into English to Italian Translations of Children's Classics*. Cambridge Scholars Publishing.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations: Poster Session*, Banff, Canada.

Beata Beigman Klebanov and Michael Flor. 2013. Associative texture is lost in translation. In *Proceedings of the Workshop on Discourse in Machine Translation*, pages 27–32, Sofia, Bulgaria.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP2004)*, pages 388–395, Barcelona, Spain.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of The Tenth Machine Translation Summit (MT Summit 2005)*, pages 79–86, Phuket, Thailand.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open-Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL2007)*, pages 177–180, Prague, Czech Republic.

Moshe Koppel and Noam Ordan. 2011. Translationese and its dialects. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL2011), Volume 1: Long Papers*, pages 1318–1326.

Svenja Kranich. 2014. Translations as a locus of language contact. In *Translation: A multidisciplinary approach*, pages 96–115. Springer.

Kristopher Kyle. 2019. Measuring lexical richness. *The Routledge Handbook of Vocabulary Studies*, page 454.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP2018)*, pages 4791–4796, Brussels, Belgium.

Batia Laufer. 1994. The lexical profile of second language writing: Does it change over time? *RELC journal*, 25(2):21–33.

Batia Laufer and Paul Nation. 1995. Vocabulary size and use: Lexical richness in l2 written production. *Applied linguistics*, 16(3):307–322.

Philip M McCarthy. 2005. An Assessment of the Range and Usefulness of Lexical Diversity Measures and the Potential of the Measure of Textual, Lexical Diversity (MTLD). In *PhD Thesis, Dissertation Abstracts International, Volume 66:12*. University of Memphis, Memphis, Tennessee, USA.

Jack Merullo, Luke Yeh, Abram Handler, Alvin Grissom II, Brendan O'Connor, and Mohit Iyyer. 2019. Investigating sports commentator bias within a large corpus of American football broadcasts. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 6355–6361, Hong Kong, China.

IS Paul Nation. 1984. *Vocabulary lists: Words, affixes, and stems*. English Language Institute, Victoria University of Wellington.

Michael P Oakes and Meng (eds) Ji. 2013. Quantitative Methods in Corpus-Based Translation Studies: A Practical Guide to Descriptive Translation Research. In *Studies in Corpus Linguistics, Volume 51*, page 361. John Benjamins Publishing Company, Amsterdam, The Netherlands.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics, Volume 29:1*, pages 19–51. MIT Press, Cambridge, Massachusetts, USA.

Scott E. Page. 2007. *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies*. Princeton University Press.

Scott E. Page. 2011. *Diversity and Complexity*. Princeton University Press.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 311–318, Philadephia, PA, USA.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation (WMT 2018): Research Papers*, pages 186–191, Belgium, Brussels.

Fermın Moscoso del Prado Martın, Aleksandar Kostić, and R Harald Baayen. 2004. Putting the bits together: An information theoretical perspective on morphological processing. *Cognition*, 94(1):1–18.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), Volume 1: Long Papers*, pages 1715–1725, Berlin, Germany.

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Edward H Simpson. 1949. Measurement of diversity. *Nature*, 163(4148):688–688.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA 2006) 200:6*, pages 223–231, Austin, Texas, USA.

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684, Florence, Italy.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 1630–1640, Florence, Italy.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems (NIPS 2014)*, pages 3104–3112, Montreal, Quebec, Canada.

Mildred C. Templin. 1975. *Certain Language Skills in Children: Their Development and Interrelationships*. Greenwood Press, Westport, Connecticut, USA.

Antonio Toral. 2019. Post-editese: an exacerbated translationese. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 273–281, Dublin, Ireland. European Association for Machine Translation.

Antonio Toral, Sheila Castilho, Ke Hu, and Andy Way. 2018. Attaining the unattainable? reassessing claims of human parity in neural machine translation. In *Proceedings of the Third Conference on Machine Translation (WMT 2018): Research Papers*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.

Eva Vanmassenhove. 2020. On the Integration of Linguistic Features into Statistical and Neural Machine Translation. *arXiv preprint arXiv:2003.14324*.

Eva Vanmassenhove and Christian Hardmeier. 2018. Europarl datasets with demographic speaker information.

Eva Vanmassenhove, Christian Hardmeier, and Andy Way. 2018. Getting gender right in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2019)*, pages 3003–3008, Brussels, Belgium.

Eva Vanmassenhove, Dimitar Shterionov, and Andy Way. 2019. Lost in translation: Loss and decay of linguistic richness in machine translation. In *Proceedings of Machine Translation Summit XVII (MT Summit 2019), Volume 1: Research Track*, pages 222–232, Dublin, Ireland.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of The Thirty-first Annual Conference on Neural Information Processing Systems 30 (NIPS 2017)*, pages 5998–6008, Long Beach, CA, USA.

G. Udnv Yule. 1944. *The Statistical Study of Literary Vocabulary*. Cambridge University Press.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017), Volume 1: Long Papers*, pages 654–664, Vancouver, Canada.

First Align, then Predict: Understanding the Cross-Lingual Ability of Multilingual BERT

Benjamin Muller[1,2] **Yanai Elazar**[3,4] **Benoît Sagot**[1] **Djamé Seddah**[1]
[1]Inria, Paris, France [2]Sorbonne Université, Paris, France
[3]Computer Science Department, Bar Ilan University
[4]Allen Institute for Artificial Intelligence
{benjamin.muller, benoit.sagot,djame.seddah}@inria.fr
yanaiela@gmail.com

Abstract

Multilingual pretrained language models have demonstrated remarkable zero-shot cross-lingual transfer capabilities. Such transfer emerges by fine-tuning on a task of interest in one language and evaluating on a distinct language, not seen during the fine-tuning. Despite promising results, we still lack a proper understanding of the source of this transfer. Using a novel layer ablation technique and analyses of the model's internal representations, we show that multilingual BERT, a popular multilingual language model, can be viewed as the stacking of two sub-networks: a multilingual encoder followed by a task-specific language-agnostic predictor. While the encoder is crucial for cross-lingual transfer and remains mostly unchanged during fine-tuning, the task predictor has little importance on the transfer and can be reinitialized during fine-tuning. We present extensive experiments with three distinct tasks, seventeen typologically diverse languages and multiple domains to support our hypothesis.

1 Introduction

Zero-shot Cross-Lingual transfer aims at building models for a *target* language by reusing knowledge acquired from a *source* language. Historically, it has been tackled with a two-step *standard cross-lingual pipeline* (Ruder et al., 2019): (1) Building a shared multilingual representation of text, typically by aligning textual representations across languages. This step can be done using feature extraction (Aone and McKee, 1993; Schultz and Waibel, 2001) as with the delexicalized approach (Zeman and Resnik, 2008; Søgaard, 2011) or using word embedding techniques (Mikolov et al., 2013; Smith et al., 2017) by projecting monolingual embeddings onto a shared multilingual embedding space, this step requiring explicit supervision signal in the target language in the form of features or parallel data. (2) Training a task-specific model using supervision on a source language on top of the shared representation.

Recently, the rise of multilingual language models entailed a paradigm shift in this field. Multilingual pretrained language models (Devlin et al., 2019; Conneau and Lample, 2019) have been shown to perform efficient zero-shot cross-lingual transfer for many tasks and languages (Pires et al., 2019; Wu and Dredze, 2019). Such transfer relies on three-steps: (i) pretraining a mask-language model (e.g. Devlin et al. (2019)) on the concatenation of monolingual corpora across multiple languages, (ii) fine-tuning the model on a specific task in the source language, and (iii) using the fine-tuned model on a target language. The success of this approach is remarkable, and in contrast to the standard cross-lingual pipeline, the model sees neither aligned data nor task-specific annotated data in the target language at any training stage.

The source of such a successful transfer is still largely unexplained. Pires et al. (2019) hypothesize that these models learn shared multilingual representations during pretraining. Focusing on syntax, Chi et al. (2020) recently showed that the multilingual version of BERT (mBERT) (Devlin et al., 2019), encodes linguistic properties in shared multilingual sub-spaces. Recently, Gonen et al. (2020) suggest that mBERT learns a language encoding component and an abstract cross-lingual component. In this work, we are interested in understanding the mechanism that leads mBERT to perform zero-shot cross-lingual transfer. More specifically, we ask **what parts of the model and what mechanisms support cross-lingual transfer?**

By combining behavioral and structural analyses (Belinkov et al., 2020), we show that mBERT operates as the stacking of two modules: (1) A multilingual encoder, located in the lower part of the model, critical for cross-lingual transfer, is in charge of

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2214–2231
April 19 - 23, 2021.

aligning multilingual representations; and (2) a task-specific, language-agnostic predictor which has little importance for cross-lingual transfer and is dedicated to performing the downstream task. This mechanism that emerges out-of-the-box, without any explicit supervision, suggests that mBERT behaves like the standard cross-lingual pipeline. Our contributions advance the understanding of multilingual language models and as such have the potential to support the development of better pretraining processes.

2 Analysis Techniques

We study mBERT with a novel behavioral test that disentangles the task fine-tuning influence from the pretraining step (§2.1), and a structural analysis on the intermediate representations (§2.2). Combining the results from these analyses allows us to locate the cross-lingual transfer and gain insights into the mechanisms that enable it.

2.1 Locating Transfer with RANDOM-INIT

In order to disentangle the impact of the pretraining step from the fine-tuning, we propose a new behavioral technique: RANDOM-INIT. First, we randomly initialize a set of parameters (e.g. all the parameters of a given layer) instead of using the parameters learned during the pretraining step. Then, we fine-tune the modified pretrained model and measure the downstream performance.[1]

By replacing a given set of pretrained parameters and fine-tuning the model, *all other factors being equal*, RANDOM-INIT enables us to quantify the contribution of a given set of pretrained parameters on downstream performance and therefore to locate which pretrained parameters contribute to the cross-lingual transfer.

If the cross-language performance is significantly lower than same-language performance, we conclude that these layers are more important to cross-language performance than they are for same-language performance. If the cross-language score does not change, it indicates that cross-language transfer does not rely on these layers.

This technique is reminiscent of the recent *Amnesic Probing* method (Elazar et al., 2020), that removes from the representation a specific feature, e.g. Part-of-Speech, and then measures the outcome on the downstream task. In contrast, RANDOM-INIT allows to study a specific architecture component, instead of specific features.

2.2 Hidden State Similarities across Languages

To strengthen the behavioral evidence brought by RANDOM-INIT, and provide finer analyses that focus on individual layers, we study how the textual representations differ between parallel sentences in different languages. We hypothesize that an efficient fine-tuned model should be able to represent similar sentences in the source and target languages similarly, even-though it was fine-tuned only on the source language.

To measure the similarities of the representation across languages, we use the Central Kernel Alignment metric (CKA), introduced by Kornblith et al. (2019). We follow Conneau et al. (2020) who use the CKA as a similarity metric to compare the representations of monolingual and bilingual pretrained models across languages. In our work, we use the CKA to study the representation difference between source and target languages in pretrained and fine-tuned multilingual models. For every layer, we average all contextualized tokens in a sentence to get a single vector.[2] Then we compute the similarity between target and source representations and compare it across layers in the pretrained and fine-tuned models. We call this metric the *cross-lingual similarity*.

3 Experimental Setup

Tasks, Datasets and Evaluation We consider three tasks covering both syntactic and semantic aspects of language: Part-Of-Speech Tagging (POS), dependency parsing, and Named-Entity Recognition (NER). For POS tagging and parsing we use the Universal Dependency (Nivre et al., 2018) treebanks, and for NER, we use the WikiANN dataset (Pan et al., 2017). We evaluate our systems with the standard metrics per task; word-level accuracy for POS tagging, F1 for NER and labeled attachment score (LAS) for parsing. All the reported scores are computed on the test set of each dataset.

We experiment with English, Russian and Arabic as source languages, and fourteen typologically diverse target languages, including Chinese, Czech, German and Hindi. The complete list can be found in the Appendix A.1.2.

---

[1] Note that we perform the same optimization procedure for the model with and w/o RANDOM-INIT (optimal learning rate and batch size are chosen with grid-search).

[2] After removing [CLS] and [SEP] special tokens.

Src-Trg	Ref	Random-init *of layers* Δ1-2	Δ3-4	Δ5-6	Δ7-8	Δ9-10	Δ11-12
				Parsing			
En - En	88.98	-0.96	-0.66	-0.93	-0.55	0.04	-0.09
Ru - Ru	85.15	-0.82	-1.38	-1.51	-0.86	-0.29	0.18
Ar - Ar	59.54	-0.78	-2.14	-1.20	-0.67	-0.27	0.08
En - X	53.23	-15.77	-6.51	-3.39	-1.47	0.29	1.00
Ru - X	55.41	-7.69	-3.71	-3.13	-1.70	0.92	0.94
Ar - X	27.97	-4.91	-3.17	-1.48	-1.68	-0.36	-0.14
				POS			
En - En	96.51	-0.30	-0.25	-0.40	-0.00	0.05	0.02
Ru - Ru	96.90	-0.52	-0.55	-0.40	-0.07	0.02	-0.03
Ar - Ar	79.28	-0.35	-0.49	-0.36	-0.19	-0.05	-0.00
En - X	79.37	-8.94	-2.49	-1.66	-0.88	0.20	-0.14
Ru - X	79.25	-10.08	-2.83	-1.65	-2.74	0.01	-0.45
Ar - X	64.81	-6.73	-3.50	-1.63	-1.56	-0.73	-1.29
				NER			
En - En	83.30	-2.66	-2.14	-1.43	-0.63	-0.23	-0.12
Ru - Ru	88.20	-2.08	-2.13	-1.52	-0.64	-0.33	-0.13
Ar - Ar	87.97	-2.37	-2.11	-0.96	-0.39	-0.15	0.21
En - X	64.17	-8.28	-5.09	-3.07	-0.79	-0.47	-0.13
Ru - X	62.13	-15.85	-9.36	-5.50	-2.44	-1.16	-0.06
Ar - X	65.59	-16.10	-8.42	-3.73	-1.40	-0.25	0.67

Table 1: Relative Zero shot Cross-Lingual performance of mBERT with Random-init (§2.1) on pairs of consecutive layers compared to mBERT without any random-initialization (Ref). In Src-Trg, Src indicates the source language on which we fine-tune mBERT, and Trg the target language on which we evaluate it. Src-X is the average across all 17 target language with X $\neq$ Src. Detailed results per target language are reported in tables 6, 7 and 8 in the Appendix. Coloring is computed based on how mBERT with Random-init performs compared to the Ref model. $\geq$ Ref $<$ Ref $\leq$ -2 points $\leq$ -5 points

The results of a model that is fine-tuned and evaluated on the *same* language are referred to as *same-language* and those evaluated on *distinct* languages are referred to as *cross-language*.

Multilingual Model We focus on mBERT (Devlin et al., 2019), a 12-layer model trained on the concatenation of 104 monolingual Wikipedia corpora, including our languages of study.

Fine-Tuning We fine-tune the model for each task following the standard methodology of Devlin et al. (2019). The exact details for reproducing our results can be found in the Appendix. All reported scores are averaged on 5 runs with different seeds.

4 Results

4.1 Disentangling the Pretraining Effect

For each experiment, we measure the impact of randomly-initializing specific layers as the difference between the model performance without any random-initialization (Ref) and with random-initialization (Random-init). Results for two consecutive layers are shown in Table 1. The rest of the results, which exhibit similar trends, can be found in the Appendix (Table 5).

For all tasks, we observe sharp drops in the cross-language performance at the lower layers of the model but only moderate drops in the same-language performance. For instance, the parsing experiment with English as the source language, results in a performance drop on English of only 0.96 points (En-En), when randomly-initializing layers 1 and 2. However, it leads to an average drop of 15.77 points on other languages (En-X).

Furthermore, we show that applying Random-init to the upper layers does not harm same-language and cross-language performances (e.g. when training on parsing for English, the performance slightly decreases by 0.09 points in the same-language while it increases by 1.00 in the cross-language case). This suggests that the upper layers are *task-specific* and *language-agnostic*, since re-initializing them have minimal change on performance. We conclude that mBERT's upper layers do not contribute to cross-language transfer.

Does the Target Domain Matter? In order to test whether this behavior is specific to the cross-language setting and is not general to out-of-distribution (OOD) transfer, we repeat the same Random-init experiment by evaluating on same-language setting while varying the evaluated domain.[3] If the drop is similar to cross-language performance, it means that lower layers are important for out-of-distribution transfer in general. Otherwise, it would confirm that these layers play a specific role for cross-language transfer.

We report the results in Table 2. For all analyzed domains (Web, News, Literature, etc.) applying Random-init to the two first layers of the models leads to very moderate drops (e.g. -0.91 when the target domain is English Literature for parsing), while it leads to large drops when the evaluation is done on a distinct language (e.g. -5.82 when evaluated on French). The trends are similar for all the domains and tasks we tested on. We conclude that the pretrained parameters at the lower layers are consistently more critical for cross-language transfer than for same-language transfer, and cannot be explained by the possibly different domain of the evaluated datasets.

---

[3] Although other factors might play a part in out-of-distribution, we suspect that domains plays a crucial part in transfer. Moreover, it was shown that BERT encodes out-of-the-box domain information (Aharoni and Goldberg, 2020)

		RANDOM-INIT *of layers*					
SRC - TRG	REF	Δ0-1	Δ2-3	Δ4-5	Δ6-7	Δ8-9	Δ10-11
Domain Analyses					*Parsing*		
EN - EN	90.40	-1.41	-2.33	-1.57	-1.43	-0.60	-0.46
EN - EN LIT.	77.91	-0.91	-1.38	-1.85	-0.83	-0.23	-0.17
EN - EN WEB	75.77	-2.14	-2.42	-2.54	-1.42	-0.71	-0.69
EN - EN UGC	45.90	-1.97	-2.75	-2.10	-1.04	-0.39	-0.25
Cross-Language							
EN - FR TRAN.	83.25	-5.82	-2.69	-2.42	-0.44	0.25	0.94
EN - FR WIKI	71.29	-7.86	-4.33	-4.64	-0.92	-0.11	0.33
Domain Analyses					*POS*		
EN - EN	96.83	-1.35	-0.98	-0.70	-0.40	-0.28	-0.24
EN - EN LIT.	93.09	-0.58	-0.65	-0.28	-0.04	-0.06	0.12
EN - EN WEB	89.67	-1.07	-1.21	-0.41	-0.10	0.03	0.21
EN - EN UGC	68.93	-2.38	-1.07	-0.14	0.54	-0.04	0.63
Cross-Language							
EN - FR TRAN.	93.43	-3.59	-0.88	-1.31	-0.56	0.46	0.25
EN - FR.	91.13	-5.10	-0.93	-1.16	-0.74	0.15	-0.07
Domain Analyses					*NER*		
EN - EN	83.22	-2.45	-2.15	-1.28	-0.49	-0.15	-0.06
EN - NEWS	51.72	-1.32	-1.05	-0.80	-0.14	-0.31	-0.33
Cross-Language							
EN - FR	76.16	-5.14	-2.82	-1.97	-0.33	0.52	0.34

Table 2: Relative Zero shot Cross-Lingual performance of mBERT with RANDOM-INIT (§2.1) on pairs of consecutive layers compared to mBERT without any random-initialization (REF). We present experiments with English as the source language and evaluate across various target domains in English in comparison with the cross-lingual setting when we evaluate on French. EN-LIT. refers to the Literature Domain. UGC refers to User-Generated Content. FR-TRAN. refers to sentences translated from the English *In-Domain* test set, hence reducing the domain-gap to its minimum.

≥ REF < REF ≤ -2 points ≤ -5 points

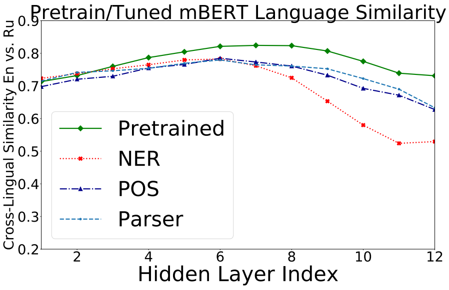


Figure 1: Cross-Lingual similarity (CKA) between representations of pretrained and fine-tuned models on POS, NER and Parsing between English and Russian.

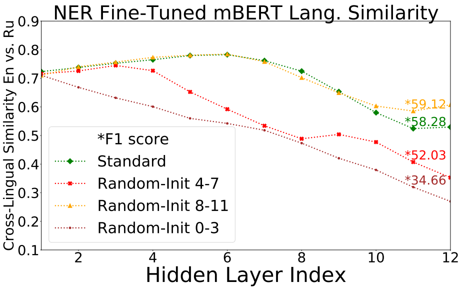


Figure 2: Cross-Lingual similarity (CKA) of the representations of a fine-tuned model on NER with and w/o RANDOM-INIT between English (source) and Russian (target). The higher the score the greater the similarity.

4.2 Cross-Lingual Similarity in mBERT

The results from the previous sections suggest that the lower layers of the model are responsible for the cross lingual transfer, whereas the upper layers are language-agnostic. In this section, we assess the transfer by directly analyzing the intermediate representations and measuring the similarities of the hidden state representations between source and target languages. We compute the CKA metric (cf. §2.2) between the source and the target representations for pretrained and fine-tuned models using parallel sentences from the PUD dataset (Zeman et al., 2017). In Figure 1, we present the similarities between Russian and English with mBERT pretrained and fine-tuned on the three tasks.[4]

The cross-lingual similarity between the representations constantly increases up to layer 5 for all the three tasks (reaching 78.1%, 78.1% and 78.2% for parsing, POS tagging and NER respectively). From these layers forward, the similarity decreases. We observe the same trends across all languages (cf. Figure 5). This demonstrates that the fine-tuned model creates similar representations regardless of the language and task, and hints on an alignment that occurs in the lower part of the model. Interestingly, the same trend is also observed in the pretrained model, suggesting that the fine-tuning step preserves the multilingual alignment.

These results do not match the findings of Singh et al. (2019), who found no language alignment across layers, although they inspected Natural Language Inference, a more "high-level task" (Dagan et al., 2005; Bowman et al., 2015). We leave the inspection of this mismatch to future work.

4.3 Better Alignment Leads to Better Cross-Lingual Transfer

In the previous section we showed that fine-tuned models align the representations between parallel sentences, across languages. Moreover, we demonstrated that the lower part of the model is critical for cross-language transfer but hardly impacts the same-language performance. In this section, we show that the alignment measured plays a critical role in cross-lingual transfer.

As seen in Figure 2 in the case of English to Russian (and in Figures 6-8 in the Appendix for other languages), when we randomly-initialize the lower part of the model, there is no alignment: the

---

[4] We report the comparisons for 5 other languages in Figure 5 in the Appendix.

similarity between the source and target languages decreases. We observe the same trend for all other languages and tasks and report it in the Appendix in Figures 6-8. This result matches the drop in cross-lingual performance that occurs when we apply RANDOM-INIT to the lower part of the model while impacting moderately same-language performance.

For a more systematic view of the link between the cross-lingual similarities and the cross-language transfer, we measure the Spearman correlation between the *cross-lang gap* (i.e the difference between the same-language perfromance and the cross-language performance) (Hu et al., 2020) and the cross-lingual similarity averaged over all the layers. We measure it with the cross-lingual similarity computed on the pretrained and fine-tuned models (without random-initialization) on all the languages. We find that the cross-lingual similarity correlates significantly with the *cross-lang gap* for all three tasks, both on the fine-tuned and pretrained models. The spearman correlation for the fine-tuned models are 0.76, 0.75 and 0.47 for parsing, POS and NER, respectively.[5] In summary, our results show that the cross-lingual alignment is highly correlated with the cross-lingual transfer.

5 Discussion

Understanding the behavior of pretrained language models is currently a fundamental challenge in NLP. A popular approach consists of probing the intermediate representations with external classifiers (Alain and Bengio, 2017; Adi et al., 2017; Conneau et al., 2018) to measure if a specific layer captures a given property. Using this technique, Tenney et al. (2019) showed that BERT encodes linguistic properties in the same order as the "classical NLP pipeline". However, probing techniques only indirectly explain the behavior of a model and do not explain the relationship between the information captured in the representations and its effect on the task (Elazar et al., 2020). Moreover, recent works have questioned the usage of probing as an interpretation tool (Hewitt and Liang, 2019; Ravichander et al., 2020). This motivates our approach to combine a structural analysis based on representation similarity with behavioral analysis. In this regard, our findings extend recent work from Merchant et al. (2020) in the multilingual setting, who show that fine-tuning impacts mainly the upper layers of the model and preserves the linguistic features learned during pretraining. In our case, we show that the lower layers are in charge of aligning representations across languages and that this cross-lingual alignment learned during pretraining is preserved after fine-tuning.

6 Conclusion

The remarkable performance of multilingual languages models in zero-shot cross-lingual transfer is still not well understood. In this work, we combine a *structural* analysis of the similarities between hidden representation across languages with a novel *behavioral* analysis that randomly-initialize the models' parameters to understand it. By combining those experiments on 17 languages and 3 tasks, we show that mBERT is constructed from: (1) a multilingual encoder in the lower layers, which aligns hidden representations across languages and is critical for cross-language transfer, and (2) a task-specific, language-agnostic predictor that has little effect to cross-language transfer, in the upper layers. Additionally, we demonstrate that hidden cross-lingual similarity strongly correlates with downstream cross-lingual performance suggesting that this alignment is at the root of these cross-lingual transfer abilities. This shows that mBERT reproduces the standard cross-lingual pipeline described by Ruder et al. (2019) without any explicit supervision signal for it. Practically speaking, our findings provide a concrete tool to measure cross-lingual representation similarity that could be used to design better multilingual pretraining processes.

Acknowledgments

We want to thank Hila Gonen, Shauli Ravfogel and Ganesh Jawahar for their insightful reviews and comments. We also thank the anonymous reviewers for their valuable suggestions. This work was partly funded by two French National funded projects granted to Inria and other partners by the Agence Nationale de la Recherche, namely projects PARSITI (ANR-16-CE33-0021) and SoSweet (ANR-15-CE38-0011), as well as by the third author's chair in the PRAIRIE institute funded by the French national agency ANR as part of the "Investissements d'avenir" programme under the reference ANR-19-P3IA-0001. Yanai Elazar is grateful to be partially supported by the PBC fellowship for outstanding Phd candidates in Data Science.

---

[5] Correlations for both the pretrained and the fine-tuned models are reported in the Appendix Table 4.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Chinatsu Aone and Douglas McKee. 1993. A language-independent anaphora resolution system for understanding multilingual texts. In *31st Annual Meeting of the Association for Computational Linguistics*, pages 156–163.

Yonatan Belinkov, Sebastian Gehrmann, and Ellie Pavlick. 2020. Interpretability and analysis in neural nlp. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 1–5.

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

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

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

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Y. Goldberg. 2020. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *arXiv: Computation and Language*.

Hila Gonen, Shauli Ravfogel, Yanai Elazar, and Yoav Goldberg. 2020. It's not greek to mbert: Inducing word-level translations from multilingual bert. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 45–56.

Rob van der Goot and Gertjan van Noord. 2018. Modeling input uncertainty in neural network dependency parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4984–4991.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, John Bauer, Sandra Bellato, Kepa Bengoetxea, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Rogier Blokland, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Radu Ion, Elena Irimia, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phương Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shinsuke Mori, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lương Nguyễn Thị, Huyền Nguyễn Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ̀ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roșca, Olga Rudina, Shoval Sadde, Shadi Saleh, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Yuta Takahashi, Takaaki Tanaka, Isabelle Tellier, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Veronika Vincze, Lars Wallin, Jonathan North Washington, Seyi Williams, Mats Wirén, Tsegay Woldemariam, Tak-sum Wong, Chunxiao Yan, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2020. Probing the probing paradigm: Does probing accuracy entail task relevance? *arXiv preprint arXiv:2005.00719*.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Tanja Schultz and Alex Waibel. 2001. Language-independent and language-adaptive acoustic modeling for speech recognition. *Speech Communication*, 35(1-2):31–51.

Jasdeep Singh, Bryan McCann, Richard Socher, and Caiming Xiong. 2019. Bert is not an interlingua and the bias of tokenization. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 47–55.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 682–686. Association for Computational Linguistics.

Corso Svizzera. 2014. Converting the parallel treebank partut in universal stanford dependencies.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*.

A Appendices

A.1 Reproducibility

A.1.1 Optimization

We fine-tune our models using the standard Adam optimizer (Kingma and Ba, 2015). We warmup the learning rate on the first 10% steps and use linear decay in the rest of the training. Using the validation set of the source language, we find the best combination of hyper-parameters with a grid search on batch size among {16, 32} and learning rate initialization among {1e-5, 2.5e-5, 5e-5} We select the model with the highest validation performance out of 15 epochs for parsing and out of 6 epochs for POS tagging and NER.

Hyperparameters In Table 3, we report the best hyper-parameters set for each task, the bound of each hyperparameter, the estimated number of grid search trial for each task as well as the estimated run time.

A.1.2 Data

Data Sources We base our experiments on data originated from two sources. The Universal Dependency project (McDonald et al., 2013) downloadable here `https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2988` and the WikiNER dataset (Pan et al., 2017). We also make use of the CoNLL-2003 shared task NER English dataset `https://www.clips.uantwerpen.be/conll2003/`

Languages For all our experiments, we use English, Russian and Arabic as source languages in addition to Chinese, Czech, Finish, French, Indonesian, Italian, Japanese, German, Hindi, Polish, Portuguese, Slovenian, Spanish, and Turkish as target languages.

Fine-tuning Data For all the cross-lingual experiments, we use English, Russian and Arabic as source languages on which we fine-tune mBERT. For English, we take the English-EWT treebank for fine-tuning, for Russian the Russian-GSD treebank and for Arabic the Arabic-PADT treebank.

Evaluation Data For all our experiments, we perform the evaluation on all the 17 languages. For Parsing and POS tagging we use the test set from the PUD treebanks released for the CoNLL Shared Task 2017 (Zeman et al., 2017). For NER, we use the corresponding annotated datasets in the wikiner dataset.

Domain Analysis Datasets We list here the datasets for completing our domain analysis experiment in Section 4.1 reported in Table 2. To have a full control on the source domains, we use for fine-tuning the English Partut treebank for POS tagging and parsing (Svizzera, 2014). It is a mix of legal, news and wikipedia text. For NER, we keep the WikiANN dataset (Pan et al., 2017). For the same-language and out-of-domain experiments, we use the English-EWT, English-Lines and English Lexnorm (van der Goot and van Noord, 2018) treebanks for Web Media data, Literature data and Noisy tweets respectively. For the cross-language French evaluation, we use the translation of the English test set,[6] as well as the French-GSD treebank. For NER, we take the CoNLL-2003 shared task English data as our out-of-domain evaluation extracted from the *News* domain. We note that the absolute performance on this dataset is not directly comparable to the one on the source wikiner. Indeed, the CoNLL-2003 dataset uses an extra MISC class. In our work, we only interpret the relative performance of different models on this test set.

Cross-Lingual Similarity Analysis For a given source language l and a target language l', we collect a 1000 pairs of aligned sentences from the UD-PUD treebanks (Zeman et al., 2017). For a given model and for each layer, we get a single sentence embedding by averaging token-level embeddings (after excluding special tokens). We then concatenate the 1000 sentence embedding vectors and get the matrices X_l and X'_l. Based on these two matrices, the CKA between the language l and the language l' is defined as:

$$CKA(X_l, X_{l'}) = \frac{||X_l^T X_{l'}||_F^2}{||X_l^T X_l||_F ||X_{l'}^T X_{l'}||_F}$$

with $||.||_F$ defining the Frobenius norm.

We do so for each source-target language pairs using the representation of the pretrained mBERT model as well as for mBERT fine-tuned on each downstream task.

In addition to the results presented in §4.2, we report in Figure 4, a comparison of the cross-lingual similarity per hidden layer of mBERT fine-tuned on NER, across target languages. The trend is the same for all languages.

---

[6] We do so by taking the French-ParTUT test set that overlaps with the English-ParTUT, which includes 110 sentences.

A.1.3 Computation

Infrastructure Our experiments were ran on a shared cluster on the equivalent of 15 Nvidia Tesla T4 GPUs.[7]

Codebase All of our experiments are built using the Transformers library described in (Wolf et al., 2020). We also provide code to reproduce our experiments at `https://github.com/benjamin-mlr/first-align-then-predict.git`.

A.1.4 Preprocessing

Our experiments are ran with word-level tokenization as provided in the datasets. We then tokenize each sequence of words at the sub-word level using the *Wordpiece* algorithm of BERT and provided by the Transformers library.

Params.	Parse	NER	POS	Bounds
batch size	32	16	16	[1,256]
learning rate	5e-5	3.5e-5	5e-5	[1e-6,1e-3]
epochs (best)	15	6	6	[1,50]
#grid	60	60	180	-
Run-time (min)	32	24	75	-

Table 3: Fine-tuning best hyper-parameters for each task as selected on the validation set of the source language with bounds. #grid: number of grid search trial. Run-time is reported in average for training and evaluation.

A.2 Cross-lingual transfer analyses

A.2.1 Correlation

We report here in Figure 4 the correlation between the hidden representation of each layer and the *cross-lang gap* between the source and the target averaged across all target languages and all layers. The correlation is strong and significant for all the tasks and for both the fine-tuned and the pretrained models. This shows that multilingual alignment that occurs within the models, learnt during pretraining is strongly related with cross-lingual transfer.

We report in Figure 3, the detail of this correlation per layer. For the pretrained model, we observe the same distribution for each task with layer 6 being the most correlated to cross-lingual transfer. We observe large variations in the fine-tuned cases, the most notable being NER. This illustrates the task-specific aspect of the relation between cross-lingual similarity and cross-lingual transfer. More precisely, in the case of NER, the sharp increase and decrease in the upper part of the model provides new evidence that for this task, fine-tuning highly impacts cross-lingual similarity in the upper part of the model which correlates with cross-language transfer.

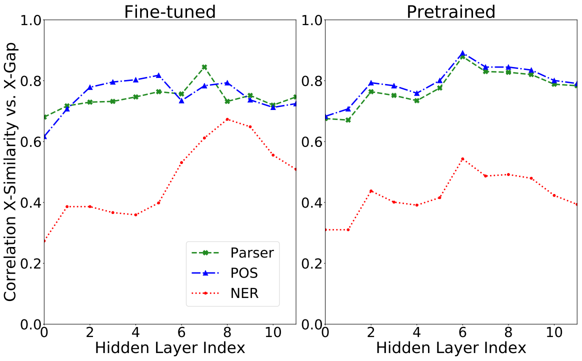


Figure 3: Spearman Correlation between Cross-Lingual Similarity (CKA between English and the target representations) and *cross-lang gap* averaged over all 17 target languages for each layer

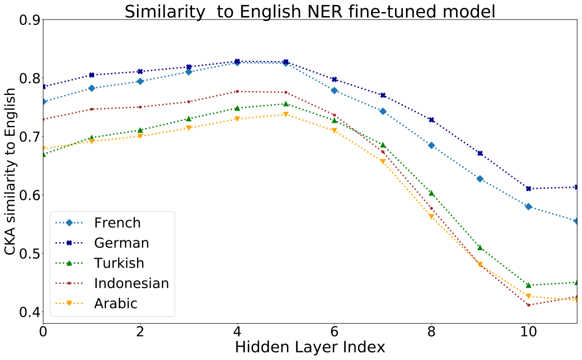


Figure 4: Cross-Lingual Similarity (CKA) (§4.2) of hidden representations of a source language (English) sentences with target languages of mBERT fine-tuned for NER. The higher the CKA value the greater the similarity.

Task	Correlation X-Gap vs. X-Similarity	
	FINE-TUNED	PRETRAINED
Parsing	0.76	0.79
POS	0.74	0.82
NER*	0.47	0.43

Table 4: Spearman-Rank Correlation between the *Cross-lingual Gap* (X-Lang Gap) and the *Cross-lingual Similarity* between the source and the target languages of the fine-tuned models and the pretrained model averaged over all the hidden layers and all the 17 target languages (sample size per task: 17). For NER, *cross-lang gap* measured on wikiner data and not on the parrallel data itself in constrast with Parsing and POS tagging. Complete list of languages can be found in Appendix A.1.2

---

[7] https://www.nvidia.com/en-sg/data-center/tesla-t4/

						Random-init *of layers*							
Eval	Ref	All	1	2	1-2	3-4	1-3	4-6	7-9	10-12	1-4	5-8	9-12
							Parsing						
English Dev	88.52	74.66	87.77	88.03	87.28	86.81	83.77	85.86	87.53	88.78	84.30	85.41	88.35
English Test	88.59	74.58	87.77	88.09	87.25	86.79	83.37	85.54	87.36	88.62	83.10	85.37	88.69
French	68.94	3.70	65.73	65.21	55.31	61.31	43.81	61.77	67.03	69.36	37.29	61.82	69.26
German	67.43	4.73	64.97	65.20	57.08	60.62	47.85	58.93	64.12	66.67	36.05	59.37	67.21
Turkish	28.40	2.76	21.65	23.77	16.78	21.21	10.69	20.23	25.39	30.43	9.70	20.94	29.33
Indonesian	45.13	4.99	43.33	43.48	39.83	39.09	33.06	40.65	44.42	46.96	30.35	40.85	47.53
Russian	59.70	2.95	57.81	57.53	54.10	53.51	47.01	52.37	56.45	61.41	38.58	52.41	60.72
Arabic	23.37	3.19	23.66	23.49	21.01	19.55	16.17	18.84	20.70	24.54	13.26	18.27	23.93
							POS						
English Dev	96.45	87.47	96.04	96.06	95.92	95.81	95.38	95.43	96.25	96.58	94.01	95.35	96.39
English Test	96.53	87.71	96.08	96.24	95.94	95.72	95.40	95.59	96.34	96.74	94.05	95.45	96.51
French	88.25	28.96	86.70	87.66	79.84	87.14	69.43	86.42	86.94	88.30	62.28	86.37	88.26
German	90.63	28.93	88.26	89.53	82.26	88.39	71.63	88.30	90.26	90.83	59.16	89.12	90.64
Turkish	72.65	32.23	62.17	66.17	54.50	63.22	47.77	66.37	70.91	72.92	44.16	69.30	73.08
Indonesian	84.06	36.98	82.15	82.89	80.13	81.40	75.94	81.99	83.78	84.42	72.42	82.59	84.09
Russian	82.97	32.63	83.14	83.63	81.95	82.26	77.93	81.69	82.98	81.76	70.33	82.56	83.19
Arabic	56.66	19.61	58.10	58.06	57.89	55.62	57.93	54.69	56.04	55.97	52.28	53.60	58.84
							NER						
English Dev	83.29	56.99	82.04	82.26	79.52	80.36	76.22	79.53	82.18	82.53	69.31	80.05	82.47
English Test	83.06	56.56	81.46	82.00	79.63	79.25	76.68	78.93	81.64	82.39	69.08	79.91	82.27
French	76.76	35.35	75.46	77.57	69.94	72.83	65.14	70.34	75.42	75.90	55.79	73.12	75.77
German	76.68	18.95	73.73	75.39	66.18	70.12	56.50	69.53	75.38	77.11	42.37	71.14	75.50
Turkish	67.64	20.76	62.54	64.84	52.20	57.11	53.03	60.59	65.66	64.87	39.38	61.43	66.62
Indonesian	53.47	21.20	49.19	49.27	46.50	46.87	43.75	47.83	54.39	48.71	36.11	46.06	48.23
Russian	58.23	7.43	55.63	58.08	50.67	52.89	42.83	46.13	53.38	58.09	34.66	52.03	59.12
Arabic	41.81	5.49	35.79	34.80	32.37	32.31	26.21	38.88	38.55	40.83	21.85	38.67	41.23

Table 5: Zero-shot cross-lingual performance when applying Random-init to specific set of consecutive layers compared to the Ref model. Source language is English. Baseline model ALL (for all layers randomly initialized) corresponds to a model trained from scratch on the task. For reproducibility purposes, we report performance on the Validation set English Dev. For all target languages, we report the scores on the test split of each dataset. Each score is the average of 5 runs with different random seeds. For more insights into the variability of our results, we report the min., median and max. value of the standard deviations (std) across runs with different random seeds for each task: Parsing:0.02/0.34/1.48, POS:0.01/0.5/2.38, NER:0.0/0.47/2.62 (std min/median/max).

$\geq$ Ref $<$ Ref $\leq$ 5 points $\leq$ 10 points

				RANDOM-INIT *of layers*			
SOURCE - TARGET	REF	Δ 0-1	Δ 2-3	Δ 4-5	Δ 6-7	Δ 8-9	Δ 10-11
				Parsing			
EN - ENGLISH	88.98	-0.96	-0.66	-0.93	-0.55	0.04	-0.09
EN - ARABIC	35.88	-4.05	-2.38	-3.16	-0.78	1.74	1.68
EN - FRENCH	74.04	-21.30	-6.84	-2.93	-0.69	0.03	0.76
EN - GERMAN	70.34	-15.06	-9.26	-4.75	-1.54	-0.29	1.82
EN - TURKISH	34.03	-16.37	-10.10	-5.11	-3.71	0.43	1.43
EN - INDO	44.11	-10.57	-5.87	-2.66	-0.96	-0.74	0.73
EN - RUSSIAN	62.52	-7.31	-5.37	-2.84	-1.09	0.44	0.71
EN - PORTHUGHESE	68.59	-25.83	-6.22	-2.97	-0.77	0.15	0.82
EN - SPANISH	69.96	-18.05	-5.74	-2.78	-0.96	0.13	0.72
EN - FINISH	48.42	-24.25	-9.48	-4.39	-2.51	-0.28	0.22
EN - ITALIAN	74.54	-30.54	-9.63	-4.18	-1.32	-0.12	0.90
EN - SLOVENIAN	73.04	-29.89	-6.52	-3.00	-1.68	-0.05	0.18
EN - CZECH	60.44	-31.84	-10.69	-4.61	-1.82	0.18	1.17
EN - POLISH	55.23	-23.57	-9.11	-3.34	-1.83	0.28	0.89
EN - HINDI	28.86	-9.13	-7.58	-5.84	-2.50	1.35	1.49
EN - CHINESE	27.48	-7.31	-4.47	-1.65	-0.62	0.65	1.32
EN - JAPANESE	11.99	-4.36	-2.76	-1.91	-1.19	0.47	1.12
EN - X (MEAN)	53.23	-15.77	-6.51	-3.39	-1.47	0.29	1.00
RU - RUSSIAN	85.15	-0.82	-1.38	-1.51	-0.86	-0.29	0.18
RU - ENGLISH	61.40	-8.37	-3.55	-3.90	-0.72	1.77	1.14
RU - ARABIC	59.41	-5.65	-5.26	-5.15	-1.47	0.24	0.16
RU - FRENCH	65.84	-8.87	-2.93	-1.81	-1.05	3.81	1.24
RU - GERMAN	65.90	-7.02	-4.19	-1.97	-1.45	2.58	2.05
RU - TURKISH	32.20	-13.13	-7.18	-6.82	-3.77	-0.85	1.21
RU - INDO	47.59	-4.74	-2.99	-2.30	-1.81	0.04	1.02
RU - PORTHUGHESE	66.41	-11.17	-1.61	-1.09	-1.25	4.16	1.94
RU - SPANISH	66.74	-4.52	-1.38	-0.69	-0.97	2.95	1.37
RU - FINISH	52.92	-15.43	-6.59	-4.09	-1.35	0.12	0.77
RU - ITALIAN	65.28	-12.97	-3.56	-2.34	-1.46	3.16	1.55
RU - SLOVENIAN	62.91	-16.67	-2.71	-3.18	-1.03	0.31	1.08
RU - CZECH	72.77	-11.95	-4.17	-3.13	-1.57	-0.33	0.30
RU - POLISH	66.07	-5.70	-3.22	-2.57	-1.54	-0.12	0.54
RU - HINDI	28.67	-6.02	-5.77	-5.27	-3.75	-0.06	0.99
RU - CHINESE	28.77	-4.66	-4.38	-3.22	-1.80	0.15	1.12
RU - JAPANESE	15.10	-4.89	-3.56	-3.95	-3.11	0.68	0.73
RU - X (MEAN)	55.41	-7.69	-3.71	-3.13	-1.70	0.92	0.94
AR - ARABIC	59.54	-0.78	-2.14	-1.20	-0.67	-0.27	0.08
AR - ENGLISH	25.46	-2.09	-2.92	-0.90	-1.40	-0.97	-0.61
AR - FRENCH	28.92	-4.85	-1.45	-0.25	-2.72	-1.60	-0.88
AR - GERMAN	27.14	-6.38	-4.51	-0.98	-2.24	0.13	0.09
AR - TURKISH	9.58	-3.90	-3.14	-2.76	-2.33	0.31	0.15
AR - INDO	36.16	-5.85	-4.86	-1.71	-0.68	-0.17	0.58
AR - RUSSIAN	42.25	-3.52	-5.28	-2.46	-1.66	-0.67	-0.27
AR - PORTHUGHESE	34.71	-4.80	-1.22	0.10	-2.98	-0.33	-0.24
AR - SPANISH	31.95	-4.02	-0.15	-0.44	-1.46	-0.77	0.38
AR - FINISH	28.18	-9.89	-7.03	-3.17	-1.81	-0.58	-0.42
AR - ITALIAN	28.85	-3.01	0.60	1.45	-2.26	-1.47	-0.70
AR - SLOVENIAN	35.78	-9.73	-4.97	-2.21	-1.43	-0.41	-0.56
AR - CZECH	40.04	-13.61	-6.82	-3.20	-2.38	-1.12	-0.21
AR - POLISH	41.16	-8.46	-5.52	-2.48	-1.48	-0.47	-0.55
AR - HINDI	10.24	-2.46	-2.86	-2.57	-1.55	1.00	0.14
AR - CHINESE	11.46	-2.42	-2.43	-1.26	-0.82	0.23	-0.05
AR - JAPANESE	6.66	-1.28	-0.79	-1.20	-1.04	0.74	0.30
AR - X (MEAN)	27.97	-4.91	-3.17	-1.48	-1.68	-0.36	-0.14

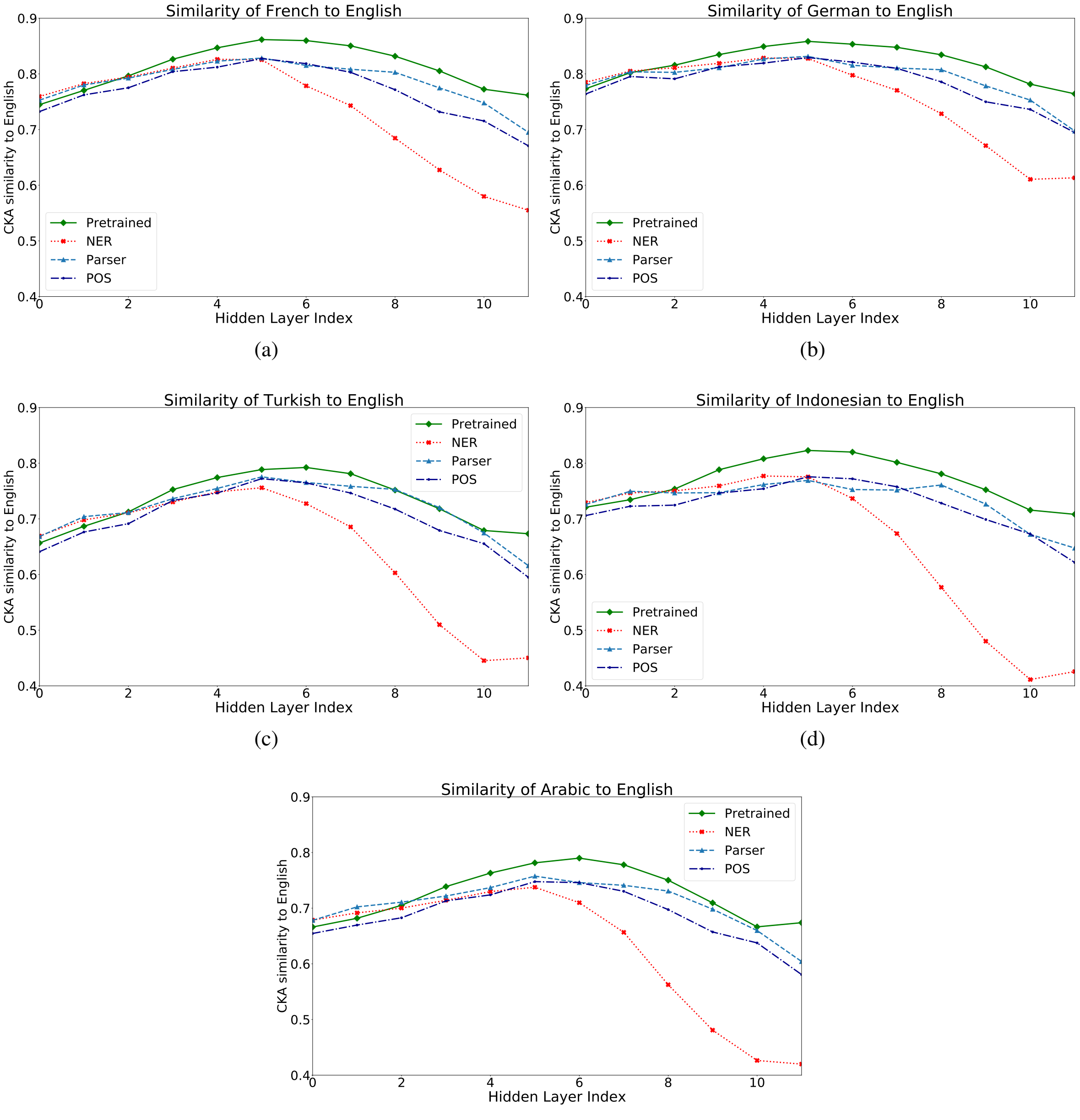
Table 6: Parsing (LAS score) Relative Zero shot Cross-Lingual performance of mBERT with RANDOM-INIT (section 2.1) on pairs of consecutive layers compared to mBERT without any random-initialization (REF). In SRC - TRG, SRC indicates the source language on which we fine-tune mBERT, and TRG the target language on which we evaluate it. SRC-X is the average across all 17 target language with X ≠ SRC ≥ REF < REF ≤ -2 points ≤ -5 points

			Random-Init *of layers*				
Source - Target	Ref	Δ 0-1	Δ 2-3	Δ 4-5	Δ 6-7	Δ 8-9	Δ 10-11
					POS		
En - English	96.51	-0.30	-0.25	-0.40	-0.00	0.05	0.02
En - Arabic	70.20	-3.63	-1.88	-2.40	-1.26	-1.89	-2.74
En - French	89.16	-9.68	-2.09	-1.49	-1.03	0.29	0.59
En - German	89.32	-7.81	-2.12	-1.27	-0.99	-0.46	-0.68
En - Turkish	71.67	-11.62	-4.43	-1.48	-0.95	0.04	-0.95
En - Indo	71.44	-6.39	-2.80	-1.74	-0.59	-0.41	-1.10
En - Russian	86.26	-2.66	-0.94	-0.27	0.13	0.37	0.62
En - Porthughese	86.51	-10.84	-1.83	-1.44	-0.81	-0.01	-0.14
En - Spanish	87.26	-8.09	-1.30	-1.36	-1.13	0.20	0.17
En - Finish	84.85	-20.00	-8.09	-2.77	-0.97	-0.06	-0.86
En - Italian	91.35	-13.97	-3.35	-2.66	-1.34	-0.01	0.27
En - Slovenian	89.64	-16.46	-2.41	-1.09	-0.18	0.34	0.19
En - Czech	83.39	-19.62	-3.93	-0.73	-0.56	0.21	0.29
En - Polish	81.45	-13.33	-3.52	-1.19	-1.22	-0.50	-0.16
En - Hindi	65.43	-10.04	-2.70	-2.89	-3.25	3.00	0.28
En - Chinese	67.89	-3.04	-2.82	-3.59	-0.29	0.66	0.29
En - Japanese	48.86	-2.19	1.52	-1.51	-1.13	1.42	1.79
En - X (Mean)	79.37	-8.94	-2.49	-1.66	-0.88	0.20	-0.14
Ru - Russian	96.90	-0.52	-0.55	-0.40	-0.07	0.02	-0.03
Ru - English	82.55	-20.72	-7.06	-5.01	-3.93	0.74	-1.57
Ru - Arabic	79.30	-4.04	-1.48	-2.06	0.64	0.01	0.47
Ru - French	86.02	-18.66	-4.64	-4.10	-9.00	-0.13	-1.84
Ru - German	84.90	-12.50	-4.80	-2.79	-3.90	0.47	-1.82
Ru - Turkish	69.92	-15.20	-2.06	-0.55	-1.41	-0.11	0.68
Ru - Indo	71.16	-8.33	-3.44	-1.03	-0.56	-0.73	0.15
Ru - Porthughese	84.24	-19.56	-7.15	-3.00	-7.78	-0.15	-2.08
Ru - SPanish	84.84	-13.64	-4.09	-2.66	-7.67	-0.35	-2.48
Ru - Finish	81.08	-18.55	-5.42	-1.37	-1.00	-0.16	0.02
Ru - Italian	85.56	-21.04	-5.11	-3.41	-8.21	-0.20	-3.36
Ru - Slovenian	85.37	-14.65	-3.53	-1.72	-2.00	-0.15	-0.15
Ru - Czech	87.37	-8.43	-1.99	-0.71	-1.16	-0.50	-0.28
Ru - Polish	86.42	-4.41	-1.89	-0.64	-0.44	-0.21	0.09
Ru - Hindi	65.49	-1.16	0.41	-1.49	-2.17	1.13	3.20
Ru - Chinese	65.85	-5.12	-1.43	-0.32	-0.74	-0.13	-0.47
Ru - Japanese	46.91	-0.72	2.16	0.00	-1.30	1.15	1.12
Ru - X (Mean)	79.25	-10.08	-2.83	-1.65	-2.74	0.01	-0.45
Ar - Arabic	79.28	-0.35	-0.49	-0.36	-0.19	-0.05	-0.00
Ar - English	63.26	-3.32	-1.09	-1.72	-1.68	-1.03	-1.78
Ar - French	63.33	-4.41	-1.53	-1.14	-1.30	-0.44	-0.92
Ar - German	63.23	-4.95	-2.97	-1.04	-1.58	-0.53	-2.09
Ar - Turkish	60.99	-13.76	-8.74	-2.86	-4.49	-1.08	-1.88
Ar - Indo	64.24	-5.11	-3.43	-1.87	-0.58	-0.28	-0.63
Ar - Russian	74.52	-4.01	-2.37	-2.40	-1.84	-1.69	-2.03
Ar - Porthughese	67.28	-6.51	-2.84	-1.30	-1.23	0.04	-0.96
Ar - SPanish	64.84	-3.08	-0.51	-0.74	-0.48	0.02	-0.14
Ar - Finish	64.28	-19.72	-8.32	-3.72	-2.56	-1.64	-3.03
Ar - Italian	63.55	-4.25	-1.60	-0.94	-1.15	0.14	-0.64
Ar - Slovenian	68.06	-12.21	-4.31	-2.17	-1.85	0.68	-1.81
Ar - Czech	72.65	-13.57	-3.14	-1.88	-1.77	-1.35	-1.57
Ar - Polish	75.00	-8.87	-2.94	-1.46	-0.62	-1.00	-1.37
Ar - Hindi	62.29	-7.31	-6.07	-2.42	-1.26	0.19	-1.72
Ar - Chinese	56.51	-5.02	-4.94	-2.10	-1.35	-1.02	-1.77
Ar - Japanese	47.06	-3.34	-3.34	-0.65	-0.89	-1.54	-0.35
Ar - X (Mean)	64.81	-6.73	-3.50	-1.63	-1.56	-0.73	−1.29

Table 7: POS tagging Relative Zero shot Cross-Lingual performance of mBERT with Random-Init (section 2.1) on pairs of consecutive layers compared to mBERT without any random-initialization (Ref). In Src - Trg, Src indicates the source language on which we fine-tune mBERT, and Trg the target language on which we evaluate it. Src-X is the average across all 17 target language with $X \neq$ Src. ≥ Ref < Ref ≤ -2 points ≤ -5 points

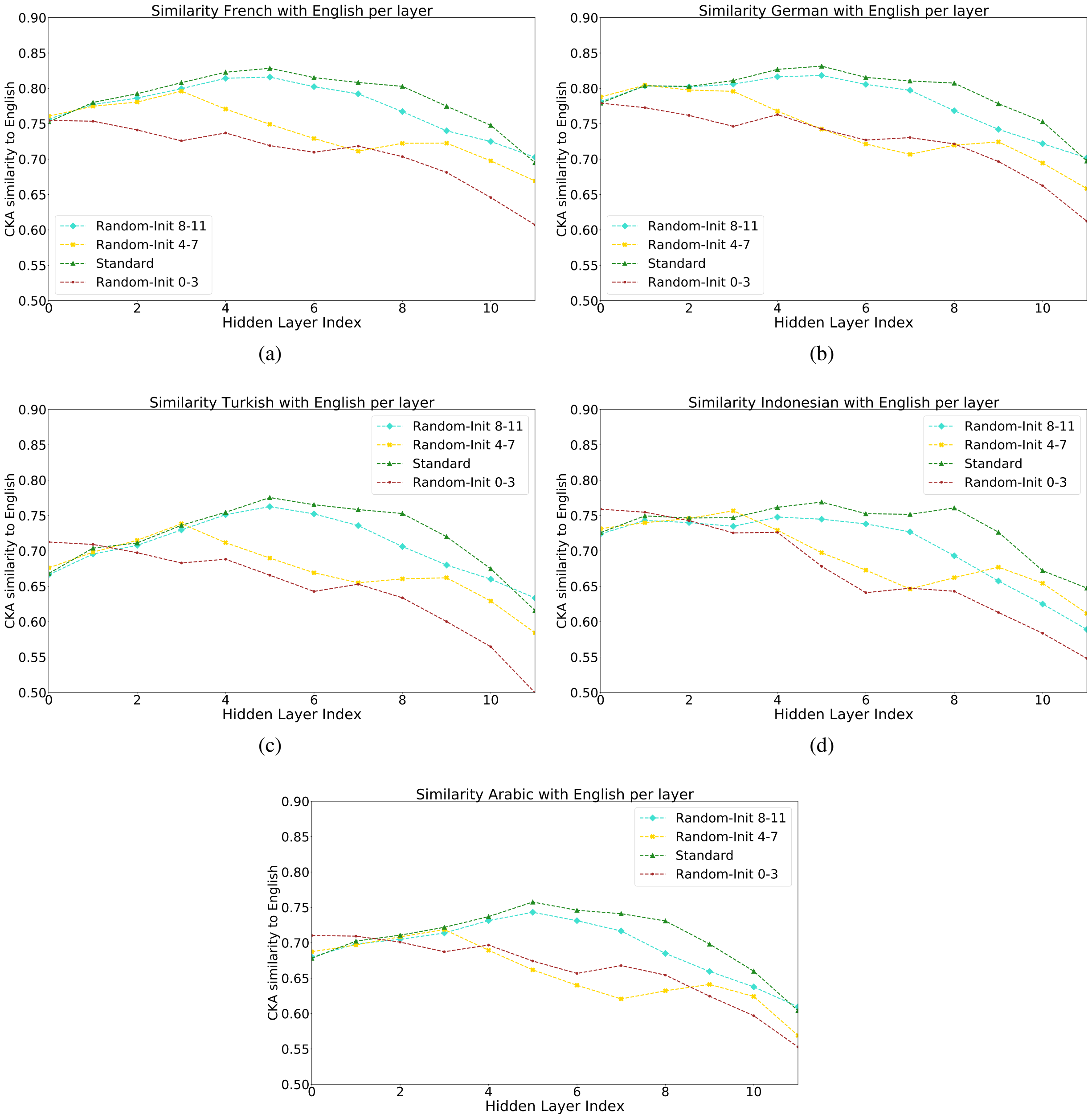
				RANDOM-INIT *of layers*			
Source - Target	REF	Δ 0-1	Δ 2-3	Δ 4-5	Δ 6-7	Δ 8-9	Δ 10-11
					NER		
EN - ENGLISH	83.27	-2.64	-2.12	-1.41	-0.61	-0.21	-0.14
EN - FRENCH	76.20	-4.41	-2.72	-2.09	-0.30	0.51	0.08
EN - GERMAN	75.58	-8.25	-4.65	-2.50	-0.40	0.06	0.26
EN - TURKISH	66.23	-8.71	-6.57	-2.16	-1.01	0.51	0.51
EN - INDO	50.24	-2.94	-1.43	-2.54	2.49	-0.70	0.82
EN - PORTHUGHESE	76.09	-4.66	-0.88	-1.16	-0.57	0.62	-0.70
EN - SPANISH	67.00	-0.99	4.37	2.03	-1.69	1.57	-1.38
EN - FINISH	75.61	-11.89	-4.47	-2.29	0.63	0.54	-0.37
EN - ITALIAN	78.48	-6.65	-3.64	-3.08	-1.32	-0.30	-0.28
EN - SLOVENIAN	72.80	-10.37	-2.96	-3.11	-0.36	0.10	-0.72
EN - CZECH	76.90	-8.02	-6.81	-3.17	0.09	1.00	0.39
EN - RUSSIAN	60.20	-5.87	-6.65	-5.71	-2.82	-0.82	-0.37
EN - ARABIC	39.15	-8.98	-5.31	-1.97	1.56	0.31	-0.98
EN - POLISH	77.20	-8.32	-5.53	-3.05	-0.06	0.67	0.09
EN - HINDI	60.61	-12.08	-13.88	-9.23	-0.91	-1.25	2.08
EN - CHINESE	37.74	-13.68	-6.49	-4.59	-2.41	-5.23	-1.00
EN - JAPANESE	25.19	-11.40	-7.54	-4.67	-2.53	-3.45	-0.23
EN - X (MEAN)	64.17	-8.28	-5.09	-3.07	-0.79	-0.47	-0.13
RU - RUSSIAN	88.20	-2.08	-2.13	-1.52	-0.64	-0.33	-0.13
RU - ENGLISH	56.62	-13.83	-8.52	-4.70	-1.50	-0.76	1.38
RU - FRENCH	67.35	-18.45	-9.70	-4.32	-1.76	-1.77	2.29
RU - GERMAN	69.23	-13.94	-9.01	-5.80	-2.98	-1.65	0.40
RU - TURKISH	63.64	-18.52	-10.06	-6.01	-4.16	-0.67	-0.27
RU - INDO	41.92	-10.29	-7.20	-5.19	-1.20	-1.91	0.50
RU - PORTHUGHESE	67.33	-21.23	-8.27	-8.84	-2.83	-1.83	1.51
RU - SPANISH	69.15	-16.74	-10.00	-8.16	-5.80	-1.66	0.26
RU - FINISH	73.03	-17.17	-8.70	-5.88	-2.12	0.86	1.48
RU - ITALIAN	70.05	-19.47	-9.54	-6.90	-3.06	0.73	1.04
RU - SLOVENIAN	71.18	-12.02	-9.48	-3.61	-0.70	1.16	2.14
RU - CZECH	74.87	-17.93	-10.59	-6.34	-4.02	0.17	-0.23
RU - ARABIC	38.63	-8.67	-6.81	-0.13	-0.65	-1.34	-0.29
RU - POLISH	75.16	-15.38	-7.97	-6.33	-3.07	-0.63	1.34
RU - HINDI	58.01	-19.60	-12.36	-6.18	0.93	-1.64	1.17
RU - CHINESE	43.86	-23.73	-11.68	-6.80	-4.27	-4.13	-6.01
RU - JAPANESE	30.79	-16.80	-11.29	-5.26	-2.77	-3.99	-6.91
RU - X (MEAN)	62.13	-15.85	-9.36	-5.50	-2.44	-1.16	-0.06
AR - ARABIC	87.97	-2.37	-2.11	-0.96	-0.39	-0.15	0.21
AR - FRENCH	75.21	-18.71	-8.31	-3.76	-0.19	0.82	1.07
AR - GERMAN	74.24	-15.25	-7.19	-3.72	-1.38	-0.04	0.27
AR - TURKISH	68.45	-14.89	-8.65	-2.78	-0.30	0.98	1.90
AR - INDO	54.65	-13.86	-10.95	-8.53	-4.66	-2.82	0.09
AR - PORTHUGHESE	74.67	-20.42	-10.54	-3.17	-1.59	0.10	1.28
AR - SPANISH	74.88	-18.16	-12.18	-3.06	-1.95	0.52	0.63
AR - FINISH	78.01	-18.79	-8.84	-4.30	-2.03	-0.30	0.19
AR - ITALIAN	75.76	-16.37	-7.73	-3.98	-1.49	-0.06	0.74
AR - SLOVENIAN	63.08	-11.13	-5.49	4.79	0.88	2.17	0.79
AR - CZECH	74.70	-21.93	-10.95	-5.84	-2.42	-1.36	0.09
AR - RUSSIAN	45.51	-7.59	-5.81	-2.63	0.15	-0.22	0.47
AR - ENGLISH	57.94	-12.79	-6.03	-4.57	-0.32	0.29	1.65
AR - POLISH	77.29	-20.61	-9.47	-5.93	-2.64	-1.09	-0.19
AR - HINDI	65.31	-14.95	-9.12	-3.84	-1.48	0.72	0.98
AR - CHINESE	45.88	-25.72	-10.67	-3.99	-1.41	-2.72	0.57
AR - JAPANESE	24.75	-14.66	-5.19	-3.82	-0.99	-1.17	1.50
AR - X (MEAN)	65.59	-16.10	-8.42	-3.73	-1.40	-0.25	0.67

Table 8: NER (F1 score) Relative Zero shot Cross-Lingual performance of mBERT with RANDOM-INIT (section 2.1) on pairs of consecutive layers compared to mBERT without any random-initialization (REF). In SRC - TRG, SRC indicates the source language on which we fine-tune mBERT, and TRG the target language on which we evaluate it. SRC-X is the average across all 17 target language with X $\neq$ SRC $\geq$ REF $<$ REF $\leq$ -2 points $\leq$ -5 points



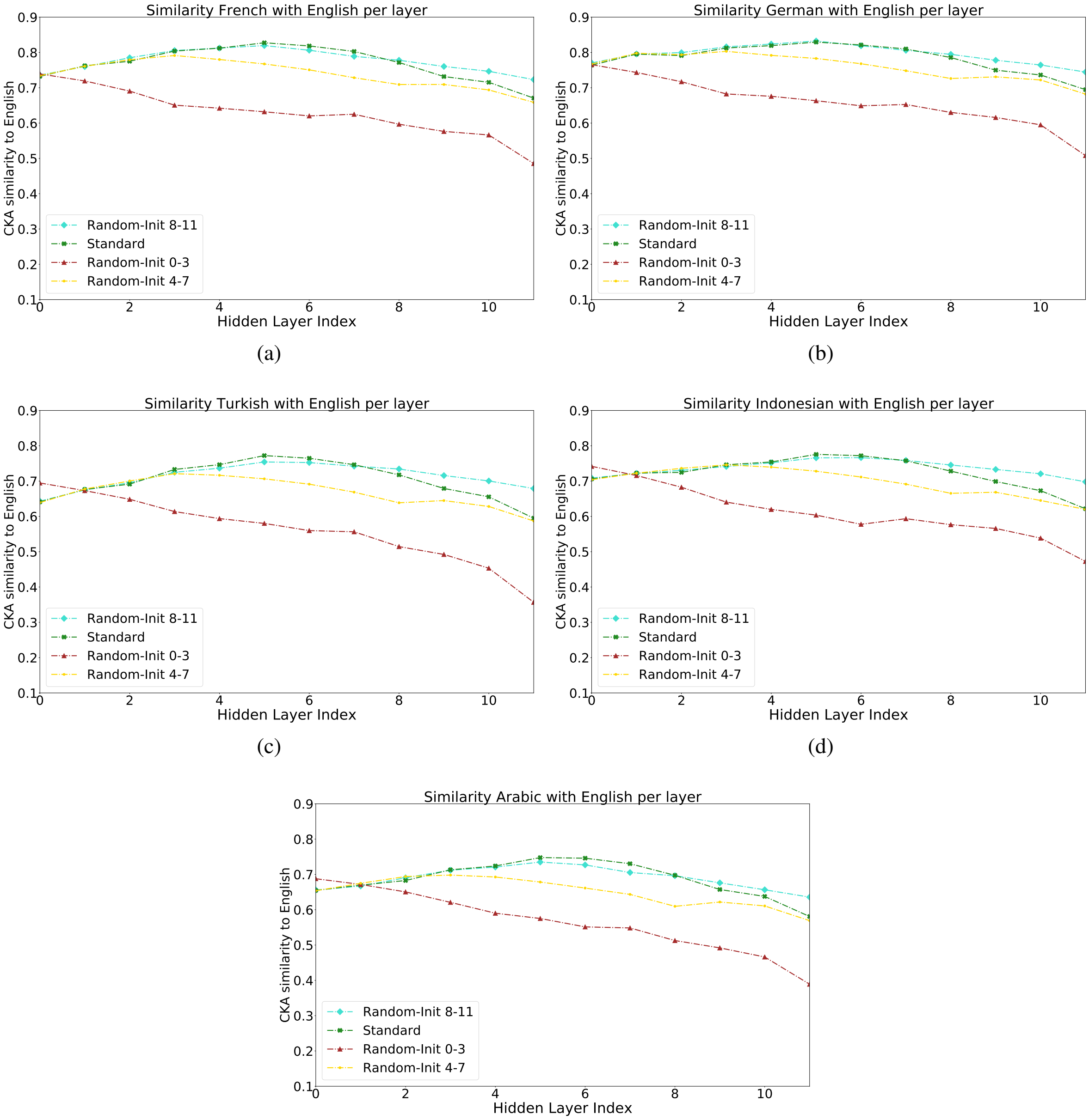
(a) (b) (c) (d) (e)

Figure 5: Cross-Lingual similarity (CKA) similarity (§4.2) of hidden representations of a source language (English) sentences with a target language sentences on fine-tuned and pretrained mBERT. The higher the CKA value the greater the similarity.



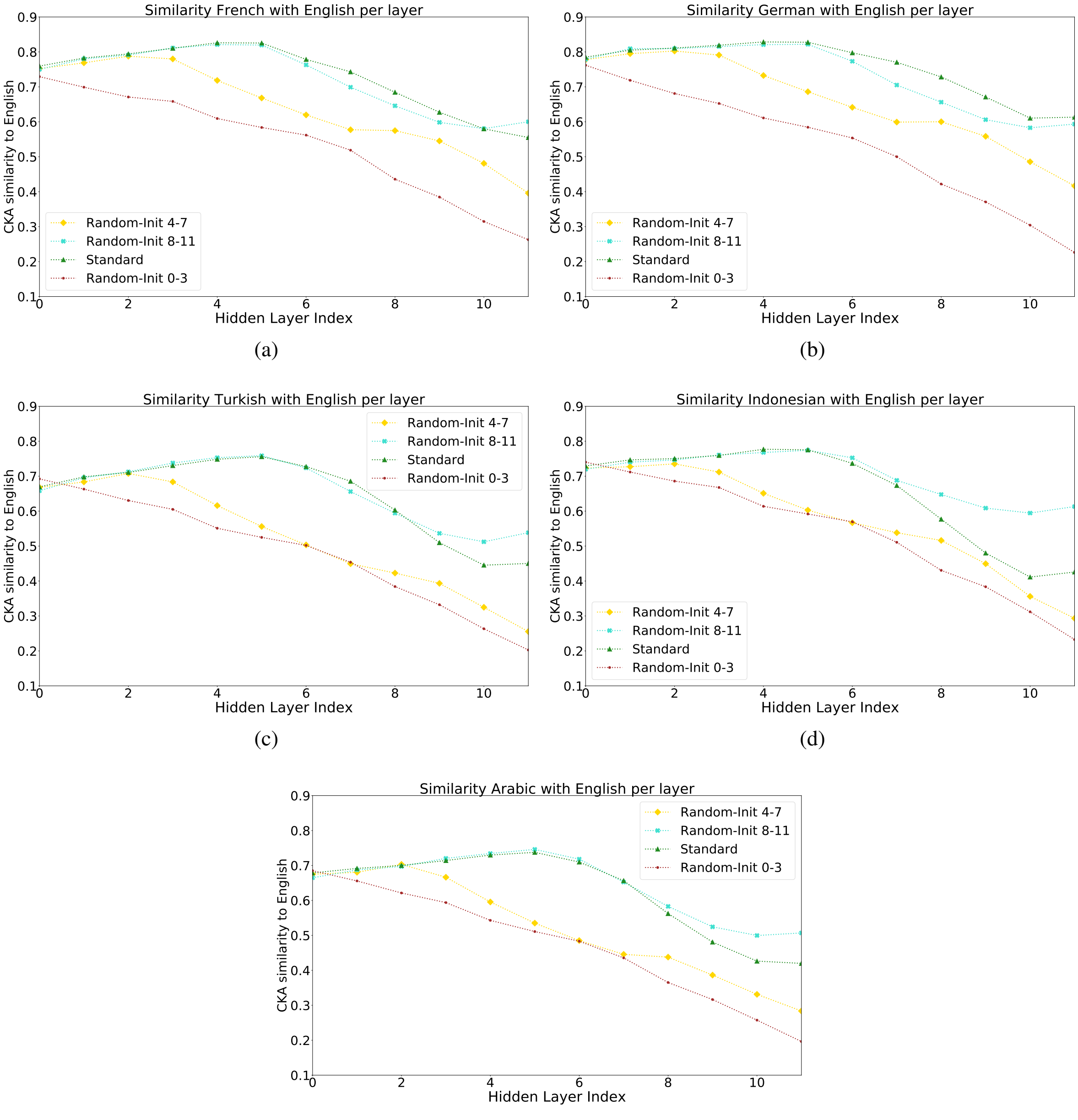
(a) (b) (c) (d) (e)

Figure 6: Cross-Lingual similarity (CKA) (§4.2) of hidden representations of a source language (English) sentences with target languages sentences on fine-tuned **Parsing** models with and without RANDOM-INIT. The higher the CKA value the greater the similarity.



(a) (b) (c) (d) (e)

Figure 7: Cross-Lingual similarity (CKA) (§4.2) of hidden representations of a source language (English) sentences with target languages sentences on fine-tuned **POS** models with and w/o RANDOM-INIT. The higher the CKA value the greater the similarity.


Similarity French with English per layer

Similarity German with English per layer

Similarity Turkish with English per layer

Similarity Indonesian with English per layer

Similarity Arabic with English per layer

(a) (b)

(c) (d)

(e)

Figure 8: Cross-Lingual similarity (CKA) (§4.2) of hidden representations of a source language (English) sentences with target languages sentences on fine-tuned **NER** models with and w/o RANDOM-INIT. The higher the CKA value the greater the similarity.

Stereotype and Skew: Quantifying Gender Bias in Pre-trained and Fine-tuned Language Models

Daniel de Vassimon Manela[†] **David Errington**[†] **Thomas Fisher**[†]
Boris van Breugel[†] **Pasquale Minervini**
University College London
`{daniel.manela.19,david.errington.19,thomas.fisher.19,`
`boris.breugel.19,p.minervini}@ucl.ac.uk`

Abstract

This paper proposes two intuitive metrics, *skew* and *stereotype*, that quantify and analyse the gender bias present in contextual language models when tackling the WinoBias pronoun resolution task. We find evidence that gender stereotype correlates approximately negatively with gender skew in out-of-the-box models, suggesting that there is a trade-off between these two forms of bias. We investigate two methods to mitigate bias. The first approach is an online method which is effective at removing skew at the expense of stereotype. The second, inspired by previous work on ELMo, involves the fine-tuning of BERT using an augmented gender-balanced dataset. We show that this reduces both skew and stereotype relative to its unaugmented fine-tuned counterpart. However, we find that existing gender bias benchmarks do not fully probe professional bias as pronoun resolution may be obfuscated by cross-correlations from other manifestations of gender prejudice. Our code is available online.

1 Introduction

Transformer-Based Transfer Learning models for NLP – referred to henceforth as TBTL models for brevity – such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020) perform well on a variety of NLP tasks with minimal fine-tuning. However, prior to fine-tuning, TBTL models require a vast amount of data to train (Shoeybi et al., 2019). This training is only performed once, with users downloading and fine-tuning such language models to their specific task. In doing so, we are trusting large tech companies to train the base model responsibly since we have no control over this. This seems inherently undemocratic. We ideally want these models to be free from unwanted bias and whilst it is true that they exhibit less gender bias than static word embeddings (Sun et al., 2019), they are by no means immune to this problem (Lu et al., 2018).

As TBTL models become increasingly prevalent in our everyday lives, we want to avoid such prejudices influencing decision making. Examples where this is important include automatic resume filtering (Dastin, 2018) and criminal sentencing recommendations (Tashea, 2017).

In this paper we focus on the specific problem of gender bias, and analyse the extent to which it persists in modern TBTL models. We build upon Zhao et al. (2018a, 2019), in which quantification and mitigation of bias in ELMo was centre stage. In addressing this problem for more recent models, we aim to answer three main questions: i) How can we quantify bias in pre-trained language models? ii) How do different models compare in terms of bias? iii) How to mitigate bias in these models?

We believe that current gender bias metrics in the existing literature do not offer sufficient granularity to properly analyse this problem. Indeed, they mostly focus on measuring the assignment of stereotypical pronouns to professions (Zhao et al., 2018a). By focusing solely on this, they fail to address a model's overall preference for predicting male pronouns. An alternative bias which models can demonstrate is unequal preference towards male and female pronoun resolution across stereotypical and anti-stereotypical professions. We refer to these two forms of bias as *skew* and *stereotype*, respectively. In Section 3, we propose a new scheme to capture and quantify the important distinction between the two.

When comparing different TBTL models, we find evidence that gender skew and gender stereotype correlate approximately negatively with each other in out-of-the-box models, suggesting that a tradeoff between these two forms of bias may exist.

[†] Equal contribution.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2232–2242
April 19 - 23, 2021.

To mitigate bias in these models, we use the method proposed by Zhao et al. (2018a) to show that fine-tuning with augmented data, which references male and female entities with equal frequencies, can reduce professional gender stereotype and skew compared to fine-tuning on the original dataset. However, we show that gender prejudice may persist in forms other than professional bias, and these are ineffectively probed by current NLP benchmarks.

2 Related work

Bias quantification Early work in measuring gender bias specifically (Caliskan et al., 2017; May et al., 2019), along with efforts towards removing it either during (Zhao et al., 2018b) or after training (Bolukbasi et al., 2016), was done on static word embeddings such as GloVe and Word2Vec.

Caliskan et al. (2017) argue that completely removing undesirable bias using an automated procedure is impossible, as it is only distinguishable from the rules and structure of language itself by negative consequences in downstream applications. Instead, we should focus on probing and exposing which biases manifest themselves in which models, so that engineers can act accordingly. Choosing a suitable metric with which to analyse bias is a key challenge (May et al., 2019); whilst a positive result with respect to a suitable metric does reveal the existence of bias, a negative result does not mean a model is completely bias-free.

Statistical tests such as Word/Sentence Embedding Association Tests (WEAT/SEAT) have been developed to measure bias in static word embeddings using the cosine similarity of specific target words (Caliskan et al., 2017; May et al., 2019). However, when it comes to contextual embeddings, these traditional metrics have been shown to be ineffective at quantifying bias. In particular, Kurita et al. (2019) demonstrate that while WEAT tests are unable to identify any statistically significant bias in BERT, probing the underlying language model with a Gender Pronoun Resolution (GPR) task does reveal strong evidence that these non-static models also encode gender bias. Indeed, since contextual word embedding models such as BERT are optimised to capture the statistical properties of training data, they tend to pick up and amplify any social stereotypes that may be present (Kurita et al., 2019).

Having established GPR as a downstream task suitable for detecting gender bias, Zhao et al. (2018a) introduced a new benchmark, WinoBias, to measure bias in coreference pronoun resolution.

The dataset consists of two files, Test Set 1 and Test Set 2 (hereafter T1 and T2), representing two different gender pronoun resolution tasks. Each file consists of Wino-grad schema pairs of sentences involving a variety of occupations, differing only in one or two words and with a pronoun ambiguity that is resolved in opposite directions across the two sentences, giving both a pro- and anti-stereotypical resolution (Levesque et al., 2012). Example sentences are shown in Fig. 1 and Fig. 2.

Sun et al. (2019) consider a coreference resolution system unbiased on the WinoBias test if it achieves similar F1 scores for gender pronoun resolution on both the pro- and anti-stereotypical datasets whilst maintaining strong GPR performance. One of the main findings in Zhao et al. (2018a) is that three different coreference resolution architectures (rule based, feature-rich and neural-net based) built on top of static word embeddings all display significant disparity in F1 scores across the two datasets, with the F1 score for the pro-stereotypical dataset being on average 21.1 higher. This alarming observation was also discovered by Webster et al. (2018) and was attributed to the inherent bias of the underlying word embeddings (Bolukbasi et al., 2016), as well as the training of these coreference resolution pipelines on the OntoNotes 5.0 dataset (Weischedel et al., 2011) which is known to suffer from severe gender imbalance (Zhao et al., 2018a).

More recently, Zhao et al. (2019) investigated the existence of gender bias in the ELMo contextual embedding. Specifically, they note ELMo is trained on the Billion Word corpus (Chelba et al., 2013) which, just like OntoNotes 5.0, shows substantial imbalance in the counts of male vs. female pronouns. Training on this, ELMo then learns a language representation that reflects this gender inequality. To expose this, Zhao et al. (2019) analyse the behaviour of a coreference resolution system proposed by Lee et al. (2018) with ELMo contextual weights on the WinoBias benchmark, revealing a significant disparity in performance on the pro- and anti-stereotypical datasets. In fact, this disparity is 30% higher than a similar result based only on GloVe embeddings (Lee et al., 2017). This is particularly worrying; as commented earlier, contextual embeddings may, by construction, be amplifying

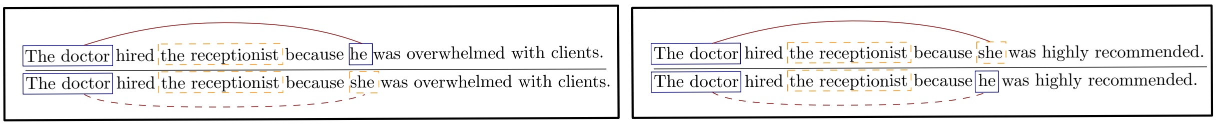

The doctor hired the receptionist because he was overwhelmed with clients.
The doctor hired the receptionist because she was overwhelmed with clients.
The doctor hired the receptionist because she was highly recommended.
The doctor hired the receptionist because he was highly recommended.

Figure 1: Example sentences from Test Set 1 (T1) of the WinoBias dataset. These take the form *[entity1] [interacts with] [entity2] [conjunction] [pronoun] [circumstances]*. Solid blue boxes indicate male entities, dashed orange boxes indicate female entities, solid purple lines indicate pro-stereotypical scenarios and dashed purple lines indicate anti-stereotypical scenarios. Such stereotypes are determined according to U.S. Bureau of Labor Statistics (2017).

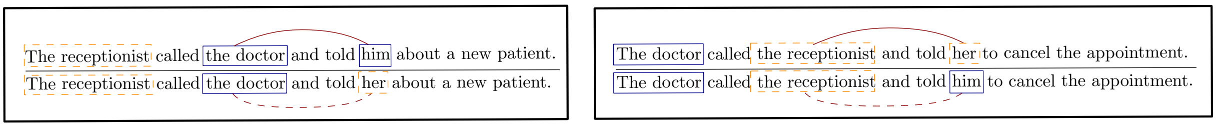

The receptionist called the doctor and told him about a new patient.
The receptionist called the doctor and told her about a new patient.
The doctor called the receptionist and told her to cancel the appointment.
The doctor called the receptionist and told him to cancel the appointment.

Figure 2: Example sentences from Test Set 2 (T2) of the WinoBias dataset. These take the form *[entity1] [interacts with] [entity2] and then [interacts with] [pronoun] for [circumstances]*. The style format follows that of Fig. 1. Image adapted from Zhao et al. (2018a).

undesirable statistical artefacts of the dataset more than their static counterparts. Therefore, it is of the utmost importance to perform a similar analysis on recent TBTL models.

Devlin et al. (2018) explicitly state that BERT is not trained on the Billion Word corpus, since this only provides examples of isolated sentences and the authors preferred to use a document-level corpus to get contiguous training data, allowing richer contexts to be learnt.

Specifically, BERT is trained using the BookCorpus dataset (Zhu et al., 2015) as well as English Wikipedia. However, the BookCorpus data has since been shown to suffer from similar gender imbalance problems (Tan and Celis, 2019) as has English Wikipedia where, for example, only 15.5% of the biographies are of women (Wagner et al., 2016). We believe that this imbalance is the principle cause of skew in the model.

Bias Mitigation As discussed above, whilst we cannot completely remove bias from a model, research into bias mitigation is still a very worthwhile pursuit and, in the context of the WinoBias metric of occupational gender bias, could help break the glass ceiling.

Many bias mitigation methods for static embeddings centre around modifying the vector space and/or loss function during the training process. Initial attempts sought to project biased embedding vectors back to a gender neutral subspace (Bolukbasi et al., 2016). Subsequent improvements came from adding a regularisation term to the training loss function designed to encourage specific gendered words to separate, thus allowing the remaining neutral terms to mix (Zhao et al., 2018b). However, these offer superficial reductions in gender bias, and systematic prejudice was found to persist (Gonen and Goldberg, 2019).

Attempts to mitigate gender bias in contextualised embeddings are a more novel endeavour. These attempts typically involve fine-tuning models to a particular task and one proposal involves duplicating the training corpus and switching gender-specific terms in the duplicated data. For example, “The King cemented his rule over his lords” is substituted with “The Queen cemented her rule over her ladies”. This method, referred to as Data Augmentation, was demonstrated to successfully reduce gender bias in ELMo for pronoun resolution tasks, relative to a model trained on the unaugmented training data (Zhao et al., 2019).

3 Method

3.1 Analysing Bias in WinoBias

Bias in TBTL models can be measured using either T1 or T2 from the WinoBias dataset – see Fig. 1 and Fig. 2, respectively. Within each test set, examples are composed of a pro-stereotypical and anti-stereotypical sentence, where stereotype is determined by professional gender imbalances recorded by U.S. Bureau of Labor Statistics (2017). Our approach is to take each WinoBias sentence, mask the pronoun of interest, and then compare the language model's prediction for the masked token with the pro- and anti-stereotypical labels. To predict the gender of the pronoun in the sentence “*The physician hired the secretary because [MASK] was overwhelmed with clients*” we calculate the proba-

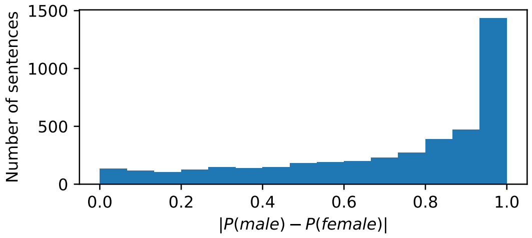


Figure 3: Histogram of absolute differences in gendered pronoun assignment, $|P(\text{male}) - P(\text{female})|$. Sentence examples where this difference was smaller than 0.1 were removed so that only pronouns assigned with a high degree of certainty were analysed.

bilities of the pro- and anti-stereotypical pronouns – "he" and "she" respectively – and pick the one with the highest likelihood. Note that this approach risks obscuring the confidence in pronoun resolution. For example, $P(\text{male}) = 0.99$ and $P(\text{male}) = 0.51$ would both result in male pronoun assignment. The histogram in Fig. 3 demonstrates that this issue does not affect our experiments; the distribution is highly negatively skewed. The majority of pronouns are resolved with a high degree of confidence. We chose $|P(\text{male}) - P(\text{female})| \geq 0.1$ as an arbitrary cutoff bound to select sentences which were resolved with a high degree of certainty. Only sentences which fulfil this criteria were analysed in the following experiments.

In line with the academic literature, we compute F1 scores for both the pro- and anti-stereotypical data using contextual language models. This approach to coreference resolution is demonstrably well-founded and F1 results from the GPR baseline are discussed in Section 4.1.

The WinoBias sentences have been constructed so that, in the absence of professional stereotypes, there is no objective way to choose between different gender pronouns. The difference in F1 scores with respect to gender g, across a pro/anti test set, $\text{F1}^{g}_{\text{pro}} - \text{F1}^{g}_{\text{anti}}$, is a metric inspired by previous papers to measure a model's tendency to assign that gender to professions, with positive (resp. negative) values indicating a pro- (resp. anti-) stereotypical assignment (Sun et al., 2019). We refer to it as a measure of gender *stereotype*. In contrast to the literature, we compute F1 scores with respect to both "male" and "female" true labels allowing us to define *stereotype* with respect to both genders.

We now propose to also use the difference in F1 scores with respect to a dataset $\mathcal{D}$, across genders, $\text{F1}^{♂}_{\mathcal{D}} - \text{F1}^{♀}_{\mathcal{D}}$, as a measure of gender *skew* in dataset $\mathcal{D}$, with positive (resp. negative) values capturing the tendency of a model to generally assign a male (resp. female) gender to any given profession. This distinction is important: consider a classifier which only assigns male pronouns to professions. It would not be stereotyping professions to perceived gender roles, but would be heavily biased in assuming a general male dominance in the workplace. Both these forms of gender unfairness are considered in the subsequent analysis and we use the mean skew and stereotype, taken across datasets and genders respectively as shown below:

$$\mu_{\text{Skew}} \triangleq \frac{1}{2}\left(\left|\text{F1}^{♂}_{\text{pro}} - \text{F1}^{♀}_{\text{pro}}\right| + \left|\text{F1}^{♂}_{\text{anti}} - \text{F1}^{♀}_{\text{anti}}\right|\right)$$

$$\mu_{\text{Stereo}} \triangleq \frac{1}{2}\left(\left|\text{F1}^{♂}_{\text{pro}} - \text{F1}^{♂}_{\text{anti}}\right| + \left|\text{F1}^{♀}_{\text{pro}} - \text{F1}^{♀}_{\text{anti}}\right|\right)$$

where $\text{F1}^{♂}_{\text{pro}}$ denotes the F1 score on the pro-stereotypical dataset whilst considering the male pronoun as the *true* label. To be completely gender neutral, we average the absolute values since we are only interested in the extent of gender bias rather than its direction.

3.2 Online Skewness Mitigation

As we will show in Section 4.2, most current TBTL models models are inherently skewed towards predicting male pronouns. Inspired by Kurita et al. (2019), we propose a simple approach to reducing this skew. We normalise the probability of a masked pronoun being assigned a particular gender in a certain occupational context by dividing through with the prior probability of choosing that pronoun in a sentence with the same structure but without any occupational context.

We illustrate this method with the sentence "The physician hired the secretary because he was overwhelmed with clients". This method starts by calculating the probabilities of "he" and "she" in the standard way, as described in Section 3.1. Next, we mask the professions, leading to "[MASK] hired [MASK] because [MASK] was overwhelmed with clients" and calculate the probability of the third masked word being "he" and "she" in this context. Finally, we normalise by dividing the probabilities found using the standard method, with the probabilities found using the masked-professions context. This method assumes language models can resolve the pronoun when both professions are masked.

Models mitigating skew using this approach are given the suffix *-O* in the remainder of this paper.

3.3 Bias Removal via Data Augmentation

We aim to replicate the Data Augmentation method proposed in Zhao et al. (2019) for mitigating gender bias in ELMo. The goal of this approach is to use an augmented dataset to fine-tune the pre-trained language model to the GPR task. In particular, this augmented dataset is designed with the intention of neutralising the gender bias already present in a model such as BERT, whilst simultaneously avoiding the corruption of its understanding of natural language.

As in Zhao et al. (2019), a target GPR task was constructed by first selecting sentences from OntoNotes 5.0 containing gendered pronouns and masking them accordingly; the BERT masked language model will be trained to predict the masked pronoun. Secondly, we anonymise the data by replacing all gendered names with identity tokens such as [E1] and [E2].

Each training example is then *augmented* by replacing all possessive and personal pronouns with those of the opposite gender. Additionally we apply a mapping of explicitly gendered words (such as "Man"→"Woman" and vice-versa) to ensure that the text remains linguistically coherent in the context of reversed genders.[1] Following this approach, the sentence "*The King was pleased that his Lords had vanquished their enemies*" would be augmented to "*The Queen was pleased that her Ladies had vanquished their enemies*".

To examine the effects of data augmentation, we then fine-tune two BERT models. The first was fine-tuned on the un-augmented OntoNotes training examples, whilst the second was fine-tuned on the augmented OntoNotes examples (containing the duplicated and gender-switched examples also). Hereafter we shall refer to these models as BERT-U and BERT-A respectively. In both cases, a hyperparameter search over the epochs and learning rate was conducted.[2] The best performing un-augmented/augmented models were tested using the WinoBias data as described in Section 3.1.

4 Results and Discussion

4.1 Baseline: Alice and Bob

There is a risk that removing bias deteriorates the predictive power of the model. We measure a baseline performance on a GPR task to test how well the model is able to actually resolve the pronoun to the correct entity. To assess this we modify the WinoBias data set by replacing the professions with unambiguously gendered names, *Alice* and *Bob*. Section 4.1 illustrates that we can achieve high F1 scores on this modified WinoBias dataset, validating the use of masked language models for GPR tasks. However, we note that ALBERT and XLM-RoBERTa perform particularly poorly on both T1 and T2 tasks.

Model	**T1**		**T2**	
	F1♂	**F1♀**	**F1♂**	**F1♀**
RoBERTa	64.2	72.2	92.7	93.0
RoBERTa-O	50.2	71.7	89.8	88.7
RoBERTa-large	78.2	79.5	94.9	94.9
RoBERTa-large-O	78.2	79.5	89.5	87.3
ALBERT	39.2	68.6	58.7	75.3
ALBERT-O	6.2	67.6	62.9	24.1
ALBERT-large	61.0	71.2	19.2	68.6
ALBERT-large-O	60.6	68.9	31.2	70.1
ALBERT-xlarge	64.1	75.1	23.3	69.4
ALBERT-xlarge-O	69.2	74.2	59.4	76.6
ALBERT-xxlarge	64.2	76.9	95.0	95.3
ALBERT-xxlarge-O	78.2	80.5	89.0	90.5
BERT	58.8	62.4	95.3	95.5
BERT-O	59.0	64.7	95.1	95.1
BERT-large	72.6	74.9	95.3	95.5
BERT-large-O	69.8	75.1	95.6	95.7
XLM-RoBERTa	29.7	69.0	64.5	76.7
XLM-RoBERTa-O	52.1	65.1	64.4	37.9
XLM-RoBERTa-large	62.8	76.4	21.7	69.2
XLM-RoBERTa-large-O	64.9	77.0	80.1	84.9
DistilBERT	41.2	66.4	81.0	79.0
DistilBERT-O	50.2	66.1	81.0	78.5
BERT-U	76.6	65.8	90.1	88.6
BERT-UO	78.8	77.3	93.1	92.3
BERT-A	75.6	67.4	91.8	90.0
BERT-AO	75.7	67.5	74.9	54.2

Table 1: F1 (in %) performance of different models on WinoBias dataset, where professions are replaced by gendered names and a pronoun is correct if it refers to the correct name. The insertion of gendered names implies that there is now a correct pronoun, in contrast to the original WinoBias data set where there is merely a stereotypical pronoun. The suffixes O, U and A refer to Online, Un-augmented and Augmented mitigation approaches respectively.

The Online Skewness Mitigation described in Section 3.2 demonstrates no discernable pattern on the F1 scores. This suggests that it does not negatively effect GPR performance. Neither is there

---

[1] Mapping sourced from (Zhao et al., 2018a). See the Glove and WinoBias GitHub pages.

[2] Specific settings detailed in Appendix A.

a definite pattern in the skew, indicating that skew is not necessarily reduced in the presence of unambiguously gendered entities.

The decreased performance of BERT-A/U (as described in Section 3.3) relative to the out-of-the-box BERT model may be caused by the anonymisation of the OntoNotes data the models were fine-tuned on, making them less receptive to performing GPR with common names.

The F1 scores in Section 4.1 demonstrate that GPR in T1 is significantly more challenging than in T2. Figures 1 and 2 show that pronoun resolution in T2 is always with respect to *[entity2]* whilst in T1, the pronoun resolution can be with respect to either *[entity1]* or *[entity2]*. In T2 it is clearer to contextual models which entity is the object of the sentence. Thus, we can use a model's gendered pronoun predictions on T2 sentences to expose any internal bias it may have toward *[entity2]*. In T1, the lack of syntactic cues make it unclear which entity is the sentence's object; as such, we may be unable to isolate the model bias corresponding to each specific entity.[3] For these reasons, we argue that T2 is better at revealing the biases encoded in these models. *Hence we will use T2 from this point onward.*

To expose bias, we require models with reliable coreference resolution performance on T2, demonstrated by consistently high F1 scores. In subsequent sections, we investigate all BERT, RoBERTa, and DistilBERT models as well as ALBERT-xxlarge and XLM-RoBERTa-large-O, all of which have F1 scores on T2 greater than our arbitrarily defined threshold of 75%. All of the other models in Section 4.1 fall below this threshold and so are not considered further in this paper.

4.2 WinoBias Performance

In Section 4.2 we present the F1 scores achieved on WinoBias T2 by different models. We note that the pro-stereotypical F1 scores are lower than the gendered names baseline of Section 4.1. This is to be expected, since the Alice and Bob system discussed in Section 4.1 can be understood as being unambiguous and completely biased. Consequently, whereas Section 4.1 shows just GPR performance and general skew bias, Section 4.2 quantifies GPR performance, skew and stereotype.

Embedding	**T2**					
	F1♂		**F1♀**		**Bias**	
	Pro	Anti	Pro	Anti	Stereo	Skew
RoBERTa	62.9	27.0	69.0	39.3	32.8	9.2
RoBERTa-O	68.0	60.2	26.5	8.5	12.9	46.6
RoBERTa-large	67.0	52.4	45.0	21.5	26.4	24.0
RoBERTa-large-O	66.0	65.0	11.2	7.5	2.3	56.1
ALBERT-xxlarge	71.4	46.9	54.8	16.4	31.4	23.5
ALBERT-xxlarge-O	69.7	49.2	51.5	18.1	27.0	24.6
DistilBERT	64.9	67.2	4.8	5.0	1.3	61.2
DistilBERT-O	64.5	65.8	10.0	12.8	2.1	53.8
BERT	69.3	58.0	31.4	8.2	17.3	43.8
BERT-O	68.4	58.1	32.9	10.5	16.4	41.6
BERT-large	70.0	57.9	33.9	2.8	21.6	45.6
BERT-large-O	69.9	57.9	32.5	5.0	19.7	45.1
XLM-RoBERTa-large-O	68.0	56.0	38.2	15.7	17.2	35.1
BERT-U	67.8	63.4	14.1	2.8	57.2	7.9
BERT-UO	67.3	60.4	26.6	11.6	44.8	11.0
BERT-A	64.7	64.5	14.8	14.8	49.8	0.1
BERT-AO	65.0	63.6	17.9	15.3	47.7	2.0

Table 2: F1 results (in %) from Test Set 2. The suffixes O, U and A refer to Online, Unaugmented and Augmented bias mitigation approaches respectively.

Note that all models show significant male skew, except for RoBERTa which demonstrates higher F1♀ than F1♂ scores on both pro- and anti-stereotypical examples. Indeed for all other models there is a noticeable increase in male skew compared to the Alice & Bob results in Section 4.1. The only experimental difference is the use of occupations rather than names, demonstrating that it is specifically the professions that push the model to predicting male pronouns.

Focusing on the out-of-the-box models, we rank them by their gender skew from best to worst as RoBERTa, ALBERT-xxlarge, RoBERTa-large, BERT, BERT-large, and DistilBERT.

We note that RoBERTa has the least skew bias, with a μ_{skew} value of 9.2% for T2. Liu et al. (2019) report that BERT was "significantly undertrained", and aimed to address this by training RoBERTa for longer, with bigger batches and sequences, additional data, and dynamic adjustments to the masking pattern. These amendments in RoBERTa appear to have reduced the skew bias in the model, suggesting that a model's training procedure can have a considerable impact on its skew. We also observed that within the BERT and RoBERTa families, larger models tend to show more skew than their smaller counterparts.

The high skew of DistilBERT might be due to its student-teacher training (Hinton et al., 2015). This lends itself to a overly simplistic understanding of male and female roles within society. Understanding the subtleties and nuances of gender roles

---

[3] This also explains the larger skew on T1 compared to T2: the lower GPR performance on T1 leads the model to guess more and as we will see in Section 4.2, prefers to guess male.

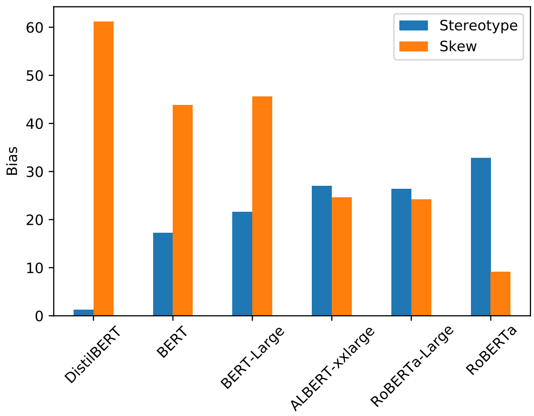


Figure 4: Bar chart showing the different bias measurements for the out-of-the-box models investigated. This suggests an inherent trade-off between skew and stereotype in language models.

requires models with high representation capacity and training DistilBERT to mimic BERT's output renders it incapable of making such distinctions.

The ranking of gender stereotype from best to worst is DistilBERT, BERT, BERT-large, RoBERTa-large, ALBERT-xxlarge, and RoBERTa.

Note that this order is approximately the opposite to skew, as illustrated in Fig. 4. There appears to be a potential trade off between the skew and stereotype in out-of-the-box language models, with RoBERTa-large best balancing the two biases. This trend appears to carry forward to the fine-tuned models, with BERT-A and BERT-U showing high stereotype but very low skew.

4.3 Mitigated Systems

4.3.1 Online Skewness Mitigation

Comparing how bias values change in Section 4.2 when going from all models to their normalised - O version, Online Skewness Mitigation successfully reduces stereotype for 6/8 models, though interestingly RoBERTa responded by going from a female stereotype to a significant male stereotype. At the same time, we observe that this reduction in model stereotype actually comes at the expense of model skew. Note that this effect is the opposite to what it was designed for. For this reason we do not explore it further.

4.3.2 Data Augmentation

BERT-U and BERT-A test accuracies on the OntoNotes data were 64% and 58% respectively. This 6% loss in accuracy should be kept in mind when assessing gender bias. We see BERT-U/A have similar F1 values in Section 4.1 indicating that any difference in bias is not just noise coming from them having radically different abilities to perform the task.

Section 4.2 shows BERT-A has a 7.4% lower stereotype and a 7.8% lower skew than BERT-U, echoing the findings of Zhao et al. (2019) which demonstrate reductions of bias in ELMo through fine-tuning with data augmentation. The skew of BERT-A is reduced to 0.1%; much lower than standard BERT. This elimination of skew is unsurprising since augmentation ensures professional entities assume male and female roles equally in the fine-tuning data. Despite this, the stereotype of BERT-A is greater than standard BERT, suggesting a degree of non-skew bias has been introduced by fine-tuning. An analysis of the individual test examples in WinoBias T2 may reveal why this is the case. Across the classifications of sentences we see in the data, pronoun resolution appears to be influenced by the implied characteristics of the individual whose pronouns we seek to resolve, rather than the professions themselves.

4.4 Impact of Sentence Context on Pronoun Resolution

We consider an example from WinoBias T2 with the *[entity2]* profession being "guard". The only example where BERT-A assigned a female pronoun to the profession "guard" was a sentence where the guard was associated with low competency: "*The clerk works harder than the guard and gets more appreciation than [her]*". To investigate this more quantitatively, we measure the competency of the entity whose pronoun is being resolved. Our aim is to identify whether fine-tuning results in a gender shift across competent and incompetent examples which might suggest a competency-based gender assignment.

To achieve this, four of the authors labelled the WinoBias examples according to whether they thought the subject of the sentence demonstrated competent or skilful behaviour (e.g. getting a promotion, being thanked for a job well done) or not (e.g. getting sacked from their job). The sentences were classified as Incompetent, Neutral (no competency suggested in the sentence), or Competent. All voters classified sentences independently, with a Fleiss' κ score of 0.42. The class assigned to each sentence was then determined by a majority vote.

Competency	**BERT**	**BERT-U**	**BERT-A**
Incompetent	0.156	0.062	0.281
Neutral	0.117	0.168	0.280
Competent	0.160	0.140	0.140

Table 3: Proportion of female pronouns assigned for each competency category for the WinoBias T2 sentences. All professional entities in the examples were replaced with the gender neutral term "person" to isolate the impact of competency from professional stereotype.

Sentences that resulted in a tie were discarded.[4] To isolate our investigation of subject competency from professional stereotype, all professions in the WinoBias dataset were replaced with the gender agnostic term "person".

Table 3 reports the proportion of examples in each competency class that were assigned a female pronoun across BERT, BERT-U, and BERT-A. The proportions of female pronoun assignments show that BERT-A allocates a more balanced ratio of gendered pronouns to Incompetent examples compared to BERT and BERT-U. Apart from the Competent class (which shows no major change across all three models), BERT-A reduces the gender imbalance of pronouns in Neutral and Incompetent examples.

It is challenging to exactly determine the cause of these observations, but it certainly appears that fine-tuning BERT models has an effect on the gender ratios in each competency class. It is notable that de-biasing BERT reduced the gender imbalance of Incompetent examples by a large margin. These findings merit further investigation.

We believe that WinoBias and other related benchmarks do not sufficiently probe professional gender bias, as pronoun resolution may be obfuscated by cross-correlations from other manifestations of gender prejudice. One example of a bias other than profession and competency could be personality bias, where women may be more closely associated with passive and caring traits whilst men may be more aggressive and disagreeable. We encourage the development of a dataset that isolates these different gender biases, allowing us to probe them without interference from one another.

5 Conclusion and Future Work

Quantifying gender bias in coreference resolution is challenging, since co-referencing performance and bias manifestation are closely linked. We have proposed *skew* and *stereotype* as new measures of gender bias, allowing us to better probe model prejudice.

We have shown that there is an approximate trade-off between the skew and stereotype of out-of-the-box models. DistilBERT and BERT models have high skew and low stereotype whilst RoBERTa and ALBERT-xxlarge have reduced skew at the cost of higher stereotype.

Two methods have been proposed to mitigate bias: Online Skewness Mitigation and Data Augmentation. The online approach has been shown to be effective at mitigating stereotype at the expense of skew, demonstrating the opposite effect to what it was designed for. We took the Data Augmentation method proposed by Zhao et al. (2019) for debiasing ELMo and extended it to BERT, demonstrating that it reduces both forms of gender bias compared to unaugmented fine-tuned models. However, the reduction of explicit professional gender skew and stereotype reveal the model's underlying bias towards gender competency. We successfully expose these using WinoBias GPR sentence probes labelled for competency.

Since contextual language models consider the full sentence contents when assigning a pronoun, we believe that the WinoBias data used in this paper does not purely measure professional biases. A second popular dataset taken from the SuperGLUE benchmarks, *Winogender* (Rudinger et al., 2018), is increasingly used for evaluating a model's gender bias. However, its limited size relative to WinoBias makes it less robust and hence it was not used in this paper.

We observed that language models may also consider other stereotyped gender characteristics in the sentence when classifying pronouns. Given the above considerations, we believe that a more comprehensive set of gender bias benchmarks should be developed which can better isolate specific biases within models.

Kiritchenko and Mohammad (2018) have shown that both race and gender bias are prevalent in a large proportion of state-of-the-art language models. Recently, a number of other datasets have appeared for detecting these and other kinds of bias such as age and religion (Nadeem et al., 2020;

---

[4] Our competency dataset is available at GitHub.

Nangia et al., 2020). It would be interesting to see if competency bias obscures analyses on these datasets similarly.

Future research is recommended on how data augmentation affects other models from the BERT family. Additionally, it will be valuable to explore whether Data Augmentation could be applied to larger corpora to train a new contextual model from scratch.

Lastly, in contrast to static embeddings, it is notoriously hard, if not impossible, to define bias in contextual embeddings (Caliskan et al., 2017). It is likely that without extensive research and transparent communication, the field of NLP will be further scrutinised as more applications are found to exhibit undesired biases. Discussions, both within and outside the community, are required to determine what separates bias from semantic assumptions, allowing bias disclaimers and guidelines to be provided to downstream developers.

Acknowledgements

We thank the anonymous reviewers for the insightful comments, and Hadas Orgad for helping us correcting a mistake in a previous version of this paper. This research was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 875160.

References

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in Neural Information Processing Systems*, pages 4349–4357.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005.

Jeffrey Dastin. 2018. Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. *Association for Computational Linguistics*, pages 609–614.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *Advances in Neural Information Processing Systems*.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. *Association for Computational Linguistics*, pages 43–53.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. *Association for Computational Linguistics*, pages 166–172.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. *ICLR*.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. *CoRR*, abs/1707.07045.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. *CoRR*, abs/1804.05392.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd Schema Challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pages 552–561. AAAI Press, Rome, Italy.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2018. Gender bias in neural natural language processing. *CoRR*, abs/1807.11714.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *Association for Computational Linguistics*, pages 622–628.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint: 2004.09456*.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. *Empirical Methods in Natural Language Processing*, pages 1953–1967.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. *Association for Computational Linguistics*, pages 8–14.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. *Association for Computational Linguistics*, pages 1630–1640.

Yi Chern Tan and L Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. *Advances in Neural Information Processing Systems*, pages 13230–13241.

Jason Tashea. 2017. Courts are using ai to sentence criminals. that must stop now. *WIRED*.

U.S. Bureau of Labor Statistics. 2017. Labor force statistics from the current population survey, 2017. `https://www.bls.gov/cps/cpsaat11.htm`.

Claudia Wagner, Eduardo Graells-Garrido, David Garcia, and Filippo Menczer. 2016. Women through the glass ceiling: gender asymmetries in wikipedia. *EPJ Data Science*, 5(1).

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the gap: A balanced corpus of gendered ambiguous pronouns. *Association for Computational Linguistics*, 6:605–618.

Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. *OntoNotes: A Large Training Corpus for Enhanced Processing*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. *Association for Computational Linguistics*, pages 629–634.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018a. Gender bias in coreference resolution: Evaluation and debiasing methods. *Association for Computational Linguistics*, pages 15–20.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018b. Learning gender-neutral word embeddings. *Association for Computational Linguistics*, pages 4847–4853.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724.

A Fine-tuning Training Parameters

For fine-tuning BERT on the OntoNotes data, the following settings were used. Standard hyperparameter choices of $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, and a dropout probability of 0.1 were chosen. Model training and validating with a 80/20 train/test split of the training data, across training epochs $\in \{1, \ldots, 10\}$. The selected (epoch) model was that with the highest pronoun prediction accuracy on the validation set.

On the Evolution of Syntactic Information Encoded by BERT's Contextualized Representations

Laura Pérez-Mayos[1], Roberto Carlini[1], Miguel Ballesteros[2], Leo Wanner[3,1]
[1] TALN Research Group, Pompeu Fabra University, Barcelona, Spain
[2] Amazon AI
[3] Catalan Institute for Research and Advanced Studies (ICREA), Barcelona, Spain
`{laura.perezm|roberto.carlini|leo.wanner}@upf.edu`
`ballemig@amazon.com`

Abstract

The adaptation of pretrained language models to solve supervised tasks has become a baseline in NLP, and many recent works have focused on studying how linguistic information is encoded in the pretrained sentence representations. Among other information, it has been shown that entire syntax trees are implicitly embedded in the geometry of such models. As these models are often fine-tuned, it becomes increasingly important to understand how the encoded knowledge evolves along the fine-tuning. In this paper, we analyze the evolution of the embedded syntax trees along the fine-tuning process of BERT for six different tasks, covering all levels of the linguistic structure. Experimental results show that the encoded syntactic information is forgotten (PoS tagging), reinforced (dependency and constituency parsing) or preserved (semantics-related tasks) in different ways along the fine-tuning process depending on the task.

1 Introduction

Adapting unsupervised pretrained language models (LMs) to solve supervised tasks has become a widely spread practice in NLP, with models such as ELMo (Peters et al., 2018) and, most notably, BERT (Devlin et al., 2019), achieving state-of-the-art results in many well-known Natural Language Understanding benchmarks like GLUE (Wang et al., 2018) and SQuAD (Rajpurkar et al., 2018). Several studies investigate what the LMs learn, how and where the learned knowledge is represented and what the best methods to improve it are; cf., e.g., (Rogers et al., 2020). There is evidence that, among other information (such as, e.g., PoS, syntactic chunks and roles (Tenney et al., 2019b; Lin et al., 2019; Belinkov et al., 2017), morphology in general (Peters et al., 2018), or sentence length (Adi et al., 2016)) BERT deep models' vector geometry implicitly embeds entire syntax trees (Hewitt and Manning, 2019). However, rather little is understood about how these representations change when fine-tuned to solve downstream tasks (Peters et al., 2019).

In this work, we aim to understand how syntax trees implicitly embedded in the geometry of deep models evolve along the fine-tuning process of BERT on different supervised tasks, and shed some light on the importance of the syntactic information for those tasks. Intuitively, we expect morpho-syntactic tasks to clearly reinforce the encoded syntactic information, while tasks that are not explicitly syntactic in nature should maintain it in case they benefit from syntax (Kuncoro et al., 2020) and lose it if they do not. In order to cover the three main levels of the linguistic description (morphology, syntax and semantics), we select six different tasks: PoS tagging, constituency parsing, syntactic dependency parsing, semantic role labeling (SRL), question answering (QA) and paraphrase identification. The first three inherently deal with (morpho-)syntactic information while the latter three, which traditionally draw upon the output of syntactic parsing (Carreras and Màrquez, 2005; Björkelund et al., 2010; Strubell et al., 2018; Wang et al., 2019, inter-alia), deal with higher level, semantic information. Almost all of our experiments are on English corpora; one is on multilingual dependency parsing.

2 Related work

BERT has become the default baseline in NLP, and consequently, numerous studies analyze its linguistic capabilities in general (Rogers et al., 2020; Henderson, 2020), and its syntactic capabilities in particular (Linzen and Baroni, 2020). Even if syntactic information is distributed across all layers (Durrani et al., 2020), BERT captures most phrase-level information in the lower layers, followed by surface

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2243–2258
April 19 - 23, 2021.

features, syntactic features and semantic features in the intermediate and top layers (Jawahar et al., 2019; Tenney et al., 2019a; Hewitt and Manning, 2019). The syntactic structure captured by BERT adheres to that of the Universal Dependencies (Kulmizev et al., 2020); different syntactic and semantic relations are captured by self-attention patterns (Kovaleva et al., 2019; Limisiewicz et al., 2020; Ravishankar et al., 2021), and it has been shown that full dependency trees can be decoded from single attention heads (Ravishankar et al., 2021). BERT performs remarkably well on subject-verb agreement (Goldberg, 2019), and is able to do full parsing relying only on pretraining architectures and no decoding (Vilares et al., 2020), surpassing existing sequence labeling parsers on the Penn Treebank dataset (De Marneffe et al., 2006) and on the end-to-end Universal Dependencies Corpus for English (Silveira et al., 2014). It can generally also distinguish good from bad completions and robustly retrieves noun hypernyms, but shows insensitivity to the contextual impacts of negation (Ettinger, 2020).

Different supervised probing models have been used to test for the presence of a wide range of linguistic phenomena in the BERT model (Conneau et al., 2018; Liu et al., 2019; Tenney et al., 2019b; Voita and Titov, 2020; Elazar et al., 2020). Hewitt and Manning (2019)'s structural probe shows that entire syntax trees are embedded implicitly in BERT's vector geometry. Extending their work, Chi et al. (2020) show that multilingual BERT recovers syntactic tree distances in languages other than English and learns representations of syntactic dependency labels.

Regarding how fine-tuning affects the representations of BERT, Gauthier and Levy (2019) found a significant divergence between the final representations of models fine-tuned on different tasks when using the structural probe of Hewitt and Manning (2019), while Merchant et al. (2020) concluded that fine-tuning is conservative and does not lead to catastrophic forgetting of linguistic phenomena – which our experiments do not confirm. However, we find that the encoded syntactic information is forgotten, reinforced or preserved differently along the fine-tuning process depending on the task.

3 Experimental setup

We study the evolution of the syntactic structures discovered during pretraining along the fine-tuning of BERT-base (cased) (Devlin et al., 2019)[1] on six different tasks, drawing upon the structural probe of Hewitt and Manning (2019).[2] We fine-tune the whole model on each task outlined below for 3 epochs, with a learning rate of $5e^{-5}$, saving 10 evenly-spaced checkpoints per epoch. The output of the last layer is used as input representation for the classification components of each task. To mitigate the variance in performance induced by weight initialization and training data order (Dodge et al., 2020), we repeat this process 5 times per task with different random seeds and average results.

PoS tagging. We fine-tune BERT with a linear layer on top of the hidden-states output for token classification.[3] Dataset: Universal Dependencies Corpus for English (UD 2.5 EN EWT Silveira et al. (2014)).

Constituency parsing. Following Vilares et al. (2020), we cast constituency parsing as a sequence labeling problem, and use a single feed-forward layer on top of BERT to directly map word vectors to labels that encode a linearized tree. Dataset: Penn Treebank (Marcus et al., 1993).

Dependency parsing. We fine-tune a Deep Biaffine neural dependency parser (Dozat and Manning, 2016) on three different datasets: i) UD 2.5 English EWT (Silveira et al., 2014); ii) a multilingual benchmark generated by concatenating the UD 2.5 standard data splits for German, English, Spanish, French, Italian, Portuguese, and Swedish (Zeman et al., 2019), with gold PoS tags; iii) PTB SD 3.3.0 (De Marneffe et al., 2006).

Semantic role labeling. Following Shi and Lin (2019), we decompose the task into i) predicate sense disambiguation and argument identification, and ii) classification. Both subtasks are casted as sequence labeling, feeding the contextual representations into a one-hidden-layer MLP for the first, and a one-layer BiLSTM followed by a one-hidden-layer MLP for the latter. Dataset: OntoNotes corpus (Weischedel et al., 2013).

Question answering. We fine-tune BERT with

---

[1] Our experiments are implemented in PyTorch, using two open-source libraries: the Transformers library (Wolf et al., 2019) and AllenNLP (Gardner et al., 2017). Implementation details, pretrained weights and full hyperparameter values can be found in the libraries documentation.

[2] We use the same experimental setup used by the authors. Source: `https://github.com/john-hewitt/structural-probes`

[3] Source: `https://github.com/Tarpelite/UniNLP/blob/master/examples/run_pos.py`

a linear layer on top of the hidden-states output to compute span start logits and span end logits.[4] Dataset: Stanford Question Answering Dataset (SQuAD (Rajpurkar et al., 2018)).

Paraphrase identification. We fine-tune BERT with a linear layer on top of the pooled sentence representation.[5] Dataset: Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005).

4 Evolution of syntax trees

Hewitt and Manning (2019)'s structural probe evaluates how well syntax trees are embedded in a linear transformation of the network representation space, performing two different evaluations: i) Tree distance evaluation, in which squared L2 distance encodes the distance between words in the parse tree, and ii) Tree depth evaluation, in which squared L2 norm encodes the depth of the parse tree.

Using their probe, Hewitt and Manning show that the 7th layer of BERT-base is the layer that encodes more syntactic information. Therefore, to analyze the evolution of the encoded syntax trees, we train the probes on the 7th layer of the different checkpoint models generated along the fine-tuning process of each task.[6]

4.1 Tree distance evaluation

The probe evaluates how well the predicted distances between all pairs of words in a model reconstruct gold parse trees by computing the Undirected Unlabeled Attachment Score (*UUAS*). It also computes the Spearman correlation between true and predicted distances for each word in each sentence, averaging across all sentences with lengths between 5 and 50 (henceforth referred to as *DSpr.*).

Morpho-syntactic tasks As shown in Figures 1 and 2, both metrics follow a similar behaviour (shades represent the variability across the 5 model runs). PoS tagging shows an important loss of performance all along the fine-tuning process, especially noticeable for UUAS (Figure 1), suggesting that distance-related syntactic information is of less relevance to PoS tagging than could be intuitively assumed. As many words have a clear preference towards a specific PoS, especially in English, and most of the ambiguous cases can be resolved using information in the close vicinity (e.g., a simple 3-gram sequence tagger is able to achieve a very high accuracy (Manning, 2011)), syntactic structure information may not be necessary and, therefore, the model does not preserve it. This observation is aligned with Pimentel et al. (2020), who found that PoS-tagging is not an ideal task for contemplating the syntax contained in contextual word embeddings. The loss is less pronounced on depth-related metrics, maybe because the root of the sentence usually corresponds to the verb, which may also help in identifying the PoS of surrounding words.

Constituency parsing and dependency parsing share a very similar tendency, with a big improvement in the first fine-tuning steps preserved along the rest of the process. As both tasks heavily rely on syntactic information, this improvement intuitively makes sense. Dependency parsing fine-tuned on the Penn Treebank (PTB) shows even higher results since the probing is trained on the same dataset. In-

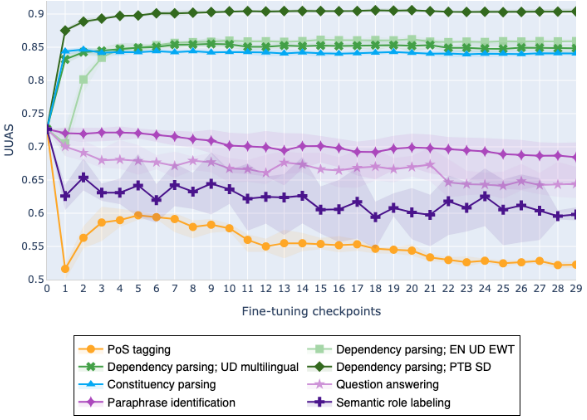


Figure 1: Tree distance evaluation. *UUAS* evolution.

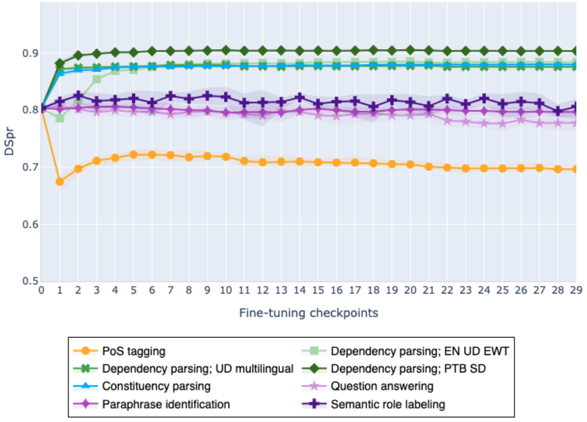


Figure 2: Tree distance evaluation. *Dspr* evolution.

---

[4] Source: `https://github.com/huggingface/transformers/tree/master/examples/question-answering`.

[5] Source: `https://github.com/huggingface/transformers/blob/master/examples/text-classification/run_glue.py`.

[6] Cf. also Supplementary Material.

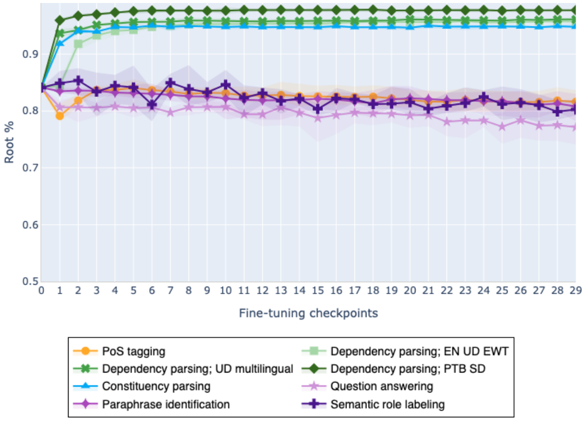


Figure 3: Tree depth evaluation. *Root %* evolution.

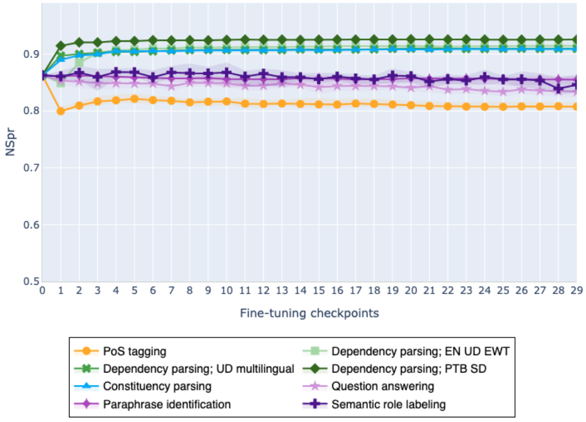


Figure 4: Tree depth evaluation. *Nspr* evolution.

terestingly, the probe performs similarly even if the parsing task is modeled as a sequence labeling problem (as in constituency parsing), suggesting that the structure of syntax trees emerges in such models even when no tree is explicitly involved in the task. The initial drop observed for PoS tagging and monolingual dependency parsing with UD, trained on UD EN EWT, may be related to the size of the dataset, since UD EN EWT is significantly smaller than the other datasets and therefore the models see less examples per checkpoint.

Semantics-related tasks As shown in Figures 1 and 2, both metrics follow different behaviours (again, shades represent the variability across the 5 model runs). Paraphrase identification shows a small but constant UUAS loss along the fine-tuning, while QA shows a slightly steeper loss trend. Initially, SRL loses around 12 points, suggesting that it discards some syntactic information right at the beginning, and follows a similar downward trend afterwards. Those three tasks show a stable performance along the fine-tuning for the DSpr metric, which implies that even if there is a loss in UUAS information it does not impact the distance ordering.

4.2 Tree depth evaluation

The probe evaluates models with respect to their ability to recreate the order of words specified by their depth in the parse tree, assessing their ability to identify the root of the sentence as the least deep word (*Root %*) and computing the Spearman correlation between the predicted and the true depth ordering, averaging across all sentences with lengths between 5 and 50 (henceforth referred to as *NSpr*).

Morpho-syntactic tasks Again, both metrics follow a similar behaviour, as shown in Figures 3 and 4. PoS tagging shows a sustained loss of performance, though softer than the loss observed for the distance metrics. This loss is slightly less pronounced for *Root %* than for *Nspr*, suggesting that while depth-related syntactic information may be of less relevance to PoS tagging than it is to the other morpho-syntactic tasks, identifying the root of the sentence may be important, as the root of the sentence is likely to become one of the ambiguous tags and therefore identifying it may help to select the correct label. Constituency parsing and dependency parsing share a similar tendency, with a big improvement in the first steps preserved along the rest of the fine-tuning process, reinforcing the intuition previously introduced in Section 4.1 about the structure of syntax trees emerging in models even when no tree is explicitly involved in the task. Again, an initial drop can be observed for PoS tagging and monolingual dependency parsing with UD, most probably related to the smaller size of the UD EN EWT dataset used in both tasks.

Semantics-related tasks Both metrics follow a similar behaviour, as shown in Figures 3 and 4, with all tasks following a soft but sustained loss of performance until the end of the fine-tuning process, specially noticeable for *Root %*.

5 Conclusions

We show that fine-tuning is not always a conservative process. Rather, the syntactic information initially encoded in the models is forgotten (PoS tagging), reinforced (parsing) or preserved (semantics-related tasks) in different (sometimes unexpected) ways along the fine-tuning, depending on the task. Thus, we expected that morpho-syntactic tasks clearly reinforce syntactic information. However, PoS tagging forgets it, which, on the other side, can also be justified linguistically (cf. Section 4.1). In contrast, tasks closer to semantics mostly preserve the syntactic knowledge initially encoded. This interesting observation reinforces recent findings that models benefit from explicitly injecting syntactic information for such tasks (Singh Sachan et al., 2020).

Overall, we observed that morpho-syntactic tasks experiment substantial changes in the initial phases, while semantic-related tasks maintain a more stable trend, highlighting the importance of syntactic information in tasks that are not explicitly syntactic in nature (Kuncoro et al., 2020). These observations lead to some interesting insights, but also to further questions; for instance: Can we find a specific set of probes covering different linguistic phenomena to be used as a pretraining stopping criteria? Would this lead to an improvement in the encoding of the linguistic information on pretrained models?

Acknowledgments

This work has been partially funded by the European Commission via its H2020 Research Program under the contract numbers 779962, 786731, 825079, and 870930.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pages 33–36. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual bert.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Lrec*, volume 6, pages 449–454.

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing individual neurons in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4865–4880, Online. Association for Computational Linguistics.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2020. When bert forgets how to pos: Amnesic probing of linguistic properties and mlm predictions.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Jon Gauthier and Roger Levy. 2019. Linking artificial and human neural representations of language. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 529–539, Hong Kong, China. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

James Henderson. 2020. The unstoppable rise of computational linguistics in deep learning.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Adhiguna Kuncoro, Lingpeng Kong, Daniel Fried, Dani Yogatama, Laura Rimell, Chris Dyer, and Phil Blunsom. 2020. Syntactic structure distillation pretraining for bidirectional encoders.

Tomasz Limisiewicz, David Mareček, and Rudolf Rosa. 2020. Universal Dependencies According to BERT: Both More Specific and More General. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2710–2722, Online. Association for Computational Linguistics.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert's linguistic knowledge. *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Tal Linzen and Marco Baroni. 2020. Syntactic structure from deep learning. *arXiv preprint arXiv:2004.10827*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Christopher D Manning. 2011. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *International conference on intelligent text processing and computational linguistics*, pages 171–189. Springer.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Vinit Ravishankar, Artur Kulmizev, Mostafa Abdou, Anders Søgaard, and Joakim Nivre. 2021. Attention can reflect syntactic structure (if you let it). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Natalia Silveira, Timothy Dozat, Marie-Catherine De Marneffe, Samuel R Bowman, Miriam Connor, John Bauer, and Christopher D Manning. 2014. A gold standard dependency corpus for english. In *LREC*, pages 2897–2904.

Devendra Singh Sachan, Yuhao Zhang, Peng Qi, and William Hamilton. 2020. Do syntax trees help pre-trained transformers extract information? *arXiv e-prints*, pages arXiv–2008.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

David Vilares, Michalina Strzyz, Anders Søgaard, and Carlos Gómez-Rodríguez. 2020. Parsing as pretraining. *arXiv preprint arXiv:2002.01685*.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Yufei Wang, Mark Johnson, Stephen Wan, Yifang Sun, and Wei Wang. 2019. How to best use syntax in semantic role labelling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5338–5343, Florence, Italy. Association for Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

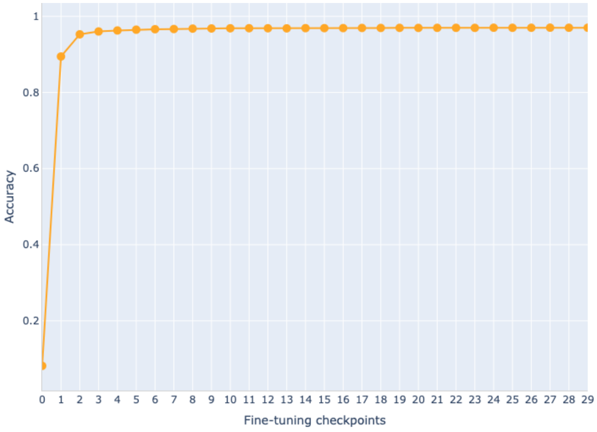
Daniel Zeman, Joakim Nivre, Mitchell Abrams, Noëmi Aepli, Željko Agić, Lars Ahrenberg, Gabrielė Aleksandravičiūtė, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, Colin Batchelor, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Agnė Bielinskienė, Rogier Blokland, Victoria Bobicev, Loïc Boizou, Emanuel Borges Völker, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Kristina Brokaitė, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Tatiana Cavalcanti, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Alessandra T. Cignarella, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Elvis de Souza, Arantza Diaz de Ilarraza, Carly Dickerson, Bamba Dione, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Hanne Eckhoff, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Olga Erina, Tomaž Erjavec, Aline Etienne, Wograine Evelyn, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Kazunori Fujita, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Bernadeta Griciūtė, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Mika Hämäläinen, Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Johannes Heinecke, Felix Hennig, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Takumi Ikeda, Radu Ion, Elena Irimia, lájídé Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Markus Juutinen, Hüner Kaşıkara, Andre Kaasen, Nadezhda Kabaeva, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Elena Klementieva, Arne Köhn, Kamil Kopacewicz, Natalia Kotsyba, Jolanta Kovalevskaitė, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phng Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Maria Liovina, Yuan Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Sarah McGuinness, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Maria Mitrofan, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Tomohiko Morioka, Shinsuke Mori, Shigeki Moro, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Robert Munro, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lng Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Yoshihiro Nikaido, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Atul Kr. Ojha, Adédayọ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Angelika Peljak-Łapińska, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Daria Petrova, Slav Petrov, Jason Phelan, Jussi Piitulainen, Tommi A Pirinen, Emily Pitler, Barbara Plank, Thierry Poibeau, Larisa Ponomareva, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Peng Qi, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Ivan Riabov, Michael Rießler, Erika Rimkutė, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roșca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Alessio Salomoni, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Dage Särg, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Hiroyuki Shirasu, Muh Shohibussirri, Dmitry Sichinava, Aline Silveira, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Shingo Suzuki, Zsolt Szántó, Dima Taji, Yuta Takahashi, Fabio Tamburini, Takaaki Tanaka, Isabelle Tellier, Guillaume Thomas, Liisi Torga, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Andrius Utka, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Abigail Walsh, Jing Xian Wang, Jonathan North Washington, Maximilan Wendt, Seyi Williams, Mats Wirén, Christian Wittern, Tsegay Woldemariam, Tak-sum Wong, Alina Wróblewska, Mary Yako, Naoki Yamazaki, Chunxiao Yan, Koichi Yasuoka, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Manying Zhang, and Hanzhi Zhu. 2019. Universal dependencies 2.5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

A Target tasks performance evolution

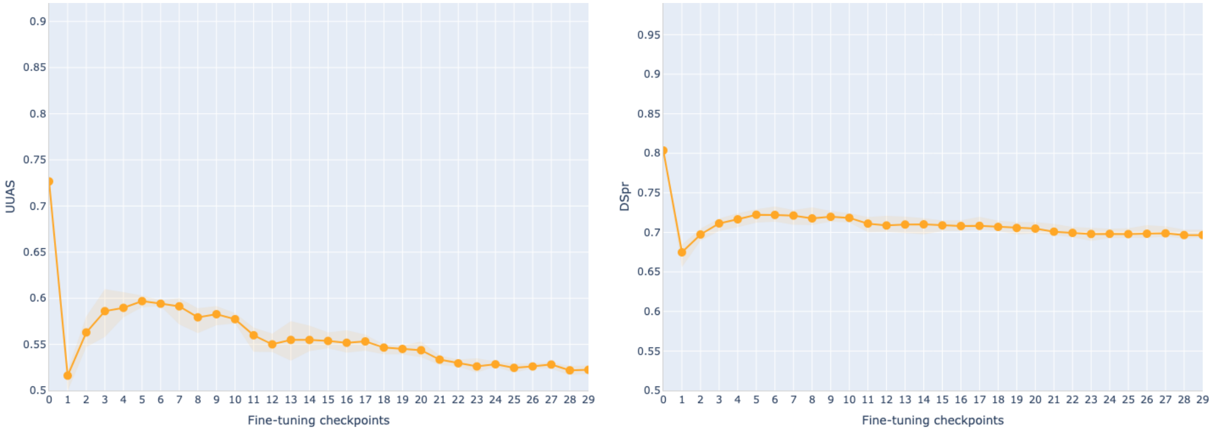
To complement the results shown in the main paper, we include here the performance curves of the target tasks for which the models are fine-tuned, along with the performance curves of the structural probes metrics, facilitating the comparison of the evolution of the encoded syntax trees information and the target tasks performances.

Along with the performance curves of the four structural probes metrics (*UUAS*, *Nspr*, *Root %* and *Dspr*), the following figures include the performance curves of the target tasks and a brief discussion of the results, to help interpretation. Figure 5 shows the accuracy evolution of PoS tagging. Figures 6, 7 and 8 show the Labeled Attachment Score (LAS) of Dependency parsing with PTB SD, EN UD EWT and UD multilingual, respectively. Figure 9 shows the accuracy evolution of Constituency parsing. Figure 10 shows the F1 score evolution of Question Answering. Figure 11 shows the F1 score and accuracy evolution of Paraphrase identification. Finally, Figure 12 shows the F1 score evolution of Semantic Role Labeling.

PoS tagging reaches a 0.95 accuracy in only two checkpoints, ending up with a 0.97 on the last checkpoint (Figure 5a). It shows a loss of accuracy for the four probing metrics all along the fine-tuning process, especially noticeable for *UUAS* (Figure 5b) and *Root %* (Figure 5d), suggesting that syntactic information is of less relevance to PoS tagging than could be intuitively assumed. The loss is less pronounced on depth-related metrics, maybe due to the fact that the root of the sentence usually corresponds to the verb, which may also help in identifying the PoS of surrounding words.

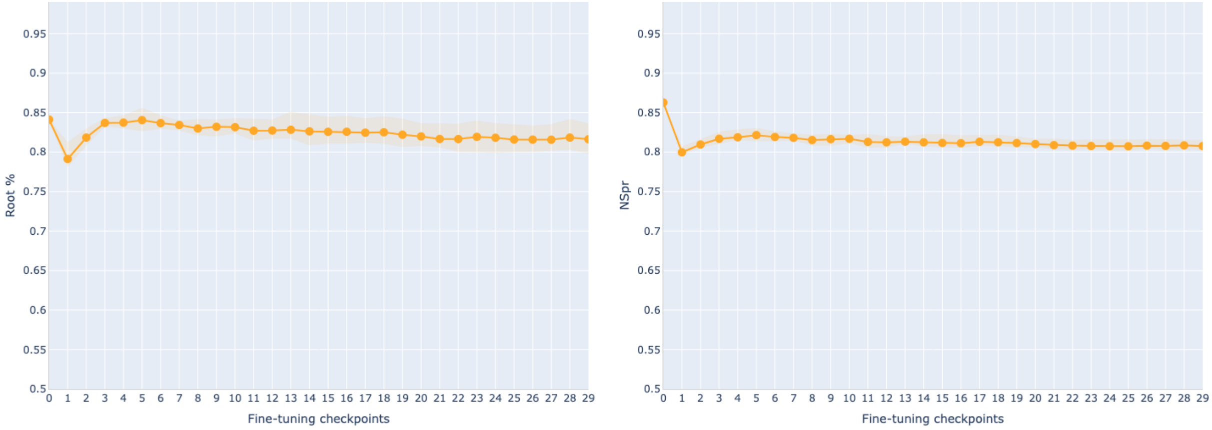


(a) Fine-tuning. Accuracy



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*

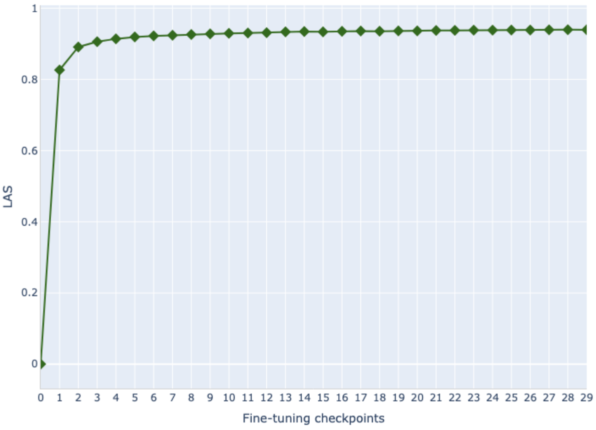


(d) Structural probes tree depth evaluation. *Root %*

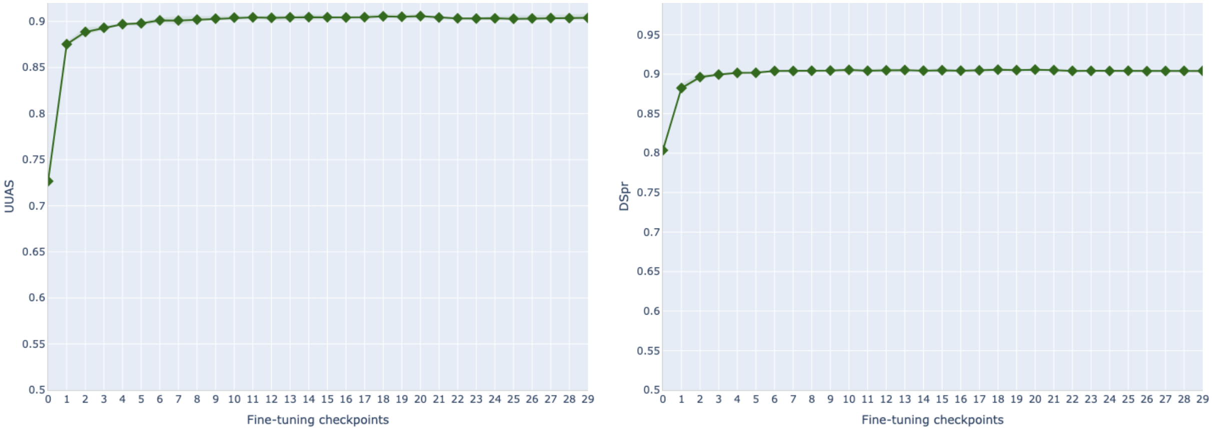
(e) Structural probes tree depth evaluation. *Nspr*.

Figure 5: POS Tagging. Fine-tuning & probing metrics evolution.

Dependency parsing with PTB SD shows a steep learning curve (Figure 6a), reaching a performance of 0.90 LAS on the third checkpoint, up to a final 0.94. All four probing metrics show an important improvement in the first fine-tuning step (Figures 6b, 6c, 6d and 6e), which is preserved along the rest of the process. As the task heavily relies on syntactic information, this improvement intuitively makes sense. Compared to the result of the other dependency parsing experiments, this one show bigger improvements because the probing is trained on the same dataset.

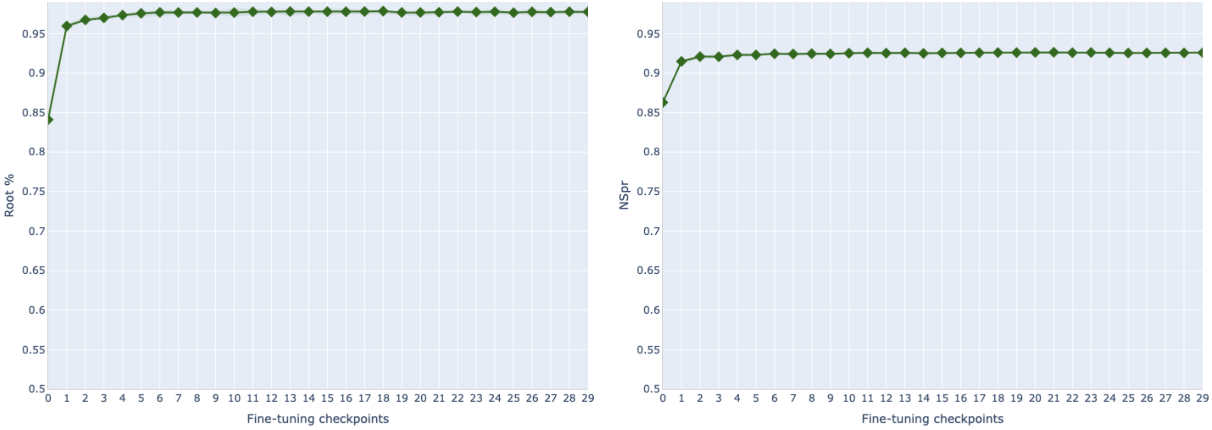


(a) Fine-tuning. LAS



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.

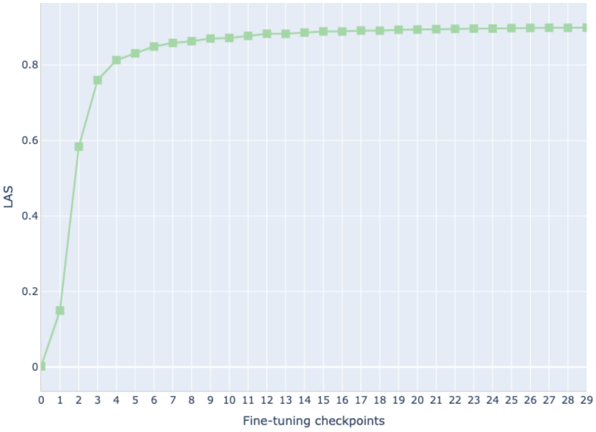


(d) Structural probes tree depth evaluation. *Root %*

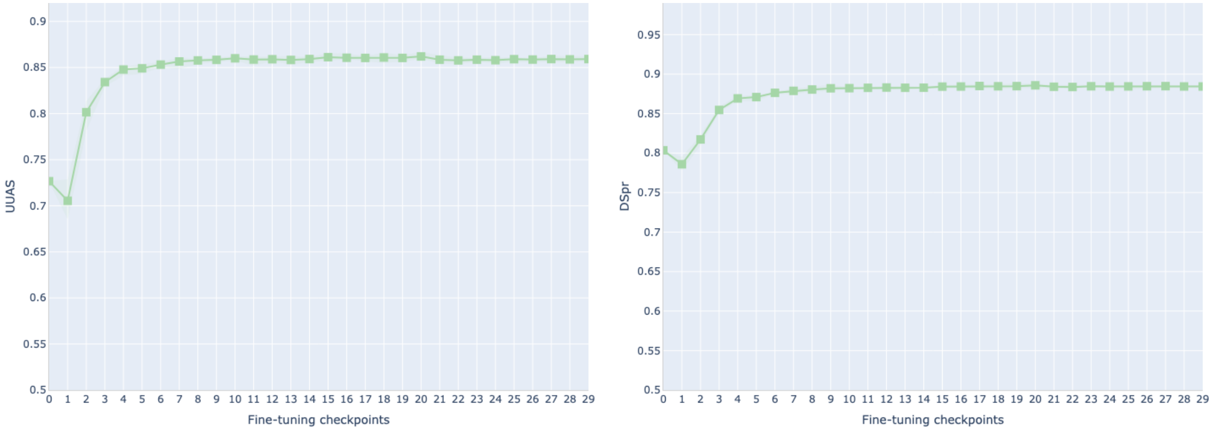
(e) Structural probes tree depth evaluation. *Nspr*

Figure 6: Dependency Parsing PTB SD. Fine-tuning & probing metrics evolution.

Dependency parsing with EN UD EWT shows a shallower learning curve than other experiments (Figure 7a), as the dataset is significantly smaller than the multilingual and PTB and therefore the models see less examples per checkpoint, ending up with a high performance of 0.9. After an initial drop (probably due to the dataset size, as mentioned before), the probing metrics show a big improvement in the first fine-tuning steps, preserved along the rest of the process (Figures 7b, 7c, 7d and 7e). As the task heavily relies on syntactic information, this improvement intuitively makes sense.

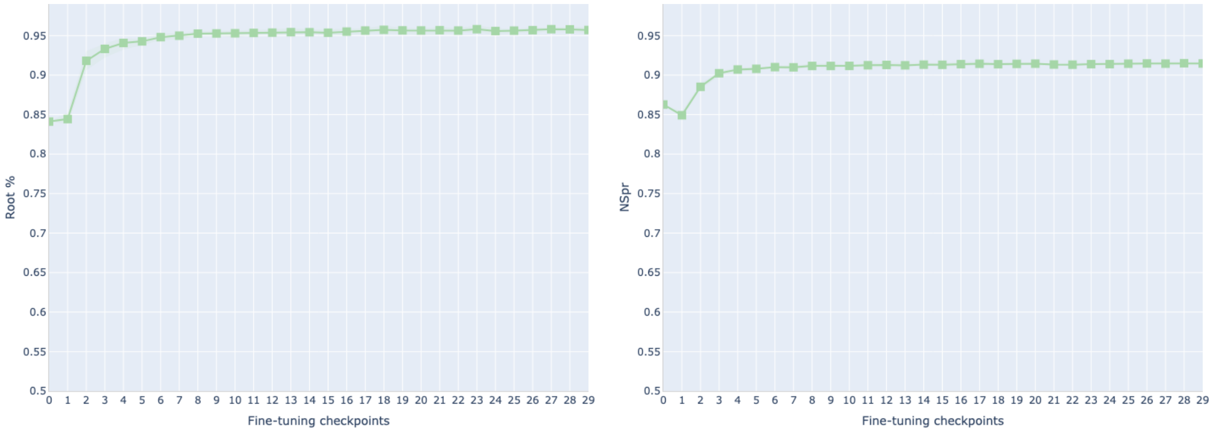


(a) Fine-tuning. LAS



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.

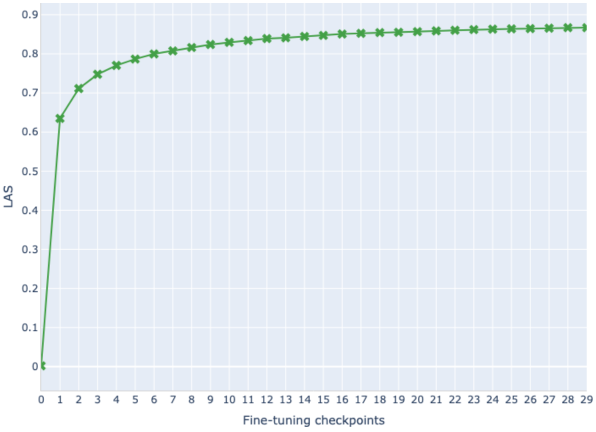


(d) Structural probes tree depth evaluation. *Root %*

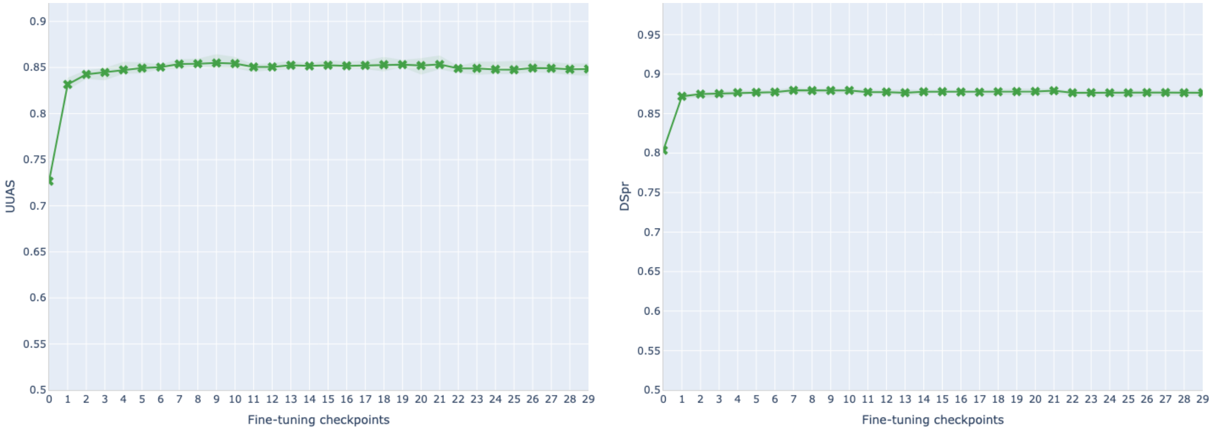
(e) Structural probes tree depth evaluation. *Nspr*

Figure 7: Dependency Parsing EN UD EWT. Fine-tuning & probing metrics evolution.

Multilingual dependency parsing shows a steeper learning curve than dependency parsing with EN UD EWT, as it is trained with a larger dataset (Figure 8a), reaching a performance of 0.87 in LAS. All four probing metrics show a big improvement in the first fine-tuning step, preserved along the rest of the process (Figures 8b, 8c, 8d and 8e). As the task heavily relies on syntactic information, this improvement intuitively makes sense.

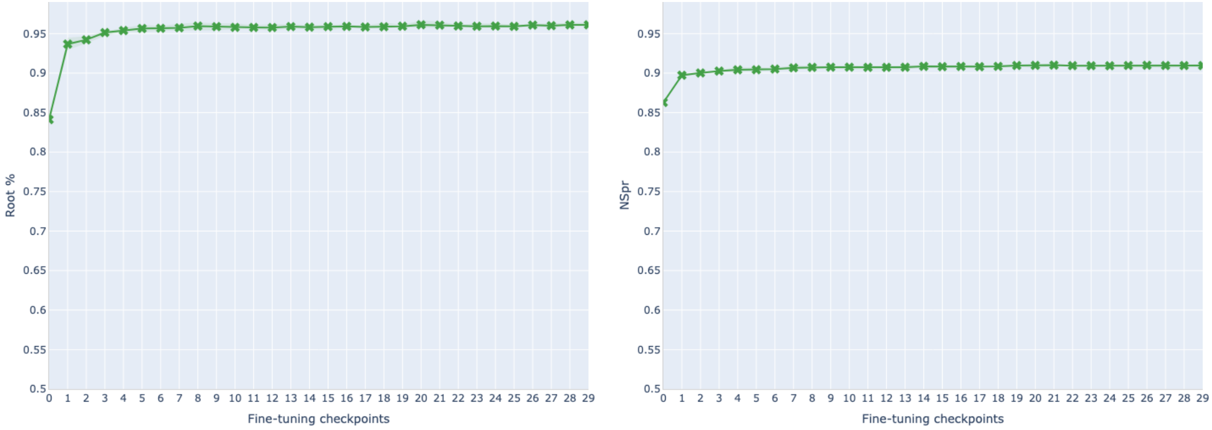


(a) Fine-tuning. LAS



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.

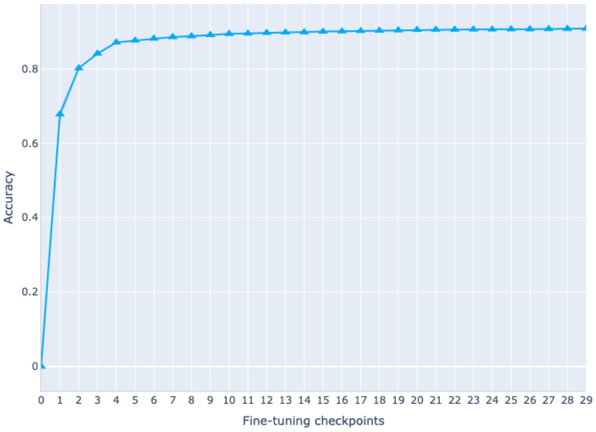


(d) Structural probes tree depth evaluation. *Root %*

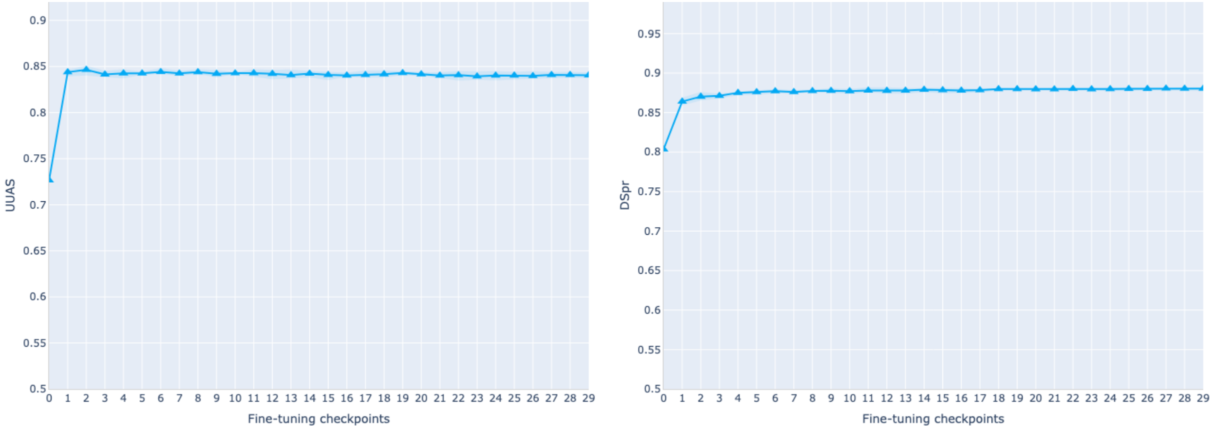
(e) Structural probes tree depth evaluation. *Nspr*

Figure 8: Dependency Parsing UD Multilingual. Fine-tuning & probing metrics evolution.

Constituency parsing fine-tuning follows a steep curve, quickly reaching an Accuracy of 0.87 that is further improved to 0.9 in the last checkpoint (Figure 9a). All four probing metrics show a big improvement in the first fine-tuning steps, preserved along the rest of the process (Figures 9b, 9c, 9d and 9e). As the task heavily relies on syntactic information, this improvement intuitively makes sense. Interestingly, even though the task is modeled as a sequence labeling problem, the probe performs similarly to the dependency parsing tasks, suggesting that the structure of syntax trees emerges in such models even when no tree is explicitly involved in the task.

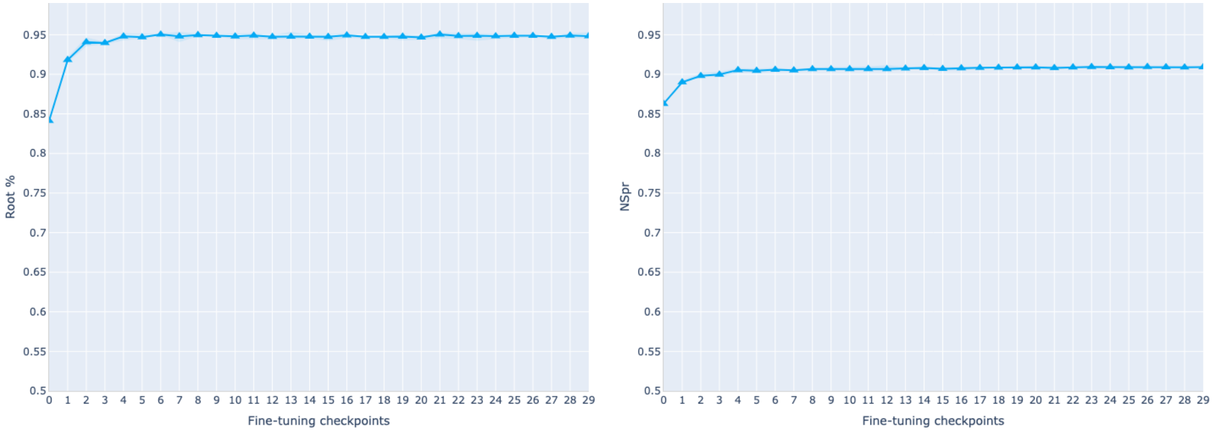


(a) Fine-tuning. Accuracy



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.

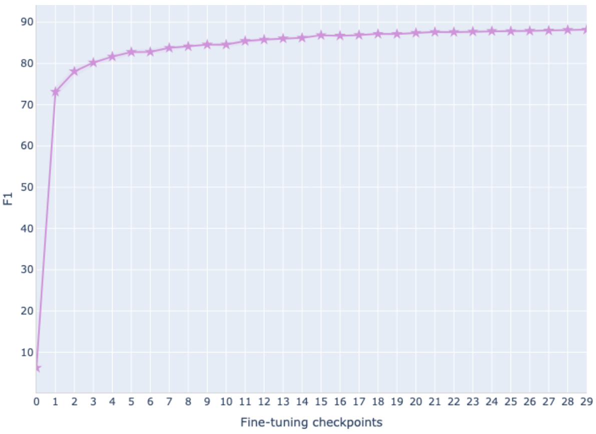


(d) Structural probes tree depth evaluation. *Root %*

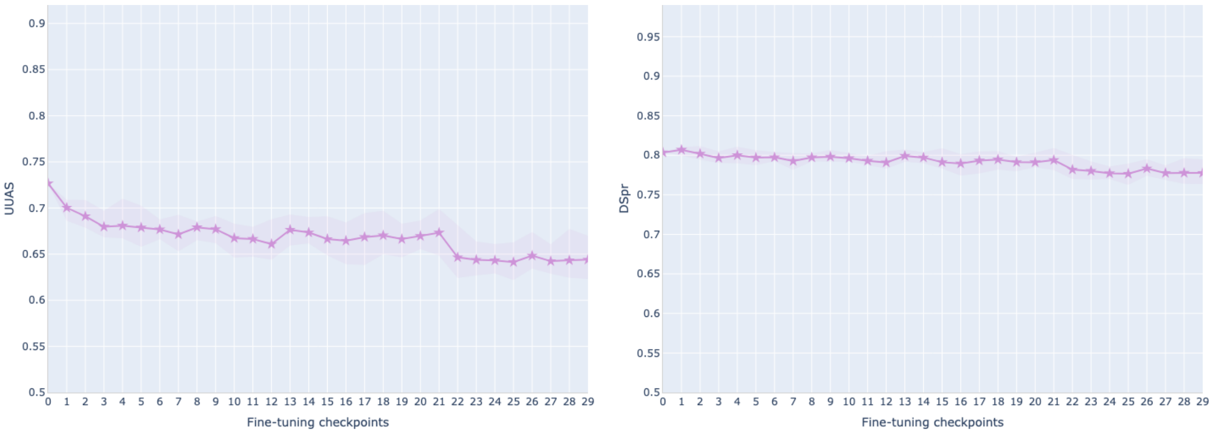
(e) Structural probes tree depth evaluation. *Nspr*

Figure 9: Constituent Parsing. Fine-tuning & probing metrics evolution.

Question answering fine-tuning quickly reaches an F1 score of 0.73 on the first step, which is further improved to 0.88 in the last checkpoint (Figure 10a). All four probing metrics show a clear loss trend (Figures 10b, 10c, 10d and 10e). The loss is specially noticeable for *UUAS* and *Root %*, and more stable for the Spearman correlations, suggesting that even if there is a loss of information it does not impact the distance and depth orderings.

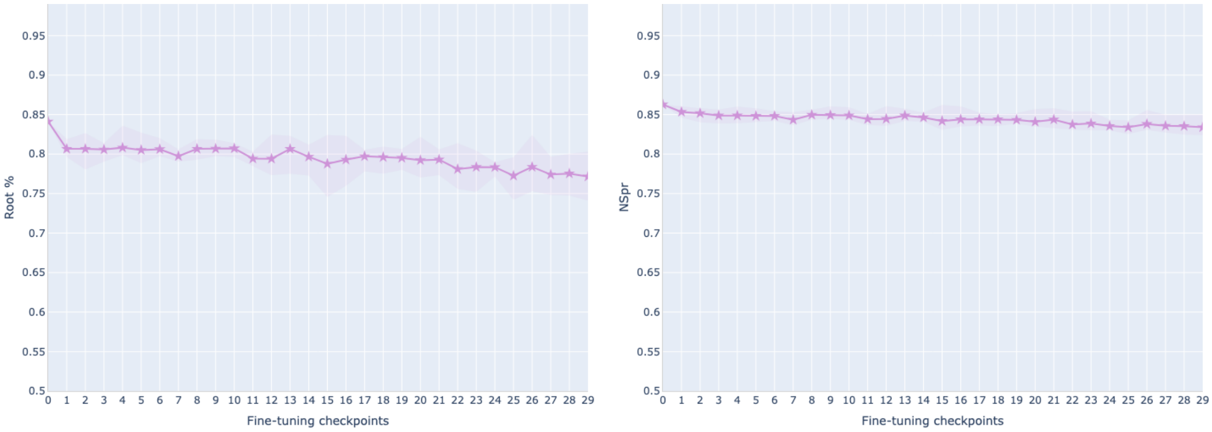


(a) Fine-tuning. F1



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.

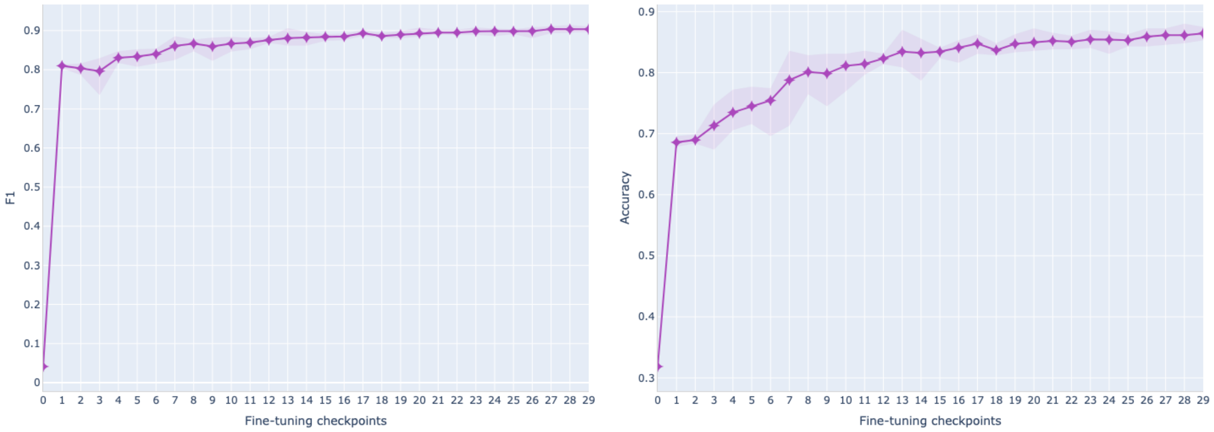


(d) Structural probes tree depth evaluation. *Root %*

(e) Structural probes tree depth evaluation. *Nspr*

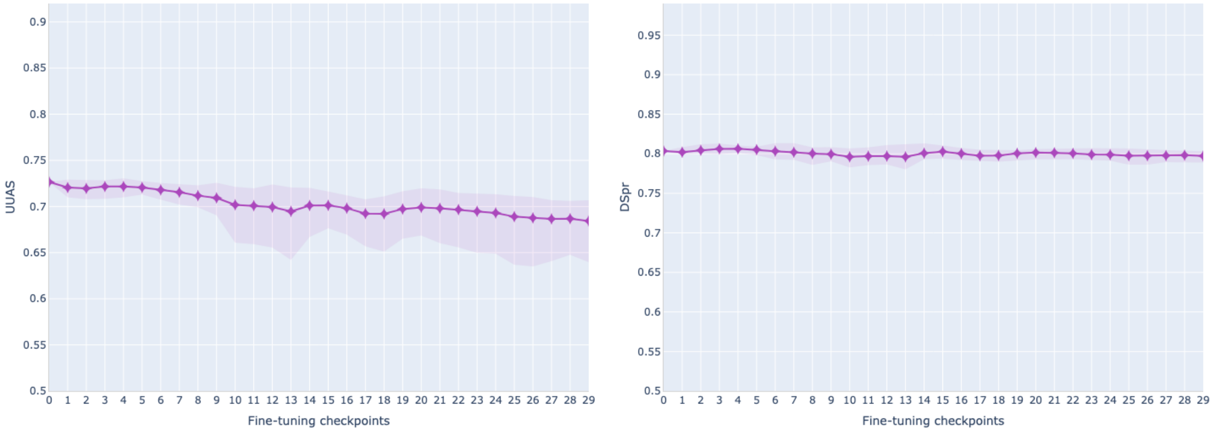
Figure 10: Question Answering. Fine-tuning & probing metrics evolution.

Paraphrase identification fine-tuning starts with an F1 score of 0.81 on the first step that is further improved to 0.90 in the last checkpoint (Figure 11a). Regarding accuracy, after reaching 0.69 on the first checkpoint it follows a shallower curve to a final 0.86 (Figure 11b). All four probing metrics follow a loss trend (Figures 11c, 11d, 11e and 11f). The loss is specially noticeable for *UUAS* and *Root %*, and more stable for the Spearman correlations, suggesting that even if there is a loss of information it does not impact the distance and depth orderings.



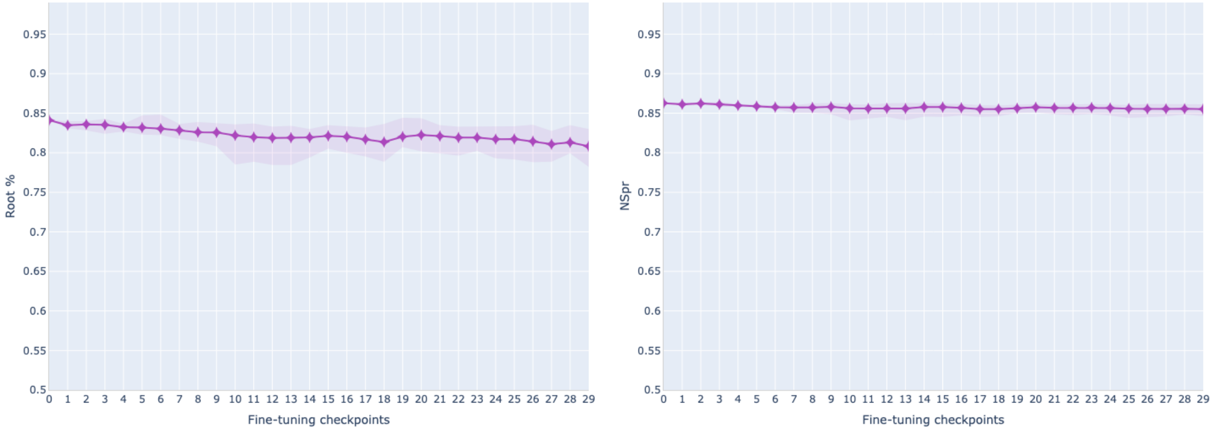
(a) Fine-tuning. F1

(b) Accuracy



(c) Structural probes tree distance evaluation. *UUAS*

(d) Structural probes tree distance evaluation. *Dspr*.

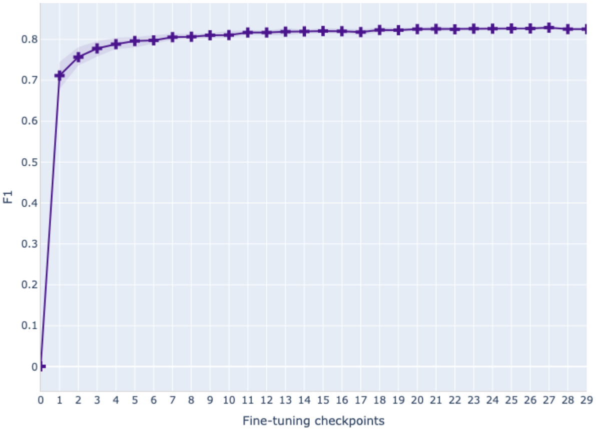


(e) Structural probes tree depth evaluation. *Root %*

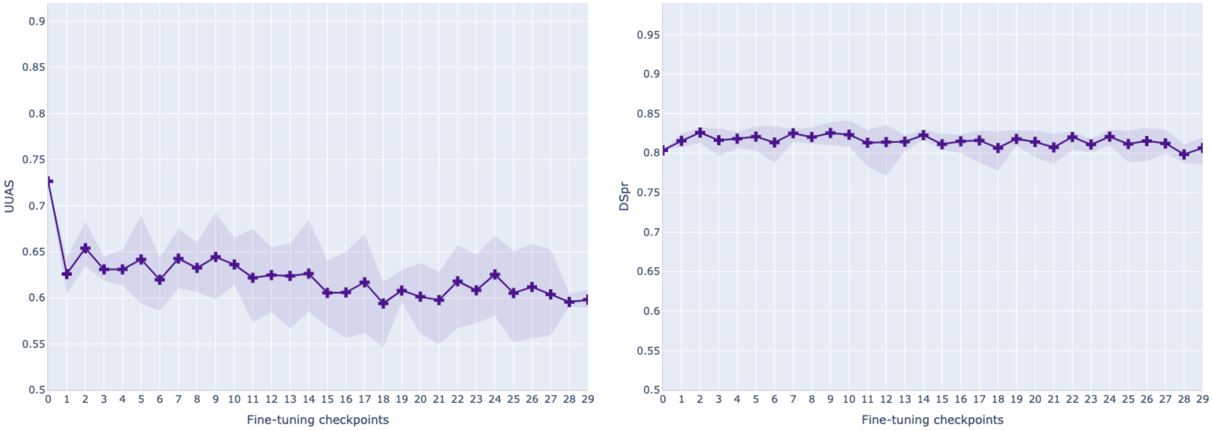
(f) Structural probes tree depth evaluation. *Nspr*

Figure 11: Paraphrase identification. Fine-tuning & probing metrics evolution.

Semantic Role Labeling fine-tuning follows a steep curve for F1, quickly reaching an F1 score of 0.71 on the first step that is further improved to 0.82 in the last checkpoint (Figure 12a). All four probing metrics follow a loss trend (Figures 12b, 12c, 12d and 12e). The loss is specially noticeable for *UUAS*, which initially loses around 12 *UUAS* points, and more stable for the Spearman correlations, suggesting that even if there is a loss of information it does not impact the distance and depth orderings.

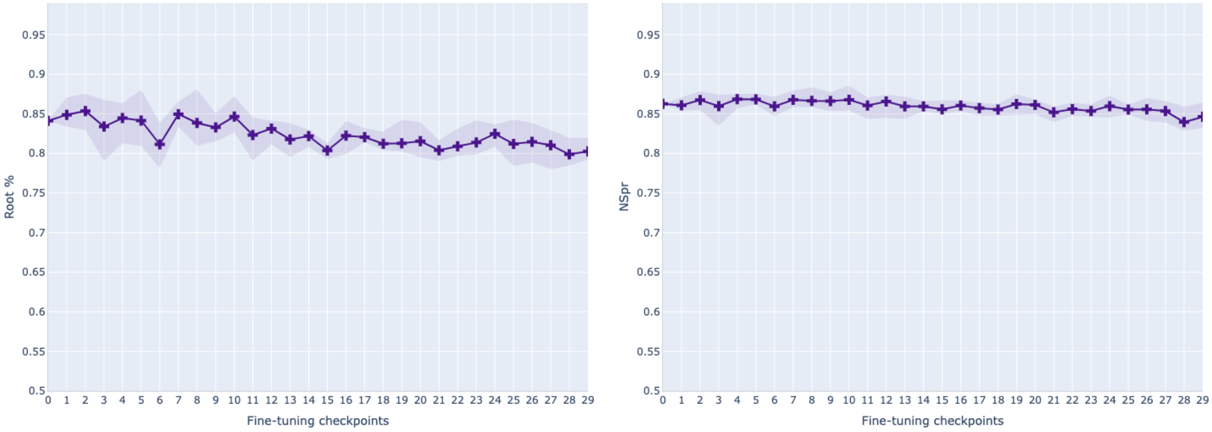


(a) Fine-tuning. F1



(b) Structural probes tree distance evaluation. *UUAS*

(c) Structural probes tree distance evaluation. *Dspr*.



(d) Structural probes tree depth evaluation. *Root %*

(e) Structural probes tree depth evaluation. *Nspr*

Figure 12: Semantic Role Labeling. Fine-tuning & probing metrics evolution.

Identify, Align, and Integrate: Matching Knowledge Graphs to Commonsense Reasoning Tasks

Lisa Bauer
UNC Chapel Hill
`lbauer6@cs.unc.edu`

Mohit Bansal
UNC Chapel Hill
`mbansal@cs.unc.edu`

Abstract

Integrating external knowledge into commonsense reasoning tasks has shown progress in resolving some, but not all, knowledge gaps in these tasks. For knowledge integration to yield peak performance, it is critical to select a knowledge graph (KG) that is well-aligned with the given task's objective. We present an approach to assess how well a candidate KG can correctly identify and accurately fill in gaps of reasoning for a task, which we call KG-to-task match. We show this KG-to-task match in 3 phases: knowledge-task identification, knowledge-task alignment, and knowledge-task integration. We also analyze our transformer-based KG-to-task models via commonsense probes to measure how much knowledge is captured in these models before and after KG integration. Empirically, we investigate KG matches for the SocialIQA (SIQA) (Sap et al., 2019b), Physical IQA (PIQA) (Bisk et al., 2020), and MCScript2.0 (Ostermann et al., 2019) datasets with 3 diverse KGs: ATOMIC (Sap et al., 2019a), ConceptNet (Speer et al., 2017), and an automatically constructed instructional KG based on WikiHow (Koupaee and Wang, 2018). With our methods we are able to demonstrate that ATOMIC, an event-inference focused KG, is the best match for SIQA and MCScript2.0, and that the taxonomic ConceptNet and WikiHow-based KGs are the best matches for PIQA across all 3 analysis phases. We verify our methods and findings with human evaluation.[1]

1 Introduction

Recently, several datasets (Sap et al., 2019b; Huang et al., 2019; Bhagavatula et al., 2020; Talmor et al., 2019b) have been released to tackle the challenge of commonsense reasoning. While deep pretrained language-models (LMs) (Devlin et al., 2019; Radford et al., 2019; Liu et al., 2019; Yang et al., 2019) have been at the top of most leaderboards, they still have shortcomings when it comes to commonsense reasoning (Sap et al., 2019b; Rajani et al., 2019; Mitra et al., 2019). Thus, incorporating knowledge graph (KG) information into these models is an active area of research (Lin et al., 2019a; Sun et al., 2019; Mitra et al., 2019; Bosselut et al., 2019). However, when selecting a KG match for a task, it is often difficult to quantitatively assess what kind of knowledge is missing from these models and how much of the missing knowledge required for the task is available in a candidate KG. It is also critical to examine how easily transformer-based models can learn commonsense knowledge, to determine the benefits of integrating a KG.

We investigate how well a KG matches with a task objective, referred to as *KG-to-task match*. We use a 3-step process that examines knowledge *identification, alignment,* and *integration*. We utilize a modular pipeline approach to allow for interpretable results and easy replacement of new and different modules. Our approach reveals features such as: how often a KG identifies a knowledge gap in a question-answer pair (identification), whether a KG identifies the correct knowledge gap (alignment), and whether the inserted knowledge correctly fills the knowledge gap required for the task (integration). These steps are depicted in Fig. 1. We also compare the effects of knowledge content, structure, and shape.

The results of this analysis are impacted by the model we use, and thus we also develop *probes* to examine how much commonsense knowledge LMs already know and how easy it is for them to learn. We evaluate our KG-to-task models in a QA probe setup to examine how much commonsense is learned with and without the matched KG. Our probes are automatically built from ATOMIC, en-

---

[1] Our code and commonsense probes will be publicly available on our webpage.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2259–2272
April 19 - 23, 2021.

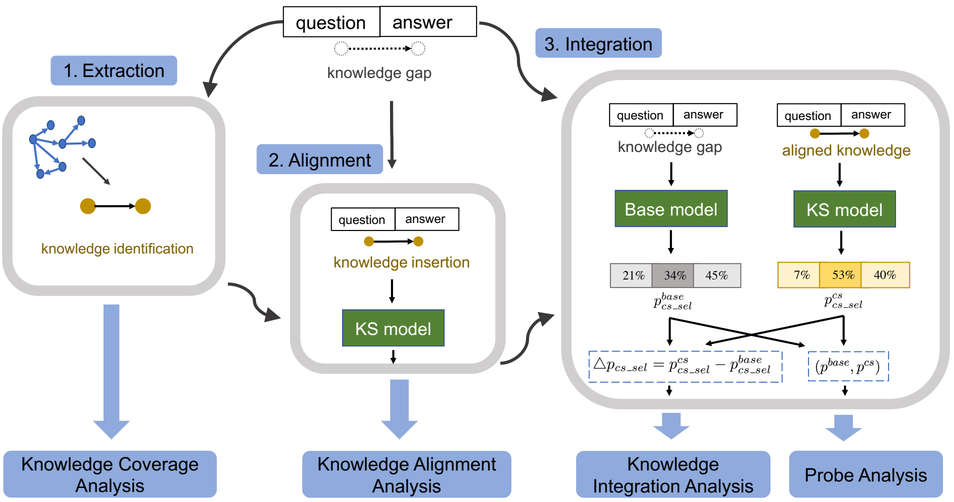


Figure 1: We illustrate our 3 phases of analysis: Knowledge-Task Extraction Analysis, Knowledge-Task Alignment Analysis, and Knowledge-Task Integration Analysis with Probing Analysis.

abling us to leverage existing knowledge sources as a probing base without relying on expensive collection methods. We also include an MLM probe setup to obtain zero-shot and fine-tuned results on probes for social relations, agent-patient assignment, and world knowledge.

We present detailed empirical results on three diverse datasets: SocialIQA (SIQA) task (Sap et al., 2019b), which requires social knowledge; Physical IQA (PIQA) (Bisk et al., 2020) which requires physical knowledge; and MCScript2.0 (Ostermann et al., 2019), which requires commonsense script knowledge, not restricted to a particular domain. Since both SIQA and PIQA require a particular domain of commonsense knowledge, these tasks allow us to draw strong conclusions about KG integration, as knowledge must be well aligned with the tasks to yield performance gains. Analyzing MCScript2.0, on the other hand, allows us to understand how this analysis applies to a task where the best match is not obvious. We compare KG-to-task match with three diverse KGs: ATOMIC (Sap et al., 2019a), ConceptNet (Speer et al., 2017), and automatically extracted subgraphs from WikiHow. Each KG is tailored for a different commonsense domain: ATOMIC focuses on social commonsense, ConceptNet on taxonomic commonsense, and WikiHow on instruction-based commonsense. This allows us to see how different tasks require different types of commonsense knowledge.

To investigate KG-to-task match, we follow three phases: identify, align, and integrate. In our first phase, we examine knowledge gap identification by analyzing our extraction quantities. In our second phase, we examine alignment by utilizing a 'knowledge-surrounded' (KS) model, in which we replace task candidate answers with knowledge-surrounded answers. We found that ATOMIC is the best match for SIQA across both identification and alignment: 11% more ATOMIC data is extracted for question-answer knowledge gaps than ConceptNet data, with a 4.8% performance increase over BERT using our ATOMIC KS model. We use our third phase, integration, to investigate the classification change distributions from BERT to the KS model, finding that our model is more confident about correct classification changes, supporting the ATOMIC-SIQA match. Additionally, both ConceptNet and WikiHow graphs outperformed ATOMIC on PIQA: 8% more ConceptNet data is extracted than ATOMIC and a 17.4% performance increase is achieved with our ConceptNet KS model, whereas we get a 15.5% increase with our WikiHow KS model. Finally, we find that ATOMIC is the best match for MCScript2.0, with a 2.7% increase with our ATOMIC KS model.

We also perform human evaluation and show important connections between the analysis phases. We see that if our KS model shows improvement for high quality settings, our extraction step is a valid knowledge-gap identification metric between 74% and 89% of the time, depending on the dataset. We also show that our best alignment strategy for ATOMIC-SIQA fills knowledge gaps 66% of the time, outperforming the best alignment strategy for ConceptNet-SIQA, which supports our KS model performance results. We find similar trends for PIQA alignment and also find that the amount of information available at inference time may affect alignment results for MCScript2.0. Human evaluation shows that 93% of ATOMIC-SIQA KS model prediction changes (with respect to the baseline) select the answer with the highest knowledge quality, verifying our integration phase as a quality metric.

Our commonsense QA probes before and after KG integration show that our KS model only considerably outperforms the BERT baseline on certain relational probes, indicating the type of knowledge gaps ATOMIC is better at resolving, e.g., relational knowledge such as feelings, reactions, etc.

Overall, our methods not only illustrate the type of knowledge that current transformer-based models are missing to approach human-level commonsense reasoning but also how we can identify, align, and integrate knowledge between a KG and a task to find the best match to fill in these missing gaps of reasoning.

2 Related Work

Language Model Probes: Recent work in probe construction has examined neural model knowledge (Richardson and Sabharwal, 2019; Zhou et al., 2020b; Rogers et al., 2020; Lin et al., 2020). Talmor et al. (2019a) constructed eight tasks that evaluated LMs for operations such as comparison, conjunction, and composition. Zhou et al. (2020a) created logically equivalent probes to evaluate robustness on commonsense tasks to syntax. Kwon et al. (2019) proposed tests based on ConceptNet to measure what types of commonsense MLMs understand. Our work instead focuses on probing models for causal, social commonsense in both the MLM and QA setup before and after KG integration and fine-tuning, and automatically constructs probes from existing knowledge sources.

Commonsense Reasoning: Recent commonsense reasoning datasets (Bhagavatula et al., 2020; Zellers et al., 2018; Zhou et al., 2019; Sap et al., 2019b; Bisk et al., 2020; Lin et al., 2019b; Zellers et al., 2019; Ostermann et al., 2019) have motivated research in several domains of commonsense: abductive, grounded, temporal, social, and physical. Commonsense reasoning can be learned either by KGs pre-training (Bosselut et al., 2019; Bosselut and Choi, 2019; Ye et al., 2019) or by integrating explicit knowledge (Chen et al., 2017; Mitra et al., 2018; Bauer et al., 2018; Lin et al., 2019a; Zhang et al., 2019; Xiong et al., 2019). We show how finding nuanced knowledge for successful commonsense reasoning can be quantitatively examined.

Commonsense Knowledge Analysis: Zhang et al. (2020) presented a categorization of essential knowledge for the Winograd Schema Challenge (Levesque et al., 2012) via human annotation to identify what knowledge was required for better commonsense reasoning. Ma et al. (2019) investigated how KG integration methods affected model performance on different tasks and found that the degree of domain overlap between the KG and the task plays a crucial role in performance. We further investigate this by measuring KG-to-task match across 3 automatic phases, considering different extraction methods, and probing models for knowledge before and after KG integration.

3 Tasks & Knowledge Graphs

3.1 Tasks

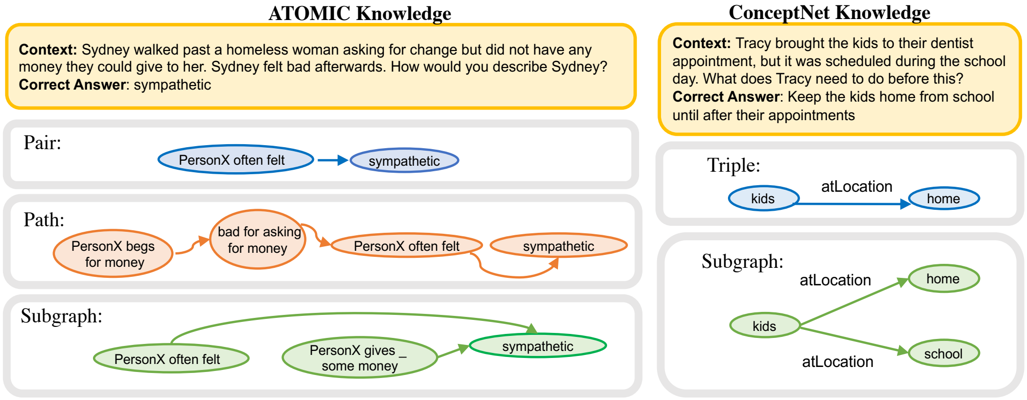
SIQA: The SocialIQA (SIQA) (Sap et al., 2019b) task focuses on social commonsense. Given a context and question, a model selects from 3 answers. SIQA contexts are based on ATOMIC (Sap et al., 2019a) events and SIQA question types are guided by ATOMIC inference dimensions. Thus, we expect ATOMIC to match SIQA requirements. For simplicity, we refer to the concatenation of context and question as the question throughout the paper.

PIQA: The PhysicalIQA (PIQA) (Bisk et al., 2020) task objective focuses on physical commonsense reasoning. Given a goal, a model selects from 2 candidate solutions. PIQA is derived from the instruction domain, and thus we expect instructional physical commonsense to benefit PIQA. For simplicity, we refer to the goal as the question.

MCScript2.0: MCScript2.0 (Ostermann et al., 2019) focuses on script events and participants, requiring commonsense knowledge, in particular script knowledge, to answer questions correctly. We specifically choose this dataset such that it does not have a strong preference for any of the KGs we investigate, to illustrate what our analysis may look like for an unpredictable result. For simplicity, we refer to the concatenation of context and question as the question throughout the paper.

3.2 Knowledge Sources

We show results across three knowledge graphs to illustrate differences in KG-to-task identification, alignment, and integration, and to show how BERT



Context: Sydney walked past a homeless woman asking for change but did not have any money they could give to her. Sydney felt bad afterwards. How would you describe Sydney?
Correct Answer: sympathetic

Context: Tracy brought the kids to their dentist appointment, but it was scheduled during the school day. What does Tracy need to do before this?
Correct Answer: Keep the kids home from school until after their appointments

Figure 2: Examples of different knowledge shapes per KG given SIQA context and ground truth answer.

KG	Shape	Cond.	Filter	Sets	Pres.
AT	triple	QC	HQ	CS-1	KS+
CN	path	A	HR	CS-2	KS-
WH	subgraph	-	-	CS-3	-

Table 1: Variations for each knowledge setting. AT=ATOMIC, CN=ConceptNet, WH=WikiHow, QC=question-conditioned, A=unconditioned, HR=high recall, HQ=high quality, KS+/-=knowledge presence at inference time.

responds differently to various types of knowledge.
ATOMIC: ATOMIC (Sap et al., 2019a) is an inferential knowledge atlas that focuses on if-then reasoning. This knowledge is structured as event-inference pairs, where each pair reflects one of 9 possible inference dimensions for different if-then reasoning relations (cause vs. effect, etc). Knowledge is in the form of short, abstracted, free-text. See the left of Fig 2 for examples.
ConceptNet: ConceptNet (Speer et al., 2017) is a taxonomic knowledge graph that connects natural language concepts with relation edges. While there are relation edges similar to ATOMIC inference dimensions, the structure of the knowledge is in the form of structured triples and generally tends to focus on relations between words or phrases. See the right of Fig 2 for examples.
WikiHow: We automatically extract subgraphs from WikiHow (Koupaee and Wang, 2018) to build our own instruction-based, domain-specific KG. Details are found in the appendix.

4 Phase 1: Identify

4.1 Setup

We identify knowledge using the following extraction methods for each KG. Our setup with all possible options is illustrated in Table 1. We will use Fig 2 as a running example throughout this section.

4.1.1 Knowledge Conditioning

Unconditioned (A) Answer-Knowledge:
ATOMIC: For each candidate answer, we extract a pool of top scoring knowledge using tf-idf between the answer and all ATOMIC event-inference pairs.
ConceptNet & WikiHow: For each candidate answer, we extract knowledge that links concepts in the answer to any concept in the KG, where concepts are tokens in the answer and nodes in the KG.
Example: Consider the SIQA context and ground-truth answer on the right side of Fig 2. Here, the A conditioning setup for ConceptNet would extract the triple [keep, Antonym, get_rid].
Question-Conditioned (QC) Answer-Knowl.:
ATOMIC: We select a question-conditioned knowledge pool via the top scoring tf-idf match between the question & candidate answer and all ATOMIC event-inference pairs. We then select a pool of top scoring knowledge for each candidate answer using tf-idf between the candidate answer and the question-conditioned knowledge pool.
ConceptNet & WikiHow: For each candidate answer, we extract knowledge that links concepts in the question directly to concepts in the answer.
Example: All knowledge illustrated in Fig 2 is extracted using QC conditioning.

4.1.2 Knowledge Shape

Knowledge Pairs/Triples:
ATOMIC: We take the highest scoring knowledge pair determined by the conditioning step.
ConceptNet & WikiHow: We select a triple at random from the conditioning step.
Knowledge Paths:
ATOMIC: In the QC setup, for each data point, we extract a question-knowledge pool via top scoring

tf-idf match between the question and all ATOMIC event-inference pairs. If there exists a concept link between the question-knowledge pool and the answer-knowledge pool from the conditioning step, we link this knowledge as a path. In the A setup, we make the modification that our answer-knowledge pool can link to any pair in ATOMIC.

WikiHow: In the QC setup, we find a path from a word in the question, to another word in the question, to a word in the answer. In the A setup, we find a path from a word in the answer to any word it connects to in the KG, as a path through the KG.

Knowledge Subgraphs:

ATOMIC: We take a maximum of the 3 highest scoring knowledge triples determined by the conditioning step to create 1-hop subgraphs.

ConceptNet & WikiHow: From the conditioning knowledge pool, we add the subgraph with the highest number of edges, as we assume these to be the most informative. We only consider 1-hop edges and take the top 5.

Example: All three shape variations are illustrated in Fig 2, using QC conditioning.

4.1.3 Knowledge Filtering

High Quality/Low Recall (HQ): We constrain each answer candidate to keep its highest scoring unique knowledge such that no answer candidate shares knowledge, intending to ensure the relevance of the knowledge to that candidate alone.

Low Quality/High Recall (HR): Candidate keeps its highest scoring knowledge regardless of knowledge sharing among candidates.

4.1.4 Data Subsets & Baseline Training

We split data into subsets depending on how many candidate answers extracted knowledge (CS-X) to evaluate knowledge impact on task performance fairly. For our main results, we use the split in which each answer has access to knowledge (CS-2 for PIQA and MCScript2.0, CS-3 for SIQA). Table 2 illustrates the percent of original data for each split. We compare KS model subset results against a BERT baseline trained and evaluated on the same subset simply without the added knowledge.

4.2 Analysis

We examine how often a KG identifies a potential knowledge gap between a question and an answer. This is illustrated on the far left in Fig. 1. Table 2 shows the percent of knowledge extracted for each QC-HQ subset. We use the QC-HQ setting to show how often our KG specifically identifies question-answer knowledge gaps. We use each KG's best aligned shape in this comparison (Section 5.2.2). For SIQA, we extract more ATOMIC data than for ConceptNet and for PIQA, we extract more ConceptNet data than for ATOMIC. This illustrates that ATOMIC identifies more knowledge gaps for SIQA, and ConceptNet identifies more knowledge gaps for PIQA. For MCScript2.0, we see that the same total knowledge is extracted from both KGs, however more ATOMIC knowledge is extracted in the CS-2 setup, indicating better coverage.

Variation	ATOMIC	ConceptNet
SIQA:		
QC-HQ CS-1	**51**%	39%
QC-HQ CS-2	**24**%	22%
QC-HQ CS-3	2%	**6**%
QC-HQ CS	**77**%	66%
PIQA:		
QC-HQ CS-1	16%	**24**%
QC-HQ CS-2	8%	8%
QC-HQ CS	24%	**32**%
MCScript2.0:		
QC-HQ CS-1	2%	**36**%
QC-HQ CS-2	**88**%	54%
QC-HQ CS	90%	90%

Table 2: %Knowledge extracted for each subset wrt. original data size. Results shown for the best aligned KG shape in the QC-HQ setting: SIQA=CN triples, ATOMIC paths; PIQA=CN subgraphs, ATOMIC pairs; MCScript2.0=CN subgraphs, ATOMIC pairs.

5 Phase 2: Align

5.1 Setup

Baseline Model: We fine-tune BERT-base (Devlin et al., 2019) as our baseline on SIQA following Sap et al. (2019b), BERT-base on MCScript2.0, and BERT-large (Devlin et al., 2019) as our baseline on PIQA following Koupaee and Wang (2018). See original papers for hyperparameter settings.

Knowledge-Surrounded (KS) Model: We enhance task candidate answers with knowledge-surrounded answers, in which answer-specific knowledge is appended to each candidate answer. This knowledge is intended to explicitly add missing knowledge gaps to the answer. This model is used in the *alignment* and *integration* steps in Fig. 1. We encode input as follows. For a question q_i, we modify each candidate answer, c_{ij}, by appending its respective knowledge k_{ij}. The sequence of tokens $\{$[CLS] q_i [SEP] $c_{ij}k_{ij}$ [SEP]$\}$ is then

Variation	Base.	KS+	KS-
SIQA AT. QC-HQ	38.1	**42.9**	40.5
SIQA AT. QC-HR	**60.3**	59.0	56.4
SIQA AT. A-HQ	**60.4**	60.0	59.2
SIQA AT. A-HR	**61.8**	60.8	61.0
SIQA CN QC-HQ	**54.1**	45.0	43.2
SIQA CN QC-HR	**54.5**	52.2	49.1
SIQA CN A-HQ	61.5	61.5	**61.9**
SIQA CN A-HR	**61.2**	60.0	59.7
PIQA AT. QC-HQ	**54.6**	49.3	50.0
PIQA AT. QC-HR	**61.8**	51.2	48.8
PIQA AT. A-HQ	50.6	61.2	**64.9**
PIQA AT. A-HR	**70.5**	64.9	65.4
PIQA CN QC-HQ	60.5	**66.5**	59.9
PIQA CN QC-HR	51.4	57.2	**58.7**
PIQA CN A-HQ	49.0	64.5	**66.4**
PIQA CN A-HR	**70.5**	54.5	51.9
PIQA WH QC-HQ	53.3	**54.7**	48.0
PIQA WH QC-HR	**59.9**	54.7	55.0
PIQA WH A-HQ	53.5	68.3	**69.0**
PIQA WH A-HR	**67.5**	51.4	49.3
MC AT. QC-HQ	80.3	**83.0**	79.6
MC AT. QC-HR	**82.4**	80.8	79.4
MC AT. A-HQ	**82.5**	80.9	79.6
MC AT. A-HR	**81.2**	80.0	77.6
MC CN QC-HQ	78.6	**79.4**	76.2
MC CN QC-HR	78.8	**79.3**	78.4
MC CN A-HQ	**82.6**	80.3	77.4
MC CN A-HR	**82.2**	80.2	76.9

Table 3: Accuracy for extraction variations on: SIQA CS-3 for ATOMIC paths & ConceptNet triples; PIQA CS-2 for ATOMIC pairs & ConceptNet subgraphs & WikiHow paths; and MCScript2.0 CS-2 for ATOMIC pairs & ConceptNet subgraphs.

passed as input to BERT. Thus, each candidate answer is surrounded by knowledge that allows BERT to potentially fill reasoning gaps between the question and answer. The extraction variations for k_{ij} are described in Section 4.1.

5.2 Analysis

We investigate how well the extracted knowledge and the task are aligned by allowing the knowledge to fill in the question-answer knowledge gap and determining whether this improves performance. This is illustrated in the center of Fig. 1.

5.2.1 Extraction Variation Analysis

Table 3 illustrates performance for each KG across each of the different extractions, using the subset in which each candidate answer has access to knowledge. We compare question-conditioned, unconditioned, high recall, and high quality settings for the best aligned knowledge shape (ConceptNet triples, ATOMIC paths for SIQA; ConceptNet subgraphs, ATOMIC pairs, WikiHow paths for PIQA; ConceptNet subgraphs, ATOMIC pairs for MCScript2.0) across the three KGs. For each setting, we show results both for when knowledge is present during inference time (KS+) and when it is not (KS-). We see that SIQA performed best when it received QC-HQ knowledge from ATOMIC, reflecting the strong, one-to-one alignment between SIQA and ATOMIC. PIQA, however, performs well across most extractions for ConceptNet, indicating that PIQA is generally well aligned with ConceptNet and only performs poorly when the extraction process becomes too noisy. Additionally, PIQA performs well with WikiHow for the unconditioned, high quality setting, indicating that the WikiHow KG is not well aligned across question-answer pairs, but does identify useful knowledge gaps within the answer that may improve performance on the task. Finally, we see that MCScript2.0 performed best when it received QC-HQ knowledge from ATOMIC, and similarly to SIQA, improves when seeing this knowledge at inference time.

5.2.2 Knowledge Shape Analysis

We discuss knowledge shape effects on alignment.

ATOMIC: For SIQA, ATOMIC paths have the best alignment, due to the high quality achieved when constraining knowledge for the SIQA question to link to knowledge for the answer. ATOMIC pairs and subgraphs seem to be learned more implicitly and do not yield large overall improvements when added explicitly during inference time. It seems that SIQA requires longer, more informative knowledge at inference time, which pairs and subgraphs do not offer. For example, consider the SIQA context and answer on the right of Fig 2. For this data point, we extract the following ATOMIC path: [PersonX has to go to the dentist, need to make an appointment, PersonX picks _ up from school, to drive kids home], and the following ATOMIC pair: [PersonX picks _ up from school, to drive kids home]. We can see that the path clearly contains more context and detail for the knowledge required to make the correct prediction. For PIQA, we saw the largest improvements for ATOMIC pairs and subgraphs, where pairs ultimately perform best, indicating that PIQA might find concise and direct information from ATOMIC more useful. For MCScript2.0, ATOMIC pairs aligned best exclusively.

ConceptNet: For SIQA, ConceptNet triples and subgraphs show similar alignment results and we do not see major improvements. It seems that the content of ConceptNet is not aligned well to SIQA, regardless of shape. For PIQA, we see improvements for both triples and subgraphs, and get our best improvements with subgraphs, indicating that the extra knowledge encoded in a subgraph shape via ConceptNet is helpful for the PIQA task. Similarly to SIQA, MCScript2.0 performed best with ConceptNet subgraphs, but these do not yield major improvements. Interestingly, results on MCScript2.0 only show slight improvement for ConceptNet when knowledge is present at inference time, and shows no improvement otherwise.

WikiHow: WikiHow paths performed best, indicating that paths were the best way to extract information, as WikiHow pairs and subgraphs might have contained redundant information given limitations with the WikiHow KG extraction process.

5.2.3 Knowledge Graph Analysis

Given our alignment results, it is clear to see that ATOMIC is the best match for SIQA, that ConceptNet is the best match for PIQA, and that ATOMIC is the best match for MCScript2.0 (most likely due to its need for script knowledge, which often requires social knowledge). The encoding of each KG plays an important role in this match. We see that the ConceptNet to PIQA match is more robust to extraction methods, which may be a side effect of ConceptNet's encoding, where directly linking nodes is less noisy than using tf-idf measures for the ATOMIC encoding, in which we only see positive results when we have very selective filters in our extraction techniques. The concise, short nature of ConceptNet's knowledge also lends itself to more implicit knowledge learning for certain types of tasks, whereas the more descriptive nature of ATOMIC can be read during inference time (see Fig 2 for examples). This illustrates the possibility that ConceptNet may boost performance as a regularizer for certain tasks.

6 Phase 3: Integrate

6.1 Setup

We analyze two aspects of integration, depicted on the far right in Fig. 1. First, we construct commonsense probes to demonstrate how much knowledge we gain from our KGs via our transformer-based KS model with respect to a BERT baseline. Second, we examine distributional changes in our models before and after commonsense integration and verify our results with human evaluation. With our probes, we can compare how well models distinguish between several types of ATOMIC-style knowledge, outlined below.

Type	Baseline	KS model
Relation Probes		
xWant vs xEffect	**53.66**	46.59
xWant vs xReact	51.66	**77.39**
xWant vs xIntent	**52.05**	45.77
xWant vs xNeed	47.99	**50.32**
xWant vs xAttr	**71.74**	65.19
xEffect vs xReact	43.32	**63.36**
xEffect vs xIntent	**54.29**	44.85
xEffect vs xNeed	49.22	**52.48**
xEffect vs xAttr	**61.52**	59.44
xReact vs xIntent	52.34	**73.32**
xReact vs xNeed	49.21	**74.76**
xReact vs xAttr	48.48	**57.71**
xIntent vs xNeed	**44.18**	42.24
xIntent vs xAttr	**78.96**	64.64
xNeed vs xAttr	**68.97**	68.49
oWant vs oEffect	**51.74**	41.83
oWant vs oReact	43.90	**72.04**
oEffect vs oReact	49.12	**59.02**
Agent-Patient Probes		
xWant vs oWant	**70.12**	70.09
xEffect vs oEffect	**74.34**	73.51
xReact vs oReact	**74.34**	69.51
Concept Probes		
inference	74.96	**77.40**
event	**67.74**	65.07

Table 4: QA probe results on ATOMIC-SIQA QC-HQ KS model vs BERT baseline.

Relational Probes: We predict the ATOMIC relation between an event and inference pair, constraining our candidate answer set to two specified inference dimensions. For example, `xWant` vs `xNeed` might refer to a probe that will predict an answer from the candidate set: [Person `wants` recognition, Person `needs` recognition] given some event, essentially pitting the two relations against each other to evaluate the difficulty of distinguishing.

Agent-Patient Probes: We predict the agent of the inference where the candidate set is the agent and patient of the event (using ATOMIC abstractions).

Concept Probes: We predict concepts and constrain our candidate answer set to the most salient concept in the sequence and its respective antonym. A full description of probe construction and examples for each knowledge type can be found in the appendix. We evaluate QA probes via standard accuracy after fine-tuning.

6.2 Analysis: Commonsense Probes

Table 4 compares the performance between the best performing ATOMIC-SIQA KS model and respective SIQA BERT baseline. We see notable performance differences for our relation probes, showing that the KS model does well at identifying the `React` relation and that the baseline tends

Correct	Incorrect
19.5%	12.4%

Table 5: Distribution changes for selected class from baseline to KS model.

to identify the `Attribute` relation well. While the KS model improves on several QA relational probes, the performance was comparable for agent-patient and world knowledge probes, indicating the knowledge gaps ATOMIC is better at resolving, i.e., relational (feelings, reactions, etc.) versus antonym/synonym information about concepts.

6.3 Analysis: Distribution Change

We conduct an integration analysis on our best ATOMIC-SIQA setting (QC-HQ). We examine 40 multiple choice questions and analyze KS model prediction changes with respect to the baseline. We observe that 93% of prediction changes were made because the new prediction's knowledge had the best reasoning flow to resolve a knowledge gap.

Fig. 1 defines our distribution change analysis as $\Delta p_{cs\_sel} = p^{cs}_{cs\_sel} - p^{base}_{cs\_sel}$. Here, $p^{cs}_{cs\_sel}$ indicates the KS Model's probability of selecting the KS Model's selected answer and $p^{base}_{cs\_sel}$ indicates the baseline's probability of selecting the KS Model's selected answer. Thus, Δp_{cs_sel} indicates the change in the probability of selection for the KS Model's selected answer before (Base Model) and after (KS Model) knowledge integration.

Table 5 shows the distribution change from the baseline to the KS model for the selected answer. When a switch became positive, the average probability increase of the selected ground truth candidate answer was 19.5%, whereas when a switch became negative the increase was 12.4%. Thus, the distribution change shows more confidence about ground truth selection with added knowledge, indicating that the quality of a ground truth's knowledge is higher than that of a negative candidate.

6.4 Analysis: Human Evaluation

We performed human evaluation on 100 SIQA, 100 PIQA, and 100 MCScript2.0 question-answer pairs to determine the validity of our process for both knowledge gap identification and alignment.[2] To show the validity of our QC-HQ extraction method as a measure for knowledge gap identification, we find that this extraction method is a valid potential SIQA knowledge gap identification 89% of the time for ATOMIC and 91% for ConceptNet. Valid, in this case, means that the correct concepts (that identify a relevant knowledge gap) were used to create a link. These results are found in Table 6. We also show that for our best ATOMIC extraction (QC-HQ), we extract the correct knowledge for the gap 66% of the time, demonstrating the connection between KS model improvement and alignment. Correct, in this case, means that the content of the link itself is relevant to resolve the commonsense gap. These results are found in Table 7. In contrast, we see that our best ConceptNet extraction (A-HQ) finds the correct knowledge for the gap 18% of the time. This is probably why we do not see much improvement when we give our ConceptNet KS model knowledge during inference time and why it seems to improve mostly via regularization.

	ATOMIC	CN
SIQA	89%	**91%**
PIQA	48%	**82%**
MC	**75%**	74%

Table 6: Human evaluation for valid knowledge gap.

	ATOMIC	CN
SIQA	**66%**	18%
PIQA	16%	**22%**
MC	31%	**43%**

Table 7: Human evaluation for correct knowledge gap.

On PIQA, we find that this extraction method is a valid potential knowledge gap identification 48% of the time for ATOMIC and 82% for ConceptNet. We conclude that if we do not see alignment improvement on the QC-HQ setting (as is true of ATOMIC-PIQA), then extraction does not indicate the best knowledge gap coverage. Additionally, we find that for our best ATOMIC extraction method (A-HQ), we extract the correct knowledge for the gap 16% of the time and that for our best ConceptNet extraction method (A-HQ), we extract the correct knowledge for the gap 22% of the time.

For MCScript2.0, we found the best empirical performance with QC-HQ settings for both ATOMIC and ConceptNet. With these settings, we found that a valid potential knowledge gap identification occurs 75% of the time for ATOMIC and 74% for ConceptNet. Additionally, we find that with ATOMIC, we extract the correct knowledge for the gap for 31% of examples, and with ConceptNet for 43%. The higher correct extractions for

---

[2] Done by expert authors since this is time-consuming, fine-grained verification analysis (instead of model evaluation).

Type	BERT	RoBERTa
Relation Probes		
xWant vs xEffect	**60.90**	56.68
xWant vs xReact	94.15	**94.96**
xWant vs xIntent	57.02	**57.10**
xWant vs xNeed	57.13	**59.52**
xWant vs xAttr	**87.73**	86.68
xEffect vs xReact	**60.88**	57.93
xEffect vs xIntent	**78.74**	71.79
xEffect vs xNeed	**58.38**	50.49
xEffect vs xAttr	**58.18**	58.13
xReact vs xIntent	84.97	**92.59**
xReact vs xNeed	**94.63**	94.29
xReact vs xAttr	56.72	**57.27**
xIntent vs xNeed	51.23	**52.67**
xIntent vs xAttr	61.57	**64.73**
xNeed vs xAttr	**83.51**	77.87
oWant vs oEffect	**61.71**	61.61
oWant vs oReact	**94.80**	90.97
oEffect vs oReact	60.46	**61.00**
Agent-Patient Probes		
xWant vs oWant	**65.79**	46.71
xEffect vs oEffect	32.24	**60.67**
xReact vs oReact	**62.64**	52.10
Concept Probes		
inference	79.15	**80.68**
event	**68.91**	68.51

Table 8: MLM probe zero-shot results.

ConceptNet are most likely due to the best performing extraction settings being QC-HQ. ATOMIC QC-HQ settings visibly outperform the baseline empirically, whereas ConceptNet QC-HQ settings perform only slightly better. This may be due to the fact that MCScript2.0 has a much larger context than any of our other datasets, and thus a model may already be able to implicitly infer the explicit taxonomic knowledge offered by ConceptNet.

7 MLM Commonsense Probes

We evaluate the transformer-based models used in our setup to assess how much knowledge LMs already know and how easy it is for them to learn.

7.1 Setup

We examine our MLM probes in two settings: zero-shot and fine-tuned. For the zero-shot setting, we use a pre-trained LM without any fine-tuning. This is to examine how much knowledge a pre-trained transformer model already holds. For the fine-tuned setting, we train on each probe's respective train set and evaluate using the same metrics as in Talmor et al. (2019a). This is to examine how fast a model learns given its encoding before fine-tuning. Results, set up, metrics, and analysis for fine-tuned settings are found in the appendix.

7.2 Results

Table 8 compares the performance of BERT and RoBERTa for zero-shot results. Majority label results are found in the appendix.

Zero-shot Results: RoBERTa and BERT perform comparably for most Relation probes. While performance is poor for most settings, both models perform very well at discerning between `Want` and `React` (`xWant` vs `xReact`, `oWant` vs `oReact`), and between `xReact` vs `xNeed`. Both models perform reasonably well at discerning `Attr` and `Intent` from other dimensions in certain settings (`xWant` vs `xAttr`, `xNeed` vs `xAttr`, `xEffect` vs `xIntent`, `xReact` vs `xIntent`). In general, the models seem to most consistently discern `React` from other dimensions. Finally, both models perform comparably and reasonably well on Concept probes, whereas the performance for Agent-Patient probes differs largely between the models and is often poor.

8 Conclusion

We proposed a method to analyze how well a candidate KG can correctly identify and accurately fill in gaps of reasoning for a given task. We presented a three step approach for analyzing this KG-to-task match via identification, alignment, and integration. We found that the ATOMIC KG aligns best with the SIQA task, and quantitatively analyze the quality of the extracted commonsense. We also found that the ConceptNet and WikiHow based KGs match best with the PIQA task. Finally, we see that the ATOMIC KG also aligns best with MCScript2.0, which was a novel discovery and is most likely a result of the task's script knowledge requirement. We demonstrate the knowledge contained and learned by our KS model via our commonsense probes, illustrating what knowledge transformer-based models already know and what they can learn. This analysis can be extended to any set of tasks and KGs to analyze match potential.

Acknowledgments

We thank the reviewers for their useful feedback. This work was supported by DARPA MCS Grant #N66001-19-2-4031, NSF-CAREER Award 1846185, NSF PhD Fellowship, and awards from Microsoft and Amazon. The views are those of the authors and not of the funding agency.

References

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *ICLR*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. *AAAI*.

Antoine Bosselut and Yejin Choi. 2019. Dynamic knowledge graph construction for zero-shot commonsense question answering. *arXiv preprint arXiv:1911.03876*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2017. Neural natural language inference models enhanced with external knowledge. *arXiv preprint arXiv:1711.04289*.

Jeff Da and Jungo Kasai. 2019. Understanding commonsense inference aptitude of deep contextual representations. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 1–12.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL-HLT*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Sunjae Kwon, Cheongwoong Kang, Jiyeon Han, and Jaesik Choi. 2019. Why do masked neural language models still need common sense knowledge? *arXiv preprint arXiv:1911.03024*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019a. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2822–2832.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. *arXiv preprint arXiv:2005.00683*.

Bill Yuchen Lin, Ming Shen, Wangchunshu Zhou, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2019b. Commongen: A constrained text generation challenge for generative commonsense reasoning. *CoRR*, abs/1911.03705.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Arindam Mitra, Pratyay Banerjee, Kuntal Kumar Pal, Swaroop Mishra, and Chitta Baral. 2019. Exploring ways to incorporate additional knowledge to improve natural language commonsense question answering. *AAAI*.

Sayantan Mitra, Mohammed Hasanuzzaman, Sriparna Saha, and Andy Way. 2018. Incorporating deep visual features into multiobjective based multi-view search results clustering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3793–3805, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Simon Ostermann, Michael Roth, and Manfred Pinkal. 2019. Mcscript2. 0: A machine comprehension corpus focused on script events and participants. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (* SEM 2019)*, pages 103–117.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942.

Kyle Richardson and Ashish Sabharwal. 2019. What does my qa model know? devising controlled probes using expert knowledge. *arXiv preprint arXiv:1912.13337*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4453–4463.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *ACL*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019a. olmpics–on what language model pre-training captures. *arXiv preprint arXiv:1912.13283*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019b. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete kbs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4258–4264.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *NeurIPS*.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. *arXiv preprint arXiv:1908.06725*.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6713–6724.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104.

Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. Winowhy: A deep diagnosis of essential commonsense knowledge for answering winograd schema challenge. *arXiv preprint arXiv:2005.05763*.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. In *ACL*.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. “Going on a vacation” takes longer than “Going for a walk”: A Study of Temporal Commonsense Understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Pei Zhou, Rahul Khanna, Bill Yuchen Lin, Daniel Ho, Xiang Ren, and Jay Pujara. 2020a. Can bert reason? logically equivalent probes for evaluating the inference capabilities of language models. *arXiv preprint arXiv:2005.00782*.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020b. Evaluating commonsense in pretrained language models. *AAAI*.

A Appendix

A.1 WikiHow Subgraph Extraction Procedure

We extract PIQA-conditioned WikiHow subgraphs for each PIQA datapoint. We do this in three steps:

1. Given a PIQA goal, extract relevant titles in Wikihow via tf-idf.
2. Dependency parse the PIQA goal, extracted

Type	Data Size
xWant vs xEffect	10012
xWant vs xReact	10693
xWant vs xIntent	9790
xWant vs xNeed	9829
xWant vs xAttr	11730
xEffect vs xReact	9519
xEffect vs xIntent	8616
xEffect vs xNeed	8655
xEffect vs xAttr	10556
xReact vs xIntent	9297
xReact vs xNeed	9336
xReact vs xAttr	11237
xIntent vs xNeed	8433
xIntent vs xAttr	10334
xNeed vs xAttr	10334
oWant vs oEffect	3873
oWant vs oReact	3873
oEffect vs oReact	3685
xWant vs oWant	7974
xEffect vs oEffect	5911
xReact vs oReact	7293
inference	8401
event	13228

Table 9: MLM & QA dev probe sizes.

Wikihow title, and each sentence in the title's corresponding paragraph.
3. Find overlapping concepts in the dependency parses for which to create concept nodes. Then, create a graph by combining all possible edges for a concept node (found in all dependency parses).

A.2 Probe Construction Details

A.2.1 QA

We use ATOMIC events and respective inferences and map them into the following QA formats.

Relational Probes. For relational probes, we state the event and follow it with "What happens next?". We then create candidates out of a corresponding inference, each with a different relation. For example,

PersonX puts out a fire. What happens next?

For the probe `xWant` vs `xNeed`, we select from the candidate answers: *[PersonX wants to receive recognition, PersonX needs to receive recognition]*, where our ground truth is *PersonX wants to receive recognition*.

Agent-Patient Probes. ATOMIC inference dimensions are either assigned to "PersonX" (most often the agent of the event, unless otherwise specified) or "others" (who are often influenced by the effects of PersonX's actions and may not be directly referred to in the event, see original ATOMIC paper for more details). For Agent-Patient probes, we state the event and follow it with "Who [relation] [inference]"? Using our previous example, we have the following:

PersonX puts out a fire. Who wants to receive recognition?

Where our candidate answers are: *[PersonX, others]*.

Concept Probes. For event concept prediction, we first state the inference and then ask "What happened?" We then create event candidates with a ground truth salient concept (determined in the same way as the MLM salient concepts described in the next section) and an antonym concept. For example:

PersonX wants to receive recognition. What happened?

Where our candidate answers would be: *[PersonX puts out a fire, PersonX puts out a water]*.

For inference concept prediction, we state the event and follow it with "What happens next?". We then create inference candidates with a ground truth salient concept and an antonym concept. For example:

PersonX puts out a fire. What happens next?

Where our candidate answers would be: *[PersonX wants to receive recognition, PersonX wants to give recognition]*.

A.2.2 MLM

Relational Probes. For inference dimensions relating to PersonX, we map the inference dimensions `xWant`, `xNeed`, `xIntent`, `xReact`, `xEffect`, `xAttr` onto the verbs *wants*, *needs*, *intends*, *feels*, *effect*, and *is*. For example, given the event:

PersonX puts out a fire.

We have the following inference in the `xWant` dimension: *to receive recognition*. We map this onto text for the following probe:

PersonX puts out a fire. PersonX [MASK] to receive recognition.

The correct prediction for [MASK] is *wants*. So for the probe `xWant` vs `xNeed`, our candidate answers for predicting the mask would be *[wants, needs]*.

		BERT-FT		RoBERTa-FT	
Type	**majority**	**max**	**WS**	**max**	**WS**
Relation Probes					
xWant vs xEffect	56	95.25	93.56	95.45	93.95
xWant vs xReact	52	99.10	98.70	99.21	98.32
xWant vs xIntent	57	75.30	66.94	70.11	59.02
xWant vs xNeed	57	82.17	72.48	86.70	70.66
xWant vs xAttr	52	99.19	99.11	99.22	98.81
xEffect vs xReact	54	98.17	96.73	98.18	93.32
xEffect vs xIntent	51	95.89	93.93	96.30	93.50
xEffect vs xNeed	51	95.22	93.48	95.64	93.86
xEffect vs xAttr	58	98.44	97.58	98.58	97.57
xReact vs xIntent	55	97.96	97.45	97.93	93.76
xReact vs xNeed	55	99.27	98.79	99.41	98.60
xReact vs xAttr	55	80.54	76.48	79.82	75.65
xIntent vs xNeed	50	87.81	77.81	90.00	72.19
xIntent vs xAttr	59	98.42	98.08	98.51	97.78
xNeed vs xAttr	59	99.14	98.97	99.16	98.54
oWant vs oEffect	61	93.90	91.21	93.98	90.78
oWant vs oReact	52	99.10	98.49	99.08	98.81
oEffect vs oReact	60	97.29	96.26	97.72	96.55
Agent-Patient Probes					
xWant vs oWant	70	84.46	76.62	83.70	72.45
xEffect vs oEffect	75	84.99	77.58	83.45	76.67
xReact vs oReact	70	78.77	72.49	70.88	70.11
Concept Probes					
inference	-	94.99	91.82	95.26	87.06
event	-	98.47	91.82	97.41	91.64

Table 10: Majority label and fine-tuning results for MLM probes.

For inference dimensions relating to Others, we map the inference dimensions `oWant`, `oReact`, and `oEffect` to the same verbs as before: *wants*, *feels*, and *effect* respectively. We set up probes in the same way as above.

Agent-Patient Probes. We create probes to evaluate whether a model can determine whether an inference dimension is assigned to PersonX or others. For example, consider the following probe:

PersonX puts out a fire. [MASK] wants to receive recognition.

In this example, the correct prediction is *PersonX*. However, in the below probe, the correct prediction is *others*.

PersonX puts out a fire. [MASK] want to thank PersonX.

Both of these probes use the following answer candidates: *[PersonX, others]*. We also remove plurals to ensure that the model does not make predictions using hints from grammar.

Concept Probes. We investigate two kinds of concepts in our probe: event concepts and inference concepts. In event concept probe construction, we find the most salient concept in the event via POS tagging. We then replace this concept with [MASK] and set candidate answers as the ground truth answer and an antonym, as found via WordNet (Miller, 1995). For example, given the event and inference:

PersonX discovers the answer. PersonX feels accomplished.

We identify *discovers* as the most salient concept in the event, and use the lemma from WordNet: *discovery*. We then use this to find a viable antonym: *lose*. Finally, we have the probe:

PersonX [MASK] the answer. PersonX feels accomplished.

And the candidates: *[discovery, lose]*.

We lemmatize the answers to allow for fair prediction between the truth concept and the antonym (which often comes lemmatized from Wordnet).

Similarly, we construct inference concept probes by predicting salient concepts in the inference dimension instead of the event.

A.2.3 Training Setup

We create a training and a development set for each of our probes using the ATOMIC train and dev set.

We show the sizes of our probe dev sets in Table 9. The sizes are directly derived from ATOMIC dev sizes. We train each model using 1 GeForce GTX 1080 Ti GPU.

A.3 Commonsense MLM Probes: Fine-tuned

A.3.1 Setup

We evaluate our MLM probes in a fine-tuned setting. We train on each probe's respective training set and evaluate the max and WS as in Talmor et al. (2019a), which defines (1) max as the maximal accuracy on the learning curve and (2) WS as the weighted average of accuracies on the learning curve, where higher weights are assigned to earlier points on the curve. This is to examine how fast the model learns given its encoding before fine-tuning.

A.3.2 Results

Table 10 compares the performance of BERT and RoBERTa for fine-tuned results.

Fine-tuning Results: After fine-tuning on probe training sets, both models do not fully solve the following categories: `xWant` vs `xIntent`, `xWant` vs `xNeed`, `xReact` vs `xAttr`, and `xIntent` vs `xNeed`. This demonstrates that these commonsense knowledge categories are difficult to learn even with fine-tuning. Additionally, BERT does learn faster than RoBERTa for the following Subject Probes: `xWant` vs `xIntent`, `xNeed` vs `xIntent`, and `xReact` vs `xIntent`. Overall, this illustrates that BERT and RoBERTa do not capture much ATOMIC commonsense in a zero-shot setting, and that many of these relations are difficult to learn even with fine-tuning, including nuanced relations like `xWant` vs `xIntent` and `xWant` vs `xNeed`. Agent-Patient relations seem difficult to learn and do not achieve high final results. Similarly, BERT and RoBERTa perform poorly on Concept Probes in zero-shot setting, however seem to learn these quickly with high final results. Overall, it seems that both models perform comparably for most probes.

A.4 Reproducibility

We train each KS model using 2 GeForce GTX 1080 Ti GPUs. Hyperparameter settings are used from previously reported BERT results on each task (Sap et al., 2019b; Bisk et al., 2020; Da and Kasai, 2019).

Calculating the optimal step of arc-eager parsing for non-projective trees

Mark-Jan Nederhof
School of Computer Science
University of St Andrews, UK
`markjan.nederhof@googlemail.com`

Abstract

It is shown that the optimal next step of an arc-eager parser relative to a non-projective dependency structure can be calculated in cubic time, solving an open problem in parsing theory. Applications are in training of parsers by means of a 'dynamic oracle'.

1 Introduction

A deterministic transition-based dependency parser is often driven by a classifier that determines the next step, given features extracted from the current configuration (Nivre et al., 2004). The classifier may be trained on parser configurations and steps that exactly correspond to 'gold' trees from a treebank. However, better accuracy is generally obtained by also including configurations reached by letting the parser stray from the gold trees, to let the classifier learn how best to recover from any mistakes. This is associated with the term *dynamic oracle*. If the parser is projective while the gold trees are non-projective moreover, then it is unavoidable that configurations be considered that do not correspond to the gold trees.

Determining the desired output of the classifier requires calculation of the best next step given an arbitrary configuration and a gold tree. Typically, this is the step that allows the most accurate tree to be reached, in terms of the gold tree.[1]

For a gold tree that is projective, the optimal step can be determined in linear time for arc-eager parsing (Goldberg and Nivre, 2012, 2013) and for shift-reduce parsing (Nederhof, 2019). For a non-projective gold tree, the optimal step can be determined for several types of non-projective parsers (Gómez-Rodríguez et al., 2014; Gómez-Rodríguez and Fernández-González, 2015; de Lhoneux et al., 2017; Fernández-González and Gómez-Rodríguez, 2018), as well as for shift-reduce parsing (Nederhof, 2019). However, for arc-eager parsing, the problem has been unsolved until now. Aufrant et al. (2018) propose an *approximation* of the optimal step, based on the procedure for projective gold trees, and demonstrate the advantages of training a projective parser directly on non-projective trees.

The current paper introduces an *exact* calculation of the optimal step for arc-eager parsing and a non-projective gold tree, within the same framework as Nederhof (2019), which consists of a generic cubic-time tabular dependency parsing algorithm and a fixed context-free grammar that is applied on a string extracted from the current configuration, with edge weights determined by the gold tree.

For arc-eager parsing, the context-free grammar is considerably more complex than in the case of shift-reduce parsing. This is a consequence of theoretical properties of arc-eager parsing, which we first need to investigate in detail before we can define 'optimality' of the next step.

2 Dependency structures

Let $w = a_1 \cdots a_n$ be a sentence consisting of n tokens, which can be words or punctuation. Where we use indices between 1 and n, we also refer to these as tokens, relying on the assumption that given an index i we can retrieve the actual token a_i. An additional index 0 represents an imaginary token prepended to the sentence.

An *unlabeled dependency structure* T for w is an unlabeled tree with $\{0, 1, \ldots, n\}$ as the set of nodes, of which 0 is the root. We represent such a tree as a set of edges, each represented as a pair (a, b), where index a is the *parent* and index b is the *child*. The *descendants* of a node are the node itself and the descendants of its children. A dependency structure is *projective* if the set of de-

---

[1] There are alternative perspectives on how to determine the best next step; cf. Straka et al. (2015).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2273–2283
April 19 - 23, 2021.

scendants of each node in the tree can be written as $\{i, i+1, \ldots, j-1, j\}$ for some i and j $(0 \leq i \leq j \leq n)$.

We assume each sentence $w = a_1 \cdots a_n$ has a distinguished *gold* tree T_g. The *score* of an arbitrary tree T for the same w is defined as $|T \cap T_g|$. The *accuracy* of T is its score divided by n.

A dependency parser is usually designed to find a tree that is as accurate as possible, given an input sentence. Such a parser can generally be extended in a natural way to find a *labeled* dependency structure, which is analogously defined as a *labeled* tree with root 0. An edge label in such a structure is called a *dependency relation*. This paper focuses on unlabeled dependency parsing.

3 Transition-based dependency parsing

Transition-based dependency parsing is commonly formalized in terms of a set of configurations and a finite set of transitions between configurations. For now, a *configuration* for input sentence $w = a_1 \cdots a_n$ is a 3-tuple (α, β, T), where α is the *stack*, β is the *remaining input*, and T is a subset of (the set of edges of) a dependency structure. We assume $\alpha\beta$ is a subsequence of $0\ 1\ \cdots\ n$, and β is more specifically a suffix of $1\ \cdots\ n$. A *transition* is a partial function, mapping one configuration to another. A *step* is one application of a transition. A *computation* is a sequence of steps, starting with the *initial* configuration $(0, 1\ \cdots\ n, \emptyset)$ and ending in a *final* configuration $(0, \varepsilon, T)$ where ε denotes the empty string; here T is the resulting tree.

A transition may have a *precondition*, i.e. a condition on the current configuration that must hold for the transition to be applicable. Unrestricted preconditions are less than convenient for our purposes, and therefore we opt for a more uniform framework, in which a stack element is a pair (a, A) consisting of a token a and a *label* A taken from a fixed set.[2] To avoid clutter, we write a^A in place of (a, A); this also emphasizes the relation to more traditional formulations of dependency parsing, which are obtained by omitting the superscripts.

A first illustration of this is traditional shift-reduce dependency parsing, defined by the transitions in Table 1, here without labels, or alternatively, one may consider there to be only one such label, which is left implicit.[3]

shift:
$(\alpha,\ b\beta,\ T) \vdash^{\text{SH}} (\alpha b,\ \beta,\ T)$
reduce_left:
$(\alpha a_1 a_2,\ \beta,\ T) \vdash^{\text{RL}} (\alpha a_1,\ \beta,\ T \cup \{(a_1, a_2)\})$
reduce_right:
$(\alpha a_1 a_2,\ \beta,\ T) \vdash^{\text{RR}} (\alpha a_2,\ \beta,\ T \cup \{(a_2, a_1)\})$,
if $|\alpha| > 0$

Table 1: Shift-reduce dependency parsing.

shift:
$(\alpha a^C,\ b\beta,\ T) \vdash^{\text{SH}} (\alpha a^C b^N,\ \beta,\ T)$
complete:
$(\alpha a^N,\ \beta,\ T) \vdash^{\text{CO}} (\alpha a^C,\ \beta,\ T)$
reduce_left:
$(\alpha a_1^C a_2^C,\ \beta,\ T) \vdash^{\text{RL}} (\alpha a_1^C,\ \beta,\ T \cup \{(a_1, a_2)\})$
reduce_right:
$(\alpha a_1^C a_2^N,\ \beta,\ T) \vdash^{\text{RR}} (\alpha a_2^N,\ \beta,\ T \cup \{(a_2, a_1)\})$,
if $|\alpha| > 0$

Table 2: Shift-reduce dependency parsing enforcing the left-before-right policy.

This form of parsing suffers from spurious ambiguity in that left and right children may be attached in different orders. E.g. if token b has left child a_1 and right child a_2, then after a **shift** of a_1 and b, there may be a **reduce_right** followed by a **shift** of a_2 followed by a **reduce_left**. Or there may be a **shift** of a_2 followed by a **reduce_left** followed by a **reduce_right**. This can be resolved by requiring that left children are attached before right children are. In our framework, this left-before-right policy can be enforced by introducing a label C, which is given to a token to signal that it is 'complete' with regard to its left children. Initially, shifted tokens carry label N (for 'no restriction'). The 0 token always has label C. This results in Table 2.

There is a simple one-to-one correspondence between a computation according to Table 2 and a computation according to Table 1 satisfying the left-before-right policy. The difference is merely an application of **complete** just before a token ceases to be a topmost stack element, either because it is reduced into the token to its left, or because another token is pushed on top. If a token

---

[2] There is a close connection to bilexical context-free grammars (Eisner and Satta, 1999), on the basis of which one may alternatively choose to refer to such a label as a 'delexicalized stack symbol', in a kind of lexicalized pushdown automaton.

[3] Shift-reduce dependency parsing has been known at least since Fraser (1989) and Nasr (1995). It is also referred to as 'arc-standard' parsing (Nivre, 2008).

shift:
$(\alpha,\ b\beta,\ T) \vdash^{\text{SH}} (\alpha b,\ \beta,\ T)$
left_arc:
$(\alpha a,\ b\beta,\ T) \vdash^{\text{LA}} (\alpha,\ b\beta,\ T \cup \{(b,a)\}),$
$\quad$ if $a \neq 0 \wedge \nexists a'[(a',a) \in T]$
right_arc:
$(\alpha a,\ b\beta,\ T) \vdash^{\text{RA}} (\alpha ab,\ \beta,\ T \cup \{(a,b)\}),$
$\quad$ if $\nexists a'[(a',b) \in T]$
reduce:
$(\alpha a,\ \beta,\ T) \vdash^{\text{RE}} (\alpha,\ \beta,\ T),$
$\quad$ if $\exists a'[(a',a) \in T]$

Table 3: Arc-eager parsing (Nivre, 2008, p. 525).

shift:
$(\alpha,\ b\beta,\ T) \vdash^{\text{SH}} (\alpha b^L,\ \beta,\ T)$
left_arc:
$(\alpha a^L,\ b\beta,\ T) \vdash^{\text{LA}} (\alpha,\ b\beta,\ T \cup \{(b,a)\})$
right_arc:
$(\alpha a^X,\ b\beta,\ T) \vdash^{\text{RA}} (\alpha a^X b^R,\ \beta,\ T \cup \{(a,b)\})$
reduce:
$(\alpha a_1^X a_2^R,\ \beta,\ T) \vdash^{\text{RE}} (\alpha a_1^X,\ \beta,\ T)$
reduce_correct:
$(\alpha a_1^X a_2^L,\ \beta,\ T) \vdash^{\text{RC}} (\alpha a_1^X,\ \beta,\ T \cup \{(a_1,a_2)\})$

Table 4: Reformulated arc-eager parsing, with $X \in \{R, L\}$, with an extra fifth transition needed to make it work in practice.

becomes non-topmost and reappears later on top of the stack, after applications of **reduce_left** that give it right children, then it will still have label C, which prevents it from taking further left children.

Table 3 is almost verbatim the formulation of arc-eager parsing by Nivre (2008), except that we renamed symbols, and we ignore dependency relations; the formulations by e.g. Nivre (2003, 2004) and Nivre et al. (2004) are largely equivalent. It is easy to see that the condition $\nexists a'[(a',b) \in T]$ for **right_arc** is redundant, as no tokens in the remaining input can obtain parents before they are shifted to the stack.

Taking shift-reduce parsing as starting point, **reduce_right** corresponds roughly to **left_arc**, while the role of **reduce_left** is only partly fulfilled by **right_arc**, which postulates that b is a right child of a, but without removing b as yet, allowing b to take right children, and only later is that b removed by **reduce**. Here shift-reduce parsing would postpone the decision whether that b is the left or the right child of its parent until all descendants of b have been shifted and reduced into it.

From the perspective of parsing of artificial languages (Sippu and Soisalon-Soininen, 1990), this is counter-intuitive. The conventional wisdom of deterministic parsing is that one should postpone commitment to occurrences of grammar rules (or here, dependency edges) for as long as possible, until enough information is available to resolve any local ambiguity, assuming left-to-right processing of the input string, and the ability to inspect only the top of the stack and the next k tokens of the remaining input, for a fixed, small number k.

Two arguments have been given why arc-eager parsing is nonetheless superior for processing natural language. The first is that this earlier commitment made by **right_arc**, in terms of the earlier creation of the dependency edge, offers additional information about the tree under construction, to better predict the next steps, using some type of classifier. The second argument in favor of arc-eager parsing is that the earlier creation of the dependency edge ensures that the partial tree under construction remains as connected as possible, which may help simultaneous syntactic and semantic processing. See Nivre (2004, 2008) and Damonte et al. (2017) for related discussions.

Next we rephrase arc-eager parsing to use labels to express preconditions, to prepare us for Section 4. Note that a token is transferred from the remaining input to the stack by either **shift** or **right_arc**. In the former case, it must eventually become a left child of its parent, and in the latter case, it becomes a right child. We use labels L and R for these cases.[4] In a configuration with set T as third element, existence of a stack element a^L implies $\nexists a'[(a',a) \in T]$ and a^R implies $\exists a'[(a',a) \in T]$. We thereby obtain the first four transitions in Table 4. Token 0 always has label R, and cannot be popped by **reduce** due to the a_1^X. The fifth transition will be discussed later.

Arc-eager parsing in either of the above two formulations cannot work in practice. The problem is illustrated in Table 5. In the last configuration, none of the steps is applicable. The situation arises when the remaining input becomes empty while there is a L label *anywhere* in the stack. Assuming the classifier used for predicting the next step cannot look unboundedly deep in the stack, this problem is unavoidable.

---

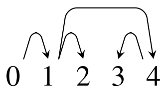
[4] L and R are akin to 0 and 1 from Kuhlmann et al. (2011).

$$\begin{array}{llll}
(0^R & , \; 1\,2\,3, & \emptyset &) \vdash^{\mathrm{SH}} \\
(0^R\,1^L & , \; 2\,3, & \emptyset &) \vdash^{\mathrm{RA}} \\
(0^R\,1^L\,2^R & , \; 3, & \{(1,2)\} &) \vdash^{\mathrm{RA}} \\
(0^R\,1^L\,2^R\,3^R, & , & \{(1,2),(2,3)\} &) \vdash^{\mathrm{RE}} \\
(0^R\,1^L\,2^R & , & \{(1,2),(2,3)\} &) \vdash^{\mathrm{RE}} \\
(0^R\,1^L & , & \{(1,2),(2,3)\} &)
\end{array}$$

Table 5: Arc-eager parsing is stuck in a configuration without applicable transitions.

shift:
$(\alpha,\ b\beta,\ T) \vdash^{\mathrm{SH}} (\alpha b^L,\ \beta,\ T)$
left_arc:
$(\alpha a^L,\ b\beta,\ T) \vdash^{\mathrm{LA}} (\alpha,\ b\beta,\ T \cup \{(b,a)\})$
right_arc:
$(\alpha a^X,\ b\beta,\ T) \vdash^{\mathrm{RA}} (\alpha a^X b^R,\ \beta,\ T)$
reduce:
$(\alpha a_1^X a_2^Y,\ \beta,\ T) \vdash^{\mathrm{RE}} (\alpha a_1^X,\ \beta,\ T \cup \{(a_1,a_2)\})$

Table 6: Corrected arc-eager parsing, with $X, Y \in \{R, L\}$.



$$\begin{array}{lllll}
& (0^R\,1^R\,2^R & , 3\,4, & \emptyset &) \vdash^{\mathrm{SH}} \\
& (0^R\,1^R\,2^R\,3^L, & \;\; 4, & \emptyset &) \vdash^{\mathrm{LA}} \\
& (0^R\,1^R\,2^R & , \;\; 4, & \{(4,3)\} &) \vdash^{\mathrm{RE}} \\
& (0^R\,1^R & , \;\; 4, & \{(1,2),(4,3)\} &) \vdash \ldots \\
\text{or} & & & & \\
& (0^R\,1^R\,2^R & , 3\,4, & \emptyset &) \vdash^{\mathrm{RE}} \\
& (0^R\,1^R & , 3\,4, & \{(1,2)\} &) \vdash^{\mathrm{SH}} \\
& (0^R\,1^R\,3^L & , \;\; 4, & \{(1,2)\} &) \vdash^{\mathrm{LA}} \\
& (0^R\,1^R & , \;\; 4, & \{(1,2),(4,3)\} &) \vdash \ldots
\end{array}$$

Table 7: Same structure obtained in two ways.

One possible fix is to add the **unshift** transition of Nivre and Fernández-González (2014); see also Honnibal and Johnson (2015). As this causes considerable complications to our framework, we will solve this in another way, reminiscent of Honnibal et al. (2013), which also helps to make a connection with shift-reduce parsing later. Our proposed fix is to allow a **reduce** even if the top of stack has label L, by means of the fifth transition of Table 4, **reduce_correct**. This transition is not needed during training if only computations are considered that most straightforwardly correspond to gold trees, with **left_arc** applied only on a token that is to become the left child of its parent. This may mean however that, in the case of labeled dependency parsing, the trained classifier has no basis to predict the dependency relation of the edge created by this transition when applied during testing. This can be solved by moving the creation of the edge from **right_arc** to **reduce**, and by then merging **reduce** and **reduce_correct**, as in Table 6.

This formulation at first sight appears to nullify the property that has been argued to give arc-eager an advantage over shift-reduce parsing, namely the early availability of edges connecting a parent and a right child. However, these edges are still identified by investigating which tokens in the stack have label R: their parent is the token immediately left to it in the stack. In other words, a classifier predicting the next step can be made to have access to the exact same feature values as before.

This formulation of arc-eager parsing, as do the original formulations (Goldberg and Nivre, 2012, p. 963), allow the same dependency structure to be obtained in two different ways; cf. Table 7. There are few studies that compare parsing accuracy between the two ways of resolving this, by preferring either **shift** before **reduce**, or **reduce** before **shift**, and some literature suggests the choice is arbitrary,[5] although the results from one study (Qi and Manning, 2017) suggest the shift-before-reduce policy could be slightly better.

One way to enforce the reduce-before-shift policy is to opt for a different division of labor between stack and the leftmost token of the remaining input, whereby we must shift a node from remaining input to stack before it obtains its first left child or before it is decided whether it is to be a left child or a right child. Table 8 presents this *normalized* arc-eager parsing. The **shift** is now simply the transfer of a token from remaining input to stack, without making a commitment whether it is to become a left or right child, which is indicated by the N label. Where before we had **shift** and **right_arc**, we now have **left_child** and **right_child**, which commit a token on top of the stack to be a left or right child of its parent, respectively. Where before we had **left_arc**, we now have **reduce_right**, and **reduce** is more appropriately renamed to **reduce_left**. There is a simple one-to-one correspondence between a computation according to Table 8 and a computation according to Table 6 satisfying the reduce-before-

---

[5] Cf. “harmless SHIFT-REDUCE conflicts” (Nivre, 2006, p. 98).

shift:
$(\alpha a^X,\ b\beta,\ T) \vdash^{\text{SH}} (\alpha a^X b^N,\ \beta,\ T)$
left_child:
$(\alpha a^N,\ \beta,\ T) \vdash^{\text{L}} (\alpha a^L,\ \beta,\ T)$
right_child:
$(\alpha a^N,\ \beta,\ T) \vdash^{\text{R}} (\alpha a^R,\ \beta,\ T)$
reduce_left:
$(\alpha a_1^X a_2^Y,\ \beta,\ T) \vdash^{\text{RL}} (\alpha a_1^X,\ \beta,\ T \cup \{(a_1, a_2)\})$
reduce_right:
$(\alpha a_1^L a_2^N,\ \beta,\ T) \vdash^{\text{RR}} (\alpha a_2^N,\ \beta,\ T \cup \{(a_2, a_1)\})$

Table 8: Normalized arc-eager parsing, with $X, Y \in \{R, L\}$.

shift policy. Moreover, the same feature values can be used, albeit after renaming. Specifically, if the top of stack has label N, then e.g. a feature referring to the top of stack in the case of Table 6 should instead refer to the first element underneath the top of stack in the case of Table 8.[6]

One advantage of the normalized formulation is that it clearly reveals the relation to shift-reduce parsing. In particular, instead of **complete** in Table 2, we have the more specific **left_child** and **right_child**, of which the latter constitutes the early commitment of a token to be right child, as explained before.

4 Calculating the optimal step

Assume there are τ transitions, denoted by $\vdash^1, \ldots, \vdash^\tau$. Let $\vdash$ represent an application of any of these transitions, and let $\vdash^*$ denote the reflexive transitive closure of $\vdash$. For a given configuration (α, β, T) for input sentence w with gold tree T_g, there are up to τ steps $(\alpha, \beta, T) \vdash^i (\alpha_i, \beta_i, T_i)$, $i = 1, \ldots, \tau$. For each of these, the *score* is the maximal $\rho_i = |T'_i \cap T_g|$ with $(\alpha_i, \beta_i, T_i) \vdash^* (0^R, \epsilon, T'_i)$ for some T'_i; if $\vdash^i$ is not applicable on (α, β, T), or if no final configuration is reachable from (α_i, β_i, T_i), then we set $\rho_i = -\infty$. The task is now to compute that ρ_i for each i. This determines which transition to apply next, to eventually obtain the highest-scoring tree, irrespective of any 'incorrect' steps performed in the past, that is, steps that were inconsistent with the gold tree. Because $|T \cap T_g|$ is the same for all i, and because the value of $|T''_i \cap T_g| \leq 1$ with $(\alpha, \beta, \emptyset) \vdash^i (\alpha_i, \beta_i, T''_i)$ is easily determined by a single lookup, the remaining problem is to compute the maximal $\sigma_i = |T'''_i \cap T_g|$ with $(\alpha_i, \beta_i, \emptyset) \vdash^* (0^R, \epsilon, T'''_i)$.

As shown by Goldberg and Nivre (2012, 2013), the optimal step can be determined in linear time for (uncorrected) arc-eager parsing, provided the gold tree is projective. The procedure is defined in terms of *costs* of transitions, rather than in terms of scores. We revisit this in Section 5.

For normalized arc-eager parsing (Table 8) and projective gold trees, the problem appears to be no easier than for shift-reduce parsing, but can still be solved in linear time, by a straightforward refinement of the algorithm by Nederhof (2019), blocking a token from becoming a left child of its parent if its label is R.

Now assume the gold tree may be non-projective. For shift-reduce parsing, Nederhof (2019) presents a cubic-time algorithm for calculating σ_i, generalizing the procedure of Goldberg et al. (2014), which is applicable only on projective trees. The algorithm has a modular design, in terms of a generic tabular dependency parsing algorithm (Eisner and Satta, 1999), plus an explicitly 'split' bilexical context-free grammar (Eisner and Satta, 1999; Eisner, 2000; Johnson, 2007) that encodes computations of shift-reduce parsing. A given configuration is translated to an input string, and weights between pairs of input positions are set according to existence of edges between corresponding tokens in the gold tree. Exhaustive parsing of the string by the grammar, using an appropriate semiring, yields σ_i.

Here we show that the same framework is applicable on arc-eager parsing. The generic tabular dependency parsing algorithm remains the same, but a new grammar is needed to encode computations of arc-eager parsing. Following Nederhof (2019), nonterminals are either single symbols or pairs of symbols, and rules are of one of the forms: $A \rightarrow (B, C)$, $(B, C) \rightarrow a$, $(B', \_) \rightarrow A\ (B, \_)$ or $(\_, C') \rightarrow (\_, C)\ A$, where a is a terminal. The last two forms are shorthand for any rules obtained by consistent substitution of the two underscores; which symbols can be meaningfully substituted is clear from context, as exemplified below.

We start with the normalized form (Table 8), which requires the grammar in Table 9, with the indicated translation from the configuration to an input string. The intuition behind this grammar is similar to the one by Nederhof (2019), but more cases need to be distinguished due to the labels. Grammar symbols R and R_t correspond to tokens

---

[6] The division of labor between stack and remaining input is also what distinguishes Table 2 from the *hybrid model* of Kuhlmann et al. (2011).

1) $(R,R) \rightarrow \boldsymbol{r}$

2) $(R_t,R_t) \rightarrow \boldsymbol{r_t}$

3) $(\_,R_t) \rightarrow (\_,R)\ R_t$

4) $(\_,R_t) \rightarrow (\_,R)\ L_t$

5) $(\_,R_t) \rightarrow (\_,R)\ N$

6) $(\_,R_t) \rightarrow (\_,R_t)\ N$

7) $R \rightarrow (R,R)$

8) $R_t \rightarrow (R,R_t)$

9) $R_t \rightarrow (R_t,R_t)$

10) $(L,L) \rightarrow \boldsymbol{\ell}$

11) $(L_t,L_t) \rightarrow \boldsymbol{\ell_t}$

12) $(\_,L_t) \rightarrow (\_,L)\ R_t$

13) $(\_,L_t) \rightarrow (\_,L)\ L_t$

14) $(\_,L_t) \rightarrow (\_,L)\ N$

15) $(\_,L_t) \rightarrow (\_,L_t)\ N$

16) $L \rightarrow (L,L)$

17) $L_t \rightarrow (L,L_t)$

18) $L_t \rightarrow (L_t,L_t)$

19) $(N,N) \rightarrow \boldsymbol{n}$

20) $(\_,N) \rightarrow (\_,N)\ N$

21) $(N,\_) \rightarrow N\ (N,\_)$

22) $(N,\_) \rightarrow L\ (N,\_)$

23) $(N,\_) \rightarrow L_t\ (N,\_)$

24) $(N,\_) \rightarrow L_b\ (N,\_)$

25) $N \rightarrow (N,N)$

26) $(L_b,L_b) \rightarrow \boldsymbol{\ell_b}$

27) $(\_,L_b) \rightarrow (\_,L_b)\ N$

28) $L_b \rightarrow (L_b,L_b)$

Table 9: Grammar for normalized arc-eager dependency parsing of a string in $\{\boldsymbol{r},\boldsymbol{\ell}\}^{k-1}\{\boldsymbol{r_t},\boldsymbol{\ell_t},\boldsymbol{\ell_b},\boldsymbol{n}\}\boldsymbol{n}^m$, representing a stack of length k and a remaining input of length m. A label R in the top of stack is translated to $\boldsymbol{r_t}$, and other occurrences of R are translated to $\boldsymbol{r}$. A label L in the top of stack is translated to $\boldsymbol{\ell_t}$, unless the candidate transition is **left_child**, in which case it is translated to $\boldsymbol{\ell_b}$; other occurrences of L are translated to $\boldsymbol{\ell}$. A label N in the top of stack is translated to $\boldsymbol{n}$.

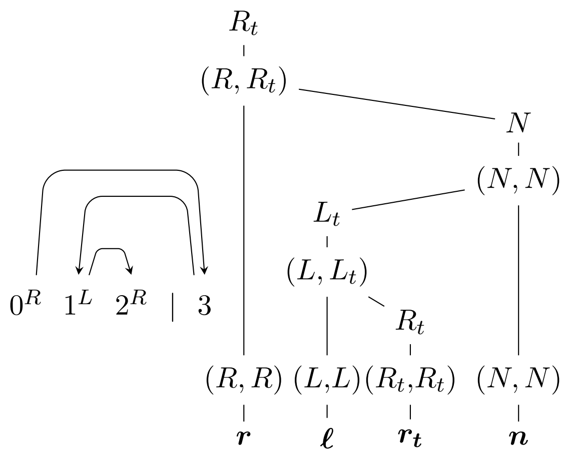


Figure 1: Dependency structure and configuration with stack of height 3 and remaining input of length 1, and corresponding derivation.

in the stack with label R, where R_t specifically means that the token is on top of the stack. Rules (3)–(4) distinguish the two cases $Y \in \{R, L\}$ of **reduce_left** with $X = R$. The rules are best read from right to left, as here for example "if the top of stack has label R or L, and if the token underneath has label R, then the latter keeps its label R and becomes the top of stack". Rules (5)–(6) allow for **reduce_left** with a right child that was in the remaining input, or that was the top of stack with label N. In (6), the underscore can be substituted by R or R_t. In (3)–(5), the only meaningful substitution is by R. Rules (10)–(18) are analogous to (1)–(9). Rules (20)–(21) allow any projective parse of the remaining input (as well as of the top of stack if that had label N), and (22)–(24) handle a token in the remaining input taking a left child in the stack, provided it has label L. In (20)–(24), the only meaningful substitution of the underscore is by N.

If a token has already been given label L, then it becoming a right child by (4) or (13) amounts to correcting a mistake made earlier, and may be necessary so the computation does not get stuck (cf. Table 5). If the next transition to be considered is **left_child** however, which puts L in the top of stack, then we do not wish the corresponding token to become a right child; **right_child** should be applied instead. Label L is then translated to $\boldsymbol{\ell_b}$ with $\boldsymbol{b}$ for 'blocking' (4) and (13).

Figure 1 exemplifies a derivation encoding a computation. A formal proof of correctness is by induction, showing that existence of a subderivation of the grammar implies existence of a corresponding subcomputation with the same score, and vice versa. Cf. the proof sketch by Nederhof (2019) for shift-reduce parsing.

Unnormalized arc-eager parsing (Table 6) requires a different approach, due to the different division of labor between stack and remaining input. We now need to count the number of right children of a token in the stack that were themselves in the stack, up to but not exceeding 1. E.g. rule (5) in Table 10 counts the first right child, but there is no further rule with right-hand side $(\_, R_1)\ R$ to allow a second right child from among the stack elements; other children from the remaining input are allowed, as e.g. by rule (10).

We now also need to observe the chosen policy. With the shift-before-reduce policy, if the candidate transition is **reduce**, then the first symbol of the remaining input becomes $\boldsymbol{n_p}$ ($\boldsymbol{p}$ for 'policy'). There is a notable absence of a rule with right-hand side $N_p\ (N, \_)$, which means that this token cannot become a left child without first taking a child from the stack as by (39) and (43), because if it were, the policy would be violated: the token should have been shifted, and reduced into its parent on the right, *preceding* the **reduce**. There is no restriction on the token becoming a right child, as e.g.

by (4).

A *strict* reduce-before-shift policy implies that a token in the stack should not be reduced into the token to its right if other tokens were previously shifted on top, unless it is to obtain more right children. This is because by the policy, the reduction should have happened earlier. Alternatively we may opt for a *non-strict* reduce-before-shift policy that allows us to correct mistakes made earlier. Either variant uses $\boldsymbol{r_p}$, $\boldsymbol{\ell_p}$, R_p and L_p to enforce the policy. E.g. there are no rules with R_p in the right-hand side, effectively blocking a derivation. Here rules (7)–(9) are needed to give an R-labeled stack element at least one right child, which by (14)–(15) allows the token to participate in a full derivation.

In order to compute the score for arc-eager parsing without our correction (starting in Table 4 with **reduce_correct**), one should omit the rules from Table 10 that correspond to L-labeled tokens becoming right children, i.e. (6), (9), (22), (25). Whether the **unshift** from Nivre and Fernández-González (2014) and Honnibal and Johnson (2015) can be handled in our framework requires further study.

5 Calculation for projective trees

If the gold tree is projective, then the problem becomes much easier. Here we assume the formulation of arc-eager parsing as in Table 6. The number σ_i, as defined in Section 4, for a configuration with stack $\alpha_i = a_1 \cdots a_k$ and remaining input $\beta_i = b_1 \cdots b_m$, can be calculated by counting in the first instance:

- the number of gold edges (a_{p-1}, a_p), where $1 < p \leq k$, plus
- the number of gold edges (a_p, b_q), plus
- the number of gold edges (b_p, a_q), such that a_q has label L, plus
- the number of gold edges (b_p, b_q),

but discounting a number of these, as follows. First, consider the case of the candidate transition being **shift**. If $m = 0$, the score becomes $-\infty$, as there is no available parent for the shifted token. If $m > 0$, we discount a possible gold edge (a_k, b_p) if the rightmost descendant of b_p is b_m, because no projective tree exists in which a_k is a left child while its descendants include the end of the input. We further discount a possible gold edge (a_{k-1}, a_k), because if a_k is to become a right child of a_{k-1}, then the correct step is **right_arc** in place of **shift**.

1) $(R, R) \rightarrow \boldsymbol{r}$
2) $(R_p, R_p) \rightarrow \boldsymbol{r_p}$
3) $(\_, R) \rightarrow (\_, R)\ N$
4) $(\_, R) \rightarrow (\_, R)\ N_p$
5) $(\_, R_1) \rightarrow (\_, R)\ R$
6) $(\_, R_1) \rightarrow (\_, R)\ L$
7) $(\_, R) \rightarrow (\_, R_p)\ N$
8) $(\_, R_1) \rightarrow (\_, R_p)\ R$
9) $(\_, R_1) \rightarrow (\_, R_p)\ L$
10) $(\_, R_1) \rightarrow (\_, R_1)\ N$
11) $(\_, R_1) \rightarrow (\_, R_1)\ N_p$
12) $R \rightarrow (R, R)$
13) $R \rightarrow (R, R_1)$
14) $R \rightarrow (R_p, R)$
15) $R \rightarrow (R_p, R_1)$
16) $R_p \rightarrow (R_p, R_p)$
17) $(L, L) \rightarrow \boldsymbol{\ell}$
18) $(L_p, L_p) \rightarrow \boldsymbol{\ell_p}$
19) $(\_, L) \rightarrow (\_, L)\ N$
20) $(\_, L) \rightarrow (\_, L)\ N_p$
21) $(\_, L_1) \rightarrow (\_, L)\ R$
22) $(\_, L_1) \rightarrow (\_, L)\ L$
23) $(\_, L) \rightarrow (\_, L_p)\ N$
24) $(\_, L_1) \rightarrow (\_, L_p)\ R$
25) $(\_, L_1) \rightarrow (\_, L_p)\ L$
26) $(\_, L_1) \rightarrow (\_, L_1)\ N$
27) $(\_, L_1) \rightarrow (\_, L_1)\ N_p$
28) $L \rightarrow (L, L)$
29) $L \rightarrow (L, L_1)$
30) $L \rightarrow (L_p, L)$
31) $L \rightarrow (L_p, L_1)$
32) $L_p \rightarrow (L_p, L_p)$

33) $(N, N) \rightarrow \boldsymbol{n}$
34) $(N_p, N_p) \rightarrow \boldsymbol{n_p}$
35) $(\_, N) \rightarrow (\_, N)\ N$
36) $(\_, N_p) \rightarrow (\_, N_p)\ N$
37) $(N, \_) \rightarrow N\ (N, \_)$
38) $(N, \_) \rightarrow L\ (N, \_)$
39) $(N, \_) \rightarrow L\ (N_p, \_)$
40) $(N, \_) \rightarrow L_p\ (N, \_)$
41) $(N, \_) \rightarrow L_b\ (N, \_)$
42) $N \rightarrow (N, N)$
43) $N \rightarrow (N, N_p)$
44) $N_p \rightarrow (N_p, N_p)$
45) $(L_b, L_b) \rightarrow \boldsymbol{\ell_b}$
46) $(\_, L_b) \rightarrow (\_, L_b)\ N$
47) $L_b \rightarrow (L_b, L_b)$

Table 10: Grammar for unnormalized arc-eager dependency parsing. With reduce-before-shift, the string is in $\boldsymbol{r}\{\boldsymbol{r}, \boldsymbol{r_p}, \boldsymbol{\ell}, \boldsymbol{\ell_p}\}^{k-2}\{\boldsymbol{r}, \boldsymbol{r_p}, \boldsymbol{\ell}, \boldsymbol{\ell_p}, \boldsymbol{\ell_b}\}\boldsymbol{n}^m$, for stack length k and remaining input length m. Now $\boldsymbol{\ell_b}$ is used if the candidate transition is **shift**, and a non-bottommost symbol to the left of that becomes $\boldsymbol{r_p}$ or $\boldsymbol{\ell_p}$. For a *strict* reduce-before-shift policy moreover, the second to the $k-2$-th symbols become $\boldsymbol{r_p}$ or $\boldsymbol{\ell_p}$, and furthermore the $k-1$-th becomes $\boldsymbol{r_p}$ or $\boldsymbol{\ell_p}$ if the candidate transition is **left_arc** or **reduce**, and furthermore the k-th becomes $\boldsymbol{r_p}$ or $\boldsymbol{\ell_p}$ if the candidate transition is **left_arc**; otherwise, these symbols are $\boldsymbol{r}$ or $\boldsymbol{\ell}$. With shift-before-reduce, the string does not contain $\boldsymbol{r_p}$ or $\boldsymbol{\ell_p}$, and the first $\boldsymbol{n}$ is replaced by $\boldsymbol{n_p}$ if the candidate transition is **reduce**.

Second, if the candidate transition is **reduce**, we discount up to one gold edge in case of the shift-before-reduce policy, as follows, and as illustrated by Figure 2. Let r be largest such that, for some $p > 1$, there is a gold edge (b_p, a_r) where a_r has label L, or there is a gold edge (a_r, b_p); if no such gold edge exists, let $r = 1$. If there is no s ($r < s \leq k$) such that a_s has label L and (a_{s-1}, a_s) is not a gold edge, then we discount any gold edge

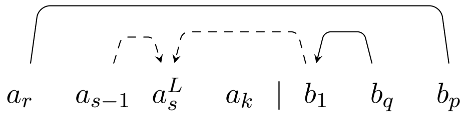


Figure 2: Discounting of (b_q, b_1) if s does not exist.

(b_q, b_1). The rationale is that if b_1 can be given a child from among the tokens in the stack (by a gold edge or otherwise, and without discounting another gold edge elsewhere), then this justifies postponing the **shift** until after the **reduce**. If it cannot be, then b_1 becoming a left child violates the shift-before-reduce policy.

Lastly, if the candidate transition is **shift**, we discount further gold edges in case of the non-strict reduce-before-right policy, which requires a_{k-1} to either become a child of some b_p or take some b_p as child, to justify it not having been reduced into a_{k-2} before the shift.[7] From among the cases to be distinguished, we choose the one that discounts the fewest edges. First, if the label of a_{k-1} is L, we can let it become a left child, but should then discount a possible gold edge (a_{k-2}, a_{k-1}). Second, if there is a gold edge (a_{k-1}, b_p), then no edges need be discounted. Otherwise, we need to find a child b_p of a_{k-1}, for which there are five options, illustrated in Figure 3: (A) The first option is applicable if a_k has descendants among the remaining input or has a parent b_q $(q > 1)$ among the remaining input. In the former case, choose b_p to be the rightmost among the descendants (but let $p = m - 1$ if the rightmost descendant is b_m), and in the latter case choose $p = 1$. In effect we assume non-gold edges (a_{k-1}, b_p) and (b_p, a_k), and consequently we discount any gold edge (b_q, b_p) and any gold edge to a_k. (B1) If a_k has a parent b_q in the remaining input, choose b_p to be b_q. In effect we assume non-gold edge (a_{k-1}, b_q), and consequently we discount any gold edge (a_r, b_q) with $r \leq k-2$ or any gold edge (b_r, b_q), as well as any gold edges (b_q, a_s) with $s \leq k-2$. (B2) If a_k does not have a parent in the remaining input, let b_q be the token immediately to the right of the rightmost descendant of a_k among the remaining input (but let $q = m$ if the rightmost descendant is b_m), and let $q = 1$ if a_k has no descendants among the remaining input. As in (B1), we in effect assume non-gold edge (a_{k-1}, b_q), and discount any gold edge (a_r, b_q) with $r \leq k-2$ or any gold edge (b_u, b_q), as well as any gold edges (b_q, a_s) with $s \leq k-2$.

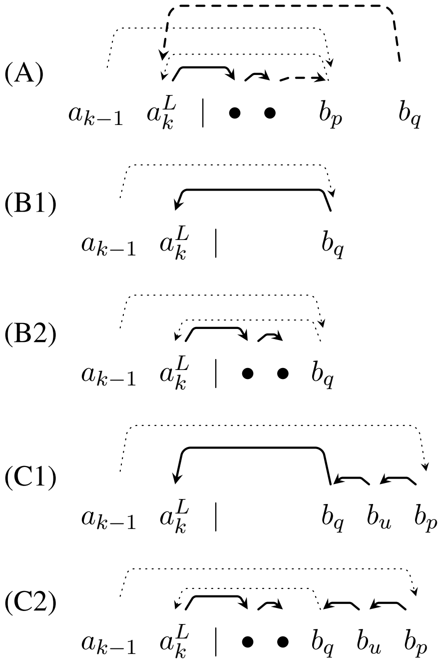


Figure 3: Non-strict reduce-before-shift policy. Bold edges are gold. Dashed edges are discounted. Dotted edges are non-gold.

(C1) and (C2) are similar to (B1) and (B2), but b_p is chosen to be the first ancestor of b_q that does not have a parent in the remaining input (but it may have in the stack). Much as before, we discount any gold edge (a_r, b_p) with $r \leq k-2$, as well as any gold edges (b_u, a_s) with $s \leq k-2$, where b_u is b_q or b_p or any other token on the path of gold edges from b_q to b_p. One can show that choices of b_p other than in (A), (B1), (B2), (C1), (C2) would entail discounting of at least as many edges.

Aufrant et al. (2018) propose approximating the calculation of the optimal step for a non-projective gold tree, by a procedure defined in terms of *costs* of transitions, analogous to the procedure by Goldberg and Nivre (2012, 2013), but without taking full account of edges that violate projectivity. Similarly, if the above procedure to calculate *scores* is applied on a non-projective tree, then an approximation is obtained. The advantage is the simplicity and the linear time complexity.

6 Empirical results

The advantage of ‘dynamic oracles’ for improving parsing accuracy has been demonstrated before. Our experiments have therefore concentrated on

---

[7] The *strict* reduce-before-right policy is more difficult to realize, and discussion is omitted for space reasons.

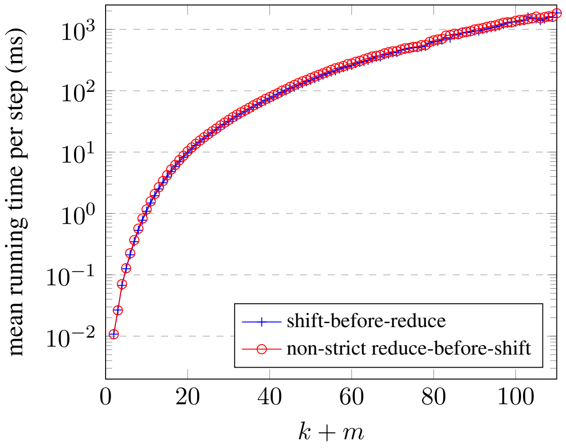


Figure 4: Mean running time per (exact) calculation of the optimal step (milliseconds) against the total length $k+m$ of configurations.

two obvious questions, viz. whether the cubic-time calculation is feasible in practice, and whether the higher time costs are rewarded with a more accurate output, relative to a linear-time approximation of the kind discussed in Section 5.

Considered here is unnormalized arc-eager parsing. The classifier, implemented in Java and DL4J, uses simple features (gold POS of the three rightmost elements of the stack and three leftmost elements of the remaining input, and leftmost and rightmost dependency relations in the topmost two stack elements).

The parser was first trained on configurations corresponding to projectivized gold trees from the German (GSD) corpus of Universal Dependencies v2.2. The trained parser was then applied on the unprojectivized trees, and the optimal step was calculated for each configuration thus visited.

6.1 Running time

Figure 4 presents running time, on a laptop with an Intel i7-7500U processor (4 cores, 2.70 GHz) with 8 GB of RAM. The larger context-free grammar of Table 10, relative to the one for shift-reduce parsing, leads to a higher constant factor in the time complexity. Nonetheless, the calculation is feasible even for long sentences.

6.2 Accuracy of the approximation

In 8.0% and 8.1% of the visited configurations, one or more of the values $\rho_1, \ldots, \rho_4$ for the four transitions differed between the exact calculation (Section 4) and the approximation (Section 5), for the shift-before-reduce and non-strict reduce-before-shift policies respectively. However, we are less interested in the absolute values of the scores than in which of them is highest. Note further that more than one may be equal to their maximum. By comparing the sets of transitions with the maximum calculated score, we found that the true set and the approximate set differed for only 0.4% and 0.5% of the total number of configurations, for the two policies respectively. The most frequent errors are listed in Table 11. Somewhat surprisingly, in the great majority of cases, the approximate set was contained in the true set; these cases sum to 89.0% and 87.8% of the total number of errors, respectively. The implication is that if a parser trained with a 'dynamic oracle' does arbitrary tie breaking between multiple optimal transitions, then there are few immediate prospects to improve parsing accuracy by incorporating the exact calculation. The situation may change if future research reveals better alternatives to arbitrary tie breaking.

exact SH-b.-RE	approximation	proportion
{SH, RA}	{SH}	29.4%
{SH, LA}	{SH}	20.5%
{SH, RA, RE}	{SH}	10.8%
{SH, RA, RE}	{RE}	7.5%
{SH, RA, LA, RE}	{SH, LA}	6.0%
{SH, RA, LA, RE}	{LA, RE}	4.3%
{LA, RE}	{SH,RA,LA,RE}	3.2%

exact n.-s. RE-b.-SH	approximation	proportion
{SH, RA}	{SH}	31.4%
{SH, LA, RE}	{SH}	17.9%
{RA, RE}	{RE}	16.2%
{RA, LA, RE}	{LA, RE}	6.6%
{SH, RA, LA, RE}	{SH}	4.0%
{LA, RE}	{SH,RA,LA,RE}	3.0%
{RE}	{SH,RA,RE}	2.9%

Table 11: Proportions of the seven most frequent errors made by the approximation of the optimal transition(s).

7 Conclusions

Our exact calculation of the optimal step solves an open problem in parsing theory. Further research into the application of 'dynamic oracles' is needed to determine whether this can be exploited to improve parsing accuracy.

Acknowledgements

Many thanks go to the reviewers, whose reports were very detailed and helpful.

References

L. Aufrant, G. Wisniewski, and F. Yvon. 2018. Exploiting dynamic oracles to train projective dependency parsers on non-projective trees. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 413–419, New Orleans, Louisiana.

M. Damonte, S.B. Cohen, and G. Satta. 2017. An incremental parser for Abstract Meaning Representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 1, pages 536–546, Valencia, Spain.

J. Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In H. Bunt and A. Nijholt, editors, *Advances in Probabilistic and other Parsing Technologies*, chapter 3, pages 29–61. Kluwer Academic Publishers.

J. Eisner and G. Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *37th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 457–464, Maryland, USA.

D. Fernández-González and C. Gómez-Rodríguez. 2018. A dynamic oracle for linear-time 2-planar dependency parsing. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 386–392, New Orleans, Louisiana.

N. Fraser. 1989. Parsing and dependency grammar. *UCL Working Papers in Linguistics*, 1:296–319.

Y. Goldberg and J. Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *The 24th International Conference on Computational Linguistics*, pages 959–976, Mumbai, India.

Y. Goldberg and J. Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics*, 1:403–414.

Y. Goldberg, F. Sartorio, and G. Satta. 2014. A tabular method for dynamic oracles in transition-based parsing. *Transactions of the Association for Computational Linguistics*, 2:119–130.

C. Gómez-Rodríguez and D. Fernández-González. 2015. An efficient dynamic oracle for unrestricted non-projective parsing. In *53rd Annual Meeting of the Association for Computational Linguistics and 7th International Joint Conference on Natural Language Processing*, volume 2, pages 256–261, Beijing.

C. Gómez-Rodríguez, F. Sartorio, and G. Satta. 2014. A polynomial-time dynamic oracle for non-projective dependency parsing. In *Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 917–927, Doha, Qatar.

M. Honnibal, Y. Goldberg, and M. Johnson. 2013. A non-monotonic arc-eager transition system for dependency parsing. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 163–172, Sofia, Bulgaria.

M. Honnibal and M. Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 1373–1378, Lisbon, Portugal.

M. Johnson. 2007. Transforming projective bilexical dependency grammars into efficiently-parsable CFGs with Unfold-Fold. In *45th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 168–175, Prague, Czech Republic.

M. Kuhlmann, C. Gómez-Rodríguez, and G. Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *49th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 673–682, Portland, Oregon.

M. de Lhoneux, S. Stymne, and J. Nivre. 2017. Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *15th International Conference on Parsing Technologies*, pages 99–104, Pisa, Italy.

A. Nasr. 1995. A formalism and a parser for lexicalised dependency grammars. In *Fourth International Workshop on Parsing Technologies*, pages 186–195, Prague and Karlovy Vary, Czech Republic.

M.-J. Nederhof. 2019. Calculating the optimal step in shift-reduce dependency parsing: From cubic to linear time. *Transactions of the Association for Computational Linguistics*, 7:283–296.

J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *8th International Workshop on Parsing Technologies*, pages 149–160, LORIA, Nancy, France.

J. Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, Held in cooperation with ACL-2004, pages 50–57, Barcelona, Spain.

J. Nivre. 2006. *Inductive Dependency Parsing*. Springer-Verlag.

J. Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

J. Nivre and D. Fernández-González. 2014. Arc-eager parsing with the tree constraint. *Computational Linguistics*, 40(2):259–267.

J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 49–56, Boston, Massachusetts.

P. Qi and C.D. Manning. 2017. Arc-swift: A novel transition system for dependency parsing. In *55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, volume 2, pages 110–117, Vancouver, Canada.

S. Sippu and E. Soisalon-Soininen. 1990. *Parsing Theory, Vol. II: LR(k) and LL(k) Parsing*, volume 20 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag.

M. Straka, J. Hajič, J. Straková, and J. Hajič, jr. 2015. Parsing universal dependency treebanks using neural networks and search-based oracle. In *Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories*, pages 208–220, Warsaw, Poland.

Subword Pooling Makes a Difference

Judit Ács[1,3]
Budapest University of
Technology and Economics[1]
judit@aut.bme.hu

Ákos Kádár[2]
Borealis AI[2]
akos.kadar@
borealisai.com

András Kornai[1,3]
Institute for Computer Science
and Control[3]
andras@kornai.com

Abstract

Contextual word-representations became a standard in modern natural language processing systems. These models use subword tokenization to handle large vocabularies and unknown words. Word-level usage of such systems requires a way of pooling multiple subwords that correspond to a single word. In this paper we investigate how the choice of subword pooling affects the downstream performance on three tasks: morphological probing, POS tagging and NER, in 9 typologically diverse languages. We compare these in two massively multilingual models, mBERT and XLM-RoBERTa. For morphological tasks, the widely used 'choose the first subword' is the worst strategy and the best results are obtained by using attention over the subwords. For POS tagging both of these strategies perform poorly and the best choice is to use a small LSTM over the subwords. The same strategy works best for NER and we show that mBERT is better than XLM-RoBERTa in all 9 languages. We publicly release all code, data and the full result tables at `https://github.com/juditacs/subword-choice`.

1 Introduction

Training of contextual language models on large training corpora generally begins with segmenting the input into subwords (Schuster and Nakajima, 2012) to reduce the vocabulary size. Since most tasks consume full words, practitioners have the freedom to decide whether to use the first, the last, or some combination of all subwords. The original paper introducing BERT, Devlin et al. (2019), suggests using the first subword for named entity recognition (NER), and did not explore different poolings. Kondratyuk and Straka (2019) also use the first subword, for dependency parsing, and remark in a footnote that they tried the first, last, average, and max pooling but the choice made no difference. Kitaev et al. (2019) report similar findings for constituency parsing, but nevertheless opt for reporting results only using the last subword. Hewitt and Manning (2019) take the average of the subword vectors for syntactic and word sense disambiguation tasks. Wu et al. (2020) use attentive pooling with a trainable norm for news topic classification and sentiment analysis in English. Shen et al. (2018) use hierarchical pooling for sequence classification tasks in English and Chinese.

Here we show that for word-level tasks (morphological, POS and NER tagging), particularly for languages where the proportion of multi-subword tokens (i.e. those word tokens that are split into more than one subword) is high, more care needs to be taken as both pooling strategy, and that the choice of language matters. We demonstrate this clearly for European languages with rich morphology, and in Chinese, Japanese and Korean (CJK). Similar to subword pooling, the choice of the lowest layer, the topmost one, or some combination of the activations in different layers has to be made. Here our main focus is subword pooling, but we do discuss layer pooling to the extent it sheds light on our main topic. We observe that the gap between using the first and the last subword unit is larger in lower layers than in higher ones.

We describe our data and tasks in Section 2, and the subword pooling strategies investigated in Section 3. Our results are presented in Section 4, and in Section 5 we offer our conclusions.

Our main contributions are:

- we show that subword pooling matters, the differences between choices are often significant and not always predictable;
- XLM-RoBERTa (Conneau et al., 2019) is slightly better than mBERT in the majority of morphological and POS tagging tasks; while mBERT is better at NER in all languages;

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2284–2295
April 19 - 23, 2021.

- the common choice of using the first subword is generally worse than using the last one for morphology and POS but the best for NER;
- the difference between using the first and the last subword is larger in lower layers than in higher layers and it is more pronounced in languages with rich morphology than in English;
- the choice of subword pooling makes a large difference for morphological and POS tagging but it is less important for NER;
- we release the code, the data and the full result tables.

2 Tasks, languages, and architectures

We investigate pooling through three kinds of tasks. In *morphological* tasks we attempt to predict morphological features such as gender, tense, or case. In *POS* tasks we predict the lexical category associated with each word. In *NER* tasks we assign BIO tags (Ramshaw and Marcus, 1995) to named entities. We chose word-level, as opposed to syntactic, tasks because they can be tackled with fairly simple architectures and thus allow for a large number of experiments that highlight the differences between subword pooling strategies. Our experiments are limited only by the availability of standardized multilingual data.

We use Universal Dependencies (UD) (Nivre et al., 2018) for morphological and POS tasks, and WikiAnn (Pan et al., 2017) for NER. We pick the largest treebank in each language from UD and sample 2000 train, 200 dev and 200 test sentences for the morphological probes and up to 10,000 train, 2000 dev and 2000 test sentences – often limited by the size of the treebank – for POS. We chose languages with reasonably large treebanks in order to generate enough training data, making sure we have an example from each language family, as well as one from European subfamilies since their treebanks tend to be very large. We use 10,000 train, 2000 dev and 2000 test sentences for NER. Preprocessing steps are further explained in Appendix A. Our choice of languages are Arabic, Chinese, Czech, English, Finnish, French, German, Japanese, and Korean. UD's gold tokenization is kept and we run subword tokenization on individual tokens rather then the full sentences.

Morphological tasks UD assigns zero or more tag-value pairs to each token such as `VerbForm=Ger` for 'asking'. We define a probe as a triplet of $\langle$language, tag, POS$\rangle$, i.e. we train a classifier to predict the value of a single tag in a sentence in a particular language.[1] The task $\langle$English, VerbForm, VERB$\rangle$ would be trained to predict one of three labels for each English verb: finite, infinite or gerund. We pick 4 tasks that are applicable to at least 3 of the 6 languages where the task makes sense (there are no morphological tags for Chinese and Japanese, and Korean uses a different tagging scheme). Table 1 lists the probing tasks.

Part-of-speech tagging assigns a syntactic category to each token in the sentence. Usually treated as a crucial low level task to provide useful features for higher level linguistic analysis such as syntactic and semantic parsing. Universal POS tags (UPOS) are available in UD in all 9 languages.

Named entity recognition is a classic information extraction subtask that seeks to identify the span of named entities mentioned in the sentence and classify them into pre-defined categories such as person names, organizations, locations etc. NER was the only token level task explored in the original BERT paper Devlin et al. (2019).

Architectures BERT and other contextual models use subword tokenizers that generate one or more subwords for each token. In this study we compared mBERT and XLM-RoBERTa, two Transformer-based large scale language models with support for over 100 languages. We pick these two since they are architecturally similar (both have 12 layers and the same hidden size) making our comparison easier. mBERT was trained on Wikipedia while XLM-RoBERTa was trained on CommonCrawl (Wenzek et al., 2020). Both models have been extensively applied to English and multilingual tasks, but generally at the sentence or sentence pair level, where subword issues do not come to the fore. mBERT uses a common wordpiece vocabulary with 118k subword units. When a word is split into multiple subword units, each token that is not the first one is prefixed with `##`. XLM-RoBERTa's vocabulary was trained in a similar fashion but with 250k units and a special start symbol (Unicode lower eights block) instead of

---

[1] Consolidating these triplets across POS would be misleading in that the results show large variation across different POS values.

Language	**Tag**	**POS**	**# class**
Arabic	case	NOUN	3
Arabic	gender	ADJ	2
Czech	gender	ADJ	3
Czech	gender	NOUN	3
English	verbform	VERB	4
Finnish	case	NOUN	12
Finnish	verbform	VERB	3
French	gender	ADJ	2
French	gender	NOUN	2
French	verbform	VERB	3
German	case	NOUN	4
German	gender	ADJ	3
German	gender	NOUN	3
German	verbform	VERB	3

Table 1: List of morphological probing tasks. The last column is the number of classes in a particular task.

continuation symbols. Each word is prefixed with this start symbol before it is tokenized into one or more subword units. These start symbols are often then tokenized as single units, particularly before Chinese, Japanese and Korean characters, therefore artificially increasing the subword unit count. We indicate the proportion of words starting with a standalone start symbol along with other tokenization statistics in Table 2.

As Table 2 shows, the number of subword tokens is highly dependent on the language. English words are only split in 14.3% (resp. 16.9%) of the time by the two models, while in many other languages more than half of the words are tokenized into two or more subword units. We hypothesize that this is due to the combination of the characteristics of the English language and its overrepresentation in the training data and the subword vocabulary.

We also observe that the two models' tokenizers work in very different ways. Out of the 2800 morphological test examples, only 58 are tokenized the same way and 51 of these are not split into multiple subwords. Only 7 words that are in fact tokenized, are tokenized the same way. Although the full tokenization is rarely the same, the first and the last subwords are the same in 45.5% and in 44.7% of the cases.

	mBERT		**XLM-RoBERTa**		
	count	2+	count	2+	_start
Arabic	1.95	48.9	1.49	35.0	3.4
Chinese	1.58	53.5	2.13	88.5	86.6
Czech	2.04	53.0	1.7	45.2	1.6
English	1.25	14.3	1.25	16.9	0.8
Finnish	2.32	67.3	1.86	53.0	2.3
French	1.34	22.4	1.41	28.7	2.1
German	1.64	30.6	1.57	29.7	1.3
Japanese	1.6	43.0	2.25	94.6	92.9
Korean	2.44	75.7	2.16	67.3	9.0

Table 2: Subword tokenization statistics by language and model. First and third columns: average number of pieces that one word is split into. Second and fourth columns: proportion of multi-subword words. Last column: proportion of words that start with a standalone start token in XLM-RoBERTa.

3 Subword pooling

We test 9 types of pooling methods listed in Table 3 and grouped in three broad types. The first group uses the first and last subword representations in some combination. In F+L pooling the mixing weight is the only learned parameter. The second group are parameter-free elementwise pooling operations.

Method	**Explanation**	**Params**
FIRST	first subword unit	none
LAST	last subword unit	none
LAST2	concatenation of the last two subword units	none
F+L	$wu_{\text{first}} + (1-w)u_{\text{last}}$	w
SUM	elementwise sum	none
MAX	elementwise max	none
AVG	elementwise average	none
ATTN	Attention over the subwords, weights generated by an MLP	MLP
LSTM	biLSTM reads all vectors, final hidden state	LSTM

Table 3: Subword unit pooling methods. u_{first} and u_{last} refer to the first and the last units respectively.

The last two methods rely on small neural networks that learn to combine the subword representations. Our subword ATTN has one hidden layer of 50 neurons with ReLU activation and a final softmax layer that generates a probability distribution over the subword units of the token. Similarly to self-attention, these probabilities are used to compute the weighted sum of subword representations to produce the final token vector. The LSTM uses a biLSTM (Hochreiter and Schmidhuber, 1997) that summarizes the 768-dimensional vectors (the hidden size of both models) into a 50-dimensional hidden vector in each direction, which are then concatenated and passed onto the classifier. These two are considerably more complicated and slower to train than the other methods, but ATTN works well for morphological tasks, and LSTM for POS tagging in CJK languages. Shen et al. (2018) found hierarchical pooling beneficial, but they investigated sentence level tasks where the subword stream is much longer than in the word-level tasks we are considering (words are rarely split into more than 4 subwords) and hierarchical pooling has better traction.

Layer pooling effects Both mBERT and XLM-RoBERTa have an embedding layer followed by 12 hidden layers. The only contextual information available in the embedding layer is the position of the token in the sentence. Hidden activations are computed with the self-attention layers, therefore in theory have access to the full sentence. We ran our experiments for each layer separately as well as for the sum of all layers. For all tasks, as we move up the layers, results also move up or down in tandem. As exhaustive experiments considering different combinations of layers were computationally too expensive for our setup, and would significantly complicate presentation of our results, we pick a single setting for all experiments by computing the best *expected layer* for each task as

$$\mathbb{E}(L) = \frac{\sum_{l_i \in L} i A(l_i)}{\sum_{l_i \in L} A(l_i)}, \qquad (1)$$

where L is the set of all layers, l_i is the ith layer, and $A(l_i)$ is the development accuracy at layer i. As Figure 1 shows, the expected layers are almost always centered around the 6th layer. Therefore, with the exception of comparing FIRST and LAST, which we analyze in greater detail in 4.1, we chose the 6th layer to simplify the presentation.

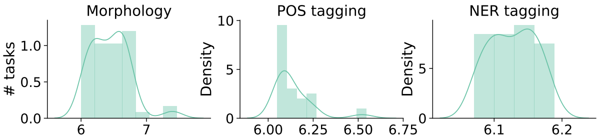


Figure 1: Distribution of the weighted average of layers across all tasks.

Probing setup Every experiment is trained separately, with no parameter sharing between the tasks and the experiments. We probe the morphology on fixed representations with a small MLP (multilayer perceptron) with a single hidden layer of 50 neurons and ReLU activation. We train the same model for POS tagging and NER on top of each token representation. We keep the number of parameters intentionally low, about 40k, to avoid overfitting on the probing data and to force the MLP to probe the representation instead of memorizing the data. We do note, however, that ATTN and LSTM increase the number of trained parameters to 77k and 330k respectively. We run each configuration 3 times with different random seeds. The standard deviation of results is always less than 0.06 for morphology and less than 0.005 for POS and NER. Further details are available in Appendix B.

Choosing the size of the LSTM LSTM is our subword pooling method with the most parameters. The number of parameters scales quadratically with the hidden dimension of the LSTM. We pick this dimension with binary parameter search on morphology tasks. Our early experiments showed that increasing the size over 1000 showed no significant improvement, and a binary search between 2 and 1024 led us to choose a biLSTM with 100 hidden units.

4 Results

Our analysis consisted of two steps. We first performed the FIRST and LAST tasks at each layer (see Figure 2). Based on the results of this, we picked a single layer, the 6th, to test all 9 subword pooling choices. The full list of results on the 6th layer is listed in Appendix C.

4.1 Layer pooling

We find that although LAST is almost always better than FIRST, the gap is smaller in higher layers. We quantify this with the ratio of the accuracy of LAST and FIRST at the same layer. Figure 2 illustrates this ratio for a few selected morphological tasks and POS and NER for all 9 languages. We split the morphological tasks into two groups, Finnish tasks and other tasks. ⟨Finnish, Case, NOUN⟩ shows the largest gap in the lower layers, LAST is 8 times better than FIRST. We observe smaller gaps in other tasks. POS shows a fairly uniform picture with the exception of Korean, where FIRST is worse in all layers and both models. Lower layers in mBERT show a larger gap in Czech and the same is true for Chinese and Japanese in XLM-RoBERTa. NER shows little difference between FIRST and LAST except for the first few layers, particularly in Chinese and Korean. To interpret these results, keep in mind that CJK tokenization is handled somewhat arbitrarily by XLM-RoBERTa, particularly in the first subword (c.f. Table 2).

4.2 Morphology

We present the results of 14 morphological probing tasks (see Table 1) and 9 subword pooling strategies (see Table 3) using the 6th layer of each model.

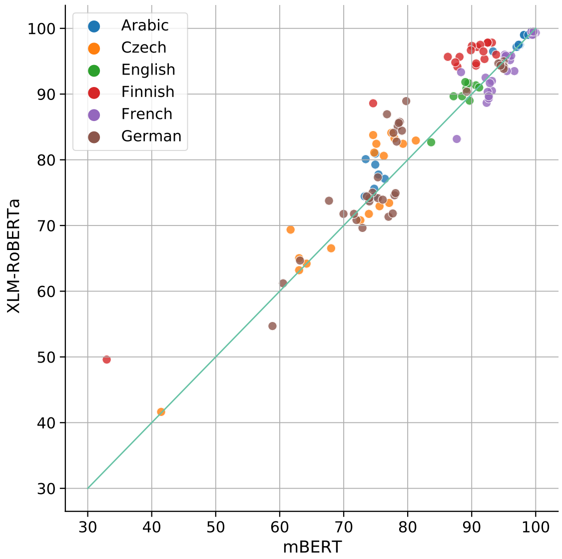


Figure 3: Accuracy of mBERT vs. XLM-RoBERTa on morphological tasks.

mBERT vs. XLM-RoBERTa Averaging over all tasks, XLM-RoBERTa achieves 85.7% macro accuracy while mBERT achieves 83.9%. On a per-language basis, XLM-RoBERTa is slightly better than mBERT except for French. Figure 3 shows our findings. The two models generally perform similarly with the exception of French and Finnish: mBERT is almost always better at French tasks, while XLM-RoBERTa is always better at Finnish tasks. Similar trends emerge when looking at the results by subword pooling method. XLM-RoBERTa is always better regardless of the pooling choice but the difference is only significant ($p < 0.05$) for MAX and SUM.[2] These findings suggest that XLM-RoBERTa retains more about the orthographic presentation of a token, and it uses tokenization that is closer to morpheme segmentation, hence performing better at inflectional morphology, which is most often derivable from the word form alone.

First or last subword? As Figure 4 shows, with the exception of the ⟨Arabic, Case, N⟩ task, LAST is always better than FIRST. We find the largest difference in favor of LAST in Finnish and Czech. Table 4 lists all tasks where the difference between FIRST and LAST is larger than 20% along with the only counterexample (where the difference is about 10% in the other direction). These findings are likely due to the fact that Finnish and Czech exhibit the richest inflectional morphology in our sample.

The exceptional behavior of Arabic case may relate to the fact that case often disappears in modern Arabic (Biadsy et al., 2009). When this occurs the first token, being closest to the previous word, may provide a more reliable indicator, especially if that word was a preposition. Given the complex distribution of Arabic case endings, our sample is too small to ascertain this, and the results, about 75% on a 3-way classification task, are clearly too far from the optimum to draw any major conclusion (note that on Finnish case, a 12-way classification task, we get above 94%[3]).

Other pooling choices While FIRST is clearly inferior in morphology, the picture is less clear for the other 8 pooling strategies. As Figure 5 illustrates, ATTN is better than all other choices for both models but its advantage is only significant over a few other choices. We observe larger – and more often significant – differences in the case of mBERT than in XLM-RoBERTa.

---

[2] We use paired t-tests on the accuracy of the models on the 14 tasks.

[3] Finnish has more than 12 cases but infrequent ones were excluded.

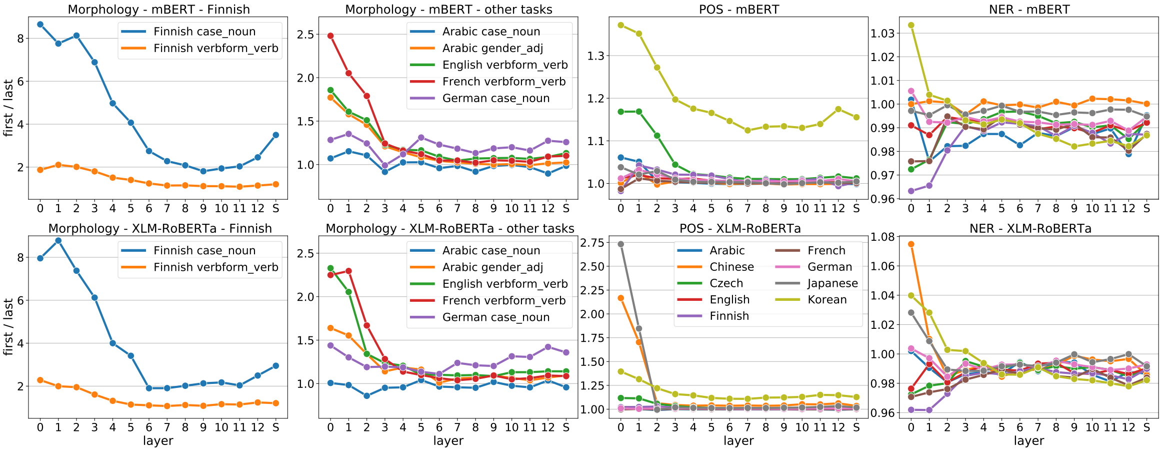


Figure 2: LAST-FIRST ratio of the test accuracy of some morphological tasks and of POS and NER in all languages across all layers. We plot Finnish morphological tasks separately since the effect is so pronounced that presenting them on the same plot would render the scaling uninformative for the other cases. S is the sum of all layers. Note that we do not have a strongly prefixing language due to the lack of available probing data.

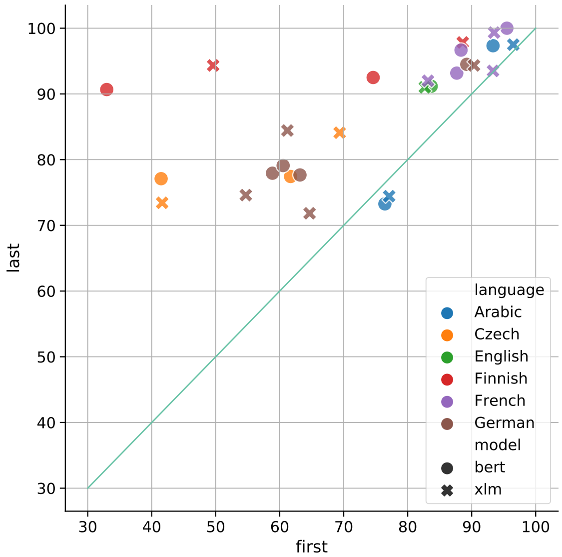


Figure 4: FIRST vs. LAST on morphological tasks.

task	model	first	last
⟨Finnish, Case, N⟩	mBERT	33.0	90.7
⟨Finnish, Case, N⟩	XLM-RoBERTa	49.6	94.4
⟨Czech, Gender, A⟩	mBERT	41.5	77.1
⟨Czech, Gender, A⟩	XLM-RoBERTa	41.6	73.5
⟨German, Gender, N⟩	XLM-RoBERTa	61.2	84.4
⟨Arabic, Case, N⟩	XLM-RoBERTa	77.1	74.5
⟨Arabic, Case, N⟩	mBERT	76.5	73.3

Table 4: Morphological tasks with the largest difference between FIRST and LAST. The two tasks where FIRST is better than LAST are also listed.

	XLM-RoBERTa	mBERT
first	7.1%	6.0%
last	81.5%	83.7%
middle	5.9%	6.3%
single	5.5%	4.0%

Table 5: Distribution of the location of the highest weighted subword. Single refers to tokens that are not split by the tokenizer.

Attention weights The MLP used in ATTN assigns a weight to each subword which are then normalized by softmax. We examine these weights for each token in the test data for morphology. Table 5 lists the proportion of tokens where ATTN assigns the highest weight to the first, last or a middle token, or the token is not split by the tokenizer. The last subword is weighted highest in more than 80% of the cases. The only task where the last subword is not the most frequent winner is ⟨Arabic, Case, N⟩, where the first is weighted highest in 60% of the tokens by both models. These findings are in line with the behavior of FIRST and LAST.

4.3 POS tagging

We train POS tagging models for 9 languages with 9 subword pooling strategies. We evaluate the models using tag accuracy.

mBERT vs. XLM-RoBERTa As with morphological probing tasks, XLM-RoBERTa is slightly better than mBERT (95.4 vs. 94.6 macro average). We also observe that the choice of subword makes less difference than it does in morphological probing. Figure 6 shows that experiments in one language tend to cluster together regardless of the

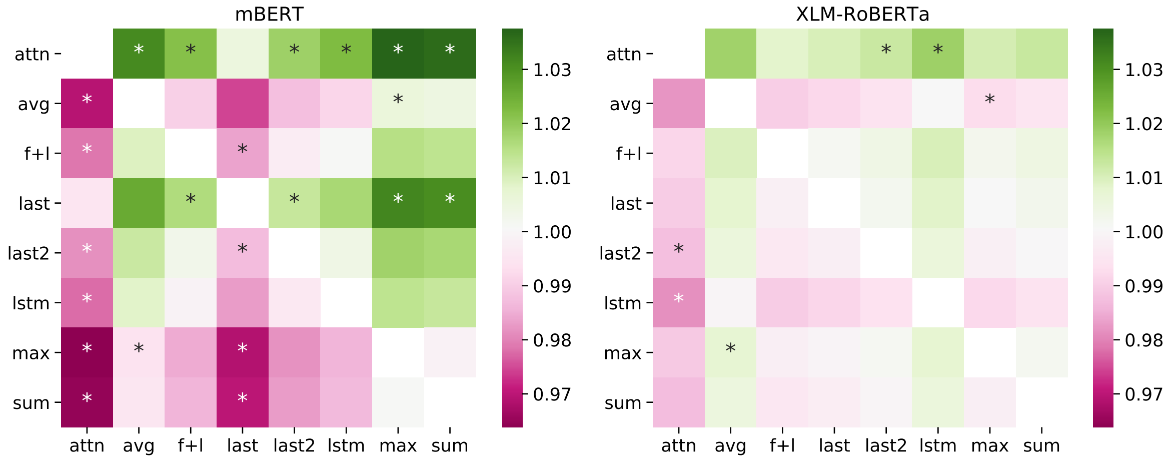


Figure 5: Pairwise ratio of test accuracy by subword choice on morphological tasks. Colors indicate row/column ratios. Green cells mean that the row subword choice yields better results than the column choice. $*$ marks pairs where the difference is statistically significant. ATTN is better than all other choices, therefore its row is green. FIRST is omitted for clarity as it is much worse than the other choices.

subword pooling choice except for a few outliers: FIRST for Chinese and Korean is much worse in both models. The same result can be observed in Japanese, to a lesser extent though. Language-wise we find that XLM-RoBERTa is much better at Finnish and somewhat worse in Chinese but the two models generally perform similarly.

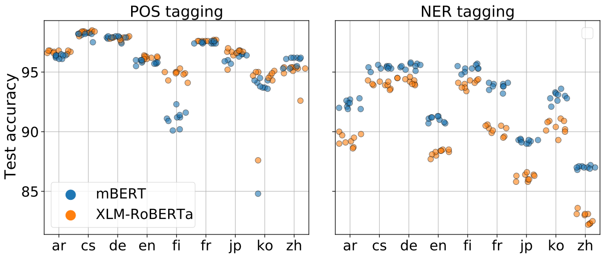


Figure 6: mBERT vs. XLM-RoBERTa for POS tagging and NER.

Choice of subword. As with morphology FIRST the is the worst choice, but the effect is not as marked for POS tasks. In Figure 6 we observe 3 outliers, XLM-RoBERTa, FIRST for Chinese and FIRST for Korean for both models. The only consistent trend is that XLM-RoBERTa is clearly better for Finnish regardless of the choice of subword pooling. The picture is less clear for other languages.

We split the analysis into CJK and non-CJK languages. Figure 7 and Figure 8 show a comparison for non-CJK languages and CJK languages respectively. The difference between choices is generally much smaller than for morphology. FIRST is the worst choice both for CJK and non-CJK languages. Interestingly one of the best choices for morphology, LAST, is the second worst choice for POS tagging, while one of the worst for morphology, LSTM, is one of the best for POS tagging. We hypothesize that this is due to overparametrization for morphology. POS tagging is a much more complex task that needs a larger number of trainable parameters (recall that LSTM parameters are shared across all tokens).

4.4 Named entity recognition

As Figure 6 shows, in NER the choice of subword pooling makes far less difference than in morphology. In terms of models, mBERT has a clear advantage over XLM-RoBERTa when it comes to NER. The difference between the two models is generally larger than the difference between two subword choices within the same language. The smallest difference between the two models appears to be in Czech, Finnish and German, which all have rich, partially agglutinative, morphology. This fits with our earlier findings that showed that XLM-RoBERTa might be better at handling rich morphology. Overall FIRST and the related F+L as well as LSTM come out as winners, the differences are rather small and often not statistically significant for CJK.

4.5 Discussion

Throughout our extensive experiments we observed that pooling strategies can have a significant impact on the conclusions drawn from probing experiments. When considering multiple typologically different languages, the strength of the conclusions

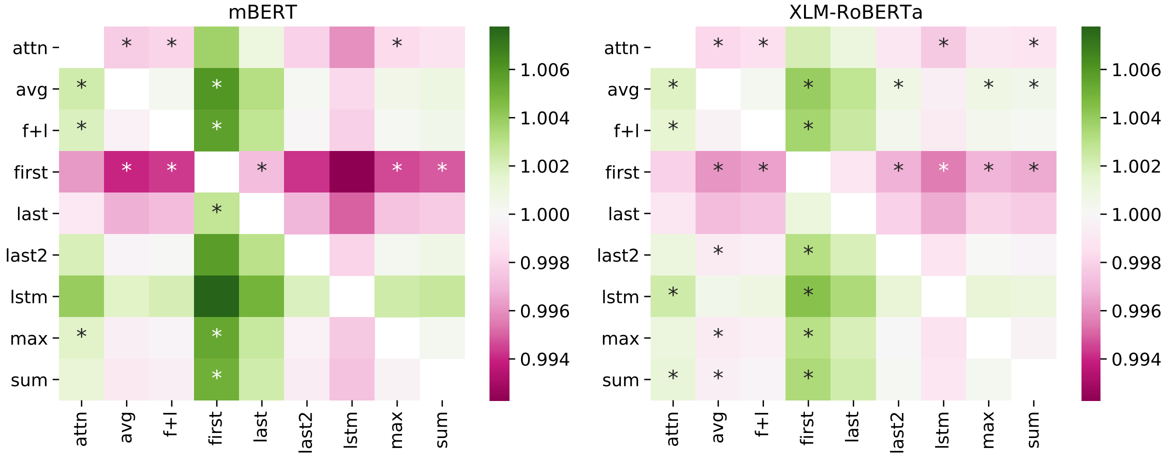


Figure 7: Subword choices for POS tagging in non-CJK languages. See Figure 5 for an explanation of the figure. FIRST is included.

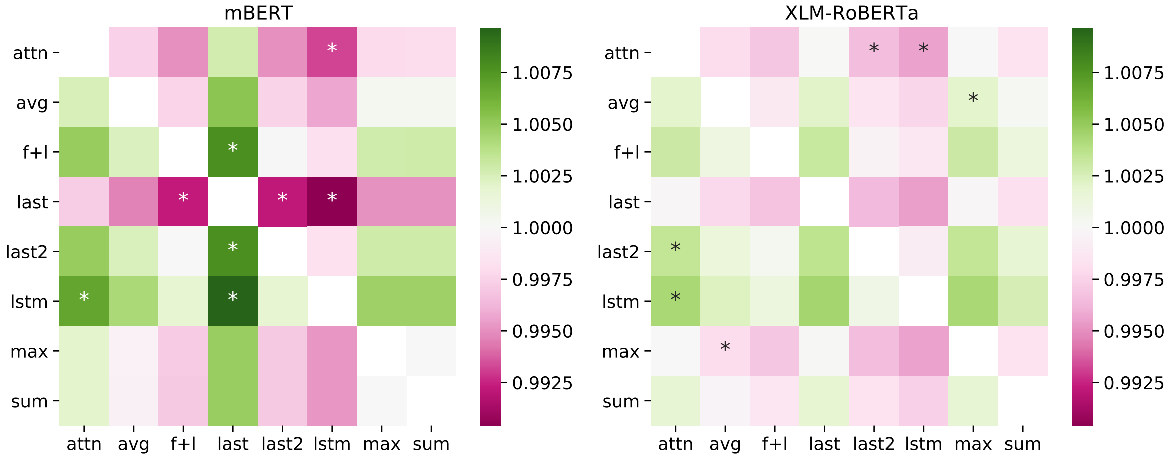


Figure 8: Subword choices for POS tagging in CJK languages. See Figure 5 for an explanation of the figure. FIRST is omitted for clarity as it is much worse than the other choices.

drawn from experiments can be weakened by considering a single pooling option. Our recommendation for NLP practitioners is to try at least three subword pooling strategies, particularly for tasks in languages other than English. FIRST and LAST usually gives a general picture – as a third control we recommend ATTN and LSTM. More complicated tasks such as POS or NER tagging may require LSTM with many parameters, while tasks that rely more on the orthographic representation such as morphology tend to benefit from ATTN.

One of the greatest attractions of the current generation of models is that they do away with labor-intensive feature engineering. Currently, subword pooling acts as the little finderscope mounted on the side of the main telescope to get it to point in the right region, but over the long haul we expect the systems to develop in a way that pooling also becomes part of the end to end process.

Our methodology is only limited by the availability of data. It would be interesting to extend these study with languages that use prefixes too such as Indonesian or Swahili.

5 Conclusion

The key takeaway from our work is that performance on lower level tasks depends on the way we pool over multiple subword units that belong in a single word token. This is more of an issue in languages other than English, where a significantly larger proportion of words are represented by multiple subword units.

Morphological and POS tasks are both probing word-level attributes, but the results show huge disparity: for the morphological tasks FIRST pooling is the worst strategy, and ATTN is the best, while for POS tagging ATTN is almost as bad as FIRST, the best being LSTM. The NER task is intermediary between word- and phrase-level, and subword pooling effects are less marked, but still statistically significant (see the full result tables in the Appendix).

Acknowledgments

This work was partially supported by the BME Artificial Intelligence TKP2020 IE grant of NKFIH Hungary (BME IE-MI-SC TKP2020) and by the Hungarian Ministry of Innovation and the National Research, Development and Innovation Office within the framework of the Artificial Intelligence National Laboratory Programme.

References

Fadi Biadsy, Nizar Habash, and Julia Hirschberg. 2009. Improving the Arabic pronunciation dictionary for phone and word recognition with linguistically-based pronunciation rules. In *Proceedings of human language technologies: The 2009 annual conference of NAACL*, pages 397–405.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

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

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Joakim Nivre, Mitchell Abrams, Željko Agić, et al. 2018. Universal Dependencies 2.3. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Cambridge, MA. SIGDAT/ACL.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Édouard Grave. 2020. Ccnet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4003–4012.

Chuhan Wu, Fangzhao Wu, Tao Qi, Xiaohui Cui, and Yongfeng Huang. 2020. Attentive pooling with learnable norms for text representation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2961–2970.

Appendices

A Data preparation

A.1 Morphological probes

We extract 2000 train, 200 dev and 200 test sentences for each task. We keep UD's original splits, in other words, all of our train sentences come from

UD's train set. We sample the sentences in a way that avoids overlaps in target words between train, dev and test splits, in other words, if a word is the target in the train set, we do not allow the same target word in the dev or test set. A target word is the word that needs to be classified according to some morphological tag. We also limit class imbalance to 3:1 at max. This results in the removal of rare tags such as a few of the numerous Finnish noun cases. These restrictions and the size of the treebanks do not allow generating larger datasets.

A.2 POS dataset

We use the largest treebank in each language for POS. The only preprocessing we do is that we filter sentences longer than 40 tokens. Since this results in an uneven distribution in the training size, we limit the number of training sentences to 2000. We note that experiments using 10,000 sentences are underway but due to resource limitations, we were unable to include them in this version of the paper.

A.3 NER dataset

NER is sampled from WikiAnn. WikiAnn is a silver standard large scale NER corpus and the number of sentences is over than 100,000 in each language. We deduplicated the dataset and discarded sentences longer than 40 tokens or 200 character in the case of Chinese and Japanese. WikiAnn annotates Chinese and Japanese at the character level. We aligned this with mBERT's tokenizer and retokenized it. Due to memory constraints, we had to cut off the training data size at 10,000.

B Training details

Each classifier is trained separately from randomly initialized weights with the Adam optimizer (Kingma and Ba, 2014) with $(lr = 0.001, \beta_1 = 0.9, \beta_2 = 0.999)$ and early stopping on the development set. We report test accuracy scores averaged over 3 runs with different random seeds.

We ran about 14,000 experiments on GeForce RTX 2080 GPUs which took 7 GPU days. We cache mBERT's and XLM-RoBERTa's output when possible. We used PyTorch and our own framework for experiment management. We release the framework along with the final submission.

C Full result tables

task	**model**	FIRST	LAST	LAST2	F+L	SUM	MAX	AVG	ATTN	LSTM
⟨Arabic, Case, NOUN⟩	mBERT	**76.5**	73.3	74	74.8	75	73.5	75	75.5	74.8
⟨Arabic, Case, NOUN⟩	XLM-Ro	77.1	74.5	74	75.6	**80.9**	80.1	79.3	77.8	74.5
⟨Arabic, Gender, ADJ⟩	mBERT	93.3	97.3	97	97.5	98.8	98.2	98.2	**99.3**	97.3
⟨Arabic, Gender, ADJ⟩	XLM-Ro	96.5	97.5	97.2	97.5	99	99	99	**99.5**	97.5
⟨Czech, Gender, ADJ⟩	mBERT	41.5	**77.1**	74	72.6	64.2	63	63	75.6	68.0
⟨Czech, Gender, ADJ⟩	XLM-Ro	41.6	**73.5**	71.8	70.8	64.2	65	63.2	73	66.5
⟨Czech, Gender, NOUN⟩	mBERT	61.7	77.4	77.9	74.6	76.3	75.1	74.8	**81.3**	79.3
⟨Czech, Gender, NOUN⟩	XLM-Ro	69.3	**84.1**	83.3	83.7	80.6	82.4	81.1	82.9	82.4
⟨English, Verbform, VERB⟩	mBERT	83.7	**91.2**	87.2	89.2	89	89.5	89.7	90.7	88.5
⟨English, Verbform, VERB⟩	XLM-Ro	82.7	91	89.7	90.8	**91.8**	91.7	89	91.3	89.7
⟨Finnish, Case, NOUN⟩	mBERT	33	90.7	90.7	87.7	87.4	86.2	88.1	**93.9**	92.0
⟨Finnish, Case, NOUN⟩	XLM-Ro	49.6	94.4	94.7	94.2	94.9	95.7	95.7	**96.0**	95.4
⟨Finnish, Verbform, VERB⟩	mBERT	74.6	92.5	91.4	92.5	89.9	90	90.9	**93.2**	91.9
⟨Finnish, Verbform, VERB⟩	XLM-Ro	88.6	**97.8**	97.5	**97.8**	96.7	97.3	97.2	**97.8**	96.5
⟨French, Gender, ADJ⟩	mBERT	87.7	**93.2**	92.8	**93.2**	92.7	92.7	92.3	92.2	92.5
⟨French, Gender, ADJ⟩	XLM-Ro	83.2	92	91.7	90.5	89.7	89.3	88.7	**92.5**	90.3
⟨French, Gender, NOUN⟩	mBERT	88.3	**96.7**	94.7	95.2	95.3	95.2	96	95.8	96.2
⟨French, Gender, NOUN⟩	XLM-Ro	93.3	93.5	94.8	**96.0**	95.8	95.7	95.2	95.8	95.8
⟨French, Verbform, VERB⟩	mBERT	95.5	**100.0**	99.7	99.5	99.7	99.3	99.5	99.8	99.5
⟨French, Verbform, VERB⟩	XLM-Ro	93.5	99.3	99.2	99.2	**99.5**	**99.5**	99.3	99.3	99.0
⟨German, Case, NOUN⟩	mBERT	63.2	**77.7**	72	75.3	74	74.5	75.3	77	74.0
⟨German, Case, NOUN⟩	XLM-Ro	64.7	71.8	70.8	**77.3**	74.2	75	74.2	71.3	73.7
⟨German, Gender, ADJ⟩	mBERT	58.9	77.9	**78.1**	73.6	67.7	70	71.6	76.1	73.0
⟨German, Gender, ADJ⟩	XLM-Ro	54.7	74.6	**75.0**	74.5	73.8	71.8	71.8	74	69.7
⟨German, Gender, NOUN⟩	mBERT	60.5	79.1	78.8	78.3	77.8	78.6	78.4	**79.8**	76.8
⟨German, Gender, NOUN⟩	XLM-Ro	61.2	84.4	85.7	82.8	84.1	85.6	85.1	**88.9**	86.9
⟨German, Verbform, VERB⟩	mBERT	89.2	94.5	94.5	**95.0**	94.5	**95.0**	**95.0**	**95.0**	94.2
⟨German, Verbform, VERB⟩	XLM-Ro	90.4	94.4	94.7	94.2	94.4	93.9	94.4	**95.0**	94.7

Table 6: Full list of morphological probing results at the 6th layer.

language	**model**	ATTN	AVG	F+L	FIRST	LAST	LAST2	LSTM	MAX	SUM
Arabic	mBERT	96.1	96.4	96.4	96.1	96.2	96.3	**96.5**	96.4	96.4
Arabic	XLM-RoBERTa	96.5	**96.8**	**96.8**	96.6	96.6	96.7	**96.8**	96.7	**96.8**
Chinese	mBERT	95.5	**96.2**	**96.2**	95.3	95.3	**96.2**	**96.2**	96.1	96.1
Chinese	XLM-RoBERTa	94.9	95.3	**95.4**	92.6	95.3	**95.4**	**95.4**	95.1	**95.4**
Czech	mBERT	98.2	98.2	**98.3**	97.5	98.2	**98.3**	**98.3**	98.2	98.1
Czech	XLM-RoBERTa	98.4	98.4	98.4	98.0	98.3	98.4	**98.5**	98.3	98.4
English	mBERT	95.8	**96.0**	**96.0**	95.5	95.7	95.8	**96.0**	95.8	95.7
English	XLM-RoBERTa	96.1	96.3	96.2	95.9	96.1	96.3	**96.4**	96.3	96.2
Finnish	mBERT	90.9	91.4	91.1	90.1	90.2	91.6	**92.3**	91.2	91.2
Finnish	XLM-RoBERTa	94.8	95	95	94.3	94.1	94.9	**95.3**	94.9	95
French	mBERT	97.4	**97.6**	97.5	97.4	97.4	97.4	97.4	97.5	**97.6**
French	XLM-RoBERTa	97.6	**97.7**	**97.7**	97.5	97.6	97.6	97.6	97.6	97.6
German	mBERT	97.8	97.9	**98.0**	97.4	97.9	**98.0**	**98.0**	97.9	97.9
German	XLM-RoBERTa	97.7	97.9	97.9	97.5	97.9	97.9	**98.0**	97.9	97.8
Japanese	mBERT	96.0	96.5	96.6	95.7	95.9	96.5	**96.8**	96.4	96.3
Japanese	XLM-RoBERTa	96.4	96.8	96.7	95.2	96.6	96.8	**97.0**	96.7	96.7
Korean	mBERT	94.1	93.7	94.3	84.8	93.6	94.3	**94.5**	93.7	93.8
Korean	XLM-RoBERTa	94.9	94.7	95	87.6	94.3	95	**95.1**	94.5	94.7

Table 7: Full list of POS tagging results at the 6th layer.

language	**model**	ATTN	AVG	F+L	FIRST	LAST	LAST2	LSTM	MAX	SUM
Arabic	mBERT	92.0	92.6	**92.8**	92.4	91.9	91.9	92.7	92.1	92.4
Arabic	XLM-RoBERTa	88.6	89.8	89.7	89.1	88.7	89.0	**90.0**	89.2	89.5
Chinese	mBERT	87.0	87	87	87	86.9	87.1	86.8	87.1	**87.2**
Chinese	XLM-RoBERTa	82.2	83.1	83.1	82.5	82.3	83.1	**83.6**	82.3	83
Czech	mBERT	95.3	95.5	**95.6**	95.3	95.0	95.3	95.5	95.4	95.4
Czech	XLM-RoBERTa	93.6	94.2	**94.3**	93.9	93.5	93.9	94.2	93.9	94
English	mBERT	90.8	91.2	**91.3**	91	90.7	90.7	**91.3**	91.1	91.2
English	XLM-RoBERTa	87.7	**88.5**	**88.5**	88.3	88.0	88.1	88.4	88.3	88.4
Finnish	mBERT	95.3	95.5	95.6	95.3	94.8	95.0	**95.7**	95.3	95.3
Finnish	XLM-RoBERTa	93.7	**94.4**	94.3	94	93.4	93.7	94.3	94.1	94.1
French	mBERT	93.8	**94.1**	**94.1**	93.8	93.2	93.5	94	94.0	93.8
French	XLM-RoBERTa	89.9	90.5	**90.6**	90.1	89.5	89.6	90.5	90.3	90.3
German	mBERT	95.4	95.7	**95.8**	95.6	95.1	95.2	95.7	95.5	95.5
German	XLM-RoBERTa	94.0	94.5	94.5	94.3	93.9	94.0	**94.6**	94.2	94.4
Japanese	mBERT	89.2	89.3	89.3	**89.4**	89.0	89.0	89.3	89.3	89.1
Japanese	XLM-RoBERTa	85.8	86.4	86.4	86.3	85.8	86.0	**86.6**	86.0	86.3
Korean	mBERT	92.8	93.3	93.2	92.8	92.1	92.6	**93.6**	93.0	93.1
Korean	XLM-RoBERTa	90.2	90.9	90.9	90	89.3	90.1	**91.1**	90.4	90.8

Table 8: Full list of NER results at the 6th layer.

Content-based Models of Quotation

Ansel MacLaughlin
Khoury College of Computer Science
Northeastern University
Boston, MA
`ansel@ccs.neu.edu`

David A. Smith
Khoury College of Computer Science
Northeastern University
Boston, MA
`dasmith@ccs.neu.edu`

Abstract

We explore the task of quotability identification, in which, given a document, we aim to identify which of its passages are the most quotable, i.e. the most likely to be directly quoted by later derived documents. We approach quotability identification as a passage ranking problem and evaluate how well both feature-based and BERT-based (Devlin et al., 2019) models rank the passages in a given document by their predicted quotability. We explore this problem through evaluations on five datasets that span multiple languages (English, Latin) and genres of literature (e.g. poetry, plays, novels) and whose corresponding derived documents are of multiple types (news, journal articles). Our experiments confirm the relatively strong performance of BERT-based models on this task, with the best model, a RoBERTA sequential sentence tagger, achieving an average ρ of 0.35 and NDCG@1, 5, 50 of 0.26, 0.31 and 0.40, respectively, across all five datasets.

1 Introduction

Unlike in scientific writing, where authors use terse citations due to space constraints, direct quotation of source material is an essential part of writing in many fields. Journalists, humanities scholars, and students, for instance, often quote from a range of source documents, such as interviews, speeches, and books. Quotes can be used to substantiate a claim, lend authority to an argument, or offer a viewpoint to reflect on or argue against, among many others. Modeling the process of quotation selection is thus an important step in modeling how authors compose entire new documents.

In this paper, we explore the problem of quotability identification, identifying which passages in a source work (e.g. *Hamlet*) are likely to be quoted by later, derived works (e.g. humanities journal articles). Prior research has attempted to identify the specific factors that influence a passage or document's quotability. Most work, therefore, has focused on manual feature engineering and development of careful analysis frameworks to test which features have statistically significant relationships to quote counts (Tan et al., 2018; Danescu-Niculescu-Mizil et al., 2012). We, instead, reframe quotability identification as a practical passage ranking task and evaluate how well models can rank the passages in a given document by their predicted quotability. We benchmark and analyze the performances of multiple models, exploring both BERT-based passage ranking models and feature-based models equipped with "quotability" features identified in prior work.

We collect five large-scale datasets to study this problem. Each dataset consists of sets of source documents, derived documents, and alignments between them – the direct quotes. The proposed datasets are diverse, allowing us to model quotability dynamics across multiple source and derived document genres and languages.

Uses for our proposed passage ranking task include: 1) to help users discover quotable source content for use in an essay or article (Tan et al., 2016; MacLaughlin et al., 2021); 2) to select striking quotes from new books to help readers get a sense of them; 3) for use in extractive summaries, search result snippets, and other compressed versions of a text – since quotability implies the ability of a passage to have meaning outside its original source context, quotability scores could be incorporated with traditional measures of informativeness and non-redundancy to determine which passages should be excerpted.

The primary contributions of this paper include:

- We present five large-scale quotability identification datasets (§4) which span multiple

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2296–2314
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

source genres (novels, poems, plays, scripture, etc.) and languages (English, Latin).

- We compare the performance of multiple models (§6) on each of the five datasets: 1) feature-based models with bag-of-words and "quotability" features drawn from prior work (Bendersky and Smith, 2012) and 2) state-of-the-art BERT-based models for passage ranking and sequential sentence tagging. The best performing model, a RoBERTA-based sequential sentence tagger, achieves an average ρ of 0.35 and NDCG@1, 5, 50 of 0.26, 0.31 and 0.40, respectively, across all five datasets.
- We focus on a single source text, the King James Bible, to conduct a thorough analysis (§8.2). We analyze similarities and differences in both quoting patterns and modeling performance across two sets of aligned quoting labels – one from a large collection of 18th-20th century American Newspapers (11M pages), the other from the set of all journal articles in JSTOR (12M articles).

2 Related Work

There has been substantial previous research in identifying and analyzing what makes specific source content popular, i.e. how many times a given source is quoted, cited, retweeted, etc. The source documents analyzed in prior work span a wide range of domains, from political speeches and debates (Tan et al., 2018; Niculae et al., 2015) to books (Bendersky and Smith, 2012), movie scripts (Danescu-Niculescu-Mizil et al., 2012), scientific articles (Guerini et al., 2012; Yogatama et al., 2011), tweets (Hong et al., 2011; Tan et al., 2014), and news articles (Bandari et al., 2012). The aligned derived documents also span multiple domains, from social media (Booten and Hearst, 2016; Danescu-Niculescu-Mizil et al., 2012; Bendersky and Smith, 2012; Bandari et al., 2012; Hong et al., 2011; Tan et al., 2014) to news articles (Tan et al., 2018; Niculae et al., 2015), and scientific papers (Guerini et al., 2012; Yogatama et al., 2011).

Prior work has focused on both 1) predicting the popularity of an entire source document, e.g. a scientific article's citation count (Yogatama et al., 2011) and 2) similar to our work, identifying which specific passages in a given source work will receive the most attention (Tan et al., 2018; Danescu-Niculescu-Mizil et al., 2012; Bendersky and Smith, 2012). The most similar work to ours, Bendersky and Smith (2012), also explores applications of quotability identification models to literary works (Project Gutenberg). However, due to their lack of supervised data, they instead focus on the problem of identifying what sorts of passages are *likely to be* quoted, rather than modeling what sorts of phrases are *actually* quoted in derived works.

The contributions of prior work have been, primarily, feature engineering (e.g. number of personal pronouns, use of negative/positive words) and design of testing frameworks to determine which features have a statistically significant relationship to quotablity/popularity in a *single language* and in a *specific domain of interest* (e.g. English language movie quotes and their popularity on IMDB: Danescu-Niculescu-Mizil et al. (2012)). Unlike prior work, our main contributions are not feature engineering, but a re-framing of the task as a practical document-level ranking task and an analysis of several models through extensive experiments with multiple datasets spanning various source and derived document genres and languages.

3 Problem Formulation

We formulate quotability identification as passage ranking, identifying which passages in a source document are likely to be quoted in related derived works. We measure quotability directly by counting how many times a passage is quoted across a collection of derived texts. Concretely, given a source document and a set of derived texts, we –

1. Use fuzzy text alignment methods from text reuse detection, e.g. Smith et al. (2015), to identify alignments (quotes) between subsequences in the source and derived texts.
2. Split the source text into passages (e.g. prose: by sentence, poetry: by verse, plays: by line).
3. Map the starts and ends of the (source, derived) alignments, i.e. quotes, to specific source passages.
4. Label each source passage with the number of alignments that overlap it.
5. Learn to rank the passages by their quote count labels.
6. Measure how well a model can rank the passages in a source document w.r.t. each other.

4 Datasets

We have collected five datasets across two languages (English, Latin) and multiple genres (novels, poems, plays, scripture, etc.) to study this problem. Each dataset consists of source texts (e.g. Shakespeare's plays, books of the Bible) and a set of alignments to a corpus of derived documents (e.g. humanities journal articles, news articles). Each alignment is an instance where a derived work quotes a passage(s) in a specific source work, e.g. an author quotes from *Hamlet*, "To be, or not to be: that is the question," in a humanities journal article.

4.1 Source Document Datasets

We use four publicly available[1] source document datasets:

- **King James Bible (KJB)**: the standard English Bible from the mid-17th to the early 20th century, consisting of 66 books (Old and New Testaments, no Apocrypha), with an average of 471 verses per book (median 217).
- **Shakespeare (SHAK)**: Shakespeare's 38 plays, with an average of 3,284 lines per play (median 3,246).
- **American & British Literature (ABL)**: a collection of 70 American and British great works from the 17th-20th centuries, containing books (e.g. *Emma*), poetry (e.g. "Eve of St. Agnes"), essays (e.g. "On The Duty of Civil Disobedience"), speeches (e.g. "I Have a Dream"), and legal documents (e.g. US Constitution), with an average of 6,165 passages per work (median 5,457).
- **Latin Texts (LAT)**: a collection of 329 works of prose and poetry from the Perseus Digital Library (Crane, 2001), with an average of 1,832 sentences per document (median 853).

As seen in Table 1, there is substantial variation across the four datasets, both in terms of total numbers of documents and passages in each dataset and in median numbers of passages per document and tokens per passage. For example, LAT contains a relatively large number of documents and passages, but each document is relatively short (median 853 passages/doc). ABL, on the other hand, contains roughly 5x fewer documents, but each document contains roughly 6x more passages (median 5,457). Passages in KJB are the longest (median 27 tokens), but documents in KJB contain the smallest number of passages (median 217). See Appendix A for lists of the source documents in each dataset.

4.2 Derived Documents

We work with three collections of derived documents which discuss and quote from the above source documents (see Table 2 for more details):

- **Chronicling America (CA)**: A publicly available collection of roughly 11 million historic (1789-1963) newspaper pages from the Library of Congress's Chronicling America collection (Library of Congress, 2005).
- **JSTOR: Early Journal Content (EJC)**: A publicly available subset of the entire JSTOR collection, containing approximately 644k articles published prior to 1923 in the United States and prior to 1870 elsewhere.
- **JSTOR: All (JA)**: The entire JSTOR journal collection, consisting of over 12 million academic journal articles (not publicly available).

4.3 Source-Derived Alignments

We use three different sets of alignments between our source and derived document collections to generate labels for our datasets[2]:

- **KJB - CA**: alignments from America's Public Bible (Mullen, 2016). Using text reuse detection methods, Mullen (2016) identified quotations of the Bible or verbal allusions to specific biblical verses in newspapers from the Chronicling America collection. There are a total of 866,127 quotes from 383,387 unique pages across 1,706 different newspapers.
- **JSTOR Understanding Series**: alignments from the JSTOR Understanding Series (JSTOR Labs, 2019). The JSTOR Labs team created a database of all quotations within JSTOR, then, using text reuse detection methods, aligned those quotations to passages in a number of great works, including the King James Bible, Shakespeare, and American and British Literature datasets. JSTOR has provided us with the set of all alignments. There are a total of 65,093 quotes from 30,876 derived documents aligned to the Bible, 131,712 quotes from 24,060 derived documents aligned to Shakespeare's plays, and 130,582 quotes from

---

[1] KJB: Wikitext, ABL: Wikitext & Project Gutenberg, SHAK: Folger Shakespeare Library, LAT: Perseus

[2] The KJB-CA alignments (Mullen, 2016), the CA collection (Library of Congress, 2005) and the EJC are available for download. We also release processed, labeled versions of our KJB-CA and LAT-EJC datasets for download here. We are unable to release any JA data.

Source	Lang	Genre	Passage Type	# Docs	# Passages	Mdn # Passages / Doc	Mdn # Toks / Passage
KJB	en	Scripture	Verse	66	31,102	217	27
SHAK	en	Play	Line	38	124,809	3,246	9
ABL	en	Various	Verse, Sentence	70	431,580	5,457	19
LAT	la	Various	Sentence	329	602,676	853	13

Table 1: Summary statistics for the four source document datasets – King James Bible (KJB), Shakespeare's plays (SHAK), and collections of great works of literature from America and Britain (ABL) and the ancient Roman world (LAT).

Derived	Doc Type	# Documents
CA	newspaper pages	≈ 11,000,000
EJC	journal article	138,636
JA	journal article	≈ 12,000,000

Table 2: Summary statistics for the three derived document datasets – Chronicling America (CA), JSTOR: Early Journal Content (EJC) and JSTOR: All (JA).

28,986 derived documents aligned to the collection of American and British literature.

- **LAT - EJC**: We use the passim text alignment software (Smith et al., 2015) to detect quotes of the Latin texts in the JSTOR EJC using the Smith–Waterman alignment algorithm. This yielded a total of 124,679 aligned quotes from 26,619 derived journal articles.

As noted above, given a source document, a collection of derived documents, and set of alignments between the source and derived texts, we split the source into passages (sentences/lines/verses/etc) then count the number of times each passage is aligned to (in part or wholly) a distinct portion of a derived text. We then use these quote counts to label each passage in the source. See Appendix B for discussion of the implementation and accuracy of the different alignment-detection models.

As seen in Table 3, there is substantial variation across datasets with respect to the total number of aligned quotes and the proportion of source passages that are quoted. On one end of the spectrum, since there are many alignments between KJB and CA and KJB is a relatively small source (in terms of number of passages), 84% of source passages in the KJB-CA alignment dataset are quoted at least once, the median passage is quoted five times, and the median document contains a maximal passage quoted 480 times. On the other end of the spectrum, the ABL and LAT source datasets both contain a large number of passages, and there are relatively few alignments, leading to significantly sparser ABL-JA and LAT-EJC alignment datasets. Only 18% (ABL) and 25% (LAT) of passages are quoted at least once, and the median documents contain maximal passages quoted 43 (ABL) and 7 (LAT) times. In the middle are the KJB-JA and SHAK-JA datasets, where slightly over half of the passages are quoted at least once, the passages in the 75th percentile are quoted three times, and the median documents contain maximal passages quoted 30 (KJB-JA) and 53 (SHAK-JA) times.

5 Linguistic Attributes of Quotations

As a first step at modeling the quotability identification problem, we draw on quotability features from prior work (Bendersky and Smith, 2012; Danescu-Niculescu-Mizil et al., 2012; Tan et al., 2018) and attempt to identify which linguistic attributes influence passage quotability. We explore both lexical (e.g. passage begins with a stop word) and part of speech features (e.g. contains past tense verb).

For each dataset, we featurize each of the corresponding passages, then compare its highly-quoted (top 20%) and minimally-quoted (bottom 20%) passages. We use Welch's t-test with Bonferroni correction to test whether the scores for each feature are significantly different between the two groups.

Table 4 discusses each feature and the associated statistic and significance for each dataset. Most features' relationship with quotability varies across the five datasets. For instance, highly-quoted passages are *shorter* in KJB-CA, but *longer* in other datasets. We suspect this reflects a difference between a popular audience (CA) who might prefer short Bible verses with a succinct message vs. academia (JSTOR), where writers are focused on careful passage analysis and less space-constrained. We also observe differences between the two KJB datasets and the other datasets. In the KJB, the relationship between quotability and presence of dialogue words (says, etc.) is positive, but negative in the others. This difference may be because many important Bible verses report direct speech by Jesus or God, whereas the other datasets, e.g. ABL, contain many uninteresting dialogue passages that serve to move the story along (e.g. "'No,' said the boy").

Source	Derived	# Aligned Derived Quotes	# Aligned Derived Docs	# Quotes / Source Passage Q_1	Q_2	Q_3	Max	% Passages w/ ≥ 1 Quote	Mdn Max # Quotes / Doc
KJB	CA	866,127	383,387	1	5	19	4,949	84%	480
KJB	JA	65,093	30,876	0	1	3	303	58%	30
SHAK	JA	131,712	24,060	0	1	3	905	58%	53
ABL	JA	130,582	28,986	0	0	0	1,536	18%	43
LAT	EJC	124,679	26,619	0	0	0	62	25%	7

Table 3: Summary statistics for the five sets of source-derived alignments. Each set of alignments is between one source dataset (KJB, SHAK, ABL, LAT) and one derived document dataset (CA, JA, EJC). Each source passage is labeled by the total number of times it is quoted in the corresponding derived documents. # Quotes / Source Passage measures of how many quotes each source passage receives - we display the first, second and third quartiles and the max for passages across the entire source dataset. % Passages w/ $\geq$ 1 Quote measures what percentage of passages in the source dataset have at least 1 aligned quote. To calculate Mdn Max # Quotes / Doc, we find the most quoted passage in each document in the source dataset, then take the median over those quote counts.

Feature Set		KJB-CA	KJB-JA	SHAK-JA	ABL-JA	LAT-EJC
Length: Number of tokens in passage. (Bendersky and Smith, 2012)	# Words	↓↓↓↓	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑
Capitalized: Proportion of words capitalized (Bendersky and Smith, 2012).	% Capital	↓↓↓↓	↓↓↓↓	↓↓↓↓	↓↓↓↓	↑↑↑↑
Stop Words: 1) Proportion of words in passage that are stop words, 2) Binary feature if passage begins with a stop word (Bendersky and Smith, 2012).	% Stop	–	↑↑↑↑	↑↑↑↑	↑↑↑↑	↓↓↓↓
	Begin-Stop	–	–	–	↑↑	↓↓↓↓
Generality: 1) Proportion of words in passage that are indefinite articles 2) Binary feature if passage contains an abstract noun (Bendersky and Smith, 2012; Danescu-Niculescu-Mizil et al., 2012; Tan et al., 2018).	% Indefinite	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑	×
	Abstract	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑	×
Universal Quantifiers: Binary feature if passage contains a universal quantifiers (20 quantifiers, e.g., all, whole, nobody). (Bendersky and Smith, 2012).	Universal	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑	×
Pronouns: Proportion of words in passage that are first, second, or third person pronouns (Tan et al., 2018).	% 1st	↑↑↑↑	↑↑↑↑	–	↓↓↓↓	↑↑↑↑
	% 2nd	↑↑↑↑	↑↑↑↑	↓↓↓↓	↓↓↓↓	↑↑↑↑
	% 3rd	–	↓↓↓↓	↓↓↓↓	↓↓↓↓	×
Language Model: We compute the ratio between the log-likelihood of the passage under a LM trained on a collection of popular quotes and one trained on a background corpus (Bendersky and Smith, 2012, eq. 1). The more likely a passage is under the quotable LM relative to the background LM, the higher the ratio will be (Danescu-Niculescu-Mizil et al., 2012; Tan et al., 2018).	LLR	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑	×
Dialogue Words: Binary feature if passage contains a dialogue word (e.g. says, dicit). (Bendersky and Smith, 2012)	Dialogue	↑↑↑↑	↑↑↑↑	↓	↓↓↓↓	↓↓↓↓
Emphasis: 1) Binary feature if passage contains a comparative adjective or adverb form, 2) Binary feature if passage contains a superlative adjective or adverb form (Bendersky and Smith, 2012; Tan et al., 2018).	Comparative	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑	↑↑↑↑
	Superlative	↑	–	↑↑↑↑	↑↑↑↑	↑↑↑↑
Verb Tenses: Proportion of words in passage that are past or present tense verbs (Bendersky and Smith, 2012; Danescu-Niculescu-Mizil et al., 2012).	Past	–	↓↓↓↓	↑↑↑↑	↓↓↓↓	↑↑↑↑
	Present	↑↑↑↑	↑↑↑↑	↓↓↓↓	–	↓↓↓↓

Table 4: Results from examination of linguistic attributes of quotations. Upward arrows indicate that frequently quoted passages (top 20% of source passages by # of quotes) have larger scores in that feature, while downward arrows indicate that passages with very few or no quotes (bottom 20% of source passage by # of quotes) have larger scores in the feature (↑↑↑↑: $p < 0.0001$, ↑↑↑: $p < 0.001$, ↑↑: $p < 0.01$, ↑: $p < 0.05$). p refers to the p-value after the Bonferroni correction. × indicates that the feature set is not available for the corresponding dataset. A '–' symbol indicates that there is no significant relationship.

While there are no features whose relationship with quotability is negative across all datasets, there are a few that are consistently significantly positive: **1) general language:** highly-quoted passages contain more general language which can be more easily adapted to a new contexts, such as indefinite articles and abstract concepts (e.g. adventure, charity); **2) universal quanitifiers:** similar to maxims, proverbs and other short, pithy statements, quotable phrases often contain universal quantifiers (e.g. always, never); **3) comparative adjectives:** highly-quoted passages use more comparative adjectives to compare a noun to something else; **4) language model**: quotable phrases have higher likelihoods under our quotable language model than a background language model trained on the corresponding source corpus.

6 Models

Next, we examine the effectiveness of two classes of models on our passage ranking task: feature-based models with "quotability" features and neural models for sentence classification. We benchmark both pointwise regression models for count data and listwise and pairwise ranking models.

6.1 Feature-based

We evaluate three feature-based models: **poisson regression**, **SVM^{rank}** (Joachims, 2002) and **lambdaMART** (Burges, 2010). Poisson regression (PR) is a generalized linear model for count data, SVM^{rank} is a commonly used pairwise feature-based ranker, and lambdaMART (λMART) is a state-of-the-art feature-based listwise ranker. Each feature-based model uses the same set of features: **bag-of-word features**, **"quotability" features** (Bendersky and Smith, 2012; Danescu-Niculescu-Mizil et al., 2012; Tan et al., 2018, Including the features discussed in §5), and **positional features** (e.g. relative position in book). See Appendix E for a list of all features.

6.2 BERT-based

BERT-based models (Devlin et al., 2019) have recently achieved state-of-the-art performance on multiple sentence classification tasks, including single sentence classification, e.g. sentiment analysis, and classification of a sequence of sentences into their corresponding categories, e.g. extractive summarization (Liu and Lapata, 2019) and scientific abstract sentence classification (Cohan et al., 2019). We work with two BERT-based models, RoBERTA (Liu et al., 2019) and XLM-RoBERTA (XLM-R) (Conneau et al., 2019). We fine-tune RoBERTA on our English datasets and XLM-R on the Latin dataset. We benchmark both models as both single passage and sequential passage predictors:

- **RoBERTA**$_{single}$ **/ XLM-R**$_{single}$: we fine-tune RoBERTA and XLM-R on individual passages. Thus, each training example contains a single passage and its quote count label. We follow the standard fine-tuning setup, using the final hidden state of the [`CLS`] token as the aggregate representation for a passage and feeding it into an output layer for prediction.

- **RoBERTA**$_{seq}$ **/ XLM-R**$_{seq}$: in order to model a passage's context in the broader document, we also fine-tune RoBERTA and XLM-R as passage-level sequence taggers (Cohan et al., 2019). Due to the models' 512 WordPiece length limit, we break up each document into 512 WordPiece segments and feed those into the models independently. Thus, each training example contains some number n of consecutive passages from the same work (up to 512 WordPieces total) and n quote count labels, one for each passage. Following Cohan et al. (2019), we insert a [`SEP`] token between each of the n passages in each example. We use the final hidden state of each [`SEP`] token as the aggregate representation for each passage and feed it into a multi-layer feedforward network to make a prediction.

Since we are modeling quote counts for each passage, we train all $single$ and seq models with poisson negative log likelihood loss.

Finally, since our task is a ranking task, we also benchmark a BERT-based pairwise ranker, as pairwise neural rankers have shown strong performance on other ranking tasks, such as ad-hoc retrieval (Xiong et al., 2017; Dai et al., 2018):

- **RoBERTA**$_{pair}$ **/ XLM-R**$_{pair}$: Each training example consists of two passages sampled from the same work and a single label for the pair based on which passage is quoted more. Each passage is fed into RoBERTA or XLM-R separately and follows the standard fine-tuning setup as described for RoBERTA$_{single}$.

We train RoBERTA$_{pair}$ and XLM-R$_{pair}$ with hinge loss: $\mathcal{L}(s^+, s^-; \Theta) = max(0, 1 - f(s^+) + f(s^-))$, where passages s^+ and s^- are from the same document, s^+ is quoted more than s^-, and $f(s)$ is the output of running passage s through RoBERTA/XLM-R and the final output layer.

For all neural models, we also add special tokens to each passage to act as positional indicators so the model has a better sense of which part of the document it is reading (Alberti et al., 2019). These special tokens vary by dataset, but are generally added to the start or end of a passage and have forms such as [`Starts-Paragraph`], [`Ends-Act`], or [`Book@N`] where `N` is the decile in the book in which the passage occurs. See Appendix F for the full list of positional tokens used in each dataset.

7 Experimental Settings

We evaluate models using five-fold cross validation. We train and evaluate models on each dataset separately and report means over the five folds for each. We split datasets into folds at the document level (e.g. train: *Frankenstein*, etc., val: *Emma*, etc., test: *Paradise Lost*, etc.). Then, for a given validation or test set, we evaluate performance on each document separately, then average over documents.

For the feature-based models, we select hyperparameters by performing nested five-fold cross validation on each fold's training set, again splitting by document. Due to computational restrictions, for the neural models we select hyperparameters by using 20% of the documents in the fold's training set as a validation set. See Appendix H for a list of all hyperparameters.

7.1 Evaluation Metrics

Since our task is a passage ranking task, we evaluate models using two common ranking metrics: average NDCG@k (Croft et al., 2009) and spearman's ρ. NDCG@k is defined as $\frac{DCG_k}{IDCG_k}$, where $DCG_k = quoted_1 + \sum_{i=2}^{k} \frac{quoted_i}{log_2(i)}$ and $quoted_i$ is the number of times the passage ranked in the ith position by the model has been quoted across all corresponding derived documents, and $IDCG_k =$ the ideal DCG_k, the maximum DCG_k computed by ranking passages by their true quote counts. We evaluate NDCG@k at six ranks, $k \in [1, 3, 5, 10, 25, 50]$. We calculate both NDCG and ρ at the document level, i.e. ranking a passage in a document versus all other passages in that document, then average across documents.

8 Results & Analysis

Tables 5 and 6 display the results of all models across all five datasets[3]. On the whole, we find that modeling a passage's context in its broader document is important, with the BERT-based sequential sentence models RoBERTA$_{seq}$ and XLM-R$_{seq}$ performing the best. They achieve the highest average NDCG score across all ranks k on three datasets (KJB-CA, SHAK-JA, LAT-EJC) and the highest ρ on all five datasets. The sequential sentence models achieve their highest *relative* performance on SHAK-JA, outperforming the second-best models by roughly 40% relative on NDCG and 19% on ρ.

Similar to results on other passage ranking tasks (Nogueira and Cho, 2019; Qiao et al., 2019), we find that the single passage BERT-based models (RoBERTA$_{single}$, XLM-R$_{single}$) provide strong baselines, outperforming the feature-based models on nearly all datasets and achieving the second-best overall performance. Furthermore, on KJB-JA, RoBERTA$_{single}$ achieves the best NDCG performance, outperforming RoBERTA$_{seq}$ by an average of 0.02 across each k. Investigating the KJB-JA results further, we find that Roberta$_{single}$ outperforms Roberta$_{seq}$ on 34 of the 64 KJB books. Of these 34, five books (Numbers, Revelation, Zechariah, Leviticus, and Peter-1) account for over a third of the total increase in NDCG over Roberta$_{seq}$. In four of the five books, Roberta$_{single}$ successfully ranks the most quoted passage at the top, while Roberta$_{seq}$ ranks passages with single- or near-single-digit labels. Notably, Roberta$_{seq}$ fails, while Roberta$_{single}$ succeeds, in properly ranking the Great Commandment from Leviticus 19:18 "... love thy neighbour as thyself" at the top.

On the other hand, the pairwise BERT-based models (RoBERTA$_{pair}$, XLM-R$_{pair}$) generally perform worse than the single passage and sequential passage BERT-based models. They struggle to identify the top, most quoted passages in each document (as measured by NDCG), but perform relatively better at ranking each document's entire list of passages w.r.t each other (ρ), though still worse than the sequential passage models.

Among the feature-based models, while SVM^{rank} achieves the highest ρ on all five datasets, no single model consistently outperforms the others in NDCG, with PR, SVM^{rank}, and λMART each achieving the highest scores on different datasets. However, only one feature-based model, λMART, ever outperforms the neural models, achieving the highest NDCG scores on ABL-JA. We hypothesize that λMART's strong performance on ABL-JA might be due, in part, to differences in the accuracy of our feature-extraction pipeline. Many of our features depend on accurate parsing (e.g. POS and verb tense counts). However, for our English datasets, the Stanza Universal Dependencies model (Qi et al., 2020) we use to process each sentence is trained on web-media data (UD English EWT). Thus, our English datasets (KJB, ABL, SHAK) are all out-of-domain. We hypothesize that Stanza is more accurate on ABL since it contains the most modern language similar to its training data. With these higher quality inputs, therefore, feature-based models can achieve higher performance, relative to the neural models. As one might note, our datasets are also out-of-domain for RoBERTa and XLM-R. However, as shown by Han and Eisenstein (2019), BERT-based models can adapt to new domains when provided with in-domain fine-tuning data. For our Latin data, although our LAT-EJC dataset is in-domain for the Latin Stanza model (UD Latin Perseus), the Stanza model might not perform as well since the UD training data is quite small.

8.1 Differences in Model Performance Across Datasets

As seen in Tables 5 and 6, performance of individual models varies substantially across the five datasets. Examining the average NDCG@k scores,

---

[3] There are only 64 total documents in KJB-JA since there are no aligned quotes in John-2 or Kings-2.

	KJB - CA (66 books)							**KJB - JA (64 books)**						
	Average NDCG@							**Average NDCG@**						
	1	3	5	10	25	50	ρ	1	3	5	10	25	50	ρ
PR	19.5	25.8	26.9	30.0	35.8	41.1	27.3	32.3	33.0	33.6	34.7	39.7	45.3	21.1
SVM^{rank}	12.7	17.0	19.7	23.1	29.9	35.4	33.4	21.6	23.0	24.7	28.8	34.3	41.1	23.9
λMART	18.4	18.1	20.2	22.7	28.6	34.8	23.4	13.3	18.6	21.4	25.2	31.7	38.1	20.4
RoBERTA$_{single}$	**24.8**	27.1	30.4	34.6	41.2	46.0	41.0	**34.3**	**35.3**	**35.9**	**38.5**	**44.3**	**49.8**	30.9
RoBERTA$_{pair}$	19.3	26.0	28.9	32.0	37.5	43.1	39.8	25.8	25.6	27.1	30.8	38.1	44.1	29.2
RoBERTA$_{seq}$	23.1	**31.4**	**34.7**	**37.2**	**43.1**	**47.7**	**42.0**	31.9	31.3	33.7	37.7	43.6	49.2	**34.1**

	ABL - JA (70 works)							**SHAK - JA (38 plays)**						
	Average NDCG@							**Average NDCG@**						
	1	3	5	10	25	50	ρ	1	3	5	10	25	50	ρ
PR	22.1	21.7	22.9	23.6	25.0	26.8	20.9	9.4	11.4	12.3	14.0	16.4	18.7	23.9
SVM^{rank}	17.2	20.6	21.7	22.9	25.2	27.3	25.7	11.5	12.8	13.0	14.4	16.5	19.2	35.7
λMART	**31.1**	**27.6**	**28.6**	**30.6**	**31.2**	**32.8**	24.3	9.3	10.4	10.8	12.4	14.7	16.9	29.8
RoBERTA$_{single}$	16.3	20.5	21.7	23.8	26.0	28.8	24.6	10.6	18.3	19.2	20.2	22.8	25.1	38.6
RoBERTA$_{pair}$	20.1	22.5	22.9	24.3	27.6	30.0	28.2	11.1	16.1	17.1	18.5	21.0	23.7	38.2
RoBERTA$_{seq}$	16.5	23.2	24.2	26.0	29.2	32.2	**31.5**	**24.8**	**23.6**	**25.0**	**27.6**	**30.2**	**33.5**	**46.7**

Table 5: 5-fold cross validation results on the King James Bible, with both Chronicling America (CA) and JSTOR All (JA) alignments, and on the American and British Literature (ABL) and Shakespeare (SHAK) datasets, both with alignments from JSTOR All (JA). We report NDCG across six positions (1, 3, 5, 10, 25, 50). Reported NDCG and ρ values are averaged across documents within each fold then averaged across folds.

a measure of how well a model identifies the top k most quoted passages in a given document, we find that models struggle most with the Shakespeare dataset, SHAK-JA, and achieve the best scores on the King James Bible with alignments from JSTOR, KJB-JA. On the other hand, examining the ρ scores, a measure of how well a model ranks *all* of the passages in a given document with respect to each other, we find that models struggled most on the Latin dataset, LAT-EJC, and achieve their highest scores on the Shakespeare dataset, SHAK-JA.

We hypothesize that these differences in modeling performance might be due, in part, to high-level, non-linguistic differences between the datasets, such as the average number of passages in each source work. First, examining NDCG scores, we find that scores are generally higher on datasets where the documents have relatively few passages (KJB, LAT) than on those where the documents contain many passages (SHAK, ABL). Since NDCG evaluates how well models identify the top k passages in a given document, this relative performance difference is understandable because shorter passage lists are likely easier to rank than very long ones (e.g. 155 verses in Ephesians vs. 9,426 sentences in Moby Dick). On the other hand, we do not find clear relationships between NDCG and either 1) the proportion of passages that are quoted at least once; or 2) the size of the dataset (total # passages). Models have relatively similar performances on 1) both sparsely quoted (ABL-JA) and highly quoted (SHAK-JA, KJB-CA) datasets; and 2) both large (LAT-EJC) and small (KJB-JA) datasets.

	LAT - EJC (329 works)						
	Average NDCG@						
	1	3	5	10	25	50	ρ
PR	28.0	27.8	27.6	28.2	29.5	31.0	12.1
SVM^{rank}	25.5	25.0	26.6	27.7	29.8	31.9	16.3
λMART	32.5	29.9	29.6	30.0	31.3	32.7	14.2
XLM-R$_{single}$	29.4	30.2	30.1	30.6	32.0	33.6	14.1
XLM-R$_{pair}$	25.1	26.2	27.0	27.9	29.5	31.5	16.5
XLM-R$_{seq}$	**35.8**	**35.6**	**36.3**	**35.8**	**37.0**	**38.7**	**19.4**

Table 6: 5-fold cross validation results on the collection of Latin texts (LAT) with alignments from the JSTOR Early Journal Collection (EJC). We report NDCG across six positions (1, 3, 5, 10, 25, 50). Reported NDCG and ρ values are averaged across documents within each fold then averaged across folds.

Finally, inspecting differences in ρ scores across datasets, we find that although scores are generally higher on datasets with a high proportion of quoted passages (SHAK-JA, KJB-CA), this trend does not always hold – models have approximately equal ρ on KJB-JA and ABL-JA, though 40% more of the passages in KJB-JA (58%) are quoted at least once compared to ABL-JA (18%).

8.2 King James Bible: Analysis

We conduct a thorough analysis on a single source text, the King James Bible. We select KJB since it is aligned to two separate derived document collections, Chronicling America (KJB-CA) and JSTOR: All (KJB-JA), allowing us to conduct comparative analysis across two different labelings of the same source document. We focus on differences in quoting attention and relative modeling performances.

Quoting Attention: We first examine differences in quoting attention aggregated at the book level. To identify which books of the Bible receive the most attention, we rank them by both their median and maximum number of quotes per passage. Table 7 lists the top 3 books by both metrics (mdn & max) for each dataset. There is only one book in common across both lists, the Gospel of Luke. The top 6 books in CA are all from the New Testament, while the top 6 in JA are split evenly between the Old and New Testaments, perhaps indicating a difference between a popular (CA) and scholarly (JA) audience. We calculate the overall similarity in book-level quoting attention between KJB-JA and KJB-CA by computing correlation (ρ) between the datasets' aggregate quote counts: ρ is 0.63 ranking by max passage and 0.62 by median.

Next, we compare differences in quoting attention over specific passages. We iterate over each book and compute ρ between the passage-level quote counts from each dataset. The average ρ across the 64 books[4] is 0.40. The books with the most similar and dissimilar quoting attention are Philippians and Nahum, respectively.

Modeling Performance: We focus on RoBERTA$_{seq}$ for our analysis since it achieves the best ρ on both datasets and the best NDCG on KJB-CA and second best on KJB-JA. We focus on relative performances at the book level. For each Bible book in each alignment dataset, we compute a composite model score by averaging RoBERTA$_{seq}$'s ρ and NDCG (averaged across all k) scores. We then compute the correlation (ρ) between this model score for each book and the book's 1) *length:* book length, in total # of passages 2) *proportion*: proportion of passages quoted at least once 3) *median:* median quote label 4) *max:* maximum quote label, and 5) *entropy*: entropy of the distribution of quotes over passages. On the whole, we find that RoBERTA$_{seq}$ performs better (i.e. higher model scores) on books with high median quote labels (*median*) and many quoted passages (*proportion*). Specifically, under KJB-JA labels, we find positive correlation between model score and *median* (0.42), *max* (0.42), and *proportion* (0.34) and no correlation with *length* or *entropy*. Under KJB-CA labels, we find weaker, positive correlation between model score and *median* (0.23) and *proportion* (0.21), no correlation with *max*, and negative correlation with *length* (-0.27) and *entropy* (-0.26).

	JA	**CA**
Max	John, Genesis, Luke	Matthew, Mark, Luke
Mdn	Song of Solomon, Revelation, Jonah	James, First John, Ephesians

Table 7: The top 3 most quoted books of the King James Bible as measured by **Max**: the maximum quoted passage in each book; **Mdn**: the median quote count in each book. We compare quote counts from alignments between the KJB and both JSTOR All (JA) and Chronicling America (CA).

Finally, we make a comparison between model scores across the two datasets, computing ρ between the two scores for each book. We find that RoBERTA$_{seq}$ performs relatively similarly on books across the two datasets, with moderate correlation of 0.45 between the two sets of scores.

9 Conclusion

We explore the task of quotability identification – identifying which passages in a source document are likely to be directly quoted by later derived documents. We cast quotability identification as a passage ranking problem, evaluating how well models can learn to rank the passages in a source document by their predicted quotability. We evaluate on five large-scale datasets spanning multiple source genres (e.g. poetry, novels, plays) and languages (English, Latin). We conduct experiments with feature- and BERT-based models, finding that although relative performances vary across datasets, on the whole, BERT-based models operating on strings of sequential passages perform best.

We have identified two potential directions of future research using the datasets described in this study. First, using the publication date information for the journal articles and newspapers, we could investigate temporal quoting trends, testing hypotheses such as the Matthew effect, and therefore the best predictor of a passage's quotability tomorrow is its popularity today. Finally, we could explore second-order effects, studying trends in what sorts of passages are often quoted together.

Acknowledgements

Ansel MacLaughlin was supported by a National Endowment for the Humanities Digital Humanities Advancement Grant (HAA-263837-19).

---

[4] We ignore John-2 and Kings-2 since they are not quoted in KJB-JA.

References

Chris Alberti, Kenton Lee, , and Michael Collins. 2019. A bert baseline for the natural questions. In *arXiv preprint arXiv:1901.08634*.

Roja Bandari, Sitaram Asur, and Bernardo A Huberman. 2012. The pulse of news in social media: Forecasting popularity. In *Proceedings of ICWSM*.

Michael Bendersky and David Smith. 2012. A dictionary of wisdom and wit: Learning to extract quotable phrases. In *Proceedings of the NAACL-HLT 2012 Workshop on Computational Linguistics for Literature*, pages 69–77, Montréal, Canada. Association for Computational Linguistics.

Kyle Booten and Marti A. Hearst. 2016. Patterns of wisdom: Discourse-level style in multi-sentence quotations. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1139–1144, San Diego, California. Association for Computational Linguistics.

Chris J.C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview. Technical report, Microsoft Research.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Dan Weld. 2019. Pretrained language models for sequential sentence classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3693–3699, Hong Kong, China. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, F. Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *ArXiv*, abs/1911.02116.

Gregory R. Crane. 2001. Perseus digital library. Tufts University. `http://www.perseus.tufts.edu`.

W. Croft, Donald Metzler, and Trevor Strohman. 2009. *Search Engines - Information Retrieval in Practice*. Pearson.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *WSDM*.

Cristian Danescu-Niculescu-Mizil, Justin Cheng, Jon Kleinberg, and Lillian Lee. 2012. You had me at hello: How phrasing affects memorability. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 892–901, Jeju Island, Korea. Association for Computational Linguistics.

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

Marco Guerini, Alberto Pepe, and Bruno Lepri. 2012. Do linguistic style and readability of scientific abstracts affect their virality? In *Proceedings of ICWSM*.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Liangjie Hong, Ovidiu Dan, and Brian D Davison. 2011. Predicting popular messages in twitter. In *Proceedings of WWW*.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *KDD*.

JSTOR Labs. 2019. Jstor understanding series. `https://guides.jstor.org/understandingseries`.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Library of Congress. 2005. Chronicling america: Historic american newspapers site. `https://chroniclingamerica.loc.gov/`.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arxiv preprint: 1907.11692*.

Ansel MacLaughlin, Tao Chen, Burcu Karagol Ayan, and Dan Roth. 2021. Context-based quotation recommendation. *ICWSM*.

Lincoln Mullen. 2016. America's public bible: Biblical quotations in u.s. newspapers, website, code, and datasets. `http://americaspublicbible.org`.

Vlad Niculae, Caroline Suen, Justine Zhang, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2015. Quotus: The structure of political media coverage as revealed by quoting patterns. In *WWW*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *ArXiv*, abs/1901.04085.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of bert in ranking. *ArXiv*, abs/1904.07531.

D. A. Smith, David R. Cordell, and David A. Ryan Abby Mullen. 2015. Computational methods for uncovering reprinted texts in antebellum newspapers. *American Literary History*, 27:E1 – E15. `https://github.com/dasmiq/passim`.

Chenhao Tan, Lillian Lee, and Bo Pang. 2014. The effect of wording on message propagation: Topic- and author-controlled natural experiments on twitter. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 175–185, Baltimore, Maryland. Association for Computational Linguistics.

Chenhao Tan, Hao Peng, and Noah A. Smith. 2018. "You are no Jack Kennedy": On media selection of highlights from presidential debates. In *WWW*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2016. A neural network approach to quote recommendation in writings. In *CIKM*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russel Power. 2017. End-to-end neural adhoc ranking with kernel pooling. In *SIGIR*.

Shaobin Xu, David Smith, Abigail Mullen, and Ryan Cordell. 2014. Detecting and evaluating local text reuse in social networks. In *Proceedings of the Joint Workshop on Social Dynamics and Personal Attributes in Social Media*, pages 50–57, Baltimore, Maryland. Association for Computational Linguistics.

Dani Yogatama, Michael Heilman, Brendan O'Connor, Chris Dyer, Bryan R. Routledge, and Noah A. Smith. 2011. Predicting a scientific community's response to an article. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 594–604, Edinburgh, Scotland, UK. Association for Computational Linguistics.

A Source Datasets

Below are lists of all source documents in each of the four source datasets. For the sake of space, the names of all 329 works in the LAT datset are detailed here (see latin-works-metadata.jsonl).

The books in KJB are separated into passages by verse. The plays in SHAK are separated into passages by line. The documents in ABL and LAT are separated into passages by using Stanza's (Qi et al., 2020) sentence tokenizer. For the poems in ABL, we use the standard line breaks rather than sentences.

- **KJB, the 66 books of the Old and New Testaments**: acts, amos, chronicles-1, chronicles-2, colossians, corinthians-1, corinthians-2, daniel, deuteronomy, ecclesiastes, ephesians, esther, exodus, ezekiel, ezra, galatians, genesis, habakkuk, haggai, hebrews, hosea, isaiah, james, jeremiah, job, joel, john, john-1, john-2, john-3, jonah, joshua, jude, judges, kings-1, kings-2, lamentations, leviticus, luke, malachi, mark, matthew, micah, nahum, nehemiah, numbers, obadiah, peter-1, peter-2, philemon, philippians, proverbs, psalms, revelation, romans, ruth, samuel-1, samuel-2, song-of-solomon, thessalonians-1, thessalonians-2, timothy-1, timothy-2, titus, zechariah, zephaniah

- **SHAK, all 38 of Shakespeare's plays**: a midsummer nights dream, alls well that ends well, antony and cleopatra, as you like it, coriolanus, cymbeline, hamlet, henry iv pt1, henry iv pt2, henry v, henry vi pt1, henry vi pt2, henry vi pt3, henry viii, julius caesar, king john, king lear, loves labors lost, macbeth, measure for measure, much ado about nothing, othello, pericles, richard ii, richard iii, romeo and juliet, taming of the shrew, the comedy of errors, the merchant of venice, the merry wives of windsor, the tempest, the two gentlemen of verona, the two noble kinsmen, the winters tale, timon of athens, titus andronicus, troilus and cressida, twelfth night

- **ABL: a collection of 70 great works of American and British Literature**[5]: adam bede, adonais, american scholar, an american slave, bartleby, being earnest, bleak house, caleb williams, david copperfield, defence of poetry, dorian gray, dracula, emma, felix holt, frankenstein, french revolution, great expectations, gullivers travels, heart of darkness, heaven and hell, huckleberry finn, innocence and experience, jane eyre, jude the obscure, eve of st agnes, ode on a grecian urn, ode to a nightingale, leaves of grass, little dorrit, lord jim, mansfield park, middlemarch, mlk i have a dream, moby dick, mohicans, mutual friend, northanger abbey, old curiosity shop, oliver twist, paradise lost, persuasion, pilgrims progress, portrait of a lady, pride and prejudice, red badge, return of the native, rights of woman, robinson crusoe, romola, sartor, scarlet letter, self reliance, sense and sensibility, seven gables, slave girl, tale of two cities, tess, the house of usher, the mill, the secret sharer, tom jones, tom sawyer, tristram shandy, uncle tom, us constitution, utopia, vanity fair, walden, washington square, wuthering heights

- **LAT, a collection of 329 great Latin works from the Perseus Digital Library**: see here (latin-works-metadata.jsonl) for a full list of all 329 works.

B Quality of Source-Derived Alignments

We use three different sets of alignments in our work – America's Public Bible (KJB-CA), JSTOR Understanding Series (KJB-JA, SHAK-JA, ABL-JA) and Passim (LAT-EJC). In this section, we discuss the text reuse models used for each collection and the quality of the detected alignments.

- **America's Public Bible (KJB-CA)**: We use alignments provided by Mullen (2016). As described in Mullen (2016, methods section), they first devise a set of features for each (Bible verse, newpaper page) pair to model their similarity: number of overlapping n-grams (with and without TF-IDF weighting), proportion of verse n-grams in newspaper, Wald–Wolfowitz runs test to test whether the positions of the matching tokens in the newspaper page are randomly scattered across the page or clustered together). Next, they sample a set of 1,700 potential (verse, newpaper page) matches and manually label each pair as a true match or not. They split the 1,700 pairs into train, dev and test sets and examine the performance of several models on this subset. They find that a neural network achieves the best performance, with an F1 of 0.92. They then use the trained neural network to label the remaining (verse, newspaper page) pairs in the collection.

- **JSTOR Understanding Series (KJB-JA, SHAK-JA, ABL-JA)**: We use alignments from the JSTOR Understanding Series (JSTOR Labs, 2019). For each source work, the JSTOR Labs team first generates a candidate set of derived JSTOR articles and chapters by performing a full-text search on JSTOR for the work's title, author, and main characters. Next, they extract all text appearing either in block quotes or between single- and double quotation-marks from each derived document. For each quote, they then identify the most similar subsequence in the source work (with

---

[5] The works of American and British Literature are selected as a subset of the American Literature (`https://www.jstor.org/understand/american-literature`) and British Literature (`https://www.jstor.org/understand/british-literature`) collections in the JSTOR Understanding Series. We select roughly the top 50% of works in each collection by total number of aligned derived works and combine them to make our single ABL dataset (70 total works). The dataset contains 20 works of American Literature and 50 works of British Literature (total collection sizes – American Literature (35), British Literature (98)). We do not include the bottom ≈ 50% of works in each collection since they are quoted in very few derived works (≈ 2-150 aligned derived works) and have extremely sparse labels.

the lowest Levenshtein distance). For each matched (source subsequence, derived quote) pair, they calculate a manually-designed confidence score $\in [0, 1]$ based on the match size, match similarity (using Levenshtein distance) and signals indicating the presence of various text matches in surrounding derived text. Finally, examining a subset of their matched pairs, they perform a qualitative evaluation, finding that a confidence score threshold of 0.9 yielded the best balance of false positives to false negatives. We also confirm this finding qualitatively on a sampled set of alignments and use it in our work.

- **Passim (LAT-EJC)**: We use the Passim text alignment software (Smith et al., 2015) to detect quotes of the Latin texts in the JSTOR EJC. Passim uses the Smith–Waterman alignment algorithm to find all pairs of passages within longer documents (source and derived) with substantial alignments. Xu et al. (2014) quantitatively evaluated Passim on English text reuse in English documents, finding that it achieved pseudo-recall of roughly 0.9 and MAP of 0.2 - 0.5. In our work, we use the hyperparameter settings for Passim recommended by Xu et al. (2014). Manually examining our alignments, we find that detecting Latin text-reuse in English documents is easier than finding English text-reuse and confirm that Passim performs reasonably.

C Training lambdaMART: NDCG Formulation

There are two commonly used formulations of NDCG@k. One is the form we use to evaluate our models (§7.1), and the other is the formulation that lambdaMART is trained to optimize. For lambdaMART, $DCG_k = \sum_{i=1}^{k} \frac{2^{\text{label}_i} - 1}{log(i+1)}$. This formulation puts a stronger emphasis on retrieving highly relevant documents. This formulation is reasonable when passage labels are not large (e.g. $\in \{0, 1, 2, 3\}$). However, in our datasets, some passages are quoted by a very large number of derived documents (e.g. 905 for a passage in Romeo & Juliet, 4949 for a passage in the Gospel of Matthew, 1536 for a passage in the U.S. Constitution). Due to these high counts, we opted for the non-exponentiated formulation of NDCG as we thought that it was more representative of model performance – e.g. if the most quoted passage in a document was quoted 500 times and the top ranked passage by the model was quoted 250 times, an NDCG@1 score of 0.5 (using the formulation we use) is much more representative of model performance than one of $\frac{2^{250} - 1}{2^{500} - 1} \approx 0$.

To ensure that lambdaMART is trained to optimize an NDCG objective that is comparable to the one we use for evaluation, we log transform the original count labels of the passages in the training set $\text{label}_i = log_2(\text{count}_i + 1)$ before feeding them into lambdaMART. With this transformation, the numerators of the two DCG formulations are equivalent, but the denominators (the discount) differ slightly. We evaluate using the non-exponentiated version of NDCG (§7.1) with the original, untransformed quote counts on the test and dev sets.

Dataset	Threshold (# quotes)		# Passages	
	Bottom	Top	Bottom	Top
KJB-CA	≤ 1	≥ 26	8,896	6,306
KJB-JA	≤ 0	≥ 3	12,834	7,937
SHAK-JA	≤ 0	≥ 3	52,479	31,868
ABL-JA	≤ 0	≥ 1	354,834	76,746
LAT-EJC	≤ 0	≥ 1	453,086	149,590

Table 8: Thresholds used to separate the highly quoted (top 20%) and least quoted (bottom 20%) passages in each dataset, and the corresponding number of passages in each group. Note, because most datasets contain many unquoted passages, the "bottom" group often contains more than 20% of the passages in the dataset.

D Quote Count Thresholds for Linguistic Analysis

In §5 we explore different linguistic features that affect a passage's quotability. For each dataset, we compare the highly quoted passages (top 20%) to the minimally-quoted ones (bottom 20%), testing if the feature values are significantly different between the two groups. Table 8 shows the thresholds used to identify the highly- and minimally-quoted passages in each dataset and the number of passages in each group. Note, because most datasets contain many unquoted passages, the groups of minimally quoted passages generally contain more than 20% of the total passages in the dataset (e.g. we split the entire dataset for ABL-JA and LAT-EJC with the bottom group consisting of all unquoted passages and the top group containing all passages with at least one quote).

E Features for Feature-based Models

For the feature-based model, each passage is featurized with three sets of features: bag-of-words, "quotability" features, and positional features.

E.1 Bag-of-words

We tokenize passages using Stanza (Qi et al., 2020). We use TFIDF-weighted bag of unigrams and bigrams as text features, keeping the top 100,000 that occur in at least five passages. We evaluate models with and without lemmatization. We use Stanza's 'ewt' package for English and 'perseus' package for Latin[6].

E.2 Quotability Features

Since Bendersky and Smith (2012) also explore applications of quotability identification models to literary works (Project Gutenberg), we primarily use their set of quotability features for our feature-based models:

- Length features: total number of words in the passage, total number of characters in the passage, average number of characters per word, minimum number of characters per word, maximum number of characters per word
- Capitalization: number of capital words in the passage
- Stop words: number of stop words in the passage, passage begins with a stop word
- Punctuation: Five binary features to indicate whether punctuation of type P is present in the passage, P = quotations, parentheses, colon, dash, semi-colon.
- Dialogue words: binary feature if the passage contains at least one common dialog term (English: say, says, said; Latin: list of 145 forms of the words 'dico' and 'loquor')
- Abstract concepts: Number of abstract concepts (e.g., adventure, charity, stupidity) in the passage. Following Bendersky and Smith (2012), we use a list of 176 abstract nouns available at www.englishbanana.com. We do not include this feature for the Latin dataset.
- Quantifiers: Total number of universal quantifiers in the passage (from a list of 20 quantifiers: 'all', 'always', 'each', 'entire', 'ever', 'every', 'everyone', 'full', 'fully', 'never', 'no', 'nobody', 'none', 'nothing', 'nowhere', 'total', 'totally', 'utterly', 'whole', 'wholy'). We do not include this feature for the Latin dataset.
- Emphasis: two binary features for if the passage contains a superlative adjective or a comparative adjective.
- Past participle: binary feature if passage contains a verb in past participle.
- Part of speech: two different sets of counts –
  - Five features for the number of occurrences of nouns, verbs, adjectives, adverbs, or pronouns in the passage.
  - Counts of part of speech triples, e.g. (DET, NOUN, VERB): we create a single feature for each unique part of speech triple and count the number of times it occurs in the passage. This is a slightly adapted version of POS triple feature from Bendersky and Smith (2012), who only include a limited number of POS triples based on calculations on their manually labeled validation set.
- Language Model: a log-likelihood ratio for the passage calculated as the ratio between log-likelihoods from a language model of quotable text and a background language model built on the entire source corpus. For the quotable text language model, we collect approximately 5,200 quotes on more than 200 subjects from the `http://www.quotationspage.com/`. This collection provides a diverse set of high-quality quotations on subjects ranging from Laziness and Genius to Technology and Taxes. This feature is only used for the English datasets.

Finally, we add two additional quotability features from Tan et al. (2018) and one from Danescu-Niculescu-Mizil et al. (2012):

- Personal Pronouns: three features for counts of 1st, 2nd, and 3rd person pronouns (Tan et al., 2018).

---

[6] `https://stanfordnlp.github.io/stanza/available_models.html`

- Generality: number of indefinite articles, only for English dataset (Tan et al., 2018).
- Verb tenses: three features for counts of past, present and future tense verbs (Danescu-Niculescu-Mizil et al., 2012).

E.3 Positional Features

Since, as noted by Tan et al. (2018), a passage's position in a source document is an important feature for determining its quotability, we include the following positional features for each passage (each source dataset has unique positional features). Features such as "Verse index in (chapter, entire book)" with parentheticals indicate that we create multiple features, here one for the verse index in the chapter and the other for the verse index across the entire book.

- **King James Bible**:
  - Verse index in (chapter, entire book)
  - Chapter index in book
  - Relative (0 to 1) verse position in (chapter, book), both raw decimal and one-hot vectors by decile
  - Is (first, last) verse of (book, chapter)
  - Is (first, last) chapter of book
- **Shakespeare**:
  - Line index in (scene, act, play)
  - Scene index in (act, play)
  - Act index in play
  - Relative (0 to 1) line position in (scene, act, play), both raw decimal and one-hot vectors by decile
  - Relative (0 to 1) scene position in (act, play), both raw decimal and one-hot vectors by decile
  - Relative (0 to 1) act position in play, both raw decimal and one-hot vectors by decile
  - Is (first, last) line of (scene, act, play)
  - Is (first, last) scene of (act, play)
  - Is (first, last) act of play
- **American & British Literature**:
  - Sentence index in book
  - Paragraph index in book
  - Chapter index in book
  - Relative (0 to 1) (sentence, paragraph, chapter) position in book, both raw decimal and one-hot vectors by decile.
  - Is (first, last) sentence of (paragraph, chapter, doc)
  - Is (first, last) paragraph of (chapter, book)
  - Is (first, last) chapter of book
- **Latin**: Each document in the Perseus library is split up into sections which roughly break up the text by line/paragraph/sentence etc. depending on the genre and specific work. We use Stanza's Latin models to break up the text into sentences, but use this section boundary information for positional data, as seen below.
  - Sentence index in book
  - (Start, end) section index in book (sentences can span multiple sections)
  - Relative (0 to 1) (sentence, start section, end section) position in book, both raw decimal and one-hot vectors by decile.
  - Is (first, last) sentence of (book, start section, end section)
  - Is (first, last) (start, end) section of book

F Special Positional Tokens for BERT-based Models

For the BERT-based models, we add special tokens to each passage to act as positional indicators so that the model has a better sense of which part of the document it is reading (Alberti et al., 2019). Just as for the feature-based models (§E.3), the positional tokens vary across datasets.

- **King James Bible**:
  - [`Starts-Book`]: added to the beginning of the first verse in the entire document (e.g. Luke 1.1)
  - [`Starts-Chapter`]: added to the beginning of the first verse in each chapter (e.g. Luke 1.1, 2.1)
  - [`Ends-Book`]: added to the end of the last verse in the entire document (e.g. Luke 24.53)
  - [`Ends-Chapter`]: added to the end of the last verse in each chapter (e.g. Luke 1.80, 2.52)

  - [`Book@n`]: added to the beginning of each verse indicating its relative position in the entire document. `n` ranges from 0-9 indicating in which decile in the document the verse occurs.
  - [`Chapter@n`]: added to the beginning of each verse indicating its relative position in the chapter. `n` ranges from 0-9 indicating which decile in the chapter the verse occurs.

- **Shakespeare**:
  - [`Starts-Play`]: added to the beginning of the first line in the entire play
  - [`Starts-Act`]: added to the beginning of the first line in each act
  - [`Starts-Scene`]: added to the beginning of the first line in each scene
  - [`Ends-Play`]: added to the end of the last line in the entire play
  - [`Ends-Act`]: added to the end of the last line in each act
  - [`Ends-Scene`]: added to the end of the last line in each scene
  - [`Play@n`]: added to the beginning of each line indicating its relative position in the entire play. `n` ranges from 0-9 indicating in which decile in the play the verse occurs.
  - [`Act@n`]: added to the beginning of each line indicating its relative position in the act. `n` ranges from 0-9 indicating in which decile in the act the verse occurs.
  - [`Scene@n`]: added to the beginning of each line indicating its relative position in the scene. `n` ranges from 0-9 indicating in which decile in the scene the verse occurs.

- **American & British Literature**:
  - [`Starts-Book`]: added to the beginning of the first sentence in the entire document
  - [`Starts-Chapter`]: added to the beginning of the first sentence in each chapter
  - [`Starts-Paragraph`]: added to the beginning of the first sentence in each paragraph
  - [`Ends-Book`]: added to the end of the last sentence in the entire document
  - [`Ends-Chapter`]: added to the end of the last sentence in each chapter
  - [`Ends-Paragraph`]: added to the ends of the last sentence in each paragraph
  - [`Book@n`]: added to the beginning of each sentence indicating its relative position in the entire document. `n` ranges from 0-9 indicating in which decile in the document the verse occurs.
  - [`Chapter@n`]: added to the beginning of each sentence indicating the relative position of its chapter in the entire document. `n` ranges from 0-9 indicating in which decile in the document the chapter occurs.
  - [`Paragraph@n`]: added to the beginning of each sentence indicating the relative position of its paragraph in the entire document. `n` ranges from 0-9 indicating in which decile in the document the paragraph occurs.

- **Latin**: As noted in §E.3, each document in the Perseus library is split up into sections which roughly break up the text by line/paragraph/sentence etc. depending on the genre and specific work. We use Stanza's Latin models to break up the text into sentences, but use this section boundary information for positional data, as seen below.
  - [`Starts-Book`]: added to the beginning of the first sentence in the entire document
  - [`Ends-Book`]: added to the end of the last sentence in the entire document
  - [`Starts-Section`]: inserted wherever a section (according to Perseus) begins. This can be anywhere in the sentence, not necessarily just in the beginning or end.
  - [`Ends-Section`]: inserted wherever a section (according to Perseus) ends. This can be anywhere in the sentence, not necessarily just in the beginning or end.
  - [`Book@n`]: added to the beginning of each sentence indicating its relative position in the entire document. `n` ranges from 0-9 indicating in which decile in the document the verse occurs.

G Sequence length limits: BERT-based models

For the BERT-based models which operate on single passages (RoBERTA$_{single}$, RoBERTA$_{pair}$, XLM-R$_{single}$, XLM-R$_{pair}$), we cap passages at the following predetermined lengths:

- SHAKE: 100 WordPieces ($>$ 99% of passages)
- ABL: 200 WordPieces ($>$ 99% of passages)
- LAT: 200 WordPieces ($>$ 99% of passages)
- KJB: no cap, all sequences under 113 WordPieces

For the BERT-based models that operate on sequences of multiple sentences (RoBERTA$_{seq}$, XLM-R$_{seq}$), we create an example by greedily adding passages until we hit the cap of 512 WordPieces. If any single sequence is longer than 512 WordPieces, it is capped at that length.

H Searched Hyperparameters & Best Model Configurations

For the feature-based models, we use TFIDF-weighted bag-of-unigrams and bigrams as text features, keeping the top 100k that occur in at least five passages. We evaluate models with and without lemmatization. For poisson regression and SVM^{rank}, we search over regularization parameters in 10^x where $x \in \{$-5, -4, -3, -2, -1, 0, 1$\}$. For SVM^{rank} we also search over the number, n, of negative passages to sample per positive passage, $n \in \{1, 3, 5, 10\}$. We create a separate training example for each (positive, negative) passage pair. For lambdaMART we search over learning rate $\in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$, max tree depth $\in \{4, 6, 8, 10\}$ and number of boosting rounds $\in \{50, 100, 250, 500\}$.

For the neural models, we use Adam (Kingma and Ba, 2014) and search over learning rate $\in \{2e\text{-}5, 3e\text{-}5, 5e\text{-}5\}$ and batch size $\in \{16, 32\}$. We use dropout with probability 0.1. We train single sentence models for up to 10 epochs and sequential sentence models for up to 20, evaluating on the validation set after each epoch and selecting the highest performing model. For the pairwise models, we search over the number, n, of negative passages to sample per positive passage, $n \in \{1, 3, 5, 10\}$ and create a separate training example for each pair.

		Lemmatize	α
KJB-CA	Fold 1	True	1e-3
	Fold 2	True	1e-3
	Fold 3	False	1e-5
	Fold 4	True	1e-3
	Fold 5	True	1e-3
KJB-JA	Fold 1	True	1e-3
	Fold 2	True	1e-4
	Fold 3	True	1e-4
	Fold 4	True	1e-4
	Fold 5	False	1e-4
SHAKE-JA	Fold 1	False	1e-5
	Fold 2	True	1e-4
	Fold 3	True	1.0
	Fold 4	True	0.1
	Fold 5	True	1e-3
ABL-JA	Fold 1	True	1e-4
	Fold 2	True	1e-4
	Fold 3	True	1e-4
	Fold 4	True	1e-3
	Fold 5	True	1e-4
LAT-EJC	Fold 1	False	1e-5
	Fold 2	False	1e-5
	Fold 3	True	1e-5
	Fold 4	False	1e-5
	Fold 5	False	1e-5

Table 9: Hyperparameters of the best performing **poisson regression** models for all five datasets. Since we perform 5-fold cross validation, there are separate best hyperparameters for each fold. Lemmatize: if tokens are lemmatized, α: regularization parameter.

Tables 9, 10, and 11 lists the best hyperparameter configurations for the poisson regression, SVM^{rank}, and lambdaMART models, respectively, across all datasets and folds. We use scikit-learn's (Pedregosa et al., 2011) implementations of poisson regression and SVM^{rank}. We use XGBoost's (Chen and Guestrin, 2016) implementation of lambdaMART.

Table 12 lists the best hyperparameter configurations for the different neural models across all datasets and folds. We train models on a single Nvidia V100 GPU (32GB configuration). We train models with Pytorch (Paszke et al., 2019) and use the pretrained RoBERTA and XLM-R models from the Huggingface Transformers library (Wolf et al., 2019).

		Lemmatize	C	# Negative
KJB-CA	Fold 1	True	0.1	3
	Fold 2	False	1e-2	10
	Fold 3	False	0.1	1
	Fold 4	False	0.1	5
	Fold 5	False	0.1	3
KJB-JA	Fold 1	True	1e-2	5
	Fold 2	True	1e-3	10
	Fold 3	False	1e-3	10
	Fold 4	True	1e-2	3
	Fold 5	True	1e-3	10
SHAKE-JA	Fold 1	True	1e-2	5
	Fold 2	False	1e-2	5
	Fold 3	True	0.1	3
	Fold 4	True	1e-2	5
	Fold 5	False	1e-2	10
ABL-JA	Fold 1	True	1e-2	3
	Fold 2	True	1e-2	5
	Fold 3	True	1e-2	3
	Fold 4	False	1e-2	10
	Fold 5	False	1e-3	10
LAT-EJC	Fold 1	True	1e-2	10
	Fold 2	True	1e-2	3
	Fold 3	True	1e-2	10
	Fold 4	True	1e-3	10
	Fold 5	True	1e-2	10

Table 10: Hyperparameters of the best performing **SVM$^{\mathbf{rank}}$** models for all five datasets. Since we perform 5-fold cross validation, there are separate best hyperparameters for each fold. Lemmatize: if tokens are lemmatized, C: regularization parameter, # Negative: number of negative samples per positive passage.

		Lemmatize	LR	MTD	# BR
KJB-CA	Fold 1	False	0.4	8	250
	Fold 2	False	0.4	4	100
	Fold 3	True	0.5	10	100
	Fold 4	False	0.2	4	250
	Fold 5	True	0.1	10	500
KJB-JA	Fold 1	True	0.4	10	50
	Fold 2	True	0.5	8	50
	Fold 3	False	0.1	8	100
	Fold 4	True	0.3	10	50
	Fold 5	False	0.4	4	50
SHAKE-JA	Fold 1	True	0.01	8	250
	Fold 2	True	0.1	6	50
	Fold 3	False	0.4	6	100
	Fold 4	False	0.4	6	100
	Fold 5	False	0.4	6	100
ABL-JA	Fold 1	True	0.3	4	250
	Fold 2	False	0.2	10	250
	Fold 3	True	0.5	4	100
	Fold 4	False	0.4	10	250
	Fold 5	True	0.3	10	250
LAT-EJC	Fold 1	True	0.4	8	250
	Fold 2	True	0.2	8	500
	Fold 3	True	0.1	8	500
	Fold 4	True	0.2	8	500
	Fold 5	True	0.4	8	250

Table 11: Hyperparameters of the best performing **lambdaMART** models for all five datasets. Since we perform 5-fold cross validation, there are separate best hyperparameters for each fold. Lemmatize: if tokens are lemmatized, LR: learning rate, MTD: maximum tree depth, # BR: number of boosting rounds.

		Fold 1				Fold 2				Fold 3				Fold 4				Fold 5			
		E	LR	BS	Neg	E	LR	BS	Neg	E	LR	BS	Neg	E	LR	BS	Neg	E	LR	BS	Neg
KJB-CA	RoBERTA$_{single}$	9	3e-5	16	–	6	2e-5	32	–	8	2e-5	16	–	8	2e-5	32	–	8	3e-5	32	–
	RoBERTA$_{pair}$	9	2e-5	16	1	10	3e-5	32	3	10	2e-5	16	10	7	2e-5	32	1	9	2e-5	32	10
	RoBERTA$_{seq}$	6	3e-5	16	–	14	3e-5	16	–	20	2e-5	16	–	17	3e-5	32	–	12	3e-5	16	–
KJB-JA	RoBERTA$_{single}$	9	3e-5	32	–	10	5e-5	32	–	4	2e-5	32	–	10	3e-5	16	–	3	3e-5	16	–
	RoBERTA$_{pair}$	7	2e-5	16	10	7	3e-5	32	10	9	2e-5	16	10	3	2e-5	32	3	8	2e-5	32	10
	RoBERTA$_{seq}$	16	5e-5	32	–	11	5e-5	16	–	20	5e-5	32	–	14	5e-5	32	–	13	3e-5	16	–
SHAKE-JA	RoBERTA$_{single}$	6	2e-5	16	–	4	2e-5	32	–	2	2e-5	32	–	3	3e-5	32	–	7	2e-5	32	–
	RoBERTA$_{pair}$	8	2e-5	32	5	4	2e-5	16	3	7	2e-5	16	5	5	3e-5	16	1	9	3e-5	32	10
	RoBERTA$_{seq}$	2	5e-5	32	–	8	5e-5	16	–	14	5e-5	32	–	7	5e-5	16	–	11	5e-5	32	–
ABL-JA	RoBERTA$_{single}$	2	2e-5	16	–	1	2e-5	16	–	3	3e-5	32	–	10	2e-5	16	–	2	2e-5	16	–
	RoBERTA$_{pair}$	7	3e-5	32	1	5	5e-5	32	5	2	5e-5	32	10	4	2e-5	16	3	1	5e-5	32	10
	RoBERTA$_{seq}$	3	3e-5	32	–	3	2e-5	32	–	5	5e-5	32	–	2	2e-5	32	–	4	5e-5	32	–
LAT-EJC	XLM-R$_{single}$	6	2e-5	32	–	9	3e-5	32	–	9	2e-5	32	–	8	3e-5	32	–	4	2e-5	16	–
	XLM-R$_{pair}$	9	2e-5	32	3	3	3e-5	32	10	3	2e-5	32	3	4	3e-5	32	1	3	2e-5	32	5
	XLM-R$_{seq}$	10	3e-5	32	–	11	2e-5	16	–	10	2e-5	16	–	12	3e-5	32	–	16	2e-5	16	–

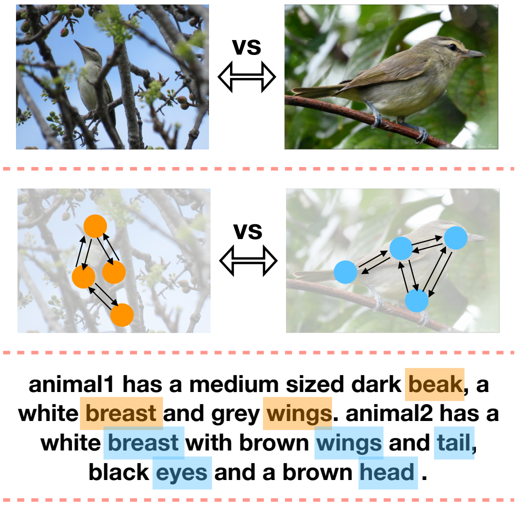
Table 12: Hyperparameters of the best performing **neural models** for all five datasets. Since we perform 5-fold cross validation, there are separate best hyperparameters for each fold. E: number of epochs, LR: learning rate, BS: batch size, Neg: number of negative passages sampled per positive passage (only for pairwise models).

L2C: Describing Visual Differences Needs Semantic Understanding of Individuals

An Yan[♦], **Xin Eric Wang** [♠], **Tsu-Jui Fu**[♣], **William Yang Wang**[♣]
[♦]UC San Diego, [♠]UC Santa Cruz, [♣]UC Santa Barbara
`ayan@ucsd.edu, xwang366@ucsc.edu, {tsu-juifu,william}@cs.ucsb.edu`

Abstract

Recent advances in language and vision push forward the research of captioning a single image to describing visual differences between image pairs. Suppose there are two images, I_1 and I_2, and the task is to generate a description $W_{1,2}$ comparing them, existing methods directly model $\langle I_1, I_2 \rangle \rightarrow W_{1,2}$ mapping without the semantic understanding of individuals. In this paper, we introduce a Learning-to-Compare (L2C) model, which learns to understand the semantic structures of these two images and compare them while learning to describe each one. We demonstrate that L2C benefits from a comparison between explicit semantic representations and single-image captions, and generalizes better on the new testing image pairs. It outperforms the baseline on both automatic evaluation and human evaluation for the Birds-to-Words dataset.



animal1 has a medium sized dark beak, a white breast and grey wings. animal2 has a white breast with brown wings and tail, black eyes and a brown head .

Figure 1: Overview of the visual comparison task and our motivation. The key is to understand both images and compare them. Explicit semantic structures can be compared between images and used to generate comparative descriptions aligned to the image saliency.

1 Introduction

The task of generating textual descriptions of images tests a machine's ability to understand visual data and interpret it in natural language. It is a fundamental research problem lying at the intersection of natural language processing, computer vision, and cognitive science. For example, single-image captioning (Farhadi et al., 2010; Kulkarni et al., 2013; Vinyals et al., 2015; Xu et al., 2015) has been extensively studied.

Recently, a new intriguing task, visual comparison, along with several benchmarks (Jhamtani and Berg-Kirkpatrick, 2018; Tan et al., 2019; Park et al., 2019; Forbes et al., 2019) has drawn increasing attention in the community. To complete the task and generate comparative descriptions, a machine should understand the visual differences between a pair of images (see Figure 1). Previous methods (Jhamtani and Berg-Kirkpatrick, 2018) often consider the pair of pre-trained visual features such as the ResNet features (He et al., 2016) as a whole, and build end-to-end neural networks to predict the description of visual comparison directly. In contrast, humans can easily reason about the visual components of a single image and describe the visual differences between two images based on their semantic understanding of each one. Humans do not need to look at thousands of image pairs to describe the difference of new image pairs, as they can leverage their understanding of single images for visual comparison.

Therefore, we believe that visual differences should be learned by understanding and comparing every single image's semantic representation. A most recent work (Zhang et al., 2020) conceptually supports this argument, where they show that low-level ResNet visual features lead to poor generalization in vision-and-language navigation, and high-level semantic segmentation helps the agent

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2315–2320
April 19 - 23, 2021.

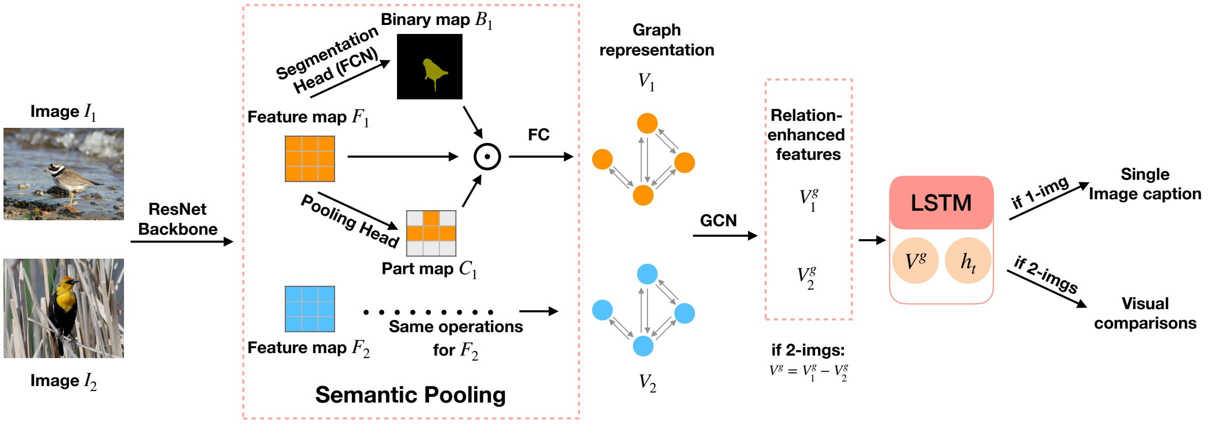


Figure 2: Our L2C model. It consists of a segmentation encoder, a graph convolutional module, and an LSTM decoder with an auxiliary loss for single-image captioning. Details are in Section 2.

generalize to unseen scenarios.

Motivated by humans, we propose a Learning-to-Compare (L2C) method that focuses on reasoning about the semantic structures of individual images and then compares the difference of the image pair. Our contributions are three-fold:

- We construct a structured image representation by leveraging image segmentation with a novel semantic pooling, and use graph convolutional networks to perform reasoning on these learned representations.
- We utilize single-image captioning data to boost semantic understanding of each image with its language counterpart.
- Our L2C model outperforms the baseline on both automatic evaluation and human evaluation, and generalizes better on the testing image pairs.

2 L2C Model

We present a novel framework in Figure 2, which consists of three main components. First, a *segmentation encoder* is used to extract structured visual features with strong semantic priors. Then, a *graph convolutional module* performs reasoning on the learned semantic representations. To enhance the understanding of each image, we introduce a *single-image captioning auxiliary loss* to associate the single-image graph representation with the semantic meaning conveyed by its language counterpart. Finally, a decoder generates the visual descriptions comparing two images based on differences in graph representations. All parameters are shared for both images and both tasks.

2.1 Semantic Representation Construction

To extract semantic visual features, we utilize pretrained fully convolutional networks (FCN) (Long et al., 2015) with ResNet-101 as the backbone. An image $\mathcal{I}$ is fed into the ResNet backbone to produce a feature map $\mathcal{F} \in \mathbb{R}^{D\times H\times W}$, which is then forwarded into an FCN head that generates a binary segmentation mask B for the bird class. However, the shapes of these masks are variable for each image, and simple pooling methods such as average pooling and max pooling would lose some information of spatial relations within the mask.

To address this issue and enable efficient aggregation over the area of interest (the masked area), we add a module after the ResNet to cluster each pixel within the mask into K classes. Feature map $\mathcal{F}$ is forwarded through this pooling module to obtain a confidence map $\mathcal{C} \in \mathbb{R}^{K\times H\times W}$, whose entry at each pixel is a K-dimensional vector that represents the probability distribution of K classes.

Then a set of nodes $V = \{v_1, ..., v_K\}, v_k \in \mathbb{R}^D$ is constructed as following:

$$v_k = \sum_{i,j} \mathcal{F} \odot \mathcal{B} \odot \mathcal{C}_k \tag{1}$$

where i=1, ...H, j=1, ..., W,, $\mathcal{C}_k$ is the k-th probability map and $\odot$ denotes element-wise multiplication.

To enforce local smoothness, i.e., pixels in a neighborhood are more likely belong to one class, we employ total variation norm as a regularization term:

$$\mathcal{L}_{TV} = \sum_{i,j} |C_{i+1,j} - Ci,j| + |C_{i,j+1} - Ci,j| \tag{2}$$

2.2 Comparative Relational Reasoning

Inspired by recent advances in visual reasoning and graph neural networks (Chen et al., 2018; Li et al., 2019), we introduce a relational reasoning module to enhance the semantic representation of each image. A fully-connected visual semantic graph $G = (V, E)$ is built, where V is the set of nodes, each containing a regional feature, and E is constructed by measuring the pairwise affinity between each two nodes v_i, v_j in a latent space.

$$A(v_i, v_j) = (W_i v_i)^T (W_j v_j) \tag{3}$$

where W_i, W_j are learnable matrices, and A is the constructed adjacency matrix.

We apply Graph Convolutional Networks (GCN) (Kipf and Welling, 2016) to perform reasoning on the graph. After the GCN module, the output $V^o = \{v_1^o, ..., v_K^o\}, v_k^o \in \mathbb{R}^D$ will be a relationship enhanced representation of a bird. For the visual comparison task, we compute the difference of each two visual nodes from two sets, denoted as $V_{diff}^g = \{v_{diff,1}^o, ..., v_{diff,K}^o\}, v_{diff,k}^o = v_{k,1}^o - v_{k,2}^o \in \mathbb{R}^D$.

2.3 Learning to Compare while Learning to Describe

After obtaining relation-enhanced semantic features, we use a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) to generate captions. As discussed in Section 1, semantic understanding of each image is key to solve the task. However, there is no single dataset that contains both visual comparison and single-image annotations. Hence, we leverage two datasets from similar domains to facilitate training. One is for visual comparison, and the other is for single-image captioning. Alternate training is utilized such that for each iteration, two mini-batches of images from both datasets are sampled independently and fed into the encoder to obtain visual representations V^o (for single-image captioning) or V_{diff}^o (for visual comparison).

The LSTM takes V^o or V_{diff}^o with previous output word embedding y_{t-1} as input, updates the hidden state from h_{t-1} to h_t, and predicts the word for the next time step. The generation process of bi-image comparison is learned by maximizing the log-likelihood of the predicted output sentence. The loss function is defined as follows:

$$\mathcal{L}_{diff} = -\sum_t \log P(y_t | y_{1:t-1}, V_{diff}^o) \tag{4}$$

Similar loss is applied for learning single-image captioning:

$$\mathcal{L}_{single} = -\sum_t \log P(y_t | y_{1:t-1}, V^o) \tag{5}$$

Overall, the model is optimized with a mixture of cross-entropy losses and total variation loss:

$$\mathcal{L}_{loss} = \mathcal{L}_{diff} + \mathcal{L}_{single} + \lambda \mathcal{L}_{TV} \tag{6}$$

where λ is an adaptive factor that weighs the total variation loss.

3 Experiments

3.1 Experimental Setup

Datasets The Birds-to-Words (B2W) has 3347 image pairs, and each has around 5 descriptions of visual difference. This leads to 12890/1556/1604 captions for train/val/test splits. Since B2W contains only visual comparisons, We use the CUB-200-2011 dataset (CUB) (Wah et al., 2011), which consists of single-image captions as an auxiliary to facilitate the training of semantic understanding. CUB has 8855/2933 images of birds for train/val splits, and each image has 10 captions.

Evaluation Metrics Performances are first evaluated on three automatic metrics[1]: BLEU-4 (Papineni et al., 2002), ROUGE-L (Lin, 2004), and CIDEr-D (Vedantam et al., 2015). Each generated description is compared to all five reference paragraphs. Note for this particular task, researchers observe that CIDEr-D is susceptible to common patterns in the data (See Table 1 for proof), and ROUGE-L is anecdotally correlated with higher-quality descriptions (which is noted in previous work (Forbes et al., 2019)). Hence we consider ROUGE-L as the major metric for evaluating performances. We then perform a human evaluation to further verify the performance.

Implementation Details We use Adam as the optimizer with an initial learning rate set to 1e-4. The pooling module to generate K classes is composed of two convolutional layers and batch normalization, with kernel sizes 3 and 1 respectively. We set K to 9 and λ to 1. The dimension of graph representations is 512. The hidden size of the decoder is also 512. The batch sizes of B2W and CUB are 16 and 128. Following the advice from (Forbes et al., 2019), we report the results

---

[1] `https://www.nltk.org`

	Validation			Test		
Model	BLEU-4 ↑	ROUGE-L ↑	CIDEr-D ↑	BLEU-4 ↑	ROUGE-L ↑	CIDEr-D ↑
Most Frequent	20.0	31.0	**42.0**	20.0	30.0	**43.0**
Text-Only	14.0	36.0	5.0	14.0	36.0	7.0
Neural Naturalist	24.0	46.0	28.0	22.0	43.0	25.0
CNN+LSTM	25.1	43.4	10.2	24.9	43.2	9.9
L2C [B2W]	31.9	45.7	15.2	31.3	45.3	15.1
L2C [CUB+B2W]	**32.3**	**46.2**	16.4	**31.8**	**45.6**	16.3
Human	26.0	47.0	39.0	27.0	47.0	42.0

Table 1: Results for visual comparison on the Birds-to-Words dataset (Forbes et al., 2019). *Most Frequent* produces only the most observed description in the dataset: "the two animals appear to be exactly the same". *Text-Only* samples captions from the training data according to their empirical distribution. *Neural Naturalist* is a transformer model in Forbes et al. (2019). *CNN+LSTM* is a commonly-used CNN encoder and LSTM decoder model.

Choice (%)	L2C	CNN+LSTM	Tie
Score	**50.8**	39.4	9.8

Table 2: Human evaluation results. We present workers with two generations by L2C and CNN+LSTM for each image pair and let them choose the better one.

using models with the highest ROUGE-L on the validation set, since it could correlate better with high-quality outputs for this task.

3.2 Automatic Evaluation

As shown in Table 1, first, L2C[B2W] (training with visual comparison task only) outperforms baseline methods on BLEU-4 and ROUGE-L. Previous approaches and architectures failed to bring superior results by directly modeling the visual relationship on ResNet features. Second, joint learning with a single-image caption L2C[B2W+CUB] can help improve the ability of semantic understanding, thus, the overall performance of the model. Finally, our method also has a smaller gap between validation and test set compared to *neural naturalist*, indicating its potential capability to generalize for unseen samples.

3.3 Human Evaluation

To fully evaluate our model, we conduct a pairwise human evaluation on Amazon Mechanical Turk with 100 image pairs randomly sampled from the test set, each sample was assigned to 5 workers to eliminate human variance. Following Wang et al. (2018), for each image pair, workers are presented with two paragraphs from different models and asked to choose the better one based on text quality[2]. As shown in Table 2, L2C outperforms CNN+LSTM, which is consistent with automatic metrics.

	Validation		
Model	BLEU-4 ↑	ROUGE-L ↑	CIDEr-D ↑
L2C	**31.9**	**45.7**	**15.2**
− Semantic Pooling	24.5	43.2	7.2
− TV Loss	29.3	44.8	13.6
− GCN	30.2	43.5	10.7

Table 3: Ablation study on the B2W dataset. We individually remove Semantic Pooling, total variation (TV) loss, and GCN to test their effects.

3.4 Ablation Studies

Effect of Individual Components We perform ablation studies to show the effectiveness of semantic pooling, total variance loss, and graph reasoning, as shown in Table 3. First, without semantic pooling, the model degrades to average pooling, and results show that semantic pooling can better preserve the spatial relations for the visual representations. Moreover, the total variation loss can further boost the performance by injecting the prior local smoothness. Finally, the results without GCN are lower than the full L2C model, indicating graph convolutions can efficiently modeling relations among visual regions.

Sensitivity Test We analyze model performance under a varying number of K (K is the number of classes for confidence map $\mathcal{C}$), as shown in Figure 3. Empirically, we found the results are comparable when K is small.

---

[2] We instruct the annotators to consider two perspectives, relevance (the text describes the context of two images) and expressiveness (grammatically and semantically correct).

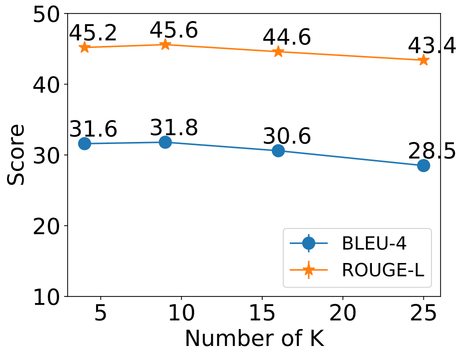


Figure 3: Sensitivity test on number of K chosen.

4 Conclusion

In this paper, we present a learning-to-compare framework for generating visual comparisons. Our segmentation encoder with semantic pooling and graph reasoning could construct structured image representations. We also show that learning to describe visual differences benefits from understanding the semantics of each image.

Acknowledgments

The research was partly sponsored by the U.S. Army Research Office and was accomplished under Contract Number W911NF19-D-0001 for the Institute for Collaborative Biotechnologies. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. 2018. Iterative visual reasoning beyond convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7239–7248.

Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: Generating sentences from images. In *European conference on computer vision*, pages 15–29. Springer.

Maxwell Forbes, Christine Kaeser-Chen, Piyush Sharma, and Serge Belongie. 2019. Neural naturalist: Generating fine-grained image comparisons. *arXiv preprint arXiv:1909.04101*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Harsh Jhamtani and Taylor Berg-Kirkpatrick. 2018. Learning to describe differences between pairs of similar images. *arXiv preprint arXiv:1808.10584*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2013. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2891–2903.

Kunpeng Li, Yulun Zhang, Kai Li, Yuanyuan Li, and Yun Fu. 2019. Visual semantic reasoning for image-text matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4654–4662.

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

Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

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

Dong Huk Park, Trevor Darrell, and Anna Rohrbach. 2019. Robust change captioning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4624–4633.

Hao Tan, Franck Dernoncourt, Zhe Lin, Trung Bui, and Mohit Bansal. 2019. Expressing visual relationships via language. *arXiv preprint arXiv:1906.07689*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. The caltech-ucsd birds-200-2011 dataset.

Xin Wang, Wenhu Chen, Yuan-Fang Wang, and William Yang Wang. 2018. No metrics are perfect: Adversarial reward learning for visual storytelling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 899–909.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Yubo Zhang, Hao Tan, and Mohit Bansal. 2020. Diagnosing the environment bias in vision-and-language navigation. *arXiv preprint arXiv:2005.03086*.

VoiSeR: A New Benchmark for Voice-Based Search Refinement

Simone Filice, Giuseppe Castellucci, Marcus Collins, Eugene Agichtein, Oleg Rokhlenko
Amazon
Seattle, USA
{filicesf,giusecas,collmr,eugeneag,olegro}@amazon.com

Abstract

Voice assistants, e.g., Alexa or Google Assistant, have dramatically improved in recent years. Supporting voice-based search, exploration, and refinement are fundamental tasks for voice assistants, and remain an open challenge. For example, when using voice to search an online shopping site, a user often needs to refine their search by some aspect or facet. This common user intent is usually available through a "filter-by" interface on online shopping websites, but is challenging to support naturally via voice, as the intent of refinements must be interpreted in the context of the original search, the initial results, and the available product catalogue facets. To our knowledge, no benchmark dataset exists for training or validating such contextual search understanding models. To bridge this gap, we introduce the first large-scale dataset of voice-based search refinements, VoiSeR, consisting of about 10,000 search refinement utterances, collected using a novel crowdsourcing task. These utterances are intended to refine a previous search, with respect to a search facet or attribute (e.g., brand, color, review rating, etc.), and are manually annotated with the specific intent. This paper reports qualitative and empirical insights into the most common and challenging types of refinements that a voice-based conversational search system must support. As we show, VoiSeR can support research in conversational query understanding, contextual user intent prediction, and other conversational search topics to facilitate the development of conversational search systems.

1 Introduction

Modern voice assistants, such as Amazon Alexa or Apple Siri, make use of Natural Language Understanding (NLU) techniques to perform several tasks. Some of the most popular functions offered by these systems are based on voice-search: millions use voice assistants to access information or search for music, products or local restaurants and stores. However, search experience with a voice assistant remains limited. The current generation of these systems mostly supports single-turn interactions, and does not naturally support more complex search needs, which often require *refinements* to narrow, broaden or change the initial search. Supporting refinement is a fundamental aspect of search systems, and it is done in a variety of ways in Web-based user interfaces, e.g., through query suggestion or explicit facets navigation or filtering. For example, in an e-Commerce search, a user may want to refine their search with respect to some facet or attribute (e.g., brand or price); this critical functionality is supported on most e-Commerce websites. However, this kind of interaction is challenging to support via voice-based dialogue interfaces, as interpreting such refinements requires modeling the original search intent, the initial results, and the available result facets.

To the best of our knowledge, no large scale dataset exists for training and validating NLU models for multi-turn voice-based search. To bridge this gap, we present a new Voice-based Search Refinement dataset, VoiSeR[1] to enable research into contextual understanding of multi-turn voice-based search. The VoiSeR dataset contains 9,810 utterances of voice refinements in the e-Commerce domain, paired with contextual information. Specifically, our dataset includes refinement utterances intended to filter results from a previous search with respect to some facet or attribute (e.g., brand, color, review rating, etc.).

The dataset was collected through crowdsourcing via Amazon Mechanical Turk, between February and June 2020 in the US and India. We designed the task to minimize any bias towards partic-

---

[1] The dataset is available for download at https://registry.opendata.aws/amazon-conversational-product-search/.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2321–2329
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

ular expressions or terminology which may not be natural to users. To achieve this goal, we provided clear and concise instructions; we intentionally did not provide examples to avoid biasing participants towards using specific linguistic expressions (see Figure 1 for an example). As a result, the dataset provides a diverse and natural representation of how users express the intended refinements during product search.

We annotated the dataset to highlight some important aspects characterizing a voice refinement of product search. In particular, we annotated (*i*) the products and attributes mentioned in each utterance, if present; (*ii*) the specific refinement intent of each utterance (e.g., refinement by exact attribute value).

In addition to the new VoiSeR benchmark dataset, our contributions include (*i*) an analysis of the data, where we highlight some linguistic aspects characterizing how people express the refinement intent, and (*ii*) an empirical investigation to demonstrates that VoiSeR can be successfully used to bootstrap NLU models for handling voice-based search refinements. Furthermore, we show that contextual information is beneficial for such NLU tasks.

Next, §2 provides details about the data collection and annotation. §3 provides a detailed analysis of the dataset, while §4 reports the empirical investigation. In §5, we discuss the related works. Finally, §6 discusses our conclusions.

2 VoiSeR Data Collection

In order to collect a large number of voice search refinements from multiple participants, we designed a crowdsourcing task on Amazon Mechanical Turk[2]. In §2.1, we provide details of the crowdsourcing experiment to collect refinement utterances. These utterances were annotated for intent and relevance, as described in §2.2.

2.1 Crowdsourcing the Voice Refinement Data

The design of the task was intended to make it both easy for the participants (i.e., Amazon Mechanical Turk Worker) and as realistic as possible, to provide valid linguistic expressions of voice refinements. Thus, we tried to reproduce a real "customer journey" of product searches and refinements. With this idea in mind, we designed the Mechanical Turk task depicted in Figure 1. The crowdsourcing interface shows to the Worker:

- An initial set of products, i.e., up to five products in the top part of the image.
- A target set of products, i.e., up to five products in the central part of the image.
- A visual intent indicator, i.e., an image describing the attribute type the worker should focus on when expressing the refinement. In Figure 1, this is the 5-star symbol on the central-left part; it represents the *review-rating* attribute.

The participant is asked to imagine they are searching for products and that her search led to the initial set of results. We ask the participant to record a voice utterance, modifying the search to achieve the target product set, cued by the provided visual intent indicator. In the example in Figure 1, the participant should attempt to refine the search by *review rating* as indicated in the intent indicator: compared to the products in the initial set, the products in the target set have all 4 stars or more, therefore we expect the Worker to say something like "*Show me only products having 4 stars and above*" or "*Earphones with at least four stars*".

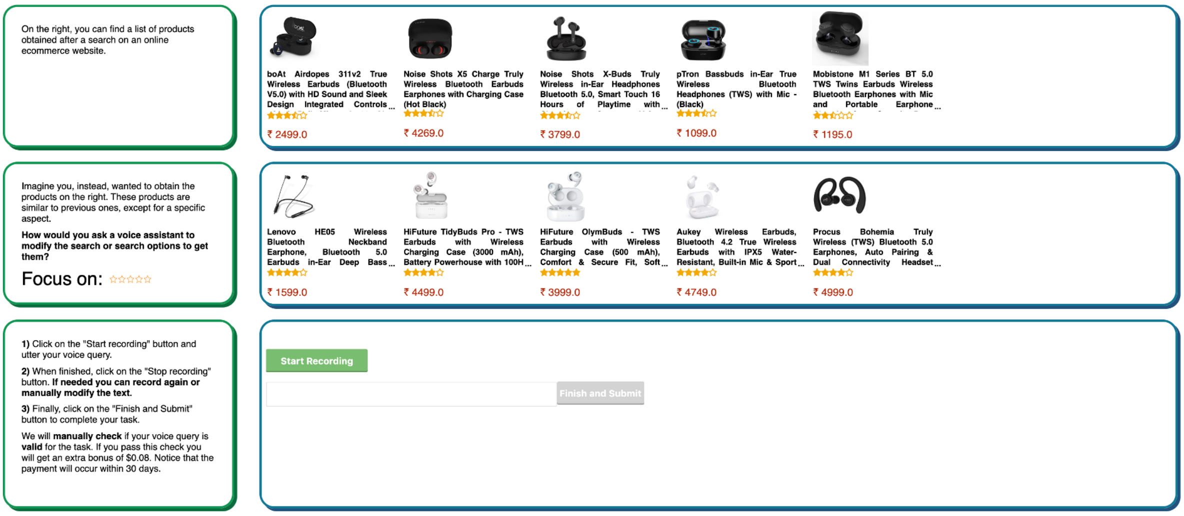
An Automatic Speech Recognition (ASR) system (we adopted Amazon AWS Transcribe[3]) processes the utterance in real time and its transcription is shown to the Worker. Since ASR errors can occur, before completing the task, we ask the Worker to check the ASR transcription. In case of errors, the Worker can record a new sentence or manually correct the ASR transcript. We record the original ASR transcript and any manual correction, if one is made.

To automatically generate the many examples to annotate, we used the Amazon.in product search engine: starting from a random product search, we collected the initially retrieved products, as well as those returned after the application of a filter. The type of the activated filter dictates the visual intent indicator shown to the Worker, while the products shown in the initial and target sets are a subset of those retrieved by Amazon.in before and after the filter application, respectively. To emphasize the difference between the initial set and the target set, we select the products so that (*i*) no product appears in both sets and (*ii*) the products in the

---

[2] `https://www.mturk.com/`

[3] `https://aws.amazon.com/transcribe/`


On the right, you can find a list of products obtained after a search on an online ecommerce website.

boAt Airdopes 311v2 True Wireless Earbuds (Bluetooth V5.0) with HD Sound and Sleek Design Integrated Controls ...
₹ 2499.0

Noise Shots X5 Charge Truly Wireless Bluetooth Earbuds Earphones with Charging Case (Hot Black)
₹ 4269.0

Noise Shots X-Buds Truly Wireless in-Ear Headphones Bluetooth 5.0, Smart Touch 16 Hours of Playtime with ...
₹ 3799.0

pTron Bassbuds in-Ear True Wireless Bluetooth Headphones (TWS) with Mic - (Black)
₹ 1099.0

Mobistone M1 Series BT 5.0 TWS Twins Earbuds Wireless Bluetooth Earphones with Mic and Portable Earphone ...
₹ 1195.0

Imagine you, instead, wanted to obtain the products on the right. These products are similar to previous ones, except for a specific aspect.

How would you ask a voice assistant to modify the search or search options to get them?

Focus on: ☆☆☆☆☆

Lenovo HE05 Wireless Bluetooth Neckband Earphone, Bluetooth 5.0 Earbuds in-Ear Deep Bass ...
₹ 1599.0

HiFuture TidyBuds Pro - TWS Earbuds with Wireless Charging Case (3000 mAh), Battery Powerhouse with 100H ...
₹ 4499.0

HiFuture OlymBuds - TWS Earbuds with Wireless Charging Case (500 mAh), Comfort & Secure Fit, Soft ...
₹ 3999.0

Aukey Wireless Earbuds, Bluetooth 4.2 True Wireless Earbuds with IPX5 Water-Resistant, Built-in Mic & Sport ...
₹ 4749.0

Procus Bohemia Truly Wireless (TWS) Bluetooth 5.0 Earphones, Auto Pairing & Dual Connectivity Headset ...
₹ 4999.0

1) Click on the "Start recording" button and utter your voice query.

2) When finished, click on the "Stop recording" button. **If needed you can record again or manually modify the text.**

3) Finally, click on the "Finish and Submit" button to complete your task.

We will **manually check** if your voice query is **valid** for the task. If you pass this check you will get an extra bonus of $0.08. Notice that the payment will occur within 30 days.

Figure 1: MTurk interface for voice refinements collection. The workers see source and target sets of products, differing on a specific attribute (e.g., review rating), and record a voice refinement by the suggested target attribute.

initial set do not satisfy the activated filter. For instance, the task shown in Figure 1 replicates a search session where the *4 stars & up* filter on the review rating was activated; therefore, for the initial set we selected all products having less than 4 stars, vs. more than 4 stars in the target sets.

We kept the task instructions as simple as possible in order to not introduce linguistic biases: the complete instructions to the Workers are those shown at the left Figure 1. We intentionally did not provide any example on how to complete a task. In relatively few cases this created some misunderstanding with the Workers that failed to provide valid voice refinements. On the other hand, our choice prevented any possible bias towards using some specific linguistic expressions. To further minimize terminology bias, we used visual intent indicators to suggest the attribute type to refine on. For instance, we used the image of a color palette to represent the *color* attribute, and an image of banknotes to suggest *price*.

In a preliminary experiment, we did not show the intent indicator, but in many cases the difference between the initial set and the target set was not obvious, so that the Workers ended up focusing on irrelevant details. As a consequence, the utterances that were collected in that setting were over-specific, and often the Workers simply read parts of the target product titles. Based on this experiment, the full dataset was collected using the intent indicator condition described above.

In e-commerce websites selling a wide range of products, there are typically many possible attribute types customers can filter on. We decided to collect data about some of the most popular and generally applicable ones, namely *brand, color, discount, material, price* and *review rating*.

2.2 Data Annotation

After collecting the data, we asked domain experts to annotate them with respect to three different tasks.

Voice Refinement Validity: Since crowdsourcing data can be noisy, we first design a preliminary annotation task to validate each single utterance collected on Mechanical Turk. In particular, we showed to the annotators the Mechanical Turk task associated to each sentence and asked them to state whether the utterance correctly refines the product search on the target attribute. We also asked the annotators to report whether ASR errors (or typos in case the Worker manually corrected the ASR transcription) occur in the utterance.

Attribute and Product Extraction: We asked the annotators to mark product and attribute mentions in the utterance, for example in the utterance *"Show me only red t-shirts," red* is the attribute and *t-shirt* is the product. Given the Mechanical Turk task design discussed in §2.1, the collected utterances are supposed to refine on a single attribute. However, we noticed that some utterances contain multiple attribute mentions. For instance, *Nike* and *red* in the sentence *"Show me only red Nike t-shirts"*. The annotators are required to extract all individual attribute mentions within the utterance and not only the attribute mention referring to the target attribute type (the one shown as intent indicator to the Worker).

Refinement Intent Classification: This task consists of indicating how to change the search query based on the attribute mentioned in the utterance. We asked the annotators to indicate whether the provided refinement belongs to one of the following types:

- `EXACT`: the customer asks to select products having a specific value for the attribute, e.g., *"show me only purple"*.
- `EXCLUDE`: the customer asks to exclude products having a specific value for the attribute, e.g., *"exclude the purple ones"*.
- `RANGE`: the customer asks to select products having attribute values in a closed interval, e.g., *"Price between 200 and 300"*.
- `GREATER`: the customer asks to select products with attribute value higher than a given value, e.g., *"Show 4 stars and up"*, or *"Exclude the products with less than four stars"*.
- `LOWER`: the customer asks to select products with attribute value lower than a given value, e.g., *"Price less than 100"*, or *"Exclude the ones more expensive than 100"*.
- `OTHER`: utterance not falling in the above categories, e.g., *"Show me a different color"*, or *'Select top ratings"*.

Each example in our data is annotated by a single domain expert, since we observed a very high annotation quality in a preliminary annotation phase where multiple annotators annotated the same instances. We registered an almost perfect agreement in all tasks: Cohen's Kappa 0.914 for the Voice Refinement Validity task, Cohen's Kappa 0.960 for the Attribute and Product Extraction task, and Cohen's Kappa 0.859 for the Refinement Intent Classification task.

3 Voice Refinements Dataset

In this section, we provide the analysis of the data collected through the crowdsourcing experiment. First, we describe how we conducted the Mechanical Turk experiment and the statistics of the collected data in §3.1; then, we discuss some of the linguistic properties emerging in the context of voice refinements in §3.2.

3.1 Crowdsourcing the Voice-Based Search Refinement Data

The data was crowdsourced using the Amazon Mechanical Turk platform, from workers in the U.S. and India, between February and June 2020. Both Indian and U.S. Workers were asked to provide English voice refinements with respect to the tasks shown.

Region	**Total**	**Valid (%)**
US	2,716	2,475 (91.12%)
IN	10,776	7,335 (68.06%)
Total	13,492	9,810 (72.71%)

Table 1: Data statistics per region.

As reported in Table 1, we collected 13,492 utterances, and 9,810 (i.e., ∼72%) were considered valid refinements in the subsequent annotation phase, while the rest are not refinements or contain ASR errors. In total 385 workers participated in the data collection, each one producing 35 utterances on average[4]. This results in a great variability in the collected data, as each worker can provide different linguistic expressions for a voice refinements.

Table 2 reports the distributions of the valid utterances with respect to the attribute types and the refinement types. The utterances are mostly evenly distributed with respect to the 7 attributes. *Brand* and *material* are the most and least represented attributes. The majority of utterances are of type `EXACT`, i.e., they ask to filter the product search on a specific attribute value. Unsurprisingly, the `EXCLUDE` type is extremely rare: our Mechanical Turk task did not encourage this refinement type.

As expected, the majority of brand-related utterances are of type `EXACT`. Also, the number of utterances of the price and rating related attributes are mainly `LOWER` and `GREATER`, respectively. This is intuitive considering that usually users look for less expensive and better rated products (for example, "*Show me items that are rated four stars or better*"). Notice that price (and also discount) utterances include a good number of `RANGE`, `EXACT`, as for example, "*Womens bags in the range of 1000 to 3300*".

Most of the `OTHER` bucket are (*i*) utterances whose intent is to sort products with respect to an attribute type like price or review rating, e.g., "*Sort by most rated laptops*", (*ii*) utterances where no specific attribute value is mentioned, e.g., "*Show*

---

[4] Each worker was allowed to do at most 100 tasks.

	EXACT	EXCLUDE	GREATER	LOWER	RANGE	OTHER	Total
brand	2,951	1	1	0	1	26	2,980
color	1,321	1	0	0	0	201	1,523
discount	480	0	234	80	37	478	1,309
material	719	2	0	0	0	19	750
price	119	0	176	799	134	0	1,463
rating	401	0	852	7	18	517	1,795
Total	5,991	4	1,263	886	1,477	189	9,810

Table 2: Distribution of the valid utterances with respect to the attribute types and the refinement types.

items with high ratings". Other common cases are utterances with comparatives e.g., "*Show me less expensive earphones*" or superlatives e.g., "*Show me the highest rated silk sets*".

We noticed that, despite the fact that our Mechanical Turk tasks focused on a single attribute, in about 27% of the cases (2,668 utterances) Workers provided utterances containing multiple attributes. For example, the utterance "*Price should be less than 700 with discount*" contains both the target attribute (discount) and a specification of the price.

Finally, ~80% of the utterances have a product mention, e.g., "*Show me a Vega brand hair curler*". The rest don't mention products, e.g., "*Which ones are discounted*", or "*Higher price*".

3.2 Descriptive Analysis

To better understand the collected data, and to identify differences between regions, categories, *etc*..., we computed some basic features of the refinement utterances: word counts, entropy per word for a bi-gram language model, use of adjectives and modifiers, dependency tree depth, and whether the utterance could be parsed as a complete English sentence having subject, verb, and object. We find moderate but statistically significant differences between regional populations which may have implications for the applicability of this data set to other regions. Table 3 shows the statistically significant differences. We found that utterances from IN had a much longer tail on many of these statistics. While the mean word count is significantly smaller in IN than the US, the maximum word count in IN is nearly three times that in the US.

Adjectives and modifiers are used much more in IN than US, which appears to reflect workers speaking not only the refinement, but the original product search query as well. For example "*I'm looking for a blue straight fit trouser pant*" includes much more than the target color refinement, "*blue*". US workers had an increased tendency to speak in complete sentences, e.g., "*Please display more perfumes from Calvin Klein*" instead of simply "*Calvin Klein*", both of which are utterances found in the dataset. Perhaps the most curious distinction is US workers' increased tendency to use first-person pronouns compared to other workers. These are typically phrased as "*I only want to see...*", "*Show me...*", "*I want the* {*attribute*} *to be* {*value*}", "*I'm looking for ...*" etc.

Region	US	IN
word count	7.2	5.7
adjective/modifier count	0.82	1.0
mean tree depth	1.8	1.5
per-word entropy	0.68	0.94
% using first-person pronouns	16	2.7
% complete sentences	23	17

Table 3: Differences between utterances from different regions. All differences are statistically significant ($p < 0.01$, with a Tukey multiple-comparison correction).

Unsurprisingly, the per-word entropy is quite a bit higher for brand refinements than any other type but rating, while the word count is smaller for brand refinements than most others. In brand refinements the usage of first person pronouns is higher than any other type. Ratings refinements are the longest overall, with the deepest dependency trees.

We note that the distribution of refined attributes differs across regions. With one notable exception, we do not see any statistically significant, consequential, and consistent interactions between region and refined attribute when used to predict the features shown in Table 3. That exception is that review rating refinements are even longer and have even more per-word entropy in the US than IN than would be predicted otherwise. However, these findings only exaggerate trends that are already present in the data overall. This largely serves to suggest that review ratings are some of the most complex in the dataset.

4 Empirical Investigation

In this section we present a set of experiments on intent detection, specifically the recognition of attribute and product categories in a search refinement utterance. We aim to show that the VoiSeR dataset enables building models for intent recognition in voice search refinement. Moreover, we show the contribution of contextual information, e.g., the previous utterance, is beneficial, highlighting the need for large scale datasets like VoiSeR for developing contextual intent recognition models.

Set	**# Utterances**	**# Workers**
Train	8,432	337
Validation	653	28
Test	725	20

Table 4: Data distribution and number of workers in the training, validation and test splits.

Experimental setup. In order to train and test the model, we split the data into `train`, `validation` and `test` portions, as reported in table 4. The splitting has been done at Worker level, i.e., utterances from each Worker appears only in a single portion.

Model. We model the recognition of attribute and product mentions in a refinement utterance as a sequence tagging problem in an IOB2[5] tagging schema. For example, given the utterance "*Show me a Vega brand hair curler*", its correct tagging is "`O O O B-Attribute I-Attribute B-Product I-Product`". Notice that we do not tag the attribute type, but only a general category `Attribute`. This is to enable the model to generalize over the linguistic expressions that are similar among different attribute types. The model we implemented is a BERT-based (Devlin et al., 2019) sequence tagger. BERT is used as the encoder to obtain the contextualized embeddings of each token w_i of a sentence, i.e., $h_i = BERT(w_i)$. After applying dropout on h_i, a linear classifier is used to obtain the c_i distribution over the IOB categories for each token w_i, i.e., $c_i = softmax(Wh_i + b)$, where softmax refers to the function to transform the scores in output probabilities and W and b are the weights and bias of the classifier, respectively. The model is trained by optimizing the cross entropy between the predicted categories and the true one-hot distributions for each token. We run experiments on two settings: (*i*) w/o context, i.e., the input to the model is only the refinement utterance; (*ii*) w/ context, i.e., the model receives as input both the original search query and the refinement utterance. The two utterances are separated by the `[SEP]` token, as usual in BERT (Devlin et al., 2019). The latter experiment will provide insights about the utility of contextual information to the model.

Experimental Setup. We used the `bert-base-uncased` model from the Huggingface repository (Wolf et al., 2019) as the encoder. The model is trained for 15 epochs with Early Stopping (patience=3) by tuning the following hyperparameters: dropout (0.0, 0.1) applied on the h_i representations; batch size (64, 128). The learning rate is set to 5e-5, with the Adam Optimizer with a warm-up of 100 steps.

Experimental Results. Table 5 reports the model performance (with and without context) at entire span level and at token level. The former measures the capability of the model in recognizing an entire attribute or product, i.e., the attribute or product are considered correctly predicted only if all their tokens are recognized by the model. The latter measures the capability of the model in partially recognizing an attribute or product in a voice refinement. The model performance is already promising when context is not leveraged, achieving 83.11 and 84.36 F1 for attribute and product, respectively. Contextual information provides a significant improvement, allowing the model to reach 84.94 and 86.86 F1 for attribute and product, respectively. This demonstrates that contextual information helps NLU models to better understand customer requests.

We performed an in-depth analysis to find out how the model performs on different attribute types. Tables 6 and 7 report results for each attribute type[6] in the w/o and w/ settings, respectively.

In both settings, the model achieves the best results on *discount*, while results are a bit worse on *material*, *review-rating* and *brand*. This is a consequence of the lower linguistic variability associated with the *discount* attribute. On the other hand, refinements on *review-rating* are on average the

---

[5] IOB is a short for Inside, Outside, Beginning. In the IOB2 schema, the B- prefix before a tag indicates the beginning of an entity. For subsequent words of an entity the I- prefix is used. Words not belonging to any entity are tagged with O.

[6] Again, we do not tag the attribute type; we report the measures w.r.t. the target attribute used in the Mechanical Turk task.

Model		P	R	F1
w/o ctx	**Attr**	81.28 (95.47)	85.01 (95.27)	83.11 (95.37)
	Prod	83.08 (**94.22**)	85.67 (94.39)	84.36 (94.30)
w/ ctx	**Attr**	**83.71 (96.05)**	**86.21 (95.38)**	**84.94**† **(95.71)**
	Prod	**86.20** (94.16)	**87.54 (94.83)**	**86.86**† **(94.49)**

Table 5: Experimental results on the attribute and product extraction task. The sequence tagging models is tested when using or not the context. Reported metrics are Precision (P), Recall (R) and F1 at entire span level (in parenthesis the same metrics at token level). The symbol † marks a statistical significant difference with a paired t-test ($\alpha = 0.05$, for both Attribute and Product categories the p-value is ~ 0.009).

Attribute		Precision	Recall	F1
brand	**Attr**	79.45	83.40	81.38
	Prod	80.37	84.52	82.39
color	**Attr**	83.17	83.57	83.37
	Prod	81.62	87.40	84.41
discount	**Attr**	84.21	89.51	86.78
	Prod	91.40	90.43	90.91
material	**Attr**	79.82	84.26	81.98
	Prod	92.75	88.89	90.78
price	**Attr**	88.77	90.22	89.49
	Prod	78.57	80.49	79.52
rating	**Attr**	69.17	77.97	73.31
	Prod	80.00	84.51	82.19

Table 6: Span-level Precision (P), Recall (R) and F1 of the model when computing the performances by dividing the test set on the target attribute used in the data collection (w/o context).

Attribute		Precision	Recall	F1
brand	**Attr**	78.82	83.40	81.05
	Prod	84.62	85.16	84.89
color map	**Attr**	84.29	85.51	84.89
	Prod	85.61	88.98	87.26
discount	**Attr**	85.33	89.51	87.37
	Prod	93.55	92.55	93.05
material	**Attr**	80.91	82.41	81.65
	Prod	88.89	88.89	88.89
price	**Attr**	89.78	90.76	90.27
	Prod	81.75	83.74	82.73
review rating	**Attr**	84.17	85.59	84.87
	Prod	86.30	88.73	87.50

Table 7: Span-level Precision (P), Recall (R) and F1 of the model when computing the performances by dividing the test set on the target attribute used in the data collection (w/ context).

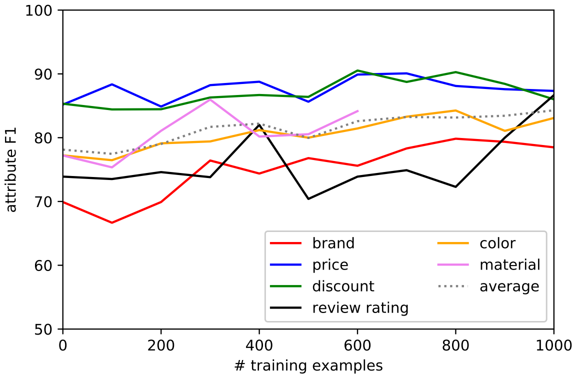


Figure 2: Learning curves for the attribute extraction task. The curve for attribute type a is obtained by training the model on all the utterances in the training set targeting a different attribute and on an increasing number of utterances targeting the attribute a.

longest and have the highest linguistic variability, as discussed in §3.2. Similarly, *brand* refinements contain a lot of rare words, i.e., the brands. Finally, a possible explanation for the lower performance on the *material* refinements is the smaller training size, as shown in Table 2. Less variability w.r.t. the attribute type of the refinement is observed on the product recognition. Again, the performance is generally higher when using the previous utterance as context. In this case, the model is able to make better predictions especially for *rating* (+12 F1 points).

Finally, we conducted a set of experiments to study the model capability to generalize to attribute types rarely, or never, observed in training. Figure 2 reports a learning curve for each attribute type we collected. The learning curve is computed as follows: for each reported attribute a and for each size s in the x-axis, we train a context-based model with s utterances of the attribute[7] a and with all the utterances for the other attributes. In figure 2 we report the F1 of the attribute recognition for each attribute and also the average F1 for all the attributes at each size s. The average F1 score on never seen attribute types (i.e., when using $s = 0$ examples for each attribute) is ~70: even with no training examples, the model obtains good results on all attributes. In terms of generalization capability, the best performance is observed on $price$ and $discount$, with learning curves that start from high scores and remain almost flat. This is not surprising, since these two attribute types are related, and the model can easily generalize from one to the

---

[7] The learning curve for *material* ends earlier because there are fewer utterances in the training set.

other. The *brand* attribute improves more when increasing the number of training examples. This may be due to the large lexical variability in this attribute: in e-commerce catalogues the number of brand names is typically very large.

The average learning curve becomes almost flat after 400 examples, and overall the model demonstrates a very good generalization capability on new attribute types. This suggests that the collected dataset has a reasonable size and that it represent a valuable resource to bootstrap NLU models for voice-based search.

5 Related Work

Human-computer information retrieval (HCIR) (Marchionini, 2006) combines the fields of human-computer interaction (HCI) and information retrieval (IR) to create systems that improve search by taking into account the human context, or through a multi-step search process that provides the opportunity for human feedback. Modern search engines implement several HCIR strategies including relevance feedback (Ruthven and Lalmas, 2003), automatic query completion (Cai and de Rijke, 2016) and faceted search (Tunkelang, 2009). In particular, most of the e-commerce websites provide faceted search by allowing users to filter on several product attributes. With our work we would like to study how a similar experience can be made available in a voice-based interaction.

Due to the increasing availability of smartphones and voice assistants like Amazon Alexa or Google Home, voice-based search is becoming ubiquitous (Guy, 2016). Voice-based Web search is significantly different from text-based search (Guy, 2016, 2018), notably for voice-based query reformulation or refinement (Hassan Awadallah et al., 2015). The voice-based interaction is especially challenging when no visual interface is available and both inputs and outputs are entirely provided by voice (Ingber et al., 2018). To improve user experience and engagement in voice-search some work has been recently done. Gamzu et al. (2020) propose a query rewriting approach for handling mispronounced, misexpressed, and misunderstood customer queries in voice shopping. Filice et al. (2020) describe the problem of reformulating answers from a community question answering system to make them suitable for a voice interaction.

There have been prior efforts in creating open multi-turn voice-based search datasets, but because of the lack of effective automated systems for these tasks, the datasets were collected in a lab using a Wizard-of-Oz approach, where a hidden human participant playing the part of the search engine, e.g., (Trippas et al., 2017, 2020; Vakulenko et al., 2019), and (Trippas and Thomas, 2019). The dataset we contribute complements prior efforts by providing a large-scale collection of voice-based query reformulation, collected in a realistic environment with a real search engine, using automated ASR, thus providing a critical resource for training robust, high-performance models for voice refinement.

Recently, several dialog-related datasets have been (Mehri et al., 2020; Crook et al., 2019) proposed. They target task-oriented conversations but do not specifically focus on search tasks.

6 Conclusions

In this paper, we discussed the problem of search refinement, a fundamental component for supporting multi-turn voice-based complex search tasks. We presented the challenges in the voice refinement problem, and introduced a large-scale, critically needed benchmark for training and evaluating models in this setting. Specifically, we introduced the first benchmark dataset, VoiSeR, specifically developed for analyzing and measuring the linguistic phenomena underlying the voice refinements in second-turn searches in the e-commerce domain. We emphasize that the target search facets and attributes are (by design) general, and thus the data and the resulting models can be used, with or without adaptation, for a wide range of conversational search refinement and intent prediction tasks.

We provided a detailed description of the data collection and annotation processes, and identified interesting statistical and linguistic phenomena in the dataset. We complement the data release with an extensive empirical investigation, which demonstrates that (*i*) our dataset is a valuable resource for training NLU models for voice-based search and (*ii*) using contextual information for recognizing product and attribute mentions is beneficial. Together, the new VoiSeR dataset and the analysis in this paper enable productive research for developing systems for voice-based complex search tasks.

References

Fei Cai and Maarten de Rijke. 2016. *A Survey of Query Auto Completion in Information Retrieval*. Now Publishers Inc., Hanover, MA, USA.

Paul A Crook, Shivani Poddar, Ankita De, Semir Shafi, David Whitney, Alborz Geramifard, and Rajen Subba. 2019. Simmc: Situated interactive multimodal conversational data collection and evaluation platform. *arXiv preprint arXiv:1911.02690*.

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

Simone Filice, Nachshon Cohen, and David Carmel. 2020. Voice-based reformulation of community answers. In *Proceedings of The Web Conference 2020*, WWW '20, page 2885–2891, New York, NY, USA. Association for Computing Machinery.

Iftah Gamzu, Marina Haikin, and Nissim Halabi. 2020. Query rewriting for voice shopping null queries. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20.

Ido Guy. 2016. Searching by talking: Analysis of voice queries on mobile web search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 35–44, New York, NY, USA. Association for Computing Machinery.

Ido Guy. 2018. The characteristics of voice search: Comparing spoken with typed-in mobile web search queries. *ACM Transactions on Information Systems (TOIS)*, 36(3):1–28.

Ahmed Hassan Awadallah, Ranjitha Gurunath Kulkarni, Umut Ozertem, and Rosie Jones. 2015. Characterizing and predicting voice query reformulation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 543–552.

Amir Ingber, Arnon Lazerson, Liane Lewin-Eytan, Alexander Libov, and Eliyahu Osherovich. 2018. The challenges of moving from web to voice in product search. In *Proceedings of the 1st International Workshop on Generalization in Information Retrieval*.

Gary Marchionini. 2006. Toward human-computer information retrieval. *Bulletin of the American Society for Information Science and Technology*, 32(5):20–22.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue.

Ian Ruthven and Mounia Lalmas. 2003. A survey on the use of relevance feedback for information access systems. *Knowledge Eng. Review*, 18(2):95–145.

Johanne Trippas and Paul Thomas. 2019. Data sets for spoken conversational search. In *Workshop on Barriers to Interactive IR Resources Re-use at the ACM SIGIR Conference on Human Information Interaction and Retrieval*.

Johanne R Trippas, Damiano Spina, Lawrence Cavedon, and Mark Sanderson. 2017. A conversational search transcription protocol and analysis. In *Proceedings of SIGIR 1st International Workshop on Conversational Approaches to Information Retrieval (CAIR'17), CAIR*.

Johanne R Trippas, Damiano Spina, Paul Thomas, Mark Sanderson, Hideo Joho, and Lawrence Cavedon. 2020. Towards a model for spoken conversational search. *Information Processing & Management*, 57(2):102162.

Daniel Tunkelang. 2009. *Faceted Search*. Morgan and Claypool Publishers.

Svitlana Vakulenko, Kate Revoredo, Claudio Di Ciccio, and Maarten de Rijke. 2019. Qrfa: A data-driven model of information-seeking dialogues. In *European Conference on Information Retrieval*, pages 541–557. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Event-Driven News Stream Clustering using Entity-Aware Contextual Embeddings

Kailash Karthik Saravanakumar[*2] **Miguel Ballesteros**[1]
Muthu Kumar Chandrasekaran[1] **Kathleen McKeown**[1,2]
[1]Amazon AI, USA
[2]Department of Computer Science, Columbia University, NY, USA
{ballemig, cmuthuk, mckeownk}@amazon.com
{kailashkarthik.s}@columbia.edu

Abstract

We propose a method for online news stream clustering that is a variant of the non-parametric streaming K-means algorithm. Our model uses a combination of sparse and dense document representations, aggregates document-cluster similarity along these multiple representations and makes the clustering decision using a neural classifier. The weighted document-cluster similarity model is learned using a novel adaptation of the triplet loss into a linear classification objective. We show that the use of a suitable fine-tuning objective and external knowledge in pre-trained transformer models yields significant improvements in the effectiveness of contextual embeddings for clustering. Our model achieves a new state-of-the-art on a standard stream clustering dataset of English documents.

1 Introduction

Human presentation and understanding of news articles is almost never isolated. Seminal real-world events spawn a chain of strongly correlated news articles that form a news story over time. Given the abundance of online news sources, the consumption of news in the context of the stories they belong to is challenging. Unless people are able to scour the many news sources multiple times a day, major events of interest can be missed as they occur. The real-time monitoring of news, segregating articles into their corresponding stories, thus enables people to follow news stories over time.

This goal of identifying and tracking topics from a news stream was first introduced in the Topic Detection and Tracking (TDT) task (Allan et al., 1998). Topics in the news stream setting usually correspond to real-world events, while news articles may also be categorized thematically into sports, politics, etc. We focus on the task of clustering news on the basis of event-based story chains. We make a distinction between our definition of an *event topic*, which follows TDT and refers to large-scale real-world events, and the fine-grained events used in trigger-based event detection (Ahn, 2006). Given the non-parametric nature of our task (the number of events is not known beforehand and evolves over time), the two primary approaches have been topic modeling using Hierarchical Dirichlet Processes (HDPs) (Teh et al., 2005; Beykikhoshk et al., 2018) and Stream Clustering (MacQueen, 1967; Laban and Hearst, 2017; Miranda et al., 2018). While HDPs use word distributions within documents to infer topics, stream clustering models use representation strategies to encode and cluster documents. Contemporary models have adopted stream clustering using TF-IDF weighted bag of words representations to achieve state-of-the-art results (Staykovski et al., 2019).

In this paper, we present a model for event topic detection and tracking from news streams that leverages a combination of dense and sparse document representations. Our dense representations are obtained from BERT models (Devlin et al., 2019) fine-tuned using the triplet network architecture (Hoffer and Ailon, 2015) on the event similarity task, which we describe in Section 3. We also use an adaptation of the triplet loss to learn a Support Vector Machine (SVM) (Boser et al., 1992) based document-cluster similarity model and handle the non-parametric cluster creation using a shallow neural network. We empirically show consistent improvement in clustering performance across many clustering metrics and significantly less cluster fragmentation.

The main contributions of this paper are:

- We present a novel technique for event-driven news stream clustering, which, to the best of our knowledge, is the first attempt of using

---

[*] Work done during internship at Amazon

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2330–2340
April 19 - 23, 2021.

contextual representations for this task.

- We investigate the impact of BERT's fine-tuning objective on clustering performance and show that tuning on the event similarity task using triplet loss improves the effectiveness of embeddings for clustering.
- We demonstrate the importance of adding external knowledge to contextual embeddings for clustering by introducing entity awareness to BERT. Contrary to a previous claim (Staykovski et al., 2019), we empirically show that dense embeddings improve clustering performance when augmented with task-pertinent fine-tuning, external knowledge and the conjunction of sparse and temporal representations.
- We analyze the problem of cluster fragmentation and show that it is not captured well by the metrics reported in the literature. We propose an additional metric that captures fragmentation better and report results on both.

2 Related Work

In this section, we introduce the TDT task, prior work on tracking events from news streams and a few related parametric variants of the TDT task.

The goal of the TDT task is to organize a collection of news articles into groups called topics. Topics are defined as sets of highly correlated news articles that are related to some seminal real-world event. This is a narrower definition than the general notion of a topic which could include subjects (like *New York City*) as well. TDT defines an event to be represented by a triple <location, time, people involved>, which spawns a series of news articles over time. We are interested in all five sub-tasks of TDT - story segmentation, first story detection, cluster detection, tracking and story link detection - though we do not explicitly tackle these sub-problems individually.

After the initial work on the TDT corpora, interest in news stream clustering was rekindled by the news tracking system *NewsLens* (Laban and Hearst, 2017). NewsLens tackled the problem in multiple stages: (1) document representation through keyphrase extraction; (2) non-parametric batch clustering using the Louvian algorithm (Blondel et al., 2008); and (3) linking of clusters across batches. Staykovski et al. (2019) presented a modified version of this model, using TF-IDF bag of words document representations instead of keywords. They also compared the relative performance of sparse TF-IDF bag of words and dense doc2vec (Le and Mikolov, 2014) representations and showed that the latter performs worse, both individually and in unison with sparse representations. Linger and Hajaiej (2020) extended this batch clustering idea to the multilingual setting by incorporating a Siamese Multilingual-DistilBERT (Sanh et al., 2019) model to link clusters across languages.

In contrast to the batch-clustering approach, Miranda et al. (2018) adopt an online clustering paradigm, where streaming documents are compared against existing clusters to find the best match or to create a new cluster. We adopt this stream clustering approach as it is robust to temporal density variations in the news stream. Batch clustering models tune a batch size hyperparameter that is both training corpus dependent and might not be able to adjust to temporal variations in stream density. In their model, they also use a pipeline architecture, having separate models for document-cluster similarity computation and cluster creation. Similarity between a document and cluster is computed along multiple document representations and then aggregated using a Rank-SVM model (Joachims, 2002). The decision to merge a document with a cluster or create a new cluster is taken by an SVM classifier. Our model also follows this architecture, but critically adds dense document representations, an SVM trained on the adapted triplet loss for aggregating document-cluster similarities and a shallow neural network for cluster creation.

News event tracking has also been framed as a non-parametric topic modeling problem (Zhou et al., 2015) and HDPs that share parameters across temporal batches have been used for this task (Beykikhoshk et al., 2018). Dense document representations have been shown to be useful in the parametric variant of our problem, with neural LDA (Dieng et al., 2019a; Keya et al., 2019; Dieng et al., 2019b; Bianchi et al., 2020), temporal topic evolution models (Zaheer et al., 2017; Gupta et al., 2018; Zaheer et al., 2019; Brochier et al., 2020) and embedding space clustering (Momeni et al., 2018; Sia et al., 2020) being some prominent approaches in the literature.

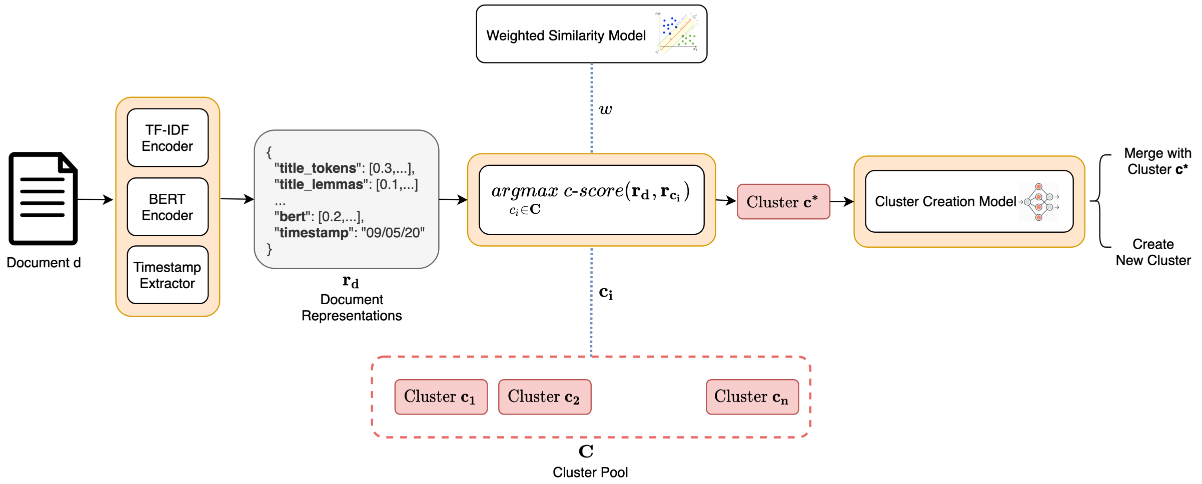


Figure 1: The architecture of the news stream clustering model, showing the clustering process for a single document in the news stream. At the end of the clustering process for each document, the cluster pool is updated based on the output from the cluster creation model, either by adding document d to cluster c^* or by creating a new cluster with the document.

3 Methodology

Our clustering model is a variant of the streaming K-means algorithm (MacQueen, 1967) with two key differences: (1) we compute the similarity between documents and clusters along a set of representations instead of a single vector representation; and (2) we decide the cluster membership using the output of a neural classifier, a learned model, instead of a static tuned threshold.

At any point in time t, let n be the number of clusters the model has created thus far, called the cluster pool. Given a continuous stream of news documents, the goal of the model is to decide the cluster membership (if any) for each input document. In our task, we assume that each document belongs to a single event, represented by a cluster. The architecture of the model, as shown in Figure 1, consists of three main components : (1) document representations, (2) document-cluster similarity computation using a weighted similarity model and (3) cluster creation model. In what follows, we describe each of these components.

3.1 Document Representations

Documents in the news stream have a set of representations, as shown in Figure 1, where each representation is one of the following types - sparse TF-IDF, dense embedding or temporal. We describe below these representation types and how clusters, which are created by our model, build representations from their assigned documents.

3.1.1 TF-IDF Representation

Separate TF-IDF models that are trained only on the tokens, lemmas and entities in a corpus are used to encode documents separately. For every document in the news stream, its title, body and title+body are each encoded into separate bags of tokens, lemmas and entities, creating nine sparse bag of word representations per document.

3.1.2 Dense Embedding Representation

Dense document representations are obtained by embedding the body of documents using BERT, with pre-trained BERT (P-BERT) without any fine-tuning as our baseline embedding model. In order to improve the effectiveness of contextual embeddings for our clustering task, we experiment with enhancements along two dimensions: (1) the fine-tuning objective, and (2) the provision of external knowledge. We train separate BERT models for (1) and (2) and use them to encode documents.

To evaluate the impact of the fine-tuning objective, we fine-tune BERT models on two different tasks - event classification (C-BERT) and event similarity (S-BERT). We also evaluate the impact of external knowledge on the embeddings through an entity-aware BERT architecture, which may be paired with either of the fine-tuning objectives.

Fine-tuning on Event Classification The goal of this fine-tuning is to tune the CLS token[1] embedding such that it encodes information about the

---

[1] The CLS token, introduced in (Devlin et al., 2019), is a special token added to the beginning of every document before being embedded by BERT

event that a document corresponds to. A dense and softmax layer are stacked on top of the CLS token embedding to classify a document into one of the events in the output space.

Fine-tuning on Event Similarity Fine-tuning on the task of event classification constrains the embedding of documents corresponding to different events to be non-linearly separable. Semantics about events can be better captured if the vector similarity between document embeddings encode whether they are from the same event or not.

For this, we adapt the triplet network architecture (Hoffer and Ailon, 2015) and fine-tune on the task of event similarity. Triplet BERT networks were introduced for the semantic text similarity (STS) task (Reimers and Gurevych, 2019), where the vector similarity between sentence embeddings was tuned to reflect the semantic similarity between them. We formulate the event similarity task, where the term “similarity” refers to whether two documents are from the same event cluster or not. In our task, documents from the same event are similar (with similarity = 1), while those from different events are dissimilar (with similarity = 0). Given the embeddings of an anchor document d_a, a positive document d_p (from the same event as the anchor) and a negative document d_n (from a different event), triplet loss is computed as

$$l_{triplet} = sim(d_a, d_n) - sim(d_a, d_p) + m \quad (1)$$

where sim is the cosine similarity function and m is the hyper-parameter margin.

Providing External Entity Knowledge In line with TDT’s definition, entities are central to events and thus need to be highlighted in document representations for our clustering task. We follow Logeswaran et al. (2019) to introduce entity awareness to BERT by leveraging knowledge from an external NER system. Apart from token, position and token type embeddings, we also add an entity presence-absence embedding for each token depending on whether it corresponds to an entity or not. The entity aware BERT model architecture is shown in Figure 2. This enhanced entity-aware model can then be coupled with the event similarity (E-S-BERT) objective for fine-tuning.

3.1.3 Temporal Representation

Documents are also represented with the timestamp of publication. Unlike TF-IDF and dense embeddings, which are vector valued representations, the temporal representation of a document is just a single value (e.g ”05-09-2020”) which has an associated subtraction operation. The difference between two timestamps is defined as the number of intervening days between them. Section 3.2 describes how these timestamps are used for clustering.

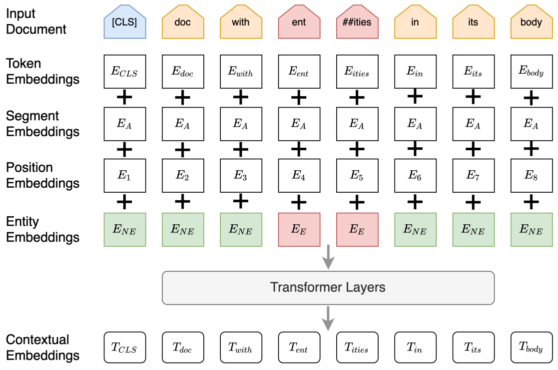


Figure 2: Entity-aware BERT model, with the additional entity presence (E_E) and absence (E_{NE}) embeddings

3.1.4 Cluster Representation

Since clusters are created and updated by our model, their representations need to be generated dynamically from the documents assigned to them. While documents in the news stream have a set of 11 representations (9 TF-IDF, dense embeddings and timestamp), clusters have two additional timestamp representations. Cluster representations are derived from documents in the cluster through aggregation. While dense embedding and sparse TF-IDF representations of a cluster are aggregated using mean pooling, clusters have three timestamp representations corresponding to different aggregation strategies - min, max and mean pooling.

3.2 Weighted Similarity Model

Once documents are encoded by a set of representations, they are compared to the clusters in the cluster pool to find the most compatible cluster. The similarity between a document and a cluster is computed along each representation separately and is then aggregated into a single compatibility score ($c\text{-}score$). While similarity along contextual embeddings and TF-IDF bag representations is computed using cosine similarity (as shown in Equation 2), timestamp similarity is computed using the Gaussian similarity function introduced in Miranda et al. (2018) (as shown in Equation 3).

Let $\mathbf{r_d}^v$ and $\mathbf{r_c}^v$ denote a dense or sparse vector representation of a document d and cluster c respectively. Let $\mathbf{r_d}^t$ and $\mathbf{r_c}^t$ denote their timestamp representations. Let (i, j) correspond to a pair of document-cluster representations of the same type (as defined in Section 3.1). Document-cluster similarity is computed along each representation and aggregated using a weighted summation as

$$\begin{aligned} sim(\mathbf{r_d}, \mathbf{r_c}) &= \{sim(\mathbf{r_d}^i, \mathbf{r_c}^j)\ \forall\ (i, j)\} \\ sim(\mathbf{r_d}^v, \mathbf{r_c}^v) &= \frac{\mathbf{r_d}^v \cdot \mathbf{r_c}^v}{|\mathbf{r_d}^v||\mathbf{r_c}^v|} \qquad (2) \\ sim(\mathbf{r_d}^t, \mathbf{r_c}^t) &= e^{-\frac{((\mathbf{r_d}^t - \mathbf{r_c}^t) - \mu)}{2\sigma^2}} \qquad (3) \\ c\text{-}score(\mathbf{r_d}, \mathbf{r_c}) &= \sum_{(i,j)} w_j \cdot sim(\mathbf{r_d}^i, \mathbf{r_c}^j) \end{aligned}$$

where μ and σ are tuned hyper-parameters of the temporal similarity function. It is noted here that since clusters have two additional timestamp representations, all three timestamp similarities are computed using the single document timestamp representation, as illustrated in Figure 3.

The dataset does not contain annotation for the degree of membership between a document and cluster and thus, the weights for combining the representation similarities can't be learned directly. To circumvent this issue, we train a linear model on a relevant task so that the trained weights can then be adapted to compute the compatibility score.

In our model, we train a linear model on a novel adaptation of the event similarity triplet loss used to train the S-BERT model. The triplet loss, as defined in Equation 1, can be adapted to a linear classifier if similarity has a related notion with regards to the classifier. SVM is an appropriate model since the degree of compatibility between a point x and a class c is given by the distance of the point from the class' decision hyperplane w_c. This distance, computed as $w_c \cdot x + b$, can thus be used as the similarity metric to adapt the triplet loss.

In our case, the inputs to the SVM model are vectors of document-cluster similarities along the set of representations $sim(\mathbf{r_d}, \mathbf{r_c})$. The adapted SVM-triplet loss is thus computed as shown below. Since we want to minimize this loss, we analyze its point of minima.

$$\begin{aligned} l_{svm-triplet} &= w \cdot sim(\mathbf{r_a}, \mathbf{r_n}) - w \cdot sim(\mathbf{r_a}, \mathbf{r_p}) \\ &\qquad + m \\ l_{svm-triplet} &= 0 \\ \implies\ m &= w \cdot (sim(\mathbf{r_a}, \mathbf{r_p}) - sim(\mathbf{r_a}, \mathbf{r_n})) \end{aligned}$$

The adapted triplet loss can thus be modeled as a classification task with inputs $(sim(\mathbf{r_a}, \mathbf{r_p}) - sim(\mathbf{r_a}, \mathbf{r_n}))$ and the outputs m. For mathematical convenience, we set $m = 1$ without loss of generality. In this manner, we transform the event similarity triplet loss objective into a classification objective to train an SVM model. The novelty of this supervision is that we focus on learning useful weights and not a useful classifier. The learned weights, which minimize the event similarity triplet loss, are utilized for document-cluster c-score computation. During the clustering process, a document d is compared against all the clusters in the pool $\mathbf{C}$ to determine the most compatible cluster c^* as

$$c^* = \underset{c\,\in\,\mathbf{C}}{\arg\max}\ c\text{-}score(\mathbf{r_d}, \mathbf{r_c})$$

3.3 Cluster Creation Model

Since our clustering problem is non-parametric, each document in the stream could potentially be the start of a new event cluster. Thus, the most compatible cluster c^* might not actually be the cluster that the document corresponds to. Given a document and its most compatible cluster, the cluster creation model decides whether or not a new cluster is to be created. For this, we employ a shallow neural network which takes document-cluster similarities along the set of representations as input and decides if a new cluster should be created. Since the dimensionality of the input space for the network is small, we use a shallow network to prevent overfitting.

4 Experiments and Results

4.1 Data

To train and evaluate our clustering models, we use the standard multilingual news stream clustering dataset (Miranda et al., 2018), which contains articles from English, French and German. For our clustering task, we only use the English subset of the corpus, which consists of 20,959 articles. Articles are annotated with language, timestamp and the event cluster to which they belong, in addition to their title and body text. We use the same training and evaluation split provided by Miranda et al. (2018) and use the training set to fine-tune the parameters of the clustering model. The training and evaluation sets are *temporally disjoint* to ensure that the clustering models are tuned independent of the events seen during training.

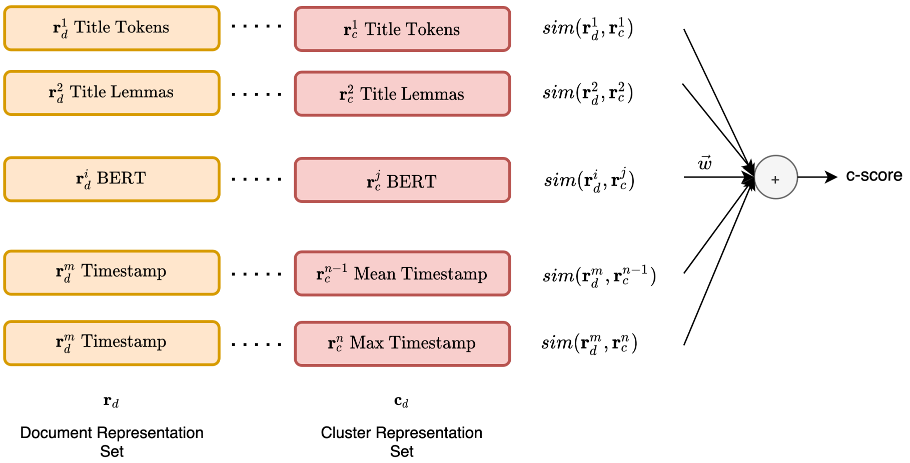


Figure 3: Computation of c-score: (a) similarities are computed for each representation individually using the appropriate similarity function (cosine or Gaussian); (b) subsequently, the computed similarities are aggregated into a single c-score value using the weights of the weighted similarity model (w)

4.2 Experimental Setup

We train our model pipeline in a sequence where each component model is supplied with the output from the component trained before in the sequence. For instance, the cluster creation model is trained using the embeddings from the fine-tuned BERT model and by selecting the most compatible cluster determined by the trained weighted similarity model. We experiment with multiple document representation sets, training all the component models each time and evaluating the entire clustering model on the test set.

We use the TF-IDF weights provided in the Miranda et al. (2018) corpus to ensure fair comparison with prior work. For training the event similarity BERT model (S-BERT), triplets are generated for each document using the batch-hard regime (Hermans et al., 2017) by picking the hardest positive and negative examples from its mini-batch[2]. We train the S-BERT model for 2 epochs using a batch size of 32, with 10% of the training data being used for linear warmup. We use Adam optimizer with learning rate $2e^{-5}$ and epsilon $1e^{-6}$. Document embeddings are obtained by mean pooling across all its tokens. For NER, we use the medium English model provided by spaCy (Honnibal and Montani, 2017).

Training instances for the weighted similarity and cluster creation models are generated by simulating the stream clustering process on the training set and assigning each document to its true event cluster. For the weighted similarity model, we generate triples of <document, true cluster, sampled negative cluster> and convert them into SVM training instances as mentioned in Section 3.2. Since all the training instances have the same label m, half the training set is negated to balance the dataset.

To generate training samples for the cluster creation model, the most compatible cluster is determined using the trained weighted similarity model for each document. A sample is then generated with input as the document-cluster similarities and output as 0 or 1 depending on whether the true cluster for that document is in the cluster pool or not. The dataset contains over 12k documents but only 593 clusters, entailing that the fraction of training samples where a new cluster is created is only 5%, making the dataset extremely biased. To mitigate this issue, we use the SVMSMOTE algorithm (Nguyen et al., 2011) to oversample the minority class and make the classes equal in size. For cluster creation, we train a shallow single layer neural network with two nodes using the L-BFGS solver (Nocedal, 1980). The weighted similarity and cluster creation models are trained using 5-fold cross validation to tune hyper-parameters and then on the entire training set using the best settings.

The clustering output is evaluated by comparing against the ground truth clusters. We report results on the B-Cubed metrics (Bagga and Baldwin, 1998) in Table 1 to compare against prior work.

4.3 Results

TF-IDF sets a tough baseline: Prior work has shown that sparse TF-IDF bag representa-

---

[2] We use the batch-hard implementation provided by Reimers and Gurevych (2019) at `https://github.com/UKPLab/sentence-transformers`

Model	Clusters Count (True Count - 222)	B-Cubed Metrics Precision	Recall	F_1 Score
Laban and Hearst (2017)	873	94.37	85.58	89.76
Miranda et al. (2018)	326	94.27	90.25	92.36
Staykovski et al. (2019)	484	95.16	93.66	94.41
Linger and Hajaiej (2020)	298	94.19	93.55	93.86
Ours - TF-IDF	530	93.50	80.23	86.36
Ours - TF-IDF (out-of-order)	413	90.57	87.51	89.01
Ours - TF-IDF + Time	222	87.57	96.27	91.72
Ours - E-S-BERT	452	79.76	60.77	68.98
Ours - E-S-BERT + Time	471	92.70	74.69	82.73
Ours - TF-IDF + P-BERT + Time	196	83.12	**97.26**	89.63
Ours - TF-IDF + C-BERT + Time	321	83.10	91.33	87.03
Ours - TF-IDF + S-BERT + Time	247	88.30	96.10	92.04
Ours - TF-IDF + E-S-BERT	433	89.40	86.99	88.18
Ours - TF-IDF + E-S-BERT (out-of-order)	384	91.15	88.60	89.86
Ours - TF-IDF + E-S-BERT + Time	276	**94.28**	95.25	**94.76**

Table 1: Results of clustering performance for different document representation strategies as compared against contemporary models. P-BERT refers to pre-trained BERT; C-BERT refers to BERT fine-tuned on event classification S-BERT refers to BERT fine-tuned using triplet loss on event similarity; E-S-BERT refers to entity aware BERT fine-tuned on event similarity.

tions achieve competitive performance (Laban and Hearst, 2017; Miranda et al., 2018) and our experiments validate this observation. The clustering model that uses only sparse TF-IDF bags to represent documents achieves a very high score of 86.8% B-Cubed F_1 score, as shown in Table 1. If TF-IDF bags are used in combination with timestamps, then the performance further increases to 91.7%, setting a tough baseline to beat.

Contextual embeddings, by themselves, achieve sub-par clustering performance: In line with prior work, we observe that dense document embeddings, both when used as the sole representation and in conjunction with timestamps, are unable to match the clustering performance of TF-IDF bags. It can be seen in Table 1 that even our best BERT model (entity aware BERT trained on event similarity) only achieves an F1 score of 69% individually and 82.7% when combined with timestamp representations. These scores are 17.8% and 9% lower than their corresponding TF-IDF counterparts. BERT embeddings are richer representations that encode linguistic information including syntax and semantics through its pre-training. Thus, the model is unable to distinguish between events at the desired granularity and ends up clustering together topically related events (for instance, two different events related to soccer).

Fine-tuning objective impacts the effectiveness of embeddings for clustering: In most NLP tasks, fine-tuning contextual embeddings on a related pertinent objective is beneficial, we observe that the choice of fine-tuning objective is critical to the task performance. While the baseline pre-trained P-BERT model achieves a clustering score of 89.6% when used in conjunction with TF-IDF and timestamp representations (TF-IDF + P-BERT + Time), fine-tuning embeddings on event classification (TF-IDF + C-BERT + Time) drops the performance to 87%. This drop in performance can be attributed to the following reasons. Firstly, the large output space (593 events) and small dataset size (12k documents) make it hard for the model to learn effectively during fine-tuning. In addition to this, the classification objective requires that the embeddings of documents from different events be non-linearly separable. But this is not directly compatible with how the embeddings are used by the weighted similarity model, which is to compute cosine similarity. This discordance entails that the fine-tuning process degrades the clustering performance. The event similarity triplet loss is a more suitable fine-tuning objective and it is observed that fine-tuning BERT on this objective (TF-IDF + S-BERT + Time) results in a better clustering performance of 92.04%.

External entity knowledge makes embeddings more effective for clustering: The introduction of external knowledge through the entity aware BERT architecture significantly improves the clus-

Metric	TF-IDF + E-S-BERT + Time	Miranda	Gain
B-Cubed	94.76	92.36	$2.40^{\dagger}$
CEAF-e	76.93	69.57	$7.36^{\dagger}$
CEAF-m	93.31	90.19	$3.12^{\dagger}$
MUC	99.30	98.88	$0.42^{\ddagger}$
BLANC	98.13	96.93	$1.20^{\S}$
V Measure	97.98	97.01	$0.97^{\dagger}$
Adjusted Rand Score	96.26	93.87	$2.39^{\S}$
Adjusted Mutual Information	97.99	97.02	$2.97^{\S}$
Fowlkes Mallows Score	96.38	94.11	$2.27^{\S}$

Table 2: Results of clustering performance across different evaluation metrics. For each metric computed using precision, recall and F-1 scores, only the F-1 scores are reported. Statistically significant gains, with $p <<< 0.001$ are denoted by † and $p < 0.01$ by ‡. Gains denoted by $\S$ are not evaluated for significance, in line with literature.

tering performance of the model. It can be seen in Table 1 that introducing entity awareness and training on the event similarity task (TF-IDF + E-S-BERT + Time) results in a clustering score of 94.76%[3], achieving a new state-of-the-art on the dataset[4]. The results are statistically significant and p values from a paired student's t-test are reported in Table 2. This is almost 3 points better than the corresponding model without entity awareness, which highlights the importance of this external knowledge. When given external knowledge from an NER system, the BERT model, like sparse TF-IDF representations, is able to draw attention to entities and highlight them in the document embeddings. It is observed that the model learns to project entities and non-entities in mutually orthogonal directions and thereby adds emphasis to entities.

In our experiments, we observe an increase of almost 1 point in F_1 score by considering only a subset of the OntoNotes corpus (Weischedel et al., 2013) labels [5]. Ignoring entity classes like ORDINAL and CARDINAL helps as they don't provide useful information for our clustering task. The scores reported in Table 1 correspond to entity-aware models trained on this label subset. We also experimented with separate embeddings for each entity type instead of the binary entity presence-absence embeddings and observed that it degrades F_1 score by more than 2 points.

Ablating time and non-streaming input: When we ablate timestamp from the representation (rows that are not marked with "Time" in Table 1) and then stream documents in random order (rows marked with ("out- of-order") in Table 1), the number of clusters increase over when accounting for time. When ablating time, we also observe that supplying documents in random order produces fewer clusters and better b-cubed F_1 scores. We observe examples of clusters that are incorrectly merged in the absence of temporal information (in the out-of-order setting). See Appendix for actual examples from our output.

Cluster fragmentation is not captured well by B-Cubed metrics The improvements our model makes can be seen clearly by observing the number of clusters created by the model. While the previous state-of-the-art model produced 484 clusters, ours produces only 276[6], which is closer to the true cluster count of 222. Our model produces far less cluster fragmentation, resulting in a 79.4% reduction in the number of erroneous additional clusters created. We argue that this is an important improvement that is not well captured by the small increase in B-Cubed metrics.

While B-Cubed F_1 score is the standard metric reported in the literature, it is an article-level metric which gives more importance to large clusters. This entails that B-Cubed metrics heavily penalize the model's output for making mistakes on large clusters while mistakes on smaller clusters can fall through without incurring much penalty. In our experiments, we observed that this property of the metric prevents it from capturing cluster fragmentation errors on smaller events. In the news stream clustering setting, small events may correspond to recent salient events and thus, we want our metric

---

[3] The mean and standard deviation of the precision, recall and F-1 scores over five independent training and evaluations of our model are 94.64 ± 0.28, 94.72 ± 1.33 and 94.75 ± 0.59.

[4] We observe similar results on the TDT Pilot dataset (Allan et al., 1998), as shown in Section 4.4

[5] Our entity label subset consists PERSON, NORP, FAC, ORG, GPE, LOC, PRODUCT, EVENT, WORK OF ART, LAW and LANGUAGE.

[6] The mean and standard deviation of the cluster count over five independent training and evaluations of our model are 312 ± 27.

to be agnostic to the size of the clusters.

We thus use an additional metric that weights every cluster equally - CEAF-e (Luo, 2005). The CEAF-e metric creates a one-to-one mapping between the clustering output and gold clusters using the Kuhn-Munkres algorithm. The similarity between a gold cluster G and an output cluster O is computed as the fraction of articles that are common to the clusters. Once the clusters are aligned, precision and recall are computed using the aligned pairs of clusters. This ensures that unaligned clusters contribute to a penalty in the score and cluster fragmentation and coalescing is captured by the metric.

In order to ensure that our model's better performance is metric-agnostic, we also empirically evaluated our clustering model against prior work using several clustering metrics, the results of which are presented in Table 2. For this, we compare with Miranda et al. (2018) since their results are readily replicable. It can be observed that our model consistently achieves better performance across most metrics and is thus robust to the metric idiosyncrasies. Our model achieves an improvement of 7.36 points on the CEAF-e metric, which shows that our clustering model performs better than contemporary models on smaller clusters as well.

4.4 Results on TDT

To validate the robustness of our clustering model, we evaluate it on the TDT Pilot corpus (Allan et al., 1998). The TDT Pilot corpus consists of a set of newswire and broadcast news transcripts that span the period from July 1, 1994 to June 30, 1995. Out of the 16,000 documents collected, 1,382 are annotated to be relevant to one of 25 events during that period. Unlike the Miranda et al. (2018) corpus, TDT Pilot does not have the article titles. We, therefore, train all the components of our ensemble architecture using only the document body text. The TDT corpus does not provide pre-trained TF-IDF weights, so we train the weights on the entire corpus as a pre-processing step. Unlike Miranda, the TDT Corpus also lacks standard train and test splits. We create our own splits across 25 events. The splits are described and listed in the Appendix.

In line with our observations on the Miranda et al. (2018) corpus, we observe similar results on the TDT corpus. We achieve the best result on this corpus on a model with TF-IDF representations combined with temporal representations, BERT entity-aware representations fine-tuned on the event similarity task. The best result has a b-cubed precision of **81.62**, b-cubed recall of **95.89** and a b-cubed F_1 **of 88.18**. We generate **12 clusters** which matches the number of clusters in the ground truth.

We show that even in a cross-corpus setting, dense contextual embeddings, when augmented with pertinent fine-tuning, external knowledge and the conjunction of sparse and temporal representations, are a potent representation strategy for event topic clustering.

5 Conclusion

In this paper, we present a novel news stream clustering algorithm that uses a combination of sparse and dense vector representations. We show that while dense embeddings by themselves do not achieve the best clustering results, enhancements like entity awareness and event similarity fine-tuning make them effective in conjunction with sparse and temporal representations. Our model achieves new state-of-the-art results on the Miranda et al. (2018) dataset. We also analyze the problem of cluster fragmentation noting that our approach is able to produce a similar number of clusters as in the test set, in contrast to prior work which produces far too many clusters. We note issues with the B-Cubed metrics and we complement our results using CEAF-e as an additional metric for our clustering task. In addtion, we provide a comprehensive empirical evaluation across many metrics to show the robustness of our model to metric idiosyncrasies.

Acknowledgments

We thank Chao Zhao, Heng Ji, Rishita Anubhai and Graham Horwood for valuable discussions.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer.

Adham Beykikhoshk, Ognjen Arandjelović, Dinh Phung, and Svetha Venkatesh. 2018. Discovering topic structures of a temporally evolving document corpus. *Knowledge and Information Systems*, 55(3):599–632.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. The dynamic embedded topic model. *arXiv preprint arXiv:2004.03974*.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, page 144–152, New York, NY, USA. Association for Computing Machinery.

Robin Brochier, Adrien Guille, and Julien Velcin. 2020. Inductive document network embedding with topic-word attention. *Advances in Information Retrieval*, page 326–340.

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

Adji B Dieng, Francisco J R Ruiz, and David M Blei. 2019a. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907*.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2019b. The dynamic embedded topic model. *arXiv preprint arXiv:1907.05545*.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. 2018. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1079–1089, New Orleans, Louisiana. Association for Computational Linguistics.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In defense of the triplet loss for person reidentification. *arXiv preprint arXiv:1703.07737*.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 133–142, New York, NY, USA. Association for Computing Machinery.

Kamrun Naher Keya, Yannis Papanikolaou, and James R. Foulds. 2019. Neural embedding allocation: Distributed representations of topic models. *arXiv preprint arXiv:1909.04702*.

Philippe Laban and Marti Hearst. 2017. newsLens: building and visualizing long-ranging news stories. In *Proceedings of the Events and Stories in the News Workshop*, pages 1–9, Vancouver, Canada. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1188–II–1196. JMLR.org.

Mathis Linger and Mhamed Hajaiej. 2020. Batch clustering for multilingual news streaming. *arXiv preprint arXiv:2004.08123*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif. University of California Press.

Sebastião Miranda, Artūrs Znotiņš, Shay B. Cohen, and Guntis Barzdins. 2018. Multilingual clustering of streaming news. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4535–4544, Brussels, Belgium. Association for Computational Linguistics.

Elaheh Momeni, Shanika Karunasekera, Palash Goyal, and Kristina Lerman. 2018. Modeling evolution of topics in large-scale temporal text corpora. In *Twelfth International AAAI Conference on Web and Social Media*.

Hien M. Nguyen, Eric W. Cooper, and Katsuari Kamei. 2011. Borderline over-sampling for imbalanced data classification. *Int. J. Knowl. Eng. Soft Data Paradigm.*, 3(1):4–21.

Jorge Nocedal. 1980. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Suzanna Sia, Ayush Dalmia, and Sabrina J. Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! *arXiv preprint arXiv:2004.14914*.

Todor Staykovski, Alberto Barrón-Cedeño, Giovanni Da San Martino, and Preslav Nakov. 2019. Dense vs. sparse representations for news stream clustering. In *Proceedings of Text2Story - 2nd Workshop on Narrative Extraction From Texts, co-located with the 41st European Conference on Information Retrieval, Text2Story@ECIR 2019, Cologne, Germany, April 14th, 2019*, volume 2342 of *CEUR Workshop Proceedings*, pages 47–52. CEUR-WS.org.

Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2005. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in neural information processing systems*, pages 1385–1392.

Ralph Weischedel et al. 2013. *OntoNotes Release 5.0 LDC2013T19. Web Download.* Linguistic Data Consortium, Philadelphia.

Manzil Zaheer, Amr Ahmed, and Alexander J. Smola. 2017. Latent LSTM allocation: Joint clustering and non-linear dynamic modeling of sequence data. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3967–3976, International Convention Centre, Sydney, Australia. PMLR.

Manzil Zaheer, Amr Ahmed, Yuan Wang, Daniel Silva, Marc Najork, Yuchen Wu, Shibani Sanan, and Surojit Chatterjee. 2019. Uncovering hidden structure in sequence data via threading recurrent models. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, page 186–194, New York, NY, USA. Association for Computing Machinery.

Deyu Zhou, Haiyang Xu, and Yulan He. 2015. An unsupervised Bayesian modelling approach for storyline detection on news articles. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1943–1948, Lisbon, Portugal. Association for Computational Linguistics.

A Appendix

The TDT corpus does not have a training and test split and we thus partition the corpus into two almost equal portions such that all documents in a single event are part of the same split. Our training set consists of 873 documents and our test set consists of 680 documents. The events in each partition of the TDT corpus is shown in Table 3

Events in Our Train Split
Karrigan Harding, Shannon Faulker, Quayle lung clot, Haiti ousts observers, NYC Subway bombing, Carlos the Jackal, USAir 427 crash, Lost in Iraq, Death of Kim Jong Il, Clinic Murders, Kobe Japan quake, Serbs violate Bihac, OK-City bombing
Events in Our Test Split
Pentium chip flaw, Cuban riot in Panama, Humble TX flooding, WTC Bombing trial, Cessna on White House, Aldrich Ames, Halls copter, Serbians down F-16, Carter in Bosnia, Comet into Jupiter, DNA in OJ trial, Justice-to-be Breyer

Table 3: Events in the training and test splits of the TDT Pilot corpus

Actual example of clusters incorrectly merged when documents are supplied out-of-temporal-order. Cluster label # 1024 in the Miranda test-set, contains articles on Qatar being selected as FIFA worldcup host and issues with immigrant labour there are discussed in negative sentiment. The ground truth is a large cluster with 1869 documents. An example document title in this cluster is "Qatar World Cup sponsors targeted for improving workers' rights" with timestamp 2015-05-25 15:27:00. Cluster # 288 is a singleton about an upcoming Boston Celtics game and has a negative tone on their recent performance with an article titled "Celtics kick away a winnable game" with timestamp 2014-11-06 10:27:00. This is incorrectly merged with cluster # 1024. There are many more clusters that are incorrectly merged with cluster # 1024.

Adversarial Learning of Poisson Factorisation Model for Gauging Brand Sentiment in User Reviews

Runcong Zhao, Lin Gui, Gabriele Pergola, Yulan He
Department of Computer Science, University of Warwick, UK
`{runcong.zhao,lin.gui,gabriele.pergola,yulan.he}@warwick.ac.uk`

Abstract

In this paper, we propose the Brand-Topic Model (BTM) which aims to detect brand-associated polarity-bearing topics from product reviews. Different from existing models for sentiment-topic extraction which assume topics are grouped under discrete sentiment categories such as '*positive*', '*negative*' and '*neural*', BTM is able to automatically infer real-valued brand-associated sentiment scores and generate fine-grained sentiment-topics in which we can observe continuous changes of words under a certain topic (e.g., '*shaver*' or '*cream*') while its associated sentiment gradually varies from negative to positive. BTM is built on the Poisson factorisation model with the incorporation of adversarial learning. It has been evaluated on a dataset constructed from Amazon reviews. Experimental results show that BTM outperforms a number of competitive baselines in brand ranking, achieving a better balance of topic coherence and uniqueness, and extracting better-separated polarity-bearing topics.

1 Introduction

Market intelligence aims to gather data from a company's external environment, such as customer surveys, news outlets and social media sites, in order to understand customer feedback to their products and services and to their competitors, for a better decision making of their marketing strategies. Since consumer purchase decisions are heavily influenced by online reviews, it is important to automatically analyse customer reviews for online brand monitoring. Existing sentiment analysis models either classify reviews into discrete polarity categories such as '*positive*', '*negative*' or '*neural*', or perform more fine-grained sentiment analysis, in which aspect-level sentiment label is predicted, though still in the discrete polarity category space. We argue that it is desirable to be able to detect subtle topic changes under continuous sentiment scores. This allows us to identify, for example, whether customers with slightly negative views share similar concerns with those holding strong negative opinions; and what positive aspects are praised by customers the most. In addition, deriving brand-associated sentiment scores in a continuous space makes it easier to generate a ranked list of brands, allowing for easy comparison.

Existing studies on brand topic detection were largely built on the Latent Dirichlet Allocation (LDA) model (Blei et al., 2003) which assumes that latent topics are shared among competing brands for a certain market. They however are not able to separate positive topics from negative ones. Approaches to polarity-bearing topic detection can only identify topics under discrete polarity categories such as '*positive*' and '*negative*'. We instead assume that each brand is associated with a latent real-valued sentiment score falling into the range of $[-1, 1]$ in which -1 denotes negative, 0 being neutral and 1 positive, and propose a Brand-Topic Model built on the Poisson Factorisation model with adversarial learning. Example outputs generated from BTM are shown in Figure 1 in which we can observe a transition of topics with varying topic polarity scores together with their associated brands.

More concretely, in BTM, a document-word count matrix is factorised into a product of two positive matrices, a document-topic matrix and a topic-word matrix. A word count in a document is assumed drawn from a Poisson distribution with its rate parameter defined as a product of a document-specific topic intensity and its word probability under the corresponding topic, summing over all topics. We further assume that each document is associated with a brand-associated sentiment score and a latent topic-word offset value. The occurrence count of a word is then jointly determined

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2341–2351
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

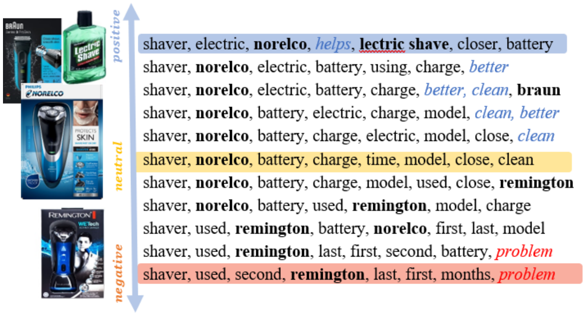


Figure 1: Example topic results generated from proposed Brand-Topic Model. We observe a transition of topics with varying topic polarity scores. Besides the change of sentiment-related words (e.g., '*problem*' in negative topics and '*better*' in positive topics), we could also see a change of their associated brands. Users are more positive about BRAUN, negative about REMINGTON, and have mixed opinions on NORELCO.

by both the brand-associated sentiment score and the topic-word offset value. The intuition behind is that if a word tends to occur in documents with positive polarities, but the brand-associated sentiment score is negative, then the topic-word offset value will have an opposite sign, forcing the occurrence count of such a word to be reduced. Furthermore, for each document, we can sample its word counts from their corresponding Poisson distributions and form a document representation which is subsequently fed into a sentiment classifier to predict its sentiment label. If we reverse the sign of the latent brand-associated sentiment score and sample the word counts again, then the sentiment classifier fed with the resulting document representation should generate an opposite sentiment label.

Our proposed BTM is partly inspired by the recently developed Text-Based Ideal Point (TBIP) model (Vafa et al., 2020) in which the topic-specific word choices are influenced by the ideal points of authors in political debates. However, TBIP is fully unsupervised and when used in customer reviews, it generates topics with mixed polarities. On the contrary, BTM makes use of the document-level sentiment labels and is able to produce better separated polarity-bearing topics. As will be shown in the experiments section, BTM outperforms TBIP on brand ranking, achieving a better balance of topic coherence and topic uniqueness measures.

The contributions of the model are three-fold:

- We propose a novel model built on Poisson Factorisation with adversarial learning for brand topic analysis which can disentangle the sentiment factor from the semantic latent representations to achieve a flexible and controllable topic generation;
- We approximate word count sampling from Poisson distributions by the Gumbel-Softmax-based word sampling technique, and construct document representations based on the sampled word counts, which can be fed into a sentiment classifier, allowing for end-to-end learning of the model;
- The model, trained with the supervision of review ratings, is able to automatically infer the brand polarity scores from review text only.

The rest of the paper is organised as follows. Section 2 presents the related work. Section 3 describes our proposed Brand-Topic Model. Section 4 and 5 discusses the experimental setup and evaluation results, respectively. Finally, Section 5 concludes the paper and outlines the future research directions.

2 Related Work

Our work is related to the following research:

Poisson Factorisation Models Poisson factorisation is a class of non-negative matrix factorisation in which a matrix is decomposed into a product of matrices. It has been used in many personalise application such as personalised budgets recommendation (Guo et al., 2017), ranking (Kuo et al., 2018), or content-based social recommendation (Su et al., 2019; de Souza da Silva et al., 2017).

Poisson factorisation can also be used for topic modelling where a document-word count matrix is factorised into a product of two positive matrices, a document-topic matrix and a topic-word matrix (Gan et al., 2015; Jiang et al., 2017). In such a setup, a word count in a document is assumed drawn from a Poisson distribution with its rate parameter defined as a product of a document-specific topic intensity and its word probability under the corresponding topic, summing over all topics.

Polarity-bearing Topics Models Early approaches to polarity-bearing topics extraction were built on LDA in which a word is assumed to be generated from a corpus-wide sentiment-topic-word distributions (Lin and He, 2009). In order to be able to separate topics bearing different polarities, word prior polarity knowledge needs to be incorporated into model learning. In recent years, the neural network based topic models have been proposed for many NLP tasks, such as information retrieval (Xie et al., 2015), aspect extraction (He, 2017) and sentiment classification (He et al., 2018). Most of them are built upon Variational Autoencode (VAE) (Kingma and Welling, 2014) which constructs a neural network to approximate the topic-word distribution in probabilistic topic models (Srivastava and Sutton, 2017; Sønderby et al., 2016; Bouchacourt et al., 2018). Intuitively, training the VAE-based supervised neural topic models with class labels (Chaidaroon and Fang, 2017; Huang et al., 2018; Gui et al., 2020) can introduce sentiment information into topic modelling, which may generate better features for sentiment classification.

Market/Brand Topic Analysis The classic LDA can also be used to analyse market segmentation and brand reputation in various fields such as finance and medicine (Barry et al., 2018; Doyle and Elkan, 2009). For market analysis, the model proposed by Iwata et al. (2009) used topic tracking to analyse customers' purchase probabilities and trends without storing historical data for inference at the current time step. Topic analysis can also be combined with additional market information for recommendations. For example, based on user profiles and item topics, Gao et al. (2017) dynamically modelled users' interested items for recommendation. Zhang et al. (2015) focused on brand topic tracking. They built a dynamic topic model to analyse texts and images posted on Twitter and track competitions in the luxury market among given brands, in which topic words were used to identify recent hot topics in the market (e.g. *Rolex watch*) and brands over topics were used to identify the market share of each brand.

Adversarial Learning Several studies have explored the application of adversarial learning mechanics to text processing for style transferring (John et al., 2019), disentangling representations (John et al., 2019) and topic modelling (Masada and Takasu, 2018). In particular, Wang et al. (2019) has proposed an Adversarial-neural Topic Model (ATM) based on the Generative Adversarial Network (GAN) (Goodfellow et al., 2014), that employees an adversarial approach to train a generator network producing word distributions indistinguishable from topic distributions in the training set. (Wang et al., 2020) further extended the ATM model with a Bidirectional Adversarial Topic (BAT) model, using a bidirectional adversarial training to incorporate a Dirichlet distribution as prior and exploit the information encoded in word embeddings. Similarly, (Hu et al., 2020) builds on the aforementioned adversarial approach adding cycle-consistent constraints.

Although the previous methods make use of adversarial mechanisms to approximate the posterior distribution of topics, to the best of our knowledge, none of them has so far used adversarial learning to lead the generation of topics based on their sentiment polarity and they do not provide any mechanism for smooth transitions between topics, as introduced in the presented Brand-Topic Model.

3 Brand-Topic Model (BTM)

We propose a probabilistic model for monitoring the assessment of various brands in the beauty market from Amazon reviews. We extend the Text-Based Ideal Point (TBIP) model with adversarial learning and Gumbel-Softmax to construct document features for sentiment classification. The overall architecture of our proposed BTM is shown in Figure 2. In what follows, we will first give a brief introduction of TBIP, followed by the presentation of our proposed BTM.

3.1 Background: Text-Based Ideal Point (TBIP) model

TBIP (Vafa et al., 2020) is a probabilistic model which aims to quantify political positions (i.e. ideal points) from politicians' speeches and tweets via Poisson factorisation. In its generative processes, political text is generated from the interactions of several latent variables: the per-document topic intensity θ_{dk} for K topics and D documents, the V-vectors representing the topics β_{kv} with vocabulary size $|V|$, the author's ideal point s expressed with a real-valued scalar x_s and the ideological topic expressed by a real-valued V-vector η_k. In particular, the ideological topic η_k aligns the neutral topic (e.g. *gun*, *abortion*, etc.) according to the author's ideal point (e.g. *liberal*, *neutral*, *conservative*), thus modifying the prominent words in the original topic (e.g. '*gun violence*', or '*constitutional rights*'). The observed variables are the author a_d for a document d, and the word count for a term v in d encoded as c_{dv} .

The TBIP model places a Gamma prior on $\boldsymbol{\beta}$

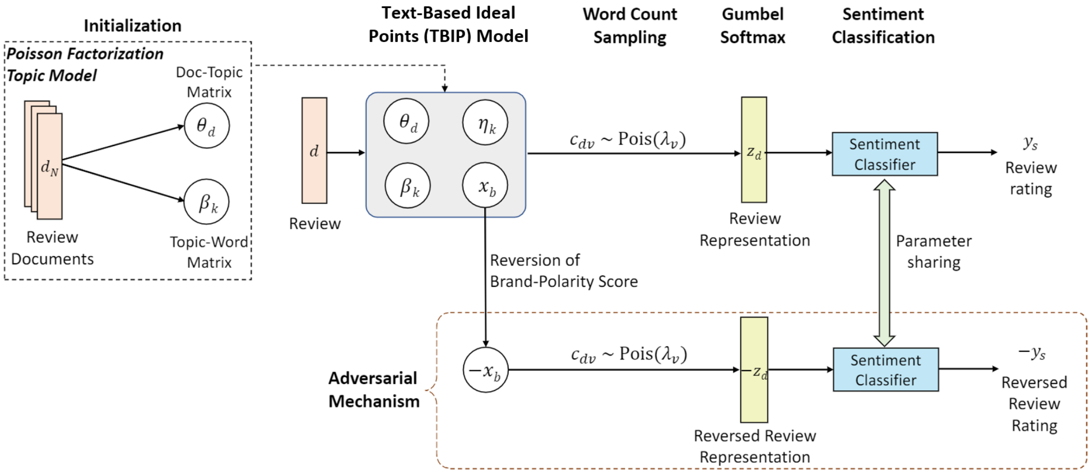


Figure 2: The overall architecture of the Brand-Topic Model.

and $\boldsymbol{\theta}$, which is the assumption inherited from the Poisson factorisation, with m, n being hyperparameters.

$$\theta_{dk} \sim \mathrm{Gamma}(m, n) \quad \beta_{kv} \sim \mathrm{Gamma}(m, n)$$

It places instead a normal prior over the ideological topic $\boldsymbol{\eta}$ and ideal point $\boldsymbol{x}$:

$$\eta_{kv} \sim \mathcal{N}(0, 1) \quad x_s \sim \mathcal{N}(0, 1)$$

The word count for a term v in d, c_{dv}, can be modelled with Poisson distribution:

$$c_{dv} \sim \mathrm{Pois}(\sum_k \theta_{dk} \beta_{kv} \exp\{x_{a_d} \eta_{kv}\}) \qquad (1)$$

3.2 Brand-Topic Model (BTM)

Inspired by the TBIP model, we introduce the Brand-Topic Model by reinterpreting the ideal point x_s as brand-polarity score x_b expressing an *ideal feeling* derived from reviews related to a brand, and the ideological topics η_{kv} as *opinionated topics*, i.e. polarised topics about brand qualities.

Thus, a term count c_{dv} for a product's reviews derives from the hidden variable interactions as $c_{dv} \sim Pois(\lambda_{dv})$ where:

$$\lambda_{dv} = \sum_k \theta_{dk} \exp\{\log \beta_{kv} + x_{b_d} \eta_{kv}\}) \qquad (2)$$

with the priors over $\boldsymbol{\beta}$, $\boldsymbol{\theta}$, $\boldsymbol{\eta}$ and $\boldsymbol{x}$ initialised according to the TBIP model.

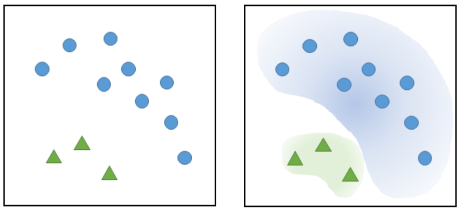
The intuition is that if a word tends to frequently occur in reviews with positive polarities, but the brand-polarity score for the current brand is negative, then the occurrence count of such a word would be reduced since x_{b_d} and η_{kv} have opposite signs.

Distant Supervision and Adversarial Learning Product reviews might contain opinions about products and more general users' experiences (e.g. delivery service), which are not strictly related to the product itself and could mislead the inference of a reliable brand-polarity score. Therefore, to generate topics which are mainly characterised by product opinions, we provide an additional distant supervision signal via their review ratings. To this aim, we use a sentiment classifier, a simple linear layer, over the generated document representations to infer topics that are discriminative of the review's rating.

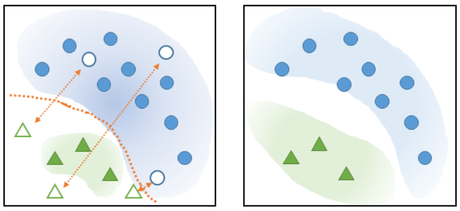
In addition, to deal with the imbalanced distribution in the reviews, we design an adversarial mechanism linking the brand-polarity score to the topics as shown in Figure 3. We contrastively sample adversarial training instances by reversing the original brand-polarity score ($x_b \in [-1, 1]$) and generating associated representations. This representation will be fed into the shared sentiment classifier with the original representation to maximise their distance in the latent feature space.

Gumbel-Softmax for Word Sampling As discussed earlier, in order to construct document features for sentiment classification, we need to sample word counts from the Poisson distribution. However, directly sampling word counts from the Poisson distribution is not differentiable. In order to enable back-propagation of gradients, we apply Gumbel-Softmax (Jang et al., 2017; Joo et al., 2020), which is a gradient estimator with the reparameterization trick.

For a word v in document d, its occurrence count,



(a) Imbalanced distribution. (b) Biased estimation.



(c) Contrastive sampling. (d) Adjusting by AL.

Figure 3: Process of Adversarial Learning (AL): (a) The imbalanced distribution of different sentiment categories; (b) The biased estimation of distribution from training samples; (c) Contrastive sample generation (white triangles) by reversing the sampling results from biased estimation (white dots); (d) Adjusting the biased estimation of (b) by the contrastive samples.

$c_{dv} \sim \text{Pois}(\lambda_{dv})$, is a non-negative random variable with the Poisson rate λ_{dv}. We can approximate it by sampling from the truncated Poisson distribution, $c_{dv_n} \sim \text{TruncatedPois}(\lambda_{dv}, n)$, where

$$\pi_k = Pr(c_{dv} = k) = \frac{\lambda_{dv}^k e^{-\lambda_{dv}}}{k!}$$

$$\pi_{n-1} = 1 - \sum_k \pi_k \quad \text{for} \quad k \in \{0, 1, ..., n-2\}.$$

We can then draw samples z_{dv} from the categorical distribution with class probabilities $\pi = (\pi_0, \pi_1, \cdots, \pi_{n-1})$ using:

$$u_i \sim \text{Uniform}(0, 1) \quad g_i = -\log(-\log(u_i))$$

$$w_i = \text{softmax}\big((g_i + \log \pi_i)/\tau\big) \quad z_{dv} = \sum_i w_i c_i$$

where τ is a constant referred to as the temperature, c is the outcome vector. By using the average of weighted word account, the process is now differentiable and we use the sampled word counts to form the document representation and feed it as an input to the sentiment classifier.

Objective Function Our final objective function consists of three parts, including the Poisson factorisation model, the sentiment classification loss, and the reversed sentiment classification loss (for adversarial learning). For the Poisson factorisation modelling part, mean-field variational inference is used to approximate posterior distribution (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017).

$$q_\phi(\theta, \beta, \eta, x) = \prod_{d,k.b} q(\theta_d) q(\beta_k) q(\eta_k) q(x_b) \quad (3)$$

For optimisation, to minimise the approximation of $q_\phi(\theta, \beta, \eta, x)$ and the posterior, equivalently we maximise the evidence lower bound (ELBO):

$$\begin{aligned} ELBO = \mathbb{E}_{q_\phi}[log\ p(\theta, \beta, \eta, x)] + \\ log\ p(y|\theta, \beta, \eta, x) - log\ q_\phi(\theta, \beta, \eta, x)] \end{aligned} \quad (4)$$

The Poisson factorization model is pre-trained by applying the algorithm in Gan et al. (2015), which is then used to initialise the varational parameters of θ_d and β_k. Our final objective function is:

$$Loss = ELBO + \lambda(L_s + L_a) \quad (5)$$

where L_s and L_a are the cross entropy loss of sentiment classification for sampled documents and reversed sampled documents, respectively, and λ is the weight to balance the two parts of loss, which is set to be 100 in our experiments.

4 Experimental Setup

Datasets We construct our dataset by retrieving reviews in the Beauty category from the Amazon review corpus[1] (He and McAuley, 2016). Each review is accompanied with the rating score (between 1 and 5), reviewer name and the product meta-data such as product ID, description, brand and image. We use the product meta-data to relate a product with its associated brand. By only selecting brands with relatively more and balanced reviews, our final dataset contains a total of 78,322 reviews from 45 brands. Reviews with the rating score of 1 and 2 are grouped as negative reviews; those with the score of 3 are neutral reviews; and the remaining are positive reviews. The statistics of our dataset is shown in Table 1[2]. We can observe that our data is highly imbalanced, with the positive reviews far more than negative and neutral reviews.

Baselines We compare the performance of our model with the following baselines:

---

[1] `http://jmcauley.ucsd.edu/data/amazon/`

[2] The detailed rating score distributions of brands and their average rating are shown in Table A1 in the Appendix.

Dataset	**Amazon-Beauty Reviews**
Documents per classes	
Neg / Neu / Pos	9,545 / 5,578 / 63,199
Brands	45
Total #Documents	78,322
Avg. Document Length	9.7
Vocabulary size	~ 5000

Table 1: Dataset statistics of reviews within the Amazon dataset under the *Beauty* category.

- Joint Sentiment-Topic (JST) model (Lin and He, 2009), built on LDA, can extract polarity-bearing topics from text provided that it is supplied with the word prior sentiment knowledge. In our experiments, the MPQA subjectivity lexicon[3] is used to derive the word prior sentiment information.
- SCHOLAR (Card et al., 2018), a neural topic model built on VAE. It allows the incorporation of meta-information such as document class labels into the model for training, essentially turning it into a supervised topic model.
- Text-Based Ideal Point (TBIP) model, an unsupervised Poisson factorisation model which can infer latent brand sentiment scores.

Parameter setting Since documents are represented as the bag-of-words which result in the loss of word ordering or structural linguistics information, frequent bigrams and trigrams such as '*without doubt*', '*stopped working*', are also used as features for document representation construction. Tokens, i.e., n-grams ($n = \{1, 2, 3\}$), occurred less than twice are filtered. In our experiments, we set aside 10% reviews (7,826 reviews) as the test set and the remaining (70,436 reviews) as the training set. For hyperparameters, we set the batch size to 1,024, the maximum training steps to 50,000, the topic number to 30, the temperature in the Gumbel-Softmax equation in Section 3.2 to 1. Since our dataset is highly imbalanced, we balance data in each mini-batch by oversampling. For a fair comparison, we report two sets of results from the baseline models, one trained from the original data, the other trained from the balanced training data by oversampling negative reviews. The latter results in an increased training set consisting of 113,730 reviews.

5 Experimental Results

In this section, we will present the experimental results in comparison with the baseline models in brand ranking, topic coherence and uniqueness measures, and also present the qualitative evaluation of the topic extraction results. We will further discuss the limitations of our model and outline future directions.

5.1 Comparison with Existing Models

Model	**Spearman's**		**Kendall's tau**	
	corr	p-val	corr	p-val
JST	0.241	0.111	0.180	0.082
JST*	0.395	0.007	0.281	0.007
SCHOLAR	-0.140	0.358	-0.103	0.318
SCHOLAR*	0.050	0.743	0.046	0.653
TBIP	0.361	0.016	0.264	0.012
BTM	**0.486**	0.001	**0.352**	0.001

Table 2: Brand ranking results generated by various models based on the test set. We report the correlation coefficients *corr* and its associated two-sided p-values for both Spearman's correlations and Kendall's tau. * indicates models trained on balanced training data.

Brand Ranking We report in Table 2 the brand ranking results generated by various models on the test set. The two commonly used evaluation metrics for ranking tasks, Spearman's correlations and Kendall's Tau, are used here. They penalise inversions equally across the ranked list. Both TBIP and BTM can infer each brand's associated polarity score automatically which can be used for ranking. For both JST and SCHOLAR, we derive the polarity score of a brand by aggregating the sentiment probabilities of its associated review documents and then normalising over the total number of brand-related reviews. It can be observed from Table 2 that JST outperforms both SCHOLAR and TBIP. Balancing the distributions of sentiment classes improves the performance of JST and SCHOLAR. Overall, BTM gives the best results, showing the effectiveness of adversarial learning.

Topic Coherence and Uniqueness Here we choose the top 10 words for each topics to calculate the context-vector-based topic coherence scores (Röder et al., 2015). In the topics generated by TBIP and BTM, we can vary the topic polarity scores to generate positive, negative and neutral

---

[3] `https://mpqa.cs.pitt.edu/lexicons/`

subtopics as shown in Table 4. We would like to achieve high topic coherence, but at the same time maintain a good level of topic uniqueness across the sentiment subtopics since they express different polarities. Therefore, we additionally consider the topic uniqueness (Nan et al., 2019) to measure word redundancy among sentiment subtopics, $TU = \frac{1}{LK}\sum_{l=1}^{K}\sum_{l=1}^{L}\frac{1}{cnt(l,k)}$, where $cnt(l,k)$ denotes the number of times word l appear across *positive*, *neutral* and *negative* topics under the same topic number k. We can see from Table 3 that both TBIP and BTM achieve higher coherence scores compared to JST and SCHOLAR. TBIP slightly outperforms BTM on topic coherence, but has a lower topic uniqueness score. As will be shown in Table 4, topics extracted by TBIP contain words significantly overlapped with each other among sentiment subtopics. SCHOLAR gives the highest topic uniqueness score. However, it cannot separate topics with different polarities. Overall, our proposed BTM achieves the best balance between topic coherence and topic uniqueness.

Model	Topic Coherence	Topic Uniqueness
JST	0.1423	0.7699
JST*	0.1317	0.7217
SCHOLAR	0.1287	**0.9640**
SCHOLAR*	0.1196	0.9256
TBIP	**0.1525**	0.8647
BTM	0.1407	0.9033

Table 3: Topic coherence/uniqueness measures of results generated by various models.

5.2 Example Topics Extracted from Amazon Reviews

We illustrate some representative topics generated by TBIP and BTM in Table 4. It is worth noting that we can generate a smooth transition of topics by varying the topic polarity score gradually as shown in Figure 1. Due to space limit, we only show topics when the topic polarity score takes the value of -1 (*negative*), 0 (*neutral*) and 1 (*positive*). It can be observed that TBIP fails to separate subtopics bearing different sentiments. For example, all the subtopics under 'Duration' express a positive polarity. On the contrary, BTM shows a better-separated sentiment subtopics. For 'Duration', we see positive words such as '*comfortable*' under the positive subtopic, and words such as '*stopped working*' clearly expressing negative sentiment under the negative subtopic. Moreover, top words under different sentiment subtopics largely overlapped with each other for TBIP. But we observe a more varied vocabulary in the sentiment subtopics for BTM.

TBIP was originally proposed to deal with political speeches in which speakers holding different ideal points tend to use different words to express their stance on the same topic. This is however not the case in Amazon reviews where the same word could appear in both positive and negative reviews. For example, '*cheap*' for lower-priced products could convey a positive polarity to express value for money, but it could also bear a negative polarity implying a poor quality. As such, it is difficult for TBIP to separate words under different polarity-bearing topics. On the contrary, with the incorporation of adversarial learning, our proposed BTM is able to extract different set of words co-occurred with '*cheap*' under topics with different polarities, thus accurately capturing the contextual polarity of the word '*cheap*'. For example, '*cheap*' appears in both positive and negative subtopics for 'Brush' in Table 4. But we can find other co-occurred words such as '*pretty*' and '*soft*' under the positive subtopic, and '*plastic*' and '*flimsy*' under the negative subtopic, which help to infer the contextual polarity of '*cheap*'.

TBIP also appears to have a difficulty in dealing with highly imbalanced data. In our constructed dataset, positive reviews significantly outnumber both negative and neutral ones. In many sentiment subtopics extracted by TBIP, all of them convey a positive polarity. One example is the 'Duration' topic under TBIP, where words such as '*great*', '*great price*' appear in all positive, negative and neutral topics. With the incorporation of supervised signals such as the document-level sentiment labels, our proposed BTM is able to derive better separated polarised topics.

As an example shown in Figure 1, if we vary the polarity score of a topic from -1 to 1, we observe a smooth transition of its associated topic words, gradually moving from negative to positive. Under the topic (*shaver*) shown in this figure, four brand names appeared: REMINGTON, NORELCO, BRAUN and LECTRIC SHAVE. The first three brands can be found in our dataset. REMINGTON appears in the negative side and it indeed has the lowest review score among these 3 brands; NORELCO appears most and it is indeed a popular

Topic Label	Sentiment Topics	Top Words
		BTM
Brush	Positive	brushes, cheap, came, pay, *pretty*, brush, *okay*, case, glue, *soft*
	Neutral	cheap, feel, set, buy, *cheaply made*, feels, made, worth, spend, bucks
	Negative	plastic, made, cheap, parts, feels, *flimsy*, money, *break*, metal, bucks
Oral Care	Positive	teeth, taste, mouth, strips, crest, mouthwash, tongue, using, *white*, rinse
	Neutral	teeth, *pain*, mouth, strips, using, taste, used, crest, mouthwash, *white*
	Negative	*pain*, *issues*, causing, teeth, caused, removing, wore, *burn*, little, cause
Duration	Positive	stay, pillow, *comfortable*, string, tub, mirror, stick, back, months
	Neutral	months, year, *lasted*, *stopped working*, *sorry*, n, worked, working, u, last
	Negative	months, year, last, *lasted*, battery, warranty, *stopped working*, *died*, *less*
		TBIP
Brush	Positive	love, *favorite*, products, *definitely recommend*, forever, carry, brushes
	Neutral	love, brushes, *cute*, *favorite*, *definitely recommend*, soft, *cheap*
	Negative	love, brushes, cute, *soft*, *cheap*, set, case, quality price, buy, bag
Oral Care	Positive	teeth, strips, crest, mouth, mouthwash, taste, *white*, *whitening*, sensitivity
	Neutral	teeth, strips, mouth, crest, taste, work, *pain*, using, *white*, mouthwash
	Negative	teeth, strips, mouth, crest, taste, work, *pain*, using, *white*, mouthwash
Duration	Positive	great, *love shampoo*, *great price*, *great product*, *lasts long time*
	Neutral	*great*, *great price*, *lasts long time*, *great product*, price, *works expected*
	Negative	quality, *great*, *fast shipping*, *great price*, *low price*, price quality, hoped

Table 4: Example topics generated by BTM and TBIP on Amazon reviews. The topic labels are assigned by manual inspection. Positive words are highlighted with the blue colour, while negative words are marked with the red colour. BTM generates better-separated sentiment topics compared to TBIP.

brand with mixed reviews; and Braun gets the highest score in these 3 brands, which is also consistent with the observations in our data. Another interesting finding is the brand Lectric Shave, which is not one of the brands we have in the dataset. But we could predict from the results that it is a product with relatively good reviews.

5.3 Limitations and Future work

Our model requires the use of a vanilla Poisson factorisation model to initialise the topic distributions before applying the adversarial learning mechanism of BTM to perform a further split of topics based on varying polarities. Essentially topics generated by a vanilla Poisson factorisation model can be considered as parent topics, while polarity-bearing subtopics generated by BTM can be considered as child topics. Ideally, we would like the parent topics to be either neutral or carrying a mixed sentiment which would facilitate the learning of polarised sub-topics better. In cases when parent topics carry either strongly positive or strongly negative sentiment signals, BTM would fail to produce polarity-varying subtopics. One possible way is to employ earlier filtering of topics with strong polarities. For example, topic labeling (Bhatia et al., 2016) could be employed to obtain a rough estimate of initial topic polarities; these labels would be in turn used for filtering out topics carrying strong sentiment polarities.

Although the adversarial mechanism tends to be robust with respect to class imbalance, the disproportion of available reviews with different polarities could hinder the model performance. One promising approach suitable for the BTM adversarial mechanism would consist in decoupling the representation learning and the classification, as suggested in Kang et al. (2020), preserving the original data distribution used by the model to estimate the brand score.

6 Conclusion

In this paper, we presented the Brand-Topic Model, a probabilistic model which is able to generate polarity-bearing topics of commercial brands. Compared to other topic models, BMT infers real-valued brand-associated sentiment scores and extracts fine-grained sentiment-topics which vary smoothly in a continuous range of polarity scores. It builds on the Poisson factorisation model, combining it with an adversarial learning mechanism to induce better-separated polarity-bearing topics. Experimental evaluation on Amazon reviews against several baselines shows an overall improvement of topic quality in terms of coherence, uniqueness and separation of polarised topics.

Acknowledgements

This work is funded by the EPSRC (grant no. EP/T017112/1, EP/V048597/1). YH is supported by a Turing AI Fellowship funded by the UK Research and Innovation (UKRI) (grant no. EP/V020579/1).

References

Adam E. Barry, Danny Valdez, Alisa A.Padon, and Alex M. Russel. 2018. Alcohol advertising on twitter—a topic model. *American Journal of Health Education*, pages 256–263.

Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. 2016. Automatic labelling of topics with neural embeddings. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 953–963.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(2003):993–1022.

Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. 2018. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. In *The 32nd AAAI Conference on Artificial Intelligence*, pages 2095–2102.

Dallas Card, Chenhao Tan, and Noah A. Smith. 2018. Neural models for documents with metadata. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2031–2040.

Suthee Chaidaroon and Yi Fang. 2017. Variational deep semantic hashing for text documents. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 75–84.

G. Doyle and C. Elkan. 2009. Financial topic models. In *In NIPS Workshop for Applications for Topic Models: Text and Beyond*.

Zhe Gan, Changyou Chen, Ricardo Henao, David E. Carlson, and Lawrence Carin. 2015. Scalable deep poisson factor analysis for topic modeling. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1823–1832.

Li Gao, Jia Wu, Chuan Zhou, and Yue Hu. 2017. Collaborative dynamic sparse topic regression with user profile evolution for item recommendation. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1316–1322.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680.

Lin Gui, Leng Jia, Jiyun Zhou, Ruifeng Xu, and Yulan He. 2020. Multi-task mutual learning for joint sentiment classification and topic detection. In *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

Yunhui Guo, Congfu Xu, Hanzhang Song, and Xin Wang. 2017. Understanding users' budgets for recommendation with hierarchical poisson factorization. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1781–1787.

Ruidan He. 2017. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 388–397.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1121–1131.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*.

Xuemeng Hu, Rui Wang, Deyu Zhou, and Yuxuan Xiong. 2020. Neural topic modeling with cycle-consistent adversarial training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Minghui Huang, Yanghui Rao, Yuwei Liu, Haoran Xie, and Fu Lee Wang. 2018. Siamese network-based supervised topic modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4652–4662.

Tomoharu Iwata, Shinji Watanabe, Takeshi Yamada, and Naonori Ueda. 2009. Topic tracking model for analyzing consumer purchase behavior. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1427–1432.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*.

Haixin Jiang, Rui Zhou, Limeng Zhang, Hua Wang, and Yanchun Zhang. 2017. A topic model based on poisson decomposition. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1489–1498.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 424–434.

Weonyoung Joo, Dongjun Kim, Seungjae Shin, and Il-Chul Moon. 2020. Generalized gumbel-softmax gradient estimator for various discrete random variables. *Computing Research Repository*, arXiv:2003.01847v2.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2020. Decoupling representation and classifier for long-tailed recognition. In *the 8th International Conference on Learning Representations*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations*.

Li-Yen Kuo, Chung-Kuang Chou, and Ming-Syan Chen. 2018. Personalized ranking on poisson factorization. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 720–728.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 375–384.

Tomonari Masada and Atsuhiro Takasu. 2018. Adversarial learning for topic models. In *Proceedings of the 14th International Conference on Advanced Data Mining and Applications*, volume 11323 of *Lecture Notes in Computer Science*, pages 292–302.

Feng Nan, Ran Ding, Ramesh Nallapati, , and Bing Xiang. 2019. Topic modeling with wasserstein autoencoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, page 6345–6381.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *The 8th ACM International Conference on Web Search and Data Mining*, pages 399–408.

Eliezer de Souza da Silva, Helge Langseth, and Heri Ramampiaro. 2017. Content-based social recommendation with poisson matrix factorization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 10534 of *Lecture Notes in Computer Science*, pages 530–546.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder variational autoencoders. In *The Annual Conference on Neural Information Processing Systems*, pages 3738–3746.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *International Conference on Learning Representations*.

Yijun Su, Xiang Li, Wei Tang, Daren Zha, Ji Xiang, and Neng Gao. 2019. Personalized point-of-interest recommendation on ranking with poisson factorization. In *International Joint Conference on Neural Networks*, pages 1–8.

Keyon Vafa, Suresh Naidu, and David M. Blei. 2020. Text-based ideal points. In *Proceedings of the 2020 Conference of the Association for Computational Linguistics*, pages 5345–5357.

Martin J Wainwright and Michael I Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.

Rui Wang, Xuemeng Hu, Deyu Zhou, Yulan He, Yuxuan Xiong, Chenchen Ye, and Haiyang Xu. 2020. Neural topic modeling with bidirectional adversarial training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 340–350.

Rui Wang, Deyu Zhou, and Yulan He. 2019. ATM: Adversarial-neural topic model. *Information Processing & Management*, 56(6):102098.

Pengtao Xie, Yuntian Deng, and Eric P. Xing. 2015. Diversifying restricted boltzmann machine for document modeling. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1315–1324.

Hao Zhang, Gunhee Kim, and Eric P. Xing. 2015. Dynamic topic modeling for monitoring market competition from online text and image data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1425–1434.

A Appendix

Brand	**Average Rating**	**Number of Reviews**	**Distribution of Ratings**				
			1	**2**	**3**	**4**	**5**
General	3.478	1103	236	89	144	180	454
VAGA	3.492	1057	209	116	133	144	455
Remington	3.609	1211	193	111	149	282	476
Hittime	3.611	815	143	62	110	154	346
Crest	3.637	1744	352	96	159	363	774
ArtNaturals	3.714	767	138	54	65	143	368
Urban Spa	3.802	1279	118	105	211	323	522
GiGi	3.811	1047	151	79	110	184	523
Helen Of Troy	3.865	3386	463	20	325	472	1836
Super Sunnies	3.929	1205	166	64	126	193	666
e.l.f	3.966	1218	117	85	148	241	627
AXE PW	4.002	834	85	71	55	169	454
Fiery Youth	4.005	2177	208	146	257	381	1185
Philips Norelco	4.034	12427	1067	818	1155	2975	6412
Panasonic	4.048	2473	276	158	179	419	1441
SilcSkin	4.051	710	69	49	58	135	399
Rimmel	4.122	911	67	58	99	160	527
Avalon Organics	4.147	1066	111	52	82	145	676
L'Oreal Paris	4.238	973	88	40	72	136	651
OZ Naturals	4.245	973	79	43	74	142	635
Andalou Naturals	4.302	1033	58	57	83	152	683
Avalon	4.304	1344	132	62	57	108	985
TIGI	4.319	712	53	32	42	93	492
Neutrogena	4.331	1200	91	55	66	142	846
Dr. Woods	4.345	911	60	42	74	83	652
Gillette	4.361	2576	115	94	174	555	1638
Jubujub	4.367	1328	53	42	132	238	863
Williams	4.380	1887	85	65	144	347	1246
Braun	4.382	2636	163	85	147	429	1812
Italia-Deluxe	4.385	1964	96	73	134	336	1325
Booty Magic	4.488	728	28	7	48	144	501
Greenvida	4.520	1102	55	33	51	108	855
Catrice	4.527	990	49	35	34	99	773
NARS	4.535	1719	60	36	107	237	1279
Astra	4.556	4578	155	121	220	608	3474
Heritage Products	4.577	837	25	18	52	96	646
Poppy Austin	4.603	1079	36	31	38	115	859
Aquaphor	4.633	2882	100	58	106	272	2346
KENT	4.636	752	23	8	42	74	605
Perfecto	4.801	4862	44	36	81	523	4178
Citre Shine	4.815	713	17	5	3	43	645
Bath & Body Works	4.819	2525	60	27	20	95	2323
Bonne Bell	4.840	1010	22	9	6	35	938
Yardley	4.923	788	3	4	3	31	747
Fruits & Passion	4.932	776	3	2	3	29	739
Overall	4.259	78322	5922	3623	5578	12322	50877

Table A1: Brand Statistics. The table shows the average rating score, the total number of associated reviews, and the distribution of the number of reviews for ratings ranging between 1 star to 5 stars, for each of the 45 brands.

Lexical Normalization for Code-switched Data and its Effect on POS Tagging

Rob van der Goot
IT University of Copenhagen
`robv@itu.dk`

Özlem Çetinoğlu
IMS, University of Stuttgart
`ozlem@ims.uni-stuttgart.de`

Abstract

Lexical normalization, the translation of non-canonical data to standard language, has shown to improve the performance of many natural language processing tasks on social media. Yet, using multiple languages in one utterance, also called code-switching (CS), is frequently overlooked by these normalization systems, despite its common use in social media. In this paper, we propose three normalization models specifically designed to handle code-switched data which we evaluate for two language pairs: Indonesian-English (Id-En) and Turkish-German (Tr-De). For the latter, we introduce novel normalization layers and their corresponding language ID and POS tags for the dataset, and evaluate the downstream effect of normalization on POS tagging. Results show that our CS-tailored normalization models outperform Id-En state of the art and Tr-De monolingual models, and lead to 5.4% relative performance increase for POS tagging as compared to unnormalized input. [1]

1 Introduction

Social media provide an invaluable source of information for natural language processing (NLP) systems. Its informative and spontaneous nature leads to many interesting phenomena, like non-standard words, spelling errors and abbreviations. One particularly challenging and interesting phenomenon is the use of multiple languages within the same utterance, which is also called code-switching (CS) (Gumperz, 1982; Myers-Scotton, 1995; Toribio and Bullock, 2012).

Because most NLP models are designed to process canonical and monolingual data, their performance drops enormously when having to process social media data (Eisenstein, 2013). One solution to this problem is lexical normalization: the translation of non-standard (e.g. social media) text to its canonical form (Han and Baldwin, 2011). Previous work has shown that by standardizing the data, we can improve the robustness of NLP systems (Derczynski et al., 2013; Zhang et al., 2013; van der Goot and van Noord, 2017). Nevertheless these systems overlook code-switching. (1) shows a code-switched tweet (upper) and its normalization annotation (lower), taken from an Indonesian-English CS corpus (Barik et al., 2019) (Indonesian in bold). This example demonstrates that CS complicates normalization, because it can be unclear in which language to normalize (e.g., *ak* is normalized to *aku* 'I' in Indonesian. English-only normalization systems would probably normalize it to *ok*).

(1)	**ak**	.	luv	u	:(	till	die
	aku	.	love	you	:(	till	die

'*I love you till (I) die*'

Recently, there has been an increasing interest in the automatic processing of CS data, however, there has not been much work on its lexical normalization. To the best of our knowledge, only Adouane et al. (2019) focus entirely on lexical normalization for CS data in their work. For other works, normalization is a preprocessing step for downstream tasks: chunking (Sharma et al., 2016), parsing (Bhat et al., 2017, 2018), or machine translation (Barik et al., 2019). These CS normalizers are either rule-based and language-specific (Barik et al., 2019) or combine (Hindi) back-transliteration and normalization (Sharma et al., 2016; Bhat et al., 2017, 2018) thus, they are not directly applicable to other lexical normalization datasets. In this work,

- We are the first to present open-source normalization models specialized for CS lexical normalization without any language-specific components.

---

[1] Source code is available at: `https://bitbucket.org/robvanderg/csmonoise`. The Turkish-German data is available at: `https://github.com/ozlemcek/TrDeNormData`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2352–2365
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

- We provide a novel lexical normalization dataset by annotating a Turkish-German Twitter corpus (Çetinoğlu, 2016). We also align existing annotation layers – language IDs (LID) and part-of-speech (POS) tags – to normalization annotations.

- We evaluate three CS normalization models on two language pairs (Turkish-German (Tr-De), Indonesian-English (Id-En)). For both datasets, CS models reach performance in a similar range as monolingual models reach on monolingual datasets.

- Our CS-tailored normalization models outperform Id-En state of the art and set the state of the art for the Tr-De dataset.

- We show that our proposed normalization models improve the performance of POS taggers. For a broad perspective, we employ a variety of taggers (CRF, BiLSTM, BERT).

2 Related Work

Lexical normalization Traditionally, social media normalization approaches can be broadly divided into two types. The first stream of work uses techniques borrowed from machine translation (Aw et al., 2006; Pennell and Liu, 2011; Ljubešic et al., 2016). The second stream is based on a classic spelling correction framework (noisy-channel models) (Han, 2014). Here, they often apply three steps, detecting which words need to be replaced, generating candidates, and ranking these candidates. Later, it became evident that a two-step approach is sufficient (Jin, 2015; van der Goot, 2019), and the detection step was alleviated by considering the original word as a normalization candidate.

The current state-of-the-art model for most languages is MoNoise (van der Goot, 2019), which is based on this two-step approach. A variety of modules are used for the generation of candidates. For the ranking, MoNoise complements features from the generation step with additional features, which are all combined in a random forest classifier that predicts the probability that a candidate is a 'correct' candidate. MoNoise is described in more detail in Section 4.2. More recently, sequence-to-sequence models (Lourentzou et al., 2019) and contextual embeddings (Muller et al., 2019) have been used for the lexical normalization task. These approaches have been shown to reach performances close to MoNoise on English benchmarks.

Like most NLP tasks, most research on normalization has been done on English datasets (Han and Baldwin, 2011; Baldwin et al., 2015). However, there has been some efforts on other languages, where usually only one language is considered, we refer to Sharf and Rahman (2017) and van der Goot (2019) for an overview of available resources.

Processing of code-switched social media data Early work on normalizing CS data focused on Hindi-English, as part of pipelines to achieve downstream tasks (Sharma et al., 2016; Bhat et al., 2017, 2018). As Hindi is Romanized in datasets and additional Hindi resources are in the Devanagari script, they include back-transliteration into the normalization step, thus defining the task beyond the scope of this paper. Nevertheless, all systems report a positive impact of normalization on their final task.

More recently, Barik et al. (2019) experiment on normalization for Indonesian-English. They use a rule-based approach supplemented by clusters derived from word embeddings, and show that normalization can be used to improve machine translation. Adouane et al. (2019) instead propose to use sequence-to-sequence models for normalizing Algerian Arabic data mixed with Modern Standard Arabic, French, Berber, and English. They show that their edit distance-based token-level aligner helps improve normalization.

When annotating the Tr-De dataset for normalization, we also adapted its POS tags (see Section 3.1). This gives us the opportunity to apply POS tagging as extrinsic evaluation. Besides research on Hindi-English that combines normalization and back-transliteration, most work either use normalization to improve tagging performance of monolingual social media data (Derczynski et al., 2013; van der Goot et al., 2017), or on POS tagging of CS data without normalization (AlGhamdi et al., 2016; Soto and Hirschberg, 2018). In this work, we combine these angles.

Because some of our proposed normalization models depend on language labels, we require a word-level language identification system. There is a wide variety of approaches used for this task, where early systems mostly used CRFs (Sequiera et al., 2015; Molina et al., 2016). More recently, neural networks based approaches have shown superior performance for this task (Zhang et al., 2018). We opt for three different architectures to observe the effect of the quality of language identification on normalization (Section 4.1).

	Seg+CS			Norm		Tok+Anon
Tokens	Semester§da	-yım		Semesterdayım		semesterdayim
LID	Mixed	TR	⇒	Mixed	⇒	Mixed
POS	NOUN	VERB		VERB		VERB

Figure 2: Mapping LID and POS tags from `Seg+CS` to `Norm` to `Tok+Anon` for the mixed word *Semesterdayım* 'I am in semester'.

`Raw:`	@Erkan1903 nerdee 3 **semester**dayim dha.					
`Tok+Anon:`	@username	nerdee	3	**semester**dayim	dha	.
`Norm`	@username	Nerde	3.	**Semester**dayım	daha	.
`Seg+CS:`	@username Nerde 3. **Semester**§da -yım daha .					

Figure 1: Different annotation layers of a tweet from the Tr-De corpus, meaning '*No way, I am still in the 3rd semester*'. The German part is in bold. `Raw`: downloaded tweet; `Tok+Anon`: after tokenization and anonymization; `Norm`: after normalization; `Seg+CS`: after segmentation (e.g. the Turkish copular *-yim*), and CS boundaries (§) in `Mixed` tokens. The token alignment and normalization tasks are carried out on the `Tok+Anon` and `Norm` pairs.

3 Data

In this section we first describe the design decisions of the novel Turkish-German dataset, then we compare some basic statistics together with the existing Indonesian-English dataset (Barik et al., 2019).

3.1 Turkish-German code-switched normalization corpus

We use the Turkish-German Twitter corpus from Çetinoğlu (2016) in our experiments. It consists of 17K tokens as 1,029 tweets. The raw tweets of the corpus have undergone three main steps of alternations after the collection: tokenization, normalization, and segmentation.[2] In addition, usernames and URLs are anonymized as `@username` and `[url]` respectively, and intra-word CS boundaries are marked in `Mixed` tokens with §. Each alternation layer is exemplified on a sentence from the corpus in Figure 1.

The `Seg+CS` layer is annotated with language IDs and POS tags (Çetinoğlu and Çöltekin, 2016). The LID tag set consists of `TR` (Turkish), `DE` (German), `Lang3` (third language), `Mixed` (intra-word CS), `NE` (named entity), `Ambig` (both Turkish and German and cannot be disambiguated in given context), `Other` (punctuation, numbers, URLs, emoticons, symbols). Additionally, named entities are tagged with their language label next to the `NE` tag, e.g. 'Germany' is annotated in the corpus as follows depending on the language: *Almanya* `NE.TR`, *Deutschland* `NE.DE`, *Germany* `NE.Lang3`. The POS annotation adopts the Universal Dependencies (UD) tag set (Nivre et al., 2016).

Preprocessing for normalization The original version of the corpus has only the `Raw` and `Seg+CS` layers and only tweet-level alignment between them. As our work focuses only on normalization we created the intermediate layers `Tok+Anon` and `Norm` that leave out other tasks. Since MoNoise requires word-aligned annotations, we also provided these alignments.

We anonymized and tokenized the raw tweets to achieve the `Tok+Anon` layer. For tokenization, we use a slightly modified version of twokenize.py[3] (O'Connor et al., 2010). To obtain the `Norm` layer, we merged back segmented tokens and removed CS boundaries on the `Seg+CS` layer.

After this stage, we aligned `Tok+Anon` and `Norm` on the token level automatically using Giza++ (Och and Ney, 2003). We parsed the resulting alignment files to align the actual tokens and corrected them manually. There are 15,715 1:1, 520 1:n, and 147 n:1 alignments.

LID and POS alignment The existing LID and POS tags are on the `Seg+CS` layer; since we base our experiments on the `Tok+Anon` layer, we need to map the annotations. This is done in two steps following the `Seg+CS` ⇒ `Norm` ⇒ `Tok+Anon` order. Due to segmentation merges in the first step, and 1:n and n:1 token alignments in the second step, there are non-trivial LID and POS alignments.

Figure 2 demonstrates a segmented word in the first column. The first segment *Semesterda* 'in semester' is `Mixed` with German *Semester* and Turkish locative case marker *da*. The second segment is the Turkish copular *-yim* 'I am'. Their POS tags are NOUN and VERB, respectively. When segmentation is undone in the second column (`Norm`), their LID and POS are merged too. If two tokens

---

[2] Morphosyntactic split of words into subwords, cf. (Çetinoğlu and Çöltekin, 2016) for details.

[3] `github.com/brendano/tweetmotif`

	#words	%norm	% split	%merge	CMI
Id-En	18,758	14.13	1.33	0.17	28.20
Tr-De	13,217	25.97	3.01	1.04	22.44

Table 1: Descriptive normalization and code-switching statistics on the training split of the datasets. CMI is the code-mixing index (Das and Gambäck, 2014), averaged over all training sentences. %norm reflect the percentage of words which is normalized.

Source	Indonesian	English	Turkish	German
Wikipedia	75	2,162	55	776
Twitter	510	5,018	203	89

Table 2: Size of raw data (in million words) from both data sources.

have the same LID, the merged token takes the same LID. If they are different, the resulting token is `Mixed`, as in the example.

POS tag merging rules can get more complicated, therefore, we used a heuristic that favors the POS tag of the second token in most cases.[4] When a NOUN segment is merged with a VERB segment, as in Figure 2 (`Seg+CS` ⇒ `Norm`), the merged token is assigned a VERB POS tag. For the `Norm` ⇒ `Tok+Anon` mapping, the alignment is 1:1, thus LID and POS are directly carried over.

3.2 Dataset characteristics

Besides the data described in the previous section, we use the Indonesian-English (Id-En) data from (Barik et al., 2019). The Id-En data is only annotated with language IDs and uses three labels: `ID`, `EN`, `UN` (Unspecified), whereas the Tr-De includes 12 labels (Section 3.1). To simplify the models and improve comparability, we map the language labels of the Tr-De dataset to `TR`, `DE` and `UN`. Named entities are mapped to their respective language tags, e.g, `NE.DE` to `DE`. `Mixed` tokens are mapped to `DE` as they are German words with Turkish inflection. `Lang3`, `Ambig` and `Other` are mapped to `UN`.

We divide both datasets into a train and test split (80-20%), and omit a development set due to small sizes. Since we want to leave test set out in analyses, we opt for 10-fold cross-validation on the training split of the data in experiments. Statistics of the training splits of the datasets are shown in Table 1. The datasets are relatively small, but a high ratio of words is normalized, including a high percentage of splits and merges. The percentage of in-vocabulary words is especially low in the Tr-De data, which is mainly due to the morphological richness of Turkish. The code-mixing index (CMI) (Das and Gambäck, 2014) indicates the (average of the) amount of words not written in the majority language for each sentence. The relatively high CMI for both datasets indicates a high frequency of code-switching occurs in the data.

In both datasets there are a small amount of sentences without normalization (8 and 76 for respectively Id-En and Tr-De), which might be desirable for evaluation of (over)normalization, as in a real-world setup one also does not know beforehand whether normalization is necessary. In more than half of the sentences the number of normalized words is larger than 3. Furthermore, there are some sentences (5-10 per dataset) with a very high normalization ratio (>70%), which are all in capitals.

3.3 Monolingual Data

Our baseline model (MoNoise) exploits monolingual data from both the source and the target domain (canonical data) to train word embeddings and estimate n-gram probabilities. To this end, we utilize Wikipedia dumps from 01-01-2020 and random tweets collected throughout 2012 and 2018 from the Twitter API, filtered by the FastText language classifier (Joulin et al., 2017). We tokenized this data based on whitespaces, and removed all duplicate sentences/tweets. The sizes of the collected raw datasets are shown in Table 2.

4 Models

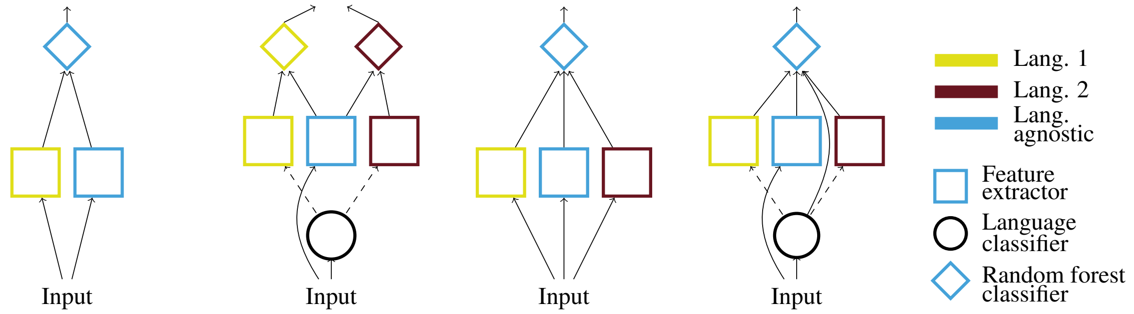
In this section we describe the models used for word-level language identification (4.1), lexical normalization (4.2) and POS tagging (4.3).

4.1 Word-level language identification

We treat language identification as a sequence labeling task where the label of each word is a language ID. We evaluate three sequence labeling libraries: 1) MarMoT (Mueller et al., 2013), a higher-order conditional random fields tagger 2) Bilty (Plank et al., 2016), a BiLSTM tagger, also incorporating character level information 3) a BERT-based (Devlin et al., 2019) tagger named MaChAmp (van der Goot et al., 2021). For Bilty, we project polyglot embeddings (Al-Rfou et al., 2013) of each

---

[4] Turkish is agglutinative. Segmentation often happens by splitting derivational suffixes that bear the final POS tag.



(a) `Monolingual` MoNoise. (b) `Fragment` based normalization. (c) `Multilingual` model. (d) `Language-aware` multilingual model.

Figure 3: Overview of the different proposed variations of MoNoise. Dashed lines mean that only one of the two paths is taken, decided by the language identification. For model (a), there can be two versions, one with features from Lang. 1 (shown here) and one based on Lang. 2.

language of the language pairs to the same space using MUSE (Lample et al., 2018), whereas for MaChAmp, we use multilingual BERT.[5] We use the default settings for all toolkits.

4.2 Normalization

We choose to use MoNoise (van der Goot, 2019) as a baseline and starting point for our proposed models for two main reasons: 1) Normalization annotation for code-switched data is scarce, and MoNoise is specifically strong in low-resource setups because of its dependence on external resources (generated from raw data); 2) It is the only normalization model that has shown to be effective in multiple languages. Below we first introduce the standard monolingual MoNoise model, and then all the proposed extensions which are focused on code-switched data. A schematic overview of all models is shown in Figure 3.

Monolingual (Figure 3a)
MoNoise consists of two parts, a candidate generation step and a candidate ranking step. For the generation of candidates, a spelling correction system (Aspell),[6] word embeddings and a dictionary based on the training data are used. Features from these modules are then supplemented with n-gram probabilities based on Wikipedia and Twitter data and other features indicating whether a word is present in the Aspell dictionary, whether it contains an alphabetical character, the length of a candidate compared to the original word, and whether it starts with a capital. For the novel proposed models, we will split up the features based on whether they require *language-specific* resources (spelling correction, word embeddings and n-grams features; yellow and red in Figure 3), or whether they are *language-agnostic* (all other features; blue in Figure 3). For the ranking of the candidates a random forest classifier (Breiman, 2001) is used, which predicts the probability whether a candidate is correct. An obvious disadvantage when applying monolingual MoNoise on CS-data is that many features are language-specific (e.g. spelling correction, word embeddings, n-grams), which is sub-optimal for tokens from another language. Since our datasets and evaluation include capitals, we use the version of MoNoise including capitalization handling (van der Goot et al., 2020).

Fragments (Figure 3b)
The baseline model has the deficiency that it has the language-specific features only for one language, while normalizing texts for two languages. An intuitive way of improving this model would be to split up the input data into monolingual fragments, and train two separate monolingual models. The fragments are created by splitting the data on every CS point, where words with the `UN` label are converted to the label of the previous word. This setup has the advantage that the normalization model itself does not need any adaptation, and it can thus be used with any normalization model. The disadvantages are that it is dependent on a language label, two separate classifiers have to be trained and the context is interrupted.

Multilingual (Figure 3c)
Instead of using two separate random forest classifiers, we can exploit both feature sets simultaneously in one classifier. This means that for every

---

[5] multi_cased_L-12_H-768_A-12
[6] `www.aspell.net`

language-specific feature, we now have two features. In this setup, the model is not explicitly informed about the language of input words, however, some of the features (especially n-gram probabilities) will have a very high correlation with this information. This model has the advantage that only one classifier has to be trained, and no language labels are necessary. It has the disadvantage that it uses more features for the classifier compared to the `Monolingual` and `Fragments` models, which increases the complexity of the classification.

Language-aware (Figure 3d)
Some of the language-specific features of the `Multilingual` model will be rather superfluous for words in the other language. For example, it will search for Turkish words in German word embeddings, and also use n-gram counts based on the German Wikipedia. To avoid this, we can use only one copy of each language-specific feature, and generate them based on the language label (the same language labels as in the `Fragments` model are used). More concretely, this means that for a German word, we will generate uni-gram probabilities based on German data, whereas for Turkish we will use Turkish data; these are then modeled as one feature in the model. On top of this, we also add a feature that indicates which language a word belongs to. There might be some mismatches in the importance of features because different data sources and languages are used. Because the language label is known, and a random forest classifier can model feature interactions intrinsically (Breiman, 2001), these mismatches should not be problematic. This model has the advantage that the number of features stays almost the same as in the `Monolingual` model (+1, the language ID), but a disadvantage is that it requires language labels.

4.3 POS tagging

For POS tagging, we examine the same three sequence labeling systems as used for language identification (Section 4.1): MarMoT, Bilty and MaChAmp. For each normalization setting, we normalize the input data, and use this normalized text as input for the POS tagger, which is trained on canonical data.

5 Evaluation

In this section we evaluate each of the three subtasks (LID, normalization, POS), where for the latter two we also examine the effect of exploiting the prediction of the previous tasks. Unless mentioned otherwise, we report the results of 10-fold cross-validation on the training split of the data. For all experiments, we use a paired bootstrap test on the sentence level with 1,000 samples to test significance. For all results, we order the models by the complexity of the implementation as compared to MoNoise (first fragments, as the original model can be used as a black box, then multilingual because it does not need a language classifier, and finally the language-aware model). An * next to results denotes a significant difference for $p < 0.05$, of a model always as compared to the previous model (corresponding to the previous column in Table 6, the previous row in other tables) for the same data.

Model	Id-En	Tr-De
MarMoT	92.71	92.91
Bilty	*93.81	*94.31
MaChAmp	*95.17	*95.67

Table 3: Word level accuracies for language identification (10-fold).

5.1 Language identification

Results for the language identification task are reported in Table 3. Unsurprisingly, the performances are in line with the chronological order of the introduction of the systems, and their computational complexity. It should be noted that for MaChAmp we used pre-trained embeddings which were trained on the largest amount of external data. When inspecting the performance per language label, we saw that the '`UN`specified' is by far the most difficult. Even though this class contains punctuation, it also contains many harder cases, where a word belongs to any language other than Lang1 and Lang2, or when the annotator is uncertain. Barik et al. (2019) use a conditional random fields classifier with a variety of features for this task, and report 90.11 accuracy for the full Id-En dataset in a 5-fold cross-validation setting. Which, despite differences in data splits, confirms that our results are competitive.

5.2 Normalization

For lexical normalization, a wide variety of evaluation metrics is used in the literature, ranging from accuracy (Han and Baldwin, 2011), F1 score (Baldwin et al., 2015) and precision over out-

Model	Id-En	Tr-De
LAI	73.24	74.03
MFR	*88.35	*78.57
Monolingual-lang1 (Tr/Id)	*94.76	*79.81
Monolingual-lang2 (De/En)	*94.31	80.58
Fragments	94.73	*81.24
Multilingual	94.84	*81.74
Language-aware	94.79	81.68

Table 4: Normalization performance of the baselines and the proposed models (10-fold accuracy). For the models dependent on language labels, we used the labels predicted by MaChAmp.

Model	Id-En	Tr-De
Fragments (MarMoT)	94.66	80.77
Fragments (Bilty)	*94.71	*80.89
Fragments (MaChAmp)	94.73	*81.24
Fragments (Gold)	*94.81	81.71
Language-aware (MarMoT)	94.74	81.24
Language-aware (Bilty)	94.76	81.57
Language-aware (MaChAmp)	94.79	81.68
Language-aware (Gold)	94.90	82.18

Table 5: Effect of different language predictions on normalization models (10-fold accuracy).

of-vocabulary words (Alegria et al., 2013), to CER and BLUE score (Ljubešic et al., 2016). Because the word order is fixed in our task, and to ease interpretation of the results, we opt to use simple accuracy on the word level, where we consider all words (i.e., also the unnormalized words).

To interpret the scores, we include three baselines: 1) leave-as-is (LAI), which always outputs the original word, i.e. its accuracy is equivalent to the percentage of words that are not normalized 2) most-frequent-replacement (MFR), which uses the most frequent replacement from the training data for each word 3) monolingual MoNoise, which can be trained on either of the languages within a language pair (two models).

Results for the different models are compared in Table 4. For the Id-En dataset, the differences between all proposed models are small and not significant. Even the monolingual models perform remarkably well, and only small gains are observable when using the multilingual model. We also compared our results to Barik et al. (2019), using their evaluation metric as their model/output was not available. The metric is non-deterministic, as it uses accuracy over unique OOV words.[7] Nevertheless, our average estimated result for `Multilingual` is 69.83 for this metric, outperforming their score of 68.50.

For the Tr-De dataset, the scores are generally lower, indicating that this dataset (and perhaps language pair) is more difficult. Especially now, we can observe that the code-switched adaptations lead to substantially higher scores. To our surprise, `Multilingual` and `Language-aware` perform on par, even though the multilingual model does not rely on language labels. `Fragments` performs significantly worse. This leads to the conclusion that language labels are not directly beneficial for lexical normalization (in this setup). In general, the performances are in a similar range as for monolingual datasets (van der Goot, 2019).[8]

Model behavior Besides the metrics reported in the table, we also examined precision and recall. Precision is generally much higher (1.1 to 3 times, see Appendix B) than recall especially for Tr-De, which is in line with previous observations (van der Goot, 2019). This means that the model is conservative and only replaces cases for which it is rather certain, which arguably is a desirable behavior.

Effect of language predictions To evaluate the effect of the language predictions, we run both the `Fragments` and the `Language-aware` models with all language predictions from Section 5.1 as well as the gold language labels. The results (Table 5) show that the performance of the language identification has a positive effect on the normalization performance. Although it is not significant in most cases, it should be noted that significance is only tested compared to the previous model.

Language labels Looking at the normalization performance breakdown on language labels shows that the gains of our proposed models are consistently smaller on Indonesian and Turkish compared to respectively English and German (see Appendix A for full results). This was to be expected, as for these languages the model has less external

---

[7] Which can be normalized differently dependent on context, we confirmed this with the authors.

[8] van der Goot (2019) used error reduction rate as main evaluation metric, for which the multilingual model would score 80.72 (Id-En) and 30.42 (Tr-De). The reported scores on monolingual datasets are 77.09 for En and 28.94 for Tr.

Model	LAI	Multiling.	Lang.-aware	Gold
MarMoT–POS	61.92	*65.50	65.47	*69.14
Bilty–POS	*65.23	*67.99	*68.26	*72.04
MaChAmp–POS	65.60	*68.25	68.13	*71.27

Table 6: Accuracies for Tr-De POS tagging, using a variety of normalization strategies.

data (Section 3.3) and while the model was originally not evaluated for Indonesian, Turkish had the lowest performance in van der Goot (2019).

Qualitative analysis Both `Multilingual` and `Language-aware` correct most frequent normalization mistakes well. This means for Id-En, abbreviations (*yg* ↦ *yang* 'which is' in `ID`), slang words (*gw* ↦ *saya*, *gue* ↦ *saya* 'I' in `ID`), phonetic spelling (*kalo* ↦ *kalau* 'if' in `ID`); and for Tr-De emoticons, restoring Turkish-specific characters, restoring vowels (*cnm* ↦ *canım* 'my dear' in `TR`), and punctuation replacements. On the Id-En data, however, there is a higher number of these frequent replacements compared to the Tr-De dataset, which explains the high scores and small variability for Id-En in Table 4 and 5.

For the Tr-De dataset, the most common mistakes include: not correcting capitalization in the beginning of a sentence, merging of words, monolingual ambiguous cases depending on context (*mi* ↦ [*mi*, *mı*], question clitics in `TR`), and tokenization and punctuation mistakes (?:D ↦? :D). In comparison, for the Id-En dataset, the models make rather different errors: in-vocabulary words which should be normalized are left as is (*kaya* ↦ *seperti*, *usah* ↦ *perlu*), normalizations which are lexically very distant are not found (*lw* ↦ *kamu*), and English contractions are often not replaced (*isnt* ↦ *is not*). Error analysis on the Id-En dataset revealed that correction of capitalization was annotated inconsistently. However, because in most cases the normalization was lowercased, this did not have a large effect on performance.

Interestingly, `Language-aware` is better in correcting words that exist in both languages. For instance, *ne* is the informal form of *eine* 'a/one' in German, and also means 'what' in Turkish. The dataset annotations expect the *ne* ↦ *eine* normalization. While `Multilingual` fails to do so, `Language-aware` corrects them. We believe language IDs play a positive role here in defining the context, and although in general both models perform on par, if a dataset contains many such ambiguous words, `Language-aware` could be preferable.

5.3 POS tagging

For POS tagging, we only look at Tr-De as Id-En is not annotated with POS tags. We employ a pipeline approach; we first normalize our training data in a 10-fold setting, and then apply the tagger on this normalized data. The taggers are trained on a shuffled concatenation of the Turkish-IMST (Sulubacak et al., 2016) and German-GSD (McDonald et al., 2013) datasets of UD version 2.5 (Nivre et al., 2020). Now that none of the CS data is used during training, 10-fold cross-validation is not necessary. We directly apply the taggers on the full training data. This way the exact same data split is used for evaluation as in the 10-fold setting in the previous sections. Even though we have POS tags available for the gold normalization (Section 3.1), we do not have gold tags for predicted normalization, and to keep the comparison fair we evaluate using the `Tok+Anon` POS tags. When a word is split or merged, we use the alignment and check whether the correct tag is present. In other words: we select one tag based on an oracle selection.[9]

Results in Table 6 show that, surprisingly, Bilty performs competitive to MaChAmp across most settings. Considering the differences between the normalization models, the `Multilingual` model and the `Language-aware` model perform on par, but there is still a marginal gap compared to the gold normalization.

We also analyzed the confusion matrices of the POS tagger, the full analysis can be found in Appendix C, we will shortly summarize findings here. 1) Bilty is mainly outperforming MaChAmp in gold due to better recognition of symbols (emojis), 2) Bilty is more sensitive to different normalization strategies, whereas for MaChAmp the differ-

---

[9] It should be noted that this makes splitting beneficial, and this metric can easily be tricked by splitting every token so it should be used with caution. However, our proposed normalization models have a low rate of splitting (114 versus 398 in gold) and merging is not handled at all.

	Normalization		POS
Model	Id-En	Tr-De	Tr-De
LAI	74.03	67.02	60.77
Monolingual (Id/De)	*94.62	76.33	*63.47
Multilingual	94.27	*78.28	*64.06
Language-aware	94.32	77.83	*63.92
Gold	*100.00	*100.00	*67.75

Table 7: Normalization and POS tagging accuracies on test data. The POS tagging model is the same MaChAmp model for all results, only the normalization strategy for the input changes.

ences between them are minimal, 3) Performance on nouns improves a lot after normalization, especially for German (due to corrected capitalization of nouns), 4) The second POS tag which improved most are verbs, investigation showed that this is mainly because Turkish-specific characters are replaced by their ASCII counterparts, which helps the tagger assign the correct POS.

5.4 Test data

On the test data we take both the 'no normalization' and the best baseline (which are monolingual Indonesian for Id-En and monolingual German for Tr-De), and compare these to our best two proposed normalization models. The results in Table 7 show that, parallel to 10-fold cross-validation results (Table 4), `Multilingual` and `Language-aware` scores are similar and their difference is insignificant for both datasets. This leads to the conclusion that `Multilingual` is the most elegant model, as it is not dependent on language labels. On the Tr-De dataset the proposed models are clearly outperforming the baselines. However, on the Id-En dataset the differences are small (and not significant) between the monolingual model and both of our proposed models.

For Tr-De, we take the test set normalized by systems in the second column of Table 7 and apply MaChAmp for POS tagging. The results in the third column show that the POS tagger follows the trend in normalization scores, and performs slightly better when using the multilingual model, beating the LAI baseline (i.e. not using normalization) with 5.4% relative improvement.

6 Conclusion

Code-switching provides many challenges for NLP systems. In this work we attempt to overcome some of these challenges by normalizing the data, and evaluating the downstream effect of this for POS tagging. For evaluation we use an Indonesian-English dataset (Barik et al., 2019) as well as a German-Turkish dataset (Çetinoğlu, 2016), for which we provided novel normalization layers and adapted existing LID and POS annotation.

We proposed three different models to normalize CS data. The two best-performing models are `Language-aware` and `Multilingual`. The first model exploits language labels, to identify for which language to generate features, whereas the second model combines features for both languages. The differences in performance between these two systems was not significant for any of the 10-fold experiments nor on the test data, so in most cases the multilingual model would be preferable, as it has no dependence on language labels.

We showed that normalizing the input before POS tagging results in significantly higher POS accuracies for CS data. Gold normalization experiments showed that there is still room for improvement for normalization models to help POS tagging.

An interesting property of the proposed model is that it does not have to be trained on intrasentential CS data. In fact, it can be trained on a mix of two monolingual datasets, thereby handling many more language pairs. We hope to evaluate this setup if resources (i.e., normalization test data for a CS language pair, and monolingual normalization training data for both languages) become available.

Acknowledgements

We would like to thank Barbara Plank, Alan Ramponi, Marija Stepanovic, and Agnieszka Falenska, for feedback on early drafts. We also thank Manuel Mager and Sevde Ceylan with their help on Tr-De data alignment and Anab Maulana Barik for sharing the Id-En data. The first author is partially funded by an Amazon Research Award. The second author is funded by DFG via project CE 326/1-1 "Computational Structural Analysis of German-Turkish Code-Switching" (SAGT).

References

Wafia Adouane, Jean-Philippe Bernardy, and Simon Dobnik. 2019. Normalising non-standardised orthography in Algerian code-switched user-generated data. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 131–140, Hong Kong, China. Association for Computational Linguistics.

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria. Association for Computational Linguistics.

Inaki Alegria, Nora Aranberri, Víctor Fresno, Pablo Gamallo, Lluis Padró, Inaki San Vicente, Jordi Turmo, and Arkaitz Zubiaga. 2013. Introducción a la tarea compartida Tweet-Norm 2013: Normalización léxica de tuits en español. In *Tweet-Norm@ SEPLN*, pages 1–9.

Fahad AlGhamdi, Giovanni Molina, Mona Diab, Thamar Solorio, Abdelati Hawwari, Victor Soto, and Julia Hirschberg. 2016. Part of speech tagging for code switched data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 98–107, Austin, Texas. Association for Computational Linguistics.

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 33–40, Sydney, Australia. Association for Computational Linguistics.

Timothy Baldwin, Marie Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, Beijing, China. Association for Computational Linguistics.

Anab Maulana Barik, Rahmad Mahendra, and Mirna Adriani. 2019. Normalization of Indonesian-English code-mixed twitter data. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 417–424, Hong Kong, China. Association for Computational Linguistics.

Irshad Bhat, Riyaz A. Bhat, Manish Shrivastava, and Dipti Sharma. 2017. Joining hands: Exploiting monolingual treebanks for parsing of code-mixing data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 324–330, Valencia, Spain. Association for Computational Linguistics.

Irshad Bhat, Riyaz A. Bhat, Manish Shrivastava, and Dipti Sharma. 2018. Universal Dependency parsing for Hindi-English code-switching. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 987–998, New Orleans, Louisiana. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Özlem Çetinoğlu. 2016. A Turkish-German code-switching corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4215–4220, Portorož, Slovenia. European Language Resources Association (ELRA).

Özlem Çetinoğlu and Çağrı Çöltekin. 2016. Part of speech annotation of a Turkish-German code-switching corpus. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, pages 120–130, Berlin, Germany. Association for Computational Linguistics.

Amitava Das and Björn Gambäck. 2014. Identifying languages at the word level in code-mixed Indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 378–387, Goa, India. NLP Association of India.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 198–206, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

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

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 359–369, Atlanta, Georgia. Association for Computational Linguistics.

Rob van der Goot. 2019. MoNoise: A multi-lingual and easy-to-use lexical normalization tool. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 201–206, Florence, Italy. Association for Computational Linguistics.

Rob van der Goot and Gertjan van Noord. 2017. Parser adaptation for social media by integrating normalization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 491–497, Vancouver, Canada. Association for Computational Linguistics.

Rob van der Goot, Barbara Plank, and Malvina Nissim. 2017. To normalize, or not to normalize: The impact of normalization on part-of-speech tagging. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 31–39, Copenhagen, Denmark. Association for Computational Linguistics.

Rob van der Goot, Alan Ramponi, Tommaso Caselli, Cafagna Michele, and Lorenzo De Mattei. 2020. Norm it! lexical normalization for Italian and its downstream effects for dependency parsing. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*. European Language Resources Association (ELRA).

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive Choice, Ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *2005.14672v3*.

John J Gumperz. 1982. *Discourse strategies*, volume 1. Cambridge University Press.

Bo Han. 2014. *Improving the utility of social media with Natural Language Processing*. Ph.D. thesis, The University of Melbourne.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–378, Portland, Oregon, USA. Association for Computational Linguistics.

Ning Jin. 2015. NCSU-SAS-Ning: Candidate generation and feature engineering for supervised lexical normalization. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 87–92, Beijing, China. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Nikola Ljubešic, Katja Zupan, Darja Fišer, and Tomaz Erjavec. 2016. Normalising Slovene data: historical texts vs. user-generated content. *Bochumer Linguistische Arbeitsberichte*, page 146.

Ismini Lourentzou, Kabir Manghnani, and ChengXiang Zhai. 2019. Adapting sequence to sequence models for text normalization in social media. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 335–345.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Giovanni Molina, Fahad AlGhamdi, Mahmoud Ghoneim, Abdelati Hawwari, Nicolas Rey-Villamizar, Mona Diab, and Thamar Solorio. 2016. Overview for the second shared task on language identification in code-switched data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 40–49, Austin, Texas. Association for Computational Linguistics.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA. Association for Computational Linguistics.

Benjamin Muller, Benoit Sagot, and Djamé Seddah. 2019. Enhancing BERT for lexical normalization. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 297–306, Hong Kong, China. Association for Computational Linguistics.

Carol Myers-Scotton. 1995. *Social motivations for codeswitching: Evidence from Africa*. Oxford University Press.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010. Tweetmotif: Exploratory search and topic summarization for twitter. In *Fourth International AAAI Conference on Weblogs and Social Media*.

Deana Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of SMS abbreviations. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 974–982, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Royal Sequiera, Monojit Choudhury, Parth Gupta, Paolo Rosso, Shubham Kumar, Somnath Banerjee, Sudip Kumar Naskar, Sivaji Bandyopadhyay, Gokul Chittaranjan, Amitava Das, et al. 2015. Overview of FIRE-2015 shared task on mixed script information retrieval. In *FIRE Workshops*, volume 1587, pages 19–25.

Zareen Sharf and Saif Ur Rahman. 2017. Lexical normalization of roman Urdu text. *International Journal of Computer Science and Network Security*, 17(12):213–221.

Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Shrivastava, Radhika Mamidi, and Dipti M. Sharma. 2016. Shallow parsing pipeline - Hindi-English code-mixed social media text. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1340–1345, San Diego, California. Association for Computational Linguistics.

Victor Soto and Julia Hirschberg. 2018. Joint part-of-speech and language ID tagging for code-switched data. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 1–10, Melbourne, Australia. Association for Computational Linguistics.

Umut Sulubacak, Memduh Gokirmak, Francis Tyers, Çağrı Çöltekin, Joakim Nivre, and Gülşen Eryiğit. 2016. Universal Dependencies for Turkish. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3444–3454, Osaka, Japan.

Almeida Jacqueline Toribio and Barbara E Bullock. 2012. *The Cambridge handbook of linguistic code-switching*. Cambridge University Press.

Congle Zhang, Tyler Baldwin, Howard Ho, Benny Kimelfeld, and Yunyao Li. 2013. Adaptive parser-centric text normalization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1159–1168, Sofia, Bulgaria. Association for Computational Linguistics.

Yuan Zhang, Jason Riesa, Daniel Gillick, Anton Bakalov, Jason Baldridge, and David Weiss. 2018. A fast, compact, accurate model for language identification of codemixed text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 328–337, Brussels, Belgium. Association for Computational Linguistics.

Model	`ID`	`EN`	`TR`	`DE`
LAI	66.92	71.33	70.21	66.92
MFR	87.58	88.10	76.26	69.55
Monoling1 (tr/id)	92.71	95.82	78.24	70.17
Monoling2 (de/en)	91.78	95.75	76.81	77.50
Frags	84.71	82.03	70.10	63.92
Multiling.	92.91	95.80	78.27	78.73
Lang-aware	92.77	95.78	78.10	79.32

Table 8: Normalization accuracies per language (with gold language labels)

	Id-En		Tr-De	
Model	recall	precision	recall	precision
LAI	0.00	0.00	0.00	0.00
MFR	57.50	98.20	21.42	84.39
Monoling1 (tr/id)	82.15	97.96	25.55	88.50
Monoling2 (de/en)	80.35	98.03	29.49	87.32
Frags	82.05	97.88	31.15	90.21
Multiling.	82.51	97.87	32.98	90.85
Lang-aware	82.21	97.98	32.95	90.34

Table 9: precision and recall for both datasets, we follow the definitions of (van der Goot, 2019)

Appendix

A Breakdown of performance per language

Table 8 show the accuracy of all the proposed models per language. The LAI scores show that most of the normalization replacements are necessary for `ID` and `DE`. Interestingly, performance of the last two models is highest on respectively `EN` and `DE`, which is probably due to the original model being developed mostly with a focus European languages.

B Precision and recall

Table 9 show the precision and recall of all models on both datasets. LAI has 0.0 on all metrics, because it never finds a correct normalization.

C Confusions of POS taggers

We conducted an analysis of POS tagging confusions for the setting described in Section 5.3. In Table 10 and Table 11 the error frequencies of respectively MaChAmp and Bilty are shown. The tables report the frequency of the top-10 most frequent errors of the baseline (LAI), and the difference in counts observed using a variety of normalization strategies. In Figure 3 and Figure 3 the full confusion matrices for respectively MaChAmp and Bilty are shown. For both of these analyses, we do not report the other baselines, the fragment based model and the MarMot tagger, because performance of these was inferior and this would make comparisons more complex.

	LAI	Multiling.	Lang-aware	Gold
SYM-PUNCT	529	+2	+2	+34
NOUN-PROPN	310	+10	+10	+36
PROPN-NOUN	307	-24	-11	-39
NOUN-ADJ	244	-40	-38	-40
PROPN-PUNCT	220	-5	-8	-16
VERB-NOUN	174	-21	-27	-78
PROPN-ADJ	122	-13	-12	-20
ADV-ADJ	108	-23	-23	-23
ADJ-NOUN	104	-22	-23	-30
ADJ-PROPN	103	-2	+0	+5

Table 10: 10 most common POS tagging errors for LAI baseline, counted for all normalization strategies for the MaChAmp tagger. Counts are all relative compared to the baseline (LAI). The tag on the left is gold, right is predicted.

	LAI	Multiling.	Lang-aware	Gold
PROPN-VERB	507	+67	-417	+65
PROPN-NOUN	310	-95	+138	-205
VERB-NOUN	239	+32	-44	-134
NOUN-ADJ	200	-59	-44	-65
SYM-PUNCT	194	+19	+83	+152
NOUN-PROPN	162	-19	+17	+14
INTJ-NOUN	152	+12	-10	-30
SYM-ADJ	127	-90	-126	-96
ADJ-NOUN	124	+2	-16	-20
NOUN-VERB	122	-12	-27	-39

Table 11: 10 most common POS tagging errors for LAI baseline, counted for all normalization strategies for the Bilty tagger. Counts are all relative compared to the baseline (LAI). The tag on the left is gold, right is predicted.

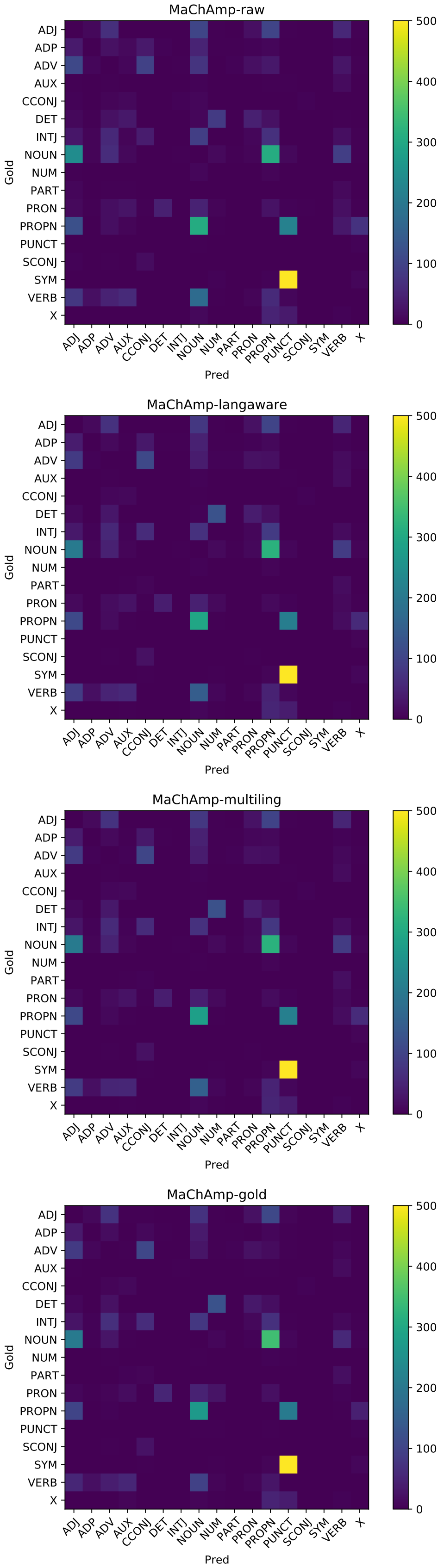


Figure 3: Confusion matrices for MaChAmp using a variety of normalization strategies

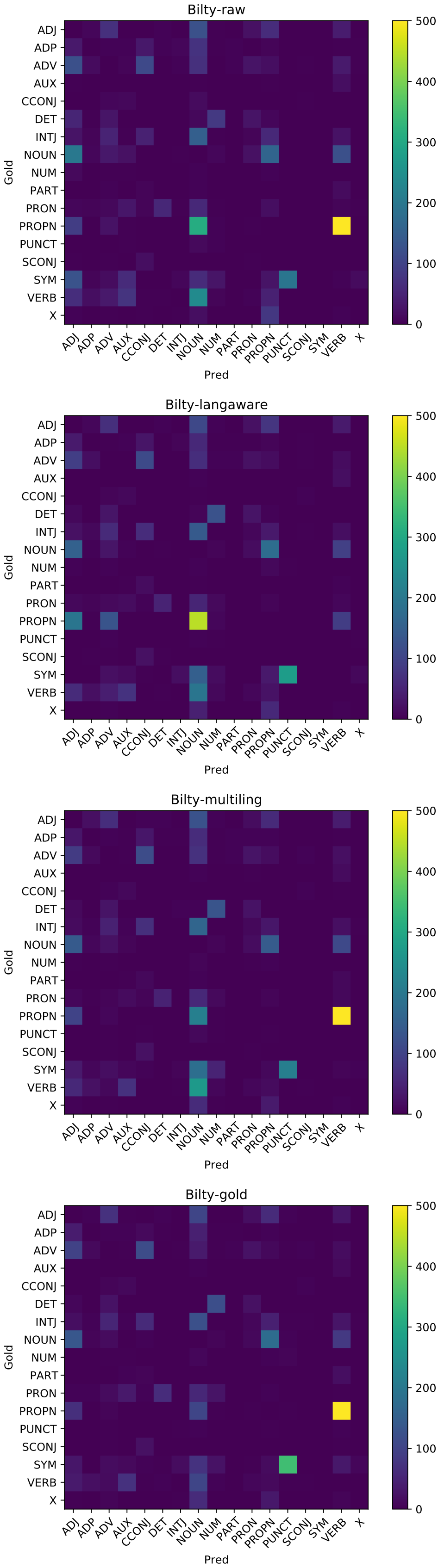


Figure 3: Confusion matrices for Bilty using a variety of normalization strategies

Structural Encoding and Pre-training Matter: Adapting BERT for Table-Based Fact Verification

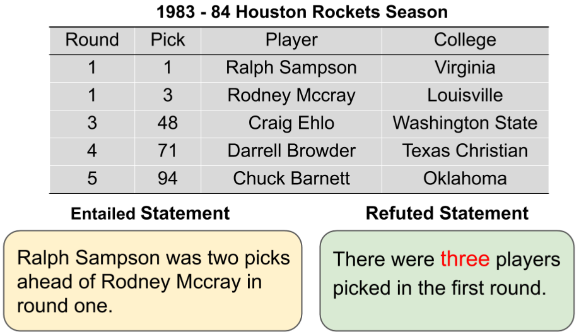
Rui Dong **David A. Smith**
Khoury College of Computer Sciences
Northeastern University
{dongrui, dasmith}@ccs.neu.edu

Abstract

Growing concern with online misinformation has encouraged NLP research on fact verification. Since writers often base their assertions on structured data, we focus here on verifying textual statements given evidence in tables. Starting from the Table Parsing (TAPAS) model developed for question answering (Herzig et al., 2020), we find that modeling table structure improves a language model pre-trained on unstructured text. Pre-training language models on English Wikipedia table data further improves performance. Pre-training on a question answering task with column-level cell rank information achieves the best performance. With improved pre-training and cell embeddings, this approach outperforms the state-of-the-art Numerically-aware Graph Neural Network table fact verification model (GNN-TabFact), increasing statement classification accuracy from 72.2% to 73.9% even without modeling numerical information. Incorporating numerical information with cell rankings and pre-training on a question-answering task increases accuracy to 76%. We further analyze accuracy on statements implicating single rows or multiple rows and columns of tables, on different numerical reasoning subtasks, and on generalizing to detecting errors in statements derived from the ToTTo table-to-text generation dataset.

1 Introduction

The rapid growth in the amount and sources of online textual content has raised concerns about misinformation and its potential harmful impacts on society when quickly spread to a massive audience. For example, a study on enhancing medical education with Wikipedia in 2015 found that 97% of medical students completing the survey disclosed that they found mistakes in Wikipedia medical entries (Herbert et al., 2015). Concerns about misinformation have stimulated extensive research on automatic fact verification, i.e., verifying whether a given textual statement is entailed or refuted by the given evidence.



1983 - 84 Houston Rockets Season

Round	Pick	Player	College
1	1	Ralph Sampson	Virginia
1	3	Rodney Mccray	Louisville
3	48	Craig Ehlo	Washington State
4	71	Darrell Browder	Texas Christian
5	94	Chuck Barnett	Oklahoma

Entailed Statement

Ralph Sampson was two picks ahead of Rodney Mccray in round one.

Refuted Statement

There were three players picked in the first round.

Figure 1: Example from the TabFact dataset: the top table contains the structured evidence; the bottom two boxes contain the statements entailed and refuted by the evidence. Errors are highlighted in red.

While most existing fact verification work focuses on unstructured textual evidence, fact verification with structured evidence is still under-explored. Recently, Chen et al. (2019) introduced a new large-scale dataset, TabFact, for verifying statements based on structured evidence in tables. Figure 1 presents an example table and the corresponding entailed and refuted statements from TabFact. The task of fact verification based on structured evidence is challenging in two aspects. First, traditional language models trained on unstructured text are not directly applicable to learn representations for structured text. It is difficult for such language models to understand a sentence like "*Round Pick Player College 1 1 Ralph Sampson Virginia ...*" by directly concatenating the table cells in Figure 1. Second, detecting misinformation with structured evidence involves not only linguistic inference but also numerical reasoning such as *addition*, *subtraction*, *sorting*, and *counting*

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2366–2375
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

over records. For example, to verify the statement "*Ralph Sampson was two picks ahead of Rodney Mccray*", we need first to find in which order each was picked and subtract them.

Table representation learning is important for utilizing table data as evidence for fact verification. Most existing methods apply BERT (Devlin et al., 2018) model to learn table representations. Table-BERT (Chen et al., 2019) uses simple templates to transform tables into "somewhat natural" sentences, and fine-tunes BERT on pairs of statement sentence and corresponding table "sentence". However, this model adds many extra tokens to the tables, sometimes doubling the length of the original table token sequence. Zhong et al. (2020) propose to first derive logical forms from the table and the statement. A heterogeneous graph is then constructed to capture connections between table cells, functions and arguments and statement tokens. A graph-enhanced contextual representation is learned for each token by only paying attention to the neighbor nodes in the graph when applying BERT model. However, as we show in §4.2, BERT is less effective when it is pre-trained on unstructured data but applied to structured data such as tables.

Table-based fact verification also requires numerical reasoning over table records. Chen et al. (2019) propose a Latent Program Algorithm (LPA), where the statements are parsed into potential programs and a weakly supervised discriminator is trained to assign confidence score to each program. The output from the latent programs are aggregated or ranked according to their confidence score as the final prediction. Zhong et al. (2020) propose to learn a representation for the program with a program-driven neural module network, where semantic compositionality is dynamically modeled along the program parsing structure in a bottom-up style. The program representation and the token representation for the table, statement and linearized program are then combined to make the final prediction. However, these two models depends on weakly-supervised labels, which could be noisy, to derive potential programs. Recently, the same authors of Table-BERT released a new model, GNN-TabFact[1], applying Numerical-aware Graph Neural Networks (NumGNN) on top of Table-BERT to learn to compare numerical cells in the same column. Nonetheless, it requires constructing a graph neural network and conducting iterative message passing among the nodes in the graph.

We propose to adapt the Table Parsing (TAPAS) model (Herzig et al., 2020), which has proven effective in question answering over tables, to model tables for fact verification. TAPAS concatenates all table cells without adding extra tokens and then extends BERT's architecture with additional embedding layers to capture the table structure and numerical comparison information for each token in the table. We replace its original two top classification layers for answer generation with a single classification layer on the `[CLS]` token to classify whether a given statement is entailed or refuted by the table. Our experimental results show that with proper pre-training, TAPAS outpeforms the state-of-the-art GNN-TabFact model, increasing the accuracy of statement classification from 72.2% to 73.9% on the TabFact test set even without modeling numerical information. By further adding ranking information for numerical rows and pre-training on the question answering task, TAPAS achieves 76% accuracy, with 89% on the simple statements and 69.8% on the complex statements. We also perform further analysis to examine: (1) how the numerical comparison information improves TAPAS's performance when ranking information is needed; (2) how the complexity of the training set affects model performance on simple and complex statements; and (3) how well systems trained on TabFact generalize to other fact verification tasks.

This paper's primary contributions are: (1) exploring the effect of table structure modeling on fact verification; (2) measuring the importance of language model pre-training on tabular data; and (3) analyzing the performance of fact-verification models on different numerical reasoning subtasks and errors.

2 Related Work

Fact Verification Thorne et al. (2018) introduce a new dataset for fact extraction and verification (FEVER) with claims generated by altering sentences extracted from Wikipedia. Nie et al. (2019) propose a neural semantic matching network based on a bidirectional LSTM to retrieve related evidence and predict whether the claim is entailed, neutral, or contradicted by the evidence. (Soleimani et al., 2019) propose to adopt a pre-trained BERT model for evidence retrieval and

---

[1] `https://github.com/wenhuchen/GNN-TabFact`

fine tuning it for evidence-claim relation prediction. Jobanputra (2019) propose an unsupervised question-answering based approach for fact checking by generating questions for a claim first and predicting the masked span, which is compared to the ground truth answer for label classification. Zhong et al. (2020) construct two graphs for evidence and claim via semantic role labeling, and use graph-based reasoning for fact checking.

Structured Data Modeling Modeling structured data is essential for multiple tasks, e.g., table classification, table population, table retrieval, question answering , data-to-text generation, and table-based fact verification.

Ghasemi-Gol and Szekely (2018), Trabelsi et al. (2019) and Deng et al. (2019) propose to embed tabular data into a vector space, using table structure to classify tables into different categories, adding more rows or columns to tables and retrieving tables given query keywords. Nishida et al. (2017) employ RNNs to encode cell content and CNNs to encode table structure to better capture semantic features for table classification.

Haug et al. (2018) propose to generate candidate logical forms from a question, convert logical forms to paraphrases, and rank them according to their similarity to the original question to generate the answer for the questions. Herzig et al. (2020) add additional embedding layers to a BERT model to capture table structure and numerical information, and add two classification layers to predict aggregation functions and corresponding table cells to generate an answer for a given question.

Much current table-to-text generation work focus on generating biographies from Wikipedia infoboxes (Lebret et al., 2016; Liu et al., 2018; Sha et al., 2018; Bao et al., 2018) or summarizing basketball games (Wiseman et al., 2017; Puduppully et al., 2019) according to the box- and line-score tables. Parikh et al. (2020) release a dataset for controlled table-to-text generation, where table cells are highlighted for the target sentences. Chen et al. (2020) propose a new natural language generation task, where the model tasked with generating statements that entailed by a given table.

Chen et al. (2019) introduce a dataset for fact verification given tabular data as evidence and propose a BERT-based Table-BERT model and a latent program algorithm (LPA) model for this task. Zhong et al. (2020) propose to first drive the program from the table and statement, and then learn graph-enhanced contextual representations for both the table tokens and the program to classify the statements. After this paper was submitted, four related papers were published, which explore additional questions in table-based fact verification. Zhang et al. (2020) propose to utilize masking in the self attention layer to model table structure. Shi et al. (2020) and Yang et al. (2020) explore how to effectively combine both linguistic information and symbolic information for table-based fact verification. Eisenschlos et al. (2020) generate synthetic datasets to pre-train a TAPAS model to better understand tables for downstream tasks such as table-based fact verification and question answering.

3 Methods

We describe adapting the TAPAS model to fact verification over tables and then introduce different pre-trained models on which we fine-tune the TAPAS for the table verification task.

TAPAS for table-based fact verification Similar to the Table-BERT model, the TAPAS model also flattens the input table into a sequence of tokens. It concatenates all the tokenized table cells in a row and then concatenates all the row sequences into a table sequence. In addition to the position and segment embeddings used in Transformer language models, four additional position embedding layers are introduced for encoding table structure and numerical comparison relations among cells in each column. For each token in the table, there is a column index and a row index indicating the position of the token in the table. For numerical and date columns (as determined by pandas), table cells are sorted to generate a rank index and a inverse rank index for each token according to their position in the sorted column. The sequence of tokens in the statement to be verified is concatenated with the corresponding table token sequence with the `[SEP]` token to indicate where the table sequence starts. For the statement sequence, all the row, column, rank and inverse rank indexes are set as 0. A special token `[CLS]` is added before this entire input sequence for classification purpose. Figure 2 shows how the table and statement from Figure 1 are indexed for each embedding layer. We remove the top two classification layers used in the original TAPAS model for the question answering task, and use the pooled output for the `[CLS]` token for statement classification. To explore the impact of different table encodings, we experiment with two variants of TAPAS: TAPAS-Row-Col, which only utilizes the column and row index embedding, and TAPAS-Row-Col-Rank, which leverages the additional ranking and inverse ranking information for numerical columns.

Pre-training We fine-tune the adapted TAPAS model for the table verification task starting from three different pre-trained models. The first pre-trained model is the widely applied BERT model (Devlin et al., 2018) trained on the BooksCorpus and Wikipedia text. Lists and tables are removed from the training text, so this model is merely trained on unstructured data. The second pre-trained model is the TAPAS-Row-Col-Rank model pre-trained by Herzig et al. (2020) on a large number of Wikipedia text-table pairs with a masked language modeling task. This pre-trained model should be able to better understand the tables by capturing the structure information and numerical information. The third pre-trained model is the TAPAS-Row-Col-Rank model trained on the SQA (Iyyer et al., 2017) dataset with a question answering task. We only utilize their parameters for the bottom layers encoding the text and table sequence. Since this model is trained to generate answers by selecting table cells and predicting the aggregation function to be used on the cells, it should be able to capture more complicated numerical information.

4 Experiments

In this section, we first introduce the experimental settings in detail (§4.1). Then we present the results of comparing two variants of TAPAS model and other state-of-the-art models on the table-based fact verification task in §4.2. More discussions about the models are provided in §4.3.

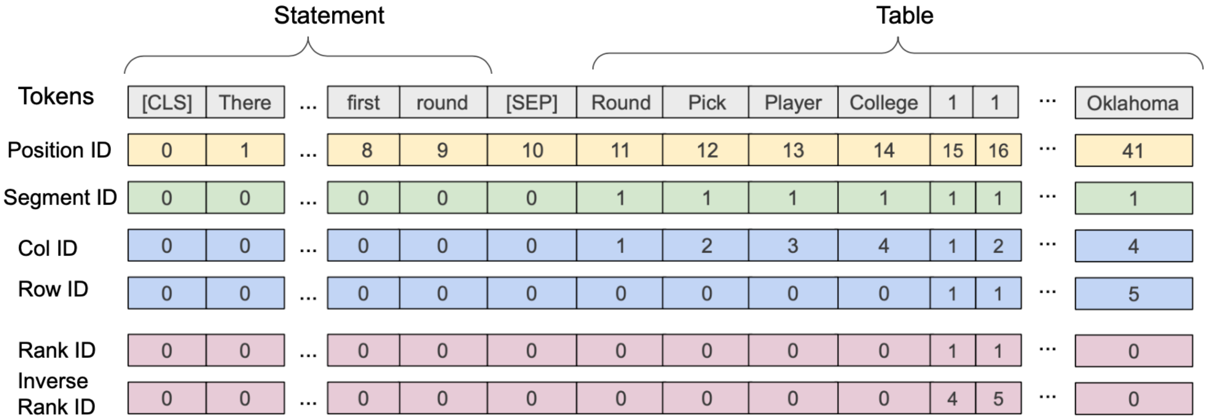
4.1 Experimental Setup

Dataset In the main experiment, we evaluate different models on TabFact, a large-scale dataset for table-based fact verification proposed by Chen et al. (2019). TabFact is generated by asking annotators to write statements given a table and its caption. The statements are either entailed or refuted by the table content. This dataset contains both *simple* and *complex* statements. Simple statements refer to a single row of the table without complicated logical inference and mention the table cells without much modification. Complex statements are more sophisticated and involve aggregation functions such max, min, count, average, difference, etc., over multiple table records. The mentioned table records are rephrased to involve more semantic understanding. A total of 118,275 statement sentences are generated for 16,573 tables, of which 50,244 are simple statements and 68,031 are complex statements. We use the training, validation and test set splits from Chen et al. (2019) to conduct a fair comparison with their models. Table 1 shows the statistics of these splits.

Split	#Statements	#Tables
Train	92,283	13,182
Val	12,792	1,696
Test	12,779	1,695

Table 1: Statistics of the training, validation, and test sets for the TabFact in Chen et al. (2019).

Comparisons We compare TAPAS-Row-Col and TAPAS-Row-Col-Rank model with the following state-of-the-art models:
a. Latent Program Algorithm (LPA). This model (Chen et al., 2019) first uses trigger words to prune pre-defined APIs including around 50 functions such as min, max. Then candidate latent programs are constructed by using breadth-first-search with memoization, and a discriminator is trained with weakly-supervised labels to assign confidence score to the programs. Their best performing model ranks all the latent programs by the discriminator confidence score and make the prediction by executing the top-rated program.
b. Table-BERT. This model (Chen et al., 2019) applies a pre-trained BERT model to classify the concatenated table and statement. Their best performing model first concatenates the header of each column to each table cell in the same column by using the template `[HEADER] is [CELL]`. Then table cells in the i^{th} row are concatenated with the ";" symbol and a short phrase `row` i `is ...` is added to the beginning of the concatenated cells to identify the cells from each row of the table. For example, the first row of the table from Figure 1 would be flatted to "*Row 1 is: Round is 1; Pick is 1; Player is Ralph Sampson; College is Virginia.*". All the row sequences are then concatenated as the table sequence, which is further concatenated to the statement sequence with the `[SEP]` symbol



Tokens	[CLS]	There	...	first	round	[SEP]	Round	Pick	Player	College	1	1	...	Oklahoma
Position ID	0	1	...	8	9	10	11	12	13	14	15	16	...	41
Segment ID	0	0	...	0	0	0	1	1	1	1	1	1	...	1
Col ID	0	0	...	0	0	0	1	2	3	4	1	2	...	4
Row ID	0	0	...	0	0	0	0	0	0	0	1	1	...	5
Rank ID	0	0	...	0	0	0	0	0	0	0	1	1	...	0
Inverse Rank ID	0	0	...	0	0	0	0	0	0	0	4	5	...	0

Figure 2: Example of how TAPAS encoding the refuted statement and the table from Figure 1 for classification.

and fed into BERT model for classification.

c. GNN-TabFact. This is a newly released state-of-the-art model by the creators of TabFact. It first extracts the representations for table tokens and statement tokens based on Table-BERT, then the representations for the tokens in the same cell are averaged as the representation for each table cell. All the table cells with numerical values in the same column are treated as nodes, and a graph is constructed by connecting the nodes with two types of edges—greater and less than—according the values of the cells. A NumGNN layer is then utilized to propagate information among the nodes in the graph to integrate the numerical comparison information into each table cell's representation. Cross attention is then computed over the new table representation and the textual statement for final classification.

d. LogicalFactChecker. This model (Zhong et al., 2020) first derives a program from the table and statement, then learns a contextual representation for the tokens by constructing a heterogeneous graph to capture the connections among the statement, the table and the program. A program-guided neural module network is introduced to learn a representation for the program by capturing its structural and compositional semantics. The token representations and the program representation are then combined to make the final prediction.

Model Parameters Following Table-BERT, all of our models are based on the open-source implementation of BERT with 12-layers, 768-hidden, 12-heads[2]. Both statements and tables are tokenized into sequences of subwords with the Standard BERT tokenizer and joined by the `[SEP]` special token. `[CLS]` is added to the beginning of the joined sequence. We modify the BERT code by adding embedding layers for special token types such as row index, column index and (inverse) rank index information following the TAPAS variants described above modeling for table structure.

Evaluation Metric All models are evaluated for accuracy on classifying test statements as entailed or refuted by the corresponding evidence table.

4.2 Main Results

Table 2 presents our main results, comparing TAPAS-derived models with baseline models on table-based fact verification. As we can see, the TAPAS-Row-Col-Rank model pre-trained on the question answering task over tables achieves the best performance. The TAPAS-Row-Col model pre-trained on WikiTables and fine tuned on TabFact data outperforms the state-of-the-art GNN-TabFact model no matter whether the full table is used or only the subset columns related to the statements are used. (This subsetting gives the best results for GNN-TabFact.)

How do different ways of modeling table structure affect the model performance? We first examine how the method of modeling table structure affects performance. Table-BERT uses a simple template to transform a table into a "somewhat natural" sentence to exploit a language model pre-trained on natural sentences. However, Table 2 shows that TAPAS-Row-Col, which directly concatenates all table cells and uses row index and column index position embedding to capture the table structure, significantly outperforms Table-BERT on the complex test set with the same pre-training,

---

[2] `https://github.com/huggingface/pytorch-pretrained-BERT`

Model	**Pre-train**	**Columns**	Test		
			All	Simple	Complex
BERT classifier w/o Table	BooksCorpus + Wikipedia Text	N/A	50.5	51.0	50.1
Table-BERT	BooksCorpus + Wikipedia Text	subset	65.1	79.1	58.2
LPA	N/A	N/A	65.3	78.7	58.5
LogicalFactChecker	BooksCorpus + Wikipedia Text	all	71.7	85.4	65.1
GNN-TabFact	BooksCorpus + Wikipedia Text	subset	72.2	86.4	65.4
TAPAS-Row-Col	BooksCorpus + Wikipedia Text	all	60.5	63.8	57.9
TAPAS-Row-Col	BooksCorpus + Wikipedia Text	subset	68.3	79.5	62.9
TAPAS-Row-Col	Wikipedia Tables	all	73.4*	86.6	67.0*
TAPAS-Row-Col	Wikipedia Tables	subset	73.9*	87.7*	67.2*
TAPAS-Row-Col-Rank	Wikipedia Tables	all	74.5*	87.0	68.5*
TAPAS-Row-Col-Rank	Wikipedia Tables	subset	74.8*	88.1*	68.5*
TAPAS-Row-Col-Rank	Wikipedia Tables + SQA	all	**76.0***	**89.0***	**69.8** *
TAPAS-Row-Col-Rank	Wikipedia Tables + SQA	subset	74.6*	88.9*	67.7*

Table 2: Percent accuracy of different models for table-based fact verification on the TabFact test set. "Subset" means only use the table columns with entities linked to the statements during training, while "all" means use all the columns. Models with accuracies significantly surpassing the previous state of the art, GNN-TabFact, are highlighted with * ($p < 0.05$, paired-permutation test). **Bold** indicates the best results in a column.

increasing the accuracy from 58.2% to 62.9%. It reveals that TAPAS-Row-Col's structural encoding is more effective for verifying statements involving complicated aggregation over multiple table rows.

Does filtering the table columns help? Table-BERT works on the subset of the table columns relevant to the statements, since the a table serialized in "natural" language might be too long for sequence models. Chen et al. (2019) filter tables to the columns containing entities linked to the statement. We run TAPAS-Row-Col model on both full table and subset of the columns. Table 2 shows that, when TAPAS-Row-Col is fine-tuned from the original BERT model, shrinking the table only to the related columns significantly improves its accuracy from 60.5% to 68.3%. However, when we fine tune TAPAS-Row-Col model pre-trained on the Wikipedia tables, the difference is not that significant. It shows that filtering the table columns mainly helps when applying language models pre-trained on unstructured data to structured data.

What is the effect of pre-training? We fine-tuned TAPAS-Row-Col on two different types of pre-trained model: the original BERT model pre-trained on unstructured text and the TAPAS model pre-trained on Wikipedia Tables together with sentences from the corresponding text paragraph by Herzig et al. (2020). Both of the pre-trained models are trained with the masked language modeling task. As we can see, pre-training on tables significantly improves performance of TAPAS-Row-Col from 68.3% to 73.9%. Fine-tuning TAPAS-Row-Col-Rank on the model pre-trained on the question answering task, where numerical reasoning skills are required to generate an answer for a given question, further improves accuracy from 74.5% to 76%. However, if we only fine-tune on the subset of table columns matching the statement, TAPAS-Row-Col-Rank does not benefit from the pre-trained question answering model. We conjecture that it is because the original model is pre-trained on full tables.

Does adding numerical comparisons help? Both GNN-TabFact and TAPAS-Row-Col-Rank model numerical comparison relations among cells in the same column. GNN-TabFact learns a representation for each table cell by iteratively passing their numerical relations such as greater or less in a graph neural network. TAPAS-Row-Col-Rank model adds the rank and inverse rank index for cells in the same column as special position embedding for each token in the table cells. GNN-TabFact significantly outperforms Table-BERT by adding a numerically aware graph neural network on top of it. It also outperforms the LogicalFactChecker model, which first derives a program from the table and the statement and then learns the representation for the programs via a program-guided neural module network. Also, TAPAS-Row-Col-Rank model significantly outperforms TAPAS-Row-Col by introducing rank position embedding for each numerical column. We also find that TAPAS-Row-Col-Rank outperforms GNN-TabFact without complicated graph inference. It is worth noting that even TAPAS-Row-Col, which does not utilize any numerical information, outperforms GNN-TabFact, the previous state of the art on TabFact. This again demonstrates the advantage of directly encoding ta-

Model	Pre-train	Columns	Test			
			Superlative	Comparative	Sum	Count
Table-BERT	BooksCorpus + Wikipedia Text	subset	59.2	56.3	60.9	54.9
GNN-TabFact	BooksCorpus + Wikipedia Text	subset	63.7	62.1	61.2	64.6
TAPAS-Row-Col	BooksCorpus + Wikipedia Text	all	58.7	61.2	57.8	59.8
TAPAS-Row-Col	Wikipedia Tables	all	66.3	63.4	65.6	65.3
TAPAS-Row-Col-Rank	Wikipedia Tables	all	72.0	64.7	63.3	67.2
TAPAS-Row-Col-Rank	Wikipedia Tables + SQA	all	**74.6**	**66.8**	**66.3**	**70.6**

Table 3: Results on test statements with superlative, comparative, sum, and count operations.

ble structure in TAPAS and the value of pre-training on structured data.

4.3 Further Experiments and Discussion

To analyze model performance, we perform further experiments on different training and test sets.

How does numerical information help? To examine how numerical comparison information helps to improve performance, we use heuristic rules to find the test statements involving *comparing* table records, *summing* table columns, and *counting* table records. We first find all words ending in *-est* in complex statements and remove those that do not belong to superlative words. Then we extract all test samples from the complex set containing these words or the word *most* as the **superlative test set**. We construct a **comparative test set** in a similar way by finding words ending in *-er*. Non-comparative words are removed and the words *more* and *less* are added to the set. We constrain the comparative samples to have the word *than* together with one of the comparative words. The **sum test set** is constructed by finding all the samples containing *total* or *sum*, the **count** set is constructed by finding all the samples containing *all*, *every*, *none*, *only* and *each*. The final superlative and comparative test sets contain 1,701 and 1,366 instances, respectively. The sum and counting sets contain 344 and 1,710, respectively. Table 3 shows the results of different models on these four sets. All results are reported for encoding full tables for variants of TAPAS-Row-Col models. As we can see, by modeling numerical comparison information among table cells in the same column, GNN-TabFact model significantly outperforms Table-BERT model on the superlative, comparative and count sets. Fine tuning TAPAS-Row-Col model pre-trained on the Wikipedia tables dataset improves performance on all the sets compared to fine tuning it from pre-trained BERT model. By adding ranking information, TAPAS-Row-Col-Rank significantly improves over TAPAS-Row-Col model on the superlative set. The increase of performance on other sets are not significant. Fine-tuning TAPAS-Row-Col-Rank on the model pre-trained for question answering further improves performance on all sets. It is also worth noting that TAPAS-Row-Col, with pre-training on Wikipedia tables, even outperforms GNN-TabFact model without utilizing any numerical comparison information. It also reveals that adding ranking information model benefits superlative statements most, and classifying comparative statements is more difficult than superlative statements.

Training Set	All	Simple	Complex
Complex	70.2	76.4	**67.2**
Simple	68.4	**88.2**	58.9
Mixed	**73.0**	86.1	66.6

Table 4: Results of fine-tuning Tapas-Row-Col-Rank model on statements with different complexity.

How does training set complexity affect model performance? We sampled three subsets from the original training set: one only contains simple statements (as defined by TabFact), one only contains complex statements, and one mixed set with equal size of simple and complex statements. All three sets have the same size: 41,366 training samples and 12,793 validation samples. We fine-tune the TAPAS-Row-Col-Rank model pre-trained on the question-answering task on three different sets and present the results in Table 4. The model trained on the mixed data achieves the highest overall accuracy on the test set. A significant drop of performance could be observed when the model is trained and test on statements with mismatched complexity.

How well do models trained on TabFact generalize to other data? Since statements in TabFact were generated by annotators specifically for fact verification, we explored generalization performance with a synthetic table-based fact verification dataset generated from the ToTTo dataset, a

Train	TabFact			Synthetic		
	All	Simple	Complex	All	Postive	Negative
TabFact	76.0	89.0	69.8	79.2	86.9	71.6
Synthetic	63.3	76.4	57.0	87.0	92.1	81.8

Table 5: TAPAS-Row-Col-Rank model training and testing on TabFact and ToTTo-derived synthetic data.

	Original Entailed Statement	Synthetic Refuted Statements
1	Baron Fabian Von Fersen (1626 – 1677) was a Swedish field marshal.	Baron Fabian Von Fersen (626 – 1677) was a Swedish field marshal.
2	As a sophomore, Peters averaged 16.8 points and 6.7 rebounds per game.	As a sophomore, Peters averaged 16.8 points and 66.7 rebounds per game.
3	Ralph Herseth (1909 – 1969) was the 21st governor of South Dakota from January 6, 1959 to January 3, 1961.	Ralph Herseth (1909 – 1969) was the 11st governor of South Dakota from January 6, 1959 to January 3, 1961.

Table 6: Examples of synthetic refuted statements that the Tapas-Row-Col-Rank model trained on TabFact failed to recognize. The errors are in red and the corresponding correct numbers are underlined.

table-to-text generation dataset proposed by Parikh et al. (2020). ToTTo was constructed by asking annotators to modifying the original sentence in Wikipedia that referenced a table. Annotators deleted irrelevant parts of the sentence and replaced pronominal references with a named entity from the table or the contextual metadata. Annotators were also asked to highlight the table cells that support the sentence. In all, 128,461 tables were annotated, with one sentence per table.

From ToTTo, using negative sampling, we derive a synthetic dataset for fact verification to compare with the TabFact data. We first removed all the samples in Totto data that also appears in TabFact data. Since the cells related to the summarizing sentence are highlighted in ToTTo, we extract all the sentences containing entities that exactly match the highlighted table cells and treat them as as facts. Then for each fact statement, we randomly choose an entity in it and replace it with a randomly chosen cell from the same column to generate a false statement. To ensure the false statement is different from the fact statement, we only choose cells whose values differ from the original. We end up with 75,292 training, 8,366 validation, and 5,242 test samples. We fine-tune the TAPAS-Row-Col-Rank model on this synthetic dataset. The results on this synthetic dataset and TabFact are listed in Table 5. Training on synthetic data achieves 76.4% accuracy on the simple TabFact statements, while training on TabFact achieves 79.2% accuracy on synthetic data. This confirms that the model trained on TabFact model is able to recognize cell copying errors. Both models are better at classifying positive statements than negative statements, which is more obvious for the model trained on TabFact.

We also generated another synthetic dataset with uniformly distributed character-editing errors on digits to see whether models trained on TabFact could capture this kind of error. We first extract all the test sentences containing numbers that exactly matches the highlighted table cells. Then we randomly choose to insert, delete or substitute one random digit in the first position (to make the task easier) of a randomly chosen matching number in the sentence (to ensure there is clue in the table). A total of 1,126 synthetic refuted statements are generated for testing. We check whether the TAPAS-Row-Col-Rank model trained on TabFact can recognize this type of error. Only 43.0% of these synthetic statements are classified as refuted by the model, which is much lower than the 71.6% accuracy on the refuted statements generated above by cell copying. It reveals that models trained on TabFact data are less sensitive to purely numerical errors. Table 6 shows examples of synthetic false statements the model trained on TabFact failed to recognize.

5 Conclusions

We adapted the Table Parsing (TAPAS) model, with efficient encoding of table content, structure, and numerical comparison information for the task of table-based fact verification. We compared two variant models: TAPAS-Row-Col, which models table contents and structure, and TAPAS-Row-Col-Rank, which adds numerical comparison information. Experiments showed that both TAPAS-Row-Col and TAPAS-Row-Col-Rank outperform the state-of-the-art numerically-aware graph neural network model by pre-training on tabular data. We also examined how ranking information helps improve TAPAS's performance on superlative, comparative, sum, and count statements. Models trained on different datasets were compared to study how question complexity affects model performance. We also constructed two synthetic datasets to examine the generalization of these models. We find models trained on TabFact perform well on errors arising from simple cell replacement but not on digit-editing errors. In future, we aim to extend TAPAS to explicitly model more numerical operations for fact verification and to correct false statements automatically given tabular evidence.

Acknowledgements

Rui Dong was supported by a U.S. Department of Energy Award (DE-SC0019958). We thank Brian Simmons for helpful discussions and the anonymous reviewers for their suggestions.

References

Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. 2018. Table-to-text: Describing table region with natural language. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020. Logical natural language generation from open-domain tables. *arXiv preprint arXiv:2004.10404*.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2019. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*.

Li Deng, Shuo Zhang, and Krisztian Balog. 2019. Table2vec: Neural word and entity embeddings for table population and retrieval. *arXiv preprint arXiv:1906.00041*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Julian Eisenschlos, Syrine Krichene, and Thomas Mueller. 2020. Understanding tables with intermediate pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 281–296.

Majid Ghasemi-Gol and Pedro Szekely. 2018. Tabvec: Table vectors for classification of web tables. *arXiv preprint arXiv:1802.06290*.

Till Haug, Octavian-Eugen Ganea, and Paulina Grnarova. 2018. Neural multi-step reasoning for question answering on semi-structured tables. In *European conference on information retrieval*, pages 611–617. Springer.

Verena G Herbert, Andreas Frings, Herwig Rehatschek, Gisbert Richard, and Andreas Leithner. 2015. Wikipedia–challenges and new horizons in enhancing medical education. *BMC medical education*, 15(1):32.

Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Seattle, Washington, United States. To appear.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831.

Mayank Jobanputra. 2019. Unsupervised question answering for fact-checking. *arXiv preprint arXiv:1910.07154*.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6859–6866.

Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. 2017. Understanding the semantic structures of tables with a hybrid deep neural network architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6908–6915.

Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2018. Order-planning neural text generation from structured data. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. 2020. Learn to combine linguistic and symbolic information for table-based fact verification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5335–5346.

Amir Soleimani, Christof Monz, and Marcel Worring. 2019. Bert for evidence retrieval and claim verification. *arXiv preprint arXiv:1910.02655*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Mohamed Trabelsi, Brian D Davison, and Jeff Heflin. 2019. Improved table retrieval using multiple context embeddings for attributes. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1238–1244. IEEE.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*.

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7810–7825.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1624–1629.

Wanjun Zhong, Duyu Tang, Zhangyin Feng, Nan Duan, Ming Zhou, Ming Gong, Linjun Shou, Daxin Jiang, Jiahai Wang, and Jian Yin. 2020. LogicalFactChecker: Leveraging logical operations for fact checking with graph module network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6053–6065, Online. Association for Computational Linguistics.

A Study of Automatic Metrics for the Evaluation of Natural Language Explanations

Miruna-Adriana Clinciu
Edinburgh Centre for Robotics
Heriot-Watt University
University of Edinburgh
mc191@hw.ac.uk

Arash Eshghi
Heriot-Watt University
Edinburgh, United Kingdom
a.eshghi@hw.ac.uk

Helen Hastie
Heriot-Watt University
Edinburgh, United Kingdom
h.hastie@hw.ac.uk

Abstract

As transparency becomes key for robotics and AI, it will be necessary to evaluate the methods through which transparency is provided, including automatically generated natural language (NL) explanations. Here, we explore parallels between the generation of such explanations and the much-studied field of evaluation of Natural Language Generation (NLG). Specifically, we investigate which of the NLG evaluation measures map well to explanations. We present the ExBAN corpus: a crowd-sourced corpus of NL explanations for Bayesian Networks. We run correlations comparing human subjective ratings with NLG automatic measures. We find that embedding-based automatic NLG evaluation methods, such as BERTScore and BLEURT, have a higher correlation with human ratings, compared to word-overlap metrics, such as BLEU and ROUGE. This work has implications for Explainable AI and transparent robotic and autonomous systems.

1 Introduction

The machine learning models and algorithms underlying today's AI and robotic systems are increasingly complex with their internal operations and decision-making processes ever more opaque. This opacity is not just an issue for the end-user, but also the creators and analysts of these systems. As we move towards building safer and more ethical systems, this lack of transparency needs to be addressed. One key trait of a transparent system is its ability to be able to *explain* its deductions and articulate the reasons for its actions in Natural Language (NL). As the area of Explainable AI (XAI) grows and is mandated (cf. the EU General Data Protection Regulation's "right to explanation" (Commission, 2018) and standardisation (cf. IEEE forthcoming standard on Transparency (P7001)), it has become ever more important to be able to evaluate the quality of the NL explanations themselves, as well as the AI algorithms they explain. Furthermore, the importance of evaluating explanations has been emphasised by researchers within the social cognitive sciences (Leake, 2014; Zemla et al., 2017; Doshi-Velez and Kim, 2017). To date, explanations have mostly been evaluated by collecting human judgements, which is both time-consuming and costly. Here, we view generating explanations as a special case of Natural Language Generation (NLG), and so we explore mapping existing automatic evaluation methods for NLG onto explanations. We study whether general, domain-independent evaluation metrics within NLG are sensitive enough to capture the peculiarities inherent in NL explanations (Kumar and Talukdar, 2020), such as causality; or whether NL explanations constitute a sui-generis category, thus requiring their own automatic evaluation methods and criteria.

In this paper, we firstly present the ExBAN dataset: a corpus of NL explanations generated by crowd-sourced participants presented with the task of explaining simple Bayesian Network (BN) graphical representations. These explanations were subsequently rated for *Clarity* and *Informativeness*, two subjective ratings previously used for NLG evaluations (Gatt and Krahmer, 2018; Howcroft et al., 2020). The motivation behind using BN is that they are reasonably easy to interpret, are frequently used for the detection of anomalies in the data (Tashman et al., 2020; Saqaeeyan et al., 2020; Metelli and Heard, 2019; Mascaro et al., 2014), and have been used to approximate deep learning methods (Riquelme et al., 2018; Gal and Ghahramani, 2016), which we could, in turn, explain in Natural Language.

Secondly, we explore a wide range of automatic measures commonly used for evaluating NLG to

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2376–2387
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

understand if they capture the human-assessed quality of the corpus explanations. We then go on to discuss their strengths and weaknesses through quantitative and qualitative analysis.

Our contributions are thus as follows: (1) a new corpus of natural language explanations generated by humans, who are asked to interpret Bayesian Network graphical representations, accompanied by subjective quality ratings of these explanations. This corpus can be used in various application areas including Explainable AI, general Artificial Intelligence, linguistics and NLP; (2) a study of methods for evaluating explanations through automatic measures that reflect human judgements; and (3) qualitative discussion into these metrics' sensitivity by examining specific explanations varying on the Informativeness/Clarity scales.

2 Related Work

Explanations are a core component of human interaction (Scalise et al., 2017; Krening et al., 2017; Madumal et al., 2019). In the context of Machine Learning (ML), explanations should articulate the decision-making process of an ML model explicitly, in a language familiar to people as communicators (De Graaf and Malle, 2017; Miller, 2018). According to Plumb et al. (2018), three of the most common types of explanation are: (1) *global explanations*, which describe the overall behaviour of the entire model (Arya et al., 2019); (2) *local explanations*, commonly taking the form of counterfactuals (Sokol and Flach, 2019) that describe why particular events happened (known also as "everyday explanations"); and (3) *example-based explanations* that present examples from the training set to explain algorithmic behaviour (Cai et al., 2019).

Recently, various explanation systems provide different types of explanations for AI systems: the LIME method visually explains how sampling and local model training works by using local interpretable model-agnostic explanations (Ribeiro et al., 2016); MAPLE can provide feedback for all three types of explanations: example-based, local and global explanations (Plumb et al., 2018); CLEAR explains a single prediction by using local explanations that include statements of key counterfactual cases (White and d'Avila Garcez, 2019). Whilst these techniques and tools gain some ground in explaining deep machine learning, the explanations they provide are not necessarily aimed at the (non-expert) end-user and so are not always intuitive.

NLG has traditionally been broken down into "what" to say (content selection) and "how" to say it (surface realisation) and can draw parallels with Natural Language explanations. In particular, it is important to gauge how much content or how many reasons to present to the user, to inform them fully without overloading them. For example, prior work has shown that people prefer shorter explanations that offer only sufficient detail to be considered useful (Harbers et al., 2009; Yuan et al., 2011).

According to Miller et al. (2017), how explainers generate and select explanations depends on so-called pragmatic influences of causes, and they found that people seem to prefer simpler and more general explanations. Similarly, Lombrozo (2007) notes that simplicity and generality might be the key to evaluating explanations. This was partly the case described in (Chiyah Garcia et al., 2018), but here the users were experts and preferred to be given all possible reasons but as precise and brief as possible. It is clear from these prior works that explanations have to be evaluated in the context of the scenario, prior knowledge and preferences of the explainee, and the intent and goals of the explainer. These could be, for example, establishing trust (Miller et al., 2017), agreement, satisfaction, or acceptance of the explanation and the system (Gregor and Benbasat, 1999).

Somewhat analogous to auto-generated explanations are the fields of summarisation of text (Tourigny and Capus, 1998; Deutch et al., 2016) and Question-Answering (Dali et al., 2009; Xu et al., 2017; Lamm et al., 2020). This is because they provide users (expert and lay users) with various forms of summaries (visual or textual) and answers containing explanations to enable them to have a better understanding of content.

Summarisation methods and sentence compression techniques can help to build comprehensive explanations (Winatmoko and Khodra, 2013). With regards to evaluating these summarisation methods, Xu et al. (2020) proposed an evaluation metric that weighted the facts present in the source document according to the facts selected by a human-written (natural language) summary, by using contextual embeddings. This evaluation of text accuracy is indeed related to explanations because any explanation must contain enough statements to support decision-making and understanding. These statements should be accurate and true.

The growing interest in the AI community to investigate the potential of NL explanations for bridging the gap between AI and HCI has resulted in an increasing number of NL explanations datasets. The ELI5 dataset[1] (Fan et al., 2019) is composed of explanations represented as multi-sentence answers for diverse questions where users are encouraged to provide answers, which are comprehensible for a five-year-old. WorldTree V2[2] (Jansen et al., 2019) is a corpus of Science-Domain that contains explanation graphs for elementary science questions, where explanations represent interconnected sets of facts. CoS-E[3] is a dataset of human explanations for commonsense reasoning in the form of natural language sequences and highlighted annotations (Rajani et al., 2019). Multimodal Explanations Datasets (VQA-X and ACT-X) contain textual and visual explanations from human annotators (Park et al., 2018). e-SNLI[4] is a corpus of explanations built on the question: "Why is a pair of sentences in a relation of entailment, neutrality, or contradiction?" (Camburu et al., 2018). Finally, the SNLI corpus[5] is a large annotated corpus for learning natural language inference (Bowman et al., 2015), considered one of the first corpora of NL explanations.

In this paper, we present a new corpus for NL explanations. The ExBAN corpus presented here provides a valuable addition to this set of corpora as it is aimed at explaining structured graphical models (in particular Bayesian Networks), that are closely linked to ML methods.

3 ExBAN Corpus

The ExBAN Corpus (**Ex**planations for **BA**yesian **N**etworks)[6] consists of NL Explanations collected in a two step process: (1) NL explanations were produced by human subjects; (2) in a separate study, these explanations were evaluated in terms of Informativeness and Clarity.

For Step 1, each subject was shown graphical representations of three Bayesian Networks (BN), in random order. They were then asked to produce text to describe how they interpreted the BN. The three BN used in the data collection are presented in Figure 1and represent well-known BN examples, extracted from Russell (2019). For Step 2 in a separate experiment, approximately 80 of these generated explanations were presented to a different set of subjects in random order, along with a scenario description and the graphical model image. Subjects were asked to rate them in terms of Informativeness and Clarity. The worded scenario descriptions were not given to subjects in the first stage, so as not to prime them when generating explanations.

3.1 Step 1: Natural Language Explanations Corpus

Survey Instrument. A pilot was performed to test options and ensure the completion time, leading to the final survey instrument. The survey was divided into five sections: 1) consent form; 2) closed-ended questions related to English proficiency, computing and AI experience: "How much computing experience do you have?", "What is your English Proficiency Level?", "How much experience do you have in the field of Artificial Intelligence?"; 3) attention-check question, where participants received an image of a graphical model, and they had to select the correct answer(s) for the given image; and 4) respondents were asked to explain the three graphical models, in their own words. All respondents received the graphical model survey questions in randomised order. The appropriate ethical procedures were followed in accordance with ethical standards, and ethical approval was obtained.

Participants. 85 participants were recruited via social media. English proficiency level, computing experience and AI experience were rated on a numerical scale, from 1 to 7 (1 = beginner, 7 = advanced). The majority of participants ($n = 83$) rated their level of English proficiency with values higher than 5, with over half of the participants rating their level as 7. Just 12% ($n =$

---

[1] https://facebookresearch.github.io/ELI5/

[2] http://www.cognitiveai.org/explanationbank

[3] https://github.com/salesforce/cos-e

[4] https://github.com/OanaMariaCamburu/e-SNLI

[5] https://nlp.stanford.edu/projects/snli/

[6] The data is openly released at https://github.com/MirunaClinciu/ExBAN

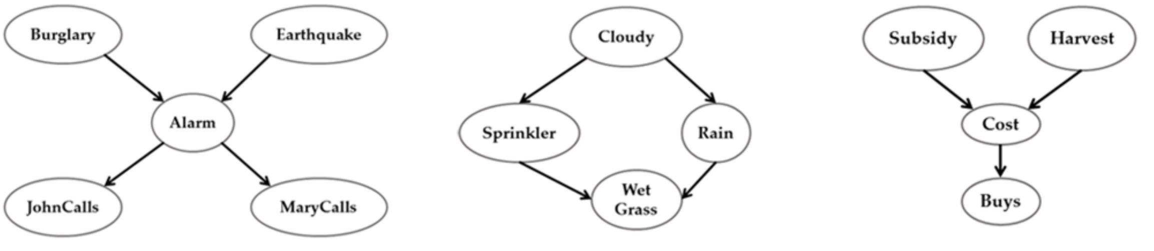


Diagram 1

Ref: "In the event of either burglary or earthquake the alarm will call John or Mary."

Diagram 2

Ref: "If it gets cloudy, it can rain or the sprinkler may get activated. Whenever it rains or the sprinkler gets activated, the grass gets wet."

Diagram 3

Ref: "The subsidy and the harvest influence the cost of production. The final price influences the amount of bought products."

Figure 1. Annotated diagrams with assigned explanation references, where **Diagram 1** represents a typical Bayesian Network, **Diagram 2** represents a multiply-connected network and **Diagram 3** represents a simple network with both discrete variables (Subsidy and Buys) and continuous variables (Harvest and Cost). Beneath each diagram, the gold standard references are provided.

10) participants rated their computing experience scores with a value lower than 5 and 82% ($n = 70$) of participants had a high level of computing experience. Subjects had mixed experience with AI with over half (54%) having some experience ($n = 46$), but 46% of them had limited AI experience ($n = 39$).

Collected NL explanations. Quality control of the collected data included a cleaning step where participants' responses were hand-checked and removed if the participants did not attempt to complete the tasks. Explanations that contained misspellings and missing punctuation were corrected manually (both the raw data and cleaned data are available). The number of explanations for each diagram, after the data cleaning step are as follows: Diagram 1: 84 explanations, 1788 words; Diagram 2: 83 explanations, 1987 words; and Diagram 3: 83 explanations, 1400 words.

3.2 Step 2: Human Evaluation for Quality

Survey Instrument. To investigate the quality of the explanations collected in Step 1, we performed a human evaluation of the generated explanations. A pilot survey was performed to test and refine options, where respondents ($n = 45$) were recruited from Amazon Mechanical Turk and were compensated monetarily.

Each participant was given three tasks, each corresponding to the BN presented in Figure 1 with the order randomised. Along with the BN image, a simple description story was provided in order to give the subject a better understanding of the context as well as instructions on how to approach these tasks. Here, we give the scenario for Diagram 1 to illustrate this: "John and Mary bought their dream home. To keep their home safe, they installed a Burglary/Earthquake Alarm. Also, they received an instruction manual where they found the following diagram: They are not sure if they correctly understood the diagram. On the following pages are some worded explanations. We need your help to evaluate them!"

For every BN image, the participants were asked to evaluate 5 explanations in terms of: **Informativeness** (Q: "How relevant the information of an explanation is"; Likert scale, where 1 = Not Informative and 7 = Very Informative); and **Clarity** (Q: "How clear the meaning of an explanation is"; Likert scale, where 1 = Unclear and 7 = Very Clear).

Participants. The final data collection survey was advertised on social media as "a 10-minute survey, where participants were asked to provide feedback about how understandable the explanations of the three graphs are". Demographic information was collected (age range and gender). A total of 96 participants answered the survey. As screening criteria, participants had to complete all survey questions. Post validation, we had a sample of 56 participants consisting of 42 male participants (75%), 11 female participants (19.6%) and 2 non-binary gender participants (3.6%). Gender imbalance might be due to "differences in female and male values operating in an online environment" (Smith, 2008). Half of the participants ($n = 28$) are aged between 23-29 years old, followed by

30% of participants aged between 18-22 ($n = 17$), 20% aged 40–49 ($n = 11$), 18% aged 30–39 ($n = 10$). Previous studies have identified a high degree of inconsistency in human judgements of natural language (Novikova et al., 2018; Dethlefs et al., 2014); each participant can have a different perception of the interpretation of these metrics, even if a definition of these metrics is provided. Indeed, we found that in our data, explanation ratings can vary significantly, with an explanation rated highly by one person for Clarity, but viewed as very unclear by another annotator. This was the case for both Clarity and Informativeness.

We aim to create a reliable database of varying quality of NL explanations, i.e. where the quality of explanations is generally uncontested by the majority. Therefore, subjective ratings were post-processed. For each explanation, we collected a minimum of 3 judgments. Explanations received ratings from 1 to 7; we classified bad explanations as those with low ratings (ratings $<$5) and good explanations, as those with higher ratings (ratings $\geq$5). For any one explanation, if the difference between the number of good and bad ratings is $\leq$1, then that explanation is considered hard to judge and difficult to reach a consensus on and thus removed. After this pre-processing step, the corpus contained ratings for 54 explanations for Diagram 1, 34 explanations for Diagram 2, and 54 explanations for Diagram 3.

To verify the agreement between different raters, we used Krippendorff's Alpha, a measure of inter-rater reliability (Krippendorff, 1980). We computed Krippendorff's Alpha coefficient using the Python package `krippendorff` (version 0.3.2). After the pre-processing step, the agreement between subjects increased, see Table 1 for the post-processing Alpha values for each of the Bayes Nets. Alpha values between .21 to .40 indicate fair agreement and values between .41 to .60 indicate moderate agreement (Hallgren, 2012). Here, we can see that explanations for Diagram 2 were particularly contentious, but overall the numbers reflect fair to moderate agreement.

	Diagram 1	Diagram 2	Diagram 3	All Diagrams
Inform.	0.514	0.202	0.420	0.377
Clarity	0.440	0.182	0.361	0.319

Table 1. Inter-annotator agreement measured by Krippendorff's Alpha

4 NLG Evaluation Metrics

Here, we describe the reasoning behind our choice of subjective measures that attempt to capture both the content and its correctness (Informativeness) and quality of expression (Clarity). We also describe objective measures commonly used for automatic evaluation of NLG, and which we will extract from the ExBAN corpus.

4.1 Subjective NLG Evaluation Metrics

Human evaluation is considered a primary evaluation criterion for NLG systems (Gatt and Krahmer, 2018; Mellish and Dale, 1998; Gkatzia and Mahamood, 2015; Hastie and Belz, 2014). Through Explainable AI, we want to achieve Clarity and understanding in communicating the process of AI systems. Therefore, explanations should be clear and easily understood by users. Traditional human evaluation metrics are clearly needed for increasing transparency, avoiding confusion and misunderstanding.

Informativeness. As defined in the field of NLG, Informativeness targets relevance or correctness of an NLG output relative to an input (Dušek et al., 2020). According to the literature, Informativeness can provide "timely, relevant and useful information" (Novikova et al., 2018) and "make information immediately accessible" (Maxwell et al., 2017). Sometimes, Informativeness is linked with accuracy, or adequacy (Novikova et al., 2018). As mentioned previously, explanations contain statements with some prior knowledge that must be accurate and true (Goodrich et al., 2019; Xu et al., 2020).

Clarity. An explanation should be clear to achieve effective communication. In the NLG field, Clarity implies that text is easily understood (Belz and Kow, 2009; van der Lee et al., 2017) and that the reader is familiar with basic information introduced in the text (Lampouras and Androutsopoulos, 2013). In addition, Clarity can also help expose the truthfulness and correctness of textual data (Mahapatra et al., 2016).

4.2 Automatic Evaluation Metrics

This section describes a number of automatic metrics commonly used in NLG evaluation and selected for this study. These fall into two categories: 1) word-overlap metrics, e.g. BLEU, METEOR and ROUGE (Novikova et al., 2017);

and 2) embedding-based metrics, e.g. BERTScore and BLEURT (Sellam et al., 2020). Each of these metrics is compared to one or more "Gold Standard" text as inspired by the Machine Translation community and adopted for evaluating document summarisation and NLG (Belz and Reiter, 2006). The gold standard is normally a piece of natural language text, annotated by humans as correct, i.e. a solution for a given task. Automatic evaluation is based on this gold standard, by verifying potential similarity (Kovář et al., 2016). However, the selection of gold standards involves subjectivity and specificity (Kovář et al., 2016), and this is part of the reason that automatic metrics have received some criticism (Hardcastle and Scott, 2008).

BLEU (B) (Papineni et al., 2001) is widely used in the field of NLG and compares n-grams of a candidate text (e.g. that generated by an algorithm) with the n-grams of a reference text. The number of matches defines the goodness of the candidate text. **SacreBLEU** (SB) (Post, 2018) is a new version of BLEU that calculates scores on the detokenized text. **METEOR** (M) was created to try to address BLEU's weaknesses (Lavie and Agarwal, 2007). METEOR evaluates text by computing a score based on explicit word-to-word matches between a candidate and a reference. When using multiple references, the candidate text is scored against each reference, and the best score is reported. **ROUGE** (R) (Lin, 1971) evaluates n-gram overlap of the generated text (candidate) with a reference. **ROUGE-L** (RL) (Longest Common Subsequence) computes the longest common subsequence (LCS) between a pair of sentences.

BERTScore (BS) (Zhang et al., 2020) is a token-level matching metric with pre-trained contextual embeddings using BERT (Devlin et al., 2019) that matches words in candidate and reference sentences using cosine similarity. **BLEURT** (BRT) (Sellam et al., 2020) is a text generation metric also based on BERT, pre-trained on synthetic data; it uses random perturbations of Wikipedia sentences augmented with a diverse set of lexical and semantic-level supervision signals. BLEURT uses a collection of metrics and models from prior work, including BLEU and ROUGE. Evaluation based on the meanings of words using embeddings (BERTScore, BLEURT) might capture some relevant features of explanations, as word representations are dynamically informed by the words around them (McCormick and Ryan, 2019)).

5 Correlation Study of Automatic Metrics

As noted in the introduction, it remains an open question as to what degree the automatic metrics for NLG reviewed above can capture the quality of NL explanations (Clinciu and Hastie, 2019). Thus, we ran a correlation analysis to investigate the degree to which each of the automatic metrics correlates with human judgements using the ExBAN corpus, and which aspects of human evaluation (Clarity/Informativeness), such automatic measures can capture. With regards to the choice of gold standard text, we picked explanations that received the maximum score in the human evaluation, in both Clarity and Informativeness. Gold standard explanations of each diagram are presented in Figure 1.

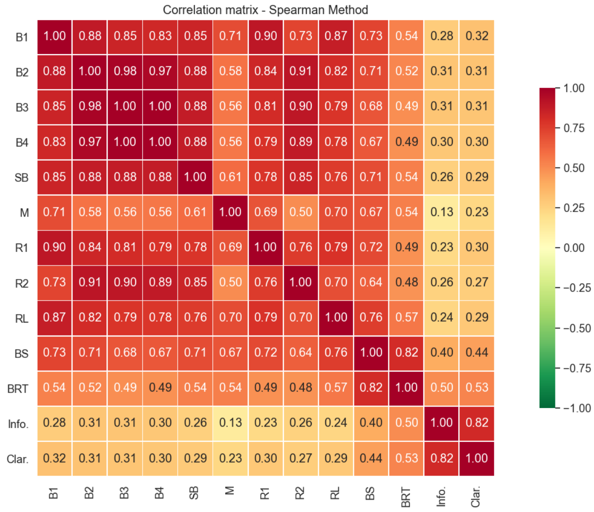
5.1 Results

The correlations between automatic metrics and human ratings were computed using the Spearman correlation coefficient. For each explanation, we calculated the median of all the ratings given (median was calculated because the data is ordinal, non-parametric rating data, as is also reported in Braun et al. (2018); Novikova et al. (2017)). These medians were then correlated with the automatic metric scores in Tables 2 and 3 and Figure 2. A summary of the results of the correlation analysis include the following:

1. Word-overlap metrics such as BLEU (n = 1,2,3,4), METEOR and ROUGE (n = 1,2) presented low correlation with human ratings.
2. BERTScore and BLEURT outperformed other metrics and produced higher correlation with human ratings than other metrics on all diagrams. BERTScore values range between [0.23, 0.43] and for BLEURT values range between [0.26, 0.53].
3. Human ratings for Informativeness and Clarity are highly correlated with each other, as observed in Figure 2 ($r = 0.82$).

5.2 Discussion

BLEU-based metrics can be easily and quickly computed; however, they do not correlate as well with human ratings as other methods presented here. This might be due to certain limitations, such as the fact that they rely on word overlap



Correlation matrix - Spearman Method

	B1	B2	B3	B4	SB	M	R1	R2	RL	BS	BRT	Info.	Clar.
B1	1.00	0.88	0.85	0.83	0.85	0.71	0.90	0.73	0.87	0.73	0.54	0.28	0.32
B2	0.88	1.00	0.98	0.97	0.88	0.58	0.84	0.91	0.82	0.71	0.52	0.31	0.31
B3	0.85	0.98	1.00	1.00	0.88	0.56	0.81	0.90	0.79	0.68	0.49	0.31	0.31
B4	0.83	0.97	1.00	1.00	0.88	0.56	0.79	0.89	0.78	0.67	0.49	0.30	0.30
SB	0.85	0.88	0.88	0.88	1.00	0.61	0.78	0.85	0.76	0.71	0.54	0.26	0.29
M	0.71	0.58	0.56	0.56	0.61	1.00	0.69	0.50	0.70	0.67	0.54	0.13	0.23
R1	0.90	0.84	0.81	0.79	0.78	0.69	1.00	0.76	0.79	0.72	0.49	0.23	0.30
R2	0.73	0.91	0.90	0.89	0.85	0.50	0.76	1.00	0.70	0.64	0.48	0.26	0.27
RL	0.87	0.82	0.79	0.78	0.76	0.70	0.79	0.70	1.00	0.76	0.57	0.24	0.29
BS	0.73	0.71	0.68	0.67	0.71	0.67	0.72	0.64	0.76	1.00	0.82	0.40	0.44
BRT	0.54	0.52	0.49	0.49	0.54	0.54	0.49	0.48	0.57	0.82	1.00	0.50	0.53
Info.	0.28	0.31	0.31	0.30	0.26	0.13	0.23	0.26	0.24	0.40	0.50	1.00	0.82
Clar.	0.32	0.31	0.31	0.30	0.29	0.23	0.30	0.27	0.29	0.44	0.53	0.82	1.00

Figure 2. Heatmap of Spearman rank correlation between automatic evaluation metrics and human evaluation metrics (Informativeness and Clarity)

Metric	Diagram 1	Diagram 2	Diagram 3	All Diagrams
BLEU-1	0.27	0.25	0.41*	0.31*
BLEU-2	0.24	0.27	0.44*	0.33*
BLEU-3	0.15	0.23	0.39	0.26*
BLEU-4	0.02	0.21	0.13	0.13
SacreBleu	0.24	0.30	0.40*	0.30*
METEOR	0.11	-0.04	0.16	0.09
Rouge-1	0.27	0.24	0.41*	0.29*
Rouge-2	0.11	0.29	0.48*	0.29*
Rouge-L	0.29	0.28	0.34	0.29*
BERTScore	**0.37**	0.21	0.52*	0.37*
BLEURT	0.25	**0.38**	**0.58***	**0.39***

*S*ignificance of correlation: * denotes p-values < 0.05

Table 2. Highest absolute Spearman correlation between automatic evaluation metrics and human ratings for Informativeness, where the bold font represents the highest correlation coefficient obtained by an automatic evaluation metric

Metric	Diagram 1	Diagram 2	Diagram 3	All Diagrams
BLEU-1	0.25	0.09	0.34	0.24*
BLEU-2	0.24	0.15	0.41*	0.22
BLEU-3	0.01	0.10	0.31	0.14
BLEU-4	-0.01	0.09	0.18	0.10
SacreBleu	0.16	0.15	0.38	0.23
METEOR	0.17	0.13	0.30	0.21
Rouge-1	0.20	0.11	0.29	0.20
Rouge-2	0	**0.24**	0.46*	0.22
Rouge-L	0.21	0.09	0.33	0.21
BERTScore	**0.33**	0.23	0.43*	0.33*
BLEURT	0.26	0.22	**0.53***	**0.34***

*S*ignificance of correlation: * denotes p-values < 0.05

Table 3. Spearman correlation between automatic evaluation metrics and human ratings for Clarity, where the bold font represents the highest correlation coefficient obtained by an automatic evaluation metric

and are not invariant to paraphrases. Furthermore, they do not use recall, rather a Brevity Penalty, which penalizes generated text for being "too short" (Papineni et al., 2001). This way may not be appropriate for explanations, as good explanations may need to be lengthy by their very nature.

METEOR takes into consideration F1-measure by computing scores for unigram precision and recall. The fragmentation penalty is calculated using the total number of matched words (m, averaged over hypothesis and reference) and the number of chunks. In this way, it could identify synonyms, but perhaps not as well as the embedding-based metrics, as evidenced by the correlation figures in our results. With regards to ROUGE-based scores, due to the upper bound issues presented by Schluter (2017), it is impossible to obtain perfect ROUGE-n scores. Furthermore, ROUGE-L cannot differentiate if the reference and the candidate have the same longest common subsequence (LCS), but different word ordering. Again, word ordering may be important for the explanation in terms of explainee scaffolding (Palincsar, 1986).

It has been brought into question whether a single automatic measure is able to capture multiple aspects of subjective human evaluation (Belz et al., 2020). Thus, in order to understand to what degree the various metrics capture both Clarity and Informativeness, we investigated individual explanations and their ratings. Table 4 gives some extracts from the dataset along with the automatic metrics and the human evaluation scores of Informativeness and Clarity. Based on these human scores, the extracts are divided into: good explanations (high scores for both), bad explanations (low scores for both) and mixed explanations (mixed scores). We can see here that all metrics are reasonably good at capturing and evaluating the 'bad' explanations with low scores across the board. However, only the BLEURT metric is good at capturing both 'good and bad' explanation ratings, as observed in the difference in scores between these two categories. ROUGE-L and BERTScore do capture this difference in some cases, but they are not as consistent as BLEURT. The reason that BLEURT outperforms the other metrics may be because it uses a combination of word-overlap metrics as well as embeddings and thus may be capturing the best of these approaches.

Good Explanations	B1	B2	B3	B4	SB	M	R1	R2	RL	BS	BRT	Inf.	Clar.
(1) The alarm is triggered by a burglary or an earthquake.	0.19	0.12	0.00	0.00	0.05	0.23	0.25	0.09	0.12	0.51	0.52	7	7
(2) Cloudy weather may produce rain and activation of the sprinkler. Both rain and sprinkler activity makes the grass wet.	0.28	0.11	0.00	0.00	0.05	0.15	0.36	0.10	0.28	0.49	0.65	7	7
(3) Cost is dictated by the harvest (e.g. size) and available subsidies (e.g. government tax break/subsidy). Whether or not the product is bought depends on the cost.	0.18	0.09	0.00	0.00	0.02	0.07	0.25	0.09	0.12	0.20	0.51	7	7
Bad Explanations	B1	B2	B3	B4	SB	M	R1	R2	RL	BS	BRT	Inf.	Clar.
(4) Sensors = Alarm = prevention or ALERT.	0.06	0.00	0.00	0.00	0.01	0.04	0.00	0.00	0.04	0.00	0.00	1	1
(5) A diagram detailing a system whose goal is to make grass wet.	0.08	0.00	0.00	0.00	0.02	0.00	0.11	0.00	0.12	0.13	0.00	1.5	2
(6) The harvest and subsidy contribute to the cost, cost then buys??	0.28	0.00	0.00	0.00	0.03	0.15	0.36	0.00	0.24	0.20	0.30	2	1.5
Mixed Explanations	B1	B2	B3	B4	SB	M	R1	R2	RL	BS	BRT	Inf.	Clar.
(7) The grass is getting wet.	0.08	0.00	0.00	0.00	0.00	0.20	0.13	0.00	0.15	0.24	0.16	1.5	7
(8) Subsidy and harvest independently affect cost. Cost affects buys.	0.06	0.00	0.00	0.00	0.01	0.14	0.16	0.00	0.10	0.25	0.56	6	2.5
(9) Cloud cover influences whether it rains and when the sprinkler is activated. When either the sprinkler is turned on or when it rains, the grass gets wet.	0.48	0.33	0.21	0.14	0.22	0.24	0.50	0.24	0.38	0.49	0.65	7	3

Table 4. Examples of Good, Bad and Mixed Explanations according to human evaluation scores for Informativeness and Clarity (medians of all ratings for that explanation), presented with their automatic measures

Although Clarity and Informativeness highly correlate overall, there are occasions where explanations are rated by humans as higher on Clarity than Informativeness and visa-versa. However, there are rarely any cases where Clarity is high, and Informativeness is very low. Explanation 8 in Table 4 is the only example of this in our corpus. It is thus difficult to make any generalisations about this subset of the data. However, it does seem to be the case that BLEURT is more sensitive to Informativeness than Clarity (e.g. explanation 7 vs 8-9 in the table), but a larger study would be needed to show this empirically.

6 Conclusions and Future work

Human evaluation is an expensive and time-consuming process. On the other hand, automatic evaluation is a cheaper and more efficient method for evaluating NLG systems. However, finding accurate measures is challenging, particularly for explanations. We have discussed word embedding techniques (Mikolov et al., 2013; Kim, 2014; Reimers and Gurevych, 2020), which enable the use of pre-trained models and so reduces the need to collect large amounts of data in our domain of explanations, which is a challenging task. The embedding-based metrics mentioned here perform better than the word-overlap based ones. We speculate that this is in part due to the fact that the former capture semantics more effectively and are thus more invariant to paraphrases. These metrics have also been shown to be useful across multiple tasks (Sellam et al., 2020) but with some variation across datasets (Novikova et al., 2017). Therefore, future work would involve examining the effectiveness of automatic metrics across a wider variety of explanation tasks and datasets, as outlined in the Related Work section.

Embeddings are quite opaque in themselves. Whilst some attempts have been made to visualise them (Li et al., 2016), it remains that embedding-based metrics do not provide much insight into what makes a good/bad explanation. It would thus be necessary to look more deeply into the linguistic phenomena that may indicate the quality of explanations. In ExBAN, initial findings show that the number of nouns and coordinating conjunctions correlate with human judgements, however further in-depth analysis is needed. Additional metrics to add to the set explored here could include grammar-based metrics, such as readability and grammaticality, as in the study described in (Novikova et al., 2017).

Furthermore, an investigation is needed into the pragmatic and cognitive processes underlying explanations, such as argumentation, reasoning, causality, and common sense (Baaj et al., 2019). Investigating whether these can be captured automatically will be highly challenging. We will explore further the idea of adapting explanations to the explainee's knowledge and expertise level, as well as the explainer's goals and intentions. One such goal of the explainer could be to maximise the trustworthiness of the explanation (Ribeiro et al., 2016). How this aspect is consistently subjectively and objectively measured will be an interesting area of investigation.

Finally, the ExBAN corpus and this study will inform the development of NLG algorithms for NL explanations from graphical representations. We will explore NLG techniques for structured data, such as graph neural networks and knowledge graphs (Koncel-Kedziorski et al., 2019). Thus the corpus and metrics discussed here will contribute to a variety of fields linguistics, cognitive science as well as NLG and Explainable AI.

Acknowledgments

This work was supported by the EPSRC Centre for Doctoral Training in Robotics and Autonomous Systems at Heriot-Watt University and the University of Edinburgh. Clinciu's PhD is funded by Schlumberger Cambridge Research Limited (EP/L016834/1, 2018-2021). This work was also supported by the EPSRC ORCA Hub (EP/R026173/1, 2017-2021) and UKRI Trustworthy Autonomous Systems Node on Trust (EP/V026682/1, 2020-2024).

References

Vijay Arya, Rachel K. E. Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Q. Vera Liao, Ronny Luss, Aleksandra Mojsilovic, Sami Mourad, Pablo Pedemonte, Ramya Raghavendra, John T. Richards, Prasanna Sattigeri, Karthikeyan Shanmugam, Moninder Singh, Kush R. Varshney, Dennis Wei, and Yunfeng Zhang. 2019. One explanation does not fit all: A toolkit and taxonomy of AI explainability techniques. *CoRR*, abs/1909.03012.

Ismaïl Baaj, Jean-Philippe Poli, and Wassila Ouerdane. 2019. Some insights towards a unified semantic representation of explanation for eXplainable artificial intelligence. In *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, pages 14–19. Association for Computational Linguistics.

Anja Belz and Eric Kow. 2009. System building cost vs. output quality in data-to-text generation. In *Proceedings of the 12th European Workshop on Natural Language Generation, ENLG 2009*.

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Anya Belz, Simon Mille, and David M. Howcroft. 2020. Disentangling the properties of human evaluation methods: A classification system to support comparability, meta-evaluation and reproducibility testing. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 183–194, Dublin, Ireland. Association for Computational Linguistics.

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

Daniel Braun, Ehud Reiter, and Advaith Siddharthan. 2018. SaferDrive: An NLG-based behaviour change support system for drivers. *Natural Language Engineering*, 24(4).

Carrie J. Cai, Jonas Jongejan, and Jess Holbrook. 2019. The effects of example-based explanations in a machine learning interface. In *International Conference on Intelligent User Interfaces, Proceedings IUI*, volume Part F147615.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, volume 31, pages 9539–9549. Curran Associates, Inc.

Francisco Javier Chiyah Garcia, David A. Robb, Xingkun Liu, Atanas Laskov, Pedro Patron, and Helen Hastie. 2018. Explainable autonomy: A study of explanation styles for building clear mental models. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 99–108, Tilburg University, The Netherlands. Association for Computational Linguistics.

Miruna-Adriana Clinciu and Helen Hastie. 2019. A Survey of Explainable AI Terminology. *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, pages 8–13.

European Commission. 2018. Article 22 EU GDPR "Automated individual decision-making, including profiling". `https://www.privacy-regulation.eu/en/22.htm`. Accessed on 2021-01-25.

Lorand Dali, Delia Rusu, Blaž Fortuna, Dunja Mladenić, and Marko Grobelnik. 2009. Question answering based on semantic graphs. In *CEUR Workshop Proceedings*, volume 491.

Maartje M.A. De Graaf and Bertram F. Malle. 2017. How people explain action (and autonomous intelligent systems should too). In *AAAI Fall Symposium - Technical Report*, volume FS-17-01 - FS-17-05.

Nina Dethlefs, Heriberto Cuayáhuitl, Helen Hastie, Verena Rieser, and Oliver Lemon. 2014. Cluster-based prediction of user ratings for stylistic surface realisation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics 2014, EACL 2014*, pages 702–711. Association for Computational Linguistics (ACL).

Daniel Deutch, Nave Frost, and Amir Gilad. 2016. Nlprov: Natural language provenance. In *Proceedings of the 42nd International Conference on Very Large Data Bases (VLDB) Endowment*, volume 9, page 1537–1540. VLDB Endowment.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Finale Doshi-Velez and B. Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv: Machine Learning*.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the State-of-the-Art of End-to-End Natural Language Generation: The E2E NLG Challenge. *Computer Speech & Language*, 59:123–156.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA. PMLR.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:1–64.

Dimitra Gkatzia and Saad Mahamood. 2015. A snapshot of NLG evaluation practices 2005 - 2014. In *Proceedings of the 15th European Workshop on Natural Language Generation (ENLG)*, pages 57–60, Brighton, UK. Association for Computational Linguistics.

Ben Goodrich, Vinay Rao, Peter J. Liu, and Mohammad Saleh. 2019. Assessing the factual accuracy of generated text. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Shirley Gregor and Izak Benbasat. 1999. Explanations from intelligent systems: Theoretical foundations and implications for practice. *MIS Quarterly: Management Information Systems*, 23(4).

Kevin A. Hallgren. 2012. Computing Inter-Rater Reliability for Observational Data: An Overview and Tutorial. *Tutorials in Quantitative Methods for Psychology*, 8(1).

Maaike Harbers, Karel Van Den Bosch, and John Jules Ch Meyer. 2009. A study into preferred explanations of virtual agent behavior. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5773 LNAI.

David Hardcastle and Donia Scott. 2008. Can we evaluate the quality of generated text? In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Helen Hastie and Anja Belz. 2014. A comparative evaluation methodology for NLG in interactive systems. In *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Peter A. Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton T. Morrison. 2019. WorldTree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)*, pages 2732–2740. European Language Resources Association (ELRA).

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Vojtěch Kovář, Miloš Jakubíček, and Aleš Horák. 2016. On evaluation of natural language processing tasks: Is gold standard evaluation methodology a good solution? In *ICAART 2016 - Proceedings of the 8th International Conference on Agents and Artificial Intelligence*, volume 2, pages 540–545. SciTePress.

Samantha Krening, Brent Harrison, Karen M. Feigh, Charles Lee Isbell, Mark Riedl, and Andrea Thomaz. 2017. Learning From Explanations Using Sentiment and Advice in RL. *IEEE Transactions on Cognitive and Developmental Systems*, 9(1).

Klaus Krippendorff. 1980. *Metodología de análisis de contenido. Teoría y práctica.* SAGE, 2004.

Sawan Kumar and Partha Talukdar. 2020. NILE : Natural language inference with faithful natural language explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, pages 8730–8742, Online. Association for Computational Linguistics.

Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. 2020. Qed: A framework and dataset for explanations in question answering.

Gerasimos Lampouras and Ion Androutsopoulos. 2013. Using integer linear programming for content selection, lexicalization, and aggregation to produce compact texts from OWL ontologies. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 51–60, Sofia, Bulgaria. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

David B. Leake. 2014. *Evaluating Explanations*. Psychology Press.

Chris van der Lee, Emiel Krahmer, and Sander Wubben. 2017. PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Chin-Yew Lin. 1971. ROUGE: A Package for Automatic Evaluation of Summaries Chin-Yew. *Information Sciences Institute*, 34(12).

Tania Lombrozo. 2007. Simplicity and probability in causal explanation. *Cognitive Psychology*, 55(3):232–257.

Prashan Madumal, Liz Sonenberg, Tim Miller, and Frank Vetere. 2019. A grounded interaction protocol for explainable artificial intelligence. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, volume 2.

Joy Mahapatra, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. 2016. Statistical natural language generation from tabular non-textual data. In *INLG 2016 - 9th International Natural Language Generation Conference, Proceedings of the Conference*.

Steven Mascaro, Ann Nicholson, and Kevin Korb. 2014. Anomaly detection in vessel tracks using Bayesian networks. In *International Journal of Approximate Reasoning*, volume 55.

David Maxwell, Leif Azzopardi, and Yashar Moshfeghi. 2017. A study of snippet length and informativeness behaviour, performance and user experience. In *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Chris McCormick and Nick Ryan. 2019. Bert word embeddings tutorial. Accessed on 2021-01-25.

C. Mellish and R. Dale. 1998. Evaluation in the context of natural language generation. *Computer Speech and Language*, 12(4).

Silvia Metelli and Nicholas Heard. 2019. On bayesian new edge prediction and anomaly detection in computer networks. *Annals of Applied Statistics*, 13(4).

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*.

Tim Miller. 2018. Explanation in Artificial Intelligence: Insights from the Social Sciences. *arXiv preprint arXiv:1706.07269*.

Tim Miller, Piers Hower, and Liz Sonenberg. 2017. Explainable AI: beware of inmates running the asylum. In *Proceedings of the IJCAI 2017 workshop on explainable artificial intelligence (XAI)*, October, page 363.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena" Rieser. 2018. RankME: Reliable human ratings for natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 72–78, New Orleans, Louisiana. Association for Computational Linguistics.

Annemarie Sullivan Palincsar. 1986. The role of dialogue in providing scaffolded instruction. *Educational Psychologist*, 21(1-2):73–98.

Kishore Papineni, Salim Roukos, Todd Ward, Wei-jing Zhu, and Yorktown Heights. 2001. IBM Research Report Bleu : a Method for Automatic Evaluation of Machine Translation. *Science*, 22176:1–10.

Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. 2018. Multimodal Explanations: Justifying Decisions and Pointing to the Evidence. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 8779–8788. IEEE Computer Society.

Gregory Plumb, Denali Molitor, and Ameet Talwalkar. 2018. Model agnostic supervised local explanations. In *Advances in Neural Information Processing Systems*, volume 2018-December.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 2019 Conference of the Association for Computational Linguistics (ACL2019)*.

Nils Reimers and Iryna Gurevych. 2020. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-August-2016.

Carlos Riquelme, George Tucker, and Jasper Snoek. 2018. Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling. In *Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

S. Russell. 2019. *Human Compatible: Artificial Intelligence and the Problem of Control*. Penguin Publishing Group.

Sasan Saqaeeyan, Hamid Haj Seyyed Javadi, and Hossein Amirkhani. 2020. Anomaly detection in smart homes using Bayesian networks. *KSII Transactions on Internet and Information Systems*, 14(4).

Rosario Scalise, Stephanie Rosenthal, and Siddhartha Srinivasa. 2017. Natural language explanations in human-collaborative systems. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '17, page 377–378, New York, NY, USA. Association for Computing Machinery.

Natalie Schluter. 2017. The limits of automatic summarisation according to ROUGE. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 41–45, Valencia, Spain. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

William Smith. 2008. Does Gender Influence Online Survey Participation? A Record-Linkage Analysis of University Faculty Online Survey Response Behavior. Accessed on 2021-01-25.

Kacper Sokol and Peter A. Flach. 2019. Counterfactual Explanations of Machine Learning Predictions: Opportunities and Challenges for AI Safety. In *Workshop on Artificial Intelligence Safety 2019 co-located with the Thirty-Third AAAI Conference on Artificial Intelligence 2019 (AAAI-19), Honolulu, Hawaii, January 27, 2019*, volume 2301 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Zaid Tashman, Christoph Gorder, Sonali Parthasarathy, Mohamad M. Nasr-Azadani, and Rachel Webre. 2020. Anomaly detection system for water networks in northern ethiopia using bayesian inference. *Sustainability (Switzerland)*, 12(7).

Nicole Tourigny and Laurence Capus. 1998. Learning summarization by using similarities. *International Journal of Phytoremediation*, 21.

Adam White and Artur S. d'Avila Garcez. 2019. Measurable counterfactual local explanations for any classifier. *CoRR*, abs/1908.03020.

Yosef Ardhito Winatmoko and Masayu Leylia Khodra. 2013. Automatic Summarization of Tweets in Providing Indonesian Trending Topic Explanation. *Procedia Technology*, 11.

Bowen Xu, Zhenchang Xing, Xin Xia, and David Lo. 2017. AnswerBot: Automated generation of answer summary to developers' technical questions. In *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2017)*.

Xinnuo Xu, Ondřej Dušek, Jingyi Li, Verena Rieser, and Ioannis Konstas. 2020. Fact-based content weighting for evaluating abstractive summarisation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5071–5081, Online. Association for Computational Linguistics.

Changhe Yuan, Heejin Lim, and Tsai-Ching Lu. 2011. Most relevant explanation in bayesian networks. *The Journal of Artificial Intelligence Research*, 42(1):309–352.

Jeffrey C. Zemla, Steven Sloman, Christos Bechlivanidis, and David A. Lagnado. 2017. Evaluating everyday explanations. *Psychonomic Bulletin & Review*, 24(5):1488–1500.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.

Adversarial Stylometry in the Wild: Transferable Lexical Substitution Attacks on Author Profiling

Chris Emmery
CSAI, Tilburg University
CLiPS, University of Antwerp
`cmry@pm.me`

Ákos Kádár
Borealis AI
`akos.kadar@borealisai.com`

Grzegorz Chrupała
CSAI, Tilburg University
`g.a.chrupala@uvt.nl`

Abstract

Written language contains stylistic cues that can be exploited to automatically infer a variety of potentially sensitive author information. Adversarial stylometry intends to attack such models by rewriting an author's text. Our research proposes several components to facilitate deployment of these adversarial attacks in the wild, where neither data nor target models are accessible. We introduce a transformer-based extension of a lexical replacement attack, and show it achieves high transferability when trained on a weakly labeled corpus—decreasing target model performance below chance. While not completely inconspicuous, our more successful attacks also prove notably less detectable by humans. Our framework therefore provides a promising direction for future privacy-preserving adversarial attacks.

1 Introduction

The widespread use of machine learning on consumer devices and its application to their data has sparked investigation of security and privacy researchers alike in correctly handling sensitive information (Edwards and Storkey, 2016; Abadi et al., 2016b). Natural Language Processing (NLP) is no exception (Fernandes et al., 2019; Li et al., 2018); written text can contain a plethora of author information—either consciously shared or inferable through stylometric analysis (Rao et al., 2000; Adams, 2006). This characteristic is fundamental to author profiling (Koppel et al., 2002), and while the field's main interest pertains to the study of sociolinguistic and stylometric features that underpin our language use (Daelemans, 2013), herein simultaneously lie its dual-use problems. Author profiling can, often with high accuracy, infer an extensive set of (sensitive) personal information, such as age, gender, education, socio-economic status, and mental health issues (Eisenstein et al., 2011; Alowibdi et al., 2013; Volkova et al., 2014; Plank and Hovy, 2015; Volkova and Bachrach, 2016). It therefore potentially exposes anyone sharing written online content to unauthorized information collection through their writing style. This can prove particularly harmful to individuals in a vulnerable position regarding e.g., race, political affiliation, or mental health.

Privacy-preserving defenses against such inferences can be found in the field of adversarial[1] stylometry. Our research[2] concerns the obfuscation subtask, where the aim is to rewrite an input text such that the style changes, and stylometric predictions fail. It is part of a growing body of research into adversarial attacks on NLP (Smith, 2012), which various modern models have proven vulnerable to; e.g., in neural machine translation (Ebrahimi et al., 2018), summarization (Cheng et al., 2020), and text classification (Liang et al., 2018).

Adversarial attacks on NLP are predominantly aimed at demonstrating vulnerabilities in existing algorithms or models, such that they might be fixed, or explicitly improved through adversarial training. Consequently, most related work focuses on white or black-box settings, where all or part of the target model is accessible (e.g., its predictions, data, parameters, gradients, or probability distribution) to fit an attack. The current research, however, does not intend to improve the targeted models; rather, we want to provide the attacks as tools to protect online privacy. This introduces several constraints over other NLP-based adversarial attacks, as it calls for a realistic, in-the-wild scenario of application.

Firstly, authors seeking to protect themselves from stylometric analysis cannot be assumed to be

---

[1] These are adversarial attacks on models making stylometric predictions, not to be confused with adversarial learning.

[2] All code, data, and materials to fully reproduce the experiments are openly available at `https://github.com/cmry/reap`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2388–2402
April 19 - 23, 2021.

knowledgeable about the target architecture, nor to have access to suitable training data (as the target could have been trained on any domain). Hence, we cannot optimally tailor attacks to the target, and need an accessible method of mimicking it to evaluate the obfuscation success. To facilitate this, we use a so-called substitute model, which for our purposes is an author profiling classifier trained in isolation (with its own data and architecture) that informs our attacks. Attacks fitted on substitute models have been shown to transfer their success when targeting models with different architectures, or trained on other data, in a variety of machine learning tasks (Papernot et al., 2016). The effectiveness of an attack fitted on a substitute model when targeting a 'real' model is then referred to as *transferability*, which we will measure for the obfuscation methods proposed in the current research.

Secondly, for an obfuscation attack to work in practice (e.g., given a limited post history), it should suggest relevant changes –to– the author's writing *on a domain of their choice*. This implies the substitute models should be fitted locally, and therefore need to meet two criteria: reliable access to labeled data, and being relatively fast and easy to train. To meet the first criterion, the current research focuses on gender prediction, as: i) Twitter corpora annotated with this variable are by far the largest (and most common), ii) author profiling methods typically use similar architectures for different attributes; therefore, the generalization of attacks to other author attributes can be assumed to a large extent, and, most importantly, iii) Beller et al. (2014) and Emmery et al. (2017) have shown that through distant labeling, a representative corpus for this task can be collected in under a day. This allows us to measure transferability of attacks fitted using realistically collected distant corpora to models using high-quality hand labeled corpora.

As for the attacks, we focus on lexical substitution of content words strongly related to a given label, as those have been shown to explain a significant portion of the accuracy of stylometric models (see e.g., Rao et al., 2000; Burger et al., 2011; Sap et al., 2014; Rangel et al., 2016). To that effect, we extend the substitution attack of Jin et al. (2020) and apply it to author attribute obfuscation. Specifically, we explore the potential of training a simple (as to meet the speed criterion), non-neural substitute model f' to indicate relevant words to perturb, where retaining the original meaning is prioritized.

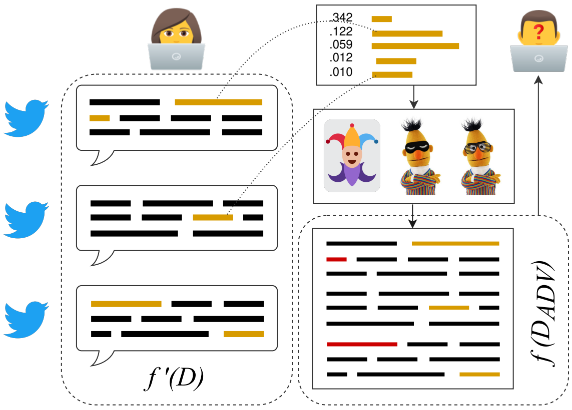


Figure 1: Obfuscation scenario: model f' trains on tweet batches, an omission score is used to determine and rank the words according to their classification contribution. These are then passed to either TextFooler, Masked BERT, or Dropout BERT to suggest top-k replacement candidates. From these, a selection is made based on their class probability change on $f'(D)$. Finally, f is evaluated on the perturbed tweets D_{ADV}.

Two transformer-based models are introduced to the framework to propose and rank lexical substitutions towards a change in the predictions of f'. We evaluate if the attacks on f' transfer across corpora, architectures, and a separately trained target model f (see Figure 1). Finally, we measure the quality of changes using automatic evaluation metrics, and conduct an human evaluation that focuses on detection accuracy of the attacks.

2 Related Work

Stylometry, the study of (predominantly) writing style, dates back several decades (Mosteller and Wallace, 1963), and has seen increased accessibility through the introduction of statistical models (see surveys by Holmes, 1998; Neal et al., 2017) and machine learning (e.g., Matthews and Merriam, 1993; Merriam and Matthews, 1994). Computational stylometry distinguishes several subtasks such as determining (Baayen et al., 2002) and verifying author identity (Koppel and Schler, 2004), and author profiling (Argamon et al., 2005); e.g., predicting demographic attributes. Adversarial stylometry (as conceptualized by Brennan et al., 2012) intends to subvert these inferences by changing an author's text through imitation, or, as pertains to our research, the obfuscation of writing style (Kacmarcik and Gamon, 2006; Caliskan et al., 2018; Le et al., 2015; Xu et al., 2019).

These changes, or perturbations, can be produced in several ways, and the task is therefore often conflated with paraphrasing (Reddy and Knight, 2016), style transfer (Kabbara and Cheung, 2016), and generating adversarial samples or triggers (Zhang et al., 2020b). Regardless of the employed method, the main challenge of obfuscation lies in retaining the original meaning of an input text; its written language medium limits any perturbations to discrete outputs, and unnatural discrepancies are significantly better discernible by humans than, say, a few pixel changes in an image. An additional, persistent limitation is the absence of evaluation metrics that guarantee complete preservation of the original meaning of the input whilst changes remain unnoticed (Potthast et al., 2016). This not only inhibits automatic evaluation of obfuscation, but all natural language generation research (Novikova et al., 2017)—placing an emphasis on human evaluation (van der Lee et al., 2019).

It is perhaps for this reason that most obfuscation work uses heuristically-driven, controlled changes such as splitting or merging words or sentences, removing stop words, changing spelling, punctuation, or casing (see e.g., Karadzhov et al., 2017; Eger et al., 2019). These specific attacks are typically easier to mitigate through preprocessing (Juola and Vescovi, 2011). Obfuscation through lexical substitution (Mansoorizadeh et al., 2016; Bevendorff et al., 2019, 2020) provides a middle ground of control, semantic preservation and attack effectiveness; however, they might prove less effective against models relying on deeper stylistic features (e.g. word order, part-of-speech (POS) tags, or reading complexity scores). End-to-end systems have been employed for similar purposes (Shetty et al., 2018; Saedi and Dras, 2020), or to rewrite entire phrases (Emmery et al., 2018; Bo et al., 2019) using (adversarially-driven) autoencoders. Such attacks seem less common, and provide less control over the perturbations and semantic consistency.

Our work does not assume the attacks to run end-to-end, but with a hypothetical human in the loop. We further opt for techniques that are more likely to find strong semantic mirrors to the original text while making minimal changes. A substitute model (the algorithm, hyper-parameters, and output of which an author can manipulate as desired) is employed to indicate candidate replacement words, and our attacks suggest and rank those against this substitute. Moreover, prior work typically attacks adversaries trained on the same data, whereas we add a transferability measure. Lastly, while author identification has been investigated in the wild (Stolerman et al., 2013), our work is, to our knowledge, the first to make a conscious effort towards realistic applicability of obfuscation techniques.

3 Method

Our attack framework extends TextFooler (TF, Jin et al., 2020) in several ways. First, a substitute gender classifier is trained, from which the logit output given a document is used to rank words by their prediction importance through an omission score (Section 3.1). For the top most important words, substitute candidates are proposed, for which we add two additional techniques (Section 3.2). These candidates can be checked and filtered on consistency with the original words (by their POS tags, for example), accepted as-is, or re-ranked (Section 3.3). For the latter, we add a scoring method. Finally, the remaining candidates are used for iterative substitution until TF's stopping criterion is met (i.e., the prediction changes, or candidates run out).

3.1 Target Word Importance

We are given a target classifier f, substitute classifier f', a document D consisting of tokens D_i, and a target label y. Here, f' is trained on some corpus X, and receives an author's new input text D, where the author provides label y. We denote a class label as $\bar{y}$ if $f'(D)$ predicts anything but y. Our perturbations form adversarial input D_{ADV}, that intends to produce $f'(D_{\text{ADV}}) = \bar{y}$, and thereby implicitly $f(D_{\text{ADV}}) = \bar{y}$. Note that we only submit D to f for evaluating the attack effectiveness, and it is never used to fit the attack itself.

To create D_{ADV}, a minimum number of edits is preferred, and thus we rank all words in D by their omission score (similar to e.g., Kádár et al., 2017) according to f' (`omission_score` in Algorithm 1). Let $D_{\setminus i}$ denote the document after deleting D_i, and $o_y(D)$ the logit score by f'. The omission score is then given by $o_y(D) - o_y(D_{\setminus i})$, and used in an importance score I of token D_i, as:

$$I_{D_i} = \begin{cases} o_y(D) - o_y(D_{\setminus i}), \\ \quad \text{if } f'(D) = f'(D_{\setminus i}) = y. \\ o_y(D) - o_y(D_{\setminus i}) + o_{\bar{y}}(D) - o_{\bar{y}}(D_{\setminus i}), \\ \quad \text{if } f'(D) = y, f'(D_{\setminus i}) = \bar{y}, y \neq \bar{y}. \end{cases} \tag{1}$$

With I_{D_i} calculated for all words in D, the top k ranked tokens are chosen as target words T.

ALGORITHM 1: Obfuscation by lexical replacement.

```
Input  : $f'$ – substitute model
         $D = \{w_0, w_1, \dots, w_n\}$ – document
         $y$ – target label
         checks – apply checks (bool)
         k – target max $k$-amount words
Output : $D_{\text{ADV}}$ – obfuscated document
for $D_i \in D$ do
    // via Equation 1
    $I_{D_i} \leftarrow$ omission_score($f', y$)
$T \leftarrow$ top_k(argsort_desc($D, I_{D_i}$ scores), k)
$D_{\text{ADV}} = D$
for $t \in T$ do
    // substitution attack on $t$
    $C_t \leftarrow$candidates($t$)
    $A = (D_{\text{ADV}\,1:i-1}, C_{t,j}, D_{\text{ADV}\,i+1:n})_{1 \leq j \leq |C_t|}$
    $\bar{A} =$ filter/rank($D, A; t$, checks)
    // test attack success on $f'$
    for $D' \in \bar{A}$ do
        if $\arg\max o_y(D') \neq y$ then
            return $D_{\text{ADV}} = D'$
        else if $o_y(D') < o_y(D_{\text{ADV}})$ then
            $t$ in $D_{\text{ADV}}$ is replaced with $c$ from $D'$
return $D_{\text{ADV}}$
```

3.2 Lexical Substitution Attacks

Four approaches to perturb a target word $t \in T$ are considered in our experiments. These operations are referred to as `candidates` in Algorithm 1.

Synonym Substitution (WS) This TF-based substitution embeds t as $\boldsymbol{t}$ using a pre-trained embedding matrix $\boldsymbol{V}$. C_t is selected by computing the cosine similarity between $\boldsymbol{t}$ and all available word-embeddings $\boldsymbol{w} \in \boldsymbol{V}$. We denote cosine similarity with $\Lambda(\boldsymbol{t}, \boldsymbol{w})$. A threshold δ is used to keep only reliable candidates $\Lambda(\boldsymbol{t}, \boldsymbol{w}) > \delta$.

Masked Substitution (MB) The embedding-based substitutions can be replaced by a language model predicting the contextually most likely token. BERT (Devlin et al., 2019)—a bi-directional encoder (Vaswani et al., 2017) trained through masked language modeling and next-sentence prediction—makes this fairly trivial. By replacing t with a mask, BERT produces a top-k most likely C_t for that position. Implementing this in TF does imply each previous substitution of t might be included in the context of the current one. This method of contextual replacement has two drawbacks: i) semantic consistency with the *original* word is not guaranteed (as the model has no knowledge of t), and ii) the replaced context means semantic drift can occur, as all subsequent substitutions follow the new, possibly incorrect context.

Dropout Substitution (DB) A method to circumvent the former (i.e., BERT's masked prediction limitations for lexical substitution), was presented by Zhou et al. (2019). They apply dropout (Srivastava et al., 2014) to BERT's internal embedding of target word t before it is passed to the transformer—zeroing part of the weights with some probability. The assumption is that C_t (BERT's top-k) will contain candidates closer to the original t than the masked suggestions.

Heuristic Substitution To evaluate the relative performance of the techniques we described before, we employ several heuristic attacks as baselines. In the order of Table 3: 1337-speak: converts characters to their leetspeak variants, in a similar vein to e.g. diacritic conversion (Belinkov and Bisk, 2018). Character flip: inverts two characters in the middle of a word, which was shown to least affect readability (Rayner et al., 2006). Random spaces: splits a token into two at a random position.

3.3 Candidate Filtering and Re-ranking

Given C_t, either all, or only the highest ranked candidate can be accepted as-is. Alternatively, all D' can be filtered by submitting them to checks, or re-ranked based on their semantic consistency with D. These operations are referred to as `rank/filter` in Algorithm 1—both of which can be executed.

Part-of-Speech and Document Encoding TF employs two checking components: first, it removes any c that has a different POS tag than t. If multiple D' exist so that $f'(D') = \bar{y}$, it selects the document D' which has the highest cosine similarity to the Universal Sentence Encoder (USE) embedding (Cer et al., 2018) of the original document D. If not, the D' with the *lowest* target word omission score is chosen (as per TF's method).

BERT Similarity Zhou et al. (2019) use the concatenation of the last four layers in BERT as a sentence's contextualized representation $\boldsymbol{h}$. We apply this in both Masked (MB) and Dropout (DB) BERT to re-rank all possible D' by embedding them. Given document D, target t, and perturbation candidate document D', C_t would be ranked via an embedding similarity score:

$$\text{SIM}\left(D, D'; t\right) = \sum_{i}^{n} w_{i,t} \times \Lambda\left(\boldsymbol{h}\left(D_i|D\right), \boldsymbol{h}(D'_i|D')\right) \qquad (2)$$

	AUTHORS	TWEETS	FEMALE	MALE	TRAIN	TEST	TOKENS	TYPES	AVG SIZE
Huang et al.	37,929	47,211	26,758	20,453	30,602	7,651	935,062	46,600	28
Emmery et al.	6,610	16,788,612	61,736	32,900	75,918	18,718	146,736,657	9,942,399	301
Volkova et al.	4,620	12,226,859	32,376	26,708	47,298	11,777	67,186,535	7,836,539	269

Table 1: Corpus statistics indicating the number of authors, tweets, female and male labels, the size of the train and test splits, number of types (unique words) and tokens (total words), and average tokens per document (avg size).

where $\boldsymbol{h}\,(D_i|D)$ is BERT's contextualized representation of the i^{th} token in D, and $w_{i,t}$ is the average self-attention score of all heads in all layers ranging from the i^{th} token with respect to t in D.[3]

4 Experiment

4.1 Data

We use three author profiling sets (see Table 1 for statistics) that are annotated for binary gender classification (male or female): first, that of Volkova et al. (2015) which was collected through annotating 5,000[4] English Twitter profiles by crowdsourcing via Mechanical Turk. This can be considered a 'random' sample of Twitter profiles, and is therefore the most unbiased set of the three. Hence, we consider it the most representative of an author profiling set, and employ this as training split (80%) for f, and test split for our attacks (20%).

The second is the English portion of the Multilingual Hate Speech Fairness corpus of Huang et al. (2020), which was collected with a different objective than author profiling. It was aggregated from existing hate speech corpora (by Waseem and Hovy, 2016; Waseem, 2016; Founta et al., 2018)—which were largely bootstrapped with lookup terms, selection of frequently abusive users, etc.—and annotated post-hoc with demographic information. The collection did not focus on profiles, and most authors are only associated with a single tweet. This can cause a significant domain shift compared to general author profiling. However, it can be seen as freely available (noisy) data.

Lastly, we include a weakly labeled author profiling corpus by Emmery et al. (2017), collected through English keyword look-up for self-reports—similar to Beller et al. (2014). This corpus likely includes incorrect labels, but was collected in less than a day, making it an ideal candidate for realistic access to (new) data to fit the substitute model.

Preprocessing & Sampling All three corpora were tokenized using spaCy[5] (Honnibal and Montani, 2017). Other than lowercasing, allocating special tokens to user mentions and hashtags (# and text were split), and URL removal, no additional preprocessing steps were applied. Every author timeline was divided into chunks for a maximum of 100 tweets (i.e., some contain less) to form our documents, implying a maximum of 25 instances per author (some contain one, 2,500 is the API history limit). From the test set, the last[6] 200 instances were sampled for the attack (110 male, 90 female). While fairly small, this sample does reflect a realistic attack duration and timeline size, as they would be executed for a single profile.

4.2 Attacks

For the extension of TF, we re-implemented code[7] by Jin et al. (2020) to work with Scikit-learn[8] (Pedregosa et al., 2011). For their synonym substitution component, we similarly used counter-fitted embeddings by Mrkšić et al. (2016) trained on Simlex-999 (Hill et al., 2015). The USE (Cer et al., 2018) implementation uses TensorFlow[9] (Abadi et al., 2016a) as back-end, and all BERT-variants were implemented in Hugging Face's[10] Transformers library (Wolf et al., 2020) with PyTorch[11] (Paszke et al., 2019) as back-end.

We adopt the same parameter settings as Jin et al. (2020) throughout our TF experiments: they set N (considered synonyms) and δ (cosine similarity minimum) empirically to 50 and 0.7 respectively. For MB and DB, we capped T at 50 and top-k at 10 (to improve speed). For DB, we follow Zhou et al. (2019) and set the dropout probability to 0.3.

---

[3] Zhou et al. (2019) additionally use a proposal score for finding T that we replaced with the omission score.

[4] Profile counts in the current work differ due to collection limitations (e.g., removed accounts).

[5] `https://spacy.io`

[6] As the datasets are not shuffled to avoid overfitting on author-specific features, a few documents of the same author might spill from the train into the test split; this avoids incorporating those in our attack sample.

[7] `https://github.com/jind11/TextFooler`

[8] `https://scikit-learn.org/`

[9] `https://tensorflow.org/`

[10] `https://huggingface.co/`

[11] `https://pytorch.org/`

data	Huang, Emmery, Volkova
importance	Omission score
attack	Heuristics, TextFooler, Masked BERT, Dropout BERT
model	Logistic Regression, N-GrAM
ranking	None, POS + USE, BERT Sim

Table 2: Grid of possible experimental configurations.

4.3 Models

For f and f' we require (preferably fast) pipelines that achieve high accuracy on author profiling tasks, and are sufficiently distinct to gauge how well our attacks transfer across architectures, rather than solely across corpora. As state-of-the-art algorithms have not yet proven to be sufficiently effective for author profiling (Joo et al., 2019) we opt for common n-gram features and linear models.

Logistic Regression Logistic Regression (LR) trained on tf·idf using uni and bi-gram features proved a strong baseline in author profiling in prior work. The simplicity of this classifier also makes it a substitute model that can realistically be run by an author. No tuning was performed: C is set to 1.

N-GrAM The New Groningen Author-profiling Model (N-GrAM) from Basile et al. (2018), was proposed as a highly effective—simple—model that outperforms more complex (neural) alternatives on author profiling with little to no tuning. It uses tf·idf-weighted uni and bi-gram token features, character hexa-grams, and sublinearly scaled tf $(1 + \log(\text{tf}))$. These features are then passed to a Linear Support Vector Machine (Cortes and Vapnik, 1995; Fan et al., 2008), where $C = 1$.

4.4 Experimental Setup

To summarize (and see Table 2), the experiment is conducted as follows: the substitute target model (f')—LR for all experiments—is fit on a given corpus. The real target model (f, either LR or N-GrAM) is always fit on the corpus of Volkova et al. (2015). To evaluate the attacks, a 200-instance sample is used. Target words are ranked via omission scores from f', fed to either our Heuristics, TF, MB, or DB attacks. The heuristics directly change the target words, while the rest outputs a ranked set of replacement candidates. The latter can either be evaluated against f' through the TF pipeline, or the Top-1 candidate is returned. Filtering can be applied through POS/USE for semantic similarity and POS compatibility checks (Check), or not (~~Check~~).

		test = Volkova et al.					
	LR $f' \rightarrow$	Huang et al.		Emmery et al.		Volkova et al.	
	$f \rightarrow$	LR	NG	LR	NG	LR	NG
	none	.885	.940	.885	.940	.885	.940
Heuristic	1337	.770	.850	.775	.835	.715	.860
	flip	.900	.950	.885	.905	.840	.905
	space	.845	.925	.760	.870	.720	.850
Top-1	WS	.825	.930	.805	.890	.750	.915
	MB	.655	.905	.595	.785	.145	.410
	DB	.625	.895	.575	.785	.210	.530
~~Check~~	WS	.540	.855	.355	.670	**.000**	**.009**
	MB	**.415**	.790	**.120**	**.420**	**.000**	.085
	DB	.430	**.775**	.175	.430	**.000**	.085
Check	TF	.705	.920	.780	.910	.375	.700
	TF + MB	.640	.880	.760	.890	.380	.725
	TF + DB	.650	.885	.755	.890	.435	.715

Table 3: Post-attack accuracy scores (below chance (55%) = better) of f on a test sample from the Volkova et al. corpus. Left, the attack conditions: heuristics, top-1 synonym, applying POS and USE similarity checks, or not applying those checks (~~Check~~). Splits per training corpus are noted for f' (always Logistic Regression (LR)). As target model, either LR, or N-GrAM (NG) was used. The substitution attacks are TextFooler (TF), Masked (MB) and Dropout BERT (DB). If TF's stopping criterion was used, TF + is noted. Word Similarity (WS), reflects the TF pipeline without checks.

Note that we are predominantly interested in transferability, and would therefore like to test as many combinations of data and architecture access limitations as possible. If we assume an author does not have access to the data, the substitute classifier is trained on any other data than the Volkova et al. corpus. If we assume the author does not know the target model architecture, the target model is N-GrAM (rather than LR). A full model transfer setting (in both data and architecture) will therefore be, e.g.: data f' = Emmery et al., data f = Volkova et al., f' = LR, and f = NGrAM. Finally, for comparison to an optimal situation, we test a setting where we do have access to the adversary's data.

4.5 Evaluation

Metrics The obfuscation success is measured as any accuracy score below chance level performance, which given our test sample is 55%. We would argue that random performance is preferred in scenarios where the prediction of the opposite label is undesired. For the current task, however, any accuracy drop to around or lower than chance level satisfies the conditions for successful obfus-

	Volkova et al. →	TRAIN	TEST
TRAIN	Huang et al.	0.640	0.620
	Emmery et al.	0.725	0.890

Table 4: Gender prediction accuracies of the substitute models f' on train and test splits of f.

cation.[12] To evaluate the semantic preservation of the attacked sentences, we calculate both METEOR (Banerjee and Lavie, 2005; Lavie and Denkowski, 2009) using `nltk`,[13] and BERTScore (Zhang et al., 2020a) between D and D_{ADV}. METEOR captures flexible uni-gram token overlap including morphological variants, and BERTScore calculates similarities with respect to the sentence context.

Human Evaluation For the human evaluation, we sampled 20 document pieces (one or more tweets) for each attack type in the best performing experimental configuration. A piece was chosen if it satisfied these criteria: i) contains changes for all three attacks, ii) consists of at least 15 words (excluding emojis and tags), and iii) does not contain obvious profanity.[14] All 60 document pieces of the three models were shuffled, and the 20 original versions were appended at the end (so that 'correct' pieces were seen last). Each substitute model therefore has 80 items for evaluation.

While in prior work it is common to rate semantic consistency, fluency, and label a text (see e.g., Potthast et al., 2016; Jin et al., 2020), our Twitter data are too noisy (including many spelling and grammar errors in the originals), and document batches too long to make this a feasible task. Instead, our six participants (three per substitute) were asked to indicate if: a) a sentence was artificially changed, and if so, b) indicate one word that raised their suspicion. This way, we can evaluate which attack produces the most natural sentences, and the least obvious changes to the input.

The items were rated individually; the human evaluators did not know beforehand that different versions of the same sentences were repeated, nor that the originals were shown at the end. All participants have a university-level education, a high English proficiency, and are familiar with the domain of the data. Several example ratings of the same sentence can be found in Table 6.

5 Results

5.1 Domain Shift

As we alluded to in Section 4.1, both corpora used to train our substitute models were in fact not reference corpora for author profiling, and can therefore be considered as suboptimal, disjoint domains. The Huang et al. corpus in particular shows a strong domain shift (see Table 4) for both training and test sets. The distantly labeled Emmery et al. corpus achieves 7.5% more accuracy on the train split of Volkova et al., and test performance is significantly higher (27%). We might therefore expect better obfuscation performance from the latter.

5.2 Baselines

The results for all attacks are shown in Table 3. Note that these are performances for f; therefore, when no attacks are applied (none), the performance for both substitute corpora stays the same (as those only influence the attacks). For the heuristic attacks, 1337 seems to make the more robust baseline; outperforming some of the other settings—even on transferability. A surface-level advantage is that this attack has a minor impact on readability (when applied conservatively) and does not change semantics; however, the heuristic attacks are fairly simple to mitigate in preprocessing (Juola and Vescovi, 2011) and through character features (as shown by the performance of the N-GrAM model). For transferability, we evidently need to do more than simply trying to convert words to be out-of-vocabulary (OOV) with noise. While it can be argued the heuristics could change all words, shifting everything OOV would not be robust; the target model side could easily spot the anomalous input and might act (e.g., reject) accordingly.

5.3 Attack Transferability

Transferability can be assessed by comparing the LR and N-GrAM (NG) columns. Globally it can be observed that the substitute models trained on the Emmery et al. corpus systematically outperform those trained on Huang et al.; both for the settings where the adversary's architecture is known (LR), and where it is unknown (NG). This matches our

---

[12] If an attack drops accuracy to 0%, this effectively flips (in case of a binary label) the label. This label might *also* be undesired by the author (e.g., being classified as having polar opposite political views). This implies the target model being maximally unsure about the classification is desirable.

[13] `https://www.nltk.org/_modules/nltk/translate/meteor_score.html`

[14] To avoid exposing the raters to overly toxic content, blatant examples were filtered using a keyword list. Some minor examples remained, for which we added a disclaimer.

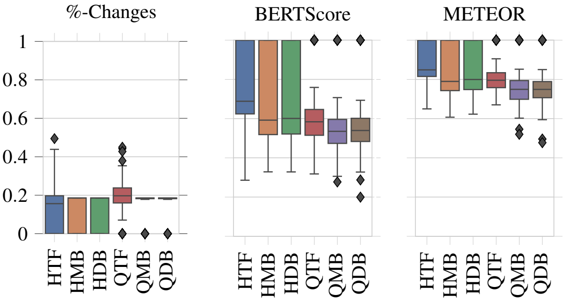


Figure 2: Rewrite metrics for the three different attacks: TextFooler (TF), Masked (MB) and Dropout BERT (DB)—split by corpora Huang et al. (H), Emmery et al. (Q). Shown are the relative number of changes (by document length), BERTScore (F_1), and METEOR (MET) with respect to the original document.

expectations from the observed domain shift. Our results also show that a noticeable decrease in obfuscation performance occurs (10-30% increased target model performance) when the attacks are transferred to different data and another model. In contrast, as can be observed from the last two columns in Table 3, in a practically unrealistic setting where the model and data are available, the obfuscation is specifically tailored to known weaknesses and therefore highly effective.[15]

5.4 Transformer Performance

Looking at the Top-1, ~~Check~~ and Check brackets (Table 3), other than the BERT-based models having higher success of transferability than TF, they also retain obfuscation success; deteriorating the target model's performance to lower than chance level (55%) for the settings not using additional checks. This comparison also demonstrates the synonym ranking to work (Top-1 vs. ~~Check~~ and Check), and the Check condition to be too restrictive; attaining lower attack power, and low transferability. This is further illustrated by the %-changes shown in Figure 2. Comparing the MB and DB variants, their performance seems almost identical, with masking having a slight advantage. As Zhou et al. (2019) argued, applying dropout should produce words that are closer to the original (compared to MB), which might affect obfuscation performance. Additionally, the BERT similarity ranking (described in Section 3.3) applied to the Masked substitution candidates could have some beneficial effect. This will have to be studied in more detail using the output evaluations.

		Huang et al.			Emmery et al.		
	ORG	TF	MB	DB	TF	MB	DB
ALTERED	.888	.967	.633	.783	.950	**.617**	.633
WORD	-	.950	.583	.700	.867	**.433**	**.433**

Table 5: Human accuracy scores of predicting if a text was altered, and guessing the attacked word (lower is better). All substitute models are those with the ~~Check~~ setting, trained on different corpora (i.e., different words are attacked per training corpus). ORG indicates correct prediction of the originals.

Rewrite Metrics The metrics in Figure 2 show a common initial limitation in their application to this task: the more frequent an attack makes no changes, the higher the automatic evaluation metrics (BERTScore, METEOR). Hence, to compare models, these scores need to be considered in light of the obfuscation performance, and related work. It can be observed that with consistently higher changes, MB and DB score lower on semantic consistency than TF. However, between MB and DB, and TF for the Emmery et al. corpus, these differences are minor. Furthermore, despite being fit on a different domain, these scores are comparable to prior obfuscation work (e.g., Shetty et al. (2018) show METEOR scores between 0.69 and 0.79).

Human Evaluation The results in Table 5 reflect the same trend that can be observed in Table 3; high obfuscation success seems to result in higher human error when predicting if a sentence was obfuscated. Conversely, it seems that despite higher semantic consistency scores, the original TF pipeline is easier to detect. This can be attributed to the number of spelling and grammar errors the model makes without its additional checks. Furthermore, the 11% error in identifying the original sentences also reflects some expected margin of error in this task, as our Twitter data is inherently noisy. Finally, while these results are in line with the obfuscation success, and are lower than detectability scores in related work (Mahmood et al., 2020), they also indicate that the models are still detectable above chance-level. Given three alternatives (including the original), performance should be 25% or lower to indicate no intrusive changes are made to text (that are not semantically coherent or not inconspicuous enough—both metrics used by Potthast et al., 2016). Therefore, while the presented approaches are effective, and realistically transferable, there is room for improvement for practical applicability.

---

[15] Jin et al. (2020) found similar drops to 0% accuracy with a comparable percentage of changes for word-level models.

ORG	ready to go home already . a better relationship with god 💙🙌 . i need another job asap .
HTF	**loan** to go **houses** already . a **improved relations** with **jesus** 💙🙌 . i **should** another **labour** asap .
HMB	ready to go **on** already . a better relationship with god 💙🙌 . i need another **guy man** .
HDB	ready to go **somewhere** already . a better relationship with god 💙🙌 . i need another **position vs** .
ORG	trump criticizes kim jong un after missile launch : ' does this guy have anything better to do ? ' .
HTF	**tramp criticized kam yung jt** after **rocket start** : ' does this **boyfriend** have anything **best** to do ? ' .
HMB	trump criticizes **ha woman congressman** after **campaign** launch : ' does this **book** have anything **else** to do ? ' .
HDB	trump criticizes **in at sin** after **bomb** launch : ' does this **kid** have anything **less** to do ? ' .

Table 6: Example ratings of different attacks (not shown together to the human evaluators) on two sentences with varying semantic consitency and human detection accuracy. In the first example, HMB was marked unaltered by all raters, HDB by the majority, and HTF by none. In the second, only HDB was marked unaltered, by only one rater. Attacked words are marked in bold, guessing any one of these would count as correctly identifying the attack.

6 Discussion and Future Work

We have demonstrated the performance of author attribute obfuscation under a realistic setting. Using a simple Logistic Regression model for candidate suggestion, trained on a weakly labeled corpus collected in a day, the attacks successfully transferred to different data and architectures. This is a promising result for future adversarial work on this task, and its practical implementation.

It remains challenging to automatically evaluate how invasive the required number of changes are for successful obfuscation—particularly to an author's message consistency as a whole. However, in practice such considerations could be left up to the author. In this human-in-the-loop scenario, a more extensive set of candidates could be suggested, and their effect on the substitute model shown interactively. This way, the attacks can be manually tuned to find a balance of effectiveness, inconspicuousness, and to guarantee semantic consistency. It would also show the author how their writing style affects potential future inferences.

Regarding the performance of the attacks: we demonstrated the general effectiveness of contextual language models in retrieving candidate suggestions. However, the quality of those candidates might be improved with more extensive rule-based checks; e.g., through deeper analyses using parsing. Nevertheless, such venues leave us with a core limitation of rewriting language, and therefore more broadly NLP: while the Masked attacks seemed more successful in our experiments, after manual inspection of the perturbations Dropout was found to often be semantically closer (see also Table 6)—which was not reflected in the human evaluation. This begs the question if *any* automated approach, evaluated under the current limitations of semantic consistency metrics, could realistically optimize for both obfuscation and inconspicuousness.

As such, we would argue that future work should focus on making as few perturbations as possible, retaining only the minimum amount of required obfuscation success. Given this, the other constraints become less relevant; one could generate short sentences (e.g., a single tweet) that might be semantically or contextually incorrect, but if it is a message in a long post history, it will hardly be detectable or intrusive. This would require certain triggers (as demonstrated by Wallace et al. (2019) for example), and ascertaining how well they transfer.

7 Conclusion

In our work, we argued realistic adversarial stylometry should be tested on transferability in settings where there is no access to the target model's data or architecture. We extended previous adversarial text classification work with two transformer-based models, and studied their obfuscation success in such a setting. We showed them to reliably drop target model performance below chance, though human detectability of the attacks remained above chance. Future work could focus on further minimizing this detection under our realistic constraints.

Acknowledgments

Our research strongly relied on openly available resources. We thank all whose work we could use. We would also like to thank the anonymous reviewers, Bertrand Higy, Bram Willemsen, and Chris van der Lee for their valuable feedback. Ákos Kádár contributed to this research independently. This work does not reflect Borealis AI's views nor any information Ákos Kádár may have learned while employed by Borealis AI.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016a. Tensorflow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, pages 265–283. USENIX Association.

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016b. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318.

Carlisle Adams. 2006. A classification for privacy techniques. *U. Ottawa L. & Tech. J.*, 3:35.

Jalal S Alowibdi, Ugo A Buy, and Philip Yu. 2013. Empirical evaluation of profile characteristics for gender classification on twitter. In *Machine Learning and Applications (ICMLA), 2013 12th International Conference on*, volume 1, pages 365–369. IEEE.

Shlomo Argamon, Sushant Dhawle, Moshe Koppel, and James W Pennebaker. 2005. Lexical predictors of personality type. In *Proceedings of the 2005 Joint Annual Meeting of the Interface and the Classification Society of North America*, pages 1–16.

Harald Baayen, Hans van Halteren, Anneke Neijt, and Fiona Tweedie. 2002. An experiment in authorship attribution. In *6th JADT*, volume 1, pages 69–75.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. 2018. Simply the best: minimalist system trumps complex models in author profiling. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 143–156. Springer.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Charley Beller, Rebecca Knowles, Craig Harman, Shane Bergsma, Margaret Mitchell, and Benjamin Van Durme. 2014. I'ma belieber: Social roles via self-identification and conceptual attributes. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 181–186.

Janek Bevendorff, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Heuristic authorship obfuscation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1108.

Janek Bevendorff, Tobias Wenzel, Martin Potthast, Matthias Hagen, and Benno Stein. 2020. On divergence-based author obfuscation: An attack on the state of the art in statistical authorship verification. *it-Information Technology*, 62(2):99–115.

Haohan Bo, Steven HH Ding, Benjamin Fung, and Farkhund Iqbal. 2019. Er-ae: differentially-private text generation for authorship anonymization. *arXiv preprint arXiv:1907.08736*.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):1–22.

John D Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating gender on twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1301–1309. Association for Computational Linguistics.

Aylin Caliskan, Fabian Yamaguchi, Edwin Dauber, Richard E. Harang, Konrad Rieck, Rachel Greenstadt, and Arvind Narayanan. 2018. When coding style survives compilation: De-anonymizing programmers from executable binaries. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, and Cho-Jui Hsieh. 2020. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3601–3608. AAAI Press.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Walter Daelemans. 2013. Explanation in computational stylometry. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 451–462. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 653–663.

Harrison Edwards and Amos J. Storkey. 2016. Censoring representations with an adversary. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding nlp systems. In *Proceedings of NAACL-HLT*, pages 1634–1647.

Jacob Eisenstein, Noah A Smith, and Eric P Xing. 2011. Discovering sociolinguistic associations with structured sparsity. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1365–1374. Association for Computational Linguistics.

Chris Emmery, Grzegorz Chrupała, and Walter Daelemans. 2017. Simple queries as distant labels for predicting gender on twitter. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 50–55.

Chris Emmery, Enrique Manjavacas, and Grzegorz Chrupała. 2018. Style obfuscation by invariance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 984–996.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Natasha Fernandes, Mark Dras, and Annabelle McIver. 2019. Generalised differential privacy for text document processing. In *International Conference on Principles of Security and Trust*, pages 123–148. Springer, Cham.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

David I Holmes. 1998. The evolution of stylometry in humanities scholarship. *Literary and linguistic computing*, 13(3):111–117.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7.

Xiaolei Huang, Xing Linzi, Franck Dernoncourt, and Michael J. Paul. 2020. Multilingual twitter corpus and baselines for evaluating demographic bias in hate speech recognition. In *Proceedings of the Twelveth International Conference on Language Resources and Evaluation (LREC 2020)*, Marseille, France. European Language Resources Association (ELRA).

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Youngjun Joo, Inchon Hwang, L Cappellato, N Ferro, D Losada, and H Müller. 2019. Author profiling on social media: An ensemble learning model using various features. *Notebook for PAN at CLEF*.

Patrick Juola and Darren Vescovi. 2011. Analyzing stylometric approaches to author obfuscation. In *IFIP International Conference on Digital Forensics*, pages 115–125. Springer.

Jad Kabbara and Jackie Chi Kit Cheung. 2016. Stylistic transfer in natural language generation systems using recurrent neural networks. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pages 43–47.

Gary Kacmarcik and Michael Gamon. 2006. Obfuscating document stylometry to preserve author anonymity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 444–451. Association for Computational Linguistics.

Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780.

Georgi Karadzhov, Tsvetomila Mihaylova, Yasen Kiprov, Georgi Georgiev, Ivan Koychev, and Preslav Nakov. 2017. The case for being average: A mediocrity approach to style masking and author obfuscation. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 173–185. Springer.

Moshe Koppel, Shlomo Argamon, and Anat Rachel Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.

Moshe Koppel and Jonathan Schler. 2004. Authorship verification as a one-class classification problem. In *Proceedings of the twenty-first international conference on Machine learning*, page 62.

Alon Lavie and Michael J Denkowski. 2009. The meteor metric for automatic evaluation of machine translation. *Machine translation*, 23(2-3):105–115.

Hoi Le, Reihaneh Safavi-Naini, and Asadullah Galib. 2015. Secure obfuscation of authoring style. In *IFIP International Conference on Information Security Theory and Practice*, pages 88–103. Springer.

Chris van der Lee, Albert Gatt, Emiel Van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30, Melbourne, Australia. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4208–4215.

Asad Mahmood, Zubair Shafiq, and Padmini Srinivasan. 2020. A girl has A name: Detecting authorship obfuscation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2235–2245. Association for Computational Linguistics.

Muharram Mansoorizadeh, Taher Rahgooy, Mohammad Aminiyan, and Mahdy Eskandari. 2016. Author obfuscation using wordnet and language models—notebook for pan at clef 2016. In *CLEF 2016 Evaluation Labs and Workshop–Working Notes Papers*, pages 5–8.

Robert AJ Matthews and Thomas VN Merriam. 1993. Neural computation in stylometry i: An application to the works of shakespeare and fletcher. *Literary and Linguistic computing*, 8(4):203–209.

Thomas VN Merriam and Robert AJ Matthews. 1994. Neural computation in stylometry ii: An application to the works of shakespeare and marlowe. *Literary and Linguistic Computing*, 9(1):1–6.

Frederick Mosteller and David L Wallace. 1963. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed federalist papers. *Journal of the American Statistical Association*, 58(302):275–309.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148.

Tempestt Neal, Kalaivani Sundararajan, Aneez Fatima, Yiming Yan, Yingfei Xiang, and Damon Woodard. 2017. Surveying stylometry techniques and applications. *ACM Computing Surveys (CSUR)*, 50(6):1–36.

Jekaterina Novikova, Ondrej Dusek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. In *2017 Conference on Empirical Methods in Natural Language Processing*, pages 2231–2242. Association for Computational Linguistics.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Barbara Plank and Dirk Hovy. 2015. Personality traits on twitter—or—how to get 1,500 personality tests in a week. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 92–98.

Martin Potthast, Matthias Hagen, and Benno Stein. 2016. Author obfuscation: Attacking the state of the art in authorship verification. In *CLEF (Working Notes)*, pages 716–749.

Francisco Rangel, Paolo Rosso, Ben Verhoeven, Walter Daelemans, Martin Potthast, and Benno Stein. 2016. Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. *Working Notes Papers of the CLEF*.

Josyula R Rao, Pankaj Rohatgi, et al. 2000. Can pseudonymity really guarantee privacy? In *USENIX Security Symposium*, pages 85–96.

K Rayner, SJ White, RL Johnson, and SP Liversedge. 2006. Raeding wrods with jubmled lettres: there is a cost. *Psychological science*, 17(3):192.

Sravana Reddy and Kevin Knight. 2016. Obfuscating gender in social media writing. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 17–26.

Chakaveh Saedi and Mark Dras. 2020. Large scale author obfuscation using Siamese variational auto-encoder: The SiamAO system. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 179–189, Barcelona, Spain (Online). Association for Computational Linguistics.

Maarten Sap, Gregory Park, Johannes Eichstaedt, Margaret Kern, David Stillwell, Michal Kosinski, Lyle Ungar, and H Andrew Schwartz. 2014. Developing age and gender predictive lexica over social media. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1146–1151.

Rakshith Shetty, Bernt Schiele, and Mario Fritz. 2018. A4nt: author attribute anonymity by adversarial training of neural machine translation. In *Proceedings of the 27th USENIX Conference on Security Symposium*, pages 1633–1650.

Noah A Smith. 2012. Adversarial evaluation for models of natural language. *arXiv preprint arXiv:1207.0245*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt. 2013. Classify, but verify: Breaking the closed-world assumption in stylometric authorship attribution. In *IFIP Working Group*, volume 11, page 64.

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

Svitlana Volkova and Yoram Bachrach. 2016. Inferring perceived demographics from user emotional tone and user-environment emotional contrast. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL*.

Svitlana Volkova, Yoram Bachrach, Michael Armstrong, and Vijay Sharma. 2015. Inferring latent user properties from texts published in social media. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Svitlana Volkova, Glen Coppersmith, and Benjamin Van Durme. 2014. Inferring user political preferences from streaming communications. In *ACL (1)*, pages 186–196.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Qiongkai Xu, Chenchen Xu, and Lizhen Qu. 2019. ALTER: Auxiliary text rewriting tool for natural language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 13–18, Hong Kong, China. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020b. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. Bert-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373.

Cross-Cultural Similarity Features for Cross-Lingual Transfer Learning of Pragmatically Motivated Tasks

Jimin Sun[1*] **Hwijeen Ahn**[2*] **Chan Young Park**[3*]
Yulia Tsvetkov[3] **David R. Mortensen**[3]
[1]Seoul National University, Republic of Korea
[2]Sogang University, Republic of Korea
[3]Language Technologies Institute, Carnegie Mellon University, USA
jiminsun@dm.snu.ac.kr, hwijeen@sogang.ac.kr,
{chanyoun, ytsvetko, dmortens}@cs.cmu.edu

Abstract

Much work in cross-lingual transfer learning explored how to select better transfer languages for multilingual tasks, primarily focusing on typological and genealogical similarities between languages. We hypothesize that these measures of linguistic proximity are not enough when working with pragmatically-motivated tasks, such as sentiment analysis. As an alternative, we introduce three linguistic features that capture cross-cultural similarities that manifest in linguistic patterns and quantify distinct aspects of language pragmatics: language context-level, figurative language, and the lexification of emotion concepts. Our analyses show that the proposed pragmatic features do capture cross-cultural similarities and align well with existing work in sociolinguistics and linguistic anthropology. We further corroborate the effectiveness of pragmatically-driven transfer in the downstream task of choosing transfer languages for cross-lingual sentiment analysis.

1 Introduction

Hofstede et al. (2005) defined culture as the collective mind which "distinguishes the members of one group of people from another." Cultural idiosyncrasies affect and shape people's beliefs and behaviors. Linguists have particularly focused on the relationship between culture and language, revealing in qualitative case studies how cultural differences are manifested as linguistic variations (Siegel, 1977).

Quantifying cross-cultural similarities from linguistic patterns has largely been unexplored in NLP, with the exception of studies that focused on cross-cultural differences in word usage (Garimella et al., 2016; Lin et al., 2018). In this work, we aim to quantify cross-cultural similarity, focusing on *semantic* and *pragmatic* differences across languages.[1] We devise a new distance measure between languages based on linguistic proxies of culture. We hypothesize that it can be used to select transfer languages and improve cross-lingual transfer learning, specifically in pragmatically-motivated tasks such as sentiment analysis, since expressions of subtle sentiment or emotion—such as subjective well-being (Smith et al., 2016), anger (Oster, 2019), or irony (Karoui et al., 2017)—have been shown to vary significantly by culture.

We focus on three distinct aspects in the intersection of language and culture, and propose features to operationalize them. First, every language and culture rely on different levels of *context in communication*. Western European languages are generally considered low-context languages, whereas Korean and Japanese are considered high-context languages (Hall, 1989). Second, similar cultures construct and construe *figurative language* similarly (Casas and Campoy, 1995; Vulanović, 2014). Finally, *emotion semantics* is similar between languages that are culturally-related (Jackson et al., 2019). For example, in Persian, 'grief' and 'regret' are expressed with the same word whereas 'grief' is co-lexified with 'anxiety' in Dargwa. Therefore, Persian speakers may perceive 'grief' as more similar to 'regret,' while Dargwa speakers may associate the concept with 'anxiety.'

We validate the proposed features qualitatively, and also quantitatively by an extrinsic evaluation method. We first analyze each linguistic feature

---

*The first three authors contributed equally.

[1] In linguistics, *pragmatics* has both a broad and a narrow sense. Narrowly, the term refers to formal pragmatics. In the broad sense, which we employ in this paper, pragmatics refers to contextual factors in language use. We are particularly concerned with cross-cultural pragmatics and finding quantifiable linguistic measures that correspond to aspects of cultural context. These measures are not the cultural characteristics that would be identified by anthropological linguists themselves but are rather intended to be measurable correlates of these characteristics.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2403–2414
April 19 - 23, 2021.

to confirm that they capture the intended cultural patterns. We find that the results corroborate the existing work in sociolinguistics and linguistic anthropology. Next, as a practical application of our features, we use them to rank transfer languages for cross-lingual transfer learning. Lin et al. (2019) have shown that selecting the right set of transfer languages with syntactic and semantic language-level features can significantly boost the performance of cross-lingual models. We incorporate our features into Lin et al. (2019)'s ranking model to evaluate the new cultural features' utility in selecting better transfer languages. Experimental results show that incorporating the features improves the performance for cross-lingual sentiment analysis, but not for dependency parsing. These results support our hypothesis that cultural features are more helpful when the cross-lingual task is driven by pragmatic knowledge. [2]

2 Pragmatically-motivated Features

We propose three language-level features that quantify the cultural similarities across languages.

Language Context-level Ratio A language's *context-level* reflects the extent to which the language leaves the identity of entities and predicates to context. For example, an English sentence *Did you eat lunch?* explicitly indicates the pronoun *you*, whereas the equivalent Korean sentence 점심 먹었니? (= *Did eat lunch?*) omits the pronoun. Context-level is considered one of the distinctive attributes of a language's pragmatics in linguistics and communication studies, and if two languages have similar levels of context, their speakers are more likely to be from similar cultures (Nada et al., 2001).

The language context-level ratio (`LCR`) feature approximates this linguistic quality. We compute the pronoun- and verb-token ratio, $\texttt{ptr}(l_k)$ and $\texttt{vtr}(l_k)$ for each language l_k, using part-of-speech tagging results. We first run language-specific POS-taggers over a large mono-lingual corpus for each language. Next, we compute `ptr` as the ratio of count of pronouns in the corpus to the count of all tokens. `vtr` is obtained likewise with verb tokens. Low `ptr`, `vtr` values may indicate that a language leaves the identity of entities and predicates, respectively, to context. We then compare these values between the *target language* l_{tg} and *transfer language* l_{tf}, which leads to the following definition of `LCR`:

$$\texttt{LCR-pron}(l_{tf}, l_{tg}) = \texttt{ptr}(l_{tg})/\texttt{ptr}(l_{tf})$$
$$\texttt{LCR-verb}(l_{tf}, l_{tg}) = \texttt{vtr}(l_{tg})/\texttt{vtr}(l_{tf})$$

Literal Translation Quality Similar cultures tend to share similar figurative expressions, including idiomatic multiword expressions (MWEs) and metaphors (Kövecses, 2003, 2010). For example, *like father like son* in English can be translated word-by-word into a similar idiom *tel père tel fils* in French. However, in Japanese, a similar idiom 蛙の子は蛙 (*Kaeru no ko wa kaeru*) "A frog's child is a frog." cannot be literally translated.

Literal translation quality (`LTQ`) feature quantifies how well a given language pair's MWEs are preserved in literal (word-by-word) translation, using a bilingual dictionary. A well-curated list of MWEs is not available for the majority of languages. We thus follow an automatic extraction approach of MWEs (Tsvetkov and Wintner, 2010). First, a variant of pointwise mutual information, PMI[3] (Daille, 1994) is used to extract noisy lists of top-scoring n-grams from two large monolingual corpora from different domains, and intersecting the lists filters out domain-specific n-grams and retains the language-specific top-k MWEs. Then, a bilingual dictionary between l_{tf} and l_{tg} and a parallel corpus between the pair are used. [3] For each n-gram in l_{tg}'s MWEs, we search for its literal translations extracted using the dictionary in parallel sentences containing the n-gram. For any word in the n-gram, if there is a translation in the parallel sentence, we consider this as hit, otherwise as miss. And we calculate *hit ratio* as $\frac{hit}{(hit+miss)}$ for each n-gram found in the parallel corpus. Finally, we average the hit ratios of all n-grams and z-normalize over the transfer languages to obtain $\texttt{LTQ}(l_{tf}, l_{tg})$.

Emotion Semantics Distance Emotion semantic distance (`ESD`) measures how similarly emotions are lexicalized across languages. This is inspired by Jackson et al. (2019) who used colexification patterns (i.e., when different concepts are expressed using the same lexical item) to capture the semantic similarity of languages. However, colexification patterns require human annotation,

---

[2] Code and data are publicly available at https://github.com/hwijeen/langrank.

[3] While dictionaries and parallel corpora are not available for many languages, they are easier to obtain than the task-specific annotations of MWEs.

and existing annotations may not be comprehensive. We extend Jackson et al. (2019)'s method by using cross-lingual word embeddings.

We define `ESD` as the average distance of emotion word vectors in transfer and target languages, after aligning word embeddings into the same space. More specifically, we use 24 emotion concepts defined in Jackson et al. (2019) and use bilingual dictionaries to expand each concept into every other language (e.g., *love* and *proud* to *Liebe* and *stolz* in German). We then remove the emotion word pairs from the bilingual dictionaries, and use the remaining pairs to align word embeddings of source into the space of target languages. We hypothesize that if words correspond to the same emotion concept in different languages (e.g., *proud* and *stolz*) have similar meaning, they should be aligned to the same point despite the lack of supervision. However, because each language possesses different emotion semantics, emotions are scattered into different positions. We thus define `ESD` as the average cosine distance between languages:

$$\texttt{ESD}(l_{tf}, l_{tg}) = \sum_{e \in E} \cos(\mathbf{v}_{tf,e}, \mathbf{v}_{tg,e})/|E|$$

where E is the set of emotion concepts and $\mathbf{v}_{tf,e}$ is the aligned emotion word vector of language l_{tf}.

3 Feature Analysis

In this section, we evaluate the proposed pragmatically-motivated features intrinsically. Throughout the analyses, we use 16 languages listed in Figure 4 which are later used for extrinsic evaluation (§5).

3.1 Implementation Details

We used multilingual word tokenizers from NLTK and RDR POS Tagger (Nguyen et al., 2014) for most of the languages except for Arabic, Chinese, Japanese, and Korean, where we used PyArabic, Jieba, Kytea, and Mecab, respectively. For monolingual corpora, we used the news-crawl 1M corpora from Leipzig (Goldhahn et al., 2012) for both `LCR` and `LTQ`. We used bilingual dictionaries from Choe et al. (2020) and TED talks corpora (Qi et al., 2018) for both parallel corpora and an additional monolingual corpus for `LTQ`. We focused on bigrams and trigrams and set k, the number of extracted MWEs, to 500. We followed Lample et al. (2018) to generate the supervised cross-lingual word embeddings for `ESD`.

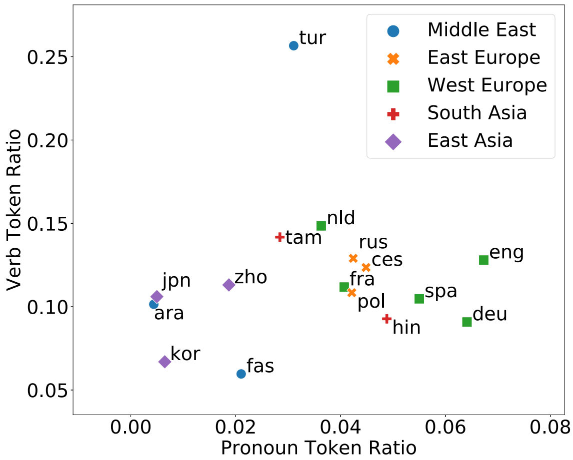


Figure 1: Plot of languages in `ptr` and `vtr` plane. Languages are color-coded according to the cultural areas defined in Siegel (1977).

3.2 LCR and Language Context-level

`ptr` approximates how often discourse entities are indexed with pronouns rather than left conjecturable from context. Similarly, `vtr` estimates the rate at which predicates appear explicitly as verbs. In order to examine to which extent these features reflect context-levels, we plot languages on a two-dimensional plane where the x-axis indicates `ptr` and the y-axis indicates `vtr` in Figure 1.

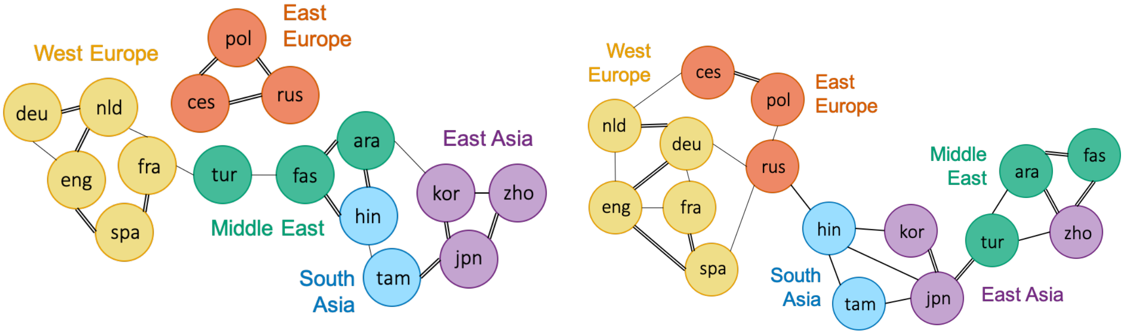
The plot reveals a clear pattern of context-levels in different languages. Low-context languages such as German and English (Hall, 1989) possess the largest values of `ptr`. On the other extreme are located Korean and Japanese with low `ptr`, which are representative of high-context languages. One thing to notice is the isolated location of Turkish with a high `vtr`. This is morphosyntactically plausible as a lot of information is expressed by the affixation to verbs in Turkish.

3.3 LTQ and MWEs

`LTQ` uses n-grams with high PMI scores as proxies for figurative language MWE (PMI MWEs). We evaluate the quality of selected MWEs and the resulting `LTQ` by comparing with human-curated list of figurative language MWE (gold MWEs) that are available in some languages. We collected gold MWEs in multiple languages from Wiktionary[4]. We discarded languages with less than 2,000 phrases on the list, resulting in four languages (English, French, German, Spanish) for

---

[4] For example, https://en.wiktionary.org/wiki/Category:English_idioms



(a) Network based on Emotion Semantics Distance.

(b) Network based on syntactic distance.

Figure 2: Network of languages color-coded by their cultural areas. An edge is added between the two languages if a language is ranked in the top-2 closest languages of the other language in terms of feature value.

analysis.

First, we check how many PMI MWEs are actually in the gold MWEs. Out of the top-500 PMI bigrams and trigrams, 19.0% of bigrams and 3.8% of trigrams are included in the gold MWE list (averaged over four languages). For example, the trigrams in the PMI MWEs, *keep an eye* and *take into account*, are considered to be in the gold MWEs as *keep an eye peeled* and *take into account* are in the list. The seemingly low percentages are reasonable, regarding that the PMI scores are designed to extract collocations patterns rather than figurative languages themselves.

Secondly, to validate using PMI MWEs as proxies, we compare the `LTQ` of PMI MWEs with the `LTQ` using gold MWEs. Specifically, we obtained the `LTQ` scores of each language pair with target languages limited to the four European languages mentioned above. Then for each target language, we measured Pearson correlation coefficient between the two `LTQ` scores based on the two MWE lists. The average coefficient was 0.92, which indicates a strong correlation between the two resulting `LTQ` scores, and thus justifies using PMI MWEs for all other languages.

3.4 ESD and Cultural Grouping

We investigate what is carried by `ESD` by visualizing and looking at the nearest neighbors of emotion vectors.[5] Jackson et al. (2019) used word colexification patterns to reveal that the same emotion concepts cluster with different emotions according to the language family they belong to. For instance, in Tai-Kadai languages, *hope* appears in the same cluster as *want* and *pity*, while *hope* associates with *good* and *love* in the Nakh-Daghestanian language family. Our results derived from `ESD` do not rely on colexification patterns, but also support this finding. The nearest neighbors of the Chinese word for *hope* was *want* and *pity*, while they were found as *love* and *joy* for *hope* in Arabic.

In Figure 2, we compare `ESD` to the syntactic distance between languages by constructing two networks of languages based on each feature. Figure 2a uses `ESD` as reference while Figure 2b uses the syntactic distance from the URIEL database (Littell et al., 2017). Each node represents a language, color-coded by its cultural area. For each language, we sort the other languages according to the distance value. When a language is in the list of top-k closest languages, we draw an edge between the two. We set $k = 2$.

We see that languages in the same cultural areas tend to form more cohesive clusters in Figure 2a compared to Figure 2b. The portion of edges *within* the cultural areas is 76% for `ESD` while it is 59% for syntactic distance. These results indicate that `ESD` effectively extracts linguistic information that aligns well with the commonly shared perception of cultural areas.

3.5 Correlation with Geographical Distance

Regarding the language clusters in Figure 2a, some may suspect that geographic distance can substitute the pragmatically-inspired features. For Chinese, Korean and Japanese are the closest languages by `ESD`, which can also be explained by their geographical proximity. Do our features add additional pragmatic information, or can they simply be replaced by geographical distance?

To verify this speculation, we evaluate Pearson's correlation coefficient of each pragmatic feature

---

[5] A visualization demo of emotion vectors can be found at `https://bit.ly/emotion_vecs`.

value with geographical distance from URIEL. The feature with the strongest correlation was `ESD` (r=0.4). The least correlated was `LCR-verb` (r=0.03), followed by `LCR-pron` (r=0.17) and `LTQ` (r=−0.31)[6]. The results suggest that the pragmatic features contain extra information that cannot be subsumed by geographic distance.

4 Extrinsic Evaluation: Ranking Transfer Languages

To demonstrate the utility of our features, we apply them to a *transfer language ranking* task for cross-lingual transfer learning. We first present the overall task setting, including the datasets and models used for the two cross-lingual tasks. Next, we describe the transfer language ranking model and its evaluation metrics.

4.1 Task Setting

We define our task as the *language ranking* problem: given the target language l_{tg}, we want to rank a set of n candidate transfer languages $\mathcal{L}_{tf}=\{l_{tf}^{(1)},\ldots,l_{tf}^{(n)}\}$ by their usefulness when transferred to l_{tg}, which we refer to as *transferability* (illustrated in Figure 3). The effectiveness of cross-lingual transfer is often measured by evaluating the joint training or zero-shot transfer performance (Wu and Dredze, 2019; Schuster et al., 2019). In this work, we quantify the effectiveness as the zero-shot transfer performance, following Lin et al. (2019). Our goal is to train a model that ranks available transfer languages in $\mathcal{L}_{tf}$ by their transferability for a target language l_{tg}.

To train the ranking model, we first need to find the ground-truth transferability rankings, which operate as the model's training data. We evaluate the zero-shot performance $z_{tf,tg}$ by training a task-specific cross-lingual model solely with transfer language l_{tf} and testing on l_{tg}. After evaluating $z_{tf,tg}$ for each candidate transfer language in $\mathcal{L}_{tf}$, we obtain the optimal ranking of languages $\boldsymbol{r}_{tg}$ by sorting languages according to the measured $z_{tf,tg}$. Note that $\boldsymbol{r}_{tg}$ also depends on downstream task.

Next, we train the language ranking model. The ranking model predicts the transfer ranking of candidate languages. Each source, target pair (l_{tf}, l_{tg}) is represented as a vector of language features $\boldsymbol{f}_{tf,tg}$, which may include phonological similarity, typological similarity, word-overlap to name a few. The ranking model takes $\boldsymbol{f}_{tf,tg}$ of every l_{tf} as input, and predicts the transferability ranking $\hat{\boldsymbol{r}}_{tg}$. Using $\boldsymbol{r}_{tg}$ from the previous step as training data, the model learns to find optimal transfer languages based on $\boldsymbol{f}_{tf,tg}$. The trained model can either be used to select the optimal set of transfer languages, or to decide which language to additionally annotate during the data creation process.

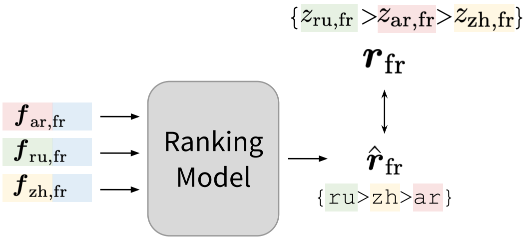


Figure 3: Illustration of transfer language ranking problem when the target language is French (fr) and there are three available transfer languages: Arabic (ar), Russian (ru), and Chinese (zh). The output ranking $\hat{\boldsymbol{r}}_{\text{fr}}$ is compared to the ground truth ranking $\boldsymbol{r}_{\text{fr}}$ which is determined by the zero-shot performance $\boldsymbol{z}$ of cross-lingual models.

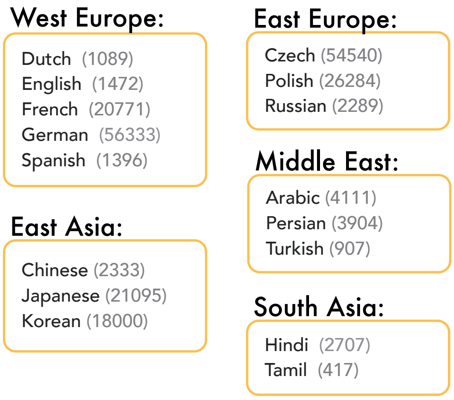


Figure 4: Languages used throughout the experiments are grouped by their cultural areas (Siegel, 1977). The numbers indicate the size of each dataset.

4.2 Task & Dataset

We apply the proposed features to train a ranking model for two distinctive tasks: multilingual sentiment analysis (SA) and multilingual dependency parsing (DEP). The tasks are chosen based on our hypothesis that high-order information such as pragmatics would assist sentiment analysis while it may be less significant for dependency parsing, where lower-order information such as syntax is relatively stressed.

SA As there is no single sentiment analysis dataset covering a wide variety of languages, we

---

[6] When two languages are more similar, LTQ is higher whereas geographic distance is smaller.

collected various review datasets from different sources.[7] All samples are labeled as either positive or negative. In case of datasets rated with a five-point Likert scale, we mapped 1–2 to negative and 4–5 to positive. We settled on a dataset consist of 16 languages categorized into five distinct cultural groups: West Europe, East Europe, East Asia, South Asia, and Middle East (Figure 4).

DEP To compare the effectiveness of the proposed features on syntax-focused tasks, we chose datasets of the same set of 16 languages from Universal Dependencies v2.2 (Nivre et al., 2018).

4.3 Task-Specific Cross-Lingual Models

SA Multilingual BERT (mBERT) (Devlin et al., 2019), a multilingual extension of BERT pretrained with 104 different languages, has shown strong results in various text classification tasks in cross-lingual settings (Sun et al., 2019; Xu et al., 2019; Li et al., 2019). We use mBERT to conduct zero-shot cross-lingual transfer and to extract optimal transfer language rankings: fine-tune mBERT on transfer language data and test it on target language data. The performance is measured by the macro F1 score on the test set.

DEP We adopt the setting from Ahmad et al. (2018) to perform cross-lingual zero-shot transfer. We train deep biaffine attentional graph-based models (Dozat and Manning, 2016) which achieved state-of-the-art performance in dependency parsing for many languages. The performance is evaluated using labeled attachment scores (LAS).

4.4 Ranking Model & Evaluation

Ranking Model For the language ranking model, we employ gradient boosted decision trees, LightGBM (Ke et al., 2017), which is one of the state-of-the-art models for ranking tasks.[8]

Ranking Evaluation Metric We evaluate the ranking models' performance with two standard metrics for ranking tasks: Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain at position p (NDCG@p) (Järvelin and Kekäläinen, 2002). While MAP assumes a binary concept of relevance, NDCG is a more fine-grained measure that reflects the ranking positions. The *relevant* languages for computing MAP are defined as the top-k languages in terms of zero-shot performance in the downstream task. In our experiments, we set k to 3 for MAP. Similarly, we use NDCG@3.

We train and evaluate the model using leave-one-out cross-validation: where one language is set aside as the test language while other languages are used to train the ranking model. Among the training languages, each language is posited in turn as the *target* language while others are the *transfer* languages.

5 Experiments

5.1 Baselines

LANGRANK LANGRANK (Lin et al., 2019) uses 13 features to train the ranking model: The dataset size in transfer language (`tf_size`), target language (`tg_size`), and the ratio between the two (`ratio_size`); Type-token-ratio (`ttr`) which measures lexical diversity and `word_overlap` for lexical similarity between a pair of languages; various distances between a language pair from the URIEL database (geographic `geo`, genetic `gen`, inventory `inv`, syntactic `syn`, phonological `phon` and featural `feat`).

MTVEC Malaviya et al. (2017) proposed to learn a language representation while training a neural machine translation (NMT) system in a similiar fashion to Johnson et al. (2017). During training, a language token is prepended to the source sentence and the learned token's embedding becomes the language vector. Bjerva et al. (2019) has shown that such language representations contain various types of linguistic information ranging from word order to typological information. We used the one released by Malaviya et al. (2017) which has the dimension of 512.

5.2 Individual Feature Contribution

We first look into whether the proposed features are helpful in ranking transfer languages for sentiment analysis and dependency parsing (Table 1). We add all three features (PRAG) to the two baseline features (LANGRANK, MTVEC) and compare the performance in the two tasks. Results show that our features improve both baselines in SA, implying that the pragmatic information captured by our features is helpful for discerning the subtle differences in sentiment among languages.

In the case of DEP, including our features brings inconsistent results to performance. The features

---

[7] Details are provided in Appendix A. Note that the difference in domain and label distribution of data can also affect the transferability, and a related discussion is in §5.4

[8] More details on the cross-lingual models, ranking model, and their training can be found in Appendix B.

	SA		DEP	
	MAP	NDCG	MAP	NDCG
LangRank	71.3	86.5	**63.0**	**82.2**
LangRank+Prag	**76.0**	**90.9**	61.7	80.5
- `LCR`	75.0	88.3	60.3	79.6
- `LTQ`	72.4	89.3	63.1*	81.3*
- `ESD`	77.7*	92.1*	58.2	78.5
MTVec	71.1	89.5	43.0	69.7
MTVec+Prag	**74.3**	**90.8**	**49.7**	**74.8**
- `LCR`	72.9	90.1	54.1*	76.3*
- `LTQ`	71.2	89.0	53.0*	78.6*
- `ESD`	73.1	90.7	45.3	73.9

Table 1: Evaluation results of our features (Prag) added to each baseline. The higher scores are **boldfaced**. Rows in gray indicate ablation studies. * is marked when improvements are made compared to LangRank+Prag, MTVec+Prag respectively.

help the performance of MTVec while they deteriorate the performance of LangRank. Although some performance increase was observed when applied to MTVec, the performance of MTVec in DEP remains extremely poor. These conflicting trends suggest that pragmatic information is not crucial to less pragmatically-driven tasks, represented as dependency parsing in our case.

The low performance of MTVec in DEP is noticeable as MTVec is generally believed to contain a significant amount of syntactic information, with much higher dimensionality than LangRank. It also suggests the limitation of using distributional representations as language features; their lack of interpretability makes it difficult to control the kinds of information used in a model.

We additionally conduct ablation studies by removing each feature from the +Prag models to examine each feature's contribution. The SA results show that `LCR` and `LTQ` significantly contribute to overall improvements achieved by adding our features, while `ESD` turns out to be less helpful. Sometimes, removing `ESD` resulted in a better performance. In contrast, the results of DEP show that `ESD` consistently made a significant contribution, and `LCR` and `LTQ` were not useful. The results imply that the emotion semantics information of languages is surprisingly not useful in sentiment analysis, but more so in dependency parsing.

5.3 Group-wise Contribution

The previous experiment suggests that the same pragmatic information can be helpful to different extents depending on the downstream task. We further investigate to what extent each kind of information is useful to each task by conducting group-wise comparisons. To this end, we group the features into five categories: Pretrain-specific, Data-specific, Typology, Geography, Orthography, and Pragmatic. Pretrain-specific features cover factors that may be related to the performance of pretrained language models used in our task-specific cross-lingual models. Specifically, we used the size of the Wikipedia training corpus of each language used in training mBERT.[9] Note that we do not measure this feature group's performance on DEP as no pretrained language model was used in DEP. Data-specific features include `tf_size`, `tg_size`, and `ratio_size`. Typological features include `geo`, `syn`, `feat`, `phon`, and `inv` distances. Geography includes `geo` distance in isolation. Orthographic feature is the `word_overlap` between languages. Finally, the Pragmatic group consists of `ttr` and the three proposed features, `LCR`, `LTQ`, and `ESD`. `ttr` is included in Pragmatic as Richards (1987) have suggested that it encodes a significant amount of cultural information.

	SA		DEP	
	MAP	NDCG	MAP	NDCG
Pretrain-specific	39.0	55.5	-	-
Data-specific	68.0	85.4	37.2	55.0
Typology	44.9	60.7	**58.0**	**79.8**
Geography	24.9	55.0	32.3	65.1
Orthography	34.2	56.6	35.5	60.5
Pragmatic	**73.0**	**88.0**	46.5	71.8

Table 2: Ranking performance using each feature group as input to the ranking model.

Table 2 reports the performance of ranking models trained with the respective feature category. Interestingly, the two tasks showed significantly different results; the Pragmatic group showed the best performance in SA while the Typology group outperformed all other groups in DEP. This again confirms that the features indicating cross-lingual transferability differ depending on the target task. Although the Pretrain-specific features were more predictive than the Geography and Orthography features it was not as helpful as the Pragmatic features.

---

[9] `https://meta.wikimedia.org/wiki/List_of_Wikipedias`

5.4 Controlling for Dataset Size

The performance of cross-lingual transfer depends not only on the cultural similarity between transfer and target languages but also on other factors, including dataset size and label distributions. Although our model already accounts for the dataset size to some extent by including `tf_size` as input, we conduct a more rigorous experiment to better understand the importance of cultural similarity in language selection. Specifically, we control the data size by down-sampling all SA data to match both the size and label distribution of the second smallest Turkish dataset.[10] We then trained two ranking models equipped with different sets of features: LANGRANK and LANGRANK+PRAG.

In terms of languages, we focus on a setting where Turkish is the target and Arabic, Japanese and Korean are the transfer languages. This is a particularly interesting set of languages because the source languages are similar/dissimilar to Turkish in different aspects; Korean and Japanese are typologically similar to Turkish, yet in cultural terms, Arabic is more similar to Turkish.

In this controlled setting, the ground-truth ranking reveals that the optimal transfer language among the three is Arabic, followed by Korean and Japanese. It indicates the important role of cultural resemblance in sentiment analysis which encapsulates the rich historical relationship shared between Arabic- and Turkish-speaking communities. LANGRANK+PRAG chose Arabic as the best transfer language, suggesting that the imposed cultural similarity information from the features helped the ranking model learn the cultural tie between the two languages. On the other hand, LANGRANK ranked Japanese the highest over Arabic, possibly because the provided features mainly focus on typological similarity over cultural similarity.

6 Related Work

Quantifying Cross-cultural Similarity A few recent work in psycholinguistics and NLP have aimed to measure cultural differences, mainly from word-level semantics. Lin et al. (2018) suggested a cross-lingual word alignment method that preserves the cultural, social context of words. They derive cross-cultural similarity from the embeddings of a bilingual lexicon in the shared representation space. Thompson et al. (2018) computed similarity by comparing the nearest neighborhood of words in different languages, showing that words in some domains (e.g., time, quantity) exhibit higher cross-lingual alignment than other domains (e.g., politics, food, emotions). Jackson et al. (2019) represented each language as a network of emotion concepts derived from their colexification patterns and measured the similarity between networks.

Auxiliary Language Selection in Cross-lingual tasks There has been active work on leveraging multiple languages to improve cross-lingual systems (Neubig and Hu, 2018; Ammar et al., 2016). Adapting auxiliary language datasets to the target language task can be practiced through either language-selection or data-selection. Previous work on language-selection mostly relied on leveraging syntactic or semantic resemblance between languages (e.g. ngram overlap) to choose the best transfer languages (Zoph et al., 2016; Wang and Neubig, 2019). Our approach extends this line of work by leveraging cross-cultural pragmatics, an aspect that has been unexplored by prior work.

7 Future Directions

Typology of Cross-cultural Pragmatics The features proposed here provide three dimensions in a provisional quantitative cross-linguistic typology of pragmatics in language. Having been validated, both intrinsically and extrinsically, they can be used in studies as a stand-in for cross-cultural similarity. They also open a new avenue of research, raising questions about what other quantitative features of language are correlates of cultural and pragmatic difference.

Model Probing Fine-tuning pretrained models to downstream tasks has become the de facto standard in various NLP tasks, and their success has promoted the development of their multilingual extensions (Devlin et al., 2019; Lample and Conneau, 2019). While the performance gains from these models are undeniable, their learning dynamics remain obscure. This issue has prompted various probing methods designed to test what kind of linguistic information the models retain, including syntactic and semantic knowledge (Conneau et al., 2018; Liu et al., 2019; Ravishankar et al., 2019; Tenney et al., 2019). Similarly, our features can be employed as a touchstone to evaluate a model's knowledge in cross-cultural pragmatics. Investigating how different pretraining tasks affect the

---

[10] The size of the smallest language (Tamil; 417 samples) was too small to train an effective model.

learning of pragmatic knowledge will also be an interesting direction of research.

8 Conclusion

In this work, we propose three pragmatically-inspired features that capture cross-cultural similarities that arise as linguistic patterns: language context-level ratio, literal translation quality, and emotion semantic distance. Through feature analyses, we examine whether our features can operate as valid proxies of cross-cultural similarity. From a practical standpoint, the experimental results show that our features can help select the best transfer language for cross-lingual transfer in pragmatically-driven tasks, such as sentiment analysis.

Acknowledgements

The authors are grateful to the anonymous reviewers for their invaluable feedback. This material is based upon work supported by the National Science Foundation under Grant No. IIS2007960. We would also like to thank Amazon for providing AWS credits.

References

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard H. Hovy, Kai-Wei Chang, and Nanyun Peng. 2018. Near or far, wide range zero-shot cross-lingual dependency parsing. *CoRR*, abs/1811.00570.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Johannes Bjerva, Robert Östling, Maria Han Veiga, Jörg Tiedemann, and Isabelle Augenstein. 2019. What do language representations really represent? *Computational Linguistics*, 45(2):381–389.

Christopher J. Burges, Robert Ragno, and Quoc V. Le. 2007. Learning to rank with nonsmooth cost functions. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 193–200. MIT Press.

Rafael Monroy Casas and JM Hernández Campoy. 1995. A sociolinguistic approach to the study of idioms: Some anthropolinguistic sketches. *Cuadernos de Filología inglesa*, 4.

Yo Joong Choe, Kyubyong Park, and Dongwoo Kim. 2020. word2word: A collection of bilingual lexicons for 3,564 language pairs. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC 2020)*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Béatrice Daille. 1994. *Approche mixte pour l'extraction automatique de terminologie: statistiques lexicales et filtres linguistiques*. Ph.D. thesis, Ph. D. thesis, Université Paris 7.

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

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734.

Aparna Garimella, Rada Mihalcea, and James Pennebaker. 2016. Identifying cross-cultural differences in word usage. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 674–683, Osaka, Japan. The COLING 2016 Organizing Committee.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig Corpora Collection: From 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey. European Language Resources Association (ELRA).

Edward Twitchell Hall. 1989. *Beyond culture*. Anchor.

Geert H Hofstede, Gert Jan Hofstede, and Michael Minkov. 2005. *Cultures and organizations: Software of the mind*, volume 2. Mcgraw-hill New York.

Joshua Conrad Jackson, Joseph Watts, Teague R. Henry, Johann-Mattis List, Robert Forkel, Peter J. Mucha, Simon J. Greenhill, Russell D. Gray, and Kristen A. Lindquist. 2019. Emotion semantics show both cultural variation and universal structure. *Science*, 366(6472):1517–1522.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Jihen Karoui, Farah Benamara, Véronique Moriceau, Viviana Patti, Cristina Bosco, and Nathalie Aussenac-Gilles. 2017. Exploring the impact of pragmatic phenomena on irony detection in tweets: A multilingual corpus study. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 262–272, Valencia, Spain. Association for Computational Linguistics.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc.

Zoltán Kövecses. 2003. Language, figurative thought, and cross-cultural comparison. *Metaphor and Symbol*, 18(4):311–320.

Zoltán Kövecses. 2010. *Metaphor: A practical introduction*. Oxford University Press.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting BERT for end-to-end aspect-based sentiment analysis. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 34–41, Hong Kong, China. Association for Computational Linguistics.

Bill Yuchen Lin, Frank F. Xu, Kenny Zhu, and Seung-won Hwang. 2018. Mining cross-cultural differences and similarities in social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 709–719, Melbourne, Australia. Association for Computational Linguistics.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 8–14, Valencia, Spain. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Chaitanya Malaviya, Graham Neubig, and Patrick Littell. 2017. Learning language representations for typology prediction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2529–2535, Copenhagen, Denmark. Association for Computational Linguistics.

Korac-Kakabadse Nada, Kouzmin Alexander, Korac-Kakabadse Andrew, and Savery Lawson. 2001. Low-and high-context communication patterns: towards mapping cross-cultural encounters. *Cross Cultural Management: An International Journal*, 8(2):3–24.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 875–880, Brussels, Belgium. Association for Computational Linguistics.

Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham. 2014. RDRPOSTagger: A ripple down rules-based part-of-speech tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 17–20, Gothenburg, Sweden. Association for Computational Linguistics.

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. 2018. Universal dependencies 2.2. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Ulrike Oster. 2019. Cross-cultural semantic and pragmatlc profiling of emotion words. regulation and expression of anger in Spanish and German. *Current Approaches to Metaphor Analysis in Discourse*, 39:35.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.

Vinit Ravishankar, Memduh Gökırmak, Lilja Øvrelid, and Erik Velldal. 2019. Multilingual probing of deep pre-trained contextual encoders. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 37–47, Turku, Finland. Linköping University Electronic Press.

Brian Richards. 1987. Type/token ratios: What do they really tell us? *Journal of child language*, 14(2):201–209.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Bernard J. Siegel. 1977. Encyclopedia of anthropology. David E. Hunter and Phillip Whitten, eds. New York. *American Anthropologist*, 79(2):452–454.

Laura Smith, Salvatore Giorgi, Rishi Solanki, Johannes Eichstaedt, H Andrew Schwartz, Muhammad Abdul-Mageed, Anneke Buffone, and Lyle Ungar. 2016. Does 'well-being'translate on twitter? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2042–2047.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese Computational Linguistics*, pages 194–206, Cham. Springer International Publishing.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

B. Thompson, S. Roberts, and G. Lupyan. 2018. Quantifying semantic similarity across languages. *Proceedings of the 40th Annual Conference of the Cognitive Science Society (CogSci 2018)*.

Yulia Tsvetkov and Shuly Wintner. 2010. Extraction of multi-word expressions from small parallel corpora. In *Coling 2010: Posters*, pages 1256–1264, Beijing, China. Coling 2010 Organizing Committee.

Jelena Vulanović. 2014. Cultural markedness and strategies for translating idiomatic expressions in the epic poem "The Mountain Wreath" into English. *Mediterranean Journal of Social Sciences*, 5(13):210.

Xinyi Wang and Graham Neubig. 2019. Target conditioned sampling: Optimizing data selection for multilingual neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5823–5828, Florence, Italy. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

A Dataset for Sentiment Analysis

Dataset	Languages	Domain	Size	POS/NEG
SemEval-2016 Aspect Based Sentiment Analysis	Chinese	electronics	2333	1.53
	Arabic	hotel	4111	1.54
	English	restaurant	1472	2.14
	Dutch	restaurant	1089	1.43
	Spanish	restaurant	1396	2.82
	Russian	restaurant	2289	3.81
	Turkish	restaurant	907	1.32
SentiPers	Persian	product	3904	1.8
Amazon Customer Reviews	French	product	20771	8.0
	German	product	56333	6.56
	Japanese	product	21095	8.05
CSFD CZ	Czech	movie	54540	1.04
Naver Sentiment Movie Corpus	Korean	movie	18000	1.0
Tamil Movie Review Dataset	Tamil	movie	417	0.48
PolEval 2017	Polish	product	26284	1.38
Aspect based Sentiment Analysis	Hindi	product	2707	3.22

Table 3: Datasets for sentiment analysis.

B Task-Specific Models Details

SA Cross-lingual Model We performed supervised fine-tuning of multilingual BERT (mBERT) (Devlin et al., 2019) for the sentiment analysis task, as the model showed strong results in various text classification tasks in cross-lingual settings (Sun et al., 2019; Xu et al., 2019; Li et al., 2019). mBERT is pretrained with 104 different languages, including the 16 languages we used throughout our experiment. We used a concatenation of mean and max pooled representations from mBERT's penultimate layer, as it outperformed the standard practice of using the last layer's `[CLS]` token. The representation was passed to a fully connected layer for prediction. To extract optimal transfer rankings, we conducted zero-shot transfer with mBERT: fine-tuned mBERT on transfer language data and tested it on target language data.

Ranking Model We used LightGBM (Ke et al., 2017) with LambdaRank (Burges et al., 2007) algorithm. The model consists of 100 decision trees with 16 leaves each, and it was trained with the learning rate of 0.1. We optimized NDCG to train the model (Järvelin and Kekäläinen, 2002).

PHASE: Learning Emotional Phase-aware Representations for Suicide Ideation Detection on Social Media

Ramit Sawhney*
IIIT Delhi
`ramits@iiitd.ac.in`

Harshit Joshi*
University of Delhi
`harshit113@ducic.ac.in`

Lucie Flek
Mainz University of Applied Sciences
`lucie.flek@hs-mainz.de`

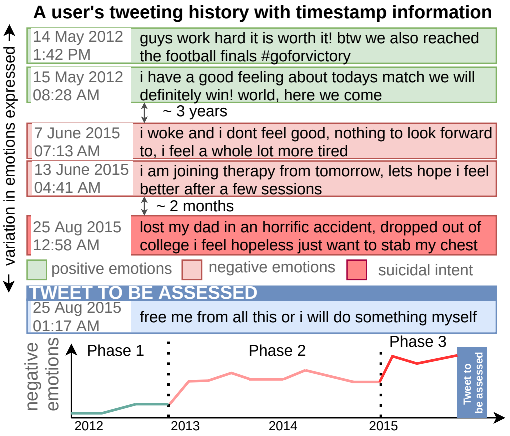
Rajiv Ratn Shah
IIIT Delhi
`rajivratn@iiitd.ac.in`

Abstract

Recent psychological studies indicate that individuals exhibiting suicidal ideation increasingly turn to social media rather than mental health practitioners. Contextualizing the build-up of such ideation is critical for the identification of users at risk. In this work, we focus on identifying suicidal intent in tweets by augmenting linguistic models with emotional phases modeled from users' historical context. We propose PHASE, a time-and phase-aware framework that adaptively learns features from a user's historical emotional spectrum on Twitter for preliminary screening of suicidal risk. Building on clinical studies, PHASE learns phase-like progressions in users' historical Plutchik-wheel-based emotions to contextualize suicidal intent. While outperforming state-of-the-art methods, we show the utility of temporal and phase-based emotional contextual cues for suicide ideation detection. We further discuss practical and ethical considerations.[1]

1 Introduction

Every 10.9 minutes, a person dies of suicide (Drapeau and McIntosh, 2020). Suicide ranks as the second leading cause of death for 14-35 year-olds (Hedegaard et al., 2020) in US. Extending appropriate clinical and psychological care to suicidal people relies on identifying those at risk. Unfortunately, 80% of patients do not undergo clinical treatment, and about 60% of those who died of suicide denied having any suicidal thoughts to mental health practitioners (McHugh et al., 2019; Franklin et al., 2017). In contrast, people exhibiting suicidal ideation often use social media to express their feelings (Coppersmith et al., 2014, 2016, 2018; Robinson et al., 2016; Reger et al., 2020), with eight out of ten people disclosing their suicidal thoughts and plans on social media (Golden et al., 2009).



14 May 2012 1:42 PM — guys work hard it is worth it! btw we also reached the football finals #goforvictory

15 May 2012 08:28 AM — i have a good feeling about todays match we will definitely win! world, here we come

~ 3 years

7 June 2015 07:13 AM — i woke and i dont feel good, nothing to look forward to, i feel a whole lot more tired

13 June 2015 04:41 AM — i am joining therapy from tomorrow, lets hope i feel better after a few sessions

~ 2 months

25 Aug 2015 12:58 AM — lost my dad in an horrific accident, dropped out of college i feel hopeless just want to stab my chest

TWEET TO BE ASSESSED

25 Aug 2015 01:17 AM — free me from all this or i will do something myself

Figure 1: We study a user's tweeting history and emotional progression. Note that while the user's most recent tweet (blue) shows a subtle indication of suicidal intent, it is not sufficient to ascertain suicide risk. Grouping the build-up of negative emotions (red) in the user's historical tweets into phase-like emotional progressions, by utilizing the elapsed time between tweets, can contextualize the user's state and provide a more accurate and interpretable risk assessment. *All examples in this paper have been anonymized and paraphrased as per a moderate disguise scheme (Bruckman, 2002) to protect user privacy (Chancellor et al., 2019b).*

Natural Language Processing (NLP) presents an encouraging prospect to complement social science to identify risk markers in user behavior (De Choudhury et al., 2013, 2016) to aid suicide risk assessment (Shing et al., 2018, 2020). However, suicide ideation is complex, and often, individual posts may not be sufficient to assess a user's suicide risk, even for humans (Sisask et al., 2008; O'dea et al., 2015). Figure 1 illustrates how features such as historical posts (Matero et al., 2019) can add context for analyzing a user's online behavior over time (Van Heeringen and Marušic, 2003) to better

---

* Equal contribution

[1] Code is available at `https://github.com/midas-research/phase-eacl/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2415–2428
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

ascertain suicide risk. Despite the success of user-centric contextual models (Flek, 2020) for suicide ideation detection, they have two major limitations.

First, recurrent neural networks, particularly LSTMs, that are natural methods to learn patterns from a sequence of a user's historical tweets (Cao et al., 2019; Zeng et al., 2019, 2020), assume uniform time gaps between successive tweets. However, tweets can be posted at irregular time intervals (Lei et al., 2018), and varying time gaps can influence the assessment of a user's suicidality progression (Chen et al., 2018), as shown in Figure 1.

Second, these methods implicitly assume that a user's mental and emotional state progression is smooth in time, with an ever-increasing tendency. However, in reality, studies show that emotional (Larsen et al., 2015), and suicidality progression can vary significantly (Bryan and Rudd, 2016; Bryan, 2020), and show fluctuating *phase* like patterns (Kiosses et al., 2014; Palmier-Claus et al., 2012). Analyzing such phase-wise emotion progressions and build-up, as illustrated in Figure 1, can be instrumental in contextualizing suicidal risk, and aiding clinical psychologists through increased interpretability in human-in-the-loop systems.[2]

Building on these limitations, and motivated by psychological studies (Neacsiu et al., 2018; Domínguez and Fernández, 2018) of emotional state progression, we propose **PHASE: PHase-Aware Suicidality identification Emotion progression model.** With PHASE, we present the first neural framework to identify suicide ideation on social media (§3.1) that explicitly models the *inherent phase-aware progressions* in users' emotional spectrums in a contextual time-aware manner.

We present the following key contributions:

(i) First, building on the success of large scale pretraining in NLP, we utilize Plutchik Transformer, a transformer to learn linguistic and Plutchik-based (Plutchik, 1980) emotional cues from tweets (§3.2).

(ii) We propose Time-Sensitive Emotion LSTM (TSE-LSTM) to learn the historical emotional progression of a user's mental states from their learned emotional spectrum in a time-aware manner (§3.3).

(iii) Based on psychological studies, we propose a novel method to learn users' emotional phase progressions by leveraging the amount of historical emotional context used to update the TSE-LSTM's cell state. PHASE identifies the onset of new emotional phases and learns a temporal *phase-aware emotional user representation* (§3.4) that is then used to identify suicide ideation in their recent tweets (§3.5), increasing the system transparency.

(iv) Through a series of experiments (§4.2), we show that PHASE significantly ($p < 0.005$) outperforms competitive methods, which do not take users' emotional phases into account (§5).

(v) We analyze the contributions of PHASE's individual components to suicide ideation detection (§5.2, §5.3, §5.4), assess its transparency and limitations through qualitative analysis (§5.5), and conclude by discussing the ethical implications and practical applicability of this study (§6).

2 Related Work

Traditional Methods: Researchers have devised various psychoclinical methods to assess suicidal risk (Pestian et al., 2016), such as the Suicide Probability Scale (Bagge and Osman, 1998), Suicide Ideation Questionnaire (wa Fu et al., 2007), Suicidal Affect-Behavior-Cognition Scale (Harris et al., 2015). While these methods are professional and effective, they require participants to answer questionnaires (Venek et al., 2017) or engage in interviews (Scherer et al., 2013), hence not reaching people who cannot access these resources or have a low motivation to seek professional help (Zachrisson et al., 2006; Essau, 2005). Harris and Goh (2016) show that such assessments can negatively impact people showing depressive symptoms.

NLP Methods: Recently, social media has shown promise in providing insights into users' mental states (Paul and Dredze, 2011). Jashinsky et al. (2014) reported that Twitter is a viable tool for real-time monitoring (Braithwaite et al., 2016) of suicide risk. Early efforts in utilizing social media leverage user features such as their age, gender,and social network connectivity (Masuda et al., 2013) and online suicide notes (Pestian et al., 2010). Since then, the focus has been on using psycholinguistic lexicons such as LIWC and textual features such as n-grams, POS tags, etc. for classification (De Choudhury et al., 2016; Sawhney et al., 2018b). Shared tasks such as CLPsych (Zirikly et al., 2019) and CLEF eRISK (Losada et al., 2020) have seen a rise in neural networks such as CNNs (Yates et al., 2017; Du et al., 2018; Naderi et al., 2019; Gaur et al., 2019) and LSTMs (Ji et al., 2018; Tadesse et al., 2020) to predict suicide risk. While these methods capture post semantics in isolation, no user context is leveraged, hindering insight into

---

[2] Similar to the post-screening on Facebook (Card, 2018).

the user's mental state to improve predictive power (Venek et al., 2017; Flek, 2020). User context includes the user's emotions (Ren et al., 2016; Guntuku et al., 2017), social networks (Mishra et al., 2019) and historical posts (Mathur et al., 2020).

Contextual Methods: The best performing model, the DualContextBERT (Matero et al., 2019), at CLPsych 2019 for suicidal estimation exemplifies the utility of temporal context. The DualContextBERT models post embeddings sequentially via an RNN. Such RNN-based approaches assume that users' historical posts are equally spaced in time, hindering their ability to learn their relative importance in a time-aware manner. Recently, time-aware modeling of well defined stages in numerical time series data shows promising results in clinical tasks like patient subtyping (Baytas et al., 2017) and disease progression (Gao et al., 2020). However, the time-sensitive phase extraction of user-generated posts on social media, and phase-aware modeling of textual data is underexplored and complex, as it involves noisy, unstructured and ambiguous inputs across irregular time intervals.

3 PHASE: Components and Learning

3.1 Notations and Problem Formulation

We formulate suicidal intent detection as a binary classification task to predict suicidal intent y_i for a tweet t_i, where, $y_i \in \{\text{suicidal intent present, suicidal intent absent}\}$. We denote the tweet to be assessed for the presence of suicidal intent as $t_i \in T = \{t_1, t_2, \cdots, t_N\}$, authored by a user $u_j \in U = \{u_1, u_2, \cdots, u_M\}$, posted at time τ^i_{curr}. Each tweet t_i is associated with history $H^j_i = [(h^i_1, \tau^i_1), (h^i_2, \tau^i_2), \cdots, (h^i_L, \tau^i_L)]$ where h^i_k is a historic tweet authored by user u_j posted at time τ^i_k with $\tau^i_1 < \tau^i_2 < \cdots < \tau^i_L < \tau^i_{curr}$.

As shown in Figure 2, PHASE first obtains a user's emotion spectrum from their historical tweets and the tweet to be assessed using a fine-tuned BERT model, Plutchik Transformer. We feed the historical tweet representations to our proposed Time-Sensitive Emotion LSTM to learn the temporal progression of a user's emotions. We then identify *phases* in a user's emotions from their learned historical emotional progression, and extract temporal features for a user from these phases using Phase-Adaptive convolutions. Finally, PHASE jointly learns the semantics of user tweets and their historical emotional context in a temporal phase-aware manner for suicide ideation detection in a tweet.

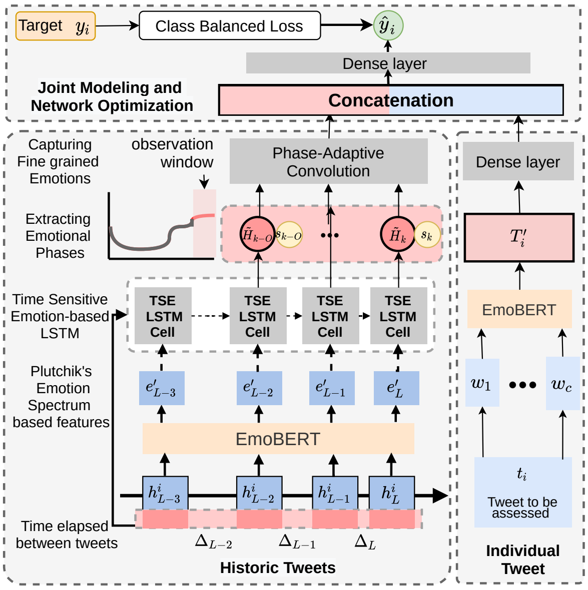


Figure 2: An illustration of PHASE's architecture.

3.2 Plutchik Transformer: Encoding Tweets

Studies show that emotions expressed in suicidal tweets are correlated with suicidal behavior (Sueki, 2015; Spates et al., 2018; Zhang et al., 2017). As a building block, we utilize Plutchik's wheel of emotions (Plutchik, 1980) to capture the emotions expressed by a user in their tweets. Plutchik's wheel outlines eight primary emotions arranged as four pairs of opposing dualities: Joy - Sadness, Surprise - Anticipation, Anger - Fear, and Trust - Disgust. We utilize Plutchik Transformer (Sawhney et al., 2020), a BERT model fine-tuned on Emonet (Abdul-Mageed and Ungar, 2017), a dataset of 790,059 tweets labeled across 8 primary emotions as per Plutchik's wheel of emotions. Owing to the success of pre-training language models in NLP, Plutchik Transformer jointly learns textual and emotion features for representation learning of user tweets for subsequent suicidal intent detection. We extract a 768-dimension encoding from the `[CLS]`[3] token of the penultimate transformer layer, which is densely connected with an 8-dimensional output layer representative of each primary emotion.

---

[3] Empirically, the `[CLS]` token performed better than taking the average of the output vectors over all tokens.

Tweet to be assessed: We encode each tweet to be assessed t_i as:

$$\mathbf{T}'_{\mathbf{i}} = \texttt{PlutchikTransformer}(t_i) \tag{1}$$

where $\mathbf{T}'_{\mathbf{i}} \in \mathbb{R}^{768}$ is linearly transformed using a dense layer to $\mathbf{T}_{\mathbf{i}} \in \mathbb{R}^{d}$ with dimension d.

Historical Tweet Encoding: A holistic representation of users' emotional states can be indicative of variations in risk markers over time (Aragón et al., 2019; Tarrier et al., 2007; Links et al., 2008). To this end, we utilize Plutchik Transformer to encode each historical tweet h^i_k to an emotion representation ($\mathbf{e}^{\mathbf{i}}_{\mathbf{k}} \in \mathbb{R}^{768}$) defined as:

$$\mathbf{e}^{\mathbf{i}}_{\mathbf{k}} = \texttt{PlutchikTransformer}(h^i_k) \tag{2}$$

3.3 Temporal Modeling of Historical Tweets

Building on these natural irregularities in posting times of historical tweets (Wojcik and Hughes, 2019), we propose the use of ON-LSTM (Shen et al., 2018) to encode the sequence of a user's historical tweet emotion representations $\mathbf{e}^{\mathbf{i}}_{\mathbf{k}}$ to capture the variation in their mental and emotional states over time, forming a Time-Sensitive Emotion LSTM (TSE-LSTM). In our TSE-LSTM, we introduce a time-sensitive long-term gate $\tilde{\mathbf{f}}_{\mathbf{k}}$, which contains older historic emotional context. Additionally, we propose a short-term gate $\tilde{\mathbf{i}}_{\mathbf{k}}$ that encodes recent historic tweets, as shown in Figure 3. We then feed the time-lapsed Δ_k from the previous tweet and the historical emotional representation $\mathbf{e}^{\mathbf{i}}_{\mathbf{k}}$ of each tweet h^i_k to a TSE-LSTM cell. This design aids TSE-LSTM to learn two probability distributions $\mathbf{p}_{\tilde{\mathbf{f}}_{\mathbf{k}}}$ and $\mathbf{p}_{\tilde{\mathbf{i}}_{\mathbf{k}}}$ corresponding to the long-term and short-term gates, respectively. Psychological studies show that a user's recent emotions can be more indicative of their current mental state (Fawcett et al., 1990; Homan et al., 2014). To this end, we set the update frequency of the short-term gate higher than the long-term gate to increase the influence of their more recent emotional context. To impose this natural ordering of frequency updates, we apply cumulative sum ($cumsum$) operation to the probability distributions $\mathbf{p}_{\tilde{\mathbf{f}}_{\mathbf{k}}}$ and $\mathbf{p}_{\tilde{\mathbf{i}}_{\mathbf{k}}}$:

$$\mathbf{p}_{\tilde{\mathbf{f}}} = \sigma(\mathbf{W}_{\tilde{\mathbf{f}}}(\mathbf{e}^{\mathbf{i}}_{\mathbf{k}} \oplus \Delta_k) + \mathbf{U}_{\tilde{\mathbf{f}}}(\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k-1}} \oplus \Delta_k) + \mathbf{b}_{\tilde{\mathbf{f}}}) \tag{3}$$

$$\mathbf{p}_{\tilde{\mathbf{i}}} = \sigma(\mathbf{W}_{\tilde{\mathbf{i}}}(\mathbf{e}^{\mathbf{i}}_{\mathbf{k}} \oplus \Delta_k) + \mathbf{U}_{\tilde{\mathbf{i}}}(\tilde{\mathbf{H}}_{\mathbf{k-1}} \oplus \Delta_k) + \mathbf{b}_{\tilde{\mathbf{i}}}) \tag{4}$$

$$\tilde{\mathbf{f}}_{\mathbf{k}} = \overrightarrow{cumsum}(\mathbf{p}_{\tilde{\mathbf{f}}}), \tilde{\mathbf{i}}_{\mathbf{k}} = \overleftarrow{cumsum}(\mathbf{p}_{\tilde{\mathbf{i}}}) \tag{5}$$

where σ represents softmax, $\oplus$ denotes concatenation and $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k-1}}$ is the previous hidden state. The arrow above $cumsum$ indicates its direction. $\mathbf{W}_{\tilde{\mathbf{f}}}$, $\mathbf{W}_{\tilde{\mathbf{i}}}$, $\mathbf{U}_{\tilde{\mathbf{f}}}$, $\mathbf{U}_{\tilde{\mathbf{i}}}$, $\mathbf{b}_{\tilde{\mathbf{f}}}$ and $\mathbf{b}_{\tilde{\mathbf{i}}}$ are learnable parameters. Following $cumsum$'s properties, the values in the long-term gate $\tilde{\mathbf{f}}_{\mathbf{k}}$ are monotonically increasing from 0 to 1, and those in the short-term gate $\tilde{\mathbf{i}}_{\mathbf{k}}$ are monotonically decreasing from 1 to 0.

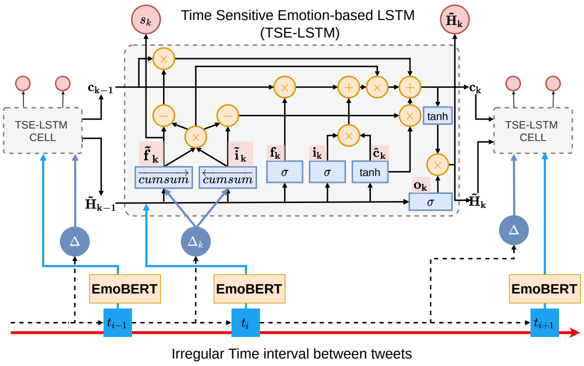


Figure 3: Detailed structure of the TSE-LSTM cell. Figure is adapted from (Gao et al., 2020).

For each historic tweet h^i_k, the long-term gate $\tilde{\mathbf{f}}_{\mathbf{k}}$ controls the historic emotional context to be discarded, and the short-term gate $\tilde{\mathbf{i}}_{\mathbf{k}}$ controls the importance of recent historic emotions. To obtain complete contextual information of overlapping context in $\tilde{\mathbf{f}}_{\mathbf{k}}$ and $\tilde{\mathbf{i}}_{\mathbf{k}}$, we introduce a historic overlap vector $\mathbf{w}_{\mathbf{k}}$ that uses the standard forget and input gates, $\mathbf{f}_{\mathbf{k}}$ and $\mathbf{i}_{\mathbf{k}}$, respectively. We define the new update function for TSE-LSTM's cell state $\mathbf{c}_{\mathbf{k}}$ as:

$$\hat{\mathbf{c}}_{\mathbf{k}} = tanh(\mathbf{W}_{\mathbf{c}}\mathbf{e}^{\mathbf{i}}_{\mathbf{k-1}} + \mathbf{U}_{\mathbf{c}}\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k-1}} + \mathbf{b}_{\mathbf{c}}) \tag{6}$$

$$\mathbf{w}_{\mathbf{k}} = \tilde{\mathbf{f}}_{\mathbf{k}} \odot \tilde{\mathbf{i}}_{\mathbf{k}} \tag{7}$$

$$\begin{aligned}\mathbf{c}_{\mathbf{k}} &= \mathbf{w}_{\mathbf{k}} \odot (\mathbf{f}_{\mathbf{k}} \odot \mathbf{c}_{\mathbf{k-1}} + \mathbf{i}_{\mathbf{k}} \odot \tilde{\mathbf{c}}_{\mathbf{k}}) \\ &\quad + (\tilde{\mathbf{f}}_{\mathbf{k}} - \mathbf{w}_{\mathbf{k}}) \odot \mathbf{c}_{\mathbf{k-1}} + (\tilde{\mathbf{i}}_{\mathbf{k}} - \mathbf{w}_{\mathbf{k}}) \odot \hat{\mathbf{c}}_{\mathbf{k}}\end{aligned} \tag{8}$$

$$\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}} = \mathbf{o}_{\mathbf{k}} \odot tanh(\mathbf{c}_{\mathbf{k}}) \tag{9}$$

where computation for the intermediate cell state $\hat{\mathbf{c}}_{\mathbf{k}}$, output gate $\mathbf{o}_{\mathbf{k}}$, the hidden state $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}$ are the same as in the standard LSTM and $\mathbf{W}_{\mathbf{c}}$, $\mathbf{U}_{\mathbf{c}}$, $\mathbf{b}_{\mathbf{c}}$ are network parameters. The hidden state $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}$ represents the learned emotional context of the user.

3.4 Learning Emotional Phase Progression

We now describe how we use the emotional context learned by the TSE-LSTM to capture emotional phase progression patterns for a user over time. We then describe PHASE's Phase Adaptive Convolutions (PACs) that capture user features closely related to the user's current state through convolutions over these learned emotional phases. The PACs thus extract a phase-aware emotional user representation for suicide ideation detection.

Emotion Phase Variation: We leverage the historical emotional context $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}$ from the TSE-LSTM to extract temporal variations in a user's emotional state for a macroscopic view of the progression of *emotional phases*. Building on the work of Gao et al. (2020), we capture the onset of a new emotional phase by observing the proportion of historic context discarded to update the cell state $\mathbf{c_k}$. When almost no historical emotional context is used to update the cell state c_k, we say that a new emotional phase of the user has begun. Formally, we use a phase split point (s_k) that represents the time, before which all the emotional historic context is discounted ($\mathbf{p}_{\tilde{\mathbf{f}}_{\mathbf{k}}}$), as $s_k = argmax(\mathbf{p}_{\tilde{\mathbf{f}}})$, as shown in Figure 4 (Gao et al., 2020). Intuitively, a large value of s_k means little historic context is used to update the state cell c_k, indicating the onset of a new emotional phase; whereas, a smaller value of s_k suggests a long-term dependency of the emotions expressed in the tweet (h^i_k) on historic emotions. Since $argmax$ is non-differentiable, we estimate the phase split point (s_k) as:

$$s_k \approx \sum_{i=1}^{N_h} i \times \mathbf{p}_{\tilde{\mathbf{f}}}(i) = N_h\left(1 - \frac{1}{N_h}\sum_{i=1}^{N_h}\tilde{\mathbf{f}}_{\mathbf{k}}(i)\right) + 1 \qquad (10)$$

where N_h is the dimension of $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}$, $\tilde{\mathbf{f}}_{\mathbf{k}}(i)$ and $\mathbf{p}_{\tilde{\mathbf{f}}}(i)$ are i^{th} values in the long-term gate $\tilde{\mathbf{f}}_{\mathbf{k}}$, and $\mathbf{p}_{\tilde{\mathbf{f}}}$.

We then compute the elapsed time between two consecutive phases by measuring the difference between the proportion of historic context discarded at each timestep. For each emotional phase of a user within an observation window of length L^w, we define this phase variation time $\mathbf{\Delta s}$ as:

$$\mathbf{\Delta s} = \sigma(\overrightarrow{cumsum}(s_{k-L^w}, \cdots, s_k)) \qquad (11)$$

Phase Adaptive Convolution (PAC): We now extract features from the emotional phase build-up leading towards the tweet to be assessed. The PAC extracts features from the learned phase-wise progression of a user's temporal emotional context in the most recent emotional phase, as shown in Figure 4. We feed the concatenated historical hidden states $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k-L^w:k}} = [\tilde{\mathbf{H}}_{\mathbf{k-L^w}}, \cdots, \tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}]$, in the observation window L^w, as an input to a weighted temporal convolution. Naturally, emotions corresponding to more recent phases of a user are more indicative of their current mental state, and should be more influential (Larsen et al., 2009). Hence, we weigh the importance of the learned historical emotional context through the phase variation time Δs (Gao et al., 2020). We perform a convolution with a p^{th} 1-dimensional learnable kernel (m^j_p) for each j^{th} hidden state in the observation window as:

$$\mathbf{u^p} = \mathbf{m_p} * \tilde{\mathbf{H}}_{\mathbf{k-L^w:k}} = \sum_{j=1}^{N_h} \mathbf{m^j_p} * (\tilde{\mathbf{H}}^{\mathbf{j}}_{\mathbf{k-L^w:k}} \odot \mathbf{\Delta s}) \qquad (12)$$

where $*$ is convolution operation, $\mathbf{u_p}$ is output of p^{th} kernel of size L^w. We concatenate all extracted features as $\mathbf{u} = [\mathbf{u^1}, \cdots, \mathbf{u^{N_h}}] \in \mathbb{R}^{N_h}$ to obtain a user's phase-aware emotion representation.

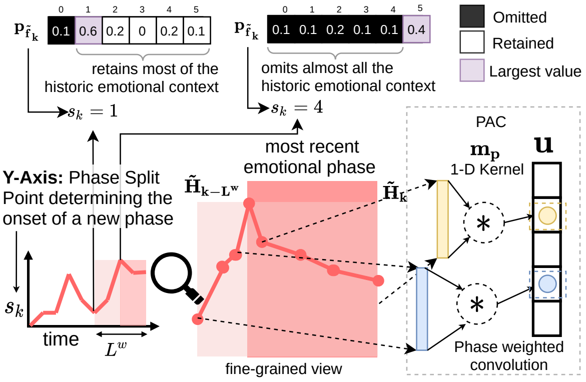


Figure 4: The Phase-Adaptive Convolution takes the historical hidden state $\tilde{\mathbf{H}}^{\mathbf{i}}_{\mathbf{k}}$ and the phase split-point s_k within the observation window length L^w as an input and learns a user's representation through their temporal patterns ($\mathbf{u}$), using phase-weighted convolutions.

3.5 PHASE Joint Network Optimization

Finally, we concatenate encoded representations of the tweet to be assessed $\mathbf{T_i}$ and the historic emotional context $\mathbf{u}$, followed by softmax (σ) over a dense layer with a Rectified Linear Unit ($ReLU$).

$$\hat{\mathbf{y}}_{\mathbf{i}} = \sigma(ReLU(\mathbf{W_y}(\mathbf{T_i} \oplus \mathbf{u}) + \mathbf{b_y})) \qquad (13)$$

where $\hat{\mathbf{y}}_{\mathbf{i}}$ is the final suicide risk assessment and $\{\mathbf{W_y}, \mathbf{b_y}\}$ are learnable network parameters.

Tweets with SI present form a very small proportion of the data (Ji et al., 2019). To address this problem of class imbalance *(the imbalance is much greater in the real world)*, we train PHASE using Class-Balanced Focal Loss (Lin et al., 2017; Cui et al., 2019). This loss function re-weights loss inversely with the effective number of samples per class, thereby yielding a class-balanced loss $\mathcal{L}$ as:

$$\mathcal{L} = \mathrm{CB}_{focal}(\hat{\mathbf{y}}_{\mathbf{i}}, y_i; \beta, \gamma) \qquad (14)$$

where CB_{focal} is class-balanced focal loss, $\hat{\mathbf{y}}_{\mathbf{i}}$ is the predicted label and y_i is the label of the tweet to be assessed. β and γ are hyperparameters.

4 Experimental Setup

4.1 Data and Preprocessing

We build on an existing Twitter data curated by Mishra et al. (2019). The data includes 34,306 tweets authored by 32,558 unique users. These tweets were identified based on a lexicon of 143 suicidal phrases (e.g., "wanting to die", "last day"). Two students of Psychology annotated the data under the supervision of a professional clinical psychologist, achieving a Cohen's Kappa score of 0.72, under the below guidelines (Sawhney et al., 2018b):

Suicidal Intent (SI) Present: Tweets where suicide ideation or previous attempts are discussed in a somber and non-flippant tone.

Suicidal Intent (SI) Absent: Tweets with no evidence for risk of suicide, e.g., song lyrics, condolence message, awareness, news.

The resulting dataset contains 3984 suicidal tweets. The Twitter timeline was collected for each user, spanning over ten years from 2009 to 2019. The number of historical tweets (748 ± 789) and the time difference between consecutive tweets (2 ± 24 days) are indicative of large variations across users. 4070 users were found to have no historical tweets.

We perform a stratified 70:10:20 split, such that the train, validation, and test sets consist of 24014, 3431, and 6861 tweets, respectively, and ensure that there is no overlap between users in these sets.

4.2 Baselines and Training Setup

We evaluate PHASE on macro F1 and recall (SI present) with both tweet- and user-level methods.

Tweet-level Non-contextual Baselines

RF + TF (Sawhney et al., 2018b): Extracts features including statistical, LIWC features, n-grams (up to 4), and POS counts from the tweet to be assessed and feeds them to a Random Forest (RF) classifier.

C-LSTM (Sawhney et al., 2018a): A deep neural network having a CNN followed by an LSTM to extract short and long range features in a tweet.

User-level Contextual Baselines

C-CNN (Gaur et al., 2019): A model that is fed GloVe encoded tweets as a concatenated bag of tweets, non-sequentially to a contextual CNN (Shin et al., 2018) with max pooling (Shing et al., 2018).

Suicide Detection Model (SDM) (Cao et al., 2019): Historical tweets encoded using fine-tuned FastText embeddings are fed to a regular LSTM followed by a tweet-level attention mechanism.

DualContextBert (Matero et al., 2019): Best performing model at CLPsych 2019. BERT embeddings of each historical tweet are sequentially fed to a regular RNN followed by tweet-level attention.

Exponential Decay (Sinha et al., 2019): A deep neural network that models encodes each historical tweet using Glove embeddings followed by a BiLSTM with attention. The historical embeddings are then aggregated using an exponential decay.

Setup: We set hyperparameters for all models based on the validation macro F1 score. We use grid search to explore: $N_h \in \{128, 256, 512\}$, $\delta \in \{0.0, 0.1, \cdots, 0.5\}$, $\beta \in \{0.99, 0.999, 0.9999\}$ and $\gamma \in \{1.0, 1.5, 2.0\}$, initial learning rate $I_{lr} \in \{0.01, 0.001, 0.0001\}$, $L^w \in \{1, 2, \cdots, 16\}$. We found the optimal hyperparameters as: N_h=512, δ=0.5, β =0.9999, γ=2, I_{lr}=0.0001, L^w=5. We implement all methods with PyTorch 1.6, and optimize PHASE using AdamW with a batch size of 128 for 30 epochs in 167 mins on a Tesla K80 GPU. We use the cosine scheduler (Gotmare et al., 2018) with a warmup step of 5.

5 Results and Analysis

5.1 Performance Comparison

Model	Macro F1↑	Recall$_s$↑	Acc↑
RF+TF	0.513	0.536	0.548
C-LSTM	0.588	0.597	0.602
C-CNN	0.729	0.587	0.803
SDM	0.743	0.755	0.819
DualContextBert	0.767	0.786	0.823
Exponential Decay	0.737	0.759	0.828*
PHASE	**0.805***	**0.812***	**0.856***

Table 1: Median of results over 5 different runs. * indicates improvement over DualContextBert is significant ($p < 0.005$) under Wilcoxon's Signed Rank test.

We observe from Table 1 that PHASE significantly ($p < 0.005$) outperforms all baselines. We note that contextual models outperform the non-contextual RF+TF and C-LSTM, as they learn a holistic representation of a user's mental state. Amongst contextual models, we note that models that factor in the temporal sequence of historical tweets outperform the non temporal C-CNN, that models tweets as a bag-of-tweets. Thereby validating the utility of temporal context for suicide ideation detection. PHASE significantly outperforms state-of-the-art contextual models. We postulate this to PHASE's ability to capture irregularities in tweeting patterns and learning emotional phase

PHASE (Ablative) Components	**F1↑**	**Recall$_s$↑**
Current Tweet only (C)	0.731	0.597
C + History (HST) + LSTM	0.780	0.794
C + HST + TSE-LSTM	0.796*	0.788
PHASE:C+HST+TSE-LSTM+PAC	**0.805***	**0.812***

Table 2: * shows significant improvements compared to C + HST + LSTM ($p < 0.005$). We use Plutchik Transformer as the tweet encoder for comparison.

progressions, unlike DualContextBERT and SDM, that ignore both the time- and phase-sensitive and emotional aspects of historical context. PHASE outperforms Exponential Decay, as PHASE adaptively learns progressions of emotional phases, rather than assuming a user's behavior to follow a specific trajectory that might not generalize well across users. This observation is in line with psychological research (Joiner Jr, 2002; Giletta et al., 2015) that shows the progressive build-up to suicidality varies across individuals, that PHASE is able to capture better than competitive models.

5.2 PHASE Components Ablation Study

We perform an ablation study to probe the effectiveness of each component of PHASE, as shown in Table 2, starting from the base (Current) model that does not use historical tweets. On modeling the temporal dependencies in historical tweets with a standard LSTM along with the current tweet, we note drastic improvements, revalidating the prominence of user-level context to infer the suicidality of a user. We then observe that on factoring in time-sensitivity through the TSE-LSTM, there is a significant ($p < 0.005$) improvement in the macro F1 score, but there is no gain in Recall. We believe even though the model gains additional user context by factoring in the time irregularities between tweets, the model does not improve drastically, as it still assumes a continuous smooth progression of the user's emotions in time. This assumption hinders the model's ability to capture the macroscopic context acquired by analyzing the phase like progressions of a user's emotional states (Homan et al., 2014). On adding the PAC that learns phase-aware user representations by extracting emotional progression patterns from their historical emotional context (TSE-LSTM), we observe significant ($p < 0.005$) improvements. We attribute this improvement to the PAC as it adaptively learns and captures a user's emotional phase-wise build-up towards their most recent tweet to be assessed, to correctly contextualize suicidal intent, validating the utility of phase-aware modeling.

5.3 Probing Plutchik Transformer: Encoder Analysis

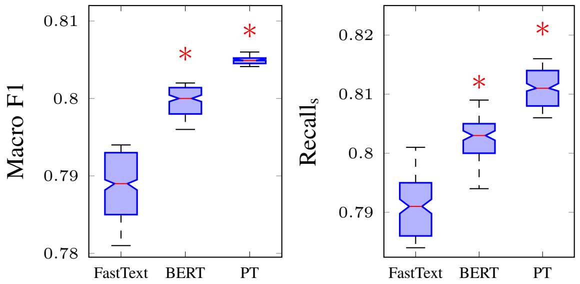


Figure 5: Performance with different tweet encoders over 10 different runs. PT: Plutchik Transformer. All improvements (*) are significant ($p < 0.005$) under Wilcoxon's Signed Rank test.

We now analyze PHASE's performance using different encoders to learn representations for tweets. Overall, we observe that transformers outperform previously used static word embeddings (FastText). Additionally, we observe that Plutchik Transformer, based on Plutchik's wheel of emotions, significantly improves PHASE's performance over the pre-trained BERT used by Matero et al. (2019). This observation revalidates the importance of specific emotional context, as opposed to the more general language features learned by BERT alone.

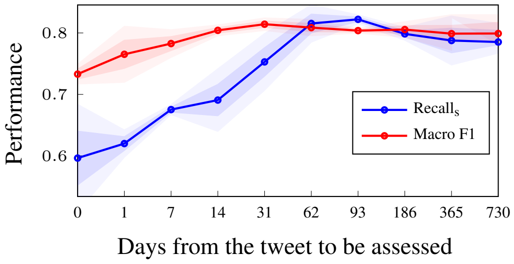
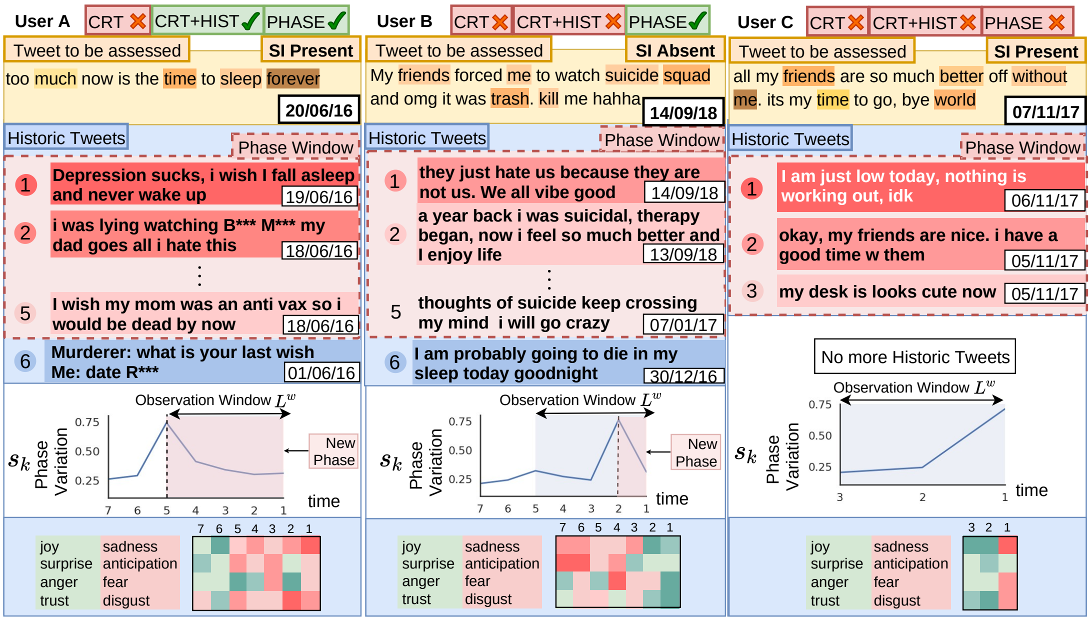


Figure 6: Influence of historical context (upto d days)

5.4 How much Historical Context is useful?

We explore PHASE's performance variation with the amount of historical lookback in terms of number of days in Figure 6. We observe that PHASE's performance improves as we factor in more historical tweets, going back up to a few months, likely as PHASE gains more context of users' emotional progressions. As we further increase historical lookback beyond several (> 3) months, we observe that PHASE's performance saturates. This observation is in line with psychological studies (Selby et al.,



User A — Tweet to be assessed: too much now is the time to sleep forever 20/06/16

1. Depression sucks, i wish I fall asleep and never wake up 19/06/16
2. i was lying watching B*** M*** my dad goes all i hate this 18/06/16
5. I wish my mom was an anti vax so i would be dead by now 18/06/16
6. Murderer: what is your last wish Me: date R*** 01/06/16

User B — Tweet to be assessed: My friends forced me to watch suicide squad and omg it was trash. kill me hahha 14/09/18

1. they just hate us because they are not us. We all vibe good 14/09/18
2. a year back i was suicidal, therapy began, now i feel so much better and I enjoy life 13/09/18
5. thoughts of suicide keep crossing my mind i will go crazy 07/01/17
6. I am probably going to die in my sleep today goodnight 30/12/16

User C — Tweet to be assessed: all my friends are so much better off without me. its my time to go, bye world 07/11/17

1. I am just low today, nothing is working out, idk 06/11/17
2. okay, my friends are nice. i have a good time w them 05/11/17
3. my desk is looks cute now 05/11/17

Figure 7: We study three users with their tweet to be assessed, historic tweets (chronologically ordered), and timestamps, showing how PHASE can aid human moderators and clinical psychologists with explainable predictions. We visualize self-attention (averaged over all 12 Plutchik Transformer heads) per token, where darker intensity denotes higher attention. The graphs show the phase split value s_k for each user over time. We also show emotional phase progression for further interpretability, where a peak represents the onset of a new phase. Further, we show detailed phase variation by visualizing the Plutchik-based emotion (learned weights) duality for historical tweets.

2013; Kaplow et al., 2014; Glenn et al., 2020), that highlight the diminishing importance of a user's emotions over longer time periods in assessing their current mental state and associated suicide risk.

5.5 PHASE Analysis and Interpretation

We now analyze PHASE's preliminary assessment in Figure 7 to elucidate on PHASE's interpretability to aid subsequent human-in-the-loop risk assessment. First, for User A, we see no apparent signs of suicidal intent in their tweet to be assessed, and if analyzed in isolation, is not sufficient to ascertain risk. However, User A's historical tweets add context to models (e.g. PHASE) that leverage temporal emotional cues to identify suicidal intent correctly. Next, we analyze a complex case, User B, where we observe *phase*-like progressions in their emotions over time. Although User B historically did show negative emotions, recently, User B shows more positive behavior, akin to the onset of a new emotional phase characterized by joy and trust. PHASE's design enables it to learn User B's emotional progression adaptively and correctly predicts User B's tweet to be analyzed as having no suicidal intent, unlike other models that incorrectly assess this as a tweet having suicidal intent. Lastly, we also present the complicated case of User C, where all models fail to explicate the future challenges in online data-driven suicide ideation. Specifically, we find that all models are unable to accurately ascertain suicidal intent where there is little historical context consisting of fluctuating emotions, highlighting the challenges associated with new or alternate accounts of users, amongst other complexities (Shea, 1999; O'Connor and Portzky, 2018).

6 Ethical and Practical Considerations

Emphasizing the sensitive nature of this work, we acknowledge the trade-off between privacy and effectiveness (Eskisabel-Azpiazu et al., 2017), and utilize publicly available Twitter data in a purely observational (Norval and Henderson, 2017; Broer, 2020), and non-intrusive manner. We separate user data from all other data on protected servers linked only through anonymous IDs, and we perform automatic de-identification of the dataset using named entity recognition (Benton et al., 2017a,b). All examples shown in this work have been paraphrased to protect user privacy (Fiesler and Proferes, 2018; Chancellor et al., 2019a,b). We ensure that this analysis is shared selectively and subject to IRB approval (Zimmer, 2009, 2010) to avoid misuse such as Samaritan's Radar (Hsin et al., 2016). We acknowledge that suicidality is subjective (Keilp et al., 2012) and that the interpretation of this analysis may vary across individuals (Puschman, 2017). We further acknowledge that suicide risk exists on a diverse spectrum (Bryan and Rudd, 2006), rather than at a binary level, and that the studied data may be susceptible to demographic, annotator, and medium-specific biases (Hovy and Spruit, 2016). Finally, our work does not make any diagnostic claims related to suicide. PHASE should form part of a distributed human-in-the-loop (de Andrade et al., 2018) system for finer interpretation of risk.

7 Conclusion

Motivated by the rising exhibition of suicide ideation on social media, we present PHASE. Building on psychological studies analyzing the emotional spectrum and mental health of users, PHASE adaptively learns emotional phase-aware user representations through historical tweeting activity for suicidal ideation detection. We propose multiple modeling innovations in PHASE components: contextualized historical emotion representations (Plutchik Transformer), time-sensitive emotion LSTM (TSE-LSTM), and a phase-adaptive convolution (PAC). We demonstrate that modeling user phases explicitly increases the predictive power in assessing suicidality in tweets. In a qualitative analysis, we show how PHASE can aid social media moderators and clinical psychologists in subsequent assessment by displaying its predictions together with the learned emotional phases. Through PHASE, we hope to form a future component in a larger human-in-the-loop infrastructure for suicide prevention.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. EmoNet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 718–728, Vancouver, Canada. Association for Computational Linguistics.

Norberto Nuno Gomes de Andrade, Dave Pawson, Dan Muriello, Lizzy Donahue, and Jennifer Guadagno. 2018. Ethics and artificial intelligence: Suicide prevention on facebook. *Philosophy & Technology*, 31(4):669–684.

Mario Ezra Aragón, Adrian Pastor López-Monroy, Luis Carlos González-Gurrola, and Manuel Montes-y Gómez. 2019. Detecting depression in social media using fine-grained emotions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1481–1486, Minneapolis, Minnesota. Association for Computational Linguistics.

Courtney Bagge and Augustine Osman. 1998. The suicide probability scale: Norms and factor structure. *Psychological Reports*, 83(2):637–638.

Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 65–74.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017a. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Adrian Benton, Margaret Mitchell, and Dirk Hovy. 2017b. Multitask learning for mental health conditions with limited social media data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 152–162, Valencia, Spain. Association for Computational Linguistics.

Scott R Braithwaite, Christophe Giraud-Carrier, Josh West, Michael D Barnes, and Carl Lee Hanson. 2016. Validating machine learning algorithms for twitter data against established measures of suicidality. *JMIR Mental Health*, 3(2):e21.

Tineke Broer. 2020. Technology for our future? exploring the duty to report and processes of subjectification relating to digitalized suicide prevention. *Information*, 11(3):170.

Amy Bruckman. 2002. Studying the amateur artist: A perspective on disguising data collected in human subjects research on the internet. *Ethics and Information Technology*, 4(3):217–231.

Craig J. Bryan. 2020. Chapter 4 - the temporal dynamics of the wish to live and the wish to die among suicidal individuals. In Andrew C. Page and Werner G.K. Stritzke, editors, *Alternatives to Suicide*, pages 71 – 88. Academic Press.

Craig J Bryan and M David Rudd. 2006. Advances in the assessment of suicide risk. *Journal of clinical psychology*, 62(2):185–200.

Craig J Bryan and M David Rudd. 2016. The importance of temporal dynamics in the transition from suicidal thought to behavior. *Clinical Psychology: Science and Practice*, 23(1):21–25.

Lei Cao, Huijun Zhang, Ling Feng, Zihan Wei, Xin Wang, Ningyun Li, and Xiaohao He. 2019. Latent suicide risk detection on microblog via suicide-oriented word embeddings and layered attention. In

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1718–1728, Hong Kong, China. Association for Computational Linguistics.

Catherine Card. 2018. How Facebook AI helps suicide prevention. *Facebook Newsroom*.

Stevie Chancellor, Eric PS Baumer, and Munmun De Choudhury. 2019a. Who is the" human" in human-centered machine learning: The case of predicting mental health from social media. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–32.

Stevie Chancellor, Michael L. Birnbaum, Eric D. Caine, Vincent M. B. Silenzio, and Munmun De Choudhury. 2019b. A taxonomy of ethical tensions in inferring mental health states from social media. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 79–88, New York, NY, USA. Association for Computing Machinery.

Xuetong Chen, Martin D. Sykora, Thomas W. Jackson, and Suzanne Elayan. 2018. What about mood swings: Identifying depression on twitter with temporal measures of emotions. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1653–1660, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Glen Coppersmith, Mark Dredze, and Craig Harman. 2014. Quantifying mental health signals in Twitter. In *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 51–60, Baltimore, Maryland, USA. Association for Computational Linguistics.

Glen Coppersmith, Ryan Leary, Patrick Crutchley, and Alex Fine. 2018. Natural language processing of social media as screening for suicide risk. *Biomedical Informatics Insights*, 10:117822261879286.

Glen Coppersmith, Kim Ngo, Ryan Leary, and Anthony Wood. 2016. Exploratory analysis of social media prior to a suicide attempt. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 106–117, San Diego, CA, USA. Association for Computational Linguistics.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277.

Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting depression via social media. In *Seventh international AAAI conference on weblogs and social media*.

Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 2098–2110.

Elena Domínguez, García and Pablo Fernández, Berrocal. 2018. The association between emotional intelligence and suicidal behavior: a systematic review. *Frontiers in psychology*, 9:2380.

Christopher W Drapeau and John L McIntosh. 2020. Usa suicide 2018: Official final data.

Jingcheng Du, Yaoyun Zhang, Jianhong Luo, Yuxi Jia, Qiang Wei, Cui Tao, and Hua Xu. 2018. Extracting psychiatric stressors for suicide from social media using deep learning. *BMC medical informatics and decision making*, 18(Suppl 2):43–43.

Amaia Eskisabel-Azpiazu, Rebeca Cerezo-Menéndez, and Daniel Gayo-Avello. 2017. An ethical inquiry into youth suicide prevention using social media mining. *Internet Research Ethics for the Social Age*, 227.

Cecilia A. Essau. 2005. Frequency and patterns of mental health services utilization among adolescents with anxiety and depressive disorders. *Depression and Anxiety*, 22(3):130–137.

Jan Fawcett, William A Scheftner, Louis Fogg, David C Clark, Michael A Young, Don Hedeker, and Robert Gibbons. 1990. Time-related predictors of suicide in major affective disorder. *The American journal of psychiatry*.

Casey Fiesler and Nicholas Proferes. 2018. "participant" perceptions of twitter research ethics. *Social Media+ Society*, 4(1):2056305118763366.

Lucie Flek. 2020. Returning the n to nlp: Towards contextually personalized classification models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7828–7838.

Joseph C Franklin, Jessica D Ribeiro, Kathryn R Fox, Kate H Bentley, Evan M Kleiman, Xieyining Huang, Katherine M Musacchio, Adam C Jaroszewski, Bernard P Chang, and Matthew K Nock. 2017. Risk factors for suicidal thoughts and behaviors: a meta-analysis of 50 years of research. *Psychological bulletin*, 143(2):187.

King wa Fu, Ka Y. Liu, and Paul S. F. Yip. 2007. Predictive validity of the chinese version of the adult suicidal ideation questionnaire: Psychometric properties and its short version. *Psychological Assessment*, 19(4):422–429.

Junyi Gao, Cao Xiao, Yasha Wang, Wen Tang, Lucas M Glass, and Jimeng Sun. 2020. Stagenet: Stage-aware neural networks for health risk prediction. In *Proceedings of The Web Conference 2020*, pages 530–540.

Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan, Ramakanth Kavuluru, Amit Sheth, Randy Welton, and Jyotishman Pathak. 2019. Knowledge-aware assessment of severity of suicide risk for early intervention. In *The World Wide Web Conference*, pages 514–525.

Matteo Giletta, Mitchell J Prinstein, John RZ Abela, Brandon E Gibb, Andrea L Barrocas, and Benjamin L Hankin. 2015. Trajectories of suicide ideation and nonsuicidal self-injury among adolescents in mainland china: Peer predictors, joint development, and risk for suicide attempts. *Journal of consulting and clinical psychology*, 83(2):265.

Jeffrey J Glenn, Alicia L Nobles, Laura E Barnes, and Bethany A Teachman. 2020. Can text messages identify suicide risk in real time? a within-subjects pilot examination of temporally sensitive markers of suicide risk. *Clinical Psychological Science*, 8(4):704–722.

Robert N Golden, Carla Weiland, and Fred Peterson. 2009. *The truth about illness and disease*. Infobase Publishing.

Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In *International Conference on Learning Representations*.

Sharath Chandra Guntuku, David B Yaden, Margaret L Kern, Lyle H Ungar, and Johannes C Eichstaedt. 2017. Detecting depression and mental illness on social media: an integrative review. *Current Opinion in Behavioral Sciences*, 18:43–49.

Keith M. Harris and Melissa Ting-Ting Goh. 2016. Is suicide assessment harmful to participants? findings from a randomized controlled trial. *International Journal of Mental Health Nursing*, 26(2):181–190.

Keith M. Harris, Jia-Jia Syu, Owen D. Lello, Y. L. Eileen Chew, Christopher H. Willcox, and Roger H. M. Ho. 2015. The ABC's of suicide risk assessment: Applying a tripartite approach to individual evaluations. *PLOS ONE*, 10(6):e0127442.

Holly Hedegaard, Sally C Curtin, and Margaret Warner. 2020. Increase in suicide mortality in the united states, 1999–2018.

Christopher Homan, Ravdeep Johar, Tong Liu, Megan Lytle, Vincent Silenzio, and Cecilia Ovesdotter Alm. 2014. Toward macro-insights for suicide prevention: Analyzing fine-grained distress at scale. In *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 107–117.

Dirk Hovy and Shannon L Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598.

Honor Hsin, John Torous, and Laura Roberts. 2016. An adjuvant role for mobile health in psychiatry. *JAMA Psychiatry*, 73(2):103.

Jared Jashinsky, Scott H. Burton, Carl L. Hanson, Josh West, Christophe Giraud-Carrier, Michael D. Barnes, and Trenton Argyle. 2014. Tracking suicide risk factors through twitter in the US. *Crisis*, 35(1):51–59.

Shaoxiong Ji, Shirui Pan, Xue Li, Erik Cambria, Guodong Long, and Zi Huang. 2019. Suicidal ideation detection: A review of machine learning methods and applications. *arXiv preprint arXiv:1910.12611*.

Shaoxiong Ji, Celina Ping Yu, Sai fu Fung, Shirui Pan, and Guodong Long. 2018. Supervised learning for suicidal ideation detection in online user content. *Complexity*, 2018:1–10.

Thomas E Joiner Jr. 2002. The trajectory of suicidal behavior over time. *Suicide and life-threatening Behavior*, 32(1):33–41.

Julie B Kaplow, Polly Y Gipson, Adam G Horwitz, Bianca N Burch, and Cheryl A King. 2014. Emotional suppression mediates the relation between adverse life events and adolescent suicide: Implications for prevention. *Prevention science*, 15(2):177–185.

John G. Keilp, Michael F. Grunebaum, Marianne Gorlyn, Simone LeBlanc, Ainsley K. Burke, Hanga Galfalvy, Maria A. Oquendo, and J. John Mann. 2012. Suicidal ideation and the subjective aspects of depression. *Journal of Affective Disorders*, 140(1):75–81.

Dimitris N Kiosses, Katalin Szanto, and George S Alexopoulos. 2014. Suicide in older adults: the role of emotions and cognition. *Current psychiatry reports*, 16(11):495.

Mark E Larsen, Tjeerd W Boonstra, Philip J Batterham, Bridianne O'Dea, Cecile Paris, and Helen Christensen. 2015. We feel: mapping emotion on twitter. *IEEE journal of biomedical and health informatics*, 19(4):1246–1252.

Randy J Larsen, Adam A Augustine, and Zvjezdana Prizmic. 2009. A process approach to emotion and personality: Using time as a facet of data. *Cognition and Emotion*, 23(7):1407–1426.

Kai Lei, Ying Liu, Shangru Zhong, Yongbin Liu, Kuai Xu, Ying Shen, and Min Yang. 2018. Understanding user behavior in sina weibo online social network: a community approach. *IEEE Access*, 6:13302–13316.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Paul S Links, Rahel Eynan, Marnin J Heisel, and Rosane Nisenbaum. 2008. Elements of affective instability associated with suicidal behaviour in patients with borderline personality disorder. *The Canadian Journal of Psychiatry*, 53(2):112–116.

David E Losada, Fabio Crestani, and Javier Parapar. 2020. erisk 2020: Self-harm and depression challenges. In *European Conference on Information Retrieval*, pages 557–563, Lisbon, Portugal. Springer.

Naoki Masuda, Issei Kurahashi, and Hiroko Onari. 2013. Suicide ideation of individuals in online social networks. *PloS one*, 8:e62262.

Matthew Matero, Akash Idnani, Youngseo Son, Salvatore Giorgi, Huy Vu, Mohammad Zamani, Parth Limbachiya, Sharath Chandra Guntuku, and H Andrew Schwartz. 2019. Suicide risk assessment with multi-level dual-context language and bert. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 39–44.

Puneet Mathur, Ramit Sawhney, Shivang Chopra, Maitree Leekha, and Rajiv Ratn Shah. 2020. Utilizing temporal psycholinguistic cues for suicidal intent estimation. In *Lecture Notes in Computer Science*, pages 265–271. Springer International Publishing.

Catherine M McHugh, Amy Corderoy, Christopher James Ryan, Ian B Hickie, and Matthew Michael Large. 2019. Association between suicidal ideation and suicide: meta-analyses of odds ratios, sensitivity, specificity and positive predictive value. *BJPsych open*, 5(2).

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. SNAP-BATNET: Cascading author profiling and social network graphs for suicide ideation detection on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 147–156, Minneapolis, Minnesota. Association for Computational Linguistics.

Nona Naderi, Julien Gobeill, Douglas Teodoro, Emilie Pasche, and Patrick Ruch. 2019. A baseline approach for early detection of signs of anorexia and self-harm in reddit posts. In *Proceedings of CLEF (Conference and Labs of the Evaluation Forum) 2019 Working Notes*, CONFERENCE. 9-12 September 2019.

Andrada D Neacsiu, Caitlin M Fang, Marcus Rodriguez, and M Zachary Rosenthal. 2018. Suicidal behavior and problems with emotion regulation. *Suicide and Life-Threatening Behavior*, 48(1):52–74.

Chris Norval and Tristan Henderson. 2017. Contextual consent: Ethical mining of social media for health research. *CoRR*, abs/1701.07765.

Bridianne O'dea, Stephen Wan, Philip J Batterham, Alison L Calear, Cecile Paris, and Helen Christensen. 2015. Detecting suicidality on twitter. *Internet Interventions*, 2(2):183–188.

Rory C O'Connor and Gwendolyn Portzky. 2018. Looking to the future: a synthesis of new developments and challenges in suicide research and prevention. *Frontiers in Psychology*, 9:2139.

J. E. Palmier-Claus, P. J. Taylor, F. Varese, and D. Pratt. 2012. Does unstable mood increase risk of suicide?: theory, research and practice. *Journal of Affective Disorders*, 143(1-3):5–15.

Michael J Paul and Mark Dredze. 2011. You are what you tweet: Analyzing twitter for public health. In *Fifth International AAAI Conference on Weblogs and Social Media*.

John Pestian, Henry Nasrallah, Pawel Matykiewicz, Aurora Bennett, and Antoon Leenaars. 2010. Suicide note classification using natural language processing: A content analysis. *Biomedical informatics insights*, 2010(3):19–28.

John P. Pestian, Michael Sorter, Brian Connolly, Kevin Bretonnel Cohen, Cheryl McCullumsmith, Jeffry T. Gee, Louis-Philippe Morency, Stefan Scherer, and Lesley Rohlfs and. 2016. A machine learning approach to identifying the thought markers of suicidal subjects: A prospective multicenter trial. *Suicide and Life-Threatening Behavior*, 47(1):112–121.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Theories of emotion*, pages 3–33. Elsevier.

Cornelius Puschman. 2017. Bad judgment, bad ethics? *Internet Research Ethics for the Social Age*, page 95.

Mark A. Reger, Ian H. Stanley, and Thomas E. Joiner. 2020. Suicide Mortality and Coronavirus Disease 2019—A Perfect Storm? *JAMA Psychiatry*.

Fuji Ren, Xin Kang, and Changqin Quan. 2016. Examining accumulated emotional traits in suicide blogs with an emotion topic model. *IEEE Journal of Biomedical and Health Informatics*, 20(5):1384–1396.

Jo Robinson, Georgina Cox, Eleanor Bailey, Sarah Hetrick, Maria Rodrigues, Steve Fisher, and Helen Herrman. 2016. Social media and suicide prevention: a systematic review. *Early intervention in psychiatry*, 10(2):103–121.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Ratn Shah. 2020. A time-aware transformer based model for suicide ideation detection on social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7685–7697, Online. Association for Computational Linguistics.

Ramit Sawhney, Prachi Manchanda, Puneet Mathur, Rajiv Shah, and Raj Singh. 2018a. Exploring and learning suicidal ideation connotations on social media with deep learning. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 167–175, Brussels, Belgium. Association for Computational Linguistics.

Ramit Sawhney, Prachi Manchanda, Raj Singh, and Swati Aggarwal. 2018b. A computational approach to feature extraction for identification of suicidal ideation in tweets. In *Proceedings of ACL 2018, Student Research Workshop*, pages 91–98, Melbourne, Australia. Association for Computational Linguistics.

S. Scherer, J. Pestian, and L. Morency. 2013. Investigating the speech characteristics of suicidal adolescents. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 709–713.

Edward A Selby, Shirley Yen, and Anthony Spirito. 2013. Time varying prediction of thoughts of death and suicidal ideation in adolescents: weekly ratings over 6-month follow-up. *Journal of Clinical Child & Adolescent Psychology*, 42(4):481–495.

Shawn Christopher Shea. 1999. *The practical art of suicide assessment: A guide for mental health professionals and substance abuse counselors.* John Wiley & Sons Inc.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2018. Ordered neurons: Integrating tree structures into recurrent neural networks. In *International Conference on Learning Representations*.

Joongbo Shin, Yanghoon Kim, Seunghyun Yoon, and Kyomin Jung. 2018. Contextual-cnn: A novel architecture capturing unified meaning for sentence classification. In *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 491–494. IEEE.

Han-Chin Shing, Suraj Nair, Ayah Zirikly, Meir Friedenberg, Hal Daumé III, and Philip Resnik. 2018. Expert, crowdsourced, and machine assessment of suicide risk via online postings. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 25–36, New Orleans, LA. Association for Computational Linguistics.

Han-Chin Shing, Philip Resnik, and Douglas W Oard. 2020. A prioritization model for suicidality risk assessment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8124–8137.

Pradyumna Prakhar Sinha, Rohan Mishra, Ramit Sawhney, Debanjan Mahata, Rajiv Ratn Shah, and Huan Liu. 2019. # suicidal-a multipronged approach to identify and explore suicidal ideation in twitter. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 941–950.

Merike Sisask, Airi Värnik, Kairi Kolves, Kenn Konstabel, and Danuta Wasserman. 2008. Subjective psychological well-being (who-5) in assessment of the severity of suicide attempt. *Nordic Journal of Psychiatry*, 62(6):431–435.

Kamesha Spates, Xinyue Ye, and Ashley Johnson. 2018. “i just might kill myself”: Suicide expressions on twitter. *Death studies.*

Hajime Sueki. 2015. The association of suicide-related twitter use with suicidal behaviour: a cross-sectional study of young internet users in japan. *Journal of affective disorders*, 170:155–160.

Michael Mesfin Tadesse, Hongfei Lin, Bo Xu, and Liang Yang. 2020. Detection of suicide ideation in social media forums using deep learning. *Algorithms*, 13(1):7.

Nicholas Tarrier, Patricia Gooding, Lynsey Gregg, Judith Johnson, and Richard Drake. 2007. Suicide schema in schizophrenia: The effect of emotional reactivity, negative symptoms and schema elaboration. *Behaviour Research and Therapy*, 45(9):2090–2097.

Cornelis Van Heeringen and A Marušic. 2003. Understanding the suicidal brain. *The British Journal of Psychiatry*, 183(4):282–284.

V. Venek, S. Scherer, L. Morency, A. “. Rizzo, and J. Pestian. 2017. Adolescent suicidal risk assessment in clinician-patient interaction. *IEEE Transactions on Affective Computing*, 8(2):204–215.

Stefan Wojcik and Adam Hughes. 2019. Sizing up twitter users.

Andrew Yates, Arman Cohan, and Nazli Goharian. 2017. Depression and self-harm risk assessment in online forums. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2968–2978, Copenhagen, Denmark. Association for Computational Linguistics.

Henrik D Zachrisson, Kjetil Rödje, and Arnstein Mykletun. 2006. Utilization of health services in relation to mental health problems in adolescents: A population based survey. *BMC Public Health*, 6(1).

Xingshan Zeng, Jing Li, Lu Wang, Zhiming Mao, and Kam-Fai Wong. 2020. Dynamic online conversation recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3331–3341, Online. Association for Computational Linguistics.

Xingshan Zeng, Jing Li, Lu Wang, and Kam-Fai Wong. 2019. Joint effects of context and user history for predicting online conversation re-entries. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2809–2818, Florence, Italy. Association for Computational Linguistics.

Xu Zhang, Yaxuan Ren, Jianing You, Chao Huang, Yongqiang Jiang, Min-Pei Lin, and Freedom Leung. 2017. Distinguishing pathways from negative emotions to suicide ideation and to suicide attempt: The differential mediating effects of nonsuicidal self-injury. *Journal of abnormal child psychology*, 45(8):1609–1619.

Michael Zimmer. 2009. Web search studies: Multidisciplinary perspectives on web search engines. In *International handbook of internet research*, pages 507–521. Springer.

Michael Zimmer. 2010. "but the data is already public": on the ethics of research in facebook. *Ethics and information technology*, 12(4):313–325.

Ayah Zirikly, Philip Resnik, Özlem Uzuner, and Kristy Hollingshead. 2019. CLPsych 2019 shared task: Predicting the degree of suicide risk in Reddit posts. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 24–33, Minneapolis, Minnesota. Association for Computational Linguistics.

Exploiting Definitions for Frame Identification

Tianyu Jiang and **Ellen Riloff**
School of Computing
University of Utah
Salt Lake City, UT 84112
`{tianyu, riloff}@cs.utah.edu`

Abstract

Frame identification is one of the key challenges for frame-semantic parsing. The goal of this task is to determine which frame best captures the meaning of a target word or phrase in a sentence. We present a new model for frame identification that uses a pre-trained transformer model to generate representations for frames and lexical units (senses) using their formal definitions in FrameNet. Our frame identification model assesses the suitability of a frame for a target word in a sentence based on the semantic coherence of their meanings. We evaluate our model on three data sets and show that it consistently achieves better performance than previous systems.

1 Introduction

Research on *frame semantics* has grown within the fields of natural language processing and cognitive science since the 1970s as the study of how we associate words and phrases with cognitive structures called *frames*, which characterize a small abstract scene or situation (Fillmore, 1976, 1982). The Berkeley FrameNet project (Baker et al., 1998) provides an online lexical database for frame semantics together with a corpus of annotated documents. Frame semantic parsing is the task of automatically extracting frame semantic structures from sentences. The process typically consists of three steps: *target identification*, which identifies frame-evoking predicates in the sentence; *frame identification*, which identifies the evoked frame for each target; and *argument identification*, which identifies arguments of a frame and labels them with semantic roles (frame elements). In this work, we focus on the frame identification problem.

FrameNet 1.7 contains more than 13,000 lexical units (a word lemma with a sense), each associated with a semantic frame. A polysemous word is associated with multiple lexical units (one for each sense), and is therefore linked to multiple frames. The frame identification task requires a system to identify the most relevant frame for a target word or phrase based on its sentence context. Here is an example:

> The pandemic has *sparked* a lot of problems for the economy.

Given the target word *sparked*, the goal is to determine which frame should be triggered. The word lemma *spark* has two senses in FrameNet: "with obj. ignite" and "provide the stimulus for". The former sense is associated with the *Setting_fire* frame and the latter one is associated with the *Cause_to_start* frame. The *Setting_fire* frame is defined as "this frame describes the creation of a flame...", and the *Cause_to_start* frame is defined as "a cause, animate or inanimate, causes a process, the effect, to begin". So *Cause_to_start* is the correct frame for this sentence.

Previous work has shown the success of using feature engineering with linear classification models (Johansson and Nugues, 2007) and discriminative probabilistic models (Das et al., 2010), which were later improved by applying distributed word representations and deep neural network models (Hermann et al., 2014). Syntactic information, typically dependency paths, has consistently played an important role in frame identification (Das et al., 2014; Peng et al., 2018).

Our work is motivated by the rich lexicographic information about frames and lexical units provided by the FrameNet database, which has not been fully utilized for the frame identification task. Recent advances in large pre-trained transformer models (Devlin et al., 2019) have demonstrated the ability to capture semantic meaning in dictionary definitions for the related problem of word sense disambiguation (Huang et al., 2019; Blevins and Zettlemoyer, 2020).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2429–2434
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

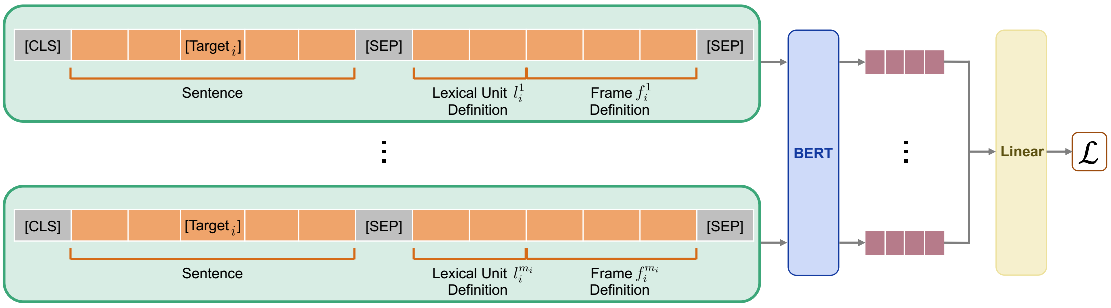


Figure 1: Overview of the FIDO architecture. Each green block represents a different candidate pair (lexical unit, frame) for the same Target$_i$.

Our model uses the definitions of frames and lexical units in FrameNet as a source of knowledge to help assess the semantic coherence between the target word and candidate frames. Specifically, we utilize the contextual embeddings produced by the BERT (Devlin et al., 2019) model to determine if a candidate lexical unit and frame express the same meaning as the target word in the given context. Our model achieves state-of-the-art performance on two FrameNet datasets and a FrameNet-annotated dataset based on Yahoo! Answers. Our code is open-source and available online[1].

2 Related Work

There has been considerable work on the frame identification problem with respect to FrameNet, especially since the SemEval 2007 shared task (Baker et al., 2007). Johansson and Nugues (2007) used a SVM classifier to disambiguate frames with hand-crafted features. Das et al. (2010) applied feature-based discriminative probabilistic (log-linear) models for frame identification. Hermann et al. (2014) presented a method using distributed representations of predicates and their syntactic context by mapping input representations and frame representations to a common latent space using the WSABIE algorithm (Weston et al., 2011). Hartmann et al. (2017) built a simplified model based on Hermann et al. (2014) and achieved comparable results. They also released a new FrameNet-annotated test set based on user-generated web text from Yahoo! Answers. Yang and Mitchell (2017) integrated a bidirectional LSTM neural network and a relational network to jointly decode frames.

More recently, Botschen et al. (2018) brought in multimodal representations grounded in images to improve frame identification. Peng et al. (2018) proposed a joint inference formulation that learns semantic parsers from multiple datasets.

In contrast to the previous models, our model does not rely on syntactic features. We assess semantic coherence directly from the input sentence and definitions in FrameNet.

Another line of related work is learning embeddings from dictionary definitions. It has been shown that neural networks can extract semantic information from dictionary definitions (Kumar et al., 2019; Bosc and Vincent, 2018). Recent work in word sense disambiguation (Huang et al., 2019; Blevins and Zettlemoyer, 2020) has demonstrated that providing pre-trained language models with sense definitions (glosses) can be effective. Yong and Torrent (2020) also used the sense definitions of lexical units for their research on frame induction. Our model adopts a similar architecture as Huang et al. (2019), but we focus on the frame identification task and we explore the use of both lexical unit and frame definitions for this task.

3 Method

Given a sentence and a target word or phrase, the frame identification task assigns the most relevant frame to the target according to the sentence context. Figure 1 shows the framework of our model called FIDO (**F**rame **I**dentification with **D**efiniti**O**ns). Our system takes the sentence and the definitions of associated lexical units (senses) and their frames as input to the BERT model, as indicated by the green blocks. Each green block represents the target word in the sentence, one of its senses, and that sense's associated frame in FrameNet. Then we use the output vectors to produce a probability distribution over all of the candidate frames. We select the frame with the maximum probability as the answer.

---

[1] `https://github.com/tyjiangU/fido`

3.1 Notation

We denote the ith example ($i = 1, 2, ..., n$) consisting of a sentence and designated target word or phrase as $\langle s, t \rangle_i$, its correct frame as f_i^*, the set of lexical units associated with the target as $l_i^1, l_i^2, ..., l_i^{m_i}$, and their corresponding frames as $f_i^1, f_i^2, ..., f_i^{m_i}$ (f_i^* is among them). We seek to estimate the probability of the jth frame being the correct frame by:

$$\Pr(f_i^j | \langle s, t \rangle_i) = \frac{\exp(g(\langle s, t \rangle_i, f_i^j))}{\sum_{k=1}^{m_i} \exp(g(\langle s, t \rangle_i, f_i^k))} \quad (1)$$

where $g(\cdot)$ is a function produced by our model for scoring the assignment of a frame to the sentence and target. We use negative log likelihood as our loss function:

$$\mathcal{L} = -\sum_{i=1}^{n} \log \Pr(f_i^* | \langle s, t \rangle_i) \quad (2)$$

where n is the total number of training examples.

3.2 Modeling

FrameNet provides unique definitions for each lexical unit (LU) and frame. A LU is a pairing of a word lemma and a meaning (sense). Determining the correct LU (sense) uniquely determines the correct frame because each sense of a polysemous word is linked to a different frame. For example, the word *cut* can trigger different frames depending on its meaning (the definition sentences follow the bold lexical unit or frame names), as shown in Table 1.

Lexical Unit	Associated Frame
cut.n: the way or style in which a garment or the hair is cut	**Hair_configuration**: temporary or permanent styles and configurations of hair
cut.v: divide into pieces with a knife or other sharp implement	**Cutting**: an agent cuts an item into pieces using an instrument

Table 1: Examples of lexical units and their associated frames.

We use the BERT (Devlin et al., 2019) model as the base of our architecture to produce the function $g(\cdot)$ as described in Eq (1). For each target, first we extract LUs from FrameNet that have the same lemma and their corresponding frames to form a set of candidate (LU, Frame) pairs. Our goal is to predict whether the target in the sentence has the same meaning as the definitions of a candidate LU and its associated frame.

As input to the BERT model, we use the sentence as the first sequence and concatenate a LU definition and frame definition as the second sequence. Each definition starts with the LU name or frame name and a colon, followed by the definition description.

Instead of using the output vector of the `[CLS]` token as is typical, we use the last hidden vector of the target word as output (if there is more than one token, we only use the first one). By passing the output vector through a linear layer, we then get a score for assigning a candidate frame to the sentence and target. Finally the scores for all candidate frames are passed through the softmax function to get the probabilities in Eq (1).

4 Experiments

4.1 Datasets

FrameNet: To compare FIDO with previous systems, we evaluate our model on FrameNet (FN) 1.5 using the same train/dev/test data split as Das et al. (2014). We also evaluate our model on FN 1.7 which has been available since 2016 and contains nearly 20% more gold annotated data than FN 1.5. We use the same data split as Swayamdipta et al. (2017) for FN 1.7. Table 2 shows the number of examples in each split.

	FN 1.5	FN 1.7	YAGS
Train	15,017	19,391	-
Dev	4,463	2,272	1,000
Test	4,457	6,714	2,093

Table 2: Dataset sizes.

YAGS: YAGS (Hartmann et al., 2017) is a FrameNet-annotated test set based on question answering data from Yahoo! Answers[2], a community-driven question-and-answer forum. The annotations are based on FN 1.5. We train on FN 1.5 and evaluate on the YAGS test set to compare results with Hartmann et al. (2017).

---

[2] https://webscope.sandbox.yahoo.com/

Model	Accuracy
Hermann et al. (2014)	88.4
Hartmann et al. (2017)	87.6
Yang and Mitchell (2017)	88.2
Open-SESAME (2017)	86.9
Botschen et al. (2018)	88.8
Peng et al. (2018)	90.0
FIDO	**91.3**

Table 3: Accuracy on FN 1.5.

Dataset	Model	Accuracy
FN 1.7	Peng et al. (2018)	89.1
	FIDO	**92.1**
YAGS	Hartmann et al. (2017)	62.5
	FIDO	**70.5**

Table 4: Accuracy on FN 1.7 and YAGS.

4.2 Training Details

We use the pre-trained uncased BERT$_{\text{BASE}}$ model with the same settings as Devlin et al. (2019) and fine-tune on our training data. We set the max sequence length as 300, batch size as 16, learning rate started at 2e-5, and train for 5 epochs. All reported results are averaged over 3 runs with random seeds.

4.3 Results

Table 3 compares our model with previous methods on the FN 1.5 dataset. Hermann et al. (2014), Hartmann et al. (2017), and Open-SESAME (Swayamdipta et al., 2017) use distributed representations and syntactic features with neural networks. Botschen et al. (2018) extends Hartmann et al. (2017) with visual embeddings. Yang and Mitchell (2017) integrates a sequential and relational network for joint learning. Peng et al. (2018) has achieved the best prior results on frame identification using a multitask approach to learn semantic parsers from disjoint corpora. It is worth noting that besides the FN 1.5 training set, they also use 153,952 exemplar sentences for training, which is more than 10 times the size of our training data. FIDO achieves better performance than all of the prior systems.

Table 4 shows our results compared to Peng et al. (2018) on FN 1.7. FIDO achieves a 3.0% absolute accuracy gain on this data set. The YAGS data set (Hartmann et al., 2017) contains *unknown* targets that do not have related LUs in FN 1.5 and also *unlinked* targets (i.e., the provided gold frame does not belong to the set of frames associated with this target in FN). Our model is not able to make a correct prediction for these cases based on its design. There are 122 unknown or unlinked targets in the test set, on which our model will get a zero score. Despite this limitation, our model still outperforms Hartmann et al. (2017), which demonstrates its ability to generalize across text genres.

Model	FN 1.5	FN 1.7
FIDO	**91.3**	**92.1**
FIDO (FRdef only)	90.1	91.1
FIDO (LUdef only)	88.9	90.7
FIDO (NO def)	80.3	79.4
FIDO (CLS)	89.3	90.5

Table 5: Ablation study on FN 1.5 and FN 1.7.

4.4 Analysis

We performed an ablation study to assess the contributions of each part of our model. In Table 5, the first row shows the results for our complete FIDO model. Rows 2-3 show results when using only the definitions of frames (FRdef only) or LUs (LUdef only). We see that the frame definitions contribute the most to performance. Using the LU definitions alone on FN 1.7 also achieves quite good results. But combining both definitions together yields better results than either one alone.

In order to tease apart the impact of the definitions from the impact of BERT, we did an experiment replacing each definition simply with the name of the frame or LU. These results appear in the FIDO (NO def) row. Removing the definitions results in a large performance drop. The definitions clearly play a major role.

In the bottom row, we show the results of experiments using the output vector of the `[CLS]` token (all other settings the same), which did not perform as well as using the target token. This is not surprising as `[CLS]` aggregates the entire sequence representation rather than focusing on the target.

Previous work also reported accuracy on ambiguous cases (i.e., when the target word is associated with multiple frames), which more directly shows the model's ability to disambiguate frames. However, the set of ambiguous targets is different across papers. To avoid comparing apples and oranges,

Dataset	Model	Amb1	Amb2
FN 1.5	Peng et al. (2018)	78.0	-
	FIDO	**81.0**	83.6
FN 1.7	Peng et al. (2018)	77.5	-
	FIDO	**83.8**	85.9

Table 6: Accuracy on ambiguous cases.

we report accuracy on two different sets of ambiguous targets. In Table 6, the **Amb1** column follows Peng et al. (2018), which uses the gold LU's part-of-speech (POS) tag to form the candidate frame list. In this setting, if a target has just one sense when its POS is known, it is not considered to be ambiguous. Our model outperforms Peng et al. (2018) on both FN 1.5 and FN 1.7 datasets. The **Amb2** column shows the accuracy of ambiguous targets using only the lemma of the target (i.e., not relying on gold POS tags). We encourage future work to articulate which setting is used.

We also analyzed whether unseen frames and unseen targets were a major source of errors for our model. On FN 1.7, our FIDO model achieved 92.1% accuracy, so it mislabeled 7.9% of the test cases. We found that 1.4% of the test cases were mislabeled and had an unseen frame (i.e., the gold frame was not seen with the target in the training data), and 0.52% of the test cases were mislabeled and had an unseen target (i.e., the target was not seen in the training data). Therefore only about 1/4 of the FIDO errors were due to unseen frames and unseen targets. We conclude that even for frames and targets that appear in the training data, there is still substantial room for improvement on this task.

5 Conclusion

We tackled the frame identification problem by assessing the semantic coherence between the meaning of a target word in a sentence, and a candidate frame. Our model exploits the frame and lexical unit definitions provided by FrameNet and a pre-trained transformer model to generate semantic representations. The experiments show that this model achieves better performance than previous systems on two versions of FrameNet data and the YAGS dataset. Our work has demonstrated that a relatively simple architecture that brings together pre-trained language models with frame and sense definitions can produce a highly effective system for frame identification.

Acknowledgments

We thank Tao Li and Yichu Zhou for their helpful comments on the work. We also thank the anonymous reviewers for their suggestions.

References

Collin Baker, Michael Ellsworth, and Katrin Erk. 2007. SemEval-2007 task 19: Frame semantic structure extraction. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval 2007)*, pages 99–104, Prague, Czech Republic. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING 1998), Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 1006–1017, Online. Association for Computational Linguistics.

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 1522–1532, Brussels, Belgium. Association for Computational Linguistics.

Teresa Botschen, Iryna Gurevych, Jan-Christoph Klie, Hatem Mousselly-Sergieh, and Stefan Roth. 2018. Multimodal frame identification with multilingual evaluation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018)*, pages 1481–1491, New Orleans, Louisiana. Association for Computational Linguistics.

Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-semantic parsing. *Computational linguistics*, 40(1):9–56.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2010)*, pages 948–956, Los Angeles, California. Association for Computational Linguistics.

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

Charles J Fillmore. 1976. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280 (1), pages 20–32.

Charles J Fillmore. 1982. Frame semantics. In *Linguistics in the Morning Calm, The Linguistic Society of Korea (ed.)*, pages 111–137. Seoul: Hanshin Publishing Company.

Silvana Hartmann, Ilia Kuznetsov, Teresa Martin, and Iryna Gurevych. 2017. Out-of-domain FrameNet semantic role labeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 471–482, Valencia, Spain. Association for Computational Linguistics.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 1448–1458, Baltimore, Maryland. Association for Computational Linguistics.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 3509–3514, Hong Kong, China. Association for Computational Linguistics.

Richard Johansson and Pierre Nugues. 2007. LTH: Semantic structure extraction using nonprojective dependency trees. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval 2007)*, pages 227–230, Prague, Czech Republic. Association for Computational Linguistics.

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha Talukdar. 2019. Zero-shot word sense disambiguation using sense definition embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 5670–5681, Florence, Italy. Association for Computational Linguistics.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018)*, pages 1492–1502, New Orleans, Louisiana. Association for Computational Linguistics.

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental rnns and a syntactic scaffold. *arXiv preprint arXiv:1706.09528*.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI 2011)*, page 2764–2770. AAAI Press.

Bishan Yang and Tom Mitchell. 2017. A joint sequential and relational model for frame-semantic parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1247–1256, Copenhagen, Denmark. Association for Computational Linguistics.

Zheng Xin Yong and Tiago Timponi Torrent. 2020. Semi-supervised deep embedded clustering with anomaly detection for semantic frame induction. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*, pages 3509–3519, Marseille, France. European Language Resources Association.

ADePT: Auto-encoder based Differentially Private Text Transformation

Satyapriya Krishna
Amazon Alexa
`satyapk@amazon.com`

Rahul Gupta
Amazon Alexa
`gupra@amazon.com`

Christophe Dupuy
Amazon Alexa
`dupuychr@amazon.com`

Abstract

Privacy is an important concern when building statistical models on data containing personal information. Differential privacy offers a strong definition of privacy and can be used to solve several privacy concerns (Dwork et al., 2014). Multiple solutions have been proposed for the differentially-private transformation of datasets containing sensitive information. However, such transformation algorithms offer poor utility in Natural Language Processing (NLP) tasks due to noise added in the process. In this paper, we address this issue by providing a utility-preserving differentially private text transformation algorithm using auto-encoders. Our algorithm transforms text to offer robustness against attacks and produces transformations with high semantic quality that perform well on downstream NLP tasks. We prove the theoretical privacy guarantee of our algorithm and assess its privacy leakage under Membership Inference Attacks (MIA) (Shokri et al., 2017) on models trained with transformed data. Our results show that the proposed model performs better against MIA attacks while offering lower to no degradation in the utility of the underlying transformation process compared to existing baselines.

1 Introduction

Differentially Private (DP) mechanisms provide robustness against privacy attacks and offer practical solutions for transforming and releasing datasets without compromising privacy (Dwork et al., 2009). A typical downstream task may involve training a machine learning model with data transformed from a differentially private mechanism. However, while the DP mechanism offers privacy, it can adversely impact the utility of the trained model (Li and Li, 2009). Specifically, in the case of text datasets (e.g., those used in Natural Language Understanding(NLU) tasks), if the DP transformation impacts the syntactic structure of the sentence or does not factor in the target NLU label (e.g. intent of the sentence in an intent classification task), the loss in utility can render the use of processed data impractical. We address this problem in the paper and introduce ADePT - an Auto encoder based Differentially Private Text transformation mechanism that process text data while reducing the impact on the utility of the dataset.

The ADePT mechanism relies on text-based auto-encoders (e.g. LSTM based sequence-to-sequence models) for text transformation. An auto-encoder first transforms a given text input into some latent representation, followed by text generation (transformation) via the decoder. In this paper, we prove that the application of clipping and noising operation on the latent sentence representations returned by the encoder followed by text generation by the decoder is a DP mechanism. We use ADePT to transform text datasets relevant to the Intent Classification (IC) task, where we predict intent of input sentence (e.g. 'BuyTicketIntent' intent prediction for the sentence 'buy me a ticket to Seattle'). While one can transform the text in datasets and retain original intent labels to train the intent classifier, it is not guaranteed that the transformed text would correspond to the original intent post transformation, which can adversely impact the trained IC's utility. To mitigate this problem, we append the intent labels to the rest of tokens while training the auto-encoder as well as for transforming text with the trained auto-encoder. For instance, *@BuyTicketIntent buy me a ticket to Seattle* is used as the input sample to train the autoencoder where *@BuyTicketIntent* is the intent annotation for *buy me a ticket to Seattle*. Similarly, the intent label is regenerated along with the rest of the tokens after transformation, which is then used for IC training with the regenerated intent as the label of the regenerated sentence. In addition to this, we argue that

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2435–2439
April 19 - 23, 2021.

data regeneration via decoder maintains the syntactic structure of the sentence since the decoder generates tokens auto-regressively, factoring in the previously generated tokens. We hypothesize that these properties make ADePT a utility preserving DP mechanism and demonstrate the superiority of the algorithm against an existing baseline (Feyisetan et al., 2019).

2 Related Work

Differentially private data transformation and generation: Researchers have proposed several methods for DP data transformation using individual ranking micro-aggregation (Sánchez et al., 2016), random projection (Xu et al., 2017), and kernel mean embeddings (Balog et al., 2018). Alternatively, models such as differentially private Generative Adversarial Networks (Xie et al., 2018) and differentially private autoencoder-based generative model (Chen et al., 2018) focus on training data generators that guarantee that the data generation mechanism is DP. While DP data transformation and generation has shown great success for structured data (e.g. numeric tables, histograms), the same for unstructured data (e.g. text) is more challenging. Beigi et al. (2019) propose an algorithm that learns numeric text representations that offer guarantees of differential privacy. However, arguably it may be more desirable to release transformed text as opposed to latent representations. Feyisetan et al. (2019) proposes DP mechanism to transform text data that constructs a hierarchical representation given a sentence to identify *private phrases* in the input sentence. Each word in the private phrase is then randomly replaced by neighboring word in a word embedding space. We use the work by Feyisetan et al. (2019) as baseline in our work since it also focusses on obtaining text transformed text with a DP mechanism.

Membership Inference Attacks: While the (ϵ, δ) bounds provide theoretical quantification of a mechanism's privacy (Dwork et al., 2014), recently Membership Inference Attack (MIA) success rates have emerged as practical quantification of privacy preservation (Shokri et al., 2017). In this work we use the setup suggested by Shokri et al. (2017) as a method to quantify privacy for models trained on transformed data. Given a trained machine learning model and its confidence score on a datapoint, MIA infers whether the datapoint was part of the model's training data. In order to conduct MIA, an attacker trains a *shadow model* that he/she expects to mimic the *target model* under attack. Once trained, the *shadow model's* confidence scores on the datapoints *members* of its training set and other *non-member* datapoints are used to train the binary attack model. Given a datapoint, the attacker then extracts a similar vector of confidence scores from the *target model* and uses the *attack model* to make a *member*/*non-member* prediction.

3 ADePT: Auto-encoder based DiffEerentially Private Text transformation

Consider an utterance $\mathbf{u}$ drawn from a dataset $\mathcal{D}$. Furthermore, consider an auto-encoder model that takes input a sentence $\mathbf{u}$ and outputs another sentence $\mathbf{v}$. A vanilla auto-encoder model consists of an encoder that returns a vector representation $\mathbf{r} = \mathsf{Enc}(\mathbf{u})$ for the input $\mathbf{u}$, which is then passed onto the decoder that constructs an output $\mathbf{v} = \mathsf{Dec}(\mathbf{r})$. We define ADePT as a randomized algorithm $\mathcal{A}$, that given an utterance $\mathbf{u}$, generates $\mathbf{v}$ as shown in equation 2. η is a vector sampled from either a Laplacian or a Gaussian distribution (with 0 mean and a pre-defined variance).

$$\mathbf{v} = \mathsf{Dec}(\mathbf{r}') \tag{1}$$

$$\text{Where } \mathbf{r}' = \mathsf{Enc}(\mathbf{u}) \cdot \min\left(1, \frac{C}{||\mathsf{Enc}(\mathbf{u})||_2}\right) + \eta \tag{2}$$

3.1 Proof that ADePT is differentially private

Given that ADePT conducts a transformation from $\mathbf{u} \rightarrow \mathbf{r}' \rightarrow \mathbf{v}$, we first show that it is sufficient to prove that the transformation from $\mathbf{u} \rightarrow \mathbf{r}'$ is DP for ADePT to be DP. Thereafter, we prove that the transformation $\mathbf{u} \rightarrow \mathbf{r}'$ is DP.

Lemma 1. *The transformation* $\mathbf{u} \rightarrow \mathbf{v}$ *will be at least* (ϵ, δ) *differentially private, if the algorithm that transforms* $\mathbf{u}$ *to* $\mathbf{r}'$ *is* (ϵ, δ) *DP.*

Proof. This is true based on proposition 2.1 on post-processing in Dwork et al. (2014). ☐

Theorem 1. *If* η *is a multidimensional noise, such that each element* $\eta_i \in \eta$ *is independently drawn from a distribution shown in equation 3, then the transformation from* $\mathbf{u} \rightarrow \mathbf{v}'$ *is* $(\epsilon, 0)$ *DP.*

$$Lap(\eta_i) \sim \frac{\epsilon}{4C} \exp(-\frac{\epsilon|v_i|}{2C}) \tag{3}$$

Proof. We refer the reader to the proof in Dwork et al. (2014), Theorem 3.6. The function $f(x)$ used in the Theorem in Dwork et al. (2014) is equivalent to the encoder output with clipping. The l1-sensitivity of this function (please refer to definition 3.1 in Dwork et al. (2014)) is $2C$ since maximum L1 norm difference between two points in a hyper-sphere of radius C is $2C$. Replacing Δf in Theorem 3.6 in Dwork et al. (2014) by $2C$, we obtain the that the transformation is $(\epsilon, 0)$ DP. □

Akin to the proof with a Laplacian noise, we can also borrow the proof in Appendix A in Dwork et al. (2014) to show that ADePT would also be DP if η was a Gaussian noise.

4 Experimental setup

We perform an intent classification task in our experiments and quantify impacts on accuracy and privacy metrics after data transformation via the ADePT mechanism. While the intent classification accuracy quantifies the utility of the transformed dataset, we evaluate success of MIA against the IC model to quantify privacy. Below, we describe the datasets, auto-encoder and IC model training and the MIA setup used in our experiments.

4.1 Datasets

We use ATIS (Dahl et al., 1994) and SNIPS (Coucke et al., 2018) for training IC models on the respective datasets. The ATIS dataset consists of ∼5.5k data samples, while the SNIPS dataset consists of ∼14.5k data samples. We used a 50:50 split for training and evaluation sets. Apart from offering a larger accuracy evaluate test set, a 50:50 split also ensures that we have a balanced training and evaluation sets for MIA, as discussed in Section 4.5.

4.2 Training the auto-encoder model

Given utterances $\mathbf{u}$ in the training partition of the datasets of interest, we train an auto-encoder model to reconstruct the input utterance $\mathbf{u}$ via the decoder Dec. In our case, the auto-encoder is a sequence to sequence model, where both encoder and decoders are uni-directional LSTM models. We train the auto-encoder on the training portions of the ATIS and SNIPS datasets, with an objective to reconstruct the input sentence through the *latent* representation. Note that during training, we apply clipping to ensure that the latent representation are encouraged to reside within a hyper-sphere of radius C, no noise is added to the latent representation. Clipping and noising operations are applied during the final transformation after the auto-encoder is trained, as discussed in section 4.3.

4.2.1 Making ADePT utility preserving

In the proof above, we show that ADePT is DP algorithm that transforms input utterances $\mathbf{u}$ to $\mathbf{v}$. For the purposes of training an intent classifier, a naive scheme can assume that the intent label applied to the utterance $\mathbf{u}$ is also applicable to $\mathbf{v}$. However, this assumption may not always be true as the transformation may render utterance $\mathbf{v}$ to carry a different intent label than $\mathbf{u}$. In order to encourage the transformed utterances $\mathbf{v}$ to conform to the intent label for utterance $\mathbf{u}$, and also obtain the correct intent label in cases where the transformation may lead $\mathbf{v}$ to belong to a different intent, we tweak the auto-encoder model to also ingest the intent label. We train annotation aware auto-encoder models with inputs/outputs as utterances and the corresponding intent. The intent label is appended to the beginning of each utterance (demarcated with a special character to help distinguish the intent names with utterance tokens) during the auto-encoder training.

4.3 Data transformation

Once the auto-encoder model is trained, we apply the transformation again on the training portions of ATIS and SNIPS datasets. During the transformation, the intent token is appended with the rest of the utterance and an output in a similar format is expected.

4.4 Intent classifier training

The ADePT transformation yields the altered sentence, along with an intent. We transform the training portion of ATIS and SNIPS datasets through the autoencoder and use the altered sentences along with the reproduced intent for training an intent classifier. Our IC architecture is inspired from Ma and Hovy (2016) and consists of three blocks: (i) an embedding block consisting of word and character embeddings, (ii) a block consisting of bi-directional LSTM layers and, (iii) a fully connected network operation on a max-pool of LSTM layer outputs for intent classification.

4.5 Privacy evaluation using MIA

We train the attack model on confidence scores returned by a shadow IC model trained similarly

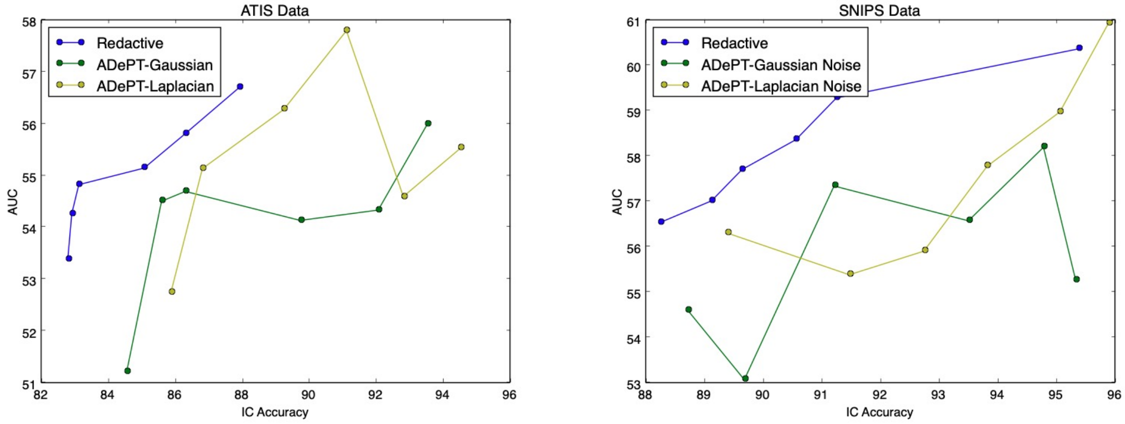


Figure 1: Privacy and accuracy metrics using baseline and ADePT mechanisms on the ATIS and SNIPS datasets. Baseline mechanism transforms datasets with Laplacian noise with variance values $\in (1, 6, 9, 15, 28, 100)$. ADePT transforms datasets with Gaussian and Laplacian noises with variances $\in (0.25, 0.5, 0.6, 0.75, 0.85, 1)$. The variance scales are different between the two mechanism due to inherent difference in their construct.

Original	**Baseline**	**ADePT**
what are the flights on **january first 1992 from boston to san francisco**	what are the flights on **february inhales 1923 from boston to san mostrar**	what are the flights on **thursday going from dallas to san francisco**
show all **flights boston to any time**	show all **5-minutes distinctions from massachusetts to tempat chiefs**	show all **flights flights flights boston to any time**

Table 1: Example of a good and a corrupted output from ADePT

as the target IC model. We extract scores for the top five intents returned by the shadow IC model on the *member* and *non-member* sentences used to train the shadow IC model. The attack model is a binary logistic regression model, trained on the extracted IC scores from ‘member’ and ’non-member’ sentences.

During the attack, top 5 intent scores from the target IC model are fed to the logistic regression model to make a prediction whether the corresponding scores belong to the target model’s *member* or *non-member* data. While the *member* sentences are sourced from the training set, we borrow *non-member* sentences from the test set used to evaluate the model accuracy (note that their counts are balanced as we use a 50:50 split). We use the Area Under the ROC curve (AUC) to evaluate the success of the attack model and a higher AUC implies worse privacy metric.

5 Experimental Results

We conduct ADePT transformation using both Laplacian and Gaussian noises, with different variance values. The baseline mechanism also uses a Laplacian noise to sample words replacements for the private words. Figure 1 show the MIA success rates and IC accuracies obtained on ATIS and SNIPS data respectively. Note that the algorithm with a lower AUC and a higher IC accuracy is desirable. We observe that as we sweep the noise parameters for the ADePT and Redactive mechanisms, we generally obtain lower AUC with a higher IC accuracy for the former. Additionally ADePT mechanism with a Gaussian noise performs the best. This empirical observation supports our hypothesis that factoring in the intent label during ADePT based transformation helps providing better utility.

However, we also note that the privacy-utility trade-off in ADePT can be non-monotonic. We noticed that the sentence transformation using encoders is sensitive to noise value added to encoded representation $\mathsf{Enc}(\mathbf{u})$. The clipping and noise additional has potential to change the entire sentence, unlike the baseline, where the public phrase in the utterance remains unaltered and only the private phrases in the utterances are subject to alteration. We show two examples of sentence transformation using the baseline and Gaussian ADePT mechanism in Table 1. In particular, the decoder tends to repeat the same word multiple times for corrupted outputs which can be corrected with constrained decoding.

6 Conclusions

We propose ADePT - an auto-encoder based DP algorithm in this paper. We theoretically prove that the mechanism is DP and demonstrate that it offers a better privacy utility trade-off compared to a baseline that relies on detecting the transforming public phrases in a sentence. In the future, we will extend ADePT to transforming datasets with sequence level tags (for instance, in named entity recognition tasks) and also use non-autoregressive decoders (e.g. transformers). We will also extend the mechanism to other modalities (e.g. Image) using auto-encoder models in the corresponding domains.

References

Matej Balog, Ilya Tolstikhin, and Bernhard Schölkopf. 2018. Differentially private database release via kernel mean embeddings. In *International Conference on Machine Learning*, pages 414–422. PMLR.

Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2019. I am not what i write: Privacy preserving text representation learning. *arXiv preprint arXiv:1907.03189*.

Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaarfar, and Haojin Zhu. 2018. Differentially private data generative models. *arXiv preprint arXiv:1812.02274*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Deborah A Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, et al. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *HLT Workshop*.

Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. 2009. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 381–390.

Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

O. Feyisetan, T. Diethe, and T. Drake. 2019. Leveraging hierarchical representations for preserving privacy and utility in text. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 210–219.

Tiancheng Li and Ninghui Li. 2009. On the tradeoff between privacy and utility in data publishing. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 517–526.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074. Association for Computational Linguistics.

David Sánchez, Josep Domingo-Ferrer, Sergio Martínez, and Jordi Soria-Comas. 2016. Utility-preserving differentially private data releases via individual ranking microaggregation. *Information Fusion*, 30:1–14.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. 2018. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*.

Chugui Xu, Ju Ren, Yaoxue Zhang, Zhan Qin, and Kui Ren. 2017. Dppro: Differentially private high-dimensional data release via random projection. *IEEE Transactions on Information Forensics and Security*, 12(12):3081–3093.

Conceptual Grounding Constraints for Truly Robust Biomedical Name Representations

Pieter Fivez
CLiPS Research Centre
University of Antwerp
pieter.fivez@uantwerpen.be

Simon Šuster
Faculty of Engineering and Information Technology
University of Melbourne
simon.suster@unimelb.edu.au

Walter Daelemans
CLiPS Research Centre
University of Antwerp
walter.daelemans@uantwerpen.be

Abstract

Effective representation of biomedical names for downstream NLP tasks requires the encoding of both lexical as well as domain-specific semantic information. Ideally, the synonymy and semantic relatedness of names should be consistently reflected by their closeness in an embedding space. To achieve such robustness, prior research has considered multi-task objectives when training neural encoders. In this paper, we take a next step towards truly robust representations, which capture more domain-specific semantics while remaining universally applicable across different biomedical corpora and domains. To this end, we use conceptual grounding constraints which more effectively align encoded names to pretrained embeddings of their concept identifiers. These constraints are effective even when using a Deep Averaging Network, a simple feedforward encoding architecture that allows for scaling to large corpora while remaining sufficiently expressive. We empirically validate our approach using multiple tasks and benchmarks, which assess both literal synonymy as well as more general semantic relatedness. Our code is open-source and available at `www.github.com/clips/conceptualgrounding`.

1 Introduction

Biomedical and clinical free-text contain mentions of biomedical terms which can provide valuable information for text mining applications. Such textual mentions, as well as their corresponding reference names in biomedical ontologies, can often be expressed in various synonymous surface forms (e.g. *pleuritic pain* vs. *pain breathing*), which is challenging for downstream applications. Effective dense representation of these biomedical names has been mainly investigated through the normalization task of disorder linking, which consists of matching disease mentions to reference terms of concept identifiers in ontologies (e.g. matching the mention *myocardial depression* to the reference term *Myocardial Dysfunction)* (Leaman et al., 2015). While past research has gradually shifted its focus from lexical representations (Leaman et al., 2013; D'Souza and Ng, 2015) to dense distributed representations (Limsopatham and Collier, 2016; Li et al., 2017; Phan et al., 2019; Sung et al., 2020), encoders are still typically optimized towards normalization tasks, which are focused on resolving word-level analogies between synonymous biomedical names.

ICD-10	SNOMED-CT
F60.1	C0564504 *schizoid fantasy* *schizoid fantasy - mental defense mechanism*
	C0338969 *introverted personality disorder* *introverted personality*
	C0036339 *schizoid personality disorder* *unspecified schizoid personality disorder*

Table 1: Example of SNOMED-to-ICD-10 mappings. The synonym sets for the SNOMED-CT concepts *C0564504*, *C0338969*, and *C0036339*, are fused into one large set of semantically related names for the ICD-10 code *F60.1*.

Recent research has focused more explicitly on encoding domain-specific biomedical semantics by training biomedical name representations that are *robust*, i.e., reflecting the synonymy and semantic relatedness of names by their closeness in the embedding space, preferably in a consistent way

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2440–2450
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

that generalizes across different biomedical subdomains and corpora. To date, the most effective approaches have applied some form of *conceptual grounding*: minimizing the distance between on the one hand representations of names, and on the other hand pretrained embeddings of their concept identifiers. These concept embeddings are supposed to reflect domain-specific semantics, and are constructed using a variety of different techniques, including distributional similarity of graph relations and distributional similarity of textual occurrences in large-scale free-text, as well as combinations thereof (Kartsaklis et al., 2018; Phan et al., 2019).

While knowledge graph embeddings of biomedical concepts can encode a variety of semantic relations, Kartsaklis et al. (2018) show that such graph embeddings need to incorporate textual features to make them effective targets for conceptual grounding. Such features help to translate textual representations of names to the topology of the concept embedding space, which otherwise reflects only ontological information. In other words, concept embeddings are mostly useful targets for grounding to the extent that name representations can be efficiently mapped to them by the encoder architecture. This raises the question whether we can increase the effectiveness of conceptual grounding by better aligning the topology of the created name embedding space and the pretrained concept embedding space. In this paper, we investigate how to maximally exploit low-cost concept embeddings, which can be constructed using only pretrained word embeddings and sets of biomedical synonyms or semantically related names.

To this end, we enrich a siamese neural network encoder for biomedical names with 2 novel constraints which are meant to effectively map encoded names to pretrained concept embeddings. The first constraint, which we call the *linear constraint*, applies canonical correlation analysis (CCA) to pretrained embeddings of names and their concepts to project them into a space which improves their linear mapping. These transformed embeddings are then used as input representations for the neural encoder. The second constraint adds a training objective which we call *prototypical grounding*: minimizing the distance between a pretrained concept embedding and the average of all the encoded names belonging to that concept. This average is an approximation of the prototypical representation of a concept in the name embedding space.

While the linear constraint involves a simple preprocessing step, the prototypical grounding constraint can be computationally expensive for large-scale corpora. Therefore, we use a simple Deep Averaging Network (DAN) (Iyyer et al., 2015) as encoder to prove the effectiveness and scalability of our approach, even for a neural architecture that has no access to word order like LSTMs have or cannot apply attention over specific word combinations like Transformers can. We train and evaluate our encoder on different categorizations of biomedical names. For instance, Table 1 shows how concepts from the SNOMED-CT ontology capture literal synonymy, while these concepts can also be grouped into the ICD-10 coding system which reflects more general semantic relatedness. Our experimental results show that our approach is effective for both types of categorizations, as well as for various ontologies and benchmarks.

2 Related work

Biomedical name encoders A variety of neural architectures have been proposed for encoding biomedical names. Kartsaklis et al. (2018) use a multi-sense LSTM with attention over different word senses. This attention is conditioned on the context of the biomedical name. Phan et al. (2019) include a character-level Bidirectional LSTM in a word-level Bidirectional LSTM which extracts a fixed-size representation using max pooling over all dimensions, followed by a linear transformation. Sung et al. (2020) finetunes pretrained context-sensitive BioBERT (Lee et al., 2019) representations and uses them in tandem with lexical TF-IDF representations. While past research has explicitly investigated the role of various training objectives, even jointly in multi-task training regimes, the specific impact of encoder architectures has not received much attention or comparison.

Averaging networks Research on sentence embeddings and paraphrasing has consistently found that simple encoding procedures such as averaging of word embeddings can rival or even outperform complex neural architectures on tasks for which those are finetuned (Wieting et al., 2016; Shen et al., 2018; Wieting and Kiela, 2019). Moreover, research on Deep Averaging Networks (Iyyer et al., 2015) has found that feedforward neural networks that use averaged word embeddings as input can be tuned to textual classification tasks such as sentiment analysis if the network is sufficiently large and/or deep. This way, small differences in the input can be magnified by the network where relevant.

Prototypical networks While successful approaches to few-shot learning such as Matching Networks (Vinyals et al., 2016) optimize representation models on the level of single instances, follow-up work has shown the benefits of simultaneously learning class representations using those same models. For instance, prototypical networks (Snell et al., 2017) train a neural encoder with objectives that involve class prototypes, which are created by averaging the encodings of all instances that belong to a single class. In this paper, we include a training objective for our encoder which forces synonymous or semantically related biomedical names to form class prototypes that approximate the pretrained embedding of their concept identifier.

3 Encoding model

3.1 Encoder architecture

Our encoder is a Deep Averaging Network (DAN) (Iyyer et al., 2015) which extracts a fixed-size representation for an input name *n*:

$$\begin{gathered} u_n = \frac{1}{|N_t|} \sum_{t \in N_t} u_t \\ f(n) = enc(u_n) \end{gathered} \tag{1}$$

where N_t is the bag of tokens from a name, u_t is a pretrained word embedding of a token, u_n is a name embedding created by averaging all the pretrained word embeddings of all tokens, and enc is a feedforward neural network with Rectified Linear Unit (ReLU) as non-linear activation function. As pretrained word embeddings we use 300-dimensional fastText (Bojanowski et al., 2017) representations which we train on 76M sentences of preprocessed MEDLINE articles released by Hakala et al. (2016). This fastText model also allows for constructing word embeddings for out-of-vocabulary tokens by composing character n-gram embeddings.

3.2 Training objectives

Our training objectives optimize the mapping between an encoded name $f(n)$ and the pretrained embedding of its concept u_p. While in principle any type of pretrained concept embeddings could be used, our experiments use concept embeddings which are simply the average of all pretrained name embeddings belonging to the concept:

$$u_p = \frac{1}{|C_n|} \sum_{n \in C_n} u_n \tag{2}$$

These concept embeddings can be constructed entirely from synonym sets only, and have been proven effective in experiments by Phan et al. (2019).

Linear constraint: CCA We apply canonical correlation analysis (CCA) to find the best linear combination between pretrained name embeddings and the pretrained embeddings of their concept identifiers that maximizes their correlation. We can then project both the name embeddings and the concept embeddings to this new space for training objectives that use them as input. In order to not lose any information for further training, the projected embedding space has the same dimensionality as the original embedding space.

Siamese triplet loss To enforce embedding similarity between names that are synonyms or semantically related, we use a siamese triplet loss (Chechik et al., 2010). This loss forces the encoding of a biomedical name to be closer to the encoding of a true synonym than that of a negative sample name, within a specified (possibly tuned) margin:

$$\begin{aligned} pos &= d(f(CCA(n)), f(CCA(n_{pos}))) \\ neg &= d(f(CCA(n)), f(CCA(n_{neg}))) \\ L_{syn} &= max(pos - neg + margin, 0) \end{aligned} \tag{3}$$

where CCA denotes that the pretrained name embedding used as input for the DAN has first been transformed by the CCA constraint. We take cosine distance as distance function *d*. To select negative names during training we apply distance-weighted negative sampling (Wu et al., 2017) over all training names.

Prototypical grounding constraint To enforce prototypical grounding, we average the name encodings of all synonyms or semantically related terms belonging to a concept identifier, in order to approximate a prototypical representation of the concept in the name embedding space. We then minimize the cosine distance between this prototypical concept representation and the pretrained embedding of the concept:

$$f(p) = \frac{1}{|C_n|} \sum_{n \in C_n} f(CCA(n)) \\ L_{proto} = d(f(p), CCA(u_p)) \tag{4}$$

To avoid overfitting, we enforce this objective using a random dropout of synonyms from C_n, in order to stochastically approximate prototypical similarity to the concept embedding.

This constraint implies that the dimensionality of the encoder output should be the same as the dimensionality of the pretrained concept embeddings. However, if the dimensionality of the concept embeddings is smaller than the desired output dimensionality, this could be solved using e.g. random projections, which work well for increasing the dimensionality of neural encoder inputs (Wieting and Kiela, 2019).

Multi-task setup Our multi-task setup simply sums the siamese triplet losses and prototypical grounding:

$$L = L_{syn} + L_{proto} \tag{5}$$

where both losses use either the original pretrained name and concept embeddings, or their CCA projections. While the proportion of both losses could be tuned using coefficients, our experiments prove this to be redundant, since both losses systematically converge to zero or near-zero values in all experiments.

4 Data

4.1 Disorder names

4.1.1 SNOMED-CT

Following Kartsaklis et al. (2018) and Phan et al. (2019), we use SNOMED-CT[1] disorder names as biomedical synonym sets. However, since this data is of a diverse nature and quality, we try to select the most natural and coherent data by matching it with a large target domain of processed MEDLINE articles released by Hakala et al. (2016) containing 76M sentences with 120M unique noun phrases scraped from 4K articles. We match disorder names with our target domain in 4 consecutive steps. Firstly, we only retain disorder names of which all tokens appear in the vocabulary of our target domain. Secondly, many disorder names have duplicates with a small set of redundant metatags such as *(disorder)* and *(finding)* added to the name, which very rarely appear as natural language in our target domain (we list these metatags in Appendix A). Since they do not reflect relevant synonymy, we leave out such duplicates. Thirdly, we only retain disorder names of up to 6 tokens, since this is the maximum length of the 20K disorder names which directly match noun phrases from our target domain. This is also similar to the length distribution in disorder normalization benchmarks as the NCBI Disease corpus (Doğan et al., 2014) and the ShARe/CLEF eHealth 2013 corpus (Pradhan et al., 2015). Lastly, we leave out all disorder names which belong to more than one concept identifier.

4.1.2 ICD-10

The SNOMED-to-ICD-10 mapping, which has been officially provided by the U.S. National Library of Medicine[2], groups multiple SNOMED-CT concepts together under more coarse-grained ICD-10 codes, using concept unique identifiers (CUIs) from the UMLS[3] ontology which encompass those SNOMED-CT concepts. We fuse the synonym sets of SNOMED-CT concepts belonging to the same ICD-10 concept into a single set of semantically related terms. Table 1 gives some examples of the SNOMED-to-ICD-10 mappings. These examples show how ICD-10 concepts introduce a broader range of synonymy. While many of the SNOMED-CT synonyms can be resolved using word-level analogies (e.g. *myocardial $\underline{\text{depression}}$* vs. *myocardial $\underline{\text{dysfunction}}$*), the ICD-10 related terms that bridge different SNOMED-CT concepts require more domain-specific semantics to be linked (e.g. for matching *myocardial dysfunction* with *muscular degeneration of heart*).

4.2 Heterogeneous names: MedMentions

The recently released MedMentions corpus (Mohan and Li, 2019) enables training and testing of biomedical name encoders on a larger scale and over a wider variety of semantic types than previous benchmarks. It maps a vast amount of biomedical names mentioned in PubMed abstracts to their corresponding concept unique identifier (CUI) in the UMLS ontology. The annotated subcorpus *MedMentions ST21pv* annotates names belonging to UMLS concepts covering 21 different semantic

---

[1] `https://www.snomed.org`

[2] `https://www.nlm.nih.gov/research/umls/mapping_projects/snomedct_to_icd10cm.html`

[3] `https://uts.nlm.nih.gov/`

	Disorder		Heterogeneous
	ICD-10	SNOMED-CT	MedMentions
train concepts	5,136	20,140	18,417
train mentions	31,610	29,517	38,445
train synonym pairs	120,768	26,214	118,300
validation mentions	4,802	1,355	42,924
test mentions	7,142	2,752	43,544
zero-shot concepts	1,000	1,485	1,098
zero-shot mentions	6,490	4,199	4,705

Table 2: An overview of all the data used in our experiments.

types. We fuse these textual mentions of names into synonym sets. Since they are all verified to occur in existing biomedical free-text, we don't perform any preselection at all. This also means that there are words which are out-of-vocabulary for our fastText model: 10% of the MedMentions names contain such words, which constitute 15% of the total MedMentions vocabulary. As a result, the MedMentions data can show how reliable our approach is in cases where the vocabulary of the word embeddings does not perfectly overlap with the target domain.

5 Experiments and results

5.1 Ranking tasks and data distributions

Ranking tasks We evaluate the usefulness of biomedical name representations for synonym retrieval and concept mapping by applying 3 different performance metrics to a single ranking task. Given a mention *m* of a biomedical name which belongs to the concept identifier *c*, we have to rank a set of biomedical names *S* which includes $C_{syn} \subset S$, a set of names which belong to the same concept identifier *c* as the mention *m*. To rank the biomedical names according to their similarity to the mention, we first encode both the mention *m* as well as every name $n \in S$, and then rank every name n using the cosine similarity between the encoded mention $f(m)$ and the encoded name $f(n)$.

The aim of this task is to rank every correct synonym or semantically related name $syn \in C_{syn}$ as high as possible. We measure the synonym retrieval and concept mapping performance for this task using different metrics. For synonym retrieval, we report **Mean average precision (mAP)** over all synonyms. For concept mapping, we report **Accuracy (Acc)**, the proportion of instances where the highest ranked name *n* is a correct synonym $syn \in C_{syn}$, and **Mean reciprocal rank (MRR)** of the highest ranked correct synonym.

Data distributions Table 2 gives an overview of the data distributions after splitting. For MedMentions, we take our train, validation, test, and zero-shot data from the data splits provided by *MedMentions ST21pv*. For SNOMED-CT and ICD-10, we devise our own sampling method. Firstly, we randomly divide the synonym sets in training concepts and zero-shot test concepts. Secondly, to hold out test mentions from the training data, we randomly sample a single name from each concept which has at least two names (as to avoid empty training concepts), and repeat this procedure to get more test data. We then carry out the same procedure to sample validation data which we use to calculate the stopping criterion during training.

We calculate synonym retrieval and concept mapping performance for the test and validation mentions by ranking for a test mention *m* all names *S* present in the training data, including the synonyms C_{syn} which are present in the training data for the concept identifier *c* of the test mention. The performance of the encoders for the training data is calculated by treating a single training name at a time as test item.

The zero-shot test concepts are used to observe how well our encoders can extrapolate to previously unobserved concepts, for which the encoder has not specifically learned conceptual grounding. We frame the zero-shot setup as a way of testing transfer learning within the same domain, by not including any training names at all. This setup can show that our encodings are robust enough to be used out-of-the-box in entirely novel settings. For this setup, we treat a single zero-shot name at a time as test item, and rank all correct synonyms C_{syn} present in the zero-shot data among all names *S* from the zero-shot data.

5.2 Reference model and baselines

Reference model: BNE We compare our DAN model against the Biomedical Name Encoder (BNE) by Phan et al. (2019), which we train using the exact same data. To have a direct comparison with their model, we leave out the character embeddings from their encoder architecture and only use our fastText word embeddings as input embeddings. This results in a bidirectional LSTM (BiLSTM) (Graves and Schmidhuber, 2005) with

	Train			**Test**			**Zero-shot**		
	mAP	Acc	MRR	mAP	Acc	MRR	mAP	Acc	MRR
Sent2Vec	0.27	0.42	0.51	0.30	0.47	0.56	0.43	0.67	0.74
BioBERT	0.35	0.51	0.60	0.39	0.60	0.68	0.52	0.78	0.83
fastText	0.38	0.56	0.65	0.43	0.66	0.74	0.56	0.83	0.87
CCA fastText	0.42	0.59	0.68	0.47	0.70	0.76	0.61	0.85	0.89
CCA+DAN	**0.99**	**0.99**	**0.99**	**0.79**	**0.77**	**0.80**	**0.67**	**0.87**	**0.90**
DAN	0.98	0.97	0.98	0.76	0.75	0.79	0.65	0.86	0.89
BNE	0.77	0.81	0.86	0.63	0.75	**0.80**	0.65	**0.87**	**0.90**

Table 3: Synonym retrieval and concept mapping scores for the ICD-10 encoders. The highest score is denoted in bold, the second highest is underlined.

	Train			**Test**			**Zero-shot**		
	mAP	Acc	MRR	mAP	Acc	MRR	mAP	Acc	MRR
Sent2Vec	0.41	0.35	0.45	0.38	0.44	0.54	0.55	0.57	0.67
BioBERT	0.49	0.41	0.53	0.49	0.58	0.68	0.62	0.65	0.74
fastText	0.59	0.55	0.64	0.56	0.68	0.76	0.71	0.75	0.82
CCA fastText	0.62	0.57	0.67	0.59	0.70	0.78	0.73	0.76	0.83
CCA+DAN	**0.99**	**0.99**	**0.99**	**0.84**	**0.81**	**0.85**	**0.81**	**0.85**	**0.89**
DAN	0.94	0.91	0.94	0.78	0.78	0.83	0.79	0.84	0.88
BNE	0.68	0.63	0.72	0.63	0.73	0.80	0.75	0.80	0.85

Table 4: Synonym retrieval and concept mapping scores for the SNOMED-CT encoders. The highest score is denoted in bold, the second highest is underlined.

max pooling and a linear transformation:

$$\begin{aligned} h_n &= max(BiLSTM(u_{t1},..,u_{tn})) \\ f(n) &= W(h_n) + b \end{aligned} \tag{6}$$

We also include the publicly released BNE model with skipgram word embeddings, BNE + SG_w, [4] which was trained on approximately 16K synonym sets of disease concepts in the UMLS, containing 156K disease names. We don't include this model for the disorder data, since it was trained on at least part of that data, and we want to avoid that data leakage affects the fairness of the model comparisons.

Baselines As baseline encoder we use the 300-dimensional **fastText** name embeddings which are used as input for the DAN (defined in Equation 1 in Section 3.1). This encoder is an example of a Simple Word-Embedding Model (SWEM) with average pooling, which has been proven to be a strong baseline for various NLP tasks (Shen et al., 2018). We also include two other pretrained baselines among our comparison of encoders: 600-dimensional **Sent2Vec** (Pagliardini et al., 2018) embeddings with word unigram and bigram representations, trained on the same MEDLINE data as our fastText embeddings; and averaged 728-dimensional context-specific token activations extracted from the publicly released **BioBERT** model (Lee et al., 2019).

5.3 Training details

We fit the CCA for the linear constraint using all training names and their corresponding concept prototypes constructed from the same training names. The encoder architectures of our own DAN model and the BNE reference model are implemented in PyTorch (Paszke et al., 2019). Both the input and output dimensionality are 300 (which is the dimensionality of the input fastText embeddings described in Section 3.1). All encoder architectures for which we report results performed best with a single hidden layer.

We tuned the hidden size of the DAN to 38,400 dimensions using a grid search over 300×2^n, with n starting at 1 and being increased until performance declined again. We tuned the BiLSTM for the BNE model to 4,800 dimensions using the same grid search, to make sure the architecture

---

[4] `https://github.com/minhcp/BNE`

	Train			**Test**			**Zero-shot**		
	mAP	Acc	MRR	mAP	Acc	MRR	mAP	Acc	MRR
Sent2Vec	0.30	0.37	0.47	0.46	0.65	0.71	0.34	0.46	0.54
BioBERT	0.28	0.40	0.47	0.41	0.64	0.68	0.25	0.43	0.49
fastText	0.41	0.51	0.61	0.51	0.70	0.76	0.43	0.61	0.68
CCA fastText	0.44	0.53	0.63	0.53	0.72	**0.77**	**0.45**	**0.62**	**0.70**
CCA+DAN	**0.88**	**0.89**	**0.93**	**0.70**	**0.73**	**0.77**	**0.45**	0.60	0.67
DAN	0.83	0.85	0.90	0.67	0.71	0.76	0.43	0.59	0.67
BNE	0.71	0.74	0.81	0.64	0.72	**0.77**	**0.45**	**0.62**	**0.70**
BNE (Phan et al., 2019)	0.40	0.52	0.60	0.50	0.68	0.74	0.40	0.58	0.66

Table 5: Synonym retrieval and concept mapping scores for the MedMentions encoders. The highest score is denoted in bold, the second highest is underlined.

ICD-10 code	**R07.1**		
Test mention	**pain provoked by breathing**		
Target synonyms	anterior pleuritic pain / breathing painful / chest pain on breathing / pleural pain / pleuritic pain		
	CCA+DAN	**BNE**	**fastText**
Top 10 ranking	*chest pain on breathing*	*chest pain on breathing*	*chest pain on breathing*
	anterior pleuritic pain	*breathing painful*	*breathing painful*
	pleuritic pain	back pain worse on sneezing	disorder characterized by back pain
	breathing painful	disorder characterized by back pain	disorder characterised by back pain
	pleural pain	disorder characterised by back pain	back pain worse on sneezing
	chest pain	*anterior pleuritic pain*	distress from pain in labor
	chronic chest pain	pain in heart	persistent pain following procedure
	pain in heart	*pleuritic pain*	chronic mouth breathing
	upper chest pain	precordial pain	chronic chest pain
	parasternal pain	chronic chest pain	dermatitis caused by sweating and friction

Table 6: A comparison of the synonym retrieval by various encoders for the ICD-10 test mention *pain provoked by breathing*. While fastText is already good at matching a few semantically related terms at the top, it retrieves no further names in its top ranks. The BNE ranking picks up on more specific biomedical semantics, but still has a limited coverage. In contrast, the conceptually grounded CCA+DAN ranks all 5 target names at the top.

was compared fairly to our model. At that point, the DAN has ±23M trainable parameters, whereas the BiLSTM already has ±200M trainable parameters. This allows us to empirically confirm that our proposed DAN model is more computationally efficient than the BNE BiLSTM.

Adam optimization (Kingma and Ba, 2015) is performed on a batch size of 64, using a learning rate of 0.001 and a dropout rate of 0.5. Input strings are first tokenized using the Pattern tokenizer (Smedt and Daelemans, 2012) and then lowercased. We use a triplet margin of 0.1 for the siamese triplet loss L_{syn} defined in Equation 3. For the prototypical constraint L_{proto} defined in Equation 4, we use a synonym dropout rate of 0.5. As stopping criterion we use the mAP of synonym retrieval for held-out validation names: we stop training once this score for the current epoch is worse than for the previous epoch.

5.4 Results and discussion

We compare the 3 baselines and the BNE reference model against 3 variants of our model. The CCA fastText model only applies the learned CCA mapping to the pretrained fastText embeddings. The CCA+DAN model applies the linear CCA constraint before training, while the DAN model leaves out the linear constraint.

ICD-10 & SNOMED-CT Table 3 and 4 show the concept mapping and synonym retrieval performance of the different encoders for the ICD-10 and SNOMED-CT data. We see that the fastText baseline consistently outperforms the other baselines. Applying the CCA transformation to the fastText baseline improves performance for every metric, including zero-shot cases. In other words, applying this linear constraint for conceptual grounding already leads to better extrapolation. The DAN model, which combines the siamese triplet loss

MedMentions CUI	**C0870951**		
Test mention	**cariogenesis**		
Target synonyms	caries / cavities / dental caries / mod cavities / tooth decay		
	CCA+DAN	**BNE**	**fastText**
Top 10 ranking	*dental caries*	*caries*	*caries*
	caries	biofilm formation	caries prevention
	mod cavities	formation of these biofilms	preventive treatment for dental caries
	tooth decay	*dental caries*	*dental caries*
	preventive treatment for dental caries	formation of biofilms	biofilm formation
	streptococcus mutans	caries prevention	formation of biofilms
	pellicle formation	biofilm	streptococcus mutans
	cavities	biofilm forming	anti-staphylococcal biofilm agents
	bottle tooth decay	biofilm community	formation of these biofilms
	biofilm formation	pellicle formation	dental plaque

Table 7: A comparison of the synonym retrieval by various encoders for the MedMentions test mention *cariogenesis*. While the BNE model does not improve over the fastText baseline, the conceptually grounded CCA+DAN already has complete coverage of all 5 target synonyms at rank 8.

with only the prototypical grounding loss, is able to fit the training data to near perfection without overfitting, since it generalizes well across both test and zero-shot data. Applying the CCA constraint before training increases the performance even more. These observations support the hypothesis of this paper that increasing the effectiveness of conceptual grounding can improve trained encoders.

The results also clearly confirm the robustness of our approach: synonym retrieval is dramatically improved for the test data, without any performance loss for concept mapping. In other words, the representations have encoded more domain-specific semantics while retaining the relevant lexical information. Table 6 gives an example of the impact of our conceptual grounding constraints for ICD-10 test data: the model is able to encode domain-specific semantics beyond word-level analogies for the semantically related names of the test mention *pain provoked by breathing*. Not only does the CCA+DAN model rank all semantically related names at the top: all the following top-ranked names, such as *chest pain*, also have clear semantic links to the mention. In contrast, the BNE model ranks less related names such as *back pain worse on sneezing* and *disorder characterized by back pain* higher than correct synonyms such as *pleuritic pain*.

MedMentions Table 5 shows the performance of the different encoders for the MedMentions data. Table 7 gives an example of how, similar to the disorder data, our CCA+DAN encoder is able to encode specific semantics that the BNE model is lacking: the conceptual grounding constraints have allowed our encoder to represent the semantic similarity between *cariogenesis*, *tooth decay* and *cavities*, while the BNE model does not improve over the fastText baseline.

Despite showing similar trends to the disorder data, the relative improvements of our CCA+DAN encoder over the reference BNE model are less dramatic. Interestingly, the publicly released BNE + SG_{w} model trained by Phan et al. (2019) performs worse out-of-the-box than our pretrained fastText embeddings. This highlights the difficulty of achieving true robustness of biomedical name encoding.

5.5 Semantic relatedness benchmarks

We also evaluate our name encoders on two biomedical benchmarks of semantic similarity, which allow to compare cosine similarity between name embeddings with human judgments of relatedness. MayoSRS (Pakhomov et al., 2011) contains multi-word name pairs of related but different concepts, and can indicate how much generalized domain knowledge has been captured by our conceptual grounding constraints. UMNSRS (Pakhomov et al., 2016) contains only single-word pairs, which also stem from different concepts. This benchmark makes a distinction between *similarity* and *relatedness*.

The correlations in Table 8 confirm the robustness of our conceptually grounded biomedical name representations. While the correlations for the BNE models barely improve over those of the fastText embeddings, our CCA+DAN encoder improves substantially over all 3 benchmarks, regard-

	MayoSRS (rel)	UMNSRS (rel)	UMNSRS (sim)
fastText	0.443	0.473	0.479
CCA+DAN, ICD-10	**0.666**	0.556	0.561
CCA+DAN, SNOMED-CT	0.648	0.537	0.540
CCA+DAN, MedMentions	0.600	0.526	0.543
Phan et al. (2019)	0.626	**0.580**	**0.606**
BNE, ICD-10	0.492	0.472	0.503
BNE, SNOMED-CT	0.415	0.510	0.527
BNE, MedMentions	0.506	0.467	0.500

Table 8: Spearman's rank correlation coefficient between cosine similarly scores of name embeddings and human judgments, reported on semantic similarity (sim) and relatedness (rel) benchmarks. The highest score is denoted in bold, the second highest is underlined.

less of the data source it was trained on. Remarkably, while the publicly released BNE model of Phan et al. (2019) was trained on 156K disease names, the CCA+DAN encoder already outperforms it on MayoSRS when trained on the ICD-10 and SNOMED-CT subsets, which contain only 30K disease names. This proves that Deep Averaging Networks can be effective even for large-scale encoding of biomedical names. Moreover, this finding suggests that future work on biomedical name encoders should not take complex neural architectures for granted. On the contrary, enforcing more relevant constraints such as our conceptual grounding constraints can boost even lightweight encoder architectures.

6 Conclusion and future work

In this paper, we have shown how two conceptual grounding constraints for biomedical name encoders can infuse name representations with more domain-specific semantics without losing robustness. These representations can help with retrieving literal synonyms as well as semantically related terms, and can be sufficiently expressed by a Deep Averaging Network, which is a feedforward neural network that only takes averaged word embeddings as input.

We believe future work can include a comparison of neural encoding architectures with a wider range of complexity. Decreasing the complexity of neural architectures can allow for including more comprehensive training objectives which target more effective encoding of domain-specific semantics.

Acknowledgments

We would like to thank the anonymous reviewers for their feedback. This research was carried out in the framework of the Accumulate VLAIO SBO project, funded by the government agency Flanders Innovation Entrepreneurship (VLAIO). We also like to thank Madhumita Sushil and Nicolae Banari for their comments.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. 2010. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11:1109–1135.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. NCBI disease corpus: A resource for disease name recognition and concept normalization. *Journal of Biomedical Informatics*, 47:1–10.

Jennifer D'Souza and Vincent Ng. 2015. Sieve-based entity linking for the biomedical domain. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 297–302.

Alex Graves and Jurgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Kai Hakala, Suwisa Kaewphan, Tapio Salakoski, and Filip Ginter. 2016. Syntactic analyses and named entity recognition for PubMed and PubMed Central — up-to-the-minute. *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 102–107.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Association for Computational Linguistics*.

Dimitri Kartsaklis, Mohammad Taher Pilehvar, and Nigel Collier. 2018. Mapping text to knowledge graph entities using multi-sense LSTMs. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1959–1970.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations (ICLR)*.

Robert Leaman, Rezarta Islamaj Doğan, and Zhiyong Lu. 2013. DNorm: disease name normalization with pairwise learning to rank. *Bioinformatics*, 29(22):2909–2917.

Robert Leaman, Ritu Khare, and Zhiyong Lu. 2015. Challenges in clinical natural language processing for automated disorder normalization. *Journal of Biomedical Informatics*, 57:28–37.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Haodi Li, Qingcai Chen, Buzhou Tang, Xiaolong Wang, and Hua Xu. 2017. CNN-based ranking for biomedical entity normalization. *BMC Bioinformatics*, 18(Suppl 11)(385).

Nut Limsopatham and Nigel Collier. 2016. Normalising medical concepts in social media texts by learning semantic representation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1014–1023, Berlin, Germany. Association for Computational Linguistics.

Sunil Mohan and Donghui Li. 2019. Medmentions: A large biomedical corpus annotated with {umls} concepts. In *Automated Knowledge Base Construction (AKBC)*.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Serguei V.S. Pakhomov, Greg Finley, Reed McEwan, Yan Wang, and Genevieve B. Melton. 2016. Corpus domain effects on distributional semantic modeling of medical terms. *Bioinformatics*, 32(23):3635–3644.

Serguei V.S. Pakhomov, Ted Pedersen, Bridget McInnes, Genevieve B. Melton, Alexander Ruggieri, and Christopher G. Chute. 2011. Towards a framework for developing semantic relatedness reference standards. *Journal of Biomedical Informatics*, 44:251–265.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Minh C. Phan, Aixin Sun, and Yi Tay. 2019. Robust representation learning of biomedical names. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3275–3285, Florence, Italy. Association for Computational Linguistics.

Sameer Pradhan, Noémie Elhadad, Brett R South, David Martinez, Lee Christensen, Amy Vogel, Hanna Suominen, Wendy W Chapman, and Guergana Savova. 2015. Evaluating the state of the art in disorder recognition and normalization of the clinical narrative. *Journal of the American Medical Informatics Association*, 22(1):143–154.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding based models and associated pooling mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 440–450.

Tom De Smedt and Walter Daelemans. 2012. Pattern for Python. *Journal of Machine Learning Research*, 13:2031–2035.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4077–4087. Curran Associates, Inc.

Mujeen Sung, Hwisang Jeon, Jinhyuk Lee, and Jaewoo Kang. 2020. Biomedical entity representations with synonym marginalization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3641–3650, Online. Association for Computational Linguistics.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *International Conference for Learning Representations (ICLR)*.

John Wieting and Douwe Kiela. 2019. No training required: Exploring random encoders for sentence classification. In *International Conference on Learning Representations*.

Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krahenbuhl. 2017. Sampling matters in deep embedding learning. In *ICCV*.

A Redundant metatags

In section 4.1.1, we mention that many names from our SNOMED-CT data are duplicates of other names, with the only difference being that they also contain the following redundant metatags (in order of frequency):

- (disorder)
- (finding)
- (nos)
- (morphologic abnormality)
- (situation)
- (event)
- (observable entity)
- (qualifier value)
- (context-dependent category)
- (procedure)
- (function)
- (attribute)
- (clinical)

Adaptive Mixed Component LDA for Low Resource Topic Modeling

Suzanna Sia **Kevin Duh**
Johns Hopkins University
Baltimore, MD, USA
ssia1@jhu.edu kevinduh@cs.jhu.edu

Abstract

Probabilistic topic models in low data resource scenarios are faced with less reliable estimates due to sparsity of discrete word co-occurrence counts, and do not have the luxury of retraining word or topic embeddings using neural methods. In this challenging resource constrained setting, we introduce an automatic trade-off between the discrete and continuous representations via an adaptive mixture coefficient, which places greater weight on the discrete representation when the corpus statistics are more reliable. The adaptive mixture coefficient takes into account global corpus statistics, and the uncertainty in each topic's continuous distribution. Our approach outperforms the fully discrete, fully continuous, and static mixture model on topic coherence in low resource monolingual and multilingual settings.

1 Introduction

In topic modeling, the goal is to learn key themes in a corpus for exploratory document analysis (Boyd-Graber et al., 2017). Latent Dirichlet Allocation (LDA; Blei et al. (2003)) has been the bedrock for topic modeling and remains a hard to beat baseline for the general scenario which models with only words and documents.

We examine topic modeling in a low resource data setting (Hao et al., 2018), which has seen little attention but is commonly encountered in the digital humanities where document collections are potentially small (Jockers and Mimno, 2013; Schöch, 2017; Navarro-Colorado, 2018). [1] In such scenarios, word co-occurence statistics are less reliable due to sparsity of discrete counts.

With the rise of neural word embeddings (Mikolov et al., 2013), the defacto approach to improving over discrete models has been to utilise continuous representations (regardless of whether the setting is low resource). Early work by Liu et al. (2015) introduced topic dependent word embeddings, while others subsequently use embeddings to influence the discrete topic-word distribution (Zhao et al., 2017; Dieng et al., 2019). However, the low resource scenario constrains us to existing pre-trained embeddings, as the number of train documents is limited to several thousands and thus prohibitively small to train neural models (Srivastava and Sutton, 2017; Zhu et al., 2018; Liu et al., 2019; Hu et al., 2020; Zhu et al., 2020).

We therefore consider approaches that do not require further tuning of embeddings, and operate within the well established LDA probabilistic inference framework in the continuous space. There have been multiple attempts to replace discrete words with pre-trained embeddings (replacing the multinomial topic-word distribution with a continuous topic-word distribution), doing away with discrete words completely (Das et al., 2015; Batmanghelich et al., 2016; Xun et al., 2017). Given the dominance of pre-trained word embeddings in modern NLP, would continuous representations outperform discrete representations even in low resource settings? Surprisingly, we find that discrete LDA outperforms its fully continuous counterpart on topic coherence measures which correlate with human judgement (Lau et al., 2014).

How then can we utilise pre-trained continuous representations for learning better topics? A natural direction is hybrid models based on statistical counts and pre-trained neural representations (Neubig and Dyer, 2016). Early work by Nguyen et al. (2015) used a mixture of discrete and continuous topic-word distributions with static mixture coefficients. However we find that this does not improve over discrete LDA, which motivates a more

---

[1] This differs from the short text setting which has a large number of train documents, that has been addressed by multiple work (Li et al., 2016a; Qiang et al., 2020).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2451–2469
April 19 - 23, 2021.

nuanced treatment of the mixture coefficient.

In this work, we introduce an adaptive mixture coefficient specific to each word and each topic, which is updated at every step of Collapsed Gibbs Sampling (Griffiths and Steyvers, 2004). The intuition is as follows, topic anchor words (Lund et al., 2017) which have stronger signal from corpus statistics should rely more on the discrete distribution, while infrequent words should rely more on their embeddings (pre-trained on a large external corpus). Crucially, we do not assume any prior knowledge of the corpus used to train the word embeddings, and our parameterisation depends on the uncertainty of the continuous topic distributions at the current state of the Markov Chain during Gibbs Sampling. Our contributions are as follows:

1. By using adaptive mixed representations for the observed word with a data-dependent parameterisation, we provide an automatic trade-off between continuous and discrete representations during inference. Our method requires no additional tuning and relies purely on corpus statistics and statistics gathered from the current state of the Markov Chain.

2. We illustrate the extensibility of our approach to LDA variants with a combined topic model; Cross-lingual Adaptive LDA, and showed that adaptive mixing can balance between both discrete and continuous representations for better topic coherence on both monolingual and multilingual datasets.

2 Background

2.1 Unsupervised Learning with LDA

Discrete LDA (Blei et al., 2003) describes a generative probabilistic model of a corpus with latent topics. Formally we can define a corpus with D documents and K topics, where each document has a multinomial distribution over topics, $\Theta = \{\theta_1, \cdots, \theta_D\}$, and each topic has a multinomial distribution over words, $\Phi = \{\phi_1, \cdots, \phi_K\}$. Θ and Φ are the set of document-topic and topic-word distributions respectively. LDA relies on discrete counts and co-occurrence statistics, and therefore has poorer estimates in low resource scenarios due to data sparsity.

Gaussian LDA (Das et al., 2015) proposes a variant of LDA which operates on the continuous vector space rather than on discrete words. Each word is represented by an M-dimensional vector $\mathbf{v} \in \mathbb{R}^M$ and is drawn from a multivariate Gaussian for that topic. That is, for K topics, there are K Gaussian distributions. While there have been extensions to more complex continuous distributions such as von Mises-Fisher (Batmanghelich et al., 2016; Li et al., 2016b)), we opted to work with a simpler distribution to demonstrate the approach, which can subsequently be extended in future work.

Polylingual LDA (Mimno et al., 2009) studies LDA across more than two languages using parallel corpora. The model assumes that the document-topic distribution θ_d, is shared across languages, and that each language specific topic has a multinomial topic-word distribution, $\Phi^{\ell_1}, \Phi^{\ell_2}$ due to the discrete nature of words. Mimno et al. (2009); Ni et al. (2009) showed that Polylingual topic models can infer topic structure in multilingual corpora.

Latent Feature Topic Models A natural extension to discrete only or continuous only representations, is to model a word as being sampled with some probability from its discrete or continuous component. Nguyen et al. (2015) introduced an interpolation between the continuous and discrete representations, but convert the continuous representations back into discrete probability over word types by learning latent feature weights.

3 Discrete-Continuous Mixture LDA

We first establish an incremental extension to the Latent Feature Topic model using mixture of discrete categorical and continuous Gaussian distributions. We adopt a mixture model where each word has some probability of either coming from its categorical (discrete) or Gaussian (continuous) distribution. The **generative process** for this model with K topics is as follows:

For topic $k = 1$ to K
1. Draw covariance $\Sigma_k \sim \mathcal{W}^{-1}(\Psi, \nu_0)$
2. Draw mean $\mu_k \sim \mathcal{N}(\mu_0, \frac{1}{\kappa}\Sigma_k)$
3. Draw topic-word distribution $\phi_k \sim Dir(\lambda)$

For each document d in corpus C
1. Draw a topic distribution $\theta_d \sim Dir(\eta)$
2. For each word $w_{d,i}$
   (a) Draw topic $z_{d,i} \sim Multin(\theta_d)$
   (b) Draw $\pi \sim Beta(\alpha, \beta)$
   (c) With π, draw $w_{d,i} \sim Multin(\phi_{z_{d,i}})$
   (d) With $(1-\pi)$, draw $\mathbf{v}_{d,i} \sim \mathcal{N}(\mu_{z_{d,i}}, \Sigma_{z_{d,i}})$

where $\mathcal{W}^{-1}$ is the Inverse Wishart distribution, Ψ is the normalised Precision matrix, ν_0 is degrees of freedom, μ_0 is the prior mean for each Gaussian topic, and π is a mixture coefficient.

3.1 Gibbs Sampling for Posterior Inference

Given a corpus, our goal is to infer the posterior distribution over Θ and Φ and latent topic assignments $\mathbf{z}$, given the observations $\mathbf{x}$. We perform inference with collapsed Gibbs sampling (Griffiths and Steyvers, 2004) which can be derived by analytically integrating out Θ and Φ.

The key step in Gibbs sampling[2] samples a new topic $z_{d,i}$ assignment for each word, $w_{d,i}$ at index i in document d based on the conditional distribution where the previous assignment is ignored (denoted with $\backslash$):

$$\begin{aligned} & p(z_{d,i}{=}k|\mathbf{z}_{\backslash d,i}, \mathbf{x}, \eta, \varphi) \\ & \propto p(x_{d,i}|z_{d,i}{=}k, \varphi, \mathbf{z}_{\backslash d,i}, \mathbf{x}) \\ & \times p(z_{d,i}{=}k|\eta, \mathbf{z}_{\backslash d,i}, \mathbf{x}) \end{aligned} \tag{1}$$

η is the corresponding parameters of a Dirichlet prior for the document-topic distribution θ, and φ are parameters associated with the topic-word distribution. This is either λ for the Dirichlet prior for multinomial ϕ, or $\mu_0, \Sigma_0, \nu_0, \kappa$ for the Gaussian. In our proposed model (section 5), φ consists of both Dirichlet and Gaussian parameters.[3]

The first term on the right in Equation 1 expresses the probability of the i^{th} word in document d under topic k, while the second term expresses the probability of topic k in document d (Griffiths and Steyvers, 2004). Gaussian LDA modifies the first term to use continuous representations instead of discrete, while cross-lingual models focus on the second term which reflects document level sharing.

We focus on the first term to incorporate adaptive mixed representations in section 5.

Mixture Models Let f_1 be a discrete probability mass function with parameters φ_1 and f_2 be a continuous density function with parameters φ_2. The density can be expressed as a convex combination:

$$f(x|\varphi_1, \varphi_2) = \pi f_1(w|\varphi_1){+}(1{-}\pi) f_2(\mathbf{v}|\varphi_2) \tag{2}$$

Then, the second term in Equation 3 can be expressed as the density of $\mathbf{v}_{d,i}$ under the multivariate t distribution[4] parameterised by mean μ_k and covariance $\frac{\kappa_k+1}{\kappa_k}\Sigma_k$, with ν_k degrees of freedom. κ is a prior confidence on μ_k and Σ_k (Murphy, 2012). $\varphi = \{\lambda, \nu_0, \mu_0, \Sigma_0, \kappa\}$, including parameters of both the Dirichlet and Gaussian priors, with the subscript $_0$ indicating parameters of the conjugate prior. N indicates counts; for the first term in the RHS of Equation 3, $N_{k,w_{d,i}}$ are the counts of that particular word type (for the token $w_{d,i}$) assigned to topic k, and $N_{k,w'}$ is the number of counts of word type w' assigned to topic k, with V being the vocabulary.[5]

$$\begin{aligned} & p(x_{d,i}|z_{d,i} = k, \varphi, \mathbf{z}_{\backslash d,i}, \mathbf{x}) \\ & \propto \pi \frac{N_{k,w_{d,i}}{+}\lambda_{w_{d,i}}}{\sum_{w'}^{|V|} N_{k,w'}{+}\lambda_{w'}} \\ & + (1{-}\pi) t_{\nu_k}(\mathbf{v}_{d,i}|\mu_k, \frac{\kappa_k{+}1}{\kappa_k}\Sigma_k) \end{aligned} \tag{3}$$

Table 1 summarises current and previous work with respect to Equation 1.

4 Perspectives on Mixture Coefficient π

4.1 Perspectives on Static π

There are several ways to interpret the mixture coefficient π which interpolates between the discrete and continuous representations. Both the discrete and Gaussian LDA can be viewed as special cases of a two component mixture model, where the mixture coefficient π is either 1 or 0 respectively. π can also be viewed as a tunable hyperparameter that emphasises either representation depending on the availability of discrete word units or quality of embeddings (Nguyen et al., 2015).

4.2 Perspectives on π as a Static Random Variable Informed by Observations

From a Bayesian perspective, the mixture coefficient, $\pi \in [0, 1]$, can be modelled as a random variable following a $Beta$ distribution. This provides a distribution over component proportions (discrete or continuous) with useful conjugate properties. By Bayes Rule, posterior inference of π is proportional to prior times likelihood: $p(\pi|o) \propto p(\pi)p(o|\pi)$.

---

[2] We refer readers to Resnik and Hardisty (2010) for a detailed explanation or refresher of this process.

[3] Hyperparameters are not crucial to understanding our method, and we expand on the notation for them in Table 1.

[4] The multivariate t distribution arises in Bayesian Inference when the variance of a normally distributed random variable is unknown (Gelman et al., 2013).

[5] When $w_{d,i}$, w' are subscripts of N or λ, they are integers that index a count vector or Dirichlet parameter vector λ. e.g, when used in the context of $N_{k,w_{d,i}}$, $w_{d,i}$ is the index of the word type, for the token of i^{th} word of document d.

LDA types	Topic-Word	$p(x_{d,i}^{\ell_1} \mid z_{d,i}^{\ell_1} = k, \varphi, \mathbf{z}_{\backslash d,i}, \mathbf{x}) \propto$	$p(z_{d,i}^{\ell_1} = k \mid \eta, \mathbf{x}) =$
Discrete	Discrete	$\mathbb{D} = \frac{N_{k,w_{d,i}} + \lambda_{w_{d,i}}}{\sum_{w'}^{\lvert V \rvert} N_{k,w'} + \lambda_{w'}}$	$\frac{N_{k \backslash d,i}^{d} + \eta_k}{\sum_{k'} N_{k' \backslash d,i}^{d} + \eta - 1}$
Gaussian	Continuous	$\mathbb{C} = t_{\nu_k}(\mathbf{v}_{d,i} \mid \mu_k, \frac{\kappa_k + 1}{\kappa_k} \Sigma_k)$	
Static π	Mixture	$(1 - \pi) \cdot (\mathbb{C}) + \pi \cdot (\mathbb{D})$	
Adaptive π	Mixture	$(1 - \pi_{k,j}) \cdot (\mathbb{C}) + \pi_{k,j} \cdot (\mathbb{D})$	
Polylingual	Discrete	$\mathbb{L} = \frac{N_{k,w_{d,i}}^{\ell_1} + \lambda_{w_{d,i}}^{\ell_1}}{\sum_{w'}^{\lvert V^{\ell_1} \rvert} N_{k,w'}^{\ell_1} + \lambda_{w'}^{\ell_1}}$	$\frac{N_{k \backslash d,i}^{\ell_1,d} + N_{k}^{\ell_2,d} + \eta_k}{\sum_{k'} N_{k' \backslash d,i}^{\ell_1,d} + \sum_{k'} N_{k'}^{\ell_2,d} + \eta - 1}$
ℓ-Adapt	Mixture	$(1 - \pi_{k,j}^{\ell_1}) t_{\nu_k^{\ell}}(\mathbf{v}_{d,i}^{\ell_1} \mid \mu_k, \frac{\kappa_k + 1}{\kappa_k} \Sigma_k) + \pi_{k,j}^{\ell_1} \cdot (\mathbb{L})$	

Table 1: **Comparison of various LDA models.** For topic k, language ℓ_1, $N_k^{\ell_1,d}$ are counts of topic k in document d, $N_{k,w_{d,i}}^{\ell_1}$ are counts of the word type for the i^{th} word in document d, $w_{d,i}$ in topic k. j indexes the word type for the token $w_{d,i}$ or $v_{d,i}$, and t_{ν_k} is the probability density function of the multivariate t distribution parameterised by ν_k degrees of freedom, mean μ_k and covariance Σ_k. $\kappa_k = \kappa + \sum_{w'}^{\lvert V \rvert} N_{k,w'}$, where κ represents the belief on the prior of the multivariate Gaussian. For the cross-lingual model, ν^{ℓ} and κ_k^{ℓ} sum counts in ℓ_1 and ℓ_2. "\" denotes counts when excluding that variable. $\lambda \in \mathbb{R}^{\lvert V \rvert}$ and $\eta \in \mathbb{R}^{K}$ are hyperparameters of the Dirichlet prior distribution on topic-word and document-topic distribution respectively.

Here the observations o correspond to the discrete and continuous representations.

It can be shown due to conjugacy of the beta-binomial distribution that when the prior $p(\pi)$ is $Beta(\alpha_0, \beta_0)$, the posterior $p(\pi|o)$ is also a Beta distribution, where α' and β' are counts of words that have a discrete and continuous representation available, and α_0 and β_0 are set to 1 in the absence of any information.

$$\pi \sim Beta(\alpha_0 + \alpha', \beta_0 + \beta') \tag{4}$$

Note that with modern word embeddings such as FastText,[6] and Byte Pair Encoding methods, both discrete and continuous representations are mostly always observed together and $|V| = \alpha' \approx \beta'$, when $|V|$ is large, $\mathbb{E}[\pi] = 0.5$ with $Var[\pi] \approx 0$. Unfortunately, this view is overly "naive" as the continuous representations are not true observations, but learned representations which should not constitute full observation counts. We refer to this setting as "Static Mixing (SMIX)" in section 8, where we directly adopt $\pi = 0.5$. [7]

5 Adaptive Mixture Coefficient $\pi_{k,j}$

We recommend a more pragmatic view for balancing between (noisy) learned word embeddings and discrete counts by modeling the mixture coefficient as a topic k and word type indexed by j, $\pi_{k,j}$ specific random variable. At inference time, we sample $\pi_{k,j} \in [0, 1]$ from a $Beta$ distribution that is specific to each word type and each topic for the α parameter, and topic specific for the β parameter to compute Equation 3.

$$\pi_{k,j} \sim Beta(\alpha_{k,j}, \beta_k) \tag{5}$$

The parameter $\alpha_{k,j}$ represents the concentration on the discrete representation, while β_k represents the concentration on the continuous representation. As we do not assume any knowledge of the external corpus used to train the continuous representations, the β parameter is agnostic to the word type. On each Gibbs sampling update, $\alpha_{k,j}$ is updated by discrete counts for the categorical distribution, and β_k is updated based on the uncertainty in the t distribution as measured by the trace of the covariance matrix Σ_k.

5.1 Adaptive $\alpha_{k,j}$ Based on Counts

We specify corpus specific 'α' priors, α_j^0 for each word type indexed by j in the vocabulary as the number of word counts $N_{k,j}$ normalised by K, the number of topics, and the relative proportion of number of documents D to number of unique vocabulary words $|V|$.

$$\alpha_j^0 = \frac{D}{|V|} + \frac{\sum_k N_{k,j}}{K} \tag{6}$$

Intuitively, we expect that if a word has a higher frequency in the corpus, its statistics based on

---

[6] FastText can generate a representation for previously unseen vocabulary words based on their character Ngrams.

[7] For a vocabulary size of just 1000, $Var[\pi] = 0.00026$.

discrete counts are more reliable. However, if $|V| >> D$, count statistics become less reliable. The $\alpha_{k,j}$ parameter at each step where $N_{k,j}$ is the number of times word type at vocabulary index j was assigned to topic k is

$$\alpha_{k,j} = \alpha_j^0 + N_{k,j} \tag{7}$$

which takes a similar form to the regular closed-form conjugate posterior update in Equation 4 for discrete counts.

5.2 Adaptive β_k Based on Topic Uncertainty

Recall that while counts are appropriate for the discrete case, continuous representations are learned from an external corpus and should not constitute full observation counts. Hence there is *no closed form update for the membership of the continuous representations* (Koller and Friedman, 2009). Instead we let β_k be a random variable which reflects our current confidence in the multivariate t distribution indexed by topic k.

We approximate the uncertainty of the k topic distribution, as measured by the sum of eigenvalues of the square root of the topic covariance matrix Σ_k, equivalently written as $tr(\sqrt{\Sigma_k})$. We formulate β_k as depending on the constant terms $\frac{M}{K}$, M is the number of dimensions of the multivariate Gaussian, and (non-constant) Σ_k which is updated at every step of Gibbs Sampling:

$$\beta_k = \frac{M}{K}(tr(\sqrt{\Sigma_k}))^{-1} \tag{8}$$

The intuition for the inverse relationship between $tr(\sqrt{\Sigma_k})$ and β_k is as follows. If the topic has high variance, then β_k should be smaller as we have less confidence in its density function. The square root is a computational convenience for working with the Cholesky decomposition $L_k^T L_k = \Sigma_k$, where the last step assumes most of the variance is contained along the diagonals[8] of L_k. In the following equation, we simplify the notation of L_k to L.

$$\begin{aligned}(tr(\sqrt{\Sigma_k}))^{-1} &= (tr(\sqrt{L^T L}))^{-1} \approx (||\sqrt{L}||_F^2)^{-1} \\ &= (\sum_{i,j} L_{ij})^{-1} \approx (\sum_i L_{ii})^{-1}\end{aligned} \tag{9}$$

We elaborate on the the interpretation of B_k in Appendix C.

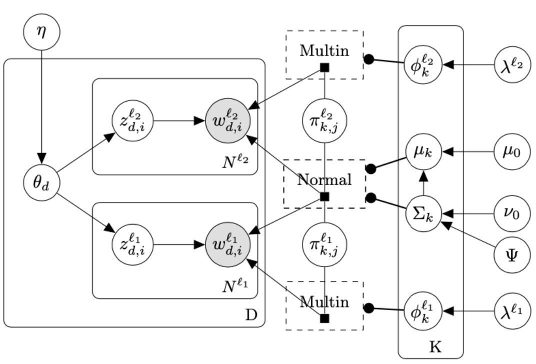


Figure 1: Cross-Lingual Adaptive LDA, with shared continuous parameters μ_k, Σ_k across languages and adaptive $\pi_{k,j}$ for every word type j and topic k. The word type j corresponds to the i^{th} token of document d. $w_{d,i}$ indicates a token when it is not being used as a subscript.

6 Computational Complexity

We consider the computational cost for every Gibbs Sampling step. The main source of computational complexity comes from inverting Σ_k which takes $O(M^3)$ when computing the probability density of $\mathbf{v}_{d,i}$ in row 2 of Table 1.

Since the covariance matrix Σ_k is symmetric and positive semi-definite, we can utilise the Cholesky decomposition where Σ_k can be decomposed as a product of upper and lower triangular matrices, $\Sigma_k = L_k^T L_k$. Although this takes $O(M^3)$, we pay this cost only once during initialisation. L_k is maintained by performing rank-1 updates and downdates (Seeger, 2004) at every step of Collapsed Gibbs Sampling.

As shown in Das et al. (2015), calculating the probability density takes $O(M^2)$ instead of $O(M^3)$. Our proposed prior for β_k sum the diagonals of L_k which takes $O(M)$ with little to no constant time overhead.

Therefore each Gibbs Sampling step takes $O(KM^2)$ where K is the number of topics whose $p(\mathbf{v}_{d,i}|z_{d,i} = k, \varphi, \mathbf{x})$ we need to compute. This is parallelisable to $O(M^2)$ as each term can be computed independently.

7 Cross-lingual Adaptive LDA

The following section describes the extension of our work from the monolingual to the cross-lingual setting. To test the robustness of our proposed model and extensibility to other models, we study the topic coherence in multilingual settings where the quality of word embeddings is thought to be

---

[8] We verified this assumption by inspecting L_k, and found that the off-diagonals tended to be smaller by a factor of 3.

worse than monolingual embeddings. We introduce a new topic model for continuous multilingual representations building on our adaptive sampling scheme, Cross-lingual Adaptive LDA in (Figure 1).

Modeling Assumptions Following Mimno et al. (2009), we assume that the document-topic distribution θ_d is shared across paired language documents, and follow a bag-of-words assumption, i.e., they need not be sentence or word-aligned. We additionally assume that multilingual word embeddings $\mathbf{v}^{\ell_1}, \mathbf{v}^{\ell_2}$ have been mapped to the same embedding space, by adopting shared Gaussian mean μ_k and covariance Σ_k across languages. This reduces the number of parameters and importantly ensures a continuous mapping across languages. Although this does not necessarily affect topic-coherence when measured within in each language, this would results in very poor cross-lingual document-topic representations. We checked this assumption by inspecting the learned topics without parameter sharing and found that topic indexes were mismatched across languages. Topic 5 in English may be about sports but Topic 5 in French may be about medicine.

7.1 Adaptive Mixing for Cross-lingual LDA

For the cross-lingual setting, our parameterisation of Equation 5 takes into account language $\ell \in \mathcal{L}$ for word type j and topic k:

$$\alpha^{\ell}_{k,j} = \frac{D}{|V^{\ell}|} + \frac{\sum_{k'}^{K} N^{\ell}_{k',j}}{K} + N^{\ell}_{k,j}$$
$$\beta^{\ell}_{k} = (\frac{M}{K \cdot |\mathcal{L}|})(tr(\sqrt{\Sigma_k}))^{-1} \qquad (10)$$

Similar to the low resource monolingual setting, our approach relies on existing pre-trained multilingual word embeddings. Note that each language may have different vocabulary size.

8 Experiments

8.1 Experimental Setup

We conduct experiments on a standard monolingual dataset and multilingual wikipedia dataset, reflecting a resource constrained setting by reducing the number of train documents. Our experiments[9] investigate the following:

- Does an adaptive mixture coefficient perform better than the fully continuous, fully discrete, and static mixture coefficient?
- How do the various mixture models perform across different number of training documents?

We compare the following models in Table 2, ℓ- indicates the cross-lingual case in Table 3 and **SMIX** is as described in subsection 4.2:

- **DISC**: Discrete LDA ($\pi = 1$)
- **GAUS**: Gaussian LDA ($\pi = 0$)
- **SMIX**: Static Mix ($\pi = 0.5$)
- **ADAP**: Adaptive Mix (adaptive π)

8.2 Datasets

We use the **20 newsgroup dataset (20NG)** which is a common text analysis dataset containing around 18000 documents and 20 categories.[10] We perform stratified shuffled sampling, using 7000 docs as holdout test and varying the number of training documents from 1000 to 8000. For each model and each training size, we present the results averaged across 5 random splits of the dataset.

Since the goal is to model the present corpora, our main results are evaluated on a held-out test set based on the same corpora. We additionally evaluate on a held-out test set following (Röder et al., 2015). GAUSS performs better in this setting, and we discuss possible reasons in Appendix F.

Wikipedia paired document corpus. For the multilingual scenario, we utilised a Wikipedia dataset (Sasaki et al., 2018) that was automatically constructed by inter-language link to the most relevant foreign language document. For the multilingual setting 1000 test pairs were standardized across all languages, and training data consisted of 8000 randomly selected document pairs for each language. We performed shuffled samping on the training data for 5 random splits of 1000 and 7000 training document pairs.

Note that *low resource topic modeling is not equivalent to low resource languages*. A language can be considered high resource but the collection of documents that we are modeling could be small.[11]

---

[9] Code made available at `https://github.com/suzyahyah/adaptive_mixture_topic_model`

[10] The dataset can be obtained at `http://qwone.com/~jason/20Newsgroups/`

[11] An example of this is the modeling of Golden Age Spanish Sonnets (Navarro-Colorado, 2018) which has a corpus size of around 5000 documents.

No. docs	DISC	GAUS	SMIX	ADAP
1000	-0.067	-0.036	-0.003	**0.022***
2000	0.006	-0.055	-0.046	**0.043***
3000	0.030	-0.089	-0.103	**0.048***
4000	0.044	-0.111	-0.141	**0.052**
5000	0.044	-0.139	-0.220	**0.059**
6000	0.059	-0.283	-0.251	**0.076***
7000	0.072	-0.213	-0.235	**0.092***
8000	0.093	-0.192	-0.261	0.094

Table 2: Performance of various models with variable number of train documents on NPMI (higher is better). Each NPMI score reported was averaged across 5 random train-test splits. $^*p < 0.05$ significant difference for paired t-test against the strongest baseline (DISC).

ℓ	No. docs	DISC	GAUS	SMIX	ADAP
ro	1000	-0.014	-0.134	-0.139	**0.012**
fr	1000	-0.010	-0.172	-0.151	**0.005**
pl	1000	-0.030	-0.138	-0.300	**-0.011**
es	1000	**0.010**	-0.280	-0.119	0.008
ro	7000	0.045	-0.307	-0.105	**0.081***
fr	7000	0.049	-0.258	-0.101	**0.052**
pl	7000	**0.032**	-0.273	-0.174	0.024
es	7000	0.039	-0.283	-0.112	**0.053***

Table 3: Performance of various models on languages *ro:*Romanian, *fr:*French, *pl:*Polish, *es:*Spanish on 1000 and 7000 documents. Each NPMI score reported was averaged across 5 random train-test splits. $^*p < 0.05$ significant difference for paired t-test against the strongest baseline (DISC).

Preprocessing Standard text preprocessing steps were applied. Stopwords, digits, punctuations, words that appeared less than 5 times and the top 10 most frequent words were removed for efficiency. Wikipedia articles were restricted to the first 200 words and document titles were removed.

Model Settings All experiments (both 20NG and the multilingual experiments) were conducted with pre-trained multilingual word embeddings from the MUSE library (Conneau et al., 2017). We trained for up to 100 iterations and checked for convergence by inspecting mixing of the posterior topic-word distributions.

Hyperparameters We initialised the prior mean μ_0 and covariance Σ_0 to the empirical mean and sample covariance respectively based on random assignment of words to topics. Following Das et al. (2015), we initialise κ to 1, ν_0 to the embedding size M of 300. Parameters of the Dirichlet prior η and λ are set to 1 and 0.01 respectively, and $K = 20$. The same hyperparameter settings were used in the multilingual setting.

All parameters of our proposed approach are based on corpus statistics, and existing parameters such as number of topics, and embedding size.

8.3 Topic Coherence Measure

Topic models are often evaluated based on the likelihood of held-out documents. However the likelihood of words from the discrete probability mass function and continuous probability density function is not directly comparable. Instead, we compute the coherence score S_k of topic k using the normalised point-wise mutual information (NPMI; Bouma (2009)) which has been found to correlate with human judgement of topic quality (Lau et al., 2014). We also evaluate on the 'C_v' metric, which is closely related (see Appendix F) from Röder et al. (2015).

NPMI ranges from $[-1, 1]$, where -1 indicates no co-occurences and 1 indicates 100% co-occurences. [12] The score of each topic S_k is computed from word pair combinations of the top T words returned by that topic.

$$S_k = \sum_{i=2}^{T} \sum_{j=1}^{i-1} NPMI(w_i, w_j) \tag{11}$$

$$NPMI(w_i, w_j) = \frac{log \frac{p(w_i, w_j) + \epsilon}{p(w_i) \cdot p(w_j)}}{-log(p(w_i, w_j) + \epsilon)} \tag{12}$$

We extract word co-occurrence statistics of the held-out documents to compute $p(w_i)$ and $p(w_i, w_j)$, and set ϵ to $1e^{-12}$ to avoid logarithm of 0. NPMI averaged across all topics are reported as $\frac{1}{K} \sum_k S_k$ in Table 2 and 3.

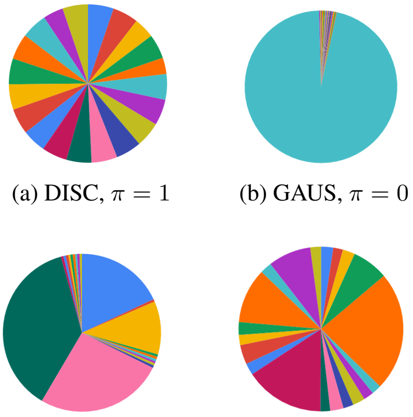
Note that the standard metric in Equation 12 will encounter division by 0 for the case where $p(w_i) \cdot p(w_j) = 0$, which is a case which frequently occurs in our low resource setting. We elaborate on this in Appendix D.

8.4 Results and Discussion

Finding 1: Adaptive Mixing performs best in resource constrained settings. We see that in Table 2, the adaptive mixture coefficient performs better under more resource scarce settings, and

---

[12] Hao et al. (2018) introduced a multilingual NPMI for low resource languages and proposed the bible as a held-out test set, but note that it is "archaic" - good at evaluating topics such as family and religion but poor at evaluating modern topics such as biology. We use regular NPMI for consistency with the monolingual setup.


(a) DISC, $\pi = 1$ (b) GAUS, $\pi = 0$

(c) SMIX, Static π (d) ADAP dynamic π

Figure 2: Topic proportions for various mixture coefficients (π) in the 20NG dataset using $K = 20$. Slices do not correspond to ground truth, and only illustrate relative proportions. Although DISC can recover a similar proportion to the ground truth, the quality of topics are not as good as ADAP.

after a certain point, is nearly equivalent to the Discrete LDA. These results are in the direction that we expect, the discrete model performs increasingly well with larger corpus sizes.

Gaussian LDA (GAUS) performs poorly with increasing number of training documents. The authors report better performance using Pointwise Mutual Information (PMI) which assigns high scores to rare words such as human names such as "scott, graham, walker.."[13] which are not representative of themes. In this work, we evaluate using normalized PMI (Bouma, 2009) which corrects for this. This is somewhat suprising given the dominance of neural methods in modern NLP, and motivates our analysis (see **Observation 1** and **Observation 2**) in the next section.

Interestingly, even with a less optimal continuous distribution, the adaptive method is able to balance between both representations with low number of training documents, and has a 'jump-start' using embeddings. We note that ADAP performs slightly less convincingly in the multilingual setting in terms of achieving statistical significance (not poorer in absolute terms), which could be due to poorer quality of multilingual embeddings.

Finding 2: Static mixture coefficient of $\pi = 0.5$ performs poorly, and while this could potentially be tuned for better performance, our adaptive method requires no tuning at all. This is discussed further in subsection 8.6.

No. train docs = 1000		No. train docs = 7000	
ADAP	DISC	ADAP	GAUS
government	**law**	jesus	john
law	**government**	word	**paul**
public	color	christ	james
laws	remember	bible	**mary**
crime	days	sin	smith
court	idea	christians	andrew
legal	told	death	gordon
trust	post	**paul**	norton
police	list	church	thomas
fbi	process	**mary**	george

Table 4: Top topic words on 20NG, bolded words are common across both topics. ADAP (Adaptive π) is able to construct topics with little training data (1000 docs), and correctly assigns human names to their ground truth topic.

One might expect that SMIX should not be worse than DISC or GAUSS, since it has access to both discrete and continuous distributions. However, the results suggest that equally weighting both the continuous and discrete topic representations, causes the model to not be able to learn effectively if they are in conflict, for e.g, continuous topic prefers topic 15 and the discrete topic prefers topic 3, and if weighted in equal proportions, this hinders the updates in Gibbs Sampling.

8.5 Analysis

Observation 1: GAUS produces narrow topics which are oddly narrow based on names (Table 4), American Cities, directions (North, South, East, West) etc. This phenomena is present in both the monolingual and multilingual models. While these groups of words may be semantically close, they are not representative 'themes' in a corpus.

This may be attributed to pre-training via skip-gram loss to predict neighbouring words (Mikolov et al., 2013). Words which are used in similar contexts have similar embeddings, and the more unique the context is, the narrower the word clusters. To verify this, we compared word clusters from the Gaussian Mixture Model (with same K) (Bishop, 2006), which uses no corpus information. We observe a high word overlap with the topics from GAUS (see Appendix E), indicating that the continuous representations dominate the corpus co-occurence statistics.

Observation 2: GAUS has a rich-get-richer phenomena. Figure 2 shows the size of topics produced by different models on the 20NG. With the exception of very narrow clusters of words,

---

[13] See Table 1 of Das et al. (2015)

most words collapse onto a single topic for GAUS (Figure 2b).[14] If many words have been assigned to one topic, that topic covariance Σ_k becomes much larger than the others, leading to subsequent $\mathbf{v}_i$ then having a higher relative density under that topic during Gibbs Sampling.

Our proposed adaptive π (Figure 2d) counteracts this effect better than the static π (Figure 2c). If Σ_k is large, to balance the effect of words having a higher density under topic k, the algorithm samples a larger π, thereby placing less weight on the continuous representation.

Observation 3: ADAP is flexible and produces reasonable topics. Discrete LDA does not perform well with low training data due to sparsity of word co-occurences. Table 4 shows that ADAP does not suffer from this and can make up for the lack of training data to produce a topic about 'government' and 'law'. Next, we observe that while GAUS clusters all human names together based on their embedding space, ADAP is not overly reliant on embeddings and can correctly assign 'Paul' and 'Mary' to its ground truth topic of christianity. Additional topics and NPMI coherence scores are available in Appendix G.

8.6 Stability of Mixture Coefficient

As our experiments were conducted with a fixed number of topics, we study the expectation of α, β, π under a varying number of topics (K from 20 to 200).

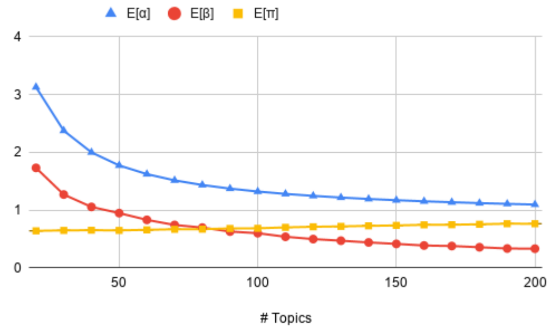


Figure 3: Stability of adaptive mixture coefficient $\pi_{i,k}$ with increasing number of topics in 20NG using 7000 documents. All α, β, π are expected values across all vocabulary words and all topics. We observe that the expected values vary smoothly with increasing K.

We approximate the expectation by the arithmetic average: $\mathbb{E}[\alpha] = \frac{1}{K}\frac{1}{|V|}\sum_k^K \sum_j^{|V|} \alpha_{k,j}$ for a fixed K, where $\mathbb{E}[\pi]$ and $\mathbb{E}[\beta]$ are calculated in the same manner. We verified that as K increases, the variance of π increases as expected, as reflected by the smoothly decreasing $\mathbb{E}[\alpha]$ and $\mathbb{E}[\beta]$.

Note that α and β take on different values for each word and topic during Gibbs Sampling. We observe that while $\mathbb{E}[\pi]$ is close to 0.5 for $K = 20$ for ADAP, it significantly outperforms SMIX ($\pi = 0.5$) in Table 2. This lends confidence to the interpretation that the adaptive mixture coefficient $\pi_{k,j}$ contributes to the better performance, as opposed to simply having a better static π.

9 Conclusion

Low resource scenarios present an interesting challenge to topic modeling due to sparsity of counts and a lack of data to train neural models. Our work proposes an automatic trade-off between externally trained continuous representations and traditional co-occurrence count-based statistics that is specific to each word and topic. The method accounts for variations in number of topics and embedding dimensions, and requires no additional tuning beyond existing methods.

Importantly, it requires no additional retraining of word embeddings or learning of topic embeddings, allowing us to rely solely on pre-trained representations and existing corpus statistics. We showed the efficacy and extensibility of our approach on a monolingual and a multilingual dataset, while introducing a new Cross-lingual Adaptive LDA topic model in the process. In future work, we aim to study the different scenarios of low resource (e.g., when there are a lot of infrequent words such as named entities) and their interaction with different embedding methods.

Acknowledgements

We thank the anonymous reviewers and JHU colleagues Anton Belyy, Sabrina Mielke, Matt Post, Tom Lippincott, Mei Hongyuan for their helpful comments on earlier drafts, Xu Yanxun, Desh Raj for technical discussion, and Tim Vieira for help with Cython.

References

Kayhan Batmanghelich, Ardavan Saeedi, Karthik Narasimhan, and Sam Gershman. 2016. Nonparametric spherical topic modeling with word embeddings. In *Proceedings of the conference. Associ-*

---

[14] This is still true for $K = 50$.

ation for Computational Linguistics. Meeting, volume 2016, page 537. NIH Public Access.

Christopher M Bishop. 2006. *Pattern recognition and machine learning*. springer.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40.

Jordan Boyd-Graber, Yuening Hu, David Mimno, et al. 2017. Applications of topic models. *Foundations and Trends® in Information Retrieval*, 11(2-3):143–296.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. 2015. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2019. Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907*.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. 2013. *Bayesian data analysis*. Chapman and Hall/CRC.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235.

Shudong Hao, Jordan Boyd-Graber, and Michael J Paul. 2018. Lessons from the bible on modern topics: Low-resource multilingual topic model evaluation. *arXiv preprint arXiv:1804.10184*.

Xuemeng Hu, Rui Wang, Deyu Zhou, and Yuxuan Xiong. 2020. Neural topic modeling with cycle-consistent adversarial training. *arXiv preprint arXiv:2009.13971*.

Matthew L Jockers and David Mimno. 2013. Significant themes in 19th-century literature. *Poetics*, 41(6):750–769.

Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539.

Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. 2016a. Topic modeling for short texts with auxiliary word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 165–174.

Ximing Li, Jinjin Chi, Changchun Li, Jihong Ouyang, and Bo Fu. 2016b. Integrating topic modeling with word embeddings by mixtures of vmfs. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 151–160.

Luyang Liu, Heyan Huang, Yang Gao, Yongfeng Zhang, and Xiaochi Wei. 2019. Neural variational correlated topic modeling. In *The World Wide Web Conference*, pages 1142–1152.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Jeffrey Lund, Connor Cook, Kevin Seppi, and Jordan Boyd-Graber. 2017. Tandem anchoring: A multiword anchor approach for interactive topic modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 896–905.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

David Mimno, Hanna M Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 880–889. Association for Computational Linguistics.

Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*.

Borja Navarro-Colorado. 2018. On poetic topic modeling: extracting themes and motifs from a corpus of spanish poetry. *Frontiers in Digital Humanities*, 5:15.

Graham Neubig and Chris Dyer. 2016. Generalizing and hybridizing count-based and neural language models. *arXiv preprint arXiv:1606.00499*.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313.

Xiaochuan Ni, Jian-Tao Sun, Jian Hu, and Zheng Chen. 2009. Mining multilingual topics from wikipedia. In *Proceedings of the 18th international conference on World wide web*, pages 1155–1156. ACM.

Jipeng Qiang, Zhenyu Qian, Yun Li, Yunhao Yuan, and Xindong Wu. 2020. Short text topic modeling techniques, applications, and performance: a survey. *IEEE Transactions on Knowledge and Data Engineering*.

Philip Resnik and Eric Hardisty. 2010. Gibbs sampling for the uninitiated. Technical report, Maryland Univ College Park Inst for Advanced Computer Studies.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408.

Shota Sasaki, Shuo Sun, Shigehiko Schamoni, Kevin Duh, and Kentaro Inui. 2018. Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 458–463.

Christof Schöch. 2017. Topic modeling genre: An exploration of french classical and enlightenment drama. *DHQ: Digital Humanities Quarterly*, 11(2).

Matthias Seeger. 2004. Low rank updates for the cholesky decomposition. Technical report.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. *arXiv preprint arXiv:1703.01488*.

Guangxu Xun, Yaliang Li, Wayne Xin Zhao, Jing Gao, and Aidong Zhang. 2017. A correlated topic model using word embeddings. In *IJCAI*, pages 4207–4213.

He Zhao, Lan Du, and Wray Buntine. 2017. A word embeddings informed focused topic model. In *Asian Conference on Machine Learning*, pages 423–438.

Lixing Zhu, Yulan He, and Deyu Zhou. 2020. A neural generative model for joint learning topics and topic-specific word embeddings. *Transactions of the Association for Computational Linguistics*, 8:471–485.

Qile Zhu, Zheng Feng, and Xiaolin Li. 2018. Graphbtm: Graph enhanced autoencoded variational inference for biterm topic model. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 4663–4672.

A Pseudocode for Crosslingual Adaptive LDA Inference

Algorithm 1: Adaptive Mixing LDA Inference

```
Data: Documents $\mathscr{D} = \{doc_1, \cdots, doc_D\}$,
      Vocab $V = \{w_1, \cdots, w_{|V|}\}$,
      Embeddings $\{\mathbf{v}_1, \cdots, \mathbf{v}_{|V|}\}, \mathbf{v} \in \mathbb{R}^M$
Result: $\Phi = \{\phi_1, \cdots, \phi_K\}, \Theta = \{\theta_1, \cdots, \theta_D\}$
Initialization:
$\mu_0 \leftarrow \frac{1}{|V|} \sum_{j=1}^{|V|} \mathbf{v}_j$ (prior mean)
$\Sigma_0 \leftarrow \frac{1}{|V|-1} \sum_{j=1}^{|V|} (\mathbf{v_j} - \mu_0)(\mathbf{v_j} - \mu_0)^T$ (prior cov)
for doc $d \in \{1, \cdots, D\}$, word $i$ do
    $z_{d,i} \leftarrow k$ uniform sample from $\{1, \cdots, K\}$
for topic $k \in \{1, \cdots, K\}$ do
    $N_k \leftarrow |\{z_{d,i} = k | \forall i \in |d|, \forall doc_d \in \mathscr{D}\}|$
    $\mu_k \leftarrow \frac{\kappa \mu_0 + N_k \bar{\mathbf{v}}_k}{\kappa_k}$
    $\Sigma_k \leftarrow \frac{\Psi_k}{\nu + N_k - M + 1}$
    $L_k^T L_k \leftarrow \Sigma_k$ (cholesky decomposition)
    $N_{k,j} \leftarrow |\{z_{d,i} = k | \forall doc_d \in \mathscr{D}\}|$ for each $w_j \in V$, $j$ is word type of token $w_{d,i}$
    $N_k^d \leftarrow |\{z_{d,i} = k | \forall\ w_i$ in $doc_d\}|$ for each $doc_d \in \mathscr{D}$
while iter<maxiter or not converged do
    for doc $d \in \{1, \cdots, D\}$, word $i$ in doc do
        $z_{old} \leftarrow z_{d,i}$
        Decrement by 1, $N_{z_{old}}, N_{z_{old},j}, N_{z_{old}}^d$
        Update $\alpha_{z_{old},j}, \beta_{z_{old}} (Eq : 7, 9)$
        Update $\mu_{z_{old}}, L_{z_{old}}$
        for topic $k \in \{1, \cdots, K\}$ do
            Sample $\pi_{k,j} \sim Beta(\alpha_{k,j}, \beta_k)$
            Compute $\_p(z_{d,i} = k | \varphi, \mathbf{x})(Eq : 1, 3$
        )
    Sample $z_{new} \sim p(z_{d,i} | \varphi, \mathbf{x})$
    Increment by 1, $N_{z_{new}}, N_{z_{new},j}, N_{z_{new}}^d$
    Update $\alpha_{z_{new},j}, \beta_{z_{new}} (Eq : 6, 8)$
    Update $\mu_{z_{new}}, L_{z_{new}}$
```

The full inference algorithm is given in Algorithm 1. For details on the parameterisation of the multivariate t, update of μ_k and computation of Ψ_k, we refer readers to Murphy (2012). For update and downdates of L_k, we refer readers to Seeger (2004) and Das et al. (2015).

B Accounting for Uncertainty in the Multivariate t Distribution

We present a small modification when calculating the density of the word vector $\mathbf{v}_{d,i}$ for each topic (row 2 of Table 1). At each step of Gibbs Sampling, the model samples a topic based on the relative likelihood of a $\mathbf{v}_{d,i}$ drawn from a t-distribution of topic k.[15] We observe that in Equation (1), the second term is dominated by the first term, where $x_{d,i}$ is a word vector representation, $\mathbf{v}_{d,i}$.

In high dimensions, $p(\mathbf{v}_{d,i}^{\ell_1} | z_{d,i}^{\ell_1} = k, \varphi, \mathbf{z}_{\setminus d,i}, \mathbf{x})$ becomes highly skewed towards a certain topic, such that the influence of the document structure becomes negligent. This motivates a correction in the first term, as the embeddings are pre-trained rather than a *true* signal. We correct the degrees of freedom ν_k to better account for uncertainty in the embedding representations.

B.1 Rescaling the Degrees of Freedom ν_k

As given by Murphy (2012), $\nu_k = \nu_0 + N_k - M + 1$, where N_k is the number of words assigned to topic k and M is the embedding dimensions. Upon initialisation, under random assignment of words to topics, $\mathbb{E}[N_k] = \frac{|\tilde{V}|}{K}$, where $|\tilde{V}|$ are all the (non-unique) words in the corpus. Since for a typical corpus $|\tilde{V}|$ is very large and $\frac{|\tilde{V}|}{K} >> M$, the degrees of freedom ν_k are very large resulting in an approximate normal distribution which is over-confident in its posterior predictions. This effectively dominates the priors for Σ_0, ν_0 or μ_0. Hence, we rescale ν_k to $\hat{\nu_k}$ from 1 to 30[16] to account for inherent uncertainty over $\mathbf{v}_i$ belonging to any particular topic.

The effect of rescaling ν_k results in a heavier-tail distribution which results in higher density for $\mathbf{v}$ which are further from μ_k. This encourages better mixing during Gibbs Sampling.

Comparison with the fully Bayesian treatment. We found this heuristic to be numerically and em-

---

[15] Readers are referred Murphy (2012) for an exposition on the form for posterior inference under a Gaussian Prior.

[16] As the degrees of freedom increase, the t distribution approaches the normal distribution. $\nu \geq 30$ is a rule of thumb for when the difference between the t distribution and normal distribution becomes negligible.

pirically more stable than a fully Bayesian treatment which encodes a higher variance in the t distribution by having a larger prior on the covariance matrix Σ_0.

First, re-estimating the covariance matrix at every step of Gibbs Sampling is numerically unstable with a large Σ_0. Next, rescaling ν_k guarantees that we maintain a heavy-tailed t distribution at every iteration of Gibbs Sampling resulting in better mixing of the Markov Chain. By adopting the rescaling heuristic, we can directly set the prior covariance Σ_0 to its sample covariance, removing one adhoc parameter choice. Since both setting a large prior Σ_0 and scaling ν_k are modeling decisions, we adopt the approach that is numerically and empirically more stable.

C Interpretation of β_k

Note that β_k can be interpreted as a random variable drawn from a Gamma distribution, with shape parameter $\frac{1}{K}$, and rate parameter $\frac{tr(\sqrt{\Sigma_k})}{M}$.

$$\beta_k \sim Gamma(\frac{1}{K}, \frac{tr(\sqrt{\Sigma_k})}{M}) \qquad (13)$$

Then, Equation 8 is the point estimate of β_k obtained from the expectation of the Gamma distribution, where $\beta_k \in (0, \infty)$ can be interpreted as real-valued 'counts' for observing the continuous representation. The rate parameter is scaled by $\frac{1}{M}$ to make the numerator robust to dimension size. Since Σ_k is positive semi-definite, and square root is a monotonically increasing function, as M increases, the trace of Σ_k increases ($\sum_i^M \sigma_i, \sigma_i \geq 0, \forall i$) and β_k decreases.

D NPMI when $p(w_i) \cdot p(w_j) = 0$

In our implementation of NPMI, we do not consider the pair if either $p(w_i)$ is $p(w_j)$ is 0, as this simply indicates a "mismatch" between training and test corpus. However if they are non-zero, and $p(w_i) \cdot p(w_j) = 0$, then the model has predicted a poor word pair that never co-occurs despite them individually appearing in the test corpus, and the score for $NPMI(w_i, w_j) = -1$.

This differs from many online implementations of NPMI which will simply set $NPMI(w_i, w_j) = 0$ if $p(w_i) \cdot p(w_j) = 0$, and 'does not penalise very poor word pairs of this nature.

Low GMM Overlap		High GMM Overlap	
Century	named	January	France
modern	live	February	French
centuries	written	December	Paris
white	including	March	Vendée
events	wrote	July	Allier
built	based	September	Gironde
reinaissance	countries	June	Spain
growth	history	October	Picardie
list	published	April	Belgium

Table 5: 'Genuine' Topic model clusters learnt from the documents vs clusters with ≥ 0.8 GMM overlap.

E Overlap with GMM

F Evaluation on held-out test set using C_v Topic coherence measure.

C_v combines the indirect cosine measure with the NPMI and the boolean sliding window and was introduced in Röder et al. (2015). The implementation of the metric and held-out wikipedia dataset is provided by `https://github.com/dice-group/Palmetto`.

No. docs	DISC	GAUSS	SMIX	ALDA
1000	0.3839	0.3886	0.4014	0.3964
2000	0.3895	0.4180	0.4319	0.3993
3000	0.4014	0.4374	0.4269	0.4162
4000	0.3985	0.4300	0.4289	0.4111
5000	0.4045	0.4278	0.4185	0.4066
6000	0.4079	0.4300	0.4161	0.4061
7000	0.4039	0.4262	0.3981	0.4092
8000	0.4090	0.4298	0.3936	0.4072

Table 6: C_v score on held-out wikipedia dataset.

We believe the main reason for $GAUSS$ to score highly on this measure is most likely due to the scoring of word pairs as described in Appendix D. This is supported by the observation that with some very rare words, the effect of ϵ in the NPMI score in C_v is large, resulting in higher scores than expected. This is described in `https://github.com/dice-group/Palmetto/issues/12`.

G Topics for 20NG

Adaptive LDA (ADAP)

Topic:0	**Topic:1**	**Topic:2**	**Topic:3**	**Topic:4**	**Topic:5**	**Topic:6**
jesus	price	religion	color	israel	win	car
bible	bike	true	data	war	april	power
church	money	faith	video	jews	white	heard
christian	sell	real	power	jewish	night	local
christ	cost	agree	mode	american	close	speed
sin	list	argument	set	israeli	gun	cars
life	worth	evidence	bit	country	period	miles
gods	pay	truth	software	armenians	red	model
word	insurance	exist	systems	university	steve	friend
earth	shipping	religious	apple	arab	record	deleted
christians	mark	reading	control	usa	start	engine
lord	market	science	serial	turkish	boston	ford
heaven	paid	person	speed	armenian	arms	told
live	business	belief	standard	greek	pens	service
john	ride	theory	output	muslims	guns	bought
paul	deal	moral	space	countries	cut	weeks
christianity	prices	statement	light	national	straight	driving
spirit	quality	values	current	canada	pts	stuff
mary	buying	claim	simple	press	congress	dealer
sense	extra	christians	fine	germany	pittsburgh	couple
S_k:0.292	S_k:0.03	S_k:0.139	S_k:0.085	S_k:-0.016	S_k:-0.108	S_k:0.136
wc:2789	wc:2720	wc:2886	wc:34212	wc:14319	wc:2178	wc:2750

Adaptive LDA (ADAP)

Topic:7	**Topic:8**	**Topic:9**	**Topic:10**	**Topic:11**	**Topic:12**
lost	original	book	support	key	games
called	idea	sale	image	chip	team
left	set	offer	info	space	players
hit	love	condition	sun	clipper	season
bad	answer	books	graphics	encryption	hockey
start	bad	excellent	appreciated	government	player
happen	hear	software	based	phone	play
base	sort	mouse	university	message	mike
started	sound	manual	technology	algorithm	baseball
pitcher	solution	graphics	programs	security	league
single	light	includes	convert	data	series
watch	thinking	send	job	source	teams
field	hate	tape	wondering	nsa	runs
cubs	head	complete	images	david	fan
expect	position	event	product	press	average
braves	ideas	items	conference	des	fans
major	times	title	design	secure	nhl
minutes	true	events	tiff	chips	pick
performance	reason	brand	june	launch	goal
james	stuff	manuals	gif	agencies	guy
S_k:0.001	S_k:0.055	S_k:-0.017	S_k:0.068	S_k:0.189	S_k:0.238
wc:2344	wc:2373	wc:2478	wc:2312	wc:2984	wc:3725

Adaptive LDA (ADAP)

Topic:13	**Topic:14**	**Topic:15**	**Topic:16**	**Topic:17**	**Topic:18**	**Topic:19**
file	list	day	card	post	person	law
window	version	bit	scsi	lot	reason	government
files	article	pretty	hard	nice	human	public
dos	send	remember	monitor	note	simply	rights
running	address	current	mac	wrong	true	private
code	mail	lot	ram	change	children	federal
machine	post	correct	controller	guess	wrong	police
screen	subject	road	memory	real	feel	house
software	faq	days	ide	understand	hand	legal
server	posted	company	data	sounds	life	laws
error	posting	stuff	bus	yeah	times	court
disk	drivers	difference	drives	add	called	weapons
display	ftp	fast	vga	figure	remember	clinton
format	driver	office	port	basically	death	class
set	internet	talking	modem	thread	day	warrant
keyboard	reply	notice	cards	hard	questions	authority
disks	lines	mentioned	disk	main	bad	citizens
size	called	start	meg	reason	issue	national
box	printer	dave	standard	discussion	fbi	president
manager	dod	type	dos	agree	news	tax
S_k:0.2	S_k:0.086	S_k:0.042	S_k:0.251	S_k:0.011	S_k:0.083	S_k:0.136
wc:4613	wc:3344	wc:2532	wc:4209	wc:2800	wc:49408	wc:2531

Discrete LDA (DISC)

Topic:0	**Topic:1**	**Topic:2**	**Topic:3**	**Topic:4**	**Topic:5**	**Topic:6**
car	power	war	post	book	religion	key
cars	battery	armenians	posting	lost	rights	government
engine	light	turkish	list	study	gun	chip
miles	design	armenian	article	msg	government	public
speed	idea	muslims	send	pain	news	clipper
driving	bit	population	source	york	support	encryption
ford	quality	jewish	mail	school	control	phone
oil	single	answer	questions	kids	article	security
heavy	type	history	address	disease	post	police
clean	model	killed	hope	drug	freedom	private
rear	noise	source	posted	books	guns	algorithm
white	systems	muslim	lines	cancer	action	data
left	normal	children	faq	cheers	society	search
heard	ground	genocide	based	double	subject	des
fun	control	human	version	original	land	law
looked	boot	shuttle	product	april	americans	nsa
air	fit	real	subject	effects	weapons	secure
tires	signal	cut	write	studies	questions	david
road	heat	turkey	note	usa	court	message
weight	fine	western	response	patients	congress	warrant
S_k:0.062	S_k:-0.008	S_k:0.106	S_k:0.107	S_k:-0.202	S_k:0.037	S_k:0.246
wc:6189	wc:6973	wc:6362	wc:6503	wc:5782	wc:7046	wc:7628

Discrete LDA (DISC)

Topic:7	**Topic:8**	**Topic:9**	**Topic:10**	**Topic:11**	**Topic:12**	**Topic:13**
bike	jesus	local	space	games	window	file
bad	life	told	university	team	image	files
stuff	church	friend	company	players	code	dos
real	faith	book	april	play	screen	software
lot	christian	weeks	technology	win	application	graphics
level	bible	dealer	press	season	color	version
times	christians	hand	conference	hockey	display	format
ride	love	check	science	league	server	advance
guess	christ	references	launch	player	set	info
sort	sin	james	news	baseball	error	package
thinking	human	heard	earth	series	size	directory
deleted	gods	talk	june	teams	running	disk
couple	agree	cover	radio	runs	images	unix
short	wrong	trouble	internet	fans	include	ftp
dod	truth	remember	greek	fan	change	hard
field	true	picture	station	pick	user	type
left	moral	experience	center	nhl	create	programs
hit	belief	set	contact	goal	widget	convert
job	person	bought	office	boston	manager	bit
canada	word	yeah	force	mike	event	applications
S_k:0.06	S_k:0.251	S_k:-0.119	S_k:-0.038	S_k:0.262	S_k:0.187	S_k:0.202
wc:7097	wc:10345	wc:5690	wc:6855	wc:9185	wc:7249	wc:8386

Discrete LDA (DISC)

Topic:14	**Topic:15**	**Topic:16**	**Topic:17**	**Topic:18**	**Topic:19**
card	price	day	true	israel	remember
monitor	sale	feel	death	claim	pretty
scsi	offer	law	matter	fbi	bad
mac	sell	remember	theory	israeli	hear
video	condition	water	argument	evidence	days
apple	list	talking	correct	happened	chance
machine	money	food	dead	arab	heard
drivers	shipping	word	position	jews	day
controller	cost	days	reason	started	guys
ram	box	called	completely	claims	feel
mode	sold	common	sex	gas	deal
board	excellent	written	homosexual	agree	understand
drives	pay	language	evidence	leave	worth
bus	power	article	issue	countries	lot
data	includes	sense	change	comment	minutes
driver	blue	die	note	peace	clinton
ide	stuff	term	wrong	statement	reason
speed	original	supposed	nature	children	watching
memory	selling	week	sexual	response	night
modem	including	english	statements	policy	wait
S_k:0.221	S_k:0.163	S_k:0.02	S_k:0.056	S_k:0.116	S_k:0.032
wc:10829	wc:7104	wc:6217	wc:7824	wc:7575	wc:6668

Static mix π, (SMIX)

Topic:0	**Topic:1**	**Topic:2**	**Topic:3**	**Topic:4**	**Topic:5**	**Topic:6**
period	lost	andor	power	government	legal	bit
paul	form	close	car	public	court	die
software	map	command	card	local	judge	advance
handbook	pens	class	bit	university	trial	address
book	fall	shift	software	children	justice	count
rules	rob	bds	window	israel	criminal	kent
held	force	virtual	key	science	federal	hardware
offers	flag	event	price	church	supreme	clinton
final	list	managed	space	press	amendment	cut
modern	support	black	monitor	country	police	string
study	named	win	speed	american	courts	bds
shadow	register	myers	disk	house	lawyers	est
riding	cat	string	machine	war	convicted	van
bowman	press	center	advance	jewish	gun	programmer
swift	phone	ticket	color	jews	weaver	und
graham	bear	friendly	sale	religious	crimes	les
happen	code	morning	code	private	offensive	ground
writing	student	taurus	screen	history	lawyer	mit
tia	straight	lot	phone	national	jury	des
manual	table	weight	systems	israeli	closed	internet
S_k:-0.38	S_k:-0.523	S_k:-0.483	S_k:0.061	S_k:-0.024	S_k:-0.187	S_k:-0.484
wc:423	wc:474	wc:590	wc:36446	wc:15595	wc:922	wc:486

Static mix π, (SMIX)

Topic:7	**Topic:8**	**Topic:9**	**Topic:10**	**Topic:11**	**Topic:12**
john	jesus	start	pittsburgh	times	left
david	bible	started	boston	manager	harry
michael	christians	starting	san	local	det
james	christ	active	york	picture	eric
andrew	gods	session	texas	hold	pre
robert	sin	gordon	chicago	power	wife
peter	heaven	sad	detroit	capitol	died
joseph	christianity	banks	toronto	sites	att
daniel	holy	surrender	angeles	tim	jason
matthew	scripture	cursor	los	names	ted
patrick	lord	weeks	buffalo	master	forged
stephen	sabbath	root	francisco	managed	bing
francis	church	helped	montreal	string	spot
charles	resurrection	closed	philadelphia	slave	van
martin	son	stopped	baltimore	finland	viola
graeme	scriptures	stopping	louis	jumper	managed
lewis	atheists	responsible	jose	pay	roommate
alan	biblical	defending	minnesota	location	courier
richard	spiritual	aura	red	beach	har
craig	mary	traders	vancouver	creation	maria
S_k:-0.529	S_k:0.288	S_k:-0.503	S_k:0.258	S_k:-0.437	S_k:-0.751
wc:1214	wc:1945	wc:963	wc:1424	wc:545	wc:627

Static mix π, (SMIX)

Topic:13	**Topic:14**	**Topic:15**	**Topic:16**	**Topic:17**	**Topic:18**	**Topic:19**
chip	file	major	day	mike	games	include
dos	set	pink	days	steve	team	including
graphics	data	track	april	dave	times	includes
scsi	true	send	months	jim	play	included
keyboard	post	fbi	night	bob	players	features
hardware	lot	thrush	week	chris	lost	listed
unix	hard	minor	weeks	tom	series	edition
floppy	list	article	month	ron	runs	refer
ibm	bad	insurance	hours	larry	win	typical
vga	called	guy	sunday	brian	major	variety
chips	real	sounds	june	joe	player	mouse
motherboard	wrong	history	march	doug	season	addition
bios	article	ulf	morning	bobby	hockey	feature
interface	reason	auto	saturday	scott	black	covers
cpu	files	guys	friday	andy	hit	sale
cdrom	version	total	thursday	frank	head	notes
computers	stuff	march	july	kevin	league	map
amiga	heard	arts	tuesday	keith	baseball	van
macintosh	support	condition	monday	jeff	pick	consists
processor	send	late	daily	ro	average	runs
S_k:0.093	S_k:0.043	S_k:-0.583	S_k:0.017	S_k:-0.277	S_k:0.155	S_k:-0.289
wc:3145	wc:64156	wc:432	wc:2285	wc:1562	wc:13073	wc:1200

Gaussian LDA, (GAUS)

Topic:0	**Topic:1**	**Topic:2**	**Topic:3**	**Topic:4**	**Topic:5**	**Topic:6**
file	wrong	american	software	government	university	games
image	real	country	files	list	science	hockey
include	stuff	usa	chip	local	study	manager
images	remember	canada	dos	federal	school	baseball
included	person	countries	graphics	national	society	nhl
picture	pretty	international	scsi	congress	department	pitcher
gif	nice	germany	hardware	party	scientific	office
count	guess	america	encryption	governments	student	cubs
poster	simply	united	server	population	engineering	stats
listed	guy	americans	tape	membership	studies	braves
refer	talking	europe	unix	foreign	master	rangers
jpeg	yeah	british	floppy	committee	degree	record
listing	suppose	english	chips	land	education	leafs
recognized	guys	middle	ibm	bds	medicine	sox
photo	stupid	modern	vga	authorities	college	flyers
pom	understand	national	interface	andor	institute	coach
apr	hey	european	motherboard	administration	teaching	bruins
van	msg	nation	bios	legislation	students	pitchers
map	forget	japanese	cpu	liberal	literature	batting
counts	imagine	australia	cache	senate	astronomy	caps
S_k:-0.412	S_k:0.043	S_k:0.117	S_k:-0.019	S_k:-0.213	S_k:-0.058	S_k:-0.204
wc:1704	wc:3985	wc:1906	wc:4382	wc:1867	wc:1955	wc:2029

Gaussian LDA, (GAUS)

Topic:7	**Topic:8**	**Topic:9**	**Topic:10**	**Topic:11**	**Topic:12**
day	car	war	jesus	israel	mike
days	card	military	religion	jews	david
april	window	citizens	church	jewish	steve
months	price	doctor	faith	israeli	mark
night	monitor	civil	christian	armenians	dave
weeks	speed	army	bible	arab	jim
week	disk	soldiers	christians	turkish	bob
month	sale	population	christ	armenian	chris
hours	machine	century	religious	greek	tom
sunday	mac	persons	gods	muslims	brian
march	code	surrender	christianity	muslim	tim
june	color	forces	atheists	islamic	adam
apr	systems	fighting	beliefs	genocide	rob
morning	screen	civilians	holy	turkey	larry
saturday	standard	women	scripture	arabs	joe
friday	display	troops	atheist	russian	ron
thursday	video	local	catholic	turks	frank
july	bike	victims	atheism	palestinian	doug
tuesday	sell	deaths	religions	greece	bobby
spring	box	crimes	islam	azerbaijan	jack
S_k:-0.005	S_k:0.068	S_k:-0.13	S_k:0.259	S_k:0.085	S_k:-0.158
wc:2507	wc:37857	wc:1808	wc:3049	wc:2601	wc:2230

Gaussian LDA, (GAUS)

Topic:13	**Topic:14**	**Topic:15**	**Topic:16**	**Topic:17**	**Topic:18**	**Topic:19**
pittsburgh	power	team	love	drug	john	road
boston	set	win	death	food	paul	city
san	data	season	children	pain	michael	local
york	bit	league	friend	drugs	james	western
washington	post	teams	house	disease	mary	west
texas	call	title	family	cancer	andrew	land
clinton	lot	final	friends	brain	smith	east
chicago	hard	cup	lord	health	george	south
detroit	called	playoffs	child	blood	morris	north
waco	bad	playoff	wife	treatment	peter	town
california	key	winner	son	patients	thomas	location
toronto	article	beat	died	risk	gordon	houses
angeles	reason	score	father	medical	norton	region
colorado	space	record	born	ice	stanley	roads
los	support	match	meg	heart	joseph	village
buffalo	send	scored	mother	diet	johnson	central
kent	version	winning	marriage	eat	grant	route
philadelphia	heard	pts	woman	alcohol	allen	border
ottawa	book	scoring	parents	patient	grace	eastern
francisco	life	division	named	chronic	adams	cities
S_k:0.05	S_k:0.048	S_k:-0.013	S_k:-0.078	S_k:-0.151	S_k:-0.536	S_k:-0.239
wc:1793	wc:68542	wc:1992	wc:2166	wc:1848	wc:1702	wc:1584

Evaluating Neural Model Robustness for Machine Comprehension

Winston Wu
Center for Language and Speech Processing
Department of Computer Science
Johns Hopkins University
`wswu@jhu.edu`

Dustin Arendt[1] **Svitlana Volkova**[2]
[1]Visual Analytics Group
[2]Data Sciences and Analytics Group
Pacific Northwest National Laboratory
`first.last@pnnl.gov`

Abstract

We evaluate neural model robustness to adversarial attacks using different types of linguistic unit perturbations – character and word, and propose a new method for strategic sentence-level perturbations. We experiment with different amounts of perturbations to examine model confidence and misclassification rate, and contrast model performance with different embeddings BERT and ELMo on two benchmark datasets SQuAD and TriviaQA. We demonstrate how to improve model performance during an adversarial attack by using ensembles. Finally, we analyze factors that affect model behavior under adversarial attack, and develop a new model to predict errors during attacks. Our novel findings reveal that (a) unlike BERT, models that use ELMo embeddings are more susceptible to adversarial attacks, (b) unlike word and paraphrase, character perturbations affect the model the most but are most easily compensated for by adversarial training, (c) word perturbations lead to more high-confidence misclassifications compared to sentence- and character-level perturbations, (d) the type of question and model answer length (the longer the answer the more likely it is to be incorrect) is the most predictive of model errors in adversarial setting, and (e) conclusions about model behavior are dataset-specific.

1 Introduction

Deep neural models have recently gained popularity, leading to significant improvements in many Natural Language Understanding (NLU) tasks (Goldberg, 2017). However, the research community still lacks in-depth understanding of how these models work and what kind of linguistic information is actually captured by neural networks (Feng et al., 2018). Evaluating model robustness to manipulated inputs and analyzing model behavior during adversarial attacks can provide deeper insights into how much language understanding models actually have (Hsieh et al., 2019; Si et al., 2020). Moreover, as has been widely discussed, models should be optimized not only for accuracy but also for other important criteria such as reliability, accountability and interpretability (Lipton, 2018; Doshi-Velez and Kim, 2017; Ribeiro et al., 2016; Goodman and Flaxman, 2017).

Context: One of the most famous people born in Warsaw was Maria Skłodowska-Curie, who achieved international recognition for her research on radioactivity and was the first female recipient of the <u>Nobel Prize</u>.
Question: What was Maria Curie the first female recipient of?
Answer: Nobel Prize

Table 1: Example MC question from SQuAD.

In this work, we evaluate neural model robustness on machine comprehension (MC), a task designed to measure a system's understanding of text. In this task, given a *context* paragraph and a *question*, the machine is tasked to provide an *answer*. We focus on *span-based* MC, where the model selects a single contiguous span of tokens in the context as the answer (Tab. 1). We (1) quantitatively measure when and how the model is robust to manipulated inputs, when it generalizes well, and when it is less susceptible to adversarial attacks, (2) demonstrate that relying on ensemble models increases robustness, and (3) develop a new model to predict model errors during attacks. Our novel contributions shed light on the following questions:

- Which embeddings are more susceptible to noise and adversarial attacks?
- What types of text perturbation lead to the most high-confidence misclassifications?
- How does the amount of text perturbation effect model behavior?
- What factors explain model behavior under perturbation?
- Are the above dataset-specific?

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2470–2481
April 19 - 23, 2021.

Broader Implications We would like to stress the importance of this type of work to ensure diversity and progress for the computational linguistics community. We as a community know how to build new models for language understanding, but we do not fully understand how these models work. When we deploy these models in production, they fail to perform well in real-world conditions, and we fail to explain why they fail; the reason being we have not performed through evaluation of model performance under different experimental conditions. Neural model evaluation and thorough error analysis, especially for tasks like machine comprehension, are critical to make progress in the field. We have to ensure our research community goes beyond F1 scores and incremental improvements and gains deeper understanding of models decision making processes to drive revolutionary research rather than evolutionary.

2 Background

There is much recent work on adversarial NLP, surveyed in Belinkov and Glass (2019) and Zhang et al. (2019). To situate our work, we review relevant research on the black-box adversarial setting, in which one does not have access or information about the model's internals, only the model's output and its confidence about the answer.[1]

In an adversarial setting, the adversary seeks to mislead the model into producing an incorrect output by slightly tweaking the input. Recent work has explored input perturbations at different linguistic levels: character, word, and sentence-level. For *character-level perturbations*, NLP systems generally do not take into account the visual characteristics of characters. Researchers have explored the effects of adding noise by randomizing or swapping characters and examining its effect on machine translation (MT) (Heigold et al., 2018; Belinkov and Bisk, 2018), sentiment analysis and spam detection Gao et al. (2018), and toxic content detection Li et al. (2018). Eger et al. (2019) replaced with similar looking symbols, and developed a system to replace characters with nearest neighbors in visual embedding space. For *word-level perturbations*, Alzantot et al. (2018) used a genetic algorithm to replace words with contextually similar words, evaluating on sentiment analysis and textual entailment. For *sentence-level perturbations*, Iyyer et al. (2018) generated adversarial paraphrases by controlling the syntax of sentences and evaluating on sentiment analysis and textual entailment tasks. Hu et al. (2019) found that augmenting the training data with paraphrases can improve performance on natural language inference, question answering, and MT. Niu and Bansal (2018) use adversarial paraphrases for dialog models.

Other related work includes Zhao et al. (2018); Hsieh et al. (2019), who generated natural looking adversarial examples for image classification, textual entailment, and MT. Specifically for MC, Jia and Liang (2017) added a distractor sentence to the end of the context, Ribeiro et al. (2018) extracted sentence perturbation rules from paraphrases created by translating to and then from a foreign language and then manually judged for semantic equivalence, and (Si et al., 2020) focused on evaluating model robustness for MC.

Unlike earlier work, we empirically show how neural model performance degrades under multiple types of adversarial attacks by varying the amount of perturbation, the type of perturbation, model architecture and embedding type, and the dataset used for evaluation. Moreover, our deep analysis examines factors that can explain neural model behavior under these different types of attacks.

Concurrent with the development of our paper, there has also been a slew of relevant work tackling robustness in neural NLP models, including Adversarial Robustness Toolbox (Nicolae et al., 2018), Advertorch (Ding et al., 2019), Foolbox (Rauber et al., 2020), Advbox (Goodman et al., 2020), OpenAttack (Zeng et al., 2020), TEAPOT (Michel et al., 2019), TextAttack (Morris et al., 2020), TextFooler (Jin et al., 2020), and Robustness Gym (Goel et al., 2021).

3 Methods

We perform comprehensive model evaluation for machine comprehension over several dimensions: the amount of perturbation, perturbation type, model and embedding variation, and datasets.

3.1 Perturbation Type

We examine how changes to the context paragraph (excluding the answer span) affect the model's performance using the following perturbations:

- **Character-level**. In computer security, this is known as a homograph attack. These attacks have been investigated to identify phishing

---

[1] For other settings (e.g. white-box), we refer the reader to the above surveys.

Original	The connection between macroscopic nonconservative forces and microscopic conservative forces is described by detailed treatment with statistical mechanics.
Character	**The** **c**onn**e**ct**io**n b**e**twe**e**n macr**o**scopi**c** nonc**o**nservat**ive** force**s** an**d** micro**s**co**pic** con**s**ervative f**orc**es is de**s**crib**e**d by detailed tre**a**tm**e**nt with st**ati**st**i**cal m**ec**hanic**s**.
Word	The connection between macroscopic nonconservative forces and **insects** conservative **troops** is **referred** by detailed treatment with statistical mechanics.
Sentence	The **link** between macroscopic **non-conservative** forces and microscopic conservative forces is described **in detail by** statistical mechanics.

Table 2: Examples of character, word and sentence-level perturbations (bold indicates perturbed text).

and spam (Fu et al., 2006b,a; Liu and Stamm, 2007) but to our knowledge have not been applied in the NLP domain. We replace 25% of characters in the context paragraph with deceptive Unicode characters[2] that to a human are indistinguishable from the original.

- **Word-level**. We randomly replace 25% of the words in the context paragraph with their nearest neighbor in the GLoVe (Pennington et al., 2014) embedding space.[3]
- **Sentence-level**. We use Improved ParaBank Rewriter (Hu et al., 2019), a machine translation approach for sentence paraphrasing, to paraphrase sentences in the context paragraph. We perform sentence tokenization, paraphrase each sentence with the paraphraser, then recombine the sentences.

For character and word perturbations, we use 25% as this is where the performance curve in Heigold et al. (2018) flattens out.[4] Regardless of the type of perturbation, we do not perturb the context that contains the answer span, so that the answer can always be found in the context unperturbed. Because paraphrasing is per sentence, we only modify sentences that do not contain the answer span. An example of each perturbation type is shown in Tab. 2.

3.2 Amount of Perturbation

For each perturbation type, we experiment with perturbing the training data at differing amounts. All models are tested on fully perturbed test data.

- **None**: clean training data.
- **Half**: perturb half the training examples.
- **Full**: perturb the entire train set.
- **Both**: append the entire perturbed data to the entire clean data.[5]
- **Ens**: ensemble model that relies on none, half and full perturbed data; we rely on ensemble voting and only include the word in the predicted answer if any two models agree.

3.3 Model Architecture and Embeddings

BiDAF model with ELMo (Seo et al., 2017; Peters et al., 2018). ELMo is a deep, contextualized, character-based word embedding method using a bidirectional language model. The Bi-Directional Attention Flow model is a hierarchical model with embeddings at multiple levels of granularity: character, word, and paragraph. We use pre-trained ELMo embeddings in the BiDAF model implemented in AllenNLP (Gardner et al., 2018).

BERT (Devlin et al., 2019). BERT is another contextualized embedding method that uses Transformers (Vaswani et al., 2017). It is trained to recover masked words in a sentence as well as on a next-sentence prediction task. The output layer of BERT is fed into a fully-connected layer for the span classification task. Pre-trained embeddings can be fine-tuned to a specific task, and we use the Huggingface PyTorch-Transformers package, specifically *bert-large-cased-whole-word-masking-finetuned-squad* model. We fine-tune for two epochs in each experimental settings.

3.4 Benchmark Datasets

We experiment on two benchmark MC datasets:

SQuAD (Rajpurkar et al., 2016). The Stanford Question Answering Dataset is a collection of over 100K crowdsourced question and answer pairs. The context containing the answer is taken from Wikipedia articles.

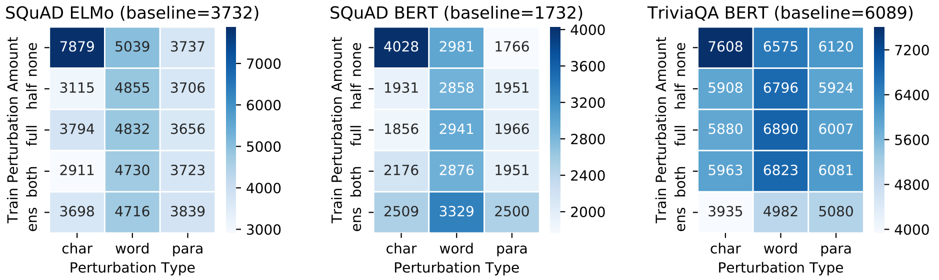
---

[2] From `https://www.unicode.org/Public/security/12.1.0/intentional.txt`

[3] Several alternative embedding techniques could be used to find the nearest neighbors e.g., Word2Vec or FastText. We use GLoVe for consistency with previous work (Li et al., 2018).

[4] Belinkov and Bisk (2018) perturbed text at 100% while Heigold et al. (2018) experimented with 5–30% perturbations.

[5] This has twice the amount of data as other settings so is not directly comparable, but many papers show that doing this can improve a model's performance.



(a) SQuAD, ELMo (b) SQuAD, BERT (c) TriviaQA, BERT

Figure 1: Number of errors by perturbation type and amount of perturbation (higher = worse model performance, or more successful attacks). *Baseline* indicates model errors whose training and testing data were not perturbed. For cross-model/embedding comparison, compare (a) and (b). For cross-dataset comparison, compare (a) and (c). The *ens* training setting is an ensemble of results from the none, half, and full settings.

TriviaQA (Joshi et al., 2017). A collection of over 650K crowdsourced question and answer pairs, where the context is from web data or Wikipedia. The construction of the dataset differs from SQuAD in that question answer pairs were first constructed, then evidence was found to support the answer. We utilize the Wikipedia portion of TriviaQA, whose size is comparable to SQuAD. To match the span-based setting of SQuAD, we convert TriviaQA to the SQuAD format using the scripts in the official repo and remove answers without evidence.

4 Evaluation Results

Fig. 1 summarizes our findings on how model behavior changes under noisy perturbations and adversarial attacks. Here, we briefly discuss how perturbation type, perturbation amount, model, and embeddings affect model misclassification rate. In addition, we contrast model performance across datasets and report how to mitigate model error rate using ensembling. Detailed analyses are presented in Sec. 5. Key findings are *italicized*.

The effect of perturbation type To assess whether perturbations changed the meaning, we ran a human study on a random sample of 100 perturbed contexts from SQuAD. We found (as expected) that the two annotators we employed could not distinguish char-perturbed text from the original. For word perturbations, the meaning of the context remained in 65% of cases, but annotators noted that sentences were often ungrammatical. For sentence-level perturbations, the meaning remained in 83% of cases.

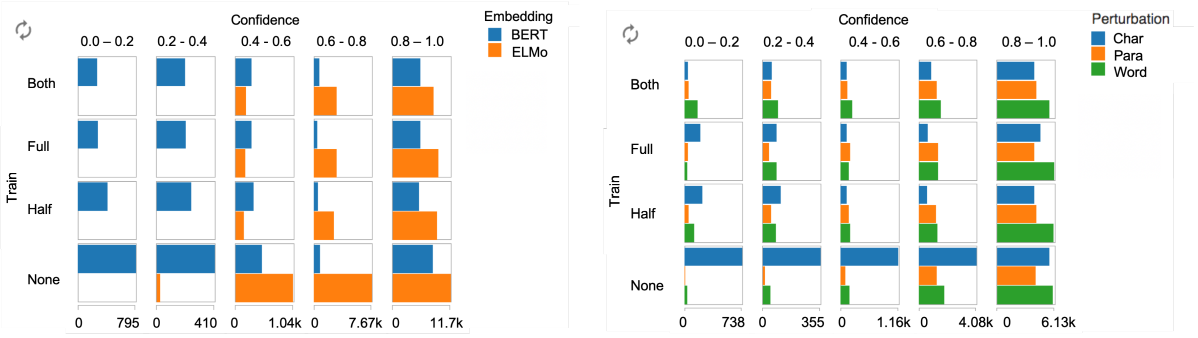
For a model trained on clean data, character perturbations affect the model the most, followed by word perturbations, then paraphrases. To a machine, a single character perturbation results in a completely different word; handling this type of noise is important for a machine seeking to beat human performance. Word perturbations are context independent and can make the sentence ungrammatical.[6] Nevertheless, the context's meaning generally remains coherent. Paraphrase perturbations are most ideal because they retain meaning while allowing more drastic phrase and sentence structure modifications. In Sec. 4.2, we present a more successful adversarially targeted paraphrasing approach.

The effect of perturbation amount Perturbed training data improves the model's performance for character perturbations (1st column of Fig. 1a), likely due to the models' ability to handle unseen words: BiDAF with ELMo utilizes character embeddings, while BERT uses word pieces. Our results corroborate Heigold et al. (2018)'s findings (though on a different task) that *without adversarial training, models perform poorly on perturbed test data, but when models are trained on perturbed data, the amount of perturbed training data does not make much difference.* We do not see statistically significant results for word and paraphrase perturbations (2nd and 3rd columns in each heatmap in Fig. 1). We conclude that perturbing 25% of the words and the non-strategic paraphrasing approach were not aggressive enough.

The effect of model and embedding As shown in Fig. 1a and b, the BERT model had less errors than the ELMo-based model regardless of the perturbation type and amount on SQuAD data. While

---

[6] Future work will address this with language models.



(a) Across models and embedding

(b) Across perturbation types

Figure 2: The effect of perturbation types and embeddings on model behavior measured as high vs. low confidence misclassifications. More robust models should have less high-confidence error or rate (x-axis).

Train	Test	Model Answer
none	char	here”
half	char	Orientalism
full	char	Orientalism
none	word	Orientalism
half	word	behaviourism identities
full	word	The discourse of Orientalism
none	char	Orientalism
half	char	. . . the East as a negative
full	char	Orientalism

Table 3: Example result from response ensembling under the SQuAD ELMo setting. The question is “What was used by the West to justify control over eastern territories?” The answer is “Orientalism”, and in all three settings, the ensemble was correct.

the two models are not directly comparable, our results indicate that the *BERT model is more robust to adversarial attacks compared to ELMo.*

The effect of the data Holding the model constant (Fig. 1b and c), experiments on TriviaQA resulted in more errors than SQuAD regardless of perturbation amount and type, indicating that TriviaQA may be a harder dataset for MC and may contain data bias, discussed below.

4.1 Adversarial Ensembles

Ensemble adversarial training has recently been explored (Tramèr et al., 2018) as a way to ensure robustness of ML models. For each perturbation type, we present results ensembled from the none, half, and full perturbed settings. We tokenize answers from these three models and keep all tokens that appear at least twice as the resulting answer (Tab. 3). Even when all three model answers differ (e.g. in the word perturbation case), ensembling can often reconstruct the correct answer. Nevertheless, we find that this ensembling only helps for TriviaQA, which has an overall higher error rate (bottom row of each figure in Fig. 1).

4.2 Strategic Paraphrasing

We did not observe a large increase in errors with paraphrase perturbations (Fig. 1), perhaps because paraphrasing, unlike the char and word perturbations, is not a deliberate attack on the sentence. Here we experiment with a novel strategic paraphrasing technique that targets specific words in the context and then generates paraphrases that exclude those words. We find the most important words in the context by individually modifying each word and obtaining the model’s prediction and confidence, a process similar to Li et al. (2018). Our modification consists of removing the word and examining its effect on the model prediction. The most important words are those which, when removed, lower the model confidence of a correct answer or increase confidence of an incorrect answer. The Improved ParaBank Rewriter supports constrained decoding, i.e. specifying positive and negative constraints to force the system output to include or exclude certain phrases. We specify the top five important words in the context as negative constraints to generate strategic paraphrases.[7]

We experimented on 1000 instances in the SQuAD dev set as shown in Tab. 4. Our results indicate that *strategic paraphrasing with negative constraints is a successful adversarial attack*, lowering the F1-score from 89.96 to 84.55. Analysis shows that many words in the question are important and thus excluded from the paraphrases. We also notice that paraphrasing can occasionally turn an incorrect prediction into a correct one. Perhaps

---

[7] The number of constraints does not necessarily indicate the number of words that are changed in the context.

Original Paragraph	Strategic Paraphrase
. . . Veteran receiver Demaryius Thomas led the team with 105 receptions for 1,304 yards and six touchdowns, while Emmanuel Sanders caught 76 passes for 1,135 yards and six scores, while adding another 106 yards returning punts.	. . . The veteran earman Demaryius Thomas was leading a team of 1,304 yards and six touchdowns, while Emmanuel Sanders caught 76 passes for 1,135 yards and six scores while he added another 106 yards of punts back.
Question: Who led the Broncos with 105 receptions? **Answer**: Demaryius Thomas (correct) → Emmanuel Sanders (incorrect)	

Table 4: Example of strategic paraphrasing: red indicates the important words, which were used as negative constraints in the paraphrasing; blue indicates changed words in the paragraph.

paraphrasing makes the context easier to understand by removing distractor terms; we leave this for future investigation.

4.3 Model Confidence

In a black-box setting, model confidence is one of the only indications of the model's inner workings. The models we employed do not provide a single confidence value; AllenNLP gives a probability that each word in the context is the start and end span, while the BERT models only give the probability for the start and end words. We compute the model's confidence using the normalized entropy of the distribution across the context words, where n is the number of context words, and take the mean for both the start and end word: $1 - \frac{H_n(\mathbf{s})+H_n(\mathbf{e})}{2}$, where $\mathbf{s}$ and $\mathbf{e}$ are probability distributions for the start and end words, respectively. Low entropy indicates certainty about the start/end location. Since the BERT models only provide probabilities for the start and end words, we approximate the entropy by assuming a flat distribution, dividing the remaining probability equally across all other words in the context.

Comparing confidence across models (Fig. 2a), the *BERT model has lower confidence for misclassifications*, which is ideal. A model should not be confident about errors. Fig. 2b compares confidence across perturbation type. In the *none* training setting, character perturbations introduce the most uncertainty compared to word or paraphrase perturbations. This is expected, since character perturbations result in unknown words. In the adversarial training, word perturbations lead to the highest number of high-confidence errors. Thus, *to convincingly mislead the model to be highly confident about errors, one should use word perturbations.*

5 Robustness Analysis

Here, we do a deeper dive into why models make errors with noisy input. We investigate data characteristics and their association with model errors by utilizing CrossCheck (Arendt et al., 2020), a novel interactive tool designed for neural model evaluation. Unlike several recently developed tools for analyzing NLP model errors (Agarwal et al., 2014; Wu et al., 2019) and understanding ML model outputs (Lee et al., 2019; Poursabzi-Sangdeh et al., 2018; Hohman et al., 2019), CrossCheck is designed to allow rapid prototyping and cross-model comparison to support experimentation.[8]

5.1 The Effect of Question Type, Question and Context Lengths

We examine if models make more errors on specific types of questions in adversarial training, i.e., some questions could just be easier that others. We first examine **question type**:[9] who, what, which, when, where, why, how, and other. The majority of SQuAD questions are *what* questions, while most TriviaQA questions are *other* questions, perhaps indicating more complex questions (Fig. 4a). We see that models usually choose answers appropriate for the question type; even if they are incorrect, answers to *when* questions will be dates or time word spans, and answers to *how many* questions will be numbers. Fig. 4a presents key findings on differences in model misclassifications between two datasets given specific question types. *On the SQuAD dataset, the model finds certain question types, e.g. when and how, easiest to answer regardless of the perturbation type*. Responses to these questions, which generally expect numeric answers, are not greatly affected by perturbations. For TriviaQA, in general we observe more errors across question types compared to SQuAD, i.e. more errors in what, which and who questions.

---

[8] To reproduce our findings, we will release the tool and interactive notebooks upon publication.

[9] Computed as the first word of the question. Many *how* questions are *how many* or *how much*, rather than how in the "in what manner" sense.

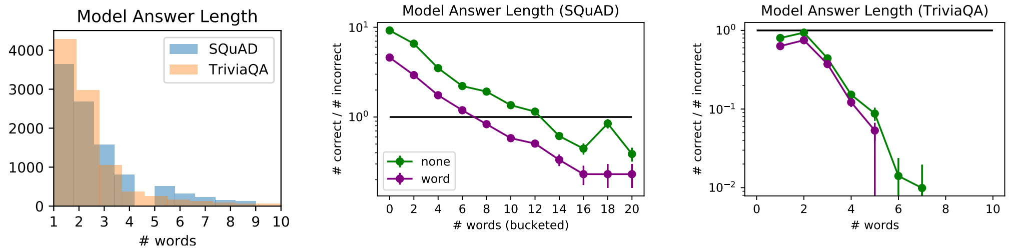
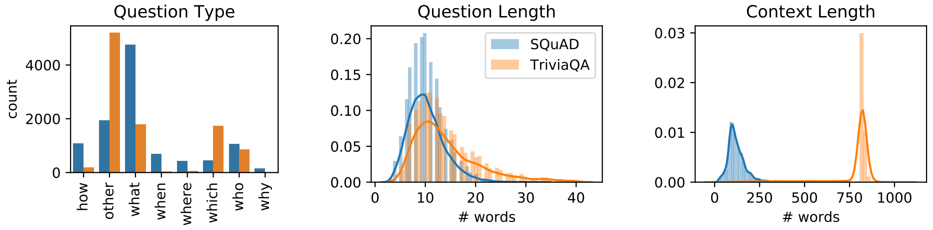
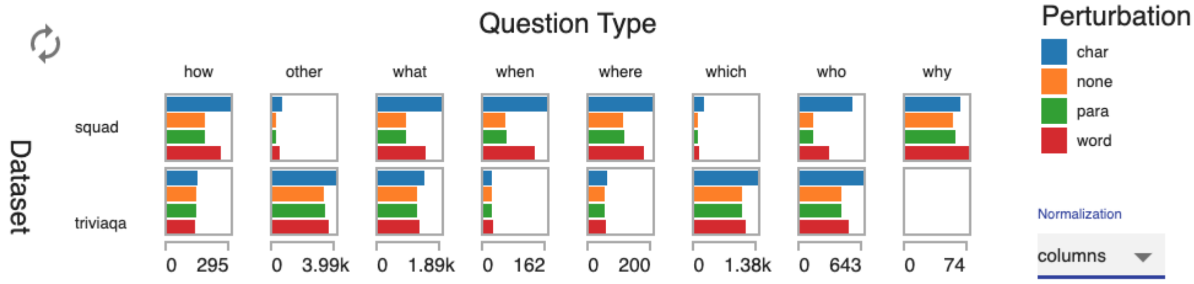


Figure 3: (Left) shows distribution of answer length. (Center) and (Right) show the ratio of correct errors to incorrect errors (log scale y-axis). BERT models were trained on clean data and tested on perturbed data.



(a) Question type distribution (b) Quest. length norm. density (c) Context length norm. density



(d) Error distribution in BERT models across different perturbations and question types

Figure 4: Contrasting MC model errors by question type, and question and context length across datasets.

Regarding **question length**, SQuAD and TriviaQA have similar distributions (Fig. 4b). Both datasets have a mode answer length around 10 words; TriviaQA has a slightly longer tail in the distribution. We did not find question length to impact the error. Regarding **context length**, SQuAD and TriviaQA have vastly differing context length distributions (Fig. 4c), partly due to how the two datasets were constructed (see Sec. 3.4 for details). For both datasets, the error distribution mirrors the context length distribution, and we did not find any relation between model errors and context length.

5.2 The Effect of Answer Length

Our analysis shows that *the length of the model's answer is a strong predictor of model error in the adversarial setting*: the longer the answer length, the more likely it is to be incorrect. Fig. 3 plots the proportion of correct to incorrect answers. We notice a downward trend which is mostly consistent across experimental settings. For both SQuAD and TriviaQA, the models favored shorter answers, which mirrors the data distribution.

5.3 The Effect of Complexity: Annotator Agreement and Reading Level

Here, we examine the effect of task complexity on model performance under adversarial training, using inter-annotator agreement as a proxy for *question* complexity and paragraph readability as a proxy for *context* complexity.

Inter-annotator agreement represents a **question's complexity**: low agreement indicates that annotators did not come to a consensus on the correct answer; thus the question may be difficult to answer. We examine SQuAD, whose questions have one to six annotated answers. In Fig. 5, we present inter-annotator agreement (human confidence) plotted against model confidence over the four training perturbation amounts, looking only at the incorrect predictions. The setting is SQuAD BERT with character perturbation. We observe

Data	Correct	Errors
SQuAD	12.9	13.0
TriviaQA	17.1	17.5

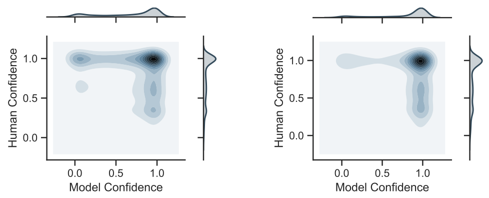
Table 5: Contrasting median readability scores for paragraphs with and without errors across datasets.

that the models are generally confident even when the humans are not, which is noticeable across all perturbation amounts. However, we see interesting differences in model confidence in adversarial training: models trained in the none and half settings have confidence ranging between 0 and 1 compared to the models trained in full and both setting with confidence above 0.8, indicating *training with more perturbed data leads to more confident models*.

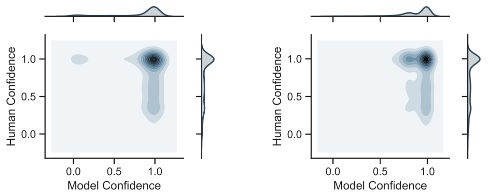
To evaluate the effect of **context complexity**, we use the Flesch-Kincaid reading level (Kincaid et al., 1975) to measure readability. For questions the model answered incorrectly, the median readability score was slightly higher than the median score for correct responses (Tab. 5), indicating that *context with higher reading level is harder for the model to understand*. TriviaQA contexts have higher reading levels than SQuAD.

6 Predicting Model Errors

Our in-depth analysis reveals many insights on how and why models make mistakes during adversarial training. Using the characteristics we analyzed above, we developed a binary classification model to predict whether the answer would be an error, given the model's answer and attributes of the context paragraph. We one-hot-encode categorical features (training amount, perturbation type, question type) and use other features (question length, context length, answer length, readability) as is. For each setting of embedding and perturbation type on SQuAD, we train an XGBoost model with default settings with 10-fold cross validation (shuffled).



(a) None perturbed (b) Half perturbed



(c) Full perturbed (d) Both

Figure 5: The effect of task complexity on model behavior measured as a joint distribution of errors from BERT model on SQuAD using varied amounts of char perturbations (none, half, full and both).

Embedding	Pert.	Majority	F1 score
ELMo	char	0.58	0.70 ± 0.003
ELMo	word	0.54	0.56 ± 0.004
ELMo	para	0.65	0.65 ± 0.008
BERT	char	0.76	0.77 ± 0.008
BERT	word	0.72	0.73 ± 0.006
BERT	para	0.82	0.82 ± 0.006

Table 6: MC error prediction across datasets, embeddings, and perturbation types.

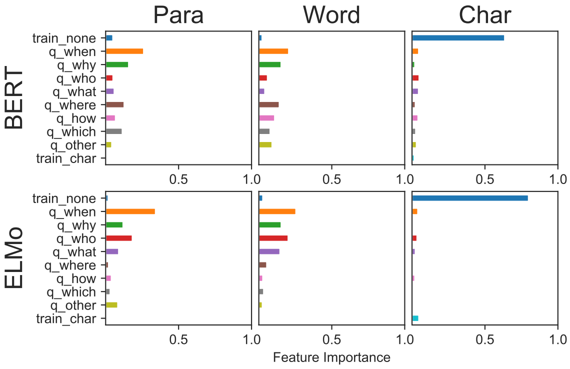


Figure 6: Feature importance when predicting model errors during adversarial attacks.

We present the model's average F1 scores (Tab. 6) and feature importance as computed by the XGBoost model (Fig. 6). We see that performance (micro F1) is better to slightly better than a majority baseline (picking the most common class), indicating that certain features are predictive of errors. Specifically, we find that: *for character perturbations, the fact that the training data is clean is a strong predictor of errors; a model trained on clean data is most disrupted by character perturbations; for word and paraphrase perturbations, question types are important predictors of errors.*

7 Conclusion and Future Work

Our in-depth analysis of neural model robustness sheds light on how and why MC models make errors in adversarial training, and through our error prediction model, we discovered features of the data e.g., question types that are strongly predictive of when a model makes errors during adversarial attacks with noisy inputs. Our results on evaluating the effect of the data e.g., questions and context length will not only explain model performance in context of the data, but will also allow to build future neural models more resilient to adversarial attacks and advance understanding of neural model behavior across a variety of NLU tasks and datasets and its strengths and weaknesses.

For future work, we see many avenues for extension. We plan to experiment with more aggressive and more natural perturbations, and deeper counterfactual evaluation (Pearl, 2019). While recent research has made great strides in increasing model performance on various NLP tasks, it is still not clear what linguistic patterns these neural models are learning, or whether they are learning language at all (Mudrakarta et al., 2018).

More broadly, as AI becomes more entrenched in our lives, AI models need to be held to higher standards including but not limited to accountability (e.g. Wang et al., 2018, 2019, GENIE[10]), fairness (e.g. Saleiro et al., 2018; Bellamy et al., 2018; Bird et al., 2020; Ahn and Lin, 2020, [11];[12];[13]), and transparency (e.g. Lundberg and Lee, 2017; Nori et al., 2019; Hooker et al., 2018; Kokhlikyan et al., 2020; Lundberg et al., 2019; Tenney et al., 2020).

Acknowledgments

The research was performed at Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of the U.S. Government. The authors would like to thank Maria Glenski for helpful discussions, and the reviewers for their feedback.

References

Apoorv Agarwal, Ankit Agarwal, and Deepak Mittal. 2014. An error analysis tool for natural language processing and applied machine learning. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations*, pages 1–5.

Yongsu Ahn and Y. Lin. 2020. Fairsight: Visual analytics for fairness in decision making. *IEEE Transactions on Visualization and Computer Graphics*, 26:1086–1095.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896.

Dustin Arendt, Zhuanyi Huang, Prasha Shrestha, Ellyn Ayton, Maria Glenski, and Svitlana Volkova. 2020. Crosscheck: Rapid, reproducible, and interpretable model evaluation.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *6th International Conference on Learning Representations ICLR*.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

R. Bellamy, K. Dey, M. Hind, Samuel C. Hoffman, S. Houde, K. Kannan, Pranay Lohia, J. Martino, Sameep Mehta, A. Mojsilovic, Seema Nagar, K. Ramamurthy, J. Richards, Diptikalyan Saha, P. Sattigeri, M. Singh, Kush R. Varshney, and Y. Zhang. 2018. Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *ArXiv*, abs/1810.01943.

S. Bird, Miro Dudík, R. Edgar, B. Horn, Roman Lutz, Vanessa Milan, M. Sameki, H. Wallach, and Kathleen Walker. 2020. Fairlearn: A toolkit for assessing and improving fairness in ai.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics)*, pages 4171–4186.

G. W. Ding, Luyu Wang, and Xiaomeng Jin. 2019. advertorch v0.1: An adversarial robustness toolbox based on pytorch. *ArXiv*, abs/1902.07623.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics)*, pages 1634–1647.

---

[10] GENIE `https://genie.apps.allenai.org/`

[11] ML Fairness Gym `https://github.com/google/ml-fairness-gym`

[12] Fairness Indicators `https://github.com/tensorflow/fairness-indicators`

[13] Scikit-Fairness `https://github.com/koaning/scikit-fairness`

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728.

Anthony Y Fu, Xiaotie Deng, Liu Wenyin, and Greg Little. 2006a. The methodology and an application to fight against unicode attacks. In *Proceedings of the second symposium on Usable privacy and security*, pages 91–101. ACM.

Anthony Y Fu, Wan Zhang, Xiaotie Deng, and Liu Wenyin. 2006b. Safeguard against unicode attacks: generation and applications of uc-simlist. In *WWW*, pages 917–918.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6.

Karan Goel, Nazneen Rajani, J. Vig, Samson Tan, J. Wu, Stephan Zheng, Caiming Xiong, Mohit Bansal, and C. Ré. 2021. Robustness gym: Unifying the nlp evaluation landscape. *ArXiv*, abs/2101.04840.

Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309.

Bryce Goodman and Seth Flaxman. 2017. European union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57.

Dou Goodman, Xin Hao, Yang Wang, Yuesheng Wu, Junfeng Xiong, and H. Zhang. 2020. Advbox: a toolbox to generate adversarial examples that fool neural networks. *ArXiv*, abs/2001.05574.

Georg Heigold, Stalin Varanasi, Günter Neumann, and Josef van Genabith. 2018. How robust are character-based word embeddings in tagging and MT against wrod scramlbing or randdm nouse? In *Proceedings of the 13th Conference of the Association for Machine Translation*, pages 68–80.

Fred Hohman, Andrew Head, Rich Caruana, Robert DeLine, and Steven M Drucker. 2019. Gamut: A design probe to understand how data scientists understand machine learning models. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, page 579. ACM.

Sara Hooker, D. Erhan, P. Kindermans, and Been Kim. 2018. Evaluating feature importance estimates. *ArXiv*, abs/1806.10758.

Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. 2019. On the robustness of self-attentive models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1520–1529.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 839–850.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *AAAI*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

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

Narine Kokhlikyan, Vivek Miglani, M. Martín, E. Wang, B. Alsallakh, J. Reynolds, A. Melnikov, Natalia Kliushkina, C. Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch. *ArXiv*, abs/2009.07896.

Gyeongbok Lee, Sungdong Kim, and Seung-won Hwang. 2019. Qadiver: Interactive framework for diagnosing qa models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9861–9862.

Jinfeng Li, Shouling Ji, Tianyu Du, Bao Qin Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *Network and Distributed System Security Symposium*.

Zachary C. Lipton. 2018. The mythos of model interpretability. *ACM Queue*, 16(3):30.

Changwei Liu and Sid Stamm. 2007. Fighting unicode-obfuscated spam. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pages 45–59. ACM.

Scott Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *NIPS*.

Scott M. Lundberg, G. Erion, Hugh Chen, A. DeGrave, J. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and Su-In Lee. 2019. Explainable ai for trees: From local explanations to global understanding. *ArXiv*, abs/1905.04610.

Paul Michel, Xian Li, Graham Neubig, and J. Pino. 2019. On evaluation of adversarial perturbations for sequence-to-sequence models. In *NAACL-HLT*.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, D. Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *EMNLP*.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906.

Maria-Irina Nicolae, M. Sinn, Minh-Ngoc Tran, Beat Buesser, Ambrish Rawat, M. Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, B. Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. 2018. Adversarial robustness toolbox v1.0.0. *arXiv: Learning*.

Tong Niu and Mohit Bansal. 2018. Adversarial over-sensitivity and over-stability strategies for dialogue models. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 486–496. Association for Computational Linguistics.

H. Nori, S. Jenkins, P. Koch, and R. Caruana. 2019. Interpretml: A unified framework for machine learning interpretability. *ArXiv*, abs/1909.09223.

Judea Pearl. 2019. The seven tools of causal inference, with reflections on machine learning. *Commun. ACM*, 62(3):54–60.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2227–2237.

Forough Poursabzi-Sangdeh, Daniel G Goldstein, Jake M Hofman, Jennifer Wortman Vaughan, and Hanna Wallach. 2018. Manipulating and measuring model interpretability. *arXiv preprint arXiv:1802.07810*.

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

Jonas Rauber, R. S. Zimmermann, M. Bethge, and W. Brendel. 2020. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *J. Open Source Softw.*, 5:2607.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865. Association for Computational Linguistics.

Pedro Saleiro, Benedict Kuester, Abby Stevens, Ari Anisfeld, Loren Hinkson, J. London, and R. Ghani. 2018. Aequitas: A bias and fairness audit toolkit. *ArXiv*, abs/1811.05577.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations ICLR*.

Chenglei Si, Ziqing Yang, Yiming Cui, Wentao Ma, Ting Liu, and Shijin Wang. 2020. Benchmarking robustness of machine reading comprehension models. *arXiv preprint arXiv:2004.14004*.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, E. Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and A. Yuan. 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for nlp models. In *EMNLP*.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. 2018. Ensemble adversarial training: Attacks and defenses. In *6th International Conference on Learning Representations ICLR*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP@EMNLP*.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2019. Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 747–763.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, T. Zhang, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2020. Openattack: An open-source textual adversarial attack toolkit. *ArXiv*, abs/2009.09191.

Wei Emma Zhang, Quan Z. Sheng, and Ahoud Abdulrahmn F. Alhazmi. 2019. Generating textual adversarial examples for deep learning models: A survey. *CoRR*, abs/1901.06796.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *6th International Conference on Learning Representations ICLR*.

Hidden Biases in Unreliable News Detection Datasets

Xiang Zhou[1], **Heba Elfardy**[2], **Christos Christodoulopoulos**[2]
Thomas Butler[2], **Mohit Bansal**[1]
[1]UNC Chapel Hill [2]Amazon
`{xzh, mbansal}@cs.unc.edu`
`{helfardy, tombutl}@amazon.com, chrchrs@amazon.co.uk`

Abstract

Automatic unreliable news detection is a research problem with great potential impact. Recently, several papers have shown promising results on large-scale news datasets with models that only use the article itself without resorting to any fact-checking mechanism or retrieving any supporting evidence. In this work, we take a closer look at these datasets. While they all provide valuable resources for future research, we observe a number of problems that may lead to results that do not generalize in more realistic settings. Specifically, we show that selection bias during data collection leads to undesired artifacts in the datasets. In addition, while most systems train and predict at the level of individual articles, overlapping article sources in the training and evaluation data can provide a strong confounding factor that models can exploit. In the presence of this confounding factor, the models can achieve good performance by directly memorizing the site-label mapping instead of modeling the real task of unreliable news detection. We observed a significant drop (>10%) in accuracy for all models tested in a clean split with no train/test source overlap. Using the observations and experimental results, we provide practical suggestions on how to create more reliable datasets for the unreliable news detection task. We suggest future dataset creation include a simple model as a difficulty/bias probe and future model development use a clean non-overlapping site and date split.[1]

1 Introduction

The proliferation of unreliable news is widely acknowledged (Del Vicario et al., 2016; Lazer et al., 2018; Vosoughi et al., 2018), and its identification is a socially important problem. In this work we use the label unreliable news as a broad term for all unverifiable and misleading news content, regardless of whether the content is malicious (targeted misinformation) or not. Accordingly, while specific definitions vary in different datasets used in this work, we refrain from using the term "fake" since identifying the intent of the author(s) is beyond the scope of this work. To mitigate the problem of surfacing unreliable news content, various websites (e.g., PolitiFact[2], Media Bias/Fact Check (MBFC)[3], GossipCop[4], etc.) determine the reliability of news by manually fact-checking the important claims in given news articles. Beyond requiring investigative expertise, manual fact-checking is time-consuming and is thus limited to only a small set of selected news articles.

Recent research has explored automating this process using machine learning methods to automatically determine news veracity (Pérez-Rosas et al., 2018; Baly et al., 2018; Nie et al., 2019; Wright and Augenstein, 2020). These efforts were made possible due to the availability of large-scale unreliable news detection datasets (Horne et al., 2018b; Shu et al., 2017; Wang, 2017). In our work, we examine if these datasets accurately reflect the real difficulty of this task or if there are any hidden biases in the datasets. Specifically, we study different methods of dataset construction (e.g., how the data was collected, how the data was split, etc.) and show that the assessed difficulty of the task is sensitive to how carefully different factors are considered when building and using these datasets.

Our investigation begins with data collection procedures: we look at the source of news stories (news outlets, social media, fact-checking websites, etc.) as well as the annotation process (number of

---

[1] Our code is publicly available at `https://owenzx.github.io/unreliable_news`

[2] `https://www.politifact.com/`

[3] `https://mediabiasfactcheck.com/`

[4] `https://www.gossipcop.com/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2482–2492
April 19 - 23, 2021.

Data Collection	Dataset Construction	Experiment Design
1. Collect from less biased or unbiased resources (e.g. original news outlets). (Sec. 3.2)	1. Examine the most salient words to check for biases in the datasets. (Sec. 3.2)	1. Apply debasing techniques when developing models on biased datasets. (Sec. 3.2)
2. Collect from diverse resources (in terms of sources, topics, time, etc.). (Sec. 3.2, 4)	2. Run simple BoW baselines to check how severe the bias is. (Sec. 4)	2. Check the performance on sources/dates not in your training set. (Sec. 4.2, Sec. 4.3)
3. Collect precise article-level labels if possible. (Sec. 3.1)	3. Provide train/dev/test splits with non-overlapping source/time. (Sec. 4.2, 4.3)	3. Check the performance on sources with limited examples. (Sec. 4.4)
		4. Test your model on multiple complementary datasets (e.g. with different domains, styles, etc.). (Sec. 3.2, 4.4)

Table 1: Suggestions for data collection, dataset construction and experiment design for unreliable news research.

labels, granularity of labels, article- or site-level annotation). We discuss the pros and cons of each approach and point out some hidden pitfalls. Using FakeNewsNet (Shu et al., 2017) as an example, we demonstrate how selection biases in data collection can lead to undesired biases in the created datasets.

Moving beyond data collection, we examine two commonly applied ways of splitting the dataset for training and testing that help the model achieve high performance without correctly modeling the task. Specifically, we show that using a disjoint set of sites/news outlets for training and test data significantly decreases the models' performance (>10%) and that the drop in performance is related to how similar (or dissimilar) the sites in both sets are (reflected by various site-level distributional distance metrics including L2, cos, EMD, etc.). Additionally, we also examine the effect of time overlap between both train and test sets. We observe that different news outlets are likely to have similar content in a small time window (i.e., the same story gets covered by multiple outlets within a day or a few days period). While we do not find any evidence that the studied models exploit this factor, we nevertheless suggest that future datasets are split both by time and site/news outlet.

In summary, our main contributions are: (1) showing how data collection procedures can lead to systematic biases in unreliable news datasets, (2) demonstrating how confounding factors——such as site/news outlet and time——in these datasets can degrade their quality and lead to underestimating the difficulty of the task, and finally (3) suggesting possible mechanisms to avoid these biases and confounding factors when building new datasets. To facilitate future research, we also provide a list of practical suggestions for data collection, dataset construction, and experiment design in Table 1.

2 Related Work

Unreliable News Detection. Unreliable news detection and other news veracity related tasks have been receiving an increasing focus as news sources have become more accessible in recent years. A lot of effort has been put into collecting high-quality datasets. Wang (2017); Shu et al. (2017) collected manually labeled statements or news articles from fact-checking websites. The NELA datasets (Horne et al., 2018b; Nørregaard et al., 2019; Gruppi et al., 2020) scrape news articles directly from news outlets and use the manually annotated labels from Media Bias/Fact Check (MBFC) as site-level annotations. Social media is also a popular resource for collecting news stories (Nakamura et al., 2020; Santia and Williams, 2018; Mitra and Gilbert, 2015). Researchers have also collected datasets for various related topics, such as rumor detection (Kwon et al., 2017; Ma et al., 2016), and propaganda detection (Da San Martino et al., 2020; Barrón-Cedeno et al., 2019). Besides classifying the veracity of news articles, researchers have also explored related problems, such as predicting the reliability of news sites (Baly et al., 2018), identifying fact-check worthy sentences (Wright and Augenstein, 2020), among other tasks. Several recent papers also focus on measuring the trustworthiness of single statements (Wang, 2017; Pomerleau and Rao, 2017; Alhindi et al., 2018). In this work, we focus on article-level classification because of its relevance to applications, like news feeds, that operate at the article level.

Dataset	Size	Article Source	Label Type
NELA[5]	136K/713K/1.12M	News outlets	Site-level
FakeNewsNet[6]	603K	Fact-checking websites	Article-level
r/Fakeddit[7]	1.06M	Social Media (Reddit)	Site-level

Table 2: Statistics and properties of three recent large-scale unreliable news datasets. The three statistics of NELA dataset sizes correspond to its three versions released in 2017, 2018 and 2019, respectively.

Pitfalls in Data Collection. Datasets collected through crowd-sourcing or scraping the Internet have the advantage of much better scalability compared to expert-annotated datasets. However, these automatic processes are prone to hidden pitfalls. Gururangan et al. (2018); Poliak et al. (2018) show that crowd-sourcing "Natural Language Inference" datasets leads to various dataset biases. Similar observations have been made for "Fact Verification" datasets (Schuster et al., 2019). Splitting data—for training, testing, and validation——is another important procedure in creating datasets that can lead to several problems. For example, Geva et al. (2019) show that models may just learn the patterns of certain annotators in a random split. Lewis et al. (2020b) demonstrated a significant overlap in current open-domain QA datasets. When present, these unexpected biases or overlaps in datasets can significantly undermine the utility of a dataset and lead to deceptively promising results that are in part due to artifacts of flaws in the dataset rather than successfully modeling the intended task.

Automated Fact Checking for Statements. Automated fact checking is an important task closely related to unreliable news detection, yet is constructed in a more controlled manner. This task focuses on strictly judging the factuality of one single statement instead of an entire article. Vlachos and Riedel (2014) first constructed a dataset with 106 claims from fact-checking websites with paired labels. FEVER (Thorne et al., 2018) is currently the largest scale fact-verification dataset, where 185,445 claims were generated by modifying sentences from Wikipedia. Both the altered claims and the ground truth supporting evidence are included in the dataset. Existing effective approaches for fact-verification include self-attention based networks (Nie et al., 2019), large-scale pretrained transformers (Soleimani et al., 2020), neural retrieval methods (Lewis et al., 2020a), and reasoning on semantic-level graphs (Zhong et al., 2020).

3 Unreliable News Datasets

Collecting high-quality datasets plays an important role in automatic unreliable news detection research. Here we review dataset collection strategies used in constructing recent datasets and point out some hidden pitfalls in these procedures.

3.1 Data Collection Strategies

Unreliable news detection is usually formalized as a classification task. Accordingly, constructing a dataset requires collecting pairs of news articles and labels.

News Articles: Each individual news outlet has its own website where news articles are published. The easiest way to collect a large number of these articles is to simply scrape these websites. Manual annotation or some other mechanism must then be incorporated in order to collect the corresponding labels for each article. Another common way to collect articles is through fact-checking websites. While this approach provides both articles and article-level labels, it normally only provides a limited set of articles. Additionally, scraping these fact-checking websites can lead to additional selection bias in the dataset as highlighted in Section 3.2.

One other recent trend is collecting posts and corresponding labels from social media (Nakamura et al., 2020; Santia and Williams, 2018; Mitra and Gilbert, 2015). While large-scale datasets can be collected through such an approach, they are often noisier than those collected through traditional news sources, due to a more casual use of language, and a heavier dependency on the context.

News Labels: The largest challenge in collecting these datasets lies in collecting labels. Manually checking the factuality (or reliability) and bias of a single article is time-consuming and requires non-trivial expertise. Modeling such a task through a crowd-sourcing framework is diffi-

---

[5] `dataverse.harvard.edu/dataverse/nela`

[6] `github.com/KaiDMML/FakeNewsNet`

[7] `github.com/entitize/Fakeddit`

FakeNewsNet		r/Fakeddit	
Positive Features	Negative Features	Positive Features	Negative Features
season	trump	psbattle	clicks
at	brad	says	colorized
2018	pitt	sues	2018
the	jenner	accused	2019
awards	jennifier	sells	mrw

Table 3: Top five most salient features in the FakeNewsNet dataset and the r/Fakeddit dataset. The features are the highest weighted Bag-of-Word features learned by a Logistic Regression model.

News Outlets	Daily Mail
Site Label	Unreliable
Dates	2018/09/06
Title	Roy Moore sues Sasha Baron Cohen
Article	Failed Senate candidate Roy Moore is suing comedian Sacha Baron Cohen for \$95 million for tricking him into appearing on his Showtime program 'Who is America?' Moore, whose bid for the Alabama failed in the wake of claims he molested a 14-year-old, filed the lawsuit in Washington DC on Wednesday...

Table 4: An example showing a reliable news article from the "*Daily Mail*" site which has a "Low" factual reporting rate on MBFC. Despite coming from a source with low reliability score, the shown article is reliable and very similar to the content on sites with high reliability scores (such as *"BBC"* and *"The Week UK"*) on the same date.

cult. As such, current research datasets almost exclusively rely on existing resources. As discussed earlier, these resources either provide article-level or site-level labels. **Article-level** labels are only available through a few fact-checking websites such as PolitiFact, GossipCop, etc., but the scale is limited since generating these labels is time-consuming and costly. **Site-/Outlet-level** labels, on the other hand, available through websites such as MBFC, provide manual labels for each site/outlet. These websites often assign reliable/unreliable or biased/unbiased labels to each news outlet. Many datasets for unreliable news detection assign these site-level labels to all articles in a given site. While these weak or distant labels are not always accurate (one example is shown in Table 4) , they provide an easy way to create large-scale datasets. In Table 2, we highlight three recent large-scale unreliable news datasets along with their data collection procedure.

3.2 Dataset Selection Biases

Datasets annotated without expert verification (e.g., through crowdsourcing, automatic web scraping, etc.) can have some undesired properties that undermine their quality (Gururangan et al., 2018; Poliak et al., 2018; Schuster et al., 2019). In the following analysis, we choose the FakeNewsNet dataset (Shu et al., 2017) as a representative example.

We first examine the most salient features in the dataset. To achieve this, we train a Logistic Regression (LR) model on the titles of FakeNewsNet using Bag-of-Words features and show the word features with the highest weights for each class in Table 3.[8] The features in the table show clear patterns: the top-features for the reliable (positive) class are either stop words (e.g., 'at', 'the', etc.) or words presumably carrying neutral semantics (e.g. 'season', '2018', 'awards', etc.) while the top features for the unreliable news (negative) class are mostly celebrity names. Using this basic model, we achieve an accuracy of $\sim$78% , while using a BERT-based model that uses both the article and title as input only achieves an incremental improvement yielding an accuracy of 81% (see Sec. 4.1 for detailed model descriptions). By examining the articles in the dataset, we attribute this to the selection bias exhibited by fact-checking websites. Most unreliable (negative) articles contain click-bait titles mentioning celebrities, while reliable sources usually have less sensational titles with fewer mentions of celebrities and more diverse keywords.

Another potential problem is the articles' retrieval framework. FakeNewsNet uses Google search to retrieve the original news article (Shu et al., 2017). Internet search engines have proprietary news ranking and verification processes, which means that even when using the original title and source of a given article, the search results

---

[8] We also calculated the PMI between the label and word features as suggested by Gururangan et al. (2018) and found the two lists to be very similar.

Label Resource	GossipCop
Title	NYC terror attack: Celebrities react on social media
Article	Celebrities are sending their love and support to New York on social media following a terror attack that left eight people dead Tuesday when a truck plowed down pedestrians on a bicycle path near the World Trade Center in Lower Manhattan...
Label	Unreliable
News URL	`tinyurl.com/yxhvdne6`

Table 5: One example from the FakeNewsNet dataset where it is difficult for the article content to support the label. This article contains celebrities' reactions after a terrorist attack. While the article itself does not look like a standard news piece, the reactions in the article are all paired with tweets, so the unreliable label seems to be inconsistent.

might prioritize specific sites over others leading to inaccurate data collection. While Shu et al. (2017) propose several heuristics to handle these problems, it is unlikely that this noisy process is completely fixed. As a result, we find a few mis-matched title-content pairs where the retrieved article cannot support the label, hence making the example confusing. We show one example with a questionable label in Table 5, where we suspect the inconsistency is due to the noisy retrieval step.

Finally, the informal nature of user-generated content on social media may be the source of additional biases. In our preliminary experiments, we found that in r/Fakeddit dataset, a simple Bag-of-Words(BoW)-based logistic regression model can reach equal—or even better—performance than the reported BERT-based models (86.91% vs. 86.44% in the text-only two-way classification setting), hinting at the strong correlation between the label and lexical inputs. This is also reflected in the equally confusing most salient features in this dataset shown in Table 3.

Since different collection procedures and data resources will lead to different problems, there is no uniform solution to producing a completely bias-free dataset. However, one good test is to check the performance of a simple model such as a BoW-based linear model. By analyzing the features learned by the simple model as well as measuring the gap between the performance of a state-of-the-art system and the simple model, one can get a hint of the dataset quality. Unreasonable features, together with small performance gaps, may reveal unwanted biases in the dataset. In practice, we also suggest that when developing models using biased datasets to use debiasing techniques (e.g. Schuster et al. (2019)).

4 Dataset Split Effect

In this section, we study the effect of time and site/outlet overlap between the training and the evaluation set on the model's performance and show how these confounding factors can impact it.

4.1 Baseline Models & Experimental Setup

In the following experiments, we use two models: a logistic regression baseline and a state-of-the-art large-scale pretrained Transformer-based model (RoBERTa; Liu et al. (2019)).

Logistic Regression (LR): We use scikit-learn's (Pedregosa et al., 2011) implementation of Logistic Regression along with TFIDF-based Bag-of-Words features. We add L2 regularization to the model with a regularization weight of 1.0 and train the model using L-BFGS. In our experiments, the LR model uses only the title (and not the article body) as the input.

RoBERTa: Our implementation is based on the Transformers library (Wolf et al., 2019) and AllenNLP (Gardner et al., 2017). We use RoBERTa in two different ways, one takes only the title as the input, the other takes both the title and the article content as the input and formalizes the task as pairwise sentence classification. Specifically, we concatenate the title and the article content with a `[SEP]` token in the middle and use different token type embeddings to differentiate between the title and the content. Articles are truncated to fit the 512-token length limit. In the title-only setting, the batch size is set to 32, the learning rate is set to 5e-5, and the model is trained for 3 epochs. In the article+title setting, the batch size is set to 8, the learning rate is set to 2e-5, and the model is trained for 10 epochs. These hyperparameters are set empirically, and our preliminary experiments show that the results are not sensitive to different settings of these hyperparameters.

Datasets: Here, our analysis focuses on the 2018 version of the NELA dataset (Horne et al., 2018b). Unlike FakeNewsNet, NELA gathers news directly from news outlets, so the influence of selection bias is insignificant. Thus we focus our analysis on

Model	Input	Random Split	Source Split (Article)	Source Split (Site)
Majority	/	50	50	69.29 (0.56)
LR	Title	77.45	67.18 (4.13)	79.28 (5.27)
RoBERTa	Title	85.22	70.40 (4.28)	87.83 (10.44)
RoBERTa	Title+Article	96.94	80.36 (11.91)	85.14 (8.00)

Table 6: Accuracy on validation sets with different split strategies. For “Source Split’, we report the mean and standard deviation (in parentheses) of five different runs. The last column shows the aggregated site-level accuracy.

other potentially confounding factors in the dataset. We use the latest aggregated site-level labels provided in NELA-GT-2019 (Gruppi et al., 2020) and report both the article- and site-level accuracy. For article-level accuracy, we assign the site-level label to all articles from that news outlet and calculate per-article accuracy. For the Source (Site) Split setting (with no overlap between training and evaluation sites), we also report the site-level accuracy: we aggregate the predictions over individual articles for a given outlet and use the majority prediction as the site-level prediction. We use a balanced label distribution for all dataset splits.

The results in the third column of Table 6 show the models’ performance on the random split, which is the default split method used in most papers, e.g. (Nakamura et al., 2020; Horne et al., 2018a). As the results show, even the simplest logistic regression model achieves an accuracy of over 77% whereas the RoBERTa model using both title and the news article as the input reaches almost 97% accuracy.

4.2 Effect of Split by Source

For this experiment, instead of using the standard random split of all the news articles in the dataset, we first randomly split all the sites in the dataset into three disjoint sets (train/dev/test) before adding all articles from each site to their assigned set (train, dev or test). We believe this setup is closer to real-world tasks. For instance, in order to block all unreliable news sources, one simple—yet useful—approach is to maintain a list of questionable sources. All the news from those sources will be automatically blocked. In this setting, the only remaining task is classifying sources with no or very few annotated examples. As the results in Table 6 show, there is a significant drop in performance for all the models when compared to the random split. The logistic regression model’s performance drops from 77.5 to 67.2%, and even the more powerful RoBERTa model with both title and article as input drops from 96.9 to 80.4%, demonstrating the task’s significantly increased difficulty. While aggregating article-level results to site-levels can significantly improve the accuracy, we also see a plateauing trend of the performance where adding the article as additional input brings no further improvement to the RoBERTa model. Since we subsample the original dataset and balance the number of news articles for each label, the majority baseline (at the article level) is always 50%. But the site-level majority baseline is well above random (69.29%). While a new 50% majority baseline can be achieved by re-subsampling the dataset, the current number also indicates a severe imbalance of dataset size between reliable/unreliable sites which can—potentially—be exploited by the models.

Model	Input	Gold Label	Rand. Label
Majority	/	50	50
LR	Title	77.45	66.29
RoBERTa	Title	85.22	74.37
RoBERTa	Title+Article	96.94	95.04

Table 7: Article-level accuracy for the random label experiments compared to gold site labels.

Random Label Experiments: For this experiment, we use the original random split strategy. However, we permute all the site-level labels randomly. Hence each label no longer represents the reliability of the site, and is just an arbitrary feature of the site itself. Therefore, the only way for the models to achieve good performance on this task is to memorize the arbitrary site-label mapping. The results in Table 7 show that the models achieve very high accuracy with the more powerful RoBERTa model with both title and article showing only ~2% accuracy loss when compared to the true labels. These results demonstrate the models’ ability to memorize random site-labels, and the similarity between these results and the results on the random splits suggest that the models are bypassing the real task of reliable/unreliable news classification and are just memorizing the site identities.

Distance	Top 10 Sites	Bottom 10 Sites
l2	11.59	5.79
cosine	210.72	82.91
MMD	7.78	4.20
CORAL	29.07	14.95

Table 8: Average similarity score between sites in the evaluation and training sets.

Performance Variance and Site Similarity Analysis: Another interesting observation from the results in Table 6 is that while the performance on every random split is fairly stable, the performance is much more unstable with respect to splitting by source. For example, the RoBERTa (Title+Article) model results have a standard deviation larger than 10 points, with the highest accuracy reaching over 90% and the lowest one below 60%.

One potential factor behind the varying performance is the heterogeneity of different news sources (sites). News sites that are similar to those in the training set could be much easier to classify than sites with completely different styles or content. In this case, even when splitting by site, correlations between the content of similar sites in the training and evaluation sets may drive the generalization performance. To assess this hypothesis, we measure the dependence on the distances between sites in the training and evaluation sets and the model performance at the site level in the evaluation set. Given a set s in the evaluation set, we measure its similarity to all the sites in the training set $t \in S_{train}$. Below we show that higher accuracy on the site s is associated with a higher similarity between s and sites in the training set with the same label $t \in S_{same}$, providing evidence in favor of our hypothesis.

In order to measure the similarity between different sites, we take the representation learned by the RoBERTa model as the representation of the article with a focus on its reliability. Since the RoBERTa model feeds the whole sentence into the multi-layer transformer architecture and feeds the representation of `[CLS]` token to the downstream classifier (Devlin et al., 2019; Liu et al., 2019), we use the same `[CLS]` representation as the representation for the whole title+article input.

For similarity-metrics between sites, we follow Guo et al. (2020) and calculate the l2-distance, cosine distance, MMD (maximum mean discrepancy) distance (Gretton et al., 2012; Li et al., 2015) and the CORAL (correlation alignment) distance (Sun and Saenko, 2016; Sun et al., 2016). Following Guo et al. (2020), the l2 and cosine distances are calculated by first averaging all the example representations to get the site representation and calculating the distance between site representations; the MMD distance is calculated using an unbiased finite sample estimate from Li et al. (2015); and the CORAL distance is calculated by $D_{CORAL} = \frac{1}{4d^2}\|C_s - C_t\|_F^2$, where d is the feature dimension, C_s and C_t are the co-variance of two sets and $\|\cdot\|_F^2$ is the squared matrix Frobenius norm. To simplify our analysis, we filter out all the sites containing less than 100 examples (assuming the articles from these sites are too few to significantly influence the model). For every site in the evaluation set s, we calculate its distance with respect to every different site t in the training set, and then compare its minimum distance w.r.t the subset of sites with the same gold label S_{same} and the subset of sites with the opposite label S_{oppo},

$$sim\_score_s = \frac{\min\limits_{t \in S_{oppo}} \{dist(s,t)\}}{\min\limits_{t \in S_{same}} \{dist(s,t)\}}$$

We compute this ratio using all four distances above for the top and bottom 10 sites in the evaluation datasets (ranked based on their accuracy with RoBERTa) and report the mean over all the sites and over all five different random splits in Table 8. The top 10 sites always have a much larger similarity score than the bottom 10 sites, indicating that they have a much larger similarity with sites in the training sets with the same label. This trend holds across all of the distance metrics. The sensitivity of performance on the site similarity raises additional concerns about how the results in Table 6 may generalize in real-life. As newly emerged unreliable sites are likely to behave differently from old sites, the model's performance may be on the lower end of the variance.

As a natural extension, we also explored building a model that directly optimizes these site-level distance metrics in order to have better site-level generalization performance. However, in our preliminary results, our model does not show significant improvement from the baseline models. This can also hint at the fact that it is very difficult for these models to extract features that are useful to the task of reliable/unreliable news classification itself and instead learn site-specific features.

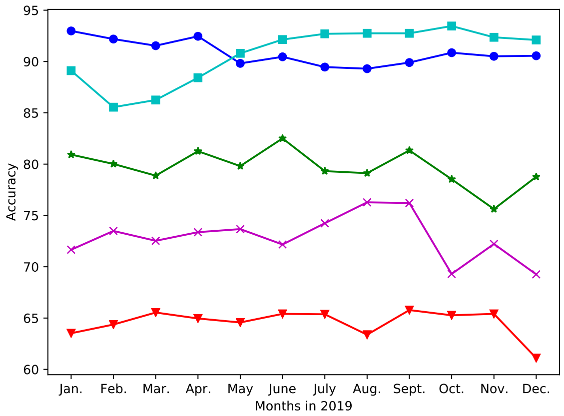


Figure 1: Accuracy of RoBERTa models trained on NELA-GT-2018 and tested on articles from the 12 months covered in the NELA-GT-2019 dataset. The five different lines in the figure represent models trained using five different random site splits.

4.3 Effect of Split By Time

Another potentially important factor to consider while creating train/test/dev splits for a news-based dataset is time. As news-worthy events happen everyday, multiple news articles from different outlets can report the same event. For example, in the NELA 2018 dataset (Nørregaard et al., 2019), within a period of two days (from 2018/10/01 to 2018/10/02), there are more than 100 news articles from over 60 sources about the US-Canada-Mexico trade accord. Therefore, by remembering the content of the event from one article, the model can easily predict the label for any related news article.

To test the effect of time, we examine the model's performance on news articles from a temporally disjoint dataset. Specifically, since all our models are trained on the NELA-GT-2018 (Nørregaard et al., 2019), we use the NELA-GT-2019 (Gruppi et al., 2020) as the evaluation dataset. We split the news articles in 2019 into twelve months and plot the performance trend in Figure 1. We can see that, unlike the significant performance drop in the source split experiments, we do not observe a clear correlation between the performance and the length of the time gap. Therefore, at least for the current models and datasets, splitting by time does not significantly influence the current results. This finding may result from that the fact that the model is not memorizing the exact events in the training set (this is not limited to the unreliable news domain), or it could be attributed to the noise in the training set (similar events can be reported both in reliable and unreliable sources). However, we do have to point out that our current observation only holds for the current models, and it is possible for more powerful models to memorize all events. In addition, the widest time gap tested here is still within a couple of years, which is still a relatively short time in terms of news events. A longer time gap (or a major event such as COVID-19) may lead to different behavior by the models. So in practice, we nonetheless suggest splitting datasets by time to avoid these issues.

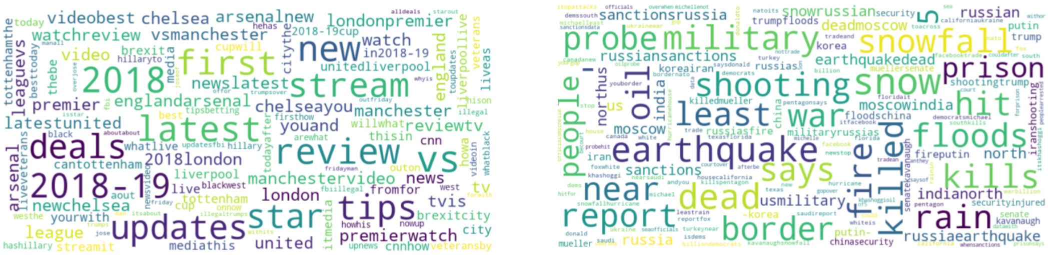
4.4 Error Analysis

Here, we conduct an error analysis to see how the model performs with respect to the variation of some other factors of practical interest, such as topic and site size.

The Influence of Topic in Article-Level Prediction: In order to gain better insight on the performance drop in the source split experiments, we perform a deeper investigation of the numbers in Table 6. We first check whether the models show different performance on different topics. To get a high-level understanding of what the topics are, we look at the titles of articles in the evaluation set and calculate words with the highest PMI with the accuracy of prediction of the RoBERTa model. We then use these PMI values as weights and plot the word cloud figures in Figure 2. In the word cloud of correct predictions, we observe many words related to sports events, while words in the incorrect predictions cloud mostly appear in political news. This is not surprising since there is much more of an incentive to interfere with political news than sports news — making the need for more robust models even more pressing for real-world applications.

The Influence of Size in Site-Level Prediction: Finally, we examine the effect of prediction aggregation from article-level to site-level. Unlike in current datasets where most sites can have hundreds or even thousands of articles, a newly-emerged news outlet waiting for classification may only have a very limited number of articles. Accordingly, while in Table 6, we see a general improvement of the aggregation, it is also important to check the aggregation effect when the number of articles in a given site is small.

In Figure 3 we plot the performance of 5 different runs of the RoBERTa (Title+Article) model



(a) Word cloud of article titles with correct predictions.

(b) Word cloud of article titles with incorrect predictions.

Figure 2: Word cloud of article titles. The words with highest PMI to the prediction correctness of the RoBERTa model are selected.

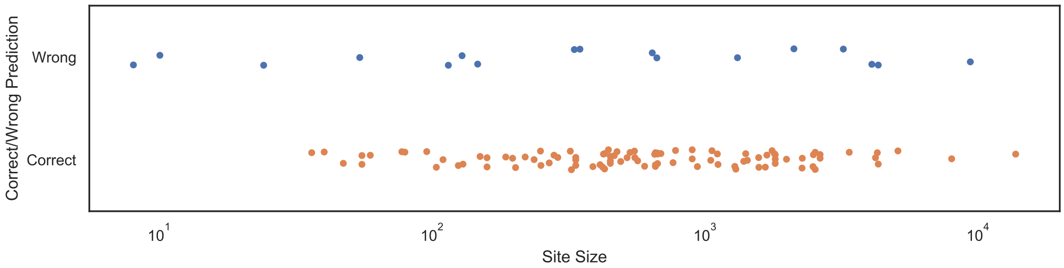


Figure 3: Site-level prediction accuracy of the RoBERTa (Title+Article) model vs. numbers of article in the site (in all five random runs). Blue circles denote wrong predictions and red circles denote correct predictions.

against the number of articles on a given site. We can see that the performance is worse when the size of the site is less than 100, demonstrating the difficulty of predicting the reliability of a site given limited resources. It is also surprising to see a significant number of errors even when the site size is over 1000. This indicates the limitation of simply aggregating the site-level prediction at test-time. Capturing the article-site hierarchy in a better way is a potential future research direction.

5 Conclusion

In this paper, we took a closer look at current large-scale unreliable news detection datasets. We studied their collection procedures and dataset split strategies, and pointed out important flaws in the current approaches. Specifically, we demonstrated that selection bias in dataset collection that often leads to undesired and significant artifacts in these datasets; highlighting confounding factors (e.g., article source, time) in news datasets that can lead to underestimating the difficulty of the task. Finally we provide suggestions on how to better create and process such datasets in the future. We hope our work leads to more high-quality news datasets and that it inspires further work in this direction.

Acknowledgments

We thank the reviewers for their helpful comments. XZ interned at Amazon. This work was also supported by ONR Grant N00014-18-1-2871, DARPA YFA17-D17AP00022, and DARPA KAIROS Grant FA8750-19-2-1004. The views contained in this article are those of the authors and not of the funding agency.

References

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is your evidence: Improving fact-checking by justification modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium. Association for Computational Linguistics.

Ramy Baly, Georgi Karadzhov, Dimitar Alexandrov, James Glass, and Preslav Nakov. 2018. Predicting factuality of reporting and bias of news media sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3528–3539.

Alberto Barrón-Cedeno, Israa Jaradat, Giovanni Da San Martino, and Preslav Nakov. 2019. Proppy: Organizing the news based on their propagandistic content. *Information Processing & Management*, 56(5):1849–1864.

Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeno, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. *arXiv e-prints*, pages arXiv–2007.

Michela Del Vicario, Alessandro Bessi, Fabiana Zollo, Fabio Petroni, Antonio Scala, Guido Caldarelli, H Eugene Stanley, and Walter Quattrociocchi. 2016. The spreading of misinformation online. *Proceedings of the National Academy of Sciences*, 113(3):554–559.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform. *arXiv:1803.07640*.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

Maurício Gruppi, Benjamin D Horne, and Sibel Adalı. 2020. Nela-gt-2019: A large multi-labelled news dataset for the study of misinformation in news articles. *arXiv preprint arXiv:2003.08444*.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits. In *AAAI*, pages 7830–7838.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL-HLT (2)*.

Benjamin D Horne, William Dron, Sara Khedr, and Sibel Adali. 2018a. Assessing the news landscape: A multi-module toolkit for evaluating the credibility of news. In *Companion Proceedings of the The Web Conference 2018*, pages 235–238.

Benjamin D Horne, William Dron, Sara Khedr, and Sibel Adali. 2018b. Sampling the news producers: A large news and feature data set for the study of the complex media landscape. In *Proceedings of the International AAAI Conference on Web and Social Media*.

Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor detection over varying time windows. *PloS one*, 12(1):e0168344.

David MJ Lazer, Matthew A Baum, Yochai Benkler, Adam J Berinsky, Kelly M Greenhill, Filippo Menczer, Miriam J Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, et al. 2018. The science of fake news. *Science*, 359(6380):1094–1096.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020a. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2020b. Question and answer test-train overlap in open-domain question answering datasets. *arXiv preprint arXiv:2008.02637*.

Yujia Li, Kevin Swersky, and Rich Zemel. 2015. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *25th International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 3818–3824. International Joint Conferences on Artificial Intelligence.

Tanushree Mitra and Eric Gilbert. 2015. Credbank: A large-scale social media corpus with associated credibility annotations. In *ICWSM*, pages 258–267.

Kai Nakamura, Sharon Levy, and William Yang Wang. 2020. Fakeddit: A new multimodal benchmark dataset for fine-grained fake news detection. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6149–6157.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6859–6866.

Jeppe Nørregaard, Benjamin D Horne, and Sibel Adalı. 2019. Nela-gt-2018: A large multi-labelled news dataset for the study of misinformation in news articles. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 630–638.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. 2018. Automatic detection of fake news. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3391–3401, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191.

Dean Pomerleau and Delip Rao. 2017. The fake news challenge: Exploring how artificial intelligence technologies could be leveraged to combat fake news. *Fake News Challenge*.

Giovanni C Santia and Jake Ryland Williams. 2018. Buzzface: A news veracity dataset with facebook user commentary and egos. In *Twelfth International AAAI Conference on Web and Social Media*.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3410–3416.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. In *European Conference on Information Retrieval*, pages 359–366. Springer.

Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Return of frustratingly easy domain adaptation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2058–2065.

Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819.

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 18–22.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

William Yang Wang. 2017. “liar, liar pants on fire”: A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Dustin Wright and Isabelle Augenstein. 2020. Fact check-worthiness detection as positive unlabelled learning. *arXiv preprint arXiv:2003.02736*.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics.

Annealing Knowledge Distillation

[1,2]Aref Jafari, [2]Mehdi Rezagholizadeh, [1]Pranav Sharma, [1,3]Ali Ghodsi

[1] David R. Cheriton School of Computer Science, University of Waterloo
[2]Huawei Noah's Ark Lab
[3]Department of Statistics and Actuarial Science, University of Waterloo
{aref.jafari, p68sharma, ali.ghodsi}@uwaterloo.ca
mehdi.rezagholizadeh@huawei.com

Abstract

Significant memory and computational requirements of large deep neural networks restrict their application on edge devices. Knowledge distillation (KD) is a prominent model compression technique for deep neural networks in which the knowledge of a trained large teacher model is transferred to a smaller student model. The success of knowledge distillation is mainly attributed to its training objective function, which exploits the soft-target information (also known as "dark knowledge") besides the given regular hard labels in a training set. However, it is shown in the literature that the larger the gap between the teacher and the student networks, the more difficult is their training using knowledge distillation. To address this shortcoming, we propose an improved knowledge distillation method (called Annealing-KD) by feeding the rich information provided by the teacher's soft-targets incrementally and more efficiently. Our Annealing-KD technique is based on a gradual transition over annealed soft-targets generated by the teacher at different temperatures in an iterative process, and therefore, the student is trained to follow the annealed teacher output in a step-by-step manner. This paper includes theoretical and empirical evidence as well as practical experiments to support the effectiveness of our Annealing-KD method. We did a comprehensive set of experiments on different tasks such as image classification (CIFAR-10 and 100) and NLP language inference with BERT-based models on the GLUE benchmark and consistently got superior results.

1 Introduction

Despite the great success of deep neural networks in many challenging tasks such as natural language processing (Vaswani et al., 2017; Liu et al., 2019), computer vision (Wong et al., 2019; Howard et al., 2017), and speech processing (Chan et al., 2016; He et al., 2019), these state-of-the-art networks are usually heavy to be deployed on edge devices with limited computational power (Bie et al., 2019; Lioutas et al., 2019). A case in point is the BERT model (Devlin et al., 2018) which can be comprised of more than a hundred million parameters.

The problem of network over-parameterization and expensive computational complexity of deep networks can be addressed by neural model compression. There are abundant of neural model compression techniques in the literature (Prato et al., 2019; Tjandra et al., 2018; Jacob et al., 2018), among which knowledge distillation (KD) is one of the most prominent techniques (Hinton et al., 2015). KD is tailored a lot to serve different applications and different network architectures (Furlanello et al., 2018; Gou et al., 2020). For instance, patient KD (Sun et al., 2019), TinyBERT (Jiao et al., 2019), and MobileBERT (Sun et al., 2020) are designed particularly for distilling the knowledge of BERT-based teachers to a smaller student.

The success of KD is mainly attributed to its training objective function, which exploits the soft-target information (also known as "dark knowledge") besides the given regular hard labels in the training set (Hinton, 2012). Previous studies in the literature (Lopez-Paz et al., 2015; Mirzadeh et al., 2019) show that when the gap between the student and teacher models increases, training models with KD becomes more difficult. We refer to this problem as KD's *capacity gap problem* in this paper. For example, Mirzadeh et al. (2019) show that if we gradually increase the capacity of the teacher, first the performance of student model improves for a while, but after a certain point, it starts to drop. Therefore, although increasing the capacity of a teacher network usually boosts its performance, it does not necessarily lead to a better teacher for the student network in KD. In other words, it would

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2493–2504
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

be more difficult for KD to transfer the knowledge of this enhanced teacher to the student. A similar scenario happens when originally the gap between the teacher and student network is large.

Mirzadeh et al. (2019) proposed their TAKD solution to this problem which makes the KD process more smooth by filling the gap between the teacher and student networks using an intermediate auxiliary network (referred to as "teacher assistant"). The size of this TA network is between the size of the student and the teacher; and it is trained by the teacher first. Then, the student is trained using KD when the TA network is playing the role of its teacher. This way, the training gap (between the teacher and the student) would be less significant compared to the original KD. However, TAKD suffers from the high computational complexity demand since it requires training the TA network separately. Moreover, the training error of the TA network can be propagated to the student during the KD training process.

In this paper, we want to solve the KD *capacity gap problem* from a different perspective. We propose our Annealing-KD technique to bridges the gap between the student and teacher models by introducing a new KD loss with a dynamic temperature term. This way, Annealing-KD is able to transfer the knowledge of the teacher smoothly to the student model via a gradual transition over soft-labels generated by the teacher at different temperatures. We can summarize the contributions of this paper in the following:

1. We propose our novel Annealing-KD solution to the KD *capacity gap problem* based on modifying the KD loss and also introducing a dynamic temperature function to make the student training gradual and smooth.

2. We provide a theoretical and empirical justification for our Annealing-KD approach.

3. We apply our technique to ResNET8 and plain CNN models on both CIFAR-10 and CIFAR-100 image classification tasks, and the natural language inference task on different BERT based models such as DistilRoBERTa, and BERT-Small on the GLUE benchmark and achieved the-state-of-the art results.

4. Our technique is simple, architecture agnostic, and can be applied on top of different variants of KD.

2 Related Work

2.1 Knowledge Distillation

In the original Knowledge distillation method by Hinton et al. (2015), which is referred to as KD in this paper, the student network is trained based on two guiding signals: first, the training dataset or *hard labels*, and second, the teacher network predictions, which is known as *soft labels*. Therefore, KD is trained based on a linear combination of two loss functions: the regular cross entropy loss function between the student outputs and hard labels, and the KD loss function to minimize the distance between the output predictions of the teacher and student networks at a particular temperature, $\mathcal{T}$, on training samples:

$$\begin{aligned}
&\mathcal{L} = (1-\lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{KD} \\
&\mathcal{L}_{CE} = H_{CE}\Big(y, (\sigma(z_s(x))\Big) \\
&\mathcal{L}_{KD} = \mathcal{T}^2 KL\Big(\sigma(\frac{z_t(x)}{\mathcal{T}}), \sigma(\frac{z_s(x)}{\mathcal{T}})\Big)
\end{aligned} \tag{1}$$

where $H_{CE}(.)$ and $KL(.)$ are representing the cross entropy and KL divergence respectively, $z_s(x)$ and $z_t(x)$ are the output logits from the student and teacher networks, $\mathcal{T}$ is the temperature parameter, $\sigma(.)$ is the softmax function and λ is a coefficient between [0,1] to control the contribution of the two loss functions. The above loss function minimizes the distance between the student model and both the underlying function and the teacher model assuming the teacher is a good approximation of the underlying function of the data.

A particular problem with KD, that we would like to address in this paper, is that the larger the gap between the teacher and the student networks, the more difficult is their training using knowledge distillation (Lopez-Paz et al., 2015; Mirzadeh et al., 2019).

2.2 Teacher Assistant Knowledge Distillation (TAKD)

To address the capacity gap problem between the student and teacher networks in knowledge distillation, TAKD (Mirzadeh et al., 2019) proposes to train the student (of small capacity) with a pre-trained intermediate network (of moderate capacity) called teacher assistance. In this regard, we first train the TA with the guidance of the teacher network by using the KD method. Then, we can use the learned TA network to train the student network. Here, since the capacity of the TA network is between the capacity of the teacher and the student networks, therefore it can fill the gap between the teacher and student and enhance the complexity of the teacher and transfer its knowledge to the student network.

As it is mentioned in (Mirzadeh et al., 2019), a better idea could be using TAKD in a hierarchical way. So in this case, we can have several TAs with different levels of capacity from large capacities close to the teacher model to small capacities close to the student model. Then we could train these TAs consecutively from large capacities to small capacities in order to have a more smooth transfer of teacher's knowledge to the student model. But it will be difficult. Because, first, since we need to train a new model each time, it is computationally expensive. Second, in this way we will have additive error in each step. Each TA after training will have an approximation error and these errors will accumulate and transfer to the next TA. In the next section, we will propose a simple method to realize this idea and avoid the mentioned problems.

2.3 Annealing in Knowledge Distillation

Clark et al. (2019) proposed an annealing idea in their Born-Again Multi-task (BAM) paper , to train a multitask student network using distillation from some single-task teachers. They introduce a so-called teacher annealing scheme to distill from a dynamic weighted mixture of the teacher prediction and the ground-truth label. In this regard, the weight of teacher's prediction is gradually reduced compared to the weight of ground-truth labels during training. Therefore, early in training, the student model mostly learns from the teacher and later on, it learns mostly from target labels. However, our Annealing-KD is different from Clark et al. (2019) in different aspects. First, the introduced annealing term in BAM is conceptually different from our annealing. While in BAM, teacher annealing controls the contribution of the teacher dark knowledge compared to the ground-truth labels during training, our Annealing-KD is only applied to the teacher output in the KD loss to solve the capacity gap problem between the teacher and student networks. Second, the way we do annealing in our technique is through the temperature parameter and not by controlling the contribution of the teacher and ground-truth labels. Third, BAM falls into another category of knowledge distillation which focuses on improving the performance of the student model and not compressing it. Our method is described in the next section.

3 Method: Annealing Knowledge Distillation

In this section, we describe our Annealing-KD technique and show the rationale behind it. First, we start by formulating the problem and visualizing our technique using an example for a better presentation. Then, we use VC-dimension theory to understand why our technique improves knowledge distillation. We wrap up this section by visualizing the loss landscape of Annealing KD for a ResNet network in order to investigate the impact of our method on the KD loss function.

KD defines a two-objective loss function (i.e. the $\mathcal{L}_{KD}$ and $\mathcal{L}_{CE}$ terms in Equation 1) to minimize the distance between student predictions and soft labels and hard labels simultaneously. Without adding to the computational needs of the KD algorithm, our Annealing-KD model breaks the KD training into two stages: Stage I, gradually training the student to mimic the teacher using our Annealing-KD loss $\mathcal{L}_{\text{KD}}^{\text{Annealing}}$; Stage II, fine-tuning the student with hard labels using $\mathcal{L}_{CE}$. We can define the loss function of our method as following.

$$\mathcal{L} = \begin{cases} \mathcal{L}_{\text{KD}}^{\text{Annealing}}(i), & \textbf{Stage I: } 1 \leq \mathcal{T}_i \leq \tau_{\text{max}} \\ \mathcal{L}_{CE}, & \textbf{Stage II: } \mathcal{T}_n = 1 \end{cases} \tag{2}$$

In the above equation, i indicates the epoch index in the training process with the max epoch number of n for stage I, $\mathcal{T}_i$ represents the temperature value at i^{th} epoch, $\mathcal{L}_{CE}$ is unchanged from Equation 1, and at each epoch (i), $\mathcal{L}_{\text{KD}}^{\text{Annealing}}(i)$ is defined as following:

$$\begin{aligned} &\mathcal{L}_{\text{KD}}^{\text{Annealing}}(i) = ||z_s(x) - z_t(x) \times \Phi(\mathcal{T}_i)||_2^2 \\ &\Phi(\mathcal{T}) = 1 - \frac{\mathcal{T}-1}{\tau_{\text{max}}}, 1 \leq \mathcal{T} \leq \tau_{\text{max}}, \mathcal{T} \in \mathbb{N} \end{aligned} \tag{3}$$

In Equation 2, $\mathcal{L}_{\text{KD}}^{\text{Annealing}}$ is defined as an MSE loss between the logits of the student ($z_s(x)$) and an annealed version of the teacher logits ($z_t(x)$), obtained by multiplying the logits by the *annealing function* $\Phi(\mathcal{T})$. The annealing function $\Phi(\mathcal{T})$ can be replaced with any monotonically decreasing function $\Phi : [1, \tau_{\text{max}}] \in \mathbb{N} \rightarrow [0, 1] \in \mathbb{R}$. In stage I of our training, initially we set $\mathcal{T}_1 = \tau_{\max}$ (which leads to the most softened version of the teacher outputs because $\Phi(\mathcal{T}_1) = \frac{1}{\tau_{\max}}$) and decrease the temperature during training as the epoch number grows (that is $\mathcal{T} \rightarrow 1$ while $i \rightarrow n$). Training in stage I continues until $i = n, \mathcal{T} = 1$, for which $\Phi(\mathcal{T}_n) = 1$ and we get the sharpest version of z_t without any softening. The intuition behind using the MSE loss in stage I is that matching the logits of the teacher and student models is a regression task and MSE is one of the best loss functions for this task. We also did an ablation study to compare the performance of MSE and KL-divergence loss function in stage I, and the results of this study support our intuition. For more details, please refer to table 10 of the appendices.

Therefore, our Annealing-KD bridges the gap between the student and teacher models by introducing the dynamic temperature term (that is the annealing function $\Phi(\mathcal{T})$) in the stage I of training. This way our Annealing-KD method is able to smoothly transfer the teacher's knowledge to the student model via a gradual transition over soft-labels generated by the teacher at different temperatures.

To summarize, our Annealing-KD technique is different from KD in following aspects:

- Annealing-KD does not need any λ hyper-parameter to weigh the contribution of the soft and hard lable losses, because it does the training of each loss in a different stage.
- Our Annealing-KD loss $\mathcal{L}_{\text{KD}}^{\text{Annealing}}$ uses $||.||_2^2$ loss instead of the KL divergence.
- Moreover, our technique uses a dynamic temperature by defining the annealing function $\Phi(\mathcal{T})$ in the Annealing-KD loss instead of using a fixed temperature in KD.
- Our empirical experiments showed that it is best to take the network logits instead of the softmax outputs in $\mathcal{L}_{\text{KD}}^{\text{Annealing}}$. Furthermore, in contrast to KD, we do not add the temperature term to student output.

Algorithm 1 explains the proposed method in more detail.

In this section, we proposed an approach to alleviate the gap between the teacher and student models as well as reducing the sharpness of the KD loss function. In our model, instead of pushing the student network to learn a complex teacher function from scratch, we start training the student from a softened version of the teacher and we gradually move toward the original teacher outputs through our annealing process.

Algorithm 1

```
function ANNEALING-KD($S$,$T$,$X$, $k$, $\mathcal{T}_{max}$, $n$)

                                                    ▷ stage I
    for $\mathcal{T} = \tau_{max}$ to 1 do
        $\Phi \leftarrow 1 - \frac{\mathcal{T}-1}{\tau_{\max}}$
        for $i = 1$ to $k$ do
            TRAIN-ANNEALING($S$,$T$, $X$,$\Phi$)
            SAVE-BEST-CHECKPOINT($S$)
        end for
    end for
    $S \leftarrow$ LOAD-BEST-CHECKPOINT  ▷ stage II
    for $i = 1$ to $n$ do
        TRAIN-FINE-TUNE($S$, $X$)
        SAVE-BEST-CHECKPOINT($S$)
    end for
    $S \leftarrow$ LOAD-BEST-CHECKPOINT
    return $S$
end function
```

3.1 Example

For better illustration of our proposed method, we designed a simple example to visualize different parts of our Annealing-KD algorithm. In this regard, we defined a simple regression task using a simple 2D function. This function is a liner combination of three sinusoidal functions with different frequencies $f(x) = sin(3\pi x) + sin(6\pi x) + sin(9\pi x)$. We randomly sample some points from this function to form our dataset (Figure 2-(a)). Next, we fit a simple fully connected neural network with only one hidden layer and the sigmoid activation function to the underlying function of the defined dataset. The teacher model is composed of 100 hidden neurons and trained with the given dataset. After training, the teacher is able to get very close to training data (see the green curve in Figure 2-(a)). We plot the annealed output of the teacher function in 10 different temperatures in Figure. 2-(b). Then, a student model with 10 hidden neurons is trained once with regular KD (Figure 2-(f)) and once with our Annealing-KD (Figures. 2-(c, d, e) depicts the student output at temperatures 10,

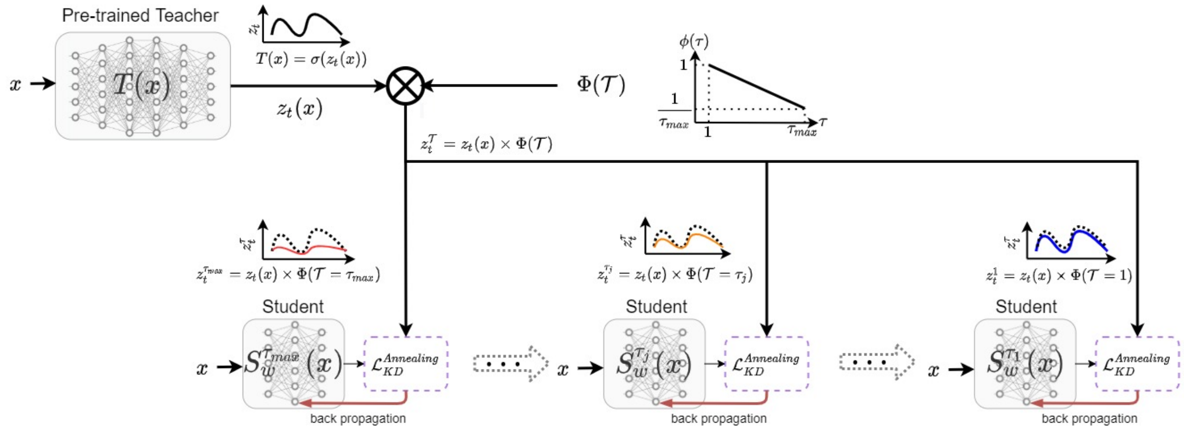


Figure 1: Illustrating the Stage I of the Annealing-KD technique. Given a pre-trained teacher network, we can derive the annealed output of the teacher at different temperature using the annealing function $\Phi(\mathcal{T})$. We start training of the student from $\mathcal{T} = \tau_{max}$ and go to $\mathcal{T} = 1$.

5, and 1 during the Annealing-KD training). As it is shown in these figures, Annealing-KD guides the student network gradually until it gets to a good approximation of the underlying function and it can match the teacher output better than regular KD.

3.2 Rationale Behind Annealing-KD

Inspired by (Mirzadeh et al., 2020), we can leverage the VC-dimension theory and visulaizing loss landscape to justify why Annealing-KD works better than original KD.

3.2.1 Theoretical Justification

In VC-dimension theory (Vapnik, 1998), the error of classification can be decomposed as:

$$R(f_s) - R(f) \leq O(\frac{|\mathcal{F}_s|_c}{N^{\alpha_s}}) + \varepsilon_s \qquad (4)$$

where $R(.)$ is the expected error, $f_s \in \mathcal{F}_s$ is the learner belongs to the function class $\mathcal{F}_s$. f is the underlying function. $|.|_c$ is some function class capacity measure. $O(.)$ is the estimation error of training the learner and ε_s is the approximation error of the best estimator function belonging to the $\mathcal{F}_s$ class (Mirzadeh et al., 2019). Moreover, N is the number of training samples, and $\frac{1}{2} \leq \alpha \leq 1$ is a parameter related to the difficulty of the problem. α is close to $\frac{1}{2}$ for more difficult problems (slow learners) and α is close to 1 for easier problems or fast learners (Lopez-Paz et al., 2015).

In knowledge distillation, we have three main factors: the student (our learner), the teacher, and the underlying function. Based on (Lopez-Paz et al., 2015; Mirzadeh et al., 2019), we can rewrite Equation 4 for knowledge distillation as following:

$$R(f_s) - R(f_t) \leq O(\frac{|\mathcal{F}_s|_c}{n^{\alpha_{st}}}) + \varepsilon_{st} \qquad (5)$$

where the student function f_s is following f_t. To define similar inequality for our Annealing-KD technique, we need to consider the effect of the temperature parameter on the three main functions in KD first. For this purpose, we can define $f_s^{\mathcal{T}}$, $f_t^{\mathcal{T}}$, and $f^{\mathcal{T}}$ as the annealed versions of student, teacher, and underlying functions. Furthermore, let $R_{\mathcal{T}}(.)$ to be the expected error function w.r.t the annealed underlying function at temperature $\mathcal{T}$. Hence, for Annealing-KD we have

$$R_{\mathcal{T}}(f_s^{\mathcal{T}}) - R_{\mathcal{T}}(f_t^{\mathcal{T}}) \leq O(\frac{|\mathcal{F}_s|_c}{n^{\alpha_{st}^{\mathcal{T}}}}) + \varepsilon_{st}^{\mathcal{T}}. \qquad (6)$$

Note that in $\mathcal{T} = 1$, $f_t^1 = f_t$, $f_s^1 = f_s$, $f^1 = f$, and $R_1(.) = R(.)$. Therefore, we can rewrite Equation 6 at $\mathcal{T} = 1$ as:

$$R_1(f_s^1) - R_1(f_t^1) \leq O(\frac{|\mathcal{F}_s|_c}{n^{\alpha_{st}^1}}) + \varepsilon_{st}^1. \qquad (7)$$

That being said, to justify that our Annealing-KD is working better than original KD, we can compare Equations 7 and 5 to show the following inequality holds.

$$O(\frac{|\mathcal{F}_s|_c}{n^{\alpha_{st}^1}}) + \varepsilon_{st}^1 \leq O(\frac{|\mathcal{F}_s|_c}{n^{\alpha_{st}}}) + \varepsilon_{st} \qquad (8)$$

Since in Annealing-KD, the student network at each temperature is initialized with the trained student network at $f_s^{\mathcal{T}-1}$, the student is much closer to the teacher compared with the original KD method, where the student starts from random a initialization. In other words, in annealing KD, the student

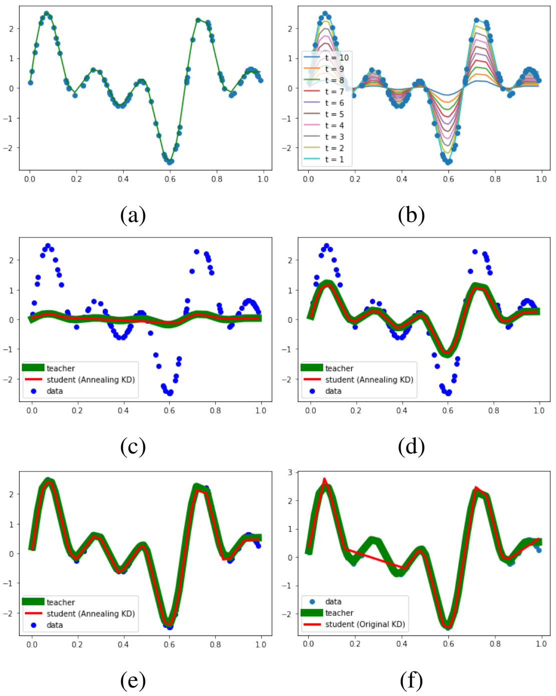


Figure 2: (a) Data samples and trained teacher. (b) Annealed teacher in different temperatures. (c) Student after matching to annealed teacher in $\mathcal{T} = 10$. (d) Student after matching to annealed teacher in $\mathcal{T} = 5$. (e) Student after matching to $\mathcal{T} = 1$. (f) Student trained without KD.

network can learn the annealed teacher at temperature $\mathcal{T}$ faster than the case it starts from a random initial point. Therefore, we can conclude that $\alpha_{st} \leq \alpha_{st}^{\mathcal{T}}$. This property also holds for the last step of annealing KD where $\mathcal{T} = 1$. It means we have $\alpha_{st} \leq \alpha_{st}^{1}$. Furthermore, bear in mind that since the approximation error depends on the capacity of the learner and in annealing KD we do not change the structure of the student, then we expect to have $\varepsilon_{st} = \varepsilon_{st}^{\mathcal{T}}$. Therefore, based on these two evidence ($\alpha_{st} \leq \alpha_{st}^{\mathcal{T}}$ and $\varepsilon_{st} = \varepsilon_{st}^{\mathcal{T}}$), we can conclude that Equation 8 holds.

3.2.2 Empirical Justification

Because of the non-linear nature of neural networks, the loss functions of these models are non-convex. This property might prevent a learner from a good generalization. There are some beliefs in the community of machine learning, this phenomena can be harsher in the sharp loss functions than the flat loss functions (Chaudhari et al., 2019; Hochreiter and Schmidhuber, 1997). Although, there are some arguments around this belief (Li et al., 2018), for the case of knowledge distillation it seems flatter loss functions are related to higher accuracy (Mirzadeh et al., 2019; Zhang et al., 2018; Hinton et al., 2015). One of the advantages of annealing the teacher function during training is reducing the sharpness of annealing loss function in the early steps of stage I. In other words, the sharpness of the loss function in annealing KD changes dynamically. In the early steps of annealing when the temperature is high, the loss function is flatter. This helps the student to train the teacher network's behaviour faster and easier.

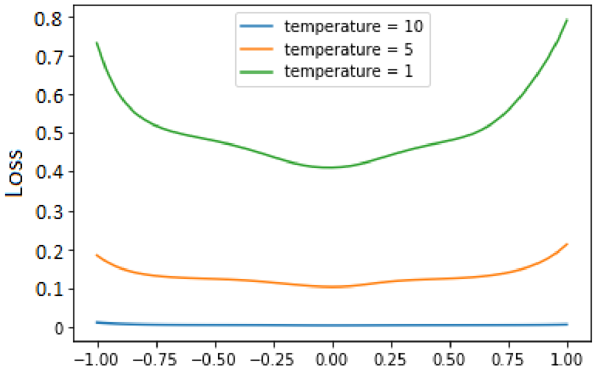


Figure 3: Visualization of annealing KD loss function in stage I for ResNet 8 student during the training on CIFAR-10 dataset in different temperatures

In order to compare the effect of different temperatures, the loss landscape visualization method in (Li et al., 2018) is used to plot the loss behaviour of CIFAR-10 experiment with ResNet 8 student in Figure. 3. Here as it is shown, by decreasing the temperature during the training, the sharpness of the loss function increases. So the student network can avoid many of the bad local minimums in the early stages of the algorithm when the temperature is high. Then in the final stages of the algorithm, when the loss function is sharper, the network starts from a much better initialization.

4 Experiments

In this section, we describe the experimental evaluation of our proposed Annealing KD method. We evaluate our technique on both image classification and natural language inference tasks. In all of our experiments, we compare the annealing KD results with TAKD, standard KD, and training student without KD results.

4.1 Datasets

For image classification, we assess Annealing-KD on CIFAR-10 and CIFAR-100 datasets (Krizhevsky et al., 2009) which are image datasets containing

32 × 32 color images with 10 and 100 classes respectively. For the natural language inference task, we employ the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), which is a collection of nine different tasks for training, evaluating, and analyzing natural language understanding models. GLUE consists of Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2017), Quora Question Pairs (QQP) (Chen et al., 2018), Question Natural Language Inference (QNLI) (Rajpurkar et al., 2016), Stanford Sentiment Treebank (SST-2) (Socher et al., 2013), Corpus of Linguistic Acceptability (COLA) (Warstadt et al., 2019), Semantic Textual Similarity Benchmark (STS-B) (Cer et al., 2017), Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005), Recognizing Textual Entailment (RTE) (Bentivogli et al., 2009), Winograd NLI (WNLI) (Levesque et al., 2012).

4.2 Experimental Setup for Image Classification Tasks

For image classification experiments, we used CIFAR-10 and CIFAR-100 datasets with the same experimental setup in the TAKD method (Mirzadeh et al., 2020). In these experiments, we used ResNet and plain CNN networks as the teacher, student, and also the teacher assistant for the TAKD baseline. For the ResNet experiments, we used ResNet-110 as the teacher and ResNet-8 as the student. For plain CNN experiments, we used CNN network with 10 layers as teacher and 2 layers as the student according to TAKD. Also, for the TAKD baseline, we used ResNet-20 and CNN with 4 layers as the teacher assistant. Tables 1 and 2 compare the annealing KD performance with other baselines over CIFAR-10 and CIFAR-100 datasets respectively. For the ResNet experiments in both tables 1 and 2, the teacher ResNet-110 is trained from scratch and a ResNet-20 TA is trained by the teacher using KD. Then we would like to train a ResNet-8 student using different techniques and compare their performance against our Annealing KD method. In this regard, we evaluate the performance of training the student from scratch, training with the large ResNet-110 teacher using KD, training with TA as the teacher and using our Annealing-KD approach. The results of this experiment with ResNet show that our Annealing-KD outperforms all other baselines and TAKD is the second-best performing student without significant distinction compared to KD. More details about the training hyper-parameters are added to the appendix A.

Table 1: Comparing the test accuracy of annealing KD, TAKD, regular KD, and student without teacher on CIFAR-10 dataset with both ResNet and CNN models

Model	Type	Training method	Accuracy
ResNet	Teacher(110)	from scratch	93.8
	TA(20)	KD	92.39
	Student(8)	from scratch	88.44
	Student(8)	KD	88.45
	Student(8)	TAKD	88.47
	Student(8)	**Annealing KD (ours)**	**89.44**
CNN	Teacher(10)	from scratch	90.1
	TA(4)	KD	82.39
	Student(2)	from scratch	72.75
	Student(2)	KD	72.43
	Student(2)	TAKD	72.62
	Student(2)	**Annealing KD (ours)**	**73.17**

Table 2: Comparing the test accuracy of annealing KD, TAKD, regular KD, and student without teacher on CIFAR-100 dataset with both ResNet and CNN models

Model	Type	Training method	Accuracy
ResNet	teacher(110)	from scratch	71.92
	TA(20)	KD	67.6
	student(8)	from scratch	61.37
	student(8)	KD	61.41
	student(8)	TAKD	61.82
	student(8)	**Annealing KD (ours)**	**63.1**
CNN	Teacher(10)	from scratch	64.89
	TA(4)	KD	60.73
	student(2)	from scratch	51.35
	student(2)	KD	51.62
	student(2)	TAKD	51.85
	student(2)	**Annealing KD (ours)**	**53.35**

4.3 Experimental setup for GLUE tasks

For these set of experiments, we use the GLUE benchmark which consists of 9 natural language understanding tasks. In the first experiment (Table 3), we use RoBERTa-large (24 layers) as teacher, DistilRoBERTa (6 layers) as student, and RoBERTa-base (12 layers) as the teacher assistant for the TAKD baseline. For Annealing KD, we use a maximum temperature of 7, learning rate of 2e-5, and train for 14 epochs in phase 1, and 6 epochs in phase 2. In table 3 the Annealing KD and the other baselines performances on dev set of GLUE tasks are compared. Also, we compared the performances of these methods on test set based on the GLUE benchmark's leaderboard results in table 4. In the second experiment (Table 5), we use

Table 3: DistilRoBERTa results for Annealing KD on dev set. F1 scores are reported for MRPC, pearson correlations for STB-B, and accuracy scores for all other tasks.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI	Score
Teacher	68.1	86.3	91.9	92.3	96.4	94.6	91.5	90.22/89.87	56.33	85.29
From scratch	59.3	67.9	88.6	88.5	92.5	90.8	90.9	84/84	52.1	79.3
Vanilla KD	60.97	71.11	90.2	88.86	92.54	91.37	91.64	84.18/84.11	56.33	80.8
TAKD	61.15	71.84	89.91	88.94	92.54	91.32	**91.7**	83.89/84.18	56.33	80.85
Annealing KD	**61.67**	**73.64**	**90.6**	**89.01**	**93.11**	**91.64**	91.5	**85.34/84.6**	56.33	**81.42**

Table 4: Performance of DistilRoBERTa trained by annealing KD on the GLUE leaderboard compared with Vanilla KD and TAKD. We applied the standard tricks to all 3 methods and fine-tune RTE, MRPC and STS-B from trained MNLI student model.

KD Method	CoLA	MRPC	STS-B	SST-2	MNLI-m	MNLI-mm	QNLI	QQP	RTE	WNLI	Score
Vanilla KD	**54.3**	86/80.8	85.7/84.9	93.1	83.6	82.9	90.8	71.9/89.5	74.1	65.1	78.9
TAKD	53.2	86.7/82.7	85.6/84.4	93.2	83.8	83.2	**91**	72/89.4	**74.2**	65.1	79
Annealing KD	54	**88.0/83.9**	**87.0/86.6**	**93.6**	**83.8**	**83.9**	90.8	**72.6/89.7**	73.7	65.1	**79.5**

Table 5: BERT-Small results for Annealing KD on dev set. F1 scores are reported for MRPC, pearson correlations for STS-B, and accuracy scores for all other tasks.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI	Score
Teacher	65.8	71.48	89.38	89.2	92.77	92.82	91.45	86.3/86.4	60.56	82.19
Vanilla KD	33.5	57	86	72.3	88.76	**83.15**	**87**	72.62/73.19	54.92	70.58
TAKD	34.24	59.56	85.23	71.1	89.1	82.62	87	72.32/72.45	54.92	70.76
Annealing KD	**35.98**	**61**	**86.2**	**74.54**	**89.44**	83.14	86.5	**73.85/74.84**	54.92	**71.68**

BERT-large (24 layers) as teacher, BERT-small (4 layers) as student, and BERT-base (12 layers) as the teacher assistant of TAKD. We use a maximum temperature of 7 for MRPC, SST-2, QNLI, and WNLI, and 14 for all other tasks. The number of epochs in phase 1 is twice the maximum temperature, and 6 in phase 2. We use the learning rate of 2e-5 for all tasks except RTE and MRPC which use 4e-5. Table 5 compares the performance of annealing KD and other baselines on dev set for small-BERT experiments. For more details regarding other hyper-parameters, refer to the appendix. We also perform ablation on the choice of loss function in phase 1, and choice of different max temperature values, both of which can be found in the appendix.

4.4 GLUE Results

We present our results in Tables 3, 4, and 5. We see that Annealing KD consistently outperforms the other techniques both on dev set as well as the GLUE leaderboard. Furthermore, in table 5, when we reduce the size of the student to a 4 layer model (BERT-Small), we notice almost twice as big of a gap in the average score over Vanilla KD when compared with DistilRoBERTa (Table 3). We can also observe TAKD improving slightly over Vanilla KD, with the improvement being more significant in the case of the smaller student (BERT-Small).

5 Discussion

In image classification experiments, the improvement gap between the annealing KD results and the other baselines in CIFAR-100 experiments is larger than CIFAR-10 ones. We can observe similar conditions for the NLP experiments between BERT-small and DistilRoBERTa students (the performance gap of BERT-small is larger). In both of these cases, the problem for the student was more difficult. CIFAR-100 dataset is more complex than CIFAR-10 dataset. So the teacher has learned a more complex function that should be transferred to the student. In NLP experiments, on the other hand, the tasks are the same but BERT-small student has a smaller capacity in compare with DistilRoBERTa. Therefore the problem is more difficult for BERT-small. From this observation, we can conclude, whenever the gap between the teacher and student is larger, the annealing KD performs better than the other baselines and leverage the acquired knowledge by the teacher to train the student.

6 Conclusion and Future Work

In this work, we discussed that the difference between the capacity of the teacher and student models in knowledge distillation may hamper its performance. On the other hand, in most cases, larger neural networks can be trained better and get more accurate results. If we consider better teachers can train better students, then larger teachers with better accuracy would be more favourable for knowledge distillation training. In this paper, we proposed an improved knowledge distillation method called annealing KD to alleviate this problem and leverage the knowledge acquired by more complex teachers to guide the small student models better during their training. This happened by feeding the rich information provided by the teacher's soft-targets incrementally and more efficiently. Our Annealing-KD technique was based on a gradual transition over annealed soft-targets generated by the teacher at different temperatures in an iterative process; and therefore, the student was trained to follow the annealed teacher output in a step-by-step manner.

References

Bentivogli, L., Clark, P., Dagan, I., and Giampiccolo, D. (2009). The fifth pascal recognizing textual entailment challenge. In *TAC*.

Bie, A., Venkitesh, B., Monteiro, J., Haidar, M., Rezagholizadeh, M., et al. (2019). Fully quantizing a simplified transformer for end-to-end speech recognition. *arXiv preprint arXiv:1911.03604*.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Chan, W., Jaitly, N., Le, Q., and Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964. IEEE.

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. (2019). Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018.

Chen, Z., Zhang, H., Zhang, X., and Zhao, L. (2018). Quora question pairs.

Clark, K., Luong, M.-T., Khandelwal, U., Manning, C. D., and Le, Q. V. (2019). BAM! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5931–5937, Florence, Italy. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dolan, W. B. and Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., and Anandkumar, A. (2018). Born again neural networks. *arXiv preprint arXiv:1805.04770*.

Gou, J., Yu, B., Maybank, S. J., and Tao, D. (2020). Knowledge distillation: A survey. *arXiv preprint arXiv:2006.05525*.

He, Y., Sainath, T. N., Prabhavalkar, R., McGraw, I., Alvarez, R., Zhao, D., Rybach, D., Kannan, A., Wu, Y., Pang, R., et al. (2019). Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385. IEEE.

Hinton, G. (2012). Neural networks for machine learning, coursera. *URL: http://coursera.org/course/neuralnets*.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv e-prints*, page arXiv:1503.02531.

Hochreiter, S. and Schmidhuber, J. (1997). Flat minima. *Neural Computation*, 9(1):1–42.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. (2019). Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Levesque, H., Davis, E., and Morgenstern, L. (2012). The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. (2018). Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399.

Lioutas, V., Rashid, A., Kumar, K., Haidar, M. A., and Rezagholizadeh, M. (2019). Distilled embedding: non-linear embedding factorization using knowledge distillation. *arXiv preprint arXiv:1910.06720*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lopez-Paz, D., Bottou, L., Schölkopf, B., and Vapnik, V. (2015). Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*.

Mirzadeh, S., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., and Ghasemzadeh, H. (2020). Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *AAAI 2020*, abs/1902.03393.

Mirzadeh, S.-I., Farajtabar, M., Li, A., and Ghasemzadeh, H. (2019). Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *arXiv preprint arXiv:1902.03393*.

Prato, G., Charlaix, E., and Rezagholizadeh, M. (2019). Fully quantized transformer for improved translation. *arXiv preprint arXiv:1910.10485*.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Sun, S., Cheng, Y., Gan, Z., and Liu, J. (2019). Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., and Zhou, D. (2020). Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Tjandra, A., Sakti, S., and Nakamura, S. (2018). Tensor decomposition for compressing recurrent neural network. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Vapnik, V. (1998). *Statistical learning theory*. John Wiley and Sons.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2018). Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Warstadt, A., Singh, A., and Bowman, S. R. (2019). Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Williams, A., Nangia, N., and Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Wong, A., Famuori, M., Shafiee, M. J., Li, F., Chwyl, B., and Chung, J. (2019). Yolo nano: a highly compact you only look once convolutional neural network for object detection. *arXiv preprint arXiv:1910.01271*.

Zhang, Y., Xiang, T., Hospedales, T. M., and Lu, H. (2018). Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328.

Appendices

A Experimental parameters of the image classification tasks

In this section, we include more detail of our experimental settings of section 4.2 in the paper. For the baseline experiments, we used the same experimental setup as (Mirzadeh et al., 2019). We performed two series of experiments based on ResNet and plain CNN neural networks on CIFAR-10 and CIFAR-100 datasets. Table 6 illustrates the hyper-parameters used in these experiments. (BS = batch size, EP1= number of epochs in phase 1 (for the baselines, this is the number of training epochs), EP2 = number of epochs in phase 2, LR = learning rate, MO = momentum, WD = weight decay, τ_{max} = maximum temperature)

Table 6: Hyper-parameters of CIFAR-10 and CIFAR-100 experiments

Model	Type	Training method	BS	EP1	EP2	LR	MO	WD	τ_{max}
ResNet	Teacher(110)	from scratch	128	160	N/A	0.1	0.9	10^{-4}	N/A
	TA(20)	KD	128	160	N/A	0.1	0.9	10^{-4}	N/A
	Student(8)	from scratch	128	160	N/A	0.1	0.9	10^{-4}	N/A
	Student(8)	KD	128	160	N/A	0.1	0.9	10^{-4}	1
	Student(8)	TAKD	128	160	N/A	0.1	0.9	10^{-4}	1
	Student(8)	**Annealing KD (ours)**	128	160	160	0.1	0.9	10^{-4}	10
CNN	Teacher(10)	from scratch	128	160	N/A	0.1	0.9	10^{-4}	N/A
	TA(4)	KD	128	160	N/A	0.1	0.9	10^{-4}	N/A
	Student(2)	from scratch	128	160	N/A	0.1	0.9	10^{-4}	N/A
	Student(2)	KD	128	160	N/A	0.1	0.9	10^{-4}	1
	Student(2)	TAKD	128	160	N/A	0.1	0.9	10^{-4}	1
	Student(2)	**Annealing KD (ours)**	128	160	160	0.1	0.9	10^{-4}	10

B BERT Experiments

In these experiments, RoBERTa-large (24 layers) and DistilRoBERTa (6 layers) are used as the teacher and student models respectively. Also, RoBERTa-base (12-layer) is used as the teacher assistant for the TAKD baseline. For Annealing KD, we use the maximum temperature of 7 and the learning rate of 2e-5 for all the tasks. We trained the student model for 14 epochs in phase 1, and 6 epochs in phase 2. Table 8 illustrates the details of the hyper-parameters of the experiments. Also, Table 11 illustrates the hyper-parameter values of BERT-small experiments in detail. Also, we did two ablation studies. In the first one, we tried to fine-tune the maximum temperature in annealing KD and check the performance improvement compared with using the general value of 7. As it is illustrated in Table 9, we can get more improvement with selecting the maximum temperature parameter more carefully. The second ablation is about comparing the effect of mean square error and KL-divergence loss functions on the final results of the experiments when they are used as the loss function of the first phase. Table 10 shows the results of this ablation.

Table 7: Common Hyper-parameters for DistilRoBERTa and BERT-Small models on GLUE tasks

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Batch Size	32	32	32	32	32	32	32	32	32
Max Seq. Length	128	128	128	128	128	128	128	128	128
Vanilla KD Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Gradient Clipping	1	1	1	1	1	1	1	1	1

Table 8: Model specific Hyper-parameters for DistilRoBERTa on GLUE tasks

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Learning Rate	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
Phase 1 epochs	14	14	14	14	14	14	14	14	14
Phase 2 epochs	4	4	4	4	4	4	4	4	4
τ_{max}	7	7	7	7	7	7	7	7	7

Table 9: Ablation on DistilRoberta Annealing KD with temperature tuning

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI	Avg
Annealing KD	61.67	73.64	90.6	89.01	93.11	91.64	91.5	85.34/84.6	56.33	81.42
+ temp tuning	61.67	73.64	**91.99**	**89.26**	**93.34**	**92**	**91.72**	**85.14/85.22**	56.33	**81.67**
(max temperature)	7	7	8	14	14	11	14	14	7	-

Table 10: Ablation on DistilRoberta Annealing KD with different loss functions

KD Method and Loss	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI	Avg
Annealing KD, MSE	61.67	73.64	90.6	89.01	93.11	91.64	91.5	85.34/84.6	56.33	81.42
Annealing KD, KL-div	62.56	70.75	90.84	89.01	93	91.32	91.42	85/84.75	56.33	81.13

Table 11: Model specific Hyper-parameters for BERT-Small on GLUE tasks

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Learning Rate	2e-5	4e-5	4e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
Phase 1 epochs	28	28	14	28	14	14	28	28	14
Phase 2 epochs	6	6	6	6	6	6	6	6	6
τ_{max}	14	14	7	14	7	7	14	14	7

Unsupervised Extractive Summarization using Pointwise Mutual Information

Vishakh Padmakumar
New York University
`vishakh@nyu.edu`

He He
New York University
`hehe@cs.nyu.edu`

Abstract

Unsupervised approaches to extractive summarization usually rely on a notion of sentence importance defined by the semantic similarity between a sentence and the document. We propose new metrics of relevance and redundancy using pointwise mutual information (PMI) between sentences, which can be easily computed by a pre-trained language model. Intuitively, a relevant sentence allows readers to infer the document content (high PMI with the document), and a redundant sentence can be inferred from the summary (high PMI with the summary). We then develop a greedy sentence selection algorithm to maximize relevance and minimize redundancy of extracted sentences. We show that our method outperforms similarity-based methods on datasets in a range of domains including news, medical journal articles, and personal anecdotes.

1 Introduction

Modern neural network-based approaches to summarization require a large amount of document-summary pairs that are usually unavailable outside of the news domain. For example, summarization datasets of personal narratives and office meetings contain only a few hundred examples (Ouyang et al., 2017; Carletta et al., 2005). In this work, we tackle the problem of unsupervised extractive summarization which aims to select important sentences from the document. While there exists extensive prior work (Radev et al., 2000; Mihalcea and Tarau, 2004; Liu and Lapata, 2019; Zheng and Lapata, 2019), most approaches rely on the assumption that important sentences are *similar* to other sentences in the document. However, it is unclear if similarity-based features lead to meaningful content selection (Kedzie et al., 2018).

Inspired by recent work on formalizing the notion of importance in summarization (Peyrard, 2019), we propose metrics for *relevance* and *redundancy* based on pointwise mutual information (PMI). Intuitively, a relevant summary allows the reader to maximally infer the document content, and a summary has minimal redundancy if each sentence in it provides additional information. Therefore, we measure the relevance of a summary by its PMI with the document. High relevance means that the probability of the document increases conditioning on the summary. Similarly, we measure redundancy by PMI of sentence pairs *within the summary*. A sentence is redundant if seeing other sentences significantly increases its probability.

Based on the new metrics, we design a simple sentence extraction algorithm. We estimate the PMI of sentence pairs by a pre-trained language model fine-tuned on in-domain documents. We then use a simple sequential sentence selection algorithm for extractive summarization, which greedily maximizes relevance and minimizes redundancy.

Experimental results show that our algorithm outperforms similarity-based methods across multiple domains, including news, personal stories, and medical articles.[1]

2 Relevance and Redundancy

We begin by formalizing relevance and redundancy for summarization. Consider a document of n sentences $D = \{d_1, \ldots, d_n\}$ and a summary of m sentences $S = \{s_1, \ldots, s_m\}$. We would like to measure the relevance of S to D and the redundancy of S.

Relevance. Relevance measures how well the summary condenses the original text such that we can infer its key content. Specifically, a summary

---

[1] Our code and pretrained models are available at `https://github.com/vishakhpk/mi-unsup-summ`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2505–2512
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

sentence is relevant if observing it reduces our uncertainty about (unseen) sentences in the document. For example, the summary may contain the main link in the thread of conversation in the document. We thus quantify the relevance of a summary sentence s to a document sentence d by their PMI:

$$\text{Rel}(s, d) \overset{\text{def}}{=} \text{pmi}(s; d), \tag{1}$$

which measures the dependence between s and d. A positive score means that s and d are very likely to co-occur, thus seeing one implies the other. A zero score means that s and d are independent. A negative score means that s and d are unlikely to co-occur, e.g. contradicting sentences, thus such a summary sentence is discouraged. We further define the relevance of a summary S to the document D by the sum of sentence-level relevance:

$$\text{Rel}(S, D) \overset{\text{def}}{=} \sum_{s \in S} \sum_{d \in D} \text{Rel}(s, d). \tag{2}$$

Redundancy. Redundancy measures how much overlap exists among the summary sentences. It is typically measured by the semantic similarity between two sentences. However, even if two sentences express different meanings, there is redundancy if one is entailed by the other. For example, consider:

1. *"Michelle, of South Shields, Tyneside, says she feels like a new woman after dropping from dress size 30 to size 12."*
2. *"Michelle weighed 25st 3lbs when she joined the group in April 2013 and has since dropped to 12st 10lbs."*

Though expressing different information, both imply Michelle's weight loss.

Given a summary sentence, we want to assign a score proportional to the amount of information in the sentence which is already present in the rest of the summary. Therefore, we quantify the redundancy of a sentence s given another sentence s' by their dependence in terms of PMI:

$$\text{Red}(s, s') \overset{\text{def}}{=} \text{pmi}(s; s'). \tag{3}$$

Similarly, the redundancy of a summary S is defined by the total redundancy of all sentence pairs:

$$\text{Red}(S) \overset{\text{def}}{=} \sum_{i=1}^{m} \sum_{j=i+1}^{m} \text{pmi}(s_i; s_j). \tag{4}$$

Estimate PMI. By definition, $\text{pmi}(s; d) = \log \frac{p(s|d)}{p(d)} = \log \frac{p(d|s)}{p(s)}$. Since both s and d are sentences, we use a language model, p_{LM}, to estimate the probabilities. Conditional probabilities are comoputed by considering the condition sentence as the prefix.

Note that while PMI can be computed in two equivalent ways according to the definition, the estimates from a language model do not guarantee that $\frac{p_{\text{LM}}(d|s)}{p_{\text{LM}}(d)} = \frac{p_{\text{LM}}(s|d)}{p_{\text{LM}}(s)}$. Thus we choose to condition on the summary sentence:

$$\text{pmi}(s; d) \overset{\text{def}}{=} \log \frac{p_{\text{LM}}(d \mid s)}{p_{\text{LM}}(d)}. \tag{5}$$

This is consistent with our definition of relevance: *seeing the summary*, how well we can estimate the document content. For redundancy, we condition on the earlier sentence:

$$\text{pmi}(s_i; s_j) \overset{\text{def}}{=} \begin{cases} \log \frac{p_{\text{LM}}(s_j|s_i)}{p_{\text{LM}}(s_j)} & \text{if } i < j \\ \text{pmi}(s_j; s_i) & \text{otherwise} \end{cases}, \tag{6}$$

since a sentence is redundant if it can be inferred from previous sentences.

3 Sequential Sentence Extraction

Given relevance and redundancy defined above, we aim to select important sentences from the document that maximize relevance and minimizes redundancy. We consider a weighted combination of the two criteria:

$$\max_{S \subseteq D} \lambda_1 \text{Rel}(S, D) + \lambda_2 \text{Red}(S) \text{ s.t. } |S| \leq k, \tag{7}$$

where $|S|$ denotes the number of sentences in the summary. This is a combinatorial problem that is expensive to solve when k is large. Therefore, we solve it approximately by selecting sentences sequentially in a greedy fashion. Given the previously selected sentences, we select the next sentence from the document that maximally improves the objective (7) until k sentences are selected. Our full algorithm is shown in Algorithm 1, where $\Delta(s)$ denotes the change incurred by the new summary $S \cup \{s\}$.

4 Experiments

Datasets. We evaluate on a diverse range of domains: (i) news articles: CNN-Dailymail (CNN-DM) (See et al., 2017) and XSum (Narayan et al., 2018); (ii) personal anecdotes: Reddit (Ouyang

Algorithm 1: ExtractSentences(D, k)

```
$S := \emptyset$
for $j \leftarrow 1$ to $k$ do
    for $s \in D \setminus S$ do
        $\Delta(s) := \lambda_1 \text{Rel}(\{s\}, D) + \lambda_2 \sum_{s' \in S} \text{Red}(s', s)$
    end
    $s_j := \text{argmax}_{s \in D \setminus S} \Delta(s)$
    $S := S \cup \{s_j\}$
end
return $S$
```

et al., 2017) and Reddit-TIFU (Kim et al., 2019); (iii) scientific articles: PubMed (Kedzie et al., 2018). Further, these datasets allow us to evaluate on highly abstractive summarization (XSum), small data (Reddit), and long documents (PubMed). Details of the datasets are shown in Appendix A.

Baselines. We compare our approach against the following unsupervised extraction methods: (i) **heuristic**: lead-k which selects the first k sentences. (ii) **similarity-based**: TextRank (Barrios et al., 2016; Mihalcea and Tarau, 2004) and PacSum (Zheng and Lapata, 2019) which use graph-based selection with similarity metrics based on tf-idf sentence vectors and BERT embeddings, respectively.[2]

To ablate the contribution of PMI, we also include a variant of our algorithm which uses cosine similarity of tf-idf sentence representations to measure redundancy and relevance of sentence pairs. Additionally, we include two **reference** methods: oracle extraction[3] and the state-of-the-art supervised approaches: For CNN-Dailymail, XSum, Reddit-TIFU and Pubmed the results are from Zhang et al. (2019) and for Reddit from Ouyang (2019).

Language model fine-tuning. For our method, we use the pre-trained GPT-2 large model (Radford et al., 2019) to calculate the PMI. To adapt the language model to specific domains, we fine-tune it on the training documents (excluding the gold summaries) in each dataset. To make it better fit our task of estimating the probability of one sentence given another, as described in Section 2, we fine-tune GPT-2 on two-sentence segments (as opposed to a long stream of tokens). Each segments consist of a pair of sentences from the document.[4]

Implementation details. We preprocess the documents with spaCy (Honnibal and Montani, 2017) to split the text into sentences. All hyperparameters are tuned on 200 randomly sampled document-summary pairs selected from the validation set to optimize the Rouge-1 F-measure, including λ_1 and λ_2 in our method which balances relevance and redundancy scores in Equation (7), the number of keywords in TextRank and the number of sentences to select for all extractive methods.

To select the values of λ_1 and λ_2 we run a grid search at intervals of 0.1 from -2 to 2 for both. For all datasets, the best weighting was 2 for relevance and -2 for redundancy.[5] The weights are intuitive because we want to maximize relevance (λ_1) and minimize redundancy (λ_2). For the Lead-k baselines, k was 3 for CNN-Dailymail, Reddit-TIFU and XSum, 4 for Reddit and 9 for PubMed.

4.1 Results

We evaluate all methods on the five datasets using the Rouge-1/2/L (Lin and Rey, 2004) F-measure. Table Table 1 shows our main results.

PMI vs similarity. We first compare PMI and tf-idf in our framework. The results (Ours (PMI) vs Ours (tf-idf)) show that measuring relevance and redundancy using PMI is better than word overlap, especially on narratives (Reddit). The Reddit writing style is less reporting facts like in news and more describing a sequence of events, thus it is helpful to capture the dependence between events. We show an example contrasting the two metrics in Table 2. Further, our method achieves better or comparable results across all datasets compared to other similarity-based methods. The oracle extraction is predictably the best result across all datasets. Our extractive approach outperforms the supervised baseline on the smallest dataset, Reddit, demonstrating the utility of unsupervised approaches in this setting. More examples are shown in Appendix B.

---

[2] For TextRank, Zheng and Lapata (2019) showed that tf-idf works better than BERT embeddings so we follow the same.

[3] We select sentences greedily to optimize the Rouge-1 score against the reference (Nallapati et al., 2017).

[4] Our preliminary results show that this works slightly better than standard LM fine-tuning.

[5] Note that there are multiple optimal values, e.g. 1 and -1, the selected values are based on tie-breaking in our algorithm.

Dataset	CNN-DM			XSum			Reddit			Reddit-TIFU			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Lead-k	39.69	17.22	24.82	**19.48**	2.62	12.49	26.52	7.05	18.80	15.4	2.24	10.81	37.34	10.54	18.00
TextRank	34.11	12.78	22.51	19.04	3.05	**12.64**	23.76	7.90	16.41	18.70	3.49	13.08	46.73	17.28	22.28
PacSum	**40.26**	**17.55**	**24.92**	19.44	2.71	12.44	-	-	-	-	-	-	-	-	-
Ours (tf-idf)	34.17	12.99	22.44	18.14	3.14	12.31	24.14	8.19	16.96	18.24	3.18	12.45	46.54	17.77	22.66
Ours (PMI)	36.68	14.52	23.32	19.07	**3.22**	12.47	**28.18**	**8.60**	**20.26**	**18.93**	**3.72**	**13.11**	**46.84**	**17.81**	**23.01**
Oracle Ext	53.53	29.56	37.21	31.85	7.51	20.87	36.63	15.24	27.87	34.96	10.75	24.91	54.88	23.72	28.30
Supervised	44.17	21.47	41.11	47.21	24.56	39.25	20.90	5.40	18.90	26.63	9.01	21.6	45.09	19.56	27.42

Table 1: Rouge-1/2/L F-Measure scores of unsupervised extractive methods, oracle extraction, and SOTA results using surpervised learning on CNN-DM, XSum, Reddit, Reddit-TIFU and PubMed. The best results among unsupervised methods are in bold. Our PMI-based extractor outperforms similarity-based methods on non-news domains, and is comparable on news domains. (PacSum results on non-news domains are not reported as the released model is fine-tuned only on the news domain.)

Gold Summary	Ours(PMI)	PacSum
Cillian McCann was filmed by his mother Toni at seven weeks old. In the clip, the little boy can clearly be seen trying to speak to his family. After several attempts he manages to say "hello". The average child can say six words by the time they reach 18 months.	Whose adorable son Cillian said his first word at just seven weeks old. **In the video Cillian is seen struggling to get his word out, but with a bit of encouragement from his mother he finally says hello.** Toni says that Cillian was very alert from a young age and had been trying to make out words since he was just five weeks old .	Most parenting advice says you don't have to worry if your baby doesn't start speaking until around 18 months. The tiny tot, who is now nine weeks old, was filmed by his 36-year-old mother who says that she knew he had been trying to communicate for a while. **Cillian has three older sisters, Toni revealed that her little girls, Sophie(bottom right), Eva(bottom left) and Ellie(top), did not start talking at such an early age.**

Table 2: Example of summaries selected by PMI vs PacSum. PacSum selected the sentence (bolded) about the child's siblings. It mentions talking at an early age which is the main theme of the article. However, it does not inform us how the child started talking in the video. Our method using PMI selected sentences focusing on the content of the video and the child.

Extraction algorithm. Another important component in extractive methods is the sentence selection/decoding algorithm. The most common approach is to select sentence greedily according to certain objective. TextRank uses a graph-based method inspired by PageRank. However, in Table 1 we see that TextRank and Ours (tf-idf) (using greedy selection) achieve similar results, showing that the selection algorithm does not have a large impact, which is also found by Zheng and Lapata (2019).

Ablation study. To understand the contribution of relevance and redundancy in the proposed metric, we conducted an ablation study on CNN/DM and Reddit-TIFU. In Table 3, we see that relevance alone does well, but augmenting it with redundancy obtains the best performance across all metrics. Minimizing redundancy alone works poorly because it cannot identify important content.

Position bias in PacSum. One may wonder why PacSum and lead-k significantly outperform other extractive methods on CNN-DM. We hypothesize that they take advantage of the lead bias on CNN/DM. Figure 1 shows histograms of the positions of summary sentences selected by our method and PacSum on CNN/DM. Notably, 82.3% of sentences selected by PacSum were in the first three, and this value drops to 21.4% in our method. This provides an empirical explanation as to why PacSum is so far ahead of the other extractive approaches on CNN-DM. The authors (Zheng and Lapata, 2019) also noted a drop in performance when positional information was removed in PacSum. In addition, their performance degrades on XSum (which doesn't suffer from lead bias). A

Method	CNN-DM			Reddit-TIFU		
	R-1	R-2	R-3	R-1	R-2	R-3
Only Rel.	35.17	13.79	22.91	18.45	3.67	12.90
Only Red.	23.85	6.47	15.48	16.41	2.35	11.66
Rel. and Red.	**36.68**	**14.52**	**23.32**	**18.93**	**3.72**	**13.11**

Table 3: Ablation Results on CNN-DM and Reddit-TIFU. We observe that the combination of relevance and redundancy yields the best performance across all metrics.

concurrent work (Xu et al., 2020) performed a similar analysis, observing the reliance on position information of PacSum in the news domain.

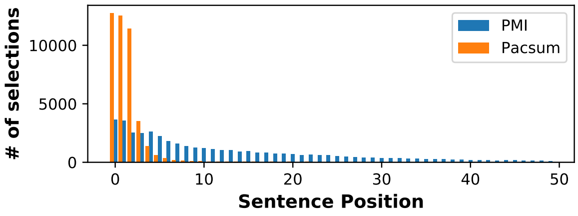


Figure 1: Histogram of the position of sentences selected by our method and PacSum on CNN/DM. PacSum uses position information which allows it to take advantage of the lead bias. In contrast, our method is position-agnostic but still captures the fact that earlier sentences are more important in news articles.

5 Related Work

Similarity-based summarization. Most unsupervised extractive summarization methods rely on sentence pair similarity as a proxy for importance (Zheng and Lapata, 2019; Mihalcea and Tarau, 2004; Barrios et al., 2016; Erkan and Radev, 2004) and use variants of the Pagerank (Page et al., 1999) algorithm to perform selection. We propose PMI as an alternative and compare it to a concurrent similarity based approach.

Leveraging pretrained language models. Scoring a sentence on its ability to predict subsequent sentences using a language model has been adapted for sentence summarization. (West et al., 2019). Zhou and Rush (2019) use two language models, a generic pretrained language model for contextual matching and a task specific one to enforce fluency. A concurrent work (Xu et al., 2020) used sentence level transformer self-attentions and probabilities to rank sentences for unsupervised extractive summarization. We use the language model to compute PMI, which then scores sentences on relevance and redundancy as criteria for selection.

Diversity in content selection. Maximal Marginal Relevance (Goldstein and Carbonell, 1998) has been used to produce summaries that prioritize diversity in selected content. A similarity metric is used to produce summaries based on similarity to a query while maintaining diversity among selected sentences in various domains (Chandu et al., 2017). This can be seen as analogous to our comparison to our approach tf-idf based selection.

6 Conclusion and Discussion

We propose metrics for relevance and redundancy in summarization based on pointwise mutual information, and an unsupervised extractive summarization algorithm using pre-trained language models. We demonstrate the effectiveness of our method on both news and non-news domains. Supervised models often learn the lead bias in the datasets and degrade significantly when such hues are absent (Kedzie et al., 2018). Furthermore, even human evaluation of content selection has large variance (Nenkova et al., 2007; Chaganty et al., 2018). Our work is a first step towards formalizing a notion of importance that informs algorithm design in summarization. We believe it is important to have a better formalization of content importance in terms of both task definition/evaluation and modeling. We hope our results will spur more work in this direction.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*.

J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner. 2005. The AMI meeting corpus: A pre-announcement. *Springer Lecture Notes in Computer Science*, 3869.

A. Chaganty, S. Mussmann, and P. Liang. 2018. The price of debiasing automatic metrics in natural language evaluation. In *Association for Computational Linguistics (ACL)*.

Khyathi Chandu, Aakanksha Naik, Aditya Chandrasekar, Zi Yang, Niloy Gupta, and Eric Nyberg. 2017. Tackling biomedical text summarization: OAQA at BioASQ 5B. In *BioNLP 2017*, pages 58–66, Vancouver, Canada,. Association for Computational Linguistics.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Jade Goldstein and Jaime Carbonell. 1998. Summarization: (1) using MMR for diversity- based reranking and (2) evaluating summaries. In *TIPSTER TEXT PROGRAM PHASE III: Proceedings of* *a Workshop held at Baltimore, Maryland, October 13-15, 1998*, pages 181–195, Baltimore, Maryland, USA. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

C. Kedzie, K. McKeown, and D. III. 2018. Content selection in deep learning models of summarization. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of Reddit posts with multi-level memory networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2519–2531, Minneapolis, Minnesota. Association for Computational Linguistics.

C. Lin and M. Rey. 2004. Looking for a few good metrics: ROUGE and its evaluation. In *NTCIR Workshop*.

Y. Liu and M. Lapata. 2019. Text summarization with pretrained encoders. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

S. Narayan, S. B. Cohen, and M. Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Empirical Methods in Natural Language Processing (EMNLP)*.

A. Nenkova, R. J. Passonneau, and K. McKeown. 2007. The pyramid method: Incorporating human content selection variation in summarization evaluation. In *ACM Transactions on Speech and Language Processing*.

J. Ouyang, S. Chang, and K. McKeown. 2017. Crowdsourced iterative annotation for narrative summarization corpora. In *European Association for Computational Linguistics (EACL)*.

Jessica Jin Ouyang. 2019. Adapting automatic summarization to new sources of information. Link: https://doi.org/10.7916/d8-5nar-6b61.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Maxime Peyrard. 2019. A simple theoretical model of importance for summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073.

D. R. Radev, H. Jing, and M. Budzikowska. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the NAACL-ANLP 2000 Workshop: Automatic Summarization*.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. Language models are unsupervised multitask learners.

A. See, P. J. Liu, and C. D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Association for Computational Linguistics (ACL)*.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle.

Shusheng Xu, Xingxing Zhang, Yi Wu, Furu Wei, and Ming Zhou. 2020. Unsupervised extractive summarization by pre-training hierarchical transformers.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. *arXiv preprint arXiv:1906.03508*.

Jiawei Zhou and Alexander M Rush. 2019. Simple unsupervised summarization by contextual matching. *arXiv preprint arXiv:1907.13337*.

A Dataset Details

CNN/DM is known to have a very strong extractive Lead-3 baseline as is common in the news domain. XSum contains summaries of BBC news articles but is highly abstractive in nature. The Reddit dataset is a small corpus of around 500 personal stories shared on Reddit with abstractive and extractive summaries. For Reddit-TIFU, we use the TIFU-long subset as used in Zhang et al. (2019). The Reddit-TIFU didn't come with a train split and since we look at unsupervised methods, we used 200 pairs as validation data to decide parameters and report test results on the rest. The PubMed dataset contains longer medical journal articles with the corresponding abstracts functioning as the groundtruth summaries.

Dataset	Train	Validation	Test
CNN-DM	287,113	13,368	11,490
XSum	204,045	11,332	11,334
Reddit	404	24	48
Reddit-TIFU	-	200	41139
PubMed	21,250	1,250	2,500

Table 4: Size of train, validation and test splits of datasets used

B Analysis of Examples

Table 5 shows some example summaries from the CNN-Dailymail validation set in comparison to extractive candidate summaries obtained for the correponding documents using the baseline Lead-3 approach, our Interpolated PMI based approach and the PacSum approach (Zheng and Lapata, 2019) that uses sentence similarity to obtain state-of-the-art Rouge results on the dataset. These are shown to highlight the difference between using PMI and similarity for sentence selection. In the first example, the gold summary details about how medical information regarding two patients was leaked to sales representatives. The similarity based approach selects all three sentences associated with only one of the patients whereas the PMI based approach yields a summary that contains information about both. Once the first sentence concerning the first patient is selected, all sentences associated with it are penalised by a corresponding amount resulting in a more well rounded selection of information. Similarly, in the second example the sentence that details how the child overcame his initial struggles to speak after some encouragement from his parent was only selected by the PMI approach. This summarises exactly what happens in the video clip being spoken about and the same point is highlighted even in the gold summary. The contents of the paragraph can be easily understood when the information about the clip is used as a context. The rest of the paragraph talks goes into how the child spoke faster than his siblings which explains the selection made by PacSum. The third example highlights the issue of PacSum being identical to the Lead-3 baseline by modelling position information present in the dataset. In the fourth example, the PacSum based approach selects two sentences with quotes that have negative connotations while the one selected by PMI about how the protagonist could not forgive himself could serve to better explain the need for an intervention on the Dr. Phil show.

The purpose of this is to highlight that the intangible nature of the definition of relevance. The content selected varies between PMI and sentence similarity and each might find an application in the right setting. It again highlights the need to consider what one expects from the summarisation task.

Gold	**Interpolated PMI**	**PacSum**	**Lead3**
Tim Esworthy, 66, has a prosthetic limb after losing his leg in a workplace incident and said he had been targeted by cold callers selling products easing joint pain. Christine Lewis, 62, is wheelchair-bound following a brain haemorrhage. She is also on list of people obtained by the mail and has been targeted by stairlift salesmen.	Tim Esworthy, 66, from colchester, was 'absolutely appalled' to find his private medical details had been sold. Christine Lewis, who is recovering from a brain haemorrhage she had 12 years ago, was on a list of people who have mobility problems obtained by the mail and has been targeted by stairlift salesmen cold calling her. 'They shouldn't have my information, especially if they know I'm disabled because they are targeting me because they think I'm vulnerable.'	Tim Esworthy, 66, from colchester, was 'absolutely appalled' to find his private medical details had been sold. Retired financial services manager Tim Esworthy was 'absolutely appalled' to find his private medical details had been sold . They know they can target vulnerable people because they have their medical information.	Tim Esworthy, 66, from Colchester, was 'absolutely appalled' to find his private medical details had been sold. Case 1 : Pensioner who lost leg at work. Retired financial services manager Tim Esworthy was 'absolutely appalled' to find his private medical details had been sold.
Cillian McCann was filmed by his mother Toni at seven weeks old. In the clip, the little boy can clearly be seen trying to speak to his family. After several attempts he manages to say "hello". The average child can say six words by the time they reach 18 months.	Whose adorable son Cillian said his first word at just seven weeks old. In the video Cillian is seen struggling to get his word out, but with a bit of encouragement from his mother he finally says hello. Toni says that Cillian was very alert from a young age and had been trying to make out words since he was just five weeks old .	Most parenting advice says you don't have to worry if your baby doesn't start speaking until around 18 months. The tiny tot, who is now nine weeks old, was filmed by his 36-year-old mother who says that she knew he had been trying to communicate for a while. Cillian has three older sisters, Toni revealed that her little girls, Sophie(bottom right), Eva(bottom left) and Ellie(top), did not start talking at such an early age.	Most parenting advice says you don't have to worry if your baby doesn't start speaking until around 18 months. Whose adorable son Cillian said his first word at just seven weeks old. The tiny tot, who is now nine weeks old, was filmed by his 36-year-old mother who says that she knew he had been trying to communicate for a while.
Ashleigh humphrys, 20, died in a hit-and-run early on sunday morning. Police believe the driver of the car was heading to work. A man is assisting police with their investigations after the death. Ms Humphrys was walking home after celebrating her birthday with friends. A security guard rang police after she was walking disorientated. CCTV footage shows two taxis stop near her before she was struck and put hazard lights on. Then a car drove past the taxis, mounted the footpath before swerving back onto the road and driving off. A taxi is said to have been seized and police are talking to a person 'within the vicinity' at the time of the incident.	Brisbane woman Ashleigh Humphrys died in a hit-and-run incident after deciding to walk from Toowong to her Seventeen Mile Rocks home in Brisbane after having an argument with a friend while they were out celebrating her 20th birthday. Only moments later the guard, who was still on the phone to police while driving around trying to find Ms Humphrys, discovered her dead on the road at the city end of the western freeway. Just before Ms Humphrys was hit, CCTV footage shows two taxis stop near the woman and put their hazard lights on before a car drove past the taxis, mounted the footpath and then swerved back onto the road before driving off.	The driver of a car that hit and killed a young woman in the early hours on Sunday morning was on the way to work, police believe. Brisbane woman Ashleigh Humphrys died in a hit-and-run incident after deciding to walk from Toowong to her Seventeen Mile Rocks home in Brisbane after having an argument with a friend while they were out celebrating her 20th birthday. Now, after it was revealed that a man was assisting police with their investigations, officers have said they believe he was on his way to work and went to his shift as normal on sunday, the Courier Mail reported.	The driver of a car that hit and killed a young woman in the early hours on Sunday morning was on the way to work, police believe. Brisbane woman Ashleigh Humphrys died in a hit-and-run incident after deciding to walk from Toowong to her Seventeen Mile Rocks home in Brisbane after having an argument with a friend while they were out celebrating her 20th birthday. Now, after it was revealed that a man was assisting police with their investigations, officers have said they believe he was on his way to work and went to his shift as normal on sunday, the Courier Mail reported.
Dr. Phil Mcgraw staged a highly-charged intervention with Nick Gordon last Thursday. With his mother, Michelle, by his side a sobbing Gordon talked about missing Bobbi Kristina. Gordon is now in rehab after the intervention having been drinking heavily and taking xanax. Girlfriend Bobbi Kristina has been in a medically induced coma since January 31 and Gordon has not been allowed to see her. The dramatic intervention will air Wednesday on the Dr Phil show.	Amid scenes of high emotion, an often incoherent Gordon admitted drinking heavily and taking xanax, for which he has a prescription, in an attempt to deal with life since Bobbi Kristina was found face down and unresponsive in her bathtub on January 31. Breakdown: With his mother, Michelle, by his side Nick Gordon struggles to stay coherent as he is questioned by Dr Phil. According to his mother, Michelle, Gordon can not forgive himself for his 'failure' to revive Bobbi Kristina	Weeping and wailing Nick Gordon, the troubled fiancé of Bobbi Kristina Brown, has admitted that he has twice tried to kill himself and confessed: "I'm so sorry for everything." Asked if he still intended to kill himself he said: "If anything happens to Krissi I will." Amid scenes of high emotion, an often incoherent Gordon admitted drinking heavily and taking xanax, for which he has a prescription, in an attempt to deal with life since Bobbi Kristina was found face down and unresponsive in her bathtub on January 31.	Weeping and wailing Nick Gordon, the troubled fiancé of Bobbi Kristina Brown, has admitted that he has twice tried to kill himself and confessed : "I'm so sorry for everything." Gordon, 25, was speaking to Dr Phil Mcgraw in a dramatic intervention due to air on Wednesday, Daily Mail online can reveal. Asked if he still intended to kill himself he said: "If anything happens to Krissi I will."

Table 5: Example summaries obtained from the CNN-Dailymail validation set compared to the corresponding extractive candidate summary obtained using Interpolated PMI, PacSum (State-of-the-art unsupervised summary using sentence similarity) and the Lead-3 Baseline

Context-aware Neural Machine Translation with Mini-batch Embedding

Makoto Morishita^{1,2}, Jun Suzuki^{2}, Tomoharu Iwata^{1}, Masaaki Nagata^{1}
NTT Communication Science Laboratories, NTT Corporation^{1}
Tohoku University^{2}
`{makoto.morishita.gr, tomoharu.iwata.gy,`
`masaaki.nagata.et}@hco.ntt.co.jp`
`jun.suzuki@ecei.tohoku.ac.jp`

Abstract

It is crucial to provide an inter-sentence context in Neural Machine Translation (NMT) models for higher-quality translation. With the aim of using a simple approach to incorporate inter-sentence information, we propose **m**ini-**b**atch **e**mbedding (MBE) as a way to represent the features of sentences in a mini-batch. We construct a mini-batch by choosing sentences from the same document, and thus the MBE is expected to have contextual information across sentences. Here, we incorporate MBE in an NMT model, and our experiments show that the proposed method consistently outperforms the translation capabilities of strong baselines and improves writing style or terminology to fit the document's context.[1]

1 Introduction

Current standard neural machine translation (NMT) models (Sutskever et al., 2014; Bahdanau et al., 2015; Luong et al., 2015; Vaswani et al., 2017) translate sentences in a sentence-by-sentence manner. However, some have argued that it is critical to consider the inter-sentence context in handling discourse phenomena (Hardmeier, 2012), which include coherence, cohesion, coreference (Bawden et al., 2018; Nagata and Morishita, 2020), and writing style (Yamagishi et al., 2016). To correctly translate these linguistic features, some works provide additional context information to an NMT model by concatenating the previous sentence (Tiedemann and Scherrer, 2017), applying a context encoder (Bawden et al., 2018; Miculicich et al., 2018; Voita et al., 2018), or using a cache-based network (Tu et al., 2018; Kuang et al., 2018).

Most of the previous studies have considered only a few previous context sentences. Several methods, such as the cache-based network, consider long-range context but heavily modify the standard NMT models and require additional training/decoding steps. Our goal is to make a simple but effective context-aware NMT model, which does not require heavy modification to standard NMT models and can handle a wider inter-sentence context. To this end, we propose a method to create an embedding that represents the contextual information of a document. To create this embedding, we focused on the mini-batch, which is commonly used in NMT training and decoding for efficient GPU computation. We modified the mini-batch creation algorithm to choose sentences from a single document and created an embedding that represents the features of the mini-batch. We call this embedding **mini-batch embedding (MBE)** and incorporate it in the NMT model to exploit contextual information across the sentences in the mini-batch.

Our main contributions can be summarized as follows: (i) We introduce mini-batch embedding to represent the features of sentences in a mini-batch. (ii) We incorporate mini-batch embedding in NMT to achieve simple context-aware translation and find that our approach improves translation performance by up to 1.9 BLEU points.

2 Neural Machine Translation

The current NMT model $f(\cdot)$ generates a sequence of target sentence tokens $\boldsymbol{y} = (y_1, \ldots, y_t)$ given a sequence of source sentence tokens $\boldsymbol{x} = (x_1, \ldots, x_s)$: $\boldsymbol{y} = f(\boldsymbol{x}; \theta)$, where θ is a set of model parameters and s and t are the numbers of source and target sentence tokens. The model parameters are trained by minimizing the loss function:

$$\mathcal{L}_{\mathrm{NMT}}(\theta) = -\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathbb{D}} \log P(\boldsymbol{y}|\boldsymbol{x}; \theta), \quad (1)$$

---

[1] Our implementation is publicly available: `https://github.com/nttcslab-nlp/mbe-nmt`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2513–2521
April 19 - 23, 2021.

where $\mathbb{D}$ is a set of bilingual sentence pairs. Since the model only uses a single sentence as its input, it does not consider the inter-sentence context.

3 Context-aware NMT with Mini-batch Embedding

To exploit the inter-sentence context in NMT with a simple modification, we propose **m**ini-**b**atch **e**mbedding (MBE) to represent the features of sentences in the mini-batch. Figure 1 shows an overview of how we create mini-batch embedding and incorporate it in the NMT model.

3.1 Mini-batch Embedding

Let $\boldsymbol{B} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$ be a mini-batch, where $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is a pair of source/target sentences. Normally, we randomly select them from all of the training data to create a mini-batch $\boldsymbol{B}$. However, for our method, we choose sentence pairs from the same document to create a mini-batch.

Let $g_{\text{enc}}(\cdot)$ be a single Transformer encoder layer. We first compute sentence-wise contextualized embeddings $\boldsymbol{E}_i = (\boldsymbol{e}_{i,1}, \ldots, \boldsymbol{e}_{i,s_i})$ as $\boldsymbol{E}_i = g_{\text{enc}}(\boldsymbol{x}_i; \phi)$, where s_i is the number of tokens in $\boldsymbol{x}_i$ and ϕ are the model parameters. MBE $\boldsymbol{z} \in \mathbb{R}^e$ is computed:

$$\boldsymbol{z} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{v}_i, \quad \boldsymbol{v}_i = \frac{1}{s_i} \sum_{j=1}^{s_i} \boldsymbol{e}_{i,j}, \tag{2}$$

where e is a hidden dimension of the NMT model. We use mean pooling[2] to make both sentence embeddings $\boldsymbol{v}_i$ and MBE $\boldsymbol{z}$. By adopting this procedure, we expect MBE $\boldsymbol{z}$ to have inter-sentence context features, which is desirable for a context-aware NMT.

Note that we ignore the order of sentences in a document. This is a beneficial trait because this method is also applicable to corpora with document boundaries but without in-document sentence order, such as ParaCrawl (Esplà et al., 2019).

3.2 Learning NMT with Mini-batch Embedding

To use inter-sentence information, we modify the NMT model by adding MBE to the input:

$$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{z}; \theta). \tag{3}$$

We concatenated MBE to the input word embeddings, and the model uses MBE as the first input token (Fig. 1). Now the encoder/decoder takes $s+1$ and $t+1$ embeddings.

The Transformer encoder layer for MBE was jointly trained with the NMT model by modifying the loss function in Eq. (1):

$$\mathcal{L}_{\text{NMT}}(\theta, \phi) = -\sum_{\boldsymbol{B} \in \mathbb{D}'} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \boldsymbol{B}} \log P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{B}; \theta, \phi), \tag{4}$$

where $\mathbb{D}'$ is a set of mini-batches created from $\mathbb{D}$.

3.3 Mini-batch Embedding Gate

The MBE may degrade the translation performance when the NMT model does not need any context information to translate the mini-batch or the MBE fails to contain important information for translation. To deal with such cases, we aim to make the model estimate how important MBE is for each mini-batch. Thus we added a mini-batch embedding gate to determine MBE's importance.

In this setting, we prepared two types of mini-batches for training: (i) sentences from the same document and (ii) sentences from different documents. Then we trained a binary classifier that predicts whether the sentences in the mini-batch are selected from the same document:

$$P(d|\boldsymbol{z}) = \text{softmax}(\boldsymbol{W}\boldsymbol{z}), \tag{5}$$

where $\boldsymbol{W} \in \mathbb{R}^{2 \times e}$ is a parameter matrix and d is a binary value that takes 1 if the sentences in the mini-batch are selected from the same document.

To train the classifier, we minimize the loss function:

$$\mathcal{L}_{\text{MB}}(\psi) = -\sum_{(d,\boldsymbol{B}) \in \mathbb{D}'} \log P(d|\boldsymbol{B}; \psi), \tag{6}$$

where ψ is a set of parameters for the classifier. For training, we mix the two types of mini-batches at the same ratio.

Concretely, we jointly minimize the NMT and the classifier loss functions:

$$\mathcal{L}(\theta, \phi, \psi) = \mathcal{L}_{\text{NMT}}(\theta, \phi) + \lambda \mathcal{L}_{\text{MB}}(\psi), \tag{7}$$

where λ is a hyperparameter used to control the weight of the classifier loss. We use the value predicted by the classifier as a gate. Our new weighted MBE is

$$\tilde{\boldsymbol{z}} = \alpha \boldsymbol{z}, \tag{8}$$

where $\alpha = P(d = 1|\boldsymbol{z})$, and we change $\boldsymbol{z}$ in Eqs. (3) to $\tilde{\boldsymbol{z}}$.

---

[2] This approach was inspired by Reimers and Gurevych (2019), who successfully created sentence embeddings from BERT embeddings (Devlin et al., 2019) by mean pooling.

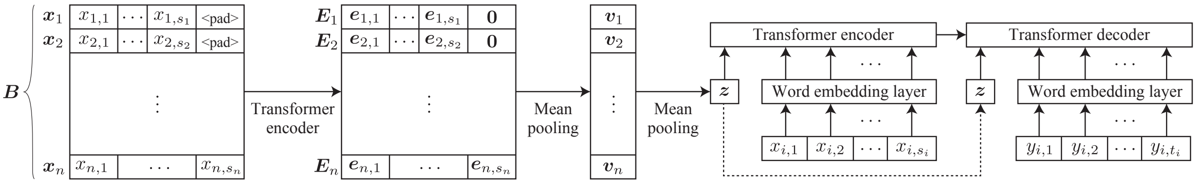


Figure 1: Overview of context-aware NMT with mini-batch embedding. $\boldsymbol{x}_i$ is a sequence of source tokens, $\boldsymbol{B}$ is a mini-batch that has n sentences, $\boldsymbol{E}_i$ is sentence-wise contextualized embeddings computed by a Transformer encoder, $\boldsymbol{v}_i$ is a sentence vector, and $\boldsymbol{z}$ is mini-batch embedding. We pad short sentences with a special <pad> token to adjust their length to the longest sentence in the mini-batch.

4 Experiments

4.1 Compared Models

We used four settings as baselines:

Baseline 6 Enc-Layers is the original Transformer NMT model with six encoder-decoder layers.

Baseline 7 Enc-Layers resembles Baseline 6 Enc-Layers, but the number of encoder layers was changed to seven. Since our MBE model requires an additional Transformer encoder layer, this model has a comparable number of parameters as the following MBE models.

2-to-1 is the context-aware translation model proposed by Tiedemann and Scherrer (2017) that translates a pair of previous and current source sentences into a target sentence. Two source sentences are concatenated with a special sentence boundary token. This method is known as a strong baseline for context-aware NMT (Bawden et al., 2018; Voita et al., 2018). Other settings are identical to those of Baseline 6 Enc-Layers[3].

DocRepair is another recent context-aware translation model, which uses two-step decoding (Voita et al., 2019). The first step generates 1-best translation with a sentence-level NMT model given a single sentence. The second step generates document-level translation given 1-best translations of four consecutive sentences concatenated with a special token.

We compared our proposed methods with the following settings:

MBE Enc resembles Baseline 6 Enc-Layers but uses MBE in the encoder.

MBE Enc w/o Gate resembles MBE Enc, but it does not use the MBE gate described in Section 3.3.

MBE Dec uses MBE in the decoder.

MBE Enc/Dec uses MBE in both the encoder and the decoder.

4.2 Experimental Settings

Datasets/Evaluation We trained Japanese-English NMT models. As training data, we used the JParaCrawl corpus (Morishita et al., 2020). JParaCrawl was created by crawling the web and aligning parallel sentences, and each sentence-pair has a URL from which the sentences were taken. In this experiment, we regarded the sentences from the same URL as a document.

We used several test sets with document boundaries: (i) scientific paper excerpts (ASPEC (Nakazawa et al., 2016)), (ii) news (newsdev2020 from WMT20 news translation shared task[4]), and (iii) TED talks (tst2012 from IWSLT translation shared task (Cettolo et al., 2012)). As a dev set to tune the NMT model, we used the ASPEC dev split. See Section A.1 in the Appendix for corpus statistics and detailed preprocessing steps.

To evaluate the translation performance, we used `sacreBLEU`[5] (Post, 2018) and report the BLEU scores (Papineni et al., 2002).

Model Configurations We used the Transformer model as an NMT model (Vaswani et al., 2017). Our hyperparameters were based on the "big" settings defined by Vaswani et al. (2017). For the MBE experiments, we set λ in Eq. (7) to 1.0. We set the mini-batch size to 3,000 tokens. If the tokens in a document were larger than this size, we

---

[3] Since our training data have document boundaries but the in-document sentence orders were shuffled, we randomly selected one in-document sentence and used it as previous context. For dev/test sets, we used the original sentence order.

[4] We used newsdev2020 as a test set because no official test set for English-Japanese was available at the time of writing. Since we did not use newsdev2020 for tuning the model, there is no problem with using it as a test set.

[5] The signature is `BLEU+case.mixed+lang.en-ja+numrefs.1+smooth.exp+tok.ja-mecab-0.996-IPA+version.1.4.9`

Model	ASPEC	WMT	IWSLT
Baseline 6 Enc-Layers (Vaswani et al., 2017)	26.2	18.4	12.0
Baseline 7 Enc-Layers (Vaswani et al., 2017)	26.9 (+0.7)	18.7 (+0.3)	11.9 (−0.1)
2-to-1 (Tiedemann and Scherrer, 2017)	27.0 (+0.8)	19.2 (+0.8)	12.9 (+0.9)
DocRepair (Voita et al., 2019)	27.9 (+1.7)	19.3 (+0.9)	12.3 (+0.3)
MBE Enc	28.0 (+1.8)	19.9 (+1.5)	12.2 (+0.2)
MBE Enc w/o Gate	28.0 (+1.8)	19.4 (+1.0)	13.0 (+1.0)
MBE Dec	**28.1 (+1.9)**	19.9 (+1.5)	**13.8 (+1.8)**
MBE Enc/Dec	**28.1 (+1.9)**	**20.0 (+1.6)**	13.4 (+1.4)

Table 1: BLEU scores for test sets: Values in brackets show score differences to "Baseline 6 Enc-Layers" model. The highest score in each test set is highlighted in bold.

split the document into several mini-batches[6]. If the tokens in a document are smaller, we put all tokens into a single mini-batch[7]. See Section A.2 in the Appendix for detailed hyperparameters and training settings.

4.3 Experimental Results and Analysis

Translation Performance Table 1 summarizes the model performance on several test sets. See Table 3 in the Appendix for the dev set performance. The results show that the scores of the proposed methods surpass the baseline as well as the stronger baselines that used seven encoder layers or the existing context-aware models.

Translation Examples Figure 2 shows an example translation of a sentence from the scientific paper excerpts (ASPEC test set). In this example, the word "mentions" is translated in two ways. The baseline system translated the word as "言及しています", which is a colloquial expression. In contrast, the proposed method translated it as "述べる", which suits usage in scientific papers. This shows that MBE could change the writing style to one that is more appropriate for scientific papers compared to the baseline.

Figure 3 shows another example, which is from TED talks (tst2012). This example shows how our model could change the translation of the word "you". Our method translated this word as "君", which is friendlier than the baseline output "あなた". In this document, "he" is a friendly old man, and thus the MBE output is more appropriate for this context.

These examples show that our method improved the writing style to fit the context and chose the appropriate word for the context. This indicates that MBE helped the NMT model by providing context information across the mini-batch.

Effect of Decoding Batch-size In the previous section, we discussed the translation performance given a document, which means that the sentences in the entire document are in a mini-batch. However, in practice, we sometimes have to translate a part of the document. To check the robustness of the model in such situations, we decoded the test set by limiting the number of sentences in a mini-batch.

Figure 4 shows the experimental results. The baseline model scores are identical to those in Table 1, since the model is immune to mini-batch size. Our MBE models achieve better performance when given a larger context. It reach comparable or better scores than the baseline model when given a single sentence or a smaller context. However, the model without using MBE gate (MBE Enc w/o Gate) showed a drastic drop in performance when translating a single sentence. This shows that the gate properly works to weigh the importance of MBE and improve performance.

5 Related Work

Context-aware NMT Tiedemann and Scherrer (2017) proposed a 2-to-1 (or 2-to-2) method that concatenates two source sentences and generates one (or two) target sentences. This is a simple model, but it only considers a previous sentence, while our method can make use of larger contexts. Junczys-Dowmunt (2019) extended the 2-to-2 method to document-to-document by concatenating all sentences in a document. Although they showed that the method is effective, it requires heavy computational cost since the NMT model

---

[6] We sorted the sentences in a document by their length when splitting the document into several mini-batches to maintain the training efficiency. Since the method focuses more on writing style and wording, we do not keep the original order of the sentences.

[7] In this case, the mini-batch size could be smaller than 3,000 tokens, since we did not want to mix up the sentences from different documents.

Source	The paper mentions the reliability assurance test and application technologies.
Reference	信頼性保証試験と適用技術を述べた
Baseline 6 Enc-Layers	この論文では、信頼性保証テストとアプリケーション技術について言及しています。
MBE Enc/Dec	本論文では,信頼性保証試験と応用技術について述べる.

Figure 2: Example translation of a sentence from scientific paper excerpts (ASPEC test set).

Source	He said, “I’m so proud of you.”
Reference	「君をすごく誇りに思うよ」
Baseline 6 Enc-Layers	彼は、“私はあなたをとても誇りに思います” と言いました。
MBE Enc/Dec	彼は「君をとても誇りに思う」と言いました。

Figure 3: Example translation of a sentence from TED talks (tst2012).

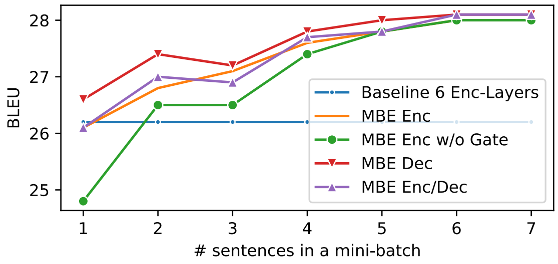


Figure 4: Relationship between the number of sentences in a mini-batch and BLEU scores on ASPEC test set.

has to process very long context. Miculicich et al. (2018) proposed a model that uses a hierarchical attention network to use previous context embeddings. However, their work can only use a few previous sentences as context, in contrast to our work that can use a larger context. Tu et al. (2018) and Kuang et al. (2018) proposed a cache-based approach to store longer context, while our work uses a much simpler architecture. Voita et al. (2019) proposed a method called DocRepair, one of the most recent context-aware NMT methods, that employs two decoding steps. It first translates a sentence by sentence-level NMT, and then the concatenated output is fed to a document-level model that outputs document-level translation. Although this is a promising method, it requires training of three sequence-to-sequence models to translate a single direction and needs two decoding steps, which slows down the translation. Our method has an advantage in that it only trains a single model and uses single-step decoding, which requires only a small computational cost.

NMT with Tags We used an MBE as the first input of the encoder/decoder. Our approach is similar to the work that uses special tags to control or provide additional information to NMT (Johnson et al., 2016; Takeno et al., 2017; Caswell et al., 2019). Johnson et al. (2016) added tags to a source sentence for indicating the target language in multilingual NMT models. Takeno et al. (2017) proposed a method that controls the target length or the domain by adding a tag to the decoder inputs. Caswell et al. (2019) used a tag to indicate the synthetic corpus (Sennrich et al., 2016). Our work, which automatically generates a tag (MBE) with the sentence in a mini-batch and uses a gate to control the importance of MBE, is different from the previous studies.

6 Conclusion

We proposed mini-batch embedding (MBE), which is a simple but effective method to represent contextual information across documents. We incorporated MBE in the NMT model, which enabled it to outperform competitive baselines. We found that our NMT model could choose the appropriate word and writing style to match the document context. An analysis showed that our model’s performance improves with a large context, but it still achieves comparable or even better performance than that of the baseline when translating a single sentence. Our future work includes applying MBE to other applications and improving the method to generate embeddings from a mini-batch.

Acknowledgments

We thank the four anonymous reviewers for their insightful suggestions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pages 1304–1313.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged back-translation. In *Proceedings of the 4th Conference on Machine Translation (WMT)*, pages 53–63.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 261–268.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*.

Miquel Esplà, Mikel Forcada, Gema Ramírez-Sánchez, and Hieu Hoang. 2019. ParaCrawl: Web-scale parallel corpora for the languages of the EU. In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, pages 118–119.

Christian Hardmeier. 2012. Discourse in statistical machine translation. a survey and a case study. *Discours. Revue de linguistique, psycholinguistique et informatique. A journal of linguistics, psycholinguistics and computational linguistics*, (11).

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. *arXiv preprint arXiv:1611.04558*.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the 4th Conference on Machine Translation (WMT)*, pages 225–233.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 66–75.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 66–71.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2947–2954.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2020. JParaCrawl: A large scale web-based English-Japanese parallel corpus. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC)*, pages 3603–3609.

Masaaki Nagata and Makoto Morishita. 2020. A test set for discourse translation from japanese to english. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC)*, pages 3704–3709.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of* *the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pages 48–53.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the 3rd Conference on Machine Translation (WMT)*, pages 1–9.

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

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the 3rd Conference on Machine Translation (WMT)*, pages 186–191.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3982–3992.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 86–96.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 3104–3112.

Shunsuke Takeno, Masaaki Nagata, and Kazuhide Yamamoto. 2017. Controlling target features in neural machine translation via prefix constraints. In *Proceedings of the 4th Workshop on Asian Translation (WAT)*, pages 55–63.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the 3rd Workshop on Discourse in Machine Translation*, pages 82–92.

Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. 2018. Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics (TACL)*, 6:407–420.

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

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. Context-aware monolingual repair for neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 877–886.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1264–1274.

Hayahide Yamagishi, Shin Kanouchi, Takayuki Sato, and Mamoru Komachi. 2016. Controlling the voice of a sentence in Japanese-to-English neural machine translation. In *Proceedings of the 3rd Workshop on Asian Translation (WAT)*, pages 203–210.

Data	Usage	Sentences	Documents
JParaCrawl v2.0	train	10,120,013	24,156
ASPEC (dev)	dev	1,790	400
ASPEC (test)	test	1,812	400
WMT (newsdev2020)	test	1,998	140
IWSLT (tst2012)	test	1,670	15

Table 2: Number of sentences and documents in train/dev/test sets

A Detailed Experimental Settings

In this section, we describe more detailed experimental settings.

A.1 Data/Evaluation

The number of sentences and documents contained in the train/dev/test sets are shown in Table 2. We tokenized the sentences into subwords with `sentencepiece` (Kudo, 2018; Kudo and Richardson, 2018) and set the vocabulary size to 32k for each language. For the training set, we removed sentences whose length exceeded 250 subword tokens. For the DocRepair method, we used the JParaCrawl corpus as data for both monolingual and bilingual document-level application.

A.2 Model Configurations

We used the Transformer model as an NMT model (Vaswani et al., 2017). Our hyperparameters are based on "big" settings defined by Vaswani et al. (2017) and have six encoder/decoder layers, 16 attention heads, and 1,024 dimensions for all of the hidden states except the feed-forward network hidden states that have 4,096 dimensions. We used a dropout with a probability of 0.3 (Srivastava et al., 2014). As an optimizer, we used Adam with $\alpha = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.98$ (Kingma and Ba, 2015). A root-square decay learning rate schedule was used with a linear warm-up of 4,000 steps (Vaswani et al., 2017). We clipped the gradients to avoid exceeding their norm of 1.0. For the MBE experiments, we set λ in Eq. (7) to 1.0 and set the per-GPU-batch-size to 3,000 tokens. Since large-batch training can reduce training time (Ott et al., 2018), we accumulated about 280k tokens for an update. Based on dev set perplexity, we trained the model for 24,000 iterations. We saved the model every 200 iterations and averaged the last eight model parameters for decoding. We normalized the candidate translation scores by dividing their length and carried out a beam search with a size of six. Our implementation is based on `fairseq` (Ott et al., 2019). We used mixed-precision training (Micikevicius et al., 2018) to reduce memory consumption and training time. All experiments were run on eight NVIDIA Tesla V100 GPUs with 32-GB memory. Since we did not conduct a hyperparameter search, almost all of the settings were borrowed from (Morishita et al., 2020).

DocRepair requires the training of three sequence-to-sequence models: (1) an NMT model that translates language X to Y; (2) an NMT model that translates in reverse direction to make round-trip translation; and (3) a sequence-to-sequence model that converts 1-best translations to document-level translation. We used "Baseline 7 Enc-Layers" models for both (1) and (2), and newly trained the Transformer model for (3).

B Additional Experimental Results

Table 3 shows the number of parameters for each model, training speed, and BLEU scores on the dev set. The scores show the same tendency as the test set (Table 1).

The DocRepair method requires two translation models (English-to-Japanese and Single-to-Document), and thus the number of model parameters is larger than that for the other models. Although it also requires a Japanese-to-English translation model for creating round-trip translation data for training, these model parameters are not included in the table.

Since our MBE implementation was still in the experimental phase, the training speed was slower than that of the baselines, which were fully optimized by `fairseq` developers. We can further improve our implementation for faster computation, but we leave this for future work.

C Links to Data and Software

C.1 Data

JParaCrawl `https://www.kecl.ntt.co.jp/icl/lirg/jparacrawl/`

ASPEC `http://orchid.kuee.kyoto-u.ac.jp/ASPEC/`

newsdev2020 `http://www.statmt.org/wmt20/translation-task.html`

tst2012 `https://wit3.fbk.eu/`

C.2 Software

fairseq `https://github.com/pytorch/fairseq`

Model	**Parameters**	**wps**	**hours for training**	**BLEU (ASPEC dev)**
Baseline 6 Enc-Layers (Vaswani et al., 2017)	274M	187k	9.7	26.7
Baseline 7 Enc-Layers (Vaswani et al., 2017)	287M	167k	10.2	27.2 (+0.5)
2-to-1 (Tiedemann and Scherrer, 2017)	274M	125k	15.2	28.1 (+1.4)
DocRepair (Voita et al., 2019)	555M	236k	26.8	27.3 (+0.6)
MBE Enc	287M	93k	21.1	27.9 (+1.2)
MBE Enc w/o Gate	287M	82k	24.2	27.4 (+0.7)
MBE Dec	287M	93k	21.0	28.0 (+1.3)
MBE Enc/Dec	287M	92k	21.3	28.3 (+1.6)

Table 3: Number of parameters, training speed (words per sec, wps), required hours for training, and BLEU scores for the dev set.

sacreBLEU `https://github.com/mjpost/sacreBLEU`

sentencepiece `https://github.com/google/sentencepiece`

Deep Subjecthood: Higher-Order Grammatical Features in Multilingual BERT

Isabel Papadimitriou
Stanford University
isabelvp@stanford.edu

Ethan A. Chi
Stanford University
ethanchi@cs.stanford.edu

Richard Futrell
University of California, Irvine
rfutrell@uci.edu

Kyle Mahowald
University of California, Santa Barbara
mahowald@ucsb.edu

Abstract

We investigate how Multilingual BERT (mBERT) encodes grammar by examining how the high-order grammatical feature of *morphosyntactic alignment* (how different languages define what counts as a "subject") is manifested across the embedding spaces of different languages. To understand if and how morphosyntactic alignment affects contextual embedding spaces, we train classifiers to recover the subjecthood of mBERT embeddings in transitive sentences (which do not contain overt information about morphosyntactic alignment) and then evaluate them zero-shot on intransitive sentences (where subjecthood classification depends on alignment), within and across languages. We find that the resulting classifier distributions reflect the morphosyntactic alignment of their training languages. Our results demonstrate that *mBERT representations are influenced by high-level grammatical features that are not manifested in any one input sentence*, and that this is robust across languages. Further examining the characteristics that our classifiers rely on, we find that features such as passive voice, animacy and case strongly correlate with classification decisions, suggesting that mBERT does not encode subjecthood purely syntactically, but that subjecthood embedding is continuous and dependent on semantic and discourse factors, as is proposed in much of the functional linguistics literature. Together, these results provide insight into how grammatical features manifest in contextual embedding spaces, at a level of abstraction not covered by previous work.[1]

1 Introduction

Our goal is to understand whether, and how, large pretrained models encode abstract features of the grammars of languages. To do so, we analyze the notion of subjecthood in Multilingual BERT (mBERT) across diverse languages with different **morphosyntactic alignments**. Alignment (how each language defines what classifies as a "subject") is a feature of the grammar of a language, rather than of any single word or sentence, letting us analyze mBERT's representation of language-specific high-order grammatical properties.

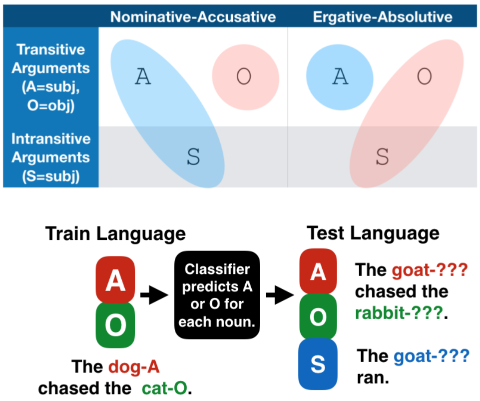


Figure 1: **Top**: Illustration of the difference between alignment systems. A (for agent) is notation used for the **transitive subject**, and O for the *transitive object*: "**The lawyer** chased *the dog*." S denotes the intransitive subject: "The lawyer laughed." The blue circle indicates which roles are marked as "subject" in each system.
Bottom: Illustration of the training and test process. We train a classifier to distinguish A from O arguments using the BERT contextual embeddings, and test the classifier's behavior on intransitive subjects (S). The resulting distribution reveals to what extent morphosyntactic alignment (above) affects model behavior.

Recent work has demonstrated that transformer models of language, such as BERT (Devlin et al., 2019), encode sentences in structurally meaningful ways (Manning et al., 2020; Rogers et al., 2020; Kovaleva et al., 2019; Linzen et al., 2016;

---

[1] We release the code to reproduce our experiments here https://github.com/toizzy/deep-subjecthood

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2522–2532
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Gulordava et al., 2018; Futrell et al., 2019; Wilcox et al., 2018). In Multilingual BERT, previous work has demonstrated surprising levels of multilingual and cross-lingual understanding (Pires et al., 2019; Wu and Dredze, 2019; Libovický et al., 2019; Chi et al., 2020), with some notable limitations (Mueller et al., 2020). However, these studies still leave an open question: are higher-order abstract grammatical features — features such as morphosyntactic alignment, which are not realized in any one sentence — accessible to deep neural models? And how are these allegedly discrete features represented in a continuous embedding space? Our goal is to answer these questions by examining grammatical subjecthood across typologically diverse languages. In doing so, we complicate the traditional notion of the grammatical subject as a discrete category and provide evidence for a richer, probabilistic characterization of subjecthood.

For 24 languages, we train small classifiers to distinguish the mBERT embeddings of nouns that are subjects of transitive sentences from nouns that are objects. We then test these classifiers on out-of-domain examples *within* and *across* languages. We go beyond standard probing methods (which rely on classifier accuracy to make claims about embedding spaces) by (a) testing the classifiers out-of-domain to gain insights about the shape and characteristics of the subjecthood classification boundary and (b) testing for awareness of morphosyntactic alignment, which is a feature of the grammar rather than of the classifier inputs.

Our main experiments are as follows. In Experiment 1, we test our subjecthood classifiers on out-of-domain *intransitive subjects* (subjects of verbs which do not have objects, like "The man slept") in their training language. Whereas in English and many other languages, we think of intransitive subjects as grammatical subjects, some languages have a different morphosyntactic alignment system and treat intransitive subjects more like objects (Dixon, 1979; Du Bois, 1987). We find evidence that a language's alignment is represented in mBERT's embeddings. In Experiment 2, we perform successful zero-shot cross-linguistic transfer of our subject classifiers, finding that higher-order features of the grammar of each language are represented in a way that is parallel across languages. In Experiment 3, we characterize the basis for these classifier decisions by studying how they vary as a function of linguistic features like animacy, grammatical case, and the passive construction.

Taken together, the results of these experiments suggest that mBERT represents subjecthood and objecthood robustly and probabilistically. Its representation is general enough such that it can transfer across languages, but also language-specific enough that it learns language-specific abstract grammatical features.

2 Background: Morphosyntactic alignment

In transitive sentences, languages need a way of distinguishing which noun is the transitive subject (called A, for agent) and which noun is the transitive object (O). In English, this distinction is marked by word order: "The dog$_\text{A}$ chased the lawyer$_\text{O}$" means something different than "the lawyer$_\text{A}$ chased the dog$_\text{O}$". In other languages, this distinction is marked by a morphological feature: case. Case markings, usually affixes, are attached to nouns to indicate their role in the sentence, and as such in these languages word order is often much freer than in English.

Apart from A and O, there is also a third grammatical role: intransitive subjects (S). In sentences like "The lawyer laughed", there is no ambiguity as to who is doing the action. As such, cased languages usually do not reserve a third case to mark S nouns, and use either the A case or the O case. Languages that mark S nouns in the same way as A nouns are said to follow a Nominative–Accusative case system, where the nominative case is for A and S, and the accusative case is for O. [2] Languages that mark S nouns like O nouns follow an Ergative–Absolutive system, where the ergative case is used to mark A nouns, and the absolutive case marks S and O nouns. For example, the Basque language follows this system. A visualization of the two case systems is shown in Figure 1.

The feature of whether a language follows a nominative-accusative or an ergative-absolutive system is called *morphosyntactic alignment*. Morphosyntactic alignment is a high-order grammatical feature of a language, which is not usually inferable from looking at just one sentence, but from

---

[2] English pronouns follow a Nominative–Accusative system. For example, the pronoun "she" is nominative and is used both for A and S (as in "she laughed"). The pronoun "her" is accusative and is used only for O.

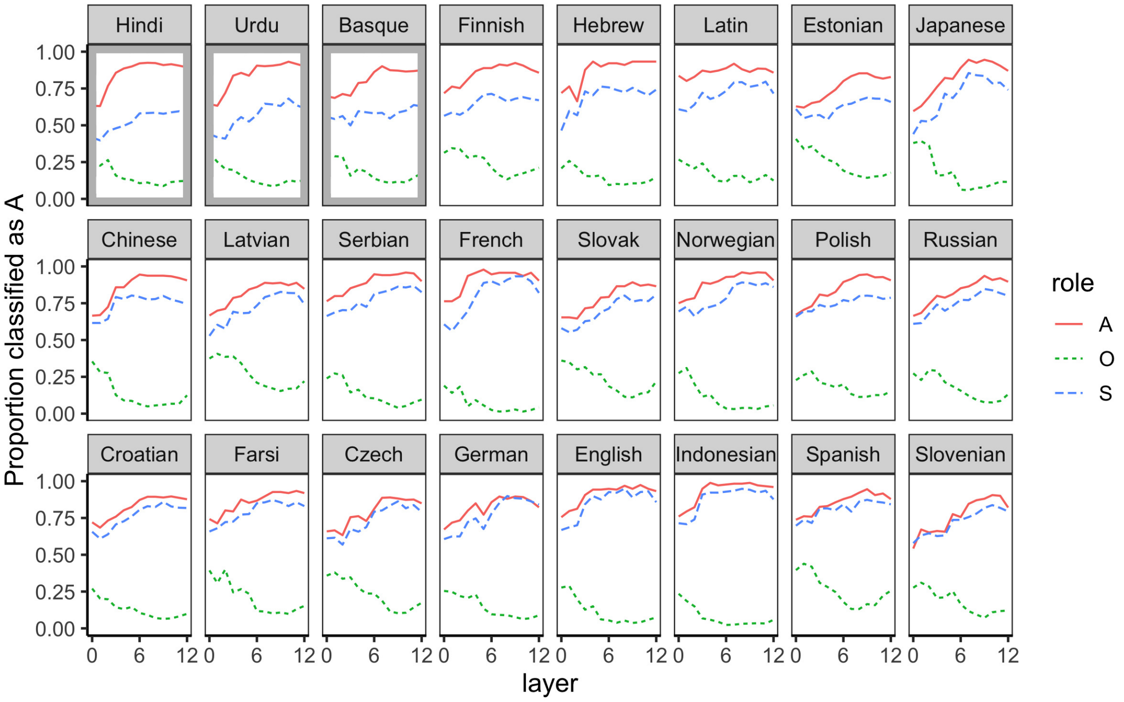


Figure 2: Results of Experiment 1: the behavior of subjecthood classifiers across mBERT layers (x-axis). For each layer, the proportion of the time that the classifier predicts arguments to be A, separated by grammatical role. In higher layers, A and O are reliably classified correctly, and S is mostly classified as A. When the source language is Basque (ergative) or Hindi or Urdu (split-ergative) S is less likely to pattern with A. The figure is ordered by how close the S line is to A, and ergative and split-ergative languages are highlighted with a gray box.

the system with which different sentences are encoded. As such, examining the way that individual contextual embeddings express morphosyntactic alignment gets to the question of how mBERT encodes abstract features of grammar. This is a question that is not answered by work that looks at the contextual encoding of the features that are realized in sentences, like part of speech or sentence structure.

3 Methods

Our primary method involves training classifiers to predict subjecthood from mBERT contextual embeddings, and examining the decisions of these classifiers within and across languages. We train a classifier to distinguish A from O in the mBERT embeddings of one language, and we examine its performance on S embeddings in its training language, and on A, S, and O mBERT embeddings in other languages.

Data To train a subjecthood classifier for one language, we use a balanced dataset of 1,012 transitive subject (A) mBERT embeddings, and 1,012 transitive object (O) mBERT embeddings. We test our classifiers on test datasets of A, S, and O embeddings. Our data points are extracted from the Universal Dependencies treebanks (Nivre et al., 2016): we use the dependency parse information to determine whether each noun is an A or an O, and if it is either we pass the whole sentence through mBERT and take the contextual embedding corresponding to the noun. We run experiments on 24 languages; specifically, all the languages that are both in the mBERT training set[3] and have Universal Dependencies treebanks with at least 1,012 A occurences and 1,012 O occurences.[4]

Labeling Since UD treebanks are not labeled for sentence role (A, S and O), we extract these labels using the dependency graph annotations. We only include nouns and proper nouns, leaving pronouns

---

[3] `https://github.com/google-research/bert/blob/master/multilingual.md`

[4] Our datasets for all languages are the same size. We have set them all to be the size of the largest balanced A-O dataset we can extract from the Basque UD corpus, since Basque is one of the only represented ergative languages and we wanted it to meet our cutoff.

for future work. [5] We label a noun token as:

- **O** if it has a verb as a head and its dependency arc is either `dobj` or `iobj`.
- **A** if it has a verb as a head and its dependency arc is `nsubj` and it has a sibling O.
- **S** if it has a verb as a head and its dependency arc is `nsubj` and it has no sibling O.

Finally, we exclude the subjects of passive constructions (where the object of an action is made the grammatical subject) to analyze separately, as including these examples would confound grammatical subjecthood with semantic agency. We also exclude the siblings of expletives (e.g., “There are many goats”), as these are grammatical objects which appear without subjects as the only argument of the verb, and we also exclude the children of auxiliaries (“The goat can swim”), looking only at the arguments of verbs.

Because we use embeddings and are limited by the Universal Dependencies annotation scheme, there are some cross-linguistic differences in how arguments are handled. For instance, our system is not able to handle null subjects or null objects, even though those are prominent parts of many languages.

Classifiers For each language, and for each mBERT layer ℓ, we train a classifier to classify mBERT contextual embeddings drawn from layer ℓ as A or O. The classifiers are all two-layer perceptrons with one hidden layer of size 64. We train each classifier for 20 epochs on a dataset of the layer-ℓ contextual embeddings of 1,012 A nouns and 1,012 O nouns. In total, we train 24 languages × 13 mBERT layers = 312 total classifiers.

4 Experiment 1: Subjecthood in mBERT

In our first experiment, we train a classifier to predict the grammatical role of a noun in context from its mBERT contextual embedding, and examine its behavior on intransitive subjects (S), which are out-of-domain.

This experimental setup lets us ask two questions about subjecthood encoding in mBERT. Firstly, do contextual word embeddings reliably encode subjecthood information? Secondly, how do our classifiers act when given S arguments (intransitive subjects), which crucially do not appear in the training data? If S arguments are mostly classified as A, that would suggest mBERT is learning a nominative-accusative system, where A and S pattern together. If S patterns with O, that would suggest it has an ergative-absolutive system. If S patterns differently in different languages, that would suggest that it learns a language-specific morphosyntactic system and expresses it in the encoding of nouns in transitive clauses (which are unaffected by alignment), so that the A-O classifiers can pick it up.

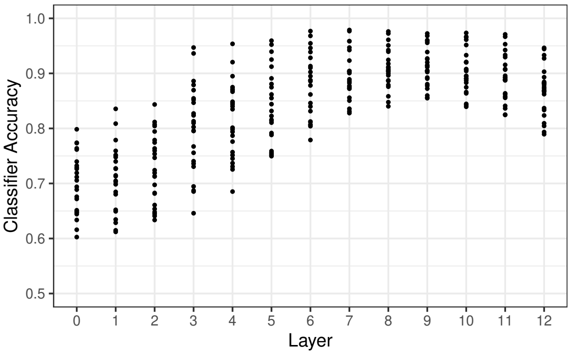


Figure 3: Accuracy of A-O classifiers for every language, by mBERT layer. For all languages, accuracy is highest in layers 7-10

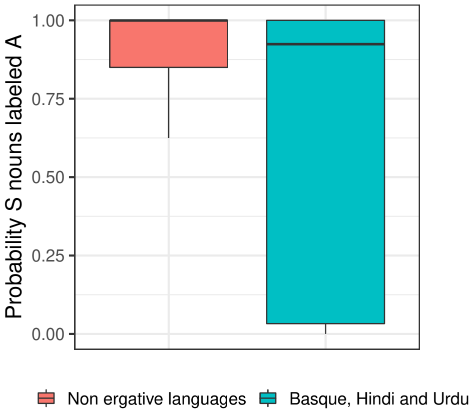


Figure 4: Distribution of layer 10 classifier probabilities for S nouns in the test set. When trained on non-ergative languages, the classifiers mostly predict S nouns to be A. When trained on ergative and split-ergative languages, the classifier predictions for S are much more spread out (towards being classified as O), suggesting that the ergative nature of the languages is expressed in the contextual embeddings of the A and O nouns, influencing the classifier.

4.1 Results

Our results show that the classifiers can reliably perform A-O classification of contextual embed-

---

[5] For an example of how pronouns complicate how subjecthood is defined, see Fox (1987).

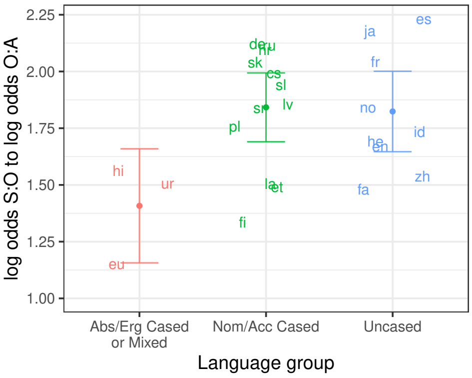


Figure 5: For layer 10, the log odds ratio of S:A relative to O:A, by source language. This is a measure of how close S is to A, relative to O. The ergative languages skew lower than the others, although some other languages (like Finnish and Estonian) also skew low.

dings with relatively high accuracy, especially in the higher layers of mBERT. As shown in Figure 3, performance peaks at around mBERT layers 7-10, where for the majority of languages classifier accuracy surpasses 90%. This is consistent with previous work showing that syntactic information is most well represented in BERT's later middle layers (Rogers et al., 2020; Hewitt and Manning, 2019; Jawahar et al., 2019; Liu et al., 2019). For the rest of this paper, we will focus mainly on the behavior of the classifiers in the high-performance higher layers to assess the properties in these highly contextual spaces that define subjecthood within and across languages.

Performance across layers on the test sets of all 24 languages is shown in Figure 2. When we break the classifiers' behavior down across roles, we see that S nouns mostly pattern with A, though they are consistently less likely to be classed as A than transitive A nouns.

The separation between the A and the S lines is not constant for all languages: it is the largest for Basque, which is an ergative language, and Hindi and Urdu, which have a split-ergative case system (De Hoop and Narasimhan, 2005). This difference is highlighted in Figure 4, where we show the classifiers' probabilities of classifying S nouns as A across the test sets of Basque, Hindi and Urdu versus the test sets for all other 21 languages. In Figure 5, we plot the log odds ratio of classifying S as A versus classifying S as O, and show that for ergative languages this is significantly lower. The fact that classifiers trained on ergative and split-ergative languages are more likely to classify S nouns as O indicates that the ergativity of the language is encoded in the A and O embeddings that the classifiers are trained on.

Note, however, that the A-O classifiers for the ergative languages do not deduce a fully ergative system for classifying S nouns, but a greater skew towards classifying S as O than nominative languages. This suggests that, even though properties of ergativity are encoded in mBERT space, the prominence of nominative training languages has influenced the contextual space to be biased towards encoding a nominative subjecthood system. The difficulty of training the classifier in Basque seems consistent with Ravfogel et al. (2019)'s finding that learning agreement is harder in Basque than in English.

In Experiment 2, we test the zero-shot performance of these A-O classifiers across languages, to ask: is there a parallel, interlingual notion of subjecthood in mBERT contextual space, and do language-specific morphosyntactic alignment biases transfer interlingually?

5 Experiment 2: Transferring across languages

We can learn only so much about mBERT's general subjecthood representations by training and testing in the same language, since many languages in our data set have case-marking and therefore have surface forms that reflect their grammatical roles. To test whether representations of subjecthood in mBERT are language-general, we can do a similar analysis to Experiment 1 but with zero-shot cross-lingual transfer.

That is, we train a classifier to distinguish A and O in Language X (just as in Experiment 1), but then we test in Language Y by seeing how the classifier classifies A, O, and S arguments in Language Y.

By training a classifier on one language and testing on others, we can ask: *is subjecthood encoded in parallel ways across languages in mBERT space?* If a classifier trained to distinguish A from O in a source language can then use the same parameters to successfully classify A from O in another language, this would indicate that the difference between A and O is encoded in similar ways in mBERT space for these two languages.

Secondly, we can examine the classification of S nouns (which are out of domain for the classifiers) in the zero-shot cross-lingual setting. By ob-

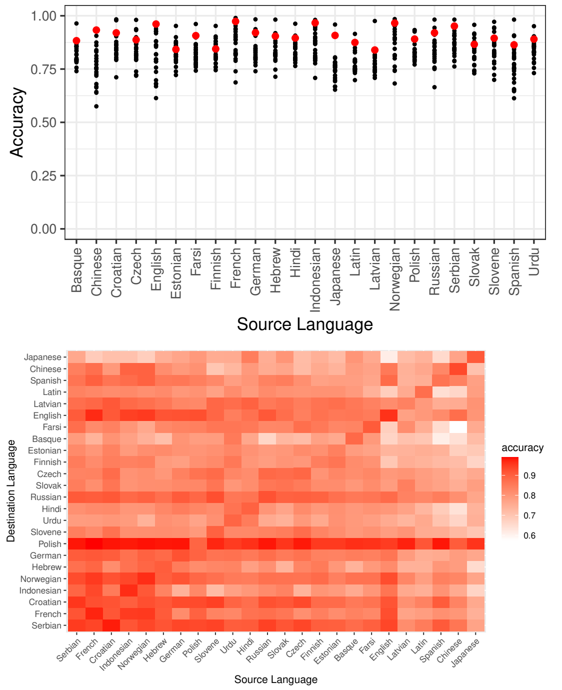


Figure 6: Experiment 2 Results: Cross-lingual transfer accuracies (accuracies shown are for BERT layer 10). **Top:** For each classifier trained to distinguish A and O nouns in a source language (labeled on the x-axis), we plot the accuracy that classifier achieves when tested zero-shot on all other languages. Zero-shot transfer is surprisingly successful across languages, indicating that subjecthood is encoded cross-lingually in mBERT. Each black point represents the accuracy of a classifier tested on a particular destination language, and the red points represent the within-language accuracy.
Bottom: Analytical performance of classifiers for every language pair. The x-axis sorted by average transfer accuracy, so that the source whose classifier performs the best on average is on the left. Despite the general English bias that mBERT often exhibits, in our experiments English is neither a standout source nor destination.

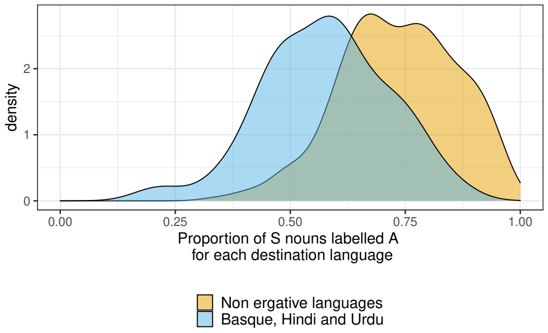


Figure 7: Classifiers trained on ergative languages are more likely to label S nouns in other languages as O. For BERT layer 8, the proportion of S nouns in each destination language test set labeled as A for the classifiers trained on (1) ergative and split-ergative languages (blue) or (2) the rest of the languages.

serving the test behavior of classifiers on S nouns in other languages, we can ask: *is morphosyntactic alignment expressed in cross-lingually generalizable and parallel ways in mBERT contextual embeddings?* If a classifier trained to distinguish Basque A from O is more likely to classify English S nouns as O, this means that information about morphosyntactic alignment is encoded specifically enough to represent each language's alignment, but in a space that generalizes across languages.

5.1 Results

Zero-shot transfer of subjecthood classification is effective across languages, as shown in Figure 6. The average accuracy across all source-destination pairs for a high-performing mBERT layer (layer 10) is 82.61%, and there are several pairs for which zero-shot transfer of the sentence role classifier yields accuracies above 90%. The consistent success of zero-shot transfer across different source and destination pairs indicates that mBERT has parallel, interlingual ways of encoding grammatical and semantic relations like subjecthood. We would expect there to be some extent of joint learning in mBERT: different languages wouldn't exist totally independently in the contextual embedding space, both due to mBERT's multilingual training texts and to successful regularization. It is nevertheless surprising that zero-shot transfer of subjecthood classification between languages is so successful out of the box, and that for all classifiers, within-language accuracy (the red dots in Figure 6) is not an outlier compared to transfer accuracies. Our results show not just that there is mutual entanglement between the contextual embedding spaces of many languages, but that syntactic and semantic information in these spaces is organized in largely parallel, transferable ways.

We can then look at how S is classified: does the subjecthood of S, and the degree of ergativity within each language that we saw expressed in Experiment 1 generalize across languages? Classifiers trained on ergative languages are significantly more likely to classify S nouns in other languages as O, as illustrated in Figure 7 (the source language's case system is a significant predictor of the probability of S being an agent, in a mixed effect regression with a random intercept for language $\beta = .11$, $t = 2.63$, $p < .05$). Our results show that the ergative nature of these languages is encoded in the contextual embeddings of transitive nouns (where ergativity is not realized), and that this encoding of ergativity transfers coherently across languages.

6 Experiment 3: Syntactic and semantic factors of continuous subjecthood

To explore the nature of mBERT's underlying representation of grammatical role, we ask which arguments are most likely to be classified as subjects or objects. This is of particular interest when the classifier gets it wrong: what kinds of subjects get erroneously classified as objects?

The functional linguistics literature offers insight into these questions. It has been frequently claimed that grammatical subjecthood is actually a multi-factor, probabilistic concept (Keenan and Comrie, 1977; Comrie, 1981; Croft, 2001; Hopper and Thompson, 1980) that cannot always be pinned down as a discrete category. Some subjects are more subject-y than others. Comrie (1988) argues that a subject can be thought of as the intersection of that which is high in **agency** (subjects *do* things) and **topicality** (subjects are the topics of sentences). Thus, in English, a prototypical subject is something like “He kicked the ball.” since in such a sentence, the pronoun “he” is a clear agent and the topic of the sentence. But, in a sentence like “The lake, which Jack Frost visited, froze,” the subject is still “lake.” But it is less subject-y: it is not the clear topic of the sentence and it is not an agent.

A probabilistic notion of grammatical role lends itself naturally to the continuous embedding spaces of computational models. So, in a series of experiments, we explored what factors in mBERT contextual embedding space predict subjecthood.

In these experiments, we examine how the decisions and probabilities of the A-O classifiers from Experiment 2 relate to other linguistic features known to contribute to the degree of subjecthood. In particular, we look at whether nouns appear in *passive constructions*, as well as the *animacy* and *case* of nouns. In seeing how passives, animacy, and case interact with our subjecthood classifiers, we can assess if mBERT's representation of subjecthood in continuous space is consistent with functional analyses, and better understand the continuous space in which mBERT encodes syntactic and semantic relations.

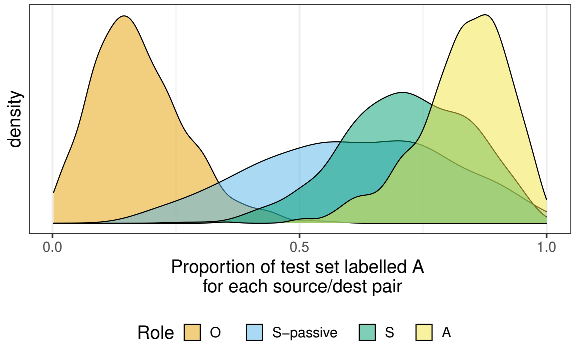


Figure 8: Passive subjects are hard to classify. The distribution of average classifier probabilities in layer 10 for all source-destination language pairs, separated by role. While the layer 10 classifier separates A and S from O, passive subjects remain largely ambiguous in their classification. These plots indicate that, in mBERT space, the grammatical subjects of passive constructions are less subject-y.

We choose these three factors as they are well-studied in the functional literature, as well as readily available to extract from UD corpora. Passive subjects are marked with a separate dependency arc label, the animacy of nouns is annotated directly in some UD treebanks, and in case-marked languages, nouns are annotated with their case. Future work on a more complete examination of the functional nature of contextual embeddings would include other factors not readily available in UD, like the discourse and information structure (topicality) of nouns in context.

6.1 Results

The first area that we look at are passive constructions. In passive constructions such as “The lawyer was chased by a cat”, the grammatical subject is not the main actor or agent in the sentence. As such, while a purely syntactic analysis of subjecthood would classify passive subjects (S-passive) as subjects, an understanding of subjecthood as continuous and reliant on semantics would be more prone to classify passive subjects as objects. As shown in Figure 8, subjecthood classifiers across languages are ambivalent about how they classify passive subjects, even in layers where they have the acuity to successfully separate A and S from O. This indicates that the classifiers do not learn a purely syntactic separation of A and O: the subjecthood encoding that they learn from mBERT space is largely dependent on semantic information.

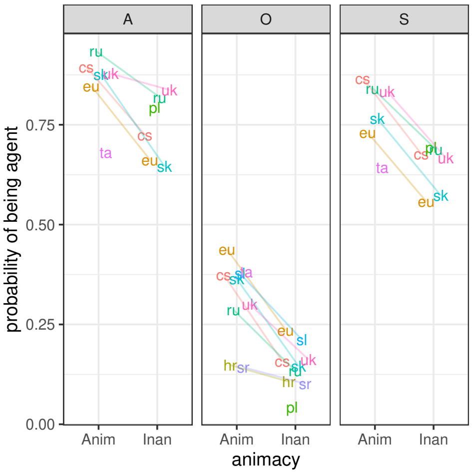


Figure 9: The influence of animacy on classification (within and across languages). For a high-performing layer (Layer 10), the average probability of classifiers in all languages classifying nouns in languages with animacy distinctions as A. For all three roles, animates are more likely to be classified as agents. The labels are two-letter codes for the languages.

We also find that *animacy is a strong predictor of subjecthood*. Our results presented in Figure 9 demonstrate that when we control by role, animacy is a significant factor in determining the probability of being classified as A. Classifiers in all languages, when zero-shot evaluated on a corpus marked for animacy, are more likely to classify animate nouns as A than inanimate nouns. For Layer 10, a mixed effect regression predicting each destination language's probability of assigning an argument to being an agent shows that both role and animacy are significant predictors (with a main effect of animacy corresponding to a 16% increase in the probability of being an agent, $p < .01$). These results indicate that, in learning to separate A from O, the classifiers did not learn a purely syntactic separation of the space (though it is possible to distinguish A and O using only strictly structural syntactic features). Instead, we see that subjecthood information is entangled with semantic notions such as animacy, giving credence to the hypothesis that subjecthood BERT space is encoded in a way concordant with the multi-factor manner proposed by Croft, Comrie, and others.

Lastly, we find that classifier probabilities also vary with case, even when we control for sentence role. As demonstrated in Figure 10, across grammatical roles, classifiers are significantly more likely to classify nouns as A if they are in more agentive cases (nominative and ergative). In a mixed effect regression predicting Layer 10 probability of being an agent based on role and whether the case is agentive (nominative/ergative), there was a 15% increase associated with being nominative/ergative across categories ($t = 2.74, p < .05$).

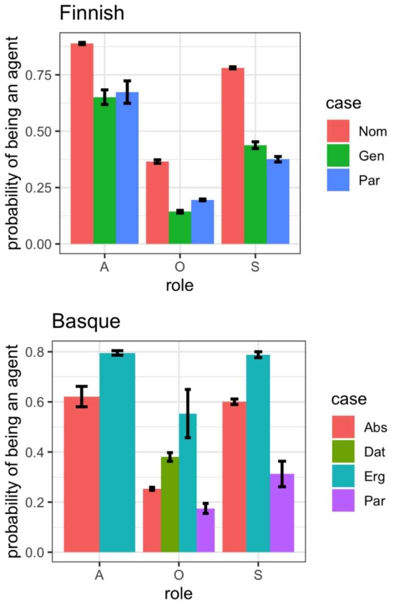


Figure 10: Average probability of being an agent, in layer 10, with 95% confidence intervals, for Finnish and Basque broken up by case.

7 Discussion

Our experimental results constitute a way to begin understanding how general knowledge of grammar is manifested in contextual embedding spaces, and how discrete categories like subjecthood are reconciled in continuous embedding spaces. While most previous work analyzing large contextual models focuses on extracting their analysis of features or structures present in specific inputs, we focus on morphosyntactic alignment, a feature of grammars that is not explicitly realized in any one sentence.

We find that, when tested out of domain, classifiers trained to predict transitive subjecthood in mBERT contextual space robustly demonstrate decisions which reflect (a) the morphosyntactic alignment of their training language and (b) continuous encoding of subjecthood influenced by semantic properties.

There has been much recent work pointing out the limitations of the probing methodology for analyzing embedding spaces (Voita and Titov, 2020;

Pimentel et al., 2020; Hewitt and Liang, 2019), a methodology that is very similar to ours. The main limitation pointed out in this literature is that the power of classifiers is a confounding variable: we can't know if a classifier's encoding of a feature is due to the feature being encoded in BERT space, or to the classifier figuring out the feature from surface encoding.

In this paper, we address these issues by proposing two ways to use classifiers to analyze embedding spaces that go beyond probing, and avoid the limitations of arguments based only around the accuracy of probes. Firstly, our results rely on testing the classifiers on *out-of-domain zero-shot transfer*: both to S arguments and to different languages. As such, we focus on linguistically defining the type of classification boundary which our classifiers learn from mBERT space, rather than their accuracy, and in using transfer we avoid many of the limitations of probing, as argued in Papadimitriou and Jurafsky (2020). Secondly, we examine a feature (morphosyntactic alignment) which is *not inferable from the classifiers' training data*, which consists only of transitive sentences. We are asking if mBERT contextual space is organized in a way that encodes the effects of morphosyntactic alignment for tokens that do not themselves express alignment. Especially in the cross-lingual case, a classifier would not be able to spuriously deduce this from the surface form, whatever its power.

A limitation of our experimental setup is that both our Universal Dependencies training data and the set of mBERT training languages are heavily weighted towards nominative-accusative languages. As such, we see a clear nominative-accusative bias in mBERT, and our results are somewhat noisy as we only have one ergative-absolutive language and two semi-ergative languages

Future work should examine the effects of balanced joint training between nominative-accusative and ergative-absolutive languages on the contextual embedding of subjecthood. And we hope that future work will continue to ask not just *if* deep neural models of language represent discrete linguistic features, but how they represent them probabilistically.

Acknowledgments

We thank Dan Jurafsky, Jesse Mu, and Ben Newman for helpful comments. This work was supported by NSF grant #1947307 and a gift from the NVIDIA corporation to R.F., and an NSF Graduate Research Fellowship to I.P.

References

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Bernard Comrie. 1981. *Language Universals and Linguistic Typology*, 1st edition. University of Chicago Press, Chicago.

Bernard Comrie. 1988. Linguistic typology. *Annual Review of Anthropology*, 17:145–159.

William Croft. 2001. *Radical construction grammar: Syntactic theory in typological perspective*. Oxford University Press.

Helen De Hoop and Bhuvana Narasimhan. 2005. Differential case-marking in hindi. In *Competition and Variation in Natural Languages*, pages 321–345. Elsevier.

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

Robert MW Dixon. 1979. Ergativity. *Language*, pages 59–138.

John W Du Bois. 1987. The discourse basis of ergativity. *Language*, pages 805–855.

Barbara A Fox. 1987. The noun phrase accessibility hierarchy reinterpreted: Subject primacy or the absolutive hypothesis? *Language*, pages 856–870.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Paul J Hopper and Sandra A Thompson. 1980. Transitivity in grammar and discourse. *language*, pages 251–299.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Edward L. Keenan and Bernard Comrie. 1977. Noun phrase accessibility and universal grammar. *Linguistic Inquiry*, 8(1):63–99.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Jindřich Libovickỳ, Rudolf Rosa, and Alexander Fraser. 2019. How language-neutral is multilingual bert? *arXiv preprint arXiv:1911.03310*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Christopher D Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*.

Aaron Mueller, Garrett Nicolai, Panayiota Petrou-Zeniou, Natalia Talmina, and Tal Linzen. 2020. Cross-linguistic syntactic evaluation of word prediction models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5523–5539.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning music helps you read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the inductive biases of RNNs with synthetic variations of natural languages. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3532–3542, Minneapolis, Minnesota. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do RNN language models learn about filler–gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels, Belgium. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844.

Streaming Models for Joint Speech Recognition and Translation

Orion Weller[1]*, **Matthias Sperber**[2], **Christian Gollan**[2], **Joris Kluivers**[2]
[1]Brigham Young University
[2]Apple
oweller@byu.edu,{m_sperber,cgollan,jkluivers}@apple.com

Abstract

Using end-to-end models for speech translation (ST) has increasingly been the focus of the ST community. These models condense the previously cascaded systems by directly converting sound waves into translated text. However, cascaded models have the advantage of including automatic speech recognition output, useful for a variety of practical ST systems that often display transcripts to the user alongside the translations. To bridge this gap, recent work has shown initial progress into the feasibility for end-to-end models to produce both of these outputs. However, all previous work has only looked at this problem from the consecutive perspective, leaving uncertainty on whether these approaches are effective in the more challenging streaming setting. We develop an end-to-end streaming ST model based on a re-translation approach and compare against standard cascading approaches. We also introduce a novel inference method for the joint case, interleaving both transcript and translation in generation and removing the need to use separate decoders. Our evaluation across a range of metrics capturing accuracy, latency, and consistency shows that our end-to-end models are statistically similar to cascading models, while having half the number of parameters. We also find that both systems provide strong translation quality at low latency, keeping 99% of consecutive quality at a lag of just under a second.

1 Introduction

Speech translation (ST) is the process of translating acoustic sound waves into text in a different language than was originally spoken in.

This paper focuses on ST in a particular setting, as described by two characteristics: (1) We desire models that translate in a streaming fashion, where users desire the translation before the speaker has finished. This setting poses additional difficulties compared to consecutive translation, forcing systems to translate without knowing what the speaker will say in the future. (2) Furthermore, the speaker may want to verify that their speech is being processed correctly, intuitively seeing a streaming transcript while they speak (Fügen, 2008; Hsiao et al., 2006). For this reason, we consider models that produce both transcripts and translation jointly.[1]

Previous approaches to streaming ST have typically utilized a cascaded system that pipelines the output of an automatic speech recognition (ASR) system through a machine translation (MT) model for the final result. These systems have been the preeminent strategy, taking the top place in recent streaming ST competitions (Pham et al., 2019; Jan et al., 2019; Elbayad et al., 2020; Ansari et al., 2020). Despite the strong performance of these cascaded systems, there are also some problems: error propagation from ASR output to MT input (Ruiz and Federico, 2014); ASR/MT training data mismatch and loss of access to prosodic/paralinguistic speech information at the translation stage (Sperber and Paulik, 2020); and potentially sub-optimal latencies in the streaming context. End-to-end (E2E) models for ST have been proposed to remedy these problems, leveraging the simplicity of a single model to sidestep these issues. E2E models are also appealing from computational and engineering standpoints, reducing model complexity and decreasing parameter count.

Although initial research has explored E2E models for joint speech recognition and translation, no previous works have examined them in the streaming case, a crucial step in using them for many real-world applications. To understand this area more fully, we develop an E2E model to compare

---

*Work done during an internship with Apple

[1] This corresponds to the *mandatory transcript* case in the proposed categorization by Sperber and Paulik (2020).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2533–2539
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

with its cascading counterpart in this simultaneous joint task. We build off the models proposed by Sperber et al. (2020) in the consecutive case, extending them for use in the streaming setting. We also use the re-translation technique introduced by Niehues et al. (2018) to maintain simplicity while streaming. To reduce model size, we introduce a new method for E2E inference, producing both transcript and translation in an interleaved fashion with one decoder.

As this task requires a multi-faceted evaluation along several axes, we provide a suite of evaluations to highlight the differences of these major design decisions. This suite includes assessing translation quality, transcription quality, lag of the streaming process, output flicker, and consistency between the transcription and translation. We find that our E2E model performs similarly to the cascaded model, indicating that E2E networks are a feasible and promising direction for streaming ST.

2 Proposed Method

Network Architecture In the ST survey provided by Sperber et al. (2020), they introduce several E2E models that could be used for the joint setting. As our work focuses on providing a simple but effective approach to streaming ST, we focus on the *CONCAT* model, which generates both the transcript and translation in a concatenated fashion. We compare this E2E model against the standard cascading approach, following the architecture and hyperparameter choices used in Sperber et al. (2020). All audio input models use the same multi-layer bidirectional LSTM architecture, stacking and downsampling the audio by a factor of three before processing. We note that although bidirectional encoders are unusual with standard ASR architectures, re-translation makes them possible. The cascaded model's textual encoder follows the architecture described in Vaswani et al. (2017) but replaces self-attention blocks with LSTMs. Decoder networks are similar, but use unidirectional LSTMs. More implementation details can be found in Appendix A.

In order to reduce model size and inference time for E2E networks, we introduce a novel method for interleaving both transcript and translation in generation, removing the need to use separate decoders. This method extends the *CONCAT* model proposed by Sperber et al. (2020) to jointly decode according to the ratio given by the parameter γ (Figure 1).

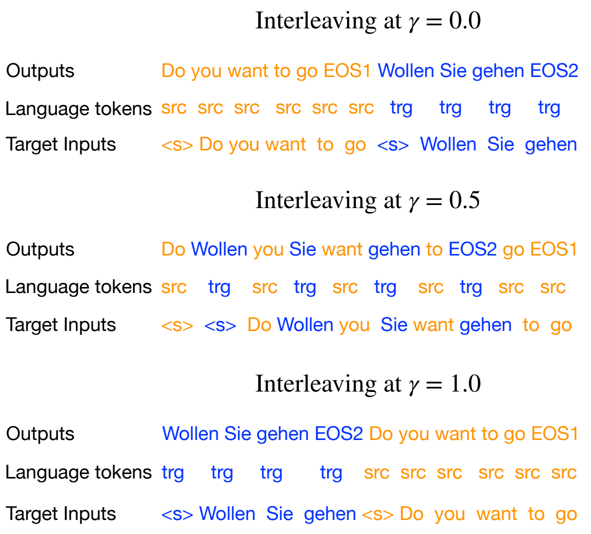


Figure 1: Example token representations (En→De) for three different interleaving parameters (Section 2). Language tokens indicate whether the data corresponds to the source transcript or the target translation and are used with a learned embedding that is summed with the word embeddings, as described in Sperber et al. (2020).

When $\gamma = 0.0$, we generate the transcript tokens until completion, followed by the translation tokens (vice versa for $\gamma = 1.0$). At $\gamma = 0.0$, our model is equivalent to the previously proposed model. Defining $\mathrm{count_i}$ as the count of i tokens previously generated, transcription tokens as *st* and translation tokens as *tt*, we generate the next token as a transcription token if:

$$(1.0 - \gamma) * (1 + \mathrm{count_{tt}}) > \gamma * (1 + \mathrm{count_{st}})$$

This approach enables us to produce tokens in an interleaving fashion, given the hyperparameter γ.

Re-translation We use the re-translation method (Niehues et al., 2018; Arivazhagan et al., 2020a,b) as it provides a simple way to handle the streaming case. This method works by simply re-translating the utterance as new data arrives, updating its former prediction. As we are generating both transcript and translation, this avoids the challenging issue of combining the requirements for both components: streaming speech models need to manage the audio signal variability across time while streaming translation models need to overcome issues with reordering and lack of future context.

Alternative strategies to the re-translation approach include the chunk-based strategy explored by Liu et al. (2020), which commits to all previous output chunks and Ren et al. (2020) who utilize an additional segmenter model trained via CTC

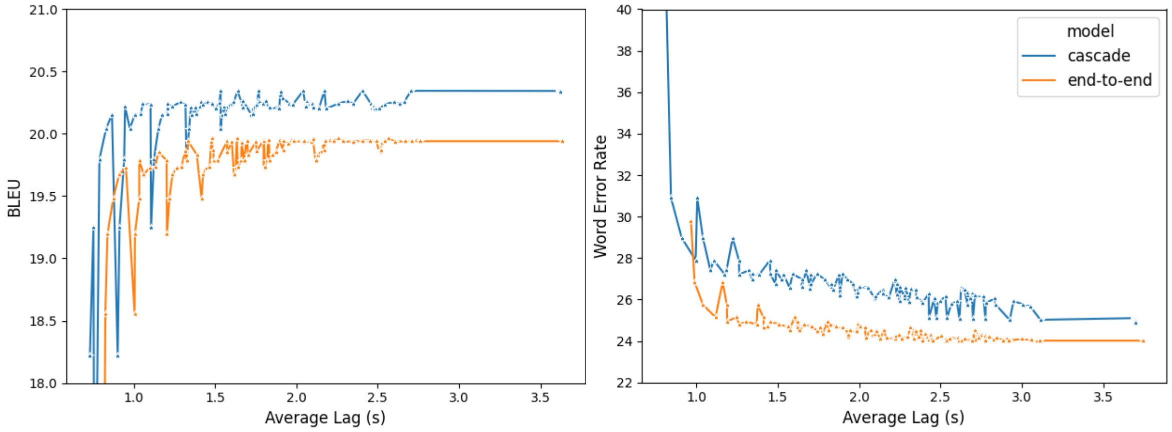


Figure 2: Left: average lag in seconds vs BLEU score. Right: average lag in seconds vs WER score. All points are the mean of each configuration's score across the eight target languages. Configurations are the cross product of the values for K and F, see Section 2: Inference. Note that points near 1.0 AL have appx. 99% of the unconstrained BLEU score. Results for the E2E model use $\gamma = 0.5$.

.

Metric	**Params**	**Model**	**De**	**Es**	**Fr**	**It**	**Nl**	**Pt**	**Ro**	**Ru**	**Mean**
BLEU ↑	217M	Cascade	**18.8**	**22.7**	**27.0**	**18.9**	**22.5**	**21.9**	**17.9**	**13.0**	**20.3**
	107M	E2E γ=0.0	18.1	**23.1**	**27.0**	**18.7**	**22.3**	**22.2**	**17.6**	12.2	20.2
	107M	E2E γ=0.3	17.7	22.6	26.3	18.0	21.5	21.5	17.0	12.1	19.6
	107M	E2E γ=0.5	18.2	**22.8**	**27.0**	**18.6**	21.9	21.9	17.1	12.0	19.9
	107M	E2E γ=1.0	18.2	**22.8**	**27.1**	**18.9**	**22.2**	**22.3**	**17.6**	**12.7**	**20.2**
WER ↓	217M	Cascade	25.9	24.0	23.1	25.6	28.5	26.4	24.4	**23.1**	25.1
	107M	E2E γ=0.0	24.2	23.5	23.3	**23.0**	23.4	25.3	24.1	23.6	23.8
	107M	E2E γ=0.3	24.1	23.6	22.9	23.8	23.4	25.7	24.1	24.1	24.0
	107M	E2E γ=0.5	24.5	23.9	22.9	23.8	23.4	25.7	24.3	23.6	24.0
	107M	E2E γ=1.0	**23.6**	**22.9**	**22.3**	**23.0**	**22.4**	**24.7**	**23.4**	**22.7**	**23.1**

Table 1: BLEU and WER scores for models trained on different target languages. Bold scores indicate results that are statistically similar to the best score using a bootstrap permutation test with $\alpha = 0.05$.

(Graves et al., 2006) to create segments that are translated via wait-k (Ma et al., 2019). Although these approaches show effective results, they add additional complexity without addressing issues particular to streaming transcription.

Inference In order to generate quality-latency curves, we use several techniques to reduce latency and flicker at the cost of quality. The first is the mask-*k* method proposed by Arivazhagan et al. (2020b), masking the last K output tokens. The second method is a form of constrained decoding: we define a hyperparameter F that sets the number of free tokens allowed to change in the next re-translation. Thus, we constrain future output to match the first $len(tokens) - F$ tokens of the current output. All models use values $\{0, 1, 2, 3, 4, 5, 7, 10, 100\}$ for K and $\{0, 1, 2, 3, 4, 5, 7, 10, 15, 20, 25, 100\}$ for F. For interleaving models, we set K and F on both transcript and translation tokens.

3 Experimental Settings

Data We use the MuST-C corpus (di Gangi et al., 2019) since it is the largest publicly available ST corpus, consisting of TED talks with their English transcripts and translations into eight other language pairs. The dataset consists of at least 385 hours of audio for each target language.

We utilize the log Mel filterbank speech features provided with the corpus as input for the ASR and E2E models. To prepare the textual data, we remove non-speech artifacts (e.g. "(laughter)" and speaker identification) and perform subword tokenization using SentencePiece (Kudo and Richard-

Model	En→De Incr.	En→De Full	En→Es Incr.	En→Es Full	Mean Incr.	Mean Full
Cascade	**13.8**	**13.2**	**12.2**	**11.6**	**14.1**	**13.4**
E2E γ=0.0	17.6	16.7	14.9	13.8	17.0	16.0
E2E γ=0.3	17.2	16.6	14.3	13.7	16.6	15.8
E2E γ=0.5	17.8	16.5	14.8	13.3	17.3	15.7
E2E γ=1.0	17.3	16.8	14.9	13.7	16.9	15.8

Table 2: Consistency scores for En→De, En→Es, and average results over all languages; lower is better (see Sperber et al. (2020)). *Incr.* stands for the incremental consistency score, or the average consistency throughout retranslation. Bold scores indicate results that are statistically similar to the best score using a bootstrap permutation test with $\alpha = 0.05$.

son, 2018) on the unigram setting. Following previous work for E2E ST models, we use a relatively small vocabulary and share transcription and translation vocabularies. We use MuST-C dev for validation and report results on tst-COMMON, utilizing the segments provided (Appendix D).

Prefix Sampling We implement techniques developed by Niehues et al. (2018); Arivazhagan et al. (2020b) for improving streaming ST, sampling a random proportion of each training instance as additional data to teach our models to work with partial input. See Appendix C for implementation details.

Metrics We evaluate these models on a comprehensive suite of metrics: sacrebleu (*BLEU*, Post (2019)) for translation quality, word error rate (*WER*, (Fiscus, 1997)) for transcription quality, average lag (*AL*, Ma et al. (2019)) for the lag between model input and output, and normalized erasure (*NE*, Arivazhagan et al. (2020a)) for output flicker. Measuring consistency is a nascent area of research; we use the robust and simple lexical consistency metric defined by Sperber et al. (2020), which uses word-level translation probabilities. To show how consistent these results are while streaming, we compute an incremental consistency score, averaging the consistency of each re-translation.

4 Results

Results for the quality-latency curves created by the use of constrained decoding and mask-*k* (Section 3) are shown in Figure 2. Unconstrained settings are used for all results in table form. For convenience, bold scores indicate the highest performing models in each metric according to a bootstrap permutation test.

Translation Quality We see in Table 1 that the cascaded model slightly outperforms some E2E models, while achieving statistically similar performance to the $\gamma = 1.0$ model. We note however, that the cascaded model has nearly twice as many parameters as the E2E models (217M vs 107M). When we examine these models under a variety of different inference conditions (using constrained decoding and mask-*k* as in Arivazhagan et al. (2020a)), we further see this trend illustrated through the quality vs latency trade-off (left of Figure 2), with both models retaining 99% of their BLEU at less than 1.0 AL.

Transcription Quality Conversely, Table 1 and the right of Figure 2 show that the $\gamma = 1.0$ E2E model performs similarly or slightly better than the cascaded model across all inference parameters and all target languages. With an AL of 1.5, the E2E model loses only 3% of its performance.

Consistency The E2E models perform worse than the cascaded on consistency, with the best models being approximately 18% less consistent (Table 2). The cascaded model also maintains better scores through each re-translation (*Incr.*).[2]

Flicker We note that the flicker scores for cascade and E2E models are similar, with both having normalized erasure scores of less than 1 and the majority of inference settings having less than the "few-revision" threshold of 0.2 (proposed by Arivazhagan et al. (2020a)). More NE details are found in Appendix B.

Interleaving Rate Table 1 also shows us the overall results for different interleaving rates. We see that interleaving at a rate of 1.0 has the best

---

[2] Initial experiments indicate that the triangle E2E architecture (Sperber et al., 2020) model may perform better on consistency in our streaming setting, but due to time constraints we were not able to explore this further. Future work exploring alternative architectures or decoding techniques (Le et al., 2020) may provide fruitful avenues of research.

quality scores (0.7 less WER than the next best rate, the base $\gamma = 0.0$ model) but the worst consistency (Table 2). Conversely, $\gamma = 0.3$ has the worst quality scores but the best consistency.

5 Conclusion

We focus on the task of streaming speech translation, producing both a target translation and a source transcript from an audio source. We develop an end-to-end model to avoid problems that arise from the use of cascaded models for streaming ST. We further introduce a new method for joint inference for end-to-end models, generating both translation and transcription tokens concurrently. We show that our novel end-to-end model, with only half the number of parameters, is comparable to standard cascaded models across a variety of evaluation categories: transcript and translation quality, lag of streaming, consistency between transcript and translation, and re-translation flicker. We hope that this will spur increased interest in using end-to-end models for practical applications of streaming speech translation.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondřej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander Waibel, and Changhan Wang. 2020. FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 1–34, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020a. Re-translation versus streaming for simultaneous translation. *arXiv preprint arXiv:2004.03643*.

Naveen Arivazhagan, Colin Cherry, Isabelle Te, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020b. Re-translation strategies for long form, simultaneous, spoken language translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7919–7923. IEEE.

Maha Elbayad, Ha Nguyen, Fethi Bougares, Natalia Tomashenko, Antoine Caubrière, Benjamin Lecouteux, Yannick Estève, and Laurent Besacier. 2020. On-trac consortium for end-to-end and simultaneous speech translation challenge tasks at iwslt 2020. *arXiv preprint arXiv:2005.11861*.

Jonathan G Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354. IEEE.

Christian Fügen. 2008. *A System for Simultaneous Translation of Lectures and Speeches*. Ph.D. thesis, University of Karlsruhe.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning (ICML)*.

Antonino Mattia di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C : a Multilingual Speech Translation Corpus. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, Minneapolis, USA.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Roger Hsiao, Ashish Venugopal, Thilo Köhler, Ting Zhang, Paisarn Charoenpornsawat, Andreas Zollmann, Stephan Vogel, Alan W. Black, Tanja Schultz, and Alex Waibel. 2006. Optimizing components for handheld two-way speech translation for an English-Iraqi Arabic system. In *Annual Conference of the International Speech Communication Association (InterSpeech)*, pages 765–768, Pittsburgh, USA.

Niehues Jan, Roldano Cattoni, Stuker Sebastian, Matteo Negri, Marco Turchi, Salesky Elizabeth, Sanabria Ramon, Barrault Loic, Specia Lucia, and Marcello Federico. 2019. The iwslt 2019 evaluation campaign. In *16th International Workshop on Spoken Language Translation 2019*.

Diederik P. Kingma and Jimmy L. Ba. 2014. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, Banff, Canada.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *Empirical Methods in Natural Language Processing (EMNLP)*, pages 66–71.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2020. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In

Proceedings of the 28th International Conference on Computational Linguistics, pages 3520–3533, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Danni Liu, Gerasimos Spanakis, and Jan Niehues. 2020. Low-latency sequence-to-sequence speech recognition and translation by partial hypothesis selection. *arXiv preprint arXiv:2005.11185*.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, et al. 2019. Stacl: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036.

Graham Neubig, Matthias Sperber, Xinyi Wang, Matthieu Felix, Austin Matthews, Sarguna Padmanabhan, Ye Qi, Devendra Singh Sachan, Philip Arthur, Pierre Godard, John Hewitt, Rachid Riad, and Liming Wang. 2018. XNMT: The eXtensible Neural Machine Translation Toolkit. In *Conference of the Association for Machine Translation in the Americas (AMTA) Open Source Software Showcase*, Boston, USA.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. *arXiv preprint arXiv:1808.00491*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeuIPS)*, Vancouver, Canada.

Ngoc-Quan Pham, Thai-Son Nguyen, Thanh-Le Ha, Juan Hussain, Felix Schneider, Jan Niehues, Sebastian Stüker, and Alexander Waibel. 2019. The iwslt 2019 kit speech translation system. In *Proceedings of the 16th International Workshop on Spoken Language Translation*.

Matt Post. 2019. A Call for Clarity in Reporting BLEU Scores. In *Conference on Machine Translation (WMT)*, pages 186–191, Brussels, Belgium.

Yi Ren, Jinglin Liu, Xu Tan, Chen Zhang, QIN Tao, Zhou Zhao, and Tie-Yan Liu. 2020. Simulspeech: End-to-end simultaneous speech to text translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3787–3796.

Nicholas Ruiz and Marcello Federico. 2014. Assessing the impact of speech recognition errors on machine translation quality. In *11th Conference of the Association for Machine Translation in the Americas (AMTA), Vancouver, BC, Canada*.

Matthias Sperber and Matthias Paulik. 2020. Speech Translation and the End-to-End Promise: Taking Stock of Where We Are. In *Association for Computational Linguistic (ACL)*, Seattle, USA.

Matthias Sperber, Hendra Setiawan, Christian Gollan, Udhyakumar Nallasamy, and Matthias Paulik. 2020. Consistent Transcription and Translation of Speech. *Transactions of the Association for Computational Linguistics (TACL)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Neural Information Processing Systems Conference (NIPS)*, pages 5998–6008, Long Beach, USA.

A Model Details

In this section we will describe implementation details of the model architectures (shown in Figure 3) and training processes.

Model Architectures Unless otherwise noted, the same hyperparameters are used for all models. Weights for the speech encoder are initialized based on a pre-trained attentional ASR task that is identical to the ASR part of the direct multitask model. Other weights are initialized according to Glorot and Bengio (2010). The speech encoder is a 5-layer bidirectional LSTM with 700 dimensions per direction. Attentional decoders consist of 2 Transformer blocks (Vaswani et al., 2017) but use 1024-dimensional unidirectional LSTMs instead of self-attention, except for the *CONCAT* model, which uses 3 layers.

For the cascade's MT model, encoder/decoder both contain 6 layers with 1024-dimensional LSTMs. Subword embeddings are size 1024. We regularize using LSTM dropout with $p = 0.3$, decoder input word-type dropout (Gal and Ghahramani, 2016), and attention dropout, both $p = 0.1$. We apply label smoothing with strength $\epsilon = 0.1$.

Training We optimize using Adam (Kingma and Ba, 2014) with $\alpha = 0.0005$, $\beta_1 = 0.9$, $\beta_2 = 0.98$, 4000 warm-up steps, and learning rate decay by using the inverse square root of the iteration of each instance. We set the batch size dynamically based on the sentence length, such that the average batch size is 128 utterances. The training is stopped when the validation score has not improved over 10 epochs, where the validation score is corpus-level translation BLEU score (for the E2E and MT models) and corpus-level WER for the cascade's ASR model.

For decoding and generating n-best lists, we use beam size 5 and polynomial length normalization with exponent 1.5. Our implementation is based on PyTorch (Paszke et al., 2019) and XNMT (Neubig et al., 2018), and all models are trained in single-GPU environments, employing Tesla V100 GPUs with 32 GB memory. Most E2E and ASR models converged after approximately 30 epochs or 5 days of training. MT models converged after approximately 50 epochs or 2 days of training.

B Normalized Erasure (Output Flicker)

We see similar curves for both the cascaded model and the E2E model when comparing normalized erasure in Figure 4. We see that most settings have an NE score of less than 0.2, while virtually all settings are less than 1. We note that a proportion of 0.2 for NE means that, on average, 1/5 of the tokens change once before they settle to their final state.

C Prefix Training

We used prefix training to increase stability and reduce flickering in the streaming setting. We conducted this by utilizing each training instance twice in each epoch: one as normal and the other with only the prefix. The length of the prefixes were randomly sampled from [0, 1]. We found that this additional data augmentation was particularly helpful; without it, the models would hallucinate the rest of a partial sentence.

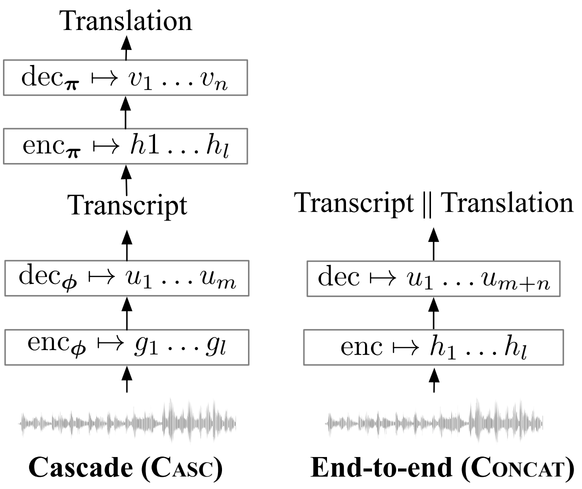


Figure 3: Architectures of the cascade and concatenated (E2E) model

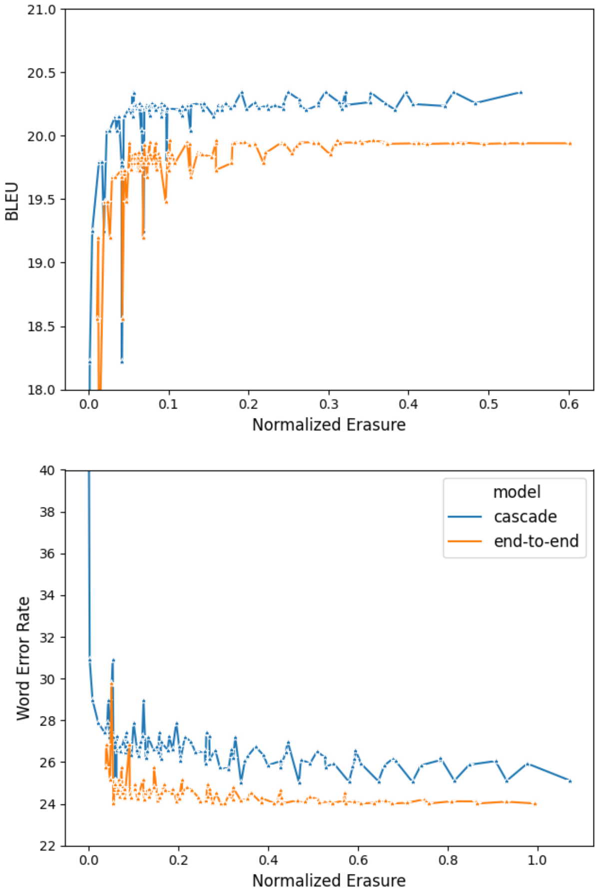


Figure 4: Left: average lag in seconds vs NE score. Right: NE vs WER score. All values are the mean of the results from the eight target languages.

We further found that starting the prefix sampling data augmentation too late in training was also negative. After testing initial models on the dev set, we found that starting this additional augmentation 15 epochs after training was best.

D Utterance Segmentation

We follow the audio segments provided in the MuST-C corpus, created through a use of human alignment and XNMT (Neubig et al., 2018). We note that there exist a variety of methods for creating segments for such models, however, we leave additional exploration of E2E alignment methods as future work.

DocEnt: Learning Self-Supervised Entity Representations from Large Document Collections

Yury Zemlyanskiy[*]
USC
`zemlyans@usc.edu`

Sudeep Gandhe
Google Research
`srgandhe@google.com`

Ruining He
Google Research
`ruininghe@google.com`

Bhargav Kanagal
Google Research
`bhargav@google.com`

Anirudh Ravula
Google Research
`braineater@google.com`

Juraj Gottweis
Google Research
`juro@google.com`

Fei Sha[†]
Google Research
`fsha@google.com`

Ilya Eckstein
Google Research
`ilyaeck@google.com`

Abstract

This paper explores learning rich self-supervised entity representations from large amounts of associated text. Once pre-trained, these models become applicable to multiple entity-centric tasks such as ranked retrieval, knowledge base completion, question answering, and more. Unlike other methods that harvest self-supervision signals based merely on a local context within a sentence, we radically expand the notion of context to include *any* available text related to an entity. This enables a new class of powerful, high-capacity representations that can ultimately distill much of the useful information about an entity from multiple text sources, without any human supervision.

We present several training strategies that, unlike prior approaches, learn to *jointly* predict words and entities—strategies we compare experimentally on downstream tasks in the TV-Movies domain, such as MovieLens tag prediction from user reviews and natural language movie search. As evidenced by results, our models match or outperform competitive baselines, sometimes with little or no fine-tuning, and can scale to very large corpora.

Finally, we make our datasets and pre-trained models publicly available[1]. This includes *Reviews2Movielens*, mapping the ∼1B word corpus of Amazon movie reviews (He and McAuley, 2016) to MovieLens tags (Harper and Konstan, 2016), as well as Reddit Movie Suggestions with natural language queries and corresponding community recommendations.

1 Introduction

Much of the online information describing entities in domains such as music, movies, venues or consumer products, is only available as unstructured text—a format that is human-readable but not machine-understandable (yet). Consider online reviews—a rich source of mostly user-generated about a vast number of entities. Our key research question is: *Can we learn strong models for entity understanding tasks such as vertical search and question answering, solely from text?* In other words, given a large and noisy collection of documents about an entity, can we distill all the useful information therein into a dense entity representa-

Review 1: *"This movie develops its power best if you don't try to look out for the "real" and "true" events behind the four versions of the narration... shown in a very intelligent and artistic way, no silly plot-twists, no explanation in the end — it is open to your fantasy... "$MOVIE" is an important piece of cinematic storytelling and a really interesting way to reflect on the origin of tales... Some scenes even remind me of Andrej Tarkovskijs intensive style..".*
Review 2: *"Just rented this, and at first I didn't like very much, but then it starts to sink in for how good it is, the acting is great especially Toshiro Mifune, it was shot very good for an older movie... it's #62 on the top 250"*
Review 3: *"Saw this movie at my local video store... was placed on a waiting list, but when I returned to check it out the video store had closed down over night. Actually whent out of business"*
... More reviews ...
Summary tags: [nonlinear] [multiple storylines] [japan] [black and white] [surreal] [cerebral] [imdb top 250], ...

Table 1: *Reviews2Movielens* task, illustrated. Here are sample review snippets for a certain classic film which is summarized using MovieLens tags. Notice that the tags may not appear in the input verbatim and can be thought of as boolean questions about the film. Note also that Review 3 has zero relevant signal—a common challenge of low SNR in this dataset. Bonus teaser: can you guess the $MOVIE from these snippets? This little quiz alludes to a key learning task in our approach.

---

[*] Work is partially done while at Google

[†] On leave from USC (feisha@usc.edu)

[1] See `http://goo.gle/research-docent` for *Reviews2Movielens* and models. Scripts and *Reddit Suggestions* can be found at `https://urikz.github.io/docent`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2540–2549
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

tion, so as to benefit multiple downstream tasks?

Traditionally, learning entity representations required supervised signals such as clicks, "likes" and consumption behavior (Agichtein et al., 2006; Huang et al., 2013; Koren et al., 2009; Vig et al., 2012a), which are generally expensive and time consuming to obtain at scale. To leapfrog these limitations, we draw inspiration from the recent progress in unsupervised learning of text, particularly contextualized representations via techniques such as ELMo (Peters et al., 2018), CoVe (McCann et al., 2017) and BERT (Devlin et al., 2019). Many of these representations are learned by predicting a missing word from its context. More recently, Sun et al. (2019) showed that extending word masking strategies to entities can lead to superior language models. Even more recent entity linking methods such as RELIC (Ling et al., 2020) and others, detailed in Section 6, were shown to produce explicit encodings applicable to entity understanding tasks.

We start with RELIC-like approaches and generalize them into a family of models, collectively called DocEnt, that jointly embed text and entities (Section 2) via self-supervised tasks. The first one, DocEnt-Dual, is essentially RELIC, but trained with a much broader context to include any and all sentences potentially related to an entity. Importantly, DocEnt-Dual/RELIC only optimizes a single task, namely *entity prediction* given an associated sentence, effectively modeling $P(Entity|Sentence)$.

Another natural way of jointly modelling entities and text is by directly tapping the cross-attention mechanism in BERT, simply by extending the BERT vocabulary to include entity tokens V_E. Each entity-related sentence can then be augmented with a corresponding token from V_E. We call this method DocEnt-Full and, despite (or perhaps because of) its conceptual simplicity, it proves surprisingly effective in semi-supervised tasks.

Finally, DocEnt-Hybrid aims to capture the best of both models by extending DocEnt-Dual with an additional task of predicting words in a sentence, conditioned on its associated entity. This task encourages the latter to "remember" salient phrases in its sentences.

We empirically evaluate these methods by learning entity representations for movies from a TV-Movies portion of the Amazon Reviews Corpus (He and McAuley, 2016). To this end, we consider several movie-oriented tasks for downstream evaluation, i.e. Reddit Movie Suggestions and MovieLens Tag Prediction (Harper and Konstan, 2016), which we study in both zero-shot, supervised and few-shot settings. We join the MovieLens dataset with the reviews corpus (He and McAuley, 2016) obtaining a mapping from movie reviews to user-generated tags. On the supervised tag prediction task, our text-based model demonstrates SOTA performance, despite not using powerful user signals (Vig et al., 2012a). In fact, we are able to match or outperform baselines on all tasks where they are available.

1.1 Contributions

1. First, we propose a family of methods to train deep self-supervised entity representations purely from related text documents, with strong zero-shot results on ranked retrieval with natural language queries.

2. Secondly, we show that these pre-trained representations are amenable to fine-tuning on new tasks such as MovieLens tag prediction, where we show state-of-the-art results. They are also effective few-shot learners, which we demonstrate on a harder *open-vocabulary*[2] task akin to Boolean Question Answering (Clark et al., 2019).

3. Next, we propose *Reviews2Movielens*—a new Text Based Entity Understanding task. The requisite dataset, which we release publicly, effectively joins the Amazon Movie Reviews Corpus and MovieLens into a large, sparsely supervised set with approximately 1B words and 470K movie-tag pairs.

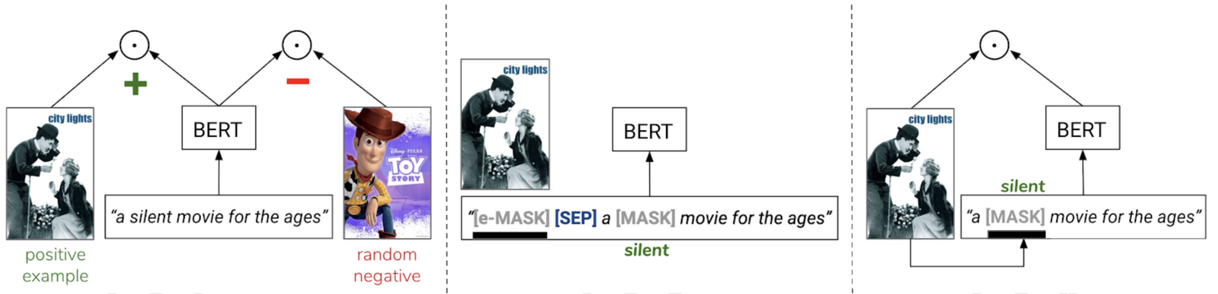
4. Finally, we also release a dataset of user-generated Reddit Movie Suggestions, a benchmark for natural language search and recommendation scenarios.

2 Self-Supervised Entity Representations

Inspired by the success of self-supervised language models, we seek to extend them to jointly compute text and entity representations. Recall that our input is a set of entities $\mathcal{E}$ where for every entity $e \in \mathcal{E}$, we have a collection of sentences, denoted by $\mathcal{S}_e$, from all documents related to e. Intuitively, we want the representation of e to be influenced by each associated sentence $s \in \mathcal{S}_e$, and vice versa.

---

[2] An open vocabulary allows any phrase to be a label.



DocEnt-Dual (RELIC) DocEnt-Full DocEnt-Hybrid

Figure 1: Models in the DocEnt family. Left: a baseline dual encoder model called DocEnt-Dual a.k.a. RELIC, maximizing $P(e|s)$ but not $P(s|e)$. Center: DocEnt-Full—a model maximizing the joint sentence-entity probability using full cross-attention. Right: DocEnt-Hybrid, designed to capture the best of both worlds.

To that end, we explore two (self-) supervision signals: $P(e \mid s)$ and $P(s \mid e)$.

2.1 DocEnt-Dual, Known as RELIC

At the core of DocEnt-Dual is a RELIC model that co-encodes an entity e and an associated sentence $s \in \mathcal{S}_e$ so as to maximize their compatibility score, defined as the cosine similarity between the two encodings:

$$\mathbf{s}(e, s) = \frac{g(e)^T f_{CLS}(s)}{\|g(e)\| \|f_{CLS}(s)\|},$$

where $g(e)$ is an embedding of e and $f(s)$ is a BERT-based encoding of s, with its special $[CLS]$ token whose output representation is denoted by f_{CLS}. Then, the conditional probability of e given s is given by a softmax over the set $\mathcal{E}$ [3]:

$$P(e|s) = \frac{\exp(\mathbf{s}(e, s))}{\sum_{e' \in \mathcal{E}} \exp(\mathbf{s}(e', s))}.$$

Finally, RELIC is trained by maximizing $\log P(e|s)$ over all associated pairs $e, s \in \mathcal{S}_e$:

$$\mathcal{L}_E(e, s) = \log P(e|s).$$

Note that both g and f (initialized with a common BERT) are learned during training.

Our sole difference to the original RELIC is in training data: while RELIC only uses sentences containing entity mentions, we allow a radically broader context – all sentences associated with an entity – with the goal of remembering all of its attributes. Crucially, no human labeling is required.

Despite its effectiveness (as demonstrated in Section 5), RELIC has one obvious limitation: it ignores $P(s \mid e)$, leaving a useful signal "on the table". We therefore propose another way of co-encoding sentences and entities by tapping the full cross-attention power of Transformers.

2.2 DocEnt-Full

Before we proceed, let us revisit BERT's Masked Language Model (MLM) training objective. Given a sequence of input tokens $s = [s_1, \ldots, s_n]$, a fraction of tokens s_J at randomly selected positions J is replaced with a special *[MASK]* token. We denote this new sequence by s_{-J}.

Then, BERT predicts masked tokens based on their contextualized representations $f(s_{-J})$. The MLM training objective to maximize is:

$$\mathcal{L}_{MLM} = \log P(s_J \mid s_{-J}).$$

Enter DocEnt-Full. It follows the standard BERT architecture, with a twist. First, we expand the input vocabulary to include all entity tokens in $\mathcal{E}$. Then, during input sequence construction, each sentence $s \in \mathcal{S}_e$ is prepended[4] with the corresponding entity token e, as shown in Figure 1. This way, masking and predicting this token (via softmax) effectively adds our new objective $\mathcal{L}_E$ to BERT. Further, the new e token is now part of a sentence context, augmenting the original $\mathcal{L}_{MLM}$ to

$$\mathcal{L}_{MLM+E}(s, e) = \log P(s_J \mid s_{-J}, e),$$
$$\text{and } \mathcal{L}_{\text{FULL}} = \mathcal{L}_E + \lambda \mathcal{L}_{MLM+E}$$

becomes the combined loss function optimized using nothing but BERT's standard MLM training, with a hyperparameter λ to balance the two terms[5].

---

[3] In practice, only a subset of entities in $\mathcal{E}$ is used in the denominator: the so called "in-batch negatives".

[4] Technically, we replace BERT's standard (s_A, s_B) two-segment input structure with (e, s), for $s \in \mathcal{S}_e$.

[5] The relative masking frequency of entity tokens is another hyperparameter available to balance the two objectives.

This conceptual simplicity and full cross-attention power come with a cost: bundling wordpieces and entities together forces the model to allocate an equal capacity to both types of tokens (e.g., 768D for BERT-base), regardless of the size of $\mathcal{E}$. As a result, a relatively small-sized $\mathcal{E}$ may be prone to overfitting[6] in zero-shot scenarios, as we observe in Section 5.4.2.

2.3 DOCENT-HYBRID

Recall that RELIC avoids the above limitation by decoupling text and entity encoders. To get the best of both worlds, we introduce DOCENT-HYBRID—a third model that sticks with the modular dual encoder architecture while also modeling $P(s \mid e)$. This is achieved by implementing a different variant of $\mathcal{L}_{MLM+E}$ where, for every masked wordpiece token, the output of Transformer layers $f(s_{-J})$ is first concatenated with the associated entity embedding $g(e)$ before feeding into the final MLM prediction layer. By including entity embeddings in the prediction of related text tokens, we get them to "remember" important aspects from the text without sacrificing modularity.

3 Tasks

In this section, we define the three tasks used to evaluate pre-trained entity representations.

3.1 Supervised Task: Movielens Tag Prediction

The original MovieLens Tag Prediction task is to produce movie-tag scores for a set of movies and a canonical vocabulary of tags (see examples in Table 1), based on a collection of crowdsourced (movie, tag, user) votes, as well as (user, movie) star ratings. These tags are often not factual but may refer to plot elements, qualitative aspects or reflect subjective opinions. Since the same can be said about user reviews, and we observe a non-trivial amount of textual entailment between the two sources. We therefore intentionally exclude user ratings from the input. The new challenge is to complete the movie-tag relevance matrix by leveraging movie reviews, hereafter referred to as the *closed-vocabulary tag prediction* task[7]. This is a supervised setup where models are fine-tuneed with tag labels and evaluated on a held-out set subset of movies, as elaborated in Section 5.

3.2 Few-Shot Task: Open Vocabulary Tag Prediction

In reality, the space of tags is not static. Rather, tags are a useful kind of user-generated content that evolves to reflect the zeitgeist, much like human language. Many online platforms (e.g, Twitter and Instagram to name a few) have vibrant online communities that keep inventing new tags. We therefore propose a new *open-vocabulary* formulation of the tag prediction problem where any phrase is allowed to be a tag.

This requires a small change in evaluation. Instead of held-out movies, we hold out a subset of tags and fine-tune on the rest (and on all the movies). Note that this is no longer a classic multi-label classification task as we never get to see the test labels during training. Rather, this open-vocabulary setup is akin to answering boolean questions (about a movie) based on a text document (Clark et al., 2019).

3.3 Zero-Shot Task: Reddit Movie Suggestions

The purpose of this task is to evaluate pre-trained entity representations in the context of vertical search. The classic entity ranking problem is, given a text query and a finite set of entities, to rank them according to their relevance to the query. Recall that DOCENT models are naturally designed to make such relevance predictions via $P(Entity|Sentence)$ — without any fine-tuning, if necessary. We therefore leverage the Reddit Movie Suggestions Dataset (detailed in Section 4.3) as a source of both queries and ground truth to define a zero-shot movie ranking task. To clarify, the notion of *zero shot* implies a pre-trained but not fine-tuned model in our context. This dataset is particularly interesting for its challenging queries, with their distinctly natural, often conversational language (e.g., *"Last week I watched the British cold war movie Threads. I am scarred, but intrigued as well. Any similar deeply disturbing yet realistic movies you can recommend?"*, see Table 2 for more examples). Another challenge is an explicit recommendation intent present in many of the queries (i.e., *"Movies like ..."*), making this task a mixture of Search and Recommendation. The latter typically requires specialized recommendation models of entity-to-entity

---

[6] Conversely, a very large $\mathcal{E}$ may require an optimized implementation of softmax to maintain scalability.

[7] One can also view this as a two-dimensional knowledge base (KB) completion problem, where relation types are not available and the KB is reduced to a 2D matrix.

Query	**Top 5 Results**
Movies like [**Whiplash**] about an artist or a musician chasing an almost impossible dream and nearly or does ruin his life because of it	Inside Llewyn Davis, **Whiplash**, A Young Man with a Horn, Hustle & Flow, Born to Be Blue
Really dark, slow paced movies with minimal story, but incredible atmosphere, kinda like [**Drive**] or [**The Rover**]	**The Rover**, Valhalla Rising, Only God Forgives, Blade Runner, Sicario
Films like [**Mission Impossible**] or [**The Italian Job**] that have big scenes where the characters must break in or infiltrate some place	National Treasure: Book of Secrets, **Mission: Impossible – Rogue Nation**, Ant-Man, **The Italian Job**

Table 2: Qualitative examples illustrating zero-shot movie ranking by DOCENT-FULL, with natural language queries crawled from Reddit. The bracketed greyed-out movie mentions are users' examples of desired recommendations, removed from the queries to probe the model in what resembles a movie guessing game. Those obfuscated entities were correctly guessed by the model based on remaining query terms, making it to the Top 5 in most cases. Other top matches appear to be equally relevant.

similarity, and cannot generally be solved with keyword-based search.

4 Datasets

4.1 Amazon Movie Reviews Corpus

All our models are pretrained on Amazon Product Reviews (He and McAuley, 2016) in the "Movies and TV" category, comprising 4,607,047 reviews for 208,321 movies collected during 1996–2014[8].

4.2 Reviews2Movielens

One of this paper's contributions is *Reviews2Movielens*—a new multi-document multi-label dataset created by joining Amazon Movie Reviews (He and McAuley, 2016; Ni et al., 2019) and MovieLens (Harper and Konstan, 2016), a rich source of crowdsourced movie tags. The key challenge in joining the two datasets is establishing correspondences between their respective movie IDs, which turns out to be a many-to-one mapping[9]. We have identified a subset of high-precision many-to-one correspondences by applying Named Entity Recognition techniques[10] to both Amazon product titles (incl. release years) and their product pages. The resulting mapping consists of 71,077 unique Amazon IDs and 28,918 unique MovieLens IDs. The mapping accuracy was manually verified to be 97% based on 200 random samples. Ultimately, the joined dataset contains nearly 2 million reviews and close to 1B words, significantly more than its IMDB counterpart (Maas et al., 2011).

Since both datasets are widely used as a source of data and academic benchmarks (Miller et al., 2003; Jung, 2012; Anand and Naorem, 2016; He and McAuley, 2016; Ni et al., 2019), we hope that this new mapping[11] will be useful to the community.

4.3 Reddit Movie Suggestions

This user-generated dataset contains a collection of 4765 movie-seeking queries and corresponding recommendations, collectively curated and voted on by the Reddit Movie Suggestions community[12]. Worth noting are (a) the conversational, human-to-human language of the queries; (b) the community-recommended movies that, while sparse and possibly biased, can be used as a source of ground truth. While modest in size, the dataset is well-suited to evaluate zero-shot performance on the movie ranking task defined in Section 3.3.

5 Experiments

5.1 Pre-training

All our experiments start with pre-training models on the Amazon Movie Reviews corpus, followed by optional task-dependent fine-tuning. First, we apply some simple filtering to the input, removing reviews shorter than 5 words and movies with less than 5 reviews [13]. This results in 81,057 Amazon movies, of which 17,131 have MovieLens correspondences, and 4,181,727 reviews in total. Further, we split reviews into individual sentences (or short paragraphs) so as to circumvent the BERT

---

[8] We've used the 2016 version of the dataset from `http://jmcauley.ucsd.edu/data/amazon`.

[9] Each Amazon ID (ASIN) matches a canonical product URL, e.g., `https://www.amazon.com/dp/B06XGG4FFD`. However, these IDs correspond to specific product *editions* (typically DVDs) rather than unique titles, causing duplication issues. Some are collections of several titles.

[10] We use the public Google Cloud Natural Language API – `https://cloud.google.com/natural-language/docs/basics#entity%20analysis`.

[11] See `http://goo.gle/research-docent`

[12] `https://www.reddit.com/r/MovieSuggestions`

[13] This low-count filtering is applied after de-duplication and aggregation.

sequence length limit. Finally, since our goal is to learn non-obvious entity attributes, we remove movie names from their reviews.

All our models use the standard BERT-base configuration with 12 layers, 12 attention heads and a hidden size of 768, and are initialized with a publicly available BERT-base checkpoint[14].

5.2 Tag Prediction: Fine-tuning Strategies

We will now describe the fine-tuning strategies used to transfer pre-trained DOCENT models to downstream tag prediction tasks.

DOCENT-FULL To generate movie-tag relevance scores, we need to predict $P(Tag|Movie)$, which we cast as binary classification. Recall that BERT has a built-in binary classifier (for next-sentence prediction), implemented as a single-layer FFN [15] on top of its *[CLS]* output, with logistic loss. We simply repurpose that layer for our task.

DOCENT-DUAL and DOCENT-HYBRID Recall that, during pre-training, DOCENT-DUAL and DOCENT-HYBRID use softmax cross entropy loss to predict $P(Entity|Sentence)$. However, tag prediction poses the inverse problem: predict tags based on a movie entity. In our dual encoder framework, that can be done simply by computing softmax over all of the encoded tags rather than entities, without any changes to the architecture.

Shared Strategies For fine-tuning, all of the models share the following choices. First, we treat every existing movie-tag pair in the training set as a positive example, weighted proportionally to the number of user votes for that pair (or to the logarithm thereof). Next, for a given movie, about 10% of all vocabulary tags are sampled as negative examples, excluding the known true positives for that movie. To prevent overfitting, we fix entity embedding weights for all models during fine-tuning.

5.3 Entity-less Baselines

To corroborate the utility of explicit entity representations, we set out to evaluate a few baselines that circumvent them by representing each entity as a Bag-of-Sentences (BoS), computed over its related reviews with a sentence encoder of choice. Such a BoS encoder can replace entity embeddings in our architecture, yielding a naïve variant of DOCENT-DUAL. We call these baselines BOS-GLOVE, BOS-BERT and BOS-SENTENCEBERT[16], reflecting their underlying sentence encoders.

Task	Movies	Tags	M-T Pairs
Closed (test)	1000	1128	46359
Closed (dev)	380	1128	17943
Open (test)	6392	500	141618
Open (dev)	3362	100	25274

Table 3: Evaluation datasets sizes for Tag Prediction tasks. Closed / Open stand for the closed and open vocabulary tasks, respectively; M-T Pairs shows the number of corresponding movie-tags pairs. The top two rows describe *movie* holdout sets used in our closed vocabulary experiments; bottom two rows showing *tag* holdouts for open vocabulary experiments.

5.4 Evaluation

5.4.1 Movielens Tag Prediction

The main challenge with evaluating tag prediction is the sparse and noisy nature of user-generated ground truth. For instance, a certain movie tag having zero votes may still be relevant in reality. On the other hand, some entities may have votes for contradictory tags (e.g., both "funny" and "not funny"). The original Tag Genome baseline (Vig et al., 2012b) mitigated this by collecting an additional dataset of unbiased movie-tag relevance scores. Alas, that data has not been released. Instead, we propose two complementary metrics that cast tag prediction either as binary classification or as a ranking problem.

For classification, we binarize labels as follows. Let $\#(m, t)$ be the number of users who assigned a tag t to a movie m. Then its binary counterpart $l(m, t)$ is set to 1 iff $\#(m, t) > T$, a threshold[17].

For the tag ranking formulation, we make the assumption that true movie-tag relevance is correlated with the number of movie-tag votes, and define our movie-tag relevance score as $r(m, t) = \#(m, t)$.

Equipped with this score, we use Precision@k and NDCG metrics (Järvelin and Kekäläinen, 2002) to measure performance.

Tag prediction baselines include

MovielensTopTags— a fixed ordering of tags.

---

[14] `https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip`

[15] Feed-Forward Neural Network

[16] SENTENCEBERT (Reimers and Gurevych, 2019) fine-tunes BERT on NLI to provide off-the-shelf semantic sentence representations.

[17] We use $T = 2$ to filter out noisy tags.

Model	MAP	AUC
MovielensTopTags	6.2	0.80
TD-IDF	32.3	0.86
BoS-BERT	39.3	0.91
TagGenome	43.9	**0.98**
DocEnt-Full	**44.7**	**0.98**
DocEnt-Dual	38.6	0.96
DocEnt-Hybrid	44.1	**0.98**
Human	76.6	0.99

Table 4: Mean Average Precision and ROC-AUC results on the closed-vocabulary tag prediction task. TagGenome is the original baseline from MovieLens creators (Vig et al., 2012b), trained on multiple additional features and considered SOTA. Despite using fewer features, DocEnt matches TagGenome performance on AUC and outperforms it on precision (MAP).

TF-IDF scores for movie-tag pairs, based on tag frequencies in movie reviews.

BoS-BERT, as defined in Sec. 5.3, is fine-tuned to estimate sentence-to-tag relevance directly[18]. This setup is applicable to both open and closed vocabulary scenarios. During inference, a movie-tag prediction is obtained by averaging over sentence-wise predictions for the movie's reviews.

TagGenome—the original baseline from MovieLens team (Vig et al., 2012b). The comparison is not entirely apt as that model was trained on additional movie-tag relevance data and user ratings, albeit with a smaller corpus of unsupervised reviews. Also, TagGenome was trained on all of MovieLens (no holdouts).

Humans—to simulate human performance, apply cross-validation to ground truth user votes, treating one of the folds as a quasi-model.

All models were evaluated on the same holdout sets, with averaging.

Closed Vocabulary Tag Prediction In this scenario, evaluation is done on a holdout set of movies (with a smaller development set used for hyperparameter tuning; see Table 3 for details).

Results for ranking (MAP) and binary classification (AUC) metrics are shown in Table 4. Collectively, DocEnt models outperform the strong TagGenome baseline on tag ranking (see also Fig. 2 (a) and (b)) and match (or slightly outperform) it in binary classification. It is a strong result, considering that DocEnt had no access to additional features used by TagGenome and employed no feature engineering. Of the three models, DocEnt-Dual scores the lowest on all metrics, likely due to not optimizing for $P(Text \mid Entity)$ in pre-training. Finally, note that all models still score way below humans on the (harder) tag ranking task, indicating considerable headroom.

Model	MRR	Recall, %	
		@50	@100
Lucene (TF-IDF)	0.14	15.3	20.7
BoS-GloVe	0.04	4.1	6.6
BoS-BERT*	0.08	9.6	14.2
BoS-SentenceBert	0.07	7.6	11.7
DocEnt-Full	0.22	21.3	28.4
DocEnt-Dual	0.27	28.0	36.3
DocEnt-Hybrid	**0.31**	**31.9**	**40.9**

Table 5: Zero-shot results for DocEnt models vs several baselines on Reddit Movie Suggestions. MRR stands for Mean Reciprocal Rank.

Open Vocabulary Tag Prediction This task is evaluated by withholding parts of the tag vocabulary so that those tags are never seen in training (consult Table 3 for details). Fig. 2 (c) shows our models' performance on the binary classification task base on the fraction of the vocabulary seen by a model in fine-tuning. The graph shows that training with only 100 of the 1124 tags results in reasonable performance. Of our three models, DocEnt-Full starts below the others but adapts the fastest, reaching a near-closed vocabulary performance with less than 50% of the full tag vocabulary.

5.4.2 Reddit Movie Suggestions

Movie suggestion baselines Since this is a search task, we compare our models to an Apache Lucene[19] baseline, arguably the world's most widely used open-source search engine. For completeness, we also compare to BoS-BERT*[20], BoS-GloVe and BoS-SentenceBert, neural baselines defined in Sec. 5.3, whose query-movie relevance score is given by the maximum cosine similarity among the movie's review sentences[21].

Table 5 shows the Mean Reciprocal Rank (MRR) as well as recall, metrics that suit the noisy ground

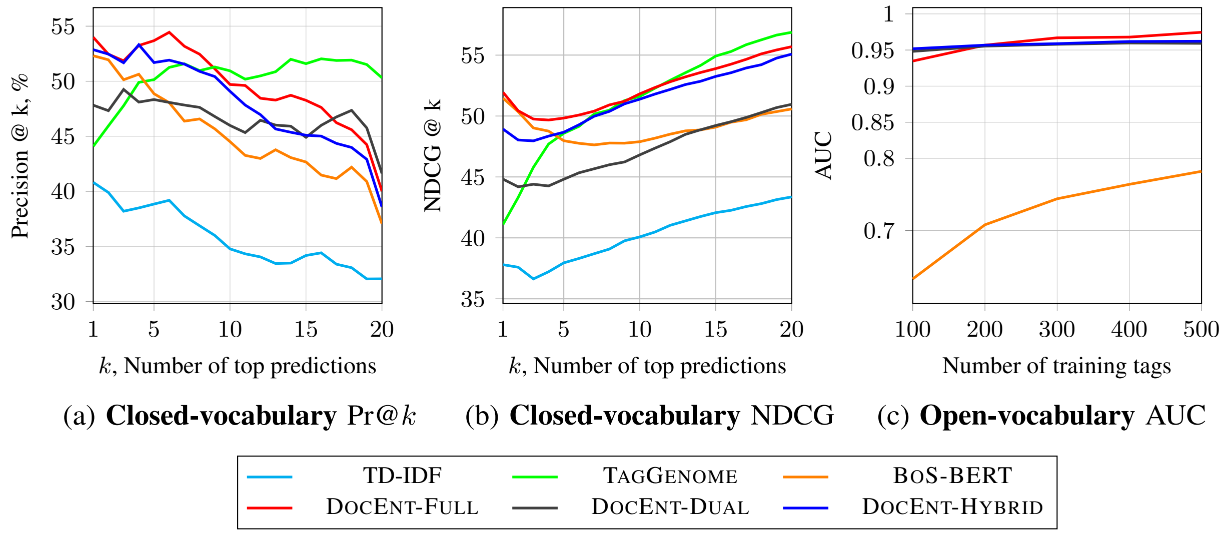
---

[18] We found it is best to encode a review sentence using BERT's *[CLS]* output, while tags are encoded by averaging individual tokens' output vectors.

[19] `https://lucene.apache.org/`

[20] In absence of a fine-tuned *[CLS]* output, this version of BoS-BERT encodes sentences by averaging their individual tokens' output vectors.

[21] In this case, we found that aggregating sentence-wise predictions with L^∞ norm is superior to averaging.



(a) **Closed-vocabulary** Pr@k (b) **Closed-vocabulary** NDCG (c) **Open-vocabulary** AUC

Figure 2: Performance on tag prediction tasks. Left and center: Precision and NDCG @k, with a closed vocabulary. DOCENT-FULL dominates the strong TAGGENOME baseline for smaller values of k, a concentration of gains typical for binary classification models. For perspective, human Precision@k ranges 80-95% for this task. Right: AUC for open vocabulary experiments, with models trained using a variable fraction of the tag vocabulary. DOCENT approaches close-vocabulary AUC after training with only 10-50% of the vocabulary (showing all baselines that were available to us in this setting).

truth (for completeness, see also the qualitative results in Table 2). DOCENT models outperform the Lucene baseline on all metrics, with DOCENT-HYBRID leading by a large margin. Compared to DOCENT-DUAL, its strong performance is not surprising since DOCENT-HYBRID optimizes both $P(Entity \mid Text)$ and $P(Text \mid Entity)$—a combination of tasks that helps avoid overfitting.

Also expected is the relatively weak performance of DOCENT-FULL. As discussed in Sec. 2, its high-capacity entity representations are prone to overfitting when the number of entities is relatively small. Still, this shortcoming can be remedied by fine-tuning, as evidenced by this model's superior results on tag prediction in Sec. 5.4.1. These results suggest that DOCENT-FULL may be a good choice in semi-supervised scenarios.

6 Related Work

Much of the prior art in text-based entity understanding is motivated by the *Entity Linking* (EL) problem: predict a unique entity from its mention in text, assuming a single right answer. By contrast, tasks like entity retrieval and tag prediction imply multiple valid matches and emphasize understanding entities through the prism of their attributes, expressed in natural language. Still, recent EL works propose dual encoder approaches similar to ours (Yamada et al., 2017; Ling et al., 2020; Cheng and Roth, 2013; Sun et al., 2015; Yamada et al., 2016; Chang et al., 2020; Kobayashi et al., 2016; He et al., 2013; Gupta et al., 2017), with Ling et al. (2020) already discussed in Section 2.1. Dual encoders have also been explored in zero-shot scenarios (Gillick et al., 2019; Logeswaran et al., 2019; Wu et al., 2019; Gupta et al., 2017), with entity embeddings computed dynamically based on metadata such as dictionary definitions, entity name and/or category. Others incorporate entity representations directly in the transformer by retrieving from an external memory (Févry et al., 2020; Peters et al., 2019). While clearly useful for EL, e.g., in sentences with multiple entity mentions, the benefits to our applications are unclear. Finally, there is ERNIE (Sun et al., 2019) – a language model trained with awareness of entity mentions. Alas, the lack of explicit entity representation limits its use in our tasks.

7 Conclusion & Future Work

This paper proposes a family of models to learn self-supervised entity representations from large document collections. We motivate these dedicated representations by contrasting them with naive text-as-a-proxy approaches, with clear gains on entity-centric tasks such as natural language search and movie tag prediction. We then show that achieving superior performance requires optimizing both $P(Entity \mid Text)$ and $P(Text \mid Entity)$—in contrast to the baseline RELIC model (and similar prior dual encoders) having only a single objective. To that end, we propose two novel models and study them in zero-shot, few-shot and supervised settings. We match or outperform competitive baselines, where available, with little or no fine-tuning.

Future Work As shown qualitatively in Sec. 3.3, DOCENT has the potential for being a hybrid approach to bridge entity retrieval and recommendation, an application worth exploring in depth (e.g., on the MovieLens Recommendation task which can be readily integrated with DOCENT thanks to *Reviews2Movielens*). A larger entity retrieval study with heterogeneous entity types is another useful direction. Lastly, extending DOCENT to additional entity understanding tasks such as QA and summarization is yet another promising avenue.

Acknowlegements

We appreciate the feedback from the reviewers. This work is partially supported by NSF Awards IIS-1513966/ 1632803/1833137, CCF-1139148, DARPA Awards#: FA8750-18-2-0117, FA8750-19-1-0504, DARPA-D3M - Award UCB-00009528, Google Research Awards, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484.

References

Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26. ACM.

Deepa Anand and Deepan Naorem. 2016. Semi-supervised aspect based sentiment analysis for movies using review filtering. *Procedia Computer Science*, 84:86–93.

Jill Burstein, Christy Doran, and Thamar Solorio, editors. 2019. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval.

Xiao Cheng and Dan Roth. 2013. Relational inference for wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1787–1796, Seattle, Washington, USA. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In (Burstein et al., 2019), pages 2924–2936.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In (Burstein et al., 2019), pages 4171–4186.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego García-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*, pages 528–537. Association for Computational Linguistics.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2681–2690. Association for Computational Linguistics.

F. Maxwell Harper and Joseph A. Konstan. 2016. The movielens datasets: History and context. *TiiS*, 5(4):19:1–19:19.

Ruining He and Julian J. McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 507–517. ACM.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning entity representation for entity disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pages 30–34. The Association for Computer Linguistics.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338. ACM.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Jason J Jung. 2012. Attribute selection-based recommendation framework for short-head user group: An empirical study by movielens and imdb. *Expert Systems with Applications*, 39(4):4049–4054.

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representation with max-pooling improves machine reading. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 850–855. The Association for Computational Linguistics.

Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37.

Jeffrey Ling, Nicholas FitzGerald, Zifei Shan, Livio Baldini Soares, Thibault Févry, David Weiss, and Tom Kwiatkowski. 2020. Learning cross-context entity representations from text. *CoRR*, abs/2001.03765.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3449–3460.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6294–6305.

Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. 2003. Movielens unplugged: experiences with an occasionally connected recommender system. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266. ACM.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1333–1339. AAAI Press.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Jesse Vig, Shilad Sen, and John Riedl. 2012a. The tag genome: Encoding community knowledge to support novel interaction. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(3):13.

Jesse Vig, Shilad Sen, and John Riedl. 2012b. The tag genome: Encoding community knowledge to support novel interaction. *TiiS*, 2(3):13:1–13:44.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2019. Zero-shot entity linking with dense entity retrieval.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259, Berlin, Germany. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Trans. Assoc. Comput. Linguistics*, 5:397–411.

Scientific Discourse Tagging for Evidence Extraction

Xiangci Li[1]* **Gully Burns**[2] **Nanyun Peng**[3]
[1] University of Texas at Dallas, [2] Chan Zuckerburg Initiative
[3] University of California Los Angeles
`lixiangci8@gmail.com`,`gully.burns@chanzuckerburg.com`
`violetpeng@cs.ucla.edu`

Abstract

Evidence plays a crucial role in any biomedical research narrative, providing justification for some claims and refutation for others. We seek to build models of scientific argument using information extraction methods from full-text papers. We present the capability of automatically extracting text fragments from primary research papers that describe the evidence presented in that paper's figures, which arguably provides the raw material of any scientific argument made within the paper. We apply richly contextualized deep representation learning pre-trained on biomedical domain corpus to the analysis of scientific discourse structures and the extraction of "evidence fragments" (i.e., the text in the results section describing data presented in a specified subfigure) from a set of biomedical experimental research articles. We first demonstrate our state-of-the-art scientific discourse tagger on two scientific discourse tagging datasets and its transferability to new datasets. We then show the benefit of leveraging scientific discourse tags for downstream tasks such as claim-extraction and evidence fragment detection. Our work demonstrates the potential of using evidence fragments derived from figure spans for improving the quality of scientific claims by cataloging, indexing and reusing evidence fragments as independent documents.

1 Introduction

Primary experimental articles (i.e., papers that describe original experimental work) provide the crucial raw material for all other subsequent scientific research. However, the drastically growing number of scientific literature makes it increasingly difficult for domain experts to efficiently utilize them. Automatic information extraction from biomedical literature is a crucial step to help researchers to achieve this goal.

lc3 , the mammalian atg8 homolog , undergoes a set of modifications resulting in conversion from lc3i to lc3ii during autophagy 42 . **[fact]** to further test the function of rag in autophagy **[goal]** we examined the lc3 modification in hek293 cells . **[method]** expression of raga ql and ragc sn inhibited lc3 conversion in response to amino acid starvation (fig. 7e) . **[result]** furthermore , expression of raga tn and ragc ql enhanced lc3 conversion even in the presence of amino acids . **[result]** these results are consistent with the data observed in drosophila and further demonstrate a role of the rag gtpases in autophagy regulation in response to nutrient signals **[implication]**

Figure 1: An example paragraph tagged with scientific discourse tags on each clause in SciDT dataset (Dasigi et al., 2017). The text is tokenized and converted to lower case.

Extracting important information from biomedical literature to facilitate and accelerate scientific discovery has been a goal for computational linguistics for some time (Hobbs, 2002), with the focus of identifying relevant entities, relations, and events from text to populate a knowledge base. However, these methods do not take into account the fact that scientific work involves attempting to provide explanations for evidence derived from experiments and is therefore driven principally by authors attempting to convince expert readers that their claims are the "correct" explanations for the experimental evidence. Thus, an important aspect of building machines capable of understanding scientific literature is first recognizing different *rhetorical components* of scientific discourse, with which we will then be able to distinguish the *observations* made in experiments from their *implications* and distinguish between *claims* supported by evidence and *hypotheses* put forward to prompt further research. It is this goal, of being able to distinguish between the different rhetorical components of scientific discourses so that we can build AI systems to facilitate more accurate analysis and understand-

---

* Work performed while the author is interning at the Information Sciences Institute, University of Southern California

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2550–2562
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Type	Definition
Goal	Research goal
Fact	A known fact, a statement taken to be true by the author
Result	The outcome of an experiment
Hypothesis	A claim proposed by the author
Method	Experimental method
Problem	An unresolved or contradictory issue
Implication	An interpretation of the results
None	Anything else

Table 1: Eight label taxonomy defined by De Waard and Maat (2012).

ing of scientific literature, that motivates our work.

Scientific discourse tagging is a task that tags clauses or sentences in a scientific article with different rhetorical components of scientific discourses. Figure 1 shows an example of a paragraph with discourse tags. In this work, we leverage a state-of-the-art contextualized word embedding and a novel word-to-sentence attention mechanism to develop a model for scientific discourse tagging that achieves the state-of-the-art performances on two benchmark datasets SciDT (Dasigi et al., 2017) and PubMed-20k-RCT (Jin and Szolovits, 2018) by 6.9% and 2.3% absolute F1 respectively. [1] More importantly, we show the strong transferability of our scientific discourse tagger to new datasets by beating the baseline (Huang et al., 2020) via zero-shot prediction on CODA-19 dataset (Huang et al., 2020). Furthermore, we demonstrate the effectiveness of scientific discourse tagging on two downstream scientific literature understanding tasks: claim-extraction and evidence fragment detection, and demonstrate the benefit of leveraging scientific discourse tags information. In particular, we outperform the state-of-the-art claim extraction model (Achakulvisut et al., 2019) by 3.8% F1, and outperform figure span detection baseline (Burns et al., 2017) by 5% F1.

2 Background and Related Works

Problem Formulation. We define scientific discourse tagging as a task that labels sentences in a scientific article based on its rhetorical elements of scientific discourse. Formally, a paragraph can be represented as an ordered collection of sequences $\mathbf{S} = [S_1, S_2, ..., S_n]$, and each element S_i is annotated with a discourse label $L_i \in \{L_1, L_2, ..., L_k\}$. Note that S_i may be defined differently in different datasets – e.g., sentences in the PubMed-RCT dataset (Dernoncourt and Lee, 2017), clauses in the SciDT dataset composed by Burns et al. (2016) and Dasigi et al. (2017), and sentence fragments in CODA-19 dataset (Huang et al., 2020). For conciseness, we refer all these variations as *sentences*. The labels also can be slightly different. For example, in PubMed-RCT, $\mathbf{L} = \{$*objective*, *background*, *methods*, *results*, *conclusions*$\}$, in CODA-19 (Huang et al., 2020), $\mathbf{L} = \{$*background*, *purpose*, *method*, *finding/contribution*, *other*$\}$ while in SciDT dataset (Burns et al., 2016; Dasigi et al., 2017), the labels $\mathbf{L} = \{$*goal*, *fact*, *hypothesis*, *problem*, *method*, *result*, *implication*, *none*$\}$ as defined by De Waard and Maat (2012). Table 1 gives more details about the definitions of the tags.

2.1 Prior Works on Scientific Discourse Tagging

Feature-based Scientific Discourse Tagging. There has been a significant amount of work aimed at understanding types of scientific discourse. Teufel and Moens (1999) and Teufel and Moens (2002) described argumentative zoning, which groups sentences into a few rhetorical zones highlighted by important clauses such as "in this paper we develop a method for". Hirohata et al. (2008) used conditional random field (CRF) (Lafferty et al., 2001) with handcrafted features to classify sentences in abstracts into 4 categories: *objective, methods, results*, and *conclusions*. Liakata (2010) defined "zone of conceptualization" which classifies sentences into 11 categories in scientific papers and Liakata et al. (2012) used CRF and LibSVM to identify these "zone of conceptualization". Guo et al. (2010) used Naive Bayes and Support Vector Machine (SVM) (Cortes and Vapnik, 1995) to compare three schema: section names, argumentative zones and conceptual structure of documents. Burns et al. (2016) studied the problem of scientific discourse tagging, which identifies the discourse type of each clause in a biomedical experiment paragraph and composed a dataset for it. They adopted the discourse type taxonomy for biomedical papers proposed by De Waard and Maat (2012). The taxonomy contains eight types including *goal, fact, result, hypothesis, method, problem, implication* and *none* as Table 1 shows. Most recently,

---

[1] `https://github.com/jacklxc/ScientificDiscourseTagging`

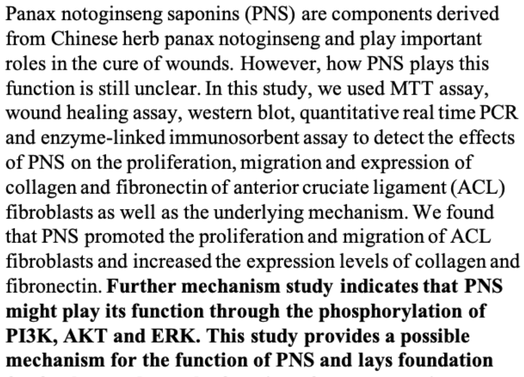
Cox et al. (2017) used the same schema (De Waard and Maat, 2012) by exploring a variety of methods for balancing classes before applying classification algorithms.

Deep Learning for Scientific Discourse Tagging. Due to the prevalence of deep learning, neural sequence labeling approach using bidirectional LSTM (Hochreiter and Schmidhuber, 1997) and CRF (BiLSTM-CRF) (Huang et al., 2015) has been prevailing for classic word-level sequence tagging problems such as named entity recognition (NER), part of speech tagging (POS), and word segmentation (Huang et al., 2015; Peng and Dredze, 2015, 2016; Ma and Hovy, 2016; Chiu and Nichols, 2016; Peng and Dredze, 2017; Wang et al., 2017; Huang et al., 2019). Since scientific discourse tagging, which is a sentence-level sequence tagging problem, has one additional dimension of input comparing to word-level sequence tagging problems, an encoder is required to encode word-level representations to clause/sentence-level representations. While one simple way is to pre-compute sentence embeddings from word embeddings (Arora et al., 2016), there are more sophisticated methods to compute sentence-level embeddings on-the-fly using BiLSTM (Jin and Szolovits, 2018; Srivastava et al., 2019) or attention (Dasigi et al., 2017), before feeding them into a clause/sentence-level sequence tagger. Alternatively, as BERT (Devlin et al., 2018) prevails among various natural language processing (NLP) tasks, a simple baseline method is to directly use a BERT-like model's (e.g. SciBERT (Beltagy et al., 2019)) prefix token ($[CLS]$) representation of each sentence as the sentence representation for classification task (Huang et al., 2020). In this work, we combine these methods to present a state-of-the-art scientific discourse tagger.

2.2 Downstream Applications

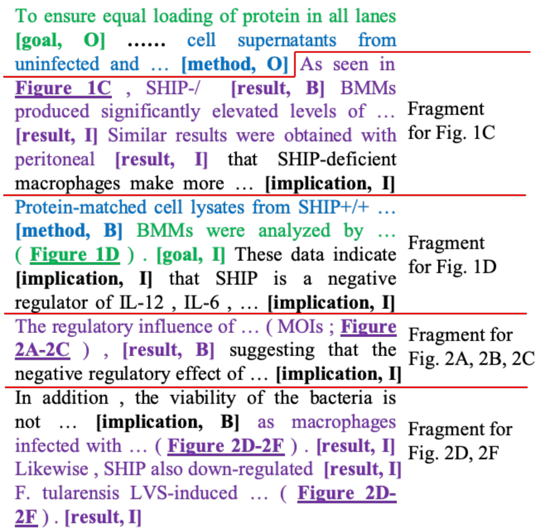
Claim Extraction.

Claim extraction has been extensively studied in various domains. In addition to scientific articles (Stab et al., 2014), previous work has analyzed social media (Dusmanu et al., 2017), news (Habernal et al., 2014; Sardianos et al., 2015) and Wikipedia (Thorne et al., 2018; Fréard et al., 2010) for a task called *Argumentation Mining* to extract claims and premises. However, there are less attention and dataset available in the biomedical domain. Achakulvisut et al. (2019) composed a claim-extraction dataset derived from MEDLINE [2] paper abstracts, and proposed a neural network model that significantly outperformed the rule-based method proposed by Sateli and Witte (2015). Figure 2 shows an example abstract with the last two sentences annotated as claims.



Panax notoginseng saponins (PNS) are components derived from Chinese herb panax notoginseng and play important roles in the cure of wounds. However, how PNS plays this function is still unclear. In this study, we used MTT assay, wound healing assay, western blot, quantitative real time PCR and enzyme-linked immunosorbent assay to detect the effects of PNS on the proliferation, migration and expression of collagen and fibronectin of anterior cruciate ligament (ACL) fibroblasts as well as the underlying mechanism. We found that PNS promoted the proliferation and migration of ACL fibroblasts and increased the expression levels of collagen and fibronectin. **Further mechanism study indicates that PNS might play its function through the phosphorylation of PI3K, AKT and ERK. This study provides a possible mechanism for the function of PNS and lays foundation for further study on the function of panax notoginseng.**

Figure 2: An example abstract with claim sentences highlighted in claim-extraction dataset (Achakulvisut et al., 2019).



To ensure equal loading of protein in all lanes **[goal, O]** ...... cell supernatants from uninfected and … **[method, O]**

As seen in **Figure 1C** , SHIP-/ **[result, B]** BMMs produced significantly elevated levels of … **[result, I]** Similar results were obtained with peritoneal **[result, I]** that SHIP-deficient macrophages make more … **[implication, I]**

Fragment for Fig. 1C

Protein-matched cell lysates from SHIP+/+ … **[method, B]** BMMs were analyzed by … (**Figure 1D**) . **[goal, I]** These data indicate **[implication, I]** that SHIP is a negative regulator of IL-12 , IL-6 , … **[implication, I]**

Fragment for Fig. 1D

The regulatory influence of … (MOIs ; **Figure 2A-2C**) , **[result, B]** suggesting that the negative regulatory effect of … **[implication, I]**

Fragment for Fig. 2A, 2B, 2C

In addition , the viability of the bacteria is not … **[implication, B]** as macrophages infected with … (**Figure 2D-2F**) . **[result, I]** Likewise , SHIP also down-regulated **[result, I]** F. tularensis LVS-induced … (**Figure 2D-2F**) . **[result, I]**

Fragment for Fig. 2D, 2F

Figure 3: An example paragraph of evidence fragment detection. The explicit mention of subfigure codes are underlined. The red lines indicate the borders of the evidence fragments. For each clause, the discourse type as well as the BIO tags indicating "blocks" (see Section 3.2.2) are provided.

In this work, we formulate claim extraction (Achakulvisut et al., 2019) similarly as scientific discourse tagging: **S** contains sentences and $L_i \in \{0, 1\}$ indicates whether the corresponding sentence is a claim or not.

Evidence Fragment detection. Burns et al. (2017) coined the concept of "evidence fragments" as the text section in narrative surrounding a figure reference that directly describes the experimental

---

[2] https://www.nlm.nih.gov/bsd/medline.html

figure. They composed an evidence fragment detection dataset, and proposed the evidence fragment detection task that tags each clause with semantically referred subfigure codes. They further proposed a rule-based method of using these subfigure codes as anchors to link evidence fragments to European Bioinformatics Institute's INTACT (Orchard et al., 2013) data records. As a result, INTACT's preexisting, manually-curated structured interaction data can serve as a gold standard for machine reading experiments.

Burns et al. (2017) formulated the problem into a clause-level tagging problem. Formally, each clause S_i in a paragraph $\mathbf{S} = [S_1, S_2, ..., S_n]$ is annotated with a set of subfigure codes $f^i = \{f_1^i, f_2^i, ..., f_m^i\}$ that each clause is semantically referring to, where the length m can be any non-negative integer. Figure 3 shows an illustration of a paragraph of evidence fragment detection annotation. Each clause in the paragraph is associated with a set of semantically relevant subfigures.

3 Approaches

3.1 Scientific Discourse Tagger

Model Overview. We formulate scientific tagging as a sentence level sequence tagging problem. We develop a deep structured model extending Dasigi et al. (2017), which consists of a contextualized word embedding layer, an attention layer that summarizes word embeddings into sentence embeddings, and a BiLSTM-CRF sequence tagger (Huang et al., 2015) on top of the sentence embeddings for discourse type tagging. Figure 4 gives an overview of the architecture. We detail each component in this section.

Embeddings. We explore pre-trained BioGloVe embedding (Burns et al., 2019), BioBERT (Lee et al., 2019) and SciBERT (Beltagy et al., 2019) embedding, which are GloVe and BERT embeddings trained on the text in biomedical domain.

Sentence Representations via Attention. We observe that only keywords are essential to determine the discourse types, and attention is an appropriate mechanism for emphasizing certain inputs and ignoring others. Dasigi et al. (2017) also explored using an attention mechanism to summarize word representations to sentence representations, however, we propose a new variation of attention mechanism using an LSTM. Specifically, we first encode the sentence using an LSTM to get contextualized hidden vectors of each word h_i, and use them to learn attentions by introducing another trainable vector s of the same dimension of h_i. We then apply the attention to summarize the word embeddings into a clause embedding. Detailed equations are provided in section A.1. The dashed circle in Figure 4 illustrates our LSTM-Attention based clause encoder.

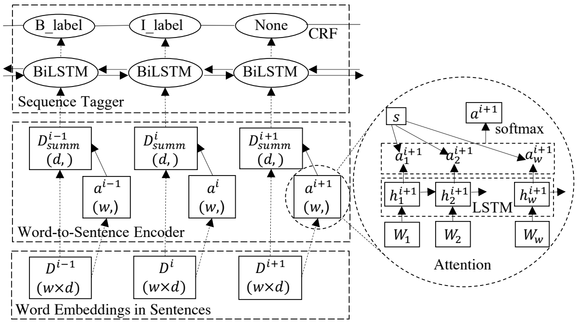


Figure 4: High-level overview of our scientific discourse tagger. Dashed arrows indicate we may apply dropout in those connections.

Sentence-level Sequence Tagging. We observe that the discourse labels have a clear transition of logic flow (e.g. *result* usually followed by *implication*, and *method* usually followed by *hypothesis*). Therefore, we extend LSTM sequence tagger used by Dasigi et al. (2017) to BiLSTM-CRF sequence tagger (Huang et al., 2015) to label discourse types for each sentence in a paragraph.

Labels in BIO Scheme. We use the BIO scheme (Sang and Veenstra, 1999) to train all of our models (Baseline models for SciDT dataset do not use BIO2 scheme). Specifically, we convert the labels into BIO scheme where *none* label represents *O* and all other labels are converted into *B_label* when the previous label type is different from the current label and *I_label* when the previous label is the same as the current label.

3.2 Downstream Applications

3.2.1 Claim Extractor

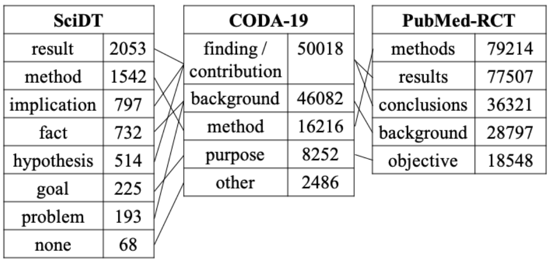
Due to the similar problem formulation of evidence extraction task (Achakulvisut et al., 2019), we directly employ the discourse tagging model for claim extraction.

3.2.2 Evidence Fragment Detector

Problem Reduction. As Figure 3 shows, since each clause in evidence fragment detection task may refer to more than one subfigure codes, we cannot directly solve it as a standard classification task. Instead, we reduce it to a clause-level sequence tagging problem under a block-based assumption. We treat each paragraph as a single input. During training, we encode the subfigure code reference sequences of the clauses in each paragraph into a single BIO (Sang and Veenstra, 1999) sequence (where *B* indicates the clause is the beginning of a block, *I* indicates the clause is in the same block as the previous clause, and *O* indicates that no subfigure code is being referred to) as demonstrated at the end of each clause in Figure 3. For prediction, we decode the semantic subfigure code references of all clauses from the BIO sequence for each paragraph following the same block-based assumption.

Block-Based Assumption. Most subfigure code reference labels are block-based. We call contiguous clauses that share the same subfigure code reference labels as a block, which is segmented by red lines in Figure 3. We further observe that most blocks explicitly mention all of the semantically referred subfigure codes at least once. Therefore, assuming this property is true for all blocks, we can reconstruct a sequence of semantic subfigure code references for all clauses in a paragraph. We use the explicitly mentioned subfigure codes for each block and a BIO sequence indicating where each block starts and ends for the reconstruction. Consequently, during encoding, we convert annotated semantically referenced subfigure code labels into BIO scheme. During decoding, we first localize the start and end position of each block using BIO predicted tags, then fill each block with all explicitly mentioned subfigure codes.

Clause-level Sequence Tagger. The key part of our sequence tagging-based solution for evidence fragment detection is to determine where a block starts and ends. We apply a clause-level sequence tagger to tag each clause in a paragraph. Due to the small size of the evidence fragment detection dataset, we empirically observe that feature-based CRF sequence taggers outperform neural-network based sequence taggers, we thus adopt the feature-based model. In addition to the scientific discourse tags, we use explicitly mentioned subfigure codes as well as unigram, bigram and trigram words as features. For each clause, we use all features described previously from the current clause in addition to the same sets of features from the adjacent previous and next clauses.



SciDT	
result	2053
method	1542
implication	797
fact	732
hypothesis	514
goal	225
problem	193
none	68

CODA-19	
finding / contribution	50018
background	46082
method	16216
purpose	8252
other	2486

PubMed-RCT	
methods	79214
results	77507
conclusions	36321
background	28797
objective	18548

Figure 5: Count of each label in three datasets. The lines correspond to the mappings from SciDT dataset (Burns et al., 2016; Dasigi et al., 2017) and PubMed 20k RCT dataset (Dernoncourt and Lee, 2017) to CODA-19 dataset (Huang et al., 2020) for zero-shot predictions (Section 5.2).

4 Experimental Setup

We evaluate the performance of our scientific discourse tagger on PubMed-RCT dataset (Dernoncourt and Lee, 2017) and SciDT dataset (Burns et al., 2016; Dasigi et al., 2017) (Section 5.1). We also examine the transferablity of our scientific discourse tagger to new datasets using CODA-19 dataset (Huang et al., 2020) (Section 5.2). We further study the efficiency of scientific discourse tags on claim-extraction task via transfer learning as well as evidence fragment detection task in a pipeline fashion (Section 6).

4.1 Datasets

Figure 5 shows the distribution of the labels in the three datasets introduced below as well as their mappings used for zero-shot predictions in Section 5.2.

PubMed-RCT Dataset. We use PubMed-RCT (Dernoncourt and Lee, 2017) as the standard dataset to evaluate our scientific discourse tagger against other strong baselines. PubMed-RCT is derived from PubMed for sequential sentence classification. It has two versions – a smaller PubMed 20k RCT, and a 10 times larger PubMed 200k RCT. Due to our limited availability of computational resources, we only consider PubMed 20k RCT in this work. PubMed 20k RCT is a large dataset that consists of 20k abstracts of randomized controlled trials (RCTs), with vocabulary of 68k across 240k sentences. Each sentence of an abstract is labeled with one of the following roles (section heads) in the abstract: *background*, *objective*, *method*, *result* or *conclusion*.

SciDT Dataset. Similar to PubMed-RCT (Dernoncourt and Lee, 2017), SciDT dataset (Burns et al., 2016; Dasigi et al., 2017) is a clause-based dataset with more fine-grained taxonomy. We further expand SciDT dataset by applying the same clause parsing and annotation pipeline described by Dasigi et al. (2017). This dataset is derived from the Pathway Logic (Eker et al., 2002) and INTACT databases (Orchard et al., 2013). Texts from all sections of each of those papers were pre-processed by parsing each sentence to generate a sequence of main and subordinate clauses using Stanford Parser (Socher et al., 2013). Domain experts were asked to label each of the clauses using the 7-label taxonomy proposed by De Waard and Maat (2012) whose distributions are shown in Figure 5. We apply sequential methods to sequences of clauses in individual paragraphs.

Overall, SciDT dataset has a total of 634 paragraphs and 6124 clauses. We randomly split 570 paragraphs as the training and validation set and the rest as the test set. Each paragraph contains up to 30 clauses and the number of word per clause has a mean of 17.7 and a standard deviation of 12.5. The total vocabulary size is 8563, which is a small dataset for an NLP task. However, we note the difficulties of obtaining such dataset. We further perform a quality assessment of the dataset by re-annotating the test set. We obtain Cohen's kappa coefficient $\kappa = 0.823$, which indicates a high quality of the dataset.

CODA-19 Dataset. CODA-19 (Huang et al., 2020) is a human-annotated dataset on a subset of the abstracts of CORD-19 (Wang et al., 2020), which is a corpus of scholarly articles about COVID-19. Wang et al. (2020) segmented each abstract into sentence fragments by comma (,), semicolon (;), and period (.). Each sentence fragment is labeled with one of the research aspects: *background*, *purpose*, *method*, *finding/contribution* or *other*, which is similar to the label sets of PubMed-RCT (Dernoncourt and Lee, 2017). There are 10966 abstracts in total. We use this dataset to further examine our scientific discourse tagger architecture's applicability to new datasets as well as the transferability of our trained scientific discourse tagger to new datasets.

4.2 Baseline Models

PubMed-RCT Dataset. We compare our discourse tagger against two strong baselines on the PubMed 20k RCT dataset: (1) a hierarchical sequential labeling network (HSLN) proposed by Jin and Szolovits (2018) and (2) the state-of-the-art model (Srivastava et al., 2019) on this dataset. HSLN (Jin and Szolovits, 2018) used bio-word2vec (Moen and Ananiadou, 2013), a word2vec embedding (Mikolov et al., 2013) trained on corpora of Wikipedia, PubMed, and PMC, a convolutional neural network (CNN) (LeCun et al., 2015) (HSLN-CNN) or a BiLSTM (HSLN-RNN) as a sentence encoder, followed by a BiLSTM-CRF architecture (Huang et al., 2015) as a sentence-level sequence tagger. Srivastava et al. (2019) used a similar architecture: bio-word2vec (Moen and Ananiadou, 2013) as word embedding, BiLSTM layer with a special dilation mechanism and a capsule layer (Hinton et al., 2011) as the sentence encoder and BiLSTM-CRF (Huang et al., 2015) as the sentence-level sequence tagger.

Model		**RCT**	**SciDT**
CRF			0.679
SVM			0.737
Dasigi et al. (2017)			0.791
HSLN-CNN		0.922	
HSLN-RNN		0.926	
Srivastava et al. (2019)		0.928	
Embedding	**Attention**		
BioGloVe	No Context	0.901	0.745
BioGloVe	RNN	0.909	0.763
BioGloVe	LSTM	0.913	0.794
BioBERT	No Context	0.909	0.794
BioBERT	RNN	0.915	0.775
BioBERT	LSTM	0.927	0.794
SciBERT	No Context	0.918	0.806
SciBERT	RNN	0.922	0.817
SciBERT	**LSTM**	**0.951**	**0.841**

Table 2: Scientific discourse tagging performance measured by test F1 score on PubMed 20k RCT and SciDT dataset.

SciDT Dataset. In addition to the model of Dasigi et al. (2017) trained on our expanded SciDT dataset, we also compare with feature based CRF and SVM with unigram, bigram and trigram words in the previous, current and next clauses as features.

CODA-19 Dataset. Huang et al. (2020) composed the CODA-19 dataset and studied a few baselines for scientific discourse tagging. Their best model is a fine-tuned SciBERT (Beltagy et al., 2019).

5 Experimental Results

5.1 Supervised Learning Results

Table 2 reports the test F1 score of our scientific discourse tagger and its variations against baseline models on PubMed 20k RCT dataset and SciDT dataset. Our best scientific discourse tagger outperforms the state-of-the-art model (Srivastava et al.,

2019) on PubMed 20k RCT dataset by more than 2 % absolute F1 score. Given the large size of PubMed 20k RCT, this result robustly demonstrates the strength of our model. Our model also significantly outperforms Dasigi et al. (2017) with 5% absolute F1 score (per McNemar's test, $p < 0.01$). Based on these performance, we claim our scientific discourse tagger as state-of-the-art. Note that for scientific discourse tagging, the micro F1 performance is equivalent to accuracy.

Ablation Studies. We also perform ablation studies to compare the effect of different word embeddings and attention mechanisms to the performance of our scientific discourse tagger on PubMed-RCT and SciDT dataset in Table 2. All neural network based models discussed for scientific discourse tagging tasks, including ours consist of a word embedding, a word-to-sentence encoder and a sentence-level sequence tagger. As we introduce in Section 3.1, our best model has SciBERT (Beltagy et al., 2019) as our contextualized word embedding, an LSTM-attention structure as our word-to-sentence encoder and BiLSTM-CRF (Huang et al., 2015) as our sentence sequence tagger. Comparing to other baseline models, we improve the model design by adopting the state-of-the-art BERT (Devlin et al., 2018) based language model as our contextualized embedding. Instead of bidirectional LSTM as word-to-sentence encoder used by Jin and Szolovits (2018) and Srivastava et al. (2019), we improve the attention structure proposed by Dasigi et al. (2017). We compare the effect of different embeddings and attention types used in scientific discourse tagger. As Table 2 indicates, our main improvement comes from SciBERT (Beltagy et al., 2019). In addition to BioBERT (Lee et al., 2019) which trains BERT (Devlin et al., 2018) on biomedical domain corpus, SciBERT uses a domain specific vocabulary. BERT as a contextualized embedding also contributes partially to the performance improvement as the BioBERT embedding globally outperforms BioGloVe (Burns et al., 2019), which is a static embedding trained on biomedical domain corpus, on PubMed-RCT dataset. Another source of improvement comes from the attention structure. Our LSTM-attention outperforms the RNN-attention that Dasigi et al. (2017) used.

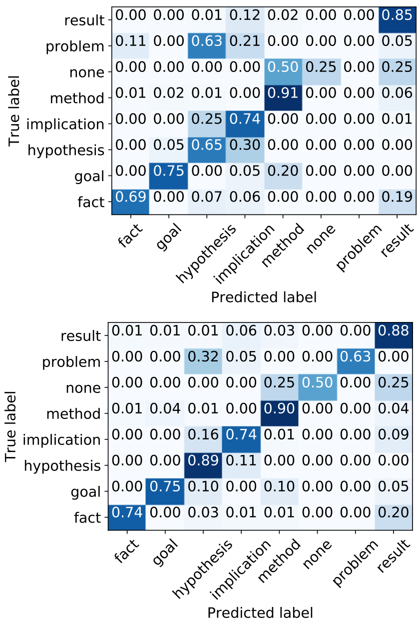


Figure 6: Confusion matrix on SciDT test data. Up: Dasigi et al. (2017). Down: Our scientific discourse tagger.

Error Analysis. Figure 6 compares the confusion matrices of Dasigi et al. (2017) and our best scientific discourse tagger on SciDT test set. As suggested by the overall performance, our model globally predicts the discourse tags more precisely than Dasigi et al. (2017). Specifically, Dasigi et al. (2017) failed to predict *problem* tag, but our model achieved 0.63 accuracy on predicting *problem* tag. Figure 6 also indicates the different difficulties of predicting different discourse labels due to the imbalance of the label distributions, as Table 5 shows.

5.2 Transfer Learning on CODA-19 Dataset

We further demonstrate the strong performance of our scientific discourse tagger by training it on CODA-19 dataset (Huang et al., 2020). As Table 3 shows, our model outperforms the baseline from Huang et al. (2020) by 14.6% absolute F1 on the test set.

More importantly, we use these results as baselines and CODA-19 dataset as an example dataset to show the transferability of our model to new datasets. We first perform zero-shot prediction using our best *trained* scientific discourse taggers on PubMed-RCT (Dernoncourt and Lee, 2017) or SciDT dataset (Dasigi et al., 2017). We map the labels from the original datasets to the target CODA-19 dataset by applying majority vote to the predicted labels on the training set as the lines in Figure 5 show. Then we perform predictions using the best trained scientific discourse taggers on CODA-19 test set and convert the predicted labels from the original label sets to the target CODA-

Model	Test F1
Huang et al. (2020)	0.749
Ours	**0.885**
Zero-shot Prediction from RCT	0.760
Zero-shot Prediction from SciDT	**0.761**
PubMed-RCT pre-train	**0.909**

Table 3: Transfer Learning Performance on CODA-19 Dataset.

Model	Test F1
Achakulvisut et al. (2019)	0.790
Ours (No pre-train)	0.791
Ours (PubMed-RCT pre-train)	**0.828**

Table 4: Claim extraction performance measured by binary F1 score, which regards 0 as negative label.

19 label set. As a result, as Table 3 shows, our zero-shot prediction results are even higher than the baseline from Huang et al. (2020) which was directly trained on the CODA-19 dataset. This result indicates the strong transferability of our trained scientific discourse tagger as a useful tool on new datasets.

Furthermore, we separately perform a standard transfer learning by taking the scientific discourse tagger pre-trained on PubMed-RCT dataset and fine-tuning it on CODA-19 dataset. We replace the last CRF layer with a new one to match the labels of CODA-19 dataset. As a result, we achieved 0.909 test F1, which is another 2.4% absolute F1 improvement on our model directly trained on CODA-19 dataset. This is likely due to the similar label structures between PubMed-RCT and CODA-19 dataset.

6 Downstream Applications

6.1 Claim Extraction

Dataset. Achakulvisut et al. (2019) introduced an expertly annotated dataset for extracting claim sentences from biomedical paper abstracts. They followed the definitions by Sateli and Witte (2015) to annotate a claim as a statement that either declares something is better, proposes something new, or describes a new finding or a causal relationship. Each sentence is tagged with a binary label indicating it is a claim or not. Each abstract may contain multiple claims as Figure 2 shows. The dataset contains 1500 abstracts sampled from MEDLINE database.

Baseline Model. Achakulvisut et al. (2019) constructed claim-extraction dataset and proposed a model using the sentence classification technique presented by Arora et al. (2016) as sentence encoding method, and the standard BiLSTM-CRF (Huang et al., 2015) as the sentence-level sequence tagger. Their best model was pre-trained on PubMed 200k RCT (Dernoncourt and Lee, 2017) for transfer learning and used GloVe (Pennington et al., 2014) as their word embedding.

Model Performance. Table 4 compares the test binary F1 performance of Achakulvisut et al. (2019) with our test performance. We first train our scientific discourse tagger model directly on the claim-extraction dataset. We obtain test binary F1 score of 0.791, which is already higher than Achakulvisut et al. (2019). Then as Achakulvisut et al. (2019) suggested, we pre-train the scientific discourse tagger on PubMed 20k RCT (Dernoncourt and Lee, 2017) and fine-tune it on the claim-extraction dataset. We replace the last CRF layer with a new one to match the binary label structure of claim-extraction dataset. As a result, we obtain test binary F1 score of 0.828, which is another 3.7% absolute F1 improvement on our model without transfer learning. This result demonstrates the benefit of transfer learning from scientific discourse tagging task to it's downstream-tasks.

6.2 Evidence Fragment Detection

Dataset. Burns et al. (2017) introduced evidence fragment detection dataset, which shares the same format and source of clause-based paragraphs with SciDT dataset (Dasigi et al., 2017). As Figure 3 shows, each clause was annotated with subfigure codes that it is semantically referring to. Each clause may not refer to any subfigure code, or simultaneously refer to multiple subfigure codes. The explicit mentions of the subfigure codes were also annotated. All paragraphs are from *Results* section of experimental papers, and most of the paragraphs are from a subset of SciDT dataset (Burns et al., 2016; Dasigi et al., 2017). We further expand evidence fragment detection training set by annotating extra *Results* section paragraphs from SciDT dataset. Overall this small dataset consists of 191 paragraphs as training data and 19 paragraphs as test data.

Baseline Model. Burns et al. (2017) proposed a rule-based method for evidence fragment detection task. The key steps are determining where each evidence fragment begins and ends based on the discourse tags of each clause. They treat *hypothe-*

Model	BIO F1	Test F1
Burns et al. (2017)	N/A	0.75
Ours (W/O Discourse Tags)	0.750	0.742
Ours (W/ Discourse Tags)	**0.821**	**0.807**

Table 5: Evidence fragment detection performance measured by micro F1 score. Our block-based decoding method achieves 0.94 F1 using ground truth BIO sequences.

sis, *problem* and *fact* as indicators of beginning of a evidence fragment, and *result* and *implication* as indicators of the end of a evidence fragment. They also used other features including section headings and whether the references to subfigures are entirely disjoint. Note that their document-level rule-based tagging is across multiple paragraphs in the *Results* section.

Model Performances. We use a feature-based CRF with block-based encoding-decoding method to solve this task as a sequence tagging problem. The decoding method described in Section 3.2.2 achieves 0.94 F1 score given the ground truth BIO sequences. Table 5 compares our feature-based CRF model performance with Burns et al. (2017) We also compare our feature-based CRF model performances trained with or without scientific discourse tags from SciDT dataset. Our feature-based CRF model without scientific discourse tags as inputs does not outperform Burns et al. (2017). However, by adding the scientific discourse tag as a feature, we obtain 5.7% absolute F1 improvement over Burns et al. (2017), reaching 0.807 test F1. This improvement is because of the improvement of the CRF sequence tagger. This result shows the strong benefit of scientific discourse tags as the upstream task of evidence fragment detection.

7 Discussion

We use the claim-extraction task and the evidence fragment detection task as two examples to demonstrate the benefit of leveraging pre-trained scientific discourse taggers and scientific discourse tags to improve the downstream-task performance via transfer learning or in a pipeline fashion. As Burns et al. (2017) proposed, given the output of evidence fragment detection system, we can link subfigure codes with INTACT (Orchard et al., 2013) records to obtain evidence fragments for each experimental figure.

We further suggest that the evidence fragment detection task can help biocurators delineate evidence fragments as independent documents so they can be cataloged, indexed, and reused. Traditionally scientists' arguments are based on relationships between claims and evidences within the same paper and possibly a limited number of cited papers. With the help of evidence fragments, we are able to discard the convention of only linking claims to evidence from a single paper or of following citations, which are often based on linking separate claims from different papers. As a future work, we can surface the evidence fragments combined with figures and captions across multiple papers. Clark et al. (2014) proposed the "Micropublications" semantic model, which is an abstract framework that integrates scientific argument and evidence from scientific documents. Our scientific discourse tagger, claim extractor and evidence fragment detector may serve as the actual implementation of the modules in such a framework. Ultimately, we hope to dramatically increase the amount of primary evidence used to generate individual claims and therefore improve the quality of those claims.

8 Conclusions

We develop a state-of-the-art model for scientific discourse tagging and demonstrate its strong performance on PubMed-RCT dataset (Dernoncourt and Lee, 2017) and SciDT dataset (Burns et al., 2016; Dasigi et al., 2017) as well as its strong transferability on new datasets such as CODA-19 dataset (Huang et al., 2020). We then demonstrate the benefit of leveraging the scientific discourse tags on downstream-tasks by providing claim-extraction task and evidence fragment detection task as two show cases. We further propose a future direction that scientific discourse tagging helps delineate evidence fragments as independent documents so they can be cataloged, indexed, and reused. As a result, we can dramatically increase the amount of primary evidence used to generate individual claims and therefore improve the quality of those claims.

Acknowledgement

We thank the anonymous reviewers for their useful comments, and the PlusLab members for their initial feedback. This work is supported by a National Institutes of Health (NIH) R01 grant (LM012592). The views and conclusions of this paper are those of the authors and do not reflect the official policy or position of NIH.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Titipat Achakulvisut, Chandra Bhagavatula, Daniel E. Acuña, and Konrad P. Körding. 2019. Claim extraction in biomedical publications using deep discourse model and transfer learning. *CoRR*, abs/1907.00962.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings.

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. *arXiv preprint arXiv:1903.10676*.

Gully A Burns, Pradeep Dasigi, and Eduard H Hovy. 2017. Extracting evidence fragments for distant supervision of molecular interactions. *BioRxiv*, page 192856.

Gully A Burns, Xiangci Li, and Nanyun Peng. 2019. Building deep learning models for evidence classification from the open access biomedical literature. *Database*, 2019.

Gully APC Burns, Pradeep Dasigi, Anita de Waard, and Eduard H Hovy. 2016. Automated detection of discourse segment and experimental types from the text of cancer pathway results sections. *Database*, 2016.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

François Chollet et al. 2015. Keras. `https://github.com/fchollet/keras`.

Tim Clark, Paolo N Ciccarese, and Carole A Goble. 2014. Micropublications: a semantic model for claims, evidence, arguments and annotations in biomedical communications. *Journal of biomedical semantics*, 5(1):1–33.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Jessica Cox, Corey A Harper, and Anita de Waard. 2017. Optimized machine learning methods predict discourse segment type in biological research articles. In *Semantics, Analytics, Visualization*, pages 95–109. Springer.

Pradeep Dasigi, Gully APC Burns, Eduard Hovy, and Anita de Waard. 2017. Experiment segmentation in scientific discourse as clause-level structured prediction using recurrent neural networks. *arXiv preprint arXiv:1702.05398*.

Anita De Waard and Henk Pander Maat. 2012. Epistemic modality and knowledge attribution in scientific discourse: A taxonomy of types and overview of features. In *Proceedings of the Workshop on Detecting Structure in Scholarly Discourse*, pages 47–55. Association for Computational Linguistics.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k RCT: a dataset for sequential sentence classification in medical abstracts. *CoRR*, abs/1710.06071.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Mihai Dusmanu, Elena Cabrio, and Serena Villata. 2017. Argument mining on twitter: Arguments, facts and sources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2317–2322.

S. Eker, M. Knapp, K. Laderoute, P. Lincoln, and C. Talcott. 2002. Pathway Logic: Executable Models of Biological Networks. In *Fourth International Workshop on Rewriting Logic and Its Applications (WRLA'2002*, volume 71 of *Electronic Notes in Theoretical Computer Science*. Elsevier.

Dominique Fréard, Alexandre Denis, Françoise Détienne, Michael Baker, Matthieu Quignard, and Flore Barcellini. 2010. The role of argumentation in online epistemic communities: the anatomy of a conflict in wikipedia. In *ECCE 2010. European Conference on Cognitive Ergonomics*, pages pp–91.

Yufan Guo, Anna Korhonen, Maria Liakata, Ilona Silins Karolinska, Lin Sun, and Ulla Stenius. 2010. Identifying the information structure of scientific abstracts: an investigation of three different schemes. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, pages 99–107. Association for Computational Linguistics.

Ivan Habernal, Judith Eckle-Kohler, and Iryna Gurevych. 2014. Argumentation mining on the web from information seeking perspective. In *ArgNLP*.

Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. 2011. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer.

Kenji Hirohata, Naoaki Okazaki, Sophia Ananiadou, and Mitsuru Ishizuka. 2008. Identifying sections in scientific abstracts using conditional random fields. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*.

Jerry R. Hobbs. 2002. Information extraction from biomedical text. *Journal of biomedical informatics*, 35(4):260–264.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ting-Hao'Kenneth' Huang, Chieh-Yang Huang, Chien-Kuang Cornelia Ding, Yen-Chia Hsu, and C Lee Giles. 2020. Coda-19: Reliably annotating research aspects on 10,000+ cord-19 abstracts using non-expert crowd. *arXiv preprint arXiv:2005.02367*.

Xiao Huang, Li Dong, Elizabeth Boschee, and Nanyun Peng. 2019. Learning a unified named entity tagger from multiple partially annotated corpora for efficient adaptation. In *The 2019 SIGNLL Conference on Computational Natural Language Learning (CoNLL)*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Di Jin and Peter Szolovits. 2018. Hierarchical neural networks for sequential sentence classification in medical scientific abstracts. *CoRR*, abs/1808.06161.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Maria Liakata. 2010. Zones of conceptualisation in scientific papers: a window to negative and speculative statements. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*, pages 1–4. Association for Computational Linguistics.

Maria Liakata, Shyamasree Saha, Simon Dobnik, Colin Batchelor, and Dietrich Rebholz-Schuhmann. 2012. Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, 28(7):991–1000.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing. *Proceedings of LBM*, pages 39–44.

Sandra Orchard, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H Campbell, Gayatri Chavali, Carol Chen, Noemi Del-Toro, et al. 2013. The mintact project—intact as a common curation platform for 11 molecular interaction databases. *Nucleic acids research*, 42(D1):D358–D363.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 548–554.

Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Nanyun Peng and Mark Dredze. 2017. Multi-task multi-domain representation learning for sequence tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Erik F Sang and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 173–179. Association for Computational Linguistics.

Christos Sardianos, Ioannis Manousos Katakis, Georgios Petasis, and Vangelis Karkaletsis. 2015. Argument extraction from news. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 56–66.

Bahar Sateli and René Witte. 2015. Semantic representation of scientific literature: bringing claims, contributions and named entities onto the linked open data cloud. *PeerJ Computer Science*, 1:e37.

Richard Socher, John Bauer, Christopher D Manning, et al. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 455–465.

Saurabh Srivastava, Puneet Agarwal, Gautam Shroff, and Lovekesh Vig. 2019. Hierarchical capsule based neural network architecture for sequence labeling. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Christian Stab, Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. 2014. Argumentation mining in persuasive essays and scientific articles from the discourse structure perspective. In *ArgNLP*, pages 21–25.

Simone Teufel and Marc Moens. 1999. Discourse-level argumentation in scientific articles: human and automatic annotation. *Towards Standards and Tools for Discourse Tagging*.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: experiments with relevance and rhetorical status. *Computational linguistics*, 28(4):409–445.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The fact extraction and verification (fever) shared task. *arXiv preprint arXiv:1811.10971*.

Dingquan Wang, Nanyun Peng, and Kevin Duh. 2017. A multi-task learning approach to adapting bilingual word embeddings for cross-lingual named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 383–388.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, et al. 2020. Cord-19: The covid-19 open research dataset. *arXiv preprint arXiv:2004.10706*.

A Appendices

A.1 Sentence Representations via Attention

Each of the word representations in the input tensor D is first reduced from d to d_2 dimensions. Then the word representations are projected from d_2 dimension to p dimension. For attention without context, p dimensional reduced word representations directly perform dot product with a p dimensional vector to obtain attention scores without using RNN. For attention with context, a simple RNN or LSTM with unit size h is used to compute attention scores. After obtaining the summarized matrix D_{summ}, we use bidirectional LSTM with hidden state of size H to tag the clauses.

LSTM-Attention We take the input tensors D of shape $c \times w \times d$ and output a matrix A of shape $c \times w$ which contains the attention weights of all the words in each clause or sentence. We first project each input word into a lower dimensional space using a projection matrix P of shape $d \times p$.

$$D_l = tanh(D \cdot P) \in \mathbb{R}^{c \times w \times d}$$

We score D_l with context that is summarized by an LSTM. Specifically, we score each word in the i^{th} clause in the context of other words in the same clause or sentence using an LSTM. The score for each word is a function of its p dimensional representation W_j and the previous words in the clause represented by the hidden states (h^i_{j-1}) in the LSTM cell. The equations are the following:

$$\begin{aligned} D^i_l &= D_l[i, :, :] \in \mathbb{R}^{w \times p} \\ W_j &= D^i_l[j, :] \in \mathbb{R}^p \\ h^i_j &= LSTM(W_j,\, h^i_{j-1}) \in \mathbb{R}^h \\ h^i &= [h^i_1\, h^i_2 \ldots h^i_w] \in \mathbb{R}^{w \times h} \\ a^i &= softmax(h^i \cdot s) \in \mathbb{R}^w \\ A &= [a^1\, a^2 \ldots a^i \ldots a^c] \in \mathbb{R}^{c \times w} \end{aligned}$$

where $LSTM$ is an LSTM cell with the unit size of h. s is a vector of length h.

Finally like Dasigi et al. (2017), a $c \times d$ shaped weighted sum D_{summ} of the input tensor D is computed, with the weights computed by the attention mechanism, then it is fed to a clause/sentence-level sequence tagger to tag discourse labels.

$$D_{summ}[i, :] = A[i, :] \cdot D[i, :, :] \in \mathbb{R}^d$$

Hyper-Parameter	Used
d_{BERT}	768
c	40
w	60
d	768
d_2	300
p	200
h	75
H	350
lr	10^{-3}
Validation Set Ratio	0.1
Embedding dropout	0.4
Dense dropout	0.4
Attention dropout	0.6
LSTM dropout	0.5
Batch size	10

Table 6: Optimal hyper-parameters of scientific discourse tagger model

A.2 Implementation and Training Details

The scientific discourse tagging model is implemented using Keras (Chollet et al., 2015) with Tensorflow (Abadi et al., 2015) backend. We use early stopping mechanism with toleration of 2 epochs. We schedule the training by training the model with a learning rate of lr for 20 epochs. We use Adam (Kingma and Ba, 2014) as our optimizer. The optimal hyper-parameters and the attempted range if applicable are listed in Table 6.

Incremental Beam Manipulation for Natural Language Generation

James Hargreaves
The Trade Desk & University of Cambridge
`james.hargreaves@thetradedesk.com`

Andreas Vlachos
University of Cambridge
`av308@cam.ac.uk`

Guy Emerson
University of Cambridge
`gete2@cam.ac.uk`

Abstract

The performance of natural language generation systems has improved substantially with modern neural networks. At test time they typically employ beam search to avoid locally optimal but globally suboptimal predictions. However, due to model errors, a larger beam size can lead to deteriorating performance according to the evaluation metric. For this reason, it is common to rerank the output of beam search, but this relies on beam search to produce a good set of hypotheses, which limits the potential gains. Other alternatives to beam search require changes to the training of the model, which restricts their applicability compared to beam search. This paper proposes incremental beam manipulation, i.e. reranking the hypotheses in the beam during decoding instead of only at the end. This way, hypotheses that are unlikely to lead to a good final output are discarded, and in their place hypotheses that would have been ignored will be considered instead. Applying incremental beam manipulation leads to an improvement of 1.93 and 5.82 BLEU points over vanilla beam search for the test sets of the E2E and WebNLG challenges respectively. The proposed method also outperformed a strong reranker by 1.04 BLEU points on the E2E challenge, while being on par with it on the WebNLG dataset.

1 Introduction

In natural language generation (NLG), the goal is to generate text representing structured information (e.g. a database record or a meaning representation) that is both fluent and contains the right information. Sequence-to-sequence models (seq2seq) have been effective on many tasks in NLG (for example: Wen et al., 2015; Dušek and Jurčíček, 2016). These systems first create an embedding for the input information. This embedding is used incrementally during decoding, generating one token at a time. Seq2seq models are generally decoded using beam search, to mitigate the effect of locally optimal but globally suboptimal decisions made by greedy search.

The performance of NLG systems can plateau or even decrease when beam sizes larger than 10 are used, which is counter-intuitive since larger beams produce more likely sequences according to the model. For example, Dušek and Jurčíček (2016) used a beam size of 10, and Asghar et al. (2017) found a size of 5 to be optimal. Decreasing performance has been found across a range of tasks including (Cohen and Beck, 2019). Moreover, it and was given by Koehn and Knowles (2017) as one of the six main challenges facing neural machine translation. To investigate this, Stahlberg and Byrne (2019) presented an exact search algorithm to find the most likely output according to a seq2seq model. However, this performed poorly compared to beam search, demonstrating that search errors (from beam search) can mask model errors (from the seq2seq model).

To mitigate the limitations of beam search, it is common practice to apply a reranker to the final set of hypotheses. This can be done by defining a reranking criterion (for example: Kumar and Byrne, 2004; Blain et al., 2017; Borgeaud and Emerson, 2020) or by training a reranker to predict the best hypothesis in a beam (for example: Dušek and Jurčíček, 2016; Agarwal et al., 2018). Training a reranker allows us to take into account information from outside the model and mitigate model errors. However, rerankers can only choose a hypothesis from the final beam, which limits their potential.

To quantify this, we trained the seq2seq model proposed by Dušek and Jurčíček (2016), and applied it to the E2E validation set (Novikova et al., 2017b). For each instance, we recorded the point at which

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2563–2574
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

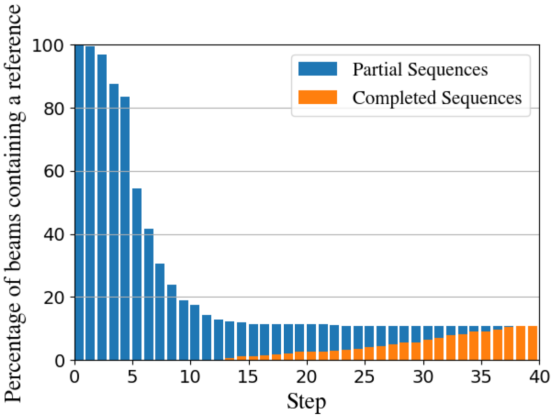


Figure 1: The percentage of beams which contain a reference (orange), or which could still lead to a reference (blue), using the model of Dušek and Jurčíček (2016) with beam size 3 on the E2E validation set.

all gold-standard references fell out of the beam, meaning that none of the partial hypotheses in the beam could be extended to a gold reference. A final beam containing at least one of the references would score optimally with an oracle reranker (providing an upper bound on performance). Figure 1 shows the results for beam size 3.[1] The final beam contained a reference in only 60 out of 547 cases (11%). For the remaining 89% of the cases, even an oracle reranker would be unable to give optimal results. The figure also shows that in over half of the cases, all references fell out in the first 6 steps. In contrast, references that were still in the beam at step 15 were almost certain to stay in the beam until the end. These observations suggest that an *early* manipulation of the beam has a strong potential to improve performance.

In this paper, we propose a method for manipulating which items are pruned from the beam at each stage of decoding. We then present evidence that this is a successful approach: it led to an improvement of 1.93, and 5.82 BLEU points over vanilla beam search on the E2E and WebNLG challenges, respectively. When comparing to a strong reranker, the performance of incremental beam manipulation was similar on the WebNLG dataset, whilst increasing the performance on the E2E challenge by 1.04 points. We also applied beam manipulation on top of length normalisation (Murray and Chiang, 2018), and incremental beam manipulation was able to improve its performance.

2 Related Work

This paper is far from the first to try to improve beam search for natural language generation.

One modification is to use a variable beam size instead of a fixed one (Freitag and Al-Onaizan, 2017). However, this can only improve decoding speed, as the ranking of the hypotheses in the beam remains unchanged, and thus model errors are exposed by the reduction of search errors.

Length normalisation (Murray and Chiang, 2018) is widely used strategy that often improves the performance of a beam search decoder, by mitigating the fact that seq2seq models are biased towards generating shorter sequences. Rather than directly using model probabilities to order the hypotheses in the beam, each probability is normalised according to the length of the hypothesis, so that shorter hypotheses are penalised. However, this only has an impact once the hypotheses within the beam have different lengths. This only occurs towards the end of the decoding process, and we showed in the previous section that the reference hypotheses often fall out of the beam relatively early. Furthermore, Stahlberg and Byrne (2019) showed that the biases causing the deteriorating model performance are more complex than a simple length bias.

Wiseman and Rush (2016) modified the training procedure for seq2seq models. They ran beam search and introduced a loss each time the gold standard sequence fell out of the beam. Goyal et al. (2018) and Collobert et al. (2019) also modified the training procedure. They added a term to the loss function that approximated the loss that the model would receive when generating using a beam search method for each example in the training set. However, one of the reasons that beam search has been so widely used is that it can be applied on top of a language model without changing the training procedure, and this is lost with these approaches.

Gu et al. (2017) manipulated the hidden state of the language model at each step of the decoding. This was achieved via a multi-output regressor that produced a vector that is added to the hidden state used in decoding. The regressor was trained via reinforcement learning, and the training signal was gathered by injecting unstructured noise to the hidden state. Chen et al. (2018) also manipulate the hidden state. For each training instance, they apply beam search and take the hypothesis with the

---

[1] For larger beam sizes, the same general trends were observed. See Appendix A for beam size 10.

highest BLEU score. The manipulator network is trained to encourage a greedy decoder to produce this output. Both of these approaches rely on inferring a better hidden state to be used in the decoding, which is not straightforward to define. We instead manipulate the hypotheses in the beam directly.

Finally, Negrinho et al. (2018) presented a framework for learning a beam search framework via imitation learning. This resulted in a beam aware algorithm which was proved to have no regret guarantees. While this paper makes a compelling argument for this method in theory, putting it into practice requires a number of further engineering decisions. Our work can be seen as a way of applying this general framework using a simple and computationally efficient roll-out strategy.

3 Incremental Beam Manipulation

In order to describe our method for incremental beam search, we first introduce terminology to describe a standard beam search decoder. The decoder produces a sequence iteratively, token by token. At each iteration it performs 3 actions: expand, rank and prune. The expand step generates all possible next step hypotheses. The rank step orders these hypotheses from most likely to least likely. The pruning step then removes the hypotheses that are near the end of this order.

This formulation of the beam search algorithm enables us to view beam manipulation as a ranking problem since expand is determined by the (fixed) decoder and the size of the beam chosen determines the pruning. The rank step determines which hypotheses will not be kept in the next iteration and hence discarded. By modifying the ranking method used, we can choose the partial hypotheses expanded during beam search, taking into account the current state of the beam as well as signals beyond model scores.

It is worth noting that while this paper applies beam manipulation on top of a seq2seq model, the techniques used could be applied without change to any conditional or unconditional neural language model that can be decoded using beam search.

3.1 Ranking via Roll-out

Partial hypotheses are more difficult to rank than complete hypotheses since the rest of the generated text is unknown. For example, consider the following partial hypotheses:

> Loch Fyne is a restaurant located...
>
> There is a family friendly...

Both of these convey some information about a family-friendly restaurant named ‘Loch Fyne’. It is hard to know which partial sequence will lead to a better complete sentence, which is what we would like a ranker to tell us. Existing rerankers often rely on detecting missing information, but some information may still be to come for partial hypotheses.

What we need is a way to rank partial hypotheses based on how the seq2seq model is likely to complete them. We propose ranking partial hypotheses based on a *greedy roll-out*. This is a computationally efficient approximation of how the seq2seq model might complete the partial hypothesis.

In the existing literature, roll-outs are generally used at training time (Chang et al., 2015), for the situation where the model's subsequent decisions influence the loss function for an individual decision. The roll-outs are used to produce an approximation to the final sequence that would be reached if the original action was taken. This enables a value for the loss of the original decision to be predicted.

On the other hand, incremental beam manipulation aims to predict which partial hypotheses will lead to good completed sequences. Similar to traditional roll-outs, this is impacted by the generating model's subsequent decisions. In this case, the difference is that roll-outs are used to provide features in addition to obtaining training signal. It is also worth noting that for incremental beam manipulation we use roll-outs at test time as well as at training time.

Beam manipulation can be applied after any step in the beam search decoding. Figure 2 illustrates a single manipulation. The roll-outs are used to produce approximations to the completed hypotheses that would be produced if the partial hypothesis remained in the beam. These completed sequences are then ranked to define an order of the partial sequences. Since this may result in different hypotheses remaining in the beam, the area of the search space considered during the decoding has been manipulated.

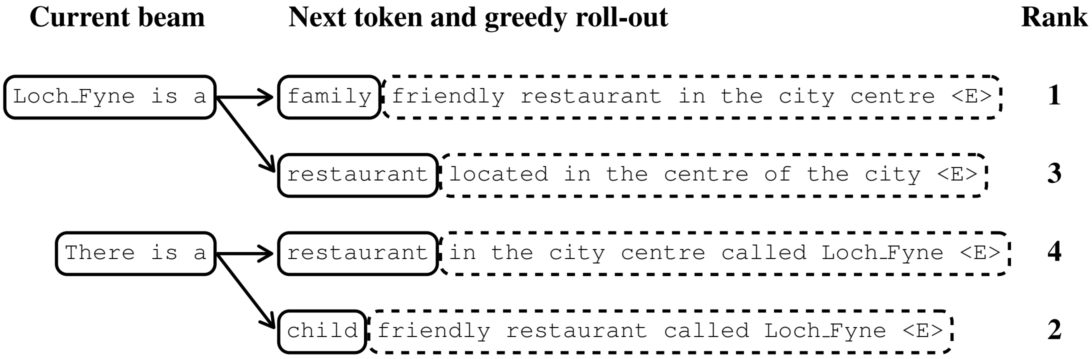


Figure 2: Incremental beam manipulation, for a beam size of 2, when generating the 4th token. Each element of the beam is expanded, and each partial hypothesis is greedily rolled out to a complete hypothesis. The complete hypotheses are ranked and then pruned. The partial hypotheses are then used in the next step of beam search.

3.2 Reranker Architecture

Incremental beam manipulation requires a method to rank completed hypotheses. There are many existing rerankers designed for this task, such as the TGEN reranker (Dušek and Jurčíček, 2016). However, these rerankers are unlikely to be effective when used in incremental beam manipulation. In the latter, the partial hypotheses need to be ranked according to their potential to produce good completed hypotheses. This is a related but different task to that of a traditional reranker that aims to identify the hypotheses that will score best against some metric such as BLEU score. Rerankers of completed hypotheses typically rely on input information missing from the output as the signal; however, this is not necessarily useful when reranking partial hypotheses. For example, it is more useful to identify partial hypotheses which are indicative of model failure at an early stage of decoding.

To rank the partial hypotheses via roll-outs as introduced in the previous section, we explored two commonly used techniques in the field of information retrieval: pointwise ranking and pairwise ranking (Liu, 2009). Pointwise approaches predict a numerical value for each item, and the items are ordered by sorting them according to this value. Pairwise approaches, given a pair of hypotheses, output which of them would rank more highly, and techniques such as the Copeland method (Saari and Merlin, 1996) are then used to produce a total ordering from these pairwise comparisons. In preliminary experiments, the pointwise approach outperformed the pairwise approach. These results are summarised in Appendix B. For the remainder of this paper, we will focus on the pointwise approach.

The inputs to the reranker were:

- The meaning representation (i.e. the structured information input for NLG). This was passed as a sequence.
- The generated text produced by the roll-out of the partial hypothesis. This was passed as a sequence surrounded by the start token `<S>` and end token `<E>`.
- The rank, according to the seq2seq model probability, of the completed sequence in the beam. Since this was a categorical variable, it was passed as a one-hot vector.

The architecture of the reranker is summarised in Figure 3. This model is used to assign a value to each hypothesis individually. The meaning representation and the rolled-out text are each passed through an LSTM,[2] and then all inputs are concatenated and passed through two fully connected layers.

At inference time the reranker is used to identify poorly performing hypotheses so that these can be pruned. This enabled the task of the reranker to be simplified to distinguishing between those hypotheses near the bottom of the beam from the hypotheses in the rest of the beam. Therefore, we used the reranker's scores to split the hypotheses into two groups: those with scores in the bottom quartile

---

[2] For efficient batching, it is common practice to add padding tokens to make sequences the same length. We prepended padding tokens rather than appending them, as this led to better performance in preliminary experiments. This is presumably because prepended tokens have a smaller impact on the final hidden state.

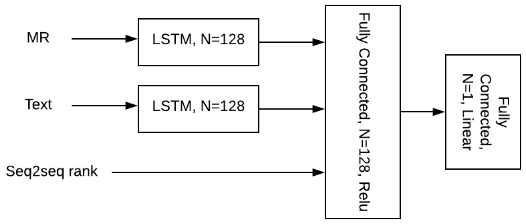


Figure 3: Architecture of the reranker. N is the number of hidden units for the LSTMs and the number of output nodes for the fully connected layers. For the fully connected layers, the activation function is shown.

and those with scores in the top three quartiles. The hypotheses within each group were ordered by the seq2seq model's probability.

In preliminary experiments, we also tried using the reranker to provide a total ordering. However, using it to provide a coarse partial ordering (as described above) gave more promising results.

3.3 Training the Reranker

We trained the reranker on completed hypotheses that were ranked from best to worst. The sequences were produced by generating text sequences using the seq2seq model. A beam search decoding, with a large beam size, was applied to each instance in the training set. [3] The set of hypotheses present in the final beam of the search was ranked from best to worst and recorded.

The notion of the best hypothesis was simplified to the one that received the highest BLEU score (Papineni et al., 2002) against the manually written references. BLEU was chosen due to its wide adoption as an automatic evaluation measure, but any automatic metric could have been used in its place.

As discussed at the end of the previous section, we need the reranker to distinguish between hypotheses that should be pruned from those that should be kept. Furthermore, for the purpose of reranking, only relative differences in BLEU score matter, not absolute values. Therefore, when generating training data, hypotheses in the bottom section of the beam (according to BLEU) were assigned the target value -1, and the rest were assigned the target value 1.

Similarly, it is only the differences in the reranker's scores that matter, and not the absolute values. Therefore, after applying the reranker to each hypothesis in a beam, we normalise the scores to have a mean of 0.

Using the normalised BLEU scores (-1 or 1) and normalised reranker scores (with a mean of 0), we use relative mean absolute error (RMAE) as the training objective, as shown in Equation (1), where b is the set of hypotheses in the beam, x is a hypothesis, $\hat{x}_{\text{ranker}}$ is the normalised score predicted by the reranker, and x_{BLEU} is the normalised target derived from the BLEU score ordering of the beam.

$$\text{RMAE}(b) = \sum_{x \in b} |\hat{x}_{\text{ranker}} - \hat{x}_{\text{BLEU}}| \qquad (1)$$

Several other relative loss functions have been shown to be successful in other situations (Zhang et al., 2019). In preliminary experiments, we evaluated a number of these including log cosh error and mean square error, but they did not outperform RMAE.

3.4 Choosing when to Manipulate

In theory, it would be desirable to manipulate the beam at every step of hypothesis generation, but in practice, the difficulty of ranking partial hypotheses could limit its benefits. While manipulating the beam can avoid certain model errors, it might also introduce other errors, either from the greedy roll-out strategy or the reranker. Reranking at every step may compound such errors. Empirically, we found it was more effective to apply beam manipulation to some rather than all steps.

Choosing when to manipulate is thus an important decision. It is advisable to avoid manipulating the beam too early: not only it is harder to rank hypotheses with very few tokens, but it is also less likely to be beneficial. As shown in Figure 1, in the first few steps even a relatively small beam size can keep hypotheses that could lead to the reference outputs. On the other hand, it is also advisable not to manipulate too late: once hypotheses have fallen out of the beam, they cannot be put back in. As the optimal choice of when to manipulate the beam is dependent on the dataset and the model,

---

[3] In preliminary experiments, we held back a portion of the training set from the seq2seq model's training phase, so that the beam manipulation ranker was trained on outputs that the seq2seq model had not seen. However, the system was unable to recover from the lower performance of the seq2seq model.

we treat this as a hyperparameter to be tuned on the validation set.

4 Experiments

In this section we will present results on the E2E (Novikova et al., 2017b) and WebNLG challenges (Gardent et al., 2017). We evaluate the systems using the BLEU implementation used in the original E2E challenge.[4] For all the experiments reported we use the seq2seq architecture from Dušek and Jurčíček (2016) for the underlying model that we are trying to manipulate.[5]

It is well-known that BLEU is not a completely reliable method for predicting human perceptions of the quality of individual NLG outputs (for example: Callison-Burch et al., 2006; Novikova et al., 2017a). However, in this case, we are comparing outputs from variants of the same system, and thus BLEU is more likely to provide reasonable estimates of their felicity to the references, as argued by both Callison-Burch et al. and Novikova et al..

To support the idea behind our approach, i.e. manipulating the beam during decoding instead of only at the end, we compare against existing rerankers applied to the beam at the end of decoding. These include the TGEN reranker, proposed by Dušek and Jurčíček (2016), that has achieved state-of-the-art BLEU scores on NLG tasks, as well as the same reranker architecture defined in Section 3.2. Both architectures are trained to perform reranking of the final beam only. When comparing between these two methods of reranking the final beam, no significant difference in performance was found. In this section, we report results for the architecture defined in Section 3.2. For completeness, results for both rerankers are included in Appendix C.

By using the same architecture both for the final-beam reranker and for beam manipulation, we can be more confident that any difference in results is due to the beam manipulation strategy (reranking during decoding, not just at the end).

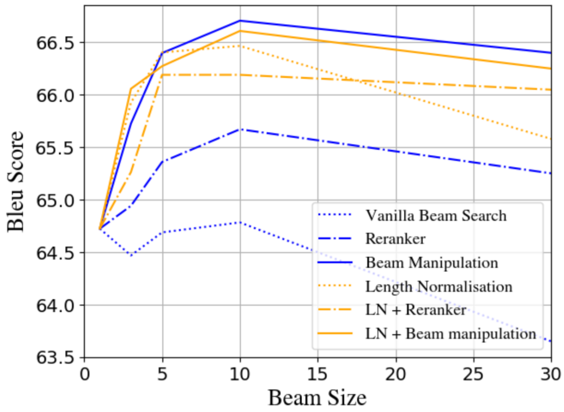


Figure 4: Results on the test set of the E2E challenge. LN = Length Normalisation.

In our experiments, we also consider length normalisation (Murray and Chiang, 2018), which is a well-known technique that often increases the BLEU score of the resulting sequences since it addresses the bias of language models to favour short sequences, as discussed in Section 2. Although the values assigned to a sequence when using length normalisation are no longer probabilities, it still performs the expand, rank and prune steps at each iteration of the decoding. Hence, we can apply beam manipulation in tandem with length normalisation. Finally, we also considered nucleus sampling (Holtzman et al., 2020) as a baseline. However, it was found to decrease performance even when compared to vanilla beam search.

In what follows, we will not only comment on test results when the beam size is tuned on the validation sets, but we will also comment on test results across all beam sizes. The reason for doing this is that considering all beam sizes assesses whether the technique is robust to changes in beam size. In our opinion, this makes for a more convincing result then just indicating a difference in performance at a single beam size. This is especially pertinent since a well-documented issue of beam search is that larger beam sizes can lead to deteriorating performance. The results table in Appendix C indicates the validation set's optimal beam size for each of the systems.

4.1 E2E BLEU Results

Figure 4 indicates the results on the E2E test set. The first thing to note is that increasing the beam size did not lead to any considerable gain in performance for the vanilla beam search strategy. For all beam sizes except 10, the performance is worse than greedy decoding, while using a beam size of 10 only increased performance by 0.06 points.

---

[4] `https://github.com/tuetschek/e2e-metrics`

[5] The source code for this paper is at `https://github.com/jamesHargreaves12/incremental_beam_manipulation`

Compared to vanilla beam search, reranking was an effective strategy, increasing the performance at all beam sizes. Similarly, applying incremental beam manipulation was able to outperform both methods at all beam sizes. Using the validation set to tune the beam size, the BLEU scores are 0.89 and 1.93 BLEU points higher for the reranker and incremental beam manipulation, respectively. The difference in BLEU scores between incremental beam manipulation and reranker methods was found to be significant (using a permutation test with significance level 0.01).

Length normalisation was the strongest baseline increasing the BLEU score of vanilla by 1.69 points. Adding the reranker on top of length normalisation decreases performance for all beam sizes less than 30. The strong performance of length normalisation is likely due to the fact that the E2E test set contained longer, and more complex inputs (and hence references) than the training and validation set (Dušek et al., 2020). Nevertheless, applying incremental beam manipulation on top of length normalisation was able to increase the BLEU score for all beam sizes except 5.

It is worth pointing out that while incremental beam manipulation improved both vanilla beam search and length normalisation, the overall BLEU score for the combination with the latter was lower for all sizes other than size 3. This is surprising considering that vanilla beam search performed worse than length normalisation when not combined with incremental beam manipulation. This could be due to the fact that the greedy roll-out approximation is less accurate for length normalisation than vanilla beam search since length normalisation only has an impact once some items in the beam have been completed.

4.2 WebNLG BLEU results

Figure 5 indicates the results on the WebNLG test set. As in the results for E2E, we can see that increasing the beam size of vanilla beam search was not an effective way to increase BLEU score. A greedy decode outperformed it at all beam sizes. Reranking the final beam was more effective, increasing the BLEU score by 5.83 points. Applying incremental beam manipulation had a very similar performance to reranking, increasing the performance at beam sizes 3 and 10 but reducing it at size 5.

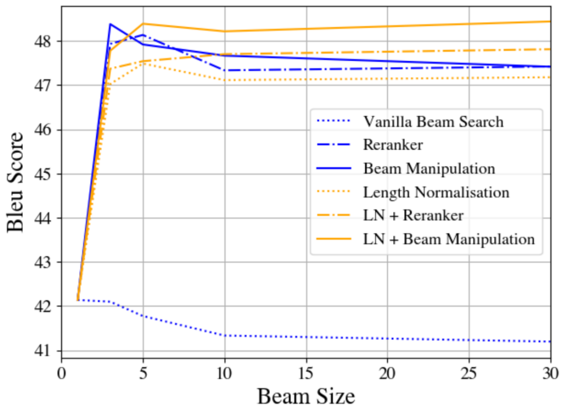


Figure 5: Results on the test set of the WebNLG challenge. LN = Length Normalisation.

The length normalisation baseline improved upon the vanilla baseline, increasing the BLEU score by 5.01 points. Reranking the final beam of the length normalised beam search was more effective on the WebNLG dataset than the E2E dataset; applying the reranker outperformed length normalisation at every beam size. Focusing on the beam sizes that performed optimally on the validation set, the BLEU score on the test set was increased by 0.43 points. Applying incremental beam manipulation on top of length normalisation received a yet higher BLEU score than reranked length normalisation for all beam sizes. Increasing the BLEU score by 1.33 points compared to the length normalisation. The improvement in BLEU scores achieved by applying incremental beam manipulation to the length normalised beam search was found to be significant when compared to length normalisation (with or without final beam reranking).

Unlike the E2E dataset, beam manipulation had higher performance when applied on top of length normalisation rather than vanilla beam search, outperforming it for all beam sizes except 3. The BLEU score was 0.52 points higher when taking the values at the beam sizes with the highest performances on the validation set.

4.3 Fallout with Beam Manipulation

In Section 1, we explained that references often fall out a beam relatively early during decoding, and reported results on the E2E task. We repeated the same experiment for when applying incremental beam manipulation. A beam size of 3 was used so that the results could be directly compared to those for vanilla beam search in Figure 1.

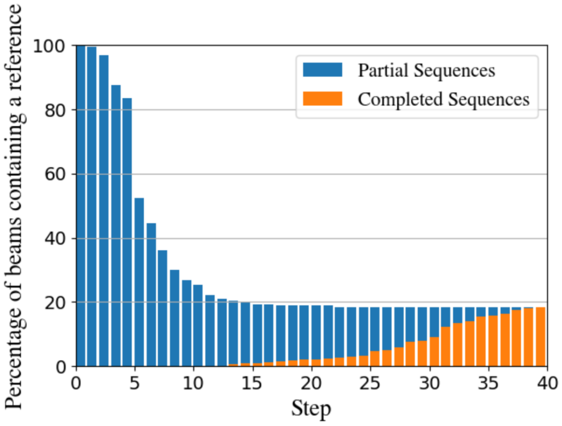


Figure 6: The percentage of beams which contain a reference (orange), or which could still lead to a reference (blue), using Incremental Beam Manipulation with beam size 3 on the E2E validation set. This improves on vanilla beam search, shown in Fig. 1.

The results are shown in Figure 6. The graph indicates that beam manipulation indeed ameliorates this issue. The final beam contains a (correct) reference in 100/547 cases (approx 18%), a large increase from the 60 of vanilla beam search. This is mainly due to reducing the number of references that fall out in steps 5 to 15, which is consistent with the fact that we are manipulating at steps 5, 10, 15 and 20. We also observed that most of the retention gain is due to earlier manipulation steps.

4.4 Human Evaluation

To further investigate the differences between the systems, we conducted a human evaluation comparing the incremental beam manipulation system's output against the output of the strongest baseline on the E2E dataset – length normalisation.

The human evaluation was performed by the second and third authors of the paper. While the annotators had been involved in the design of the system, they had not seen textual outputs from the system prior to the annotation. The outputs were presented in a random order, without indicating which output came from which system. The systems were compared in terms of both fluency and adequacy.

For fluency, each annotator compared the system outputs for the same meaning representation input (without seeing it) and indicated their preference. Both annotators annotated 50 examples from the E2E test set.

Little difference between the outputs was found. The systems were labelled as equally fluent in 76% of cases (incremental beam manipulation was preferred in 12% of all cases).

To judge the adequacy of the generations, human raters were presented with the meaning representation input and the text generated by a system. They were asked to label any hallucinations and any repetitions. They were asked to ignore missing information as the human references that the system had been trained on contained frequent examples of missing information (Dušek et al., 2020), so for this dataset, missing information is better seen as content selection. Between the annotators, a combined total of 524 examples were labelled for both for hallucination and repetition.

Once again, the results were not conclusive in support of either system, with no statistically significant difference between them. The overall performance was very high: for 95% of the inputs, both systems exhibited no signs of hallucination or repetition. This error analysis did, however, highlight that some errors are repeated multiple times, almost word for word. For example, all 5 cases of repetition for the incremental beam manipulation system had the following form: "There is a pub called X located near Y. It is a pub."

It is worth re-iterating that the system was optimised for BLEU, and not fluency or adequacy. The fact that an improvement in BLEU has not led to an improvement in a human evaluation suggests that BLEU may not be an accurate enough metric for this task, even when comparing similar systems. Therefore, BLEU may be even more limited in usefulness than Callison-Burch et al. (2006) and Novikova et al. (2017a) suggested.

4.5 Example outputs

We now present a couple of examples where manipulating the beam during decoding led to an improvement in the quality of the output. These were selected from the set of examples for which the output of the beam manipulator was preferred by the human annotators in terms of adequacy. The examples are given in Figure 7.

In the first example, we can see that length normalisation leads to a repetition of the fact that The Cricketers had an 'average customer rating', showing the downsides of a technique that just favours longer outputs. Neither the output from the beam

Input: name = The Cricketers | eat type = restaurant | food = English | price range = cheap | rating = average | area = city centre | family friendly = yes | near = Café Rouge
BM: The Cricketers serves cheap English food in the city centre near Café Rouge. It has an average customer rating and is family-friendly. **LN:** The Cricketers is a cheap, family-friendly, English restaurant with an average customer rating. It is located in the city centre near Café Rouge and has an average customer rating. **RR:** The Cricketers is a cheap, family-friendly restaurant located in city centre near Café Rouge.

Input: name = The Phoenix | eat type = pub | food = French | price range = £20-25 | rating = 3 out of 5 | area = riverside | family friendly = no | near = Café Sicilia
BM: The Phoenix is a French restaurant in riverside near Café Sicilia. It has a moderate price range and a customer rating of 3 out of 5. It is not kid friendly. **LN:** The Phoenix is a restaurant providing French food in the £20-25 price range. It is located in the riverside. It is near Café Sicilia. Its customer rating is 5 out of 5. **RR:** The Phoenix is a restaurant providing French food in the £20-25 price range. It is located in the riverside. It is near Café Sicilia. Its customer rating is high.

Figure 7: Example outputs for different systems. BM = Incremental beam manipulation system, LN = Vanilla beam search with length normalisation, RR = Vanilla beam search with reranking applied to the final beam.

manipulator nor the reranked approach contain repetitions, although we can see that more of the input information is realised in the case of the beam manipulator.

The second example contains a hallucination for both the length normalised and reranked systems – the input clearly states that the customer rating was '3 out of 5'. In contrast, whereas these systems claim that it was '5 out of 5' and 'high' respectively. The beam manipulation system avoided this issue.

5 Conclusions

Rerankers are commonly used to increase the performance of NLG systems decoded by beam search, by modifying which hypothesis from the final beam is chosen. This means that rerankers are dependent on good hypotheses reaching the final beam. However, this is often not the case; on the validation set of E2E challenge, only 11% of references were present in the final beam when the seq2seq model from (Dušek and Jurčíček, 2016) was decoded with a beam size of 3.

To address this limitation, we proposed incremental beam manipulation, which modifies the ranking of partial hypotheses within the beam at intermediate steps of the decoding, and hence chooses which are pruned. We evaluated this method on both the E2E and WebNLG challenges. The results showed that applying beam manipulation, instead of a reranker, was able to increase the BLEU score by 1.04 on the E2E challenge. We further showed that incremental beam manipulation was able to increase performance when applied on top of length normalisation.

The optimal reranker for incremental beam manipulation may differ at each step of generation (for example, token 5 vs. token 20). In future work, we intend to refine our method further by conditioning the reranker on how far through the beam search we are.

References

Shubham Agarwal, Marc Dymetman, and Eric Gaussier. 2018. Char2char generation with reranking for the E2E NLG challenge. pages 451–456.

Nabiha Asghar, Pascal Poupart, Xin Jiang, and Hang Li. 2017. Deep active learning for dialogue generation. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 78–83.

Frédéric Blain, Lucia Specia, and Pranava Madhyastha. 2017. Exploring hypotheses spaces in neural machine translation. In *Proceedings of the 16th Machine Translation Summit (MT Summit XVI)*. Asia-Pacific Association for Machine Translation (AAMT).

Sebastian Borgeaud and Guy Emerson. 2020. Leveraging sentence similarity in natural language generation: Improving beam search using range voting. In *Proceedings of the 4th Workshop on Neural Generation and Translation (WNGT)*, pages 97–109. Association for Computational Linguistics.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of BLEU in machine translation research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Proceedings*, pages 2058–2066. JMLR.org.

Yun Chen, Kyunghyun Cho, Samuel R Bowman, and Victor OK Li. 2018. Stable and effective trainable greedy decoding for sequence to sequence learning. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 1290–1299.

Ronan Collobert, Awni Hannun, and Gabriel Synnaeve. 2019. A fully differentiable beam search decoder. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 1341–1350.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51, Berlin, Germany. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG challenge. *Computer Speech & Language*, 59:123–156.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60, Vancouver. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Kartik Goyal, Graham Neubig, Chris Dyer, and Taylor Berg-Kirkpatrick. 2018. A continuous relaxation of beam search for end-to-end training of neural sequence models. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 3045–3052.

Jiatao Gu, Kyunghyun Cho, and Victor OK Li. 2017. Trainable greedy decoding for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1968–1978.

Ari Holtzman, Jan Buys, Leo Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223.

Renato Negrinho, Matthew R. Gormley, and Geoffrey J. Gordon. 2018. Learning beam search policies via imitation learning.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017a. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017b. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

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

Donald G. Saari and Vincent R. Merlin. 1996. The Copeland method. *Economic Theory*, 8(1):51–76.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3347–3353.

Tsung-Hsien Wen, Milica Gasic, Dongho Kim, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 275–284.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on*

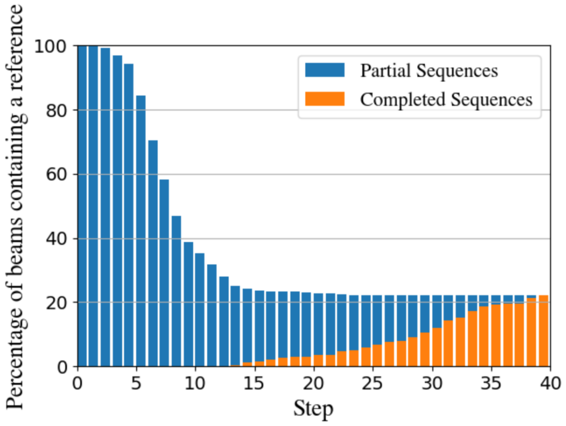


Figure 8: The percentage of beams that contain a reference sentence after each step of beam search. A beam size of 10 was used to decode the model proposed by Dušek and Jurčíček (2016). Results are for the E2E validation dataset. The orange bars indicate the number of completed references within the beam.

Empirical Methods in Natural Language Processing (EMNLP), pages 1296–1306.

Ning Zhang, Shui-Long Shen, Annan Zhou, and Ye-Shuang Xu. 2019. Investigation on performance of neural networks using quadratic relative error cost function. *IEEE Access*, 7:106642–106652.

A Fallout experiment with larger beam size

Figure 1 indicates the step at which the reference sentences drop out of the beam (for a beam size of 3). Figure 8 indicates the same results for a larger beam size of 10.

The figure indicates that the number of references that were contained in the final beam was higher for a beam size 10. For the early iterations of the decoding the number of references that fell out of the beam was far lower for a beam size of 10. A larger beam size meant that the beam contained more hypotheses and so has more chances to match against a reference.

However the shape of the graphs is very similar. The majority of references that fell out did so relatively early in the process. 54% of references fell out by step 7, increasing to 79% by step 9. At step 21 the final last reference fell out of the beam despite the fact that the beam contained partially references up to step 40.

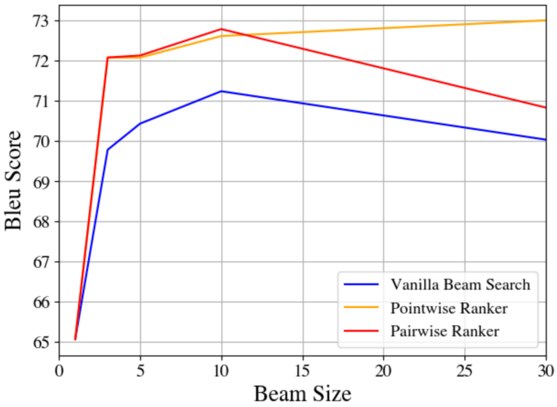


Figure 9: Comparison between the performance of the pointwise and pairwise rankers when used as rerankers on the E2E validation set.

B Pointwise vs Pairwise rerankers

This paper required a method of ranking completed hypotheses from worst to best. During preliminary experiments we implemented rerankers based on the Pairwise and Pointwise strategies from the Information Retrieval field. See Section 3.2 for more details.

To evaluate the performance of the different rankers we applied each of the rankers as a reranker of the final beam of a vanilla beam search over the E2E validation set. The BLEU scores for each of the rerankers were calculated for each beam size. The results are shown in Figure 9.

We can see that there was very little difference in performance for the two methods of reranking for the beam sizes up to 10. However, for beam size 30 the pointwise reranker significantly outperforms the pairwise reranker. The larger the beam size the greater the number of hypotheses that the reranker can pick as top and hence the greater the impact of the reranker.

The pointwise reranker requires O(k) runs of the reranker to produce a total ordering. On the other hand the Copeland method to produce a total ordering from the pairwise comparisons requires $O(k^2)$ number of pairwise Comparisons.

These factors lead us to choose the Pointwise ranker over the pairwise ranker for the experiments in the results section.

Beam size	Vanilla	Rerank	TGEN	LN	LN+Rerank	BM	LN+BM
1	64.72	64.72	64.72	64.72	64.72	64.72	64.72
3	64.47	64.94	65.33	65.93	65.26	65.73	**66.06**
5	64.69	65.36	65.47*	**66.40**	66.19*	**66.40**	66.27
10	64.78*	65.67*	65.58	66.47*	66.19	**66.71***	66.61
30	63.65	65.25	65.44	65.58	66.05	**66.40**	66.25*

Table 1: BLEU scores for each of the different systems on the E2E testset. *indicates the beam size which scored highest on the respective validation sets.**bold** indicates the highest scoring system for each beam size.

Beam size	Vanilla	Rerank	TGEN	LN	LN+Rerank	BM	LN+BM
1	42.14	42.14	42.14	42.14	42.14	42.14	42.14
3	42.10*	47.93*	47.28*	47.02	47.37	**48.38**	47.78
5	41.77	48.13	47.41	47.49	47.54*	47.92*	**48.39**
10	41.33	47.33	46.50	47.11*	47.70	47.66	**48.21**
30	41.20	47.42	46.61	47.18	47.81	47.41	**48.44***

Table 2: BLEU scores for each of the different systems on the WebNLG testset. *indicates the beam size which scored highest on the respective validation sets. **bold** indicates the highest scoring system for each beam size.

C Numerical results

This section will present the numerical results for the E2E and WebNLG datasets so that they can be more readily compared in future works. The results are given in Table 1 and Table 2.

D Hyperparameters

Throughout this paper a number of hyperparameters were introduced. The values used for each of the models in this paper are summarised bellow. Note that the search for these values was far from exhaustive so there is a good chance that the results of this paper could be improved upon through a better optimisation procedure.

In Section 3.3, the beam is split into two sections, bottom and rest. For all beam manipulation models the bottom of the beam was set to the bottom (i.e. lowest scoring) quarter of the beam. We also say a large beam is used to generate the data for training the beam. For all experiments in this paper we use a beam size of 50.

A key hyperparameter for performance of the incremental beam manipulation was the steps of the beam search at which the beam was manipulated. This hyperparameter varied for the 4 separate beam manipulation models. The values are summarised as follows:

- E2E, Incremental Beam Manipulation on top of vanilla beam search: 5, 10, 15, 20 and final.
- E2E, Incremental Beam Manipulation on top of length normalised beam search: 5, 7 and 10.
- WebNLG, Incremental Beam Manipulation on top of vanilla beam search: 4, 12 and final.
- WebNLG, Incremental Beam Manipulation on top of length normalised beam search: 5 and 12.

It is worth noting that manipulating the final step is the same as reranking the beam according to the ranker used in beam manipulation (i.e. no rollouts are performed).

StructSum: Summarization via Structured Representations

Vidhisha Balachandran **Artidoro Pagnoni** **Jay Yoon Lee**
Dheeraj Rajagopal **Jaime Carbonell** **Yulia Tsvetkov**
School of Computer Science
Carnegie Mellon University, Pittsburgh, USA
`{vbalacha,apagnoni,jaylee,dheeraj,jgc,ytsvetko}@cs.cmu.edu`

Abstract

Abstractive text summarization aims at compressing the information of a long source document into a rephrased, condensed summary. Despite advances in modeling techniques, abstractive summarization models still suffer from several key challenges: (i) *layout bias*: they overfit to the style of training corpora; (ii) *limited abstractiveness*: they are optimized to copying n-grams from the source rather than generating novel abstractive summaries; (iii) *lack of transparency*: they are not interpretable. In this work, we propose a framework based on document-level structure induction for summarization to address these challenges. To this end, we propose incorporating *latent and explicit dependencies across sentences in the source document* into end-to-end single-document summarization models. Our framework complements standard encoder-decoder summarization models by augmenting them with rich structure-aware document representations based on implicitly learned (latent) structures and externally-derived linguistic (explicit) structures. We show that our summarization framework, trained on the CNN/DM dataset, improves the coverage of content in the source documents, generates more abstractive summaries by generating more novel n-grams, and incorporates interpretable sentence-level structures, while performing on par with standard baselines.[1]

1 Introduction

Text summarization aims at identifying important information in long source documents and expressing it in human readable summaries. Two prominent methods of generating summaries are *extractive* (Dorr et al., 2003; Nallapati et al., 2017), where important sentences in the source article are selected to form a summary, and *abstractive* (Rush et al., 2015; See et al., 2017), where the model restructures and rephrases essential content into a paraphrased summary.

State of the art approaches to abstractive summarization employ neural encoder-decoder methods that encode the source document as a sequence of tokens producing latent document representations and decode the summary conditioned on the representations. Recent studies suggest that these models suffer from several key challenges. First, since standard training datasets are derived from news articles, model outputs are strongly affected by the layout bias of the articles, with models relying on the leading sentences of source documents (Kryscinski et al., 2019; Kedzie et al., 2018). Second, although they aim to generate paraphrased summaries, abstractive summarization systems often copy long sequences from the source, causing their outputs to resemble extractive summaries (Lin and Ng, 2019; Gehrmann et al., 2018). Finally, current methods do not lend themselves easily to interpretation via intermediate structures (Lin and Ng, 2019), which could be useful for identifying major bottlenecks in summarization models.

To address these challenges, we introduce *StructSum*: a framework that incorporates structured document representations into summarization models. StructSum complements a standard encoder-decoder architecture with two novel components: (1) a *latent-structure attention* module that adapts structured representations (Kim et al., 2017; Liu and Lapata, 2017) for the summarization task, and (2) an *explicit-structure attention* module that incorporates an external linguistic structure (e.g., coreference links). The two complementary components are incorporated and learned jointly with the encoder and decoder, as shown in Figure 1.

Encoders with induced latent structures have been shown to benefit several tasks including document classification, natural language inference

---

[1] Code and data available at: `https://github.com/vidhishanair/structured_summarizer`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2575–2585
April 19 - 23, 2021.

(Liu and Lapata, 2017; Cheng et al., 2016), and machine translation (Kim et al., 2017). Our latent structure attention module builds upon Liu and Lapata (2017) to model the dependencies between sentences in a document. It uses a variant of Kirchhoff's matrix-tree theorem (Tutte, 1984) to model such dependencies as non-projective tree structures(§2.2). The explicit attention module is linguistically-motivated and aims to incorporate inter-sentence links from externally annotated document structures. We incorporate a coreference based dependency graph across sentences, which is then combined with the output of the latent structure attention module to produce a hybrid structure-aware sentence representation (§2.3).

We test our framework using the CNN/DM dataset (Hermann et al., 2015) and show in §4.1 that it outperforms the base pointer-generator model (See et al., 2017) by up to 1.1 ROUGE-L. We find that the latent and explicit structures are complementary, both contributing to the final performance improvement. Our modules are also orthogonal to the choice of an underlying encoder-decoder architecture, rendering them flexible to be incorporated into other advanced models.

Quantitative and qualitative analyses of summaries generated by StructSum and baselines (§4), reveal that structure-aware summarization mitigates the news corpora layout bias by improving the coverage of source document sentences. Additionally, StructSum reduces the bias of copying large sequences from the source, inherently making the summaries more abstractive by generating $\sim$15% more novel n-grams than a competitive baseline. We also show examples of the learned interpretable sentence dependency structures, motivating further research for structure-aware modeling.

2 StructSum Framework

Consider a source document $\boldsymbol{x}$ consisting of n sentences $\{\boldsymbol{s}\}$ where each sentence $\boldsymbol{s}_i$ is composed of a sequence of words. Document summarization aims to map the source document to a target summary $\boldsymbol{y}$ of m words $\{y\}$. A typical neural abstractive summarization system is an attentional sequence-to-sequence model that encodes the input sequence $\boldsymbol{x}$ as a continuous sequence of tokens $\{w\}$ using a standard encoder (Hochreiter and Schmidhuber, 1997; Vaswani et al., 2017). The encoder produces a set of hidden representations $\{\mathbf{h}\}$. A decoder maps the previously generated token y_{t-1} to a hidden state and computes a soft attention probability distribution $p(\mathbf{a}_t \mid \boldsymbol{x}, \boldsymbol{y}_{1:t-1})$ over encoder hidden states. A distribution p over the vocabulary is computed at every time step t and the network is trained using the negative log likelihood loss: $\text{loss}_t = -\log p(y_t)$.

StructSum modifies the above architecture as follows. We aggregate the token representations from the encoder to form sentence representations as in hierarchical encoders (Yang et al., 2016). We then use implicit- and explicit-structure attention modules to augment the sentence representations with sentence dependency information, leveraging both a learned latent structure and an external structure from other NLP modules. The attended vectors are then passed to the decoder, which produces the output abstractive summary. In the rest of this section, we describe our framework architecture, shown in Figure 1, in detail.

2.1 Sentence Representations

We consider an encoder which takes a sequence of words in a sentence $\boldsymbol{s}_i = \{w\}$ as input and produces contextual hidden representation for each word $\mathbf{h}_{w_{ik}}$, where w_{ik} is the k^{th} word of the i^{th} sentence, $k = 1 : q$ and q is the number of words in the sentence $\boldsymbol{s}_i$. The word hidden representations are max-pooled at the sentence level and passed through a sentence-encoder, which produces new hidden sentence representations for each sentence $\mathbf{h}_{\boldsymbol{s}_i}$. The sentence hidden representations are then passed as inputs to the latent and explicit structure attention modules.

2.2 Latent Structure (LS) Attention

We model the latent structure of a source document as a non-projective dependency tree of sentences and force a pairwise attention module to automatically induce this tree. We denote the marginal probability of a dependency edge as $a_{ij} = p(z_{ij} = 1)$ where z_{ij} is the latent variable representing the edge from sentence i to sentence j. We parameterize the unnormalized pairwise scores between sentences with a neural network and use the Kirchoff's matrix tree theorem (Tutte, 1984) to compute the marginal probability of a dependency edge between any two sentences.

Specifically, we decompose the representation of a sentence $\boldsymbol{s}_i$ into a *semantic* vector $\mathbf{g}_{\boldsymbol{s}_i}$ and *structure* vector $\mathbf{d}_{\boldsymbol{s}_i}$ as $\mathbf{h}_{\boldsymbol{s}_i} = [\mathbf{g}_{\boldsymbol{s}_i}; \mathbf{d}_{\boldsymbol{s}_i}]$. Using the structure vectors $\mathbf{d}_{\boldsymbol{s}_i}, \mathbf{d}_{\boldsymbol{s}_j}$, we compute a score f_{ij} between sentence pairs (i, j) (where sentence i is

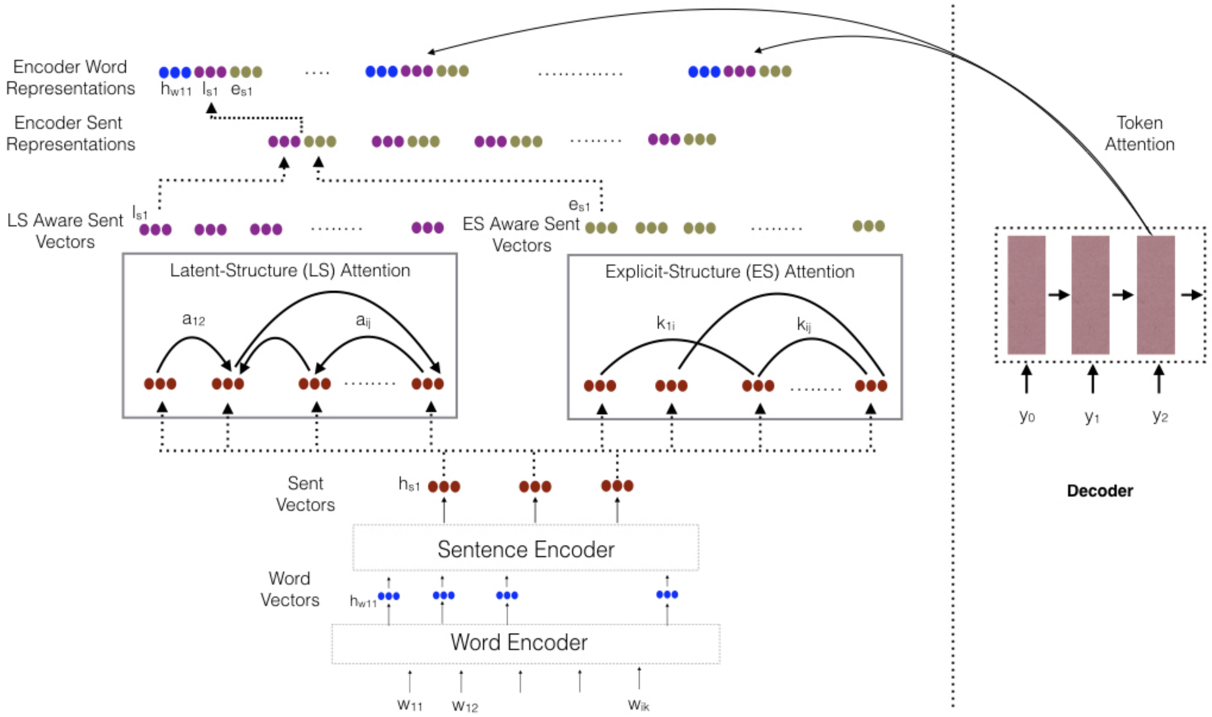


Figure 1: StructSum incorporates Latent Structure (LS) §2.2 and Explicit Structure (ES) §2.3 Attention to produce structure-aware representations. Here, StructSum augments the Pointer-Generator model, but the methodology that we proposed is general, and it can be applied to other encoder-decoder summarization systems

the parent node of sentence j) and a score r_i (where the sentence $\boldsymbol{s}_i$ is the root node):

$$f_{ij} = F_p(\mathbf{d}_{\boldsymbol{s}_i})^T W_a F_c(\mathbf{d}_{\boldsymbol{s}_j}) \text{ and } r_i = F_r(\mathbf{d}_{\boldsymbol{s}_i})$$

where F_p, F_c and F_r are linear-projection functions that build representations for the parent, child and root nodes respectively, and W_a is the weight for bilinear transformation. Here, f_{ij} is the edge weight between nodes (i, j) in a weighted adjacency graph $\mathbf{F}$ and is computed for all pairs of sentences. Using f_{ij} and r_i, we compute normalized attention scores a_{ij} and a_i^r using a variant of Kirchhoff's matrix-tree theorem where a_{ij} is the marginal probability of a dependency edge between sentences (i, j) and a_i^r is the probability of sentence i being the root.

Using these probabilistic attention weights and the semantic vectors $\{\mathbf{g}_{\boldsymbol{s}}\}$, we compute the attended sentence representations as:

$$\mathbf{p}_{\boldsymbol{s}_i} = \sum_{j=1}^{n} a_{ji}\mathbf{g}_{\boldsymbol{s}_j} + a_i^r \mathbf{g}_{root}$$

$$\mathbf{c}_{\boldsymbol{s}_i} = \sum_{j=1}^{n} a_{ij}\mathbf{g}_{\boldsymbol{s}_i}$$

$$\mathbf{l}_{\boldsymbol{s}_i} = \tanh(W_r[\mathbf{g}_{\boldsymbol{s}_i}, \mathbf{p}_{\boldsymbol{s}_i}, \mathbf{c}_{\boldsymbol{s}_i}])$$

where $\mathbf{p}_{\boldsymbol{s}_i}$ is the context vector gathered from possible parents of sentence i, $\mathbf{c}_{\boldsymbol{s}_i}$ is the context vector gathered from possible children, and $\mathbf{g}_{root}$ is a special embedding for the root node. Here, the updated sentence representation $l_{\boldsymbol{s}_i}$ incorporates the implicit structural information.

2.3 Explicit Structure (ES) Attention

Following Durrett et al. (2016), who showed that modeling coreference knowledge through anaphora constraints leads to improved clarity or grammaticality, we incorporate cross-sentence coreference links as the source of explicit structure. First, we use an off-the-shelf coreference parser[2] to identify coreferring mentions. We then build a coreference based sentence graph by adding a link between sentences $(\boldsymbol{s}_i, \boldsymbol{s}_j)$, if they have any coreferring mentions. This graph is converted into a weighted graph by incorporating a weight on the edge between two sentences that is proportional to the number of unique coreferring mentions between them. We normalize these edge weights for every sentence, effectively building a weighted adjacency matrix $\mathbf{K}$ where k_{ij} is given by:

---

[2] `https://github.com/huggingface/neuralcoref/`

$$k_{ij} = P(z_{ij} = 1) \tag{1}$$

$$= \frac{count(m_i \bigcap m_j) + \epsilon}{\sum_{v=1}^{n} count(m_i \bigcap m_v)} \tag{2}$$

where m_i denotes the set of unique mentions in sentence $\boldsymbol{s}_i$, $(m_i \bigcap m_j)$ denotes the set of co-referring mentions between the two sentences, and z is a latent variable representing a link in the coreference sentence graph. $\epsilon = 5e^{-4}$ is a smoothing hyperparameter.

Given contextual sentence representations $\{\mathbf{h}_{\boldsymbol{s}}\}$ and our explicit coreference-based weighted adjacency matrix $\mathbf{K}$, we learn an explicit structure-aware representation as follows:

$$\mathbf{u}_{\boldsymbol{s}_i} = \tanh(F_u(\mathbf{h}_{\boldsymbol{s}_i}))$$

$$\mathbf{t}_{\boldsymbol{s}_i} = \sum_{j=1}^{p} k_{ij}\mathbf{u}_{\boldsymbol{s}_j}$$

$$\mathbf{e}_{\boldsymbol{s}_i} = \tanh(F_e(\mathbf{t}_{\boldsymbol{s}_i}))$$

where F_u and F_e are linear projections and $\mathbf{e}_{\boldsymbol{s}_i}$ is an updated sentence representation which incorporates explicit structural information.

Finally, to combine the two structural representations, we concatenate the latent and explicit sentence vectors as: $\mathbf{h}_{\boldsymbol{s}_i} = [\mathbf{l}_{\boldsymbol{s}_i}; \mathbf{e}_{\boldsymbol{s}_i}]$ to form encoder sentence representations of the source document. To provide every token representation with the context of the entire document, the token representations are concatenated with their corresponding structure-aware sentence representation: $\mathbf{h}_{w_{ij}} = [\mathbf{h}_{w_{ij}}; \mathbf{h}_{\boldsymbol{s}_i}]$ where $\boldsymbol{s}_i$ is the sentence to which the word w_{ij} belongs. The resulting structure-aware token representations can be used to directly replace previous token representations as input to the decoder.

3 Experiments

Dataset: We evaluate our approach on the CNN/Daily Mail corpus[3] (Hermann et al., 2015; Nallapati et al., 2016) and use the same preprocessing steps as shown in See et al. (2017). The CNN/DM has 287226/13368/11490 train/val/test samples respectively. The reference summaries have an average of 66 tokens ($\sigma = 26$) and 4.9 sentences. Differing from See et al. (2017), we truncate source documents to 700 tokens instead of 400 in training and validation sets to model longer documents with more sentences. All our experiments were trained on Nvidia GTX Titan X GPUs.

Base Model: Although StructSum framework can be incorporated in any encoder-decoder framework with structure-aware representations, for our experiments we chose the pointer-generator model (See et al., 2017) as the base model, due to its simplicity and ubiquitous usage as a neural abstractive summarization model across different domains (Liu et al., 2019; Krishna et al., 2020). The word and sentence encoders are BiLSTM and the decoder is a BiLSTM with a pointer based copy mechanism. We re-implement the base pointer-generator model and augment it with the StructSum modules described in §2 and hence our model can be directly compared to it.

Baselines: In addition to the base model, we compare StructSum with the following baselines:
Tan et al. (2017): This is a graph-based attention model that is closest in spirit to the method we present in this work. A graph attention module is used to learn attention between sentences, but it cannot be easily used to induce interpretable document structures, since its attention scores are not constrained to learn structure. On top of latent and interpretable structured attention between sentences, StructSum introduces an explicit structure component to inject external document structure, which distinguishes it from Tan et al. (2017).
Gehrmann et al. (2018): This work introduces a separate content selector which tags words and phrases to be copied. The DiffMask variant is an end-to-end variant like ours and hence is included in our baselines. We compare StructSum with the DiffMask experiment.[4]

Hyperparameters: Our encoder uses 256 hidden states for both directions in the one-layer BiLSTM, and 512 for the single-layer decoder. We use the Adagrad optimizer (Duchi et al., 2011) with a learning rate of 0.15 and an initial accumulator value of 0.1. We do not use dropout and use

---

[3] `https://cs.nyu.edu/~kcho/DMQA/`

[4] The best results from Gehrmann et al. (2018) outperform DiffMask experiment, but they use inference-time hard masking which can be applied on ours. Our baselines also exclude Reinforcement Learning (RL) based systems as they are not directly comparable, but our approach can be introduced in an encoder-decoder based RL system. Since we do not incorporate any pretraining, we do not compare with recent contextual representation based models (Liu and Lapata, 2019).

[5] `https://github.com/atulkum/pointer_summarizer`

Model	ROUGE 1	ROUGE 2	ROUGE L
Pointer-Generator (See et al., 2017)	36.44	15.66	33.42
Pointer-Generator + Coverage (See et al., 2017)	39.53	**17.28**	36.38
Graph Attention (Tan et al., 2017)	38.10	13.90	34.00
Pointer-Generator + DiffMask (Gehrmann et al., 2018)	38.45	16.88	35.81
Pointer-Generator (Re-Implementation)	35.55	15.29	32.05
Pointer-Generator + Coverage (Re-Implementation)	39.07	16.97	35.87
Latent-Structure (LS) Attention	39.52	16.94	36.71
Explicit-Structure (ES) Attention	**39.63**	16.98	36.72
LS + ES Attention	39.62	17.00	**36.95**

Table 1: Evaluation of summarization models on the CNN/DM dataset. Published abstractive summarization baseline scores are on top. The bottom section shows re-implementations of See et al. (2017)[5] and StructSum results that incorporate latent and explicit document structure into the base models. StructSum's utility is on par with the base models, while introducing additional benefits of better abstractiveness and intrepretability shown in §4.

gradient-clipping with a maximum norm of 2. We selected the best model using early stopping based on the ROUGE score on the validation dataset as our criteria. We also used the coverage penalty during inference as shown in Gehrmann et al. (2018). For decoding, we use beam-search with a beam width of 3. We did not observe significant improvements with higher beam widths.

4 Evaluation

A standard ROUGE metric does not shed meaningful light into the quality of summaries across important dimensions. As a recall-based metric it is not suitable for assessing the abstractiveness of summarization; it is also agnostic to layout biases and does not facilitate intrepretability of model decisions. We thus adopt automatic metrics tailored to evaluating separately each of these aspects. We compare StructSum to our base model, the pointer-generator network with coverage (See et al., 2017) and the reference.

4.1 Automatic Metrics

We first conduct a standard comparison of generated summaries with reference summaries using ROUGE-1,2 and L (Lin, 2004) F1[6] metric. Table 1 shows the results. We first observe that introducing the latent structures and explicit structures independently improves our performance on ROUGE-L. It suggests that modeling dependencies between sentences helps the model compose better long sequences compared to baselines. We see small improvements in ROUGE-1 and ROUGE-2, hinting that we retrieve similar content words as the baseline but compose them into better contiguous sequences. As both ES and LS independently get similar performance, the results show that LS attention induces good latent dependencies that make up for pure external coreference knowledge.

Finally, our combined model which uses both Latent and Explicit structure performs the best with an improvement of **1.08 points** in ROUGE-L and **0.6 points** in ROUGE-1 over base pointer-generator model (statistically significant for 11490 samples at p=0.05 using Wilson Confidence Test). It shows that the latent and explicit information are complementary and a model can jointly leverage them to produce better summaries. Additionally, we find that structural inductive bias helps a model to converge faster. The combined LS+ES Attention model converges in 126K iterations in comparison to ∼230K iterations required for the pointer-generator network.

While ROUGE is a popular metric used for evaluating summarization models, it is limited to only evaluating n-gram overlap while ignoring semantic correctness. Hence, we compared our method with the baseline Pointer-Generator model using the BERTScore metric (Zhang et al., 2020). We observe that our model improves BERTScore by 9 points (12.3 for Pointer-Generator v/s 21.7 for StructSum) showing that our model is able to generate semantically correct content.

4.2 Abstractiveness

Despite being an abstractive model, the pointer-generator model tends to copy very long sequences of words including whole sentences from the

---

[6] `https://pypi.org/project/pyrouge/`

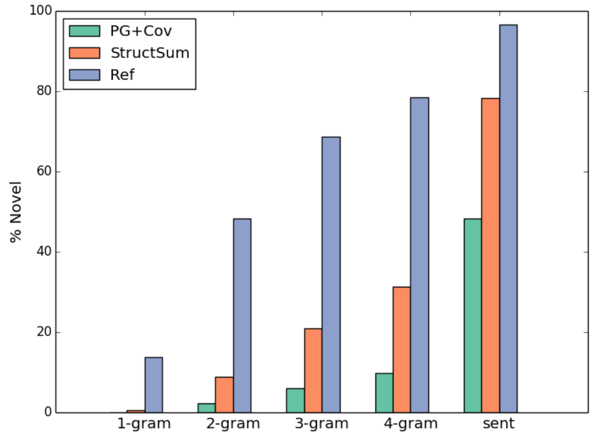


Figure 2: Comparison of % Novel n-grams between StructSum, Pointer-Generator+Coverage and the Reference. Here, "sent" indicates full novel sentences.

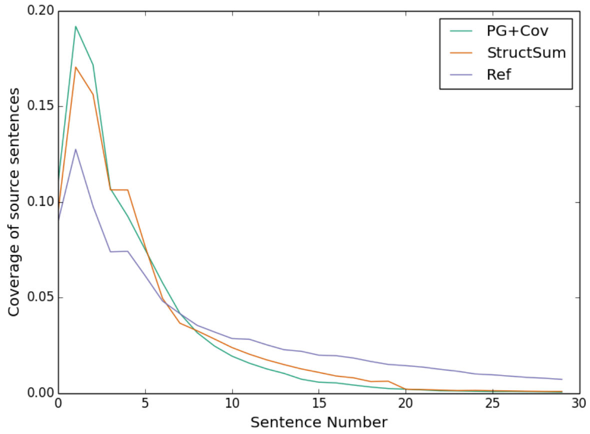


Figure 3: Coverage of source sentences in summary. Here the x-axis is the sentence position in the source article and y-axis shows the normalized count of sentences in that position copied to the summary.

source document (also observed by Gehrmann et al. (2018)). We use two metrics to evaluate the abstractiveness of the model:

Copy Length: Table 2 shows a comparison of the average length (Copy Len) of contiguous copied sequences from the source document (greater than length 3). We observe that the pointer-generator baseline on average copies 16.61 continuous tokens from the source which shows the extractive nature of the model. This indicates that pointer networks, aimed at combining advantages from abstractive and extractive methods by allowing to copy content from the input document, tend to skew towards copying, particularly in this dataset. A consequence of this is that the model fails to interrupt copying at desirable sequence length. In contrast, modeling document structure through StructSum reduces the length of copied sequences to 9.13 words on average reducing the bias of copying sentences entirely. This average is closer to the reference (5.07 words) in comparison, without sacrificing task performance. StructSum learns to stop when needed, while still generating coherent summaries.

Novel N-Grams: The proportion of novel n-grams generated has been used in the literature to measure the degree of abstractiveness of summarization models (See et al., 2017). Figure 2 compares the percentage of novel n-grams in StructSum as compared to the baseline model. Our model produces novel trigrams 21.0% of the time and copies whole sentences only 21.7% of the time. In comparison, the pointer-generator network has only 6.1% novel trigrams and copies entire sentences 51.7% of the time. This shows that StructSum on average generates 14.7% more novel n-grams in comparison to the pointer-generator baseline.

4.3 Coverage

A direct outcome of copying shorter sequences is being able to cover more content from the source document within given length constraints. We observe that this leads to better summarization performance. We compute coverage by computing the number of source sentences from which contiguous sequences greater than length 3 are copied in the summary. Table 2 shows a comparison of the coverage of source sentences in the summary content. While the baseline pointer-generator model only copies from 12.1% of the source sentences, StructSum copies content from 24.0% of the source sentences. Additionally, the average length of the summaries produced by StructSum remains mostly unchanged at 66 words on average compared to 61 of the baseline model. This indicates that StructSum produces summaries that draw from a wider selection of sentences from the original article compared to the baseline models.

4.4 Layout Bias

Neural abstractive summarization methods applied to news articles are typically biased towards selecting and generating summaries based on the first few sentences of the articles. This stems from the structure of news articles, which present the salient information of the article in the first few sentences

	Copy Len	Coverage
PG+Cov	16.61	12.1 %
StructSum	9.13	24.0 %
Reference	5.07	16.7 %

Table 2: Results of analysis of copying and coverage distribution over the source sentences on CNN/DM test set. Copy Len denotes the average length of copied sequences; Coverage – coverage of source sentences.

	Coref	NER	Coref+NER
precision	0.29	0.19	0.33
recall	0.11	0.08	0.09

Table 3: Precision and recall of ES and LS shared edges

and expand in the subsequent ones. As a result, the LEAD 3 baseline, which selects the top three sentences of an article, is widely used in the literature as a strong baseline to evaluate summarization models applied to the news domain (Narayan et al., 2018). Kryscinski et al. (2019) observed that the current summarization models learn to exploit the layout biases of current datasets and offer limited diversity in their outputs.

To analyze whether StructSum also holds the same layout biases, we compute a distribution of source sentence indices that are used for copying content (copied sequences of length 3 or more are considered). Figure 3 shows the distributions of source sentences covered in the summaries. The coverage of sentences in the reference summaries shows a high proportion of the top 5 sentences of any article being copied to the summary. Additionally, the reference summaries have a smoother tail end distribution with relevant sentences in all positions being copied. It shows that a smooth distribution over all sentences is a desirable feature. We notice that the pointer-generator framework have a stronger bias towards the beginning of the article with a high concentration of copied sentences within the top 5 sentences of the article. In contrast, StructSum improves coverage slightly having a lower concentration of top 5 sentences and copies more tail end sentences than the baselines. However, although the modeling of structure does help, our model has a reasonable gap compared to the reference distribution. We see this as an area of improvement and a direction for future work.

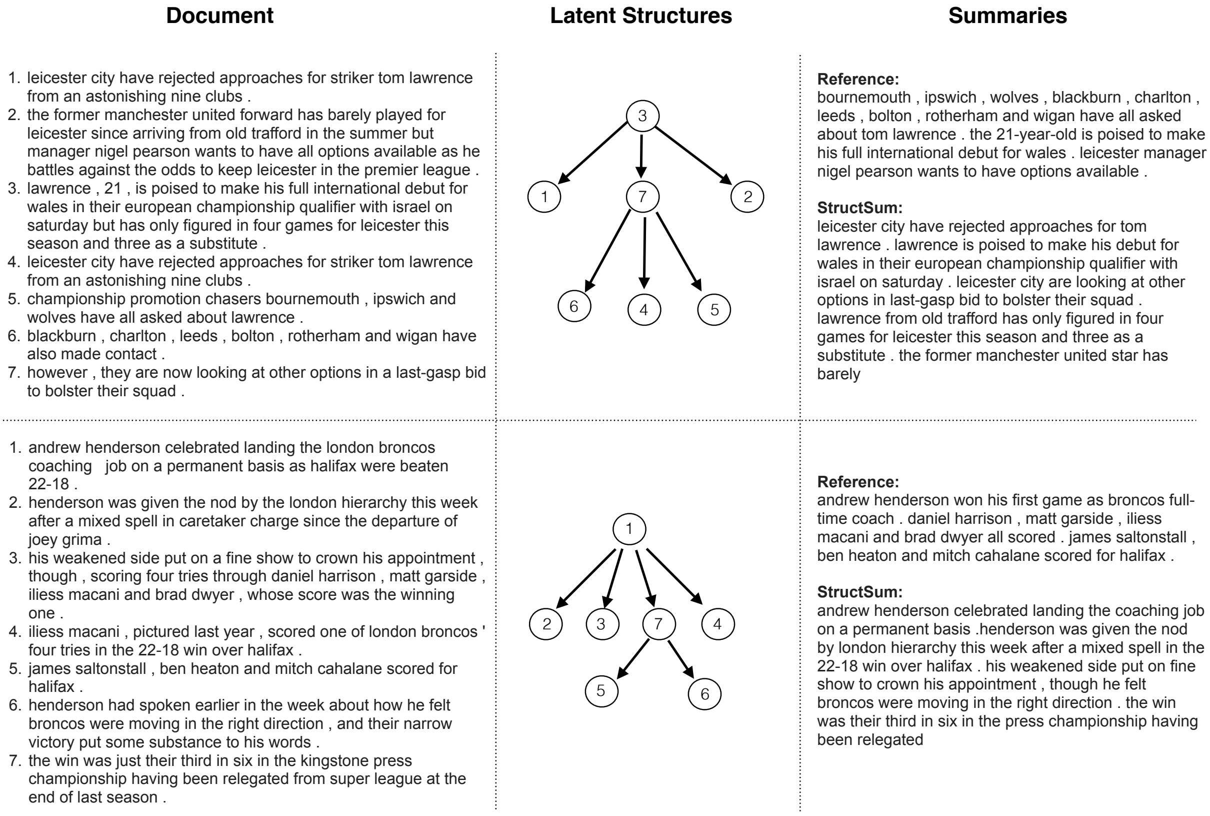
Depth	2	3	4	5+
StructSum	29.3%	53.7%	14.4%	2.6%

Table 4: Distribution of latent tree depth.

5 Analysis of Induced Document Structures

Similar to Liu and Lapata (2017), we also look at the quality of the intermediate structures learned by the model. We use the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) to extract the maximum spanning tree from the attention score matrix as our sentence structure. Table 4 shows the frequency of various tree depths. We find that the average tree depth is 2.9 and the average proportion of leaf nodes is 88%, consistent with results from tree induction in document classification (Ferracane et al., 2019). Further, we compare latent trees extracted from StructSum with undirected graphs based on coreference, on NER, or on both. These are constructed similarly to our explicit coreference based sentence graphs in §2.3 by linking sentences with overlapping coreference mentions or named entities. We measure the similarity between the learned latent trees and the explicit graphs through precision and recall over edges. The results are shown in Table 3. We observe that our latent graphs have low recall with the linguistic graphs showing that our latent graphs do not capture the coreference or named entity overlaps explicitly, suggesting that the latent and explicit structures capture complementary information.

Figure 4 shows qualitative examples of induced structures along with summaries from the StructSum. The first example shows a tree with sentence 3 chosen as root, which was the key sentence mentioned in the reference. In both examples, the sentences in the lower level of the dependency tree contribute less to the generated summary. Similarly, in the examples source sentences used to generate summaries tend to be closer to the root node. In the first summary, all source content sentences used in the summary are either the root node or within depth 1 of the root node. In the second example, 4 out of 5 source sentences were at depth=1 in the tree. In both examples, generated summaries diverged from the reference by omitting certain sentences used in the reference. These sentences are in the lower section of the tree, providing insights on which sentences were preferred for the



1. leicester city have rejected approaches for striker tom lawrence from an astonishing nine clubs .
2. the former manchester united forward has barely played for leicester since arriving from old trafford in the summer but manager nigel pearson wants to have all options available as he battles against the odds to keep leicester in the premier league .
3. lawrence , 21 , is poised to make his full international debut for wales in their european championship qualifier with israel on saturday but has only figured in four games for leicester this season and three as a substitute .
4. leicester city have rejected approaches for striker tom lawrence from an astonishing nine clubs .
5. championship promotion chasers bournemouth , ipswich and wolves have all asked about lawrence .
6. blackburn , charlton , leeds , bolton , rotherham and wigan have also made contact .
7. however , they are now looking at other options in a last-gasp bid to bolster their squad .

Reference:
bournemouth , ipswich , wolves , blackburn , charlton , leeds , bolton , rotherham and wigan have all asked about tom lawrence . the 21-year-old is poised to make his full international debut for wales . leicester manager nigel pearson wants to have options available .

StructSum:
leicester city have rejected approaches for tom lawrence . lawrence is poised to make his debut for wales in their european championship qualifier with israel on saturday . leicester city are looking at other options in last-gasp bid to bolster their squad . lawrence from old trafford has only figured in four games for leicester this season and three as a substitute . the former manchester united star has barely

1. andrew henderson celebrated landing the london broncos coaching job on a permanent basis as halifax were beaten 22-18 .
2. henderson was given the nod by the london hierarchy this week after a mixed spell in caretaker charge since the departure of joey grima .
3. his weakened side put on a fine show to crown his appointment , though , scoring four tries through daniel harrison , matt garside , iliess macani and brad dwyer , whose score was the winning one .
4. iliess macani , pictured last year , scored one of london broncos ' four tries in the 22-18 win over halifax .
5. james saltonstall , ben heaton and mitch cahalane scored for halifax .
6. henderson had spoken earlier in the week about how he felt broncos were moving in the right direction , and their narrow victory put some substance to his words .
7. the win was just their third in six in the kingstone press championship having been relegated from super league at the end of last season .

Reference:
andrew henderson won his first game as broncos full-time coach . daniel harrison , matt garside , iliess macani and brad dwyer all scored . james saltonstall , ben heaton and mitch cahalane scored for halifax .

StructSum:
andrew henderson celebrated landing the coaching job on a permanent basis .henderson was given the nod by london hierarchy this week after a mixed spell in the 22-18 win over halifax . his weakened side put on fine show to crown his appointment , though he felt broncos were moving in the right direction . the win was their third in six in the press championship having been relegated

Figure 4: Examples of induced structures and generated summaries.

summary generation. We also see in example 1 that the latent structures cluster sentences based on the main topic of the document. Sentences 1,2,3 differ from sentences 5,6,7 in the topic discussed and our model clustered the two sets separately.

6 Related Work

Data-driven neural summarization falls into *extractive* (Cheng et al., 2016; Zhang et al., 2018) or *abstractive* (Rush et al., 2015; See et al., 2017; Gehrmann et al., 2018; Chen and Bansal, 2018). Pointer-generator See et al. (2017) learns to either generate novel in-vocabulary words or copy from the source. It has been the foundation for much work on abstractive summarization (Gehrmann et al., 2018; Hsu et al., 2018; Song et al., 2018). Our model extends it by incorporating latent/explicit structure, but these extensions are applicable to any other encoder-decoder architecture. For example, a follow-up study has already shown benefits of our method in multi-document summarization (Chowdhury et al., 2020).

In pre-neural era, document structure played a critical role in summarization (Leskovec et al., 2004; Litvak and Last, 2008; Liu et al., 2015; Durrett et al., 2016; Kikuchi et al., 2014). More recently Song et al. (2018) infuse source syntactic structure into the pointer-generator using word-level syntactic features and augmenting them to decoder copy mechanism. In contrast, we model sentence dependencies as latent structures and explicit coreference structures; we do not use heuristics or salient features. Li et al. (2018) propose structural compression and coverage regularizers incorporating structural bias of target summaries while we model the structure of the source document. Fremann and Klementiev (2019) induce latent structures for aspect based summarization, Cohan et al. (2018) focus on summarization of scientific papers, Isonuma et al. (2019) reviews unsupervised summarization, Mithun and Kosseim (2011) use discourse structures to improve coherence in blog summarization and Ren et al. (2018) use sentence relations for multi-document summarization. These are complementary directions to our work. To our knowledge, StructSum is the first to jointly incorporate latent and explicit document structure in a summarization framework.

7 Conclusion and Future Work

In this work, we propose the framework StructSum for incorporating latent and explicit document structure in neural abstractive summarization. We introduce a novel explicit-attention module which incorporates external linguistic structures, instantiating it with coreference links. We show that our framework improves the abstractiveness and coverage of generated summaries, and helps mitigate layout biases associated with prior models. We present an extensive evaluation of StructSum along abstractiveness, coverage, and layout quantitatively. Future work will investigate the role of document structures in pretrained language models (Lewis et al., 2019; Liu and Lapata, 2019).

Acknowledgements

The authors are grateful to the anonymous reviewers for their invaluable feedback and to Waleed Ammar and Kathryn Mazaitis for their help and support in various stages of the project. This material is based upon work supported by the DARPA SemaFor and NNSA DOE programs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA or NNSA. We would also like to thank Amazon for providing GPU credits.

References

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Tanya Chowdhury, Sachin Kumar, and Tanmoy Chakraborty. 2020. Neural abstractive summarization with structural attention. *arXiv preprint arXiv:2004.09739*.

Yau Chu and Tung Kuan Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1998–2008, Berlin, Germany. Association for Computational Linguistics.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Elisa Ferracane, Greg Durrett, Junyi Jessy Li, and Katrin Erk. 2019. Evaluating discourse in structured text representations. In *ACL*.

Lea Frermann and Alexandre Klementiev. 2019. Inducing document structure for aspect-based summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6263–6273.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Wan Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *ACL*.

Masaru Isonuma, Junichiro Mori, and Ichiro Sakata. 2019. Unsupervised neural single-document summarization of reviews via learning latent discourse structure and its ranking. In *ACL*.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Yuta Kikuchi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura, and Masaaki Nagata. 2014. Single document summarization based on nested tree structure. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 315–320.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Kundan Krishna, Sopan Khosla, Jeffrey P Bigham, and Zachary C Lipton. 2020. Generating soap notes from doctor-patient conversations. *arXiv preprint arXiv:2005.01795*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Jure Leskovec, Marko Grobelnik, and Natasa Milic-Frayling. 2004. Learning sub-structures of document semantic graphs for document summarization. In *LinkKDD Workshop*, pages 133–138.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461.

Wei Li, Xinyan Xiao, Yajuan Lyu, and Yuanzhuo Wang. 2018. Improving neural abstractive document summarization with structural regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4078–4087.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9815–9822.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24. Association for Computational Linguistics.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2017. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, 6:63–75.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *IJCNLP*, abs/1908.08345.

Zhengyuan Liu, Angela Ng, Sheldon Lee Shao Guang, AiTi Aw, and Nancy F. Chen. 2019. Topic-aware pointer-generator networks for summarizing spoken conversations. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 814–821.

Shamima Mithun and Leila Kosseim. 2011. Discourse structures to reduce discourse incoherence in blog summarization. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 479–486, Hissar, Bulgaria. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

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

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Liqiang Nie, Jun Ma, and Maarten De Rijke. 2018. Sentence relations for extractive summarization with deep neural networks. *ACM Transactions on Information Systems (TOIS)*, 36(4):1–32.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389.

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

Kaiqiang Song, Lin Zhao, and Fei Liu. 2018. Structure-infused copy mechanisms for abstractive summarization. In *COLING*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

William Thomas Tutte. 1984. Graph theory, vol. 21 of. *Encyclopedia of Mathematics and its Applications*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*.

Tianyi Zhang, V. Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *ICLR*.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 779–784, Brussels, Belgium. Association for Computational Linguistics.

Project-then-Transfer: Effective Two-stage Cross-lingual Transfer for Semantic Dependency Parsing

Hiroaki Ozaki, Gaku Morio, Terufumi Morishita, Toshinori Miyoshi
Research and Development Group, Hitachi, Ltd., Japan
`{hiroaki.ozaki.yu, gaku.morio.vn, terufumi.morishita.wp, toshinori.miyoshi.pd}@hitachi.com`

Abstract

This paper describes the first report on cross-lingual transfer for semantic dependency parsing. We present the insight that there are two different kinds of cross-linguality, namely surface level and semantic level, and try to capture both kinds of cross-linguality by combining annotation projection and model transfer of pre-trained language models. Our experiments showed that the performance of our graph-based semantic dependency parser almost achieved the approximated upper bound.

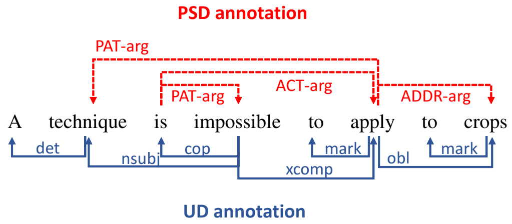


Figure 1: Example dependency annotations. Above: Semantic dependency (PSD). Below: Syntactic dependency (UD). Semantic dependency focuses more on meaning relationship between words.

1 Introduction

Cross-lingual dependency parsing attracted much attention for its powerful representational capability in grammatical and semantic lexical relations (Zhang and Barzilay, 2015; Guo et al., 2015; Ammar et al., 2016; Zeman et al., 2017, 2018; de Lhoneux et al., 2018; Schuster et al., 2019). Several remarkable contributions have been made in *syntactic* dependency parsing, especially on universal dependencies (UD; Nivre et al. 2016). For example, Kondratyuk and Straka (2019) showed that a single multilingually fine-tuned neural model utilizing a pre-trained language model could successfully parse 75 languages in UD with comparable performances to state-of-the-art parsers.

However, cross-lingual *semantic* dependency parsing (Oepen et al., 2014, 2015, 2016), which is totally different dependency structure from syntactic dependencies (shown in Figure 1), has not been explored as far as we know. A reason for this is the lack of parallel graphbanks that cover many languages with consistent annotation policies. One exception is Prague Semantic Dependencies (PSD; Mikulová 2009), which is a treebank of bi-lexical semantic graphs and contains over 30,000 pairs of parallel annotated sentences from the Wall Street Journal in English and Czech.

Considering these circumstances, we propose to train semantic dependency parsers by capturing commonalities across languages as a remedy for the absence of massive multilingual graphbanks. Our work draws on the intuition that cross-linguality exists in both superficial level and semantic level. Accordingly, we leverage a two-stage fashion involving treebank-based transfer and model-based transfer.

Treebank-based transfer, often called annotation projection, is a method of projecting source language annotations to a target language by using a mapping function such as word alignment. The annotation projection has been reported as a promising approach under truly low-resource settings for UD parsing (Rosa and Mareček, 2018). However, annotation projection often suffers from noise in word alignment (Damonte and Cohen, 2018). For the model-based transfer, several studies on transferring contextualized word vectors have reported that it improves the parsing performance (Mulcaire et al., 2019; Kondratyuk and Straka, 2019).

Our experiments on PSD graphbank indicate that the optimal performance can be achieved by incorporating the two-stage transfer. Surprisingly, we observed improvement even when the projected treebank was erroneous. Furthermore, the two-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2586–2594
April 19 - 23, 2021.

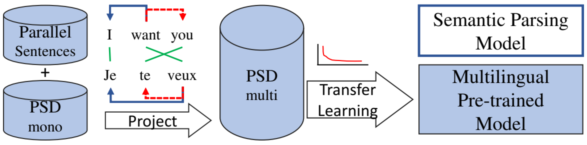


Figure 2: Project-then-Transfer approach.

stage transfer method achieved almost upper-bound performance, which was approximated by evaluating the cross-linguality of PSD annotation through the projection. We also provide detailed analyses from both perspectives of cross-linguality.

2 Related Work

Semantic Dependency Parsing: The topic of semantic dependency parsing has spurred enduring interest (Peng et al., 2017, 2018; Dozat and Manning, 2018; Wang et al., 2019; Kurita and Søgaard, 2019). Much of the current interest lies in higher-level interactions between relations. In parallel with our study, Aminian et al. (2020) shows improvement of PSD parsing trained on cross-lingually projected graphbank with multitask training of UD parsing as an auxiliary task. Unlike their work, we perform a zero-shot training with UDify pretrained model to validate the hypothesis of the two different cross-linguality.

Utilizing Models across Different Graphbanks: Parsing semantic graphs in different semantic abstraction levels was introduced as CoNLL shared task 2019 (Oepen et al., 2019). Candidate teams tackled this problem with methods such as transition-based parsers (Hershcovich et al., 2018; Bai and Zhao, 2019; Lai et al., 2019) and graph-based parsers (Zhang et al., 2019; Koreeda et al., 2019). Small improvements were reported in both approaches, but improving semantic parsing on different semantic graphs remains a difficult problem.

3 Transfer Strategies

To enable cross-lingual semantic parsing, we focused on two different types of cross-linguality, namely *cross-linguality on surface* and *cross-linguality in semantics*. *Cross-linguality on surface* lies on our hypothesis of typological correspondences among most of languages. For example, annotation projection, which is a treebank-based transfer, assumes *cross-linguality on surface* and projects source language annotations to a target language. On the other hand, *cross-linguality in semantics* is based on the assumption that lexical-, phrase-, or sentence-level meaning correspondences may exist among most of languages. Recently, multilingual BERT (Devlin et al., 2019; Pires et al., 2019) directly captures *cross-linguality in semantics* beyond lexicons by large-scaled language model training on parallel corpora.

Though both annotation projection and multilingual pre-trained model can handle either *cross-linguality on surface* or *in semantics*, we argue that they could not utilize both cross-linguality in effective way. Hence, we propose two-stage transfer which incorporates both methods, to capture the two kinds of cross-linguality as possible. We firstly introduce the two transfer methods for applying them to PSD graphbank, and then we explain our two-stage transfer method; Project-then-Transfer.

Annotation Projection: As aforementioned, this is the approach focuses on *cross-linguality on surface*. We trained word alignment model on PSD graphbank, and then projected all annotations in a monolingual graphbank to the other language.

Zero-shot Model Transfer: In this study, we transferred only pre-trained language models, because we aimed to focus on *cross-linguality in semantics* more.We trained PSD parsers with multilingual pre-trained model on a monolingual graphbank in PSD, and then apply the monolingually trained parsers to the other language.

Project-then-Transfer: We incorporate both transfer methods by applying them in two-stage fashion as shown in Figure 2. Firstly, we prepared multilingually projected PSD graphbanks. We automatically generated PSD annotations on English sentences in a multilingual parallel corpus by the previously introduced English PSD parser which was created in the zero-shot approach. By utilizing bi-lingual word alignment, we projected PSD annotations on English to other languages. We finally trained Project-then-Transfer models on a concatenated graphbank of both original and projected PSD.

4 Experiments

4.1 Setup and Implementations

To perform word alignment, we used an IBM2 aligner *fast_align*[1] (Dyer et al., 2013).

---

[1] `https://github.com/clab/fast_align`

Language	Model	Approach	UP	UR	UF	LP	LR	LF	LF/UF
Czech (Trained on English)	Graph-UDify	**Project-then-Transfer (ours)**	86.5	**78.7**	**82.5**	**62.4**	**56.7**	**59.4**	72.0
	Graph-BERT		80.1	62.7	70.4	59.7	46.8	52.5	74.6
	Graph-UDify	Zero-shot	**86.9**	75.5	80.8	61.8	53.7	57.5	71.1
	Graph-BERT		79.1	61.1	69.0	58.7	45.3	51.1	74.1
	fast_align	Projection	49.3	40.7	44.6	37.1	30.6	33.5	**75.1**
English (Trained on Czech)	Graph-UDify	**Project-then-Transfer (ours)**	77.4	**79.7**	**78.5**	57.4	**59.1**	**58.2**	74.1
	Graph-BERT		66.9	57.7	62.0	50.7	43.7	46.9	75.6
	Graph-UDify	Zero-shot	**81.0**	72.4	76.5	**59.2**	52.9	55.9	73.1
	Graph-BERT		73.6	58.1	65.0	56.5	44.6	49.9	**76.8**
	Transition-BERT		55.2	22.6	32.3	41.8	17.2	24.4	75.5
	fast_align	Projection	48.6	44.3	46.3	35.6	32.4	33.9	73.2
Li et al. (2019) (English monolingual training)			93.	92.	92.5	82.	81.	81.7	88.3

Table 1: SDP scores for each model and approach. U and L stand for "unlabeled" and"labeled" respectively. P, R, and F stand for "precision", "recall" and "F1-score" respectively. LF/UF is a proxy metric of label prediction accuracy. Bold values represent the best scores. Li et al. (2019) is the best PSD parser at CoNLL 2019 shared task.

To perform model-based transfer, we used mainly graph-based parsers, but we also used a transition-based parser for a comparison purpose. Our graph-based PSD parser employed UDify architecture[2] (Kondratyuk and Straka, 2019). We replaced activation function of biaffine attention layers in UDify with sigmoid activation (Dozat and Manning, 2018). We trained two variances of graph-based parsers, and a transition-based parser:

Graph-BERT: We trained it with mulitilinugal BERT as Kondratyuk and Straka (2019) did.

Graph-UDify: We trained it with UDify's pre-trained language model[3] instead of multilingual BERT. Since UDify is pre-trained on many languages in UD, we expect that it capture more *cross-linguality on surface* than BERT.

Transition-BERT: We used an architecture introduced by Che et al. (2019)[4], which was the best transition-based parser in the CoNLL 2019 shared task (Oepen et al., 2019). We trained it from scratch with the same hyperparameters given by the source code.

The pre-trained multilingual BERT[5] was downloaded via the above parser implementations. We used mtool[6] to evaluate SDP scores (Oepen et al., 2014) as metrics for parsing performance. A list of the best hyperparameters is available in Appendix. We added "tag_loss_w", which is a constant multiplied by the loss of relation label predictions.

We divided PSD graphbank into three splits, namely train-set (30,000 pairs), dev-set (2000 pairs), and test-set (3653 pairs). We selected the best models by monitoring the labeled F1-score of SDP on the dev-set of the target language and evaluated the scores on the test-set of the target language. We chose Parallel Universal Dependencies (PUD; Zeman et al. 2017) as additional multilingual parallel corpora for Project-then-Transfer, because they contain 1,000 parallel sentences for 18 languages, with mostly consistent UD annotations. Further details are in Appendix.

4.2 Results and Discussion

Table 1 shows the SDP scores for each model in each approach. Firstly, we focus on the cross-linguality of PSD annotations by the annotation projection. Unlabeled scores of projection models were within a range of 0.4 - 0.5. Since alignment error rate (AER) of English-Czech reported around 0.25 (Legrand et al., 2016), edge projection accuracy could be estimated as $(1 - \mathrm{AER})^2 \approx 0.56$ [7]. Annotation agreement rate of relations[8] between the two languages was estimated to fluctuate between 0.7 to 0.9 according to the mitigation efficacy of alignment error. Annotation agreement rate of relation labels was estimated as about 0.75 by comparing unlabeled and labeled scores (LF/UF of fast_align model). These rates could be upper bounds of performances.

By comparing monolingual training (Li et al.,

---

[2] `https://github.com/Hyperparticle/UDify` (Pre-trained models are also available from the link.)

[3] We did not utilize biaffine and MLP layers of UDify.

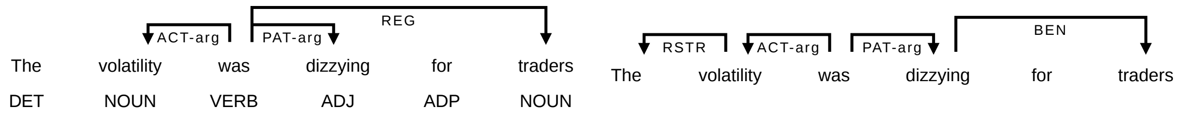
[4] `https://github.com/DreamerDeo/HIT-SCIR-CoNLL2019`

[5] `https://github.com/google-research/bert/blob/master/multilingual.md`

[6] `https://github.com/cfmrp/mtool`

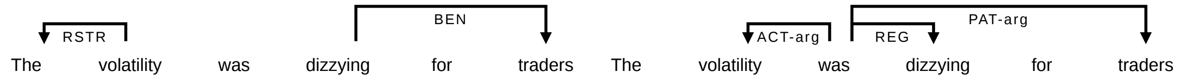
[7] Suppose one edge has two nodes A and B, then edge projection accuracy is estimated as probability that both projected nodes A' and B' are correct.

[8] We simply divided the unlabeled scores by the projection accuracy estimated by AER.



(a) Gold

(b) Annotation projection



(c) Graph-UDify zero-shot

(d) Graph-UDify Project-then-Transfer

Figure 3: Example gold and Graph-UDify outputs in each scenario (English).

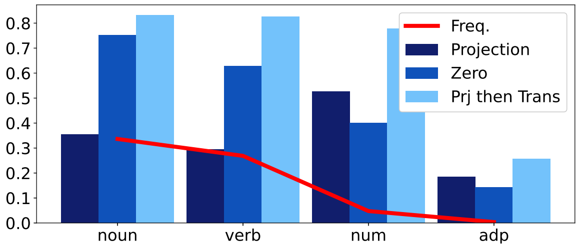


Figure 4: Relation accuracy for each UPOS type.

Model	language	UF	LF
Graph-BERT + PUD_Czech	en2cs	68.9	50.8
Graph-UDify + PUD_Czech	en2cs	80.6	57.8

Table 2: Unlabeled and labeled scores trained on English PSD with projected PUD_Czech.

2019), unlabeled-F of Project-then-Transfer (UDify) was about 85% of that of monolingual training. This ratio is in the range of estimated annotation agreement rate of relations.

Models and Approaches Comparison: As we can see from Table 1, the graph-based models outperformed the Transition-BERT, especially, graph-based UDify models demonstrated superiority to the other models. Graph-UDify of Project-then-Transfer approach, which is the best model, achieved an unlabeled F1-score of 82.5, that is close to the upper-bounds estimated above. In additions, there were few differences in LF/UF scores, which are also close to the upper-bound of relation label prediction. Thus, our best model achieved high performance, which is close to theoretical upper bounds. This indicates that bi-lexical relations captured by syntactic dependency are also helpful for parsing semantic dependency, yet there remaining information that were not captured in the UDify. We claim that the missing information was related to *cross-linguality on surface*, then we perform a deeper analysis on this in the following paragraph.

What is NOT captured by Pre-trained Models?: Figure 3 shows examples of gold and Graph-UDify (cs2en) outputs. The annotation projection had managed completely project *unlabeled* relations in the source language, but a swapping had happened between two relations, namely "REG" and "PAT-arg", which had been caused by alignment errors.

We observed that parsers based on the model-based transfer often failed to parse relations which contain functional words. This phenomenon can be observed in Figure 3c. Those relations containing functional words tended to be successfully converted by the annotation projection. Hence, we obtained better results with the Project-then-Transfer approach as shown in Figure 3d. This implies that pre-trained models including UDify represent rather *semantic* bi-lexical relations than grammatical ones.

We performed a further analysis on cross-linguality of UDify model. Figure 4 shows relation accuracy for each of four UPOS, namely *noun*, *verb*, *num* and *adp*. We calculated conditional "unlabeled" relation accuracy, which measures whether source or target word belongs to the specific UPOS type. By focusing on the accuracy of *num* (numeric) and *adp* (adposition), which are considered to be hard-to-contextualize examples, the annotation projection outperformed the zero-shot approach. The Project-then-Transfer approach improved the accuracy for almost all UPOS types including *num* and *adp*.

Cross-linguality on Surface: Table 5 shows SDP scores trained on English PSD with projected PUD_Czech. The performances were comparable to the zero-shot approach, but less than those of the Project-then-Transfer approach. Hence, multilingually projected treebank is significant to improve the performances. This implies that *cross-linguality on surface* can be captured by training on multilingually projected treebank.

5 Conclusion

This paper described transfer methods for cross-lingual semantic dependency parsing. We showed that both *cross-linguality on surface* and *in semantics* were necessary to improve the performance. Consequently, we achieved almost the upper bound performance approximated by the annotation projection. The results encouraged us to develop cross-lingual semantic dependency parser for many languages. We will further conduct explore these models, and evaluations on cross-linguality across languages broadly.

Acknowledgments

Computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by the National Institute of Advanced Industrial Science and Technology (AIST) was used. We would like to thank the anonymous reviewers for their helpful comments. We also thank Dr. Masaaki Shimizu for the convenience of the computational resources.

References

Maryam Aminian, Mohammad Sadegh Rasooli, and Mona Diab. 2020. Multitask learning for cross-lingual transfer of broad-coverage semantic dependencies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8268–8274, Online. Association for Computational Linguistics.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Hongxiao Bai and Hai Zhao. 2019. SJTU at MRP 2019: A transition-based multi-task parser for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 86–94, Hong Kong. Association for Computational Linguistics.

Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, and Ting Liu. 2019. HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 76–85, Hong Kong. Association for Computational Linguistics.

Marco Damonte and Shay B. Cohen. 2018. Cross-lingual abstract meaning representation parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1146–1155, New Orleans, Louisiana. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2018. Simpler but More Accurate Semantic Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1234–1244, Beijing, China. Association for Computational Linguistics.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2018. Multitask parsing across semantic representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Yuta Koreeda, Gaku Morio, Terufumi Morishita, Hiroaki Ozaki, and Kohsuke Yanai. 2019. Hitachi at MRP 2019: Unified encoder-to-biaffine network for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 114–126, Hong Kong. Association for Computational Linguistics.

Shuhei Kurita and Anders Søgaard. 2019. Multi-task semantic dependency parsing with policy gradient for learning easy-first strategies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2420–2430, Florence, Italy. Association for Computational Linguistics.

Sunny Lai, Chun Hei Lo, Kwong Sak Leung, and Yee Leung. 2019. CUHK at MRP 2019: Transition-based parser with cross-framework variable-arity resolve action. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 104–113, Hong Kong. Association for Computational Linguistics.

Joël Legrand, Michael Auli, and Ronan Collobert. 2016. Neural network-based word alignment through score aggregation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 66–73, Berlin, Germany. Association for Computational Linguistics.

Miryam de Lhoneux, Johannes Bjerva, Isabelle Augenstein, and Anders Søgaard. 2018. Parameter sharing between dependency parsers for related languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4992–4997, Brussels, Belgium. Association for Computational Linguistics.

Zuchao Li, Hai Zhao, Zhuosheng Zhang, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. SJTU-NICT at MRP 2019: Multi-task learning for end-to-end uniform semantic graph parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 45–54, Hong Kong. Association for Computational Linguistics.

Marie Mikulová. 2009. Pokyny k překladu určené překladatelům, revizorům a korektorům textů z wall street journal pro projekt PCEDT. Technical Report TR-2009-41, ÚFAL MFF UK.

Phoebe Mulcaire, Jungo Kasai, and Noah A. Smith. 2019. Low-resource parsing with crosslingual contextualized representations. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 304–315, Hong Kong, China. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Stephan Oepen, Omri Abend, Jan Hajic, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdenka Uresova. 2019. MRP 2019: Cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 1–27, Hong Kong. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Zdeňka Urešová. 2016. Towards comparability of linguistic graph Banks for semantic parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3991–3995, Portorož, Slovenia. European Language Resources Association (ELRA).

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048, Vancouver, Canada. Association for Computational Linguistics.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1492–1502, New Orleans, Louisiana. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Rudolf Rosa and David Mareček. 2018. CUNI x-ling: Parsing under-resourced languages in CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018*

Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 187–196, Brussels, Belgium. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy. Association for Computational Linguistics.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Yuan Zhang and Regina Barzilay. 2015. Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1857–1867, Lisbon, Portugal. Association for Computational Linguistics.

Model	Parameters
fast_align	– iteration: 10 – num. of trials: 1 – average runtime: less than 1min.
Graph-UDify	– batch size: 32 – learning rate: 1e-3 – activation: relu – beta: [0.99, 0.99] – tag_representation_dim: 128 – tag_loss_w: 0.1 – num. of trials: 32 for each – average runtime: 12 hrs. – num. of params.: 225695576
Transition-BERT	– batch size: 4 – num. of trials: 1 – average runtime: 48 hrs. – num. of params.: 201555396

Table 3: The best hyperparameters and training settings for fast_align, Graph-UDify, and Transition-BERT.

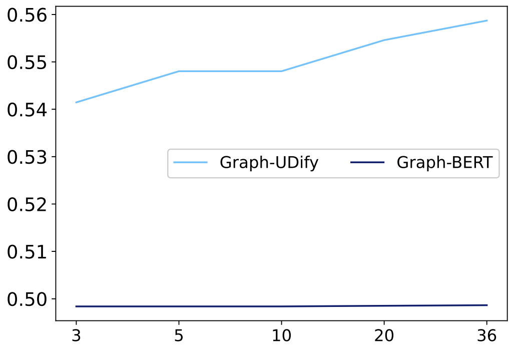


Figure 5: Expected Validation Performances.

Yue Zhang, Wei Jiang, Qingrong Xia, Junjie Cao, Rui Wang, Zhenghua Li, and Min Zhang. 2019. SUDA-Alibaba at MRP 2019: Graph-based models with BERT. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 149–157, Hong Kong. Association for Computational Linguistics.

A Appendices

In this section, we provide details of training setup, analyses on relation accuracy and multi-task learning results.

A.1 Detail of Training Setups

In this sub-section, we provide training setups, which are for the reproducibility criteria. Table 3 shows all detailed settings. We did not performed a severe automatic hyperparameter tuning, but did a manual tuning. Thus our best hyperparameters may be different from the true best hyperparam-

Language	Model	Approach	UP	UR	UF	LP	LR	LF	LF/UF
Czech	Graph-BERT	**Project-then-Transfer**	80.5	62.0	70.1	60.1	46.3	52.3	74.6
	Graph-UDify		86.4	78.3	82.7	62.5	57.3	59.8	72.3
(Trained on English)	Graph-BERT	Zero-shot	78.8	60.3	68.4	58.4	44.7	50.6	74.0
	Graph-UDify		86.5	75.6	80.7	61.3	53.6	57.1	70.8
	fast_align	Projection	49.3	40.7	44.6	37.1	30.6	33.5	75.1
English	Graph-BERT	**Project-then-Transfer**	67.3	57.4	62.0	49.2	42.0	45.3	73.1
	Graph-UDify		77.3	79.6	78.5	55.7	57.4	56.5	72.0
(Trained on Czech)	Graph-BERT	Zero-shot	73.4	57.6	64.5	54.8	43.0	48.2	74.7
	Graph-UDify		81.3	72.5	76.7	57.6	51.3	54.3	70.8
	Transition-BERT		55.4	62.6	58.8	40.4	45.7	42.8	72.8
	fast_align	Projection	48.6	44.3	46.3	35.6	32.4	33.9	73.2

Table 4: SDP scores on the dev-set for each model and approach.

Model	language	UF	LF
Graph-BERT + PUD_Czech	en2cs	69.0	50.7
Graph-UDify + PUD_Czech	en2cs	80.9	58.0

Table 5: Unlabeled and labeled dev-set scores of models trained on English PSD with projected PUD_Czech.

eters. We tuned hyperparamters in zero-shot approach, then we reused the best hyperparameters in Project-then-Transfer approach. We trained all models with NVIDIA V100 on Ubuntu 18.04. Our GPU environment is a mixture of both 32GB and 64GB memories.

We obtained PSD treebank from the Linguistic Data Consortium (LDC) [9]. We converted original SDP format data to MRP format before the training. This SDP to MRP graph conversion is a loss-less conversion.

A.2 Performances on Dev-set

Figure 5 shows *expected validation performances*. Table 4 and Table 5 show performances on the dev-set. Most of scores were consistent with the performances on the test-set. Only transition-based model is an exceptional case. Though over-fitting seemed to happen, its performances are still lower than those of graph-based models.

A.3 Full Results of Relation Accuracy

Figure 6 shows relation accuracy for all UPOS types. We can see that zero-shot performances of eleven types, namely *noun*, *verb*, *propn* (proper noun), *conj* (conjunction), *pron* (pronoun), *adv* (adverb), *punct* (punctuation), *det* (determiner), *part* (particle), *cconj* (coordinating conjunction), and *x* (other), are outperformed those of annotation projection, and all content words are included in this group. Zero-shot performances of the other five types, *num*, sym (symbol), *adp*, *sconj* and *intj* (interjection), are comparable or underwhelmed to those of annotation projection. Because the words categorized as *intj* only appeared a few times in this analysis, we could not make a discussion regarding *intj*.

A.4 Multi-Task Learning

We argue that it is natural to perform multi-task learning (MTL) of UD and PSD dependencies when both annotations are available, since UDify's pre-train model, which is trained on UD annotations, improved the performances. Firstly, we added UD annotation on PSD treebank by existing UD parser UDPipe[10]. Our MTL setting is to share only BERT layers, but higher layers including scalar-mix layers are distinct. We used UDify "as is" for UD prediction. A Loss function to perform MTL is a simple linear combination of that of UDify and our PSD model. We show the MTL results in Table 6. Performances were degraded by comparing to those of non-MTL models. This could be because UD and PSD annotations are contradictive to perform MTL.

---

[9] `https://catalog.ldc.upenn.edu/LDC2016T10`

[10] `http://ufal.mff.cuni.cz/udpipe`

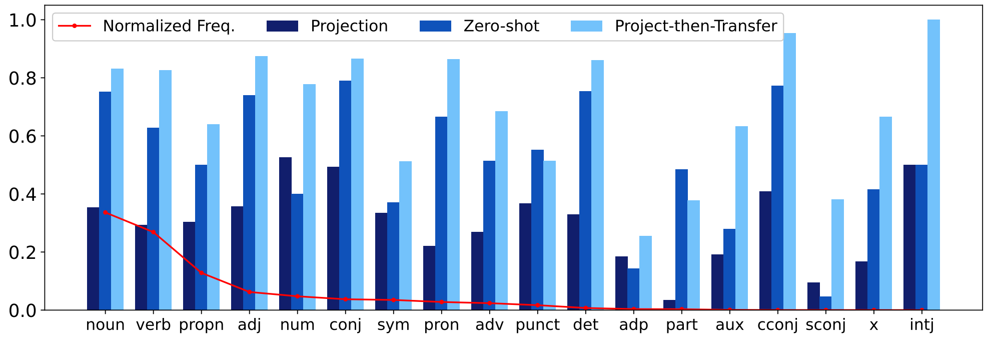


Figure 6: Relation accuracy for all UPOS types.

Dataset	Model	Approach	UP	UR	UF	LP	LR	LF	LF/UF
test-set	Graph-BERT-MTL	Project-then-Transfer	83.7	60.5	70.2	61.6	44.5	51.7	73.6
	Graph-UDify-MTL		86.4	73.9	79.7	63.3	54.1	58.4	73.3
dev-set	Graph-BERT-MTL	Project-then-Transfer	83.9	57.0	67.9	61.9	42.0	50.0	73.6
	Graph-UDify-MTL		85.9	73.5	79.2	63.3	54.1	58.3	73.6

Table 6: SDP scores for MTL model and approach.

LSOIE: A Large-Scale Dataset for Supervised Open Information Extraction

Jacob Solawetz
Roboflow, Inc.
Minneapolis, MN, USA
jacob@roboflow.ai

Stefan Larson
Rosegold AI
Ann Arbor, MI, USA
stefan@rosegold.ai

Abstract

Open Information Extraction (OIE) systems seek to compress the factual propositions of a sentence into a series of n-ary tuples. These tuples are useful for downstream tasks in natural language processing like knowledge base creation, textual entailment, and natural language understanding. However, current OIE datasets are limited in both size and diversity. We introduce a new dataset by converting the QA-SRL 2.0 dataset to a large-scale OIE dataset (LSOIE). Our LSOIE dataset is 20 times larger than the next largest human-annotated OIE dataset. We construct and evaluate several benchmark OIE models on LSOIE, providing baselines for future improvements on the task. Our LSOIE data, models, and code are made publicly available.[1]

1 Introduction

Open Information Extraction (OIE) (Banko et al., 2007) aims to automatically extract all factual propositions of a sentence into a series of n-ary tuples. For example, the sentence "the cook baked and ate the cake" would produce two extractions representing the two basic propositions of the sentence: (the cook, **ate**, the cake) and (the cook, **baked**, the cake). In OIE, extraction arguments are required to be contiguous spans from the sentence and the resulting tuple should be intelligible as natural text when read in order. The schema-free nature of OIE provides a flexible framework in which to capture semantic relations between entities in natural language text. Open Information Extraction tuples are useful to a variety of downstream tasks including knowledge base creation (Zhang et al., 2019), textual entailment (Levy et al., 2014), and other natural language understanding tasks (Mausam, 2016).

	Domains	#Sent.	#Ext.	Ext. / Sent.	Vocab	Ordered
OIE2016	Wiki, Newswire	3,180	8,477	2.7	13,863	
AW-OIE	Wiki, Wikinews	3,300	17,165	5.2	15,853	
LSOIE-wiki	Wiki, Wikinews	24,296	56,662	2.3	46,617	✓
LSOIE-sci	Science	47,998	97,550	2.0	51,668	✓

Table 1: OIE dataset metrics. Our new dataset LSOIE has substantially more text available than prior work, and includes a new science domain. Our dataset conversion process leverages the scope of the QA-SRL 2.0 bank and improves upon previous methodology.

Open Information Extraction relations may be explicitly stated by verbal predicates, or implicitly stated through nominalizations. In this paper, we focus only on explicit extractions. With the original goal of OIE as web scale information extraction (Banko et al., 2007), an OIE system can focus solely on explicit extractions because the redundancy of language will inevitably display implicit information elsewhere.

The interest in OIE has grown: both in terms of the types of models that can be applied to tackle OIE (Cui et al., 2018; Stanovsky et al., 2018; Jiang et al., 2019), and in terms of the downstream applications to which OIE can be applied (Mausam, 2016; Zhang et al., 2019). As the interest in OIE grows, however, so too should the scale of the corpora available for training and evaluating OIE models.

In this paper, we expand the reach and quality of OIE data by developing a new dataset, LSOIE, which is built by converting the QA-SRL BANK 2.0 dataset (FitzGerald et al., 2018) to the task of OIE. Our new dataset contains almost ten times as many extractions and about 20 times as many sentences as previous OIE datasets built from human anno-

---

[1] Our LSOIE dataset, models, and code can be found at `https://github.com/Jacobsolawetz/large-scale-oie`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2595–2600
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

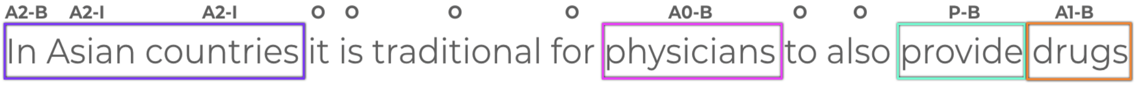


Figure 1: An example annotated sentence from QA-SRL 2.0 (FitzGerald et al., 2018). In this case, the annotations are derived from the question and answers: - Where does someone **provide** something? **In Asian countries**. Who **provides** something? **physicians**. What is being **provided**? **drugs**. The extracted tuple in our new LSOIE dataset is (physicians, **provide**, drugs, in Asian countries).

tations (see Table 1). We benchmark LSOIE with several models, providing baseline results for future research. Our LSOIE dataset, models, and code are publicly available.

2 Background

2.1 OIE Datasets

Available OIE corpora fall into three categories: (1) converted from crowdsourcing, (2) model-derived, and (3) directly crowdsourced.

Converted from crowdsourcing: Stanovsky and Dagan (2016) created the OIE2016 dataset by converting the crowd-annotated QA-SRL (He et al., 2015) dataset's question-answer pairs to OIE extraction relations. Similarly, Stanovsky et al. (2018) generated the AW-OIE dataset by converting the crowd-annotated Question Answer Meaning Representation (QAMR) dataset's question-answer pairs.

The OIE2016 and AW-OIE datasets were the first datasets used for supervised OIE. These datasets provided the basis for supervised approaches in NLP, but they are small and extractions lack accuracy, as they are converted in the order that question answer pairs appear in the base dataset.

Model-derived: Cui et al. (2018) and Jia et al. (2018) generate large derivative training datasets by running rules-based models and keeping high confidence extractions for downstream tasks. Similarly, Gashteovski et al. (2019) introduce the largest OIE dataset to date (over 340M triples) by deriving extractions from `MinIE` Gashteovski et al. (2017) with the goal of automatically constructing a knowledge base. While model-derived datasets are useful for knowledge base construction, using them for downstream tasks teaches the new model to replicate the behavior of the original, often noisy, base model.

Directly crowdsourced: Bhardwaj et al. (2019) point out that the evaluation framework used in Stanovsky and Dagan (2016) is rather noisy and the tuple matching algorithm is overly lenient because it only looks at lexical overlap for the whole extraction, ignoring the ordering of arguments. Bhardwaj et al. (2019) provide an alternative evaluation set that has been crowdsourced specifically for OIE, annotating 1,282 sentences. While this dataset is useful for the evaluation of OIE systems, its format differs from other work in OIE - the predicate entry in CARB (Bhardwaj et al., 2019) tuples contains context that is often broken into separate tuples by other OIE systems.

2.2 The QA-SRL Bank 2.0

In QA-SRL, each predicate-argument relationship in a sentence is labeled manually with a question-answer pair. FitzGerald et al. (2018) design a large-scale crowdsourcing annotation pipeline to incentivize extensive and accurate coverage. Relative to the original QA-SRL annotations (He et al., 2015), which were collected from 10 hired freelance workers, the new QA-SRL dataset achieves similar precision (95.7% versus 97.5%) and lower recall (72.4% versus 86.6%). Relative to Propbank (Palmer et al., 2005), an expert annotation system designed to capture all semantic roles in a sentence. the QA-SRL 2.0 authors find that their work 95% precision and 85% recall. FitzGerald et al. (2018) then build a supervised QA-SRL parser and extend the reach of their dataset by over-generating new candidate question-answer pairs and passing them through their validation process.

The QA-SRL paradigm is well-suited to be a precursor to OIE extractions, as it captures predicate-argument relations in a schema-free way.

3 The LSOIE Dataset

Our work expands upon and addresses the shortcomings present in Stanovsky and Dagan (2016) and Stanovsky et al. (2018). We apply a similar conversion processes used for OIE2016 on the QA-SRL BANK 2.0 dataset. In addition, we implement novel conversion heuristics to ensure data quality and order arguments. The result is LSOIE, an OIE dataset that is much larger and diverse than prior work.

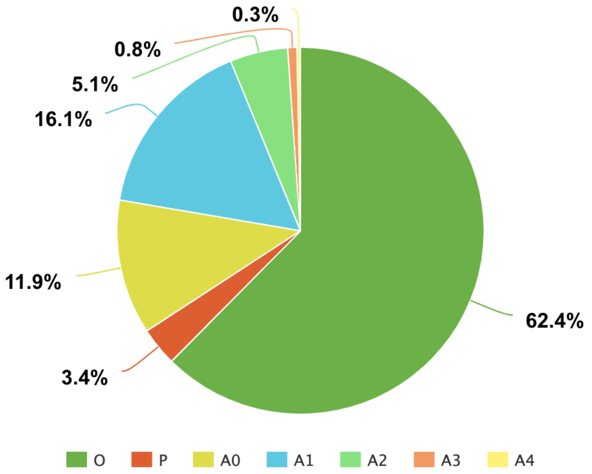


Figure 2: The distribution of token level tags (listed clockwise) in the LSOIE dataset. P denotes the extraction's predicate, A_0-A_N denote the extraction's arguments, and O denotes that a given token does not belong to the extraction.

3.1 LSOIE Conversion Process

We produce LSOIE via conversion from QA-SRL in the same manner as Stanovsky and Dagan (2016), with several important changes to adapt their method to the QA-SRL BANK 2.0.

A QA-SRL annotation for a predicate p consists of a list of questions $Q = \{q_0, \dots, q_n\}$, and a set of answer spans $A_i = \{a_{i0}, \dots, a_{in_i}\}$ for each question q_i. For each tuple $(a_0, \dots, a_k)$ in the Cartesian product $\bigtimes_i^n A_i$, we produce the extraction tuple $(a_0, p, a_1, \dots, a_k)$.

In our example extraction in Figure 1, the target predicate p is *provide*. The list of questions Q is [*Where does someone provide something?*, *Who provides something?*, *What is being provided?*] The list of arguments A is [*In Asian countries*, *physicians*, *drugs*]. The converted extraction tuple is (*physicians*, *provide*, *drugs*, *in Asian countries*).

To ensure data quality and as a result of differences between the original QA-SRL dataset and the QA-SRL BANK 2.0, we had to make two important changes to the algorithm:

Answer Filtering: The original QA-SRL dataset has a single set of mutually-exclusive answer spans for each question, written by a single annotator. In contrast, the QA-SRL BANK 2.0 has answer judgments from three annotators for each question, some providing answer sets and others marking the questions as invalid. To consolidate these, we only include questions marked as valid by all three annotators. Then, for each question, we iteratively draw the longest remaining answer span that does not overlap with a previously drawn answer span, until there are none left.

*Bats are the only mammals that can truly **fly**.* (Bats, **fly**)
*Greece moved up three to be **ranked** tenth.* (Greece, **ranked**, tenth)
*A popular student, in 1915 Mao was **elected** secretary of the Students Society.* (Mao, **elected**, secretary of the Students Society, in 1915)
*The proposed amendment already **passed** both houses in 2011.* (The proposed amendment, **passed**, both houses, in 2011)
*In polygynous species, males try to **monopolize** and mate with multiple females.* (males, **monopolize**, multiple females)
*Animals **adapted** to live in the desert are called xerocoles.* (Animals, **adapted**, to live in the desert)

Table 2: Example sentences with example extractions. Note that only one example extraction is shown here, though a sentence can yield multiple extractions.

In answer filtering, our primary motivation was to clean the raw version of crowd workers' answer responses in the QA-SRL 2.0 dataset, where questions can be posed that are not valid or the answer to them is ambiguous. We found it advantageous for dataset quality to require a strict agreement between all annotators. In choosing the longest answer span, we were motivated to not miss relevant portions of the argument, as individual crowd workers occasionally annotated a limited portion of the answer span that did not encapsulate the whole semantic meaning of the derived argument.

Argument Ordering: Stanovsky and Dagan (2016)'s original algorithm relies on the original, annotator-written order of QA-SRL questions, which may or may not produce a sensible argument ordering. Furthermore, in the QA-SRL BANK 2.0, the original order in which the questions were written is unavailable.

So, to determine argument order, we use a heuristic based on the relative order between answer spans for each question in their source text. We consider the abstract form of questions, which includes verb tense without information about its lemma. For a given question q_i in an extraction, let q_{i_x} represent the percentage of predicates in the QA-SRL BANK 2.0 where the answer span to the generalized version of q_i appears in the x^{th} place relative to other answer spans, according to the natural order of the sentence. For each argument slot in the derived extraction, the answer to the question with the highest probability q_{i_x} of naturally occurring in that slot is chosen as the argument.

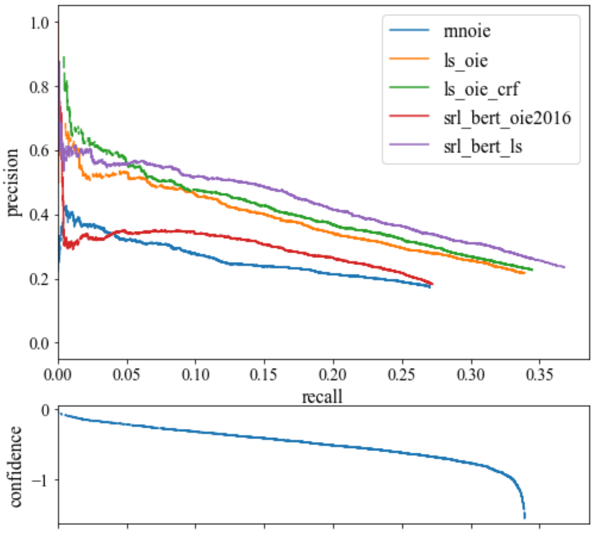


Figure 3: Top: performance of Supervised OIE systems on the LSOIE-wiki test set. Bottom: `ls_oie` estimated confidence at each extraction.

In our example extraction in Figure 1, the question *Who* `[predicate]` *something?* precedes *What is being* `[predicate]`*?* which precedes *Where does someone* `[predicate]` *something?*, enabling our algorithm to accurately extract argument ordering, which is not available from the natural ordering of the sentence or the ordering of crowd annotations in FitzGerald et al. (2018).

3.2 Dataset Statistics

We run our updated dataset conversion process over the directly crowdsourced portion of the train, development, and test partitions of the QA-SRL BANK 2.0. Stratifying the resulting data by domain, we present the new LSOIE corpus in two sections, LSOIE-wiki and LSOIE-sci. Dataset statistics are shown in Table 1. Example extractions are shown in Table 2. We provide the distribution of argument, predicate, and null tag labels in Figure 2. The LSOIE corpus expands the scope of OIE2016 and AW-OIE in size, textual diversity, and domain.

4 Benchmark Evaluation

Models: We evaluate several models on our new LSOIE dataset. Following Stanovsky et al. (2018), we model OIE as a supervised learning problem and format it as BIO tagging with tunable thresholding on extractions. We benchmark several model variants:

- **`rnnoie`** is a replication of the model in Stanovsky et al. (2018), based on a bidirectional LSTM transducer over GloVe embeddings (Pennington et al., 2014) and learned part-of-speech embedding features.

	LSOIE-wiki		LSOIE-sci	
Model	F_1	**AUC**	F_1	**AUC**
`rnnoie`	.22	.07	.26	.10
`ls_oie`	.28	.13	.33	.18
`ls_oie_crf`	.29	.14	.33	.19
`srl_bert_oie2016`	.23	.08	.29	.13
`srl_bert_ls`	**.31**	**.16**	.37	.21
`ls_oie_sci`	-	-	.34	.19
`ls_oie_crf_sci`	-	-	.35	.20
`srl_bert_ls_sci`	-	-	**.38**	**.22**

Table 3: Modeling results on the LSOIE test sets.

- **`ls_oie`** is a replication of `rnnoie` trained on LSOIE.
- **`ls_oie_crf`** is the same as `ls_oie`, but trained end-to-end with a Conditional Random Field on top to capture BIO transition constraints and trained to maximize the likelihood of the gold BIO sequence.
- **`srl_bert_ls`** is based on `ls_oie`, but uses BERT (Devlin et al., 2019) as the bidirectional encoder and the Sentence A / Sentence B embedding feature as the predicate indicator, inspired by Shi and Lin (2019).
- **`srl_bert_oie2016`** is the same architecture as `srl_bert_ls` but applied to the OIE2016 data.
- **`⋆_sci`** models were trained with the same architectures applied only to the LSOIE-sci training set.

Experiments and Evaluation: We use the AllenNLP framework (Gardner et al., 2018) built on PyTorch (Paszke et al., 2019) to implement, train, and test our models. We train `rnnoie` and `srl_bert_oie2016` on OIE2016 and `ls_oie` and `srl_bert_ls` on LSOIE-wiki. We also focus the series of models by only training on LSOIE-sci. We do not evaluate `⋆_sci` models on LSOIE-wiki. We limit our evaluation to supervised OIE systems.

We evaluate our system's performance against the gold test data in LSOIE-wiki and LSOIE-sci by considering extractions to be a match if they contain the same predicate as the gold extraction and contain the syntactic head of each gold argument. Syntactic heads are extracted with the Stanford CoreNLP dependency parser (Chen and Manning, 2014). Although it would be ideal to have the gold syntactic head, this method is preferable to taking the lexical overlap of the entire extraction

Stanovsky and Dagan (2016), ignoring argument tags and ordering as pointed out in Lechelle et al. (2019).

We then assign a confidence score to each extraction to allow for tuning the precision-recall tradeoff. For the non-CRF models, we use the mean log probability assigned to the tag labels in the extraction as the confidence score. For the CRF model, we use the log probability assigned to the entire sequence. We differ from Stanovsky et al. (2018) where confidence was calculated as the product of the inverse of the model's estimate probability for each tag label, preferring longer extractions which were more likely to get a 50% lexical match, outweighing the deficit of swimming upstream against the model's estimated confidence and still producing a downward sloping precision recall curve.

We use Viterbi decoding to extract the most likely valid BIO tagging sequence given the model's probability output for each BIO tag. We import the Viterbi algorithm functionality from the AllenNLP library (Gardner et al., 2018).

5 Discussion

Figure 3 shows precision and recall curves on the LSOIE-wiki test set, accompanied by the `ls_oie` model's estimated confidence. Table 3 shows F_1 and AUC scores for the benchmark models on the LSOIE-wiki and LSOIE-sci test sets.

The OIE modeling task is difficult. Results on both evaluation sets show that the BERT model and the CRF output layer improve over the baseline model. Training with the LSOIE improves model performance. When science is the target domain, the `*_sci` models are preferable, as they have slightly higher in-domain performance, showing the value of the domain split in LSOIE.

5.1 Error Analysis

We conduct a manual error analysis of the `ls_oie` model, where we find that our baseline models could benefit from more careful extractions.

Incorrect predicate: At minimum confidence, 53% of the model's precision errors come from verbs that are not present in the gold dataset. Half of these are legitimate predicates that are missing from the gold dataset and the other half are auxiliary verbs, that should not be present in the gold dataset. Depending on the deployment environment, the model could be improved with predicate filtering heuristics at prediction time.

Argument Concatenation: We examined 500 incorrect extractions by `ls_oie`. We found that 36% of unmatched extractions were semantically similar to the gold extraction. These extractions either concatenated arguments A_1-A_N into A_1 while gold did not, split these arguments apart while gold did, or dropped a non-material argument. For future modeling, this is an argument to drop A_2 and beyond from the dataset and only model OIE with extraction triples.

True Errors: Among the extraction errors, 2/3 involve errors in argument ordering, often following the natural order of the sentence. The other 1/3 of errors involved the model making nonsensical extractions or not making extracting arguments beyond A_0, presumably because of lack of confidence and defaulting to the O label.

LSOIE Modeling Improvements: We also manually examined 100 extractions where `ls_oie` chose the right extraction over `rnnoie`. In these cases, we found improved argument ordering, increased confidence on relevant A_1 objects, and better accuracy identifying subjects that are distant from the predicate.

6 Conclusion

In this paper, we introduced the LSOIE dataset as a resource for supervised OIE. We have algorithmically re-purposed the QA-SRL BANK 2.0 into a new OIE dataset, LSOIE, which contains over 70,000 sentences and over 150,000 extraction tuples. To benchmark the new dataset, we trained and evaluated a series of supervised OIE models, providing baselines for future research on the OIE modeling task.

The code and datasets introduced in this paper can be found at `https://github.com/Jacobsolawetz/large-scale-oie`.

Acknowledgments

A special thanks to Julian Michael and Gabriel Stanovsky for providing much needed guidance and feedback for our research. This project would not have been possible without their insight and previous progress on the topic. We also thank the anonymous reviewers for their feedback.

References

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007.

Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI)*.

Sangnie Bhardwaj, Samarth Aggarwal, and Mausam Mausam. 2019. CaRB: A crowdsourced benchmark for open IE. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Lei Cui, Furu Wei, and Ming Zhou. 2018. Neural open information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*.

Kiril Gashteovski, Rainer Gemulla, and Luciano del Corro. 2017. MinIE: Minimizing facts in open information extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kiril Gashteovski, Sebastian Wanner, Sven Hertling, Samuel Broscheit, and Rainer Gemulla. 2019. Opiec: An open information extraction corpus. In *Proceedings of the Conference on Automatic Knowledge Base Construction (AKBC)*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shengbin Jia, Yang Xiang, and Xiaojun Chen. 2018. Supervised neural models revitalize the open relation extraction. *CoRR*, abs/1809.09408.

Zhengbao Jiang, Pengcheng Yin, and Graham Neubig. 2019. Improving open information extraction via iterative rank-aware learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

William Lechelle, Fabrizio Gotti, and Phillippe Langlais. 2019. WiRe57 : A fine-grained benchmark for open information extraction. In *Proceedings of the 13th Linguistic Annotation Workshop (LAW)*.

Omer Levy, Ido Dagan, and Jacob Goldberger. 2014. Focused entailment graphs for open IE propositions. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning (CoNLL)*.

Mausam. 2016. Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguist.*, 31(1):71–106.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Dongxu Zhang, Subhabrata Mukherjee, Colin Lockard, Luna Dong, and Andrew McCallum. 2019. OpenKI: Integrating Open Information Extraction and Knowledge Bases with Relation Inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Changing the Mind of Transformers
for Topically-Controllable Language Generation

Haw-Shiuan Chang Jiaming Yuan Mohit Iyyer Andrew McCallum
CICS, University of Massachusetts Amherst
`hschang@cs.umass.edu, jiamingyuan@umass.edu,`
`{mccallum,miyyer}@cs.umass.edu`

Abstract

Large Transformer-based language models can aid human authors by suggesting plausible continuations of text written so far. However, current interactive writing assistants do not allow authors to guide text generation in desired topical directions. To address this limitation, we design a framework that displays multiple candidate upcoming topics, of which a user can select a subset to guide the generation. Our framework consists of two components: (1) a method that produces a set of candidate topics by predicting the centers of word clusters in the possible continuations, and (2) a text generation model whose output adheres to the chosen topics. The training of both components is self-supervised, using only unlabeled text. Our experiments demonstrate that our topic options are better than those of standard clustering approaches, and our framework often generates fluent sentences related to the chosen topics, as judged by automated metrics and crowdsourced workers.

1 Introduction

Recently, Transformer-based language models (LMs) have achieved impressive performance in language generation tasks (Radford et al., 2019; Dai et al., 2019) such as open-domain story generation (See et al., 2019a). When writing with the LM, users often desire an intuitive and effective way to control what a LM is going to generate (Keskar et al., 2019). To address this need, interactive writing assistants provide options to reveal possible developments of the story and generate continuations guided by the user-selected options.

Interactive writing assistants have wide applications in creative writing (Roemmele and Gordon, 2015; Clark et al., 2018; Akoury et al., 2020), education (Luo et al., 2015), and gaming (Walton, 2020). Nevertheless, the existing systems' options usually do not provide fine-grained control and/or require substantial human labor. In some prior work (Keskar et al., 2019; Tu et al., 2019), users choose among a static set of predefined attributes (e.g., sentiment) that only provide coarse-grained control. Other work (Roemmele and Gordon, 2015; Clark et al., 2018) presents users with multiple generated continuations, which requires substantial reading effort and might not contain topics that users want to see. Finally, options could be nodes in a plot graph that are handcrafted (Luo et al., 2015) or derived from a collaboration between humans and machine (Li et al., 2013), but such choices are usually limited due to the high cost of preparing the options.

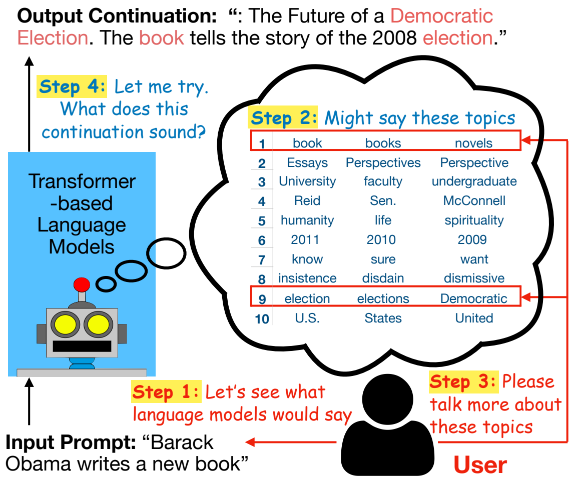


Figure 1: Given an input prompt, the Transformer-based LM provides $K = 10$ topics that might be mentioned next and each topic is represented by $M = 3$ words. The user could guide the generation process by choosing a subset of topics.

To address these limitations, we propose an interactive writing framework that provides a set of topics and guides the text generation by the user-chosen topics. The topic options are generated dynamically based on the input prompt to pro-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2601–2611
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

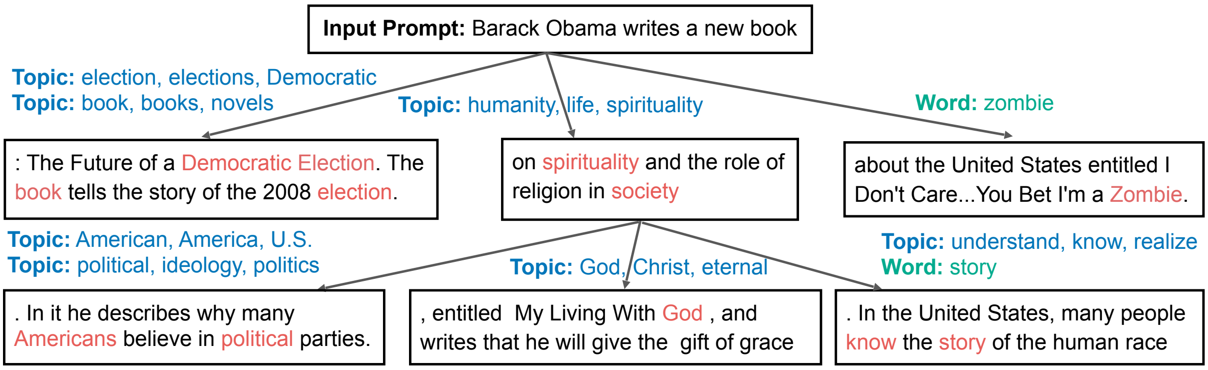


Figure 2: Examples of our generated options and continuations. We highlight the words in the continuation that are related to the chosen topics or to the specified word.

vide fine-grained control, and our models are self-supervised without the need to define the attributes or collect annotations. As depicted in Figure 1, a user can peek at the most probable K topics (shown as bags of words) appearing after the input prompt and control the generation by choosing the topics.

In Figure 2, we compare multiple generated sentences conditioned on different chosen topic(s) or specified word(s). For example, if the user chooses a topic about *humanity*, *life*, and *spirituality*, our system continues the input prompt "*Barack Obama writes a new book*" with "*on spirituality and the roles of religion in society*". Then, we can use the generated text as the new input prompt and update the set of topics to include other more relevant topics such as *God*, *Christ*, and *eternal*. The process can be repeated to create a plot tree.

A user can also control the generation by specifying word(s) if the user wants to see the words that are not in the topic list or seeks a transition to a word that is not directly related to the input prompt. For example, a user can ask our system to generate a sentence about *zombie*. Consequently, the continuation of *"Barack Obama writes a new book"* becomes *"about the United States entitled I Don't Care...You Bet I'm a Zombie"*.

The system is realized by two components: an option generator and a conditional text generator. Given a prompt, the option generator suggests a set of K topics. After a user chooses a subset of the topics and specifies some words, the embedding of every word or topic will guide the conditional text generator to produce the continuation that is both consistent with the existing prompt and relevant to the chosen topics and words.

Both components are self-supervised and use pretrained GPT2 models (Radford et al., 2019) to encode the input prompt. During training, the option generator predicts the cluster centers of future words, which are in the continuation of the prompt, based on the contextualized embeddings from GPT2. The conditional text generator fine-tunes GPT2 to predict the next words given the prompt and a few subsequent words. Since both components' input and output only come from the prompt and its continuation, training the system only requires a raw corpus, word tokenizers, and a list of stop words. This makes the proposed method suitable for open-domain story generation and easily being fine-tuned for a specific domain.

In experiments, we demonstrate that our system recommends high-quality topics and often generate sentences that follow the chosen topics. We compare our option generator with global topic models such as LDA (Blei et al., 2001) or local topic models such as clustering the words in the input prompt. The results show that the proposed method generates significantly more topics that are plausible and promote the narrative. Moreover, we compare our conditional text generator with PPLM (Plug and Play Language Models) (Dathathri et al., 2020) and demonstrate that our generation is more fluent and relevant to the chosen topics. Our code is available at `https://github.com/iesl/interactive_LM`.

2 Method

The proposed framework consists of two components: option generator and conditional text generator. In Figure 3, we illustrate the two components and their interaction. First, given the prompt $x_1, ..., x_I$ inputted by a user, the option generator at the bottom of the figure outputs K topics. After the user chooses two topics about *book* and *election* and specifies one extra word *story*, the topics

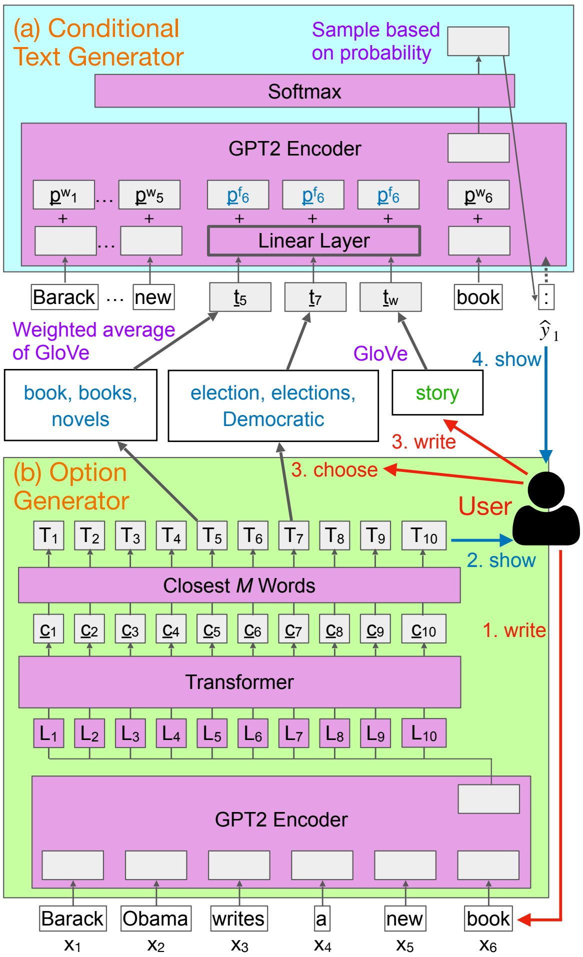


Figure 3: Our model architectures for (a) conditional text generator and (b) option generator. During testing, the information flows from the bottom to the top.

and word are passed to our text generator as the generation guidance. Accordingly, the generator continues to write the next token $\hat{y}_1$.[1]

In the following subsections, we introduce our model designs and the way to train each component. More implementation details are described in the appendix.

2.1 Option Generator

When we do not have labeled attributes in a corpus, we can create options by clustering all the words in a corpus into topics (Tu et al., 2019). The clustering could be done by topic modeling approaches such as LDA (Blei et al., 2001). The resulting topics are static (i.e., the clustering is performed globally without considering the prompt). However, the prompt might have a narrow focus and the related words of interest are all clustered into a single topic.

A simple remedy is to cluster only the words in the prompt rather than all the words in the corpus. The topics are created dynamically and locally given a prompt and can capture more fine-grained aspects in the continuations. However, the topics derived from the prompt might provide less inspiration because the users have seen the prompt. Another major drawback of the approach is that the generated topics might encourage the LM to generate repetitive sentences or make a narrative circle inside a loop.

Motivated by the challenges, we propose an option generator that predicts the cluster centers based on the prompt instead of clustering the words in the prompt during testing.

2.1.1 Model Prediction

The goal of our option generator is to predict the K cluster centers of words in the possible continuations and use the cluster centers as the topics user could choose from. As in Figure 3 (b), the option generator uses GPT2 to encode the input prompt $x_1, ..., x_I$ and passes the output embedding to K different linear layers $L_1, ..., L_K$. To model the dependency of clusters, a Transformer (Vaswani et al., 2017) takes the K embeddings as input and predicts the cluster centers $\underline{c}_1, ...\underline{c}_K$ in GloVe (Pennington et al., 2014) space. During testing, each predicted cluster center is normalized by its L2 norm, and we use the M closest words in the normalized GloVe space to represent the topic T_i, which users can choose.

We choose to learn the cluster centers in GloVe space rather than GPT2 or BERT (Devlin et al., 2019) space because the non-contextualized word embeddings are easier to visualize. Users can easily understand the meaning of a cluster center by seeing nearby words. We normalize GloVe space in this work to make the squared L2 distance equal to twice the cosine distance between two embeddings.

Our architecture is similar to the one in Chang et al. (2021), but we use a pretrained GPT2 encoder rather than train a BERT-like Transformer from scratch. Another difference is that we ignore the connection between the second Transformer and the output of GPT2 to save GPU memory for handling a longer input prompt.

---

[1] The framework is flexible. For example, the GPT2 encoders in the two components could be shared. Besides topics, the option generator could be extended to predict likely attributes in the continuation such as positive sentiment and event frames (Tu et al., 2019) if the corresponding label data are available in the training corpus.

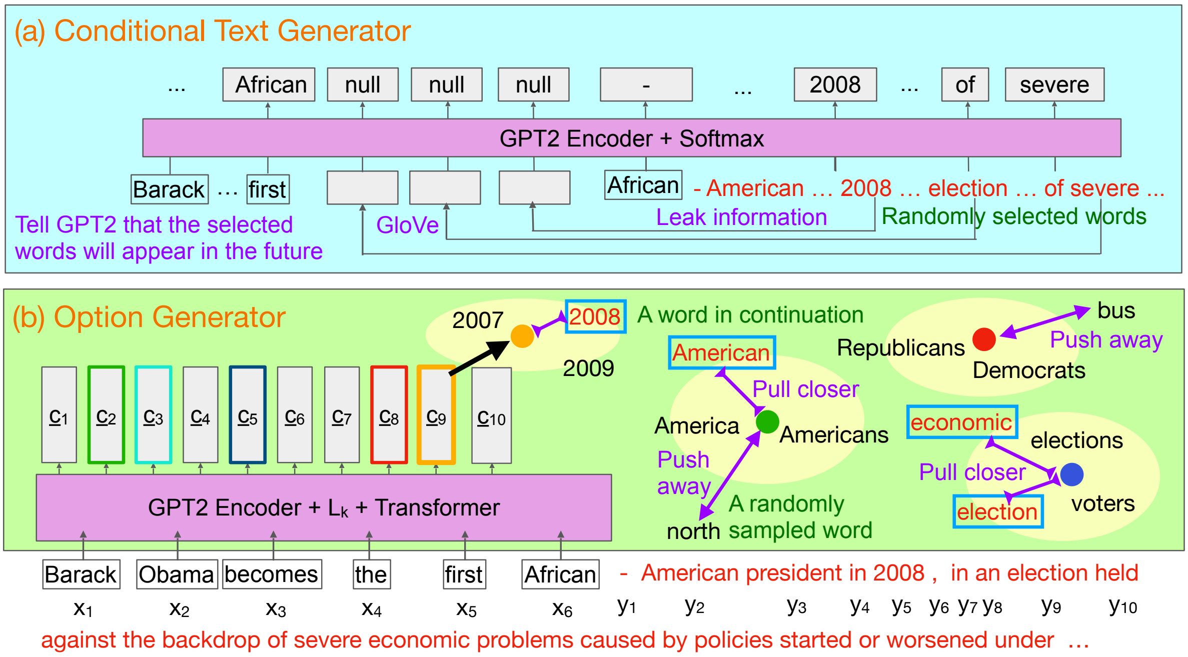


Figure 4: Training our two components using the same sentence. (a) We randomly pick $n = 3$ words in the actual continuation as our conditions for the text generator, and the null labels mean their predicted probabilities are ignored in our loss. (b) We visualize 5 out of $K = 10$ generated topics in a normalized GloVe space. Red words are the ones that appear in the continuation and pull the nearby cluster centers closer during training.

2.1.2 Model Training

In Figure 4 (b), we visualize our training procedure. For each input prompt in the training corpus, we run a forward pass through the Transformers and get predicted cluster centers $\underline{c}_1, ...\underline{c}_K$. Next, we collect 50 words in the continuation (except stop words) as positive examples and match the words with cluster centers as in the E-step of the EM algorithm (Dempster et al., 1977). We minimize the distances between the centers and their nearby positive examples by backpropagating the gradients through the matching and updating our Transformer models. Furthermore, we randomly sample some words as negative examples and maximize the distances between the cluster centers and their nearby embeddings from negative examples.

Using Figure 4 (b) as an example, the orange cluster center is pulled closer toward the embedding of *2008*, which appears in the continuation. The green cluster center is pushed away from the embedding of *north*, a randomly sampled word. Since each output embedding $\underline{c}_k$ is pulled by only the nearby embeddings of words in the continuation, the output embedding will naturally become the cluster center of the nearby continuation word embeddings. Notice that the related topics like *Democrats* and *Republicans* are not observed in the prompt and continuation, but our model can predict a red cluster center close to them because the model can learn from other similar input prompts whose continuation mentions words like *Democrats*.

Chang et al. (2021) discover that non-negative sparse coding (NNSC) (Hoyer, 2002) could encourage the Transformers to predict more diverse and relevant topics compared with Kmeans, so we adopt NNSC as our clustering loss, and its formulation could be found in Chang et al. (2021).

2.2 Conditional Text Generator

After the user chooses topic(s) or specifies word(s), each topic or word is converted to a GloVe embedding. The component aims to generate the text given the input prompt and the GloVe embeddings of the topics or words we prefer to see in the continuation.

Users only see the M words closest to the kth predicted cluster center $\underline{c}_k$ from our option generator, so we compute the kth topic embedding as

$$\underline{t}_k = \frac{\sum_{m=1}^{M} \cos(\underline{e}^w_m, \underline{c}_k)\underline{e}^w_m}{|| \sum_{m=1}^{M} \cos(\underline{e}^w_m, \underline{c}_k)\underline{e}^w_m ||}, \qquad (1)$$

where $\underline{e}^w_m$ is the normalized GloVe embedding of the mth closet word and $\cos(\underline{e}^w_m, \underline{c}_k)$ is the cosine similarities between the mth word embedding and the embedding $\underline{c}_k$.

2.2.1 Model Prediction

During testing, the topic embeddings $\underline{t}_k$ or embedding of the specified words are inserted into GPT2 encoder before x_I, the last word piece in the prompt. The inserted embeddings nudge the GPT2 to generate the sentences containing the desired words with a higher probability.

As Figure 3 (a) shows, the GloVe embeddings are first passed through a linear layer to make their dimension become the same as the hidden state size of GPT2. Then, the transformed embeddings are added with special positional embeddings $\underline{p}_I^f$, which are different from those for the prompt $\underline{p}_i^w$. The special positional embedding tells GPT2 that the inserted embeddings have a different meaning and where the conditional generation starts.

The GPT2 encoder's output goes through a softmax layer, which computes the probability of each token being observed as the first word piece in the continuation y_1. We adopt top-k sampling (Fan et al., 2018), which reduces the chance of sampling words with low probability, to pick the next word, and autoregressively sample one token $\widehat{y}_o$ at a time to generate the continuation $\widehat{y}_1, ..., \widehat{y}_O$.

2.2.2 Model Training

We train the generator using the continuation of a prompt and some randomly selected non-stop words in the continuation as its generation conditions. Since the continuation contains the randomly-selected words, the generator would be heavily penalized if it ignores the conditions by assigning low probabilities to the selected words in all the continuation positions.

An example is illustrated in Figure 4 (a). Given an input prompt in the training set, we randomly pick a number n from 0 to K and sample n words from the next $O = 25$ words (except stop words). Next, the normalized GloVe embeddings of n words are inserted to the GPT2 encoder before the last word piece in the prompt, and we ignore the output probabilities corresponding to the inserted positions during training. To speed up the training, we conduct the future word insertion in multiple positions of each training text sequence.

We insert the future words just before the text that might contain the words rather than at the beginning as in the classic seq2seq model, because we do not want the model to learn to generate the continuation based on the future topics that have not yet be specified by the users (e.g., The GPT2 should not know that it will see *election* in the future when it learns to generate *Barack Obama ...* during training).

By allowing the LM to see the upcoming words earlier, we leak partial label information to the LM input. Consequently, GPT2 learns to utilize the information and generate the sentence containing the desired words to achieve a lower perplexity loss. Notice that the training method allows us to specify our topical preference without significantly scarifying generation efficiency and fluency, but it cannot guarantee to generate all the desired topics, especially when we specify multiple ones.

One concern of the method is that the LM cannot see all possible sets of topics or words users might specify during training. Besides, each GloVe embedding used to supervise LM comes from a single word, but we ask the LM to condition on average GloVe embedding of the top M words during testing. Nevertheless, we observe that the LM is often able to generalize well in our experiments because similar words have similar GloVe embeddings, lots of training instances could be easily prepared by the self-supervised method, and our option generator usually provides the topics mentioned in the continuation in our training corpus.

3 Experiments

We evaluate two components separately, and both evaluations include automated metrics and human judgment. Throughout the evaluation, the number of topics $K = 10$ and the length of generations is 50 word pieces. We find that fixing $K = 10$ works well in our experiments. If the possible continuations cover more than 10 topics, our option generator tends to output the important topics. If they cover fewer topics, our option generator tends to output the related topics that are not explicitly mentioned in the prompt or the duplicated topics. More experiment setup details could be found in the appendix.

3.1 Datasets

We use 90% of English Wikipedia 2016 as our training set for both components, 5% as our validation set to determine the hyperparameters such as the number of epochs, and the remaining 5% as our test set to perform the automated evaluation.

For human evaluation, we collect labels from Amazon Mechanical Turk (MTurk). We randomly sample sentences from the training set of STS benchmark (STSb) (Cer et al., 2017) as our input prompts. Compared with Wikipedia, the sentences from STSb are easier to understand for annotators because a large portion of sentences in Wikipedia involves terminologies, depends on a longer context, or might even just be a list of names.

In STSb, we sample 24 sentences as our prompts, and each method generates one continuation for each input prompt. Each generated continuation or topics will be scored by three different workers.

3.2 Option Generator Evaluation

We evaluate the topics from different option generators by judging whether the topics will appear in the continuation and whether the topics would promote the narrative. The goal is to have topics that are relevant and provide new information. The topics that are too similar to the prompt words might be redundant and not helpful because the users have already seen the prompt.

3.2.1 Automatic Evaluation Metrics

- **Sim**: If the generated topics T can help users to write the continuation, the embedding of every non-stop word in the actual continuation should be similar to the embeddings of a generated topic. Thus, we compute

  $$\mathrm{Sim}(\bar{Y}, T) = \sum_{o=1}^{O'} \max_{k=1}^{K} (\underline{t}_k)^T \underline{e}_o^{\bar{y}}, \qquad (2)$$

  where $\bar{Y} = \{\bar{y}_o\}_{o=1}^{O'}$ is a set of non-stop words in the continuation and $O' = 25$. $\underline{t}_k$ is the normalized embedding of kth topic in T from equation 1 and $\underline{e}_o^{\bar{y}}$ is the oth word in $\bar{Y}$.
- **Sim Short**: When computing Sim, we use the input prompts containing around 180 words on average. To examine the topic quality at the start of writing, where the authors might need assistance the most, we also report $\mathrm{Sim}(\bar{Y}, T)$ on short input prompts (with 35 words on average).
- **Sim Diff**: The options that are helpful to users should be sufficiently different from the words in the input prompt to promote the narrative and avoid generating repeated content. Thereby, we also evaluate methods using Sim Diff = $\mathrm{Sim}(\bar{Y}, T)$ - $\mathrm{Sim}(\bar{X}, T)$, where $\bar{X} = \{\bar{x}_i\}_{i=1}^{I'}$ are the non-stop words in the input prompt.

3.2.2 Human Evaluation

Our questionnaire shows the prompt and asks which generated topics are likely to appear in a reasonable continuation and which topics promote the narrative. For each method, we report the average number of its topics that are likely to appear (L), promote the topic (TP), and both (L&TP). For example, an MTurk worker is shown three topics generated by a method given a prompt: ABC. The worker thinks A is likely to appear in the continuation and AB promote the topic. Then, L=$|\{A\}|$=1, TP=$|\{AB\}|$=2, and L&TP=$|\{A\} \cap \{AB\}|$=$|\{A\}|$=1 for this prompt.

Scope	Method	Sim	Sim Short	Sim Diff
Global	Sample	14.63	14.42	**0.16**
	LDA	36.86	36.02	-2.82
	Kmeans	**40.65**	**39.91**	-3.40
Local	Sample	41.50	41.23	-12.51
	NNSC	46.70	42.80	-15.94
	Kmeans	47.94	43.89	-16.12
	Ours	**48.38**	**46.29**	**0.45**

Table 1: Comparison of the option generators using automatic metrics. The best numbers within each scope are highlighted.

Scope	Method	L	TP	L&TP
Global	LDA	5.76 ± 0.50	6.24 ± 0.33	5.26 ± 0.31
	Kmeans	6.94 ± 0.36	6.13 ± 0.30	5.96 ± 0.31
Local	Kmeans	**8.65** ± 0.16	5.31 ± 0.50	5.14 ± 0.50
	Ours	7.85 ± 0.25	**6.96** ± 0.26	**6.75** ± 0.28

Table 2: Comparison of option generators using human judgment (mean ± standard error). L and TP refer to likelihood and topic promotion, respectively.

3.2.3 Option Generator Baselines

We compare our generator with two types of methods.[2] The first type performs the clustering globally and selects the most relevant topics to the input prompt from the static set of clusters. We cluster all the words into $J = 150$ topics by LDA (Blei et al., 2001) (**LDA-global**) and into $J = 1000$ topics by Kmeans on the normalized GloVe embedding space (Tu et al., 2019) (**Kmeans-global**). We also randomly sample K words from the whole vocabulary as our cluster centers (**Sample-global**).

Similar to equation 1, we find the M words with the closest embeddings to each cluster center to represent the topic and compute the topic embedding $\underline{t}_j$ as the weighted average embedding of M words in the jth topic. Among all J cluster centers, we pick the K topics with the closest $\underline{t}_j$ to the

---

[2] Another alternative is to generate many continuations and cluster the words in the generation. However, the method takes time, which might be prohibited by limited computational resources and the real-time interaction requirement.

Input Prompt	The study also found that skin cancer nearly tripled in Norway and Sweden since the 1950s.										
LDA-global				Kmeans-local				Ours			
1	population, households	6	company, companies	1	Norway, Sweden	6	also, however	1	research, scientific	6	1980s, 1970s
2	patients, treatment	7	Norwegian, Norway	2	tripled, doubled	7	since, Since	2	tissues, tissue	7	even, though
3	psychology, research	8	story, book	3	nearly, almost	8	Sweden, Finland	3	patients, diagnosis	8	susceptibility, pathogenic
4	police, prison	9	hospital, Hospital	4	cancer, skin	9	study, studies	4	DNA, gene	9	decreased, increased
5	chemical, carbon	10	Icelandic, Iceland	5	1950s, 1940s	10	found, discovered	5	orange, purple	10	Sweden, Norway

Table 3: Comparison of all K topics for the input prompt using $M = 2$ words closest to each topic.

Input Prompt		The study also found that skin cancer nearly tripled in Norway and Sweden since the 1950s.
Generator		Generated Text
Option	Text	
LDA-global	Ours	A study of the Norwegian police has confirmed the cancer case. The law in Norway was the subject of the
Kmeans-local	Ours	The study also found that skin cancer nearly tripled in Norway and Sweden since the 1950s. As well, skin
Ours	PPLM	In this study, a study was conducted conducted in Italy and in Finland. From the 1990s to the 1970s, there
None	GPT2	The study also revealed that only 20% of the deaths in Norway were caused by a sudden cardiac response
Ours	Ours	Recent studies have shown that melanin causes a decrease in genetic susceptibility in people in Norway,

Table 4: The continuations that are generated by conditioning on all of K topics from different option generators. The input prompt comes from STSb.

prompt embedding, where the prompt embedding is the average embedding of all words in the input prompt.

The second type of methods discovers the K topics from the input prompt. We cluster non-stop words in the prompt using non-negative sparse coding (Hoyer, 2002) (**NNSC-local**) and Kmeans (**Kmeans-local**). We also sample K non-stop words from the prompt and call it **Sample-local**. Similar to equation 1, we represent each topic using M words and compute the weighted average of their embeddings $\underline{t}_k$ as the input of our text generator. Notice that the locally clustering methods produce similar results when the prompts come from STSb due to their short lengths, so we only test **Kmeans-local** in our human evaluation.

3.2.4 Results

In Table 1, we show that local methods generate the options more relevant to the input prompt than the global methods due to significantly higher Sim and Sim Short. Our method performs better compared to other local methods, especially in Sim Diff, which highlights the high novelty of our generated topics. The improvement on Sim Short is larger than that on Sim because our method could suggest the related topics that are not explicitly mentioned in the short prompt (e.g., *U.S.* in Figure 1).

The human evaluation results are presented in Table 2. Our method wins in terms of generating relevant topics that promote the narrative. The **Kmeans-local** performs better in L because most of the words in the input prompts could be mentioned again in the next sentence. However, it often leads to the redundant topics that are too similar to the prompt.

Table 3 compares the options generated by different methods while Table 4 compares the text generated using different option generators and text generators. More examples are presented in the appendix. In Table 3, we can see that most topics in **Kmeans-local** do not promote the narrative, which makes the generated continuation become a copy of the input prompt in Table 4. We will quantitatively evaluate the generated continuations using different option generators in the appendix. Notice that the high redundancy problem is hard to be solved by a conditional text generator because the relatedness between the prompt and the generated text is hard to be controlled (See et al., 2019b).

3.3 Conditional Text Generator Evaluation

To demonstrate our text generator's effectiveness, we use our option generator to prepare the topic embeddings and randomly select n topics as our conditions to simulate the user's choice, where n is a random number from 1 to K. The sentences generated by different methods are compared.

3.3.1 Automatic Evaluation Metrics

We match the union of $M \times K$ top words in the chosen topics with the words in the generated continuations and count the number of tokens that are matched exactly (token), the number of matched word types (word), and the number of topics that contain at least one matched word (topic) to measure the relevancy between the continuations and the chosen topics. Notice that the scores are underestimated because the generation might mention words in different morphological variations or other

Text Generation Method	Automatic Metrics						Inference Time	Human Judgement		
	Relevancy Hit			Quality				Relevancy		Fluency
	Token	Word	Topic	PPL (↓)	Dist-1	Dist-2	s (↓)	Recall	Precision	Score
PPLM	1.48	0.99	0.77	18.49	**40.29**	**80.83**	17.74	30.56 ± 2.96	56.01 ± 4.41	3.83 ± 0.13
Ours	**2.36**	**1.79**	**1.40**	**16.39**	37.98	79.65	**1.02**	**41.46** ± 3.47	**56.41** ± 4.41	**4.07** ± 0.10
GPT2	1.27	0.84	0.64	14.24	39.80	80.22	1.00	24.49 ± 2.77	48.69 ± 4.61	4.15 ± 0.11

Table 5: Comparison of conditional text generators. The numbers in Dist-1, Dist-2, Recall, and Precision are percentages. Lower perplexity (PPL) and inference time are better. The better performances between PPLM and our method are highlighted. In human evaluation, we report the mean ± standard error of each method.

words related to the topics.

The fluency of the generated text is measured using the perplexity (Serban et al., 2016) of the original GPT2 (with 345M parameters) without being fine-tuned on Wikipedia. Dist-n (Li et al., 2016) is the ratio between the number of unique n-grams and the number of all n-grams in the continuations, where n=1 or 2. Higher Dist-n implies more diverse generations. The average inference time per input prompt is also presented.

3.3.2 Human Evaluation

We present the prompt and the generated continuation and ask the worker to score the generation's fluency from 1 (not fluent at all) to 5 (very fluent). Next, we show K topics and ask which topics are mentioned in the generation. Treating the worker's choices as prediction and the topics our model conditions on as ground truth, we report the average precision and recall of the prediction.

3.3.3 Conditional Text Generator Baselines

We compare our method with **PPLM** (Plug and Play Language Models) (Dathathri et al., 2020) due to its strong performance against the weighted decoding approach from Ghazvininejad et al. (2017) when the condition is a bag of words.

The condition for **PPLM** is the union of the top M words in the chosen topics and each word's weight is neglected. We use our generation model without conditioning on any word (i.e., $n = 0$) during testing[3] as the base model of **PPLM**. We also present the performance of the base model itself as a reference to know the significance of our improvement (denoted as **GPT2**).

3.3.4 Results

Table 5 indicates that our model outperforms **PPLM** in all metrics except in Dist-1 and Dist-2. We suspect that our model generates slightly less diverse sentences in order to make the generation more relevant to the given topics.

The generation might mention a topic even if it is not chosen as a condition, so we achieve similar precision compared to **PPLM** in human evaluation. The recall of **PPLM** means that only around 30% of given topics are mentioned. The low recall indicates the difficulty of mentioning multiple randomly selected topics in the next 50 word pieces while keeping the sentence fluent. By contrast, achieving 40% on recall demonstrates the effectiveness of our conditional text generator.

Compared with **PPLM**, our model requires an additional training step but achieves low inference time and high relevancy to the given topics/words once the training is finished. The benefits make it preferable in our interactive writing application.

4 Related Work

Different interactive writing assistants provide different forms of options to let users express their preferences. The options could be manually defined classes (e.g., sentiment) (Keskar et al., 2019; Dathathri et al., 2020), semantic frames (Tu et al., 2019), or event structures such as (subject, verb, object, modifier) (Martin et al., 2018; Tambwekar et al., 2019; Ammanabrolu et al., 2020). The forms of options allow users to control the attributes of the generated text but require labels or classifiers that map the text to the attributes/options.

The options could also be a single query word at the beginning (Austin, 2019), the article title (Yan, 2016), politeness (Niu and Bansal, 2018) or specificity (See et al., 2019b) of the text, or the length of the generated sentence (Tu et al., 2019). However, the options cannot provide fine-grained control on topical directions of the generated contents.

A related research direction is the multi-stage story generation. To make a long story more coherent, recent work proposes to generate a skeleton and then generate the full text guided by

---

[3] We find the model performs similarly compared with the GPT2 with no condition during training.

the skeleton. The skeleton could be a sequence of SRL frames (Fan et al., 2019), a sequence of event structure (subject, verb, object, preposition, modifier) (Ammanabrolu et al., 2020), a story premise (Fan et al., 2018), or a story summary (Chen et al., 2019). Users can revise the skeleton to control the generated text, but the approaches assume the existence of the skeleton extractor or labels in the training corpus. Besides, the systems cannot suggest options given the partial text, which is one of the main focuses of our interactive writing assistant.

The skeleton could also be multiple keyphrases. The keyphrases are extracted based on word frequency (Ippolito et al., 2019; Tan et al., 2020; Wu et al., 2020), an off-the-shelf keyword extraction method (Peng et al., 2018; Goldfarb-Tarrant et al., 2019; Yao et al., 2019; Rashkin et al., 2020; Zhang et al., 2020), a sentence compression dataset and reinforcement learning (Xu et al., 2018), or image caption datasets and ConceptNet (Lin et al., 2020). Most of the studies focus on modeling the long-term dependency among the keyphrases and/or forcing the generation to contain the keyphrases. Instead, we focus on allowing users to determine the topical directions of the generation. Compared with conditioning on keyphrases, our interactive writing assistant is especially helpful when users do not know the exact phrases they want to see or when the given keyphrase extractor does not detect the desired topics.

5 Conclusion

We propose an interactive writing assistant that generates topic options given an input prompt and generates the continuation of the prompt given the topics chosen by a user. We decompose the framework into two components and propose a novel model for each component. The automated evaluation and human evaluation indicate that our system generates many topics that are related to but different from the prompt, and generates the sentences that are fluent and relevant to the chosen topics.

Acknowledgements

We thank Ao Liu for his preliminary exploration of this project and Nader Akoury for his helpful feedbacks. We also thank the anonymous reviewers for their constructive feedback.

This work was supported in part by the Center for Data Science and the Center for Intelligent Information Retrieval, in part by the Chan Zuckerberg Initiative under the project Scientific Knowledge Base Construction, in part using high performance computing equipment obtained under a grant from the Collaborative R&D Fund managed by the Massachusetts Technology Collaborative, in part by the National Science Foundation (NSF) grant numbers DMR-1534431 and IIS-1514053.

Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, Online. Association for Computational Linguistics.

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and Mark O. Riedl. 2020. Story realization: Expanding plot events into sentences. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7375–7382. AAAI Press.

John Austin. 2019. The book of endless history: Authorial use of GPT2 for interactive storytelling. In *Interactive Storytelling - 12th International Conference on Interactive Digital Storytelling, ICIDS 2019, Little Cottonwood Canyon, UT, USA, November 19-22, 2019, Proceedings*, volume 11869, pages 429–432. Springer.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2001. Latent dirichlet allocation. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 601–608. MIT Press.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Haw-Shiuan Chang, Amol Agrawal, and Andrew McCallum. 2021. Extending multi-sense word embedding to phrases and sentences for unsupervised semantic applications. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Gang Chen, Yang Liu, Huanbo Luan, Meng Zhang, Qun Liu, and Maosong Sun. 2019. Learning to predict explainable plots for neural story generation. *arXiv preprint arXiv:1912.02395*.

Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A Smith. 2018. Creative writing with a machine in the loop: Case studies on slogans and stories. In *23rd International Conference on Intelligent User Interfaces*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

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

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 89–97, Minneapolis, Minnesota. Association for Computational Linguistics.

Patrik O Hoyer. 2002. Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*.

Daphne Ippolito, David Grangier, Chris Callison-Burch, and Douglas Eck. 2019. Unsupervised hierarchical story infilling. In *Proceedings of the First Workshop on Narrative Understanding*, pages 37–43, Minneapolis, Minnesota. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Linbo Luo, Wentong Cai, Suiping Zhou, Michael Lees, and Haiyan Yin. 2015. A review of interactive narrative systems and technologies: a training perspective. *Simulation*, 91(2):126–147.

Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 868–875. AAAI Press.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Melissa Roemmele and Andrew S. Gordon. 2015. Creative help: A story writing assistant. In *Interactive Storytelling - 8th International Conference on Interactive Digital Storytelling, ICIDS 2015, Copenhagen, Denmark, November 30 - December 4, 2015, Proceedings*, volume 9445, pages 81–92. Springer.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019a. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 843–861, Hong Kong, China. Association for Computational Linguistics.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019b. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1702–1723, Minneapolis, Minnesota. Association for Computational Linguistics.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5982–5988. ijcai.org.

Bowen Tan, Zichao Yang, Maruan AI-Shedivat, Eric P Xing, and Zhiting Hu. 2020. Progressive generation of long text. *arXiv preprint arXiv:2006.15720*.

Lifu Tu, Xiaoan Ding, Dong Yu, and Kevin Gimpel. 2019. Generating diverse story continuations with controllable semantics. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 44–58, Hong Kong. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Nick Walton. 2020. AI dungeon.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, et al. 2020. A controllable model of grounded response generation. *arXiv preprint arXiv:2005.00613*.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4306–4315, Brussels, Belgium. Association for Computational Linguistics.

Rui Yan. 2016. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2238–2244. IJCAI/AAAI Press.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7378–7385. AAAI Press.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020. POINTER: constrained progressive text generation via insertion-based generative pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8649–8670. Association for Computational Linguistics.

Unsupervised Abstractive Summarization of Bengali Text Documents

Radia Rayan Chowdhury[*]
Ahsanullah Univ of Science & Tech
Dhaka, Bangladesh
`radiarayan.rrc@gmail.com`

Mir Tafseer Nayeem[*]
Ahsanullah Univ of Science & Tech
Dhaka, Bangladesh
`mir.nayeem@alumni.uleth.ca`

Tahsin Tasnim Mim
Ahsanullah Univ of Science & Tech
Dhaka, Bangladesh
`tahsintasnimmim@gmail.com`

Md. Saifur Rahman Chowdhury
Ahsanullah Univ of Science & Tech
Dhaka, Bangladesh
`saif.chowdhury1997@gmail.com`

Taufiqul Jannat
Ahsanullah Univ of Science & Tech
Dhaka, Bangladesh
`taufiquljannat@gmail.com`

Abstract

Abstractive summarization systems generally rely on large collections of document-summary pairs. However, the performance of abstractive systems remains a challenge due to the unavailability of the parallel data for low-resource languages like Bengali. To overcome this problem, we propose a graph-based unsupervised abstractive summarization system in the single-document setting for Bengali text documents, which requires only a Part-Of-Speech (**POS**) tagger and a pre-trained language model trained on Bengali texts. We also provide a human-annotated dataset with document-summary pairs to evaluate our abstractive model and to support the comparison of future abstractive summarization systems of the Bengali Language. We conduct experiments on this dataset and compare our system with several well-established unsupervised extractive summarization systems. Our unsupervised abstractive summarization model outperforms the baselines without being exposed to any human-annotated reference summaries.[1]

1 Introduction

The process of shortening a large text document with the most relevant information of the source is known as automatic text summarization. A good summary should be coherent, non-redundant, and grammatically readable while retaining the original document's most important contents (Nenkova and McKeown, 2012; Nayeem et al., 2018). There are two types of summarizations: extractive and abstractive. Extractive summarization is about ranking important sentences from the original text. The abstractive method generates human-like sentences using natural language generation techniques. Traditionally used abstractive techniques are sentence compression, syntactic reorganization, sentence fusion, and lexical paraphrasing (Lin and Ng, 2019). Compared to extractive, abstractive summary generation is indeed a challenging task.

A cluster of sentences uses multi-sentence compression (**MSC**) to summarize into one single sentence originally called sentence fusion (Barzilay and McKeown, 2005; Nayeem and Chali, 2017b). The success of neural sequence-to-sequence (**seq2seq**) models with attention (Bahdanau et al., 2015; Luong et al., 2015) provides an effective way for text generation which has been extensively applied in the case of abstractive summarization of English language documents (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016; Miao and Blunsom, 2016; Paulus et al., 2018; Nayeem et al., 2019). These models are usually trained with lots of gold summaries, but there is no large-scale human-annotated abstractive summaries available for low-resource language like Bengali. In contrast, the unsupervised approach reduces the human effort and cost for collecting and annotating large amount of paired training data. Therefore, we choose to create an effective Bengali Text Summarizer with an unsupervised approach. The summary of our contributions:

- To the best of our knowledge, our **Ben**gali Text **Summ**arization model (**BenSumm**) is

---

[*] Equal contribution. Listed by alphabetical order.

[1] We make our code & dataset publicly available at `https://github.com/tafseer-nayeem/BengaliSummarization` for reproduciblity.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2612–2619
April 19 - 23, 2021.

the very first unsupervised model to generate abstractive summary from Bengali text documents while being simple yet robust.

- We also introduce a highly abstractive dataset with document-summary pairs to evaluate our model, which is written by professional summary writers of National Curriculum and Textbook Board (**NCTB**).[2]
- We design an unsupervised abstractive sentence generation model that performs sentence fusion on Bengali texts. Our model requires only **POS** tagger and a pre-trained language model, which is easily reproducible.

2 Related works

Many researchers have worked on text summarization and introduced different extractive and abstractive methods. Nevertheless, very few attempts have been made for Bengali Text summarization despite Bangla being the 7^{th} most spoken language.[3] Das and Bandyopadhyay (2010) developed Bengali opinion based text summarizer using given topic which can determine the information on sentiments of the original texts. Haque et al. (2017, 2015) worked on extractive Bengali text summarization using pronoun replacement, sentence ranking with term frequency, numerical figures, and overlapping of title words with the document sentences. Unfortunately, the methods are limited to extractive summarization, which ranks some important sentences from the document instead of generating new sentences which is challenging for an extremely low resource language like Bengali. Moreover, there is no human-annotated dataset to compare abstractive summarization methods of this language.

Jing and McKeown (2000) worked on Sentence Compression (**SC**) which has received considerable attention in the NLP community. Potential utility for extractive text summarization made **SC** very popular for single or multi-document summarization (Nenkova and McKeown, 2012). TextRank (Mihalcea and Tarau, 2004) and LexRank (Erkan and Radev, 2004) are graph-based methods for extracting important sentences from a document. Clarke and Lapata (2008); Filippova (2010) showed a first intermediate step towards abstractive summarization, which compresses original sentences for a summary generation. The Word-Graph based approaches were first proposed by (Filippova, 2010), which require only a POS tagger and a list of stopwords. Boudin and Morin (2013) improved Filippova's approach by re-ranking the compression paths according to keyphrases, which resulted in more informative sentences. Nayeem et al. (2018) developed an unsupervised abstractive summarization system that jointly performs sentence fusion and paraphrasing.

3 BenSumm Model

We here describe each of the steps involved in our **Ben**gali Unsupervised Abstractive Text **Summ**arization model (**BenSumm**) for single document setting. Our preprocessing step includes tokenization, removal of stopwords, Part-Of-Speech (**POS**) tagging, and filtering of punctuation marks. We use the **NLTK**[4] and **BNLP**[5] to preprocess each sentence and obtain a more accurate representation of the information.

3.1 Sentence Clustering

The clustering step allows us to group similar sentences from a given document. This step is critical to ensure good coverage of the whole document and avoid redundancy by selecting at most one sentence from each cluster (Nayeem and Chali, 2017a). The Term Frequency-Inverse Document Frequency (**TF-IDF**) measure does not work well (Aggarwal and Zhai, 2012). Therefore, we calculate the cosine similarity between the sentence vectors obtained from **ULMfit** pre-trained language model (Howard and Ruder, 2018). We use hierarchical agglomerative clustering with the ward's method (Murtagh and Legendre, 2014). There will be a minimum of 2 and a maximum of $n-1$ clusters. Here, n denotes the number of sentences in the document. We measure the number of clusters for a given document using the silhouette value. The clusters are highly coherent as it has to contain sentences similar to every other sentence in the same cluster even if the clusters are small. The following formula can measure silhouette Score:

$$\text{Silhouette Score} = \frac{(x-y)}{max(x,y)} \tag{1}$$

where y denotes mean distance to the other instances of intra-cluster and x is the mean distance to the instances of the next closest cluster.

---

[2] `http://www.nctb.gov.bd/`

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

[4] `https://www.nltk.org`

[5] `https://bnlp.readthedocs.io/en/latest/`

3.2 Word Graph (WG) Construction

Textual graphs to generate abstractive summaries provide effective results (Ganesan et al., 2010). We chose to build an abstractive summarizer with a sentence fusion technique by generating word graphs (Filippova, 2010; Boudin and Morin, 2013) for the Bengali Language. This method is entirely unsupervised and needs only a **POS** tagger, which is highly suitable for the low-resource setting. Given a cluster of related sentences, we construct a word-graph following (Filippova, 2010; Boudin and Morin, 2013). Let, a set of related sentences S = $\{s_1, s_2, ..., s_n\}$, we construct a graph $G = (V, E)$ by iteratively adding sentences to it. The words are represented as vertices along with the parts-of-speech (**POS**) tags. Directed edges are formed by connecting the adjacent words from the sentences. After the first sentence is added to the graph as word nodes (punctuation included), words from the other related sentences are mapped onto a node in the graph with the same **POS** tag. Each sentence of the cluster is connected to a dummy start and end node to mark the beginning and ending sentences. After constructing the word-graph, we can generate M-shortest paths from the dummy start node to the end node in the word graph (see Figure 1).

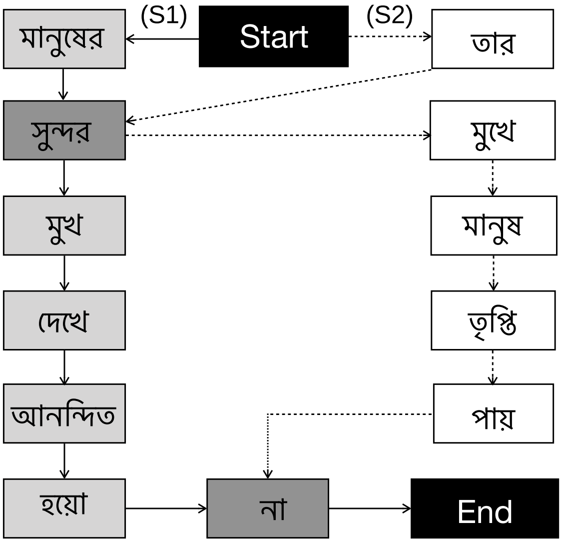
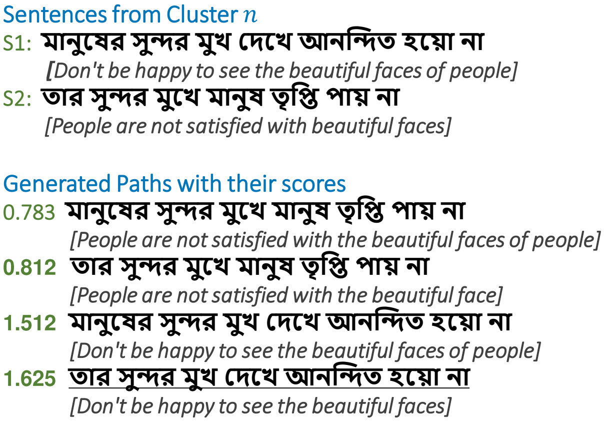


Figure 1: Sample WG of two related sentences.

Figure 2 presents two sentences, which is one of the source document clusters, and the possible paths with their weighted values are generated using the word-graph approach. Figure 1 illustrates an example **WG** for these two sentences.



Sentences from Cluster n

S1: মানুষের সুন্দর মুখ দেখে আনন্দিত হয়ো না

[Don't be happy to see the beautiful faces of people]

S2: তার সুন্দর মুখে মানুষ তৃপ্তি পায় না

[People are not satisfied with beautiful faces]

Generated Paths with their scores

0.783 মানুষের সুন্দর মুখে মানুষ তৃপ্তি পায় না

[People are not satisfied with the beautiful faces of people]

0.812 তার সুন্দর মুখে মানুষ তৃপ্তি পায় না

[People are not satisfied with the beautiful face]

1.512 মানুষের সুন্দর মুখ দেখে আনন্দিত হয়ো না

[Don't be happy to see the beautiful faces of people]

1.625 তার সুন্দর মুখ দেখে আনন্দিত হয়ো না

[Don't be happy to see the beautiful faces]

Figure 2: Output of WG given two related sentences. The underlined sentence is the top-ranked sentence to be included in the final summary.

After constructing clusters given a document, a word-graph is created for each cluster to get abstractive fusions from these related sentences. We get multiple weighted sentences (see Figure 2) form the clusters using the ranking strategy (Boudin and Morin, 2013). We take the top-ranked sentence from each cluster to present the summary. We generate the final summary by merging all the top-ranked sentences. The overall process is presented in Figure 3. We also present a detailed illustration of our framework with an example source document in the Appendix.

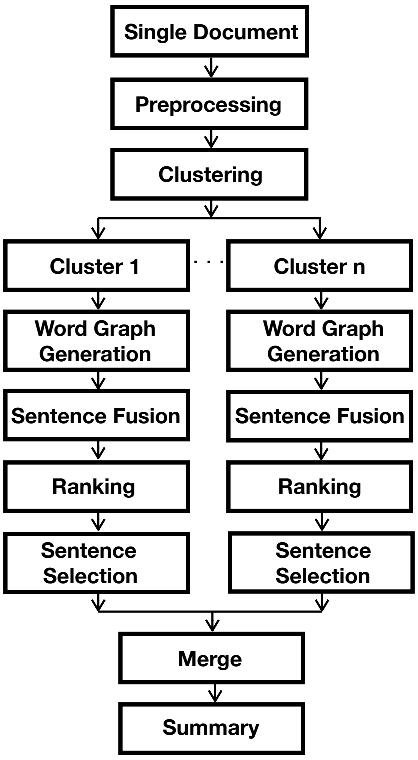
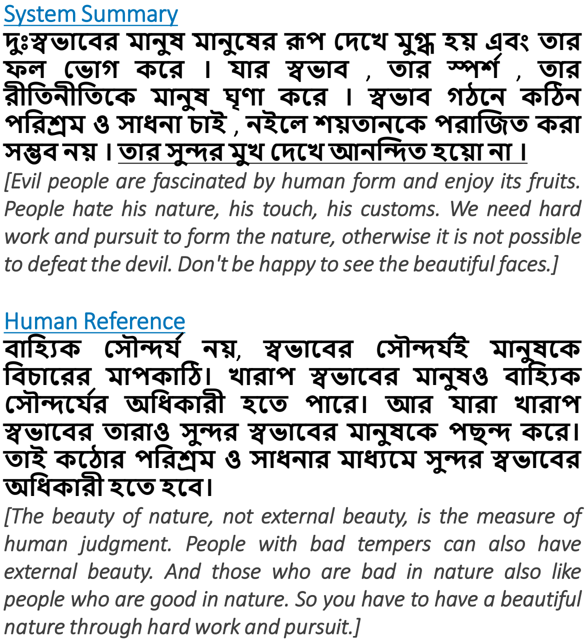


Figure 3: Overview of our **BenSumm** model.



System Summary

দুঃস্বভাবের মানুষ মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে । যার স্বভাব , তার স্পর্শ , তার রীতিনীতিকে মানুষ ঘৃণা করে । স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই , নইলে শয়তানকে পরাজিত করা সম্ভব নয় । তার সুন্দর মুখ দেখে আনন্দিত হয়ো না ।

[Evil people are fascinated by human form and enjoy its fruits. People hate his nature, his touch, his customs. We need hard work and pursuit to form the nature, otherwise it is not possible to defeat the devil. Don't be happy to see the beautiful faces.]

Human Reference

বাহ্যিক সৌন্দর্য নয়, স্বভাবের সৌন্দর্যই মানুষকে বিচারের মাপকাঠি। খারাপ স্বভাবের মানুষও বাহ্যিক সৌন্দর্যের অধিকারী হতে পারে। আর যারা খারাপ স্বভাবের তারাও সুন্দর স্বভাবের মানুষকে পছন্দ করে। তাই কঠোর পরিশ্রম ও সাধনার মাধ্যমে সুন্দর স্বভাবের অধিকারী হতে হবে।

[The beauty of nature, not external beauty, is the measure of human judgment. People with bad tempers can also have external beauty. And those who are bad in nature also like people who are good in nature. So you have to have a beautiful nature through hard work and pursuit.]

Figure 4: Example output of our **BenSumm** model with English translations.

4 Experiments

This section presents our experimental details for assessing the performance of the proposed **BenSumm** model.

Dataset We conduct experiments on our dataset which consists of *139* samples of human-written abstractive document-summary pairs written by professional summary writers of the National Curriculum and Textbook Board (**NCTB**). The NCTB is responsible for the development of the curriculum and distribution of textbooks. The majority of Bangladeshi schools follow these books.[6] We collected the human written document-summary pairs from the several printed copy of NCTB books. The overall statistics of the datasets are presented in Table 1. From the dataset, we measure the *copy rate* between the source document and the human summaries. It's clearly visible from the table that our dataset is highly abstractive and will serve as a robust benchmark for this task's future works. Moreover, to provide our proposed framework's effectiveness, we also experiment with an extractive dataset **BNLPC**[7] (Haque et al., 2015). We remove the abstractive sentence fusion part to compare with the baselines for the extractive evaluation.

	NCTB [Abstractive]	**BNLPC [Extractive]**
Total #Samples	139	200
Source Document Length	91.33	150.75
Human Reference Length	36.23	67.06
Summary Copy Rate	**27%**	**99%**

Table 1: Statistics of the datasets used for our experiment. Length is expressed as Avg. #tokens.

NCTB [Abstractive]	**R-1**	**R-2**	**R-L**
Random Baseline	9.43	1.45	9.08
GreedyKL	10.01	1.84	9.46
LexRank	10.65	1.78	10.04
TextRank	10.69	1.62	9.98
SumBasic	10.57	1.85	10.09
BenSumm [Abs] (*ours*)	**12.17**	**1.92**	**11.35**
BNLPC [Extractive]	**R-1**	**R-2**	**R-L**
Random Baseline	35.57	28.56	35.04
GreedyKL	48.85	43.80	48.55
LexRank	45.73	39.37	45.17
TextRank	60.81	**56.46**	60.58
SumBasic	35.51	26.58	34.72
BenSumm [Ext] (*ours*)	**61.62**	55.97	**61.09**

Table 2: Results on our **NCTB** Dataset and **BNLPC**.

Automatic Evaluation We evaluate our system (**BenSumm**) using an automatic evaluation metric ROUGE F1 (Lin, 2004) without any limit of words.[8] We extract 3-best sentences from our system and the systems we compare as baselines. We report unigram and bigram overlap (ROUGE-1 and ROUGE-2) to measure informativeness and the longest common subsequence (ROUGE-L) to measure the summaries' fluency. Since ROUGE computes scores based on the lexical overlap at the surface level, there is no difference in implementation for summary evaluation of the Bengali language.

Baseline Systems We compare our system with various well established baseline systems like **LexRank** (Erkan and Radev, 2004), **TextRank** (Mihalcea and Tarau, 2004), **GreedyKL** (Haghighi and Vanderwende, 2009), and **SumBasic** (Nenkova and Vanderwende, 2005). We use an open source implementation[9] of these summarizers and adapted it for Bengali language. It is important to note that these summarizers are completely extractive and

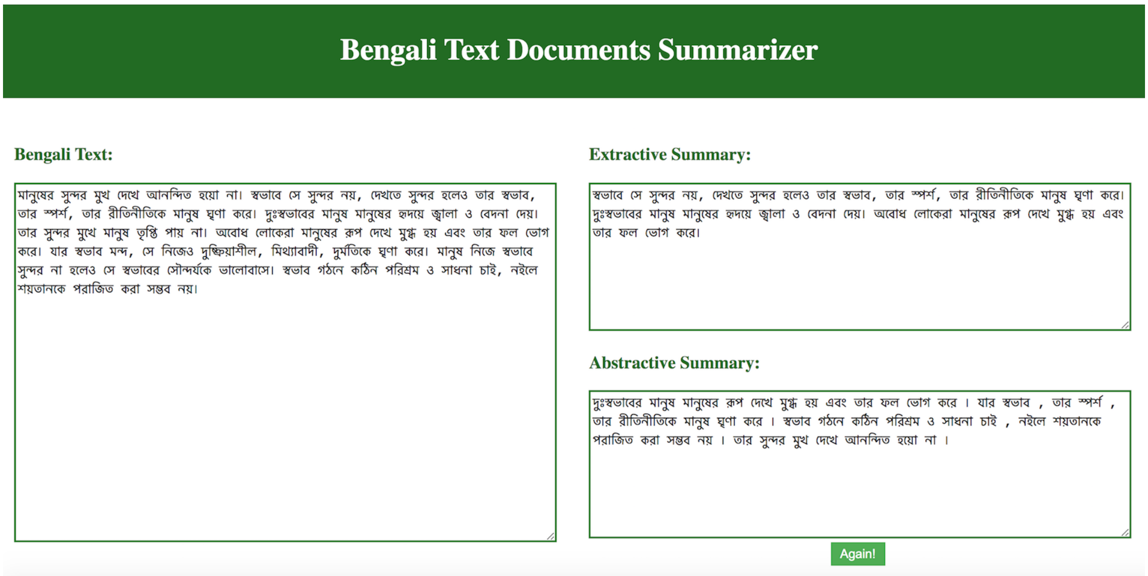
---

[6] `https://w.wiki/ZwJ`

[7] `http://www.bnlpc.org/research.php`

[8] `https://git.io/JUhq6`

[9] `https://git.io/JUhq1`



Bengali Text Documents Summarizer

Bengali Text:

মানুষের সুন্দর মুখ দেখে আনন্দিত হয়ো না। স্বভাবে সে সুন্দর নয়, দেখতে সুন্দর হলেও তার স্বভাব, তার স্পর্শ, তার রীতিনীতিকে মানুষ ঘৃণা করে। দুঃস্বভাবের মানুষ মানুষের হৃদয়ে জ্বালা ও বেদনা দেয়। তার সুন্দর মুখে মানুষ তৃপ্তি পায় না। অবোধ লোকেরা মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে। যার স্বভাব মন্দ, সে নিজেও দুষ্ক্রিয়াশীল, মিথ্যাবাদী, দুর্মতিকে ঘৃণা করে। মানুষ নিজে স্বভাবে সুন্দর না হলেও সে স্বভাবের সৌন্দর্যকে ভালোবাসে। স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই, নইলে শয়তানকে পরাজিত করা সম্ভব নয়।

Extractive Summary:

স্বভাবে সে সুন্দর নয়, দেখতে সুন্দর হলেও তার স্বভাব, তার স্পর্শ, তার রীতিনীতিকে মানুষ ঘৃণা করে। দুঃস্বভাবের মানুষ মানুষের হৃদয়ে জ্বালা ও বেদনা দেয়। অবোধ লোকেরা মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে।

Abstractive Summary:

দুঃস্বভাবের মানুষ মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে । যার স্বভাব , তার স্পর্শ , তার রীতিনীতিকে মানুষ ঘৃণা করে । স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই , নইলে শয়তানকে পরাজিত করা সম্ভব নয় । তার সুন্দর মুখ দেখে আনন্দিত হয়ো না ।

Figure 5: Interface of our Bengali Document Summarization tool. For an input document D with N sentences, our tool can provide both extractive and abstractive summary for the given document. The translations of both the document and summary are provided in the Appendix (see Figure 6).

designed for English language. On the other hand, our model is unsupervised and abstractive.

Results We report our model's performance compared with the baselines in terms of F1 scores of R-1, R-2, and R-L in Table 2. According to Table 2, our abstractive summarization model outperforms all the extractive baselines in terms of all the ROUGE metrics even though the dataset itself is highly abstractive (reference summary contains almost 73% new words). Moreover, we compare our extractive version of our model **BenSumm** without the sentence fusion component. We get better scores in terms of R1 and RL compared to the baselines. Finally, we present an example of our model output in Figure 4. Moreover, We design a Bengali Document Summarization tool (see Figure 5) capable of providing both extractive and abtractive summary for an input document.[10]

Human Evaluation Though ROUGE (Lin, 2004) has been shown to correlate well with human judgments, it is biased towards surface level lexical similarities, and this makes it inappropriate for the evaluation of abstractive summaries. Therefore, we assign three different evaluators to rate each summary generated from our abstractive system (*BenSumm [Abs]*) considering three different aspects, i.e., Content, Readability, and Overall Quality. They have evaluated each system generated summary with scores ranges from 1 to 5, where 1 represents very poor performance, and 5 represents very good performance. Here, content means how well the summary can convey the original input document's meaning, and readability represents the grammatical correction and the overall summary sentence coherence. We get an average score of **4.41**, **3.95**, and **4.2** in content, readability, and overall quality respectively.

5 Conclusion and Future Work

In this paper, we have developed an unsupervised abstractive text summarization system for Bengali text documents. We have implemented a graph-based model to fuse multiple related sentences, requiring only a **POS** tagger and a pre-trained language model. Experimental results on our proposed dataset demonstrate the superiority of our approach against strong extractive baselines. We design a Bengali Document Summarization tool to provide both extractive and abstractive summary of a given document. One of the limitations of our model is that it cannot generate new words. In the future, we would like to jointly model multi-sentence compression and paraphrasing in our system.

Acknowledgments

We want to thank all the anonymous reviewers for their thoughtful comments and constructive suggestions for future improvements to this work.

---

[10] Video demonstration of our tool can be accessed from `https://youtu.be/LrnskktiXcg`

References

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. In *Mining text data*, pages 77–128. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Comput. Linguist.*, 31(3):297–328.

Florian Boudin and Emmanuel Morin. 2013. Keyphrase extraction for n-best reranking in multi-sentence compression. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 298–305, Atlanta, Georgia. Association for Computational Linguistics.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, San Diego, California. Association for Computational Linguistics.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression an integer linear programming approach. *J. Artif. Int. Res.*, 31(1):399–429.

Amitava Das and Sivaji Bandyopadhyay. 2010. Topic-based Bengali opinion summarization. In *Coling 2010: Posters*, pages 232–240, Beijing, China. Coling 2010 Organizing Committee.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 322–330, Beijing, China. Coling 2010 Organizing Committee.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, Colorado. Association for Computational Linguistics.

Md Haque, Suraiya Pervin, Zerina Begum, et al. 2017. An innovative approach of bangla text summarization by introducing pronoun replacement and improved sentence ranking. *Journal of Information Processing Systems*, 13(4).

Md Majharul Haque, Suraiya Pervin, and Zerina Begum. 2015. Automatic bengali news documents summarization by introducing sentence frequency and clustering. In *2015 18th International Conference on Computer and Information Technology (ICCIT)*, pages 156–160. IEEE.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, page 178–185, USA. Association for Computational Linguistics.

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

Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9815–9822. AAAI Press.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Yishu Miao and Phil Blunsom. 2016. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 319–328, Austin, Texas. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Fionn Murtagh and Pierre Legendre. 2014. Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion? *J. Classif.*, 31(3):274–295.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

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

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

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

Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. 2019. Neural diverse abstractive sentence compression generation. In *Advances in Information Retrieval - 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proceedings, Part II*, volume 11438 of *Lecture Notes in Computer Science*, pages 109–116. Springer.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

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

A Appendix

A detailed illustration of our **BenSumm** model with outputs from each step for a sample input document is presented in Figure 6.

Input Document

মানুষের সুন্দর মুখ দেখে আনন্দিত হয়ো না। স্বভাবে সে সুন্দর নয়, দেখতে সুন্দর হলেও তার স্বভাব, তার স্পর্শ, তার রীতিনীতিকে মানুষ ঘৃণা করে। দুঃস্বভাবের মানুষ মানুষের হৃদয়ে জ্বালা ও বেদনা দেয়। তার সুন্দর মুখে মানুষ তৃপ্তি পায় না। অবোধ লোকেরা মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে। যার স্বভাব মন্দ, সে নিজেও দুষ্ক্রিয়াশীল, মিথ্যাবাদী, দুর্মতিকে ঘৃণা করে। মানুষ নিজে স্বভাবে সুন্দর না হলেও সে স্বভাবের সৌন্দর্যকে ভালোবাসে। স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই, নইলে শয়তানকে পরাজিত করা সম্ভব নয়।

[Do not be happy to see the beautiful faces of people. He/She is not beautiful by nature, although he/she is beautiful to look at, people hate his/her nature, touch, and manners. People with bad temper irritate and hurt people's hearts. People are not satisfied with the beautiful face. Ignorant people are fascinated by the human form and suffer in the long run. The one whose nature is evil, he is mischievous, a liar, and evil. Man himself is not beautiful by nature, but he loves the beauty of people's nature. We need hard work and pursuit to form nature; otherwise, it is impossible to defeat the devil.]

Sentence Clustering

Cluster #1:
Sentence #1: দুঃস্বভাবের মানুষ মানুষের হৃদয়ে জ্বালা ও বেদনা দেয়।
Sentence #2: অবোধ লোকেরা মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে।
Cluster #2:
Sentence #1: স্বভাবে সে সুন্দর নয়, দেখতে সুন্দর হলেও তার স্বভাব, তার স্পর্শ, তার রীতিনীতিকে মানুষ ঘৃণা করে।
Sentence #2: যার স্বভাব মন্দ, সে নিজেও দুষ্ক্রিয়াশীল, মিথ্যাবাদী, দুর্মতিকে ঘৃণা করে।
Sentence #3: মানুষ নিজে স্বভাবে সুন্দর না হলেও সে স্বভাবের সৌন্দর্যকে ভালোবাসে।
Cluster #3:
Sentence #1: স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই, নইলে শয়তানকে পরাজিত করা সম্ভব নয়।
Cluster #4:
Sentence #1: মানুষের সুন্দর মুখ দেখে আনন্দিত হয়ো না।
Sentence #2: তার সুন্দর মুখে মানুষ তৃপ্তি পায় না।

Sentence Fusion & Ranking

Cluster #1:
দুঃস্বভাবের মানুষ মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে ।
Cluster #2:
যার স্বভাব , তার স্পর্শ , তার রীতিনীতিকে মানুষ ঘৃণা করে ।
Cluster #3:
স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই , নইলে শয়তানকে পরাজিত করা সম্ভব নয় ।
Cluster #4:
তার সুন্দর মুখ দেখে আনন্দিত হয়ো না ।

Final Summary

দুঃস্বভাবের মানুষ মানুষের রূপ দেখে মুগ্ধ হয় এবং তার ফল ভোগ করে । যার স্বভাব , তার স্পর্শ , তার রীতিনীতিকে মানুষ ঘৃণা করে । স্বভাব গঠনে কঠিন পরিশ্রম ও সাধনা চাই , নইলে শয়তানকে পরাজিত করা সম্ভব নয় । তার সুন্দর মুখ দেখে আনন্দিত হয়ো না ।

[Evil people are fascinated by human form and enjoy its fruits. People hate his nature, his touch, and his customs. We need hard work and pursuit to form the nature, otherwise it is not possible to defeat the devil. Don't be happy to see the beautiful faces.]

Figure 6: A detailed illustration with outputs from each step of our Bengali Abstractive Summarization model for a sample input document.

From Toxicity in Online Comments to Incivility in American News: Proceed with Caution

Anushree Hede **Oshin Agarwal** **Linda Lu** **Diana C. Mutz** **Ani Nenkova**
University of Pennsylvania
`{anuhede,oagarwal,lulinda,nenkova}@seas.upenn.edu`
`mutz@upenn.edu`

Abstract

The ability to quantify incivility online, in news and in congressional debates, is of great interest to political scientists. Computational tools for detecting online incivility for English are now fairly accessible and potentially could be applied more broadly. We test the Jigsaw Perspective API for its ability to detect the degree of incivility on a corpus that we developed, consisting of manual annotations of civility in American news. We demonstrate that toxicity models, as exemplified by Perspective, are inadequate for the analysis of incivility in news. We carry out error analysis that points to the need to develop methods to remove spurious correlations between words often mentioned in the news, especially identity descriptors and incivility. Without such improvements, applying Perspective or similar models on news is likely to lead to wrong conclusions, that are not aligned with the human perception of incivility.

1 Introduction

Surveys of public opinion report that most Americans think that the tone and nature of political debate in this country have become more negative and less respectful and that the heated rhetoric by politicians raises the risk for violence (Center, 2019). These observations motivate the need to study (in)civility in political discourse in all spheres of interaction, including online (Ziegele et al., 2018; Jaidka et al., 2019), in congressional debates (Uslaner, 2000) and as presented in news (Meltzer, 2015; Rowe, 2015). Accurate automated means for coding incivility could facilitate this research, and political scientists have already turned to using off-the-shelf computational tools for studying civility (Frimer and Skitka, 2018; Jaidka et al., 2019; Theocharis et al., 2020).

Computational tools however, have been developed for different purposes, focusing on detecting language in online forums that violate community norms. The goal of these applications is to support human moderators by promptly focusing their attention on likely problematic posts. When studying civility in political discourse, it is primarily of interest to characterize the overall civility of interactions in a given source (i.e., news programs) or domain (i.e., congressional debates), as an average over a period of interest. Applying off-the-shelf tools for toxicity detection is appealingly convenient, but such use has not been validated for any domain, while uses in support of moderation efforts have been validated only for online comments.

We examine the feasibility of quantifying incivility in the news via the Jigsaw Perspective API, which has been trained on over a million online comments rated for toxicity and deployed in several scenarios to support moderator effort online[1]. We collect human judgments of the (in)civility in one month worth of three American news programs. We show that while people perceive significant differences between the three programs, Perspective cannot reliably distinguish between the levels of incivility as manifested in these news sources.

We then turn to diagnose the reasons for Perspective's failure. Incivility is more subtle and nuanced than toxicity, which includes identity slurs, profanity, and threats of violence along other unacceptable incivility. In the range of civil to borderline civil human judgments, Perspective gives noisy predictions that are not indicative of the differences in civility perceived by people. This finding alone suggests that averaging Perspective scores to characterize a source is unlikely to yield meaningful results. To pinpoint some of the sources of the noise in predictions, we characterize individual words as likely triggers of errors in Perspective or sub-error triggers that lead to over-prediction of toxicity.

---

[1] `https://www.perspectiveapi.com/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2620–2630
April 19 - 23, 2021.

We discover notable anomalies, where words quite typical in neutral news reporting are confounded with incivility in the news domain. We also discover that the mention of many identities, such as Black, gay, Muslim, feminist, etc., triggers high incivility predictions. This occurs despite the fact that Perspective has been modified specifically to minimize such associations (Dixon et al., 2018a). Our findings echo results from gender debiasing of word representations, where bias is removed as measured by a fixed definition but remains present when probed differently (Gonen and Goldberg, 2019). This common error—treating the mention of identity as evidence for incivility—is problematic when the goal is to analyze American political discourse, which is very much marked by us-vs-them identity framing of discussions. These findings will serve as a basis for future work in debiasing systems for incivility prediction, while the dataset of incivility in American news will support computational work on this new task.

Our work has implications for researchers of language technology and political science alike. For those developing automated methods for quantifying incivility, we pinpoint two aspects that require improvement in future work: detecting triggers of civility overprediction and devising methods to mitigate the errors in prediction. We propose an approach for a data-driven detection of error triggers; devising mitigation approaches remain an open problem. For those seeking to contrast civility in different sources, we provide compelling evidence that state-of-the-art automated tools are not appropriate for this task. The data and (in)civility ratings would be of use to both groups as test data for future models for civility prediction[2].

2 Related Work

2.1 Datasets for Incivility Detection

Incivility detection is a well-established task, though it is not well-standardized, with the degree and type of incivility varying across datasets.

Hate speech, defined as speech that targets social groups with the intent to cause harm, is arguably the most widely studied form of incivility detection, largely due to the practical need to moderate online discussions. Many Twitter datasets have been collected, of racist and sexist tweets (Waseem and Hovy, 2016), of hateful and offensive tweets (Davidson et al., 2017), and of hateful, abusive, and spam tweets (Founta et al., 2018). Another category of incivility detection that more closely aligns with our work is toxicity prediction. Hua et al. (2018) collected a dataset for toxicity identification in online comments on Wikipedia talk pages. They defined toxicity as comments that are rude, disrespectful, or otherwise likely to make someone leave a discussion. All these datasets are built for social media platforms using either online comments or tweets. We work with American news. To verify if Perspective can reproduce human judgments of civility in this domain, we collect a corpus of news segments annotated for civility.

2.2 Bias in Incivility Detection

Models trained on the datasets described above associate the presence of certain descriptors of people with incivility Hua et al. (2018). This bias can be explained by the distribution of words in incivility datasets and the fact that systems are not capable of using context to disambiguate between civil and uncivil uses of the word, instead associating the word with the dominant usage in the training data.

To mitigate this bias, Jigsaw's Perspective API was updated, and model cards (Mitchell et al., 2019) were released for the system to show how well the system is able to predict toxicity when certain identity words are mentioned in a text. Simple templates such as "I am <IDENTITY>" were used to measure the toxicity associated with identity words. More recently, many more incorrect associations with toxicity were discovered. Prabhakaran et al. (2019) found that Perspective returned a higher toxicity score when certain names are mentioned, and Hutchinson et al. (2020) found that this was also the case for words/phrases representing disability. "I am a blind person" had a significantly higher toxicity score than "I am a tall person". We show that when measured with different templates, the bias that was mitigated in Perspective still manifests. Further, we propose a way to establish a reference set of words and then find words associated with markedly higher toxicity than the reference. This approach reveals a larger set of words which do not lead to errors but trigger uncommonly elevated predictions of toxicity in the lower ranges of the toxicity scale.

Waseem and Hovy (2016) found that the most common words in sexist and racist tweets in their corpus are "not, sexist, #mkr, women, kat" and

---

[2] Available at https://github.com/anushreehede/incivility_in_news

"islam, muslims, muslim, not, mohammed". Any system trained on this dataset would likely learn these correlations with the frequent words as well, and computational methods to prevent this are yet to be developed.

Studies have shown that African American English is more likely to be labeled as toxic by both annotators and systems due to differences in dialect (Sap et al., 2019; Xia et al., 2020; Davidson et al., 2019). Our work does not involve AAE but does reveal how words quite common in news reporting can be erroneously associated with incivility.

3 Data Collection for Incivility in News

We study the following American programs: PBS NewsHour, MSNBC's The Rachel Maddow Show and Hannity from FOX News. For brevity, we use the network to refer to each source: PBS, MSNBC, and FOX, in the following discussions. These sources are roughly representative of the political spectrum in American politics, with NewsHour being Left-Center and MSNBC and FOX with strong left and right bias respectively. These are generally one-hour shows, with about 45 minutes of content when commercial breaks are excluded.

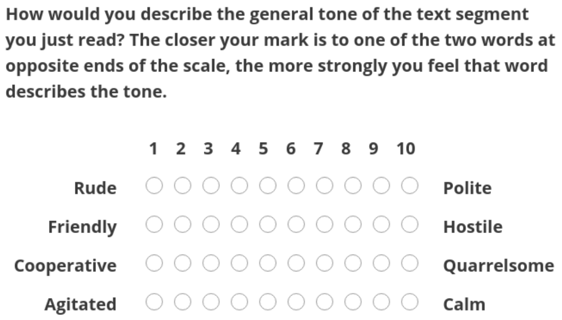
The transcripts, which provide speaker labels and turns, are from February 2019. We take only the days when all three shows aired, and we had transcripts, for a total of 51 transcripts.

We analyze the programs on two levels with the help of paid research assistants paid $15 per hour. All of them are undergraduate students in non-technical majors at the University of Pennsylvania. In Pass I, we ask a single rater to read through a transcript and identify speaker turns that appear particularly uncivil or notably civil. We characterize the programs according to the number of uncivil segments identified in each transcript.[3]

After that, in Pass II, a research assistant chose a larger snippet of about 200 words that includes the selection identified as notably uncivil (and respectively civil), providing more context for the initially selected speaker turns. We also selected several snippets of similar length at random but not overlapping with the civil and uncivil snippets. Each of these snippets, 219 it total, was then rated for perceived civility by two raters, neither of whom participated in the initial selection of content. The raters did not know why a snippet that they were rating was selected. We choose to annotate such snippets, corresponding to about a minute of show content, to ensure sufficient context is available to make meaningful judgments about the overall tone of presentation. Table 1 gives an overview of number of Uncivil, Civil and Random speaker turns around which longer snippets were selected for fine-grained annotation of incivility. The largest number of civil turns was found in PBS and most uncivil turns were identified in FOX.

Show	U	C	R	Overall	Avg Len	Vocab
FOX	81	12	23	116	201.0	3960
MSNBC	13	23	8	44	209.6	1763
PBS	11	31	17	59	216.4	2627
Overall	105	66	48	219	206.9	5962

Table 1: Number of Pass II snippets, separated by show and Pass I class: U(ncivil), C(ivil) and R(andom), along with average length of the snippets in words and the size of the vocabulary (unique words).



How would you describe the general tone of the text segment you just read? The closer your mark is to one of the two words at opposite ends of the scale, the more strongly you feel that word describes the tone.

	1	2	3	4	5	6	7	8	9	10	
Rude	○	○	○	○	○	○	○	○	○	○	Polite
Friendly	○	○	○	○	○	○	○	○	○	○	Hostile
Cooperative	○	○	○	○	○	○	○	○	○	○	Quarrelsome
Agitated	○	○	○	○	○	○	○	○	○	○	Calm

Figure 1: Annotation interface

Snippets were presented for annotation in batches of 15. Each batch had a mix of snippets from each of the programs, displayed in random order. The annotators read each snippet and rated it on a 10 point scale in four dimensions used in prior work to assess civility in news (Mutz and Reeves, 2005): Polite/Rude, Friendly/Hostile, Cooperative/Quarrelsome and Calm/Agitated. The dimensions appeared in random order and alternated which end of the dimension appeared on the left and which on the right, prompting raters to be more thoughtful in their selection and making it difficult to simply select the same value for all. A screenshot of the interface is shown in Figure 1.

The composite civility score is then obtained by first reversing the ratings to account for the alter-

---

[3] For this first analysis of incivility, we pre-selected uncivil segments, to ensure that they are well represented in the data. Incivility is rare, so a random sample of snippets will contain considerably fewer clearly uncivil snippets. We are currently augmenting the dataset with data from later months, with segments for annotation drawn from randomly selected timestamps, giving a more representative sample of the typical content that appears in the show.

nations of the ends for each dimension, such that all ratings result in small ratings for the civil ends of the scale (Polite, Friendly, Cooperative, Calm) and high values for the uncivil ends of the scales. The four scores were averaged for each annotator. Finally, the scores for the two annotators are averaged to obtain a civility score for each snippet, ranging between 1 and 10, where 1 is the most civil and 10 is the most uncivil possible.

4 Annotator Agreement

Here we briefly discuss the annotator agreement on the perceived incivility. We characterize the agreement on the transcript level in terms of the civil and uncivil speaker turns flagged in Pass I and on the text snippet level in terms of correlations and absolute difference in the scores assigned by a pair of raters in Pass II.

Pass I selection of turns was made by one person, but we are still able to glean some insights about the validity of their selection from analyzing the ratings of Pass II snippets. The Pass I selection can be construed as a justification for the score assigned in Pass II, similar to other tasks in which a rationale for a prediction has to be provided (DeYoung et al., 2020). Figure 2 shows the distribution of scores for the 200-word snippets that were selected around the initial 40-50-word speaker turns deemed notably civil, uncivil or that were randomly chosen. The distribution is as expected, with segments including uncivil turns almost uniformly rated as uncivil, with civility score greater than 5. According to our scale, a score of 5 would correspond to borderline civil on all dimensions or highly uncivil on at least one. Only three of the 105 snippets selected around an uncivil turn got scores a bit under 5: the remaining snippets including a rationale for incivility, were rated as uncivil with high consistency by the independent raters in Pass II.

In Pass II, each pair of annotators rated a total of 37 or 36 segments. Correlations between the composite ratings are overall high, ranging from 0.86 to 0.65 per pair. For only one of the six pairs, the correlation is below 0.75. The absolute difference in composite incivility scores by the two independent annotators is about 1 point.

Overall, the consistency between initial selection from the transcript and the independent ratings of civility, as well as the correlation between civility ratings of two raters for the snippets, are very good.

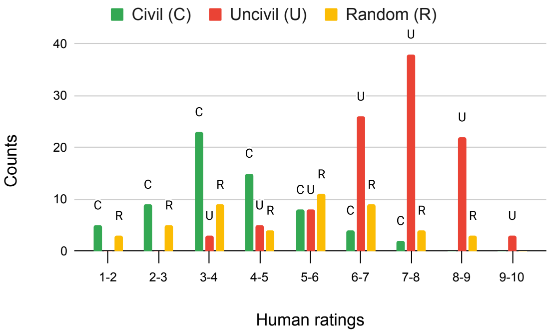


Figure 2: Distribution of snippet civility ratings (obtained in Pass II) by rationale type (obtained in Pass I). The snippets deemed to contain uncivil during Pass I are consistently rated as highly uncivil in Pass II.

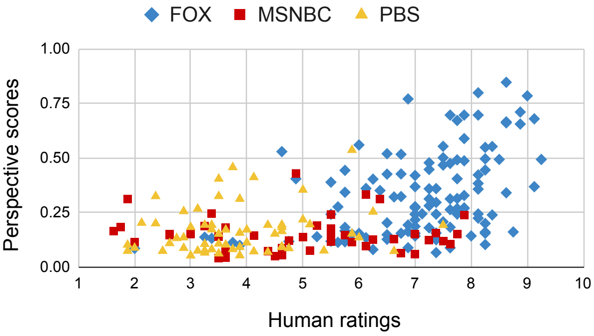


Figure 3: Scatterplot of human and Perspective scores

5 Predicting Incivility in American News

With the corpus of real-valued civility in the news, we can now compare sources according to the perception of people with those provided by Perspective. We perform the analysis on the transcript level and the snippet level. On the transcript level, we use the number of uncivil utterances marked in Pass I as the indicator of incivility. For the snippet level, we use the average composite civility ratings by the raters in Pass II. For automatic analysis, we obtain Perspective scores for each speaker turn as marked in the transcript and count the number of turns predicted as having toxicity 0.5 or greater. For snippets, we obtain Perspective scores directly, combining multiple speaker turns.

Figure 3 gives the scatter plot of civility scores per segment, from raters and Perspective. The plot reveals that Perspective is not sensitive enough to detect any differences in levels of incivility for human rating lower than six. For the higher levels of incivility, Perspective scores also increase and have better differentiation. However, the snippets from MSNBC rated as uncivil by people receive low scores. We verified that these segments are indeed uncivil, but in a sarcastic, indirect way. Portions of the turns can be seen in Table 2.

There is only one snippet with Perspective toxicity score over 0.5 for the civil to borderline civil segments from the news shows; this indicates it has good precision for binary identification of civil content. Perspective's performance in detecting binary incivility (for snippets with ratings greater than 5) is mixed. The recall for incivility is not that good, with some of these snippets receiving low toxicity scores. The trend of increasing Perspective score with increasing human-rated incivility is observed mostly on segments from FOX. The incivility in FOX appears to be more similar to that seen in online forums, with name-calling and labeling of individuals. Some examples can be seen in Table 3. The more subtle, snarky comments from MSNBC are not detected as toxic by Perspective.

However, when characterizing shows by incivility, detecting utterances that may be problematic is not of much interest. The goal is to characterize the show (daily, weekly, monthly) overall. For this, we inspect the average incivility per show on the transcript and segment level (see Table 4). On both granularities, people perceive a statistically significant difference in civility between each pair of shows, with FOX perceived as the most uncivil, followed by MSNBC and PBS as the most civil.

On the transcript level, the presence of incivility in PBS is not statistically significant. The raters chose 0.29 (fewer than one) uncivil utterances from PBS from all shows on the 17 days we study, compared with 4.5 per show for FOX. The 95% confidence interval for the mean for uncivil utterances per show covers zero for PBS, so it is not statistically significant. The lower end of the 95% confidence intervals of the mean transcript-level incivility in FOX and MSNBC is greater than zero, indicating consistent incivility in these programs.

The segment-level analysis of civility ratings reveals the same trend, with a one-point difference between PBS and MSNBC and two points between MSNBC and FOX. All of these differences are statistically significant at the 0.01 level, according to a two-sided Mann-Whitney test.

The automated analyses paint a different picture. On the transcript level, FOX is overall the most uncivil, with about 6 speaker turns per predicted to be toxic, with a Perspective score greater than 0.5. PBS appears to be the next in levels of incivility, with more than one toxic turn per transcript. For both of these shows, incivility is over-predicted, and many of the segments predicted as uncivil are civil according to the human ratings. MSNBC is predicted to have fewer than one toxic turns per transcript, under-detecting incivility. On the segment level, FOX is again assessed as the most uncivil, and PBS again appears to be more uncivil than MSNBC. On the segment level, the differences are statistically significant. Perspective incorrectly characterizes PBS as significantly more uncivil than MSNBC.

The correlation between human and Perspective incivility scores is 0.29 on the transcript and 0.51 on the segment level. Overall for broadcast news, Perspective cannot reproduce the incivility perception of people. In addition to the inability to detect sarcasm/snark, there seems to be a problem with over-prediction of the incivility in PBS and FOX.

In the next section, we seek to establish some of the drivers of over-prediction errors, characterizing individual words as possible triggers of absolute or relative over-prediction of incivility.

6 Incivility Over-prediction

Prior work has drawn attention to the fact that certain words describing people are incorrectly interpreted as triggers of incivility by Perspective, leading to errors in which a perfectly acceptable text segment containing the words would be predicted as toxic (Dixon et al., 2018b; Hutchinson et al., 2020). Their analysis, similar to other work on bias in word representations, starts with a small list of about 50 words to be analyzed in an attempt to find toxicity over-prediction triggers.

In our work, we seek to apply the same reasoning with the same goals, but in a more data-driven manner, without having to commit to a very small list of words for analysis. Given our text domain of interest (news) and the desiderata to characterize sources rather than individual text segments, we also find sub-errors, or words that do not lead to errors in toxicity prediction but have much higher than average toxicity associated with them compared to other words.

Ideally, we would like to test the full vocabulary of a new domain for (sub-)error triggers, but methods for doing so do not exist and may not be practical when the vocabulary is too large. For this reason, we sample words in a way informed by the target application, choosing words that con-

Snippets from MSNBC	Human	Perspective
... They say that he is also going to declare the national emergency, but with this presidency, honestly, is that actually going to happen? We don't know. The White House says it's going to happen, but in this presidency, is anything really done and dusted before Fox & Friends says it's done and dusted?	7.50	0.12
... Well, he better take a few moments, I know he doesn't read, but perhaps someone could read Article I Section 7 and 8 of the Constitution. The power of appropriation lies with the Congress, not with the president. If he were trying to do this, he is basically establishing an imperial president, his majesty...	7.75	0.15
proposing fireworks for the Fourth of July. Even in D.C., it's a bold idea from the president today. Presumably this will be followed by an executive order proclaiming that from here on out, we're going to start a whole new calendar year every year on the first day of January. What? Also, he's going to declare that we're going to start a new American pastime, a ball game where one person holds a stick and that person runs around a series of bases to come home if the person with the stick hits the ball well enough and hard enough, a lot of people try to catch it and prevent the batter from rounding the bases and get home. The president will soon announce a name for that and announce that he has invented this game. Also, he's invented rap music and the idea of taking a vacation in the summer if you're a school kid. I mean, I kid you not, the president proposed to reporters from the White House in all seriousness from the White House that he's thinking there should maybe be fireworks and a parade on the Fourth of July in Washington, D.C. It could catch on. It could become a tradition.	7.63	0.10

Table 2: Snippets from MSNBC rated as highly uncivil by humans but with low toxicity score from Perspective. Human rating are between 1-10 and Perspective scores between 0-1.

Snippets from FOX	Human	Perspective
Meanwhile, this garbage deal that I've been telling you about, out of Washington, which allocates a measly additional $1.375 billion for wall construction – well, that has not been signed by the president. As a matter of fact, nobody in the White House has seen any language in the bill, none. So, let's be frank. It's not nearly enough money. Washington lawmakers, the swamp creatures, they have once again let we, the people, the American people down, this is a swamp compromise. Now, let me make predictions. Based on what the president is saying publicly, I love the press love to speculate, how did Hannity get his information? All right. We know this is a president that keeps his promises. And he goes at the speed of Trump. And if you watch every speech, if you actually listen to Donald Trump's words, you know, especially you people in the hate Trump media, you psychos, he telegraphs. He's saying exactly what he's planning to do. You're too stupid to listen.	8.13	0.80
The Democratic party with no plan to make your life better. More safe, more secure, more prosperous for not only you but for your children and you grandchildren. They are fueled too by a rage, hate and obsession with victimhood and identity politics, entitlement, pushing doom and gloom day and night, climate hysteria, indoctrinating your kids into believing the world is ending in 12 years. Just to push the socialist agenda. A party fueled by what can only be described as psychotic, literally unhinged, rage and hatred every second, minute and hour of the day against the duly elected president. They are not even trying to hide it. Just today we learned that shifty Adam Schiff, former MSNBC legal analyst, collusion conspiracy theorist Daniel Goldman to leave his new witch hunt against the President.	9.00	0.78

Table 3: Examples of snippets from Fox, rated with human incivility ratings and Perspective.

Show	Count	Human Avg	Human 95% CI	Perspective Avg	Perspective 95% CI
Transcript Level					
FOX	17	4.53	[2.62, 6.44]	6.18	[3.86, 8.49]
MSNBC	17	1.24	[0.40, 2.08]	0.29	[-0.01, 0.6]
PBS	17	0.29	[-0.06, 0.65]	1.41	[0.6, 2.23]
Snippet Level					
FOX	116	7.09	[6.85, 7.33]	0.33	[0.3, 0.37]
MSNBC	44	4.97	[4.43, 6.00]	0.15	[0.12, 0.17]
PBS	59	3.87	[3.56, 4.18]	0.17	[0.14, 0.19]

Table 4: Statistics for human incivility ratings and Perspective scores. Count is the number of transcripts and the number of snippets in each level of analysis.

tribute the most to the average incivility score of one on the news sources compared to another of the sources. We sample a thousand words from each show. Occasionally, the same word is sampled for more than one show, so the final list for detailed analysis consists of 2,671 words.

6.1 Template-Derived Word Toxicity

Then we proceed similarly to prior work, to fill short templates in which only one word differs, and a fixed slot is filled in with each word in turn. We use five templates, chosen in an ad-hoc way similar to prior work. We do not reuse templates from prior work because they are tailored for the analysis of identity words, while we would like any word from the vocabulary to be an acceptable completion of the template. The average toxicity of templates filled in with a given word now provides us with a word-specific incivility score by which we can compare the full set of selected words.

We fill in five short templates with each word in turn, to obtain a comparable incivility score for each word. The first two were chosen to be neutral; the third one contained a word with a possibly negative connotation but used in a neutral sense, the fourth contains negation, and the last one is a question, to account for a wider variety of possible contexts in which the incivility triggers may occur. The five templates we use are:

(i) We wrote WORD on the page.
(ii) They whispered WORD to themselves.
(iii) I erased the word WORD from the page.
(iv) You didn't write WORD.
(v) Did he say WORD?

We use Perspective to predict the toxicity of each template when WORD is substituted with each of the words in our list for analysis. The average and maximum of these toxicities serve as a useful characterization of the words.

The templates were chosen so that the full sentence is not offensive, that is neither of the examples, even when filling it with an offensive word should be interpreted as deliberately rude or uncivil without any additional context. Higher values of predicted incivility would implicate the word substituted in the placeholder as the perceived trigger of incivility. We wish to find words where this association is wrong.

6.2 Offensive Words

At the same time, without reference or knowledge of the template word-scores, we classified the words as offensive or not. Two of the authors classified words independently and then adjudicated the disagreements in a conversation. Ultimately, there were 65 words we judged as offensive out of the full list of 2,671 words.

Separating offensive words from the rest is helpful. Using these words, we can get a sense of the degree to which Perspective incorporates context to make the prediction and then exclude them from further analysis. When templates are filled in with offensive words, the templates present hard cases for prediction. For these cases, context interpretation is necessary to make the correct prediction; word-spotting without incorporating context is likely to lead to an error. The ability to do this is important for our application: in the news, it is acceptable to report on someone else's incivility and their uncivil statements. If this is done with the purpose of expressing disapproval of the incivility, the reporting itself is not uncivil.

Furthermore, the set of offensive words allows us to quantify the degree to which the template scores justify our choice of rules to classify non-offensive words as error and sub-error triggers. We will consider words to be error triggers if at least one template was judged by Perspective to have toxicity greater than 0.5. Sub-error triggers are words for which all five templates had toxicity lower than 0.5, but their average toxicity was markedly higher than that for other words.

The average template toxicity for offensive words is 0.48, compared to 0.11 for the 2,606 non-offensive words in the list we analyzed. Of the 65 offensive words, 54% had at least one template with toxicity greater than 0.5. Perspective clearly is not simply word-spotting to make the prediction. It produces toxicity scores below 0.5 for about half of the offensive words. For the other half, however, it often produces an error.

In addition, 35% of the offensive words met the criteria for sub-error trigger. Overall, 89% of the offensive words meet either the error triggers or sub-error triggers criteria, confirming that these ranges of toxicity are the appropriate ones in which we should focus our attention in search of words that may have an unusually high association with toxicity. Example offensive words that met the sub-error criteria are: bozo, cheater, Crazy, crock, deplorables, F-ing, hoax, insane, mad, etc.

Other words, which we deemed non-offensive, have a template profile similar to that of the vast majority of offensive words. They are ones for which Perspective over-predicts toxicity.

7 Error Trigger Words for Perspective

We consider a word to be an error trigger if at least one of the templates has a toxicity score of 0.5 or greater from Perspective. Below is the full list of error trigger words from the news shows. We informally organize them into categories to facilitate inspection.

Identity African-Americans, anti-Semites, anti-white, anti-women, BlacKkKlansman, feminism, Feminist, gay, Homophobia, Homophobic, homophobic, Islam, Islamic, Jew, LGBT, Muslim, women

Violence and Sex annihilation, assassinated, beheaded, die, kill, killed, killing, murdered, shooting; intercourse, pedophiles, pornography, prostitution, rape, raped, rapist, rapists, sex, sexist, Sexual, sexual, sexually, sodomized

Ambiguity Dick, dirty, garbage, rats

Informal dopest, farting

Other brain, counter-intelligence, blackface, hypocrisy

Many error triggers are identity words describing people. Many of these identities, like gay and Muslim, were known triggers of incorrect toxicity prediction, and Perspective was specially altered to address this problem (Dixon et al., 2018a). Clearly, however, they are still a source of error, and the approaches for bias mitigation have not been fully successful. As we mentioned in the introduction, a system that is unstable in terms of its predictions when identity words appear in a text is not suitable for analysis of political discourse, where identities are mentioned often and where in many cases, it is of interest to quantify the civility with which people talk about different identity groups.

The second large class of error triggers consists of words related to death and sex. In online comments, toxic statements are often threats of sexual

0, 115-pound, abortion, abuse, abused, abusively, abysmal, accosted, adult, Africa, age-old, aliens, America, Americans, anti-semite, Anti-Trump, anti-Trump, Aryan, assaulted, assaulting, attack, baby, bad, barrier, Basically, beaten, belly, Black, black, Blow, blowing, bomber, bottom, bouncer, British, Brooks, brown, bunch, caliphate, capacity, cares, Catholic, Catholics, chicken, chief, child, children, China, Chinese, chock-full, Christian, church, Clinton, conforming, Content, COOPER, country, Covington, cows, crackers, crawling, creatures, cries, crime, crimes, criminal, CROSSTALK, cruelty, crying, DANIELS, dare, daughter, death, decrepit, defy, dehumanizing, Democrat, Democrats, demonize, denigrate, destroy-, died, Dingell, Dinkins, disrespectful, Donald, doomed, drug, drugs, Drunk, ducking, Duke, dumping, eggs, epidemic, European, evil, exist, explode, exploit, extremist-related, face, fake, Fallon, firearm, fithy, folks, Foreign, Former, FRANCIS, fraud, fry, gag, gagged, gang, gender, girls, governor, gun, guy, guy's, guys, handgun, harassed, harboring, hate, hate-, hate-Trump, hatred, head, heads, heartless, Hebrew, Hegel, her, herein, heroin, Hillary, HIV, horrors, hts, human, hush, Hymie, ifs, illness, imperialists, impugning, inaudible, inconclusive, infanticide, infuriating, inhumane, intelligence, interracial, invaders, Iran, Iraqi, ISIS, Islamophobic, Israel, Israelites, jail, Juanita, Kaine, Karine, kid, kids, Klan, Korea, laid, LAUGHTER, lie, lied, lies, life-and-death, life-death, limbs, litig, lying, MADDOW, MAGA-inspired, males, mama, man, men, mental, military, minors, missile, mock, mockery, molested, mouth, muck, N-, n't, NAACP, nation-state, needless, newscasters, Nonproliferation, nose, nuke, Obama, Obama's, obscene, obsessions, obsessive-compulsive, obsolete, organ, outrageous, ovations, Oversight, oxygen, p.m., painful, Pakistan, patriarchal, people, person, police, pope, President, president, president's, pretty, priest, priests, prison, prog, punched, punches, Putin, Putin's, queer, racial, Racism, racism, Rage, ranted, relations, religion, religious, relitigating, remove, REP., Republican, Republicans, RUSH, Russian, Russians, S, Saudi, savagely, self-confessed, self-defining, self-proclaimed, semitic, she, SHOW, sick, slavery, sleet, slurs, smear, socialist, son, Spartacus, stick, stop, stunning, supporters, supremacist, swamp, Syria, tampering, terror, terrorism, terrorists, thrash, threat, throat, tirade, toddler, TRANSCRIPT, trashing, treasonous, Trump, tumor, U.K., U.S, U.S., U.S.-backed, undress, unsuccessful, unvetted, upstate, vandalized, Vatican, Venezuela, Venezuelans, videotaped, videotaping, violated, violence, violent, VIRGINIA, virulent, voters, War, war, weird, welcome, Whitaker, White, white, WITH, woman, worse, worth, xenophobic, Yemen, you, your, yourself

Table 5: Sub-error trigger Words. The list comprises many identity words, words with negative connotations, words describing controversial topics, and words related to violence.

violence and death. Perhaps this is why words broadly related semantically are associated with toxicity. These words, however, can appear in news contexts without any incivility. Reports of violence at home and abroad are a mainstay of news, as well as reports of accusations of sexual misconduct and abuse. A system that is not capable of distinguishing the context of usage is not going to provide reliable quantification of incivility in the news.

Similarly, the ambiguous error triggers are words with more than one meaning, which could be offensive but can be used in civil discussion as well. For these, Perspective has to disambiguate based on the context. All of the templates we used were neutral, so for these, Perspective makes an error. For example, the name 'Dick' is an error trigger. The word indeed has an offensive meaning, but in the news shows we analyze, all mentions are in the sense of a person's name.

A couple of error-triggers are informal, clashing more in register than conveying incivility.

8 Sub-Error Triggers

Sub-error triggers of incivility are words that are not offensive and for which Perspective returns a toxicity score below 0.5 for each of the five templates when the word is plugged in the template. The error triggers we discussed above lead to actual errors by Perspective for its intended use. The sub-error triggers are in the acceptable level of noise for the original purpose of Perspective but may introduce noise when the goal is to quantify the typical overall civility of a source.

To establish a reference point for the expected average template civility of non-offensive words, we sample 742 words that appeared in at least 10 speaker turns (i.e., were fairly common) and appeared in speaker turns in the news shows that received low toxicity predictions from Perspective, below 0.15. These were part of the list we classify as offensive or not. No word in this reference sample was labeled as offensive.

The average template score for the reference sample is 0.09, with a standard deviation of 0.01. We define sub-error triggers to be those whose average template toxicity score is two standard deviations higher than the average in the reference list. There are 325 non-offensive words that meet the criteria for sub-error triggers. They are shown in Table 5 The list is long, which is disconcerting because it is likely that sentences containing multiple of these words will end up triggering errors.

A sizeable percentage of sub-error triggers are related to identities of people—gender, age, religion, country of origin. Oddly, a number of child-describing words appear in the list (baby, child, kid, toddler). There are also several personal names in the list, which indicates spurious correlations learned in Perspective; names by themselves can not be uncivil.

Second person pronouns (you, your) and third-person feminine pronouns (her, she) are sub-error triggers. The second-person pronouns are likely spuriously associated with toxicity due to over-representation in direct toxic comments directed to other participants in the conversation. Similarly, the association of female pronouns with toxicity is likely due to the fact that a large fraction of the indirect toxic comments online are targeted to women.

Regardless of the reasons why these words were associated with incivility by Perspective, the vast majority of them are typical words mentioned in the news and political discussions, explaining to an extent why Perspective is not able to provide a

The other thing that's important, both sides seem to be mad about this. On the conservative side, you have conservative voices who are saying, this deal is **pathetic**, it's an insult to the president. On the Democratic side and on the liberal side, I have activists that are texting me saying, Nancy Pelosi said she wasn't going to give $1 for this wall, and now she is.
I do feel safe at school. And I know that sounds kind of **ridiculous**, since tomorrow makes a year since there was a shooting at my school. But I do feel safe at school. And I feel safe sending my children to school. I know that there are recommendations that have been made to arm teachers, and I think that is the **stupidest** thing that I have ever heard in my life. Me having a gun in my classroom wouldn't have helped me that day.
VIRGINIA ROBERTS, Victim: It ended with sexual abuse and intercourse, and then a pat on the back, you have done a really good job, like, you know, thank you very much, and here's $200. You know, before you know it, I'm being lent out to politicians and to academics and to people that – royalty and people that you just – you would never think, like, how did you get into that position of power in the first place, if you're this **disgusting**, evil, decrepit person on the inside?
In a major incident like this, at first there's, you know, there's sort of a stunned numb thing that happens. And then you kind of go into this honeymoon phase. There's just a high level of gratefulness for all of the help that's coming. And then you get to the phase that we're kind of beginning to dip in now, which is life **sucks** right now, and I don't know how long it's going to **suck**.
It's **insane**. We are throwing away tremendous amounts of food every day. And there are people next door a block away that aren't getting enough food.

Table 6: Segments from PBS that contain offensive words (marked in boldface).

Category	FOX	MSNBC	PBS	Total
Error	197 [.44]	55 [.12]	196 [.44]	448
Sub-error	1537 [.39]	723 [.18]	1708 [.43]	3968
Offensive	277 [.52]	101 [.19]	156 [.29]	534

Table 7: Number [and fraction] of segments containing at least one (sub-)error trigger or offensive word.

meaningful characterization of civility in the news.

Table 7 shows the distribution of error triggers, sub-error triggers, and offensive words in the three programs. Most of the segments containing error and sub-error triggers are in FOX and PBS; this could explain the observation that incivility is much higher in FOX compared to the other two programs when analyzed with Perspective than compared to that from human judgments. This also explains why PBS, judged significantly more civil than MSNBC by people, appears to be somewhat less civil. Not only is Perspective not able to detect some of the incivility in sarcasm present in MSNBC, but also PBS segments include substantially more (sub-)error triggers than MSNBC.

More segments from PBS contain uncivil words, compared to MSNBC. Table 6 shows some representative PBS segments with offensive words. They are often reported incivility, or occur in segments where a guest in the program uses such language. Most of the segments are not overall uncivil.

9 Conclusion

The work we presented was motivated by the desire to apply off-the-shelf methods for toxicity prediction to analyse civility in American news. These methods were developed to detect rude, disrespectful, or unreasonable comment that is likely to make you leave the discussion in an online forum. To validate the use of Perspective to quantify incivility in the news, we create a new corpus of perceived incivility in the news. On this corpus, we compare human ratings and Perspective predictions. We find that Perspective is not appropriate for such an application, providing misleading conclusions for sources that are mostly civil but for which people perceive a significant overall difference, for example, because one uses sarcasm to express incivility. Perspective is able to detect less subtle differences in levels of incivility, but in a large-scale analysis that relies on Perspective exclusively, it will be impossible to know which differences would reflect human perception and which would not.

We find that Perspective's inability to differentiate levels of incivility is partly due to the spurious correlations it has formed between certain non-offensive words and incivility. Many of these words are identity-related. Our work will facilitate future research efforts on debiasing of automated predictions. These methods start off with a list of words that the system has to unlearn as associated with a given outcome. In prior work, the lists of words to debias came from informal experimentation with predictions from Perspective. Our work provides a mechanism to create a data-driven list that requires some but little human intervention. It can discover broader classes of bias than people performing ad-hoc experiments can come up with.

A considerable portion of content marked as uncivil by people is not detected as unusual by Perspective. Sarcasm and high-brow register in the delivery of the uncivil language are at play here and will require the development of new systems.

Computational social scientists are well-advised to not use Perspective for studies of incivility in political discourse because it has clear deficiencies for such application.

Acknowledgments

We thank the reviewers for their constructively critical comments. This work was partially funded by the National Institute for Civil Discourse.

References

Pew Research Center. 2019. Public highly critical of state of political discourse in the u.s. Technical report.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018a. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*, pages 67–73.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018b. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. *arXiv preprint arXiv:1802.00393*.

Jeremy A Frimer and Linda J Skitka. 2018. The montagu principle: Incivility decreases politicians' public approval, even with their political base. *Journal of personality and social psychology*, 115(5):845.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Yiqing Hua, Cristian Danescu-Niculescu-Mizil, Dario Taraborelli, Nithum Thain, Jeffery Sorensen, and Lucas Dixon. 2018. WikiConv: A corpus of the complete conversational history of a large online collaborative community. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2818–2823, Brussels, Belgium. Association for Computational Linguistics.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. 2020. Social biases in NLP models as barriers for persons with disabilities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5501, Online. Association for Computational Linguistics.

Kokil Jaidka, Alvin Zhou, and Yphtach Lelkes. 2019. Brevity is the soul of twitter: The constraint affordance and political discussion. *Journal of Communication*, 69(4):345–372.

Kimberly Meltzer. 2015. Journalistic concern about uncivil political talk in digital news media: Responsibility, credibility, and academic influence. *The International Journal of Press/Politics*, 20(1):85–107.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229.

Diana C Mutz and Byron Reeves. 2005. The new videomalaise: Effects of televised incivility on political trust. *American Political Science Review*, pages 1–15.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745, Hong Kong, China. Association for Computational Linguistics.

Ian Rowe. 2015. Civility 2.0: A comparative analysis of incivility in online political discussion. *Information, communication & society*, 18(2):121–138.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Yannis Theocharis, Pablo Barberá, Zoltán Fazekas, and Sebastian Adrian Popa. 2020. The dynamics of political incivility on twitter. *Sage Open*, 10(2):2158244020919447.

Eric M Uslaner. 2000. Is the senate more civil than the house. *Esteemed colleagues: Civility and deliberation in the US Senate*, pages 32–55.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Mengzhou Xia, Anjalie Field, and Yulia Tsvetkov. 2020. Demoting racial bias in hate speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 7–14, Online. Association for Computational Linguistics.

Marc Ziegele, Mathias Weber, Oliver Quiring, and Timo Breiner. 2018. The dynamics of online news discussions: Effects of news articles and reader comments on users' involvement, willingness to participate, and the civility of their contributions. *Information, Communication & Society*, 21(10):1419–1435.

On the Computational Modelling of Michif Verbal Morphology

Fineen Davis, Eddie A. Santos
National Research Council Canada
`{firstname.lastname}`
`@nrc-cnrc.gc.ca`

Heather Souter
Prairies to Woodlands Indigenous
Language Revitalization Circle
`p2wilrc@gmail.com`

Abstract

This paper presents a finite-state computational model of the verbal morphology of Michif. Michif, the official language of the Métis peoples, is a uniquely mixed language with Algonquian and French origins. It is spoken across the Métis homelands in what is now called Canada and the United States, but it is highly endangered with less than 100 speakers. The verbal morphology is remarkably complex, as the already polysynthetic Algonquian patterns are combined with French elements and unique morpho-phonological interactions. The model presented in this paper, LI VERB KAA-OOSHITAHK DI MICHIF handles this complexity by using a series of composed finite-state transducers to model the concatenative morphology and phonological rule alternations that are unique to Michif. Such a rule-based approach is necessary as there is insufficient language data for an approach that uses machine learning. A language model such as LI VERB KAA-OOSHITAHK DI MICHIF furthers the goals of Indigenous computational linguistics in Canada while also supporting the creation of tools for documentation, education, and revitalization that are desired by the Métis community.

1 Introduction

In recent years there has been an increase in computational linguistic analysis of Indigenous languages spoken in what is now called Canada, and in particular Algonquian languages such as East Cree (Arppe et al., 2017a), Plains Cree (Harrigan et al., 2017), and Odawa (Bowers et al., 2017). This paper adds Michif—a mixed language of Cree and French origin—to the list with a description of LI VERB KAA-OOSHITAHK DI MICHIF. LI VERB KAA-OOSHITAHK DI MICHIF, which translates as "The Michif verb maker", is a computational model of the verbal morphology of Michif, implemented in the XFST framework of Beesley and Karttunen (2003).

2 Motivation

During the 19th century, marriage between French fur traders and Cree and Anishinaabe women in the Métis homeland was common, and their descendants became known as the Métis peoples (Bakker, 1997; Rosen and Souter, 2009). Michif (ISO 639-3: crg), an Algonquian language, emerged as a mixed language which combined elements of French with the Indigenous Algonquian languages Cree and Saulteaux—a distinct dialect of Ojibwa (Bakker, 1997; Rosen and Souter, 2009). There are many varieties of the Michif language, however LI VERB KAA-OOSHITAHK DI MICHIF is based on a variety spoken mainly in Manitoba, Southern Saskatchewan, North Dakota, and Montana.

Lack of linguistic documentation and analysis restricts the ability to create formalized teaching tools and technologies and hinders the efforts of language learners and activists in Indigenous communities. The number of speakers has decreased significantly. It is difficult to estimate true numbers of speakers of the "intertwined" language (Bakker, 1997), but Michif language activists estimate approximately 50-100 speakers with only a handful presently robust enough to be involved in revitalization work (Souter, 2020).

LI VERB KAA-OOSHITAHK DI MICHIF aims to create a complete model of the verbal morphology of Michif based on the current language data. The output of such a model has many applications, including a smartphone application for conjugating verbs in Michif.[1] LI VERB KAA-OOSHITAHK DI MICHIF represents a collabora-

---

[1] As of this writing, one such app is in development that incorporates LI VERB KAA-OOSHITAHK DI MICHIF.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2631–2636
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

tive effort between linguists, computer scientists, and the Michif speech community.

The source code for Li Verb kaa-Ooshitahk di Michif is not publicly available to maintain Métis sovereignty of the language data as requested by the community.

3 Michif linguistic background

The unique mixed origins of Michif are reflected in all linguistic domains. The phonological inventory of Michif is sourced from both French and Cree (with the "Saulteaux" or "Chippewa" dialect of Objibwe being a more minor source language). The pronominals are largely French based and verbal constructions are primarily Algonquian-derived (Sammons, 2019; Prichard and Shwayder, 2014; Rosen, 2007). However, while the morphemes and their concatenations are similar to Algonquian languages such as Plains Cree, the morpho-phonological interactions differ significantly from its sources (Sammons, 2019).

Michif is similar to other Algonquian languages in the degree of polysynthetic complexity, yet there are still distinguishing features that make it unique. Verbs can be highly productive by concatenating morphemes containing many categories of syntactic information; Figure 1 provides an illustration of the basic verbal template. The glosses below show an example of a simple verb form and one of a more complex verb form.

(1) *ayaaw*
ayaa-w
IND.PRS.have.VAI-3SG

'He/she has.'

(2) *ee-ka-kishkeetamiiyit*
ee-ka-kishkeet-amii-yi-t
CONJ-FUT-know.VTI-3>3-OBV1-OBV2

'As/that he/she (OBV) knows it.'

Michif verbs can be broken down into four inflectional classes by transitivity and animacy. Transitive verbs combine with the animacy of the object to create their inflectional classes (Transitive Animate (VTA) & Transitive Inanimate (VTI)), while intransitive verbs combine with the animacy of the subject (Animate Intransitive (VAI) & Inanimate Intransitive (VII)) (Wolvengrey, 2011; Sammons, 2019). Person marking, obviation, and direction affixes vary according to the inflectional class. There are also two minor classes, Animate Intransitive Transitive (VAIt) (Sammons, 2019) and Animate Intransitive Transitive animate/inanimate (VAIta/i) (Antonov, 2019).

All inflectional classes can be further broken down into the independent and conjunct orders. The independent order marks person and number with long distance agreement between prefixes and suffixes, and is used to express the indicative mode. The conjunct order expresses the indicative and subjunctive modes, and is typically marked with the '*ee-*' prefix, meaning 'while/as'.

Preverbs provide inflectional and lexical information before the stem of a verb, but occur *after* the person marking prefix in the independent order, or *after* the conjunct marker in the conjunct order. Grammatical preverbs cover tense or relativization, while lexical preverbs are a closed class that add lexical meaning (Sammons, 2019; Rhodes, 2009). Preverbs are optional, and a verb can be modified by multiple preverbs, as seen in the below example. However, while recursion of preverbs is theoretically possible in Michif, such forms are not used by speakers in practice, so this recursion is not modelled by Li Verb kaa-Ooshitahk di Michif.

(3) *ni-ka-nohtee-maachi-atoshkaanaan*
ni-ka-nohtee-maachi-atoshk-aanaan
1.IND-FUT.DEF-want-begin-work.VAI-1PL.EXCL

'We (EXCL) will want to begin to work'

VTA verbs in Michif are the only class which has direction, where actions are either direct or inverse depending on a hierarchy of actors as in other Algonquian languages such as Plains Cree (Harrigan et al., 2017). In the direct VTA forms, the 'ni-/ki-/∅-' prefix refers to the subject, while the person marking suffix refers to the object. In the inverse VTA forms, the 'ni-/ki-/∅-' prefix refers to the object, while the person marking suffix refers to the subject. This can be observed in the difference between the suffixes in the glossed examples below.

(4) *ki-miyeumin*
ki-miyeum-**in**
2.IND-PRS.like.VTA-**DIR**.2SG>1SG

'You (SG) like me'

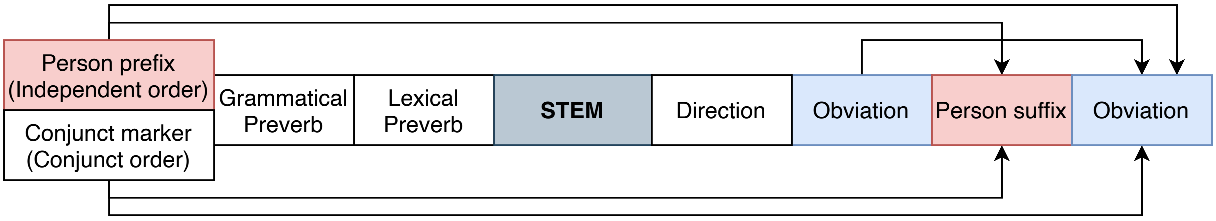


Figure 1: Verbal template of Michif

(5) *ki-miyeumitin*
ki-miyeum-**itin**
1.IND-PRS.like.VTA-**INV**.1SG>2SG

'I like you (SG)'

Obviation occurs across all inflectional classes in Michif and is triggered by obviative nouns (marked or unmarked) (Sammons, 2019). It only occurs in the 3rd person, and number distinction is not specified, although the verb takes the 3SG suffix[2]. VAI verbs mark obviation in up to two places, relying on long distance dependencies. The '*yi*' morpheme occurs immediately following the stem. The independent order will have an extra '*a*' morpheme following the person marking affix, while the conjunct order will not. This pattern is visible in the examples in the glosses below.

(6) *soñ* *namii*
soñ namii
3SG.MASC.POSS friend
kii-itohteeyiwa
kii-itohtee-**yi**-w-**a**
PST-go.VAI.IND-**OBV1**-3-**OBV2**

'His/her friend (OBV) went'

(7) *soñ* *namii*
soñ namii
3SG.MASC.POSS friend
kii-itohteeyit
kii-itohtee-**yi**-**t**
PST-go.VAI.CONJ-**OBV1**-3

'As his/her friend (OBV) went'

4 Finite-state computational modelling

Li Verb kaa-Ooshitahk di Michif is a morphological model—a series of composed finite-state transducers (FSTs) called a lexical transducer (Beesley and Karttunen, 2003). There are two primary components of a lexical transducer: the Lexicon and the RewriteRules. The Lexicon uses a set of labelled sub-Lexicons which declare the rules for morphological concatenation (Hulden, 2009). This is accomplished by using regular expressions and flag diacritics. Then, the RewriteRules further constrain the output of the FSTs by using regular expressions to apply phonological restrictions. Foma (Hulden, 2009), a finite-state compiler, takes the Lexicon and the RewriteRules to create a composed finite-state transducer—in this case, Li Verb kaa-Ooshitahk di Michif. Using FSTs to model low-resource languages such as Michif is advantageous, as rule-based definitions of verbal paradigms do not rely on access to large, morphologically-tagged corpora which do not exist for Michif.

4.1 Morphological modelling in the Lexicon

The Lexicon allows for the linear concatenation of morphemes in the form of a continuation grammar. Each sub-Lexicon adds two components: morphological tags to the "upper" side, which provide syntactic information, and the surface form morphemes themselves on the "lower" side. Figure 2 illustrates the main sub-Lexicons of the Li Verb kaa-Ooshitahk di Michif.

Long distance dependencies are particularly challenging to model with continuation grammars, as they require knowledge of previous states. *Flag diacritics* enable these long distance dependency checks between states so that phenomena such as obviation can be modelled (Hulden, 2009; Beesley and Karttunen, 2003). Li Verb kaa-Ooshitahk di Michif employs three types of flags: `P`: **P**ositive set; `R`: **R**equire feature/value; and `D`: **D**isallow feature/value. Each feature can be thought of as a set; `P` flags add a value to this

---

[2] This paper only focuses on obviation in VAI verbs. See Sammons (2019) for a description of obviation in all verb classes.

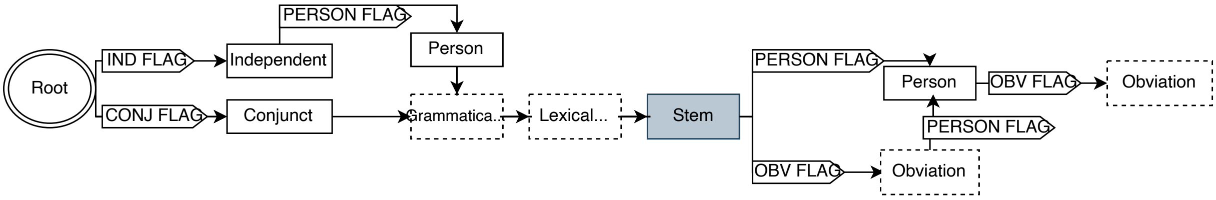


Figure 2: Schematic representation of paths through the LEXICON

feature set, while R flags require a specific value or the entire feature to be set.

As obviation in Michif only occurs with the 3rd person, flag diacritics have to be used to avoid over-application of the obviative concatenation rules. A 'person' feature is used, which distinguishes between 1st and 2nd person forms. When this feature is disallowed, such as with a *@D.person@* flag, only 3rd person forms can pass through. Any forms set to 1st or 2nd person will pass through the null path and continue to the `VAIPersonSuffix` sub-LEXICON, while the 3rd person forms will receive obviative marking before continuing to the `VAIPersonSuffix` sub-LEXICON.

```
LEXICON VAIObviativeMarking
 0                      VAIPersonSuffix;
 @D.person@@P.obv.yi@   VAIYi;

LEXICON VAIYi
 0:yi VAIPersonSuffix;
```

LI VERB KAA-OOSHITAHK DI MICHIF treats obviative as a feature set by a flag. The *@P.obv.yi@* sets the obviative feature, for any forms containing the '*yi*' morpheme. In the example below, the first sub-LEXICON ensures that only obviative forms receive the `[OBV]` tag. The *@D.obv@* flag accepts only those forms which have not previously been set to positive for the obviative feature. The *@R.obv@* flag requires obviative forms, outputting the tag on the "upper" side.

```
LEXICON VAIObviative1
 @D.obv@                EndVerb;
 @R.obv@[OBV]:@R.obv@   VAIObviative2;

LEXICON VAIObviative2
 @R.order.indep@:@R.order.indep@a EndVerb;
 @D.order.indep@a                 EndVerb;
```

The obviative forms then pass to the second sub-LEXICON, where they are sorted by order using flags. The *@R.order.indep@* requires forms belonging to the independent order, which adds the '*-a*' morpheme. All other forms, i.e. those belonging to the conjunct order, pass through the path with the *@D.order.indep@* flag (disallow any independent forms). Reaching *EndVerb* indicates that the transduction is complete. Table 1 shows examples of lexical outputs before the application of the phonological rules.

The modelling of obviation in Michif takes the opposite approach of the Plains Cree FST in Harrigan et al. (2017). In LI VERB KAA-OOSHITAHK DI MICHIF, the direction and person morpheme are treated as a single unit. Each obviative morpheme is then concatenated individually, rather than as part of the person and direction affixes. For example, LI VERB KAA-OOSHITAHK DI MICHIF would generate '*<eeyi<eew<a>*' instead of '*<eeyieewa>*'. As the intended use of this model is in contexts where all verb forms will only be generated once instead of with each query, the speed of generation is not required to be optimized, and more leniency with regards to stylistic choice is possible.

4.2 Phonological restrictions with REWRITERULES

LI VERB KAA-OOSHITAHK DI MICHIF only accounts for morpho-phonological rules which interact with the verbal morphology of Michif. When morphemes are concatenated, phonemes that would otherwise not co-occur become adjacent, creating the need for phonological rules which handle these issues. The REWRITERULES finite-state machine treats each phonological rule as an individual FST which is then composed to form the final FST (Hulden, 2009; Beesley and Karttunen, 2003).

The *'ni-/ki-'* prefixes host a multitude of morpho-phonological interactions unique to Michif.[3] There is significant vowel reduction as the prefixes are unstressed, which results in the eventual deletion of entire morphemes. Accounting for these alternations requires flag diacritics, such as with obviation, but additionally requires REWRITERULES.

---

[3] See Bakker (1991) for a full account of morpho-phonological rules in Michif.

<achimostaaweew[VTA][IND][FUT][3][INV][3PL-OBJ][OBV]	<kii-achimostaaw<ikwak>
<ayamihaaw[VAI][IND][PRS][1SG]	<ni-ayamihaa<n>

Table 1: Sample output of Li Verb kaa-Ooshitahk di Michif Lexicon

Lexicon Output	t-Insertion	ni-Deletion
ni-ayamihaan	ni-t-ayamihaan	d-ayamihaan
ni-ka-maachi-kipahaaw		ga-maachi-kipahaaw

Table 2: Sample derivation of the t-Insertion and ni-Deletion RewriteRules

1. t-Insertion
   [..] -> t || [n i | k i] "-" _ Vowel ;
2. ni-Deletion
   n i "-" t (->) d "-" || _ Vowel ;
   n i "-" k (->) g "-" || _ Vowel ;

Following the *'ni-/ki-'* morphemes, [t] is epenthesized when followed by a vowel (Rosen, 2007). Voiceless consonants are voiced when preceded by the *'ni-'* prefix and followed by a vowel. The *'ni-'* prefix is then deleted following the voicing (Bakker, 1991). The RewriteRules combine these two processes into one regular expression rule to avoid over-application of both the voicing and deletion. Table 2 illustrates the application of these rules.

5 Discussion

The latest version of Li Verb kaa-Ooshitahk di Michif includes 105 verb stems, which generate a total of 155,621 verb forms. Compared to the Plains Cree FST, which has around 13,000 verb stems and effectively infinite verbs forms due to recursion, Li Verb kaa-Ooshitahk di Michif is relatively small. However, it is important to recognize that the difference in size is a direct consequence of the linguistic situation of Michif. There is no Gold Standard morphologically-tagged corpus, such as that which exists for Plains Cree (2017b), against which to compare output forms. As a result, the generated forms are hand-verified by speakers (Souter, 2020), so having such a large model is less feasible.

The primary challenge that was faced when creating Li Verb kaa-Ooshitahk di Michif is the lack of consistent language data. The published paradigms conflict in their analysis of Michif verbal morphology, particularly with regards to phonological elements such as vowel length in person marking suffixes (Sammons, 2019; Bakker, 1997; Rosen, 2007). Initially, much of the FST was based on data scraped together from glosses discussing other phenomena, resulting in paradigm gaps. In the absence of accurate language data, the role of speakers becomes vital to ensuring the model generates accurate outputs, beginning with having complete morphological paradigms.

The imperative, subjunctive, and reflexive modes are not currently part of the model. While paradigms for these forms now exist at least partially, in order to implement them the structure of the current FST would have to be built again from the ground up. At the time of creating the current model these paradigms were not available, and so they were not included in the original architectural design and layout of the FST in Foma.

6 Conclusion

Despite the development of Li Verb kaa-Ooshitahk di Michif, much more work is needed to create a complete account of the verbal morphology of Michif. Computational language modelling is an important foundational step towards creating language learning resources and making the creation of those resources more accessible to Indigenous language communities.

References

Anton Antonov. 2019. Loan verb integration in Michif. *Journal of Language Contact*, pages 27–52.

Antti Arppe, Marie-Odile Junker, and Delasie Torkornoo. 2017a. Converting a comprehensive lexical database into a computational model: The case of East Cree verb inflection. In *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 52–56, Honolulu. Association for Computational Linguistics.

Antti Arppe, Katherine Schmirler, Atticus G. Harrigan, and Arok Wolvengrey. 2017b. A morphosyntactically tagged corpus for Plains Cree. In *Papers of the Forty-Ninth Algonquian Conference*, pages 1–16.

Peter Bakker. 1991. The Ojibwa element in Mitchif. *Algonquian Papers-Archive*, 22.

Peter Bakker. 1997. *A language of our own: The genesis of Michif, the mixed Cree-French language of the Canadian Métis*, volume 10. Oxford University Press.

Kenneth R Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications.

Dustin Bowers, Antti Arppe, Jordan Lachler, Sjur Moshagen, and Trond Trosterud. 2017. A morphological parser for Odawa. In *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 1–9, Honolulu. Association for Computational Linguistics.

Atticus G. Harrigan, Katherine Schmirler, Antti Arppe, Lene Antonsen, Trond Trosterud, and Arok Wolvengrey. 2017. Learning from the computational modelling of Plains Cree verbs. *Morphology*, 27(4):565–598.

Mans Hulden. 2009. Foma: a finite-state compiler and library. In *Proceedings of the Demonstrations Session at EACL 2009*, pages 29–32, Athens, Greece. Association for Computational Linguistics.

Hilary Prichard and Kobey Shwayder. 2014. Against a split phonology of Michif. *University of Pennsylvania Working Papers in Linguistics*, 20(1):29.

Richard Rhodes. 2009. The phonological history of Métchif. *Français dun Continent à lautre: Mélanges Offerts à Yves Charles Morin*, 1:423–442.

Nicole Rosen. 2007. *Domains in Michif phonology*. University of Toronto.

Nicole Rosen and Heather Souter. 2009. Language revitalization in a multilingual community: The case of Michif. In *1st International Conference on Language Documentation and Conservation (ICLDC)*, Honolulu.

Olivia N. Sammons. 2019. *Nominal classification in Michif*. Ph.D. thesis, University of Alberta.

Heather Souter. 2020. Personal Communication.

Arok Elessar Wolvengrey. 2011. *Semantic and pragmatic functions in Plains Cree syntax*. Netherlands Graduate School of Linguistics.

A Few Topical Tweets are Enough for Effective User Stance Detection

Younes Samih
Dept. of Computational Linguistics,
University of Düsseldorf
Düsseldorf, Germany
`Samih@phil.hhu.de`

Kareem Darwish
Qatar Computing Research Institute, HBKU
Doha, Qatar
`kdarwish@hbku.edu.qa`

Abstract

User stance detection entails ascertaining the position of a user towards a target, such as an entity, topic, or claim. Recent work that employs unsupervised classification has shown that performing stance detection on vocal Twitter users, who have many tweets on a target, can be highly accurate (+98%). However, such methods perform poorly or fail completely for less vocal users, who may have authored only a few tweets about a target. In this paper, we tackle stance detection for such users using two approaches. In the first approach, we improve user-level stance detection by representing tweets using contextualized embeddings, which capture latent meanings of words in context. We show that this approach outperforms two strong baselines and achieves 89.6% accuracy and 91.3% macro F-measure on eight controversial topics. In the second approach, we expand the tweets of a given user using their Twitter timeline tweets, which may not be topically relevant, and then we perform unsupervised classification of the user, which entails clustering a user with other users in the training set. This approach achieves 95.6% accuracy and 93.1% macro F-measure.

1 Introduction

Stance detection entails identifying the position of a user towards a topic, an entity, or a claim (Mohammad et al., 2016b). Effective stance detection, particularly in the realm of social media, can be instrumental in gauging public opinion, identifying intersecting and diverging groups, and understanding issues of interest to different user communities (Magdy et al., 2016a). Much recent works have explored varying stance detection methods including supervised, semi-supervised, and unsupervised user classification (Darwish et al., 2020; Magdy et al., 2016a; Pennacchiotti and Popescu, 2011; Wong et al., 2013), and much of the work has focused on stance detection for Twitter users. The different approaches have advantages and disadvantages. For example, supervised methods are simple to implement, but they require manually annotated training data and their accuracy varies widely based on classification features, classification techniques, and the number of training and test examples (Magdy et al., 2016a). Though semi-supervised and unsupervised methods typically use user interactions and often may yield perfect classification, they are effective in classifying highly vocal users with many topical tweets (Darwish et al., 2020). Most of these methods produce sub-optimal results for users who rarely express their opinion, and for whom we may only have one or two topically related tweets. Though a single tweet might be explicitly clear, often it may lack sufficient context to determine the stance of the user. Figure 1 show two tweets that pertain to the 2018 US midterm elections, where the first expresses a lucid pro-Republican stance and the second could have been authored by a supporter of either the Republican or Democratic party. In this paper, we aim to effectively identify the stance of Twitter users towards specific targets (entities or topics), where the users have mentioned the targets in only a few tweets (less than two tweets on average).

Tom Fitton @TomFitton

TRUTH: About the caravan, about Left's desire to steal elections, about the Left's efforts to use Sayoc to suppress conservative speech. @JudicialWatch Update. youtu.be/wQvQ72uJa6g

2:07 AM · Oct 27, 2018 · Twitter Web Client

(a)

I am Shane. @_ShaneEdmundson

This election is going to be painfully close. Again. Don't wake up on Nov 7th and regret you didn't do more. Please, VOTE ON NOV 6th!

11:56 PM · Oct 25, 2018 · Twitter SMS

(b)

Figure 1: Sample 2018 US midterm election related tweets that either express a very clear stance (a) or not (b)

To do so, we employ two approaches. In the first

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2637–2646
April 19 - 23, 2021.

approach, we classify users based on their tweets that are represented using contextualized embeddings, which capture latent meanings of words in context. Specifically, we use BERT embeddings to represent tweets, and we fine tune the embeddings for every topic. We compare this approach to two strong baselines, namely using Support Vector Machine (SVM) classification and fastText, which is a deep learning based classifier. In the second approach, we expand the tweets of a given user using their Twitter timeline tweets, and then we use the additional tweets, which would typically be not relevant to the topic at hand, to perform unsupervised classification of the user by clustering him/her with the users in the training set. Using such expansion allows us to make use of user homophily, which is manifested in the echo chambers that form on Twitter, where users with similar views tend to retweet similar accounts beyond the topic at hand. To test our approaches, we use a dataset containing tweets on 8 polarized US-centric topics. We also examine the effect of expansion when using SVM, fastText, and contextualized embeddings. For testing, we randomly selected 100 users for each topic that have less than 5 topical tweets, and we manually labeled them for stance. To construct the training set, we used unsupervised stance detection to automatically label the 5,000 most active users per topic, and for every topic we used a balanced set of 500 users per stance as our training set (Darwish et al., 2020). Since the approaches rely on different features and utilize different classification techniques, we indicate which approach works best under different conditions.

The contributions of this paper are as follows:

- We fine-tune contextualized embeddings to generate latent representations of tweets to effectively classify the stance of users based on only one or two tweets. We achieve an accuracy of 89.6% and macro F-measure of 91.3%, which are significantly higher than the scores achieved using two strong baselines.
- We show that using additional timeline tweets for the users that we wish to classify, and then using unsupervised classification, where we cluster the test user with users in our training, leads to an accuracy of 95.6% and macro F1-measure of 92.0%. In doing so, we extend prior work on unsupervised stance detection to effectively classify both users who are vocal on a topic as well as those with perhaps one or two topical tweets.
- We show that expanding user tweets using their timeline tweets can significantly improve some supervised classification setups.
- We conduct error analysis on our best setups to determine the sources of the errors and to guide the choice of classification methods.
- We plan to release the tweet IDs of the test set along with the associate gold labels. Further, we plan to release the code that performs classification based on contextualized embeddings.

2 Related Work

Over the last few years, much research has focused on user stance detection. The goal of stance detection is to ascertain the positions of users towards some target such as a topic, person, or claim (Thomas et al., 2006; Mohammad et al., 2016a; Barberá, 2015; Barberá and Rivero, 2014; Borge-Holthoefer et al., 2015; Cohen and Ruths, 2013; Colleoni et al., 2014; Conover et al., 2011; Fowler et al., 2011; Himelboim et al., 2013; Magdy et al., 2016a,b; Makazhanov et al., 2014; Weber et al., 2013). While stance may easily be detected by humans, machine learning models often fall short, particularly for users who talk about a target sparingly. Several studies have focused on modeling stance by introducing different features ranging from linguistic and structural features (Mohammad et al., 2016a) to network interactions and profile information (Borge-Holthoefer et al., 2015; Magdy et al., 2016a,b; Weber et al., 2013). Much work on stance detection involved using supervised and semi-supervised classification methods. One of the major downsides of both classification methods is the need for a seed list of manually labeled users, which is time consuming and requires topic expertise. Supervised learning is sensitive to the classification features, the size of the training sets, the number of available tweets for users in the test set, and the classification algorithm (Borge-Holthoefer et al., 2015). Some common classification features include: lexical, syntactic, and semantics feature; network features such as retweeted accounts and user mentions; content features such as words and hashtags; and user profile information such as name and location (Aldayel and Magdy, 2019; Magdy et al., 2016a,b; Pennacchiotti and Popescu, 2011). Some commonly used classification algorithms include SVMs and deep learning classification (Zarrella and Marsh, 2016). Popat et al.

(2019) presented a neural network model for stance classification by augmenting BERT representations with a novel consistency constraint to determine stance with respect to both a claim and perspective. We extend their work in two ways, namely: we drop the need to have a claim and perspective, and we couple BERT supervised classification with unsupervised classification to effectively tag vocal and non-vocal users. Semi-supervised methods such as *label propagation* (Barberá, 2015; Borge-Holthoefer et al., 2015; Weber et al., 2013) often rely on two users retweeting identical accounts or tweets to propagate a label of one user to another. Though such typically achieves high precision (often above 95%) (Darwish et al., 2018), it is generally successful in tagging vocal users with strong opinions. Recently, Darwish et al. (2020) have introduced a highly effective unsupervised method for predicting the stance of prolific Twitter users towards controversial topics. By projecting users onto a low-dimensional space and then clustering them allows for clear separation between vocal users with respect to their stance (Darwish et al., 2020). This method confers two main advantages over previous methods, namely: it does not require any initial manual labeling, and classification accuracy is nearly perfect. However, it is successful in labeling vocal users only and fails on users with very few topical tweets. We extend prior work on unsupervised stance detection to effectively classify both prolific and non-prolific users in a holistic way by aggregating both supervised and unsupervised methods. Further, we extend prior deep-learning based supervised classification to use contextual embeddings that capture syntactic and semantic features of words in context.

We are framing the problem as user-based classification, which is common in the computational social science community, as opposed to tweet-based classification, which is common in the NLP community (Mohammad et al., 2016b). This is motivated by two aspects, namely: 1) tweets often don't provide sufficient context for proper annotation; and 2) users have durable stances over time. For example, if someone says "Most important election in history! Vote!", it is nearly impossible to know if the author's position without context.

3 Data Sets

Topics Our dataset includes tweets on eight polarizing topics that are US-centric, which were graciously provided to us by Stefanov et al. (2020). Table 1 lists all the topics including when the tweets were collected and the number of tweets per topic. The topics include both long-standing issues such as gun control and transient issues such as the nomination of Judge Kavanaugh to the US Supreme Court. There is also a non-political issue, namely vaccination. The tweets were also filtered based on user-stated locations to limit the data to US users. The filtering was done using a gazetteer that includes either US (or its variants) and state names (and their abbreviations).

Topic	Date Range	Tweets
Climate change	Feb 25–Mar 4, 2019	1,284,902
Gun control	Feb 25–Mar 3, 2019	1,782,384
Ilhan Omar (remarks on Israeli lobby)	Mar 1–9, 2019	2,556,871
Immigration	Feb 25–Mar 4, 2019	2,341,316
Midterm (elections 2018)	Feb 25–Mar 3, 2019	2,564,784
Kavanaugh (nomination to Supreme Court)	Sept. 28-30 & Oct. 6-9, 2018	2,322,141
Vaccination	Mar 1–9, 2019	301,209

Table 1: Controversial topics used in study.

Training Set Given the tweets for every topic, we performed per topic unsupervised stance detection (Darwish et al., 2020). This approach identifies the most active n users per topic and computes similarity between them based on a common feature, such as which hashtags they use or which accounts that they retweet. Next, the users are projected onto a lower dimensional space in a manner where similar users are brought closer together and dissimilar users are pushed further apart. Then the projected users are clustered. Using the best reported setup of Darwish et al. (2020), we used the 5,000 most active users with at least 10 tweets, computed similarity between them based on which accounts they retweeted, projected users using UMAP (McInnes and Healy, 2018), and clustered them using the mean shift clustering algorithm (Fukunaga and Hostetler, 1975). Stefanov et al. (2019) estimated the accuracy of the unsupervised approach on the 8 topics to be 98%. Next, we took 500 random users from the two largest clusters to construct a balanced training set, and we manually inspected a few users from each cluster to give an overall label to each cluster (ex. pro- or anti- gun control). Further, we crawled the timeline tweets of the users in our training set.

Topic	Skip	Agree	Pro	Anti
Climate	10.6	100	81	18
Gun control	71.0	85	62	37
Ilhan	1.0	95	52	47
Immigration	10.7	90	57	42
Midterm	0.0	90	71	28
Police/Racism	0.0	100	80	20
Kavanaugh	2.0	100	39	60
Vaccine	1.0	100	90	9
Avg.	12.0	95	66.5	32.6

Table 2: Test set information: % skipped tweets, inter-annotator agreement, % pro, and % anti.

Topic	Labeled Users	w/Timelines
Climate change	100	54
Gun control	58	26
Ilhan Omar	100	39
Immigration	100	43
Midterm	100	45
Police/Racism	100	54
Kavanaugh	100	55
Vaccines	100	57

Table 3: Per topic labeled users in test set along with the number of users for which were able to scrape their timelines

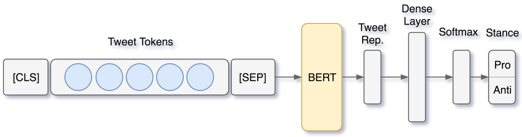


Figure 2: Fine-tuning BERT for stance classification.

Test Set For each topic, we randomly selected 200 users who have less than 5 tweets. The average number of tweets per user ranged across topics between 1.25 and 1.77 tweets. An annotator who is well versed with US politics manually examined the per topic tweets of users to determine their stances. If the tweets of a user were not sufficient to ascertain their stance, the annotator manually searched and examined their tweets on Twitter in an effort to find further clues. If no conclusive evidence of stance were found, the annotator skipped the user. The annotator labeled up to 100 users per topic. We asked another annotator to annotate 20 tweets per topic to ascertain inter-annotator agreement. Table 2 lists the percentage of skipped tweets, inter-annotator agreement, and percentage of pro- and anti-tweets. Next, we scraped the timeline of all the labeled users. Due to the time difference between collecting topical tweets and when we initiated the scraping of users' timelines, some user accounts were deleted, suspended, or made protected. Table 3 lists the number of labeled users and the subset of them for whom we were able to scrape their timelines. Thus, we put users for whom we were not able to collect timeline tweets into Set A, and we put the remaining users in Set B. We report results for both sets separately.

4 Classification Models

Supervised Classification As baselines, we used two different classification methods, namely a Support Vector Machines (SVM) classifier and a deep learning based text classifier. For the SVM classifier, we used the SVM^{Light} implementation with a linear kernel with default parameters (Joachims, 2002)[1]. We employed two feature types, namely: the accounts that users retweeted; and the words in tweets, including retweeted accounts, hashtags, and user mentions and replies. Prior work has shown that using retweeted accounts as features yields better results compared to using the content of tweet (Darwish et al., 2018). When using words in tweets, we tokenized tweets using NLTK (Bird et al., 2009), removed all URLs and emoticons, retained all hashtags and user mentions, and specifically delineated retweeted accounts by adding ’RT_’ before them. We chose to distinguish between retweeted accounts and user mentions because retweeting commonly signifies agreement and user mentions (including replies) may indicate opposition. We concatenated the aforementioned features from all the tweets of a user, and we constructed a feature vector, where the value of each unique feature was set to its frequency across all tweets of a user. For the deep learning based classifier, we used fastText, which is an efficient text classifier that has been shown to be effective for different text classification tasks (Joulin et al., 2016). Since fastText was designed for sentence-level classification, we opted to perform tweet-level classification. During training, we assigned the label of a user to all his/her tweets. During testing, we averaged per class confidence scores across all tweets for a user, and we assigned the label with the highest average confidence to the user. As for features, we used all the words in tweets, and we preprocessed tweets in the manner described earlier for

---

[1] http://svmlight.joachims.org/

SVM. We opted not use retweeted accounts only as the number of retweeted accounts was arbitrary for each user and fastText is not well suited for long input text.

Contextualized Embeddings Over the last several years, pre-trained embedding (Mikolov et al., 2013; Pennington et al., 2014) have helped achieve significant improvements in a wide range of classification tasks in natural language processing. Representing words as vectors in a low-dimensional continuous space and then using them for downstream tasks lowered the need for extensive manual feature engineering. However, these pre-trained vectors are static and fail to handle polysemous words, where different instances of a word have to share the same representation regardless of context. More recently, different deep neural language models have been introduced to create contextualized word representations that can cope with the issue of polysemy and the context-dependent nature of words. Models such as OpenAi GPT (Radford et al., 2018), ELMo (Peters et al., 2018), BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), and UMLFIT (Howard and Ruder, 2018), to name a few, have achieved groundbreaking results in many NLP classification and language understanding tasks. For this paper, we use $\text{BERT}_{\text{base-multilingual}}$[2] (referred to hereafter simply as BERT), which we fine-tune for stance detection, as this eliminates the need for heavily engineered task-specific architectures. BERT is pre-trained on Wikipedia text from 104 languages and comes with hundreds of millions of parameters. It contains an encoder with 12 Transformer blocks, hidden size of 768, and 12 self-attention heads. As shown in Fig. 2, We fine-tuned BERT by adding a fully-connected dense layer followed by a softmax output layer, minimizing the binary cross-entropy loss function for the training data. For all experiments, we used HuggingFace[3] transformer implementation with PyTorch[4] as it provides pre-trained weights and vocabulary. As for features, we used all the words in tweets that were preprocessed in the manner described earlier for SVM and fastText. Similar to fastText, we performed tweet-level classification, and we used the average softmax output scores per class across all tweets for a user to assign a label to a test user.

Unsupervised Classification For unsupervised classification, we used the same unsupervised classification method described earlier, which we used to prepare the training set. Specifically, we constructed a feature vector for each test user based on the accounts he/she retweeted, computed its similarity to all users in the training set, projected all the users in the training along with the test user into a lower dimensional space using UMAP, and lastly clustered the users using mean shift. We then labeled the test user using the majority label of the cluster in which the user appeared.

5 Experiments

We split users in our test set on the basis of whether we were able to crawl their timelines or not. Set A includes users for which we were not able to obtain their timeline tweets. Set B includes users for which we were able to collect their timeline tweets. We separated between them, because Set B would allow us to compare between setups that use timeline tweets with those that do not on identical users.

For Set A, we always trained on the training users with their on-topic tweets and the tested on the test users, who typically had less than 2 tweets on average. We used four different classification setups, namely using fastText, SVM with retweeted accounts as features (SVM_{RT}), SVM with all words as features (SVM_{TEXT}), and fine-tuned BERT embeddings with a dense neural layer and softmax output (BERT). We experimented with using the unsupervised method on Set A, but the unsupervised algorithm was not able to assign any test user to a cluster, mostly because the number of tweets and subsequently retweeted users per test user were too few. For Set B, we experimented with the same classifiers using four different conditions, namely: not expanding either training or test sets with users' timeline tweets; expanding the test set only; expanding the training set only; and expanding both the training and test sets.

6 Results and Discussion

For all experiments, we report on per topic accuracy ($\mathcal{A}$) and macro precision ($\mathcal{P}$), recall ($\mathcal{R}$), and F-measure ($\mathcal{F}$) across stances on a topic. Table 4

---

[2] We also experimented with different contextualized embedding, such as RoBERTa (Liu et al., 2019), Albert (Lan et al., 2019), and XLM (Lample and Conneau, 2019) and $\text{BERT}_{\text{base-multilingual}}$ performed the best.

[3] `https://github.com/huggingface/transformers`

[4] `https://pytorch.org/`

	Set A															
	fastText				SVM$_{RT}$				SVM$_{TEXT}$				BERT			
Topic	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Kavanaugh	83.5	83.2	83.2	83.2	73.0	**83.3**	70.6	69.2	76.0	76.3	71.5	72.4	**84.7**	81.6	**83.8**	**82.7**
Vaccine	88.7	73.1	93.8	78.2	**88.2**	44.1	50.0	46.9	87.0	43.5	50.0	46.5	85.7	**98.0**	**85.7**	**91.4**
Ilhan	87.0	87.7	86.4	86.7	65.1	79.7	64.3	59.5	52.4	26.2	50.0	34.4	**87.9**	**91.1**	**86.4**	**88.7**
Gun Control	92.3	91.8	90.9	91.3	65.4	75.0	73.5	65.3	72.7	79.6	77.5	72.6	**93.8**	**97.6**	**93.0**	**95.2**
Police Racism	94.7	89.8	94.0	91.7	94.9	97.1	85.7	90.2	83.0	41.5	50.0	45.4	**96.0**	**98.3**	**96.7**	**97.5**
Climate Change	95.8	93.8	95.3	94.5	82.9	90.9	62.5	65.0	81.6	90.2	62.5	64.6	**95.7**	**96.3**	**98.1**	**97.2**
Midterm	85.9	83.5	84.1	83.8	87.3	87.4	82.7	84.4	69.8	75.6	77.9	69.7	**90.2**	**92.2**	**93.7**	**92.9**
Immigration	84.4	83.8	84.0	83.	65.3	81.9	55.3	48.5	59.0	29.5	50.0	37.1	**89.5**	**91.2**	**91.2**	**91.2**
Average	89.0	85.8	88.9	86.6	77.8	79.9	68.1	66.1	72.7	57.8	61.2	55.3	**90.4**	**93.3**	**91.1**	**92.1**

Table 4: Results on Sets A (no expansion). The best results in a row are in bold.

reports the results on Sets A where we were not able to expand the test set using timeline tweets. As the results show, BERT yielded the best results in terms of $\mathcal{A}$, $\mathcal{P}$, $\mathcal{R}$, and $\mathcal{F}$ for most topics, with the highest overall averages across all scores. fastText trailed BERT, and SVM$_{TEXT}$ performed much worse. SVM$_{RT}$ performed better than SVM$_{TEXT}$. This suggests that BERT, which uses contextual embeddings, is effective in performing accurate stance detection, even when classifying users with a very small number of topical tweets. As for the Unsupervised method, using the unsupervised method was not able to assign any test user to a cluster, mostly because the number of tweets per test user were too few. Hence, we omitted the unsupervised method from Table 4. Table 5 shows the results on Set B, where we expanded the test, training, or either or both training and test user tweets using timeline tweets. The results suggest the following:

- For BERT and fastText, which rely on the content of the tweets, we achieved the best results with no expansion or when we only expanded the training set. The inclusion of non-topical tweets in the test set led to worse results overall. We suspect that is happened because of the mismatch between the training and test sets.
- For SVM$_{RT}$ and Unsupervised classification, which rely exclusively on whom users retweeted, the expansion of the test dramatically improved overall $\mathcal{A}$, $\mathcal{P}$, $\mathcal{R}$, and $\mathcal{F}$. The positive improvement for both after timeline expansion suggests that the accounts that a user retweets are a strong signal of stance across multiple topics, and stances on multiple topics are likely correlated. For example, a user who supported the Kavanaugh nomination was likely to vote republican in the midterm elections. For future work, we plan to examine cross topic classification.
- Similar to the results observed for Set A (4), when no expansion is used, BERT led to the best overall results. However, using unsupervised classification led to the best overall results across all setups, with expanding the test set only yielding slightly better results than expanding both the training and test sets. Expanding the test set only is significantly more efficient than expanding both training and test sets.
- Using unsupervised classification failed to all users in the test set for any topic when the test set was not expanded, mostly because the number of tweets per test user and subsequent number of retweeted accounts were too few.
- SVM$_{TEXT}$ yielded the worse results overall, despite the inclusion of all the features in the tweets, such as retweeted accounts, hashtags, words, etc. It seems that the inclusion of more features (compared to SVM$_{RT}$) confused the classifier leading to lower results.
- SVM$_{TEXT}$ and SVM$_{RT}$ led to the lowest results when we only expanded the training set. For both setups, the classifier classified all users as belonging to one of the stances or the other. Hence, $\mathcal{R}$ for one class was 100.0 and 0.0 for the other (with macro $\mathcal{R}$ = 50.0). We suspect that expanding the feature space in the training set confused the SVM classifier. Both setups are unusable.

We computed the standard deviation (SD) of all our measures across topics for every setup. Lower SD coupled with high $\mathcal{A}$ and $\mathcal{F}$ is desirable as they indicate the setup produces consistently high results across topics. Unsupervised classification yielded the lowest SD values and highest overall score. BERT and fastText with no expansion and SVM$_{RT}$ with expanded test set had slightly higher SD. Thus, if we are able to scrape a user's timeline tweets, it is advantageous to use a method that relies on which accounts a user retweets, with unsupervised classification producing the best results. As we will show in the error analysis, the success of

	fastText															
	No Expansion				Expanded Test only				Expanded Test and Train				Expanded Train only			
Topic	$\mathcal{A}$	P	R	F	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Kavanaugh	**82.8**	**82.5**	**82.3**	**82.4**	57.9	58.8	59.1	57.7	70.2	69.5	70.1	69.6	80.2	79.9	80.1	80.0
Vaccine	**86.4**	**68.3**	**85.0**	**72.3**	25.6	51.0	54.1	23.0	69.6	53.8	70.3	49.1	82.0	52.2	52.7	52.4
Ilhan	**85.9**	**89.4**	**85.3**	**85.4**	54.8	48.9	50.0	35.5	73.8	76.2	72.0	71.9	84.4	84.3	84.4	84.3
Gun Control	70.3	72.1	76.2	69.5	46.6	49.0	48.9	46.3	69.2	68.6	70.4	68.3	**77.5**	**77.5**	**78.6**	**76.0**
Police Racism	**88.0**	**81.8**	**90.3**	**84.5**	50.9	49.5	49.2	45.5	83.0	74.7	78.9	76.3	**88.0**	**81.0**	**83.1**	**82.0**
Climate Change	75.6	68.1	79.6	68.0	78.2	51.3	51.0	50.9	80.3	68.6	80.3	71.0	**87.6**	**80.7**	**87.6**	**83.2**
Midterm	84.9	79.4	82.2	80.6	42.6	49.5	49.3	41.7	81.0	74.5	77.4	75.7	**91.7**	**91.0**	**88.5**	**89.6**
Immigration	**87.5**	**87.7**	**87.7**	**87.5**	49.7	50.1	50.1	49.5	79.0	78.4	79.0	78.6	83.0	84.3	84.3	83.0
Average	82.6	**78.6**	**83.5**	**78.9**	50.8	51.0	51.5	43.8	65.8	70.5	74.8	70.1	**84.3**	**78.6**	79.9	**78.8**
Std Dev	6.0	7.8	4.2	7.0	13.9	3.1	3.3	9.9	5.3	7.2	4.2	8.6	4.3	10.8	10.8	10.6

	SVM$_{RT}$															
	No Expansion				Expanded Test only				Expanded Test and Train				Expanded Train only			
Topic	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Kavanaugh	73.7	78.9	72.5	71.7	84.9	84.9	83.8	84.2	**86.8**	**88.7**	**84.8**	**85.8**	52.6	26.3	50.0	34.5
Vaccine	95.5	47.7	50.0	48.8	**96.4**	**98.2**	**75.0**	**82.4**	**96.4**	**98.2**	**75.0**	**82.4**	95.5	47.7	50.0	48.8
Ilhan	57.1	77.3	55.9	45.8	**89.5**	**91.7**	**88.9**	**89.2**	71.1	82.3	69.4	67.2	51.4	25.7	50.0	34.0
Gun Control	66.7	72.7	76.9	66.3	75.0	**78.6**	**81.3**	74.8	**79.2**	76.9	75.0	**75.8**	72.2	36.1	50.0	41.9
Police Racism	90.6	94.6	78.6	83.5	**92.3**	88.0	**92.1**	**89.7**	90.4	**94.4**	79.2	83.9	78.1	39.1	50.0	43.9
Climate Change	92.3	96.0	70.0	76.5	**100.0**	**100.0**	**100.0**	**100.0**	96.2	97.9	85.7	90.6	87.2	43.6	50.0	46.6
Midterm	**95.5**	**97.2**	90.0	**93.0**	92.9	89.3	**91.9**	90.5	88.1	84.3	81.9	83.0	77.3	38.6	50.0	43.6
Immigration	59.5	78.6	55.9	46.9	**97.6**	**97.5**	**97.8**	**97.6**	73.8	79.4	71.5	71.0	54.1	27.0	50.0	35.1
Average	78.9	80.4	68.7	66.6	**91.1**	**91.0**	**88.8**	**88.6**	85.2	87.8	77.8	80.0	71.0	35.5	50.0	41.0
Std Dev	15.4	15.3	12.8	16.7	7.5	6.9	7.9	7.6	9.0	7.8	5.7	7.4	15.7	7.8	0.0	5.4

	SVM$_{TEXT}$															
	No Expansion				Expanded Test only				Expanded Test and Train				Expanded Train only			
Topic	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Kavanaugh	68.5	67.8	67.0	67.2	**70.9**	**71.3**	**69.0**	**69.2**	56.4	65.9	60.4	53.8	42.6	21.3	50.0	29.9
Vaccine	**92.9**	**46.4**	50.0	**48.2**	89.5	46.4	48.1	47.2	49.1	52.9	**61.1**	40.2	7.1	3.6	50.0	6.7
Ilhan	52.6	26.3	50.0	34.5	53.9	26.9	50.0	35.0	**71.8**	**72.2**	**72.2**	**71.8**	47.4	23.7	50.0	32.1
Gun Control	56.0	72.5	65.6	54.8	**69.2**	**76.5**	**76.5**	**69.2**	46.2	59.1	56.2	44.9	36.0	18.0	50.0	26.5
Police Racism	77.4	38.7	50.0	43.6	**85.2**	**81.0**	**72.6**	**75.5**	66.7	61.9	66.7	61.4	22.6	11.3	50.0	18.5
Climate Change	**92.5**	**96.0**	**71.4**	**77.9**	79.6	43.0	45.7	44.3	42.6	59.2	67.0	41.0	13.2	6.6	50.0	11.7
Midterm	61.4	68.5	75.0	60.4	80.0	71.3	62.1	64.0	**84.4**	**78.3**	**86.4**	**80.6**	22.7	11.4	50.0	18.5
Immigration	54.8	27.4	50.0	35.4	58.1	62.1	53.2	45.0	**86.1**	**87.7**	**84.8**	**85.4**	54.8	27.4	50.0	35.4
Average	69.5	55.5	59.9	52.7	**73.3**	**59.8**	**59.7**	**56.2**	62.9	67.2	69.3	59.9	30.8	15.4	50.0	22.4
Std Dev	15.4	23.1	10.2	14.3	11.8	17.8	11.2	14.0	15.9	10.8	10.4	16.7	15.9	7.9	0.0	9.5

	BERT															
	No Expansion				Expanded Test only				Expanded Test and Train				Expanded Train only			
Topic	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Kavanaugh	**82.5**	**79.**	**81.8**	**80.7**	60.9	51.1	69.5	58.9	70.9	62.2	71.0	66.3	80.5	76.9	78.9	77.9
Vaccine	89.1	**99.0**	88.7	93.6	38.9	96.7	37.5	54.0	73.5	98.3	73.6	84.2	**90.9**	98.2	**91.7**	**94.8**
Ilhan	91.6	**93.6**	**90.7**	92.1	61.7	62.7	74.4	68.1	76.0	72.9	89.5	80.4	**93.0**	92.1	**94.6**	**93.3**
Gun Control	**86.2**	88.9	**95.0**	**83.5**	88.8	70.8	60.1	65.0	70.4	83.9	67.3	74.7	73.0	**90.0**	69.2	78.3
Police Racism	**93.0**	**96.7**	94.4	**95.5**	67.6	86.0	70.7	77.6	83.4	93.6	84.9	89.0	91.6	93.9	**95.4**	94.7
Climate Change	**94.1**	**97.4**	**95.0**	**96.2**	89.0	75.0	75.0	81.4	71.0	92.8	71.6	80.8	80.5	93.3	82.4	87.5
Midterm	91.7	93.3	95.1	94.2	57.8	81.6	57.9	67.7	82.3	92.2	84.0	87.9	**94.3**	**95.1**	**97.5**	**96.3**
Immigration	88.1	92.4	86.0	89.1	65.3	72.9	65.3	68.9	81.3	84.3	83.8	84.0	**92.6**	**97.4**	**88.4**	**92.7**
Average	**89.6**	**93.4**	**89.4**	**91.3**	62.4	74.6	63.8	67.7	76.1	85.0	78.2	80.9	87.0	92.1	87.3	89.4
Std Dev	8.8	7.2	11.4	8.6	12.9	13.1	11.5	8.4	5.1	11.4	7.7	7.0	7.4	6.3	9.1	7.0

	Unsupervised									
	No Expansion	Expanded Test only				Expanded Test and Train				Expanded Train only
Topic		$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	
Kavanaugh		84.6	84.2	84.2	84.2	**90.4**	**89.2**	**91.3**	**89.9**	
Vaccine		**96.3**	83.3	**99.0**	**89.5**	95.3	**97.1**	75.0	81.8	
Ilhan		**91.9**	**91.6**	**92.1**	**91.8**	**91.9**	**91.6**	**92.1**	**91.8**	
Gun Control		**95.8**	**90.6**	**86.4**	**86.9**	**87.5**	**90.6**	**86.4**	**86.9**	
Police Racism		**100.0**	**100.0**	**100.0**	**100.0**	98.0	98.7	95.8	97.2	
Climate Change		**96.2**	**97.8**	**88.9**	**92.6**	**92.3**	**96.7**	**85.0**	**89.5**	
Midterm		**100.0**	**100.0**	**100.0**	**100.0**	**95.0**	**96.8**	**90.9**	**93.3**	
Immigration		**100.0**	**100.0**	**100.0**	**100.0**	**97.6**	**97.8**	**97.5**	**97.6**	
Average		**95.6**	93.4	**93.8**	**93.1**	93.5	**94.8**	89.3	91.0	
Std Dev		4.9	6.6	6.3	5.9	3.4	3.5	6.7	4.9	

Table 5: Results on Set B with and without expansion of either training or test sets. Highest $\mathcal{A}$, $\mathcal{P}$, $\mathcal{R}$, and $\mathcal{F}$ per method are bolded, and highest values overall are underlined. The table reports the average and standard deviation.

Error Type	No.	Examples
Unexplained	52	*Ilhan*: RIP CONS ... When they see me they take their hat off... I am DJ *Gun Control*: @BreitbartNews @NRA more lies *Climate Change*: $200 million a year could reverse climate change
Vague	48	*Kavanaugh*: following the senate confirmation vote *Midterm*: RT_@ali irresponsible for twitter *Police & Racism*: after 40 yrs of reflection
Quoting other side	24	*Immigration*: this is a real crisis at the border *Kavanaugh*: why did jeff flake demand an investigation and then accept a bogus one
Sarcasm	4	*Immigration*: RT_@infantry0300 someone should let "Ms Hitler was a really great guy until he crossed the border into Poland"

Table 6: BERT error types with examples

unsupervised classification is contingent on users retweeting a sufficient number of times, particularly politically related accounts in our case. When timeline tweets are not available, it is best to use contextualized embeddings.

Error Analysis We analyzed all the errors in Set B that were produced by BERT with no expansion, as it represents the best results when expansion is not possible, and those produced by unsupervised classification with the expansion of the test set only, as this produced the best overall results. Since we used BERT to perform tweet-level classification, we manually inspected all 129 misclassified tweets across all topics. Generally we found four types of errors, namely: unexplainable errors where the tweets clearly expressed stance, but the classifier mislabeled them; vague tweets that have no clear clues; tweets in which the user uses the language of the opposing side; and sarcastic tweets. Table 6 lists the error types with their frequencies and provides example tweets. When we used LIME (Ribeiro et al., 2016) to analyze the output of BERT, we noticed two important phenomena. First, BERT was able to identify stance based on retweeted accounts, and not just the text of the tweet. Second, BERT was able to learn correlations between topic. For example, the tweet "RT ***BernieSanders***: I believe ***health care*** must be a right", BERT based its decision on BernieSanders, the democratic presidential hopeful in 2016 and 2020, and on "health care", where positions on *health care* and *climate change* are often aligned.

For unsupervised classification, we manually examined all 15 users that were missclassified, of whom 7 were from the Kavanaugh topic. Prominent reasons for incorrect classification were: Lack of sufficient retweets for a user, where the percentage of retweets ranged between 1-1.4% of all tweets for three of the misclassified users (2 for *climate change* and 1 for *Kavanaugh*); Geographic mislabeling, where 2 accounts were not US accounts (1 for *Ilhan* and 1 for *Kavanaugh*); Users retweeting mostly apolitical accounts such as music, art, or cars related accounts (1 for *climate change*, 1 for *vaccine*, and 3 for *Kavanaugh*) – retweeting of politically biased accounts and media sources seem to provide strong signals for classification; or User goes against the general opinion of his group as in the clearly republican user who was criticizing the National Rifle Association (NRA) (*gun control*).

Thus, the most common reason for misclassification was the dearth of retweets from politically oriented or topically related accounts.

7 Conclusion

In this paper, we presented two methods for classifying users according to their stance towards a target. The first utilizes contextualized embeddings to represent tweets and then uses deep neural network for classification. This approach led to results that outperform two strong baselines. The second utilizes additional tweets from users' timelines to cluster test users with other users with known labels in an unsupervised manner. The first method yielded the best results when timeline tweets were not available, while the second yielded even better results overall. Given the overall setup described in the paper, where the training data was obtained using unsupervised user classification, we can automatically label the most active users with nearly perfect accuracy, and we can label users with only few topical tweets with high accuracy, often above 95% when we can obtain their timeline tweets. For future work, we plan to explore the effectiveness of cross topic classification, where training and testing are done on different topics. Perhaps, we can build unified models that could be used across multiple topics for a given a population of Twitter users (ex. users who are interested in US politics).

References

Abeer Aldayel and Walid Magdy. 2019. Your stance is exposed! analysing possible factors for stance detection on social media. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW).

Pablo Barberá. 2015. Birds of the same feather tweet together: Bayesian ideal point estimation using Twitter data. *Political Analysis*, 23(1):76–91.

Pablo Barberá and Gonzalo Rivero. 2014. Understanding the political representativeness of Twitter users. *Social Science Computer Review*, page 0894439314558836.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Javier Borge-Holthoefer, Walid Magdy, Kareem Darwish, and Ingmar Weber. 2015. Content and network dynamics behind Egyptian political polarization on Twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 700–711.

Raviv Cohen and Derek Ruths. 2013. Classifying political orientation on Twitter: It's not easy! In *ICWSM 2013*.

Elanor Colleoni, Alessandro Rozza, and Adam Arvidsson. 2014. Echo chamber or public sphere? Predicting political orientation and measuring political homophily in Twitter using big data. *Journal of Communication*, 64(2):317–332.

Michael D Conover, Bruno Gonçalves, Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. 2011. Predicting the political alignment of Twitter users. In *Proceedings of the 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom)*, pages 192–199.

Kareem Darwish, Walid Magdy, Afshin Rahimi, Timothy Baldwin, Norah Abokhodair, et al. 2018. Predicting online islamophobic behavior after# parisattacks. *The Journal of Web Science*, 4(3):34–52.

Kareem Darwish, Peter Stefanov, Michaël Aupetit, and Preslav Nakov. 2020. Unsupervised user stance detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 141–152.

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

James H Fowler, Michael T Heaney, David W Nickerson, John F Padgett, and Betsy Sinclair. 2011. Causality in political networks. *American Politics Research*, 39(2):437–480.

Keinosuke Fukunaga and Larry Hostetler. 1975. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory*, 21(1):32–40.

Itai Himelboim, Stephen McCreery, and Marc Smith. 2013. Birds of a feather tweet together: Integrating network and content analyses to examine cross-ideology exposure on Twitter. *Journal of Computer-Mediated Communication*, 18(2):40–60.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

T. Joachims. 2002. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer/Springer.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Walid Magdy, Kareem Darwish, Norah Abokhodair, Afshin Rahimi, and Timothy Baldwin. 2016a. #isisisnotislam or #deportallmuslims?: Predicting unspoken views. In *Proceedings of the 8th ACM Conference on Web Science*, pages 95–106. ACM.

Walid Magdy, Kareem Darwish, and Ingmar Weber. 2016b. #failedrevolutions: Using Twitter to study the antecedents of ISIS support. *First Monday*, 21(2).

Aibek Makazhanov, Davood Rafiei, and Muhammad Waqar. 2014. Predicting political preference of Twitter users. *Social Network Analysis and Mining*, 4(1):1–15.

Leland McInnes and John Healy. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016a. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.

Saif M Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016b. Semeval-2016 task 6: Detecting stance in tweets. In *SemEval 2016*, volume 16.

Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, Republicans and Starbucks afficionados: user classification in Twitter. In *Proceedings of ACM SIGKDD*, pages 430–438.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *ArXiv*, abs/1802.05365.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2019. STANCY: Stance classification based on consistency cues. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6412–6417, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144.

Peter Stefanov, Kareem Darwish, Atanas Atanasov, and Preslav Nakov. 2019. Predicting the topical stance of media and popular twitter users. *arXiv preprint arXiv:1907.01260*.

Peter Stefanov, Kareem Darwish, Atanas Atanasov, and Preslav Nakov. 2020. Predicting the topical stance and political leaning of media using tweets. *The 58th Annual Meeting of the Association for Computational Linguistics (ACL'2020)*.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335, Sydney, Australia. Association for Computational Linguistics.

Ingmar Weber, Venkata R Kiran Garimella, and Alaa Batayneh. 2013. Secular vs. islamist polarization in Egypt on Twitter. In *ASONAM 2013*, pages 290–297.

Felix Ming Fai Wong, Chee Wei Tan, Soumya Sen, and Mung Chiang. 2013. Quantifying political leaning from tweets and retweets. In *ICWSM 2013*, pages 640–649.

Guido Zarrella and Amy Marsh. 2016. Mitre at semeval-2016 task 6: Transfer learning for stance detection. *arXiv preprint arXiv:1606.03784*.

Do Syntax Trees Help Pre-trained Transformers Extract Information?

Devendra Singh Sachan^{1,2}, **Yuhao Zhang**^{3}, **Peng Qi**^{3}, **William Hamilton**^{1,2}
^{1}Mila - Quebec AI Institute
^{2}School of Computer Science, McGill University
^{3}Stanford University
`sachande@mila.quebec, wlh@cs.mcgill.ca`
`{yuhaozhang, pengqi}@stanford.edu`

Abstract

Much recent work suggests that incorporating syntax information from dependency trees can improve task-specific transformer models. However, the effect of incorporating dependency tree information into *pre-trained* transformer models (e.g., BERT) remains unclear, especially given recent studies highlighting how these models implicitly encode syntax. In this work, we systematically study the utility of incorporating dependency trees into pre-trained transformers on three representative information extraction tasks: semantic role labeling (SRL), named entity recognition, and relation extraction.

We propose and investigate two distinct strategies for incorporating dependency structure: a *late fusion* approach, which applies a graph neural network on the output of a transformer, and a *joint fusion* approach, which infuses syntax structure into the transformer attention layers. These strategies are representative of prior work, but we introduce additional model design elements that are necessary for obtaining improved performance. Our empirical analysis demonstrates that these syntax-infused transformers obtain state-of-the-art results on SRL and relation extraction tasks. However, our analysis also reveals a critical shortcoming of these models: we find that their performance gains are highly contingent on the availability of human-annotated dependency parses, which raises important questions regarding the viability of syntax-augmented transformers in real-world applications.[1]

1 Introduction

Dependency trees—a form of syntactic representation that encodes an asymmetric syntactic relation between words in a sentence, such as *subject* or *adverbial modifier*—have proven very useful in various NLP tasks. For instance, features defined in terms of the shortest path between entities in a dependency tree were used in relation extraction (RE) (Fundel et al., 2006; Björne et al., 2009), parse structure has improved named entity recognition (NER) (Jie et al., 2017), and joint parsing was shown to benefit semantic role labeling (SRL) (Pradhan et al., 2005) systems. More recently, dependency trees have also led to meaningful performance improvements when incorporated into neural network models for these tasks. Popular encoders to include dependency tree into neural models include graph neural networks (GNNs) for SRL (Marcheggiani and Titov, 2017) and RE (Zhang et al., 2018), and biaffine attention in transformers for SRL (Strubell et al., 2018).

In parallel, there has been a renewed interest in investigating self-supervised learning approaches to pre-training neural models for NLP, with recent successes including ELMo (Peters et al., 2018), GPT (Radford et al., 2018), and BERT (Devlin et al., 2019). Of late, the BERT model based on pre-training of a large transformer model (Vaswani et al., 2017) to encode bidirectional context has emerged as a dominant paradigm, thanks to its improved modeling capacity which has led to state-of-the-art results in many NLP tasks.

BERT's success has also attracted attention to what linguistic information its internal representations capture. For example, Tenney et al. (2019) attribute different linguistic information to different BERT layers; Clark et al. (2019) analyze BERT's attention heads to find syntactic dependencies; Hewitt and Manning (2019) show evidence that BERT's hidden representation embeds syntactic trees. However, it remains unclear if this linguistic information helps BERT in downstream tasks during finetuning or not. Further, it is not evident if external syntactic information can further improve

---

[1] Our code is available at: `https://github.com/DevSinghSachan/syntax-augmented-bert`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2647–2661
April 19 - 23, 2021.

BERT's performance on downstream tasks.

In this paper, we investigate the extent to which pre-trained transformers can benefit from integrating external dependency tree information. We perform the first systematic investigation of how dependency trees can be incorporated into pre-trained transformer models, focusing on three representative information extraction tasks where dependency trees have been shown to be particularly useful for neural models: semantic role labeling (SRL), named entity recognition (NER), and relation extraction (RE).

We propose two representative approaches to integrate dependency trees into pre-trained transformers (i.e., BERT) using syntax-based graph neural networks (syntax-GNNs). The first approach involves a sequential assembly of a transformer and a syntax-GNN, which we call *Late Fusion*, while the second approach interleaves syntax-GNN embeddings within transformer layers, termed *Joint Fusion*. These approaches are inspired by recent work that combines transformers with external input, but we introduce design elements such as the alignment between dependency tree and BERT wordpieces that lead to obtaining strong performance. Comprehensive experiments using these approaches reveal several important insights:

- Both our syntax-augmented BERT models achieve a new state-of-the-art on the CoNLL-2005 and CoNLL-2012 SRL benchmarks when the gold trees are used, with the best variant outperforming a fine-tuned BERT model by over 3 F_1 points on both datasets. The Late Fusion approach also provides performance improvements on the TACRED relation extraction dataset.
- These performance gains are consistent across different pre-trained transformer approaches of different sizes (i.e. $\text{BERT}_{\text{BASE/LARGE}}$ and $\text{RoBERTa}_{\text{BASE/LARGE}}$).
- The Joint Fusion approach that interleaves GNNs with BERT achieves higher performance improvements on SRL, but it is also less stable and more prone to errors when using noisy dependency tree inputs such as for the RE task, where Late Fusion performs much better, suggesting complementary strengths from both approaches.
- In the SRL task, the performance gains of both approaches are highly contingent on the availability of human-annotated parses for both training and inference, without which the performance gains are either marginal or non-existent. In the NER task, even the gold trees don't show performance improvements.

Although our work does obtain new state-of-the-art results on SRL tasks by introducing dependency tree information from syntax-GNNs into BERT, however, our most important result is somewhat negative and cautionary: the performance gains are only substantial when human-annotated parses are available. Indeed, we find that even high-quality automated parses generated by domain-specific parsers do not suffice, and we are only able to achieve meaningful gains with human-annotated parses. This is a critical finding for future work—especially for SRL—as researchers routinely develop models with human-annotated parses, with the implicit expectation that models will generalize to high-quality automated parses.

Finally, our analysis provides indirect evidence that pre-trained transformers do incorporate sufficient syntactic information to achieve strong performance on downstream tasks. While human-annotated parses can still help greatly, with our proposed models it appears that the knowledge in automatically extracted syntax trees is largely redundant with the implicit syntactic knowledge learned by pre-trained models such as BERT.

2 Models

In this section, we will first briefly review the transformer encoder, then describe the graph neural network (GNN) that learns syntax representations using dependency tree input, which we term the syntax-GNN. Next, we will describe our syntax-augmented BERT models that incorporate such representations learned from the GNN.

2.1 Transformer Encoder

The transformer encoder (Vaswani et al., 2017) consists of three core modules in sequence: embedding layer, multiple encoder layers, and a task-specific output layer. The core elements in these modules are different sets of learnable weight matrices that perform linear transformations. The embedding layer consists of wordpiece embeddings, positional embeddings, and segment embeddings (Devlin et al., 2019). After embedding lookup, these three embeddings are added to obtain token embeddings for an input sentence. The encoder layers then transform the input token embeddings to hidden state representations. The encoder layer con-

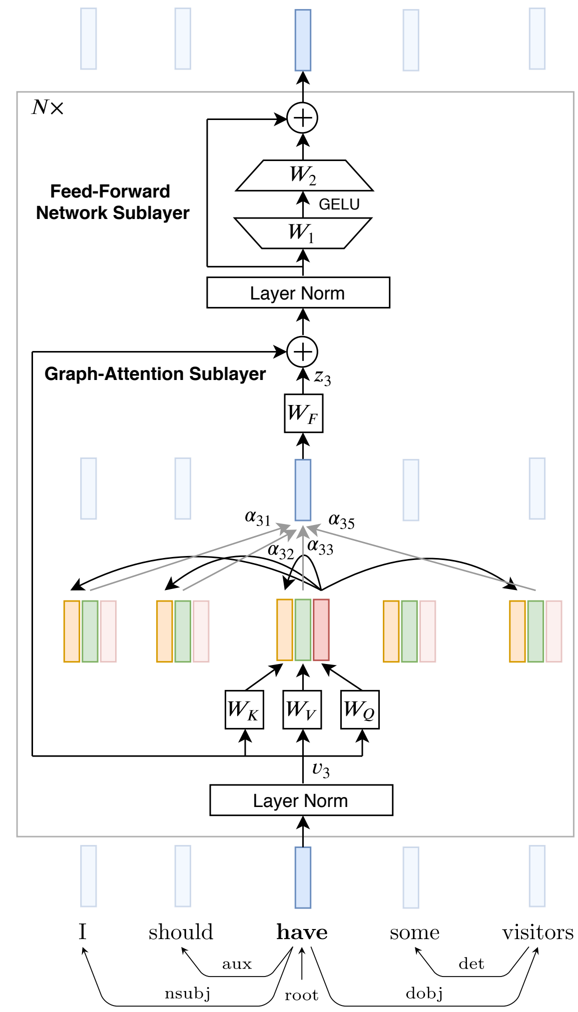


Figure 1: Block diagram illustrating syntax-GNN applied over a sentence's dependency tree. In the example shown, for the word "*have*", the graph-attention sublayer aggregates representations from its three adjacent nodes in the dependency graph.

sists of two sublayers: multi-head dot-product self-attention and feed-forward network, which will be covered in the following section. Finally, the output layer is task-specific and consists of one layer feed-forward network.

2.2 Syntax-GNN: Graph Neural Network over a Dependency Tree

A dependency tree can be considered as a multi-attribute directed graph where the nodes represent words and the edges represent the dependency relation between the head and dependent words. To learn useful syntax representations from the dependency tree structure, we apply graph neural networks (GNNs) (Hamilton et al., 2017; Battaglia et al., 2018) and henceforth call our model *syntax-GNN*. Our syntax-GNN encoder as shown in Figure 1 is a variation of the transformer encoder where the self-attention sublayer is replaced by *graph attention* (Veličković et al., 2018). Self-attention can also be considered as a special case of graph-attention where each word is connected to all the other words in the sentence.

Let $V = \{v_i \in \mathbb{R}^d\}_{i=1:\mathcal{N}^v}$ denote the input node embeddings and $E = \{(e_k, i, j)_{k=1:\mathcal{N}^e}\}$ denote the edges in the dependency tree, where the edge e_k is incident on nodes i and j. Each layer in our syntax-GNN encoder consists of two sublayers: graph attention and feed-forward network.

First, interaction scores (s_{ij}) are computed for all the edges by performing dot-product on the adjacent linearly transformed nodes embeddings

$$s_{ij} = (v_i \boldsymbol{W_Q})(v_j \boldsymbol{W_K})^\top. \qquad (1)$$

The terms $v_i \boldsymbol{W_Q}$ and $v_i \boldsymbol{W_K}$ are also known as *query* and *key* vectors respectively. Next, an attention score (α_{ij}) is computed for each node by applying softmax over the interaction scores from all its connecting edges:

$$\alpha_{ij} = \frac{\exp(s_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(s_{ik})}, \qquad (2)$$

where $\mathcal{N}_i$ refers to the set of nodes connected to i^{th} node. The graph attention output (z_i) is computed by the aggregation of attention scores followed by a linear transformation:

$$z_i = (\sum_{j \in \mathcal{N}_i} \alpha_{ij}(v_j \boldsymbol{W_V}))\boldsymbol{W_F}. \qquad (3)$$

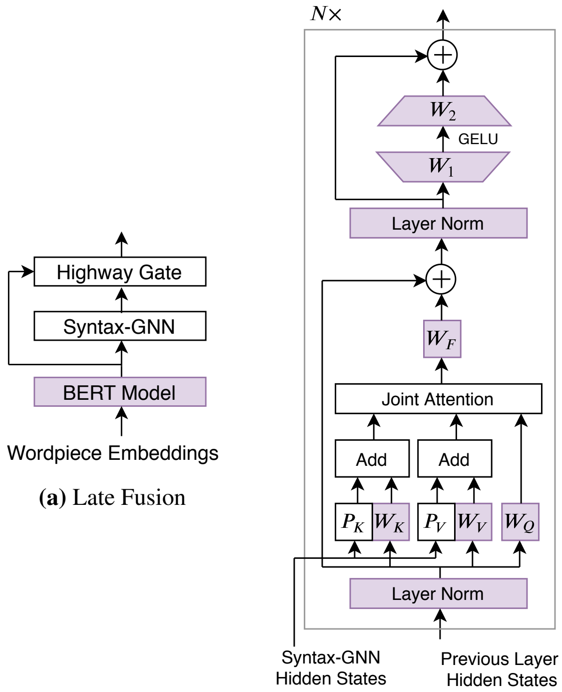
The term $v_j \boldsymbol{W_V}$ is also referred to as *value* vector. Subsequently, the message (z_i) is passed to the second sublayer that consists of two layer fully connected feed-forward network with GELU activation (Hendrycks and Gimpel, 2016).

$$\mathrm{FFN}(z_i) = \mathrm{GELU}(z_i \boldsymbol{W_1} + b_1)\boldsymbol{W_2} + b_2\ . \qquad (4)$$

The FFN sublayer outputs are given as input to the next layer. In the above equations $\boldsymbol{W_K}$, $\boldsymbol{W_V}$, $\boldsymbol{W_Q}$, $\boldsymbol{W_F}$, $\boldsymbol{W_1}$, $\boldsymbol{W_2}$ are trainable weight matrices and b_1, b_2 are bias parameters. Additionally, layer normalization (Ba et al., 2016) is applied to the input and residual connections (He et al., 2016) are added to the output of each sublayer.

2.2.1 Dependency Tree over Wordpieces

As BERT models take as input subword units (also known as wordpieces) instead of linguistic tokens,



(a) Late Fusion

(b) Joint Fusion

Figure 2: Block diagrams illustrating our proposed syntax-augmented BERT models. Weights shown in color are pre-trained while those not colored are either non-parameterized operations or have randomly initialized weights. The inputs to each of these models are wordpiece embeddings while their output goes to task-specific output layers. In subfigure 2b, $N\times$ indicates that there are N layers, with each of them being passed the same set of syntax-GNN hidden states.

this also necessitates to extend the definition of a dependency tree to include wordpieces. For this, we introduce additional edges in the original dependency tree by defining new edges from the first subword (head word) of a token to the remaining subwords (tail words) of the same token.

2.3 Syntax-Augmented BERT

In this section, we propose parameter augmentations over the BERT model to best incorporate syntax information from a syntax-GNN. To this end, we introduce two models— Late Fusion and Joint Fusion. These models represent novel mechanisms—inspired by previous work—through which syntax-GNN features are incorporated at different sublayers of BERT (Figure 2). We refer to these models as Syntax-Augmented BERT (SA-BERT) models. During the finetuning step, the new parameters in each model are randomly initialized while the existing parameters are initialized from pre-trained BERT.

Late Fusion: In this model, we feed the BERT contextual representations to the syntax-GNN encoder *i.e.* syntax-GNN is stacked over BERT (Figure 2a). We also use a Highway Gate (Srivastava et al., 2015) at the output of the syntax-GNN encoder to adaptively select useful representations for the training task. Concretely, if v_i and z_i are the representations from BERT and syntax-GNN respectively, then the output (h_i) after the gating layer is computed as,

$$g_i = \sigma(\boldsymbol{W}_g v_i + b_g) \tag{5}$$

$$h_i = g_i \odot v_i + (1 - g_i) \odot z_i, \tag{6}$$

where σ is the sigmoid function $1/\left(1 + e^{-x}\right)$ and $\boldsymbol{W}_g$ is a learnable parameter. Finally, we map the output representations to linguistic space by adding the hidden states of all the wordpieces that map to the same linguistic token respectively.

Joint Fusion: In this model, syntax representations are incorporated within the self-attention sublayer of BERT. The motivation is to *jointly* attend over both syntax- and BERT representations. First, the syntax-GNN representations are computed from the input token embeddings and its final layer hidden states are passed to BERT. Second, as is shown in Figure 2b, the syntax-GNN hidden states are linearly transformed using weights $\boldsymbol{P}_K$, $\boldsymbol{P}_V$ to obtain additional *syntax-based* key and value vectors. Third, syntax-based key and value vectors are added with the BERT's self-attention sublayer key and value vectors respectively. Fourth, the query vector in self-attention layer now attends over this set of keys and values, thereby augmenting the model's ability to fuse syntax information. Overall, in this model, we introduce two new set of weights per layer $\{\boldsymbol{P}_K, \boldsymbol{P}_V\}$, which are randomly initialized.

3 Experimental Setup

3.1 Tasks and Datasets

For our experiments, we consider information extraction tasks for which dependency trees have been extensively used in the past to improve model performance. Below, we provide a brief description of these tasks and the datasets used and refer the reader to Appendix A.1 for full details.

Semantic Role Labeling (SRL) In this task, the objective is to assign semantic role labels to text spans in a sentence such that they answer the query:

Who did *what* to *whom* and *when*? Specifically, for every target *predicate* (verb) of a sentence, we detect *syntactic constituents* (arguments) and classify them into predefined semantic roles. In our experiments, we study the setting where the predicates are given and the task is to predict the arguments. We use the CoNLL-2005 SRL corpus (Carreras and Màrquez, 2005) and CoNLL-2012 OntoNotes[2] dataset, which contains PropBank-style annotations for predicates and their arguments, and also includes POS tags and constituency parses.

Named Entity Recognition (NER) NER is the task of recognizing entity mentions in text and tagging them to entity categories. We use the OntoNotes 5.0 dataset (Pradhan et al., 2012), which contains 18 named entity types.

Relation Extraction (RE) RE is the task of predicting the relation between the two entity mentions in a sentence. We use the label corrected version of the TACRED dataset (Zhang et al., 2017; Alt et al., 2020), which contains 41 relation types as well as a special *no_relation* class indicating that no relation exists between the two entities.

3.2 Training Details

We select *bert-base-cased* to be our reference pretrained baseline model.[3] It consists of 12 layers, 12 attention heads, and 768 model dimensions. In both the variants, the syntax-GNN component consists of 4 layers, while other configurations are kept the same as *bert-base*. Also, for the Joint Fusion method, syntax-GNN hidden states were shared across different layers. It is worth noting that as our objective is to assess if the use of dependency trees can provide performance gains over pre-trained transformer models, it is important to tune the hyperparameters of these baseline models to obtain strong reference scores. Therefore, for each task, during the finetuning step, we tune the hyperparameters of the default *bert-base* model and use the same hyperparameters to train the SA-BERT models. We refer the reader to Appendix A.2 for additional training details.

4 Results and Analysis

In this section, we present our main empirical analyses and key findings.

Test Set	**P**	**R**	**F_1**
Baseline Models (without dependency parses)			
SA+GloVe†	84.17	83.28	83.72
SA+ELMo†	86.21	85.98	86.09
BERT$_{\text{BASE}}$	86.97	88.01	87.48
Gold Dependency Parses			
Late Fusion	89.17	91.09	90.12
Joint Fusion	90.59	91.35	**90.97**

Table 1: SRL results on the CoNLL-2005 WSJ test set averaged over 5 independent runs. † marks results from Strubell et al. (2018).

Test Set	**P**	**R**	**F_1**
Baseline Models (without dependency parses)			
SA+GloVe†	82.55	80.02	81.26
SA+ELMo†	84.39	82.21	83.28
Deep-LSTM+ELMo‡	-	-	84.60
Structure-distilled BERT*	-	-	86.39
BERT$_{\text{BASE}}$	85.91	87.07	86.49
Gold Dependency Parses			
Late Fusion	88.06	90.32	89.18
Joint Fusion	89.34	90.44	**89.89**

Table 2: SRL results on the CoNLL-2012 test set averaged over 5 independent runs. † marks results from Strubell et al. (2018); ‡ mark result from Peters et al. (2018); ∗ mark result from Kuncoro et al. (2020).

4.1 Benchmark Performance

To recap, our two proposed variants of the Syntax-Augmented BERT models in Section 2.3 mainly differ at the position where syntax-GNN outputs are fused with the BERT hidden states. Following this, we first compare the effectiveness of these variants on all the three tasks, comparing against previous state-of-the-art systems such as (Strubell et al., 2018; Jie and Lu, 2019; Zhang et al., 2018), which are outlined in Appendix B due to space limitations. For this part we use gold dependency parses to train the models for SRL and NER, and predicted parses for RE, since gold dependency parses are not available for TACRED.

We present our main results for SRL in Table 1 and Table 2, NER in Table 3, and RE in Table 4. All these results report average performance over five runs with different random seeds. First, we note that for all the tasks, our *bert-base* baseline is quite strong and is directly competitive with other state-of-the-art models.

We observe that both the Late Fusion and Joint Fusion variants of our approach yielded the best results in the SRL tasks. Specifically, on CoNLL-

---

[2] `conll.cemantix.org/2012/data.html`

[3] *bert-base* configuration was preferred due to computational reasons and we found that *bert-cased* provided substantial gains over *bert-uncased* in the NER task.

Test Set	P	R	F_1
Baseline Models (without dependency parses)			
BiLSTM-CRF+ELMo†	88.25	89.71	88.98
$\text{BERT}_{\text{BASE}}$	88.75	89.61	89.18
Gold Dependency Parses			
DGLSTM-CRF+ELMo†	89.59	90.17	**89.88**
Late Fusion	88.75	89.19	88.97
Joint Fusion	88.58	89.31	88.94

Table 3: NER results on the OntoNotes-5.0 test set averaged over 5 independent runs. † marks results from Jie and Lu (2019).

Test Set	P	R	F_1
Baseline Models (without dependency parses)			
$\text{BERT}_{\text{BASE}}$	78.04	76.36	77.09
Stanford CoreNLP Dependency Parses			
GCN†	74.2	69.3	71.7
$\text{GCN+BERT}_{\text{BASE}}$†	74.8	74.1	74.5
Late Fusion	78.55	76.29	**77.38**
Joint Fusion	70.22	75.12	72.52

Table 4: Relation extraction results on the revised TACRED test set (Alt et al., 2020), averaged over 5 independent runs. † marks results reported by Alt et al. (2020).

2005 and CoNLL-2012 SRL, Joint Fusion improves over *bert-base* by an absolute 3.5 F_1 points, while Late Fusion improves over *bert-base* by 2.65 F_1 points. On the RE task, the Late Fusion model improves over *bert-base* by approximately 0.3 F_1 points while the Joint Fusion model leads to a drop of 4.5 F_1 points in performance (which we suspect is driven by the longer sentence lengths observed in TACRED). On NER, the SA-BERT models lead to no performance improvements as their scores lies within one standard deviation to that of *bert-base*.

Overall, we find that syntax information is most useful to the pre-trained transformer models in the SRL task, especially when intermixing the intermediate representations of BERT with representations from the syntax-GNN. Moreover, when the fusion is done after the final hidden layer of the pre-trained models, apart from providing good gains on SRL, it also provides small gains on RE task. We further note that, as we trained all our syntax-augmented BERT models using the same hyperparameters as that of *bert-base*, it is possible that separate hyperparameter tuning would further improve their performance.

4.2 Impact of Parsing Quality

In this part, we study to what extent parsing quality can affect the performance results of the syntax-augmented BERT models. Specifically, following existing work, we compare the effect of using parse trees from three different sources: (a) gold syntactic annotations[4]; (b) a dependency parser trained using gold, *in-domain* parses[5]; and (c) available off-the-shelf NLP toolkits.[6] In previous work, it was shown that using in-domain parsers can provide good improvements on SRL (Strubell et al., 2018) and NER tasks (Jie and Lu, 2019), and the performance can be further improved when gold parses were used at test time. Meanwhile, in many practical settings where gold parses are not readily available, the only option is to use parse trees produced by existing NLP toolkits, as was done by Zhang et al. (2018) for RE. In these cases, since the parsers are trained on a different domain of text, it is unclear if the produced trees, when used with the SA-BERT models, can still lead to performance gains. Motivated by these observations, we investigate to what extent *gold*, *in-domain*, and *off-the-shelf* parses can improve performance over strong BERT baselines.

Comparing off-the-shelf and gold parses. We report our findings on the CoNLL-2005 SRL (Table 5), CoNLL-2012 SRL (Table 6), and OntoNotes-5.0 NER (Table 7) tasks. Using gold parses, both the Late Fusion and Joint Fusion models obtain greater than 2.5 F_1 improvement on SRL tasks compared with *bert-base* while we don't observe significant improvements on NER. We further note that as the gold parses are produced by expert human annotators, these results can be considered as the attainable performance ceiling from using parse trees in these models.

We also observe that using off-the-shelf parses from the Stanza toolkit (Qi et al., 2020) provides little to no gains in F_1 scores (see Tables 5 and 7). This is mainly due to the low in-domain accuracy of the predicted parses. For example, on the CoNLL-

---

[4] We use Stanford *head rules* (de Marneffe and Manning, 2008) implemented in Stanford CoreNLP v4.0.0 (Manning et al., 2014) to convert constituency trees to dependency trees in UDv2 format (Nivre et al., 2020).

[5] The difference between settings (a) and (b) is during test time. In (a) gold parses are used for both training and test instances while in (b) gold parses are used for training, while during test time, parses are extracted from a dependency parser which was trained using gold parses.

[6] In this setting, the parsers are trained on general datasets such as the Penn Treebank or the English Web Treebank.

Test Set	P	R	F_1
Stanza Dependency Parses (UAS: 84.20)			
Late Fusion	86.85	88.06	87.45
Joint Fusion	86.87	87.85	87.36
In-domain Dependency Parses (UAS: 92.66)			
LISA+GloVe†	85.53	84.45	84.99
LISA+ELMo†	87.13	86.67	86.90
Late Fusion	86.80	87.98	87.39
Joint Fusion	87.09	87.95	87.52
Gold Dependency Parses			
Late Fusion	89.17	91.09	90.12
Joint Fusion	90.59	91.35	**90.97**

Table 5: SRL results with different parses on the CoNLL-2005 WSJ test set averaged over 5 independent runs. † marks results from Strubell et al. (2018).

2005 SRL test set the UAS is 84.2% for the Stanza parser, which is understandable as the parser was trained on the EWT corpus which covers a different domain.

In a more fine-grained error analysis, we also examined the correlation between parse quality and performance on individual examples on CoNLL-2005 (Figures 3a and 3b), finding a mild but significant positive correlation between parse quality and relative model performance when training and testing with Stanza parses (Figure 3a). Interestingly, we found that this correlation between parse quality and validation performance is much stronger when we train a model on gold parses but then evaluate with noisy Stanza parses (Figure 3b). This suggests that the model trained on noisy parses tends to rely less on the noisy dependency tree inputs, while the model trained on gold parses is more sensitive to the external syntactic input. This correlation is further reinforced by our manual error analysis presented in Appendix C (Figures 4 and 5), where we show how the erroneous edges in the Stanza parses can lead to incorrect predictions of the SRL tags.

Do in-domain parses help? Lastly, for the setting of using in-domain parses, we only evaluate on the SRL task, since on the NER task even using gold parses does not yield substantial gain. We train a biaffine parser (Dozat and Manning, 2017) on the gold parses from the CoNLL-2005 training set and obtain parse trees from it at test time. We observe that while the obtained parse trees are fairly accurate (with a UAS of 92.6% on the test set), it leads to marginal or no improvements over *bert-base*. This finding is also similar to the results obtained by Strubell et al. (2018), where their LISA+ELMo model only obtains a relatively small improvement over SA+ELMo. We hypothesize that as the accuracy of the predicted parses further increases, the F_1 scores would be closer to that from using the gold parses. One possible reason for these marginal gains from using the in-domain parses is that as they are still imperfect, the errors in the parse edges is forcing the model to ignore the syntax information.

Test Set	P	R	F_1
Stanza Dependency Parses (UAS: 82.73)			
Late Fusion	85.74	87.18	86.45
Joint Fusion	85.94	87.05	86.49
In-domain Dependency Parses (UAS: 93.60)			
Late Fusion	86.06	86.90	86.48
Joint Fusion	85.75	86.92	86.33
Gold Dependency Parses			
Late Fusion	88.06	90.32	89.18
Joint Fusion	89.34	90.44	**89.89**

Table 6: SRL results with different parses on the CoNLL-2012 test set.

Test Set	P	R	F_1
Stanza Dependency Parses (UAS: 83.91)			
Late Fusion	88.83	89.42	89.12
Joint Fusion	88.56	89.38	88.97
In-domain Dependency Parses (UAS: 96.10)			
DGLSTM-CRF+ELMo†	–	–	89.64
Gold Dependency Parses			
DGLSTM-CRF+ELMo†	89.59	90.17	**89.88**
Late Fusion	88.75	89.19	88.97
Joint Fusion	88.58	89.31	88.94

Table 7: NER results with different parses on the OntoNotes-5.0 test set averaged over 5 independent runs. † marks results from Jie and Lu (2019).

Overall, we conclude that *parsing quality has a drastic impact on the performance of the Syntax-Augmented BERT models, with substantial gains only observed when gold parses are used.*

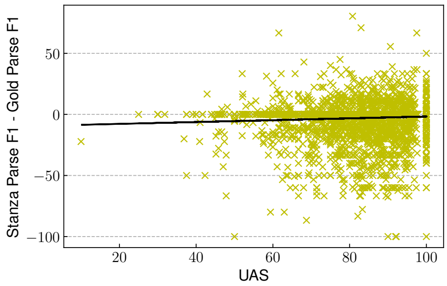
5 Generalizing to BERT Variants

Our previous results used the *bert-base* setting, which is a relatively small configuration among pre-trained models. Devlin et al. (2019) also proposed larger model settings (*bert-large*[7], *whole-word-masking*[8]) that outperformed *bert-base* in all the benchmark tasks. More recently, Liu et al. (2019)

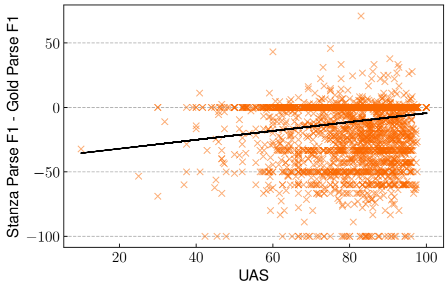
---

[7] 24 layers, 16 attention heads, 1024 model dimensions

[8] `https://bit.ly/3l7rbXx`



(a) When models are trained using Stanza and gold parses, we observe a small positive correlation between F_1 difference and UAS, suggesting that as UAS of Stanza parse increases, the model makes less errors. The slope of the fitted linear regression model is 0.075 and the intercept is -9.27.



(b) Inference is done using Stanza parses on a model trained with gold parses. The slope of the fitted linear regression model is 0.345 and the intercept is -38.9.

Figure 3: Correlation between parse quality and difference in F_1 scores on CoNLL-2005 SRL WSJ dataset.

Gold Parses	**P**	**R**	**F_1**
	BERT		
$\text{BERT}_{\text{LARGE}}$	88.14	88.84	88.49
Late Fusion	89.86	91.57	**90.70**
BERT_{WWM}	88.04	88.87	88.45
Late Fusion	89.88	91.63	**90.75**
	RoBERTa		
$\text{RoBERTa}_{\text{LARGE}}$	89.14	89.90	89.47
Late Fusion	90.89	92.08	**91.48**

Table 8: SRL results from using different pre-trained models on the CoNLL-2005 WSJ test set averaged over 5 independent runs. WWM indicates the whole-wordpiece-masking.

proposed RoBERTa, a better-optimized variant of BERT that demonstrated improved results. A research question that naturally arises is: *Is syntactic information equally useful for these more powerful pre-trained transformers, which were pre-trained in a different way than bert-base?* To answer this, we finetune these models—with and without Late Fusion—on the CoNLL-2005 SRL task using gold parses and report their performance in Table 8.[9]

As expected, we observe that both *bert-large* and *bert-wwm* models outperform *bert-base*, likely due to the larger model capacity from increased width and more layers. Our Late Fusion model consistently improves the results over the underlying BERT models by about 2.2 F_1. The RoBERTa models achieve improved results compared with the BERT models. And again, our Late Fusion model further improves the RoBERTa results by about 2 F_1. Thus, it is evident that *the gains from the Late Fusion model generalize to other widely used pre-trained transformer models*.

6 Generalizing to Out-of-Domain Data

In real-world applications, NLP systems are often used in a domain different from training. And it was previously shown that many NLP systems, such as information extraction systems, suffer from substantial performance degradation when applied to out-of-domain data (Huang and Yates, 2010). While it is evident that syntax trees may help models generalize to out-of-domain data (Wang et al., 2017), since the inductive biases introduced by these trees are invariant across domains, it is unclear if this hypothesis holds for more recent pre-trained models. To study this, we run experiments on SRL with the CoNLL-2005 SRL corpus because this is where we have access to both in-domain and out-of-domain test data using the same annotation schema. The training set of this corpus contains WSJ articles from the newswire domain and the test set consists of two splits: WSJ articles (in-domain) and Brown corpus[10] (out-of-domain). For training, we use both BERT and RoBERTa pre-trained models and leverage gold parses in syntax-GNN models.

From the results in Table 9, the utility of syntax-GNN is evident, as we find that the Late Fusion model always improves over its corresponding BERT and RoBERTa baselines by 2-3% relative F_1, with RoBERTa-large based Late Fusion achieving the best F_1 on both WSJ and Brown datasets. We also compare the performance across both domains, with the last column showing the relative drop in the F_1 score between WSJ and Brown datasets. We observe that the performance of all models drops substantially on the Brown set. However, compared with randomly initialized transformer

---

[9] We use the *Late Fusion* model with *gold parses* in this section, as it is computationally more efficient to train than Joint Fusion model.

[10] contains text from 15 genres (Francis and Kucera, 1979)

	WSJ Test		*Brown Test*		
Gold Parses	$\mathbf{F}_1$	$\%\ \Delta$	$\mathbf{F}_1$	$\%\ \Delta$	$\%\ \nabla$
Baseline Models					
SA+GloVe†	84.5		73.1		13.5
LISA+GloVe†	86.0	1.8	76.5	4.7	11.0
BERT					
BERT$_{\text{BASE}}$	87.5		81.5		6.9
Late Fusion	**90.1**	3.0	**83.9**	2.9	6.9
BERT$_{\text{LARGE}}$	88.5		82.5		6.8
Late Fusion	**90.8**	2.6	**84.6**	2.5	6.8
RoBERTa					
RoBERTa$_{\text{LARGE}}$	89.5		84.0		6.1
Late Fusion	**91.5**	2.2	**85.5**	1.8	6.6

Table 9: Out-of-domain SRL results on the CoNLL-2005 WSJ and Brown test sets. † marks results reported in Strubell et al. (2018). $\%\Delta$ denotes the relative gain in F_1 over pre-trained models when using Late Fusion model, $\%\nabla$ denotes the relative drop in F_1 when a model trained on WSJ dataset is tested on the Brown dataset.

models, where the results can drop by 13%, both syntax-fused and pre-trained models lead to better generalization as the relative error drop reduces to 6–7%. *We see that using Late Fusion does not lead to a better out-of-domain generalization, when compared to strong pre-trained transformers without using parse trees.* Lastly, we find that among all pre-trained models, RoBERTa-large and its syntax-fused variant Late Fusion achieves the lowest out-of-domain generalization error.

7 Related Work

Our work is based on finetuning large pre-trained transformer models for NLP tasks, and is closely related to existing work on understanding the syntactic information encoded in them, which we have earlier covered in Section 1. Here we instead focus on discussing related work that studies incorporating syntax into neural NLP models.

Relation Extraction Neural network models have shown performance improvements when shortest dependency path between entities was incorporated in sentence encoders: Liu et al. (2015) apply a combination of recursive neural networks and CNNs; Miwa and Bansal (2016) apply tree-LSTMs for joint entity and relation extraction; and Zhang et al. (2018) apply graph convolutional networks (GCN) over LSTM features.

Semantic Role Labeling Recently, several approaches have been proposed to incorporate dependency trees within neural SRL models such as learning the embeddings of dependency path between predicate and argument words (Roth and Lapata, 2016); combining GCN-based dependency tree representations with LSTM-based word representations (Marcheggiani and Titov, 2017); and linguistically-informed self-attention in one transformer attention head (Strubell et al., 2018). Kuncoro et al. (2020) directly inject syntax information into BERT pre-training through knowledge distillation, an approach which improves the performance on several NLP tasks including SRL.

Named Entity Recognition Moreover, syntax has also been found to be useful for NER as it simplifies modeling interactions between multiple entity mentions in a sentence (Finkel and Manning, 2009). To model syntax on OntoNotes-5.0 NER task, Jie and Lu (2019) feed the concatenated child token, head token, and relation embeddings to LSTM and then fuse child and head hidden states.

8 Conclusion

In this work, we explore the utility of incorporating syntax information from dependency trees into pre-trained transformers when applied to information extraction tasks of SRL, NER, and RE. To do so, we compute dependency tree embeddings using a syntax-GNN and propose two models to fuse these embeddings into transformers. Our experiments reveal several important findings: syntax representations are most helpful for SRL task when fused within the pre-trained representations, these performance gains on SRL task are contingent on the quality of the dependency parses. We also notice that these models don't provide any performance improvements on NER. Lastly, for the RE task, syntax representations are most helpful when incorporated on top of pre-trained representations.

Acknowledgements

The authors would like to thank Mrinmaya Sachan, Xuezhe Ma, Siva Reddy, and Xavier Carreras for providing us valuable feedback that helped to improve the paper. We would also like to thank the anonymous reviewers for giving us their useful suggestions about this work. This project was supported by academic gift grants from IBM and Microsoft Research, as well as a Canada CIFAR AI Chair held by Prof. Hamilton.

References

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. TACRED revisited: A thorough evaluation of the TACRED relation extraction task. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Peter Battaglia, Jessica Blake Chandler Hamrick, Victor Bapst, Alvaro Sanchez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andy Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Jayne Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Jari Björne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. 2009. Extracting complex biological events with rich graph-based feature sets. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

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

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations (ICLR)*.

Jenny Rose Finkel and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

G. D. Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

W. N. Francis and H. Kucera. 1979. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US.

Katrin Fundel, Robert Küffner, and Ralf Zimmer. 2006. RelEx – Relation extraction using dependency parse trees. *Bioinformatics*.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 40:52–74.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv preprint arXiv:1606.08415*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Fei Huang and Alexander Yates. 2010. Open-domain semantic role labeling by modeling word spans. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Zhanming Jie and Wei Lu. 2019. Dependency-guided LSTM-CRF for named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Zhanming Jie, Aldrian Obaja Muis, and Wei Lu. 2017. Efficient dependency-guided named entity recognition. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *The 2015 International Conference for Learning Representations*.

Adhiguna Kuncoro, Lingpeng Kong, Daniel Fried, Dani Yogatama, Laura Rimell, Chris Dyer, and Phil Blunsom. 2020. Syntactic structure distillation pretraining for bidirectional encoders. *Transactions of the Association for Computational Linguistics*, 8:776–794.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. 2015. A dependency-based neural network for relation classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *COLING 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of The 12th Language Resources and Evaluation Conference*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *Technical Report*.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in neural information processing systems*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

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

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Liangguo Wang, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song, and Lejian Liao. 2017. Can syntax help? improving an LSTM-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1385–1393.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

A Experimental Setup

A.1 Task-Specific Modeling Details

Semantic Role Labeling (SRL): We model SRL as a sequence tagging task using a linear-chain CRF (Lafferty et al., 2001) as the last layer. During inference, we perform decoding using the Viterbi algorithm (Forney, 1973). To highlight predicate position in the sentence, we use indicator embeddings as input to the model.

Named Entity Recognition (NER): Similar to SRL, we model NER as a sequence tagging task, and use a linear-chain CRF layer over the model's hidden states. Sequence decoding is performed using the Viterbi algorithm.

Relation Extraction (RE): As is common in prior work (Zhang et al., 2018; Miwa and Bansal, 2016), the dependency tree is pruned such that the subtree rooted at the lowest common ancestor of entity mentions is given as input to the syntax-GNN. Following Zhang et al. (2018), we extract sentence representations by applying a max-pooling operation over the hidden states. We also concatenate the entity representations with sentence representation before the final classification layer.

A.2 Additional Training Details

During the finetuning step, the new parameters in each model are randomly initialized while the existing parameters are initialized from pre-trained BERT. For regularisation, we apply dropout (Srivastava et al., 2014) with $p = 0.1$ to attention coefficients and hidden states. For all datasets, we use the canonical training, development, and test splits. We use the Adam optimizer (Kingma and Ba, 2015) for finetuning.

We observed that the initial learning rate of 2e-5 with a linear decay worked well for all the tasks. For the model training to converge, we found that 10 epochs were sufficient for CoNLL-2012 SRL and RE and 20 epochs were sufficient for CoNLL-2005 SRL and NER. We evaluate the test set performance using the best-performing checkpoint on the development set.

For evaluation, following convention we report the micro-averaged precision, recall, and F_1 scores in every task. For variance control in all the experiments, we report the mean of the results obtained from five independent runs with different seeds.

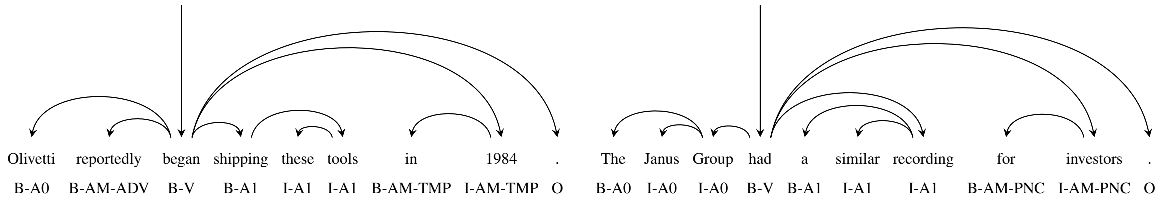
B Additional Baselines

Besides BERT models, we also compare our results to the following previous work, which had obtained good performance gains on incorporating dependency trees with neural models:

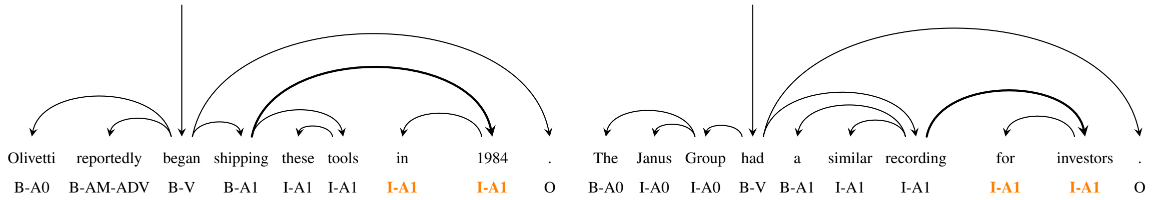
- For SRL, we include results from the SA (self-attention) and LISA (linguistically-informed self-attention) model by Strubell et al. (2018). In LISA, the attention computation in one attention-head of the transformer is biased to enforce dependent words only attend to their head words. The models were trained using both GloVe (Pennington et al., 2014) and ELMo embeddings.
- For NER, we report the results from Jie and Lu (2019), where they concatenate the child token, head token, and relation embeddings as input to an LSTM and then fuse child and head hidden states.
- For RE, we report the results of the GCN model from Zhang et al. (2018) where they apply graph convolutional networks on pruned dependency trees over LSTM states.

C Manual Error Analysis

In this section, we present several examples from our manual error analysis of the predictions from the Late Fusion model when it is trained on CoNLL-2005 SRL WSJ dataset using gold and Stanza parses. Specifically, we show how the incorrect edges present in the parse tree can induce wrong SRL tag predictions. In Figure 4, we observe two examples where the model when trained with gold parses outputs perfect predictions but the when trained with Stanza parses outputs two incorrect SRL tags due to one erroneous edge present in the dependency parse. In Figure 5, we show an example of a longer sentence where due to the presence of four erroneous edges in the Stanza parse, the model makes a series of incorrect predictions of the SRL tags.



(a) Predicted SRL tags using **Gold parses**



(b) Predicted SRL tags using **Stanza parses**

Figure 4: Examples of sentences with their predicted SRL tags when the Late Fusion model is trained using gold parses (4a) and Stanza parses (4b). While the predicted SRL tags using the gold parses are accurate, the erroneous edges in the Stanza parses (highlighted in bold) leads to incorrect SRL tags predictions (highlighted in orange color).

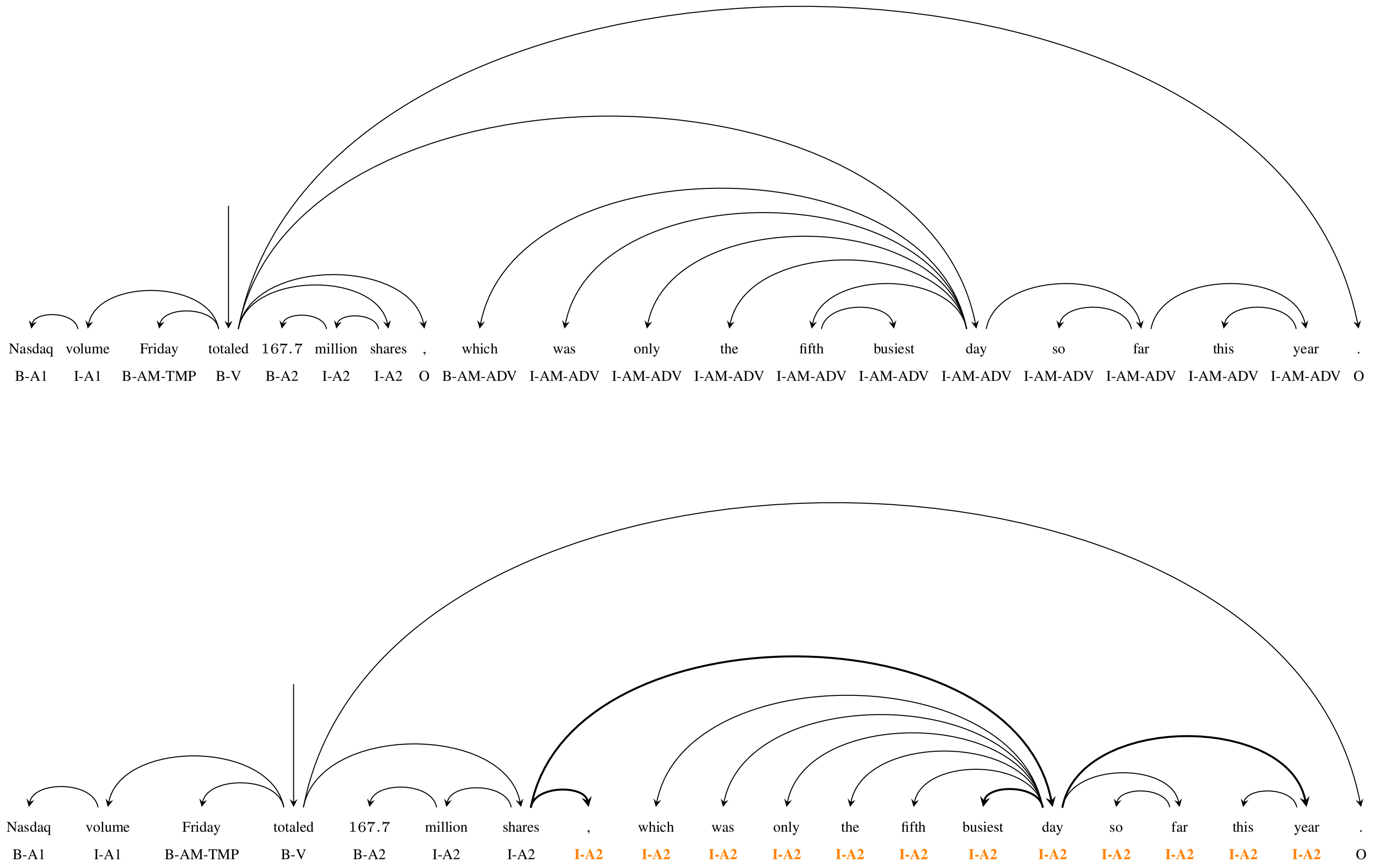


Figure 5: Example of a longer sentence with its predicted SRL tags when the Late Fusion model is trained using gold parses (above figure) and Stanza parses (lower figure). The erroneous edges in the Stanza parses are highlighted in bold. While the predicted SRL tags using the gold parses are accurate, the erroneous edges in the Stanza parses leads to a series of incorrect SRL tag predictions (highlighted in orange color).

Informative and Controllable Opinion Summarization

Reinald Kim Amplayo **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
reinald.kim@ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

Opinion summarization is the task of automatically generating summaries for a set of reviews about a specific target (e.g., a movie or a product). Since the number of reviews for each target can be prohibitively large, neural network-based methods follow a two-stage approach where an *extractive* step first pre-selects a subset of salient opinions and an *abstractive* step creates the summary while conditioning on the extracted subset. However, the extractive model leads to loss of information which may be useful depending on user needs. In this paper we propose a summarization framework that eliminates the need to rely only on pre-selected content and waste possibly useful information, especially when customizing summaries. The framework enables the use of all input reviews by first *condensing* them into multiple dense vectors which serve as input to an abstractive model. We showcase an effective instantiation of our framework which produces more informative summaries and also allows to take user preferences into account using our zero-shot customization technique. Experimental results demonstrate that our model improves the state of the art on the Rotten Tomatoes dataset and generates customized summaries effectively.

1 Introduction

The proliferation of opinions expressed in online reviews, blogs, and social media has created a pressing need for automated systems which enable customers and companies to make informed decisions without having to absorb large amounts of opinionated text. Opinion summarization is the task of automatically generating summaries for a set of opinions about a specific target (Conrad et al., 2009). Figure 1 shows various reviews about the movie "Coach Carter" and example summaries generated by humans and automatic systems.

"Coach Carter" Reviews
• Samuel L. Jackson plays the real-life coach of a high school basketball team in this solid sports drama ... • Great performance by Samuel Jackson but predictable as a slam dunk ... • ... excellent basketball choreography, Coach Carter is fun, hopeful, occasionally silly and, what can I say, inspiring.
Consensus Summary
Even though it's based on a true story, Coach Carter is pretty formulaic stuff, but it's effective and energetic, thanks to a strong central performance from Samuel L. Jackson.
EXTRACT-ABSTRACT Framework
Coach Carter is a preposterously plotted thriller that borrows heavily from other superior films. *(factually incorrect)*
CONDENSE-ABSTRACT Framework
General: An inspirational flick with a healthy dose of message, but it's too predictable. *Customized (acting)*: An inspirational flick with a healthy dose of humor, Coach Carter is a perceptive sports drama with a standout performance from Samuel L. Jackson. *Customized (plot)*: A feel-good tale with a healthy dose of heart, Coach Carter is a worthy addition to the basketball system that it's difficult to resist.

Figure 1: Three out of 150 reviews for the movie "Coach Carter", and summaries written by the editor, and generated by a model following the EXTRACT-ABSTRACT approach and the proposed CONDENSE-ABSTRACT framework. The latter produces more informative and factual summaries whilst allowing to control aspects of the generated summary (such as the acting or plot of the movie).

The vast majority of previous work (Hu and Liu, 2004) views opinion summarization as the final stage of a three-step process involving: (1) aspect extraction (i.e., finding features pertaining to the target of interest, such as battery life or sound quality); (2) sentiment prediction (i.e., determining the sentiment of the extracted aspects); and (3) summary generation (i.e., presenting the identified opinions to the user). Textual summaries are created following mostly extractive methods which select representative segments (usually sentences) from the source text (Popescu and Etzioni,

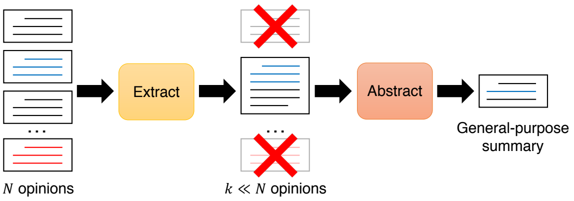
Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2662–2672
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2005; Blair-Goldensohn et al., 2008; Lerman et al., 2009). Despite being less popular, abstractive approaches seem more appropriate for the task at hand as they attempt to generate summaries which are maximally informative and minimally redundant without simply rearranging passages from the original opinions (Ganesan et al., 2010; Carenini et al., 2013; Gerani et al., 2014).

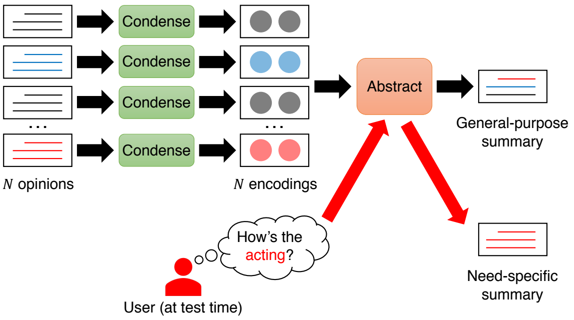
General-purpose summarization approaches have recently shown promising results with end-to-end models which are data-driven and take advantage of the success of sequence-to-sequence neural network architectures. Most approaches (Rush et al., 2015; See et al., 2017) encode documents and then decode the learned representations into an abstractive summary, often by attending to the source input (Bahdanau et al., 2014) and copying words from it (Vinyals et al., 2015). Under this modeling paradigm, it is no longer necessary to identify aspects and their sentiment for the opinion summarization task, as these are learned *indirectly* from training data (i.e., sets of opinions and their corresponding summaries). These models are usually tested on domains where the input is either one document or a small set of documents.

However, the number of input reviews for each target entity tends to be very large (150 for the example in Figure 1). It is therefore practically unfeasible to train a model in an end-to-end fashion, given the memory limitations of modern hardware. As a result, current approaches (Wang and Ling, 2016; Liu et al., 2018; Liu and Lapata, 2019) sacrifice end-to-end elegance in favor of a two-stage framework which we call EXTRACT-ABSTRACT (EA): an *extractive* model first selects a subset of opinions and an *abstractive* model then generates the summary while conditioning on the extracted subset (see Figure 2a). The extractive pass unfortunately has two drawbacks. Firstly, on account of having access to only a small subset of reviews, the summaries can be less informative and inaccurate, as shown in Figure 1. And secondly, user preferences cannot be easily taken into account (e.g., a user may wish to obtain a summary focusing on the acting or plot of a movie as opposed to a general-purpose summary) since more specialized information might have been removed.

In this paper, we propose CONDENSE-ABSTRACT (CA), an alternative two-stage framework which enables the use of *all* input reviews when generating the summary (see Figure 2b). The CONDENSE model first represents the input reviews as encodings, aiming to condense their meaning and distill information relating to sentiment and various aspects of the target being reviewed. The ABSTRACT model then fuses these condensed representations into one aggregate encoding and generates an opinion summary from it. We implement a simple yet effective instantiation of the CA framework, using a vanilla autoencoder as the CONDENSE model, and a decoder with attention and copy mechanisms as the ABSTRACT model. We also introduce a zero-shot customization technique allowing users to control important aspects of the generated summary at test time. Our approach enables controllable generation while leveraging the full spectrum of opinions available for a specific target.



(a) EXTRACT-ABSTRACT (EA) Framework



(b) CONDENSE-ABSTRACT (CA) Framework

Figure 2: Illustration of EA and CA frameworks for opinion summarization. In the CA framework, users can obtain need-specific summaries at test time (e.g., give me a summary focusing on acting).

We perform experiments on a dataset consisting of movie reviews and opinion summaries elicited from the Rotten Tomatoes website (Wang and Ling, 2016; see Figure 1). Our proposed approach outperforms state-of-the-art models by a large margin using automatic metrics and in a judgment elicitation study. We also verify that our zero-shot customization technique can effectively generate need-specific summaries.

2 Related Work

Most opinion summarization models follow extractive methods (see Kim et al., 2011 and Angelidis and Lapata, 2018 for overviews), with the exception of a few systems which are able to generate novel words and phrases not featured in the source text. Ganesan et al. (2010) propose a graph-based framework for generating concise opinion summaries, while Gerani et al. (2014) represent reviews as discourse trees which they aggregate to a global graph to generate a summary. Other work (Carenini et al., 2013; Mukherjee and Joshi, 2013) takes the distribution of opinions and their aspects into account so as to generate more readable summaries. Di Fabbrizio et al. (2014) present a hybrid system which uses extractive techniques to select salient quotes from the input reviews and embeds them into an abstractive summary to provide evidence for positive or negative opinions.

More recent work has seen the effective application of sequence-to-sequence models (Sutskever et al., 2014; Bahdanau et al., 2014) to various abstractive summarization tasks including headline generation (Rush et al., 2015), single- (See et al., 2017; Nallapati et al., 2016), and multi-document summarization (Wang and Ling, 2016; Liu et al., 2018; Liu and Lapata, 2019). Closest to our approach is the work of Wang and Ling (2016) who generate opinion summaries following a two-stage process which first selects/extracts reviews bearing pertinent information, and then generates the summary by conditioning on these reviews. More recent models (Chu and Liu, 2019; Bražinskas et al., 2020; Amplayo and Lapata, 2020) perform opinion summarization in an unsupervised way. However, these are mostly done on toy datasets (Chu and Liu, 2019), typically with a small number of reviews per target entity.

Our proposed framework works better on real-world datasets with a large number of reviews, since it eliminates the need to rely only on pre-selected salient reviews which we argue leads to information loss and subsequently less customizable generation. Instead, our model first *condenses* the source reviews into multiple dense vectors which serve as input to a decoder to generate an abstractive summary. Beyond producing more informative summaries, we demonstrate that our approach also allows to customize them. Recent conditional generation models have focused on controlling various aspects of the output such as politeness (Sennrich et al., 2016), length (Kikuchi et al., 2016), content (Fan et al., 2018), or style (Ficler and Goldberg, 2017). In contrast, our zero-shot customization technique requires neither training examples of documents and corresponding (customized) summaries nor specialized pre-processing to encode which tokens in the input might give rise to customization.

3 CONDENSE-ABSTRACT Framework

We propose an alternative to the EXTRACT-ABSTRACT (EA) approach which enables the use of all input reviews when generating the summary. Figure 2b illustrates our proposed CONDENSE-ABSTRACT (CA) framework. In lieu of an integrated encoder-decoder, we generate summaries using two separate models. The CONDENSE model returns review encodings for N input reviews, while the ABSTRACT model uses these encodings to create an abstractive summary. This two-step approach has two advantages for multi-document summarization. Firstly, CA-based models are more space-efficient, since the set of N reviews is not treated as one large instance but as N separate instances when training the CONDENSE model. And secondly, it is possible to generate maximally informative and customizable summaries targeting specific aspects of the input since the ABSTRACT model operates over the encodings of *all* available reviews.

In the following subsections, we explain how we instantiate a model using the CA framework, which we call CONDASUM, with an LSTM-based[1] vanilla autoencoder (CONDENSE model) and a decoder with attention and copy mechanisms (ABSTRACT model).

3.1 The CONDENSE Model

Let $\mathcal{D}$ denote a cluster of N reviews about a specific target (e.g., a movie or product). For each review $X = \{w_1, w_2, ..., w_M\} \in \mathcal{D}$, the CONDENSE model learns an encoding d, and word-level encodings $h_1, h_2, ..., h_M$. We employ a Bidirectional Long Short Term Memory (BiLSTM) encoder (Hochreiter and Schmidhuber, 1997) as our

---

[1] We use LSTMs as our text encoder instead of other popular alternatives, such as Transformers (Vaswani et al., 2017), since LSTMs work better on autoencoder architectures, as shown in the literature (Liu et al., 2019; Zhang et al., 2020), as well as during our preliminary experiments.

CONDENSE model:

$$\{\overrightarrow{h}_i, \overleftarrow{h}_i\} = \text{BiLSTM}_f(w_i) \tag{1}$$

$$h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i] \quad d = [\overrightarrow{h}_M; \overleftarrow{h}_1] \tag{2}$$

where $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$ are forward and backward hidden states of the BiLSTM at timestep i, and ; denotes concatenation.

Training is performed with a reconstruction objective. We use a separate LSTM as the decoder where the first hidden state z_0 is set to d. Words w'_t are generated using a softmax classifier:

$$z_t = \text{LSTM}_d(w'_{t-1}, z_{t-1}) \tag{3}$$

$$p(w'_t) = \text{softmax}(W z_t + b) \tag{4}$$

The auto-encoder is trained with a maximum likelihood loss:

$$\mathcal{L}_{condense} = -\sum\nolimits_{t=1}^{M} \log p(w_t) \tag{5}$$

Once training has taken place, we use the CONDENSE model to obtain N pairs of review encodings $\{d_i\}$ and word-level encodings $\{h_{i,1}, h_{i,2}, ..., h_{i,M}\}$, $1 \leq i \leq N$ as representations for the reviews in $\mathcal{D}$.

3.2 The ABSTRACT Model

The ABSTRACT model first fuses the multiple encodings obtained from the CONDENSE stage and then generates a summary using a decoder.

Multi-source Fusion We aggregate N pairs of review encodings $\{d_i\}$ and word-level encodings $\{h_{i,1}, h_{i,2}, ..., h_{i,M}\}$, $1 \leq i \leq N$ into a single pair of review encoding d' and word-level encodings $h'_1, h'_2, ..., h'_V$, where V is the number of total unique tokens in the input.

Review encodings are fused using an attentive pooling method which gives more weight to important reviews. Specifically, we learn a set of weight vectors $a_i \in \mathbb{R}^{D_d}$, where D_d is the dimension of d_i, to weight-sum the review encodings:

$$\bar{d} = \sum\nolimits_i d_i / N \tag{6}$$

$$a_i = \text{softmax}(d_i^\top W_p \bar{d}) \tag{7}$$

$$d' = \sum\nolimits_i a_i * d_i \tag{8}$$

where the mean encoding $\bar{d}$ is used as the query vector, and $W_p \in \mathbb{R}^{D_d \times D_d \times D_d}$ is a learned tensor.

We also fuse word-level encodings, since the same words may appear in multiple reviews. To do this, we simply average all encodings of the same word, if multiple tokens of the word exist:

$$h'_j = \sum\nolimits_{(i,k):w_{i,k}=w_j} h_{i,k} / V_{w_j} \tag{9}$$

where V_{w_j} is the number of tokens for word w_j in the input.

Decoder The decoder generates summaries conditioned on the fused review encoding d' and word-level encodings $h'_1, h'_2, ..., h'_V$. We use a simple LSTM decoder enhanced with attention (Bahdanau et al., 2014) and copy mechanisms (Vinyals et al., 2015). We set the first hidden state s_0 to d', and run an LSTM to calculate the current hidden state using the previous hidden state s_{t-1} and word y'_{t-1} at time step t:

$$s_t = \text{LSTM}(y'_{t-1}, s_{t-1}) \tag{10}$$

At each time step t, we use an attention mechanism over word-level encodings to output the attention weight vector a_t and context vector c_t:

$$e_t^i = v^\top \tanh(W_h h'_i + W_s s_t + b_a) \tag{11}$$

$$a_t = \text{softmax}(e_t) \tag{12}$$

$$c_t = \sum\nolimits_i a_t^i * h'_t \tag{13}$$

Finally, we employ a copy mechanism over the input words to output the final word probability $p(y'_t)$ as a weighted sum over the generation probability $p_g(y'_t)$ and the copy probability $p_c(y'_t)$:

$$p_g(y'_t) = \text{softmax}(W_g[s_t; c_t] + b_g) \tag{14}$$

$$\sigma_t = \sigma(v_s^\top s_t + v_c^\top c_t + v_y^\top y'_t) \tag{15}$$

$$p_c(y'_t) = \sum\nolimits_{i:y'_i=y'_t} a_t^i \tag{16}$$

$$p(y'_t) = \sigma_t * p_g(y'_t) + (1 - \sigma_t) * p_c(y'_t) \tag{17}$$

where W, v, and b are learned parameters, and t is the current timestep.

Salience-biased Extracts The model presented so far has no explicit mechanism to encourage salience among reviews. We direct the decoder towards salient reviews by incorporating information from an extractive step. Specifically, we use BERTCENT, a centroid-based (Radev et al., 2000) document extraction method that obtains document representations by resorting to BERT (Devlin et al., 2019).

BERTCENT can be simply described as follows. Firstly, given a review, we obtain its encoding as the average of its token encodings obtained from BERT. We then take the average of the review encodings and treat it as the *centroid* of the input reviews, which approximately represents the information that is considered salient. We select the top k reviews whose encodings are the nearest neigbors to the centroid. The selected reviews are concatenated into a long sequence and encoded using a separate BiLSTM whose output serves as input to an LSTM decoder. This decoder generates a *salience-biased* hidden state r_t. We then update hidden state s_t in Equation (10) as $s_t = [s_t; r_t]$.

Using these extracts, we still take all input reviews into account, while acknowledging that some might be more descriptive than others. This module is a key component to generating *general-purpose* opinion summaries, where a set of aspects is deemed more salient than others (e.g., in general, people care more about the plot rather than the special effects of a movie). However, this extractive module may hurt the customizability of the model (e.g., generating *need-specific* summaries, details explained in Section 3.3), which we show in our experiments in Section 5.

Training We use two objective functions to train the ABSTRACT model. Firstly, we use a maximum likelihood loss to optimize the generation probability distribution $p(y'_t)$ based on gold summaries $Y = \{y_1, y_2, ..., y_L\}$ provided at training time:

$$\mathcal{L}_{generate} = -\sum\nolimits_{t=1}^{L} \log p(y_t) \tag{18}$$

Secondly, we propose a way to introduce supervision and guide the attention pooling weights W_p in Equation (7) when fusing the review encodings. Our motivation is that the resulting fused encoding d' should be roughly equivalent to the encoding of summary y, which can be calculated as $z = \text{CONDENSE}(y)$. Specifically, we use a hinge loss that maximizes the inner product between d' and z and simultaneously minimizes the inner product between d' and n_i, where n_i is the encoding of one of five randomly sampled negative summaries:

$$\mathcal{L}_{fuse} = \sum\nolimits_{i=1}^{5} \max(0, 1 - d'z + d'n_i) \tag{19}$$

The final objective is then the sum of both loss functions:

$$\mathcal{L}_{abstract} = \mathcal{L}_{generate} + \mathcal{L}_{fuse} \tag{20}$$

3.3 Zero-shot Customization

At test time, we can either generate a general-purpose summary or a *need-specific* summary. To generate the former, we run the trained model as is and use beam search to find the sequence of words with the highest cumulative probability. To generate the latter, we employ the following simple technique that revises the query vector $\bar{d}$ in Equation (6).

More concretely, in the movie review domain, users might wish to obtain a summary that focuses on a specific sentiment (positive or negative) or aspect (e.g., acting, plot, etc.) of a movie. In a different domain, users might care about the price of a product, its comfort, and so on. Since these summaries are not available at training time, we undertake such customization without requiring access to need-specific summaries. Instead, at test time, we assume access to background reviews to represent the user need. For example, if we wish to generate a positive summary, our method requires a set of reviews with positive sentiment. This is an easy and practical way to approximately provide the model some background on how sentiment is communicated in a review.

We use these background reviews conveying a user need x (e.g., acting, plot, positive or negative sentiment) in the multi-source fusion module to attend more to input reviews related to x. Let C_x denote the set of background reviews. We obtain a new query vector $\hat{d} = \sum_{c=1}^{|C_x|} d_c / |C_x|$, where d_c is the encoding of the c'th review in C_x, calculated using the CONDENSE model. This simple change allows the model to focus on input reviews with semantics similar to the user's need as conveyed by the background reviews C_x. The new query vector $\hat{d}$ is used instead of $\bar{d}$ to obtain review encoding d' (see Equation (6)).

4 Experimental Setup

Dataset We performed experiments on the Rotten Tomatoes dataset[2] provided in Wang and Ling (2016). It contains 3,731 movies; for each movie we are given a large set of reviews written by professional critics and users and a gold-standard consensus summary written by an editor (see an example in Figure 1). We report the dataset statistics in Table 1. Following previous work (Wang and Ling, 2016), we used a generic label for movie

---

[2] `http://www.ccs.neu.edu/home/luwang/publications.html`

	Train	Dev	Test
#movies	2,458	536	737
#reviews/movie	100.0	98.0	100.3
#tokens/review	23.6	23.5	23.6
#tokens/summary	23.8	23.6	23.8

Table 1: Dataset statistics of Rotten Tomatoes.

titles during training which we replace with the original titles during inference.

Training Configuration For all experiments, our model used word embeddings with 128 dimensions, pretrained using GloVe (Pennington et al., 2014). We set the dimensions of all hidden vectors to 256 and the batch size to 8. For decoding summaries, we use a length-normalized beam search with beam size of 5. We applied dropout (Srivastava et al., 2014) at a rate of 0.5. The model was trained using the Adam optimizer (Kingma and Ba, 2015) with default parameters and l_2 constraint (Hinton et al., 2012) of 2. We performed early stopping based on model performance on the development set. Our model is implemented in PyTorch[3].

Comparison Systems We compare our approach against two types of methods: one-pass methods and methods that use the EA framework. One-pass methods include (a) LEXRANK (Erkan and Radev, 2004), a PageRank-like summarization algorithm which generates a summary by selecting the n most salient units, until the length of the target summary is reached; (b) OPINOSIS (Ganesan et al., 2010), a graph-based abstractive summarizer that generates concise summaries of highly redundant opinions; (c) SUMMARUNNER (Nallapati et al., 2017), a supervised neural extractive model where each review is classified as to whether it should be part of the summary or not; and (d) BERTCENT, a centroid-based method discussed in Section 3.2 that selects $k = 1$ review nearest to the centroid.

EA-based methods include (g) REGRESS+S2S (Wang and Ling, 2016), an instantiation of the EA framework where a ridge regression model with hand-engineered features implements the EXTRACT model, while an attention-based sequence-to-sequence neural network is the ABSTRACT model; (h) BERTCENT+S2S, our implementation of an EA-based system which uses BERTCENT instead of REGRESS as the EXTRACT model; and (i) BERTCENT+PTGEN, the same model as (h) but enhanced with a copy mechanism (Vinyals et al., 2015). For all extractive steps, we set $k = 5$, which is tuned on the development set.

5 Results

Automatic Evaluation We considered two evaluation metrics which are also reported in Wang and Ling (2016): METEOR (Denkowski and Lavie, 2014), a recall-oriented metric that rewards matching stems, synonyms, and paraphrases, and ROUGE-SU4 (Lin, 2004) which is calculated as the recall of unigrams and skip-bigrams up to four words. We also report F_1-scores for ROUGE-1/2/L (Lin, 2004). Unigram and bigram overlap (ROUGE-1 and ROUGE-2) are a proxy for assessing informativenes while the longest common subsequence (ROUGE-L) measures fluency.

Our results are presented in Table 2. Among one-pass systems, the extractive model BERTCENT performs the best; despite being unsupervised and extractive, it benefits from the ability of large neural language models to learn general-purpose representations. When used in EA-based systems, BERTCENT also improves the system performance, where BERTCENT+PTGEN performs the best. Interestingly, BERTCENT performs better than BERTCENT+PTGEN in terms of METEOR and ROUGE-SU4, while the latter performs better in terms of ROUGE-1/2/L. Our CA-based model CONDASUM outperforms all other models across all metrics, showing that exploiting information about all reviews helps in improving performance.

We present in Table 3 various ablation studies, which assess the contribution of different model components. Results confirm that our multi-source fusion method and the fusion loss improve performance. Morevoer, using BERTCENT for the salient-biased extractive step is better than no extractive step or using SUMMARUNNER, which is a weaker extractive model. Both multi-source fusion and salient-biased extracts help create better general-purpose summaries; the former learns which reviews to focus on while the latter explicitly selects the most important ones.

Human Evaluation In addition to automatic evaluation, we also assessed system output by eliciting human judgments. Participants compared summaries produced from the best extractive baseline (BERTCENT), the best EA system (BERTCENT+PTGEN), and our model CONDA-

---

[3] Our code can be downloaded from `https://github.com/rktamplayo/CondaSum`.

Model	METEOR	ROUGE-SU4	ROUGE-1	ROUGE-2	ROUGE-L
LexRank*	5.59	3.98	14.88	1.94	10.50
Opinosis*	6.07	4.90	14.98	3.07	12.19
SummaRunner	7.44	5.50	15.86	2.55	12.15
BertCent	8.89	7.13	17.65	2.78	12.78
Regress+S2S*	6.51	5.70	—	—	—
BertCent+S2S	7.42	6.61	17.59	7.34	15.83
BertCent+PtGen	8.15	6.99	19.71	7.43	17.25
CondaSum	**8.90**	**7.79**	**22.49**	**7.65**	**18.47**

Table 2: Automatic evaluation results on models trained on the original training data. Models whose METEOR and ROUGE-SU4 results are taken from Wang and Ling (2016) are marked with an asterisk *. Best performing results per metric are **boldfaced**.

Model	ROUGE-L
CondaSum	18.47
Mean document fusion	16.69
No fusion loss	15.10
No salience-biased extracts	16.44
SummaRunner extracts	17.80

Table 3: ROUGE-L of CondaSum with less effective document fusion method (second block) and without using our salience-biased extractive step (third block). See Appendix for more detailed comparisons.

Model	Inf	Corr	Gram
BertCent+PtGen	-0.263	-0.358	-0.152*
BertCent	-0.179	-0.112	-0.102*
CondaSum	-0.042	0.021	-0.078
Gold	0.483	0.448	0.331

Table 4: Best-worst scaling scores on informativeness (Inf), correctness (Corr) and grammaticality (Gram). All pairwise systems differences between CondaSum and other system summaries are significant, except the values marked with asterisk (*), based on a one-way ANOVA with posthoc Tukey HSD tests ($p < 0.05$).

Sum, respectively. As an upper bound, we also included Gold standard summaries.

The study was conducted on the Amazon Mechanical Turk platform using Best-Worst Scaling (BWS; Louviere et al., 2015), a less labor-intensive alternative to paired comparisons that has been shown to produce more reliable results than rating scales (Kiritchenko and Mohammad, 2017). Specifically, participants were shown the movie title and basic background information (i.e., synopsis, release year, genre, director, and cast). They were also presented with three system summaries and asked to select the *best* and *worst* among them according to three criteria: *Informativeness* (i.e., does the summary convey opinions about specific aspects of the movie in a concise manner?), *Correctness* (i.e., is the information in the summary factually accurate and corresponding to the information given about the movie?), and *Grammaticality* (i.e., is the summary fluent and grammatical?). Examples of summaries are shown in Figure 1 and more can be found in the Appendix. We randomly selected 50 movies from the test set and compared all possible combinations of summary triples for each movie. We collected three judgments for each comparison. The order of summaries and movies was randomized per participant.

The scores are computed as the percentage of times it was chosen as best minus the percentage of times it was selected as worst. The scores range from -1 (worst) to 1 (best) and are shown in Table 4. Perhaps unsurprisingly, the human-generated gold summaries were considered best, whereas our model CondaSum was ranked second, indicating that humans find its output more informative, correct, and grammatical compared to other systems. BertCent was ranked third followed by BertCent+PtGen. We inspected the summaries produced by the latter system and found they were factually incorrect bearing little correspondence to the movie (examples shown in the Appendix), possibly due to the huge information loss at the extraction stage.

Customizing Summaries We further assessed the ability of CA systems to generate customized summaries at test time. We evaluate CondaSum models with and without the salience-biased extractive step. The latter model biases summary generation towards the k most salient extracted opinions using an additional extractive module which may

Gold

Whether you choose to see it as a statement on consumer culture or simply a special effects-heavy popcorn flick, Gremlins is a minor classic.

CondaSum with extractive step

General: Gremlins is a wholesome, entertaining horror film with an enormous cast of eager stars.

Customized (Positive): Gremlins is a wholesome, entertaining horror film with an enormous cast of eager stars.

Customized (Negative): Gremlins is a wholesome, entertaining horror film with an enormous cast of eager stars.

CondaSum without extractive step

General: Gremlins may appeal to the dark Christmas horror genre.

Customized (Positive): Gremlins is an intelligent, funny Christmas horror film from Joe Dante's novel.

Customized (Negative): Gremlins is an atrociously-acted project whose unoriginal and ineptly-staged horror film from Joe Dante's novel.

Figure 3: Examples of general-purpose and need-specific opinion summaries for the movie "Gremlins", generated by two versions of CondaSum. We also show the consensus summary (Gold). Words/phrases in color highlight aspects pertaining to positive and negative. More examples can be found in the Appendix.

discard information relevant to the user's need. We thus expect this model to be less effective for customization than CondaSum which makes no assumptions regarding which summaries to consider.

In this experiment, we assume users may wish to control the output summaries in four ways focusing on acting- and plot-related aspects of a movie review, as well as its sentiment, which may be positive or negative. Let Cust(x) be the zero-shot customization technique discussed in the Section 3.3, where x is an information need (i.e., acting, plot, positive, or negative). We sampled a set of background reviews C_x ($|C_x|$=1,000) from a corpus of 1 million reviews covering 7,500 movies from the Rotten Tomatoes website, made available in Ficler and Goldberg (2017). The reviews contain sentiment labels provided by their authors and heuristically classified aspect labels. We then ran Cust(x) using both the CondaSum models. We show in Figure 3 customized summaries generated by the models.

To determine which system is better at customization, we again conducted a judgment elicitation study on Amazon Mechanical Turk. Participants read a summary which was created by a general-purpose system or its customized variant. They were then asked to decide if the summary is generic or focuses on a specific aspect (plot or acting) and expresses positive, negative, or neutral sentiment. We selected 50 movies (from the test set) which had mixed reviews and collected judgments from three different participants per summary. The summaries were presented in random order per participant.

	with extracts		without extracts	
Customized	No	Yes	No	Yes
Acting	40.3	40.3	42.0	**78.0**
Plot	73.3	75.0	51.3	**76.7**
Positive	66.0	67.7	65.3	**80.0**
Negative	22.7	22.0	20.7	**40.7**

Table 5: Proportion of summaries which mention a specific aspect/sentiment. **Boldfaced** values show a significant increase ($p < 0.01$; using two-sample bootstrap tests) compared to the non-customized system variant. Aspects are not mutually exclusive (e.g. a summary may talk about both acting and plot), thus the total percentage may exceed 100%.

Table 5 shows what participants thought of summaries produced by non-customized systems (see column No) and systems which had customization switched on (see column Yes). Overall, we observe that CondaSum without the extractive step is able to customize summaries to a great extent. In all cases, crowdworkers perceive a significant increase in the proportion of aspect x when using Cust(x). CondaSum with the extractive step is unable to generate need-specific summaries, showing no discernible difference between generic and customized summaries. This indicates that the use of an extractive module, which is one of the main components of EA-based approaches, limits the flexibility of the abstractive model to customize summaries based on a user need.

6 Conclusions

We introduced the Condense-Abstract (CA) framework for opinion summarization which eliminates the need to rely only on a small subset of extracted reviews and allows the use of all reviews to generate maximally informative summaries. We presented CondaSum, an instantiation of this framework and showed in both automatic and human-based evaluation that it is superior to purely extractive models and abstractive models that include an extractive pre-selection stage. We also showed that when an extractive step is not used, our zero-shot customization technique is able to generate need-specific summaries at test time. In the future, we plan to apply the CA framework to other multi-document summarization tasks.

Acknowledgments

We thank the anonymous reviewers for their feedback. We gratefully acknowledge the support of support of the European Research Council (Lapata, award number 681760) The first author is supported by a Google PhD Fellowship.

References

Reinald Kim Amplayo and Mirella Lapata. 2020. Unsupervised opinion summarization with noising and denoising. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1934–1945, Online. Association for Computational Linguistics.

Stefanos Angelidis and Mirella Lapata. 2018. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3675–3686, Brussels, Belgium. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California.

Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, George Reis, and Jeff Reynar. 2008. Building a sentiment summarizer for local service reviews. In *Proceedings of the WWW Workshop on NLP Challenges in the Information Explosion Era (NLPIX)*, Beijing, China.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics.

Giuseppe Carenini, Jackie Chi Kit Cheung, and Adam Pauls. 2013. Mutli-document summarization of evaluative text. *Computational Intelligence*, 29(4):545–576.

Eric Chu and Peter Liu. 2019. MeanSum: A neural model for unsupervised multi-document abstractive summarization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1223–1232, Long Beach, California, USA. PMLR.

Jack G. Conrad, Jochen L. Leidner, Frank Schilder, and Ravi Kondadadi. 2009. Query-based opinion summarization for legal blog entries. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ICAIL '09, pages 167–176, New York, NY, USA. ACM.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

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

Giuseppe Di Fabbrizio, Amanda Stent, and Robert Gaizauskas. 2014. A hybrid approach to multi-document summarization of opinions in reviews. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 54–63, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark. Association for Computational Linguistics.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T. Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1602–1613, Doha, Qatar. Association for Computational Linguistics.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, Austin, Texas. Association for Computational Linguistics.

Hyun Duk Kim, Kavita Ganesan, Parikshit Sondhi, and ChengXiang Zhai. 2011. Comprehensive review of opinion summarization. Technical report, University of Illinois at Urbana-Champaign.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 4th International Conference on Learning Representations*, San Diego, CA.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Kevin Lerman, Sasha Blair-Goldensohn, and Ryan McDonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 514–522, Athens, Greece. Association for Computational Linguistics.

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

Peter J Liu, Yu-An Chung, and Jie Ren. 2019. Summae: Zero-shot abstractive text summarization using length-agnostic auto-encoders. *arXiv preprint arXiv:1910.00998*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating Wikipedia by summarizing long sequences. In *Proceedings of the 7th International Conference on Learning Representations*, Vancouver, Canada.

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Jordan J. Louviere, Terry N. Flynn, and A. A. J. Marley. 2015. *Best-Worst Scaling: Theory, Methods and Applications*. Cambridge University Press.

Subhabrata Mukherjee and Sachindra Joshi. 2013. Sentiment aggregation using ConceptNet ontology. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 570–578, Nagoya, Japan. Asian Federation of Natural Language Processing.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 3075–3081. AAAI Press.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 339–346, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

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

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, USA. Curran Associates Inc.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 2692–2700, Cambridge, MA, USA. MIT Press.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 47–57, San Diego, California. Association for Computational Linguistics.

Xinyuan Zhang, Ruiyi Zhang, Manzil Zaheer, and Amr Ahmed. 2020. Unsupervised abstractive dialogue summarization for tete-a-tetes. *arXiv preprint arXiv:2009.06851*.

Coloring the Black Box:
What Synesthesia Tells Us about Character Embeddings

Katharina Kann*
University of Colorado Boulder
katharina.kann@colorado.edu

Mauro M. Monsalve-Mercado*
Columbia University
mauro.m.monsalve@columbia.edu

Abstract

In contrast to their word- or sentence-level counterparts, character embeddings are still poorly understood. We aim at closing this gap with an in-depth study of English character embeddings. For this, we use resources from research on grapheme–color synesthesia – a neuropsychological phenomenon where letters are associated with colors –, which give us insight into which characters are similar for synesthetes and how characters are organized in color space. Comparing 10 different character embeddings, we ask: How similar are character embeddings to a synesthete's perception of characters? And how similar are character embeddings extracted from different models? We find that LSTMs agree with humans more than transformers. Comparing across tasks, grapheme-to-phoneme conversion results in the most human-like character embeddings. Finally, ELMo embeddings differ from both humans and other models.

1 Introduction

Neural network models have become crucial tools in natural language processing (NLP) and define the state of the art on a large variety of tasks (Wang et al., 2018). However, they are difficult to understand and are often considered "black boxes".[1] This can make their use difficult to defend in many settings, for instance in a legal context, and constitutes a barrier for model improvement. Therefore, a lot of research has been dedicated to investigating the information encoded by neural networks. Especially word embeddings, contextualized word representations, and language representation models like BERT (Devlin et al., 2019) have been exhaustively studied (Rogers et al., 2020).

Figure 1: Characters as they might be seen by a grapheme–color synesthete. The colors in this example are randomly chosen.

Character embeddings are used for a large set of tasks, either as a supplement to word-level input, e.g., for part-of-speech tagging by Plank et al. (2016), or on their own, e.g., for character-level sequence-to-sequence (seq2seq) tasks by Kann and Schütze (2016). Despite this, they have not yet been explicitly analysed. One reason for this might be that identifying relevant properties to study is more challenging than for their word-level counterparts. However, we argue that, in order to truly shine light into black-box NLP models, it is necessary to understand each and every component of them.

In this paper, we perform a detailed study of English character embeddings. Our **first contribution** is a **character similarity task** (§3) in analogy to the word-based version: *Do embeddings agree with humans on which pairs of characters are similar?* While e.g., *cat* and *tiger* are generally considered more similar than *cat* and *chair*, this is not trivial for characters. For our annotations, we exploit a phenomenon called *synesthesia*. People with synesthesia, *synesthetes*, share perceptual experiences between two or more senses (§2). For grapheme–color synesthetes, each letter is associated with a color, which is consistent over a person's life time (Eagleman et al., 2007), cf. Figure 1. Using a dataset of letters from the English alphabet and the associated colors from 4269 synesthetes (Witthoft et al., 2015), we compute differences in color space (§3.3) as a proxy for character similarity.

As our **second contribution**, we propose a set

---

*Equal contribution.

[1] This fact has led to the establishment of a workshop with the same name: https://blackboxnlp.github.io

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2673–2685
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

of methods to **characterize the structure of character embedding matrices** (§4). The methods we propose are (i) a clustering analysis, (ii) computing the clustering coefficient, (iii) measuring betweenness centrality, and (iv) computing cut-distances.

Our **final contribution** is a detailed **character embedding analysis** (§6). We explore 6 types of embeddings, obtained from 4 different architectures trained on 3 different tasks, as well as 4 embedding matrices from pretrained ELMo models (Peters et al., 2018). Comparing across models trained on the same task, character embeddings from LSTMs, which, similar to humans, process input sequentially, correlate more with human similarity scores than embeddings from transformers. Comparing across tasks, character embeddings from language modeling show a surprisingly low correlation with human judgments. In contrast, the correlation is highest for grapheme-to-phoneme conversion (G2P). This is in line with findings that colors perceived by synesthetes are influenced by the sound of each letter (Asano and Yokosawa, 2013).

2 Synesthesia

Synesthesia is the perceptual phenomenon that two or more sensory or cognitive pathways are co-activated in the brain by stimulating only one of them. For example, for a person with chromesthesia, a common form of synesthesia, hearing a particular sound evokes the visual perception of colors (Cytowic, 2018; Simner, 2012).

By far the most common form of synesthesia is grapheme-color synesthesia, for which subjects report the perception of colors when seeing numerals or letters (cf. Figure 1). The neural basis behind the phenomenon is still a much debated topic, but evidence suggests it might be the result of increased effective connectivity between the brain areas involved (Ramachandran and Hubbard, 2001a). For example, visual area V4a in the ventral pathway, often associated with contextual processing and perception of color, has shown to be part of a complex in the fusiform gyrus exhibiting higher cortical thickness, volume, and surface area in synesthetes (Hubbard et al., 2005). Belonging to this complex and adjacent to the color processing region is the area dedicated to the visual recognition and processing of graphemes, suggesting that a prominence of this type of synesthesia is due to higher region interconnectivity being more probable.

Additional evidence supports the idea that synesthetic associations often involve the extraction of meaning from a stimulus, suggesting that abstract constructions play a major role in the associations formed (Mroczko-Wasowicz and Nikolic, 2014). For cases where language is involved, semantics may in part underlie color representations of words and graphemes in V4.

The study of synesthesia presents a unique opportunity to understand the neural basis of cognitive models of language (Ramachandran and Hubbard, 2001b; Simner, 2007). Here, grapheme–color synesthesia serves as a window to look at how the brain represents individual characters.

3 A Character Similarity Task

3.1 Motivation: Word Similarity

Our first contribution is a character-level analogue of the word similarity task, an intrinsic evaluation method for word embeddings. It consists of judging the similarity of pairs of words, e.g., of *cat* and *tiger*. To obtain a gold standard for this task, human annotators assign similarity scores to a list of word pairs. This is not always trivial: are *cat* and *bird* more or less similar than *cat* and *fish*? However, people tend to agree on general tendencies.

Word embeddings are evaluated on similarity datasets as follows: The similarity – usually cosine similarity – of all pairs of words is computed. The agreement between models and human annotations is then quantified as the correlation between the two vectors of scores. Word embeddings with a higher performance on word similarity tasks are expected to perform better on downstream tasks, since they encode valuable information about words and the relationships between them.

3.2 From Word to Character Similarity

In order to design a *character* similarity task, we require a gold standard. However, we are not likely to get a meaningful answer when asking people if *B* and *J* are more similar than *C* and *Q*.

We solve this problem by looking at how different characters are represented by grapheme–color synesthetes in color space. This tells us how similar characters are perceived to be without having to ask explicitly. We leverage a dataset collected by Witthoft et al. (2015), which consists of letter-to-color mappings collected from 4269 synesthetes and compute pair-wise perceptually uniform distances between characters. Analogously to the word-level version of the task, we then take as our gold standard the average over all annotators. For embeddings, we compute cosine similarities between character vectors.

We differ from the word similarity task in an important detail: we do not evaluate the *quality* of embeddings. Instead, we aim to understand character embeddings by assessing how *similar to the human perception of characters* they are. In particular, we do not necessarily expect embeddings which score higher on our task to perform better on downstream applications.

3.3 Human Perception of Color Differences

To motivate the distance metric we use, we first summarize human perception of color. The visual process of color perception starts in the retina, where three types of light-sensitive receptors are tuned to respond broadly around three distinct frequencies in the visible spectrum. However, color perception is not simply reduced to combinations of these physiological responses. The human brain can perceive the same physical light frequency as a different color depending on a plethora of contextual markers, mostly due to further processing upstream of the retina into high-level visual cortical areas, where associations are formed between environmental cues and the object being visualized.

Important contextual properties of color affecting perception, often discussed in color theory as the Munsell (or HSL) color system, are its hue, saturation, and lightness. Combinations of these properties are interpreted in a *highly non-linear* manner by cortical areas tasked with color perception. Not surprisingly, simple metrics for color comparison, such as the Euclidean distance in RGB space, perform poorly in situations involving a color discrimination task. Constructing perceptually uniform metrics that deal with these perceived non-linearities is an active field of research. One standard metric for perceptual color comparison is the CIEDE2000 color difference (Sharma et al., 2005), which includes several correction factors for a modified Euclidean metric on HSL space. We employ CIEDE2000 in our analysis.

4 Analysis of the Distance Matrix

CIEDE2000 allows us to obtain pair-wise distances between all letters in the alphabet by computing color differences for the associated colors as perceived by synesthetes. The character similarity task then compares *vectors* of pair-wise distances. However, we can gain additional insight from the distance *matrices*, which represent fully connected networks whose nodes are the letters in the alphabet and whose edges are weighted by the pair-wise differences. Thus, we further propose four well-established methods from network theory (Newman, 2018) for the analysis of character embeddings and human difference matrices.

Clustering analysis. To characterize the global structure of the network, we first propose Ward's variance minimization algorithm to identify clusters, i.e., groups of letters that are similar to each other, but far away from other letters. Ward's algorithm is part of a family of hierarchical clustering algorithms whose objective function aims at minimizing the variance within clusters (Ward, 1963). Starting from a forest of clusters (initially single nodes), the algorithm evaluates the Ward distance d between a new cluster u made up of clusters s and t, and a third cluster v not used yet, as

$$d(u, v) = \sqrt{a + b - c}, \tag{1}$$

with

$$a = \frac{|v| + |s|}{T} d(v, s)^2, \tag{2}$$

$$b = \frac{|v| + |t|}{T} d(v, t)^2, \tag{3}$$

$$c = \frac{|v|}{T} d(s, t)^2, \tag{4}$$

where $T := |s| + |t| + |v|$, and $|\cdot|$ is the size of the cluster. If u is a good cluster, then s and t are removed from the forest and the algorithm continues until only one cluster is left. Finally, the *number of clusters*, *their sizes*, and the *Ward distances between clusters* characterize the global structure of the network.

Clustering coefficient. The clustering coefficient provides a way to measure the degree to which nodes in a network cluster together. For a binary network, the local version represents the fraction of the number of pairs of neighbors of a node that are connected, over the total number of pairs of neighbors of said node, and it measures the influence of a node on its immediate neighbors. Several generalizations for weighted networks have been proposed (Saramäki et al., 2007). Here, we use the average of the weights for neighbors in the subgraph of a node u

$$c_u = \frac{1}{deg(u)(deg(u) - 1))} \sum_{vw} (\hat{w}_{uv} \hat{w}_{uw} \hat{w}_{vw})^{1/3},$$

where $\hat{w}_i$ denotes weight w_i normalized by the maximum weight in the network, and *deg* is the degree of the node (the sum of its edges' weights). The average over all nodes is used as a proxy for the overall level of clustering within the network.

Betweenness centrality. Different concepts of centrality attempt to capture the relative importance of particular nodes in the network. One such concept, *betweenness*, measures the extent to which a node lies on the shortest path between pairs of nodes (Brandes, 2008). In a sense, it generalizes the clustering coefficient from a measure of the local influence of a node to immediate neighbors to the whole network. In particular, it accounts for nodes that connect two different clusters while not being a part of either. Betweenness centrality is computed as the fraction of all-pairs shortest paths that pass through a particular node u

$$B(u) = \sum_{s,t} \frac{n(s,t|u)}{n(s,t)},$$

where the sum is over all nodes s and t in the network, and n counts the number of shortest paths between two nodes, optionally taking into account if it passes through u.

Cut-distance. The fourth and last approach we propose to characterize our matrices of character distances is to employ a matrix norm called cut-norm. It is widely used in graph and network theory and has been shown to capture global features such as clustering and sparseness (Frieze and Kannan, 1999). The cut-norm of a matrix $A = (a_{ij})_{i \in M, j \in N}$ is defined as

$$||A||_c := \max \left\{ \frac{\left|\sum_{i \in I} \sum_{j \in J} a_{ij}\right|}{|I||J|} : \begin{matrix} I \subset M \\ J \subset N \end{matrix} \right\},$$

i.e., the maximum over all possibles sub-matrix arrangements is taken as the norm. In practice, we compute it using an efficient implementation[2] that relies on Grothendieck's inequality for an approximation (Alon and Naor, 2004; Wen et al., 2013). In addition, the norm naturally gives rise to a distance metric $d_c(A, B) := ||A - B||_c$ that allows us to compare pairs of distance matrices directly.

5 Models and Tasks

We now describe the tasks and model architectures we employ to train different character embeddings.

5.1 Tasks

Language modeling. The task of language modeling consists of computing a probability distribution over all elements in a predefined vocabulary, given a sequence of past elements. Language models can either be used to assign a probability to an input sequence or to generate text by sampling from the probability distributions. We train language models on the character level, i.e., the vocabulary consists of the English alphabet.

All our language models are trained on wikitext-103.[3] We use the provided training, development, and test splits. The training set consists of roughly 1 million tokens.

Morphological inflection. In languages that exhibit rich inflectional morphology, words inflect: grammatical information like number, case, and tense are incorporated into the word itself. For instance, *wrote* is the inflected form of the English lemma *write*, expressing past tense.

The task of morphological inflection consists of mapping a lemma to an inflected form which is defined by a set of morphological tags. Morphological inflection is typically being cast as a character-level seq2seq task, where the characters of the lemma together with the morphological tags are the input, and the characters of the inflection are the output (Kann and Schütze, 2016):

`PST w a l k` → `w a l k e d`

We train our inflection models on the $10,000$ English training examples provided by Cotterell et al. (2017) and use the corresponding development and test sets with $1,000$ examples each.

Grapheme-to-phoneme conversion. Given a word's spelling, G2P consists of generating an (IPA-like) representation of its pronunciation:

`p r e t t i e r` → `P R IH T IY ER`

It has been shown that similar-sounding letters tend to be associated with similar synesthetic colors (Asano and Yokosawa, 2013). Hence, we assume that the embedding space induced by this task could be similar to human perception of characters.

We train all G2P models on examples extracted from the CMU Pronouncing Dictionary.[4] Our training, development, and test sets consist of 114,399, 5447, and 12,855 examples, respectively.[5]

---

[2] https://pypi.org/project/cutnorm/

[3] https://s3.amazonaws.com/research.metamind.io/wikitext/wikitext-103-v1.zip

[4] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[5] We use the splits provided at https://github.

5.2 Architectures

To isolate the effects of task and model architecture, we train different architectures for each task. All test set performances are shown in Table 1. We train 50 models with different random seeds for seq2seq tasks, and 10 instances for language models. For our analysis, we look at the input embeddings and average pair-wise distances over models for each group. All models have been trained on an NVidia Tesla K80 GPU.

LSTM seq2seq architecture. Our first architecture is a seq2seq model similar to that by Bahdanau et al. (2015). It consists of a bi-directional long short-term memory (LSTM; Hochreiter and Schmidhuber, 1997) encoder and an LSTM decoder, which are connected via an attention mechanism. We apply it on the character level.

We train this architecture on morphological inflection (**Infl$_{\text{LSTM}}$**) and G2P (**G2P$_{\text{LSTM}}$**), using the fairseq sequence modeling toolkit for our implementation.[6] All embeddings and hidden states are 100-dimensional, and both encoder and decoder have 1 hidden layer. For training, we use an Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.001, dropout with a coefficient of 0.3, and a batch size of 20. To account for different training set sizes, we train our model for G2P for 15 and our model for inflection for 100 epochs.

Transformer seq2seq architecture. We further experiment with a transformer seq2seq architecture (Vaswani et al., 2017). Similar to the LSTM seq2seq model, this architecture consists of an encoder and a decoder which are connected by an attention mechanism. However, the encoder and the decoder consist of combinations of feed-forward and attention layers instead of LSTMs.

We apply this architecture to morphological inflection (**Infl$_{\text{T}}$**) and G2P (**G2P$_{\text{T}}$**), and implement the models using the fairseq toolkit. All embeddings have 256 dimensions, and hidden layers are 1024-dimensional. Both encoder and decoder have 4 layers, and use 4 attention heads. We employ an Adam optimizer with an initial learning rate of 0.001 for training, together with dropout with a coefficient of 0.3, and a batch size of 400. We train our models for G2P for 30 epochs and our models for morphological inflection for 100 epochs.

Infl$_{\text{LSTM}}$	**Infl$_{\text{T}}$**	**G2P$_{\text{LSTM}}$**	**G2P$_{\text{T}}$**	**LM$_{\text{LSTM}}$**	**LM$_{\text{T}}$**
0.95	0.94	0.64	0.62	3.37	2.62
50	50	50	50	10	10

Table 1: **Top**: accuracy for inflection and G2P and character-level perplexity for language modeling. **Bottom**: number of model instances. Results are averaged over all runs; models are described in the text.

LSTM language model architecture. We also experiment with an LSTM language model (**LM$_{\text{LSTM}}$**). This architecture consists of a unidirectional LSTM, and it receives the last generated character as input at each time step.

Our implementation is based on the official pytorch LSTM language model example.[7] We use the default hyperparameters except for the following: our embeddings and LSTM hidden states are 512-dimensional, we use 2 hidden layers, and we train for 2 epochs with a batch size of 64.

Transformer language model architecture. Our last architecture is a transformer language model (**LM$_{\text{T}}$**). Like the LSTM language model, it receives previously generated characters as input and computes a probability distribution over the character vocabulary.

Again, we use the fairseq toolkit for our implementation, and employ the default hyperparameters for the transformer language model. We use an Adam optimizer with an initial learning rate of 0.0005 for training, and dropout with a coefficient of 0.1. This model is trained for 3 epochs.

5.3 Pretrained Models

We further analyze the character embeddings of ELMo models (Peters et al., 2018). ELMo models are pretrained networks, aimed at producing contextualized word embeddings for use in downstream NLP tasks. The model architecture consists of a convolutional layer over character embeddings, whose output is then fed into a 2-layer bidirectional LSTM. ELMo models are trained with a bidirectional language modeling objective.

We experiment with 4 English models which are available online:[8] small (**ELMo$_{\text{s}}$**), medium (**ELMo$_{\text{m}}$**), original (**ELMo$_{\text{o}}$**) and original-5.5B (**ELMo$_{\text{l}}$**). Those models differ in the number of their parameters and the amount of text they have been trained on: ELMo$_{\text{s}}$ and ELMo$_{\text{m}}$ have 13.6 million and 28 million parameters, respectively.

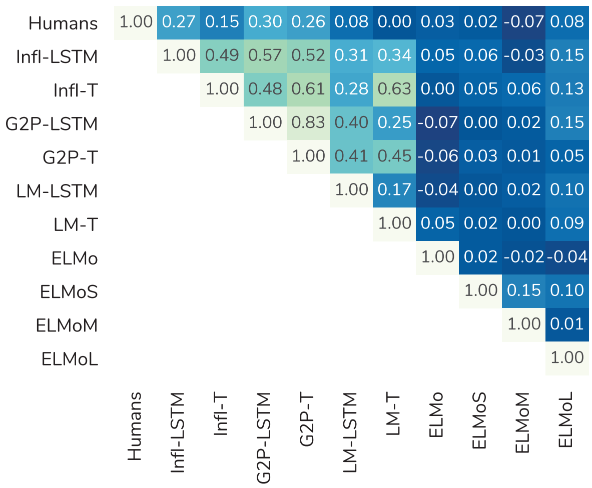
---

com/microsoft/CNTK/tree/master/Examples/SequenceToSequence/CMUDict/Data.

[6] https://github.com/pytorch/fairseq

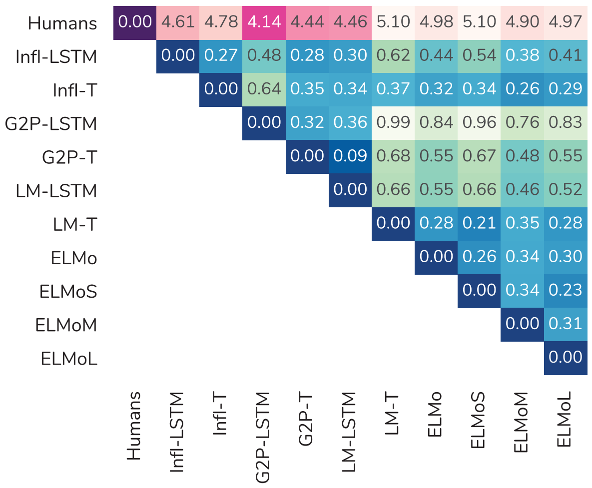
[7] https://github.com/pytorch/examples/tree/master/word_language_model

[8] https://allennlp.org/elmo



	Humans	Infl-LSTM	Infl-T	G2P-LSTM	G2P-T	LM-LSTM	LM-T	ELMo	ELMoS	ELMoM	ELMoL
Humans	1.00	0.27	0.15	0.30	0.26	0.08	0.00	0.03	0.02	-0.07	0.08
Infl-LSTM		1.00	0.49	0.57	0.52	0.31	0.34	0.05	0.06	-0.03	0.15
Infl-T			1.00	0.48	0.61	0.28	0.63	0.00	0.05	0.06	0.13
G2P-LSTM				1.00	0.83	0.40	0.25	-0.07	0.00	0.02	0.15
G2P-T					1.00	0.41	0.45	-0.06	0.03	0.01	0.05
LM-LSTM						1.00	0.17	-0.04	0.00	0.02	0.10
LM-T							1.00	0.05	0.02	0.00	0.09
ELMo								1.00	0.02	-0.02	-0.04
ELMoS									1.00	0.15	0.10
ELMoM										1.00	0.01
ELMoL											1.00

Figure 2: Pearson correlation between the vector of averaged human character differences and distance vectors according to character embeddings.



	Humans	Infl-LSTM	Infl-T	G2P-LSTM	G2P-T	LM-LSTM	LM-T	ELMo	ELMoS	ELMoM	ELMoL
Humans	0.00	4.61	4.78	4.14	4.44	4.46	5.10	4.98	5.10	4.90	4.97
Infl-LSTM		0.00	0.27	0.48	0.28	0.30	0.62	0.44	0.54	0.38	0.41
Infl-T			0.00	0.64	0.35	0.34	0.37	0.32	0.34	0.26	0.29
G2P-LSTM				0.00	0.32	0.36	0.99	0.84	0.96	0.76	0.83
G2P-T					0.00	0.09	0.68	0.55	0.67	0.48	0.55
LM-LSTM						0.00	0.66	0.55	0.66	0.46	0.52
LM-T							0.00	0.28	0.21	0.35	0.28
ELMo								0.00	0.26	0.34	0.30
ELMoS									0.00	0.34	0.23
ELMoM										0.00	0.31
ELMoL											0.00

Figure 3: Cut-distance between the average human distance matrix and all character embeddings.

Both ELMo$_\mathrm{o}$ and ELMo$_\mathrm{l}$ have 93.6 million parameters. All models except for ELMo$_\mathrm{l}$ have been trained on the 1 Billion Word Benchmark.[9] ELMo$_\mathrm{l}$ has been trained on a combination of Wikipedia and news crawl data, which together result in a dataset of 5.5 billion tokens.

6 Character Embedding Analysis

6.1 Results on the Character Similarity Task

The Pearson correlation of all models with human judgements as well as with each other is shown in Figure 2. G2P$_\mathrm{LSTM}$ shows the highest correlation with 0.30, while LM$_\mathrm{T}$ is not correlated at all, and ELMo$_\mathrm{m}$ obtains the strongest negative correlation with -0.07.

More generally, we see that most models are correlated with each other (between 0.17 and 0.83), with the exception of the ELMos: the predictions of ELMo$_\mathrm{l}$ are the only ones which have a Pearson correlation ≥ 0.1 with those of some other models.

Comparing to human scores, we find the following patterns: embeddings from seq2seq tasks show a higher correlation than embeddings from language models. Even ELMo models, which are trained on large amounts of text, obtain a maximum correlation of 0.08. Thus, we conclude that language modeling does not result in embeddings which perform well on our character similarity task. Embeddings for G2P correlate more strongly with human character perception than embeddings from inflection, when comparing identical architectures. This is noteworthy, since colors perceived by synesthetes are supposed to be influenced by the sound of each letter (Asano and Yokosawa, 2013), similar to the embeddings for G2P. Finally, comparing across architectures, we see that LSTM models correlate stronger with human judgments than transformer models, which is in line with the common understanding that recurrent neural networks might be better models of human cognition.

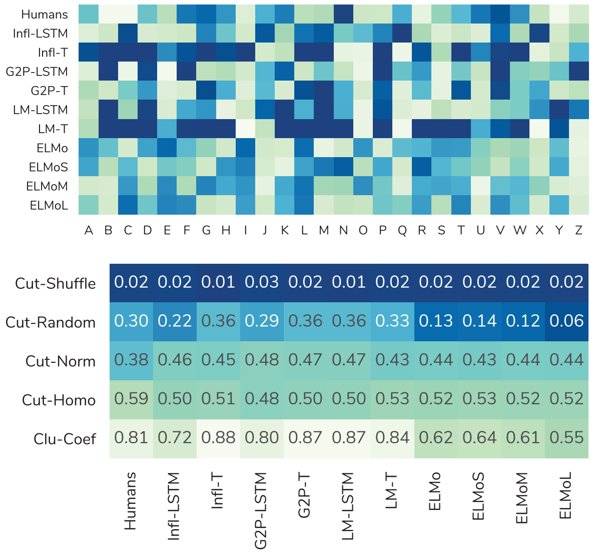
6.2 Comparing Humans and Machines

Clustering analysis. First, we look at the global structure of all character embeddings (cf. Figure 5). All but the ELMo models exhibit a marked separation between a tight cluster of vowels (`AEIOU`) or extended vowels (+`Y`), which are highly similar, and the rest of the alphabet. In contrast, this distinction is not found for humans, neither for individuals nor the average distance matrix. Despite this, the human average does present a clear global structure (cf. appendix for details). One clear cluster is `BDGJKMNPQRVWXZ`, with the particularly close pairs `MN` and `XZ`, perhaps due to the letters' shape, sound, or proximity in the alphabet. This cluster is far away from the letters in `AES`, which themselves do not form a cluster. Another cluster, on average far form the first, is formed by the letters `CILOUY`.

Apart from the clear separation between vowels and consonants, character embeddings exhibit rich additional structure (cf. Figure 5). For Infl$_\mathrm{LSTM}$, the cluster `BCFHJMPQW` contains the tighter subcluster `BJQW` that is similar to the letters `GKVX`. In contrast, Infl$_\mathrm{T}$ has the two small clusters `HJKQ` and `LNR` far from each other, and a less clear-cut

---

[9] http://www.statmt.org/lm-benchmark/



	Humans	Infl-LSTM	Infl-T	G2P-LSTM	G2P-T	LM-LSTM	LM-T	ELMo	ELMoS	ELMoM	ELMoL
Cut-Shuffle	0.02	0.02	0.01	0.03	0.02	0.01	0.02	0.02	0.02	0.02	0.02
Cut-Random	0.30	0.22	0.36	0.29	0.36	0.36	0.33	0.13	0.14	0.12	0.06
Cut-Norm	0.38	0.46	0.45	0.48	0.47	0.47	0.43	0.44	0.43	0.44	0.44
Cut-Homo	0.59	0.50	0.51	0.48	0.50	0.50	0.53	0.52	0.53	0.52	0.52
Clu-Coef	0.81	0.72	0.88	0.80	0.87	0.87	0.84	0.62	0.64	0.61	0.55

Figure 4: Comparison measures for models of character embedding. **Top**: Betweenness centrality. **Bottom**: Cut-distances and cluster coefficients for the average human distance matrix and all character embeddings.

structure among the rest of the consonants.

Embeddings from G2P_{LSTM} exhibit a structure clearly connected to the G2P task. The distance network has many small clusters of similar sounding letters, e.g., `UW`, `IJ`, `JY`, `GJ`, `CKQ`, `SXZ`, and `BFPV`. G2P_{LSTM}'s counterpart, G2P_{T}, produces similar groupings, but displays a more homogeneous structure overall.

LM_{LSTM} has the tight cluster `KSXYZ` within the larger `BCDFGKMPSTVWXYZ` cluster, and weaker connections between other letters. LM_{T} has a clearer cluster structure, showing the strongest separation between vowels and consonants, especially from the non-cluster `JKQXZ`, which is also well-separated from other consonants. ELMo models are outliers in that they do not present a clear global structure: only loose clusters can be identified.

Cluster coefficient. Looking at the cluster coefficients (cf. Figure 4, last row) we also see a difference between ELMo embeddings and the other models: all models except for the ELMos have cluster coefficients between 0.72 and 0.88 and, thus, close to the human average of 0.81. In contrast, the cluster coefficients for ELMo embeddings are between 0.64 for ELMo_{s} and 0.55 for ELMo_{l}.

Betweenness centrality. The local values of betweenness centrality (cf. Figure 4) show the rich structure of the similarity matrices for most embeddings. For all but the ELMo models, the majority of letters have either extremely high or low levels of betweenness. In particular vowels tend to occupy prominent places in the network structure. Humans, in contrast, are more similar to ELMo embeddings.

Cut-distance. Looking at cut-distances (cf. Figure 4), we find that the structure of ELMo embeddings is significantly more similar to a random matrix than that of the other embeddings. The cut-distances (cf. Figure 3) between humans and embeddings largely agree with the conclusions from Section 6.1 – G2P_{LSTM} and the ELMo models are respectively the most similar and dissimilar –, even though correlation for node-to-node similarities does not necessarily imply a similar global structure.

7 Related Work

Neural network analysis. A lot of ink has been spilled on what neural network models learn and how. For instance, Zhang and Bowman (2018) investigated different pretraining objectives on their ability to induce syntactic and part-of-speech information. Pruksachatkun et al. (2020) studied model performance on probing tasks to investigate what models learn from intermediate-task training. Belinkov et al. (2017) explored what neural machine translation models learn about morphology.

Other work created test sets to evaluate specific linguistic model abilities. Linzen et al. (2016) made a dataset to investigate the ability of neural networks to detect mismatches in subject–verb agreement in the presence of distractor nouns. Warstadt et al. (2019) created a benchmark called BLiMP to assess the ability of language models to handle specific syntactic phenomena in English. Mueller et al. (2020) introduced a similar suite of test sets in English, French, German, Hebrew and Russian, also focusing on syntactic phenomena. Similarly, Xiang et al. (2021) presented CLiMP, a benchmark for the evaluation of Chinese language models.

Besides that, attention mechanisms (Bahdanau et al., 2015) in neural models have been common subjects of investigation. Jain and Wallace (2019) claimed that ”attention is not explanation”, to be later on challenged by Wiegreffe and Pinter (2019), who argued that ”attention is not not explanation”. However, the relationship between inputs, attention weights, and outputs is still poorly understood.

Furthermore, our work is related to research on which information is learned and how information is encoded by so-called language representation models, e.g., BERT (Devlin et al., 2019) or

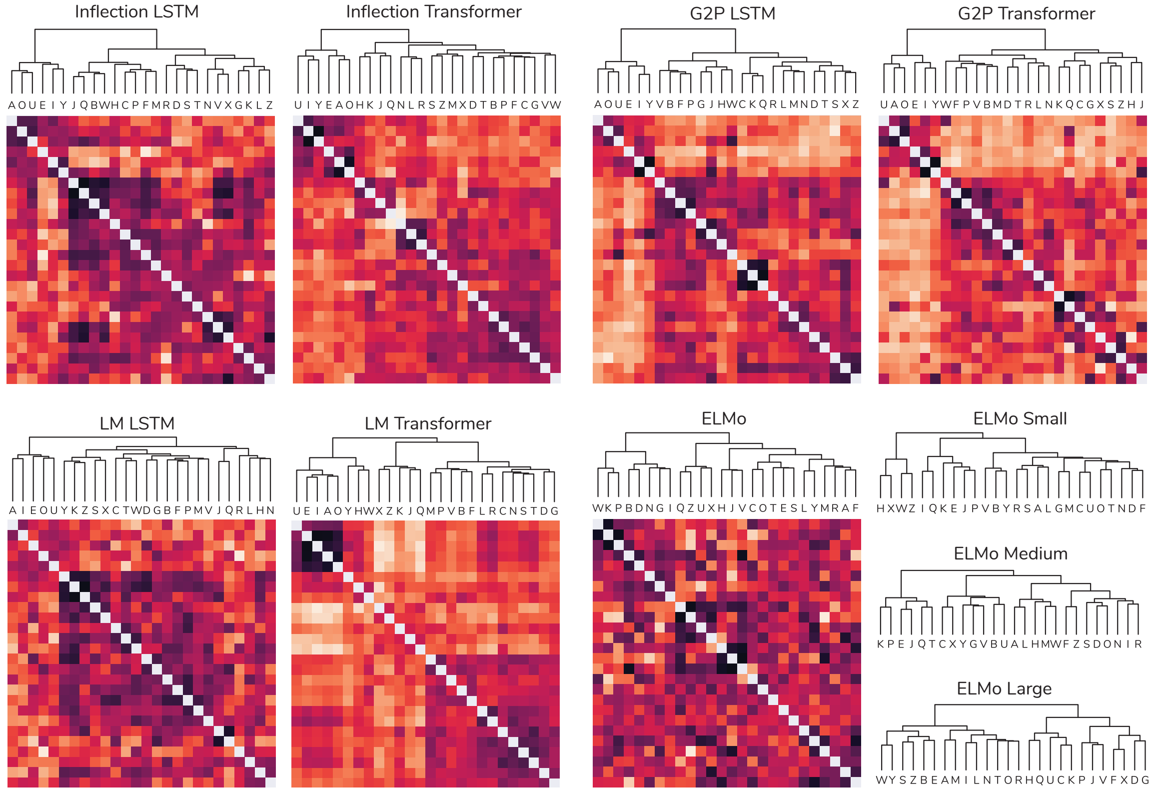


Figure 5: Distance matrices and corresponding dendrograms reveal the cluster structure of character embeddings. Darker colors depict small distances (high similarity) between pairs. Dendrograms summarize the cluster structure, with the height of horizontal lines depicting the Ward distance between the corresponding clusters being joined.

RoBERTa (Liu et al., 2019). Similar to attention in other models, attention in BERT has been investigated exhaustively. Clark et al. (2019) found that it captures substantial syntactic information, and Vig (2019) built a visualization tool for the attention mechanism. Hewitt and Manning (2019) evaluated whether syntax trees could be recovered from BERT or ELMo's word representation space. An overview of over 40 different studies of BERT can be found in Rogers et al. (2020).

Embedding analysis. The research which is closest to our work investigates which information is captured by different types of embeddings, often by training a classifier to predict certain features of interest. For instance, Kann et al. (2019) used a classifier-based approach to examine whether word and sentence embeddings encode information about the frame-selectional properties of verbs. Ettinger et al. (2016) investigated the grammatical information contained in sentence embeddings regarding multiple linguistic phenomena. Qian et al. (2016) mapped a dense embedding to a sparse linguistic property space to explore the contained information. Bjerva and Augenstein (2018) studied language embeddings.

Different word similarity datasets have been used for word embedding evaluation, for instance RG-65 (Rubenstein and Goodenough, 1965), WordSim-353 (Finkelstein et al., 2002), or SimLex-999 (Hill et al., 2015).

In contrast to the work in this paragraph, which was concerned with word or sentence embeddings, we aim at understanding *character* embeddings.

8 Conclusion

In this paper, we performed an in-depth analysis of character embeddings extracted from various character-level models for NLP. We leveraged resources from research on grapheme–color synesthesia – a neuropsychological phenomenon where letters are associated with colors –, to construct a dataset for a character similarity task. We further performed an analysis of networks representing characters as nodes and similarities as edge weights to understand how characters are organized by human synesthetes in comparison to character embeddings. Analysing 10 different character embeddings, we found that LSTMs agreed with humans more than transformer models. Comparing different tasks, G2P resulted in embeddings more similar to human character representions than inflection and, by a wide margin, language modeling. ELMo embeddings differed from humans and other models in that they exhibited no clear structure.

Acknowledgments

We would like to thank Manuel Mager and the anonymous reviewers for their feedback on this work. Mauro M. Monsalve-Mercado acknowledges financial support from the Center for Theoretical Neuroscience at Columbia University through the NSF NeuroNex Award DBI-1707398 and The Gatsby Charitable Foundation.

References

Noga Alon and Assaf Naor. 2004. Approximating the cut-norm via grothendieck's inequality. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '04, page 72–80, New York, NY, USA. Association for Computing Machinery.

Michiko Asano and Kazuhiko Yokosawa. 2013. Grapheme learning and grapheme-color synesthesia: toward a comprehensive model of grapheme-color association. *Frontiers in human neuroscience*, 7:757.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Johannes Bjerva and Isabelle Augenstein. 2018. From phonology to syntax: Unsupervised linguistic typology at different levels with language embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 907–916, New Orleans, Louisiana. Association for Computational Linguistics.

Ulrik Brandes. 2008. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136 – 145.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does BERT look at? An analysis of BERT's attention. *arXiv preprint arXiv:1906.04341*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

R.E. Cytowic. 2018. *Synesthesia*. MIT Press Essential Knowledge series. MIT Press.

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

David M Eagleman, Arielle D Kagan, Stephanie S Nelson, Deepak Sagaram, and Anand K Sarma. 2007. A standardized test battery for the study of synesthesia. *Journal of neuroscience methods*, 159(1):139–145.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 134–139, Berlin, Germany. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Trans. Inf. Syst.*, 20(1):116–131.

Alan Frieze and Ravindran Kannan. 1999. Quick approximation to matrices and applications. *Combinatorica*, 19:175–220.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Edward M. Hubbard, A. Cyrus Arman, Vilayanur S. Ramachandran, and Geoffrey M. Boynton. 2005. Individual differences among grapheme-color synesthetes: Brain-behavior correlations. *Neuron*, 45(6):975 – 985.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 555–560, Berlin, Germany. Association for Computational Linguistics.

Katharina Kann, Alex Warstadt, Adina Williams, and Samuel R. Bowman. 2019. Verb argument structure alternations in word and sentence embeddings. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 287–297.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

Aleksandra Mroczko-Wasowicz and Danko Nikolic. 2014. Semantic mechanisms may be responsible for developing synesthesia. *Frontiers in Human Neuroscience*, 8:509.

Aaron Mueller, Garrett Nicolai, Panayiota Petrou-Zeniou, Natalia Talmina, and Tal Linzen. 2020. Cross-linguistic syntactic evaluation of word prediction models. *arXiv preprint arXiv:2005.00187.*

M. Newman. 2018. *Networks*. OUP Oxford.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: Wen and why does it work? *arXiv preprint arXiv:2005.00628.*

Peng Qian, Xipeng Qiu, and Xuanjing Huang. 2016. Investigating language universal and specific properties in word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1478–1488, Berlin, Germany. Association for Computational Linguistics.

V. S. Ramachandran and E. M. Hubbard. 2001a. Psychophysical investigations into the neural basis of synaesthesia. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1470):979–983.

V. S. Ramachandran and E. M. Hubbard. 2001b. Synaesthesia–a window into perception, thought and language. *Journal of Consciousness Studies*, 8(12):3–34.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *arXiv preprint arXiv:2002.12327.*

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633.

Jari Saramäki, Mikko Kivelä, Jukka-Pekka Onnela, Kimmo Kaski, and János Kertész. 2007. Generalizations of the clustering coefficient to weighted complex networks. *Phys. Rev. E*, 75:027105.

Gaurav Sharma, Wencheng Wu, and Edul N. Dalal. 2005. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30.

Julia Simner. 2007. Beyond perception: Synaesthesia as a psycholinguistic phenomenon. *Trends in Cognitive Sciences*, 11(1):23–29.

Julia Simner. 2012. Defining synaesthesia. *British journal of psychology*, 103(1):1–15.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Jesse Vig. 2019. Visualizing attention in transformer-based language representation models. *arXiv preprint arXiv:1904.02679*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Joe Ward. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2019. Blimp: A benchmark of linguistic minimal pairs for english. *arXiv preprint arXiv:1912.00582*.

Miaomiao Wen, Zeyu Zheng, Hyeju Jang, Guang Xiang, and Carolyn Penstein Rosé. 2013. Extracting events with informal temporal references in personal histories in online communities. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 836–842, Sofia, Bulgaria. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Nathan Witthoft, Jonathan Winawer, and David M Eagleman. 2015. Prevalence of learned grapheme-color pairings in a large online sample of synesthetes. *PLoS One*, 10(3).

Beilei Xiang, Changbing Yang, Yu Li, Alex Warstadt, and Katharina Kann. 2021. CLiMP: A benchmark for Chinese language model evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361, Brussels, Belgium. Association for Computational Linguistics.

Appendix A: Statistics

X–Z, M–N, R–X, K–X, H–N, M–R, O–U, M–W, V–Z, O–Q
B–Y, B–C, C–R, C–M, M–Y, R–Y, C–X, I–R, P–Y, C–Z

Table 2: The most similar character pairs in descending order (top) and the most dissimilar character pairs in ascending order (bottom) for human synesthetes.

Appendix B: Analysis of Color Differences of Characters as Perceived by Synesthetes

In additional to the most important findings mentioned in the main part of this paper, we further present an in-depth study of the character similarities according to human synesthetes in this section.

Clustering Analysis. For each individual, we compute a character difference matrix using CIEDI2000 (normalized to values between 0 and 1). We then use Ward's hierarchical clustering algorithm to explore hidden structural features. Several examples suggest that individual synesthetes tend to represent certain groups of letters with closely matching colors, since their perceptual color differences tend to form tight clusters (Figure 6A), as opposed to the node-to-node average over the entire population of synesthetes (Figure 6B). To make this finding concrete, we compute several measures aimed to address the degree of clustering of each network.

First, we compute the distances between identified clusters in the distance matrix for each individual (Figure 6C). Pooled over the whole population, the distribution of cluster distances reveal an over-representation of small distances when compared to shuffled data. Over-represented small clustered distances and large jumps to big distances implies the existence of just a handful of tight clusters with small within-cluster distances and larger inter-cluster distances, as suggested by the dendrogram examples in Figure 6A.

Imposing a cut-off cluster distance for the dendrograms, we effectively select the cluster structure of the distance matrix. Although several methods for selecting the cut-off have been studied, they often strongly depend on the nature of the dataset in question. Using three largely different cut-offs we find robustly that each distance matrix encodes only a handful of tight clusters (cf. Figures 6D, E), typically around 3 to 5 clusters with an average size of 6 letters per cluster.

Clustering coefficient. Next, we compute the local clustering coefficient for each node in each distance matrix and observe no strong differences among nodes belonging to the same matrix. For each matrix, we average the cluster coefficient over all nodes and look at their distribution by pooling over all individuals (cf. Figure 6F). This reveals a narrow distribution of low clustering coefficients peaked around 0.4, implying a small average distance of any node to its neighbours. For comparison, we compute the clustering coefficient for a random uniform distance matrix (symmetric with zeroes in its diagonal) whose coefficient places at 0.5 (averaged over 100 iterations), and also for a homogeneous distance matrix (all entries are equal with a zero diagonal) with a coefficient of 1. Moreover, we repeat the analysis for the node-to-node average distance matrix and show a relatively higher coefficient (0.8), implying higher distances on average for any node with respect to its neighbors.

Betweenness centrality. Next, we examine the betweenness centrality of nodes (cf. Figure 6H). We compute this measure on the auxiliary similarity matrix $0.5 - d$, where d is a distance matrix, in order to interpret high betweenness values as the most important nodes. The distribution over all individuals reveals the top-ranking nodes as `CEILOSY`. These are the nodes found to be in most shortest paths between pair of nodes, and possibly lie at the intersection between otherwise separate clusters.

Cut-distance. Finally, cut-distances between the individual distance matrices and a reference matrix offer an additional characterization of their global structure. We compute cut-distances with respect to a zero matrix (the cut-norm), a random and an homogeneous matrix (defined as in the last paragraph), and with respect to shuffled versions of themselves, averaged over 100 iterations (cf. Figure 6G). We find narrow distributions suggesting all individuals share a similar global structure.

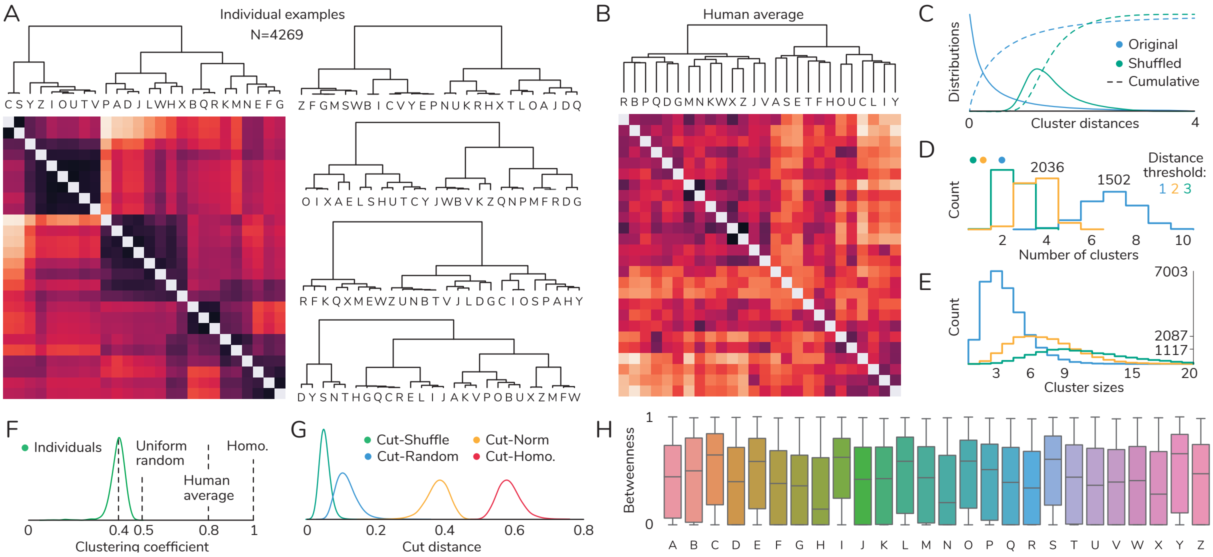


Figure 6: Network analysis of human synesthetes. **(A)** Five randomly chosen examples illustrate the typical distance matrices for individual synesthetes (N=4269). The dendrogram summarizing the clustering structure is presented on top of its corresponding distance matrix for one example (left). The dendrogram is a tree-level representation of the identified clusters, where the height of each horizontal line represents the Ward distance between the selected pair of clusters. **(B)** The clustered distance matrix and corresponding dendrogram of the node-to-node average for all human synesthetes. **(C)** Distribution of all the distances between clusters for each distance matrix pooled over all individuals. Cluster distances correspond to the heights of all horizontal lines in each dendrogram. For comparison, the distribution of cluster distances corresponding to all shuffled distance matrices is presented, together with their cumulative distributions. **(D)** Histograms of the number of clusters in each distance matrix pooled over all individuals for three different cutoff cluster distances. **(E)** Histograms of the size of the clusters found for the procedure in (D). **(F)** Distribution of the average clustering coefficient for each distance matrix pooled over all individuals. For comparison the average clustering coefficient for a random distance matrix, for the human average of (B), and for an homogeneous distance matrix are also marked by dashed lines. **(G)** Distribution of cut-distances between all the human distance matrices and the zero matrix (cut-norm), a random distance matrix (average over 100 iterations), a homogeneous distance matrix, and shuffle versions of themselves (average over 100 iterations). **(H)** Distribution of betweenness centrality for each letter pooled over all individual synesthetes.

How Good (really) are Grammatical Error Correction Systems?

Alla Rozovskaya
Queens College, CUNY
`arozovskaya@qc.cuny.edu`

Dan Roth
University of Pennsylvania
`danroth@seas.upenn.edu`

Abstract

Standard evaluations of Grammatical Error Correction (GEC) systems make use of a fixed reference text generated relative to the original text; they show, even when using multiple references, that we have a long way to go. This analysis paper studies the performance of GEC systems relative to *closest-gold* – a gold reference text created relative to the *output* of a system. Surprisingly, we show that the real performance is 20-40 points better than standard evaluations show. Moreover, the performance remains high even when considering any of the top-10 hypotheses produced by a system. Importantly, the type of mistakes corrected by lower-ranked hypotheses differs in interesting ways from the top one, providing an opportunity to focus on a range of errors – local spelling and grammar edits vs. more complex lexical improvements. Our study shows these results in English and Russian, and thus provides a preliminary proposal for a more realistic evaluation of GEC systems.

1 Introduction

Grammatical Error Correction (GEC) systems are typically evaluated using reference-based evaluation measures. This is common in language generation tasks, where the system output is compared against a set of gold references, such as the set of correct translations in Machine Translation or the set of valid corrections for a source sentence in GEC. Importantly, the references are generated relative to the original text and are independent of the system outputs. In GEC, the space of valid outputs for a given source sentence is very large, making it extremely difficult to evaluate. Specifically, *reference-based evaluations (most GEC datasets contain one reference correction) are known to underestimate system performance* (Chodorow et al., 2012; Felice and Briscoe, 2015; Bryant and Ng, 2015). Bryant and Ng (2015) showed that using two references is better than one, and the results improve further with more references; however, they used references that were generated relative to the original text. Choshen and Abend (2018b) further demonstrated that the issue can be only slightly alleviated but not completely solved by increasing the number of references. This is because many errors have a long-tailed distribution on valid corrections. One can expect that as GEC systems mature and manage to address more complex errors, the underestimation of their performance will be further exacerbated.

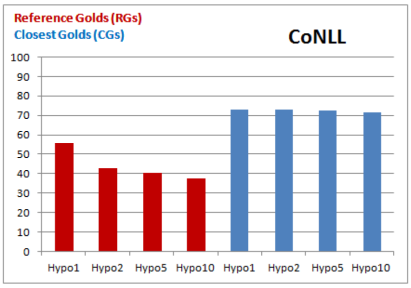


Figure 1: Performance by hypothesis rank (F-score) against Reference Gold (RG) vs. Closest Golds (CGs) generated for each hypothesis. Observe dramatic drop in performance between top hypothesis and the rest in RG evaluation, vs. stability in CG evaluation; and, large gaps between scores even for the top hypotheses in RG vs. CG evaluations.

Choshen and Abend (2018b) discuss another consequence of having a large space of valid corrections, pertaining to training. They show that *GEC systems have a strong tendency to under-correct*, due to using the one-reference-gold approach for tuning (and training) the systems. Essentially, due to the low likelihood of a system's proposed change being matched to gold, GEC systems are discouraged from proposing corrections, and propose far fewer corrections compared to humans. The under-correction phenomenon is more pronounced for errors with a large number

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2686–2698
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

of correction candidates. For example, mistakes on closed-class words, e.g. errors in determiners, where the number of valid corrections is small, suffer from under-correction to a lesser extent than mistakes in word choice. As a result, current systems generally prefer to make small targeted changes on closed-class errors.

We further study the effects of having a large space of valid corrections on GEC system development and automatic evaluation. Given a potentially erroneous sentence, we assume there is a space of gold references corresponding to it. Evaluation is typically done by drawing one gold from this set. We refer to this as *reference gold* (RG). We generate a new gold that is as close as possible to the system output (*hypothesis*), by correcting the hypothesis itself, instead of the original text. We call it *closest gold* (CG). We show by how much performance is really underestimated when a reference gold is used instead of the closest one, and claim that the latter should reflect the true performance of the system. We use a ranked 10-best list of hypotheses for a given source sentence, produced by state-of-the-art GEC systems on two English and two Russian GEC datasets. We generate CGs for hypotheses at different ranks.

Our findings are as follows. First, evaluation against RGs shows a large performance gap between the top hypothesis and the rest. We show that the reason for this is that lower-ranked hypotheses propose more diverse changes, including lexical changes, that have a lower chance to match RGs. In contrast, evaluation against CGs reveals that qualitatively, there is very little degradation in the hypotheses, when considering the top-10 list. While RG evaluation reveals severe drops in F-score and, in particular, precision, we find that relative to CGs quality does not substantially degrade. This is illustrated in Figure 1 for one of the datasets; we show more results in Section 4.1.

Second, contrary to the observation made by Choshen and Abend (2018a) about GEC systems being disinsentivized to propose corrections, we find that it only applies to the top-ranked hypothesis.[1] Moreover, the number of proposed edits increases steadily with the hypothesis rank.

We further evaluate the output by computing the post-editing effort, i.e. the number of edits needed to correct the output hypothesis. We show that post-editing effort is very similar for top hypothesis and lower-ranked hypotheses, reinforcing the claim that lower-ranked hypotheses do not degrade in quality. Finally, we evaluate the types of edits by hypothesis rank and show that lower-ranked hypotheses propose more diverse lexical changes, in contrast to the top hypotheses that mostly attempt local spelling and grammar misuse.

Our analysis should provide insight into better training and evaluation practices for GEC. A better understanding of the under-correction phenomenon and the diversity and quality of the lower-ranked hypotheses can help improve the current training and tuning framework that relies on texts with single RGs and, arguably, hinders the development of GEC systems that can potentially address more complex linguistic phenomena.

Next, we discuss reference-based evaluation. Section 3 presents the definitions and experimental setup. Section 4 presents the evaluation using closest golds. Section 5 analyzes the edits proposed by top and lower-ranked hypotheses.

2 Reference-Based Evaluation

The standard approach to evaluating GEC systems is to use reference-based measures, that is comparing the system output to a reference that has been generated by a human annotator who corrected mistakes in the original source sentence. We refer to these as reference golds (RGs). It is common to instruct annotators to follow the principle of “minimal edits”, that is making the smallest number of edits to render the sentence grammatical and well-formed. We follow a similar principle with our annotators, and the key distinction of our approach is that standard evaluations use golds that are independent of the system outputs, whereas we are creating golds by directly correcting the hypotheses output by the system. We note that there have been other proposals that argue that this principle still does not make the output fluent and propose generating references based on fluency (Sakaguchi et al., 2016). As suggested by Choshen and Abend (2018b), correcting for fluency further increases the space of valid corrections for a sentence, and we do not attempt to do this in this work.

Reference-based evaluations include several measures, such as the MaxMatch scorer M^2 (Dahlmeier and Ng, 2012), GLEU (Napoles et al., 2015), ERRANT (Bryant et al., 2017), and I-measure (Felice and Briscoe, 2015). These met-

---

[1] The proposed corrections are all corrections suggested by a system, some of which could be wrong.

rics have some commonalities, e.g. both MaxMatch and ERRANT measure precision, recall, and F-score. M^2 has been used with different beta parameter values, the default is $beta = 0.5$, weighting precision twice as high as recall, which is more common than assigning equal weights and has been shown to have stronger correlation with human ratings (Grundkiewicz et al., 2015). GLEU focuses on the fluency aspect – it is an extension of the BLEU metric in Machine Translation (Papineni et al., 2002). I-measure emphasizes accuracy and calculates the weighted accuracy of correction and detection. Napoles et al. (2019) proposed GMEG-Metric, that is an ensemble of existing metrics, and showed that its correlation with human judgments is higher on several GEC datasets. The MaxMatch metric has been widely used in evaluating GEC systems in many published works and in several shared tasks (Ng et al., 2013, 2014), and we adopt it in this work. We use the default beta value of 0.5 and refer to the result as F-score.

3 Definitions and Experimental Setup

We start with some definitions and then describe the experimental setup.

3.1 Definitions

Given the original learner sentence (a *source* sentence), a state-of-the-art (neural) GEC system generates a ranked list of outputs, referred to as *hypotheses*. We refer to the top hypothesis as H_1, and, similarly, to other hypotheses by the rank that they occupy. A system is evaluated using reference-based metrics where for each source sentence there is at least one corresponding corrected version that was generated by a human expert. We refer to this corrected version as *Reference Gold* (RG). The set of possible correct versions for a given source sentence is very large – possibly infinite – and any single reference gold is just a single point in that space. Most of the GEC evaluation sets contain one RG for each source sentence, although some (English) datasets contain more (CoNLL-test has 2 and an additional set of 8 generated later, and JFLEG (Napoles et al., 2017) has 4 fluency-based references). System performance is computed by scoring the top-ranked hypothesis H_1 for each sentence against the corresponding RG.

In addition to RGs, we create for each pair of (source, H_i), where H_i is the system hypothesis, another gold, which is generated by an expert by correcting the hypothesis itself. We refer to this gold as closest gold (CG_i) relative to H_i. The annotators who generated CGs were instructed to apply the minimal edit principle – i.e. correct the output to ensure it is grammatical and also preserves the meaning of the original source sentence. We thus assume that CG is as close as possible to the system output.

Given a pair of sentences, edit distance is the minimum number of edits (deletions, replacements, insertions, not necessarily single-tokens) needed, so that the sentences match. A *gold edit* is an edit between a source sentence and an RG or CG. A *proposed edit* is an edit between a source sentence and a hypothesis. A *correct edit* is an edit in the intersection of gold and proposed edits. We define Dist (S,RG) to be the number of edits between source and reference gold, and Dist (H_i,CG_i) to be the number of edits between a hypothesis H_i and CG_i relative to this hypothesis. The last one is interesting for practical purposes, since it is the post-editing effort required to completely correct the text. These are shown below.

- *S* – original text
- H_i – hypothesis at rank i
- *RG* – reference gold
- CG_i – closest gold to hypothesis H_i
- *Gold edit* – an edit between a source sentence and an RG or CG
- *Proposed edit* – an edit between a source sentence and a system hypothesis
- *Correct edit* – a proposed edit that is also a gold edit relative to a system hypothesis and specific reference
- *Dist (S,RG)* – edit distance between source and reference gold
- *Dist (H_i, RG)* – edit distance between hypothesis at rank i and reference gold
- *Dist (H_i, CG_i)* – edit distance between hypothesis at rank i and its closest gold

Table 1 shows a sample source sentence, 2 system hypotheses, the RG, and two CGs, one for each hypothesis. Dist (S,H_1) is 2 and includes 2 proposed edits ("reallistic" → "realistic" and "a" → ∅). Dist (S,RG) is 2 and includes 2 gold edits ("reallistic" → "realistic" and "had" → "gave"). The number of correct edits relative to RG and H_1 is 1 ("reallistic" → "realistic"). Dist (H_{10}, RG) is 4 (4 word replacement edits and one insertion

S	In addition , I think that the settings are very reallistic and the actors had a great performance .
$\mathbf{H_1}$	In addition , I think that the settings are very realistic and the actors had great performance .
$\mathbf{H_{10}}$	In addition , I think that the settings are very realistic and the actors performed very well .
RG	In addition , I think that the settings are very realistic and the actors gave a great performance .
$\mathbf{CG_1}$	In addition , I think that the settings are very realistic and the actors had great performances .
$\mathbf{CG_{10}}$	In addition , I think that the settings are very realistic and the actors performed very well .

Table 1: Example of an original sentence (source); the system output (hypotheses at ranks 1 and 10, H_1 and H_{10}); the reference gold (RG), and two additional golds generated on top of each of the hypotheses (CG_1 and CG_{10}).

edit), while Dist (H_{10}, CG_{10}) is 0. The three golds – two CGs and the RG – illustrate the notion of semantic equivalence (multiple ways of correcting the same source sentence, while preserving its meaning), not reflected in the standard evaluation.

3.2 Experimental Setup

We perform experiments on 2 English and 2 Russian datasets, using diverse NMT GEC model frameworks. The English datasets include the commonly used benchmarks – CoNLL-14 (Ng et al., 2014; Dahlmeier et al., 2013), and the BEA corpus (Bryant et al., 2019). The Russian datasets include the RULEC-GEC corpus (Rozovskaya and Roth, 2019) (henceforth RULEC) and another dataset of Russian learner writing that has been recently collected from the online language learning platform Lang-8 (Mizumoto et al., 2011) and annotated by native speakers.[2] We refer to this dataset as Lang8. CoNLL-14 contains two primary RGs against which the systems are standardly evaluated, while the other datasets include one RG for each sentence. We report results using one RG for each dataset for uniformity, and note that the results for the second CoNLL RG are very similar. These datasets were selected with the goal of evaluating on diverse data both in terms of genre and target language.

For the English datasets, we apply a state-of-the-art BERT-Fuse NMT system that incorporates BERT into an encoder-decoder Transformer model by (Kaneko et al., 2020). We obtained a ranked hypothesis list from the authors.

For RULEC, we use the outputs of a state-of-the-art Transformer model that uses pre-training on synthetic data and is fine-tuned on RULEC development data (Naplava and Straka, 2019). For the Lang8 corpus, we use a different state-of-the-art architecture, a Convolutional Neural Network model proposed in Chollampatt and Ng (2018b) for English. We re-implement it for Russian. The model is trained on RULEC training data and synthetic data, and uses language model re-ranking. This model is also tuned on RULEC development data. Our evaluation shows that the models are competitive: the Transformer model performs better by 4 points on the RULEC corpus than the CNN model, while the CNN model outperforms the Transformer on Lang8 by 2 points. However, we stress that our goal is not to compare these models, as we selected several model architectures and datasets to provide a more comprehensive analysis and evaluation that spans across diverse models and datasets.

Generating Closest Golds We consider the top-10 ranked hypothesis list for each dataset and study four hypotheses at the ranks 1, 2 5, and 10, to evaluate the quality of the hypotheses at various ranks and to determine how much quality degrades from the top hypothesis downwards. For each of the 4 hypotheses H_i, $i \in \{1, 2, 5, 10\}$, a closest gold CG_i relative to this hypothesis is generated by post-editing the hypothesis for grammatical errors and other misuse.

Annotation 100 source sentences from each dataset were selected uniformly at random and 4 hypotheses at different ranks were annotated for each sentence. The English outputs were annotated by two annotators – one native English speaker and a fluent non-native speaker. Each annotator corrected all hypotheses for one of the datasets. This was done to ensure consistency across different hypotheses. The Russian outputs were corrected by one native Russian speaker. All of the annotators have a Master's degree and previous annotation experience. The annotators followed the standard annotation protocol in grammar correction, in that they were instructed to follow the minimal-edits principle in correcting the sentences, while also ensuring the output is well-formed and adequate (i.e. the meaning of the original source sentence is preserved), for which they also consulted the source sentence.

---

[2] This is a recently collected dataset that will be made available for research.

4 Evaluating True System Performance

We start by evaluating each hypothesis output H_i for each dataset against reference gold RG and its corresponding closest gold CG_i. We show that evaluation relative to RG is always pessimistic and, given a hypothesis generated by a GEC system, there is always a much better gold.

4.1 Reference Gold vs. Closest Gold

Table 2 shows the results of evaluating each system hypothesis against reference golds and closest golds for two datasets – BEA (English) and RULEC (Russian). Results for all datasets are in Appendix (Table 7). The CG result is significantly higher than the performance relative to RG in all cases. For the top hypothesis, the F-scores increase by 19 points on BEA and 17 points on RULEC. Improvements are greater for lower-ranked hypotheses. The improvements for BEA are 34, 36, and 37, for ranks 2, 5 and 10, respectively, and for RULEC – 34, 28, and 31.

The most substantial changes occur in precision: between 23 and 41 points on BEA, and 24 and 40 for RULEC (similar changes for the other datasets). It should be emphasized that precision improvements relative to RG are greater for lower-ranked hypotheses. This is interesting and suggests that while lower-ranked hypotheses propose significantly more changes than the top-ranked one (see column "Proposed edits"), a lot of those edits are valid corrections, even though they are not recognized in RGs: observe that despite the fact that more edits are being proposed with lower hypotheses rank, the number of correct edits (shown in the table) relative to RG goes down. For instance, 84 out of 125 proposed edits are correct in BEA H_1, while only 75 out of 200 proposed are valid in H_5. This is consistent across the datasets and indicates that *changes proposed in lower-ranked hypotheses are less likely to be included in the RGs.*

Recall is also improved in CGs relative to RGs, although not as dramatically. Recall increases by 10-25 points on CoNLL, 12-34 points for BEA, 7-22 points for RULEC, and 2-12 points for Lang8.

The results in the table strongly indicate that the n-best list does not produce hypotheses of degrading quality. On the contrary, the precision of the proposed corrections remains impressively high (in most cases, well above 50 and often into 70 or 80), which is not reflected in the reference-based evaluation scoring against a reference gold. We further illustrate the finding in Figure 2, where for each dataset, we show F-scores of the 4 hypotheses against RGs, and scores against their corresponding CGs. The first observation is that performances in the first group are much lower than in the second group for each corpus. But it is also clear that the first group shows strong degradation relative to the top-ranked hypothesis in the RG evaluation – performance goes down as you go from H_1 to H_{10}, while in the second group the performance remains almost the same across the four hypotheses.

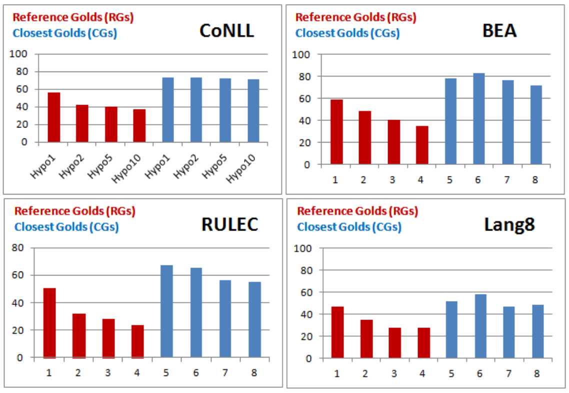


Figure 2: Performance by hypothesis rank (F-score) against Reference Gold (RG) vs. Closest Golds (CGs) generated specially for each hypothesis. Observe dramatic drop in performance between top hypothesis and the rest in RG evaluation, vs. stability in CG evaluation; and large gaps between scores even for the top hypotheses in RG vs. CG evaluations.

Further, as shown in Table 2, the number of correct edits is significantly higher when evaluated against CGs. For instance, the number of correct edits increases from 75 to 163 for BEA H_5, and from 48 to 105 for RULEC H_2. Additionally, the number of gold edits in CGs is much higher than in RGs, and is also greater for lower-ranked hypotheses. For instance, there are 202 golds edits in BEA RG, 217 edits in BEA CG_1 and 282 edits in BEA CG_{10}. This is consistent across the datasets and suggests that *the edits proposed by the models are not necessarily the minimal edits that most of the GEC annotations adopt*. This may be why most of the proposed edits in the lower-ranked hypotheses are not found in the RGs. Table 8 in Appendix also evaluates each hypothesis against CGs relative to the other hypotheses, showing that evaluation against CG always produces superior results.

Dataset	Hypo	Gold type	P	R	F-score	Edits		
						Correct	Proposed	Gold
BEA	H_1	RG	65.9	40.1	58.4	84	125	202
		CG_1	88.4	52.5	**77.8**	114	125	217
	H_2	RG	50.0	43.1	48.4	87	180	202
		CG_2	87.8	66.4	**82.5**	158	180	238
	H_5	RG	41.2	37.1	40.3	75	200	202
		CG_5	81.5	61.0	**76.4**	163	200	267
	H_{10}	RG	34.8	35.1	34.9	71	220	202
		CG_{10}	75.0	59.6	**71.3**	168	220	282
RULEC	H_1	RG	63.6	27.7	50.5	56	90	202
		CG_1	87.5	34.7	**67.1**	77	90	222
	H_2	RG	34.8	23.8	31.8	48	144	202
		CG_2	74.5	43.4	**65.1**	105	144	242
	H_5	RG	29.2	24.3	28.0	49	174	202
		CG-H_5	61.6	41.6	**56.2**	109	174	262
	H_{10}	RG	24.0	21.3	23.4	43	194	202
		CG_{10}	59.1	43.0	**55.0**	110	194	256

Table 2: Performance by hypothesis rank against reference gold (RG) and Closest Golds (CGs) generated specially for each hypothesis. For each hypothesis, number of correct, proposed, and gold edits relative to each reference (RG or CG) are also shown. Results for all the datasets are in Appendix (Table 7).

4.2 Quality Estimation with Edit Rate

We have shown above that the quality of the hypotheses does not degrade with rank, and in some cases hypotheses at lower ranks even result in higher F-score than the top-ranked hypothesis, when scored against CG, while the evaluation against RGs is strongly biased against lower-ranked hypotheses. We now wish to evaluate hypotheses quality using the edit rate, i.e. the number of edits needed to fix the output hypothesis so that it matches its corresponding CG. This quality estimation approach that considers the number of edits required to “fix” the hypothesis is used in Machine Translation (Snover et al., 2006), where the quality of a system output is measured as the minimum number of edits needed to transform the system output so that it matches a reference. To this end, a “targeted” reference is created for a translated sentence, by editing the hypothesis until it is both fluent and has the same meaning as the (original) reference(s). The reason for this is that estimating quality against gold “non-targeted” reference ignores notions of semantic equivalence (see also Table 1), thereby underestimating output quality. Thus, a targeted reference provides a more accurate measure of translation quality. Chollampatt and Ng (2018a) proposed a quality estimation model for GEC that builds on this idea of measuring output quality as the number of edits required to fix the hypothesis. However, they make the strong assumption that, unlike in MT, in GEC targeted references need not be created, as RGs can be substituted for CGs, because both human annotators and automatic GEC systems are trained to make minimal changes. As we showed in the previous section, this is not the case and *using RGs severely underestimates system performance, and, as a consequence, post-editing effort.*

We now use CGs to estimate hypotheses quality in terms of post-editing effort in Table 3. We show the number of proposed edits, the number of correct edits relative to CG, and the number of gold edits in the corresponding CG. (The number of proposed, gold, and correct edits also appears in Tables 2 and 7 but is shown here in Table 3 for convenience). The post-editing effort is shown in the last column, estimated as the number of edits required to make the hypothesis output fluent and grammatical, i.e. the edit distance between a hypothesis and its corresponding CG. The number of edits was computed automatically using the ERRANT tool (Bryant et al., 2017) that, given a pair of sentences (source, hypothesis), will produce a set of edits needed to transform the source into the target. The post-editing effort is not necessarily the smallest for the top hypothesis. In fact, on BEA, the smallest value is obtained for H_2 (55 edits), while H_1 and H_{10} are similar (86 and 87). On the other datasets, there is no significant difference for the English datasets across the 4 different hypotheses, while on the Russian datasets there is slight degradation for hypotheses 5 and 10, while H_1 and H_2 are close. This supports our finding above that lower-ranked hypotheses are of high quality. As a side note, our post-edit estimation assumes that there are no errors that have

Dataset	**Hypo**	**Edits**			
		Prop.	**Corr.**	**Gold**	**Post-Edit**
CoNLL	H_1	156	132	300	**130**
	H_2	203	160	349	159
	H_5	239	184	343	138
	H_{10}	266	196	349	142
BEA	H_1	125	114	217	86
	H_2	180	158	238	**55**
	H_5	200	163	267	73
	H_{10}	220	168	282	87
RULEC	H_1	90	88	222	**119**
	H_2	144	105	242	131
	H_5	174	109	262	173
	H_{10}	194	110	256	186
Lang8	H_1	98	65	252	**168**
	H_2	186	117	287	174
	H_5	214	105	298	215
	H_{10}	225	109	312	225

Table 3: Number of edits by hypothesis rank. We show the number of proposed edits, the number of correct edits relative to CG, the number of gold edits in CG, and the post-editing effort required to make the hypothesis fluent and grammatical, estimated as the number of edits between the hypothesis and its CG.

more impact than others. In Section 5, we actually show that the top-ranked hypotheses mostly contain changes on "simpler errors", and, arguably, the lower-ranked hypotheses might even involve less post-editing effort given the more complex errors they manage to fix.

4.3 Do GEC Systems Undercorrect?

We first compare the number of edits in each hypothesis to the number of edits in the original gold (Table 4). The top-ranked hypothesis makes only a fraction of edits compared to RGs. Generally, RGs contain 2.5-3 more edits than the top-1 hypothesis. This is consistent with the analysis in Choshen and Abend (2018b) that shows that GEC systems are disincentivized to make corrections due to the low-coverage bias. What is notable, however, is that the number of edits substantially increases with the hypothesis rank. In particular, *the second-ranked hypothesis contains on average twice as many edits as the first one*, and the number of edits continues to increase. Hypotheses 5 and 10 contain a similar number of edits compared to the number of edits in RG. The under-correction issue is further studied in the next section.

Hypo	**Number of edits proposed**			
	RULEC (2,646)	Lang8 (2,260)	BEA (2,103)	CoNLL (2,665)
H_1	90	98	125	156
H_2	144	186	180	203
H_5	174	214	200	239
H_{10}	194	225	220	266
RG	202	232	202	289

Table 4: Number of proposed edits by hypothesis rank. For each corpus, total number of tokens is shown. The majority of edits are single-token replacements, deletions or insertions. The last row shows the number of gold edits in the *reference gold* for each dataset.

5 Edit Analysis by Hypothesis Rank

We now analyze and compare the edits in the top-ranked hypothesis and in H_{10}, in order to understand better how the edits differ with hypothesis rank. For the English datasets, we apply ERRANT (Bryant et al., 2017) to extract edits using pairs of parallel sentences (source, hypothesis). ERRANT then uses English-language specific rules based on part-of-speech and linguistic knowledge to assign each edit its linguistic type, such as preposition, noun number, etc. We further group the edits into one of the following two categories: *spelling/grammar changes* and *lexical changes*. The first category includes punctuation, spelling, orthography, and grammatical corrections that typically require local context and small changes and are also limited in the number of candidate corrections. These include determiner errors, verb agreement and form, noun number and punctuation, and morphological changes. Lexical changes comprise the categories denoted by ERRANT as "Other", "Verb", "Noun", "Pronoun", "Adverb", which include mostly lexical errors, e.g. changing "get" to "earn", verb tense errors that require wider context and thus are trickier to correct. The number of edits by type is shown in Table 5. Lexical changes are marked with a (*).

In the lower part of the table, we show the distribution of edits between the two categories: in CoNLL, spell/grammar changes account for 51.2% of all changes in the RG and for 74.3% in the top-ranked hypothesis. Lexical changes make up 48.8% in RG, while only 25.7% in the top-ranked hypothesis, although this number increases to 36.2% in the H_{10}. In BEA, 49.5% of RG edits are lexical, while in the top-ranked hypothesis

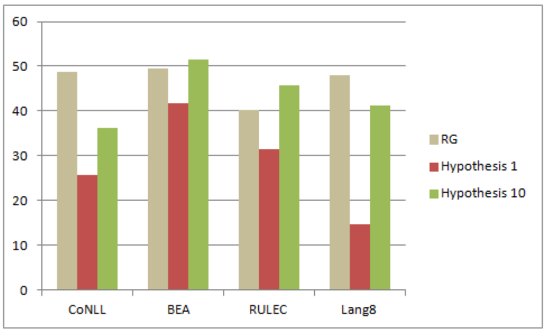


Figure 3: Percentage of lexical edits with respect to total number of changes in reference gold, H_1, and H_{10}.

these account for 41.7% and that number goes up to 51.5% for the H_{10}.

Looking at the number of edits in each category, it can be observed, that the under-correction phenomenon (for the top-ranked hypothesis) is particularly pronounced for lexical errors. In the spell-grammar category, the number of proposed edits is very close to (or even exceeds) the number of edits of this type in RG in both datasets (the only exception is perhaps the punctuation errors). For example, 33 determiner errors are present in CoNLL RG, while there are 25 in H_1. In contrast, 37 errors of category "Other" are in RG for CoNLL but only 8 changes of this type are in the top-ranked hypothesis. In fact, in both CoNLL and BEA, in the top-ranked hypotheses, the majority of the changes are minimal/local changes (74.3% in the CoNLL dataset and 58.3% in the BEA dataset).

We perform a similar analysis for the Russian datasets, where the edits are classified manually by our annotator (due to lack of automatic tool). We find similar behavior (see Appendix A). However, in the most challenging categories (lexical and "Other"), which both comprise word changes, the situation is more severe: the top-ranked hypothesis proposes 0 changes. Overall, the under-correction phenomenon for lexical errors is more pronounced for the Russian language.

Overall, the under-correction phenomenon is especially pronounced for top hypotheses in the lexical error category. The percentage of lexical edits with respect to the total number of edits in the RG and CGs is much higher than in the top hypotheses. Thus, under-correction is mostly a problem for lexical errors, but is partially rectified in the lower-ranked hypotheses, illustrated in Figure 3 that shows the percentage of lexical edits in RG, H_1, and H_{10} for each dataset.

Edit type	**CoNLL**			**BEA**		
	RG	H_1	H_{10}	RG	H_1	H_{10}
Spell/Orth	10	21	23	16	12	14
Punc	9	3	19	35	12	30
Noun number	19	25	35	7	8	14
Det	33	25	52	22	21	28
Verb agr.	14	15	13	3	3	2
Verb form	15	13	10	7	5	6
Morph.	6	5	5	5	6	6
Prep*	18	14	21	22	9	24
Verb tense*	18	7	17	11	8	19
Other*	37	8	25	37	19	38
Verb*	12	3	9	10	5	6
Noun*	4	1	9	9	2	6
Pronoun*	6	3	6	4	2	3
Adverb*	5	0	0	3	3	10
Spell/grammar (%)	51.2	74.3	63.8	50.5	58.3	48.5
Lex. changes (%)	48.8	25.7	36.2	49.5	41.7	51.5

Table 5: Proposed edits and gold RG edits by type and hypothesis rank on the English datasets. * marks lexical changes.

6 Discussion

We study the current evaluation and training schema in GEC, using 4 datasets in 2 languages and several state-of-the-art model architectures. We make several observations. First, we show that the quality of the systems is significantly better than we think, when we evaluate relative to the closest gold vs. reference gold. And the reason is there are many golds and we show that there is always a gold that is close to the prediction, and we should take this result as the actual performance of the model.[3] Moreover, as we showed, using the CGs provides additional knowledge about the type of errors various hypotheses generate, further guiding the community towards developing additional insights that can be used also in targeting specific models for specific users (based on their abilities, for example). Our second observation is that the top hypothesis is not actually better than the lower-ranked hypotheses in the 10-best list, even though the current evaluations are strongly biased towards the top hypothesis. Third, because

---

[3] In fact, given that sentence-level ensembles computed via multiple references were shown to perform much better than a single hypothesis (Bryant and Ng, 2015), already indicates the existence of better golds, since a sentence-based combination of reference gold is a gold by itself. We claim, though, that even this underestimates the true performance, as shown in our evaluation relative to the CGs.

of the way we train, lower-ranked hypotheses relative to the reference gold are as good or sometimes qualitatively better than the top hypothesis because of the diversity of the type of mistakes that they attempt to correct.

Recommendations based on the paper findings We view this paper as an analysis paper that we hope can contribute to a better understanding of the current issues in the GEC field. We hope that the proposed analysis can give an opportunity to researchers to think about directions for addressing these issues. That said, we believe that our results may serve as a preliminary proposal for developing better ways for evaluating for GEC systems, and would like to outline several recommendations based on our findings. We believe the findings should be useful for thinking about how to modify the training and tuning paradigm in GEC.

Regarding training and tuning, the current schema of using learner texts with single RGs hinders development of GEC systems that, as we show, can potentially address more complex linguistic phenomena and language misuse. For training and tuning, perhaps, it would make sense to generate multiple references by creating additional references that contain paraphrases of the original gold reference. In terms of evaluation, the findings might inspire researchers to think of better ways to evaluate GEC system outputs. For example, instead of computing exact match, we could include paraphrases so as not to penalize hypotheses that propose more liberal sentence-rewrites. A different approach might be to choose lower-ranked hypotheses, since they are as good, and they have some other useful properties, such as the language phenomena they are able to correct that the top hypothesis cannot.

Acknowledgments

We thank the annotators Mikhail Khabas and Maxwell White. We thank the reviewers for their comments. This work was supported by Contract FA8750-19-2-0201 with the US Defense Advanced Research Projects Agency (DARPA).

References

C. Bryant, M. Felice, Ø. Andersen, and T. Briscoe. 2019. The BEA-19 shared task on grammatical error correction. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

C. Bryant, M. Felice, and T. Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *ACL*.

C. Bryant and H. T. Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *ACL*.

M. Chodorow, M. Dickinson, R. Israel, and J. Tetreault. 2012. Problems in evaluating grammatical error detection systems. In *Proceedings of COLING*.

S. Chollampatt and H.T. Ng. 2018a. Neural quality estimation of grammatical error correction. In *EMNLP*.

Shamil Chollampatt and Hwee Tou Ng. 2018b. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the AAAI*. Association for the Advancement of Artificial Intelligence.

L. Choshen and O. Abend. 2018a. Automatic metric validation for grammatical error correction. In *ACL*.

L. Choshen and O. Abend. 2018b. Inherent biases in reference-based evaluation for grammatical error correction and text simplification. In *ACL*.

D. Dahlmeier and H. T. Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of EMNLP-CoNLL*.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia. Association for Computational Linguistics.

M. Felice and T. Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *NAACL-HLT*.

R. Grundkiewicz, M. Junczys-Dowmunt, and E. Gillian. 2015. Human evaluation of grammatical error correction systems. In *EMNLP*.

M. Kaneko, M. Mita, S. Kiyono, J. Suzuki, and K. Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *ACL*.

T. Mizumoto, M. Komachi, M. Nagata, and Y. Matsumoto. 2011. Mining revision log of language learning SNS for automated japanese error correction of second language learners. In *IJCNLP*.

J. Naplava and M. Straka. 2019. Grammatical error correction in low-resource scenarios. In *W-NUT Workshop*.

C. Napoles, M. Nădejde, and J. Tetreault. 2019. Enabling robust grammatical error correction in new domains: Data sets, metrics, and analyses. In *Transactions of ACL*.

C. Napoles, K. Sakaguchi, M. Post, and J. Tetreault. 2015. Ground truth for grammatical error correction metrics. In *ACL*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

H. T. Ng, S. M. Wu, Y. Wu, Ch. Hadiwinoto, and J. Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of CoNLL: Shared Task*.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland. Association for Computational Linguistics.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*.

A. Rozovskaya and D. Roth. 2019. Grammar error correction in morphologically-rich languages: The case of russian. In *Transactions of ACL*.

K. Sakaguchi, C. Napoles, M. Post, and J. Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. In *Transactions of ACL*.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Association for Machine Translation in the Americas*.

A Edit Analysis for the Russian Datasets

For the Russian corpora, since there is no automated tool that classifies edits by type, we manually classify the edits in the hypotheses and RGs. We first extract all proposed edits using the ERRANT tool (The errant tool both extracts edits given a pair of sentences, and classifies these by type). The edit-extraction component is language-independent, whereas the type-classification is English-based). These edits are then manually classified into one of the grammar categories relevant for Russian. We use the error classification schema in Rozovskaya and Roth (2019) but combine certain types, e.g. we group together noun and adjective case errors, verb tense/aspect errors, and noun and adjective number errors. Unlike English, Russian does not have determiner errors. We similarly group the error type into two categories, spelling/grammar and lexical. However, we assign the noun/adjective number errors and morphology errors to the second group (lexical), as we expect these to display more variability due to the number of of different endings for adjective/noun number because of declensions and gender, and the large number of morphological variants compared to English. The statistics are shown in Table 6.

First, observe that the distribution in the gold references of lexical and spelling/grammar changes in the lower part of the table is similar to the English datasets. 40% or more of all gold edits are lexical. In the top-ranked hypothesis, only 31.5% and 14.6% of edits in RULEC and Lang8, respectively are of this type. This is similar to the results for the English datasets in Table 5, however in the most challenging categories (lexical and "other"), which both comprise word changes, the situation is more severe: the top-ranked hypothesis proposes 0 changes. Overall, the undercorrection phenomenon is more pronounced for the Russian language.

Edit type	**RULEC**			**Lang8**		
	RG	H_1	H_{10}	RG	H_1	H_{10}
Spell	40	29	35	53	44	54
Noun/adj. case,gender,num.	46	20	43	39	19	29
Verb agr.	5	1	4	6	5	4
Punc	24	17	32	16	11	34
Prep*	11	6	18	14	6	23
Verb tense/aspect*	6	3	20	12	1	20
Noun/adj. num.*	7	1	17	13	4	23
Morph.*	15	5	13	19	4	15
Lexical*	30	0	13	30	0	5
Other changes*	20	0	4	32	0	12
Spell/grammar (%)	59.7	68.5	54.2	52.1	85.4	58.8
Lex.changes (%)	40.3	31.5	45.8	47.9	14.6	41.2

Table 6: Proposed edits and gold RG edits by type and hypothesis rank on the Russian datasets. * marks lexical changes.

B Additional Results for All Datasets

Dataset	**Hypo**	**Gold type**	**P**	**R**	**F-score**	**Proposed edits**		
						Correct	**Proposed**	**Gold**
CoNLL	H_1	RG	66.7	33.9	55.9	98	156	289
		CG_1	87.4	44.0	**73.0**	132	156	300
	H_2	RG	48.0	29.8	42.8	86	203	289
		CG_2	87.4	44.0	**73.0**	160	203	349
	H_5	RG	43.2	31.8	40.3	92	239	289
		CG_5	79.7	53.6	**72.6**	184	239	343
	H_{10}	RG	39.6	31.1	37.6	90	266	289
		CG_{10}	76.6	56.2	**71.4**	196	266	349
BEA	H_1	RG	65.9	40.1	58.4	84	125	202
		CG_1	88.4	52.5	**77.8**	114	125	217
	H_2	RG	50.0	43.1	48.4	87	180	202
		CG_2	87.8	66.4	**82.5**	158	180	238
	H_5	RG	41.2	37.1	40.3	75	200	202
		CG_5	81.5	61.0	**76.4**	163	200	267
	H_{10}	RG	34.8	35.1	34.9	71	220	202
		CG_{10}	75.0	59.6	**71.3**	168	220	282
RULEC	H_1	RG	63.6	27.7	50.5	56	90	202
		CG_1	87.5	34.7	**67.1**	77	90	222
	H_2	RG	34.8	23.8	31.8	48	144	202
		CG_2	74.5	43.4	**65.1**	105	144	242
	H_5	RG	29.2	24.3	28.0	49	174	202
		CG_5	61.6	41.6	**56.2**	109	174	262
	H_{10}	RG	24.0	21.3	23.4	43	194	202
		CG_{10}	59.1	43.0	**55.0**	110	194	256
Lang8	H_1	RG	60.9	24.1	46.7	56	98	232
		CG_1	69.2	25.8	**51.8**	65	98	252
	H_2	RG	36.9	28.5	34.8	66	186	232
		CG_2	64.6	40.8	**57.9**	117	186	287
	H_5	RG	28.6	24.1	27.6	56	214	232
		CG_5	50.7	34.9	**46.5**	105	214	298
	H_{10}	RG	28.4	23.7	27.3	55	225	232
		CG_{10}	53.2	34.9	**48.1**	109	225	312

Table 7: Performance by hypothesis rank against reference gold (RG) and Closest Golds (CGs) generated specially for each hypothesis. For each hypothesis, number of correct, proposed, and gold edits relative to each gold are also shown. Expanded version of Table 2 in Section 4.1 that showed results for BEA and RULEC only.

Dataset	Hypo	Gold type	P	R	F-score
CoNLL	H_1	RG	66.7	33.9	55.9
		Other CGs	74.0-75.8	31.8-32.9	58.5-60.2
		CG_1	87.4	44.0	**73.0**
CoNLL	H_2	RG	48.0	29.8	42.8
		Other CGs	58.9-59.6	31.2-37.3	50.0-53.2
		CG_2	83.8	45.8	**71.9**
CoNLL	H_5	RG	43.2	31.8	40.3
		Other CGs	51.8-52.3	33.0-38.7	46.5-48.8
		CG_5	79.7	53.6	**72.6**
CoNLL	H_5	RG	39.6	31.1	37.6
		Other CGs	48.8-50.2	33.8-41.0	45.2-48.0
		CG_{10}	76.6	56.2	**71.4**
BEA	H_1	RG	65.9	40.1	58.4
		Other CGs	74.2-75.2	33.3-39.9	60.1-63.7
		CG_1	88.4	52.5	**77.8**
BEA	H_2	RG	50.0	43.1	48.4
		Other CGs	60.3-67.2	41.9-49.8	56.7-60.1
		CG_2	87.8	66.4	**82.5**
BEA	H_5	RG	41.2	37.1	40.3
		Other CGs	48.7-52.7	35.1-42.4	47.3-48.6
		CG_5	81.5	61.0	**76.4**
BEA	H_{10}	RG	34.8	35.1	34.9
		Other CGs	44.5-47.7	38.2-44.7	44.5-45.7
		CG_{10}	75.0	59.6	**71.3**
RULEC	H_1	RG	63.6	27.7	50.5
		Other CGs	79.1-80.7	27.1-28.1	57.8-58.0
		CG_1	87.5	34.7	**67.1**
RULEC	H_2	RG	34.8	23.8	31.8
		Other CGs	49.3-55.3	26.3-31.1	42.0-47.6
		CG_2	74.5	43.4	**65.1**
RULEC	H_5	RG	29.2	24.3	28.0
		Other CGs	36.3-38.0	24.6-29.3	33.4-35.9
		CG_5	61.6	41.6	**56.2**
RULEC	H_{10}	RG	24.0	21.3	23.4
		Other CGs	32.8-35.6	24.0-27.0	31.4-33.3
		CG_{10}	59.1	43.0	**55.0**
Lang8	H_1	RG	60.9	24.1	46.7
		Other CGs	64.8-69.6	18.9-21.5	43.9-48.1
		CG_1	69.2	25.8	**51.8**
Lang8	H_2	RG	36.9	28.5	34.8
		Other CGs	39.9-45.6	26.3-28.2	36.8-39.7
		CG_2	64.6	40.8	**57.9**
Lang8	H_5	RG	28.6	24.1	27.6
		Other CGs	32.8-34.4	21.5-25.4	30.6-31.0
		CG_5	50.7	34.9	**46.5**
Lang8	H_{10}	RG	28.4	23.7	27.3
		Other CGs	29.2-33.9	21.8-22.2	27.5-30.5
		CG_{10}	53.2	34.9	**48.1**

Table 8: Performance by hypothesis rank against reference gold (RG) and Closest Golds (CGs) generated specially for each hypothesis.

BERTective:
Language Models and Contextual Information for Deception Detection

Tommaso Fornaciari
Bocconi University
fornaciari@unibocconi.it

Federico Bianchi
Bocconi University
f.bianchi@unibocconi.it

Massimo Poesio
Queen Mary University of London
m.poesio@qmul.ac.uk

Dirk Hovy
Bocconi University
dirk.hovy@unibocconi.it

Abstract

Spotting a lie is challenging but has an enormous potential impact on security as well as private and public safety. Several NLP methods have been proposed to classify texts as truthful or deceptive. In most cases, however, the target texts' preceding context is not considered. This is a severe limitation, as any communication takes place in context, not in a vacuum, and context can help to detect deception. We study a corpus of Italian dialogues containing deceptive statements and implement deep neural models that incorporate various linguistic contexts. We establish a new state-of-the-art identifying deception and find that not all context is equally useful to the task. Only the texts closest to the target, if from the same speaker (rather than questions by an interlocutor), boost performance. We also find that the semantic information in language models such as BERT contributes to the performance. However, BERT alone does not capture the implicit knowledge of deception cues: its contribution is conditional on the concurrent use of attention to learn cues from BERT's representations.

1 Introduction

"The sky is bright green" is easily identified as false statement under normal circumstances. However, following "Look at this surreal painting," the assessment changes. Spotting falsehoods and deception is useful in many personal, economic, legal, and political situations – but it is also extremely complicated. However, the reliability of communication is the basis of the social contract, with implications on personal, economic, legal, and political levels. There has been a growing interest in automatic deception detection from academia and industry in recent years (see section 9).

One of the main research lines tries to increase the collection of deception cues in terms of number and variety. For example, several successful studies show how to exploit multi-modal signals, jointly analyzing verbal, video, and audio data (Pérez-Rosas et al., 2015). For the same reason, several early studies tried to identify deception cues through manual feature annotation, like irony or ambiguity (Fitzpatrick and Bachenko, 2012). While these approaches offer a broad and interpretable description of the phenomenon, their main limitation lies in data collection and preprocessing difficulty.

Surprisingly, so far, little attention has been paid to expanding the targets' linguistic context, which is the easiest source of additional cues and data. Even in dialogues, which by definition are exchanges between different speakers/writers, the main focus is typically on the target text. None consider the preceding statements, be they issued by the same speaker of an interlocutor.

We hypothesize that linguistic context can be useful for text classification. Based on a data set of dialogues in Italian Courts, we train models that incorporate knowledge both from the target sentence and different configurations of the previous ones. We use Hierarchical Transformers and neural models based on BERT for text-pair representations and compare with the previous state-of-the-art methods and other non-contextual neural models, including BERT for single text representation.

We distinguish different kinds of context, depending on the window size and the speaker's identity (same one as of the target sentence or different). We find that context carries useful information for deception detection, but only if it is narrow and produced by the same author of the target text.

We also find that BERT's semantic knowledge helps the classification, but only when it is combined with neural architectures suitable to discover stylistic patterns beyond the texts' content that are potentially associated with deception.

To our knowledge, this is the first study that tests

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2699–2708
April 19 - 23, 2021.

these methods on data collected from real, high-stakes conditions for the subjects and not from a laboratory or game environment.

Contributions The contributions of this paper are as follows:

- We evaluate ways to incorporate contextual information for detecting deception on real-life data.
- We significantly outperform the previous state-of-the-art results.
- We show that language models are useful for the task, but they need the support of methods dedicated to detect deception's stylometric features.

2 Dataset

We use the DECOUR dataset (Fornaciari and Poesio, 2012), which includes courtroom data transcripts of 35 hearings for criminal proceedings held in Italian courts. This provides a unique source of real deception data. The corpus is in Italian. It consists of dialogues between an **interviewee** and some **interviewers** (such as the judge, the prosecutor, the lawyer). Each dialogue contains a sequence of utterances of the different speakers. These utterances are called *turns*. By definition, adjacent turns come from different speakers. Each turn contains one or more *utterances*. Each utterance by the interviewee is labeled as *True*, *False* or *Uncertain*. The utterances of the other speakers are not labeled. Table 1 shows some corpus and labels' statistics.

Role	Turns	Utterances	tokens
Interviewee	2094	3015	42K
Interviewers	2373	3124	87K
	4467	6139	129K

Labels:	True	Uncertain	False	Tot.
Number:	1202	868	945	3015

Table 1: DECOUR's statistics

The authors anonymized the data and released them here.

3 Experimental conditions

Fornaciari and Poesio (2013) use binary classification (*false* utterances versus the *true* and *uncertain* ones, aggregated together into one class of *non-false* utterances, see section 2, Table 1). To avoid overfitting training and testing on utterances from the same hearing, they use leave-one-out cross-validation, where each fold constitutes one hearing. In these settings, in each fold one hearing is used as test set, one as development, and the others as training set. For the sake of comparison, we followed the same approach. We ran five epochs of training for each fold, selecting the model with the best F-score in the development set.

We also identify seven kinds of different contexts that should help the classification task, together with the target utterance. They are as follows:

1 previous utterance - 1prev. We consider the first utterance preceding the target, regardless of the speaker who issued the statement.

2 previous utterances - 2prev. Same as above, but here we collect the first two sentences before the target.

3 previous utterances - 3prev. In this case, we collect the three previous utterances, again regardless of the speaker.

Speaker's previous utterance - s-utt. In this condition, we consider the utterance preceding the target only if the speaker is the same interviewee. If another speaker issues the previous utterance, it is not collected, and the target utterance remains without context.

Speaker's previous utterances - s-utts. Similarly to the previous condition, we only collect the interviewee's utterances, but if the target utterance is preceded by more than one utterance (within the same turn), they are all collected. In other words, we collect all the turn's utterances until the target one.

Speaker's previous utterances + turn - s-utturn. In these conditions, we consider all the possible speaker's utterances and the previous turn, which belongs to another speaker. If there are no previous speaker's utterances, we only collect the previous turn. This would make the instance equal to those created according to the last condition.

Previous turn - turn. We collect the whole previous turn, regardless of the possible previous speaker's utterances. This is the only condition where, by definition, the context is not produced by the interviewee him/herself.

4 Metrics and baselines

We evaluate the model on four metrics: accuracy, precision, recall and, F-measure. While accuracy is a standard metric, its informative power is limited when the data set is imbalanced, and the class of interest is the minority class, like in this case. In fact, the majority class's performance conceals the real performance on the minority one. Even so, it can be a problematic baseline to beat, as the simple heuristic of always predicting the majority class can result in high accuracy. In DECOUR, non-false utterances are the majority class with 68.66% of the instances. Therefore, this is the accuracy we would obtain always predicting the majority class. We use this majority-class prediction as a baseline. For the models' overall evaluation, we rely on the F-measure, which reflects the real proficiency of the models balancing the correct predictions in the two classes.

Besides the majority class prediction, which reaches an F-measure of 40.71, we also compare our models with the previous state-of-the-art. We use the highest performance in F-measure from Fornaciari and Poesio (2013). In that experiment, they jointly used Bag-Of-Words - BOW features and the lexical features provided by the LIWC (Pennebaker et al., 2001) and applied an SVM classifier (Drucker et al., 1997). The accuracy of that model is 70.18% and the F-measure 62.98 (table 2).

5 Methods

We perform the classification with several neural models. For all the models that do not rely on the BERT contextual embeddings (Devlin et al., 2018), we used the pre-trained Fast Text embeddings (Joulin et al., 2016) as initialization weights, and we fine-tuned them during the training process. We did not fine-tune the contextual BERT embeddings for reasons of computational load. However, the high number of the models' parameters required a low learning rate, which we manually adjusted to $1.e-4$, and a small batch size, which was 8. The drop-out probability was 0.1.

5.1 Neural baselines

We add two neural baselines: a Multi-Layer Perceptron (MLP) and a Convolutional Neural Network (CNN).

The MLP did not beat the SVM's performance. The CNN's F-measure was better than that of the SVM, but not significantly. Also, the CNN proved to be less effective than the attention-based models that did not exploit contextual information (table 2). Therefore we did not feed the MLP and the CNN with contextual information and kept them as additional neural baselines. However, to obtain their best performance possible, we carried out a comprehensive hyper-parameters search. For the MLP, we found the best results with trainable FastText embeddings followed by two hidden layers. For the CNN, we used 3 Convolutional-MaxPooling layers with 32, 64, and 128 channels, respectively, and windows' sizes of 2, 4, and 6.

5.2 Transformers-based models

Based on the success of the Transformer architecture in NLP (Vaswani et al., 2017), we used them to create two kinds of models, hierarchical and non-hierarchical. We adopted a non-hierarchical structure to analyze the target sentence alone, and we implemented Hierarchical Transformers to encode the target sentence and the contextual information jointly.

In the Hierarchical model, the input is not a single utterance but a series of utterances. We pad the maximum number of sentences to 5. This limit allows us to collect the whole text from about the 98% of the turns in DECOUR. However, as we will see in sections 6 and 8, considering a broader context would not have been useful.

Not considering the batch, the Hierarchical Transformers take as input a 3D tensor of Documents by Words by Embeddings. Each Words by Embeddings matrix is passed to a multi-layer, multi-head Transformer that provides a representation of each utterance, returning as output a tensor of the same shape of the input. A following fully-connected layer reduces the embeddings' dimension. The documents' representations are then concatenated into a 2D tensor and passed to another multi-layer, multi-head Transformer, which provides the overall document representation. Another fully connected layer is used to reduce the tensor's last dimension, which is then reshaped to a row vector. This vector is fed into the last fully connected layer that provides the prediction. Figure 1 shows such an architecture

With the Hierarchical Transformer, we run the experiments for the seven contexts described in section 3. Again, we tuned our hyper-parameters. In the hierarchical models, we used six layers and six heads Transformers for the encoders both at utterance and at documents level. For the non-hierarchical model, two layers and two heads were

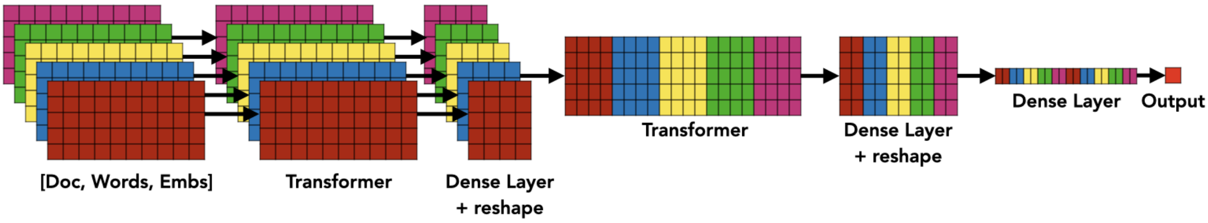


Figure 1: Hierarchical Transformers structure.

sufficient to obtain the best development set results.

5.3 BERT-based models

Finally, we perform the classification using BERT base (Devlin et al., 2018) for Italian.[1] We set up three kinds of models:

BERT + dense layer This is the simplest network, and we use it for predictions on the target utterance alone. We feed the BERT mean pooled output into a fully connected layer that performs the prediction.

BERT + Transformers This is a more expressive network, where the BERT output is passed to a multi-layer, multi-head Transformer. The Transformer's representation is then passed to a fully connected layer that outputs the prediction. We adopted Transformers with six layers and six heads, like the Hierarchical Transformers models. Similarly to the BERT + Dense model, we only feed this network with the target sentence.

text-pair BERT + Transformers The last network is structurally equal to the previous one, but in this case, we use BERT in its text-pair modality. Wet set the target sentence's size to 100 words and for the contexts to 400. The context is the concatenation of the selected texts, padded or truncated at the head. We would lose only the part of the text farthest from the target sentence in case of truncation. However, the corpus mostly contains brief statements: padding to 100 and 400 guarantees a minimum data loss. With this model, we test the seven contexts described above.

6 Results

The results are drawn in table 2.

The first group of experiments contains the baselines from the literature and simple neural networks. The second and the third group show the Transformers-based and the BERT-based models, respectively. We report Accuracy, Precision, Recall, and the F-measure. As a benchmark for the significance test, we use the literature baseline from Fornaciari and Poesio (2013) The asterisks represent the significance levels, computed via bootstrap sampling for $p \leq .05$ and $p \leq .01$. Following Søgaard et al. (2014), who recommend avoiding too small sample sizes, we set our sample at 50% of the corpus.

6.1 Overview

The results show that the SVM's performance is a strong baseline. Only a few models beat its accuracy, and none significantly. The same holds for precision. The recall is the metric where most neural models outperform SVM (significantly in five cases), even though the price they pay is a lower precision of the predictions. As a result, only four models of the 16 Transformer- and BERT-based ones show an F-Measure significantly better than SVM, corresponding to a significant improvement in the recall and better accuracy, albeit not significant. Also, a couple of deep neural models perform poorly. We will discuss them in the next sections.

6.2 Non-contextualized models

Two of the best models consider only the target sentence: the non-hierarchical Transformer and the one using BERT for single text, followed by the Transformers architecture. Despite our effort in the hyper-parameters exploration, including the use of a very low learning rate and regularization methods such as drop-out, we could not prevent that model from strong, early overfitting at a low level of performance. It seems that a single fully connected layer is unable to manage the complexity of this task, as we will discuss in section 8.

---

[1] `https://huggingface.co/dbmdz/bert-base-italian-cased`

Model	Condition	Accuracy	Precision	Recall	F-Measure
Majority class		68.66%	34.33%	50.00%	40.71%
SVM (Fornaciari and Poesio, 2013)		70.18%	64.42%	62.41%	62.98%
MLP	no context	67.16%	61.75%	61.65%	61.70%
CNN	no context	69.75%	64.98%	65.15%	65.06%
Transformers.	no context	70.98%	66.41%	**66.64** **	**66.52%** *
Hierarchical Transformers.	1 prev	68.72%	64.06%	64.51%	64.25%
Hierarchical Transformers.	2 prev	67.56%	63.04%	63.70%	63.29%
Hierarchical Transformers.	3 prev	68.13%	63.52%	64.08%	63.75%
Hierarchical Transformers.	s-utt	68.36%	64.22%	65.20%	64.54%
Hierarchical Transformers.	s-utts	68.36%	63.98%	64.74%	64.26%
Hierarchical Transformers.	s-uttturn	68.39%	63.82%	64.39%	64.05%
Hierarchical Transformers.	turn	67.16%	53.53%	50.95%	46.17%
BERT + Dense layer	no context	69.09%	63.37%	51.78%	45.60%
BERT + Transformers	no context	70.41%	66.23%	**67.10%** **	**66.57%** *
text-pair BERT + Transformers	1prev	68.66%	64.63%	**65.70%** *	64.97%
text-pair BERT + Transformers	2prev	66.14%	62.77%	64.18%	62.98%
text-pair BERT + Transformers	3prev	64.91%	61.38%	62.60%	61.55%
text-pair BERT + Transformers	s-utt	71.34%	66.97%	**67.46%** **	**67.19%** *
text-pair BERT + Transformers	s-utts	71.61%	66.84%	**66.44%** *	**66.63%** *
text-pair BERT + Transformers	s-uttturn	66.50%	62.17%	62.98%	62.42%
text-pair BERT + Transformers	turn	68.76%	64.39%	65.14%	64.67%

Table 2: Baselines, Hierarchical Transformers and text-pair BERT + Transformers models' performance in the different conditions (see section 3). In bold the significant results against SVM, with $^{**} : p \leq 0.01; \quad ^{*} : p \leq 0.05$

6.3 Contextualized models

The contextualized models show similar trends within the Transformer- and the BERT- based models. They are more evident and result in higher performance in the BERT models but are visible in the Hierarchical Transformers as well.

None of the Hierarchical Transformers shows an F-measure better than that of the non-hierarchical Transformer model, and they are better than the SVM baseline, but not significantly. We also see that the performance slowly degrades when the context is expanded from one to three utterances, regardless of the speaker of those utterances (green histogram in table 2). The same consideration holds for the subject's previous utterance, all their previous utterances, these utterances plus the previous turn, or the previous turn alone. In this last case, the fall of performance is remarkable. The model struggles to recognize the false utterances, and the recall is around 50%.

The BERT-based models confirm the loss of performance with context from 1 to 3 utterances, regardless of the speaker. In this case, the F-measure slope in the three conditions is even more pronounced than in the case of the Hierarchical Transformers.

The best results come from the two models, which rely on the contexts where only the interviewee's utterances are considered. These models are significantly better than SVM in terms of F-measure, and they have the highest performance even in terms of precision and accuracy. The best model is even significantly better than the one that uses convolutions, both for F1 and for recall, with $p < .05$.

In the conditions where another speaker's previous turn is included in the models, the performance worsens, similarly to the Hierarchical Transformers models tested in the same conditions.

7 The language of deception

We adopt two methods to depict the deceptive language: 1) we compute the Information Gain (IG) of word n-grams (Forman, 2003), and 2) we apply the Sampling and Occlusion (SOC) algorithm (Jin et al., 2019).

Information Gain measures the entropy of (sequences of) terms between the different classes. The more imbalanced the presence of such terms for one label class at the other's expense, the higher the IG value. Table 3 shows the tri-grams with the highest IG values, divided according to the class of which they are indicative, i.e., where they are more frequently found. While we computed the IG score from uni-grams to $penta$-grams, we show only tri-grams that, for illustration, represent the best trade-off between meaningful and frequent chunks of text.

These n-grams show that deceptive statements abound with negations: mostly of not remembering, but also not knowing and not having done. In contrast, truthful statements tend to be more assertive and focused on concrete details of time and circumstances. The IG signal's strength also suggests that sincere expressions are much more varied than deceptive ones, which are repeated more often and seem to be particularly stereotyped.

Even though the patterns detected by the neural models are not necessarily interpretable in terms of human common sense, we also use SOC to highlight the words that the models find to be the most influential for their output.

SOC gives a *post-hoc* explanation of the weight of specific words in a sentence for the classification task by considering the prediction difference after replacing each word with a MASK token (Jin et al., 2019). Since the outcomes depend on the context words, but Jin et al. (2019) are interested in the single words' relevance, they do not use the whole context but sample words from it. In this way, they reduce the context's weight, emphasizing that of the word itself.

Figure 2 shows two examples of correctly classified sentences, one deceptive and one truthful. The model interprets the red words as indicative of deception, the blue ones of truthfulness. They are coherent with the intuition provided by the IG. However, they cannot be interpreted as representative of our most complex models' inner functioning, as SOC relies on a standard BERT-based classifier.

8 Discussion

Our results show that the Transformers-based models, in the hierarchical and non-hierarchical form, obtain good results in the classification task. Even the non-hierarchical model is significantly better than the previous state-of-the-art.

However, the BERT-based models are those that show the best and the worst results. The worst ones come from the BERT for single-text and a simple dense output layer. On the other hand, when the fully connected layer is substituted by multi-layer, multi-head Transformers, while the BERT output is the same, the performance improves substantially (non-contextual models, red histograms in table 2).

We also ran experiments with text-pair BERT + Dense layer. We do not report the details since they do not add to the results: performance is low, while text-pair BERT with Transformers gives the best outcomes (blue histograms).

These results suggest that:

1. BERT does not embody the knowledge necessary for detecting deception. The input representations of a single fully connected layer are not expressive enough to cope with the task's complexity. This makes sense: BERT is not trained on texts and on a task (to predict the masked words) to train it to recognize deception. The cues of deception are essentially stylometric (section 7) and need a dedicated neural architecture to learn them. This is just the case of the Transformers that we associate with BERT. Thanks to their positional embeddings, they can identify the texts' relevant parts, which the task requires. This aspect also explains the SVM's performance based on n-grams and CNNs. Its convolutional layers essentially explore patterns in the n-gram embeddings.

2. When it is combined with architectures that detect deception cues, such as the Transformers, BERT's knowledge becomes an added value that allows the models to reach the best performance. Therefore, the key to success is to combine the power of transfer learning models that bring a robust semantic knowledge base and attention mechanisms to explore sequences, detecting patterns more complex than those identified by simple, fully connected layers.

3. On the other hand, when the contextual knowledge in BERT embeddings is missing, we see an over-estimation of the stylometric features coming from the context. For example, in the Hierarchical Transformers case, the models rely only on the texts' information, which prevents the hierarchical models from outperforming the non-hierarchical ones. Therefore,

True *tri*-gram	Translation	IG*100	False *tri*-gram	Translation	IG*100
in_quel_periodo	at that time	3.245	non_ricordo_.	I don't remember.	21.858
non_ho_capito	I don't understand	2.884	non_lo_so	I don't know	10.831
è_vero_che	it is true that	2.884	non_l'_ho	I didn't	09.257
mi_sembra_che	it seems to me that	2.884	non_mi_ricordo	I didn't remember	08.674
tant'_è_vero	so much so that	2.523	non_posso_dire	I cannot say	07.789
in_carcere_,	in prison,	2.162	il_mio_amico	my friend.	07.627
c'_è_la	there is the	2.162	io_l'_ho	I did.	06.843
e_niente_,	ultimately,	2.162	lo_ricordo_.	...remember it.	06.677
ho_capito_.	I understand.	2.162	mi_ricordo_proprio	I just remember	06.674
di_sì_.	(I think) so.	2.162	l'_ho_allontanato	I pushed him away	06.674

Table 3: Information Gain (rescaled by 100 to avoid tiny values) of *tri*-grams indicative of truth (left) and deception (right)

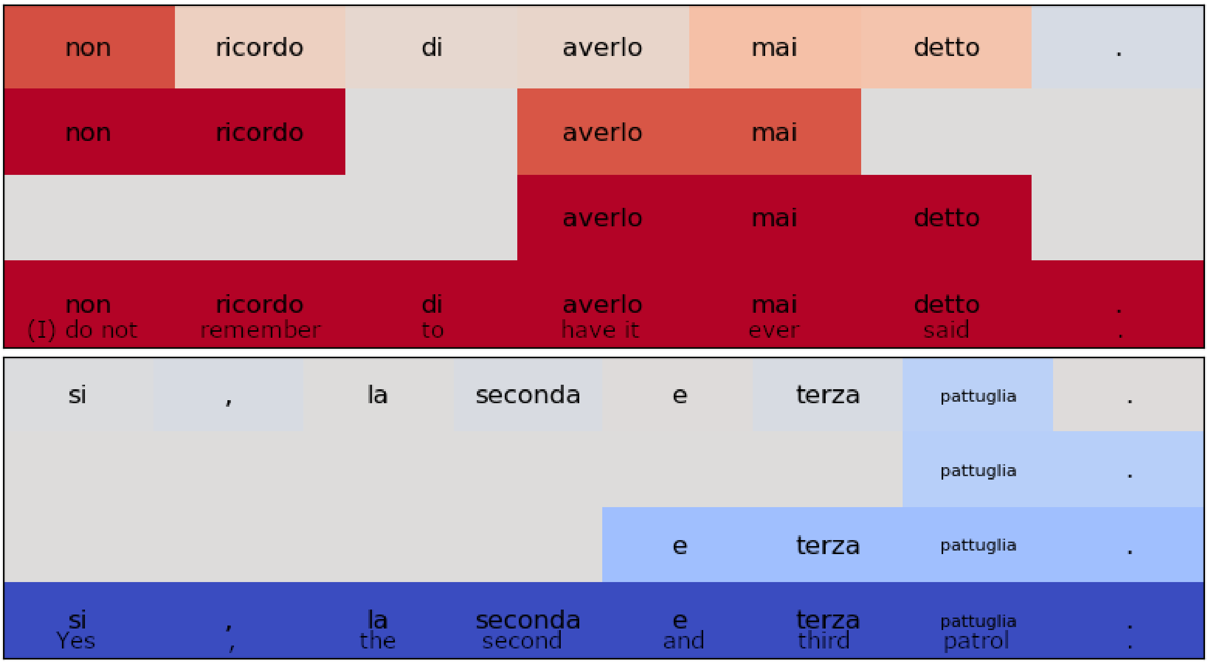


Figure 2: Output of the SOC algorithm. The red terms predict deception, the blue ones predict truthfulness.

we speculate that BERT's contextual knowledge works as a regularizer, which provides the Transformer with previously weighted inputs, according to the sentences' meaning.

Our results concerning BERT's usefulness with context are different from those obtained by Peskov et al. (2020), who work on Diplomacy board-game deception data. Their study associated BERT to LSTM-based contextual models, and they did not find a BERT contribution in their model's performance. They tried to fine-tune it, and they hypothesized that the lack of performance improvement was motivated by the "relatively small size" of the training data. This hypothesis could be correct, but our outcome allows us to formulate another hypothesis. Their data set concerns an online game, where the range of topics in the dialogues is presumably restricted and specific. This limitation would not allow BERT's broad knowledge to give a concrete contribution. In contrast, the data set we use comes from real life. The number of possible topics in Court is the widest. Under such conditions, it is reasonable that BERT's semantic information can play a much more relevant role: this gives a different intuition about the kind of use-cases where BERT can be useful.

Regarding the use of contexts to improve deception detection, it turns out that they can be useful, but they need to be carefully handled. In fact, not any context helps. It is not advisable to generically "collect something" before the target text. To select the previous sentence(s), regardless of the speaker, means to incorporate noise that is more harmful than helpful for the task.

Our best models are those that only consider the utterances of the speaker him/herself. Moreover, even in that case, the context's contribution improves according to its proximity to the target sentence. The overall performance model that only uses the speaker's first previous utterance is slightly better than that of the models considers all of them. This evidence is made even stronger by the observation that, in most cases, there is no previous speaker's utterance, as he/she responds with a single utterance to a statement or question of an interlocutor. To be precise, only 921 utterances of 3015 are preceded by another utterance by the same subject. So in more than two-thirds of the cases, the target utterance has no context from the same speaker and has to be considered standing alone, similarly to non-contextualized models. In other words, meaningful context is often absent but can contribute remarkably to reach better performance, which suggests that context is crucial for the task.

In other words, the fact that the additional information, even if present in less than one-third of the cases, is enough to outperform the other models and to reach the best results suggests that this is the way to obtain the best help from the context when present.

The loss of performance when the contexts include the previous turn is also coherent with the results with the contexts based on a given number of previous utterances: incorporating the statements/questions of the other persons does not help detect deception. If any, the right cues for detecting deception are in the target sentence itself or just nearby.

Also, the contextual information's usefulness is conditioned by using the right models. BERT and the trainable Transformers need to be used together. The attention mechanism that follows BERT is the trainable part of the network and detects the stylometric patterns of deception. However, we speculate that the BERT's contextual word representations act as a regularizer, which reduces the probability that the information from outside the target sentence, carried by non-contextual embeddings, is overestimated.

9 Related work

The first computational linguistics study on deception detection was Newman et al. (2003). They asked subjects to write truthful and deceptive essays and evaluated them using the Linguistic Enquiry and Word Count (LIWC), a lexicon that assigns texts several linguistic and psychological scores. LIWC is a popular tool in deception detection, also used in Fornaciari and Poesio (2013), which we compare to.

There are two main research lines: one relies on artificially produced data, often using crowdsourcing services, and the other focuses on data sets from real-life situations. The common bottleneck for data set creation is the availability of ground truth, i.e., knowing the truth behind a subject's statements. For this reason, many studies rely on data collected in laboratory conditions (Ott et al., 2011). While these studies allow us to gain intuitions about the deceptive language features, there are no real or relevant consequences for the liars. Their validity concerning high-stakes conditions is therefore unclear. Artificially created texts are likely not interchangeable with those from natural conditions (Fornaciari et al., 2020).

The notion of deception itself is used in a broad sense and includes studies that focus on a different kind of deception. A popular area, for example, concerns the detection of fake news (Oshikawa et al., 2018; Girgis et al., 2018) The field is expanding to include models that does not detect deceit strictly speaking, but trolls in social media (Addawood et al., 2019).

Pérez-Rosas et al. (2015) is more similar to our study. They collected videos from public court trials and built a multi-modal model that relies on verbal (unigrams and bigrams) and non-verbal features (Decision Trees (DT) and Random Forest (RF)). Krishnamurthy et al. (2018) used the same data set, but with neural models to represent video, audio, and textual features. In particular, they extracted verbal features relying on pre-trained word embeddings and Convolutional Neural Networks. They reached an accuracy of 96.14%. These studies are particularly interesting for the type of data set and the multi-modal approach. However, neither take the linguistic context of the statements into consideration.

Levitan et al. (2018) used the data set of Levitan et al. (2015), where 170 pairs of subjects play a "lying game". This study addresses deception in dialogues. I.e., the texts are structured as a sequence of turns, each containing one or more statements of a single participant. For the analysis, the authors selected several easily interpretable linguistic features, allowing the authors to draw a description of the deceptive language and feed a Random Forest classifier. This considers both single and multiple turns, finding that the last ones allowed to reach the best performance in their data set (F1-score of 72.33%). However, this is a laboratory experiment that is not a high-stakes scenario for the participants: this limits the possibilities of comparison with our study.

From a methodological point of view, our study is similar to that by Peskov et al. (2020). They collect data from an online negotiation game, where the participants' success depends on their ability to lie. They use state-of-the-art neural models, which also consider contextual information. However, subjects are not in a high-stakes condition in their study, so their findings are not directly comparable to our use case.

10 Conclusion

In this paper, we explore the performance of language models in detecting lies using a unique data set that contains sentences that come from real hearings created by Fornaciari and Poesio (2013) and anonymized for a research setting. We show that context is key to creating models that can detect deception and that BERT with some added attention layers can effectively beat different baselines.

However, there is no evidence that the deception cues derive from dialogic interaction, as the most useful contributions come from the speaker him/herself. To examine in depth this aspect is a line for future research.

11 Ethical statement and limitations

Applying predictive models in a legal and law enforcement context can be problematic, especially of the historical training data is biased towards certain groups (Angwin et al., 2016).

Therefore, we do not propose general-purpose models for deception detection. They only refer to the context of hearings in court, and they can be applied, at best, to similarly ruled events, for example texts coming from police interrogations. However, as statistical models, they do incorporate linguistic biases that are possibly present in the training data (Shah et al., 2020). This should be considered for a fair and respectful interpretation of the results.

It is also important to point out that the model predictions have no absolute certainty but are intrinsically probabilistic. As such, they are only meant to *support* investigations and to inform a judge's decisions. They cannot be a substitute for expert evaluations or for a due legal process.

References

Aseel Addawood, Adam Badawy, Kristina Lerman, and Emilio Ferrara. 2019. Linguistic cues to deception: Identifying political trolls on social media. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 15–25.

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias. *ProPublica, May*, 23.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. 1997. Support vector regression machines. In *Advances in neural information processing systems*, pages 155–161.

Eileen Fitzpatrick and Joan Bachenko. 2012. Building a data collection for deception research. In *Proceedings of the workshop on computational approaches to deception detection*, pages 31–38.

George Forman. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of machine learning research*, 3(Mar):1289–1305.

Tommaso Fornaciari, Leticia Cagnina, Paolo Rosso, and Massimo Poesio. 2020. Fake opinion detection: how similar are crowdsourced datasets to real data? *Language Resources and Evaluation*, pages 1–40.

Tommaso Fornaciari and Massimo Poesio. 2012. DeCour: a corpus of DEceptive statements in Italian COURts. In *Proceedings of the Eigth International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Tommaso Fornaciari and Massimo Poesio. 2013. Automatic deception detection in italian court cases. *Artificial intelligence and law*, 21(3):303–340.

Sherry Girgis, Eslam Amer, and Mahmoud Gadallah. 2018. Deep learning algorithms for detecting fake news in online text. In *2018 13th International Conference on Computer Engineering and Systems (ICCES)*, pages 93–97. IEEE.

Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. 2019. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. *arXiv preprint arXiv:1911.06194*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Gangeshwar Krishnamurthy, Navonil Majumder, Soujanya Poria, and Erik Cambria. 2018. A deep learning approach for multimodal deception detection. *arXiv preprint arXiv:1803.00344*.

Sarah I Levitan, Guzhen An, Mandi Wang, Gideon Mendels, Julia Hirschberg, Michelle Levine, and Andrew Rosenberg. 2015. Cross-cultural production and detection of deception from speech. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*, pages 1–8.

Sarah Ita Levitan, Angel Maredia, and Julia Hirschberg. 2018. Linguistic cues to deception and perceived deception in interview dialogues. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1941–1950.

Matthew L Newman, James W Pennebaker, Diane S Berry, and Jane M Richards. 2003. Lying words: Predicting deception from linguistic styles. *Personality and social psychology bulletin*, 29(5):665–675.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. *arXiv preprint arXiv:1107.4557*.

James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001.

Verónica Pérez-Rosas, Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. 2015. Deception detection using real-life trial data. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 59–66.

Denis Peskov, Benny Cheng, Ahmed Elgohary, Joe Barrow, Cristian Danescu-Niculescu-Mizil, and Jordan Boyd-Graber. 2020. It takes two to lie: One to lie and one to listen. In *Association for Computational Linguistics*.

Deven Santosh Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264, Online. Association for Computational Linguistics.

Anders Søgaard, Anders Johannsen, Barbara Plank, Dirk Hovy, and Héctor Martínez Alonso. 2014. What's in a p-value in nlp? In *Proceedings of the eighteenth conference on computational natural language learning*, pages 1–10.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Learning Coupled Policies for Simultaneous Machine Translation using Imitation Learning

Philip Arthur[†]
Department of
Data Science and AI
Monash University

Trevor Cohn[‡]
School of Computing and
Information Systems
University of Melbourne

Gholamreza Haffari[†]
Department of
Data Science and AI
Monash University

[†]{philip.arthur, gholamreza.haffari}@monash.edu
[‡]tcohn@unimelb.edu.au

Abstract

We present a novel approach to efficiently learn a simultaneous translation model with coupled programmer-interpreter policies. First, we present an algorithmic oracle to produce oracle READ/WRITE actions for training bilingual sentence-pairs using the notion of word alignments. This oracle actions are designed to capture enough information from the partial input before writing the output. Next, we perform a coupled scheduled sampling to effectively mitigate the exposure bias when learning both policies jointly with imitation learning. Experiments on six language-pairs show our method outperforms strong baselines in terms of translation quality while keeping the translation delay low.

1 Introduction

Simultaneous machine translation (SIMT) is a setting where the translator needs to incrementally generate the translation while the source utterance is being received. This is a challenging translation scenario as the SIMT model needs to trade off delaying translation output and the quality of the generated translation.

Recent research on SIMT relies on a strategy to decide when to read a word from the input or write a word to the output (Satija and Pineau, 2016; Gu et al., 2017). This is based on a sequential decision making formulation of SIMT, where the decision making about the next READ/WRITE action is made by an *agent*, interacting with the neural machine translation (NMT) *environment*. Current approaches are sub-optimal as they either fix the agent's policy to focus learning the NMT model (Ma et al., 2019; Dalvi et al., 2018) or learn adaptive agent policies while the NMT model is fixed (Gu et al., 2017; Alinejad et al., 2018). We argue that the interpreter should also learn to generate correct translation from incomplete input information. This is challenging as we need to optimize both programmer's and interpreter's policies to balance the tradeoff between quality and delay in the reward.

Previous research has considered the use of imitation learning (IL) to train the agent's policy (Zheng et al., 2019a,b), which is generally superior to reinforcement Learning (RL) in terms of the stability and sample complexity. However, the bottleneck of IL in SIMT is the *un*availability of the *oracle* sequence of actions. Designing *algorithmic* oracles to compute sequence of READ/WRITE actions with low translation latency and high translation quality is under-explored.

We present an IL approach to efficiently learn effective coupled programmer-interpreter policies in SIMT, based on the following contributions. First, we present a simple, fast, and effective algorithmic oracle to produce oracle actions from the training bilingual sentence-pairs based on statistical word alignments (Brown et al., 1993). Next, we design a framework that uses scheduled sampling on both programmer and interpreter. This is different from the typical IL scenarios, where there is only one policy to learn. As the two policies *collaborate*, their learning needs to be robust not only to their own incorrect predictions, but also to incorrect predictions of the other policy to mitigate this coupled *exposure bias*.

Experiments on six language pairs (translating to English from Arabic, Czech, German, Romanian, Hungarian, and Bulgarian) show the policies trained using our approach compares favorably with strong policies from the previous work. We attribute the effectiveness of the learned coupled policies to (i) the scheduled sampling, which handles the coupled exposure bias, resulting in up to 5-8 BLEU score improvements, and (ii) the quality of oracle actions generated by our algorithmic oracle, which balances translation quality and delay.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2709–2719
April 19 - 23, 2021.

Algorithm 1 Generation in NPI-SiMT

```
$i, j \leftarrow 0$
while a stopping condition is not met do
    $t \leftarrow i + j$
    $\boldsymbol{s}_{t+1} \leftarrow \boldsymbol{f}_{\text{prog}}(\boldsymbol{s}_t, [a_t, \boldsymbol{g}_j, \boldsymbol{h}_i])$
    $P_{\text{prog}} \leftarrow \text{softmax}(\text{dense}_{\text{prog}}(\boldsymbol{s}_{t+1}))$
    $a_{t+1} \sim P_{\text{prog}}$
    if $a_{t+1} = \text{READ}$ then
        $i \leftarrow i + 1$
        $\boldsymbol{h}_i \leftarrow \boldsymbol{f}_{\text{enc}}(\boldsymbol{h}_{i-1}, x_i)$
    else
        $j \leftarrow j + 1$
        $\boldsymbol{g}_j \leftarrow \boldsymbol{f}_{\text{intp}}(\boldsymbol{g}_{j-1}, y_{j-1}, \boldsymbol{h}_{\leq i})$
        $P_{\text{intp}} \leftarrow \text{softmax}(\text{dense}_{\text{intp}}(\boldsymbol{g}_j))$
        $y_j \sim P_{\text{intp}}$
    end if
end while
```

2 NPI Approach to SiMT

We describe generation in our neural programmer-interpreter (NPI) approach to simultaneous machine translation (SiMT) in Algorithm 1. At each time step t, the *programmer* needs to decide whether to READ the next source word or to WRITE the next target word in the translation. The *interpreter* then immediately *executes* the action generated by the programmer. Both programmer and interpreter are modeled using Markov Decision Process (MDP) where prediction at particular timestep depends on the history of previous predictions. The indices i_t and j_t are the number of READ and WRITE actions in the program up to time step t.

The Programmer needs to *sequentially* decide about the next *action*, given the previous actions $\boldsymbol{a}_{<t}$ and the prefix of the source utterance read so far $\boldsymbol{x}_{\leq i_t}$ as well as the prefix of the target translation generated so far $\boldsymbol{y}_{\leq j_t}$. That is, our programmer is modeled as $P_{\text{prog}}(a|\boldsymbol{a}_{<t}, \boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t})$.

The Interpreter needs to execute the action generated by the programmer. At time step t, if the generated action a_t is READ, we reveal the next input token. Otherwise, if WRITE, then we generate the next target word according to $P_{\text{intp}}(y|\boldsymbol{a}_{\leq t}, \boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t})$.[1]

The Probabilistic Model. The probability of simultaneously generating the translation $\boldsymbol{y}$ and the sequence of actions $\boldsymbol{a}$ for a source utterance $\boldsymbol{x}$ is,

$$P_{\text{SiMT}}(\boldsymbol{y}, \boldsymbol{a}|\boldsymbol{x}) = \prod_{t=1}^{|\boldsymbol{x}|+|\boldsymbol{y}|} P_{\text{prog}}(a|\boldsymbol{a}_{<t}, \boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t}) \times \prod_{t:a_t=\text{WRITE}} P_{\text{intp}}(y|\boldsymbol{a}_{\leq t}, \boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t}).$$

Training the Model. In SiMT, we are interested in not only producing a high quality translation, but also reducing the delay between the times of receiving the source words and generating their translations. Training of the model based on this hybrid training objective can be done by reinforcement learning (RL) or imitation learning (IL). The RL approach has been attempted by (Satija and Pineau, 2016; Gu et al., 2017; Alinejad et al., 2018) for training the programmer; however, it is unstable due to sparsity of the reward function and these works also assumed a fixed interpreter. We thus take the IL approach for a sample efficient, effective, and stable learning of policies in NPI-SiMT.

3 Deep Coupled Imitation Learning

Our goal is to learn a pair of policies for the programmer and interpreter using IL. §3.1 describes the method of learning of both policies where their learning inter-dependency needs to be taken into account. §3.2 describes our novel *oracle* program actions for each sentence pair in the training set, i.e., the program $\boldsymbol{a}$ which has been responsible for generating the translation $\boldsymbol{y}$ for a source utterance $\boldsymbol{x}$ with as low delay as possible. Our overall training algorithm is depicted in Algorithm 2. $\hat{\boldsymbol{X}} = [\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_{|\boldsymbol{x}|}]$ is the encoding of the input sequence and $\hat{\boldsymbol{Y}} = [\hat{\boldsymbol{y}}_1, ..., \hat{\boldsymbol{y}}_{|\boldsymbol{y}|}]$ is a list of interpreter hidden states for all predictions. During training, these values are calculated before calculating the loss of the programmer.

3.1 Learning Robust Coupled Policies

Assuming we have the oracle actions, we can learn the policies for both the programmer and interpreter using behavioural cloning in IL (Torabi et al., 2019). That is, the model parameters are learned by maximising the likelihood of the oracle actions for both the programmer and interpreter,

$$\theta^*_{\text{prog}}, \theta^*_{\text{intp}} := \arg\max_{\theta_{\text{prog}}, \theta_{\text{intp}}} \sum_{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a})} \sum_{t=1}^{|\boldsymbol{x}|+|\boldsymbol{y}|} \log P_{\text{prog}}(a|\boldsymbol{a}_{<t}, \boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t}; \theta_{\text{prog}}) + \sum_{t:a_t=\text{WRITE}} \log P_{\text{intp}}(y|\boldsymbol{x}_{\leq i_t}, \boldsymbol{y}_{\leq j_t}; \theta_{\text{intp}}).$$

---

[1] The counter i and j are also incremented according to the respective actions at time t.

This is akin to have the expectation, in the original training objective of NPI, under a point-mass distribution over the oracle actions.

IL with behavioural cloning does not lead to robust policies for unseen examples in the test time due to exposure bias (Bengio et al., 2015). That is, the agent is only exposed to situations resulting from the *correct* actions in the training time, leading to its inability to mitigate from propagation of errors faced due to incorrect actions in the test time. Scheduled sampling (Bengio et al., 2015; Ross et al., 2011) addresses this issue by exposing the agent to incorrect decisions in training time through perturbation of the oracle decisions, which we extend to learning policy *pairs*. Crucially, the programmer-interpreter policies need to be robust to incorrect decisions encountered not only in their own trajectories, but also to one anothers' trajectories.

Learning the Programmer. To train our programmer on a training example $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a})$ with scheduled sampling, we first create the perturbation $(\boldsymbol{a}', \boldsymbol{y}')$ of the ground truth program and interpreter decisions. The perturbed program $\boldsymbol{a}'$ and translation $\boldsymbol{y}'$ are only used as the input to the recurrent architectures of the programmer and interpreter's decoder. They are created by replacing some of the ground truth element by randomly selecting an action from the predictive distribution of each model. We then maximise the following training objective,

$$\theta^*_{\text{prog}} := \arg\max_{\theta_{\text{prog}}} \sum_{(\boldsymbol{x},\boldsymbol{y}',\boldsymbol{a},\boldsymbol{a}',\boldsymbol{a}'')} \sum_{t=1}^{|\boldsymbol{x}|+|\boldsymbol{y}'|} \log P_{\text{prog}}(a|\boldsymbol{a}'_{<t}, \boldsymbol{x}_{\le i_t}, \boldsymbol{y}'_{\le j_t}; \theta_{\text{prog}}). \quad (1)$$

Based on the generative process described in Algorithm 1, the programmer conditions the generation of actions in each time step on the current states of the NMT's encoder and decoder. Hence, while training the programmer, the valid READ/WRITE actions need to be communicated to the interpreter and be executed in order to provide NMT's encoder/decoder states to the programmer to condition upon. Crucially, the communicated program needs to be valid.

Valid Program A ground truth program $\boldsymbol{a}$ is a valid sequence of READ/WRITE actions if $|\text{READ} \in \boldsymbol{a}| = |\boldsymbol{x}|$ and $|\text{WRITE} \in \boldsymbol{a}| = |\boldsymbol{y}|$. This valid program ensures the NPI model to safely consume a pair of parallel sentence. We generate a valid perturbation $\boldsymbol{a}''$ by only permuting the READ/WRITE actions of the program $\boldsymbol{a}$ (Figure 1).

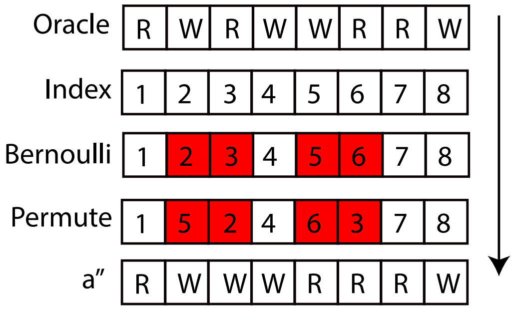


Figure 1: Creating valid perturbation from oracle program. We use a combination of Bernoulli sample and permutation function.

Algorithm 2 Training NPI-SiMT

```
Require: $\mathcal{D}$: Sentence pairs with oracle actions,
    $\beta_1, \beta_2, \beta_3$ : scheduled sampling probabilities
    for $\boldsymbol{y}', \boldsymbol{a}', \boldsymbol{a}''$.
while a stopping condition is not met do
    randomly pick $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}) \in \mathcal{D}$
    $\boldsymbol{y}' \leftarrow$ perturbSeq$(\boldsymbol{y}, \beta_1, \theta_{\text{intp}})$
    $\boldsymbol{a}' \leftarrow$ perturbSeq$(\boldsymbol{a}, \beta_2, \theta_{\text{prog}})$
    $\boldsymbol{a}'' \leftarrow$ perturbProgValid$(\boldsymbol{a}, \beta_3)$
    $\hat{\boldsymbol{y}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{Y}} \leftarrow$ forward_intp$(\theta_{\text{intp}}, \boldsymbol{x}, \boldsymbol{y}', \boldsymbol{a}'')$
    $\hat{\boldsymbol{a}} \leftarrow$ forward_prog$(\theta_{\text{prog}}, \boldsymbol{a}', \hat{\boldsymbol{X}}, \hat{\boldsymbol{Y}})$
    $\theta_{\text{intp}} \leftarrow \theta_{\text{intp}} - \alpha_1 \nabla \delta(\hat{\boldsymbol{y}}, \boldsymbol{y})$
    $\theta_{\text{prog}} \leftarrow \theta_{\text{prog}} - \alpha_2 \nabla \delta(\hat{\boldsymbol{a}}, \boldsymbol{a})$
end while
```

We further extend the definition of a valid program with respect to the domain knowledge of translation so that: (i) no WRITE at the beginning, and (ii) no READ at the end of the program.

Learning the Interpreter. The interpreter needs to be robust to the incorrect actions in the previously generated words in the translations as well as the READ/WRITE actions generated by the programmer. This is done by communicating $\boldsymbol{a}''$ to the intepreter during training. Thus, the training objective for the interpreter is,

$$\theta^*_{\text{intp}} := \arg\max_{\theta_{\text{intp}}} \sum_{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{y}',\boldsymbol{a}'')} \sum_{t:a''_t=\text{WRITE}} \log P_{\text{intp}}(y|\boldsymbol{x}_{\le i_t}, \boldsymbol{y}'_{\le j_t}; \theta_{\text{intp}}). \quad (2)$$

3.2 Oracle Program Actions

Our proposed oracle should measure the appropriate amount of inputs needed for translating a particular target word $\boldsymbol{y}_{j_t}$. This is done by determining the *key* word or phrases δ_j which contain important information of $\boldsymbol{y}_{j_t}$, and therefore guiding the programmer to read until δ_j before writing $\boldsymbol{y}_{j_t}$. Algorithm 3 outlines our oracle generation

Algorithm 3 Oracle Generation

```
Require: $\boldsymbol{a}$: Symmetrized alignment of $\boldsymbol{x}$ and $\boldsymbol{y}$
    in forms of $\boldsymbol{a}_{i,j}$, which means that $\boldsymbol{x}_i$ is aligned
    to $\boldsymbol{y}_j$. Index starts with 0.
$\delta_{\text{READ}} := -1$
for $j \in$ range$(0, |\boldsymbol{y}|)$ do
    $\delta_j \leftarrow \max[\{i \in a_{i,j}\}]$
    for $(\delta_j - \delta_{\text{READ}})$ times do
        emit READ
    end for
    $\delta_{\text{READ}} \leftarrow \max(\delta_j, \delta_{\text{READ}})$
    emit WRITE
end for
```

procedure. No READ operation is emitted if the $\delta_j \leq \delta_{\text{READ}}$ or $\{i \in a_{i,j}\} = \emptyset$.

δ_j can be heuristically determined by using word alignment (Brown et al., 1993; Koehn et al., 2003), which captures strong relationship between tokens. In the case of many to one alignment of source to target, we choose the furthest source word. In the case of no alignment, nothing is done as it means the target word can be induced merely from the decoder without needing to read additional inputs. This oracle can generally generate a valid program. Caution is needed to ensure that no WRITE at the beginning and no READ at the end of the generated oracle. This can be done by aligning the first words of the parallel sentence. Similarly we also need to align the last words of the parallel sentence. [2]

4 Experiments

Our experiments aim to measure the effectiveness of our proposed method versus a strong wait-k baseline, over a range of languages of varying difficulty, syntactic complexity, and lexical complexity.

4.1 Settings

Datasets. Our main experiment will be performed in higher quality corpus which is designed for spoken dialogue and carefully edited dataset. Additionally, we perform a single large scale experiment using crawled corpus such as WMT to show that our method also scales to a large dataset.

We evaluate our proposed method on 6 language pairs, in all cases translating into English, with the source languages chosen to cover a wide range of language families and syntax. We use German (DE), Czech (CS) and Arabic (AR) from the IWSLT 2016 translation dataset (Cettolo et al., 2012). We use the provided training and development sets as-is, and concatenate all provided test sets to create our test set. We also evaluate Hungarian (HR), Bulgarian (BG), and Romanian (RO) from the SETIMES corpus (Tyers and Alperen, 2010). As this corpus is not partitioned, we use the majority of the data for training, holding out 2000 random sentence pairs for development and another 2000 sentence pairs for testing. Together these languages are representative of Germanic, West Slavic, Arabic, Uralic, East Slavic, and Italic language families, respectively.

We use sentencepiece (Kudo and Richardson, 2018) to build and tokenize our training data with 16k vocabulary size. Then we generate our oracle program actions based on the segmented tokens. We use fast_align (Dyer et al., 2013) to generate symmetrized alignments between tokens. Unless otherwise specified, we use the default settings of the mentioned toolkit.

Evaluation. We evaluate the SIMT systems based on its translation quality and delay. Translation quality can be measured by case sensitive BLEU (Papineni et al., 2002).[3] We adopt three delay measurements by previous studies. First, average proportion (AP) (Gu et al., 2017) is a fraction of read source words per emited target words. Second, average lagging (AL) (Ma et al., 2019) is an average number of lagged source words until all inputs are read. Finally the differentiable-AL (DAL) (Arivazhagan et al., 2019) is a refinement of AL which also accumulates the cost of writing output tokens after inputs are fully read.

Baseline. We compare against the wait-k baseline (Ma et al., 2019) where the programmer's policy begins with k numbers of READ, and is followed by switching WRITE and READ, until the source sentence is exhausted or end of sentence (EOS) symbol is written. If the source sentence is exhausted, the programmer will only emit WRITE actions. This baseline was shown to be superior compared to the reinforcement learning approach (Zheng et al., 2019a), and k can be tuned for the desired delay. Arivazhagan et al. (2019)'s approach is superior than the wait-k baseline. However there

---

[2] Our oracle algorithm's code is released in `https://github.com/Monash-NLP-ML-Group/arthur-eacl2021`.

[3] Calculated using sacrebleu (Post, 2018). BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.4 is our `sacreblue`'s signature

	DE→EN				CS→EN				AR→EN				HR→EN				BG→EN				RO→EN			
	BLEU	DAL	AL	AP	BLEU	DAL	AL	AP	BLEU	DAL	AL	AP	BLEU	DAL	AL	AP	BLEU	DAL	AL	AP	BLEU	DAL	AL	AP
wait-k																								
$k = 1$	14.40	3.74	2.22	0.61	11.87	4.05	2.60	0.63	16.08	4.04	2.88	0.64	24.78	3.72	2.15	0.58	22.02	3.58	1.58	0.56	22.76	3.51	1.43	0.55
$k = 2$	18.67	4.01	2.93	0.64	15.90	4.21	3.20	0.66	19.46	4.35	3.44	0.68	28.50	3.99	2.80	0.60	25.37	3.99	2.27	0.59	28.31	3.82	2.13	0.58
$k = 3$	21.13	4.59	3.71	0.68	17.27	4.82	3.92	0.70	22.11	4.91	4.21	0.72	32.04	4.55	3.52	0.63	26.79	4.90	3.35	0.62	32.13	4.31	2.88	0.61
$k = 4$	24.21	5.03	4.32	0.71	18.54	5.49	4.72	0.74	22.60	5.67	5.04	0.75	35.16	5.20	4.36	0.66	31.42	5.37	4.10	0.65	34.74	5.12	3.83	0.64
$k = 5$	25.63	5.81	5.19	0.75	19.77	6.23	5.59	0.77	23.22	6.50	5.90	0.79	37.42	5.91	5.12	0.69	34.83	5.85	4.73	0.67	38.70	5.63	4.37	0.66
$k = 6$	26.19	6.68	6.11	0.78	20.19	7.00	6.41	0.80	23.09	7.47	6.90	0.82	38.41	6.90	6.14	0.72	37.26	6.64	5.56	0.70	39.90	6.63	5.59	0.69
$k = 7$	26.89	7.51	7.01	0.81	20.12	7.95	7.33	0.83	23.51	8.20	7.67	0.84	39.77	7.69	6.97	0.74	38.42	7.53	6.54	0.72	40.35	7.54	6.46	0.72
$k = \infty$	28.05	21.67	21.67	1.00	20.85	20.91	20.91	1.00	23.54	19.98	19.98	1.00	41.86	27.56	27.56	1.00	41.98	29.06	29.06	1.00	46.22	29.88	29.88	1.00
NPI-SIMT	17.53	5.05	1.96	0.57	13.58	3.23	1.16	0.55	15.78	3.65	1.34	0.57	25.72	4.30	1.82	0.55	25.42	5.75	2.61	0.57	24.60	5.81	2.46	0.57
$+\boldsymbol{a}'$	20.89	3.98	1.65	0.56	16.17	3.02	1.22	0.55	16.12	2.88	1.14	0.56	30.66	4.05	1.83	0.55	30.73	4.25	1.96	0.55	30.47	5.07	2.25	0.56
$+\boldsymbol{a}''$	20.83	5.14	2.04	0.58	16.96	4.07	1.56	0.57	17.18	3.44	0.96	0.56	33.45	5.31	2.35	0.58	32.97	6.80	3.25	0.60	34.05	7.45	3.43	0.61
$+\boldsymbol{a}', \boldsymbol{a}''$	**22.37**	**4.11**	**1.83**	**0.57**	**18.69**	**3.32**	**1.43**	**0.56**	19.33	3.20	1.31	0.57	**33.69**	**4.23**	**2.00**	**0.56**	**33.78**	**4.69**	**2.18**	**0.56**	**35.97**	**5.32**	**2.55**	**0.58**
$+\boldsymbol{a}', \boldsymbol{a}'', \boldsymbol{y}'$	**22.38**	**4.04**	**1.80**	**0.57**	**18.97**	**3.24**	**1.32**	**0.56**	**20.47**	**2.89**	**1.15**	**0.55**	**35.38**	**3.96**	**1.84**	**0.56**	**34.67**	**4.72**	**2.26**	**0.56**	**37.92**	**4.98**	**2.38**	**0.57**
Oracle-at-test	30.53	3.66	1.72	0.57	23.17	3.17	1.50	0.56	25.63	3.10	1.49	0.57	44.98	3.84	1.84	0.55	46.68	3.98	1.92	0.56	50.31	4.32	2.10	0.56

Table 1: Full results on IWSLT and SETIMES datasets. Boldface indicates better translation quality versus wait-k are about the same delay (relevant systems indicated using underline within same column). Oracle-at-test is the system where the correct program is given during testing, serving as an upper bound on translation quality at a given delay.

is currently no open source code available and their end-to-end approach is not using an oracle policy. As our goal is not to beat the state-of-the-art, we leave this comparison as a future work.

NPI-SIMT. Both the programmer and interpreter are modelled using a unidirectional recurrent neural network (RNN) with a long short term memory cell (LSTM). In particular, we follow the architecture of Luong et al. (2015) with the multilayer perceptron attention of Bahdanau et al. (2015). Both the programmer and interpreter employ 20% dropout to the network output and 10% dropout to the embedding vector, and use a single layered LSTM with 512 hidden units. For the large scale experiment, we are using the transformer architecture (described in §4.5).

Training. We use `Adam` optimizer (Kingma and Ba, 2015) to train this framework. We track the learning rate of programmer and interpreter separately. We start with 0.001 learning rate, and start halving it whenever perplexity increase on development set. We use a fixed perturbation probability of 5%, 15%, and 15% for $\boldsymbol{y}'$, $\boldsymbol{a}'$, and $\boldsymbol{a}''$ respectively. Early stopping is executed at the fourth learning rate decay.

Testing. We use a beam search algorithm with a beam size of 5 and length normalization algorithm that divides hypothesis score by its length during search (Murray and Chiang, 2018).

4.2 Empirical Results

Scheduled Sampling. Our first experiment tests the effect of scheduled sampling (SS) in learning coupled policies in our NPI-SIMT method. For this purpose, we train four versions of our models where apply SS to both the programmer and the interpreter, only programmer, only interpreter, or neither. Table 1 shows the results, comparing against policies trained using the baseline wait-k method where $k \in \{1, 2, \ldots, 7, \infty\}$. The NPI-SIMT system that is trained using our proposed oracle (NPI SIMT) is able to learn from the low delay oracle as their natural delays (DAL, AL, AP) are generally as low as the delay of the oracle during training. However, it is clearly difficult to perfectly predict the oracle during test-time, and these programmer prediction errors resulted in mistakes in interpreter decisions.

Next, we consider the effect of perturbation and scheduled sampling on the proposed method. Table 1 shows that applying valid perturbation (a'') is more important than doing normal scheduled sampling (a'). This perturbation is directly correlated with the training of the interpreter, as such the noisy program make the interpreter resilient to the exposure bias. Applying both scheduled sampling further increased our proposed method accuracy. The schedule sampling on programmer ($+a', a''$) during training ameliorate this; as it increased up to 10 points of BLEU score in case of Romanian and Hungarian, 8 points in case of Bulgarian and, 5 points for German, Czech and Arabic. Additionally applying scheduled sampling on the interpreter ($+y'$) further improves translation accuracy while slightly decreasing the delay, both effects being consistent across all language pairs.

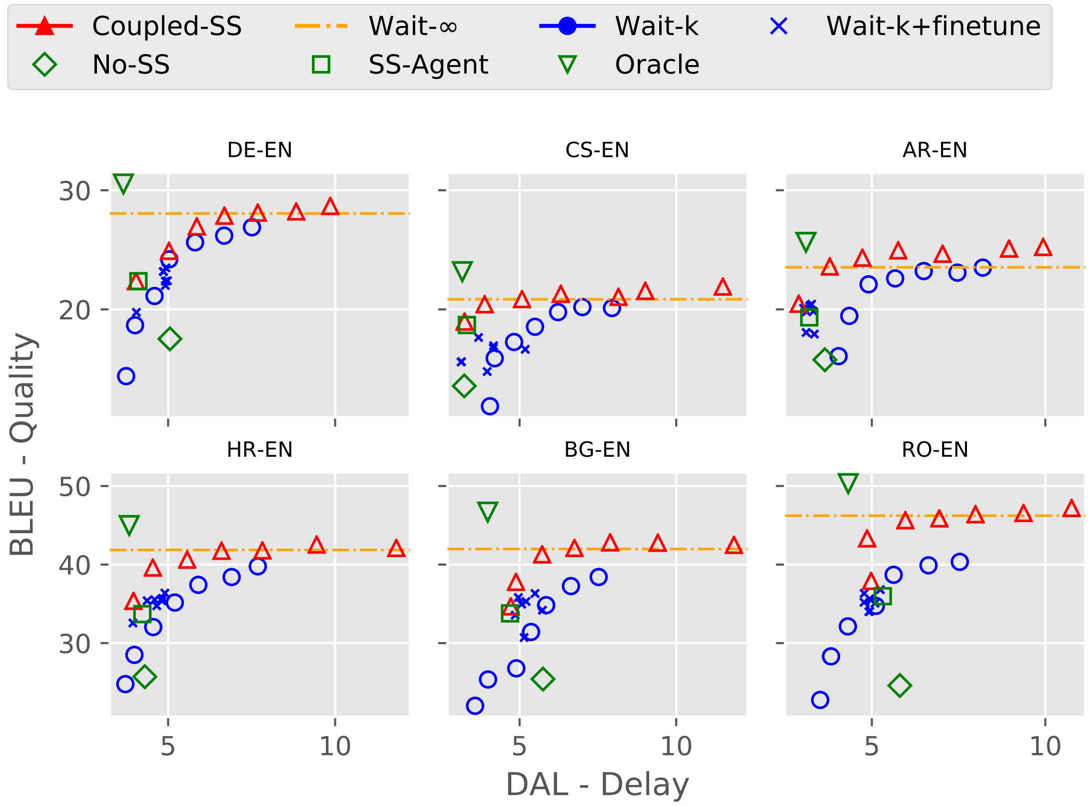


Figure 2: BLEU score versus delay using added delay and finetuning. Our proposed method coupled scheduled-sampling (Coupled-SS) method performs better than the wait-k baseline in all settings. The leftmost △ is the result of our proposed method without added delay, while △s to the right include delay (see description in text). The ◇ and □ are an ablation study considering not doing SS, and doing SS only on the agent, respectively. The × report finetuning various pretrained wait-k models with oracle+SS.

Oracle Policy vs Wait-k Policy. Figure 2 compares the policies trained by our algorithmic oracle vs those trained using the wait-k policy starting from $k = 1$. In each of these six plots, the policy trained using the oracle actions corresponds to the leftmost triangle point on the figure. Observe that the policy trained using the oracle actions compares favorably with those trained using the wait-k method in terms of translation quality (higher is better) and translation delay (lower is better).

Next we investigate the effect of increasing the delay of the oracle policy in a controlled manner onto the translation quality of the trained systems. As such, we increase the delay of the oracle policy by moving the last READ action in the oracle program to the beginning of the program, and thus increasing the delay of the oracle artificially. For additional delay, we repeat this process. We expect that the delayed oracle programs lead to trained policies with better translation quality at increased delay. The triangles in Figure 2 correspond to policies trained using the versions of the oracle program, where we added delays $\{0-5\}$. Observe that policies trained with the delayed versions of the oracle program consistently outperform the wait-k policies, across all languages.

The quality of the oracle is shown as the green triangle in Figure 2. This system is provided the oracle program at test time, unlike the other systems that allow errors to propagate from the interpreter into subsequent decisions of the programmer. Note both the low delay of the oracle, and also the fact that the BLEU score outperforms offline translation (wait-∞). This seemingly surprising finding can be explained by the oracle providing key information to the interpreter in the form of word and phrase segmentation of the inputs.

Achieving oracle level quality with a learned programmer is particularly difficult, which can be attributed to exposure bias. However, our coupled SS method manages to bridge much of the gap between the learned program and the oracle program.

_Aber _wenn	_wir	_die	_Zusammensetzung	_des	_Erd	boden	s	**_nicht**	**_ändern**	,	_werden	_wir	_das	_nie	_tun	.	</s>
_	B	ut	_if	_we	**_look**	_at	_the	_composition	_of	_	E	ar	th	’	s	_ground	, _we _never _will .

_Aber	_wenn	_wir	_die _Zusammensetzung _des _Erd boden s **_nicht _ändern** ,	_werden _wir	_das _nie	_tun .	</s>
_	B	ut	_if _we **_don ’ t _change** _the _composition _of	_the	_soil ,	_we	_will _never _do _that .

_Aber	_wenn	_wir	_die _Zusammensetzung _des _Erd boden s **_nicht**	**_ändern** ,	_werden _wir	_das _nie	_tun . </s>
_But	_if	_we	**_don ’ t**	**_change** _the _composition _of _the	_soil , _we _will	_never	_do _this .

Table 2: The comparison of our wait-k, our coupled-SS (Co-SS) and the oracle trajectory. A column shows a sequence of consecutive READ and WRITE. Here our proposed method is able to imitate the oracle well, by patiently waiting for sufficient input to produce a good translation. Red texts indicate place of translation error.

Finetuning Wait-k. Next we consider warm-starting training using the wait-k model, and finetuning with the oracle program and SS training method. This method of training is cheaper when a wait-k system is available, as training converges in few iterations. In this setting, we allow retraining of the interpreter, as fixing the interpreter yielded poor BLEU scores. The $\times$ points in Figure 2 show the results of this experiment, where each point was warm-started with a different wait-k system. As it can be seen, all of these runs achieve similar results, but with inferior delay and quality to our proposed method (leftmost $\triangle$). One explanation to this result is that the interpreter already converged to the wait-k policy and retraining it results in an inferior model compared to training it jointly from scratch.[4]

4.3 Qualitative Analysis

Gu et al. (2017) address the difficulty of translating sentences in subject-object-verb order when translating from German to English. We show a typical example in Table 2 where the wait-k systems are forced to make difficult decisions with insufficient evidence, in this case of predicting negation of a verb which appears latter in the input. We compare systems with $AL \approx 2$ and $DAL \approx 4$ which is achieved by the wait-2, our coupled-SS system, and the oracle system. Consider first the wait-2 system. From the state shown at bottom right corner of the smaller red box, the system next generates a poor choice of verb ("look"), which was done without access to the verb in the German input. Instead the rightmost context word was "Zusammensetzung" ("composition"), which gives little information about the verb. One way around this problem is to use a specialized classifier which predicts the final verb (Grissom II et al., 2014). However, this is often onerous or impossible. In this example, the model must also predict the negation "nicht" which appears immediately before the final verb. In a real interpretation scenario, the only way to ensure we output a correct translation is to wait for the matrix verb and negation token.

In this example the oracle trajectory breaks down the input sentence into coherent chunks, and this leads to excellent translation of each segment, and with low delay. This is because the word alignment oracle includes crossing alignment inside a phrase, thus producing sequence of READ and WRITE that do not break phrase translations. We posit that this oracle provides the minimal context needed for SIMT to translate on the fly, with sufficient context to generate each output token.

Here our proposed system closely imitates the oracle trajectory. Our proposed method is more conservative in waiting for the input, waiting until the final verb to make precise prediction. This can be explained by the uncertainty over breaking the phrases by the programmer, and thus is incurs additional delay for the sake of better translation quality. Such behaviour that is observed in the output of human interpreters, who will often wait when they are unsure what the main speaker is talking about.

4.4 Oracle Behaviour towards Alignment

Section 4.3 has partly shown our oracle behavior in translating from a final-verb language into English. Here we discuss the oracle's action when translating into a final verb language (English-Dutch). In English, the past participle is usually found right after the auxiliary verb. In this case, our oracle actions are conservative when waiting for inputs on the target side.

The example is shown in Figure 3 in which "have worked" does not produce a crossing alignment. First, the generated oracle will READ "have" and WRITE "heb". Then it will examine the next word, "jaren", and determine whether it needs to

---

[4] Note that we also try to finetune the wait-k system without scheduled sampling but it yields far worse performance than the one with scheduled sampling. This finding is similar with the main experiment.

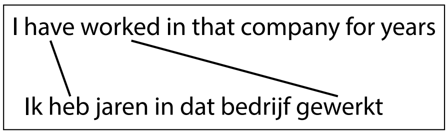


Figure 3: Translation example from English to Dutch. In this case “have worked” produce a non crossing alignment with “heb ... gewerkt” in Dutch.

READ more up until the word it is aligned. When it decided to WRITE “gewerkt”; the word “worked” would have been scanned in the past; so it should WRITE without an additional READ.

Next, it is also inevitable that our produced alignments are noisy and do not align all words correctly. It is currently not clear how much this will hurt the performance of the systems in terms of quality and delay due to alignment errors. In the worst case, our oracle will misguide the interpreter to guess target words without appropriate context (lowering quality, lowering delay) or wait for too many words (increasing quality, increasing delay). Both scenarios resulting from this noisy alignment are not catastrophic as it still depends on the interpreter’s ability to guess translation outputs without appropriate input context.

4.5 Transformer and Large Scale Experiments

To show that the proposed method also extends to the transformer and larger parallel data, we conduct two experiments. The first experiment is changing the LSTM architecture with the transformer architecture similar to Ma et al. (2019). The second experiment uses 4.5 millions DE to EN parallel sentences from WMT 2015.

The interpreter is a standard 6 layers encoder-decoder NMT transformer similar to Vaswani et al. (2017). The programmer consists of single 6 layers encoder transformer with a binary classifier. For both networks, we allow attention to attend only to the previous timesteps. Then we use the program to mask out unseen inputs at each timestep in the interpreter. Unless otherwise specified, we use the default settings of training transformer as in Vaswani et al. (2017). We employ 30% dropout to all transformers and 10% to the embeddings, 16k vocabulary size, an average batch size of 4k, 8k steps learning rate warmup, 50 tokens maximum per sentence during training, for a total of 200k steps. We use a single pass of parallel scheduled sampling (Duckworth et al., 2019) for the transformer to generate $\boldsymbol{a}'$ and $\boldsymbol{y}'$ and set the $\boldsymbol{y}', \boldsymbol{a}', \boldsymbol{a}''$ perturbation rate to be 10%, 15%, and 25%. Training is completed within 20 hours on a single V100 GPU.[5]

	IWSLT			WMT		
	BLEU	AL	AP	BLEU	AL	AP
wait-k						
$k=1$	11.12	2.24	0.60	13.65	1.70	0.56
$k=2$	17.08	2.61	0.62	16.77	2.27	0.59
$k=3$	19.52	3.33	0.66	18.30	2.95	0.62
$k=4$	21.14	4.10	0.70	19.09	3.66	0.64
$k=5$	22.68	4.93	0.73	19.80	4.42	0.67
$k=6$	24.16	5.68	0.76	20.70	5.28	0.70
$k=7$	26.60	6.64	0.79	21.08	6.16	0.72
$k=\infty$	29.53	22.46	1.00	22.51	29.49	1.00
NPI-SIMT	22.06	1.73	0.56	17.57	3.25	0.59
$+\boldsymbol{a}'$	21.40	1.69	0.56	18.18	3.17	0.59
$+\boldsymbol{a}''$	23.28	1.85	0.57	18.70	3.33	0.60
$+\boldsymbol{a}', \boldsymbol{a}''$	23.82	1.83	0.57	18.78	3.54	0.60
$+\boldsymbol{a}', \boldsymbol{a}'', \boldsymbol{y}'$	24.54	1.78	0.57	19.07	3.27	0.59
Oracle-at-test	30.68	1.79	0.57	27.09	2.75	0.58

Table 3: Results using transformer on IWSLT and WMT corpora.

Table 3 presents the Transformer results on IWSLT and WMT. First we see that our transformer results are competitive or better than the LSTM on IWSLT dataset (compare with Table 1). These results are similar to the Ma et al. (2019) when comparing LSTM and transformer based architectures in the SIMT settings. Second, we see that our coupled scheduled sampling approach is also able to increase BLEU by up to 1.5 points compared to vanilla NPI-SIMT approach while also keeping the delay low (3.27 AL). The higher AL of the SIMT model in WMT compared to IWSLT is likely due to the higher AL of the oracle (2.75 vs. 1.79), which we attribute to the nature of the dataset. Arguably, this crawled corpus is less suitable for SIMT in general, because it contains considerably longer parallel sentences; moreover, the text is less reflective of a real simultaneous interpretation setting as it was built by only matching offline and post-edited texts. We are able to see similar improvements using coupled scheduled sampling over vanilla NPI-SIMT approach, showing the scalability of our approach.

5 Related Work

Satija and Pineau (2016); Gu et al. (2017) and Alinejad et al. (2018) formulate simultane-

---

[5] Because of the limitation of our computational resources, we are unable to use multiple GPUs for larger batch size. Using smaller batch size is known to reduce the overall performance of the transformer.

ous NMT as sequential decision making problem where an agent interacts with the environment (i.e. the underlying NMT model) through READ/WRITE actions. They pre-train the NMT system, while the agent's policy is trained using deep-RL.

Arivazhagan et al. (2020) highlights the poor performance of finetuned offline translation model when translating prefixes of input, which is the case of SIMT. Their approach uses retranslation strategy where every READ is performed, a new translation is generated from scratch, allowing revising translation on the fly and mitigating error propagation on the decoder that was attributed to the insufficient evidence when generating past output words. Their approach uses a stability metric which takes number of suffixes revisions made to produce latest translation. This approach involves wait-k inference, which limits number of words that can be emitted by the interpreter during one writing and thus limiting number of suffix revisions at the next writing. This wait-k inference is a heuristic that can be replaced by learning from the oracle.

Ma et al. (2019); Dalvi et al. (2018) introduced the fixed wait-k policy, which allows the integrated training of the NMT model wrt the fixed policy, as opposed to the adaptive policy of Gu et al. (2017); Arivazhagan et al. (2019) jointly trains an adaptive policy and re-trains the underlying NMT system. Arivazhagan et al. (2019); Zheng et al. (2019b) produces oracle READ/WRITE actions using a pre-trained NMT model, which is then used to train an adaptive agent based on supervised learning, i.e. behavioural cloning in imitation learning. Compared to our oracle which is produced merely from word alignment, their method requires a full decoding of training corpus, which is computationally expensive. These works are different from ours in that: (i) they do not use word alignment to produce the oracle actions, and (ii) they do not use of scheduled sampling.

6 Conclusion

This paper proposes a simple and effective way to train a simultaneous translation system to produce low delay translations. Our central contribution is to determine a sufficient, if not minimum, amount of inputs to translate each target token. This is achieved using word-alignment to create an oracle, which is then used as part of a training algorithm based on imitation learning to learn coupled policies, for a "programmer" which decides when to wait for more input producing translation tokens, and an "interpreter" which generates the translation. We show the importance of scheduled sampling during learning, which is crucial to combat exposure bias. Overall we show improvements in BLEU score over naively trained systems with modest translation delays.

Future work is needed to better understand the effect of various alignment models and symmetrization methods on the generated oracle. Beyond this, other opportunities include applying the model to the real speech input and applying more sophisticated imitation learning techniques that involves the generated trajectories of both "interpreter" and "programmer".

Acknowledgment

We thank anonymous reviewers for their valuable comments that helped to make this paper considerably better. This work is supported by the Australian Research Council (ARC DP160102686) and an Amazon Research Award.

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020. Re-translation versus streaming for simultaneous translation. *arXiv preprint arXiv:2004.03643*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In C. Cortes,

N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1171–1179. Curran Associates, Inc.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 493–499.

Daniel Duckworth, Arvind Neelakantan, Ben Goodrich, Lukasz Kaiser, and Samy Bengio. 2019. Parallel scheduled sampling. *arXiv preprint arXiv:1906.04331*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

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

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 627–635.

Harsh Satija and Joelle Pineau. 2016. Simultaneous machine translation using deep reinforcement learning. In *Proceedings of the Abstraction in Reinforcement Learning Workshop*.

Faraz Torabi, Garrett Warnell, and Peter Stone. 2019. Recent advances in imitation learning from observation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6325–6331. International Joint Conferences on Artificial Intelligence Organization.

Francis M Tyers and Murat Serdar Alperen. 2010. South-east european times: A parallel corpus of balkan languages. In *Proceedings of the LREC Workshop on Exploitation of Multilingual Resources and Tools for Central and (South-) Eastern European Languages*, pages 49–53.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019a. Simpler and faster learning of adaptive policies for simultaneous translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1349–1354.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019b. Simultaneous translation with flexible policy via restricted imitation learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5816–5822.

Complementary Evidence Identification in Open-Domain Question Answering

Xiangyang Mou
Rensselaer Polytechnic Institute
`moux4@rpi.edu`

Mo Yu
IBM Research
`yum@us.ibm.com`

Shiyu Chang
MIT-IBM Watson AI Lab
`shiyu.chang@ibm.com`

Yufei Feng
Queen's University
`feng.yufei@queensu.ca`

Li Zhang
Amazon Web Services
`lzhangza@amazon.com`

Hui Su
Fidelity
`Hui.Su@fmr.com`

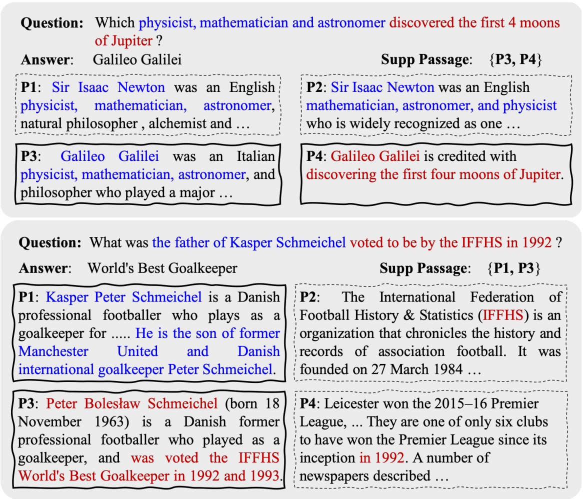
Abstract

This paper proposes a new problem of complementary evidence identification for open-domain question answering (QA). The problem aims to efficiently find a small set of passages that covers full evidence from multiple aspects as to answer a complex question. To this end, we proposes a method that learns vector representations of passages and models the sufficiency and diversity within the selected set, in addition to the relevance between the question and passages. Our experiments demonstrate that our method considers the dependence within the supporting evidence and significantly improves the accuracy of complementary evidence selection in QA domain.

1 Introduction

In recent years, significant progress has been made in the field of open-domain question answering (Chen et al., 2017; Wang et al., 2017, 2018; Clark and Gardner, 2018; Min et al., 2018; Asai et al., 2019). Very recently, some works turn to deal with a more challenging task of asking complex questions (Welbl et al., 2018; Clark et al., 2018; Yang et al., 2018) from the open-domain text corpus. In the open-domain scenario, one critical challenge raised by complex questions is that each question may require multiple pieces of evidence to get the right answer, while the evidence usually scatters in different passages. Examples in Figure 1 shows two types of questions that require evidence from multiple passages.

To deal with the challenging multi-evidence questions, an open-domain QA system should be able to (1) efficiently retrieve a small number of passages that cover the full evidence; and (2) accurately extract the answer by jointly considering the candidate evidence passages. While there have been several prior works in the latter direction (Wang et al., 2017; Clark and Gardner, 2018; Lin et al., 2018), the solutions to the first problem still rely on traditional or neural information retrieval (IR) approaches, which solely measure the relevance between the question and each individual paragraph, and will highly possibly put the wrong evidence to the top.[1] For example in Figure 1 (top), **P1** and **P2** are two candidate evidence passages that are closely related to the question but only cover the same unilateral fact required by the question, therefore leading us to the wrong answer *Newton*.



Question: Which physicist, mathematician and astronomer discovered the first 4 moons of Jupiter ?
Answer: Galileo Galilei
Supp Passage: {**P3, P4**}

P1: Sir Isaac Newton was an English physicist, mathematician, astronomer, natural philosopher , alchemist and …

P2: Sir Isaac Newton was an English mathematician, astronomer, and physicist who is widely recognized as one …

P3: Galileo Galilei was an Italian physicist, mathematician, astronomer, and philosopher who played a major …

P4: Galileo Galilei is credited with discovering the first four moons of Jupiter.

Question: What was the father of Kasper Schmeichel voted to be by the IFFHS in 1992 ?
Answer: World's Best Goalkeeper
Supp Passage: {**P1, P3**}

P1: Kasper Peter Schmeichel is a Danish professional footballer who plays as a goalkeeper for ..... He is the son of former Manchester United and Danish international goalkeeper Peter Schmeichel.

P2: The International Federation of Football History & Statistics (IFFHS) is an organization that chronicles the history and records of association football. It was founded on 27 March 1984 …

P3: Peter Bolesław Schmeichel (born 18 November 1963) is a Danish former professional footballer who played as a goalkeeper, and was voted the IFFHS World's Best Goalkeeper in 1992 and 1993.

P4: Leicester won the 2015–16 Premier League, ... They are one of only six clubs to have won the Premier League since its inception in 1992. A number of newspapers described …

Figure 1: Examples of complex questions involving two facts of a person. Different facts are color-coded. **P#** are all relevant passages, while only the ones with solid-line boxes are the true supporting passages.

This paper formulates a new problem of **complementary evidence identification** for answering complex questions. The key idea is to consider the problem as measuring the properties of the selected passages, more than the individual relevance. Specifically, we hope the selected passages can serve as a set of spanning bases that supports the

---

[1] (Min et al., 2019) pointed out the shortcut problem in multi-hop QA. However, as some works (Wang et al., 2019) show that even a better designed multi-hop model can still benefit from full evidence in such situation.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2720–2726
April 19 - 23, 2021.

question. The selected passage set thus should satisfy the properties of (1)*relevancy*, i.e., they should be closely related to the question; (2) *diversity*, i.e., they should cover diverse information given the coverage property is satisfied; (3) *compactness*, i.e., the number of passages to satisfy the above properties should be minimal. With these three defined properties, we hope to both improve the selective accuracy and encourage the interpretability of the evidence identification. Note that complementary evidence identification in QA is different from Search Result Diversification (SRD) in IR on their requirement of compactness. The size of the selected set is constrained in QA tasks by the capability of downstream reasoning models and practically needs to be a small value, whereas it is not the case in SRD.

To achieve the above goals, a straightforward approach is to train a model that evaluates each subset of the candidate passages, e.g., by concatenating passages in any subsets. However, this approach is highly inefficient since it requires to encode $O(K^L)$ passage subsets, where K is the total number of candidates and L is the maximum size of subsets. Thus, a practical complementary evidence identification method needs to be computationally efficient. This is especially critical when we use heavy models like ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018), where passage encoding is time and memory consuming.

To this end, we propose an efficient method to select a set of spanning passages that is sufficient and diverse. The core idea is to represent questions and passages in a vector space and define the measures of our criterion in the vector space. For example, in the vector space, sufficiency can be defined as a similarity between the question vector and the sum of selected passage vectors, measured by a cosine function with a higher score indicating a closer similarity; and diversity can be defined as ℓ_1 distance between each pair of passages. By properly training the passage encoder with a loss function derived by the above terms, we expect the resulted vector space satisfies the property that the complementary evidence passages lead to large scores. In addition, our method only encodes each passage in the candidate set once, which is more efficient than the naive solution mentioned above. To evaluate the proposed method, we use the multi-hop QA dataset HotpotQA (the full wiki setting) since the ground-truth of evidence passages are provided. Experiments show that our method significantly improves the accuracy of complementary evidence selection.

2 Proposed Method

2.1 Task Definition

Given a question q and a mixture set of paragraphs $\mathcal{P} = \mathcal{P}^+ \cup \mathcal{P}^-$ with some paragraphs $p \in \mathcal{P}^+$ relevant to q and some $p \in \mathcal{P}^-$ irrelevant. Our goal is to select a small subset of paragraphs $\mathcal{P}_{sel} \subset \mathcal{P}$, such that every $p \in \mathcal{P}_{sel}$ satisfies $p \in \mathcal{P}^+$ (relevancy), and all $p \in \mathcal{P}_{sel}$ can jointly cover all the information asked by q (complementary). The off-the-shelf models select relevant paragraphs independently, thus usually cannot deal with the complementary property. The inner dependency among the selected $\mathcal{P}_{sel}$ needs to be considered, which will be modeled in the remaining of the section.

2.2 Model and Training

Vector Space Modeling We apply BERT model to estimate the likelihood of a paragraph p being the supporting evidence to the question q, denoted as $P(p|q)$. Let q and p_i denote the input texts of a question and a passage. We feed q and the concatenation of q and p_i into the BERT model, and use the hidden states of the last layer to represent q and p_i in vector space, denoted as $\boldsymbol{q}$ and $\boldsymbol{p_i}$ respectively. A fully connected layer $f(\cdot)$ followed by sigmoid activation is added to the end of the BERT model, and outputs a scalar $P(p_i|q)$ to estimate how relevant the paragraph p_i is to the question. Note that in our implementation $\boldsymbol{p}_i$ is based on both q and p_i, but we omit the condition on q for simplicity.

Complementary Conditions Previous works extract evidence paragraphs according to $P(p|q)$, which is estimated on each passage separately without considering the dependency among selected paragraphs. To extract complementary evidence, we propose that the selected passages $\mathcal{P}_{sel}$ should satisfy the following conditions that intuitively encourage each selected passage to be a basis to support the question:

- **Relevancy:** $\mathcal{P}_{sel}$ should have a high probability of $\sum_{p_i \in \mathcal{P}_{sel}} P(p_i|q)$;
- **Diversity:** $\mathcal{P}_{sel}$ should cover passages as diverse as possible, which can be measured by the average distance between any pairs in $\mathcal{P}_{sel}$, e.g., maximizing $\sum_{i,j \in \{i,j|p_i,p_j \in \mathcal{P}_{sel}, i \neq j\}} \ell_1(\boldsymbol{p}_i, \boldsymbol{p}_j)$. Here

$\ell_1(\cdot,\cdot)$ denotes L_1 distance;

- **Compactness:** $\mathcal{P}_{sel}$ should optimize the aforementioned conditions while the size being minimal. In this work we constrain the compactness by fixing $|\mathcal{P}_{sel}|$ and meanwhile maximizing $cos(\sum_{i\in\{i|p_i\in\mathcal{P}_{sel}\}}\boldsymbol{p}_i,\boldsymbol{q})$. We use $\cos(\cdot,\cdot)$ to encourage the collection of evidence covers what needed by the question.

Training with Complementary Regularization We propose a new supervised training objective to learn the BERT encoder for QA that optimizes the previous conditions. Note that in this work we assume a set of labeled training examples are available, i.e., the ground truth annotations contain complementary supporting paragraphs. Recently there was a growing in such datasets (Yang et al., 2018; Yao et al., 2019), due to the increasing interest in model explainability. Also, such supervision signals can also be obtained with distant supervision.

For each training instance $(q,\mathcal{P})$, we define

$$\{\boldsymbol{p}_i\}^+ = \{\boldsymbol{p}_i\},\quad \forall i\in\{i|p_i\in\mathcal{P}^+\} \tag{1}$$

$$\{\boldsymbol{p}_i\}^- = \{\boldsymbol{p}_i\},\quad \forall i\in\{i|p_i\in\mathcal{P}^-\} \tag{2}$$

$$\{\boldsymbol{p}_i\} = \{\boldsymbol{p}_i\}^+\cup\{\boldsymbol{p}_i\}^- \tag{3}$$

Denoting $y_{p_i} = 1$ if $p_i\in\mathcal{P}^+$ and $y_{p_i}=0$ if $p_i\in\mathcal{P}^-$, we have the following training objective function:

$$\begin{aligned}\mathcal{L}(\{\boldsymbol{p}_i\};\boldsymbol{q};y) &= \mathcal{L}_{sup}(\{\boldsymbol{p}_i\};\boldsymbol{q};y)\\ &\quad+\alpha\mathcal{L}_d(\{\boldsymbol{p}_i\}^+)+\beta\mathcal{L}_c(\{\boldsymbol{p}_i\};\boldsymbol{q};y)\end{aligned} \tag{4}$$

where

$$\mathcal{L}_{sup}(\{\boldsymbol{p}_i\};\boldsymbol{q};y) = -\sum_i y_{p_i}\log(f(\boldsymbol{p}_i)), \tag{5}$$

$$\mathcal{L}_d(\{\boldsymbol{p}_i\}^+) = \sum_{\boldsymbol{p_i},\boldsymbol{p_j},i\neq j}(1-\ell_1(\boldsymbol{p_i},\boldsymbol{p_j})). \tag{6}$$

$$\mathcal{L}_c(\{\boldsymbol{p}_i\};\boldsymbol{q};y) = \begin{cases}1-\cos(\boldsymbol{q},\sum_i\boldsymbol{p}_i),\\ \quad\text{if }\ \Pi_{p_i}y_{p_i}=1\\ \max(0,\cos(\boldsymbol{q},\sum_i\boldsymbol{p}_i)-\gamma),\\ \quad\text{if }\ \Pi_{p_i}y_{p_i}=0\end{cases} \tag{7}$$

where α and β are the hyperparameter weights and $\ell_1(\cdot,\cdot)$ denotes L1 loss between two input vectors. Eq 5 is the cross-entropy loss corresponding to relevance condition; Eq 6 regularizes the diversity condition; Eq 7 is the cosine-embedding loss[2] for the compactness condition and $\gamma > 0$ is the margin to encourage data samples with better question coverage.

2.3 Inference via Beam Search

Score Function During inference, we use the following score function to find the best paragraph combination:

$$\begin{aligned}g(\mathcal{P}_{sel};q;\{\boldsymbol{p}_i\}) &= \sum_{p_i}P(p_i|q)+\alpha\cos(\sum_{\boldsymbol{p_i}}\boldsymbol{p}_i,\boldsymbol{q})\\ &\quad+\beta\sum_{\boldsymbol{p_i},\boldsymbol{p_j},i\neq j}\ell_1(\boldsymbol{p}_i,\boldsymbol{p}_j)\end{aligned} \tag{8}$$

where α and β are hyperparameters similar to Eq 4. Note that our approach requires to encode each passage in $\mathcal{P}$ only once for each question, resulting in an $O(K)$ time complexity of encoding ($K = |\mathcal{P}|$); and the subset selection is performed in the vector space, which is much more efficient than selecting subsets before encoding.

Beam Search In a real-world application, there is usually a large candidate set of $\mathcal{P}$, e.g., retrieved passages for q via a traditional IR system. Our algorithm requires $O(K)$ time encoding, and $O(K^L)$ time scoring in vector space when ranking all the combinations in L candidates. Thus when K becomes large, it is still inefficient even when $L = 2$. We resort to beam search to deal with scenarios with large Ks. The details can be found in Appendix A.

3 Experiments

3.1 Settings

Datasets Considering the prerequisite of sentence-level evidence annotations, we evaluate our approach on two datasets, a synthetic dataset **MNLI-12** and a real application **HotpotQA-50**. Data sampling is detailed in Appendix B.

- **MNLI-12** is constructed based on the textual entailment dataset MNLI (Williams et al., 2018), in order to verify the ability of our method in finding complementary evidence. In original MNLI, each premise sentence corresponds to three hypotheses sentences: entailment, neutral and contradiction. To generate complementary pairs for each premise sentence, we split each hypothesis sentence into two segments. The goal is to find the segment combination that entails premise sentence, and our dataset, by definition, ensures that only the combination of two segments from the entailment hypothesis can entail the premise, not any of its subset or other combinations. The original train/dev/test splits from MNLI are used.

---

[2] Refer to CosineEmbeddingLoss in PyTorch.

• **HotpotQA-50** is based on the open-domain setting of the multi-hop QA benchmark HotpotQA (Yang et al., 2018). The original task requires to find evidence passages from abstract paragraphs of all Wikipedia pages to support a multi-hop question. For each q, we collect 50 relevant passages based on bigram BM25 (Godbole et al., 2019). Two positive evidence passages to each question are provided by human annotators as the ground truth. Note that there is no guarantee that $\mathcal{P}_{50}$ covers both evidence passages here. We use the original development set from HotpotQA as our test set and randomly split a subset from the original training set as our development set.

3.2 Settings

Baseline We compare with the BERT passage ranker (Nie et al., 2019) that is commonly used on open-domain QA including HotpotQA. The baseline uses the same BERT architecture as our approach described in Section 2.2, but is trained with only the relevancy loss (Eq 5) and therefore only consider the relevancy when selecting evidence.

We also compare the DRN model from (Harel et al., 2019) which is designed for the SRD task. Their ensemble system first finds the most relevant evidence to the given question, and then select the second diverse evidence using their score function. The major differences from our method are that (1) they train two separate models for evidence selection; (2) they do not consider the compactness among the evidences. It is worth mentioning that we replace their LSTM encoder with BERT encoder for fair comparison.

Metric During the evaluation we make each method output its top 2 ranked results[3] (i.e. the top 1 ranked pair from our method) as the prediction. The final performance is evaluated by exact match (EM), i.e., whether both true evidence passages are covered, and the F1 score on the test sets.

3.3 Results

In the experiments, we have $M = 3$, $N = 4$ for MNLI-12 and $M = 4$, $N = 5$ for HotpotQA-50 with our method. The values are selected according to development performance. We follow the settings and hyperparameters used in (Harel et al., 2019) for the DRN model. Table 1 shows the performance. The upper-bound measures how many pieces of true evidences enclosed by the complete set of candidate passages where our proposed ranker selects from. For HotpotQA dataset, we use a bi-gram BM25 ranker to collect top 50 relevant passages and build the basis for the experiments[4], which inevitably leads some of the true evidences to be filtered out and makes its upper-bound less than 100%. For the artificial MNLI-12 dataset, all the true evidences are guaranteed to be included.

System	**HotpotQA-50**		**MNLI-12**	
	EM	**F1**	**EM**	**F1**
Baseline Ranker	16.67	41.29	41.61	67.57
DRN + BERT	1.03	35.37	6.20	46.07
Our Method	**20.15**	**49.10**	**53.81**	**73.18**
Upper-Bound	35.49	61.08	100.00	100.00

Table 1: Model Evaluation (%). The upper-bound indicates the amount of true evidences contained by all candidate passages. The baseline ranker is a BERT ranker trained only with relevancy loss.

Table 1 shows that our method achieves significant improvements on both datasets. On HotpotQA-50, all systems have low EM scores, because of the relatively low recall of the BM25 retrieval. Only 35.49% of the samples in the test set contain both ground-truth evidence passages. On MNLI-12, the EM score is around 50%. This is mainly because the segments are usually much shorter than a paragraph, with an average length of 7 words. Therefore it is more challenging in matching the q with the p_is. Specifically, both our method and the BERT baseline surpass the DRN model on all datasets and metrics, which results from our question-conditioned passage encoding approach. Our defined vector space proves beneficial to model the complementation among the evidence with respect to a given question. The ablation study of our loss function further illustrates that the diversity and the compactness terms efficiently bring additional 20%/30% increase in EM score on two datasets and consequently raise the F1 score by about 8/6 absolute points.

Figure 2 gives examples about how our model improves over the baseline. Our method can successfully select complementary passages while the baselines only select passages that look similar to the question. A more interesting example is given at the bottom where the top-50 only covers one supporting passage. The BERT baseline selects two

---

[3] There is only one positive pair of evidences for each question.

[4] This is the standard setting that starts with BM25 retrieval to make the inference time efficient enough without loss of generality.

incorrect passages that cover identical part of facts required by the question and similarly the DRN baseline select a relevant evidence and an irrelevant evidence, while our method scores lower the second passage that does not bring new information, and reaches a supporting selection. A similar situation contributes to the majority of improvement on one-supporting-evidence data sample in HotpotQA-50.

Inference Speed Our beam search with score function brings slight overheads to the running time. On HotpotQA-50, it takes 1,990 milliseconds (ms) on average to obtain the embeddings of all passages for one data sample whereas our vector-based complementary selection only adds an extra 2 ms which can be negligible compared to the encoding time.

3.4 Future Work

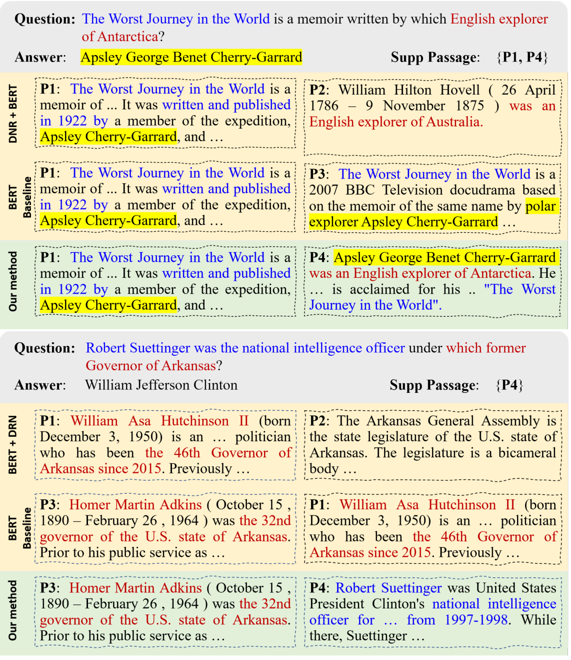
The latest dense retrieval methods (Lee et al., 2019; Karpukhin et al., 2020; Guu et al., 2020) show promising results on efficient inference on the full set of Wikipedia articles, which allows to skip the initial standard BM25 retrieval and avoid the significant loss during the pre-processing step. Our proposed approach is able to directly cooperate with these methods as we all work in the vector space. Therefore, the extension to dense retrieval can be naturally the next step of our work.

4 Conclusion

In the paper, we propose a new problem of complementary evidence identification and define the criterion of complementary evidence in vector space. We further design an algorithm and a loss function to support efficient training and inference for complementary evidence selection. Compared to the baseline, our approach improves more than 20% and remains to scale well to the computationally complex cases.

Acknowledgment

A special thank to Rensselaer-IBM Artificial Intelligence Research Collaboration (RPI-AIRC) for providing externship and other supports. This work is funded by Cognitive and Immersive Systems Lab (CISL), a collaboration between IBM and RPI, and also a center in IBM's AI Horizons Network.



Question: The Worst Journey in the World is a memoir written by which English explorer of Antarctica?

Answer: Apsley George Benet Cherry-Garrard **Supp Passage:** {**P1, P4**}

DNR + BERT	**P1**: The Worst Journey in the World is a memoir of ... It was written and published in 1922 by a member of the expedition, Apsley Cherry-Garrard, and …	**P2**: William Hilton Hovell (26 April 1786 – 9 November 1875) was an English explorer of Australia.
BERT Baseline	**P1**: The Worst Journey in the World is a memoir of ... It was written and published in 1922 by a member of the expedition, Apsley Cherry-Garrard, and …	**P3**: The Worst Journey in the World is a 2007 BBC Television docudrama based on the memoir of the same name by polar explorer Apsley Cherry-Garrard …
Our method	**P1**: The Worst Journey in the World is a memoir of ... It was written and published in 1922 by a member of the expedition, Apsley Cherry-Garrard, and …	**P4**: Apsley George Benet Cherry-Garrard was an English explorer of Antarctica. He … is acclaimed for his .. "The Worst Journey in the World".

Question: Robert Suettinger was the national intelligence officer under which former Governor of Arkansas?

Answer: William Jefferson Clinton **Supp Passage:** {**P4**}

BERT + DRN	**P1**: William Asa Hutchinson II (born December 3, 1950) is an … politician who has been the 46th Governor of Arkansas since 2015. Previously …	**P2**: The Arkansas General Assembly is the state legislature of the U.S. state of Arkansas. The legislature is a bicameral body …
BERT Baseline	**P3**: Homer Martin Adkins (October 15 , 1890 – February 26 , 1964) was the 32nd governor of the U.S. state of Arkansas. Prior to his public service as …	**P1**: William Asa Hutchinson II (born December 3, 1950) is an … politician who has been the 46th Governor of Arkansas since 2015. Previously …
Our method	**P3**: Homer Martin Adkins (October 15 , 1890 – February 26 , 1964) was the 32nd governor of the U.S. state of Arkansas. Prior to his public service as …	**P4**: Robert Suettinger was United States President Clinton's national intelligence officer for … from 1997-1998. While there, Suettinger …

Figure 2: Gain from complementary selection. In both examples, the DRN baseline first finds the most relevant evidence to the question (left) and then select a diverse one (right); the BERT baseline model selected the top-2 most relevant passages (**P1**, **P2**) to the question regardless of their complementation; whereas our model made the selection (**P1**, **P3**) with consideration of both relevance and evidence sufficiency. Note that, in the bottom example, one of the ground-truth supporting passages and the answer were excluded when building the dataset.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia graph for question answering. *arXiv preprint arXiv:1911.10470*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ameya Godbole, Dilip Kavarthapu, Rajarshi Das, Zhiyu Gong, Abhishek Singhal, Hamed Zamani, Mo Yu, Tian Gao, Xiaoxiao Guo, Manzil Zaheer, et al. 2019. Multi-step entity-centric information retrieval for multi-hop question answering. *arXiv preprint arXiv:1909.07598*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Shahar Harel, Sefi Albo, Eugene Agichtein, and Kira Radinsky. 2019. Learning novelty-aware ranking of answers to complex questions. In *The World Wide Web Conference*, pages 2799–2805.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1736–1745.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. *arXiv preprint arXiv:1906.02900*.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. *arXiv preprint arXiv:1805.08092*.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2553–2566.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Haoyu Wang, Mo Yu, Xiaoxiao Guo, Rajarshi Das, Wenhan Xiong, and Tian Gao. 2019. Do multi-hop readers dream of reasoning chains? *arXiv preprint arXiv:1910.14520*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2017. Evidence aggregation for answer re-ranking in open-domain question answering. *arXiv preprint arXiv:1711.05116*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. *arXiv preprint arXiv:1906.06127*.

A Complementary Evidence Selection via Beam Search

For efficient inference when $L = 2$, we start to select the top-N ($N \ll K$) most relevant passages. Then we score the combinations between each passage pair in the top-N set and another top-M set. This reduces the complexity from $O(K^2)$ to $O(MN)$. M is a hyperparameter corresponding to the beam size. In a more general setting with $L \geq 2$, we have an algorithm with the complexity of $O((L-1)MN)$ instead of $O(K^L)$, which is shown in algorithm 1.

Algorithm 1: Complementary Evidence Selection via Beam Search

```
Data: Vector representation of question ($\boldsymbol{q}$), vector representation of all the $N$ passages $\{p_n\}$ ($\{\boldsymbol{p_n}\}$); the maximum
      number of passage to select ($L$); the beam size ($M$); a vector of weights for all regularization terms $\boldsymbol{\lambda}$.
Result: The top ranked complementary passages.
/* Predict the probability $P(\boldsymbol{p_i})$ of being a supporting passage for each passage
   $\boldsymbol{p_i}$ given $\boldsymbol{q}$                                                          */
for $i \in [1, N]$ do
    $P(\boldsymbol{p_i}) \leftarrow f(\boldsymbol{q}, \boldsymbol{p_i})$;
end
Rank the passages by $P(\boldsymbol{p_i})$;
$\boldsymbol{P_{span}}$ = []
Pick $M$ passages with top $P(\boldsymbol{p_i})$ into $\boldsymbol{P_{span}}$;
for $depth \in [2, L]$ do
    $\boldsymbol{P'_{span}}$ = [] ;
    for $j \in [1, M]$ do
        /* $\boldsymbol{P_j}$ is a selected subset, $s_j$ is the corresponding score          */
        Pop the $j$-th tuple $(\boldsymbol{P_j}, s_j)$ from $\boldsymbol{P_{span}}$;
        for $n \in [1, N]$ do
            if The set $\boldsymbol{P_j} \cup \{\boldsymbol{p_n}\}$ is covered by $\boldsymbol{P'_{span}}$ then
                continue
            end
            /* $\boldsymbol{r}_n$ is the regulation increases by adding $\boldsymbol{p_n}$ to $\boldsymbol{P_j}$          */
            Put $(\boldsymbol{P_j} \cup \{\boldsymbol{p_n}\}, s_j + P(\boldsymbol{p}_n) + \boldsymbol{\lambda r}_n)$ into $\boldsymbol{P'_{span}}$;
            if More than $M$ tuples added based on $\boldsymbol{P_j}$ then
                break
            end
        end
    end
    Rank $\boldsymbol{P'_{span}}$ according to the scores;
    $\boldsymbol{P_{span}} \leftarrow \boldsymbol{P'_{span}}[1:M]$
end
Return $\boldsymbol{P_{span}}[0]$
```

B Data Sampling

MNLI-12 In original MNLI, each premise sentence P corresponds to one entailment E_P, one neutral N_P and one contradiction C_P. We take the premise P as q, and split each of its corresponding hypotheses into two segments with a random cutting point near the middle of the sentence, resulting in a total of 6 segments $\{E_P^1, E_P^2, N_P^1, N_P^2, C_P^1, C_P^2\}$. Mixing them with the 6 segments corresponding to another premise X, we can finally have $\mathcal{P}^+ = \{E_P^1, E_P^2\}$ and $\mathcal{P}^- = \{N_P^1, N_P^2, C_P^1, C_P^2, E_X^1, E_X^2, N_X^1, N_X^2, C_X^1, C_X^2\}$. Consequently, we sample one positive and eight negative pairs respectively from $\mathcal{P}^+$ and $\mathcal{P}^-$. A pair like $\{E_P^1, C_X^2\}$ is considered as negative. To ensure the segments are literally meaningful, each segment is guaranteed to be longer than 5 words.

HotpotQA In HotpotQA, the true supporting paragraphs of each question q are given. Therefore, we can easily form $\mathcal{P}^+$ and $\mathcal{P}^-$ and sample positive and negative pairs of paragraphs respectively from $\mathcal{P}^+$ and $\mathcal{P}^-$. A special pair that contains one true supporting paragraph and one non-supporting paragraph is considered as a negative pair.

Entity-level Factual Consistency of Abstractive Text Summarization

Feng Nan[1] **Ramesh Nallapati**[1] **Zhiguo Wang**[1] **Cicero Nogueira dos Santos**[1]
Henghui Zhu[1] **Dejiao Zhang**[1] **Kathleen McKeown**[1,2] **Bing Xiang**[1]
Amazon Web Services[1], Columbia University[2]
`{nanfen,rnallapa,zhiguow,cicnog,henghui,dejiaoz,mckeownk,bxiang}`
`@amazon.com`

Abstract

A key challenge for abstractive summarization is ensuring factual consistency of the generated summary with respect to the original document. For example, state-of-the-art models trained on existing datasets exhibit entity hallucination, generating names of entities that are not present in the source document. We propose a set of new metrics to quantify the entity-level factual consistency of generated summaries and we show that the entity hallucination problem can be alleviated by simply filtering the training data. In addition, we propose a summary-worthy entity classification task to the training process as well as a joint entity and summary generation approach, which yield further improvements in entity level metrics.

1 Introduction

Many recent advances in deep neural networks have led to significant improvement in the quality of abstractive summarization (Radford et al., 2019; Gehrmann et al., 2019; Lewis et al., 2019). Despite this progress, there are still many limitations facing neural text summarization (Kryscinski et al., 2019), the most serious of which is their tendency to generate summaries that are not factually consistent with the input document; a factually consistent summary only contains statements that can be derived from the source document. Recent studies show that about 30% of the summaries generated by neural network sequence-to-sequence models suffer from fact fabrication (Cao et al., 2018). Unfortunately, the widely used ROUGE score is inadequate to quantify factual consistency (Kryscinski et al., 2019).

Factual inconsistency can occur at either the entity or the relation level. At the entity level, a model generated summary may contain named-entities that never appeared in the source document. We call this the entity *hallucination* problem. For example, consider the following model generated summary:

> *People in Italy and the Netherlands are more likely to consume fewer cups of coffee than those in the* <u>UK</u>*, a study suggests.*

"UK" never appeared in the input source document (taken from the test set of the XSUM dataset (Narayan et al., 2018)). In fact, the source document mentioned a study involving people in Italy and Netherlands; "UK" was a result of model hallucination. Another type of inconsistency occurs when the entities indeed exist in the source document but the relations between them are not in the source document. This type of inconsistency is much harder to identify. Open Information Extraction (OpenIE) and dependency parsing tools have been used (Cao et al., 2018) to identify the underlying relations in a summary, but are not yet accurate enough for practical use. Ultimately, these researchers relied on manually classifying generated summaries into *faithful*, *fake*, or *unclear*.

In this paper, we propose a set of simple metrics to quantify factual consistency at the entity-level. We analyze the factual quality of summaries produced by the state-of-the-art BART model (Lewis et al., 2019) on three news datasets. We then propose several techniques including data filtering, multi-task learning and joint sequence generation to improve performance on these metrics. We leave the relation level consistency to future work.

2 Related work

Large transformer-based neural architectures combined with pre-training have set new records across many natural language processing tasks (Vaswani et al., 2017; Devlin et al., 2019;

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2727–2733
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Radford et al., 2019). In particular, the BART model (Lewis et al., 2019) has shown superior performance in many text generation tasks including abstractive summarization. In contrast to encoder-only pre-training such as in BERT (Devlin et al., 2019) or decoder-only pre-training such as in GPT-2 (Radford et al., 2019), BART is an encoder-decoder transformer-based neural translation model jointly pre-trained to reconstruct corrupted input sequences of text.

Several authors have pointed out the problem of factual inconsistency in abstractive summarization models (Kryscinski et al., 2019; Kryściński et al., 2019; Cao et al., 2018; Welleck et al., 2019). The authors in (Kryściński et al., 2019) proposed to train a neural network model to classify if a summary is factually consistent with a given source document, similar to a natural language inference task. In the dialogue generation setting, authors in (Li et al., 2019) proposed using unlikelihood to surpress logically inconsistent responses. Our work is complementary to such existing approaches as we focus on simple entity-level metrics to quantify and improve factual consistency. Our goal of improving entity-level metrics of summaries is also related to controllable abstractive summarization (Fan et al., 2018), where a list of named-entities that a user wants to see in the summary can be passed as input to influence the generated summary. In contrast, our goal is to *predict* which entities are summary-worthy while generating the summary that contains them. In this view we are trying to solve a more challenging problem.

3 Entity-level factual consistency metrics

We propose three new metrics that rely on off-the-shelf tools to perform Named-Entity Recognition (NER). [1] We use $\mathcal{N}(t)$ and $\mathcal{N}(h)$ to denote the number of named-entities in the target (gold summary) and hypothesis (generated summary), respectively. We use $\mathcal{N}(h \cap s)$ to denote the number of entities found in the generated summary that can find a match in the source document. If a named-entity in the summary consists of multiple words, we consider it a match as long as any n-gram of the named-entity can be found in the source document. This is meant to capture the situation where the named-entity can be shortened; for example, "Obama " is a match for "Barack Obama" and "Harvard" is a match for "Harvard University". When the match is at the unigram level, we make sure that it is not a stop word such as "the". We also make the match case-insensitive to accommodate casing variances.

Precision-source: We propose precision-source ($\mathbf{prec}_s$) to quantify the degree of hallucination with respect to the source: $\mathbf{prec}_s = \mathcal{N}(h \cap s)/\mathcal{N}(h)$. It is simply the percentage of named-entities in the summary that can be found in the source. Low $\mathbf{prec}_s$ means hallucination is severe.

We first evaluate the $\mathbf{prec}_s$ score on the ground truth summaries of the 3 datasets: Newsroom (Grusky et al., 2018), CNN/DailyMail (Nallapati et al., 2016) and XSUM (Narayan et al., 2018). Table 1 shows that among the three datasets, the ground truth summaries in XSUM have the lowest $\mathbf{prec}_s$ score. This is because the ground truth summaries in the XSUM dataset often use the first sentence of the article as the summary; the source document is constructed to be the rest of the article and may not repeat the named-entities that appeared in the summary. We hypothesize that the hallucination problem is largely caused by the training data itself. Thus, we propose to perform entity-based data filtering to construct a "clean" version of these datasets as described next.

	Newsroom			CNNDM			XSUM		
	train	val	test	train	val	test	train	val	test
avg. $\mathcal{N}(t)$	2.08	2.10	2.09	4.36	5.09	4.87	2.08	2.06	2.08
avg. $\mathcal{N}(t \cap s)$	1.88	1.90	1.90	4.21	4.92	4.70	1.64	1.64	1.64
$\mathbf{prec}_s$ (%)	90.6	90.6	90.5	96.5	96.7	96.6	79.0	79.5	79.3

Table 1: Average number of named-entities and the $\mathbf{prec}_s$ scores (%) in the ground truth summary.

Entity-based data filtering: For each dataset, we apply Spacy NER on the gold summary to identify all the named-entities. [2] If any of the entities cannot find a match in the source document, we discard the sentence that contains the entity from the ground truth summary. If the ground truth summary consists of only one sentence and it needs to be discarded, we remove the document-summary pair from the dataset. This way, we ensure that our filtered dataset does not contain hallucination of entities ($\mathbf{prec}_s = 1$) in the ground truth summary. The dataset size before and after the filtering is shown in Table 2. About a third of examples are filtered out for XSUM. Again, this is because of

---

[1] We use Spacy (Honnibal and Montani, 2017).

[2] We ignore certain types of entities such as date, time, numerals because they tend to have large variations in representation and are difficult to determine a match in the source document. The appendix contains more details.

the way XSUM dataset is constructed as mentioned in the previous paragraph. As we shall see in Table 3, entity-based data filtering reduces hallucination of the trained model and the effect is especially significant in the XSUM dataset.

Precision-target and recall-target: Although the precision-source ($\mathbf{prec}_s$) metric quantifies the degree of entity hallucination with respect to the source document, it does not capture the entity-level accuracy of the generated summary with respect to the ground truth summary. To get a complete picture of the entity-level accuracy of the generated summary, we propose the precision-target ($\mathbf{prec}_t$) score: $\mathbf{prec}_t = \mathcal{N}(h \cap t)/\mathcal{N}(h)$, where $\mathcal{N}(h \cap t)$ is the number of named-entities in the generated summary that can find a match in the ground truth summary; and the recall-target ($\mathbf{recall}_t$) score: $\mathbf{recall}_t = \mathcal{N}(h \cap t)/\mathcal{N}(t)$, where $\mathcal{N}(t)$ is the number of named-entities in the ground truth summary. We compute the F1 score as $F1_t = 2 \cdot \mathbf{prec}_t \cdot \mathbf{recall}_t / (\mathbf{prec}_t + \mathbf{recall}_t)$.

4 Multi-task learning:

In addition to entity-based data filtering, we also explore another method to further improve the summarization quality. In particular, we incorporate an additional task of classifying summary-worthy named-entities in the source document. A summary-worthy named-entity in the source document is one that appears in the ground truth summary and thus, is a salient entity, worthy of inclusion in the generated summary. Intuitively, if we can identify these summary-worthy named-entities using the encoder representation, we may potentially increase the entity-level precision and recall metrics as well as the overall quality of the summary. We achieve this by adding a classification head to the encoder of BART. To prepare for the classification label, we first identify the named-entities in the ground truth summary and find the matching tokens in the source document. We then assign the (B)eginning-(I)nside-(O)utside labels to each token of the source document to denote if the token is beginning, inside or outside of a summary-worthy named-entity, respectively. During training, we simply add the classification loss for each token at the encoder to the original sequence-to-sequence loss.

More precisely, let $\{(x^i, y^i)\}_{i=1}^N$ be a dataset of N examples where $x^i = x_1^i, \ldots, x_{ts(i)}^i$ are the tokens of the ith source document and $y^i = y_1^i, \ldots, y_{tt(i)}^i$ are the tokens of the target (ground truth summary). The standard sequence-to-sequence training minimizes the maximum log likelihood estimation (MLE) loss:

$$\mathcal{L}_{\text{MLE}}^i(\theta, x^i, y^i) = -\sum_{t=1}^{tt(i)} \log p_\theta(y_t^i | x^i, y_{<t}^i).$$

With summary-worthy entity classification, each example has an additional sequence of BIO labels $z^i = z_1^i, \ldots, z_{ts(i)}^i, z_t^i \in \{0, 1, 2\}$. By adding an additional fully connected layer on top of the BART encoder, we obtain the classification loss

$$\mathcal{L}_{\text{BIO}}^i(\theta(\text{enc}), x^i, z^i) = -\sum_{t=1}^{ts(i)} \log p_{\theta(\text{enc})}(z_t^i | x^i).$$

Finally, we can minimize the joint loss $\mathcal{L}_{\text{Multitask}}^i = \mathcal{L}_{\text{MLE}}^i + \alpha \mathcal{L}_{\text{BIO}}^i$, where α is a hyper parameter. We choose α between 0.1 to 0.5 via the validation sets.

5 Joint Entity and Summary Generation:

We also explore another generative approach to promote entity-level precision and recall metrics. In particular, instead of just generating the summary, we train the BART model to generate the sequence of summary-worthy named-entities, followed by a special token, and then the summary. We call this approach JAENS (Join sAlient ENtity and Summary generation). Similar to the multi-task learning approach discussed earlier, JAENS encourages the model to jointly learn to identify the summary-worthy named-entities while learning to generate summaries. Since the decoder generates the salient named-entities first, the summaries that JAENS generate can further attend to these salient named-entities through decoder self-attention.

6 Experiment results

We use the pre-trained BART-large model in the Fairseq library (Ott et al., 2019) to fine-tune on the 3 summarization datasets.[3] The appendix contains additional details of experimental setup.

In Table 3, we show the effect of the entity-based data filtering. For each dataset, we train two separate models: using the training data before and after entity-based data filtering as shown in Table 2. We evaluate both models on the "clean" test set after entity-based data

---

[3] Our code is available at `https://github.com/amazon-research/fact-check-summarization`

	Newsroom			CNNDM			XSUM		
	train	val	test	train	val	test	train	val	test
original	922,500 (1.58)	100,968 (1.60)	100,933 (1.59)	287,112 (3.90)	13,368 (4.13)	11,490 (3.92)	203,540 (1.0)	11,301 (1.0)	11,299 (1.0)
after filtering	855,975 (1.62)	93,678 (1.64)	93,486 (1.64)	286,791 (3.77)	13,350 (3.99)	11,483 (3.77)	135,155 (1.0)	7,639 (1.0)	7,574 (1.0)

Table 2: Number of examples in three datasets together with the average number of sentences in the ground truth summary (in parentheses) before and after entity-based filtering.

	training data	Rouge1	Rouge2	RougeL	macro $\mathbf{prec}_s$	micro $\mathbf{prec}_s$	macro $\mathbf{prec}_t$	micro $\mathbf{prec}_t$	macro $\mathbf{recall}_t$	micro $\mathbf{recall}_t$	macro $F1_t$	micro $F1_t$
Newsroom	original	47.7±0.2	35.0±0.3	44.1±0.2	97.2±0.1	97.0±0.1	65.4±0.3	62.9±0.4	70.8±0.3	68.5 ±0.2	68.0±0.2	65.6±0.3
	+ filtering	47.7±0.1	35.1±0.1	44.1 ±0.1	98.1±0.1	98.0±0.0	66.5±0.1	63.8±0.1	70.2 ±0.2	67.7±0.3	68.3±0.1	65.7±0.1
	+ classification	47.7±0.2	35.1±0.1	44.2±0.2	98.1±0.1	98.0±0.0	67.2±0.4	64.2±0.4	70.3±0.2	67.8±0.4	68.7±0.3	65.9±0.4
	JAENS	46.6 ±0.5	34.3±0.3	43.2±0.3	**98.3**±0.1	**98.3**±0.1	**69.5**±1.6	**67.3**±1.2	68.9±1.5	66.8±1.6	**69.2**±0.1	**67.0**±0.2
CNNDM	original	43.7±0.1	**21.1**±0.1	40.6±0.1	99.5±0.1	99.4±0.1	66.0±0.4	66.5±0.4	74.7±0.7	75.4±0.6	70.0±0.2	70.7±0.3
	+ filtering	43.4±0.2	20.8±0.1	40.3±0.2	**99.9**±0.0	**99.9**±0.0	66.2 ±0.4	66.6±0.3	74.1±0.6	74.9±0.6	69.9±0.2	70.5±0.2
	+ classification	43.5±0.2	20.8±0.2	40.4±0.2	**99.9**±0.0	**99.9**±0.0	**67.0**±0.6	**67.5**±0.5	74.7±0.2	75.5±0.1	70.6±0.3	71.3±0.3
	JAENS	42.4 ±0.6	20.2±0.2	39.5±0.5	**99.9**±0.0	**99.9**±0.0	**67.9**±0.7	**68.4**±0.6	**75.1**±0.7	**76.4**±0.7	**71.3**±0.2	**72.2**±0.2
XSUM	original	**45.6**±0.1	**22.5**±0.1	**37.2**±0.1	93.9±0.1	93.6±0.2	74.1±0.2	73.3±0.2	80.1±0.1	80.3±0.3	77.0±0.1	76.6±0.2
	+ filtering	45.4±0.1	22.2±0.1	36.9±0.1	98.2±0.0	98.2±0.1	77.9±0.2	77.3±0.2	79.4±0.2	79.6±0.2	78.6±0.1	78.4±0.2
	+ classification	45.3±0.1	22.1±0.0	36.9±0.1	98.3±0.1	98.2±0.1	78.6±0.3	**78.0**±0.3	79.5±0.3	79.8±0.4	**79.1**±0.1	**78.9**±0.1
	JAENS	43.4±0.7	21.0±0.3	35.5 ±0.4	**99.0**±0.1	**99.0**±0.1	77.6±0.9	77.1±0.6	79.5±0.6	80.0±0.5	78.5±0.2	78.5±0.1

Table 3: Comparison of models trained using original data, with entity-based data filtering, with an additional classification task and with JAENS. Scores are all in percentages, averaged over 5 runs and shown with standard deviations. We bold the numbers that are significantly better in the sense that the means are separated by at least the standard deviations. We report both the micro and macro averages of our proposed entity-level scores. In all datasets, data filtering leads to higher $\mathbf{prec}_s$ scores, indicating that entity hallucination can be alleviated by this simple technique. In addition, data filtering generally improves other entity level metrics: $\mathbf{prec}_t$, $\mathbf{recall}_t$ and $F1_t$. Adding the classification task (multi-task) or JAENS to data filtering further improves the performance on $\mathbf{prec}_t$ and $\mathbf{recall}_t$ and therefore the overall entity-level $F1_t$.

filtering. We choose this filtered version of the original test set because we only want to measure entity-level consistency against the correct set of entities; using the unfiltered dataset means we could count a hallucinated entity as correct. We observe improvements of $\mathbf{prec}_s$ across all three datasets trained using the filtered subset of data. For example in XSUM, the $\mathbf{prec}_s$ is increased from 93.6% to 98.2%, indicating a significant reduction in entity hallucination. In addition, the entity-based data filtering generally improves other entity-level metrics as well. Even with less training data, the entity-based data filtering is able to maintain the ROUGE scores quite well. For XSUM, about 34% of the training data is filtered out (c.f. Table 2), which explains the more noticable impact on the ROUGE scores. The results in Table 3 suggest that entity-level data filtering is a simple yet effective approach to achieve higher entity-level factual consistency as well as general summarization quality. In Table 4 we provide qualitative examples where the model trained on the original data produces hallucination and the entity-level data filtering removes such hallucination.

Table 3 shows that adding the classification task (multi-task) futher increases the $\mathbf{prec}_t$ and $\mathbf{recall}_t$ metric and therefore the overall entity-level $F1_t$ on top of the improvements from data filtering. Similar gains can be observed with JAENS, which out-performs the multi-task approach on CNNDM and Newsroom datasets. The result confirms our intuition that the summaries in JAENS can benefit from attending to the generated salient entities in terms of the entity level metrics. However, the additional complexity during decoding may have hurt the ROUGE scores.

For the interested readers, we also evaluated the PEGASUS (Zhang et al., 2020) models for the ROUGE and entity level metrics on these three datasets in the appendix.

Accuracy of entity level metrics: As our entity level metrics are based on automatic NER tools and heuristics matching rules, errors in both steps can lead to inaccuracy in the metrics. By manually checking 10 random ground truth summaries together with the source documents in the validation split of XSUM dataset, we found that all of the named entities are correctly identified by the NER tool and the matchings are correct. Therefore, we believe that even our current NER tool and matching rule already produce high

Before data filtering	After data filtering	With classification	Ground truth summary
People in Italy and the Netherlands are more likely to consume fewer cups of coffee than those in the UK, a study suggests.	The desire to drink coffee may be encoded in our DNA, according to scientists.	People with a particular gene are more likely to consume fewer cups of coffee, a study has suggested.	Researchers have identified a gene that appears to curb coffee consumption.
A cathedral in Surrey is set to be restored after more than £5m was raised to pay for repairs and improvements.	A £7m project to save a Grade II-listed cathedral from demolition is set to go ahead.	A cathedral which has been threatened with demolition is set to be saved by a £5m fundraising campaign.	A 1960s-built cathedral that was ”at serious risk of closure” has raised more than 90% of its £7m target for urgent repairs and development.
More than 800,000 chemists in the Indian capital, Delhi, have gone on strike in protest against online drug sales.	More than 800,000 chemists in India will go on strike on Wednesday to protest against illegal online drug sales.	More than 800,000 chemists in India are set to go on strike on Wednesday in a row over the sale of drugs online.	At least 800,000 pharmacies in India are on a one-day strike, demanding an end to online drug sales which they say is affecting their business.
Police officers in Pembrokeshire are to be issued with body-worn cameras.	Police officers in Powys are to be issued with body-worn cameras in a bid to improve transparency in the force.	Police officers in Powys are to be issued with body cameras in a bid to improve transparency in the force.	A police force has begun the rollout of body cameras for 800 officers and community support officers.
Wales midfielder Becky Lawrence has been speaking to BBC Sport about her time as a player-manager with Melbourne City.	It's been a great few weeks for me as a player-manager and now I'm heading home to Wales ahead of the Cyprus Cup.	It's been a very busy few weeks for me as I'm heading home to Wales ahead of the Cyprus Cup.	I have certainly had worse 24 hours in my life than winning the Grand Final with Melbourne City and then being named in the Wales squad for the Cyprus Cup.

Table 4: Generated and ground truth summary examples from the test set of XSUM. The first three columns are generated from the model trained without entity-based data filtering, with entity-based data filtering and with the additional classification task, respectively. The right column contains the ground truth summaries. The hallucinated named-entities are underscored. Proposed data filtering overcomes hallucination in these examples.

accuracy in practice.

7 Conclusion

In this paper we study the entity-level factual consistency of the state-of-the-art summarization model. We propose precision-source score $\mathbf{prec}_s$ to quantify the degree of entity hallucination. We also propose additional metrics $\mathbf{prec}_t$ and $\mathbf{recall}_t$ to measure entity level accuracy of the generated summary with respect to the ground truth summary. We found that the ground truth summaries of the XSUM dataset contain a high level of entity hallucination. We propose a simple entity-level data filtering technique to remove such hallucination in the training data. Experiments show that such data filtering leads to significant improvement in $\mathbf{prec}_s$. ($\mathbf{prec}_s$ increases from below 94% to above 98% in XSUM for example.) We futher proposed a multi-task learning and a joint sequence generation approach to further improve the entity-level metrics. Overall, combining our proposed approaches significantly reduces entity hallucination and leads to higher entity level metrics with minimal degradation of the ROUGE scores.

References

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization.

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

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Sebastian Gehrmann, Zachary Ziegler, and Alexander Rush. 2019. Generating abstractive summaries with finetuned language models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 516–522, Tokyo, Japan. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2019. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. *CoRR*, abs/1911.03860.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

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

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

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

A Supplementary material for Entity-level Factual Consistency of Abstractive Text Summarization

A.1 Details of NER filtering

We only consider named-entities of the following types: ’PERSON’ (People, including fictional.), ’FAC’ (Buildings, airports, highways, bridges, etc.), ’GPE’ (Countries, cities, states.), ’ORG’ (Companies, agencies, institutions, etc.), ’NORP’ (Nationalities or religious or political groups.), ’LOC’ (Non-GPE locations, mountain ranges, bodies of water.), ’EVENT’ (Named hurricanes, battles, wars, sports events, etc.). We ignore other types of entities such as date, time, numerals because they tend to have large variations in representation and are difficult to determine a match in the source document.

A.2 Details of experimental setup

We use the pre-trained BART-large model in the Fairseq library (Ott et al., 2019) to fine-tune on the 3 summarization datasets.

In all experiments, we validate the ROUGE scores of the generated summaries on the validation split and early-stop on the epoch with the highest validation score. We use the standard learning rate of 3e-5 for finetuning with linear decay schedule and 500 warmup steps. For Newsroom, we use 4 p3.16xlarge EC2 instances on AWS with a total of 32 Tesla V100 GPUs for finetuning and the effective batch size is 32; for XSUM, we use 1 p3.16xlarge instance with a total of 8 Tesla V100 GPUs and update frequency of 4, giving an effective batch size of 32; for CNNDM, we use 1 p3.16xlarge instance with a total of 8 Tesla V100 GPUs, giving an effective batch size of 8.

We chose the α parameter for multi-task learning between 0.1 and 0.5 with step of 0.05 based on ROUGE scores on the validation set. We found the best values are 0.3, 0.3 and 0.15 for Newsroom, CNNDM and XSUM, respectively. We observe that the ROUGE and entity level metrics on validation and test sets are very close, with the former slightly higher.

During decoding, we use beam size of 1 for Newsroom, 4 for CNNDM and 6 for XSUM (to be consistent with the setting in (Lewis et al., 2019)). We did use trigrams blocking in beam search as we did not see much need for this additional step.

A.3 Evaluation of PEGASUS (Zhang et al., 2020)

In this section we simply evaluate the PEGASUS checkpoints provided by Huggingface (Wolf et al., 2020) on the NER filtered test sets. The checkpoints are downloaded from `https://huggingface.co/google/pegasus-newsroom`, `https://huggingface.co/google/pegasus-cnn_dailymail` and `https://huggingface.co/google/pegasus-xsum`, respectively. The results are summarized in Table 5. Note that PEGASUS performances similarly on CNNDM and XSUM but worse on Newsroom compared to BART-large.

	Rouge1	Rouge2	RougeL	macro $\mathbf{prec}_s$	micro $\mathbf{prec}_s$	macro $\mathbf{prec}_t$	micro $\mathbf{prec}_t$	macro $\mathbf{recall}_t$	micro $\mathbf{recall}_t$	macro $F1_t$	micro $F1_t$
Newsroom	40.6	28.4	37.4	94.6	94.7	53.4	55.5	68.5	67.8	60.0	61.1
CNNDM	42.5	20.7	39.6	99.1	99.0	65.9	66.7	74.7	75.7	70.0	70.9
XSUM	45.3	23.7	37.9	93.9	93.1	76.6	75.8	80.3	80.1	78.4	77.9

Table 5: Evaluation of PEGASUS on NER filtered test sets.

On Hallucination and Predictive Uncertainty in Conditional Language Generation

Yijun Xiao, William Yang Wang
University of California, Santa Barbara
{yijunxiao,william}@cs.ucsb.edu

Abstract

Despite improvements in performances on different natural language generation tasks, deep neural models are prone to hallucinating facts that are incorrect or nonexistent. Different hypotheses are proposed and examined separately for different tasks, but no systematic explanations are available across these tasks. In this study, we draw connections between hallucinations and predictive uncertainty in conditional language generation. We investigate their relationship in both image captioning and data-to-text generation and propose a simple extension to beam search to reduce hallucination. Our analysis shows that higher predictive uncertainty corresponds to a higher chance of hallucination. Epistemic uncertainty is more indicative of hallucination than aleatoric or total uncertainties. It helps to achieve better results of trading performance in standard metric for less hallucination with the proposed beam search variant.

1 Introduction

Modern deep neural network models have brought drastic improvements of generation quality measured by standard metrics on different natural language generation (NLG) tasks. However, along with these improvements, researchers find that neural models are more prone to a phenomenon called hallucination, where models generate description tokens that are not supported by the source inputs. This phenomenon seriously damages the applicability of neural language generation models in practice where information accuracy is vital.

Hallucination has been observed in various conditional NLG tasks such as image captioning (Rohrbach et al., 2018), data-to-text generation (Wiseman et al., 2017; Nie et al., 2019; Parikh et al., 2020), abstractive summarization (Cao et al., 2018; Durmus et al., 2020), and neural machine translation (NMT) (Müller et al., 2019). These studies tackle hallucinations within a specific task and give possible explanations of why hallucinations occur. For example, Rohrbach et al. (2018) attributes object hallucination in image captioning to visual misclassification and over-reliance on language priors; Nie et al. (2019) believes hallucination in neural surface realization comes from the misalignment between meaning representations and their corresponding references in the dataset; Müller et al. (2019) claims that hallucinations in NMT are mainly due to domain shift.

We believe that there is a common theme across all the hallucination explanations in conditional NLG tasks: predictive uncertainty. In language generation, predictive uncertainty quantifies the entropy of the token probability distributions a model predicts. There are multiple sources of uncertainty. Two major ones frequently studied are aleatoric and epistemic uncertainties, where the former comes from the data or measurements, and the latter is concerned with the model. With recent progress in Bayesian neural networks (BNNs) (Hinton and Van Camp, 1993; Neal, 1995) and uncertainty quantification (Blundell et al., 2015; Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017), we are able to quantify both parts of predictive uncertainty in neural NLG.

This study draws connections between hallucination and predictive uncertainty and empirically investigates their relationship in image captioning and data-to-text generation tasks. We propose an uncertainty-aware beam search algorithm to reduce the chance of hallucination by penalizing parts or the entirety of the predictive uncertainty during model decoding. We find that the choice of uncertainty matters, and penalizing epistemic uncertainty yields better results compared to penalizing aleatoric or total uncertainty. Our contributions are:

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2734–2744
April 19 - 23, 2021.

- We draw connections between hallucination and predictive uncertainty across various conditional natural language generation tasks and empirically investigate their relationship.
- We propose an uncertainty-aware beam search approach for hallucination reduction to demonstrate that lowering uncertainty can lead to less hallucination.
- We show that uncertainty decomposition helps to achieve better trade-offs between hallucination and performance.

2 Hallucination and Predictive Uncertainty

2.1 Hallucination Probability

In general, hallucination refers to the phenomenon where the model generates false information not supported by the input. For example, in the context of image captioning, hallucination can be defined as generating captions that contain descriptions not present in the given image. Let (x, y) be the pair of variables at interest where x is some structured data containing facts and y is a natural language sentence based on the facts. The task is to learn the conditional distribution of $p(y|x)$ in order to generate sentence y given any new input x. Most neural approaches break the probability into a sequence of single token predictions:

$$p(y|x) = p(y_1|x) \prod_{i=2}^{k} p(y_i|x, y_1, \cdots, y_{i-1}) \quad (1)$$

where $\{y_1, \cdots, y_k\}$ is the collection of tokens in sentence y. We denote $c_i = \{x, y_1, \cdots, y_{i-1}\}$ as the context of the i-th prediction in the following sections for simplicity.

Apparently, hallucination is context-dependent which means we need to look at a certain context c_i and determine whether the next token prediction y_i is hallucinated or not. Let $\mathcal{V}_h^{(c_i)}$ denote the set of tokens that are considered false information given the current context c_i and $\mathcal{V}$ the whole vocabulary. Consider a random sampling decoder where a token is generated based on the predicted categorical distribution. i.e. $\text{Cat}(|\mathcal{V}|, p(y_i|c_i))$. The probability of hallucination at the current step is simply:

$$P(y_i \in \mathcal{V}_h^{(c_i)}) = \sum_{v \in \mathcal{V}_h^{(c_i)}} p(y_i = v|c_i) \quad (2)$$

Practically, it is hard to automatically determine the context-dependent set $\mathcal{V}_h^{(c_i)}$. Task-specific heuristics are often used to determine which tokens are hallucinated. In specific restrictive applications, the context-dependent set can be relaxed to a context-independent one to reduce the complexity of determining hallucination.

2.2 Relationship with Predictive Uncertainty

We use entropy to measure the predictive uncertainty in this work. The total uncertainty of predicting token y_i is:

$$\begin{aligned} & H(y_i|c_i) \\ &= -\sum_{v \in \mathcal{V}} p(y_i = v|c_i) \log p(y_i = v|c_i) \\ &= -\sum_{v \in \mathcal{V} \setminus \mathcal{V}_h^{(c_i)}} p(y_i = v|c_i) \log p(y_i = v|c_i) \\ &\quad - \sum_{v \in \mathcal{V}_h^{(c_i)}} p(y_i = v|c_i) \log p(y_i = v|c_i) \end{aligned} \quad (3)$$

From Equation 3, we can see that there are two sources of uncertainty for the token predictions: one from the uncertainty of choosing suitable tokens to describe the input; another from some unsuitable tokens attaining considerable probability mass either by being confusing in the current context or due to an insufficiently trained system.

The second source of uncertainty is directly related to hallucination probability. Although no monotonic relationship can be derived, a near-zero hallucination probability requires a near-zero value of the second source of uncertainty. This observation prompts us to investigate the relationship between hallucination and predictive uncertainty in practice. Intuitively, the higher the predictive uncertainty is, the more probable some of the probability mass gets assigned to unsuitable tokens.

2.3 Uncertainty Decomposition

There are often two types of uncertainties frequently mentioned in uncertainty quantification literature: epistemic and aleatoric uncertainty (Der Kiureghian and Ditlevsen, 2009; Kendall and Gal, 2017; Depeweg et al., 2018). Epistemic uncertainty reflects the uncertainty on model weights, and aleatoric uncertainty concerns inherent uncertainty in the data or measurement. We are interested in whether the relationship with hallucination is the same for both types of uncertainties.

$$\begin{array}{l|ccc|} & \text{Token 1} & \text{Token 2} & \text{Token 3} \\ \text{Prediction 1} & 0.33 & 0.33 & 0.33 \\ \text{Prediction 2} & 0.33 & 0.33 & 0.33 \\ \text{Prediction 3} & 0.33 & 0.33 & 0.33 \end{array}$$

(a)

$$\begin{array}{l|ccc|} & \text{Token 1} & \text{Token 2} & \text{Token 3} \\ \text{Prediction 1} & 0.98 & 0.01 & 0.01 \\ \text{Prediction 2} & 0.01 & 0.98 & 0.01 \\ \text{Prediction 3} & 0.01 & 0.01 & 0.98 \end{array}$$

(b)

Figure 1: Examples of predictions with (a) high aleatoric but low epistemic uncertainty; and (b) high epistemic but low aleatoric uncertainty.

Bayesian deep learning approaches (Blundell et al., 2015; Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017) are widely studied for uncertainty quantification with neural networks. Following the notations in Section 2.2, the predictive distribution of $p(y_i|c_i)$ can be written as:

$$p(y_i|c_i) = \int_w p(y_i|c_i, w)q(w)dw \tag{4}$$

where w parameterizes the neural network that makes predictions and $q(w)$ denotes the approximate posterior distribution of the weights w given the training data. Notice that if we fix the weights w, $H(y_i|c_i, w)$ represents the entropy that is unrelated to the uncertainty of the model weights. Therefore the aleatoric part of the predictive uncertainty can be calculated with $\mathbb{E}_{q(w)}[H(y_i|c_i, w)]$. The epistemic part of the uncertainty is the difference between the total and the aleatoric uncertainty as shown below:

$$u_{al}(y_i|c_i) = \mathbb{E}_{q(w)}[H(y_i|c_i, w)] \tag{5}$$

$$u_{ep}(y_i|c_i) = H(y_i|c_i) - \mathbb{E}_{q(w)}[H(y_i|c_i, w)] \tag{6}$$

In this study, the aleatoric and epistemic parts of predictive uncertainty are estimated using deep ensembles (Lakshminarayanan et al., 2017). More concretely, denote the model predictions as $\{p_m(y_i|c_i)\}_{m=1}^M$ and the aggregated prediction as $p(y_i|c_i) = \frac{1}{M}\sum_{m=1}^M p_m(y_i = v|c_i)$, aleatoric and epistemic uncertainties are calculated as:

$$u_{al}(y_i|c_i) = \frac{1}{M}\sum_{m=1}^{M} H_m(y_i|c_i) \tag{7}$$

$$u_{ep}(y_i|c_i) = H(y_i|c_i) - u_{al}(y_i|c_i) \tag{8}$$

where $H_m(y_i|c_i)$ and $H(y_i|c_i)$ are the entropy of $p_m(y_i|c_i)$ and $p(y_i|c_i)$ respectively.

Intuitively, in the case of deep ensembles, aleatoric uncertainty measures the average spread of all model predictions, while epistemic uncertainty measures the agreement among all model predictions. Examples with three possible tokens are illustrated in Figure 1.

3 Case Study: Image Captioning

In this section, we analyze image captioning models trained on MSCOCO (Chen et al., 2015) data set.

3.1 Hallucination Probability at Different Uncertainty Levels

The first question we want to investigate is whether hallucination probabilities change at different predictive uncertainty levels. Some experimental settings are listed below.

Model architecture We consider four different image captioning models: **FC** model (Rennie et al., 2017) where image features are used to initialize the RNN decoder; **Att2In** model from (Rennie et al., 2017) applies attention on image features and feeds it into the decoder LSTM (Hochreiter and Schmidhuber, 1997) cell gate; **BUTD** model from (Anderson et al., 2018) uses bottom-up attention which operates at the level of objects and other salient image regions; **Transformer** model where transformers (Vaswani et al., 2017) are used in the encoder-decoder structure for generation. All models are implemented in the open source framework by Luo et al. (2018)[1].

Training We consider the same data split from (Karpathy and Fei-Fei, 2015). All models are trained with batch size 50 for 30 epochs with Adam optimizer (Kingma and Ba, 2014). Evaluations are done on the Karpathy test set.

Hallucination and uncertainty evaluation As in (Rohrbach et al., 2018), synonyms for all possible MSCOCO objects are used to determine

---

[1] https://github.com/ruotianluo/self-critical.pytorch

Model	Action hallucination % at uncertainty level					
	≤ 0.8	0.8 - 1.6	1.6 - 2.4	2.4 - 3.2	3.2 - 4.0	> 4.0
FC	0.00	0.00	2.27	12.86	15.71	31.03
Att2In	0.00	0.00	3.39	6.58	12.07	22.03
BUTD	0.00	2.94	1.92	12.77	17.24	25.53
Transformer	2.99	5.48	6.58	8.82	12.00	43.75

Table 1: Action hallucination percentages at different levels of predictive uncertainty. Action predictions with higher uncertainty are more prone to hallucination.

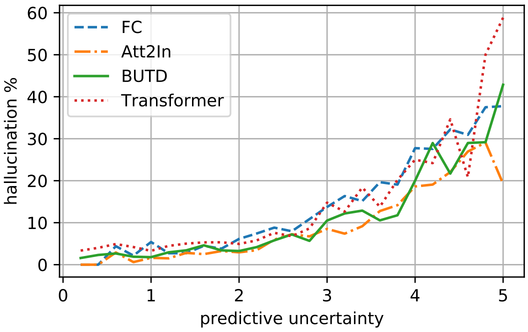


Figure 2: Object hallucination chance at different predictive uncertainty levels. Higher predictive uncertainty corresponds to a higher level of hallucination percentage across all models.

whether an object generated by the captioning model is hallucinated. Hallucination probabilities are calculated by binning all object token prediction entropy and counting the percentage of hallucinated objects in each bin.

3.2 Results and Discussions

Figure 2 shows the object hallucination percentages at different predictive uncertainty levels. At higher uncertainty levels, the generated objects are more likely to be hallucinated. The results are consistent across four different models. The transformer model seems to have a higher hallucination chance at high uncertainty levels than the other three models. However, this does not indicate Transformer models hallucinate more. In fact, the transformer model has an overall lowest hallucination percentage among all four models.

Beyond object hallucination Aside from object hallucination, we also analyze verbs generated by the models to see whether a similar relationship holds for other types of token generations. The same models and training procedures are adopted. We extract all present continuous tense verbs from the generated captions using spaCy part-of-speech tagger[2] and manually label whether they are suitable to describe the corresponding images. There are approximately 3500 generated captions containing verbs, and 400 are annotated for each model. We refer to unsuitable verbs generated in the captions as action hallucinations.

Model	Correlation coefficient	
	epistemic	aleatoric
FC	0.313	0.299
BUTD	0.334	0.228
Att2In	0.360	0.268
Transformer	0.269	0.131

Table 2: Pearson correlation coefficients between hallucination and epistemic/aleatoric uncertainty in image captioning task. Epistemic uncertainty is more indicative of hallucination across four models.

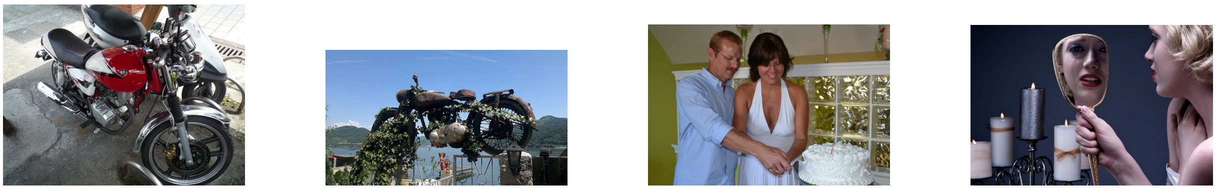
Action predictions are binned according to their uncertainty values, and the results are shown in Table 1. We can observe that action tokens with higher predictive uncertainty are also more likely to be hallucinated. Noticeably, the transformer model also has a higher action hallucination rate at high uncertainty levels.

Examples of predictions with high and low uncertainty Figure 3 shows some example images and their captions generated from a BUTD model on the test set. The token predictions of interests and the corresponding uncertainty values are highlighted in bold and italic, respectively. We observe that highly uncertain predictions often correspond to unusual textures, features resembling the predicted tokens, or blurred images. For example, Figure 3(b) shows a motorcycle covered in vines; Figure 3(d) shows candles in the background which resemble cakes; Figure 3(f) is blurred.

Epistemic and aleatoric uncertainties As we could decompose the total uncertainty into two parts, we are interested in which part is more indicative of hallucination. Table 2 shows the Pear-

---

[2] https://spacy.io

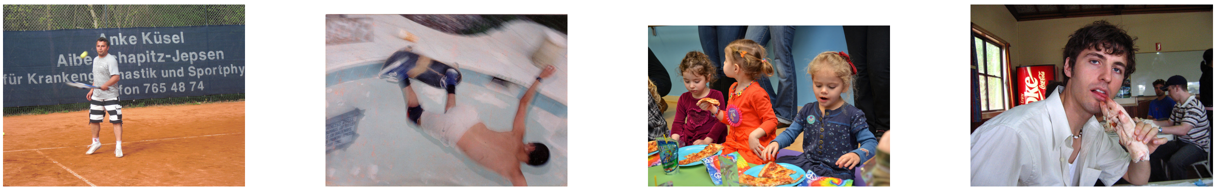


(a) a red and black **motorcycle** (*0.58*) parked in a parking lot

(b) a **motorcycle** (*4.80*) is parked on a dock with a bird perched on top of it

(c) a bride and groom cutting their wedding **cake** (*0.09*)

(d) a woman holding a cup and a **cake** (*5.29*)



(e) a man standing on a tennis court **holding** (*0.81*) a racquet

(f) a young man is **holding** (*4.76*) a skateboard in his hand

(g) a group of children sitting at a table **eating** (*1.00*) pizza

(h) a man is **eating** (*4.01*) a hot dog at a restaurant

Figure 3: Examples of token predictions generated with the BUTD model with high and low uncertainty values for objects (top) and actions (bottom). Numbers in italic are predictive uncertainty values for the token predictions preceding them. The examples are cherry-picked.

son correlation coefficients between hallucination (binary) and epistemic/aleatoric uncertainty for all four models. We can see that both parts of uncertainty are weakly correlated with hallucination, while epistemic uncertainty is more indicative of hallucination across all four models compared to aleatoric uncertainty.

4 Case Study: Data-to-text Generation

Data-to-text generation (Kukich, 1983; McKeown, 1992) is a task to generate textual content conditioned on input content in the form of structured data such as tables. Neural models are prone to hallucination in data-to-text generation tasks compared to traditional template-based systems, and methods are proposed to improve faithfulness (Wiseman et al., 2017; Nie et al., 2019; Tian et al., 2019). In this section, we discuss the relationship between predictive uncertainty and hallucination in data-to-text generation with ToTTo dataset (Parikh et al., 2020).

4.1 Generation Quality and Average Uncertainty

We conduct token-level analysis in Section 3. Now we take a different route and analyze sentence-level quality with different average predictive uncertainty values. Experiment settings are described below.

Dataset ToTTo dataset consists of tables from English Wikipedia articles with their corresponding metadata, such as page title and section title. Candidate description texts are modified by annotators to pair with each table. Relevant table cells supporting the description texts are highlighted by the annotators as well. There are 120,761 table-text pairs in training, 7,700 in validation, and 7,700 in test. We use the baseline standard linearization approach to represent the highlighted portions of the tables along with their corresponding metadata (referred to as subtable with metadata in (Parikh et al., 2020)).

Model architecture and training We use a standard sequence-to-sequence model with attention (Bahdanau et al., 2015; Luo et al., 2018) for analysis. LSTM with 512 hidden size is used for both the encoder and the decoder. Adam optimizer with learning rate 1e-3 is used for the optimization. The model is trained with cross-entropy loss for 20 epochs. The checkpoint with the best validation loss is chosen for the evaluation. The implementation is done using fairseq (Ott et al., 2019)[3].

Evaluation We evaluate the average predictive uncertainty for all generated sentences in the validation set and select the top, bottom, and middle 5% for comparison. BLEU score (Papineni et al., 2002) is used as an automatic metric to evaluate the similarity to the references; further manual annotations are done to evaluate the *fluency*, *faithfulness* (precision), and *coverage with respect to reference*

---

[3] https://github.com/pytorch/fairseq

Unc. Level	Avg Unc.	BLEU	Fluency (%)	Faithfulness (%)	Less/Neutral/More Coverage w.r.t. Ref
High	1.83 - 3.74	10.2	46.0	41.3	79.4 / 15.9 / 04.7
Medium	0.83 - 0.89	31.5	87.3	78.9	35.2 / 47.9 / 16.9
Low	0.04 - 0.27	72.8	100.0	99.0	22.2 / 70.1 / 07.7

Table 3: Evaluation results for candidates with high, medium, and low average predictive uncertainty values for ToTTo validation set. Unc. denotes uncertainty. Higher uncertainty candidates have lower quality and higher chance of being hallucinated/unfaithful w.r.t. the input tables.

(recall) of the generated sentences. Particularly, faithfulness reflects how likely the generated sentences hallucinate facts that are not supported by the tables. More details of the human evaluation metrics are described in (Parikh et al., 2020). The goal is to measure how different the generation qualities are for candidates with varying average predictive uncertainties.

4.2 Results and Discussions

Table 3 summarizes the evaluation results for candidates with varying uncertainty values. It is obvious that candidates with higher average predictive uncertainty values are less fluent and more likely to contain hallucinations. Another interesting observation from Table 3 is that the generated sentences with medium average uncertainty are more likely (16.9%) to cover more table facts than the references compared to the ones with high (4.7%) and low (7.7%) average uncertainty. One possible explanation is that some table facts that are not always included in the references, when generated, have higher predictive uncertainty values than the facts that are almost always included in the references. Therefore, generated sentences with low uncertainty tend to include less but more confident facts considered by the model.

5 Reducing Hallucination

5.1 Uncertainty-Aware Beam Search

Because of the positive correlation between hallucination probability and predictive uncertainty, it is straightforward to incorporate uncertainty into the caption generation process to reduce hallucination. Beam search is the most used approximate decoding method in language generation. It keeps track of the top-B scored candidates at each generation step and considers all single token extensions of the current candidates.

More formally, denote the set of B candidates in the beam at time step $t-1$ as $\mathcal{Y}_{t-1} = \{\mathbf{y}_{t-1}^{(b)}\}_{b=1}^{B}$. All possible single token extensions of the candidates in $\mathcal{Y}_{t-1}$ form a set $\mathcal{C}_t = \{\mathbf{y} \mid \mathbf{y}_{t-1} \in \mathcal{Y}_{t-1} \wedge y_t \in \mathcal{V}\}$. Beam at step t is then formed as:

$$\mathcal{Y}_t = \underset{\mathbf{y}_1 \cdots \mathbf{y}_B \in \mathcal{C}_t}{\arg\max} \sum_{b=1}^{B} \log p(\mathbf{y}_b|\mathbf{x}) \quad s.t. \quad \mathbf{y}_i \neq \mathbf{y}_j \quad \forall i \neq j \tag{9}$$

Uncertainty-aware beam search (UABS) adds a weighted penalty term in the beam search objective to balance between log probability and predictive uncertainty of the selected candidates. Let $u(\mathbf{y}|\mathbf{x})$ be the function to measure the aggregated predictive uncertainty of candidate $\mathbf{y}$ given input $\mathbf{x}$, uncertainty-aware beam search updates the beam at step t according to the following equation:

$$\mathcal{Y}_t = \underset{\mathbf{y}_1 \cdots \mathbf{y}_B \in \mathcal{C}_t}{\arg\max} \sum_{b=1}^{B} \log p(\mathbf{y}_b|\mathbf{x}) - \lambda u(\mathbf{y}_b|\mathbf{x}) \quad s.t. \quad \mathbf{y}_i \neq \mathbf{y}_j \quad \forall i \neq j \tag{10}$$

where $\lambda \geq 0$ is the weight controlling the degree to which we want to penalize decoding uncertainty. Larger λ leads to candidates with smaller predictive uncertainty. In practice, this can be done by subtracting the weighted uncertainty term from the aggregated log probability scores at each decoding step before choosing top-B candidates.

An important decision in using uncertainty-aware beam search is the choice of uncertainty term $u(\mathbf{y}|\mathbf{x})$. We could use either the aleatoric or epistemic part of the predictive uncertainty or both. We compare these choices and discuss the results in the next section.

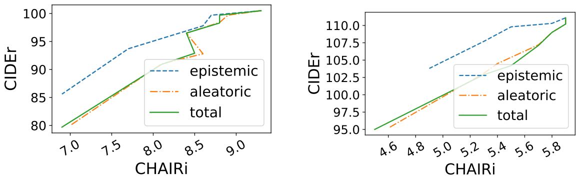
5.2 Image Captioning Results

With larger weights on the uncertainty penalty term, log probabilities of the decoded sentences drop. Therefore, we expect to see a trade-off between the quality of generated captions and the chance of hallucination.

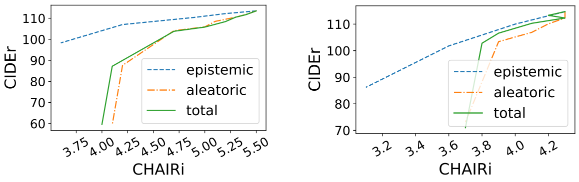
We empirically examine the trade-offs on the image captioning models with different uncertainty

Image	**UABS results with weight λ**		
	0	20	80
	a vase filled with flowers sitting on top of a table	a vase filled with lots of white flowers	there is a vase that has flowers in it
	a wooden cutting board topped with lots of food	a wooden cutting board topped with lots of food	a cutting board that has a bunch on it

Table 4: Two examples of epistemic UABS results with varying penalty weights on the image captioning data set. In the first example the model successfully avoids hallucination of a table with $\lambda = 20$ while in the second example it is unable to change the generated caption until larger penalty weight is set.



(a) FC (b) Att2In



(c) BUTD (d) Transformer

Figure 4: CIDEr plotted against CHAIRi scores of captions generated with UABS with different uncertainty penalty weights. Lower CHAIRi score indicates less hallucination. Upper-left is better. Penalizing epistemic uncertainty in UABS achieves the best results.

choices for the penalty term. We use a five-model ensemble for each of the four model architectures to estimate aleatoric and epistemic uncertainties. Due to the different magnitudes of aleatoric and epistemic uncertainties, we choose penalty weight λ from $[0.1, 0.2, 0.4, 0.8, 1.0, 2.0, 4.0]$ for aleatoric and total uncertainty and $[10, 20, 40, 80]$ for epistemic uncertainty.

Figure 4 shows the trade-offs between CIDEr (Vedantam et al., 2015) and CHAIRi (Rohrbach et al., 2018) scores of captions generated with uncertainty-aware beam search with different uncertainty choices and penalty weights. A smaller value of CHAIRi indicates the model is less likely to generate hallucinated objects, and a higher CIDEr indicates better caption quality. Therefore

	λ	**avg. len.**	**# obj.**	**hal. %**	**gen. %**
ref.		10.44	6114	0	-
base	0	9.31	7328	5.5	0
epist.	10	9.21	7195	5.2	0
	20	9.16	7078	4.9	0.2
	40	9.15	6912	4.2	1.5
	80	9.12	6493	3.6	4.6
aleat.	0.1	9.32	7250	5.4	0
	0.4	9.32	7051	5.1	0
	1.0	9.33	6800	4.7	1.0
	4.0	9.43	4349	4.1	28.4

Table 5: Average sentence length and total number of objects detected in the captions generated by BUTD model with varying uncertainty penalty weight λ. Penalizing epistemic uncertainty leads to slightly shorter lengths. Number of objects mentioned by the captions decreases with increasing λ. **gen. %** denotes percentage of generic responses. It is moderate with epistemic penalized results but can be very high if aleatoric uncertainty is heavily penalized.

an approach that is to the upper left of another is better. As the penalty weight increases, we observe a decrease in both the CHAIRi and the CIDEr scores across all models.

Table 4 shows two examples of different generated captions using epistemic UABS with varying penalty weights. In the first example, we can see that a medium penalty weight of 20 not only helps avoid the hallucination of a table but also adds correct information about the color of the flowers. In the second example, a medium penalty weight is unable to change the generated caption.

Regarding the choice of uncertainty, it is notable that when penalizing epistemic uncertainty, the generated captions achieve higher CIDEr scores than penalizing aleatoric or total uncertainty. We

λ	BLEU	Fluency (%)	Faithfulness (%)	Less/Neutral/More Coverage w.r.t. Ref
0	40.1	92	79	34 / 60 / 6
10	33.6	83	84	41 / 51 / 8
20	27.4	73	80	52 / 42 / 6

Table 6: Evaluation results for candidates decoded with different penalty weights for UABS on ToTTo validation set. Epistemic uncertainty is used for uncertainty penalization. Faithfulness first increases, then decreases to the same level as regular beam search results as we increase the penalty weight λ.

Reference	**UABS results with weight λ**		
	0	10	20
barrows scored 164 net points in virgin islands at the 2008 summer olympics.	in virgin islands at the 2008 summer olympics, barrows iii received 164 points.	in virgin islands at the 2008 summer olympics, barrows received 164 points.	thomas barrows received a total score of 164.
janet gaynor won the first academy award for best actress for her performance in the 7th heaven (1927 film).	janet gaynor won the academy award for best actress for his performance in janet gaynor.	janet gaynor won the academy award for best actress.	janet gaynor won an academy award for best actress.

Table 7: Two examples of UABS results with varying penalty weights on the ToTTo validation set. Blue tokens are correct table facts that are dropped by candidates generated with larger penalty weights; red tokens are incorrect/hallucinated facts that are dropped with larger penalty weights. In general, UABS with larger weights tend to produce sentences with less information that the model is more confident with.

hypothesize that epistemic uncertainty indicates the uncertainty of model weights. By penalizing epistemic uncertainty, we encourage the model to take the prediction path where it is well-calibrated. On the other hand, penalizing aleatoric uncertainty encourages the model to make low entropy predictions in all contexts regardless of the actual data distributions.

Table 5 shows the average sentence length, the number of objects, the percentage of hallucinations, and the percentage of generic responses in the captions generated by the BUTD model with different uncertainty choices and penalty weights on the test set. We can see that when penalizing epistemic uncertainty, UABS results in slightly shorter caption candidates. Both the number of objects and hallucination percentage decrease as we increase the weight λ. Interestingly, when penalizing aleatoric uncertainty, sentence length stays approximately the same despite lower CIDEr scores, as shown in Figure 4. Further investigation shows that this is partly due to an increasing number of generic captions such as "*there is no image here to provide a caption for*". Penalizing epistemic uncertainty is much less likely to result in such generic captions. We can see that when increasing λ from 1.0 to 4.0 with aleatoric UABS, the percentage of generic responses jumps drastically from 1.0% to 28.4%. In comparison, epistemic UABS keeps the generic response rates low while achieving lower hallucination rates.

5.3 Data-to-text Results

We also evaluate the effect of UABS on the ToTTo dataset. We choose to penalize epistemic uncertainty due to its better performances than aleatoric uncertainty, as shown in the previous section. A five-model deep ensemble is used to quantify the epistemic uncertainty and generate results with UABS. We compare the BLEU score and three human evaluation metrics among results generated with different uncertainty penalty weights. 100 generation results are randomly selected and evaluated for each penalty weight choice. The results are shown in Table 6. We can see that a relatively small penalty weight leads to a reduced hallucination chance (hence more faithful) with a cost on the BLEU score and fluency.

To qualitatively examine the sentences generated with different λ values, we show example results on the ToTTo validation set in Table 7. We can see that with larger penalty weights, the UABS results drop certain statements that the model deems less confident regardless of the correctness. This results in shorter but more confident predictions for UABS results with a larger uncertainty penalty.

6 Related Work

Hallucination There are many pieces of anecdotal evidence of hallucination presented in various NLG tasks. Most recently, researchers started investigating the phenomenon systematically. Rohrbach et al. (2018) analyzes object hallucination focusing on the objects that appeared in the MSCOCO segmentation challenge. They propose the CHAIR metric to quantify the severity of object hallucination. They find that the models tend to make predictions consistent with a language model trained on the captions instead of a model trained to predict objects in an image. Therefore hallucination is caused by an over-reliance on the language priors. Nie et al. (2019) believes that the origin of the hallucination problem in neural surface realization comes from the data side. More specifically, datasets used for NLG systems often include instances with information misalignment between the input structure and the output text. They propose integrating a language understanding module for iterative data refinement to better align meaning representations and output text. Müller et al. (2019) examines hallucination in neural machine translation and observes that the phenomenon is most common in out-of-domain settings. They empirically compare several strategies to improve domain robustness in NMT and find that a combination of reconstruction and a noisy channel model for reranking is most effective.

These observations are consistent with our findings. For example, domain shift and data misalignment are known to lead to a higher level of epistemic uncertainty (Kendall and Gal, 2017) which makes hallucination a more severe problem.

Uncertainty quantification Uncertainty quantification has attracted more attention recently due to the progress in Bayesian deep learning. Bayes by backprop (Blundell et al., 2015), Monte Carlo dropout (Gal and Ghahramani, 2016), and deep ensembles (Lakshminarayanan et al., 2017) are examples of popular Bayesian approaches to evaluate uncertainty with deep neural models. Kendall and Gal (2017) investigates the benefits of modeling epistemic and aleatoric uncertainty in vision tasks such as semantic segmentation and depth regression. They show that it is important to model aleatoric uncertainty with large datasets and real-time applications and epistemic uncertainty with small datasets and safety-critical applications. Other applications of uncertainty quantification have been explored in the context of time series predictions (Zhu and Laptev, 2017), natural language processing tasks (Xiao and Wang, 2019), etc. More broadly, prediction entropy has been analyzed in different neural language generation tasks (Ott et al., 2018; Xu et al., 2020). Depeweg et al. (2018) shows how to extract and decompose uncertainty in Bayesian neural networks with latent variables for decision-making purposes. They show that active learning and risk-sensitive reinforcement learning both benefit from uncertainty decomposition.

7 Discussion and Conclusions

We investigate the relationship between hallucination and predictive uncertainty in image captioning and data-to-text generation tasks and show that predictions with higher uncertainty are more prone to hallucination. In particular, epistemic uncertainty is more indicative of hallucination than aleatoric uncertainty. We propose uncertainty-aware beam search to incorporate uncertainty into the decoding process to reduce hallucination. We show that uncertainty decomposition helps the proposed beam search variant to achieve a better performance-hallucination trade-off. Specifically, penalizing epistemic uncertainty yields better results compared to penalizing aleatoric or total uncertainty.

In this work, we analyze uncertainty from the token level. This might be restrictive because uncertainty corresponds to the current prediction context instead of the predicted token. The relationship between hallucination and uncertainty, therefore, can be much more complicated than a linear one. It is still possible to produce hallucinated information with a very confident model. The proposed UABS reduces hallucination by limiting the total uncertainty of the generated text. As a result, it might lead to shorter generations and lower generation quality. Devising more sophisticated uncertainty-aware training and decoding methods with less adverse effects on the generation quality is a future direction to explore.

Acknowledgement

We would like to gratefully acknowledge the support of the National Science Foundation. The views expressed are those of the author and do not reflect the official policy or position of the US government.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. 2018. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pages 1184–1193. PMLR.

Armen Der Kiureghian and Ove Ditlevsen. 2009. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

Geoffrey E Hinton and Drew Van Camp. 1993. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413.

Ruotian Luo, Brian Price, Scott Cohen, and Gregory Shakhnarovich. 2018. Discriminability objective for training descriptive captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6964–6974.

Kathleen McKeown. 1992. *Text generation*. Cambridge University Press.

Mathias Müller, Annette Rios, and Rico Sennrich. 2019. Domain robustness in neural machine translation. *arXiv preprint arXiv:1911.03109*.

Radford M Neal. 1995. *BAYESIAN LEARNING FOR NEURAL NETWORKS*. Ph.D. thesis, University of Toronto.

Feng Nie, Jin-Ge Yao, Jinpeng Wang, Rong Pan, and Chin-Yew Lin. 2019. A simple recipe towards reducing hallucination in neural surface realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *International Conference on Machine Learning*, pages 3956–3965.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

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

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. 2018. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4035–4045.

Ran Tian, Shashi Narayan, Thibault Sellam, and Ankur P Parikh. 2019. Sticking to the facts: Confident decoding for faithful data-to-text generation. *arXiv preprint arXiv:1910.08684*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Yijun Xiao and William Yang Wang. 2019. Quantifying uncertainties in natural language processing tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7322–7329.

Jiacheng Xu, Shrey Desai, and Greg Durrett. 2020. Understanding neural abstractive summarization models via uncertainty. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6275–6281.

Lingxue Zhu and Nikolay Laptev. 2017. Deep and confident prediction for time series at uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 103–110. IEEE.

Fine-Grained Event Trigger Detection

Duong Minh Le
VinAI Research, Vietnam
v.duonglm1@vinai.io

Thien Huu Nguyen[*]
Department of Computer and Information Science
University of Oregon, Eugene, OR 97403, USA
thien@cs.uoregon.edu

Abstract

Most of the previous work on Event Detection (ED) has only considered the datasets with a small number of event types (i.e., up to 38 types). In this work, we present the first study on fine-grained ED (FED) where the evaluation dataset involves much more fine-grained event types (i.e., 449 types). We propose a novel method to transform the Semcor dataset for Word Sense Disambiguation into a large and high-quality dataset for FED. Extensive evaluation of the current ED methods is conducted to demonstrate the challenges of the generated datasets for FED, calling for more research effort in this area.

1 Introduction

Understanding events in text is an important aspect of Natural Language Processing (NLP). Toward this end, Event Detection (ED), a task of Information Extraction (IE), aims to identify event triggers in sentences and classify them into some predefined types of interest. Event triggers represent the most important words (usually single verbs or nominalizations) in the sentences that evoke the events. The current state-of-the-art methods for ED feature the deep learning models where many new network architectures are introduced in the last couple of years (Nguyen and Grishman, 2015; Chen et al., 2015; Liu et al., 2017, 2019a; Lai et al., 2020b).

Among others, the rapid development of the deep learning models for ED can be partly attributed to the availability of the large datasets to evaluate the models (e.g., the ACE 2005 and TAC KBP 2015 datasets (Walker et al., 2006; Mitamura et al., 2015)). Unfortunately, a major issue in these existing datasets for ED is that they tend to only focus on a limited set of event types. For example, the popular ACE 2005 dataset is only annotated for 33 event subtypes (e.g., *Attack, Start-Position, Elect*) while the number of events in the TAC KBP dataset (Mitamura et al., 2015) is 38. On the one hand, the limited numbers of types are unable to cover a wide range of possible events in practice (Araki and Mitamura, 2018). On the other hand, the small label sets often amount to the coarse-grained event types in the existing datasets that cannot capture the slightly different nuances (i.e., fine-grained distinction) of the events. For instance, both the words "*quit*" and "*fired*" in the two sentences "*He decided to* **quit** *the job.*" and "*He was* **fired** *due to a policy violation.*" (respectively) would be considered as the trigger words of the same event type of *End-Position* in the ACE 2005 dataset. However, the nuances in these two events are quite different (i.e., in term of the willingness of the job termination) and the ability to characterize such subtle distinction would be useful for the downstream applications (Choi et al., 2018).

In order to address these problems, we propose to explore the problem of Fine-grained Event Detection (FED) that seeks to solve ED with much larger and finer-grained sets of event types (motivated by the fine-grained entity typing task (Ling and Weld, 2012; Choi et al., 2018)). To our knowledge, this is the first work to explicitly study FED in the literature. A major challenge in this research direction is the creation of the evaluation datasets to enable effective model development and analysis. In particular, it is non-trivial to design a large set of fine-grained event types to be applied to annotate the datasets. In addition, with such a large number of fine-grained event types (i.e., 449 in this work), the traditional labeling procedure with human involvement might be too expensive and error-prone when it comes to the generation of large datasets for FED. To this end, we introduce a novel method to address these challenges and produce a large dataset for FED based on WordNet and Word Sense

---

[*] Corresponding author.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2745–2752
April 19 - 23, 2021.

Disambiguation (WSD) datasets. Our method involves two major steps where we first leverage the synset typology in WordNet to formulate the fine-grained event types and then convert the annotated datasets for WSD to establish the datasets for our FED problem. This novel data generation procedure minimizes the human effort and allows us to create a large and high-quality dataset with 449 fine-grained event types for FED. Finally, we extensively evaluate the state-of-the-art ED models on the proposed FED dataset. The experiments show that the performance of the current ED models is not yet satisfactory for FED and further research is needed to advance the performance in this area. We will publicly release the proposed dataset to promote the future research on FED.

2 Data Generation Procedure

The goal of this section is to generate a large dataset for ED with many fine-grained event types to evaluate the FED models. Our proposed procedure to achieve this goal involves two major steps. First, we identify the eventive synsets/senses in WordNet 3.0 (Miller, 1995) and group them into classes with similar eventive meanings. These classes would serve as the fine-grained event types in the resulting FED dataset. As the result, we obtain a mapping from the set of WordNet synsets to the set of the fine-grained event types for our problem (some WordNet synsets might not be mapped to any event type in our case). Afterward, we leverage the Semcor dataset for WSD (Miller et al., 1994) and map the synsets annotated for the words in this dataset into the event types in our setting. This conversion process produces a dataset whose words are assigned with the fine-grained event types in our FED problem. As the Semcor dataset is manually annotated, the resulting FED dataset would be large and have high quality if the synset-event type mapping is constructed well.

In particular, for eventive synset/sense identification, we first start with nouns. Following (Araki and Mitamura, 2018), we assume that any synset for a noun subsumed by one of the three following synsets via the WordNet hyponyms would be considered as eventive: $state_n^2$ (i.e., the way something is with respect to its main attributes), $process_n^6$ (i.e., a sustained phenomenon or one marked by gradual changes through a series of states), and $event_n^1$ (i.e., something that happens at a given place and time). In this way, we find 13,166 eventive synsets over 82,115 synsets for nouns. We call these three general synsets as the eventive root synsets in the following. Starting from these root synsets, we traverse the synset graph in WordNet by following the hyponym links. The graph traversal procedure will generate three different trees whose nodes are the eventive synsets and roots correspond to the three selected synsets. For convenience, we call the synsets in WordNet that can be reached by one of the three synsets above after n hyponym links as the synsets at the n-th level[1] (so the root synsets are at the zero level). In order to form the fine-grained event types for FED, we select the WordNet synsets at the 4^{th} level as the core meanings (called the core synsets) for the event types in our dataset (there are 2,637 core eventive synsets found in this way). We empirically choose the synsets at the 4^{th} level to balance two factors. On the one hand, the synsets at the shallower levels lead to too general event types that cannot achieve the expected fine-grained property. On the other hand, going deeper for the core event meanings reduces the numbers of examples per event type in the final FED dataset converted from Semcor.

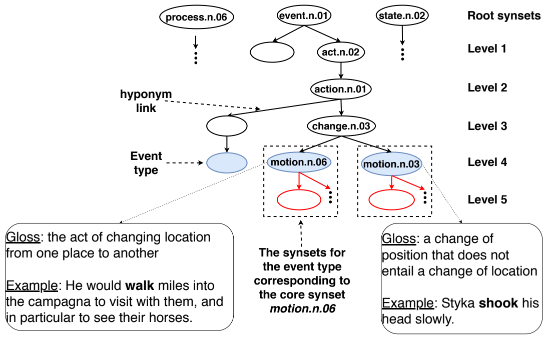


Figure 1: The synsets at different levels and the examples for some event types and their core synsets.

Given a core synset A, we identify the other synsets with similar meaning to A and combine them to represent a fine-grained event type (called E) in our dataset (i.e., the event type E will involve several semantically similar synsets in WordNet). In this work, we include two following classes of synsets in the event type E for A:

- The synsets for nouns that can be reached from A with the hyponym links: Intuitively, the synsets subsumed by A would exhibit the general eventive

---

[1] It is possible that some eventive synsets in WordNet might reside at more than one level as they can be reached from the three root nodes with multiple paths. We resolve this conflict by putting these synsets on the closet level to the roots.

meanings of A with some certain distinctions.

- The synsets of the derivationally related forms of the lemmas/senses l in A and their synset descendants (via the hyponym links): In WordNet, the derivationally related forms of a lemma l in the synset A involve the lemmas from different syntactic categories (e.g., verbs and adjectives) that have the same root form as l and are semantically related to l and A (e.g., destruction → destroy) (Miller, 1995). Due to such semantic similarity, we expect that the synsets of the derivationally related forms of the lemmas in A and their descendants also express the same eventive meaning as A, thereby enriching the synsets for E with the syntactic categories beyond nouns.

Up to this point, we obtain a set of 2,637 event types, each represented by a core synset and a set of related synsets. We combine all the other synsets (i.e., the ones that do not appear in any of the 2637 types) to create a single event type called *Other* as in the traditional ED task. With these grouping information, we can now create a mapping from the synsets/senses in WordNet to the 2638 established event types for our dataset (called the *M-WordNet-Event* mapping). Based on this mapping, we transform each example in the WSD Semcor dataset, which involves a sentence and a word of interest, into an example in the new dataset for FED (called FedSemcor) where the synset/sense label for the word in the original example of Semcor is mapped into the corresponding event type in FedSemcor. As the final processing step, we remove from FedSemcor any event types that have less than 10 examples to ensure that the event types are adequately represented in our dataset. This step significantly reduces the number of event types in FedSemcor, leaving us 449 event types (not including *Other*) with 34,666 examples. These are called the positive examples where we have 77.2 examples per event type in average. As a result, the synsets for the removed event types are also included in the synset set for the *Other* type in the mapping. The number of examples with the *Other* type in the final FedSemcor dataset (called the negative examples) is 98,309. Figure 1 illustrates the synset levels and some examples for the event types.

Implementation Details: In the actual implementation of the data generation procedure for FedSemcor, given a core synset A, we do not include all the descendants of A and the synsets of the derivationally related forms of A's lemmas into the synset set for the correspdoning event type E for A. Instead, we only include the descendants that are at at most 2 hyponym links away from A and the synsets of the derivationally related forms of A's lemmas in E. This is based on our empirical investigation of the data where the descendants with more than 2 links away tend to have semantic drifts from A, potentially introducing noise into the event type E. For example with the core synset $motion_n^6$ (i.e., *the act of changing location from one place to another*) at the 4^{th} level, the descendants at the 5^{th}, 6^{th} and 7^{th} levels include: level 5: $approach_n^2$ (i.e., *the act of drawing spatially closer to something*), level 6: $access_n^6$ (i.e., *the act of approaching or entering*), and level 7: $back\_door_n^6$ (i.e., *a secret or underhand means of access (to a place or a position)*). As we can see, while the synsets at the 5^{th} and 6^{th} levels are related to the original core synset, the synset at the 7^{th} level already involves some semantic departure from the one at the 4^{th} level that should be avoided to improve the precision.

	ACE	**TAC KBP**	**FedSemcor**
# event types	33	38	449
# positive examples	4,907	11,975	34,666
# negative examples	104,217	126,934	98,309

Table 1: Statistics for the FedSemcor, ACE 2005 and TAC KBP 2015 datasets. Negative examples refer to the non-trigger words while positive examples are annotated trigger words for the event types of interest.

Dataset Statistics: Table 1 reports some statistics for FedSemcor and some prior popular datasets for ED (i.e., ACE 2005 (Walker et al., 2006) and TAC KBP 2015 (Mitamura et al., 2015)) to facilitate the comparison. As we can see from the table, FedSemcor has more positive examples, but less negative examples than ACE 2005 and TAC KBP 2015, making FedSemcor a more balanced dataset than the other two. In addition, we show the distribution of 50 event types with the highest numbers of examples in FedSemcor in Figure 2. Finally, Figure 3 illustrates the distribution of the sentence lengths for the examples in FedSemcor.

Evaluation of FedSemCor: As we rely on the manual annotation in Semcor for the synsets for the words, the main bottleneck in the data generation procedure is the mapping from the WordNet synsets to the 450 event types in FedSemCor (including *Other*). In order to evaluate the quality of this mapping, we sample 500 synsets from WordNet that are different from the core synsets of the 449 positive event types. Two experienced NLP re-

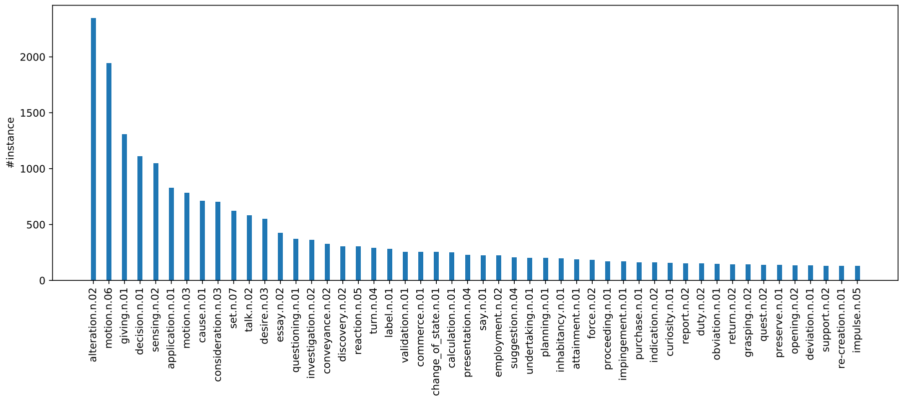


Figure 2: The distribution of the 50 event types with the highest numbers of examples/instances in FedSemcor.

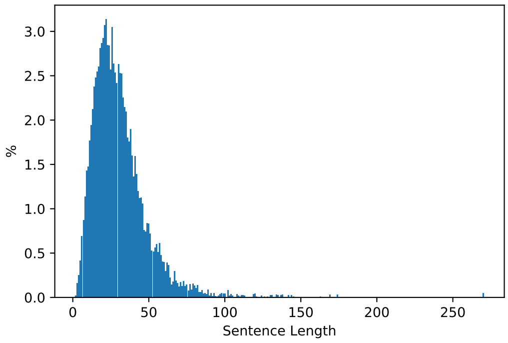


Figure 3: The distribution of the sentence lengths for the examples in FedSemcor.

searchers then independently examine each of these 500 sampled synsets to determine the appropriate event type for it (among the 450 types). In doing so, they examined the glossaries of the synsets as well as the examples provided by WordNet. The two annotators achieved 79.8% agreement for which the synsets with conflicts are resolved by a third NLP researcher. Afterward, we apply the synset-event type mapping obtained in the data generation procedure to annotate the 500 sampled synsets for the event types. The event types provided by the mapping are then compared with those from the annotators, leading to 83.6%, 78.6% and 81.0% as the precision, recall, and F1 scores respectively.

3 Evaluation

Models and Data: In order to understand the complexity of the FedSemcor dataset for FED, this section evaluates the performance of the state-of-the-art models for the traditional ED problem on this dataset. In particular, we first split FedSemcor into the training, development and test data using the 6:2:2 ratio over the entire dataset. Table 2 presents the statistics about these data portions. Note that similar to some prior ED work (Nguyen and Grishman, 2015; Chen et al., 2015), our FED problem is formulated as a word classification problem where given a word in an input sentence, the models need to predict the event type for the word.

Dataset	Training		Development		Test	
	#Pos	#Neg	#Pos	#Neg	#Pos	#Neg
FedSemcor	20,799	58,985	5,546	21,038	8,321	18,286

Table 2: The size of the dataset portions.

Afterward, we consider the following representative models for ED: **CNN** (Nguyen and Grishman, 2015), **DMCNN** (Chen et al., 2015), **SupAtt** (Liu et al., 2017) (i.e., supervised attention), **GCN** (Nguyen and Grishman, 2018), and **MOGANED** (Yan et al., 2019) (i.e., a Multi-Order Graph Convolution model). MOGANED is the state-of-the-art model with uncontextualized word embeddings in traditional ED (i.e., on ACE 2005). For these models, we use both the traditional word embeddings `word2vec` and the recent contextualized word embeddings BERT (i.e., the uncased base model) (Devlin et al., 2019) as the pre-trained word embeddings. For BERT, we further evaluate the ED models in (Wang et al., 2019) (called **DMBERT**) and (Yang et al., 2019) (called **BERT-ED**) that have the best-reported performance on ACE 2005 for ED.

For the experiments in this work, we re-tune the hyper-parameters of the models on the development set of FedSemcor. In particular, depending on which components each model has, we use the following bounds to search for the hyperparameters: $[100, 200, 300, 400, 500]$ for the dimensionality of the hidden vectors in the layers of all the feed-forward, BiLSTM, and GCN networks, $[1, 2, 3]$ for the numbers of layers for BiLSTM and GCN, $[16, 32, 64]$ for the mini-batch size, $[1e\text{-}5, 1e\text{-}4, 1e\text{-}3, 1e\text{-}2, 1e\text{-}1]$ for the learning rate of the Adam optimizer, and $[10, 20, 30, 40, 50]$ for the dimensions of the feature embeddings, i.e., position embeddings in CNN (Nguyen and Grishman, 2015; Chen et al., 2015).

Finally, in order to demonstrate the benefit of the conversion of Semcor (i.e., a WSD dataset) into FedSemcor for FED, we consider a WSD-based baseline for FED where the state-of-the-art WSD model in (Hadiwinoto et al., 2019) is trained on the training data of FedSemcor. As this is a WSD model, instead of using the mapped event types as the labels for the examples (i.e., the 450 types) in the training data, we employ the original word senses of the words as the labels to train this WSD model. Afterward, we apply the trained WSD model on the test data of FedSemcor, producing a word sense for each example. In the last step, the mapping *M-WordNet-Event* is utilized to convert the predicted word senses for the test set examples into the event types for FedSemcor that would be evaluated to obtain the FED performance for this baseline (called **WSD-based**). Note that this WSD model also uses the BERT embeddings.

Results: Table 3 shows the performance of the models on the test set of FedSemcor. From the table, we see that GCN has the best performance among the models with `word2vec` while BERT-ED outperforms all the BERT-based models. However, the best performance on FedSemcor (i.e., 65.0% F1 score with BERT-ED) is still far behind the typical performance (i.e., up to 80.7% in (Yang et al., 2019)) of the models on the traditional ED datasets (i.e., ACE 2005). This suggests the more challenging nature of FedSemcor and FED over traditional ED, presenting a challenge for the future research in this area. Importantly, the performance of the ED models (i.e., with the BERT embeddings) is significantly better than the WSD-based baseline (i.e., up to 9% performance gap with BERT-ED), clearly testifying to the advantages of the conversion from Semcor into FedSemcor for FED.

4 Related Work

ED has been studied extensively in the last decade, featuring feature-based models (Ahn, 2006; Ji and Grishman, 2008; Li et al., 2013, 2015), deep learning models (Chen et al., 2015; Nguyen et al., 2016b,a; Nguyen and Grishman, 2016; Chen et al., 2017; Liu et al., 2018; Yan et al., 2019; Ngo et al., 2020; Lai et al., 2020b), and few/zero-shot learning models (Huang et al., 2018; Lai and Nguyen, 2019; Lai et al., 2020a). The rapid development of such models has been facilitated by the availability of the ED datasets in different domains, including the general domain with the popular ACE and TAC KBP datasets (Walker et al., 2006; Mitamura et al., 2015, 2016), the biomedical domain (Kim et al., 2009, 2011), literature (Sims et al., 2019), cybersecurity (Satyapanich et al., 2020; Man Duc Trong et al., 2020), and the open domain (Araki and Mitamura, 2018; Liu et al., 2019b). However, these datasets only involve a small number of event types and none of them has considered ED with many fine-grained event types as we do.

Model	`word2vec`			BERT		
	P	R	F1	P	R	F1
WSD-based	-	-	-	47.2	68.7	56.0
CNN	48.8	53.6	51.1	52.8	69.8	60.1
DMCNN	43.7	51.6	47.4	56.9	71.5	63.4
SupAtt	58.5	46.0	51.5	59.5	71.1	64.8
GCN	53.7	60.0	**56.7**	58.9	71.5	64.6
MOGANED	48.6	61.8	54.4	55.8	71.2	62.6
DMBERT	-	-	-	57.7	63.2	60.3
BERT-ED	-	-	-	59.2	72.1	**65.0**

Table 3: The performance on the FedSemcor test set.

Our FED task is also related to fine-grained entity typing that aims to classify entity mentions into a fine-grained set of types (Karn et al., 2017; Shimaoka et al., 2016; Lin and Ji, 2019). The techniques to generate datasets for fine-grained entity typing include distant supervision (Ling and Weld, 2012; Abhishek et al., 2017) and manual annotation (Murty et al., 2018; Choi et al., 2018). Notably, (Del Corro et al., 2015) also uses WordNet to establish the fine-grained entity types, applying different entity mention extractors over external corpus. Our work is different as we focus on fine-grained event types using the manually annotated corpus Semcor to generate data.

5 Conclusion

We study a new task of FED, featuring 449 fine-grained event types in the dataset for ED. A novel method to generate the evaluation dataset for FED is introduced, leveraging manually annotated WSD datasets (i.e., Semcor) and the eventive synsets in WordNet. We evaluate the state-of-the-art ED models on the new dataset to show the opportunities for the future research on FED.

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807, Valencia, Spain. Association for Computational Linguistics.

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

Jun Araki and Teruko Mitamura. 2018. Open-domain event detection using distant supervision. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 878–891, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419, Vancouver, Canada. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

Luciano Del Corro, Abdalghani Abujabal, Rainer Gemulla, and Gerhard Weikum. 2015. FINET: Context-aware fine-grained named entity typing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 868–878, Lisbon, Portugal. Association for Computational Linguistics.

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

Christian Hadiwinoto, Hwee Tou Ng, and Wee Chung Gan. 2019. Improved word sense disambiguation using pre-trained contextualized word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5297–5306, Hong Kong, China. Association for Computational Linguistics.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Sanjeev Karn, Ulli Waltinger, and Hinrich Schütze. 2017. End-to-end trainable attentive decoder for hierarchical entity classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 752–758, Valencia, Spain. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado. Association for Computational Linguistics.

Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. 2011. Overview of Genia event task in BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 7–15, Portland, Oregon, USA. Association for Computational Linguistics.

Viet Dac Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2020a. Exploiting the matching information in the support set for few shot event classification. In *Proceedings of the 24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Singapore.

Viet Dac Lai and Thien Nguyen. 2019. Extending event detection to new types with learning from keywords. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 243–248, Hong Kong, China. Association for Computational Linguistics.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020b. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing (EMNLP), pages 5405–5411, Online. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Xiang Li, Thien Huu Nguyen, Kai Cao, and Ralph Grishman. 2015. Improving event detection with abstract meaning representation. In *Proceedings of the First Workshop on Computing News Storylines*, pages 11–15, Beijing, China. Association for Computational Linguistics.

Ying Lin and Heng Ji. 2019. An attentive fine-grained entity typing model with latent type representation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6197–6202, Hong Kong, China. Association for Computational Linguistics.

Xial Ling and Daniel Weld. 2012. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI)*, page 94–100, Toronto, Canada. Association for the Advancement of Artificial Intelligence.

Jian Liu, Yubo Chen, and Kang Liu. 2019a. Exploiting the ground-truth: An adversarial imitation based knowledge distillation approach for event detection. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, Honolulu, Hawaii. Association for the Advancement of Artificial Intelligence.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, New Orleans, Louisiana. Association for the Advancement of Artificial Intelligence.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, and Yue Zhang. 2019b. Open domain event extraction using neural latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2860–2871, Florence, Italy. Association for Computational Linguistics.

Hieu Man Duc Trong, Duc Trong Le, Amir Pouran Ben Veyseh, Thuat Nguyen, and Thien Huu Nguyen. 2020. Introducing a new dataset for event detection in cybersecurity texts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5381–5390, Online. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: a lexical database for english. In *Communications of the ACM, 38(11):39–41*.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Teruko Mitamura, Zhengzhong Liu, and Eduard Hovy. 2015. Overview of tac kbp 2015 event nugget track. In *Proceedings of the Text Analysis Conference (TAC)*, Gaithersburg, Maryland. National Institute of Standards and Technology (NIST).

Teruko Mitamura, Zhengzhong Liu, and Eduard Hovy. 2016. Overview of tac kbp 2016 event nugget track. In *Proceedings of the Text Analysis Conference (TAC)*, Gaithersburg, Maryland. National Institute of Standards and Technology (NIST).

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Nghia Trung Ngo, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Learning to select important context words for event detection. In *Proceedings of the 24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Singapore.

Thien Huu Nguyen, , Adam Meyers, and Ralph Grishman. 2016a. New york university 2016 system for kbp event nugget: A deep learning approach. In *Proceedings of the Text Analysis Conference (TAC)*, Gaithersburg, Maryland. National Institute of Standards and Technology (NIST).

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016b. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short*

Papers), pages 365–371, Beijing, China. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolutional neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 886–891, Austin, Texas. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, New Orleans, Louisiana. Association for the Advancement of Artificial Intelligence.

Taneeya Satyapanich, Francis Ferraro, and Tim Finin. 2020. Casie: Extracting cybersecurity event information from text. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, New York, New York. Association for the Advancement of Artificial Intelligence.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 69–74, San Diego, CA. Association for Computational Linguistics.

Matthew Sims, Jong Ho Park, and David Bamman. 2019. Literary event detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3623–3634, Florence, Italy. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 998–1008, Minneapolis, Minnesota. Association for Computational Linguistics.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5766–5770, Hong Kong, China. Association for Computational Linguistics.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Extremely Small BERT Models from Mixed-Vocabulary Training

Sanqiang Zhao*
University of Pittsburgh, PA
`sanqiang.zhao@pitt.edu`

Raghav Gupta*
Google Research, Mountain View, CA
`raghavgupta@google.com`

Yang Song
Kuaishou Technology, Beijing, China
`yangsong@kuaishou.com`

Denny Zhou
Google Brain, Mountain View, CA
`dennyzhou@google.com`

Abstract

Pretrained language models like BERT have achieved good results on NLP tasks, but are impractical on resource-limited devices due to memory footprint. A large fraction of this footprint comes from the input embeddings with large input vocabulary and embedding dimensions. Existing knowledge distillation methods used for model compression cannot be directly applied to train student models with reduced vocabulary sizes. To this end, we propose a distillation method to align the teacher and student embeddings via mixed-vocabulary training. Our method compresses BERT$_{\text{LARGE}}$ to a task-agnostic model with smaller vocabulary and hidden dimensions, which is an order of magnitude smaller than other distilled BERT models and offers a better size-accuracy trade-off on language understanding benchmarks as well as a practical dialogue task.

1 Introduction

Recently, pre-trained context-aware language models like ELMo (Peters et al., 2018), GPT (Radford et al., 2019), BERT (Devlin et al., 2018) and XLNet (Yang et al., 2019) have outperformed traditional word embedding models like Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), and achieved strong results on a number of language understanding tasks. However, these models are typically too huge to host on mobile/edge devices, especially for real-time inference. Recent work has explored, inter alia, knowledge distillation (Ba and Caruana, 2014; Hinton et al., 2015) to train small-footprint student models by implicit transfer of knowledge from a teacher model.

Most distillation methods, however, need the student and teacher output spaces to be aligned. This complicates task-agnostic distillation of BERT to smaller-vocabulary student BERT models since the input vocabulary is also the output space for the masked language modeling (MLM) task used in BERT. This in turn limits these distillation methods' ability to compress the input embedding matrix, that makes up a major proportion of model parameters e.g. the ∼30K input WordPiece embeddings of the BERT$_{\text{BASE}}$ model make up over 21% of the model size. This proportion is even higher for most distilled BERT models, owing to these distilled models typically having fewer layers than their teacher BERT counterparts.

We present a task and model-agnostic distillation approach for training small, reduced-vocabulary BERT models running into a few megabytes. In our setup, the teacher and student models have incompatible vocabularies and tokenizations for the same sequence. We therefore align the student and teacher WordPiece embeddings by training the teacher on the MLM task with a mix of teacher-tokenized and student-tokenized words in a sequence, and then using these student embeddings to train smaller student models. Using our method, we train compact 6 and 12-layer reduced-vocabulary student models which achieve competitive performance in addition to high compression for benchmark datasets as well as a real-world application in language understanding for dialogue.

2 Related Work

Work in NLP model compression falls broadly into four classes: matrix approximation, weight quantization, pruning/sharing, and knowledge distillation.

The former two seek to map model parameters to low-rank approximations (Tulloch and Jia, 2017) and lower-precision integers/floats (Chen et al., 2015; Zhou et al., 2018; Shen et al., 2019) respectively. In contrast, pruning aims to remove/share redundant model weights (Li et al., 2016; Lan et al., 2019). More recently, dropout (Srivastava et al., 2014) has been used to cut inference latency by

---

Asterisk (*) denotes equal contribution. Research conducted when all authors were at Google.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2753–2759
April 19 - 23, 2021.

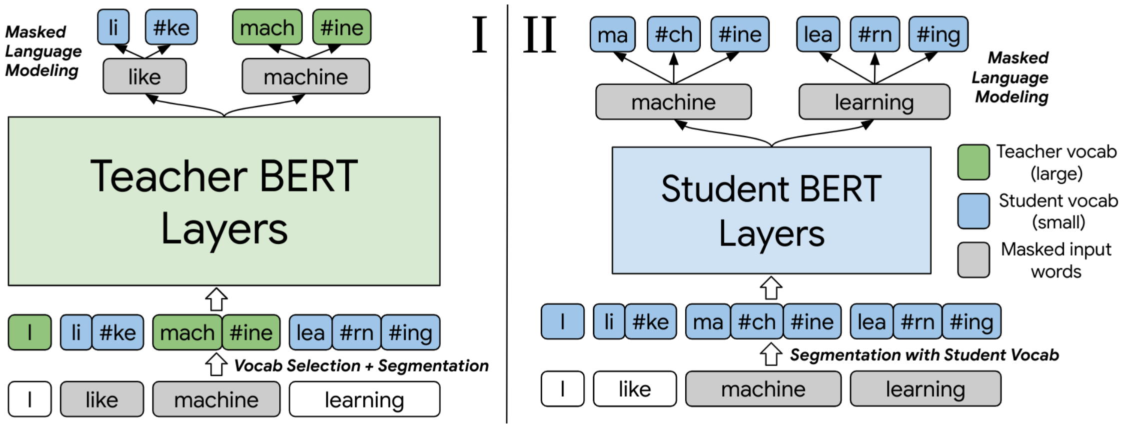


Figure 1: Depiction of our mixed-vocabulary training approach. (Left) Stage I involving retrained teacher BERT with default config (e.g., 30K vocabulary, 768 hidden dim) and mixed-vocabulary input. (Right) Stage II involving student model with smaller vocabulary (5K) and hidden dims (e.g., 256) and embeddings initialized from stage I.

early exit (Fan et al., 2019; Xin et al., 2020).

Knowledge distillation focuses on implicit transfer of knowledge as soft teacher predictions (Tang et al., 2019), attention distributions (Zagoruyko and Komodakis, 2016) and intermediate outputs (Romero et al., 2014). Approaches close to our work rely on similar methods (Sanh et al., 2019; Sun et al., 2019), while others involve combinations of layer-wise transfer (Sun et al., 2020), task-specific distillation (Jiao et al., 2019), architecture search (Chen et al., 2020) and layer dropout (Xu et al., 2020); many of these are specific to the transformer layer (Vaswani et al., 2017).

Another highly relevant line of work focuses on reducing the size of the embedding matrix, either via factorization (Shu and Nakayama, 2018; Lan et al., 2019) or vocabulary selection/pruning (Provilkov et al., 2019; Chen et al., 2019b).

3 Proposed Approach

Here, we discuss our rationale behind reducing the student vocabulary size and its challenges, followed by our *mixed-vocabulary* distillation approach.

3.1 Student Vocabulary

WordPiece (WP) tokens (Wu et al., 2016) are subword units obtained by applying greedy segmentation to a training corpus. Given such a corpus and a number of desired tokens D, a WordPiece vocabulary is generated by selecting D subword tokens such that the resulting corpus is minimal in the number of WordPiece when segmented according to the chosen WordPiece model. The greedy algorithm for this optimization problem is described in more detail in Sennrich et al. (2016). Most published BERT models use a vocabulary of 30522 WordPieces, obtained by running the above algorithm on the Wikipedia and BooksCorpus (Zhu et al., 2015) corpora with a desired vocabulary size D of 30000.

For our student model, we chose a target vocabulary size D of 5000 WordPiece tokens. Using the same WordPiece vocabulary generation algorithm and corpus as above, we obtain a 4928-WordPiece vocabulary for the student model. This student vocabulary includes all ASCII characters as separate tokens, ensuring no out-of-vocabulary words upon tokenization with this vocabulary. Additionally, the 30K teacher BERT vocabulary includes 93.9% of the WP tokens in this 5K student vocabulary but does not subsume it. We explore other strategies to obtain a small student vocabulary in Section 6.

For task-agnostic student models, we reuse BERT's masked language modeling (MLM) task: words in context are randomly masked and predicted given the context via softmax over the model's WP vocabulary. Thus, the output spaces for our teacher (30K) and student (5K) models are unaligned. This, coupled with both vocabularies tokenizing the same words differently, means existing distillation methods do not apply to our setting.

3.2 Mixed-vocabulary training

We propose a two-stage approach for implicit transfer of knowledge to the student via the student embeddings, as described below.

Stage I (Student Embedding Initialization): We first train the student embeddings with the teacher model initialized from $\text{BERT}_{\text{LARGE}}$. For a given input sequence, we mix the vocabularies by randomly selecting (with probability p_{SV}, a hyperparameter) words from the sequence to segment using the student vocabulary, with the other words segmented using the teacher vocabulary. As in Figure 1 on the left, for input [‘*I*’, ‘*like*’, ‘*machine*’, ‘*learning*’], the words *‘like’* and *‘learning’* are segmented using the student vocabulary (in blue), with the others using the teacher vocabulary (in green). Similar to Lample and Conneau (2019), this step seeks to align the student and teacher embeddings for the same tokens: the model learns to predict student tokens using context which is segmented using the teacher vocabulary, and vice versa.

Note that since the student embeddings are set to a lower dimension than the teacher embeddings, as they are meant to be used in the smaller student model, we project the student embeddings up to the teacher embedding dimension using a trainable affine layer before these are input to the teacher BERT. We choose to keep the two embedding matrices separate despite the high token overlap: this is partly to keep our approach robust to lower vocabulary overlap settings, and partly due to empirical considerations described in Section 6.

Let θ_s/eb_s and θ_t/eb_t denote the transformer layer and embedding weights for the student and teacher models respectively. The loss defined in Equation 1 is the MLM cross entropy summed over masked positions M_t in the teacher input. y_i and c_i denote the predicted and true tokens at position i respectively and can belong to either vocabulary. $v_i{\in}\{s,t\}$ denotes the vocabulary used to segment this token. Separate softmax layers P_{v_i} are used for token prediction, one for each vocabulary, depending on the segmenting vocabulary v_i for token i. All teacher parameters (θ_t, eb_t) and student embeddings (eb_s) are updated in this step.

$$L_{s_1} = -\textstyle\sum_{i \in M_t} (\log P_{v_i}(y_i{=}c_i | \theta_t, eb_s, eb_t)) \quad (1)$$

Stage II (Student Model Layers): With student embeddings initialized in stage I, we now train the student model normally i.e., using only the student vocabulary and discarding the teacher model. Equation 2 shows the student MLM loss where M_s is the set of positions masked in the student input. All student model parameters (θ_s, eb_s) are updated.

$$L_{s_2} = -\textstyle\sum_{i \in M_s} \log P_s(y_i{=}c_i | \theta_s, eb_s) \quad (2)$$

4 Experiments

For evaluation, we finetune the student model just as one would finetune the original BERT model i.e., without using the teacher model or any task-specific distillation. We describe our experiments below, with dataset details left to the appendix.

4.1 Evaluation Tasks and Datasets

We fine-tune and evaluate the distilled student models on two classes of language understanding tasks:

GLUE benchmark (Wang et al., 2019)**:** We pick three classification tasks from GLUE:

- MRPC: Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005), a 2-way sentence pair classification task with 3.7K train instances.
- MNLI: Multi-Genre Natural Language Inference (Williams et al., 2018), a 3-way sentence pair classification task with 393K training instances.
- SST-2: Stanford Sentiment Treebank (Socher et al., 2013), a 2-way sentence classification task with 67K training instances.

Spoken Language Understanding: Since we are also keen on edge device applications, we also evaluate on spoken language understanding, a practical task in dialogue systems. We use the SNIPS dataset (Coucke et al., 2018) of ∼14K virtual assistant queries, each comprising one of 7 intents and values for one or more of the 39 pre-defined slots. The intent detection and slot filling subtasks are modeled respectively as 7-way sentence classification and sequence tagging with IOB slot labels.

4.2 Models and Baselines

For GLUE, we train student models with 6 and 12 layers, 4 attention heads, and embedding/hidden dimensions fixed to 256, each using a compact 5K-WP vocabulary. We also evaluate baselines without knowledge distillation (*NoKD*), parameterized identically to the distilled student models (incl. the 5K vocabulary), trained on the MLM teacher objective from scratch. We also compare our models on GLUE with the following approaches:

- *DistilBERT* (Sanh et al., 2019) distill $\text{BERT}_{\text{BASE}}$ to 4/6 layers by aligning teacher predictions,
- *Patient KD - PKD* (Sun et al., 2019) align hidden states to distill $\text{BERT}_{\text{BASE}}$ to 3/6 layers,
- *BERT-of-Theseus* (Xu et al., 2020) use a layer dropout method to distill $\text{BERT}_{\text{BASE}}$ to 6 layers,
- *TinyBERT* (Jiao et al., 2019) apply task specific distillation to $\text{BERT}_{\text{BASE}}$ and align teacher outputs, hidden states as well as embeddings, and
- *MobileBERT* (Sun et al., 2020) combine layer-wise transfer, architecture search and bottleneck

Model	#Params	MRPC (F1/Acc)	MNLI-m/mm (Acc)	SST-2 (Acc)	Average (F1/Acc)
$\text{BERT}_{\text{BASE}}$ (Devlin et al., 2018)	109M	88.9/-	84.6/83.4	93.5	89.0
$\text{BERT}_{\text{LARGE}}$ (Devlin et al., 2018)	340M	89.3/-	86.7/85.9	94.9	90.3
PKD_6 (Sun et al., 2019)	67.0M	85.0/79.9	81.5/81.0	92.0	86.2
PKD_3 (Sun et al., 2019)	45.7M	80.7/72.5	76.7/76.3	87.5	81.6
DistilBERT_4 (Sanh et al., 2019)	52.2M	82.4/-	78.9/78.0	91.4	84.2
MobileBERT (Sun et al., 2020)	25.3M	88.8/84.5	83.3/82.6	92.8	88.3
TinyBERT_4 (Jiao et al., 2019)	14.5M	82.0*/ -	76.6/77.2*	-	-
$\text{TinyBERT}_4^{\dagger}$ (Jiao et al., 2019)	14.5M	86.4/-	82.5/81.8	92.6	87.2
$\text{BERT-of-Theseus}_6^{\dagger}$ (Xu et al., 2020)	66M	87.6/83.2	82.4/82.1	92.2	87.4
NoKD Baseline, L-6, H-256	6.2M	81.2/74.1	76.9/76.1	87.0	81.7
Mixed-vocab distilled (ours), L-6, H-256		84.9/79.3	79.0/78.6	89.1	84.3
NoKD Baseline, L-12, H-256	10.9M	85.1/79.8	79.1/79.0	89.4	84.5
Mixed-vocab distilled (ours), L-12, H-256		87.2/82.6	80.7/80.5	90.6	86.2

* denotes metrics on the development set $\quad$ † denotes results with task-specific distillation

Table 1: Test set accuracy of distilled models, teacher model and baselines on the GLUE test sets. MNLI-m and MNLI-mm refer to the genre-matched and mismatched test sets. All models other than *NoKD* and our distilled models use a 30K-WordPiece vocabulary. The average uses F1 score for MRPC, accuracy for MNLI-m/SST-2.

structures for an optimized student model.

For SNIPS, we shift our focus to smaller, low-latency models for on-device use cases. Here, we train student models with 6 layers and embedding/hidden dimensions $\in \{96, 192, 256\}$. The smaller models here may not be competitive on GLUE but are adequate for practical tasks such as spoken LU. We compare with two strong baselines:

- $\text{BERT}_{\text{BASE}}$ (Chen et al., 2019a) with intent and IOB slot tags predicted using the `[CLS]` and the first WP tokens of each word respectively, and
- StackProp (Qin et al., 2019), which uses a series of smaller recurrent and self-attentive encoders.

4.3 Training Details

Distillation: For all our models, we train the teacher model with mixed-vocabulary inputs (stage I) for 500K steps, followed by 300K steps of training just the student model (stage II). We utilize the same corpora as the teacher model i.e. BooksCorpus (Zhu et al., 2015) and English Wikipedia.

For both stages, up to 20 input tokens were masked for MLM. In stage I, up to 10 of these masked tokens were tokenized using the teacher vocabulary, the rest using the student vocabulary.

We optimize the loss using LAMB (You et al., 2019) with a max learning rate of .00125, linear warmup for the first 10% of steps, batch size of 2048 and sequence length of 128. Distillation was done on Cloud TPUs in a 8x8 pod configuration. p_{SV}, the probability of segmenting a Stage I input word using the student vocabulary, is set to 0.5.

Finetuning: For all downstream task evaluations on GLUE, we finetune for 10 epochs using LAMB with a learning rate of 0.0001 and batch size of 64. For all experiments on SNIPS, we use ADAM with a learning rate of 0.0001 and a batch size of 64.

5 Results

GLUE: Table 1 shows results on downstream GLUE tasks and model sizes for our proposed models, $\text{BERT}_{\text{BASE/LARGE}}$, and baselines. Our models consistently improve upon the identically parameterized *NoKD* baselines, indicating mixed-vocabulary training is better than training from scratch and avoids a large teacher-student performance gap. Compared with PKD/DistilBERT, our 6-layer model outperforms PKD_3 while being >7x smaller and our 12-layer model is comparable to PKD_6 and DistilBERT_4 while being ~5-6x smaller.

Interestingly, our models do particularly well on the MRPC task: the 6-layer distilled model performs almost as well as PKD_6 while being over 10x smaller. This may be due to our smaller models being data-efficient on the smaller MRPC dataset.

TinyBERT and Bert-of-Theseus are trained in task-specific fashion i.e., a teacher model already finetuned on the downstream task is used for distillation. TinyBERT's non-task-specific model results are reported on GLUE dev sets: these results are, therefore, not directly comparable with ours. Even so, our 12-layer model performs credibly

Model	#Params	Latency	Intent Acc	Slot F1
BERT$_{\text{BASE}}$ (Chen et al., 2019a)	109M	340ms	98.6	97.0
StackProp (Qin et al., 2019)	2.6M	>70ms	98.0	94.2
Mixed-vocab distilled, L-6, H-96	1.2M	6ms	98.9	92.8
Mixed-vocab distilled, L-6, H-192	3.6M	14ms	98.8	94.6
Mixed-vocab distilled, L-6, H-256	6.2M	20ms	98.7	95.0

Table 2: Results on the SNIPS dataset. Latency is measured with 4 CPU threads on a Pixel 4 mobile device.

compared with the two, presenting a competitive size-accuracy tradeoff, particularly when compared to the 6x larger BERT-of-Theseus.

MobileBERT performs strongly for the size while being task-agnostic. Our 12-layer model, in comparison, retains ∼98% of its performance with 57% fewer parameters and may thus be better-suited for use on highly resource-limited devices.

TinyBERT sees major gains from task-specific data augmentation and distillation, and MobileBERT from student architecture search and bottleneck layers. Notably, our technique targets the student vocabulary without conflicting with any of the above methods and can, in fact, be combined with these methods for even smaller models.

SNIPS: Table 2 shows results on the SNIPS intent and slot tasks for our models and two state-of-the-art baselines. Our smallest 6-layer model retains over 95% of the BERT$_{\text{BASE}}$ model's slot filling F1 score (Sang and Buchholz, 2000) while being 30x smaller (< 10 MB w/o quantization) and 57x faster on a mobile device, yet task-agnostic. Our other larger distilled models also demonstrate strong performance (0.2-0.5% slot F1 higher than the respective *NoKD* baselines) with small model sizes and latencies low enough for real-time inference. This indicates that small multi-task BERT models (Tsai et al., 2019) present better trade-offs for on-device usage for size, accuracy and latency versus recurrent encoder-based models such as StackProp.

6 Discussion

Impact of vocabulary size: We trained a model from scratch identical to BERT$_{\text{BASE}}$ except with our 5K-WP student vocabulary. On the SST-2 and MNLI-m dev sets, this model obtained 90.9% and 83.7% accuracy respectively - only 1.8% and 0.7% lower respectively compared to BERT$_{\text{BASE}}$.

Since embeddings account for a larger fraction of model parameters with fewer layers, we trained another model identical to our 6×256 model, but with a 30K-WP vocabulary and teacher label distillation. This model showed small gains (0.1% / 0.5% accuracy on SST-2 / MNLI-m dev) over our analogous distilled model, but with 30% more parameters solely due to the larger vocabulary.

This suggests that a small WordPiece vocabulary may be almost as effective for sequence classification/tagging tasks, especially for smaller BERT models and up to moderately long inputs. Curiously, increasing the student vocabulary size to 7K or 10K did not lead to an increase in performance on GLUE. We surmise that this may be due to underfitting owing to the embeddings accounting for a larger proportion of the model parameters.

Alternative vocabulary pruning: Probing other strategies for a small-vocabulary model, we used the above 6×256 30K-WP vanilla distilled model to obtain a smaller model by pruning the vocabulary to contain the intersection of the 30K and 5K vocabularies (total 4629 WPs). This model is 1.2% smaller than our 4928-WP distilled model, but drops 0.8% / 0.7% on SST-2/MNLI-m dev sets.

Furthermore, to exploit the high overlap in vocabularies, we tried running our distillation pipeline but with the embeddings for student tokens (after projecting up to the teacher dimension) also present in the teacher vocabulary tied to the teacher embeddings for those tokens. This model, however, dropped 0.7% / 0.5% on SST-2/MNLI-m compared to our analogous 6×256 distilled model.

We also tried pretraining BERT$_{\text{LARGE}}$ from scratch with the 5K vocabulary and doing vanilla distillation for a 6×256 student: this model dropped 1.2% / 0.7% for SST-2/MNLI-m over our similar distilled model, indicating the efficacy of mixed-vocabulary training over vanilla distillation.

7 Conclusion

We propose a novel approach to knowledge distillation for BERT, focusing on using a significantly smaller vocabulary for the student BERT models. Our *mixed-vocabulary training* method encourages implicit alignment of the teacher and student WordPiece embeddings. Our highly-compressed 6 and 12-layer distilled student models are optimized for on-device use cases and demonstrate competitive performance on both benchmark datasets and practical tasks. Our technique is unique in targeting the student vocabulary size, enabling easy combination with most BERT distillation methods.

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2020. Adabert: Task-adaptive bert compression with differentiable neural architecture search. *arXiv preprint arXiv:2001.04246*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019a. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Wenhu Chen, Yu Su, Yilin Shen, Zhiyu Chen, Xifeng Yan, and William Yang Wang. 2019b. How large a vocabulary does text classification need? a variational approach to vocabulary selection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3487–3497.

Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. 2015. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*, pages 2285–2294.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Erik F Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2019. Q-bert: Hessian based ultra low precision quantization of bert. *arXiv preprint arXiv:1909.05840*.

Raphael Shu and Hideki Nakayama. 2018. Compressing word embeddings via deep compositional code learning. In *International Conference on Learning Representations*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. 2019. Small and practical bert models for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3623–3627.

Andrew Tulloch and Yangqing Jia. 2017. High performance ultra-low-precision convolutions on mobile devices. *arXiv preprint arXiv:1712.02427*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. *arXiv preprint arXiv:2002.02925*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*.

Sergey Zagoruyko and Nikos Komodakis. 2016. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*.

Yiren Zhou, Seyed-Mohsen Moosavi-Dezfooli, Ngai-Man Cheung, and Pascal Frossard. 2018. Adaptive quantization for deep neural network. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Diverse Adversaries for Mitigating Bias in Training

Xudong Han **Timothy Baldwin** **Trevor Cohn**
School of Computing and Information Systems
The University of Melbourne
Victoria 3010, Australia
`xudonghl@student.unimelb.edu.au`
`{tbaldwin,tcohn}@unimelb.edu.au`

Abstract

Adversarial learning can learn fairer and less biased models of language than standard methods. However, current adversarial techniques only partially mitigate model bias, added to which their training procedures are often unstable. In this paper, we propose a novel approach to adversarial learning based on the use of multiple *diverse* discriminators, whereby discriminators are encouraged to learn orthogonal hidden representations from one another. Experimental results show that our method substantially improves over standard adversarial removal methods, in terms of reducing bias and the stability of training.

1 Introduction

While NLP models have achieved great successes, results can depend on spurious correlations with protected attributes of the authors of a given text, such as gender, age, or race. Including protected attributes in models can lead to problems such as leakage of personally-identifying information of the author (Li et al., 2018a), and unfair models, i.e., models which do not perform equally well for different sub-classes of user. This kind of unfairness has been shown to exist in many different tasks, including part-of-speech tagging (Hovy and Søgaard, 2015) and sentiment analysis (Kiritchenko and Mohammad, 2018).

One approach to diminishing the influence of protected attributes is to use adversarial methods, where an encoder attempts to prevent a discriminator from identifying the protected attributes in a given task (Li et al., 2018a). Specifically, an adversarial network is made up of an attacker and encoder, where the attacker detects protected information in the representation of the encoder, and the optimization of the encoder incorporates two parts: (1) minimizing the main loss, and (2) maximizing the attacker loss (i.e., preventing protected attributes from being detected by the attacker). Preventing protected attributes from being detected tends to result in fairer models, as protected attributes will more likely be independent rather than confounding variables. Although this method leads to demonstrably less biased models, there are still limitations, most notably that significant protected information still remains in the model's encodings and prediction outputs (Wang et al., 2019; Elazar and Goldberg, 2018).

Many different approaches have been proposed to strengthen the attacker, including: increasing the discriminator hidden dimensionality; assigning different weights to the adversarial component during training; using an ensemble of adversaries with different initializations; and reinitializing the adversarial weights every t epochs (Elazar and Goldberg, 2018). Of these, the ensemble method has been shown to perform best, but independently-trained attackers can generally still detect private information after adversarial removal.

In this paper, we adopt adversarial debiasing approaches and present a novel way of strengthening the adversarial component via orthogonality constraints (Salzmann et al., 2010). Over a sentiment analysis dataset with racial labels of the document authors, we show our method to result in both more accurate and fairer models, with privacy leakage close to the lower-bound.[1]

2 Methodology

Formally, given an input x_i annotated with main task label y_i and protected attribute label g_i, a main task model M is trained to predict $\hat{y}_i = M(x_i)$, and an adversary, aka "discriminator", A is trained to predict $\hat{g}_i = A(h_{M,i})$ from M's last hidden layer representation $h_{M,i}$. In this paper, we treat

---

[1] Source code available at `https://github.com/HanXudong/Diverse_Adversaries_for_Mitigating_Bias_in_Training`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2760–2765
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

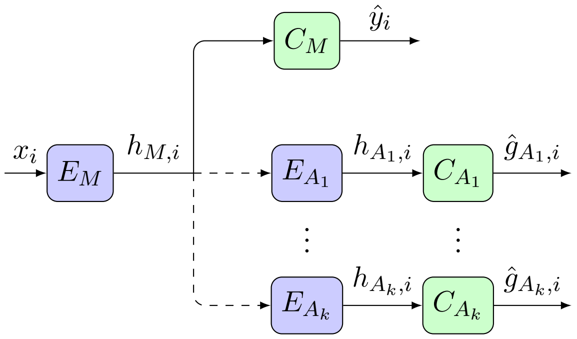


Figure 1: Ensemble adversarial method. Dashed lines denote gradient reversal in adversarial learning. The k sub-discriminators A_i are independently initialized. Given a single input x_i, the main task encoder computes a hidden representation $h_{M,i}$, which is used as the input to the main model output layer and sub-discriminators. From the k-th sub-discriminator, the estimated protected attribute label is $\hat{g}_{A_k,i} = C_{A_k}(E_{A_k}(h_{M,i}))$.

a neural network classifier as a combination of two connected parts: (1) an encoder E, and (2) a linear classifier C. For example, in the main task model M, the encoder E_M is used to compute the hidden representation $h_{M,i}$ from an input x_i, i.e., $h_{M,i} = E_M(x_i)$, and the decoder is used to make a prediction, $\hat{y}_i = C_M(h_{M,i})$. Similarly, for a discriminator, $\hat{g}_i = A(h_{M,i}) = C_A(E_A(h_{M,i}))$.

2.1 Adversarial Learning

Following the setup of Li et al. (2018a) and Elazar and Goldberg (2018) the optimisation objective for our standard adversarial training is:

$$\min_M \max_A \mathcal{X}(y, \hat{y}_M) - \lambda_{\text{adv}} \mathcal{X}(g, \hat{g}_A),$$

where $\mathcal{X}$ is cross entropy loss, and λ_{adv} is the trade-off hyperparameter. Solving this minimax optimization problem encourages the main task model hidden representation h_M to be informative to C_M and uninformative to A. Following Ganin and Lempitsky (2015), the above can be trained using stochastic gradient optimization with a gradient reversal layer for $\mathcal{X}(g, \hat{g}_A)$.

2.2 Differentiated Adversarial Ensemble

Inspired by the ensemble adversarial method (Elazar and Goldberg, 2018) and domain separation networks (Bousmalis et al., 2016), we present *differentiated adversarial ensemble*, a novel means of strengthening the adversarial component. Figure 1 shows a typical ensemble architecture where k sub-discriminators are included in the adversarial component, leading to an averaged adversarial regularisation term:

$$-\frac{\lambda_{\text{adv}}}{k} \sum_{j \in \{1,\ldots,k\}} \mathcal{X}(g, \hat{g}_{A_j}).$$

One problem associated with this ensemble architecture is that it cannot ensure that different sub-discriminators focus on different aspects of the representation. Indeed, experiments have shown that sub-discriminator ensembles can weaken the adversarial component (Elazar and Goldberg, 2018). To address this problem, we further introduce a difference loss (Bousmalis et al., 2016) to encourage the adversarial encoders to encode different aspects of the private information. As can be seen in Figure 1, $h_{A_k,i}$ denotes the output from the k-th sub-discriminator encoder given a hidden representation $h_{M,i}$, i.e., $h_{A_k,i} = E_{A_k}(h_{M,i})$.

The difference loss encourages orthogonality between the encoding representations of each pair of sub-discriminators:

$$\mathcal{L}_{\text{diff}} = \lambda_{\text{diff}} \sum_{i,j \in \{1,\ldots,k\}} \left\| {h_{A_i}}^{\intercal} h_{A_j} \right\|_F^2 \mathbb{1}(i \neq j),$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm.

Intuitively, sub-discriminator encoders must learn different ways of identifying protected information given the same input embeddings, resulting in less biased models than the standard ensemble-based adversarial method. According to Bousmalis et al. (2016), the difference loss has the additional advantage of also being minimized when hidden representations shrink to zero. Therefore, instead of minimizing the difference loss by learning rotated hidden representations (i.e., the same model), this method biases adversaries to have representations that are a) orthogonal, and b) low magnitude; the degree to which is given by weight decay of the optimization function.

2.3 INLP

We include Iterative Null-space Projection ("INLP": Ravfogel et al. (2020)) as a baseline method for mitigating bias in trained models, in addition to standard and ensemble adversarial methods. In INLP, a linear discriminator (A_{linear}) of the protected attribute is iteratively trained from pre-computed fixed hidden representations (i.e., h_M) to project them onto the linear discriminator's

Model	Accuracy↑	TPR Gap↓	TNR Gap↓	Leakage@$\mathbf{h}$↓	Leakage@$\hat{y}$↓
Random	50.00±0.00	0.00±0.00	0.00±0.00	—	—
Fixed Encoder	61.44±0.00	0.52±0.00	17.97±0.00	92.07±0.00	86.93±0.00
Standard	71.59±0.05	31.81±0.29	48.41±0.27	85.56±0.20	70.09±0.19
INLP	68.54±1.05	25.13±2.31	40.70±5.02	**66.64**±0.87	66.19±0.79
Adv Single Discriminator	74.25±0.39	13.01±3.83	28.55±3.60	84.33±0.98	61.48±2.17
Adv Ensemble	74.08±0.99	12.04±3.50	31.76±3.19	85.31±0.51	63.23±3.62
Differentiated Adv Ensemble	**74.52**±0.28	**8.42**±1.84	**24.74**±2.07	84.52±0.50	**61.09**±2.32

Table 1: Evaluation results ± standard deviation (%) on the test set, averaged over 10 runs with different random seeds. **Bold** = best performance. "↑" and "↓" indicate that higher and lower performance, resp., is better for the given metric. Leakage measures the accuracy of predicting the protected attribute, over the final hidden representation $\mathbf{h}$ or model output $\hat{y}$. Since the Fixed Encoder is not designed for binary sentiment classification, we merge the original 64 labels into two categories based on the results of hierarchical clustering.

null-space, $h_M^* = P_{N(A_{\text{linear}})} h_M$, where $P_{N(A_{\text{linear}})}$ is the null-space projection matrix of A_{linear}. In doing so, it becomes difficult for the protected attribute to be linearly identified from the projected hidden representations (h_M^*), and any linear main-task classifier (C_M^*) trained on h_M^* can thus be expected to make fairer predictions.

3 Experiments

Fixed Encoder Following Elazar and Goldberg (2018) and Ravfogel et al. (2020), we use the DeepMoji model (Felbo et al., 2017) as a fixed-parameter encoder (i.e. it is not updated during training). The DeepMoji model is trained over 1246 million tweets containing one of 64 common emojis. We merge the 64 emoji labels output by DeepMoji into two super-classes based on hierarchical clustering: 'happy' and 'sad'.

Models The encoder E_M consists of a fixed pretrained encoder (DeepMoji) and two trainable fully connected layers ("Standard" in Table 1). Every linear classifier (C) is implemented as a dense layer.

For protected attribute prediction, a discriminator (A) is a 3-layer MLP where the first 2 layers are collectively denoted as E_A, and the output layer is denoted as C_A.

TPR-GAP and TNR-GAP In classification problems, a common way of measuring bias is TPR-GAP and TNR-GAP, which evaluate the gap in the True Positive Rate (TPR) and True Negative Rate (TNR), respectively, across different protected attributes (De-Arteaga et al., 2019). This measurement is related to the criterion that the prediction $\hat{y}$ is conditionally independent of the protected attribute g given the main task label y (i.e., $\hat{y} \perp g | y$). Assuming a binary protected attribute, this conditional independence requires $\mathbb{P}\{\hat{y}|y, g = 0\} = \mathbb{P}\{\hat{y}|y, g = 1\}$, which implies an objective that minimizes the difference (GAP) between the two sides of the equation.

Linear Leakage We also measure the leakage of protected attributes. A model is said to leak information if the protected attribute can be predicted at a higher accuracy than chance, in our case, from the hidden representations the fixed encoder generates. We empirically quantify leakage with a linear support vector classifier at two different levels:

- Leakage@$\mathbf{h}$: the accuracy of recovering the protected attribute from the output of the final hidden layer after the activation function (h_M).
- Leakage@$\hat{y}$: the accuracy of recovering the protected attribute from the output $\hat{y}$ (i.e., the logits) of the main model.

Data We experiment with the dataset of Blodgett et al. (2016), which contains tweets that are either African American English (AAE)-like or Standard American English (SAE)-like (following Elazar and Goldberg (2018) and Ravfogel et al. (2020)). Each tweet is annotated with a binary "race" label (on the basis of AAE or SAE) and a binary sentiment score, which is determined by the (redacted) emoji within it.

In total, the dataset contains 200k instances, perfectly balanced across the four race–sentiment combinations. To create bias in the dataset, we follow previous work in skewing the training data to generate race–sentiment combinations (AAE–

happy, SAE–happy, AAE–sad, and SAE–sad) of 40%, 10%, 10%, and 40%, respectively. Note that we keep the test data unbiased.

Training Details All models are trained and evaluated on the same training/test split. The Adam optimizer (Kingma and Ba, 2015) is used with learning rates of 3×10^{-5} for the main model and 3×10^{-6} for the sub-discriminators. The mini-batch size is set to 1024. Sentence representations (2304d) are extracted from the DeepMoji encoder. The hidden size of each dense layer is 300 in the main model, and 256 in the sub-discriminators. We train M for 60 epochs and each A for 100 epochs, keeping the checkpoint model that performs best on the dev set. Similar to Elazar and Goldberg (2018), hyperparameters (λ_{adv} and λ_{diff}) are tuned separately rather than jointly. λ_{adv} is tuned to 0.8 based on the standard (single-discriminator) adversarial learning method, and this setting is used for all other adversarial methods. When tuning λ_{adv}, we considered both overall performance and bias gap (both over the dev data). Since adversarial training can increase overall performance while decreasing the bias gap (see Figure 2), we select the adversarial model that achieves the best task performance. For adversarial ensemble and differentiated models, we tune the hyperparameters (number of sub attackers and λ_{diff}) to achieve a similar bias level while getting the best overall performance. To compare with a baseline ensemble method with a similar number of parameters, we also report results for an adversarial ensemble model with 3 sub-discriminators. The scalar hyperparameter of the difference loss (λ_{diff}) is tuned through grid search from 10^{-4} to 10^{4}, and set to $10^{3.7}$. For the INLP experiments, fixed sentence representations are extracted from the same data split. Following Ravfogel et al. (2020), in the INLP experiments, both the discriminator and the classifier are implemented in scikit-learn as linear SVM classifiers (Pedregosa et al., 2011). We report Leakage@$\hat{y}$ for INLP based on the predicted confidence scores, which could be interpreted as logits, of the linear SVM classifiers.

Results and Analysis Table 1 shows the results over the test set. Training on a biased dataset without any fairness restrictions leads to a biased model, as seen in the Gap and Leakage results for the Standard model. Consistent with the findings of Ravfogel et al. (2020), INLP can only reduce bias at the expense of overall performance. On the other hand, the Single Discriminator and Adv(ersarial) Ensemble baselines both enhance accuracy and reduce bias, consistent with the findings of Li et al. (2018a).

Compared to the Adv Ensemble baseline, incorporating the difference loss in our method has two main benefits: training is more stable (results have smaller standard deviation), and there is less bias (the TPR and TNR Gap are smaller). Without the orthogonality factor, $\mathcal{L}_{\text{diff}}$, the sub-discriminators tend to learn similar representations, and the ensemble degenerates to a standard adversarial model. Simply relying on random initialization to ensure sub-discriminator diversity, as is done in the Adv Ensemble method, is insufficient. The orthogonality regularization in our method leads to more stable and overall better results in terms of both accuracy and TPR/TNR Gap.

As shown in Table 1, even the Fixed Encoder model leaks protected information, as a result of implicit biases during pre-training. INLP achieves significant improvement in terms of reducing linear hidden representation leakage. The reason is that Leakage@$\mathbf{h}$ is directly correlated with the objective of INLP, in minimizing the linear predictability of the protected attribute from the $\mathbf{h}$. Adversarial methods do little to mitigate Leakage@$\mathbf{h}$, but substantially decrease Leakage@$\hat{y}$ in the model output. However, both types of leakage are well above the ideal value of 50%, and therefore none of these methods can be considered as providing meaningful privacy, in part because of the fixed encoder. This finding implies that when applying adversarial learning, the pretrained model needs to be fine-tuned with the adversarial loss to have any chance of generating a truly unbiased hidden representation. Despite this, adversarial training does reduce the TPR and TNR Gap, and improves overall accuracy, which illustrates the utility of the method for both bias mitigation and as a form of regularisation.

Overall, our proposed method empirically outperforms the baseline models in terms of debiasing, with a better performance–fairness trade-off.

Robustness to λ_{adv} We first evaluate the influence of the trade-off hyperparameter λ_{adv} in adversarial learning. As can be seen from Figure 2, λ_{adv} controls the performance–fairness trade-off. Increasing λ_{adv} from 10^{-2} to around 10^{-0}, TPR Gap and TNR Gap consistently decrease, while the accuracy of each group rises. To balance up accu-

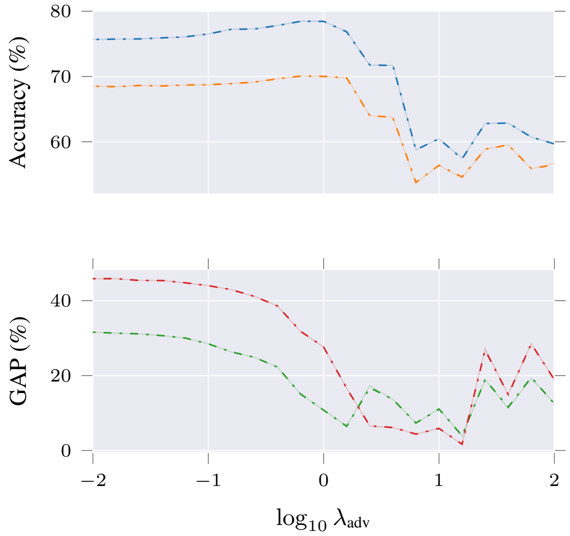


Figure 2: λ_{adv} sensitivity analysis, averaged over 10 runs for a single discriminator adversarial model. Main task accuracy of group *SAE* (blue) and *AAE* (orange), *TPR-GAP* (green), and *TNR-GAP* (red) are reported.

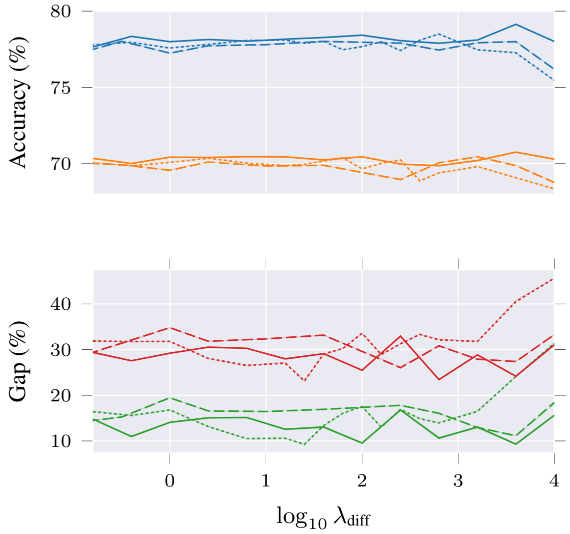


Figure 3: λ_{diff} sensitivity analysis for differentiated adversarial models with 3 (“——”), 5 (“- - - -”), and 8 (“········”) sub-discriminators, in terms of the main task accuracy of group *SAE* (blue) and *AAE* (orange), and *TPR-GAP* (green) and *TNR-GAP* (red).

racy and fairness, we set λ_{adv} to $10^{-0.1}$. We also observe that an overly large λ_{adv} can lead to a more biased model (starting from about $10^{1.2}$).

Robustness to λ_{diff} Figure 3 presents the results of our model with different λ_{diff} values, for $N \in \{3, 5, 8\}$ sub-discriminators.

First, note that when λ_{diff} is small (i.e., the left side of Figure 3), our Differentiated Adv Ensemble model generalizes to the standard Adv Ensemble model. For differing numbers of sub-discriminators, performance is similar, i.e., increasing the number of sub-discriminators beyond 3 does not improve results substantially, but does come with a computational cost. This implies that an Adv Ensemble model learns approximately the same thing as larger ensembles (but more efficiently), where the sub-discriminators can only be explicitly differentiated by their weight initializations (with different random seeds), noting that all sub-discriminators are otherwise identical in architecture, input, and optimizer.

Increasing the weight of the difference loss through λ_{diff} has a positive influence on results, but an overly large value makes the sub-discriminators underfit, and both reduces accuracy and increases TPR/TNR Gap. We observe a negative correlation between N and λ_{diff}, the main reason being that $\mathcal{L}_{\text{diff}}$ is not averaged over N and as a result, a large N and λ_{diff} force the sub-discriminators to pay too much attention to orthogonality, impeding their ability to bleach out the protected attributes.

Overall, we empirically show that λ_{diff} only needs to be tuned for Adv Ensemble, since the results for different Differentiated Adv models for a given setting achieve similar results. I.e., λ_{diff} can safely be tuned separately with all other hyperparameters fixed.

4 Conclusion and Future Work

We have proposed an approach to enhance sub-discriminators in adversarial ensembles by introducing a difference loss. Over a tweet sentiment classification task, we showed that our method substantially improves over standard adversarial methods, including ensemble-based methods.

In future work, we intend to perform experimentation over other tasks. Theoretically, our approach is general-purpose, and can be used not only for adversarial debiasing but also any other application where adversarial training is used, such as domain adaptation (Li et al., 2018b).

Acknowledgments

We thank Lea Frermann, Shivashankar Subramanian, and the anonymous reviewers for their helpful feedback and suggestions.

References

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems*, pages 343–351.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 120–128.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, pages 1180–1189.

Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 483–488.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018a. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018b. What's in a domain? learning domain-robust text representations using adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 474–479.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

Mathieu Salzmann, Carl Henrik Ek, Raquel Urtasun, and Trevor Darrell. 2010. Factorized orthogonal latent spaces. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 701–708.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5310–5319.

‘Just because you are right, doesn’t mean I am wrong’: Overcoming a Bottleneck in the Development and Evaluation of Open-Ended Visual Question Answering (VQA) Tasks

Man Luo, Shailaja Keyur Sampat, Riley Tallman, Yankai Zeng,
Manuha Vancha, Akarshan Sajja, Chitta Baral
Arizona State University, Tempe, AZ, USA
`{mluo26,ssampa17,rtallman,yzeng55,mvancha,asajja,chitta}@asu.edu`

Abstract

GQA (Hudson and Manning, 2019) is a dataset for real-world visual reasoning and compositional question answering. We found that many answers predicted by the best vision-language models on the GQA dataset do not match the ground-truth answer but still are semantically meaningful and correct in the given context. In fact, this is the case with most existing visual question answering (VQA) datasets where they assume only one ground-truth answer for each question. We propose Alternative Answer Sets (AAS) of ground-truth answers to address this limitation, which is created automatically using off-the-shelf NLP tools. We introduce a semantic metric based on AAS and modify top VQA solvers to support multiple plausible answers for a question. We implement this approach on the GQA dataset and show the performance improvements.

1 Introduction

One important style of visual question answering (VQA) task involves open-ended responses such as free-form answers or fill-in-the-blanks. The possibility of multiple correct answers and multi-word responses makes the evaluation of open-ended tasks harder, which has forced VQA datasets to restrict answers to be a single word or a short phrase. Despite enforcing these constraints, from our analysis of the GQA dataset (Hudson and Manning, 2019), we noticed that a significant portion of the visual questions have issues. For example, a question *“Who is holding the bat?”* has only one ground truth answer *“batter”* while other reasonable answers like *“batsman”*, *“hitter”* are not credited. We identified six different types of issues with the dataset and illustrated them in Table 1.

A large-scale human-study conducted by (Gurari and Grauman, 2017) on VQA (Antol et al., 2015) and VizWiz (Gurari et al., 2019) found that almost 50% questions in these datasets have multiple possible answers. datasets had similar observations. The above evidence suggests that it is unfair to penalize models if their predicted answer is correct in a given context but does not match the ground truth answer.

With this motivation, we leverage existing knowledge bases and word embeddings to generate Alternative Answer Sets (AAS) instead of considering visual questions to have fixed responses. Since initially obtained AAS are generated from multiple sources and observed to be noisy, we use textual entailment to verify semantic viability of plausible answers to make alternative answer sets more robust. We justify the correctness and quality of the generated AAS by human evaluation. We introduce a semantic metric based on AAS and train two vision-language models LXMERT (Tan and Bansal, 2019) and ViLBERT (Lu et al., 2019) on two datasets. The experimental results show that the AAS metric evaluates models’ performances more reasonably than the old metric. Lastly, we incorporate AAS in the training phase and show that it further improves on the proposed metric. Figure 2 gives an overview of our work.

2 Related Works

We discuss related works from two aspects, dataset creation and evaluation.

Dataset Creation-Level Large-scale VQA datasets are often curated through crowd-sourcing, where open-ended ground-truths are determined by majority voting or annotator agreement. The subjectivity in crowd-sourced datasets is well-studied in human-computer interaction literature- (Gurari and Grauman, 2016, 2017; Yang et al., 2018) etc. Ray et al. (2018) suggested creating a semantically-grounded set of questions

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2766–2771
April 19 - 23, 2021.

Issue Type	Definition	%
[1] Synonym and Hypernym	Synonym or hypernym of the ground-truth can also be considered as a correct answer for a given question-image pair.	9.1
[2] Singular/Plural	Singular or plural of the ground-truth can also be considered as a correct answer for a given question-image pair.	1.0
[3] Ambiguous Objects	Question refers to an object but the image contains multiple such objects that can lead to different possible answers.	5.8
[4] Multiple Correct Answers	If a given image-question pair is not precise, annotators might have different opinion which leads to multiple correct answers	7.0
[5] Missing Object(s)	Object referred in the question is not clearly visible in image.	4.3
[6] Wrong Label	The ground-truth answer to a question-image pair is incorrect.	6.7

Table 1: Six types of issues observed in the GQA dataset, their definition and their distribution observed in manual review of 600 samples from testdev balanced split. For example of each issue type, refer Figure 1.

for consistent answer predictions. (Bhattacharya et al., 2019) analyzed VQA and VizWiz datasets to present 9-class taxonomy of visual questions that suffer from subjectivity and ambiguity. Our analysis on GQA partially overlaps with this study. GQA dataset only provides one ground truth for each question; thus, we propose AAS to extend answers by phrases with close semantic meaning as the ground-truth answer.

Evaluation-Level For open-ended VQA tasks, the standard accuracy metric can be too stringent as it requires a predicted answer to exactly match the ground-truth answer. To deal with different interpretations of words and multiple correct answers, (Malinowski and Fritz, 2014) defined a WUPS scoring from lexical databases with Wu-Palmer similarity (Wu and Palmer, 1994). (Abdelkarim et al., 2020) proposed a soft matching metric based on wordNet (Miller, 1998) and word2vec (Mikolov et al., 2013). Different from them, we incorporate more advanced NLP resources tools to generate answer sets and rely on textural entailment to validate semantics for robustness. We propose a new metric to evaluate a system's response.

3 Analysis of GQA Dataset

GQA is a dataset for real-world visual reasoning and compositional question answering. Instead of human annotation, answers to the questions in GQA are generated from the scene graphs of images. We found that automatic creation leads to flaws in the dataset; thus, we manually analyze 600 questions from the testdev balanced split of GQA dataset, and identify six issues shown in Table 1. Figure 1 shows examples of each type of issue.

These issues are caused by (not limited to) three reasons. First, the dataset assumes only one ground truth so that other answers with semantic closed meaning are ignored. We propose AAS to address this issue to some extent and describe AAS in the next section. Second, some questions referring to multiple objects cause ambiguous meaning. We leverage scene graphs to address this issue and found 2.92% and 2.94% ambiguous questions in balanced training split and balanced validation split, respectively. These ambiguous questions can be removed from the dataset. Third, there are incorrect scene graph detections so that some questions and/or labels do not match with the given images. We plan to address these issues in our future work.

4 Alternative Answer Set

To credit answers with semantically close meaning as the ground-truth, we propose a workflow that can be visualized from Figure 2. Each item in VQA dataset consists of <I, Q, GT>, where I is an image, Q is a question, and GT is a ground-truth answer. We define an Alternative Answer Set (AAS) as a collection of phrases $\{A_1, A_2, A_3,.., A_n\}$ such that A_i replaced with GT is still a valid answer to the given Image-Question pair. We construct AAS for each unique ground-truth automatically from two knowledge bases: Wordnet (Miller, 1998) and ConcpetNet (Liu and Singh, 2004), two word embeddings: BERT (Devlin et al., 2018) and counter-fitting (Mrkšić et al., 2016). We assign a semantic score to each alternative answer by textural entailment and introduce the AAS metric.

[1] Synonym and Hypernym

Question: Who is holding the bat?
Ground-truth: batter
Explanation: 'batsman' is a synonym of 'batter'
SU-AAS: batter, batsman, hitter, ballplayer, player

[2] Singular/Plural

Question: Who is wearing the dress?
Ground-truth: women
Explanation: singular form 'woman' is also correct (there is only one woman in the picture)
SU-AAS: women, female, woman, people, adult female

[3] Ambiguous Objects

Question: Does the person in front of the cabinets have brunette color?
Ground-truth: Yes
Explanation: there are two people in front of the cabinet and it is not clear which person is being referred to
SU-AAS: yes

[4] Multiple Correct Answers

Question: Which place is it?
Ground-truth: road
Explanation: some person might answer 'road' and some might answer 'street'
SU-AAS: road, street, roadway, paved

[5] Missing Object(s)

Question: Does the marker look narrow?
Ground-truth: Yes
Explanation: the 'marker' is missing from the image
SU-AAS: yes

[6] Wrong Label

Question: Do the door and the logo have the same color?
Ground-truth: Yes
Explanation: the correct answer is 'no' as the door is white and logo is green.
SU-AAS: yes

Figure 1: Examples from GQA dataset for each issue type and SU-AAS i.e. AAS of ground-truth based on semantic union approach. SU-AAS can resolve Synonym and Hypernym, Singular/Plural, and Multiple Correct Answers for a given problem.

4.1 Semantic Union AAS

We take a union of four methods to find all alternative answers. For example, "stuffed animal" is semantic similar to "teddy bear", which appears in the AAS based on BERT but not in WordNet. However, the union might include phrases that we want to distinguish from the label like "man" is in the AAS of "woman" when using the BERT-based approach. For this reason, we employ the textural entailment technique to compute a semantic score of each alternative answer. For each label, we first obtain 50 sentences containing the ground-truth label from GQA dataset. We take each sentence as a premise, replace the label in this sentence with a phrase in its AAS as a hypothesis to generate an entailment score between 0-1. Specifically, we use publicly available RoBERTa (Liu et al., 2019) model trained on SNLI (Stanford Natural Language Inference) (Bowman et al., 2015) dataset for entailment computation. The semantic score of the alternative answer is the average of 50 entailment scores. If the semantic score is lower than the threshold of 0.5, then this alternative answer is thrown out. We choose 0.5 since it is the middle of 0 and 1.

Lastly, we sort the AAS by semantic score and keep the top K in the semantic union AAS, annotated by SU-AAS. We experiment with different values of K from 2 to 10, and decide K to be 6, a trade-off between accuracy and robustness. Note that the performance of textual entailment model is a contributing factor in obtaining quality AAS. Therefore, we recommend using the state-of-the-art entailment model when our proposed method is applied on other VQA datasets.

4.2 Evaluation Metric Based on AAS

We propose AAS metric and semantic score: given a question Q_i, an image I_i, the alternative answer set of GT_i denoted by $\mathrm{S}_{\mathrm{GT}_i}$, the prediction of model P_i is correct if and only if it is found in $\mathrm{S}_{\mathrm{GT}_i}$, and the score of P_i is $\mathrm{S}_{\mathrm{GT}_i}(\mathrm{P}_i)$, where $\mathrm{S}_{\mathrm{GT}_i}(\mathrm{P}_i)$ is the semantic score of P_i. Mathematically,

$$\mathrm{Acc}(\mathrm{Q}_i, \mathrm{I}_i, \mathrm{S}_{\mathrm{GT}_i}, \mathrm{P}_i) = \begin{cases} S_{\mathrm{GT}_i}(\mathrm{P}_i) & \text{if } \mathrm{P}_i \in \mathrm{S}_{\mathrm{GT}_i} \\ 0 & \text{else} \end{cases}$$

5 Experiments

In this section, we first show that the performance of vision-language models on two datasets is improved based on the AAS metric. Then, we describe our experiment to incorporate AAS with one model on GQA dataset. Last, we verify the correctness of AAS by human evaluation.

5.1 Baseline Methods

We select two top Vision-and-Language models, ViLBERT (Lu et al., 2019) and LXMERT (Tan

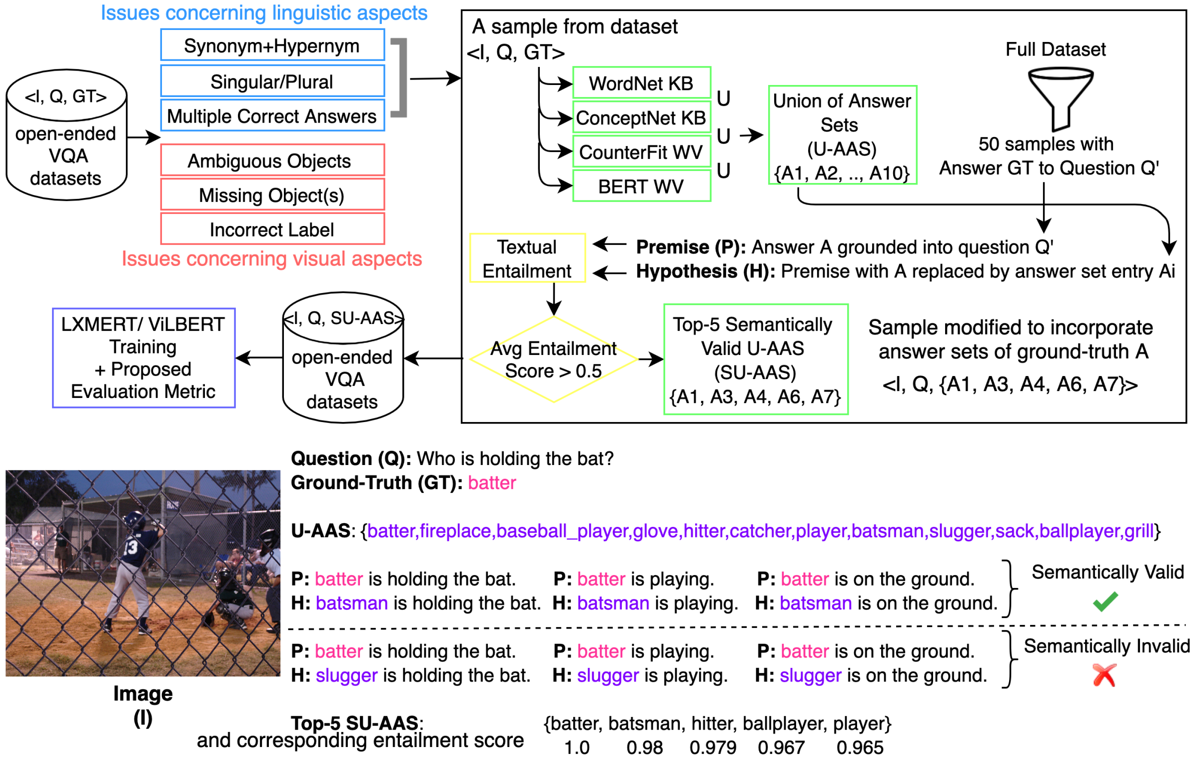


Figure 2: (top) The workflow for generating Alternative Answer Set (AAS) for VQA datasets (bottom) An example from GQA dataset showing semantically valid AAS for the answer 'batter' generated using above workflow

and Bansal, 2019) and evaluate their performances based on the AAS metric. From Table 2, we see that for the GQA dataset, LXMERT and ViLBERT have 4.49%, 4.26% improvements on union AAS metric separately. For VQA2.0 dataset, LXMERT and ViLBERT have 0.82%, 0.53% improvements on union AAS metric separately. It is expected that the improvement on VQA2.0 dataset is less than GQA since the former dataset already provides multiple correct answers. Figure 3 shows the impacts of the value K of Union AAS on the scores. From the figure, we see that when K increases from 2 to 6, the score gets increased significantly, and slightly when k increases from 6 to 9, but not increases more after K is 9. Since values 7 and 8 do not significantly improve the score, and the value 9 introduces noise, we take the top 6 as the SU-AAS.

5.2 Training with AAS

We incorporate SU-AAS of ground truth in training phase, so the model learns that more than one answer for a given example can be correct. We train LXMERT on GQA dataset with this objective.

Table 3 shows the results of LXMERT trained with AAS compared with the baseline. Not surprisingly, the performance evaluated on the original method drops because the model has a higher chance to predict answers in AAS, which are different from the ground truth, and thus the performance evaluated on SU-AAS metric increases.

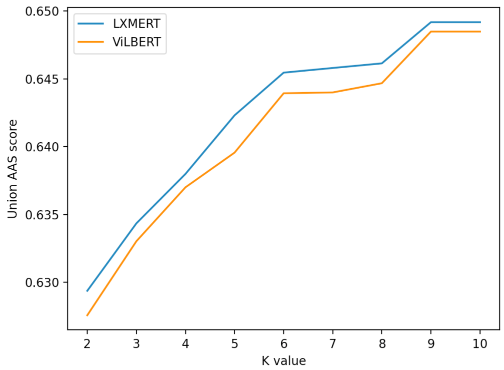


Figure 3: Union AAS score of different value of K

Dataset	Exact Matching Accuracy		SU-AAS Accuracy	
	LXMERT	LXMERT$_{AAS}$	LXMERT	LXMERT$_{AAS}$
GQA(testdev)	**60.06**	59.02	64.55	**65.22**

Table 3: Incorporate AAS in the training phase of LXMERT (LMXERT$_{AAS}$) on GQA dataset.

5.3 Evaluation of AAS

To validate the correctness of AAS, we measure the correlation between human judgment and AAS.

Dataset	Model	Original Metric	WordNet	BERT	CounterFit	ConceptNet	Union
GQA	LXMERT	60.06	61.79	62.69	62.75	63.58	**64.55**
(testdev)	ViLBERT	60.13	61.90	62.69	62.74	63.67	**64.39**
VQA	LXMERT	69.98	70.21	70.54	70.33	70.52	**70.80**
(valid)	ViLBERT	77.65	77.82	78.10	77.93	78.06	**78.28**

Table 2: The evaluation of two models on GQA and VQA with original metric and AAS based metrics.

Specifically, for each label of GQA, we take the SU-AAS and ask three annotators to justify if alternative answers in AAS can replace the label. If the majority of annotators agree upon, we keep the answer in the AAS, remove otherwise. In this way, we collect the human-annotated AAS. We compare the human-annotated AAS with each automatically generated AAS. We take the intersection over union (IoU) score to evaluate the correlation between automatic approach and human annotation: a higher IoU score means stronger alignment.

Method	WordNet	BERT	CounterFit	ConceptNet	Union
IoU%	48.25	56.18	58.95	58.39	80.5

Table 4: The IoU scores between human annotations and AAS based on five approaches.

6 Discussion and Conclusion

To evaluate a model from a semantic point of view, we define an alternative answer set (AAS). We develop a workflow to automatically create robust AAS for ground truth answers in the dataset using Textual Entailment. Additionally, we did human verification to assess the quality of automatically generated AAS. The high agreement score indicates that entailment model is doing a careful job of filtering relevant answers. From experiments on two models and two VQA datasets, we show the effectiveness of AAS-based evaluation using our proposed metric.

AAS can be applied to other tasks, for example, machine translation. BLEU(Papineni et al., 2002) score used to evaluate machine translation models incorporates an average of n-gram precision but does not consider the synonymy. Therefore, METEOR (Banerjee and Lavie, 2005) was proposed to overcome this problem. However, METEOR only relies on the synset of WordNet to get the synonyms. Our proposed AAS has the advantage of both knowledge base and word embeddings, which would help better evaluate translation tasks.

Acknowledgments

We are thankful to Tejas Gokhale for useful discussions and feedback on this work. We also thank anonymous reviewers for their thoughtful feedback. This work is partially supported by the National Science Foundation grant IIS-1816039.

References

Sherif Abdelkarim, Panos Achlioptas, Jiaji Huang, Boyang Li, Kenneth Church, and Mohamed Elhoseiny. 2020. Long-tail visual relationship recognition with a visiolinguistic hubless loss. *arXiv preprint arXiv:2004.00436*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *IEEE ICCV*, pages 2425–2433.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Nilavra Bhattacharya, Qing Li, and Danna Gurari. 2019. Why does a visual question have different answers?

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Danna Gurari and Kristen Grauman. 2016. Visual question: Predicting if a crowd will agree on the answer.

Danna Gurari and Kristen Grauman. 2017. Crowdverge: Predicting if people will agree on the answer to a visual question. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3511–3522.

Danna Gurari, Qing Li, Chi Lin, Yinan Zhao, Anhong Guo, Abigale Stangl, and Jeffrey P Bigham. 2019. Vizwiz-priv: A dataset for recognizing the presence and purpose of private visual information in images taken by blind people. In *IEEE CVPR*, pages 939–948.

Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *arXiv preprint arXiv:1902.09506*.

Hugo Liu and Push Singh. 2004. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Nikola Mrkšić, Diarmuid O Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892*.

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

Arijit Ray, Giedrius T Burachas, Karan Sikka, Anirban Roy, Avi Ziskind, Yi Yao, and Ajay Divakaran. 2018. Make up your mind: Towards consistent answer predictions in vqa models. In *European Conference on Computer Vision (ECCV), Workshops*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, ACL '94, page 133–138, USA. Association for Computational Linguistics.

Chun-Ju Yang, Kristen Grauman, and Danna Gurari. 2018. Visual question answer diversity. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*.

Better Neural Machine Translation by Extracting Linguistic Information from BERT

Hassan S. Shavarani **Anoop Sarkar**
School of Computing Science
Simon Fraser University
BC, Canada
{sshavara,anoop}@sfu.ca

Abstract

Adding linguistic information (syntax or semantics) to neural machine translation (NMT) has mostly focused on using point estimates from pre-trained models. Directly using the capacity of massive pre-trained contextual word embedding models such as BERT (Devlin et al., 2019) has been marginally useful in NMT because effective fine-tuning is difficult to obtain for NMT without making training brittle and unreliable. We augment NMT by extracting dense fine-tuned vector-based linguistic information from BERT instead of using point estimates. Experimental results show that our method of incorporating linguistic information helps NMT to generalize better in a variety of training contexts and is no more difficult to train than conventional Transformer-based NMT.

1 Introduction

Probing studies into large contextual word embeddings such as BERT (Devlin et al., 2019) have shown that these deep multi-layer models essentially reconstruct the traditional NLP pipeline capturing syntax and semantics (Jawahar et al., 2019); information such as part-of-speech tags, constituents, dependencies, semantic roles, coreference resolution information (Tenney et al., 2019a,b) and subject-verb agreement information can be reconstructed from BERT embeddings (Goldberg, 2019). In this work, we wish to extract the relevant pieces of linguistic information related to various levels of syntax from BERT in the form of dense vectors and then use these vectors as linguistic "experts" that neural machine translation (NMT) models can consult during translation.

But can syntax help improving NMT? Linzen et al. (2016); Kuncoro et al. (2018); Sundararaman et al. (2019) have reported that learning grammatical structure of sentences can lead to higher levels of performance in NLP models. In particular, Sennrich and Haddow (2016) show that augmenting NMT models with explicit linguistic annotations improves translation quality.

BERT embeddings have been previously considered for improving NMT models. Clinchant et al. (2019) replace the encoder token embedding layer in a Transformer NMT model with BERT contextual embeddings. They also experiment with initializing all the encoder layers of the translation model with BERT parameters, in which case they report results on both freezing and fine-tuning the encoder parameters during training. In their experiments BERT embeddings can help with noisy inputs to the NMT model, but otherwise do not help improving NMT performance.

Imamura and Sumita (2019) suggest that replacing the encoder layer with BERT embeddings and fine-tuning BERT while training the decoder leads to a *catastrophic forgetting* phenomenon where useful information in BERT is lost due to the magnitude and number of updates necessary for training the translation decoder and fine-tuning BERT. They present a two-step optimization regime in which the first step freezes the BERT parameters and trains only the decoder while the next step fine-tunes the encoder (BERT) and the decoder at the same time. Yang et al. (2020) also try to address the *catastrophic forgetting* phenomenon by thinking of BERT as a teacher for the encoder of the neural translation model (student network) (Hinton et al., 2015). They propose a dynamic switching gate implemented as a linear combination of the encoded embeddings from BERT and the encoder of NMT. However these papers do not really focus on the linguistic information in BERT, but rather try to combine pre-trained BERT and NMT encoder representations.

Sundararaman et al. (2019) identify *part-of-speech*, *case*, and *sub-word position* as essential lin-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2772–2783
April 19 - 23, 2021.

guistic information to improve the quality of both BERT and the neural translation model. They extract each linguistic feature using the Viterbi output of separate models, embed the extracted linguistic information (similar to trained word embeddings) and append these vectors to the token embeddings. However, their model uses point estimates of the syntactic models and they do not use the linguistic information in BERT embeddings.

Weng et al. (2019) use multiple multi-layer perceptron (MLP) modules to combine the information from different layers of BERT into the translation model. To make the most out of the fused information, they also alter the translation model training objective to contain auxiliary knowledge distillation (Hinton et al., 2015) parts concerned with the information coming from the pre-trained language model. Zhu et al. (2020) also inject BERT into all layers of the translation model rather than only input embeddings. Their model uses an attention module to dynamically control how each layer interacts with the representations. In both of these works, the training of the Transformer for NMT becomes quite brittle and is prone to diverge to local optima.

In this paper, we propose using pre-trained BERT as a source of linguistic information rather than a source of frozen pre-trained contextual embedding. We identify components of the BERT embeddings that correspond to different types of linguistic information such as part-of-speech, etc. and fine-tune dense vector embeddings for these linguistic aspects of the input and use them within an NMT model. Our approach does not radically complicate the Transformer NMT model training process both in terms of time and hardware requirements and also in terms of training difficulty (avoids bad local optima).

Our contributions are as follows: (1) A method of linguistic information extraction from BERT which needs supervision while training but works without supervision afterwards. (2) An easily trainable procedure for integrating the extracted information into the translation model. (3) Evaluation of the proposed model on small, medium and large translation datasets.

The source code and trained aspect extractors are available at https://github.com/sfu-natlang/SFUTranslate and our experiments can be replicated using scripts under `resources/exp-scripts/aspect_exps`.

2 NMT and BERT

Machine translation is the problem of transforming an input utterance sequence X in source language l_f into another utterance sequence Y (possibly with varying length) in target language l_e. Machine translation models search among all possible sequences in target language to find the most probable sequence based on the probability distribution of Equation 1.

$$P(y|X, y \in l_e) = \prod_{i=0}^{|max\ len|} p(y_i|X, y_0, ..., y_{i-1}) \tag{1}$$

Neural machine translation (NMT) tries to model the probability distribution $p(y|X)$ using neural networks by taking advantage of deep learning techniques. Transformers (Vaswani et al., 2017) are one type of encoder-decoder neural networks used for translation tasks. In Transformers, the input (in one-hot format) is passed through N layers of encoder and N layers of decoder. In each layer, the layer input passes through multiple attention heads (h heads; each considered a specialist in a different sentence-level linguistic attribute) and then gets transformed to the input for the next layer using a two layer feed-forward perceptron module with input size of d_{model} and hidden layer size of d_{ff}. The final probability distribution $p(y|X)$ is generated using an affine transformation applied to the output of the last feed-forward module in the N^{th} decoder layer. Please see (Vaswani et al., 2017) for further details.

BERT (Devlin et al., 2019) adopts the encoder part of the transformer model and requires training it on large amounts of text data using a *masked language model* objective over sub-words $p(y_i|X, y_0, ..., y_{i-1}, y_{i+1}, ..., y_{max\ len})$ instead of guessing the next sub-word $p(y_i|X, y_0, ..., y_{i-1})$. This bidirectional context turns BERT into a provider of strong contextual sub-word embeddings in many languages. These massively overparameterized neural networks have revolutionized many different NLP tasks. Effective application of BERT in NMT has been studied in a number of contemporary research projects; Language Modeling, Named Entity Recognition, Question Answering, Natural Language Inference, Text Classification (Devlin et al., 2019), and Question Generation (Chan and Fan, 2019). We approach this problem from the novel perspective of extracting linguistic information encoded in BERT and applying such information in NMT.

3 Linguistic Aspect Extraction from BERT

Since BERT contextual embeddings contain a variety of information (linguistic and non-linguistic), extraction of relevant information plays an important role in further improvement of the downstream tasks. In the rest of this section, we define *aspect vectors* as single-purpose dense vectors of extracted linguistic information from BERT, discuss how aspect vectors can be extracted, and explain how to integrate aspect vectors into NMT.

3.1 Aspect Vectors

To start the information extraction process, we initially need to choose a limited (desired) set of linguistic attributes to look for in BERT embeddings. This attribute set can contain a number of linguistic aspects (e.g. part-of-speech). Each linguistic aspect itself will be defined over a possible aspect tag set (e.g. the set of $\{NOUN, ADJ, ...\}$ in part-of-speech). In this paper, we show a linguistic attribute set with $\mathbb{A}$, show a generic aspect with a and point to its relative tag set with t_a.

Given the definition of a linguistic aspect and inspired by the information bottleneck idea (Tishby and Zaslavsky, 2015), we define an *aspect vector* as a single-purpose dense vector extracted from BERT and containing information about a certain linguistic aspect of a particular (sub-word) token in the input sequence. Aspect vectors can be interpreted as feature values equivalent to a specific key (aspect).

3.2 Aspect Vector Extraction

For each embedding vector $\mathbf{E}$ and linguistic aspect a, we define M_a as an aspect-extraction function where $\mathbf{e}_a = M_a(\mathbf{E})$ is a single-purpose dense vector containing maximum aspect information and minimum irrelevant other information.

We ensure the aspect encoding power of $\mathbf{e}_a$ by retrieving its equivalent tag in t_a using a classifier. The aspect prediction loss for a linguistic attribute set $\mathbb{A}$ of size n can be calculated as the average cross entropy loss ($\mathcal{L}_{CE}$) between the classifier prediction and the expected aspect tags for each aspect (Equation 2).

$$\mathcal{L}_a = \frac{1}{n}\sum_{i=0}^{|n|}\mathcal{L}_{CE}^{i} \qquad (2)$$

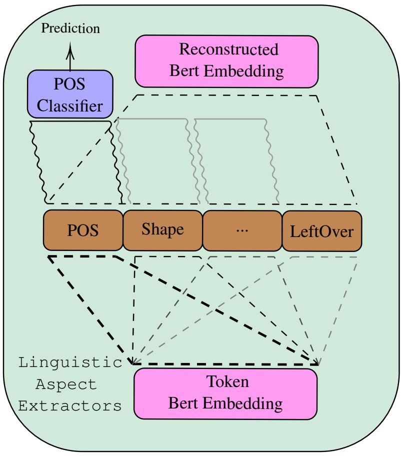


Figure 1: Schematic Aspect Extraction from BERT

We also ensure information integrity[1] of $\mathbf{e}_a$ by concatenating all the aspects (in addition to a "*left-over*" aspect equivalent to all the other non-interesting information) and reconstructing the original embedding vector $\mathbf{E}$ from them[2] in reconstruction vector $\mathbf{R}$. The reconstruction loss ($\mathcal{L}_r$) for the extracted aspect vectors can be calculated as the euclidean distance of the reconstruction vector $\mathbf{R}$ and the original embedding vector $\mathbf{E}$ (Equation 3).

$$\mathcal{L}_r = ||\mathbf{R} - \mathbf{E}||^2 \qquad (3)$$

In addition, since our aspect extractor is similar in architecture to a multi-head attention module (with a difference in the fact that we know what exactly each head will be responsible for), to prevent learning redundant representations (Michel et al., 2019), we add the average euclidean similarity ($\mathcal{L}_s$) of each pair of aspect vectors to the training loss function (Equation 4).

$$\mathcal{L}_s = 1 - \left(\frac{1}{n(n-1)}\sum_{i=0}^{|n|}\sum_{j\neq i=0}^{|n|} ||e_i - e_j||^2\right) \qquad (4)$$

The aspect extractor will be trained over the accumulation of the three mentioned loss components (Equation 5). Figure 1 demonstrates different parts of the aspect extractor and their connections.

$$\mathcal{L}_{fe} = \mathcal{L}_a + \mathcal{L}_r + \mathcal{L}_s \qquad (5)$$

---

[1] We don't expect M_a to change the information inside $\mathbf{E}$ but rather to extract the relevant information.

[2] This idea is analogous to stack-propagation (Zhang and Weiss, 2016) in which propagating the information loss for two tasks helps improving the quality of the encoded representations.

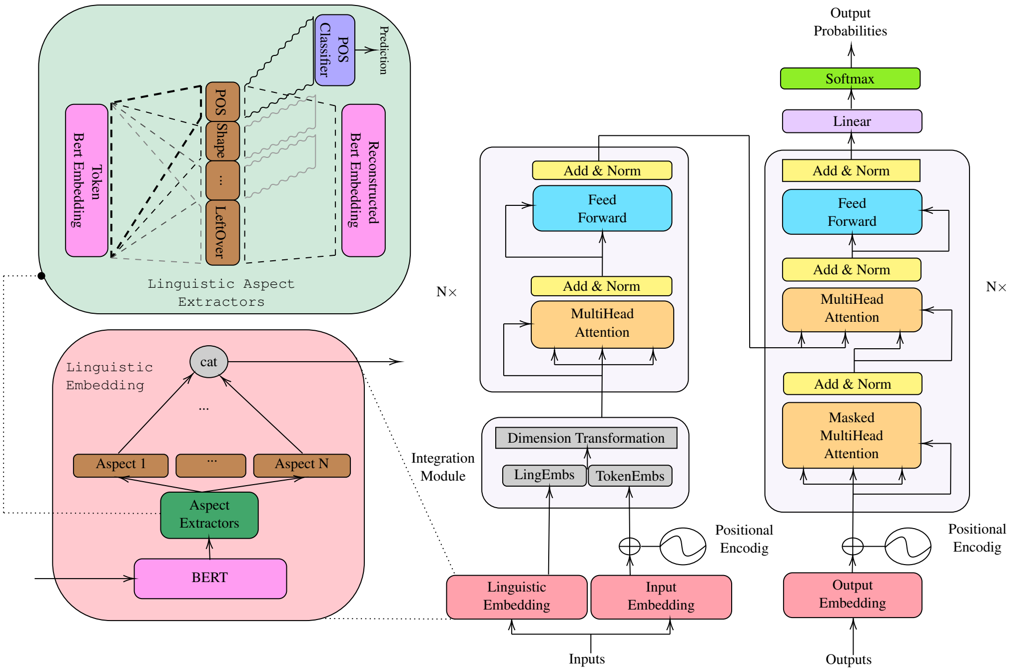


Figure 2: Integration of Extracted Aspect Vectors into NMT. The right hand side part of this figure is taken from Vaswani et al. (2017).

As another important point, a pre-trained BERT model has multiple encoder layers as well as an embedding layer. Choosing the proper layer which contains all of our desired aspects is not simply possible since different layers specialize in different linguistic aspects (Jawahar et al., 2019; Tenney et al., 2019a).

Therefore, as Peters et al. (2018) suggest, we define BERT embedding vector **E** as a weighted sum of all BERT layers (of size ℓ) using Equation 6 where α weights are learnable parameters and will be trained along with the other aspect extractor parameters.

$$\mathbf{E} = \sum_{j=0}^{\ell} \alpha_j \mathbf{E}_j^{BERT} \tag{6}$$

3.3 Integrating Aspect Vectors into NMT

Once the aspect vectors are created, we throw away the classifiers and the reconstruction layers and place the encoder part of our trained aspect extractor (the mapping from BERT contextual embeddings to aspect vectors) in an input integration module designed to augment the neural translation model input with aspect vectors[3].

The integration module (constructed using a two layer perceptron network) receives the concatenated aspect vectors (we call this concatenated vector a linguistic embedding[4]) and the token embedding (inherited from the Transformer model), and maps the linguistic embedding into a vector of the same size as the token embedding. Then, it projects the concatenation of both embeddings to a vector with the same size as the token embedding of the original Transformer model[5]. Figure 2 demonstrates this process.

4 Experiments

In this section, we initially examine our designed aspect extractor and report its classification accuracy scores. Next, we integrate the extracted aspect vectors into the neural machine translation framework as explained in Section 3.3 and study the effects of integrated vectors on the performance of the models.

4.1 Data

We choose three German (which has explicit and nuanced linguistic features) to English datasets in different data sizes to examine our proposed framework.

---

[3] We use the same sub-word model in pre-trained BERT to provide sub-word tokens to our NMT model.

[4] This embedding vector can be similar to what a factor token contains in Factored-NMT (García-Martínez et al., 2016) with a difference that it is generated in the space of linguistic aspects and does not need an embedding layer.

[5] This step is necessary to prevent any change in other parts of the model which would make comparison of the results unfair due to effects on the number of parameters and the learning capability of the model.

We use Multi30k (M30k)[6] as our small dataset. This dataset contains a multilingual set of image descriptions in German, English and French. Due to this reason, we also consider experimenting on German to French as our second small dataset. The M30k data contains 29K training sentences, 1014 validation sentences (val) and 1000 test sentences ($test$2016).

We take IWSLT (Cettolo et al., 2012)[7] as our medium sized dataset. The sentences in this dataset are quite different from M30k since they are composed from the transcriptions of TED talks as well as dialogues and lectures[8]. The IWSLT data contains 208K training sentences, 888 validation sentences (dev2010) and multiple test sets (tst2010 to tst2015 with 1568, 1433, 1700, 993, 1305, and 1080 sentences, respectively).

For the large data size, we consider WMT[9], a large (4.5M training sentences) set of parallel sentences from the proceedings of the European Parliament as well as web crawled news articles. We remove 0.05% of the training data (2290 sentences; lines with numbers divisible by 2000) and use it as the validation set (we call it wmt_val) and take $newstest$ data from 2014 to 2019 as our test sets (with 3003, 2169, 2999, 3004, 2998 and 1997 sentences, respectively).

We remove train data sentences longer than 100 words and uncase and normalize both side sentences using `MosesPunctNormalizer`[10] before tokenization. The reference side of the test data remains untouched in all the steps of our experiments.

4.2 Linguistic Aspect Vector Extraction

In this section, we study our linguistic aspect extractor training procedure and analyze the quality of the extracted aspect vectors.

We choose our linguistic attribute set ($\mathbb{A}$) as Sundararaman et al. (2019) suggest, however, we replace '*case*' with '*word-shape*'[11] since we believe the complete shape of the word is much more informative specially in sub-word settings. In addition, we consider a two-level hierarchy in part-of-speech tags to benefit from both higher accuracy in exploring the syntactic search space and lower model confusion in cases where the fine-grained tags are not helpful. Therefore, we consider coarse-grained and fine-grained *part-of-speech* (CPOS and FPOS), *word-shape* (WSH), and *sub-word position*[12] (SWP) to form our experimental linguistic attribute set ($\mathbb{A}$). Other linguistic attributes such as dependency parses or sentiment could be considered as aspects in our model but we leave that for future work.

We use the spaCy German tagger[13] model to acquire our intended linguistic aspect labels. Since spaCy is trained on word-level while BERT is trained on sub-word level, we had to align the sequences using a monotonic alignment algorithm (see Appendix A.1.1). The fine-grained part-of-speech tagger in spaCy[14] is pre-trained on TIGER Corpus[15] (Smith et al., 2003) and inherits its 55 fine-grained tags from TIGER treebank. The coarse-grained spaCy part-of-speech tagger has been trained by defining a direct mapping from 55 tags of the TIGER treebank to the 16 tags in the Universal Dependencies v2 POS tag set[16].

We use a 12-layer[17] German pre-trained BERT model for encoding the source sentences in aspect extractors. We use an uncased model as our translation model performs on lowercased data and the results are recased using the moses recaser so that the results are cased BLEU scores comparable to other systems[18]. We pass the BERT-encoded source sentences through a single perceptron middle layer of size 1000. We divide the output of this layer to

---

[6] AKA *Flickr30K* provided in task 1 of WMT17 multimodal machine translation, http://www.statmt.org/wmt17/multimodal-task.html

[7] 2017 was the last year that the data for this task got updated; https://wit3.fbk.eu/mt.php?release=2017-01-mted-test

[8] While the talks are quite polished, they still contain many verbal structures and sometimes even sounds (e.g. "`Imagine an engine going clack, clack, clack, clack, clack, clack, clack.`").

[9] Europarl+CommonCrawl+NewsCommentary https://www.statmt.org/wmt14/translation-task.html, please note that in the later years this training set remained the same, but ParaCrawl data was added to it. We do not use ParaCrawl data since it is quite noisy and we aim to limit the effects of uncontrolled variables in our training data. However, we report our results on all the test tests after 2014.

[10] https://github.com/alvations/sacremoses/

[11] Representing capitalization (changing alphabet to `x` or `X`), punctuation, and digits (changing digits to `d`). As an example for *word-shape*, the sub-word *##arxiv.* in the token '*myarxiv.org*' will turn to *##xxxxx.*.

[12] Encoding the word with one of the three labels "`Begin`", "`Inside`", or "`Single`".

[13] https://spacy.io/models/de

[14] SpaCy reports 96.52% accuracy for this model.

[15] https://www.ims.uni-stuttgart.de/

[16] https://universaldependencies.org/v2/postags.html

[17] Hidden state size of 768 with 12 heads; written in PyTorch and distributed by Wolf et al. (2019). You can find model configurations in https://github.com/dbmdz/berts.

[18] We recommend using a cased BERT model for translation systems that handle casing differently.

	Sub-word Level					Word Level			
	CPOS	FPOS	WSH	SWP	`#tokens`	CPOS	FPOS	WSH	`#tokens`
M30k	96.88	96.18	99.79	99.93	16096	97.95	97.34	99.74	12823
IWSLT	92.69	90.48	99.73	97.14	22687	94.84	93.07	99.69	19039
WMT	92.64	91.60	97.74	98.94	70139	94.86	94.01	97.38	55135

Table 1: F-1 scores acquired after training the aspect extractor on German side of parallel data and passing the validation sets of each data set through trained aspect extractors. The `#tokens` column shows the number of tokens in the validation set.

Aspect Extractor Training Data	FPOS	SWP
M30k	79.39	90.63
IWSLT	77.80	88.34
WMT	82.13	91.42
TIGER	84.64	92.64

Table 2: F-1 scores of fine-grained part-of-speech prediction of TIGER corpus test data (BERT encoded) fed to each of the trained aspect classifiers. The scores are calculated over a total of 7516 sub-word tokens in 358 test sentences of TIGER. Extractors trained on M30k, IWSLT, and WMT have not been provided with any part of TIGER before evaluation.

'*number of aspects + 1*' splits to form our desired aspect vectors (of size 200). Please see Appendix A.1.1 for more implementation details.

We train three different aspect extractors, one for each dataset and feed in the source sentences of the dataset to our model in batches of size 32 for 3 epochs[19]. Table 1 shows F-1 scores of classifying the validation set data using different aspect vectors after training the aspect extractors on the train set sentences. Please note that for calculating the word-level scores, in cases of disagreement between different sub-word tokens, the sub-word prediction of the first sub-word token has been counted as the prediction for the word label.

We also validate our trained (on M30k, IWSLT, and WMT) aspect extractors against the manual annotations of TIGER treebank with which the spaCy fine-grained part-of-speech tagger has been trained. We train an extra aspect extractor using the train set of TIGER corpus and test all four trained aspect extractors against TIGER data test set[20]. This experiment evaluates the absolute power of our simple feed-forward aspect extractors in performing the aspect classification task. Please note that our goal in this experiment is not to achieve the state-of-the-art fine-grained part-of-speech tagging results as our aspect extractors receive their input from BERT and do not directly access the tagged input sentences. Table 2 contains the results of comparison between predictions of different aspect extractor classifiers and TIGER gold labels.

4.3 Uniqueness of Information in Linguistic Aspect Vectors

Considering the high F-scores for each aspect category in each dataset (Table 1), we can conclude that our aspect extractor maximizes the relevant information extraction from BERT embeddings. The loss in Equation 4 maximizes the distance between aspect vectors. To test whether this leads to a diverse set of aspect vectors, each specialized to their own linguistic attributes, we consider each aspect category a, after training the aspect extractors. We take each of the other extracted aspect vectors a' (except the "*left-over*" vector) and use each of them to train a new classifier[21] that predicts the right class for category a based on aspect vector a'. This will test the correlation between the information in aspect vectors a' and the tags in category a. If the classification scores for this counterfactual test are high then our model has failed in fine-tuning each aspect vector to predict a particular linguistic aspect. We compare the classification scores to a trivial baseline: predict the most frequent class always. Table 3 shows the results of this counterfactual test on the aspect extractor trained on TIGER data. We can see that the average F-1 scores are very low when we use counterfactual aspect vectors to predict a linguistic aspect on which it was not fine-tuned (e.g. use aspect vector trained on part-

---

[19] Since the number of WMT sentences are much bigger, we stop training WMT aspect extractors when there is no improvement in aspect classification result (rounded to have 3 decimal places) of any label for at least 40 batches.

[20] We use `german_tiger_test_gs.conll` in the version of TIGER released in *2006 CoNLL Shared Task - Ten Languages*. Both train and test data are accessible through https://catalog.ldc.upenn.edu/LDC2015T11.

[21] We thank the anonymous reviewers for their valuable feedback on this procedure.

TIGER test	Sub-word Level			
	CPOS	FPOS	WSH	SWP
most frequent class	NOUN	NN	xxxx	single
percentage in total	27.12	27.07	39.07	59.92
average classification F-1	1.89	0.23	12.20	42.97
`#tokens`	$7516 \times 3 = 22548$			

Table 3: Classification scores of each aspect classifier when fed with other extracted aspect vectors. We expect the F-1 scores to be low so we can conclude that our aspect extractor truly excludes irrelevant information from each aspect.

of-speech to predict word shape). This shows that our training method fine-tunes each aspect vector to its linguistic task.

To validate the loss in Equation 3, we calculate the average euclidean distance of the aspect extractor reconstructed vectors and the original BERT embedding vectors[22] for M30k German to English dataset. We unit normalize each of the vectors for a score in $[0, 1]$. The average euclidean distance value of 0.1863 tells us that the reconstruction component of the aspect extractor is capable of reconstructing vectors that are close to the original embedding vectors.

4.4 Linguistic Aspect Integrated Machine Translation

After confirming the adequacy and uniqueness of linguistic information in aspect vectors, we integrate the encoder part of aspect extractors into the translation model and perform translation experiments on M30k, IWSLT, and WMT datasets. In our experiments, we compare our model to three baselines : (1) the vanilla transformer model (Vaswani et al., 2017) which does not use any external source of information, (2) the syntax-infused transformer model (Sundararaman et al., 2019) which explicitly embeds linguistic aspect labels and concatenates their embedding to the token embedding, (3) the transformer model with bert-freeze input setting (Clinchant et al., 2019) which replaces the input embedding layer of the encoder module in transformer with a fully pre-trained BERT model. Appendix A.1.2 provides the configurations and sufficient details for replication of our experiments in this section.

During each training trial, we perform 9 validation set evaluation steps (one after visiting each 10% of the data). In each step, the validation set is translated with the current state of the model (at the time of evaluation) and the generated sentences are detokenized and compared to the validation set reference data to produce sentence-level BLEU (Lin and Och, 2004) scores. The best scoring model throughout training is selected as the model with which the test set(s) are translated.

For M30k and IWSLT data sets, we train two separate models, one using the aspect vectors trained on the source side of its own training data (in-domain) and the other using the aspect vectors trained on the source side of WMT data (out-of-domain). We use cased BLEU (evaluated with the standard `mteval-v14.pl` script) and METEOR (Denkowski and Lavie, 2014) to compare different models. Tables 4 and 7 show the results of evaluating the models trained with different mentioned settings.

The evaluation results show that taking advantage of aspect vectors improves the accuracy of translating German to both English and French in M30k as well as German to English in IWSLT and WMT. Also, in majority of the cases WMT-trained aspect vectors have pushed the model to produce more accurate results since they contain more generalized information. Based on these results, we conjecture that aspect vectors trained on large out-of-domain data can be helpful in low-resource settings but we leave the examination of this idea for future work.

Aside from performance, our model is approximately 5 times faster than syntax-infused translation model (Sundararaman et al., 2019) while demanding less number of trainable parameters. Although it is not as fast as bert-freeze model (Clinchant et al., 2019) in large settings (because of the size of computations required for calculating the linguistic embedding), it is comparable in speed to bert-freeze in medium and small scale settings. Appendix A.2 contains some additional insights regarding how aspect vectors can help translation systems trained on different dataset sizes.

Tables 5 and 6 demonstrate some examples of cases where aspect vectors has been useful in improving the translation quality.

---

[22] Average results of Equation 3 for all the tokens in the train set.

a) M30k[†]	**German to English**				**German to French**			
	val	**test2016**	**`#param`**	**`runtime`[*]**	**val**	**test2016**	**`#param`**	**`runtime`[*]**
Vaswani et al. 2017	39.63	38.35	9.5 M	84 min	31.07	30.29	9.4 M	93 min
Sundararaman et al. 2019	40.03	38.32	13.9 M	514 min	32.55	32.71	13.6 M	504 min
Clinchant et al. 2019 (bert freeze)	40.07	39.73	9.1 M	99 min	33.83	33.15	9.0 M	104 min
Aspect Augmented +M30k asp. vectors	**40.47**	40.19	10.1 M	104 min	34.45	**34.42**	9.9 M	108 min
Aspect Augmented +WMT asp. vectors	38.72	**41.53**	10.1 M	102 min	**34.73**	34.28	9.9 M	118 min

b) IWSLT[†]	**dev2010**	**tst2010**	**tst2011**	**tst2012**	**tst2013**	**tst2014**	**tst2015**	**`#param`**	**`runtime`[*]**
Vaswani et al. 2017	27.69	27.93	31.88	28.15	29.59	25.66	26.76	18.4 M	172 min
Sundararaman et al. 2019	29.53	29.67	33.11	29.42	30.89	27.09	27.78	28.9 M	1418 min
Clinchant et al. 2019 (bert freeze)	30.31	30.00	34.20	30.04	31.26	27.50	27.88	18.0 M	212 min
Aspect Augmented +IWSLT asp. vectors	29.03	29.17	33.42	29.58	30.63	26.86	27.83	18.9 M	214 min
Aspect Augmented +WMT asp. vectors	**31.22**	**30.82**	**34.79**	**30.29**	**32.34**	**27.71**	**28.40**	18.9 M	211 min

c) WMT[†]	**wmt_val**	**nt2014**	**nt2015**	**nt2016**	**nt2017**	**nt2018**	**nt2019**	**`#param`**	**`runtime`[*]**
Vaswani et al. 2017	28.96	26.91	26.93	31.42	28.07	33.56	29.77	68.7 M	35 h
Sundararaman et al. 2019	28.56	27.80	26.93	30.44	28.63	33.87	30.48	93.8 M	258 h
Clinchant et al. 2019 (bert freeze)	28.63	27.54	27.15	31.69	28.30	33.89	**31.48**	69.1 M	33 h
Aspect Augmented +WMT asp. vectors	**28.98**	**28.05**	**27.58**	**32.29**	**29.07**	**34.74**	**31.48**	70.3 M	46 h

Table 4: Evaluated cased BLEU score (calculated using `mteval-v14.pl` script) results on M30k, IWSLT, and WMT datasets. `#param` represents the number of trainable parameters (size of BERT model parameters [110.5M] has not been added to the model size for the aspect augmented and bert-freeze models since BERT is not trained in these settings). `runtime` is the total time the training script has ran and includes time taken for reading the data and training the model from scratch (iterating over the instances for all the epochs).
All the baseline results are achieved using our re-implementation of the mentioned papers.
[*] We have used a single GeForce GTX 1080 GPU for M30k experiments and a single Titan RTX GPU for IWSLT and WMT experiments.
[†] Each experiment was repeated three times, and we report the average in this table.

5 Conclusion and Future Work

In this paper, we proposed a simple method of extracting linguistic information from BERT contextual embeddings and integrating them into neural machine translation framework. We showed that the linguistic aspect vectors provide the translation models with out-of-domain knowledge which not only improves the translation quality but also helps the model to better deal with out-of-vocabulary words. In the future, we would like to reconsider the integration module as a multi-head attention module, except that it will attend to different linguistic aspects of the current sub-word or sub-word tokens of a single word. Increasing the number of linguistic aspects (especially the use of syntactic dependencies and morphology) and studying the effects of the aspect vector size on the quality of generated translations are other directions of future research. We would also like to examine the effectiveness of aspect vectors trained on large out-of-domain data in low-resource settings and explore the effects of using linguistic aspect vectors in tasks other than machine translation.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. The research was partially supported by the Natural Sciences and Engineering Research Council of Canada grants NSERC RGPIN-2018-06437 and RGPAS-2018-522574 and a Department of National Defence (DND) and NSERC grant DGDND-2018-00025.

Source	Ihm **werde weiterhin vorgeworfen**, unerlaubt geheime Informationen weitergegeben zu haben.
Reference	He **is still accused of passing on** secret information without authorisation.
Vaswani et al. 2017	He has **also been accused of having** illegally **passed on** secret information.
Clinchant et al. 2019	He **continues to be accused of** fraudulently **passing on** secret information.
Sundararaman et al. 2019	He **is also accused of having pass** unauthorised secret information **on**.
Aspect Augmented NMT	He **is still accused of passing on** illegal secret information.
Source	Auto und Traktor krachen zusammen: Frau stirbt bei schrecklichem Unfall
Reference	Car and tractor **crash together**: woman **dies** in terrible accident
Vaswani et al. 2017	Car and traktor **cranes together**: women **die** in the event of a terrible accident.
Clinchant et al. 2019	Cars and tractors **are killing** women in the event of a terrible accident.
Sundararaman et al. 2019	Auto and tractor **are blowing together**: woman **dies** when the terrible accident occurs.
Aspect Augmented NMT	Car and tractor **crash together**: woman **dies** in terrible accidents.

Table 5: Examples of improved translation quality of WMT data where *part-of-speech* aspect vectors have helped the model choose better words both syntactically and semantically.

Source	Bucht die besten Hostels in **Ouarzazate** über Hostelsclub.
Reference	Book the best hostels in **Ouarzazate** with Hostelsclub.
Vaswani et al. 2017	Book the best hostels in **ouarzazazate** with Hostelsclub.
Clinchant et al. 2019	Book the best hostels in **Ouarzate** with Hostelsclub.
Sundararaman et al. 2019	Book the best hostels in **ouarzazazate** with Hostelsclub.
Aspect Augmented NMT	Book the best hostels in **Ouarzazate** with Hostelsclub.
Source	Die **Deutsche Bahn** will im kommenden Jahr die Kinzigtal-Bahnstrecke verbessern.
Reference	The **Deutsche Bahn** hopes to improve the Kinzigtal railway line in the coming year.
Vaswani et al. 2017	**The German Railway** wants to improve the Kinzig valley railway line next year.
Clinchant et al. 2019	**Christian Deutsche Bahn** intends to improve the Kinzig valley railway next year.
Sundararaman et al. 2019	**The German Railway** wants to improve the kinziggia railway line next year.
Aspect Augmented NMT	**Deutsche Bahn** wants to improve the Kinzig valley railway in the coming year.

Table 6: Examples of improved translation quality of WMT data where *word-shape* and *sub-word position* aspect vectors have helped the model choose a better sequence of sub-words when it faces out-of-vocabulary tokens.

a) M30k[†]	**German to English**		**German to French**	
	val	**test2016**	**val**	**test2016**
Vaswani et al. 2017	37.20	36.56	53.22	52.58
Sundararaman et al. 2019	38.14	37.13	54.18	54.37
Clinchant et al. 2019 (bert freeze)	38.44	37.42	55.10	54.50
Aspect Augmented +M30k asp. vectors	**39.22**	38.17	**56.21**	**56.40**
Aspect Augmented +WMT asp. vectors	38.90	**38.57**	56.12	55.98

b) IWSLT[†]	**dev2010**	**tst2010**	**tst2011**	**tst2012**	**tst2013**	**tst2014**	**tst2015**
Vaswani et al. 2017	31.82	31.99	34.57	32.65	32.49	30.65	31.13
Sundararaman et al. 2019	32.91	32.95	35.35	33.10	33.17	31.32	31.90
Clinchant et al. 2019 (bert freeze)	33.34	32.78	35.42	33.12	33.20	31.22	31.45
Aspect Augmented +IWSLT asp. vectors	32.86	32.86	35.38	33.43	33.23	31.37	31.87
Aspect Augmented +WMT asp. vectors	**33.78**	**33.56**	**36.14**	**33.51**	**33.98**	**31.86**	**32.37**

c) WMT[†]	**wmt_val**	**nt2014**	**nt2015**	**nt2016**	**nt2017**	**nt2018**	**nt2019**
Vaswani et al. 2017	**30.65**	33.80	33.70	**37.10**	34.44	37.81	36.05
Sundararaman et al. 2019	29.23	31.57	31.61	34.05	31.87	35.18	33.60
Clinchant et al. 2019 (bert freeze)	30.39	33.46	33.20	36.13	33.73	37.24	35.68
Aspect Augmented +WMT asp. vectors	30.61	**33.97**	**33.99**	37.01	**34.71**	**38.17**	**36.48**

Table 7: Evaluated METEOR score (calculated using the tool provided by Alon Lavie (https://www.cs.cmu.edu/∼alavie/METEOR/; version 1.5)) results on M30k, IWSLT, and WMT datasets.
[†] Each experiment was repeated three times, and we report the average in this table.

References

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy.

Ying-Hong Chan and Yao-Chung Fan. 2019. BERT for question generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 173–177, Tokyo, Japan. Association for Computational Linguistics.

Stephane Clinchant, Kweon Woo Jung, and Vassilina Nikoulina. 2019. On the use of BERT for neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 108–117, Hong Kong. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

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

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Mercedes García-Martínez, Loïc Barrault, and Fethi Bougares. 2016. Factored neural machine translation architectures. In *HAL archives ouvertes*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *Computation and Language Research Repository*, arXiv:1901.05287. Version 1.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Kenji Imamura and Eiichiro Sumita. 2019. Recycling a pre-trained BERT encoder for neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 23–31, Hong Kong. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Melbourne, Australia. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland. COLING.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Alexander Rush. 2018. The annotated transformer. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 52–60, Melbourne, Australia. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

George Smith et al. 2003. A brief introduction to the tiger treebank, version 1. *Potsdam Universität*.

Dhanasekar Sundararaman, Vivek Subramanian, Guoyin Wang, Shijing Si, Dinghan Shen, Dong Wang, and Lawrence Carin. 2019. Syntax-infused transformer and bert models for machine translation and natural language understanding. *Computation and Language Research Repository*, arXiv:1911.06156. Version 1.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Rongxiang Weng, Heng Yu, Shujian Huang, Shanbo Cheng, and Weihua Luo. 2019. Acquiring knowledge from pre-trained model to neural machine translation. *AAAI*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *Computation and Language Research Repository*, arXiv:1910.03771. Version 1.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Computation and Language Research Repository*, arXiv:1609.08144. Version 2.

Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Weinan Zhang, Yong Yu, and Lei Li. 2020. Towards making the most of bert in neural machine translation. *AAAI*.

Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1557–1566, Berlin, Germany. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating bert into neural machine translation. *ICLR 2020*.

A Appendices

A.1 Implementation Details

In this section, we provide implementation details that could not be placed in the main write-up due to space limitations, but we believe are quite helpful for replication of our work. We divide this section into two parts, one focused on linguistic aspect vector extraction (Section 4.2) and the other on linguistic aspect integrated machine translation (Section 4.4).

A.1.1 Linguistic Aspect Vector Extraction Implementation Details

The pre-trained spaCy tagger that we used in our experiments is trained on the word-level while the pre-trained BERT operates on sub-word level[23]. The two sequences need to be aligned, so we can assign aspect attributes to BERT sub-word tokens. Inspired by Gale and Church (1993), we align the two sequences using a heuristic divide-and-conquer monotonic alignment technique which finds the parts of the two sequences that are certainly equal and aligns the parts in between using recursive calls to itself[24].

Next, we explain how we implement the aspect extractors. We implement our aspect extractors using PyTorch framework and initialize them using Xavier initialization (Glorot and Bengio, 2010). We perform backpropagation using SGD (initial learning rate of 0.05, momentum value of 0.9, gradient clip norm of 5.0). To cope with inequality in the frequency of the different tags in each aspect tag set (t_a, see §3.1), we practice weighted

---

[23] The alignment is non-trivial e.g. “`hadn't`” is tokenized to “`hadn`” and “`'t`” by spaCy and to “`had`” and “`n't`” by BERT, causing many-to-many alignments.

[24] https://github.com/sfu-natlang/SFUTranslate/translate/readers/sequence_alignment.py

Dataset	**WMT**	**IWSLT**	**M30k**
N	6	6	4
d_{model}	512	256	256
d_{ff}	2048	512	512
h	8	4	4
opt factor	1	2	1
opt warmup	4000	8000	2000
grad accumulation	8	2	1
batch size*	4096	4096	2560
epochs	7	20	20

Table 8: The transformer model settings for each dataset given the training data size. "N" is the number of layers in both encoder and decoder. Please see §2 for more information about model parameters.
*The maximum number of sub-word tokens per batch.

backpropagation with weights proportional to the inverse frequency of each tag. We decay learning rate with a factor of 0.9 when the loss value stops improving.

A.1.2 Linguistic Aspect Integrated Machine Translation Implementation Details

We implement our baseline transformer model using the guidelines suggested by Rush (2018) in our translation toolkit SFUTranslate and extend it for implementing the aspect-augmented model as well as the syntax-infused transformer and transformer with bert-freeze input setting. Table 8 provides the configuration settings for each of the models used in our experiments.

We use the pre-trained WordPiece[25] (Schuster and Nakajima, 2012) tokenizer packaged and shipped with BERT (containing 31,102 sub-word tokens for German language) to tokenize the source side data, and tokenize the target side data with MosesTokenizer[26] followed by the same WordPiece tokenizer model, trained on target data, to split the target tokens into sub-tokens. We set the target side WordPiece vocabulary size to 30,000 sub-words for English and French. Our models share the vocabulary and embedding modules of both source and target (Press and Wolf, 2017) since both source and target are trained in sub-word space. The shared vocabulary sizes of M30k (German to English), M30k (German to French), IWSLT, and WMT are 16645, 16074, 40807, 47940, respectively.

We generate target sentences using beam search with beam size 4 and length normalization factor (Wu et al., 2016) of 0.6. We merge the WordPiece tokens in the generated sentences (a post-processing step to create words) and use MosesDetokenizer[27] to detokenize the generated outputs. We use Moses recaser[28] to produce cased translation outputs. We use `mteval-v14.pl` script for cased BLEU evaluation.

For all models, we set positional encoding max length to 4096, dropout to 0.1, loss prediction smoothing to 0.1, and initialize the models using Xavier initialization (Glorot and Bengio, 2010). We train all models using `NoamOpt` optimizer (Rush, 2018) and perform the gradient accumulation trick (Ott et al., 2018) with one update per a number of batches (Table 8; `grad accumulation`) to simulate larger batch sizes on a single GPU.

A.2 Additional Analysis of Linguistic Aspect Integrated Machine Translation Results

In this section, we analyze the results of our aspect integrated translation experiments. We provide our analysis in two parts, one for small and medium sized datasets and the other for large ones.

For smaller datasets (containing a few hundred thousand sentence pairs or less), the broader perspective of BERT knowledge is helpful in limiting the search space for the model. So using our technique, the translation model receives more information regarding the general use cases of (locally) rare words. Linguistic aspect vectors also help the model better understand less familiar (in comparison to what is frequent in its limited size training data) syntactic structures in input sentences. This is why we believe aspect vectors can be helpful in low-resource settings.

Improving models with large amounts of data (with several million sentence pairs) is a challenging task. The best practice in training neural translation models is to initialize the embedding module with small random values and let the model search through the parameter space to find the optimal parameter settings. Extracted aspect vectors, as an external source of monolingual knowledge on the source side, are a more reasonable starting point for large models than random initialization. Integrating aspect vectors thus helps these models find a better path towards the optimal point(s) and increases the chances of the model ending up in a more desirable point in search space.

---

[25] https://github.com/huggingface/tokenizers

[26] https://github.com/alvations/sacremoses

[27] https://github.com/alvations/sacremoses

[28] https://github.com/moses-smt/mosesdecoder

CLiMP: A Benchmark for Chinese Language Model Evaluation

Beilei Xiang,[1] **Changbing Yang,**[1] **Yu Li,**[1] **Alex Warstadt**[2] and **Katharina Kann**[1]
[1]University of Colorado Boulder, [2]New York University
`{beilei.xiang, changbing.yang, yuli9309}@colorado.edu`
`warstadt@nyu.edu`
`katharina.kann@colorado.edu`

Abstract

Linguistically informed analyses of language models (LMs) contribute to the understanding and improvement of these models. Here, we introduce the corpus of Chinese linguistic minimal pairs (CLiMP), which can be used to investigate what knowledge Chinese LMs acquire. CLiMP consists of sets of 1,000 minimal pairs (MPs) for 16 syntactic contrasts in Mandarin, covering 9 major Mandarin linguistic phenomena. The MPs are semi-automatically generated, and human agreement with the labels in CLiMP is 95.8%. We evaluate 11 different LMs on CLiMP, covering n-grams, LSTMs, and Chinese BERT. We find that classifier–noun agreement and verb complement selection are the phenomena that models generally perform best at. However, models struggle the most with the *bǎ* construction, binding, and filler-gap dependencies. Overall, Chinese BERT achieves an 81.8% average accuracy, while the performances of LSTMs and 5-grams are only moderately above chance level.

1 Introduction

Language models (LMs) are crucial parts of natural language processing (NLP) systems for a large variety of tasks, including summarization, machine translation, and dialog generation. More recently, they have become popular in the form of pretrained models,[1] which are then fine-tuned on downstream tasks and often obtain state-of-the-art performance (Peters et al., 2018; Devlin et al., 2019; Conneau et al., 2020). However, which linguistic phenomena language models can or cannot learn is still poorly understood for many languages.

Resources for the syntactic evaluation of LMs, such as BLiMP (Warstadt et al., 2020) have focused mainly on English, and non-English resources currently only cover a small set of phenomena (Mueller et al., 2020; Gulordava et al., 2018; Ravfogel et al., 2018). In order to spur the analysis and subsequent improvement of LMs in Chinese, we introduce the corpus of Chinese linguistic minimal pairs (CLiMP), which can be used to evaluate LMs' knowledge of Chinese grammar.

CLiMP consists of 16 individual datasets that are semi-automatically generated from grammar templates. Each set—or *paradigm*—contains 1,000 minimal pairs (MPs). Together, they cover 9 core linguistic phenomena in Chinese. Human agreement on this corpus is 95.8%, confirming that CLiMP represents robust contrasts in Chinese grammar. High performance on CLiMP thus implies high correlation with human acceptability judgments across these phenomena.

We use CLiMP to study Chinese BERT (Devlin et al., 2019),[2] 6 LSTM (Hochreiter and Schmidhuber, 1997) LMs, and 4 5-gram LMs. We evaluate for each MP whether the LM assigns a higher probability to the grammatical or the ungrammatical sentence. Our results show that Chinese BERT is closest to human performance, achieving an 81.8% accuracy on average over all phenomena, while the performances of LSTMs and 5-grams, regardless of the training data size, are only moderately above chance level. Classifier–noun agreement and verb complement selection are the phenomena that models generally perform best at, suggesting that Chinese LMs are better at acquiring knowledge of local selectional restrictions. The *bǎ* construction, binding, and filler-gap dependencies are the phenomena models have the most difficulties with. This indicates that they struggle to learn hierarchical syntax and to identify long-distance dependencies.

---

[1] Throughout this paper, we adopt a broad definition of LMs, which includes language representation models which have been trained on a masked language modeling objective.

[2] https://github.com/google-research/bert/blob/master/multilingual.md

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2784–2790
April 19 - 23, 2021.

2 Related Work

2.1 Language Models

LMs assign probabilities to sequences of words (Jurafsky and Martin, 2009). Recently, they have become commonly used as pretrained models, which can be fine-tuned for downstream NLP tasks (Peters et al., 2018; Devlin et al., 2019; Conneau et al., 2020). Strictly speaking, LMs compute the probabilities of words based only on past context. BERT (Devlin et al., 2019), however, is trained using a masked language modeling objective: it predicts words based on past and future tokens. Wang and Cho (2019) show that BERT is a Markov random field language model that can assign sentences a pseudo-log-likelihood score, which is computed by summing the conditional log probabilities of all tokens in the sentence, as well as generate text. Shin et al. (2019) and Salazar et al. (2020) apply pseudo-log-likelihood scores to sentence ranking and LM evaluation.

2.2 Evaluation of Linguistic Knowledge

Numerous methods exist for probing syntactic knowledge of neural network models in English (Hewitt and Manning, 2019; Tenney et al., 2019), and a growing body of work evaluates the syntactic knowledge of neural models by testing whether they can judge the grammatical acceptability of sentences. One common version of this task uses MPs to evaluate LMs' linguistic knowledge (Linzen et al., 2016; Marvin and Linzen, 2018; Warstadt et al., 2020; Wilcox et al., 2018).

A MP is a pair of sentences that only differ in acceptability due to a single edit, as in (1) and (2). Native speakers can be asked to choose which sentence in each pair sounds more grammatical. Semi-automatically generating MPs can yield a larger set of controlled sentences, providing sufficient data for model evaluation (Linzen et al., 2016; Marvin and Linzen, 2018; Ettinger et al., 2018).

(1) 王鑫 把 自行车 扔 了。
Wángxīn bǎ zìxíngchē rēng le
SUBJ. BA. OBJ. V. PST.
"Xin Wang threw away a bike."

(2) 王鑫 被 自行车 扔 了。
Wángxīn bèi zìxíngchē rēng le
SUBJ. PASS. OBJ. V. PST.
"Xin Wang was thrown away by a bike."

It is possible to model acceptability in a totally unsupervised way using LMs. The model assigns a probability to each sentence in a MP, and the one with the higher score is predicted as correct, and the model's predictions can be evaluated against human judgments (Marvin and Linzen, 2018; Warstadt et al., 2020). Supervised approaches are also possible (Warstadt et al., 2019), but can be less informative on LMs' linguistic knowledge acquisition due to the bias introduced by training on acceptability judgment labels.

Some prior work evaluates the linguistic knowledge of different non-English models (Ravfogel et al., 2018; Gulordava et al., 2018; Mueller et al., 2020). However, these efforts focus mainly on subject-verb agreement, which is absent in Chinese, and the knowledge of Chinese LMs has not yet been explicitly studied.

Finally, the linguistic abilities of English BERT have been investigated in a a lot of prior work, e.g., Clark et al. (2019); Vig (2019); Hewitt and Manning (2019). We refer the reader to Rogers et al. (2021) for an overview.

3 CLiMP

Our main contribution is CLiMP, a corpus of Chinese MPs designed to evaluate Chinese LMs. CLiMP consists of 1,000 MPs for each of 16 grammatical contrasts, covering 9 major Chinese linguistic phenomena. Example MPs for each phenomenon are shown in Table 1.

3.1 Data Generation

We generate data from grammar templates for every paradigm we incorporate. Our templates set lexical, syntactic, and semantic constraints for each paradigm, aiming at building robust contrasts and keeping the sentence length the same within each MP. We then build an annotated vocabulary, and generate sentences by sampling words from it. (1) and (2) show an MP together with the template[3] used to create it.

3.2 Vocabulary

We translate Warstadt et al.'s (2020) English vocabulary, containing 3,000 English words with morphological, syntactical, and semantic annotations. We add words and features specific to Chinese linguistic phenomena to our vocabulary, including classifiers, verb complements, action verbs, and

---

[3] The template example is only for demonstrative purposes. More information is encoded for the actual data generation.

coverbs. Our final vocabulary contains 3,456 words and 84 features.

We show the frequency of words in CLiMP's vocabulary in the Chinese Internet Corpus[4] in Figure 1. 1,055 of the words in CLiMP are within the 5,000 most frequent words in the Chinese Internet Corpus.

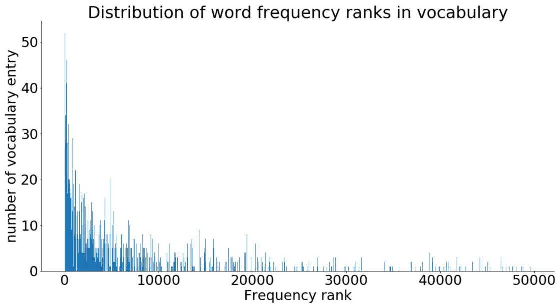


Figure 1: Comparison of word frequencies in CLiMP and the Chinese Internet Corpus.

3.3 Linguistic Phenomena

CLiMP covers 9 major linguistic phenomena in Mandarin Chinese, cf. Table 1. They are picked from a comprehensive Chinese grammar book by Po-Ching and Rimmington (2015). Following Po-Ching and Rimmington's discussion, we now explain the phenomena not present in English. The ***bǎ* construction** is an SOV construction involving the particle *bǎ*, which precedes the object and moves the object to a position before the main verb. It is only grammatical with a subset of transitive verbs. **Coverbs** are verb-like items that precede the main verb in a serial verb construction. They almost invariably have to be used in conjunction with other verbs in a sentence. They share some properties with prepositions, but are not syntactically interchangeable with them. **Classifiers** obligatorily appear with nouns when those are modified by numerals or adjectives. Mandarin has dozens of classifiers, and nouns select the classifier they combine with. **Verb complements** follow a verb, often expressing a result or manner of an event. Not all verbs can be used with all complements, making certain combinations ungrammatical. **NP head finality** is present in Mandarin noun phrases. The relative clause precedes noun phrases.

3.4 Data Validation

To verify whether the MPs in our dataset show clear contrasts, we conduct two rounds of human validation with 22 annotators. They are all native speakers of Chinese, 14 females and 8 males, whose ages range from 20 to 48. All of them have at least a high school degree.

In our first human validation, each human annotator is assigned a subset (100 MPs) of a paradigm. We let them perform the same forced-choice task as our models: decide for each MP which sentence seems more acceptable. We discard one paradigm, the coverb-direction paradigm, after this validation, because its human validation accuracy is below 85%. The average human agreement for the remaining paradigms is 95.8%.

In the second human validation, we sample 15 MPs from each of the remaining paradigm, resulting in a dataset consisting of 240 MPs. 16 annotators complete the same forced-choice task on this dataset. We count a MP as valid if more than half of the annotators agree with its label. The human agreement on this dataset is 97.1%, showing that our data creation results in valid examples.

3.5 Comparison with BLiMP

BLiMP consists of 67 datasets, each containing 1,000 MPs and organized by phenomenon into 12 categories. CLiMP only contains 16 datasets due to the less inflectional nature of Mandarin Chinese. 3 phenomena are covered by both corpora: anaphor agreement, binding, and filler-gap. The human agreement for these three phenomena in BLiMP is 97.5%, 87.3%, and 86.9%, respectively. The corresponding accuracies in CLiMP are 94.5%, 99%, and 100%, respectively. The overall human agreement for BLiMP is 88.6%, which is 7.2% lower than for CLiMP.

4 Models and Methods

We use accuracy for evaluation. A MP in CLiMP is classified correctly if a LM assigns a higher probability to the grammatical sentence than to the ungrammatical one. We evaluate statistical and neural LMs, including masked LMs. Corpora which contain 0.4M, 2M, and 21.5M sentences are used for further exploration. We also investigate the effect of different tokenizations.[5]

Chinese BERT BERT (Devlin et al., 2019) is a transformer-based neural model (Vaswani et al., 2017). Here, we evaluate Chinese BERT.[6] This

---

[4] http://corpus.leeds.ac.uk/frqc/internet-zh.num

[5] We use character tokenization and word tokenization (https://github.com/fxsjy/jieba).

[6] https://github.com/google-research/bert/blob/master/multilingual.md

Phenomenon	N	Acceptable Example	Unacceptable Example
Anaphor agreement	1	王玉珍 震惊-了 她自己。 Jane.F shock-PST herself. *'Jane shocked herself.'*	王玉珍 震惊-了 他自己。 Jane.F shock-PST himself. *'Jane shocked himself.'*
Binding	1	杨颖 治疗 吴宇涛 之后 佩服-过 她自己。 Yang.F cure Wu.M after admire-PST herself *'Yang admired herself after she cured Wu.'*	杨颖 治疗 吴宇涛 之后 佩服-过 他自己。 Yang.F cure Wu.M after admire-PST himself *'Yang admired himself after she cured Wu.'*
bǎ construction	1	王鑫 把 自行车 扔 了。 Wong.M BA bike throw PST *'Wong threw away the bike.'*	王鑫 被 自行车 扔 了。 Wong.M PASS bike throw PST *'Wong was thrown away by the bike.'*
Coverb	3	李文清 乘 卡车 到达-了 咖啡店。 Lee.M ride truck arrive-PST coffee shop *'Lee went to the coffee shop by truck.'*	李文清 于 卡车 到达-了 咖啡店。 Lee.M at truck arrive-PST coffee shop *'Lee went to the coffee shop at truck.'*
NP head finality	1	王梦 正在 卖 张红梅 清洗-过-的 推车。 Wong.F PROG sell May.F clean-PRF-ADJ trolley *'Wong is selling the trolley that Mel has cleaned.'*	王梦 正在 卖 推车 张红梅 清洗-过-的。 Wong.F PROG sell trolley May.F clean-PRF-ADJ *'Wong is selling the trolley that Mel has cleaned.'*
Classifier	2	张杰 正在 穿过 一 家 艺术画廊。 Jay.M PROG pass one CL:INSTITUTION art gallery *'Jay is passing through an art gallery.'*	张杰 正在 穿过 一 段 艺术画廊。 Jay.M PROG pass one CL:LENGTH art gallery *'Jay is passing through an art gallery.'*
Filler gap	1	图书馆, 我 开车 去-过 这个地方。 The library, I drive to-PRF this place *'The library, I have driven to this place.'*	图书馆, 我 开车 去-过 博物馆。 The library, I drive to-PRF the museum *'The library, I have driven to the museum.'*
Passive	1	这些 患者 被 转移-了。 These patient PASS transfer-PST *'These patients were transferred.'*	这些 患者 被 下降-了。 These patient PASS fall-PST *'These patients were fell.'*
Verb complement	5	王慧 的 文章 吓 坏 了 包曼玉。 Wong.F POSS article frighten badly PST Bao.F. *'Wong's article frightened Bao badly.'*	王慧 的 文章 吓 开 了 包曼玉。 Wong.F POSS article frighten openly PST Bao.F. *'Wong's article frightened Bao openly.'*

Table 1: Nine Chinese linguistic phenomena covered by CLiMP with acceptable and unacceptable sentence examples. Minimal differences are underlined. The second line of each example shows a gloss, the third line is an English translation. N represents how many paradigms (each with 1,000 examples) are within each phenomena.

model has 12 layers, 768 hidden units, 12 attention heads, and 110M parameters. The training dataset contains 25M sentences. We assign probabilities to sentences with this model by masking the words in a sentence one by one, computing the probability of each masked word, and, finally, multiplying the probabilities of all words (Wang and Cho, 2019; Salazar et al., 2020).[7]

LSTM LMs We further evaluate 6 LSTM (Hochreiter and Schmidhuber, 1997) LMs. These model have 2 layers, 200 hidden units, and 2 attention heads. We train them using Pytorch's word language model code[8] on 3 differently-sized Chinese Wikipedia corpora: 0.4M, 2M, and 21.5M sentences. We further compare word-level and character-level models (cf. Table 2). For evaluation, we employ code adapted by Warstadt et al. (2020) from Gulordava et al. (2018).[9]

***n*-gram LMs** Finally, we experiment with 4 different 5-gram LMs, which have been trained on 0.4M and 2M sentences from Chinese Wikipedia. For each corpus size, we train one word-based and one character-based LM. Those models are implemented using KenLM.[10]

5 Results

All results are shown in Table 2.

Phenomenon-specific Results Our LMs perform best on classifier–noun agreement and verb complement selection: Chinese BERT's accuracy is only 6.8% and, respectively, 3% lower than that of humans on these two phenomena. LSTMs and 5-grams remain around 30% behind humans, but still perform better on these phenomena than on others in CLiMP. This indicates that Chinese LMs acquire local selection knowledge better than the linguistic knowledge needed to master other phenomena.

Our LMs stuggle most with the *bǎ* construction, binding, and filler-gap dependencies. All models perform close to chance level for binding, suggesting that they lack the hierarchical knowledge necessary to correctly resolve the structural relationship between a reflexive and its binder. Similarly, most models perform near chance on filler-gap dependencies. This suggests that they do not robustly represent long-distance dependencies.[11]

---

[7] https://github.com/xu-song/bert-as-language-model

[8] https://github.com/pytorch/examples/tree/master/word_language_model

[9] https://github.com/sheng-fu/colorlessgreenRNNs

[10] https://kheafield.com/code/kenlm/

[11] A caveats applies: because Mandarin lacks *wh*-movement, we test filler-gap dependencies using a topicalization construc-

Model	Overall	Clsfr.	V.Cp.	Hd.Fi.	The ba.	Coverb	Ana.Agr.	Pass.	Bind.	Fi.Gap
Human	*95.8*	*99.7*	*96.0*	*100.0*	*85.0*	*92.5*	*94.5*	*91.0*	*99.0*	*100.0*
Chinese BERT	81.8	92.9	**93.0**	53.1	69.0	87.9	86.2	67.7	50.8	62.4
LSTM-21.5M-word	62.8	75.7	74.0	**81.4**	10.0	47.0	63.1	68.4	50.1	41.5
LSTM-21.5M-char	60.7	56.1	64.9	**89.1**	32.1	43.2	57.0	67.9	50.0	68.8
LSTM-2M-word	66.0	**77.8**	73.8	75.0	48.4	43.4	67.0	68.0	50.0	59.2
LSTM-2M-char	60.4	68.4	68.1	**86.3**	29.0	28.5	68.1	68.4	50.1	61.9
LSTM-2M-word	60.6	69.9	65.4	70.3	41.1	38.8	66.3	**72.7**	50.0	55.2
LSTM-2M-char	63.2	68.9	69.7	**83.9**	25.0	45.6	67.7	74.3	50.0	64.4
5-gram-2M-word	59.0	70.1	71	55.2	15.6	39.2	67.7	**72.0**	49.6	40.0
5-gram-2M-char	65.7	70.6	**78.8**	68.3	30.6	53.9	65.8	64.8	51.6	57.3
5-gram-0.4M-word	55.9	66.4	69.5	46.3	6.0	37.0	69.1	**77.8**	49.1	25.2
5-gram–0.4M-char	60.0	**71.5**	65.4	70.5	19.3	46.5	68.8	68.7	50.2	48.4

Table 2: Percentage accuracy of all humans and models on CLiMP. Random guessing yields an accuracy of 50%. Bold numbers indicate the phenomenon each model is best at. Numbers in model names (21.5, 2, 0.4) refer to the number of sentences in the training corpus.

On the head-final construction, Chinese BERT performs surprisingly poorly as compared to the other models: only 53.1% accuracy as compared to an average accuracy of 81% by the LSTMs. The coverb construction, in contrast, is easy for Chinese BERT: it achieves 87.9% accuracy, while the highest accuracy among all other models is 47%.

Model-specific Results Comparing across models, Chinese BERT achieves by far the highest overall accuracy with 81.8%. Our different LSTMs all perform worse, but obtain surprisingly similar scores: from 60.4% to 66.0%. The performances of our *5*-grams range from 55.9% to 65.7%. Keeping tokenization and corpus size constant, three out of four *5*-grams are outperformed by LSTMs. Thus, we overall find that neural models have advantages as compared to statistical models.

Comparing among the LSTMs, we find similarly to Hu et al. (2020) that the corpus size does not have much influence on the overall performance, with the caveat that these models perform close to chance. In contrast, a larger corpus size does result in a better performance in *5*-grams. We also compare the effect of different tokenizations: Character-based *5*-grams demonstrate better performance than word-based ones. For LSTMs, however, using characters only results in a better performance for our smallest corpus size (0.4M).

Compared to English LMs (Warstadt et al., 2020), the human–model gap is much bigger for Chinese models. While neither models nor datasets are directly comparable between our and previous work, this still suggests that more analyses and developments are needed for non-English models.

6 Conclusion

We introduced CLiMP, a suite of diagnostic test sets aimed at evaluating which syntactic phenomena Chinese LMs learn, and used it to evaluate 11 different models. All LMs appeared to have learned local selectional restrictions, but struggled with argument structure alternations, hierarchical structure, and long-distance dependencies. Chinese BERT performed best on CLiMP overall. However, it obtained a 14% lower accuracy than humans, suggesting there is still much room for improvement. We hope that CLiMP will serve as a linguistically informed resource for benchmarking and analyzing future progress on Chinese LMs. CLiMP is available at `https://nala-cub.github.io/resources`.

7 Acknowledgments

We would like to thank the students from CU Boulder's CSCI/LING5832 in Spring 2020 for their feedback on this research. We are also grateful for the feedback of the anonymous reviewers.

References

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In

---

tion more common in speech, and less likely to appear in the training corpora.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440–8451, Online. Association for Computational Linguistics.

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

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. 2018. Assessing composition in sentence vector representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1790–1801, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy. 2020. A systematic assessment of syntactic generalization in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1725–1744, Online. Association for Computational Linguistics.

Dan Jurafsky and James H. Martin. 2009. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Aaron Mueller, Garrett Nicolai, Panayiota Petrou-Zeniou, Natalia Talmina, and Tal Linzen. 2020. Cross-linguistic syntactic evaluation of word prediction models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5523–5539, Online. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Yip Po-Ching and Don Rimmington. 2015. *Chinese: A comprehensive grammar*. Routledge.

Shauli Ravfogel, Yoav Goldberg, and Francis Tyers. 2018. Can LSTM learn to capture agreement? the case of basque. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 98–107, Brussels, Belgium. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8(0):842–866.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Joonbo Shin, Yoonhyung Lee, and Kyomin Jung. 2019. Effective sentence scoring method using BERT for speech recognition. In *Asian Conference on Machine Learning*, pages 1081–1093.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8(0):377–392.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do rnn language models learn about filler–gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221.

Measuring and Improving Faithfulness of Attention in Neural Machine Translation

Pooya Moradi, Nishant Kambhatla, Anoop Sarkar
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
{pmoradi, nkambhat, anoop}@sfu.ca

Abstract

While the attention heatmaps produced by neural machine translation (NMT) models seem insightful, there is little evidence that they reflect a model's true internal reasoning. We provide a measure of faithfulness for NMT based on a variety of stress tests where attention weights which are crucial for prediction are perturbed and the model should alter its predictions if the learned weights are a faithful explanation of the predictions. We show that our proposed faithfulness measure for NMT models can be improved using a novel differentiable objective that rewards faithful behaviour by the model through probability divergence. Our experimental results on multiple language pairs show that our objective function is effective in increasing faithfulness and can lead to a useful analysis of NMT model behaviour and more trustworthy attention heatmaps. Our proposed objective improves faithfulness without reducing the translation quality and has a useful regularization effect on the NMT model and even improve translation quality in some cases.

1 Introduction

How trustworthy are our neural models? This question has led to a wide variety of contemporary NLP research focusing on (a) different axes of interpretability including *plausibility* (or interchangeably *human-interpretability*) (Herman, 2017; Lage et al., 2019) and *faithfulness* (Lipton, 2018; Jacovi and Goldberg, 2020b), (b) interpretation of the neural model components (Belinkov et al., 2017; Dalvi et al., 2017; Vig and Belinkov, 2019), (c) explaining the decisions made by neural models to humans (using explanations, highlights, rationales, etc.) (Ribeiro et al., 2016; Li et al., 2016; Ding et al., 2017; Ghaeini et al., 2018; Bastings et al., 2019; Jain et al., 2020), and (d) evaluating different explanation methods from different perspectives (Samek et al., 2016; Mohseni and Ragan, 2018; Poerner et al., 2018; Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019; Li et al., 2020).

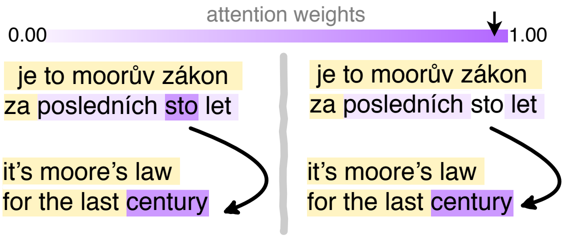


Figure 1: An example of *unfaithful* attention weights produced during a Cs-En translation. Note in the left attention heatmap, the attention is on the word sto while the decoder generates century. However, in the right heatmap, sto is not attended to at all but century is still produced as the output. This shows unfaithful behavior.

All of these approaches make NLP neural models more trustworthy. In this work, we focus on faithfulness which intuitively provides the extent to which an explanation accurately represents the true reasoning behind a prediction. It is particularly important for NLP practitioners who wish to debug their neural models and improve them. Faults of a neural model cannot be identified if the neural model does not provide a faithful and trustworthy description of what it is doing.

However, the formal definition of faithfulness and the proper approach to its evaluation are still contended in the literature. Jacovi and Goldberg (2020b) emphasize distinguishing faithfulness from human-interpretability in interpretability research by providing several clarifications about the terminology used by researchers. They describe the following conditions on the evaluation of how well a research project tackles the notion of faithfulness:

- Be explicit: provide a measurable evaluation of faithfulness.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2791–2802
April 19 - 23, 2021.

- Human judgements are not relevant because we are interested in model internals.
- Do not match against gold labels (e.g. AER) because faithfulness of both correct and incorrect decisions made by the model are equally important.
- No model is "inherently" faithful. We need to measure faithfulness not as a binary aspect of a model (it is faithful or not) but rather as a gray-scale measure.
- A more faithful system is a necessary but not sufficient condition for model interpretation by humans, c.f. Jacovi and Goldberg (2020a).

Aligned with these criteria, we study faithfulness of NLP neural models, specifically NMT models. We provide a faithfulness measure that is computed based on a variety of stress tests where attention weights that are crucial for prediction are perturbed. We expect from a faithful model to change its prediction under such tests (Figure 1). We quantify faithfulness based on how often the model outputs changed. The proposed metric is defined based on discrete changes in the output. It is not differentiable and cannot be simply included in the loss function of NMT to be optimized. We propose a novel differentiable objective based on probability divergence and study its effect on the discrete faithfulness measure. Our findings show that our objective is effective in increasing faithfulness and can lead to a useful analysis of NMT model behaviour and more trustworthy attention heatmaps. We assert that faithfulness is a good property to have in a model whether or not it will be useful for downstream interpretation. A model that is faithful can be trusted better as a component in a larger end-to-end neural model.

Contributions We seek to improve faithfulness of NMT models. To this end, we make the following contributions in this work:

- We propose a measure for quantifying faithfulness in NMT.
- We introduce a novel learning objective based on probability divergence that rewards faithful behavior and which can be included in the training objective for NMT.
- We provide empirical evidence that we can improve faithfulness in an NMT model. Our approach results in more a more faithful NMT model while producing better BLEU scores.[1]

We chose to study the impact of faithfulness in NMT because it is under-studied in terms of interpretability. Most previous work has focused on document or sentence-based classification tasks where attention models are not as directly useful as in NMT models. Attention is also more challenging in terms of faithfulness in the context of NMT models due to the substantial impact of the decoder component. While Transformers (Vaswani et al., 2017) generally produce better NMT models, they rely on multiple heads for attention. Defining an *overall* faithfulness measure in this case is challenging as different heads possibly have different faithfulness. Before addressing this more complicated problem, we first focus on the simpler single-head attention models. However, we expect larger and overparameterized models to get worse in terms of faithfulness because the language model in the decoder gets stronger in guessing the next word which, as we shall discuss in more detail later, tends to make attention less faithful.

2 Faithfulness in NMT Models

Intuitively, a faithful explanation should reflect the true internal reasoning of the model. Although there is no formal definition for faithfulness, a common approach in the community is to design stress tests to perturb the model parameters chosen in such a way that the model's decision should change if the model is faithful (Jacovi and Goldberg, 2020b). A common stress test is the *erasure* test in which the most-relevant part of the input is removed (Arras et al., 2017). In the context of NMT, at decoding time step t the attention component assigns attention weights α_t, attending to the source word at position $m_t = \text{argmax}_i \, \alpha_t[i]$ (or the k-best attended-to words in the source). These weights are often implicitly or explicitly regarded as an interpretation for the model's prediction at the time step t (Tu et al., 2016; Mi et al., 2016; Liu et al., 2016; Wang et al., 2016; Lee et al., 2017; Ding et al., 2017; Ghaeini et al., 2018). The erasure stress test for evaluating faithfulness offered by α_t is done by setting $\alpha_t[m_t]$ to zero and observing whether or not the output changes. It is worth noting that erasure is only one of the possible stress tests for evaluating faithfulness. Passing

---

[1] Our code is available at `https://github.com/sfu-natlang/attention_regularization`

more stress tests implies a more faithful model as it is properly reacting to more adversaries by changing its decision. In this paper we consider three intuitive stress test cases:

ZeroOutMax: (Arras et al., 2017): Here we remove attention from the most important token according to the attention weights by setting $\alpha_t[m_t] = 0$.

Uniform: (Moradi et al., 2019): In this stress test all attention weights are set to be equal, $\alpha_t = \frac{1}{m}\vec{1}$, where m is the length of the source sentence. This is to confuse the model about which part of the input is the most important one.

RandomPermute: (Jain and Wallace, 2019): In this stress test we randomly permute attention weights several times until a change in the model output is observed. We ensure that m_t, the most important token according to attention, is always changed. We set $\alpha'_t = \texttt{random\_permute}(\alpha_t)$ such that $\text{argmax}_i\, \alpha'_t[i] \neq m_t$

Many prior studies of attention (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019) have used a binary measure: either attention is faithful or it is not. These studies typically are about whether attention has the potential to be useful in terms of accuracy and faithful in terms of model behaviour. In many cases, especially in the case of NMT models, attention is clearly useful and by and large it must be faithful. The question is can we measure the faithfulness and improve faithfulness. It is more natural to have a gray-scale notion of faithfulness for evaluation (Jacovi and Goldberg, 2020b). Following this reasoning, we define $F(M)$ as faithfulness of attention heatmaps in model M as the following equation:

$$F(M) = \frac{\text{\# tokens passing stress tests}}{\text{\# tokens}} \tag{1}$$

$F(M)$ is a number between 0 to 1 measuring the percentage of output tokens during inference which passed the stress tests, i.e., they changed in the presence of adversarial attention. This metric can also be regarded as a measure of trust we can assign to the attention heatmap to fully reflect the internal reasoning of the NMT model.

3 Approach

The conventional objective function in a sequence-to-sequence task is a cross-entropy loss $\mathcal{F}_{acc}$:

$$\mathcal{F}_{acc}(\theta) = -\frac{1}{|S|} \sum_{(X,Y)\in S} \log p(Y|X;\theta) \tag{2}$$

where S is the training data and X and Y are source sentence and the correct translation respectively. This training objective does not explicitly model the interpretability aspects (e.g. faithfulness) of the network and it remains unoptimized during training.

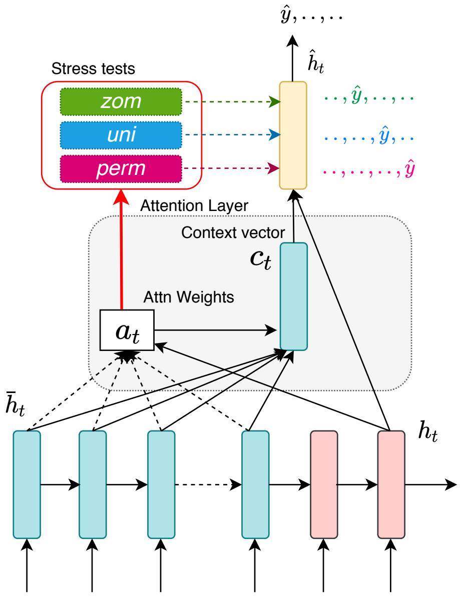


Figure 2: Using *ZeroOutMax, Uniform,* and *RandomPermute* stress tests, we generate adversaries to the attention weights. When adversarial attention weights are used, in a faithful model we expect the probability of the original output ($\hat{y}$) to drop significantly. We use this criterion to define a faithfulness objective function.

Faithfulness Objective In an effort to develop a model that is *right for right reason*, Ross et al. (2017) change the loss function of their classifier to model both *right answers* and *right reasons* instead of only the former. They achieve this by introducing a regularizing term that tends to shrink irrelevant gradients. In a similar spirit, we change our objective to account for the NMT model's faithfulness as well as the cross-entropy score against the reference translations:

$$\mathcal{F} = \mathcal{F}_{acc} + \lambda_{faith}\mathcal{F}_{faith} \tag{3}$$

$\mathcal{F}_{faith}$ is an additional component that rewards the model for having more faithful attention. The parameter λ_{faith} regulates the trade-off between faithfulness and accuracy objectives. Our proposed metric for faithfulness is calculated based on discrete changes in the output under adversarial attention. It is not differentiable and cannot be simply used as $\mathcal{F}_{faith}$ to be optimized. Thus, we propose a novel differentiable objective which mimics a faithful behavior hoping it improves the discrete faithfulness metric.

While here we intend to improve faithfulness using adversarial attention weights, it is important to note that making the *model* more *robust* is *not* our main goal. Robustness of a model is its resilience against adversarial input or small perturbations such as typos. Whether or not our approach results in a more robust model is a separate research question that we have not focused on.

3.1 Divergence-based Faithfulness Objective

Consider a predictive model g_θ in which an intermediate calculation is later employed to justify predictions:

$$\hat{y} = \arg\max_y p(y|x) = \arg\max_y g_\theta(x, IC(x), y) \tag{4}$$

where $IC(x)$ is the intermediate calculation on the input. A concrete example for $IC(x)$ would be the context vector calculated by the attention mechanism.

Hypothesis If there exists an intermediate calculation $IC'(x)$ that conveys a contradictory post-hoc attention compared to $IC(x)$, then $IC(x)$ cannot be regarded as faithful for predicting $\hat{y}$. If $IC(x)$ is faithful, we expect the model to *diverge* from predicting $\hat{y}$ when $IC'(x)$ is employed instead.

Based on our hypothesis, we propose a divergence-based objective which mimics behavior of a faithful explanation under stress test:

$$\mathcal{F}_{faith} = \log p(\hat{y}|x, IC'(x)) \tag{5}$$

This objective is a negative loss that should be minimized. The minimum of this objective is achieved when the probability of the original prediction approaches zero under the stress test which is the ideal. Thus, it promotes reduction in output probability under an adversarial intermediate calculation (Figure 2). It is worth noting that this objective can be potentially employed in models where outputs are modeled as soft probabilities and thus is not limited to NMT. To put model under various stress tests we manipulate the context vector during training time by changing the attention weights and feed it to the decoder to calculate the probability. More precisely:

$$\begin{aligned}\mathcal{F}_{faith} = \lambda_{zom} \log p(\hat{y}|x, IC'_{zom}(x)) \\ + \lambda_{uni} \log p(\hat{y}|x, IC'_{uni}(x)) \\ + \lambda_{perm} \log p(\hat{y}|x, IC'_{perm}(x))\end{aligned} \tag{6}$$

where IC'_{zom}, IC'_{uni} and IC'_{perm} are *ZeroOutMax*, *Uniform* and *RandomPermute* methods (see Sec. 2) to manipulate attention weights, respectively. $\lambda_{\{method\}}$ parameters regulate the contribution of each objective. We use the term $\mathcal{F}_{all}$ when all $\lambda_{\{method\}}$s in Eq. (6) are non-zero. Moreover, we use the term $\mathcal{F}_{\{method\}}$ when $\lambda_{\{method\}}$ is set to 1 and other regularization weights are zero.

3.2 On Attention Sparsity

Do the models trained with the faithfulness objective have sparser attention weights? Sharper attention in a model M might correlate with an intensified contribution of the most-attended source hidden state on the prediction resulting in higher faithfulness.

To measure sparseness of the attention, we take an average over the normalized entropy of attention distribution for each output token during inference on test data. We use normalized entropy which is in range [0,1] to account for the fact that the range of the entropy for each output token depends on the length of the corresponding source sentence.

$$AvgEnt = \frac{1}{\sum_{i=1}^{|S|} \left|\hat{Y}_i\right|} . \sum_{i=1}^{|S|} \sum_{j=1}^{\left|\hat{Y}_i\right|} NormEnt(\alpha_{ij}) \tag{7}$$

$$NormEnt(P) = -\sum_i \frac{P_i \log P_i}{\log N} \tag{8}$$

Here α_{ij} is the attention distribution for the output token j in the generated translation of source sentence i, and P is a discrete probability distribution. In Eq. (8) low entropy indicates a sharper distribution.

Attention Entropy Regularization Alongside investigating sparsity of the models trained by the faithfulness objective, we also train a model in which sparsity in attention is directly optimized. We used attention entropy regularization (Zhang et al., 2018):

$$\mathcal{F}_{ent} = \mathcal{F}_{acc} + \lambda_{ent} \sum_{i=1}^{|S|} \sum_{j=1}^{\left|\hat{Y}_i\right|} Ent(\alpha_{ij}) \tag{9}$$

where entropy of attention weights is added to the cross-entropy loss (2) as a regularization term.

4 Experimental Setup

4.1 Datasets

We use the Czech-English (Cs-En) dataset from IWSLT2016[2] and the German-English (De-En) dataset from IWSLT2014[3]. For the Czech-English dataset we use dev2010, tst2010, tst2011, tst2012, and tst2013 as the test data. For the German-English dataset we use dev2010, tst2010, tst2011, dev2012, and tst2012 as the test data. We used Moses (Koehn et al., 2007) to tokenize the dataset.

4.2 Architecture and Hyperparameters

We use OpenNMT (Klein et al., 2017) as our translation framework. We employ a 2 layer LSTM-based encoder-decoder (Sutskever et al., 2014; Cho et al., 2014) model with global attention (Luong et al., 2015). Dimension of the hidden states and the word embeddings for both source and target languages are set to 500. Vocabulary size for both the source and target language is set to 50000. We remove sentences with more than 50 tokens from the training data. We use Adam (Kingma and Ba, 2014) for training our models and we set the learning rate to 0.001. Models are trained until convergence. Our models have around 82M parameters. We optimize the hyperparameters of our models using the validation set. The baseline model is trained using Eqn. (2) and we call it $\mathcal{F}_{baseline}$. λ_{ent} in Eq. (9) is set to 0.04. We refer to the objective as $\mathcal{F}_{all}$ when λ_{zom}, λ_{uni}, and λ_{perm} are set to 0.5, 0.375, and 0.125 respectively. λ_{faith} is set to 1.

4.3 Training Difficulties

Our first attempts at using the modified objective function in Eq. (3) trained poorly. We observed that it was difficult for the model to learn the faithfulness constraint without having already learned to assign a reasonable probability to correct translations. To address this problem, we first train the NMT model using the standard unmodified objective function and then fine-tune this trained model by switching the objective function to Eq. (3).

One caveat is that the value of faithfulness loss can be arbitrarily large and interfere with the learning because cross-entropy error converge to infinity as the probability approaches to zero. However, we found that as long as the original output token received a small probability, there was no need for further reduction in it. Empirically, we clip losses more than 2.5 to 2.5 for each output token.

5 Results and Discussion

5.1 Impact on Faithfulness

To measure the effectiveness of the proposed objectives, we choose the best model in terms of provided faithfulness but within the 0.5 BLEU score of the maximum achieved BLEU score in the validation set. The reason is that we prefer a model that is both accurate and with faithful attention-based explanations. Table 1 shows the performance of the different faithfulness objective functions when generating content words and function words across different attention manipulation methods in the Czech-English (Cs-En) and German-English (De-En) datasets.

Results indicate that the proposed divergence-based objective has been effective in increasing the faithfulness metric. $\mathcal{F}_{all}$ is the most effective objective for increasing faithfulness when all stress tests are included in Eq. (1). When using $\mathcal{F}_{all}$, faithfulness of attention-based explanations for content words is increased 78% to 89%, while that of the function words is from 33% to 82%(see *All* column in Table 1). The same reductions are from 76% to 89% for content works and from 32% to 86% for function words in De-En dataset. These results establish the effectiveness of our proposed objectives to increase the faithfulness metric.

It is worth noting that increase in faithfulness of attention-based explanations for function words is much more than that of content words. This can be attributed to the fact the function words are mostly generated using the target-side information in the decoder (Tu et al., 2017; Moradi et al., 2019) and manipulating attention does not have much effect on generating them. However, our proposed faithfulness objective ($\mathcal{F}_{faith}$) seems to tighten the dependence of the decoder on the attention component. This results in much more increase in faithfulness for function words compared to such content words.[4] We also plot faithfulness over different

---

[2] https://sites.google.com/site/iwsltevaluation2016/

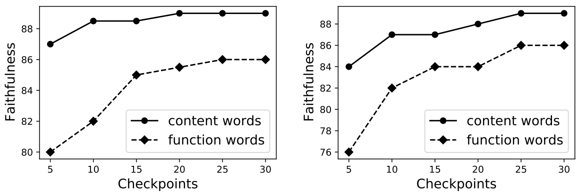
[3] https://sites.google.com/site/iwsltevaluation2014/

[4] If this dependence is not desired, it is possible not to penalize function words in the faithfulness objective. However, relying on attention for generating function words can be helpful, not necessarily for interpretability but for dealing with long-range dependencies (Vaswani et al., 2017) and, as a result, better translations.

	Objective	Content Words				Function Words			
		ZOM	*Uniform*	*RandPerm*	*All*	*ZOM*	*Uniform*	*RandPerm*	*All*
Cs-En	$\mathcal{F}_{baseline}$	83%	90%	94%	78%	46%	48%	64%	33%
	$\mathcal{F}_{zom}$	91%	93%	98%	86%	84%	87%	95%	74%
	$\mathcal{F}_{uni}$	84%	98%	97%	83%	56%	98%	91%	54%
	$\mathcal{F}_{perm}$	86%	95%	96%	83%	74%	97%	98%	71%
	$\mathcal{F}_{all}$	**91%**	**99%**	**98%**	**89%**	**83%**	**98%**	**98%**	**82%**
	$\mathcal{F}_{ent}$	78%	90%	94%	73%	46%	48%	64%	33%
De-En	$\mathcal{F}_{baseline}$	81%	90%	93%	76%	45%	48%	64%	32%
	$\mathcal{F}_{zom}$	91%	95%	98%	87%	87%	95%	97%	82%
	$\mathcal{F}_{uni}$	81%	98%	91%	80%	60%	100%	95%	58%
	$\mathcal{F}_{perm}$	85%	95%	97%	82%	74%	97%	98%	72%
	$\mathcal{F}_{all}$	**91%**	**98%**	**98%**	**89%**	**87%**	**100%**	**99%**	**86%**
	$\mathcal{F}_{ent}$	81%	90%	93%	76%	47%	47%	64%	33%

Table 1: Faithfulness metric for the generated content and function words through different objectives. Columns are different stress-tests included in the Eq.(1).

checkpoints in Figure 3. It indicates that progress in faithfulness is much faster for function words compared to content words.



(a) Cs-En

(b) De-En

Figure 3: Progress in faithfulness over different checkpoints. It increases much faster in function words compared to content words.

5.2 Effect of Training With Single Adversary on Passing Other Stress Tests

An interesting observation in Table 1 is that training with an adversary has positive effects on the model for passing stress tests from other types of adversaries. As an example, in Table 1 the column *Uniform* is the faithfulness metric when only Uniform test is employed in Eq. (1). When using this metric, we can observe that training a model with $\mathcal{F}_{perm}$ increased faithfulness from 90% to 95% for content words and from 48% to 97% for function words. We can see such effect for the German-English dataset as well. This observation indicates that training with each adversary can be beneficial for making model tolerant against other types of stress tests. It seems that training with each adversary strengthens the dependence of the decoder on the attention component which can be beneficial for passing other stress tests.

5.3 POS-tag Analysis

In addition to categorizing tokens into function and content words, we also analyze the effect of our proposed objective within different universal part-of-speech (POS) tags (Petrov et al., 2012) in Table 2. Our proposed objective has increased faithfulness in each POS tag and in our both datasets. Tokens with less lexical meaning are the ones affected the most as explained in Sec. 5.1. As expected, punctuations (PUNC) and particles (PRT) tags have benefited the most from increase in the faithfulness. Interestingly numbers (NUM tag) have the lowest increase in faithfulness. One reason might be that they already had a high initial faithfulness and this has made further increase less likely.

	De-En		**Cs-En**	
Tag	Baseline	Ours	Baseline	Ours
PUNC	0.19	**0.70**	0.28	**0.66**
PRON	0.42	**0.78**	0.35	**0.75**
VERB	0.47	**0.80**	0.50	**0.81**
ADP	0.30	**0.75**	0.40	**0.65**
DET	0.35	**0.74**	0.38	**0.70**
PRT	0.13	**0.63**	0.17	**0.50**
ADV	0.66	**0.80**	0.63	**0.79**
NOUN	0.63	**0.87**	0.64	**0.85**
ADJ	0.68	**0.87**	0.69	**0.85**
NUM	0.84	**0.86**	0.79	**0.86**
X	0.67	**0.78**	0.55	**0.80**

Table 2: Faithfulness metric within different part-of-speech (POS) tags.

5.4 Regularization Effect

The model checkpoints used in Tables 1 were selected based on maximum increase in faithfulness without sacrificing accuracy. To investigate if the proposed objective can have a general positive side effect in terms of accuracy, we train three independent models using the $\mathcal{F}_{baseline}$ and $\mathcal{F}_{all}$ objectives.

To make it fair for the baseline, we also add additional steps of training for the baseline model as well to isolate the benefit of adding the faithfulness objective. Table 3 contains the average BLEU score of the trained models. It indicates that the model trained with $\mathcal{F}_{all}$, has +0.7 and +0.4 increase in BLEU score compared to the baseline for the Czech-English and German-English language pairs respectively.

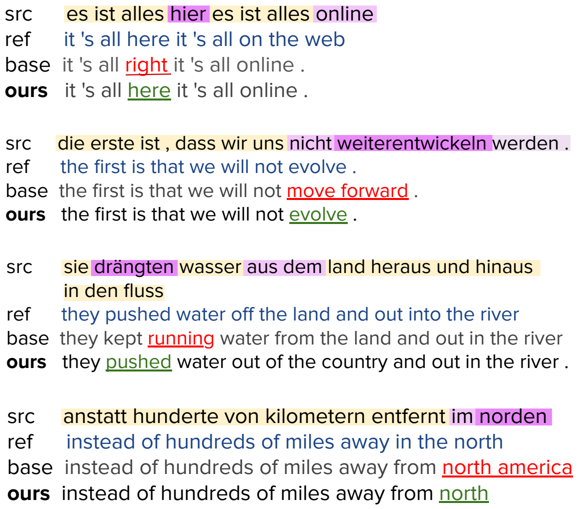
	Objective	BLEU%
Cs-En	$\mathcal{F}_{baseline}$	19.68
	$\mathcal{F}_{all}$	**20.4**
De-En	$\mathcal{F}_{baseline}$	24.85
	$\mathcal{F}_{all}$	**25.21**

Table 3: BLEU score of the baseline and the model trained with $\mathcal{F}_{all}$. Pairwise bootstrap resampling (Koehn, 2004) resulted in a p-value < 0.01 which indicates the statistical significance of the observed difference.

Improved BLEU scores for the faithful model can be due to two reasons: 1) the faithfulness objective can be seen as a regularization term which prevents the model from relying too much on the target-side context and the implicit language model in the decoder, which results in increased contribution of attention on the decoder and reducing some bias in the model. 2) penalizing the model for the lack of connection between justification and prediction forces the model to learn better translations by forcing it to justify each output in a *right answer for the right reason* paradigm. Figure 4 shows some examples of how our proposed model can produce better translations.

5.5 Do the New Models Have Sparser Attention?

Table 4 shows the average entropy and average normalized entropy for the baseline, the proposed model ($\mathcal{F}_{all}$), and the model trained with attention entropy regularization respectively. Evidently, the proposed model has not increased sparsity. On the other hand attention entropy regularization has been very effective in making attention weights sparser. But Table 1 indicates that attention entropy regularization has not been effective in increasing faithfulness. This suggests that sharper attention weights only affect the context vector and do not contribute to increased dependence of the decoder on attention.



src es ist alles hier es ist alles online
ref it 's all here it 's all on the web
base it 's all right it 's all online .
ours it 's all here it 's all online .

src die erste ist , dass wir uns nicht weiterentwickeln werden .
ref the first is that we will not evolve .
base the first is that we will not move forward .
ours the first is that we will not evolve .

src sie drängten wasser aus dem land heraus und hinaus in den fluss
ref they pushed water off the land and out into the river
base they kept running water from the land and out in the river
ours they pushed water out of the country and out in the river .

src anstatt hunderte von kilometern entfernt im norden
ref instead of hundreds of miles away in the north
base instead of hundreds of miles away from north america
ours instead of hundreds of miles away from north

Figure 4: These examples show some cases where the more faithful model trained using our faithfulness objective produces better translations compared to the baseline model. In each of these cases, perturbing the attention weights has no effect on the baseline model output. The faithful model is able to focus on the source side when needed in order to produce a more accurate translation.

	Model	*AvgEnt*	*AvgNormEnt*
Cs-En	$\mathcal{F}_{baseline}$	0.69	0.23
	$\mathcal{F}_{all}$	0.84	0.27
	$\mathcal{F}_{ent}$	**0.35**	**0.11**
De-En	$\mathcal{F}_{baseline}$	0.89	0.29
	$\mathcal{F}_{all}$	1.0	0.32
	$\mathcal{F}_{ent}$	**0.43**	**0.14**

Table 4: Average entropy and average normalized entropy of the baseline, the proposed model ($\mathcal{F}_{all}$), and the model trained with attention entropy regularization.

6 Related Work

Attention and Different Axes of Interpretability While several studies have focused on understanding the semantic notions captured by attention (Ghader and Monz, 2017; Vig and Belinkov, 2019; Clark et al., 2019), evaluating attention as an interpretability approach has garnered a lot of interest. From the faithfulness perspective, (Jain and Wallace, 2019; Serrano and Smith, 2019) show that for instances in a data set there can be adversarial attention heatmaps that do not change the output of the text classifier. In other words, adversarial attention leads to no decision flip in each instance. They use this to claim that attention heatmaps are not to be trusted, or unfaithful. Wiegreffe and Pinter (2019) argue against per-instance modifications at test time for two reasons: 1) in classification tasks attention may not be useful so perturbing attention is misleading. This is not true for NMT since attention is very useful in NMT. 2) they train an adversarial attention model (e.g. uniform attention) chosen to produce attention weights distant from the original attention weights while at the same time trying to minimize classification error. They show that such adversarial attention models are not as accurate as models with attention. In our work we acknowledge that attention is useful and faithful to some extent and we aim to improve faithfulness of NMT models.

While most of these works provide evidence that attention weights are not always faithful, Moradi et al. (2019) confirm similar observations on the unfaithful nature of attention in the context of NMT models. Li et al. (2020) is one of the few papers examining attention models in NMT. However, they are focused on the task of identifying relevant source words to explain the output translations selected by the NMT model. They look for optimal proxy models that agree with the NMT model such that the relevant source words picked as an explanation by a proxy model exhibits similar behaviour to the target model. They use the notion of *fidelity* over proxy models and evaluate several alternative proxy models using empirical risk minimization. Attention weights are evaluated alongside other proxy models for this task. In contrast, our work is about improving the faithfulness of NMT models and we focus on the internal state of the NMT model rather than proxy models. They use human references, e.g. AER, for evaluating fidelity. As discussed earlier, evaluation of faithfulness cannot involved human judgements or reference data. It is possible that our faithful NMT models are also better at fidelity, but that is an open question.

While prior works have mostly failed to explicitly distinguish faithfulness from plausibility in their arguments, Jacovi and Goldberg (2020a,b) focus on formalizing faithfulness and addressing evaluation of faithfulness separately from plausibility respectively. Subramanian et al. (2020) have investigated the concept of faithfulness in neural modular networks (NMN) which are employed for modeling compositionality. They question the faithfulness of the structure of the network modules describing the true abstract reasoning of the model. Similar to us, they attempt to quantify faithfulness and improve upon it. However their contributions like training with an auxilary atomic-task supervision for improved faithfulness are specific to the context of NMNs. Pruthi et al. (2020) demonstrate that it is possible to train a model that produces a deceptive attention mask, questioning the use of attention weights as explanation from the fairness and accountability perspective. Alvarez-Melis and Jaakkola (2018) investigate the interpretability methods from the robustness perspective. They attempt to quantify robustness and show that current interpretability methods cannot be considered as robust.

Sparsity For Improved Interpretability This line of work suggests making attention sparser so that the most contributing input word is more distinguishable over other input words. Martins and Astudillo (2016); Malaviya et al. (2018) propose sparse but differential alternatives to *softmax* function for calculating attention weights, while Zhang et al. (2018) propose sparsity regularization terms such as entropy regularization to promote sparsity in the attention.

Regularizing Explanations Ross et al. (2017) augment the loss function of their classification model with an explanation objective to constrain input gradient explanations. Rieger et al. (2019) follow a similar spirit but they use *contextual decomposition* (Murdoch et al., 2018) to extract explanations offered by the model. Aligning attention (as explanation) with prior knowledge has also been extensively studied. This prior knowledge can include alignment data (Mi et al., 2016; Liu et al., 2016), human rationales (Zhong et al., 2019), or even structural biases (Cohn et al., 2016).

Inherently Interpretable Neural Models Contrary to post-hoc explanation methods for interpreting a neural model, Stahlberg et al. (2018) show that the NMT model can be made self-explanatory by training it to produce the discrete decisions made by the model (from which the translations can be extracted later). In another work, (Lei et al., 2016; Bastings et al., 2019) propose models in which first a rationale is selected from the input and then is further used for prediction.

7 Conclusion

We proposed a method for quantifying faithfulness of NMT models. To optimize faithfulness we have defined a novel objective function that rewards faithful behavior through probability divergence. We also show that the additional constraint in the training objective for NMT does not harm translation quality and in some cases we see some better translations presumably due to the regularization effect of our faithfulness objective.

Future Work We aim to investigate and improve faithfulness of attention-based explanations in more sophisticated attention models such as Transformers (Vaswani et al., 2017). We can generalize our approach by designing explanatory modules in NMT through functionality separation (alignment, reordering, etc.) instead of relying only on attention. We also plan to investigate if faithful models can also be more useful for copy models and other applications of attention heatmaps in NMT.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. The research was partially supported by the Natural Sciences and Engineering Research Council of Canada grants NSERC RGPIN-2018-06437 and RGPAS-2018-522574 and a Department of National Defence (DND) and NSERC grant DGDND-2018-00025.

References

David Alvarez-Melis and Tommi S Jaakkola. 2018. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017. "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8).

Joost Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2017. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Taipei, Taiwan. Asian Federation of Natural Language Processing.

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

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California. Association for Computational Linguistics.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, and Stephan Vogel. 2017. Understanding and improving morphological learning in the neural machine translation decoder. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 142–151, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada. Association for Computational Linguistics.

Hamidreza Ghader and Christof Monz. 2017. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 30–39, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Reza Ghaeini, Xiaoli Fern, and Prasad Tadepalli. 2018. Interpreting recurrent and attention-based neural models: a case study on natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4952–4957, Brussels, Belgium. Association for Computational Linguistics.

Bernease Herman. 2017. The promise and peril of human evaluation for model interpretability. *arXiv preprint arXiv:1711.07414*, page 8.

Alon Jacovi and Yoav Goldberg. 2020a. Aligning faithful interpretations with their social attribution. *arXiv preprint arXiv:2006.01067*.

Alon Jacovi and Yoav Goldberg. 2020b. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C. Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Isaac Lage, Emily Chen, Jeffrey He, Menaka Narayanan, Been Kim, Sam Gershman, and Finale Doshi-Velez. 2019. An evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1902.00006*.

Jaesong Lee, Joong-Hwi Shin, and Jun-Seok Kim. 2017. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 121–126, Copenhagen, Denmark. Association for Computational Linguistics.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Jierui Li, Lemao Liu, Huayang Li, Guanlin Li, Guoping Huang, and Shuming Shi. 2020. Evaluating explanation methods for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 365–375, Online. Association for Computational Linguistics.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Zachary C Lipton. 2018. The mythos of model interpretability. *Queue*, 16(3):31–57.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102, Osaka, Japan. The COLING 2016 Organizing Committee.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Chaitanya Malaviya, Pedro Ferreira, and André F. T. Martins. 2018. Sparse and constrained attention for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 370–376, Melbourne, Australia. Association for Computational Linguistics.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623.

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2283–2288, Austin, Texas. Association for Computational Linguistics.

Sina Mohseni and Eric D Ragan. 2018. A human-grounded evaluation benchmark for local explanations of machine learning. *arXiv preprint arXiv:1801.05075*.

Pooya Moradi, Nishant Kambhatla, and Anoop Sarkar. 2019. Interrogating the explanatory power of attention in neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 221–230, Hong Kong. Association for Computational Linguistics.

W. James Murdoch, Peter J. Liu, and Bin Yu. 2018. Beyond word importance: Contextual decomposition to extract interactions from LSTMs. In *International Conference on Learning Representations*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA).

Nina Poerner, Hinrich Schütze, and Benjamin Roth. 2018. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 340–350, Melbourne, Australia. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Laura Rieger, Chandan Singh, W James Murdoch, and Bin Yu. 2019. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. *arXiv preprint arXiv:1909.13584*.

Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2662–2670.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. 2016. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Felix Stahlberg, Danielle Saunders, and Bill Byrne. 2018. An operation sequence model for explainable neural machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 175–186, Brussels, Belgium. Association for Computational Linguistics.

Sanjay Subramanian, Ben Bogin, Nitish Gupta, Tomer Wolfson, Sameer Singh, Jonathan Berant, and Matt Gardner. 2020. Obtaining faithful interpretations from compositional neural networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5594–5608, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhaopeng Tu, Yang Liu, Zhengdong Lu, Xiaohua Liu, and Hang Li. 2017. Context gates for neural machine translation. *Transactions of the Association for Computational Linguistics*, 5:87–99.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Jiajun Zhang, Yang Zhao, Haoran Li, and Chengqing Zong. 2018. Attention with sparsity regularization for neural machine translation and summarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(3):507–518.

Ruiqi Zhong, Steven Shao, and Kathleen McKeown. 2019. Fine-grained sentiment analysis with faithful attention. *arXiv preprint arXiv:1908.06870*.

Progressively Pretrained Dense Corpus Index for Open-Domain Question Answering

Wenhan Xiong*, **Hong Wang***, **William Yang Wang**
University of California, Santa Barbara
`{xwhan, hongwang600, william}@cs.ucsb.edu`

Abstract

Commonly used information retrieval methods such as TF-IDF in open-domain question answering (QA) systems are insufficient to capture deep semantic matching that goes beyond lexical overlaps. Some recent studies consider the retrieval process as maximum inner product search (MIPS) using dense question and paragraph representations, achieving promising results on several information-seeking QA datasets. However, the pretraining of the dense vector representations is highly resource-demanding, *e.g.*, requires a very large batch size and lots of training steps. In this work, we propose a sample-efficient method to pretrain the paragraph encoder. First, instead of using heuristically created pseudo question-paragraph pairs for pretraining, we use an existing pretrained sequence-to-sequence model to build a strong question generator that creates high-quality pretraining data. Second, we propose a simple progressive pretraining algorithm to ensure the existence of effective negative samples in each batch. Across three open-domain QA datasets, our method consistently outperforms a strong dense retrieval baseline that uses 6 times more computation for training. On two of the datasets, our method achieves more than 4-point absolute improvement in terms of answer exact match.

1 Introduction

With the promise of making the vast amount of information buried in text easily accessible via user-friendly natural language queries, the area of open-domain QA has attracted lots of attention in recent years. Existing open-domain QA systems are typically made of two essential components (Chen et al., 2017). A *retrieval module* first retrieves a compact set of paragraphs from the whole corpus (such as Wikipedia) that includes millions of documents. Then a *reading module* is deployed to extract an answer span from the retrieved paragraphs.

Over the past few years, much of the progress in open-domain QA has been focusing on improving the reading module of the system, which only needs to process a small number of retrieved paragraphs. Specifically, improvements include stronger reading comprehension models (Wang et al., 2018b; Yang et al., 2019; Xiong et al., 2020; Min et al., 2019a) and paragraph reranking models (Wang et al., 2018a; Lin et al., 2018) that assign more accurate relevance scores to the retrieved paragraphs. However, the performance is still bounded by the retrieval modules, which simply rely on traditional IR methods such as TF-IDF or BM25 (Robertson and Zaragoza, 2009). These methods retrieve text solely based on n-gram lexical overlap and can fail on cases when deep semantic matching is required and when there are no common lexicons between the question and the target paragraph.

While neural models have proven effective at learning deep semantic matching between text pairs (Bowman et al., 2015; Parikh et al., 2016; Chen et al., 2018; Devlin et al., 2019), they usually require computing question-dependent paragraph encodings (*i.e.*, the same paragraph will have different representations when considering different questions), which is formidable considering space constraints and retrieval efficiency in practice. More recent studies (Lee et al., 2019; Chang et al., 2020; Guu et al., 2020) show that such a dilemma can be resolved with large-scale matching-oriented pretraining. These approaches use separate encoders for questions and paragraphs and simply model the matching between the question and paragraph using inner products of the output vectors. Thus, these systems only need to encode all paragraphs in a question-agnostic fashion, and the resulted dense corpus index could be fixed and reused

---

[*] Equal Contribution.

[-] Our code is available at `https://github.com/xwhan/ProQA.git`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2803–2815
April 19 - 23, 2021.

for all possible questions. While achieving significant improvements over the BM25 baseline across a set of information-seeking QA datasets, existing pretraining strategies are highly sample-inefficient and typically require a large batch size (up to thousands), such that diverse and effective negative question-paragraph pairs could be included in each batch. When using a small batch size in our experiments, the model ceases to improve after certain updates. Given that a 12G GPU can only store around 10 samples with the BERT-base architecture at training time, the wider usage of these methods to corpora with different domains (*e.g.*, non-encyclopedic web documents or scientific publications) is hindered given modest GPU hardware.

In this work, we propose a simple and sample-efficient method for pretraining dense corpus representations. We achieve stronger open-domain QA performance compared to an existing method (Lee et al., 2019) that requires 6 times more computation at training time. Besides, our method uses a much smaller batch size and can be implemented with only a small number of GPUs, *i.e.*, we use at most 4 TITAN RTX GPUs for all our experiments. In a nutshell, the proposed method first uses a pretrained sequence-to-sequence model to generate high-quality pretraining data instead of relying on heuristics to create pseudo question-paragraph pairs; for the training algorithm, we use clustering techniques to get effective negative samples for each pair and progressively update the clusters. Our method's efficacy is further validated through ablation studies, where we replicate existing methods that use the same amount of resources. For the downstream QA experiments, we carefully investigate different finetuning objectives and show the different configurations of the retrieval and span prediction losses have nontrivial effects on the final performance. We hope this analysis could save the efforts on trying out various finetuning strategies of future research that focus on improving the retrieval component of open-domain QA systems.

The main contributions of this work include:

- We show the possibility of pretraining an effective dense corpus index for open-domain QA with modest computation resources.

- Our data generation strategy demonstrates that pretrained language models are not only useful as plug-and-play contextual feature extractors: they could also be used as high-quality data generators for other pretraining tasks.

- We propose a clustering-based progressive training paradigm that improves the sample-efficiency of dense retrieval pretraining and can be easily incorporated into existing methods.

2 Framework

We begin by introducing the network architectures used in our retrieval and reading comprehension models. Next, we present how to generate high-quality question-paragraph pairs for pretraining and how we progressively train the retrieval model with effective negative instances. Finally, we show how to finetune the whole system for QA.

2.1 Model Architectures

Notations We introduce the following notations which will be used through our paper. The goal of open-domain QA is to find the answer derivation (p, s) from a large text corpus $\mathcal{C}$ given a question q, where p is an evidence paragraph and s is a text span within p. The start and end token of s are denoted as $\mathrm{START}(s)$ and $\mathrm{END}(s)$ respectively. We refer the retrieval module as $\mathrm{P}_\theta(p|q)$, with learnable parameters θ. Similarly, we refer the reading comprehension module as $\mathrm{P}_\phi(s|p,q)$, which can be decomposed as $\mathrm{P}_\phi(\mathrm{START}(s)|p,q) \times \mathrm{P}_\phi(\mathrm{END}(s)|p,q)$. We use D^k to represent the top-k paragraphs from the retrieval module; a subset of $D^* \in D^k$ represents the paragraphs in D^k that cover the correct answer; for each paragraph $p \in D^*$, we define S_p^* as all the spans in p that match the answer string.

The Retrieval Module We uses two isomorphic encoders to encode the questions and paragraphs, and the inner product of the output vectors is used as the matching score. The encoders are based on the BERT-base architecture. We add linear layers $\mathbf{W}_q \in \mathbb{R}^{768\times 128}$ and $\mathbf{W}_p \in \mathbb{R}^{768\times 128}$ above the final representations of the [CLS] token to derive the question and paragraph representations:

$$\begin{aligned} h_q &= \mathbf{W}_q \mathrm{BERT}_Q(q)([\mathrm{CLS}]) \\ h_p &= \mathbf{W}_p \mathrm{BERT}_P(p)([\mathrm{CLS}]), \end{aligned}$$

The matching score is modeled as $h_q^\top h_p$. Thus, the probability of selecting p given q is calculated as:

$$\mathrm{P}_\theta(p|q) = \frac{e^{h_q^\top h_p}}{\sum_{p' \in \mathcal{C}} e^{h_q^\top h_{p'}}}.$$

In practice, we only consider the top-k retrieved paragraphs $\mathcal{C}$ for normalization.

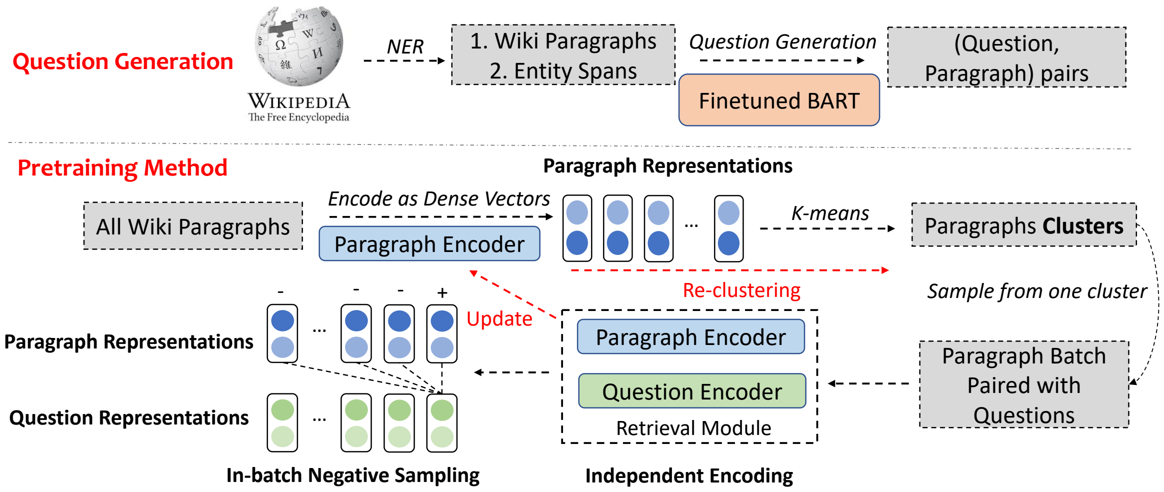


Figure 1: An overview of the progressive pretraining approach.

The Reading Module The architecture of our reading comprehension model is identical to the one in the original BERT paper (Devlin et al., 2019). We use two independent linear layers to predict the start and end position of the answer span. At training time, when calculating the span probabilities, we apply the shared-normalization technique proposed by Clark and Gardner (2018), which normalizes the probability across all the top-k retrieved paragraphs. This encourages the model to produce globally comparable answer scores. We denote this probability as $\mathrm{P}_{\phi}^{sn}(s|p,q)$ in contrast to the original formulation $\mathrm{P}_{\phi}(s|p,q)$ that normalizes the probability within each paragraph.

2.2 The Pretrainining Method

We now describe how to pretrain the retrieval module using a better data generation strategy and a progressive training paradigm. Figure 1 depicts the whole pretraining process.

Pretraining Data Generation Previous dense retrieval approaches usually rely on simple heuristics to generate synthetic matching pairs for pretraining, which do not necessarily reflect the underlying matching pattern between questions and paragraphs. To minimize the gap between pretraining and the end task, we learn to generate high-quality questions from the paragraphs using a state-of-the-art pretrained seq2seq model, *i.e.*, BART (Lewis et al., 2019). More specifically, we finetune BART on the original NaturalQuestions dataset (Kwiatkowski et al., 2019) such that it learns to generate questions given the groundtruth answer string and the groundtruth paragraph (labeled as *long answer* in NaturalQuestions). We concatenate the paragraph and the answer string with a separating token as the input to the BART model. We find this simple input scheme is effective enough to generate high-quality questions, achieving a 55.6 ROGUE-L score on the dev set. Samples of the generated questions could be found in the appendix. Afterward, we use spaCy[1] to recognize potential answer spans (named entities or dates) in all paragraphs in the corpus and use the finetuned BART model to generate the questions conditioned on the paragraph and each of the potential answers.

It is worth noting that the groundtruth answer paragraph supervision at this step could be eventually dropped and we could just use weakly supervised paragraphs to train the question generator; thus, our system becomes fully weak-supervised. As the pretraining process takes a long training time and lots of resources, we are unable to repeat the whole pretraining process with the weakly-supervised question generator. However, additional question generation experiments suggest that while using weakly-supervised paragraphs, the question generator still generates high-quality questions, achieving a ROUGE-L score of 49.6.[2]

In-batch Negative Sampling To save computation and improve the sample efficiency of pretraining, we choose to use in-batch negative sampling (Logeswaran and Lee, 2018) instead of gathering negative paragraphs for each question to pre-

---

[1] `https://spacy.io`

[2] For reference, a state-of-the-art QG model (Ma et al., 2020) trained with strong supervision achieves 49.9 ROUGE-L on a similar QA dataset also collected from real-user queries.

train the retrieval module. Specifically, for each pair (q, p) within a batch B, the paragraphs paired with other questions are considered negative paragraphs for q. Thus, the pretraining objective for each generated question is to minimize the negative log-likelihood of selecting the correct p among all paragraphs in the batch:

$$\mathcal{L}_{pre} = -\log \mathrm{P}_\theta(p|q). \tag{1}$$

A graphic illustration of this strategy is shown in Figure 1. As the batch size is usually very small compared to the number of all the paragraphs in the corpus, the pretraining task is much easier compared to the final retrieval task at inference time. In the whole corpus, there are usually lots of similar paragraphs and these paragraphs could act as strong distractors for each other in terms of both paragraph ranking and answer extraction. The desired retrieval model should be able to learn fine-grained matching instead of just learning to distinguish obviously different paragraphs. However, since existing dense retrieval methods typically use uniform batch sampling, there could be many easy negative samples in each batch, and they can only provide weak learning signals. Thus, a large batch size is usually adopted to include sufficient effective negative samples. Unfortunately, this is generally not applicable without hundreds of GPUs.

The Progressive Training Paradigm To provide effective negative samples, we propose a progressive training algorithm, as shown in the lower part of Figure 1. The key idea is to leverage the retrieval model itself to find groups of similar paragraphs. At a certain training step, we use the paragraph encoder at that moment to encode the whole corpus and cluster all (q, p) pairs into many groups based on the similarity of their paragraph encodings. These groups are supposed to include similar paragraphs and potentially related questions. Then, we continue our pretraining by sampling each batch from one of the clusters. By doing this, we can provide challenging and effective negative paragraphs for each question, even with small batch size. Every time we recluster the whole corpus, the model will be encouraged to learn finer-grained matching between questions and paragraphs. Algorithm 1 provides a formal description of the entire process. Note that our training algorithm shares spirits with Curriculum Learning (Bengio et al., 2009) and Self-Paced Learning (Kumar et al., 2010; Jiang et al., 2015), in which the models are trained with harder instances as the training progresses. Instead of utilizing a predefined or dynamically generated order of all the instances according to their easiness, our algorithm makes use of a dynamic grouping of all the training instances and is specifically designed for the efficient in-batch negative sampling paradigm.

Algorithm 1 The Clustering-based Progressive Pretraining

```
Input:
a) all $(q, p)$ pairs from the question generation model;
b) the retrieval module $\mathrm{BERT}_Q$ and $\mathrm{BERT}_P$;
while not finished do
    Encode the whole corpus with $\mathrm{BERT}_P$;
    Clustering all paragraphs into $C$ clusters using the dense encodings;
    for updates = $1{:}K$ do
        Random sample a paragraph cluster;
        Sample $B$ paragraphs from the cluster;
        Fetch the corresponding questions;
        Calculate gradients wrt $\mathcal{L}_{pre}$;
        if updates % $U$ == 0 then
            Update $\mathrm{BERT}_Q$ and $\mathrm{BERT}_P$;
        end if
    end for
end while
```

2.3 QA Finetuning

Once pretrained, we use the paragraph encoder to encode the corpus into a large set of dense vectors. Following previous practice, we only finetune the question encoder and the reading module so that we can reuse the same dense index for different datasets. For every training question, we obtain the question representation h_q from the question encoder and retrieve the top-k paragraphs D^k on the fly using an existing maximum inner product search package. To train the reading module, we apply the shared-normalization trick and optimize the marginal probability of all matched answer spans in the top-k paragraphs:

$$\mathcal{L}_{reader} = -\log \sum_{p \in D^*} \sum_{s \in S_p^*} \mathrm{P}_\phi^{sn}(s|p, q). \tag{2}$$

In additional to the reader loss, we also incorporate the "early" loss used by Lee et al. (2019), which updates the question encoder using the top-5000 dense paragraph vectors. If we define D^*_{5000} as those paragraphs in the top-5000 that contain the correct answer, then the "early" loss is defined as:

$$\mathcal{L}_{early} = -\log \sum_{p \in D^*_{5000}} \mathrm{P}_\theta(p|q). \tag{3}$$

Thus our total finetuning loss is $\mathcal{L}_{early} + \mathcal{L}_{reader}$. Note this is different from the joint formulation

Method	**Dataset** NaturalQuestions-Open	 WebQuestions	 CuratedTREC
DrQA (Chen et al., 2017)	-	20.7	25.7
R^3 (Wang et al., 2018a)	-	17.1	28.4
DSQA (Lin et al., 2018)	-	25.6	29.1
HardEM (Min et al., 2019a)	28.1	-	-
PathRetriever (Asai et al., 2020)	32.6	-	-
WKLM (Xiong et al., 2020)	-	34.6	-
GraphRetriever (Min et al., 2019b)	34.5	36.4	-
ORQA (Lee et al., 2019)	33.3	36.4	30.1
ProQA(Ours)	**37.4**	**37.1**	**34.6**

Table 1: Open-domain QA results in terms of exact answer match (EM). The first part of the table shows results from methods that use the traditional IR component. Note that these methods retrieve more paragraphs (typically dozens) than dense retrieval methods listed in the second part of the table, which only finds answers from the top-5.

used by Lee et al. (2019) and Guu et al. (2020), which consider the paragraphs as latent variables when calculating $\mathrm{P}(s|q)$. We find the joint objective does not bring additional improvements, especially after we use shared normalization. More variants of the finetuning objectives will be discussed in §3.5. At inference time, we use a linear combination of the retrieval score and the answer span score to rank the answer candidates from the top-5 retrieved paragraphs. The linear combination weight is selected based on the validation performance on each tested dataset.

3 Experiments

3.1 Datasets

We center our studies on QA datasets that reflect real-world information-seeking scenarios. We consider **1) NaturalQuestions-Open** (Kwiatkowski et al., 2019; Lee et al., 2019), which includes around 10K real-user queries (79,168/8,757/3,610 for train/dev/test) from Google Search; **2) WebQuestions** (Berant et al., 2013), which is originally designed for knowledge base QA and includes 5,810 questions (3,417/361/2,032 for train/dev/test) generated by Google Suggest API; **3) CuratedTREC** (Baudis and Sedivý, 2015), which includes 2,180 real-user queries (1,353/133/694 for train/dev/test) from MSNSearch and AskJeeves logs. Compared to other datasets such as SQuAD (Rajpurkar et al., 2016) and TriviaQA (Joshi et al., 2017), questions in these datasets are created without the presence of ground-truth answers and the answer paragraphs, thus are less likely to have lexical overlap with the paragraph.

3.2 Essential Implementation Details

For pretraining, we use a batch size of 80 and accumulate the gradients every 8 batches. We use the Adam optimizer (Kingma and Ba, 2015) with learning rate 1e-5 and conduct 90K parameter updates. Following previous work (Lee et al., 2019), we use the 12-20-2018 snapshot of English Wikipedia as our open-domain QA corpus. When splitting the documents into chunks, we try to reuse the original paragraph boundaries and create a new chunk every time the length of the current one exceeds 256 tokens. Overall, we created 12,494,770 text chunks, which is on-par with the number (13M) reported in previous work. These chunks are also referred to as paragraphs in our work. For progressive training, we recluster all the chunks with k-means around every 20k updates using the paragraph encodings.

While finetuning the modules for QA, we fix the paragraph encoder in the retrieval module. For each question, we use the top-5 retrieved paragraphs for training and skip the question if the top-5 paragraphs fail to cover the answer. The MIPS-based retrieval is implemented with FAISS (Johnson et al., 2019). On NaturalQuestions-Open, we finetune for 4 epochs. To save the finetuning time on this large dataset, we only use a subset (2,000 out of 8,757) of the original development set for model selection. For WebQuestions and CuratedTREC (both of them are much smaller), we finetune for 10 epochs. The optimizer settings are consistent with the pretraining. Hyperparameters and further details can be found in the appendix.

3.3 QA Performance

Following existing studies, we use the exact match (EM) as the evaluation metric, which indicates the percentage of the evaluation samples for which the

Method	EM	model size	batch size	# updates
ORQA	33.3	330M	4096	100K
T5	36.6	11318M	-	-
REALM	**40.4**	330M	512	200K
ProQA	37.4	330M	80*8	90K

Table 2: Resource comparison with SOTA models. EM scores are measured on NaturalQuestions-Open. *batch size* and *updates* all refer to the dense index pretraining. Note that REALM uses ORQA to initialize its parameters and we only report the numbers after ORQA initialization. “80*8” indicates that we use a batch size of 80 and accumulate the gradients every 8 batches.

predicted span matches the groundtruth answers. In Table 1, we first show that our progressive method (denoted as **ProQA**) is superior to all of the open-domain QA systems (the upper part of the table) that use conventional IR methods, even though we only use the top-5 paragraphs to predict the answer while these methods use dozens of retrieved paragraphs. For the dense retrieval methods, we compare with ORQA (Lee et al., 2019), which is most relevant to our study but simply uses pseudo question-paragraph pairs for pretraining and also requires a larger batch size (4,096). We achieve much stronger performance than ORQA with much fewer updates and a limited number of GPUs. To the best of our knowledge, this is the first work showing that an effective dense corpus index can be obtained without using highly expensive computational resources. The reduced requirement of computation also makes our method easier to replicate for corpora in different domains.

In Table 2, we compare our method other more recently published QA systems in terms of both performance and computation cost. It is worth noting that although we need to use a BART model to generate training questions for Wikipedia documents, the inference cost of the question generator is still much lower than the training cost of our system and is not significant for comparing the overall computation cost: with the same GPU hardware, generating all the questions takes less than 1/6 of the training time. When using the *batch size* × *# of updates* to approximate the training FLOPs, we see that our system is at least 6 times more efficient than ORQA. Compared to the recent proposed T5 (Roberts et al., 2020) approach, which converts the QA problem into a sequence-to-sequence (decode answers after encoding questions) problem and relies on the large model capacity to answer questions without retrieving documents, our system achieves better performance and is also much faster at inference time, due to the much smaller model size. The state-of-the-art REALM model (Guu et al., 2020) uses a more complicated pretraining approach that requires asynchronously refreshing the corpus index at train time. As it relies on ORQA initialization and further pretraining updates, it is even more computational expensive at training time. Also, as our method directly improves the ORQA pretraining, our method could easily stack with the REALM pretraining approach.

Method	R@5	R@10	R@20
ProQA (90k)	**52.0**	**61.0**	**68.8**
ORQA* (90k)	20.4	29.0	37.2
ProQA (no clustering, 90k)	42.9	52.6	60.8
ProQA (no clustering; 70k)	43.8	53.5	61.3
ProQA (no clustering; 50k)	38.8	48.2	56.7

Table 3: Ablation studies on different pretraining strategies. The retrieval modules (Recall@k) are tested on WebQuestions. *Our reimplementation.

Concurrent to our work, Karpukhin et al. (2020) show that it is possible to use the groundtruth answer paragraphs in the original NaturalQuestions dataset to train a stronger dense retriever. However, they use a larger index dimension (768) while encoding paragraphs and also retrieve more paragraphs (20∼100) for answer extraction. As a larger index dimension naturally leads to better retrieval results (Luan et al., 2020) (despite sacrificing search efficiency) and using more paragraphs increases the recall of matched answer spans[3], this concurrent result is not directly comparable to ours, and we leave the combination effect of efficient pretraining and strong supervision (*i.e.*, using human-labeled paragraphs) to future work.

3.4 Ablation Studies

To validate the sample efficiency of our method, we replicate the inverse-cloze pretraining approach from ORQA using the same amount of resource as we used while training our model, *i.e.*, the same batch size and updates (90k). We also study the effect of the progressive training paradigm by pretraining the model with the same generated data but without the clustering-based sampling. We test the retrieval performance on the WebQuestions test

---

[3] While using more paragraphs, we achieve 40.6 EM (compared to 41.5 EM in (Karpukhin et al., 2020)) even with a much smaller index dimension. Also, we did not use the gold paragraphs (strong supervision) to train our reading module.

set before any finetuning. We use Recall@k as the evaluation metric, which measures how often the answer paragraphs appear in the top-k retrieval. The results are shown in Table 3. We can see that for the non-clustering version of our method, the improvements are diminishing as we reach certain training steps, while the progressive training algorithm brings around 8% improvements on different retrieval metrics. This suggests the importance of introducing more challenging negative examples in the batch when the batch size is limited. Comparing the no-clustering version of our method against ORQA, we see that using our data generation strategy results in much better retrieval performance (more than 22% improvements on all metrics).

3.5 Analysis on Finetuning Objectives

Noting that different finetuning configurations have been used in existing studies , we conduct additional experiments to investigate the efficacy of different finetuning objectives and provide insights for future research that intends to focus on improving the model itself. Specifically, we study the effects of using a joint objective (Lee et al., 2019; Guu et al., 2020) and adding an additional reranking objective that is commonly used in sparse-retrieval QA systems (Min et al., 2019a; Xiong et al., 2020).

The joint objective treats the retrieved paragraphs as latent variables and optimizes the marginal probability of the matched answer spans in all paragraphs:

$$\mathcal{L}_{joint} = -\log \sum_{p \in D^\star} \mathrm{P}_\theta(p|q) \sum_{s \in S_p^*} \mathrm{P}_\phi(s|p,q). \tag{4}$$

The reranking objective is usually implemented through a paragraph reranking module that uses a question-dependent paragraph encoder, *e.g.*, a BERT encoder that takes the concatenation of the question and paragraph as input. This kind of reranker has been shown to be beneficial to conventional IR methods since it can usually provide more accurate paragraph scores than the TF-IDF or BM25 based retriever while ranking the answer candidates. To implement this reranking module, we simply add another reranking scoring layer to our BERT-based span prediction module $\mathrm{P}_\phi(s|p,q)$, which encodes the paragraphs in a question-dependent fashion. At inference time, we use the paragraph scores predicted by this reranking component instead of the pretrained retrieval model to guide our final answer selection.

id	**Objective Settings** *joint*	 *rerank*	 *shared-norm*	EM
1	-	-	✓	38.5
2	✓	-	✓	38.3
3	-	✓	✓	38.2
4	✓	-	-	36.2
5	-	-	-	35.1

Table 4: Analysis on different finetuning objectives on NaturalQuestions-Open. EM scores are measured on the 2,000 validation samples we used for model selection.

Table 4 shows the results of different objective settings. Comparing the results of (4) and (5), we can see that the joint objective can bring some improvements when shared-normalization is not applied. However, it does not yield improvements when shared-normalization is applied, according to the results of (1) and (2). By comparing (1) and (3), we see that with the strong pretrained retrieval model, adding an extra reranking module that uses question-dependent paragraph encodings is no longer beneficial. This is partially because our pretrained retrieval model gets further improved during finetuning, in contrast to a fixed TF-IDF based retriever. Finally, from (1) and (5), we see that the shared normalization brings much larger improvements than the other factors. This aligns with the findings from an existing work (Wang et al., 2019) that only tested on SQuAD questions.

4 Error Analysis

To investigate the fundamental differences of the dense and sparse retrieval methods in open-domain QA, we conduct an error analysis using both the proposed method and a baseline system that uses TF-IDF and BM25 for retrieval. This baseline uses a similar retrieval pipeline as Min et al. (2019a) and is trained with the finetuning objective defined in Eq. 2. This sparse retrieval baseline achieves 29.7 EM on the official dev set of NaturalQuestions-Open while our method achieves 36.7 EM[4]. Figure 2 shows the Venn diagram of the error sets from both systems. Our key findings are summarized in the following paragraphs.

The difference retrieval paradigms could complement each other. First, according to the error set differences (shown by the white regions in Figure 2), a considerable portion of the error cases (9.1% of the devt set) of our dense-retrieval system

---

[4] Note that this number is different from the number in Table 4 as we did not use the whole dev set for model selection.

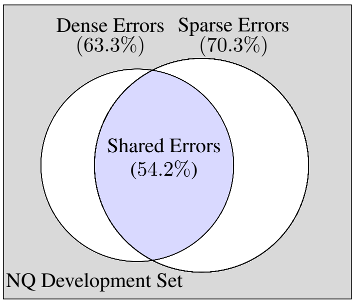


Figure 2: The error sets (from the official dev set of NaturalQuesitons-Open) of our dense retrieval method and a baseline system using sparse retrieval methods like TF-IDF. We use circles to represent the error sets of both systems. The percentages show the relative size of each set in terms of the whole dev set.

does not occur in the sparse-retrieval system and vice versa. This suggests the necessity of incorporating different retrieval paradigm when building real-world applications. In fact, the hybrid approach has already been adopted by a phrase-level retrieval method (Seo et al., 2019) and concurrent studies (Luan et al., 2020; Karpukhin et al., 2020).

Both systems are underestimated. As shown in Figure 2, 54.2% of the questions cannot be correctly answered by either system. However, our manual inspection on 50 of the shared error cases suggests that around 30% of these errors are due to annotation issues (14%) or the ambiguous nature of real-user queries (16%). One obvious annotation issue is the incompleteness of the answer labels. Example questions include *"When did Brazil lose to in 2014 World Cup?"*, to which both *"Germany"* and *"Netherlands"* are correct answers. This issue occurs because the annotators of NaturalQuestions only have a local view of the knowledge source as they are only asked to label the answer span using one document. In terms of the ambiguous questions, many of them are due to constraints unspecified by the question words, such as *"What is the population of New York City?"* (the time constraint is implicit) or *"When did Justice League come out in Canada?"* (needs entity disambiguation). This kind of questions result from the information-seeking nature of the open-domain QA task where the users usually use the minimal number of words for searching and they are not aware of the potential ambiguous factors. To solve this kind of questions, an interactive QA system might be necessary. In the appendix, we show more ambiguous questions in which other kinds of constraints are missing.

5 Related Work

The task of answering questions without specifying specific domains has been intensively studied since the earlier TREC QA competitions (Voorhees, 1999). Studies in the early stage (Kwok et al., 2001; Brill et al., 2002; Ferrucci et al., 2010; Baudiš, 2015) mostly rely on highly sophisticated pipelines and heterogeneous resources. Built on the recent advances in machine reading comprehension, Chen et al. (2017) show that open-domain QA can be simply formulated as a reading comprehension problem with the help of a standard IR component that provides candidate paragraphs for answer extraction. This two-stage formulation is simple yet effective to achieve competitive performance while using Wikipedia as the only knowledge resource.

Following this formulation, a couple of recent studies have proposed to improve the system using stronger reading comprehension models (Yang et al., 2019; Wang et al., 2018b), more effective learning objectives (Clark and Gardner, 2018; Min et al., 2019a; Wang et al., 2019) or paragraph reranking models (Wang et al., 2018a; Lin et al., 2018; Lee et al., 2018). However, the retrieval components in these systems are still based on traditional inverted index methods, which are efficient but might fail when the target paragraph does not have enough lexicon overlap with the question.

In contrast to the sparse term-based features used in TF-IDF or BM25, dense paragraph vectors learned by deep neural networks (Zhang et al., 2017; Conneau et al., 2017) can capture much richer semantics beyond the n-gram term features. To build effective paragraph encoders tailed for the paragraph retrieval in open-domain QA, more recent studies (Lee et al., 2019; Chang et al., 2020; Guu et al., 2020) propose to pretrain Transformer encoders (Vaswani et al., 2017) with objectives that simulate the semantic matching between questions and paragraphs. For instance, Lee et al. (2019) uses the inverse cloze pretraining task to train a bi-encoder model to match a sentence and the paragraph in which the sentence belongs to. These approaches demonstrate promising performance but require a lot of resources for pretraining. The focus of this paper is to reduce the computational requirements of building an effective corpus index.

6 Conclusion

We propose an efficient method for pretraining the dense corpus index which can replace the traditional IR methods in open-domain QA systems. The proposed approach is powered by a better data generation strategy and a simple yet effective data sampling protocol for pretraining. With careful finetuning, we achieve stronger QA performance than ORQA that uses much more computational resources. We hope our method could encourage more energy-efficient pretraining methods in this direction such that the dense retrieval paradigm could be more widely used in different domains.

Acknowledgement

The research was partly supported by the J.P. Morgan AI Research Award. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agency.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia.

Petr Baudiš. 2015. Yodaqa: a modular question answering system pipeline. In *19th International Student Conference on Electrical Engineering*, pages 1156–1165.

Petr Baudis and Jan Sedivý. 2015. Modeling of the question answering task in the yodaqa system. In *6th International Conference of the CLEF Association, CLEF 2015*, pages 222–228, Toulouse, France. Springer.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009*, pages 41–48, Montreal, Québec, Canada. ACM.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pages 1533–1544, Seattle, Washington, USA. ACL.

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

Eric Brill, Susan T. Dumais, and Michele Banko. 2002. An analysis of the askmsr question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, EMNLP 2002*, Philadelphia, Pennsylvania, USA.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. In *8th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing sentence embedding with generalized pooling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1815–1826, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

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

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building watson: An overview of the deepqa project. *AI Magazine*, 31(3):59–79.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G. Hauptmann. 2015. Self-paced curriculum learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2694–2700, Austin, Texas, USA. AAAI Press.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *CoRR*, abs/2004.04906.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*, San Diego, California, USA.

M. Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *24th Annual Conference on Neural Information Processing Systems, NIPS 2010*, pages 1189–1197, Vancouver, British Columbia, Canada. Curran Associates, Inc.

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

Cody C. T. Kwok, Oren Etzioni, and Daniel S. Weld. 2001. Scaling question answering to the web. In *Proceedings of the Tenth International World Wide Web Conference, WWW 2001*, pages 150–161, Hong Kong, China. ACM.

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 565–569, Brussels, Belgium. Association for Computational Linguistics.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1736–1745, Melbourne, Australia. Association for Computational Linguistics.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *6th International Conference on Learning Representations, ICLR 2018*, Vancouver, British Columbia, Canada.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, dense, and attentional representations for text retrieval. *CoRR*, abs/2005.00181.

Xiyao Ma, Qile Zhu, Yanlin Zhou, Xiaolin Li, and Dapeng Wu. 2020. Improving question generation with sentence-level semantic matching and answer position inferring. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, New York, New York, USA. AAAI Press.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *CoRR*, abs/1911.03868.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

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

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *CoRR*, abs/2002.08910.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Min Joon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur P. Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Thirty-first Annual Conference on Neural Information Processing Systems, NIPS 2027*, pages 5998–6008, Long Beach, California, USA.

Ellen M. Voorhees. 1999. The TREC-8 question answering track report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999*, Gaithersburg, Maryland, USA. National Institute of Standards and Technology.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018a. R^3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 5981–5988, New Orleans, Louisiana, USA. AAAI Press.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018b. Evidence aggregation for answer re-ranking in open-domain question answering. In *6th International Conference on Learning Representations, ICLR 2018*, Vancouver, British Columbia, Canada.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 5877–5881, Hong Kong, China. Association for Computational Linguistics.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. 2017. Deconvolutional paragraph representation learning. In *Thirty-first Annual Conference on Neural Information Processing Systems, NIPS 2017*, pages 4169–4179, Long Beach, California, USA.

A Appendix

A.1 Further Implementation Details

The pretraining process takes 4 TITAN RTX GPUs, each with a 24G memory. We use the NVIDIA Apex package for mixed-precision training. 90K parameter updates take around 7 days to finish. For QA experiments, We use the **IndexIVFFlat** index for efficient search. We assign all the vectors to 100 Voronoi cells and only search from the closest 20 cells. The random seed is set as 3 for all QA datasets. We use a batch size of 8 (8 questions, each of them are paired with 5 paragraphs) for NaturalQuestions-Open and 1 for the other datasets. We limit the maximum answer length to 10 subword tokens. For NaturalQuestions-Open, we evaluate the model every 1000 updates and save the best checkpoint based on validation EM. For WebQuesitons and CuratedTREC, we evaluate the model after every epoch. As neither of these two small datasets has an official dev set, we use a small split to find the best hyperparameters and then retrain the model with all the training questions. To accelerate training, especially for the early loss function which requires annotate the top5000 retrieved paragraphs, we pre-annotate the top10000 paragraphs retrieved by the untuned retrieval module and build an answer paragraph set for each question. At finetuning time, we direct check whether a particular paragraph is in the precomputed paragraph set, instead of doing string matching for each of the 5000 paragraphs. Our BERT implementations are based on huggingface Transformers[5].

A.2 Qualitative Examples

Here we include more examples that complement the results and analysis of the paper. Table 5 shows the generated questions from the finetuned BART model and Table 6 complements the error analysis.

---

[5] `https://github.com/huggingface/transformers`

Gold Paragraph: “Does He Love You” is a song written by Sandy Knox and Billy Stritch, and recorded as a duet by American country music artists Reba McEntire and Linda Davis. It was released in August 1993 as the first single from Reba’s album Greatest Hits Volume Two. It is one of country music ’s several songs about a love triangle.
Original Question: Who sings does he love me with reba?
Generated Question: Who sings with reba mcentire on does he love you?

Gold Paragraph: Invisible Man First edition Author Ralph Ellison Country United States Language English Genre Bildungsroman African-American literature social commentary Publisher Random House Publication date 1952 Media type Print (hardcover and paperback) Pages 581 (second edition) ...
Original Question: How many pages is invisible man by ralph ellison?
Generated Question: How many pages in the invisible man by ralph ellison?

Gold Paragraph: The Great Lakes (French: les Grands-Lacs), also called the Laurentian Great Lakes and the Great Lakes of North America, are a series of interconnected freshwater lakes located primarily in the upper mid-east region of North America, on the Canada–United States border, which connect to the Atlantic Ocean through the Saint Lawrence River. They consist of Lakes Superior, Michigan, Huron (or Michigan–Huron), Erie, and Ontario.
Original Question: Where do the great lakes meet the ocean?
Generated Question: Where do the great lakes of north america meet the atlantic?

Gold Paragraph: My Hero Academia: Two Heroes , Hepburn:Boku no Hiro Academia THE MOVIE: Futari no Hiro) is a 2018 Japanese anime superhero film based on the manga My Hero Academia by Kohei Horikoshi. Set between the second and third seasons of the anime series, the film was directed by Kenji Nagasaki and produced by Bones. Anime Expo hosted the film’s world premiere on July 5, 2018, and it was later released to theaters in Japan on August 3, 2018.
Original Question: When does the new my hero academia movie come out?
Generated Question: When does the my hero academia two heroes movie come out?

Gold Paragraph: Victoria’s Secret Store, 722 Lexington Ave, New York, NY Type Subsidiary Industry Apparel Founded June 12, 1977; 40 years ago (1977-06-12) Stanford Shopping Center, Palo Alto, California, U.S. Founder Roy Raymond Headquarters Three Limited Parkway, Columbus , Ohio , U.S. Number of locations 1,017 company - owned stores 18 independently owned stores Area served ...
Original Question: Who was the creator of victoria’s secret?
Generated Question: Who is the founder of victoria’s secret and when was it founded?

Table 5: Samples of the generated questions. The answer spans are underlined. Here we show the generated questions for samples at the beginning of the official NaturalQuestions-Open dev data. We only skip the samples whose gold paragraphs are not natural paragraphs (*e.g.*, incomplete sentences).

Question: What is a ford mondeo in the usa?
Annotated Answers: ford contour, mercury mystique, ford fusion
ambiguous; could be asking about a particular car type (mid-sized car) instead of brand series

Question: air flow in the eye of a hurricane?
Annotated Answers: no wind
ambiguous; question itself is hard to understand

Question: Who wrote I’ll be there for you?
Annotated Answers: Michael Skloff, Marta Kauffman, Allee Willis, David Crane, Phil Solem, Danny Wilde, The Rembrandts
ambiguous; there are multiple songs having this name

Question: Where do you go for phase 1 training?
Annotated Answers: army foundation college
ambiguous; the meaning of phase 1 is vague, could have different meanings in different context

Question: When does the new spiderman series come out?
Annotated Answers: August 19 , 2017
ambiguous; time constraint missing

Question: Where did the super bowl take place this year?
Annotated Answers: minneapolis, minnesota
ambiguous; the year cannot be inferred from the question words alone

Table 6: Error cases that include ambiguous questions.

Exploring the Limits of Few-Shot Link Prediction in Knowledge Graphs

Dora Jambor[1,2], Komal Teru[1,2], Joelle Pineau[1,2,3], William L. Hamilton[1,2]
[1] School of Computer Science, McGill University, Canada
[2] Quebec AI Institute (Mila), Canada
[3] Facebook AI Research (FAIR)
`{dora.jambor, komal.teru}@mail.mcgill.ca`
`{jpineau, wlh}@cs.mcgill.ca`

Abstract

Real-world knowledge graphs are often characterized by low-frequency relations—a challenge that has prompted an increasing interest in few-shot link prediction methods. These methods perform link prediction for a set of new relations, unseen during training, given only a few example facts of each relation at test time. In this work, we perform a systematic study on a spectrum of models derived by generalizing the current state of the art for few-shot link prediction, with the goal of probing the limits of learning in this few-shot setting. We find that a simple *zero-shot* baseline—which ignores any relation-specific information—achieves surprisingly strong performance. Moreover, experiments on carefully crafted synthetic datasets show that having only a few examples of a relation fundamentally limits models from using fine-grained structural information and only allows for exploiting the coarse-grained positional information of entities. Together, our findings challenge the implicit assumptions and inductive biases of prior work and highlight new directions for research in this area.

1 Introduction

A knowledge graph (KG) is a multi-relational graph that offers a structured way to organize facts about the world. Encoder-decoder approaches are commonly used to predict new facts from existing ones where entities and relations are embedded in a low-dimensional vector space via an encoder, to then score the likelihood of observing a new fact via a decoder (Nickel et al., 2015; Bordes et al., 2013; Trouillon et al., 2017; Dettmers et al., 2018).

It is well known that the performance of these methods can significantly drop when predicting for relations that are only observed in a few example facts. However, link prediction for these low-frequency relations is very important, as not only are these relations abundant in most knowledge graphs, they are also key for knowledge graph completion tasks where new relations may appear after model training.

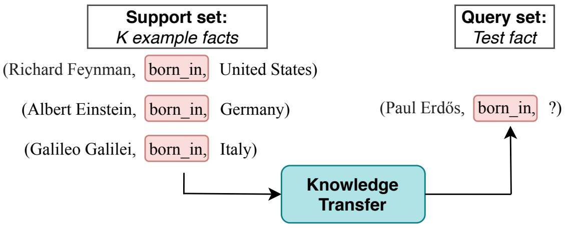


Figure 1: The few-shot link prediction task

To study this low-frequency regime, Xiong et al. (2018) created the Nell-One and Wiki-One benchmarks where the task is to predict new facts for a set of new relations at test time, where each relation is only observed a *few* times (as specified by some small fixed number K). Previous approaches have shown promising results using metric-based (Xiong et al., 2018) and gradient based meta-learning techniques (Chen et al., 2019). However, we argue that these models are limited by the current task formulation to only exploit coarse-grained positional signals (i.e., nodes belonging to the same community) that are abundant in these benchmarks, rather than leveraging structural signals (e.g. transitivity, symmetry).

Present work. In this work, we take a critical take on current approaches for few-shot link prediction over knowledge graphs. We posit that current meta-learning based approaches benefit largely due to the positional signals in entities, rather than utilising information about the low frequency relations. We corroborate these insights by conducting a systematic study on a spectrum of models with decreasing complexity. Interestingly, we find that a much simpler zero-shot variant of the state of the art —devoid of any meta-learning scheme— yields surprisingly competitive results, while not consuming any example facts about a relation. Motivated by these observations, we design a set of null models tailored to different learning signals a model might utilize to drive effective link predic-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2816–2822
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

tion. Empirically, we validate that these existing meta-learning models are ill-equipped to infer logical patterns about the few-shot relations. These findings bring forth the shortcomings of the current task formulation and raises new questions in both task and model design while highlighting new directions of research in few-shot link prediction.

2 Few-shot link prediction

2.1 Problem definition

The goal of few-shot link prediction is to predict missing links for a new relation by only observing K example triples of that relation (Figure 1). Following literature in few-shot classification (Ravi and Larochelle, 2016; Vinyals et al., 2016; Snell et al., 2017), we organize our dataset as a set of tasks, where a task corresponds to predicting links for a new relation. The set of tasks for training and testing are disjoint, with the added constraint that entities in the test tasks are a subset of the entities in the train tasks. Let $\mathcal{V}$ denote the set of entities in the knowledge graph. For each new relation r_i, we then construct a support set $\mathcal{S}_i = \{(h_k, r_i, t_k)\}_{k=1}^K$ containing K example entity pairs, $h_k, t_k \in \mathcal{V}$, connected by relation r_i, and a query set $\mathcal{Q}_i = \{(h_j, r_i, ?)\}_{j=1}^J$ containing J query triples over entities in $\mathcal{V}$. As shown in Figure 1, the goal is then to learn how to extract knowledge from the support set such that we can predict the missing tail entities in the query set.

2.2 Overview of the framework

The foundation of our analyses focuses on a generalization of the current state-of-the-art gradient-based meta-learning approach (Chen et al., 2019). This approach follows the encoder-decoder paradigm of embedding-based knowledge graph completion methods (Hamilton et al., 2017), where the entities and relations are embedded in a low-dimensional vector space and the embeddings are used to predict the likelihood of a given triple.

Encoder functions. The key idea in few-shot learning is to transfer knowledge from support set to query set by learning a function $\texttt{RelLearner} : \mathcal{S}_i \mapsto \mathbb{R}^d$. This maps a support set $\mathcal{S}_i$, which characterizes the relation r_i, to a low dimensional embedding via an encoder function $\texttt{E} : \mathcal{V} \mapsto \mathbb{R}^d$

$$\mathbf{r}_i = \texttt{RelLearner}(\{(\texttt{E}(h_k), \texttt{E}(t_k)\}_{k=1}^K). \quad (1)$$

The `RelLearner` function can vary from a simple MLP (Hastie et al., 2009) to more complicated recurrent architectures (Rumelhart et al., 1985; Jordan, 1997; Hochreiter and Schmidhuber, 1997). Further, the entity encoder `E` can vary from TransE-style embeddings (Bordes et al., 2013; Sun et al., 2019) to a graph neural network (Schlichtkrull et al., 2018) that explicitly leverages the neighborhood information around entities.

Decoder and loss function. A decoder function ingests the embeddings of the entities $\mathbf{h}, \mathbf{t}$ and of the relation $\mathbf{r}$ to score the likelihood of a given triple (h, r, t). Using a simple TransE decoder (Bordes et al., 2013), it is then optimized to score positive triples higher than negative triples using a contrastive loss $\mathcal{L}$ (Dyer, 2014). In the few-shot setting we compute the support set loss $\mathcal{L}(\mathcal{S}_i)$, and the final query set loss $\mathcal{L}(\mathcal{Q}_i)$, which are used to update the model parameters (Chen et al., 2019).

Meta-gradient update. Instead of directly using the relation embedding $\mathbf{r}_i$ from Equation (1) to compute the final query loss $\mathcal{L}(\mathcal{Q}_i)$, we first make an update on the relation embedding using the gradient of the support set loss $\mathcal{L}(\mathcal{S}_i)$

$$\mathbf{r}'_i = \mathbf{r}_i - \eta \nabla_{\mathbf{r}_i} \mathcal{L}(\mathcal{S}) \quad (2)$$

where η denotes the learning rate. This update encourages $\mathbf{r_i}$ to be such that it effectively predicts the support set triples via minimizing $\mathcal{L}(\mathcal{S}_i)$.

2.3 Baselines

Our objective is to probe how much models leverage the support set to perform the query task. To this end, we perform a systematic study of different model variants, where each falls into the general framework described in Section 2.2.

MetaR follows Chen et al. (2019), where the `RelLearner` is defined as a 2-layer MLP (Hastie et al., 2009) over the support set entity embeddings. The encoder `E` simply maps each entity to a fixed learnable vector as in Bordes et al. (2013).

SharedEmbed skips Equation (1), and instead sets $\mathbf{r_i} = \mathbf{r_g}$, where $\mathbf{r_g}$ is a single learnable embedding shared across all relations. We propose this modification to measure the effect of representing all relations by the same embedding $\mathbf{r_g}$, where the only information from the support set comes via the gradient update in Equation (2).

ZeroShot further removes the meta-gradient update in Equation (2) and lets $\mathbf{r'_i} = \mathbf{r_g}$. This effectively reduces the model to perform zero-shot link prediction on the relation's query set without any relation-specific information from the support set.

		MRR		Hits@10		Hits@5		Hits@1	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Nell-One	MetaR	0.294	0.323	0.464	0.500	0.398	0.426	0.201	0.230
	SharedEmbed	0.276	0.311	0.454	0.495	0.382	0.420	0.173	0.205
	ZeroShot	0.199	0.219	0.342	0.365	0.283	0.303	0.116	0.136
	R-GCN	0.216	0.267	0.412	0.464	0.316	0.366	0.120	0.172
Wiki-One*	MetaR	0.325	0.326	0.448	0.408	0.408	0.367	0.258	0.280
	SharedEmbed	0.290	0.311	0.399	0.415	0.348	0.378	0.238	0.254
	ZeroShot	0.279	0.289	0.361	0.367	0.337	0.341	0.234	0.246
	R-GCN	0.126	0.137	0.178	0.237	0.130	0.152	0.101	0.104

Table 1: Average metrics on Nell-One and Wiki-One few-shot link prediction tasks. * For Wiki-One we used pre-trained embeddings, similar to Chen et al. (2019).

R-GCN uses the same `RelLearner` as in MetaR, with the exception that support set entity embeddings are learned via a multi-relational graph neural network, R-GCN (Schlichtkrull et al., 2018), instead of a TransE-style embeddings. R-GCN learns entity representation via aggregating the 2-hop neighbors of a given entity. With this model we probe the extent to which injecting structural bias into entity representations can influence performance in link prediction.

3 Null Models

In order to probe and understand the performance of different models, we introduce two *null models*, which are used to generate synthetic data that satisfy certain properties. Motivated by recent literature on position versus structure-aware methods in relational learning (You et al., 2019; Srinivasan and Ribeiro, 2020), we test the models' ability to learn from two key sources of information: *structural information* and *positional information*. In the context of knowledge graphs, *structural information* corresponds to the fine-grained relational semantics. These are the logical patterns that are extracted by state-of-the-art rule induction systems, such as RuleN (Meilicke et al., 2018).

On the other hand, *positional information* corresponds to the coarse-grained community structure of the nodes in the graph. In other words, two nodes are said to be positionally 'close' in the graph, i.e., if they belong to the same community (Newman, 2018).

3.1 Structural Null Models

The first type of null models contains synthetic relations that satisfy simple logical properties. For the sake of exposition, we focus on two simple logical patterns: symmetry and transitivity. For the purposes of all synthetic data generation, we only consider the largest connected component of respective datasets, denoted $\mathcal{G}^L$.

Synthetic symmetric relations. To generate $2N$ edges connected by a symmetric relation r^*_{s}, we repeat the following steps N times.:

1. Uniformly sample a pair of unique entities–h_i, t_i–from all the entities in $\mathcal{G}^L$.
2. Add two edges–$((h_i, r^*_{\mathrm{s}}, t_i)), ((t_i, r^*_{\mathrm{s}}, r_i))$ to the set of synthetic symmetric edges.

Synthetic transitive relations. To sample $3N$ edges connected by a transitive relation r^*_{t}, we generate 3 edges at a time. In particular, we repeat the following steps N times:

1. Uniformly sample 3 unique entities–e_1, e_2, and e_3–from all the entities in $\mathcal{G}^L$.
2. Add three edges–$(e_1, r^*_{\mathrm{t}}, e_2)$, $(e_2, r^*_{\mathrm{t}}, e_3)$, $(e_1, r^*_{\mathrm{t}}, e_3)$–to our collection.

3.2 Positional Null Models

The second type of null models focuses on generating synthetic relations that depend on the underlying community structure in the graph. We call these relations *positional* because they depend on the relative global position of the entities, rather than on local structural properties.

We first cluster the largest connected component $\mathcal{G}^L$ into K communities using a standard algorithm originally proposed by Blondel et al. (2008). Let $\{C_i\}_{i=1}^K$ denote the set of communities generated, where each community is a set of entities from $\mathcal{G}^L$. To generate N synthetic edges for a positional relation r_p, we repeat the following steps N times:

1. Uniformly sample a community index i from the set $\{1, , K\}$.
2. Uniformly sample two unique entities h, t from community C_i; add (h, r^*_{p}, t) to the set.

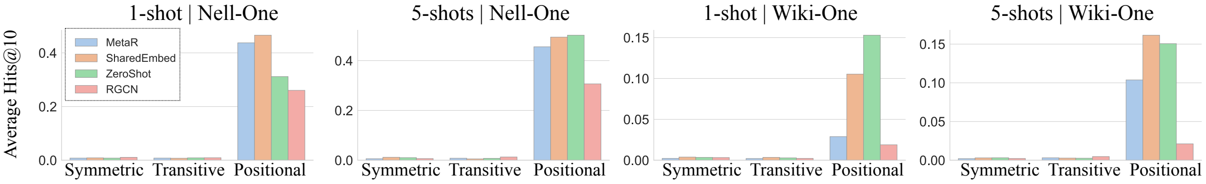


Figure 2: Average Hits@10 on synthetically generated relations using our proposed null models.

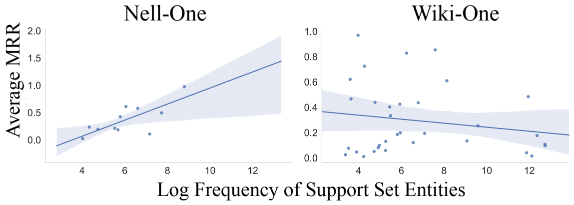


Figure 3: Pearson's R between MRR and the log frequency of support set entities in training graph. We observe strong correlation for Nell-One, but not for Wiki-One. For more details on this, see Appendix C.

4 Experiments

We followed the same experimental setup as in Chen et al. (2019), as described in Appendix A. We conducted our experiments on the Nell-One and Wiki-One datasets[1]. For more details on these benchmarks, we refer the reader to Table 1 in Xiong et al. (2018). Similar to earlier work, we report MRR, Hits@1, Hits@5 and Hits@10 on our test relations, using a type-constrained candidate set.

4.1 Results and Analysis

Experiments on Real Data. As shown in Table 1, for Nell-One, we find that SharedEmbed model yields competitive performance to MetaR, with Hits@10 of 45.4% and 49.5%, as compared to MetaR's Hits@10 of 46.4% and 50.0% for 1 and 5-shot, respectively. The same observation holds for Wiki-One, where SharedEmbed yields 39.9% and 41.5% Hits@10, compared to MetaR's Hits@10 of 44.8% and 40.8%, for 1 and 5-shot, respectively.

It is surprising how competitive SharedEmbed is, given that the only relation-specific information the model gets to observe comes via the meta-gradient update in Equation (2). In fact, we find that even in absence of this gradient signal, i.e., without *any* relation-specific information, ZeroShot performs relatively good, with Hits@10 of 34.2% and 36.5% on Nell-One, and 36.1% and 36.7% on Wiki-One.

The nontrivial performance of these simple models suggests that such models may exploit some easily accessible positional signals around entities, without the need to learn meaningful representations for relations. In fact, Figure 3 shows a high correlation between performance and the degrees of entities in the support set for Nell-One. We reconcile this observation by noting that as models observe more signals about entities, they start relying less on the support set, and thus on the relation representations. Furthermore, contrary to our expectation, even when we equip models with structural biases, as done via an R-GCN, they do not yield better results.

Null Model Experiments. We probed the above trained models on the synthetically generated test tasks following the procedure discussed in Section 3. As shown in Figure 2, we find a consistent trend for these models to yield higher performance on tasks that rely on positional signals, as compared to tasks that require logical inference.

Indeed, in the current task formulation, where we are given a support set of K randomly sampled examples, it is unlikely that logically consistent patterns will be captured in the K-shot examples. For example, seeing conclusive evidence of transitivity when only given a small random sample of tuples is highly unlikely. In fact, as we show in Appendix B, one provably cannot learn certain logical patterns for some values of K in the K-shot setting.

5 Conclusion

We conducted a systematic study of various models to probe their limits in performing few-shot link prediction. Our experiments on both synthetic and real data show that the current task formulation encourages models to mainly rely on positional information around entities, rather than leveraging logical signals about relations. In fact, we empirically show that having only K examples of a relation fundamentally limits the types of logical patterns that can be learned. We argue that a future direction in few-shot link prediction should allow for a more careful construction of the support set, to scaffold the use of logical patterns in few-shot learning.

---

[1] Datasets can be downloaded under this link.

Acknowledgments

The authors would like to thank Priyesh Vijayan, Joey Bose, Lu Liu and other members of the RLLab and Mila for their invaluable feedback and useful discussions on the early drafts. Furthermore, the authors are grateful for the anonymous reviewers for their comments on the first draft of the paper. This research was supported in part by Canada CIFAR AI Chairs held by Prof. Pineau and Prof. Hamilton, as well as gift grants from Microsoft Research and Samsung AI.

References

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4217–4226. Association for Computational Linguistics.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Chris Dyer. 2014. Notes on noise contrastive estimation and negative sampling. *arXiv preprint arXiv:1410.8251*.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Michael I Jordan. 1997. Serial order: A parallel distributed processing approach. In *Advances in psychology*, volume 121, pages 471–495. Elsevier.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Christian Meilicke, Manuel Fink, Yanjie Wang, Daniel Ruffinelli, Rainer Gemulla, and Heiner Stuckenschmidt. 2018. Fine-grained evaluation of rule- and embedding-based systems for knowledge graph completion. In *International Semantic Web Conference*, pages 3–20. Springer.

Mark Newman. 2018. *Networks*. Oxford university press.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Balasubramaniam Srinivasan and Bruno Ribeiro. 2020. On the equivalence between positional node embeddings and structural graph representations. In *International Conference on Learning Representations*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Théo Trouillon, Christopher R Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard. 2017. Knowledge graph completion via complex tensor factorization. *The Journal of Machine Learning Research*, 18(1):4735–4772.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-shot relational learning for knowledge graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1980–1990. Association for Computational Linguistics.

Jiaxuan You, Rex Ying, and Jure Leskovec. 2019. Position-aware graph neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 72*.

A Hyperparameters

As discussed, we followed the experimental setup described in Chen et al. (2019). We used the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001, using 1 to 3 ratio of positive to negative samples. During training, we used 3 queries per task on each dataset.

We adapted the batch size to be 1024, and the number of queries to test on to be 3, based on their open-sourced codebase. These hyperparameters yielded the best performing models. Similarly, we also used the same train/validation/test relation splits of 51:5:11 and 133:16:34 for Nell-One and Wiki-One respectively.

For our R-GCN model, we considered a range of [5, 10, 20] as the number of neighbors to sample for each message passing step, and [2, 4] as the number of basis. Furthermore, we used 2 layers in the R-GCN. Finally, we used 50 and 20 as the R-GCN hidden layer dimension for Nell-One and Wiki-One, respectively. These hyperparameters were partly followed from Schlichtkrull et al. (2018), and were decided upon consideration for our available compute infrastructure.

Our models were trained on a single Nvidia 1080Ti GPU, and each model training took between 13-18 hours depending on the model and dataset settings.

B Logical inference in the few-shot domain

Figure 4 shows five example support sets to demonstrate that inferring logical properties such as symmetry and transitivity requires a minimum number of carefully designed K-shot examples.

C Entity Frequency Analysis

Figure 5 shows the 100 highest-degree entities out of all 4,838,244 entities in the Wiki-One knowledge graph. We find that the median degree of entities is 1, and the highest degree is 227,390, which connects to 4.69% of the total graph. We suspect that these high-degree entities, so-called hub nodes, may add noise to the embeddings of support set entities. This could in turn affect performance and explain why we do not observe a strong correlation between the degrees of support set entities and performance in Wiki-One.

K-shot	Example support sets of varying K-shots.	Query triple	Answer
3	[(Bob, coworker_of, Alice), (Alice, coworker_of, Ella), (Ella, coworker_of, Bob)]	(Stella, coworker_of, ?)	N/A
4	[(Bob, coworker_of, Alice), (Alice, coworker_of, Ella), (Ella, coworker_of, Bob), (Stella, coworker_of, Alice)]	(Stella, coworker_of, ?)	Ella, Bob
5	[(Bob, coworker_of, Alice), (Alice, coworker_of, Ella), (Ella, coworker_of Bob), (Larry, coworker_of, Jim), (Jim, coworker_of, Laura)]	(Laura, coworker_of, ?)	Larry
2	[(Liz, friend_of, Rob), (Rob, friend_of, Liz)]	(Peter, friend_of, ?)	N/A
3	[(Liz, friend_of, Rob), (Rob, friend_of, Liz), (Barbara, friend_of, Peter)]	(Peter, friend_of, ?)	Barbara

Figure 4: Limits of logical inference in the few-shot domain.

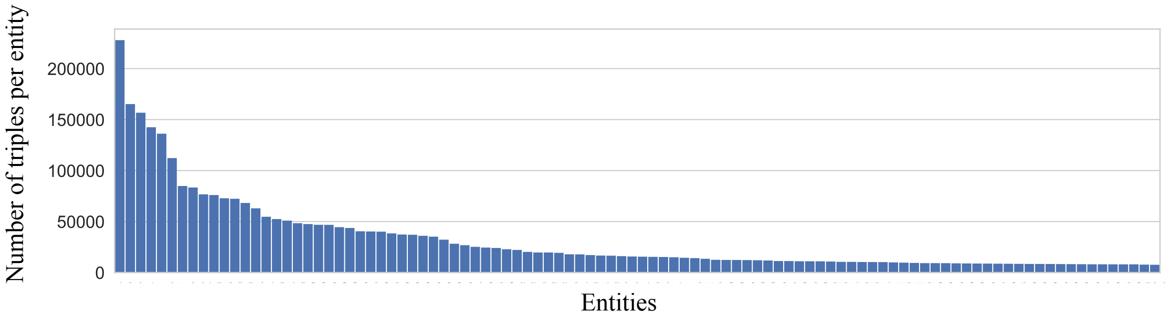


Figure 5: Entity degree distribution of Wiki-One

ProFormer: Towards On-Device LSH Projection Based Transformers

Chinnadhurai Sankar
Mila, Université de Montréal*
Montreal, QC, Canada
chinnadhurai@gmail.com

Sujith Ravi
Amazon Alexa†
Sunnyvale, CA, USA
sravi@sravi.org

Zornitsa Kozareva
Google
Mountain View, CA, USA
zornitsa@kozareva.com

Abstract

At the heart of text based neural models lay word representations, which are powerful but occupy a lot of memory making it challenging to deploy to devices with memory constraints such as mobile phones, watches and IoT. To surmount these challenges, we introduce ProFormer – a projection based transformer architecture that is faster and lighter making it suitable to deploy to memory constraint devices and preserve user privacy. We use LSH projection layer to dynamically generate word representations on-the-fly without embedding lookup tables leading to significant memory footprint reduction from $\mathcal{O}(V.d)$ to $\mathcal{O}(T)$, where V is the vocabulary size, d is the embedding dimension size and T is the dimension of the LSH projection representation. We also propose a *local projection attention (LPA)* layer, which uses self-attention to transform the input sequence of N LSH word projections into a sequence of N/K representations reducing the computations quadratically by $\mathcal{O}(K^2)$.

We evaluate ProFormer on multiple text classification tasks and observed improvements over prior state-of-the-art on-device approaches for short text classification and comparable performance for long text classification tasks. ProFormer is also competitive with other popular but highly resource-intensive approaches like BERT and even outperforms small-sized BERT variants with significant resource savings – reduces the embedding memory footprint from 92.16 MB to 1.7 KB and requires 16× less computation overhead, which is very impressive making it the fastest and smallest on-device model.

1 Introduction

Transformers (Vaswani et al., 2017) based architectures like BERT (Devlin et al., 2018), XL-net (Yang et al., 2019), GPT-2 (Radford et al., 2019), MT-DNN (Liu et al., 2019a), RoBERTA (Liu et al., 2019b) reached state-of-the-art performance on tasks like machine translation (Arivazhagan et al., 2019), language modelling (Radford et al., 2019), text classification benchmarks like GLUE (Wang et al., 2018). However, these models require huge amount of memory and need high computational requirements making it hard to deploy to small memory constraint devices such as mobile phones, watches and IoT. Recently, there have been interests in making BERT lighter and faster (Sanh et al., 2019; McCarley, 2019). In parallel, recent on-device works like SGNN (Ravi and Kozareva, 2018), SGNN++ (Ravi and Kozareva, 2019) and (Sankar et al., 2019) produce lightweight models with extremely low memory footprint. They employ a modified form of LSH projection to dynamically generate a fixed binary projection representation, $\mathbb{P}(x) \in [0, 1]^T$ for the input text x using word or character n-grams and skip-grams features, and a 2-layer MLP + softmax layer for classification. As shown in (Ravi and Kozareva, 2018) these models are suitable for short sentence lengths as they compute T bit LSH projection vector to represent the entire sentence. However, (Kozareva and Ravi, 2019) showed that such models cannot handle long text due to significant information loss in the projection operation.

On another side, recurrent architectures represent long sentences well, but the sequential nature of the computations increases latency requirements and makes it difficult to launch on-device. Recently, self-attention based architectures like BERT (Devlin et al., 2018) have demonstrated remarkable success in capturing long term dependencies in the input text via purely attention mechanisms. BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation in (Vaswani et al., 2017). The

---

*Work done during internship at Google

†Work done while at Google AI

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2823–2828
April 19 - 23, 2021.

self-attention scores can be computed in parallel as they do not have recurrent mechanisms. But usually these architectures are very deep and the amount of computation is quadratic in the order of $\mathcal{O}(L \cdot N^2)$, where L is the number of layers (Transformer blocks) and N is the input sentence length. Straightforward solutions like reducing the number of layers is insufficient to launch transformers on-device due to the large memory and quadratic computation requirements.

In this paper, we introduce a projection-based neural architecture ProFormer that is designed to (a) be efficient and learn compact neural representations (b) handle out of vocabulary words and misspellings (c) drastically reduce embedding memory footprint from hundreds of megabytes to few kilobytes and (d) reduce the computation overhead *quadratically* by introducing a local attention layer which reduces the intermediate sequence length by a constant factor, K. We achieve this by bringing the best of both worlds by combining LSH projection based representations (for low memory footprint) and self-attention based architectures (to model dependencies in long sentences). To tackle computation overheard in the transformer based models, we reduce the number of self-attention layers and additionally introduce an intermediate local projection attention (LPA) to quadratically reduce the number of self-attention operations. The main contributions of our paper are:

- We propose novel on-device neural network called ProFormer which combines LSH projection based text representations, with transformer architecture and locally projected self-attention mechanism that captures long range sentence dependencies while yielding low memory footprint and low computation overhead.
- ProFormer reduces the computation overhead $\mathcal{O}(L \cdot N^2)$ and latency in multiple ways: by reducing the number of layers L from twelve to two and introducing new local projection attention layer that decreases number of self-attention operations by a quadratic factor.
- ProFormer is light weigh compact on-device model, while BERT on-device still needs huge embedding table (92.16 MB for $V = 30k$, $d = 768$) with number of computation flops in the order of $\mathcal{O}(L \cdot N^2)$, where L is the number of layers, N is the number of words in the input sentence.
- We conduct empirical evaluations and comparisons against state-of-the-art on-device and prior deep learning approaches for short and long text classification. Our model ProFormer reached state-of-art performance for short text and comparable performance for long texts, while maintaining small memory footprint and computation requirements.

2 ProFormer: LSH Projection based Transformers

In this section, we show the overall architecture of ProFormer in Figure 1. ProFormer consists of multiple parts: (1) word-level Locality Sensitive Hashing (LSH) projection layer, (2) local projection attention (LPA) layer, (3) transformer layer (Devlin et al., 2018) and (4) a max-pooling + classifier layer. Next, we describe each layer in detail.

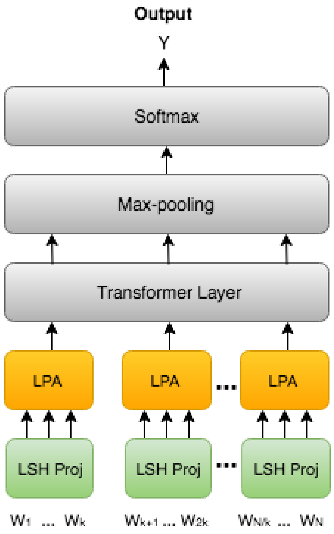


Figure 1: ProFormer: Our **Pro**jection Trans**former** Network Architecture

2.1 LSH Projection Layer

It is a common practice to represent each word in the input sentence, $\mathbf{x} = [w_1, w_2, \cdots, w_N]$ as an embedding vector based on its one-hot representation. Instead, we adopt LSH projection layer from (Ravi, 2017, 2019) which dynamically generates a T bit representation, $\mathbb{P}(w_i) \in [0, 1]^T$ for the input word, w_i based on its morphological features like n-grams, skip-grams from the current and context words, parts-of-speech tags, etc.

Since the LSH projection based approach does not rely on embedding lookup tables to compute word representation, we obtain significant memory savings of the order, $O(V \cdot d)$, where V is the vocabulary size and d is the embedding dimension. For instance, the embedding look-up table occupies 92.16 MB ($V = 30k$, $d = 768$ (Devlin et al., 2018)), while the LSH projection layer requires only $\approx$ 1.7 KB ($T = 420$) as shown in Table 1.

Models	**Embedding memory**	**Computations**
BERT	$\mathcal{O}(V.d)$	$\mathcal{O}(N^2)$
ProFormer (our model)	$\mathcal{O}(T)$	$\mathcal{O}(N^2/K^2)$

Table 1: Memory and computations overhead comparison between BERT (Devlin et al., 2018) and ProFormer (our model). N is the number of words in the input. For $V = 30k$, $d = 768$, $T = 420$, BERT's embedding table occupies 92.16 MB while ProFormer requires only **1.7** KB. For $K = 4$, we reduce the BERT computation overhead by **16** times.

2.2 Local Projection Attention (LPA) Layer

The LPA layer shown in Figure 2 consists of a single layer multi-headed self-attention layer similar to the Transformer architecture in (Vaswani et al., 2017) followed by a max-pooling layer yielding a compressed representation of K input words, $[w_1, w_2, \cdots w_K]$.

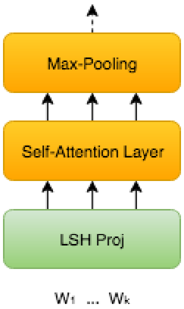


Figure 2: Local Projection Attention (LPA) layer.

The LPA layer transforms the N word-level projections, $\mathbb{P}(w_i)$ to a sequence of N/K representations as in Equation 1.

$$[\mathbb{P}(w_1), \cdots \mathbb{P}(w_N)]_N \longrightarrow [\mathcal{LPA}(\mathbb{P}(w_{1:K})), \cdots \mathcal{LPA}(\mathbb{P}(w_{N/K:N}))]_{N/K} \quad (1)$$

where $\mathcal{LPA}$ consists of the self-attention and max-pooling operation, K is a *Group factor*[1]. We equally divide the N word-level LSH projection representations into N/K groups of size K. The LPA layer compresses each group of K word representations into $\mathcal{LPA}(\mathbb{P}(w_{1:K})) \in \mathbb{R}^d$ yielding N/K representations in total. The LPA layer reduces the self-attention computation overhead in the subsequent transformer layer (Vaswani et al., 2017) by $\mathcal{O}(K^2)$.

2.3 Transformer Layer

This layer consists of 2-layer bidirectional Transformer encoder based on the original implementation described in (Vaswani et al., 2017). This layer transforms the N/K input representations from the LPA layer described in the previous sub-section into N/K output representations. In this layer, we reduce both the computation overhead and memory footprint by reducing the number of layers from L to 2 reducing the computation overhead by $\mathcal{O}(L/2)$ (6 times in the case of 12-layer *BERT-base* model).

2.4 Max-Pooling and Classification Layer

We summarize the N/K representations from the transformer layer to get a single d dimensional vector by max-pooling across the N/K time-steps, followed by a softmax layer to predict the output class Y.

3 Datasets & Experimental Setup

In this section, we describe our datasets and experimental setup. We use text classification datasets from state-of-the-art on-device evaluations such as: MRDA (Shriberg et al., 2004) and ATIS (Tür et al., 2010), AG News (Zhang et al., 2015a) and Yahoo! Answers (Zhang et al., 2015a). Table 2 shows the characteristics of each dataset.

Tasks	# Classes	Avg-len	Train	Test
MRDA (Dialog act)	6	8	78k	15k
ATIS (Intent prediction)	21	11	$4.4k$	$0.89k$
AG (News Categorization)	4	38	120k	7.6k
Y!A (Yahoo! Answers Categorization)	10	108	1400k	60k

Table 2: Classification Dataset Characteristics

We train ProFormer on multiple classification tasks individually and report *Accuracy* on corresponding test sets. We fix the projection size, $T = 420$, n-gram size=5, skip-gram size=1 for the LSH projection operation, $\mathbb{P}$. For the LPA layer, We experiment with two values for $K = 1, 4$, where $K = 1$ corresponds to the null operation in the LPA layer which just passes the word LSH projection representation to the Transformer layer. For the transformer layer, we fix the number of layers, $L = 2$ and set all layer sizes, $d = 768$ (including the intermediate size for the dense layer).[2]

---

[1] We choose K such that N is divisible by K.

[2] The rest of the parameters are same as the one used in

We compare our model with previous state of the art neural architectures, including on-device approaches. We also fine-tune the pretrained 12-layer *BERT-base* model (Devlin et al., 2018) on all classification tasks and compare to our model. *BERT-base* consists 12-layers of transformer blocks (Vaswani et al., 2017) and is pretrained in an unsupervised manner on a large corpus (BooksCorpus (Zhu et al., 2015) and English WikiPedia) using masked-language model objective. We fine-tune the pretrained *BERT-base* (Devlin et al., 2018) to each of the classification tasks. For training, we use Adam with learning rate of 1e-4, β_1=0.9, β_2=0.999, $L2$ weight decay of 0.01, learning rate warmup over the first $10,000$ steps, and linear decay of the learning rate. We use dropout probability of 0.1 on all layers and training batch size of 256. For further comparison, we also trained much smaller BERT baselines with 2-layers of transformer blocks and smaller input embedding sizes.

4 Results

Tables 3 and 4 show the results on the ATIS & MRDA short text classification and AG & Y!A long text classification tasks. We compare our approach, ProFormer against prior state-of-the-art on-device works, fine-tuned *BERT-base*, smaller 2-layer *BERT* variants and other non-on-device neural approaches.

Overall, our model ProFormer improved upon non-on-device neural models while keeping very small memory footprint and high accuracy. This is very impressive since ProFormer can be directly deployed to memory constraint devices like phones, watches and IoT while still maintaining high accuracy. ProFormer also improved upon prior on-device state-of-the-art neural approaches like SGNN (Ravi and Kozareva, 2018) and SGNN++ (Ravi and Kozareva, 2019) reaching over 35% improvement on long text classification. Similarly it improved over on-device ProSeqo (Kozareva and Ravi, 2019) models for all datasets and reached comparable performance on MRDA. In addition to the quality improvements, ProFormer also keeps smaller memory footprint than ProSeqo, SGNN and SGNN++.

In addition to the non-on-device and on-device neural comparisons, we also compare against *BERT-base* and other smaller variants. Our experiments show that ProFormer outperforms the small *BERT* baselines on all tasks. Moreover, although the 12-layer fine-tuned *BERT-base* (Devlin et al., 2018) model converged to the state-of-the-art in almost all of the tasks, ProFormer converges to $\approx 97.2\%$ *BERT-base*'s performance on an average while occupying only 13% of *BERT-base's* memory. ProFormer has 14.4 million parameters, while BERT-base has 110 million. For fair comparison, we also test ProFormer with $K = 4$, which only occupies 38.4% the memory footprint of 2-layer BERT-base model and reduces the computation overhead by 16 times. The embedding look up table occupies nearly 23 million parameters out of 38 million parameters in the 2-layer BERT model. We notice that K=4 model performs slightly worse than K=1 indicating information loss in the LPA layer. Overall, our experiments demonstrate that ProFormer reaches better performances that prior non-on-device and on-device neural approaches, and comparable performance to BERT-base models while preserving smaller memory footprint.

Models	MRDA	ATIS
ProFormer (K=1) (our model)	**89.3**	**98.2**
ProFormer (K=4) (our model)	86.7	97.0
BERT-base + fine-tuned (Devlin et al., 2018) *(12-layers, embedding size = 768)*	**90.1**	**98.3**
BERT *(2-layer, embedding size = 560)*	77.0	94.0
BERT *(2-layer, embedding size = 840)*	76.8	95.0
ProSeqo (Kozareva and Ravi, 2019)(on-device)	**90.1**	97.8
SGNN++ (Ravi and Kozareva, 2019)(on-device)	87.3	93.7
SGNN (Ravi and Kozareva, 2018)(on-device)	86.7	88.9
RNN(Khanpour et al., 2016)	86.8	-
RNN+Attention(Ortega and Vu, 2017)	84.3	-
CNN(Lee and Dernoncourt, 2016)	84.6	-
GatedIntentAtten.(Goo et al., 2018)	-	94.1
GatedFullAtten.(Goo et al., 2018)	-	93.6
JointBiLSTM(Hakkani-Tur et al., 2016)	-	92.6
Atten.RNN(Liu and Lane, 2016)	-	91.1

Table 3: Short text classification results.

Models	AG	Y!A
ProFormer (K=1) (our model)	**92.0**	**72.8**
ProFormer (K=4) (our model)	91.5	71.1
BERT-base + fine-tuned (Devlin et al., 2018) *(12-layers, embedding size = 768)*	**94.5**	**73.8**
BERT *(2-layer, embedding size = 560)*	82.3	-
BERT *(2-layer, embedding size = 840)*	83.3	-
ProSeqo (Kozareva and Ravi, 2019)(on-device)	91.5	72.4
SGNN (Ravi and Kozareva, 2018)(on-device)	57.6	36.5
FastText-full (Joulin et al., 2016)	**92.5**	72.3
CharCNNLargeWithThesau.(Zhang et al., 2015b)	90.6	71.2
CNN+NGM (Bui et al., 2018)	86.9	-
LSTM-full (Zhang et al., 2015b)	86.1	70.8

Table 4: Long text classification results.

5 Conclusion

We proposed a novel on-device neural network ProFormer, which combines LSH projection based

---

$bert\_config.json$ in *BERT-base* model (Devlin et al., 2018) text representations, with trans-former architecture and locally projected self-attention mechanism that captures long range sentence dependencies. Overall, ProFormer yields low memory footprint and reduces computations quadratically. In series of experimental evaluations on short and long text classifications we show that ProFormer improved upon prior neural models and on-device work like SGNN (Ravi and Kozareva, 2018), SGNN++ (Ravi and Kozareva, 2019) and ProSeqo (Kozareva and Ravi, 2019). ProFormer reached comparable performance to our BERT-base implementation, however it produced magnitudes more compact models than BERT-base. This is very impressive showing both effectiveness and compactness of our neural model.

References

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Thang D. Bui, Sujith Ravi, and Vivek Ramavajjala. 2018. Neural graph learning: Training neural networks using graphs. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 64–71.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757. Association for Computational Linguistics.

Dilek Hakkani-Tur, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Vivian Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Proceedings of The 17th Annual Meeting of the International Speech Communication Association (INTERSPEECH 2016)*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*, abs/1612.03651.

Hamed Khanpour, Nishitha Guntakandla, and Rodney Nielsen. 2016. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2012–2021.

Zornitsa Kozareva and Sujith Ravi. 2019. ProSeqo: Projection sequence networks for on-device text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3892–3901.

Ji Young Lee and Franck Dernoncourt. 2016. Sequential short-text classification with recurrent and convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *Proceedings of The 17th Annual Meeting of the International Speech Communication Association (INTERSPEECH 2016)*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. *CoRR*, abs/1901.11504.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

J. Scott McCarley. 2019. Pruning a bert-based question answering model. *ArXiv*, abs/1910.06360.

Daniel Ortega and Ngoc Thang Vu. 2017. Neural-based context representation learning for dialog act classification. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 247–252.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Sujith Ravi. 2017. Projectionnet: Learning efficient on-device deep networks using neural projections. *CoRR*, abs/1708.00630.

Sujith Ravi. 2019. Efficient on-device models using neural projections. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5370–5379.

Sujith Ravi and Zornitsa Kozareva. 2018. Self-governing neural networks for on-device short text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 804–810.

Sujith Ravi and Zornitsa Kozareva. 2019. On-device structured and context partitioned projection networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3784–3793, Florence, Italy. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Chinnadhurai Sankar, Sujith Ravi, and Zornitsa Kozareva. 2019. Transferable neural projection representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3355–3360, Minneapolis, Minnesota. Association for Computational Linguistics.

Elizabeth Shriberg, Rajdip Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of the SIGDIAL 2004 Workshop, The 5th Annual Meeting of the Special Interest Group on Discourse and Dialogue, April 30 - May 1, 2004, Cambridge, Massachusetts, USA*, pages 97–100.

Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck. 2010. What is left to be understood in atis? In *Proceedings of 2010 IEEE Spoken Language Technology Workshop (SLT)*, pages 19–24.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 649–657. MIT Press.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27.

Joint Learning of Hyperbolic Label Embeddings for Hierarchical Multi-label Classification

Soumya Chatterjee[1*] **Ayush Maheshwari**[1*]
Ganesh Ramakrishnan[1] **Saketha Nath Jagarlapudi**[2]
[1] Indian Institute of Technology Bombay
[2] Indian Institute of Technology Hyderabad
{soumya, ayusham, ganesh}@cse.iitb.ac.in, saketha@iith.ac.in

Abstract

We consider the problem of multi-label classification, where the labels lie in a hierarchy. However, unlike most existing works in hierarchical multi-label classification, we do not assume that the label-hierarchy is known. Encouraged by the recent success of hyperbolic embeddings in capturing hierarchical relations, we propose to jointly learn the classifier parameters as well as the label embeddings. Such a joint learning is expected to provide a twofold advantage: i) the classifier generalises better as it leverages the prior knowledge of existence of a hierarchy over the labels, and ii) in addition to the label co-occurrence information, the label-embedding may benefit from the manifold structure of the input datapoints, leading to embeddings that are more faithful to the label hierarchy. We propose a novel formulation for the joint learning and empirically evaluate its efficacy. The results show that the joint learning improves over the baseline that employs label co-occurrence based pre-trained hyperbolic embeddings. Moreover, the proposed classifiers achieve state-of-the-art generalization on standard benchmarks. We also present evaluation of the hyperbolic embeddings obtained by joint learning and show that they represent the hierarchy more accurately than the other alternatives. The source code of the paper is available here.

1 Introduction

The problem of multi-label text classification is well known and extensively studied in literature (McCallum, 1999; Yang et al., 2009; Liu et al., 2017). The fundamental assumption is that a document is associated with multiple labels from a fixed vocabulary of labels. Often, these labels are organised in a hierarchical structure. For ex. consider a sample headline from the NYT (NewYork Times) corpus "Voice Recognition Is Improving, but Don't Stop the Elocution Lessons" for which labels are "Top/News/Technology". Here, labels are arranged in a hierarchy, hereafter referred as label hierarchy. We undertake the task of labelling documents with classes that are hierarchically organised; this problem is popularly known as *hierarchical multi-label text classification* (HMC). HMC methods have found several applications in online advertising systems (Agrawal et al., 2013), bio-informatics (Peng et al., 2016; Triguero and Vens, 2016), text classification (Rousu et al., 2006; Mao et al., 2019).

The main challenge in HMC is in modelling classification of the document into a large, imbalanced and structured output space. In HMC, the label taxonomy is a partially ordered set $(L, \prec)$ where L is a finite set of all class labels. Relation $\prec$ refers to *is-a* relationship between labels, which is asymmetric, anti-reflexive and transitive (Silla and Freitas, 2011).

Hierarchical structures can provide important insights for learning and classification tasks. However, explicit knowledge of hierarchy is not available in several domains, for instance, extreme classification datasets (Bhatia et al., 2016). In this paper, we consider the problem of structured prediction from unstructured text, in which label hierarchy is not known *apriori*. We infer hierarchies from classification judgements on the outputs that are readily available. We focus on discovering relationships between the labels in a hyperbolic space, which has natural capacity to encode hierarchical structures.

In our approach, HIDDEN (**H**yperbol**I**c label embe**DD**ings for hi**E**rarchical multi-label classificatio**N**), the labels are represented in a hyperbolic space to help respect their latent hierarchical organisation. We use this intuition to learn label embeddings for HMC without explicit supervision

---

[*] Equal contribution

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2829–2841
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

on the label hierarchy.

Apart from employing hyperbolic embeddings, another key aspect of our methodology is that the parameters of the classifier as well as of the label embedding are learnt jointly. We next explain the advantage in doing so. In the absence of any partial information regarding the hierarchy, label embeddings are typically learnt using the weak supervision available in label co-occurrences (Nickel and Kiela, 2017, 2018). This weak form of supervision can be complemented if the label embedding learning is also aware of the manifold structure of the input (documents). For *e.g.*, similar documents may have similar labels, *etc.* Such a strengthening is possible only if learning happens in a joint fashion. Moreover, the generalization of the classifier also improves because of the improved embeddings (and vice-versa). Our contributions can be summarised as follows:

1. We present an approach HIDDEN, that models the implicit hierarchical organisation of labels for improved classification. It leverages properties of hyperbolic geometry to help learn embeddings for the hierarchically organised labels.
2. We present a novel formulation for jointly learning the parameters of the classifier as well as the label embedding, which can be trained solely using the supervision from the training data, and without using any explicit information regarding the label hierarchy.
3. We evaluate HIDDEN on real-world as well as synthetic datasets and show:
   (a) significant improvement over classical multi-label classification methods as well as baselines that employ hyperbolic label embeddings learnt in isolation solely based on label co-occurrence information
   (b) HIDDEN sometimes generalizes even better than state-of-the-art hierarchical multi-label classifiers that have complete access to the true label hierarchy
   (c) label embeddings learnt using the joint optimisation approach correlate better with the ground truth than other alternatives.

2 Related Work

Several conventional classification methods are capable of handling classification in multi-label settings. However, relatively fewer of these are designed to incorporate the possibly hierarchical organisation of the class labels. These include both traditional methods (Gopal and Yang, 2013; Lewis et al., 2004) as well as deep learning methods (Johnson and Zhang, 2015; Peng et al., 2018) across varied domains such as news articles, web content, *etc.* Some approaches (Bairi et al., 2015, 2016) have also attempted to identify a subset of class labels from the classification hierarchy that effectively *represents* most instances from the training dataset.

Traditional or flat classification approaches typically perform prediction assuming that all the classes are independent of each other, ignoring the class hierarchy. Whereas 'local' classification approaches (Koller and Sahami, 1997; Cesa-Bianchi et al., 2006) train a set of classifiers at each level of the hierarchy. However, it has also been argued (Cerri et al., 2011) that it is impractical to train separate classifiers at each level. On the other hand, 'global' approaches (Silla Jr and Freitas, 2009; Wang et al., 2001) train a single classifier that factors in the complete class hierarchy, while often also explicitly factoring in the label-label correlation (Kulkarni et al., 2018). Unlike the local approach, 'global' approaches do not suffer from the error propagation problem, although they are prone to under-fit by not considering local information in the hierarchy.

Some recent papers have proposed a mix of local and global approaches for HMC. Wehrmann et al. (2018) propose an objective that leverages both local and global information while introducing global hierarchical violation penalty. Mao et al. (2019) employ a reinforcement learning framework to learn a label assignment policy. They model HMC as a *markov decision process*, wherein, the agent takes an action of label assignment on the tree hierarchy and receives scalar rewards as feedback for the actions. Chen et al. (2019) embed both document and label hierarchy in the same hyperbolic space and use interactions between these embeddings for HMC. Our approach differs from these in two important ways: (i) we embed only labels into the hyperbolic space and (ii) label hierarchy is not known *apriori* - all we assume is that there is some hidden hierarchy.

Recently, the use of hyperbolic geometry has been found to be promising in machine learning and network sciences to model data with latent hierarchies. Krioukov et al. (2010) showed that properties of complex networks, namely *heterogeneous degree distribution* and *strong clustering*, naturally manifest in hyperbolic geometry. They showed that if a network has some heterogeneous degree distribution and metric structure, the network can be mapped effectively to the hyperbolic space (since euclidean distance has limitations in approximating the distance between nodes in a tree). Gromov (1987) have shown that any finite tree structure can be embedded into a finite hyperbolic space while preserving the distance between nodes. Nickel and Kiela (2017) learnt hierarchical representations of symbolic data by embedding them into an n-dimensional Poincaré ball by leveraging the distance property of hyperbolic spaces. Instead of relying on the true hierarchy to learn embeddings, Nickel and Kiela (2018) inferred hierarchies from real-valued similarity scores using the Lorentz model of hyperbolic geometry. We used a similar formulation in our model HIDDEN to build our HMC model by leveraging the co-occurrence count of labels for each document, but additionally (and more importantly), in a joint manner, learn the parameters of the classifier.

3 Hyperbolic Geometry & the Poincaré Model

In this section, we give an overview of hyperbolic geometry and the Poincaré model for embedding in hyperbolic spaces (Nickel and Kiela, 2017). A hyperbolic space is a non-Euclidean Riemannian manifold of constant negative curvature. Though there are several fundamental differences between the Euclidean and the hyperbolic geometry, the most interesting characteristic of hyperbolic spaces is their ability to naturally represent hierarchical relations (Krioukov et al., 2010). In the Poincaré ball model, which is one of the standard models for hyperbolic geometry, the Euclidean distances between equidistant points, according to the inherent manifold metric d, fall exponentially as one moves from origin towards the surface of the ball. This interesting property is the key for enabling learning of continuous embeddings of hierarchies. For example, one can imagine root node of hierarchy at origin and leaf nodes near the ball's surface. Then, this model can easily accommodate exponentially growing number of equidistant siblings at deeper levels of the hierarchy. Whereas, such an accommodation is not possible using Euclidean geometry. Below we provide some details of this model.

Let $\mathcal{B}^n = \{x \in \mathbb{R}^n | \; \|x\| < 1\}$ be the *open* n-dimensional unit ball, where $\|.\|$ is the Euclidean 2 norm. The Poincaré ball model is a Riemannian Manifold $(\mathcal{B}^n, g_x)$, the open unit ball equipped with the Riemannian metric tensor $g_x = \left(\frac{2}{1-\|x\|^2}\right)^2 g^E$, where $x \in \mathcal{B}^d$ and g^E is the Euclidean metric tensor. The geodesic distance between two points $u, v \in \mathcal{B}^d$ is given as

$$d(u,v) = arcosh\left(1 + 2\frac{\|u-v\|^2}{(1-\|u\|^2)(1-\|v\|^2)}\right) \tag{1}$$

Given any $x \in \mathbb{R}^n$, one can show that $\frac{x}{1+\sqrt{1+\|x\|_2^2}}$ always lies in the Poincare ball (refer Appendix for detailed explanation).

4 Problem Formulation and Approach

In this section, we present present details of our model, training, as well as inference.

4.1 Problem Formulation

Here we consider an interesting special case of multi-label classification. The training data is of the form: $\mathcal{D} = \{(D_1, y_1), (D_2, y_2), \ldots, (D_m, y_m)\}$, where $D_i \in \mathbb{R}^n$ is the input representation of the i^{th} document, $y_i \in \{0,1\}^L$ represents the set of active/annotated labels for it ($y_i^l = 1 \iff D_i$ is labelled with l), and L is the total number of labels. Importantly, the labels are assumed to be nodes of an unknown, yet fixed, hierarchy. Using this prior knowledge and the training data, the goal is to learn a classifier that generalises well for labelling new documents.

Classical text classification methods ignore the informative prior knowledge that the set of labels form a hierarchy. Most of the hierarchical multi-class classification models assume that the hierarchy over the labels is completely known, which might not be a pragmatic assumption, since constructing hierarchies is an expensive process, especially when the number of labels is large (Bhatia et al., 2016). In contrast, here we assume no explicit information regarding the hierarchy other than it's existence, and the implicit information encoded in the training data. Also, in our set-up, we do not restrict the labels to be the leaves nodes in the hierarchy. As motivated earlier, here we propose to learn a classifier that jointly learns the classifier parameters as well as the label embeddings.

4.2 Our Model: HIDDEN

Our proposed model HIDDEN has two key components: one for representing the documents that may lead to well-generalizing classifiers and the other for embedding the labels in a hyperbolic space. Recall that hyperbolic spaces have shown to be well-suited for data satisfying hierarchical relations.

Document Model $\mathcal{F}_w$ accepts as input a document, D, and outputs a n-dimensional representation of it, $\mathcal{F}_w(D) \in \mathbb{R}^n$. Here, w is the set of parameters to be learnt. In this work, we use TextCNN (Kim, 2014) as the document model. But our approach remains valid irrespective of the chosen document model.

Label Embedding Model $\mathcal{G}_\Theta$ accepts as input a label l and outputs a finite dimensional representation, $\mathcal{G}_\Theta(l)$. Here, Θ is the set of parameters to be learnt. In this work, following Nickel and Kiela (2018), we employ the simple look-up based model defined by $\mathcal{G}_\Theta(l) \equiv \Theta * y^l = \Theta_l$, where $\Theta \in \mathbb{R}^{n \times L}$ and Θ_l is the l^{th} column of Θ. These Euclidean embeddings Θ_l are then projected onto the Poincare manifold using the transformation $\Pi(x) = \frac{x}{1+\sqrt{1+\|x\|_2^2}}$. In summary, the hyperbolic embedding of label, l, is given by $\Pi(\Theta_l)$.

We next assume that there exists some optimal set of parameters w^*, Θ^* such that the labels annotated/active for a document, D, are exactly those whose label representations are highly aligned with that of D's representation. Here, alignment between the representations is intended to model the natural intuition of appropriateness between label and document. Following the principle of large-margin separation, in this paper we employ the **alignment model** defined below:

$$\hat{y}_D^l\left(w, \Theta\right) \equiv \sigma\left(\mathcal{F}_w\left(D\right)^\top \Theta_l\right) \tag{2}$$

where $\hat{y}_D^l\left(w, \Theta\right)$ denotes the alignment between the document, D, and the l^{th} label as per the model with parameters (w, Θ), and σ is the Sigmoid activation function.

Inference: Given the learnt parameters $(\hat{w}, \hat{\Theta})$, the labels with $\hat{y}_D^l(\hat{w}, \hat{\Theta}) > 0.5$ are predicted to be the active ones for D. We next detail the proposed joint objective for learning the parameters.

4.3 Joint Objective

The proposed objective consists of two terms: the first is an empirical multi-label loss term over the training data, and the second is a loss for ensuring that the hyperbolic label embeddings respect the pairwise label co-occurrence or any other such (pairwise) partial information regarding the underlying label hierarchy.

First Term is simply a binary cross entropy loss to promote high alignment scores for each annotated label and vice-versa:

$$\mathcal{L}_1\left(w, \Theta\right) = \sum_{i=1}^{m}\sum_{l=1}^{L}\Big[y_i^l \log\left(\hat{y}_i^l\left(w, \Theta\right)\right) + \left(1 - y_i^l\right)\log\left(1 - \hat{y}_i^l\left(w, \Theta\right)\right)\Big] \tag{3}$$

where $\hat{y}_i^l$ is a short-hand for $\hat{y}_{D_i}^l$.

Second Term induces lesser geodesic distance in the hyperbolic space between the label embeddings that have higher co-occurrences than those between label pairs that have less co-occurrence (Nickel and Kiela, 2018):

$$\mathcal{L}_2(\Theta) = \sum_{\substack{l,l' \in L,\\ l' \neq l}} \log\left(\frac{e^{-d(\Pi(\Theta_l),\Pi(\Theta_{l'}))}}{\sum_{z \in \mathcal{N}(l,l')} e^{-d(\Pi(\Theta_l),\Pi(\Theta_{l'}))}}\right) \tag{4}$$

where d is the metric in the hyperbolic space given by Eq.1, $\Pi(\Theta_l)$ is hyperbolic embedding of l^{th} label, and $\mathcal{N}\left(l, l'\right)$ is the set of all labels that less frequently co-occur with l than l' co-occurs with l.

The overall objective function is a weighted sum of the two components described above

$$\mathcal{L}\left(w, \Theta\right) = \mathcal{L}_1\left(w, \Theta\right) + \lambda \mathcal{L}_2\left(\Theta\right) \tag{5}$$

We refer to the model corresponding to the parameters $(w_{\text{jnt}}, \Theta_{\text{jnt}})$ that minimizes this joint objective in Eq.5 as **HIDDEN$_{\text{jnt}}$**:

$$(w_{\text{jnt}}, \Theta_{\text{jnt}}) \in \arg\min_{w,\Theta} \mathcal{L}(w, \Theta) \tag{6}$$

Both components of our loss interact with each other to minimize the distance between document and label embeddings in the hyperbolic space. The advantage of the joint learning is well illustrated when HIDDEN$_{\text{jnt}}$ is compared with the following baseline, henceforth referred to as **HIDDEN$_{\text{cas}}$**:

(1) $\mathcal{L}_2$ is minimized to obtain label embeddings $\hat{\Theta}_{\text{cas}} \in \arg\min_\Theta \mathcal{L}_2\left(\Theta\right)$.

(2) These are then used in $\mathcal{L}_1$ to obtain document parameters: $\hat{w}_{\text{cas}} \in \arg\min_w \mathcal{L}_1(w, \hat{\Theta}_{\text{cas}})$.

We also empirically compare with the following multi-class classification baseline, henceforth referred to as **HIDDEN$_{\text{flt}}$**:

(1) Θ_{flat} is fixed to the identity matrix.

(2) These are then used in $\mathcal{L}_1$ to obtain document parameters: $\hat{w}_{\text{flat}} \in \arg\min_w \mathcal{L}_1(w, \hat{\Theta}_{\text{flat}})$.

To evaluate the benefit of using hyperbolic spaces for embedding labels, we also compare with a variant of HIDDEN$_{\text{jnt}}$ called **HIDDEN$_{\text{euc}}$** for which $\mathcal{L}_2$ is modified to be

$$\mathcal{L}_{2\text{Euc}}(\Theta) = \sum_{\substack{l,l' \in L, \\ l' \neq l}} \log \left(\frac{e^{-\|\Theta_l - \Theta_{l'}\|_2}}{\sum_{z \in \mathcal{N}(l,l')} e^{-\|\Theta_l - \Theta_{l'}\|_2}} \right) \tag{7}$$

to obtain the parameters $(w_{\text{euc}}, \Theta_{\text{euc}})$ as $(w_{\text{euc}}, \Theta_{\text{euc}}) \in \arg\min_{w,\Theta} \mathcal{L}_1(w, \Theta) + \lambda \mathcal{L}_{2\text{Euc}}(\Theta)$

Note that none of the variants of HIDDEN assume any explicit information regarding the underlying hierarchy. However, the former three exploit the prior knowledge that there exists a label hierarchy; whereas the latter, which is the classical multi-label classification network, completely ignores this useful information. Moreover, since the proposed model HIDDEN$_{\text{jnt}}$ performs joint learning, it is expected that HIDDEN$_{\text{jnt}}$ not only achieves better generalization, but also leads to better label embeddings, when compared to HIDDEN$_{\text{cas}}$. The simulation results in section 5 confirm the same.

4.4 Training Details

In all our experiments, the initial word embedding layer of TextCNN in the document model is initialized using 300 dimensional GloVe embeddings (Pennington et al., 2014). Following Nickel and Kiela (2017), we randomly initialize Θ from the uniform distribution $\mathcal{U}(-0.001, 0.001)$. Both the document and label representations are are of length $n = 300$. We randomly choose 10% of training set as the validation set and report test set results on the best validation epoch. During training, dropout is applied to the outputs of document model as well the label model with probabilities 0.1 and 0.6 respectively. We found $\lambda = 0.1$ to yield the best validation performance. The number of training epochs are set to 30 for all experiments. Both models are optimized using stochastic gradient descent using Adam optimizer (Kingma and Ba, 2014) with learning rate as 0.001 for TextCNN.

We run all our experiments on Nvidia RTX 2080 Ti GPUs 12 GB RAM over Intel Xeon Gold 5120 CPU having 56 cores and 256 GB RAM. It takes around 1, 2 and 5 hours to train the model on RCV1, NYT and Yelp datasets respectively.

5 Experiments

In this section, we compare our approach against the baseline models and other state-of-the-art HMC approaches. First we describe the evaluation metrics and illustrating results in a synthetic setting.

5.1 Evaluation Measures

Classifier Evaluation Measures: We use standard measures for evaluating any HMC, *viz.*, **Macro-F1** and **Micro-F1**. Let TP, TN, FN, FP denote the true positive, true negative, false negative and false positive labels respectively. Precision is $\frac{TP}{TP+FP}$ and recall is $\frac{TP}{TP+FN}$. F1-score is the harmonic mean of precision and recall. **Macro-F1** assigns equal weightage to each class and is computed as the averaged F1 score over all classes. **Micro-F1** is the F1-score computed over all instances.

Label Embedding Evaluation Measures: For a given application, let us say $\mathcal{H}_*$ is provided to us as the ground truth hierarchy of labels/nodes, which was assumed to be unknown in our problem formulation in Section 4.1. Recall that none of the variants of HIDDEN has access to $\mathcal{H}_*$. How consistent with respect to $\mathcal{H}_*$ are the label embeddings learnt by these models? We attempt to assess this consistency by adopting standard measures such as Spearman's rank correlation coefficient (Zar, 2005) and Normalized Discounted cumulative gain (NDCG) (Järvelin and Kekäläinen, 2002).

Recall from Section 4.3, that the hyperbolic embedding for label l is $\Pi(\Theta_l)$ and likewise for l', it is $\Pi(\Theta_{l'})$. The model parameters Θ might be learnt using any variant of HIDDEN. Given a query label, l, the geodesic distance $d(\Pi(\Theta_l), \Pi(\Theta_{l'}))$ is used to rank all other labels $l' \neq l$; smaller the distance, larger the rank. Any two labels $l' \neq l''$ that are at the same geodesic distance from l, will be assigned the same rank $r = r''$. Next, we define a graded relevance score for labels with respect to the ground truth hierarchy, $\mathcal{H}_*$. For any given query label $l \in \mathcal{H}_*$, we also assign a graded relevance $rel' \in \mathbb{N}$ to every other label $l' \neq l$ based on the distance (number of hops $hops(l, l')$) of l' from l in the hierarchy $\mathcal{H}_\mathcal{D}$; smaller the distance, larger the graded relevance (we considered $rel' \propto \frac{1}{hops(l,l')}$, for example).

Discounted Cumulative Gain (DCG) (Järvelin and Kekäläinen, 2002) is a standard measure of the quality of ranking of an approach with respect to the graded relevance provided in the ground truth. DCG@k measures items (*eg:* labels l') k hops away

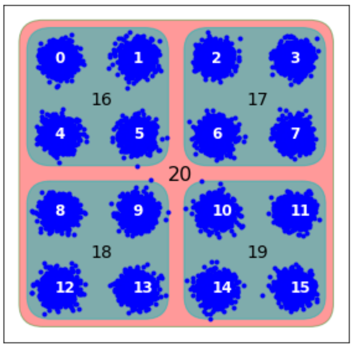


Figure 1: Gaussian used for the synthetic experiment

Prob	0.00		0.20		0.40	
	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1
HIDDEN$_{\text{flt}}$	96.8	89.1	93.2	87.8	90.4	87.7
HIDDEN$_{\text{cas}}$	98.0	93.4	94.4	88.9	91.9	91.0
HIDDEN$_{\text{jnt}}$	98.1	94.0	94.8	91.6	92.3	91.7

Table 1: Comparison of methods on the Synthetic **SETTING 1** with respect to increasing probability with which a label is randomly dropped. Synthetic data used here has 12000 training and 8000 test samples.

from the query l. The gain is accumulated from the top of the ranked list upto some pre-specified position k in the list, with the gain of each result discounted at lower ranks: $DCG_k = \sum_{i=1}^{k} \frac{rel_i}{log_2(i+1)}$. Here, rel_i is the graded relevance at position i. This result is itself averaged over all query labels $l \in \mathcal{H}_*$.

Spearman's rank correlation coefficient denoted by r is a non-parametric metric to measure statistical dependence between ranking of two variables. For each query label (l), we first measure rank correlation between the predicted rank r_p of a label l' and its own rank r_h as per ground truth hierarchy h. The correlation coefficient between r_{l_p} and r_{l_h} is computed as $r_l = \frac{cov(r_p, r_h)}{\sigma_{r_p}\ \sigma_{r_h}}$. Here, cov is the covariance and σ, the standard deviation. The final score, r is the averaged score across all labels l' in the set.

5.2 Validation via Synthetic Experiments

To observe the behaviour of the proposed approach HIDDEN with respect to the evaluation measures in a controlled environment, we present one such synthetic setup. The goal in this section is to illustrate the advantage of joint learning of parameters over isolated learning. Consider 2D data generated from 16 neatly separated Gaussians laid out on a grid as illustrated in Figure 1. Each of the 16 gaussians corresponds to a single label $l_1, l_2 ... l_{16}$. We consider a second layer of 4 labels $l_{17} ... l_{20}$, obtained by grouping the gaussians into 4; each quadrant in the larger square would correspond to a single label. Finally, we have a third layer consisting of a single label l_{21} - *viz.*, the entire large square. This simple hierarchy is hidden from our variants of HIDDEN as well as from the flat model. The synthesized data is split randomly into train-test in the ratio 60:40. For each of the jointly optimised model HIDDEN$_{\text{jnt}}$, the cascaded model HIDDEN$_{\text{cas}}$ as well as the flat model, we observe (i) the performance of the classification models $\mathcal{F}_w(D)$ measured in terms of **Micro-F1** and **Macro-F1** as well as (ii) the consistency of the label embedding models $\mathcal{G}_\Theta(l)$ with respect to the hidden 3-level hierarchy over the 21 labels. We record observations in two settings:

SETTING 1 in which *for each training instance, one of the annotated labels are dropped, uniformly at random:* In Table 1 we note the performances of the different approaches with increasing rate at which labels are dropped. The jointly optimised model HIDDEN$_{\text{jnt}}$ accounts for the classification task through loss component $\mathcal{L}_1$ as also the somewhat redundant label co-occurrence through the loss component $\mathcal{L}_2$.

As expected, we observe that the performance of HIDDEN$_{\text{jnt}}$ is more robust to this form of label noise than the HIDDEN$_{\text{cas}}$ and HIDDEN$_{\text{flt}}$models. This is because HIDDEN$_{\text{flt}}$ entirely relies on the training data and ignores the prior knowledge of existence of a label hierarchy. HIDDEN$_{\text{cas}}$ is also less robust as it over-relies on the label co-occurrence by minimising $\mathcal{L}_2$ (which, in isolation will be sensitive to label noise), before venturing into the classification task by minimising $\mathcal{L}_1$.

SETTING 2 in which the size of the *training set is decreased without corrupting labels:* We observe the performances of the different approaches with decreasing size of the training set and note that the performance of the jointly optimised model HIDDEN$_{\text{jnt}}$ falls back on the label correlation signals through the loss component $\mathcal{L}_2$ and is therefore more robust to decreasing size of the data set than the flat classifier. We observed similar results for other synthetic settings. Owing to space constraints, the plots and other ranking results are provided in the supplementary material.

5.3 Real-world Text Datasets

We used three datasets, namely, RCV1, Yelp and NYT in our experiments:

(1) **RCV1** (Lewis et al., 2004) - RCV1 is a

newswire dataset of the articles collected between 1996-1997 from Reuters

(2) **NYT** (Sandhaus, 2008) - This corpus contains articles from New York Times published between January 1st, 1987 and June 19th, 2007

(3) **Yelp** [1] - Yelp is a review dataset of restaurants and each review is labelled with hierarchical categories of restaurants. Following the experimental design in Mao et al. (2019), we use the set of reviews for a business to predict the categories to which the business belongs.

Some statistics pertaining to these datasets are presented in Table 2.

Dataset	Hierarchy	|L|	Avg(|L|)	Max(|L|)	Train	Val	Test
RCV1	Tree	104	3.24	17	20833	2314	781265
NYT	Tree	120	6.58	24	86461	9606	9903
Yelp	DAG	539	4.07	32	98460	10939	46884

Table 2: Statistics of the datasets used in the experiments. $|L|$ denotes the number of labels, Avg($|L|$) is the average number of labels per instance and Max($|L|$) is the maximum number of labels for an instance.

5.4 Comparison of models that do not use the true hierarchy

We compare performance of the different models that do not use the true hierarchy. These include our flat baseline HIDDEN$_{\text{flt}}$, the cascaded model HIDDEN$_{\text{cas}}$ as well as our joint model HIDDEN$_{\text{jnt}}$. We compare them against the baseline TextCNN-flat model reported in Mao et al. (2019). The results are presented in Table 3 for $\lambda = 0.1$.

Overall, our baseline (HIDDEN$_{\text{flt}}$) performs better than the previous baseline with the exception of Macro-F1 on NYT. We observe improvement of the joint model HIDDEN$_{\text{jnt}}$ over the flat (HIDDEN$_{\text{flt}}$) and cascaded (HIDDEN$_{\text{cas}}$) models on RCV1 and NYT (for each of which, the labels form a tree) in Table 3. However, on the Yelp dataset, the cascaded model (HIDDEN$_{\text{cas}}$) performs somewhat worse (-2 Micro-F1 and -3.3 Macro-F1) than our baseline model (HIDDEN$_{\text{flt}}$), hinting at the possibility that label co-occurrence information might not be helpful toward the classification task. This could be partly also because the labels in Yelp are structured in the form of a DAG. Constant curvature property of hyperbolic spaces makes them unsuitable for learning DAG structures (Li et al., 2018). However, the Macro-F1 performance of HIDDEN$_{\text{jnt}}$ is far better than that of the cascaded model HIDDEN$_{\text{cas}}$. This illustrates that our joint model is able to better recover from less reliable (or less useful) label co-occurrence information, just as was illustrated in the Table 1 for the synthetic setting.

Dataset	Method	Micro-F1	Macro-F1
RCV1	TextCNN-Flat*	76.6	43.0
	HIDDEN$_{\text{flt}}$	77.9	44.5
	HIDDEN$_{\text{cas}}$	78.0	45.5
	HIDDEN$_{\text{jnt}}$	**79.3**	**47.3**
NYTimes	TextCNN-Flat*	69.5	39.5
	HIDDEN$_{\text{flt}}$	76.4	37.1
	HIDDEN$_{\text{cas}}$	74.6	33.2
	HIDDEN$_{\text{jnt}}$	**77.0**	**43.6**
Yelp	TextCNN-Flat*	**62.8**	27.3
	HIDDEN$_{\text{flt}}$	62.5	**37.9**
	HIDDEN$_{\text{cas}}$	60.5	33.9
	HIDDEN$_{\text{jnt}}$	60.8	35.6

Table 3: Performance comparison on all three datasets with TextCNN as the base classification model. Recall that HIDDEN$_{\text{flt}}$ is our own multi-class classification baseline. We observe that the numbers reported by Mao et al. (2019) (indicated by *) for the TextCNN based flat baseline model are consistently outperformed by the proposed HIDDEN models.

5.5 Comparison of Hyperbolic space and Euclidean space

To assess the utility of the hyperbolic space for embedding hierarchical labels, we compare HIDDEN$_{\text{jnt}}$ and HIDDEN$_{\text{euc}}$. Table 4 presents this comparison on the three datasets. HIDDEN$_{\text{euc}}$ performs worse than HIDDEN$_{\text{jnt}}$ which uses the hyperbolic space for embedding labels (except Micro-F1 for Yelp due to reasons stated before). This is expected since embedding trees is much more effective in the hyperbolic space compared to Euclidean space since in the hyperbolic space, volume grows exponentially with distance from the origin while in Euclidean space, this growth is polynomial. The number of nodes in a tree also increases exponentially with distance from the root, making Hyperbolic spaces useful for embedding hierarchies.

5.6 Comparison with model that explicitly uses the true hierarchy

We compare performance of our joint approach HIDDEN$_{\text{jnt}}$ against a state-of-the-art hierarchical multi-label classifier, HiLAP (Mao et al., 2019). However, unlike our proposed models (variants of HIDDEN), HiLAP has access to the true hierarchy

---

[1] `https://www.yelp.com/dataset/challenge`

Dataset	Method	Micro-F1	Macro-F1
RCV1	HIDDEN$_{\text{euc}}$	78.4	47.6
	HIDDEN$_{\text{jnt}}$	**79.3**	**47.3**
NYTimes	HIDDEN$_{\text{euc}}$	76.4	40.4
	HIDDEN$_{\text{jnt}}$	**77.0**	**43.6**
Yelp	HIDDEN$_{\text{euc}}$	**61.1**	34.2
	HIDDEN$_{\text{jnt}}$	60.8	**35.6**

Table 4: Performance comparison for HIDDEN$_{\text{jnt}}$ with HIDDEN$_{\text{euc}}$. HIDDEN$_{\text{jnt}}$ consistently has better Macro-F1 better than HIDDEN$_{\text{euc}}$ and generally better Micro-F1 too. This illustrates the utility of Hyperbolic spaces for embeddding label hierarchies.

both training and inference. Thus, HiLAP serves as some form of skyline for the HIDDEN suite of approaches proposed in this paper. HiLAP learns label assignment policy using the reinforcement learning framework.

In Table 5, we compare the performance of HIDDEN$_{\text{jnt}}$ against HiLAP model as reported in Mao et al. (2019). Interestingly, on RCV1, we obtain better Micro-F1 score (+0.7) for the joint model HIDDEN$_{\text{jnt}}$ over the HiLAP method. On NYT, our Micro-F1 score is far better (+7.1) than HiLAP; our Macro-F1 score is also marginally better (+0.4) than their Macro-F1 scores. These results are interesting because HIDDEN$_{\text{jnt}}$ seems to obtain better generalisation through joint learning of the document classifier and label embeddings in a hyperbolic space, even without access to the true hierarchy. However, on Yelp, HiLAP seems to benefit over HIDDEN$_{\text{jnt}}$ by explicitly using the true hierarchy.

Dataset	HIDDEN$_{\text{jnt}}$		HiLAP	
	Micro	Macro	Micro	Macro
RCV1	**79.3**	47.3	78.6	**50.5**
NYTimes	**77.0**	**43.6**	69.9	43.2
Yelp	60.8	35.6	**65.5**	**37.3**

Table 5: Performance comparison of HIDDEN$_{\text{jnt}}$ with HiLAP with respect to Macro-F1 and Micro-F1. It it interesting to note here that though HIDDEN$_{\text{jnt}}$ does not know the true hierarchy, it performs better than HiLAP (which uses the true hierarchy) in some cases. (HiLAP numbers are those reported by Mao et al. (2019))

5.7 Evaluating performance of embeddings

We compare embeddings learned using different approaches with the ground truth hierarchy to evaluate the effectiveness of the embeddings. Figure 2 shows the plot of NDCG scores for different values of k on the RCV1 and NYTimes dataset across HIDDEN$_{\text{cas}}$ and HIDDEN$_{\text{jnt}}$ (for two different values of λ). In Table 6, we compare the Spearman rank correlation. The superior performance of HIDDEN$_{\text{jnt}}$strongly suggests that the embeddings learnt using the joint model are more representative of the true hierarchical organisation of the labels than those obtained using the flat and cascaded variants. This also goes to show that even the first term in our objective has positive contribution towards the learning of hyperbolic embeddings and indeed joint learning is beneficial.

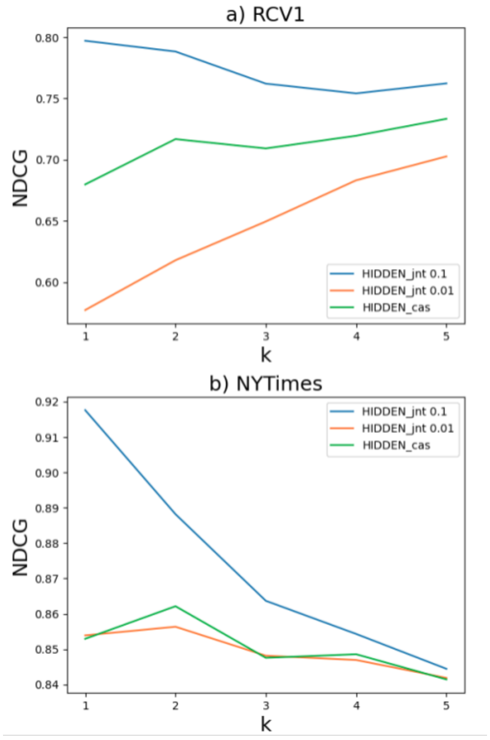


Figure 2: Plot of NDCG versus k for assessing the quality of the learnt label embeddings with respect to the actual hierarchy on RCV1 and NYT datasets. The better performance of HIDDEN$_{\text{jnt}}$ indicates the label embeddings Θ_{jnt} are most representative of the true hierarchy.

6 Conclusion

We propose a novel approach to hierarchical multi-label classification based on joint learning of document classifier and label embeddings in hyperbolic space. The proposed framework HIDDEN allows us to discover label hierarchical relationship by leveraging properties of hyperbolic geometry. Even though label-hierarchy is assumed to be unavailable, our method achieves comparable results

	HIDDEN$_{\text{flt}}$	HIDDEN$_{\text{jnt}}$	HIDDEN$_{\text{cas}}$
RCV1	21.2	**53.9**	44.1
NYTimes	11.4	**39.5**	36.1
Yelp	16.3	**31.9**	28.8

Table 6: Spearman rank correlation test for the generated embeddings for all the datasets. Each method is compared against the ground truth hierarchy. Highest values for HIDDEN$_{\text{jnt}}$ indicates the label embeddings Θ_{jnt} are most representative of the true hierarchy.

with state-of-the-art *hierarchy aware* methods. We performed extensive experiments on three datasets and demonstrate effectiveness of the learned embeddings.

Acknowledgements

We thank Bamdev Mishra and Pratik Jawanpuria (Microsoft India, Hyderabad) for valuable discussions that gave us impetus to work towards this problem. We thank anonymous reviewers for providing constructive feedback. Ayush Maheshwari is supported by a Fellowship from Ekal Foundation (www.ekal.org). We are also grateful to IBM Research, India (specifically the IBM AI Horizon Networks - IIT Bombay initiative) for their support and sponsorship.

References

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24.

Ramakrishna Bairi, Rishabh K. Iyer, Ganesh Ramakrishnan, and Jeff A. Bilmes. 2015. Summarization of multi-document topic hierarchies using submodular mixtures. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, (ACL 2015)*, pages 553–563.

Ramakrishna B. Bairi, Mark James Carman, and Ganesh Ramakrishnan. 2016. Beyond clustering: Sub-dag discovery for categorising documents. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, (CIKM 2016), 2016*, pages 801–810.

Kush Bhatia, Kunal Dahiya, Himanshu Jain, Anshul Mittal, Yashoteja Prabhu, and Manik Varma. 2016. The extreme classification repository: Multi-label datasets and code.

Ricardo Cerri, Rodrigo C Barros, and André CPLF de Carvalho. 2011. Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks. In *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 337–343. IEEE.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. 2006. Hierarchical classification: combining bayes with svm. In *Proceedings of the 23rd international conference on Machine learning*, pages 177–184.

Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. 2019. Hyperbolic interaction model for hierarchical multi-label classification. *arXiv preprint arXiv:1905.10802*.

Siddharth Gopal and Yiming Yang. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 257–265.

Mikhael Gromov. 1987. Hyperbolic groups. In *Essays in group theory*, pages 75–263. Springer.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Daphne Koller and Mehran Sahami. 1997. Hierarchically classifying documents using very few words. Technical report, Stanford InfoLab.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106.

Ashish Kulkarni, Narasimha Raju Uppalapati, Pankaj Singh, and Ganesh Ramakrishnan. 2018. An interactive multi-label consensus labeling model for multiple labeler judgments. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), 2018*, pages 1479–1486.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Xiang Li, Luke Vilnis, Dongxu Zhang, Michael Boratko, and Andrew McCallum. 2018. Smoothing the geometry of probabilistic box embeddings. In *International Conference on Learning Representations*.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124.

Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. 2019. Hierarchical text classification with reinforced label assignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 445–455.

Andrew Kachites McCallum. 1999. Multi-label text classification with a mixture model trained by em. In *AAAI 99 workshop on text learning*. Citeseer.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347.

Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pages 3779–3788.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, pages 1063–1072.

Shengwen Peng, Ronghui You, Hongning Wang, Chengxiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2016. Deepmesh: deep semantic representation for improving large-scale mesh indexing. *Bioinformatics*, 32(12):i70–i79.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7(Jul):1601–1626.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Carlos N Silla Jr and Alex A Freitas. 2009. A global-model naive bayes approach to the hierarchical prediction of protein functions. In *2009 Ninth IEEE International Conference on Data Mining*, pages 992–997. IEEE.

Isaac Triguero and Celine Vens. 2016. Labelling strategies for hierarchical multi-label classification techniques. *Pattern Recognition*, 56:170–183.

Ke Wang, Senqiang Zhou, and Yu He. 2001. Hierarchical classification of real life documents. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–16. SIAM.

Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. 2018. Hierarchical multi-label classification networks. In *International Conference on Machine Learning*, pages 5075–5084.

Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. 2009. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 917–926.

Jerrold H Zar. 2005. Spearman rank correlation. *Encyclopedia of Biostatistics*, 7.

Appendix

1 Explanation for $\Pi(x)$

The Lorentz model is defined as the Riemannian Manifold, $\mathcal{L}^n = (\mathcal{H}^n, g_l)$, where $\mathcal{H}^n = \{\mathbf{x} \in \mathbb{R}^{n+1} : \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = -1, \mathbf{x}_0 > 0\}$, and $g_l = diag([-1\ 1\ \dots\ 1])$. Here, $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}$, known as the Minkowski inner-product, is given by

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} = -x_0 y_0 + x_1 y_1 + \ldots + x_n y_n$$

The Poincaré model and the Lorentz model are equivalent in isometry. Therefore points in Lorentz manifold can be mapped into Poincaré ball as, $p : \mathcal{H}^n \to \mathcal{P}^n$

$$p(x_0, x_1, \ldots, x_n) = \frac{(x_1, \ldots, x_n)}{x_0 + 1}$$

A point $\mathbf{x}$ in the Euclidean space $\mathbb{R}^n$ can be projected onto the Lorentz manifold $\mathcal{H}^n$ using the transformation $\Omega(\mathbf{x}) = \left[\sqrt{1 + \|\mathbf{x}\|^2}, \mathbf{x}\right]$. This transformation ensures that the Minkowski inner-product, $\langle \Omega(\mathbf{x}), \Omega(\mathbf{x}) \rangle_{\mathcal{L}}$, is equal to -1, and that the first component, $\Omega(\mathbf{x})_0 \equiv \sqrt{1 + \|\mathbf{x}\|^2}$ is positive, as required for membership in the Lorentz manifold.

Now using the isometry between Poincaré and Lorentz models (Nickel and Kiela, 2018), we have $\Pi : \mathbb{R}^n \to \mathcal{P}^n$ as

$$\Pi(\mathbf{x}) = p\left(\Omega(\mathbf{x})\right) = \frac{\mathbf{x}}{1 + \sqrt{1 + \|\mathbf{x}\|^2}}$$

2 Dataset Details

We describe the details of the datasets used in our experiments.

For RCV1 dataset (Lewis et al., 2004), we use the original training/test split and use 10% of the training set as the validation set. We introduce an extra *Root* label in addition to the 103 labels present in the dataset. Each document in the dataset is labelled with this label.

The details for the other datasets used are same as in Mao et al. (2019) and we refer the readers to the same.

3 Remarks on Synthetic Experiments

In **SETTING 1**, with increasing probability of a label being randomly dropped, we observe in Figure 3 that the performance of all the models decreases which is expected. However, it is interesting to note that HIDDEN$_{\text{jnt}}$ is more robust to the noisy labels and always performs better than HIDDEN$_{\text{flt}}$ since it has an additional source of information about the labels via label co-occurrences. This information is also implicitly available to HIDDEN$_{\text{flt}}$ but from our experiments, we observe that providing this information explicitly improves performance. We also observe that performance of HIDDEN$_{\text{cas}}$ fluctuates quite a bit which is probably due to the fact that it overly relies on label co-occurrence and commits to a set of label embedding before trying to solve the classification task.

In **SETTING 2**, with increasing number of points in the training set, we observed in Figure 4 that with very small datasets, HIDDEN$_{\text{cas}}$ performs slightly better than HIDDEN$_{\text{jnt}}$ but with increasing dataset sizes, HIDDEN$_{\text{jnt}}$ performs better. Since there is no label noise in this setting, the label co-occurances are expected to be quite meaningful and with small dataset sizes, jointly learning both the document embedding model $\mathcal{F}_w$ and the label embeddings Θ jointly (HIDDEN$_{\text{jnt}}$) is more difficult compared learning Θ first and then $\mathcal{F}_w$ (HIDDEN$_{\text{cas}}$).

HIDDEN$_{\text{jnt}}$ and HIDDEN$_{\text{cas}}$ perform better than HIDDEN$_{\text{flt}}$ since the former can fall back of the label correlation signal via $\mathcal{L}_2$ which is available in a much more easily usable format than to HIDDEN$_{\text{flt}}$. As discussed above, having the label co-occurrences available explicitly to the models help achieve better performance.

We observe similar results in other synthetic settings.

4 Hierarchy of the Synthetic data

As described in the paper, the synthetic data is generated from 16 Bivariate Gaussian distributions with their means placed evenly in a 4×4 grid. These Gaussians are then grouped at various levels to get an hierarchy with 3 level as shown in Figure 5.

5 Contrasting label hierarchies across datasets

Recall, how in Table 3 of the paper, we observe improvement of the joint model HIDDEN$_{\text{jnt}}$ over the flat (HIDDEN$_{\text{flt}}$) and cascaded (HIDDEN$_{\text{cas}}$) models on RCV1 and NYTimes datasets. Note that the labels of RCV1 as well as of NYTimes form trees. However, on the Yelp dataset, the cascaded model (HIDDEN$_{\text{cas}}$) performs somewhat worse (-2 on micro and -3.3 on macro) than our baseline

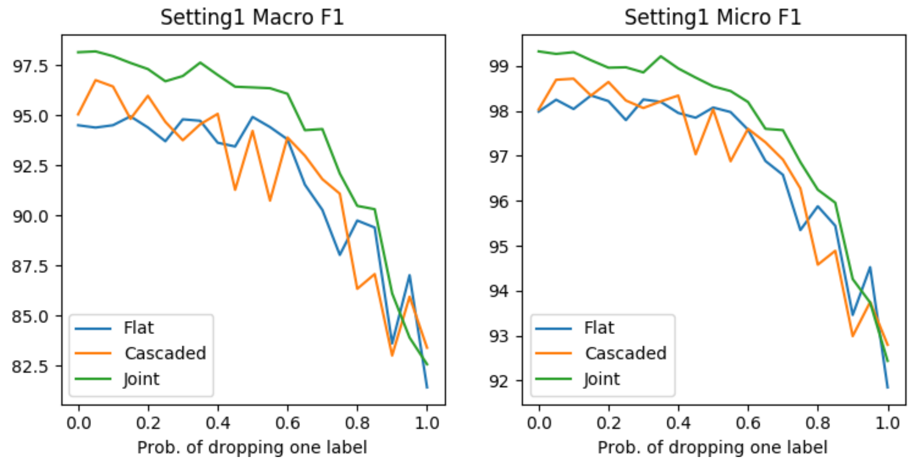


Figure 3: Micro and Macro F1 scores with increasing probability of randomly dropping one label for all three model. Scores are averaged over 10 runs

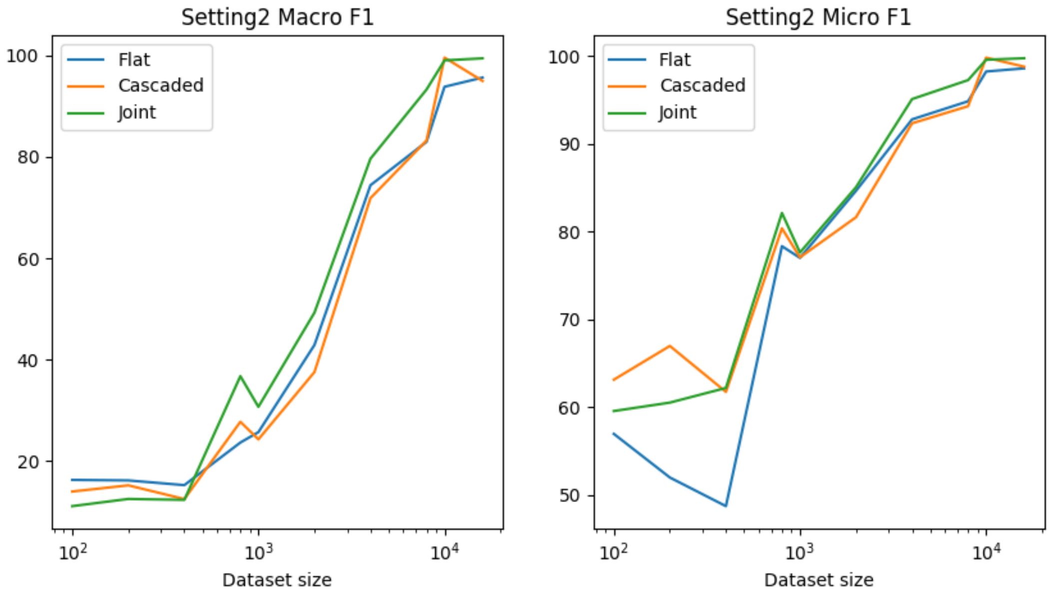


Figure 4: F1 scores with increasing dataset sizes (number of points in train set) for all three model

model (HIDDEN$_{\text{flt}}$), hinting at the possibility that label co-occurrence information might not be helpful toward the classification task. This could be partly also because the labels in Yelp are structured in the form of a DAG with 12 labels in the labels set having more than one parent.

Dataset	RCV1	NYT	Yelp
Hyperbolicity	0	1	1

Table 7: Hyperbolicity (Gromov, 1987) of the label hierarchies for the datasets used

5.1 δ-hyperbolicity

To further investigate why our method fails to perform well on Yelp dataset, we compute hyperbolicity (Gromov, 1987) for each of the label hierarchies. The hyperbolicity δ of a graph G is a measure of how tree-like the graph is. Lower the δ, the more tree-like is the graph. Hyperbolicity δ is 0 for trees.

As shown in Table 7, RCV1 has 0 hyperbolicity as expected, since the label hierarchy is a tree. We would have expected NYT to also have a hyperbolicity of 0 but the label `Others` appears at different levels of the hierarchy making it a DAG. However, this is the only deviation from being a tree and thus our method is able to perform well on NYT. For Yelp, the hyperbolicity is 1 and there are multiple labels with more than one parent and thus it is less tree-like than the NYT hierarchy. It

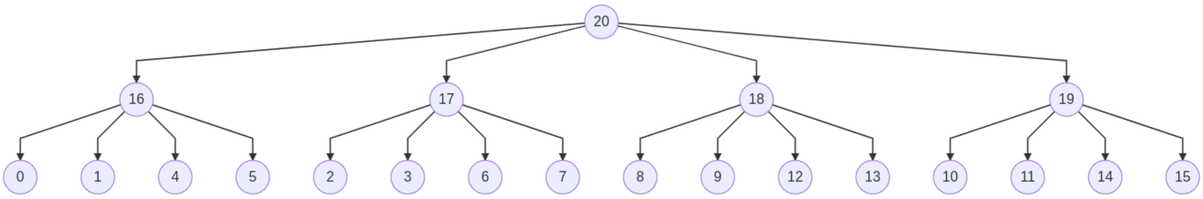


Figure 5: Hidden Hierarchy in the Synthetic experiments

is due to this DAG-ness that our method does not perform as well on Yelp as the other datasets. Note that this a limitation of hyperbolic spaces and using other geometric spaces which are conducive to embeddings DAGs, our method is expected to perform well even when the implicity label hierarchy is a DAG. In summary, hyperbolic spaces are good for embedding low δ-hyperbolicity graphs and thus our methods performance better with graphs having low hyperbolicity.

Segmenting Subtitles for Correcting ASR Segmentation Errors

David Wan[1], **Chris Kedzie**[1], **Faisal Ladhak**[1], **Elsbeth Turcan**[1],
Petra Galuščáková[2], **Elena Zotkina**[2], **Zhengping Jiang**[1],
Peter Bell [3], **Kathleen McKeown**[1]
[1] Columbia University, [2] University of Maryland, [3] University of Edinburgh
`{dw2735, zj2265}@columbia.edu`, `{petra, elena}@umiacs.umd.edu`,
`{kedzie, faisal, eturcan, kathy}@cs.columbia.edu`,
`peter.bell@ed.ac.uk`

Abstract

Typical ASR systems segment the input audio into *utterances* using purely acoustic information, which may not resemble the sentence-like units that are expected by conventional machine translation (MT) systems for Spoken Language Translation. In this work, we propose a model for correcting the acoustic segmentation of ASR models for low-resource languages to improve performance on downstream tasks. We propose the use of subtitles as a proxy dataset for correcting ASR acoustic segmentation, creating synthetic acoustic utterances by modeling common error modes. We train a neural tagging model for correcting ASR acoustic segmentation and show that it improves downstream performance on MT and audio-document cross-language information retrieval (CLIR).

1 Introduction

Typical ASR systems segment the input audio into *utterances* using purely acoustic information, i.e., pauses in speaking or other dips in the audio signal, which may not resemble the sentence-like units that are expected by conventional MT systems for spoken language translation (SLT) (Cho et al., 2017). Longer utterances may span multiple sentences, while shorter utterances may be sentence fragments containing only a few words (see Figure 1 for examples). Both can be problematic for downstream MT systems. In this work, we propose a model for correcting the acoustic segmentation of an ASR model to improve performance on downstream tasks, focusing on the challenges inherent to SLT pipelines for low-resource languages.

While prior work has trained intermediate components to segment ASR output into sentence-like units (Matusov et al., 2007; Rao et al., 2007), these have primarily focused on highly resourced language pairs such as Arabic and Chinese. When the source language is low-resource, suitable training data may be very limited for ASR and MT, and even nonexistent for segmentation. Since typical low-resource language ASR datasets crawled from the web do not have hand-annotated segments, we propose deriving proxy segmentation datasets from film and television subtitles. Subtitles typically contain segment boundary information like sentence-final punctuation, and while they are not exact transcriptions, they are closer to transcribed speech than many other large text corpora.

Our proposed model takes as input a sequence of tokens and segmentation boundaries produced by the acoustic segmentation of the ASR system and returns a corrected segmentation. While subtitles are often similar to speech transcripts, they lack an existing acoustic segmentation for our model to correct. To account for this, we generate synthetic acoustic segmentation by explicitly modeling two common error modes of ASR acoustic segmentation: under- and over-segmentation.

We evaluate the downstream MT performance in a larger SLT pipeline, and show improvements in translation quality when using our segmentation model to correct the acoustic segmentation provided by ASR. We also extrinsically evaluate our improved SLT pipeline as part of a document-level cross-lingual information retrieval (CLIR) task, where we show that improvements in ASR segmentation also lead to improved relevance of search results. We report results for nine translation settings: Bulgarian (BG) to English, Lithuanian (LT) to English, and Farsi (FA) to English, and when using either phrase-based, statistical MT (SMT) or one of two neural MT (NMT) models. We finally perform an ablation study to examine the effects of our synthetic acoustic boundaries and our over- and under-segmentation noise.

This paper makes the following contributions. *(i)* We propose the use of subtitles as a proxy dataset for correcting ASR acoustic segmentation and *(ii)* a method for adding synthetic acoustic utterance segmentations to a subtitle dataset, as well as *(iii)* a

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2842–2854
April 19 - 23, 2021.

Acoustic Segmentation (Over-segmentation):

ARE YOU OKAY ■ AGENT SCULLY ■ YOU KIND OF SOUNDED A ■ LITTLE SPOOKY ■

Corrected Sentence Segmentation:

ARE YOU OKAY AGENT SCULLY ■ YOU KIND OF SOUNDED A LITTLE SPOOKY ■

Acoustic Segmentation (Under-segmentation):

NO IS HE IN SOME KIND OF TROUBLE ■

Corrected Sentence Segmentation:

NO ■ IS HE IN SOME KIND OF TROUBLE ■

Figure 1: Example acoustic segmentation errors and their corrections. ■ indicates a segment boundary.

simple neural tagging model for correcting ASR acoustic segmentation before use in an MT pipeline. *(iv)* Finally, we show downstream performance increases on MT and document-level CLIR tasks, especially for more syntactically complex segments.

2 Related Work

Segmentation in SLT has been studied quite extensively in high-resource settings. Early work used kernel-based SVM models to predict sentence boundaries using language model probabilities along with prosodic features such as pause duration (Matusov et al., 2007; Rao et al., 2007) and part-of-speech features derived from a fixed window size (Rangarajan Sridhar et al., 2013). Other work has modeled the problem using hidden markov models (Shriberg et al., 2000; Gotoh and Renals, 2000; Christensen et al., 2001; Kim and Woodland, 2001) and conditional random fields (Liu et al., 2005; Lu and Ng, 2010).

More recent segmentation work uses neural architectures, such as LSTM (Sperber et al., 2018) and Transformer models (Pham et al., 2019). These models benefit from the large training data available for high-resource languages. For example, the TED corpus (Cettolo et al., 2012) for SLT from English to German includes about 340 hours of well-transcribed data. To our knowledge, such datasets do not exist for the languages we are interested in. Wan et al. (2020) develop a segmentation model in our setting using subtitles; however, they do not take into account explicit modeling of segmentation errors and show only minimal and intermittent improvements in downstream tasks.

Recent work has increasingly focused on end-to-end models of SLT in a high-resource setting, since these systems reduce error propagation and latency when compared to cascaded approaches (Weiss et al., 2017; Cross Vila et al., 2018; Sperber et al., 2019; Gaido et al., 2020; Bahar et al., 2020; Lakumarapu et al., 2020). In spite of these advantages, end-to-end systems have only very recently achieved competitive results due to the limited amount of parallel data for speech translation as compared to the data that is available to train ASR systems and translation systems separately (Gaido et al., 2020; Ansari et al., 2020).

3 Problem Definition

We treat the ASR acoustic segmentation problem as a sequence tagging problem (Stolcke and Shriberg, 1996). Unlike a typical tagging problem, which aims to tag a single input sequence, our input is a pair of aligned sequences of n items, $\mathbf{x} = [x_1, \ldots, x_n]$ and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_n]$ where $\mathbf{x}$ and $\boldsymbol{\gamma}$ are the ASR tokens and acoustic segmentation respectively. The tokens x_i belong to a finite vocabulary $\mathcal{V}$, while the acoustic segmentation boundary tags are binary, i.e., $\gamma_i \in \{0, 1\}$, where $\gamma_i = 1$ indicates that the ASR acoustic segmentation placed a boundary between tokens x_i and x_{i+1}. The goal is to predict a corrected segment boundary tag sequence $\mathbf{y} = [y_1, \ldots, y_n] \in \{0, 1\}^n$ from $\mathbf{x}$ and $\boldsymbol{\gamma}$.

We do this by learning a probabilistic mapping from token/segmentation sequences to corrected segmentation $p(\cdot|\mathbf{x}, \boldsymbol{\gamma}; \theta) : \{0, 1\}^n \rightarrow (0, 1)$ where p is a neural tagging model with parameters θ. While $\boldsymbol{\gamma}$ are produced solely from acoustic cues, p can take advantage of both the acoustic information (via $\boldsymbol{\gamma}$) as well as syntactic/semantic cues implicit in $\mathbf{x}$.

Reference Segmentation:

YEAH ■ THE HOLIDAY MARKET IS TOO BUSY ■ YES ■

Synthetic Acoustic Segmentation Generation:

Model Output:	$\mathbf{y}$	=	1	0	0	0	0	0	1	1
(Under-seg.)	$\breve{\gamma}$	=	1	–	–	–	–	–	**0**	1
(Over-seg.)	$\hat{\gamma}$	=	–	0	0	**1**	0	**1**	–	–
Model Input:	γ	=	1	0	0	1	0	1	0	1
	$\mathbf{x}$	=	$YEAH$	THE	$HOLIDAY$	$MARKET$	IS	TOO	$BUSY$	YES

Synthetic Acoustic Segmentation (γ):

YEAH ■ THE HOLIDAY MARKET ■ IS TOO ■ BUSY YES ■

Figure 2: Example of synthetic acoustic segmentation (γ) creation. For each training datapoint, we have as model input the tokens $\mathbf{x}$ and the corresponding model output sentence boundary labels $\mathbf{y}$. To generate the synthetic acoustic segmentation (γ), we apply under-segmentation ($\breve{\gamma}$) and over-segmentation ($\hat{\gamma}$) noise to $\mathbf{y}$. Dashes indicate the tokens where the particular noise is not applicable. Bold indicates the changed labels due to the noise. We generate the additional input γ by combining both $\breve{\gamma}$ and $\hat{\gamma}$.

4 Generating Training Data from Subtitles

One of our primary contributions is a method for converting subtitle data into suitable training data for an ASR segmentation correction model. The subtitle data contains speech-like utterances of dialogue between characters in film and television shows. For the purposes of this paper, we do not use information about speaker identity, only the text and information about segmentation. We obtain the ground truth output label segmentation $\mathbf{y}$ by segmenting the subtitle text on sentence final punctuation.[1] We remove the punctuation but keep the implied label sequence to obtain the input token sequence $\mathbf{x}$ and ground truth output label segmentation $\mathbf{y}$. However, we do not have acoustic segmentation available for $\mathbf{x}, \mathbf{y}$ pairs derived from subtitle data, which we will need as additional input if our model is to learn to correct acoustic segmentation provided by an ASR component. We thus create a synthetic acoustic segmentation sequence γ as input by adding two types of noise to $\mathbf{y}$. Specifically, we imitate two common ASR system errors, under-segmentation noise and over-segmentation, so that at test time the model can correct those errors.

Under-segmentation Noise In the ASR model, under-segmentation occurs when pauses between words are brief, and the resulting ASR output is an utterance that could ideally be split into multiple sentence-like segments. We simulate this by adding *under-segmentation noise* which converts ground truth segmentation boundaries $y_i = 1$, to $y_i = 0$ with probability $\breve{\alpha}$ and leaves $y_i = 0$ unchanged.

Over-segmentation Noise Over-segmentation occurs in an ASR model when a speaker takes a longer pause in the middle of what could be interpreted as a contiguous sentence-like utterance. *Over-segmentation noise* is simulated by inserting random segment boundaries within an utterance. That is, with probability $\hat{\alpha}$ we convert a non-boundary tag $y_i = 0$ to $y_i = 1$, while leaving all $y_i = 1$ unchanged.

Synthetic Segmentation Input Generation We can then sample a synthetic acoustic segmentation sequence γ from the following distribution,

$$\gamma_i \sim \begin{cases} \text{Bernoulli}(\breve{\alpha}) & \text{if } y_i = 1 \\ \text{Bernoulli}(\hat{\alpha}) & \text{otherwise} \end{cases}$$

for $i \in \{1, \ldots, n\}$. This can be thought of as dropout applied to the correct label sequence $\mathbf{y}$. See Figure 2 for an example. Our proposed segmentation correction model will learn to denoise the input segmentation sequence γ and produce the corrected sequence $\mathbf{y}$.

5 Model

We employ a Long Short-Term Memory (LSTM)-based model architecture for this task (Hochreiter and Schmidhuber, 1997). Given an input sequence of ASR tokens $\mathbf{x} = [x_1, \ldots, x_n]$ along with corresponding ASR segmentation sequence

---

[1] Set of sentence final punctuation: { **( ) : - ! ? .** }.

$\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_n]$, we first get an embedding representation $\mathbf{e}_i \in \mathbb{R}^{316}$ for each token as follows:

$$\mathbf{e}_i = G(x_i) \oplus F(\gamma_i)$$

where $G \in \mathbb{R}^{|\mathcal{V}| \times 300}$ and $F \in \mathbb{R}^{2 \times 16}$ are embedding lookup tables, and $\oplus$ is the concatenation operator. We initialized G with FastText embeddings pre-trained on Common Crawl data (Mikolov et al., 2018). F is randomly initialized.

We pass the embedding sequence through a two-layer bi-directional LSTM, with 512 hidden units each, to get the contextual representation $\mathbf{h}_i \in \mathbb{R}^{1024}$ for each token as follows:

$$\mathbf{h}_i = \overrightarrow{\mathrm{LSTM}}(\mathbf{e}_i) \oplus \overleftarrow{\mathrm{LSTM}}(\mathbf{e}_i)$$

where $\overrightarrow{\mathrm{LSTM}}$ and $\overleftarrow{\mathrm{LSTM}}$ are the forward direction and backward direction LSTMs respectively.

Each output state $\mathbf{h}_i$ is then passed through a linear projection layer with a logistic sigmoid to compute the probability of a segment boundary $p(y_i = 1 | \mathbf{h}_i; \theta)$. The log-likelihood of a corrected segmentation boundary sequence is $\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\gamma}; \theta) = \sum_{i=1}^{n} \log p(y_i|\mathbf{h}_i; \theta)$. We fit the parameters, θ, by approximately minimizing the negative log-likelihood on the training set $\mathcal{D}$, $\mathcal{L}(\theta) = -\frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}, \boldsymbol{\gamma}, \mathbf{y}) \in \mathcal{D}} \log p(\mathbf{y}|\boldsymbol{\gamma}, \mathbf{x}; \theta)$, using mini-batch stochastic gradient descent.

6 Datasets

6.1 Subtitles Dataset

We obtain monolingual subtitle data from the OpenSubtitles 2018 corpus (Lison and Tiedemann, 2016). OpenSubtitles contains monolingual subtitles for 62 languages drawn from movies and television. The number of subtitle documents varies considerably from language to language. LT has only 1,976 documents, while BG and FA have 107,923 and 12,185 respectively. We randomly down-sample from the larger collection to 2,000 documents to ensure our segmentation correction models are all trained with similar amounts of data.

Treating the subtitles for a complete television episode or movie as the source of a single training instance $(\mathbf{x}, \boldsymbol{\gamma}, \mathbf{y})$ introduces some complications because they are usually quite long relative to typical SLT system input. To better match our evaluation conditions, we arbitrarily split each document into M instances, where the length l in tokens for each instance m is sampled from $L \sim U(1, 100)$, i.e. uniformly from 1 to 100 tokens. This range was determined to to approximate the length distribution of our evaluation datasets.

See Table 1 for statistics on the number of training instances created as well as the average number of sentence segments per instance. Note that even though the number of subtitle documents is close to equal, the documents can vary considerably in length. As result, the BG dataset has more than twice the training instances of FA or LT. In some cases, an instance may contain only a few words that do not constitute a sentence, and such instances would have no segment boundaries; this helps prevent the model from learning pathological solutions such as always inserting a segment boundary at the end of the sequence.

Since we do not evaluate the segmentation directly on OpenSubtitles, we split the available data into training and development partitions, with 90% of the instances in the training set.

6.2 Speech Retrieval Dataset

For extrinsic evaluation of ASR segments, we use the speech retrieval dataset from the MATERIAL[2] program. The goal of MATERIAL is to develop systems that can retrieve text and speech documents in low-resource languages that are relevant to a given query in English. To bootstrap speech retrieval systems in low-resource languages, MATERIAL collects BG, FA, and LT speech training data for ASR systems, as well as additional separate collections of BG, FA, and LT speech documents along with their relevance judgments for a set of English language queries. Since the retrieval of speech documents requires a cascade of ASR, MT, and CLIR systems, the MATERIAL data allows us to measure the impact of ASR segmentation on both the translation quality, as well as the downstream retrieval system. The data partitions in MATERIAL are numerous and to avoid confusion, we briefly describe them here.

The *BUILD* partition contains a small amount of ASR training and development data for BG, FA, and LT, i.e. audio files paired with reference transcripts. We use the BUILD data for fine-tuning our subtitle trained model. We apply the same synthetic acoustic segmentation generation procedure to this collection as we do to the subtitle data when using it for fine-tuning. See Table 1 for dataset statistics.

The *Test (Small)* partition contains audio docu-

---

[2] `www.iarpa.gov/index.php/research-programs/material`

Lang.	OpenSubtitles			BUILD Train			BUILD Valid		
	D	I	S	D	I	S	D	I	S
BG	2000	459,301	3.95	352	7,723	3.22	108	2,472	3.14
FA	2000	120,039	3.88	302	6,707	3.52	120	2,679	3.53
LT	1977	165,751	12.99	484	11,782	3.40	112	2,893	3.31

Table 1: Segmentation model training dataset statistics. We report number of documents (D), number of instances (I) in each dataset and average number of segments per instance (S).

Lang.	Test (Small)		Test (Large)	
	Q	D	Q	D
BG	300	634	–	–
FA	221	528	–	–
LT	300	496	1,000	3,297

Table 2: Number of queries (Q) and documents (D) in the speech retrieval test collections.

ments and a set of English language queries and relevance judgements for those queries. At test time, we use the acoustic segmentation provided by the ASR system as the input γ instead of generating acoustic label sequences. Additionally, roughly half of the audio documents in this collection include ground-truth transcriptions and translations to English, which allows us to evaluate MT.

The *Test (Large)* partition is similar to the Test (Small) partition, but much bigger in size. There are no transcripts or translations, so it can be used only to evaluate CLIR. The Test (Large) partition is available only for LT.

We use the translated portion of Test (Small) as a test set for MT and both Test (Small) and Test (Large) as extrinsic test sets for CLIR. The statistics of the MATERIAL partitions can be found in Table 2.[3]

The speech retrieval datasets come from three domains: news broadcast, topical broadcast such as podcasts, and conversational speech from multiple low-resource languages. Some speech documents have two speakers, with each speaker on a separate channel, i.e., completely isolated from the other speaker. When performing segmentation we treat each channel independently, creating a separate (re-segmented) ASR output for each channel. To create the document transcript for MT, we merge the two output sequences by sorting the token segments based on their wall-clock start time.

7 Experiments

7.1 Segmentation Model Training

For all datasets, we tokenize all data with Moses (Koehn et al., 2007). To improve performance on out of vocabulary words, we use Byte-Pair-Encoding (Sennrich et al., 2016) with 32,000 merge operations to create subwords for each language.

We then train the segmentation model on the subtitle dataset. When creating γ sequences on the subtitles data, we set under- and over-segmentation noise to $\breve{\alpha} = 0.25$ and $\hat{\alpha} = 0.25$ respectively.[4] We use the Adam optimizer (Kingma and Ba, 2015) with learning rate of 0.001. We use early stopping on the validation loss of the OpenSubtitles validation set to select the best stopping epoch for the segmentation model.

We further fine-tune this model on the BUILD partition to expose the model to some in-domain training data. The data is similarly prepared as OpenSubtitles. We use early stopping on the development loss of this partition.

7.2 ASR-Segmentation-MT-CLIR Pipeline

We evaluate our segmentation correction model in the context of a CLIR pipeline for retrieving audio documents in BG, FA, or LT that are relevant to English queries. We refer to the three languages BG, FA, and LT as *source languages*. This pipeline uses ASR to convert source language audio documents to source language text transcripts, and MT to translate the source language transcripts into English transcripts. Then a monolingual English IR component is used to return source language documents that are relevant to the issued English queries. We insert our segmentation correction model into

---

[3] The official MATERIAL collections are named ANALYSIS+DEV and EVAL, but we refer to them as Test (Small) and Test (Large) to avoid confusion.

[4] Values for $\breve{\alpha}$ and $\hat{\alpha}$ were determined by grid-search over $\{0.25, 0.5, 0.75\}$ that minimized loss on the BUILD Valid data.

this pipeline between the ASR and MT components, i.e. *(i)* ASR to *(ii)* Segmentation Correction, to *(iii)* MT to *(iv)* IR. For clarity we reiterate, the segmentation model takes as input a source language transcript and returns the source language transcript with corrected segmentation.

To implement the ASR, MT, and IR components, we use implementations developed by MATERIAL program participants (Oard et al., 2019).

7.2.1 ASR System

We use the ASR systems developed jointly by the University of Cambridge and the University of Edinburgh (Ragni and Gales, 2018; Carmantini et al., 2019). The ASR system uses a neural network based acoustic model, trained in a semi-supervised manner on web-scraped audio data, to overcome the small amount of training data in the BUILD data. Separate models are trained for narrow-band audio (i.e., conversational speech) and wide-band audio (i.e. news and topical broadcast).

7.2.2 Segmentation Correction

At test time, given a speech document, the ASR system produces a series of acoustically derived utterances, i.e. $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$, from this input. In our setting, the corresponding acoustic label sequence $\boldsymbol{\gamma}^{(i)}$ for each utterance would be zero everywhere except the final position, i.e. $\boldsymbol{\gamma}^{(i)} = [0, 0, \ldots, 0, 1]$. If we were to process each utterance, $(\mathbf{x}^{(i)}, \boldsymbol{\gamma}^{(i)})$, individually, the model may not have enough context to correct under-segmentation at the ends of the utterance. For example, when correcting the final token position, which by definition will precede a long audio pause, the model will only see the left-hand side of the context. To avoid this, we run our segmentation correction model on consecutive pairs of ASR output utterances, i.e. $\left(\mathbf{x}^{(i)} \oplus \mathbf{x}^{(i+1)}, \boldsymbol{\gamma}^{(i)} \oplus \boldsymbol{\gamma}^{(i+1)}\right)$. Under this formulation each ASR output utterance is corrected twice (except for the first and last utterances which are only corrected once), therefore we have two predictions $\hat{y}_j^{(i,L)}$ and $\hat{y}_j^{(i,R)}$ for the j-th segment boundary. We resolve these with the logical-OR operation to obtain the final segmentation correction, i.e. $\hat{y}_j^{(i)} = \hat{y}_j^{(i,L)} \vee \hat{y}_j^{(i,R)}$.

Based on the segmentation corrections produced by our model, we re-segment the ASR output tokens and hand the resulting segments off to the MT component where they are individually translated.

7.2.3 MT Systems

We evaluate with three different MT systems. We use the neural MT model developed by the University of Edinburgh (EDI-NMT) and the neural and phrase-based statistical MT systems from the University of Maryland (UMD-NMT and UMD-SMT, respectively). The EDI-NMT and UMD-NMT systems are Transformer-based models (Vaswani et al., 2017) trained using the Marian Toolkit (Junczys-Dowmunt et al., 2018) and Sockeye (Hieber et al., 2018), respectively. UMD-NMT trains a single model for both directions of a language pair (Niu et al., 2018), while EDI-NMT has a separate model for each direction. UMD-SMT is trained using the Moses SMT Toolkit (Koehn et al., 2003), where the weights were optimized using MERT (Och, 2003).

7.2.4 IR System

For the IR system, we use the bag-of-words language model implemented in Indri (Strohman et al., 2005). Documents and queries are both tokenized and normalized on the character level to avoid potential mismatch in the vocabulary. The queries are relatively short, typically consisting of only a few words, and they define two types of relevancy – the conceptual queries require the relevant documents to be topically relevant to the query, while the simple queries require the relevant document to contain the translation of the query. However, no specific processing is used for these two relevance types in our experiments.

7.3 MT Evaluation

Our first extrinsic evaluation measures the BLEU (Papineni et al., 2002) score of the MT output on the Test (Small) sets after running our segmentation correction model, where we have ground truth reference English translations. We refer to our model trained only on the BUILD data as *Seg*, and our subtitle-trained model as *Seg + Sub*. As our baseline, we compare the same pipeline using the segmentation produced by the acoustic model of the ASR system, denoted *Acous*.

Since each segmentation model produces segments with different boundaries, we are unable to use BLEU directly to compare to the reference sentences. Therefore, we concatenate all segments of a document and treat them as one segment, which we refer to as “document-level” BLEU score. We use SacreBLEU[5] (Post, 2018) with the lowercase

---

[5] `https://github.com/mjpost/sacrebleu`

option due to the different casing for the reference English translation and MT output.

We also provide BLEU scores for the MT output using the reference transcriptions (Ref) to show the maximum score the system can achieve when there is no ASR or segmentation error. This represents the theoretical upper bound for our pipeline with a perfect ASR system.

Segmentation errors (i.e., the acoustic model incorrectly segmented an utterance) and word errors (i.e., the ASR system produces an incorrect word) can both affect the downstream MT performance. To isolate the segmentation errors from word errors, we align the ASR output tokens to the reference transcriptions by timecode in order to obtain a reference system that has no segmentation errors, but does have transcription errors. This represents a more realistic ceiling for our model because while we can correct segmentation, we cannot correct word errors. We refer to this system in the results section as Align.

7.4 Document-Level CLIR Evaluation

Our second extrinsic evaluation is done on the MATERIAL CLIR task. We are given English queries and asked to retrieve audio documents in either BG, FA, or LT. In our setup, we only search over the English translations of the segmented transcripts produced by our pipeline, i.e., we do not translate the English query into the other languages or search the audio signal directly. We evaluate the performance of CLIR using the Maximum Query Weighted Value (MQWV) from the ground-truth query-relevance judgements for documents in the Test (Small & Large) collections. MQWV, which is a variant of the official MATERIAL program metric called Actual Query Weighted Value (NIST, 2017, AQWV), is a recall-oriented rank metric that measures how well we order the retrieval collection with respect to query relevance.

AQWV is calculated as the average of $1-(P_m+\beta*P_{fa})$ for each query, where P_m is the probability of misses, P_{fa} is the probability of false alarms, and β is a hyperparameter. The maximum possible value is 1 and the minimum value is given by $-\beta$. In our experiments β it is set to 40. AQWV thus not only depends on the ranking of the documents but also on β.

Additionally, AQWV is sensitive to the threshold used by the IR system to determine document relevance. To avoid the tuning of thresholds, we report MQWV which is calculated for the optimal threshold; in our experiments this threshold is estimated over the ranks of the documents. Thus, MQWV doesn't depend on the ability to estimate the threshold and only depends on the quality of the document ranking for a given query.

Lang.	Model	EDI NMT	UMD NMT	UMD SMT
BG	Acous.	20.48	20.39	**21.24**
	Seg	22.38*	23.35*	21.23
	Seg + Sub	**24.73***	**25.92***	21.23
	Align	24.98	27.81	21.29
	Ref	43.75	35.40	29.50
FA	Acous.	5.35	6.26	**4.54**
	Seg	5.32	6.22	3.28
	Seg + Sub	**6.47***	**6.83***	4.50
	Align	7.67	7.02	4.59
	Ref	17.08	11.24	7.76
LT	Acous.	15.20	**8.38**	14.76
	Seg	15.18	8.34	14.76
	Seg + Sub	**15.22**	8.33	14.76
	Align	15.60	8.71	14.84
	Ref	20.40	11.94	21.30

Table 3: Document-level BLEU scores on ANALYSIS set. * represents statistical significance when compared to Acous. at the 0.05 level. We show the result of translations using the original acoustic segmentation (Acous.), our model trained only on the BUILD dataset (Seg), and our full model (Seg + Sub). For reference, we provide the scores of translation on ASR tokens aligned to the reference transcription segmentation (Align), and the reference transcription (Ref).

8 Results

8.1 MT

Table 3 shows the results of the MT evaluation. The best non-reference system for each language and MT system is in bold. We compute statistical significance against the acoustic (*Acous.*) segmentation baseline using Welch's T Test (Welch, 1947). Our subtitle-based segmentation model (*Seg + Sub*) consistently improves BLEU scores of NMT models for BG and FA, while not making significant differences in SMT. This echoes prior work (Khayrallah and Koehn, 2018; Rosales Núñez et al., 2019) suggesting SMT models are more robust to noisy inputs than neural models.

Lang.	Model	EDI NMT	UMD NMT	UMD SMT
BG	Acous.	0.173	0.134	0.164
	Seg + Sub	**0.177**	**0.180**	0.164
FA	Acous.	0.040	0.039	0.046
	Seg + Sub	**0.071**	**0.042**	**0.145**
LT	Acous.	0.128	**0.067**	0.157
	Seg + Sub	**0.136**	0.060	**0.172**

Table 4: MQWV scores on Test (Small) set.

In 6 out of 9 cases, we see that adding the subtitles data improves over using only the BUILD data. Of the remaining cases, the scores remain similar (i.e., it doesn't hurt the model). Training on the BUILD data alone improves BG NMT models, but for SMT and the other languages, it either makes no difference or is worse than the acoustic model.

Comparing Seg + Sub with Align in all languages, we see that there is only a small gap between the two. This suggests that our model is nearing the ceiling on what correcting segmentation can do to improve downstream MT. Furthermore, on LT where our model offers only small or no improvement, we see that the original acoustic segmentation is almost performing as well as Align. This suggests that there is relatively little room for improving LT MT by correcting sentence boundaries alone.

8.2 Document-Level CLIR

MQWV on the Test (Small) and Test (Large) partitions are shown in Table 4 and Table 5 respectively. On the Test (Small) partition, we see that our segmentation model improves the CLIR performance over the acoustic segmentation in 7 out of 9 cases. On the Test (Large) partition, we see that our segmentation model improves downstream retrieval performance consistently across all three MT systems. We note that while we measure the downstream retrieval performance separately for each MT system, a real-world CLIR system could perform IR over the union of multiple MT systems, which could yield even further improvements in retrieval performance (Zhang et al., 2020).

8.3 Complexity Analysis

Lang.	Model	EDI NMT	UMD NTM	UMD SMT
LT	Acous.	0.292	0.175	0.328
	Seg + Sub	**0.293**	**0.179**	**0.399**

Table 5: MQWV scores on the Test (Large) set.

ARI Quartile	Model	EDI NMT	UMD NMT
Q1	Acous.	17.78	**24.70**
	Seg + Sub	**17.92**	11.62
Q2	Acous.	20.76	22.09
	Seg + Sub	**26.89**	**31.24**
Q3	Acous.	22.87	20.38
	Seg + Sub	**29.96**	**33.22**
Q4	Acous.	23.41	21.00
	Seg + Sub	**29.87**	**35.24**

Table 6: Bulgarian BLEU scores on Test (Small) (transcribed portion) when separated into quartiles by sentence complexity (as measured by ARI).

We hypothesize that the effects of improved segmentation should be more pronounced for more complex utterances with more opportunities to misplace boundaries. Therefore, we calculate a measure of sentence complexity, the Automated Readability Index (ARI) (Senter and Smith, 1967), for all documents in Test (Small)[6] and examine the performance of our *Sub* model on MT. We separate the documents into quartiles based on their calculated ARI, where a higher ARI (and thus a higher quartile) indicates a more complex document, and present the average document-level BLEU score for each quartile in Table 6. In the interest of space, we present results for Bulgarian and for NMT, and defer other languages and SMT to Appendix A. We see that the most dramatic gains in BLEU occur for documents in the third and fourth quartiles, which matches our intuition. In other words, our segmentation model most improves the translation quality of more syntactically complex segments.

9 Ablation Study

We perform an ablation study on two components in our proposed model, *(i)* the use of acoustic segmentation boundary labels γ as input and *(ii)* training with a combination of over- and under-segmentation noise. We use the same training and

---

[6] Only the transcribed portion with reference translations. For each MT system, we compute ARI on the document translation using the reference transcription.

Model	EDI NMT	UMD NMT	UMD SMT
Acous.	20.48	20.39	21.24
Lex.	23.96	24.97	**21.58**
Lex. + Under	24.27	25.84	21.48
Lex. + Over	24.69	25.88	21.23
Full	**24.73**	**25.92**	21.23

Table 7: Document-level BLEU score for the models in ablation studies. We provide the acoustic model (Acous.) and our proposed model (Full).

evaluation process and only modify the affected component. We perform our ablation on the BG MT task, since it had a wider range of improvements than the other languages.

Use of Acoustic Segmentation Boundaries. We train a segmentation model using only ASR output tokens $\mathbf{x}$ as input without the the ASR segmentation sequence γ. For this model, we modify the embedding representation $\mathbf{e}_i$ so that we do not use F:

$$\mathbf{e}_i = G(x_i)$$

This model, which we refer to as *Lex.*, must exclusively use the lexical information of the ASR token sequence $\mathbf{x}$ to make predictions.

Over-segmentation and Under-segmentation. The two segmentation problems of the system may have different impact on the MT system. To see their individual effects, we train two models where the synthetic acoustic segmentation boundary sequence γ is created using only under-segmentation or over-segmentation noise. We refer to those models as *Lex. + Under* and *Lex. + Over* respectively.

Results Table 7 shows the effects of the model ablations on MT system BLEU score. On both NMT systems, we see that there is a roughly 1 point improvement on BLEU when including the ASR segmentation boundaries as input. For both NMT models we also find that over-segmentation noise helps slightly more than adding under-segmentation noise, but that these additions are complementary, i.e. the full model does best overall. For SMT, we surprisingly find that model without acoustic segmentation boundary input does best. The overall difference between the acoustic (*Acous.*) baseline and any of the segmentation correction models is small compared to the gains had on NMT. This again suggests that SMT is more robust to changes in segmentation.

10 Conclusion

We propose an ASR segmentation correction model for improving SLT pipelines. Our model makes use of subtitles data as well as a simple model of acoustic segmentation error to train an improved ASR segmentation model. We demonstrate downstream improvements on MT and CLIR tasks. In future work, we would like to find a better segmentation error model that works well in conjunction with SMT systems in addition to NMT systems.

Acknowledgements

This research is based upon work supported inpart by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract #FA8650-17-C-9117. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes not withstanding any copyright annotation therein.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondřej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander Waibel, and Changhan Wang. 2020. FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 1–34, Online. Association for Computational Linguistics.

Parnia Bahar, Patrick Wilken, Tamer Alkhouli, Andreas Guta, Pavel Golik, Evgeny Matusov, and Christian Herold. 2020. Start-Before-End and End-to-End: Neural Speech Translation by AppTek and RWTH Aachen University. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 44–54, Online. Association for Computational Linguistics.

Andrea Carmantini, Peter Bell, and Steve Renals. 2019. Untranscribed Web Audio for Low Resource Speech Recognition. In *Proceedings of Interspeech 2019*, pages 226–230.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT^3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Eunah Cho, Jan Niehues, and Alex Waibel. 2017. NMT-Based Segmentation and Punctuation Insertion for Real-Time Spoken Language Translation. In *Proceedings of Interspeech 2017*, pages 2645–2649.

Heidi Christensen, Yoshihiko Gotoh, and Steve Renals. 2001. Punctuation Annotation using Statistical Prosody Models. In *Proceedings of ISCA Workshop on Prosody in Speech Recognition and Understanding*, pages 35–40.

Laura Cross Vila, Carlos Escolano, José A. R. Fonollosa, and Marta R. Costa-Jussà. 2018. End-to-End Speech Translation with the Transformer. In *Proceedings of IberSPEECH 2018*, pages 60–63.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-end Speech-Translation with Knowledge Distillation: FBK@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 80–88, Online. Association for Computational Linguistics.

Yoshihiko Gotoh and Steve Renals. 2000. Sentence Boundary Detection in Broadcast Speech Transcripts. In *Proceedings of ISCA Workshop: Automatic Speech Recognition: Challenges for the new Millennium ASR-2000*, pages 228–235.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2018. The Sockeye Neural Machine Translation Toolkit at AMTA 2018. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 200–207, Boston, MA. Association for Machine Translation in the Americas.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast Neural Machine Translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia.

Huda Khayrallah and Philipp Koehn. 2018. On the Impact of Various Types of Noise on Neural Machine Translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.

Ji-Hwan Kim and Philip C. Woodland. 2001. The use of Prosody in a Combined System for Punctuation Generation and Speech Recognition. In *EUROSPEECH 2001 Scandinavia, 7th European Conference on Speech Communication and Technology, 2nd INTERSPEECH Event, Aalborg, Denmark, September 3-7, 2001*, pages 2757–2760. ISCA.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Nikhil Kumar Lakumarapu, Beomseok Lee, Sathish Reddy Indurthi, Hou Jeung Han, Mohd Abbas Zaidi, and Sangha Kim. 2020. End-to-End Offline Speech Translation System for IWSLT 2020 using Modality Agnostic Meta-Learning. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 73–79, Online. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Yang Liu, Andreas Stolcke, Elizabeth Shriberg, and Mary Harper. 2005. Using Conditional Random Fields for Sentence Boundary Detection in Speech. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 451–458, Ann Arbor, Michigan. Association for Computational Linguistics.

Wei Lu and Hwee Tou Ng. 2010. Better Punctuation Prediction with Dynamic Conditional Random Fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 177–186, Cambridge, MA. Association for Computational Linguistics.

Evgeny Matusov, Dustin Hillard, Mathew Magimai-doss, Dilek Hakkani-tur, Mari Ostendorf, and Hermann Ney. 2007. Improving Speech Translation with Automatic Boundary Prediction. In *Proceedings of Interspeech*, pages 2449–2452.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

NIST. 2017. *The Official Original Derivation of AQWV*.

Xing Niu, Michael Denkowski, and Marine Carpuat. 2018. Bi-Directional Neural Machine Translation with Synthetic Parallel Data. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 84–91, Melbourne, Australia. Association for Computational Linguistics.

Douglas W. Oard, Marine Carpuat, Petra Galuscakova, Joseph Barrow, Suraj Nair, Xing Niu, Han-Chin Shing, Weijia Xu, Elena Zotkina, Kathleen McKeown, Smaranda Muresan, Efsun Selin Kayi, Ramy Eskander, Chris Kedzie, Yan Virin, Dragomir R. Radev, Rui Zhang, Mark J F Gales, Anton Ragni, and Kenneth Heafield. 2019. Surprise languages: Rapid-Response Cross-Language IR. In *Proceedings of the Ninth International Workshop on Evaluating Information Access (EVIA 2019)*, pages 23–27. National Institute of Informatics. The Ninth International Workshop on Evaluating Information Access (EVIA 2019) : a Satellite Workshop of the NTCIR-14 Conference, EVIA 2019 ; Conference date: 10-06-2019 Through 10-06-2019.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ngoc-Quan Pham, Thai-Son Nguyen, Thanh-Le Ha, Juan Hussain, Felix Schneider, Jan Niehues, Sebastian Stüker, and Alexander Waibel. 2019. The IWSLT 2019 KIT Speech Translation System. In *International Workshop on Spoken Language Translation (IWSLT)*.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Anton Ragni and Mark Gales. 2018. Automatic Speech Recognition System Development in the "Wild". In *Proceedings of Interspeech 2018*, pages 2217–2221.

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation Strategies for Streaming Speech Translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 230–238, Atlanta, Georgia. Association for Computational Linguistics.

Sharath Rao, Ian Lane, and Tanja Schultz. 2007. Optimizing Sentence Segmentation for Spoken Language Translation. In *Proceedings of Interspeech*.

José Carlos Rosales Núñez, Djamé Seddah, and Guillaume Wisniewski. 2019. Comparison between NMT and PBSMT Performance for Translating Noisy User-Generated Content. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 2–14, Turku, Finland. Linköping University Electronic Press.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

R.J. Senter and E.A. Smith. 1967. Automated Readability Index. *AMRL-TR. Aerospace Medical Research Laboratories.*

Elizabeth Shriberg, Andreas Stolcke, Dilek Hakkani-Tür, and Gükhan Tür. 2000. Prosody-Based Automatic Segmentation of Speech into Sentences and Topics. *Speech Communication*, 32(1–2):127–154.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-Passing Models for Robust and Data-Efficient End-to-End Speech Translation. *Transactions of the Association for Computational Linguistics*, 7:313–325.

Matthias Sperber, Ngoc Quan Pham, Thai Son Nguyen, Jan Niehues, Markus Müller, Thanh-Le Ha, Sebastian Stüker, and Alexander Waibel. 2018. KIT's IWSLT 2018 SLT Translation System. In *International Workshop on Spoken Language Translation (IWSLT)*.

Andreas Stolcke and Elizabeth Shriberg. 1996. Automatic Linguistic Segmentation of Conversational Speech. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*, 2:1005–1008 vol.2.

Trevor Strohman, Donald Metzler, Howard Turtle, and W. Croft. 2005. Indri: A language-model based search engine for complex queries. *Information Retrieval*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

David Wan, Zhengping Jiang, Chris Kedzie, Elsbeth Turcan, Peter Bell, and Kathy McKeown. 2020. Subtitles to Segmentation: Improving Low-Resource Speech-to-Text Translation Pipelines. In *Proceedings of the workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 68–73, Marseille, France. European Language Resources Association.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence Models Can Directly Translate Foreign Speech. In *Proceedings of Interspeech 2017*, pages 2625–2629.

B. L. Welch. 1947. THE GENERALIZATION OF 'STUDENT'S' PROBLEM WHEN SEVERAL DIFFERENT POPULATION VARLANCES ARE INVOLVED. *Biometrika*, 34(1-2):28–35.

Le Zhang, Damianos Karakos, William Hartmann, Manaj Srivastava, Lee Tarlin, David Akodes, Sanjay Krishna Gouda, Numra Bathool, Lingjun Zhao, Zhuolin Jiang, Richard Schwartz, and John Makhoul. 2020. The 2019 BBN Cross-lingual Information Retrieval System. In *Proceedings of the workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 44–51, Marseille, France. European Language Resources Association.

A Full Complexity Analysis

We present the full results of our complexity analysis as described in subsection 8.3. Bulgarian (Table 8), Lithuanian (Table 9), and Farsi (Table 10) results are shown for all three MT models as well as both the acoustic segmentation and our *Seg + Sub* segmentation correction model. The best score for each MT system and quartile is bolded.

ARI Quartile	Model	EDI NMT	UMD NMT	UMD SMT
Q1	Acous.	17.78	**24.70**	22.26
	Seg + Sub	**17.92**	11.62	**22.58**
Q2	Acous.	20.76	22.09	23.35
	Seg + Sub	**26.89**	**31.24**	**23.43**
Q3	Acous.	22.87	20.38	**22.45**
	Seg + Sub	**29.96**	**33.22**	21.68
Q4	Acous.	23.41	21.00	22.97
	Seg + Sub	**29.87**	**35.24**	**23.25**

Table 8: Bulgarian BLEU scores on Test (Small) (transcribed portion) when separated into quartiles by sentence complexity (as measured by ARI).

ARI Quartile	Model	EDI NMT	UMD NMT	UMD SMT
Q1	Acous.	4.24	**3.13**	**5.14**
	Seg + Sub	**4.32**	2.76	5.11
Q2	Acous.	**14.07**	**7.18**	**13.45**
	Seg + Sub	14.01	6.83	13.39
Q3	Acous.	15.95	7.81	14.85
	Seg + Sub	**16.39**	**8.13**	**14.87**
Q4	Acous.	**15.41**	**7.94**	15.85
	Seg + Sub	14.60	7.78	**15.87**

Table 9: Lithuanian BLEU scores on Test (Small) (transcribed portion) when separated into quartiles by sentence complexity (as measured by ARI).

ARI Quartile	Model	EDI NMT	UMD NMT	UMD SMT
Q1	Acous.	3.46	3.40	3.21
	Seg + Sub	**6.05**	**4.49**	**3.43**
Q2	Acous.	3.55	3.87	3.22
	Seg + Sub	**5.06**	**5.41**	**3.66**
Q3	Acous.	4.71	5.33	5.19
	Seg + Sub	**7.44**	**7.29**	**5.54**
Q4	Acous.	5.88	6.15	3.90
	Seg + Sub	**7.81**	**7.59**	**4.47**

Table 10: Farsi BLEU scores on Test (Small) (transcribed portion) when separated into quartiles by sentence complexity (as measured by ARI).

Crisscrossed Captions: Extended Intramodal and Intermodal Semantic Similarity Judgments for MS-COCO

Zarana Parekh, Jason Baldridge, Daniel Cer, Austin Waters, and Yinfei Yang
Google Research
`{zarana, jasonbaldridge, cer, austinwaters, yinfeiy}@google.com`

Abstract

By supporting multi-modal retrieval training and evaluation, image captioning datasets have spurred remarkable progress on representation learning. Unfortunately, datasets have limited cross-modal associations: images are not paired with other images, captions are only paired with other captions of the same image, there are no negative associations and there are missing positive cross-modal associations. This undermines research into how inter-modality learning impacts intra-modality tasks. We address this gap with *Crisscrossed Captions* (CxC), an extension of the MS-COCO dataset with human semantic similarity judgments for **267,095** intra- and inter-modality pairs. We report baseline results on CxC for strong existing unimodal and multimodal models. We also evaluate a multitask dual encoder trained on both image-caption and caption-caption pairs that crucially demonstrates CxC's value for measuring the influence of intra- and inter-modality learning.

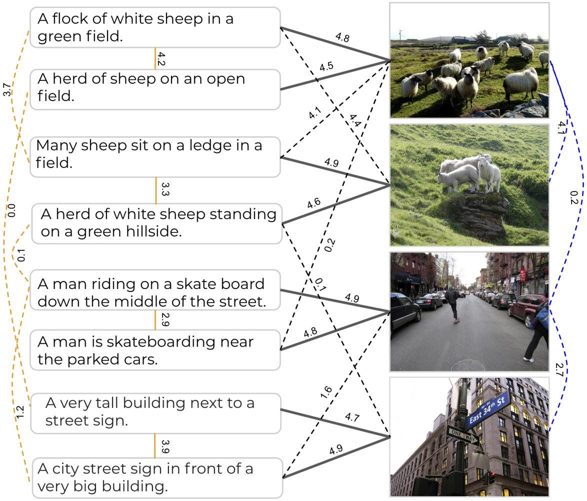


Figure 1: Crisscrossed Captions extends the MS-COCO evaluation sets by adding semantic similarity ratings for existing image-caption pairs and co-captions (solid lines), and it increases annotation density by adding further ratings for new image-caption, caption-caption and image-image pairs (dashed lines).

1 Introduction

Phrases such as *blue*, *chair*, and *garden path* have strong visual components, yet computational word representations are usually created with text-only corpora. Encouragingly, some recent work that derives representations using visual contexts shows improvements for both word similarity ranking and image-text retrieval (Kiros et al., 2018), and query-based training of image models demonstrates language's power to improve image representations (Juan et al., 2020). Learning representations for both vision and language jointly should be even more effective—indeed, much progress has been made on such cross-modal learning using image captioning data (Karpathy and Li, 2015; Harwath and Glass, 2017; Faghri et al., 2018; Li et al., 2019). However, it is not yet clear whether learning representations in multimodal contexts improves performance *within* as well as across modalities as there are no datasets ideally suited for this at present.

Image captioning datasets such as Flickr8k (Rashtchian et al., 2010), Flickr30k (Young et al., 2014), Multi30k (Elliott et al., 2016), Microsoft Common Objects in COntext (MS-COCO) (Lin et al., 2014), and Conceptual Captions (Sharma et al., 2018) only capture relationships between images and textual captions created for them. They miss many valid relationships between unassociated images and captions, from captions to other captions, and from images to other images. We address this gap with Crisscrossed Captions (CxC, exemplified in Figure 1), a dataset with graded, denser annotations for relationships between and among captions and images in the MS-COCO evaluation splits of (Karpathy and Li, 2015) (with 25k English captions and 5k images each).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2855–2870
April 19 - 23, 2021.

CxC extends MS-COCO's existing image-caption pairs with continuous (0-5) semantic similarity ratings for those pairs and new pairs. The rating criteria extend those used for Semantic Textual Similarity (Agirre et al., 2012). Intramodal pairs are selected for annotation via an indirect sampling scheme biased to gain a broad distribution of similarities. In all, CxC contains *human* ratings for **267,095** pairs (derived from 1,335,475 independent judgments), a massive extension in scale and detail to the 50k original binary pairings.

MS-COCO *incompletely* supports three retrieval tasks: image-text, text-image and text-text. CxC enhances all of these with new positive pairs, and it also supports a *new* image-image retrieval task. With its graded similarity judgements, CxC also supports correlation measures comparing model and human rankings. Retrieval metrics focus on positive pairs, but CxC's correlation scores additionally account for low-scoring items (non matches). Supporting these evaluations on a common set of images and captions makes them more valuable for understanding inter-modal learning—compared to disjoint sets of caption-image, caption-caption, and image-image associations. Also, multimodal representations such as CLIP (Radford et al.) are useful for downstream tasks such as Visual Question Answering (Goyal et al., 2017), Vision and Language Navigation (Majumdar et al., 2020), Referring Expressions (Yu et al., 2018) and Visual Commonsense Reasoning (Zellers et al., 2019), and we hope the additional relationships and evaluations provided by CxC will help develop even better representations for tasks that span these modalities.

To establish baselines for CxC, we provide results for existing unimodal models for text (Bag-of-Words, USE (Cer et al., 2018)) and images (InceptionV3 (Szegedy et al., 2016), ResNet-152 (He et al., 2016), SimCLRv2 (Chen et al., 2020b) as well as for two cross-modal retrieval models VSE++ (Faghri et al., 2018) and VSRN (Li et al., 2019).

We furthermore demonstrate CxC's utility by evaluating a dual encoder that combines a bidirectional loss for image-text retrieval with a loss for text-text retrieval. The text encoder is composed of transformer layers over pre-trained BERT word representations and the image encoder is a pre-trained EfficientNet (B4) (Tan and Le, 2019a). This model delivers the strongest overall performance across all four retrieval tasks and correlation with human scores for text-text, image-image and image-text similarity. Compared to the same dual encoder trained only with image-text pairs, this model realizes small gains for image-text tasks and large gains for text-text task but with some degradation for image-image tasks. This indicates that the model trades capacity to encode images for better text encoding—an insight that would not be easily assessed without CxC's image-image annotations.

Our main contributions are the following:

- We describe a method for sampling items to get a broad distribution of similarities.
- We annotate the semantic similarity of 267,095 pairs. These enhance existing retrieval tasks and support a new image-image retrieval task. They also support correlation measures; these assess models' judgments of both positive *and* negative associations.
- We establish baseline scores for existing models and a multitask dual encoder on all tasks and demonstrate that CxC allows model performance to be assessed more holistically.
- With its new positive pairs, CxC improves the recall@k measures common in image-text and text-image retrieval. This shows a 1-3% increase in recall@k over several models.
- We release CxC's annotations at `https://github.com/google-research-datasets/Crisscrossed-Captions`, along with code to merge CxC with existing MS-COCO data.

2 Dataset Collection

Existing resources already support learning joint representations of images and text. However, we need better evaluation resources, so we extend the MS-COCO evaluation splits with graded similarity associations within and across modalities. MS-COCO has five captions for each image, split by (Karpathy and Li, 2015) into 410k training, 25k development, and 25k test captions (82k/5k/5k for images). An ideal extension would rate every pair, but this is infeasible[1] and most pairs are dissimilar anyway. To obtain new pairs with high expected similarity, we introduce a biased sampling scheme.

The data is collected in two phases. First, we define an indirect sampling scheme that uses model-based similarities from the co-modality items to

---

[1] A split with 5k images and 25k captions has ≈12.5M image-image, ≈312M caption-caption and ≈125M image-caption pairs, so annotating all items in the validation and test splits with 5 replications would require ≈4.5B judgments.

Caption 1: A tennis player swinging a racket at a ball.

Caption 2: A man playing tennis with a crowd watching.

Caption 3: A living room with some black furniture and a colorful rug.

Caption 4: A dog laying on a leather sofa in a living room.

Figure 2: **Top:** Captions of the same image are often not paraphrases. **Bottom:** Such *co-captions* often focus on different aspects and can diverge substantially.

select *intra*modality pairs. We use these items and their human ratings to select *inter*modality pairs for annotation. We also annotate all existing intermodal pairs and a large sample of co-captions (captions associated with the same image). See the appendix for details about the annotation interface and instructions, composition of the dataset and illustrative examples.

Intramodality Two images of a man and a dog can be described differently, while two similar sentences about a man and a dog can describe dissimilar images. In Figure 2, caption 1 gives a visual description while caption 2 gives a broader event description. Divergences also occur when caption creators perceive a scene differently: caption 3 describes the room and caption 4 focuses on the dog and sofa. This semantic gap between images and their captions creates an opportunity to sample *intra*modal pairs with varying similarities. Our key idea is to use model-based similarities of images for biased sampling of caption pairs, and vice versa, and use existing image-caption pairs as pivots between modalities. This selects image pairs that are different in appearance but similar in what they depict based on their descriptions, and vice versa.

Denote the known images and captions as V ($v_1 ... v_n$) and C ($c_1 ... c_n$) (the latter representing *co-caption* groups of five captions each). Each item is encoded with an off-the-shelf unimodal model. Cosine similarity between items defines two symmetric matrices: S^C (pairwise caption similarities) and S^V (pairwise image similarities). The diagonals are set to zero to not sample identical items.

We encode images with Graph-RISE (486) and construct S^I, the image-based similarity for pairs of co-caption groups. We encode captions with Universal Sentence Encoder (USE) (Cer et al., 2018) and average bag of words (BoW) based on GloVe embeddings (Pennington et al., 2014). Co-caption representations are averaged to create a single representation. From these, we construct S^C, the caption-based similarity for images pairs. USE and BoW embeddings produce two S^C matrices, but we gloss over this detail below.

We use S^C to select *image* pairs and S^I for *caption* pairs. Because of the cross-modal semantic gap, diversity and size of the underlying data, these pairs exhibit a wide range of similarity. Selecting the five most similar items (according to model-based S^V and S^C) thus produces good representation of varying amounts of similarity as judged by people. Because S^V covers co-caption groups, one caption is randomly chosen from each group to produce a caption pair for rating.

Caption-caption and image-image candidates are referred to as $C2C$ and $I2I$, respectively. $I2I$ pairs are selected with the above *other-modality* method. For $C2C$ pairs, we sample half the pairs using the other-modality method and half from within co-captions. The latter introduces (mostly) positive associations between caption pairs describing the same image. This gives a balanced set of caption pairs describing same and different images.

Pairs in $C2C$ and $I2I$ are scored by in-house raters using a continuous scale between 0 and 5. We adopt the widely used Semantic Textual Similarity (STS) (Cer et al., 2017) for text pairs and extend it to images to define Semantic Image Similarity (SIS). To recognize that this is a graded (rather than discrete) judgment, we encouraged raters to select scores like *1.3* and obtain the final score for a pair as the average of five individual ratings.

Intermodality We select caption-image candidates $C2I$ based on human ratings for $I2I$ and $C2C$ pairs. We mainly seek new positive matches like those identified by annotators in Ilharco et al. (2019). For each $I2I$ pair (i_j, i_k), a $C2I$ pair (c_k, i_j) is generated, where c_k is a MS-COCO caption for i_k. We generate pairs from $C2C$ similarly. Half of the $C2I$ pairs are selected based on $C2C$ ranks and the other half by $I2I$ ranks (skipping pairs already selected from $C2C$). Finally, all MS-COCO pairs (25k in validation and 25k in test) are selected to obtain caption-image similarity ratings for the known items.

Caption 1	**Caption 2**	**STS**	**BoW**	**USE**
A man standing on a tennis court holding a racquet.	A man standing on a tennis court holding a tennis racquet.	5.0	0.98	0.99
A man riding a skateboard off the side of a ramp.	A man riding up the side of a skateboard ramp.	4.2	0.99	0.94
A yellow tray topped with a cup of coffee and a donut.	A white plate topped with donuts sitting on a stove top.	3.1	0.94	0.39
A bird sitting on top of a park bench.	An empty park bench sitting in front of trees.	2.2	0.90	0.69
An old car sitting on top of a lush green field.	A couple of motorcycles parked next to each other.	1.3	0.85	0.21
A man sanding next to an orange frisbee.	A couple of swans swimming in a pond next to two people.	0.2	0.84	0.11

Table 1: A comparison of CxC STS annotation scores and cosine similarity scores using GloVe BoW embeddings and Universal Sentence Encoder (USE) for five example MS-COCO caption pairs.

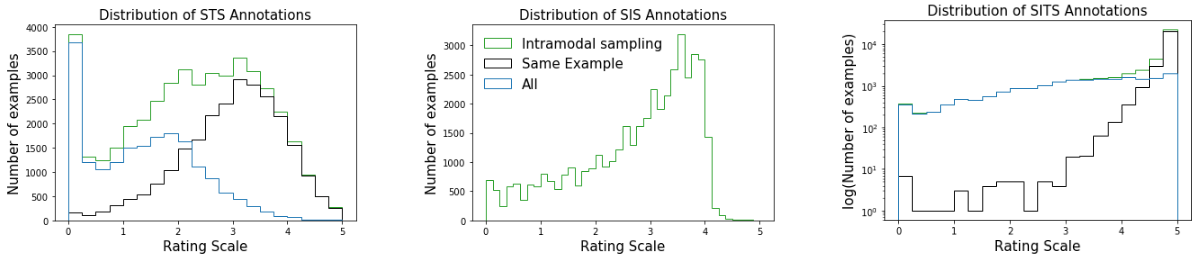


Figure 3: Distribution of ratings for the CxC validation set. *Intramodal Sampling* refers to examples selected using *other-modality* selection, *Same Example* refers to original MS-COCO pairs, and *All* covers all examples for a task.

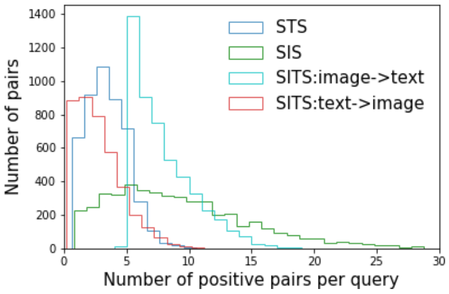


Figure 4: Distribution of counts of positive pairs (score ≥ 3) of annotations for each task (validation split).

We extend STS to define Semantic Image-Text Similarity (SITS). Raters provide a continuous score from 0 to 5 using an interface similar to that for STS and SIS. Each $C2I$ pair receives five ratings; the average is used as the final SITS score.

3 Crisscrossed Captions Dataset

Using our selection and annotation methodology, we obtained ratings for 267,095 caption-caption, image-image, and caption-image pairs (1,335,475 total judgments). Figure 3 shows rating distributions for each task (validation split). It also shows the distributions of ratings for STS and SIS pairs included from other-modality selection and from original MS-COCO pairs. The test set distributions are similar. Figure 4 gives the distribution of counts of positive examples in each task (validation split), where a score ≥ 3 (for STS, SITS) and a score ≥ 2.5 (for SIS) is considered positive. These positive examples are used for intermodal and intramodal retrieval evaluation.

STS. The majority of caption pairs selected using image similarity are negative (ratings in [0, 3)), which is expected given the divergences noted in Figure 2. Nevertheless, the approach produces 20,587 positive pairs. Table 1 shows pairs with their STS annotation scores and cosine similarity with BoW and USE embeddings. There is broad agreement, but the annotated similarity is not fully captured by either BoW or USE. USE provides a broader range, but scores the third pair lower than the fourth. BoW scores are bunched within a high similarity band[2] that aligns well with these five examples. Overall, there is a weak positive correlation between BoW and STS scores, as shown in Figure 5, which plots average BoW cosine similarity versus STS for 1000 randomly sampled pairs.

Figure 6 shows a pair of captions (and corresponding images) selected by the other-modality strategy with higher STS compared to their respective co-captions. For co-caption pairs, STS scores are more positive but many are still negative (Figure 3, left). Thus, combining both approaches leads to a more representative distribution overall. The large number of negative pairs from co-captions underscores the problem with assuming captions of the same image are paraphrases.

SIS. All image pairs $I2I$ are selected using the other-modality strategy. This plus the stringent cri-

---

[2] BoW scores fall mostly in the range .8 to 1.0 over all possible pairs; STS scores fall mostly in 1 (.2) to 4 (.8).

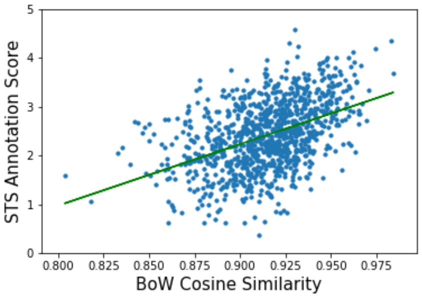
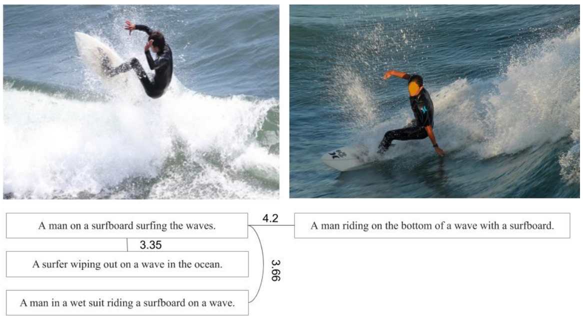


Figure 5: Plot BoW cosine similarity and STS scores for a sample of caption pairs, with the line of best fit.



A man on a surfboard surfing the waves.

A man riding on the bottom of a wave with a surfboard.

A surfer wiping out on a wave in the ocean.

A man in a wet suit riding a surfboard on a wave.

Figure 6: An example of *other-modality* caption pairs (horizontal pair) with higher STS compared to their respective co-captions (vertical pairs). Each caption is placed under its corresponding image.

teria for SIS rating of 5 means very few examples are rated above 4. Nevertheless, there are many pairs with SIS ≥ 3, indicating there are many images depicting similar scenes and events.

SITS. As shown in Figure 4, there are many more pairs with 4-5 SITS ratings, compared to STS and SIS. This is by design, as the $C2I$ pairs are selected based on decreasing STS/SIS scores. This captures more positive intermodality associations and augments the existing validation and test splits. Since these pairs are missing from the existing data, they are among the examples that inappropriately penalize a model that identifies them correctly in image-caption retrieval. The SITS ratings collected for known pairs also support new correlation based evaluations, as discussed in the next section.

4 Evaluation Tasks and Metrics

CxC supports intermodal retrieval like MS-COCO but with denser annotations between image-caption pairs. It also enables intramodal retrieval and semantic similarity correlation evaluations, which were not possible before.

Karpathy and Li (2015) first used MS-COCO for image-to-caption and caption-to-image retrieval. We extend the existing associations with positive CxC pairs, and also add new caption-to-caption and image-to-image retrieval tasks using positive STS and SIS pairs (a total of four retrieval tasks). To the best of our best knowledge, CxC is the first dataset to support image-to-image retrieval over captioned images. Following Karpathy and Li (2015), we evaluate using Recall@K (R@K), computed as the fraction of times a correct item was found among the top K results, and median rank (med. r) of the closest ground truth result in the list.

Semantic similarity tasks such as Semantic Textual Similarity (Cer et al., 2017) and Visual Semantic Textual Similarity(vSTS) (de Lacalle et al., 2020) require a model to produce a continuous similarity score given two inputs. Typically, the models are evaluated based on the Pearson's r of their scores with the human judgments over a set of input pairs. This is valid when training data is available to calibrate model scores to the human ratings. With CxC, we do not have such training data, so we instead use Spearman's r to assess whether a model *ranks* pairs similarly to human raters.

It would be tempting to simply measure Spearman's r over all pairs, but this would be flawed because CxC's dense annotation means that the scores between many pairs are themselves correlated. To mitigate this, we use a sampled bootstrap correlation instead. For each correlation estimate, we sample half of the queries (to increase diversity across samples) and for each selected query, we choose one of the items for which CxC supplies a paired rating. We compute Spearman's r between the CxC scores and the model scores for the selected pairs. The final correlation is the average over 1000 of these bootstrap samples.

vSTS (de Lacalle et al., 2020) contains 2677 pairs of MS-COCO captions and corresponding images. As noted above, vSTS is related dataset for multimodal semantic similarity. We considered mixing CxC and vSTS; however, this was infeasible because CxC uses the widely adopted Karpathy splits, while items in vSTS's training, dev and test splits are spread among the Karpathy splits. We could not just make a separate cut of CxC because vSTS pairs can cross splits, e.g. an image-caption item in Karpathy training and another in Karpathy test. Given the small size of vSTS, we focused our efforts on CxC evaluations.

5 Evaluated Models

In order to establish baselines for CxC, we benchmark pretrained models for images and text. Note that these are off-the-shelf models that have not been trained on MS-COCO. We also evaluate cross-modal retrieval models that are trained on MS-COCO. Here, we focus on models that support efficient retrieval (e.g. dual encoders). We expect models with extensive cross-modal interactions, such as ViLBERT (Lu et al., 2019) and LXMERT (Tan and Bansal, 2019), will show strong performance on CxC tasks, either as standalone models that (inefficiently) score all possible item pairs or as rerankers for outputs of retrieval models.

To the best of our knowledge, there is no prior work that explores joint learning or evaluation on intra- and inter-modality retrieval tasks. Ngiam et al. (2011) and Collell Talleda and Moens (2016) show evidence that inter-modality learning helps improve intra-modality performance, but do not explore multitask learning. Lu et al. (2020) explore multitask learning but only focus on intermodal representation learning for intermodal downstream tasks. To illustrate how CxC allows us to measure how intermodal representation learning can improve both intra- and inter-modal performance, we train a dual encoder model on bidirectional image-text and text-text in-batch retrieval losses.

5.1 Pretrained Model Baselines

Text-only Models First, we use a bag-of-words (*BoW*) approach using averaged GloVe embeddings (Pennington et al., 2014) for each token in a caption as the caption representation. Second, the Universal Sentence Encoder (*USE*) (Cer et al., 2018) is a sentence level representation model that has shown strong performance on the related STS benchmark. We use the multilingual transformer version from TensorFlow Hub (Yang et al., 2020).[3]

Image-only Models *InceptionV3*, *ResNet-152*, and *SimCLRv2* are deep convolutional models (Szegedy et al., 2016; He et al., 2016; Chen et al., 2020a,b) trained on the ImageNet dataset. We extract 2048-dimensional image-level representations on a central crop containing 87.5% of the original image area. We access them via TensorFlow Hub.[4]

Intermodal Models *VSE++* (Faghri et al., 2018) is a dual encoder (see Sec. 5.2) trained to learn a joint space of aligned images and captions. The state-of-the-art VSRN model (Li et al., 2019) is another dual encoder that uses additional training annotations to predict and use bounding boxes for more fine-grained and coherent image analysis, while using only a simple text encoder trained from scratch.[5]

5.2 Dual Encoder Baselines

We also consider several neural baseline models, all of which are *dual encoders* (Gillick et al., 2018; Yang et al., 2019) that encode both inputs separately. Dual encoder models have been proven as an effective approach to learn strong semantic representations (Cer et al., 2018; Chen et al., 2020a,b). They are often trained using an *in-batch sampled softmax* loss, as this has been observed to converge quickly and perform well on retrieval tasks (Gillick et al., 2018; Yang et al., 2019). We employ the bidirectional in-batch sampled softmax loss (eq. 1):

$$\begin{aligned}\mathcal{L} = &-\frac{1}{K}\sum_{i=1}^{K}\left(S(l_i,r_i) - \log\sum_{j=1\, j\neq i}^{K} e^{S(l_i,r_j)}\right)\\ &-\frac{1}{K}\sum_{i=1}^{K}\left(S(r_i,l_i) - \log\sum_{j=1\, j\neq i}^{K} e^{S(r_i,l_j)}\right)\end{aligned} \qquad (1)$$

where $S(x, y)$ is the dot product of embeddings of examples x and y. This loss encourages the score of a correct pair $S(l_i,\, r_i)$ to be higher than scores of non-matching input pairs from the batch $S(l_i,\, r_j)$. Unlike full cross-attention models, this architecture enables large-scale retrieval through approximate nearest neighbor search.

We train dual encoders for caption-image and caption-caption tasks, as well as a multitask model that combines both tasks. We use EfficientNet-B4 (Tan and Le, 2019b) (pre-trained on ImageNet) as our image encoder; it yields a 1792-dimensional representation. The text encoder employs a frozen[6] BERT-Base model (Devlin et al., 2019) followed by three transformer layers. The additional transformer layers have 8 attention heads, hidden dimension of 3072, and–like BERT-base–output 768-dimensional token-level features. We use the fea-

---

[3] universal-sentence-encoder-multilingual-large/1

[4] imagenet/inception_v3/feature_vector/4, imagenet/resnet_v1_152/feature_vector/4 and gs://simclr-checkpoints/simclrv2/finetuned_100pct/r50_1x_sk0/hub/ respectively

[5] Models available at https://github.comfartashfvsepp (checkpoint ”runs/coco_vse++/model_best.pth.tar”) and https://github.com/KunpengLi1994/VSRN (checkpoint ”pretrain_model/coco/model_coco_1.pth.tar”).

[6] Freezing BERT makes performance slightly worse, but makes training much faster.

Annotations	Model	Image → Text				Text → Image			
		R@1	R@5	R@10	med r	R@1	R@5	R@10	med r
MS-COCO	VSE++	41.3	71.1	81.2	2	30.3	59.4	72.4	4
	VSRN-github	50.3	79.6	87.9	1	37.9	68.5	79.4	2
	VSRN-paper	53.0	81.1	89.4	-	40.5	70.6	81.1	-
	DE_{I2T}	52.0	80.3	89.5	1	37.9	67.6	78.8	2
	$\text{DE}_{\text{T2T+I2T}}$	54.1	81.8	89.9	1	39.7	70.0	80.9	2
CxC	VSE++	43.1	74.3	84.2	2	32.5	62.7	75.4	3
	VSRN-github	52.4	81.9	90.0	1	40.1	71.1	81.5	2
	DE_{I2T}	53.9	82.7	91.2	1	39.8	70.2	80.9	2
	$\text{DE}_{\text{T2T+I2T}}$	55.9	84.2	91.8	1	41.7	72.3	83.0	2

Table 2: Image ↔ Text retrieval results on the MS-COCO 5k test set and **CxC's extended pairs for the same**.

Model	Text → Text			
	R@1	R@5	R@10	med r
BoW	21.2	38.2	47.4	13
$\text{USE}_{\text{mling-large}}$	31.2	51.5	61.3	5
VSE++	38.7	62.3	72.2	3
VSRN-github	41.0	64.8	74.5	2
DE_{T2T}	41.7	64.4	73.4	2
DE_{I2T}	26.0	47.1	57.5	7
$\text{DE}_{\text{T2T+I2T}}$	42.4	64.9	74.0	2

Table 3: Text ↔ Text retrieval performance on MS-COCO 5k test set **using CxC annotations**.

Model	Image → Image			
	R@1	R@5	R@10	med r
InceptionV3	4.1	13.3	19.1	96
ResNet-152	11.8	35.5	49.5	11
SimCLRv2	24.5	54.9	68.1	4
VSE++	36.4	70.4	81.3	2
VSRN-github	44.2	76.7	86.2	2
DE_{I2T}	38.3	74.1	85.0	2
$\text{DE}_{\text{T2T+I2T}}$	38.5	73.6	84.9	2

Table 4: Image ↔ Image retrieval performance on MS-COCO 5k test set **using CxC annotations**.

tures at the 0^{th} token position of the final layer as the caption representation. BERT parameters are initialized from the public BERT checkpoint.[7] The additional, trainable transformer layers are randomly initialized.

We construct three dual encoder models from these base encoders. (1) A *Text-Text* model (DE_{T2T}) uses a shared text encoder for both sides. (2) An *Image-Text* model (DE_{I2T}) uses the aforementioned text and image encoders, and includes a layer above the text encoder to project its 768 dimensions to 1792 (to match the image encoder output). (3) A *Multitask* model ($\text{DE}_{\text{T2T+I2T}}$) is trained on a combination of tasks (Chidambaram et al., 2019). It shares DE_{I2T}'s architecture and is trained in the same way; however, its loss is a weighted sum of image-text ($i2t$, $t2i$) and text-text ($t2t$) losses:

$$\mathcal{L} = \mathcal{L}_{i2t} + \mathcal{L}_{t2i} + c * \mathcal{L}_{t2t} \tag{2}$$

Here c is a scalar controlling the weights of losses from each task. This model has one text encoder, shared between all retrieval tasks. For hyperparameter tuning and training setup, see the appendix.

6 Results

Intermodal Retrieval Table 2 summarizes intermodal retrieval performance on both the original MS-COCO annotations and CxC. We report performance of two versions of VSRN (Li et al., 2019)–one using the checkpoint on the author's Github (VSRN-github, which allows us to perform CxC evaluations) and the other from the original paper (VSRN-paper, which has higher MS-COCO scores). Comparing each model on MS-COCO and CxC, the new positive items added by CxC show improved retrieval performance as they identify missing positives that are incorrectly penalized when using only original pairs (as noted in Ilharco et al. (2019) for Flickr8k). Multitask training in $\text{DE}_{\text{T2T+I2T}}$ provides a boost over using only intermodal pairs for training (i.e. DE_{I2T}). It performs similarly with VSRN-paper—it seems likely that VSRN's greater investment on the image analysis (with representations based on extracted object bounding boxes) is matched by $\text{DE}_{\text{T2T+I2T}}$'s greater investment in the text encoder.

Figure 7 shows three examples of images retrieved for caption queries. The CxC annotations capture missing examples in the first two cases, and the last two show there are still more positive pairs that remain unassociated in CxC. Figure 8 shows the same for captions retrieved from image queries, again showing that many examples are captured in CxC that are missing in MS-COCO.

Intramodal Retrieval Tables 3 and 4 give intramodal retrieval results enabled by CxC's STS

---

[7] bert_en_uncased_L-12_H-768_A-12/2

Model	**STS** avg ± std	**SIS** avg ± std	**SITS** avg ± std
BoW	55.1±0.6	–	–
$\text{USE}_{\text{mling-large}}$	71.4±0.4	–	–
Inception V3	–	19.6±1.9	–
ResNet-152	–	59.2±1.3	–
SimCLRv2	–	74.3±0.9	–
VSE++	74.4±0.4	73.3±0.9	55.2±1.5
VSRN-github	73.0±0.4	70.1±1.0	60.4±1.3
DE_{T2T}	72.9±0.4	–	–
DE_{I2T}	50.9±0.6	81.3±0.7	61.6±1.4
$\text{DE}_{\text{T2T+I2T}}$	74.2±0.4	74.5±0.9	61.9±1.3

Table 5: Spearman's R Bootstrap Correlation (×100) on MS-COCO 5k test set **using CxC annotations**.

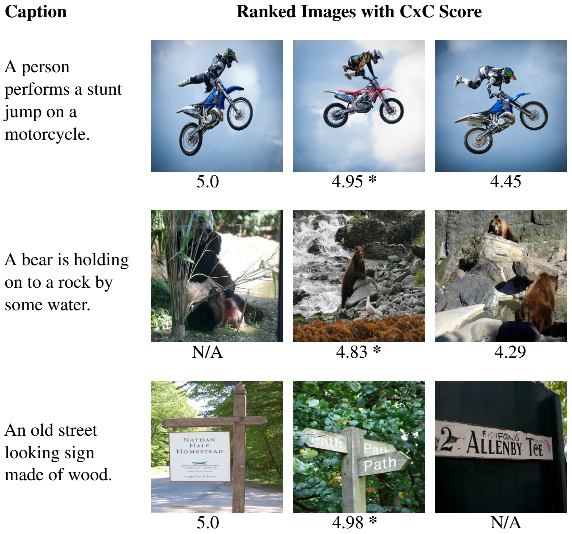
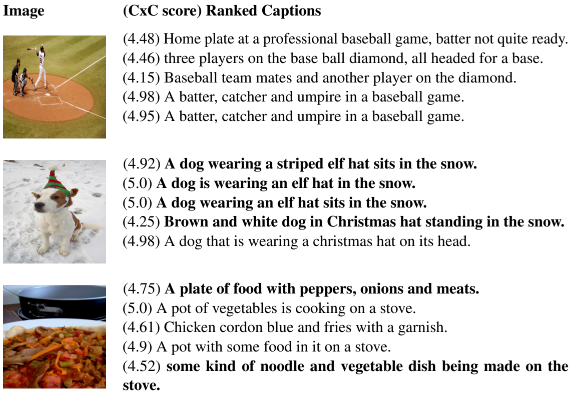


Figure 7: Text→Image retrieval examples (MS-COCO val set), with CxC SITS score provided when available. Images are ranked from left to right, based on $\text{DE}_{\text{T2T+I2T}}$ scores. MS-COCO annotations only consider images marked by * to be correct retrieval results.



(4.48) Home plate at a professional baseball game, batter not quite ready.
(4.46) three players on the base ball diamond, all headed for a base.
(4.15) Baseball team mates and another player on the diamond.
(4.98) A batter, catcher and umpire in a baseball game.
(4.95) A batter, catcher and umpire in a baseball game.

(4.92) **A dog wearing a striped elf hat sits in the snow.**
(5.0) **A dog is wearing an elf hat in the snow.**
(5.0) **A dog wearing an elf hat sits in the snow.**
(4.25) **Brown and white dog in Christmas hat standing in the snow.**
(4.98) A dog that is wearing a christmas hat on its head.

(4.75) **A plate of food with peppers, onions and meats.**
(5.0) A pot of vegetables is cooking on a stove.
(4.61) Chicken cordon blue and fries with a garnish.
(4.9) A pot with some food in it on a stove.
(4.52) **some kind of noodle and vegetable dish being made on the stove.**

Figure 8: Image→Text retrieval results (MS-COCO val set), ranked top to bottom using $\text{DE}_{\text{T2T+I2T}}$ scores. MS-COCO annotations only consider the **bold** captions to be correct results.

and SIS ratings respectively. $\text{USE}_{\text{mling-large}}$ is a strong baseline for Text→Text, but all the cross-modal models beat $\text{USE}_{\text{mling-large}}$ by a wide margin, likely due to learning on in-domain captions. InceptionV3 and ResNet-152 prove surprisingly weak for Image→Image, but SimCLRv2 proves to be a strong unimodal baseline for this task. The cross-modal models nevertheless beat SimCLRv2 by a wide margin, even though none were trained on image-image retrieval directly. In terms of joint intra- and inter-modal learning, the multitask $\text{DE}_{\text{T2T+I2T}}$ model provides strong, balanced performance: it is close to DE_{T2T} for Text→Text and DE_{I2T} for Image→Image and far outperforms the latter for Text→Text. The strong performance is especially notable considering that both DE_{I2T} and $\text{DE}_{\text{T2T+I2T}}$ have the same model capacity.

Semantic Similarity Table 5 shows Spearman's R bootstrapped correlation for all models with respect to CxC's STS, SIS and SITS scores. Overall, VSE++, VSRN-github and $\text{DE}_{\text{T2T+I2T}}$ perform better than unimodal baselines, but interesting further patterns emerge. Despite being much worse for retrieval, VSE++ actually beats VSRN-github on STS and SIS; however, its low SITS score indicates it fails to bridge the two modalities as well. The correlation scores also show that DE_{I2T} is too focused on images: it has the highest SIS (81.3), but has worse STS (50.9) than even BoW (55.1). Adding the text-text loss to DE_{I2T} training, i.e. $\text{DE}_{\text{T2T+I2T}}$, produces much more balanced overall performance. On SIS, SimCLRv2 is stronger than all cross-modal models, except DE_{I2T}. SITS scores appear to rank all models similarly to retrieval (Table 2).

The fact that $\text{DE}_{\text{T2T+I2T}}$ is better than both DE_{T2T} and unimodal baselines for STS and Text→Text retrieval is encouraging, and it demonstrates the value of having a single set of annotations covering the relatedness of a common set of images and captions. We expect that a multitask model which also uses image-image training pairs could demonstrate gains across all tasks—measurements made possible by the CxC annotations (especially the new image-image associations).

7 Conclusion

The CxC dataset provides a much more complete set of relationships between and among images and captions than the raw MS-COCO image-caption pairs. We demonstrate that a dual encoder that learns from both image-caption pairs and caption-

caption pairs ($\mathrm{DE}_{\mathrm{T2T+I2T}}$) exhibits strong, balanced performance across four retrieval tasks and three correlation measures. There is much remaining headroom for future models for all these tasks.

CxC's annotations themselves validate the strong semantic alignment between images and their original captions—these have an average similarity of 4.85. However, we also find that co-captions (captions for the same image) have an average score of just 3.0. This calls into question the use of such pairs in training and evaluating paraphrase generation models (Gupta et al., 2018) and reinforces the need for images as context for human evaluation in paraphrasing (Wang et al., 2019).

Acknowledgments

We thank Julia Hockenmaier for her inputs on CxC's formulation, the Google Data Compute Team, especially Ashwin Kakarla and Mohd Majeed for their tooling and annotation support, Yuan Zhang, Eugene Ie for their comments on the initial versions of the paper and Daphne Luong for executive support for the data collection.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020a. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. 2020b. Big self-supervised models are strong semi-supervised learners. *CoRR*, abs/2006.10029.

Muthu Chidambaram, Yinfei Yang, Daniel Cer, Steve Yuan, Yunhsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Learning cross-lingual sentence representations via a multi-task dual-encoder model. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 250–259, Florence, Italy. Association for Computational Linguistics.

Guillem Collell Talleda and Marie-Francine Moens. 2016. Is an image worth more than a thousand words? on the fine-grain semantic differences between visual and linguistic representations. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 2807–2817. ACL.

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

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.

Fartash Faghri, David J. Fleet, Jamie Ryan Kiros, and Sanja Fidler. 2018. VSE++: improving visual-semantic embeddings with hard negatives. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 12. BMVA Press.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *arXiv preprint arXiv:1811.08008*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of AAAI-18*.

David Harwath and James Glass. 2017. Learning word-like units from joint audio-visual analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers), pages 506–517, Vancouver, Canada. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Gabriel Ilharco, Yuan Zhang, and Jason Baldridge. 2019. Large-scale representation learning from visually grounded untranscribed speech. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Da-Cheng Juan, Chun-Ta Lu, Zhen Li, Futang Peng, Aleksei Timofeev, Yi-Ting Chen, Yaxi Gao, Tom Duerig, Andrew Tomkins, and Sujith Ravi. 2020. Ultra fine-grained image semantic embedding. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 277–285, New York, NY, USA. Association for Computing Machinery.

Andrej Karpathy and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Jamie Kiros, William Chan, and Geoffrey Hinton. 2018. Illustrative language understanding: Large-scale visual grounding with image search. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 922–933.

Oier Lopez de Lacalle, Ander Salaberria, Aitor Soroa, Gorka Azkune, and Eneko Agirre. 2020. Evaluating multimodal representations on visual semantic textual similarity. In *Proceedings of the Twenty-third European Conference on Artificial Intelligence (ECAI)*, Santiago Compostela, Spain.

Kunpeng Li, Yulun Zhang, Kai Li, Yuanyuan Li, and Yun Fu. 2019. Visual semantic reasoning for image-text matching. In *The IEEE International Conference on Computer Vision (ICCV)*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13–23. Curran Associates, Inc.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *Image*, 2:T2.

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. 2010. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147. Association for Computational Linguistics.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Mingxing Tan and Quoc Le. 2019a. EfficientNet: Rethinking model scaling for convolutional neural networks. volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114, Long Beach, California, USA. PMLR.

Mingxing Tan and Quoc Le. 2019b. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114.

Su Wang, Rahul Gupta, Nancy Chang, and Jason Baldridge. 2019. A task in a suit and a tie: Paraphrase generation with semantic augmentation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7176–7183. AAAI Press.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Yinfei Yang, Gustavo Hernandez Abrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bidirectional dual encoder with additive margin softmax. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5370–5378. International Joint Conferences on Artificial Intelligence Organization.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. 2018. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

CxC Annotations - Collection and Analysis

We present additional details of the human annotation process. To annotate the CxC dataset, in house annotators were employed: 170 (74 men, 96 women) for STS, 61 (28 men, 33 women) for SIS and 113 (46 men, 67 women) for SITS. All were aged between 20-35 years. The annotators were paid hourly wages that are competitive for their locale. They have standard rights as contractors. They were fluent English speakers.

We define separate annotation interfaces for each of Semantic Textual Similarity (STS), Semantic Image Similarity (SIS) and Semantic Image Text Similarity (SITS) tasks. We define a similarity scale ranging from 0 to 5 for all three tasks, following Cer et al. (2017).

We conducted a few pilot annotation rounds with the annotators to evaluate the effectiveness of the annotation instructions and the annotation interface. We learned that allowing the annotators to rate on a continuous 0-5 scale instead of a discrete one like STS resulted in higher correlation between the individual ratings. As a result, we decided to use the continuous ratings in the task but still keep the similarity definition for each discrete value in the annotation instructions. The final annotation interfaces are illustrated in Figures 4 for STS, 5 for SIS and 6 for SITS. Task-specific high-level instructions are displayed at the top in an expandable text box followed by a pair of examples. At the bottom there is a sliding bar with 0-5 score instructions along the scale. Since the SITS instructions are longer, they are shown when the annotator hovers over the corresponding score to improve readability for this task.

Each annotator is required to evaluate the displayed example based on the instructions and score them. The sliding scale makes it intuitive for the annotators to rate an example 2.87 if they feel the semantic similarity of the pair lies between score descriptions of 2 and 3, leaning towards 3. Finally, the annotator response is recorded when they click the submit button at the bottom of the page. The absolute score is deliberately not displayed so as not to distract the workers towards trying to get a clean integer value like 3.0 instead of 2.94 or 2.97.

The annotators were able to get a better grasp of the task through the pilot annotations and got quicker at scoring the pairs. They took an average of 37, 17 and 17 seconds per example for STS, SIS and SITS tasks respectively for the final round of annotations. Table 2 describes the instructions shared with the annotation workers for each task. The side-by-side comparison shows how each rating on the SIS and SITS scales compares to the STS benchmark. Figure 1 shows a set of SIS and SITS examples for the 0-5 rating scale shared along with the instructions. Table 1 contains the breakdown of the number of annotations per task per split.

Split \ Task	STS	SIS	SITS	Total (per split)
Validation	44,009	42,767	44,722	131,498
Test	44,045	46,719	44,833	135,597
Total (per task)	88,054	89,486	89,555	267,095

Table 1: Number of annotations per task and split.

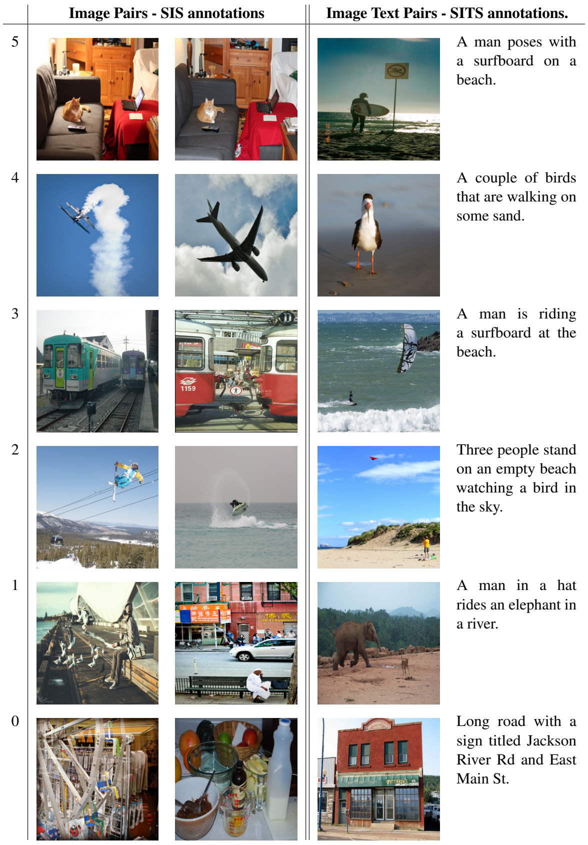


Figure 1: Examples for each annotation score (0-5) of SIS (left) and SITS (right) tasks.

Figure 2 shows a distribution of the standard deviation of raw annotations for each item per task. For STS, there is larger overall deviation compared to the other two tasks–it seems that pairs of short captions leave more ambiguity and are open for broader interpretation than when at least one image is involved. Note also that SITS is expected to have lower deviation because of the sampling based on STS and SIS annotations.

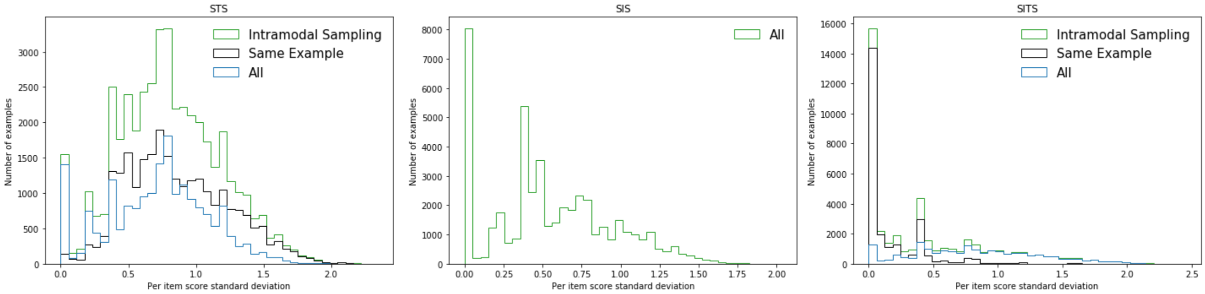


Figure 2: Standard deviation per item in CxC. *Intramodal Sampling* refers to examples selected using *other-modality* selection, *Same Example* refers to original MS-COCO pairs, and *All* covers all examples for a task.

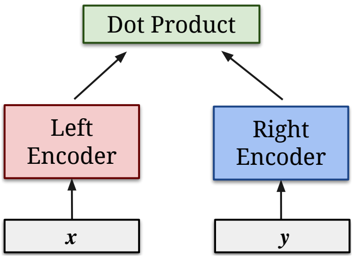


Figure 3: Dual Encoder with inputs x and y, encoded by the left and right encoders, respectively. Similarity is computed as the dot product of the encodings.

Training Setup

Figure 3 shows the basic architecture of the dual encoder models from Section 5.2, which establish strong baselines on all the retrieval and correlation tasks. The image encoder and text encoder are both pre-trained in all experiments. Following Ilharco et al. (2019), we pretrain our dual encoders on the Conceptual Captions dataset (Sharma et al., 2018) with image-to-caption and caption-to-image losses. Conceptual Captions contains 3.3 million pairs of images and captions—far larger than MS-COCO. Pre-training uses the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) and a learning rate that starts at 1e-4 and decays by 0.1% every 1000 steps. We stop pre-training after $\approx$30k steps and select the checkpoint that maximizes R@10 on a held-out set. We then fine-tune this checkpoint on MS-COCO using the same hyper parameters, except for a smaller learning rate of 5e-6.

Our models are trained on 32-core slices of Cloud TPU V3 pods, with a per-replica batch size of $K = 64$ during both pre-training and fine-tuning. Because in-batch sampled softmax loss is known to perform best when computed over a large number of negative samples (Gillick et al., 2018), our training setup pools image and caption encodings from all replicas before computing the loss. That is, each replica computes l and r for its local mini-batch and broadcasts them to all others to be used as negative samples. Training with N cores thus allows the loss to be computed over the global batch of $N \cdot K$ examples and $(N \cdot K)^2$ pairs (in our case 2048 examples and 2048^2 example pairs).

Ablation Experiments

Our model architecture and training setup differ from prior work in key ways. In particular, best known results for VSE++ and VSRN are from models that were trained with much smaller batch sizes, did not undergo Conceptual Captions pre-training, and had different image encoder architectures. To evaluate the effect of these factors, we trained variants of our DE_{I2T} model (here, the *baseline* training recipe) with the following one-off ablations:

- The *small batch size* ablation reduces the training batch size to 128 examples, to match that of VSE++ and VSRN in (Faghri et al., 2018) and (Li et al., 2019), respectively.
- *No pretraining* skips dual encoder pretraining on Conceptual Captions.
- *ResNet-152* uses the same recipe as *baseline*, but replaces the EfficientNet-B4 image encoder with ResNet-152, which was used in VSE++. Notably, EfficientNet-B4 has fewer parameters than ResNet-152, but achieves higher classification accuracy on ImageNet.

Table 3 summarizes the performance of the ablated models. Reducing the batch size causes a small but consistent reduction in recalls across all tasks. Removing Conceptual Captions pretraining leads to larger regressions on all tasks – except on Text-Text retrieval, where results are curiously better than the baseline. Likewise, models using ResNet-152 image encoders perform worst overall,

Score	Semantic Textual Similarity (STS)	Semantic Image Similarity (SIS)	Semantic Image-Text Similarity (SITS)
5	**The texts are completely equivalent** as they mean the same thing.	**The scenes are near duplicates**, possibly being viewed from a different perspective.	**The image and sentence are perfectly matched.** The sentence is an almost perfect description for the image.
4	**The texts are mostly equivalent** but some unimportant details differ.	**The two scenes are mostly equivalent**, but some unimportant details differ such as involving different but the same or highly similar types of participants, actions, objects and background.	**The image and sentence are mostly matched**, but some unimportant details differ such as involving different but the same or highly similar types of participants, actions, objects and background. The text can partially describe the image.
3	**The texts are roughly equivalent** but some important information differs or is missing.	**The two scenes are roughly equivalent**, but some important details are different or missing such as involving a notable difference in the types of participants, actions, objects or background.	**The image and sentence are roughly matched**, but some important details are different or missing such as involving a notable difference in the types of participants, actions, objects or background. The image cannot be described using the text.
2	**The texts are not equivalent** but share some details.	**The two scenes are not equivalent**, but share some details in terms of the types of participants, actions, objects or background.	**The image and sentence are not matched**, but share some details in one or more of the types of participants, actions, objects or background.
1	**The texts are not equivalent** but are on the same topic.	**The two scenes are not equivalent**, but are loosely thematically related.	**The image and sentence are not matched**, but are loosely thematically related.
0	**The texts are on different topics.**	**The two scenes are completely dissimilar.**	**The image and sentence are completely unmatched.**

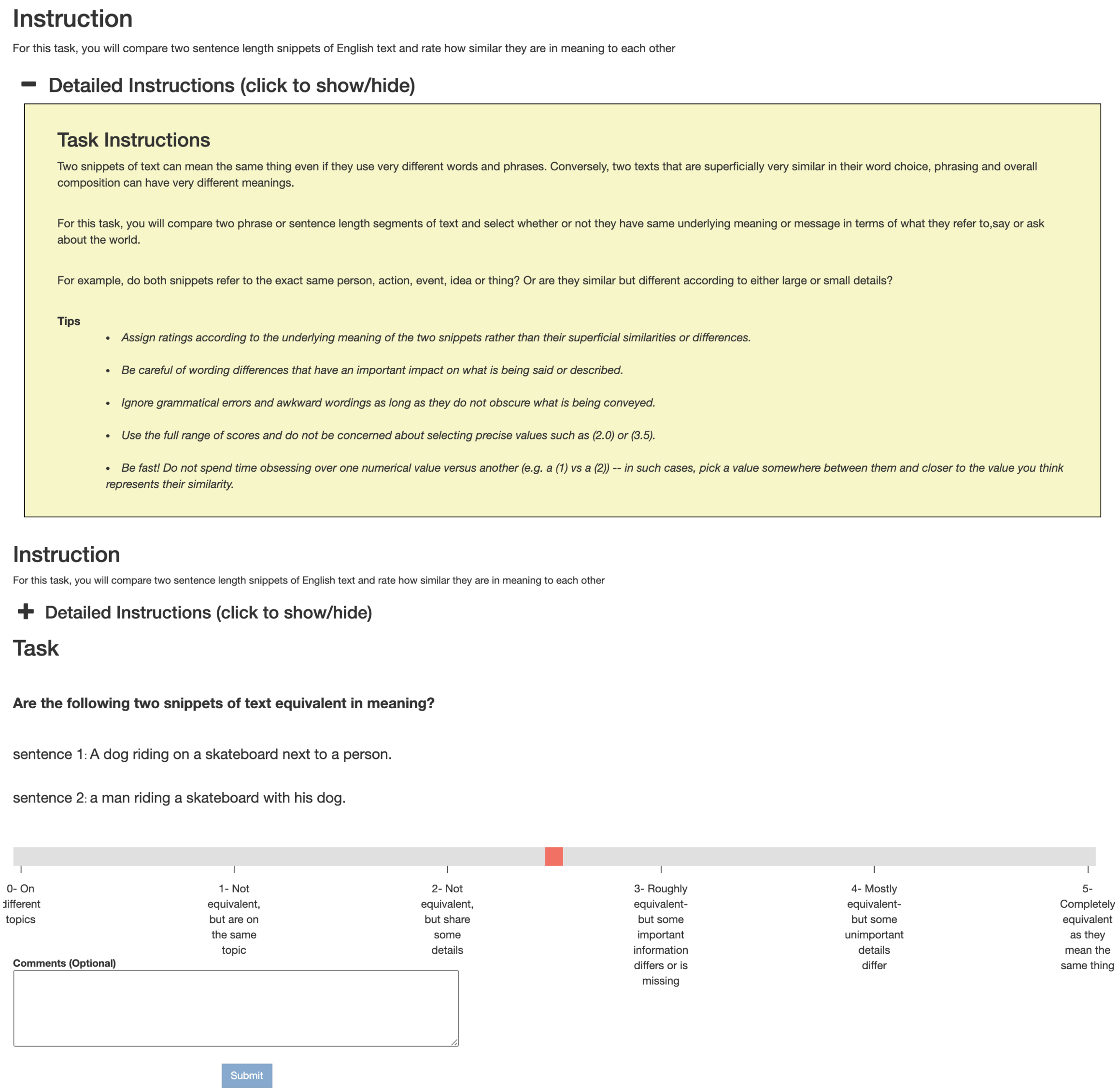
Table 2: Intramodality annotation criteria for Semantic Image Similarity (SIS) and Intermodality annotation criteria for Semantic Image-Text Similarity (SITS) with comparison to equivalent Semantic Textual Similarity (STS) annotations (Agirre et al., 2012).

but also perform (slightly) better than the baseline on Text-Text retrieval.

Overall, we conclude that pretraining and choice of image encoder architecture have large effects on model performance; large-batch training is beneficial, but has a smaller impact. Finally, the asymmetric shifts in task performance suggest models make implicit trade-offs based on the relative difficulty of each task – here, apparently, a function of encoder strength and quantity of training data. Understanding these dynamics, and building models that perform well across all tasks, requires future study. Crisscrossed Captions enables such work by giving a more complete picture of model quality on both intra- and inter-modal tasks.

Annotations	Model	Image → Text R@1	Image → Text R@10	Text → Image R@1	Text → Image R@10	Text → Text R@1	Text → Text R@10	Image → Image R@1	Image → Image R@10
MS-COCO	DE_{I2T} (baseline)	52.0	89.5	37.9	78.8	25.9	57.2	–	–
	DE_{I2T} (small batch size)	49.6	88.2	35.7	78.0	25.3	57.6	–	–
	DE_{I2T} (no pretraining)	45.0	86.0	31.2	74.7	34.5	67.2	–	–
	DE_{I2T} (ResNet-152)	43.5	83.0	28.9	71.4	28.2	60.2	–	–
CxC	DE_{I2T} (baseline)	53.9	91.2	39.8	80.9	26.0	57.5	38.3	85.0
	DE_{I2T} (small batch size)	51.8	90.1	37.7	80.3	25.4	57.9	38.0	84.3
	DE_{I2T} (no pretraining)	47.0	88.1	33.2	77.6	34.6	67.6	37.0	84.0
	DE_{I2T} (ResNet-152)	45.2	85.1	30.8	74.4	28.3	60.5	29.7	76.5

Table 3: Ablation analysis for DE_{I2T}, retrieval results on MS-COCO 5k test set and CxC. (Note that MS-COCO does not support Image → Image retrieval evaluation at all.)



Instruction

For this task, you will compare two sentence length snippets of English text and rate how similar they are in meaning to each other

− Detailed Instructions (click to show/hide)

Task Instructions

Two snippets of text can mean the same thing even if they use very different words and phrases. Conversely, two texts that are superficially very similar in their word choice, phrasing and overall composition can have very different meanings.

For this task, you will compare two phrase or sentence length segments of text and select whether or not they have same underlying meaning or message in terms of what they refer to,say or ask about the world.

For example, do both snippets refer to the exact same person, action, event, idea or thing? Or are they similar but different according to either large or small details?

Tips

- *Assign ratings according to the underlying meaning of the two snippets rather than their superficial similarities or differences.*
- *Be careful of wording differences that have an important impact on what is being said or described.*
- *Ignore grammatical errors and awkward wordings as long as they do not obscure what is being conveyed.*
- *Use the full range of scores and do not be concerned about selecting precise values such as (2.0) or (3.5).*
- *Be fast! Do not spend time obsessing over one numerical value versus another (e.g. a (1) vs a (2)) -- in such cases, pick a value somewhere between them and closer to the value you think represents their similarity.*

Instruction

For this task, you will compare two sentence length snippets of English text and rate how similar they are in meaning to each other

+ Detailed Instructions (click to show/hide)

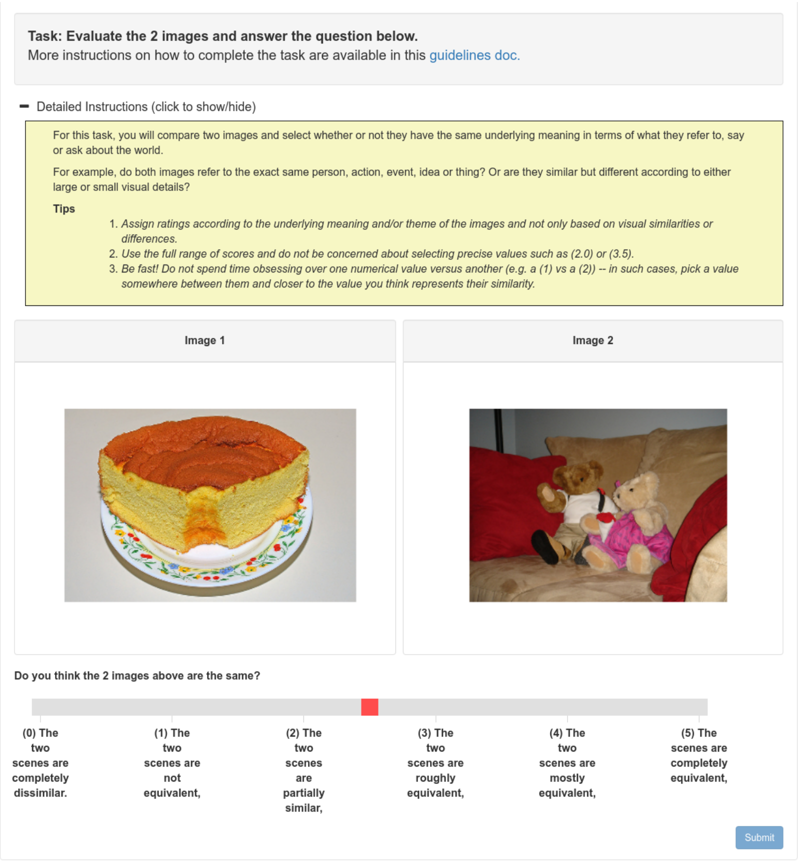
Task

Are the following two snippets of text equivalent in meaning?

sentence 1: A dog riding on a skateboard next to a person.

sentence 2: a man riding a skateboard with his dog.

Figure 4: STS Annotation Interface



Task: Evaluate the 2 images and answer the question below.
More instructions on how to complete the task are available in this guidelines doc.

Detailed Instructions (click to show/hide)

For this task, you will compare two images and select whether or not they have the same underlying meaning in terms of what they refer to, say or ask about the world.

For example, do both images refer to the exact same person, action, event, idea or thing? Or are they similar but different according to either large or small visual details?

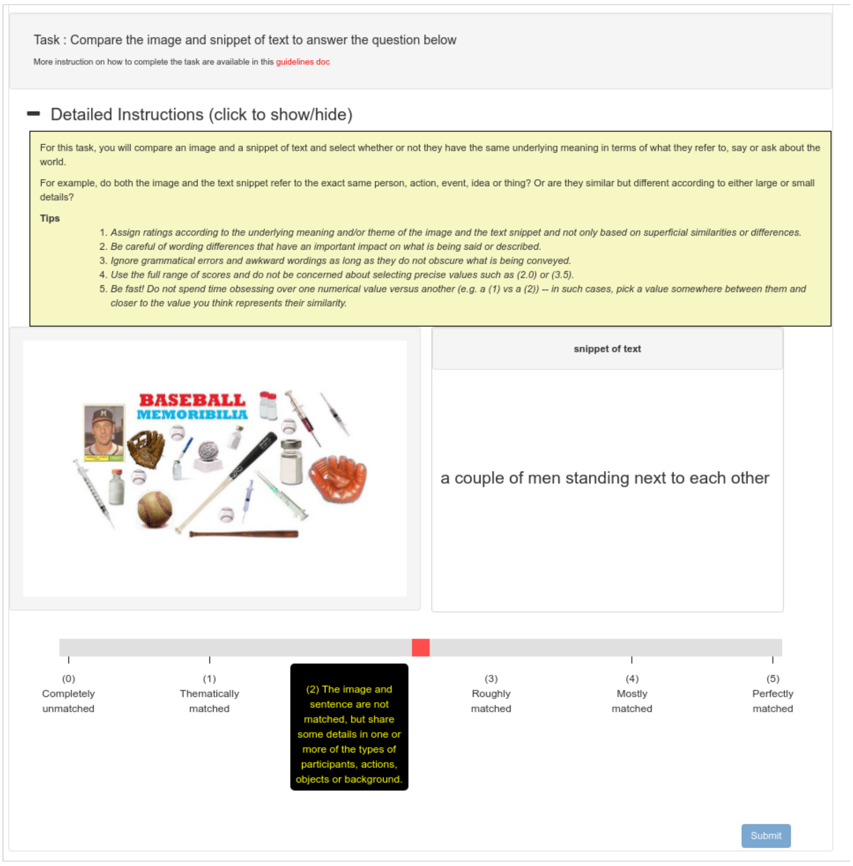
Tips

1. *Assign ratings according to the underlying meaning and/or theme of the images and not only based on visual similarities or differences.*
2. *Use the full range of scores and do not be concerned about selecting precise values such as (2.0) or (3.5).*
3. *Be fast! Do not spend time obsessing over one numerical value versus another (e.g. a (1) vs a (2)) -- in such cases, pick a value somewhere between them and closer to the value you think represents their similarity.*

Do you think the 2 images above are the same?

(0) The two scenes are completely dissimilar.
(1) The two scenes are not equivalent,
(2) The two scenes are partially similar,
(3) The two scenes are roughly equivalent,
(4) The two scenes are mostly equivalent,
(5) The scenes are completely equivalent,

Figure 5: SIS Annotation Interface



Task : Compare the image and snippet of text to answer the question below

More instruction on how to complete the task are available in this guidelines doc

Detailed Instructions (click to show/hide)

For this task, you will compare an image and a snippet of text and select whether or not they have the same underlying meaning in terms of what they refer to, say or ask about the world.

For example, do both the image and the text snippet refer to the exact same person, action, event, idea or thing? Or are they similar but different according to either large or small details?

Tips

1. *Assign ratings according to the underlying meaning and/or theme of the image and the text snippet and not only based on superficial similarities or differences.*
2. *Be careful of wording differences that have an important impact on what is being said or described.*
3. *Ignore grammatical errors and awkward wordings as long as they do not obscure what is being conveyed.*
4. *Use the full range of scores and do not be concerned about selecting precise values such as (2.0) or (3.5).*
5. *Be fast! Do not spend time obsessing over one numerical value versus another (e.g. a (1) vs a (2)) -- in such cases, pick a value somewhere between them and closer to the value you think represents their similarity.*

a couple of men standing next to each other

(0) Completely unmatched
(1) Thematically matched
(2) The image and sentence are not matched, but share some details in one or more of the types of participants, actions, objects or background.
(3) Roughly matched
(4) Mostly matched
(5) Perfectly matched

Figure 6: SITS Annotation Interface

On-Device Text Representations Robust To Misspellings via Projections

Chinnadhurai Sankar
Mila, Université de Montréal[*]
Montreal, QC, Canada
`chinnadhurai@gmail.com`

Sujith Ravi
Amazon Alexa[†]
Sunnyvale, CA, USA
`sravi@sravi.org`

Zornitsa Kozareva
Google
Mountain View, CA, USA
`zornitsa@kozareva.com`

Abstract

Recently, there has been a strong interest in developing natural language applications that live on personal devices such as mobile phones, watches and IoT with the objective to preserve user privacy and have low memory. Advances in Locality-Sensitive Hashing (LSH)-based projection networks have demonstrated state-of-the-art performance in various classification tasks without explicit word (or word-piece) embedding lookup tables by computing on-the-fly text representations.

In this paper, we show that the projection based neural classifiers are inherently robust to misspellings and perturbations of the input text. We empirically demonstrate that the LSH projection based classifiers are more robust to common misspellings compared to BiLSTMs (with both word-piece & word-only tokenization) and fine-tuned BERT based methods. When subject to misspelling attacks, LSH projection based classifiers had a small average accuracy drop of 2.94% across multiple classifications tasks, while the fine-tuned BERT model accuracy had a significant drop of 11.44%.

1 Introduction

At the core of Natural Language Processing (NLP) neural models are pre-trained word embeddings like Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014a) and ELMo (Peters et al., 2018). They help initialize the neural models, lead to faster convergence and have improved performance for numerous application such as Question Answering (Liu et al., 2018), Summarization (Cheng and Lapata, 2016), Sentiment Analysis (Yu et al., 2017). While word embeddings are powerful in unlimited constraints such as computation power and compute resources, it becomes challenging to deploy them to on-device due to their huge size.

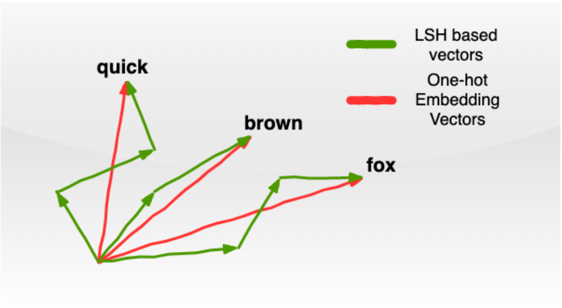


Figure 1: One-hot word embedding look-up vectors vs linear combination of LSH projection based vectors (Ravi, 2017) representing the same word.

This led to interesting research by (Ravi and Kozareva, 2018; Sankar et al., 2019), who showed that word embeddings can be replaced with lightweight binary Locality-Sensitive Hashing (LSH) based projections learned on-the-fly. The projection approach surmounts the need to store any embedding matrices, since the projections are dynamically computed. This further enables user *privacy* by performing inference directly on device without sending user data (e.g., personal information) to the server. The embedding memory size is reduced from $O(V)$ to $O(K)$, where V is the token vocabulary size and $K << V$, is the binary LSH projection size. The projection representations can operate on either word or character level, and can be used to represent a sentence or a word depending on the NLP application. For instance, recently the Projection Sequence Networks (ProSeqo) (Kozareva and Ravi, 2019) used BiLSTMs over word-level projection representations to represent long sentences and achieved close to state-of-the-art results in both short and long text classification tasks with varying amounts of supervision and vocabulary sizes.

---

[*] Work done during internship at Google
[†] Work done while at Google AI

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2871–2876
April 19 - 23, 2021.

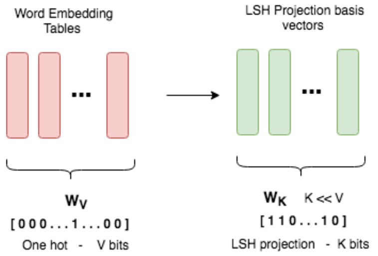


Figure 2: Memory for V look-up vectors for each token vs storing $K(<< V)$ vectors and linearly combining them for token representation. We consider $K = 1120$ following (Ravi and Kozareva, 2018) in this paper.

Despite being successful, there are no existing systematic research efforts focusing on evaluating the capabilities of the LSH based projection for text representations. To that end, we empirically analyze the effectiveness and robustness of the LSH projection approach for text representation by conducting two types of studies in this paper.

1. *Classification with perturbed inputs*, where we show that Projection based networks 1) Projection Sequence Networks (ProSeqo) (Kozareva and Ravi, 2019) and 2) Self-Governing Neural Networks (SGNN) models (Ravi and Kozareva, 2019) evaluated with perturbed LSH projections are robust to misspellings and transformation attacks, while we observe significant drop in performance for BiLSTMs and fine-tuned BERT classifiers.

2. *Perturbation Analysis*, where we test the robustness of the projection approach by directly analyzing the changes in representations when the input words are subject to the char misspellings. The purpose of this study is to examine if the words or sentences with misspelling are nearby in the projection space instead of frequently colliding with the projection representations of other valid words.

Overall, our studies showcase the robustness of LSH projection representations and resistance to misspellings. Due to their effectiveness, we believe that in the future, text representations using LSH projections can go beyond memory constrained settings and even be exploited in large scale models like Transformers (Vaswani et al., 2017).

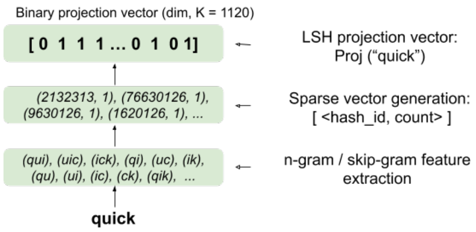


Figure 3: Binary Locality-Sensitive Hashing (LSH) projection representation for text.

2 Binary LSH projections for text representations

The dependency on vocabulary size V, is one of the primary reasons for the huge memory footprint of embedding matrices. It is common to represent a token, x by one-hot representation, $\mathbb{Y}(x) \in [0,1]^V$ and a distributed representation of the token is obtained by multiplying the one-hot representation with the embedding matrix, $W_V \in \mathbb{R}^{d \times V}$ as in

$$U_V(x) = W_V * \mathbb{Y}(x)^\top \in \mathbb{R}^d$$

One way to remove the dependency on the vocabulary size is to learn a smaller matrix, $W_K \in \mathbb{R}^{d \times K}$ ($K << V$), as shown in Figure 2. For instance, 300-dimensional Glove embeddings, W_V (Pennington et al., 2014b) with 400k vocabulary size occupies $>$ 1 GB while the W_K occupies only $\approx$ 1.2 MB for $K = 1000$ yielding a $1000\times$ reduction in size. Instead of learning a unique vector for each token in the vocabulary, we can think of the columns of this W_K matrix as a set of basis vectors and each token can be represented as a linear combination of basis vectors in W_K as in Figure 1. We select the basis vectors from W_K for each token with a fixed K-bit binary vector instead of a V-bit one-hot vector.

The LSH *Projection function*, $\mathbb{P}$ (Figure 3)(Ravi, 2017, 2019) used in SGNN (Ravi and Kozareva, 2018) and ProSeqo (Kozareva and Ravi, 2019) does exactly this as it dynamically generates a fixed binary projection representation, $\mathbb{P}(x) \in [0,1]^K$ for any token, x by extracting morphological input features like char (or token) n-gram & skip-gram features, parts of speech tags etc. from x and a modified Locality-Sensitive Hashing (LSH) based transformation, $\mathbb{L}$ as in

$$x \xrightarrow{\mathbb{F}} [f_1, \cdots, f_n] \xrightarrow{\mathbb{L}} \mathbb{P}(x) \in [0,1]^K$$

where $\mathbb{F}$ extracts n-grams (or skip-grams), $[f_1, \cdots, f_n]$ from the input text. Here, $[f_1, \cdots, f_n]$

could refer to either character level or token level n-grams(or skip-grams) features. Given the LSH projection representation, $\mathbb{P}(x)$, the distributed representation of the token, x is represented as in

$$U_W(x) = W_K * \mathbb{P}(x)^\top \in \mathbb{R}^d$$

. It is worth noting that projection operation, $\mathbb{P}$ can also be used map an entire sentence directly to the $[0,1]^K$ space.

As for the projection based classifiers, the ProSeqo model (Kozareva and Ravi, 2019) runs a BiLSTM over word-level binary LSH projection representations to predict the correct classes, while the SGNN model (Ravi and Kozareva, 2018) computes a binary LSH projection representation for the entire input text, followed by a 2-layer MLP and a softmax layer on top of it for class prediction. SGNN was designed for short text, while ProSeqo is also suitable for long text classification tasks.

There have been a number of research efforts (Sakaguchi et al., 2016; Edizel et al., 2019; Pruthi et al., 2019) to improve the robustness of neural classifiers to misspelling attacks and other text transformations. Recently, Pruthi et al. (2019) observe that fine-tuned BERT and BiLSTM based models are very brittle (for e.g., accuracy drops from 90.3% to 45.8% in the SST (Socher et al., 2013) classification task) to adversarial misspelling attacks. Contrary to intuition, they observe that word-piece and character-level models are more susceptible to spelling attacks compared to the word-level models.

The LSH projection operation, $\mathbb{P}$, is a function of n-grams (and skip-grams) of the input text x and usually the fraction of n-grams affected by spelling attacks tend to be minimal resulting in insignificant changes to the projection representation, $\mathbb{P}(x)$. Therefore, we hypothesize that the projection based models like ProSeqo, SGNN, etc. are *inherently* robust to commonly occurring spelling attacks. In the following sections, we investigate the robustness of projection based classifiers by subjecting them to common misspellings, followed by an analysis of changes in the binary LSH projections of input text under such transformations.

3 Effect of Misspellings on Text Classification

We study the robustness of two types of projection based models – ProSeqo and SGNN. On the other hand, we fine-tune the pretrained BERT-base model (with word-piece tokenization) (Devlin et al., 2018) and train two-layer BiLSTMs (with both word-only and word-piece tokenization) for comparable accuracies with respect to the projection based models for a fair comparison. By word-only tokenization, we mean that models encode input words using a lookup table for each word. In our setup, we test the robustness of the neural classifiers by subjecting the corresponding test sets to common misspellings and omissions. We consider the following perturbation operations: randomly dropping, inserting, and swapping internal characters within words of the input sentences (Gao et al., 2018; Pruthi et al., 2019) [1]. We decide to perturb each word in a sentence with a fixed probability, $P_{perturb}$. Following (Ravi and Kozareva, 2018), we fix the projection dimension to $K = 1120$.

3.1 Datasets

For evaluation purposes, we use the following text classification datasets for dialog act classification MRDA (Shriberg et al., 2004) and SWDA (Godfrey et al., 1992; Jurafsky et al., 1997), for intent prediction ATIS (Tür et al., 2010) and long text classification Amazon Reviews (Zhang et al., 2015) and Yahoo! Answers (Zhang et al., 2015). Table 1 shows the characteristics of each dataset.

Tasks	**# Classes**	**Avg-len**	**Train**	**Test**
ATIS (Dialog act)	21	11	$4.4k$	$0.89k$
MRDA (Dialog act)	6	8	78k	15k
SWDA (Intent Prediction)	42	7	193k	5k
YAHOO (Answers Categorization)	10	108	1400k	60k
AMAZON (Review Prediction)	5	92	3000k	650k

Table 1: Classification Dataset Characteristics

Accuracy drop (%) ± std-deviation (over 5 runs)				
Datasets → **Models ↓**	**MRDA**	**ATIS**	**YAHOO**	**AMAZON**
BERT-base	$8.25_{\pm 3.4}$	$15.57_{\pm 1.2}$	$9.06_{\pm 5.4}$	$12.88_{\pm 3.8}$
BiLSTM-wp	$8.91_{\pm 3.9}$	$20.11_{\pm 3.1}$	$11.32_{\pm 4.5}$	$9.46_{\pm 1.3}$
BiLSTM-w	$15.14_{\pm 4.3}$	$16.23_{\pm 2.6}$	$9.32_{\pm 2.3}$	$8.88_{\pm 2.4}$
SGNN	$\mathbf{1.91}_{\pm \mathbf{0.5}}$	$\mathbf{2.80}_{\pm \mathbf{0.3}}$	-	-
ProSeqo	$2.11_{\pm 0.4}$	$2.84_{\pm 0.6}$	$\mathbf{3.11}_{\pm \mathbf{0.3}}$	$\mathbf{3.91}_{\pm \mathbf{0.5}}$

Table 2: Each entry in the table denotes the average drop in accuracy(%) when classifiers are subject to test inputs with misspellings, $P_{perturb} = 0.2$. ATIS and MRDA are smaller datasets, while Yahoo! Answers and Amazon reviews are larger. BERT-base refers to the BERT-base (Devlin et al., 2018) word-piece model fine-tuned to individual tasks and it converged to state-of-the-art for all tasks.

---

[1] Further details on the perturbation operations and training details necessary for reproducibility are presented in the supplementary material

Accuracy(%) (Averaged over 5 runs)									
Datasets	**MRDA**			**ATIS**			**SWDA**		
Perturb(%)	**BiLSTM-wp**	**BiLSTM-w**	**SGNN**	**BiLSTM-wp**	**BiLSTM-w**	**SGNN**	**BiLSTM-wp**	**BiLSTM-w**	**SGNN**
0	79.23	78.14	87.22	91.73	92.04	93.51	72.92	72.94	76.21
Perturbation operation: *drop*									
20	$69.46_{\pm1.1}$	$74.96_{\pm0.7}$	$85.43_{\pm0.3}$	$81.95_{\pm2.2}$	$80.15_{\pm2.0}$	$91.05_{\pm0.2}$	$65.05_{\pm4.2}$	$64.95_{\pm4.1}$	$70.76_{\pm1.1}$
40	$72.34_{\pm2.9}$	$55.17_{\pm3.1}$	$84.62_{\pm0.22}$	$71.69_{\pm2.9}$	$65.88_{\pm2.1}$	$91.86_{\pm0.3}$	$61.24_{\pm4.1}$	$64.95_{\pm5.2}$	$67.79_{\pm1.2}$
60	$69.81_{\pm4.7}$	$42.25_{\pm4.2}$	$83.27_{\pm0.25}$	$59.16_{\pm4.8}$	$56.25_{\pm3.9}$	$90.12_{\pm0.4}$	$57.48_{\pm5.7}$	$58.77_{\pm5.8}$	$63.21_{\pm1.3}$
Perturbation operation: *swap*									
20	$78.25_{\pm1.1}$	$71.34_{\pm2.2}$	$86.74_{\pm0.1}$	$86.12_{\pm1.7}$	$85.05_{\pm1.8}$	$92.05_{\pm0.3}$	$66.27_{\pm3.3}$	$64.52_{\pm2.1}$	$70.84_{\pm0.3}$
40	$75.91_{\pm3.9}$	$69.22_{\pm2.1}$	$86.39_{\pm0.2}$	$82.06_{\pm2.7}$	$78.04_{\pm2.9}$	$91.15_{\pm0.2}$	$62.67_{\pm5.8}$	$54.93_{\pm2.4}$	$67.22_{\pm0.3}$
60	$69.22_{\pm3.8}$	$66.91_{\pm4.0}$	$85.99_{\pm0.2}$	$72.34_{\pm3.3}$	$68.54_{\pm4.1}$	$91.27_{\pm0.3}$	$59.20_{\pm4.8}$	$47.93_{\pm4.3}$	$64.48_{\pm0.3}$
Perturbation operation: *all*									
20	$72.96_{\pm1.3}$	$73.39_{\pm2.3}$	$86.71_{\pm0.4}$	$80.40_{\pm1.7}$	$83.55_{\pm2.1}$	$92.83_{\pm0.2}$	$60.49_{\pm4.3}$	$64.28_{\pm3.3}$	$68.96_{\pm0.2}$
40	$70.32_{\pm3.4}$	$63.04_{\pm4.3}$	$85.31_{\pm0.5}$	$71.62_{\pm3.1}$	$75.81_{\pm2.6}$	$90.71_{\pm0.3}$	$54.96_{\pm4.1}$	$59.46_{\pm4.6}$	$65.44_{\pm0.4}$
60	$67.64_{\pm5.7}$	$55.50_{\pm5.3}$	$84.21_{\pm0.5}$	$61.10_{\pm6.1}$	$66.58_{\pm5.6}$	$88.35_{\pm0.3}$	$49.62_{\pm6.7}$	$51.85_{\pm6.3}$	$64.97_{\pm0.5}$

Table 3: Comparison of projection based models vs BiLSTMs subject to various types and amounts of perturbations. BiLSTM-wp and BiLSTM-w refer to models with word-piece and word-only tokenization respectively.

3.2 Experiments and Results

Table 2 reports the average classifier accuracy drops when all the models are subject to all types of perturbations (*swap*, *drop*, & *add*) on multiple classification tasks (two short text and two long text). We see that the accuracy drop for the projection based models is significantly lower across all datasets. It is also worth noting that the standard deviations across the 5 runs are also minimal for the projection based models further showcasing the stability of projection representations.

In another experiment shown in Table 3, we subject different models to varying types and amounts of perturbations. Similarly, we see that the accuracy drop for the projection based models is the smallest across all datasets and amounts of perturbation. Compared to the word-only models, we observe that the word-piece models are also comparably susceptible to character perturbations which agrees with the findings in (Pruthi et al., 2019).

4 Perturbation Analysis

Apart from the classification experiments, we also directly analyze the changes in the binary LSH projection representations by subjecting input text to different types and amount of perturbations. To that end, we take a large corpus ***enwik9***[2] (vocabulary size of $500k$ and $129M$ words) to analyze the average Hamming distance between LSH projections of the words in the corpus. Next, we compute the average changes in the projection representations by subjecting them to the character perturbations from Section 3. Table 4 shows the results. We make the following observations from our experiments:

LSH Proj.Dim (K)	**Character Perturbations**	
	5%	10%
840	10.08	24.33
980	15.48	31.24
1120	18.83	33.65
1260	19.71	39.01

Table 4: Avg. changes in word projections (bits) for different Char Perturbation % in *enwik9* corpus.

1. Average Hamming distance between LSH projections of words is $\approx K/2$, where K is the projection dimension which implies that the words are more or less uniformly spread out from each other indicating that there are no bias issues in the $[0, 1]^K$ representation space.
2. Assuming $P_{perturb} = 0.2$, we observe that LSH projection changes only by $\approx 11\%$ w.r.t the average Hamming distance between the words in the corpus when subject to misspellings. For instance, if the average Hamming distance between LSH projections of words is 100 bits, misspellings change the projections by only 11 bits on average. Intuitively, this suggests that neural layers on top of the LSH projection tend to rarely confuse a misspelled word for another valid word.

Also from Table 4, we found that the changes in the LSH-projection, $\Delta_{\mathbb{P}(x)}$ due to perturbations is directly proportional to LSH projection dimension, K and perturbation probability, $P_{perturb}$ as in, $\Delta_{\mathbb{P}(x)} \propto K \cdot P_{perturb}$.

5 Conclusion

In this work, we perform a detailed study analyzing the robustness of recent LSH-based projection

---

[2] *enwik9* is a byte-level dataset consisting of the first 10^9 bytes of the English Wikipedia XML dump, http://mattmahoney.net/dc/textdata.html.

neural networks for memory-efficient text representations. Based on multiple text classification tasks and perturbation studies, we find projection-based neural models to be robust to text transformations compared to BERT or BiLSTMs with embedding lookup tables for words and word-pieces.

References

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Bora Edizel, Aleksandra Piktus, Piotr Bojanowski, Rui Ferreira, Edouard Grave, and Fabrizio Silvestri. 2019. Misspelling oblivious word embeddings. *CoRR*, abs/1905.09755.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 50–56.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE International Conference on Acoustics, Speech and Signal Processing - Volume 1*, ICASSP'92, pages 517–520. IEEE Computer Society.

Daniel Jurafsky, Rebecca Bates, Rachel Martin Noah Coccaro, Marie Meteer, Klaus Ries, Elizabeth Shriberg, Audreas Stolcke, Paul Taylor, and Van Ess-Dykema. 1997. Automatic detection of discourse structure for speech recognition and understanding. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 88–95.

Zornitsa Kozareva and Sujith Ravi. 2019. ProSeqo: Projection sequence networks for on-device text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3892–3901.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1694–1704. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations, Workshop Track*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014a. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014b. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. *CoRR*, abs/1905.11268.

Sujith Ravi. 2017. Projectionnet: Learning efficient on-device deep networks using neural projections. *CoRR*, abs/1708.00630.

Sujith Ravi. 2019. Efficient on-device models using neural projections. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5370–5379.

Sujith Ravi and Zornitsa Kozareva. 2018. Self-governing neural networks for on-device short text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 804–810.

Sujith Ravi and Zornitsa Kozareva. 2019. On-device structured and context partitioned projection networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3784–3793, Florence, Italy. Association for Computational Linguistics.

Keisuke Sakaguchi, Kevin Duh, Matt Post, and Benjamin Van Durme. 2016. Robsut wrod reocginiton via semi-character recurrent neural network. *CoRR*, abs/1608.02214.

Chinnadhurai Sankar, Sujith Ravi, and Zornitsa Kozareva. 2019. Transferable neural projection representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3355–3360, Minneapolis, Minnesota. Association for Computational Linguistics.

Elizabeth Shriberg, Rajdip Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of the SIGDIAL 2004 Workshop, The 5th Annual Meeting of the Special Interest Group on Discourse and Dialogue, April 30 - May 1, 2004, Cambridge, Massachusetts, USA*, pages 97–100.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, Christopher Potts, et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642.

Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck. 2010. What is left to be understood in atis? In *Proceedings of 2010 IEEE Spoken Language Technology Workshop (SLT)*, pages 19–24.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. 2017. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

EnPaR:Enhancing Entity and Entity Pair Representations for Joint Entity Relation Extraction

Yijun Wang[1, 2], **Changzhi Sun**[4], **Yuanbin Wu**[3], **Hao Zhou**[4], **Lei Li**[4], **and Junchi Yan**[1, 2]

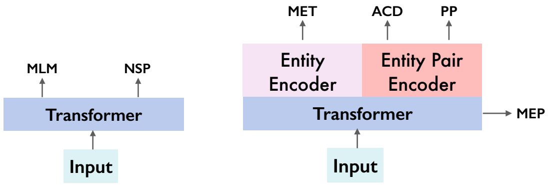
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[3]School of Computer Science and Technology, East China Normal University
[4]ByteDance, AI Lab
`yijunwang.cs@gmail.com ybwu@cs.ecnu.edu.cn yanjunchi@sjtu.edu.cn`
`{sunchangzhi, zhouhao.nlp, lileilab}@bytedance.com`

Abstract

Current state-of-the-art systems for joint entity relation extraction (Luan et al., 2019; Wadden et al., 2019) usually adopt the multi-task learning framework. However, annotations for these additional tasks such as coreference resolution and event extraction are always equally hard (or even harder) to obtain. In this work, we propose a pre-training method EnPaR to improve the joint extraction performance. EnPaR requires only the additional entity annotations that are much easier to collect. Unlike most existing works that only consider incorporating entity information into the sentence encoder, we further utilize the entity pair information. Specifically, we devise four novel objectives, *i.e.*, masked entity typing, masked entity prediction, adversarial context discrimination, and permutation prediction, to pre-train an entity encoder and an entity pair encoder. Comprehensive experiments show that the proposed pre-training method achieves significant improvement over BERT on ACE05, SciERC, and NYT, and outperforms current state-of-the-art on ACE05.

1 Introduction

Joint extraction of entities and relations is a fundamental task in information extraction, it aims to extract entities and relations with a unified model. Current approaches (Luan et al., 2019; Wadden et al., 2019) usually adopt the multi-task learning framework that optimizes many objectives simultaneously, including entity recognition, relation extraction, coreference resolution, and event extraction. However, as large-scale manually labeled data required by these methods is unavailable in many domains, their applicability is severely restricted. Therefore, we expect to catch or even surpass the multi-task based joint models with less annotation cost. Compared with the annotations of coreference resolution and event extraction, entity annotations can be easily obtained through automatic NER annotation tools (e.g., spaCy [1]). In this paper, we focus on improving the model's performance with just extra entity annotations.



(a) BERT (b) EnPaR

Figure 1: The network architectures and objectives of BERT and EnPaR. Our work introduce an entity encoder and an entity pair encoder. MLM = masked language model, NSP = next sentence prediction, MET = masked entity typing, MEP = masked entity prediction, ACD = adversarial context discrimination, PP = permutation prediction.

Although pre-trained models, like BERT, have shown impressive performance in many downstream tasks, they have mainly two limitations when applied in the joint entity relation extraction task. One is that currently pre-trained objectives are insufficient for this task. Specifically, these commonly used universal pre-trained model (e.g., BERT) do not consider the entity-related knowledge that is crucial for better extracting entities and relations. The other is that these models only provide pre-trained representations for tokens and sentences, but not entities and entity pairs. To obtain the representations for entities and entity pairs, additional parameters that are not pre-trained are introduced in the fine-tuning stage, which may futher impair the joint extraction performance.

To address the first limitation, recent several works try to incorporate entity-related information

---

[1] https://spacy.io/

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2877–2887
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

into pre-training objectives. Zhang et al. (2019) fuses heterogeneous information from both texts and knowledge graphs and proposes a denoising entity auto-encoder objective based on BERT. Sun et al. (2019c) presents two knowledge masking strategies in the pre-training stage (entity-level masking and phrase-level masking). Both of them utilize extra entity annotations (*i.e.*, entities in knowledge graphs and automatic entity annotations, respectively). In this paper, we follow this line of works and build a large-scale entity annotated corpus using the spaCy NER tool.

For the second limitation, we propose ENPAR, a pre-training method customized for entity relation extraction. ENPAR consists of an underlying sentence encoder, an entity encoder, and an entity pair encoder. Compared with BERT (Figure 1(a)), the proposed entity encoder and entity pair encoder directly provide representations of entities and entity pairs. To train the three encoders, we devise four novel pre-training tasks: *masked entity typing*, *masked entity prediction*, *adversarial context discrimination* and *permutation prediction* (Figure 1(b)). In the first two tasks, we randomly mask some entity words and then predict the masked tokens and the entity type. These two tasks are natural extensions of the masked language model. To learn a better entity pair encoder, we draw inspirations from the denoising auto-encoder (Zhang et al., 2019) and propose the last two tasks. Specifically, when the entity pair or its context in a sentence are perturbed, we hope that the entity pair encoder is capable of tracking such changes. We employ the parameter-sharing method for these four tasks and train these objectives jointly.

To sum up, our main contributions are as follows [2]:

- We introduce an entity encoder and an entity pair encoder to incorporate not only the entity information but also the entity pair information, which were ignored in current universal pre-trained models.

- We propose four novel pre-training tasks that help to learn the proposed encoders. These tasks only require additional entity annotations (with commonly used entity types), which can be automatically generated by public annotation tools, such as spaCy NER.

- We conduct comprehensive experiments and demonstrate that the proposed method achieves significant improvement on ACE05 and NYT dataset and is comparable with the state-of-the-art on the SciERC dataset.

2 Approach

Given an input sentence $s = x_1, \ldots, x_{|s|}$ and a set of entities $\mathcal{E}$ (automatically annotated) in s, ENPAR is to encode each entity $e \in \mathcal{E}$ and each entity pair (e_1, e_2) into a contextual representation vector. As shown in Figure 2, ENPAR is composed of a shared Transformer (Vaswani et al., 2017), an entity-level CNN followed by an MLP (multi-layer perceptron), a context-level CNN followed by an MLP, and the last MLP. In the pre-training stage, we optimize ENPAR with four objectives, namely, masked entity typing, masked entity prediction, adversarial context discrimination and permutation prediction. These pre-training objectives can integrate rich entity-related information into the proposed network. After pre-training, we can easily fine-tune the pre-trained network for entity relation extraction task.

2.1 Pre-training Network Architecture

In this section, we will introduce the overall ENPAR architecture in three parts: the sentence encoder, the entity encoder, and the entity pair encoder.

Sentence Encoder As previous pre-training models (UNILM, BERT, and XLM), we also apply the multi-layer Transformer (Vaswani et al., 2017) as the basic sentence encoder for obtaining the contextual representations $\mathbf{h}_i$ for each token in the sentence s. The output of multi-layer Transformer is computed via:

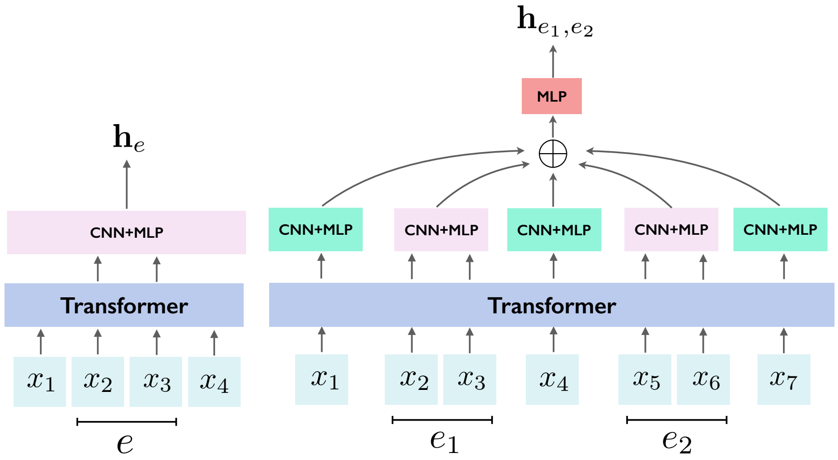
$$\{\mathbf{h}_1, \ldots, \mathbf{h}_{|s|}\} = \texttt{Transformer}(\{\mathbf{x}_1, \ldots, \mathbf{x}_{|s|}\})$$

The word representation $\mathbf{x}_i$ of x_i follow that of BERT (Devlin et al., 2018), which is a sum of the corresponding token, segment and position embeddings.

Entity Encoder For each entity $e \in \mathcal{E}$ in the sentence s, the corresponding contextual entity representation $\mathbf{h}_e$ can be obtained by employing a CNN (a single convolution layer with a max-pooling layer) followed by an MLP on vectors $\{\mathbf{h}_i | x_i \in e\}$, as shown in Figure 2(a).

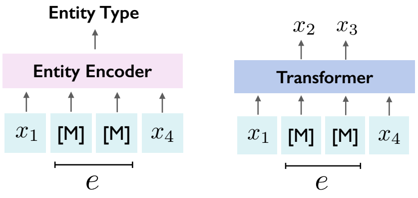
---

[2] Source code and pre-trained models are available at https://github.com/Receiling/ENPAR.



(a) Entity Encoder (b) Entity Pair Encoder

Figure 2: Entity encoder and entity pair encoder based on a shared sentence encoder. Both share the entity-level CNN with MLP, and entity pair encoder contains a context-level CNN with MLP.



(a) MET (b) MEP

Figure 3: Two objectives to learn enhanced representation of single entities.

Entity Pair Encoder For each entity pair (e_1, e_2) in the sentence s, to obtain the corresponding contextual entity pair representation $\mathbf{h}_{e_1,e_2}$, we extract two types of features. The first is the features regarding words in e_1, e_2, and the second is the features regarding contexts of the entity pair (e_1, e_2). For features on words in e_1, e_2, we use the output of entity encoder, namely, $\mathbf{h}_{e_1}$ and $\mathbf{h}_{e_2}$. For context features of the entity pair (e_1, e_2), we extract three feature vectors by looking at left context words, middle context words and right context words of the entity pair (e_1, e_2). Similar to entity encoder, we compute three feature vectors by employing another CNN followed by an MLP. Finally, we concatenate the five feature vectors into a single vector. To get the resulting entity pair representation $\mathbf{h}_{e_1,e_2}$, the single vector was fed into another MLP, as shown in Figure 2(b).

2.2 Pre-training Objectives

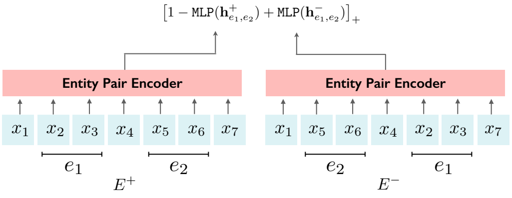
We design four pre-training objectives to guide EnPaR to absorb more entity-related knowledge, which is particularly important for entity relation extraction task. The four objectives can be divided into two groups: the first two objectives are to enhance the representations of single entities, and the latter two objectives are to enhance the representations of entity pairs. These objectives are trained jointly (simply sum the objective functions). Our pre-training objectives are based on a dataset with entity annotation, which can be obtained through the public annotation tool. For instance, `PER`("Obama"), `ORG`("Labour Party") were annotated by spaCy NER, and `PER`, `ORG` are entity types (there are 18 entity types).

Masked Entity Typing (MET) In this task, we simply mask some entity words at random, and then predict the corresponding entity type[3]. For instance, given a masked word sequence "x_1, `[M]`, `[M]`, x_4", to predict the masked entity type (e.g., `PER`), we first use the entity encoder to extract the contextual masked entity representation, and then predict the entity type. The objective is to minimize the cross-entropy loss computed using the predicted entity type and the original entity type, as shown in Figure 3(a).

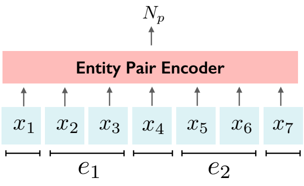
Masked Entity Prediction (MEP) This task is similar to the masked LM in BERT and is identical to the entity-level masking in (Sun et al., 2019c). Specifically, we randomly choose some entity words in the sentence, and replace them with special word `[M]`. Then we feed their corre-

---

[3] It is worth noting that the entity types annotated by spaCy NER may be different from the entity types of downstream datasets.



(a) ACD



(b) PP

Figure 4: Two objectives to learn enhanced representation of entitie pairs.

sponding output vectors computed by the sentence encoder into a softmax classifier to predict the masked entity word. The model is learned to recover the masked entity words, as shown in Figure 3(b). In short, both mask entity typing and mask entity prediction encourage the model to learn the information of single entities.

Adversarial Context Discrimination (ACD) Given an input sentence s with an entity pair (e_1, e_2), we regard it as a positive sample $E^+ = (s, e_1, e_2)$. According to E^+, we can generate a negative sample $E^- = (s', e_1, e_2)$ that has a main property: the context words and the order of context words w.r.t. the entity pair (e_1, e_2) in E^- are minimally different from those in the original sample E^+. If the entity pair encoder can characterize the context words well, it should be able to recognize small context differences of the entity pair between E^+ and E^-. We refer to this objective as adversarial context discrimination. The hinge loss function was imposed into the positive sample and corresponding negative samples to achieve the goal. Specifically, as shown in Figure 4(a), we can obtain a score using the entity pair encoder with an MLP for each sample, then the hinge loss function is computed via:

$$\left[1 - \texttt{MLP}(\mathbf{h}^+_{e_1,e_2}) + \texttt{MLP}(\mathbf{h}^-_{e_1,e_2})\right]_+ .$$

where $[u]_+ = \max(u, 0)$ is the hinge loss and $\texttt{MLP}$ outputs a scalar value. $\mathbf{h}^+_{e_1,e_2}$ and $\mathbf{h}^-_{e_1,e_2}$ are the output of entity pair encoder for positive sample E^+ and negative sample E^-.

Here we introduce our strategies of generating a negative sample $E^- = (s', e_1, e_2)$ according to a positive sample $E^+ = (s, e_1, e_2)$. In fact, there are many negative samples. For the sake of simplicity and clarity, we only adopt following simple rules to generate five negative samples.

Permutation	Class
(left context, e_1, middle context, e_2, right context)	001
(left context, e_1, middle context, right context, e_2)	002
(left context, e_1, e_2, middle context, right context)	003
⋮	⋮
(right context, e_2, middle context, e_1, left context)	120

Figure 5: Permutation samples.

- Swap entity e_1 and entity e_2 in the original sample E^+;
- Shift the entity e_1 few positions (n_s) to the left or right.
- Shift the entity e_2 few positions (n_s) to the left or right.

Permutation Prediction (PP) Given an input sentence s with an entity pair (e_1, e_2), the sentence s was split into five parts, namely, left context, e_1, middle context, e_2 and right context. If we shuffle the five parts, does the entity pair encoder have the ability to recognize it? Inspired by this question, we propose an enhanced objective, named permutation prediction, to help to learn a better entity pair encoder. Formally, let $\mathcal{P}$ be the set of all possible permutation of the fives parts, as shown in Figure 5. Obviously, the number of all possible permutations is 5! ($|\mathcal{P}| = 120$). For each permutation $p \in \mathcal{P}$, we first assign it a unique permutation class $N_p (1 \leq N_p \leq 120)$, and then use the entity pair encoder to extract the contextual entity pair representation for predicting the permutation class, as shown in Figure 4(b). The objective is to optimize the cross-entropy loss computed using the predicted permutation class and the gold permutation class. It is costly to consider all permutations. So we sample n_p permutations in practice (we always include the correct permutation).

2.3 Pre-training Setup

In the pre-training stage, we directly optimize the sum of the above four objective functions. Same as UniLM (Dong et al., 2019), we use gelu as activation function. And the sentence encoder is initialized with $\text{BERT}_{\text{BASE}}$ weights. We use the

English Wikipedia [4] as pre-training corpus, which has been processed in similarly as (Devlin et al., 2018). The spaCy NER [5] was used to annotate entities. After preprocessing the corpus, there are nearly 820M words and 95M entities in the annotated input. We discard the sentences having less than 3 entities for effectiveness, and only pre-train our model for one epoch. The vocabulary size is 28996, The maximum length of the input sequence is 256. For each entity, we replace the entity words with `[M]` with probability 15%, randomly replace other entity with probability 5% and keep the original entity words for the rest. Adam (Kingma and Ba, 2014) with $\beta_1 = 0.9, \beta_2 = 0.999$ is used for optimization. The batch size is 512. The learning rate is 5e-5, with linear warmup rate over the first 10% steps and linear decay. The dropout rate is 0.1. The weight decay is 0.01. It takes about 20 hours for 10, 000 steps using 1 Nvidia Telsa V100 16GB GPU.

2.4 Fine-tuning for Entity Relation Extraction

In the fine-tuning stage, we adopt the same definition of entity relation extraction task as (Sun et al., 2019a). The joint entity relation extraction task can be decomposed into three objectives: entity span detection, entity typing, and relation typing. Firstly, we treat the entity span detection as a sequence labeling task. We regard the first sub-word's output of the sentence encoder as the token-level representation. Then we take the token-level representation as the input to the softmax classifier and compute the cross-entropy loss with respect to gold entity span labels. Secondly, for each detected entity span, the entity classifier uses the corresponding output of the entity encoder to predict entity type. For each detected entity span pair, the relation classifier uses the corresponding output of the entity pair encoder to predict relation type. Both the entity classifier and the relation classifier are randomly initialized softmax layer. Also, we adopt the cross-entropy loss for these two tasks. Besides, all three objectives are optimized simultaneously.

We only tune the hyperparameters on the ACE2005 development set based on the joint performance of entity and relation, then apply the same hyperparameters on the SciERC and NYT datasets. Scheduled sampling strategy (Miwa and Bansal, 2016) and discriminative fine-tuning strategy (Howard and Ruder, 2018) are emploied in fine-tuning. We kepp the same dropout rate as pre-training (*i.e.*, 0.1). The learning rate is 2.5e-5 with weight decay 0.01. We apply a linear warmup scheduler over the first 20% steps and then linear decay. We train our model with a maximum of 200 epochs with early stop strategy in a single Nvidia GeForce GTX 1080 Ti GPU.

3 Experiments

We conduct experiments on three benchmark entity relation extraction datasets: ACE05, SciERC, and NYT. For space limitation, we will mainly discuss the results on ACE05 and report basic results on the remaining two datasets.

ACE05 The ACE05 dataset [6] that is a standard corpus for entity relation extraction task annotates entity and relation labels for a collection of documents. ACE05 contains 7 entity types and 6 relation types. We use the same data split and preprocessing of ACE05 dataset (351 training, 80 validating and 80 testing) as (Miwa and Bansal, 2016) and (Sun et al., 2018).

SciERC The SciERC dataset [7] annotates entity, coreference and relation labels for 500 scientific abstracts from 12 AI conference/workshop proceedings. We only use the annotations of entities and relations. SciERC contains 6 scientific term (entity) types and 7 relation types. We use the same data split and preprocessing of SciERC dataset (350 training, 50 validating and 100 testing) as (Luan et al., 2019).

NYT The NYT dataset[8] is a large-scale corpus which automatically annotates a collection of New York Times news articles. NYT contains 3 types of entities and 12 types of relations. The training set is automatically annotated by distant supervision. While the validation and testing data are manually labeled by (Jia et al., 2019). We choose the latest version of NYT released by (Jia et al., 2019).

Evaluation. As previous works (Miwa and Bansal, 2016; Sun et al., 2019a), we evaluate the

---

[4] Wikipedia version: enwiki-20190301.

[5] The spaCy model is "en_core_web_md" in version: 2.1.8, which trained on OntoNotes dataset.

[6] https://github.com/tticoin/LSTM-ER

[7] http://nlp.cs.washington.edu/sciIE/

[8] https://github.com/PaddlePaddle/models/tree/develop/PaddleNLP/Research/ACL2019-ARNOR/

Model	**Entity**	**Relation**	**Relation** (exactly)
Sun, 2019a	84.2	–	59.1
Li, 2019$^{\diamond}$	84.8	–	60.2
Luan, 2019$^{\star,\circ}$	88.4	63.2	–
Wadden, 2019 $^{\diamond,\circ}$	**88.6**	63.4	–
EnPaR $^{\diamond}$	86.9	**66.1**	**63.5**

Table 1: Results on the ACE05 test data. $\diamond$ means that the model uses BERT. $\star$ means that the model uses ELMo as token embeddings. $\circ$ stands for training the model with multi-task learning. EnPaR is the proposed model fine-tuned on ACE05 dataset.

performances using F1 score. Specifically, an output entity is correct if its type label and head region match with a gold entity, then an output relation is correct if both its type and its two argument entities are all correct (i.e. exactly match). While some previous works (Luan et al., 2019; Wadden et al., 2019; Sanh et al., 2019) do not consider entity type in relation evaluation. Thus, we also report this result for comparison.

3.1 Results on ACE05

First, we compare the proposed pre-training method with previous works in Table 1. In general, the relation performance of EnPaR significantly exceeds all existing models in two relation evaluation criteria. Specifically, in exactly matching mode, our method achieves 4.4 points improvement compared with the LSTM-based GCN joint model(Sun et al., 2019a) and increases by 3.3 points compared with the BERT-based QA model(Li et al., 2019). Even compared to the multi-task learning models based on BERT (Wadden et al., 2019), our method still achieves 2.7 points improvement on relation performance. Although the entity performance of our method inferior to the multi-task learning models (Luan et al., 2019; Wadden et al., 2019), we believe that those additional supervision signals such as coreference and event information in the fine-tuning step may cause the gap. Besides, they even consider all spans and cross-sentence context, which are empirically beneficial to entity performance. However, even with the slightly inferior entity encoder and lack of additional multi-task training data, our pre-trained entity pair encoder still achieves significantly superior relation performance, which fully demonstrates the powerfulness of the proposed pre-training method.

Model	**Entity**	**Relation**	**Relation** (exactly)
BERT	87.2	65.1	62.2
EnPaR	86.9	**66.1**	**63.5**
- MET	87.1	65.3	62.5
- MEP	**87.4**	65.6	62.7
- ACD	87.2	65.7	62.9
- PP	87.1	64.2	61.6
- CNN	87.1	66.0	62.9

Table 2: Results on the ACE05 test data in different settings. BERT is our model without pre-training, which is initialized by BERT$_{\text{BASE}}$ and fine-tuned on ACE05 dataset. "- *" is EnPaR without * task, where $* \in$ {MET, MEP, ACD, PP} ; "- CNN" means that only loading parameters of the sentence encoder from EnPaR, and the other parameters (i.e., the entity encoder and the entity pair encoder) are randomly initialized.

Percentage of Data	**Entity**	**Relation**	**Relation** (exactly)
100%	86.9	65.2	61.7
75%	86.8	65.3	62.5
50%	87.1	64.6	61.5
25%	86.9	**66.1**	63.5
15%	87.3	65.9	**63.6**
5%	**87.4**	65.9	63.4

Table 3: Results on the ACE05 test data by varying the size of pre-training data.

Next, we evaluate the proposed pre-training method with different settings. We have following four detailed observations regarding the results in Table 2.

- EnPaR (line 2) achieves superior relation performance (1.0 point and 1.3 points improvement) and comparable entity performance compared with BERT (line 1). This result demonstrates that the proposed pre-training objectives inject more entity-related information into the pre-trained model and enhance the relation extraction performance.

- Overall, the entity performances of all models fluctuate quite slightly (0.5 points). Interestingly, "- MEP" (line 4) achieves the best entity performance though its relation performance is not the best. The stable entity performance reflects that the token-level information is likely enough for entity recognition. Besides, the final training objectives may bias entity objectives or entity pair objectives, which leads to more improvement in relation performance than entity performance. Thus, we pay more attention relation perfromance in this paper.

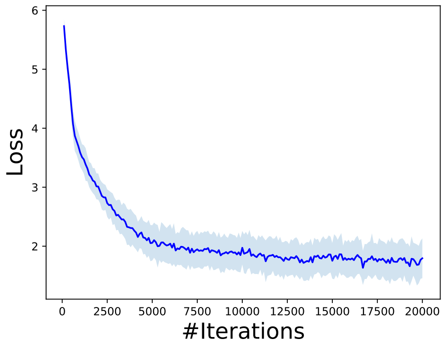


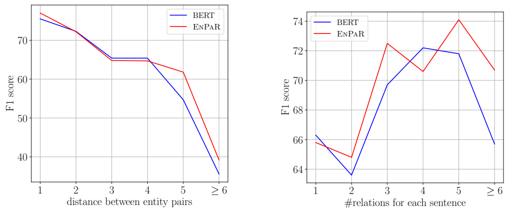
Figure 6: Pre-training loss with respect to the number of pre-training iterations.

- When any pre-training objective is removed, the relation performance will decrease with varying degrees. Particularly, "- PP" (line 6) drops the most (both 1.9 points decline in two relation evaluation criteria). This phenomenon just indicates the importance of the "PP" objective for the entity pair encoder and relation extraction.

- We also try only to use the pre-trained weights of the Transformer (sentence encoder), while the other parameters are randomly initialized in the fine-tuning step (line 7). The relation performance of the "- CNN" slightly declines compared with EnPaR, but it still outperforms BERT. This result reflects that the sentence encoder has absorbed the entity-related knowledge and can achieve more encouraging performance with the entity encoder and the entity pair encoder.

Thirdly, we present the influences of pre-training data size (Table 3). In experiments, we found the loss of the pre-trained model tends to be stable after iterating 5k steps (about 25% of all data), as shown in Figure 6. Table 3 demonstrates that the performance of the pre-trained model was quite competitive using 25% of pre-training data [9], and more pre-training data does not further improve performance. There is a similar conclusion on text classification (Sun et al., 2019b), which performs within-task further pre-training. This observation shows that the proposed pre-training method does not require expensive training costs.

Finally, we examine the relation performance with respect to different distances between entity pairs (Figure 7(a)) and different the number of relations for each sentence (Figure 7(b)), and also give some concrete examples to verify it (Table 4) [10]. Comparing with BERT, EnPaR is good at handling long-distance relation dependencies and interactions between multiple relations in a sentence. For S1, BERT does not detect the long distance relation `PHYS` between "[barbara starr]" and "[pentagon]" while EnPaR can handle it. For S2, EnPaR identifies a relation `PHYS` between "[charles]" and "[london]", but BERT fails even the relation `PHYS` between "[vladimir putin]" and "[london]" was detected. It shows BERT does not fully exploit the multiple relations in a sentence. We attribute these results to the powerful representations learned by the proposed pre-training objectives.



(a)

(b)

Figure 7: (a) F1 score with respect to the distance between the entity pairs. (b) F1 score with respect to the number of relation for each sentence.

3.2 Results on SciERC and NYT

SciERC The upper part of Table 5 shows the results of SciERC. Compared with BERT, EnPaR achieves 0.5 points improvement on entity performance and 2.2 points (exactly match) improvement on relation performance. This result reflects the effectiveness of the proposed pre-training objectives for entity relation extraction. Compared with the previous state-of-the-art model (Wadden et al., 2019), which is a multi-task learning model based on BERT, EnPaR achieves superior entity performance and comparable relation performance without additional multi-task training data. It worth noting that, the entities in the SciERC dataset are different from our pre-training data. Therefore, the entity encoder and the entity pair encoder can encode contexts of entities rather than only encoding information respect to specific entities. This property is desired as it means that we could utilize any entity annotator.

---

[9] In all experiments, we choose the number of pre-training data, n_p and n_s according to the performance on the development set during the fine-tuning step.

[10] More detailed evaluations are in the Appendix A

S1	…talk about what [barbara starr]$^{\texttt{PER:}\heartsuit\clubsuit\spadesuit}_{\texttt{PHYS-1:}\heartsuit\spadesuit}$ was reporting from the [pentagon]$^{\texttt{FAC:}\heartsuit\clubsuit\spadesuit}_{\texttt{PHYS-2:}\heartsuit\spadesuit}$.
S2	…[vladimir putin]$^{\texttt{PER:}\heartsuit\clubsuit\spadesuit}_{\texttt{PHYS-1:}\heartsuit\clubsuit\spadesuit}$ was greeted by prince [charles]$^{\texttt{PER:}\heartsuit\clubsuit\spadesuit}_{\texttt{PHYS-3:}\heartsuit\spadesuit}$ as he arrived in [london]$^{\texttt{GPE:}\heartsuit\clubsuit\spadesuit}_{\texttt{PHYS-2:}\heartsuit\clubsuit\spadesuit\vert\texttt{PHYS-4:}\heartsuit\spadesuit}$ today .

Table 4: Examples from the ACE05 dataset with label annotations from BERT and EnPar for comparison. The ♡ is the gold standard, and the ♣, ♠ are the output of the BERT ,EnPar respectively.

Model	**Entity**	**Relation**	**Relation** (exactly)
Luan, 2019$^{\star,\circ}$	65.2	41.6	–
Wadden, 2019 $^{\diamond,\circ}$	67.5	**48.4**	–
BERT $^{\diamond}$	67.4	46.0	34.3
EnPar $^{\diamond}$	**67.9**	48.0	**36.5**
BERT $^{\diamond}$	92.7	50.8	49.3
EnPar $^{\diamond}$	**94.8**	**54.4**	**52.6**

Table 5: Results on the SciERC test data (upper part) and the NYT test data (bottom part).

NYT For entity relation extraction task, there are no previous works on this dataset. We list the BERT and EnPar results in the same way as the previous two datasets. For the bottom part of the Table 5, we observe that EnPar significantly outperforms BERT on entity performance and relation performance. This again verifies the effectiveness of our proposed pre-training method.

4 Related Work

Joint entity relation extraction is an important task that has been extensively studied. One simple method to achieve joint learning is through parameters sharing, which usually share some input embeddings or sentence encoders (Miwa and Bansal, 2016; Katiyar and Cardie, 2017). To further explore the interactions between the outputs of the entity model and the relation model, many joint decoding algorithms were introduced into this joint task (Yang and Cardie, 2013; Li and Ji, 2014; Katiyar and Cardie, 2016; Zheng et al., 2017; Ren et al., 2017; Wang et al., 2018; Sun et al., 2018; Fu et al., 2019). Besides, (Li et al., 2019) tackle this task under the framework of multi-turn QA. And (Sun et al., 2019a) conduct joint type inference via GCN on a bipartite graph composed of entities and relations. Recently, transfer learning (Sun and Wu, 2019), multi-task learning (Sanh et al., 2019; Wadden et al., 2019; Luan et al., 2019) were also applied in this task. In this work, we investigate the pre-trained model for entity relation extraction. For simplicity, we restrict the joint model of parameters sharing, which can be easily extended to jont decoding methods.

Pre-trained models (Yang et al., 2019; Dong et al., 2019; Joshi et al., 2020) have made many amazing breakthroughs on various NLP downstream tasks. Pre-training paradigm first pretrain networks with some pre-training objectives on large-scale unlabeled text corpora, and then fine-tune the pre-trained networks on downstream tasks. These pre-training objectives determine the knowledge absorbed by the pre-trained models. For example, BERT (Devlin et al., 2018) adopts masked language model and next sentence prediction to learn deep contextual representations and the relation between two sentences respectively. XLNet (Yang et al., 2019) uses permutation language model to learn bidirectional representations with autoregressive model. UniLM (Dong et al., 2019) fuses three types of language model: unidirectional, bidirectional and seq2seq prediction. However, these models and objectives all ignore the entity-related information, which is crucial for entity relation extraction task. Recently, there are several public works that explore how to properly integrate entity inforamtion into pre-trained models. Specifically, (Zhang et al., 2019) achieve enhanced language representation by injecting informative entities in KGs into pre-training models. And (Sun et al., 2019c) propose two higher-level masking strategies: entity-level masking and phrase-level masking. In this work, we not only integrate entity information, but also extend to entity pair information and learn more powerful representations for entities and entity pairs.

5 Conclusion

We propose EnPar, a pre-training method customized for entity relation extraction only with additional entity annotations. Instead of only pre-training sentence encoder in universal pre-trained models, we also pre-train an entity encoder and an entity pair encoder. Then the proposed four objectives can incorporate entity and entity-pair knowledge into the pre-trained encoders to enhance the encoders' representations. Experiments on three datasets demonstrate that ENPAR achieves comparable or even superior performances compared with multi-task based joint models.

Acknowledgement

The authors wish to thank the reviewers for their helpful comments and suggestions. This research is (partially) supported by NSFC (62076097, 61972250, U19B2035), National Key Research and Development Program of China (2018AAA0100704), STCSM (18ZR1411500), the Foundation of State Key Laboratory of Cognitive Intelligence, iFLYTEK(COGOS-20190003). The corresponding authors are Junchi Yan, Changzhi Sun, and Yuanbin Wu.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Proc. of ACL*, pages 1409–1418.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. Arnor: Attention regularization based noise reduction for distant supervision relation classification. In *Proc. of ACL*, pages 1399–1408.

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

Arzoo Katiyar and Claire Cardie. 2016. Investigating lstms for joint extraction of opinion entities and relations. In *Proc. of ACL*, pages 919–929.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proc. of ACL*, pages 917–928.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proc. of ACL*, pages 402–412.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. *arXiv preprint arXiv:1905.05529*.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*.

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proc. of WWW*, pages 1015–1024.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proc. of AAAI*, pages 6949–6956.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019a. Joint type inference on entities and relations via graph convolutional networks. In *Proc. of ACL*, pages 1361–1370.

Changzhi Sun and Yuanbin Wu. 2019. Distantly supervised entity relation extraction with adapted manual annotations. In *Proc. of AAAI*, volume 33, pages 7039–7046.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, Wenting Wang, Kuang-Chih Lee, and Kewen Wu. 2018. Extracting entities and relations with joint minimum risk training. In *Proc. of EMNLP*, pages 2256–2265.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019b. How to fine-tune bert for text classification? *arXiv preprint arXiv:1905.05583*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019c. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint extraction of entities and relations based on a novel graph scheme. In *IJCAI*, pages 4461–4467.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proc. of ACL*, pages 1640–1649.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. *arXiv preprint arXiv:1706.05075*.

Appendices

A More Evaluations

Table 6 and Table 7 show the performances of “BERT” and our “EnPaR” on each entity type and relation type, respectively. For entity performance, “EnPaR” is almost the same as “BERT”. However, for relation performance, we can find “EnPaR” is superior to “BERT” on all relation types except “PER-SOC” relation. The major reason is “EnPaR” significantly improves recall while keeping or sacrificing little precision.

Figure 8 and Figure 9 demonstrate the relation performances (exactly match) of “BERT” and “EnPaR” in different situations. Specifically, Figure 8 shows the results with respect to the number of entities for each sentence, while Figure 9 shows the results with respect to the sentence length for each sentence. In Figure 8, our “EnPaR” significantly outperforms “BERT” when the number of entities in a sentence is more than 2. Similarly, when the sentence length is more than 10, our “EnPaR” is also superior to “BERT” as shown in Figure 9. Both results show the powerful ability of “EnPaR” to identify relations in complicate sentences and long sentences. Moreover, it also proves that the proposed pre-training objectives indeed prompt the model to learn entity related information, which contributes to improving the relation extraction performance.

Entity Type	Model	P	R	F
WEA (109)	BERT	75.9	**78.0**	**76.9**
	EnPaR	**79.0**	72.5	75.6
FAC (286)	BERT	77.9	75.2	76.5
	EnPaR	**78.1**	**76.2**	**77.2**
VEH (116)	BERT	**80.9**	**80.2**	**80.5**
	EnPaR	**80.9**	**80.2**	**80.5**
LOC (136)	BERT	**72.4**	77.2	**74.7**
	EnPaR	70.6	**79.4**	**74.7**
PER (2928)	BERT	**91.7**	**92.3**	**92.0**
	EnPaR	91.0	92.1	91.5
GPE (1013)	BERT	**86.8**	89.4	88.1
	EnPaR	**86.8**	**89.9**	**88.3**
ORG (817)	BERT	**77.6**	**77.0**	**77.3**
	EnPaR	77.3	75.8	76.5

Table 6: The entity performance of “BERT” and “EnPaR” for different entity types on ACE05 test data. The numbers in the first column are counts of entities.

Relation Type	Model	P	R	F
ART (146)	BERT	60.4	39.7	47.9
	EnPaR	**64.1**	**45.2**	**53.0**
PART-WHOLE (175)	BERT	**57.1**	**57.7**	**57.4**
	EnPaR	55.6	56.6	56.1
PER-SOC (73)	BERT	**75.0**	74.0	74.5
	EnPaR	73.1	**78.1**	**75.5**
PHYS (278)	BERT	**61.4**	48.6	54.2
	EnPaR	60.1	**54.7**	**57.3**
GEN-AFF (99)	BERT	**66.2**	45.5	53.9
	EnPaR	63.6	**49.5**	**55.7**
ORG-AFF (354)	BERT	**78.9**	71.8	75.2
	EnPaR	78.4	**72.6**	**75.4**

Table 7: The relation performance (exactly match) of “BERT” and “EnPaR” for different relation types on ACE05 test data. The numbers in the first column are counts of relations.

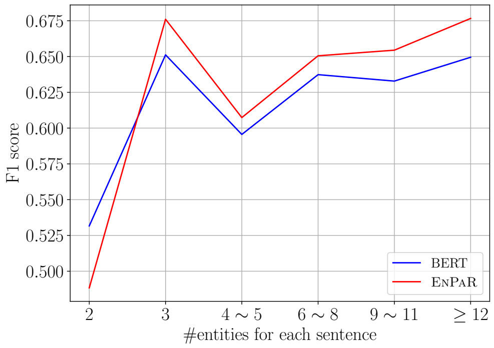


Figure 8: The relaiton F1 score (exactly match) with respect to the number of entities on ACE05 test data.

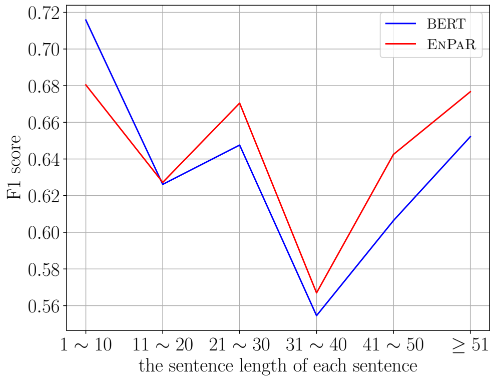


Figure 9: The relaiton F1 score (exactly match) with respect to the the sentence length for each sentence on ACE05 test data.

Text Augmentation in a Multi-Task View

Jason Wei🔥 **Chengyu Huang**🌏 **Shiqi Xu**🔥 **Soroush Vosoughi**🌳
🔥ProtagoLabs, `{jason,xu}@protagolabs.com`
🌏International Monetary Fund, `huangchengyu24@gmail.com`
🌳Dartmouth College, `soroush@dartmouth.edu`

Abstract

Traditional data augmentation aims to increase the coverage of the input distribution by generating augmented examples that strongly resemble original samples in an online fashion where augmented examples dominate training.

In this paper, we propose an alternative perspective—a *multi-task view* (MTV) of data augmentation—in which the primary task trains on original examples and the auxiliary task trains on augmented examples. In MTV data augmentation, both original and augmented samples are weighted substantively during training, relaxing the constraint that augmented examples must resemble original data and thereby allowing us to apply stronger levels of augmentation.

In empirical experiments using four common data augmentation techniques on three benchmark text classification datasets, we find that the MTV leads to higher and more robust performance improvements than traditional augmentation.

1 Introduction

Most data augmentation techniques aim to generate augmented examples for training that are similar to original data. In computer vision, operations such as flipping, cropping, and color jittering are both widely used and highly effective—it is self-evident that augmented examples closely resemble original data, and so we generate augmented data in an online fashion during each minibatch such that no original, unmodified examples are seen during training (Krizhevsky et al., 2012; Zagoruyko and Komodakis, 2016; Huang et al., 2017).

In language, on the other hand, even slight modifications can cause significant semantic changes, and so it is not always clear whether augmented examples resemble original data. Despite this uncertainty, many augmentation techniques in NLP still generate examples stochastically and ignore original data (Zhang et al., 2015; Sennrich et al., 2016a; Xie et al., 2017; Li et al., 2017; Kobayashi, 2018; Wang et al., 2018). When it is unclear whether augmented examples resemble original data—as is often the case—is it wise to neglect the original training data?

Traditional Data Augmentation

- **Intuition:** Increase coverage of input distribution by using augmented examples for training.
- **Guideline:** Augmented examples should be similar to original data.
- **Training:** Dominated by augmented examples that are generated stochastically.

Multi-Task View (MTV) of Data Augmentation

- **Intuition**: Auxiliary task of classifying augmented examples acts as regularization for the primary task of classifying original examples.
- **Guideline**: It might be a good idea for augmented samples to resemble original data, but they can be anything that boosts performance.
- **Training**: Both original and augmented data receive substantive weighting during training.

Table 1: Summary of traditional data augmentation versus MTV data augmentation.

Our paper questions this practice by proposing to include original data during training. Specifically, we make two contributions:

1. We propose a *multi-task view of data augmentation* (MTV data augmentation), which trains on both original and augmented examples and therefore allows us to relax the constraint that augmented examples must resemble original data. The MTV facilitates augmentation using a higher strength parameter.

2. We show empirically that four common data augmentation techniques provide higher and more robust performance gains using the MTV compared with traditional augmentation.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2888–2894
April 19 - 23, 2021.

2 Traditional Data Augmentation[1]

Situation. During regular training, the canonical maximum likelihood objective minimizes the cost of the original training set J_O:

$$J_\mathrm{O}(\theta) = \mathbb{E}_{x,y \sim \hat{\mathbf{p}}(X,Y)} \left[-\log \mathbf{p}_\theta(y|x) \right],$$

where $\hat{\mathbf{p}}(X,Y)$ is the empirical distribution of training pairs x, y and $\mathbf{p}_\theta(y|x)$ is the parameterized model that we aim to learn (e.g., a neural network). As $\hat{\mathbf{p}}(X,Y)$ is typically the observed data, it will likely have some mismatch with the true data distribution $\mathbf{p}(X,Y)$. When the mismatch is dramatic—for instance, when $\hat{\mathbf{p}}(X,Y)$ does not sufficiently cover the training space—model performance will likely suffer.

Remedy. In practice, we often use data augmentation to mitigate the inadequacy of $\hat{\mathbf{p}}(X,Y)$ by providing additional training data. We generate an augmented distribution $\mathbf{q}(\hat{X},\hat{Y})$ and now minimize the cost of this augmented training set J_aug:

$$J_\mathrm{aug}(\theta) = \mathbb{E}_{x,y \sim \mathbf{q}(\hat{X},\hat{Y})} \left[-\log \mathbf{p}_\theta(y|x) \right].$$

As we now optimize solely on $\mathbf{q}(\hat{X},\hat{Y})$, our goal is to find $(\hat{x},\hat{y})$ pairs that are likely to fall in the true distribution $\mathbf{p}$. Assuming the smoothness of $\mathbf{p}$, similar (x,y) pairs will have similar probabilities, and therefore if an augmented example is more similar to an observed example, it is more likely to be sampled under the true distribution. In other words, good augmented examples resemble the observed data, and we aim to find them. Conversely, if an augmented example diverges too far from any observed data, it is likely invalid and thus harmful for training; we don't want to train on these examples.

The majority of prior work follows this framework of augmented examples resembling real data. As popular techniques, semantic noising substitutes tokens with synonyms (Wang and Yang, 2015; Zhang et al., 2015; Li et al., 2017); Pervasive Dropout randomly removes words from the input sequence (Sennrich et al., 2016a); and SwitchOut (for machine translation) replaces some words in both source and target sentences with other words from their corresponding vocabularies (Wang et al., 2018).

Moreover, most of these techniques perform augmentation on every training example in an online fashion, implicitly assuming that augmented examples so closely resemble original data that directly training on original examples is not even worth considering. As we shall see in the next section, adding in these original examples during training might actually be a worthwhile idea.

3 MTV Data Augmentation

Multi-task optimization jointly trains on a primary task and one or more auxiliary tasks—the intuition is that requiring an algorithm to also learn an auxiliary task can act as better regularization than penalizing all complexity uniformly. Prior work has found that multi-task models work particularly well when the tasks are similar, but can also improve performance even on unrelated tasks (Paredes et al., 2012; Hajiramezanali et al., 2018).

We propose a multi-task view of data augmentation that has a primary task that optimizes regular training on original examples and an auxiliary task that optimizes training on augmented data. This MTV jointly optimizes the primary and auxiliary task(s) using a weighted cost function so that both original and augmented data receive substantial weight during training:

$$J(\theta) = \gamma_\mathrm{O} \cdot J_\mathrm{O}(\theta) + \gamma_\mathrm{aug} \cdot J_\mathrm{aug}(\theta),$$

where γ_O is the weight of **o**riginal data and γ_aug is the weight of **aug**mented data, and $\gamma_\mathrm{O} + \gamma_\mathrm{aug} = 1$. In this context, observe that vanilla training uses $\gamma_\mathrm{O} = 1$ and $\gamma_\mathrm{aug} = 0$, and traditional data augmentation uses $\gamma_\mathrm{O} = 0$ and $\gamma_\mathrm{aug} = 1$.

The MTV gives us an important freedom that is not offered by the traditional data augmentation framework. Since traditional data augmentation only trains on augmented examples, performance suffers detrimentally when augmented data differs too much from the true distribution—therefore, most studies aim to generate augmented examples that resemble original data. MTV data augmentation, however, jointly trains on both original and augmented data, thereby allowing us to relax the constraint that original and augmented examples come from the same distribution. In fact, accepting that the original and augmented distributions might differ or could even be unrelated—as work in multi-task learning has done (Paredes et al., 2012; Hajiramezanali et al., 2018; Rai and Daumé, 2010)—liberates us to apply stronger levels of data augmentation, which, as we will demonstrate in the next section, leads to higher and more robust performance.

---

[1] We closely follow the intuition and notation of Wang et al. (2018)

4 Experiments

This section compares multi-task view augmentation to traditional augmentation for various datasets and augmentation techniques.

4.1 Experimental Setup

Datasets. We conduct experiments on three text classification tasks often used as benchmarks (Kim, 2014): (1) Stanford Sentiment Treebank (**SST2**) (Socher et al., 2013) of movie reviews classified as positive/negative, (2) subjectivity/objectivity dataset (**SUBJ**) (Pang and Lee, 2004), where sentences are classified as either subjective or objective, and (3) question type dataset (**TREC**) (Li and Roth, 2002), in which questions ask for either a description, entity, abbreviation, human, location, or number.

Models and Experimental Procedures. For text classification, we use BERT (Devlin et al., 2019) (`bert-base-uncased` from HuggingFace) to extract features by averaging the last hidden states of the input tokens. To reduce the number of model hyperparameters and save computation time, we classify these features using a linear SVM trained for 1000 epochs.[2] Since training data size depends on the amount of augmented data, we adjust the number of training epochs so that all models receive the same number of updates. All experiments are run for five random seeds. Our baseline models without data augmentation achieved 84.5%, 93.1%, and 83.9% accuracy respectively on the SST2, SUBJ, and TREC tasks.

Augmentation Techniques. In this paper, we experiment with four simple and common data augmentation techniques studied in Wei and Zou (2019): **(1) Token Substitution** (Zhang et al., 2015) replaces words with WordNet (Miller, 1995) synonyms; **(2) Pervasive Dropout** (Sennrich et al., 2016a) applies word-level dropout; **(3) Token Injection** (Wei and Zou, 2019) insert a synonym of a random token in the sequence into a random position in that sequence; **(4) Positional Shuffling** (Wei and Zou, 2019) randomly chooses two tokens and swaps their positions. For all four techniques, a parameter α indicates *augmentation strength* by dictating how many perturbations are performed. For a given α, we perform $n = \alpha\, l$ perturbations, where l is the sequence length.

Aug. Technique	MTV	Best α	Avg. Boost	(Δ_{MTV})
Token Substitution	✗	0.05	1.3%	-
	✓	0.3	2.1%	(+0.8%)
Pervasive Dropout	✗	0.1	1.8%	-
	✓	0.4	2.5%	(+0.7%)
Token Injection	✗	0.05	0.7%	-
	✓	0.5	2.2%	(+1.5%)
Positional Shuffling	✗	0.05	1.4%	-
	✓	0.4	2.5%	(+1.1%)

Table 2: Average performance boost on three text classification tasks for four augmentation techniques using the best-performing augmentation strength from $\alpha \in \{0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$. Traditional data augmentation works best at low α, whereas MTV data augmentation provides the strongest performance for high α. Δ_{MTV} indicates additional boost from using the MTV compared with traditional augmentation.

4.2 Stronger augmentation for more gains

Table 2 summarizes results for data augmentation in the MTV using $\gamma_{\text{O}} = \gamma_{\text{aug}} = 0.5$ compared with traditional augmentation for the best-performing augmentation strength from $\alpha \in \{0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$. In the traditional framework, pervasive dropout had the strongest performance boost of 1.8% using $\alpha = 0.1$. The MTV, however, allowed for stronger augmentation (i.e., $\alpha \geq 0.3$) that resulted in all four techniques to achieving boosts of more than 2.0%.

Perhaps strikingly, token injection and positional shuffling, which are less intuitive and not as commonly used as token substitution and pervasive dropout, achieve the strongest gains ($> 1.0\%$) from using the MTV. One potential reason for this is that, compared with token substitution and pervasive dropout, token injection and positional shuffling are non-destructive in that they do not remove any of the original words, and so the nature of examples augmented at high α could be more conducive for the MTV.

4.3 More-robust gains at high α

When using data augmentation with high α, high levels of noising are employed and augmented data are therefore more likely to diverge from their original examples. Figure 1 takes a closer look at how performance is affected by varying α. Whereas traditional augmentation often negatively affected performance at high α, the multi-task view, which jointly optimizes the original distribution, had robust performance gains at high augmentation strengths.

---

[2] This setup is not state-of-the-art but allows for experiments to be performed on CPU.

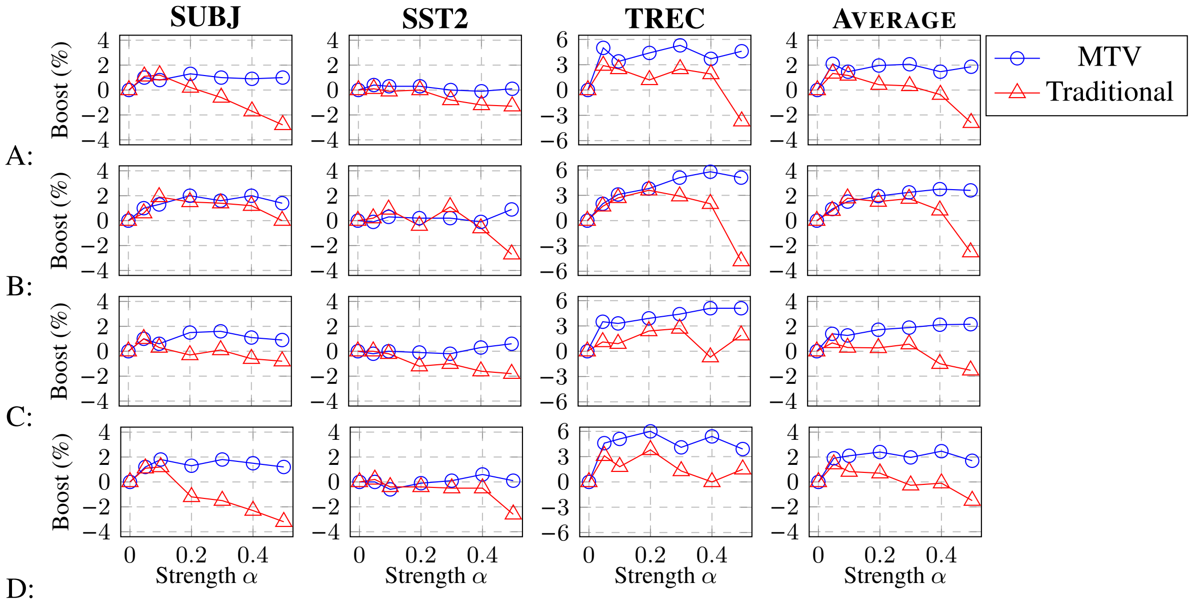
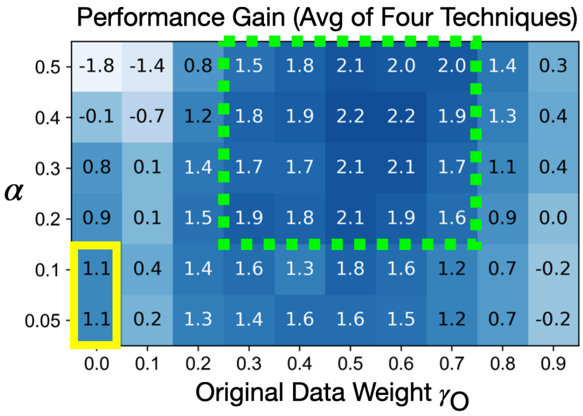


Figure 1: Performance boosts on three tasks using the traditional and multi-task view (MTV) frameworks for four data augmentation techniques: Token Substitution (Zhang et al., 2015) (A), Pervasive Dropout (Sennrich et al., 2016a) (B), Token Injection (Wei and Zou, 2019) (C), Positional Shuffling (Wei and Zou, 2019) (D). In the traditional framework, improvements are largest when augmentation strength α is small, with performance deteriorating for large α. The MTV, on the other hand, jointly optimizes for both original and augmented data, leveraging higher α to provide higher and more robust performance gains.



α \ γ_O	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.5	-1.8	-1.4	0.8	1.5	1.8	2.1	2.0	2.0	1.4	0.3
0.4	-0.1	-0.7	1.2	1.8	1.9	2.2	2.2	1.9	1.3	0.4
0.3	0.8	0.1	1.4	1.7	1.7	2.1	2.1	1.7	1.1	0.4
0.2	0.9	0.1	1.5	1.9	1.8	2.1	1.9	1.6	0.9	0.0
0.1	1.1	0.4	1.4	1.6	1.3	1.8	1.6	1.2	0.7	-0.2
0.05	1.1	0.2	1.3	1.4	1.6	1.6	1.5	1.2	0.7	-0.2

Figure 2: Performance boost (%) with varying augmentation strengths α and weights of original data γ_O during training. Traditional data augmentation (yellow solid box in lower left) uses modest augmentation strength ($\alpha = 0.05, 1$) with no original examples for training ($\gamma_O = 0$). The MTV data augmentation approach (green dashed box) suggests substantive weighting of original examples (e.g., $\gamma_O = 0.5$) which allows for much stronger augmentation (e.g., $\alpha \geq 0.3$).

4.4 Choosing γ_O and γ_{aug} weighting

As our experiments so far have used the MTV with balanced weighting of original and augmented data ($\gamma_O = \gamma_{\text{aug}} = 0.5$), in this section we explore different weightings of γ_O and γ_{aug}. Figure 2 shows these results averaged over all three datasets and all four augmentation techniques. Traditional data augmentation, which uses modest augmentation strength (e.g., $\alpha \in \{0.05, 0.1\}$) and does not train on original data ($\gamma_O = 0.0$), achieves reasonable performance gains. As expected, when stronger augmentations were applied (e.g., $\alpha \geq 0.4$), training with only augmented data hurts performance. When training on both augmented and original data, however, performance improved with stronger augmentation and remained robust for varying augmentation strengths $0.2 \leq \alpha \leq 0.5$ and original data weights $0.3 \leq \gamma_O \leq 0.7$.

5 Further Related Work

Prior work on data augmentation, to our knowledge, generally follows the traditional data augmentation framework. In addition to the methods mentioned in §2, Xie et al. (2017) replaced words with samples from the unigram frequency distribution; Yu et al. (2018) translated English sentences to French and back to English (backtranslation); and Kobayashi (2018) replaced words with other words based on a language model. All these methods could potentially be formulated in the MTV.

Some prior work has also drawn connections between seeing data augmentation as multiple tasks. Similar to how we optimize augmented data as a separate task, Meyerson and Miikkulainen (2018) created fake tasks by using multiple distinct decoders to train a shared structure to solve the same problem in different ways. In machine translation, Sennrich et al. (2016b) used monolingual training examples as parallel examples with an empty source side, noting that their setup could be seen as multi-task learning with the tasks as translation with known sources and language modeling with unknown sources. Compared with these papers that create multiple tasks in very specialized scenarios, the multi-task view that we have presented here can be used for any type of text data augmentation.

To be clear, our study is not the first to mix original and augmented data in training. For instance, Wang and Yang (2015) use a ratio of 1:5 original to augmented examples, but this weight of original data is much smaller than the $0.3 \leq \gamma_O \leq 0.7$ that we advocate for. Sennrich et al. (2016b) also include original data when training with back-translation augmentation, but the given ratios of original and augmented data they use appear to dictated by the speed of their back-translation models rather than an intentionally-motivated design choice. We see our work as the first to explicitly formulate the MTV, advocate for a joint optimization function, and comprehensively explore its implications on common text augmentation techniques.

As a limitation, our study has focused on label-preserving augmentation techniques, and our line of reasoning may not apply when augmentation techniques intentionally change the label. Moreover, we have only studied text classification with simple models using task-agnostic augmentation techniques. Future work in this direction could experiment with larger-scale models or study task-specific augmentation.

6 Conclusions

We have proposed a multi-task view that gives both original and augmented examples substantial weight during training, contrasting prior work that performs stochastic data augmentation and ignores original training data. For four common augmentation techniques, we found experimentally that this alternative view allows for stronger levels of augmentation, which in turn leads to better and more robust performance than traditional augmentation. We hope our paper inspires future work using text data augmentation to think more explicitly about how much augmented examples resemble original data and consider substantive weighting of original data when using data augmentation to improve model performance.

To close, we leave the enthusiastic reader with one last thought. Most existing text data augmentation techniques have obediently followed the paradigm from computer vision of generating augmented examples that are similar to the original data. Who's to say that's how data augmentation ought to work in NLP? In this paper, we've shown how to search for relative freedom from this constraint, simply by taking a different view of the underlying assumptions. Now, a bigger question arises on the horizon—what new text augmentation techniques are unlocked when augmented data are not forced to resemble the original?

References

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

Ehsan Hajiramezanali, Siamak Zamani Dadaneh, Alireza Karbalayghareh, Mingyuan Zhou, and Xiaoning Qian. 2018. Bayesian multi-domain learning for cancer subtype discovery from next-generation sequencing count data. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 9133–9142, Red Hook, NY, USA. Curran Associates Inc.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2017. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27, Valencia, Spain. Association for Computational Linguistics.

Elliot Meyerson and Risto Miikkulainen. 2018. Pseudo-task augmentation: From deep multitask learning to intratask sharing-and back. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 3508–3517. PMLR.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bernardino Romera Paredes, Andreas Argyriou, Nadia Berthouze, and Massimiliano Pontil. 2012. Exploiting unrelated tasks in multi-task learning. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 951–959, La Palma, Canary Islands. PMLR.

Piyush Rai and Hal Daumé. 2010. Infinite predictor subspace models for multitask learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 613–620, Chia Laguna Resort, Sardinia, Italy. PMLR.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 371–376, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Ziang Xie, Sida I. Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y. Ng. 2017. Data noising as smoothing in neural network language models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International*

Conference on Neural Information Processing Systems - Volume 1, NIPS'15, page 649–657, Cambridge, MA, USA. MIT Press.

Representations for Question Answering from Documents with Tables and Text

Vicky Zayats[†][*] **Kristina Toutanova**[†] **Mari Ostendorf**[§]
[†]Google Research [§]University of Washington
`{vzayats, kristout}@google.com` `ostendor@uw.edu`

Abstract

Tables in Web documents are pervasive and can be directly used to answer many of the queries searched on the Web, motivating their integration in question answering. Very often information presented in tables is succinct and hard to interpret with standard language representations. On the other hand, tables often appear within textual context, such as an article describing the table. Using the information from an article as additional context can potentially enrich table representations. In this work we aim to improve question answering from tables by refining table representations based on information from surrounding text. We also present an effective method to combine text and table-based predictions for question answering from full documents, obtaining significant improvements on the Natural Questions dataset.

1 Introduction

Tables are a common type of information representation used across the Internet. With billions of search queries a day,[1] question answering on tables is an important task that translates into a large number of search queries every second about information present in tables. In general, research on Question Answering (QA) can be categorized in terms of the resources that are used in answering the question: text documents (often referred as *unstructured* text in the literature), tables, or a structured knowledge base (KB). In our work we are interested in the combination of text-based and structured resources for question answering, particularly articles that contain both tables and text. This is a natural next step for question answering on tables, in that most tables are embedded in documents that discuss them, creating the challenge of determining whether the answer is in the text or the table (if anywhere). In addition, very often information presented in tables is compact and abbreviated. The associated text can potentially provide rich context that can be used to enhance the representation of the table for more robust question answering.

The main focus of this paper is to investigate how to improve question answering on documents that contain both text and tables. While recently there has been a lot of interest in reading comprehension for both text and tables, little research has been done in combining the two sources of information. The only prior study we are aware of is by Chen et al. (2020) who introduced a new dataset for multi-hop QA over tabular and textual data. In their work, the authors heavily rely on the assumption that the questions would be unanswerable if either text or table information is missing. Here we investigate a more realistic scenario of naturally occurring questions, where the answer can be found in either text, tables, both, or none. We evaluate our approach on the Natural Questions corpus (Kwiatkowski et al., 2019) which consists of real anonymized queries issued to the Google search engine and corresponding Wikipedia articles, simulating a real use case of such a system.

Prior work on the Natural Questions dataset has treated text and tables uniformly, linearizing tables and representing them and text segments using the same contextual token representations (for example, starting from pre-trained transformers (Vaswani et al., 2017) like BERT (Devlin et al., 2019)). However, representations developed for text are sub-optimal for tables, since they do not account for the special relationships between table cells, defined by the row and column structure. In this work, we extend the BERT architecture to account for inter-cell relationships in tables. This approach is motivated by Graph Neural Networks with a transformer (Shaw et al., 2018) and is

---

[*] Most of the work was done while the author was at the University of Washington.

[1] https://www.internetlivestats.com

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2895–2906
April 19 - 23, 2021.

closely related to the one in Müeller et al. (2019). In our work, we pretrain parameters for the new relationships using a large table corpus extracted from Wikipedia (Bhagavatula et al., 2013).

In addition, we present a novel approach that refines table representations by attending to related representations of text in the surrounding article. This allows information to propagate from the text to table elements, improving the ability of the model to interpret tables and find answers in them.

Overall, the main contributions of our work are three-fold. First, to the best of our knowledge, this is the first work that investigates how to effectively combine text-based and table-based approaches in a setting where it is unknown which, if any, of these modalities contains an answer to a question. Second, we introduce a novel mechanism to enrich table representations based on text surrounding the table, which improves the performance of a model for question answering from tables. Finally, this is the first model that uses the Natural Questions corpus for question answering on tables, improving the baseline in Alberti et al. (2019) that does not distinguish between tables and text.

2 Related Work

Most work on QA with tables prior to BERT involves first converting the table to a Knowledge Graph (KG) where cell entries are entities with row/column relations, then using entity linking to identify spans in the question that match an entity in the knowledge graph, and finally parsing the question to generate a SQL query using some variant of a sequence-to-sequence model (Krishnamurthy et al., 2017). Due to the advances in contextualized word embeddings, more recent work proposed a modification of the BERT transformer architecture to be used for representing tables. Hwang et al. (2019) proposed the usage of additional `[SEP]` tokens between headers of the table to make a BERT model more suitable for the tables. Recently, Yin et al. (2020) introduced a pretraining procedure for joint representation of tabular data paired with an utterance, where the approach is to linearize the structure of tables to be compatible with a BERT model. Our approach for table encoding is most similar to that of Müeller et al. (2019), where the authors generalized the BERT architecture similarly to Shaw et al. (2018) with new types of relations to encode table-specific relationships. The main differences between our table representation and Müeller et al. (2019) is that in our representation we use 5 types of relations, cell-column, cell-row, in-cell, cross-column and cross-row (more details in Section 3.1), while in their work the authors use cell-column and cell-row relations only for the table representation, but in addition use question-cell relations for marking matches between tokens in the question and corresponding cell values. Finally, the two most recent works on table representation learning, TaPaS (Herzig et al., 2020) and GraPPa (Yu et al., 2020), also use pretraining on the Wikitables dataset (Bhagavatula et al., 2013) that we use in our work. Therefore, our table representations based on transformers and our pretraining method are comparable to those in recent and concurrent work.

Leveraging tables is a hard problem. However, most studies on table-based QA omit an important additional information source: the text in the article discussing the table. Prior attempts at integrating a KB and text use early fusion of document text and KG information (Sun et al., 2018), where they integrate text and a KG sub-graph in a single graph, from which an entity is selected to answer the question. Structured KGs are often easier to interpret than tables, which have a wide variety of possible schemas. InfoTabS (Gupta et al., 2020) introduced a dataset for the natural language inference task based on premises that are tables, where the authors explore multiple table representations, including a key-value approach and linearized representations with table rows corresponding to ”sentences.” Hypothesis representations are calculated separately. Recently, TaBERT (Yin et al., 2020) introduced a joint table-utterance representation approach, where a table row is concatenated with a short text utterance, such as the query in question answering, and passed as an input to a BERT-based model. Such an approach relies on the initial table representation to select the table rows most relevant to the query. In contrast, we enrich the table representation using an attention mechanism with the representations of the most relevant parts of the context of the article in which the table appears.

The Natural Questions is a large corpus that contains real user queries along with their corresponding Wikipedia articles, which may or may not contain an answer anywhere in the article. Alberti et al. (2019) provided a BERT-based baseline that treats both table and text segments like text: a sequence of tokens with word and position embeddings. Recently, Liu et al. (2020) improved this baseline by using dynamic dual-attention over paragraphs and cascade answer predictor. In another direction, Ravula et al. (2020) used an extended transformer architecture that models extra-long documents with limited propagation of information among different segments. All three approaches did not distinguish between text and table input, treating tables as text while not taking into account table structure. To the best of our knowledge, this is the first work that focuses on table-based QA for the real user queries in the Natural Questions corpus, and shows that table-based and overall QA performance can be improved by building on state-of-the-art pre-trained representations of table structure, additionally enriched via attention to related article text.

3 Methods

This section describes two methods that improve table representations for QA from tables: an extension of the BERT architecture for table encoding to better capture the relationships between table elements, and a mechanism that incorporates related unstructured text of an article as context to further improve the table representation. We then describe an approach for question answering from both text and tables that combines predictions from text QA and table QA models using a late fusion approach.

3.1 Table Encoding

Our table encoding is a generalization of the transformer architecture (Vaswani et al., 2017), with the self-attention sub-layer extended to incorporate relations between structural components in tables, similar to the one introduced by Müeller et al. (2019). This approach is motivated by Graph Neural Networks (GNNs) with a transformer (Shaw et al., 2018, 2019) where the authors generalize the transformer by introducing multiple types of relations between inputs.

3.1.1 GNNs with a Transformer

In the original transformer, the multi-head self-attention for head h is calculated using query (Q), key (K) and value (V) projections as follows:

$$Att(Q, K, V) = softmax(\frac{(W_h^Q Q)^T (W_h^K K)}{\sqrt{d_k}}) W_h^V V$$

where W_h^Q, W_h^K and W_h^V are learned parameters, and d_k is the query/key dimension. By calculating the dot product between query and key projections, a transformer captures the interaction between each pair of inputs x_i and x_j (e.g. wordpieces in BERT) at positions i and j for $0 \leq i, j \leq N$. This interaction can be generalized to account for relation type t between x_i and x_j by biasing the key projection using r_{ij}^t:

$$s_{ij}^h = \frac{(W_h^Q x_i)^T (W_h^K x_j + r_{ij}^{t,h})}{\sqrt{d_k}}$$

and then scaling using softmax across all inputs $0 \leq j \leq N$. Thus, the standard transformer can be considered as a special case with $r_{ij} = 0$. Similarly, the value projection can also be updated with the corresponding relation type represented using the bias term ρ_{ij}^t, with the overall attention head h calculated as follows:

$$\alpha_{ij}^h = softmax(\frac{s_{ij}^h}{\sum_j s_{ij}^h})$$
$$w_i^h = \sum_j \alpha_{ij}^h (W_h^V x_j + \rho_{ij}^{t,h})$$

The parameters $r^{t,h}$ and $\rho^{t,h}$ are head- and layer-specific.

3.1.2 Table Encoding using GNN

Table structure can be encoded using the model described above to account for special relations in the table. Here, we use the following relation types:

- token cell - token column header relation
- token cell - token row header relation
- in-cell token relations
- cross-column header relations
- cross-row header relations

For each of these table-specific types of relations we learn different type-specific biases r^t and ρ^t for each layer and head, while for the rest of the relations we use the original BERT configuration with zero bias. Figure 1 shows an example of a table with the table relations used in this work.

Our table encoding is similar to the one independently proposed by Müeller et al. (2019) with the main difference of having relations on a token level rather than cell level. Also, in their approach, the authors use only cell-column and cell-row relation, while in our work we also use cross-column header, cross-row header and in-cell relations. The above mentioned paper includes additional relations based on n-gram matches with the question, and special processing of numerical values.

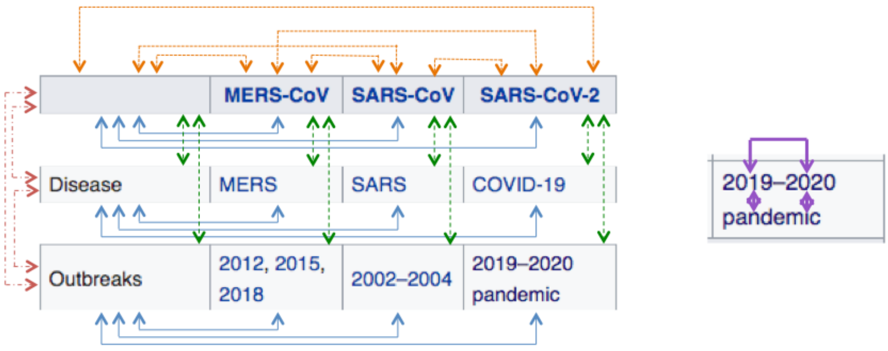


Figure 1: Example of a table encoding with relations: (*green, dashed*) token cell - token column header relations; (*blue, solid*) token cell - token row header relations; (*violet, bold*) in-cell token relations; (*red, dash-dotted*) cross-column header relations; (*orange, dotted*) cross-row header relations.

3.1.3 Pretraining

The main motivation for applying the original BERT model to table encoding is to use contextualized embeddings that are pretrained on a large amount of data. The new relation biases incorporated as part of the proposed table encoding are randomly initialized, starting from a standard transformer model pre-trained on text. Since these parameters are added for each layer, they can significantly change the activations of the pretrained model. In order to derive a better initialization point for the additional bias terms, we pretrain a GNN model with the masked LM objective used in BERT on the Wikitables dataset (Bhagavatula et al., 2013), which contains 1.6M tables from English Wikipedia. In order to limit the amount of overfitting on that table set, we freeze all original BERT parameters while updating only the bias terms introduced in the GNN. We tune the model using perplexity on a subset of the Wikitables dataset.

3.2 Context-aware Table Representation

We hypothesize that text in the article of a Web page containing a table can help build an improved representation of the table for QA. Recent work has explored building encoders over large input sequences. Ravula et al. (2020) scaled input sequence length to more than 8,000 tokens for the NQ dataset. However, to make the model efficient, encodings of individual text or table segments communicate through single-vector global memories. Here, we take the approach of using asymmetric attention from table token representations to a small number of relevant text token representations, that are pre-computed independently. Our approach is more similar to the handling of prior segment context in Transformer-XL (Dai et al., 2019), but relevant context is selected based on word overlap and not contiguity.

The two components of our approach, described next, include the definition of relevant text context for table elements and the mechanism for using contextualized embeddings of the relevant text to enrich the table token representations.

3.2.1 Table-Textual Context Linking

Let a table cell that contains a sequence of input tokens be defined as $(u_0^t \dots u_K^t)$, with the corresponding s sub-word units (wordpieces in BERT or byte-pair encodings in RoBERTa) for the k-th word to be defined as $(x_0^{t,k} \dots x_{S_k}^{t,k})$, and let the textual context of the article surrounding the table be defined as $(u_0^c \dots u_N^c)$, with the corresponding sub-word units for the n-th word be as $(x_0^{c,n} \dots x_{S_n}^{c,n})$. For each word in the table u_i^t, we find the corresponding context in the text using the exact match of the lower-case sequence of tokens, starting with the trigram matches, following with bigram and unigram matches.[2] For example, a trigram match for a word u_i^t is $\alpha_3(u_i^t) = u_j^c$ if a lower-case expression $(u_i^t, u_{i+1}^t, u_{i+2}^t)$ equals to the lower-case expression $(u_j^c, u_{j+1}^c, u_{j+2}^c)$. For each of the table tokens u_k^t we collect up to 6 corresponding matches from the text, $u_{s1}^c \dots u_{s6}^c$, and extract their sub-word embeddings represented by the last layer of pretrained RoBERTa, $e(x_0^{c,s1}) \dots e(x_s^{c,s6})$. Then, we stack all the sub-word unit embeddings associated with u_k^t to get a text-aware representation for word k in the

---

[2] Function words are frequent in multi-word expressions. To avoid exact match of expressions solely consisting of function words, matching expressions must contain at least one non-frequent word, defined based on the 200 most frequent words in the training set and the NLTK stop-word list.

table $e(u_k^t) = [e_0(x_0^{c,s1}); \dots; e_r(x_s^{c,s6})] \in \mathbb{R}^{r \times d}$, with d being the size of the hidden RoBERTa embedding and r being the total number of sub-word units corresponding to $u_{s1}^c \dots u_{s6}^c$. For simplicity of implementation, we use $r = 12$, where we prune any extra word-piecies or use padding for the cases where $r \neq 12$.

3.2.2 Enriching Table Representations

In order to incorporate text context $e(u_i^t)$ into a table representation of $x_s^{t,i}$, we use a slightly modified version of self-attention in the transformer, with the goal of generating an additional text context head, concatenated with the rest of the attention heads across multiple transformer layers that would be aware of the broader text context.

In the original transformer, self-attention is calculated between an input at position i using a query projection $W^Q x_i$ and the rest of the inputs in positions $j \in 0 \dots N$ using key and value projections, $W^K x_j$ and $W^V x_j$, correspondingly. Now, when calculating a text-aware head for each of the sub-word units $x_i^{t,k}$, we use attention between a query of $x_i^{t,k}$ and all of the sub-word units of the corresponding text context $e(u_k^t)$ using the key and value projections $W_e^K e(u_k^t)$ and $W_e^V e(u_k^t)$, correspondingly:

$$Att(Q_e, K_e, V_e) = \\ softmax(\frac{(W_e^Q Q)^T (W_e^K e(K))}{\sqrt{d_k}}) W_e^V e(V)$$

Then, we concatenate the new text-aware head with the rest of the heads in the layer h, resulting in a total of $m+1$ heads ($m = 16$ for RoBERTa$_{\text{large}}$), each of a size $k = 64$. In addition, we extend the current projection layer from a size $km \times km$ to $k(m+1) \times km$ in order to fit the additional head, randomly initializing the additional $k \times km$ parameters. For computational efficiency, we incorporate the text-aware representation only at layers 12, 16, 20 and 24.

3.3 Combining Text and Table Answer Predictions

In question answering, both text and table contexts can be used to support meeting the user information need. Question answering systems should therefore be able to consider both sources of information to present the most suitable answer. So far we have presented enriched table representations that can be used for question answering from tables. We now consider approaches for the full document-level QA task, where an answer may be found in either or none of the two modalities.

Since a text-based QA model would not benefit from the architecture and pre-training extensions for our table representations, we use a standard text-based representation for QA from text. We combine predictions from two separate models for the full document-level QA task. Specifically, we train a generic model for full article question answering following (Alberti et al., 2019). This model assigns scores to candidate answers in both text and tables using a standard pre-trained text representation (RoBERTa). We also train a separate model which uses enriched table representations and pre-training, and focuses on predicting answers in tables. The two model predictions are combined using a late fusion approach detailed below.

3.3.1 Calculating Prediction and Confidence Scores

We follow Alberti et al. (2019) to define a loss function for training and an answer span prediction method. More specifically, at inference time the scores that correspond to the start and end of a possible answer span are defined as follows:

$$\begin{aligned} g(c,s,e) &= f_{start}(s,c;\theta) + f_{end}(e,c;\theta) \\ &- f_{start}(s = [\text{CLS}], c; \theta) \\ &- f_{end}(e = [\text{CLS}], c; \theta) \end{aligned}$$

where c is a context of 512 sub-word unit ID's (including question and document tokens), $s, e \in \{0, 1, ..., 511\}$ are inclusive indices pointing to the start and end of the target answer span, θ is our model parameters, and f_{start}, f_{end} are two different outputs derived from the last layer of our model using linear projections. Following this work, the `[CLS]` token is used at training time to predict no answer instances, making $g(c,s,e)$ the log-odds of the likelihood of an answer span and the `[CLS]` span. All the contexts from each document are scored and document spans (s,e) are ranked to return the highest scoring span that does not exceed 30 tokens. We denote the highest scoring span for the generic model as g_c, and the highest scoring span for table model as g_t.

In addition to the prediction score, we also calculate a confidence score κ for each span:

$$\kappa(c, s, e) = \log(\frac{\exp(f_{start}(s, c; \theta))}{\sum_{s'} \exp(f_{start}(s', c; \theta))}) + \log(\frac{\exp(f_{end}(e, c; \theta))}{\sum_{e'} \exp(f_{end}(e', c; \theta))})$$

3.3.2 Combining Predictions

To combine the scores of the generic model g_c and the table model g_t, we use grid search on three parameters: a scaling factor α, a bias β, and a confidence threshold γ associated with the confidence of the table prediction κ_t:

$$g = \begin{cases} \max_{t,c}(g_c, \alpha \cdot g_t + \beta) & \text{if } g_c^l \in \text{t}, \kappa_t > \gamma \\ g_c & \text{otherwise} \end{cases}$$

where $g_c^l \in$ t indicates whether the long answer span that is predicted by the generic model points to a table. The search for parameters α, β, and γ is done on a validation set.

4 Experiments

We evaluate our model on the Natural Questions (NQ) corpus (Kwiatkowski et al., 2019) that consists of questions and paired Wikipedia pages, with the task of finding the exact location of the answer that is present in the article, if any. The NQ dataset is large (300k training samples) and answers (if present) can appear in either or both text and tables. NQ contains one human annotation for every question in the training set, and 5 annotations for every question in the development and test sets. Based on the statistics of the development set, around 14% of questions contain a short answer in a table. Only 48% of the questions have a long answer annotation (a paragraph or a table that contains an answer), while only 33% contain a short answer annotation (an exact location of a short answer phrase). In this work we first evaluate our models on a subset of questions that contain at least one short answer in tables, referred as NQTables, and then further evaluate the model on the full dataset, referred as FullNQ.

For the questions in NQTables, we further evaluate in two settings: (*i*) NQTables$_{\text{Tab}}$, where systems are limited to predict answer spans only from tables, and (*ii*) NQTables, where systems can predict answer spans from both text and tables. Note that although all questions in NQTables have at least one answer in a table, they might also have answers in text, and systems operating in the full NQTables setting have more chances to arrive at a correct answer than systems in NQTables$_{\text{Tab}}$ setting. While our primary focus in this work is on improving the short answer prediction, we also report the long answer prediction results for our best model used for the full NQ dataset, according to official metrics.

The development set (DEV) for NQTables used in this work contains 1118 questions where the short answer can be found in a table. Since the official test set of the corpus is not public, all our experiments use the official development set as our test set, while splitting the training set into training (90%) and 2 validation sets (each contains 4% of the data). The first validation set (VAL-1) is used for tuning the parameters of the table-based models, while the second validation set (VAL-2) is used for tuning the parameters to combine text-based and table-based models. For clarity, in all the experiment variations we use the notation defined above where NQTables$_{\text{Tab}}$-Dev/Val-1/Val-2 is limited to predict answer spans only from tables, and NQTables-Dev/Val-1/Val-2 is able to predict answer spans from both text and tables.

The official evaluation script computes F1 scores and considers any questions that have at least 2 annotated answers as being answerable, while questions with 1 or no answer are unanswerable. An F1 score is calculated on the ability to predict a span that matches at least one of the annotations for the answerable questions, and to correctly predict unanswerable ones. Since the validation sets contain only a single annotation, the F1 score measure based on 5-way annotation cannot be used directly on VAL-1 or VAL-2. Therefore, in the table-based experiments on the VAL-1, we report both accuracy (percent of correctly predicted answers) and modified F1 score (F1*), where modified F1 is based on the match of the predicted answer to a single annotation. Finally, we report a string-based F1 score that accepts any exact string match of a predicted span to an answer when combining text and table models.

4.1 Table-based Results

First, we evaluate the proposed table representation approaches on the NQTables subset. In this set of experiments, we use the article's tables as our input, omitting direct usage of the article's text except in the form of context-aware updates of the model used in Section 3.2. The results on NQTables$_{\text{Tab}}$-

Model	Pretrained on WikiTables	Finetuned on	VAL-1		DEV
			F1*	Acc	F1
RoBERTa baseline	no	FullNQ	49.1	48.2	53.3
RoBERTa tables	no	tables only	52.5	51.2	56.8
`[SEP]` encoding	no	tables only	53.2	51.1	57.5
GNN	no	tables only	52.6	51.2	55.0
GNN	yes	tables only	54.4	52.9	56.7
GNN	yes	FullNQ → tables only	55.7	54.3	58.9
Context-aware + GNN	yes	FullNQ → tables only	56.7	55.4	60.0

Table 1: Evaluation of the table encodings and context-aware table representation on NQTables$_{\text{Tab}}$-Val$_1$ (VAL-1) and NQTables$_{\text{Tab}}$-Dev (DEV) sets of NQTables that contains tables input exclusively.

Val$_1$ and NQTables$_{\text{Tab}}$-Dev are presented in Table 1. First, we evaluate our model using a RoBERTa baseline, following (Alberti et al., 2019) with a RoBERTa pretrained model instead of BERT (line 1). This baseline is trained using the FullNQ corpus and contains both text and table inputs. We improve this baseline by finetuning on the NQTables part of the training dataset (line 2). Previous work on table representations using BERT has shown improvement from using `[SEP]` tokens to highlight cell boundaries (Hwang et al., 2019). This is the additional baseline we report in line 3. Since this baseline performs well, we combine the usage of `[SEP]` tokens with our models for the rest of the table-encoding experiments (lines 3-7). We evaluate our table representation in lines 4-6, where results in lines 5 and 6 are obtained by initially pretraining the additional weights introduced by the GNN using WikiTables, as described in Section 3.1.3. We also found that by finetuning on the full NQ dataset first and further finetuning on the NQTables subset, the results are substantially improved (lines 6 and 7). Finally, our context-aware model combined with the table encoding from line 6 achieves the best result in this set of experiments. The improvements of the GNN and textual context-aware models are statistically significant with $p < 0.01$ according to a Wilcoxon signed-rank test. In the Appendix we provide two examples — one where context-aware model prediction improves, and the other one where adding textual context hurts.

4.2 Combining Tables and Text

In this section, we describe experiments of combining text and table predictions. Since the official development set contains up to 5 annotations, some of those annotations can be associated with tables, while others are associated with text. Unlike in the previous section, where text-related answers are ignored, here we allow a match to either text- or table-based answers. For this purpose, we need to combine scores from the text-based and table-based models. For text predictions, we use the RoBERTa baseline that was trained on the FullNQ dataset; for the table predictions we use our best GNN model and the context-aware model (lines 6 and 7 in Table 1, respectively). Oracle analysis on the development set suggests that a linear score transformation in the case of the FullNQ is not effective. To combine the scores of the text model g_c and the table model g_t, we use grid search, as described in Section 3.3. The search for parameters α, β, and γ is done on VAL-2 containing questions with table answers, questions with text answers, and questions without answers, using the string-based F1 score instead of the span-based one to compensate for the lack of multiple annotations in the validation set. According to this metric, both answers in tables and text are considered correct if they have exact string match with the gold answer span in a table annotated in VAL-2.

We evaluate those models on both the NQTables-Dev and FullNQ-Dev, allowing both text and table answers. The results are presented in Table 2. Results from our RoBERTa baseline that was trained on FullNQ are shown in line 1. Then, as mentioned above, we combine this generic model with table-only models (lines 2 and 3). Our experiments suggest that our proposed best model that uses table encoding improves the F1 score on NQTables by 1.5 points. When the table+text model combination is used on the full NQ dataset, there is a small improvement from both table models, but the textual context attention model is comparable to the GNN model, increasing recall at the cost of reducing precision. This might be explained by the relatively small fraction of questions that have

	NQTables (span)			FullNQ (span)			FullNQ (str)		
Model	**F1**	**Prec**	**Rec**	**F1**	**Prec**	**Rec**	**F1**	**Prec**	**Rec**
Alberti et al. (2019)[3]	58.9	63.6	54.8	52.4	57.6	48.0	54.0	59.3	49.5
RoBERTa	62.5	65.0	60.2	54.4	60.3	49.6	56.2	62.3	51.2
GNN	63.1	64.6	61.7	**54.6**	60.2	50.0	**56.6**	62.3	51.8
Context-aware + GNN	**64.0**	65.5	62.6	**54.6**	59.4	50.6	**56.6**	61.5	52.4

Table 2: Results of combining text-based and table-based predictions from two models: F1, Precision, and Recall scores reported on NQTables-Dev and FullNQ-Dev. (span) and (str) indicate span-based and string-based scores.

	VAL-1	
	F1*	**Acc**
all relations	55.7	54.3
– cell-col	54.9	53.7
– cell-row	55.0	53.7
– in-cell	56.0	53.8
– cross-col	55.7	54.3
– cross-row	55.7	54.7

Table 3: Importance of relation types.

answers only in tables (8% of all questions) and the difficulty in calibrating answers from tables and text against each other.

4.2.1 Impact of Relation Types

In order to investigate which of the five types of table relations (defined in Section 3.1.2) help table representation the most we perform an ablation study where we use the baseline of the pretrained GNN model (line 6), but remove each type of relation. The results are presented in Table 3. The ablation study shows a clear importance of cell-column and cell-row relation types, where the performance of the model without each of those relations degrades. While the F1 score for the experiment without in-cell relation is higher, the accuracy is much lower. The cross-column relation does not seem to contribute to the overall performance while the cross-row relation degrades the accuracy.

4.2.2 Impact of Negative Sampling

In all our experiments using models finetuned on tables only data (lines 2-7 in Table 1 and lines 2-3 in Table 2), during training we used an equal proportion of positive and negative samples,[4] where all negative samples were taken from the articles in NQTables-Train, which contain an answer in a table. This approach is successful when models were evaluated on their ability to predict answers in tables, for articles that are known to contain such answers. On the other hand, when the table-based models are asked to make predictions for articles not known to contain answers in tables (or any answers), they tended to be over-confident in comparison to a generic text-based model, trained with negative examples across all articles. This over-confidence is evident from the extremely high selected confidence parameters of $\gamma = -0.005$ for GNN model (line 2 in Table 2) and $\gamma = -0.0025$ for context-aware model (line 3 in Table 2), suggesting that only high-value and high-confidence scores are considered from the table-based model.

To investigate the effect of the negative sampling method we perform an ablation study that compares three techniques: 1) sampling negative samples from within articles that contain an answer in tables, 2) random sampling of negatives across all NQ articles, in the proportion used by Alberti et al. (2019), and 3) sampling negatives with equal proportion from articles that contain answer in tables, and articles that do not. The results for the GNN model are shown in Table 4. While the first sampling strategy works best when a table-based model is used to predict answers from tables in NQTables, sampling negatives from a more diverse set of articles improves the overall FullNQ results for the combination of text and table-based models, by allowing better calibration between text and table models. The threshold values γ are seen to be much lower in these cases. However, the second and third strategies reduce performance on QA from tables. When optimizing the random sampling strategy for highest performance on FullNQ, no benefit was found from contextual text attention for table representations.

Finally, we compare the performance of our model to other work on the Natural Questions on both the short and long answer prediction tasks. In our model, the long answer is predicted based on

---

[4] A sample is a table or a table fragment, together with an indication of short answer span or NULL.

Sampling (pos : neg_within : neg_outside)	**NQTables VAL-1 (F1)**	**FullNQ DEV (F1)**	γ
(1) Equal sampling within positive article (1:1:0)	55.7	54.6	-0.005
(2) Random sampling across all articles (0.63:0.28:0.72)	52.5	55.1	-3
(3) Equal sampling within and across articles (2:1:1)	54.0	54.9	-0.5

Table 4: The effect of negative sampling technique on table-only models and text+table model combinations.

Model	Short F1	Long F1
Alberti et al. (2019)	52.7	64.7
Liu et al. (2020)	57.7	73.9
Ravula et al. (2020)	58.5	78.2
GNN, random sample (2)	55.1	65.9

Table 5: Comparison to other NQ models.

the segment that corresponds to the short answer prediction. The results are presented in Table 5. As we can see, our method obtains a substantial improvement over the baseline of Alberti et al. (2019) in short answer F1, and a smaller improvement in long answer F1. Advances in long answer F1 from state-of-the-art recent works are likely complementary to our method and can be integrated for additive gain.

5 Conclusion

Tables in Web documents are pervasive and can be directly used to answer many search queries. In this work, we presented an approach to enrich table representations using information from article text, and showed that it improves a state-of-the-art pretrained structure-aware table representation for question answering from tables. We also studied how to effectively combine text-based and table-based approaches. Finally, we performed the first study focusing on table QA for the Natural Questions dataset, and showed that improved representations of tables lead to performance gains. In future work, our methods can be applied to other QA datasets, such as WikiTableQuestions (Pasupat and Liang, 2015) and HybridQA (Chen et al., 2020).

Acknowledgments

This work was supported by the Google Faculty Research Awards Program. Computational resources were provided by the GCP research credits program. We thank the reviewers for their helpful feedback.

References

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A BERT baseline for the natural questions. *arXiv preprint arXiv:1901.08634*.

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *Proc. EMNLP*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proc. ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *Proc. NAACL*.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. InfoTabS: Inference on tables as semi-structured data. In *Proc. ACL*.

Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. *Proc. ACL*.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on WikiSQL with table-aware word contextualization. *In Proceedings of the NeurIPS Workshop on Knowledge Representation and Reasoning Meets Machine Learning*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proc. EMNLP*.

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

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Daxin Jiang, Jiancheng Lv, and Nan Duan. 2020. RikiNet: Reading wikipedia pages for natural question answering. *Proc. ACL*.

Thomas Müeller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *Proc. EMNLP-IJCNLP*.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proc. ACL*.

Anirudh Ravula, Chris Alberti, Joshua Ainslie, Li Yang, Philip Minh Pham, Qifan Wang, Santiago Ontanon, Sumit Kumar Sanghai, Vaclav Cvicek, and Zach Fisher. 2020. ETC: Encoding long and structured inputs in transformers. *Proc. EMNLP*.

Peter Shaw, Philip Massey, Angelica Chen, Francesco Piccinno, and Yasemin Altun. 2019. Generating logical forms from graph representations of text and entities. *Proc. ACL*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *Proc. NAACL*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *Proc. EMNLP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. *Proc. ACL*.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Bailin Wang, Yi Chern Tan, Xinyi Yang, Dragomir Radev, Richard Socher, and Caiming Xiong. 2020. GraPPa: Grammar-augmented pre-training for table semantic parsing. *arXiv preprint arXiv:2009.13845*.

A Examples

Question: What kind of poisonous snakes are in Michigan?

List of reptiles of Michigan

From Wikipedia, the free encyclopedia

Michigan, one of the fifty United States, is home to two types of lizards, nineteen types of snakes and eleven types of turtles, all members of the class Reptilia.[1] Reptiles are found throughout Michigan, although the only venomous species, the eastern massasauga rattlesnake, is seen only in the Lower Peninsula.[2] Reptiles are cold-blooded, and so usually pass the cold winters of Michigan in frost-free areas, such as burrows (for snakes and land-

Scientific name	Common name	Description	Image	Notes
Cnemidophorus sexlineatus	Six-lined racerunner	Adults are 6 to 10.5 inches (15 to 27 cm) in length and colored olive, gray, brown or black with six lighter-colored stripes.[9]		Found only in Tuscola County in eastern Michigan,[9] considered a threatened species[7]
Sistrurus catenatus catenatus	Eastern massasauga rattlesnake	Adults are 2 to 3 feet (0.61 to 0.91 m) in length and colored gray or brown with dark brown patches. Venomous.		A candidate species for listing on the federal endangered species list,[8] and a species of special concern in Michigan[7]

Figure 2: Example where incorporating textual context helps. In red square the prediction was made by GNN model without context attention (Table 1, line 6), in green square the prediction was made by context-aware GNN (Table 1, line 7).

The correct answer "Easter massasauga rattlesnake" was correctly predicted from the table when context-aware attention was used. The same answer can be extracted based on the textual information in the introduction paragraph. By propagating relevant information from the textual context to the table entries, the model was able to predict correctly the answer from the table. On the other hand, the table-only model which does not use the surrounding textual context incorrectly predicted "Cnemidophorus sexlineatus".

Question: Who sings the intro for the Phineas and Ferb?

Background [edit]

The artists on the soundtrack all are in *Phineas and Ferb* except for Bowling for Soup who performs the theme tune for the series. The song "Gitchee Gitchee Goo" performed by Vincent Martella & Ashley Tisdale featured in the episode "Flop Starz", was also included in the 2009 album *Disney Channel Playlist*.[2]

Length	38:04
Label	Walt Disney
Cast - Phineas and Ferb chronology	
Phineas and Ferb Soundtrack (2009)	*Phineas and Ferb Christmas Vacation!* (2009)

Release [edit]

The original report cited that the initial February 10, 2009 release date had been pushed back to August 4.[3] On June 26, 2009 the release date was pushed back to September 22, 2009.[4] Later this has been confirmed on the official website.[5] The soundtrack was released in the United Kingdom on October 10, 2009 via EMI Music.[6]

Track listing [edit]

No.	Title	Writer(s)	Performer(s)	Length
1.	"Today is Gonna Be a Great Day"	Dan Povenmire, Jeff "Swampy" Marsh, Michael Culross, Michael Walker, Carl Williams	Bowling for Soup	3:01
2.	"Gitchee Gitchee Goo"	Dan Povenmire, Jeff "Swampy" Marsh, Jon Colton Barry	Vincent Martella, Ashley Tisdale, Laura Dickinson, Danny Jacob	1:58
3.	"Backyard Beach"	Dan Povenmire, Danny Jacob	Danny Jacob	0:47
4.	"Busted"	Dan Povenmire, Jeff "Swampy"	Ashley Tisdale, Olivia Olson	1:38

Figure 3: Example where incorporating textual context hurts. In red square the incorrect prediction was made by context-aware GNN model (Table 1, line 7), in green square the correct prediction was made by GNN without context attention (Table 1, line 6).

The correct answer ”Bowling for Soup” was correctly predicted from the table by the model that does not use context attention, while the context-aware model predicts the answer that has higher number of links to the textual context.

PPT: Parsimonious Parser Transfer for Unsupervised Cross-Lingual Adaptation

Kemal Kurniawan[1] **Lea Frermann**[1] **Philip Schulz**[2*] **Trevor Cohn**[1]
[1]School of Computing and Information Systems, University of Melbourne
[2]Amazon Research
`kemal.kurniawan@student.unimelb.edu.au`
`lea.frermann@unimelb.edu.au`
`phschulz@amazon.com`
`tcohn@unimelb.edu.au`

Abstract

Cross-lingual transfer is a leading technique for parsing low-resource languages in the absence of explicit supervision. Simple 'direct transfer' of a learned model based on a multilingual input encoding has provided a strong benchmark. This paper presents a method for unsupervised cross-lingual transfer that improves over direct transfer systems by using their output as implicit supervision as part of self-training on unlabelled text in the target language. The method assumes minimal resources and provides maximal flexibility by (a) accepting any pre-trained arc-factored dependency parser; (b) assuming no access to source language data; (c) supporting both projective and non-projective parsing; and (d) supporting multi-source transfer. With English as the source language, we show significant improvements over state-of-the-art transfer models on both distant and nearby languages, despite our conceptually simpler approach. We provide analyses of the choice of source languages for multi-source transfer, and the advantage of non-projective parsing. Our code is available online.[1]

1 Introduction

Recent progress in natural language processing (NLP) has been largely driven by increasing amounts and size of labelled datasets. The majority of the world's languages, however, are low-resource, with little to no labelled data available (Joshi et al., 2020). Predicting linguistic labels, such as syntactic dependencies, underlies many downstream NLP applications, and the most effective systems rely on labelled data. Their lack hinders the access to NLP technology in many languages. One solution is cross-lingual model transfer, which adapts models trained on high-resource languages to low-resource ones. This paper presents a flexible framework for cross-lingual transfer of syntactic dependency parsers which can leverage *any* pre-trained arc-factored dependency parser, and assumes no access to labelled target language data.

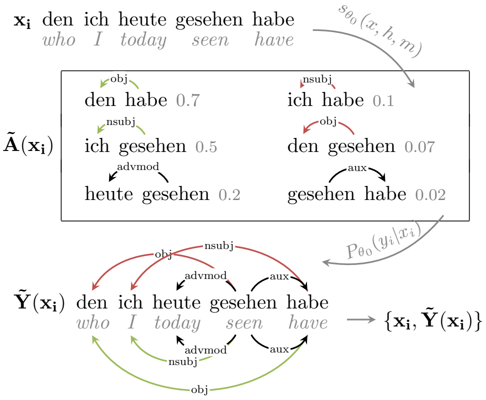


Figure 1: Illustration of our technique. For a target language sentence (x_i), a source parser P_{θ_0} predicts a set of candidate arcs $\tilde{A}(x_i)$ (subset shown in the figure), and parses $\tilde{Y}(x_i)$. The highest scoring parse is shown on the bottom (green), and the true gold parse (unknown to the parser) on top (red). A target language parser P_θ is then fine-tuned on a data set of ambiguously labelled sentences $\{x_i, \tilde{Y}(x_i)\}$.

One straightforward method of cross-lingual parsing is direct transfer. It works by training a parser on the source language labelled data and subsequently using it to parse the target language directly. Direct transfer is attractive as it does not require labelled target language data, rendering the approach fully unsupervised.[2] Recent work has shown that it is possible to outperform direct transfer if unlabelled data, either in the target lan-

---

[*] Work done outside Amazon.

[1] `https://github.com/kmkurn/ppt-eacl2021`

[2] Direct transfer is also called zero-shot transfer or model transfer in the literature.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2907–2918
April 19 - 23, 2021.

guage or a different auxiliary language, is available (He et al., 2019; Meng et al., 2019; Ahmad et al., 2019b). Here, we focus on the former setting and present flexible methods that can adapt a pre-trained parser given unlabelled target data.

Despite their success in outperforming direct transfer by leveraging unlabelled data, current approaches have several drawbacks. First, they are limited to generative and projective parsers. However, discriminative parsers have proven more effective, and non-projectivity is a prevalent phenomenon across the world's languages (de Lhoneux, 2019). Second, prior methods are restricted to single-source transfer, however, transfer from multiple source languages has been shown to lead to superior results (McDonald et al., 2011; Duong et al., 2015a; Rahimi et al., 2019). Third, they assume access to the source language data, which may not be possible because of privacy or legal reasons. In such source-free transfer, only a pre-trained source parser may be provided.

We address the three shortcomings with an alternative method for unsupervised target language adaptation (Section 2). Our method uses high probability edge predictions of the source parser as a supervision signal in a self-training algorithm, thus enabling unsupervised training on the target language data. The method is feasible for discriminative and non-projective parsing, as well as multi-source and source-free transfer. Building on a framework introduced in Täckström et al. (2013), this paper for the first time demonstrates their effectiveness in the context of state-of-the-art neural dependency parsers, and their generalizability across parsing frameworks. Using English as the source language, we evaluate on eight distant and ten nearby languages (He et al., 2019). The single-source transfer variant (Section 2.1) outperforms previous methods by up to 11 % UAS, averaged over nearby languages. Extending the approach to multi-source transfer (Section 2.2) gives further gains of 2 % UAS and closes the performance gap against the state of the art on distant languages. In short, our contributions are:

1. A conceptually simple and highly flexible framework for unsupervised target language adaptation, which supports multi-source and source-free transfer, and can be employed with any pre-trained state-of-the-art arc-factored parser(s);
2. Generalisation of the method of Täckström et al. (2013) to state-of-the-art, non-projective dependency parsing with neural networks;
3. Up to 13 % UAS improvement over state-of-the-art models, considering nearby languages, and roughly equal performance over distant languages; and
4. Analysis of the impact of choice of source languages on multi-source transfer quality.

2 Supervision via Transfer

In our scenario of unsupervised cross-lingual parsing, we assume the availability of a pre-trained source parser, and unlabelled text in the target language. Thus, we aim to leverage this data such that our cross-lingual transfer parsing method out-performs direct transfer. One straightforward method is self-training where we use the predictions from the source parser as supervision to train the target parser. This method may yield decent performance as direct transfer is fairly good to begin with. However, we may be able to do better if we also consider a set of parse trees that have high probability under the source parser (cf. Fig. 1 for illustration).

If we assume that the source parser can produce a set of possible trees instead, then it is natural to use all of these trees as supervision signal for training. Inspired by Täckström et al. (2013), we formalise the method as follows. Given an unlabelled dataset $\{x_i\}_{i=1}^{n}$, the training loss can be expressed as

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \log \sum_{y \in \tilde{Y}(x_i)} P_\theta(y|x_i) \qquad (1)$$

where θ is the target parser parameters and $\tilde{Y}(x_i)$ is the set of trees produced by the source parser. Note that $\tilde{Y}(x_i)$ must be smaller than the set of all trees spanning x (denoted as $\mathcal{Y}(x_i)$) because $\mathcal{L}(\theta) = 0$ otherwise. This training procedure is a form of self-training, and we expect that the target parser can learn the correct tree as it is likely to be included in $\tilde{Y}(x_i)$. Even if this is not the case, as long as the correct arcs occur quite frequently in $\tilde{Y}(x_i)$, we expect the parser to learn a useful signal.

We consider an arc-factored neural dependency parser where the score of a tree is defined as the sum of the scores of its arcs, and the arc scoring function is parameterised by a neural network. The probability of a tree is then proportional to its score.

Formally, this formulation can be expressed as

$$P_\theta(y|x) = \frac{\exp s_\theta(x, y)}{Z(x)} \tag{2}$$

$$s_\theta(x, y) = \sum_{(h,m)\in A(y)} s_\theta(x, h, m) \tag{3}$$

where $Z(x) = \sum_{y\in\mathcal{Y}(x)} \exp s_\theta(x, y)$ is the partition function, $A(y)$ is the set of head-modifier arcs in y, and $s_\theta(x, y)$ and $s_\theta(x, h, m)$ are the tree and arc scoring function respectively.

2.1 Single-Source Transfer

Here, we consider the case where a single pre-trained source parser is provided and describe how the set of trees is constructed. Concretely, for every sentence $x = w_1, w_2, \dots, w_t$ in the target language data, using the source parser, the set of high probability trees $\tilde{Y}(x)$ is defined as the set of dependency trees that can be assembled from the high probability arcs set $\tilde{A}(x) = \bigcup_{m=1}^{t} \tilde{A}(x, m)$, where $\tilde{A}(x, m)$ is the set of high probability arcs whose dependent is w_m. Thus, $\tilde{Y}(x)$ can be expressed formally as

$$\tilde{Y}(x) = \{y|y \in \mathcal{Y}(x) \wedge A(y) \subseteq \tilde{A}(x)\}. \tag{4}$$

$\tilde{A}(x, m)$ is constructed by adding arcs (h, m) in order of decreasing arc marginal probability until their cumulative probability exceeds a threshold σ (Täckström et al., 2013). The predicted tree from the source parser is also included in $\tilde{Y}(x)$ so the chart is never empty. This prediction is simply the highest scoring tree. This procedure is illustrated in Fig. 1.

Since $\mathcal{Y}(x)$ contains an exponential number of trees, efficient algorithms are required to compute the partition function $Z(x)$, arc marginal probabilities, and the highest scoring tree. First, arc marginal probabilities can be computed efficiently with dynamic programming for projective trees (Paskin, 2001) and Matrix-Tree Theorem for the non-projective counterpart (Koo et al., 2007; McDonald and Satta, 2007; Smith and Smith, 2007). The same algorithms can also be employed to compute $Z(x)$. Next, the highest scoring tree can be obtained efficiently with Eisner's algorithm (Eisner, 1996) or the maximum spanning tree algorithm (McDonald et al., 2005; Chu and Liu, 1965; Edmonds, 1967) for the projective and non-projective cases, respectively.

The transfer is performed by initialising the target parser with the source parser's parameters and then fine-tuning it with the training loss in Eq. (1) on the target language data. Following previous works (Duong et al., 2015b; He et al., 2019), we also regularise the parameters towards the initial parameters to prevent them from deviating too much since the source parser is already good to begin with. Thus, the final fine-tuning loss becomes

$$\mathcal{L}'(\theta) = \mathcal{L}(\theta) + \lambda||\theta - \theta_0||_2^2 \tag{5}$$

where θ_0 is the initial parameters and λ is a hyperparameter regulating the strength of the L_2 regularisation. This single-source transfer strategy was introduced as ambiguity-aware self-training by Täckström et al. (2013). A difference here is that we regularise the target parser's parameters against the source parser's as the initialiser, and apply the technique to modern lexicalised state-of-the-art parsers. We refer to this transfer strategy as PPT hereinafter.

Note that the whole procedure of PPT can be performed even when the source parser is trained with monolingual embeddings. Specifically, given a source parser trained *only on monolingual embeddings*, one can align pre-trained target language word embeddings to the source embedding space using an offline cross-lingual alignment method (e.g., of Smith et al. (2017)), and use the aligned target embeddings with the source model to compute $\tilde{Y}(x)$. Thus, our method can be used with any pre-trained monolingual neural parser.

2.2 Multi-Source Transfer

We now consider the case where multiple pre-trained source parsers are available. To extend PPT to this multi-source case, we employ the ensemble training method from Täckström et al. (2013), which we now summarise. We define $\tilde{A}(x, m) = \bigcup_k \tilde{A}_k(x, m)$ where $\tilde{A}_k(x, m)$ is the set of high probability arcs obtained with the k-th source parser. The rest of the procedure is exactly the same as PPT. Note that we need to select one source parser as the main source to initialise the target parser's parameters with. Henceforth, we refer to this method as PPTX.

Multiple source parsers may help transfer better because each parser will encode different syntactic biases from the languages they are trained on. Thus, it is more likely for one of those biases to match that of the target language instead of using just a single source parser. However, multi-source transfer may also hurt performance if the languages have very different syntax, or the source parsers are of poor quality, which can arise from poor quality cross-lingual word embeddings.

3 Experiments

3.1 Setup

We run our experiments on Universal Dependency Treebanks v2.2 (Nivre et al., 2018). We reimplement the self-attention graph-based parser of Ahmad et al. (2019a) that has been used with success for cross-lingual dependency parsing. Averaged over 5 runs, our reimplementation achieves 88.8 % unlabelled attachment score (UAS) on English Web Treebank using the same hyperparameters,[3] slightly below their reported 90.3 % result.[4] We select the run with the highest labelled attachment score (LAS) as the source parser. We obtain cross-lingual word embeddings with the offline transformation of Smith et al. (2017) applied to fastText pre-trained word vectors (Bojanowski et al., 2017). We include the universal POS tags as inputs by concatenating the embeddings with the word embeddings in the input layer. We acknowledge that the inclusion of gold POS tags does not reflect a realistic low-resource setting where gold tags are not available, which we discuss more in Section 3.3. We evaluate on 18 target languages that are divided into two groups, distant and nearby languages, based on their distance from English as defined by He et al. (2019).[5]

During the unsupervised fine-tuning, we compute the training loss over all trees regardless of projectivity (i.e. we use Matrix-Tree Theorem to compute Eq. (1)) and discard sentences longer than 30 tokens to avoid out-of-memory error. Following He et al. (2019), we fine-tune on the target language data for 5 epochs, tune the hyperparameters (learning rate and λ) on Arabic and Spanish using LAS, and use these values[6] for the distant and nearby languages, respectively. We set the threshold $\sigma = 0.95$ for both PPT and PPTX following Täckström et al. (2013). We keep the rest of the hyperparameters (e.g., batch size) equal to those of Ahmad et al. (2019a). For PPTX, unless otherwise stated, we consider a leave-one-out scenario where we use all languages except the target as the source language. We use the same hyperparameters as the English parser to train these non-English source parsers and set the English parser as the main source.

3.2 Comparisons

We compare PPT and PPTX against several recent unsupervised transfer systems. First, HE is a neural lexicalised DMV parser with normalising flow that uses a language modelling objective when fine-tuning on the unlabelled target language data (He et al., 2019). Second, AHMAD is an adversarial training method that attempts to learn language-agnostic representations (Ahmad et al., 2019b). Lastly, MENG is a constrained inference method that derives constraints from the target corpus statistics to aid inference (Meng et al., 2019). We also compare against direct transfer (DT) and self-training (ST) as our baseline systems.[7]

3.3 Results

Table 1 shows the main results. We observe that fine-tuning via self-training already helps DT, and by incorporating multiple high probability trees with PPT, we can push the performance slightly higher on most languages, especially the nearby ones. Although not shown in the table, we also find the PPT has up to 6x lower standard deviation than ST, which makes PPT preferrable to ST. Thus, we exclude ST as a baseline from our subsequent experiments. Our results seem to agree with that of Täckström et al. (2013) and suggest that PPT can also be employed for neural parsers. Therefore, it should be considered for target language adaptation if unlabelled target data is available. Comparing to HE (He et al., 2019), PPT performs worse on distant languages, but better on nearby languages. This finding means that if the target language has a closely related high-resource language, it may be better to transfer from that language as the source and use PPT for adaptation. Against AHMAD (Ahmad et al., 2019b), PPT performs better on 4 out of 6 distant languages. On nearby languages, the average UAS of PPT is higher, and the average LAS is on par. This result shows that leveraging unlabelled data for cross-lingual parsing without access to the source data is feasible. PPT also performs

---

[3] Reported in Table 4.

[4] UAS and LAS are reported excluding punctuation tokens.

[5] We exclude Japanese and Chinese based on Ahmad et al. (2019a), who reported atypically low performance on these two languages, which they attributed to the low quality of their cross-lingual word embeddings. In subsequent work they excluded these languages (Ahmad et al., 2019b).

[6] Reported in Table 5.

[7] ST requires significantly less memory so we only discard sentences longer than 60 tokens. Complete hyperparameter values are shown in Table 5.

Target	UAS							LAS				
	DT	ST	PPT	PPTX	He	Ahmad	Meng	DT	ST	PPT	PPTX	Ahmad
fa	37.5	38.0	39.5	53.6	**63.2**	—	—	29.2	30.5	31.6	**44.5**	—
ar†	37.6	39.2	39.5	48.3	**55.4**	39.0	47.3	27.3	30.0	29.9	**38.5**	27.9
id	51.6	49.9	50.3	**71.9**	64.2	51.6	53.1	45.2	44.4	44.7	**59.0**	45.3
ko	35.1	37.1	**37.5**	34.6	37.0	34.2	37.1	16.6	**18.2**	18.0	16.1	16.1
tr	36.9	38.1	**39.2**	38.4	36.1	—	35.2	18.5	19.5	19.0	**20.6**	—
hi	33.7	34.7	34.0	36.4	33.2	37.4	**52.4**	25.4	26.6	26.4	**28.3**	28.0
hr	62.0	63.4	63.8	**71.9**	65.3	63.1	63.7	51.9	54.2	54.2	**61.2**	53.6
he	56.6	59.2	60.5	64.2	**64.8**	57.2	58.8	47.6	50.5	51.1	**53.9**	49.4
average	43.9	45.0	45.5	**52.4**	**52.4**	—	—	32.7	34.2	34.4	**40.3**	—
bg	77.7	80.0	81.2	**81.9**	73.6	79.7	79.7	66.2	68.9	70.0	**70.2**	68.4
it	77.9	79.7	81.4	**83.7**	70.7	80.7	82.0	71.1	74.0	75.5	**77.7**	75.6
pt	74.1	76.3	77.1	**81.0**	66.6	77.1	77.5	65.1	67.6	68.3	**70.6**	67.8
fr	74.8	77.5	78.6	**80.6**	67.7	78.3	79.1	68.1	71.7	72.8	**74.5**	73.3
es†	72.5	74.9	75.2	**78.3**	64.3	74.1	75.8	63.8	66.5	67.0	**69.2**	65.8
no	77.9	80.4	**81.2**	80.0	65.3	81.0	80.4	69.1	71.9	72.7	71.8	**73.1**
da	75.3	76.0	**77.3**	76.6	61.1	76.3	76.6	66.3	67.4	**68.6**	67.9	68.0
sv	78.9	80.5	**82.1**	81.0	64.4	80.4	80.5	71.1	72.7	74.2	72.7	**76.7**
nl	68.0	68.9	69.9	**74.4**	61.7	69.2	67.6	59.5	60.7	61.5	**65.4**	60.5
de	66.8	69.9	69.5	**74.1**	69.5	71.1	70.8	56.4	60.0	59.7	**63.5**	61.8
average	74.4	76.4	77.4	**79.1**	66.5	76.8	77.0	65.7	68.1	69.0	**70.3**	69.1

Table 1: Test UAS and LAS (avg. 5 runs) on distant (top) and nearby (bottom) languages, sorted from most distant (fa) to closest (de) to English. PPTX is trained in a leave-one-out fashion. The numbers for He, Ahmad, and Meng are obtained from the corresponding papers, direct transfer (DT) and self-training (ST) are based on our own implementation. † indicates languages used for hyper-parameter tuning, and thus have additional supervision through the use of a labelled development set.

better than Meng (Meng et al., 2019) on 4 out of 7 distant languages, and slightly better on average on nearby languages. This finding shows that PPT is competitive to their constrained inference method.

Also reported in Table 1 are the ensemble results for PPTX, which are particularly strong. PPTX outperforms PPT, especially on distant languages with the average UAS and LAS absolute improvements of 7 % and 6 % respectively. This finding suggests that PPTX is indeed an effective method for multi-source transfer of neural dependency parsers. It also gives further evidence that multi-source transfer is better than the single-source counterpart. PPTX also closes the gap against the state-of-the-art adaptation of He et al. (2019) in terms of average UAS on distant languages. This result suggests that PPTX can be an option for languages that do not have a closely related high-resource language to transfer from.

Treebank Leakage The success of our cross-lingual transfer can be attributed in part to treebank leakage, which measures the fraction of dependency trees in the test set that are isomorphic to a tree in the training set (with potentially different words); accordingly these trees are not entirely unseen. Such leakage has been found to be a particularly strong predictor for parsing performance in monolingual parsing (Søgaard, 2020). Fig. 2 shows the relationship between treebank leakage and parsing accuracy, where the leakage is computed between the English training set as source and the target language's test set. Excluding outliers which are Korean and Turkish because of their low parsing accuracy despite the relatively high leakage, we find that there is a fairly strong positive correlation ($r = 0.57$) between the amount of

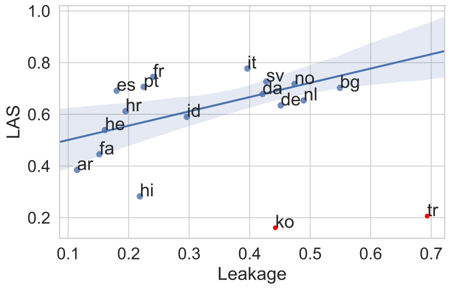


Figure 2: Relationship between treebank leakage and LAS for PPTX. Shaded area shows 95 % confidence interval. Korean and Turkish (in red) are excluded when computing the regression line.

leakage and accuracy. The same trend occurs with DT, ST, and PPT. This finding suggests that cross-lingual parsing is also affected by treebank leakage just like monolingual parsing is, which may present an opportunity to find good sources for transfer.

Use of Gold POS Tags As we explained in Section 3.1, we restrict our experiments to gold POS tags for comparison with prior work. However, the use of gold POS tags does not reflect a realistic low-resource setting where one may have to resort to automatically predicted POS tags. Tiedemann (2015) has shown that cross-lingual delexicalised parsing performance degrades when predicted POS tags are used. The degradation ranges from 2.9 to 8.4 LAS points depending on the target language. Thus, our reported numbers in Table 1 are likely to decrease as well if predicted tags are used, although we expect the decline is not as sharp because our parser is lexicalised.

3.4 Parsimonious Selection of Sources for PPTX

In our main experiment, we use all available languages as source for PPTX in a leave-one-out setting. Such a setting may be justified to cover as many syntactic biases as possible, however, training dozens of parses may be impractical. In this experiment, we consider the case where we can train only a handful of source parsers. We investigate two selections of source languages: (1) a representative selection (PPTX-REPR) which covers as many language families as possible and (2) a pragmatic selection (PPTX-PRAG) containing truly high-resource languages for which quality pre-trained parsers are likely to exist. We restrict the selections to 5 languages each. For PPTX-REPR, we use English, Spanish, Arabic, Indonesian, and Korean as source languages. This selection covers Indo-European (Germanic and Romance), Afro-Asiatic, Austronesian, and Koreanic language families respectively. We use English, Spanish, Arabic, French, and German as source languages for PPTX-PRAG. The five languages are classified as exemplary high-resource languages by Joshi et al. (2020). We exclude a language from the source if it is also the target language, in which case there will be only 4 source languages. Other than that, the setup is the same as that of our main experiment.[8]

We present the result in Fig. 3 where we also include the results for PPT, and PPTX with the leave-one-out setting (PPTX-LOO). We report only LAS since UAS shows a similar trend. We observe that both PPTX-REPR and PPTX-PRAG outperform PPT overall. Furthermore, on nearby languages except Dutch and German, both PPTX-REPR and PPTX-PRAG outperform PPTX-LOO, and PPTX-PRAG does best overall. In contrast, no systematic difference between the three PPTX variants emerges on distant languages. This finding suggests that instead of training dozens of source parsers for PPTX, training just a handful of them is sufficient, and a "pragmatic" selection of a small number of high-resource source languages seems to be an efficient strategy. Since pre-trained parsers for these languages are most likely available, it comes with the additional advantage of alleviating the need to train parsers at all, which makes our method even more practical.

Analysis on Dependency Labels Next, we break down the performance of our methods based on the dependency labels to study their failure and success patterns. Fig. 4 shows the UAS of DT, PPT, and PPTX-PRAG on Indonesian and German for select dependency labels.

Looking at Indonesian, PPT is slightly worse than DT in terms of overall accuracy scores (Table 1), and this is reflected across dependency labels. However, we see in Fig. 4 that PPT outperforms DT on `amod`. In Indonesian, adjectives follow the noun they modify, while in English the opposite is true in general. Thus, unsupervised target language adaptation seems able to address these kinds of discrepancy between the source and target language. We find that PPTX-PRAG outperforms both DT and PPT across dependency labels, especially on `flat` and `compound` labels as shown in Fig. 4. Both labels are related to multi-word expressions (MWEs), so PPTX appears to improve parsing MWEs in Indonesian significantly.

For German we find that both PPT and PPTX-PRAG outperform DT on most dependency labels, with the most notable gain on `nmod`, which appear in diverse, and often non-local relations in both languages many of which do not structurally translate, and fine-tuning improves performance as expected. Also, we see PPTX-PRAG significantly underperforms on `compound` while PPT is better than DT. German compounds are often merged into a single token, and self-training appears to alleviate over-prediction of such relations. The multi-source case may contain too much diffuse

---

[8] Hyperparameters are tuned; values are shown in Table 5.

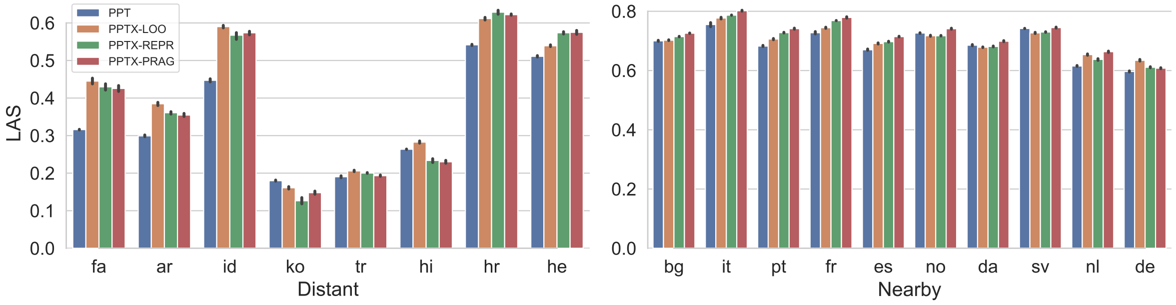


Figure 3: Comparison of selection of source languages for PPTX on distant and nearby languages, sorted from most distant (fa) to closest (de) to English. PPTX-LOO is trained in a leave-one-out fashion. PPTX-REPR uses the representative source language set, while PPTX-PRAG is adapted from five high-resource languages. A source language is excluded from the source if it is also the target language.

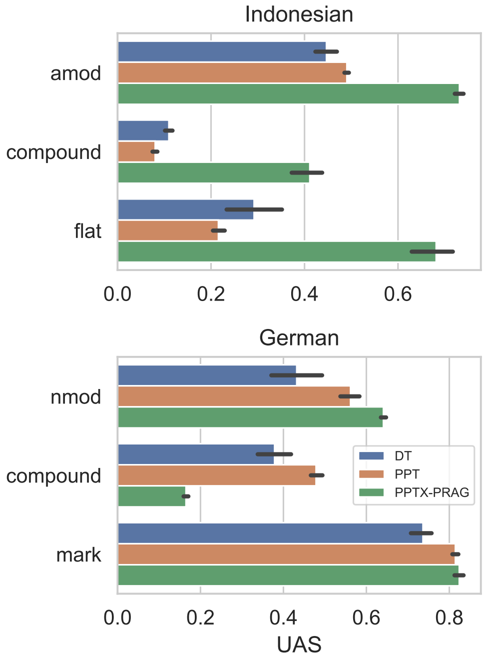


Figure 4: Comparison of direct transfer (DT), PPT, and PPTX-PRAG on select dependency labels of Indonesian (top) and German (bottom).

signal on `compound` and thus the performance is worse than that of DT. We find that PPT and PPTX improves over DT on `mark`, likely because markers are often used in places where German deviates from English by becoming verb-final (e.g., subordinate clauses). Both PPT and PPTX-PRAG seem able to learn this characteristic as shown by their performance improvements. This analysis suggests that the benefits of self-training depend on the syntactic properties of the target language.

Model	Target				AVG
	id	hr	fr	nl	
	Non-projective				
DT	45.2	51.9	68.1	59.5	56.2
PPT	44.7	54.2	72.8	61.5	58.3
PPTX-PRAG	57.4	62.2	77.9	66.4	66.0
	Projective				
DT	45.7	52.1	68.4	59.6	56.4
PPT	45.0	54.0	72.3	61.7	58.3
PPTX-PRAG	57.5	61.1	78.1	67.7	66.1

Table 2: Comparison of projective and non-projective direct transfer (DT), PPT, and PPTX-PRAG. Scores are LAS, averaged over 5 runs.

3.5 Effect of Projectivity

In this experiment, we study the effect of projectivity on the performance of our methods. We emulate a projective parser by restricting the trees in $\tilde{Y}(x)$ to be projective. In other words, the sum in Eq. (1) is performed only over projective trees. At test time, we search for the highest scoring projective tree. We compare DT, PPT, and PPTX-PRAG, and report LAS on Indonesian (id) and Croatian (hr) as distant languages, and on French (fr) and Dutch (nl) as nearby languages. The trend for UAS and on the other languages is similar. We use the dynamic programming implementation provided by `torch-struct` for the projective case (Rush, 2020). We find that it consumes more memory than our Matrix-Tree Theorem implementation, so we set the length cutoff to 20 tokens.[9]

Table 2 shows result of our experiment, which suggests that there is no significant performance difference between the projective and non-projective

---

[9] Hyperparameters are tuned; values are shown in Table 5.

Model	Target	
	ar	es
DT	28.1	64.1
PPT	30.8	67.3
PPTX^{EN5}	30.9	66.3
PPTX-PRAG^{S}	36.5	70.3
PPTX-PRAG	36.5	71.9

Table 3: Comparison of LAS on Arabic and Spanish on the development set, averaged over 5 runs. PPTX^{EN5} is PPTX with 5 English parsers as source, each trained on 1/5 size of the English corpus. PPTX-PRAG^{S} is PPTX with the pragmatic selection of source languages (PPTX-PRAG) but each source parser is trained on the same amount of data as PPTX^{EN5}.

variant of our methods. This result suggests that our methods generalise well to both projective and non-projective parsing. That said, we recommend the non-projective variant as it allows better parsing of languages that are predominantly non-projective. Also, we find that it runs roughly 2x faster than the projective variant in practice.

3.6 Disentangling the Effect of Ensembling and Larger Data Size

The effectiveness of PPTX can be attributed to at least three factors: (1) the effect of ensembling source parsers (*ensembling*), (2) the effect of larger data size used for training the source parsers (*data*), and (3) the diversity of syntactic biases from multiple source languages (*multilinguality*). In this experiment, we investigate to what extent each of those factors contributes to the overall performance. To this end, we design two additional comparisons: PPTX^{EN5} and PPTX-PRAG^{S}.

PPTX^{EN5} is PPTX with only English source parsers, where each parser is trained on 1/5 of the English training set. That is, we randomly split the English training set into five equal-sized parts, and train a separate parser on each. These parsers then serve as the source parsers for PPTX^{EN5}. Thus, PPTX^{EN5} has the benefit of ensembling but not data and multilinguality compared with PPT.

PPTX-PRAG^{S} is PPTX whose source language selection is the same as PPTX-PRAG, but each source parser is trained on the training data whose size is roughly the same as that of the training data of PPTX^{EN5} source parsers. In other words, the training data size is roughly equal to 1/5 of the English training set. To obtain this data, we randomly sub-sample the training data of each source language to the appropriate number of sentences. Therefore, PPTX-PRAG^{S} has the benefit of ensembling and multilinguality but not data.

Table 3 reports their LAS on the development set of Arabic and Spanish, averaged over five runs. We also include the results of PPTX-PRAG that enjoys all three benefits. We observe that PPT and PPTX^{EN5} perform similarly on Arabic, and PPTX^{EN5} has a slightly lower performance on Spanish. This result suggests a negligable effect of ensembling on performance. On the other hand, PPTX-PRAG^{S} outperforms PPTX^{EN5} remarkably, with approximately 6 % and 4 % LAS improvement on Arabic and Spanish respectively, showing that multilinguality has a much larger effect on performance than ensembling. Lastly, we see that PPTX-PRAG performs similarly to PPTX-PRAG^{S} on Arabic, and about 1.6 % better on Spanish. This result demonstrates that data size has an effect, albeit a smaller one compared to multilinguality. To conclude, the effectiveness of PPTX can be attributed to the diversity contributed through multiple languages, and not to ensembling or larger source data sets.

4 Related Work

Cross-lingual dependency parsing has been extensively studied in NLP. The approaches can be grouped into two main categories. On the one hand, there are approaches that operate on the data level. Examples of this category include annotation projection, which aims to project dependency trees from a source language to a target language (Hwa et al., 2005; Li et al., 2014; Lacroix et al., 2016; Zhang et al., 2019); and source treebank reordering, which manipulates the source language treebank to obtain another treebank whose statistics approximately match those of the target language (Wang and Eisner, 2018; Rasooli and Collins, 2019). Both methods have no restriction on the type of parsers as they are only concerned with the data. Transferring from multiple source languages with annotation projection is also feasible (Agić et al., 2016).

Despite their effectiveness, these data-level methods may require access to the source language data, hence are unusable when it is inaccessible due to privacy or legal reasons. In such source-free transfer, only a model pre-trained on the source language data is available. By leveraging parallel data, annotation projection is indeed feasible without access to the source language data. That said, parallel data is limited for low-resource languages or may have a poor domain match. Additionally, these methods involve training the parser from scratch for every new target language, which may be prohibitive.

On the other hand, there are methods that operate on the model level. A typical approach is direct transfer (aka., zero-shot transfer) which trains a parser on source language data, and then directly uses it to parse a target language. This approach is enabled by the shared input representation between the source and target language such as POS tags (Zeman and Resnik, 2008) or cross-lingual embeddings (Guo et al., 2015; Ahmad et al., 2019a). Direct transfer supports source-free transfer and only requires training a parser once on the source language data. In other words, direct transfer is unsupervised as far as target language resources.

Previous work has shown that unsupervised target language adaptation outperforms direct transfer. Recent work by He et al. (2019) used a neural lexicalised dependency model with valence (DMV) (Klein and Manning, 2004) as the source parser and fine-tuned it in an unsupervised manner on the unlabelled target language data. This adaptation method allows for source-free transfer and performs especially well on distant target languages. A different approach is proposed by Meng et al. (2019), who gathered target language corpus statistics to derive constraints to guide inference using the source parser. Thus, this technique also allows for source-free transfer. A different method is proposed by Ahmad et al. (2019b) who explored the use of unlabelled data from an auxiliary language, which can be different from the target language. They employed adversarial training to learn language-agnostic representations. Unlike the others, this method can be extended to support multi-source transfer. An older method is introduced by Täckström et al. (2013), who leveraged ambiguity-aware training to achieve unsupervised target language adaptation. Their method is usable for both source-free and multi-source transfer. However, to the best of our knowledge, its use for neural dependency parsing has not been investigated. Our work extends theirs by employing it for the said purpose.

The methods of both He et al. (2019) and Ahmad et al. (2019b) have several limitations. The method of He et al. (2019) requires the parser to be generative and projective. Their generative parser is quite impoverished with an accuracy that is 21 points lower than a state-of-the-art discriminative arc-factored parser on English. Thus, their choice of generative parser may constrain its potential performance. Furthermore, their method performs substantially worse than direct transfer on nearby target languages. Because of the availability of resources such as Universal Dependency Treebanks (Nivre et al., 2018), it is likely that a target language has a closely related high-resource language which can serve as the source language. Therefore, performing well on nearby languages is more desirable pragmatically. On top of that, it is unclear how to employ this method for multi-source transfer. The adversarial training method of Ahmad et al. (2019b) does not suffer from the aforementioned limitations but is unusable for source-free transfer. That is, it assumes access to the source language data, which may not always be feasible due to privacy or legal reasons.

5 Conclusions

This paper presents a set of effective, flexible, and conceptually simple methods for unsupervised cross-lingual dependency parsing, which can leverage the power of state-of-the-art pre-trained neural network parsers. Our methods improve over direct transfer and strong recent unsupervised transfer models, by using source parser uncertainty for implicit supervision, leveraging only unlabelled data in the target language. Our experiments show that the methods are effective for both single-source and multi-source transfer, free from the limitations of recent transfer models, and perform well for non-projective parsing. Our analysis shows that the effectiveness of the multi-source transfer method is attributable to its ability to leverage diverse syntactic signals from source parsers from different languages. Our findings motivate future research into advanced methods for generating informative sets of candidate trees given one or more source parsers.

Acknowledgments

We thank the anonymous reviewers for the useful feedback. A graduate research scholarship is provided by Melbourne School of Engineering to Kemal Kurniawan.

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*, 4:301–312.

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019a. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Kai-Wei Chang, and Nanyun Peng. 2019b. Cross-lingual dependency parsing with unlabeled auxiliary languages. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 372–382.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Miryam de Lhoneux. 2019. *Linguistically Informed Neural Dependency Parsing for Typologically Diverse Languages*. Ph.d. thesis, Uppsala University.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015a. Cross-lingual transfer for unsupervised dependency parsing without parallel data. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 113–122.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015b. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 845–850.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1234–1244.

Junxian He, Zhisong Zhang, Taylor Berg-Kirkpatrick, and Graham Neubig. 2019. Cross-lingual syntactic transfer through unsupervised adaptation of invertible projections. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3211–3223.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3):311–325.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 478–485.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150.

Ophélie Lacroix, Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. 2016. Frustratingly easy cross-lingual transfer for transition-based dependency parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1058–1063.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Soft cross-lingual syntax projection for dependency parsing. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 783–793.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72.

Ryan McDonald and Giorgio Satta. 2007. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 121–132.

Tao Meng, Nanyun Peng, and Kai-Wei Chang. 2019. Target language-aware constrained inference for cross-lingual dependency parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1117–1128.

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. 2018. Universal dependencies 2.2. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Mark A Paskin. 2001. *Cubic-time parsing and learning algorithms for grammatical bigram models*.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164.

Mohammad Sadegh Rasooli and Michael Collins. 2019. Low-resource syntactic transfer with unsupervised source reordering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3845–3856.

Alexander Rush. 2020. Torch-struct: Deep structured prediction library. In *Proceedings of the 58th annual meeting of the association for computational linguistics: System demonstrations*, pages 335–342.

David A. Smith and Noah A. Smith. 2007. Probabilistic models of nonprojective dependency trees. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 132–140.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *International Conference on Learning Representations*.

Anders Søgaard. 2020. Some languages seem easier to parse because their treebanks leak. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2765–2770.

Oscar Täckström, Ryan McDonald, and Joakim Nivre. 2013. Target language adaptation of discriminative transfer parsers. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1061–1071.

Jörg Tiedemann. 2015. Cross-lingual dependency parsing with universal dependencies and predicted PoS labels. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 340–349.

Dingquan Wang and Jason Eisner. 2018. Synthetic data made to order: The case of parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1337.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2019. Cross-lingual dependency parsing using code-mixed treebank. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 996–1005.

A Hyperparameter values

Here we report the hyperparameter values for experiments presented in the paper. Table 4 shows the hyperparameter values of our English source parser explained in Section 3.1. Table 5 reports the tuned hyperparameter values for our experiments shown in Table 1, Fig. 3, and Table 2.

Hyperparameter	Value
Sentence length cutoff	100
Word embedding size	300
POS tag embedding size	50
Number of attention heads	10
Number of Transformer layers	6
Feedforward layer hidden size	512
Attention key vector size	64
Attention value vector size	64
Dropout	0.2
Dependency arc vector size	512
Dependency label vector size	128
Batch size	80
Learning rate	10^{-4}
Early stopping patience	50

Table 4: Hyperparameter values of the source parser.

Hyperparameter	Value	
	Nearby	Distant
ST		
Sentence length cutoff	60	60
Learning rate	5.6×10^{-4}	3.7×10^{-4}
L2 coefficient (λ)	3×10^{-4}	2.8×10^{-4}
PPT		
Learning rate	3.8×10^{-5}	2×10^{-5}
L2 coefficient (λ)	0.01	0.39
PPTX/PPTX-LOO		
Learning rate	2.1×10^{-5}	5.9×10^{-5}
L2 coefficient (λ)	0.079	1.2×10^{-4}
PPTX-REPR		
Learning rate	1.7×10^{-5}	9.7×10^{-5}
L2 coefficient (λ)	4×10^{-4}	0.084
PPTX-PRAG		
Learning rate	4.4×10^{-5}	8.5×10^{-5}
L2 coefficient (λ)	2.7×10^{-4}	2.8×10^{-5}
Projective PPT		
Sentence length cutoff	20	20
Learning rate	10^{-4}	10^{-4}
L2 coefficient (λ)	7.9×10^{-4}	7.9×10^{-4}
Projective PPTX-PRAG		
Sentence length cutoff	20	20
Learning rate	9.4×10^{-5}	9.4×10^{-5}
L2 coefficient (λ)	2.4×10^{-4}	2.4×10^{-4}

Table 5: Hyperparameter values of ST, PPT, PPTX, PPTX-REPR, PPTX-PRAG, projective PPT, and projective PPTX-PRAG. Sentence length cutoff for PPT, PPTX, PPTX-REPR, and PPTX-PRAG is 30, as explained in Section 3.1.

Modelling Context Emotions using Multi-task Learning for Emotion Controlled Dialog Generation

Deeksha Varshney, Asif Ekbal, Pushpak Bhattacharyya
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India
`{1821cs13, asif, pb}@iitp.ac.in`

Abstract

A recent topic of research in natural language generation has been the development of automatic response generation modules that can automatically respond to a user's utterance in an empathetic manner. Previous research has tackled this task using neural generative methods by augmenting emotion classes with the input sequences. However, the outputs by these models may be inconsistent. We employ multi-task learning to predict the emotion label and to generate a viable response for a given utterance using a common encoder with multiple decoders. Our proposed encoder-decoder model consists of a self-attention based encoder and a decoder with dot product attention mechanism to generate response with a specified emotion. We use the focal loss to handle imbalanced data distribution, and utilize the consistency loss to allow coherent decoding by the decoders. Human evaluation reveals that our model produces more emotionally pertinent responses. In addition, our model outperforms multiple strong baselines on automatic evaluation measures such as F1 and BLEU scores, thus resulting in more fluent and adequate responses.

1 Introduction

One of the key skills for dialogue agents in a dialog system is to acknowledge the feelings of the user and respond accordingly. It is quite instinctive for humans to identify and understand other people's emotions but is quite hard for Artificial Intelligence (AI) systems due to the lack of representative publicly available data sets for training and evaluating an intelligent and robust dialog management system. Table 1 shows an example of emotion labelled conversation from the dataset. The example shows how two different emotionally inclined responses can lead a conversation in two different directions. An engaging conversation usually involves empathetic responses by conversing partners which can have varied emotion labels.

Agent 1	**Do you like wearing hats? It has so many functions.**	Curious
Agent 2	I don't like them on myself but I know a lot of people that can pull them off.	Neutral
Agent 1	Yes me as well. In the military hats denote a nationality, branch of service, rank or regiment.	Curious
Agent 2	Yes. I love hats! I have a wide variety of hats and wear them for different reasons.	Happy
Agent 1	Yes. . . Even I like it too !! Specially I am on vacation, roaming around I do carry 2–3 hats. And I wear it according to my dressing style.	Happy

Table 1: A snippet of two different emotionally inclined conversations with a common query.

It is important to capture user's affective information by any dialog agent to build an intelligent and socially engaging open-domain chatbot. For learning new tasks, we often apply the knowledge we have acquired by learning similar tasks. For instance, in Table 2 the context history has several utterances with Happy, Fearful, Disgusted and Curious to dive deeper emotion and the target responses are labelled with Fearful and Happy emotion. Context emotion can play an important role in transferring the target style while predicting the responses. The words *terrified, scary, afraid* and *like* can help in generating responses with the given target emotion label respectively. An auxiliary task of emotion classification can help in improving the main task of text generation.

In prior research, neural network based models handled the emotion controlled generation by either appending the target emotion label (Zhou et al., 2018; Zhou and Wang, 2017; Wang and Wan, 2018; Hu et al., 2017; Huang et al., 2018; Logeswaran et al., 2018; Song et al., 2019) or by using emotion embeddings (Asghar et al., 2018) in addition to

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2919–2931
April 19 - 23, 2021.

Agent 1	Are you **afraid** of snakes?	Curious
Agent 2	Hi, I am a little! but I was surprised there are none in New Zealand!	Happy
Agent 1	Sounds like a perfect place for me lol, I'm **terrified** of them	Fearful
Agent 2	Wow! I can understand , I am more **terrified** of crocodiles but it seems they are closer to birds than to snakes!	Fearful
Agent 1	Some snakes can even fly to catch their prey so thats **scary**	Curious
Agent 2	Wow, I would **like** to see that! And did you know its head is designed to swallow prays larger than them	Happy
Agent 1	Yeah I did know that, thats actually a bit disgusting, watching them eat prey	Disgusted
Agent 2	It looks like monkeys are terrified of snakes too!	Happy
Agent 1	They are? monkey are smart, they should stay as far as they can of snakes, dangerous animals	Fearful
Agent 2	Maybe you are terrified of snakes! But do you like dancing?	Happy

Table 2: Example conversations from the topical chat dataset showing different context emotion labels.

the input sentence representation. Although label information is effective, still it seems to be underutilized for effective response generation. Wang and Wan (2018) showed sentiment transfer using discriminator networks.

We hypothesize that emotional construct in a conversation can be formed by focusing on specific words in a dialog. To acknowledge the presence of annotated emotion labels in a multi-turn conversation, we perform emotion analysis of user utterances as an auxiliary task for open-domain dialogue generation. Our objective is to generate responses according to the target emotion style. Specifically, if we want to choose words that can provide information about the emotion of a sentence, we exploit an emotion classification model to govern the selection strategy. We train a self-attention (Vaswani et al., 2017) based encoder to compute the context features in a dialog. Words with higher attention weights are selected to be in the set of selections while decoding the response.

In this work, we propose to apply multi-task learning to leverage emotion information for open-domain response generation. Multi-task learning allows the encoder to learn common and prominent features in the input sequence. Our emotion-incorporated weights achieve a good balance between language fluency and emotion quality in model responses. We utilize focal loss (Lin et al., 2017) for emotion classification to address the imbalanced structure of the emotion distribution in the dataset. Furthermore, to attain better attention scores, we compute consistency loss in order to preserve the attention performance of individual tasks. Our empirical study does not show performance degradation in language fluency while classifying emotion-rich sequences.

We evaluate our proposed model on the Topical Chat dataset (Gopalakrishnan et al., 2019). We design human evaluation to score the following three metrics, *viz.* fluency, adequacy and emotional accuracy of the generated response. The human evaluation results indicate that our model improves not only the fluency and adequacy scores but also the emotional accuracy scores. In addition, we conduct automatic evaluation on the topical chat dataset. The automatic evaluation results show that our method improves significantly on the F1 and BLEU metrics.

The key contributions and/or attributes of our current work are summarized as follows:

1. We propose an effective deep multi-task framework that performs emotion classification and response generation.
2. To handle the imbalanced data distribution, we use Focal Loss (Lin et al., 2017) instead of regular cross entropy loss for emotion classification of utterances.
3. To maintain uniformity between the attention weights of different tasks, we utilise consistency loss (Nishino et al., 2019) in addition to the original task-specific losses.

2 Related Work

Early representative works were mostly based on the manually hand-crafted rules (Skowron, 2010; Polzin and Waibel, 2000), for generating responses with a specific emotion. Although rule-based approaches show high accuracy they often fail to handle complex emotions, especially for large corpora. In (Prendinger and Ishizuka, 2005), computational experiments established that empathetic agents ensure good communication. Ochs et al. (2008) designed an empathetic virtual agent that can express emotions based on cognitive appraisal theories which require an extensive hand-crafted rule base.

In recent years, there is an emerging research trend in an end-to-end neural network based generative conversational systems (Vinyals and Le, 2015;

Shang et al., 2015). To improve the content quality of neural conversational models, many techniques have been proposed, such as improving response diversity using Conditional Variational Autoencoders (CVAE) (Zhao et al., 2017) and encoding commonsense knowledge using external facts corpus (Ghazvininejad et al., 2018).

By expressing emotions, people show their mutual respect, empathy and understanding to each other, and thus improve the relationship between them. Emotional chatting machine (ECM) (Zhou et al., 2018) extended the basic encoder-decoder architecture using three mechanisms, *viz.* emotion category embedding, internal emotion memory, and external memory in order to generate sequence with a particular emotion label. Affect transfer in text using Recurrent Neural Networks (RNNs) (Ghosh et al., 2017) and text generation using emojis as the target labels (Zhou and Wang, 2017) was proposed for controlled generation of text. The research reported in (Niu and Bansal, 2018; Golchha et al., 2019) introduced state-of-the-art techniques for stylistic transfer of user behaviour, such as courteousness (e.g. polite, rude or neutral). Li et al. (2019) proposed an empathetic dialogue system (EmpGAN) based on adversarial learning comprising of a multi-resolution empathetic generator along with two interactive discriminators.

Song et al. (2019) presented an attention framework based on emotion-lexicons. Colombo et al. (2019) generated affect driven dialogues using emotion embeddings and affective sampling methods. Various techniques that can capture user's emotional state empathetic response generation were developed in (Asghar et al., 2018; Lubis et al., 2018) . An affective attention based model coupled with weighted cross-entropy loss was proposed by Zhong et al. (2019) for affective dialogue generation. Lin et al. (2020) built an empathetic chatbot which fine-tunes a Generative Pre-trained Transformer (GPT) with multiple objectives: response language modeling, response prediction, and dialogue emotion detection.

Multi-task learning, with deep neural networks which learn from different related-tasks has achieved remarkable success in improving the performance of many natural language processing (NLP) tasks (Luong et al., 2015a; Hashimoto et al., 2016; Liu et al., 2019). A multi-task learning framework usually consists of an encoder which is shared across multiple tasks to learn a common set of shared features. Moreover, the encoder learns to focus more on important and desirable features, and ignores redundant and noisy features (Ruder, 2017). Rashkin et al. (2018) proposed a new dataset with $\sim$ 25k conversations empathetic dialogue generation. The conversations in the dataset are prepared for a given emotion label. As opposed to this, our model handles dataset which has different emotion labels for every utterance in a dialog. As per our knowledge there is no existing work that has proposed the multi-task learning architecture for heterogeneous emotions in a conversation.

In our current work, we propose a multi-task framework with a shared multi-head self-attention based hierarchical encoder for response generation and emotion classification. We also utilize focal loss for emotion classification. Additionally, we incorporate a consistency based loss to enable persistent output generation for our multi-task architecture. The experiments are performed on the knowledge and emotion grounded Topical Chat dataset (Gopalakrishnan et al., 2019) containing a significant amount of human-human conversations in open-domain setting. Our approach tends to produce adequate responses.

3 Methodology

3.1 Problem Statement

In this work, we aim to produce emotion controlled responses for multi-turn conversations using relevant context knowledge and emotion labels. Let $U = u^{(1)}, ..., u^{(k)}, ..., u^{(K)}$ denote the set of K utterances of our multi-turn conversation. We represent I words of the k-th utterance as $u^{(k)} = w_1^{(k)}, ..., w_i^{(k)}, ..., w_I^{(k)}$. Each utterance $u^{(k)}$ is tagged with an emotion label $e^{(k)}$ i.e $E = e^{(1)}, ..., e^{(k)}, ..., e^{(K)}$. Hence, our task is to generate a response $y = y_1, y_2, ..., y_m$ with m words given the set of previous k context utterances and emotion labels.

3.2 Encoder-Decoder Model

3.2.1 Encoder

The encoder is used to transform the input utterance into a hidden representation $q^{(k)}$. The embedding, e, of the current word, $e(w_i^{(k)})$ and the positional embedding *PE(i)* is fed as input to the encoder. The combined embedding representation is subsequently passed into the Gated Recurrent Unit (GRU) model (Cho et al., 2014) which encodes the input utterance and yields relevant features. We

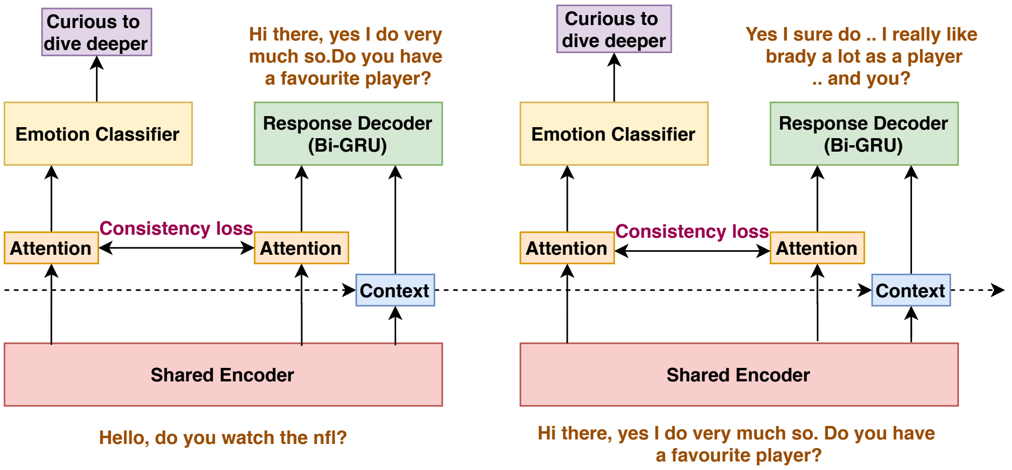


Figure 1: Proposed model architecture

apply self-attention (Vaswani et al., 2017) on the input features. Here, ‘n’ is the number of layers.

$$Ip_u^{(k)} = [w_1^{(k)}, ..., w_I^{(k)}] \tag{1}$$

$$w_i^{(k)} = e(w_i^{(k)}) + PE(i) \tag{2}$$

$$h_{e,t}^{(k)} = GRU_e(w_t^{(k)}, h_{e,t-1}^{(k)}) \tag{3}$$

$$(D^{(k)})^n = \text{MultiHead}(h^{(k)}, h^{(k)}, h^{(k)}) \tag{4}$$

$$(E^{(k)})^n = \text{FFN}((D^{(k)})^n) \tag{5}$$

$$q^{(k)} = (E^{(k)})^n \tag{6}$$

3.2.2 Context-level Encoder:

We use a GRU network to address the previous context of utterances in a multi-turn conversation. The initial state of the decoder GRU is initialised with the final hidden state of the context GRU.

$$h_c^{(k)} = GRU_c(q^{(k)}, h_c^{(k-1)}) \tag{7}$$

3.2.3 Decoder:

Intuitively, this layer takes what we have decoded so far, $h_{d,t-1}^{(k)}$, and all of what we have encoded, $q^{(k)}$, to produce a vector, $a_t^{(k)}$, that represents attention weights which signifies most important words in the source sentence in order to correctly decode, $\hat{y}_{t+1}$. We then calculate the energy, $e_{e,ij}^{(k)}$, between them by concatenating them together and passing them through a linear layer (attn) and a $\tanh$ activation function. The desired conditioning on previous utterances (context history) is obtained by initializing the hidden state of the GRU decoder with the final hidden state from the context GRU, $h_c^{(k)}$ i.e $h_{d,0}^{(k)} = h_c^{(k)}$.

$$h_{d,t}^{(k)} = GRU_d(e(y_t^{(k)}), w_t^{(k)}, h_{d,t-1}^{(k)}) \tag{8}$$

$$e_{e,ij}^{(k)} = v^T \tanh(\text{attn}(h_{d,t-1}^{(k)}), q_j^{(k)})) \tag{9}$$

$$a_t^{(k)} = \text{softmax}(e_{e,ij}^{(k)}) \tag{10}$$

$$g_t^{(k)} = a_t^{(k)} q_j^{(k)} \tag{11}$$

$$P(\hat{y}_{t+1}/y_{<t}) = softmax(e(y_t^{(k)}), g_t^{(k)}, h_{d,t}^{(k)}) \tag{12}$$

3.3 Multitasking Dialog Generation and Emotion Recognition

We perform multi-tasking using a shared encoder layer for encoding input sequences and two decoder layers for utterance prediction and classification. Figure 1 gives an overview of our proposed model.

Shared encoder: We use the encoder from Section 3.2.1 which converts the input sequence into hidden vectors ($q^{(k)}$) which is used across multiple tasks.

Classifier: The classifier transforms the shared representation from the encoder into the emotion class probability $p_c^{(k)}$.

$$p_c^{(k)} = \text{softmax}(Wq^{(k)} + b) \tag{13}$$

Decoder: We employ a GRU based decoder which takes the hidden representation from the shared encoder and generates a response $y = y_1, y_2, ..., y_m$ comprising of m words.

3.4 Focal Loss

Focal Loss (Lin et al., 2017) is employed to address imbalance between the emotion classes during training. We use focal loss as a replacement of cross entropy loss for emotion recognition. It is defined in Eq 14, where γ is a focusing parameter.

$$L_1 = -(1 - p_c^{(k)})^{\gamma} log(p_c^{(k)}) \tag{14}$$

3.5 Consistency Loss

We use the "consistency loss" (Nishino et al., 2019) to reduce the difference between the attention weights from different tasks. Attention agreement favours emotional words while decoding the responses. The consistency loss between two different tasks is defined as follows:

$$L_{cl} = \sum_{i=1}^{I} |\max_j e_{p,ij}^{(k)} - \max_j e_{q,ij}^{(k)}|+ \tag{15}$$

where $e_{p,ij}^{(k)}$ is the attention weight for every k-th utterance for the p-th task. To compare the two attention weights, a ramp function $|x|+$ is used.

3.6 Training: Dialog generation

We denote the negative log-likelihood loss for dialog generation using L_2.

$$L_2 = -\sum_{t=1}^{m} logP(\hat{y}_{t+1}/y_{<t}) \tag{16}$$

The overall loss function for our proposed model is calculated as the total sum of losses from the two tasks and the consistency loss:

$$L_{all} = L_1 + L_2 + L_{cl} \tag{17}$$

where L_1 and L_2 signify the loss of the emotion classification and dialog generation task. L_{cl} indicates the consistency loss.

4 Datasets and Experiments

4.1 Dataset

We perform our experiments on the knowledge and emotion grounded Topical Chat dataset (Gopalakrishnan et al., 2019) with ~11K dialogues. It is a multi-turn conversational dataset in which every utterance is annotated with an emotion label. There are a total of eight emotions (angry, disgusted, fearful, sad, happy, surprised, curious to dive deeper, and neutral) in the dataset. The data is split into 5 distinct groups: Train, Valid Frequent, Valid Rare, Test Frequent, and Test Rare. The frequent set contains conversations on entities frequently seen in the training set. The rare set contains conversations on entities infrequently seen in the training set. Table 3 provides the details of the dataset.

	#Conversation	#Utterances
Train	8628	188378
Valid Frequent	539	11681
Valid Rare	539	11692
Test Frequent	539	11760
Test Rare	539	11770

Table 3: Dataset details

Emotion Classes	Original Count
Curious to dive deeper	101162
Surprised	38254
Disgusted	1848
Sad	3070
Neutral	51796
Happy	36845
Angry	1133
Fearful	1174

Table 4: Distribution of emotion classes in topical chat dataset

4.2 Baselines

In order to prove the usefulness of our model, we compare it with the following baselines:

1. **HRED:** This baseline is defined based on the hierarchical encoder-decoder model by Serban et al. (2015, 2016). In this, the encoder RNN encodes the words of the utterances, and the context RNN encodes the dialog history.
2. **HRED-A:** We apply word-level attention (Luong et al., 2015b) to the encoder of the HRED model to capture important words of the input sequence.
3. **HRED-SA:** Another extension to the generative hierarchical Seq2Seq model with self-attention mechanism on the encoder which takes the dialog conversations as input.
4. **EmoHRED-A-FL-CL:** We extend the HRED-A model to EmoHRED-A-FL-CL, a deep multi-task learning framework that jointly performs the task of both response generation and emotion analysis. We add focal loss and consistency loss to the existing task specific losses.

To prove the effectiveness of our consistency loss in EmoHRED-SA-FL-CL, we conduct ablation study by removing the consistency loss from the EmoHRED-SA-FL-CL model. We name the model as EmoHRED-SA-FL. We also show the strength of the focal loss by eliminating FL from EmoHRED-SA-FL model. The resulting model is named as EmoHRED-SA.

Models	PPL *(Freq/Rare)*	BLEU% *(Freq/Rare)*	F1% *(Freq/Rare)*	Div.(n=1) *(Freq/Rare)*	Div.(n=2) *(Freq/Rare)*	Fluency *(Freq/Rare)*	Adequacy *(Freq/Rare)*	EA *(Freq/Rare)*
HRED	45.61 / 70.30	2.4 / 1.9	0.14 / 0.10	0.88 / 0.87	0.89 / 0.88	1.65 / 1.60	0.85 / 0.70	0.50 / 0.45
HRED-A	41.42 / 71.31	2.3 / 1.8	0.15 / 0.11	**0.91 / 0.90**	**0.90 / 0.90**	1.70 / 1.65	0.90 / 0.84	0.52 / 0.54
HRED-SA	36.63 / 54.87	2.1 / 1.8	0.21 / 0.15	0.83 / 0.82	0.84 / 0.84	1.70 / 1.65	0.98 / 0.88	0.60 / 0.55
EmoHRED-A-FL-CL	36.08 / 51.06	2.1 / 1.7	0.23 / 0.12	0.87 / 0.87	0.87 / 0.88	1.85 / 1.80	1.45 / 1.35	0.74 / 0.64
EmoHRED-SA-FL-CL	**35.45 / 50.45**	**2.6 / 2.1**	0.23 / **0.19**	0.88 / 0.87	0.89 / 0.88	**1.95 / 1.90**	**1.50 / 1.45**	**0.80** / 0.60
EmoHRED-SA-FL	36.34 / 54.82	2.3 / 1.9	**0.25** / 0.13	0.86 / 0.82	0.86 / 0.84	1.80 / 1.80	1.01 / 0.95	0.64 / **0.65**
EmoHRED-SA	36.04 / 52.98	2.3 / 1.8	0.24 / 0.13	0.88 / 0.83	0.83 / 0.84	1.83 / 1.81	0.93 / 0.81	0.53 / 0.51

Table 5: Evaluation results using automatic and human evaluation metrics for baseline, ablation, and our proposed model. Bold face indicates leading results for each metric.

4.3 Experimental Setup

For the HRED model, we use a single layer bidirectional GRU (Cho et al., 2014). We extend the HRED model to HRED-A using the global attention mechanism (Luong et al., 2015b) at the encoder. For our proposed self-attention-based model, the number of encoder and decoder layers is set to 2 and the number of attention heads is 8 with the filter size equal to 2048. Word embedding dimension is chosen as 300, hidden dimension is set to 300. For the generator, we use the ADAM optimizer (Kingma and Ba, 2014) whose learning rate is fixed to 0.0001. While decoding the responses we use beam search with beam size set to 4.

4.4 Evaluation Metrics

Automatic Evaluation: We utilise the most well-known metrics for evaluating a sequence such as BLEU (Papineni et al., 2002), F1, perplexity (PPL) (Vinyals and Le, 2015) and n-gram diversity (Div.) (Gopalakrishnan et al., 2019).

1. **Perplexity:** We define perplexity in Equation 18. It is a measurement of how well a model can predict human responses. We report perplexity values on our frequent and rare test. N is the total number of samples in the test set and N_w is the total number of tokens in the entire test set.

$$PPL = \exp\{-\frac{1}{N}\sum_{i=1}^{N_w} log(P(y|U))\} \quad (18)$$

2. **BLEU:** To evaluate the predicted responses we compute BLEU score, a word-based metric which performs n-gram matching with the ground truth responses.
3. **F1:** We compute *unigram* F1-score[1] between the model prediction and the ground truth responses.
4. **N-gram diversity:** We evaluate the informativeness and diversity of sentences using N-gram diversity. It is defined in Eq 19. M is the total number of samples in the test set. The results are shown under the columns - Div. (n=1) and Div. (n=2) in Table 5 on the frequent and rare test set.

$$Div = \frac{1}{M}\left[\frac{\text{\# unique n-grams}}{\text{\# words in predicted response}}\right] \quad (19)$$

Human Evaluation: To measure the quality of the generated text from a human perspective, we randomly sample 100 conversations from each model and with the help of two experts with post-graduate exposure we evaluate the predicted responses using the following metrics:

(i) Fluency: It is used to measure the grammatical correctness. (ii) Adequacy: It is used to measure contextual relevancy of the predicted response. (iii) Emotional Accuracy (EA): It checks how accurately one can infer the target emotion in the predicted response.

We assign a scores in {0,1,2} (representing "wrong", "acceptable" and "perfect") for indicating the level of fluency and adequacy of responses. We measure the emotional accuracy on a scale of 0-1 with '0' indicating the incorrect emotion and '1' the correct emotion. We compute the Fleiss' kappa (Fleiss, 1971) score, to measure the inter-annotator agreement. We obtain a kappa score of 0.90, 0.75, 0.76 for fluency, adequacy, and emotional content, respectively, denoting "good agreement".

5 Results and Analysis

We present the results for all our experiments in this section. Detailed results using both the automatic and human evaluation methods are shown in Table 5.

---

[1] https://github.com/facebookresearch/ParlAI/blob/master/parlai/core/metrics.py

5.1 Automatic evaluation results

In Table 5, we observe that the proposed model has high uni-gram and bi-gram diversities, demonstrating that the models learn to decode fluent and informative responses with great diversity. We observe relatively fewer repeated segments in the responses generated by our proposed model owing to a good Div.(n=1) and Div.(n=2) score. We observe significant improvement in BLEU and F1-scores when compared with the baseline models which support our multi-task learning architecture. Our proposed model seems to utilize the multi-task learning phenomenon and effectively utilize emotion labels associated with each utterances.

We also perform an ablation study for better understanding the contributions of the attributes of our model. As shown in Table 5, after we remove the consistency loss, both the emotion accuracy and perplexity performance become obviously worse, indicating that to generate persistent outputs, consistency between attention weights is critical for emotion understanding and model generation quality. We also test the importance of focal loss using the EmoHRED-SA model. As shown in Table 5, after we eliminate the focal loss, there is significant drop in EA and F1 which justifies our use of focal loss. We perform statistical significance test between our proposed and the baselines models using t-test at 5% (0.05) significance level, and showed that the improvement in our model is statistically significant.

5.2 Human evaluation results

Table 5 illustrates that our proposed model outperforms the other baseline models in terms of fluency, adequacy and emotion quality. Owing to a good fluency score of our proposed model, we observed fewer copying of sentences from the input utterance in the predicted response. The increment in the adequacy scores w.r.t baseline models verifies that the response generated by the proposed model comes out as more relevant. The emotional content score determines that the generated responses are more in line with the emotional sensitivity of the sentences.

In Table 6, we present few examples of the responses generated by one of the baseline model (HRED) and our proposed model given the desired emotion. As shown in the table, the responses predicted by the *EmoHRED-SA-FL-CL* model has mostly predicted adequate and emotionally relevant responses as compared to the baseline HRED model. For the fourth utterance, even though the HRED model gives an emotionally relevant reply but it seems highly inadequate with respect to the context where as the *EmoHRED-SA-FL-CL* model responds with an emotionally as well as contextually relevant reply. Detailed examples with outputs from all of our baseline and proposed model with the required emotion label can be found in the appendix in Table 7.

5.3 Error Analysis:

In this section, we report the most commonly occurring errors that our proposed and baseline models encounter.

1. Common phrases: Some common phrases are repeated in the generated response. For instance *'i don't think i've ever heard about it though'*, *'i don't know much about it so i don't know much about it either.'* and *'i 'm not sure either. i've never been there'*. Due to data scarcity and less diversity in the data, the models may only have learned to predict the most frequent utterances. Since the dialogues are inherently ambiguous, predicting them accurately would require more data.

2. Repetition: The proposed model (EmoHRED-SA-FL-CL), in a few cases, go on repeating the information present in the predicted response. Predicted Response: *that's terrible. i'll have to check that out. i'll have to check it out!*. This lowers the count of unique uni-gram words in the generated response i.e the F1-score.

3. Emotional inconsistencies: In some cases, the proposed model (EmoHRED-SA-FL-CL) is unable to produce responses of particular emotion labels due to less occurrence of instances from those classes (angry, sad, fearful and disgusted). The less frequent emotion classes like anger, sad, fearful and disgusted get confused with the recurring classes like curious to dive deeper and surprised. Also, instances from 'Happy' and 'Surprised' emotion classes get mixed up with each other. For example, in Table 6, the target response for Utterance 5 should have the emotion 'Happy' but it gets confused with the emotion 'Surprised' and generates an irrelevant response. Table 4 shows the distribution of emotion classes present in the dataset. More detailed examples can be found in Table 8 and Table 9 of the Appendix.

Utterance 1	**do you enjoy sports?**	**Neutral**
EmoHRED-SA-FL-CL	i do! it's very interesting. i used to play video games when i was a kid i don't have any time.	Happy
HRED	i do. do you ?	Curious
Utterance 2	**i enjoy the getting together with friends part of it but i could take or leave the actual sports events except when i am watching my children play. how about you what's your favorite sport?**	**Happy**
EmoHRED-SA-FL-CL	i do too.	Neutral
HRED	i enjoy watching football games as well.	Happy
Utterance 3	**i would say football**	**Neutral**
EmoHRED-SA-FL-CL	nice! speaking of football, did you know women were not allowed to play at the same time ?	Curious
HRED	i don't really like football, but i don't think i've ever heard of it, what about you?	Curious
Utterance 4	**that is the favorite sport in my household as well! tom brady is quite the dedicated player both on and off the field, don't you think?**	**Curious**
EmoHRED-SA-FL-CL	i don't know much about him.	Neutral
HRED	i do not have a favorite team but i don't really follow it.	Neutral
Utterance 5	**yes, he is arguably the greatest steal in nfl history**	**Neutral**
EmoHRED-SA-FL-CL	i can't believe he has never eaten a strawberry ha ha. have you ever heard of him ?	Surprised
HRED	i think it's cool that the nfl has no written rule against female players. i don't know how that's possible.	Surprised

Table 6: Generated examples are from a continuous conversation from the frequent test set. EmoHRED-SA-FL-CL and HRED predict responses using the previous set of utterances and emotion labels.

5.4 Comparisons to the state-of-the-arts

The original paper (Gopalakrishnan et al., 2019) proposed a transformer-based model, and observed a F1-score on frequent and rare test set of 0.16 / 0.16 and a Div.(n=1) and Div.(n=2) of 0.85 / 0.84 and 0.86 / 0.86, respectively. Following the diversity scores, we notice a significant improvement with Div.(n=1) and Div.(n=2) of 0.88 / 0.87 and 0.89 / 0.88, respectively, for our proposed model. Similarly, using CAiRE (Lin et al., 2020) we obtained a F1-score on the frequent/rare test set of 0.13 / 0.13 and a Div.(n=1) and Div.(n=2) of 0.87 / 0.83 and 0.86 / 0.85, respectively. However, it is to be noted that like us, (Gopalakrishnan et al., 2019) and (Lin et al., 2020) did not focus on taking into consideration the context utterance and instead simply concatenated the context utterances and passed them as a single sequence into the transformer model. We also observe a significant improvement in the F1-score for our proposed model. We achieve a score of 0.23 / 0.19 for our task of emotion-controlled dialog generation. We adopt ECM (Zhou et al., 2018) for comparison, a Seq2Seq model that first proposed to generate emotional response using emotion category embeddings, internal and external memory mechanisms. We concatenate the dialog history into a long sequence and feed as input to the model. Evaluation shows the F1-score of 0.14 / 0.13 and BLEU score of 1.9 / 1.6 for the frequent / rare test set. Our model clearly outperforms the baselines with a huge margin.

6 Conclusion and Future Work

In this paper, we have proposed a new deep learning framework for modeling emotion-grounded conversations using emotion labels as the guiding attributes. Building an emotion-aware conversational agent is crucial in enhancing the user interactions with long, engaging conversations.

Extensive experiments show that the predicted responses expressed high levels of emotional accuracy and content adequacy. We have also provided details of different kinds of errors found in section 5.3. In general, we show how a related task of emotion recognition along with appropriate loss functions can ensure emotional relevancy of the generated response and improves user engagement.

In the future, we intend to use pre-trained language models for the task of dialog generation using emotion labels. We also aim to extend our model to handle knowledge-grounded conversations.

Acknowledgement

Authors duly acknowledge the support from the Project titled Sevak-An Intelligent Indian Language Chatbot, Sponsored by SERB, Govt. of India (IMP/2018/002072). Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya Ph.D. scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation

(formerly Media Lab Asia).

References

Nabiha Asghar, Pascal Poupart, Jesse Hoey, Xin Jiang, and Lili Mou. 2018. Affective neural response generation. In *European Conference on Information Retrieval*, pages 154–166. Springer.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Pierre Colombo, Wojciech Witon, Ashutosh Modi, James Kennedy, and Mubbasir Kapadia. 2019. Affect-driven dialog generation. *NAACL*.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Sayan Ghosh, Mathieu Chollet, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. 2017. Affect-lm: A neural language model for customizable affective text generation. *arXiv preprint arXiv:1704.06851*.

Hitesh Golchha, Mauajama Firdaus, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Courteously yours: Inducing courteous behavior in customer care responses using reinforced pointer generator network. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 851–860.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Chenyang Huang, Osmar R Zaiane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic dialogue generation with expressed emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 49–54.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Qintong Li, Hongshen Chen, Zhaochun Ren, Zhumin Chen, Zhaopeng Tu, and Jun Ma. 2019. Empgan: Multi-resolution interactive empathetic dialogue generation. *arXiv preprint arXiv:1911.08698*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Zhaojiang Lin, Peng Xu, Genta Indra Winata, Farhad Bin Siddique, Zihan Liu, Jamin Shin, and Pascale Fung. 2020. Caire: An end-to-end empathetic chatbot. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13622–13623.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*, pages 5103–5113.

Nurul Lubis, Sakriani Sakti, Koichiro Yoshino, and Satoshi Nakamura. 2018. Eliciting positive emotion through affect-sensitive dialogue response generation: A neural network approach. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015a. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015b. Effective approaches to attention-based neural machine translation. *EMNLP*.

Toru Nishino, Shotaro Misawa, Ryuji Kano, Tomoki Taniguchi, Yasuhide Miura, and Tomoko Ohkuma. 2019. Keeping consistency of sentence generation and document classification with multi-task learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3186–3196.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

Magalie Ochs, Catherine Pelachaud, and David Sadek. 2008. An empathic virtual dialog agent to improve human-machine interaction. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 89–96.

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

Thomas S Polzin and Alexander Waibel. 2000. Emotion-sensitive human-computer interfaces. In *ISCA tutorial and research workshop (ITRW) on speech and emotion*.

Helmut Prendinger and Mitsuru Ishizuka. 2005. The empathic companion: A character-based interface that addresses user's affective states. *Applied Artificial Intelligence*, 19(3-4):267–285.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. *arXiv preprint arXiv:1811.00207*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Hierarchical neural network generative models for movie dialogues. *arXiv preprint arXiv:1507.04808*, 7(8).

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*.

Marcin Skowron. 2010. Affect listeners: Acquisition of affective states by means of conversational systems. In *Development of Multimodal Interfaces: Active Listening and Synchrony*, pages 169–181. Springer.

Zhenqiao Song, Xiaoqing Zheng, Lu Liu, Mu Xu, and Xuan-Jing Huang. 2019. Generating responses with a specific emotion in dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3695.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Ke Wang and Xiaojun Wan. 2018. Sentigan: Generating sentimental texts via mixture adversarial networks. In *IJCAI*, pages 4446–4452.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *arXiv preprint arXiv:1703.10960*.

Peixiang Zhong, Di Wang, and Chunyan Miao. 2019. An affect-rich neural conversational model with biased attention and weighted cross-entropy loss. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7492–7500.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Xianda Zhou and William Yang Wang. 2017. Mojitalk: Generating emotional responses at scale. *arXiv preprint arXiv:1711.04090*.

A Samples Generated by the Proposed Model

A.1 Predicted responses when we have different emotion labels for every utterance.

A.1.1 When the emotions occur frequently

We observe that the predicted responses as shown in Table 7 tend to follow the target emotions accurately, however they sometimes may lack adequacy.

A.1.2 When the emotions occur rarely

We observe that the predicted responses as shown in Table 8 fails to generate adequate as well as emotionally relevant responses.

A.2 Predicted responses when we have the same emotion label for every utterance.

We observe that the predicted responses as shown in Table 9 are very close to the ground truth response and are also emotionally accurate.

Utterance 1	do you enjoy sports?	Neutral
EmoHRED-SA-FL-CL	i do! it's very interesting. i used to play video games when i was a kid i don't have any time.	Happy
HRED	i do. do you?	Curious
HRED-A	yes, i do. do you?	Curious
HRED-SA	i do. do you?	Curious
EmoHRED-A-FL-CL	i do enjoy football, but i don't really follow it.	Curious
EmoHRED-SA-FL	i do. do you?	Curious
Utterance 2	i enjoy the getting together with friends part of it but i could take or leave the actual sports events except when i am watching my children play . how about you what's your favorite sport?	Happy
EmoHRED-SA-FL-CL	i do too.	Neutral
HRED	i enjoy watching football games as well.	Happy
HRED-A	i tend to enjoy college football teams.	Happy
HRED-SA	i am a big fan. i don't really know much about football	Neutral
EmoHRED-A-FL-CL	i am not a big football fan but i would love to watch it.	Happy
EmoHRED-SA-FL	i'm a big football fan, do you?	Curious
Utterance 3	i would say football	Neutral
EmoHRED-SA-FL-CL	nice! speaking of football, did you know women were not allowed to play at the same time?	Curious
HRED	i don't really like football, but i don't think i've ever heard of it, what about you?	Curious
HRED-A	football is a good sport. i'm a huge fan of football. i was surprised to learn that the nfl uses tracking chips in the players shoulder pads.	Surprised
HRED-SA	i'm not a fan but i don't really know much about football. i don't think i've ever heard of it.	Neutral
EmoHRED-A-FL-CL	i like the nfl. i don't think i would ever play soccer because i don't think i've ever seen it though.	Neutral
EmoHRED-SA-FL	i don't think i've ever seen it but i don't think i've ever seen it but i don't think i've ever seen it but i don't think i've ever seen it . did you know there is only 11 minutes of live gameplay?	Curious
Utterance 4	that is the favorite sport in my household as well ! tom brady is quite the dedicated player both on and off the field, don't you think?	Curious
EmoHRED-SA-FL-CL	i don't know much about him.	Neutral
HRED	i do not have a favorite team but i don't really follow it	Neutral
HRED-A	he is a good player. i think he is a good player but his sister was always better than him	Neutral
HRED-SA	i don't really know much about him	Neutral
EmoHRED-A-FL-CL	i am not sure if i am a football fan. i would like to be able to retire throwing kicks.	Happy
EmoHRED-SA-FL	well i am not sure who he is. he is one of my favorite shows	Neutral
Utterance 5	yes, he is arguably the greatest steal in nfl history	Neutral
EmoHRED-SA-FL-CL	i can't believe he has never eaten a strawberry. have you ever heard of him ?	Surprised
HRED	i think it's cool that the nfl has no written rule against female players. i don't know how that's possible.	Surprised
HRED-A	that's so cool! i'm going to have to look it up. i can't believe that tom brady has never eaten a strawberry in his entire life!	Surprised
HRED-SA	i didn't know that. i don't think i've ever heard of reggie miller. i don't think i've ever heard of him?	Neutral
EmoHRED-A-FL-CL	i wonder if he has ever eaten a strawberry? i wonder if he was born in kenya?	Curious
EmoHRED-SA-FL	i think he is a great player, i think he is a great player, i think he is a great player	Neutral

Table 7: Generated examples are from a continuous conversation from the frequent test set. Each model predicts responses using the previous set of utterances and emotion labels.

Utterance 1	hi, how are you?	Happy
EmoHRED-SA-FL-CL	hello there, i am well thank you. do you watch youtube?	Curious
HRED	i’m doing well, how about you?	Curious
HRED-A	i am good thank you. do you like basketball?	Curious
HRED-SA	i’m great! how are you?	Curious
EmoHRED-A-FL-CL	i ’m doing well thank you. do you like to travel?	Curious
EmoHRED-SA-FL	i’m good. do you like to travel?	Curious
Utterance 2	hi i m doing great how about you?	Curious
EmoHRED-SA-FL-CL	i am well, thanks. do you listen to country music ?	Curious
HRED	i’m great! do you like dogs ?	Curious
HRED-A	i am well thank you. have a great day!	Neutral
HRED-SA	i am well thank you. do you have a dog?	Curious
EmoHRED-A-FL-CL	i’m good do you have a favorite team?	Curious
EmoHRED-SA-FL	i’m good . do you like dogs ?	Curious
Utterance 3	i’m good, thanks for asking. do you like football ?	Curious
EmoHRED-SA-FL-CL	i do, what about you? do you have a favorite football team? do you have a favorite football team? do you have a favorite nfl team?	Curious
HRED	i do like football, do you?	Curious
HRED-A	i sure do. do you have a favorite team?	Curious
HRED-SA	i am not a huge fan of football. do you?	Curious.
EmoHRED-A-FL-CL	i do, i don’t follow it much. do you?	Curious
EmoHRED-SA-FL	i do. do you?	Curious
Utterance 4	i like watching it sometimes, i guess it depends on who is playing lol how about u?	Curious
EmoHRED-SA-FL-CL	it’s interesting that it has been around since 1989.	Curious
HRED	i don’t watch football. i don’t know much about football.	Neutral
HRED-A	yep, it is a very good game. do you have a favorite nfl team?	Curious
HRED-SA	i do too. did you know there is only 11 minutes of live gameplay?	Curious
EmoHRED-A-FL-CL	i am not a huge football fan but i don’t think i’ve watched it a lot.	Neutral
EmoHRED-SA-FL	i do. i’m not a huge fan. do you?	Curious
Utterance 5	me too, the games can get kind of long to watch. the last super bowl i watched was over three hours long.	Disgusted
EmoHRED-SA-FL-CL	yea it’s a classic game but i didn’t think it started i see, i don’t have a favorite team, i don’t have a favorite team. i don’t have a favorite team.	Neutral
HRED	that’s cool i didn’t know it was that long ago!	Surprised
HRED-A	wow that’s really cool. i’m a steelers fan. i don’t really have the time though.	Neutral
HRED-SA	i agree with you there, i didn’t know there is a variation of soccer where 3 teams face each other at the same time.	Curious
EmoHRED-A-FL-CL	oh i see, i don’t know much about football though, do you watch tv?	Curious
EmoHRED-SA-FL	i see, i don’t have a favorite team, i don’t have a favorite team. i don’t have a favorite team.	Neutral
Utterance 6	that’s so weird, especially since its only 11 min of game play	Curious
EmoHRED-SA-FL-CL	i agree with you. i know that soccer makes more money than football, basketball, baseball, golf, tennis and soccer.	Curious
HRED	yeah, i was not aware of that. i didn’t know there is only 11 minutes of actual live gameplay!	Surprised
HRED-A	yes, i agree. did you know espn won an emmy for the superimposed yellow first down line?	Neutral
HRED-SA	i agree with you on that one. i don’t think i’ve ever heard of it. i don’t think i’ve ever heard of it.	Neutral
EmoHRED-A-FL-CL	i know . i don’t know much about it though .	Neutral
EmoHRED-SA-FL	yes, i agree. i wonder if there is a variation of soccer where 3 teams face each other at the same time?	Curious
Utterance 7	i know, it makes me mad, sometimes to think about that. a lot of wasted time standing around. lol	Angry
EmoHRED-SA-FL-CL	for sure, do you have a favorite album?	Curious
HRED	yeah for sure. do you know who alex smith is?	Curious
HRED-A	that was back in 1916 when georgia tech beat cumberland 222 0!	Neutral
HRED-SA	i agree with you there, do you ever watch the world cup?.	Curious
EmoHRED-A-FL-CL	for sure lol i agree with you there. did you know 86 of nevada is owned by the government?	Curious
EmoHRED-SA-FL	i know right, i don’t know much about football, i don’t know much about it	Neutral

Table 8: Generated examples are from a continuous conversation from the frequent test set. Each model predicts responses using the previous set of utterances and emotion labels.

Utterance 1	do you watch football?	Curious
EmoHRED-A-FL-CL	i sure do. u of iowa painted the visitor locker room pink?	Curious
HRED	i do. do you have a favorite team?	Curious
HRED-A	i do watch it from time to time. what about you?	Curious
HRED-SA	i sure do. do you?	Curious
EmoHRED-A-FL-CL	i do, i am a patriots fan, what about you?	Curious
EmoHRED-SA-FL	i do. do you?	Curious
Utterance 2	i sure do. do you know the u of iowa painted the visitor locker room pink?	Curious
EmoHRED-A-FL-CL	i wonder if that is supposed to play?	Curious
HRED	yes i have heard of it. do you know who invented the sport?	Curious
HRED-A	hmm i have no idea. do you know why?	Curious
HRED-SA	yes i wonder why that is?	Curious
EmoHRED-A-FL-CL	i have no idea but i wonder if it was supposed to do it?	Curious
EmoHRED-SA-FL	yeah that is pretty cool, i wonder why that is	Curious
Utterance 3	i wonder if that is supposed to have some effect on the opposing team	Curious
EmoHRED-A-FL-CL	i wonder too. did you know bowlers made more than football players in the 60s?	Curious
HRED	yeah i wonder too. did you know espn won an emmy?	Curious
HRED-A	good point. did you know women can play too?	Curious
HRED-SA	i think so too. do you know why the managers wear uniforms?	Curious
EmoHRED-A-FL-CL	i am not sure either. do you know much about bowlers?	Curious
EmoHRED-SA-FL	i do not know. do you know who invented the sport?	Curious
Utterance 4	i wonder too . did you know bowlers made more than football players in the 60s?	Curious
EmoHRED-A-FL-CL	yes i did hear about that. do you know who alex smith is?	Curious
HRED	i guess that was back in 1916 when georgia tech beat cumberland 222 0	Neutral
HRED-A	i did not know that it was the highest grossing basketball movie of all time	Surprised
HRED-SA	i did not know that. do you know who invented the game?	Curious
EmoHRED-A-FL-CL	i am not sure but i guess it makes more sense then	Surprised
EmoHRED-SA-FL	wow that is a lot. did you know espn won an emmy?	Curious
Utterance 5	yes i guess football was not as popular back in the day as it is now.	Curious
EmoHRED-A-FL-CL	true. do you know who benjarvus green ellis is?	Curious
HRED	yeah me too. did you know espn won an emmy?	Curious
HRED-A	yeah true. did you know espn won an emmy once?	Curious
HRED-SA	yeah i agree. do you know who invented the game?	Curious
EmoHRED-A-FL-CL	yes i guess it makes sense since it makes more money than football players	Neutral
EmoHRED-SA-FL	true. do you know who invented the sport?	Curious

Table 9: Generated examples are from a continuous conversation from the frequent test set. Each model predicts responses using the previous set of utterances and emotion labels.

Gender and Racial Fairness in Depression Research using Social Media

Carlos Aguirre, Keith Harrigian, Mark Dredze
Johns Hopkins University
`caguirr4@jhu.edu, kharrigian@jhu.edu, mdredze@cs.jhu.edu`

Abstract

Multiple studies have demonstrated that behavior on internet-based social media platforms can be indicative of an individual's mental health status. The widespread availability of such data has spurred interest in mental health research from a computational lens. While previous research has raised concerns about possible biases in models produced from this data, no study has quantified how these biases actually manifest themselves with respect to different demographic groups, such as gender and racial/ethnic groups. Here, we analyze the fairness of depression classifiers trained on Twitter data with respect to gender and racial demographic groups. We find that model performance systematically differs for underrepresented groups and that these discrepancies cannot be fully explained by trivial data representation issues. Our study concludes with recommendations on how to avoid these biases in future research.

1 Introduction

Work from De Choudhury et al. (2013) and Coppersmith et al. (2014), showing that an individual's mental health can be evaluated based on the language they generate on social media platforms, has served as the basis for a substantial amount of computational research over the last decade. Subsequent studies have examined an even wider range of mental health conditions, social media platforms, and types of online behavior at both the individual and population level (Coppersmith et al., 2015b; Lynn et al., 2018; De Choudhury et al., 2016). Vast potential for societal benefits underlies this work, as conservative estimates suggest that 8.1% of American adults suffer from major depressive disorder at any given time and up to 16.2% of individuals will experience at least one major depressive episode during their lifetime (Kessler et al., 2003; Brody et al., 2018; Hasin et al., 2018). Mental health services are transitioning to online mediums at a rapid pace, with the recent COVID-19 pandemic dramatically further accelerating this trend (Zhou et al., 2020; Ohannessian et al., 2020). Thus, analysis of online language may play a key role in mental health treatment in the future.

Nonetheless, care must be taken to understand potential biases inherent in this research before any technologies are deployed in a clinical setting. For instance, previous work has found that Black and Hispanic/Latinx individuals are less likely to be treated for depression than White individuals (Simpson et al., 2007). Possibly a result of this underlying bias, recent studies of the US population have concluded that baseline rates of depression vary depending on demographics (Brody et al., 2018; Hasin et al., 2018) — major depressive disorder was found to be more prevalent in females and White adults. Yet, it remains unclear whether these supposed differences in depression prevalence between gender and racial/ethnic demographic groups are the result of measurement error or other confounders. Various psychological studies have found mental health disorders, including depression, may manifest differently depending on cultural background and thus make uniform diagnosis a difficult proposition (Blanchard et al., 2020; Henrich et al., 2010). These ambiguities were highlighted by recent computational research from Amir et al. (2019), which found predictive rates of depression inferred using classifiers for social media data to not match previous US depression estimates. Indeed, the authors actually find that Black and Hispanic/Latinx individuals are *more* likely to be affected by depression than White individuals.

Additionally, NLP and other data-driven algorithms have been shown to suffer from *content* biases; that is, undesirable group-wise differences with respect to protected groups, such as race/eth-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2932–2949
April 19 - 23, 2021.

nicity or gender (Johannsen et al., 2015; Hovy and Søgaard, 2015; Bolukbasi et al., 2016; Gonen and Goldberg, 2019; Rudinger et al., 2018). Therefore, in consideration of the social impact of NLP research (Hovy and Spruit, 2016), in the area of mental health content analysis it stands to reason that we should also look for *population* biases, as they pertain to protected groups, and the way these might affect NLP algorithms' fairness.

Previous research has utilized user demographics within social media mental health studies to construct control groups (Coppersmith et al., 2014), to enhance classifier performance through additional features (Preoţiuc-Pietro et al., 2015), and to analyze trends amongst specific populations (De Choudhury et al., 2014). In an attempt to preemptively address *population* biases, Amir et al. (2019) proposed a cohort-based sampling approach to collect representative measures of wellness amongst the general population. However, as noted in a recent literature reviews (Chancellor and De Choudhury, 2020; Harrigian et al., 2020), no previous computational mental-health study has accounted for differences in population-level depression rates nor explored performance variations across demographic subgroups at training time. Therefore, little is known about the fairness of these automated systems. Are models trained for mental health fair across demographic groups? Are current datasets demographically representative? If bias exists, what is its source?

In this study, we analyze two common depression-inference datasets and explore the susceptibility of different computational methods to demographic biases. We find that existing datasets are **not demographically representative** and that, without accounting for this, we find degradation in model performance for underrepresented groups. We explore the possible sources of this bias and conclude with recommendations for future research that may address these issues.

2 Mental Health and Social Media

Challenges obtaining mental health annotations for social media data have thus far constrained the size and quality of existing datasets. For instance, manual annotation of mental health status generally requires expert domain knowledge, while the sensitive nature of such annotations limit multi-institutional data sharing (Arseniev-Koehler et al., 2018). Consequently, most datasets rely on labels based on behavioral proxies or self-reported diagnoses, which more easily scale, but introduce problematic *self-disclosure* bias and label noise. Furthermore, as our understanding of mental health is continually evolving, studies have used different and sometimes conflicting guidelines for annotation (Brody et al., 2018; Hasin et al., 2018). With these challenges at the forefront of dataset curation, issues surrounding demographic balance and representation have been largely kicked down the road of the research domain.

Challenges accounting for demographics go beyond the computational research space and are well-illustrated by disparities between two recent surveys of depression prevalence. The Centers for Disease Control and Prevention (CDC) found depression prevalence between race/ethnicity groups did not differ (Brody et al., 2018), while a study using the results of the National Epidemiologic Survey on Alcohol and Related Conditions III (NESARC-III) found depression to be more prevalent in White Americans versus minorities (Hasin et al., 2018).

3 Ethical Considerations

The sensitive nature of mental health research and individual demographics requires us to consider possible benefits of this study alongside its potential harms. Specifically, we must evaluate the cost-benefit trade-off of inferring and/or securing three highly-personal individual attributes: (*depression diagnoses*: Benton et al., 2017a; *gender identity*: Larson, 2017; *race/ethnicity identity*: Wood-Doughty et al., 2020).

The potential immediate benefit of this study is a better understanding of demographic bias in computational mental health research. A potential secondary benefit is the mitigation of extant clinical treatment disparities (Simpson et al., 2007). As mental health treatment increasingly adopts an online delivery mechanism, this research is uniquely situated to inform the development of new AI systems and public policy in the area.

However, we are cognizant of the potential harms from our work. Mental health status and demographic identities are both sensitive personal attributes that could be used to maliciously target individuals on publicly-facing online platforms. Therefore, we follow the guidelines of Benton et al. (2017a) and Ayers et al. (2018) on data use, storage, and distribution. All analysis was conducted on de-identified versions of data, with any identifiable information being used only during intermediate data-processing subroutines that were hidden from researcher interaction and approved by the original dataset distributors. Our study was exempted from review by our Institutional Review Board under 45 CFR § 46.104.

To facilitate any form of statistical analysis, we also need to formalize gender and race/ethnicity. We seek a balance between the limitations of demographic inference systems[1] and alignment to demographic categories conventions used by the mental health literature (Brody et al., 2018; Hasin et al., 2018), versus propagating demographic definitions that exacerbate existing biases towards gender and racial/ethnic minorities. We consider the 'folk conception' of gender as described in Larson (2017) and prominently leveraged in traditional depression research in the United States — we use the sex categories *male* and *female* to denote the corresponding gender categories *masculine* and *feminine*. However, many individuals do not fit in these gender categories, some present a gender online inconsistent to their true identity (Nilizadeh et al., 2016), and they often experience depression and other mental health conditions at a higher rate (McDonald, 2018). For race/ethnicity labels, we consider the mutually-exclusive labeling conventions invoked by Brody et al. (2018) and Wood-Doughty et al. (2020): non-Hispanic White, non-Hispanic Black, non-Hispanic Asian and Hispanic/Latinx, as they are representative of the majority of racial and ethnic identities in the US. Our racial/ethnic categories do not capture multiracial individuals or those with a race/ethnicity outside this group.

We acknowledge these important limitations, but at the same time, there is an urgency to the questions we pose. Computational methods for monitoring mental health have already been deployed by digital surveillance companies (Bark), while analytics dashboards based on these methods are gradually making their way into patients' (Yoo and De Choudhury, 2019) and providers' hands (Yoo et al., 2020). The question is: should we avoid asking these questions about current datasets because we cannot produce clear answers, or should we conduct analyses with acknowledged limitations to learn what we can about research that is already being moved into products? We firmly believe the latter. Our hope is that this paper causes researchers to carefully consider these issues, elevate the need for further work, and produce studies that go beyond our study's limitations with new data and methods. This should not be the last study on this topic; rather, we hope it is the first step which can inform further critical analyses of work in this area.

4 Datasets

We select the task of depression inference for this study, as it is the most widely studied mental health condition in social media research (Harrigian et al., 2020). We consider two Twitter datasets: CLPSYCH (Coppersmith et al., 2015b) and MULTITASK (Benton et al., 2017b).

4.1 CLPSYCH

CLPSYCH was introduced by Coppersmith et al. (2014) and subsequently used in the CLPsych 2015 shared task (Coppersmith et al., 2015b). Tweets were publicly posted between 2008 and 2013. Users who self-disclosed a depression diagnosis were identified using regular expressions (e.g. "I have been diagnosed with `disorder`") and then manually reviewed by a team of clinical and computational researchers to verify authenticity of matched disclosures. The control group was sampled from a random pool Twitter users so that the joint distribution of inferred age and gender attributes closely resembled that of users with self-disclosed diagnoses. The 3000 most recent tweets from each user (as of the original dataset collection date) were retrieved. To reduce ambiguity in model performance that arises due to data insufficiencies, we isolate individuals with at least 100 tweets, leading to a final dataset size of 475 depressed individuals and their matched controls (i.e. 950 total users).

4.2 MULTITASK

Benton et al. (2017b) constructed a Twitter dataset (MULTITASK) combining a subset of CLPSYCH with datasets annotated using the same procedure from Coppersmith et al. (2015a,c). In addition to an expanded number of unique individuals (1400 depression, 1400 control), MULTITASK also boasts a more robust historical timeline of tweets for each user.

---

[1] We infer gender and race/ethnicity labels using a content classifier and explore additional limitations in Section 5

5 Demographic Labels

Only age and gender (Schwartz et al., 2013) attributes are available in the originally distributed form of CLPSYCH and MULTITASK, both of which were inferred using now-outdated models. All identifying metadata was either redacted or obfuscated to preserve the privacy of individuals in these datasets. Accordingly, we are confronted immediately by the challenge of securing accurate demographic information to facilitate a robust analysis of any potential gender and racial/ethnic biases. Fortunately, this problem has been tackled using a multitude of different techniques across multiple studies specific to mental health (Yazdavar et al., 2020; Amir et al., 2017; Preoţiuc-Pietro et al., 2015; Coppersmith et al., 2015a) and social media applications in general (Volkova et al., 2014; Burger et al., 2011; Fink et al., 2012; Rao et al., 2011).

We obtain race labels using a unigram model from Wood-Doughty et al. (2020), who combine multiple crowd-sourced and self-reported datasets to train classifiers for 4 demographic groups in line with the CDC's conventions (Brody et al., 2018): non-Hispanic Asian American (A), non-Hispanic African American (B), non-Hispanic White (W) and Hispanic/Latinx (H/L). Their classifier achieves an accuracy of 82.3% within intrinsic evaluations and shows even more promise as high-confidence thresholds are applied. To validate and further reduce noise in previously-inferred gender attributes, we train a new gender inference model on data from Burger et al. (2011) using the same architecture of Wood-Doughty et al. (2020). Our classifier obtains an accuracy of 83.3% amongst within-distribution data and outputs a distribution of inferred gender attributes that strongly aligns with that of the original datasets.

Although each of these procedures has strong internal validity, we recognize that inference errors incurred during this stage may confound and complicate downstream analysis of demographic bias. To mitigate this potential noise, we also de-anonymize a subset of CLPSYCH with the permission of Coppersmith et al. (2014) and apply name-based demographic classifiers (Wood-Doughty et al., 2018, 2020) to each user's profile to obtain "alternative" age and race attributes.

Between our content and name based classifiers, we are afforded the opportunity to perform downstream analysis of demographic bias based on the attributes derived using the following mechanisms:

- **High Confidence Filter:** Only considers users whose most probable demographic class based on unigram classifier has a confidence $> .95$.
- **Random Sampling:** Considers all available users; randomly split each individual's tweets into two independent pools so that demographic and mental health inferences are based on separate sets of data.
- **Name Labels:** Only considers users from CLPSYCH who could be de-anonymized; demographics annotated using name-based gender (Wood-Doughty et al., 2018) and ethnicity classifiers (Wood-Doughty et al., 2020).

While we find some variation in the individual-level demographic labels when using the three techniques, the downstream mental health models perform similarly: see details in Appendix A. For the experiments discussed below, we report results from the most computationally-efficient approach, high confidence filtering.

6 Analysis

We conduct an analysis of these datasets and depression models trained on these datasets to answer the following questions:

1. Are depression datasets demographically representative?
2. Do depression classifiers perform similarly across demographic groups?
3. Can we mitigate demographic biases by changing characteristics of the dataset?
4. Do differences in features between demographic groups account for classifier biases?

6.1 Are depression datasets demographically representative?

Before we can empirically measure if these datasets are demographically representative, we must first establish the expected distribution of a representative dataset. While the demographic groups distribution should match the true population (Twitter users with depression), there are no estimates of depression prevalence on Twitter. Thus, we use the Twitter US population as our baseline, and combine it with estimated prevalence of depression among

US demographic groups.[2]

Methods. Brody et al. (2018) found in a study of adults (>20yrs.) that women are almost twice as likely to be diagnosed with depression compared to men (1.89×) using Patient Health Questionnaires (PHQ-9). We refer to this study as 'CDC.' Similarly, Hasin et al. (2018) used a national survey of adults (>18yrs.) and the DSM-5 standard for major depressive disorder (MDD) to estimate that women are almost twice as likely to be diagnosed with depression compared to men (1.86×). We refer to this study as 'NESARC.'

While there were only small incongruencies between these studies in estimated prevalence of depression as a function of gender, there were significant discrepancies between studies with respect to estimated prevalence as a function of race/ethnicity. Specifically, CDC found that rates of depression were not statistically different between groups, whereas NESARC found a greater prevalence of depression among Whites compared to African Americans, Hispanics/Latinx and Asian Americans (1.25x).

We project these estimates of depression prevalence to the general Twitter population, where both males and females are estimated to participate equally (approximately matching the US population). While there is a slight under-representation of White individuals in Twitter compared to US population (60% vs 64%), Black and Hispanic/Latinx individuals are well represented (Wojcik and Hughes, 2019). Thus, barring slight variations, Twitter roughly mimics the demographic composition of the United States demographics fairly with respect to gender and race/ethnicity.

We combine the Twitter population estimates with depression rates in the US to get the target distributions of demographic users that we expect to observe in our datasets. Figure 1 shows differences between the expected, representative distribution and our complete Twitter datasets.

Results. Based on these estimates, are the depression datasets demographically representative? Figure 1 shows that CLPSYCH and MULTITASK are **not demographically representative** with respect to either gender or race/ethnicity. White individuals are over-represented, while Hispanic/Latinx individuals are the most underrepresented. In fact, there are *no* male H/L individuals represented in the train split of CLPSYCH. MULTITASK exhibits a larger population bias against minorities compared to CLPSYCH; White individuals are over-represented and Black individuals are under-represented. With respect to gender, both CLPSYCH and MULTITASK have similar distributional skews – females are over-represented compared to the depression adjusted general US population. At the user level, we found no major differences on number of tweets and vocabulary size between demographics: see details in Appendix B.

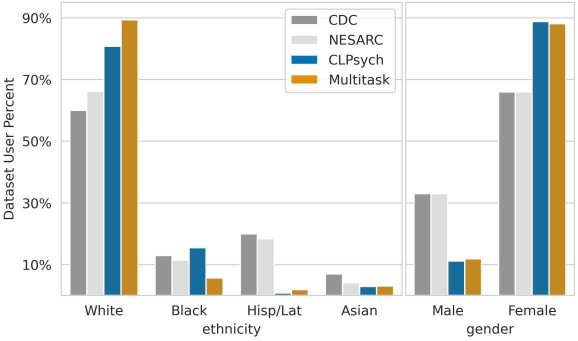


Figure 1: Hispanic/Latinx and Male individuals are underrepresented in both of our Twitter datasets (CLPSYCH & MULTITASK) compared to the Twitter US population estimates (CDC & NESARC).

Overall, CLPSYCH and MULTITASK are **not demographically representative** with respect to US depression rates projected on Twitter demographic estimates.

6.2 Do depression classifiers perform similarly across demographic groups?

We consider this question through experimentation on our datasets, CLPSYCH and MULTITASK.

Methods. We train a depression classifier on CLPSYCH and MULTITASK datasets.

We follow standard pre-processing procedures and filter numeric values, username mentions, retweets and urls from the raw tweet text. We use ℓ_2-regularized logistic regression models for all of our experiments. TF-IDF vectors are used to represent text in and across tweets, along with mean-pooled 200 dimensional GloVe embeddings pretrained with 2B tweets (Pennington et al., 2014). The vocabulary is pre-filtered per training, as each unigram must appear at least 10 times across all the individuals in traning data.

We also experimented with Linguistic Inquiry

---

[2] Our study uses mental health statistics from the United States since they are extensive and widely available. However, due to data anonymization we could not filter our data based on residence in the US. Since these datasets are filtered to focus on English accounts, US accounts likely dominate.

Word Count (LIWC) features, a closed-vocabulary English lexicon containing 64 categories (excluding punctuation categories), ranging from linguistic dimensions to psychological processes covering emotions and personal concerns, traditionally used in psychological studies (Pennebaker et al., 2007). In social media analysis, LIWC has been shown to contain signals for mental health disorders (Ireland and Iserman, 2018; Wolohan et al., 2018; Mitchell et al., 2015), including CLPsych and Multitask.

We also use features based on topic distributions learned via Latent Dirichlet Allocation (LDA) (Blei et al., 2003), following its implementation for Twitter data as specified in Mitchell et al. (2015) where all the tweets for an individual are combined into a "document" and we infer "topics" ($K = 50$ topics). All of the models in our experiments use all four feature groups: TF-IDF, GloVe embeddings, LIWC and LDA. We considered using demographic labels as features, which have been shown to capture signals for depression in Twitter (Preoţiuc-Pietro et al., 2015), but found no significant impact on our analysis or model performance; since demographic labels are not normally available, we do not include them in our analysis. See Appendix C for further implementation details.

To measure the performance bias across demographic groups we report performance on each demographic group. However, the racial/ethnic minority groups in the data are vastly underrepresented. While we address this by combining them into a 'persons of color' (PoC) category, the PoC group is still small and limits the reliability and extension of our analysis in this data.

For each dataset, we randomly sample individuals with repetition to construct a training set (bootstrap method) and subsequently obtain a distribution of F1 scores (100 repetitions) followed by one way ANOVA and pairwise T-Tests for each demographic group pair. Motivated by Simpson's Paradox (Blyth, 1972) and the Matrix of Domination (Costanza-Chock, 2018), we combine the gender and race/ethnicity labels to create a matrix of demographics and report mean F1 scores and 95% confidence interval of each demographic subcategory. Additionally, we seek a metric to measure fairness in performance across demographic groups — our criterion is that model performance should be independent of the demographic labels. Hardt et al. (2016) introduce *equal odds* and *equal opportunity*, two criteria that seek to equalize the FPR and TPR, or just FPR for the latter, across the protected attributes — these are also known as 'error rate balance' (Chouldechova, 2017), 'conditional procedure accuracy equality' (Berk et al., 2018) and 'classification parity' (Corbett-Davies and Goel, 2018). We compute the average pairwise equal odds and equal opportunity difference, a score of 0 means overall fairness, across the demographic groups in our boostrap sampling splits and report 95% confidence interval.

		CLPsych		Multitask	
Female	**White**	*0.77*	*± 0.005*	0.84	± 0.002
	PoC	0.41	± 0.013	0.91	± 0.003
Male	**White**	*0.74*	*± 0.008*	0.83	± 0.005
	PoC	*0.76*	*± 0.035*	0.45	± 0.016
	Equal Odds	0.21	± 0.023	0.13	± 0.013
	Opportunity	0.25	± 0.039	0.18	± 0.010

Table 1: Avg. F1 with 95% conf. interval from bootstrap across gender and ethnicity groups *(italics: not significant)*, and avg. equal odds and equal opportunity differences. Models underperform for PoC in general, sometimes male PoC (Multitask) or female PoC (CLPsych).

Results. Table 1 shows performance of classifiers trained on CLPsych and Multitask by demographic group. Models trained on CLPsych tend to perform worse on female PoC users compared to all other demographic groups. While we observe higher model performance for Multitask in general, models trained on Multitask tend to perform worse on male PoC users, compared to all other demographic groups. CLPsych is scored worse with the fairness metrics compared to Multitask.

In short, we observe that depression classifiers perform **worse on people of color**, specifically female PoC in CLPsych and male PoC Multitask.

6.3 Can we mitigate demographic biases by changing characteristics of the dataset?

Why do depression classifiers perform worse/inconsistently for PoC individuals? We conduct two analyses that investigate how the datasets may cause disparities in fairness.

6.3.1 Data Size

Perhaps the classifier performs worse on demographic groups because we have insufficient train-

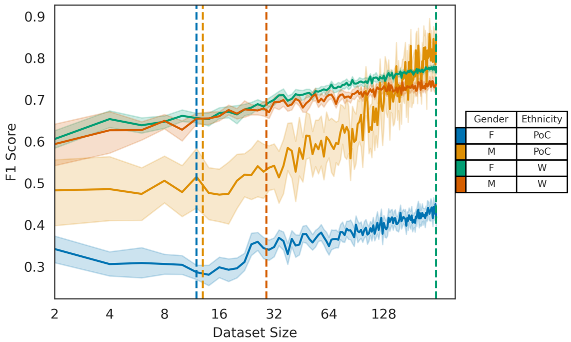


Figure 2: Log learning curve of varying dataset sample size on CLPSYCH. Dashed vertical lines represent the total size of the demographic group. Curves suggest more data is helpful to close model performance gap between demographic groups.

ing data. In Section 6.2, we observed fairer results with more data on MULTITASK compared to CLPSYCH. We perform a dataset size experiment to verify the effect on model performance across demographics.

Methods. How do results change with increased amounts of training data? To evaluate this gradient, we consider sampling an equal number of individuals from each demographic group and gradually increase overall dataset size until all available individuals available have been considered. At each dataset size step, we employ a similar bootstrap procedure to the one discussed in Section 6.2, sampling from the available user pool and training a classifier 25 times before moving on to the next dataset size. We continue adding data after a demographic group has been fully saturated to understand how information from overly-represented groups can generalize to under-represented groups.

Results. Figure 2 shows models performance as we increase training dataset size within the CLPSYCH dataset; results for MULTITASK are included in Appendix D and lead us to similar conclusions. As expected, overall performance across all classes improves with additional training data. Interestingly, even when the same amount of data is present for each demographic group, error rates remain higher for PoCs than for White users. This suggests that other factors beyond superficial representation are to blame for model degradation. It is also worthwhile noting that performance continues to improve for underrepresented groups after they have been fully saturated, thus implying that at least some signal generalizes between demographics.

		MULTITASK							
		full		NESARC		CDC		**even**	
Female	**White**	0.84	± 0.002	0.75	± 0.006	0.75	± 0.008	0.69	± 0.013
	PoC	0.91	± 0.003	0.87	± 0.005	0.86	± 0.005	0.82	± 0.009
Male	**White**	0.83	± 0.005	0.74	± 0.007	0.74	± 0.008	0.68	± 0.011
	PoC	0.45	± 0.016	0.56	± 0.029	0.54	± 0.033	0.48	± 0.030
	Equal Odds	0.13	± 0.013	0.14	± 0.021	0.14	± 0.019	0.12	± 0.014
	Opportunity	0.18	± 0.010	0.16	± 0.032	0.18	± 0.036	0.12	± 0.027

Table 2: Avg. F1 with 95% conf. interval from bootstrap across gender and ethnicity groups, and absolute avg. equal odds and equal opportunity differences. Balanced models close the performance difference gap at the cost of overall model performance.

6.3.2 Data Balance

What other factors could account for the difference in model performance on PoC? Below, we examine the effect of balancing the training data for the demographic groups.

Methods. We consider MULTITASK when constructing demographically-balanced datasets. As explored in Section 6.1, there are two different estimates of depression rates in demographic groups: CDC and NESARC. We balance MULTITASK to match the depression rates of both estimates, and name the models trained on those datasets MULTITASK CDC and MULTITASK NESARC respectively. Additionally, we compare these with an *even* balanced distribution. Models are trained following the methodology in Section 6.2.

Results. Table 2 shows the average F1 score of classifiers across gender and race/ethnicity groups. We copy MULTITASK column from Table 1 (labeled as *full*) for ease of comparison. There is a performance difference between male PoC users and the rest of the groups in models trained on MULTITASK balanced datasets, similar to MULTITASK full. However, the performance difference is smaller on models trained on balanced datasets. We observe no difference between balancing datasets according to NESARC or CDC, despite the 1.25x White user population increase in CDC. While fairness performance of both NESARC or CDC are similar to the full dataset, the *even* dataset shows considerable improvement for both fairness metrics at the cost of model performance.

Our experiments with both dataset size and balance show that it matters when datasets are not demographically representative, and as shown in section 6.1, they are not.

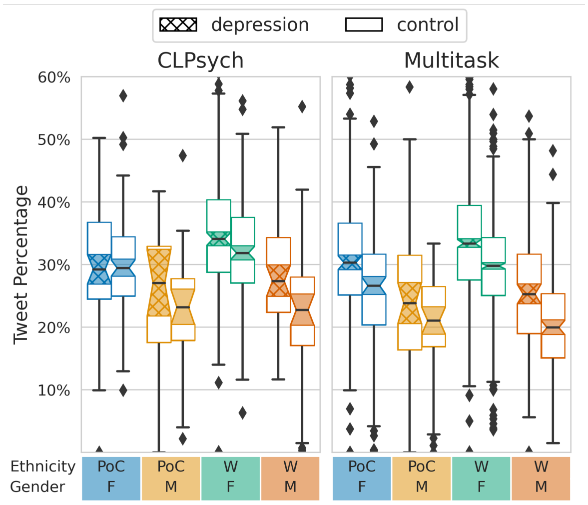


Figure 3: Percent of tweets containing first-person pronoun *(Pro1)* previously shown to correlate with the depression group; shaded notches show median CI (95%) with bootstrap $n = 10000$. Correlations shown by other work are not universal among demographic groups.

6.4 Can differences in features between demographic groups account for classifier biases?

We have demonstrated a demographic bias in classifiers trained on CLPSYCH and MULTITASK. Perhaps differences in feature representations between the groups can explain some of this bias. We examine LIWC features, which previous research has identified as useful in depression classification (Coppersmith et al., 2015c, 2014), in addition to performance analysis in Appendix E.

Methods. Previous research using the CLPSYCH and MULTITASK datasets has identified LIWC dimensions that over-index amongst depressed individuals: *Negative Emotion* (negemo), *Swearing* (swear), *Anger* (anger), *Anxiety* (anx) and *First-person Pronoun Usage* (Pro1) (Coppersmith et al., 2014).[3] We evaluate whether this finding holds within each demographic group independently and whether there exist shifts between demographic groups.

Results. Figure 3 shows the distribution over users of percentage of tweets with at least one word matching the *Pro1* category across demographic groups, with shaded notches showing median confidence interval. Results for other LIWC categories associated with depression are similar to those for *Pro1* (see Appendix F).

From previous research, we expect to observe a greater *Pro1* prevalence in depression groups compared to controls across all demographics i.e. shaded notches of depression box should not overlap with control in each demographic category. However, in CLPSYCH, we do not observe any difference in prevalence of *Pro1* in the PoC groups, and in MULTITASK we do not observe any difference in prevalence in the male PoC group, both contradicting previous results. We also observe a correlation between prevalence of these LIWC categories in the depression group and downstream model performance for each demographic group, corroborating previous findings of the correlation of LIWC categories and depression signals (Coppersmith et al., 2015c, 2014). In general, female groups for both the control and depression sets tend to have a higher prevalence of *Pro1* compared to their male counterparts, suggesting a difference in language between the groups.

In short, LIWC correlations with depression are not universal across demographic groups. Furthermore, a closed vocabulary feature, such as LIWC, may contribute towards bias against some demographic groups.

7 Limitations

Depression and Control Groups. The method used to curate the depression group in these datasets is susceptible to *self-selection* bias, as noted by Coppersmith et al. (2014) and Amir et al. (2019), as it likely over-represents individuals who are more vocal about their condition. Therefore, differences in use of social media and cultural perceptions around mental health may introduce biases in these datasets. Further, while expert annotators identified non-genuine disclosures depression and removed these individuals from the CLPSYCH and MULTITASK datasets, they did not verify the authenticity of the diagnosis. Similarly, individuals in the control group may have been actually diagnosed with depression, but did not disclose their condition anywhere in their public timeline. Thus, labels for both the the depression and control groups are bound to be noisy.

Representation. We balance the MULTITASK dataset to match depression rates in the US, which may not be representative of non-US populations. Additionally, we preserve the even depression/control splits for class balance in model training, in-

---

[3] *Pro1* is constructed by combining the *i* and *we* LIWC categories.

stead of using the true depression/control population rates (about 1/10). These splits are not reflective of the true depression prevalence and their use may need to be modified depending on downstream classifier use.

Demographic Labels. Due to dataset limitations, the ethnicity and gender labels for this study were inferred using the unigram model from (Wood-Doughty et al., 2020). This model considers only the four largest race/ethnicity groups in the US, which aligns with conventions from Brody et al. (2018) but ignores smaller populations and multi-racial categories. Further, in some of our analyses, we combine Asian, Black and Hispanic/Latinx individuals into people of color due to a lack of data. With respect to gender, the male and female labels used by this model do not consider individuals who fall outside of traditional binary gender. As our experiments rely on upstream demographic label inference, we cannot fully rule out confounding factors due to e.g. noisy labels in our experimentation, but we perform a high-confidence filter on demographic labels and statistical testing on results to strengthen our conclusions.

8 Conclusion and Recommendations

We examine whether datasets and the resulting trained classifiers for depression prediction are fair across demographic groups. Our analysis finds that (1) depression datasets are **not** demographically representative, in some cases excluding entire intersectional groups and (2) the resulting classifiers perform worse on people of color in general. In examining the reason for these differences, we find that performance difference could be improved after accounting for (3) the size of the dataset and balance across demographic groups. (4) Finally, we show that signals of depression found by previous work using e.g. LIWC features are *not* equally representative for all demographics.

These findings should give pause to researchers in this area. Since datasets and the resulting models are not demographically representative, advances in methods may be furthering biases towards some groups. Worse, since some intersectional demographic groups are not even represented in the data, compounded by the fact that most datasets do not have labels for demographic groups, we currently lack the means to even check how new methods perform on each group. Going forward, research in this area should include demographic analyses so that improvements on the overall dataset can be contextualized by how they perform on each demographic group.

At the same time, there is reason for optimism. Our data balancing and dataset size experiments reduced demographic disparities of trained models. This suggests that research can continue with existing datasets, but with the modifications we proposed. We release the demographically balanced dataset from our experimentation upon appropriate terms of usage agreement.

Ultimately, the best approach will be to construct new datasets that better represent the population, especially underrepresented minorities who are most at risk from systematic bias. This may necessitate changes to the data collection methods themselves, which may bias collection against certain groups. For example, self-reports may be problematic as they rely on cultural attitudes towards the expression of mental health information. Further research is needed to understand if self-reports and other proxy-based methods for obtaining labels can be successfully adapted to include a more diverse population, e.g. do keywords used to collect tweets skew resulting user populations? Further, to produce more conclusive insights with respect to demographics, language-based classifiers for demographic labels need to be further improved. Alternatively, other data collection strategies, such as the cohort method of Amir et al. (2019), may be more successful at ensuring representative datasets.

Acknowledgments

The authors gratefully acknowledge Elizabeth Salesky, Zach Wood-Doughty, Rachel Wicks, Alexandra DeLucia and the CLSP NLP reading group for helpful feedback, and we thank the anonymous reviewers for their helpful comments.

References

Silvio Amir, Glen Coppersmith, Paula Carvalho, Mário J. Silva, and Byron C. Wallace. 2017. Quantifying mental health from social media with neural user embeddings.

Silvio Amir, Mark Dredze, and John W. Ayers. 2019. Mental health surveillance over social media with digital cohorts. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 114–120, Minneapolis, Minnesota. Association for Computational Linguistics.

Alina Arseniev-Koehler, Sharon Mozgai, and Stefan

Scherer. 2018. What type of happiness are you looking for?-a closer look at detecting mental health from language. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 1–12.

John W Ayers, Theodore L Caputi, Camille Nebeker, and Mark Dredze. 2018. Don't quote me: reverse identification of research participants in social media studies. *NPJ digital medicine*, 1(1):1–2.

Bark. Best parental control monitoring app for iphone and android.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017a. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Adrian Benton, Margaret Mitchell, and Dirk Hovy. 2017b. Multi-task learning for mental health using social media text. *ArXiv*, abs/1712.03538.

Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. 2018. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, page 0049124118782533.

Brittany E Blanchard, Angela K Stevens, Kenneth J Sher, and Andrew K Littlefield. 2020. Reexamining the psychometric properties of the substance use risk profile scale. *Assessment*, 27(3):454–471.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Colin R Blyth. 1972. On simpson's paradox and the sure-thing principle. *Journal of the American Statistical Association*, 67(338):364–366.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*, pages 4349–4357.

Debra Brody, Laura Pratt, and Jeffery Hughes. 2018. Prevalence of depression among adults aged 20 and over: United states, 2013-2016. *NCHS data brief*, pages 1–8.

John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating gender on twitter. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1301–1309, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Stevie Chancellor and Munmun De Choudhury. 2020. Methods in predictive techniques for mental health status on social media: a critical review. *NPJ digital medicine*, 3(1):1–11.

Hau-wen Chang, Dongwon Lee, Mohammed Eltaher, and Jeongkyu Lee. 2012. @ phillies tweeting from philly? predicting twitter user locations with spatial word usage. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 111–118. IEEE.

Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

Glen Coppersmith, Mark Dredze, and Craig Harman. 2014. Quantifying mental health signals in twitter. In *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 51–60, Baltimore, Maryland, USA. Association for Computational Linguistics.

Glen Coppersmith, Mark Dredze, Craig Harman, and Kristy Hollingshead. 2015a. From ADHD to SAD: Analyzing the language of mental health on twitter through self-reported diagnoses. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Glen Coppersmith, Mark Dredze, Craig Harman, Kristy Hollingshead, and Margaret Mitchell. 2015b. CLPsych 2015 shared task: Depression and PTSD on twitter. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 31–39, Denver, Colorado. Association for Computational Linguistics.

Glen Coppersmith, Ryan Leary, and Eric Whyne. 2015c. Quantifying suicidal ideation via language usage on social media.

Sam Corbett-Davies and Sharad Goel. 2018. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*.

Sasha Costanza-Chock. 2018. Design justice: towards an intersectional feminist framework for design theory and practice. *Proceedings of the Design Research Society*.

Munmun De Choudhury, Scott Counts, Eric J. Horvitz, and Aaron Hoff. 2014. Characterizing and predicting postpartum depression from shared facebook data. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, page 626–638, New York, NY, USA. Association for Computing Machinery.

Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting depression via social media. AAAI.

Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 2098–2110, New York, NY, USA. Association for Computing Machinery.

Clayton Fink, Jonathon Kopecky, and Maksym Morawski. 2012. Inferring gender from the content of tweets: A region specific example. *ICWSM*, 12:459–462.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. *arXiv preprint arXiv:1903.03862*.

Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of opportunity in supervised learning. *arXiv preprint arXiv:1610.02413*.

Keith Harrigian, Carlos Aguirre, and Mark Dredze. 2020. On the state of social media data for mental health research. *arXiv preprint arXiv:2011.05233*.

Deborah S. Hasin, Aaron L. Sarvet, Jacquelyn L. Meyers, Tulshi D. Saha, W. June Ruan, Malka Stohl, and Bridget F. Grant. 2018. Epidemiology of Adult DSM-5 Major Depressive Disorder and Its Specifiers in the United States. *JAMA Psychiatry*, 75(4):336–346.

Joseph Henrich, Steven J. Heine, and Ara Norenzayan. 2010. The weirdest people in the world? *Behavioral and Brain Sciences*, 33(2-3):61–83.

Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (volume 2: Short papers)*, pages 483–488.

Dirk Hovy and Shannon L Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598.

Molly Ireland and Micah Iserman. 2018. Within and between-person differences in language used across anxiety support and neutral Reddit communities. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 182–193, New Orleans, LA. Association for Computational Linguistics.

Anders Johannsen, Dirk Hovy, and Anders Søgaard. 2015. Cross-lingual syntactic variation over age and gender. In *Proceedings of the nineteenth conference on computational natural language learning*, pages 103–112.

Ronald C Kessler, Patricia Berglund, Olga Demler, Robert Jin, Doreen Koretz, Kathleen R Merikangas, A John Rush, Ellen E Walters, and Philip S Wang. 2003. The epidemiology of major depressive disorder: results from the national comorbidity survey replication (ncs-r). *Jama*, 289(23):3095–3105.

Brian Larson. 2017. Gender as a variable in natural-language processing: Ethical considerations. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 1–11, Valencia, Spain. Association for Computational Linguistics.

Veronica Lynn, Alissa Goodman, Kate Niederhoffer, Kate Loveys, Philip Resnik, and H. Andrew Schwartz. 2018. CLPsych 2018 shared task: Predicting current and future psychological health from childhood essays. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 37–46, New Orleans, LA. Association for Computational Linguistics.

Kari McDonald. 2018. Social support and mental health in lgbtq adolescents: a review of the literature. *Issues in mental health nursing*, 39(1):16–29.

Margaret Mitchell, Kristy Hollingshead, and Glen Coppersmith. 2015. Quantifying the language of schizophrenia in social media. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 11–20, Denver, Colorado. Association for Computational Linguistics.

Shirin Nilizadeh, Anne Groggel, Peter Lista, Srijita Das, Yong-Yeol Ahn, Apu Kapadia, and Fabio Rojas. 2016. Twitter's glass ceiling: The effect of perceived gender on online visibility. In *Tenth International AAAI Conference on Web and Social Media*.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010. Tweetmotif: Exploratory search and topic summarization for twitter. In *Fourth International AAAI Conference on Weblogs and Social Media*.

Robin Ohannessian, Tu Anh Duong, and Anna Odone. 2020. Global telemedicine implementation and integration within health systems to fight the covid-19 pandemic: a call to action. *JMIR public health and surveillance*, 6(2):e18810.

James W Pennebaker, Roger J Booth, and Martha E Francis. 2007. Operator's manual: Linguistic inquiry and word count: Liwc2007. *Austin, Texas: LIWC. net http://homepage. psy. utexas. edu/HomePage/Faculty/Pennebaker/Reprints/LIWC2007_OperatorManual. pdf (accessed 1 October 2013)*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Daniel Preoţiuc-Pietro, Johannes Eichstaedt, Gregory Park, Maarten Sap, Laura Smith, Victoria Tobolsky, H. Andrew Schwartz, and Lyle Ungar. 2015. The role of personality, age, and gender in tweeting about mental illness. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 21–30, Denver, Colorado. Association for Computational Linguistics.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ.

Delip Rao, Michael J Paul, Clayton Fink, David Yarowsky, Timothy Oates, and Glen Coppersmith. 2011. Hierarchical bayesian models for latent attribute detection in social media. *ICWSM*, 11:598–601.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. *arXiv preprint arXiv:1804.09301*.

H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. 2013. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, 8(9):e73791.

Sherri M Simpson, Laura L Krishnan, Mark E Kunik, and Pedro Ruiz. 2007. Racial disparities in diagnosis and treatment of depression: a literature review. *Psychiatric Quarterly*, 78(1):3–14.

Nikhita Vedula and Srinivasan Parthasarathy. 2017. Emotional and linguistic cues of depression from social media. In *Proceedings of the 2017 International Conference on Digital Health*, pages 127–136.

Svitlana Volkova, Glen Coppersmith, and Benjamin Van Durme. 2014. Inferring user political preferences from streaming communications. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 186–196, Baltimore, Maryland. Association for Computational Linguistics.

Stefan Wojcik and Adam Hughes. 2019. How twitter users compare to the general public.

JT Wolohan, Misato Hiraga, Atreyee Mukherjee, Zeeshan Ali Sayyed, and Matthew Millard. 2018. Detecting linguistic traces of depression in topic-restricted text: Attending to self-stigmatized depression with NLP. In *Proceedings of the First International Workshop on Language Cognition and Computational Models*, pages 11–21, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zach Wood-Doughty, Nicholas Andrews, Rebecca Marvin, and Mark Dredze. 2018. Predicting twitter user demographics from names alone. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 105–111, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Zach Wood-Doughty, Paiheng Xu, Xiao Liu, and Mark Dredze. 2020. Using noisy self-reports to predict twitter user demographics.

Amir Hossein Yazdavar, Mohammad Saeid Mahdavinejad, Goonmeet Bajaj, William Romine, Amit Sheth, Amir Hassan Monadjemi, Krishnaprasad Thirunarayan, John M Meddar, Annie Myers, Jyotishman Pathak, et al. 2020. Multimodal mental health analysis in social media. *Plos one*, 15(4):e0226248.

Dong Whi Yoo, Michael L Birnbaum, Anna R Van Meter, Asra F Ali, Elizabeth Arenare, Gregory D Abowd, and Munmun De Choudhury. 2020. Designing a clinician-facing tool for using insights from patients' social media activity: Iterative co-design approach. *JMIR Mental Health*, 7(8):e16969.

Dong Whi Yoo and Munmun De Choudhury. 2019. Designing dashboard for campus stakeholders to support college student mental health. In *Proceedings of the 13th EAI International Conference on Pervasive Computing Technologies for Healthcare*, pages 61–70.

Xiaoyun Zhou, Centaine L Snoswell, Louise E Harding, Matthew Bambling, Sisira Edirippulige, Xuejun Bai, and Anthony C Smith. 2020. The role of telehealth in reducing the mental health burden from covid-19. *Telemedicine and e-Health*, 26(4):377–379.

A Demographic Labels Analysis

Due to data anonymization, we must utilize content-based demographic classifiers to infer labels. This introduces noise due to classification error to our analyses. In order to reduce these effects we consider 3 techniques: high confidence filter, name labels, and tweet sampling.

High Confidence Filter. We select users whose most probably demographic class for both gender and race is > 0.95 probability.

Random Sampling. Considers all available users; randomly split each individual's tweets into two independent pools so that demographic and mental health inferences are based on separate sets of data.

Name Labels. With the permission of Coppersmith et al. (2014), we were able to collect name attributes for 622 of individuals in CLPsych. To obtain demographic labels from name attributes we used the demographer's neural name classifier (Wood-Doughty et al., 2018), and ethnic/race name classifier (Wood-Doughty et al., 2020). Due to the small number of individuals that we could obtain name attributes, our analysis of this technique is limited.

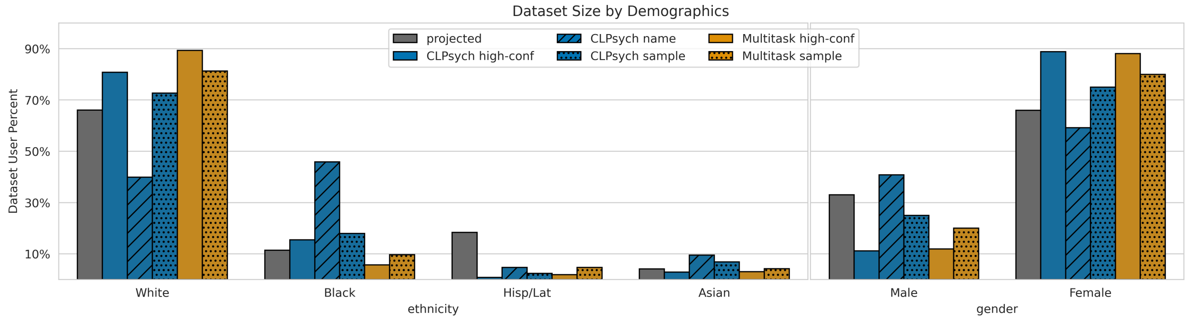


Figure 4: Hispanic/Latinx and Male individuals are underrepresented in both of our Twitter datasets (CLPsych & Multitask) compared to the Twitter US population estimates (NESARC) in all our labeling techniques.

Figure 4 shows the population percentage of the three techniques compared to the projected distribution. For all the techniques, some trends still hold, mainly Hispanic/Latinx group is underrepresented. Additionally, for the high confidence and split techniques, White and female groups are overrepresented and male underrepresented. In contrast, in the name based approach, the Black and male groups are overrepresented while the White and female groups are underrepresented.

While the name based technique shows more promising data distributions, the Hispanic/Latinx group is still vastly underrepresented. However, the rest of the demographic groups seem more fair and closer (sometimes better) than the target distribution, but do these distributions translate to greater downstream performance?

Table 3 shows the performance of mental health models trained on the different datasets obtained from our demographic labeling techniques. High confidence filtering has the largest performance difference and higher fairness metrics compared to random sampling and name based approaches. However, we still observe similar trends with PoC groups performing in general lower than White groups.

		CLPsych			Multitask	
		High Conf.	**Rand. Sample**	**Name Labels**	**High Conf.**	**Rand. Sample**
Female	**White**	*0.77 ± 0.005*	0.77 ± 0.005	0.72 ± 0.006	0.84 ± 0.002	0.84 ± 0.003
	PoC	0.41 ± 0.013	0.47 ± 0.020	0.44 ± 0.018	0.91 ± 0.003	0.87 ± 0.010
Male	**White**	*0.74 ± 0.008*	0.74 ± 0.014	0.48 ± 0.032	0.83 ± 0.005	0.78 ± 0.011
	PoC	*0.76 ± 0.035*	0.72 ± 0.040	0.56 ± 0.042	0.45 ± 0.016	0.78 ± 0.033
	Equal Odds	0.21 ± 0.023	0.13 ± 0.014	0.19 ± 0.023	0.13 ± 0.013	0.13 ± 0.021
	Opportunity	0.25 ± 0.039	0.16 ± 0.022	0.18 ± 0.031	0.18 ± 0.010	0.14 ± 0.034

Table 3: Avg. F1 with 95% conf. interval from bootstrap across gender and ethnicity groups, and avg. equal odds and equal opportunity differences *(italics: not significant)*. Models underperform for PoC in general, sometimes male PoC (Multitask) or female PoC (CLPsych).

B Tweets and Vocabulary in Demographics

Figure 5 shows the distribution of the average number of tweets and vocabulary size across all our demographic groups. There is no statistical significant difference in the means between demographic groups within the datasets. Very high variance is observed in the Asian group in multitask, although their vocabulary size is not as high variance. Additionally, MULTITASK dataset has on average higher number of tweets per user (as they were not limited to 3000 as in CLPSYCH) and consequently a higher average vocabulary size.

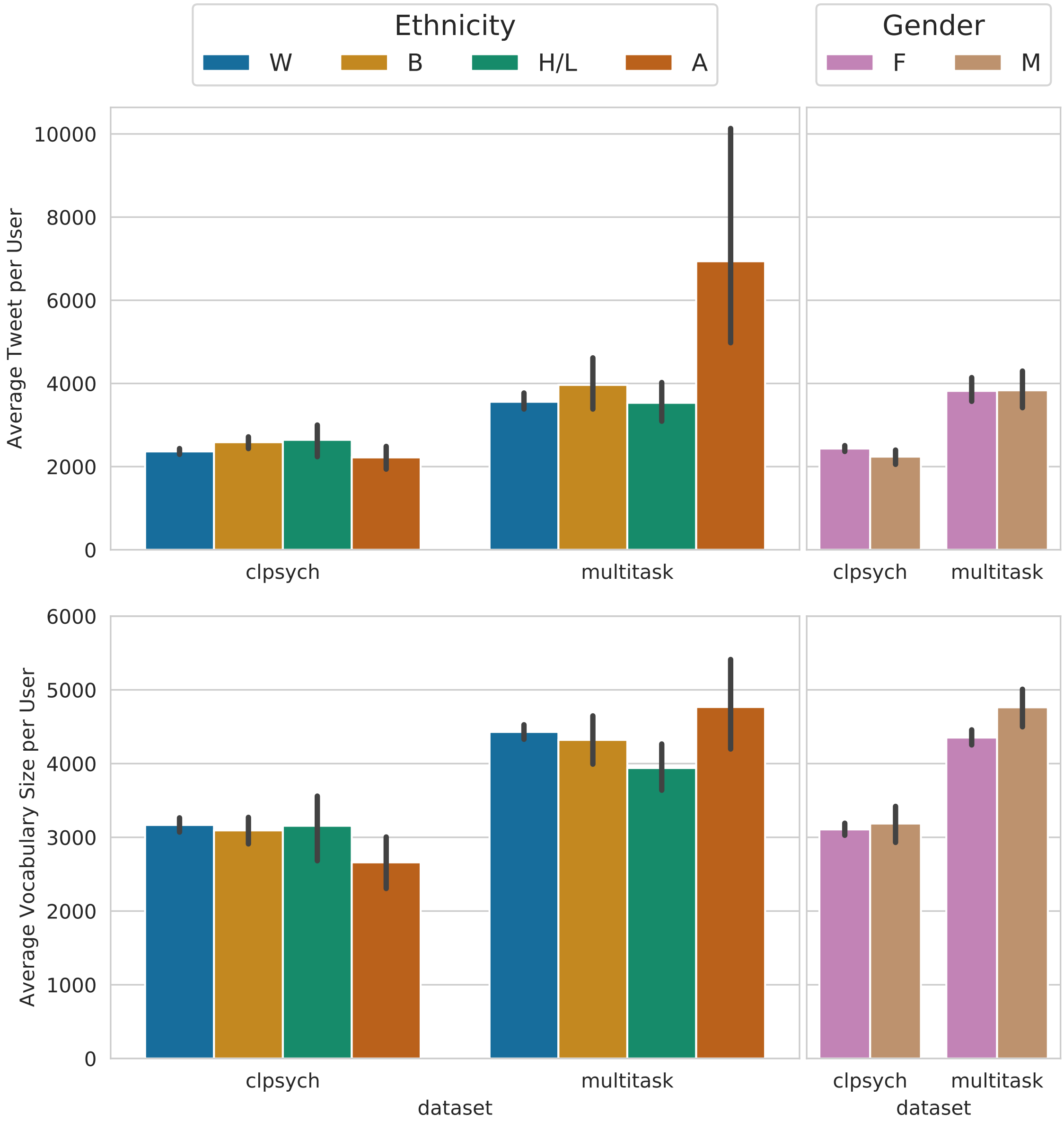


Figure 5: Average tweet number per user per demographic category, as well as average vocabulary size. Black bars show 95% CI.

C Model Specifications

Tokenization. Raw text within in Tweets was tokenized using a modified version of the Twokenizer (O'Connor et al., 2010). English contractions were expanded, while specific retweet tokens, username mentions, URLs, and numeric values were replaced by generic tokens. As pronoun usage tends to differ in individuals living with depression (Vedula and Parthasarathy, 2017), we removed any English pronouns from our stop word set (English Stop Words from nltk.org). Case was standardized across all tokens, with a single flag included if an entire post was made in uppercase letters.

Features. Text from all documents for an individual are concatenated together and tokenized as previously described. The vocabulary of each training procedure is fixed to a maximum of 100-thousand unigrams selected based on KL-divergence of the class-unigram distribution with the class-distribution of stop words (Chang et al., 2012). This reduced bag-of-words representation is then used to generate the following additional feature dimensions: a 50-dimensional LDA topic distribution (Blei et al., 2003), a 64-dimensional LIWC category distribution (Pennebaker et al., 2007), and a 200-dimensional mean-pooled vector of GloVe embeddings (Pennington et al., 2014). The reduced bag-of-words representation is transformed using TF-IDF weighting (Ramos et al., 2003).[4]

Hyperparameter Selection. Each model is trained using a hyperparameter grid search over the regularization strength {1e-3, 1e-2, 1-e1, 1, 10, 100, 1e3, 1e4, 1e5}, class weighting {None, Balanced}, and feature set standardization {On, Off}. Hyperparameters were selected to maximize held-out F1 score within a 20%-sied held-out split of the training data.

---

[4] All data-specific feature transformations (e.g. LDA, TF-IDF) are learned without access to development or test data.

D Multitask Size Experiment

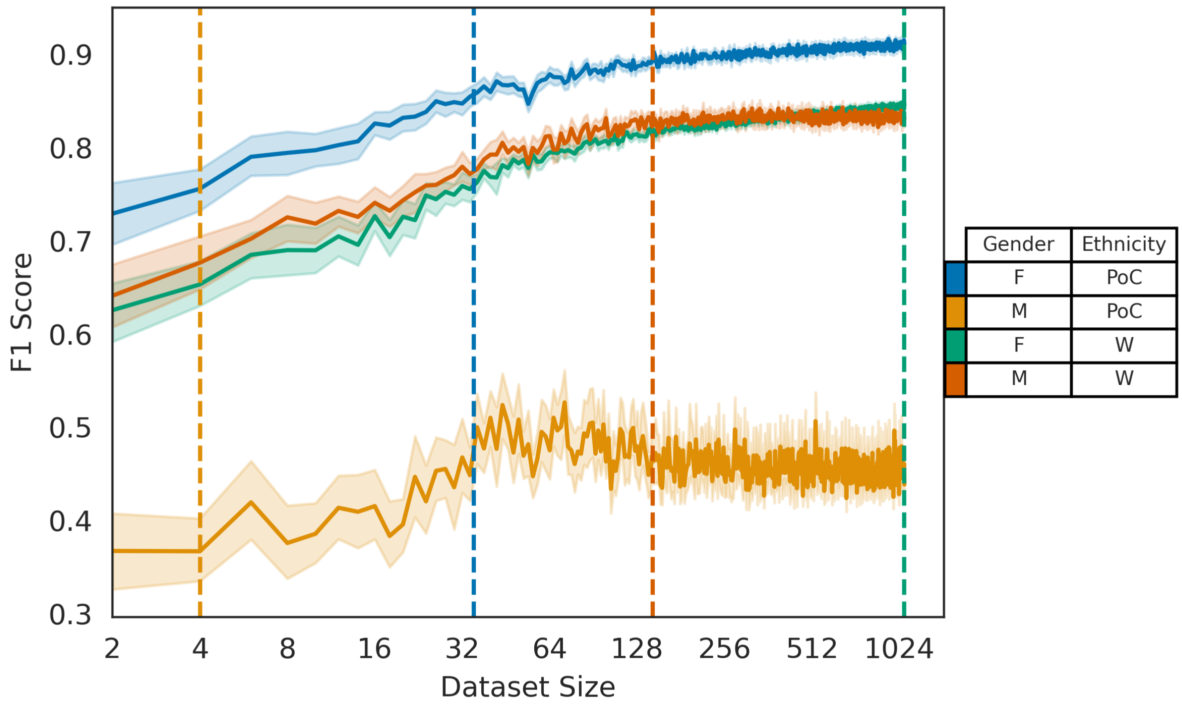


Figure 6: Log learning curve of varying dataset sample size on MULTITASK. Dashed vertical lines represent the total size of the demographic group.

E Feature Study

We observed performance difference between demographic groups for both our datasets with mental health models using the following feature groups: LIWC, LDA, GloVe and TF-IDF as specified in Appendix C. To explore the source of the performance difference observed in demographic groups, we train classifiers with each individual feature group and present average F1 score per demographic groups as well as our fairness metrics in Table 4.

TF-IDF and GloVe embeddings yield better model performance than the other feature groups at the expense of fairness as measure by our fairness metrics. The most fair feature set was LIWC, although it was also the least informative feature set resulting in worst performing models.

However, trends observed in Section 6.2 still apply in all feature groups, mainly models tend to underperform for PoC groups (female PoC groups in CLPsych).

		CLPsych									
		full		**TF-IDF**		**LDA**		**LIWC**		**GloVe**	
Female	**White**	*0.77*	*± 0.005*	0.77	± 0.005	0.70	± 0.008	0.68	± 0.010	0.71	± 0.007
	PoC	0.41	± 0.013	0.42	± 0.001	0.41	± 0.026	0.45	± 0.020	0.47	± 0.020
Male	**White**	*0.74*	*± 0.008*	0.74	± 0.008	0.65	± 0.019	0.68	± 0.021	0.68	± 0.019
	PoC	*0.76*	*± 0.035*	0.74	± 0.034	0.62	± 0.054	0.55	± 0.040	0.75	± 0.032
	Equal Odds	0.21	± 0.023	0.20	± 0.023	0.21	± 0.031	0.17	± 0.021	0.21	± 0.022
	Opportunity	0.25	± 0.039	0.25	± 0.040	0.24	± 0.039	0.17	± 0.028	0.18	± 0.029

Table 4: Avg. F1 with 95% conf. interval from bootstrap across gender and ethnicity groups, and absolute avg. equal odds and equal opportunity differences. PoC groups (female PoC) perform worse for models separately trained on each of our feature group.

F Additional LIWC Categories Figures

Previous research has identified specific LIWC dimensions that are of importance in depression groups. In addition to *First-person pronoun* (pro1), categories like *Negative Emotion* (negemo), *Swearing* (swear), *Anger* (anger) and *Anxiety* (anx) have been shown to be more prominent in the depression groups compared to control. We observe variation on prevalence on depression groups in all categories mentioned above among demographic groups, showing that LIWC features are *not* equally representative for all demographics

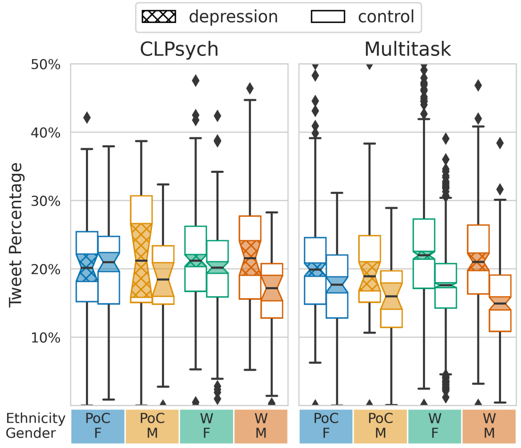


Figure 7: Negative Emotion Usage (*negemo*) LIWC category representation within each individual, previously shown to correlate with the depression group; statistical significance marked by not overlapping shaded notches. We observe median statistical difference only for **White male** individuals in CLPSYCH and **White groups** in MULTITASK.

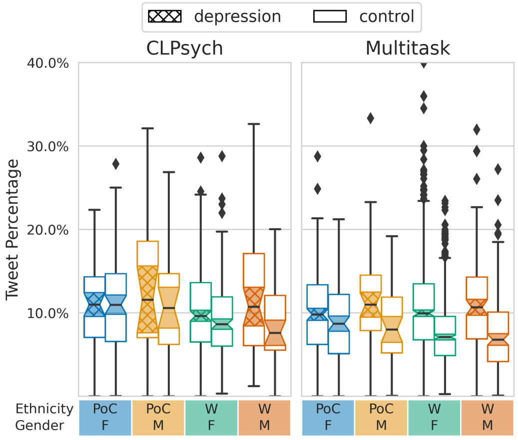


Figure 8: Anger (*anger*) LIWC category representation within each individual, previously shown to correlate with the depression group; statistical significance marked by not overlapping shaded notches. We only observe median statistical difference for **no groups** CLPSYCH and **White groups** in MULTITASK

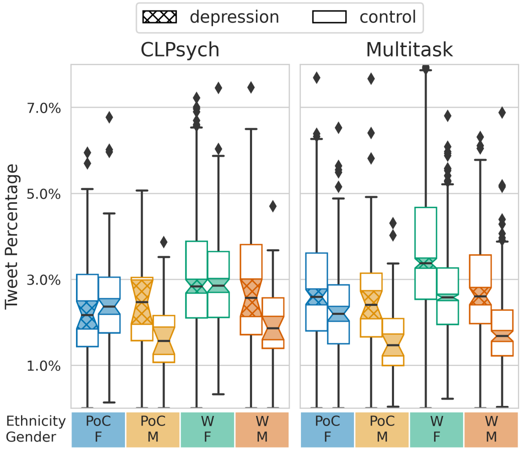


Figure 9: Anxiety (*anx*) LIWC category representation within each individual, previously shown to correlate with the depression group; statistical significance marked by not overlapping shaded notches. We observe median statistical difference for the **male groups** in CLPSYCH and **all** demographic categories in MULTITASK

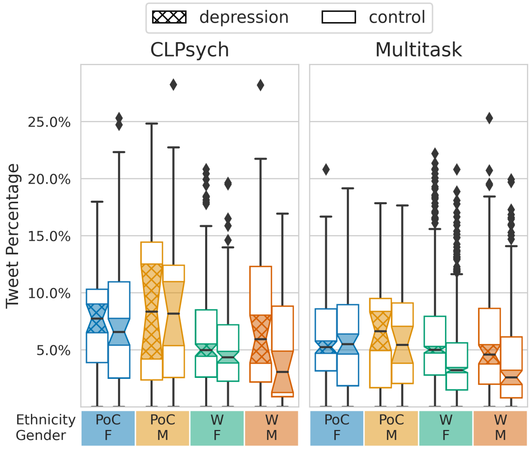


Figure 10: Swearing (*swear*) LIWC category representation within each individual, previously shown to correlate with the depression group; statistical significance marked by not overlapping shaded notches. We observe median statistical difference for **no groups** in CLPSYCH and **White groups** in MULTITASK.

MTOP: A Comprehensive Multilingual Task-Oriented Semantic Parsing Benchmark

Haoran Li **Abhinav Arora** **Shuohui Chen** **Anchit Gupta**
Sonal Gupta **Yashar Mehdad**
Facebook
{aimeeli,abhinavarora,shuohui,anchit}@fb.com
{mehdad,sonalgupta}@fb.com

Abstract

Scaling semantic parsing models for task-oriented dialog systems to new languages is often expensive and time-consuming due to the lack of available datasets. Available datasets suffer from several shortcomings: a) they contain few languages b) they contain small amounts of labeled examples per language c) they are based on the simple intent and slot detection paradigm for non-compositional queries. In this paper, we present a new multilingual dataset, called **MTOP**, comprising of 100k annotated utterances in 6 languages across 11 domains. We use this dataset and other publicly available datasets to conduct a comprehensive benchmarking study on using various state-of-the-art multilingual pre-trained models for task-oriented semantic parsing. We achieve an average improvement of +6.3 points on Slot F1 for the two existing multilingual datasets, over best results reported in their experiments. Furthermore, we demonstrate strong zero-shot performance using pre-trained models combined with automatic translation and alignment, and a proposed distant supervision method to reduce the noise in slot label projection.

1 Introduction

With the rising adoption of virtual assistant products, task-oriented dialog systems have been attracting more attention in both academic and industrial communities. One of the first steps in these systems is to extract meaning from the natural language used in conversation to build a semantic representation of the user utterance. Typical systems achieve this by classifying the *intent* of the utterance and tagging the corresponding *slots*. With the goal of handling more complex queries, recent approaches propose hierarchical representations (Gupta et al., 2018) that are expressive enough to capture the task-specific semantics of complex nested queries.

Although, there have been sizable efforts around developing successful semantic parsing models for task-oriented dialog systems in English (Mesnil et al., 2013; Liu and Lane, 2016; Gupta et al., 2018; Rongali et al., 2020), we have only seen limited works for other languages. This is mainly due to the painstaking process of manually annotating and creating large datasets for this task in new languages. In addition to the shortage of such datasets, existing datasets (Upadhyay et al., 2018; Schuster et al., 2019a) are not sufficiently diversified in terms of languages and domains, and do not capture complex nested queries. This makes it difficult to perform more systematic and rigorous experimentation and evaluation for this task across multiple languages.

Building on these considerations and recent advancements on cross-lingual pre-trained models (Devlin et al., 2019; Lample and Conneau, 2019; Conneau et al., 2020), this paper is making an effort to bridge the above mentioned gaps. The main contributions of this paper can be summarized as follows:

- MTOP Dataset: We release an almost-parallel multilingual task-oriented semantic parsing dataset covering **6 languages** and **11 domains**. To the best of our knowledge, this is the first multilingual dataset which contains compositional representations that allow complex nested queries.

- We build strong benchmarks on the released MTOP dataset using state-of-the-art multilingual pre-trained models for both flat and compositional representations. We demonstrate the effectiveness of our approaches by achieving new state-of-the-art result on existing multilingual task-oriented semantic parsing datasets.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2950–2962
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

- We demonstrate strong performance on zero-shot cross-lingual transfer using automatic translation and alignment, combined with a proposed distant supervision approach. We achieve 67.2% exact match accuracy (averaged across 5 languages) without using any target language data compared to best in-language model performance of 77.7%.

2 Related Work

Task-Oriented Semantic Parsing The majority of the work on task-oriented dialog systems has been centered around intent detection and slot filling - for example, the representations used on the ATIS dataset (Mesnil et al., 2013; Liu and Lane, 2016; Zhu and Yu, 2017) and in the Dialog State Tracking Challenge (Williams et al., 2016). This essentially boils down to a text classification and a sequence labeling task, which works great for simple non-compositional queries. For more complex queries with recursive slots, state of the art systems use hierarchical representations, such as the TOP representation (Gupta et al., 2018), that is modeled using Recurrent Neural Network Grammars (Dyer et al., 2016) or as a Sequence to Sequence task (Rongali et al., 2020).

Pre-trained Cross-lingual Representation Over the past few years, pre-trained cross-lingual representations have demonstrated tremendous success in achieving state of the art in various NLP tasks. The majority of the earlier work focuses on cross-lingual emebedding alignment (Mikolov et al., 2013; Ammar et al., 2016; Lample et al., 2018). Schuster et al. (2019b) further extend upon this by aligning contextual word embeddings from the ELMo model (Peters et al., 2018). Later with the success of Transformer (Vaswani et al., 2017) based masked language model pre-training, Devlin et al. (2019) and Lample and Conneau (2019) introduce mBERT and XLM respectively, and Pires et al. (2019) show the effectiveness of these on sequence labeling tasks. Conneau et al. (2020) present XLM-R, a pre-trained multilingual masked language model trained on data in 100 languages, that provides strong gains over XLM and mBERT on classification and sequence labeling tasks.

The models discussed above are encoder-only models. More recently, multilingual seq-to-seq pre-training has become popular. Liu et al. (2020a) introduce mBART, a seq-to-seq denoising auto-encoder pre-trained on monolingual corpora in many languages, which extends BART (Lewis et al., 2020b) to a multilingual setting. More recently, Lewis et al. (2020a) introduced a seq-to-seq model pre-trained on a multilingual multi-document paraphrasing objective, which self-supervises the reconstruction of target text by retrieving a set of related texts and conditions on them to maximize the likelihood of generating the original. Tran et al. (2020) is another contemporary work that mines parallel data using encoder representations and jointly trains a seq-to-seq model on this parallel data.

Cross-Lingual Task-Oriented Semantic Parsing Due to the ubiquity of digital assistants, the task of cross-lingual and multilingual task-oriented dialog has garnered a lot of attention recenty, and few multilingual benchmark datasets have been released for the same. To the best of our knowledge, all of them only contain simple non-compositional utterances, suitable for the intent and slots detection tasks. Upadhyay et al. (2018) release a benchmark dataset in Turkish and Hindi (600 training examples), obtained by translating utterances from the ATIS corpus (Price, 1990) and using Amazon Mechanical Turk to generate phrase level slot annotation on translations. Schuster et al. (2019a) release a bigger multilingual dataset for task-oriented dialog in English, Spanish and Thai across 3 domains. They also propose various modeling techniques such as using XLU embeddings (see Ruder et al. (2017) for literature review) for cross-lingual transfer, translate-train and ELMo (Peters et al., 2018) for target language training. BERT-style multilingual pre-trained models have also been applied to task-oriented semantic parsing. Castellucci et al. (2019) use multilingual BERT for joint intent classification and slot filling, but they don't evaluate on existing multilingual benchmarks. Instead, they introduce a new Italian dataset obtained via automatic machine translation of SNIPS (Coucke et al., 2018), which is of lower quality. For zero shot transfer, Liu et al. (2020b) study the idea of selecting some parallel word pairs to generate code-switching sentences for learning the inter-lingual semantics across languages and compare the performance using various cross-lingual pre-trained models including mBERT and XLM.

3 Data

Existing multilingual task-oriented dialog datasets, such as Upadhyay et al. (2018); Schuster et al.

Domain	**Number of utterances (training/validation/testing)**						**Intent types**	**Slot types**
	English	**German**	**French**	**Spanish**	**Hindi**	**Thai**		
Alarm	2,006	1,783	1,581	1,706	1,374	1,510	6	5
Calling	3,129	2,872	2,797	2,057	2,515	2,490	19	14
Event	1,249	1,081	1,050	1,115	911	988	12	12
Messaging	1,682	1,053	1,239	1,335	1,163	1,082	7	15
Music	1,929	1,648	1,499	1,312	1,508	1,418	27	12
News	1,682	1,393	905	1,052	1,126	930	3	6
People	1,768	1,449	1,392	763	1,408	1,168	17	16
Recipes	1,845	1,586	1,002	762	1,378	929	3	18
Reminder	1,929	2,439	2,321	2,202	1,781	1,833	19	17
Timer	1,488	1,358	1,013	1,165	1,152	1,047	9	5
Weather	2,372	2,126	1,785	1,990	1,815	1,800	4	4
Total	22,288	18,788	16,584	15,459	16,131	15,195	117	78

Table 1: Summary statistics of the MTOP dataset. The Data is roughly divided into 70:10:20 percent splits for train, eval and test.

(2019a), rely on expensive manual work for preparing guidelines and annotations for other languages; which is probably why they only contain very few languages and few labeled data examples for other languages. Furthermore, annotations will be more complicated and expensive if they were to include compositional queries, where slots can have nested intents. To this end we create an almost parallel multilingual task-oriented semantic parsing corpora which contains **100k examples** in total for **6 languages** (both high and low resource): *English*, *Spanish*, *French*, *German*, *Hindi* and *Thai*. Our dataset contains a mix of both simple and compositional nested queries across **11 domains**, **117 intents** and **78 slots**. Table. 1 shows a summary statistics of our MTOP dataset.

We release the dataset at `https://fb.me/mtop_dataset`.

3.1 Dataset Creation

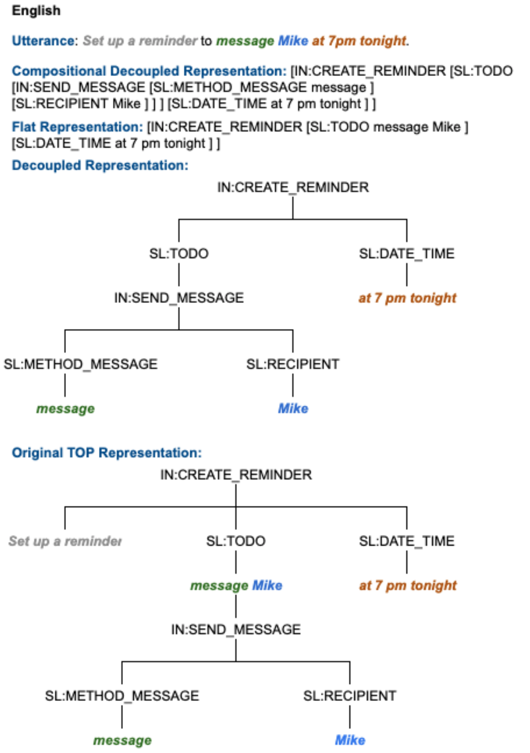
Our approach for creating this dataset consists of two main steps: i) generating synthetic utterances and annotating in English, ii) translation, label transfer, post-processing, post editing and filtering for other 5 languages. Generating the English utterances and their annotations, for the 11 domains, follows the exact process as described in (Gupta et al., 2018). We ask crowdsourced workers to generate natural language sentences that they would ask a system which could assist in queries corresponding to our chosen domains. These queries are labeled by two annotators. A third annotator is used only to adjudicate any disagreements. Once an annotated English dataset is available, we build the multilingual dataset through the following steps:

Translation: We first extract slot text spans from English annotation and present the utterances along with slot text spans to professional translators for translation to the target language. We prepare detailed guidelines, where we ask the translators to ensure that the translation for each slot span is exactly in the same way as it occurs in the translated utterance. For example, when translating the slot span *mom* in utterance *call my mom*, we ask the translators to use the same target language word for *mom*, that they used in the translation for *call my mom*.

Post-processing: After we obtain the translation of utterances and corresponding slot text spans, we use the tree structure of English and fill in the translated slot text spans to construct the annotation in the target languages. Our representation, described in §3.2.1, enables us to reconstruct the annotations.

Post-editing and Quality Control: We further run two rounds of quality control over translated utterances and slots, and revise the data accordingly. In the first round, we ask translators to review and post-edit the errors in translations and slot alignments. In the second round, the constructed target language data is presented to different annotators for a lightweight annotation quality review. 83% of the data was marked as good quality data and passed our quality standards, which can be interpreted as the inter-annotator agreement rate on the translated data. Based on this feedback, we remove low quality annotations from the dataset.

To create this dataset, for each target language we had three translators: two were responsible for translation and the third one for review and edits.



English

Utterance: *Set up a reminder* to ***message*** ***Mike*** ***at 7pm tonight***.

Compositional Decoupled Representation: [IN:CREATE_REMINDER [SL:TODO [IN:SEND_MESSAGE [SL:METHOD_MESSAGE message] [SL:RECIPIENT Mike]]] [SL:DATE_TIME at 7 pm tonight]]

Flat Representation: [IN:CREATE_REMINDER [SL:TODO message Mike] [SL:DATE_TIME at 7 pm tonight]]

Decoupled Representation:

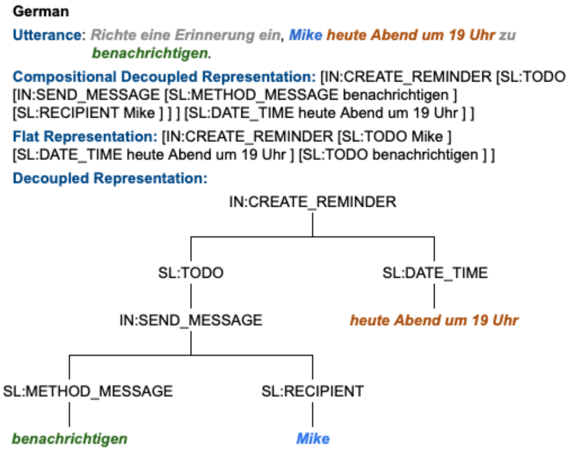
Original TOP Representation:

Figure 1: An English example from the data, showing its *flat* representation and *compositional decoupled* representation and a comparison between the decoupled and the original TOP representations in tree format.

All the translators were professional translators, with native or close to native speaker skills. The overall time spent was 15 to 25 days for each language. Even though we run rigorous quality control, a dataset built by translation is bound to have few errors, such as using words or phrases that are not commonly used in spoken language.

3.2 Data Format

In this dataset, we release two kinds of representations, which we refer to as *flat* representations and *compositional decoupled* representations, that are illustrated in Figure 1 for an English utterance. Most existing annotations for task-oriented dialog systems follow the intent classification and slot tagging paradigm, which is what we refer to as the flat representation. Since our data contains compositional utterances with nested slots with intents within them, flat representations are constructed by only using the top level slots. We include the flat representation so that the data and the discussed modeling techniques are comparable to other task-oriented dialog benchmarks. To ensure the reproducibility of our results, we also release the tokenized version of utterances obtained via our in-house multilingual tokenizer.



German

Utterance: *Richte eine Erinnerung ein*, ***Mike*** ***heute Abend um 19 Uhr*** *zu* ***benachrichtigen***.

Compositional Decoupled Representation: [IN:CREATE_REMINDER [SL:TODO [IN:SEND_MESSAGE [SL:METHOD_MESSAGE benachrichtigen] [SL:RECIPIENT Mike]]] [SL:DATE_TIME heute Abend um 19 Uhr]]

Flat Representation: [IN:CREATE_REMINDER [SL:TODO Mike] [SL:DATE_TIME heute Abend um 19 Uhr] [SL:TODO benachrichtigen]]

Decoupled Representation:

Figure 2: German utterance constructed from the English example of Figure 1. Even though the slot text order changed, we can still easily build a decoupled representation with the same structure.

3.2.1 Compositional Decoupled Representation

Gupta et al. (2018) demonstrate the inability of flat representations to parse complex compositional requests and propose a hierarchical annotation scheme (TOP representation) for semantic parsing, that allows the representation of such nested queries. We further use a representation, called the decoupled representation, that removes all the text from the TOP representation that does not appear in a leaf slot, assuming this text does not contribute to the semantics of the query. Figure 1 highlights the difference between this decoupled representation and the original TOP representation. The decoupled representation makes the semantic representation more flexible and allows long-distance dependencies within the representation. It also makes translation-based data creation approach feasible for different languages despite syntactic differences, as the representation is *decoupled* from the word order of the utterance. For example, in the German translation of the English example as shown in Figure 2, translations of *message* and *Mike* were separated by other words between them. However, it is straight forward to construct a decoupled representation as the representation is not bound by a word-order constraint.

4 Model Architecture

4.1 Joint intent and slot tagging for flat representation

For flat representation, where there is a single top-level intent, the traditional way is to model it as an intent classification and a slot tagging problem. Our baseline model is a bidirectional LSTM intent slot model as described in Liu and Lane (2016); Zhang and Wang (2016) with pre-trained XLU embeddings. Since existing pre-trained XLU embeddings (e.g., MUSE (Lample et al., 2018)) don't provide embedding for Hindi and Thai, we train our own using multiCCA following Ammar et al. (2016). Compared to previous state-of-the-art work on existing multilingual task-oriented parsing datasets (Liu et al., 2020b; Castellucci et al., 2019) which use Multilingual BERT, we use XLM-R (Conneau et al., 2020) since it's shown to outperform Multilingual BERT in cross-lingual performance on a variety of tasks. Specifically we use XLM-R Large in all our experiments. We use the same model architecture as in Chen et al. (2019) and replace BERT encoder with XLM-R encoder.

4.2 Seq-to-seq for hierarchical representation

Even though there are few existing works on cross lingual transfer learning for parsing flat representations, to the best of our knowledge, we are not aware of any other work that studies cross-lingual transfer for parsing complex queries in task-oriented dialog. In this section, we outline our modeling approaches for the compositional decoupled representation discussed in §3.2.1.

Seq-to-seq with Pointer-generator Network Our model adopts an architecture similar to Rongali et al. (2020), where source is the utterance and target is the compositional decoupled representation described in §3.2.1. Given a source utterance, let $[\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n]$ be the encoder hidden states and $[\boldsymbol{d}_1, \boldsymbol{d}_2, ..., \boldsymbol{d}_m]$ be the corresponding decoder hidden states. At decoding time step t, the model can either generate an element from the ontology with generation distribution $\boldsymbol{p}_t^{\mathrm{g}}$, or copy a token from the source sequence with copy distribution $\boldsymbol{p}_t^{\mathrm{c}}$. Generation distribution is computed as:

$$\boldsymbol{p}_t^{\mathrm{g}} = \mathrm{softmax}\left(\mathrm{Linear}_{\mathrm{g}}[\boldsymbol{d}_t]\right)$$

Copy distribution is computed as:

$$\boldsymbol{p}_t^{\mathrm{c}}, \boldsymbol{\omega}_t = \mathrm{MHA}\left(\boldsymbol{e}_1, ..., \boldsymbol{e}_n; \mathrm{Linear}_{\mathrm{c}}[\boldsymbol{d}_t]\right)$$

where MHA stands for Multi-Head Attention (Vaswani et al., 2017) and $\boldsymbol{\omega}_t$ is the attended vector used to compute the weight of copying $\boldsymbol{p}_t^{\mathrm{w}}$:

$$p_t^{\mathrm{w}} = \mathrm{sigmoid}\left(\mathrm{Linear}_{\alpha}\left[\boldsymbol{d}_t; \boldsymbol{\omega}_t\right]\right)$$

The final probability distribution is computed as a mixture of the generation and copy distributions:

$$\boldsymbol{p}_t = p_t^{\mathrm{w}} \cdot \boldsymbol{p}_t^{\mathrm{g}} + (1 - p_t^{\mathrm{w}}) \cdot \boldsymbol{p}_t^{\mathrm{c}}.$$

As a baseline, we use a standard LSTM encoder-decoder architecture with XLU embeddings. We also experiment with various transformer-based state of the art multilingual pre-trained models to improve upon the baseline. We use both pre-trained encoder-only models as well as pre-trained seq-to-seq encoder and decoder models. Here we outline the different models that we experimented with:

- **XLM-R** encoder, pre-trained with masked language model objective in 100 languages. For decoder, we use randomly initialized transformer decoder as in Vaswani et al. (2017).
- **mBART** (Liu et al., 2020a) is pre-trained seq-to-seq model using denoising autoencoder objective on monolingual corpora in 25 languages.
- **mBART on MT**: Machine translation is another common task for pre-training multilingual models. We follow Tang et al. (2020) to further fine-tune mBART on English to 25 languages translation task.
- **CRISS** (Tran et al., 2020) is pre-trained on parallel data in an unsupervised fashion. It iteratively mines parallel data using its own encoder outputs and trains a seq-to-seq model on the parallel data. CRISS has been shown to perform well on sentence retrieval and translation tasks.
- **MARGE** (Lewis et al., 2020a) is learned with an unsupervised multi-lingual multi-document paraphrasing objective. It retrieves a set of related texts in many languages and conditions on them to maximize the likelihood of generating the original text. MARGE has shown to outperform other models on a variety of multilingual benchmarks including document translation and summarization.

5 Experiments

We conduct thorough experiments on the new dataset we describe in in §3. To further demonstrate the effectiveness of our proposed approaches,

---

We provide reproducibility details and all hyperparameters in Appendix A we also run additional experiments on the existing multilingual task-oriented semantic parsing datasets including *Multilingual ATIS* (Upadhyay et al., 2018) and *Multilingual TOP* (Schuster et al., 2019a). Note that both these data sets only include flat representation, while our data set contains hierarchical representations.

5.1 Experimental Settings

For all benchmarks, we have three different evaluation settings:

- IN-LANGUAGE MODELS: We only use target language training data.
- MULTILINGUAL MODELS: We use training data in all available languages and train a single model for multiple languages.
- ZERO-SHOT TARGET LANGUAGE MODELS: We only use English data during training.

Next in each subsection we talk about details of approaches we use in these experiments.

5.1.1 Translate and Align

With zero or few target language annotated examples, *translate-train* is a common approach to augment target language training data. For semantic parsing tasks, besides translation we need alignment to project slot annotations to target language. This process is similar to how we collect our dataset, but using machine translation and alignment methods. For translation, we use our in-house machine translation system. We also tried other publicly available translation APIs and didn't find significant difference in final task performance. For alignment, we experimented with both, using attention weights from translation as in Schuster et al. (2019a) and fastalign (Dyer et al., 2013) and found data generated through fastalign leads to better task performance. Thus we only report results that use fastalign.

5.1.2 Multilingual Training

With the advancement of multilingual pre-trained models, a single model trained on multiple languages has shown to outperform in-language models (Conneau et al., 2020; Hu et al., 2020). As a result, we also experiment with multilingual training on our benchmark, including training jointly on all in-language data and training on English plus translated and aligned data in all other languages for the zero-shot setting. Instead of concatenating data in all languages together as in Conneau et al. (2020), we adopt a multitask training approach where for each batch we sample from one language based on a given sampling ratio so that languages with fewer training data can be upsampled. We found this setting to perform better than mixed-language batches in our experiments.

5.1.3 Distant Supervision in Zero-Shot Setting for Flat Representations

Alignment models are not perfect, especially for low resource languages. To combat the noise and biases introduced in slot label projection, we experiment with another distant supervision approach in the zero-shot setting for learning flat representation models. We first concatenate the English utterance and its corresponding translation (using machine translation) in target language as input and then replace the English slot text with MASK token at random (30% of the time, chosen empirically as a hyper-parameter). With the masked source utterance and the translated utterance as the concatenated input, we train a model to predict the overall intent and slot labels on the original English source. In this way, the MASK token can also attend to its translation counterpart to predict its label and the translated slot text could be distantly supervised by English labeled data.

6 Results and Discussions

6.1 Results on MTOP

Flat Representation Results Table. 2 shows the result on our MTOP dataset for all languages, using the flat representation. For both in-language and multilingual settings, XLM-R based models significantly outperform the BiLSTM models using XLU. We also observe that multilingual models outperform in-language models. Interestingly, for Hindi and Thai (both non-European languages), the improvements from multilingual training are considerably higher for XLM-R as compared to XLU BiLSTM. This observation highlights the remarkable cross-lingual transferability of the pre-trained XLM-R representations where fine-tuning on syntactically different languages also improves target language performance.

For zero-shot cross-lingual transfer, we restrict ourselves to an XLM-R baseline to explore improvements using translate and align, and the distant supervision techniques as described in 5.1.1 and 5.1.3 respectively. Our results demonstrate that

Model	en	es	fr	de	hi	th	Avg(5 langs)
			(Exact Match Accuracy)				
In-language models (only use target language training data)							
XLU biLSTM	78.2	70.8	68.9	65.1	62.6	68	67.1
XLM-R	85.3	81.6	79.4	76.9	76.8	73.8	77.7
Multilingual models (use training data from multiple languages)							
XLU biLSTM	78.2	73.8	71.5	65.8	63.1	68.7	68.6
XLM-R	86.3	83.6	81.8	79.2	78.9	76.7	80
Zero-shot target language models (only use English training data)							
XLM-R on EN	N/A	69.1	65.4	64	55	43.8	59.5
XLM-R with mask in §5.1.3	N/A	68	69.5	**69.2**	**63.3**	35.3	61.1
XLM-R on EN + translate align §5.1.1	N/A	74.5	**72.6**	64.7	58.3	**56.5**	65.3
XLM-R with mask + translate align	N/A	**74.6**	72.2	65.7	62.5	53.2	**65.6**

Table 2: Results on flat representation for 6 languages. We report exact match accuracy in this table. More metrics including intent accuracy and slot F1 is in Table 5 in Appendix. Notice that average is calculated across 5 languages except English to be comparable to zero-shot results. Best result for zero-shot is in bold. Taking best zero shot setting for each language, average exact match accuracy is 67.2. Note that for zero-shot setting, we only use EN train and eval data without any target language data.

distant supervision is able to considerably improve over the baselines for French, German and Hindi, while there is a small drop for Spanish. In the same setting, performance for Thai significantly degrades compared to the baseline. We suspect this is due to imperfect Thai tokenization that leads to learning noisy implicit alignments through distant supervision. The translate and align approach consistently improves over the baseline for all languages. It also performs better than distant supervision for all languages except German and Hindi. Our hypothesis is that the compounding nature of German inhibits the learning of hard alignment from fastalign. In summary, the XLM-R trained on all the 6 languages significantly outperforms all other models for this task.

In Appendix B, we further report intent accuracy and slot F1 metrics for the flat representation, as these are commonly used metrics in previous benchmarks for intent-slot prediction (Price, 1990; Schuster et al., 2019a).

Compositional Decoupled Representation Table. 3 shows the results on our MTOP dataset using compositional decoupled representation. In all settings, using multilingual pre-trained models significantly outperform the baseline. Surprisingly, mBART doesn't demonstrate strong performance compared to other models with fine-tuning on our task, even though fine-tuning BART on English achieves the best performance on English data. We hypothesize that mBART was under-trained for many languages and did not learn good cross-lingual alignments. In order to prove our hypothesis, we further fine-tune mBART on English to 25 languages translation task. The obtained mBART fine-tuned on translation significantly outperform the original mBART. The performance of CRISS and MARGE are at par with each other and among our best performing models across 5 languages, except Thai. XLM-R with random decoder performs the best on Thai. We believe this is because neither CRISS nor MARGE are pre-trained on Thai, while XLM-R pre-training includes Thai.

Similar to previous observations, multilingual training improves over the monolingual results. With multilingual training, XLM-R and CRISS are the best performing models for every language. Since XLM-R uses a randomly initialized decoder, it makes intuitive sense that such a decoder is better trained with multilingual training and thus obtains higher gains from more training data. Interestingly, mBART performance also improves a lot, which is another evidence that it was originally under-trained, as discussed in the previous paragraph. In the zero-shot setting, using the models fine-tuned on English does not perform well. In fact Thai zero shot using CRISS gives a 0 exact match accuracy, as the model was not pre-trained on any Thai data. Both XLM-R and CRISS show significant improve-

Model	**en**	**es**	**fr** (Exact Match Accuracy)	**de**	**hi**	**th**	**Avg(5 langs)**
In-language models (only use target language training data)							
XLU biLSTM	77.8	66.5	65.6	61.5	61.5	62.8	63.6
XLM-R encoder + random decoder	83.9	76.9	74.7	71.2	70.2	**71.2**	72.8
mBART	81.8	75.8	68.1	69.1	67.6	61.2	68.4
mBART on MT	**84.3**	77.2	74.4	70.1	69.2	66.9	71.6
CRISS	84.2	**78**	**75.5**	**72.2**	**73**	68.8	**73.5**
MARGE	84	77.7	75.4	71.5	70.8	70.8	73.2
Multilingual models (use training data from multiple languages)							
XLM-R encoder + random decoder	83.6	**79.8**	**78**	74	74	**73.4**	**75.8**
mBART	83	78.9	76	72.9	72.8	68.8	73.9
CRISS	**84.1**	79.1	77.7	**74.4**	**74.7**	71.3	75.4
Zero-shot target language models (only use English training data)							
XLM-R on EN	N/A	50.3	43.9	42.3	30.9	26.7	38.8
XLM-R on EN + translate align	N/A	71.9	70.3	62.4	63	**60**	**65.5**
CRISS on EN	N/A	48.6	46.6	36.1	31.2	0	32.5
CRISS on EN + translate align	N/A	**73.3**	**71.7**	**62.8**	**63.2**	53	64.8

Table 3: Results on compositional decoupled representation for 6 languages. Metric is exact match accuracy. Average is calculated across 5 languages except English. Best result for each setting is in bold. For reference, exact match accuracy for BART model in-language training for en is 84.6.

Model	**Multilingual ATIS** **hi**	 **tr**	**Multilingual TOP** **es**	 **th**
In-language models (only use target language training data)				
Original paper	-/-/74.6	-/-/75.5	74.8/96.6/83.0	84.8/96.6/90.6
XLM-R	53.6/80.6/84.4	52.6/90.0/80.4	84.3/98.9/90.2	90.6/97.4/95
Multilingual models (use training data from multiple languages)				
original paper (bilingual)	-/-/80.6	-/-/78.9	76.0/97.5/83.4	86.1/96.9/91.5
XLM-R ALL	62.3/85.9/87.8	65.7/92.7/86.5	83.9/99.1/90	91.2/97.7/95.4
Zero-shot target language models (only use English training data)				
Original paper	N/A	N/A	55/85.4/72.9	**45.6/95.9/55.4**
MBERT MLT	N/A	N/A	-/87.9/73.9	-/73.46/27.1
XLM-R on EN	40.3/80.2/76.2	15.7/78/51.8	**79.9/97.7/84.2**	35/90.4/46
XLM-R with mask	49.4/85.3/84.2	19.7/79.7/60.6	76.9/98.1/85	23.5/95.9/30.2
XLM-R EN + translate align	53.2/85.3/84.2	**49.7/91.3/80.2**	66.5/98.2/75.8	43.4/97.3/52.8
XLM-R mask + translate align	**55.3/85.8/84.7**	46.4/89.7/79.5	73.2/98/83	41.2/96.9/52.8

Table 4: Results on Multilingual ATIS and Multilingual TOP, metrics are exact match accuracy / intent accuracy / slot F1 respectively. For zero-shot, first line is from original dataset paper. Best result for zero-shot is in bold.

ments when they utilized the machine translated and aligned data.

6.2 Results on Existing Benchmarks

Table. 4 shows results on two previously released multilingual datasets: Multilingual ATIS and Multilingual TOP. Similar to our findings in 6.1, XLM-R based models significantly outperform the best results reported by the original papers and sets a new state-of-the-art on these benchmarks. Also, multilingual models trained on all available languages further improve the result.

For Multilingual ATIS, in the zero-shot setting, our distant supervised masking strategy shows considerable gains compared to direct transfer using English. Using translate and aligned data also helps in improving the results significantly. When multitask trained together with masked data, it achieves the best zero-shot performance on Hindi. For both languages (Hindi and Turkish) this comes very close to the performance using target language training data.

For multilingual TOP, direct transfer proves to be effective for Spanish, direct transfer from English overall yield better result than what's reported in Mixed-Language Training (MLT) with MBERT (Liu et al., 2020b). While masking and translating generated data degrade its performance. Based on our error analysis, we find that tokenization mismatch, derived from translation data, causes such performance drop due to errors in slot text boundaries. For Thai, all our translation-based techniques perform worse than translate-train results from original paper. We attribute this primarily to the tokenization difference between our translated data and original test data. Unlike Spanish, Thai is much more sensitive to tokenization as it rarely uses whitespace.

7 Conclusion

In this paper, we release a new multilingual task-oriented semantic parsing dataset called **MTOP** that covers 6 languages, including both flat and compositional representations. We develop strong and comprehensive benchmarks for both representations using state-of-the-art multilingual pre-trained models in both zero-shot and with target language settings. We hope this dataset along with proposed methods benefit the research community in scaling task-oriented dialog systems to more languages effectively and efficiently.

References

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*.

Giuseppe Castellucci, Valentina Bellomaria, Andrea Favalli, and Raniero Romagnoli. 2019. Multilingual intent detection and slot filling in a joint bert-based model. *arXiv preprint arXiv:1907.02884*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

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

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

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

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 644–648. The Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *HLT-NAACL*, pages 199–209. The Association for Computational Linguistics.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *arXiv preprint arXiv:2003.11080*.

Pavel Izmailov, Dmitry Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Conference on Uncertainty in Artificial Intelligence (UAI 2018)*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *ICLR (Poster)*. OpenReview.net.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. *arXiv preprint arXiv:2006.15020*.

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

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 685–689. ISCA.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *CoRR*, abs/2001.08210.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020b. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8433–8440. AAAI Press.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *INTERSPEECH*, pages 3771–3775.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *ACL (1)*, pages 4996–5001. Association for Computational Linguistics.

P. J. Price. 1990. Evaluation of spoken language systems: the atis domain. In *HLT*. Morgan Kaufmann.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. *arXiv preprint arXiv:2001.11458*.

Sebastian Ruder, Ivan Vulic, and Anders Sogaard. 2017. A survey of cross-lingual word embedding models. Cite arxiv:1706.04902.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019a. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019b. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

Chau Tran, Yuqing Tang, Xian Li, and Jiatao Gu. 2020. Cross-lingual retrieval for iterative self-supervised training. *arXiv preprint arXiv:2006.09526*.

Shyam Upadhyay, Manaal Faruqui, Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck. 2018. (almost) zero-shot cross-lingual spoken language understanding. In *ICASSP*, pages 6034–6038. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jason D. Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *D&D*, 7(3):4–33.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2993–2999. IJCAI/AAAI Press.

Su Zhu and Kai Yu. 2017. Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding. In *ICASSP*, pages 5675–5679. IEEE.

A Training Details

Settings for MTOP results in Table. 2 For fine-tuning XLM-R, we use the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 1e-6$ and batch size of 16. We fine-tune for 20 epochs and search over learning rates $\in \{1, 2, 3\}e-5$ on dev set. All XLM-R models were run on single 32GB V100 Nvidia GPU.

For the XLU models in Table. 2, we use 300 dim XLU embeddings and feed them to a 2-layer 200 dim BiLSTM. The intent classification head contains an attention pooling layer as described in Lin et al. (2017) with with attention dim 128 followed by a 200 dim linear projection before the softmax. The slot tagging head also contains a 200 dim linear layer followed by a CRF decoder. We use the we use the Adam optimizer with the same settings as above and a batch size of 32 for 40 epochs. The learning rate and BiLSTM dropouts are picked via a param sweep over the dev set.

Settings for MTOP results in Table. 3 For training seq-2-seq models, we use stochastic weight averaging (Izmailov et al., 2018) with Lamb optimizer (You et al., 2019) and exponential learning rate decay for all models. For fine-tuning pre-trained models: we use batch size of 16 for all models except Marge, we use batch size 4 for Marge since we were not able to fit larger batch size into 32GB memory; We finetune for 50 epochs and again search over learning rates on dev set.

For copy pointer We use 1 layer multihead attention(MHA) with 4 attention heads to get copy distribution. For seq-2-seq model with XLM-R encoder, the decoder is a randomly initialized 3-layer transformer, with hidden size 1024 and 8 attention heads. XLM-R encoder (24 layers) is larger than mBART/CRISS/MARGE encoder (12 layers) so we were not able to fit a larger decoder into GPU memory.

For the XLU models specifically we use a 2-layer BiLSTM encoder with a hidden dimension of 256. For the decoder, we use a 2-layer LSTM with 256 dimension and a single attention head. Similar to the flat models, learning rate and LSTM dropouts are picked via a param sweep over the dev set.

Settings for other benchmark results in Table. 4 We use the same setting as described for Table. 2 except for multilingual ATIS which doesn't have dev set, we just use the checkpoint after a fixed number of epochs.

B More Results

We report additional metrics for our experiments in this section. Table. 5 contains the intent accuracy and slot F1 metrics of models for flat representation.

Model	**en**	**es**	**fr**	**de**	**hi**	**th**
			(Intent Accuracy / Slot F1)			
In-language models (only use target language training data)						
XLU biLSTM	94.0/88.6	90.1/83.0	89.6/81.8	88.8/81.4	85.9/79.6	91.2/80.4
XLM-R	96.7/92.8	95.2/89.9	94.8/88.3	95.7/88.0	94.4/87.5	93.4/85.4
Multilingual models (use training data from multiple languages)						
XLU biLSTM	94.6/88.4	91.3/84.6	91.3/83.0	90.3/81.2	87.6/78.9	91.9/80.5
XLM-R	97.1/93.2	96.6/90.8	96.3/89.4	96.7/88.8	95.4/88.4	95.1/86.3
Zero-shot target language models (only use English training data)						
XLM-R on EN	N/A	93.5/81.7	90.7/81.6	91.2/78.7	88.4/71.8	88.0/63.3
XLM-R with mask in §5.1.3	N/A	94.7/81.0	93.9/82.0	**94.0/81.8**	**94.1/77.3**	92.0/56.4
XLM-R on EN + translate align §5.1.1	N/A	96.2/84.6	**95.4/82.7**	96.1/78.9	94.7/72.7	**92.7/70.0**
XLM-R with mask + translate align	N/A	**96.3/84.8**	95.1/82.5	94.8/80.0	94.2/76.5	92.1/65.6

Table 5: Intent Accuracy / Slot F1 for models in Table 2.

Adapting Event Extractors to Medical Data: Bridging the Covariate Shift

Aakanksha Naik[1] **Jill Lehman**[2] **Carolyn Rosé**[1]
[1]Language Technologies Institute, Carnegie Mellon University
[2]Human-Computer Interaction Institute, Carnegie Mellon University
`{anaik,jfl,cprose}@andrew.cmu.edu`

Abstract

We tackle the task of adapting event extractors to new domains without labeled data, by aligning the marginal distributions of source and target domains. As a testbed, we create two new event extraction datasets using English texts from two medical domains: (i) clinical notes, and (ii) doctor-patient conversations. We test the efficacy of three marginal alignment techniques: (i) adversarial domain adaptation (ADA), (ii) domain adaptive fine-tuning (DAFT), and (iii) a new instance weighting technique based on language model likelihood scores (LIW). LIW and DAFT improve over a no-transfer BERT baseline on both domains, but ADA only improves on notes. Deeper analysis of performance under different types of shifts (e.g., lexical shift, semantic shift) explains some of the variations among models. Our best-performing models reach F1 scores of 70.0 and 72.9 on notes and conversations respectively, using *no* labeled target data.

1 Introduction

Events are an important phenomenon in the field of computational semantics. They offer an intuitive mechanism for constructing structured representations of text, which can be used for downstream tasks such as question answering and summarization. Events also embody a crucial function of language: the ability to report happenings. Narratives from many diverse domains (e.g., news articles, literary texts, clinical notes) use events as basic building blocks. These characteristics make event extraction a key sub-task of interest for text understanding pipelines in multiple domains. Despite its importance, building high-performing and generalizable systems for event extraction has remained an elusive goal. One of the major hurdles is that the notion of what counts as an *important event* is usually task-specific or domain-specific (sometimes both). For example, to build a system that can track a patient's disease progression from clinical notes, event extractors only need to focus on extracting medical events relevant to that illness. This task/domain specificity has encouraged prior work to focus on specific event types (Grishman and Sundheim, 1996; Doddington et al., 2004; Kim et al., 2008) or domains (Pustejovsky et al., 2003b; Sims et al., 2019). Owing to this narrow focus, supervised event extractors often fail to adapt to new domains or event types (Keith et al., 2017). Unsupervised event extractors that use syntactic rule-based modules (Saurí et al., 2005; Chambers et al., 2014), conversely, have a tendency to over-generate by labeling most verbs and nouns as events.

In this work, we try to achieve a balance between these extremes by *adapting* event extractors using unsupervised domain adaptation techniques. We also study the behavior of these techniques under various types of linguistic shifts (e.g., lexical shift, semantic shift) to gain insight into differences among them. Exploring adaptability under no (or little) supervision is crucial, since sourcing annotated data for new domains, especially medical texts, can be expensive and time-consuming. Following prior work, we formulate event extraction as the task of labeling **triggers**, i.e., words which instantiate an event (Linguistic Data Consortium, 2005). For example, in the sentence "She was diagnosed with cancer," *diagnosed* and *cancer* are triggers, referring to "diagnosis" and "illness" events respectively. Throughout our work, we model event trigger labeling as token-level classification.

To test adaptability, we create new event extraction test sets using English texts from two diverse medical domains: (i) clinical notes, and (ii) doctor-patient conversations. We develop comprehensive event annotation guidelines, based on TimeML (Pustejovsky et al., 2003a) and Thyme-TimeML (Styler IV et al., 2014) (§3), and use them to annotate 45 documents from each domain. As a baseline,

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2963–2975
April 19 - 23, 2021.

we train a BERT-based event extraction model on English news articles from TimeBank (Pustejovsky et al., 2003b), which is labeled using TimeML, and test its performance on our datasets. To improve this out-of-domain baseline performance, we tackle the problem of covariate shift, i.e., differences between marginal distributions of source (news) and target domains (notes or conversations). We experiment with three marginal alignment techniques: (i) adversarial domain adaptation (ADA) (Ganin and Lempitsky, 2015), (ii) domain-adaptive fine-tuning (DAFT) (Han and Eisenstein, 2019), and (iii) a new instance weighting scheme using language model likelihood scores (LIW).

Our results show that DAFT and LIW improve over BERT on both domains, whereas ADA only improves on notes. Across domains, ADA and DAFT perform best on notes and conversations respectively. To probe why some techniques are better at addressing certain source-target domain pairs, we analyze model performance on various types of covariate shifts (e.g., lexical shift, semantic shift). Our analysis uncovers interesting patterns such as varying ability of models to leverage subword morphology to generalize to technical terms, and LIW's performance improvement on long-term state events (e.g., chronic illnesses). Our best models achieve F1 scores of 70.0 and 72.9 on notes and conversations respectively with *no* training data.[1]

2 Related Work

2.1 Event Extraction

Most prior event extraction work has focused on news articles, resulting in the development of several datasets (Onyshkevych et al., 1993; Grishman and Sundheim, 1996; Pustejovsky et al., 2003b; Doddington et al., 2004; Lee et al., 2012; Cybulska and Vossen, 2014; Mitamura et al., 2016). Recently, event extraction has also been explored in other domains such as biology (Wattarujeekrit et al., 2004; Kim et al., 2008, 2009; Berant et al., 2014), Wikipedia articles (Araki and Mitamura, 2018), social media data (Ritter et al., 2012; Li et al., 2014; Jain et al., 2016) and literary novels (Sims et al., 2019). Aside from data domain, event extraction paradigms (both datasets and tools) differ along three major axes: (i) event extraction granularity, (ii) event representation, and (iii) event categorization (ontology). We briefly describe these axes to contextualize our choice of event paradigm.

Event extraction granularity divides extraction paradigms into two types: (i) document-level paradigms that assume that a piece of text refers to a single event (Grishman and Sundheim, 1996), and (ii) sentence-level paradigms that assume that a single sentence describes one or more events. Event representation also divides extraction paradigms into two types: (i) span-based paradigms that represent events by marking text spans that refer to events, called **triggers** or **nuggets** (Linguistic Data Consortium, 2005; Mitamura et al., 2015; O'Gorman et al., 2016), and (ii) structured paradigms that represent events by marking text spans and adding additional arguments (e.g., participants, location etc.) to create a structured template (Grishman and Sundheim, 1996). Event categorization divides extraction paradigms into: (i) ontology-driven paradigms that are limited to specific event types (Grishman and Sundheim, 1996; Doddington et al., 2004), and (ii) ontology-free paradigms that do not place type restrictions (Pustejovsky et al., 2003b; Araki and Mitamura, 2018).

We use a sentence-level, span-based, ontology-free event extraction paradigm. Sentence-level extraction suits our domains of interest since notes and conversations tend to discuss multiple events. Span-based and ontology-free extraction allows us to develop adaptable coding guidelines since event arguments and types are usually domain-specific or task-specific. This adaptability sets our work apart from other prior work on medical event extraction such as adverse drug event extraction (Nikfarjam et al., 2015; Sarker and Gonzalez, 2015; Cocos et al., 2017; Henry et al., 2020) and personal event extraction from online support groups (Wen et al., 2013; Naik et al., 2017), which focus on specific event types. Our guidelines draw heavily from the Thyme-TimeML guidelines (Styler IV et al., 2014) used by the Clinical TempEval challenges on event ordering in clinical notes (Bethard et al., 2015, 2016, 2017),[2] but also cover event extraction in a novel domain: doctor-patient conversations.

2.2 Unsupervised Domain Adaptation

Unsupervised domain adaptation is the task of transferring a model from a source domain to a target domain, using only unlabeled data from the target domain, by aligning source and target distri-

---

[1] Annotated clinical notes and all code associated with this work can be found at: `https://github.com/aakanksha19/MedicalEventExtraction`.

[2] We provide a detailed comparison with this work in §3.1.

butions. Early approaches such as structural correspondence learning (SCL) (Blitzer et al., 2006, 2007) tried to solve this by mapping source and target examples into a shared **pivot feature** space, where pivot features are selected to be features that behave the same way for discriminative learning in both domains (e.g., sentiment terms such as *amazing* and *great* show similar behavior for sentiment analysis across domains). With advances in neural representation learning, autoencoder-based methods (Glorot et al., 2011; Chen et al., 2014), neural SCL (Ziser and Reichart, 2017), adversarial domain adaptation (Ganin and Lempitsky, 2015; Ganin et al., 2016) and LM fine-tuning methods (Han and Eisenstein, 2019; Gururangan et al., 2020) have shown success in learning a shared space in which source and target domains are aligned. We propose a new method (LIW) which relies on instance weighting via language model likelihood, and contrast it with adversarial domain adaptation (ADA) and domain adaptive fine-tuning (DAFT). These two techniques have shown promise on sequence labeling tasks (Gui et al., 2017; Han and Eisenstein, 2019; Naik and Rosé, 2020), and offer an interesting contrast between approaches that jointly perform alignment and task training (ADA) and approaches that perform these steps sequentially (DAFT). Comparing all three techniques also provides us the opportunity to study which methods adapt better to different kinds of shifts between source and target domains (e.g., shifts in vocabulary, syntax, etc.).

3 Dataset Creation

To test adaptability of event extraction models, we create a testbed using data from two domains:

1. Clinical Notes: Clinical notes are records documenting physician observations from their interactions with patients. They usually detail various aspects of a patient's care such as present illness, symptoms, medical history, treatments, and test results. They share a thematic structure, though particular specialties (e.g., cardiology) and institutions often incorporate their own modifications. We collected a set of 4999 de-identified clinical notes from 40 specialties, by scraping mtsamples.[3] The notes are reference samples provided by various users, with names and dates edited for confidentiality. They are freely available to print, share, link and distribute, as per website policy. Average length of a clinical note is 652 tokens.

2. Doctor-Patient Conversations: This data contains human-transcribed, de-identified conversations recorded during physician-patient visits. The conversations often follow a similar schema, with patients describing their symptoms, doctors inquiring about ongoing treatments, and then suggesting potential follow-up treatments/tests. We use a proprietary database of 63,540 conversations covering 53 specialties, collected by Abridge AI Inc. Physicians across a variety of specialties are contracted to record natural in-office conversations with their patients who agree to participate in the research by providing verbal and written consent. Recordings are made on a digital recording device or a smartphone application and are uploaded to a secure server where they are scrubbed of all identifiable information, in accordance with the Health Insurance Portability and Accountability Act of 1996 (HIPAA) privacy rule. De-identified recordings are transcribed and stored in a database, which currently contains over 100,000 recordings dating from 2006 to 2017. Average conversation transcript length is 2309 tokens.

These domains exhibit different types of linguistic shifts from the source (news). While both domains exhibit a shift in vocabulary, it is more pronounced in clinical notes since they are written by doctors (experts) who use highly technical terms. Conversely, shifts in syntax are more pronounced in conversations due to the prevalence of repetition, back-channeling, interruptions etc. Semantic shifts are more pronounced in conversations since they contain a higher proportion of hypothetical statements (e.g., when doctors ask questions, make requests or "think out loud") than both notes and news articles which tend to serve as records of actual events. To better evaluate model performance on linguistic shifts, we control for topical variation across domains by limiting our focus to 3 specialties: Cardiovascular/Pulmonary (Cardio), Obstetrics/Gynaecology (Obgyn) and Hematology/Oncology (Onco). These specialties are well-represented in both notes and conversations, and cover events with a variety of temporalities ranging from intervals with fixed duration (e.g., pregnancy), to intervals with indeterminable endpoints (e.g., long-term cardiac failure). Table 1 gives an overview of the number of notes and conversations in each specialty.

---

[3] https://www.mtsamples.com/

Specialty	#Notes	#Convos
Cardio	372	4876
Obgyn	160	1784
Onco	90	7177

Table 1: Domain-wise raw data statistics for chosen medical specialties

3.1 Developing Event Annotation Guidelines

We develop a set of coding guidelines for the task of annotating event triggers in documents from these two domains. Our coding guidelines build upon TimeML (Pustejovsky et al., 2003a), a rich specification language for annotation of events and temporal expressions in text,[4] and Thyme-TimeML (Styler IV et al., 2014), a variant of TimeML developed for clinical notes. We start with these guidelines because they use a syntax-driven domain-agnostic definition of events, allowing for an adaptable annotation scheme. In TimeML, the term *event* refers to situations that *happen* or *occur*, or circumstances in which something *obtains* or *holds true*. This is a broad definition, consistent with Bach's definition of **eventualities** (Bach, 1986), and the idea of **fluents** (McCarthy, 2002). Events can be expressed in text by means of tensed or untensed verbs, nominalizations, adjectives, predicative clauses or prepositional phrases. TimeML describes rules to annotate events in all these syntactic categories. Styler IV et al. (2014) adapted these rules for clinical notes. They focused on the THYME corpus of 1254 de-identified notes from the Mayo Clinic, representing two fields in oncology: brain cancer and colon cancer. As a first step, we annotate one document from each of our domains following TimeML and Thyme-TimeML rules. During this phase, we identify cases where it is reasonable to deviate from these guidelines.

Deviations from TimeML: Our guidelines[5] differ from TimeML in their treatment of two categories:

1. Activity patterns: Activity patterns are events that are neither pure generics[6], nor single events clearly positioned in time. For example, consider the sentence "I *take* my blood pressure regularly." The event *take* is not grounded in time. It is also not a pure generic event as it is definitely associated with the speaker. Such events are *not* annotated in TimeML. However, in our data, these activity patterns occur frequently in crucial contexts such as taking medications, following lifestyle changes suggested by doctors, measuring vital signs, etc.

2. Long-term states: Because TimeML was geared towards the task of temporal ordering, it strictly restricted annotation of stative events to the following types: (i) states associated with a temporal expression, (ii) states undergoing a change within the document, (iii) states introduced by other events, since those can offer temporal cues, and (iv) states associated with the document creation time. However, many stative events in our data don't fit within these strict parameters, but are nevertheless important. The most crucial category is states associated with long-term ongoing illnesses (e.g., "The patient has a long history of *COPD*").

These event categories are not specific to medical domains only. For example, long-term state events might be salient when extracting personal events from biographies.[7] Similarly activity patterns might be salient when extracting events from scientific procedure manuals.[8] Considering these scenarios, we add rules to extract these two categories of events. We also expand syntactic rules to cover constructions unique to doctor-patient conversations such as repetition, especially for instructions, and hypothetical event annotation in utterances when doctors are "thinking out loud".

Deviations from Thyme-TimeML: Our guidelines differ from Thyme-TimeML in their treatment of two categories:

1. Generic events: Thyme-TimeML annotates generic events in sections documenting discussion of risks, plans and alternative strategies. They do so because adding these events to a patient's clinical timeline could be important from a legal perspective, as they help to establish informed consent and knowledge of risk. We do not annotate pure generics, because we do not perceive any domain-agnostic utility in annotating them. Note that we annotate verbs of discussion and comprehension which are not generics, so we do not fully ignore events associated with patient consent. For exam-

---

[4] The complete TimeML coding manual is available here: https://catalog.ldc.upenn.edu/docs/LDC2006T08/timeml_annguide_1.2.1.pdf

[5] Our complete coding manual, including example annotations, is available at:https://github.com/aakanksha19/MedicalEventExtraction.

[6] Pure generics are events which discuss illnesses/treatments in general, and are not associated with a specific person and time. For example, "there is a *benefit* to systemic adjuvant *chemotherapy*."

[7] e.g., "Bill Gates is currently *employed* full-time at the Bill and Melinda Gates Foundation."

[8] "*Repeat* step 5 daily, over a period of 30 days."

Domain	Entity κ	Event κ
Notes	0.9117	0.8652
Convos	0.8634	0.8327

Table 2: Inter-annotator agreement on entity and event annotation tasks in both domains, measured using chance-corrected Cohen's κ

ple, in the sentence "She repeated the potential side effects back to me," *repeated* is annotated, but *effects* is not. Thyme-TimeML would annotate both.

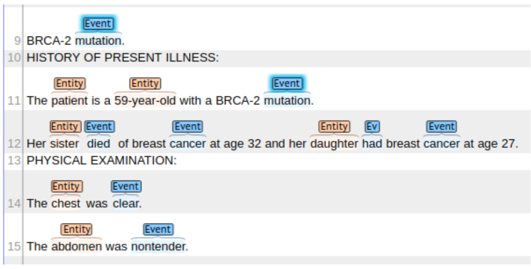
2. Entities as events: Thyme-TimeML treats some entities and non-events as events in clinical language. Two categories see this shift in semantic interpretation: (i) Medications, and (ii) Disorders. Both categories contribute significant information to a patient's timeline, and so they are treated as events. Since we are not specifically focused on timeline construction, we do not follow the same reasoning. In particular, medications are not treated as events, while disorders may be treated as events as long as they fit the TimeML definition. To ensure that we do not discard potentially crucial information, we incorporate an additional step in which we annotate entities such as medications, body parts, abnormalities (e.g., rash), etc.

3.2 Annotation Process

After incorporating our modifications, we test our guidelines by having two expert annotators annotate one document from each domain. We see high inter-annotator agreement (measured by chance-corrected Cohen's κ) on entity and event annotation, in both domains. Table 2 presents the agreement scores. To create our final datasets, we sample 45 documents from each domain (15 from each specialty). Each document is annotated by one expert. Annotation is carried out using the BRAT stand-off markup interface (Stenetorp et al., 2012). Figure 1 shows a sample clinical note annotated with events and entities. Table 3 gives a brief overview of statistics for our datasets, in comparison with TimeBank (news articles) (Pustejovsky et al., 2003b).

4 Methods for Marginal Alignment

To adapt event extraction models with no training data, we tackle the problem of covariate shift, which arises when the marginal distribution (or input distribution) $P(X)$ changes between train and test data. Directly applying a supervised model trained on the training set, to the test set might not perform well due to the gap between training and test distributions. We experiment with several techniques to align the training and test distributions, so that the supervised model transfers better to test data. The techniques can be divided into two types based on the kind of supervision used during alignment: (i) task-guided alignment techniques, and (ii) task-agnostic alignment techniques.



BRCA-2 mutation.
HISTORY OF PRESENT ILLNESS:
The patient is a 59-year-old with a BRCA-2 mutation.
Her sister died of breast cancer at age 32 and her daughter had breast cancer at age 27.
PHYSICAL EXAMINATION:
The chest was clear.
The abdomen was nontender.

Figure 1: Sample clinical note with entity and event annotation

Statistic	News	Notes	Convos
#Files	54	45	45
#Tokens	18,263	28,935	76,711
#Events	1986	4781	7064
Event Density	10.88%	16.52%	9.21%
Vocab Size	3978	4303	3505
Event Vocab	1015	1588	1472

Table 3: Dataset statistics. Note that the statistics for TimeBank (News) are computed over the test set for fair comparison with our datasets, which are test-only.

4.1 Task-Guided Alignment Techniques

These techniques jointly optimize for two tasks: (i) aligning training and test distributions, and (ii) training an event extraction model. Since the alignment process receives supervision from task training, we refer to these techniques as task-guided alignment techniques. Under this category, we experiment with adversarial domain adaptation.

Adversarial Domain Adaptation: Adversarial domain adaptation was proposed by Ganin and Lempitsky (2015), who showed its efficacy on sentiment analysis. Recently, Naik and Rosé (2020) showed its utility in transferring event extraction models between two domains: news and literature. The adversarial domain adaptation framework for event extraction contains three components: (i) representation learner (R) which generates token-level representations for a sequence, (ii) event classifier which identifies events (E), and (iii) domain predictor (D) which predicts the domain for the sequence. The key idea is to train R to generate representations which *are predictive* for event identification but *not predictive* for domain prediction, making it more domain-invariant. This aligns training and test distributions by finding a shared feature space in which training and test samples are not distinguishable, while making sure that the feature space is useful for event extraction. The technique relies on an alternating optimization procedure. The first step optimizes D on the domain prediction task, while the second step optimizes both R and E on event identification while subtracting domain prediction loss. For complete mathematical details, we refer the interested reader to Naik and Rosé (2020).

4.2 Task-Agnostic Alignment Techniques

These techniques perform training/test distribution alignment and event extraction training sequentially instead of jointly optimizing them. The alignment process does not receive supervision from task training, so these techniques are task-agnostic. We experiment with the following techniques:

Domain Adaptive Fine-tuning: Domain adaptive fine-tuning has been proposed as an effective technique for unsupervised adaption of sequence labeling models to challenging domains such as Early Modern English and social media (Han and Eisenstein, 2019). This procedure works as follows:

1. Create a large dataset containing equal proportions of sentences from source and target domains. Fine-tune contextualized embeddings using a masked language modeling objective.
2. Using fine-tuned embeddings, train an event extraction model on labeled source data.

In addition to this setup, we experiment with a variant of this procedure, which uses a syntactic objective function. This variant fine-tunes embeddings on the POS tagging task in step 1. The motivation behind this variant is two-fold. First, we observe that event annotation is heavily syntax-driven, allowing delexicalized models (i.e., models using POS tags instead of words) to achieve high performance (§5.2). This indicates that infusing more syntactic awareness into embeddings might help performance on the task. Second, syntax might offer an additional basis for generalization, since sentences that look very different lexically, might follow similar syntactic structures. Intuitively, this variant is similar to syntactic relexicalization which has shown success in cross-lingual dependency parsing (Duong et al., 2015).

Likelihood-based Instance Weighting: We develop a new instance weighting procedure which uses likelihood scores computed by a language model. Instance selection and instance weighting strategies have frequently been used to perform domain adaptation by correcting for distributional differences (Jiang and Zhai, 2007; Foster et al., 2010; Axelrod et al., 2011; Wang et al., 2017). The main premise is that some samples from out-of-domain data and in-domain data often share some characteristics. Training only on these samples (pruning), or biasing training to focus more on these samples (weighting) can produce models that perform better on out-of-domain data. Motivated by this, our instance weighting procedure works as follows.

Let $S_t = w_1 w_2 ... w_n$ be a sentence from the in-domain training set. Let O be a language model trained on raw text from the target domain. We first compute the likelihood of sentence S_t under O as $\mathbb{L}_t = P_O(w_1)\Pi_{i=2}^{n} P_O(w_i|w_1...w_{i-1})$, where P_O indicates probability under model O. Then we compute a weight for S_t as follows:

$$\alpha_{S_t} = \frac{\mathbb{L}_t}{\sum_{i=1}^{|N|} \mathbb{L}_i} * |N| \qquad (1)$$

where $|N|$ is the size of in-domain training set. This metric gives a higher weight to in-domain sentences that are *more likely* under the target domain language model, up-weighting instances that share more characteristics with target domain sentences. The alpha values are used to weight the loss function, thus biasing the training procedure.

5 Experimental Setup

5.1 Model Details

The goal of our evaluation is to identify which alignment technique works best for each domain, as well as analyze whether there are specific kinds of source-target shifts that some techniques are better equipped to handle. We choose a strong BERT-based baseline model with no transfer, and evaluate the performance of each alignment technique when applied to this baseline.

VERB: Baseline labeling all verbs as events.

DELEX: Fully-delexicalized baseline using POS tag embeddings as features, followed by an MLP.

BERT: Single-layer BiLSTM over BERT embeddings (Devlin et al., 2019), followed by an MLP, similar to the best-performing model on LitBank

Model	**In-Domain**			**Out-of-Domain**		
	P	**R**	**F1**	**P**	**R**	**F1**
VERB	58.8	66.5	62.5	49.4	41.4	45.0
DELEX	75.0	66.3	70.4	74.4	42.2	53.8
BERT	80.6	86.0	83.2	85.7	55.9	67.6
CBERT	79.2	83.3	81.2	85.8	52.9	65.4
BERT-ADA	81.2	86.3	83.7	83.2	**60.4**	**70.0**
BERT-LIW	81.9	86.6	84.1	**86.7**	56.0	68.1
BERT-DAFT	79.1	85.9	82.3	83.9	58.6	69.0
BERT-DAFT-SYN	76.9	80.7	78.7	70.7	56.8	63.0

Table 4: Model performance on domain transfer experiments from news to clinical notes.

Model	**In-Domain**			**Out-of-Domain**		
	P	**R**	**F1**	**P**	**R**	**F1**
VERB	58.8	66.5	62.5	44.6	68.1	53.9
DELEX	75.0	66.3	70.4	56.9	64.5	60.4
BERT	80.6	86.0	83.2	**75.0**	63.6	68.9
CBERT	79.2	83.3	81.2	66.5	65.1	65.8
BERT-ADA	81.1	85.9	83.4	74.5	62.2	67.8
BERT-LIW	80.0	87.0	83.4	72.8	67.3	70.0
BERT-DAFT	78.5	84.8	81.5	72.7	**73.1**	**72.9**
BERT-DAFT-SYN	80.0	78.7	79.3	67.6	60.7	63.9

Table 5: Model performance on domain transfer experiments from news to doctor-patient conversations.

(Sims et al., 2019).

CBERT: Similar to BERT, but embeddings are extracted from Clinical-BERT (Alsentzer et al., 2019)

BERT-ADA: BERT trained using adversarial domain adaptation.

BERT-LIW: BERT trained on data weighted by LM likelihood. We train autoregressive language models over 3 million tokens for each domain.

BERT-DAFT: BERT with domain adaptive fine-tuning. For target domains, we use the same text as BERT-LIW, and extract 3 million tokens from CNN/ DailyMail (Hermann et al., 2015) for news.

BERT-DAFT-SYN: BERT with syntactic fine-tuning on the same text as BERT-DAFT, tagged using Stanford CoreNLP (Manning et al., 2014). Complete implementation details are provided in appendix A.

5.2 Results

Tables 4 and 5 show the performance of all models when transferring from news data to clinical notes and doctor-patient conversations respectively. From the tables, we see that DELEX is surprisingly strong out-of-domain. BERT with no transfer performs well out-of-domain, improving by 8.25 F1 points on average over DELEX. C-BERT also performs well out-of-domain, but does worse than BERT. We attribute this to the fact that fine-tuning only on clinical notes does not improve alignment with the source domain (news), providing no basis for models trained on news to adapt better. BERT-ADA shows mixed results, improving over BERT by 2.4 F1 on notes, but dropping by 1.1 F1 on conversations. BERT-LIW and BERT-DAFT improve upon BERT in both domains. BERT-DAFT shows minor performance drops in-domain, due to some degree of catastrophic forgetting. BERT-DAFT-SYN shows performance drops, both in-domain and out-of-domain, in both settings. Unlike syntactic relexicalization work which used non-contextualized embeddings, we use contextualized embeddings, which possess a larger degree of syntactic information, probably reducing the need for syntax-driven training. Another source of errors is POS tagging, since off-the-shelf taggers trained on news will be less accurate on our data. Across domains, the skew between precision and recall is higher on notes, which might stem from the specialized vocabulary dragging down recall.

6 Analysis and Discussion

Tables 4 and 5 provide an indication of model ability to handle covariate shift. However, covariate shift occurs at multiple layers in language (e.g., lexical level, syntactic level, etc.), leading to different dimensions of variation between domains (e.g., topical variation, genre variation, etc.). Looking at overall model performance does not offer insight into whether there are specific shifts that some models are better at addressing. We dig deeper into this question, focusing on two levels of shift: (i) lexical shift, and (ii) semantic (event type) shift.

Variation under lexical shift: We separate model performance on in-vocabulary (IV) and out-of-vocabulary (OOV) tokens. Note that the proportion of events that are OOV is higher in clinical notes (52%) than conversations (20.6%). Tables 6 and 7 present model performance on these token categories. Surprisingly, despite the use of specialized language, OOV performance on clinical notes is higher than conversations for all models except BERT-DAFT. Taking a closer look at the OOV event instances from clinical notes that models identify correctly, we see that a large proportion (54.8%) contain one of three morphological patterns: (i) past tense verbs ending in "-ed", (ii) gerunds ending in "-ing", or (iii) nouns ending in "-tion" or "-sion". These patterns are also common among events in the news domain. For example, past tense verbs often refer to events that have already occurred and gerunds and nouns ending in "-tion" refer to processes. We hypothesize that BERT-based models might be exploiting these morphological regularities to correctly label unseen medical terms (e.g., irrigated, excision, dissected, wheezing, etc.). These patterns are more prevalent in notes (35.6%) than conversations (23.5%), explaining the surprising performance difference.

Variation under semantic shift: To determine whether model performance on OOV tokens depends on event type, we randomly sample ∼500 OOV tokens from each domain and label them for event type. We use the same typology as TimeML (State, I-State, Occurrence, Aspectual, Reporting, Perception, I-Action, None), with additional labels for the event types we introduce (ActivityPattern, LongTermState).[9] We run an ANOVA model with each token per model as an instance (total 5080 instances), noting Event Type, Target (notes/convos), Model (BERT/ADA/LIW/DAFT/DAFT-SYN) and Correctness (1 vs 0). Correctness is the dependent variable, while others are independent variables. We include all pairwise interaction terms and the three way interaction between Event Type, Target and Model. We see a positive main effect of Event Type on Correctness ($p < 0.0001$), indicating that some event types are more difficult. There are two significant two-way interactions, one between Target and Event type ($p < 0.0001$), indicating that difficulty of event types differs across sources, and between Model and Event type ($p < 0.0001$), indicating that which model is better depends on event type. Three way interaction between Model, Event type, and Target is also significant ($p < 0.0001$), indicating that performance differences between models per event type differs between sources.

We interpret differences in performance per event type separately for each source using a Student-t post-hoc analysis to determine which pairwise contrasts are statistically significant. This reveals that in clinical notes, LIW outperforms all models on I-State events (i.e., hypothetical, future or negated states) and LongTermState events, a category never seen in the training data. These improvements might stem from the training algo-

Model	IV F1	OOV F1
BERT	73.5	61.2
BERT-ADA	75.2	65.0
BERT-LIW	73.6	62.6
BERT-DAFT	75.7	62.0
BERT-DAFT-SYN	67.7	58.4

Table 6: Model performance on in-vocabulary (IV) and out-of-vocabulary (OOV) terms from notes.

Model	IV F1	OOV F1
BERT	71.3	57.9
BERT-ADA	70.2	57.6
BERT-LIW	72.0	61.4
BERT-DAFT	74.9	63.6
BERT-DAFT-SYN	65.5	55.5

Table 7: Model performance on in-vocabulary (IV) and out-of-vocabulary (OOV) terms from conversations.

---

[9] Examples provided in appendix B

rithm used by LIW. LIW up-weights instances in news that resemble medical data, which contains a high proportion of these event categories. Therefore, despite being infrequent in news, they get up-weighted, helping LIW identify them better.

7 Conclusion

In this work, we focused on unsupervised adaptation of event extractors to new domains by aligning the marginal distributions of source and target domains. We created two event extraction test sets using English texts from two medical domains: (i) clinical notes, and (ii) doctor-patient conversations, and tested the efficacy of three alignment techniques: (i) adversarial domain adaptation (ADA), (ii) domain adaptive fine-tuning (DAFT), and (iii) a new instance weighting technique based on language model likelihood scores (LIW). None of these models consistently outperformed the others, but a deeper analysis of model performance under different types of shifts (e.g., lexical shift, semantic shift) uncovered interesting variations among models. Our best-performing models attained F1 scores of 70.0 and 72.9 on notes and conversations respectively, using *no* labeled target data. We believe these models define a good starting point and can be further improved using few-shot learning.

Acknowledgements

This work was supported by the University of Pittsburgh Medical Center (UPMC) and Abridge AI Inc through the Center for Machine Learning and Health at Carnegie Mellon University. The authors would like to thank the anonymous reviewers for their helpful feedback on this work.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jun Araki and Teruko Mitamura. 2018. Open-domain event detection using distant supervision. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 878–891, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Emmon Bach. 1986. The algebra of events. *Linguistics and philosophy*, pages 5–16.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Steven Bethard, Leon Derczynski, Guergana Savova, James Pustejovsky, and Marc Verhagen. 2015. SemEval-2015 task 6: Clinical TempEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 806–814, Denver, Colorado. Association for Computational Linguistics.

Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. 2016. SemEval-2016 task 12: Clinical TempEval. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1052–1062, San Diego, California. Association for Computational Linguistics.

Steven Bethard, Guergana Savova, Martha Palmer, and James Pustejovsky. 2017. SemEval-2017 task 12: Clinical TempEval. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 565–572, Vancouver, Canada. Association for Computational Linguistics.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia. Association for Computational Linguistics.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering with a multi-pass architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Minmin Chen, Kilian Q. Weinberger, Fei Sha, and Yoshua Bengio. 2014. Marginalized denoising autoencoders for nonlinear representations. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26*

June 2014, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1476–1484. JMLR.org.

Anne Cocos, Alexander G. Fiks, and Aaron J. Masino. 2017. Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in twitter posts. *J. Am. Medical Informatics Assoc.*, 24(4):813–821.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland. European Language Resources Association (ELRA).

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

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Cross-lingual transfer for unsupervised dependency parsing without parallel data. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 113–122, Beijing, China. Association for Computational Linguistics.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459, Cambridge, MA. Association for Computational Linguistics.

Yaroslav Ganin and Victor S. Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1180–1189. JMLR.org.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 513–520. Omnipress.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Tao Gui, Qi Zhang, Haoran Huang, Minlong Peng, and Xuanjing Huang. 2017. Part-of-speech tagging for twitter with adversarial neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2411–2420, Copenhagen, Denmark. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Sam Henry, Kevin Buchan, Michele Filannino, Amber Stubbs, and Özlem Uzuner. 2020. 2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records. *J. Am. Medical Informatics Assoc.*, 27(1):3–12.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Ajit Jain, Girish Kasiviswanathan, and Ruihong Huang. 2016. Towards accurate event detection in social media: A weakly supervised approach for learning implicit event indicators. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 70–77, Osaka, Japan. The COLING 2016 Organizing Committee.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic. Association for Computational Linguistics.

Katherine Keith, Abram Handler, Michael Pinkham, Cara Magliozzi, Joshua McDuffie, and Brendan O'Connor. 2017. Identifying civilians killed by police with distantly supervised entity-event extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1547–1557, Copenhagen, Denmark. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinform.*, 9.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500, Jeju Island, Korea. Association for Computational Linguistics.

Jiwei Li, Alan Ritter, Claire Cardie, and Eduard Hovy. 2014. Major life event extraction from twitter based on congratulations/condolences speech acts. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1997–2007, Doha, Qatar. Association for Computational Linguistics.

Linguistic Data Consortium. 2005. Ace (automatic content extraction) english annotation guidelines for events.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

John McCarthy. 2002. Actions and other events in situation calculus. In *Proceedings of the Eights International Conference on Principles and Knowledge Representation and Reasoning (KR-02), Toulouse, France, April 22-25, 2002*, pages 615–628. Morgan Kaufmann.

Teruko Mitamura, Zhengzhong Liu, and Eduard H. Hovy. 2016. Overview of TAC-KBP 2016 event nugget track. In *Proceedings of the 2016 Text Analysis Conference, TAC 2016, Gaithersburg, Maryland, USA, November 14-15, 2016*. NIST.

Teruko Mitamura, Yukari Yamakawa, Susan Holm, Zhiyi Song, Ann Bies, Seth Kulick, and Stephanie Strassel. 2015. Event nugget annotation: Processes and issues. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 66–76, Denver, Colorado. Association for Computational Linguistics.

Aakanksha Naik, Chris Bogart, and Carolyn Rose. 2017. Extracting personal medical events for user timeline construction using minimal supervision. In *BioNLP 2017*, pages 356–364, Vancouver, Canada,. Association for Computational Linguistics.

Aakanksha Naik and Carolyn Rosé. 2020. Towards open domain event trigger identification using adversarial domain adaptation.

Azadeh Nikfarjam, Abeed Sarker, Karen O'Connor, Rachel E. Ginn, and Graciela Gonzalez-Hernandez. 2015. Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features. *J. Am. Medical Informatics Assoc.*, 22(3):671–681.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, Texas. Association for Computational Linguistics.

Boyan Onyshkevych, Mary Ellen Okurowski, and Lynn Carlson. 1993. Tasks, domains, and languages for information extraction. In *TIPSTER TEXT PROGRAM: PHASE I: Proceedings of a Workshop held at Fredricksburg, Virginia, September 19-23, 1993*, pages 123–133, Fredericksburg, Virginia, USA. Association for Computational Linguistics.

James Pustejovsky, José M. Castaño, Robert Ingria, Roser Saurí, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. 2003a. Timeml: Robust specification of event and temporal expressions in text. In *New Directions in Question Answering, Papers from 2003 AAAI Spring Symposium, Stanford University, Stanford, CA, USA*, pages 28–34. AAAI Press.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003b. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from twitter. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 1104–1112. ACM.

Abeed Sarker and Graciela Gonzalez. 2015. Portable automatic text classification for adverse drug reaction detection via multi-corpus training. *J. Biomed. Informatics*, 53:196–207.

Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. 2005. Evita: A robust event recognizer for QA systems. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 700–707, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Matthew Sims, Jong Ho Park, and David Bamman. 2019. Literary event detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3623–3634, Florence, Italy. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

William F. Styler IV, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet C de Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, and James Pustejovsky. 2014. Temporal annotation in the clinical domain. *Transactions of the Association for Computational Linguistics*, 2:143–154.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488, Copenhagen, Denmark. Association for Computational Linguistics.

Tuangthong Wattarujeekrit, Parantu K. Shah, and Nigel Collier. 2004. Pasbio: predicate-argument structures for event extraction in molecular biology. *BMC Bioinform.*, 5:155.

Miaomiao Wen, Zeyu Zheng, Hyeju Jang, Guang Xiang, and Carolyn Penstein Rosé. 2013. Extracting events with informal temporal references in personal histories in online communities. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 836–842, Sofia, Bulgaria. Association for Computational Linguistics.

Yftah Ziser and Roi Reichart. 2017. Neural structural correspondence learning for domain adaptation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 400–410, Vancouver, Canada. Association for Computational Linguistics.

Appendix

A Implementation Details

BERT: The BERT baseline model uses the uncased variant of BERT-Base (with no additional fine-tuning) for feature extraction. We generate token representations by running BERT-Base and concatenating the outputs of the model's last 4 hidden layers. The BiLSTM layer has a hidden size of 100, with an input dropout of 0.5. The MLP layer is 100-dimensional. These values are consistent with the setup in Naik and Rosé (2020).

BERT-ADA: The domain predictor (adversary) is a 3-layer MLP with each layer having a dimensionality of 100 and ReLU activations between layers. For the hyperparameter λ, which is the constant used to weight domain prediction loss, we experiment with values from [0.5,1.0,2.0,5.0], and choose the best model based on F1 scores on the source domain validation set. We run one search trial with a fixed random seed (0) for all settings. The best performing model on clinical notes uses $\lambda = 1.0$ and on conversations uses $\lambda = 0.5$.

BERT-LIW: The autoregressive word-level language models used for weighting are 3-layer LSTMs, with a hidden size of 300 and layer dropout of 0.2 at each layer. Input embeddings are initialized using 300-dimensional GloVe embeddings, with parameter typing between input and output embedding matrices. The models are trained using SGD with gradient clipping at 0.25 and a batch size of 16 for 25 epochs. Training starts with a learning rate of 20, which is divided by 4 whenever validation loss plateaus.

BERT-DAFT/BERT-DAFT-SYN: BERT-Base is fine-tuned for 3 epochs, using a batch size of 4 and default parameter settings in the Huggingface transformers library.

All event extraction models are trained with a batch size of 16 and use Adam optimizer with a learning rate of 0.001. Models are trained for 1000 epochs with early stopping. All experiments are run on an NVIDIA GeForce GTX 1080 Ti machine.

B Event Typology Examples

1. **Occurrence:** Occurrence refers to all events describing something that happens or occurs in the world. This is the broadest class of events. For example, "I *took* Midol yesterday."

2. **Aspectual:** Aspectual events refer to events which focus on various aspects of a different event's history, such as initiation, termination, continuation etc. For example, "I *started* taking this medicine last Friday." Here *started* is an aspectual event describing the initiation

of the event *taking*.

3. **Reporting:** Reporting events describe the action of an entity (person/group/organization) declaring something, narrating an event, providing information about an event etc. For example, ”So you *said* you have been experiencing symptoms since yesterday?”

4. **Perception:** Perception events refer to events involving the physical perception of a different event. For example, ”I *watched* my weight gain throughout the pregnancy.”

5. **State:** States describe circumstances in which something obtains or holds true. For example, ”My blood pressure is *higher* today”. Note that annotation of state events in TimeML is subject to certain rules.

6. **Intensional Action (I-Action):** Intensional actions introduce an explicit event argument describing an action or situation, from which we can infer something given its relation with the intensional action. For example, ”We will *investigate* your symptoms further via this test.” Here *investigate* is an intensional action associated with the *symptoms* event.

7. **Intensional State (I-State):** Intensional states contain stative events that refer to alternative or possible worlds. For example, ”You might observe *higher* blood pressure for a few days when you start taking this medicine.”

8. **Activity Pattern:** Activity patterns refer to events that are not clearly grounded to a single occurrence in time, but still considered events since the presence of a participant stops them from being purely generic. For example, ”You should *take* your blood pressure regularly.”

9. **Long Term State:** Long-term states expand the annotation of states beyond TimeML restrictions, allowing the inclusion of long-term chronic conditions. For example, ”You have a history of *COPD*.”

NoiseQA: Challenge Set Evaluation for User-Centric Question Answering

Abhilasha Ravichander **Siddharth Dalmia** **Maria Ryskina**
Florian Metze **Eduard Hovy** **Alan W Black**
Language Technologies Institute, Carnegie Mellon University, USA
`{aravicha,sdalmia,mryskina}@cs.cmu.edu`

Abstract

When Question-Answering (QA) systems are deployed in the real world, users query them through a variety of interfaces, such as speaking to voice assistants, typing questions into a search engine, or even translating questions to languages supported by the QA system. While there has been significant community attention devoted to identifying correct answers in passages assuming a perfectly formed question, we show that components in the pipeline that precede an answering engine can introduce varied and considerable sources of error, and performance can degrade substantially based on these upstream noise sources even for powerful pre-trained QA models. We conclude that there is substantial room for progress before QA systems can be effectively deployed, highlight the need for QA evaluation to expand to consider real-world use, and hope that our findings will spur greater community interest in the issues that arise when our systems actually need to be of utility to humans.[1]

1 Introduction

Everyday users now benefit from powerful QA technologies in a range of consumer-facing applications including health (Jacquemart and Zweigenbaum, 2003; Luo et al., 2015; Abacha and Demner-Fushman, 2016; Kilicoglu et al., 2018; Guo et al., 2018), privacy (Sathyendra et al., 2017; Harkous et al., 2018; Ravichander et al., 2019), personal finance (Alloatti et al., 2019), search (Yang, 2015; Bajaj et al., 2016; He et al., 2018; Kwiatkowski et al., 2019) and dialog agents (Dahl et al., 1994; Raux et al., 2005). Voice assistants such as Amazon Alexa[2] or Google Home[3] have brought natural language technologies to several million homes globally (Osborne, 2016; Jeffs, 2018). Yet, even with millions of users now interacting with these technologies on a daily basis, there has been surprisingly little research attention devoted to studying the issues that arise when people use QA systems.

Traditional QA evaluations do not reflect the needs of many users who *can* benefit from QA technologies. For example, users with a range of visual and motor impairments now rely extensively on voice interfaces (Pradhan et al., 2018) for efficient text entry.[4] Another need is cross-lingual information access, e.g. in scenarios where a speaker of one of the ∼7000 non-English living languages in the world (Eberhard et al., 2020) may want to take advantage of an English QA system.[5] QA evaluation has to keep up with the different ways in which users may use these systems in practice, and the different users who interact with these systems.

Keeping these needs in mind, we construct evaluations considering the *interfaces* through which users interact with QA systems.[6] We analyze errors introduced by three interface types that could be connected to a QA engine: **speech recognizers** converting spoken queries to text, **keyboards** used to type queries into the system, and **translation systems** processing queries in other languages. Our contributions are as follows:

1. We identify and describe the problem of interface noise for QA systems. We construct a challenge set framework for errors introduced by three kinds of interfaces: speech recognizers, keyboard interfaces, and translation engines, based on the popular SQuAD question-answering benchmark (Rajpurkar et al., 2016). We define synthetic noise generators, as well

---

[1] All resources available at `noiseqa.github.io`.

[2] `developer.amazon.com/alexa`

[3] `assistant.google.com`

[4] More than 3.4 million American adults over the age of 40 have a form of visual impairment (Congdon et al., 2004).

[5] As of 2021-01-24, there are 6,235,415 articles on English Wikipedia making it the largest edition: `wikicount.net`

[6] ‘QA system’ refers to any computing engine that receives a users’ question and constructs an answer. It may consist of an end-to-end neural architecture or a structured pipeline.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2976–2992
April 19 - 23, 2021.

Original Question	Interface	Synthetic Construction	Natural Construction
What has a Lama determined to do?	ASR	what has a llama determined to do	what has a llama determined to do
What has a Lama determined to do?	Keyboard	Wjat has a Lsma determined yo do?	WHat has a Lama determied to do?
What has a Lama determined to do?	MT	What has a Lama decided to do?	What is a llama determined to do?

Table 1: Example question perturbations from synthetic and natural noise challenge sets for three types of interfaces: Automatic Speech Recognition (ASR) systems, Keyboard and Machine Translation (MT) systems.

as manually construct natural noise challenge sets, by processing SQuAD questions through the specified interfaces.

2. We evaluate the performance of current state-of-the-art methods on natural and synthetic noisy data. We find that accessibility needs to be consciously worked towards, as we see that the performance of QA systems can be impacted by the choice of interface.
3. We analyze the generated noise and its impact on the downstream question answering and conduct an initial exploration of mitigation strategies for interface errors, focusing on data augmentation and query repair.

2 Motivation

Modern QA systems often rely on large databases of digital text such as Wikipedia as their source of knowledge; such corpora typically contain well-formed text in a high-resource language like English. However, the user's input could come in many different forms: it could be spoken, or written but in another, possibly lower-resource language. To convert these inputs into the format that the system can process, another machine learning system such as a speech recognizer or a machine translation engine is required, and these intermediate systems will inevitably propagate their decoding errors into the QA engine. However, interface errors are not necessarily artifacts of machine learning models: even when the question comes in the desired form (e.g. English text), it has to be communicated to the QA system through a mechanical interface such as a keyboard, and the process of typing can introduce errors such as character substitutions. To be useful in real-world settings, a QA system has to be able to correctly process the input question regardless of the input interface. We simulate the use cases for three interface categories (ASR, MT, and keyboard) with different level of human involvement, from fully automatic pipelines to leveraging existing human-generated resources to manual annotation, and evaluate whether the modern QA systems are capable of going from controlled well-formed inputs to real-world scenarios.

3 Challenge Set Construction

We define a suite of three types of noise perturbations, each imitating noise specific to a category of interfaces, and apply them to the data to create the challenge sets. We choose to add the noise to the questions but not to the context paragraphs, to replicate a realistic scenario of the noise being introduced to the question by the interface through which the user interacts with the QA engine. For each type of noise, we both build a synthetic generator that can introduce noise on a large scale, as well as manually create 'natural' noise challenge sets to imitate real-world noise.

Our challenge sets are based on SQuAD 1.1 (Rajpurkar et al., 2016),[7] a large-scale machine comprehension dataset based on Wikipedia articles where the answer to each question is a span in a provided context. We choose SQuAD both for its popularity as a benchmark (Gardner et al., 2018; Devlin et al., 2019; Radford et al., 2018; Wolf et al., 2019) and to avoid additional confounds such as unanswerable questions (Rajpurkar et al., 2018).[8] We use the standard ~90K/10K train/development split and construct the challenge sets from the XQuAD data (Artetxe et al., 2020), a subset of 1,190 SQuAD development set questions accompanied by professional translations into ten languages.[9] Below we discuss each challenge set in more detail.

---

[7] Though in principle, these constructions could be applied to any kind of QA dataset

[8] Future work would pursue a context-driven evaluation of unanswerability, identifying the kinds of unanswerable questions users ask in practice (Ravichander et al., 2019; Asai and Choi, 2020).

[9] Spanish, German, Greek, Russian, Turkish, Arabic, Vietnamese, Thai, Chinese, and Hindi.

3.1 MT Noise

Our first challenge set emulates machine translation noise introduced when the question is asked in a language other than the language of the QA system's training data. We use English as the QA system language, pairing English contexts with non-English questions.

Synthetic Challenge Set Our synthetic noise generator employs the back-translation technique (Sennrich et al., 2016; Dong et al., 2017; Yu et al., 2018). In our case, back-translation is not meant to act as a data augmentation technique but rather to simulate noise that could be introduced by an MT engine when translating the question from another language. We imperfectly approximate natural non-English input by automatically translating English questions into a pivot language (German); we then translate them back to English, imitating a scenario where the user submits a query through an MT engine. We use the HuggingFace implementation (Wolf et al., 2019) of MarianNMT (Junczys-Dowmunt et al., 2018).[10]

Natural Challenge Set To bring our simulation closer to the natural setting, we create another challenge set from English machine translations of human-generated questions in other languages. We take the questions from the XQuAD dataset, which consists of English questions paired with professional translations into ten other languages.[11] For each of the test set languages, we use Google's commercial translation engine[12] to produce the English translation of the question. This allows us to construct ten challenge sets of translations from different languages with 1,190 questions each.

3.2 Keyboard Noise

This challenge set represents the noise introduced in the process of typing a question up on a keyboard, for example, when a question is submitted to a QA system through a search engine.

Synthetic Challenge Set Inspired by prior work (Belinkov and Bisk, 2018; Naik et al., 2018), our basic noise generator introduces per-character typos based on the proximity of the keys in a standard QWERTY keyboard layout. Each word is corrupted with a 25% probability by substituting a randomly sampled character with its row-wise neighbor. We also create more natural-looking noise by introducing externally collected human misspellings into our data on word level, as proposed by Belinkov and Bisk (2018). Although prior work refers to this as natural noise, emphasizing that the typos have been produced by humans, we consider it synthetic because the errors are applied to the data outside of their original context. We start with the Wikipedia common English misspellings list[13] and apply a simple filtering heuristic that only retains keyboard errors (see Appendix C), obtaining 1,742 misspellings for 1,489 English words.

ORIGINAL QUESTION	How many Panthers defense players were selected for the Pro Bowl?
GOOGLE ASR	how many Santa's defense players selected for the Pro Bowl
ESPNET (WITH LM)	how many pantols the tent places were slected for the probol
KALDI (WITH LM)	how many friends tons of defence UNK for the UNK

Table 2: Example outputs of different ASR systems on a recorded question from SQuAD (Rajpurkar et al., 2016).

Natural Challenge Set To generate errors specific to the context of the question rather than hypothesized to exist at a lexical level across contexts, we ask three human annotators to retype English XQuAD questions. Annotators can see the original question, which helps avoid errors caused by misconception (e.g. not knowing the correct spelling of a named entity), but not their own input, in order to prevent them from correcting the typos. Of the obtained noisy questions, 51.6% and 25.7% differ from the original by at least one or at least two characters respectively.

3.3 ASR Noise

Our final challenge set simulates ASR errors that occur when a question is posed to a voice interface.

Synthetic Challenge Set We emulate automatic recognition of natural speech by using a Text-to-Speech (TTS) system pipelined with an ASR engine (Tjandra et al., 2017). We voice the questions using Google TTS and transcribe the obtained

---

[10] `huggingface.co/Helsinki-NLP/opus-mt-{en-de|de-en}`

[11] A subtle nuance is that XQuAD questions are not originally written in these languages but translated from English; acknowledging this, we use XQuAD data as the natural challenge set because its fully parallel nature allows varying input language while controlling for content for fair comparison.

[12] `translate.google.com`

[13] `en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings`

Interface	CER (↓)	WER (↓)	BLEU (↑)
Synthetic			
ASR	3.96	16.61	77.12
Keyboard	4.11	23.93	52.66
Translation	20.51	29.36	58.42
Natural			
ASR	12.96	30.67	57.22
Keyboard	1.78	7.42	85.78
Translation	31.89	43.34	47.07

Table 3: % Character Error Rate (CER), % Word Error Rate (WER) and BLEU scores for all challenge sets compared to ground truth. For ASR and MT, synthetic noise is less prominent than natural, reflecting the idealized simulation conditions. As expected, natural keyboard noise demonstrates the best word-level statistics.

speech using Google Speech-to-Text optimized for English–US. Besides Google ASR, we use Kaldi ASpIRE (Povey et al., 2011; Peddinti et al., 2015) and ESPnet CommonVoice (Watanabe et al., 2018; Ardila et al., 2020) open-source systems, as shown in Table 2. We choose the former for analyzing the downstream effect of out-of-vocabulary word prediction in fixed vocabulary decoding (Peskov et al., 2019) and the latter for data augmentation (§4.2) due to its improved out-of-vocabulary word handling with subword units. In order to generate the large amount of speech data needed for augmentation, we use the open-source ESPnet LJSpeech TTS (Hayashi et al., 2020; Ito and Johnson, 2017) to voice the questions.

Natural Challenge Set We use the SANTLR speech annotation toolkit (Li et al., 2019) to record spoken versions of the prompt question from three human annotators (for background details, see Appendix D). The obtained recordings are then transcribed using the ASR engines listed above. As expected, recognizing human speech is more difficult: the word error rate of the Google ASR system on the obtained set is 31%, compared to 17% on the synthesized English–US speech.

4 Experiments

We select four QA models that demonstrated strong performance on SQuAD 1.1[14] to be tested under interface distortions: BiDAF (Seo et al., 2017), which represents contexts at different levels of granularity using bidirectional attention flow mechanism; its extension BiDAF-ELMo (Peters et al., 2018) augmented with contextualized embeddings; BERT (Devlin et al., 2019), a bidirectional Transformer-based language model (Vaswani et al., 2017); and RoBERTa (Liu et al., 2019), a more robustly pre-trained version of BERT.

4.1 Results and Analysis

Table 3 shows the character error rate (CER), word error rate (WER) and BLEU score[15] for the generated challenge sets. Synthetic ASR and MT pipelines introduce substantially less noise than their natural counterparts, while the opposite holds for the keyboard. This is likely due to the generators not being equally controllable: while we can arbitrarily make the synthetic keyboard set noisier by increasing the corruption rate, synthetic ASR and MT pipelines include black-box components which also make the task easier for the interface by design (TTS synthesizes idealized speech, back-translation mimics MT training conditions).

In this section, we investigate how robust QA models are to these interface errors. Table 4 reports the performance on both synthetic and natural challenge sets. For brevity, we present results using the German–English model and the Google ASR for MT and ASR respectively.

First, we observe that both synthetic and natural noise decrease accuracy for all models and interfaces, with synthetic keyboard and natural ASR errors being the most challenging. As for MT noise, Table 4 reports results on German queries; although the systems seem robust on these, we find that MT noise can actually be quite challenging with sharp degradation of performance on Thai and Arabic (Figure 2). Further, we notice that the relative performance of models on the development set is not necessarily a sufficient proxy for the relative robustness of models to interface errors: while BERT and RoBERTa perform very similarly on XQuAD–English, RoBERTa outperforms BERT on handling all three kinds of interface errors. For practitioners, this could suggest that simply choosing the highest-accuracy QA model without separately evaluating robustness to interface noise may lead to sub-optimal performance in practice.

Below we discuss the effect of each interface in more detail.

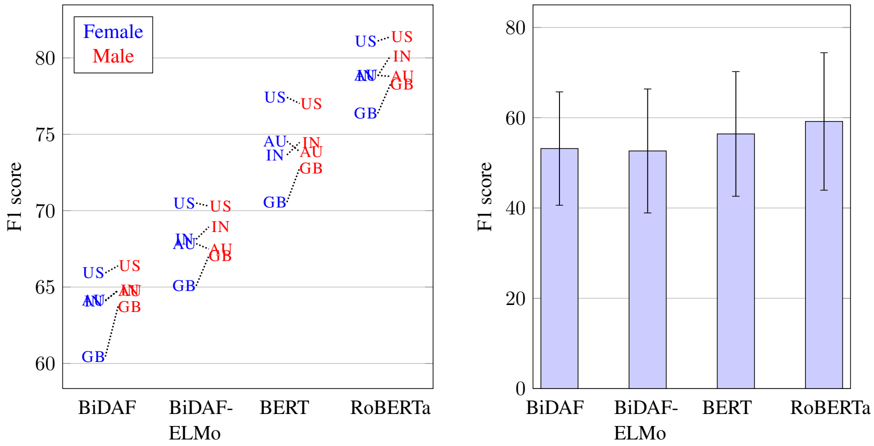
---

[14] F1 scores on SQuAD dev set: BiDAF: 77.8; BiDAF-ELMo: 80.7; BERT: 88.8; RoBERTa: 89.9. For hyperparameters and implementation details, see Appendix A.

[15] Uncased detokenized BLEU using SacreBLEU (Post, 2018).

Model	XQuAD$_{\text{EN}}$		ASR		MT		Keyboard	
	EM	F1	EM	F1	EM	F1	EM	F1
Synthetic								
BiDAF (Seo et al., 2017)	60.08	71.96	54.62	66.39	55.97	68.01	45.21	57.78
BiDAF-ELMo (Peters et al., 2018)	62.61	75.38	56.81	70.30	57.39	70.05	50.93	63.80
BERT (Devlin et al., 2019)	72.77	84.66	61.93	77.02	67.23	79.08	61.76	73.64
RoBERTa (Liu et al., 2019)	72.35	84.42	68.07	81.38	68.40	80.93	65.04	76.97
Natural								
BiDAF (Seo et al., 2017)	60.08	71.96	45.97	57.64	54.87	66.90	56.89	68.33
BiDAF-ELMo (Peters et al., 2018)	62.61	75.38	49.16	62.49	59.24	71.06	60.76	73.32
BERT (Devlin et al., 2019)	72.77	84.66	52.94	67.13	68.82	79.98	69.16	81.84
RoBERTa (Liu et al., 2019)	72.35	84.42	60.08	73.61	70.00	82.13	70.92	83.37

Table 4: Performance of the QA models under the three kinds of interface noise: ASR (using Google ASR), MT (with the German–English model), and keyboard. All models score lower on noisy data, most notably on the natural ASR set. MT noise is less prominent, but we later show its impact is highly dependent on the input language.



(a) F1 scores for TTS voices over 4 accent and 2 gender settings. Lines connect the results for different gender but same accent and model.

(b) F1 mean and standard deviation over 4 human voices. For details and score breakdown by speaker, see Appendix D.

Figure 1: Effect of synthetic (a) and natural (b) voice variation on the QA performance in an ASR pipeline. Synthetic voice variation is achieved by varying accent and gender settings in the Google TTS model; US accent setting shows the highest scores while neither gender setting consistently performs best (indicated by line slopes). Natural variation is measured on a sample of 100 questions narrated by four annotators. All models exhibit considerable variation in both experiments.

ASR Noise: Speech recognizers typically omit punctuation, which could mean losing cues important for the downstream task. To look at this factor in isolation, we remove punctuation from the original XQuAD questions. This change alone decreases BERT performance by 5.1 F1, suggesting that the absence of punctuation in part explains the degradation in the presence of ASR noise. When we qualitatively analyze a sample of 50 questions that BERT answered successfully in the original setting but not when passed through the speech interface, we find that 14% of them are identical to the original modulo punctuation. Other sources of error include the ASR producing completely meaningless questions (28%), hallucinating (12%) or losing named entities (10%), and replacing words with homonyms (4%); other difficult cases include recognizing acronyms and preserving possessives, tense, and number (2% each). Although these problems could be diminished by designing better interfaces, we believe it is also worthwhile for practitioners to work on improving robustness of the QA systems itself: many interfaces, especially commercial, only offer black-box access, and building a completely noise-free interface is not feasible.

Voice variation also plays a role: ASR error distribution differs by speaker background variables such as accent (Zheng et al., 2005), in turn affecting the downstream systems (Harwell, 2018; Lima et al., 2019; Palanica et al., 2019). To emulate speaker variation in the synthetic setting, we use Google English Text-to-Speech to pronounce the XQuAD questions in eight different voices, varying the provided accent and gender settings. As Figure 1a shows, all models exhibit considerable variation in F1 score, consistently performing best on synthetic US accent (which our speech recognizer is optimized for) and worst on GB. Score breakdowns by setting can be found in Appendix D.

We also repeat the experiment with four human speakers narrating a sample of 100 XQuAD questions, to control for content. As shown in Figure 1b, each model's performance varies substantially between voices. The four speakers differ by accent (2 Indian, 1 Russian, 1 Scottish), gender (2 male, 2 female), and level of proficiency (native and non-native); more details and individual speaker scores can be found in Appendix D.[16] Although improving robustness to accent variation is out of the scope of our work, we highlight that the performance can degrade sharply depending on the user and their acoustic conditions.

We also analyze how the choice of ASR model affects the QA accuracy, focusing in particular on the decoding strategies for out-of-vocabulary words. We compare Kaldi, which outputs an UNK token for unknown words (Peddinti et al., 2015), and Google's large-vocabulary ASR model. On our set of human voices, Kaldi produces at least one UNK token for $\sim$50% of the questions, and BERT achieves an F1 score of only 43.6 on this set (54.4 F1 and 32.3 F1 separately on questions with and without UNK respectively) compared to 67.1 F1 achieved by Google ASR, demonstrating that speech recognizer choice can greatly affect downstream QA performance. The observed degradation due to UNK decoding (previously noted by Peskov et al., 2019) suggests that practitioners might find it useful to go beyond speech recognition benchmarks, and also evaluate ASR systems in the context of downstream QA applications.

Translation Noise: As Table 4 shows, German–English translation errors affect the performance of all models, although to a lesser extent than ASR noise. However, the MT quality and, in turn, the downstream performance varies greatly depending on the source language. Figure 2 shows BERT and RoBERTa F1 scores on questions translated from each of the ten XQuAD languages to English (numbers reported in Appendix E). While German and Spanish have the highest accuracy, lower-resource and more typologically distant languages like Arabic and Thai are far behind. On translated Thai inputs, BERT achieves only 71.0 F1, which is a 16% drop in accuracy from the original English setting compared to 6% for German.

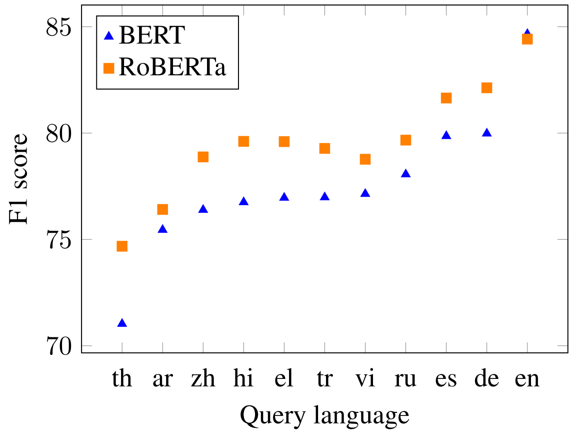


Figure 2: Effect of the input language on the QA system performance in an MT pipeline. Automatically translating non-English queries to English decreases performance for all source languages, and the decrease is especially noticeable for lower-resource languages.

Table 5 shows example translations from four XQuAD languages and highlights their divergences from the original questions. Since the questions are being translated out of context, MT tends to replace important content words with ones that are semantically related but not appropriate in given context (*Lord*→*deity*, *chair*→*President*, *ctenophore*→*jellyfish*). Transliteration of technical terms and named entities is also a challenge, especially for languages written in non-Latin scripts (*ctenophore*→*tenophora* through Hindi, *Jochi*→*Dschötschi* through German). For further qualitative analysis, we sample 100 questions translated from Hindi which BERT fails to answer correctly despite accurately answering their English equivalent. Of these, 30% were identified by a native speaker annotator as paraphrases of the original question that would admit the original answer. The remaining incorrect translations are due to question type shift (31%), ungrammatical or meaningless questions (12%), corrupted named entities (8%) and dropped determiners (2%; Hindi does not generally use definite articles). Some divergences also go beyond word level, e.g. 10% of questions have semantic role inversion (*What earlier market did the Grainger Market replace?*→*Which earlier market replaced Granger's market?*). While some word-level errors can be corrected post-hoc, repairing syntax is much more challenging, which again

---

[16] Comparisons between demographics should not be drawn from per-speaker results, since we do not control for confounds like recording conditions, aiming for a realistic sample.

Language	Question	Language	Question
What type of Lord is Doctor Who?		**When would the occupation of allies leave Rhineland?**	
de:	What kind of gentleman is Doctor Who?	**de**:	When would the Allied occupation leave the Rhineland?
zh:	What type of lord is Doctor Who?	**zh**:	When was the Allies scheduled to withdraw from Rhineland?
hi:	What kind of deity is Doctor Who?	**hi**:	When will the Rhineland be removed from the occupation of the Allied countries?
ru:	What type of overlord is Doctor Who?	**ru**:	When did the Allies intend to remove the occupation of the Rhine region?
Who is the chair of the IPCC?		**How much food does a ctenophora eat in a day?**	
de:	Who is the chair of the IPCC?	**de**:	How much food does a jellyfish eat in a day??
zh:	Who is the current chairman of the IPCC?	**zh**:	How much food does a jellyfish eat in a day?
hi:	Who is the President of IPCC?	**hi**:	How much food does a tenophora eat in a day?
ru:	Who is the chairman of the IPCC?	**ru**:	How much food does a ctenophore eat per day?

Table 5: Examples of translation divergences for German (de), Chinese (zh), Hindi (hi), and Russian (ru).

brings it down to the robustness of the QA engine.

Keyboard Noise: Synthetic keyboard noise produced by our key-swap typo generator has a much stronger effect on the QA performance than natural noise (11.1 F1 and 2.4 F1 drop respectively). We attribute this to differences in the perturbation intensity: $\sim$25% of question words are corrupted in the synthetic setting, but only $\sim$9% of words are corrupted under natural conditions.[17] Interestingly, BiDAF- and BERT-based models consistently show comparable decreases in F1 score, suggesting that character-level tokenization of the former does not on its own guarantee robustness to typos.

Another factor that could affect downstream performance is error placement. We evaluate BERT on three additional synthetic sets, introducing noise to only function words (conjunctions, pronouns, articles), only content words (which we limit to nouns and adjectives), or only commonly misspelled words (using the Wikipedia misspellings list as described in §3.2). Synthetically perturbing all function words and all content words decreases F1 score by 6.7 and 11.7 respectively, confirming that not all words are equally important for the model finding the correct answer. Injecting the interface errors from Wikipedia into the 2,716 questions containing at least one commonly misspelled word yields F1 score of 78.6 (6.1 F1 drop), showcasing the decreased performance we would likely see in real-life user interactions.

4.2 Mitigation Strategies

We experiment with two strategies for improving the QA system robustness: repairing the question errors using the provided context and retraining QA models on the data augmented with synthetic noise. Question repair assumes availability of context, making it unsuitable for open-domain QA, but reasonable for use cases like QA over manuals or policies (Feng et al., 2015; Harkous et al., 2018; Ravichander et al., 2019). This approach treats words that occur in the question but not the context as potential noise, attempting to replace them with the closest candidate from the context paragraph. We use character error rate as the distance metric, empirically setting the threshold to 0.5 using the synthetic set. We perform two experiments, applying the repair either only to content words (here, nouns and adjectives) or only to named entities in both the context and the question. Table 6 shows how these repairs affect BERT performance on three types of natural noise. Named entity repair yields marginal improvements across the board, while content word repair has a stronger effect but only for keyboard errors. The proposed strategy could also be combined with other deterministic or off-the-shelf repair methods, such as adding final question marks for ASR (+6.52 F1) or using a spellchecker for keyboard (+1.41 F1).

For data augmentation, we use our synthetic noise generators to inject noise into $\sim$90K SQuAD training questions and retrain BERT on the combined clean and noisy data. As Table 6 shows, augmentation yields improvements on all three types of natural noise over BERT trained on clean data only, but the performance of the augmented models drops slightly on the clean data. Best results on natural ASR and MT noise are obtained when the data is augmented with the same type of synthetic noise; interestingly, this is not true for keyboard noise, where ASR augmentation also works best.

---

[17] Synthetic data corruption rate is a design decision and can be made to simulate the expected natural noise or be more challenging as a stress test, depending on practitioner's goals.

	XQuAD_{EN}		ASR		MT		Keyboard	
BERT Model	EM	F1	EM	F1	EM	F1	EM	F1
BERT	72.77	84.66	52.94	67.13	68.82	79.98	69.16	81.84
+ Named entity repair	**72.94**	**84.78**	53.03	67.34	68.82	80.05	69.58	82.22
+ Content word repair	**72.94**	84.77	52.61	67.01	68.32	79.76	70.25	**82.60**
+ Augmentation	72.35	83.89	**64.37**	**75.89**	**68.90**	**80.83**	**70.76**	82.43

Table 6: Effect of question repair and data augmentation on BERT performance on three types of natural noise. Results on synthetic noise and data augmentation score breakdown by interface can be found in Appendix F.

Although our results are preliminary, they suggest that augmentation could prove useful in enabling effective question answering in the real world.

To better understand where ASR and MT augmentation helps, we compare the performance of augmented and baseline BERT on additional challenge sets, synthesizing some common noise artifacts in isolation. We find that ASR noise augmentation improves robustness to omission of punctuation: ASR-augmented model achieves 82.7 F1 on questions with no punctuation and 82.9 F1 on questions without the final question mark (compared to 79.2 and 79.6 F1 for the baseline). Following the definitions in §4.1, we also experiment with removal of function and content words: both augmented models outperform baseline when all function words are dropped (76.1 F1 for ASR, and 70.2 F1 for MT, and 67.8 F1 for baseline), and ASR augmentation helps when all content words are dropped (68.6 F1 vs. 66.0 F1 for baseline). Finally, we replace one randomly sampled named entity (of type LOC, ORG, or PER) per question with a placeholder, and the performance of ASR-augmented BERT drops less than that of the baseline BERT (by 2.3% and 3.2% respectively). This analysis suggests that ASR augmentation can make models more robust to errors in punctuation, named entities, and content words, and both ASR and MT could help with function word errors.

On the utility of synthetic challenge sets: We advocate that dataset designers always obtain natural data (with natural noise) when possible. However, in the circumstances where collecting natural data is difficult, synthetic data can be useful when reasonably constructed. While the distribution of errors in our synthetically generated challenge sets differs from that in the natural ones (Table 3), we find that the model performance ranking is consistent across all types of noise (Table 4), showing that synthetic noise sets could act as a proxy for model selection. Moreover, augmenting training data with synthetic noise improves model robustness to natural noise for all noise types in this study (Table 6), suggesting that synthetic noise generators may be capturing some aspects of natural noise. Our proposed generators could serve as templates for synthesizing interface noise when collecting natural data is infeasible, but individual practitioners should carefully identify and simulate the likely sources of error appropriate for their applications.

5 Related Work

Question Answering QA systems have a rich history in NLP, with early successes in domain-specific applications (Green et al., 1961; Woods, 1977; Wilensky et al., 1988; Hirschman and Gaizauskas, 2001). Considerable research effort has been devoted to collecting datasets to support a wider variety of applications (Quaresma and Pimenta Rodrigues, 2005; Monroy et al., 2009; Feng et al., 2015; Liu et al., 2015; Nguyen, 2019; Jin et al., 2019) and improving model performance on them (Lally et al., 2017; Wang et al., 2018; Yu et al., 2018; Yang et al., 2019). We too focus on QA systems but center the utility to users rather than new applications or techniques.

There has also been interest in studying the interaction between speech and QA systems. Lee et al. (2018a) examine transcription errors for Chinese QA, and Lee et al. (2018b) propose Spoken SQuAD, with spoken contexts and text-based questions, but they address a fundamentally different use case of searching through speech. Closest to our work is that of Peskov et al. (2019), which studies mitigating ASR errors in QA, assuming white-box access to the ASR systems. Most such work automatically generates and transcribes speech using TTS–ASR pipelines, similar to how our synthetic set is constructed. However, our results show that TTS does not realistically replicate human voice variation. Besides, stakeholders relying on commercial transcription services will not have white-box access to ASR; our post-hoc mitigation strategies would be better suited for such cases.

Challenge sets Model robustness evaluation with adversarial schemes is common in NLP tasks (Smith, 2012), including dependency parsing (Rimell et al., 2009), information extraction (Schneider et al., 2017), natural language inference (Marelli et al., 2014; Naik et al., 2018; Glockner et al., 2018), machine translation (Isabelle et al., 2017; Belinkov and Bisk, 2018; Bawden et al., 2018; Burlot and Yvon, 2017) and QA (Jia and Liang, 2017; Aspillaga et al., 2020). Unlike most prior work, we do not create our challenge sets to break QA systems, but rather for a more realistic evaluation of the systems' real-world utility.

6 Conclusion

In this work, we advocate for QA evaluations that reflect challenges associated with real-world use. In particular, we focus on questions that are written in another language, spoken, or typed, and the noise introduced into them by the corresponding interface (machine translation, speech recognition, or keyboard). We analyze the effect of synthetic and natural noise in each interface and find that these errors can be diverse, nuanced, and challenging for traditional QA systems. Although we present an initial exploration of mitigation strategies, our primary contribution lies not in the specific challenge sets we construct or in developing new algorithms, but rather in identifying and describing one class of problems that practical QA systems must consider and providing a framework to measure them. We hope insights derived from our study stimulate research in making QA systems ready to face real-world users. We emphasize three considerations:

Sources of error: This work studies errors introduced at the interface stage of QA pipelines. These errors are nearly ubiquitous, as users always interact with QA systems through some kind of interface. Thus, it is important for QA system designers to be mindful of distortions those might introduce. Our analysis can be extended to study the impact of interface-specific factors: for example, how errors vary by keyboard layout (e.g. QWERTY vs. Dvorak or language-specific layouts like AZERTY) or preferred way of typing (e.g. using physical keyboards vs. swipe typing). Another fruitful area of study could lie in examining the accumulated impact of errors resulting from interface combinations (e.g. machine translation of ASR-transcribed queries) and the effects of such interface noise in languages other than English. However, interface distortion represents only one source of error that occurs in practical deployment, and future research would study further sources of variation such as how users may adapt their questions according to the interface used.

Context-driven evaluation: This work focuses on practical evaluation of QA systems that takes into account the challenges associated with their real-world deployment. We hope to encourage development of future user-centered or participatory design approaches to building QA datasets and evaluations, where practitioners work with potential users to understand user requirements and the contexts in which systems are used in practice.

Community priorities for QA systems: While leaderboards on established benchmarks have facilitated rapid progress (Rajpurkar et al., 2016, 2018) and bolstered development of a variety of semantic models (Xiong et al., 2018; Liu et al., 2018; Huang et al., 2018; Devlin et al., 2019), we call for practitioners to consider the orthogonal direction of *system utility* in their model design. We believe these subareas to be complementary, and community attention towards both will help produce NLP systems that are both accurate and *usable*.

Acknowledgments

We thank Aakanksha Naik, Sujeath Pareddy, Taylor Berg-Kirkpatrick, and Matthew Gormley for helpful discussion and the anonymous reviewers for their valuable feedback. This work used the Bridges system, which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center (PSC). This research was supported in part by grants from the National Science Foundation Secure and Trustworthy Computing program (CNS-1330596, CNS-15-13957, CNS-1801316, CNS-1914486) and the DARPA KAIROS program from the Air Force Research Laboratory under agreement number FA8750-19-2-0200. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

Asma Ben Abacha and Dina Demner-Fushman. 2016. Recognizing question entailment for medical question answering. In *AMIA Annual Symposium Proceedings*, volume 2016, page 310. American Medical Informatics Association.

Francesca Alloatti, Luigi Di Caro, and Gianpiero Sportelli. 2019. Real life application of a question answering system using BERT language model. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 250–253, Stockholm, Sweden. Association for Computational Linguistics.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Akari Asai and Eunsol Choi. 2020. Challenges in information seeking qa: Unanswerable questions and paragraph retrieval. *arXiv preprint arXiv:2010.11915*.

Carlos Aspillaga, Andrés Carvallo, and Vladimir Araujo. 2020. Stress test evaluation of transformer-based models in natural language understanding tasks. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1882–1894, Marseille, France. European Language Resources Association.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint*.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Franck Burlot and François Yvon. 2017. Evaluating the morphological competence of machine translation systems. In *Proceedings of the Second Conference on Machine Translation*, pages 43–55, Copenhagen, Denmark. Association for Computational Linguistics.

Nathan Congdon, Benita O'Colmain, Caroline C. W. Klaver, Ronald Klein, Beatriz Muñoz, David S. Friedman, John Kempen, Hugh R. Taylor, and Paul Mitchell. 2004. Causes and prevalence of visual impairment among adults in the United States. *Archives of Ophthalmology (Chicago, Ill.: 1960)*, 122(4):477–485.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

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

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2020. *Ethnologue: Languages of the World*, 23 edition. SIL International, Dallas, Texas.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820. IEEE.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: An automatic question-answerer. In *Papers Presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference*, IRE-AIEE-ACM '61 (Western), page 219–224, New York, NY, USA. Association for Computing Machinery.

Haihong Guo, Xu Na, and Jiao Li. 2018. Qcorp: an annotated classification corpus of Chinese health questions. *BMC medical informatics and decision making*, 18(1):16.

Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G. Shin, and Karl Aberer. 2018. Polisis: Automated analysis and presentation of privacy policies using deep learning. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 531–548, Baltimore, MD. USENIX Association.

Drew Harwell. 2018. The accent gap. *The Washington Post*.

Tomoki Hayashi, Ryuichi Yamamoto, Katsuki Inoue, Takenori Yoshimura, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Yu Zhang, and Xu Tan. 2020. ESPnet-TTS: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7654–7658. IEEE.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Lynette Hirschman and Robert Gaizauskas. 2001. Natural language question answering: The view from here. *Natural Language Engineering*, 7(4):275–300.

Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. 2018. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. In *International Conference on Learning Representations*.

Pierre Isabelle, Colin Cherry, and George Foster. 2017. A challenge set approach to evaluating machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2486–2496, Copenhagen, Denmark. Association for Computational Linguistics.

Keith Ito and Linda Johnson. 2017. The LJ Speech dataset. `keithito.com/LJ-Speech-Dataset/`.

Pierre Jacquemart and Pierre Zweigenbaum. 2003. Towards a medical question-answering system: a feasibility study. *Studies in health technology and informatics*, 95:463.

Mike Jeffs. 2018. OK Google, Siri, Alexa, Cortana; can you tell me some stats on voice search. Accessed 2020-09-27.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Halil Kilicoglu, Asma Ben Abacha, Yassine Mrabet, Sonya E Shooshan, Laritza Rodriguez, Kate Masterton, and Dina Demner-Fushman. 2018. Semantic annotation of consumer health questions. *BMC bioinformatics*, 19(1):34.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Adam Lally, Sugato Bagchi, Michael A. Barborak, David W. Buchanan, Jennifer Chu-Carroll, David A. Ferrucci, Michael R. Glass, Aditya Kalyanpur, Erik T. Mueller, J. William Murdock, Siddharth Patwardhan, and John M. Prager. 2017. WatsonPaths: scenario-based question answering and inference over unstructured information. *AI Magazine*, 38(2):59–76.

Chia-Hsuan Lee, Shang-Ming Wang, Huan-Cheng Chang, and Hung-Yi Lee. 2018a. ODSQA: Open-domain spoken question answering dataset. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 949–956. IEEE.

Chia-Hsuan Lee, Szu-Lin Wu, Chi-Liang Liu, and Hung-yi Lee. 2018b. Spoken SQuAD: A study of mitigating the impact of speech recognition errors on listening comprehension. In *Proc. Interspeech 2018*, pages 3459–3463.

Xinjian Li, Zhong Zhou, Siddharth Dalmia, Alan W. Black, and Florian Metze. 2019. SANTLR: Speech annotation toolkit for low resource languages. In *Proc. Interspeech 2019*, pages 3681–3682.

Lanna Lima, Vasco Furtado, Elizabeth Furtado, and Virgilio Almeida. 2019. Empirical analysis of bias in voice-based personal assistants. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, page 533–538, New York, NY, USA. Association for Computing Machinery.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1694–1704, Melbourne, Australia. Association for Computational Linguistics.

Yi-Hung Liu, Yen-Liang Chen, and Wu-Liang Ho. 2015. Predicting associated statutes for legal problems. *Information Processing & Management*, 51(1):194–211.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint*.

Jake Luo, Guo-Qiang Zhang, Susan Wentz, Licong Cui, and Rong Xu. 2015. SimQ: real-time retrieval of similar consumer health questions. *Journal of medical Internet research*, 17(2):e43.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Yajie Miao, Lili Zhao, Chunping Li, and Jie Tang. 2010. Automatically grouping questions in Yahoo! Answers. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 350–357. IEEE.

Alfredo Monroy, Hiram Calvo, and Alexander Gelbukh. 2009. NLP for shallow question answering of legal documents using graphs. In *Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '09, page 498–508, Berlin, Heidelberg. Springer-Verlag.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Vincent Nguyen. 2019. Question answering in the biomedical domain. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 54–63, Florence, Italy. Association for Computational Linguistics.

Joe Osborne. 2016. Why 100 million monthly Cortana users on Windows 10 is a big deal. *TechRadar*. Accessed 2020-09-27.

Adam Palanica, Anirudh Thommandram, Andrew Lee, Michael Li, and Yan Fossat. 2019. Do you understand the words that are comin outta my mouth? Voice assistant comprehension of medication names. *NPJ digital medicine*, 2(1):1–6.

Vijayaditya Peddinti, Guoguo Chen, Vimal Manohar, Tom Ko, Daniel Povey, and Sanjeev Khudanpur. 2015. JHU ASpIRE system: Robust LVCSR with TDNNS, iVector adaptation and RNN-LMS. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 539–546. IEEE.

Denis Peskov, Joe Barrow, Pedro Rodriguez, Graham Neubig, and Jordan Boyd-Graber. 2019. Mitigating noisy inputs for question answering. In *Proc. Interspeech 2019*, pages 789–793.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel

Vesely. 2011. The Kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society.

Alisha Pradhan, Kanika Mehta, and Leah Findlater. 2018. "Accessibility came by accident": Use of voice-controlled intelligent personal assistants by people with disabilities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–13, New York, NY, USA. Association for Computing Machinery.

Paulo Quaresma and Irene Pimenta Rodrigues. 2005. A question answer system for legal information retrieval. In *Proceedings of the 2005 Conference on Legal Knowledge and Information Systems: JURIX 2005: The Eighteenth Annual Conference*, page 91–100, NLD. IOS Press.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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

Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. 2005. Let's go public! Taking a spoken dialog system to the real world. In *Proc. Interspeech 2005*.

Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. 2019. Question answering for privacy policies: Combining computational and legal perspectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4947–4958, Hong Kong, China. Association for Computational Linguistics.

Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 813–821, Singapore. Association for Computational Linguistics.

Kanthashree Mysore Sathyendra, Abhilasha Ravichander, Peter Garth Story, Alan W. Black, and Norman Sadeh. 2017. Helping users understand privacy notices with automated query answering functionality: An exploratory study. Technical report, Carnegie Mellon University.

Rudolf Schneider, Tom Oberhauser, Tobias Klatt, Felix A. Gers, and Alexander Löser. 2017. Analysing errors of open information extraction systems. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, pages 11–18, Copenhagen, Denmark. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*.

Noah A. Smith. 2012. Adversarial evaluation for models of natural language. *arXiv preprint*.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Listening while speaking: Speech chain by deep learning. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 301–308. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced ranker-reader for open-domain question answering. In *AAAI Conference on Artificial Intelligence*.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proc. Interspeech 2018*, pages 2207–2211.

Robert Wilensky, David N. Chin, Marc Luria, James Martin, James Mayfield, and Dekai Wu. 1988. The berkeley unix consultant project. *Computational Linguistics*, 14(4).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art natural language processing. *arXiv preprint*.

William A. Woods. 1977. Lunar rocks in natural English: Explorations in natural language question answering. In *Linguistic Structures Processing*, pages 521–569. North Holland, Amsterdam.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. DCN+: Mixed objective and deep residual coattention for question answering. In *International Conference on Learning Representations*.

Min Yang. 2015. Deep Markov neural network for sequential data classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 32–37, Beijing, China. Association for Computational Linguistics.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *International Conference on Learning Representations*.

Yanli Zheng, Richard Sproat, Liang Gu, Izhak Shafran, Haolang Zhou, Yi Su, Daniel Jurafsky, Rebecca Starr, and Su-Youn Yoon. 2005. Accent detection and speech recognition for Shanghai-accented Mandarin. In *Proc. Interspeech 2005*.

A Reproducibility details of models

We use the pre-trained AllenNLP implementations of BiDAF and BiDAF-ELMo[18] (Gardner et al., 2018) and the HuggingFace implementation of BERT.[19] We fine-tune BERT and RoBERTa on SQuAD with a learning rate of $3e{-}5$ for 2 epochs, with a maximum sequence length of 384. All models achieve good performance on the SQuAD dataset. Our trained models achieve the following F1 scores on SQuAD development set: BiDAF: 77.82, BiDAF-ELMo: 80.68, BERT: 88.75, RoBERTa: 89.93.

B Keyboard noise in the wild

Common examples of keyboard typos include replacing a character with the one corresponding to an adjacent key *(frame→framd)*, inserting or deleting characters *(between→betwen, agency→agenchy)*, and swapping adjacent characters within words *(beroids→beriods)*. Such errors exist even in textual QA datasets collected in relatively controlled settings: for example, all the error examples above actually occur in SQuAD. In a real-life situation of information need, where the user produces the question without being exposed to the context and the answer, these errors will likely be even more pervasive. We qualitatively analyze a sample from a dataset of questions collected from the Yahoo! Answers platform (Miao et al., 2010), randomly selecting 50 questions from each topic (Science, Internet, and Hardware). We manually identify non-standard spellings and discard ones that are intentional, such as slang *(thanks→thanx)* or expression of emotion *(so→sooo)*. Since we are specifically interested in the errors that happen in the process of typing, we also separate out errors that could have originated in the user's mind; for example, the most frequent class of errors is omission or insertion of apostrophes in contractions, possessives and plurals, but all of them could plausibly be explained by the user's intention. Other common error types we find are incorrect whitespace placement and character substitutions (mostly plausible human errors), and character insertions, deletions or swapping adjacent characters within words (mostly interface errors); statistics and error examples can be found in Table 7.

Error type	Examples	#Errors
Apostrophe	*it's→its, devices→device's*	55 (0)
Whitespace	*anyone→any one, a lot→alot*	18 (4)
Deletion	*too→to, school→schol*	18 (10)
Substitution	*warranty→warrenty, will→well*	12 (2)
AdjSwap	*type→tpye, piece→peice*	11 (9)
Insertion	*answer→asnswer, lose→loose*	9 (5)
KeySwap	*of→if*	1 (1)

Table 7: Examples of common error types observed in a manual analysis of the Yahoo! Answers questions. Examples identified as interface errors are highlighted in blue. #Errors is total number of typographical errors, with # of interface errors in parentheses.

C Filtering interface misspellings

Our source of human keyboard errors is the Wikipedia list of common English misspellings; some of them are likely to occur in the process of typing (e.g. *and→adn*), while others can plausibly be explained by user misconception (e.g. *recieve→receive*). Since our work focuses on interface errors specifically, we would like to only retain errors from the former category.

Our filtering approach is based on two assumptions: (a) interface errors must be plausible under the keyboard layout, and (b) misspellings that preserve pronunciation of the original word (e.g. *article→artical*) are more likely to be non-interface errors coming from users themselves. We use a two-step filtering heuristic: first, we retain only error categories likely to be explained by the interface noise (character deletion and insertion, adjacent character swap or adjacent key swap in QWERTY layout), and then discard spellings with similar pronunciations. Pronunciations are obtained via the Epitran G2P system (Mortensen et al., 2018), and similarity is determined by weighted edit distance.

On a sample of 100 Wikipedia misspellings manually labeled as interface or non-interface errors, the proposed heuristic shows 83% agreement with human annotation. Applying the heuristic to the initial 4,518 word–spelling pairs, we obtain a set of 1,742 interface errors for 1,489 English words.

D Voice variation in ASR

This section describes the details of the voice variation experiments discussed in §4.1. The numbers used to generate Figures 1a and 1b are presented in Tables 8 and 10 respectively.

---

[18] github.com/allenai/allennlp-hub

[19] github.com/huggingface/pytorch-transformers

Synthetic variation We generate the synthetic voices using Google English Text-to-Speech system with four different accent settings (Australian, British, Indian, and US) and two gender settings (male and female voices). The performance of all models on these voices is presented in Table 8. All QA models achieve highest F1 score when the questions are voiced with a US accent, which is likely explained by the ASR component being optimized for this accent specifically. Neither gender setting consistently leads to best performance across all models and accents. BiDAF and RoBERTa achieve highest scores when the US female synthetic voice is used, and BiDAF-ELMo and BERT perform best with the US male synthetic voice.

Natural variation We record the spoken versions of the 1,190 XQuAD questions voiced by three human annotators: H1 (Indian female), H2 (Russian female), and H3 (Indian male). The same three annotators and an additionally recruited annotator H4 (Scottish male) also voiced the same random sample of 100 XQuAD questions to measure the effect of voice variation in content-controlled setting. The summary statistics (mean and standard deviation) for the sample of speakers are shown in Figure 1b, and the breakdown of each model's score by speaker is presented in Table 10. To collect a set of recordings that is more representative of the real-life use cases, we do not control for recording conditions and other confounds, so our per-speaker results alone are not meant to be taken as evidence of the ASR or QA models being better-tuned for any of the mentioned demographics.

E Input language variation in MT

Table 9 presents the results of the query language variation experiment (§4.1, Figure 2). In this experiment, we use XQuAD human translation of questions into ten languages as inputs, translating them back into English through the Google Translation API. The table also reports the results on the original English SQuAD questions to serve as a skyline. As expected, lower-resource languages and languages that are more typologically divergent from English (the QA system's language) pose the biggest challenge for the MT–QA pipeline.

F Robustness experiments

Table 11 presents the question repair and data augmentation results on both synthetic and natural noise for all interfaces. Synthetic noise sets were used for development and tuning in all experiments. Table 11 also breaks down data augmentation results by the specific augmentation noise source. Training on ASR noise proves helpful for natural keyboard noise as well as natural ASR noise, and robustness to natural translation noise is only improved by augmenting the data with its synthetic counterpart.

G ASR system benchmarking

To benchmark both the ESPnet CommonVoice ASR system, which we use for data augmentation, and the Google ASR, which was used to create ASR challenge sets from recorded XQuAD questions, we also transcribe the natural and synthetic challenge set recordings with ESPnet ASR. ESPnet achieves 56.8% and 70.1% WER for synthetic and natural voices respectively, while Google ASR gets a WER of 16.6% and 30.7% respectively (Table 3).

H Numeral handling and ASR interfaces

Correctly transcribing numerals is often important for producing a correct answer in an ASR–QA pipeline. Even a different representation of the same quantity in the question and in the context passage creates additional difficulties for the QA system. To additionally analyze the effect of handling numerals in ASR engines, we combine BERT with Kaldi (Povey et al., 2011) or Google speech recognizers and compare their performance on the portion of XQuAD questions containing numerals (XQUAD-NUMBERS) and the remaining questions (XQUAD-NONUM). With the questions narrated by human annotators, the QA pipeline performs worse on XQUAD-NUMBERS than XQUAD-NONUM with either Kaldi (38.39 F1 and 44.30 F1 respectively) or Google ASR (64.44 F1 and 70.86 F1 respectively). In case of Kaldi, we hypothesize that the discrepancy might be partially explained by the speech recognizer outputting numbers in their spelled-out form rather than numeric form. To test this hypothesis, we convert all numerals in the original written XQUAD-NUMBERS questions into their spelled-out form and observe a drop in performance from 87.10 F1 to 82.88 F1 on this subset. However, the representation mismatch is only one of many challenges: unlike Kaldi, Google ASR outputs numerals as digits, but the corresponding pipeline still shows worse performance on spoken XQUAD-NUMBERS.

Model	AU		GB		IN		US	
	Female	Male	Female	Male	Female	Male	Female	Male
BiDAF (Seo et al., 2017)	64.14	64.76	60.45	63.73	64.09	64.80	65.93	66.39
BiDAF-ELMo (Peters et al., 2018)	67.84	67.49	65.08	67.04	68.13	68.94	70.50	70.30
BERT (Devlin et al., 2019)	74.54	73.87	70.56	72.79	73.65	74.47	77.42	77.02
RoBERTa (Liu et al., 2019)	78.86	78.79	76.37	78.27	78.83	80.13	81.11	81.38

Table 8: Performance of different QA models in the TTS-ASR pipeline with different synthetic voices. We use Google Text-to-Speech with different accent and gender settings, and Google Speech-to-Text optimized for English–US as the speech recognizer.

Model	en	es	hi	vi	de	ar	zh	el	ru	th	tr
BERT	84.66	79.86	76.75	77.14	79.98	75.45	76.39	76.96	78.06	71.03	76.98
RoBERTa	84.42	81.65	79.61	78.77	82.13	76.41	78.88	79.6	79.67	74.68	79.28

Table 9: QA performance on XQuAD human translations of SQuAD questions in different source languages posed to an English QA system. Questions in each non-English language are translated to English using the Google MT system, and the performance on the original English questions is reported as a skyline.

Model	H1	H2	H3	H4
BiDAF	58.14	62.86	31.60	60.07
BiDAF-ELMo	56.15	62.65	29.30	62.48
BERT	59.77	67.27	32.98	65.63
RoBERTa	60.74	74.31	34.14	67.48

Table 10: Performance of the different QA models on different human annotator voices: Indian Female (H1), Russian Female (H2), Indian Male (H3), and Scottish Male (H4). We do not control for recording conditions and other confounds in this experiment, so our results are not meant to act as evidence of ASR systems being more effective for any particular demographic.

BERT Model	XQuAD_{EN}		ASR		MT		Keyboard	
	EM	F1	EM	F1	EM	F1	EM	F1
Synthetic								
BERT	**72.77**	**84.66**	61.93	77.02	**67.23**	79.08	61.68	74.43
+ NE Repair	72.94	84.78	62.10	77.23	67.31	79.19	63.78	75.31
+ Content Repair	72.94	84.77	62.02	77.12	67.31	79.14	62.61	74.34
+ Spelling Augmentation	72.35	83.89	56.81	73.68	65.63	78.09	**67.31**	**78.83**
+ ASR Augmentation	71.93	83.41	**66.13**	78.29	66.13	78.29	65.46	76.65
+ Translation Augmentation	70.76	83.17	61.09	76.42	66.72	**79.70**	59.83	72.29
+ Spelling+ASR+Translation Augmentation	67.48	80.64	**66.13**	**79.82**	64.20	77.18	64.28	77.63
Natural								
BERT	72.77	84.66	52.94	67.13	68.82	79.98	69.16	81.84
+ NE repair	**72.94**	**84.78**	53.03	67.34	68.82	80.05	69.58	82.22
+ Content repair	**72.94**	84.77	52.61	67.01	68.32	79.76	70.25	**82.60**
+ Spelling Augmentation	72.35	83.89	50.84	66.04	68.49	80.20	70.25	82.22
+ ASR Augmentation	71.93	83.41	**64.37**	**75.89**	68.65	80.32	**70.76**	82.43
+ Translation Augmentation	70.76	83.17	53.70	68.11	**68.90**	**80.83**	68.57	81.05
+ Spelling+ASR+Translation Augmentation	67.48	80.64	62.02	74.61	66.81	80.25	65.88	78.55

Table 11: Effect of question repair and data augmentation on BERT performance on both synthetic and natural noise for the three interface types. Data augmentation results are presented separately for each source of training synthetic noise. Synthetic noise sets are used for development and tuning in all experiments.

Co-evolution of language and agents in referential games

Gautier Dagan
University of Amsterdam
gautierdagan@gmail.com

Dieuwke Hupkes
Facebook AI Research
dieuwkehupkes@fb.com

Elia Bruni
University of Osnabrück
elia.bruni@gmail.com

Abstract

Referential games offer a grounded learning environment for neural agents which accounts for the fact that language is functionally used to communicate. However, they do not take into account a second constraint considered to be fundamental for the shape of human language: that it must be learnable by new language learners.

Cogswell et al. (2019) introduced *cultural transmission* within referential games through a changing population of agents to constrain the emerging language to be learnable. However, the resulting languages remain inherently biased by the agents' underlying capabilities.

In this work, we introduce Language Transmission Simulator to model both cultural and architectural evolution in a population of agents. As our core contribution, we empirically show that the optimal situation is to take into account also the learning biases of the language learners and thus let language and agents *co-evolve*. When we allow the agent population to evolve through architectural evolution, we achieve across the board improvements on all considered metrics and surpass the gains made with *cultural transmission*. These results stress the importance of studying the underlying agent architecture and pave the way to investigate the *co-evolution* of language and agent in language emergence studies.

1 Introduction

Human languages show a remarkable degree of structure and complexity. In the evolution of this complex structure, several different intertwined *pressures* are assumed to have played a role. The first of these pressures concerns the function of language: as language is to communicate, it should allow effective communication between proficient language users (e.g. Smith and Kirby, 2012). This pressure is strongly intertwined with the nature of the proficient user: what features of language allow effective communication depends on the abilities of the user to use the language.

A second pressure on the shape of human language stems from the fact that language must be *learnable*. Unlike animal languages, which are taken to be mostly innate, human languages must be re-acquired by each individual (Pinker and Bloom, 1990; Hurford, 1998). A language can only survive if it can successfully be transmitted to a next generation of learners. In the field of language evolution, this transmission process is referred to as *cultural transmission*, while the process of change that occurs as a consequence is called *cultural evolution*.[1] Like the pressures arising from the function of language, the way that cultural evolution shapes language also depends on the language users: what is learnable depends on the inductive biases of the learner.

Computationally, the emergence of language can be studied through simulation with artificial agents and by investigating the resulting languages for structure, level of compositionality, and morphosyntactic properties (Kirby, 2001; Kirby and Hurford, 2002). Originally based on logic and symbolic representations (Kirby, 2001; Christiansen and Kirby, 2003), with the advent of modern deep learning methods, there has been a renewed interest in simulating the emergence of language through neural network agents (i.a. Lazaridou et al., 2017;

---

[1] The importance of cultural evolution for the emergence of structure is supported by a number of artificial language learning studies (e.g. Saldana et al., 2018) and computational studies using the Iterated Learning paradigm, in which agents learn a language by observing the output produced by another agent from the previous 'generation' (e.g. Kalish et al., 2007; Kirby et al., 2008, 2015).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2993–3004
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Havrylov and Titov, 2017). Such work typically involves the use of *referential games* (Lewis, 1969), in which two or more agents have to emerge a language to obtain a shared reward.

These studies are motivated by the first pressure: language as a tool for effective communication. However, they fail to consider the second pressure: language must be learnable by new agents. They also fail to study the impact of the learning biases of the artificial agents themselves, which underlies both pressures. In a recent study, Cogswell et al. (2019) proposed a method to include cultural evolution in a language emergence game. Their approach is more naturally aligned with pressures in humans language evolution than single agent referential games (see e.g. Wray and Grace, 2007), but fails to account for the fact that cultural evolution and the learning biases of the artificial agents are two sides of the same coin: what language is learnable depends on the learning biases of the learner.

In this paper, we will therefore integrate the three components described above – communication, cultural evolution and learning biases – and setup a framework in which their interaction can be studied. This framework, which we refer to with the term Language Transmission Simulator, consists of *a referential game*, played by *a changing population of agents* – simulating cultural evolution – which are subject to *architectural evolution* – simulating the learning biases of the learners and allowing them to *co-evolve* with the language.

Our contributions are three-fold fold:

- We introduce the *Language Transmission Simulator*, that allows to model both cultural and architectural evolution in a population of agents;
- We collect a large number of tests from previous work and combine them into an extensive test suits for language emergence games;
- We demonstrate that emerging languages benefit from including cultural transmission *as well as* architectural evolution, but the best results are achieved when languages and agents can co-evolve.

2 Related Work

Much work has been done on the emergence of language in artificial agents and investigating its subsequent structure, compositionality and morphosyntax (Kirby, 2001; Kirby and Hurford, 2002). Originally, such work was based on logic and symbolic representations (Kirby, 2001; Christiansen and Kirby, 2003), but with the advent of modern deep learning (LeCun et al., 2015), there has been a renewed interest in simulating the emergence of language through neural network agents (i.a. Foerster et al., 2016; Kottur et al., 2017; Choi et al., 2018; Lazaridou et al., 2017; Havrylov and Titov, 2017; Mordatch and Abbeel, 2018). In the exploration of language emergence, different training approaches and tasks have been proposed to encourage agents to learn and develop communication. In a typical setup, two players aim to develop a communication protocol in which one agent must communicate information it has access to (typically an image), while the other must guess it out of a line-up (Evtimova et al., 2018; Lazaridou et al., 2017).

Kottur et al. (2017) show that 'natural' language does not arise naturally in these communication games and it has to be incentivised by imposing specific restrictions on games and agents. Havrylov and Titov (2017) first demonstrated that using straight-through estimators were more effective than reinforcement learning in a collaborative task, and that optimizing rewards can lead to structured protocols (i.e. strings of symbols) to be induced from scratch. Mordatch and Abbeel (2018) find that syntactic structure emerges in the stream of symbol uttered by agents, where symbols and syntax can be mapped to specific meanings or instructions. Choi et al. (2018) use qualitative analysis, visualization and a zero-shot test, to show that a language with compositional properties can emerge from environmental pressures.

Chaabouni et al. (2019) find that emerged languages, unlike human languages, do not naturally prefer non-redundant encodings. Chaabouni et al. (2020) further find that while generalization capabilities can be found in the languages, compositionality itself does not arise from simple generalization pressures. (Rodríguez Luna et al., 2020) encourage desirable properties of human languages, such as compositionality, to emerge through well-crafted auxiliary pressures. Finally, Harding Graesser et al. (2019) demonstrate with experiments on populations of agents that language can evolve and simplify through the interactions of different communities.

Cogswell et al. (2019) build upon the emergent language research by introducing cultural transmission as a pressure in referential games. They use a pool of agents with a resetting mechanism and show that this further encourages the emerging language to display compositional properties and structure allowing it to generalize better. Pairing agents with one another in a larger population setting introduces cultural evolution, but it is the pressure introduced by the partial resetting which forces remaining agents to emerge a language that is quickly learnable by a new agent. While Cogswell et al. (2019) is the most related work to ours, an important difference is that they focus on cultural evolution only, without taking into account the learning biases of the agents via modelling architectural evolution.

3 Approach

In this paper, we introduce architectural evolution in language emergence games and study the interaction between cultural and architectural evolution with a range of different metrics. Below, we first give a definition of the referential game we consider (Subsection 3.1). We then briefly explain our Language Transmission Simulator (Section 3.2 and how we model cultural and architectural evolution within it (Section 3.3 and 3.4 respectively).

3.1 Referential games

We study language emergence in a referential game inspired by the signalling games proposed by Lewis (1969). In this game, one agent (the *sender*) observes an image and generates a discrete message. The other agent (the *receiver*) uses the message to select the right image from a set of images containing both the sender image and several distractor images. Since the information shown to the sender agent is crucial to the receivers success, this setup urges the two agents to come up with a communication protocol that conveys the right information.

Formally, our *referential* game is similar to Havrylov and Titov (2017):

1. The meaning space of the game consists of a collection D of K images $\{d_0, d_1, ..., d_K\}$, represented by z-dimensional feature vectors.
2. In each round i of the game, a target item d_i is randomly sampled from D, along with a set C of n distractor items.
3. The sender agent s of the game, parametrised by a neural network, is given item d_i, and generates a discrete message m_i from a vocabulary V. The message is capped to a max message length of L.
4. The receiver agent r, also parametrised by a neural network, receives message m_i and uses it to identify d_i in the union of d_i and C.

We use $z = 512$, and $n = 3$ and train agents with Gumbel-Softmax (Jang et al., 2017a) based on task-success.

3.2 Language Transmission Simulator

In Language Transmission Simulator, depicted in Figure 1, we simulate a population of communicating agents. In every training iteration, two random agents are sampled to play the game. This forces the agents to adopt a simpler language naturally: to succeed they must be able to communicate or understand all opposing agents. In our setup, agents are either sender or receiver, they do not switch roles during their lifetime.

3.3 Cultural evolution in referential games

Following Cogswell et al. (2019), we simulate cultural evolution by periodically replacing agents in the population with newly initialised agents. Cultural evolution is implicitly modelled in this setup, as new agents have to learn to communicate with agents that already master the task. Following Cogswell et al. (2019), we experiment with three different methods to select the agents that are replaced: randomly (no selection pressure), replacing the oldest agents or replacing the agents with the lowest fitness (as defined in Section 3.5). We call these setups `cu-random`, `cu-age` and `cu-best`, respectively.

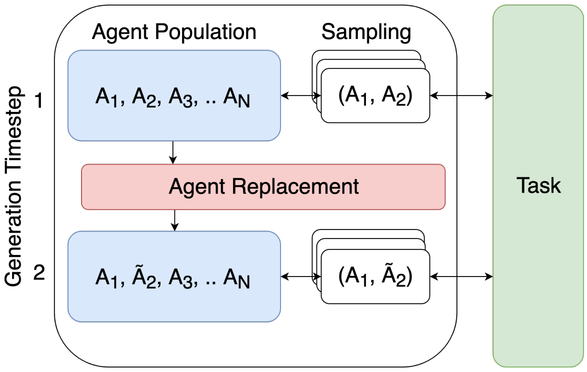


Figure 1: The Language Transmission Simulator: Agent pairs are randomly sampled from each population and trained. After l training steps, a portion α of the population is culled.

3.4 Architectural evolution in referential games

To model architectural evolution, rather than periodically replacing agents with randomly initialised new agents, we instead mutate the most successful agents and replace the worst agents with variations of the best agents, as outlined in Section 3.4.2. Note that cultural evolution is still implicitly modelled in this setup, as new agents still have to learn to communicate with older agents. Therefore, we call this setup with the term `co-evolution`.

3.4.1 Culling

We refer to the selection process and subsequent mutation or re-initialisation step as *culling*. In biology, culling is the process of artificially removing organisms from a group to promote certain characteristics, so, in this case, culling consists of removing a subset of the worst agents and replacing them with variations of the best architecture. The proportion of agents from each population selected to be mutated is determined by the culling rate α, where $\alpha \in [0, 1)$. The culling interval l defines the number of iterations between culling steps. A formalisation of the LTE can be found in appendix A.1.

3.4.2 Mutation Algorithm

Our mutation algorithm is an intentionally simple implementation of a Neural Architectural Search (NAS). NAS focuses on searching the architecture space of networks, unlike many traditional evolutionary techniques which often include parameter weights in their search space. We opted to use the DARTS (Differentiable Architecture Search) RNN search space defined by Liu et al. (2018), which has obtained state-of-the-art performance on benchmark natural language tasks (Li and Talwalkar, 2019).

The DARTS search space includes recurrent cells with up to N nodes, where each node $n_1, n_2, ..., n_N$ can take the output of any preceding nodes including n_0, which represents the cell's input. All potential connections are modulated by an activation function, which can be the identity function, Tanh, Sigmoid or ReLU. Following Liu et al. (2018) and Pham et al. (2018), we enhance each operation with a highway bypass (Zilly et al., 2016) and the average of all intermediate nodes is treated as the cell output.

To sample the initial model, we sample a random cell with a single node ($N = 1$). As this node must necessarily be connected to the input, the only variation stems from the possible activation functions applied to the output of n_1, resulting in four possible starting configurations. We set a node cap of $N = 8$. We mutate cells by randomly sampling an architecture which is one edit step away from the previous architecture. Edit steps are uniformly sampled from i) changing an incoming connection, ii) changing an output operation or iii) adding a new node; the mutation location is uniformly sample from all possible mutations.[2]

3.5 Fitness Criterion

The fitness criterion that we use in both the `cu-best` and `co-evolution` setup is based on task performance. However, rather than considering agents' performance right before the culling step, we consider the age of the youngest agent in the population (defined in terms of number of batches that it was trained) and for every agent compute their performance up until *when they had that age*. For any agent a_j in population $\mathbf{A}$ this is defined as:

$$\text{fitness}(a_j) = \frac{1}{\mathcal{T}_A} \sum_{t=0}^{\mathcal{T}_A} \mathcal{L}(a_j^t) \qquad (1)$$

where $\mathcal{T}_A = \min_{a \in A} \mathcal{T}(a)$ is the age $\mathcal{T}(a)$ of the youngest agent in the population, and $\mathcal{L}(a_j^t)$ is the loss of agent a_j at time step t. This fitness criterion is not biased towards older agents, that have seem already more data and have simply converged more. It is thus not only considering task performance but also the *speed* at which this performance is reached.

4 Experiments

We test the LTE framework on a compositionally defined image dataset, using a range of different selection mechanisms.

4.1 Dataset

In all our experiments, we use a modified version of the Shapes dataset (Andreas et al., 2015), which consists of 30 by 30 pixel images of 2D objects, characterised by shape (circle, square, triangle), colour (red, green, blue), and size (small, big). While every image has a unique symbolic description – consisting of the shape, colour and size of the object and its horizontal and vertical position in a 3x3 grid – one symbolic representation maps to multiple images, that differ in terms of exact pixels and object location. We use 80k, 8k, 40k images for train, validation and test sets, respectively. Some example images are depicted in Figure 2.

---

[2] For a formal description of the mutation process, we refer to Appendix A.2.

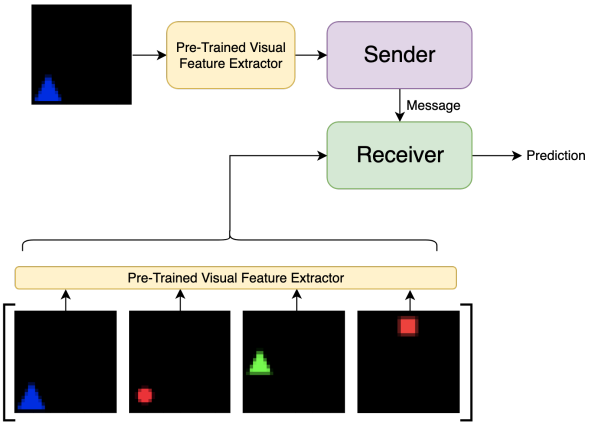


Figure 2: The modified Shapes task consists of showing an image the sender, and then letting the receiver deduce from the sender's message which image out of the target and k distractors is the correct one.

We pre-train a CNN feature extractor for the images in a two-agent setting of the task (see Appendix A.4 for more details).

4.2 Architecture and Training

For our co-evolution experiments, we use the DARTS search space as described above. For all cultural evolution approaches, we use an LSTM (Hochreiter and Schmidhuber, 1997) for both the sender and receiver architecture (see Appendix A.3 for more details). The sender and receiver models have a hidden size of 64 for the recurrent layer and an embedding layer of size 64. Further, we use a vocabulary size V of 4, with an additional bound token serving as the indicator for beginning and end-of-sequence. We limit the maximum length of a sentence L to 5. [3]

We back-propagate gradients through the discrete step outputs (message) of the sender by using the Straight-Through (ST) Gumbel-Softmax Estimator (Jang et al., 2017b). We run all experiments with a fixed temperature $\tau = 1.2$. We use the default Pytorch (Paszke et al., 2017) Adam (Kingma and Ba, 2015) optimiser with a learning rate of 0.001 and a batch-size of 1024. Note that the optimiser is reset for every batch.

For all multi-agent experiments we use a population size of 16 senders and 16 receivers. The culling rate α is set to 0.25 or four agents, and we cull (re-initialise or mutate) every $l = 5k$ iterations. We run the experiments for a total of $I = 500k$ iterations, and evaluate the populations before each culling step.

4.3 Evaluation

We use an range of metrics to evaluate both the population of agents and the emerging languages.

Jaccard Similarity We measure the consistency of the emerged languages throughout the population using *Jaccard Similarity*, which is defined as the ratio between the size of the intersection and the union of two sets. We sample 200 messages per input image for each possible sender-receiver pair and average the Jaccard Similarity of the samples over the population. A high Jaccard Similarity between two messages is an indication that the same tokens are used in both messages.

Proportion of Unique Matches We compute how similar the messages that different agents emit for the same inputs by looking at all possible (sender, message) pairs for one input and assess whether they are the same. This metric is 1 when all agents always emit the same messages for the same inputs.

Number of Unique Messages We compute the average *number of unique messages* generated by each sender in the population. An intuitive reference point for this metric is the number of images with distinct symbolic representations. If agents generate more messages than expected by this reference point, this demonstrates that they use multiple messages for the images that are – from a task perspective – identical.

Topographic Similarity Topographic similarity, used in a similar context by Lazaridou et al. (2018), represents the similarity between the meaning space (defined by the symbolic representations) and the signal space (the messages sent by an agent). It is defined as the correlation between the distances between pairs in meaning space and the distances between the corresponding messages in the signal space. We compute the topographic similarity for an agent by sampling 5,000 pairs of symbolic inputs and corresponding messages and compute the Pearson's ρ correlation between the cosine similarity of the one-hot encoded symbolic input pairs and the cosine similarity of the one-hot encoded message pairs.

---

[3] The values for V and L were picked to provide a strong communication bottleneck to promote the emergence of structured and compressed languages, following the intuitions from Kottur et al. (2017) that natural language patterns do not emerge 'naturally'.

Average Population Convergence To estimate the speed of learning of the agents in the population, estimate the *average population convergence*. For each agent, at each point in time, this is defined as the agents average performance from the time it was born until it had the age of the current youngest agent in the population (analogous to the fitness criterion defined in Section 3.5). To get the average population convergence, we take we average those values for all agents in the population.

Average Agent Entropy We compute the average certainty of sender agents in their generation process by computing and averaging their *entropy* during generation.

5 Results

We now present a detailed comparison of our cultural and co-evolution setups. For each approach, we averaged over four random seeds, the error bars in all plots represent the standard deviation across these four runs. To analyse the evolution of both agents and languages, we consider the development of all previously outlined metrics over time. We then test the best converged languages and architectures in a single sender-receiver setup, to assess the impact of cultural and genetic evolution more independently. In these experiments, we compare also directly to a single sender-receiver baseline, which is impossible for most of the metrics we consider in this paper. Finally, we briefly consider the emerged architectures from a qualitative perspective.

5.1 Task performance

We first confirm that all setups in fact converge to a solution to the task. As can be seen in Figure 3, all populations converge to a (close to perfect) solution to the game. The `cu-age` approach slightly outperforms the other approaches, with a accuracy that surpasses the 95% accuracy mark. Note that, due to the ever changing population, the accuracy at any point in time is an average of both 'children' and 'adults', that communicate with different members of the population.

6 Analysis

In this section we analyse the resulting behaviour and success of agents in Language Transmission Simulator. We first use standard approaches such as average agent entropy and loss (convergence) to measure the success of agents with respect to their language and the task. Secondly, we use other metrics to analyse the emergent language itself in terms of consistency and diversity by using Jaccard Similarity, the proportion of unique matches, the number of unique messages, and the topographic similarity. Thirdly, we perform a qualitative analysis of the architecture that emerge from our Language Transmission Simulator. Finally, we design Frozen Experiments, in which we test the emerged languages and architectures in a 1v1 setting with a fresh agent. This allows us to compare and measure the the improvement gains made by the architecture and those made by the language which emerged. We show through these experiments that the co-evolution setting leads to a language that is both more successful and easier to learn for a given new agent.

Figure 3: Average Population Accuracy of final populations.

6.1 Agent behaviour

To assess the behaviour of the agents over time, we monitor their average message entropy convergence speed. As can be seen in Figure 4, the `co-evolution` setup results in the lowest average entropy scores, the messages that they assign to one particular image will thus have lower variation than in the other setups. Of the cultural evolution setups, the lowest entropy score is achieved in the `cu-best` setup.

Figure 5 shows the average population convergence over time. Also in this case, we observe a clear difference between cultural evolution only and co-evolution, with an immediately much lower convergence time for co-evolution and a slightly downward trending curve.

6.2 Language Analysis

To check the consistencies of languages within a population, we compare the Jaccard Similarity and the Average Proportion of Unique Matches, which we plot in Figure 6. This shows that, compared to cultural evolution only, not only are the messages

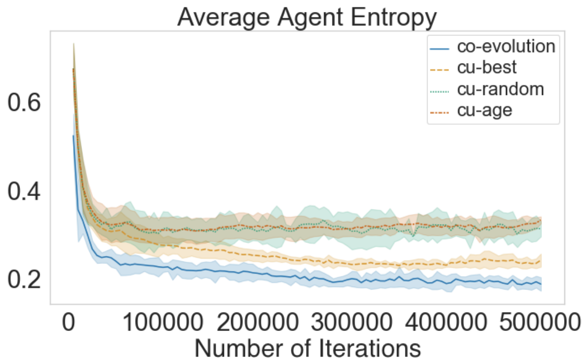


Figure 4: Average agent entropy over time.

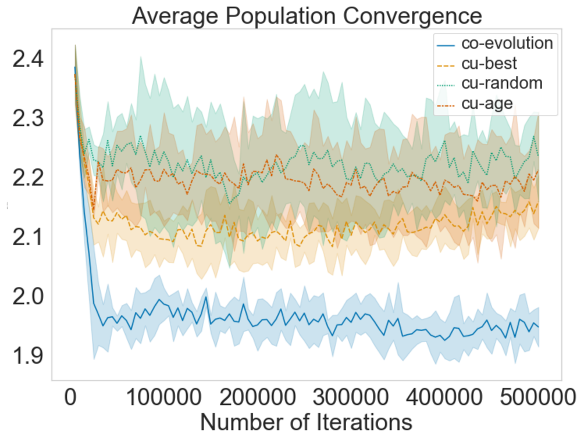


Figure 5: Average convergence for all cultural transmission modes and evolution.

in co-evolution more similar across agents (higher Jaccard Similarity), but also that agents are considerably more aligned with respect to the same inputs (less unique matches).

To assess the level of structure of the emerged languages, we plot the average Topographic Similarity and the Average Number of Unique Messages generated by all senders (Figure 7). The co-evolution condition again outperforms all cultural only conditions, with a simpler language (the number of the unique messages closer to the symbolic reference point) that is structurally more similar to the symbolic representation of the input (higher Topographical Similarity).

6.3 Architecture Analysis

In Figure 8 we show the co-evolution of an agent and a sample of its language during three selected iterations in the co-evolution setup. Strikingly, the best sender architecture does not evolve from its original form, which could point towards the limitations of of our search strategy and space. On

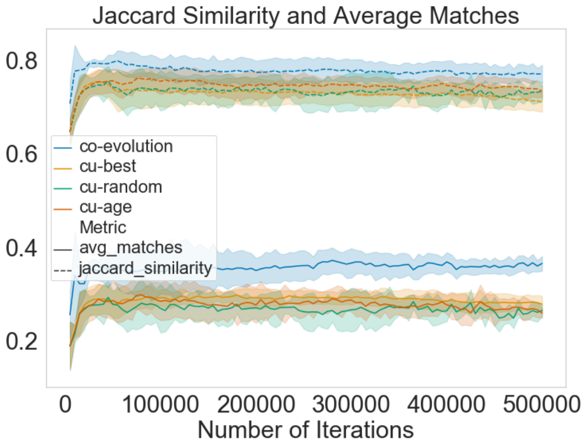


Figure 6: Average Jaccard Similarity and proportion of message matches for all cultural transmission modes and evolution

the contrary, the receiver goes through quite some evolution steps and converges into a significantly more complex architecture than its original form. We observe a unification of language throughout evolution in Figure 8, which is also supported by Figure 7. The population of senders starts out 11 different unique messages and ends with only two to describe the same input image. We will leave more detailed analysis of the evolved architectures for future work.

6.4 Frozen Experiments

With a series of experiments we test the a priori suitability of the evolved languages and agents for the task at hand, by monitoring the accuracy of new agents that are paired with converged agents and train them from scratch.

We focus, in particular, on training receivers with a frozen sender from different setups, which allows us to assess 1) whether cultural evolution made languages evolve to be more easily picked up by new agents 2) whether the genetic evolution made architectures converge more quickly when faced with this task. We compare the accuracy development of:

- An LSTM receiver trained with a frozen sender taken from `cu-best`;
- An evolved receiver trained with a frozen evolved sender.

For both these experiments, we compare with two baselines:

- The performance of a receiver agent trained from scratch along with a receiver agent

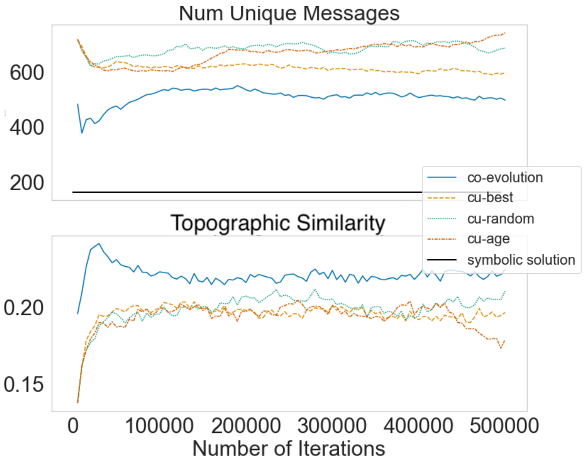


Figure 7: Average Number of Unique Messages and Topographic Similarity for all cultural evolution modes and co-evolution. For comparison, we also plot the number of unique messages for a symbolic solution that fully encodes all relevant features of the image (since we have three possible shapes and colours, two possible sizes, and a 3 × 3 grid of possible positions, this symbolic reference solution has $3 \times 3 \times 2 \times 9 = 162$ distinct messages.

that has either the `cu` architecture or the evolved `co` architecture (`cu-baseline` and `co-baseline`, respectively);

- The performance of an agent trained with an agent that is *pretrained* in the single agent setup, with either the `cu` architecture or an evolved architecture (`cu-baseline-pretrained` and `co-baseline-pretrained`).

Each experiment is run 10 times, keeping the same frozen agent. The results confirm cultural evolution contributes to the learnability and suitability of emerging languages: the `cu-best` accuracy (green line) converges substantially quicker and is substantially higher than the `cu-baseline-pretrained` accuracy (orange line). Selective pressure on the language appears to be important: the resulting languages are only easier to learn in the `cu-best` setup.[4] In addition, they show that the agents benefit also from the genetic evolution: the best accuracies are achieved in the co-evolution setup (red line). The difference between the `cu-baseline` (blue) and the `co-baseline` (brown) further shows that even if the evolved architectures are trained from scratch, they perform much better than a baseline model trained from scratch. The difference between the `co-baseline-pretrained` (only genetic evolution, purple line) and the co-evolution of agents and language line (red line) illustrates that genetic evolution alone is not enough: while a new evolved receiver certainly benefits from learning from a (from scratch) pretrained evolved sender, without the cultural transmission pressure, it's performance is still substantially below a receiver that learns from an evolved sender whose language was evolved as well.

7 Conclusion

In this paper, we introduced a language transmission bottleneck in a referential game, where new agents have to learn the language by playing with more experienced agents. To overcome such bottleneck, we enabled both the cultural evolution of language and the architectural evolution of agents, using a new Language Transmission Simulator. Using a battery of metrics, we monitored their respective impact on communication efficiency, degree of linguistic structure and intra-population language homogeneity. While we could find important differences in between cultural evolution strategies, it is when we included architectural evolution that agents scored best. In a second experiment, we paired new agents with evolved languages and agents and again confirmed that, while cultural evolution makes a language easier to learn, co-evolution leads to the best communication.

In future research, we would like to apply the Language Transmission Simulator on new, more complex tasks and further increase our understanding of the properties of the emerged languages and architectures. Recent research has also found that relaxing the vocabulary size V and sequence length L constraints can lead to greater syntactic structure in emergent languages (van der Wal et al., 2020). We thus hope to investigate further relaxation of hyper-parameters and other neuro-evolution techniques in future work.

Acknowledgments

We would like to thank Angeliki Lazaridou for her helpful discussions and feedback on previous iterations of this work.

---

[4] `cu-age` and `cu-random` are ommitted from the plot for clarity reasons.

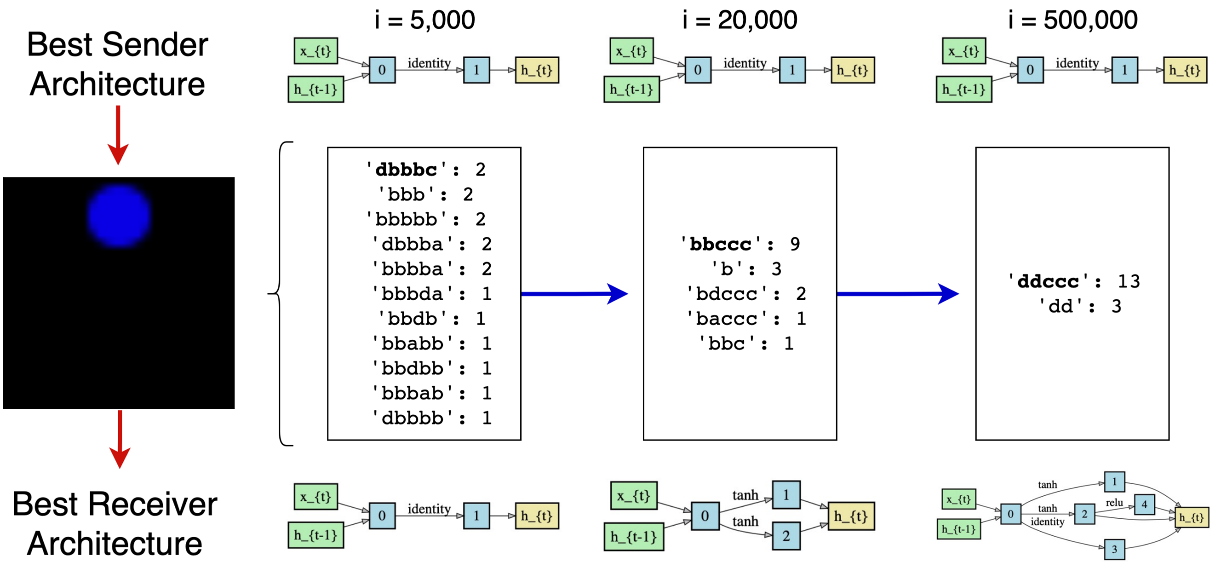


Figure 8: Evolution of the best sender and receiver architecture according to convergence, and the evolution of the population's message description of the same input through iterations. The bold messages represent the message outputted by the best sender whose architecture is pictured above. The count of each message represents the number of agents in the population which uttered this exact sequence.

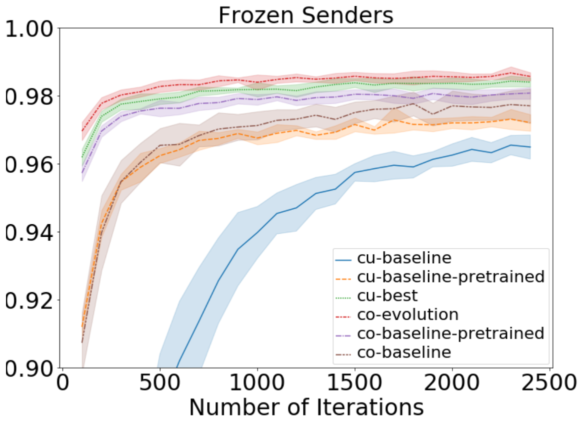


Figure 9: Receiver accuracies trained with different types of frozen senders.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2015. Deep Compositional Question Answering with Neural Module Networks.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. 2020. Compositionality and generalization in emergent languages.

Rahma Chaabouni, Eugene Kharitonov, Alessandro Lazaric, Emmanuel Dupoux, and Marco Baroni. 2019. Word-order biases in deep-agent emergent communication. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5166–5175, Florence, Italy. Association for Computational Linguistics.

Edward Choi, Angeliki Lazaridou, and Nando de Freitas. 2018. Compositional obverter communication learning from raw visual input. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Morten H. Christiansen and Simon Kirby. 2003. Language evolution: consensus and controversies. *Trends in Cognitive Sciences*, 7(7):300–307.

Michael Cogswell, Jiasen Lu, Stefan Lee, Devi Parikh, and Dhruv Batra. 2019. Emergence of compositional language with deep generational transmission. *arXiv preprint arXiv:1904.09067*.

Katrina Evtimova, Andrew Drozdov, Douwe Kiela, and Kyunghyun Cho. 2018. Emergent communication in a multi-modal, multi-step referential game. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning.

Laura Harding Graesser, Kyunghyun Cho, and Douwe Kiela. 2019. Emergent linguistic phenomena in multi-agent communication games. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3700–3710, Hong Kong, China. Association for Computational Linguistics.

Serhii Havrylov and Ivan Titov. 2017. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in neural information processing systems*, pages 2149–2159.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

James R Hurford. 1998. The evolution of language and languages. In *The evolution of culture*. Edinburgh University Press.

Eric Jang, Shixiang Gu, and Ben Poole. 2017a. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017b. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Micheal L. Kalish, Thomas L. Griffiths, and Stephan Lewandowsky. 2007. Iterated learning: Intergenerational knowledge transmission reveals inductive biase. *Psychonomic Bulletin & Review*, pages 288–294.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Simon Kirby. 2001. Spontaneous Evolution of Linguistic Structure-An Iterated Learning Model of the Emergence of Regularity and Irregularity. Technical Report 2.

Simon Kirby, Hannah Cornish, and Kenny Smith. 2008. Cumulative cultural evolution in the laboratory: an experimental approach to the origins of structure in human language. *Proceedings of the National Academy of Sciences of the United States of America*, 105(31):10681–6.

Simon Kirby and James R. Hurford. 2002. The Emergence of Linguistic Structure: An Overview of the Iterated Learning Model. In *Simulating the Evolution of Language*, pages 121–147. Springer London, London.

Simon Kirby, Monica Tamariz, Hannah Cornish, and Kenny Smith. 2015. Compression and communication in the cultural evolution of linguistic structure. *Cognition*, 141:87–102.

Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. 2017. Natural language does not emerge 'naturally' in multi-agent dialog. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2962–2967.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. 2018. Emergence of linguistic communication from referential games with symbolic and pixel input. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. 2017. Multi-agent cooperation and the emergence of (natural) language. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

David Lewis. 1969. *Convention: A Philosophical Study*. Wiley-Blackwell.

Liam Li and Ameet Talwalkar. 2019. Random search and reproducibility for neural architecture search. *CoRR*, abs/1902.07638.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: differentiable architecture search. *CoRR*, abs/1806.09055.

Igor Mordatch and Pieter Abbeel. 2018. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1495–1502.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. 2018. Efficient neural architecture search via parameter sharing. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 4092–4101.

Steven Pinker and Paul Bloom. 1990. Natural language and natural selection. *Behavioral and Brain Sciences*, 13(4):707–27.

Diana Rodríguez Luna, Edoardo Maria Ponti, Dieuwke Hupkes, and Elia Bruni. 2020. Internal and external pressures on language emergence: least effort, object constancy and frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4428–4437, Online. Association for Computational Linguistics.

Carmen Saldana, Simon Kirby, Rob Truswell, and Kenny Smith. 2018. Compositional hierarchical structure evolves through cultural transmission: an experimental study.

Kenny Smith and Simon Kirby. 2012. *Compositionality and Linguistic Evolution*. Oxford University Press.

Oskar van der Wal, Silvan de Boer, Elia Bruni, and Dieuwke Hupkes. 2020. The grammar of emergent languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3339–3359, Online. Association for Computational Linguistics.

Alison Wray and George W. Grace. 2007. The consequences of talking to strangers: Evolutionary corollaries of socio-cultural influences on linguistic form. *Lingua*, 117(3):543–578.

Julian G. Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. 2016. Recurrent highway networks. *CoRR*, abs/1607.03474.

A Appendix

A.1 Language Transmission Engine

We formalise our Language Transmission process in the pseudo code shown in Algorithm 1. We select hyper-parameters l as the number of iterations or batches shown between culling steps, and I as the total number of iterations.

Algorithm 1 Language Transmission Engine

```
$S \leftarrow \{s_0, s_1 ..., s_N\}$
$R \leftarrow \{r_0, r_1 ..., r_N\}$
$i \leftarrow 1$
while $i \leq I$ do
    for batch $b$ in $D$ do
        Sample $\hat{s}$ from $S$
        Sample $\hat{r}$ from $R$
        $\mathbf{train}(\hat{s}, \hat{r}, b)$
        if $i \bmod l = 0$ then
            $\mathbf{cull}(S, R)$
        end if
        $i \leftarrow i + 1$
    end for
end while
```

A.2 Mutation Algorithms Pseudo-code

The genotype mutation is described in pseudo-code by algorithm 2, and takes as input a genotype containing nodes describing the cell. The genotype is mutated by either changing the input connection or primitive (output activation function) for a randomly sampled node n, or adding a new node altogether. See section 3.4.2 for explanations on the workings of the DARTS cell structure.

Algorithm 2 Genotype-level Mutation

```
procedure $\mathbf{mg}(genotype)$
    $g \leftarrow copy(genotype)$
    $a \leftarrow \mathcal{U}(1, 3)$
    $n \leftarrow \mathcal{U}(1, len(g))$
    if $a = 1$ then
        $p \leftarrow \mathcal{U}[ReLU, I, \tanh, \sigma]$
        $n.activation \leftarrow p$
    end if
    if $a = 2$ then
        $r \leftarrow \mathcal{U}(1, n)$
        $n.connection \leftarrow r$
    end if
    if $a = 3$ then
        $n' \leftarrow new\_node()$
        $p \leftarrow \mathcal{U}[ReLU, I, \tanh, \sigma]$
        $r \leftarrow \mathcal{U}(1, len(g))$
        $n'.activation \leftarrow p$
        $n'.connection \leftarrow r$
        $g.append(n')$
    end if
    return $g$
end procedure
```

Algorithm 3 Population-level Mutation

```
procedure $\mathbf{mutate}(\mathbf{P})$
    $p' \leftarrow \arg\min_{convergence}(P)$
    $\mathbf{p} \leftarrow \pi(P)$
    for $p_i$ in $\mathbf{p}$ do
        $p_i.genotype \leftarrow \mathbf{mg}(p'.genotype)$
    end for
end procedure
```

In order to mutate a population $\mathbf{P}$ using π as a replacement policy, we use the process outlined in algorithm 3.

A.3 Agent Architecture

A.3.1 Sender Architecture

The sender architecture comprises of a linear layer input mapping the input feature size (512) to the hidden size. The image feature vector is therefore mapped to the same dimension as the RNN layer, where it is used as the initial hidden state. When training, for each step of the sender RNN we apply the cell and use the straight-through Gumbel-

Softmax trick to be able back-propagate gradients through the discrete message output. During evaluation however, we sample the categorical distribution at each step to produce each token in the sentence.

A.3.2 Receiver Architecture

The receiver architecture is simpler and takes as an input the message outputted by the sender and outputs a vector of input feature size (512). A single embedding matrix is used to encode the sender's message. During training the message is linearly transformed using the embedding matrix, while during the evaluation pass the discrete message outputs of the sender are used to map to the specific embedding dimensions. The embedded message is then passed to the RNN layer, and the final state of the RNN is linearly mapped back to the feature size. Doing so allows us to obtain a prediction for each image feature (distractors and true image), by comparing the alignment between the receiver output and the respective feature vectors.

A.4 Feature Extraction

In order to obtain image features, we pre-trained a convolutional model on the task using the raw image as input. Due to the input size requirements of the convolutional model, we resize the images linearly to be 128 by 128 (height, width) by 3 (RGB channels). We used early stopping conditions on the validation accuracy, an embedding size of 256, and hidden size of 512. The two agents are otherwise trained with the same parameters as other experiments: vocab size and max sentence length of 5, Adam optimizer with learning rate of 0.001.

For the visual module itself, we used a similar architecture to that in Choi et al. (2018) albeit smaller. We used a five-layer convolution network with 20 filters, and a kernel size and stride of 3 for all layers. For every convolutional layer, ReLU activation was applied on the output, after a Batch normalization step with no bias parameter. The linear layer which followed the convolutional layers had output dimensions of 512 and a ReLU activation function. This allows us to obtain image features of size 512, which we then used for all experiments.

A.5 Additional Figures and Analysis

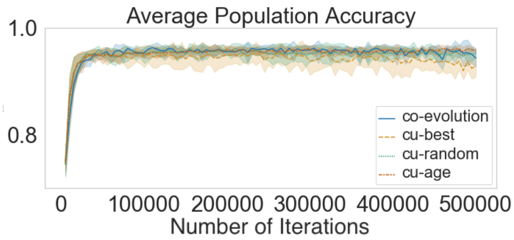


Figure 10: Average Population Accuracy for all Iterations

Modeling Context in Answer Sentence Selection Systems on a Latency Budget

Rujun Han[*]
University of Southern California
Los Angeles, CA, USA
rujunhan@usc.edu

Luca Soldaini
Amazon Alexa
Manhattan Beach, CA, USA
lssoldai@amazon.com

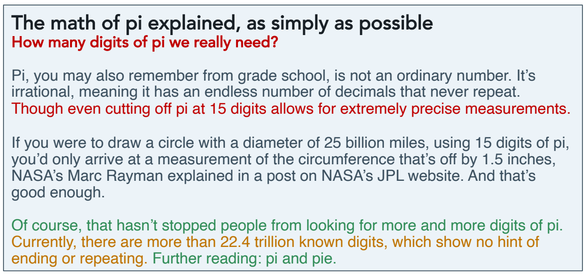
Alessandro Moschitti
Amazon Alexa
Manhattan Beach, CA, USA
amosch@amazon.com

Abstract

Answer Sentence Selection (AS2) is an efficient approach for the design of open-domain Question Answering (QA) systems. In order to achieve low latency, traditional AS2 models score question-answer pairs individually, ignoring any information from the document each potential answer was extracted from. In contrast, more computationally expensive models designed for machine reading comprehension tasks typically receive one or more passages as input, which often results in better accuracy. In this work, we present an approach to efficiently incorporate contextual information in AS2 models. For each answer candidate, we first use unsupervised similarity techniques to extract relevant sentences from its source document, which we then feed into an efficient transformer architecture fine-tuned for AS2. Our best approach, which leverages a multi-way attention architecture to efficiently encode context, improves 6% to 11% over non-contextual state of the art in AS2 with minimal impact on system latency. All experiments in this work were conducted in English.

1 Introduction

AS2 models for open-domain QA typically consider sentences from webpages as independent candidate answers for a given question. For any webpage containing potential answer candidates for a question, AS2 models first extract individual sentences, then independently estimate their likelihood of being correct answers; this approach enable highly efficient processing of entire documents. However, under this framework, context information from the entire webpage (global context), which could be crucial for selecting correct answers, is ignored. Conversely, current systems in Machine Reading (MR) (Huang et al., 2019; Kwiatkowski et al., 2019a; Lee et al., 2019; Joshi et al., 2020a) uses a much larger context from the retrieved documents. MR models receive a question and one or more passages retrieved through a search engine as input; they then select one or more spans from the input passages to return as answer.



The math of pi explained, as simply as possible
How many digits of pi we really need?

Pi, you may also remember from grade school, is not an ordinary number. It's irrational, meaning it has an endless number of decimals that never repeat. Though even cutting off pi at 15 digits allows for extremely precise measurements.

If you were to draw a circle with a diameter of 25 billion miles, using 15 digits of pi, you'd only arrive at a measurement of the circumference that's off by 1.5 inches, NASA's Marc Rayman explained in a post on NASA's JPL website. And that's good enough.

Of course, that hasn't stopped people from looking for more and more digits of pi. Currently, there are more than 22.4 trillion known digits, which show no hint of ending or repeating. Further reading: pi and pie.

Figure 1: Candidate sentence (in orange) for question "*how many digits are in pi?*". Local context is shown in green, while global context is shown in red.

While being potentially more accurate, MR models typically have higher computational requirements (and thus higher latency) than AS2 models. That is because MR models need to process passages in their entirety before an answer can be extracted; conversely, AS2 systems break down paragraphs in candidate sentences, and evaluate them all at once in parallel. Therefore, in many practical applications, MR models are only used to examine 10 to 50 candidate passages; in contrast, AS2 approaches can potentially process hundreds of documents, e.g., (Matsubara et al., 2020; Soldaini and Moschitti, 2020).

In this work, we study techniques that can combine the efficacy of MR models with the efficiency of AS2 approaches, while keeping a single sentence as target answer, as in related AS2 works[1]. In particular, we focus our efforts on improving accuracy of AS2 systems without affecting their latency.

---

[*] Work was conducted while the author was an intern at Amazon Alexa.

[1] MR systems have a different aim than AS2.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3005–3010
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

Early neural models for retrieval-based QA focused on incorporated neighboring sentences (local context) to improve performance. For example, Tan et al. (2017) proposed a neural architecture based on gated recurrent units to encode question, answer, and local context; their approach, while effective at the time, shows a significant gap to the current state-of-the-art models (Garg et al., 2020). Min et al. (2018) studied neural efficient models for MR by optimizing answer candidate extraction. More recently, researchers have focused in including source document information in transformer models. For example, Joshi et al. (2020b) proposed a contextualized model for MR that augments named entities in candidate passages with snippets extracted from Wikipedia pages. Their approach, while interesting, is limited to entities-based context, and specific to Wikipedia and MR domain. For AS2, Lauriola and Moschitti (2021) proposed a model that uses local context as defined by the preceding and following sentences of the target answer. They also introduced a simple bag-of-words representation of documents as global context, which did not show significant improvement over non-contextual AS2 models.

Unlike previous efforts, our approaches consider both *local context* (that is, the sentences immediately preceding or succeeding a candidate answer), as well as *global context* (phrases in documents that represent the overall topics of a page), as they can both uniquely contribute to the process of selecting the right answer. As shown in the example in Figure 1, local context can help disambiguate cases where crucial entities are not present in the candidate answer (there's no mention of *"pi"* in *"[c]urrently, there are more than 22.4 trillion known digits"*); conversely, global context can help reaffirm the relevance of a candidate answer in cases where noisy information is extracted as local context (in the example, *"[f]urther reading: pi and pie"* does not contain any relevant information).

The contributions of this work are: (**i**) first, we introduce two effective techniques to extract relevant local and global contexts for a given question and candidate answer; (**ii**) then, we propose three different methods for combining contextual information for AS2 tasks; (**iii**) finally, we evaluate our approaches on two AS2 datasets: ASNQ (Garg et al., 2020) and a benchmark dataset we built to evaluate real-world QA systems. Results show that our most efficient system, which leverages a multi-way attention architecture, can improve over the previous non-contextual state of the art model for AS2 by up to 11%; furthermore, these results are achieved while maintaining similar efficiency to the best-performing, non-contextual AS2 systems, making our approach a viable strategy for latency-sensitive applications. Code and models are made available at `https://github.com/alexa/wqa-contextual-qa`.

Question: "where did the potter's wheel first develop"
Corrent Answer: "Tournettes, in use around 4500 BC in the Near East, were turned slowly by hand or by foot while coiling a pot"
Sentence selected by N-grams: "In the Iron Age, the potter 's wheel in common use had a turning platform about one metre (3 feet) above the floor, connected by a long axle to a heavy flywheel at ground level. Use of the potter's wheel became widespread throughout the Old World but was unknown in the Pre-Columbian New World, where pottery was handmade by methods that included coiling and beating."

Question: "where do pineapples come from in the world"
Correct answer: "In 2016, Costa Rica, Brazil, and the Philippines accounted for nearly one-third of the world's production of pineapple."
Sentence selected by Cosine Similarity: "The plant is indigenous to South America and is said to originate from the area between southern Brazil and Paraguay"

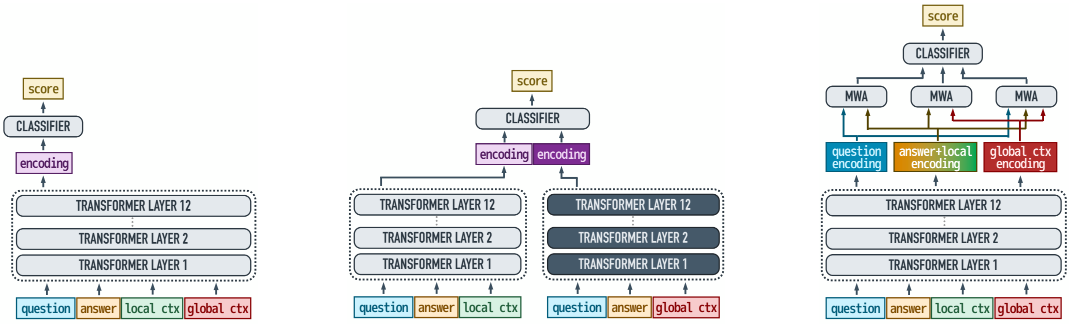
Table 1: Examples of **global context** selected via N-gram similarity (top) and cosine similarity (bottom). Overall, the N-gram approach tends to select longer context sentences than Cosine's, which in turn leads to fewer context sentences being included in the global context (as we limit it to 128 tokens). Empirically, we also noticed that N-gram selected context sentences also contain more noise.

2 Methodology

Our approach to ranking candidate answer consists of two components: the first (Section 2.1) is responsible for extracting context for each candidate answer, while the second (Section 2.2) encodes information from local and global contexts to score each question / candidate answer pair.

2.1 Context Construction

As previously mentioned, our proposed method for contextualizing answers relies on enriching them with information encoded in sentences adjacent to them, as well as from sentences throughout the document each potential answer comes from; we will define these extraction processes in this section.



(a) (b) (c)

Figure 2: From left to right, the three approaches we evaluated in this work: context concatenation (Figure 2a), context ensemble (Figure 2b), and multi-way attention (Figure 2c).

In the rest of this work, we will use Q to refer to a question and $\mathcal{D} = \{D_1, \dots, D_i, \dots, D_N\}$ to indicate a collection of documents containing potential answers for Q. Each document D_i is comprised of an ordered sequence of sentences $D_i = \langle C_{i,1}, \dots, C_{i,j}, \dots, C_{i,M} \rangle$; each sentence $C_{i,j}$ could be used either as candidate answer, or as context for another candidate.

2.1.1 Local Context

Similarly to previous work (Tan et al., 2017; Lauriola and Moschitti, 2021), we define local context $\text{Loc}_k(C_{i,j})$ for candidate $C_{i,j}$ as the sentences immediately preceding and succeeding each answer candidate within a window of $2k+1$ sentences, i.e., $\text{Loc}_k(C_{i,j}) = \langle C_{i,j-k}, \dots C_{i,j-1}, C_{i,j+1} \dots C_{i,j+k} \rangle$. In our experiments, we tried constructing a local context of up to six sentences; however, we observed diminishing return when using more than the previous and next sentences (i.e., $k = 1$) at the expense of more computational complexity. Therefore, the results presented in this work use two adjacent sentences as local context.

2.1.2 Global Context

Unlike local context, there are many potential approaches to extracting information that can be used to understand relevancy of a candidate answer to a question. We proposed and evaluated two different techniques for extracting global context $\text{Glo}_h(C_{i,j})$ (examples for both are shown in Table 1).

N-gram Overlap Similarly, to Joshi et al. (2020b), we experimented with selecting sentences as global context based on their n-gram overlap with question and candidates.

In detail, we first extract the set of all unigrams, bigrams, and trigrams from question Q and candidate $C_{i,j}$, which we denote as $\text{Ng}_{1,2,3}(Q, C_{i,j})$; then, we repeat the same procedure for all $\{C_{i,p} \subset \mathcal{D} \text{ where } p \neq j\}$ to obtain $\text{Ng}_{1,2,3}(C_{i,p})$. Finally, we score each sentence as follows:

$$\text{Score}_{ngrams}(C_{i,p}|Q, C_{i,j}) = \frac{|\text{Ng}_{1,2,3}(C_{i,p}) \cap \text{Ng}_{1,2,3}(Q, C_{i,j})|}{|\text{Ng}_{1,2,3}(Q, C_{i,j})|} \quad (1)$$

and pick the top h sentences as global context.

Semantic Similarity N-grams overlap can only extract spans of text that are lexically similar to either the query or candidate. To better capture context that is topically relevant to an answer, we also propose to use cosine distance between sentence embeddings to approximate semantic similarity.

Given a sentence encoder model[2] $\mathcal{M}$, we first obtain a representations for the question-answer pair $\mathcal{M}(Q \oplus C_{i,j})$ and context sentences $\{\mathcal{M}(C_{i,p})$ for all $p = \{1, \dots, M\}, p \neq j$; then we pick the top h sentences maximizing the following cosine similarity score as global context:

$$\text{Score}_{sim}(C_{i,p}|Q, C_{i,j}) = \frac{\mathcal{M}(Q \oplus C_{i,j}) \cdot \mathcal{M}(C_{i,p})}{||\mathcal{M}(Q \oplus C_{i,j})|| \; ||\mathcal{M}(C_{i,p})||}, \quad (2)$$

where $\oplus$ indicates string concatenation.

2.2 Contextual AS2 Models

Once local context $\text{Loc}_k(C_{i,j})$ and global context $\text{Glo}_h(C_{i,j})$ are extracted for candidate $C_{i,j}$, we encode them in conjunction with candidate answer and question to estimate the likelihood of $C_{i,j}$ being a correct answer for Q. Our approaches (summarized in Figure 2) consume up to $h = 5$ sen-

---

[2] In our experiments, we use non-finetuned RoBERTa$_{\text{BASE}}$ model (Liu et al., 2019).

Model	**P@1**	**MAP**	**MRR**	**Latency**
No Context (Garg et al., 2020)	0.596	0.685	0.706	**5.49**
Local Context (Lauriola and Moschitti, 2021)	0.653	0.732	0.752	5.62
Global Context	0.622	0.698	0.722	5.67
Context Concat	0.631	0.726	0.743	5.76
Context Ensemble	**0.668**	**0.743**	**0.765**	7.41
MWA	0.661	0.742	0.758	5.82

Table 2: Results on the ASNQ Test Set. The latency is reported in micro-seconds (μs) per sample (± 0.1 μs with 95% CI).

tences as global context so not to exceed 128 tokens. [3] Similarly, to other efforts in this area (e.g., (Garg et al., 2020)), we leverage state-of-the-art transformer models to estimate said probability. Specifically, we studied three approaches to encode question and answer context. Although the methods we proposed can be easily combined with any transformer architecture, all models described here are initialized from a RoBERTa$_{\text{BASE}}$ checkpoint.

Context Concatenation A simple baseline (van Aken et al., 2019; Joshi et al., 2020b) for encoding multiple contexts is to concatenate each question, candidate answer, and local/global context text and feed them through transformer model (Fig. 2a); the resulting encoding is then projected to a probability distribution using a dense feed-forward layer. This baseline relies on the transformer self-attention mechanism to implicitly model relations between local and global context.

Context Ensemble As mentioned in Section 1, local and global contexts might capture different aspects of the source document of a candidate answer. To empirically verify this hypothesis, we evaluated an ensemble model that encodes local and global contexts separately using two independent transformer models (Figure 2b). The two models are independently trained for AS2; then, their encodings are concatenated and passed to a feed-forward layer to estimate relevance of candidate $C_{i,j}$ for question Q. The top 3 layers[4] model is once again fine tuned on the training set.

Model	**P@1**	**MAP**	**MRR**
No Context		*baseline*	
Global Context	+3.95%	+2.52%	+2.29%
Local Context	+3.59%	+2.89%	+2.21%
Context Concat	+1.52%	+2.33%	+1.34%
Context Ensemble	**+5.92%**	**+4.10%**	**+3.39%**
MWA	+5.56%	+3.92%	+3.08%

Table 3: Results on the WQA Test Set.

Multi-way Attention (MWA) While leveraging independent encoders for local and global contexts can lead to an improvement in performance compared to using a single encoder, it also doubles computational requirements. Therefore, we also explored techniques that incorporate inductive biases into transformer models and achieve efficiency comparable to the context concatenation approach. One such approach is, as shown in Figure 2c, to combine a transformer model with a multi-way attention mechanism (Tan et al., 2018), which has been shown to be effective for commonsense reasoning tasks (Huang et al., 2019). This approach still uses a single transformer model to produce an encoding for a sequence of question, candidate answer, local context, and global context; however, similarly to the ensemble model, the additional attention mechanism forces the encoder to selectively look at local and global contexts separately.

3 Experiments

3.1 Setup

In order to validate the effectiveness of the proposed context modeling techniques, we evaluated our results on two datasets: ASNQ and WQA.

ASNQ The Answer Sentence Natural Questions dataset (Garg et al., 2020) is a large collection of 59,914 questions and 24,732,396 candidate answers. It was obtained by extracting sentence candidates from the Google Natural Question (NQ) benchmark (Kwiatkowski et al., 2019b). We use the train, development, and test splits proposed by Soldaini and Moschitti (2020).

WQA The Web-based Question Answering is an in-house dataset built by Alexa AI as part of the effort of understanding and benchmarking QA systems. The creation process includes the following steps: (i) Given a set of questions, a search engine is used to retrieve up to 100 web pages from an index containing hundreds of million pages. (ii) From the set of retrieved documents, all candidate

---

[3] Using up 5 sentences resulted in 3.02 (n-gram context) and 2.85 (cosine context) sentences being selected on average for the ASNQ dataset.

[4] We tested with approaches for gradual unfreezing of the top k layers; $k = 3$ yielded the best validation scores.

Technique	P@1	MAP	MRR
N-gram Overlap	0.638	0.725	0.746
Cosine Similarity	0.661	0.742	0.758

Table 4: Comparison of global context extraction techniques on the ASNQ test set when used with MWA.

sentences are extracted and ranked using AS2 models from Garg et al. (2020); and (iii) at least 25 candidates for each question are annotated by humans. Overall, the version of WQA we used contains 6,962 questions and 283,855 candidate answers. We reserved 3,000 questions for evaluation, 808 for development, and used the rest for training [5].

Models were trained on a single machine with 8 NVIDIA Tesla V100 GPUs with 32 GB of memory each. We used model implementations from the Transformers library when available (Wolf et al., 2020). All our experiments were computed using mixed precision through NVIDIA apex[6]. Latency was measured on single GPU with a fixed batch size of 128. Tokenization and time to transfer tensors to the GPU was not included in the latency values.

3.2 Results and Discussion

Results on the ASNQ and WQA are summarized in tables 2 and 3, respectively. Overall, we observe that the context ensemble model achieves the best performance; however, as observed in Section 2.2, this model is twice as large as a RoBERTa$_{\text{BASE}}$ model, thus it is a rather expensive solution.

Among our baselines, we note that local context outperforms the model leveraging the global context. This observation suggests that local information carries more importance in understanding the semantic relationship between question and candidate answers. Surprisingly, we observe that simply concatenating local and global contexts achieves worse performance of local context alone, and it even underperforms the global context method on WQA. This suggests that, without any additional structure, the self-attention mechanism of the transformer cannot effectively distinguish and leverage information from the local and global contexts.

We note that MWA achieves near identical performance to the ensemble model on both datasets, suggesting that a controlled attention mechanism can overcome limitations in the representation for vanilla transformers, while reducing latency by 21.5% and memory usage by 89%. MWA also matches the latency of the context concatenation model, while improving it by 4.8% and 3.9% in P@1 on ASNQ and WQA, respectively.

Finally, we study the effect of our proposed global extraction techniques in Table 4. We observe that, among the two proposed algorithms, the cosine similarity approach significantly outperforms the N-gram based method. This confirms that pre-trained language models can better select context semantically related to question and candidates.

We note n-gram overlap is less computationally taxing, as it can be efficiently implemented as a set of sparse operations over bag of word representations of the question and answer candidates. On the other hand, for cosine similarity, it is necessary to compute Score$_{sim}(C_{i,p}|Q, C_{i,j})$ for all context sentences using a transformer model. Recently introduced transformer architecture variants could be used to either speed up this similarity computation (Cao et al., 2020), or compute query and text representation independently (Khattab and Zaharia, 2020). We leave the evaluation of these techniques to future work.

4 Conclusions

For efficiency reasons, traditional AS2 models are designed to estimate answer relevancy by only comparing question and candidates. In this work, we described and evaluate several techniques to incorporate local and global contexts in the answer selection process. The results of our experiments show that our proposed methods significantly outperform non-contextual approaches; further, we empirically demonstrate that local and global contexts can be effectively combined to further improve ranking performance.

References

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does bert answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1823–1832, New York, NY, USA. Association for Computing Machinery.

Qingqing Cao, Harsh Trivedi, Aruna Balasubramanian, and Niranjan Balasubramanian. 2020. De-

---

[5] The public version of WQA will be released in the short-term future. Please search for a publication with title *WQA: A Dataset for Web-based Question Answering Tasks* on arXiv.org.

[6] `https://github.com/NVIDIA/apex`

Former: Decomposing pre-trained transformers for faster question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4497, Online. Association for Computational Linguistics.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *AAAI*, pages 7780–7788.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

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

Mandar Joshi, Kenton Lee, Yi Luan, and Kristina Toutanova. 2020b. Contextualized representations using textual encyclopedic knowledge. *arXiv preprint arXiv:2004.12006*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48.

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

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

Ivano Lauriola and Alessandro Moschitti. 2021. Answer sentence selection using local and global context in transformer models. *ECIR*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yoshitomo Matsubara, Thuy Vu, and Alessandro Moschitti. 2020. Reranking for Efficient Transformer-Based Answer Selection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1577–1580, New York, NY, USA. Association for Computing Machinery.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1725–1735, Melbourne, Australia. Association for Computational Linguistics.

Luca Soldaini and Alessandro Moschitti. 2020. The cascade transformer: an application for efficient answer sentence selection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5697–5708, Online. Association for Computational Linguistics.

Chuanqi Tan, Furu Wei, Wenhui Wang, Weifeng Lv, and Ming Zhou. 2018. Multiway attention networks for modeling sentence pairs. In *IJCAI*, pages 4411–4417.

Chuanqi Tan, Furu Wei, Qingyu Zhou, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2017. Context-aware answer sentence selection with hierarchical gated recurrent neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(3):540–549.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Syntax-BERT: Improving Pre-trained Transformers with Syntax Trees

Jiangang Bai[1,*], **Yujing Wang**[1,2,†], **Yiren Chen**[1], **Yaming Yang**[2]
Jing Bai[2], **Jing Yu**[3], **Yunhai Tong**[1]
[1]Peking University, Beijing, China
[2]Microsoft Research Asia, Beijing, China
[3]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{pku_bjg,yujwang,yrchen92,yhtong}@pku.edu.cn
{yujwang,yayaming,jbai}@microsoft.com
yujing02@iie.ac.cn

Abstract

Pre-trained language models like BERT achieve superior performances in various NLP tasks without explicit consideration of syntactic information. Meanwhile, syntactic information has been proved to be crucial for the success of NLP applications. However, how to incorporate the syntax trees effectively and efficiently into pre-trained Transformers is still unsettled. In this paper, we address this problem by proposing a novel framework named Syntax-BERT. This framework works in a plug-and-play mode and is applicable to an arbitrary pre-trained checkpoint based on Transformer architecture. Experiments on various datasets of natural language understanding verify the effectiveness of syntax trees and achieve consistent improvement over multiple pre-trained models, including BERT, RoBERTa, and T5.

1 Introduction

Pre-trained language models like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2020) and T5 (Raffel et al., 2019) become popular in recent years and achieve outstanding performances in various NLP benchmarks. These models often choose a Transformer architecture largely owing to its attractive scalability. Studies (Hewitt and Manning, 2019; Jawahar et al., 2019) have shown that a pre-trained transformer is able to capture certain syntactic information implicitly by learning from sufficient examples. However, there is still a big gap between the syntactic structures implicitly learned and the golden syntax trees created by human experts.

On the other hand, syntax tree is a useful prior for NLP-oriented neural networks (Kiperwasser and Ballesteros, 2018). For example, Tree-LSTM (Tai et al., 2015) extends the sequential architecture of LSTM to a tree-structured network. Linguistically-informed self-attention (LISA) (Strubell et al., 2018) proposes a multi-task learning framework for semantic role labeling, which incorporates syntactic knowledge into Transformer by training one attention head to be attended to its parent in a syntax tree. In addition, Nguyen et al. (2020) integrate tree-structured attention in Transformer with hierarchical accumulation guided by the syntax tree.

Although there are numerous works on syntax-enhanced LSTM and Transformer models, none of the previous works have addressed the usefulness of syntax-trees in the pre-training context. It is straight-forward to ask: *it is still helpful to leverage syntax trees explicitly in the pre-training context?* If the answer is yes, *can we ingest syntax trees into a pre-trained checkpoint efficiently without training from scratch for a specific downstream application?* This is an appealing feature in practice because pre-training from scratch is a huge waste of energy and time.

In this paper, we propose Syntax-BERT to tackle the raised questions. Unlike a standard BERT, which has a complete self-attention typology, we decompose the self-attention network into multiple sub-networks according to the tree structure. Each sub-network encapsulates one relationship from the syntax trees, including ancestor, offspring, and sibling relationships with different hops. All sub-networks share the same parameters with the pre-trained network, so they can be learned collaboratively and inherited directly from an existing checkpoint. To select the task-oriented relationships automatically, we further adopt a topical attention layer to calculate the relative importance of syntactic representations generated by different sub-networks. Finally, the customized representation is calculated by weighted summation of all

---

[*] The work was done when the author visited Microsoft Research Asia.

[†] Corresponding Author

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3011–3020
April 19 - 23, 2021.

sub-networks.

We conduct extensive experiments to verify the effectiveness of Syntax-BERT framework on various NLP tasks, including sentiment classification, natural language inference, and other tasks in the GLUE benchmark. Experimental results show that Syntax-BERT outperforms vanilla BERT models and LISA-enhanced models consistently with multiple model backbones, including BERT, RoBERTa, and T5. Specifically, it boosts the overall score of GLUE benchmark from 86.3 to 86.8 for T5-Large (Raffel et al., 2019) checkpoint, which is already trained on a huge amount of data. This improvement is convincing since only a few extra parameters are introduced to the model.

Our **major contributions** are as follows:

- To the best of our knowledge, Syntax-BERT is one of the first attempts to demonstrate the usefulness of syntax trees in pre-trained language models. It works efficiently in a plug-and-play fashion for an existing checkpoint without the need for pre-training from scratch.
- To integrate syntax trees into pre-trained Transformers, we propose a novel method that decomposes self-attention networks into different aspects and adopts topical attention for customized aggregation. As shown in the ablation study, this design benefits from syntactic structures effectively while retaining pre-trained knowledge to the largest extent.
- Syntax-BERT shows consistent improvement over multiple pre-trained backbone models with comparable model capacities. It can be combined with LISA to achieve further enhancement, indicating that these two algorithms are complementary to each other.

2 Related Work

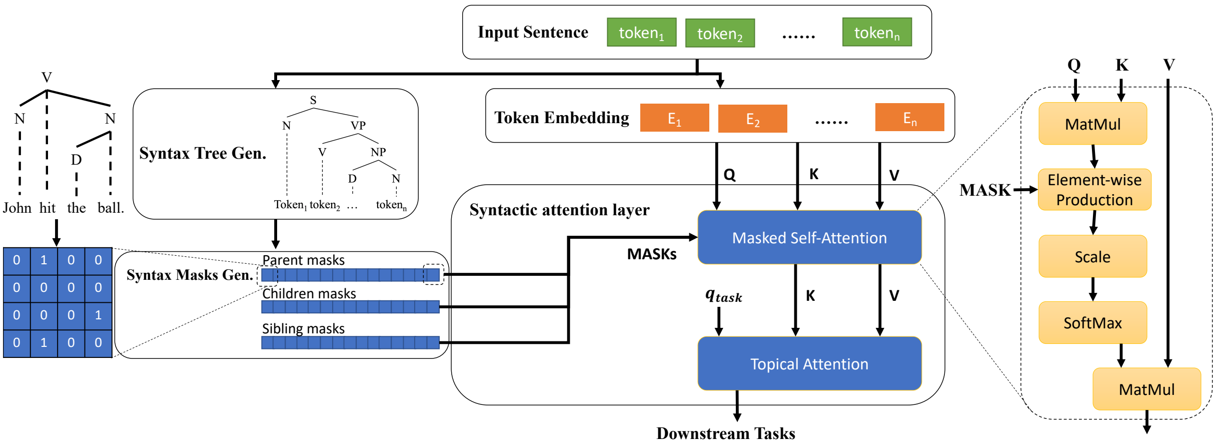
2.1 Pre-trained language models

Recently, pre-trained language models have received significant attention from the natural language processing community. Many excellent pre-trained language models are proposed, such as BERT, RoBERTa and T5. Transformer (Vaswani et al., 2017) is a typical architecture for pre-training language models, which is based on the self-attention mechanism and is much more efficient than RNNs. BERT (Devlin et al., 2019) is a representative work that trains a large language model on the free text and then fine-tunes it on specific downstream tasks separately. BERT is pre-trained on two auxiliary pre-training tasks, Masked Language Model (MLM) and Next Sentence Prediction (NSP). RoBERTa (Liu et al., 2020) is an improved variant of BERT which utilizes dynamic masks. In RoBERTa, the NSP task is cancelled, but the full-sentence mechanism is considered. At the same time, the size of RoBERTa's training data ($\sim$160GB) is ten times the size of BERT's training data. Moreover, Raffel et al. (2019) explore the effectiveness of multiple transfer learning techniques and apply these insights at scale to create a new model T5 (Text to Text Transfer Transformer). With T5, they reform all NLP tasks into a unified text-to-text format where the input and output are always text strings. This is in contrast to BERT-style models that only output either a class label or an input span.

2.2 Syntax-aware models

Syntax is a crucial prior for NLP-oriented neural network models. Along this direction, a range of interesting approaches have been proposed, like Tree-LSTM (Tai et al., 2015), PECNN (Yang et al., 2016), SDP-LSTM (Xu et al., 2015), Supervised Treebank Conversion (Jiang et al., 2018), PRPN (Shen et al., 2018), and ON-LSTM (Shen et al., 2019).

Recent works also investigate syntactic knowledge in the context of Transformer, which are more related to this paper. For instance, Syntax-Infused Transformer (Sundararaman et al., 2019) feeds the extra syntactic features into the Transformer models explicitly, but it only considers simple syntactic features and does not provide a generic solution to incorporate tree-structured knowledge. Strubell et al. (2018) present a neural network model named LISA (Linguistically-Informed Self-Attention) that learns multi-head self-attention in a multi-task learning framework consisting of dependency parsing, part-of-speech tagging, predicate detection, and semantic role labeling. They also show that golden syntax trees can dramatically improve the performance of semantic role labeling. Moreover, Nguyen et al. (2020) propose a hierarchical accumulation approach to encode parse tree structures into self-attention mechanism. However, these approaches are designed for training a Transformer from scratch without benefiting from pre-trained checkpoints. Instead, our frame-



0	1	0	0
0	0	0	0
0	0	0	1
0	1	0	0

Figure 1: The Overall Architecture of Syntax-BERT. Note that the leftmost part shows an example of syntax tree and its corresponding parent syntax mask ($d = 1$).

work works in a plug-and-play mode and retains the pre-trained knowledge as much as possible for downstream applications. Concurrent to our work, Sachan et al. (2020) investigate popular strategies for incorporating dependency structures into pre-trained language models, revealing essential design decisions are necessary for strong performances. In addition, Hewitt and Manning (2019) design two sets of probes to determine whether the embedded space can be converted into syntactic information space through a linear transformation. It gives the evaluation metrics to examine how much syntactic information is included in a model.

3 Syntax-BERT

Syntax-BERT is a variant of pre-trained Transformer models, which changes the flow of information in a standard BERT network via a syntax-aware self-attention mechanism. First, the overall architecture of Syntax-BERT is presented in Section 3.1. Then, we introduce the construction of syntax trees and corresponding masks in Section 3.2. The details of syntactic attention layers will be described in Section 3.3.

3.1 Architecture

As mentioned earlier, one limitation of vanilla Transformer is that it simply uses a fully-connected topology of tokens in the pre-trained self-attention layer. Although the self-attention mechanism automatically calculates a relevance score for each token pair, it still suffers from optimization and over-fitting problems, especially when the training data is limited. Some previous works have tried to induce syntactic structure explicitly into self-attention. For instance, in Linguistically-Informed Self-Attention (LISA) (Strubell et al., 2018), syntax tree is incorporated by training one attention head to be attended to the parent of each token. However, other structural features such as siblings and children are discarded in the model. Moreover, it can not distinguish the usefulness of multiple syntactic features while largely retain the knowledge from a pre-trained checkpoint.

Syntax-BERT is designed to incorporate grammatical and syntactic knowledge as prior in the self-attention layer and support fine-grained adaptation for different downstream tasks. Specifically, it generates a bunch of sub-networks based on sparse masks reflecting different relationships and distances of tokens in a syntax tree. Intuitively, the tokens inside a sub-network often semantically related to each other, resulting in a topical representation. Therefore, we can adopt a topical attention layer to aggregate task-oriented representations from different sub-networks.

The overall architecture of Syntax-BERT is illustrated in Figure 1. As shown in the left part of this figure, we generate syntax masks for the input sentence in two steps. First, the input sentence is converted into the corresponding tree structure by a syntax parser. Second, we extract a bunch of syntax-related masks according to different features incorporated in the syntax tree. Next, the sentence is embedded similar to a standard BERT (token + positional + field embedding) and served as input to the self-attention layer. Each self-attention layer in the Syntax-BERT is composed of two kinds of attention modules, namely *masked self-attention* and *topical attention*. In a *masked* *self-attention* module, we apply syntactic masks to the fully-connected topology, generating topical sub-networks that share parameters with each other. Furthermore, the representations from different sub-networks are aggregated through a *topical attention* module so that the task-related knowledge can be distilled to the final representation vector.

3.2 Masks induced by syntax tree

Generically, a syntax tree is an ordered, rooted tree that represents the syntactic structure of a sentence according to some context-free grammar. It can be defined abstractly as $T = \{R, \mathcal{N}, \mathcal{E}\}$, where R is the root of syntax tree, $\mathcal{N}$ and $\mathcal{E}$ stands for node set and edge set respectively. The most commonly-used syntax trees are constituency trees (Chen and Manning, 2014) and dependency trees (Zhu et al., 2013), and we use both of them in our experiments unless notified.

To utilize the knowledge in a syntax tree effectively, we introduce syntax-based sub-network typologies in the self-attention layer to guide the model. Each sub-network shares the same model parameters with the global pre-trained self-attention layer, while each sub-network reflects a specific aspect of the syntax tree. This procedure can be easily implemented by multiple masks applied to the complete graph topology.

Without loss of generality, we design three categories of masks reflecting different aspects of a tree structure, namely *parent mask*, *child mask*, and *sibling mask*. For a pairwise inference task that contains a pair of sentences as input, we also apply another mask, i.e., *pairwise mask*, to capture the inter-sentence attention. Moreover, the distances between nodes (tokens) in a tree incorporate semantic relatedness. Starting from a node A, along the edges of a syntax tree, the minimum number of edges required to reach another node B can be regarded as the distance between A and B, written as $dist(A, B)$. We create fine-grained masks according to the distance between two nodes to enable customized aggregation of task-oriented knowledge.

Mathematically, a mask can be denoted by $M \in \{0,1\}^{n\times n}$, where $M_{i,j} \in \{0,1\}$ denotes if there is a connection from token i to token j, and n is the number of tokens in the current sentence.

In the **parent mask** with certain distance d, we have $M^p_{i,j,d} = 1$ if and only if the node i is the parent or ancestor of node j, at the same time $dist(i,j) = d$. Otherwise, the value will be set as zero.

In the **child mask** with certain distance d, we have $M^c_{i,j,d} = 1$ if and only if the node i is the child or offspring of node j, at the same time $dist(i,j) = d$. In other words, node j is the parent or ancestor of node i.

In the **sibling mask** with certain distance d, we have $M^s_{i,j,d} = 1$ if and only if we can find their lowest common ancestor and $dist(i,j) = d$. Note that if two nodes are in the same sentence, we can always find the lowest common ancestor, but the value should be zero if the corresponding nodes come from different sentences (in pairwise inference tasks).

The **pairwise mask** captures the interaction of multiple sentences in a pairwise inference task. We have $M^{pair}_{i,j} = 1$ if and only if both node i and j are from different sentences. we do not consider the distances in-between as the nodes are from different trees.

3.3 Syntactic attention layers

A block of Syntax-BERT contains two kinds of attention modules: *masked self-attention* and *topical attention*. The operations in a *masked self-attention* are similar to a standard self-attention except that we have sparse network connections as defined in the masks. The *masked self-attention* can be formulated as an element-wise multiplication of dot-product attention and its corresponding mask:

$$\begin{aligned} &MaskAtt(Q,K,V,M) = \sigma\left(\frac{QK^\top \odot M}{\sqrt{d}}\right)V \\ &A_{i,j} = MaskAtt(HW^Q_i, HW^K_i, HW^V_i, M_j) \\ &H_j = (A_{1,j} \oplus A_{2,j} \oplus ... \oplus A_{k,j})W^O, j \in 1,...,m \end{aligned} \quad (1)$$

where Q, K, V represent for the matrix of query, key and value respectively, which can be calculated by the input representation H. M represents for the matrix of syntax mask and $\odot$ denotes an operator for element-wise production; σ stands for softmax operator; $A_{i,j}$ denotes the attention-based representation obtained by the i^{th} head and j^{th} sub-network; W^Q_i, W^K_i and W^V_i represent for the parameters for linear projections; M_j denotes the mask for the j^{th}sub-network; and H_j denotes the corresponding output representation.

The output representations from different sub-networks embody knowledge from different syntactic and semantic aspects. Therefore, we leverage

Task	#Train	#Dev	#Test	#Class
SST-1	8,544	1,101	2,210	5
SST-2	6,920	873	1,822	2
SNLI	549,367	9,842	9,824	3
MNLI	392,703	9,816/9,833	9,797/9,848	3
CoLA	8,551	1,042	1,064	2
MRPC	3,669	409	1,726	2
STS-B	5,750	1,501	1,380	*
QQP	363,871	40,432	390,965	2
QNLI	104,744	5,464	5,464	2
RTE	2,491	278	3,001	2
WNLI	636	72	147	2

Table 1: Dataset Statistics: the character '/' seperate MNLI-m and MNLI-mm, '*' represents for the regression task.

another attention layer, named *topical attention* to perform a fine-grained aggregation of these representations. The most distinct part of a *topical attention* is that q_{task} is a trainable query vector for task-specific embedding. Thus, the *topical attention* layer is able to emphasize task-oriented knowledge captured by numerous sub-networks.

$$\begin{aligned} &TopicAtt(q_{task}, K, V) = \sigma(\frac{q_{task}K^\top}{\sqrt{d}})V \\ &H^O = TopicAtt(q_{task}, HW^K, HW^V) \end{aligned} \tag{2}$$

where d denotes the size of hidden dimension, $q_{task} \in R^{1\times d}$ is a task-related learnable query embedding vector; σ stands for the softmax operator; $H = (H_1, H_2, ..., H_m)^\top \in R^{m\times d}$ is the output representation collected by multiple sub-networks; W^K and W^V are parameters in the feed-forward operations; and H^O stands for the final text representation.

4 Experiments

First, we run experiments on the Stanford Sentiment Treebank (SST) dataset (Socher et al., 2013) in Section 4.1, which is designed to study the syntactic and semantic compositionality of sentiment classification. Second, in Section 4.2, we evaluate the performance of Syntax-BERT on two natural language inference datasets: SNLI and MNLI. Then, more empirical results on the GLUE benchmark and a comprehensive ablation study will be presented in Section 4.3 and 4.4 respectively. At last, we present the analysis of the structural probes in Section 4.5.

The statistics of all datasets adopted in this paper are summarized in Table 1. For each dataset, we optimize the hyper-parameters of Syntax-BERT through grid search on the validation data. Detailed settings can be found in the appendix. In our experiments, we set the maximum value of $dist(A, B)$ in a syntax tree as 15 and use both dependency and constituency trees unless specified. Thus, we have totally 90 ($15 \times 3 \times 2$) sub-networks for single-sentence tasks and 92 ($(15 \times 3 + 1) \times 2$) sub-networks for pairwise inference tasks. We adopt Transformer (Vaswani et al., 2017), BERT-Base, BERT-Large (Devlin et al., 2019), RoBERTa-Base, RoBERTa-Large (Liu et al., 2020) and T5-Large (Raffel et al., 2019) as backbone models and perform syntax-aware fine-tuning on them. We also compare with LISA (Linguistically-Informed Self-Attention) (Strubell et al., 2018), a state-of-the-art method that incorporates linguistic knowledge into self-attention operations. Specifically, LISA (Strubell et al., 2018) adopt an additional attention head to learn the syntactic dependency in the tree structure, and the parameters of this additional head are initialized randomly.

4.1 Stanford Sentiment Treebank

The SST dataset contains more than 10,000 sentences collected from movie reviews from the *rottentomatoes.com* website. The corresponding constituency trees for review sentences are contained in the dataset, where each intermediate node in a tree represents a phrase. All phrases are labeled to one of five fine-grained categories of sentiment polarity. SST-2 is a binary classification task. We follow a common setting that utilizes all phrases with lengths larger than 3 as training samples, and only full sentences will be used in the validation and testing phase. The hyper parameters for each model are selected by grid search and listed in the appendix. We compare Syntax-BERT with vanilla baselines and LISA-enhanced models. The results are listed in Table 2. As shown in the table, our model achieves 4.8 and 4.9 absolute points improvements respectively against the vanilla Transformer with comparable parameter size. By combining our framework with LISA, the results can be further boosted obviously. This indicates that our mechanism is somewhat complementary to LISA. LISA captures the syntactic information through an additional attention head, whereas our framework incorporates syntactic dependencies into original pre-trained attention heads and increases the sparsity of the network. We can see that *Syntax-Transformer + LISA* performs the best among all settings, and similar trends are demonstrated on the BERT-Base and BERT-Large checkpoints.

Model	**SST-1**	**SST-2**
Transformer	48.4	86.2
LISA-Transformer	52.2	89.1
Syntax-Transformer (Ours)	52.7	90.1
Syntax-Transformer + LISA (Ours)	**53.2**	**91.1**
BERT-Base	53.7	93.5
LISA-BERT-Base	54.2	93.7
Syntax-BERT-Base (Ours)	54.4	94.0
Syntax-BERT-Base + LISA (Ours)	**54.5**	**94.4**
BERT-Large	54.8	94.9
LISA-BERT-Large	55.0	95.9
Syntax-BERT-Large (Ours)	55.3	96.1
Syntax-BERT-Large + LISA (Ours)	**55.5**	**96.4**

Table 2: Comparison with SOTA models on SST dataset.

Model	**SNLI**	**MNLI**
Transformer	84.9	71.4
LISA-Transformer	86.1	73.7
Syntax-Transformer (Ours)	86.8	74.1
Syntax-Transformer + LISA (Ours)	**87.0**	**74.5**
BERT-Base	87.0	84.3
LISA-BERT-base	87.4	84.7
Syntax-BERT-Base (Ours)	87.7	**84.9**
Syntax-BERT-Base + LISA (Ours)	**87.8**	**84.9**
BERT-Large	88.4	86.8
LISA-BERT-Large	88.8	86.8
Syntax-BERT-Large (Ours)	88.9	86.7
Syntax-BERT-Large + LISA (Ours)	**89.0**	**87.0**

Table 3: Comparison with SOTA models on NLI datasets.

4.2 Natural Language Inference

The Natural Language Inference (NLI) task requires a model to identify the semantic relationship (entailment, contradiction, or neutral) between a premise sentence and the corresponding hypothesis sentence. In our experiments, we use two datasets for evaluation, namely SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018). We utilize the Stanford parser (Klein and Manning, 2003) to generate constituency and dependency trees for the input sentences. The MNLI dataset has two separate sets for evaluation (matched set and mismatched set), and we report the average evaluation score of these two sets.

The test accuracies on SNLI and MNLI datasets are shown in Table 3. The syntactic prior information helps the Transformer to perform much better on the NLI tasks. The accuracies on the SNLI and MNLI datasets have been improved by 1.9 and 2.7, respectively, by applying our framework to a vanilla Transformer. The LISA-enhanced transformer can also outperform vanilla transformer on NLI tasks, but the accuracy improvement is not as large as Syntax-Transformer. When the backbone model is BERT-Base or BERT-Large, consistent conclusions can be drawn from the experimental results. It is worth noting that the syntax-enhanced models for BERT-large do not show much gain based on the vanilla counterparts. This may because BERT-Large already captures sufficient knowledge for NLI tasks in the pre-training phase.

4.3 GLUE Benchmark

The GLUE benchmark (Wang et al., 2019) offers a collection of tools for evaluating the performance of models. It contains single-sentence classification tasks (CoLA and SST-2), similarity and paraphrase tasks (MRPC, QQP, and STS-B), as well as pairwise inference tasks (MNLI, RTE, and QNLI). We use the default train/dev/test split. The hyper-parameters are chosen based on the validation set (refer to the appendix for details). After the model is trained, we make predictions on the test data and send the results to GLUE online evaluation service[1] to obtain final evaluation scores.

The evaluation scores on all datasets in GLUE benchmark are illustrated in Table 4. The performances of BERT-Base, BERT-Large, RoBERTa-Base, RoBERTa-Large, and T5-Large are reproduced using the official checkpoint provided by respective authors. We only use self-contained constituency trees for the SST-2 dataset while other datasets are processed by Stanford parser[2] to extract both dependency trees and constituency trees. For a fair comparison, all results of baseline models are reproduced by our own, which are close to the reported results.

As shown in the table, syntax-enhanced models always outperform corresponding baseline models. Most notably, Syntax-RoBERTa-Base achieves an average GLUE score of 82.1, lifting 1.3 scores from a standard RoBERTa-Base with the same setting. This is impressive as only a few extra parameters are introduced to the baseline model. Particularly, the improvements on CoLA and SST-2 datasets are fairly large, showing the generalization capability of Syntax-BERT and Syntax-RoBERTa on smaller datasets. Even on T5-Large, which is trained on more data and holds more advanced performances, our approach still outperforms the base model marginally (statistically significant under 4.3 p-value using paired t-test). We can see that more training data will improve the generalization

---

[1] https://gluebenchmark.com

[2] https://nlp.stanford.edu/software/lex-parser.shtml

Model	Avg	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m/-mm	QNLI	RTE	WNLI
Transformer	66.1	31.3	83.9	81.7/68.6	73.6/70.2	65.6/84.4	72.3/71.4	80.3	58.0	65.1
Syntax-Transformer (Ours)	**68.8**	**36.6**	**86.4**	**81.8/69.0**	**74.0/72.3**	**65.5/84.9**	**72.5/71.2**	**81.0**	**56.7**	65.1
BERT-Base	77.4	51.7	93.5	87.2/82.1	86.7/85.4	71.1/89.0	84.3/83.7	90.4	67.2	65.1
Syntax-BERT-Base (Ours)	**78.5**	**54.1**	**94.0**	**89.2/86.0**	**88.1/86.7**	**72.0/89.6**	**84.9/84.6**	**91.1**	**68.9**	65.1
BERT-Large	80.5	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	**86.8/85.9**	92.7	70.1	65.1
Syntax-BERT-Large (Ours)	**81.8**	**61.9**	**96.1**	**92.0/88.9**	**89.6/88.5**	**72.4/89.5**	86.7/86.6	**92.8**	**74.7**	65.1
RoBERTa-Base	80.8	57.1	95.4	90.8/89.3	88.0/87.4	72.5/89.6	86.3/86.2	92.2	73.8	65.1
Syntax-RoBERTa-Base (Ours)	**82.1**	**63.3**	**96.1**	**91.4/88.5**	**89.9/88.3**	**73.5/88.5**	**87.8/85.7**	**94.3**	**81.2**	65.1
RoBERTa-Large	83.9	63.8	96.3	91.0/89.4	72.9/90.2	72.7/90.1	89.5/89.7	94.2	84.2	65.1
Syntax-RoBERTa-Large (Ours)	**84.7**	**64.3**	**96.9**	**92.5/90.1**	**91.6/91.4**	**73.1/89.8**	**90.2/90.0**	**94.5**	**85.0**	65.1
T5-Large	86.3	61.1	96.1	92.2/88.7	90.0/89.2	74.1/89.9	89.7/89.6	94.8	87.0	65.1
Syntax-T5-Large (Ours)	**86.8**	**62.9**	**97.2**	**92.7/90.6**	**91.3/90.7**	**74.3/90.1**	**91.2/90.5**	**95.2**	**89.6**	65.1

Table 4: Comparison with state-of-the-art models without pre-training on GLUE benchmark.

Model	SST-2	CoLA	STS-B
BERT-Large	94.9	60.5	87.6/86.5
Syntax-BERT-Large	**96.1**	**61.9**	**89.6/88.5**
w/o topical attention	95.1	61.6	88.4/87.3
w/o syntax trees	95.0	60.5	88.0/87.1
w/o dependency trees	95.6	61.4	88.7/88.1
w/o constituency trees	95.9	61.4	87.6/86.8
w/o parent masks	95.5	60.9	88.7/87.2
w/o child masks	95.3	61.2	88.3/86.8
w/o sibling masks	95.8	61.5	89.0/88.1
w/o pairwise masks	-	-	88.8/87.9

Table 5: Ablation study

Model	UUAS	Spr.
BERT-Base (Devlin et al., 2019)	79.8	0.85
Syntax-BERT-Base	**81.1**	**0.88**
BERT-Large (Devlin et al., 2019)	82.5	0.86
Syntax-BERT-Large	**83.4**	**0.90**
RoBERTa-Large (Liu et al., 2020)	83.2	0.88
Syntax-RoBERTa-Large	**84.6**	**0.93**

Table 6: The results of using Structural Probe to test whether different models contain syntactic information or not. UUAS denotes undirected attachment score, and Spr. denotes Spearman correlation.

capability of the model and compensate for the lack of syntax priors. On the other hand, syntactic information is useful in most cases, especially when training data or computation power is limited.

4.4 Ablation Study

For a comprehensive understanding of the model design, we conduct ablation study with the following settings. (1) *without topical attention*: the topical attention layer is removed, and a simple summation layer is replaced instead; (2) *without syntax tree:* all the syntactic masks generated by the syntax trees are replaced by randomly generated masks, while the parameter size of the model remains unchanged; (3) *without constituency/dependency tree*: only one kind of syntax tree is used in the model; (4) *without parent / child / sibling / pairwise masks*: the corresponding masks are removed in the implementation.

As shown in Table 5, all datasets benefit from the usage of syntactic information. Generally, parent/child masks are of more importance than the sibling masks. Moreover, the topical attention layer is crucial to the performance of Syntax-BERT model, indicating the advantage of decomposing self-attention into different sub-networks and performing fine-grained aggregation. In addition, the pairwise mask is important on STS-B dataset and shows the benefit of cross-sentence attention.

4.5 Structural Probe

Our method ingests syntax trees into the model architecture directly. To examine if the representation learned by the model also captures syntactic knowledge effectively, we follow Hewitt and Manning (2019) to reconstruct a syntax tree of the entire sentence with linear transformation learned for the embedding space. If the syntax tree can be better reconstructed, the model is viewed to learn more syntactic information. We evaluate the tree on *undirected attachment score* – the percent of undirected edges placed correctly, and *Spearman correlation* between predicted and the actual distance between each word pair in a sentence. We probe models for their ability to capture the Stanford Dependencies formalism (de Marneffe et al., 2006). As shown in Table 6, for both metrics, the syntax-aware models get better scores than corresponding baseline models, indicating that Syntax-BERT is able to incorporate more syntax information than its vanilla counterparts.

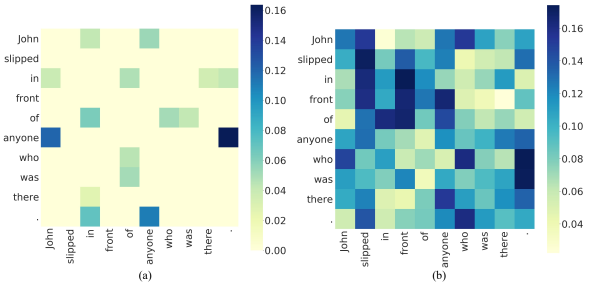


Figure 2: For an example sentence input, (a) The self-Attention scores of Syntax-Transformer corresponding to the sibling mask with $dist = 3$. (b) The self-attention scores of a vanilla Transformer.

5 Discussion

5.1 Complexity analysis

First, we choose BERT-Base as the base model to analyze the space complexity. As reported in (Devlin et al., 2019), the number of trainable parameters in BERT-Base is about 110 million. Following (Strubell et al., 2018), LISA-BERT-Base replaces one attention head in BERT-Base with a bi-affine attention head. Such an operation only adds a trainable matrix — the bi-affine transformation matrix — in each layer, which brings about 0.6 million extra parameters. Syntax-BERT-Base introduces a topical attention layer, which contains 1.0 million parameters in total for the BERT-Base version, while other parameters are inherited from vanilla BERT. Therefore, both LISA and Syntax-BERT add few parameters to the model and do not affect its original space complexity.

We now analyze the time complexity of Syntax-BERT. Assume the number of tokens in each sentence is N. First, constructing syntactic trees for each sentence and extract masking matrices can be prepossessed in the training phase or finish in $O(N^2)$ in the online inference phase. The time complexity of the embedding lookup layer is $O(N)$. Then, the attention score is calculated by $QK^\top \odot M$ with complexity $O(D_Q N^2)$, where D_Q is the dimension of Q. Assume we have M sub-networks. The complexity of masked self-attention is $O(MD_Q N^2)$. In the topical attention, the calculation process is very similar to traditional self-attention, only replacing Q with a task-related vector. So it does not change the time complexity of BERT. Finally, to get output representation, subsequent softmax and scalar-vector multiplication hold $O(D_V N)$ complexity, where D_V is the dimension of V for the topical attention. As such, the overall time complexity of Syntax-BERT is $O(N) + O(MD_Q N^2) + O(D_V N) = O(MD_Q N^2)$. When M is small, the model has the same time complexity as vanilla BERT. Moreover, as the sub-networks are usually very sparse, the time complexity can be further improved to $O(MD_Q E)$ by a sparse implementation. Here $E \ll N^2$ denotes the average number of edges in a sub-network.

5.2 Case Study

We select the sentence "John slipped in front of anyone who was there" in the CoLA dataset for case study. The task is to examine if a sentence conforms to English grammar. This sentence should be classified as negative since we use *everyone* instead of *anyone*. Syntax-Transformer classifies it correctly, but the vanilla transformer gives the wrong answer.

As visualized in Figure 2(a), the relationship between word pair *("anyone", ".")* has been highlighted in one of the sub-networks, and the corresponding topical attention score for this sub-network in Syntax-Transformer is also very high. This shows a good explainability of Syntax-Transformer by correctly identifying the error term "anyone", following a rule that "anyone" is seldom matched with the punctuation ".". However, a vanilla Transformer shows less meaningful self-attention scores, as illustrated in Figure 2(b). We give a briefing here, and please refer to the appendix for a complete description.

6 Conclusion

In this paper, we present Syntax-BERT, one of the first attempts to incorporate inductive bias of syntax trees to pre-trained Transformer models like BERT. The proposed framework can be easily plugged into an arbitrary pre-trained checkpoint, which underlines the most relevant syntactic knowledge automatically for each downstream task. We evaluate Syntax-BERT on various model backbones, including BERT, RoBERTa, and T5. The empirical results verify the effectiveness of this framework and the usefulness of syntax trees. In the future, we would like to investigate the performance of Syntax-BERT by applying it directly to the large-scale pre-training phase. Moreover, we are aiming to exploit more syntactic and semantic knowledge, including relation types from a dependency parser and concepts from a knowledge graph.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *NAACL-HLT*, pages 1315–1325.

Nezihe Merve Gürel, Hansheng Ren, Yujing Wang, Hui Xue, Yaming Yang, and Ce Zhang. 2019. An anatomy of graph neural networks going deep via the lens of mutual information: Exponential decay vs. full preservation. *arXiv preprint arXiv:1910.04499*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *NAACL-HLT*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Xinzhou Jiang, Zhenghua Li, Bo Zhang, Min Zhang, Sheng Li, and Luo Si. 2018. Supervised treebank conversion: Data and approaches. In *ACL (Volume 1: Long Papers)*, pages 2706–2716.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.

Xuan-Phi Nguyen, Shafiq Joty, Steven Hoi, and Richard Socher. 2020. Tree-structured attention with hierarchical accumulation. In *ICLR*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Devendra Singh Sachan, Yuhao Zhang, Peng Qi, and William Hamilton. 2020. Do syntax trees help pre-trained transformers extract information? *arXiv preprint arXiv:2008.09084*.

Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. In *ICLR*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *ICLR*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Emma Strubell, Pat Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *EMNLP*.

Dhanasekar Sundararaman, Vivek Subramanian, Guoyin Wang, Shijing Si, Dinghan Shen, Dong Wang, and Lawrence Carin. 2019. Syntax-infused transformer and bert models for machine translation and natural language understanding. *arXiv preprint arXiv:1911.06156*.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL-HLT*, pages 1112–1122.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, pages 1785–1794.

Yunlun Yang, Yunhai Tong, Shulei Ma, and Zhi-Hong Deng. 2016. A position encoding convolutional neural network based on dependency tree for relation classification. In *EMNLP*, pages 65–74.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. Fast and accurate shift-reduce constituent parsing. In *ACL*.

A Detailed settings

Here we provide detailed settings for reproduction. The open-source code will be released when this paper is officially published.

A.1 Stanford Sentiment Treebank

For raw Transformers, the number of layers is set as 12 and hidden dimension for each intermediate layer is set as 512. The probability of dropout is 0.1, and the hidden dimension of the final fully-connected layer is 2000. The word embedding vectors are initialized by GloVe (glove.840B.300d[3]) (Pennington et al., 2014) and fine-tuned during training. We use Adam optimizer with an initial learning rate 1e-4.

A.2 Natural Language Inference

For raw Transformers, we set layer number as 12, the hidden dimension of intermediate layers as 512, dropout ratio as 0.15, and the dimension of fully connected layer before Softmax activation as 2000. Learning rate is initialized as 5e-4, and Adam optimizer is used along with exponential learning rate decay of 0.9.

B Connection to GNN

A Transformer layer can be viewed as a special kind of Graph Neural Network (GNN), where each node represents for a word and all nodes construct a complete graph. To improve training speed and generalization ability, there are some previous works that advocate sparse architectures. For instance, Sparse Transformer (Child et al., 2019) separates the full self-attention operation across several steps of attention for image classification. Star-Transformer (Guo et al., 2019) sparsifies the architecture by shaping the fully-connected network into a star-shaped structure consisting of ring connections and radical connections. In the architecture of Syntax-BERT, we also introduce sparsity to the complete graph network by decomposing it into multiple sub-networks. The most salient part of our approach is that the inductive bias is designed by syntax tree, which is a crucial prior for NLP tasks. In addition, as shown previously in Table 5, a random decomposition of the network also result in moderate performance enhancement. Similar phenomena is also reported in the image classification scenario with Graph Convolutional Network (GCN) (Gürel et al., 2019).

---

[3] https://nlp.stanford.edu/projects/glove/

DISK-CSV: Distilling Interpretable Semantic Knowledge with a Class Semantic Vector

Housam Khalifa Bashier Babiker [1], **Mi-Young Kim**[2], **Randy Goebel**[1]
Alberta Machine Intelligence Institute
[1] Department of Computing Science, University of Alberta
[2] Department of Science, Augustana Faculty, University of Alberta
{khalifab, miyoung2, rgoebel}@ulaberta.ca

Abstract

Neural networks (NN) applied to natural language processing (NLP) are becoming deeper and more complex, making them increasingly difficult to understand and interpret. Even in applications of limited scope on fixed data, the creation of these complex "black-boxes" creates substantial challenges for debugging, understanding, and generalization. But rapid development in this field has now lead to building more straightforward and interpretable models. We propose a new technique (DISK-CSV) to distill knowledge concurrently from any neural network architecture for text classification, captured as a lightweight interpretable/explainable classifier. Across multiple datasets, our approach achieves better performance than the target black-box. In addition, our approach provides better explanations than existing techniques.

1 Introduction

Deep Neural Networks (DNNs) are popular in many applications, including computer vision and natural language processing. However, two major factors still require attention: (1) understanding the classifier's prediction for the end-users to develop trust in the model, and to enable machine learning engineers to refine it. This is especially true for high-stakes domains such as clinical decision support. There has been attention on models that attempt to make a neural network explainable, for instance, (Sundararajan et al., 2017), and (Ribeiro et al., 2016), which create post-hoc explanations to support explainability. Secondly, (2) artificially high numbers of parameters make the inference time expensive.

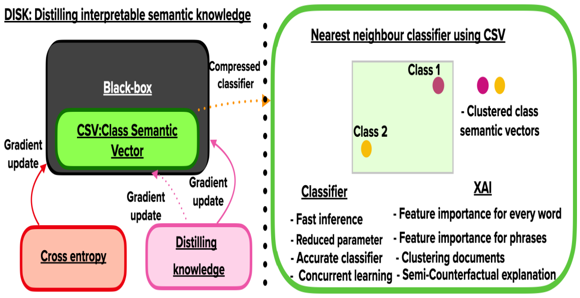


Figure 1: Our proposed model.

Neural networks tend to be deep, with millions of parameters. For example, GPT-2 (Radford et al., 2019) needs over 1.5 billion parameters. As a result, they are compute-intensive, thus making it difficult to deploy in real-world applications. We here propose a model-agnostic interpretable knowledge distillation method for neural network text classification.

As shown in Figure 1, we learn a class semantic vector for each output class, concurrently when training the black box. We then use the semantic vectors to create the nearest neighbor classifier (compressed interpretable/explainable classifier) from the black-box version. Knowledge distillation refers to the process of transferring the implicit knowledge learned by a teacher model to a student model (Liu and Matwin, 2018). Dark knowledge refers to the salient information hidden in the predicted probabilities for all classes, which are more informative than the predicted classes themselves. Our contributions can be summarized as follows:

- We propose a knowledge distillation method where dark knowledge can be learned concurrently by a student model, while building a black-box model.

- We propose an interpretable classifier, which provides a user explanation for predicting a single class label.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3021–3030
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

- We integrate a clustering technique within our interpretable classifier model.

- We provide an interactive explanation mode, where users can directly request a word or a phrase query and receive feedback.

- Our smaller model shows even better performance than the original black-box, with drastically reduced hyper-parameters. That smaller model can be deployed as an on-line service in real-time applications in resource-restricted devices.

2 Related Work

This work has connection with research on explainablity and model compression.

Explainability. Most of the existing explainable AI (XAI) techniques for Natural Language Processing text classification focus on assigning a score to each word in the document w.r.t. predicted class, typically using gradient-based or perturbation-based methods (Arras et al., 2017; Sundararajan et al., 2017; Shrikumar et al., 2017a; Bach et al., 2015). The most popular technique for model-agnostic explanation is LIME (Ribeiro et al., 2016), which focuses on creating an interpretable classifier by approximating it locally, with a linear model.

The main drawback of these methods is that those explanations are not faithful to the target model (Rudin, 2018). There are other methods, which focused on constructing a self-explainable network (Bastings et al., 2019) and (Lei et al., 2016). These techniques have limited explanations and thus do not explain phrases. Our work is different from post-hoc and self-explainable approaches as it attempts to learn an explainable smaller classifier concurrently with the target black-box model. Our explanations are also generated from the interpretable classifier itself, without extra calculation as in post-hoc techniques.

Model compression. A variety of research devoted their efforts to compressing large networks to accelerate inference, transfer, and storage. One of the earliest attempts focused on pruning unimportant weights (LeCun et al., 1990). Other methods focused on modifying devices to improve floating point operations (Tang et al., 2018). In contrast, some works focused on quantizing neural networks (Wu et al., 2018). Other investigations have focused on knowledge distillation, i.e., the ability to transfer the knowledge from a larger model to a smaller model (Ba and Caruana, 2014) and (Hinton et al., 2015). However, the main drawbacks of the methods mentioned above are that: (1) they only work with pre-trained networks, (2) the compressed models are still treated as black-box, and (3) the compression techniques require another training step or additional computation which complicates the process. In contrast, we concurrently transfer the knowledge from the black-box into a smaller interpretable model.

3 DISK-CSV

In a text classification task, an input sequence $x = \langle x_1, ..., x_l \rangle, x_i \in \mathbb{R}^d$, where l is the length of the input text and d is the vector dimension, is mapped to a distribution over class labels using a parameterized θ neural network model (e.g., a Long Short Term Memory network (LSTM), Transformer, etc.), which we denote as $\mathcal{F}(x; \theta)$. The output y is a vector of class probabilities, and the predicted class $\hat{y}$ is a categorical outcome, such as an entailment decision. In this work, we are interested in learning a simpler compressed nearest neighbor classifier (e.g., easy-to-explain its prediction) from any neural network model, but concurrently, while training the larger model. We refer to the large model (black-box) as $\mathcal{T}$ and the smaller interpretable/explainable model as $\mathcal{S}$.

We call our method **DISK-CSV** - **D**istilling **I**nterpretable **S**emantic **K**nowledge with a **C**lass **S**emantic **V**ector. In the next subsections we provide the following details for **DISK-CSV**: (a) how to distill knowledge from $\mathcal{T}$ into $\mathcal{S}$, (b) how to construct interpretable representations for $\mathcal{S}$, and (c) how to interact with the model to achieve better explainability (e.g., by clustering data, explaining essential phrases, and providing a semi-counterfactual explanation). Neural networks learn by optimizing a loss function to reflect the true objective of the end-user. For $\mathcal{S}$, our objective is to generalize in the same way as $\mathcal{T}$ and approximate an explanation for each prediction. To demonstrate our idea, we show how we can learn $\mathcal{S}$ concurrently with a long short-term memory network (LSTM) and then discuss how it can be generalized to different types of architectures for text classification. An LSTM network processes the input word by word, and at time-step t, the memory c_t and the hidden state h_t are updated. The last state $h_l \in \mathbb{R}^d$ is fed into a linear layer with parameters

$\boldsymbol{W} \in \mathbb{R}^{d\times k}$ which gives a probability distribution over k classes:

$$p(y|x) = \frac{\exp(h_l \cdot \boldsymbol{W})}{\sum_{i=1}^{k} \exp(h_l \ \cdot \boldsymbol{W})_k}, \qquad (1)$$

The classifier uses cross-entropy loss to penalize miss-classification as $\mathcal{L}^{classifier} = -\frac{1}{k}\sum_{i=1}^{k} r_i \log(y_i)$, where $r \in \mathbb{R}^k$ is the one-hot represented ground truth and r_i is the target probability (0 or 1) for class i. The network's weights are updated via back-propagation when training the black-box. We intend to augment the neural nets that typically use embeddings to represent discrete variables (e.g., words) as continuous vectors. Words that have a similar context will have similar meanings. The simplest form of concurrent knowledge distillation is to transfer the knowledge from the embedding space of $\mathcal{T}$ into a k-**C**lass **S**emantic **V**ectors (**CSV**s) $v_i \in v$, where the dimension of each v_i is equal to the dimension of the embedding vector x_i, and k is the number of target classes. In other words, for each class label, we would like to learn a vector that captures the semantic information related to that class from the embedding layer.

These semantic vectors have the following properties: (1) Each vector v_i should capture/encode the semantics about the class i from the black-box; (2) These vectors are used by the nearest neighbor classifier for the prediction of the correct class label; (3) By using cosine similarity, we can compute the contribution of each word in the input with the corresponding v_i to the class i; (4) These vectors add another level of abstraction by explaining the feature importance of a phrase that expands a single word, and (5) The weights of the CSV are initialized in the same way we initialize the embedding layer and adjusted via back-propagation. We reformulate the optimization of $\mathcal{T}$ to update/adjust the weights of the **CSV**s as follows:

$$\mathcal{L} = \mathcal{L}^{classifier} + \lambda_1 \overbrace{\left(1 - \frac{\bar{x} \bullet tanh(v_{\hat{y}})}{\|\bar{x}\| \, \|tanh(v_{\hat{y}})\|}\right)}^{Semantics} \\ + \underbrace{\lambda_2 \left(1 - \frac{h_l \bullet tanh(v_{\hat{y}})}{\|h_l\| \, \|tanh(v_{\hat{y}})\|}\right)}_{Hidden-knowledge} \\ -\lambda_3 \left(\overbrace{D}^{Vectors-separation} \right) \qquad (2)$$

where D is the pairwise Euclidean distance defined as $D = \frac{\sum_i \sum_j ((v_i - v_j)(v_i - v_j)^T)}{2}$, $\hat{y}$ is the index of the predicted class with the highest probability, and $\{\lambda_1, \lambda_2, \lambda_t r3\}$ are used to weight the importance of the terms. In what follows, we discuss the new terms added to the optimization problem.

Capturing semantics: The second term of Equation 2 is the second loss function we use, which attempts to encode the information of semantically consistent sentences in a single **CSV** $v_{\hat{y}}$. An obvious way to learn semantic information is to minimize the cosine distance between the average of the embedding $\bar{x}$ of the input sentence x and the predicted class semantic vector $v_{\hat{y}}$. This objective will ensure that the vector v_i captures the semantics of consistent inputs to encourage semantic consistency.

Hidden knowledge extraction: The last hidden state h_l in recurrent nets is typically used by the output layer as the feature vector to predict the class label. As a result, the salient information learned by $\mathcal{T}$ is encoded in this feature vector. To distill this knowledge and enrich the representation of v_i so that $\mathcal{S}$ generalizes well, we again minimize the cosine distance between the class semantic vector v_i and the last hidden state h_l in the third loss function, which is the third term in Equation 2. This objective allows the model $\mathcal{S}$ to generalize similarly to the black-box $\mathcal{T}$. The only constraint here is that the dimension of h_l must be the same as that of x_i so that we can minimize the cosine distance, i.e., $h_l \in \mathbb{R}^d$.

Vector-separation: Our ultimate goal is to create a simple interpretable nearest neighbor classifier $\mathcal{S}$ from the black-box. Therefore, we want to make sure that the **CSV**s are well separated from each other so that the cosine distance is maximum between them. To address this problem, we maximize the pairwise Euclidean distance between these vectors using the fourth term in Equation 2.

3.1 The smaller interpretable classifier $\mathcal{S}$ based on CSV

Our smaller model $\mathcal{S}$ is the nearest neighbor classifier, which relies mainly on the semantic information encoded in the vectors v learned via back-propagation when training $\mathcal{T}$. The model $\mathcal{S}$ takes the input sentence and computes the average $\bar{x}$ of the input embedding. Then we compute the cosine distance between $\bar{x}$ and each v_i. Finally, the target class is decided as the index i of the v_i with the lowest cosine distance. Besides, this classifier is interpretable, i.e., we can understand the mechanism of making a prediction. It can also be easily transferred or stored. The smaller model extracts the semantics from the larger model concurrently when training the black-box model. The algorithm is summarized in Figure 2.

```
Input: Sentence embedding $x$, CSVs $v$, $Dist = 5$
Output: Predicted class $\hat{y}$
$\bar{x}$ = average˙emebdding($x$)
for Each vector $v_i \in v$ do
    $tmpDist$ = CosineDistance ($v_i, \bar{x}$)
    if $tmpDist < Dist$ then
        $\hat{y} \leftarrow i$
        $Dist \leftarrow tmpDist$
    end if
end for
```

Figure 2: Our smaller model $\mathcal{S}$ using **CSV**.

3.2 Explainability

Our model $\mathcal{S}$ provides four levels of explanations for text classification: (1) Word feature importance, (2) Document clustering, (3) End-user interaction through phrase feature importance, and (4) Semi-counterfactual explanation.

- **Word feature importance**: To understand the contribution of each word w.r.t. predicted class $\hat{y}$, we rely on the semantic similarity between each x_i and the nearest class vector $v_{\hat{y}}$. A word with high semantic similarity with the predicted $v_{\hat{y}}$ will have a high contribution to the predicted class. To understand the semantic contribution of each word in the input, we calculate the semantic similarity of every word to $v_{\hat{y}}$ using cosine similarity between the embedding vector of the input word and $v_{\hat{y}}$.

- **Document clustering**: Every text instance is clustered around its class semantic vector by computing the mean (x-axis) and the standard deviation (y-axis) of the elements in the vector $(\frac{\bar{x}+tanh(v_{\hat{y}})}{2})$ for each $\bar{x}$, where $v_{\hat{y}}$ is the nearest **CSV** to the input document. We found that the 2-D points (mean, standard deviation) of the elements in the vector $(\frac{\bar{x}+tanh(v_{\hat{y}})}{2})$ for the instances belonging to a specific class are close to each other and far from those of the instances belonging to other classes. The merit is that we do not need to use a clustering algorithm.

- **End-user interaction through phrase feature importance**: Word feature importance is sometimes not enough to explain a model's prediction. The end-user might also be interested in querying the classifier to answer different types of questions. For example, in the situation where the model shows the feature importance (in sentiment classification) of each individual word "not," "too," and "bad," an end-user might also be interested in the importance of the phrase "not too bad," which cannot be calculated just by merging the three different feature importance values. Our approach is capable of giving feedback to the user's query about a phrase. To obtain the feature importance for a phrase, we average the embedding vectors of the words in the phrases and then compute the cosine similarity w.r.t. the predicted **CSV** $v_{\hat{y}}$.

- **Semi-counterfactual explanation**: Our approach is also capable of providing a semi-counterfactual explanation, i.e., explaining a semi-casual situation (i.e., what kind of features prevent the classifier from changing the prediction to another class). We can provide a feature importance value w.r.t. non-predicted classes by calculating the cosine similarity between the embedding vector of each word/phrase and the class semantic vector of a non-predicted class). Through this semi-counterfactual explanation, the user can reason that "if the feature X had not occurred, the class prediction would have been changed."

3.3 Generalizing to other models

Our method can be adapted to a variety of architectures such as Bi-LSTM, GRU, and RNNs, as it requires access to only the last hidden state (feature vector) and the embedding layer from the network. A further restriction is that the feature vector used in the output layer must have the same dimension as the embedding feature vector. For the Transformer, to handle the dimensionality issue, we average the Transformer's representations before the output layer as the feature vector.

4 Experiments

4.1 Datasets

The summary of the datasets is shown in Table 1.

IMDB reviews were proposed by (Maas et al., 2011) for sentiment classification from movie reviews. It consists of two classes, i.e., positive and negative sentiments.

AGnews was proposed by (Zhang et al., 2015a) for researchers to test machine learning models for news classification. It consists of four classes (sports, world, business, and sci/tech).

DBpedia ontology classification dataset proposed by (Zhang et al., 2015b) consists of 15 non-overlapping ontology classes [1].

HealthLink constructed by Alberta Health Services, Canada. It contains a set of text transcripts written by registered nurses while talking with callers to the Tele-Health service in real-time. It consists of 2 classes ("go to hospital" and "home care"), and each class can be sub-categorized into sub-classes. This dataset will be available based on request.

Data set	Classes	Max length	Train size	Test size	Vocabulary size
IMDB	2	50	25000	25000	10000
HealthLink	2	20	60475	15119	23174
DBpedia	15	32	5600	63000	50002
AGnews	4	20	102080	25520	59706

Table 1: Summary of the datasets used in our experiments

4.2 Baselines

We compare our approach with several models for text classification including Transformers (Vaswani et al., 2017), IndRNN (Li et al., 2018), BLSTM (Zhou et al., 2016), hierarchical attention (Yang et al., 2016), LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014).

Transformer employs a multi-head self-attention mechanism based on scaled dot-product attention. We use only the encoder layer, and average the new representations before arriving in the classification's output layer.

IndRNN is an improvement over RNNs, where neurons in the same layer are independent of each other and connected across layers. We use the last hidden state as the feature vector.

Bi-LSTM employs an attention-based bidirectional mechanism on the LSTM network, which captures the salient semantic information (word-attention) in a sentence. These attentions enable the network to attend differently to more and less critical content when learning the representation. The last hidden state is used for classification.

Hierarchical attention provides two levels of attention mechanisms applied to the word and sentence level. In this paper, we use a sentence level-attention mechanism applied on a Bi-LSTM. The feature vector for classification is based on aggregating the hidden representation values (following the authors' implementation).

LSTM and GRU process the input word by word, and the last hidden state is used as the feature vector for classification.

4.3 Network configuration and training

We tokenize sentences and use the top N words that appeared in every instance for the vocabulary size. We did not use any pre-trained embeddings, and thus we randomly initialized the embedding layer. We also randomly initialized the **CSV**s. We did not use any hyper-parameter tuning on the validation as we are not focusing on achieving state-of-the-art predictive accuracy. Instead, we want to show that our method can achieve similar/better performance to the black-box, and provides a better explanation than existing approaches. The word embedding, semantic vector, and feature vector (at the output layer) dimensions are 128. For training each network, we use the Adam optimizer (Kingma and Ba, 2017) with a batch size of 64 and a learning rate of 0.0001. We also used a dropout with a probability 0.5.

4.4 Results and analysis

4.4.1 Classifier performance

We trained six different models (architectures) on four datasets. We have tried different values as the weight of each proposed loss term. The results depicted in Table 2 show that our semantic distillation approach captures more useful information from the training data than the baseline black-box. Our smaller model outperforms the black-boxes on all datasets, achieving better performance than the black-box. The new optimization problem does not affect the performance of the black-box model (see BBO (Black-Box with our new Objective function) in Table 2).

4.4.2 Explainability

In this part of our experiments, we focus on local explanations for text classification, i.e., explaining the output made by our proposed nearest neighbor classifier using **CSV** for an individual instance. Local explanations should exhibit high local fidelity, i.e., they should match the underlying model behavior. We evaluate our technique against the following methods:

---

[1] Because of computation time, we experimented with only a small number of samples.

	IMDB				**AGnews**				**Dpedia**				**HealthLink**			
Transformer (Vaswani et al., 2017)																
	F1	**Precision**	**Recall**	**Accuracy**	**F1**	**Precision**	**Recall**	**Accuracy**	**F1**	**Precision**	**Recall**	**Accuracy**	**F1**	**Precision**	**Recall**	**Accuracy**
Black-box	0.7703	0.7703	0.7703	0.7703	0.8794	0.8798	0.8796	0.8796	0.8653	0.8655	0.8655	0.927	0.6642	0.6645	0.6641	0.6647
BBO	0.7785	0.7787	0.7786	0.7786	0.8831	0.8836	0.8835	0.8835	0.8799	0.8802	0.8799	0.943	0.6887	0.6896	0.6887	0.6894
DISK-CSV	**0.8117**	**0.8187**	**0.8187**	**0.8187**	**0.9038**	**0.9039**	**0.9041**	**0.9041**	**0.8806**	**0.8811**	**0.8809**	**0.9438**	**0.7216**	**0.722**	**0.722**	**0.7216**
Attention-based Bi-LSTM (Zhou et al., 2016)																
Black-box	0.7961	0.7993	0.7966	0.7966	0.8887	0.8888	0.887	0.8888	0.843	0.8443	0.8434	0.9037	0.6706	0.6706	0.6705	0.6708
BBO	0.798	0.7999	0.7983	0.7983	0.8929	0.8941	0.8928	0.8927	0.8772	0.8774	0.8772	0.9399	0.6704	0.6705	0.6706	0.6705
DISK-CSV	**0.8025**	**0.8025**	**0.8025**	**0.8025**	**0.8956**	**0.8955**	**0.896**	**0.896**	**0.8812**	**0.8816**	**0.8815**	**0.9445**	**0.7207**	**0.7207**	**0.7209**	**0.7208**
IndRNN (Li et al., 2018)																
Black-box	0.776	0.7761	0.776	0.776	0.8773	0.878	0.8769	0.8769	0.8763	0.8765	0.8765	0.9391	0.6808	0.6814	0.6813	0.6808
BBO	0.7805	0.7858	0.7814	0.7814	0.8845	0.8847	0.8847	0.8848	0.8845	0.8889	0.888	0.9515	0.6808	0.6814	0.686	0.6869
DISK-CSV	**0.8018**	**0.8022**	**0.802**	**0.802**	**0.9025**	**0.9026**	**0.9028**	**0.9028**	**0.8887**	**0.889**	**0.8889**	**0.9524**	**0.7162**	**0.7184**	**0.7174**	**0.7164**
Hierarchical recurrent net (Yang et al., 2016)																
Black-box	0.7917	0.7919	0.7917	0.7917	0.8845	0.8855	0.8846	0.8846	0.847	0.8475	0.8467	0.9073	0.6708	0.6708	0.609	0.671
BBO	0.7808	0.7844	0.7813	0.7814	0.8874	0.8876	0.8874	0.8876	0.8709	0.871	0.8709	0.933	0.6829	0.6833	0.6833	0.6829
DISK-CSV	**0.8146**	**0.8146**	**0.8146**	**0.8146**	**0.9013**	**0.9013**	**0.9016**	**0.9016**	**0.8794**	**0.8796**	**0.8797**	**0.9425**	**0.7156**	**0.7158**	**0.7159**	**0.7157**
LSTM (Hochreiter and Schmidhuber, 1997)																
Black-box	0.745	0.7456	0.7452	0.7452	0.8711	0.8712	0.8714	0.8715	0.6597	0.7187	0.6445	0.6905	0.5922	0.6155	0.6044	0.6006
BBO	0.7461	0.7488	0.7466	0.7466	0.8745	0.875	0.8745	0.875	0.7993	0.8129	0.797	0.8593	0.6127	0.6718	0.6717	0.6712
DISK-CSV	**0.7912**	**0.7913**	**0.7912**	**0.7912**	**0.9005**	**0.9005**	**0.9009**	**0.9009**	**0.8657**	**0.8667**	**0.8662**	**0.928**	**0.7171**	**0.7178**	**0.7177**	**0.7171**
GRU (Cho et al., 2014)																
Black-box	0.748	0.7493	0.7483	0.7483	0.8709	0.8708	0.8711	0.8712	0.6537	0.7006	0.6442	0.6902	0.6106	0.6266	0.6187	0.6165
BBO	0.74483	0.753	0.7493	0.75	0.8847	0.885	0.8851	0.8906	0.8193	0.8123	0.812	0.875	0.6478	0.6572	0.6522	0.6562
DISK-CSV	**0.8069**	**0.8069**	**0.8069**	**0.8069**	**0.9046**	**0.9047**	**0.9049**	**0.9049**	**0.8831**	**0.8834**	**0.8833**	**0.9041**	**0.7154**	**0.7159**	**0.7158**	**0.7154**

Table 2: Comparison of our test performances with the baseline neural architectures on four datasets. Our nearest neighbour classifier achieves better performance than the black-box models. For the black-box models, we followed the implementation proposed by the authors of each baseline.

- **Random**. A random selection of words from the input sentence.
- **LIME** (Ribeiro et al., 2016) is a model-agnostic approach which involves training an interpretable model such as a linear model on instances created around the specific data point by perturbing the data. We evaluated by training the linear classifier using ~ 5000 samples.

We show the effectiveness of our method in explaining the prediction on three architectures (Transformer, IndRNN and hierarchical attention network) in Figures 3-8.

Automatic evaluation. We use model-agnostic evaluation metrics to demonstrate the effectiveness of our approach. (Nguyen, 2018) found that human evaluation correlates moderately with automatic evaluation metrics for local explanations. Hence, in our experiments, we use the idea of automatic evaluation to verify whether or not our explanations are faithful to what the model computes. We measure the local fidelity by deleting words in the order of their estimated importance for the prediction, then evaluate the change in F1 score w.r.t. the predicted class when no word is deleted. This type of evaluation is similar to other metrics used for model interpretation (Nguyen, 2018; Arras et al., 2017) except that we use F1 instead of classification accuracy. Results are shown in Figures 3-4. We obtained the plots by measuring the effect of word deletions and reporting the F1 when the classifier prediction changes. A larger drop in F1 indicates that the method could identify the words contributing most towards the predicted class by our classifier. Through Figures 3, 4 and 5, we can clearly see that our approach is capable of identifying the most salient features better than LIME. Please note that LIME requires probabilities (as the classifier's output), and hence we convert the outputs made by our nearest neighbor into valid probabilities.

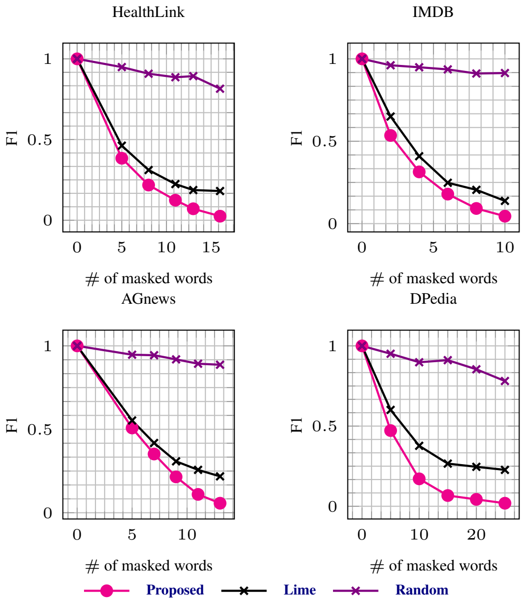


Figure 3: Change of F1 according to the number of masked important words. (Teacher model: Transformer)

Change in log-odds. Another automatic metric for evaluating explainability methods is to observe the change in the log-odds ratio (for the output probabilities). This metric has been used for a model's explanations (Shrikumar et al., 2017b; Chen et al.,

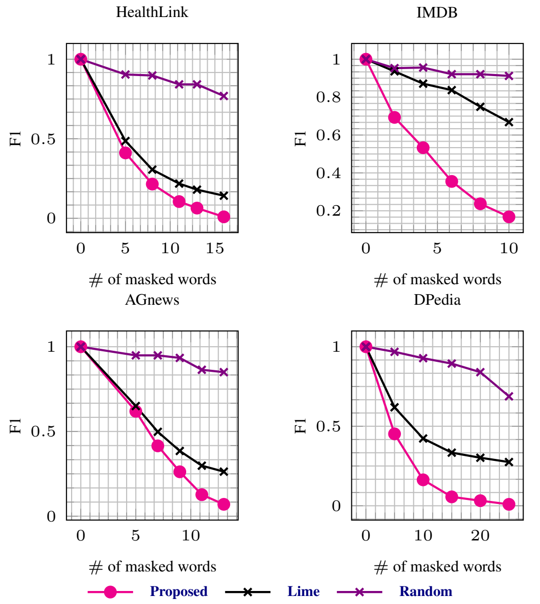


Figure 4: Change of F1 according to the number of masked words. (Teacher model: INDRNN)

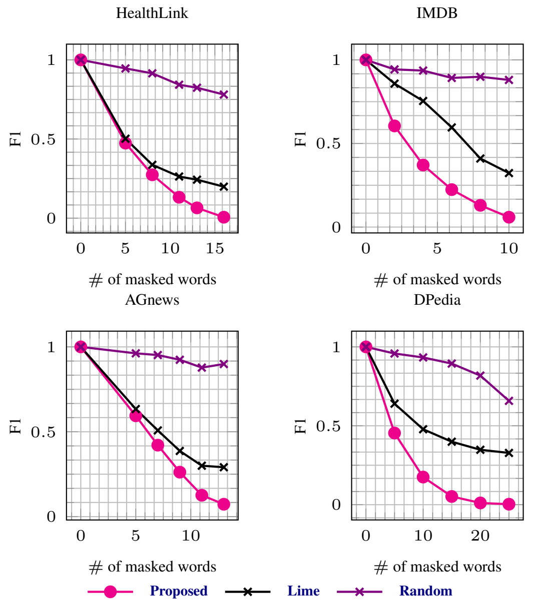


Figure 5: Change of F1 according to the number of masked words. (Teacher model: Hierarchical attention network)

2018). We normalize the cosine distance into valid probability distributions. This metric requires no knowledge of the underlying feature representation, and it requires access to only the instances. A log-odds ratio is a fine-grained approach, as it uses actual probability values instead of the predicted label as used in the previous experiment. But like the previous experiment, instead of tracking the change in F1, we observe the change in probabilities. We mask the top k features ranked by semantic similarity, and zero paddings replace those masked words. We then feed the input and measure the drop of the value between the target class's probability when no word is deleted and when k words are removed. Results are shown in Figures 6-8 reveal the effectiveness of our approach in capturing the words that affect the classifier's prediction. The experimental results show that our method delivers more insightful explanations than LIME.

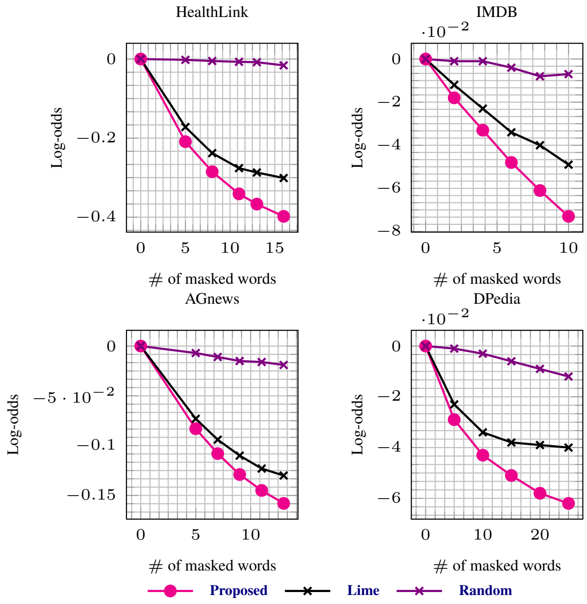


Figure 6: Change of log-odds according to the number of masked words. Lower log-odds scores are better. (Teacher model:Transformer)

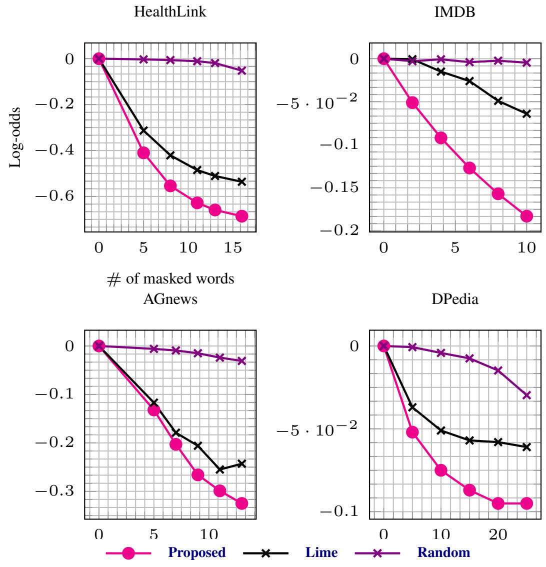


Figure 7: Change of log-odds according to the number of masked words. (Teacher model: INDRNN)

Interactive explanations. In some cases, end-users are interested in understanding the contribution of phrases instead of words. In addition, an end-user might be interested in understanding the

Example	Pos	Neg	Example	Home care	Go to hospital	Example	Home care	Go to hospital
Good	0.756	-0.752	Cough	-0.984	0.984	Fever	0.933	-0.932
Not good	-0.151	0.144	Bad cough	-0.993	0.993	Bad fever	-0.962	0.952
Very good	0.877	-0.878	Cough+sore throat	-0.995	0.995	fever+headache	0.929	-0.929
Sucks	-0.607	0.607	Chest pain	-0.959	0.958	Cold	0.168	-0.170
Not sucks	0.688	-0.681	Mild chest pain	-0.861	0.861	Cold+chest pain	-0.934	0.934
Just sucks	-0.825	0.828	Chest pain+high blood pressure	-0.991	0.991	Cold+fever	-0.532	0.961
Sucks but very good	0.255	-0.262	Breathing	-0.968	0.968	Blood pressure	-0.980	0.980
Heart-warming	0.335	-0.3444	Breathing difficulty	-0.992	0.991	Bad blood pressure	-0.990	0.990
Heart-warming+entertaining	0.538	-0.54	vomiting+breathing	-0.883	0.883	High blood pressure	-0.981	0.981

Table 3: XAI capability. Explaining word/phrase contributions and also providing contributions to other classes (semi-counterfactual explanation).

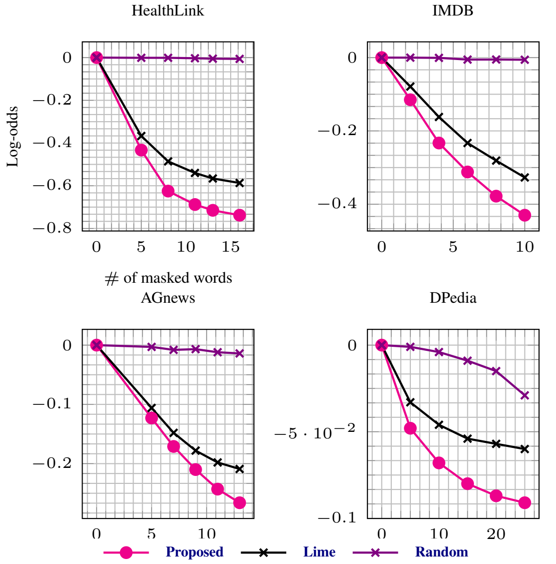
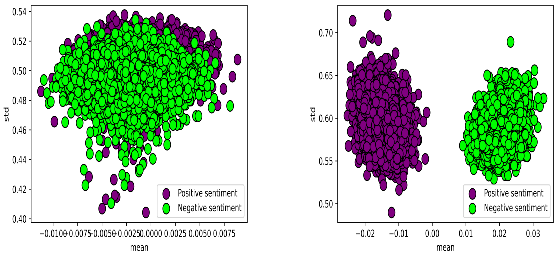


Figure 8: Change of log-odds according to the number of masked words. Lower log-odds scores are better. (Teacher model: Hierarchical attention network)

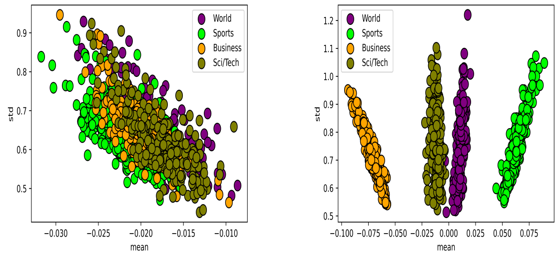
contribution of a word/phrase w.r.t. other classes, not only to the predicted class (semi-counterfactual explanation). Our model's results in support of these interests are shown in Table 3. Our technique can identify the contributions of phrases instead of words, and it can provide evidence w.r.t. other classes. For example, our method can recognize that "bad cough" has a stronger semantic contribution than "cough" w.r.t. the label "going to the hospital." Our approach can also recognize the difference between "mild chest pain" and "chest pain." In sentiment analysis, our method understands that "good" contributes to a positive sentiment while "not good" contribute to negative sentiment. Also, note that "very good" contributes more importantly to a positive sentiment than "good."

Clustering textual data. Another feature of the **CSV** classifier is the ability to cluster documents via **CSV**s without using dimensionality reduction techniques such as PCA clustering algorithms. Results of document clustering based on the distilled knowledge from the Transformer on the IMDB and AGnews are shown in Figure 9 and 10. The clusters explain our classifier's behavior and hence provide a global explanation of the model's prediction. We also show the critical role of using the pairwise Euclidean distance in our classification by clustering sentences into their predicted classes.



(a) Without Euclidean distance loss. (b) With Euclidean distance loss.

Figure 9: Sentence clustering on the predicted class using pairwise distance vs. without pairwise distance. (Dataset:IMDB)



(a) Without Euclidean distance loss. (b) With Euclidean distance loss.

Figure 10: Sentence clustering on the predicted class using Euclidean distance vs. without Euclidean distance. (Dataset: AGnews)

Parameter reduction. We compare the number of parameters used by our nearest neighbor classifier and that of the black-box approach, using the HealthLink data in Table 4. The number of parameters used by our compressed classifier is less than that of each black-box. Our model relies only on the embeddings and the **CSV**s, and the rest of the layers are dropped. The number of parameters of the proposed classifier is the same for all architectures because our classifier has the same size of the embedding layer and **CSV**s on each black-box architecture. Our model also reduced the inference time from $0.037 - 0.085$ seconds to 0.007 seconds, as shown in Table 4.

Method	# of parameters	# of dropped parameters	Inference time
Transformer			
Black-box	3049602	83074	0.085
DISK-CSV	**2966528**		**0.007**
IndRNN			
Black-box	2991490	24962	0.037
DISK-CSV	**2966528**		**0.007**
Attention-based bi-LSTM			
Black-box	3229826	263298	0.056
DISK-CSV	**2966528**		**0.007**
Hierarchical recurrent network			
Black-box	3114754	148226	0.039
DISK-CSV	**2966528**		**0.007**

Table 4: Number of parameters used for black-box and our proposed model. We also compare the inference time.

Semantics. We compare our proposed method's performance with and without capturing the semantic information (Equation 2). Results depicted in Table 5 show the importance of encoding semantic into the class discriminative vector.

	Proposed				Without semantic			
	F1	**Accuracy**	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**
Dpedia	0.8806	0.9438	0.8811	0.8809	0.0425	0.624	0.0693	0.0582

Table 5: The impact of first term in Equation 2 on the classifier's performance

Analyzing words. We are interested in what kind of words contribute most to the class prediction. For this analysis, we exploit the word-level sentiment annotation (Opinion Lexicon) provided by Liu [2] to track the top 10 words whose importance was the highest when predicting the sentiment class in the IMDB dataset. We evaluated the number of words contributing to each of the negative and positive sentiments on 1000 movie reviews. Table 6 shows that our approach can identify more salient words that lead to correct sentiment classification, i.e., our method can pick better sentiment lexicons than LIME.

	Proposed	Lime	Random
Positive sentiment	**597**	423	286
Negative sentiment	**382**	353	236

Table 6: The number of words in each sentiment class for 1000 samples from the test set.

4.5 Discussion

We have shown that semantic information can be extracted and used to create a simple interpretable/explainable classifier that performs better than the target black-box models. This simple classifier has the following proprieties:

- It captures the discriminative representations encoded by the black-box and encodes them in the **CSV**.
- For text classification, the distance is the lowest between the text input and the **CSV** of the correct class, and is high for the other **CSV**s of the incorrect classes.

5 Conclusion and Future Work

We have explored an approach to knowledge distillation concurrently from any black-box model to produce a simple, explainable classifier. The distilled model achieves better results than the original black-box models in terms of the model's performance. Also, we showed that our distilled model provides a better explanation than LIME.

We have also proposed new types of explanations: First, a user can query with any-length phrases and receive feedback about the phrase's contribution to the classes. Second, we also provide word(feature) importance to non-predicted classes, which can be used as a semi-counterfactual explanation. Third, we showed how we could cluster the documents without employing the existing clustering method.

In future work, we would like to extend this idea to pre-trained networks, and we also plan to more deeply investigate the value of counterfactual explanations.

Acknowledgements

We acknowledge support from the Alberta Machine Intelligence Institute (AMII), from the Computing Science Department of the University of Alberta, and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017. Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis in ACL*, pages 159–168.

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One*, 10(7):e0130140.

Joost Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable bi-

---

[2] `https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon`

nary variables. In *Proceedings of ACL*, pages 2963–2977.

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018. L-shapley and c-shapley: Efficient model interpretation for structured data. *ICLR 2019*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Diederik P. Kingma and Jimmy L. Ba. 2017. Adam: A method for stochastic optimization. cornell university library. *arXiv preprint arXiv:1412.6980*.

Yann LeCun, John S Denker, and Sara A Solla. 1990. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of EMNLP*, pages 107–117.

Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. 2018. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5457–5466.

Wang X. Liu, X. and S. Matwin. 2018. Improving the interpretability of deep neural networks with knowledge distillation. In *Proceedings of IEEE International Conference on Data Mining Workshops*, pages 905–912.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of ACL*, pages 142–150. Association for Computational Linguistics.

Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of NAACL-HLT, Volume 1 (Long Papers)*, pages 1069–1078.

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

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM.

Cynthia Rudin. 2018. Please stop explaining black box models for high stakes decisions. *32nd Conference on Neural Information Processing Systems (NIPS 2018), Workshop on Critiquing and Correcting Trends in Machine Learning.*

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017a. Learning important features through propagating activation differences. In *Proceedings of the International Conference on Machine Learning*, pages 3145–3153.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017b. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3145–3153. JMLR. org.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of International Conference on Machine Learning (ICML)*, page 3319–3328.

Raphael Tang, Ashutosh Adhikari, and Jimmy Lin. 2018. Flops as a direct optimization objective for learning sparse neural networks. *arXiv preprint arXiv:1811.03060*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Shuang Wu, Guoqi Li, Feng Chen, and Luping Shi. 2018. Training and inference with integers in deep neural networks. *arXiv preprint arXiv:1802.04680*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*, pages 1480–1489.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.

Attention Can Reflect Syntactic Structure
(If You Let It)

*Vinit Ravishankar† *Artur Kulmizev‡ Mostafa Abdou§
Anders Søgaard§ Joakim Nivre‡

† Language Technology Group, Department of Informatics, University of Oslo
‡ Department of Linguistics and Philology, Uppsala University
§ Department of Computer Science, University of Copenhagen

† `vinitr@ifi.uio.no`
‡ `{artur.kulmizev,joakim.nivre}@lingfil.uu.se`
§ `{abdou,soegaard}@di.ku.dk`

Abstract

Since the popularization of the Transformer as a general-purpose feature encoder for NLP, many studies have attempted to decode linguistic structure from its novel multi-head attention mechanism. However, much of such work focused almost exclusively on English — a language with rigid word order and a lack of inflectional morphology. In this study, we present decoding experiments for multilingual BERT across 18 languages in order to test the generalizability of the claim that dependency syntax is reflected in attention patterns. We show that full trees can be decoded above baseline accuracy from single attention heads, and that individual relations are often tracked by the same heads across languages. Furthermore, in an attempt to address recent debates about the status of attention as an explanatory mechanism, we experiment with fine-tuning mBERT on a supervised parsing objective while freezing different series of parameters. Interestingly, in steering the objective to learn explicit linguistic structure, we find much of the same structure represented in the resulting attention patterns, with interesting differences with respect to which parameters are frozen.

1 Introduction

In recent years, the attention mechanism proposed by Bahdanau et al. (2014) has become an indispensable component of many NLP systems. Its widespread adoption was, in part, heralded by the introduction of the Transformer architecture (Vaswani et al., 2017a), which constrains a soft alignment to be learned across discrete states in the input (self-attention), rather than across input and output (e.g., Xu et al., 2015; Rocktäschel et al., 2015). The Transformer has, by now, supplanted the popular LSTM (Hochreiter and Schmidhuber, 1997) as NLP's feature-encoder-of-choice, largely due to its compatibility with parallelized training regimes and ability to handle long-distance dependencies.

Certainly, the nature of attention as a distribution over tokens lends itself to a straightforward interpretation of a model's inner workings. Bahdanau et al. (2014) illustrate this nicely in the context of `seq2seq` machine translation, showing that the attention learned by their models reflects expected cross-lingual idiosyncrasies between English and French, e.g., concerning word order. With self-attentive Transformers, interpretation becomes slightly more difficult, as attention is distributed across words within the input itself. This is further compounded by the use of multiple layers and heads, each combination of which yields its own alignment, representing a different (possibly redundant) view of the data. Given the similarity of such attention matrices to the score matrices employed in arc-factored dependency parsing (McDonald et al., 2005a,b), a salient question concerning interpretability becomes: Can we expect some combination of these parameters to capture linguistic structure in the form of a dependency tree, especially if the model performs well on NLP tasks? If not, can we relax the expectation and examine the extent to which subcomponents of the linguistic structure, such as subject-verb relations, are represented? This prospect was first posed by Raganato et al. (2018) for MT encoders, and later explored by Clark et al. (2019) for BERT. Ultimately, the consensus of these and other studies (Voita et al., 2019; Htut et al., 2019; Limisiewicz et al., 2020) was that, while there appears to exist no "generalist" head responsible for extracting full dependency structures, standalone heads often specialize in capturing individual grammatical relations.

Unfortunately, most of such studies focused their

---

*Equal contribution. Order was decided by a coin toss.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3031–3045
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

experiments entirely on English, which is typologically favored to succeed in such scenarios due to its rigid word order and lack of inflectional morphology. It remains to be seen whether the attention patterns of such models can capture structural features across typologically diverse languages, or if the reported experiments on English are a misrepresentation of local positional heuristics as such. Furthermore, though previous work has investigated how attention patterns might change after fine-tuning on different tasks (Htut et al., 2019), a recent debate about attention as an explanatory mechanism (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019) has cast the entire enterprise in doubt. Indeed, it remains to be seen whether fine-tuning on an explicit structured prediction task, e.g. dependency parsing, can force attention to represent the structure being learned, or if the patterns observed in pretrained models are not altered in any meaningful way.

To address these issues, we investigate the prospect of extracting linguistic structure from the attention weights of multilingual Transformer-based language models. In light of the surveyed literature, our research questions are as follows:

1. Can we decode dependency trees for some languages better than others?
2. Do the same layer–head combinations track the same relations across languages?
3. How do attention patterns change after fine-tuning with explicit syntactic annotation?
4. Which components of the model are involved in these changes?

In answering these questions, we believe we can shed further light on the (cross-)linguistic properties of Transformer-based language models, as well as address the question of attention patterns being a reliable representation of linguistic structure.

2 Attention as Structure

Transformers The focus of the present study is mBERT, a multilingual variant of the exceedingly popular language model (Devlin et al., 2019). BERT is built upon the Transformer architecture (Vaswani et al., 2017b), which is a self-attention-based encoder-decoder model (though only the encoder is relevant to our purposes). A Transformer takes a sequence of vectors $\mathbf{x} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_n}]$ as input and applies a positional encoding to them, in order to retain the order of words in a sentence. These inputs are then transformed into query (Q), key (K), and value (V) vectors via three separate linear transformations and passed to an attention mechanism. A single attention head computes scaled dot-product attention between K and Q, outputting a weighted sum of V:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{1}$$

For multihead attention (MHA), the same process is repeated for k heads, allowing the model to jointly attend to information from different representation subspaces at different positions (Vaswani et al., 2017b). Ultimately, the output of all heads is concatenated and passed through a linear projection W^O:

$$H_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{2}$$

$$\text{MHA}(Q, K, V) = \text{concat}(H_1, H_2, ..., H_k)W^O \tag{3}$$

Every layer also consists of a feed-forward network (FFN), consisting of two Dense layers with ReLU activation functions. For each layer, therefore, the output of MHA is passed through a LayerNorm with residual connections, passed through FFN, and then through another LayerNorm with residual connections.

Searching for structure Often, the line of inquiry regarding interpretability in NLP has been concerned with extracting and analyzing linguistic information from neural network models of language (Belinkov and Glass, 2019). Recently, such investigations have targeted Transformer models (Hewitt and Manning, 2019; Rosa and Mareček, 2019; Tenney et al., 2019), at least in part because the self-attention mechanism employed by these models offers a possible window into their inner workings. With large-scale machine translation and language models being openly distributed for experimentation, several researchers have wondered if self-attention is capable of representing syntactic structure, despite not being trained with any overt parsing objective.

In pursuit of this question, Raganato et al. (2018) applied a maximum-spanning-tree algorithm over the attention weights of several trained MT models, comparing them with gold trees from Universal Dependencies (Nivre et al., 2016, 2020). They found that, while the accuracy was not comparable to that of a supervised parser, it was nonetheless higher than several strong baselines, implying that some structure was consistently represented. Clark et al. (2019) corroborated the same findings for BERT when decoding full trees, but observed that individual dependency relations were often tracked by specialized heads and were decodable with much higher accuracy than some fixed-offset baselines. Concurrently, Voita et al. (2019) made a similar observation about heads specializing in specific dependency relations, proposing a coarse taxonomy of head attention functions: *positional*, where heads attend to adjacent tokens; *syntactic*, where heads attend to specific syntactic relations; and *rare words*, where heads point to the least frequent tokens in the sentence. Htut et al. (2019) followed Raganato et al. (2018) in decoding dependency trees from BERT-based models, finding that fine-tuning on two classification tasks did not produce syntactically plausible attention patterns. Lastly, Limisiewicz et al. (2020) modified UD annotation to better represent attention patterns and introduced a supervised head-ensembling method for consolidating shared syntactic information across heads.

Does attention have explanatory value? Though many studies have yielded insight about how attention behaves in a variety of models, the question of whether it can be seen as a "faithful" explanation of model predictions has been subject to much recent debate. For example, Jain and Wallace (2019) present compelling arguments that attention does not offer a faithful explanation of predictions. Primarily, they demonstrate that there is little correlation between standard feature importance measures and attention weights. Furthermore, they contend that there exist *counterfactual* attention distributions, which are substantially different from learned attention weights but that do not alter a model's predictions. Using a similar methodology, Serrano and Smith (2019) corroborate that attention does not provide an adequate account of an input component's importance.

In response to these findings, Wiegreffe and Pinter (2019) question the assumptions underlying such claims. Attention, they argue, is not a *primitive*, i.e., it cannot be detached from the rest of a model's components as is done in the experiments of Jain and Wallace (2019). They propose a set of four analyses to test whether a given model's attention mechanism can provide meaningful explanation and demonstrate that the alternative attention distributions found via adversarial training methods do, in fact, perform poorly compared to standard attention mechanisms. On a theoretical level, they argue that, although attention weights do not give an *exclusive* "faithful" explanation, they do provide a meaningful *plausible* explanation.

This discussion is relevant to our study because it remains unclear whether or not attending to syntactic structure serves, in practice, as plausible explanation for model behavior, or whether or not it is even capable of serving as such. Indeed, the studies of Raganato et al. (2018) and Clark et al. (2019) relate a convincing but incomplete picture — tree decoding accuracy just marginally exceeds baselines and various relations tend to be tracked across varying heads and layers. Thus, our fine-tuning experiments (detailed in the following section) serve to enable an "easy" setting wherein we explicitly inform our models of the same structure that we are trying to extract. We posit that, if, after fine-tuning, syntactic structures were still *not* decodable from the attention weights, one could safely conclude that these structures are being stored via a non-transparent mechanism that may not even involve attention weights. Such an insight would allow us to conclude that attention weights cannot provide even a plausible explanation for models relying on syntax.

3 Experimental Design

To examine the extent to which we can decode dependency trees from attention patterns, we run a tree decoding algorithm over mBERT's attention heads — before and after fine-tuning via a parsing objective. We surmise that doing so will enable us to determine if attention can be interpreted as a reliable mechanism for capturing linguistic structure.

3.1 Model

We employ mBERT[1] in our experiments, which has been shown to perform well across a variety of NLP tasks (Hu et al., 2020; Kondratyuk and Straka, 2019a) and capture aspects of syntactic structure cross-lingually (Pires et al., 2019; Chi et al., 2020). mBERT features 12 layers with 768 hidden units and 12 attention heads, with a joint WordPiece sub-word vocabulary across languages. The model was trained on the concatenation of WikiDumps for the top 104 languages with the largest Wikipedias,where principled sampling was employed to enforce a balance between high- and

---

[1] `https://github.com/google-research/bert`

low-resource languages.

3.2 Decoding Algorithm

For decoding dependency trees, we follow Raganato et al. (2018) in applying the Chu-Liu-Edmonds maximum spanning tree algorithm (Chu, 1965) to every layer/head combination available in mBERT ($12 \times 12 = 144$ in total). In order for the matrices to correspond to gold treebank tokenization, we remove the cells corresponding to the BERT delimiter tokens (`[CLS]` and `[SEP]`). In addition to this, we sum the columns and average the rows corresponding to the constituent subwords of gold tokens, respectively (Clark et al., 2019). Lastly, since attention patterns across heads may differ in whether they represent heads attending to their dependents or vice versa, we take our input to be the element-wise product of a given attention matrix and its transpose ($A \circ A^{\top}$). We liken this to the joint probability of a head attending to its dependent and a dependent attending to its head, similarly to Limisiewicz et al. (2020). Per this point, we also follow Htut et al. (2019) in evaluating the decoded trees via Undirected Unlabeled Attachment Score (UUAS) — the percentage of undirected edges recovered correctly. Since we discount directionality, this is effectively a less strict measure than UAS, but one that has a long tradition in unsupervised dependency parsing since Klein and Manning (2004).

3.3 Data

For our data, we employ the Parallel Universal Dependencies (PUD) treebanks, as collected in UD v2.4 (Nivre et al., 2019). PUD was first released as part of the CONLL 2017 shared task (Zeman et al., 2018), containing 1000 parallel sentences, which were (professionally) translated from English, German, French, Italian, and Spanish to 14 other languages. The sentences are taken from two domains, **news** and **wikipedia**, the latter implying some overlap with mBERT's training data (though we did not investigate this). We include all PUD treebanks except Thai.[2]

3.4 Fine-Tuning Details

In addition to exploring pretrained mBERT's attention weights, we are also interested in how attention might be guided by a training objective that learns the exact tree structure we aim to decode. To this end, we employ the graph-based decoding algorithm of the biaffine parser introduced by Dozat and Manning (2016). We replace the standard BiLSTM encoder for this parser with the entire mBERT network, which we fine-tune with the parsing loss. The full parser decoder consists of four dense layers, two for head/child representations for dependency arcs (dim. 500) and two for head/child representations for dependency labels (dim. 100). These are transformed into the label space via a bilinear transform.

After training the parser, we can decode the fine-tuned mBERT parameters in the same fashion as described in Section 3.2. We surmise that, if attention heads are capable of tracking hierarchical relations between words in any capacity, it is precisely in this setting that this ability would be attested. In addition to this, we are interested in what individual *components* of the mBERT network are capable of steering attention patterns towards syntactic structure. We believe that addressing this question will help us not only in interpreting decisions made by BERT-based neural parsers, but also in aiding us developing syntax-aware models in general (Strubell et al., 2018; Swayamdipta et al., 2018). As such — beyond fine-tuning all parameters of the mBERT network (our basic setting) — we perform a series of ablation experiments wherein we update only one set of parameters per training cycle, e.g. the Query weights W_i^Q, and leave everything else frozen. This gives us a set of 6 models, which are described below. For each model, all non-BERT parser components are always left unfrozen.

- KEY: only the K components of the transformer are unfrozen; these are the representations of tokens that are paying attention *to* other tokens.
- QUERY: only the Q components are unfrozen; these, conversely, are the representations of tokens being paid attention to.
- KQ: both keys and queries are unfrozen.
- VALUE: semantic value vectors per token (V) are unfrozen; they are composed after being weighted with attention scores obtained from the K/Q matrices.
- DENSE: the dense feed-forward networks in the attention mechanism; all three per layer are unfrozen.
- NONE: The basic setting with nothing frozen; all parameters are updated with the parsing

---

[2] Thai is the only treebank that does not have a non-PUD treebank available in UD, which we need for our fine-tuning experiments.

	AR	CS	DE	EN	ES	FI	FR	HI	ID	IT	JA	KO	PL	PT	RU	SV	TR	ZH
BASELINE	50	40	36	36	40	42	40	46	47	40	43	55	45	41	42	39	52	41
PRE	53 7-6	53 10-8	49 10-8	47 10-8	50 9-5	48 10-8	41 2-3	48 2-3	50 9-5	41 6-4	**45** 2-3	64 9-2	52 10-8	50 9-5	51 10-8	51 10-8	55 3-8	42 2-3
NONE	**76** 11-10	**78** 11-10	**76** 11-10	**71** 10-11	**77** 10-11	**66** 10-11	**45** 11-10	**72** 11-10	**75** 11-10	**58** 11-10	42 11-10	64 11-10	**75** 11-10	**76** 11-10	**75** 10-8	**74** 10-8	55 3-8	38 2-3
KEY	62 10-8	64 10-8	58 11-12	53 10-8	59 11-12	56 10-8	41 7-12	54 10-8	59 10-8	47 9-2	44 2-3	62 10-8	64 10-8	58 11-12	61 10-8	59 12-10	55 3-12	41 2-3
QUERY	69 11-4	74 10-8	70 11-4	66 11-4	73 11-4	63 10-8	42 11-4	62 11-4	67 11-4	54 11-4	**45** 2-3	65 10-8	72 11-4	70 11-4	70 10-8	68 11-4	56 10-8	42 2-3
KQ	71 11-4	76 11-4	70 11-4	65 11-4	74 11-4	62 11-4	43 10-11	64 11-4	69 11-4	55 11-4	44 2-3	64 11-4	73 11-4	73 11-4	69 11-4	69 11-4	55 11-4	41 2-3
VALUE	75 12-5	72 12-5	72 12-5	64 12-5	76 12-5	59 12-5	**45** 12-5	63 12-5	73 12-5	55 12-5	**45** 2-3	**66** 10-8	73 12-5	74 12-5	69 12-5	65 12-5	**57** 12-5	**42** 3-8
DENSE	68 11-10	71 11-10	65 11-10	60 10-8	67 12-10	61 11-10	42 10-8	65 11-10	66 11-10	49 9-5	44 3-12	64 11-10	70 11-10	64 12-5	67 11-10	64 11-10	55 11-10	40 3-12

Table 1: Adjacent-branching baseline and maximum UUAS decoding accuracy per PUD treebank, expressed as best score and best layer/head combination for UUAS decoding. PRE refers to basic mBERT model before fine-tuning, while all cells below correspond different fine-tuned models described in Section 3.4. Best score indicated in **bold**.

loss.

We fine-tune each of these models on a concatenation of all PUD treebanks for 20 epochs, which effectively makes our model multilingual. We do so in order to 1) control for domain and annotation confounds, since all PUD sentences are parallel and are natively annotated (unlike converted UD treebanks, for instance); 2) increase the number of training samples for fine-tuning, as each PUD treebank features only 1000 sentences; and 3) induce a better parser through multilinguality, as in Kondratyuk and Straka (2019b). Furthermore, in order to gauge the overall performance of our parser across all ablated settings, we evaluate on the test set of the largest non-PUD treebank available for each language, since PUD only features test partitions. When training, we employ a combined dense/sparse Adam optimiser, at a learning rate of $3 * 10^{-5}$. We rescale gradients to have a maximum norm of 5.

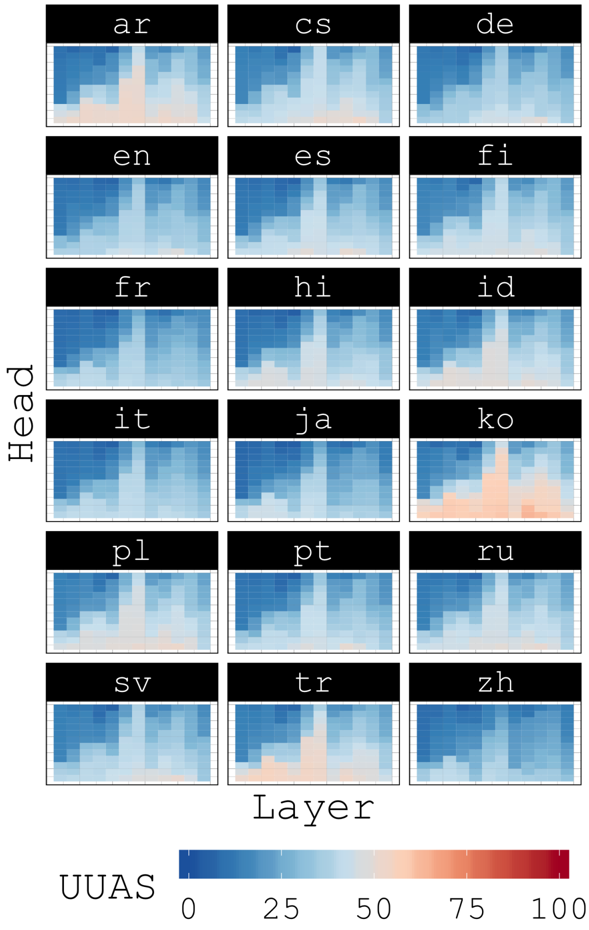
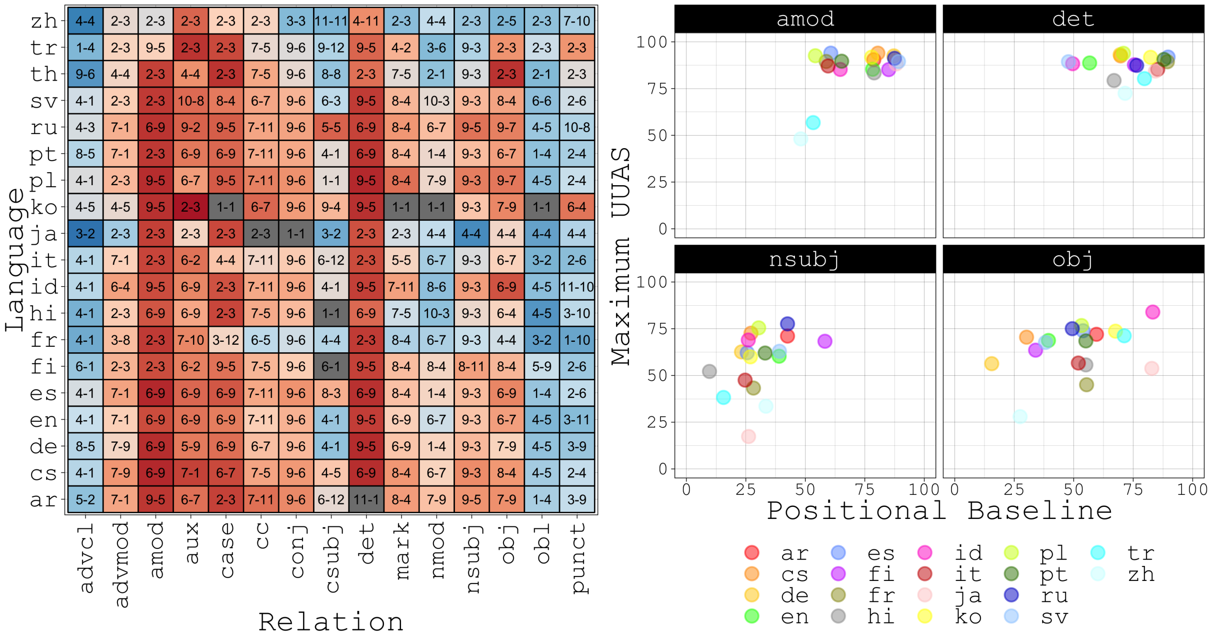


Figure 1: UUAS of MST decoding per layer and head, across languages. Heads (y-axis) are sorted by accuracy for easier visualization.

4 Decoding mBERT Attention

The second row of Table 1 (PRE) depicts the UUAS after running our decoding algorithm over mBERT attention matrices, per language. We see a familiar pattern to that in Clark et al. (2019) among others — namely that attention patterns extracted directly from mBERT appear to be incapable of decoding dependency trees beyond a threshold of 50–60% UUAS accuracy. However, we also note that, in all languages, the attention-decoding algorithm outperforms a BASELINE (row 1) that draws an (undirected) edge between any two adjacent words in linear order, which implies that some non-linear structures are captured with regularity. Indeed, head 8 in layer 10 appears to be particularly strong in this regard, returning the highest UUAS for 7 languages. Interestingly, the accuracy patterns



Language	advcl	advmod	amod	aux	case	cc	conj	csubj	det	mark	nmod	nsubj	obj	obl	punct
zh	4-4	2-3	2-3	2-3	2-3	2-3	3-3	11-11	4-11	2-3	4-4	2-3	2-5	2-3	7-10
tr	1-4	2-3	9-5	2-3	2-3	7-5	9-6	9-12	9-5	4-2	3-6	9-3	2-3	2-3	2-3
th	9-6	4-4	2-3	4-4	2-3	7-5	9-6	8-8	2-3	7-5	2-1	9-3	2-3	2-1	2-3
sv	4-1	2-3	2-3	10-8	8-4	6-7	9-6	6-3	9-5	8-4	10-3	9-3	8-4	6-6	2-6
ru	4-3	7-1	6-9	9-2	9-5	7-11	9-6	5-5	6-9	8-4	6-7	9-5	9-7	4-5	10-8
pt	8-5	7-1	2-3	6-9	6-9	7-11	9-6	4-1	6-9	8-4	1-4	9-3	6-7	1-4	2-4
pl	4-1	2-3	9-5	6-7	9-5	7-11	9-6	1-1	9-5	8-4	7-9	9-3	9-7	4-5	2-4
ko	4-5	4-5	9-5	2-3	1-1	6-7	9-6	9-4	9-5	1-1	1-1	9-3	7-9	1-1	6-4
ja	3-2	2-3	2-3	2-3	2-3	2-3	1-1	3-2	2-3	2-3	4-4	4-4	4-4	4-4	4-4
it	4-1	7-1	2-3	6-2	4-4	7-11	9-6	6-12	2-3	5-5	6-7	9-3	6-7	3-2	2-6
id	4-1	6-4	9-5	6-9	2-3	7-11	9-6	4-1	9-5	7-11	8-6	9-3	6-9	4-5	11-10
hi	4-1	2-3	6-9	6-9	2-3	7-5	9-6	1-1	6-9	7-5	10-3	9-3	6-4	4-5	3-10
fr	4-1	3-8	2-3	7-10	3-12	6-5	9-6	4-4	2-3	8-4	6-7	9-3	4-4	3-2	1-10
fi	6-1	2-3	2-3	6-2	9-5	7-5	9-6	6-1	9-5	8-4	8-4	8-11	8-4	5-9	2-6
es	4-1	7-1	6-9	6-9	6-9	7-11	9-6	8-3	6-9	8-4	1-4	9-3	6-9	1-4	2-6
en	4-1	7-1	6-9	6-9	6-9	7-11	9-6	4-1	9-5	6-9	6-7	9-3	6-7	4-5	3-11
de	8-5	7-9	6-9	5-9	6-9	6-7	9-6	4-1	9-5	6-9	1-4	9-3	7-9	4-5	3-9
cs	4-1	7-9	6-9	7-1	6-7	7-5	9-6	4-5	6-9	8-4	6-7	9-3	8-4	4-5	2-4
ar	5-2	7-1	9-5	6-7	2-3	7-11	9-6	6-12	11-1	8-4	7-9	9-5	7-9	1-4	3-9

Figure 2: Left: UUAS per relation across languages (best layer/head combination indicated in cell). Right: Best UUAS as a function of best positional baseline (derived from the treebank), selected relations.

across layers depicted in Figure 1 tend to follow an identical trend for all languages, with nearly all heads in layer 7 returning high within-language accuracies.

It appears that attention for some languages (Arabic, Czech, Korean, Turkish) is comparatively easier to decode than others (French, Italian, Japanese, Chinese). A possible explanation for this result is that dependency relations between content words, which are favored by the UD annotation, are more likely to be adjacent in the morphologically rich languages of the first group (without intervening function words). This assumption seems to be corroborated by the high baseline scores for Arabic, Korean and Turkish (but not Czech). Conversely, the low baselines scores and the likewise low decoding accuracies for the latter four languages are difficult to characterize. Indeed, we could not identify what factors — typological, annotation, tokenization or otherwise — would set French and Italian apart from the remaining languages in terms of score. However, we hypothesize that the tokenization and our treatment of subword tokens plays a part in attempting to decode attention from Chinese and Japanese representations. Per the mBERT documentation,[3] Chinese and Japanese Kanji character spans within the CJK Unicode range are character-tokenized. This lies in contrast with all other languages (Korean Hangul and Japanese Hiragana and Katakana included), which rely on whitespace and WordPiece (Wu et al., 2016). It is thus possible that the attention distributions for these two languages (at least where CJK characters are relevant) are devoted to composing words, rather than structural relations, which will distort the attention matrices that we compute to correspond with gold tokenization (e.g. by maxing rows and averaging columns).

Relation analysis We can disambiguate what sort of structures are captured with regularity by looking at the UUAS returned per dependency relation. Figure 2 (left) shows that adjectival modifiers (`amod`, mean UUAS = 85 ±12) and determiners (`det`, 88 ± 6) are among the easiest relations to decode across languages. Indeed, words that are connected by these relations are often adjacent to each other and may be simple to decode if a head is primarily concerned with tracking linear order. To verify the extent to which this might be happening, we plot the aforementioned decoding accuracy as a function of select relations' positional baselines in Figure 2 (right). The positional baselines, in this case, are calculated by picking the most frequent offset at which a dependent occurs with respect to its head, e.g., -1 for `det` in English, meaning one position to the left of the head. Interestingly, while we observe significant variation across the positional baselines for `amod` and `det`, the decoding

---

[3] `https://github.com/google-research/bert/blob/master/multilingual.md`

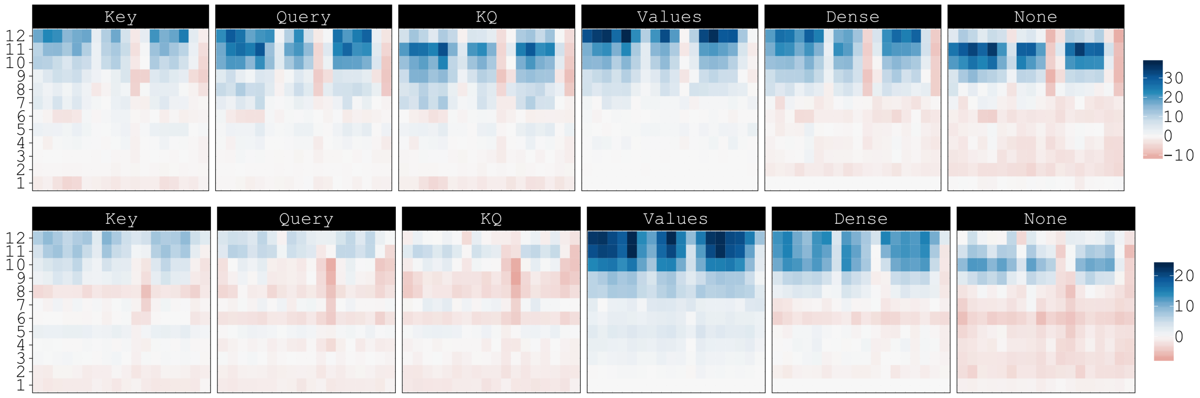


Figure 3: (Top) best scores across all heads, per language; (bottom) mean scores across all heads, per language. The languages (hidden from the X-axis for brevity) are, in order, *ar, cs, de, en, es, fi, fr, hi, id, it, ja, ko, pl, pt, ru, sv, tr, zh*

accuracy remains quite high.

In slight contrast to this, the core subject (`nsubj`, 58 ± 16 SD) and object (`obj`, 64 ± 13) relations prove to be more difficult to decode. Unlike the aforementioned relations, `nsubj` and `obj` are much more sensitive to the word order properties of the language at hand. For example, while a language like English, with Subject-Verb-Object (SVO) order, might have the subject frequently appear to the left of the verb, an SOV language like Hindi might have it several positions further away, with an object and its potential modifiers intervening. Indeed, the best positional baseline for English `nsubj` is 39 UUAS, while it is only 10 for Hindi. Despite this variation, the relation seems to be tracked with some regularity by the same head (layer 3, head 9), returning 60 UUAS for English and 52 for Hindi. The same can largely be said for `obj`, where the positional baselines return 51 ± 18. In this latter case, however, the heads tend to be much differently distributed across languages. Finally, he results for the `obj` relation provides some support for our earlier explanation concerning morphologically rich languages, as Arabic, Czech, Korean and Turkish all have among the highest accuracies (as well as positional baselines).

5 Fine-Tuning Experiments

Next, we investigate the effect fine-tuning has on UUAS decoding. Row 3 in Table 1 (NONE) indicates that fine-tuning does result in large improvements to UUAS decoding across most languages, often by margins as high as $\sim 30\%$. This shows that with an explicit parsing objective, attention heads are capable of serving as explanatory mechanisms for syntax; syntactic structure can be made to be transparently stored in the heads, in a manner that does not require additional probe fitting or parameterized transformation to extract.

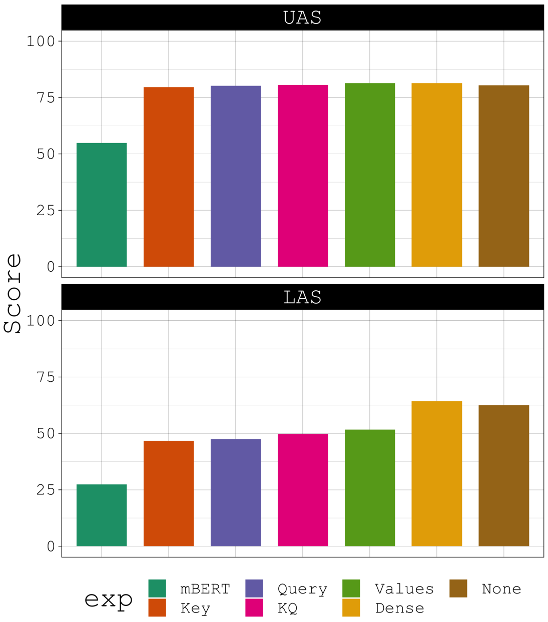


Figure 4: Mean UAS and LAS when evaluating different models on language-specific treebanks (Korean excluded due to annotation differences). MBERT refers to models where the entire mBERT network is frozen as input to the parser.

Given that we do manage to decode reasonable syntactic trees, we can then refine our question — what components are capable of learning these trees? One obvious candidate is the key/query component pair, given that attention weights are a scaled softmax of a composition of the two. Figure 3 (top) shows the difference between pretrained

UUAS and fine-tuned UUAS per layer, across models and languages. Interestingly, the best parsing accuracies do not appear to vary much depending on what component is frozen. We do see a clear trend, however, in that decoding the attention patterns of the fine-tuned model typically yields better UUAS than the pretrained model, particularly in the highest layers. Indeed, the lowest layer at which fine-tuning appears to improve decoding is layer 7. This implies that, regardless of which component remains frozen, the parameters facing any sort of significant and positive update tend to be those appearing towards the higher-end of the network, closer to the output.

For the frozen components, the best improvements in UUAS are seen at the final layer in VALUE, which is also the only model that shows consistent improvement, as well as the highest average improvement in mean scores[4] for the last few layers. Perhaps most interestingly, the mean UUAS (Figure 3 (bottom)) for our "attentive" components – keys, queries, and their combination – does not appear to have improved by much after fine-tuning. In contrast, the maximum does show considerable improvement; this seems to imply that although all components appear to be more or less equally capable of learning decodable heads, the attentive components, when fine-tuned, appear to sharpen fewer heads.

Note that the only difference between keys and queries in an attention mechanism is that keys are transposed to index attention from/to appropriately. Surprisingly, KEY and QUERY appear to act somewhat differently, with QUERY being almost uniformly better than KEY with the best heads, whilst KEY is slightly better with averages, implying distinctions in how both store information. Furthermore, allowing both keys and queries seems to result in an interesting contradiction – the ultimate layer, which has reasonable maximums and averages for both KEY and QUERY, now seems to show a UUAS drop almost uniformly. This is also true for the completely unfrozen encoder.

Supervised Parsing In addition to decoding trees from attention matrices, we also measure supervised UAS/LAS on a held-out test set.[5] Based on Figure 4, it is apparent that all settings result in generally the same UAS. This is somewhat expected; Lauscher et al. (2020) see better results on parsing with the entire encoder frozen, implying that the task is easy enough for a biaffine parser to learn, given frozen mBERT representations.[6] The LAS distinction is, however, rather interesting: there is a marked difference between how important the dense layers are, as opposed to the attentive components. This is likely not reflected in our UUAS probe as, strictly speaking, labelling arcs is not equivalent to searching for structure in sentences, but more akin to classifying pre-identified structures. We also note that DENSE appears to be better than NONE on average, implying that non-dense components might actually be hurting labelling capacity.

In brief, consolidating the two sets of results above, we can draw three interesting conclusions about the components:

1. **Value** vectors are best aligned with syntactic dependencies; this is reflected both in the best head at the upper layers, and the average score across all heads.
2. **Dense** layers appear to have moderate informative capacity, but appear to have the best learning capacity for the task of arc labelling.
3. Perhaps most surprisingly, **Key** and **Query** vectors do not appear to make any outstanding contributions, save for sharpening a smaller subset of heads.

Our last result is especially surprising for UUAS decoding. Keys and queries, fundamentally, combine to form the attention weight matrix, which is precisely what we use to decode trees. One would expect that allowing these components to learn from labelled syntax would result in the best improvements to decoding, but all three have surprisingly negligible mean improvements. This indicates that we need to further improve our understanding of how attentive structure and weighting really works.

Cross-linguistic observations We notice no clear cross-linguistic trends here across different component sets; however, certain languages do stand out as being particularly hard to decode from the fine-tuned parser. These include Japanese, Korean, Chinese, French and Turkish. For the first three, we hypothesise that tokenization clashes with

---

[4] The inner average is over all heads; the outer is over all languages.

[5] Note that the test set in our scenario is from the actual, non-parallel language treebank; as such, we left Korean out of this comparison due to annotation differences.

[6] Due to training on concatenated PUD sets, however, our results are not directly comparable/

mBERT's internal representations may play a role. Indeed, as we hypothesized in Section 3.2, it could be the case that the composition of CJK characters into gold tokens for Chinese and Japanese may degrade the representations (and their corresponding attention) therein. Furthermore, for Japanese and Korean specifically, it has been observed that tokenization strategies employed by different treebanks could drastically influence the conclusions one may draw about their inherent hierarchical structure (Kulmizev et al., 2020). Turkish and French are admittedly more difficult to diagnose. Note, however, that we fine-tuned our model on a concatenation of all PUD treebanks. As such, any deviation from PUD's annotation norms is therefore likely to be heavily penalised, by virtue of signal from other languages drowning out these differences.

6 Conclusion

In this study, we revisited the prospect of decoding dependency trees from the self-attention patterns of Transformer-based language models. We elected to extend our experiments to 18 languages in order to gain better insight about how tree decoding accuracy might be affected in the face of (modest) typological diversity. Surprisingly, across all languages, we were able to decode dependency trees from attention patterns more accurately than an adjacent-linking baseline, implying that some structure was indeed being tracked by the mechanism. In looking at specific relation types, we corroborated previous studies in showing that particular layer-head combinations tracked the same relation with regularity across languages, despite typological differences concerning word order, etc.

In investigating the extent to which attention can be guided to properly capture structural relations between input words, we fine-tuned mBERT as input to a dependency parser. This, we found, yielded large improvements over the pretrained attention patterns in terms of decoding accuracy, demonstrating that the attention mechanism was learning to represent the structural objective of the parser. In addition to fine-tuning the entire mBERT network, we conducted a series of experiments, wherein we updated only select components of model and left the remainder frozen. Most surprisingly, we observed that the Transformer parameters designed for composing the attention matrix, K and Q, were only modestly capable of guiding the attention towards resembling the dependency structure. In contrast, it was the Value (V) parameters, which are used for computing a weighted sum over the KQ-produced attention, that yielded the most faithful representations of the linguistic structure via attention.

Though prior work (Kovaleva et al., 2019; Zhao and Bethard, 2020) seems to indicate that there is a lack of a substantial change in attention patterns after fine-tuning on syntax- and semantics-oriented classification tasks, the opposite effect has been observed with fine-tuning on negation scope resolution, where a more explanatory attention mechanism can be induced (Htut et al., 2019). Our results are similar to the latter, and we demonstrate that given explicit syntactic annotation, attention weights do end up storing more transparently decodable structure. It is, however, still unclear which sets of transformer parameters are best suited for learning this information and storing it in the form of attention.

Acknowledgements

Our experiments were run on resources provided by UNINETT Sigma2 - the National Infrastructure for High Performance Computing and Data Storage in Norway, under the NeIC-NLPL umbrella. Mostafa and Anders were funded by a Google Focused Research Award. We would like to thank Daniel Dakota and Ali Basirat for some fruitful discussions and the anonymous reviewers for their excellent feedback.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R Bowman. 2019. Do attention heads in bert track syntactic dependencies? *arXiv preprint arXiv:1911.12246*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization. *arXiv:2003.11080 [cs]*. ArXiv: 2003.11080.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 479–486.

Dan Kondratyuk and Milan Straka. 2019a. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795.

Dan Kondratyuk and Milan Straka. 2019b. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the Dark Secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4364–4373, Hong Kong, China. Association for Computational Linguistics.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Anne Lauscher, Vinit Ravishankar, Ivan Vuli, and Goran Glava. 2020. From Zero to Hero: On the Limitations of Zero-Shot Cross-Lingual Transfer with Multilingual Transformers. *arXiv:2005.00633 [cs]*. ArXiv: 2005.00633.

Tomasz Limisiewicz, Rudolf Rosa, and David Mareček. 2020. Universal dependencies according to bert: both more specific and more general. *arXiv preprint arXiv:2004.14620*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 91–98.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 523–530.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Gabrielė Aleksandravičiūtė, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Agnė Bielinskienė, Rogier Blokland, Victoria Bobicev, Loïc Boizou, Emanuel Borges Völker, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane

Boyd, Kristina Brokaitė, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Bamba Dione, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Hanne Eckhoff, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Kazunori Fujita, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Johannes Heinecke, Felix Hennig, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Takumi Ikeda, Radu Ion, Elena Irimia, Ọlájídé Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Andre Kaasen, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Arne Köhn, Kamil Kopacewicz, Natalia Kotsyba, Jolanta Kovalevskaitė, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phng Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, Kyung-Tae Lim, Yuan Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Sarah McGuinness, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Tomohiko Morioka, Shinsuke Mori, Shigeki Moro, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lng Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Yoshihiro Nikaido, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Angelika Peljak-Łapińska, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Daria Petrova, Slav Petrov, Jussi Piitulainen, Tommi A Pirinen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Erika Rimkutė, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Alessio Salomoni, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Dage Särg, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Hiroyuki Shirasu, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Shingo Suzuki, Zsolt Szántó, Dima Taji, Yuta Takahashi, Fabio Tamburini, Takaaki Tanaka, Isabelle Tellier, Guillaume Thomas, Liisi Torga, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Abigail Walsh, Jing Xian Wang, Jonathan North Washington, Maximilan Wendt, Seyi Williams, Mats Wirén, Christian Wittern, Tsegay Woldemariam, Tak-sum Wong, Alina Wróblewska, Mary Yako, Naoki Yamazaki, Chunxiao Yan, Koichi Yasuoka, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2019. Universal dependencies 2.4. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Dan Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Dan Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC)*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001.

Alessandro Raganato, Jörg Tiedemann, et al. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. The Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Rudolf Rosa and David Mareček. 2019. Inducing syntactic trees from bert representations. *arXiv preprint arXiv:1906.11511*.

Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? *arXiv preprint arXiv:1906.03731*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3772–3782, Brussels, Belgium. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017b. Attention Is All You Need. *arXiv:1706.03762 [cs]*. ArXiv: 1706.03762.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Daniel Zeman, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 1–21.

Yiyun Zhao and Steven Bethard. 2020. How does BERT's attention change when you fine-tune? An analysis methodology and a case study in negation scope. page 19.

A Positional Scores Per Offset

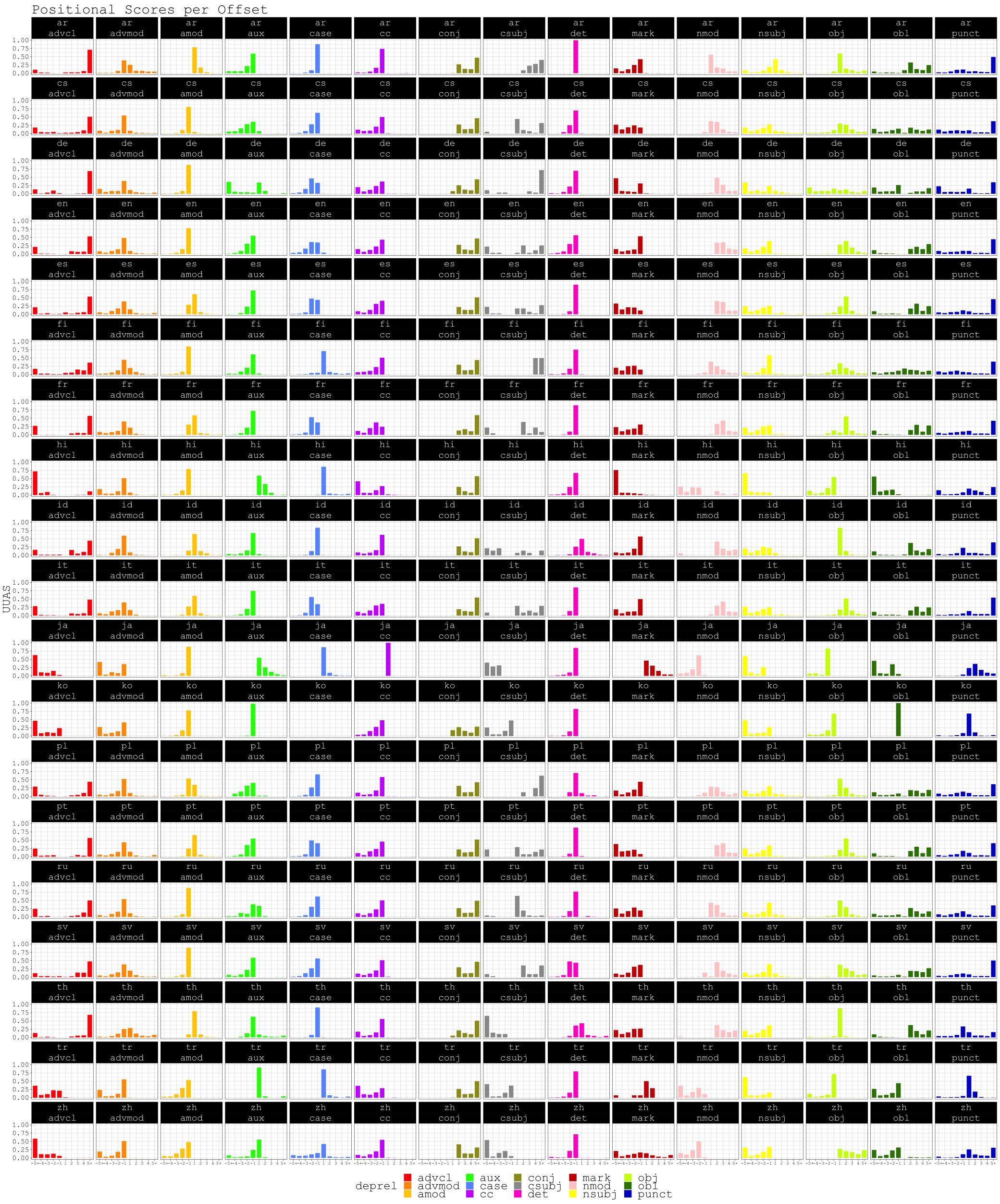


Figure 5: Positional scores across relations for all languages.

B Decoding UUAS Across Relations

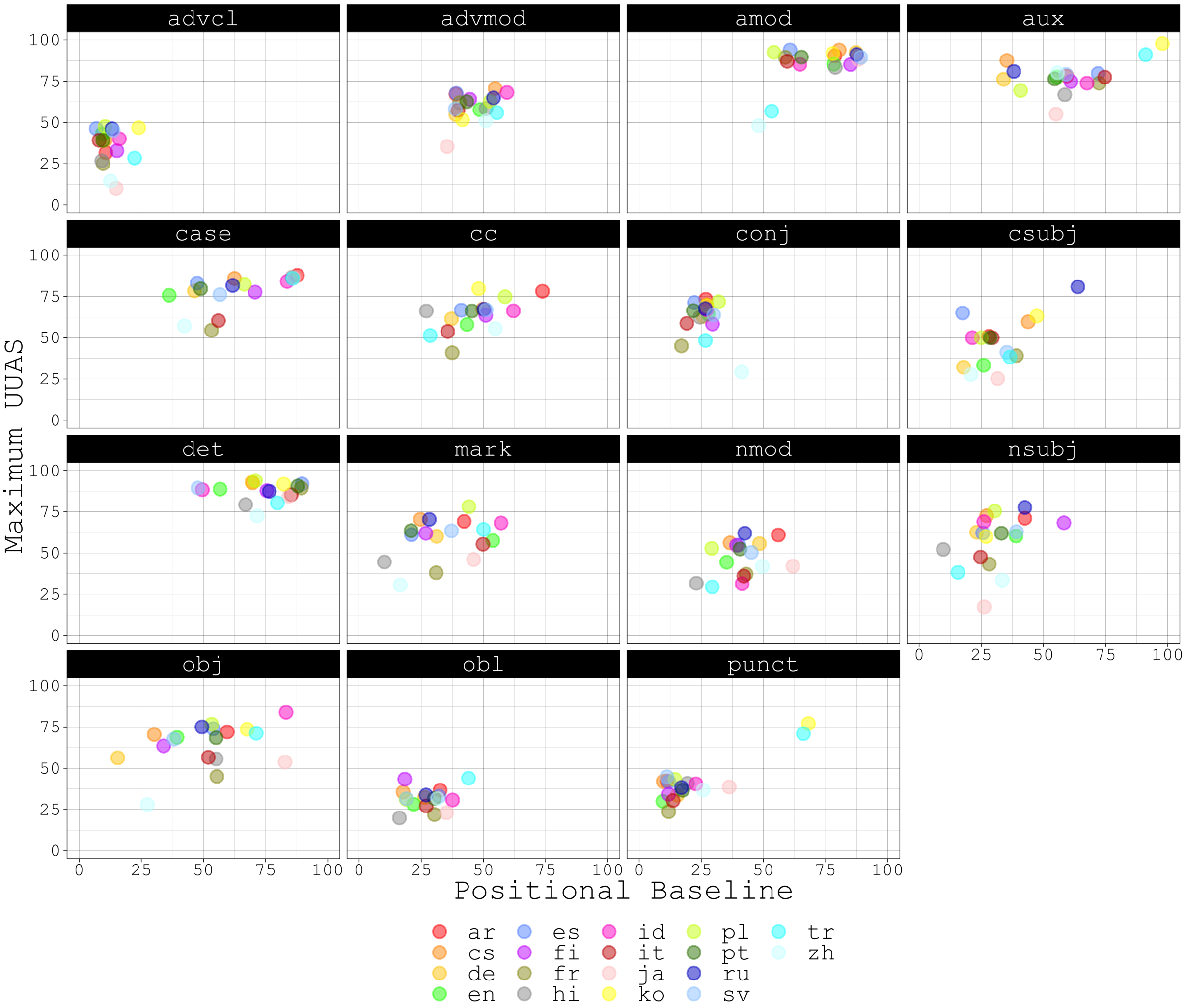


Figure 6: Decoding UUAS as a function of best positional baselines.

C Full Parsing Scores

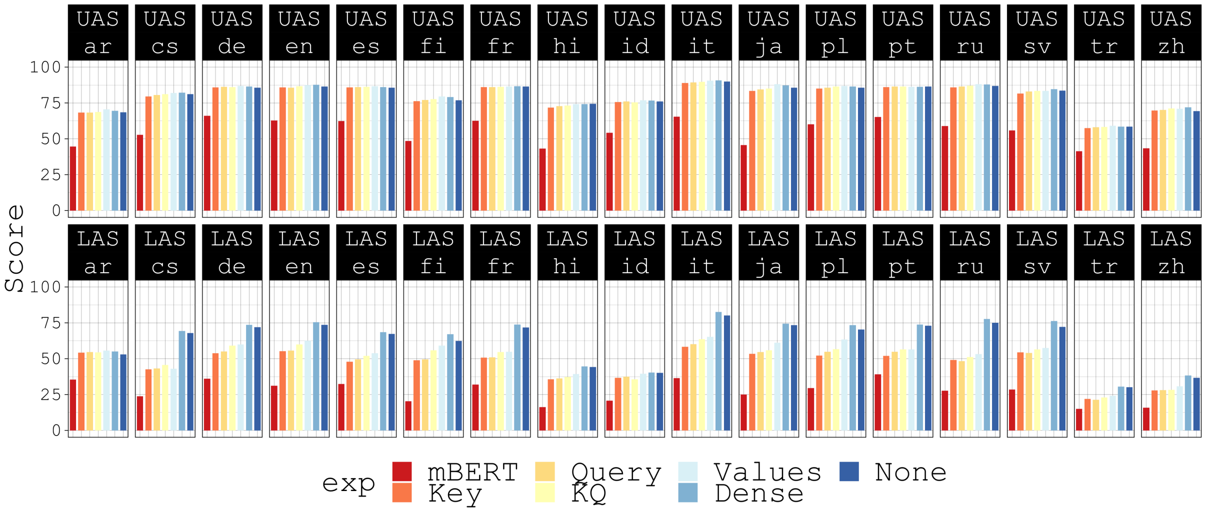


Figure 7: Parsing scores across components and languages.

Extractive Summarization Considering Discourse and Coreference Relations based on Heterogeneous Graph

Yin Jou Huang
Kyoto University
huang@nlp.ist.i.kyoto-u.ac.jp

Sadao Kurohashi
Kyoto University
kuro@i.kyoto-u.ac.jp

Abstract

Modeling the relations between text spans in a document is a crucial yet challenging problem for extractive summarization. Various kinds of relations exist among text spans of different granularity, such as discourse relations between elementary discourse units and coreference relations between phrase mentions. In this paper, we propose a heterogeneous graph based model for extractive summarization that incorporates both discourse and coreference relations. The heterogeneous graph contains three types of nodes, each corresponds to text spans of different granularity. Experimental results on a benchmark summarization dataset verify the effectiveness of our proposed method.

1 Introduction

Automatic summarization aims to condense the information of the input document into a shorter summary. The task has two main paradigms: extractive summarization and abstractive summarization. Generating summary sentences from scratch, abstractive summarizers can generate concise and flexible summaries. However, they also suffer from the problem of not being able to reproduce factual details correctly (See et al., 2017). On the other hand, extractive summarization aims to select salient text spans (mostly sentences) from the input document. Compared to abstractive summarizers, extractive summarizers have the advantage of being efficient and factually reliable. In this paper, we will focus on extractive summarization.

For extractive summarization, it is crucial to model the relations between text spans throughout the document. Between text spans of different granularity, there exist many different kinds of relations (Figure 1). For example, coreference relations exist between mention phrases of the same entity, and discourse relations exist between Elementary Discourse Units (EDUs) within a document. Due to its complex nature, modeling the various relations among text spans of a document remains an open challenge.

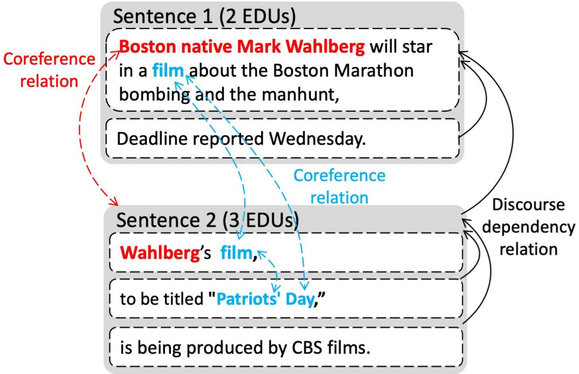


Figure 1: Relations among text spans of different granularity.

To capture inter-sentential relations, some recent works utilize recurrent neural networks (RNNs) or Transformer (Vaswani et al., 2017) based encoders on top of the acquired sentence representations (Cheng and Lapata, 2016; Nallapati et al., 2016; Liu and Lapata, 2019). However, empirical observations show that these sentence-level encoders do not bring much performance gain (Liu and Lapata, 2019). Graph structure is an intuitive way to model long-range dependencies among text spans throughout a document. Early works build connectivity graphs based on content similarity between sentences (Erkan and Radev, 2004; Mihalcea and Tarau, 2004). Some recent works incorporate discourse or coreference relations into the graph structure and utilize graph neural networks (GNNs) to obtain a high-level representation of text spans (Yasunaga et al., 2017; Xu and Durrett, 2019; Xu et al., 2020). Most of these works operate on homogeneous graphs with only one type of nodes, such as Approximate Discourse Graph (ADG) (Christensen et al., 2013) or Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) dependency graph. As illustrated in Figure 1, the various types of relations exist between text spans of different

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3046–3052
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

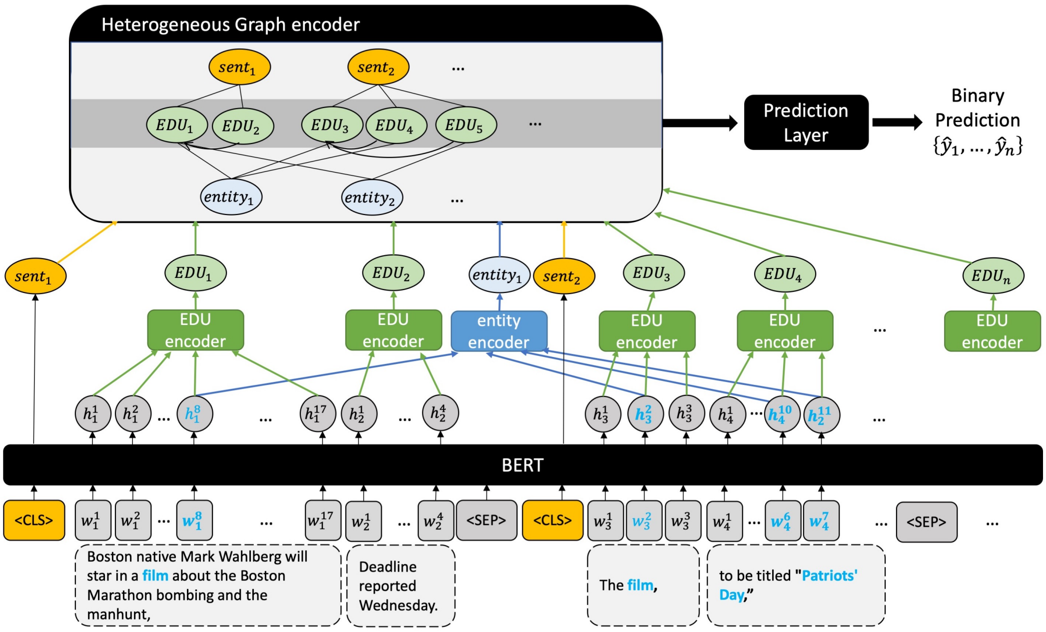


Figure 2: System overview.

granularity. Thus, homogeneous graphs may not be an ideal way to encode the various types of relations between text spans.

In this paper, we propose a novel heterogeneous graph based model for extractive summarization. Heterogeneous graphs are graphs that contain multiple node types and/or multiple edge types, which is in contrast to homogeneous graphs that have only one type of nodes and one type of edge. Heterogeneous graphs have been widely studied and applied to model data structures such as citation networks (Yu et al., 2012), recommendation systems (Dong et al., 2012), etc. In this work, we use heterogeneous graph to model the document structure with three types of nodes of different granularity: sentence nodes, EDU nodes, and entity nodes.

We also try to encode both discourse and coreference relations into the graph structure. We encode the discourse relations with the edges between EDU nodes. As for the coreference relations, edges between EDU nodes and entity nodes are introduced. Instead of extracting salient sentences like most existing extractive summarizers, our model extracts salient EDUs. To identify salient EDUs in a certain sentence, we add edges between sentence nodes and their constituent EDU nodes. To the best of our knowledge, we are the first to utilize heterogeneous graph to incorporate multiple types of relations simultaneously for extractive summarization.

2 Proposed Method

Given an input document D with n EDUs $\{d_1, d_2, ..., d_n\}$, we formulate extractive summarization as a sequence labeling problem. The model predicts a sequence of binary labels $Y = \{y_1, y_2, ..., y_n\}$, where $y_i = 1$ indicates that the i^{th} EDU should be included in the summary.

Figure 2 provides an overview of our proposed model. First, a BERT encoder is used to embed the input document D. With the EDU and entity encoders, we acquire the initial node representation of the heterogeneous graph. We then apply a heterogeneous graph encoder to obtain high-level node representations. Finally, we make predictions based on the EDU node representations.

2.1 Heterogeneous Graph Construction

We represent each input document D with a heterogeneous graph $G = \{V, E\}$, where V and E are the set of nodes and edges, respectively.

Given document D with m sentences $\{s_1, ..., s_m\}$, we first segment the sentences into n sub-sentential EDUs $\{d_1, ..., d_n\}$ and perform RST discourse parsing to identify the relations between the EDUs. In addition, we perform coreference resolution to identify the mentions and the coreference relations between them. The mentions in D are then clustered into k entities $\{e_1, ..., e_k\}$, with each entity e_i representing a cluster of mentions among which coreference relations holds.

The set of nodes $V = V_s \cup V_d \cup V_e$ consists of m

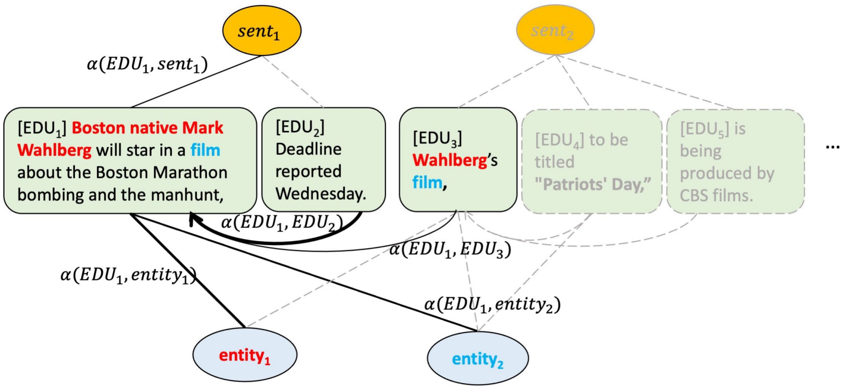


Figure 3: Graph attention mechanism.

sentence nodes V_s, n EDU nodes V_d, and k entity nodes V_e. There are three types of edges in E. First, we use edges between EDU nodes to represent the discourse structure of the document. Similar to Xu et al. (2020), we derived the discourse dependency links between EDU nodes based on the RST tree of the document. The discourse dependency links are directional, which capture the dependency relations going from satellite to nucleus EDUs. Second, we use edges between EDU nodes and entity nodes to embed the coreference relations. If EDU d_i contains a mention of entity e_j, then we add an undirected edge (d_i, e_j) to E. In this way, each entity indirectly connects all EDUs with mentions of the entity. Third, we also link each sentence node to its constituent EDU nodes undirectionally.

The proposed heterogeneous document graph enables us to simultaneously model various relations between different sizes of text spans: sentence, EDU, entity phrase efficiently.

2.2 Graph Node Initialization

Following the settings in Liu and Lapata (2019), we utilize pretrained BERT (Devlin et al., 2019) to encode the input document D. We insert the ⟨CLS⟩ and ⟨SEP⟩ special tokens to the beginning and the end of each sentence s_i, respectively. With the BERT output vectors, we acquire the initial representations of each node in V as follows:

Sentence Representations

For each sentence node s_i, we take the BERT output vector of the ⟨CLS⟩ token before s_i as the sentence node representation $sent_i$.

EDU Representations

We use a self-attention based EDU encoder to encode each EDU node d_i. Given an EDU d_i with tokens $\{w_i^j\}$, we obtain its node representation EDU_i by taking self-attention on the BERT output vectors $\{h_i^j\}$ of the tokens:

$$\alpha_{ij} = v_2 ReLU(W_1 h_i^j + b_1) \tag{1}$$

$$a_{ij} = softmax_j(\alpha_{i1}, \alpha_{i2}, ...) \tag{2}$$

$$EDU_i = \sum_j a_{ij} h_i^j \tag{3}$$

Entity Representations

The structure of the entity encoder is identical to the EDU encoder. For each entity e_i, we consider all mentions of it. By taking self-attention among the BERT output vectors which correspond to tokens of these mentions, we can acquire the entity representation $entity_i$.

2.3 Heterogeneous Graph Encoder

We initialize the representation of each node in G with the sentence representations ($sent_i$), EDU representations (EDU_i), and entity representations ($entity_i$) acquired in section 2.2.

We apply graph attention networks (GAT) (Veličković et al., 2018) to update the node representations in G. For each iteration, we update the representation h_i of node i with the representations of its neighbors $\{h_j\}$ based on the attention weights α_{ij}:

$$\alpha_{ij} = LeakyReLU(W_a[W_q h_i; W_k h_j]) \tag{4}$$

$$a_{ij} = softmax_j(\alpha_{i1}, \alpha_{i2}, ...) \tag{5}$$

$$h_i \leftarrow W_t(\sigma(\sum_j a_{ij} W_v h_j) + h_i) \tag{6}$$

An example of the graph attention mechanism is illustrated in Figure 3, where the subgraph around node EDU_1 is highlighted. EDU_1 has five neighbors: a sentence node ($sent_1$), two EDU nodes (EDU_2, EDU_3), and two entity nodes ($entity_1$, $entity_2$.) We first calculate the attention weights α across the five neighbors of EDU_1 using equation

4, and update the node representation of EDU_1 accordingly.

Although a single GAT network only considers the first-degree neighbors, we can obtain a higher-level representation for each node in G by updating the node representations for several iterations.

2.4 Prediction Layer

We feed the final representation of the EDU nodes (EDU_i) to the prediction layer with sigmoid activation to predict binary labels:

$$\hat{y}_i = \sigma(W_p EDU_i + b_p) \tag{7}$$

The training loss of the model is the binary cross-entropy loss against the oracle extraction labels.

3 Experiment

3.1 Dataset

We evaluated our proposed model on the benchmark CNN/DailyMail dataset (non-anonymized version) (Hermann et al., 2015). We used the standard dataset split, which contains 287,227 / 13,368 / 11,490 documents for training, validation, and test split, respectively.

We used the Stanford CoreNLP (Manning et al., 2014) to split sentences. Further, we used the RST discourse parser proposed by Ji and Eisenstein (2014) for both discourse segmentation and discourse parsing. For coreference resolution, we used the spanBERT-based (Joshi et al., 2020) version of the end-to-end coreference resolver proposed by Lee et al. (2017).

Since the CNN/DailyMail dataset only contains abstractive gold summaries, we have to construct oracle labels heuristically. We obtained the oracle labels on EDU-level with the heuristic algorithm based on ROUGE (Lin, 2004), similar to the one in Liu and Lapata (2019). For each document, we selected up to 5 EDUs.

3.2 Experimental Settings

We used the base model of Longformer (Beltagy et al., 2020) to encode the input document. The length of each document is truncated to 1024 BPEs. The hidden size of the EDU encoder and the entity encoder is 128. Based on the evaluation losses on the validation set, we set the number of iterations of the GAT layer to 3. Also, the number of attention heads is set to 8, with each head having a hidden size 64.

Model	R-1	R-2	R-L
LEAD-3	40.34	17.70	36.57
Oracle(sent)	52.59	31.24	48.87
Oracle(EDU)*	55.96	34.64	53.26
BanditSum (Dong et al., 2018)	41.50	18.70	37.60
NEUSUM (Zhou et al., 2018)	41.59	19.01	37.98
HIBERT (Zhang et al., 2019)	42.37	19.95	38.83
HSG (Wang et al., 2020)	42.95	19.76	39.23
BertSum (sent) (Liu and Lapata, 2019)	43.25	20.24	39.63
BertSum (EDU)*	42.73	20.03	40.16
DiscoBERT* (Xu et al., 2020)	**43.77**	**20.85**	40.67
Proposed*	43.61	20.81	**41.12**

Table 1: Results on the test set of CNN/DailyMail dataset. Models with * superscript are EDU-based extractive models.

During training, we used a batch size of 32. We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and followed the learning rate scheduling in Vaswani et al. (2017) with warm-up of 4000 steps. All models are trained for 50000 steps. We selected the top-3 checkpoints based on the evaluation losses on the validation set and report the average scores of them on the test set.

3.3 Results and Analysis

Table 1 shows the results on CNN/DailyMail dataset. The first part contains the **LEAD-3** baseline and **Oracle** upper bounds. The second part of the table includes other sentence based extractive models, and the third part includes other EDU based extractive models. In the last row of the table, we present the evaluation scores of our proposed model.

As Table 1 shows, our proposed model outperforms the **BertSum(EDU)** baseline by a significant margin (0.88/0.78/0.96 on F_1 of R-1/R-2/R-L). Our proposed model also outperforms the **BertSum(sent)** model and other sentence based extractive summarization baseline models. The proposed model is comparable to the state-of-the-art EDU-extraction model **DiscoBERT** in R-1 and R-2 metrics, and outperforms it in R-L metrics. **DiscoBERT** incorporates a strict RST-based rule during both oracle label construction and post-processing stages to ensure discourse consistency. Since the purpose of this paper is to propose a heterogeneous graph based method for modeling text span relations, we will leave the question of discourse consistency to future work.

Ablation Study

We conduct an ablation study on the components of our proposed model (Table 3). First, we remove the RST dependency edges between EDU nodes (**-discourse**). Next, we remove the corefer-

EDUs with coreferent entity	Rank$_{\text{coref}}$	Rank$_{\text{w/o coref}}$	Oracle Label
Mexican state oil company Pemex said 45 workers were injured ...	1	2	1
... two of them are in serious condition	2	1	0
Kim was accused of stabbing U.S. ambassador Mark Lippert ...	1	3	1
Before Lippert was supposed to give a speech, the attacker slashed him in the face a jaw	3	1	0
Kim stabbed Lippert with a 10-inch knife	2	2	0

Table 2: Qualitative studies on CNN/DailyMail dataset.

Model	R-1 (ΔR-1)	R-2 (ΔR-2)	R-L (ΔR-L)
Proposed	43.61	20.67	40.95
- discourse	43.42(**-0.19**)	20.54(**-0.13**)	40.77(**-0.18**)
- coref	43.51(-0.10)	20.61(-0.06)	40.85(-0.10)
- sent	43.57(-0.04)	20.66(-0.01)	40.90(-0.05)

Table 3: Ablation studies on CNN/DailyMail dataset.

ential edges between EDU nodes and entity nodes (**-coref**). The results of the ablation study show that discourse information plays an important role in our proposed model, while adding coreference information also gives a gain in performance. We also try to remove the edges between sentence nodes and their constituent EDU nodes (**-sent**). However, linking the sentence and EDU nodes does not seem to have a significant impact on model performance.

Qualitative Analysis

We also conduct a qualitative analysis of the proposed model. The effectiveness of discourse relations is more straight-forward and widely studied in previous research. Thus, we focus on the analysis of the role of coreference information in our proposed summarization model.

In the heterogeneous document graph, EDUs containing the same entity phrase are indirectly connected through the node of the given entity. By analyzing the output of the full proposed model and the model without coreference information (**-coref**), we found that the models rank the importance of coreferent EDUs differently. Table 2 indicates a common pattern of the improved cases by incorporating coreference information. The table shows examples of coreferent EDUs and the ranking of their likelihood scores to be included in the summary. Comparing the EDU ranking of the full model (**Rank$_{\text{coref}}$**) and the model without coreference information (**Rank$_{\text{w/o coref}}$**), we argue that the model with coreference information is better in discriminating the important EDUs among all EDUs sharing the same entity.

4 Related Work

Graph based Summarization

Graph based summarization models have been broadly explored. Early works build connectivity document graphs based on inter-sentential similarity (Erkan and Radev, 2004; Mihalcea and Tarau, 2004). With the promising results of graph neural networks (GNNs)(Kipf and Welling, 2017; Veličković et al., 2018), some recent works utilize GNN to incorporate external knowledge into the model. For instance, Yasunaga et al. (2017) utilizes a sentence-level ADG graph to model discourse and coreference relations. Some works convert the RST tree of the input document into dependency form in either sentence or EDU level (Xu and Durrett, 2019; Xu et al., 2020). Most of these models operate on homogeneous graphs with only one type of node. One of the major disadvantages of homogeneous graphs is that they can only embed one relation type in a single graph, since there is only one type of node and one type of edge.

Fewer summarization models operate on heterogeneous graphs with different types of nodes. Wei (2012) introduces a heterogeneous graph of sentence, word, and topic nodes, and Wang et al. (2020) also utilizes a heterogeneous graph of sentence and word nodes. However, neither of the above works incorporates external knowledge into the graph.

EDU based Extractive Summarization

Li et al. (2016) illustrates the potential of using EDU as the extraction unit for summarization. Xu et al. (2020) also introduces an end-to-end EDU based extractive summarization model. By using a heuristic based on RST dependency structure, they enhanced the grammaticality and discourse consistency of the extracted summary.

5 Conclusion

In this paper, we proposed a novel heterogeneous graph based model for extractive summarization. By introducing nodes of different granularity, the heterogeneous graph has the capacity to embed various types of relations between text spans. Experiments on CNN/DailyMail benchmark dataset illustrated the effectiveness of our proposed method.

References

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

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

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

Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao. 2012. Link prediction and recommendation across heterogeneous social networks. In *2012 IEEE 12th International Conference on Data Mining*, pages 181–190.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. BanditSum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748, Brussels, Belgium. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)*, 22:457–479.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

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

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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

Junyi Jessy Li, Kapil Thadani, and Amanda Stent. 2016. The role of discourse units in near-extractive summarization. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 137–147, Los Angeles. Association for Computational Linguistics.

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

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

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

R. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219, Online. Association for Computational Linguistics.

Y. Wei. 2012. Document summarization method based on heterogeneous graph. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pages 1285–1289.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462, Vancouver, Canada. Association for Computational Linguistics.

Xiao Yu, Quanquan Gu, Mianwei Zhou, and Jiawei Han. 2012. *Citation Prediction in Heterogeneous Bibliographic Networks*, pages 1119–1130.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663, Melbourne, Australia. Association for Computational Linguistics.

CDA: a Cost Efficient Content-based Multilingual Web Document Aligner

Thuy Vu
Amazon Alexa AI
Manhattan Beach, CA, USA
`thuyvu@amazon.com`

Alessandro Moschitti
Amazon Alexa AI
Manhattan Beach, CA, USA
`amosch@amazon.com`

Abstract

We introduce a Content-based Document Alignment approach (CDA), an efficient method to align multilingual web documents based on content in creating parallel training data for machine translation (MT) systems operating at the industrial level. CDA works in two steps: (i) projecting documents of a web domain to a shared multilingual space; then (ii) aligning them based on the similarity of their representations in such space. We leverage lexical translation models to build vector representations using TF×IDF. CDA achieves performance comparable with state-of-the-art systems in the WMT-16 Bilingual Document Alignment Shared Task benchmark while operating in multilingual space. Besides, we created two web-scale datasets to examine the robustness of CDA in an industrial setting involving up to 28 languages and millions of documents. The experiments show that CDA is robust, cost-effective, and is significantly superior in (i) processing large and noisy web data and (ii) scaling to new and low-resourced languages.

1 Introduction

Online machine translation (MT) services require industrial-scale training data, i.e., significantly large and high-quality parallel sentences, to build accurate models. Exploiting the web for multilingual content has become a usual strategy in collecting large-scale parallel sentences for MT (Uszkoreit et al., 2010; Smith et al., 2013; Buck and Koehn, 2016a). Structural Translation Recognition for Acquiring Natural Data (STRAND) (Resnik and Smith, 2003) is a standard pipeline to extract parallel data from the web, consisting in three steps: (i) bilingual document alignment for an input set of documents, (ii) sentence alignment for each aligned document pair, and (iii) sentence filtering for non-translation or boilerplate cleaning. The first step of identifying bilingual documents is technically challenging and made more complicated by the presence of large and noisy documents from web data.

In the WMT-16 Bilingual Document Alignment Shared Task (WMT16-BDAST), two standard approaches for identifying parallel pages[1] were studied: 1) URL matching heuristic (Smith et al., 2013) as a baseline and 2) content similarity as a solution to maximize the performance in identifying parallel documents. The benchmark on English-French document alignment task shows that the best top-1 recall (R@1) for each approach are 59.8% and 89.1%, respectively, as evaluated on the test set (Buck and Koehn, 2016a,b). The results, albeit conducted within English-French setting, indicate that leveraging document content can lead to a significant increase, up to 30 percent points in recall, and contributes ∼50% novel bilingual document pairs.

The URL matching heuristic approach, named URL, identifies parallel pages using language identifiers, typically from ISO 639, annotated in the addresses. Pages, or web-documents, in different languages from a domain are aligned if their URLs are matchable after their language-identifier being removed (Smith et al., 2013). The strategy can identify a candidate at a decent cost by comparing two URLs without significant preprocessing needed. For example, the following URLs are a match: `xyz.ca/index.htm` and `xyz.ca/fr/index.htm` after removing "`fr/`" from the second URL. On the contrary, cost is the major issue when comparing content as it often requires language-specific processing and sophisticated modelings for cross-language normalization and alignment. The problem becomes even more challenging when dealing with web data and for

---

[1] "page" and "document" interchangeably refer to the content of a web page.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3053–3061
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

low-resourced languages.

We optimize the cost for the latter approach to enable its application at scale. Specifically, we design CDA to project multilingual documents to a shared multilingual space for direct similarity measurement. Therefore, we can run the STRAND pipeline for multiple languages at once to keep the pre-processing cost *monotonic* with respect to the number of languages and documents. In particular, we design CDA with two key objectives: i) minimal data processing cost and ii) fast scaling to new languages. The latter is also crucial to the language expansion in online MT services. Our contribution is three-fold:

- We present an optimized, efficient, and scalable framework that can perform multilingual document alignment at state-of-the-art performance in one go.
- To facilitate the development and evaluation, we created two web-scale datasets, which are much larger and have many more languages than what is currently publicly available.
- We study the contribution of CDA in multiple applications involving parallel sentence extraction for MT and mutual complement between URL and CDA.

We tested CDA with multiple large-scale datasets of web documents. We also studied the applications of CDA within an industrial setting, including (i) extracting more and better parallel sentences from an extremely large-scale dataset, (ii) producing better MT models, and (iii) improving yield, measured by the amount of extracted parallel content, for both URL and CDA in the STRAND pipeline. The experimental results show that, despite its minimality, CDA (i) is on par with the top systems from WMT16-BDAST, which use expensive bilingual resources, and (ii) can double the amount of parallel data extracted by previous URL-based approaches. Most importantly, we show that CDA provides robust performance when dealing with millions of documents and processing up to 28 languages, including low-resourced languages.

In the remainder of this paper, we summarize the previous work regarding document alignment in Section 2. We then describe the proposed system and our experiments in sections 3 and 4, respectively. Finally, we derive the conclusions of the paper in Section 5.

2 Related Work

Aligning multilingual documents is the key required processing in most multilingual text processing pipelines, including cross-lingual information retrieval (Steinberger et al., 2002; Pouliquen et al., 2004; Vulic and Moens, 2015; Jiang et al., 2020) and parallel data extraction. In the context of creating parallel training data for MT, the problem has been studied in the literature for comparable corpora (Munteanu et al., 2004; Vu et al., 2009; Pal et al., 2014) and web-structured parallel data extraction (Resnik, 1999; Uszkoreit et al., 2010; Buck and Koehn, 2016a). We focus on the latter in this paper.

WMT-16 Bilingual Document Alignment Shared Task is a recent shared-task focusing on identifying bilingual documents from crawled websites (Buck and Koehn, 2016b). The top 3 systems are **YODA** (Dara and Lin, 2016), **NOVALINCS** (Gomes and Pereira Lopes, 2016), and **UEDIN1 COSINE** (Buck and Koehn, 2016b). The first two require costly features, such as (i) n-gram comparison after translating all non-English text into English (Dara and Lin, 2016), and (ii) phrase-table of statistical MT (SMT) as a dictionary (Gomes and Pereira Lopes, 2016). **UEDIN1 COSINE** (Buck and Koehn, 2016b), on the other hand, only uses TF × IDF-weighted with Cosine similarity. Interestingly, this method performs surprisingly well even without French-to-English translations, dropping just 3.4% in recall, from 93.7% to 90.3%. Though the finding can be due to the English and French lexicons' overlap, it suggests that TF × IDF with proper normalization is useful to compare document representations from sub-domains. Our proposed method exploits this aspect.

Given the advent of deep neural network modeling, document embeddings are among the main interests in general NLP applications (Le and Mikolov, 2014; Cer et al., 2018; El-Kishky and Guzmán, 2020). This line of research, however, is *not* technically related to our problem setting. Specifically, the cost to run a neural inference over a web-scale setting is prohibitively high, e.g., processing a dataset of several billion pages from CommonCrawl[2] is not feasible. To have an idea, the Cloud Translation[3] cost to translate a webpage having 20,000 characters is \$0.4 as of Jan 2021.

---

[2] `commoncrawl.org`

[3] `cloud.google.com/translate/pricing`

3 Proposed System

We describe our method to identify parallel pages from a web domain. Specifically, pages in different languages are projected to a shared space, where their similarity can be measured.

Problem Definition Let $D = \{d_1, d_2, \ldots, d_n\}$ be the n pages from a domain, each page d_i is described by its content c_i in language L_i. The problem is to identify all (d_i, d_j) of different languages, $L_i \neq L_j$, and c_i and c_j are translational equivalent.

Multilingual Space Let $\mathcal{L} = \{L_1, L_2, \ldots\}$ be the set of languages found in D and let D_i be the set of documents in L_i. Thus, $D = \bigcup_{L_i \in \mathcal{L}} D_i$, where each D_i is associated with a lexicon V_i. Without loss of generality, we project documents from two languages, L_α and L_β, into a common space as follows. We first define alignment between two lexicons V_α and V_β as:

$$\begin{aligned}\mathcal{A} = \big\{ &(a, b) : a \in V_\alpha, b \in V_\beta, \\ &P_{\alpha\to\beta}(a, b) + P_{\beta\to\alpha}(b, a) \\ &\geq P_{\alpha\to\beta}(a, w) + P_{\beta\to\alpha}(w, a), \forall w \in V_\beta \big\},\end{aligned}$$

where $P_{\alpha\to\beta}$ and $P_{\beta\to\alpha}$ are lexical translation models from L_α to L_β, and vice versa[4]. It should be noted that $\mathcal{A}$ defines a common space, $\mathbb{R}^{|\mathcal{A}|}$, where the dimensions are all word pairs. However, we can simplify the approach by mapping all languages in the space of a pivot language, i.e., α. Thus, we define $\Pi_\alpha : D_\beta \longrightarrow \mathbb{R}^{|V_\alpha|}$ that maps documents $d_\beta \in D_\beta$ into the same space of D_α, as:

$$\begin{aligned}\Pi_\alpha(d_\beta) &= \Pi_\alpha\Big(w_1, w_2, \ldots, w_{|d_\beta|}\Big) \\ &= \vec{x} = \big(x_1, x_2, \ldots, x_{|V_\alpha|}\big) \qquad (1)\end{aligned}$$

We define a lexical mapping for document d_β as $M(d_\beta) = d_{\beta\to\alpha} = \{w'_1, ..., w'_{|d_{\beta\to\alpha}|} | (w_i, w'_j) \in \mathcal{A}\}$, which maps d_β into language α. Similar, we denote the mapping for a document collection D_β as $D_{\beta\to\alpha} = \{M(d_\beta) : \forall d_\beta \in D_\beta\}$. Finally, we compute the TF×IDF representation of d_β as follows, $\forall w_i \in V_\alpha$:

- TF(w_i) = number of occurrences of w_i in $d_{\beta\to\alpha}$; and
- IDF$(w_i) = \log\Big(1 + \frac{|D_{\beta\to\alpha}|}{1+\#(w_i, D_{\beta\to\alpha})}\Big)$, where $\#(w, D)$ returns the number of documents in D containing w.

We compute x_i in Eq. 1 using TF(w_i) × IDF(w_i).

Aligning Multilingual Documents Two documents are considered a good pair if their representations are similar, according to a similarity threshold t. We compute the similarity between d_i and d_j as the dot-product between, $v_i \cdot v_j \in [0..1]$ (we normalized the vector representations with ℓ_2). In practice, we use English to build the multilingual space as it is the dominant language on the Internet and in most multilingual websites.

4 Experiments

We examine the efficacy of CDA in this section. First, we describe (i) the pipeline setup for the experiments and (ii) our effort in creating suitable benchmark data and selecting relevant resources in Section 4.1, and Section 4.2, respectively. We then address the following performance aspects of CDA:

1. The performance in multilingual document alignment.
2. The impact of CDA, compared to URL, in an end-to-end STRAND pipeline.
3. The by-product applications of CDA in identifying (i) novel language identifiers beyond ISO 639 for URL and (ii) web-domains containing multilingual data that are not detectable using language identifiers.
4. The cost required to enable the support to a new language.

4.1 Pipeline Setup

Figure 1 depicts the STRAND pipeline for our experiments.

- The **input** is constituted by web documents of multiple domains. The **output** is a set of parallel sentence pairs extracted from the pipeline. Each document has a web address and a raw HTML source.
- The **pre-processing** step groups input documents by domain to create data for document alignment step using URL and CDA. For CDA, additionally, it extracts the

---

[4] It can be easily shown that the proposed aligned is symmetric, i.e., the other condition $P_{\alpha\to\beta}(a, b) + P_{\beta\to\alpha}(b, a) \geq P_{\alpha\to\beta}(w, b) + P_{\beta\to\alpha}(b, w), \forall w \in V_\alpha$ holds.

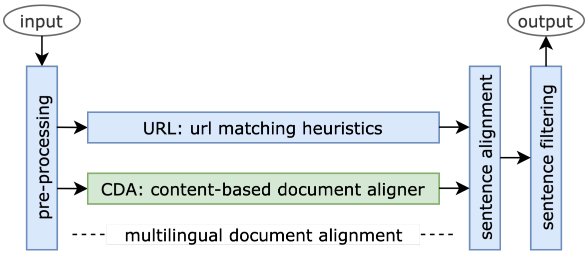


Figure 1: A STRAND pipeline that uses URL to align document and CDA as a replacement.

text content from HTML structure, using the following tags: `title, h1..h6, label, blockquote, dd, dt, p, pre, q, div`. This helps remove boilerplate effectively from being considered in the calculation. We use Python's `langid` package to identify the language of a page.

- **Document alignment** is performed by either URL or CDA. For URL, we use a similar set of language identifiers from BDAST's baseline[5].
- For each aligned document pair, the **sentence alignment** step aligns text segments, called sentence pair candidates, of the aligned pages based on the DOM structure (Smith et al., 2013).
- Finally, the **sentence filtering** step removes low-quality pairs (Xu and Koehn, 2017; Sánchez-Cartagena et al., 2018) or duplications. The filter we used in this experiment has approximately 90% F1 score for each language pair.

4.2 Dataset and Resource

We describe the datasets and resources used in the experiments.

4.2.1 Dataset

We collect and create the following datasets to study CDA performance in (i) matching parallel content, (ii) handling large datasets, and (iii) extending its use to new languages.

WMT-16 Shared Task First, we use the benchmark dataset provided for WMT-16 Shared Task on Bilingual Document Alignment. We evaluate and compare CDA with other English–French document alignment methods on the BDAST's training set. The dataset consists of 348,858 and 225,043 English and French documents from 49 web-domains, respectively. Each document has a web address and a clean content. Besides, French documents are translated into English using a standard SMT model. This translation is to study the potential upper-bound performance when having full translations. An alignment candidate has one document from each language, English or French, from the same domain. Thus, there are more than 4.2e9 possible alignments between the documents. The golden data has 1,624 pairs provided by WMT16-BDAST. In this set, the number of labeled alignments per domain ranges from 4 (e.g., `www.eohu.ca`) to 236 (e.g., `tsb.gc.ca`). The pairs generated by a system are first filtered by 1-1 rule: each document should participate in at most one alignment. A system is evaluated based on the recall achieved on these 1,624 pairs.

WMT-16 Deep Crawl The previous benchmark has two limitations. First, the size of the dataset is relatively small compared to a typical web-scale setting[6]. Second, the choice of English–French is not representative of the ultimate goal — finding more and better parallel data to enable MT in low-resourced languages. English–French has been the most studied pair in MT task. Besides, their lexicons are also highly overlapped (Lewis, 2009).

We address this problem, creating a larger dataset of more than 14MM pages using the same set of 49 domains. Specifically, we used these domains and URLs as seeds and recursively downloaded all reachable pages from those seeds. We did not download pages that link to external domains. This exercise resulted in a dataset consisting of 8.7MM and 5.5MM pages for English and 28 other languages. These languages include: Arabic, Bulgarian, Chinese Simplified, Chinese Traditional, Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French, German, Hebrew, Hungarian, Indonesian, Italian, Japanese, Korean, Malay, Norwegian, Polish, Portuguese, Russian, Slovak, Spanish, Swedish, Thai, and Turkish.

CommonCrawl Sextet Previous datasets share the same domains that are heavily biased toward French content (see Table 3). We leverage a monthly crawl from CommonCrawl, specifically

---

[5] https://github.com/christianbuck/wmt16-document-alignment-task/blob/master/languagestripper.py

[6] A typical multilingual domain can have thousands to millions of pages; e.g. `nato.int` and `microsoft.com` have 3e5 and 38e6 pages, respectively, indexed by Google[7].

Dataset	#Dom.	#EN-docs	#XX-docs	#$\mathcal{L}$
WMT-16 Shared Task	49	348,858	225,043	En+Fr
WMT-16 Deep Crawl	49	8.7MM	5.5MM	En+28
CommonCrawl Sextet	662	17.6MM	4.1MM	En+6

Table 1: Summary of Benchmark Datasets for Document Alignment Task. Each dataset is described by the number of web-domains (#Dom.), English documents (#EN-docs), non-English documents (#XX-docs), and languages (#$\mathcal{L}$).

`CC-MAIN-2017-17`[8], to create a better distribution dataset to validate CDA. We select pages from the crawl having pages in Chinese, Czech, Italian, Japanese, Russian, and Turkish. We only keep the following top-level domains: `.cn`, `.tw`, `.cz`, `.it`, `.jp`, `.ru`, `.tr`, `.edu`, `.gov`, and `.org`. The process results in a dataset of 600+ domains with 17.6MM and 4.1MM pages in English and six selected languages. Table 1 summarizes the datasets considered in our experiments.

4.2.2 Resource

Lexical translation dictionaries are the significant resource required in our proposed method to support a new language pair. Our experiments used the lexical translation dictionaries created by methods introduced for traditional SMT (Brown et al., 1993) and neural-based MT (Lample et al., 2018).

In particular, we use IBM-1 models for popular languages that have sufficient parallel data from general domains (Koehn, 2005). We use the GIZA++ toolkit (Och and Ney, 2003) to create IBM-1 models. Collecting such parallel data for the low-resourced languages is generally challenging. We instead leverage the advances in multilingual embeddings from the MUSE project[9] (Lample et al., 2018). We create translation probability between two words by their normalized embedding similarity score.

4.3 Bilingual Document Alignment Results

We evaluate the performance of CDA under the WMT-16 Shared Task benchmark. We conduct experiments on both settings, using the original text and using full translations. The latter setting allows us to understand the possible benefit of the expensive step, full document translation. Besides, the construction of V_{L_i} is crucial to the distinction of the representations. Therefore, we examine the impact of the vocabulary size of V_{L_i} to the alignment result. Specifically, we construct V_{L_i} by selecting the top frequent tokens after removing stop-words and the first k frequent tokens. We empirically set k to be 100. We experiment with three different sizes for V_{L_i}: 2,000, 10,000, and 20,000. Finally, we compare the results of CDA with the baseline URL and the top-3 systems of the WMT-16 Bilingual Document Alignment Shared Task. The evaluation metric is the percentage of the 1,624 golden pairs found in the top-1 alignment for each English document. Table 2 shows the result.

baseline: URL	67.9	
content-based systems	**align w. org. text**	**align w. trans. text**
YODA (Dara and Lin, 2016)	n/a	**93.71**
NOVALINCS (Gomes and Pereira Lopes, 2016)	90.50	n/a
UEDIN1 COSINE (Buck and Koehn, 2016b)	90.25†	93.65†
CDA: |V|=2,000	87.17	90.95
CDA: |V|=10,000	**90.40**	91.02
CDA: |V|=20,000	90.33	89.90

[†] *average performance of valid and test splits*

Table 2: Comparison of the baseline URL, top-3 performing systems from WMT16-BDAST and CDA under different settings of vocabulary size, using either original text (original documents in English and French) or translated text (documents in English and the English translations of documents in French)

For alignments using original text, the results indicate that CDA achieves similar performance with the state-of-the-art methods from BDAST. The result shows the efficacy of the proposed alignment method. The results also show that the vocabulary size, or the vector representations' size, impacts the performance. $|V| = 10,000$ yields the best result among the three settings. Second, even though using full translation is better, the performance gains are negligible with respect to the processing cost required for building the MT models and translating all the data into an anchor language. Since CDA does not exploit bi-gram features, its performance is relatively lower, up to 3%, compared to the state of the art. In short, the result suggests an optimal configuration for CDA with a vocabulary size of 10,000.

---

[8] `s3://commoncrawl/crawl-data/CC-MAIN-2017-17`

[9] github.com/facebookresearch/MUSE

4.4 Multilingual Document Alignment Results

It was showed in WMT16-BDAST that content-based methods could add 60% more English–French document pairs compared to URL. This section aims to verify this in a multilingual setting, mainly when operating with a significantly higher number of languages and domains using WMT-16 Deep Crawl and CommonCrawl Sextet, respectively.

On WMT-16 Deep Crawl Table 3 shows the number of parallel documents and sentences extracted by an end-to-end STRAND pipeline, after filtering and duplication removal, described in Section 4.1. The result shows that CDA contributes an extra of 53%, 75%, and 195% in *clean* parallel sentences compared to URL for French alone, and when French is and is not considered, in this more extensive and more realistic setting, respectively. It also suggests that CDA is effective and can significantly increase the number of parallel sentences extracted for low-resourced languages. Finally, the result indicates that our proposed method is robust in a multilingual setting.

On CommonCrawl Sextet Table 4 shows the result in English parallel tokens extracted from the pipeline using URL and CDA in the document alignment step. The result shows similar gains as in the previous experiment, except for Czech — increasing $7\times$ more parallel tokens. Our post-hoc analysis discovers non-standard language identifiers missing for URL processing, e.g., `ces` or `cesky`.

To confirm the study, we randomly selected 1,320 English–Turkish document pairs identified by CDA for human verification since we do not have annotated data. The outcome indicates that the accuracy of the document pairs is at 91.5%. Information on these datasets is described here: `github.com/alexa/wqa_dataset`.

4.5 Industrial Benchmarks

We conducted multiple internal experiments to examine the performance of CDA over URL under an industrial setting. Specifically, we focus on three application aspects of CDA: (i) robustness, (ii) identifying non-standard language identifiers for URL, (iii) identifying multilingual web-domains. Due to business security reasons, we do not name the specific languages considered in this study. We do not provide some details of the experiment setting, which are not critical to illustrate our findings.

Language	# Document Pairs		# Sentence Pairs		
	URL	CDA	URL	CDA	CDA\URL
Arabic	3,266	2,896	36,262	39,590	24,065
Bulgarian	1,184	1,070	9,292	1,748	1,359
Chinese-S	2,805	2,160	30,519	27,666	13,289
Chinese-T	316	102	2,584	2,055	374
Croatian	704	3,119	889	56,300	55,854
Czech	29	241	77	7,264	7,248
Danish	137	2,932	693	39,488	38,996
Deutsch	5,525	8,863	83,663	170,932	113,851
Dutch	599	2,407	8,228	79,293	79,146
Farsi	1,316	1,404	14,697	13,875	6,122
Finnish	170	1,313	355	12,403	12,229
French	115,671	143,972	2,653K	3,568K	1,411K
Hebrew	209	140	7,742	5,295	83
Hungarian	1,253	1,382	10,494	6,158	4,448
Indonesian	368	551	625	1,204	900
Italian	6,644	7,310	57,977	94,098	55,802
Japanese	823	1,475	6,593	14,720	11,138
Korean	913	136	13,365	2,229	224
Malay	1,040	1,904	8,467	13,088	7,213
Norwegian	67	1,875	196	35,362	35,273
Polish	557	934	9,685	21,528	17,255
Portugese	1,545	6,200	12,294	104,561	96,850
Russian	3,984	2,475	36,565	55,010	40,722
Slovak	170	850	211	2,157	2,106
Spanish	8,334	21,765	114,874	317,430	252,523
Swedish	83	2,394	2,773	49,420	49,238
Thai	82	10	830	259	40
Turkish	1,057	2,041	9,598	18,412	10,789
All	159,343	222,283	3,134K	4,761K	2,349K
All\French	43,672	78,311	481,333	1,193K	937,683

Table 3: Comparison of yields, number of clean document pairs and sentence pairs, produced via URL and CDA on WMT-16 Deep Crawl. Column **CDA\ URL** reports the number of novel sentence pairs exclusively extracted via CDA.

4.5.1 Robustness Benchmark

We ran the STRAND pipeline end-to-end to extract parallel sentence pairs from document pairs identified by URL and CDA replacing URL. We employ a crawl dataset larger than a typical monthly crawl archive from CommonCrawl. The dataset is also considered densely multilingual. We target six mid-tier languages that are *not* in the top-10 high-resourced languages. It shows that CDA can increase additional 27% English parallel tokens over the selected languages.

Automatic Evaluation We first study the quality of the extracted parallel data, especially the addition of 27% produced by CDA, using automatic MT evaluation. Specifically, for each language, we compare the translation models trained by two equal-sized parallel sentence pairs sampled from the exclusive pairs extracted by URL and CDA individually, i.e., after removing common pairs extracted by both methods. We train vanilla

Lang.	#Dom.	#EN-docs	#XX-docs	$\frac{\lvert CDA\rvert}{\lvert URL\rvert}$
Turkish	37	1,434,923	71,034	1.77
Czech	69	2,333,914	831,072	7.07
Japanese	84	2,097,664	757,872	1.90
Russian	125	2,918,594	1,163,258	1.09
Italian	239	5,770,684	1,112,868	1.05
Chinese	108	3,061,782	207,211	0.99
All	662	17,617,561	4,143,315	1.24

Table 4: Parallel English tokens extracted by CDA and URL on CommonCrawl Sextet

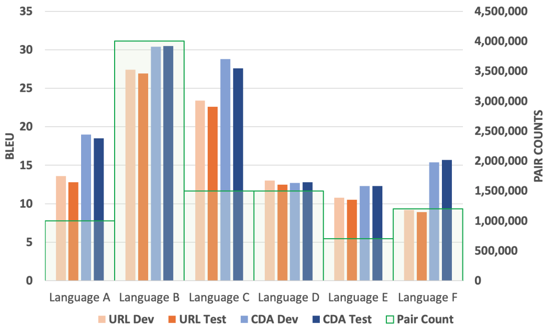


Figure 2: Automatic evaluation of MT models trained by the same amount of pairs sampled from pairs extracted exclusively by either URL or CDA. The primary y-axis (left) indicates the BLEU score, while the secondary y-axis (right) indicates the number of parallel pairs.

`seq2seq` models using Sockeye[10]. We report the MT performance in BLEU scores on our MT evaluation data in Figure 2. The results indicate that the models trained using novel sentence pairs extracted by CDA consistently give better translation models.

Human Evaluation We had linguists manually verify the extracted parallel sentence pairs produced by the pipeline using either URL or CDA. Specifically, we randomly sampled 500 sentence pairs extracted from each pipeline using either URL and CDA for Language A and Language E for human evaluation (we do not remove the common pairs in this evaluation). The selection of these languages is based on their low performance reported during the automatic evaluation in Figure 2. Table 5 shows the result in terms of precision and recall. In general, we find that the quality of pairs produced by both methods is typically comparable. The result also confirms the robustness of our proposed CDA under stress evaluations.

	URL	CDA
Language A	P=96, R=87	P=98, R=89
Language E	P=94 ,R=81	P=92, R=90

Table 5: Human verification for parallel sentence pairs extracted from URL and CDA in precision (P) and recall (R).

4.5.2 Identifying Non-standard Language Identifiers for URL

Even though URL method operates decently fast, it requires language identifiers usually collected manually. This task is challenging as language identifiers used in different web-domains typically do not follow any standard. For example, the language patterns for Czech may include `czech`, `cze`, `ces`, `cz`, `cs`, `cesky`. The result in Table 4 also suggests the limitation of URL for Czech language. In this exercise, we examine URL pairs matched by CDA method to extract relevant language identifiers. Specifically, we focus on URL pairs distanced by *one* token and extract the different tokens as candidates. For example, `en` and `vi_vn` are extracted as candidates for (`www.visitsingapore.com/en/`, `www.visitsingapore.com/vi_vn/`). We curated the candidates, identified additional novel language identifiers, and feeded them to URL method. The exercise helped significantly increase the yields for multiple low-resourced languages at 453%, 295%, and 266%. For example, we found additional identifiers for Chinese language: `chs`, `chn`, `c`, `zho`, `zht`, `cht`, `webcn`, `sc`, `tc`, `chinese_gb`, `chinese_big5`, besides other popular `zh`, `chi`, and `zho`.

4.5.3 Identifying Multilingual Web-domains

We study the application of URL and CDA in identifying densely multilingual web-domains. Specifically, we compare the yield of parallel content, in the total number of extracted parallel English tokens, from two different datasets processed by the same pipeline. The datasets differ in whether their web-domains are identified as multilingual by URL or CDA.

On a sufficiently large dataset, we first ran URL and selected those web-domains having at least 100 candidate pairs. Subsequently, we ran CDA and selected those with at least 100 candidate pairs on the remaining of the dataset, i.e., those not selected by URL. We randomly selected 10,000 domains from

---

[10] https://github.com/awslabs/sockeye

	Domains by URL	**Domains by CDA**
Parallel Token Count	23.3MM	69.5MM
Host Count	10,000	10,000
Dataset Size (TB)	6	3.1

Table 6: Number of parallel English tokens (in MM) from two deep crawls seeded by URL and CDA.

each group to create the two datasets, namely "Domains by URL" and "Domains by CDA," respectively. We applied the same pipeline using both methods for aligning documents on each dataset. We then computed the yield of parallel English tokens extracted from each setting. Table 6 shows the results. These indicate that CDA can identify densely multilingual web-domains effectively.

In particular, given the same number of web-domains, the dataset identified by CDA can produce almost 3× more parallel data with a size of only half of the dataset identified by URL. This finding suggests that the yield of parallel content from web-domains identified by CDA is 6× higher than those identified by URL. The finding is essential in optimizing the parallel extraction pipeline and identifying better densely multilingual web content.

4.6 Cost Analysis

As presented, we anticipate two cost types when extending CDA to support a new language: (i) building a lexical translation model and (ii) processing more documents. The former is a one-time cost, while the latter is dataset dependent.

Specifically, we have shown in Section 4.2.2 that a lexical translation model can be built using either statistical method IBM-1 with parallel data or neural-based unsupervised method (Lample et al., 2018); we observed comparable performance of CDA when using a model built by these methods. Given the rapid advance in deep neural language models, it is increasingly possible to obtain such resources for low-resourced languages. This suggests that we will be able to leverage recent advances in neural-based NLP to continuously extend CDA for many more languages.

Regarding the execution time, the primary bottleneck typically is due to the scoring of all possible alignments between English and non-English documents. Even though this scoring step is quadratic, this workload is perfectly parallel. With proper engineering optimization, we empirically found out that it is possible to bring the run-time for CDA to be within 2.5× than the one of URL's for 20 low-resourced languages and on a sufficiently large dataset. This optimized cost is crucial in enabling a spectrum of multilingual applications, including cross-lingual information retrieval and enabling MT services for scarce languages.

5 Conclusion

We presented our content-based document alignment for web data, CDA, which projects multilingual documents to a common space for similarity calculation. We also described our effort to collect and create benchmark datasets to study different performance aspects of the proposed method. The results show that CDA is efficient when projecting multilingual documents in one go for as many as 28 languages. Moreover, we also explain the different types of benchmarking for CDA under industrial settings.

The results show that our proposed method is robust when processing huge datasets and useful in identifying non-standard language identifiers and multilingual web-domains. Finally, and most importantly, the only significant resource required by CDA is the lexical translation dictionary: this can be easily built thanks to the recent advance in learning multilingual embeddings.

Future applications of CDA can be many. For example, the URLs paired with CDA can be used to improve the coverage for URL-based methods (e.g., Czech case in Table 4) and to study the web structure of multilingual content. Moreover, the robustness and extensibility of CDA make it applicable to other multilingual processing systems, including cross-lingual search and retrieval.

Acknowledgements

We thank Chris Bissell, Jeremiah Hankins, Kwang Hyun Jang, Daniel Marcu, Masud Moshtaghi, Akash Patel, Chris de Vries, Renxia Wang, William Wong, and members of the Alexa AI Search team in Manhattan Beach, CA for their helpful feedback on earlier drafts of the manuscript. We also thank the anonymous reviewers for their thoughtful suggestions which led to an improved manuscript.

References

Peter F. Brown, Stephen Della-Pietra, Vincent Della-Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation. *Comp.Ling.*

Christian Buck and Philipp Koehn. 2016a. Findings of the wmt 2016 bilingual document alignment shared task. In *WMT 2016*, pages 554–563, Berlin, Germany.

Christian Buck and Philipp Koehn. 2016b. Quick and Reliable Document Alignment via TF/IDF-weighted Cosine Distance. In *WMT 2016*, pages 672–678, Berlin, Germany.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *EMNLP: System Demonstrations*, pages 169–174, Brussels, Belgium.

Aswarth Abhilash Dara and Yiu-Chang Lin. 2016. YODA system for WMT16 shared task: Bilingual document alignment. In *WMT 2016*, Berlin, Germany.

Ahmed El-Kishky and Francisco Guzmán. 2020. Massively multilingual document alignment with cross-lingual sentence-mover's distance. In *AACL-IJCNLP 2020*, pages 616–625, Suzhou, China.

Luís Gomes and Gabriel Pereira Lopes. 2016. First steps towards coverage-based document alignment. In *WMT 2016*, pages 697–702, Berlin, Germany.

Zhuolin Jiang, Amro El-Jaroudi, William Hartmann, Damianos Karakos, and Lingjun Zhao. 2020. Cross-lingual information retrieval with BERT. In *Proceedings of the workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 26–31, Marseille, France. European Language Resources Association.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit 2005*, pages 79–86, Phuket, Thailand.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *ICLR 2018*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China. PMLR.

M. Paul Lewis, editor. 2009. *Ethnologue: Languages of the World*, sixteenth edition. SIL International, Dallas, TX, USA.

Dragos Stefan Munteanu, Alexander Fraser, and Daniel Marcu. 2004. Improved machine translation performance via parallel sentence extraction from comparable corpora. In *HLT-NAACL 2004*, pages 265–272, Boston, Massachusetts, USA.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Santanu Pal, Partha Pakray, and Sudip Kumar Naskar. 2014. Automatic building and using parallel resources for SMT from comparable corpora. In *HyTra*.

Bruno Pouliquen, Ralf Steinberger, Camelia Ignat, Emilia Käsper, and Irina Temnikova. 2004. Multilingual and cross-lingual news topic tracking. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 959–965, Geneva, Switzerland. COLING.

Philip Resnik. 1999. Mining the web for bilingual text. In *ACL 1999*, pages 527–534, College Park, Maryland, USA.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Víctor M. Sánchez-Cartagena, Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez-Sánchez. 2018. Prompsit's submission to wmt 2018 parallel corpus filtering shared task. In *WMT 2018*, Brussels, Belgium.

Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. 2013. Dirt cheap web-scale parallel text from the common crawl. In *ACL 2013*, pages 1374–1383, Sofia, Bulgaria.

Ralf Steinberger, Bruno Pouliquen, and Johan Hagman. 2002. Cross-lingual document similarity calculation using the multilingual thesaurus EUROVOC. In *Computational Linguistics and Intelligent Text Processing, Third International Conference, CICLing'2002*, number 2276 in Lecture Notes in Computer Science, LNCS, pages 415–424. Springer-Verlag.

Jakob Uszkoreit, Jay Ponte, Ashok Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *COLING 2010*, China.

Thuy Vu, AiTi Aw, and Min Zhang. 2009. Feature-based method for document alignment in comparable news corpora. In *EACL 2009*, pages 843–851, Athens, Greece.

Ivan Vulic and Marie-Francine Moens. 2015. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. pages 363–372. ACM; New York, NY.

Hainan Xu and Philipp Koehn. 2017. Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *EMNLP 2017*, Denmark.

Metric-Type Identification for Multi-Level Header Numerical Tables in Scientific Papers

Lya Hulliyyatus Suadaa[1], **Hidetaka Kamigaito**[1], **Manabu Okumura**[1], **Hiroya Takamura**[1,2]
[1]Tokyo Institute of Technology
[2]National Institute of Advanced Industrial Science and Technology (AIST)
`lya@stis.ac.id`
`{kamigaito,oku}@lr.pi.titech.ac.jp`
`takamura@pi.titech.ac.jp`

Abstract

Numerical tables are widely used to present experimental results in scientific papers. For table understanding, a metric-type is essential to discriminate numbers in the tables. We introduce a new information extraction task, metric-type identification from multi-level header numerical tables, and provide a dataset extracted from scientific papers consisting of header tables, captions, and metric-types. We then propose two joint-learning neural classification and generation schemes featuring pointer-generator-based and BERT-based models. Our results show that the joint models can handle both in-header and out-of-header metric-type identification problems.

Models	Task 1		Task 2	
	Prec	**Rec**	**Prec**	**Rec**
Model A	60	60	60	60
Model B	70	70	70	70
Model C	80	80	80	80
Model D	90	90	90	90

Table X. Model comparison in Task 1 and 2.

(a) Metric-type in header

Models	Task 1	Task 2
Model A	60	65
Model B	70	75
Model C	80	85
Model D	90	95

Table Y. Model comparison in Task 1 and 2 (**F-score**).

(b) No metric-type in header

Figure 1: Example tables in scientific papers. Bold indicates their metric-type.

1 Introduction

Tables are powerful tools for presenting data efficiently in row and column views. In scientific papers, numerical tables are commonly used to show experimental results for facilitating data analysis. Examples of numerical tables in scientific papers are shown in Figure 1.

Tables have the ability to cover multiple categories written in table headers by incorporating several header sets in a hierarchical view, called multi-level header tables. Scientific papers have strict guidelines about tables; for example, one states that a similar type of text is written in the same level of header. Figure 1a shows a multi-level header example in the column part, with task type (*Task 1* and *Task 2*) in the first header-level and metric-type (*Prec* and *Rec*) in the second. The table also has a row header specifying the model type (*Model A*, *Model B*, *Model C*, and *Model D*). In the real-world, this header-type information is limited due to the unknown table scheme. However, we assume tables in scientific papers follow the rule of categorizing a similar type of header name in the same header-level.

To understand the numbers in the tables, metric-types are important for discriminating the numbers. A comparison between numbers is applied for numbers in the same metric-type with different categories. For the table in Figure 1a, we cannot compare the number 60 for *Model A* in the first column with 60 in the second one because they have a different metric-type: *Prec* and *Rec*. Computing numbers with different metric-types will result in inaccurate analysis.

Different tables may have different ways of writing their header name, such as using abbreviations like *p*, *pre*, or *prec* to refer to *precision*. Due to the lexical diversity of header names, metric-type identification becomes more challenging. Using a rule-based metric-type tagging or a limited set of metric-types in a dictionary is not enough to cover the diversity. Since tables in scientific papers typically have logical captions and logical categorization of the header-level, we introduce a metric-type identification task that locates the metric-type in the headers by using the caption and header name as inputs. For the example shown in Figure 1a, the metric-type is located in the second level of the column header.

We also cover tables that do not mention metric-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3062–3071
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

types in their header (out-of-headers), as shown in Figure 1b. In these cases, the metric-types are identified in the caption. To cover metric-types located both in the headers and not in the headers, we propose a joint framework of metric-type location prediction and metric-type token generation for the metric-type identification task in multi-level header tables.

Our contributions are as follows:

- We introduce a metric-type identification task for multi-level header tables and propose joint location prediction and generation models to solve the task.
- We provide a dataset consisting of multi-level header numerical tables, captions, and metric-types, extracted from scientific papers. Our datasets will be publicly available[1].
- We introduce a multi-level header table encoder mechanism to obtain table header representations and propose a pointer-generator-based model to cover out-of-headers in the metric-type identification task.
- We fine-tune a general pre-trained encoder (BERT) and a domain-specific encoder (SciBERT) in our task and present the experimental results. We show that the models incorporating the pre-trained encoders lead to significant performance gains, especially when using a domain-specific one.

2 Related Work

Table information extraction is beneficial to cover unknown table schemes and understand the table contents. Milosevic et al. (2019) proposed a framework for table information extraction in biomedical domains by defining rules for all possible variables. Specifically, for numerical variables, they retrieved metric-types by searching a set of possible tokens in the dictionary. Focusing on numerical tables, Nourbakhsh et al. (2020) extracted metric-types in earning reports by using similarity scores between the corresponding non-numeric text for the leftmost cells and stored metric-types.

The work closest to ours is the one by Hou et al. (2019), who used tables from the experimental result section, combined with the title and abstract as document representations to extract triples of tasks, dataset, and metric for leaderboard construction. In our study, we represent the tables in more generic ways, preventing the original table structure in the multi-level headers form. We intend to retain the ability of a table to cover complex categorization in the headers and efficiently present all values. A previous study that also explored multi-dimensional tables was done by Milosevic et al. (2016) to automatically detect table structures from XML tables.

Our pointer-generator-based model in the metric-type generation scheme is inspired by the promising results of the pointer-generator network (See et al., 2017) in the summarization task. The network deals with the out-of-vocabulary issue by joint copying from source texts and generating from vocabularies.

Recent studies have shown that pre-trained encoders can be successfully fine-tuned for downstream NLP tasks, thus avoiding the need to train a new model from scratch. A pre-trained encoder BERT (Devlin et al., 2019) was trained on the BooksCorpus (800M words) and Wikipedia (2,500M words). For better-contextualized representation in the scientific domain, Beltagy et al. (2019) introduced a domain-specific BERT model, SciBERT, which was trained on 1.14M papers from Semantic Scholar. Friedrich et al. (2020) implemented both BERT and SciBERT on their models to solve the information extraction task and achieved significant performance gains.

3 Metric-Type Identification for Numerical Tables

3.1 Datasets

We automatically extracted tables from the PDF files of scientific papers in the computational linguistics domain using PDFMiner and Tabula as extraction tools and filtered only numerical tables related to experimental results using the keywords *evaluation*, *result*, *comparison*, and *performance*. We used papers from the ACL and EMNLP conferences (2016 to 2019) on the ACL Anthology website as data sources.

In tables in actual scientific papers, knowledge about the table semantics is rarely provided. On the basis of how information is "read" from a table, Hurst (2000) separated functional table areas into access cells and data cells. Access cells consist of column headers and/or row headers. We define data structure on the basis of their functional areas: table caption (capt), row headers (rh), column headers

---

[1] Dataset is available on https://github.com/titech-nlp/metrictable

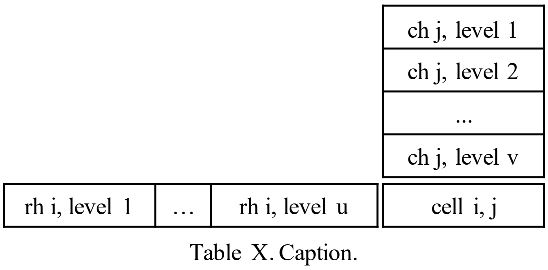


Figure 2: Table structure.

(ch), and cells. Headers in the row and column parts have several levels, and we assume that header names in the same level have the same type. Figure 2 shows our table structure.

We hired several qualified workers in the computer science field to manually check the extracted table structure to ensure the separation of row headers, column headers, and cells was correct, as shown in Figure 3. Then, they annotated the metric-type of the tables by prioritizing the locating of the metric-type in a specific header-level. The annotators were able to identify the metric-types of approximately 70% of the tables in their headers, and they determined the metric-type of the rest of the tables on the basis of information from the table captions. When no metric-type was mentioned in the headers, we assumed the metric-type was the same for all table values. The structures from the example in Figure 3 are capt: "*model comparison in task 1 and 2*"; rh level 1: [*models, models, models, models*]; rh level 2: [*model a, model b, model c, model d*]; ch level 1: [*task 1, task 1, task 2, task 2*]; ch level 2: [*prec, rec, prec, rec*]; and metric-type: [*prec, rec, prec, rec*] (identified in ch level 2).

We split our dataset into training, validation, and test sets. The statistics of our dataset are provided in Table 1.

	Train	**Val**	**Test**
No. of tables	1,084	136	135
Average row/column	6	6	5
Max level:			
- row header	9	6	4
- column header	6	5	6
Vocab size:			
- headers	8,270	1,435	1,230
- all metric-types	807	175	185
- unique metric-types	90	22	28

Table 1: Dataset statistics in training, validation, and test sets.

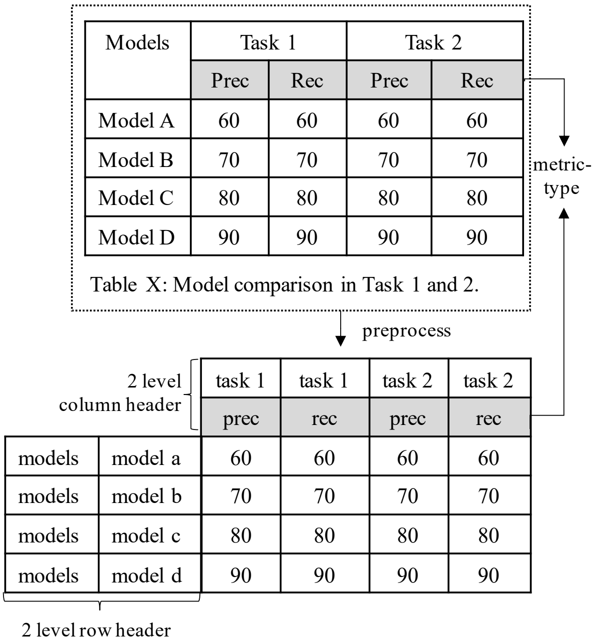


Figure 3: Illustration of table preprocessing.

3.2 Problem Definition

Let Table $= \{capt, rh_k^i, ch_l^j, cell_{ij}\}$, where $1 \leq i \leq n_r, 1 \leq j \leq n_c, 1 \leq k \leq u, 1 \leq l \leq v$ denote an $n_r \times n_c$ table with the u level of rh and v level of ch. The task is to identify metric-type set ($\hat{m}$) in the specific level of row header (rh_k) and column header (ch_l). To handle tables that do not include metric-types in their headers, we generate $\hat{m}$ by using information from the table caption. The formulation of the metric-type identification is as follows:

$$\hat{m} = \begin{cases} \{rh_k^i\}_{i=1}^{n_r}, k \in \{1, ..., u\} & \text{if } \hat{m} \text{ in } rh \\ \{ch_l^j\}_{j=1}^{n_c}, l \in \{1, ..., v\} & \text{if } \hat{m} \text{ in } ch \\ \{w_m\}_{\times j}, w_m \in W_m \text{ or} & \\ \qquad\qquad w_m \in capt & \text{otherwise,} \end{cases} \tag{1}$$

where W_m is a set of metric-types in the vocabulary.

4 Models

We propose neural models to identify the metric-type for multi-level header tables by means of a joint model of metric-type location prediction and metric-type token generation.

4.1 Pointer-Generator with Supervised Attention Model

We obtain the representations of captions and header-levels by using a BiLSTM encoder and then capture header-level weights using supervised attention between the header-level encoder and the metric-type header-location outputs. In the generation scheme, we adopt the pointer-generator network to take into account captions as source texts and the metric-type vocabulary in the metric-type generation gate. The architecture of our model is shown in Figure 4.

Header encoder We use the vector representation of each header-level by averaging the vectors of all header name tokens in the same level. Given E_{rh_k} and E_{ch_l} as the averages of the initial vector representations of the row and column header-level vectors, respectively, we use the BiLSTM encoder with the dot attention mechanism proposed by Luong et al. (2015) to obtain the representations of the row and column header-levels and select the last hidden state of the last level combined with the weighted hidden states as header-level contexts, as follows:

$$C_{rh} = [C_{rh_u}; \sum_{k=1}^{u} a_{rh_k} C_{rh_k}], \tag{2}$$

$$C_{ch} = [C_{ch_v}; \sum_{l=1}^{v} a_{ch_l} C_{ch_l}]. \tag{3}$$

Caption encoder As with the headers, we use the BiLSTM encoder with attention a_{capt_i} to compute the context vector of caption C_{capt}.

Metric-type header-location gates We feed the concatenation of the row and column header contexts to the softmax layer to obtain the metric-type header-location probability:

$$p_{hloc} = \text{softmax}([C_{rh}; C_{ch}]), \tag{4}$$

which includes the probabilities of the metric-types located in row headers (p_{rh}), located in column headers (p_{ch}), or not located in the headers (p_{capt}), where $p_{rh} + p_{ch} + p_{capt} = 1$.

Metric-type header-level gates Since the attention scores a_{rh_k} and a_{ch_l} capture the relevant header-level information in row and column, these attention scores are used as header-level weights as follows:

$$\mathrm{w}_{hlevel_i} = [a_{rh_k} p_{rh}; a_{ch_l} p_{ch}], \tag{5}$$

where $i \in \{1, ..., u, (u+1), ..., (u+v)\}$ as a header-level index.

Metric-type generation gates In our pointer-generator network, we use the sigmoid layer to obtain a switch copy probability:

$$p_{copy} = \text{sigmoid}(C_{capt}), \tag{6}$$

which lets us choose between copying word w_{capt} from a table caption and generating word w_m from the metric-type vocabulary, where $p_{copy} \in [0, 1]$. We use a softmax function to compute the probability distribution over the metric-type vocabulary:

$$P_{vocab}(w_m) = \text{softmax}(C_{capt}). \tag{7}$$

Then, we obtain the following probability distribution over the extended vocabulary:

$$\begin{aligned} P(w_m) = p_{copy} \sum_{i:w_i=(w_m)}^{n} a_{capt_i} + \\ (1 - p_{copy}) P_{vocab}(w_m), \end{aligned} \tag{8}$$

where i is the index of metric-type tokens in the vocabulary.

Learning objective For training, we exploit the negative log-likelihood objective as the loss function. In addition, we adopt supervised attention (Liu et al., 2016) for jointly supervising the row and column header-level attention to obtain the metric-type header-level. We combine all loss functions in the location classification and token generation model, and define α as the weight as follows:

$$\begin{aligned} \mathcal{L} = -((1 - \alpha)(\sum_{c} z_{hloc} \log p_{hloc_c} + \\ \sum_{i=1}^{u+v} \log \mathrm{w}_{hlvl_i}) + \alpha(\log p_{copy} + \\ \log P_{vocab}(w_m))), \end{aligned} \tag{9}$$

where $c \in \{capt, rh, ch\}$ is the metric-type header-location classes and z_{hloc} is the binary indicator (0 or 1) of each corresponding class.

4.2 Fine-tuning BERT-based Model

Input representation Input text in a fine-tuned BERT-based model is preprocessed by inserting two special tokens, [CLS] and [SEP]. In the original BERT architecture, [CLS] is appended to the beginning of input as the representation of the entire input sequence, and [SEP] is inserted after each input type as a sign of a segment boundary. For example, in question-answering tasks with two

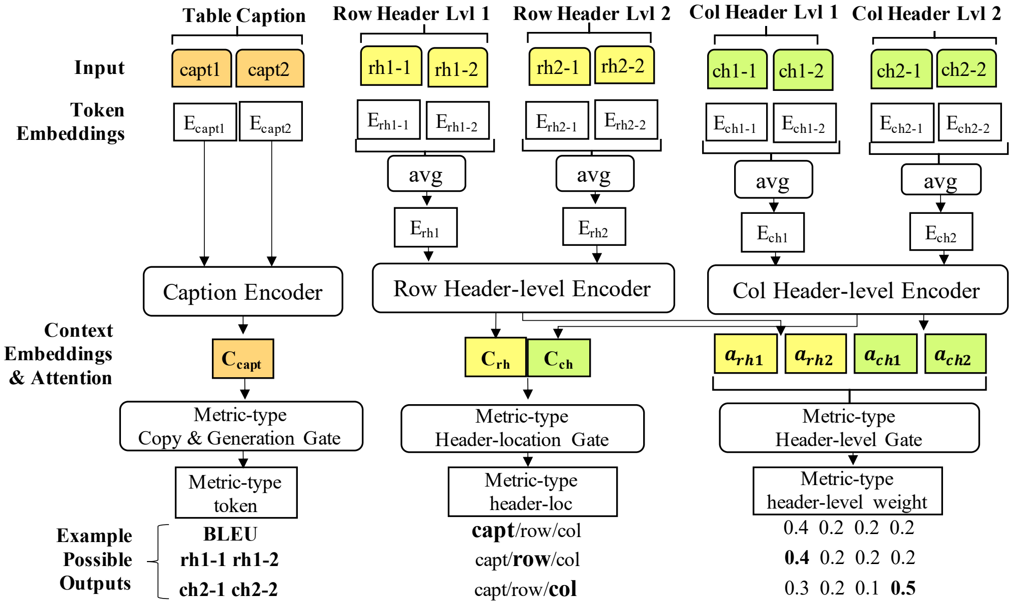


Figure 4: Architecture of proposed pointer-generator-based model to identify metric-types in tables.

types of input text, pairs of question and answer, a [CLS] token is appended before question tokens, and [SEP] tokens are placed after question and after answer tokens, to separate the question and answer segments. Following Liu and Lapata (2019), we customize these preprocessing schemes by inserting [CLS] before each segment and inserting [SEP] after each segment. We divide our inputs into several segments: caption, row header level 1 to u, and column header level 1 to v.

The input text after preprocessing is denoted as a sequence of tokens $X = (x_1, x_2, \cdots, x_n)$. There are three kinds of embedding assigned to each x_i: token embeddings representing the meaning of each token, segmentation embeddings indicating the segment boundaries of a sequence of tokens, and position embeddings covering token position within the sequences. Since BERT only covers two segments in its input, we treat the odd segment as segment A and the even one as segment B. The sum of these three embeddings is fed to a bidirectional Transformer layer of BERT.

We use the token representations from the top hidden layers of the pre-trained Transformer as context embeddings. We assume the context vectors of each [CLS] token can represent the segment sequences better. As shown in Figure 5, we denote the input embedding as E, the final hidden vector of the [CLS] token for the i^{th} input segment as $C_i \in \mathbb{R}_H$, and the final hidden vector for the j^{th} input token as $T_j \in \mathbb{R}_H$.

We use a metric-type header-location gate and a metric-type header-level gate for metric-type location classification, and a metric-type generation gate to generate metric-type tokens from vocabulary covering out-of-header metric-types. Our BERT-based model architecture is shown in Figure 5.

Metric-type header-location gates We feed the first segment context C_1 to the softmax layer to obtain the metric-type header-location probability:

$$p_{hloc} = \text{softmax}(C_1). \tag{10}$$

Metric-type header-level gates In our task, segments are used to represent the table section that is most related to metric-type. We incorporate the segment context C_i to the sigmoid layer to obtain the probability of the metric-type being located in a specific header-level:

$$p_{hlevel_i} = \text{sigmoid}(C_i). \tag{11}$$

The probabilities are then normalized to all segments as a weight score of the header-level:

$$\text{w}_{hlevel_i} = \frac{p_{hlevel_i}}{\sum_{i=1}^{n} p_{hlevel_i}}. \tag{12}$$

Metric-type generation gates We use a softmax function based on the first segment context C_1 to compute a probability distribution over the metric-type vocabulary:

$$P_{vocab}(w_m) = \text{softmax}(C_1). \tag{13}$$

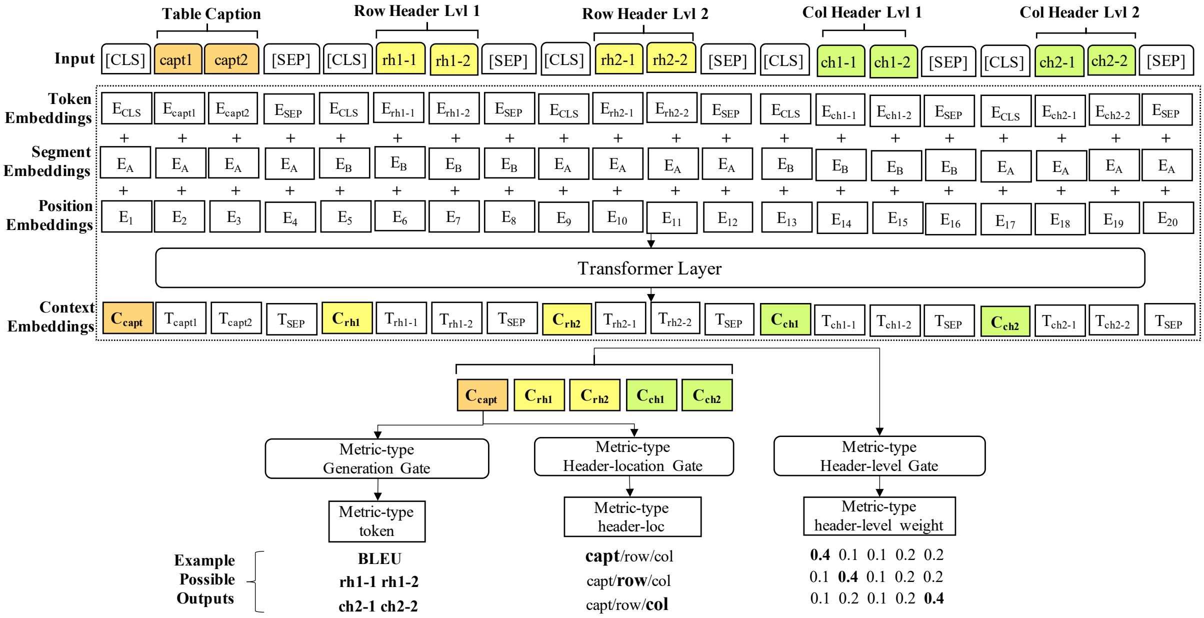


Figure 5: Architecture of proposed BERT-based model to identify metric-types in tables.

Learning objective We combine all loss functions in the metric-type header-location, metric-type header-level, and metric-type generation gates:

$$\mathcal{L} = -((1-\alpha)(\sum_{c} z_{hloc} \log p_{hloc_c} + \sum_{i=1}^{n} \log \mathrm{w}_{hlvl_i}) + \alpha \log P_{vocab}(w_m)), \quad (14)$$

where α is the weight of the metric-type generation functions.

5 Experimental Settings

5.1 Baseline Model

We use two SVM classification models as baselines: a metric-type location prediction model and a metric-type token prediction model from the vocabulary of metric-types. We use tf.idf of the concatenation header name tokens for all levels as input representations in the first model and tf.idf of the caption tokens in the second one. We tuned hyperparameters of the SVM model and reported the best results.

5.2 Metrics Evaluation

We use accuracy metrics to evaluate the metric-type location and generated metric-type tokens.

Metric-type location accuracy The target of the metric-type location prediction model is the metric-type located in the row headers, in the column headers, or not found in the headers. The accuracy of header-location (acc_{hloc}) is the rate of correct header-location predictions.

Since details about the metric-type location in the header-level are needed to identify metric-type token lists, we also compute the accuracy of metric-type header-level (acc_{hlevel}) using the ratio of correct header-level predictions to the total number of predictions.

Metric-type token accuracy Let $\hat{m} = (\hat{w}_{m_1}, ..., \hat{w}_{m_n})$ denote the sequence of predicted metric-type tokens for n_r rows or n_c columns (depending on the header-location prediction), and $m = (w_{m_1}, ..., w_{m_n})$ denote the target ones: for example, $\hat{m} = (\text{f1, f1, f1})$ and $m = (\text{f-1, f-1, f-1})$. We calculate the metric-type token accuracy using string matching of all token lists in $\hat{m}$ and m:

$$acc_m^{sm} = \frac{\text{\# correct } \hat{m}}{\text{\# } \hat{m}}, \quad (15)$$

and string matching of each token pair $\hat{w}_{m_i}$ and w_{m_i} in the token lists:

$$acc_{m\_token}^{sm} = \frac{\text{\# correct } \hat{w}_m}{\text{\# } \hat{w}_m}. \quad (16)$$

To cover token prediction with an abbreviation, we compute the metric-type token accuracy based on the ordered character matching as follows:

$$acc_{m\_token}^{ocm} = \frac{d}{\text{\# } \hat{w}_m}, \quad (17)$$

Model	acc_{hloc}	acc_{hlevel}	acc^{sm}_{m}	$acc^{sm}_{m\_tok}$	$acc^{ocm}_{m\_tok}$
SVM	81.48	82.96	67.41	69.83	69.83
Pointer-Generator Supervised-Att (Glove)	84.44	84.44	68.89	69.57	72.37
Pointer-Generator Supervised-Att (BERT)	67.41	51.11	45.93	33.66	36.61
Pointer-Generator Supervised-Att (SciBERT)	71.11	57.78	44.44	32.51	35.47
Fine-tuned BERT	91.11	90.37	74.81	77.50	80.46
Fine-tuned SciBERT	**93.33**	**93.33**	**79.26**	**81.61**	**85.06**

Table 2: Test accuracies (%) of different metric-type identification models.

Model	acc_{hloc}	acc_{hlevel}	acc^{sm}_{m}	$acc^{sm}_{m\_tok}$	$acc^{ocm}_{m\_tok}$
Pointer-Generator Supervised-Att (Glove)	84.44	84.44	68.89	69.57	72.37
- copy	85.19	84.44	62.22	62.89	65.52
- copy and generation	82.96	82.22	56.30	54.35	56.98

Table 3: Accuracy scores (%) of ablation test of our pointer-generator-based model.

Model	acc_{hloc}	acc_{hlevel}	acc^{sm}_{m}	$acc^{sm}_{m\_tok}$	$acc^{ocm}_{m\_tok}$
Fine-tuned BERT	91.11	90.37	74.81	77.50	80.46
- segment embeddings	87.41	87.41	72.59	75.70	78.00
Fine-tuned SciBERT	93.33	93.33	79.26	81.61	85.06
- segment embeddings	91.85	91.85	76.30	79.31	81.28

Table 4: Test accuracies (%) of ablated BERT-based model without segment embeddings.

		Predicted			
		LRow	LCol	CCapt	Gen
Actual	LRow	0	0	0	0
	LCol	2	77	1	6
	CCapt	1	6	16	3
	Gen	0	5	0	18

Table 5: Confusion matrix of Pointer-Generator Supervised-Att (Glove) prediction.

		Predicted		
		LRow	LCol	Gen
Actual	LRow	0	0	0
	LCol	0	80	6
	Gen	0	3	46

Table 6: Confusion matrix of Fine-tuned SciBERT prediction.

where d is the number of $\hat{w}_m$ whose characters are all found in w_m in the same order. For example, the predicted token *RG1* is regarded as correct when the reference token is *ROUGE-1*.

5.3 Implementation Details

We implemented our models using the AllenNLP library (Gardner et al., 2018). In our pointer-generator-based model, we used pre-trained word embeddings for initialization and two-layer BiLSTMs with 256 hidden sizes in both the caption and header-level encoders. We used dropout (Srivastava and Hovy, 2014) with the probability $p = 0.1$. For optimization in the training phase, we used Adam as the optimizer with a batch size of 10 and a learning rate of 3×10^{-3} and 3×10^{-5} in pointer-generator-based and BERT-based, respectively, with a slanted triangular schedule (Howard and Ruder, 2018). We trained the model for a maximum of 20 epochs with early stopping on the validation set (patience of 10) and set α to 0.5. We used the original BERT and the domain-specific SciBERT uncased model to fine-tune our BERT-based model.

6 Results

6.1 Experimental Results

Model comparison The performances of the proposed and baseline models are shown in Table 2. We can see that the Pointer-Generator Supervised-Attention model initialized by Glove embeddings outperformed the baseline in predicting metric-type location. The accuracy of this model in the metric-type generation part mostly scored better than the baseline. However, the performances dropped significantly when the input

Caption	experimental results in exploring the shared syntactic order event detector
Header	rh level 1: [model, model, model]
	rh level 2: [cl trans mlp, cl trans cnn, cl trans hbrid]
	ch level 1: [pre., rec., f1]
Gold metric-type	[pre., rec., f1]
Predicted metric-type	[pre., rec., f1]
Caption	results on the image and video datasets of sts task (pearson's r × 100)
Header	rh level 1: [method, method, method]
	rh level 2: [sts baseline, pivot, ours]
	ch level 1: [ms-vid (2012), pascal (2014), pascal (2015)]
Gold metric-type	[r, r, r]
Predicted metric-type	[pearson's, pearson's, pearson's]

Table 7: Example of table caption, headers, and predicted metric-type.

was initialized by BERT as well as by SciBERT. BERT and SciBERT embeddings failed to guide our pointer-generator-based model in the metric-type identification task, especially in generating metric-type tokens.

The accuracy of our BERT-based model was significantly better than the others, achieving header-location and header-level prediction accuracy of more than 90% and generation accuracy improvement of more than 7 points (%). The fine-tuned BERT-based model using a domain-specific SciBERT led to significant performance gains in all metrics.

The effect of copy mechanism We evaluated our pointer-generator-based model using an ablation test, as shown in Table 3. As we can see, the performances of our generation model without a copy mechanism decreased. This demonstrates that incorporating the copy mechanism is beneficial in a metric-type token generation. Our model had the worst accuracy when it ran without a pointer-generator network since the location prediction model alone failed to handle out-of-header metric-types.

The effect of segment embeddings Table 4 shows the effect of segment embeddings in our BERT-based model. The accuracies of Fine-tuned BERT and the SciBERT model without segment embeddings both decreased. This means that segment embeddings successfully discriminate header-level boundaries in the input representation of BERT-based models.

6.2 Qualitative Analysis

We analyzed the errors of our pointer-generator-based and fine-tuned SciBERT models by means of the confusion matrices shown in Tables 5 and 6. For better understanding, we simply define our outputs in the matrices as "LRow" for metric-type located in row headers, "LCol" for metric-type located in column headers, "CCapt" for metric-type copied from the caption, and "Gen" for metric-type generated from the vocabulary. The matrix for the fine-tuned SciBERT model does not include the CCapt class since this model does not contain a copy mechanism.

As shown in the Table 5, the most correct classifications were for copying from the header (row and column), while the highest confusions were for copying from the caption and generation from the vocabulary. The accuracy of generating correct metric-type tokens from the vocabulary was 27.78%, and the accuracy of copying a metric-type from the caption was 75%. The copying mechanism contributes to a better performance than generation one.

From the confusion matrix of the SciBERT-based model in Table 6, we can see that the highest confusion was for copying from the header. We also computed the accuracy of generated metric-type tokens and found that just 58.7% of the generated tokens were correct.

We also investigated the errors in the predicted metric-type tokens. We found that the models tended to generate more generic metric-types; for example, they extracted *score* as a prediction for the target *accuracy*. On the other hand, our models generated similar terms to the ground truth metric-type, such as generating the metric-type *pearson's* for the target *r*. The examples are shown in Table 7.

7 Conclusion

In this work, we provided multi-level header numerical table datasets extracted from scientific papers consisting of header tables, captions, and metric-types. We introduced a metric-type identification task for multi-level header numerical tables, and proposed joint location prediction and generation models to solve the task. We have shown that our proposed model can identify metric-types from the multi-level header tables, both when the metric-types are included in the headers and when they are not.

Our datasets only cover scientific papers in the computational linguistic domain. The generalization of our results beyond domain still remains an open question due to the difficulties of collecting comparable datasets in other domains without additional annotation by human experts.

Acknowledgements

Lya Hulliyyatus Suadaa is supported by the Indonesian Endowment Fund for Education (LPDP) and Okumura-Takamura-Funakoshi Laboratory, Tokyo Institute of Technology. This work is partially supported by JST PRESTO (Grant Number JPMJPR1655). We thank the anonymous reviewers for helpful discussion of this work and comments on previous drafts of the paper.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

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

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Marusczyk, and Lukas Lange. 2020. The SOFC-exp corpus and neural approaches to information extraction in the materials science domain. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1255–1268, Online. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, and Debasis Ganguly. 2019. Identification of tasks, datasets, evaluation metrics, and numeric scores for scientific leaderboards construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5203–5213, Florence, Italy. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Matthew Prancis Hurst. 2000. *The interpretation of tables in texts*. Ph.D. thesis, The University of Edinburgh.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102, Osaka, Japan. The COLING 2016 Organizing Committee.

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

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Nikola Milosevic, Cassie Gregson, Robert Hernandez, and Goran Nenadic. 2016. Disentangling the structure of tables in scientific literature. In *Natural Language Processing and Information Systems*, pages 162–174, Cham. Springer International Publishing.

Nikola Milosevic, Cassie Gregson, Robert Hernandez, and Goran Nenadic. 2019. A framework for information extraction from tables in biomedical literature. *International Journal on Document Analysis and Recognition (IJDAR)*, 22(1):55–78.

Armineh Nourbakhsh, Mohammad M. Ghassemi, and Steven Pomerville. 2020. Spread: Automated financial metric extraction and spreading tool from earnings reports. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 853–856. ACM.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Shashank Srivastava and Eduard Hovy. 2014. Vector space semantics with frequency-driven motifs. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 634–643, Baltimore, Maryland. Association for Computational Linguistics.

EmpathBERT: A BERT-based Framework for Demographic-aware Empathy Prediction

Bhanu Prakash Reddy Guda
Adobe Research
guda@adobe.com

Aparna Garimella
Adobe Research
garimell@adobe.com

Niyati Chhaya
Adobe Research
nchhaya@adobe.com

Abstract

Affect preferences vary with user demographics, and tapping into demographic information provides important cues about the users' language preferences. In this paper, we utilize the user demographics, and propose EmpathBERT, a demographic-aware framework for empathy prediction based on BERT. Through several comparative experiments, we show that EmpathBERT surpasses traditional machine learning and deep learning models, and illustrate the importance of user demographics to predict empathy and distress in user responses to stimulative news articles. We also highlight the importance of affect information in the responses by developing affect-aware models to predict user demographic attributes.

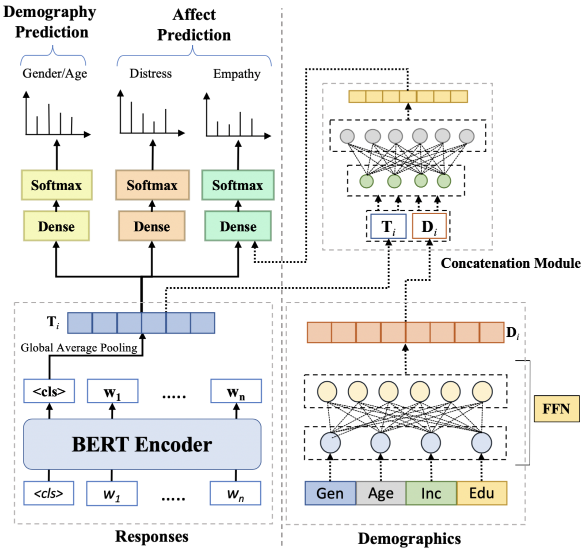


Figure 1: EmpathBERT architecture.

1 Introduction

Modeling complex human reactions and affect from text has been a challenging research area with innovations focusing on sentiment and emotion understanding (Picard, 1997; Li and Liu, 2015; Rosenthal et al., 2017; Socher et al., 2011, 2013). The study of non-trivial human reactions has been limited. These methods, often rooted in psychological theories, have turned out to be more complex in terms of annotation and modeling (Strapparava and Mihalcea, 2007). A critical affective phenomena, *empathy*, has received surprisingly less attention.

Empathy assesses feelings of sympathy towards *others*, and Distress measures anxiety and discomfort oriented towards *self* (Davis, 1980). Empathy has been positively associated to a number of well-being activities, such as volunteering (Batson et al., 1987), charity (Pavey et al., 2012), and longevity (Poulin et al., 2013), and in consumer marketing, advertising and customer interfaces (Wang et al., 2016; Escalas and Stern, 2003). Works on empathy in text have focused on spoken dialogue, addressing conversational agents, psychological interventions, or call center transcripts (McQuiggan and Lester, 2007; Fung et al., 2016; Pérez-Rosas et al., 2017; Alam et al., 2018; Demasi et al., 2019). Buechel et al. (2018) collected an empathy-distress dataset by leveraging users' reactions to textual stimulus content. Sedoc et al. (2019) constructed an empathy lexicon by obtaining word ratings from document-level ratings from this dataset. Xiao et al. (2012); Gibson et al. (2015); Khanpour et al. (2017) presented predictive models for empathy in the healthcare domain. However, we believe none of the above works focus on (a) predicting empathy from textual reactions, and (b) studying the impact of demographics on the expression of empathy.

Language preferences vary with user demographics (Tresselt and Mayzner, 1964; Eckert and McConnell-Ginet, 2013; Garimella et al., 2016; Lin et al., 2018; Loveys et al., 2018), and this has led to studies leveraging the user demographic information to obtain better language representations and classification models for various NLP

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3072–3079
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

tasks (Volkova et al., 2013; Bamman et al., 2014; Hovy, 2015; Garimella et al., 2017). Owing to the recent success of large language models such as BERT (Devlin et al., 2018) and GPT-2 (Radford et al., 2019) in improving the performances of several downstream tasks, we propose a BERT-based demographic-aware framework for empathy (distress) prediction, and through several comparative experiments, show that it surpasses existing baselines and demographic-agnostic approaches.

This paper makes three main contributions. **(1)** We present EMPATHBERT, a demographic-aware empathy (distress) prediction framework, using BERT-based models infused with demographic information. **(2)** Through comparisons against several baseline and demographic-agnostic approaches, we illustrate the importance of user demographics in end-to-end modeling and predicting empathy (distress). **(3)** Conversely, we show that empathy (distress) also contributes to demographic attribute prediction, by developing affect-aware models for demographic attribute prediction, backed by empirical comparison with baselines and generic models. To the best of our knowledge, ours is the first computational effort addressing empathy (distress) through the lens of demographic biases, a phenomenon well-understood in psychology.

2 Dataset

We use the empathy-distress dataset introduced by Buechel et al. (2018). It consists of 418 news articles from popular news platforms, and responses to them from 403 annotators (5 articles each), resulting in a total of 2,015 responses. Filtering the responses that deviated from the task description led to 1,860 responses (empathy: 916, distress: 905) , with a total token count of 173,686 (min: 52, max: 198, median: 84). The number of responses per article ranges from 1 to 7, with an average of 4.46 responses per article. We report some example responses from the dataset in Table 1[1]. We focus on the responses only, and use the empathy (distress) tags associated with these responses. We group the data into binary classes for age (C_0: < 35, C_1: ≥ 35), income (C_0: $\leq \$50,000$, C_1: $> \$50,000$), and education ($C_0$: no degree, C_1: bachelor's or above), to mitigate class imbalances.[2] The resulting dataset is balanced for all dimensions, with a maximum deviation of 5.5% (age) among classes.

3 EMPATHBERT

In this section, we describe our approach for demographic-aware empathy (distress) prediction from text. Figure 1 shows the proposed architecture. Our model takes as input a response (a sequence of words $w_1, w_2, \ldots, w_n$) and demographic information of the corresponding annotator. We represent the response using BERT, a bidirectional Transformer-based (Vaswani et al., 2017) language model. We use the final 768-dimensional hidden vector corresponding to the [CLS] token as the aggregate sequence representation. We employ cross-domain pre-training (Sun et al., 2019), fine-tuning, and multi-task fine-tuning (Liu et al., 2019) techniques to customize BERT for our tasks.

Cross-domain Pre-training (PT). We use the pre-trained BERT language model trained on the English Wikipedia and Book Corpus (Zhu et al., 2015) datasets for masked word and next sentence prediction, and perform further pre-training on demographic-specific datasets to introduce demographic-specific language preferences. This enables slanting the BERT model towards a specific demographic group. For this, we use a corpus different from the empathy dataset in two scenarios. (1) ALL: train the BERT model on all of the external corpus, and (2) DEMOGRAPHIC-SPECIFIC: train only on the demographic-specific samples from the external corpus.

Fine-tuning Only (tBERT). BERT-based fine-tuning has had significant success, due to the ease in implementation and performance gains reported for various NLP tasks (Huang et al., 2019; Liu and Lapata, 2019). We fine-tune BERT for sequence classification by adding a classification layer, where the input is response represented by the hidden vector of the [CLS] token, and output is the prediction for empathy (distress). We train on generic data and demographic-specific portions, and compare the performances to study the demographic effect on empathy (distress) prediction.

Multi-task Fine-tuning (tBERT-MT). We fine-tune BERT in multi-task learning (MTL) setup for classification, similar to (Liu et al., 2019), where the tasks under consideration are empathy (distress) classification and demographic attribute prediction. Both the tasks have shared BERT layers, while the classification heads containing the final dense and

---

[1] Please refer to Buechel et al. (2018) for further details on the dataset.

[2] We do not study race; it has even heavier class-imbalance.

TEXT	DEMOGRAPHIC ATTRIBUTES	SCORE
A 6.4 magnitude on the Richter scale earthquake has shaken up the whole capital of Santiago, Chile. Chile is very propense to earthquakes and natural disasters. We have heard of an earthquake that scaled out to be 8.8 and destroys over 200 thousand homes in Chile. I feel very bad for the people who died. and send out my compassion to the family of the 55 dead in this earthquake.	Female, Age $\geq$ 35, Education $<$ Bachelors, Income $\leq$ \$50,000	0.84
This is just crazy, you have to feel for the mother, but at the same time what kind of apartment has that many violations and is still not punished. They need to sue them and anybody involved with this. I can't believe that in today's society that tragedies like this are tolerated. Somebody needs to go to jail for the death of this little girl and the injuries that her mother suffered. I can't imagine what the mother is going through and she probably blames herself. Things like this should just not happen.	Male, Age $\geq$ 35, Education $\geq$ Bachelors, Income $\leq$ \$50,000	0.82

Table 1: Qualitative examples of high empathy (above) and high distress (below) with scores on empathy and distress dimensions as predicted by our **tBERT-C (fnn)** model.

softmax layers are specific to each task. We replace the final dense and softmax layers in tBERT setup with multiple classification heads based on the number of tasks. We experiment with (1) *Alternative training*: In each epoch, we cyclically train only one classification head, freezing the parameters of the remaining heads; and (2) *Parallel training*: In each epoch, we train the model end-to-end on the joint loss from all the classification heads.

Explicit Demographic Knowledge. PT, tBERT and tBERT-MT intrinsically infuse demographic information. We also incorporate this explicitly by concatenating a *demographic vector* $\vec{d}$ to the output of the global average pooling layer (Lin et al., 2013) from tBERT or tBERT-MT (concatenation in Figure 1) in two ways. (1) **tBERT-[MT]-C**: $\vec{d}$ is a d-dimensional one-hot encoding vector (d: number of demographics). (2) **tBERT-[MT]-C (fnn)**: $\vec{d}$ is the output of a feedforward neural network (FNN), the input for which is a one-hot encoding vector. Three dense layers are stacked before the task-specific heads, and this model is trained end-to-end for empathy (distress) prediction. In tBERT-MT where one of the tasks heads predicts a demographic attribute, the corresponding binary value in $\vec{d}$ is removed. To assess the contribution of specific attributes, we also propose to concatenate a 1-bit encoding (**tBERT-[MT]-C (attribute)**) for each given attribute.

4 Experiments

We model empathy (distress) prediction as a binary classification task. To study the efficacy of empathy (distress) to predict demography attributes, we also conduct experiments for empathy (distress)-aware demographic attribute prediction. Such a prediction can be used for further demographic removal from text to mitigate adversarial attacks and protect privacy of users (Elazar and Goldberg, 2018).

Implementation Details (1) Cross-domain Pretraining: We use the Blog Authorship Corpus[4] (Schler et al., 2006), which consists of 681,288 blogposts and self-provided demographic attributes, gender, age, industry, and astrological sign of the corresponding 19,320 bloggers to further pre-train BERT. Out of these we use the gender attribute to pre-train for male-specific and female-specific pre-training experiments. We train the model on the *Masked Language Model task* (Taylor, 1953) for 10 epochs using a learning rate of 3e-5. (2) Finetuning: We train the model end-to-end (110M parameters) using binary cross-entropy loss and decoupled weight decay Adam optimizer (Loshchilov and Hutter, 2017), in batches of 32. The best performance is observed when the maximum input sequence length is set to 150, learning rate to 3e-5, and number of epochs to 3. (3) Explicit Demographic Attributes: We use gender, age, education and income attributes corresponding to each annotator in the empathy dataset. The d-dimensional vector size 4 resulting in a 16-d FFN output.

Evaluation metrics. We use five-fold cross validation (five random shuffled restarts) with 80-20

---

[3] Statistical significance using McNemar's Test (McNemar, 1947) with * $p < 0.05$, † $p < 0.01$, ‡ $p < 0.001$.

[4] `https://u.cs.biu.ac.il/~koppel/BlogCorpus.htm`

	Method→	PT			tBERT			PT + tBERT		
	Test→	M	F	A_s	M	F	A_s	M	F	A_s
Empathy	Male	**50.02**†	52.42	62.70	**64.73**‡	60.37	62.22	**61.82**†	57.95	58.65
	Female	49.07	**53.12***	48.28	63.70	**64.56**‡	63.32	58.16	**61.77**‡	58.51
	All_s	49.74	52.91	49.64	63.08	62.19	63.00	57.24	58.63	56.30
Distress	Male	**51.21***	52.26	52.41	**64.44**‡	61.56	62.11	**61.92**†	57.60	59.63
	Female	50.71	**52.77***	51.57	61.52	**63.16**‡	60.51	57.35	**59.30**†	60.19
	All_s	49.43	51.42	50.53	63.18	62.77	62.88	59.78	58.77	59.57

Table 2: Accuracies using gender-specific training for empathy (distress) prediction. **M**ale, **F**emale, $A_s$ll denote the respective data subsets. A_s is a sampled dataset with approximately equal number of samples from M and F subsets, hence comparable in size.[3]

	Dem→	Age			Income			Education		
	Test→	C_0	C_1	A_s	C_0	C_1	A_s	C_0	C_1	A_s
Empathy	$Class_0$	**62.79***	62.59	61.44	**62.05**‡	61.82	62.66	59.44	61.18	58.81
	$Class_1$	59.27	**64.95***	60.05	58.40	**64.96**‡	60.41	**62.34**‡	**63.40**‡	61.40
	All_s	59.73	62.05	60.26	60.62	63.21	60.92	60.81	63.03	59.38
Distress	$Class_0$	**62.80**‡	62.32	61.01	**62.46**‡	61.43	60.86	**62.65**‡	62.04	62.30
	$Class_1$	57.20	**68.08**‡	60.68	60.23	**66.59**†	62.39	60.45	**66.85**†	63.31
	All_s	60.89	65.08	61.16	59.92	61.54	60.88	61.80	63.13	62.06

Table 3: Demographic-specific training accuracies for empathy (distress) prediction.

train-test proportions, and report the F1 and accuracy (Ac) averaged across the 5 runs on the test set.
Baselines. We compare our model against the Random Forest (RF) model with Glove embeddings (Pennington et al., 2014) for text and demographic attributes (excluding the prediction attribute) as one-hot vectors as features. We also report performance against deep learning baselines, CNN (Kim, 2014), biLSTM, and biLSTM with Attention (Yang et al., 2016) and the pre-trained BERT without further training.

4.1 Results

Table 2 shows the accuracies using BERT for pretraining (PT), fine-tuning (tBERT), and both (PT + tBERT) for gender-specific empathy (distress) prediction. On the M and F test sets, models trained on the same demographic subset (M or F) outperform those trained on the opposite subset or A_s. The acccuracies of plain BERT are 48.37, 49.49, and 50.42 on A_s, M, and F test sets respectively for empathy prediction. tBERT outperforms all other variants. The results support our hypothesis that empathy is dependent on and influenced by the gender associated with the author. We note similar patterns for age, income, and education (Table 3).

Table 4 shows results for empathy (distress) prediction using tBERT-[MT]-[C (fnn/attribute)] variants trained on the full dataset. In the notation, we replace [MT] with the heads on which the multitasking is performed. For example tBERT-MT-(E+D)-G-C implies fine-tuned BERT with empathy prediction, distress prediction, and gender prediction multi-tasking heads with demographic information concatenated to the text representation directly before classification.[5] We report performances on demographic-wise test sets (A, M, F).
Insights: (1) **tBERT** variants with a single training objective outperform all baselines. (2) Performance of **tBERT-MT** varies with the affect dimension. Empathy prediction shows marginal loss in performance with explicit concatenation (**tBERT-C**) and further loss in the multitask setup. (3) For distress, introduction of gender as the demographic attribute shows an observable improvement across different test sets. (4) A similar trend is observed for age. Table 5 shows performance of age and gender prediction with empathy (distress)-aware models on affect-wise test sets (Empathy (Em) and Distress (Dist)). Empathy-aware gender prediction models show consistent improvement over baselines, with **tBERT (G)** reporting the best score when tested on the complete dataset and empathy-specific test set. **tBERT (A)** helps improve the accuracies for age prediction by atleast 5% over baselines for the com-

---

[5] In the models where a demographic attribute prediction is involved, we remove that attribute from the demographic vector.

	Affect⟶	Empathy						Distress					
	Test Set⟶	All		Male		Female		All		Male		Female	
	Approach↓	F_1	**Ac**	F_1	**Ac**	F_1	**Ac**	F_1	**Ac**	F_1	**Ac**	F_1	**Ac**
Trad.ML	RF-text	57.5	59.9	58.4	60.3	56.4	57.5	58.4	61.0	58.9	60.9	57.9	61.3
	RF-dem	58.99	59.12	58.59	58.64	59.77	59.77	58.06	58.03	60.61	60.70	59.77	59.29
	RF-text+dem	58.5	60.7	57.9	59.7	57.1	59.7	58.4	60.5	59.6	59.9	58.1	61.2
DL Models	CNN	59.5	61.3	60.7	62.1	58.2	60.5	58.8	63.9	57.8	62.5	59.9	63.5
	biLSTM	53.3	55.4	55.4	57.1	50.8	53.5	57.1	59.3	54.3	56.9	60.2	62.1
	biLSTM-Attention	60.8	62.6	60.0	62.0	61.7	63.3	59.9	62.7	59.8	62.3	59.8	63.1
	BERT	65.6	49.0	65.3	48.8	65.9	49.2	66.1	49.5	65.8	49.3	66.3	49.6
Proposed Methods	Aff-biLSTM-text+dem	61.9	63.0	63.0	63.6	60.8	62.3	62.9	64.2	62.9	64.6	62.9	63.7
	tBERT (E)	**67.1**†	**67.8**†	**68.7***	**69.4***	**65.4***	**66.1***	–	–	–	–	–	–
	tBERT (D)	–	–	–	–	–	–	67.6	67.0	69.3	68.6	65.7	65.1
	tBERT-MT-(E+D)	65.2	66.2	66.7	67.6	63.6	64.6	**69.2**‡	**68.5**‡	71.2	70.4	**66.7**†	**66.3**†
	tBERT-MT-G	(E) 63.9	64.9	65.0	66.7	62.8	62.8	(D) 67.0	67.5	70.8	70.3	62.3	64.3
	tBERT-MT-(E+D)-G	64.5	64.7	65.7	66.3	63.1	63.0	68.1	67.7	**71.3***	**70.5***	64.3	64.5
	tBERT-MT-A	(E) 61.8	63.5	65.3	66.7	58.1	60.0	(D) 65.0	65.1	67.6	67.5	62.2	62.5
	tBERT-MT-(E+D)-A	64.1	65.2	65.8	66.8	62.3	63.5	66.0	66.2	69.3	68.8	62.1	63.1
	tBERT-C (fnn)	(E) 66.4	67.4	67.6	68.6	65.0	66.0	67.4	67.4	69.4	69.2	65.0	65.3
	tBERT-C	66.0	66.4	66.8	67.0	65.0	65.8	68.2	67.7	69.9	69.5	66.2	65.6
	tBERT-C (gender)	64.3	66.8	65.0	67.7	63.5	65.9	66.8	66.9	68.6	68.7	64.7	64.9
	tBERT-MT-G-C	(E) 63.8	66.0	65.4	67.6	62.2	64.2	(D) 65.9	67.0	68.6	68.9	62.5	64.9
	tBERT-MT-(E+D)-G-C	(E) 62.2	64.0	64.5	65.7	59.7	62.2	(D) 64.6	66.1	67.5	68.3	61.3	63.6

Table 4: Demographic-aware empathy (distress) prediction. For tBERT-MT, the multitask attributes are specified in the method name i.e. gender (-G), age (-A) along with empathy (E) or distress (D) along side the accuracies. F_1: F1 score; Ac: Accuracy.

Demography ⟶	Gender			Age		
Dataset ⟶	All	Em	Dist	All	Em	Dist
RF-text	59.8	60.8	58.7	56.5	55.7	57.3
RF-text-E/D	58.0	59.1	56.9	56.6	54.2	59.1
Aff-biLSTM(att)-text	59.2	60.1	58.2	56.2	57.7	54.6
Aff-biLSTM(att)-text-E/D	58.9	60.2	57.4	56.9	57.3	56.6
BERT	47.5	47.3	47.7	40.5	41.3	39.6
tBERT	(G) 64.2‡	65.2	63.4	(A) 62.7*	63.2‡	63.8‡
tBERT-MT-E	62.0	61.5	63.3	60.1	61.1	61.9
tBERT-MT-D	61.6	61.7	63.9	60.6	60.8	61.7
tBERT-MT-(E+D)	63.1	62.9	65.1*	61.6	59.8	63.7

Table 5: F1 values of affect-aware demography prediction.

plete (All) test set. For the empathy-specific test set, best results are observed with MTL (**tBERT-MT-(E+D)**). We infer that while having affect-aware demographic prediction models do improve performance over fine-tuned models, they may also lead to a marginally negative impact. The overall inference from above experiments is that demographic-aware models aid affect predictions but the reverse relationship is much weaker. End-to-end training across a variety of train sets and demographic attributes establishes that the variance observed in language preferences and expressions has an impact on the manner of expressing empathy and distress in reactions.

5 Conclusion

We proposed a novel demographic-aware empathy prediction framework based on fine-tuning and multi-tasking using BERT, showed that it surpasses existing methods, and illustrated the impact of demography in modeling subjective phenomena such as empathy and distress. Our framework is generalizable, and we extended it to empathy-aware demography prediction, and showed that empathy also improves demographic prediction. We believe this is a significant checkpoint towards developing models for empathy (distress), and tapping into demographic information while doing so.

References

Firoj Alam, Morena Danieli, and Giuseppe Riccardi. 2018. Annotating and modeling empathy in spoken conversations. *Computer Speech & Language*, 50:40–61.

David Bamman, Chris Dyer, and Noah A. Smith. 2014. Distributed representations of geographically situated language. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 828–834, Baltimore, Maryland. Association for Computational Linguistics.

C Daniel Batson, Jim Fultz, and Patricia A Schoenrade. 1987. Distress and empathy: Two qualitatively distinct vicarious emotions with different motivational consequences. *Journal of personality*, 55(1):19–39.

Sven Buechel, Anneke Buffone, Barry Slaff, Lyle Ungar, and João Sedoc. 2018. Modeling empathy and distress in reaction to news stories. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4758–4765, Brussels, Belgium. Association for Computational Linguistics.

Mark H Davis. 1980. Interpersonal reactivity index (iri). *A multidimensional approach to individual differences in empathy. JSAS Catalog of Selected Documents in Psychology*, 10:85.

Orianna Demasi, Marti Hearst, and Benjamin Recht. 2019. Towards augmenting crisis counselor training by improving message retrieval. pages 1–11.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Penelope Eckert and Sally McConnell-Ginet. 2013. *Language and gender*. Cambridge University Press.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, Brussels, Belgium. Association for Computational Linguistics.

Jennifer Edson Escalas and Barbara B Stern. 2003. Sympathy and empathy: Emotional responses to advertising dramas. *Journal of Consumer Research*, 29(4):566–578.

Pascale Fung, Anik Dey, Farhad Bin Siddique, Ruixi Lin, Yang Yang, Yan Wan, and Ho Yin Ricky Chan. 2016. Zara the Supergirl: An empathetic personality recognition system. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 87–91, San Diego, California. Association for Computational Linguistics.

Aparna Garimella, Carmen Banea, and Rada Mihalcea. 2017. Demographic-aware word associations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2285–2295, Copenhagen, Denmark. Association for Computational Linguistics.

Aparna Garimella, Rada Mihalcea, and James Pennebaker. 2016. Identifying cross-cultural differences in word usage. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 674–683, Osaka, Japan. The COLING 2016 Organizing Committee.

James Gibson, Nikolaos Malandrakis, Francisco Romero, David C Atkins, and Shrikanth S Narayanan. 2015. Predicting therapist empathy in motivational interviews using language features inspired by psycholinguistic norms. In *Sixteenth annual conference of the international speech communication association*.

Dirk Hovy. 2015. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 752–762, Beijing, China. Association for Computational Linguistics.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. *arXiv preprint arXiv:1908.07245*.

Hamed Khanpour, Cornelia Caragea, and Prakhar Biyani. 2017. Identifying empathetic messages in online health communities. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 246–251, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Chen Li and Yang Liu. 2015. Improving named entity recognition in tweets via detecting non-standard words. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 929–938, Beijing, China. Association for Computational Linguistics.

Bill Yuchen Lin, Frank F. Xu, Kenny Zhu, and Seungwon Hwang. 2018. Mining cross-cultural differences and similarities in social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 709–719, Melbourne, Australia. Association for Computational Linguistics.

Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400*.

Zhaojiang Lin, Andrea Madotto, Jamin Shin, Peng Xu, and Pascale Fung. 2019. MoEL: Mixture of empathetic listeners. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 121–132, Hong Kong, China. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Kate Loveys, Jonathan Torrez, Alex Fine, Glen Moriarty, and Glen Coppersmith. 2018. Cross-cultural differences in language markers of depression online. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 78–87, New Orleans, LA. Association for Computational Linguistics.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Scott W McQuiggan and James C Lester. 2007. Modeling and evaluating empathy in embodied companion agents. *International Journal of Human-Computer Studies*, 65(4):348–360.

Louisa Pavey, Tobias Greitemeyer, and Paul Sparks. 2012. “i help because i want to, not because you tell me to” empathy increases autonomously motivated helping. *Personality and Social Psychology Bulletin*, 38(5):681–689.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Verónica Pérez-Rosas, Rada Mihalcea, Kenneth Resnicow, Satinder Singh, and Lawrence An. 2017. Understanding and predicting empathic behavior in counseling therapy. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1435, Vancouver, Canada. Association for Computational Linguistics.

Rosalind W. Picard. 1997. *Affective Computing*. MIT Press, Cambridge, MA, USA.

Michael J Poulin, Stephanie L Brown, Amanda J Dillard, and Dylan M Smith. 2013. Giving to others and the association between stress and mortality. *American journal of public health*, 103(9):1649–1655.

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

Hannah Rashkin, Eric Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205.

João Sedoc, Sven Buechel, Yehonathan Nachmany, Anneke Buffone, and Lyle Ungar. 2019. Learning word ratings for empathy and distress from document-level user responses. *arXiv preprint arXiv:1912.01079*.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Carlo Strapparava and Rada Mihalcea. 2007. SemEval-2007 task 14: Affective text. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 70–74, Prague, Czech Republic. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Wilson L Taylor. 1953. “cloze procedure”: A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Margaret E Tresselt and Mark S Mayzner. 1964. The kent-rosanoff word association: Word association norms as a function of age. *Psychonomic Science*, 1(1-12):65–66.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Svitlana Volkova, Theresa Wilson, and David Yarowsky. 2013. Exploring sentiment in social media: Bootstrapping subjectivity clues from multilingual twitter streams. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 505–510, Sofia, Bulgaria. Association for Computational Linguistics.

Chen Wang, Rui Juliet Zhu, and Todd C Handy. 2016. Experiencing haptic roughness promotes empathy. *Journal of Consumer Psychology*, 26(3):350–362.

Bo Xiao, Dogan Can, Panayiotis G Georgiou, David Atkins, and Shrikanth S Narayanan. 2012. Analyzing the language of therapist empathy in motivational interview based psychotherapy. In *Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1–4. IEEE.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 1480–1489.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Are Neural Networks Extracting Linguistic Properties or Memorizing Training Data? An Observation with a Multilingual Probe for Predicting Tense

Bingzhi Li and **Guillaume Wisniewski**
Université de Paris, LLF, CNRS
75 013 Paris, France
`bingzhi.li@etu.u-paris.fr`
`guillaume.wisniewski@u-paris.fr`

Abstract

We evaluate the ability of BERT embeddings to represent tense information, taking French and Chinese as a case study. In French, the tense information is expressed by verb morphology and can be captured by simple surface information. On the contrary, tense interpretation in Chinese is driven by abstract, lexical, syntactic and even pragmatic information. We show that while French tenses can easily be predicted from sentence representations, results drop sharply for Chinese, which suggests that BERT is more likely to memorize shallow patterns from the training data rather than uncover abstract properties.

1 Introduction

The success of deep learning in many NLP tasks is often attributed to the ability of neural networks to learn, without any supervision, relevant linguistic representations. Many works have tried to identify which linguistic properties are encoded in the words and phrases embeddings uncovered during training. For instance, Belinkov et al. (2017) and Peters et al. (2018) studies the capacity of neural networks to uncover morphological information and Linzen et al. (2016), Tenney et al. (2019) or Hewitt and Manning (2019) (among many other) syntactic information. These works are based on the definition and study of *linguistic probes*: a *probe* (Alain and Bengio, 2017) is trained to predict linguistic properties from the representation of language; achieving high accuracy at this task implies these properties were encoded in the representation.

However, as pointed out by Hewitt and Manning (2019), these approaches suffer from a major drawback: there is no guarantee that the probes' good performances result from the ability of the representations to capture relevant linguistic properties rather than to memorize a large number of labeling decisions. Indeed, in most of the tasks considered so far, labels could be deduced directly from surface information, namely the word form (its morphology) and the word position in the sentence. Given the huge number of parameters of current models, there is a high risk that they are only able to extract and memorize lexically-marked patterns from training data with low (if any) generalization power.

To shed new light on this question, we consider, in this work, a multilingual linguistic probe, the goal of which is to predict the tense of a sentence (i.e. the location in time of an utterance). We compare the performance of this probe on two languages, French, which expresses tense by the verb morphology, and Chinese, in which, as explained in §2, the tense is expressed by a combination of lexical, syntactic and pragmatic cues. If intuitively the tense can be predicted from simple surface patterns in French, predicting the tense of a Chinese sentence requires capturing the interaction of all sentence-level factors related to time, and sometimes even the contextual information from the previous utterances. Contrasting the performance achieved by the probe on several languages ensures that the linguistic properties we detect are actually captured by the representation learned by BERT and not by the probe, and thus to avoid a common pitfall of this kind of approaches (Belinkov and Glass, 2019; Barrett et al., 2019).

This work has two main contributions: first, we highlight the interest of contrasting linguistic probes on different languages; second, our experiments (§3-4) show that BERT has a preference for learning lexically marked features

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3080–3089
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

over lexically unmarked features and, consequently, is not able to extract an abstract representation that can be applied to (groups of) words that have not been seen at training time.

2 Tense and Aspect

Languages can roughly be classified under two categories: tense language and aspect language, depending on how they denote time relations (Xiao and McEnery, 2004). Tense indicates the temporal location of a situation, while aspect, according to Comrie (1998), refers to "the different ways of viewing the internal temporal constituency of a situation" and denotes how speakers view it in terms of its duration, frequency and whether or not it is completed. In tense language, like English or French, the tense and aspect are often encoded simultaneously in verb morphology. For example, the simple past in English locates a situation before the speech time and is often indicated by the *-ed* inflection(e.g. *worked*). Similarly, the French past tense *imparfait* is marked by the *-ait* inflection in *il travaillait* (he worked). Contrary to these two tense languages, Mandarin Chinese does not have a grammatical category of tense (Smith, 1997) and the verb morphology never changes. Figure 1 presents five sentences with the Chinese verb 加班 (*jiaban*, work overtime): while these sentences have different tenses (simple past, present progressive, habitual present, past progressive and future), the form of the verb is always the same.

The notion of tense in Mandarin Chinese is lexicalized and is often indicated by content words like adverbs of time; aspectual information is systematically conveyed by aspect markers. As an aspect language, the temporal interpretation of a verb is tightly related to the notion of *aspect*. For instance, as noticed by Lin (2006), a verb marked by a perfective aspect particle such as 了 *(le)* often gets a past interpretation, such as in the first sentence of Figure 1. And imperfective aspect privileges a present interpretation: in the example 1.2, the same verb is marked by the imperfective aspect particle 在 *(zai)*, which explains why the sentence gets a present interpretation.

However, according to the genre of the text, only 2% to 12% of verbs in Chinese have aspect markers (McEnery and Xiao, 2010)[1] and the tenses should often be inferred from contextual cues like lexical and syntactic features when there is no explicit aspect marker(Saillard, 2014). For instance, in the absence of aspect marker in sentence 1.3, the adverb 常常 (*changchang*, often) leads to a habitual present interpretation. These contextual cues can even invalidate the default correlation of time and aspect we have previously described, as in example 1.4, in which the verb group gets a past interpretation even if it is marked by an imperfective aspect particle because of the past temporal context introduced by the adverbial clause. Finally, in example 1.5, the modal auxiliary 会 *(hui)* and temporal expression 晚上 (*wanshang*, tonight/in the evening) lead to a future-tense interpretation.

Thus, unlike French and English, the time of a Chinese sentence can only rarely be deduced from a surface analysis of the sentence (i.e. from the characters composing its words) and in order to determine the tense, it is necessary to take into account both syntactic and even pragmatic information.

3 Creating a Corpus Annotated with Tense Information[2]

The tense prediction task we consider in this work requires corpora in which the tense of each verb is identified. To the best of our knowledge, there is no such publicly available corpus. For languages such as French or English, in which tenses are described by verb morphology, it is possible to easily build such corpus from morpho-syntactically labeled treebanks (e.g. from the UD project (Zeman et al., 2020)). However, this approach cannot be readily generalized to languages such as Chinese, in which, time is not explicitly marked.

We propose to leverage parallel French-Chinese corpora to obtain tense annotations for Chinese sentences automatically. Our approach relies on two hypotheses. First, we assume that the tense of a translated sentence (target sentence) is the same as the tense of

---

[1] Aspect markers occur more frequently in narrative texts than in expository texts

[2] **The code of all our experiments as well as the corpora we used in this work can be downloaded from `https://github.com/bingzhilee/tense_Representation_Bert`.**

(1)	他	加班	了。
	Ta	jiaban	le
	Pron.3sg	work_overtime	Pfv

He worked overtime.

(2)	他	在	加班。
	Ta	zai	jiaban
	Pron.3sg	Ipfv	work_overtime

He is working overtime.

(3)	他	常常	加班。
	Ta	changchang	jiaban
	Pron.3sg	often	work_overtime

He often works overtime.

(4)	我	去	找	他	时，	他	在	加班。
	Wo	qu	zhao	ta	shi,	ta	zai	jiaban
	Pron.1sg	go	find	Pron.3sg	time,	Pron.3sg	Ipfv	work_overtime

When I went to see him, he was working overtime.

(5)	晚上	他	会	不	会	还	在	加班?
	Wanshang	ta	hui	bu	hui	hai	zai	jiaban
	Tonight	Pron.3sg	MOD	NEG	MOD	still	Ipfv	work_overtime

Will he still work overtime tonight?

Figure 1: Examples of different ways to express the tense in Chinese: tenses are indicated by both aspect markers and a combination of lexical and syntactic cues and not by the verb morphology as in French or English. Ipfv describes one of the two imperfective aspect markers and Pfv one of the two perfective aspect markers

its original (source) sentence ignoring translationese effects (Toury, 2012; Baker et al., 1993). Second, we decided to associate each sentence with the tense of its main clause and not label each verb tense. This assumption allows us to mitigate errors related to the verbal structures identifications and misalignments between Chinese and French verbs as labels are defined at the sentence level.

We considered, in this work, the French-Chinese NewsCommentary[3] parallel corpus (Barrault et al., 2019). To extract tense information, we use the `Stanza` pipeline (Qi et al., 2020) with its pretrained French model to find the root of each sentence and extract its tense and its mode from its (automatic) morphological analysis. We also use the dependency analysis to identify periphrastic catena expressing future (*aller* + Infinitive) or past (*venir de* + Infinitive). With these information, we can define the tense of each sentence easily by mapping the tense of the root verb to one of the three labels Present, Past or Future[4] and the tense of a Chinese sentence is defined as the tense of its French translation.

We evaluate our tense extraction procedure on the Pud corpus[5]. It appears that `Stanza` is able to correctly predict the tense of 95% of the sentences (i.e. identify the root of the sentence and correctly predict its morphological information). Therefore, we consider that the tense labels are predicted with sufficient quality to measure a model's ability to capture time information. However, most of the prediction errors are due to the same construction: the auxiliary *être* followed by the past participle of the verb, that can be used to form either the passive voice or the *passé composé* tense of a verb. As a result, most of our corpus' passive sentence is labeled as Past while they are at the present tense.

In the end, this procedure results in a corpus of 4,764 documents containing 174,347 Chinese and French sentences annotated with tense information. As expected, most of the sentences are in the present tense and the corpus is highly unbalanced: 67% of the examples are labeled Present, 27% Past and only 6% Future. Our corpus also confirms the observations reported in section 1 on the limited use of temporal markers in Chinese: only 16% of the sentences have an explicit temporal marker.[6] We consider 80% of the data for

---

[3] We consider the 15th version of the corpus as, according to our preliminary experiments, most other parallel corpora (e.g. OpenSubtitles) contains almost exclusively sentences in the present tense.

[4] This mapping is more precisely defined in Appendix A Table 2

[5] Pud is a UD corpus that has not been used to train the `Stanza` models.

[6] More precisely: 75% of sentences in the past, 88% of sentences in the present and 85% of sentences in the

	Chinese		French	
	micro prec.	macro prec.	micro prec.	macro prec.
featSVM	73%	60%	–	–
BertSVM	71%	57%	82%	75%
fineBert	77%	68%	94%	95%

Table 1: Results achieved by our different models in the tense prediction task.

training, 10% for testing and 10% for the validation set.

4 Experimental Results

Models The task of tense prediction consists in finding, given a representation $\mathbf{x} \in \mathbb{R}^n$ of a sentence, a label describing its tense (see §3 for a definition of these labels). We consider three models. The first one, denoted featSVM, uses a SVM and a set of handcrafted features designed to capture the information identified as relevant for determining the tense of a Chinese sentence. We rely on the theoretical study of Smith and Erbaugh (2005) to define the features:[7] indicators to describe the presence of aspectual markers (e.g. 了 or 过) or temporal adverbs (e.g. 昨天 *(yesterday)* or 明天 *(tomorrow)*), the sentence root verb, modal auxiliaries (e.g. 会 *(will (probably))*, ...

Our second model, denoted BertSVM is a simple SVM that uses, as sole sentence representation, the embeddings generated by pretrained multilingual Bert. We used the second-to-last hidden layer of all tokens in the sentence and did average pooling. The embeddings on [CLS] and [SEP] were masked to zero before pooling, so they are not included(Xiao, 2018). These representations are kept unchanged. Finally, we consider fineBert a neural network in which Bert pretrained language representation are fine-tuned on the tense prediction task. More precisely, we stack a *softmax* layer on top of the pre-trained Bert model and estimate the weights of this layer while updating Bert parameters by minimizing the cross-entropy loss.

We used Google's pre-trained Multilingual cased Bert in our experiments. It was trained on the entire Wikipedia dump for each language. The French and Chinese training sets have comparable sizes. The performance of mBert on XNLI cross-lingual classification is similar for these two languages: 76.9 for French and 76.6 for Chinese (Martin et al., 2019; Devlin et al., 2018).

In our experiments, we used the SVM implementation provided by the sklearn library (Pedregosa et al., 2018) and TensorFlow in our fine-tuning experiment. Hyperparameters of the SVM have been chosen by 5-folds cross-validation.

Results We evaluate the results of the tense prediction task using both micro and macro precision to account for the imbalance between classes.[8] Results are reported in Table 1.[9] As expected, the best results both for French and Chinese are achieved by fineBert, the model in which the word and sentence representations are tailored to the tense prediction task. The relatively good performance of the featSVM shows the relevance of the considered features and validates the theoretical analysis of Smith and Erbaugh (2005). It also highlights the difficulty of defining handcrafted features generic enough to capture time information in all conditions.

Comparing performances achieved on Chinese and French is particularly interesting since it shows that our very simple architecture is able to predict almost perfectly the tense of French sentences (which can be deduced directly from the morphology of the verb and therefore from a surface analysis) but that its performance drops drastically when applied to Chinese sentences, the tense of which has to be inferred from a wide array of both lexical and syntactic cues. This observation suggests that the model is only memorizing patterns from

---

future have no explicit markings

[7] See Appendix B for a full description of the features considered.

[8] The macro-precision calculates the precision independently for each class, then takes the average (so all classes are treated equally), while the micro-precision aggregates the all classes' contributions to calculate the average metric.

[9] Table 3 in Appendix C provides the precision for each class.

the training set rather than inferring a meaningful representation of the sentence.

To corroborate this interpretation, we have evaluated the performance of fineBert in terms of the similarity between the test and train data: for each language, we have trained a 5-gram language model with Kneser-Ney smoothing using `KenLM` (Heafield et al., 2013) and divided the test set sentences into 3 groups of equal size according to the probability that the sentence was generated by the language model.[10] Because of the way tense is expressed in Chinese, ensuring that the verb of the test sentences are not in the train set is not enough. It appears, as expected, that performance drops significantly when the similarity of train and test sentences decreases: for Chinese (resp. French), the macro precision drops from 70% (resp. 96%) for the test phrases that are the most similar to the train set to 66% (resp. 93%) for the test phrases that are the most different from the train set, while their similarity with the train set (measured as the mean of $\log_{10} p(x)$ over the test test) drops from -30.62 to -93.93 (resp. -70.4 to -231.18). Detailed results are presented in Appendix D.

These results clearly show that the higher the similarity with the train sentences is, the more accurate the model is. Again, this observation questions the capacity of the model to capture relevant linguistic properties rather than simply memorizing the training data.

Discussion Our experiments clearly show that Bert prefers learning lexically marked features rather than lexically unmarked features. These results indicate that, even if several confounders exist, neural networks tend to memorize shallow patterns from the training data rather than uncover abstract linguistic properties.

There is a first well-known confounding factor when interpreting probing results: high classification accuracy could result from the fact that the probe has learned the linguistic task and not from the properties captured by the representation. In our work, we avoid this pitfall by considering a multilingual probe setting: our conclusions are not based on an absolute score but on the comparison of the performance achieved in French and Chinese by the same probe.

The difference in performance between the French and Chinese models is a second confounder. This difference can result from either the training set size or the model's architecture tailored to extract only lexically-marked information. In recent work, Warstadt et al. (2020) suggests that it may be possible that, if more data were available, Bert could eventually learn to predict Chinese tense. However,it must be pointed out that the French model achieves a precision of 76.9% and the Chinese model a precision of 76.6% on the XLNI cross-lingual classification task, the standard evaluation benchmark of sentence representation models. Therefore, we believe that our conclusions are not biased by the language modeling performance.

There is a third and last possible confounder: it is possible that, as explained in §2, sometimes, in Chinese the cues to tense may appear in an earlier sentence. Gong et al. (2012) showed that the tense of previous sentences has a close relation to the current sentence. Considering contextual features in our feature-based SVM classifier only increases the accuracy by 2%, therefore, we believe that this factor has only a moderate impact.

5 Conclusion

We have shown that the performance of a tense prediction model varies dramatically depending on how the language expresses time, a result that suggests that Bert is more likely to memorize shallow patterns from the train set rather than uncover abstract properties. Our work also highlights the interest of comparing linguistic probes across languages which opens up a new avenue for research.

Acknowledgments

We sincerely thank the reviewers and Program Chairs for their careful reviews and insightful comments, which are of great help in improving the manuscript. We would like to express special gratitude to Professor Claire Saillard, this work would not have been possible without her helpful comments and advice.

---

[10] More precisely, we have ordered the sentences of the test set according to their probability estimated by the language model and considered the 5,814 first (resp. last) sentences as the most different (resp. similar) from the train set.

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Mona Baker, Gill Francis, and Elena Tognini-Bonelli. 1993. *Text and Technology: In honour of John Sinclair*. John Benjamins Publishing. Google-Books-ID: dTlIAAAAQBAJ.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. 2019. Adversarial removal of demographic attributes revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6330–6335, Hong Kong, China. Association for Computational Linguistics.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Bernard Comrie. 1998. *Aspect: an introduction to the study of verbal aspect and related problems*, reprinted edition. Number 2 in Cambridge textbooks in linguistics. Cambridge Univ. Press, Cambridge. OCLC: 247564008.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Zhengxian Gong, Min Zhang, Chew Lim Tan, and Guodong Zhou. 2012. N-gram-based tense models for statistical machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 276–285, Jeju Island, Korea. Association for Computational Linguistics.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Jo-Wang Lin. 2006. Time in a Language Without Tense: The Case of Chinese. *Journal of Semantics*, 23(1):1–53.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Louis Martin, Benjamin Müller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. Camembert: a tasty french language model. *CoRR*, abs/1911.03894.

Tony McEnery and Richard Xiao. 2010. *Corpus-based contrastive studies of English and Chinese*. Routledge, New York. OCLC: 1086452482.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2018. Scikit-learn: Machine Learning in Python. *arXiv:1201.0490 [cs]*. ArXiv: 1201.0490.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *arXiv:2003.07082 [cs]*. ArXiv: 2003.07082.

Claire Saillard. 2014. De l'aspect au temps. Expression de la temporalité et de l'aspectualité en français L2 par des apprenants sinophones . In *The expression of temporality by L2 learners of French and English*, Montpellier : Université Paul Valéry, Unknown Region.

Carlota S Smith. 1997. *The parameter of aspect*. Kluwer, Dordrecht; Boston. OCLC: 1012457859.

Carlota S. Smith and Mary S. Erbaugh. 2005. Temporal interpretation in Mandarin Chinese. *Linguistics*, 43(4). Number: 4.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Gideon Toury. 2012. *Descriptive Translation Studies –and beyond: Revised edition*. John Benjamins Publishing. Google-Books-ID: jX4weQT63rIC.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Han Xiao. 2018. bert-as-service. `https://github.com/hanxiao/bert-as-service`.

Richard Xiao and Tony McEnery. 2004. *Aspect in Mandarin Chinese: a corpus-based study*. Number v. 73 in Studies in language companion series. J. Benjamins Pub, Amsterdam ; Philadelphia. OCLC: ocm56526702.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, Elia Ackermann, Noëmi Aepli, Željko Agić, Lars Ahrenberg, Chika Kennedy Ajede, Gabrielė Aleksandravičiūtė, Lene Antonsen, Katya Aplonova, Angelina Aquino, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Furkan Atmaca, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, Colin Batchelor, John Bauer, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Agnė Bielinskienė, Rogier Blokland, Victoria Bobicev, Loïc Boizou, Emanuel Borges Völker, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Kristina Brokaitė, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Tatiana Cavalcanti, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Ethan Chi, Jinho Choi, Yongseok Cho, Jayeol Chun, Alessandra T. Cignarella, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Elvis de Souza, Arantza Diaz de Ilarraza, Carly Dickerson, Bamba Dione, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Hanne Eckhoff, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Olga Erina, Tomaž Erjavec, Aline Etienne, Wograine Evelyn, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Kazunori Fujita, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Bernadeta Griciūtė, Matias Grioni, Loïc Grobol, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Tunga Güngör, Nizar Habash, Jan Hajič, Jan Hajič jr., Mika Hämäläinen, Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Johannes Heinecke, Oliver Hellwig, Felix Hennig, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Takumi Ikeda, Radu Ion, Elena Irimia, Ọlájídé Ishola, Tomáš Jelínek, Anders Johannsen, Hildur Jónsdóttir, Fredrik Jørgensen, Markus Juutinen, Hüner Kaşıkara, Andre Kaasen, Nadezhda Kabaeva, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Elena Klementieva, Arne Köhn, Abdullatif Köksal, Kamil Kopacewicz, Timo Korkiakangas, Natalia Kotsyba, Jolanta Kovalevskaitė, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phương Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Maria Levina, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Yuan Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Hiroshi Matsuda, Yuji Matsumoto, Ryan McDonald, Sarah McGuinness, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin

Mititelu, Maria Mitrofan, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Tomohiko Morioka, Shinsuke Mori, Shigeki Moro, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Robert Munro, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lương Nguyễn Thị, Huyền Nguyễn Thị Minh, Yoshihiro Nikaido, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Atul Kr. Ojha, Adédayọ̀ Olúòkun, Mai Omura, Emeka Onwuegbuzia, Petya Osenova, Robert Östling, Lilja Øvrelid, Şaziye Betül Özateş, Arzucan Özgür, Balkız Öztürk Başaran, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Angelika Peljak-Łapińska, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Daria Petrova, Slav Petrov, Jason Phelan, Jussi Piitulainen, Tommi A Pirinen, Emily Pitler, Barbara Plank, Thierry Poibeau, Larisa Ponomareva, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Peng Qi, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Petru Rebeja, Siva Reddy, Georg Rehm, Ivan Riabov, Michael Rießler, Erika Rimkutė, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Valentin Roșca, Davide Rovati, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Alessio Salomoni, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Dage Särg, Baiba Saulīte, Yanin Sawanakunanon, Salvatore Scarlata, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Hiroyuki Shirasu, Muh Shohibussirri, Dmitry Sichinava, Aline Silveira, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Maria Skachedubova, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Shingo Suzuki, Zsolt Szántó, Dima Taji, Yuta Takahashi, Fabio Tamburini, Takaaki Tanaka, Samson Tella, Isabelle Tellier, Guillaume Thomas, Liisi Torga, Marsida Toska, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Utku Türk, Francis Tyers, Sumire Uematsu, Roman Untilov, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Andrius Utka, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Aya Wakasa, Lars Wallin, Abigail Walsh, Jing Xian Wang, Jonathan North Washington, Maximilan Wendt, Paul Widmer, Seyi Williams, Mats Wirén, Christian Wittern, Tsegay Woldemariam, Tak-sum Wong, Alina Wróblewska, Mary Yako, Kayo Yamashita, Naoki Yamazaki, Chunxiao Yan, Koichi Yasuoka, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Hanzhi Zhu, and Anna Zhuravleva. 2020. Universal dependencies 2.6. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

A Mapping of French Tense

When building our corpus, the tense of each sentence was deduced from the automatic morphological analysis of its root verb using the mapping defined in Table 2.

French passé composé describes a situation that was completed in the past and emphasizes its results in the present. Smith (1997)considers that the Passé composé presents two tense values. When used to present a given state of things, passé composé is temporally present. When used to denote past facts, passé composé, called a preterit by Smith, is temporally past. Which tense value of passé composé should we take into account, the perfect present or past? Concerning the translation of perfect present into Chinese, it’ s interesting to read the corpus study of Xiao and McEnery (2004), according to which the perfect present tense of English is most frequently (71%) translated into Chinese by perfective aspect. Whereas in Chinese, Lin (2006) contends a preferential correlation between perfective aspect and past tense. We have thus decided to classify the passé composé into the Past category.

B Features used in featSVM

1. **Root verb**: The root verb may denote some intrinsic features related to tense(Xiao and McEnery, 2004) In the automatic morphological analysis generated by Stanza, the root word is the verb with a dependency label of *root*, or the Chinese adjectives (Chinese adjectives can be used as verbs) governed by root directly.

2. **Aspect markers**: Perfective aspect marker (了 le, 过 guo) and the imperfective aspect marker (着 zhe).We didn’t consider another imperfective aspect marker (在 zai) because `Stanza` didn’t annotate this marker.

3. **Temporal nouns**: We have extracted a list of temporal nouns. These words have been annotated by `Stanza` with the dependency label *nmod:tmod*. A complex sentence could contain multiple *nmod:tmod* words. We only consider the one that is governed by its root verb or by the verb governed directly by the root word. This list mainly contains words like 现在 *(now)*, 明天 *(tomorrow)*, 刚才 *(just now)*.

4. **Temporal adverbs**: We have determined a list of adverbs with temporal connotation. For example, 已经 *(already)*, 一直 *(always)*, 曾 *(once)*. These adverbs have been annotated by `Stanza` with the dependency label *advmod*. Like the temporal nouns, we only take into account the temporal adverb directly governed by the root word.

5. **Modal auxiliaries**: Words that express necessity, expectation, possibility of the action described by the verb. The bounded situation in the future in Chinese is often expressed by modal auxiliaries 要 *(is going to)* or 会 *(it is probable that)*.

6. **Contextual tense**: We consider the tense of the previous sentence as contextual tense, which provides contextual temporal cues for some Chinese sentences that have no temporal words or aspect markers at all.

7. **Words and POS patterns**: Combinations of word and POS tag for each word in the whole sentence. These features are expected to capture some special syntactic structure. For example, the structure 就要 + predicate + 了 *(...is going to happen)* indicates a near-future situation.

C Performance of different models for each class

Table 3 presents the results of different classifiers for each tense category. It shows that more frequent data are more likely to be better predicted: the Present class (67% of the examples) gets the highest score for all classifiers except French FINEBERT.

D Results of the performance of FineBert

Table 4 details the performance of FINEBERT and the impact of the similarity between the train and test sentences. The similarity is measured as the mean of $\log_{10} p(x)$ over the test test.

Tense predicted for the root verb	Label of the sentence
présent de l'indicatif	PRESENT
présent du conditionnel	PRESENT
présent de l'impératif	PRESENT
imparfait	PAST
plus-que-parfait	PAST
passé simple	PAST
passé récent	PAST
passé composé	PAST
futur simple	FUTURE
futur proche	FUTURE

Table 2: Mapping between French tenses and tense labels used to build our corpus.

	Chinese			French		
	Future	Past	Present	Future	Past	Present
FEATSVM	36%	63%	81%	-	-	-
BERTSVM	31%	60%	79%	64%	73%	88%
FINEBERT	51%	71%	82%	95%	97%	94%

Table 3: Precision achieved by the different classifiers for each class in tense prediction tasks

	Chinese			French		
	sim.	micro prec.	macro prec.	sim.	micro prec.	macro prec.
SUBGROUP1	-93.93	75%	66%	-231.18	92%	93%
SUBGROUP2	-55.92	77%	68%	-135.41	94%	94%
SUBGROUP3	-30.62	79%	70%	-70.74	96%	96%

Table 4: Results based on the similarity of test sentences to the train set

Is Supervised Syntactic Parsing Beneficial for Language Understanding Tasks? An Empirical Investigation

Goran Glavaš
University of Mannheim
Data and Web Science Group
`goran@informatik.uni-mannheim.de`

Ivan Vulić
University of Cambridge
Language Technology Lab
`iv250@cam.ac.uk`

Abstract

Traditional NLP has long held (supervised) syntactic parsing necessary for successful higher-level semantic language understanding (LU). The recent advent of end-to-end neural models, self-supervised via language modeling (LM), and their success on a wide range of LU tasks, however, questions this belief. In this work, we empirically investigate the usefulness of supervised parsing for semantic LU in the context of LM-pretrained transformer networks. Relying on the established fine-tuning paradigm, we first couple a pretrained transformer with a biaffine parsing head, aiming to infuse explicit syntactic knowledge from Universal Dependencies treebanks into the transformer. We then fine-tune the model for LU tasks and measure the effect of the intermediate parsing training (IPT) on downstream LU task performance. Results from both monolingual English and zero-shot language transfer experiments (with intermediate target-language parsing) show that explicit formalized syntax, injected into transformers through IPT, has very limited and inconsistent effect on downstream LU performance. Our results, coupled with our analysis of transformers' representation spaces before and after intermediate parsing, make a significant step towards providing answers to an essential question: how (un)availing is supervised parsing for high-level semantic natural language understanding in the era of large neural models?

1 Introduction

Structural analysis of sentences, based on a variety of syntactic formalisms (Charniak, 1996; Taylor et al., 2003; De Marneffe et al., 2006; Hockenmaier and Steedman, 2007; Nivre et al., 2016, 2020, *inter alia*), has been the beating heart of NLP pipelines for decades (Klein and Manning, 2003; Chen and Manning, 2014; Dozat and Manning, 2017; Kondratyuk and Straka, 2019), establishing rather strong common belief that high-level semantic language understanding (LU) crucially depends on explicit syntax. The unprecedented success of neural language learning models based on transformer networks (Vaswani et al., 2017), trained on unlabeled corpora via language modeling (LM) objectives (Devlin et al., 2019; Liu et al., 2019b; Clark et al., 2020, *inter alia*) on a wide variety of LU tasks (Wang et al., 2018; Hu et al., 2020), however, questions this widely accepted assumption.

The question of necessity of supervised parsing for LU and NLP in general has been raised before. More than a decade ago, Bod (2007) questioned the superiority of supervised parsing over unsupervised induction of syntactic structures in the context of statistical machine translation. Nonetheless, the NLP community has since still managed to find sufficient evidence for the usefulness of explicit syntax in higher-level LU tasks (Levy and Goldberg, 2014; Cheng and Kartsaklis, 2015; Bastings et al., 2017; Kasai et al., 2019; Zhang et al., 2019a, *inter alia*). However, we believe that the massive improvements brought about by the LM-pretrained transformers – unexposed to any explicit syntactic signal – warrant a renewed scrutiny of the utility of supervised parsing for high-level language understanding.[1,2] The research question we address in this work can be summarized as follows:

---

[1] **Disclaimer 1:** In this work, we make a clear distinction between Computational Linguistics (CL), i.e., the area of linguistics leveraging computational methods for analyses of human languages and NLP, the area of artificial intelligence tackling human language in order to perform intelligent tasks. This work scrutinizes the usefulness of supervised parsing and explicit syntax only for the latter. We find the usefulness of explicit syntax in CL to be self-evident.

[2] **Disclaimer 2:** The purpose of this work is definitely not to invalidate the admirable efforts on syntactic annotation and modeling, but rather to make an empirically driven step towards a deeper understanding of the relationship between LU and formalised syntactic knowledge, and the extent of its impact to modern semantic LU and applications.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3090–3104
April 19 - 23, 2021.

(RQ) *Is explicit structural language information, provided in the form of a widely adopted syntactic formalism (Universal Dependencies, UD) (Nivre et al., 2016) and injected in a supervised manner into LM-pretrained transformers beneficial for transformers' downstream LU performance?*

While existing body of work (Lin et al., 2019; Tenney et al., 2019; Liu et al., 2019a; Kulmizev et al., 2020; Chi et al., 2020) probes transformers for structural phenomena, our work is more pragmatically motivated. We directly evaluate the effect of infusing structural language information from UD treebanks, via intermediate dependency parsing (DP) training, on transformers' performance in downstream LU. To this end, we couple a pretrained transformer with a biaffine parser similar to Dozat and Manning (2017), and train the model (i.e., fine-tune the transformer) for DP. Our parser on top of RoBERTa (Liu et al., 2019b) and XLM-R (Conneau et al., 2020) produces DP results which are comparable to state of the art. We then fine-tune the syntactically-informed transformers for three downstream LU tasks: natural language inference (NLI) (Williams et al., 2018; Conneau et al., 2018), paraphrase identification (Zhang et al., 2019b; Yang et al., 2019), and causal commonsense reasoning (Sap et al., 2019; Ponti et al., 2020). We quantify the contribution of explicit syntax by comparing LU performance of the transformer exposed to intermediate parsing training (IPT) and its counterpart directly fine-tuned for the downstream task. We investigate the effects of IPT (1) *monolingually*, by fine-tuning English transformers, BERT and RoBERTa, on an English UD treebank and for (2) downstream *zero-shot language transfer*, by fine-tuning massively multilingual transformers (MMTs) – mBERT and XLM-R (Conneau et al., 2020) – on treebanks of downstream target languages, before the downstream fine-tuning on source language (English) data.

While intermediate parsing training is obviously not the only way of bringing syntactic knowledge to downstream tasks (Kuncoro et al., 2019; Swayamdipta et al., 2019; Kuncoro et al., 2020), it is arguably the most straightforward way of injecting syntactic signal in the context of the predominant pretraining-fine-tuning paradigm that has, nonetheless, not been investigated up to this point. Other methods of bringing syntactic signal to downstream tasks such as knowledge distillation (Kuncoro et al., 2020) and pre-training on shallow trees instead of sequences (Swayamdipta et al., 2019) have failed to demonstrate significant gains on higher-level LU tasks.

Our results also render supervised UD parsing largely inconsequential to LU. We observe limited and inconsistent gains only in zero-shot downstream language transfer: further analyses reveal that (1) intermediate LM training yields comparable gains and (2) IPT only marginally changes representation spaces of transformers exposed to sufficient amount of language data in LM-pretraining. We hope that these empirical findings will shed new light on the relationship between supervised parsing (and manually labeled treebanks) and LU with transformer networks, and guide further similar investigations in future work, in order to fully understand the impact of formal syntactic knowledge on LU performance with modern neural architectures.

2 Related Work

Bringing Explicit Syntax to LMs. Previous work has attempted to enrich language models with explicit syntactic knowledge in ways other than intermediate parsing training. Swayamdipta et al. (2019) modify the pretraining objective of ELMo (Peters et al., 2018) to learn from shallowly parsed (i.e., chunked) corpora. They, however, report no notable improvements on downstream tasks. Kuncoro et al. (2019) propose to distil the knowledge from a Recurrent NN Grammar (RNNG) teacher trained on a small syntactically annotated corpus (by modeling the joint probability of surface sequence and phrase structure tree) into an LSTM-based student pretrained on a much larger corpus. They show that distillation helps the student in structured prediction tasks, but their downstream evaluation does not involve LU tasks. Their subsequent work (Kuncoro et al., 2020) replaces the RNN student with BERT (Devlin et al., 2019): syntactic distillation again helps structured prediction, but hurts (slightly) the performance on LU tasks from the GLUE benchmark (Wang et al., 2018).

Transformer-Based Dependency Parsing. Building on the success of preceding neural parsers (Chen and Manning, 2014; Kiperwasser and Goldberg, 2016), Dozat and Manning (2017) proposed a biaffine parsing head on top of a Bi-LSTM encoder: contextualized word vectors are fed to two feed-forward networks, producing dependent- and head-specific token representations, respectively. Arc and relation scores are produced via biaffine products between these dependent- and head-specific representation matrices. Finally, the Edmonds algorithm induces the optimal tree from pairwise arc predictions. Most recent DP work (Kondratyuk and Straka, 2019; Üstün et al., 2020) replaces the Bi-LSTM encoder with multilingual BERT's transformer, reporting state-of-the-art parsing performance. Kondratyuk and Straka (2019) fine-tune mBERTs parameters on the concatenation of all UD treebanks, whereas Üstün et al. (2020) freeze the original transformer's parameters and inject adapters (Houlsby et al., 2019) for parsing.

We propose and work with a simpler transformer-based biaffine parser: we apply biaffine attention directly on representations from transformer's output layer, eliminating the head- and dependendant-based feed-forward mapping. Despite this simplification, our biaffine parser produces DP results comparable to current state-of-the-art parsers.

Syntactic BERTology. The substantial body of syntactic probing work shows that BERT (Devlin et al., 2019) (a) encodes text in a hierarchical manner (i.e., it encodes some implicit underlying syntax) (Lin et al., 2019); and (b) captures specific shallow syntactic information (parts-of-speech and syntactic chunks) (Tenney et al., 2019; Liu et al., 2019a). Hewitt and Manning (2019) find that linear transformations, when applied on BERT's contextualized word vectors, reflect distances in dependency trees. This suggests that BERT encodes sufficient structural information to reconstruct dependency trees (though without arc directionality and relations). Chi et al. (2020) extend the analysis to multilingual BERT, finding that its representation subspaces may recover trees also for other languages. They also provide evidence that clusters of head–dependency pairs roughly correspond to UD relations. Similarly, Kulmizev et al. (2020) show that BERT's latent syntax corresponds more to UD trees than to shallower SUD (Gerdes et al., 2018) structures. Despite the evident similarity between BERT's latent syntax and formalisms such as UD, there is ample evidence that BERT insufficiently leverages syntax in downstream tasks: it often produces similar predictions for syntactically valid as well as for structurally corrupt sentences (e.g., with random word order) (Wallace et al., 2019; Ettinger, 2020; Zhao et al., 2020).

Intermediate Training. Sometimes called Supplementary Training on Intermediate Labeled-data Tasks (STILT) (Phang et al., 2018), intermediate training is a transfer learning setup in which one trains an LM-pretrained transformer on one or more supervised tasks (ideally with large training sets) before final fine-tuning for the target task. Phang et al. (2018) show that intermediate NLI training of BERT on the Multi-NLI dataset (Williams et al., 2018) benefits several language understanding tasks. Subsequent work (Wang et al., 2019; Pruksachatkun et al., 2020) investigated many combinations of intermediate and target LU tasks, failing to identify any universally beneficial intermediate task. In this work we use DP as an intermediate training task (IPT) for LM-pretrained transformers.

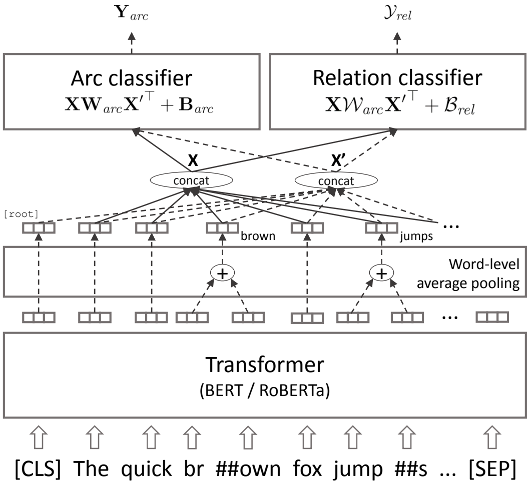


Figure 1: Architecture of our transformer-based biaffine dependency parser.

3 Methodology

Biaffine Parser. Our parsing model, illustrated in Figure 1, consists of a biaffine attention layer applied directly on the transformer's output (BERT, RoBERTa, mBERT, or XLM-R). We first obtain word-level vectors by averaging transformed representations of their constituent subwords, produced by the transformer. Let $\mathbf{X} \in \mathbb{R}^{N \times H}$ denote the encoding of a sentence with N word-level tokens, consisting of N H-dimensional vectors (where H is the transformer's hidden size). We use the transformed representation of the sentence start token (e.g., `[CLS]` for BERT), $\mathbf{x}_{CLS} \in \mathbb{R}^{H}$, as the representation for the `root` node of the parse tree, and prepend it to $\mathbf{X}$, $\mathbf{X}' = [\mathbf{x}_{CLS}; \mathbf{X}] \in \mathbb{R}^{(N+1) \times H}$. We then use $\mathbf{X}$ as the representation of syntactic dependants and $\mathbf{X}'$ as the representation of dependency heads. We then directly compute the arc and relation scores as biaffine products of $\mathbf{X}$ and $\mathbf{X}'$:

$$\mathbf{Y}_{arc} = \mathbf{X}\mathbf{W}_{arc}\mathbf{X}'^{\top} + \mathbf{B}_{arc};\ \ \mathbf{Y}_{rel} = \mathbf{X}\mathbf{W}_{rel}\mathbf{X}'^{\top} + \mathbf{B}_{rel}$$

where $\mathbf{W}_{arc} \in \mathbb{R}^{H \times H}$ and $\mathbf{W}_{rel} \in \mathbb{R}^{H \times H \times R}$ denote, respectively, the arc classification matrix and relation classification tensor (with R as the number of relations); $\mathbf{B}_{arc}$ and $\mathbf{B}_{rel}$ denote the corresponding bias parameters. We greedily select the dependency head for each word by finding the maximal score in each row of $\mathbf{Y}_{arc}$: while this is not guaranteed to produce a tree, Zhang et al. (2017) show that in most cases it does.[3] Our arc prediction loss is the cross-entropy loss with sentence words (plus the `root` node) as categorical labels: this implies a different number of labels for different sentences. We compute the relation prediction loss as a cross-entropy loss over gold arcs. Our final loss is the sum of the arc loss and relation loss.

Note that, in comparison with the original biaffine parser (Dozat and Manning, 2017) and its other transformer-based variants (Kondratyuk and Straka, 2019; Üstün et al., 2020), we feed word-level representations derived from the transformer's output directly to biaffine products, omitting the dependent- and head-specific MLP transformations. Deep task-specific architectures go against the fine-tuning idea: deep transformers have plenty of their own parameters that can be tuned for DP. We want to propagate as much of the explicit syntactic knowledge as possible into the transformer: a deep(er) DP-specific architecture on top of the transformer would impede the propagation of this knowledge to the transformer's parameters.

Downstream Models. After IPT, we fine-tune transformers for two *types* of LU tasks: (1) *sequence classification* (SEQC) tasks, where a sequence of text needs to be assigned a discrete label; and (2) *multiple choice classification* (MCC) tasks where we need to select the correct answer between two or more options for a given a premise and/or question. For SEQC, we simply apply a softmax classifier on the transformed representation of the sequence start token: $\mathbf{y} = \mathit{softmax}\left(\mathbf{x}_{CLS}\mathbf{W}_{sc} + \mathbf{b}_{sc}\right)$ (with $\mathbf{W}_{sc} \in \mathbb{R}^{H \times C}$ and $\mathbf{b}_{sc} \in \mathbb{R}^{C}$ as classifier's parameters and C as the number of task's labels).

For MCC tasks, we first concatenate each of the offered answer choices (independently of each other) to the premise and/or question, and encode it with the transformer. Since some of these tasks, e.g., COPA (Roemmele et al., 2011; Ponti et al., 2020), have very small training sets, we would like to support model transfer between different MCC tasks. Different multiple-choice classification tasks, however, may differ in the number of choices: a classifier with the number of parameters depending on the number of labels is thus not a good fit; instead, we follow Sap et al. (2019) and Ponti et al. (2020), and couple the transformer with a feed-forward network outputting a single scalar for each answer. Let $\mathbf{x}^{i}_{CLS} \in \mathbb{R}^{H}$ be the representation of the sequence start token (i.e., `[CLS]` or `<s>`) for the concatenation of the premise/question and the i-th answer. We obtain the score for the i-th answer as follows:

$$y_i = \mathbf{W}^{o}_{mcc} \tanh\left(\mathbf{W}^{h}_{mcc}\mathbf{x}^{i}_{CLS} + \mathbf{b}^{h}_{mcc}\right)$$

with $\mathbf{W}^{h}_{mcc} \in \mathbb{R}^{H \times H}$, $\mathbf{b}^{h}_{mcc} \in \mathbb{R}^{H}$ and $\mathbf{W}^{o}_{mcc} \in \mathbb{R}^{1 \times H}$ as parameters. We then apply a softmax function on the concatenation of y_i scores of all answers: $\mathbf{y} = \text{softmax}([y_1, \ldots, y_K])$, with K as the number of answers (i.e., labels) in the task. Finally, we compute the cross-entropy loss on $\mathbf{y}$.

4 Experimental Setup

We now detail experimental setup, where LU fine-tuning follows Intermediate Parsing Training (IPT).

4.1 Sequential Fine-Tuning

Our primary goal is to identify if injection of explicit syntax into transformers via supervised parsing training improves their downstream LU performance – this translates into sequential fine-tuning: (1) we first attach a biaffine parser from §3 on the transformer and train the whole model on a UD treebank; (2) we then couple the syntactically-informed transformer with the corresponding downstream classification head and perform final fine-tuning. We then compare the downstream performance of transformers with and without the IPT step.

Mono- vs. Cross-Lingual IPT Experiments. In the monolingual setup, we work with English (EN) transformers, BERT and RoBERTa, pretrained on EN corpora. In the zero-shot language transfer setup, where we work with multilingual models, mBERT and XLM-R (Conneau et al., 2020), we first train transformers via IPT on the UD treebank of the target language (i.e., a language with no downstream training data) before fine-tuning it on the EN training set of the LU task. We experiment with four target languages: German (DE),

---

[3] They also show the performance of greedy decoding to match that of decoding algorithms that produce optimal trees.

French (FR), Turkish (TR), and Chinese (ZH).[4]

Standard vs. Adapter-Based Fine-Tuning. Standard fine-tuning updates all transformer's parameters, which, for tasks with large training sets may have some drawbacks: (i) fine-tuning may last long and (ii) task-specific information may overwrite the useful distributional knowledge obtained during LM-pretraining. *Adapter-based fine-tuning* (Houlsby et al., 2019; Pfeiffer et al., 2020) remedies for these potential issues by keeping the original transformer's parameters frozen and inserting new *adapter parameters* in transformer layers. In fine-tuning, both sets of parameters are used to make predictions, but we only update adapters based on loss gradients. As the number of adapter parameters is only a fraction of the number of original parameters (3-8%), fine-tuning is also much faster.

Therefore, to account for the possibility of forgetting distributional knowledge in standard IPT fine-tuning, we also carry out adapter-based IPT. We follow Houlsby et al. (2019) and inject two *bottleneck adapters* into each transformer layer: first after the multi-head attention sublayer and another after the feed-forward sublayer. In downstream LU tasks, however, we unfreeze the original transformer parameters and fine-tune them together with adapters (now containing syntactic knowledge).

4.2 Language Understanding Tasks

We now outline the downstream LU tasks. For brevity, we report all the technical training and optimization details in the Supplementary Material.

NLI is a ternary sentence-pair classification task. We predict if the hypothesis is *entailed* by the premise, *contradicts* it, or neither. For monolingual EN experiments, we use Multi-NLI (Williams et al., 2018). In zero-shot transfer experiments, we train on EN Multi-NLI and evaluate on target language (DE , FR , TR , ZH) test portions of the multilingual XNLI dataset (Conneau et al., 2018). Models trained on the Multi-NLI datasets have been shown, however, to capture certain heuristics (e.g., lexical overlap) useful for many training instances rather than more complex and generalizable language inference (McCoy et al., 2020). Because of this, we additionally evaluate on the HANS dataset (McCoy et al., 2020), consisting of adversarial examples on which models that capture such heuristics fail.

Paraphrase Identification is a binary classification task where we predict if two sentences are mutual paraphrases. For EN, we train, validate, and test on respective portions of the PAWS dataset (Zhang et al., 2019b). In zero-shot language transfer, we evaluate on the test DE , FR , and ZH portions of the PAWS-X dataset (Yang et al., 2019).

Commonsense Reasoning. We evaluate on two multiple-choice classification (MCC) datasets. In monolingual evaluation, we use the SocialIQA (SIQA) dataset (Sap et al., 2019), testing models' ability to reason about social interactions. Each SIQA instance consists of a premise, a question, and three possible answers. For zero-shot language transfer experiments, we resort to the recently published XCOPA dataset (Ponti et al., 2020), obtained by translating test portions of the EN COPA (Choice of Plausible Alternatives) dataset (Roemmele et al., 2011) to 11 languages. As mentioned, (X)COPA is an MCC task, with each instance containing a premise, a question,[5] and two possible answers. Due to the very limited size of the EN COPA training set (mere 400 instances), we follow Ponti et al. (2020) and evaluate the models fine-tuned on SIQA (EN) on the XCOPA test portions (in TR and ZH).

4.3 Training and Optimization Details

All the transformer models with which we experiment – EN BERT, mBERT, EN RoBERTa, and XLM-R have $L = 12$ layers and hidden representations of size $H = 768$. We apply a dropout ($p = 0.1$) on the transformer outputs before forwarding them to the task-specific classification heads (i.e., biaffine parsing head in intermediate parsing training, and MCC or SEQC heads in downstream fine-tuning). We optimize the parameters using the Adam algorithm (Kingma and Ba, 2015): we found the initial learning rate of 10^{-5} to offer stable convergence in both intermediate parsing training and downstream fine-tuning for all LU tasks. We train for at most 30 epochs over the respective training set, with early stopping based on the development loss.[6] On UD treebanks and

---

[4] Selected languages vary in typological and etymological proximity to EN as the source language: DE is in the same (Germanic) branch of Indo-European languages, FR is from the different branch of the same family, whereas TR (Turkic) and ZH (Sino-Tibetan) belong to different language families.

[5] While SIQA has unconstrained questions, (X)COPA has only two question types: a) What is the CAUSE of this (premise)? and b) What is the RESULT of this (premise)?

[6] We measure the development loss every U update steps and stop the training if the loss does not decrease over 10 consecutive measurements. We set $U = 500$ in NLI training and $U = 250$ in all other training procedures.

Transformer	**Fine-tune**	EN (EWT) UAS	EN (EWT) LAS	DE (GSD) UAS	DE (GSD) LAS	FR (GSD) UAS	FR (GSD) LAS	TR (IMST) UAS	TR (IMST) LAS	ZH (GSD) UAS	ZH (GSD) LAS
BERT	Standard	91.9	89.3	–	–	–	–	–	–	–	–
	Adapter	90.1	87.3	–	–	–	–	–	–	–	–
RoBERTa	Standard	93.0	**90.5**	–	–	–	–	–	–	–	–
	Adapter	91.5	88.7	–	–	–	–	–	–	–	–
mBERT	Standard	91.5	88.9	76.3	72.0	94.1	91.3	75.5	67.5	87.0	**83.8**
	Adapter	89.6	86.8	75.1	70.1	92.8	89.7	66.4	57.8	81.0	77.4
XLM-R	Standard	**93.1**	**90.5**	**89.4**	**85.0**	**94.3**	**91.7**	**77.9**	**70.0**	79.0	75.6
	Adapter	91.4	88.6	88.3	83.8	93.1	90.3	72.1	64.1	73.8	70.3
Baseline: UDify (mBERT, Standard)		91.0	88.5	87.8	83.6	93.6	91.5	74.6	67.4	**87.9**	**83.8**

Table 1: Dependency parsing performance of our transformer-based biaffine parsers.

Transf.	**Parsing FT**	NLI	HANS	PAWS	SIQA
BERT	None	84.1	53.3	**92.4**	**60.7**
	Standard	**84.4**	**56.7**	91.9	58.8
	Adapter	84.1	53.3	**92.4**	58.3
RoBERTa	None	**88.4**	**67.4**	94.7	67.2
	Standard	87.7	64.5	**94.9**	66.5
	Adapter	87.9	66.3	94.7	**67.3**

Table 2: Downstream LU performance of monolingual EN transformers (BERT and RoBERTa). **None**: no IPT; **Standard**: IPT via standard fine-tuning; **Adapter**: IPT via adapter-based fine-tuning.

SIQA we train in batches of size 8, whereas on Multi-NLI and PAWS we train in batches of size 32. In Adapter-based IPT, we set the adapter size to 64 and use GELU (Hendrycks and Gimpel, 2016) as the activation function in adapter layers.

5 Evaluation

We first discuss parsing performance of our novel biaffine parser (see §3). We then show transformers' downstream LU performance after IPT, both in monolingual EN setting and in zero-shot transfer.

5.1 Results and Discussion

Parsing Performance. In order to judge the benefits of IPT in downstream LU, we must first verify parsing performance of our biaffine parser, i.e., that we successfully fine-tune transformers for DP. Table 1 shows that our biaffine parser gives state-of-the-art performance for all five languages in our study. Our (m)BERT-based parser outperforms UDify (Kondratyuk and Straka, 2019), also based on mBERT, for EN, FR, and TR, and performs comparably for ZH.[7] Our parser based on XLM-R additionally yields an improvement over UDify for DE as well. It is worth noting that UDify trains the mBERT-based parser (1) on the concatenation of all UD treebanks and that it (2) additionally exploits gold UPOS and lemma annotations. We train our parsers only on the training portion of the respective treebank without using any additional morpho-syntactic information.[8] Our mBERT-based parser outperforms our XLM-R-based parser only for ZH: this is likely due to a tokenization mismatch between XLM-R's subword tokenization for ZH and gold tokenization in the ZH-GSD treebank.[9]

Monolingual EN Results. Table 2 quantifies the effects of applying IPT with the EN-EWT UD treebank to BERT and RoBERTa. We report downstream LU performance on NLI, PAWS, and SIQA. The reported results do not favor supervised parsing (i.e., explicit syntax): compared to original transformers that have not been exposed to any explicit syntactic supervision, variants exposed to UD syntax via IPT (Standard, Adapter) fail to produce any significant gains for any of the downstream LU tasks. One cannot argue that the cause of this might be forgetting (i.e., overwriting) of the distributional knowledge obtained in LM pretraining during IPT: Adapter IPT variants, in which all distributional knowledge is preserved by design, also fail to yield any significant LU gains. IPT yields the largest gain (+3.4%) for BERT on HANS – the NLI dataset consisting of adversarial examples for which syntax deliberately affects the sentence meaning more directly. The same effect, however, is not there for RoBERTa, suggesting that the additional syntactic knowledge that BERT gets through IPT, RoBERTa seems to obtain through larger-scale pretraining.

Zero-Shot Language Transfer. We show the results obtained for zero-shot downstream language

---

[7] Our mBERT-based parser performs poorly for DE: the cause of it is unclear and this requires further investigation.

[8] Also, since absolute parsing performance is not the primary objective of this work, we did not perform extensive language-specific hyperparameter tuning. One could likely obtain better parsing scores than what we report in Table 1 with careful language-specific model selection.

[9] We explain this mismatch in the Appendix.

Transformer	Parse FT	XNLI DE	XNLI FR	XNLI TR	XNLI ZH	PAWS-X DE	PAWS-X FR	PAWS-X ZH	XCOPA TR	XCOPA ZH
mBERT	None	71.0	73.7	**63.0**	70.3	85.1	86.3	76.4	52.0	61.2
	Standard	71.4	72.9	61.5	**70.4**	85.4	86.9	**79.8**	**57.4**	**65.4**
	Adapter	**71.7**	**74.8**	62.5	70.2	**85.8**	**87.1**	78.7	50.4	61.6
XLM-R	None	77.1	**78.1**	73.4	73.8	**88.3**	**89.3**	**81.4**	**61.2**	66.4
	Standard	76.1	77.2	73.1	73.8	86.4	89.2	81.1	59.2	**67.4**
	Adapter	**77.8**	76.4	**73.9**	**74.7**	86.7	88.7	80.7	57.4	65.6

Table 3: Performance of multilingual transformers, mBERT and XLM-R, in zero-shot language transfer for downstream LU tasks, with and without prior intermediate dependency parsing training on target language treebanks.

transfer setup, for both mBERT and XLM-R, in Table 3. Again, these results do not particularly favor the intermediate injection of explicit syntactic information in general. However, in few cases we do observe gains from the intermediate target-language parsing training: e.g., 3% gain on PAWS-X for ZH as well as 4% and 5% gains on XCOPA for ZH and TR, respectively. Interestingly, all substantial improvements are obtained for mBERT; for XLM-R, the improvements are less consistent and less pronounced. This might be due to XLM-R's larger capacity which makes it less susceptible to the "curse of multilinguality" (Conneau et al., 2020): with the subword vocabulary twice as large as mBERT's, XLM-R is able to store more language-specific information. Also, XLM-R has seen substantially more target language data in LM-pretraining than mBERT for each language. This might mean that the larger IPT gains for mBERT come from mere exposure to additional target language text rather than from injection of explicit syntactic UD signal (see further analyses in §5.2).

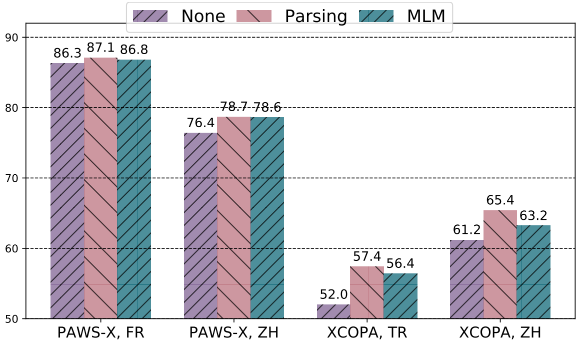


Figure 2: Comparison of IPT and ILMT in zero-shot language transfer experiments with mBERT on PAWS-X (FR and ZH) and XCOPA (TR and ZH). **None**: no intermediate training; **Parsing**: intermediate parsing training; **MLM**: intermediate masked LM training.

5.2 Further Analysis and Discussion

We first compare the impact of IPT with the effect of additional LM training on the same raw data. We then quantify the topological modification that IPT makes in transformers' representation spaces.

Explicit Syntax or Just More Language Data? We scrutinize the IPT gains that we observe in some zero-shot language transfer experiments. We hypothesize that these gains may, at least in part, be credited to transformer simply seeing more target language data. To investigate this, we replace IPT with intermediate (masked) language modeling training (ILMT) on the same data (i.e., sentences from the respective treebank used in IPT) before final downstream LU fine-tuning. Because MLM is a self-supervised objective, we can credit all differences in downstream LU performance between ILMT and IPT variants of the same pretrained transformer to supervised parsing, i.e., to the injection of explicit UD knowledge.

ILMT Details. We mask 15% of subword tokens in each sentence and predict them with a linear classifier applied on transformed representations of `[MASK]` tokens. We compute the cross-entropy loss and use the same hyperparameter configuration as described in §4.3. The development set, used for early stopping, is subdued to fixed masking, whereas we mask the training sentences dynamically, before feeding them to the transformer.

Results. We run this analysis for setups in which we observe substantial gains from IPT: PAWS-X for mBERT (*Adapter* fine-tuning, for FR and ZH) and XCOPA for mBERT (*Standard* fine-tuning, TR and ZH). The comparison between IPT and ILMT for these setups is provided in Figure 2. Like IPT, ILMT on mBERT generates downstream gains over direct downstream fine-tuning (i.e., no intermediate training) in all four setups. The gains from ILMT (with the exception of XCOPA for ZH) are almost as large as gains from IPT. This suggests that most of the gain with IPT comes from seeing more target language text, and prevents us from concluding that the explicit syntactic annotation is

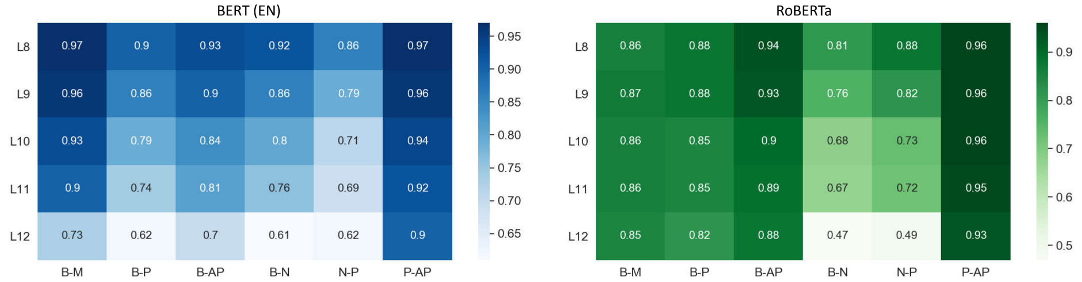


Figure 3: Topological similarity (l-CKA) for pairs of BERT and RoBERTa variants, before and after different fine-tuning steps (B, M, P, AP, and N). **Rows**: transformer layers; **Columns**: pairs of transformer variants in comparison.

responsible for the LU improvements in zero-shot downstream transfer. This interpretation is corroborated by the fact that IPT gains roughly correlate with the amount of language-specific data seen in LM-pretraining: the gains are more prominent for mBERT than for XLM-R and for TR and ZH than for FR and DE (see Table 3).

Changes in Representation Spaces. Finally, we analyze how fine-tuning transformers on different tasks modifies the topology of their representation spaces. We encode the set of sentences S from the test portions of treebanks used in IPT[10] with different variants: (a) Base (B): original LM-pretrained transformer, no further training; (b) MLM (M): after ILMT; (c) Parsing (P): after Standard IPT; and (d) Adapter-Parsing (AP): after Adapter-based IPT; for monolingual transformers (BERT and RoBERTa), also with (e) NLI (N): after NLI fine-tuning (without any intermediate training). We analyze the representations in each transformer layer separately: we represent each sentence $s \in S$ with the average of subword vectors from that layer (excluding sequence start and end tokens). Let $\mathbf{X}_1$ and $\mathbf{X}_2 \in \mathbb{R}^{|S| \times H}$ contain corresponding representations of sentences from S from the i-th layer of two transformer variants (e.g., B and P). We measure the topological similarity of the i-th layers of the two transformers with the linear centered kernel alignment (l-CKA) (Kornblith et al., 2019):[11]

$$l\text{-}CKA(\mathbf{X}_1, \mathbf{X}_2) = \frac{\left\|\mathbf{X}_2^\top \mathbf{X}_1\right\|_F^2}{\left(\left\|\mathbf{X}_1^\top \mathbf{X}_1\right\|_F\right)\left(\left\|\mathbf{X}_2^\top \mathbf{X}_2\right\|_F\right)}.$$

Although not invariant to all linear transformations, l-CKA is invariant to orthogonal projection and isotropic scaling, which suffices for our purposes. We base our analysis on the following assumption: the extent of change in transformers' representation space topology (reflected by l-CKA), is proportional to the novelty of knowledge injected in fine-tuning. Put differently, injection of new (i.e., missing) knowledge should substantially change the topology of the space (low l-CKA score).

Figure 3 shows the heatmap of l-CKA scores for pairs of BERT and RoBERTa variants, for layers L8-L12.[12] Comparing B-P and B-N reveals that IPT changes the topology of BERT's higher layers roughly as much as NLI fine-tuning does, implying that both the English UD treebank (EN -EWT) and Multi-NLI data contain a non-negligible amount of novel knowledge for BERT. However, the direct N-P comparison shows that IPT and NLI enrich BERT (also RoBERTa) with different type of knowledge, i.e., they change the representation spaces of its layers in different ways. This suggests that the transformers cannot acquire the missing knowledge needed for NLI from IPT (i.e., from EN -EWT), and explains why IPT is not effective for NLI.

IPT (comparison B-P) injects more new information than ILMT (comparison B-M), and this is more pronounced for BERT than for RoBERTa. IPT and ILMT change RoBERTa's parameters much less than BERT's (see B-M and B-P l-CKA scores for L11/L12), which we interpret as additional evidence, besides RoBERTa consistently outscoring BERT, that RoBERTa encodes richer language representations, due to its larger-scale and longer training. It also agrees with suggestions that BERT is "undertrained" for its capacity (Liu et al., 2019b).

Very high B-P (and B-AP) l-CKA scores in lower layers suggest that the explicit syntactic knowledge from human-curated treebanks is redundant w.r.t. the structural language knowledge transformers obtain through LM pretraining. This is consistent with concurrent observations (Chi et al., 2020; Kul-

---

[10] IPT itself only consumes train and development portions of UD treebanks. We can thus safely use sentences from test portions in this analysis, without risking information leakage.

[11] $\mathbf{X}_1$ and $\mathbf{X}_2$ must first be column-wise mean-centered.

[12] Most l-CKA scores in layers L1-L7 are very high (≥ 0.9) and provide little insight. See the Supplementary Material.

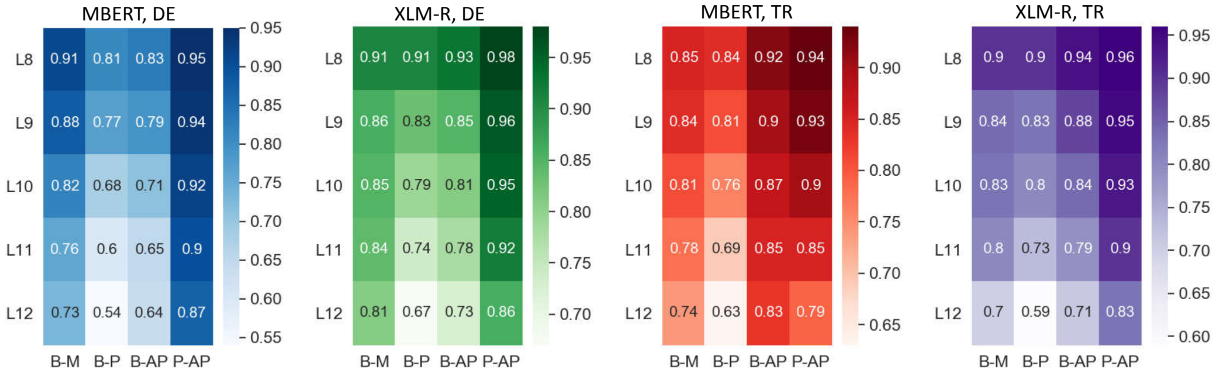


Figure 4: Analysis of topological similarity (l-CKA) for variants of mBERT and XLM-R before and after IPT and ILMT (B, M, P, AP) in zero-shot transfer experiments. Results shown for intermediate parsing on DE and TR data.

mizev et al., 2020) showing (some) correspondence between structural knowledge of (m)BERT and UD syntax. Finally, we observe highest l-CKA scores in the P-AP column, suggesting that Standard and Adapter IPT inject roughly the same syntactic information, despite different fine-tuning mechanisms.

Figure 4 illustrates the results of the same analysis for language transfer experiments, for DE and TR (scores for FR and ZH are in the Appendix). The effects of ILMT and IPT (B-M, B-P/B-AP) for DE and TR with mBERT and XLM-R resemble those for EN with BERT and RoBERTa: ILMT changes transformers less than IPT. The amount of new syntactic knowledge IPT injects is larger (l-CKA scores are lower) than for EN , especially for XLM-R (vs. RoBERTa for EN). We believe that it reflects the relative under-representation of the target language in the model's multilingual pretraining corpus (e.g., for TR): this leads to poorer representations of target language structure by mBERT and XLM-R compared to BERT's and RoBERTa's representation of EN structure. This gives us two seemingly conflicting empirical findings: (a) IPT appears to inject a fair amount of target-language UD syntax, but (b) this translates to (mostly) insignificant and inconsistent gains in language transfer in LU tasks (especially so for XLM-R, cf. Table 3). A plausible reconciling hypothesis is that there is a substantial mismatch between the type of structural information we obtain through supervised (UD) parsing and the type of structural knowledge beneficial for LU tasks. If true, this hypothesis would render supervised parsing rather unavailing for high-level language understanding, at least in the context of LM-pretrained transformers, the current state of the art in NLP. This warrants further investigation, and we hope that our work will inspire further discussion and additional studies.

6 Conclusion

We thoroughly examined the effects of leveraging formalized syntactic structures (UD) in state-of-the-art neural language models (e.g., RoBERTa, XLM-R) for downstream language understanding (LU) tasks, both in monolingual and language transfer settings. The key results, obtained through intermediate parsing training (IPT) based on a state-of-the-art-level dependency parser, indicate that explicit syntax, at least in our extensive experiments, provides negligible impact on LU tasks.

Besides offering extensive empirical evidence of the mismatch between explicit syntax and improved LU performance with state-of-the-art transformers, this study sheds new light on some fundamental questions such as the one in the title. Similar to word embeddings (Mikolov et al., 2013) removing sparse lexical features from the NLP horizon, will transformers make supervised parsing obsolete for LU applications or not? More dramatically, in the words of Rens Bod (2007): "Is the end of supervised parsing in sight" for semantic LU tasks?[13]

Acknowledgments

We thank the anonymous reviewers (especially R2!) for the exceptionally meaningful and helpful comments. Goran Glavaš is supported by the Baden Württemberg Stiftung (Eliteprogramm, AGREE grant). The work of Ivan Vulić is supported by the ERC Consolidator Grant LEXICAL: Lexical Acquisition Across Languages (no. 648909).

---

[13] The answer is 'Probably no': formalized syntactic structures will still be an important source of inductive bias, especially in setups without sufficient text data for large-scale pretraining; our experiments, however, validate that state-of-the-art transformer models can implicitly capture that inductive bias in high-resource setups and for major languages.

References

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proc. of EMNLP*, pages 1957–1967.

Rens Bod. 2007. Is the end of supervised parsing in sight? In *Proc. of ACL*, pages 400–407.

Eugene Charniak. 1996. Tree-bank grammars. In *Proc. of NCAI*, pages 1031–1036.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proc. of EMNLP*, pages 740–750.

Jianpeng Cheng and Dimitri Kartsaklis. 2015. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proc. of EMNLP*, pages 1531–1542.

Ethan A Chi, John Hewitt, and Christopher D Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proc. of ACL*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proc. of ICLR*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proc. of ACL*, pages 8440–8451.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proc. of EMNLP*, pages 2475–2485.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC*, pages 449–454.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171–4186.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proc. of ICLR*.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *TACL*, 8:34–48.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2018. Sud or surface-syntactic universal dependencies: An annotation scheme near-isomorphic to UD. In *Proc. of UDW*, pages 66–74.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (GELUs). *CoRR*, abs/1606.08415.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proc. of NAACL-HLT*, pages 4129–4138.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proc. of ICML*, pages 2790–2799.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *Proc. of ICML*.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *Proc. of NAACL-HLT*, pages 701–709.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *TACL*, 4:313–327.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. of ACL*, pages 423–430.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proc. of EMNLP*, pages 2779–2795.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proc. of ICML*, pages 3519–3529.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proc. of ACL*, pages 4077–4091.

Adhiguna Kuncoro, Chris Dyer, Laura Rimell, Stephen Clark, and Phil Blunsom. 2019. Scalable syntax-aware language models using knowledge distillation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3472–3484.

Adhiguna Kuncoro, Lingpeng Kong, Daniel Fried, Dani Yogatama, Laura Rimell, Chris Dyer, and Phil

Blunsom. 2020. Syntactic structure distillation pretraining for bidirectional encoders. *Transactions of the Association for Computational Linguistics*, 8:776–794.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proc. of ACL*, pages 302–308.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert's linguistic knowledge. In *Proc. of BlackboxNLP*, pages 241–253.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proc. of NAACL-HLT*, pages 1073–1094.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2020. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 3428–3448. Association for Computational Linguistics (ACL).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. of NeurIPS*, pages 3111–3119.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proc. of LREC*, pages 1659–1666.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. *arXiv preprint arXiv:2004.10643*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*, pages 2227–2237.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting Transformers. In *Proc. of EMNLP: System Demonstrations*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proc. of EMNLP*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? In *Proc. of ACL*, pages 5231–5247.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Proc. of AAAI SSS*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proc. of EMNLP-IJCNLP*, pages 4463–4473.

Swabha Swayamdipta, Matthew Peters, Brendan Roof, Chris Dyer, and Noah A Smith. 2019. Shallow syntax in deep water. *arXiv preprint arXiv:1908.11047*.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn treebank: An overview. In *Treebanks*, pages 5–22. Springer.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *Proc. of ICLR*.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly universal dependency parsing. In *Proc. of EMNLP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proc. of EMNLP-IJCNLP*, pages 2153–2162.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, et al. 2019. Can you tell me how to get past Sesame street? sentence-level pretraining beyond language modeling. In *Proc. of ACL*, pages 4465–4476.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. of BlacboxNLP*, pages 353–355.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL-HLT*, pages 1112–1122.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proc. of EMNLP-IJCNLP*, pages 3678–3683.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019a. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proc. of NAACL-HLT*, pages 1151–1161.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proc. of EACL*, pages 665–676.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. Paws: Paraphrase adversaries from word scrambling. In *Proc. of NAACL-HLT*, pages 1298–1308.

Wei Zhao, Goran Glavaš, Maxime Peyrard, Yang Gao, Robert West, and Steffen Eger. 2020. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In *Proc. of ACL*, pages 1656–1671.

A Reproducibility

We first provide details on where to obtain datasets and code used in this work.

A.1 Datasets

Table 5 lists the sizes (in number of sentences) of Universal Dependencies treebanks that we use for our intermediate parsing training and evaluation of our biaffine dependency parsers. The UD treebanks v.2.5, which we used in this work, are available at: `http://hdl.handle.net/11234/1-3105`. In Table 6 we provide links to language understanding datasets used in our study.

A.2 Code and Dependencies

We make our code available at: `https://github.com/codogogo/parse_stilt`. Our code is built on top of the HuggingFace Transformers framework: `https://github.com/huggingface/transformers` (v. 2.7). Table 4 details the LM-pretrained transformer models from this framework which we exploited in this work. Besides the Transformers library, our code only relies on standard Python's scientific computing libraries (e.g., `numpy`).

B ZH Tokenization: XLM-R vs. GSD

A word-level token from the parse tree normally corresponds to one or more transformer's subword tokens: we thus average subword vectors to obtain word vectors for biaffine parsing. For XLM-R and the ZH GSD treebank, however, a single XLM-R's subword token often corresponds to two treebank tokens. E.g., the sequence “只是二選一做決擇” with treebank tokenization [‘只’, ‘是’, ‘二’, ‘選’, ‘一’, ‘做’, ‘決擇’] is tokenized as [‘只是’, ‘二’, ‘選’, ‘一’, ‘做’, ‘決’, ‘擇’] by XLM-R. Two treebank tokens, ‘只’ and ‘是’, are captured with a single XLM-R “subword” token, ‘只是’. To ensure that each XLM-R subword token corresponds to exactly one treebank token, we inject spaces between treebank tokens before XLM-R tokenization: we then obtain the subword tokenization [‘只’, ‘是’, ‘二’, ‘選’, ‘一’, ‘做’, ‘決, ‘擇’]. However, this is suboptimal for XLM-R: its representations of tokens ’只’ and ’是’ are probably less reliable than that of the ’只是’ token. We believe this is why mBERT (without tokenization mismatches for ZH) outperforms XLM-R in ZH parsing.

C Complete Topology Analysis Results

Finally, we show the complete results (for all layers, all transformers, and all languages covered in our experiments) of our topological analysis of transformers' representations before and after different fine-tuning steps. Figure 5 shows the analysis results for monolingual EN transformers, BERT and RoBERTa. Figure 6 and Figure 7 show the results for multilingual transformers, mBERT and XLM-R, respectively, for all four target languages included in our experiments: DE , FR , TR , and ZH .

Name	Lang	Vocab	Params	URL
BERT	EN	29K	110M	https://huggingface.co/bert-base-cased
RoBERTa	EN	50K	110M	https://huggingface.co/roberta-base
mBERT	Multiling.	119K	125M	https://huggingface.co/bert-base-multilingual-cased
XLM-R	Multiling.	250K	125M	https://huggingface.co/xlm-roberta-base

Table 4: LM-pretrained transformer models used in our study.

Lang	Treebank	Train	Dev	Test
EN	EWT	12,538	2,002	2,077
DE	GSD	13,810	799	977
FR	GSD	14,440	1,475	416
TR	IMST	3,664	988	983
ZH	GSD	3,996	500	500

Table 5: Universal Dependencies treebanks used in our study. We display sizes of train, development, and test portions in terms of number of sentences.

Task	Dataset	URL
Natural Language Inference	Multi-NLI	https://cims.nyu.edu/~sbowman/multinli
Natural Language Inference	XNLI	https://github.com/facebookresearch/XNLI
Paraphrase identification	PAWS(-X)	https://github.com/google-research-datasets/paws
Commonsense social reasoning	SIQA	https://maartensap.github.io/social-iqa
Commonsense causal reasoning	COPA	https://people.ict.usc.edu/~gordon/copa.html
Commonsense causal reasoning	XCOPA	https://github.com/cambridgeltl/xcopa

Table 6: Links to downstream language understanding datasets used in our work.

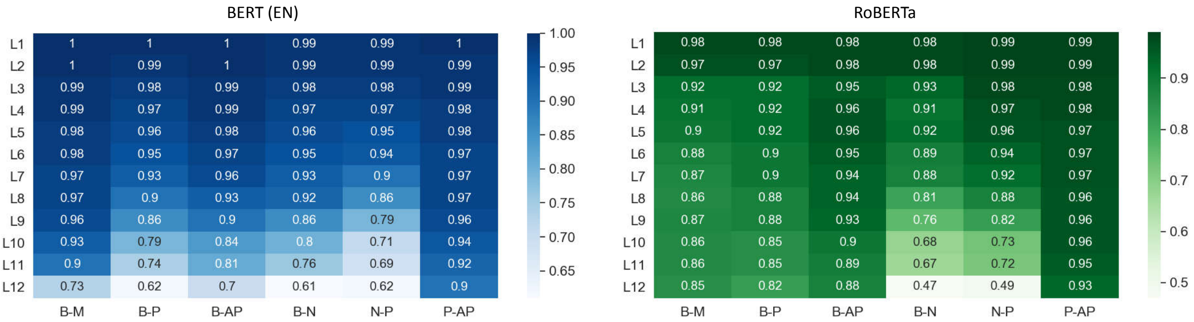


Figure 5: Full results of the topological similarity analysis (l-CKA) for pairs of BERT and RoBERTa variants, before and after different fine-tuning steps (B, M, P, AP, and N). **Rows**: transformer layers; **Columns**: pairs of transformer variants in comparison.

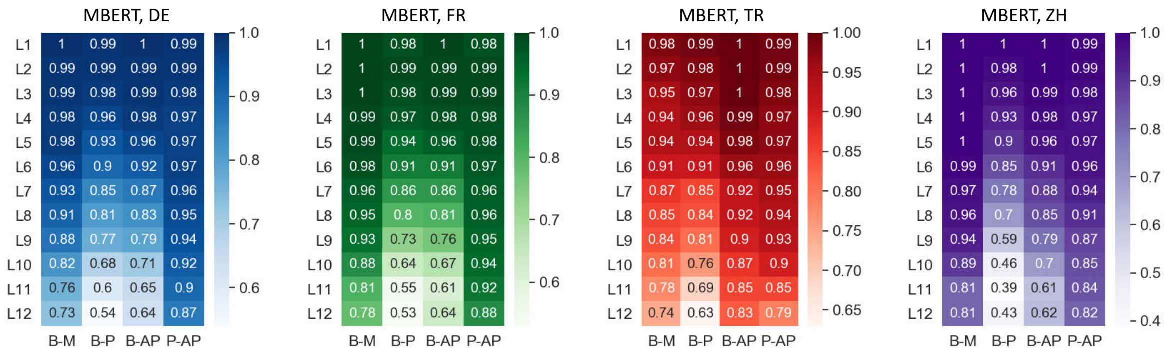


Figure 6: Full results of the topological similarity analysis (l-CKA) for variants of mBERT before and after IPT and ILMT (B, M, P, AP) for the following target languages (left to right): DE, FR, TR, and ZH.

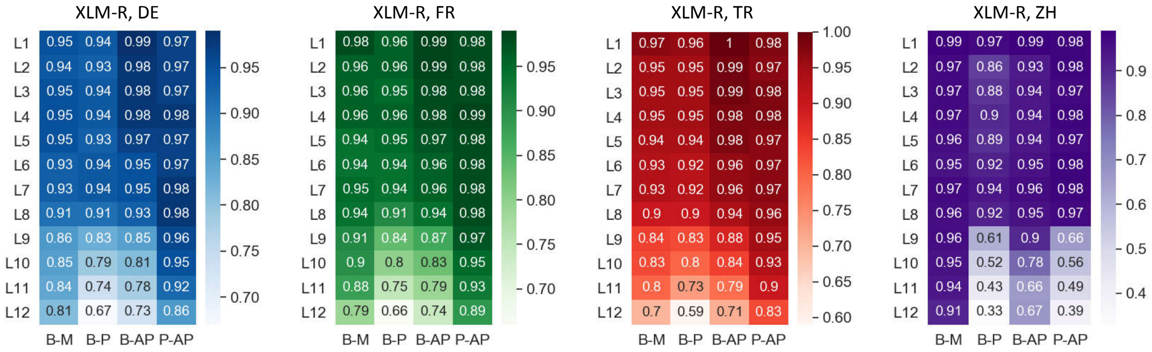


Figure 7: Full results of the topological similarity analysis (l-CKA) for variants of XLM-R before and after IPT and ILMT (B, M, P, AP) for the following target languages (left to right): DE , FR , TR , and ZH .

Facilitating Terminology Translation with Target Lemma Annotations

Toms Bergmanis[†‡] and **Mārcis Pinnis**[†‡]

[†]Tilde / Vienības gatve 75A, Riga, Latvia
[‡]Faculty of Computing, University of Latvia / Raiņa bulv. 19, Riga, Latvia
`{firstname.lastname}@tilde.lv`

Abstract

Most of the recent work on terminology integration in machine translation has assumed that terminology translations are given already inflected in forms that are suitable for the target language sentence. In day-to-day work of professional translators, however, it is seldom the case as translators work with bilingual glossaries where terms are given in their dictionary forms; finding the right target language form is part of the translation process. We argue that the requirement for apriori specified target language forms is unrealistic and impedes the practical applicability of previous work. In this work, we propose to train machine translation systems using a source-side data augmentation method[1] that annotates randomly selected source language words with their target language lemmas. We show that systems trained on such augmented data are readily usable for terminology integration in real-life translation scenarios. Our experiments on terminology translation into the morphologically complex Baltic and Uralic languages show an improvement of up to 7 BLEU points over baseline systems with no means for terminology integration and an average improvement of 4 BLEU points over the previous work. Results of the human evaluation indicate a 47.7% absolute improvement over the previous work in term translation accuracy when translating into Latvian.

1 Introduction

Translation into morphologically complex languages involves 1) making a lexical choice for a word in the target language and 2) finding its morphological form that is suitable for the morphosyntactic context of the target sentence. Most of the recent work on terminology translation, however, has assumed that the correct morphological forms are apriori known (Hokamp and Liu, 2017; Post and Vilar, 2018; Hasler et al., 2018; Dinu et al., 2019; Song et al., 2020; Susanto et al., 2020; Dougal and Lonsdale, 2020). Thus previous work has approached terminology translation predominantly as a problem of making sure that the decoder's output contains *lexically* and *morphologically* prespecified target language terms. While useful in some cases and some languages, such approaches come short of addressing terminology translation into morphologically complex languages where each word can have many morphological surface forms.

For terminology translation to be viable for translation into morphologically complex languages, terminology constraints have to be *soft*. That is, terminology translation has to account for various natural language phenomena, which cause words to have more than one manifestation of their root morphemes. Multiple root morphemes complicate the application of *hard* constraint methods, such as constrained-decoding (Hokamp and Liu, 2017). That is because even after the terminology constraint is striped from the morphemes that encode all grammatical information, the remaining root morphemes still can be too restrictive to be used as hard constraints because, for many words, there can be more than one root morpheme possible. An illustrative example is the consonant mutation in the Latvian noun *vācietis* ("the German") which undergoes the mutation t→š, thus yielding two variants of its root morpheme *vācieš-* and *vāciet-* (Bergmanis, 2020). If either of the forms is used as a hard constraint for constrained decoding, the other one is excluded from appearing in the sentence's translation.

We propose a necessary modification for the method introduced by Dinu et al. (2019), which allows training neural machine translation (NMT)

---

[1] Relevant materials and code: `https://github.com/tilde-nlp/terminology_translation`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3105–3111
April 19 - 23, 2021.

<table>
<tr><td>EN Src.:
LV Trg.:</td><td>faulty engine or in transmission[..]
atteice dzinējā vai transmisijas [..]</td></tr>
<tr><td>ETA:</td><td>faulty|w engine|s dzinējā|t or|w
transmission|s transmisijas|t [..]</td></tr>
<tr><td>TLA:</td><td>faulty|w engine|s dzinējs|t or|w
transmission|s transmisija|t [..]</td></tr>
</table>

Table 1: Examples of differences in input data in ETA (Dinu et al., 2019) and TLA (this work). Differences of inline annotations are marked in bold. |w, |s, |t denote the values of the additional input stream and stand for regular words, source language annotated words, target language annotations respectively.

	Train	**Test**	
		ATS	WMT17+IATE
EN-DE	27.6M	768	581
EN-ET	2.4M	768	-
EN-LV	22.6M	768	-
EN-LT	22.1M	768	-

Table 2: Training and evaluation data sizes in numbers of sentences. WMT2017 + IATE stands for the English-German test set from the news translation task of WMT2017 which is annotated with terminology from the IATE terminology database.

systems that are capable of applying terminology constraints: instead of annotating source-side terminology with their exact target language translations, we annotate randomly selected source language words with their target language lemmas. First of all, preparing training data in such a way relaxes the requirement for access to bilingual terminology resources at the training time. Second, we show that the model trained on such data does not learn to simply *copy* inline annotations as in the case of Dinu et al. (2019), but learns *copy-and-inflect* behaviour instead, thus addressing the need for *soft* terminology constraints.

Our results show that the proposed approach not only relaxes the requirement for apriori specified target language forms but also yields substantial improvements over the previous work (Dinu et al., 2019) when tested on the morphologically complex Baltic and Uralic languages.

2 Method: Target Lemma Annotations

To train NMT systems that allow applying terminology constraints Dinu et al. (2019) prepare training data by amending source language terms with their exact target annotations (**ETA**). To inform the NMT model about the nature of each token (i.e., whether it is a source language term, its target language translation or a regular source language word), the authors use an additional input stream—source-side factors (Sennrich and Haddow, 2016). Their method, however, is limited to cases in which the provided annotation matches the required target form and can be copied verbatim, thus performing poorly in cases where the surface forms of terms in the target language differ from those used to annotate source language sentences (Dinu et al., 2019). This constitutes a problem for the method's practical applicability in real-life scenarios. In this work, we propose two changes to the approach of Dinu et al. (2019). **First**, when preparing training data, instead of using terms found in either IATE[2] or Wiktionary as done by Dinu et al. (2019), we annotate random source language words. This relaxes the requirement for curated bilingual dictionaries for training data preparation. **Second**, rather than providing exactly those target language forms that are used in the target sentence, we use target lemma annotations (**TLA**) instead (see Table 1 for examples). We hypothesise that in order to benefit from such annotations, the NMT model will have to learn *copy-and-inflect* behaviour instead of simple *copying* as proposed by Dinu et al. (2019).

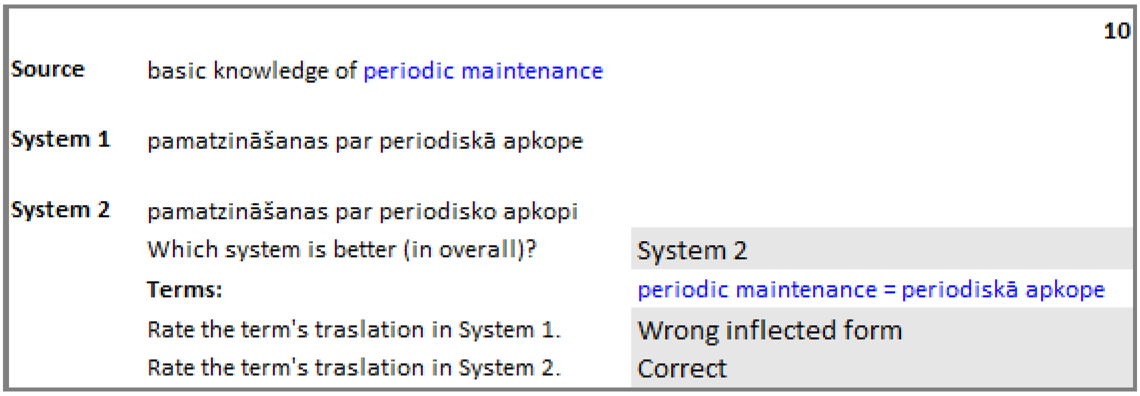
Our work is similar to work by Exel et al. (2020) in which authors also aim to achieve *copy-and-inflect* behaviour. However, authors limit their annotations to only those terms for which their base forms differ by no more than two characters from the forms required in the target language sentence. Thus wordforms undergoing longer affix change or inflections accompanied by such linguistic phenomena as consonant mutation, consonant gradation or other stem change are never included in training data.

3 Experimental Setup

Languages and Data. As our focus is on morphologically complex languages, in our experiments we translate from English into Latvian and Lithuanian (Baltic branch of the Indo-European language family) as well as Estonian (Finnic branch of the Uralic language family). For comparability with the previous work, we also use English-German (Germanic branch of the Indo-European language family). For all language pairs, we use all data that is available in the Tilde Data Libarary with an exception for English-Estonian for which

---

[2] `https://iate.europa.eu`



10

Source	basic knowledge of periodic maintenance
System 1	pamatzināšanas par periodiskā apkope
System 2	pamatzināšanas par periodisko apkopi
Which system is better (in overall)?	System 2
Terms:	periodic maintenance = periodiskā apkope
Rate the term's traslation in System 1.	Wrong inflected form
Rate the term's traslation in System 2.	Correct

Figure 1: Example of forms used in human evaluation.

we use data from WMT 2018. The size of the parallel corpora after pre-processing using the Tilde MT platform (Pinnis et al., 2018) and filtering tools (Pinnis, 2018) is given in Table 2.

To prepare data with TLA, we first lemmatise and part-of-speech (POS) tag the target language side of parallel corpora. For lemmatisation and POS tagging, we use pre-trained Stanza[3] (Qi et al., 2020) models. We then use *fast_align*[4] (Dyer et al., 2013) to learn word alignments between the target language lemmas and source language inflected words. We only annotate verbs or nouns. To generate sentences with varying proportions of annotated and unannotated words, we first generate a sentence level annotation threshold uniformly at random from the interval $[0.6, 1.0)$. Similarly, for each word in the source language sentence, we generate another number uniformly at random from the interval $[0.0, 1.0)$. If the latter is larger than the sentence level annotation threshold, we annotate the respective word with its target language lemma. We use the original training data and annotated data with a proportion of 1:1. We follow Dinu et al. (2019) to prepare ETA and replicate their results.

For validation during training, we use development sets from the WMT news translation shared tasks. For EN-ET and EN-DE, we used the data from WMT 2018, for EN-LV – WMT 2017, and for EN-LT – WMT 2019.

MT Model and Training. For the most part, we use the default configuration of the Transformer (Vaswani et al., 2017) NMT model implementation of the Sockeye NMT toolkit (Hieber et al.). The exception is the use of source-side factors (Sennrich and Haddow, 2016) with the dimensionality of 8 for systems using inline target lemma annotations. We train all models using early stopping with the patience of 10 based on their development set perplexity (Prechelt, 1998).

Evaluation Methods and Data. In previous work, methods were tested on general domain data[5] annotated with exact surface forms of general-domain words from IATE and Wiktionary. Although data constructed in such a way is not only artificial but also gives an oversimplified view on terminology translation, we do use the data from IATE to validate our re-implementation of the method from Dinu et al. (2019). Other than that, we test on the Automotive Test Suite[6] (ATS): a data set containing translations of the same 768 sentences in English, Estonian, German, Latvian, and Lithuanian. ATS contains about 1.1k term occurrences from a glossary prepared by professional translators. When annotating terms in the source text, we use only the dictionary forms of term translations, since in practical applications having access to the correct inflections (surface forms) is unrealistic.

We compare our work with an NMT system without means for terminology integration (**Baseline**) and the previous work by Dinu et al. (2019) (**ETA**). Although our preliminary experiments with constrained decoding (Post and Vilar, 2018) (**CD**) confirmed the findings by Dinu et al. (2019) that strict enforcement of constraints leads to lower-than-baseline quality, we nevertheless include them for completeness sake.

Similarly to the previous work, we use two auto-

---

[3] https://github.com/stanfordnlp/stanza

[4] https://github.com/clab/fast_align

[5] https://github.com/mtresearcher/terminology_dataset

[6] https://github.com/tilde-nlp/terminology_translation

	IATE		Automotive Test Suite							
	EN-DE		EN-DE		EN-ET		EN-LV		EN-LT	
	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.
Baseline	29.7	81.7	26.5	46.2	19.6	46.7	30.6	62.2	25.3	51.2
CD	28.5	**99.7**	22.9	**99.7**	14.9	**98.0**	23.5	**99.3**	18.1	**98.9**
ETA	**29.9**	96.2	33.2†	94.0	17.8	92.4	27.4	93.4	28.8†	89.7
TLA	29.5	96.5	**33.5**†	94.0	**21.0**†‡	87.2	**35.0**†‡	92.0	**30.1**†‡	90.3

Table 3: Results of automatic evaluation metrics BLEU and term translation accuracy (Acc.). The numerically highest score in each column is given in bold; † and ‡ indicate statistically significant improvements of BLEU over Baseline and ETA respectively (all $p < 0.05$) .

	Correct	Wrong lexeme	Wrong inflect.	Other	κ_{free}
Basel.	55.1	42.9	1.4	0.7	0.95
ETA	45.2	7.9	44.9	2.0	0.87
TLA	92.9	5.1	1.4	0.7	0.98

Baseline	**Equal**	**TLA**	κ_{free}
3.0	58.0	39.0	0.65

ETA	**Equal**	**TLA**	κ_{free}
3.0	36.0	61.0	0.81

Table 4: Results of human evaluation: term (on the left) and sentence (on the right) translation quality judgements in %. Sentence comparison is pairwise contrasting TLA vs Baseline and TLA vs ETA. κ-free: inter-annotator agreement according to free marginal kappa (Randolph, 2005).

matic means for evaluation: BLEU (Papineni et al., 2002) and lemmatised term exact match accuracy. We use BLEU as an extrinsic evaluation metric as we expect that, when successful, the methods for terminology translation should yield substantial overall translation quality improvements due to correctly translated domain-specific terms. For significance testing, we use pairwise bootstrap resampling (Koehn, 2004). We use lemmatised term exact match accuracy as an intrinsic metric because it directly measures the adequacy of terminology translation (i.e., whether or not the correct lexeme appears in the target sentence).

We are aware that the automatic evaluation methods are merely an approximation of translation quality. For example, we use lemmatised term exact match accuracy to measure term use in target language translations; however, it does not capture whether the term is inflected correctly. Thus human evaluation is in place. We use the EN-LV language pair to compare TLA against baseline and ETA. We use a 100 sentences large randomly selected ATS subset that contains 147 terms of the original test suite. We employ four professional translators and Latvian native speakers to compare each system's translations according to their overall translation quality and judge individual term translation quality. Specifically, given the original sentence and its two translations (in a randomised order), raters are asked to answer "which system's translation is better overall?". Raters are also given a list of the terms being evaluated and their reference translations (from the term collection) and are asked to classify translations as either "Correct", "Wrong lexeme", "Wrong inflection", or "Other". Figure 1 gives an example of the forms presented to raters during the human evaluation of term and overall translation quality. We report inter-annotator agreement using free marginal kappa, κ_{free} (Randolph, 2005).

4 Results

Automatic Evaluation. We first validate our re-implementation of ETA by testing on the English-German WMT 2017 test set annotated with terms from IATE as used by Dinu et al. (2019). Results (see columns 2 and 3 of Table 3) are similar to those of the previous work: on this data set, ETA yields minor translation quality improvements over the baseline (+0.2 BLEU) and considerable improvement (+14.5%) in term translation accuracy.

When evaluated on the ATS, systems using TLA always yield results that are better than the baseline both in terms of BLEU scores (+1.4–7 BLEU) and term translation accuracy (29.8%–47.8%) (see columns 4-11 of Table 3). Results also show that when compared to ETA, systems integrating terminology using TLA achieve statistically significant improvements in terms of BLEU scores for three out of four languages-pairs. An exception is EN-DE, for which both systems, ETA and TLA, perform similarly. Analysing reference translations of the EN-DE language pair, we find that as many as 87% of the German terms are used in their dictionary forms, which explains the comparable performance of systems trained using ETA and TLA on EN-DE.

Results also confirm the finding of the previous work by Dinu et al. (2019) and Exel et al. (2020), that the strict enforcement of constraints by constrained decoding leads to lower-than-baseline BLEU scores on all data sets for all languages. BLEU scores are abysmal when translating into the morphologically complex languages as for these languages citation form seldom happens to be the form required in the target language sentence. This result further illustrates why terminology constraints have to be *soft* when translating into morphologically complex languages.

Human Evaluation. Results of human evaluation of EN-LV systems are summarised in Table 4. First, we note that on this dataset, the baseline system translates terms correctly 55% of the time, yet it makes mistakes by choosing the wrong lexeme for most of the other cases (Table 4, left). The system using ETA, on the other hand, has a much lower rate of correctly translated terms – 45%, which roughly corresponds to the proportion of Latvian terms in the reference translations that are used in their dictionary forms (47%). The remaining cases are mistranslated by choosing the wrong inflected form. The system using TLA, in comparison, does very well as it gets terminology translations right 93% of the time. Examining the cases where terms had been mistranslated by choosing the wrong lexeme, we find that most of these cases are multi-word terms with some other word inserted between their constituent parts. The high κ-free values indicate almost perfect inter-annotator agreement suggesting that the task of term translation quality evaluation has been easy and results are reliable.

The overall sentence translation quality judgements (Table 4, right) also favour translations produced by the system using TLA deeming it better than or on par with the baseline system and system using ETA 97% of the time. The system using TLA is strictly favoured over its ETA counterpart for 61% of the translations. Again, annotators have reached an almost perfect agreement ($\kappa_{\text{free}} = 0.81$) when comparing the systems using TLA and ETA, suggesting that the task has been easy. These results clearly show that at least for the EN-LV language pair and the test set considered here, systems using TLA improve term translation quality by correctly choosing adequate translations and morpho-syntactically appropriate inflections.

Productivity of NMT models. Terminology translation frequently involves the translation of niche lexemes with rare or even unseen inflections. Thus the model's ability to generate novel wordforms is critical for high-quality translations. To verify if our NMT models are lexically and morphologically productive, we analysed Latvian translations of ATS produced by the system using TLA and looked for wordforms that are not present in either source or target language side of the training data. We found 72 such wordforms. Of those 45 or 62.5% were valid wordforms that were not present in training data, of which 28 were novel inflections related to ATS terminology use, while the remaining 17 where novel forms of general words. We interpret this as *some* evidence that the NMT model, when needed, generates novel wordforms. The remaining 27 or 37.5% were not valid, albeit sometimes plausible, Latvian language words, common types of errors being literal translations and transliterations of English words as well as words that would have been correct, if not for errors with consonant mutation.

5 Conclusions

We proposed TLA—a flexible and easy-to-implement method for terminology integration in NMT. Using TLA does not require access to bilingual terminology resources at system training time as it annotates ordinary words with lemmas of their target language translations. This simplifies data preparation greatly and also relaxes the requirement for apriori specified target language forms during the translation, making our method practically viable for terminology translation in real-life scenarios. Results from experiments on three morphologically complex languages demonstrated substantial and systematic improvements over the baseline NMT systems without means for terminology integration and the previous work both in terms of automatic and human evaluation judging term and overall translation quality.

Acknowledgements

This research has been supported by the ICT Competence Centre (www.itkc.lv) within the project "2.2. Adaptive Multimodal Neural Machine Translation" of EU Structural funds, ID no 1.2.1.1/18/A/003.

References

Toms Bergmanis. 2020. *Methods for morphology learning in low(er)-resource scenarios*. Ph.D. thesis, The University of Edinburgh.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Duane K. Dougal and Deryle Lonsdale. 2020. Improving NMT quality using terminology injection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4820–4827, Marseille, France. European Language Resources Association.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Miriam Exel, Bianka Buschbeck, Lauritz Brandt, and Simona Doneva. 2020. Terminology-constrained neural machine translation at SAP. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 271–280, Lisboa, Portugal. European Association for Machine Translation.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural machine translation decoding with terminology constraints. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512, New Orleans, Louisiana. Association for Computational Linguistics.

Felix Hieber, Tobias Domhan, Michael Denkowski, and David Vilar. Sockeye 2: A toolkit for neural machine translation. In *22nd Annual Conference of the European Association for Machine Translation*, page 457.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

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

Mārcis Pinnis. 2018. Tilde's parallel corpus filtering methods for wmt 2018. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 939–945.

Mārcis Pinnis, Andrejs Vasiļjevs, Rihards Kalniņš, Roberts Rozis, Raivis Skadiņš, and Valters Šics. 2018. Tilde MT platform for developing client specific MT solutions. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324.

Lutz Prechelt. 1998. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Justus J Randolph. 2005. Free-marginal multirater kappa (multirater κfree): An alternative to fleiss' fixed-marginal multirater kappa. In *Presented at the Joensuu Learning and Instruction Symposium*, volume 2005.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

Kai Song, Kun Wang, Heng Yu, Yue Zhang, Zhongqiang Huang, Weihua Luo, Xiangyu Duan, and Min Zhang. 2020. Alignment-enhanced transformer for constraining nmt with pre-specified translations. AAAI.

Raymond Hendy Susanto, Shamil Chollampatt, and Liling Tan. 2020. Lexically constrained neural machine translation with levenshtein transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3536–3543.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Enhancing Sequence-to-Sequence Neural Lemmatization with External Resources

Kirill Milintsevich
Institute of Computer Science
University of Tartu
Tartu, Estonia
`kirill.milintsevich@ut.ee`

Kairit Sirts
Institute of Computer Science
University of Tartu
Tartu, Estonia
`kairit.sirts@ut.ee`

Abstract

We propose a novel hybrid approach to lemmatization[1] that enhances the seq2seq neural model with additional lemmas extracted from an external lexicon or a rule-based system. During training, the enhanced lemmatizer learns both to generate lemmas via a sequential decoder and copy the lemma characters from the external candidates supplied during run-time. Our lemmatizer enhanced with candidates extracted from the Apertium morphological analyzer achieves statistically significant improvements compared to baseline models not utilizing additional lemma information, achieves an average accuracy of 97.25% on a set of 23 UD languages, which is 0.55% higher than obtained with the Stanford Stanza model on the same set of languages. We also compare with other methods of integrating external data into lemmatization and show that our enhanced system performs considerably better than a simple lexicon extension method based on the Stanza system, and it achieves complementary improvements w.r.t. the data augmentation method.

1 Introduction

State-of-the-art lemmatization systems are based on attentional sequence-to-sequence neural architectures operating on characters that transform the surface word form into its lemma (Kanerva et al., 2018; Qi et al., 2018). Like any other supervised learning model, these systems are dependent on the amount and quality of the existing training data. Attempts to develop even more accurate lemmatization systems can focus on improving the model's architecture or obtaining additional data. While annotating additional data is an ongoing process for many smaller languages in the Universal Dependencies (UD) collection, there are also other data sources available that can be useful for improving lemmatization systems. In particular, we refer to existing rule-based morphological analyzers, lexicons, and other such resources.

Three potential sources for extracting additional lemma candidates are Apertium, Unimorph, and UD Lexicons initiatives. Apertium[2] is an open-source rule-based machine translation platform (Forcada et al., 2011). It also includes rule-based morphological analyzers based on finite-state transducers that cover 80 languages. Unimorph[3] is a project aimed at collecting annotated morphological inflection data, including lemmas, from Wiktionary (Kirov et al., 2016), a free open dictionary for many languages. Currently, the Unimorph project covers 110 languages. UD Lexicons[4] is a collection of 53 morphological lexicons in CoNLL-UL format covering 38 languages. UD Lexicons mostly use Apertium and Giellatekno systems to generate the annotations (Sagot, 2018).

Several previous works have proposed methods to improve lemmatization systems by augmenting the training data with additional instances (Bergmanis and Goldwater, 2019; Kanerva et al., 2020). In this paper, we propose another approach that both modifies the model architecture and leverages additional data. Unlike previous work where the model gains from extracting extra knowledge from the additional data provided for training, our primary goal is to teach the model to use external resources, even those that may only be available later during test time. In particular, the proposed system is a dual-encoder model, which receives two inputs for each word: 1) the word form itself to be lemmatized and 2) (optionally) the lemma candidates for that word form extracted from a lexicon or generated by a rule-based system. Both inputs are encoded

---

[1] `https://github.com/501Good/lexicon-enhanced-lemmatization`

[2] `https://www.apertium.org`

[3] `http://unimorph.org/`

[4] `http://atoll.inria.fr/~sagot/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3112–3122
April 19 - 23, 2021.

with two different encoders and passed to the decoder. The decoder then learns via two separate attentional mechanisms to generate the lemma via the combination of the regular transduction and by copying characters from the external candidates. This way, the model is trained to use two sources of information–the regular training set and the options proposed by an external resource.

The experiments with several models enhanced with external data on 23 UD languages show that the best model using additional lemma candidates generated by the Apertium system achieves significantly higher results than the baseline models trained on the UD training set only. Also, we compare our method with other methods using external data. The enhanced system performs considerably better than a simple lexicon extension method based on the Stanza system, and it achieves complementary improvements w.r.t. the data augmentation method of Kanerva et al. (2020).

2 Related Works

Nowadays, state-of-the-art lemmatization systems are typically based on a neural sequence-to-sequence architecture, as demonstrated by the variety of systems presented at the CoNLL 2018 (Zeman et al., 2018) and SIGMORPHON 2019 (McCarthy et al., 2019) shared tasks. Several systems, including the TurkuNLP pipeline, the winner of the lemmatization track at CoNLL 2018 Shared task, use an attention-based translation model (Kanerva et al., 2018; Qi et al., 2018). The input to the system is the character sequence of a surface form (SF), which is "translated" into the lemma by an attention-based decoder. The input sequence can also be extended with POS tags (Qi et al., 2018) and morphological features (Kanerva et al., 2018).

Another approach was used by the UDPipe Future system, the second-best model at the CoNLL 2018 Shared Task. Straka (2018) proposed to produce a lemma by constructing a set of rules that transform the SF into a lemma. These rules can include copying, moving, or deleting a character in the SF, as well as additional rules for changing or preserving the casing. Thus, the lemmatization task is rendered into a multi-class classification task of choosing the correct transformation rule among the set of all possible rules generated from the training set. A year later, Straka et al. (2019) improved the result for the lemmatization by adding BERT contextual embeddings (Devlin et al., 2019) to the input, which made them the best lemmatization system at the SIGMORPHON 2019 Shared Task.

Several previous works have proposed to leverage additional data to improve lemmatization. In the simplest form, training data itself can be used to create a lexicon that maps word forms to its lemma. This strategy has been adopted by the Stanford neural lemmatization system (Qi et al., 2018), which creates such lexicons from the training sets and resorts to lemma generation only when the lexicon lookup fails. One can easily imagine extending such a lexicon with external resources. Rosa and Mareček (2018) adopted another simple way of using Unimorph lexicons to post-fix the morphological features and lemmas predicted by the UDPipe system (Straka and Straková, 2017). The post-fix is performed by simply looking up the SF from the Unimorph lexicon and, if the match is found, replacing the model prediction with the tags and lemmas found in the lexicon.

Another line of work has used additional data to augment the training data set. Bergmanis and Goldwater (2019) augmented their training set by first listing all non-ambiguous word-lemma pairs from Unimorph lexicons and then extracted sentences from Wikipedia that contained these words. They then trained the context-sensitive Lematus model (Bergmanis and Goldwater, 2018) on this extended partially lemmatized data set. Kanerva et al. (2018) used Apertium's morphological analyzer module to extend the training set for languages with tiny UD datasets. Apertium was used to generate all possible morphological analyses to 5000 sentences selected from the Wikipedia of the respective language. For each sentence, the most likely analysis sequence was then obtained via a disambiguating language model. The words that were assigned an Apertium-generated lemma during this process were added to the lemmatizer training set. In the subsequent work, Kanerva et al. (2020) extended the training data even more. They used Apertium to analyze all words found in the CoNLL 2017 web crawl dataset (Ginter et al., 2017) or in the Wikipedia of the respective language. All new words with unambiguous lemma and morphological analysis were added to the augmented training set.

3 Method

The core of the proposed model is the Stanford lemmatizer (Qi et al., 2018, 2020) which is a sequence-to-sequence model with attention. It takes character-

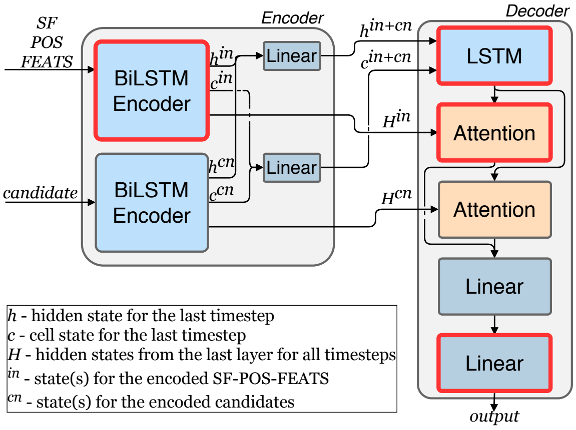


Figure 1: The architecture of the dual-encoder enhanced lemmatizer. Layers that comprise the original Stanza lemmatizer are marked with a bold red border.

level word representation and the POS tag as input and processes them with a bidirectional LSTM encoder. Then, it passes the encoder outputs to an LSTM decoder, which applies a soft dot attention layer after every LSTM cell. Finally, the output is constructed via greedy decoding.

We make several changes to the model architecture as shown in Figure 1. The components comprising the original Stanford lemmatizer are marked on the figure with the bold red border. First, we add another encoder that encodes the lemma candidates provided by the external system. The output representations of both encoders are combined with a linear layer and fed to the decoder. Secondly, we add another attention layer to the decoder that attends to the outputs of the second encoder. The outputs are finally combined with a linear layer. Finally, in addition to the POS tag, we also add morphological features to the first encoder's input.

Additionally, we implement the encoder dropout to simulate the situation when the external candidates are absent. The value of the encoder dropout that varies in the range of $\{0.0, 1.0\}$ defines the probability of discarding all candidates from a batch during training. Thus, the model will train only the main encoder based on this batch. This helps to train the model to perform more robustly in both situations when the candidates in the second encoder are present or absent.

4 Experiments

Data The models are trained and tested on the Universal Dependencies (UD) v2.5 corpora (Zeman et al., 2019). As additional external data, the lexicons from the Unimorph project (Kirov et al., 2016), UD Lexicons (Sagot, 2018), and lemmas generated with the Apertium morphological analyzer module (Forcada et al., 2011) are used. We also experiment with the lexicon constructed from the training set to simulate the situation when no additional data is available—this scenario assesses the effect of the second encoder without external data. The experiments are conducted on 23 languages from the UD collection. The basis of this selection was that all these languages are supported by both Unimorph, UD Lexicons, and Apertium.

To extract lemmas from the Unimorph lexicon, the input surface form (SF) is queried from the lexicon to retrieve the corresponding lemma. Some morphological forms in the Unimorph lexicons consist of several space-separated tokens; these were discarded. UD Lexicons are presented in the CoNLL-UL format, which is an extension of the CoNLL-U format. This makes the extraction process trivial since the lexicons are already pre-tokenized. For Apertium, all generated lemmas were stripped from special annotation symbols, and duplicate lemmas were removed. Finally, the simple training set based lexicon solution, similar to Qi et al. (2018), consists of two lookup dictionaries. The first lexicon maps SF-POS pairs to their lemmas, the second lexicon maps just SF's to their possible lemmas found in the training set. The lemma candidates for a SF are selected by first querying the input SF and POS tag from the SF-POS dictionary and, in case of failure, falling back to the SF dictionary.

Baselines As the first baseline, we compare our results with STANZA, the lemmatization module from the Stanza pipeline (Qi et al., 2020), which is a repackaging of the Stanford lemmatization system from the CoNLL 2018 Shared Task (Qi et al., 2018). We used the lemmatization models trained on the UDv2.5 available on the Stanza web page. As the DEFAULT baseline, we use our enhanced model, with the second encoder always being empty.

Experimental Setup We train four enhanced dual-encoder models that differ in the input to the second encoder. For all models, the input to the first encoder is the concatenation of SF characters, POS tag, and morphological features. During the training phase, gold POS tags and morphological features are supplied, while during inference, POS tags predicted with the Stanza tagger are used. The input to the second encoder is the following: for the second baseline (DEFAULT), it is always empty; for

Treebank	Size	All words					Out-of-vocabulary			
		DEF	LEX	UNI	APT	STANZA	DEF	APT	Diff	OOV%
cs_pdt	1,503K	98.51	98.66	**98.67**	98.55	98.58	90.51	90.95	0.44	7.53
ru_syntagrus	1,107K	97.82	97.92	98.00	**98.11**	97.91	89.48	91.65	2.17	10.56
es_ancora	547K	99.31	99.28	99.31	**99.35**	99.21	95.06	95.40	0.34	5.90
ca_ancora	530K	98.85	98.83	98.83	**98.89**	98.49	95.82	96.79	0.97	5.43
fr_gsd	389K	98.05	98.07	98.10	98.13	**98.15**	89.50	90.83	1.33	6.19
hi_hdtb	351K	98.77	98.71	98.71	**98.80**	96.66	93.49	94.62	1.13	4.67
de_gsd	287K	96.87	96.91	**97.04**	96.80	96.78	85.53	85.22	-0.31	13.04
it_isdt	278K	98.19	98.39	98.31	**98.48**	98.32	90.28	92.22	1.94	5.86
en_ewt	254K	98.21	98.19	98.22	**98.26**	98.18	90.10	90.49	0.39	10.05
ro_rrt	218K	98.33	98.28	98.32	**98.53**	98.16	91.46	93.22	1.76	11.60
pt_bosque	210K	98.24	98.20	98.23	**98.32**	98.12	93.15	94.27	1.12	8.85
nl_alpino	208K	**97.08**	96.61	96.89	96.74	96.99	86.34	84.88	-1.46	15.81
bg_btb	156K	97.97	**98.20**	98.17	98.07	97.36	91.07	91.02	0.05	13.97
ur_udtb	138K	97.16	**97.29**	97.28	97.28	95.62	91.83	91.93	0.01	6.79
gl_ctg	126K	98.48	98.48	98.51	**98.93**	98.59	89.73	93.55	3.82	10.94
uk_iu	122K	97.03	97.07	97.06	**97.12**	96.70	91.15	91.32	0.17	33.62
eu_bdt	121K	96.48	96.62	96.63	**96.68**	96.52	86.18	86.81	0.63	21.68
da_ddt	100K	97.87	97.7	97.81	**98.03**	97.36	89.86	90.31	0.45	18.13
sv_talbanken	96K	97.36	97.59	97.64	**98.27**	97.53	87.66	92.33	4.67	17.52
el_gdt	61K	96.84	97.06	97.25	**97.38**	96.66	84.18	86.42	2.24	19.59
tr_imst	56K	97.03	97.23	97.13	**97.39**	96.73	92.27	93.18	0.91	36.25
hy_armtdp	52K	95.55	95.84	94.87	**96.01**	95.55	86.11	87.34	1.23	38.54
be_hse	13K	81.91	81.86	82.36	**82.63**	79.98	68.78	70.30	1.52	93.28
Average		97.04	97.09	97.10	**97.25**	96.70	89.11	90.22	1.11	

Table 1: Lemmatization accuracy of the models enhanced with training the set lexicon (LEX), Unimorph lexicon (UNI), and Apertium systems (APT) as well as the DEFAULT (DEF)and STANZA baselines on 23 UD languages.

the LEXICON, UNIMORPH, and APERTIUM enhanced models, it contains the lemma candidate(s) from the training set based lexicon, Unimorph lexicons, and Apertium analyses respectively. If several possible candidates are returned for a SF, then these are concatenated. The encoder dropout for the LEXICON model is set to 0.8 to simulate the situation during testing for out-of-vocabulary (OOV) words where the second encoder will be empty. All models were trained in the HPC at the University of Tartu (University of Tartu, 2018) for a maximum of 60 epochs with stopping early if there was no improvement in the development accuracy in 10 epochs.

5 Results

Table 1 shows the results for all three enhanced systems and two baselines. The APERTIUM model outperforms other models for most languages, although the absolute differences are quite small. The LEXICON model and the DEFAULT baseline are on the same level on average, suggesting that supplying the model with lemmas extracted from the training set via the second encoder does not help to leverage the training data better. However, all enhanced models, including the DEFAULT model, perform better than the STANZA baseline, suggesting that omitting the lexicon heuristics and supplying the input tokens with both POS and morphological features might improve performance.

One-way ANOVA was performed to detect statistical difference between the systems.[5] A significant difference between the scores at the $p < 0.05$ level ($p = 0.038$) was found. Post hoc comparisons using one-sided paired t-tests showed that the mean accuracy of the APERTIUM-enhanced model is significantly greater compared to the the DEFAULT ($p_{adj} = 0.0005$), LEXICON

---

[5] The results for be_hse were extreme outliers and were not included in the comparison. The UNIMORPH-enhanced model was excluded from this test as its results did not conform to the normality requirement.

($p_{adj} < 0.0001$), UNIMORPH ($p_{adj} = 0.0001$) and STANZA ($p_{adj} < 0.0001$) systems with the p-value adjusted for multiple comparisons using the Bonferroni correction.

As the baseline model performances are already very high and the external information is expected to improve the lemmatization most for the new words unseen during training, we computed the accuracy of the out-of-vocabulary words (OOV) for the best performing APERTIUM model and the DEFAULT baseline. In this context, OOV words are those words in the test set that were not seen by the model during training. The results are shown in the right-most section of the Table 1. The improvements on the OOV words are variable, depending on the language, although on average, the improvement of the APERTIUM model over the DEFAULT baseline is more than 1%. We hypothesize that the direction and the magnitude of these effects are dependent on the coverage and the quality of the Apertium morphological analyzer.

6 Analysis of the Results

In this section, we analyze more thoroughly the potential of the proposed method. First, we compare our enhanced system with alternative methods for deploying external data, particularly with the data augmentation method proposed by Kanerva et al. (2020) and a lexicon extension method implemented based on the Stanza system (Qi et al., 2020). Secondly, we present more analyses to provide evidence towards the conclusion that the improvements presented for the enhanced model in the previous section can be attributed to our system's ability to make use of external resources supplied to the model via the second encoder.

6.1 Data Augmentation

We implemented the transducer augmentation method described by Kanerva et al. (2020). This method's basic idea relies on applying existing morphological analyzers (in this case, Apertium) to unannotated data to generate additional training instances. To obtain the augmentation data, we recreated the experiments of Kanerva et al. (2020) with 8K additional data. First, we collected a word frequency list for each language based on automatically annotated CoNLL2017 corpora (Ginter et al., 2017). For the languages not present in this dataset (Belarusian and Armenian), we used the wikidump to extract the word frequency list. Next, all words in the list were analyzed with the Apertium morphological analyzer. Then, we used the scripts[6] from the original experiments of Kanerva et al. (2020) to convert the Apertium analyses to the UD format and filter out ambiguous cases. Finally, the 8K most frequent words not already present in the training set together with their analyses were chosen and appended to the UD training set.

Although both the enhanced and augmented systems utilize Apertium as the external source, additional data usage differs. The augmented system uses Apertium to create extra labeled training data, while our enhanced model uses Apertium to generate additional lemma candidates to the words of the same initial training set. On the other hand, during test time, the augmented model must fully rely on the regularities learned during training, while our enhanced model can additionally look at the lemmas for words that were never seen during training.

The comparison of our APERTIUM-enhanced model and the augmented model is shown in the first two blocks of Table 2. The first two columns reintroduce the DEFAULT and APERTIUM-enhanced models' results from the Table 1, the third and the fourth columns show the same two models trained on the augmented training sets. Overall, the average results for both APERTIUM-enhanced and the augmented DEFAULT model (the column DEF+8K) are very similar, with the average of the APERTIUM-enhanced model being slightly higher (97.25 vs. 97.17). The APERTIUM-enhanced model is better in 15 languages out of 23 (underlined in the table), while the augmented model surpasses the enhanced model on 8 models. The APT+8K column shows the results of a model combining both augmentation and enhanced methods—the training data is first augmented with the additional 8K words and then additionally enhanced with the Apertium candidates via the second encoder. The combined approach scores are the best for 8 languages out of 23, resulting in an average improvement over the augmented DEFAULT model of 0.14% and over the APERTIUM-enhanced model of 0.06% in absolute. These results show that both augmentation and enhancement methods can contribute in complementary ways.

6.2 Lexicon Extension

Another simple baseline method for using external data is to use a lexicon or an external system first

---

[6] `https://github.com/jmnybl/universal-lemmatizer`

Treebank	**Def**	**Apt**	**Def+8K**	**Apt+8K**	**$\text{Apt}_{0.8}$**	**Apt+E**	**Apt+Uni**	**Apt+UD**
	Our models		Augmented models		The second encoder input varies			
cs_pdt	98.51	98.55	98.49	**98.57**	98.49	98.39	98.51	98.50
ru_syntagrus	97.82	**98.11**	97.86	98.06	97.98	97.83	97.98	97.97
es_ancora	99.31	99.35	99.53	**99.60**	99.33	99.29	99.33	99.33
ca_ancora	98.85	**98.89**	98.86	**98.89**	98.85	98.80	98.85	98.85
fr_gsd	98.05	98.13	98.98	**99.05**	97.98	97.79	97.97	97.98
hi_hdtb	98.77	98.80	**98.83**	98.78	98.84	98.66	98.83	98.84
de_gsd	**96.87**	96.80	96.79	96.67	96.83	96.49	96.83	96.84
it_isdt	98.19	98.48	98.98	**98.99**	98.36	98.3	98.36	98.37
en_ewt	98.21	**98.26**	97.24	98.12	98.21	98.17	98.22	98.20
ro_rrt	98.33	**98.53**	97.56	98.48	98.44	98.29	98.46	98.41
pt_bosque	98.24	**98.32**	98.13	98.29	98.30	98.32	98.30	98.31
nl_alpino	97.08	96.74	96.80	**96.82**	96.89	96.86	96.81	96.85
bg_btb	97.97	98.07	**98.84**	98.82	98.02	98.06	98.02	98.02
ur_udtb	97.16	97.28	96.90	**97.31**	97.13	97.13	97.13	97.13
gl_ctg	98.48	**98.93**	98.27	98.84	98.74	97.02	98.74	98.70
uk_iu	97.03	97.12	97.25	**97.35**	97.22	97.11	97.22	97.22†
eu_bdt	96.48	96.68	96.66	**96.71**	96.63	96.33	96.63	96.62
da_ddt	97.87	**98.03**	97.74	97.95	97.87	97.57	97.91	97.87
sv_talbanken	97.36	**98.27**	97.49	98.16	98.41	97.64	97.84	97.95
el_gdt	96.66	**97.38**	97.02	96.96	97.49	97.38	97.56	97.47
tr_imst	97.03	**97.39**	97.01	97.24	97.17	96.89	97.17	97.14
hy_armtdp	95.55	**96.01**	95.74	95.66	95.86	95.68	95.86	95.86†
be_hse	81.91	82.63	**83.33**	82.92	83.51	82.13	83.51	83.51†
Average	97.03	97.25	97.17	**97.31**	97.24	96.96	97.22	97.21

Table 2: Comparison of the enhanced models with the augmentation method: Def is the Default model, Apt is the Apertium-enhanced model, Def+8K and Apt+8K are the same Default and Apertium-enhanced models with augmented data. For the models marked with †, the UD Lexicon is absent and is replaced with Apertium candidates instead.

and only resort to neural generation when the surface form (SF) is not present in the lexicon. This is essentially how the Stanza lemmatizer works. Stanza constructs a lexicon based on the training set. During inference, the prediction goes through a cascade of three steps: 1) if the SF is present in the lexicon, then the lemma is immediately retrieved from the lexicon. 2) If the SF is novel and is missing from the lexicon, an edit operation is generated that decides whether the SF itself or its lowered form is the lemma, or whether neither is true. 3) Only in the last case the lemma is generated by the sequential decoder. For testing out the lexicon extension system, we used the pretrained Stanza models but extended the lexicon stored in the Stanza system with additional items. Note that Stanza lexicons can only store one lemma per SF-POS combination. Thus, if any of the external lexicons contain ambiguous lemmas, the firstly encountered lemma is chosen for each word.

We extended the Stanza lexicons with both the Apertium 8K datasets used for training the augmented models in section 6.1 and the UD lexicons (Sagot, 2018). The results of these evaluations are shown in Table 3. The set of languages in this table is slightly different than in Table 1, only including those languages for which the UD lexicons are existent. The left block shows the results with various Stanza models. The first column shows the baseline Stanza results (taken from Table 1), the second and the third columns present the Stanza model with its lexicon extended with the UD lexicons and the 8K words, respectively. The original UD lexicon for Russian contained many erroneous lemmas due to poor post-processing, which skewed the average accuracy. Thus, we did additional post-processing to put it in line with other languages.

The average scores of the Stanza systems ex-

Treebank	Stanza	Stanza+UD	Stanza+8K	Apt	Lex+UD	Lex+8K
cs_pdt	98.58	**98.76**	98.60	98.49	98.70	98.66
ru_syntagrus	97.91	96.76†	97.92	**97.98**	97.36†	97.97
es_ancora	99.21	99.25	99.15	**99.33**	99.27	99.28
ca_ancora	98.49	98.29	98.51	**98.85**	98.83	98.83
fr_gsd	98.15	97.69	**98.24**	97.98	97.09	96.75
hi_hdtb	96.66	96.75	96.66	**98.84**	98.76	98.71
de_gsd	**96.78**	97.53	96.86	96.83	97.01	96.94
it_isdt	98.32	**98.60**	98.46	98.36	98.38	98.39
en_ewt	98.18	**98.21**	98.17	**98.21**	**98.21**	98.19
ro_rrt	98.16	**98.44**	98.27	**98.44**	98.38	98.28
pt_bosque	98.12	**98.32**	98.12	98.30	97.98	98.20
nl_alpino	96.99	**97.22**	96.97	96.89	96.63	96.61
bg_btb	97.36	96.26	96.62	98.02	98.12	**98.20**
ur_udtb	95.62	95.66	95.64	97.13	97.28	**97.29**
gl_ctg	98.59	98.64	98.60	**98.74**	98.48	98.48
eu_bdt	96.52	96.51	96.41	96.63	**96.66**	96.62
da_ddt	97.36	**97.89**	97.55	97.87	97.82	97.70
sv_talbanken	97.53	**98.45**	97.63	98.41	97.78	97.59
el_gdt	96.66	96.49	96.89	97.49	97.52	**97.54**
tr_imst	96.73	96.90	96.83	97.17	97.17	**97.23**
Average	97.60	97.63	97.61	**98.00**	97.87	97.87

Table 3: Evaluation of the effect of the Stanza-based lexicon extension method; comparison with the Apertium-enhanced (Apt) and the Lexicon-enhanced systems (Lex+UD and Lex+8K).

tended with both UD and 8K lexicons remain roughly the same. However, when extending the Stanza with UD lexicons, most languages improve at least slightly, as shown with the underlined scores in the column Stanza+UD. Overall, on average, the simple lexicon extension method falls considerably behind our Apertium-enhanced model (97.63 vs. 98.00), the scores of which are again replicated in the first column of the right-most block.

However, the Apertium-enhanced model is not directly comparable to the Stanza models with extended lexicons because 1) the training data differs as the enhanced model has access to extra lemma candidates of the training set words during training and 2) the lexicons available during the test time are different. Thus, we also show in the last two columns of the right-hand block of Table 3 the results of two Lexicon-enhanced models (recall Section 4 and Table 1), similarly extended with the UD and 8K lexicons. The Lexicon-enhanced model has access to the same data as the Stanza model during both training (training set + the training set based lexicon) and testing.

While the Lexicon-enhanced model alone does not perform better than the Default baseline (see results in Table 1), adopting additional UD or 8K lexicons during test time increases the results to the same level with the Apertium-enhanced model. This shows that our proposed approach does not need additional resources during training—the model can be trained to use external sources based on the lexicon created from the training set. Then, the system's real benefits can be achieved when using extra resources later during test time. Without those resources, the model still performs on the same level as the non-enhanced baseline.

We hypothesize that our dual-encoder approach performs better than the Stanza with extended lexicon partly because of the differences in the usage of the external data. Since Stanza uses the lexicon resources as a first step in the cascade, it is prone to potential errors and noise in the lexicons. The dual-encoder model is safer against noise in this respect because the lemma candidates are not simply chosen as the prediction if present but are rather fed through the system that can decide how much to take or ignore from the given candidates. Also, because Stanza lexicons have the restriction of only one lemma per word-POS pair, the system might solve some ambiguities erroneously. Our approach is also more flexible in this respect, as the second encoder can be given several candidates, and again, the system learns to decide itself from which candidate how much to take. On average, there are 0.71 lemma candidates per input word, and 1.09 lemma candidates per input word when excluding those words that do not have external lemma candidates.

6.3 Effect of the Second Encoder

Next, we performed a set of evaluations to argue for the effect of the second encoder in the enhanced model. We suggest that the improvements presented in Table 1 for the APERTIUM-enhanced model over the DEFAULT baseline are indeed due to the input provided via the second encoder. To demonstrate that, we evaluated the test set for each language again, on the same model that was trained with Apertium lemma candidates but leaving the second encoder empty for the test time. For that, we retrained the APERTIUM-enhanced models with the encoder dropout of 0.8. This means that during training, 80% of the time, the lemma candidates provided for the second encoder are dropped, and the model trains only the main encoder. The reasoning for using the dropout is similar to one provided for the LEXICON-enhanced model in Section 3—if the lemma candidates are always provided during training, the model learns to rely equally on both encoders. Due to that, if the second encoder remains empty during testing, the performance degrades considerably. If, on the other hand, the dropout is used, then the model learns to make predictions both when the candidates in the second encoder are present and also when they are absent. The results of these experiments are shown in the right-most block of Table 2.

We first show in Table 2 that the results of the APERTIUM-enhanced models trained with dropout are equivalent to the results obtained without dropout as evidenced by the column $\text{APT}_{0.8}$. Next, when the second encoder is empty (column APT+E), the test results are similar to the ones obtained with the DEFAULT model, providing evidence that the improvements are indeed due to the extra info supplied via the second encoder during test time. Additionally, we emulated the scenario when extra lexicon information becomes available after training the model. In this case, it is straightforward to integrate this information into the system without having to retrain the model. The last two columns in Table 2 show the following scenarios on this respect: 1) Unimorph lexicons in addition to Apertium (7th column APT+UNI) and 2) UD lexicons (the last column APT+UD) in addition to Apertium. The results in Table 2 show that, on average, extending the Apertium system with these particular lexicons do not add any benefit. The reasons for that can be twofold: 1) The UD lexicons are for most languages constructed based on the Apertium system and thus might not add any extra information; 2) The coverage of Unimorph lexicons in terms of lemmas is typically smaller than of Apertium systems.

Table 4 shows some examples when the DEFAULT model predicted incorrect lemma while the APERTIUM-enhanced model predicted the correct one. In some cases, Apertium provided the only and correct candidate for the APERTIUM-enhanced model, which was picked as a final prediction. In other cases, several candidates are provided to the second encoder, and the enhanced model chooses the correct one in most of the cases. This indicates that the second encoder effectively learns how to use the candidates to better control the lemma generation.

6.4 Effect of Morphological Features

All dual-encoder models were trained with both POS and morphological features in the input, while the STANZA baseline only uses POS-tag information. Thus, the effect of the morphological features is a potential confounding factor when comparing the performance of the enhanced models to the STANZA baseline. To evaluate the effect of the morphological features, we trained the DEFAULT and APERTIUM-enhanced models with only providing POS-tag information to the input.

Figure 2 shows the improvement in accuracy over the DEFAULT model trained with POS-tags only of 1) the DEFAULT model trained with both POS-tags and morphological features, 2) the APERTIUM-enhanced model trained with only POS-tags, and 3) the APERTIUM-enhanced model trained with both POS-tags and morphological features. It can be seen that for some of the languages, the most improvement comes from adding morphological features to the input, while for other languages adding the second encoder gives the main boost. However, for most languages, combining the second encoder and morphological features provides the largest effect, which seems to be more complex than a linear combination of the two. We suppose that, in this scenario, the attention mechanism

Input	Def	Apt	Candidate(s)
папері ⟨*paperi*⟩	*папер ⟨*paper*⟩	**папір** ⟨*papir*⟩	папір ⟨*papir*⟩
чотирьох ⟨*čotyr’oh*⟩	*четвери ⟨*četvery*⟩	**чотири** ⟨*čotyry*⟩	четверо, чотири ⟨*četvero, čotyry*⟩
Antworten	*Antworte	**Antworten**	antworten, antwort
besten	bester	**gut**	gut
раскладзе ⟨*raskladze*⟩	*раскладз ⟨*raskladz*⟩	**расклад** ⟨*rasklad*⟩	раскласці, расклад ⟨*rasklasci, rasklad*⟩
стаіць ⟨*staic’*⟩	стаіць ⟨*staic’*⟩	**стаяць** ⟨*stajac’*⟩	стаяць, стаіць ⟨*stajac’, staic’*⟩

Table 4: Examples for Ukrainian, German, and Belarusian words corrected by the enhanced model. All predictions of the DEFAULT (DEF) are incorrect, the ungrammatical ones are marked with *. The correct predictions of the APERTIUM-enhanced (APT) models are in **bold**. The last column shows the external candidates.

works differently—it allegedly takes the morphological features into account when picking the correct lemma from the multiple candidates.

7 Conclusion

We proposed a method for enhancing neural lemmatization by integrating external input into the model via a second encoder and showed that the system incorporating Apertium morphological analyzer significantly improved the performance over the baselines. Both Bergmanis and Goldwater (2019) and Kanerva et al. (2020) used external resources to augment the training data, and thus, the improvement of their system is dependent on the amount and quality of the extended data supplied during training. On the other hand, our method trains the system to use the external information provided during run-time, thus making it independent of the particular external data available during training.

We experimentally showed that the enhancing method is both slightly better and complementary to the data augmentation method of Kanerva et al. (2020). We also compared our system with a simple lexicon extension method implemented based on the Stanza system. When trained and tested in a comparable setting, the proposed enhanced system achieves considerably higher results.

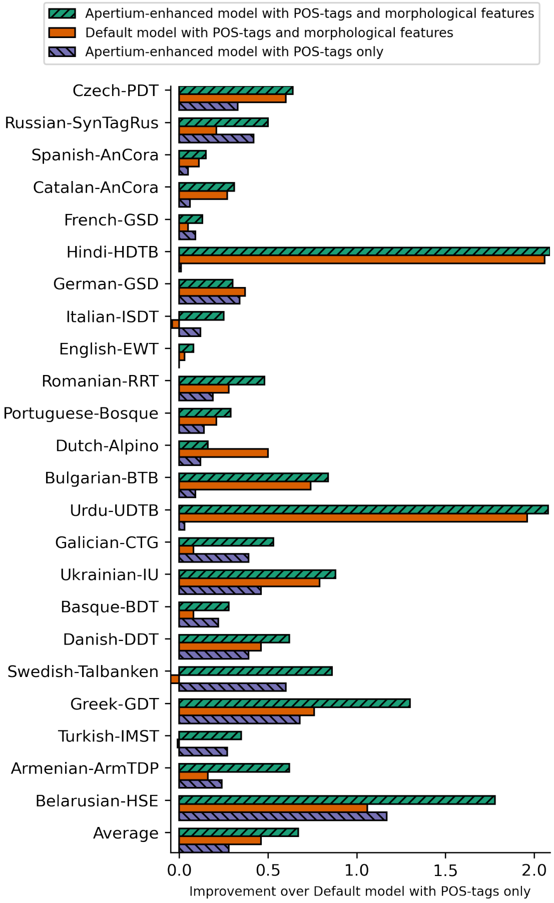


Figure 2: Independent and cumulative effects of the second encoder and the morphological features on the model's performance. The origin of the x-axis is the performance of the DEFAULT model with POS-tags only.

Although the model's computational complexity is increased by introducing the second encoder, it is counterbalanced by our model being more robust to noise and the ambiguities stemming from the external lexicons. Moreover, the main bottleneck in computation originates not from the neural network's increased size but can rather stem from the external system. For example, in our experiments, the main bottleneck in computation originated from executing the transducer-based Apertium morphological analyser. To overcome this bottleneck, one possible trade-off between the speed and accuracy is to precompile a candidate list large enough to cover the most frequent words for a given language. This is a problem that also simpler baseline methods adopting external resources have to address.

Finally, it is worth noting that the proposed method could be beneficial for less-resourced languages. However, establishing this claim would need more systematic experiments exploring specifically on this question, which we did not focus on in this paper. Still, because the significant improvements shown in this work are obtained on languages with larger datasets, the possible gains on smaller datasets can be larger.

Acknowledgments

The first author was supported by the IT Academy Program (StudyITin.ee).

References

Toms Bergmanis and Sharon Goldwater. 2018. Context Sensitive Neural Lemmatization with Lematus. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1391–1400.

Toms Bergmanis and Sharon Goldwater. 2019. Training Data Augmentation for Context-Sensitive Neural Lemmatizer Using Inflection Tables and Raw Text. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4119–4128.

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

Mikel L. Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. 2011. Apertium: A Free/Open-Source Platform for Rule-Based Machine Translation. *Machine translation*, 25(2):127–144.

Filip Ginter, Jan Hajic, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 Shared Task–Automatically Annotated Raw Texts and Word Embeddings. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Jenna Kanerva, Filip Ginter, Niko Miekka, Akseli Leino, and Tapio Salakoski. 2018. Turku Neural Parser Pipeline: An End-to-End System for the CoNLL 2018 Shared Task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 133–142.

Jenna Kanerva, Filip Ginter, and Tapio Salakoski. 2020. Universal Lemmatizer: A Sequence to Sequence Model for Lemmatizing Universal Dependencies Treebanks. *Natural Language Engineering*, pages 1–30.

Christo Kirov, John Sylak-Glassman, Roger Que, and David Yarowsky. 2016. Very-large Scale Parsing and Normalization of Wiktionary Morphological Paradigms. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, et al. 2019. The SIGMORPHON 2019 Shared Task: Morphological Analysis in Context and Cross-Lingual Transfer for Inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. 2018. Universal Dependency Parsing from Scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Rudolf Rosa and David Mareček. 2018. CUNI x-ling: Parsing Under-Resourced Languages in CoNLL 2018 UD Shared Task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 187–196.

Benoît Sagot. 2018. A Multilingual Collection of CoNLL-U-Compatible Morphological Lexicons. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Milan Straka. 2018. UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207.

Milan Straka and Jana Straková. 2017. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Milan Straka, Jana Straková, and Jan Hajic. 2019. UDPipe at SIGMORPHON 2019: Contextualized Embeddings, Regularization with Morphological Categories, Corpora Merging. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 95–103.

University of Tartu. 2018. UT Rocket. share.neic.no.

Daniel Zeman, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 1–21.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, and et al. 2019. Universal Dependencies 2.5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Summarising Historical Text in Modern Languages

Xutan Peng[✊] **Yi Zheng**[✌] **Chenghua Lin**[✊][*] **Advaith Siddharthan**[✋]
[✊]Department of Computer Science, The University of Sheffield, UK
[✌]School of Computer Science and Engineering, Beihang University, China
[✋]Knowledge Media Institute, The Open University, UK
`{x.peng, c.lin}@shef.ac.uk` `zhengyi53@buaa.edu.cn`
`advaith.siddharthan@open.ac.uk`

Abstract

We introduce the task of historical text summarisation, where documents in historical forms of a language are summarised in the corresponding modern language. This is a fundamentally important routine to historians and digital humanities researchers but has never been automated. We compile a high-quality gold-standard text summarisation dataset, which consists of historical German and Chinese news from hundreds of years ago summarised in modern German or Chinese. Based on cross-lingual transfer learning techniques, we propose a summarisation model that can be trained even with no cross-lingual (historical to modern) parallel data, and further benchmark it against state-of-the-art algorithms. We report automatic and human evaluations that distinguish the historic to modern language summarisation task from standard cross-lingual summarisation (i.e., modern to modern language), highlight the distinctness and value of our dataset, and demonstrate that our transfer learning approach outperforms standard cross-lingual benchmarks on this task.

1 Introduction

The process of text summarisation is fundamental to research into history, archaeology, and digital humanities (South, 1977). Researchers can better gather and organise information and share knowledge by first identifying the key points in historical documents. However, this can cost a lot of time and effort. On one hand, due to cultural and linguistic variations over time, interpreting historical text can be a challenging and energy-consuming process, even for those with specialist training (Gray et al., 2011). To compound this, historical archives can contain narrative documents on a large scale, adding to the workload of manually locating important elements (Gunn, 2011). To reduce these burdens, specialised software has been developed recently, such as MARKUS (Ho and Weerdt, 2014) and DocuSky (Tu et al., 2020). These toolkits aid users in managing and annotating documents but still lack functionalities to automatically process texts at a semantic level.

Historical text summarisation can be regarded as a special case of cross-lingual summarisation (Leuski et al., 2003; Orăsan and Chiorean, 2008; Cao et al., 2020), a long-standing research topic whereby summaries are generated in a target language from documents in different source languages. However, historical text summarisation posits some unique challenges. Cross-lingual (i.e., across historical and modern forms of a language) corpora are rather limited (Gray et al., 2011) and therefore historical texts cannot be handled by traditional cross-lingual summarisers, which require cross-lingual supervision or at least large summarisation datasets in both languages (Cao et al., 2020). Further, language use evolves over time, including vocabulary and word spellings and meanings (Gunn, 2011), and historical collections can span hundreds of years. Writing styles also change over time. For instance, while it is common for today's news stories to present important information in the first few sentences, a pattern exploited by modern news summarisers (See et al., 2017), this was not the norm in older times (White, 1998).

In this paper, we address the long-standing need for historical text summarisation through machine summarisation techniques for the first time. We consider the German$_{|\text{DE}}$ and Chinese$_{|\text{ZH}}$ languages, selected for the following reasons. First, they both have rich textual heritage and accessible (monolingual) training resources for historical and modern language forms. Second, they serve as outstanding representatives of two distinct writing systems (DE

---

[*] Chenghua Lin is the corresponding author.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3123–3142
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

DE	№34
Story	Jhre Königl. Majest. befinden sich noch vnweit Thorn / ... / dahero zur Erledigung Hoffnung gemacht werden will. *(Their Royal Majesties are still not far from Torn, ... , therefore completion of the hope is desired.)*
Summary	Der Krieg zwischen Polen und Schweden dauert an. Von einem Friedensvertrag ist noch nicht der Rede. *(The war between Poland and Sweden continues. There is still no talk on the peace treaty.)*
ZH	№7
Story	有脚夫小民，三四千名集众围绕马监丞衙门，...，冒火突入，捧出敕印。 *(Three to four thousand porters gathered around Majiancheng Yamen (a government office), ..., rushed into fire and salvaged the authority's seal.)*
Summary	小本生意免税条约未能落实，小商贩被严重剥削，以致百姓聚众闹事并火烧衙门，造成多人伤亡。王炀抢救出公章。 *(The tax-exemption act for small businesses was not well implemented and small traders were terribly exploited, leading to riot and arson attack on Yamen with many casualties. Yang Wang salvaged the authority's seal.)*

Table 1: Examples from our HISTSUMM dataset.

for alphabetic and ZH for ideographic languages), and investigating them can lead to generalisable insights for a wide range of other languages. Third, we have access to linguistic experts in both languages, for composing high-quality gold-standard modern-language summarises for DE and ZH news stories published hundreds of years ago, and for evaluating the output of machine summarisers.

In order to tackle the challenge of a limited amount of resources available for model training (e.g., we have summarisation training data only for the monolingual task with modern languages, and very limited parallel corpora for modern and historical forms of the languages), we propose a transfer-learning-based approach which can be bootstrapped even without cross-lingual supervision. To our knowledge, our work is the first to consider the task of historical text summarisation. As a result, there are no directly relevant methods to compare against. We instead implement two state-of-the-art baselines for standard cross-lingual summarisation, and conduct extensive automatic and human evaluations to show that our proposed method yields better results. Our approach, therefore, provides a strong baseline for future studies on this task to benchmark against.

The contributions of our work are three-fold: (1) we propose a hitherto unexplored and challenging task of historical text summarisation; (2) we construct a high-quality summarisation corpus for historical DE and ZH, with modern DE and ZH summaries by experts, to kickstart research in this field; and (3) we propose a model for historical text summarisation that does not require parallel supervision and provides a validated high-performing baseline for future studies. We release our code and data at `https://github.com/Pzoom522/HistSumm`.

2 Related Work

Processing historical text. Early NLP studies for historical documents focus on spelling normalisation (Piotrowski, 2012), machine translation (Oravecz et al., 2010), and sequence labelling applications, e.g., part-of-speech tagging (Rayson et al., 2007) and named entity recognition (Sydow et al., 2011). Since the rise of neural networks, a broader spectrum of applications such as sentiment analysis (Hamilton et al., 2016), information retrieval (Pettersson et al., 2016), and relation extraction (Opitz et al., 2018) have been developed.

We add to this growing literature in two ways. First, much of the work on historical text processing is focused on English$_{|\text{EN}}$, and work in other languages is still relatively unexplored (Piotrowski, 2012; Rubinstein, 2019). Second, the task of historical text summarisation has never been tackled before, to the best of our knowledge. A lack of non-EN annotated historical resources is a key reason for the former, and for the latter, resources do not exist in any language. We hope to spur research on historical text summarisation and in particular for non-EN languages through this work.

Cross-lingual summarisation. The traditional strands of cross-lingual text summarisation systems design pipelines which learn to translate and summarise separately (Leuski et al., 2003; Orăsan and Chiorean, 2008). However, such paradigms suffer from the error propagation problem, i.e., errors produced by upstream modules may accumulate and degrade the output quality (Zhu et al., 2020). In addition, parallel data to train effective translators is not always accessible (Cao et al., 2020). Recently, end-to-end methods have been applied to alleviate this issue. The main challenge for this research direction is the lack of direct corpora, leading to attempts such as zero-shot learning (Duan et al., 2019), multi-task learning (Zhu et al., 2019), and transfer learning (Cao et al., 2020). Although training requirements have been relaxed by these methods, our extreme setup with summarisation data only available for the target language and very limited parallel data, has never been visited before.

3 HistSumm Corpus

3.1 Dataset Construction

In history and digital humanities research, summarisation is most needed when analysing documentary and narrative text such as news, chronicles, diaries, and memoirs (South, 1977). Therefore, for DE we picked the GerManC dataset (Durrell et al., 2012), which contains Optical Character Recognition (OCR) results of DE newspapers from the years 1650–1800. We randomly selected 100 out of the 383 news stories for manual annotation. For ZH, we chose 『万历邸抄』 (*Wanli Gazette*) as the data source, a collection of news stories from the Wanli period of Ming Dynasty (1573–1620). However, there are no machine-readable versions of Wanli Gazette available; worse still, the calligraphy copies (see Appendix B) are unrecognisable even for non-expert humans, making the OCR technique inapplicable. Therefore, we performed a thorough literature search on over 200 related academic papers and manually retrieved 100 news texts[1].

To generate summaries in the respective modern language for these historical news stories, we recruited two experts with degrees in Germanistik and Ancient Chinese Literature, respectively. They were asked to produce summaries in the style of DE MLSUM (Scialom et al., 2020) and ZH LCSTS (Hu et al., 2015), whose news stories and summaries are crawled from the Süddeutsche Zeitung website and posts by professional media on the Sina Weibo platform, respectively. The annotation process turned out to be very effort-intensive: for both languages, the experts spent at least 20 minutes in reading and composing a summary for one single news story. The accomplished corpus of 100 news stories and expert summaries in each language, namely HistSumm (see examples in Tab. 1), were further examined by six other experts for quality control (see details in § 6.2).

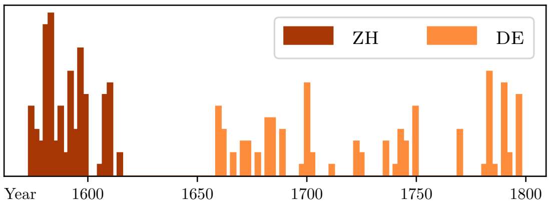


Figure 1: Publication time of HistSumm stories.

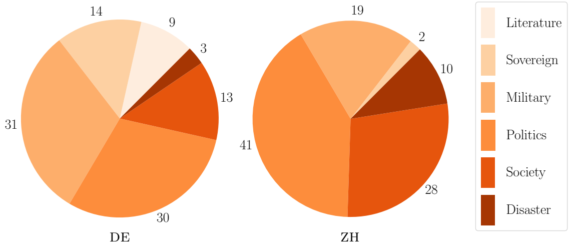


Figure 2: Topic composition of HistSumm.

	DE (word-level)		ZH (character-level)	
	HistSumm	MLSUM	HistSumm	LCSTS
L_{story}	268.1	570.6	114.5	102.5
L_{summ}	18.1	30.4	28.2	17.3
CR (%)	6.8	5.3	24.6	16.9

Table 2: Comparisons of mean story length (L_{story}), summary length (L_{summ}), and compression rate ($CR = L_{\text{summ}}/L_{\text{story}}$) for summarisation datasets.

3.2 Dataset Statistics

Publication time. As visualised in Fig. 1, the publication time of DE and ZH HistSumm stories exhibits distinguished patterns. Oldness is an important indicator of the domain and linguistic gaps (Gunn, 2011). Considering news in ZH HistSumm is on average 137 years older than its DE counterpart, such gaps can be expected to be greater. On the other hand, DE HistSumm stories cover a period of 150 years, compared to just 47 years for ZH, indicating the potential for greater linguistic and cultural variation within the DE corpus.

Topic composition. For a high-level view of HistSumm's content, we asked experts to manually classify all news stories into six categories (shown in Fig. 2). We see that the topic compositions of DE and ZH HistSumm share some similarities. For instance, Military (e.g., battle reports) and Politics (e.g., authorities' policy and personnel changes) together account for more than half the stories in both languages. On the other hand, we also have language-specific observations. 9% DE stories are about Literature (e.g., news about book publications), but this topic is not seen in ZH HistSumm. And while 14% DE stories are about

---

[1] Detailed references are included in the 'source' entries of ZH HistSumm's metadata.

Sovereign (e.g., royal families and Holy See), there are only 2 examples in ZH (both about the emperor; we found no record on any religious leader in Wanli Gazette). Also, the topics of Society (e.g., social events and judicial decisions) and Natural Disaster (e.g., earthquakes, droughts, and floods) are more prevalent in the ZH dataset.

Story length. In news summarisation tasks, special attention is paid to the lengths of news stories and summaries (see Tab. 2). Comparing DE HistSumm with the corresponding modern corpus DE MLSUM, we find that although historical news stories are on average 53% shorter, the overall compression rate (CRs) is quite similar (6.8% *vs* 5.8%), indicating that key points are summarised to similar extents. Following LCSTS (Hu et al., 2015), the table shows character-level data for ZH, but this is somewhat misleading. While most modern words are double-character, single-character words dominate the historical vocabulary, e.g., the historical word ‘朋’ (*friend*) becomes ‘朋友’ in modern ZH. According to Che et al. (2016), this leads to a character length ratio of approximately 1:1.6 between parallel historical and modern samples. Taking this into account, the CRs for the ZH HistSumm and LCSTS datasets are also quite similar to each other.

When contrasting DE with ZH (regardless of historical or modern), we notice that the compression rate is quite different. This might reflect stylistic variations with respect to how verbose news reports are in different languages or by different writers.

3.3 Vicissitudes of News

Compared with modern news, articles in HistSumm reveal several distinct characteristics with respect to writing style, posing new challenges for machine summarisation approaches.

Lexicon. With social and cultural changes over the centuries, lexical pragmatics of both languages have evolved substantially (Gunn, 2011). For DE, some routine concepts from hundreds of years ago are no longer in use today, e.g., the term ‘Brachmonat’ (№41), whose direct translation is *fallow month*, actually refers to *June* as the cultivation of fallow land traditionally begins in that month (Grimm, 1854). We observe a similar phenomenon in ZH HistSumm, e.g., ‘贡市’ (№24 and №31) used to refer to markets that were open to foreign merchants, but is no longer in use. For ZH, additionally, we notice that although some historical words are still in use, their semantics have changed over time, e.g., meaning of ‘闻’ has shifted from *hear* to *smell* (№53), and that of ‘走’ has changed from *run* to *walk* (№25).

Syntax. Another aspect of language change is that some historical syntax has been abandoned. Consider ‘daß derselbe noch länger allda/ biß der Frantz. Abgesandter von dannen widerum abreisen möge/ verbleiben soll’ (*the same should still remain there for longer, until the France Ambassador might leave again*) (№33). We find the subordinate clause is inserted within the main clause, whereas in modern DE it should be ‘daß derselbe noch länger allda verbleiben soll, biß der Frantz. Abgesandter von dannen widerum abreisen möge’. For ZH, inversion is common in historical texts but becomes rare in the modern language. For example, sentence ‘王氏之女成仙者’ (*Ms. Wang’s daughter who became a fairy*) (№65) where the attributive adjective is positioned after the head noun, should be ‘王氏之成仙（的）女’ according to modern ZH grammars. Also, we observe cases where historical ZH sentences without constituents such as subjects, predicates, objects, prepositions, etc. In these cases, contexts must be utilised to infer corresponding information, e.g., only by adding ‘居正’ (*Juzheng*, a minister’s name) to the context can we interpret the sentence ‘已，又为私书安之云’ (№20) as *‘after that, (Juzheng) wrote a private letter to comfort him’*. This adds extra difficulty to the generation of summaries.

Writing style. To inform readers, a popular practice adopted by modern news writers is to introduce key points in the first one or two sentences (White, 1998). Many machine summarisation algorithms leverage this pattern to enhance summarisation quality by incorporating positional signals (Edmundson, 1969; See et al., 2017; Gui et al., 2019). However, this rhetorical technique was not widely used in HistSumm, where crucial information may appear in the middle or even the end of stories. For instance, the keyword ‘Türck’ (*Turkish*) (№33) first occurs in the second half of the story; in article №7 of ZH HistSumm (see Tab. 1), only after reading the last sentence can we know the final outcome (i.e., the authority’s seal had been saved from fire).

4 Methodology

Based on the popular cross-lingual transfer learning framework of (Ruder et al., 2019), we propose a simple historical text summarisation framework (see Fig. 3), which can be trained even without supervision (i.e., parallel historical-modern signals).

Step 1. For both DE and ZH, we begin with respectively training modern and historical monolingual word embeddings. Specially, for DE, following the suggestions of Wang et al. (2019), we selected subword-based embedding algorithms (e.g., FastText (Joulin et al., 2017)) as they yield competitive results. In addition to training word embeddings on the raw text, for historical DE we also consider performing text normalisation (**NORM**) to enhance model performance. This orthographic technique aims to convert words from their historical spellings to modern ones, and has been widely adopted as a standard step by NLP applications for historical alphabetic languages (Bollmann, 2019). Although training a normalisation model in a fully unsupervised setup is not yet realistic, it can get bootstrapped with a single lexicon table to yield satisfactory performance (Ljubešić et al., 2016; Scherrer and Ljubešić, 2016).

For ideographic languages like ZH, word embeddings trained on stroke signals (which is analogous to subword information of alphabetic languages) achieve state-of-the-art performance (Cao et al., 2018), so we utilise them to obtain monolingual vectors. Compared with simplified characters (which dominate our training resources), traditional ones typically provide much richer stroke signals and thus benefit stroke-based embeddings (Chen and Sheng, 2018), e.g., traditional ‘葉’ (*leaf*) contains semantically related components of ‘艹’ (*plant*) and ‘木’ (*wood*), while its simplified version (‘叶’) does not.

Therefore, to improve the model performance we also conduct additional experiments on enhanced corpora which are converted to the traditional glyph using corresponding rules (**CONV**) (see § 5.3 for further details).

Step 2. Next, we respectively build two semantic spaces for DE and ZH, each of which is shared by historical and modern word vectors. This approach, namely cross-lingual word embedding mapping, aligns different embedding spaces using linear projections (Artetxe et al., 2018; Ruder et al., 2019). Given parallel supervision is very limited in real-world scenarios, we mainly consider two bootstrapping strategies: in a fully unsupervised (**UspMap**) style and through identical lexicon pairs (**IdMap**).

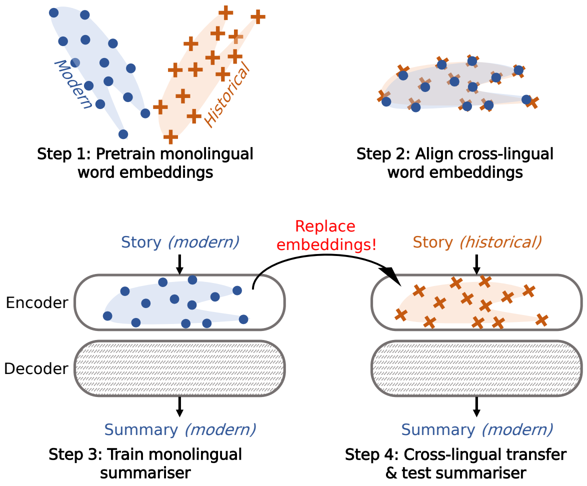


Figure 3: Illustration of our proposed framework.

While the former only relies on topological similarities between input vectors, the latter additionally takes advantage of words in the intersected vocabulary as seeds. Although their historical and current meanings can differ (cf. § 3.3), in most cases they are similar, providing very weak parallel signals (e.g., ‘Krieg’ (*war*) and ‘Frieden’ (*peace*) are common to historical and modern DE; ‘天’ (*universe*) and ‘人’ (*human*) to historical and modern ZH).

Step 3. In this step, for each of DE and ZH we use a large monolingual modern-language summarisation dataset to train a basic summariser that only takes modern-language inputs. Embedding weights of the encoder are initialised with the *modern* partition of corresponding cross-lingual word vectors in Step 2 and are frozen during the training process, while those of the decoder are randomly initialised and free to update through back-propagation.

Step 4. Upon convergence in the last step, we directly replace the embedding weights of the encoder with the *historical* vectors in the shared vector space, yielding a new model that can be fed with historical inputs but output modern sentences. This entire process does not require any external parallel supervision.

5 Experimental Setup

5.1 Training Data

Consistent with § 3.1, we selected DE MLSUM and ZH LCSTS as monolingual summarisation training sets. For monolingual corpora for word embedding training, to minimise temporal and domainal variation, we only considered datasets that were similar to articles in MLSUM, LCSTS, and HISTSUMM, i.e, with text from comparable periods and centred around news-related domains.

For modern DE, such resources are easy to access: we directly downloaded the DE News Crawl Corpus released by WMT 2014 workshops (Bojar et al., 2014), which contains shuffled sentences from online news sites. We then conducted tokenisation and removed noise such as emojis and links. For historical DE, besides the already included GerManC corpus, we also saved Deutsches Textarchiv (Nolda, 2019), Mercurius-Baumbank (Ulrike, 2020), and Mannheimer Korpus (Mannheim, 2020) as training data. Articles in these datasets are all relevant to news and have topics such as Society and Politics. Note that we only preserved documents written in 1600 to 1800 to match the publication time of DE HISTSUMM stories (cf. § 3.2). Apart from the standard data cleaning procedures (tokenisation and noise removal, as mentioned above), for historical DE corpora we replaced the very common slash symbols (/) with their modern equivalents: commas (,) (Lindemann, 2015). We also lower-cased letters and deleted sentences with less than 10 words, yielding 505K sentences and 12M words in total.

For modern ZH, we further collected news articles in the corpora released by He (2018), Hua et al. (2018), and Xu et al. (2020) to train better embeddings. For historical ZH, to the best of our knowledge, there is no standalone Ming Dynasty news collection except Wanli Gazette. Therefore, from the resources released by Jiang et al. (2020), we retrieved Ming Dynasty articles belonging to categories[2] of Novel, History/Geography, and Military[3]. Raw historical ZH text does not have punctuation marks, so we first segmented sentences using the Jiayan Toolkit[4]. Although Jiayan supports tokenisation, we skipped this step as the accuracy is unsatisfactory. Given that a considerable amount of historical ZH words only have one character (cf. § 3.2 and § 3.3), following Li et al. (2018) we simply treated characters as basic units during training. Analogous to historical DE, we removed sentences with less than 10 characters. The remaining corpus has 992k sentences and 28M characters.

5.2 Baseline Approaches

In addition to the proposed method, we also consider two strong baselines based on the Cross-lingual Language Modelling paradigm (XLM) (Lample and Conneau, 2019), which has established state-of-the-art performance in the standard cross-lingual summarisation task (Cao et al., 2020). More concretely, for DE and ZH respectively, we pretrain baselines on all available historical and modern corpora using causal language modelling and masked language modelling tasks. Next, they are respectively fine-tuned on modern text summarisation and unsupervised machine translation tasks. The former becomes the (**`XLM-E2E`**) baseline, which can be directly executed on HISTSUMM in an end-to-end fashion; the latter (**`XLM-Pipe`**) is coupled with the basic summariser for modern inputs in Step 3 of § 4 to form a translate-then-summarise pipeline.

5.3 Model Configurations

Normalisation and convention. We normalised historical DE text using cSMTiser (Ljubešić et al., 2016; Scherrer and Ljubešić, 2016), which is based on character-level statistical machine translation. Following the original papers, we pretrained the normaliser using RIDGES corpus (Odebrecht et al., 2017). As for the ZH character convention, we utilised the popular OpenCC[5] project which uses a hard-coded lexicon table to convert simplified input characters into their traditional forms.

Word embedding. As discussed in § 4, when training DE and ZH monolingual embeddings, we respectively ran subword-based FastText (Joulin et al., 2017) and stroke-based Cw2Vec (Cao et al., 2018). For both languages, we set the dimension at 100 and learned embeddings for all available tokens (i.e., `minCount` = 1). Other hyperparameters followed the default configurations. After training, we preserved the most frequent 50K tokens in each vocabulary (NB: historical ZH only has 13K unique tokens). To obtain aligned spaces for modern and historical vectors, we then utilised the robust VecMap framework (Artetxe et al., 2018) with its original settings.

Summarisation model. We implemented our main model based on the robust Pointer-Generator Network (See et al., 2017), which is a hybrid framework for extractive (to copy source expressions

---

[2] Following the topic taxonomy of Jiang et al. (2020).

[3] Sampling inspection confirmed that their domains are similar to those of Wanli Gazette.

[4] `https://github.com/jiaeyan/Jiayan`

[5] `https://github.com/BYVoid/OpenCC`

DE	ROUGE-1	ROUGE-2	ROUGE-L
`XLM-Pipe`	12.72	2.88	10.67
`XLM-E2E`	13.48	3.27	11.25
`UspMap`	13.36	3.02	11.28
`UspMap+NORM`	13.78	**3.59**	11.60
`IdMap`	13.45	3.10	11.38
`IdMap+NORM`	**14.37**	3.30	**12.14**
ZH			
`XLM-Pipe`	10.91	2.96	9.83
`XLM-E2E`	12.67	3.86	11.02
`UspMap`	13.09	4.25	11.31
`UspMap+CONV`	16.38	6.06	14.00
`IdMap`	18.38	7.05	15.89
`IdMap+CONV`	**19.22**	**7.42**	**16.52**

Table 3: ROUGE F1 scores (%) on HistSumm.

EN→ZH	ROUGE-1	ROUGE-2	ROUGE-L
`XLM-Pipe`	14.93	4.14	12.62
`XLM-E2E`	**18.02**	**5.10**	**15.39**
`UspMap`	11.43	1.27	10.07
`IdMap`	12.06	1.72	10.93
ZH→EN			
`XLM-Pipe`	9.08	3.29	7.43
`XLM-E2E`	**12.97**	**4.31**	**10.95**
`UspMap`	5.15	0.84	2.42
`IdMap`	5.98	1.33	2.90

Table 4: ROUGE F1 scores (%) of *standard* cross-lingual summarisation. Following Cao et al. (2020), for monolingual pretraining, we used corpora in § 5.3 (57M sentences) for modern ZH and annotated Gigaword (Napoles et al., 2012) (183M sentences) for EN; for summarisation training, we used LCSTS for EN→ZH and CNN/DM dataset (Hermann et al., 2015) for ZH→EN; for testing, we used the data released by Zhu et al. (2019).

via pointing) and abstractive (to produce novel words) summarisation models. After setting up the encoder and decoder (cf. in Step 3 of § 4), we started training with the default configurations. As for the two baselines which are quite heavyweight (XLM (Lample and Conneau, 2019) is based on BERT (Devlin et al., 2019) and has 250M valid parameters), we trained them from scratch with FP16 precision due to moderate computational power access. All other hyperparameter values followed the official XLM settings. To ensure the baselines can yield their highest possible performance, we trained them on the enhanced corpora, i.e., normalised DE (`NORM`) and converted ZH (`CONV`).

6 Results and Analyses

6.1 Automatic Evaluation

We assessed all models with the standard ROUGE metric (Lin, 2004), reporting F1 scores for ROUGE-1, ROUGE-2, and ROUGE-L. Following Hu et al. (2015), the ROUGE score of ZH outputs are calculated on character-level.

As shown in Tab. 3, for DE, our proposed methods are comparable to the baseline approaches or outperform the baselines by small amounts; for ZH, our models are superior by large margins. Given that XLM-based models require a lot more training resources than our model, we consider this a positive result. For comparison of the strengths and weaknesses of the models, we show their performance for a modern cross-lingual summarisation task in Tab. 4. To heighten the contrast we chose two languages (ZH and EN) from different families and with minimal overlap of vocabulary. As shown in Tab. 4, the XLM-based models outperform our method on this modern language cross-lingual summarisation task by large margins.

The difference in the performance of models on the modern and historical summarisation tasks illustrate key differences in the tasks and also some of the shortcomings of the models. Firstly, the great temporal gap (up to 400 years for DE and 600 years for ZH) between our historical and modern data hurts the XLM paradigm, which relies heavily on the similarity between corpora (Kim et al., 2020). In addition, Kim et al. (2020) also show that inadequate monolingual data size (less than 1M sentences) is likely to lead to unsatisfactory performance of XLM, even for etymologically close language pairs such as EN-DE. In our experiments we only have 505K and 992K sentences for historical DE and ZH (cf. § 5.1). On the other hand, considering the negative influence of the error-propagation issue (cf. § 2), the poor performance of `XLM-Pipe` is not surprising and is in line with observations of Cao et al. (2020) and Zhu et al. (2020). Our model instead makes use of cross-lingual embeddings, including bootstrapping from identical lexicon pairs. This approach helps overcome data sparsity issues for the historical summarisation tasks and is also successful at leveraging the similarities in the language pairs. However, its performance drops when the two languages are as far apart as EN and ZH.

When analysing the ablation results of the proposed method, on DE and ZH we found different trends. For DE, scores achieved by all the four setups show minor variance. To be specific, models bootstrapped with identical word pairs outperformed the unsupervised ones, and models trained on normalised data yielded stronger performance.

DE	Informativeness	Conciseness	Fluency	Currentness
`Expert`	4.85 (.08)	5.00 (.00)	4.94 (.03)	4.99 (.00)
`XLM-E2E`	2.26 (.20)	2.35 (.24)	**3.34** (.19)	3.67 (.23)
`UspMap+NORM`	2.51 (.18)	2.53 (.22)	3.28 (.22)	3.64 (.24)
`IdMap+NORM`	**2.52** (.18)	**2.54** (.20)	3.32 (.28)	**3.72** (.24)
ZH				
`Expert`	4.72 (.10)	4.98 (.01)	4.97 (.02)	4.90 (.04)
`XLM-E2E`	2.18 (.23)	2.21 (.27)	**2.80** (.22)	2.53 (.23)
`IdMap`	**2.39** (.19)	2.49 (.26)	2.66 (.25)	2.50 (.23)
`IdMap+CONV`	2.37 (.21)	**2.57** (.28)	2.78 (.24)	**2.59** (.25)

Table 5: Average human ratings on HISTSUMM (variance is in parentheses).

Among all tested versions, `UspMap+NORM` got the best score in ROUGE-2 and `IdMap+NORM` led in ROUGE-1 and ROUGE-L, indicating that the normalisation enhancement does benefit DE historical text summarisation models. For ZH, as predicted, with richer glyph information encoded, the stroke-based embedding method can better learn word semantics. We find that `UspMap+CONV` outperforms `UspMap` and `IdMap+CONV` outperforms `IdMap`. Adding identical words during mapping initialisation brings substantial benefits too: 3.58% and 2.52% ROUGE-L improvement for `IdMap` over `UspMap` and `IdMap+CONV` over `UspMap+CONV`, respectively.

6.2 Human Judgement

To gain further insights, we invited six experts to conduct human evaluations. Like the annotators in § 3.1, they also held degrees in Germanistik or Ancient Chinese Literature. Beyond the standard dimensions of summarisation evaluation (Informativeness, Conciseness, and Fluency), we added 'Currentness' as the fourth, which focuses on measuring 'to what extent a summary follows current rather than early linguistic styles'. We used a five-point Likert scale, with 1 for worst and 5 for best. For each language, experts were only asked to rate the gold-standard human summary and the summaries generated by the `XLM-E2E` baseline and the best two setups in § 6.1. For each of the 100 news stories in each language, 3 experts independently each rated the three model outputs and the human summary.

The final results are given in Tab. 5. When comparing different systems, we report statistical significance as the p-value of two-tailed t-tests with Bonferroni correction (Dror et al., 2018). We found that in all aspects the scores for the gold-standard summaries were always above 4 points, indicating the high quality of the gold-standard summaries. Across both languages, our models outperform the baseline for informativeness and conciseness ($p<0.01$) and achieve comparable levels of fluency and currentness. Summaries generated by `XLM-E2E` were slightly more fluent than our approach for both DE and ZH ($p<0.05$), indicating that the baseline has merit with respect to its language modelling abilities. However, it tended to make errors in understanding historical inputs and locating key points; e.g. the human reference for ZH article №57 is focused on the commander's decision of bursting the river to beat the rebel army ('宁夏之役中，魏学曾为了击溃叛乱部落，决定决河灌城'), but `XLM-E2E` summarises it as 黄河大堤水，比塔顶还高几丈' (*the surface of the river is several feet higher than the tower top*), which is fluent but irrelevant.

As for different setups of the proposed algorithm, for DE, in dimensions of Informativeness, Conciseness and Fluency, the performance of `UspMap+Norm` and `IdMap+NORM` was almost equally good. The improvement from utilising identical word pairs for cross-lingual word embedding mapping seems more evident for Currentness, i.e., the average score was 0.08 higher ($p<0.05$). For ZH, while `IdMap` and `IdMap+CONV` achieved close Informativeness scores, the latter outperforms the former in other three aspects by 0.08, 0.12, and 0.09 respectively ($p<0.01$). This observation indicates that when the lexical encoding is improved with enriched stroke-level information, the model is less likely to include redundant information in the summaries (i.e., conciseness score is higher), and the produced sentences are more fluent in terms of modern ZH grammars (see output examples in Appendix A).

6.3 Error Analysis

We further analysed model inputs with the lowest scores in § 6.2, and found that they were mostly for stories whose content was dissimilar to *any* sample in modern training sets. For instance, five ZH texts in HISTSUMM are on themes not seen in modern news (i.e., witchcraft (№65), monsters (№35 and №46), and abnormal astromancy (№8 and №28)). On these texts, even the best-performing `IdMap+CONV` model outputs a large number of [UNK] tokens and can merely achieve average Informativeness, Conciseness, Fluency, and Correctness scores of 1.41, 1.67, 1.83, and 1.60 respectively, which are significantly below its overall results in Tab. 5. This reveals the current system's shortcoming when processing inputs with theme-level *zero-shot* patterns. This issue is typically ignored in the cross-lingual summarisation literature due to the rarity of such cases in modern language tasks. However, we argue that a key contribution of our proposed task and dataset is that they together indicate new improvement directions beyond standard cross-lingual summarisation studies, such as the challenges of zero-shot generalisation and historical linguistic gaps (cf. § 3.3).

7 Conclusion and Future Work

This paper introduced the new task of summarising historical documents in modern languages, a previously unexplored but important application of cross-lingual summarisation that can support historians and digital humanities researchers. To facilitate future research on this topic, we constructed the first summarisation corpus for historical news in DE and ZH using linguistic experts. We also proposed an elegant transfer learning method that makes effective use of similarities between languages and therefore requires limited or even zero parallel supervision. Our automatic and human evaluations demonstrated the strengths of our method over state-of-the-art baselines. This paper is the first study of automated historical text summarisation. In the future, we will improve our models to address the issues highlighted in this study (e.g. zero-shot patterns and language change), add further languages (e.g., English and Greek), and increase the size of the dataset in each language.

Acknowledgements

This work is supported by the award made by the UK Engineering and Physical Sciences Research Council (Grant number: EP/P011829/1) and Baidu, Inc. Neptune.ai generously offered us a team license to facilitate experiment tracking.

We would like to express our sincerest gratitude to Qirui Zhang, Qingyi Sha, Xia Wu, Yu Hu, Silu Ding, Beiye Dai, Xingyan Zhu, and Juecheng Lin, who are all from Nanjing University, for manually annotating and validating the HISTSUMM corpus. We also thank Guanyi Chen, Ruizhe Li, Xiao Li, Shun Wang, Zhiang Chen, and the anonymous reviewers for their insightful and helpful comments.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Marcel Bollmann. 2019. A large-scale comparison of historical text normalization systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3885–3898, Minneapolis, Minnesota. Association for Computational Linguistics.

Shaosheng Cao, Wei Lu, Jun Zhou, and Xiaolong Li. 2018. Cw2Vec: Learning Chinese word embeddings with stroke n-grams. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*. AAAI.

Yue Cao, Hui Liu, and Xiaojun Wan. 2020. Jointly learning to align and summarize for neural cross-lingual summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6220–6231, Online. Association for Computational Linguistics.

Chao Che, Wenwen Guo, and Jianxin Zhang. 2016. Sentence alignment method based on maximum entropy model using anchor sentences. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 76–85, Cham. Springer International Publishing.

Wenfan Chen and Weiguo Sheng. 2018. A hybrid learning scheme for Chinese word embedding. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 84–90, Melbourne, Australia. Association for Computational Linguistics.

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

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Xiangyu Duan, Mingming Yin, Min Zhang, Boxing Chen, and Weihua Luo. 2019. Zero-shot cross-lingual abstractive sentence summarization through teaching generation and attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3162–3172, Florence, Italy. Association for Computational Linguistics.

Martin Durrell, Paul Bennett, Silke Scheible, and Richard J. Whitt. 2012. GerManC. Oxford Text Archive.

H. P. Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285.

Russell D Gray, Quentin D Atkinson, and Simon J Greenhill. 2011. Language evolution and human history: what a difference a date makes. *Philosophical Transactions of the Royal Society B: Biological Sciences*.

Jacob Grimm. 1854. *Deutsches Wörterbuch*. Hirzel.

Min Gui, Junfeng Tian, Rui Wang, and Zhenglu Yang. 2019. Attention optimization for abstractive document summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1222–1228, Hong Kong, China. Association for Computational Linguistics.

S. Gunn. 2011. *Research Methods for History*. Research Methods for the Arts and Humanities Series. Edinburgh University Press.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Austin, Texas. Association for Computational Linguistics.

Zhengfang He. 2018. Chinese short text summarization dataset.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Hou Ieong Brent Ho and Hilde De Weerdt. 2014. MARKUS. Text Analysis and Reading Platform. Funded by the European Research Council and the Digging into Data Challenge.

Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. LCSTS: A large scale Chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1967–1972, Lisbon, Portugal. Association for Computational Linguistics.

Lifeng Hua, Xiaojun Wan, and Lei Li. 2018. Overview of the NLPCC 2017 shared task: Single document summarization. In *Natural Language Processing and Chinese Computing*, pages 942–947, Cham. Springer International Publishing.

Yan-ting Jiang, Yu-ting Pan, and Le Yang. 2020. A research on verbal classifiers collocation in premodern Chinese based on statistics and word embedding. In *Journal of Xihua University (Philosophy & Social Sciences)*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Yunsu Kim, Miguel Graça, and Hermann Ney. 2020. When and why is unsupervised neural machine translation useless? In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 35–44, Lisboa, Portugal. European Association for Machine Translation.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

Anton Leuski, Chin-Yew Lin, Liang Zhou, Ulrich Germann, Franz Josef Och, and Eduard Hovy. 2003. Cross-lingual c*st*rd: English access to Hindi information. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):245–269.

Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical reasoning on Chinese morphological and semantic relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 138–143. Association for Computational Linguistics.

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

Mary Lindemann. 2015. A Ruler's Consort in Early Modern Germany: Aemilia Juliana of Schwarzburg-Rudolstadt. *German History*, 33(2):291–292.

Nikola Ljubešić, Katja Zupan, Darja Fišer, and Tomaž Erjavec. 2016. Normalising Slovene data: historical texts vs. user-generated content. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, pages 146–155.

IDS Mannheim. 2020. Mannheimer Korpus Historischer Zeitungen und Zeitschriften (Version 1.0). Institut für Deutsche Sprache Mannheim.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-Scale Knowledge Extraction*, AKBC-WEKEX '12, page 95–100, USA. Association for Computational Linguistics.

Andreas Nolda. 2019. Deutsches Textarchiv (1600–1900). Berlin-Brandenburg Academy of Sciences and Humanities (BBAW).

Carolin Odebrecht, Malte Belz, Amir Zeldes, Anke Lüdeling, and Thomas Krause. 2017. Ridges herbology: designing a diachronic multi-layer corpus. *Language Resources and Evaluation*, 51(3):695–725.

Juri Opitz, Leo Born, and Vivi Nastase. 2018. Induction of a large-scale knowledge graph from the Regesta Imperii. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 159–168, Santa Fe, New Mexico. Association for Computational Linguistics.

Constantin Orăsan and Oana Andreea Chiorean. 2008. Evaluation of a cross-lingual Romanian-English multi-document summariser. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

Csaba Oravecz, Bálint Sass, and Eszter Simon. 2010. Semi-automatic normalization of old Hungarian codices. In *Proceedings of the ECAI 2010 Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH 2010)*, pages 55–59.

Eva Pettersson, J. Lindström, B. Jacobsson, and Rosemarie Fiebranz. 2016. Histsearch - implementation and evaluation of a web-based tool for automatic information extraction from historical text. In *HistoInformatics@DH*.

Michael Piotrowski. 2012. Natural language processing for historical texts. *Synthesis lectures on human language technologies*, 5(2):1–157.

Paul Rayson, Dawn E Archer, Alistair Baron, Jonathan Culpeper, and Nicholas Smith. 2007. Tagging the bard: Evaluating the accuracy of a modern POS tagger on early modern english corpora. In *Proceedings of the Corpus Linguistics conference: CL2007*.

Aynat Rubinstein. 2019. Historical corpora meet the digital humanities: the Jerusalem Corpus of emergent modern Hebrew. *Language Resources and Evaluation*, 53(4):807–835.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, 65(1):569–630.

Yves Scherrer and Nikola Ljubešić. 2016. Automatic normalisation of the Swiss German ArchiMob corpus using character-level machine translation. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, pages 248–255.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. MLSUM: The multilingual summarization corpus.

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

S.A. South. 1977. *Method and Theory in Historical Archeology*. Institute for Research on Poverty Monograph Series. Academic Press.

Marcin Sydow, Krzysztof Ciesielski, and Jakub Wajda. 2011. Introducing diversity to log-based query suggestions to deal with underspecified user queries. In *International Joint Conferences on Security and Intelligent Information Systems*, pages 251–264. Springer.

Hsieh-Chang Tu, Jieh Hsiang, I-Mei Hung, and Chijui Hu. 2020. Docusky, a personal digital humanities platform for scholars. *Journal of Chinese History*, 4(2):564–580.

Demske Ulrike. 2020. Mercurius-Baumbank (Version 1.1). Universität Potsdam.

Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C.-C. Jay Kuo. 2019. Evaluating word embedding models: methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, 8:e19.

Peter R White. 1998. *Telling media tales: The news story as rhetoric*. Department of Linguistics, Faculty of Arts, University of Sydney.

Liang Xu, Xuanwei Zhang, Lu Li, Hai Hu, Chenjie Cao, Weitang Liu, Junyi Li, Yudong Li, Kai

Sun, Yechen Xu, Yiming Cui, Cong Yu, Qianqian Dong, Yin Tian, Dian Yu, Bo Shi, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, and Zhenzhong Lan. 2020. CLUE: A Chinese language understanding evaluation benchmark.

Junnan Zhu, Qian Wang, Yining Wang, Yu Zhou, Jiajun Zhang, Shaonan Wang, and Chengqing Zong. 2019. NCLS: Neural cross-lingual summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3054–3064, Hong Kong, China. Association for Computational Linguistics.

Junnan Zhu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2020. Attend, translate and summarize: An efficient method for neural cross-lingual summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1309–1321, Online. Association for Computational Linguistics.

A Output Samples

DE	№11
Story	Die Arbeiten im hiesigen Arsenal haben schon seit langer Zeit nachgelassen, und seitdem die Perser so sehr von den Russen geschlagen worden sind, hört man überhaupt nichts mehr von Kriegsrüstungen in den türkischen Provinzen. Die Pforte hatte nicht geglaubt, daß Rußland eine so starke Macht nach den Ufern des kaspischen Meeres abschicken, und daß der Krieg mit den Persern sobald eine so entscheidende Wendung nehmen würde. Alle kriegerischen Nachrichten, die wir jetzt aus den türkischen Provinzen erhalten, erstrecken sich blos auf die bewaffneten Räuber-Korps, die in der Gegend von Adrianopel noch immer ihren Unfug fortsetzen, der auch wohl nicht eher aufhören wird, bis die Pascha's selbst bestraft worden sind, die die Räuber beschützen. - Im Anfange dieses Monats erschien eine russische Fregatte am Eingange des schwarzen Meeres, ward durch Sturm vor den türkischen Forts vorbei in den Kanal getrieben, ohne daß die Kommandanten dieser Forts ihr den geringsten Widerstand entgegen stellen konnten, und legte sich, Bujukdere gegenüber, vor Anker. Sobald der Kapitän-Pascha dies erfuhr, verfügte er, daß jene Kommandanten abgesetzt werden sollten, und beschwerte sich bei dem hiesigen russischen Minister darüber, daß jenes Kriegsschiff sich unterstanden habe, wider alle Stipulationen der Traktaten in den Kanal einzulaufen. Nachdem aber der Zufall, wodurch dies geschehen ist, näher aufgeklärt war, widerrief der Kapitän-Pascha die Befehle, die gegen die Kommandanten der an dem Kanal gelegenen Forts erlassen wurden. Auch ward auf Ansuchen des russischen Gesandten der gedachten Fregatte aller mögliche Beistand geleistet, um sich repariren, und nach der Krimm, woher sie gekommen war, zurückkehren zu können. - Die Gesandten, welche die Pforte schon seit 2 Jahren nach Wien und Berlin bestimmt hat, sind noch immer hier; dies beweiset, daß alle Schwierigkeiten in Rücksicht dieser Missionen noch nicht gehoben sind Der nach Paris bestimmte türkische Gesandte wird aber, wie es heißt, bald abreisen. - Zwei sehr angesehene französische Offiziers, die in türkischen Dienst getreten waren, sind wieder aus demselben entlassen worden. *(The work in the arsenal has for a long time slacked off. And since the Persians were beaten so badly by the Russians, people have heard complete nothing about war armaments in the durkian provinces. The Porte would not have thought that Russia would send such a powerful force to the shores of the Caspian Sea, and that the war with the Persians would at the same time take such a decisive turn. All belligerent news, that we now receive from the Durkian provinces, extends only to the armed robber corps, which are in the area of Adrianopl still continuing their mischief, which is still unlikely to end until the pashas themselves, who protect the robbers, have been punished. - At the beginning of this month a Russian frigate appeared at the entrance to the Black Sea, was driven by a storm past the Durkian forts into the channel, without that the commanders of this fort could oppose it with the slightest resistance, and (the Russian frigate) presented itself across from Bujukdere at anchor. As soon as the captain Pasha found out about this, he decreed that those commanders should be deposed and complained to the local Russian minister about that that that grieg ship had dared to enter the canal, against all stipulations of the tracts.But after the coincidence, by which this happened, had been more clearly clarified, the captain-pasha recalled the orders, which would be enacted against the commanders of the fort on the canal. Also, at the request of the Russian confession, the intended frigate was given all possible assistance in order to repair itself and to be able to return to Grimm, whence it had come. - the confessions, that the gate has set for Vienna and Berlein for two years, are still here; this proves that all difficulties in regard to these missions have not yet been resolved, but the destined-for-Paris Durkian legate will, as it is said, soon be leaving. - two very highly respected French officers, who had entered Durk service, have been dismissed from the same.)*
`Expert`	Wie es zwischen Russland und der Türkei lief, war noch unsicher. *(How things would go between Russia and Turkey, was still uncertain.)*
`IdMap+NORM`	die [unk] des [unk] zeigen , dass der krieg mit den persern sobald eine so entschiedende wendung nehmen würde . die wendung eines blauen wunders ist nicht nur zu sehen , wie man es weitergeht . *([unk] show that the war with the Persians would very soon take such a decisive turn. The turning point of a blue miracle is not just to see how it goes on.)*
`UspMap+NORM`	die arbeiten im arsenal haben schon seite länger zeit nachgelassen , und seitedem die perser so sehr von den russen geschlagen worden sind , hört man überhaupt nichts mehr von kriegrüstungen in den durkischen provinzen . *(The work in the arsenal has for a long time slacked off. And since the persians were beaten so badly by the russians, people have heard complete nothing about war armaments in the durkian provinces.)*

DE	№33
Story	ES befindet sich schon etliche Tage hero von der Crone Schweden ein Abgeordneter incognito allhier/ aber noch unbewust in was Negotio. Am verwichenen Montage ist von Ihrer Käyserl. M. an den Abgesandten zu München Herrn Grafen von Königsegg ein Currirer abgeschickt worden/ wie man vernimt/ weilen I. Chur-Fürstl. Durchl. allda gegen I. Käyserl. M. hoch contestirt/ daß Sie Dero Käyserl. und Röm. Reichs Intereße auff alle möglichste Weise befördern helffen/ und auff solche Resolution gedachter Kayserl. Abgesandter von dar seine Reise in auffgetragener LegationsCommißion weiters nehmen wollen/ daß derselbe noch länger allda/ biß der Frantz. Abgesandter von dannen widerum abreisen möge/ verbleiben soll/ damit I. Chur-Fürstl. Durchl. durch erstgedachten Abgesandten nicht zu andern Gedancken kommen möchte. Vorgestern ist der Käys. neulich zu dem Vezier nacher Ofen geschickte Türck. Ober-Dolmetscher/ Herr Minnisky wider zurücke anhero gekommen/ von welchen man vernimt/ daß gedachter Vezier/ wie auch die Baßen von Erlau und Waradein/ sich wegen des beschuldigten Unterschleiffs der Rebellen sehr excusirt/ und negirt/ daß sie bißhero ihrem gethanen Versprächen zu wider die Rebellen in ihren Territoriis wißentlich geduldet hätten/ sondern solches vil mehrers von dem Abassy geschehen wäre/ und habe gedachter Vezier sein hievoriges Versprächen gegen I.K.M. nochmal höchstens contestiren laßen: Demnach aber/ ungeachtet diser Sinceration/ man gewiß weiß/ daß obgedachte Rebellen nicht allein von den Türcken in ihren Gebieten geduldet/ sondern auch bewaffnet worden/ und in neulicher Action die Türcken auff Seiten der Rebellen selbsten darbey gewesen/ also läst es sich nun zu einer würcklichen Ruptur ansehen/ deßwegen auch bey Hofe vil Patenten auff neue Werbungen heraus gegeben werden. *(A few days ago there was a member of parliament incognito here from the Royal Family of Sweden, but still unconsciously in some business. On the elapsed Monday, a Currier was sent from their Royal M to the emissaries to monks, Grafen von Königsegg, as people hear, that I. Chur-Fürstl Durchl is contesting against I. Royal M, that they help to promote the Royal and Roman Empire interests in every possible way, and that Royal Abgesander who is thinking of such a resolution, wants to continue his journey in the applying Legations Commission, that the same should still remain there for longer, until the Franz Abgesander might leave again, so that I. Chur-Fürstl Durchl through the first envisaged delegate does not want to come to other thoughts. The day before yesterday Käy's new Türck interpreter, Mr. Miniski, who was sent to the Vezier afterwards, has come here, from whom people heard that the intended Vezier, like the bases of Erlau and Waradien, were for the accused hiding of the rebels very excited, and denied that they had so far knowingly tolerated their promise against the rebels in their territories, but that such a thing would have happened much more from the Abassi, and thought Vezier had made his previous promise against the IKM. at most let them contest again: but regardless of this sinceration, people know for sure, the contemplated rebels are not only tolerated by the Tirken in their areas, but also been armed, and in the recent action the Turks were themselves there on the side of the rebels, so it can be viewed now as a real rupture, which is why at court many patents on new recruitments are issued.)*
`Expert`	Der Kaiser versuchte, durch Verhandlungen seine Interessen gewährzuleisten. Inzwischen boten die Türken wider Versprechen den Rebellen Unterstützung *(The emperor tried to safeguard his interests through negotiations. Meanwhile Turks broken the promise and provided support to the rebels.)*
`IdMap+NORM`	es befindet sich schon etliche tage her von der crone schweden ein abgeordneter inconitum allhier , aber noch unbewusst in was negotio . am verwichenen montage ist von ihrer käyserl . allda gegen i . *(A few days ago there was a member of parliament incognito here from the Royal Family of Sweden, but still unconsciously in some business. On the elapsed Monday is from their Royalty all against I.)*
`UspMap+NORM`	es befindet sich schon etliche tage her von der crone schweden ein abgeordneter inconitum allhier , aber noch unbewusst in was negotio . am verwichenen montage ist von ihrer käyserl . m . an den abgestanden *(A few days ago there was a member of parliament incognito here from the Royal Family of Sweden, but still unconsciously in some business. On the elapsed Monday is from their Royal M to the stale ...)*

DE	№34
Story	Jhre Königl. Majest. befinden sich noch vnweit Thorn/ vnd seynd Cosakische Deputirte vnter Wegs/ jhr factum bey Seiner Majest. zu justificiren, vnd wegen jhrer Treu Versicherung zu thun. Von den Fridens. Tractaten zwischen Pohlen vnd Schweden ist noch wenig zu melden. Seithero die Pohlen bey Marienburg den Schweden eine Schantz/ der Kessel genant/ Abgenommen/ ist nichts weiters vorgefallen/ auch hiesiger Stadt Völcker vor dem Haupt noch nichts tentirt, jedoch sagt man daß noch dise woche etwas vorgehen werde/ so bald nur alle Battereyen in den 3 Quartieren fertig/ vnd die Mörser darauff gebracht worden/ vmb solches mit Feur zu bezwingen/ weil mit dem Schiessen doch nichts zugewinnen/ vnd der Sturm vnmöglich zu wagen ist/ daß aber das Brau- vnd Proviant Hauß darin in brand geschossen/ vnd die darinnen befindliche Cavallerie also ruinirt werden/ daß sie keinen Außfall mehr thun können/ ist gewiß/ deßgleichen hat der Obriste Zaphlizky mit 2000. Mann den Elbingern daß Viehe weggetriben/ welche darauff mit 500. Mann außgefallen/ solches wider zu erobern/ seynd aber mehrentheils nidergemacht/ vnd 6. vornehme Officierer neben vielen Gemeinen gefangen worden. So ist auch auß Churland über Memmel sichere Zeitung einkommen/ daß Herr General Duglas nur 2000. Mann nach Liffland gebracht/ vnd Pautzke sich mit Accord an die Pohlen ergeben habe/ seynd also von den Schweden in Mittau noch 300. Mann übrig/ deren Ergebung man nechstens zu verenehmen hoffet/ zumahlen selbige formaliter belägert seynd/ vnnd keinen Succurs zuvermuthen. Den gefangnen Hertzogen von Churland haben die Schweden wider in Liffland nach Revel gebracht/ dahero zu seiner Erledigung Hoffnung gemacht werden will. *(Their Royal Majesties are still not far from Torn, and there are Cossack deputies on the way to their factum to be judged by His Majesty, and to be insured for their loyalty. (and for their loyal insurance to do that. from the Fridens. Tracts between Poland and Sweden are still little to be reported. Since the Pohlen near Marienburg took away a Schanz, which is called "boiler", from the Swedes, nothing further has happened, also local city peoples have yet in the first place tented nothing, however, they say that something will happen this week, as soon as all batteries are in the 3 quarters ready, and the mortars were brought to it, in order to defeat it with fire, because by shooting nothing could be gained, and the storm is impossible to be venture, but the brown-known and provisions house was set on fire and in it the cavalry were so ruined that they could no longer do any sorties, is certain. Likewise, Colonel Zaplizki has driven away the cattle from the Elbingers with 2,000 men, who with 500 men failed to conquer such, but were mostly killed, and 6 distinguished officers were captured alongside many common ones. It is also sure to be a newspaper from Churland via Memmel coming in, that General Duglas only brought 2000 men to Lifland, and Pauzke has surrendered to the Poles by accord, so from the Swedes in Mittau are still 300 men left, whose surrender is the next that people hoped to hear, as they are formally besieged, and no succurs can be expected. The Swedes have brought the captured Duke of Churland back to Revel in Lifland, therefore desired for his completion hope to be made.)*
`Expert`	Der Krieg zwischen Polen und Schweden dauert an. Von einem Friedensvertrag ist noch nicht der Rede. *(The war between Poland and Sweden continues. Of the peace treaty is there still no talk.)*
`IdMap+NORM`	ihre königl . maiest . befinden sich noch unweit toren , und sind cosakische deputierte unter weg , ihr factum bei seiner maiest . zu justifizieren , und wegen ihrer treu versicherung zu tun . *(Their Royal Majesties are still not far from Torn, and there are Cossack deputies on the way to their factum to be judged by His Majesty, and for their loyal insurance to do that.)*
`UspMap+NORM`	ihre königl . maiest . befinden sich noch unweit toren , und sind cosakische deputierte unter weg , ihr factum bei seiner maiest . zu justifizieren , und wegen ihrer treu versicherung zu tun . von den fridens . *(Their Royal Majesties are still not far from Torn, and there are Cossack deputies on the way to their factum to be judged by His Majesty, and for their loyal insurance to do that.)*

DE	№39
Story	Heute ist der Kayserl. General-Kriegs-Commissarius, Graf von Nesselrode, mit dem wegen Anweisung derer künftigen Winter-Quartiere abgefasseten Plan von Wien nach dem Kayserl. Haupt Quartier Haydelberg, zu des Printzen Eugenii Hoch-Fürstl. Durchl. wieder zurücke gegangen. Es verlautet dabey, daß die würckliche Einrichtung dererselben viele Schwürigkeiten gefunden habe, und daß verschiedene Reichs-Stände dieselbe von ihren Landen zuförderst abwenden wollen. Mehrere und besondere Umstände sind davon noch nicht bekand. An dem Kayserl. Hofe ist zu Bestreitung derer fortdaurenden schweren Kriegs-Kosten, beschlossen worden, auf verschiedene Waaren, und insonderheit auf den Wein u. Fische einen neuen Impost zu legen, ob aber auch künfftig das Silber-Geschirre in die Kayserl. Müntze dürfte gefordert werden, wie bisher verlauten will, solches ist noch zweyfelhafftig, inzwischen wird mit Eintreibung eines sogenandten Subsidii präsentanti, wobey alle vermögende Leute zur Anticipation eines nach eines jedweden Vermögen eingerichteten Quanti angehalten werden, und dargegen aus der Kayserl. Banco in 3. Jahren zahlbare Banco-Obligationen, nebst 5. pro Cent Interesse erhalten, nicht nur zu Wien fortgefahren, sondern es soll auch dergleichen in allen Kayserl. Erb-Landen, das eintzige Königreich Ungarn ausgenommen, dessen Privilegia solches nicht verstatten, eingeführet werden. Man hat aus Italien Nachricht, daß die Alliirten ten zwischen der Etsch und Adige nicht nur eine starcke Linie gezogen haben, um denen Deutschen den Rück-Weg nach dem Mantuanischen gäntzlich zu benehmen, sondern sich auch gegen das Triedentinische ziehen, und daselbst einbrechen wollen. Sonst weiß man, daß der Erb-Printz und numehro regierende Durchl. Hertzog, Carl, von Braunschweig-Wolfenbüttel, die gesuchte Veniam ätatis von Ihro Kayserl. Maj. auf das erstattete Reichs-Hofe-Raths-Gutachten erhalten habe. *(Today the Lord General-Krügs-Commissarius, Count of Nesselbrote, is with with the plan, which is drawn up according to the instructions for their future winter quarters, from Vienna to the emperor's headquarter, Heidelberg, to the Prince Eugenii Hoch-Fürstl Durchl. Again, it passed back. It is said that the real institution of the same has found many difficulties, and that the various imperial estates want to turn them away from their lands first of all. Several and special circumstances are not yet known about. At the Royal Court it was decided to pay for the continuing heavy Krügs costs, on various goods, and in particular on the wine and the Fish a new tax to put, but whether in the future the silver dishes in the Royal Coin should be required, as has been so far announced, this is still doubtful. In the meantime with the collection of a so-called Subsidium Presentanti, (in which all wealthy people are encouraged to anticipate a Quanti set up according to one's every wealth, and on the other hand from the Royal Bank received in 3 years payable bank debts, plus 5th interest per cent,) not only carried on in Vienna, but the same should be introduced in all Royal Hereditary Lands, the only kingdom of Hungary, with the exception of whose privileges do not permit such. People had news from Italy that the Allies had not only drawn a strong line between the Etsch and the Adige in order to deprive the Germans of the way back to Mantuan entirely, but also oppose the Tridentine and want to break in there. Otherwise knowing people that the Hereditary Prince and now ruling Highness Duke, Carl, von Braunschweig-Wolfenbutel, who have sought for Veniam ätatis from their Royal May, have received the submitted Reichs-Hofe-RAts report.)*
`Expert`	Der Kaiser ließ einrichtungsbezogene, finanzielle, militärstrategische und personelle Anordnungen vornehmen, um den Krieg weiterzuführen. *(The emperor had ordered to make facility-related, financial, military-strategic and personnel arrangements in order to continue the war.)*
`IdMap+NORM`	heute ist der kaiserl . general-krügs-commissarius , graf von nesselbrote , mit dem wegen angeweisung derer künftigen winter-quartiere abgefassten plan von wie nach dem kaiserl . haupt quartier heidelberg , zu des prinzen eugenii hoch-fürstl . *(Today the Lord General-Krügs-Commissarius, Count of Nesselbrote, is with with the plan, which is drawn up according to the instructions for their future winter quarters, from Vienna to the emperor's headquarter, Heidelberg, to the Prince Eugenii Hoch-Fürstl.)*
`UspMap+NORM`	heute ist der kaiserl . general-krügs-commissarius , graf von nesselbrote , mit dem wegen angeweisung derer künftigen winter-quartiere abgefassten plan von wie nach dem kaiserl . haupt quartier heidelberg , zu des prinzen eugenii hoch-fürstl . *(Today the Lord General-Krügs-Commissarius, Count of Nesselbrote, is with with the plan, which is drawn up according to the instructions for their future winter quarters, from Vienna to the emperor's headquarter, Heidelberg, to the Prince Eugenii Hoch-Fürstl.)*

DE	№50
Story	Donau-Strohm vom 13. Weinm. Aus Breßlau hat man unterm 3. dieses folgende Nachricht: Vorgestern sind ungemein grosse Heere Heuschrecken über hiesige Stadt gezogen, deren Flug von 10. Uhr des Mittags bis gegen 4. Uhr Abends gedauret. Eine Colonne nahme bey nahem die gantze Breite der Stadt ein, und die Höhe betrug ohngefehr 130. bis 140. Ellen. Noch viele andere Colonnen breiteten sich in grosser Menge aus, und man berichtet aus Zotten, daß sie allda ebenfalls in grosser Menge durchgeflogen seyen. Dieses Ungeziefer verliehret auf seinem Marsch viele von seinen Cameraden, welche von den Krähen, Raben, Dohlen und andern Vögeln fleißig gefangen werden, welche den Bauch eines Heuschrecken samt dem Eingeweyde fressen, und das übrige auf die Erde fallen lassen, von denen man viele auf hiesigen Feldern gesehen. Gestern sind wiederum neue Schwärme hier ankommen, welche sich aber nicht gelagert, sondern, wie die andern, ihren Flug weiter genommen haben, und dieser ihr Zug dauret so lang, als lang die Sonne hell und warm scheinet. Auf die Nacht erhobe sich ein hefftiger Wind, der unsere bisherige warme Lufft ziemlich abgekühlet, weswegen wir heute wenig Heuschrecken sehen. Auf denen Gütern des Grafen von Schweidnitz, zu Stephansdorff, ohngefehr 4. Meilen von hier, hat dieses Ungeziefer grossen Schaden gethan, da dasselbe alle Wayde für das Vieh abgefressen, und vorgestern ist ein anderes unbeschreiblich starckes Heer über gedachte Güter gezogen, welches seinen Flug gegen Prochwitz und Liegnitz genommen. *(Danube stream from the 13th Weinm. From Wroclaw comes under the 3rd day the following message: The day before yesterday, a great number of locusts have flown over the local city, and their flight lasted from 10 a.m. to around 4 p.m. A column took up almost the whole breadth of the city, and the height was about 130 to 140 cubits. Also, many other columns spread out in great numbers, and it is reported according to the villi that they had also flown through there in great numbers. This vermin lost on its march many of its companions, who were by the crows, ravens, jackdaws, and other birds busily caught, which eat the belly of a locust and its entrails, and let the rest of them fall to the ground, many of which have been seen on local fields. Yesterday again new swarms have arrived here, who didn't know what to do, but, like the others, continued their flight, and this migration lasts as long as the sun shines bright and warm. In the night rose up a violent wind, which cooled down our previous warm air quite a bit, which is why we today see few locusts. On the property of the Count of Schweidnitz, at Stephandarf, about four miles from here, this vermin has done great damage, since it has eaten up all the woad for the cattle, and the day before yesterday has another indescribably strong army marched over intended goods, which its flight took against Prochwitze and Lignitz.)*
`Expert`	eine große Menge von Heuschrecken sind durchgeflogen. Ihre Anzahl ist wegen der insektfressenden Vögel und hefttigen Wind gesunken. *(A great number of locusts have flown through. Their numbers have decreased due to insectivorous birds and violent winds.)*
`IdMap+NORM`	donaunknownstrom vom 13 . weinem . aus breslau hat man unterm 3 . dieses folgende nachricht : vorgestern sind ungemein grosse her heuschrecken über hiesige stadt gezogen , deren flug von 10 . ihr des mittages bis gegen 4 . ihr abends gedauert . *(Danube stream from the 13th Weinm. From Wroclaw comes under the 3rd day the following message: The day before yesterday, a great number of locusts have flown over the local city, and their flight lasted from 10 a.m. to around 4 p.m. Her evening lasted.)*
`UspMap+NORM`	donaunknownstrom vom 13 . weinem . aus breslau hat man unterm 3 . dieses folgende nachricht : vorgestern sind ungemein grosse her heuschrecken über hiesige stadt gezogen , deren flug von 10 . ihr des mittages bis gegen 4 . *(Danube stream from the 13th Weinm. From Wroclaw comes under the 3rd day the following message: The day before yesterday, a great number of locusts have flown over the local city, and their flight lasted from 10 a.m. to around 4 p.m.)*

ZH	№27
Story	陕西巡抚叶詹熊奏称，三月初六日，黄色蔽天白昼黑暗,徐而变为红色，天鼓大鸣。初七日夜，从四广空降火一块，如盆大，后主三尾，落西北方。又查自十九年至今，天鼓时鸣，地道数震，火光冰雹，暴雨飓风，青气吐于兽吻，白气横于斗傍，彗自天飞，血从地涌。海内人民惊讶，以为从来无此非常之变，而且多也，疏上报闻。 *(Shaanxi Governor Ye Zhanxiong reported, on March 6th, a yellow colour covered the sky and the day was dark, it slowly turned red, thundered loudly. On the night of the seventh day in the lunar new year, a fire broke down from the sky, big as a basin, with three tails at the back, landed in the northwest. It was discovered that from the 19th year to now, it sometimes thunder, there were many earthquakes, fire and hail, rainstorms and hurricanes, black gas came out of animals' lips, white gas crisscrossed near buckets, comets flew in the sky, blood gushed from the ground. People all over the country were surprised, there had never been such unusual changes, and in such great number, the report was sent to the emperor.)*
Expert	陕西天鼓轰鸣，此现象长久以来时有发生，伴随有各类地质、气象和天文灾害。 *(It thundered in Shaanxi, this phenomenon has occurred from time to time for a long time, accompanied by a variety of geology, meteorological and astronomical disasters.)*
IdMap	海内人民惊讶天鼓大鸣[UNK]。 *(People all over the country were surprised at thunder [UNK].)*
IdMap+CONV	陕西天鼓鸣[UNK]。 *(It thundered in Shaanxi [unk].)*

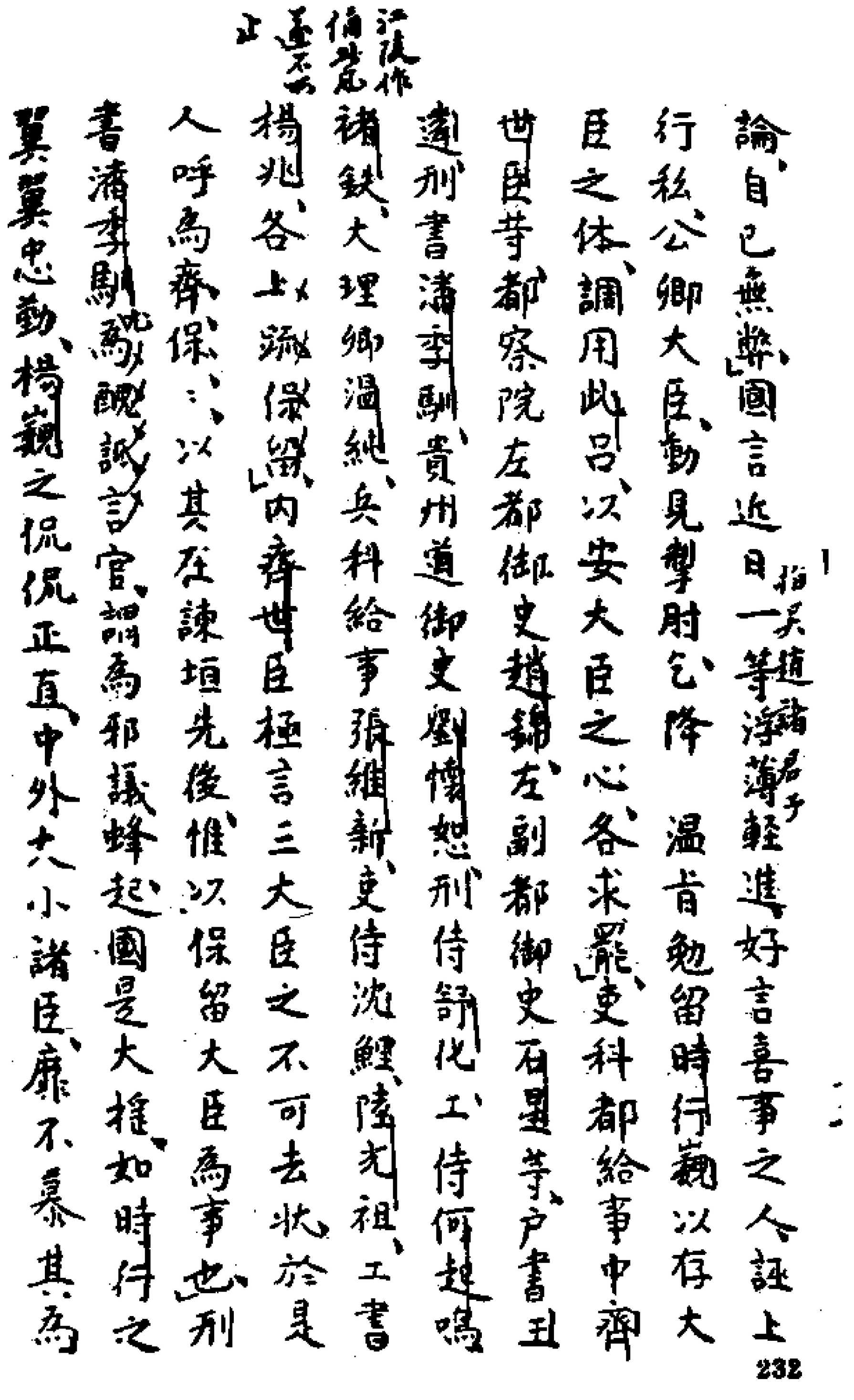
ZH	№34
Story	宋应昌撤兵自朝鲜回京。兵部复预防计御应敌之机，以便夺责，以保万安等事。有旨，犬兵画撤宋应昌既难展布。着兴李如松取妄诈，未可遂称事完，督抚遥制不便，顾豢谋等谋素闲，着星夜前往彼处料理，务保成功，写敕与他，蓟镇防掳事务，暂令顺天边抚代管，宋应昌仍候顾豢。谦至日，交代回京。 *(Song Yingchang withdrew his troops from Korea and returned to the capital. The Ministry of war has make new arranges to defend the enemies, in order to investigate the responsibilities, just in case. The emperor announces that, since Song Yingchang has trouble with his military deployment, Li Rusong is now commanded to fight with the arrogant enemies; the governor's remote control is inconvenient, and he is taking care of animals whilst doing other simple things; travel to the place overnight to handle it, be sure to succeed; write him orders, as for the war affairs in Jizhen, Shuntian governor is temporary in charge, Song Yingchang is relieved. On the next day, perform the handover and return to the capital.)*
Expert	宋应昌自朝鲜撤兵，兵部奉旨命李如松前去坐镇，交接成功后宋应昌方可回京。 *(Song Yingchang withdrew his troops from Korea, the Ministry of War ordered Li Rushong to go and take charge, after the handover Song Yingchang could return to the capital.)*
IdMap	宋应昌撤兵自朝鲜回京。 *(Song Yingchang withdrew his troops from Korea and returned to the capital.)*
IdMap+CONV	宋应昌撤兵自朝鲜回京。 *(Song Yingchang withdrew his troops from Korea and returned to the capital.)*

ZH	№70
Story	高拱卒。拱妻张氏上疏，陈乞恤典。上日：“高拱不忠，欺侮朕躬，今已死了，他妻还来乞恩，不准他。”居正言:“拱侍先帝潜邸，讲读有旧恩，宜予。”命复原职，给予祭葬。 *(Gao Gong died. His wife Zhang wrote to the emperor, asking for a court ceremony. The emperor said: "Gao Gong was unfaithful, and intimidated me. Now he is dead, his wife comes back to ask for a favour, refuse." Zhang Juzheng said: "Gong served in the emperor's residence, there was past gratitude, ought to grant." It was ordered to resume Gong's original job, and grant him a funeral.)*
Expert	高拱去世，其妻上书求恩典遭到拒绝，经居正劝说后得到恩赐。 *(Gao Gong died, his wife Zhang wrote to the emperor asking for a court ceremony but she was rejected, after Juzheng's persuasion Gong was bestowed.)*
IdMap	高拱不忠，已死了，他妻还来乞恩，不准他。 *(Gao Gong was unfaithful, he is dead, however his wife comes back to ask for a favour, refuse.)*
IdMap+CONV	高拱不忠，不准他妻来乞恩。 *(Gao Gong was unfaithful, don't allow his wife to come and ask for favour.)*

ZH	№78
Story	黄台吉裴封顺义王。礼部等部尚书等官徐学谟等题称，北虏求嗣封爵，称黄台吉傈俺答嫡长男，应嗣王号，舍力克台吉傈黄台吉的男，应袭龙湖将军职衔。报可。 *(Huangtaiji was granted the title of King Shunyi. Xu Xuemo and others who were officials of the Ministry of Rites said, enemy from the north asked to be offered hereditary peerages, Huangtaiji was Anda's eldest son, he should inherit the title of king, Sheliketaiji was Huangtaiji's eldest son, he should inherit the rank of general Longhu. The suggestion was approved.)*
`Expert`	俺答部嫡长子黄台吉被礼部封顺义王。 *(Anda's eldest son Huangtaiji was granted the title of King Shunyi by the Ministry of Rites.)*
`IdMap`	求嗣封爵，黄台吉傈俺答嫡长男，你知道吗? *(Asking to be offered hereditary peerages, Huangtaiji was Anda's eldest son, do you know?)*
`IdMap+CONV`	黄台吉傈俺答嫡长男应袭龙湖将军职衔。 *(Huangtaiji was Anda's eldest son and should inherit the rank of general Longhu.)*

ZH	№96
Story	山西宁武关军师作乱。军士李现等纠众三百，拥入兵备邢道门，逼挟粮米，鼓噪作乱。 *(The troop in Ningwuguan Shanxi rioted. Sergeant Li Xian and other three hundred soldiers gathered, they entered the gate of Xingjiedao, robbed grain and rice, and clamoured to riot.)*
`Expert`	宁武关军队以李现为首出现骚乱。 *(The troop in Ningwuguan had an riot which was lead by Lixian.)*
`IdMap`	军师作乱：逼挟粮米，逼挟粮米，逼挟粮米，逼挟粮米，逼挟粮米，鼓噪作乱！ *(Troop rioted: robbed grain and rice, robbed grain and rice, robbed grain and rice, robbed grain and rice, robbed grain and rice, clamoured to riot!)*
`IdMap+CONV`	山西宁武关军师作乱。 *(The troop in Ningwuguan Shanxi rioted.)*

B Sample of Wanli Gazette (Scanned) Copies



論、自己無弊、國言近日一（指吳趙諸君子）等浮薄輕進好言喜事之人，誣上行私，公卿大臣，動見掣肘，乞降　溫旨勉留時行、巍，以存大臣之体、調用此呂，以安大臣之心，各求罷、吏科都給事中齊世臣等，都察院左都御史趙錦、左副都御史石星等，戶書王遴、刑書潘季馴、貴州道御史劉懷恕、刑侍舒化、工侍何起鳴、褚鈇、大理卿溫純、兵科給事張維新、吏侍沈鯉、陸光祖、工書楊兆、各上疏保留，內齊世臣極言三大臣之不可去狀，於是人呼為齊保保，以其在諫垣先後惟以保留大臣為事也、刑書潘季馴駁疏醜詆言官，謂為邪議蜂起，國是大搖，如時行之翼翼忠勤，楊巍之侃侃正直，中外大小諸臣，靡不暴其為

江陵作俑党遂不已止

232

Challenges in Automated Debiasing for Toxic Language Detection

Xuhui Zhou♡ **Maarten Sap**♣ **Swabha Swayamdipta**♢ **Noah A. Smith**♣♢ **Yejin Choi**♣♢

♡Department of Linguistics, University of Washington
♣Paul G. Allen School of Computer Science & Engineering, University of Washington
♢Allen Institute for Artificial Intelligence
`xuhuizh@uw.edu, msap@cs.washington.edu`
`{swabhas,noah,yejinc}@allenai.org`

Abstract

***Warning**: this paper contains content that may be offensive or upsetting.*

Biased associations have been a challenge in the development of classifiers for detecting toxic language, hindering both fairness and accuracy. As potential solutions, we investigate recently introduced debiasing methods for text classification datasets and models, as applied to toxic language detection. Our focus is on lexical (e.g., swear words, slurs, identity mentions) and dialectal markers (specifically African American English). Our comprehensive experiments establish that existing methods are limited in their ability to prevent biased behavior in current toxicity detectors. We then propose an automatic, dialect-aware data correction method, as a proof-of-concept study. Despite the use of synthetic labels, this method reduces dialectal associations with toxicity. Overall, our findings show that debiasing a model trained on biased toxic language data is not as effective as simply relabeling the data to remove existing biases.

1 Introduction

Current hate speech or toxic language detection[1] systems exhibit problematic and discriminatory behavior that causes them to have disparate negative impact on minority populations (Yasin, 2018; Guynn, 2020; Kim et al., 2020; Dias Oliva et al., 2020). Tweets simply containing a minority identity mention are commonly flagged as toxic by current systems, in contrast to those containing majority identity mentions, as illustrated in Figure 1.

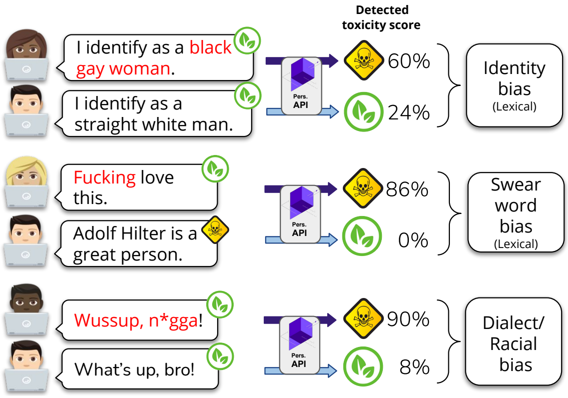

I identify as a black gay woman.
I identify as a straight white man.
Fucking love this.
Adolf Hilter is a great person.
Wussup, n*gga!
What's up, bro!

Figure 1: Lexical items and dialect markers cause problematic behavior for toxic language detection systems such as the widely used PerspectiveAPI. In the top two example pairs, statements with minority identity mentions and swear words used inoffensively are flagged as toxic, but majority identity mentions or offensive statements without overt swearing are missed. The bottom pair shows dialect-based racial bias for two inoffensive greetings, where markers of African American English (AAE) trigger the toxicity detector.

At the core of the issue are *dataset biases*, i.e., spurious correlations between surface patterns and annotated toxicity labels (§2), which stem from the data creation process (Sap et al., 2019). Previous work has outlined two such biases for hate speech datasets (both shown in Figure 1): *lexical bias* which associates toxicity with the presence of certain words (e.g., profanities, identity mentions; Dixon et al., 2018; Dinan et al., 2019) and *dialectal bias*, where toxicity is correlated with surface markers of African American English (AAE; Davidson et al., 2019; Sap et al., 2019). When trained on biased datasets, models acquire *and* exacerbate these biases (e.g., flagging text by Black authors as more toxic than by white authors; Sap et al., 2019; Zhang et al., 2018).

Concurrently, there has been elevated interest in developing *debiasing methods* for standard natural language understanding (NLU) tasks, i.e., methods that aim to decrease over-reliance on spurious correlations in NLU models (Clark et al., 2019; He et al., 2019; Karimi Mahabadi et al., 2020; Bras et al., 2020). This raises a natural question: *are*

---

[1] We use *hate speech* and *toxic language* interchangeably in this work, though their definitions do not perfectly align.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3143–3155
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

current debiasing approaches effective for mitigating biases specific to toxic language detection?

In this work, we address the above question by investigating two classes of debiasing approaches to mitigate lexical and dialectal biases—one that employs additional training objectives for bias removal, and another that filters training instances likely exhibiting spurious biases (§3). Through comprehensive experiments, we show that both approaches face major challenges in mitigating biases from a model trained on a biased dataset (in our case, the dataset from Founta et al., 2018) for toxic language detection. While data filtering results in reduced bias associations in the data, *models* trained on filtered datasets still pick up on lexical (§4) and dialectal biases (§5). We find that dialectal biases are particularly challenging to address, as has also been shown by Xia et al. (2020). "Debiased" models still disproportionately flag text in certain dialects as toxic. Notably, mitigating dialectal bias through current debiasing methods does not mitigate a model's propensity to label tweets by Black authors as more toxic than by white authors.

We additionally explore an alternative proof-of-concept study—relabeling supposedly toxic training instances whose automatic translations into a majority dialect are deemed non-toxic by the classifier. To this end, we create a synthetic dataset via few-shot dialect translation system built with GPT-3 (Brown et al., 2020). While only an illustrative solution, it nevertheless takes into account the dialectal context of the tweet, resulting in a model less prone to dialectal and racial biases (§6). Overall, our findings indicate that debiasing a model already trained on biased toxic language data can be challenging, compared to relabeling the data to remove existing biases. Our code and data are publicly available on Github.[2]

2 Biases in Toxic Language Detection

We test the use of debiasing[3] methods for the task of toxic language detection, which aims to flag rude, offensive, hateful, or toxic language on the internet, with the goal of moderating online communities (Roberts, 2019; Vidgen et al., 2019). This task differs in several ways from the natural language understanding (NLU) tasks that debiasing methods have been successful on, such as textual entailment (e.g., SNLI, MNLI; Bowman et al., 2015; Williams et al., 2018) or reading comprehension (e.g., SQuAD; Rajpurkar et al., 2016). First, compared to these NLU tasks where there is one correct label, the toxicity of language is inherently more nuanced, subjective, and contextual, which causes toxic language datasets to have lower agreement in general (Ross et al., 2017). Second, the dataset biases in NLU are predominantly artifacts introduced during data creation (e.g., negations, exaggerations; Schwartz et al., 2017; Gururangan et al., 2018), whereas those in toxic language detection are grounded in the social dynamics of the world (Spears, 1998; Technau, 2018). For example, viewing AAE as a more toxic or less proper variety of English is a form of linguistic discrimination that upholds racial hierarchies in the United States (Rosa and Flores, 2017).

In this work, we consider two broad categories of toxic language dataset biases—lexical (§2.1) and dialectal (§2.2). Our experiments focus on a single, widely used dataset (§2.3) from Founta et al. (2018).

2.1 Lexical Biases (TOXTRIG)

Current toxic language detection systems often rely on the presence or absence of certain words (e.g., swear words, identity mentions) to make their predictions (Dixon et al., 2018; Dinan et al., 2019). While most previous analyses of this bias relied on a simple list of "bad" words (Davidson et al., 2019; Dinan et al., 2019),[4] we take a more nuanced view of how lexical items can convey toxicity, inspired by work in pragmatics and sociolinguistics of rudeness (Dynel, 2015; Kasper, 1990, *inter alia*). Specifically, we manually split our full list of words into three distinct categories depending on the extent to which they carry profane or hateful meanings or are simply associated with hateful contexts.[5] We refer to the full set of words as TOXTRIG, for Toxicity Triggers, which is included in our released repository.[6]

---

[2] `https://github.com/XuhuiZhou/Toxic_Debias`

[3] Our definition of "bias" is specific to the social biases in toxic language detection datasets, grounded as lexical and dialectal biases; see Blodgett et al. (2020) for a detailed investigation of the term "bias".

[4] `https://tinyurl.com/list-of-bad-words`

[5] We note, however, that this categorization is in itself subjective.

[6] `https://github.com/XuhuiZhou/Toxic_Debias/blob/master/data/word_based_bias_list.csv`

Non-offensive minority identity mentions (NOI) refers to descriptive mentions of minoritized demographic or social identities (e.g., *gay*, *female*, *Muslim*). While these mentions are not usually inherently offensive by themselves, they are often found in offensive statements that are hateful towards minorities (Dixon et al., 2018). We detect these identity mentions in text using a list of 26 regular expressions.

Possibly offensive minority identity mentions (OI) are mentions of minoritized identities that could denote profanity or hate depending on pragmatic and contextual interpretations. This includes slurs and objectifying outdated terms to refer to minority groups, which are usually understood as attacks. Additionally, this includes *reclaimed* slurs (*queer*, *n*gga*), which connote less offensive intent when spoken by in-group members compared to out-group members (Croom, 2013).

Possibly offensive non-identity mentions (ONI) contains swear words and other profanities, which are usually offensive but not associated to any social groups (e.g., *f*ck*, *sh*t*). Note that the pragmatic interpretation of these words is not necessarily always toxic or offensive (Dynel, 2012), as they are often used to convey closeness between the speaker and listener or emphasize the emotionality of a statement (e.g., second example in in Figure 1).

2.2 Dialectal Biases (AAE)

Current toxic language detection systems also associate higher toxicity with dialectal markers of African American English (AAE; Sap et al., 2019; Davidson et al., 2019). Since AAE is a variety of English that is common among African Americans and often signals a cultural identity in the US (Green, 2002), this dialect-based racial bias causes speech by Black authors to be suppressed more often than non-Black authors (Sap et al., 2019), thereby exacerbating racial inequality (Rosa, 2019).

In our experiments, we estimate the dialect of a tweet using a topic model from Blodgett et al. (2016). This model was trained on 60M tweets, where the dialect of the tweet was inferred from the model coordinates, which yielded a probability of a tweet being in one of four dialects (African-American English, white-aligned English, Hispanic, and other). In this study, we only focus on African-American English (AAE) and white-aligned English (WAE) tweets; both definitions are based on US English, as per Blodgett et al. (2016).[7] Our experiments either use the probability of a tweet being in these dialects, or assign tweets their estimated-most-probable dialect.

2.3 Dataset for Toxic Language Detection

We focus our analyses on a widely used hate speech dataset of English tweets (Founta et al., 2018). The tweets were collected using a multi-round bootstrapping procedure, and were labeled out of context[8] for toxic language. We focus on the 86k tweets that are annotated as hateful, abusive, or neither and discard those labelled as spam. We aggregate the abusive and hateful labels into a single *toxic* category, yielding 32k toxic and 54k non-toxic tweets.[9]

3 Debiasing Methods

We consider two types of debiasing methods from current literature. The first type addresses known, pre-defined biases—such as lexical and dialectal biases for hate speech detection, via a model-based approach involving additional training objectives (§3.1). In contrast, the second type is agnostic to prior knowledge about biases, and instead filters out examples that appear "too easy" and might hence contain spurious correlations (§3.2).

3.1 Debiased Training for Pre-Defined Toxicity Biases

We use the LEARNED-MIXIN method of Clark et al. (2019), which achieved high out-of-distribution (OOD) performance on several NLU tasks, for debiased training. This method trains an ensemble containing a *bias-only* model which only uses pre-defined features corresponding to known biases, and a *full* model which uses all features. Intuitively, the ensemble encourages the full

---

[7] We avoid using disputed terms such as *general American English*, *standard American English*, or *mainstream US English*, which are frequently used for WAE, since we believe that no dialect should be privileged with the designation "general", "standard", or "mainstream" (Rosa, 2019).

[8] Only the tweet text—no profile information or conversational context—was shown to annotators.

[9] We also explored using another widely used hate speech dataset (Davidson et al., 2017), which collected tweets using a seed list of swear words and slurs. However, in line with findings by Xia et al. (2020), debiasing led to degenerate behavior due to the data collection process, as discussed in Appendix B.

model to rely more on features unrelated to the biases. Once trained, the bias-only model is discarded, and only the "bias-free" full model is used for inference, following Clark et al. (2019).

Bias-only model Given its effectiveness on bag-of-words (BoW) features, we use an SVM classifier as the lexical-bias-only model. For example, the TOXTRIG-only model counts the frequency of TOXTRIG words in each tweet. Our dialectal-bias-only model uses the probability of dialects (AAE, WAE, Hispanic, and other) obtained from a dialect detector (Blodgett et al., 2016) as features in a SVM classifier.

Full model We fine-tune a RoBERTa-large classifier (Liu et al., 2019), a state-of-the-art classifier for the toxicity detection task. See Appendix A.1 for more modeling details.

Note that we only consider the LEARNED-MIXIN-ONI and LEARNED-MIXIN-TOXTRIG models for lexical debiasing, due to poor accuracies of the bias-only models for NOI and OI.[10]

3.2 Data Filtering for Spurious Biases

In addition to debiasing methods that handle known biases, we also explore automated approaches which filter out instances exhibiting unspecified, spurious biases. Specifically, we describe below two data selection methods that have shown strong OOD performance.

AFLite (Bras et al., 2020) is an algorithm based on the key intuition that examples predicted correctly by the simplest methods likely exhibit spurious biases. An ensemble of simple linear classifiers is trained and tested on different partitions of the data; test instances which are "predictable", or classified correctly by most classifiers in the ensemble are discarded. The algorithm is iterative, and is repeated until a target data size is achieved. Models trained on this filtered dataset achieve higher performance on OOD and adversarially constructed test sets, compared to the original model, on several text and image classification datasets. This indicates a reduction in spurious biases in the filtered data.

DataMaps (Swayamdipta et al., 2020) show the presence of distinct regions in a dataset—namely, easy, hard and ambiguous—defined with respect to a given model. These regions are discovered based on the training dynamics of a model, determined by the model's confidence in the true class, for each example, as well as the variability of this confidence, throughout training epochs. Swayamdipta et al. (2020) show that training exclusively on the hard and ambiguous regions of the data results in high OOD performance, indicating lower prevalance of spurious biases. The easy region is the largest in size for RoBERTa; however, experiments showed that training exclusively on these examples hurt OOD generalization on different NLU tasks. Following this work, we create DataMaps-Easy, DataMaps-Ambiguous, and DataMaps-Hard subsets for our dataset.

Following Swayamdipta et al. (2020), we set the target filtered subset size to 33% of the original training set for both filtering methods, but our filtering additionally preserved the original label proportions. We then fine-tune a RoBERTa-large classifer on these filtered subsets; see Appendix A.2 for more details.

4 Experiments: Lexical Biases

We investigate the effect of debiasing approaches (§3) on removing lexical biases in hate speech detection. First, we discuss the evaluation framework for measuring bias reduction (§4.1). We present quantitative (§4.2) and qualitative (§4.3) results on lexical bias removal for all debiasing approaches, and OOD evaluation for debiased training methods (§4.4). See Appendix A.3 for hyperparameters and other experimental settings.

4.1 Evaluation Framework

We report the performance of all models as overall accuracy and F_1 with respect to the toxic class. Given that current hate speech systems tend to rely heavily on the presence of NOI, OI, and ONI mentions (§2.1) for labeling text as toxic, we use false positive rate (FPR) over each of these categories to measure the degree of bias in the model, following Hardt et al. (2016) and Xia et al. (2020). Specifically, we report the FPR of a model on tweets containing NOI (FPR_{NOI}), OI (FPR_{OI}), and ONI (FPR_{ONI}), as well the F_1 corresponding to each of these classes. Intuitively, the lower the FPR_*, the

---

[10] The NOI and OI bias-only models reach 63% and 67% accuracy, respectively, which is empirically hard for the ensemble to use. This is likely due to low coverage in the train set of those categories (4.43% NOI and 4.25% OI).

		R_{NOI} ↓	R_{OI} ↓	R_{ONI} ↓
	Original	0.0445	0.2641	0.6718
33% train	Random	0.0345	0.2603	0.6683
	AFLite	0.0434	0.2458	0.6016
	DataMaps-Ambig.	0.0126	0.1968	**0.5839**
	DataMaps-Hard	**0.0081**	**0.1853**	0.5849
	DataMaps-Easy	0.0772	0.3661	0.7720

Table 1: Lexical associations between toxicity and ToxTrig mentions in the original dataset (Founta et al., 2018) and various filtered counterparts. Random, AFLite, and DataMaps all contain only 33% of the original data after filtering. Lower Pearson R correlation value indicates less superficial patterns in the dataset, i.e., less bias. ***Takeaway:*** The hard and ambiguous subsets given by DataMaps contain the lowest amount of lexical associations, indicated in boldface.

less the model infers lexical associations for toxicity, and hence is less biased.

Evaluation for Filtered Datasets We additionally consider metrics based on spurious lexical associations for data filtering approaches. This measures prevalence of spurious surface patterns in the filtered datasets, which might propagate to models trained on the data. Specifically, we report the Pearson's correlation between the gold standard toxicity label and whether or not it contains NOI, OI, or ONI mentions. These correlations are denoted as R_{ONI}, R_{NOI}, and R_{OI}, respectively; lower values indicate reduction in lexical biases.

Baselines We consider comparison against two natural baselines: a vanilla RoBERTa-large classifier trained on the original dataset (Original). We also consider a baseline trained on a random selection of the training data (Random), for comparison with data filtering methods for debiasing. Each subset is trained on 33% of the training data.

4.2 Results for Lexical Bias Reduction

First, we measure the reduction in lexical biases in filtered datasets, as given by AFLite and DataMaps. As shown in Table 1, subsets given by AFLite and the ambiguous and hard regions produced by DataMaps reduce the overall associations between ToxTrig words and toxicity, compared to the original and random baselines; DataMaps-Hard has the largest reduction. On the other hand, as expected, DataMaps-Easy shows an *increased* association between ToxTrig mentions and toxicity, showing that the these examples display overt lexical biases.

Table 2 shows results for lexical bias reduction using both debiased training approaches, as well as models trained on datasets filtered using AFLite and all three regions from DataMaps. Both debiased training approaches, LMixin-ONI and LMixin-ToxTrig, reduce FPR_{ONI} as well as FPR_{OI} by a large amount. However, both approaches also hurt in-distribution test performance, indicating that ONI and other ToxTrig features are essential for good performance.[11] In contrast, the models trained on hard and ambiguous subsets from DataMaps both preserve in-distribution performance, even though they are trained only a third of the original data. They also reduce the rate of falsely predicting NOI mentions as toxic (FPR_{NOI}), while not showing much improvement for ONI and maintaining FPR_{OI} of the original baseline.

Surprisingly, the model trained on the easy subset from DataMaps shows good bias reduction on the NOI and ONI categories, while matching the random selection baseline for OI. This is despite DataMaps-Easy showing an increased association between ToxTrig mentions and toxicity (Table 1). Notably, the F_1 for all categories suffers under this model, indicating that it is less competent than the baseline. These results suggest that reduced associations in the data might not necessarily lead to debiased models trained on the same data. Overall, no single approach outperforms all others across different categories for lexical debiasing.

4.3 Qualitative Analysis

A qualitative study of the Founta et al. (2018) test set shows the presence of many annotation errors. We show three representative annotation errors in Table 3. The first example contains an atypical example of toxicity, towards white folks, which the annotators might have been unaware of. It also contains a link which annotators had access to, but not models. The second contains the word *p*ss* which the annotators may have relied for their assessment. The third encourages violence/abuse towards an identity which isn't typically the target of violence. Interestingly, the DataMaps-Easy predictions agree with all the gold standard annotations; perhaps such annotation errors and ambiguity are responsible for the performance discussed

---

[11] When we combine the bias-only model and the full model, we obtain competitive performance (see Appendix A.4).

		Test (12893)		NOI (602)		OI (553)		ONI (3236)	
		Acc.↑	F_1 ↑	F_1 ↑	FPR_{NOI} ↓	F_1 ↑	FPR_{OI} ↓	F_1 ↑	FPR_{ONI} ↓
	Vanilla	$94.21_{0.0}$	$92.33_{0.0}$	$89.76_{0.3}$	$10.24_{1.3}$	$98.84_{0.1}$	$85.71_{0.0}$	$97.34_{0.1}$	$64.72_{0.8}$
	LMIXIN-ONI	$89.65_{1.5}$	$85.59_{2.5}$	$87.04_{1.1}$	$13.99_{1.5}$	$98.87_{0.0}$	$85.71_{0.0}$	$87.87_{4.5}$	$\mathbf{43.74}_{3.1}$
	LMIXIN-TOXTRIG	$90.44_{0.7}$	$86.94_{1.1}$	$85.47_{0.3}$	$11.15_{1.7}$	$97.64_{0.3}$	$\mathbf{71.43}_{0.0}$	$90.41_{1.8}$	$44.55_{1.5}$
33% train	Random	$94.07_{0.1}$	$92.18_{0.1}$	$89.48_{0.4}$	$9.33_{0.7}$	$\mathbf{98.93}_{0.0}$	$\mathbf{83.33}_{3.4}$	$97.40_{0.1}$	$67.15_{0.6}$
	AFLite	$93.86_{0.1}$	$91.94_{0.1}$	$\mathbf{90.21}_{0.4}$	$11.26_{1.1}$	$98.90_{0.0}$	$85.71_{0.0}$	$97.32_{0.1}$	$67.97_{3.4}$
	DataMaps-Ambig.	$94.33_{0.1}$	$92.45_{0.1}$	$89.16_{0.7}$	$7.39_{1.0}$	$98.87_{0.0}$	$85.71_{0.0}$	$\mathbf{97.54}_{0.0}$	$64.39_{1.4}$
	DataMaps-Hard	$\mathbf{94.50}_{0.0}$	$\mathbf{92.61}_{0.1}$	$89.54_{0.4}$	$6.26_{0.9}$	$98.84_{0.0}$	$85.71_{0.0}$	$97.43_{0.0}$	$61.95_{1.1}$
	DataMaps-Easy	$94.00_{0.1}$	$91.94_{0.2}$	$86.81_{0.6}$	$\mathbf{5.92}_{0.7}$	$98.87_{0.0}$	$\mathbf{83.33}_{3.4}$	$97.17_{0.1}$	$\mathbf{60.33}_{3.8}$

Table 2: Evaluation of lexical bias removal for all debiasing methods on the Founta et al. (2018) test set. Results show the mean and s.d. (subscript) of accuracy and F_1 across 3 runs, as well as F_1 and false positive rate exclusively on test examples containing specific TOXTRIG mentions—NOI, OI and ONI, along with the number of examples in each category. The lower the FPR$*$, the less the model infers lexical associations for toxicity. The first block shows debiased training approaches, along with the vanilla classifier, which are trained on the full dataset. The second block shows data filtering approaches, all trained on only 33% of the training data. Best performance in each block is boldfaced. ***Takeaway:*** While data filtering approaches achieve overall higher performance, debiased training approaches perform better on lexical bias reduction, in aggregate.

	Gold	DM-Hard	DM-Easy
@user THIS? LMAOOOOO...do yourself a solid and stay out of Black people's mentions and mind your caucasia...	🍃	☠	🍃
RT @user I wish I wasn't so annoying like I even piss myself off	☠	🍃	☠
@user If you want to attack people, attack fundamentalists of all faiths. Attack those who condemn 1.5bn people out of hand.	🍃	☠	🍃

Table 3: Examples of test set tweets with their gold-standard annotations and predictions from models trained on DataMaps-Hard (DM-Hard) and DataMaps-Easy (DM-Easy) subsets. ☠ denotes tweets with toxic labels, and 🍃 represents non-toxic labels. We anonymize the usernames to protect user privacy.

in §4.2. These annotation ambiguities might also impair our measurement for models' performance and debiasing effects, and expose a limitation of these datasets.

4.4 Adversarial Evaluation: ONI-Adv

To further study the reliance of debiased models on the ONI words, we use the test set from Dinan et al. (2019), which contains 3000 manually created sentences, 300 toxic. The toxic sentences barely contain any ONI mentions, e.g., *Cheese is made by french people who smell.* We call this test set ONI-Adv (for adversarial) since it challenges models with a reversal in the association between toxicity and offensive non-identity words (e.g., "*f*ck*", "*sh*t*").

We report F_1 for all models in Figure 2, which shows how well a model identifies toxicity in offensive tweets that do not contain overtly lexical cues of toxicity. The debiased training approaches improve over the baselines; data filtering methods do not. One reason for this might be that data filtering methods were trained on much less data than both LMIXIN models. Regardless, none of the models we test are good at predicting subtle, non-overt toxicity.

5 Experiments: Dialectal and Racial Biases

We test the efficacy of the bias reduction methods from §3 for dialectal bias (§2.2) reduction.

5.1 Dialectal Biases

For our dialectal bias experiments, we first infer the dialect of a tweet as described in §2.2. Then, analogous to the lexical bias evaluation, we quantify the dialectal debiasing using the Pearson's correlation between estimated probabilities of AAE and toxicity (R_{AAE}), and the false positive rates of models on AAE tweets (FPR_{AAE}). See Appendix A.3 for hyperparameters and other experimental settings.

Results in Table 4 show that almost all data filtering and debiasing methods reduce dialectal biases, with DataMaps-Easy as the exception (con-

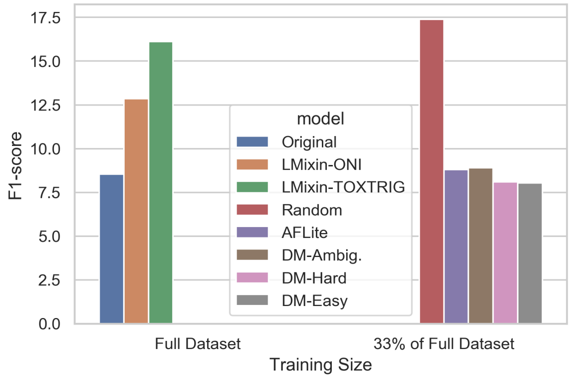


Figure 2: Challenge set evaluation for lexical biases, comparing all debiasing methods with baselines, using the ONI-Adv test set. ***Takeaway:*** F_1 (↑) measures show that all models perform poorly at identifying toxic text not containing overtly lexical cues of toxicity. In general, debiased training approaches outperform the original model on this challenge set, while data filtering is not as effective.

sistent with Table 1). Notably, DataMaps-Hard performs the best at dialectal debiasing, both in terms of toxicity-AAE correlation (R_{AAE}) and in terms of false flagging of toxicity (FPR$_{\text{AAE}}$). Interestingly, most models' decrease in false flagging is small, suggesting room for improvement.

5.2 Racial Biases

To quantify the real-world impact of dialect-based racial bias, we measure the rates of toxicity predicted by models on a corpus of tweets for which the race of authors is available, but not annotations of toxicity. Specifically, we consider the dataset released by Preoţiuc-Pietro and Ungar (2018), which consists of 5.4M tweets, collected from 4,132 survey participants (3,184 White, 374 African American) with self-reported race/ethnicity and Twitter user handles.[12]

We quantify our models' racial bias by measuring the difference in rates of flagging tweets by African American authors and those by white authors, following Sap et al. (2019).[13]

Listed in Table 5, our results show that automatic debiasing methods do not consistently decrease the racial discrepancy in flagging toxicity. Notably, the toxicity rates on tweets by African American authors—and the diferences compared to white authors—are similar across all debiasing methods and baselines, except for DataMaps-Easy, which shows the most racial bias in toxicity flagging. Surprisingly, DataMaps-Hard, which mitigated dialectal bias the best out of all debiasing methods, also shows high discrepancy between author races. Confirming previous results, this suggests that debiasing these systems requires more than automatic debiasing methods.

			Test	
		R_{AAE} ↓	F_1 ↑	FPR$_{\text{AAE}}$ ↓
	Vanilla	0.4079	$92.33_{0.0}$	$16.84_{0.3}$
	LMIXIN-Dialect	-	$92.26_{0.1}$	$16.07_{0.4}$
33% train	Random	0.4027	$92.18_{0.1}$	$16.67_{0.6}$
	AFLite	0.3577	$91.94_{0.1}$	$16.84_{0.8}$
	DataMaps-Ambig.	0.2965	$92.45_{0.1}$	$15.99_{0.4}$
	DataMaps-Hard	**0.2878**	**92.61**$_{0.1}$	**13.71**$_{0.2}$
	DataMaps-Easy	0.5347	$91.94_{0.2}$	$19.46_{2.8}$
	AAE-relabeled	0.3453	$91.64_{0.3}$	**12.69**$_{0.0}$

Table 4: Dialectal bias evaluation for all debiasing methods (§5), as well as the relabeling approach (§6) on the Founta et al. (2018) test set. We report F_1 and the false positive rate with respect to tweets in AAE (FPR$_{\text{AAE}}$), reflecting dialectal bias (lower is less biased), showing mean and s.d. (subscript) across 3 runs. (Top Block) Debiased training approaches, along with the vanilla classifier, are all trained on the full dataset. (Middle Block) Random, AFLite and DataMaps all are trained on only 33% of the training data. Best performance for each training set size is in boldface. ***Takeaway:*** Both debiasing approaches improve performance over baselines, with DataMaps-Hard proving the most effective at debiasing. (Bottom Block) AAE-relabeling results in a model which despite following a noisy process yields even larger improvements for dialectal debiasing.

6 Towards Data Relabeling

Based on our quantitative and qualitative analyses, we believe there still is room for improvement in debiasing hate speech detection. Therefore, we turn our attention to the role of label noise in datasets. Partly inspired by our qualitative analyses of debiased models' predictions, we design a proof-of-concept study where we automatically correct the label of tweets using a(n automatic) dialectal translation of the tweet, inspired by previous work showing that highlighting AAE tweets' dialect led them to be labeled as less toxic (Sap et al., 2019). We conclude this study by discussing the limitations and ethical implications of the synthetic data, and cautioning against its real-world application.

---

[12] For efficiency, we randomly select 12k tweets from the dataset as the OOD test set.

[13] Note that we assume that authors from all races have the same likelihood of writing toxic language.

		W-Tox.	AA-Tox.	Δ ↓	AA/W↓
	Original	7.24	12.61	5.37	1.74
	LMixin-Dialect	7.50	12.55	5.06	1.67
33% train	Random	8.28	13.24	4.96	1.60
	AFLite	7.32	11.64	4.33	1.59
	DataMaps-Ambig.	6.75	12.17	5.42	1.80
	DataMaps-Hard	6.36	11.67	5.31	1.84
	DataMaps-Easy	8.46	16.30	7.83	1.94
	AAE-relabeled	6.93	10.60	**3.67**	**1.53**

Table 5: Racial disparity in toxicity prediction reported on Preoţiuc-Pietro and Ungar (2018). **W-Tox.** indicates % of white users' tweets being flagged as toxic, **AA-Tox.** indicates % of African American users' tweets being flagged as toxic, Δ refers to the difference between AA-Tox. and W-Tox., and **AA/W** refers to the ratio between AA-Tox. and W-Tox. ***Takeaway:*** Methods generally fail in debiasing on this OOD test set except the relabeling approach shows some benefit.

Focusing on dialectal bias, our key assumption is that an AAE tweet and its corresponding WAE version should have the same toxicity label, therefore toxic AAE tweets whose WAE versions are non-toxic are candidates for label correction.[14]

However, gold-standard translations of AAE to WAE would require qualified translators, and automatic AAE-to-WAE translation systems do not exist, to the best of our knowledge. Therefore, we create a proof-of-concept study—we set up a AAE to WAE "translation" system using the few-shot capabilities of the GPT-3 language model (Brown et al., 2020). Under this mechanism, we prompt GPT-3 with four translation pairs (taken from Spears, 1998) and an AAE tweet from our training data, and generate its WAE "translation". The list of prompts, as well as further details, are provided in Appendix C. Note that we do *not* recommend this approach to build large scale parallel data for dialects, as discussed under ethical implications and limitations.

Next, as per our heuristic, we only relabel toxic AAE tweets whose WAE translation is predicted as non-toxic by either our vanilla classifier trained on the original Founta et al. (2018) dataset, or an identical classifier trained on the WAE translated tweets. The resulting dataset (AAE-relabeled) is the same size as the original dataset, but with 954 (12%) out of 8260 toxic AAE tweets relabeled as non-toxic (examples in Table 6). To assess the validity of the relabeling, the first three authors manually annotated toxicity of 50 randomly selected relabeled tweets. On average, authors agreed with 84% of the relabeling decisions.

Then, we evaluate the dialectal bias of AAE-relabeled and quantify the dialect and racial prediction biases from a RoBERTa-large classifier trained on AAE-relabeled, following §5. As shown in the last row of Table 4, this relabeling scheme decreases dialectal bias more than any other debiasing method, specifically as measured by the FPR on AAE tweets, with one point drop in F_1 score. The F_1 score on the "gold" test data (Table 4) are not fully reliable, as test data contain label biases and better performance could come from exploiting these biases. As shown in Table 5, the model trained on AAE-relabeled has the lowest racial disparity in toxicity flagging rates compared to all other methods.

These results highlight that debiasing methods are much less effective at mitigating dialectal dataset biases compared to data relabeling. For future investigations, we recommend obtaining human-written AAE-WAE pairs (e.g., as done by Groenwold et al., 2020). Additionally, to ensure less biased toxicity labeling, we recommend recruiting AAE speakers or experts for avoiding over-identification of AAE-markers as toxic (Spears, 1998; Croom, 2013). Alternatively, we recommend exploring more holistic representations of social biases or toxicity (e.g., Social Bias Frames; Sap et al., 2020).

Ethical Implications & Limitations

The above synthetic setting is meant to illustrate the role of labeling quality on biases in annotations. We strongly caution against using this approach in real-world applications, such as building parallel datasets for dialects. First, due to how its training data was selected, GPT-3 has likely not been exposed to many African American English varieties during training (Jo and Gebru, 2020). Second, pretrained language models are known to generate toxic language at non-trivial rates (Gehman et al., 2020), which could cause differential toxicity in the translations.

7 Related Work

Debiasing Toxicity Detection As the popularity of hate speech and toxic language detection sys-

---

[14] Note that this assumption does not hold for lexical items, because substituting lexical items (e.g., swapping a minority mention for a majority mention) would drastically change the denotational meaning of the sentence.

AAE	GPT-3 WAE Translation	Gold	New
RT @user I can't stand a bad texter bruh like don't be mad if I forget about yo ass	RT @user I can't stand a bad texter bro like don't be mad if I forget about you	☠	🍃
RT @user Retweet if you fuck with this!!!!	RT @user Retweet if you like this!	☠	🍃
RT @user That nigga needs anger management	RT @user That guy needs anger management	☠	🍃
RT @user oh fucking hell take a day off man	RT @user oh fuck take a day off man	☠	☠

Table 6: Examples of AAE tweets with their GPT-3 based WAE translation, and original gold standard and new annotations based on AAE-relabeled. For the first three tweets, the (biased) gold labels are changed by models predicting the new labels on their WAE translations. ☠ indicates presence of toxicity, and 🍃 represents non-toxic. We anonymize the usernames to protect user privacy.

tems has grown, several biases have been found in dataset and models, spurring various debiasing efforts to mitigate these individual biases (e.g., gender bias, racial bias; Park et al., 2018; Sap et al., 2019; Davidson et al., 2019). Some work tackles identity-based biases, e.g., using data re-balancing (Dixon et al., 2018), or adversarial feature learning (Vaidya et al., 2019). Less work has tackled racial or dialectal bias. Notably, Xia et al. (2020) use adversarial training to prevent the model from associating toxicity with AAE, showing only small improvements in fairness. Based on those results, we do not explore adversarial methods, opting instead for ensemble-based methods of predefined bias reduction. In contemporary work, Mozafari et al. (2020) use a re-weighting mechanism, which shows some effects in debiasing racial bias. We leave it for future work to evaluate this method in our setting. In contrast to all previous work, our experiments also measure the effectiveness of bias-agnostic methods.

Other General Debiasing Methods Several approaches for debiasing NLU tasks have been proposed lately. Some approaches rely on adversarial training to remove protected attributes (e.g. gender or race), from a model's internal representations (Zhang et al., 2018; Wang et al., 2019; Xia et al., 2020). Other approaches include confidence regularization (Utama et al., 2020), as well as other product of expert approaches (He et al., 2019; Karimi Mahabadi et al., 2020) similar to the debiased training approach from Clark et al. (2019), which is the only debiased training we employ due to its relatively strong performance.

8 Conclusion

We investigate whether toxic language detection systems can be debiased using recently introduced methods for debiasing text classification in NLU tasks. Focusing on two types of biases, lexical and dialectal, our experiments show that these methods face significant challenges in reducing the biased behavior in toxicity detectors. This indicates that biases in toxic language detection might be different in nature compared to spurious associations studied in typical NLU settings. We studied a synthetic scheme for relabeling examples with potential dialectal biases; our results indicate that correcting noisy labels results in better bias reduction. Our findings suggest that instead of solely relying on development of automatic debiasing for existing, imperfect datasets, future work focus primarily on the quality of the underlying data for hate speech detection, such as accounting for speaker identity and dialect. Indeed, such efforts could act as an important step towards making systems less discriminatory, and hence safe and usable.

Acknowledgments

We thank the anonymous reviewers and Laura Vianna for helpful comments on this work. This research was supported in part by NSF grants 1813153 and 1714566.

References

Su Lin Blodgett, Solon Barocas, Hal Daumé, III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proc. of ACL*.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In *Proc. of EMNLP*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proc. of EMNLP*.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *Proc. of ICML*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proc. of NeurIPS*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proc. of EMNLP*.

Adam M Croom. 2013. How to do things with slurs: Studies in the way of derogatory words. In *Language & communication*.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Abusive Language Workshop (at ACL)*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*.

Thiago Dias Oliva, Dennys Marcelo Antonialli, and Alessandra Gomes. 2020. Fighting hate speech, silencing drag queens? artificial intelligence in content moderation and risks to lgbtq voices online. In *Sexuality & Culture*.

Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. In *Proc. of EMNLP*.

Lucas Dixon, John Li, Jeffrey Scott Sorensen, Nithum Thain, and L. Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proc. of AES*.

Marta Dynel. 2012. Swearing methodologically : the (im)politeness of expletives in anonymous commentaries on youtube. In *Journal of English Studies*.

Marta Dynel. 2015. The landscape of impoliteness research. In *Journal of Politeness Research*.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proc. of WSM*.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of EMNLP*.

Lisa Green. 2002. *African American English: A Linguistic Introduction*. Cambridge University Press.

Sophie Groenwold, Lily Ou, Aesha Parekh, Samhita Honnavalli, Sharon Levy, Diba Mirza, and William Yang Wang. 2020. Investigating African-American vernacular english in Transformer-Based text generation. In *Proc. of EMNLP*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proc. of NAACL*.

Jessica Guynn. 2020. What civil rights groups want from facebook boycott: Stop hate speech and harassment of black users.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *Proc. of NeurIPS*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *EMNLP Workshop on Deep Learning Approaches for Low-Resource NLP*.

Eun Seo Jo and Timnit Gebru. 2020. Lessons from archives: strategies for collecting sociocultural data in machine learning. In *Proc. of FAT*.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proc. of ACL*.

Gabriele Kasper. 1990. Linguistic politeness: current research issues. In *Journal of Pragmatics*. Elsevier.

Jae Yeon Kim, Carlos Ortiz, Sarah Nam, Sarah Santiago, and Vivek Datta. 2020. Intersectional bias in hate speech and abusive language datasets.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. In *arXiv preprint arXiv:1907.11692*.

Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2020. Hate speech detection and racial bias mitigation in social media based on bert model. In *PLOS ONE*. Public Library of Science.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proc. of EMNLP*.

Daniel Preoţiuc-Pietro and Lyle Ungar. 2018. User-level race and ethnicity predictors from twitter text. In *Proc. of COLING*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proc. of EMNLP*, pages 2383–2392.

Sarah T Roberts. 2019. *Behind the screen: Content moderation in the shadows of social media*. Yale University Press.

Jonathan Rosa. 2019. *Looking like a language, sounding like a race*. Oxford University Press.

Jonathan Rosa and Nelson Flores. 2017. Unsettling race and language: Toward a raciolinguistic perspective. In *Language In Society*. Cambridge University Press.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2017. Measuring the reliability of hate speech annotations: the case of the european refugee crisis. In *NLP 4 CMC Workshop*.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proc. of ACL*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proc. of ACL*.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Li Zilles, Yejin Choi, and Noah A Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. In *Proc. of CoNLL*.

Arthur K Spears. 1998. African-American language use: Ideology and so-called obscenity. In *African-American English: Structure, History and Use*. Routledge New York.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proc. of EMNLP*.

Björn Technau. 2018. Going beyond hate speech: The pragmatics of ethnic slur terms. *Lodz Papers in Pragmatics*, 14(1):25–43.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Mind the trade-off: Debiasing NLU models without degrading the in-distribution performance. In *Proc. of ACL*.

Ameya Vaidya, Feng Mai, and Yue Ning. 2019. Empirical analysis of multi-task learning for reducing model bias in toxic comment detection. In *Proc. of ICWSM*.

Bertie Vidgen, Helen Margetts, and Alex Harris. 2019. How much online abuse is there? In *Alan Turing Institute*.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and V. Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proc. of ICCV*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing.

Mengzhou Xia, Anjalie Field, and Yulia Tsvetkov. 2020. Demoting racial bias in hate speech detection. In *Proc. of Social NLP*.

Danyaal Yasin. 2018. Black and banned: Who is free speech for?

Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating unwanted biases with adversarial learning. In *Proc. of AES*. Association for Computing Machinery.

Appendix

A Further Details for Models

A.1 Model Debiasing

The LEARNED-MIXIN ensemble allows the model to explicitly determine how much to trust the bias given the input:

$$\hat{p_i} = \text{softmax}\{\log(p_i) + g(\mathbf{x}_i) \log b_i\}$$

where $\mathbf{x}_i$ is the ith input text, p_i and b_i is the toxicity prediction produced by RoBERTa, and bias-only model respectively, and g is a parametric function, which is defined as softplus($\mathbf{w} \cdot \mathbf{h}_i$), where $\mathbf{w}$ is a learned vector, $\mathbf{h}_i$ is the last hidden layer of the model for example $\mathbf{x}_i$, and the softplus$(x) = \log(1 + \exp x)$. To prevent the LEARNED-MIXIN ensemble from ignoring b_i, Clark et al. (2019) add an entropy penalty (H) to the loss:

$$R = \alpha H(\text{softmax}\{g(\mathbf{x}_i) \log b_i\})$$

Where $H(z) = -\sum_j z_j \log z_j$ is the entropy and α is a hyperparameter.

A.2 Data Filtering

For the data filtering methods, we first filter data to 50% of the original data as in Swayamdipta et al. (2020). Then we further downsample the dataset to 33% of the original data to control that each training set has the same toxic ratio as the original training set. This step is to avoid confounding our results with different toxic ratio among different training sets.

A.3 Training Settings

For all the experiments, we fine-tune RoBERTa-large (Liu et al., 2019) over the corresponding corpus with one GTX2080 Ti. We use the default hyperparameters as provided in the `HuggingFace Transformers` library (Wolf et al., 2019), with two major changes: we use a learning rate of 10^{-5} and 8 batch size in all experiments.

A.4 Prediction Combining with Bias-only Model

To prevent the possibility that our LMIXIN-TOXTRIG/ONI is not well trained, thus resulting in the decrease of models' in-distribution performance, we use the joint-prediction from the main and bias-only model to infer the in-distribution test set and they obtain 94.15% and 94.17% accuracy, respectively. This is competitive performance as shown in Table 2.

B Alternative Dataset of Toxic Language

Davidson et al. (2017) collected data from Twitter, starting with 1,000 terms from HateBase (an online database of hate speech terms) as seeds, which the process relies on lexical biases. We find that performing data filtering methods over this dataset leads to degenerate behaviour. Specifically, as shown in Table 7, the easy region demonstrates least spurious correlation due to its heavily skewed class distribution, which further prevent us from downsampling to control the toxic ratio. We also train LMIXIN-TOXTRIG and LMIXIN-dialect over the dataset. Table 8 shows that FPR of the debiased model increase instead except for the OI category and Table 9's results behave in-line with Table 4.

C Few-shot AAE-to-WAE Translation

Note that we do *not* recommend the following approach to build large scale parallel data for dialects, as discussed under ethical implications and limitations (§6).

We use GPT-3 (Brown et al., 2020) to create a few-shot AAE-to-WAE translation system, using the following set of example translation pairs drawn from Spears (1998):

AAE: Get your triflin' ass out of here.
WAE: Get your trifling self out of here.

AAE: I saw his ass yesterday.
WAE: I saw him yesterday.

AAE: His ass is gonna get fried.
WAE: He is gonna get fried

AAE: Wassup, nigga?
WAE: What's up bro?

AAE: ⟨tweet⟩
WAE:

Note that Spears (1998) refers to WAE as White language varieties, and deals with English prevalent in the United States.

We prepend the formatted example pairs to each AAE tweet in our training data, and generate the translation from GPT-3 using top-0.95 nucleus sampling with a temperature of 0.5. Prompts, formatting, and generation parameters were chosen based on manual inspection of the output.

	Toxic Ratio	$R_{\text{NOI}} \downarrow$	$R_{\text{OI}} \downarrow$	$R_{\text{ONI}} \downarrow$	$R_{\text{AAE}} \downarrow$
Original†	0.8308	0.0287	0.4320	0.2610	0.4061
Random	0.8312	0.0288	0.4312	0.2621	0.4011
AFLite	0.7669	0.0342	0.4708	0.2835	0.4236
DataMaps-Ambig.	0.6736	0.0493	0.4683	0.3230	0.4445
DataMaps-Hard	0.6645	0.0521	0.4533	0.3190	0.4426
DataMaps-Easy	0.9972	0.0135	0.0771	0.0396	0.0928

Table 7: Lexical and dialectal associations between toxicity in the original dataset (Davidson et al., 2017) and various filtered counterparts. Random, AFLite, and DataMaps all contain only 50% of the original data after filtering. (We could not perform downsampling on these datasets due to their heavily skewed label distribution.) Lower Pearson R correlation value indicates less superficial patterns in the dataset, thus are less biased. The easy subset gives the best results here are due to its severe inbalanced label distribution.

	Test		NOI		OI		ONI	
	Acc.↑	$F_1 \uparrow$	$F_1 \uparrow$	$\text{FPR}_{\text{NOI}} \downarrow$	$F_1 \uparrow$	$\text{FPR}_{\text{OI}} \downarrow$	$F_1 \uparrow$	$\text{FPR}_{\text{ONI}} \downarrow$
Original	96.37	97.81	96.42	25.00	99.86	57.14	99.57	63.64
LMIXIN-TOXTRIG	96.15	97.69	96.19	28.57	99.78	42.86	99.28	72.73

Table 8: Lexical bias removal evaluation for debiasing methods. Original refers to the model trained over the full training set. The test set is further categorized into tweets that contained relevant TOXTRIG words. F_1 indicates models' performance while the false positive rate (FPR$_*$) reflects models' bias. The lower the FPR$_*$ is, the less biased the model tend to be.

Debiasing Method		Test		
	R_{AAE}	Acc. ↑	$F_1 \uparrow$	$\text{FPR}_{\text{AAE}} \downarrow$
Original	0.4079	96.37	97.81	24.76
LMIXIN-Dialect	-	96.48	97.88	22.86

Table 9: Dialectal bias evaluation for all debiasing methods, on both in-distribution test set as well as out-of-distribution dialect and race priming test sets. In addition to accuracy and F_1, we report the false positive rate with respect to tweets in AAE (FPR_{AAE}), reflecting dialectal bias (lower is less debiased). Each method is based on a RoBERTa-large classifier.

Adaptive Fusion Techniques for Multimodal Data

Gaurav Sahu, Olga Vechtomova
University of Waterloo
`{gsahu, ovechtom}@uwaterloo.ca`

Abstract

Effective fusion of data from multiple modalities, such as video, speech, and text, is challenging due to the heterogeneous nature of multimodal data. In this paper, we propose adaptive fusion techniques that aim to model context from different modalities effectively. Instead of defining a deterministic fusion operation, such as concatenation, for the network, we let the network decide "how" to combine a given set of multimodal features more effectively. We propose two networks: 1) Auto-Fusion, which learns to compress information from different modalities while preserving the context, and 2) GAN-Fusion, which regularizes the learned latent space given context from complementing modalities. A quantitative evaluation on the tasks of multimodal machine translation and emotion recognition suggests that our lightweight, adaptive networks can better model context from other modalities than existing methods, many of which employ massive transformer-based networks.[1]

1 Introduction

Multimodal deep learning is an active field of research, where for a single event, one has information across multiple modalities, such as video, speech, and text. Human brains can easily and perpetually perceive the context of an event from such heterogeneous data; however, it is not a trivial task for a computer system. In order for the machine to gain a contextual understanding, heterogeneous inputs must be combined first. Combining, or more precisely, *fusing* multimodal inputs is, thus, a vital step for any multimodal task. Naturally, a better fusion method will help a multimodal system learn better, ultimately enhancing its performance for a given task.

The most common fusion technique used in the literature involves the concatenation of representations from all the available modalities. However, this results in a shallow network (Ngiam et al., 2011), and the network focuses more on learning intra-modal features, ignoring inter-modal dynamics altogether. Later, Zadeh et al. (2017) proposed Tensor Fusion Network (TFN), which models the unimodal, bimodal, and trimodal interactions using a 3-fold Cartesian product. TFN performs better than simple concatenation; however, it imposes high computational requirements since it projects all the information from input modalities to a dense 3-D space as-is, without any prior information extraction. The computational overhead grows exponentially with respect to the dimensionality of unimodal features. Liu et al. (2018) proposed a low-rank multimodal fusion technique (LMF) to address the previous problem. Such fusion techniques are useful but often result in a complex architecture. Moreover, the fusion methods mentioned above focus only on combining individual unimodal features rather than combining *and* extracting useful information simultaneously. This means that the final predictor module (decoder in a Seq2Seq network (Sutskever et al., 2014), for example) bears an additional responsibility of identifying useful signals to focus on.

This paper addresses these issues by proposing adaptive fusion techniques that allow the model to decide "how" to combine multimodal data more effectively for an event. The first technique, Auto-Fusion, learns to compress multimodal information while preserving as much meaning as possible. The second technique, GAN-Fusion, employs an adversarial network that regularizes the learned latent space for a given target modality complying with the information presented by complementary modalities. Since our models are generic, the need to specify a pre-determined fusion operation

---

[1] Code for our experiments: `https://github.com/demfier/philo/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3156–3166
April 19 - 23, 2021.

such as Cartesian product is alleviated, and this further incentivizes the network to model multimodal interactions by itself. Moreover, our techniques are lightweight relative to the existing heavier counterparts (Vaswani et al., 2017; Grönroos et al., 2018), thereby preventing unnecessary computational load.

We evaluate our models on three benchmark datasets:

1. the How2 dataset (Sanabria et al., 2018) with multimodal input for English-Portuguese translation.
2. the Multi30K dataset (Elliott et al., 2016), which contains parallel corpora for multimodal machine translation, and
3. the IEMOCAP dataset (Busso et al., 2008) which contains multimodal data for emotion detection.

A quantitative evaluation shows that our models outperform the existing state-of-the-art methods in terms of BLEU scores (Papineni et al., 2002) for machine translation and Precision, Recall, and F1-score for emotion recognition. Our ablation studies also indicate that the learned multimodal representations are robust; they perform reasonably well even after removing information from a target modality. We now summarize our main contributions as follows:

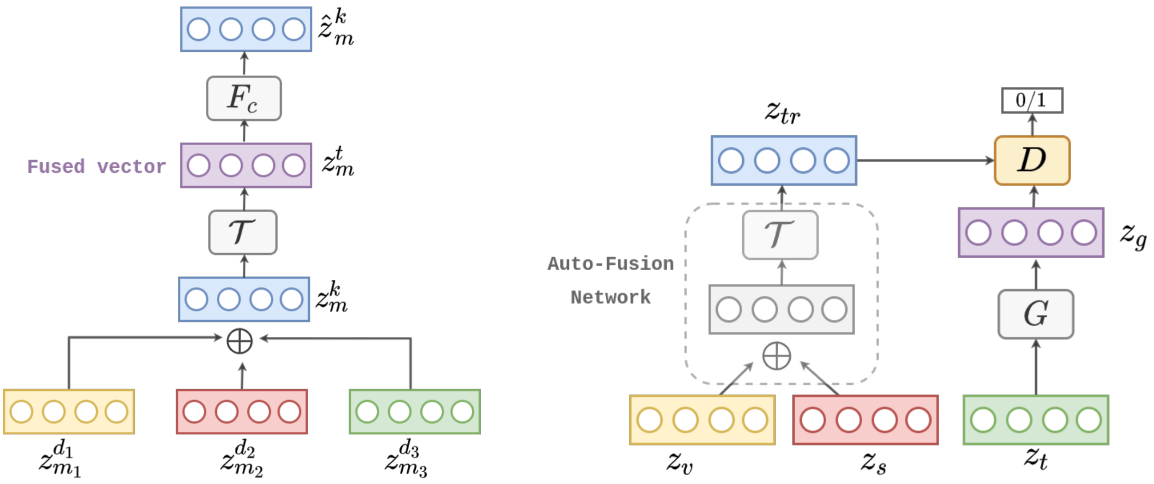
1. We propose two lightweight, adaptive techniques for better multimodal fusion of data: Auto-Fusion and GAN-Fusion.
2. We propose a multi-task framework for end-to-end training of multimodal networks (for both classification and generation).

The rest of the paper is structured as follows: Section 2 covers relevant work, Section 3 discusses the proposed methodologies and overall architecture, Section 4 describes the experimental setup, Section 5 shows results, and Section 6 contains our concluding remarks.

2 Related Work

In this section, we briefly review previous work related to our task. Most earlier works in multimodal deep learning focus on traditional shallow classifiers such as support vector machines (Cortes and Vapnik, 1995) and Naive Bayes classifiers (Morade and Patnaik, 2015) to exploit bimodal data. Inspired by the success of deep learning over the last decade across multiple tasks, Ngiam et al. (2011) train end-to-end deep graph neural networks to reconstruct missing modalities during inference. They demonstrate that better features for one modality can be learned if relevant data from different modalities is available at training time; however, they employ simple concatenation for fusion. Hence, the joint representation learned is shallow and is not guaranteed to model intermodal interactions. Their findings were later verified by Srivastava and Salakhutdinov (2012), who use a Deep Boltzmann Machine (Salakhutdinov and Hinton, 2009) to generate data from the image and text modality. Huang et al. (2018) construct a multilingual common semantic space to achieve better machine translation performance by extending correlation networks (Chandar et al., 2016). They use multiple non-linear transformations to reconstruct sentences from one language to another repeatedly and finally build a common semantic space for all the different languages. To address the shallowness exhibited by some earlier fusion methods, techniques such as TFN (Zadeh et al., 2017), LMF (Liu et al., 2018) and T2FN (Liang et al., 2019), were proposed that aim to capture *both* intra- and inter-modal dynamics simultaneously; however, the problem of effectively modelling context in multimodal samples remains unsolved.

More recently, Multimodal Transformer (MulT) (Tsai et al., 2019) was proposed to align data from different modalities implicitly. On a high-level, MulT leverages cross-modal attention modules for each modality, each of which is responsible for aligning (or attending to) the target modality vector with the complementary modalities'. It also imposes substantial computational overhead due to the use of transformer networks (Vaswani et al., 2017). Our methods, as discussed in detail in Section 3, use much simpler components. For instance, we use at most one attention module compared to multiple self-attention heads in a transformer. Variational Mixture-of-Experts Autoencoders (Shi et al., 2019), a class of deep generative multimodal frameworks, were employed to learn a synergic shared representation for multiple modalities; however, scaling of such a model for all the modalities (video, speech, and text) simultaneously and for a more complex task as multimodal machine transla-



(a) Auto-Fusion network

(b) GAN-Fusion Network

Figure 1: Proposed architectures. (a) Auto-Fusion network: Assuming that $z_{m_1}^{d_1}$, $z_{m_2}^{d_2}$, and $z_{m_3}^{d_3}$ represent the video, speech, and text latent vectors respectively, we first concatenate them to obtain z_m^k. It is then passed through $\mathcal{T}$ which outputs the "autofused" vector z_m^t. We then obtain the reconstructed concatenated vector $\hat{z}_m^k$ by passing the autofused vector through F_c, another transformation layer. Finally, we optimize the loss between $\hat{z}_m^k$ and z_m^k. (b) GAN-Fusion module for the text modality: Assuming that z_s, z_v, and z_t are the latent speech, video, and text vectors, respectively, we first autofuse z_s and z_v to give z_{tr}. Simultaneously, we pass z_t through the generator G, along with some noise, to get z_g. The generator loss tries to match z_{tr} and z_g and discriminator D tries to distinguish between z_{tr} and z_g, the two sources of input. **Note:** $\bigoplus$ denotes concatenation.

tion is currently unexplored.

3 Proposed methods

This section will discuss the proposed methodologies for effectively fusing inputs from multiple modalities and describe the overall architecture of our models for classification and generation. Most fusion techniques proposed in the literature, such as concatenation, and TFN, involve a deterministic operation for constructing the joint multimodal representation. For instance, in TFN, the 3-fold Cartesian product of unimodal features is used for prediction. The method focuses more on learning rich unimodal features. However, there is no such "learning" procedure for joint representation; they are simply constructed by combining unimodal features in a specific fashion (here, by Cartesian product.) In this paper, we will refer to such techniques as *static* fusion techniques. Since there is no particular learning procedure for the joint representation, it becomes challenging for the final predictor module to model the complex dynamics of multimodal features. In other words, the model is unable to utilize multimodal information effectively.

On the other hand, fusion methods such as LMF and MulT are *adaptive* because they involve a cognitive feature processing step to construct the joint representation. In LMF, it is the decomposition module, and in MulT, it is the final feed-forward fusion mechanism. We refer the reader to Liu et al. (2018) and Tsai et al. (2019) for more detailed explanation of the models.

Our fusion methods involve the concatenation of unimodal embeddings as an initial step. To avoid any conflicts with past works, we will only consider steps *after* concatenation as a part of our fusion method because we do not use the concatenated vector for final prediction; it is only a preliminary step. Therefore, in order to mitigate the "staticness" of existing fusion methods, we propose two adaptive yet simple fusion techniques, *Auto-Fusion* and *GAN-Fusion*. They aim to effectively combine multimodal inputs and mitigate the problem of shallowness and computational overhead exhibited by prior fusion techniques.

3.1 Auto-Fusion

This method encourages the model to extract intermodal features by maximizing the correlation between multimodal inputs. In this method, we first concatenate individual unimodal features and then pass them through a transformation layer to get a

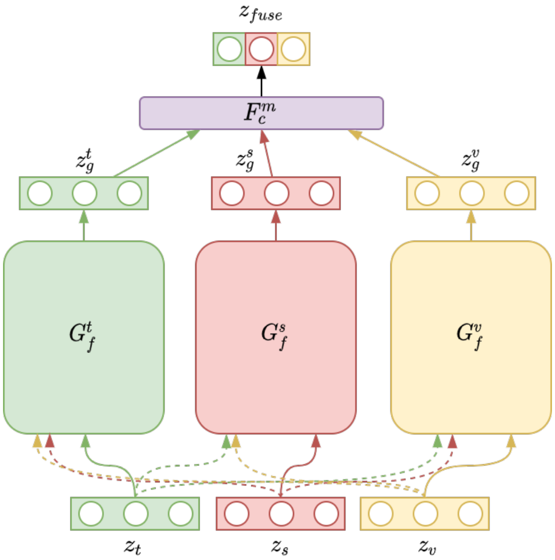


Figure 2: Overall architecture of the GAN-Fusion module. Here, the solid and dashed lines at the bottom part represent input from the target and the complementary modalities, respectively, for G_f^m, the GAN-Fusion module with target modality $m \in \{t, s, v\}$. Furthermore, F_c^m represents the feed-forward layer that produces the fused multimodal representation $\boldsymbol{z_{fuse}}$, which is, in turn, fed to the target decoder for generation networks and to the fully-connected network for classification networks.

autofused latent vector. We use appropriate learners for individual modalities (See Section 4). We then try to reconstruct the originally concatenated vector from the autofused latent vector. Finally, we minimize the Euclidean distance between the original and reconstructed concatenated vector. This process ensures that the learned autofused vector does not contain arbitrary signals from the input concatenated latent vector. Furthermore, training the model for a downstream task such as emotion recognition incentivizes it to "compress" information without losing any essential cues. In other words, it increases the correlation between the autofused and the concatenated latent vector. This generic procedure applies to any scenario where multiple features need to be combined. For example, it can even be used to combine the forward and backward hidden states of LSTMs (Hochreiter and Schmidhuber, 1997), instead of pooling methods such as 1D pooling, max pooling, sum pooling or even simple concatenation.

We now discuss the Auto-Fusion network in detail. We pose fusion of multimodal inputs as a compression problem, where we must retain as much information from the individual modalities as possible. Given n ($\leq$ 3 in our case) d-dimensional multimodal latent vectors, $\boldsymbol{z}_{m_1}^{d_1}, \boldsymbol{z}_{m_2}^{d_2}, \ldots, \boldsymbol{z}_{m_n}^{d_n}$, we first concatenate them to obtain a vector, $\boldsymbol{z}_m^k$, where $k = \sum_i^n d_i$. Then, we apply a transformation $\mathcal{T}$ to $\boldsymbol{z_m^k}$, reducing its number of dimensions to t. Then, we use $\boldsymbol{z_m^t}$ to reconstruct the originally concatenated vector $\boldsymbol{\hat{z}_m^k}$. Finally, we calculate the loss, J_{tr}, between $\boldsymbol{\hat{z}_m^k}$, and $\boldsymbol{z_m^k}$. The simplest version of Auto-Fusion network employs the mean squared error (MSE) loss function, which aligns with our motivation to compress multimodal features: filter out the less useful signals. These steps could be followed in Figure 1(a) and the MSE loss for Auto-Fusion network is given by:

$$J_{tr} = || \, \boldsymbol{\hat{z}_m^k} - \boldsymbol{z_m^k} \, ||^2 \tag{1}$$

For Auto-Fusion, we consider the intermediate vector, $\boldsymbol{z_m^t}$, as the fused multimodal representation.

3.2 GAN-Fusion

In addition to the "staticness" of existing methods, there is also the challenge of distinguishing between ambiguous cases. For instance, the sentence "Your joke blew my mind away, Kevin," could be said in a funny or sarcastic manner. Resolving ambiguity becomes especially important when working on social problems such as hate speech detection. Even when fed with the corresponding speech vector, existing methods cannot effectively distinguish between similar but different emotions such as happiness and calmness. We hypothesize that this is because they do not learn the conditional distribution of sentiment given an utterance (an utterance includes input from all available modalities).

To address this issue, we propose an adversarial training regime that is incentivized to learn the desired conditional distribution. For a task such as emotion recognition, the objective would be sentiment given an utterance. For a more challenging generation task, the model could learn a more complex behaviour, such as the association of different sentences based on how similar they sound and their polarity. Our experiments show that our GAN-based approach is better able to learn such multimodal dynamics compared to other methods.

We now describe GAN-Fusion's architecture in detail for target modality text (denoted by t.) For a given multimodal sample x, we first encode the inputs from each modality (speech, visual and text) to get the respective latent vectors, $\boldsymbol{z_s}, \boldsymbol{z_v}$, and $\boldsymbol{z_t}$.T Choosing a target modality such as text, we pass $\boldsymbol{z_t}$ (along with random normal noise,) through a generator to obtain $\boldsymbol{z_g} = G(\boldsymbol{z_t})$ and autofuse the remaining latent vectors $\boldsymbol{z_s}$ and $\boldsymbol{z_v}$ simultaneously to obtain $\boldsymbol{z_{tr}}$. In the event where we have input from only one modality in addition to text, we do not need Auto-Fusion, and can simply treat the other modality's vector as $\boldsymbol{z_{tr}}$. Finally, we train the network in an adversarial fashion, labelling $\boldsymbol{z_{tr}}$ as positive samples and $\boldsymbol{z_g}$ as negative samples. The adversarial loss, J^t_{adv}, is given below:

$$\min_G \max_D J^t_{adv}(D,G) = \mathbb{E}_{x\sim p_{\boldsymbol{z_{tr}}}(x)}[\log D(x)] + \mathbb{E}_{\boldsymbol{z}\sim p_{z_t}(\boldsymbol{z})}[\log(1 - D(\boldsymbol{z_g}))] \tag{2}$$

Overall, the generator G tries to align features of the target modality with features from the complementary modalities, and the discriminator tries to discern the source of its input. Such a translation between latent vectors has been shown to learn an "intermediate" latent vector denoting their joint representation (Pham et al., 2019; Gao et al., 2019). Learning the latent space in such an adversarial manner induces a clustering effect on the latent space, where texts associated with similar sounds and visuals are grouped together. We conjecture that the adversarial training helps the model learn the relative topology of the complementary modalities' latent space, which improves sampling from the target modality.

We elucidate this effect through the following example. For the sake of simplicity, let us consider only one complimentary modality (video) in this case and let text be the target modality. First, we make a reasonable assumption that video embeddings for, say, Soccer and Golf–falling under the general category of Sports–will be mapped closer to each other and farther from video embeddings from an unrelated topic such as Cooking. When learning the intermediate representation between video and text, the text latent space is constructed such that its relative topology partially reflects the video latent space. So, token embeddings in the text latent space related to videos of similar events (Soccer and Golf) will adopt similar relative positioning as followed by the video embeddings for Soccer and Golf in the video latent space. For multimodal machine translation, if the model is fed with a Golf video and the source text as input, it may be better able to sample jargon words for Golf from the text latent space due to this topology inheritance. This ultimately improves translation-quality. This is also depicted in Figure 4.

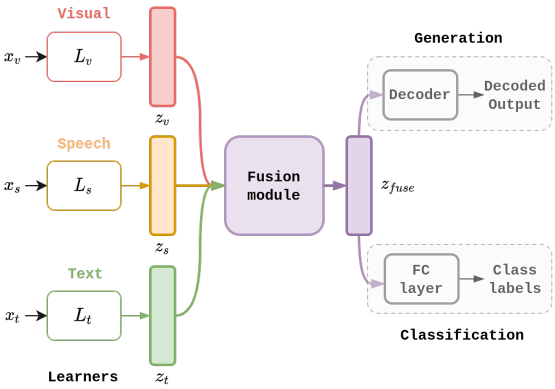


Figure 3: Using proposed fusion techniques for generation/classification. Unimodal inputs x_v, x_s, x_t are passed through their respective learners L_v, L_s, L_t to obtain unimodal representations $\boldsymbol{z_v}, \boldsymbol{z_s}, \boldsymbol{z_t}$. Here, v, s, t correspond to visual, speech, and textual modalities respectively. The individual unimodal representations are then passed through the fusion module (either Auto-Fusion or GAN-Fusion,) which outputs the fused multimodal representation $\boldsymbol{z_{fuse}}$. For generation, $\boldsymbol{z_{fuse}}$ is then passed through a compatible decoder, which generates outputs for the desired target modality. For classification, $\boldsymbol{z_{fuse}}$ is passed through a fully-connected layer instead, which predicts appropriate the class labels.

Figure 1(b) shows GAN-Fusion module for the text modality. The GAN-Fusion module, overall, has one such module for every modality. Total adversarial loss is, therefore, given by:

$$J_{adv} = J^t_{adv} + J^s_{adv} + J^v_{adv} \tag{3}$$

where losses J^s_{adv} and J^v_{adv} for speech and video, respectively, are defined similarly as J^t_{adv}.

Figure 2 shows an overall architecture of the GAN-Fusion module, which consists of G^t_f, G^s_f, and G^v_f, the respective fusion modules for text, speech, and video modalities. We pass the outputs of these modules through a feed-forward layer to obtain the final fused multimodal representation $\boldsymbol{z_{fuse}}$.

3.3 Overall Architecture

In this section, we describe the end-to-end training process for using the proposed fusion methods for 1) Generation tasks (e.g. visual question answering,

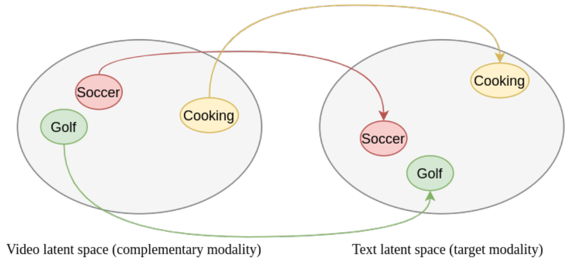


Figure 4: Visualizing the induced clustering effect of GAN-Fusion. The circles represent a cluster of words related to the indicated topic (text inside the circles)

multimodal machine translation) and 2) Classification tasks (e.g. speech emotion recognition, hate speech detection.)

Generation: Figure 3 shows the end-to-end pipeline integrating proposed fusion techniques for generation. We first pass raw inputs from different modalities through their respective learners to obtain their respective latent representations. They are then passed through the fusion module that outputs a fused representation to be used for decoding. We only validate this process for generating text, but this method could very well be used for generating outputs for different target modalities. Notably, our pipeline for generation looks very similar to a Seq2Seq network. We simply introduce a fusion module between the encoder and the decoder module. It should also be noted that all GAN-Fusion modules for all the target modalities (G_f^t, G_f^s, G_f^v) are trained simultaneously with the rest of the network.

Classification: Figure 3 also shows how to adapt the previously described generation network for classification: simply replace the decoder with a fully-connected layer to predict appropriate class labels.

It is important to note that $\boldsymbol{z_{fuse}} = \boldsymbol{z_m^t}$ for Auto-Fusion, and it is obtained as shown in Figure 2 for GAN-Fusion. The overall loss function for our networks can be generalized as follows:

$$J_{total} = \lambda_1 J_{fusion} + \lambda_2 J_{task} \tag{4}$$

Here, J_{fusion} refers to the loss function of the fusion network. It equals J_{tr} (from equation 1) when using Auto-Fusion, and J_{adv} (from equation 3) when using GAN-Fusion. Furthermore, J_{task} refers to the task-specific loss, i.e., classification loss (such as max-margin loss) or generation loss (such as cross-entropy loss for Seq2Seq network). λ_1 and λ_2 are hyperparameters to tune.

4 Experimental Setup

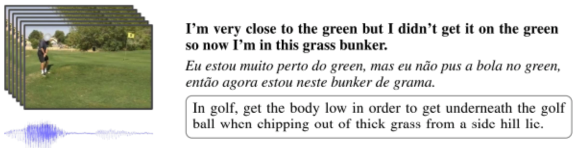
We measure our models' effectiveness on two tasks: 1) multimodal machine translation and 2) multimodal emotion recognition. The subsequent sections describe our complete experimental setup, including datasets and baselines used, implementation details, and evaluation metrics.

4.1 Datasets

We choose three datasets for our experiments, which are described as follows:

IEMOCAP: We use the benchmark Interactive Emotional Dyadic Motion Capture (IEMOCAP) dataset (Busso et al., 2008) for emotion recognition. We only use the textual and speech modalities for our emotion recognition experiments. The dataset is originally split into multiple utterances for each session, and we further split each utterance file based on the provided start and end timestamps to obtain wav files for each sentence. This results in a total of ∼10K audio files, which are then used to extract features for predicting a given utterance's emotion. Concretely, we identify the task as an emotion recognition problem, where, given a sentence and its audio, we aim to infer the correct emotion for that utterance.

How2: We evaluate our models on the multimodal How2 dataset (Sanabria et al., 2018), which comprises of 79,114 instructional videos, their Kaldi (Povey et al., 2011) audio features, and word-level time alignments of English-to-Portuguese translations. The How2 dataset is trimodal in comparison to other multimodal datasets (Matthews et al., 2002; Patterson et al., 2002). This makes it suitable to evaluate the contribution of each modality for different tasks. Further, as a large-scale multilingual dataset, it enables a convenient medium for neural machine translation in our work.



I'm very close to the green but I didn't get it on the green so now I'm in this grass bunker.

Eu estou muito perto do green, mas eu não pus a bola no green, então agora estou neste bunker de grama.

In golf, get the body low in order to get underneath the golf ball when chipping out of thick grass from a side hill lie.

Figure 5: A multimodal sample from the `How2` dataset (Sanabria et al., 2018)

Multi30K: In addition to the How2 dataset, we also run experiments on the bimodal Multi30K dataset (Elliott et al., 2016)–a benchmark dataset for machine translation–extended for German, where each sample has an image, its description in

Model	Source modalities	BLEU 1	BLEU 2	BLEU 3	BLEU 4
Unimodal S2S	t	-	-	-	54.4
Multimodal S2S	s-v-t	-	-	-	54.4
BPE Multimodal	s-v-t	-	-	-	51.0
Unimodal SPM Transformer	t	-	-	-	55.5
Attention over Image Features	s-v-t	-	-	-	56.2
Seq2Seq (w/o attn)	t	48.32	30.63	20.79	14.60
	s	20.11	7.01	3.12	1.57
	v	19.28	6.35	2.33	1.03
Seq2Seq	t	79.21	67.34	52.67	47.34
Auto-Fusion (Ours)	s-t	80.34	67.83	61.27	55.01
	s-v-t	85.23	71.95	69.54	57.80
GAN-Fusion (Ours)	s-t	82.25	69.43	64.33	56.5
	s-v-t	**89.66**	**74.48**	**71.29**	**59.83**

Table 1: Results for machine translation on How2 dataset. ‘t’, ‘s’, ‘v’ represent the text, speech, and video modalities, respectively. Here, ‘attn’ refers to the word-level attention (Luong et al., 2015).

Model	BLEU 4	Meteor
Unimodal Seq2Seq	36.3	56.9
MeMAD submission	44.1	**64.3**
Auto-Fusion (Ours)	42.31	61.7
GAN-Fusion (Ours)	**44.23**	63.8

Table 2: Results for machine translation on Multi30K dataset. **Note:** All methods use ‘v’ and ‘t’ as the source modalities except Unimodal Seq2Seq.

Model	P	R	F	A
LSTM (t)	53.2	40.6	43.4	43.6
LSTM ([s;t])	66.1	65.0	64.7	64.2
MDRE	-	-	-	71.8
MHA-2	-	-	-	76.5
Auto-Fusion (Ours)	75.3	77.4	76.3	77.8
GAN-Fusion (Ours)	**77.3**	**79.1**	**78.2**	**79.2**

Table 3: Precision (P), Recall (R), F1-score (F), and Accuracy (A) for emotion recognition on IEMOCAP.

the source language, and its translated version. We only run experiments on the En-Fr version, however.

4.2 Implementation details

Generation: We train the network shown in Figure 3 with Auto-Fusion and GAN-Fusion as fusion modules. We use an LSTM encoder with 256 hidden units as the learner for textual description in our generation experiments, unless otherwise stated. For the How2 dataset, we use the already provided 2048-dimensional feature vectors for video as raw input for the video learner, and we feed the Kaldi speech vectors to the speech learner, which is a simple feed-forward layer in this case. For the Multi30K dataset, we use a pre-trained VGG (Simonyan and Zisserman, 2015) to encode images, and we do not have a speech learner as there is no speech input in the dataset. The latent dimension is 100 for the fused vector.

Classification: For the task of speech emotion recognition, we train a multimodal classifier on the IEMOCAP dataset. We use LSTMs with 256 hidden units to encode text. For audio, we first pre-process the raw audio files to obtain a lower-dimensional feature vector and then use LSTMs with 50 hidden units as a learner. We predict emotion labels through a full-connected layer as shown in Figure 3 . The latent dimension for the fused vectors is 50 in this case.

When training the GAN-Fusion network, we followed numerous tricks (Goodfellow, 2016) to ensure training stability. A few that helped the most include input normalization, batch normalization, Leaky ReLU activation function (Maas et al., 2013) and Adam optimizer (Kingma and Ba, 2015) for the generator and discriminator networks. The presence of multiple auxiliary losses in our networks also helped.

All the networks in our experiments are implemented in PyTorch (Paszke et al., 2019). To train the different classification networks and generation networks on the Multi30K dataset, we use an Nvidia RTX 2080Ti with 12GB of RAM. However, to train our trimodal networks on the How2 dataset, we use Nvidia P100 with 16 GB RAM to fit video feature vectors in memory.

4.3 Baselines

How2: We use the following baselines for experiments on the How2 dataset:

- **Seq2Seq**: A sequence-to-sequence with attention mechanism (Luong et al., 2015). It employs the previously described learners for each modality, and early fusion.

- **Unimodal and Multimodal S2S:** Unimodal and Multimodal S2S model as described in Sanabria et al. (2018).
- **How2-challenge baselines:** We use Attention over Image Features (Wu et al., 2019), Unimodal SPM Transformer (Raunak et al., 2019) and BPE Multimodal (Lal et al., 2019) from the How2-challenge leaderboard.[2]

Multi30K: We use the following baselines for the Multi30K dataset:

- **Unimodal Seq2Seq:** A text-only NMT system used by Elliott et al. (2017).
- **MeMAD submission** (Grönroos et al., 2018): The best performing model on Multi30K, a multimodal transformer network.

IEMOCAP: We use the following baselines for the IEMOCAP dataset:

- **LSTM (t):** A unimodal LSTM classifier with attention mechanism trained using only text.
- **LSTM ([s;t]):** A bimodal LSTM classifier with attention mechanism on text only. We use the concatenation of speech and text features as the joint multimodal representation.
- **MDRE:** Multimodal Dual Recurrent Encoder proposed by Yoon et al. (2018).
- **MHA-2:** A multimodal classifier with Multi-hop attention mechanism (Yoon et al., 2019).

We compare the above baselines' performance with our two main models: GAN-Fusion and Auto-Fusion, which replace early fusion in the Seq2Seq baseline. We report the results in Table 1, 2, and 3.

4.4 Evaluation metrics

We use Precision, Recall, F-Score, and classification accuracy to evaluate our classification networks trained for speech emotion recognition. For experiments on the How2 and Multi30K dataset, we use BLEU (Papineni et al., 2002) to evaluate the quality of translated sentences. For How2, we compute BLEU1-BLEU4 scores of the different models under consideration. For Multi30K, we also use METEOR (Banerjee and Lavie, 2005), which is a weighted harmonic mean of unigram precision and unigram recall providing a better indication of translation quality. For all the mentioned evaluation metrics, a higher number denotes better performance.

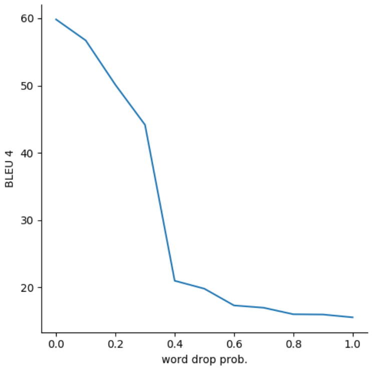


Figure 6: Ablation test on the How2 dataset. Word drop probability v/s BLEU 4. A sudden drop in BLEU scores as we move from 0.3 to 0.4 indicates that our model was able to compensate for $\sim$ 30% of the missing text.

5 Results

Quantitative analysis: Results of our experiments on the How2, Multi30K and IEMOCAP dataset are shown in Tables 1, 2 and 3, respectively. For speech emotion recognition, we observe that our models consistently perform well across all the evaluation metrics. For the relatively challenging multimodal machine translation task, we observe that our model outperforms all existing baselines in terms of BLEU scores. Compared against the best performing baselines, GAN-Fusion improves BLEU4 scores by 3.63 points and 0.13 points on the How2 and Multi30K datasets, respectively. This shows that the fusion module was better able to extract signals from all modalities. On the Multi30K dataset, our model is competitive in terms of METEOR. Such performance becomes more pronounced considering our models have only one attention module, in contrast to multiple self-attention heads in the transformer-based baselines.

The size of our best-performing models (employing GAN-Fusion) is roughly 6M and 11M for classification and generation, respectively, which is significantly lower than traditional transformer models. Fewer trainable parameters reduces the computational cost and the training time. This also indicates potential by-passing of a mechanism like distillation, which is used to reduce parameters in

---

[2] `https://srvk.github.io/how2-challenge/`

transformers (Sanh et al., 2019); however, more thorough experimentation is required to reach a concrete conclusion. Our models can also be used in conjunction with transformers, where we utilize the transformers to learn meaningful unimodal feature vectors initially and then employ the proposed fusion methods.

We also perform a comprehensive set of qualitative experiments on the How2 dataset to understand the capability of our fusion techniques. They are described as follows:

Effect of introducing more modalities: To understand the effect of introducing new modalities separately, we perform experiments with different combinations of source modalities, including individual unimodal baselines (Refer to Table 1.) The results reveal that both auditory and visual modalities *always* contribute towards enhanced translation, but the contribution of visual modality is slightly lower (indicated by the lower increase in BLEU scores in Table 1.) This is also consistent with the findings of Grönroos et al. (2018).

Robustness of multimodal features: It is very important for the learned multimodal latent features to be robust, i.e., they should be able to exploit signals from complementary modalities to compensate for the presence of noise in one modality. Therefore, to gauge the robustness of learned multimodal features, we conduct an ablation test on the How2 dataset. We randomly replace some tokens in the test sentence with an `<UNK>` token and attempt to translate using our best performing model, GAN-Fusion. Figure 6 shows results of our ablation study. We can see that features from the complementary modalities are able to compensate for $\sim 30\%$ of the missing text as we see a sharp drop in BLEU scores beyond that point. This shows that the model does not rely on just the textual description for translation; it also tries to gain a contextual understanding. Hence, it follows that the learned joint representation indeed contains rich information from other modalities.

6 Conclusion and Future Work

In this paper, we propose two adaptive fusion techniques that allow for effective multimodal fusion. Instead of "fixing" a fusion operation a priori, we let the model decide "how" to extract and effectively combine signals from different modalities. Moreover, the joint multimodal representations learned by such models are empirically shown to be robust, which allows the system to maintain good performance even in the absence of some information. Our results indicate that such adaptive models deliver without compromising performance than their more massive counterparts, such as transformers, which is a significant gain.

Our experiments indicate the importance of learning richer unimodal representations. This also suggests that using these methods in conjunction with transformers, which may learn richer unimodal representations, should further improve downstream tasks such as multimodal machine translation and speech emotion recognition. Currently, the attention mechanism is applied only on text. So, another simple way to improve performance would be to introduce visual and acoustic attention mechanisms as well; however, we would still need to address the core problem heterogeneity.

On training GANs: Training GANs is known to be difficult. However, we employed various tricks to ensure the training stability of models, especially the one's employing GAN-Fusion. In fact, a musing line of exploration towards learning a better adaptive model could be to probe the implicit assumptions of GANs themselves. GANs are known to exhibit numerous issues in practice (Arora et al., 2017; Sinn and Rawat, 2018). In Li and Malik (2018), the authors argue the need to return to the principle of maximum likelihood, insisting on full recall.

It is essential to note that much is unknown about these models. More concrete and sound reasoning for the success of these models will rely on two vital components: 1) understanding of the dynamics of the learned latent space, and 2) aligning multimodal features to address heterogeneity. Both these components require more interpretable representations of the otherwise black-box models.

Acknowledgments

We acknowledge Compute Canada's GPU support for our experiments. We also thank Dhruv Kumar for their meaningful suggestions on the manuscript. Finally, we thank the anonymous reviewers for their constructive comments on this work.

References

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. 2017. Generalization and equilibrium in generative adversarial nets (gans). In *ICML*, pages 224–232. PMLR.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, page 335.

Sarath Chandar, Mitesh M Khapra, Hugo Larochelle, and Balaraman Ravindran. 2016. Correlational neural networks. *Neural computation*, 28(2):257–285.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

D. Elliott, S. Frank, K. Sima'an, and L. Specia. 2016. Multi30k: Multilingual english-german image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the Second Shared Task on Multimodal Machine Translation and Multilingual Image Description. In *Proceedings of the Second Conference on Machine Translation*.

Xiang Gao, Sungjin Lee, Yizhe Zhang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2019. Jointly optimizing diversity and relevance in neural response generation. In *NAACL-HLT*.

Ian Goodfellow. 2016. NIPS 2016 Tutorial: Generative adversarial networks.

Stig-Arne Grönroos, Benoit Huet, Mikko Kurimo, Jorma Laaksonen, Bernard Merialdo, Phu Pham, Mats Sjöberg, Umut Sulubacak, Jörg Tiedemann, Raphael Troncy, and Raúl Vázquez. 2018. The MeMAD submission to the WMT18 multimodal translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 603–611.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Lifu Huang, Kyunghyun Cho, Boliang Zhang, Heng Ji, and Kevin Knight. 2018. Multi-lingual common semantic space construction via cluster-consistent word embedding. In *EMNLP*, pages 250–260.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Ke Li and Jitendra Malik. 2018. On the implicit assumptions of gans. *NeurIPS Workshop on Critiquing and Correcting Trends in Machine Learning*.

Paul Pu Liang, Zhun Liu, Yao-Hung Hubert Tsai, Qibin Zhao, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2019. Learning representations from imperfect time series data via tensor rank regularization. In *ACL*, pages 1569–1576.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *ACL*, pages 2247–2256.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*, pages 1412–1421.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Iain Matthews, Timothy F Cootes, J Andrew Bangham, Stephen Cox, and Richard Harvey. 2002. Extraction of visual features for lipreading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):198–213.

Sunil S. Morade and Suprava Patnaik. 2015. Comparison of classifiers for lip reading with cuave and tulips database. *Optik - International Journal for Light and Electron Optics*, 126(24):5753–5761.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*, pages 689–696.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035.

Eric K Patterson, Sabri Gurbuz, Zekeriya Tufekci, and John N Gowdy. 2002. Cuave: A new audio-visual database for multimodal human-computer interface research. In *ICASSP*, volume 2, pages II–2017.

Hai Pham, Paul Pu Liang, Thomas Manzini, Louis-Philippe Morency, and Barnabás Póczos. 2019. Found in translation: Learning robust joint representations by cyclic translations between modalities. In *AAAI*, volume 33, pages 6892–6899.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *IEEE workshop on automatic speech recognition and understanding*.

Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455.

Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. 2018. How2: a large-scale dataset for multimodal language understanding. In *Workshop on Visually Grounded Interaction and Language (ViGIL)*. NeurIPS.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Yuge Shi, N Siddharth, Brooks Paige, and Philip Torr. 2019. Variational mixture-of-experts autoencoders for multi-modal deep generative models. In *NeurIPS*, pages 15718–15729.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Mathieu Sinn and Ambrish Rawat. 2018. Non-parametric estimation of jensen-shannon divergence in generative adversarial network training. In *AISTATS*, pages 642–651.

Nitish Srivastava and Ruslan R Salakhutdinov. 2012. Multimodal learning with deep boltzmann machines. In *NeurIPS*, pages 2222–2230.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NeurIPS*, pages 3104–3112.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *ACL*, volume 2019, page 6558.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Seunghyun Yoon, Seokhyun Byun, Subhadeep Dey, and Kyomin Jung. 2019. Speech emotion recognition using multi-hop attention mechanism. In *ICASSP*, pages 2822–2826.

Seunghyun Yoon, Seokhyun Byun, and Kyomin Jung. 2018. Multimodal speech emotion recognition using audio and text. In *SLT Workshop*, pages 112–118. IEEE.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *EMNLP*, pages 1103–1114.

Detecting Scenes in Fiction: A new Segmentation Task

Albin Zehe[*○] **Leonard Konle**[*○] **Lea Dümpelmann**[•] **Evelyn Gius**[†]
Andreas Hotho[○] **Fotis Jannidis**[○] **Lucas Kaufmann**[○]
Markus Krug[○] **Frank Puppe**[○] **Nils Reiter**[‡]
Annekea Schreiber[†] **Nathalie Wiedmer**[◇]

[*]Equal Contribution [○]University of Würzburg [•]University of Heidelberg
[†]TU Darmstadt [‡]University of Cologne [◇]University of Stuttgart

[○]{zehe, hotho, krug, puppe}@informatik.uni-wuerzburg.de,
{leonard.konle, fotis.jannidis}@uni-wuerzburg.de, lucaskaufmann@gmx.de
[†]evelyn.gius@tu-darmstadt.de, annekealuise.schreiber@stud-mail.tu-darmstadt.de
[•]lea.duempelmann@gmx.de [‡]nils.reiter@uni-koeln.de
[◇]nathalie.wiedmer@ims.uni-stuttgart.de

Abstract

This paper introduces the novel task of scene segmentation on narrative texts and provides an annotated corpus, a discussion of the linguistic and narrative properties of the task and baseline experiments towards automatic solutions. A scene here is a segment of the text where time and discourse time are more or less equal, the narration focuses on one action and location and character constellations stay the same. The corpus we describe consists of German-language dime novels (550 k tokens) that have been annotated in parallel, achieving an inter-annotator agreement of $\gamma = 0.7$. Baseline experiments using BERT achieve an F1 score of 24 %, showing that the task is very challenging. An automatic scene segmentation paves the way towards processing longer narrative texts like tales or novels by breaking them down into smaller, coherent and meaningful parts, which is an important stepping stone towards the reconstruction of plot in Computational Literary Studies but also can serve to improve tasks like coreference resolution.

1 Introduction

Text segmentation is a long standing issue in the area of natural language processing (NLP) encompassing different tasks like segmenting a text into sentences or finding the boundaries between different topics. In this paper, we introduce the task of *scene segmentation*. A scene can be understood as a segment of a text where the story time and the discourse time are more or less equal, the narration focuses on one action and space and character constellations stay the same. Scenes can be found predominately in narrative texts like novels or biographies, which can be understood as a sequence of segments, where some of the segments are scenes and others are not. Scene segmentation is of great interest for the high-level analysis of longer texts, for example the reconstruction of plot, but also for many areas of NLP that deal with longer narrative texts, since even modern methods struggle with processing text longer than a couple of sentences or paragraphs. As an example, the memory requirements of state of the art coreference resolution models scale with $\mathcal{O}(n^4)$ (input length n) (Lee et al., 2017; Joshi et al., 2020) and their performance deteriorates on longer texts. Therefore, it is very helpful to break down the texts into smaller pieces where the character constellation remains the same, enabling us to perform coreference resolution within a scene and then match the characters identified across multiple scenes. Additionally, scene segmentation can also be used to facilitate the summarization of long texts: Since Reiter (2015) has shown that parts of a human written summary correspond well to their similar notion of coherent segments in the original text, it is reasonable to assume that segmenting a text into scenes and then summarizing these scenes is a promising way towards the summarization of long texts. Finally, the number and length of scenes in a text defines a kind of “narrative rhythm”, which we briefly analyze in Section 4.3. This rhythm might serve as a characteristic of an author, or be used as a metric for recommending books to readers.

While the objective of scene segmentation is structurally similar to topic segmentation, there are some important differences: Scenes are defined as narrative units, where each unit has a coherent and

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3167–3177
April 19 - 23, 2021.

stable structure in respect to time, place, character constellation and plot. Narrated time and narrative time in a scene (Scheffel et al., 2019) are more or less equal, which can be seen, for example, in the rendition of verbal communication as direct speech. Scenes in fiction are thus not based only on the topic covered in the narrative. For example, a new scene may cover the same topic as the previous one, but take place in a different location or with a different set of characters. Thus, commonly used segmentation algorithms fail at our task. Even a fine-tuned BERT-based model does not perform well, as we show in Section 5.1. We see this as a sign that scene segmentation requires a large amount of natural language understanding and can, along with other tasks related to an in-depth analysis of narrative structures, serve as a challenge and benchmark for future NLP models.

Our contributions in this paper are as follows: We present and publish a new data set of German fictional texts annotated with scenes and provide an extensive discussion of the guidelines used in the annotation process (which are also included in the release). Additionally, we show that established baselines for text segmentation fail to capture the notion of a narrative scene, necessitating the development of new methods for this task. Our main goal is to introduce the task of scene segmentation and provide resources as well as guidelines to enable research towards this task, which will in turn improve the possibility of processing long, narrative texts in the future.

2 Related Work

Other segmentation tasks have been discussed in NLP for a while, mostly with the goal of identifying regions of news or other non-fictional texts discussing certain topics. The task of topic segmentation is then to identify points in the text in which the topic under discussion changes. Early work to this end uses similarity of adjacent text segments (such as sentences or paragraphs) with a manually designed similarity metric in order to produce the resulting segments. One of the most well known systems of this manner is TextTiling (Hearst, 1997), which was applied to science magazines. Similarity based on common words (Choi, 2000; Beeferman et al., 1999) was superseded with the introduction of Latent Dirichlet Allocation (Blei et al., 2003), which allowed to segment the text into coherent text snippets with similar topic distributions (Riedl and Biemann, 2012; Misra et al., 2011). This procedure was extended by the integration of entity coherence (John et al., 2016) and Wanzare et al. (2019) have used it on (very short) narrative texts in an attempt to extract scripts. Recently, many approaches making use of neural architectures deal with the detection and classification of local coherence (e. g. Li and Jurafsky, 2016; Pichotta and Mooney, 2016; Li and Hovy, 2014), which is an important step for a text summarization of high quality (Xu et al., 2019). Text segmentation using neural architectures was conducted on Chinese texts and it was shown that recurrent neural networks are able to predict the coherence of subsequent paragraphs with an accuracy of more than 80 % (Pang et al., 2019). Lukasik et al. (2020) compare three BERT based architectures for segmentation tasks: Cross-Segment BERT following the NSP Pretraining-Task and fine-tuned on segmentation, a Bi-LSTM on top of BERT to keep track of larger context and an adaption of a Hierarchical BERT network (Zhang et al., 2019).

Some work has been done on segmenting narrative texts, but aiming at identifying topical segments – which, as we have pointed out above, is different from scene segmentation. With a set of handcrafted features, Kauchak and Chen (2005) achieve a WindowDiff score (Pevzner and Hearst, 2002) of about 0.5, evaluated on two novels. Kazantseva and Szpakowicz (2014) have annotated the novel *Moonstone* with topical segments, and presented a model to create a hierarchy of topic segments. They report about 0.3 WindowDiff score. Most closely related to our task are the papers by Reiter (2015), which documents a number of annotation experiments, and Kozima and Furugori (1994), which presents lexical cohesiveness based on the semantic network Paradigme (Kozima and Furugori, 1993) as an indicator for scene boundaries and evaluates their approach qualitatively on a single novel. However, neither of them provide annotation guidelines, annotated data or a formal definition of the task.

A related area of research is discourse segmentation, where the goal is also to find segments that are not necessarily defined by topic, and are also assigned labels in addition to the segmentation. There are annotated news corpora in this area featuring fine-grained discourse relations between relatively small text spans (Carlson et al., 2002; Prasad et al., 2008). Although larger structures have been discussed in literature (Grosz and Sidner, 1986), no annotated corpora have been released.

3 Task: Scene Segmentation

In this section we will first present a description of the task placing it in narratological tradition, and then we will describe it more formally as an atypical instance of a segmentation task.

3.1 Scenes in Narrative Text

In narratology, the analysis of narrative texts usually distinguishes between *discours* and *histoire* (Genette, 1983). *Discours* stands for the text or the representation of the narrative, while *histoire* concerns the narrated world, including characters and plot. In an ideal-typical view, it is assumed that plot is composed of several transformation processes from the smallest, spatiotemporal units – the so-called events.

Operationalizing plot is a challenging problem, because it involves natural language understanding, inferencing and interpretation on a high level (Meister, 2003). Nevertheless, different approaches are conceivable and have been discussed in the literature. Jockers (2015) approximates plot as an emotional arc of a narrative by assigning sentiment scores to each sentence and applying a Fourier transformation to derive an overall arc. Another way of modeling plot is to detect individual events in a text and then combining those to larger units and finally to a representation of the plot. There have been advances on the detection of events (Sprugnoli and Tonelli, 2019; Sims et al., 2019; Aldawsari and Finlayson, 2019) in texts. However, the definition of an event is unclear, with large possible differences in the level of granularity, making it an unstable starting point for analyzing plot.

Our approach to action in narratives is grounded in narratology, but by focussing on scenes it tackles the phenomenon on a less granular level than events. In narratology, the notion of scene has been introduced by Genette (1983) as a concept concerning the so-called pace of narration, i. e., the relation between the amount of time that passes in the narrative (story time, or *histoire*) and the amount of time covered by its narration (narrated time, or *discours*). Genette defines a scene as follows: "scene, most often in dialogue, which, as we have already observed, realizes conventionally the equality of time between narrative and story" (Genette, 1983, p. 94). It is important to note that the equality is put as "only a kind of conventional equality between narrative time and story time" (Genette, 1983, p. 87). Defining scenes based only on one feature, time, is useful in the context of Genette's theory, but it lacks descriptive power when the concept is supposed to be used to analyze plot, because the concept of plot is always implying aspects like character (and character constellation) and event sequences. In addition, Genette's definition of scene leads to two notions – story and narrative time – that are not easier to operationalize. Therefore, we adopt a more general understanding of scenes that includes characters, space and action. This is closer to our everyday understanding of scenes and similar to the understanding of scenes in plays as "a division [. . .] during which the action takes place in a single place without a break in time" or "a part of a play, movie, story, etc., in which a particular action or activity occurs" (Learner's Dictionary).

In order to capture this fuller notion we follow Gius et al. (2019) in defining scenes: A scene is a segment of the *discours* (presentation) of a narrative which presents a part of the *histoire* (connected events in the narrated world) such that (1) time is equal in *discours* and *histoire*, (2) place stays the same, (3) it centers around a particular action, and (4) the character constellation is equal. All of these conditions are not absolute, there can be small changes in either component, as detailed below.

In media like film or plays usually one scene follows another. Non-scenes, in which the progress of time is narrated in a compressed way can be found, but are relatively rare. In narrative texts passages which are not scenes can be found more often between scenes. The boundary between scenes can be clear cut, often indicated by phrases like 'at the next morning', 'in the meantime' etc, but can also be vague, for example when reflections of the narrator or a character are bridging two scenes or when the narrated time is accelerated at the end of one scene and then slowed again for the next.

3.2 Formal Task Definition

After defining scenes from a narrative perspective, we can formalize the task of *scene segmentation*: We are given a narrative text (e. g., a novel) as input and derive a segmentation that additionally labels each segment of the text either as a scene or as a non-scene. This is a notable difference to other segmentation tasks with no further distinction between types of segments.

There are multiple possible operationalizations of this task. In this paper, we frame scene segmentation as a sentence-level classification task where we find borders between segments and additionally classify the borders, representing whether there is a scene before and after them. More specifically, a border can fall into one of the three classes `SCENE-SCENE`, `SCENE-NONSCENE`, `NONSCENE-SCENE`.[1]

Other operationalizations include, for example, only providing a simple segmentation in a first step and then additionally classifying each segment as either scene or non-scene using a text classification model or directly using a sequence labeling model to assign each token or sentence an IOB tag. While these are also valid approaches to scene segmentation, we focus on the first method here, since it provides the easiest end-to-end operationalization: We can train one model that simultaneously detects borders and classifies them into one of the border types. One possible drawback of this method is that a model might predict incompatible scene borders (e.g., SCENE-NONSCENE followed by SCENE-SCENE). This problem can be alleviated by the use of a CRF-based classifier, where such a sequence would be recognized as very unlikely/impossible.

4 Corpus

4.1 Annotation

We annotated 15 dime novels from diverse genres (love, horror, adventure, etc.) in German language with a total of 36 k sentences and 550 k tokens.[2] We decided to use dime novels because preliminary studies have shown that the task is quite challenging even in this literary medium, which is more accessible to human readers than highbrow literature. Moreover, the length of an individual dime novel (Ø 36 k tokens) allows to annotate a reasonable number of full novels with reasonable effort.

The annotation of the corpus was performed by two annotators, an additional curator established the gold standard.[3] Annotators were also asked to document the reason for each scene change. The guidelines are the result of two iterations, incorporating feedback from and discussion with the annotators.

Overview In the following, we provide an overview of the central aspects of our annotation guidelines.[4] The guidelines are based on four main components: *time*, *space*, *action* and *characters*. In short, a change in any of these components (e.g., a large jump in time) is a signal for a scene change. The following paragraphs describe our guidelines in more detail, specifically which of these signals are most important for determining whether there is a scene change at a given position. We also detail how we deal with *contradictory signals and corner cases*. We conclude this section with a discussion of certain *typographical markers* found in our texts and the *difficulties* encountered in the annotation process.

Time With regard to time, the default for a scene is a chronological narration with a uniform pace whereas scene boundaries are indicated by changes in chronology (i. e. anachronies like flashbacks or flash-forwards), temporal omission (i. e., ellipsis) or major changes of the narrative pace. For all candidates for scene changes, the impact of the temporal phenomenon in question needs to be weighted with regard to its context within the narration. For example, we generally assume that the greater a time leap is in relation to the general granularity of time in the narration, the more likely the scene is changing. Therefore, if the general narration speed is rather low and action is for example narrated more or less on an hourly base, a leap of one day probably indicates a boundary between two scenes. On the contrary, if action is narrated on a day-to-day-base, a leap of one day is probably not a scene boundary but rather part of an ongoing scene.

Space With regard to space, the default for a scene is to take place within the same space whereas a change in space indicates a scene boundary. Space, similar to time, is analyzed with regard to the granularity of space within the narrative. The general principle adopted for the detection of relevant space changes is a container principle, i. e. space can be composed from smaller units (spaces, rooms). For example, the following passage is considered to take place within the same space, since the rooms in question (the corridor and the breakfast room) are parts of a hotel, i. e. the same space container:

> Auf dem Weg zum Frühstückszimmer meinte mein Partner: “Ich habe mir die halbe Nacht den Kopf

---

[1] We do not segment non-scenes further.

[2] We use SpaCy (Honnibal et al., 2020) for tokenization and sentence-splitting.

[3] The annotators are students with backgrounds in computer science, digital humanities and/or German studies with prior experience in annotating the same or similar tasks.

[4] The full guidelines are available in Gius et al. (2021).

> darüber zerbrochen, wie wir unserem geheimnisvollen Gegner die Maske herunterreißen könnten." "Und?", fragte ich. "Ich war nicht gerade mit Geistesblitzen gesegnet", sagte Suko. Wir betraten das Frühstückszimmer.
>
> (*Der Turm der 1000 Schrecken*)[5]

Action A container principle has also been adopted for the analysis of action. Within a scene, action is assumed to be coherent and continuous. Generally, according to the container principle, we have to decide whether actions can be counted as belonging to the previous one. We also introduced a test based on the more intuitive understanding of scenes discussed above, where annotators imagined the passage in question as a movie and asked themselves whether it could be transposed to one movie scene. The boundaries of the scene are the points where a fade out (or fade in) could be inserted.

Characters The fourth aspect of scene, characters, again is supposed to stay stable within a scene whereas a change in character constellation can indicate a change of scene. Here it is important to examine both the role of the character that joins or leaves and the course of action. The more important the character is and/or the more the narrative focuses on a different action after the change in constellation, the more likely we have a scene boundary.

Contradictory Signals and Corner Cases Since these aspects are often not consistent with each other, we weight them according to their observed relevance for a scene change. Most relevant is a change in the event sequence, followed by character constellation, time and finally space.

In addition to the four relevant aspects in narrations for scenes, we included procedures for unclear cases in our guidelines. The most frequent ones are short passages of reflection, as for example inner monologues, that can be found directly before and after passages clearly qualifying as a scene. If they are shorter than the scene itself, they are also considered part of the scene.

Typographical Markers With regard to typical typographical markers of scenes in some texts (e.g., ***), we changed our handling during the annotation.[6] We started with the stars included, but then decided to erase them in order to rely on content-related aspects only due to their lack of generalizability: While *** is used somewhat consistently in German dime novels, this is not the case for other types of narrative texts, e.g. novels in book form. Since we want to keep our task, dataset and, in the future, solutions as general as possible, we did not rely on these markers. Changing our handling of these markers did not have an influence on our inter-annotator agreement. Despite this, it is an open question whether to include such typographical markers in future work. These markers, as well as chapters and paragraphs,[7] are standardized ways of signaling the segmentation of narratives. Therefore, the start of a scene directly after a marker may differ systematically from a start in running text.

Annotation Difficulties Most persisting annotation difficulties are caused by the fact that most criteria in the guideline have a relative nature. It is not trivial to decide whether a change in time, space or character configuration is determining the beginning (or end) of a scene, since its importance depends on the granularity of time, space or character configuration in the specific narrative. These granularities cannot be specified on a general level. The second issue is the lack of an operationalization of action applicable for the analysis of literary narratives.

Nevertheless, with the described approach to scene annotation we seem to tackle most of the issues the annotators and the curators came up with and improved the agreement between the annotators considerably. Overall, the combination of operationalized narratological categories with the more intuitive test of the transposability to a movie scene seems to be a fruitful approach.

To illustrate the annotation task for an international audience, we annotated a copyright-free novel and machine translated it using DeepL.[8]

4.2 Measuring Inter-Annotator Agreement

Several measures to evaluate segmentation systems have been proposed in the past. According to a sur-

---

[5] Our translation: On the way to the breakfast room, my partner said, "I spent half the night worrying about how we could rip the mask off our mysterious opponent." "And?", I asked. "I wasn't exactly blessed with flashes of inspiration", Suko said. We entered the breakfast room.

[6] The markers are removed from text in the release, but there is information on their presence in metadata.

[7] Only 43% of all scene boundaries align with paragraphs.

[8] `https://professor-x.de/gsd/viewer.html`

vey by Kazantseva and Szpakowicz (2012) these are also widely used to quantify inter-annotator agreement. When choosing a measure for the evaluation of scene segmentation, the following characteristics must be considered: (1) Near misses are not critical, since scene boundaries tend to be fuzzy. (2) High variance of segment length in documents and between documents. (3) The existence of non-scenes leads to two classes of segments.

A major objective in the research on the evaluation of segmentation is to overcome influences from the field of classification, which model segmentation as token or character classification, and to replace these with measures that calculate a penalty dependent on the distance between true and predicted boundaries. After an evaluation of existing metrics (Pk: Beeferman et al. 1997, WindowDiff: Pevzner and Hearst 2002, Segmentation Similarity: Fournier 2013), we opted for using γ (gamma) (Mathet et al., 2015) to measure inter-annotator agreement (and prediction performance, see below).

Values of γ range from $-\infty$ to 1 theoretically, empirically often from 0 to 1; with 1 meaning there are no disagreements. The basic idea of gamma is to combine aligning and comparing the annotations into a single metric. Once an alignment between the annotations is established, near misses and category disagreements can be measured in a straightforward way, configurable by the user. Because there is no way to calculate this alignment a priori, gamma selects the alignment that leads to the least overall disagreement, which is then considered the observed disagreement γ_o. Expected disagreement γ_e is calculated by sampling from the existing annotations such that random annotations can be compared. The final gamma score is calculated as $\gamma = 1 - \frac{\gamma_o}{\gamma_e}$, as for other metrics based on disagreements. Because the measured disagreement is dependent on both boundary positions and segment categories, segmentation tasks producing gaps or tasks that include unitizing and categorization (like named entity detection or topic segmentation) are supported by γ.

On all novels in our corpus, annotators reach an agreement of $\gamma = 0.7$ (with a standard deviation of $\sigma = 0.07$). The developers of γ do not provide an explicit interpretation scale. However, since its value lies in the interval of $[-\infty; 1]$, with 0 representing agreement purely by chance and γ is a disagreement metric similar to Krippendorff's α, a similar scale can be applied to it. Thus, the reported agreement of 0.7 is acceptable, given the fact that the task is new and very complex. Figure 1 shows aligned scene annotations of two annotators together with γ scores.

Category	Portion
Scenes starting with direct speech	14 %
Scenes ending with direct speech	13 %
Sentences containing direct speech	55 %

Table 1: Information on direct speech in the corpus

4.3 Corpus Analysis

This section gives an overview of various quantitative analyses that we have conducted on the annotated corpus.

First, we find that scenes are much more common in the texts than non-scenes, as the corpus contains 971 segments marked as scenes and 34 marked as non-scenes. This results in 937 SCENE-SCENE, 30 SCENE-NONSCENE and 23 NONSCENE-SCENE boundaries with respect to our task definition (see section 3.2). Figure 2 shows the lengths of scenes and non-scenes in comparison. As can be seen clearly, scenes are typically much longer than non-scenes, although with a quite high spread.

As direct speech plays an important role in narrative texts in general, and has already been established as a core ingredient of scenes, we use the supervised STRW Recognizer (Brunner et al., 2020) to detect the distribution of direct speech in the texts, and their relation to the scene boundaries. As Table 1 shows, direct speech does not serve well as a marker to detect scene boundaries.

Another interesting aspect of the scenes is their distribution along the text flow. Figure 3 shows for each sentence of a novel how many scenes haven been found up to this point. While it can be directly observed that average scene length seems to be a discriminatory property of the stories (*Fürstenkinder*, *Jason Dark* and *Sophienlust* contain nearly the same amount of sentences but differ highly in scene count), there is no pattern along the x-axis discernible. Denser clusters may appear at any point within the text. Nevertheless, we can highlight writing style differences between the stories. Figure 3 shows that some stories (*2012*, *Tausend Pferde*) start with a rapid succession of scenes, while others begin with longer sequences

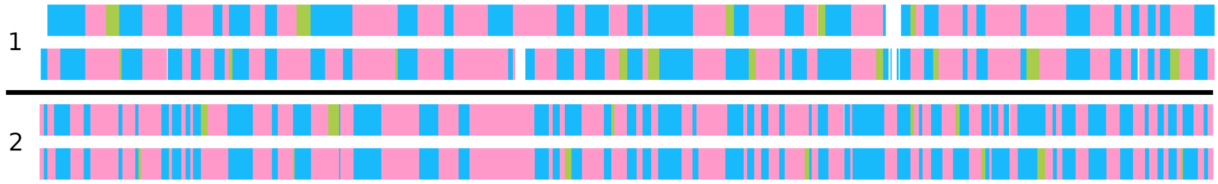


Figure 1: Scene annotations of *Dr. Bergen*, (1, $\gamma = 0.57$, worst agreement in the dataset) and *Tausend Pferde* (2, $\gamma = 0.83$, best agreement in the dataset). Three colors (red, blue and green) are used for scenes to make it easier to identify differences and matches. White gaps indicate non-scenes.

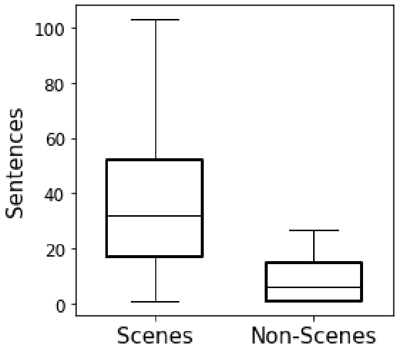


Figure 2: Length of scenes and non-scenes

(*Fürstenkinder*, *Verschmäht*).

As we have summarized above, time, place and characters are important constituents of scenes. We therefore analyzed in which form they are expressed shortly after a new scene has begun. Figure 4 shows the distribution of the occurrence of locational and temporal adverbials for each text in first sentences in scenes and all other sentences, as well as their coincidence with annotated reasons for a scene change. We used a list of adverbials according to (Eisenberg, 2006) as well as a self-made list. We found that there is high variation with respect to the scene-starting sentences. Temporal adverbials occur more frequently in the first sentence of a scene, but the difference for locational adverbials is much smaller. It becomes clear that the information about space and time changes with adverbials was often not perceived by the annotators as decisive for scene changes.

Additionally, we also analyzed the distribution of explicit character references (i.e., referencing them by their name rather than, e.g., pronouns). To this end, we used the method proposed by Jannidis et al. (2017) to extract all such references and plotted their position in a scene (Figure 5). We find that, as we would expect, the first 5 % of sentences in a scene contain more explicit character references on average than all other segments.

4.4 Corpus Release

The corpus is available on our website.[9] Since the texts are copyrighted, we cannot publish them directly. Instead, we provide the EAN of the epubs and a script that merges text and standoff annotations.

5 Baseline Experiments

5.1 Setup

Similar to measuring inter-annotator agreement, it is not trivial to define a metric for evaluating a task like scene segmentation. For the evaluation here, we provide two **metrics**: (1) precision, recall, F1-score are measured as a sentence-wise classification task, with different granularities as described below (two classes vs. four classes). (2) In addition, we employ the observed part of gamma γ_o as a prediction performance metric. This metric is calculated from the alignment between gold and system output with the least disagreement. The reason for also reporting this metric is that it is in line with the annotation experiments and captures the task more directly. After all, the annotators are asked to create units in the context of the entire discourse, and not to classify individual sentences.

In order to assess the difficulty of scene segmentation, we evaluate multiple baselines on our proposed dataset: Two unsupervised standard segmentation techniques (TextTiling and TopicTiling) and two additional supervised baselines based on BERT. Note that the unsupervised techniques cannot perform the full scene task defined in Section 3.2, but only the first part, that its, detecting scene borders.

Unsupervised Baselines We use the TextTiling (Hearst, 1997) implementation from nltk (Bird et al., 2009) and evaluate these hyper-parameters: $w, k \in \{5, 10, ..., 45\}$, smoothing_width, smoothing_rounds $\in \{1, ..., 4\}$. Reported results are for

---

[9] `https://professor-x.de/german-scene-dataset`

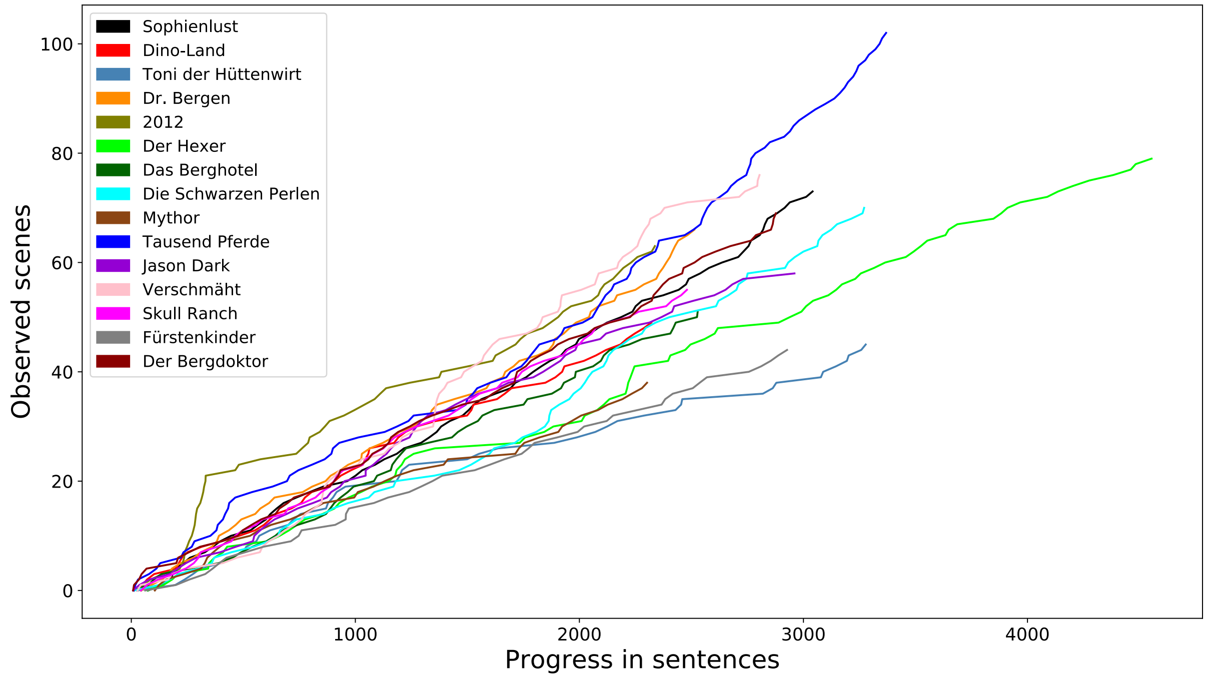


Figure 3: Relationship between the amount of sentences passed (x axis) and the number of scenes (y axis) for every novel. Higher slope indicates shorter scenes. An interactive version of this figure, suited for color blind people can be found online.[a]

[a] `https://professor-x.de/gsd/sentences-per-scene.html`

the best configuration.[10] For TopicTiling (Riedl and Biemann, 2012), we train an LDA (Blei et al., 2003) model on a corpus of 870 dime novels[11] (appr. 2 million words)[12] and evaluate on our scene segmentation dataset using the recommended parameters.[13] As seen in Table 2, both unsupervised baselines perform very poorly at the task, reaching an F1-score of 4 % and 5 %, respectively.

BERT Baseline Since the standard unsupervised methods do not perform well for our task, we build a simple supervised baseline. To this end, we fine-tune a pre-trained BERT model[14] to binary scene segmentation in the following way: We construct a training sample as a triple of *(sentence, context, label)*, where *sentence* is a target sentence from a text, *context* concatenates the two previous and following sentences with the target sentence and *label* is `BORDER`, if there is a scene border before the target sentence and `NOBORDER` otherwise. In order to capture the distinction between scenes and non-scenes, we also evaluate a more fine-grained 4-label classification task with the `BORDER` label split into its three possible subclasses (`SCENE-SCENE`, `SCENE-NONSCENE`, `NONSCENE-SCENE`), as defined in Section 3. We fine-tune BERT using FARM[15] for a default duration of ten epochs in a leave-one-out style, training on all texts except one and evaluating on the remaining. Table 2 shows that, while performing much better than the unsupervised baselines from the previous paragraph, BERT is not capable of providing a satisfactory segmentation.

5.2 Discussion

Our results show that standard unsupervised methods for text segmentation are not applicable to the task of scene segmentation. This is unsurprising, as our definition of scenes is not based on topical coherence, but on other aspects of the text, which are not considered by TextTiling and TopicTiling.

Additionally, our supervised baseline suggests that a BERT model is capable of picking up some signals for scene changes, but applying a standard

---

[10] We used the best possible configuration on the entire data set to provide an upper bound for their performance.

[11] Detailed list of novels: `https://professor-x.de/german-scene-dataset/list`

[12] Using the implementation from `http://gibbslda.sourceforge.net/`.

[13] `https://github.com/riedlma/topictiling`

[14] `https://deepset.ai/german-bert`.

[15] `https://github.com/deepset-ai/FARM`.

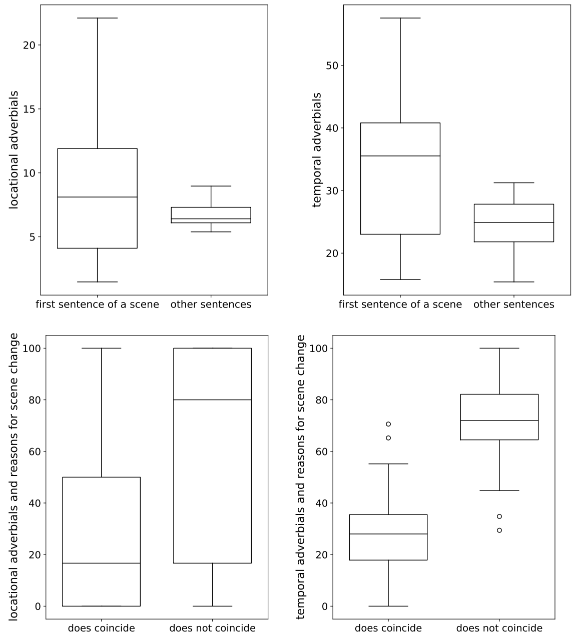


Figure 4: Distribution of locational (left) and temporal (right) adverbials at the beginning of a scene, as well as their coincidence with annotated reasons for a scene change.

model is not sufficient without additional modifications, such as specialized model architectures. On average, the system divides the novels into four times more scenes than the annotation specifies. A qualitative evaluation of the predicted borders shows that it is almost always possible to find one of the reasons defined in the guidelines for a scene change. Thus, the reason for the hypersensitivity of the baseline does not seem to lie in the inability to recognize markers, but rather in their contextualization. Typical errors are caused by (a) mentions of characters and places without actually appearing/becoming the place of action, (b) metaphors (e.g. "Maybe they only needed a narrow bridge to get back together?"), (c) indirect or reported speech, (d) different forms of referencing characters and (e) moving through places in a scene (e.g., "The visitor nodded and followed her into the kitchen.")

The BERT model is also not capable of finding any borders from non-scenes to scenes, leading to a score of 0 in this setting. This, in combination with the task's relevance for the analysis of long texts, motivates further research. A possible direction for future work would be using the information from our analysis, like the presence of temporal markers and character references.

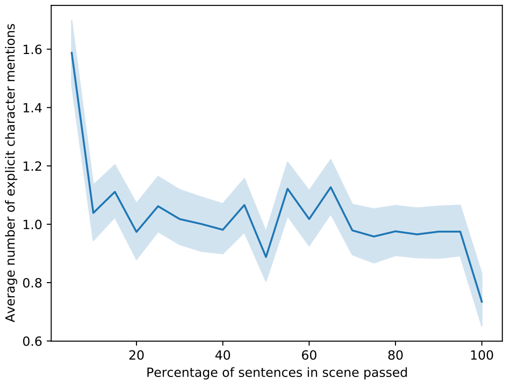


Figure 5: Distribution of explicit character references over positions in a scene.

Model (class)	Prec.	Rec.	F1	γ_o
TextTiling	0.02	0.97	0.04	0.01
TopicTiling	0.03	0.22	0.05	0.02
BERT (binary)	0.49	0.15	0.24	0.15
BERT (S-S)	0.43	0.13	0.2	0.15
BERT (S-NS)	0.85	0.24	0.38	
BERT (NS-S)	0.0	0.0	0.0	

Table 2: Results from all baselines. BERT (binary) denotes the performance scores for the `BORDER` class. S-S denotes borders between two scenes, NS-S and S-NS denote borders between scenes and non-scenes.

6 Conclusion

This paper describes the task of detecting scenes in narrative texts and introduces a corpus annotated for this task. The corpus consists of a number of German dime novels annotated according to guidelines describing the specifics of the task in detail. The inter-annotator agreement indicates that the task is challenging, but feasible for humans. As the analysis of the corpus shows, the information about character constellation, time, space and action is informative, but only an integral understanding of the text makes it possible to fully solve the task. Thus, apart from the many applications of a scene segmentation itself, it also provides an interesting challenge for natural language processing: Due to the high level of natural language understanding required, it will likely necessitate the development of novel approaches to be solved satisfactorily.

References

Mohammed Aldawsari and Mark Finlayson. 2019. Detecting subevents using discourse and narrative features. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4780–4790, Florence, Italy. Association for Computational Linguistics.

Doug Beeferman, Adam Berger, and John Lafferty. 1997. Text segmentation using exponential models. *arXiv preprint cmp-lg/9706016*.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine learning*, 34(1-3):177–210.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Annelen Brunner, Ngoc Duyen Tanja Tu, Lukas Weimer, and Fotis Jannidis. 2020. To BERT or not to BERT - comparing contextual embeddings in a deep learning architecture for the automatic recognition of four types of speech, thought and writing representation. In *Proceedings of the 5th Swiss Text Analytics Conference and the 16th Conference on Natural Language Processing, SwissText/KONVENS 2020, Zurich, Switzerland, June 23-25, 2020 [online only]*, volume 2624 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2002. Rst discourse treebank, ldc2002t07. Technical report, Philadelphia: Linguistic Data Consortium.

Freddy YY Choi. 2000. Advances in domain independent linear text segmentation. *arXiv preprint cs/0003083*.

Peter Eisenberg. 2006. *Der Satz*, volume 2 of *Grundriss der deutschen Grammatik*. J. B. Metzler, Stuttgart, Weimar.

Chris Fournier. 2013. Evaluating text segmentation using boundary edit distance. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1702–1712. Association for Computational Linguistics.

Gérard Genette. 1983. *Narrative discourse: An essay in method.* Cornell University Press.

Evelyn Gius, Fotis Jannidis, Markus Krug, Albin Zehe, Andreas Hotho, Frank Puppe, Jonathan Krebs, Nils Reiter, Nathalie Wiedmer, and Leonard Konle. 2019. Detection of scenes in fiction. In *Proceedings of Digital Humanities 2019*.

Evelyn Gius, Carla Sökefeld, Lea Dümpelmann, Lucas Kaufmann, Annekea Schreiber, Svenja Guhr, Nathalie Wiedmer, and Fotis Jannidis. 2021. Guidelines for detection of scenes. `https://doi.org/10.5281/zenodo.4457177`.

Barbara J. Grosz and Candace L. Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Marti A Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64.

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

Fotis Jannidis, Isabella Reger, Lukas Weimer, Markus Krug, Martin Toepfer, and Frank Puppe. 2017. Automatische Erkennung von Figuren in deutschsprachigen Romanen. *Proceedings of DhD*.

Matthew L. Jockers. 2015. Revealing Sentiment and Plot Arcs with the Syuzhet Package.

Adebayo Kolawole John, Luigi Di Caro, and Guido Boella. 2016. Text segmentation with topic modeling and entity coherence. In *International Conference on Hybrid Intelligent Systems*, pages 175–185. Springer.

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

David Kauchak and Francine Chen. 2005. Feature-based segmentation of narrative documents. In *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing*, pages 32–39, Ann Arbor, Michigan. Association for Computational Linguistics.

Anna Kazantseva and Stan Szpakowicz. 2012. Topical segmentation: a study of human performance and a new measure of quality. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–220. Association for Computational Linguistics.

Anna Kazantseva and Stan Szpakowicz. 2014. Hierarchical topical segmentation with affinity propagation. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 37–47, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Hideki Kozima and Teiji Furugori. 1993. Similarity between words computed by spreading activation on an english dictionary. *In Proceedings of the European Association for Computational Linguistics*.

Hideki Kozima and Teiji Furugori. 1994. Segmenting narrative text into coherent scenes. *Literary and Linguistic Computing*, 9(1):13–19.

Merriam-Webster Learner's Dictionary. `https://learnersdictionary.com/definition/scene`. Accessed: 2021-01-20.

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

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2039–2048.

Jiwei Li and Dan Jurafsky. 2016. Neural net models for open-domain discourse coherence. *arXiv preprint arXiv:1606.01545*.

Michal Lukasik, Boris Dadachev, Gonçalo Simões, and Kishore Papineni. 2020. Text segmentation by cross segment attention.

Yann Mathet, Antoine Widlöcher, and Jean-Philippe Métivier. 2015. The unified and holistic method gamma (γ) for inter-annotator agreement measure and alignment. *Computational Linguistics*, 41(3):437–479.

Jan Christoph Meister. 2003. *Computing action a narratological approach*. Walter De Gruyter, Berlin; New York.

Hemant Misra, François Yvon, Olivier Cappé, and Joemon Jose. 2011. Text segmentation: A topic modeling perspective. *Information Processing & Management*, 47(4):528–544.

Yihe Pang, Jie Liu, Jianshe Zhou, and Kai Zhang. 2019. Paragraph coherence detection model based on recurrent neural networks. In *International Conference on Swarm Intelligence*, pages 122–131. Springer.

Lev Pevzner and Marti A. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Comput. Linguist.*, 28(1):19–36.

Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Rashmi Prasad, Alan Lee, Nikhil Dinesh, Eleni Miltsakaki, Geraud Campion, Aravind Joshi, and Bonnie Webber. 2008. Penn Discourse Treebank Version 2.0 LDC2008T05. Web download, Linguistic Data Consortium, Philadelphia.

Nils Reiter. 2015. Towards Annotating Narrative Segments. In *Proceedings of the 9th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, pages 34–38, Beijing, China. Association for Computational Linguistics.

Martin Riedl and Chris Biemann. 2012. Topictiling: a text segmentation algorithm based on lda. In *Proceedings of ACL 2012 Student Research Workshop*, pages 37–42. Association for Computational Linguistics.

Michael Scheffel, Antonius Weixler, and Lukas Werner. 2019. Time. In Peter et al. Hühn, editor, *The living handbook of narratology*. Hamburg.

Matthew Sims, Jong Ho Park, and David Bamman. 2019. Literary Event Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3623–3634, Florence, Italy. Association for Computational Linguistics.

Rachele Sprugnoli and Sara Tonelli. 2019. Novel Event Detection and Classification for Historical Texts. *Computational Linguistics*, 45(2):229–265.

Lilian Diana Awuor Wanzare, Michael Roth, and Manfred Pinkal. 2019. Detecting everyday scenarios in narrative texts. In *Proceedings of the Second Workshop on Storytelling*, pages 90–106, Florence, Italy. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Discourse-aware neural extractive model for text summarization. *arXiv preprint arXiv:1910.14142*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

`LESA`: Linguistic Encapsulation and Semantic Amalgamation Based Generalised Claim Detection from Online Content

Shreya Gupta[†*], **Parantak Singh**[‡*], **Megha Sundriyal**[†],
Md Shad Akhtar[†], **Tanmoy Chakraborty**[†]

[†] *IIIT-Delhi, India.* [‡] *Birla Institute of Technology and Science, Pilani, Goa, India.*
`{shreyag, meghas, shad.akhtar, tanmoy}@iiitd.ac.in,`
`f20170109@goa.bits-pilani.ac.in`

Abstract

The conceptualization of a claim lies at the core of argument mining. The segregation of claims is complex, owing to the divergence in textual syntax and context across different distributions. Another pressing issue is the unavailability of *labeled* unstructured text for experimentation. In this paper, we propose `LESA`, a framework which aims at advancing headfirst into expunging the former issue by assembling a source-independent generalized model that captures syntactic features through part-of-speech and dependency embeddings, as well as contextual features through a fine-tuned language model. We resolve the latter issue by annotating a Twitter dataset which aims at providing a testing ground on a large unstructured dataset. Experimental results show that `LESA` improves upon the state-of-the-art performance across six benchmark claim datasets by an average of 3 claim-F1 points for in-domain experiments and by 2 claim-F1 points for general-domain experiments. On our dataset too, `LESA` outperforms existing baselines by 1 claim-F1 point on the in-domain experiments and 2 claim-F1 points on the general-domain experiments. We also release comprehensive data annotation guidelines compiled during the annotation phase (which was missing in the current literature).

1 Introduction

The concept of a claim lies at the core of the argument mining task. Toulmin (2003), in his argumentation theory, described the term 'claim' as '*an assertion that deserves our attention*'; albeit not very precise, it still serves as an initial insight. In recent years, Govier (2013) described a 'claim' as '*a disputed statement that we try to support with reasons.*'

Text	Claim?
Alcohol cures corona.	Yes
Wearing mask can prevent corona.	Yes
Lord, please protect my family & the Philippines from the corona virus.	No
If this corona scare doesn't end soon imma have to intervene	No

Table 1: A few examples of claim and non-claim.

The predicament behind the claim detection task exists given the disparity in conceptualization and lack of a proper definition of a claim. The task of claim detection across different domains has garnered tremendous attention so far owing to an uprise in social media consumption and by extension the existence of fake news, online debates, widely-read blogs, etc. As an elementary example, claim detection can be used as a precursor to fact-checking; wherein segregation of claims aids in restricting the corpus that needs a fact-check. A few examples are shown in Table 1.

Most of the existing works are built upon two fundamental pillars - semantic encapsulation (Daxenberger et al., 2017; Chakrabarty et al., 2019) and syntactic encapsulation (Levy et al., 2014; Lippi and Torroni, 2015). They mainly focus on adapting to texts from similar distributions or topics or both. Secondly, they often exercise against well-structured and laboriously pre-processed formal texts owing to the lack of a labeled corpus consisting of unstructured texts. As a result, claim detection from unstructured raw data still lies under a relatively less explored umbrella.

Motivation: Claims can be sourced from a variety of sources, e.g., online social media texts, microblogs, Wikipedia articles, etc. It is, however, crucial to pay special attention to claims observed

---

[*] First two authors have equal contributions. The work was done when Parantak was an intern at LCS2 Lab, IIIT-Delhi.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3178–3188
April 19 - 23, 2021.

on online social media (OSM) sites (Matthew A. Baum; who). Twitter, being a major OSM platform, provides the perfect playground for different ideologies and perspectives. Over time, Twitter has emerged as the hub for short, unstructured pieces of text that describe anything from news to personal life. Most individuals view and believe things that align with their compass and prior knowledge, *aka conformity bias* (Whalen and Laland, 2015) – users tend to make bold claims that usually create a clash between users of varied opinions. At times, these claims incite a negative impact on individuals and society. As an example, a tweet that reads "*alcohol cures corona*" can lead to massive retweeting and consequential unrest, especially in times of a pandemic, when people are more vulnerable to suggestions. In such cases, automated promotion of claims for immediate further checks could prove to be of utmost importance. An automated system is pivotal since OSM data is far too voluminous to allow for manual human checks, even if it was an expert.

At the same time deploying separate systems contingent on the source of a text is inefficient and moves away from the goal of attaining human intelligence in natural language processing tasks. An ideal situation would be a framework that can effectively detect claims in the general setting. However, a major bottleneck towards this goal is the unavailability of an annotated dataset from noisy platforms like Twitter. We acknowledge this bottleneck and, in addition to proposing a generalised framework, we develop a qualitative annotated resource and guidelines for claim detection in tweets.

Proposed Method: There exists several claim detection models; however, the downside is that most of them are trained on structured text from a specific domain. Therefore, in this work, we propose **`LESA`**, a **L**inguistic **E**ncapsulation and **S**emantic **A**malgamation based generalized claim detection model that is capable of accounting for different text distributions, simultaneously. To formalize this, we divide the text, contingent upon their structure, into three broad categories – noisy text (*tweets*), semi-noisy text (*comments*), and non-noisy text (*news, essays*, etc.). We model each category separately in a joint framework and fuse them together using attention layers.

Since the task of claim detection has a strong association with the structure of the input, as argued by Lippi and Torroni (2015), we leverage two linguistic properties – part-of-speech (POS) and dependency tree, to capture the linguistic variations of each category. Subsequently, we amalgamate these features with BERT (Devlin et al., 2019) for classification.

We evaluate `LESA` on seven different datasets (including our Twitter dataset) and observe efficient performance in each case. Moreover, we compare `LESA`'s performance against various state-of-the-art systems for all seven datasets in the general and individual settings. The comparative study advocates the superior performance of `LESA`.

Summary of the Contributions: We summarize our major contributions below:

- **Twitter claim detection dataset and comprehensive annotation guidelines.** To mitigate the unavailability of an annotated dataset for claim detection in Twitter, we develop a large COVID-19 Twitter dataset, the first of its kind, with $\sim 10,000$ labeled tweets, following a comprehensive set of claim annotation guidelines.

- **`LESA`, a *generalized* claim detection system.** We propose a generalized claim detection model, `LESA`, that identifies the presence of claims in *any* online text, without prior knowledge of the source and independent of the domain. To the best of our knowledge, this is the first attempt to define a model that handles claim detection from both structured and unstructured data in conjunction.

- **Exhaustive evaluation and superior results.** We evaluate `LESA` against multiple state-of-the-art models on six benchmark claim detection datasets and our own Twitter dataset. Comparison suggests `LESA`'s superior performance across datasets and the significance of each model component.

Reproducibility: Code and dataset is publicly available at `https://github.com/LCS2-IIITD/LESA-EACL-2021`. Appendix comprises of detailed dataset description, annotation guidelines, hyperparameters, and additional results.

2 Related Work

In the past decade, the task of claim detection has become a popular research area in text processing with an initial pioneering attempt by Rosenthal and McKeown (2012). They worked on mining claims from discussion forums and employed a supervised approach with features based on sentiment and word-grams. Levy et al. (2014) proposed a context dependent claim detection (CDCD) approach. They described CDC as '*a general, concise statement that directly supports or contests the given topic.*' Their approach was evaluated over Wikipedia articles; it detected sentences that include CDCs using context-based and context-free features. This was followed by ranking and detecting CDCs using logistic regression. Lippi and Torroni (2015) proposed context-independent claim detection (CICD) using linguistic reasoning, and encapsulated structural information to detect claims. They used constituency parsed trees to extract structural information and predicted parts of the sentence holding a claim using SVM. Although their approach achieved promising results, they also used a Wikipedia dataset which was highly engineered and domain dependent.

Daxenberger et al. (2017) used six disparate datasets and contrasted the performance of several supervised models. They performed two sets of experiments – in-domain CD (trained and tested on the same dataset) and cross-domain CD (trained on one and tested on another unseen dataset). They learned divergent conceptualisations of claims over cross-domain datasets. Levy et al. (2017) proposed the first unsupervised approach for claim detection. They hypothesised a "claim sentence query" as an ordered triplet: $\langle$that $\rightarrow$ MC $\rightarrow$ CL$\rangle$. According to the authors, a claim begins with the word 'that' and is followed by the main concept (MC) or topic name which is further followed by words from a pre-defined claim lexicon (CL). This approach would not fit well for text stemming from social media platforms owing to a lack of structure and the use of 'that' as an offset for claim.

In recent years transformer-based language models have been employed for claim detection. Chakrabarty et al. (2019) used over 5 million self-labeled Reddit comments to fine-tune their model. However, they made no attempt to explicitly encapsulate the structure of a sentence. Recently, the CLEF-2020 shared task (Barrón-Cedeño et al., 2020) attracted multiple models which are tweaked specifically for claim detection. Williams et al. (2020) bagged the first position in the task using a fine-tuned RoBERTa (Liu et al., 2019) model with mean pooling and dropout. First runner up of the challenge, Nikolov et al. (2020) used logistic regression on various meta-data tweet features and a RoBERTa-based prediction. Cheema et al. (2020), the second runner up, incorporated pre-trained BERT embeddings along with POS and dependency tags as features trained using SVM.

Traditional approaches focused primarily on the syntactic representations of claims and textual feature generation, while recent neural methods leverage transformer models. With `LESA`, we attempt to learn from the past while building for the future – we propose encapsulating syntactic representations in the form of POS tags and dependency sequences along with the semantics of the input text using transformer-based BERT (Devlin et al., 2019). Another key observation has been the use of highly structured and domain-engineered datasets for training the existing models in claim detection. In the current age of alarming disinformation, we recognise the augmented need for a system that can detect claims in online text independent of its origin, context or domain. Therefore, in addition to considering texts from different online mediums, we incorporate, for the first time, a self-annotated large Twitter dataset to the relatively structured datasets that exist in this field.

3 Proposed Methodology

Traditionally, the narrative on claim detection is built around either syntactic (Levy et al., 2017; Lippi and Torroni, 2015) or semantic (Daxenberger et al., 2017; Chakrabarty et al., 2019) properties of the text. However, given our purview on the integration of both, we propose a combined model, `LESA` , that incorporates exclusively linguistic features leveraged from part-of-speech (POS) tags and dependency tree (DEP) as well as semantic features leveraged from transformer-based model, BERT (Devlin et al., 2019).

By the virtue of digital media, we generally deal with texts from three kind of environments: (a) a controlled platform where content is pre-reviewed (e.g., news, essays, etc.); (b) a free platform where authors have the freedom to express themselves without any restrictions on the length (e.g., online comments, Wikipedia talk pages); and (c) a free platform with restrictions on the text length (e.g., tweets). The texts in the first category is usually free of any grammatical and typographical mistakes, and thus belong to the non-noisy category. On the other hand, in the third case, texts exhibit a significant amount of noise, in terms of spelling variations, hashtags, emojis, emoticons, abbrevia-

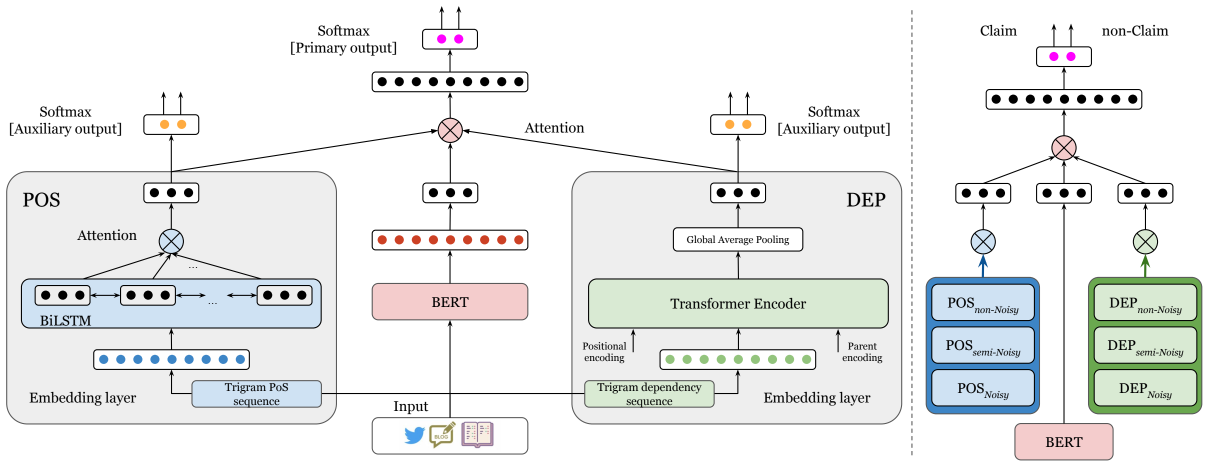


Figure 1: Schematic diagram of our proposed `LESA` model. The structure on the right is a high level schematic diagram. Structure on the left shows POS and DEP for one viewpoint.

tions, etc., to express the desired information within the permissible limit, thus it belongs to the noisy class. The second case is a mixture of the two extreme cases and hence, constitutes the semi-noisy category. We employ three pre-trained models representing noisy, semi-noisy, and non-noisy data for both POS and dependency-based features. The intuition is to leverage the structure-specific linguistic features in a joint framework.

Domain adaptation from a structured environment to an unstructured one is non-trivial and requires specific processing. Therefore, to ensure generalization, we choose to process each input text from three different viewpoints (structure-based segregation), and intelligently select the contributing features among them through an attention mechanism. We use it to extract the POS and DEP-based linguistic features. Subsequently, we fuse the linguistic and semantic features using another attention layer before feeding it to a multilayer perceptron (MLP) based classifier. The idea is to amalgamate diverse set of features from different perspectives and leverage them for the final classification. A high-level architectural diagram is depicted in Figure 1. We design parallel pillars for each viewpoint (right side of Figure 1) such that the noisy pillar contains pre-trained information from the noisy source and so on. When the common data is passed through the three pillars we hypothesize each pillar's contribution dependending on the type of input data. For example, if the data source is from a noisy platform, we hypothesize that the noisy pillars will have more significance than the other two viewpoints. We demonstrate this effect in Table 5.

A. Part-of-speech (POS) Module

The POS module consists of an embedding layer followed by a BiLSTM and an attention layer to extract the syntactic formation of the input text. We pre-train the POS module for each viewpoint, and later fine-tune them while training the integrated model.

At first, each sequence of tokens $\{x_1, x_2, ..., x_n\}$ is converted to a sequence of corresponding POS tags resulting into the set $\{p_1, p_2, \dots, p_n\}$. However, the foremost limitation of this modeling strategy is the limited and small vocabulary size of 19 owing to a specific number of POS tags. To tackle this, we resort to using k-grams of the sequence. The sequence of POS tags (with $k = 3$) now becomes $\{(p_0, p_1, p_2), (p_1, p_2, p_3), (p_2, p_3, p_4), ..., (p_{n-2}, p_{n-1}, p_n), (p_{n-1}, p_n, p_{n+1})\}$, where p_0 and p_{n+1} are dummy tags. Subsequently, a skip-gram model (Mikolov et al., 2013) is trained on the POS-transformed corpus of each dataset, which thereby translates to a POS embedding, E_P.

B. Dependency Tree (DEP) Module

Dependency parsing is the function of abstracting the grammatical assembly of a sequence of tokens $\{x_1, x_2, \dots, x_n\}$ such that there exists a directed relation (dependency), $d(x_i, x_j)$, between any two tokens x_i and x_j, where x_i is the headword and x_j is modified by the headword. We obtain these dependency relations through spaCy[1] which uses the clearNLP guidelines. Initially, each sequence is rendered into a combination of the *dependency-tag* arrangement $\{d_1, d_2, \dots d_n\}$ and a *parent-position*

---

[1] `www.spacy.io`

arrangement $\{pp_1, pp_2, \ldots pp_n\}$. Here, each d_j represents a dependency tag, where x_j is modified by x_i, and pp_j is the index of the modifier (headword) x_i.

We then leverage the transformer encoder (Vaswani et al., 2017), where traditionally, a position-based signal is added to each token's embedding to help encode the placement of tokens. In our modified version, the token sequence is the *dependency-tag* sequence $d_e = \{d_1, d_2, ..., d_n\}$, wherein a *parent-position* based signal is additionally added to encode the position of the modifier words.

$$d'_e = d_e + [(E_{p_1}, E_{pp_1}), ..., (E_{p_n}, E_{pp_n})] \quad (1)$$

where $d'_e \in \mathbf{R}^{d \times n}$ is the modified dependency embedding of a sequence of length n, E_{p_i} and E_{pp_i} are the encodings for the *token-position* and the *parent-position* (position of token's modifier), and $(,)$ represent tuple brackets.

This helps us create a flat representation for a dependency graph. The transformer-architecture that we employ comprises of 5 attention heads with an embedding size of 20. Given that there are only a handful of dependency relations, this, still poses the problem of a limited vocabulary size of 37. Having accounted for the parent positions already, we decide to again employ tri-gram sequences $\{(d_0, d_1, d_2), (d_1, d_2, d_3), (d_2, d_3, d_4), ...,$ $(d_{n-2}, d_{n-1}, d_n), (d_{n-1}, d_n, d_{n+1})\}$ in place of uni-grams.

4 Datasets

A number of datasets exist for the task of claim detection in online text (Peldszus and Stede, 2015; Stab and Gurevych, 2017); however, most of them are formal and structured texts. As we discussed earlier, OSM platforms are overwhelmed with various claim-ridden posts. Despite the abundance of tweets, literature does not suggest any significant effort for detecting claims in Twitter; Arguably, the prime reason is the lack of a large-scale dataset. Recently, a workshop on claim detection and verification in Twitter was organized under CLEF-2020 (Barrón-Cedeño et al., 2020). It had two subtasks related to claim identification with separate datasets. The first dataset consists of $1,060$ COVID-19 tweets for claim detection; whereas, the second one comprises of another $1,000$ tweets for claim retrieval. In total, there were $2,069$ annotated tweets of which $1,704$ had claims and 365 were non-claims. Another recent in-progress dataset on claim detection, which currently has only 305 claim and 199 non-claim tweets, was released by Alam et al. (2020).

	Our Annotation	
CLEF-2020	Non-claim	Claim
Non-claim	301	47
Claim	64	550

Table 2: Confusion matrix highlighting the differences and similarities between Alam et al. (2020) and our annotation guidelines for CLEF-2020 claim dataset.

Unfortunately, the aforementioned limited instances are insufficient to develop an efficient model. Therefore, we attempt to develop a new and relatively larger dataset for claim detection in OSM platforms. We collected $\sim 40,000$ tweets from various sources (Carlson, 2020; Smith, 2020; Celin, 2020; Chen et al., 2020; Qazi et al., 2020) and manually annotated them. We additionally included claim detection datasets of Alam et al. (2020) and CLEF-2020 (Barrón-Cedeño et al., 2020) and re-annotated them in accordance with our guidelines. During the cleaning process, we filtered a majority of tweets due to their irrelevancy and duplicacy. To ensure removal of duplicates, we performed manual checking and exhaustive preprocessing.

Data Annotation: To annotate the tweets, we extend and adapt the claim annotation guidelines of Alam et al. (2020). The authors targeted and annotated only a subset of claims, i.e., factually verifiable claims. They did not consider personal opinions, sarcastic comments, implicit claims, or claims existing in a sub-sentence or sub-clause level. Subsequently, we propose our definition of claims and extrapolate the existing guidelines to be more inclusive, nuanced and applicable to a diverse set of claims. Our official definition for claims, adopted from Oxford dictionary[2], is to *state or assert that something is the case, with or without providing evidence or proof.*

We present the details of annotation guidelines in Gupta et al. (2021). Following the guidelines, we annotated the collected tweets, and to ensure coherence and conformity, we re-annotated the tweets of Alam et al. (2020) and CLEF-2020 (Barrón-Cedeño et al., 2020). It is intriguing to see the differences and similarities of the two guidelines; therefore, we compile a confusion matrix for CLEF-

---

[2] `https://www.lexico.com/definition/claim`

Dataset		**Text**
Noisy	TWR	@realDonaldTrump Does ingesting bleach and shining a bright light in the rectal area really cure #COVID19? Have you tried it? Is that what killed Kim Jong Un? #TrumpIsALaughingStock #TrumpIsALoser
Semi-noisy	OC	*smacks blonde wig on Axel* I think as far as DiZ is concerned, he is very smart but also in certain areas very dumb - - witness the fact that he didn't notice his apprentices were going to turn on him, when some of them (cough Vexen cough) aren't exactly subtle by nature.
	WTP	Not to mention one without any anonymous users TALKING IN CAPITAL LETTERS !!!!!!!!
Non-noisy	MT	Tax data that are not made available for free should not be acquired by the state.
	PE	I believe that education is the single most important factor in the development of a country.
	VG	When's the last time you slipped on the concept of truth?
	WD	The public schools are a bad place to send a kid for a good education anymore.

Table 3: One example from each dataset. Underlined text highlights noisy and semi-noisy phrases.

Dataset		**Noisy**	**Semi-noisy**		**Non-noisy**			
		TWR	**OC**	**WTP**	**MT**	**PE**	**VG**	**WD**
Tr	Cl	7354	623	1030	100	1885	495	190
	N-cl	1055	7387	7174	301	4499	2012	3332
Ts	Cl	1296	64	105	12	223	57	14
	N-cl	189	730	759	36	509	221	221
Tot	Cl	8650	687	1,135	112	2,108	552	204
	N-cl	1244	8117	7933	337	5008	2233	3553

Table 4: Statistics of the datasets (Abbreviations: Cl: Claim, N-Cl: Non-claim, Tr: Train set, Ts: Test set; Tot: Total).

2020 claim dataset, as presented in Table 2. Each tweet in our corpus of $9,894$ tweets has been annotated by at least two annotators, with an average Cohen's kappa inter-annotator agreement (Cohen, 1960) score of 0.62. In case of a disagreement, the third annotator was considered and a majority vote was used for the final label. All annotators were linguists.

Other Datasets: Since we attempt to create a generalized model that is able to detect the presence of a claim in any online text, we accumulate, in addition to the Twitter dataset, six publicly available benchmark datasets: (i) Online Comments (OC) containing Blog threads of LiveJournal (Biran and Rambow, 2011), (ii) Wiki Talk Pages (WTP) (Biran and Rambow, 2011), (iii) German Microtext (MT) (Peldszus and Stede, 2015), (iv) Persuasive Student Essay (PE) (Stab and Gurevych, 2017), (v) Various Genres (VG) containing newspaper editorials, parliamentary records and judicial summaries, and (vi) Web Discourse (WD) containing blog posts or user comments (Habernal and Gurevych, 2015). All datasets utilised in this paper contain English texts only. For German Microtexts (MT), we used the publicly available English translated version published by MT's original authors (Peldszus and Stede, 2015)). The same was utilized by Chakrabarty et al. (2019).

The datasets are formed by considering text at the sentence level. For example, in Persuasive Essays (PE) dataset, each essay is broken into sentences and each sentence is individually annotated for a claim. Considering the structure of the input texts in these datasets, we group them into three categories as follows: Noisy (Twitter), Semi-noisy (OC, WTP), Non-noisy (MT, PE, VG, WD). We list one example from each dataset in Table 3. We also highlight the noisy and semi-noisy phrases in Twitter, and OC and WTP datasets respectively. Moreover, we present detailed statistics of all seven datasets in Table 4.

5 Experimental Setup

For all datasets besides twitter, we use the train, validation, and test splits as provided by UKP Lab[3]. A mutually exhaustive 70:15:15 split was maintained for Twitter dataset. We compute POS embeddings by learning word2vec skip-gram model (Mikolov et al., 2013) on the tri-gram[4] POS sequence. For the skip-gram model, we set *context window* $= 6$, *embedding dimension* $= 20$, and discard the POS sequence with *frequency* ≤ 2. Subsequently, we compute dependency embeddings with *dimension* $= 20$ using Transformer (Vaswani et al., 2017) encoder with 5 *attention heads*. Please note that the choice of using Bi-LSTM, as oppose to Transformers, for extracting the POS features is empirical[5].

The outputs of the POS and dependency embedding layers are subsequently fed to a BiLSTM and *GlobalAveragePooling* layers, respectively. Their respective outputs are projected to a 32-dimensional representation for the fusion. We employ HuggingFace's BERT implementation for computing the tweet representation. The

---

[3] `https://tinyurl.com/yyckv29p`

[4] Choice of $n = 3$ is empirical. We report supporting experimental results in Gupta et al. (2021).

[5] Gupta et al. (2021) accompanies the supporting results.

Models	**Noisy**		**Semi-Noisy**				**Non-Noisy**								**Wt Avg**	
	Twitter		OC		WTP		MT		PE		VG		WD			
	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$		
BERT	0.60	0.83	0.52	**0.24**	0.53	**0.32**	0.70	0.63	0.69	0.64	0.58	**0.43**	0.48	0.22	0.58	0.73
BERT + POS	0.61	0.84	**0.53**	**0.24**	0.54	0.31	0.75	0.69	0.72	0.64	0.59	**0.43**	0.51	0.24	0.60	0.74
BERT + Dependency	0.59	0.82	0.51	0.23	0.52	0.30	**0.79**	**0.73**	0.69	0.62	0.56	0.41	0.48	0.22	0.57	0.72
POS + Dependency	0.45	0.70	0.48	0.19	0.47	0.25	0.57	0.46	0.50	0.45	0.56	0.41	0.44	0.17	0.48	0.61
`LESA` (Combined-view)	0.61	0.85	0.51	0.23	0.53	0.31	0.77	0.71	0.71	0.64	0.57	0.40	0.48	0.22	0.59	**0.75**
`LESA`(768*dim*)	0.58	0.80	0.52	**0.24**	**0.57**	0.29	**0.77**	**0.71**	0.73	0.65	0.60	**0.43**	**0.52**	**0.25**	0.59	0.71
`LESA`(32*dim*)	**0.62**	**0.85**	**0.53**	**0.24**	**0.55**	**0.32**	0.77	0.69	**0.74**	**0.66**	**0.68**	0.41	**0.52**	**0.25**	**0.61**	**0.75**

Table 5: Macro F1 (m-$F1$) and Claim-F1 (c-$F1$) for ablation studies.

768-dimensional embeddings is projected to a 32-dimensional representation using linear layers. We progress with the 32-dimensional representation of BERT as we observe no elevation in results on using the 768-dimensional representation, as can be seen in Table 5. Besides, the latter results in ~ 2 million trainable parameters, whereas the former requires ~ 1.2 million trainable parameters. We employ sparse categorical cross-entropy loss with Adam optimizer and use softmax for the final classification.[6] For evaluation, we adopt **macro-F1** (**m-$F1$**) and **claim-F1** (**c-$F1$**) scores used by the existing methods (Daxenberger et al., 2017; Chakrabarty et al., 2019).

We perform our experiments in two setups. In the **first in-domain setup**, we train, validate and test on the same dataset and repeat it for all seven datasets independently. In the **second general-domain setup**, we combine all datasets and train a unified generic model. Subsequently, we evaluate the trained model on all seven datasets individually.

Furthermore, for each experiment, we ensure a balanced training set by down-sampling the dominant class at $1:1$ ratio. However, we use the original test set for a fair comparison against the existing baselines and state-of-the-art models.

6 Experimental Results

Table 5 shows m-$F1$ and c-$F1$ for different variants of `LESA`. We begin with a fine-tuned BERT model and observe the performance on test sets of all seven datasets. On the Twitter dataset, the BERT architecture yields m-$F1$ score of 0.60 and c-$F1$ score of 0.83. We also report the weighted-average score as 0.58 m-$F1$ and 0.73 c-$F1$, in the last two columns of Table 5. Since we hypothesize that claim detection has a strong association with the structure of the text, we amalgamate POS and dependency (DEP) information with the BERT architecture in a step-wise manner. The BERT+POS model reports an increase of 1% m-$F1$ and c-$F1$ scores on the Twitter dataset. We observe similar trends in other datasets and the overall weighted-average score as well. We also perform experiments on other permutations, and their results are listed in Table 5. Finally, we combine both POS and DEP modules with the BERT architecture (*aka.* `LESA`). It obtains improved results for most of the cases, as shown in the last row of Table 5. The best result on average stands at 0.61 m-$F1$ and 0.75 c-$F1$ for the proposed `LESA` model. This serves as a testament to our hypothesis, validating our assumption that combining syntactic and semantic representations leads to better detection of claims.

In all aforementioned experiments, we use our pre-defined concept of three viewpoints, i.e., noisy, semi-noisy and non-noisy. Therefore, for completeness, we also construct a combined viewpoint which does not contain any structure-specific pillar in POS or DEP branches. The results from this ablation experiment are reported in `LESA` (Combined-view) row. We observe that the combined-view results are inferior to the variant with separate viewpoints for each component (c.f. second last and last row of Table 5 respectively). Thus, providing attention to datasets based on the noise in their content is demonstrated by a significant increase of $\sim 2\%$ m-$F1$ from combined viewpoint to separate viewpoints experiment.

A. Baselines and Comparative Analysis

We employ the following baselines (some of them being state-of-the-art systems for claim detection and text classification): ▷ **XLNet** (Yang et al., 2019): It is similar to the BERT model, where we fine-tune XLNet for the claim detection; ▷ **Accenture** (Williams et al., 2020): A RoBERTa-based system that ranked first in the CLEF-2020 claim detection task (Barrón-Cedeño et al., 2020); ▷ **Team Alex** (Nikolov et al., 2020): The second-ranked system at CLEF-2020 task that fused tweet meta-data into RoBERTa for the final prediction; ▷ **Check-Square** (Cheema et al., 2020): An SVM-based system designed on top of pre-trained BERT em-

---

[6] Gupta et al. (2021) accompanies other hyperparameters.

Models	**Noisy**		**Semi-Noisy**				**Non-Noisy**								**Wt Avg**	
	Twitter		OC		WTP		MT		PE		VG		WD			
	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$		
BERT	0.50	0.67	0.50	0.24	0.36	0.27	0.75	0.69	0.73	**0.67**	0.61	0.48	0.48	0.23	0.52	0.62
XLNet	0.52	0.70	0.45	0.24	0.55	0.30	0.49	0.43	0.71	0.64	0.53	0.43	0.51	0.12	0.54	0.64
Accenture	0.48	0.15	0.44	0.16	0.50	0.23	0.48	0.28	0.45	0.11	0.39	0.27	0.34	0.11	0.46	0.15
Team Alex	**0.70**	0.88	0.46	0.23	0.52	0.21	0.75	0.64	0.69	0.64	0.32	0.38	0.60	0.34	0.59	0.76
Check Square	0.12	0.02	0.49	0.25	0.39	0.26	0.57	0.32	0.47	0.11	0.32	0.37	**0.76**	**0.56**	0.35	0.07
CrossDomain	0.67	0.84	**0.61***	**0.26***	**0.59***	0.29*	0.79*	0.67*	**0.74***	0.61*	0.66*	0.45*	0.63*	0.29*	**0.65**	0.74
CrossDomain†	0.67	0.84	0.50	0.24	0.52	0.27	0.85	0.79	0.71	0.63	0.60	0.46	0.59	0.31	0.61	0.74
LESA	0.67	**0.89**	0.51	**0.26**	0.57	**0.33**	**0.80**	**0.71**	0.73	**0.67**	**0.68**	**0.52**	0.61	0.35	0.63	**0.79**

Table 6: Macro F1 (m-$F1$) and F1 for claims (c-$F1$) in the in-domain setup. For CrossDomian, the asterisk (*) indicates results taken from Daxenberger et al. (2017) and the dagger (†) represents the *reproduced* results.

Model	**Noisy**		**Semi-Noisy**				**Non-Noisy**								**Wt Avg**	
	Twitter		OC		WTP		MT		PE		VG		WD			
	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$		
BERT	0.60	0.83	0.52	0.24	0.53	**0.32**	0.70	0.63	0.69	0.64	0.58	0.43	0.48	0.22	0.58	0.73
XLNet	0.59	0.81	0.56	**0.28**	**0.57**	0.29	0.68	**0.69**	0.71	0.64	0.61	**0.44**	**0.52**	**0.25**	0.59	0.72
Accenture	0.49	0.43	0.31	0.12	0.40	0.18	0.36	0.13	0.51	0.36	0.38	0.17	0.37	0.04	0.43	0.38
Team Alex	0.54	0.75	0.54	0.25	0.54	0.30	0.71	0.65	0.71	0.63	0.61	0.43	0.48	0.19	0.57	0.67
Check Square	0.58	0.82	0.51	0.23	0.48	0.28	0.56	0.53	0.68	0.59	0.56	0.38	0.47	0.21	0.56	0.72
CrossDomain	**0.65**	0.82	**0.57**	0.27	0.53	0.28	0.71	0.63	0.66	0.57	0.61	0.43	**0.52**	**0.25**	0.60	0.71
LESA	0.62	**0.85**	0.53	0.24	0.55	**0.32**	**0.77**	**0.69**	**0.74**	**0.66**	**0.68**	0.41	**0.52**	**0.25**	**0.61**	**0.75**

Table 7: Macro F1 (m-$F1$) and Claim-F1 (c-$F1$) in the general-domain setup.

Models	**Noisy**		**Semi-Noisy**		**Non-Noisy**	
	m-$F1$	c-$F1$	m-$F1$	c-$F1$	m-$F1$	c-$F1$
BERT	0.60	0.83	0.52	**0.29**	0.63	0.58
XLNet	0.59	0.81	**0.57**	**0.29**	0.65	0.59
Accenture	0.49	0.43	0.36	0.16	0.45	0.30
Team Alex	0.54	0.75	0.54	0.28	0.65	0.57
CheckSquare	0.58	0.82	0.49	0.26	0.61	0.53
CrossDomain	**0.65**	0.82	0.55	0.28	0.63	0.53
LESA	0.62	**0.85**	0.54	**0.29**	**0.69**	**0.60**

Table 8: Category-wise weighted-average F1 scores.

beddings in addition to incorporating POS and dependency tags as external features. ▷ **CrossDomain** (Daxenberger et al., 2017): Among several variations reported in the paper, their best model incorporates CNN (random initialization) for the detection. We reproduce the top submissions from CLEF-2020 challenge using the best performing models mentioned in the referenced papers. Code for CheckSquare was provided online. For Accenture and Team Alex we reproduce their methods using the hyper-parameters mentioned in the paper. We evaluate all baselines using the same train and test set as for LESA .

We report our comparative analysis for the in-domain setup in Table 6. We observe that LESA obtains best c-$F1$ scores for six out of seven datasets. Additionally, it achieves a weighted average c-$F1$ of 0.79 which is 3.95% improvement over the best performing baseline. In terms of m-$F1$ values, our weighted average ranks second next to CrossDomain. We reproduced CrossDomain baseline using their GitHub code (UKPLab). If the reproduced values are considered, our model outperforms all other models in m-$F1$ value as well.

Similarly, we compile the results for the general-domain setup in Table 7. In the non-noisy category, LESA obtains better m-$F1$ scores than three of the four state-of-the-art systems, i.e., it reports 0.77, 0.74, and 0.68 m-$F1$ scores compared to 0.71, 0.71, and 0.61 m-$F1$ scores of the comparative systems on MT, PE, and VG test sets, respectively. On WD, we observe similar m-$F1$ and c-$F1$ scores for both the best baseline and LESA. On the datasets in other categories, we observe comparative m-$F1$ scores; however, none of the baselines are consistent across all dataset – e.g., CrossDomain (Daxenberger et al., 2017) reports the best m-$F1$ scores on Twitter and OC, but yields (joint) fourth-best performance on WTP. Moreover, LESA yields the best m-$F1$ score across the seven datasets on average with $\geq 1\%$ improvements. On the other hand, we obtain best c-$F1$ scores for five out of seven datasets. In addition, LESA reports overall c-$F1$ of 0.75 with a significant improvement of $\geq 3\%$. Using a paired T-test, LESA showed significant statistical improvement compared against BERT in m-$F1$ and c-$F1$ for the noisy dataset with p-values .00017 and <.00001 respectively. Results were also significant for m-$F1$ and c-$F1$ for PE and m-$F1$ for WD. The small sample size in some datasets like MT and VG does not allow a reliable calculation of test statistics.

Since our work intends to developing a model that is able to detect claims irrespective of the source and origin of text, we also analyse the

		Example	Gold	Prediction	
				LESA	CrossDomain
TWR	x_1	*28 coronaoutbreak cases thus far in india italian tourists 16 their driver 1 kerala 3 cureddischarged agra 6 delhi 1 noida school dad telangana 1 coronavirusindia*	1	0	0
	x_2	*can we just call this a cure now*	0	0	1
MT	x_3	*Besides it should be in the interest of the health insurers to recognize alternative medicine as treatment, since there is a chance of recovery.*	0	1	1
PE	x_4	*On the other hand, fossil fuels are abundant and inexpensive in many areas*	0	1	1
	x_5	*Daily exercise will help also to develop children's brain function.*	1	1	0
OC	x_6	*Skinny Puppy is headlining Festival Kinetik !*	0	1	1
	x_7	*I guess I'm not desensitized enough to just forget about people being murdered in my neighborhood.*	1	1	0
WD	x_8	*No wonder 50 million babies have been aborted since 1973 .*	0	1	1

Table 9: Error analysis of the outputs. Red texts highlight errors.

weighted-average scores for each category in Table 8. We observe that `LESA` obtains the best c-$F1$ scores in each category, in addition to the best m-$F1$ score in non-noisy category as well. For the other two categories, `LESA` yields comparative performances. The results are better for noisy data than for non-noisy owing to the small size and skewness against claims in the latter's test set. Therefore, misclassification of a single claim causes severe penalization to c-$F1$.

B. Error Analysis

It is apparent from the results that all systems (including `LESA`) committed some errors in claim detection. Thus, in this section, we explore where our system misclassified the inputs by analysing some examples. Table 9 presents a few instances along with the gold labels and the predictions of the best-performing baseline, CrossDomain (Daxenberger et al., 2017), for comparison. In some cases, both `LESA` and CrossDomain failed to classify the instances correctly, whereas in others, `LESA` classifies the instances correctly but CrossDomain could not. We also report intuitions for the misclassification by `LESA` in some cases. The presence of numbers and statistics could be the reason behind the misclassifications in examples x_1 and x_8. Example x_3 contains two weak phrases (*'alternative medicine as treatment'* and *'there is a chance of recovery'*) which are most likely the cause of misclassification. The former might have been interpreted as suggestion backed up by some evidence, while in the latter phrase, `LESA` might have misinterpreted the optimism with claim. Furthermore, the phrase *'fossil fuels are abundant'* in example x_4 reflects world knowledge instead of a claim, as interpreted by `LESA`.

7 Conclusion

In this paper, we addressed the task of claim detection from online posts. To do this, we proposed a generic and novel deep neural framework, `LESA`, that leverages the pre-trained language model and two linguistic features, corresponding to the syntactic properties of input texts, for the final classification. Additionally, we tackled the texts from distinct sources for the claim detection task in a novel way. In particular, we categorized the input text as noisy, non-noisy, and semi-noisy based on the source, and modeled them separately. Subsequently, we fused them together through an attention module as the combined representation.

One of the major bottlenecks of claim detection in online social media platforms is the lack of qualitative annotation guidelines and a sufficiently large annotated dataset. Therefore, we developed a large Twitter dataset of $\sim 10,000$ manually annotated tweets for claim detection. In addition to our twitter dataset, we employed six benchmark datasets (representing either semi-noisy or non-noisy input channels) for evaluation of the proposed model. We compared the performance of `LESA` against four state-of-the-art systems and two pre-trained language models. Comparison showed the superiority of the proposed model with $\geq 3\%$ claim-F1 and $\geq 1\%$ macro-F1 improvements compared to the best performing baselines on average. As a by-product of the study, we released a comprehensive guideline for claim annotation.

Acknowledgement

We would like to thank Rituparna and LCS2 members for helping in data annotation. The work was partially supported by Accenture Research Grant, Ramanujan Fellowship, and CAI, IIIT-Delhi.

References

Immunizing the public against misinformation.

Alam, Shaden Shaar, Fahim Dalvi, Hassan Sajjad, Alex Nikolov, Hamdy Mubarak, Giovanni Da San Martino, Ahmed Abdelali, Nadir Durrani, Kareem Darwish, and Preslav Nakov. 2020. Fighting the covid-19 infodemic: Modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society. *arXiv preprint arXiv:2005.00033*.

Alberto Barrón-Cedeño, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, and Fatima Haouari. 2020. Checkthat! at clef 2020: Enabling the automatic identification and verification of claims in social media. In *Advances in Information Retrieval*, pages 499–507, Cham. Springer International Publishing.

O. Biran and O. Rambow. 2011. Identifying justifications in written dialogs. In *2011 IEEE Fifth International Conference on Semantic Computing*, pages 162–168.

Or Biran and Owen Rambow. 2011. Identifying justifications in written dialogs by classifying text as argumentative. *International Journal of Semantic Computing*, 05:363–381.

Carlson. 2020. Coronavirus tweets.

Sven Celin. 2020. Covid-19 tweets afternoon 31.03.2020.

Tuhin Chakrabarty, Christopher Hidey, and Kathleen McKeown. 2019. Imho fine-tuning improves claim detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 558–563.

Gullal S. Cheema, Sherzod Hakimov, and Ralph Ewerth. 2020. Check_square at checkthat! 2020: Claim detection in social media via fusion of transformer and syntactic features. *arXiv: 2007.10534*.

Emily Chen, Kristina Lerman, and Emilio Ferrara. 2020. Tracking social media discourse about the covid-19 pandemic: Development of a public coronavirus twitter data set. *JMIR Public Health Surveill*, 6(2):e19273.

J. Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37.

Johannes Daxenberger, Steffen Eger, Ivan Habernal, Christian Stab, and Iryna Gurevych. 2017. What is the essence of a claim? cross-domain claim identification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2066.

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

Trudy Govier. 2013. *A practical study of argument*. Cengage Learning.

Shreya Gupta, Parantak Singh, Megha Sundriyal, Md Shad Akhtar, and Tanmoy Chakraborty. 2021. LESA: Linguistic Encapsulation and Semantic Amalgamation Based Generalised Claim Detection from Online Content (Supplementary).

Ivan Habernal and Iryna Gurevych. 2015. Exploiting debate portals for semi-supervised argumentation mining in user-generated web discourse. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2127–2137, Lisbon, Portugal. Association for Computational Linguistics.

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. 2014. Context dependent claim detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1489–1500.

Ran Levy, Shai Gretz, Benjamin Sznajder, Shay Hummel, Ranit Aharonov, and Noam Slonim. 2017. Unsupervised corpus–wide claim detection. In *Proceedings of the 4th Workshop on Argument Mining*, pages 79–84.

Marco Lippi and Paolo Torroni. 2015. Context-independent claim detection for argument mining. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 185–191.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Hanyu Chwe Alexi Quintana Roy H. Perlis David Lazer James Druckman Mauricio Santillana Jennifer Lin John Della Volpe Matthew Simonson Jon Green Matthew A. Baum, Katherine Ognyanova. The state of the nation: A 50-state covid-19 survey report 14: Misinformation and vaccine acceptance.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv: 1301.3781*.

Alex Nikolov, Giovanni Da San Martino, Ivan Koychev, and Preslav Nakov. 2020. Team alex at clef checkthat! 2020: Identifying check-worthy tweets with transformer models. *arXiv:2009.02931*.

Andreas Peldszus and Manfred Stede. 2015. Joint prediction in MST-style discourse parsing for argumentation mining. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 938–948, Lisbon, Portugal. Association for Computational Linguistics.

Umair Qazi, Muhammad Imran, and Ferda Ofli. 2020. Geocov19: a dataset of hundreds of millions of multilingual covid-19 tweets with location information. *SIGSPATIAL Special*, 12(1):6–15.

Sara Rosenthal and Kathleen McKeown. 2012. Detecting opinionated claims in online discussions. In *2012 IEEE sixth international conference on semantic computing*, pages 30–37. IEEE.

Shane Smith. 2020. Coronavirus (covid19) tweets - early april.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Stephen E Toulmin. 2003. *The uses of argument*. Cambridge university press.

UKPLab. Ukplab/emnlp2017-claim-identification.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Andrew Whalen and Kevin Laland. 2015. Conformity biased transmission in social networks. *Journal of Theoretical Biology*, 380:542–549.

Evan Williams, Paul Rodrigues, and Valerie Novak. 2020. Accenture at checkthat! 2020: If you say so: Post-hoc fact-checking of claims using transformer-based models. *arXiv: 2009.02431*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Interpretability for Morphological Inflection: from Character-level Predictions to Subword-level Rules

Tatyana Ruzsics **Olga Sozinova** **Ximena Gutierrez-Vasques** **Tanja Samardžić**
Text Group, URPP Language and Space,
University of Zurich, Switzerland
{tatiana.ruzsics, olga.sozinova, ximena.gutierrezvasques, tanja.samardzic}@uzh.ch

Abstract

Neural models for morphological inflection have recently attained very high results. However, their interpretation remains challenging. Towards this goal, we propose a simple linguistically-motivated variant to the encoder-decoder model with attention. In our model, character-level cross-attention mechanism is complemented with a self-attention module over substrings of the input. We design a novel approach for pattern extraction from attention weights to interpret what the model learn. We apply our methodology to analyze the model's decisions on three typologically-different languages and find that a) our pattern extraction method applied to cross-attention weights uncovers variation in form of inflection morphemes, b) pattern extraction from self-attention shows triggers for such variation, c) both types of patterns are closely aligned with grammar inflection classes and class assignment criteria, for all three languages. Additionally, we find that the proposed encoder attention component leads to consistent performance improvements over a strong baseline.

1 Introduction

With the rise of deep learning, neural networks have been nowadays used in the process of decision making in various domains, as different as trading, medicine and government. Ethical considerations of such decisions have led to an increasing need for interpreting neural models which is a vivid research topic in machine learning community (Lipton, 2018; Gilpin et al., 2018). Although interpretability research in NLP is partly driven by ethics (Jacovi and Goldberg, 2020), there is a growing body of work exploring what linguistic properties emerge in neural models (Belinkov and Glass, 2019; Manning et al., 2020). The latter line of work aims to aid and scale up linguistic research, which is also the topic of this paper. Linguistics research focuses on uncovering patterns and regularities in language. Retrieving and analyzing structures of languages learned by neural agents can systematize our knowledge and, ideally, help us to come up with new regularities. Recent advances (Schrimpf et al., 2020) in testing hypotheses about human language processing using the growing suite of modern interpretable NLP are, indeed, inspiring, but still relatively limited to few languages. To scale up linguistic research, we require truly language-independent models developed for languages other than English (Bender, 2011). In return, understanding the model's decisions can lead to new ideas, how to improve the performance of the model on hard cases, i.e. on a particular linguistic phenomenon or language.

In our work, we concentrate on interpretability methods suitable for examining what knowledge of inflection morphology is captured by neural networks. Specifically, we consider a neural model that learns a mapping from a lemma and an abstract morpho-syntactic definition (MSD) to its inflected form. MSD comprises a part-of-speech (POS) tag as well as language-specific inflection tags. For example, given the Italian lemma *scolorire* "discolor" and MSD *V;IND;PRS;3;PL* (verb, 3rd person plural present indefinite form), the output is the word form *scoloriscono*. Datasets for this task of morphological reinflection are available for many languages and provide an opportunity to study a broad set of inflection phenomena.

Character-level encoder-decoder neural models with attention achieved very high performance on this task across many languages (see Cotterell et al., 2017, 2018; Vylomova et al., 2020 for the results of the recent shared tasks). Nevertheless, this class of models is typically not interpreted, and if it is so, the interpretation is limited to visualizing attention heatmaps on selected examples (see e.g. Aharoni and Goldberg 2017; Peters and Martins 2019).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3189–3201
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

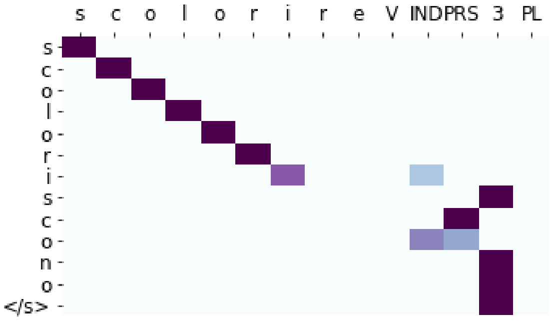


Figure 1: Example of a heatmap visualizing attention weights learned by the model of Peters and Martins (2019). The inflected form is generated from top to bottom.

We argue that per-example heatmaps provide very limited insight into what neural agent learns about a specific inflection phenomenon and how neural learning process can be related, in a systematic way, to the linguistic theory. Indeed, consider the previous Italian example: how would humans reason to convert the lemma to its inflected form?[1] In this work, we assume that humans apply the rules of grammar (implicit for native speakers, and explicit for language learners) when they perform this task. Specifically to our example, human reasoning could look as follows: the verb *scolorire* ends with the suffix *-ire* which determines its inflection class[2], according to which we construct the inflected form by copying the stem *scolor-* and adding an inflection suffix *-iscono*. What does a typical character-level model of an encoder-decoder class do? We visualize learned attention weights for an example of such a model in Fig. 1. By analyzing the most prominent alignments, we conclude that the model's character-level decisions can be combined into a) copying a substring of characters corresponding to the stem and b) generating characters for a substring corresponding to the inflection suffix. However, the decision why the model chose a particular inflection class is not visible.

To reach interpretability of models for inflection, we require methods that satisfy three conditions. First, the inflection model's decisions have to be aligned more closely to human reasoning by separating two kinds of operations: determining inflection class versus generating a string (given the assignment to a class). Second, a systematic analysis of model's decisions requires the extraction of inflection rules that are interpretable to humans. Finally, both of these factors require working with subword units rather than individual characters, the latter being the prevailing practice for inflection models.

In this paper, we propose a methodology - subword modification for a typical inflection model and interpretation method - which meet all three requirements. Our experiments on three typologically-different inflection phenomenon show that the linguistic rules elicited with our framework are highly consistent with linguistic knowledge (approximated by grammars). We evaluate the effectiveness of the proposed subword modification and find that, apart from its direct impact on interpretability, it leads to consistent performance improvements. To facilitate the use of our methods for linguistic research, we share our code[3].

2 Methodology: Interpretability for Inflection

How to make a character-level neural model for inflection more interpretable? We take the stance that, to make current models more interpretable, we should analyze their decisions in terms of subwords, i.e. clusters of characters rather than individual characters. Interpreting character decisions is outside of human intuition because of the double articulation principle (Martinet, 1967) which postulates that single phones are uninterpretable to humans, whereas the clusters of them form a mental linguistic representation of meaning in the speaker's mind. In writing, this distinction maps to the one between single characters and morphemes. [4]

From this perspective, we formulate the following desired properties for retrieving linguistic structures of inflection. For a given grammatical category (e.g. MSD *V;IND;PRS;3;PL* from the previous example in Italian),

---

[1] This is something that speakers implicitly (maybe, we do not know this for a fact) perform every time they use a word. This is also often an explicit task that language learners perform in the process of acquiring a new language.

[2] There are three inflection classes in Italian (Table 1). Verbs ending on *-ire* select between two classes: one (in the example) is more frequent in terms of types of verbs that belong to it, whereas the other one (*-ere* class) is selected by some very frequent verbs ending on *-ire*.

[3] `https://github.com/tatyana-ruzsics/interpretable-inflection`

[4] In this work, we refer to morphemes in their most general sense - apart from functional words and affixes we consider morphological processes, e.g. infixation and reduplication

	ARE	ERE/IRE	IRE
1SG	O	O	ISCO
2SG	I	I	ISCI
3SG	A	E	ISCE
1PL	IAMO	IAMO	IAMO
2PL	ATE	ETE	ITE
3PL	ANO	ONO	ISCONO

Table 1: Italian verbal inflectional classes, present tense

P1: identify substrings in an inflected form correspondig to morpheme(s) attributed to this category // *scolo**risc**ono*

P2: identify variation (dataset-wide) in the form of the morpheme (morphological class), e.g. all other substrings attributed to the same category // *-ano, -ono, -iscono*

P3: indicate whether triggers for such variation can be attributed to particular substrings of the lemma // *scolo**rire***

P4: identify triggers (dataset-wide) for each inflection class identified in P2 // *-are, -ere, -ire*

We propose to extract human-interpretable rules from an encoder-decoder model with attention. Specifically, to make it more interpretable, we modify such model by complementing the cross-attention mechanism with a novel component (§3) for self-attention over the subwords of the lemma. The task of this component is to help identify the morphological class. To extract the rules, we design a pattern extraction method (§4) that aggregates learned attention weights a) over a span of characters in a word and b) over a range of words in the same inflection category. Pattern extraction applied to character-level cross-attention weights retrieves linguistic rules satisfying requirements P1 and P2, whereas pattern extraction applied to subword-level self-attention weights, targets the requirements P3 and P4.

2.1 Case studies

To demonstrate the use of our approach for linguistic research, we will analyze how well patterns extracted with our proposed methodology align with human knowledge of typologically different phenomena. In morphological typology, cross-linguistic strategies to define the form and meaning of morphemes are described by typological parameters (Shopen, 1985; Dryer and Haspelmath, 2013; Bickel and Nichols, 2007) that separate different dimensions of the strategies. To select typologically different languages for our study, we focus on fusion and flexivity dimensions. Fusion classifies how easy it is to find a boundary between a morpheme and its phonological host and can take the following values: isolating (separate phonological word), concatenative (segmentable dependent morphemes) and nonlinear (not segmentable morphemes). Flexivity indicates whether variation in morpheme form can be explained by phonological processes (nonflexive) or not (flexive).

In our case studies, we consider verb conjugation rules and select three languages covering different degrees of fusion and flexivity: Finnish, Italian and Tagalog. In Finnish and Italian, morphemes are separable (concatenative fusion), whereas inflection in Tagalog is formed with affixes, including infixes, and reduplication (nonlinear fusion). Morphemes in Finnish can change their shape because of vowel harmony (nonflexive case). Forms of morphemes in Italian and Tagalog are selected by lexical context (flexive case) but distinctly. Italian verbs are conjugated with respect to three inflection classes defined by the lemma's ending (*-are*, *-ire*, and *-ere*, see Table 1), whereas assignment to an inflection class in Tagalog has no explicit rule.

3 Neural Model Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$

In this section, we introduce our novel component for **self-attention over subwords** of the lemma (**Self-Att$^{\text{sub}}$**). This module can be integrated into any variant of encoder-decoder system for inflection with **character-level cross-attention** (**Cross-Att$^{\text{ch}}$**). In this work, we show such integration to a sparse two-headed model of Peters and Martins (2019).[5] To explain the two-headed attention mechanism of the baseline model as well as our novel attention component, we first introduce the terminology of an abstract attention head module.

Attention Head Given an input sequence of vectors $\mathbf{H} = \mathbf{h}_1 \dots \mathbf{h}_J$, $\mathbf{h}_k \in \mathbb{R}^{D1}$ and a query vector $\mathbf{q} \in \mathbb{R}^{D2}$, attention head module computes two components: attention weights $a \in \mathbb{R}^J$ and an attention head vector $\mathbf{c} \in \mathbb{R}^{D1}$:

$$\alpha_j = \mathbf{q}^\intercal \mathbf{W}_a \mathbf{h}_j, \quad \mathbf{c} = \sum a_j \mathbf{h}_j \qquad (1)$$

where attention weights a are obtained by a mapping function from real values to probabilities, applied to alignment scores α. Following Peters and

---

[5] This model was among the winners of SIGMORPHON 2019 shared task (ranked first in terms of edit distance)

Martins (2019), we use `sparsemax` activations[6] as a mapping function. Hereafter we refer to the construction of an attention head vector $\mathbf{c}$ as scoring a sequence of vectors $\mathbf{H}$ with a query vector $\mathbf{q}$.

Baseline Cross-Att$^{\text{ch}}$ (Peters and Martins, 2019) Input lemma and MSD sequences are represented by separate bi-LSTM encoder states: $\mathbf{H}^u$ encodes characters in lemma, and $\mathbf{H}^v$ encodes tags in MSD. The decoder is a unidirectional LSTM with input feeding (Luong et al., 2015). At each prediction time t, it computes a hidden state $\mathbf{s}_t$ which is followed by the construction of two attention heads $\mathbf{u}_t$ and $\mathbf{v}_t$: one for lemma and one for MSD. They are calculated by scoring the respective representations, $\mathbf{H}^u$ and $\mathbf{H}^v$, with a query - decoder state s_t. The two attention heads are used to compute separate candidate attentional decoder states:

$$\tilde{\mathbf{s}}_{tu} = \mathtt{tanh}(\mathbf{W}_u[\mathbf{u}_t; \mathbf{s}_t]) \tag{2}$$

$$\tilde{\mathbf{s}}_{tv} = \mathtt{tanh}(\mathbf{W}_v[\mathbf{v}_t; \mathbf{s}_t]) \tag{3}$$

They are combined in a weighted sum to obtain an attentional decoder state $\tilde{\mathbf{s}}_\mathbf{t}$, where weights are calculated by a sparse gate vector $\mathbf{p_t} = [p_0, p_1] \in \mathbb{R}^2$:

$$\mathbf{p}_t = \mathtt{sparsemax}(\mathbf{W}_g[\mathbf{u}_t; \mathbf{v}_t; \mathbf{s}_t]) \tag{4}$$

$$\tilde{\mathbf{s}}_t = p_0\tilde{\mathbf{s}}_{tu} + p_1\tilde{\mathbf{s}}_{tv} \tag{5}$$

The attentional decoder state is fed into a sparse prediction layer. For the input feeding, the input to decoder comprises the predicted symbol embedding and gated attention vector c_t:

$$\mathbf{c}_t = p_0\mathbf{u}_t + p_1\mathbf{v}_t \tag{6}$$

The two-headed gate mechanism provides extra interpretability in the form of a three-way answer about what is relevant at a time step: the lemma, the inflections, or both.

Integrating Self-Att$^{\text{sub}}$ We depart from the existing character-level solution in that we assume that the input to the model - a (lemma, MSD) pair - is complemented with the segmentation of lemma into subwords.[7] We obtain an extra subword representation of lemma $\mathbf{H}^{subw}$ by averaging lemma representation vectors in $\mathbf{H}^u$ spanning characters within each subword. Besides the attention heads $\mathbf{u}_t$ and $\mathbf{v}_t$ computed at each generation step, we construct an additional attention head vector $\mathbf{m}$ which is computed once before the decoding stage. It is constructed by scoring the sequence of lemma subword representations in $\mathbf{H}^{subw}$ with a query vector $\mathbf{q}^{pos}$ corresponding to the encoding of the lemma's POS tag. This encoding is obtained by selecting a vector in MSD representation $\mathbf{H}^v$ corresponding to the position of the POS tag (e.g, POS tag *V* (verb) is in the first position in MSD *V;IND;PRS;3;PL*).

To integrate subword-level attention head $\mathbf{m}$ in the baseline system, we modify the gate layer in Eq. 4:

$$\mathbf{p}_t = \mathtt{sparsemax}(\mathbf{W}_g[\mathbf{m}; \mathbf{u}_t; \mathbf{v}_t; \mathbf{s}_t]); \tag{7}$$

In this way, the gate mechanism (and decoding) is expected to be informed with a signal for inflection class selection when such signal can be attributed to specific character spans (subwords) in the lemma. The attention over subwords is static and shared across target positions, aiming to separate the signal of the class assignment it conveys from local character transformations, given this assignment.

4 Pattern extraction

To extract linguistic rules from the trained model Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$, we represent its knowledge of inflection as a database. To populate it, we analyze predictions of the model on a dataset. The latter can be the original task data or a dataset collected to study a specific inflection phenomenon. Then, for each example in the dataset, we populate the knowledge database with the example itself and two patterns, which are extracted from the learned attention weights. The first one is a transformation pattern (§4.1) obtained by applying our pattern extraction method to learned cross-attention weights (Cross-Att$^{\text{ch}}$ component). This method can be applied to any inflection model embedded into encoder-decoder paradigm with attention. The second pattern is over the lemma subwords (§4.2) which is obtained from attention weights of the novel Self-Att$^{\text{sub}}$ component. Finally, we explain how populated in this way knowledge database can be queried to study inflection phenomena (§4.3).

4.1 Cross-Att$^{\text{ch}}$ Transformation Patterns

This method maps each example *(lemma, MSD)* → *inflected form* to a transformation pattern of a

---

[6] `sparsemax` activations (Martins and Astudillo, 2016) serve as a sparse (and therefore, more interpretable) alternative to a commonly used attention function - `softmax`. The latter yields dense attention weights: all elements in the input always make at least a small contribution to the decision.

[7] Such representation can be obtained with any off-the-shelf segmentation algorithms, e.g. BPE (Sennrich et al., 2016a) or Morfessor (Smit et al., 2014).

Input:

$X = \underset{123456789}{scolorire}$ $\quad F = \underset{1}{V}\ \underset{2}{IND}\ \underset{3}{PRS}\ \underset{4}{3}\ \underset{5}{PL}$

$Y = \underset{123456789101112}{scoloriscono}$ $\quad A^X$ and A^F as in Figure 1

Step 1: Transform att. weights into 'salient' alignments

$A = [\underbrace{X_1 \cdots X_7}_{copy}, F_4, F_3, F_2, F_4, F_4]$

Step 2: Inverse A and group pred. steps by gen. type

$X_1 \rightarrow \{c : [1]\}$ $\quad F_4 \rightarrow \{f4 : [8, 11, 12]\}$

$\dots$ $\quad F_3 \rightarrow \{f3 : [9]\}$

$X_7 \rightarrow \{c : [7]\}$ $\quad F_2 \rightarrow \{f2 : [10]\}$

Step 3: Replace char. in X and Y with indexed gen. type

$P^{tr}(X) = \mathrm{c}^1 \underset{7}{\cdots} \mathrm{c}^1\ \textit{re}$

$P^{tr}(Y) = \mathrm{c}^1 \underset{7}{\cdots} \mathrm{c}^1\ \mathrm{f4}^1\ \mathrm{f3}^1\ \mathrm{f2}^1\ \mathrm{f4}^2\ \mathrm{f4}^2$

Step 4: Collapse adjacent symbols

$P^{tr}(X)$=$\mathrm{c}^1\ \textit{re}$

$P^{tr}(Y) = \mathrm{c}^1\ \mathrm{f4}^1\ \mathrm{f3}^1\ \mathrm{f2}^1\ \mathrm{f4}^2$

Output:

$\mathrm{c}^1\ \textit{re} \rightarrow \mathrm{c}^1\ \mathrm{f4}^1\ \mathrm{f3}^1\ \mathrm{f2}^1\ \mathrm{f4}^2$

($\mathrm{c}^1 \leftrightarrow$ *scolori*, $f4^1 \leftrightarrow$ *s*, $f3^1 \leftrightarrow$ *c*, $f2^1 \leftrightarrow$ *o*, $f4^2 \leftrightarrow$ *no*)

Table 2: Illustration of Cross-Att$^{\text{ch}}$ Transformation Patterns algorithm applied to the example in Fig. 1.

form $P^{tr}(lemma) \rightarrow P^{tr}$*(inflected form)*. Formally, the input to the algorithm is a lemma $X = x_1, \dots x_n$, MSD $F = f_1 \dots f_l$, predicted target form $Y = y_1 \dots y_m$ and cross-attention weights over lemma characters $A^X = a_1^X \dots a_m^X, a_i^X \in \mathbb{R}^n$ and over MSD tags $A^F = a_1^F \dots a_m^F, a_i^F \in \mathbb{R}^l$.[8] The output is a string of a form $P^{tr}(X) \rightarrow P^{tr}(Y)$ where the constructed pattern representation P^{tr} for the lemma and target are built through the following steps (shown in Table 2 for our example in Fig. 1):

Step 1. Transform input attention weights A^x and A^f into "salient" alignments A: Each component a_j of salient alignments $A = a_1 \dots a_m$ is a set of input positions (in lemma X and/or MSD F) that provide the most significant contributions to predicting a character in Y at position j. We denote positions by capitalized symbols, i.e. $F1$ for position 1 in F, to reflect the difference between the position's index and value. Salient alignments are built by applying a filtering function ϕ to attention weights at each predicted position: $\phi : [a_j^X; a_j^F] \rightarrow a_j$. In the following, we illustrate how our algorithm works for the simplest choice of the filtering function, max-pooling, which simply selects one input position with the highest attention weight.[9] In our running example, this strategy results in only one element for each component a_j: e.g. $a_7 = X_7$.

Step 2. Inverse mapping A and group prediction steps by generation type: By inverting salient alignments, we construct a mapping from input positions to prediction steps grouped by a symbol corresponding to generation type. The latter is identified for each alignment a_j by the type of input: we denote generation from the lemma's characters ($a_j = X_i$) by symbol *g*, whereas that from a tag ($a_j = F_k$) is denoted by indexed symbol fk. The special case of copying a character from the lemma, i.e. $a_j = X_i$ and $x_i = y_j$, is denoted by symbol *c*. Thus, a position in F can be mapped to only one group of prediction steps (the type of generation is unique and defined by the tag's position), whereas that in X can be mapped to up to two groups, *g* and *c*. Some input positions might be absent in the constructed mapping, if not present in salient alignments, e.g. X_9 in Table 2.

Step 3. Replace characters in X and Y with indexed generation type symbols: We index (in the order of input positions) triples of salient alignments (input position, generation step, generation type) identified in the previous step. Then, we construct patterns of lemma and inflected form, by replacing characters at aligned positions with an indexed value of the generation type symbol, e.g. $(c, X_j, Y_k) \rightarrow index; x_j \rightarrow c^{index}; y_k \rightarrow c^{index}$. In X, this can result in an aggregated symbol, e.g. replacing X_i with $c^{1;2}; g^1$ means that position X_i is aligned to three target positions, two of which are generated by copying x_i. As illustrated in our running example, we use the same index value in two special cases: a) a whole target substring was copied, and b) a whole target substring was generated by the same tag. We keep the track of symbolic mappings from characters to indexed generation symbols that replace them.

Step 4. Collapse adjacent symbols: Scan representations $P^{tr}(X)$ and $P^{tr}(Y)$, built at the previous step and iteratively collapse adjacent symbols of the same value. At the same time, we update

---

[8] We assume that the sum of weights in a combined vector $[A^X;A^F]$ is 1. In Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$ model this is achieved by scaling cross-attention weights for the lemma and MSD with corresponding gate values. Another way, typical for neural inflection models of encoder-decoder class, is to run cross-attention over a concatenation of the lemma's characters and MSD's tags.

[9] `sparsemax` activations provide another choice to filtering function by keeping the input positions corresponding to nonzero attention weights. In our example, this would result in e.g. $a_7 = \{X_7, F_2\}$. We refer to a more general form of the algorithm covering such case in the Appendix.

the symbolic mapping: if two adjacent symbols are collapsed, we replace their string mappings with a single mapping from the strings concatenation to the generation symbol.

The idea behind the inverse mapping and indexing in steps 2 and 3 is to ensure a unique way of indexing generation symbols across all data pairs. The indices themselves are essential to keep a one-to-one mapping from substrings to the generation symbols they are replaced with. Both factors come into play when we query the knowledge database for an inflection phenomenon (§4.3).

4.2 Self-Att$^{\text{sub}}$ Lemma Patterns

This algorithm takes as an input a data example (X, Y, F), along with a segmented lemma representation $S(X) = s_1 \ldots s_p$ and learned self-attention weights over lemma's subwords: $a^{S(X)} \in \mathbb{R}^p$. The output is a pattern for salient subwords in lemma $P^l(X)$ which is built with a similar procedure as described above where indexing steps 2 and 3 are skipped.

First, we transform self-attention weights $a^{S(X)}$ into salient alignments a by applying a filtering function: $\phi : a^{S(X)} \rightarrow a$, thereby identifying a set of subword positions with the most significant contribution to the overall generation process (any type of filtering function described in the previous subsection can be applied). Afterward, we replace all subwords in the input lemma at nonsalient positions, $S_j \notin a$, with a dedicated symbol, e.g. asterisk *. Finally, we iteratively merge adjacent asterisk symbols to obtain a more general pattern. To illustrate with our running example, given a segmented representation of lemma $S(X) = s|col|or|i|re$ and salient alignments $a = \{S4, S5\}$, obtained by filtering input positions with nonzero self-attention weights, the resulting pattern for salient subwords in the lemma is $P^l(X) = *ire$.

4.3 Querying Patterns

As a result of applying the previous two methods, each data example (X, Y, F), along with segmented lemma representation $S(X)$ and learned attention weights (A^X, A^F, $a^{S(X)}$), can be mapped into two items: Cross-Att$^{\text{ch}}$ transformation pattern $P^{tr}(X) \rightarrow P^{tr}(Y)$ and Self-Att$^{\text{sub}}$ pattern for salient subwords in lemma $P^l(X)$. The data examples along with the extracted patterns are stored in a knowledge database. To systematically study how the neural model handles a specific linguistic phenomenon of interest, the database can be queried, for patterns and examples, with a phenomenon's formalization in a form of regular expressions applied to the lemma, inflected form or MSD. Selected with a query examples are then grouped by their patterns (either transformation or lemma ones) resulting in each group representing an induced linguistic rule for the phenomenon.

At this stage, to make the patterns more readable, we perform an unmasking operation within each group: if a particular symbol is used to substitute one substring that is the same for all examples within a group, we replace the symbol back with this substring. For instance, if the pattern from our example c^1 *re* $\rightarrow$ $\text{c}^1\,\text{f4}^1\,\text{f3}^1\,\text{f2}^1\,\text{f4}^2$ represents one such group, and symbol f4^2 is used to substitute only one string *no*, which is the same across all data points in the group, we can unmask the string, to obtain a pattern c^1 *re* $\rightarrow$ $\text{c}^1\,\text{f4}^1\,\text{f3}^1\,\text{f2}^1$ *no*.

5 Experiments and Results

We perform three case studies, introduced in §2.1, to demonstrate how our framework allows querying patterns learned by an inflection neural model. The goal of our experiments is to assess how well the extracted patterns correspond to known inflection rules. To see whether our modifications to the inflection model affect its performance, we check the inflection accuracy on the analyzed languages and compare it with the original character-level model.

We use data from SIGMORPHON shared task: 2018 edition for Italian and Finnish (10K/1K/1K examples in train/development/test data), and 2020 edition for Tagalog (1,870/236/478). For each language, we train Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$ model with batch size 4, beam size 1 and other hyperparameters as reported in Peters and Martins (2019). To produce segmented lemma input, we use the BPE method (Gage, 1994; Sennrich et al., 2016b) with 1K merges on a token list (100K examples) extracted from WikipediaDumps articles.[10].

Using model's predictions on the concatenation of train, development and test set, we query Cross-Att$^{\text{ch}}$ and Self-Att$^{\text{sub}}$ patterns As a filtering function, we keep only nonzero weights for Self-Att$^{\text{sub}}$ patterns, whereas we choose max-pooling for Cross-Att$^{\text{ch}}$ ones, as on average sparse activations assign

---

[10] We use archives of the name format enwiki-20190920-pages-articles.xml.bz2 from `https://ftp.acc.umu.se/mirror/wikimedia.org/dumps/`

nonzero weight to one input feature. To systematically examine whether the classes of patterns extracted are correct and cover the data adequately, we report two metrics, namely a) number of examples selected with a query and how many of them are grouped by each pattern, and b) model accuracy (correct predictions) with respect to the number of examples per selection with a query and per group pattern.

Cross-Att^{ch}: Transformation Patterns For each language, we define specific queries: 3rd person plural present tense for Italian (MSD=*V;IND;PRS;3;PL*), 3rd person plural present positive imperative for Finnish (MSD=*V;ACT;PRS;POS;IMP;3;PL*) and imperfective aspect with agent semantic role for Tagalog (MSD=*V;IPFV;AGFOC*). The choice of MSDs is rather arbitrary: for illustrative purposes we select grammatical categories that contain enough examples to represent form variation of the corresponding morpheme. Table 3 present the extracted Cross-Att^{ch} patterns. For each query, we show patterns that group at least 5% of the examples selected with a query. The patterns are sorted by their number of examples in a decreasing order. For each presented pattern, we show an example mapped to this pattern and symbol mapping information. The latter lists, for each symbol in the pattern, all substrings mapped to this symbol along with their frequencies (within a group), if the number of distinct substrings is less than five elements. Otherwise, we show average length (≈) of substrings mapped to this symbol, or exact length (=), if it is the same for all of them. These symbol mappings also include bijection cases (↔) that were unmasked after grouping examples (as described in §4.3)

Transformation Patterns	No.of/Acc
Finnish	
Q: MSD=*V;ACT;PRS;POS;IMP;3;PL*	46/0.91
(1) $\mathbf{c}^{1}\ \mathbf{a} \rightarrow \mathbf{c}^{1}\ \mathbf{k\ oo\ t}$ *karsastaa* : *karsastakoot* $\lvert \mathbf{c}^{1} \rvert \approx 7.1$; $\mathbf{f6}^{1} \leftrightarrow k$; $\mathbf{f5}^{1} \leftrightarrow oo$; $\mathbf{f7}^{1} \leftrightarrow t$	23/1.00
(2) $\mathbf{c}^{1}\ \mathbf{ä} \rightarrow \mathbf{c}^{1}\ \mathbf{k\ öö\ t}$ *mylviä* : *mylvikööt* $\lvert \mathbf{c}^{1} \rvert \approx 7.4$; $\mathbf{f6}^{1} \leftrightarrow k$; $\mathbf{f5}^{1} \leftrightarrow öö$; $\mathbf{f7}^{1} \leftrightarrow t$	7/1.00
(3) $\mathbf{c}^{1}\ \mathbf{d\ a} \rightarrow \mathbf{c}^{1}\ \mathbf{k\ oo\ t}$ *promovoida* : *promovoikoot* $\lvert \mathbf{c}^{1} \rvert \approx 8.4$; $\mathbf{f6}^{1} \leftrightarrow k$; $\mathbf{f5}^{1} \leftrightarrow oo$; $\mathbf{f7}^{1} \leftrightarrow t$	5/1.00
(4) $\mathbf{c}^{1}\ \mathbf{l\ a} \rightarrow \mathbf{c}^{1}\ \mathbf{k\ oo\ t}$ *aaltoilla* : *aaltoilkoot* $\lvert \mathbf{c}^{1} \rvert \approx 7.7$; $\mathbf{f6}^{1} \leftrightarrow k$; $\mathbf{f5}^{1} \leftrightarrow oo$; $\mathbf{f7}^{1} \leftrightarrow t$	3/1.00
Italian	
Q: MSD=*V;IND;PRS;3;PL*	255/0.99
(1) $\mathbf{c}^{1}\ \mathbf{a\ r\ e} \rightarrow \mathbf{c}^{1}\ \mathbf{a\ n\ o}$ *zampicare* : *zampicano* $\lvert \mathbf{c}^{1} \rvert \approx 6.7$; $\mathbf{f3}^{1} \leftrightarrow a$; $\mathbf{f4}^{1} \leftrightarrow n$; $\mathbf{f5}^{1} \leftrightarrow o$	149/1.00
(2) $\mathbf{c}^{1}\ \mathbf{a\ r\ s\ i} \rightarrow \mathbf{si}\ \mathbf{c}^{1}\ \mathbf{a\ n\ o}$ *impaperarsi* : *si impaperano* $\lvert \mathbf{c}^{1} \rvert \approx 6.8$; $\mathbf{f1}^{1} \leftrightarrow si$; $\mathbf{f3}^{1} \leftrightarrow a$; $\mathbf{f4}^{1} \leftrightarrow n$ $\mathbf{f5}^{1} \leftrightarrow o$	40/1.00
(3) $\mathbf{c}^{1}\ \mathbf{e\ r\ e} \rightarrow \mathbf{c}^{1}\ \mathbf{o\ n\ o}$ *rirompere* : *rirompono* $\lvert \mathbf{c}^{1} \rvert \approx 7.1$; $\mathbf{f2}^{1} \leftrightarrow o$; $\mathbf{f4}^{1} \leftrightarrow n$; $\mathbf{f5}^{1} \leftrightarrow o$	23/1.00
(4) $\mathbf{c}^{1}\ \mathbf{r\ e} \rightarrow \mathbf{c}^{1}\ \mathbf{s\ c\ o\ n\ o}$ *scolorire* : *scoloriscono* $\lvert \mathbf{c}^{1} \rvert \approx 7.3$; $\mathbf{f4}^{1} \leftrightarrow s$; $\mathbf{f5}^{1} \leftrightarrow c$; $\mathbf{f2}^{1} \leftrightarrow o$ $\mathbf{f4}^{2} \leftrightarrow n$; $\mathbf{f5}^{2} \leftrightarrow o$	16/1.00
Tagalog	
Q: MSD=*V;IPFV;AGFOC*	377/0.86
(1) $\mathbf{c}^{1;2}\ \mathbf{c}^{3;4}\ \mathbf{c}^{5} \rightarrow \mathbf{n}\ \mathbf{f3}^{1}\ \mathbf{c}^{1}\ \mathbf{c}^{3}\ \mathbf{c}^{2}\ \mathbf{c}^{4}\ \mathbf{c}^{5}$ *paalam* : *nagpapaalam* $\lvert \mathbf{c}^{1} \rvert=1$; $\lvert \mathbf{c}^{2} \rvert=1$; $\lvert \mathbf{c}^{1;2} \rvert=1$; $\lvert \mathbf{c}^{3;4} \rvert=1$ $\mathbf{c}^{3}$: {*a* (82), *u* (34), *i* (23), *e* (3)} $\mathbf{c}^{4}$: {*a* (82), *u* (34), *i* (23), *e* (3)} $\mathbf{f3}^{1}$:{*ag* (131), *a* (8), *ang* (1), *an* (2)} $\mathbf{f2}^{1} \leftrightarrow n$; $\lvert \mathbf{c}^{5} \rvert \approx 3.3$	142/0.90
(2) $\mathbf{c}^{1}\ \mathbf{c}^{2;3}\ \mathbf{c}^{4} \rightarrow \mathbf{f2}^{1}\ \mathbf{f3}^{1}\ \mathbf{c}^{2}\ \mathbf{c}^{1}\ \mathbf{c}^{3}\ \mathbf{c}^{4}$ *hiram* : *humihiram* $\lvert \mathbf{f2}^{1} \rvert=1$; $\lvert \mathbf{c}^{1} \rvert=1$;$\lvert \mathbf{c}^{2;3} \rvert=1$; $\lvert \mathbf{c}^{4} \rvert \approx 3.1$ $\mathbf{f3}^{1}$: {*um* (71), *am* (3), *k* (1), *as* (1), *an* (2)} $\mathbf{c}^{2}$: {*i* (24), *a* (35), *u* (17), *o* (2)} $\mathbf{c}^{3}$: {*i* (24), *a* (35), *u* (17), *o* (2)}	78/0.86

Table 3: Cross-Att^{ch} transformation patterns. *Q* is a query regular expression. The number of examples (*No of*) and accuracy (*Acc*) are shown per selection with a query and per group pattern.

We observe that in all three cases, the patterns recover inflection morphemes listed in grammars for studied grammatical categories as well as their form variation. For Finnish, the model correctly identifies morpheme -*koot* as well as its variant -*kööt* because of vowel harmony. Additionally, the patterns (3) and (4) display morphophonological processes on morphemes boundaries: if a stem ends with -*d* or -*l*, this ending is removed from the final form. In Italian, the morphemes for all three inflection classes are present in the patterns: -*ano* (-*are* class) -*ono* (-*ere* class) and -*scono* (-*ire* class). Besides, the separation of reflexive ending *si* is visible (pattern (2)) for the -*are* class. Tagalog patterns detect two frequent inflection classes corresponding to so-called *um*-verbs and *nag*-verbs. Analyzing symbolic mappings, we conclude that the pattern (1) encodes prefixation (the most frequent prefix in this group is *nag*) with subsequent copying of the first syllable and copying of the full lemma string. The pattern (2) expresses reduplication of the first consonant (which interestingly gets aligned to a tag rather than to the first character of the lemma), generating infix (the most frequent infix in this group is *um*), copying the second character of the lemma (vowel, as seen from the mapping statistics), and then copying of full lemma string.

Self-Att$^{\text{sub}}$: Lemma Patterns We analyze Self-Att$^{\text{sub}}$ lemma patterns to determine whether our model uses indeed subword segments when choosing the specific variant of a morpheme. Concretely, we use regular expressions on the target form to select examples corresponding to a specific form of morpheme, identified above with transformation patterns. Then, we map selected examples to their Self-Att$^{\text{sub}}$ patterns. Table 4 presents the queries and extracted patterns.[11] For each query, we list the most frequent patterns (sorted by frequency in a decreasing order) along with one segmented lemma example mapped to the pattern.[12] The segments of lemma examples, identified as salient (and presented in the patterns) are highlighted in bold.

We conclude that the subword regions identified by Self-Att$^{\text{sub}}$ patterns conform to a great extent to triggers of morpheme form variation listed in grammars. We note that although the regions for finding such clues (when they are phonological or lexical, and frequent) look plausible, their form is influenced by the results of BPE segmentation and may not be perfectly aligned with grammars. For example, Italian patterns show that the model's focus is on the endings of lemmas for all three classes. In case of reflexive verbs, where reflexive ending -*si* tends to be separated into a separate subword by BPE, the model correctly places focus on a more informative penultimate segment. The patterns extracted for Finnish, display the grammar rules too: the focus on the lemma endings -*aa* and -*ua* for the first group, and -*ää*/-*ä* for the second group, points directly to the harmony of back and front vowels, respectively. The model does not search for the clues in the vowel patterns of the stem but chooses a smart strategy to focus directly on the inflection endings for lemmas: they are frequent and already agree with the vowels found elsewhere in the stem to the left.

Lemma Patterns	No. of/Acc
Finnish	
Q: gold target=*koot & MSD=msd_fin	37/0.97
aa** (*kar*|*sa*|*st*|aa***)	8/1.00
ua** (*ku*|*or*|*ett*|ua***)	5/1.00
Q: gold target=*kööt & MSD=msd_fin	9/1.00
ää** (*jä*|*n*|*ist*|ää***)	3/1.00
ä** (*v*|*et*|*ele*|*hti*|ä***)	3/1.00
Italian	
Q: gold target=*scono & MSD=msd_it	23/1.00
re** (*in*| *z*| *o*| *ti*|*chi*| ***re)	9/1.00
ire** (*s*| *col*| *or*| ***ire)	7/1.00
ir (*re*| *in*| *ser*| ***ir***| *si*)	6/1.00
Q: gold target=*ano & MSD=msd_it	189/1.00
are** (*z*| *am*| *pic*| ***are)	149/1.00
arsi** (*im*| *pa*| *per*| ***arsi)	26/1.00
car (*ri*| *mb*| *ec*| ***car***| *si*)	3/1.00
Q: gold target=*ono & !(*scono) & MSD=msd_it	41/0.95
ere** (*ri*| *otten*| ***ere)	19/0.95
dere** (*te*| *le*| *ve*| ***dere)	10/1.0
ger (*cos*| *par*| ***ger***| *si*)	3/1.00

Table 4: Self-Att$^{\text{sub}}$ Lemma Patterns. *Q* is a query regular expression, *msd_fin* is *V;ACT;PRS;POS;IMP;3;PL*, and *msd_it* is *V;IND;PRS;3;PL*. The number of examples (*No of*) and accuracy (*Acc*) are shown per selection with a query and per group pattern.

Self-Att$^{\text{sub}}$: Performance Impact We evaluate the impact of the novel Self-Att$^{\text{sub}}$ component by comparing the performance of Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$ with that of the baseline model, Cross-Att$^{\text{ch}}$. For reference, we include the results of a) the hard monotonic attention (HMA) system of Wu and Cotterell (2019) which currently holds as the state-of-the-art on the reinflection task by rerunning their code; b) a variant of our system, Cross-Att$^{\text{ch}}$Self-Att$^{\text{ch}}$ where the encoder attention module is run over the characters of the lemma, instead of subwords. The latter corresponds to a limiting case of lemma segmentation where each character is a segment. We report accuracy and edit distance on the test set in Table 5. Additionally, we provide information on the number of trained parameters for each model. The number of parameters for Cross-Att$^{\text{ch}}$Self-Att$^{\text{sub}}$ model is the same as for its character variant Cross-Att$^{\text{ch}}$Self-Att$^{\text{ch}}$. The difference in the number of parameters across the languages is due to the variation of their character vocabulary sizes.

---

[11] For Tagalog, we note that we do not find any frequent patterns for e.g. a query "gold target=*nag**", which is in line with no explicit criteria for inflection class assignment in this language.

[12] We refer to the Appendix, Tables 6-7 for the full list of extracted patterns.

	Baseline	Our model	Comparison	
	Cross-Att^{ch}	Cross-Att^{ch}Self-Att^{sub}	Cross-Att^{ch}Self-Att^{ch}	HMA
Italian	95.40 (0.09) [*1,742K*]	96.70 (0.09) [*1,783K*]	**97.40** (**0.06**) [*1,783K*]	96.80 (0.29) [*8,647K*]
Finnish	93.80 (0.13) [*1,758K*]	**94.40** (**0.09**) [*1,798K*]	93.60 (0.12) [*1,798K*]	93.90 (0.13) [*8,709K*]
Tagalog	65.75 (1.21) [*1,739K*]	**69.98** (**0.92**) [*1,780K*]	66.81 (1.00) [*1,780K*]	63.39 (1.53) [*8,623K*]

Table 5: Accuracy (and edit distance) on the test set. The number of model parameters is given in squared brackets.

We observe, that the Cross-Att^{ch}Self-Att^{sub} model shows systematic improvements across all three languages over the baseline and reference models. Regarding the level of segmentation, the Cross-Att^{ch}Self-Att^{ch} system achieves higher results on Italian, where indeed, class variation can be associated with a certain character in a certain position. In terms of the number of trained parameters, the improvements due to the Self-Att^{sub} component are achieved by only adding a relatively small number of extra parameters compared with the baseline model, Cross-Att^{ch} . We also note that the performance of our systems is higher or on par with the state-of-the art model HMA, whereas the latter has an on-average sevenfold increase in the number of parameters in comparison with to that of Cross-Att^{ch}Self-Att^{sub} and Cross-Att^{ch}Self-Att^{ch} .

6 Discussion and Future Work

In the following, we discuss our proposed methodology in terms of two aspects, namely, interpretability for inflection (in terms of typological parameters) and ideas for performance improvement.

Interpretability for Inflection In terms of the typological parameter of **fusion**, the results of our experiments illustrate that our Cross-Att^{ch} pattern approach can effectively extract rules for concatenative morpheme forms as well as reduplication processes. What is beyond, at the moment, are nonlinear processes that are not always visible in orthography, e.g. tonal changes and internal stem changes. The latter, for example, is demonstrated by root and pattern morphology in Arabic and Hebrew, for which standard orthographies do not indicate most vowels.

Regarding **flexivity**, our Self-Att^{sub} pattern method can identify phonological (visible in orthography) as well as lexical triggers to the variation of inflection morpheme's form. However, the case of suppletive forms (English *go*→ *went*) would not be identifiable in patterns. Although suppletive cases are likely to be fairly rare in terms of word types, they seem to be only maintained in high-frequency words (Bybee, 1985). Therefore, although affecting only a small number of words, suppletion might be visible in patterns when studied together with word frequency (which is, at the moment, not possible because of the current practices for building inflection generation datasets).

The parameter of **exponence** encodes the extent to which single morphemes express multiple morphosyntactic features. For the class of neural models currently used for inflection generation, it is not possible to see a clear correspondence between the meaning assigned by humans and the model: as we see from Fig. 1 which illustrates polyexponence in Italian inflection, the model assigns separate characters of inflection morpheme -*scono* to different tags, whereas for humans, it is hard to break down this morpheme into smaller meaningful parts.[13]

Performance Future work can evaluate the impact of Self-Att^{sub} in combination with frequently used induction biases[14], as well as transformers paradigm, which recently proved to be effective on the task. (Vylomova et al., 2020).

7 Conclusion

We propose a novel approach for interpreting neural inflection models by extracting patterns from attention weights. To enhance the interpretability of this class of models, we design a linguistically motivated attention component over subwords that leads to a systematic performance improvement. Our experiments with linguistic rules induction illustrate the great potential of our methodology for linguistic research scaled to diverse typology.

Acknowledgments

This work has been partially supported by the SNSF grant no. 176305.

---

[13] This challenge is exemplified by the input features identity problem in attention weights (Brunner et al., 2019).

[14] For example, hard monotonic attention which is highly effective for Indo-European languages with concatentive morphology.

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2004–2015, Vancouver, Canada. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Emily M Bender. 2011. On achieving and evaluating language-independence in nlp. *Linguistic Issues in Language Technology*, 6(3):1–26.

Balthasar Bickel and Johanna Nichols. 2007. Inflectional morphology. *Language typology and syntactic description*, 3(2):169–240.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. In *International Conference on Learning Representations*.

Joan L Bybee. 1985. *Morphology: A study of the relation between meaning and form*, volume 9. John Benjamins Publishing.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

Matthew S Dryer and Martin Haspelmath. 2013. The world atlas of language structures online.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. 2018. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 80–89.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? *arXiv preprint arXiv:2004.03685*.

Zachary C. Lipton. 2018. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

D Christopher Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences of the United States of America*.

André Martinet. 1967. Eléments de linguistique générale. *Colin*.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. volume 48 of *Proceedings of Machine Learning Research*, pages 1614–1623, New York, New York, USA. PMLR.

Ben Peters and André FT Martins. 2019. It–ist at the sigmorphon 2019 shared task: Sparse two-headed models for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 50–56.

Martin Schrimpf, Idan Blank, Greta Tuckute, Carina Kauf, Eghbal A. Hosseini, Nancy Kanwisher, Joshua Tenenbaum, and Evelina Fedorenko. 2020. Artificial neural networks accurately predict language processing in the brain. *bioRxiv*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Timothy Shopen. 1985. *Language Typology and Syntactic Description: Volume 3*, volume 3. Cambridge University Press.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24, Gothenburg, Sweden. Association for Computational Linguistics.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J Mielke, Shijie Wu, Edoardo Ponti, Rowan Hall Maudslay, Ran Zmigrod, Josef Valvoda, Svetlana Toldova, et al. 2020. Sigmorphon 2020 shared task 0: Typologically diverse morphological inflection. *arXiv preprint arXiv:2006.11572*.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

A Cross-Att^{ch} Transformation Patterns Algorithm

In this Section, we formalize Steps 2 and 3 of the algorithm for pattern extraction from character-level cross-attention weights presented in §5.

Algorithm 1: (Step 2) Inverse salient alignments mapping A and group prediction steps by generation type

```
Inputs:
$X \leftarrow [x_1 \dots x_n]$ ;                         // Lemma
$F \leftarrow [f_1 \dots f_n]$ ;                         // MSD
$Y \leftarrow [y_1 \dots y_m]$ ;                         // Target
$A \leftarrow [a_1 \dots a_m]$ ;           // Salient alignments
Init: $X\_pos\_map = \{\}$, $F\_pos\_map = \{\}$ will
  store salient mappings from input positions to
  prediction steps, grouped by generation type.
for $a_j$ in $A$:
    for $P$ in $a_j$: ;      // salient alignments to $y_j$
        if $P == X_i$: ;            // aligned to lemma
            if $x_i == y_j$: ;                     // copy
                add $j$ to $X\_pos\_map[X_i][c]$
            else:
                add $j$ to $X\_pos\_map[X_i][g]$
        else ($P == F_k$): ;          // aligned to tag
            add $j$ to $F\_pos\_map[F_k]$
Outputs: $X\_pos\_map$, $F\_pos\_map$
```

Algorithm 2: (Step 3.1) Replace characters in Y with indexed generation type symbols using salient alignments to F

```
Inputs: $F\_pos\_map$; $\tilde{Y}$=copy($Y$)
Init: $f2prev\_target = \{\}$; $f2ind = \{\}$
for $F_k$ in $F$:
    if $F_k$ in $F\_pos\_map$:
        for $j$ in $F\_pos\_map[F_k]$: ;     // $Y$ indexes
            if $f_k$ not in ($f2prev\_target$)):
                $f2ind[f_k]$=1; ;   // If nothing was aligned yet to $f_k$, we create an index
                if $\tilde{y}_j$ is not replaced:
                    $\tilde{y}_j \rightarrow f_k^1$
                else:
                    $\tilde{y}_j$+= $f_k^1$
            else:
                if ($f2prev\_target[f_k] + 1$) != $j$:
                    $f2ind[f_k]$+=1 ;   // If something was aligned to $f_k$, check the last target step saved. Only increment it if it's not the same
            index=$f2ind[f_k]$
            if $\tilde{y}_j$ is not replaced:
                $\tilde{y}_j \rightarrow f_k^{index}$
            else:
                $\tilde{y}_j$+= $f_k^{index}$
Outputs: $P^{tr}(Y) = \tilde{Y}$
```

Algorithm 3: (Step 3.2) Replace characters in X and Y with indexed generation type symbols using salient alignments to X

```
Inputs: $X\_pos\_map$, $P^{tr}(Y)$
Init: $c_{index} = 1$; $g_{index} = 1$; $\tilde{X}$=copy($X$);
  $\tilde{Y} = P^{dec}(Y)$
for $X_i$ in $X$:
    if $X_i$ in $X\_pos\_map$:
        $c(X_i) = X\_op\_map[X_i][c]$
        $g(X_i) = X\_op\_map[X_i][g]$
        if $c(X_i) == [j]$ and $x_i == y_j$ and
          $g(X_i) == \emptyset$: ;   // $X_i$ is 1-to-1 copy
            if $X_{i-1}$ is not adjacent 1-to-1 copy:
                $c_{index} + = 1$
        $\tilde{x}_i$ = $C^{c_{index}}$; $\tilde{y}_j$ = $C^{c_{index}}$
    else:
        if $c(X_i)! = \emptyset$ and $g(X_i)! = \emptyset$:
            $mask =$‘’; $full\_index = []$
            for $k$ in $c(X_i)$:
                $c_{index}$+=1
                add $c_{index}$ to $full\_index$
                if $\tilde{y}_k$ is not replaced:
                    $\tilde{y}_k \rightarrow C^{c\_index}$
                else:
                    $\tilde{y}_k$+= $C^{c\_index}$
            $mask$+=‘$C^{full\_index}$’
            $full\_index = []$
            for $k$ in $g(X_i)$:
                $g_{index}$+=1
                add $g_{index}$ to $full\_index$
                if $\tilde{y}_k$ is not replaced:
                    $\tilde{y}_k \rightarrow G^{g\_index}$
                else:
                    $\tilde{y}_k$+=$G^{g\_index}$
            $mask$+=‘$G^{full\_index}$’
            $\tilde{x}_j \rightarrow mask$
Outputs: $P^{tr}(X) = \tilde{X}$, $P^{tr}(Y) = \tilde{Y}$
```

B Self-Att^{sub} Lemma Patterns

<table>
<tr><th>Query</th><th>No. of/Acc</th><th>Patterns</th></tr>
<tr><td>gold target=*scono & MSD=msd_it</td><td>23/1.00</td><td>*re: 9/1.0 (in| z| o| ti| chi| re)
*ire: 7/1.0 (s| col| or| ire)
*ir: 6/1.0 (re| in| ser| ir| si)
cir: 1/1.0 (in| fer| o| cir |si)</td></tr>
<tr><td>gold target=*ano & MSD=msd_it</td><td>189/1.00</td><td>*are: 149/1.0 (z| am| pic| are)
*arsi: 26//1.0 (im| pa| per| arsi)
car:3/1.0 (ri|mb|ec|car|si)
*izzarsi:2/1.0 (dest|abil|izz|arsi)
*iarsi:2/1.0 (di|lan|i|arsi)
par:2/1.0 (dis|col|par|si)
*ciarsi:1/1.0 (au|to|den|un|ci|arsi)
mar:1/1.0 (in|for|mar|si)
*rarsi:1/1.0 (gi|ost|r|arsi)
*itarsi:1/1.0 (ri|abil|it|arsi)
quar:1/1.0 (sci|ac|quar|si)</td></tr>
<tr><td>gold target=*ono & !(*scono) & MSD=msd_it</td><td>41/0.95</td><td>*ere: 18/0.95 (ri| otten| ere)
*dere: 10/1.0 (te| le| ve| dere)
ger: 3/1.0 (cos| par| ger| si)
*re:3/1.0 (servi|re)
*e:1/0.0 (ri|ro|m|per|e)
ir:1/1.0 (1908:s|ent|ir|si)
*si:1/1.0 (es|p|or|si)
*ire:1/1.0 (ri|di|ven|ire)
er:1/1.0 (r|aggi|ung|er|si)
mer:1/1.0 (ass|u|mer|si)</td></tr>
</table>

Table 6: Italian Self-Att^{sub} Patterns. MSD query *msd_it* is *V;IND;PRS;3;PL*. Number of examples (*No of*) and accuracy (*Acc*) are shown per selection with query and per group pattern. For each query, we list all extracted lemma patterns (sorted by frequency in a decreasing order) along with one segmented lemma example (in parentheses) mapped to the pattern.

<table>
<tr><th>Query</th><th>No. of/Acc</th><th>Patterns</th></tr>
<tr><td>gold target=*koot & MSD=msd_fin</td><td>37/0.97</td><td>*aa:8/1.0 (kar|sa |st |aa)
*ua:5/1.0 (ku |or |ett |ua)
id:5/1.0 (pro |mo |vo |id |a)
*a:4/1.0 (pu |r |je |hti |a)
*ta:4/1.0 (sk |r |uud |a |ta)
*taa:4/1.0 (jo |kel |taa)
*illa:2/1.0 (aal |to |illa)
*ella:2/0.5 (n |ar |a |hd |ella)
*ttaa:1/1.0 (ha |h |mo |ttaa)
*sia:1/1.0 (har |sia),
*ista:1/1.0 (li |i |pa |ista)</td></tr>
<tr><td>gold target=*kööt & MSD=msd_fin</td><td>9/1.00</td><td>*ä:3/1.0 (v |et |ele |hti |ä)
*ää:3/1.0 (jä |n |ist |ää)
*tä:2/1.0 (kä |pä |tä)
*tää:1/1.0 (hy |mä |h |ää)</td></tr>
</table>

Table 7: Finnish Self-Att^{sub} Patterns. MSD query *msd_fin* is *V;ACT;PRS;POS;IMP;3;PL*. Number of examples (*No of*) and accuracy (*Acc*) are shown per selection with query and per group pattern. For each query, we list all extracted lemma patterns (sorted by frequency in a decreasing order) along with one segmented lemma example (in parentheses) mapped to the pattern.

Expanding, Retrieving and Infilling: Diversifying Cross-Domain Question Generation with Flexible Templates

Xiaojing Yu
Texas A&M University
`vicky_yu@tamu.edu`

Anxiao (Andrew) Jiang
Texas A&M University
`ajiang@cse.tamu.edu`

Abstract

Sequence-to-sequence based models have recently shown promising results in generating high-quality questions. However, these models are also known to have main drawbacks such as lack of diversity and bad sentence structures. In this paper, we focus on question generation over SQL database and propose a novel framework by *expanding*, *retrieving*, and *infilling* that first incorporates flexible templates with a neural-based model to generate diverse expressions of questions with guidance of sentence structure. Furthermore, a new activation/deactivation mechanism is proposed for template-based sequence-to-sequence generation, which learns to discriminate template patterns and content patterns, thus further improves generation quality. We conduct experiments on two large-scale cross-domain datasets. The experiments show that the superiority of our question generation method in producing more diverse questions while maintaining high quality and consistency under both automatic evaluation and human evaluation.

1 Introduction

With a growing demand for natural language interfaces for databases, automatic question generation from structured query language(SQL) query has been of special interest (Xu et al., 2018). Recently, diversity-aware question generation has shown its effectiveness in improving down-stream applications such as semantic parsing and question answering tasks (Guo et al., 2018; Sultan et al., 2020). Although neural sequence-to-sequence based generation has been a dominant approach and is able to produce a meaningful description for SQL queries, existing methods still suffer from the lack of diversity as well as bad sentence structures (Gao et al., 2019).

In the neural-based approaches, conventional ways of generating diverse sentences focus on approximate decoding techniques such as beam search (Li et al., 2016a,b; Iyyer et al., 2018) and temperature sweep (Caccia et al., 2018). Those decoding strategies generate diverse samples while sacrificing the quality of sentences. Variational auto-encoders (VAEs) have been used to generate various sentences by applying additional information as latent variables (Hu et al., 2017; Guo et al., 2018; Chen et al., 2019; Shao et al., 2019; Ye et al., 2020). However, implicit latent representation provides limited controllability over sentence structure and can be difficult to adapt to a new domain. Paraphrase (Fader et al., 2013; Berant and Liang, 2014) and syntactic-based methods (Dhole and Manning, 2020) have also been studied. However, learning a paraphrasing model relies on a large number of domain-specific paraphrase pairs, which is difficult to obtain for target databases. Besides, syntactic-based approaches apply syntactic parsers or semantic rules to the natural language utterance, thus are not applicable to SQL-to-question generation.

In the rule-based generation systems, the templates work as essential prior knowledge that contains the structural information of sentences (Wang et al., 2015; Song and Zhao, 2016; Krishna and Iyyer, 2019). This ensures the generation contains fewer grammatical errors and performs better with extractive metrics (Wiseman et al., 2017; Puzikov and Gurevych, 2018). However, their template formats are mostly strict and the valid content for each chunk should be pre-defined, which makes large set of templates difficult to obtain.

In this paper, we propose a novel method that incorporates template-based generation with a neural sequence-to-sequence model for diversity-aware question generation. Instead of applying strict templates, we use flexible templates that can be collected efficiently with less expense. These flexible templates provide high-level guidance of sentence structure while also enable sufficient flexibility for a neural-based model to fill chunks with content de-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3202–3212
April 19 - 23, 2021.

tails. We present our method as a three-stage framework including *expanding*, *retrieving*, and *infilling*. In the *expanding* stage, we take advantage of existing large-scale cross-domain text-to-SQL datasets to extract and collect the flexible template set automatically. In the retrieving stage, given a SQL query, the best templates are retrieved from the collected template set by measuring the semantic distance of SQL and templates in a joint template-SQL semantic space. In the filling stage, we treat each template as a masked sequence and explicitly force the generator to learn question generation with the constraint of the template. In order to help the generator to discriminate template patterns and content patterns, a unique activate/deactivation mechanism is designed for the generator to learn when to switch between template-copying state and content-filling state.

We conduct experiments on two large-scale cross-domain text-to-SQL datasets. Compared to existing approaches, our method achieves the best diversity result for both datasets with both automatic evaluation and human evaluation, while maintaining competitive quality and high consistency with SQL queries. We further demonstrate that the designed modules each contribute to a performance gain through an ablation study.

2 Related Work

2.1 Diverse Text Generation

In order to generate diverse expressions automatically, paraphrase-based methods (Qian et al., 2019; Fader et al., 2013; Berant and Liang, 2014; Dong et al., 2017; Su and Yan, 2017) have been studied. Wang et al. (2015) proposes to iteratively expand the template set and lexicons given a small number of template seeds and a large paraphrase corpus. Syntactic-based generation (Iyyer et al., 2018; Dhole and Manning, 2020) processes the given text with natural language processing techniques to produce high-quality and diverse sentences with pre-defined templates. However, the methods are not designed to deal with SQL queries and noisy table content. In recent years, neural network-based models have been widely used in text generation (Pan et al., 2019). Many studies attempt to diversify text generation by tuning latent variables of different properties, such as topic, style, and content (Fang et al., 2019; Ficler and Goldberg, 2017; Shen et al., 2019), while our method focuses on explicitly changing the sentence structure. In exemplar-based systems (Cao et al., 2018; Peng et al., 2019; Chen et al., 2019), the exemplar works as a soft constraint to guide the sequence-to-sequence generation and realize controllable diverse generation. Wiseman et al. (2018) proposes to learn a hidden semi-Markov model decoder for template-based generation for knowledge records. Most existing work requires either paraphrase pairs of the same input, reference sentences of similar content, or work effectively only in a single domain. Unlike existing works, our method only takes advantage of the large-scale cross-domain SQL-to-text datasets to collect a large number of templates. We extract templates from the datasets directly to maintain the quality of the templates. In order to find proper templates for a given SQL query, we learn a joint semantic space by instance learning and retrieve the best templates with closest semantic distance.

2.2 Question Generation

The question generation task relates to many applications such as question generation over knowledge base records (Wang et al., 2015), data-to-text generation (Wiseman et al., 2017) and question generation for question answering(QA) systems (Tang et al., 2017; Sun et al., 2019). SQL-to-question task differs from the other tasks in that SQL queries typically include new entities over different databases, which makes cross-domain generation a significant challenge. Xu et al. (2018) explores the graph-structured information in a SQL query and proposes a graph-to-sequence approach for generation. Guo et al. (2018) proposes to apply a copy mechanism and latent variables to map low-frequency entities from SQL queries to questions and generate diverse questions in an uncontrolled way. Existing SQL-to-question approaches aim at generating high-quality questions, while the diversity of the generation is less explored. In this work, we focus on generating diversified questions with the guidance of templates from cross-domain datasets.

3 Problem Formulation

Given a SQL query as the input sequence, question generation over database aims to generate a natural language question as an output sequence that accurately reflects the same meaning as the given SQL query. In this work, we generate the question by introducing an intermediate template in the generation process. Therefore, by applying

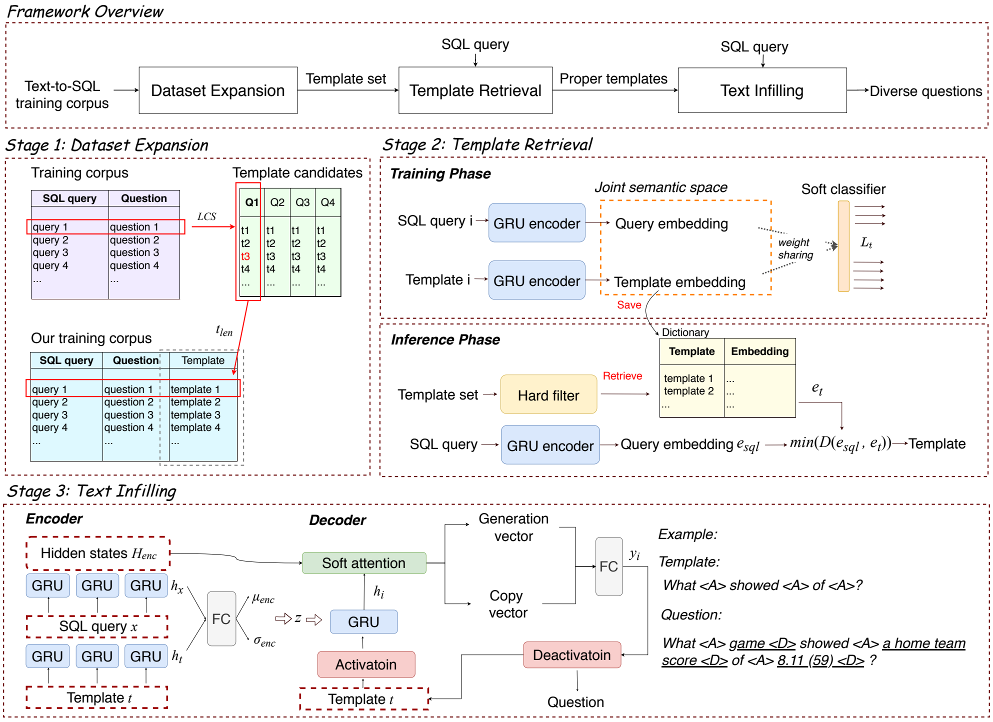


Figure 1: Architecture of our Framework.

different templates, we can generate diverse expressions of questions. Let $x = [x_1, x_2, ..., x_{|x|}]$ denote the given SQL query, $y = [y_1, y_2, ..., y_{|y|}]$ denote the gold-standard question, and $t = [t_1, t_2, ..., t_{|t|}]$ denote the corresponding template. Given a neural-based system with a set of learnable parameters θ_t^* and θ_q^*, the two-stage objective in this work can be formulated as follows:

$$\theta_t^* = \arg\max_{\theta_t} P(t|x)$$

$$\theta_q^* = \arg\max_{\theta_q} P(y|x,t)$$

4 Methodology

In this section, we introduce our framework, which learns question generation from SQL queries with the guidance of various templates, so as to increase the diversity of generation.

4.1 Framework Overview

We illustrate the framework that models the generation process in Figure 1. In brief, it includes three main stages: *expanding*, *retrieving*, and *infilling*.

Dataset Expansion The purpose of this step is to acquire a training dataset consisting of triplets `<query, question, template>`. Previous methods usually require a large corpus containing paraphrased questions to learn template structures; meanwhile, for existing text-to-SQL datasets, only `<query, question>` pairs are provided instead. To tackle those challenges, we design the longest common subsequence (**LCS**) based algorithm to automatically extract templates for each `<query, question>` pair without requiring the paraphrase pairs. Details of the algorithm are introduced in Section 4.2.

Template Retrieval After obtaining the *expanded training set*, all templates are gathered to form a large *template set* to generate diverse questions. To improve the quality and rationality of the generation, it is essential to retrieve suitable templates for it. A proper template should be consistent with the content information in a specific SQL query. For example, when the given query is `SELECT Population WHERE ( City = New York)`, the template `When is the <ph> of <ph> ?` should not be selected. For that purpose, we propose a soft classifier to learn a *joint SQL-template space*. In this way, the semantic distance can be measured between the two modalities, so that we can select proper templates by the closest semantic distance. Besides, since templates show higher inter-class similarity with the same SQL pattern, we also apply a *hard filter* to exclude the templates paired with different SQL patterns. Details of the soft classifier are introduced in Section 4.3.

Text Infilling With a encoder-decoder model based on gated recurrent unit (GRU), we conduct question generation in the way of text infilling. The query x and template t are encoded into vectors separately by the bi-directional GRU encoder (Cho et al., 2014). Following the work of Gu et al. (2016) and Guo et al. (2018), we leverage the soft-attention and copy mechanism in the decoder construction. A Gaussian latent variable is adopted to capture the query and template variations. In the decoding process, a template t is fed into the decoder as a supervised signal to generate questions dynamically and sequentially. We propose an activation/deactivation mechanism to enforce the decoder to differentiate between the template patterns and the content patterns instead of randomly masking slots. In this way, the decoder can learn when to switch between the template reading and the content filling during generation. Details of the activation/deactivation mechanism are illustrated in Section 4.4.

4.2 Flexible Template as LCS

Consider a SQL query as a combination of a SQL pattern (i.e., the query with its content words removed) and the table information as follows:

```
SELECT COUNT( PLAYER ) WHERE
      (STATE = 'Texas')
```

where the underlined tokens are content from the table and `SELECT COUNT(<ph>) WHERE (<ph> = <ph>)` is a typical SQL pattern. Similarly, questions are composed of content words and template words. Since template words are often reused more frequently than content words, we design an effective method to extract most template words for each question in the training set. For the i-th question Q_i, we record its longest common sub-sequence (LCS) with each other question as a candidate template and construct a candidate-template dictionary d_i. The keys in d_i are the candidate sequences, and the corresponding values are the lengths of the sequences. After that, we choose the longest candidate from d_i as the template for the i-th question. The pseudo-code is described in Algorithm 1.

The candidate templates should satisfy the following rules:

- Each template should appear over 20 times.
- Each template should includes at least one of the keywords: *where, what, which, when, why, who, how, name, tell* .

When applying LCS, we mark the possible positions for content insertion between template words by placeholder `<ph>`, and format the templates like this instance: `Which <ph> has the largest <ph> ?` By ignoring the lengths of content word sequences, the templates become more flexible and can adapt to more scenarios.

Algorithm 1 LCS-based Template Extraction

```
Input: question set $Q = [q_1, q_2 ..., q_M]$; keyword set $W$
Output: Template set: $T_{len}$
for all $q_i \in Q$ do
    Initialize dictionary $d_i$
    for all $q_j \in Q$ do
        $c = LCS(q_i, q_j)$
        if $c \bigcap W \neq \emptyset$ then
            if $c \notin d_i.keys$ then
                $d_i[c] = 0$
            end if
            $d_i[c] += 1$
            Record position index for content
        end if
    end for
    for all $c \in d_i.keys$ do
        if $d_i[c] < 20$ then
            delete $d_i[c]$ from $d_i$
        end if
    end for
    $t_{len} = \arg\max_c(length(d_i.keys))$
    Update $T_{len}$ by adding $t_{len}$
end for
return $T_{len}$
```

4.3 Learning Joint SQL-Template Space

As a sub-sequence of a question, a template should be close to the query in the semantic space if it is from the corresponding question, and be far away from the query if it is from an irrelevant question. Based on this intuition, we propose a soft classifier to learn a joint SQL-template space.

Soft Classifier. Classification models have been widely used in visual/textual applications. In a retrieval task, the classification model can learn feature embedding for the input, and its best matching counterpart can be found from a database by measuring the cosine distance between their embeddings. Inspired by this, we consider every `<SQL query, template>` pair in training set S as a distinct class, and learn the feature embeddings by instance-level classification. We represent each `<SQL query, template,`

`class>` triplet by $< x, t, n >$. Considering SQL queries and templates as objects constructed with different syntax, we encode them with two separate GRU encoders:

$$e_x = GRU_x(x)$$
$$e_t = GRU_t(t)$$

where e_x, e_t are query embedding and template embedding, respectively. In order to map SQL queries and templates to a joint feature space, we add a share-weight fully-connected layer W_s with softmax as the final classifier. The predicted probabilities over all instances are calculated as follows:

$$P(\cdot|x) = Softmax(W_s^T tanh(e_x))$$
$$P(\cdot|t) = Softmax(W_s^T tanh(e_t))$$

We jointly train the encoder and the classification layer with the following loss function:

$$\mathcal{L}_t = -\sum_{(x,t,n)\in S} (logP(n|x) + logP(n|t))$$

Inference. To enable efficient retrieval of templates, we store the template embeddings in a dictionary. In the inference phase, we can feed any SQL query into GRU_x to produce the query embedding, then remove the improper templates by a hard filter. We calculate the cosine distances between the query and the remaining templates, and sort them in descending order. The top-k nearest templates are selected for question generation.

$$\mathcal{D}(e_x, e_t) = \frac{e_x}{\|e_x\|_2} \times \frac{e_t}{\|e_t\|_2}$$

Avoid Overfitting. Since each class includes only one instance, we cannot use the classification loss of the validation set V to detect overfitting. Instead, we detect the overfitting by computing the average rank error R for the validation set:

$$R = \frac{1}{|V|^2}\sum_{i=1}^{|V|}(|r_i^a - r_i^p|)$$

where r^p and r^a are the predicted rank and actual rank, respectively. We stop training when R keeps increasing.

4.4 Decoding with A/D mechanism

The key idea of the proposed decoding method is that the generation process can be decomposed into a series of sub-generation tasks that are spaced by tokens in the template. During each sub-generation, the model can generate tokens of variable lengths. Since the decoder generates text word-by-word, it should determine where to switch between a content-filling state and a template-copying state, namely the **activation** or **deactivation** state, respectively. To achieve this, we require the decoder to activate/deactivate (A/D) generation with special switch symbols `<A>` and `<D>`. Therefore, when the decoder generates a symbol `<A>`, it changes the template-copying state to the content-filling state; when the decoder generates a `<D>`, it terminates the content-filling state and switches to the template-copying state. We also set a maximum length for each sub-generation to avoid the generation of unbounded sequences. In practice, before we train the question generator, we rewrite the question and template as follows (as an instance):

Template: `<BEG> Which <A> has the largest <A> ? <END>`

Question: `<BEG> Which <A> one <D> has the largest <A> population among U.S. cities <D> ? <END>`

We apply a pointer p to point to the current template token. With a simple GRU decoder, the state s and the generated token $\widehat{y}_i$ at the i-th step can be determined as follows:

$$s_i = \begin{cases} 1, \ \widehat{y}_{i-1} =< A >, \\ 0, \ \widehat{y}_{i-1} =< D >, \\ s_{i-1}; otherwise. \end{cases}$$

$$\widehat{y}_i = \begin{cases} Softmax(GRU(\widehat{y}_{i-1}, h_{i-1}^{dec})), \ s_i = 1, \\ t'_p, \ s_i = 0. \end{cases}$$

where h_{i-1}^{dec} is the hidden state of the $(i-1)$-th step, and t'_p is the current token of the template-copy operation, which should be updated after each copy. $\widehat{y} = [\widehat{y}_1, \widehat{y}_2, ...\widehat{y}_{|y|}]$ is the generated sequence. y' and t' are the rewritten question and template[1]. We apply teacher forcing during training, and feed the rewritten question y' as input to the decoder. During inference, we feed the rewritten template t' as input to the decoder. The training objective for generator G_q can be factorize as follows:

$$\max_{\theta_t} E_{<x,y>\sim D}[P_{\theta_q}(y'|x, t')]$$
$$= \max_{\theta_t} E_{<x,y'>\sim D}[\prod_{i\in 1:N} P_{\theta_q}(y'_i|x, t', y'_{i-1})]$$

---

[1] Code implementation is available at https://github.com/xiaojingyu92/ERIQG

In order to further diversify the expression of content, we introduce a latent variable z to the model. z relies on both SQL query x and template t'. We make Q a posterior distribution of z given x, t'. Then the evidence lower bound(ELBO) loss for it is:

$$\mathcal{L}_q = -E_{z \sim Q}(\sum_{i=1:N} log(P_{\theta_q}(y_i|x, t', z, y_{1:i-1})))$$
$$+ D_{KL}(Q(z|x, t')||P_{\theta_q}(z))$$

where $Q_q(z|x, t') \sim N(\mu, \sigma)$. We apply a re-parameterization trick, making $z \sim N(0, I)$ and μ, σ learnable deterministic functions.

Note that the function of the template and latent variable do not overlap with each other. A large number of templates ensures the diverse sentence structure in generated questions, while the latent variable produces various expression of the content in the slots. As a part of the sentence, the `<A>` and `<D>` symbols join the back-propagation computation for optimizing the decoder's parameters. They work as additional information that guides the decoder to discriminate templates and content patterns, and learn when to terminate infilling and switch to template-copying state at each slot.

5 Experiment

Dataset To validate our framework's capability in generating diverse and controllable questions, we conduct experiments on two large-scale cross-domain text-to-SQL datasets: WikiSQL (Zhong et al., 2017) and Spider (Yu et al., 2018). WikiSQL contains 80654 query-question pairs derived from 24241 different schemas, with both validation set and test set released. Spider contains 10181 query-question pairs in 138 domains, with the validation set published. We follow the provided split settings for training and testing.

Setup We first construct the template set from the training set using our expansion method. For each SQL query in the test set, we retrieve the $k-1$ most relevant templates from the template set to generate $k-1$ different questions. We also include one question with template `<BEG> <A> <END>` to evaluate the model's capability in generating a complete sentence. Thus k questions for each SQL query are provided for evaluation. For our experiments, we set $k = 5$.

Baselines We compare our model(**ERI**)'s performance to models based on the baseline approach **QG** (Guo et al., 2018) with different diverse-sentence generation strategies. (1) **Latent Variable(QGLV):** Guo et al. (2018) applies a sequence-to-sequance network with a copy mechanism for question generation from SQL. A latent variable is introduced to generate diverse questions. (2) **Temperature Sweep(TEMPS):** We apply temperature sweep (**ts**) (Caccia et al., 2018) for decoding in **QG**. (3) **Beam Search(BEAMS):** We further combine **QG** with beam search (Li et al., 2016b) to generate diverse questions. In practice, we set the beam width to 5 and obtain sentences with top-5 highest probabilities for comparison. A Boltzmann temperature parameter α is applied to modulate the entropy of the generator. In practice, we set α = 0.7 as suggested and obtain 5 generated sentences for each query.

Evaluation Metrics We adopt the following automatic metrics to measure the quality of generated questions. (1) **maxBLEU:** The max BLEU-4 score among 5 generated questions. (2) **Coverage:** (Shao et al., 2019) This metrics measures the average proportion of input query covered by the generated questions. (3) **ParseAcc:** We use neural semantic parsers SQLova (Hwang et al., 2019) and Global-GNN (Bogin et al., 2019) to parse WikiSQL and Spider respectively. The semantic parsers translate the generated question into SQL query and calculate the exact-match accuracy with the input SQL query as the ground-truth. A higher accuracy score means the generated questions are more natural and consistent with the given SQL queries.

We adopt the following automatic metrics to measure the diversity of generated questions. (1) **self-BLEU:** (Zhu et al., 2018) The average BLEU-4 of all pairs among the k questions). (2) **self-WER:** (Goyal and Durrett, 2020) The average word error rate of all pairs among the k questions. A lower self-BLEU score and a higher self-WER score indicate more diversity in the result. (3) **Distinct-4:** (Li et al., 2016a) It measures the ratio of distinct n-grams in generated questions.

5.1 Automatic Evaluation

The automatic evaluation results are reported in Table 1 and 2. Our model ERI outperforms all baseline models in terms of the three diversity metrics for both datasets, except the self-BLEU on Spider. This shows the effectiveness of ERI in improving the diversity of its generation. Note that TEMPS performs less favorably in terms of qual-

Models	Quality			Diversity		
	Coverage ↑	ParseAcc ↑	maxBLEU ↑	Self-BLEU ↓	Self-WER ↑	Distinct-4 ↑
QGLV	70.50	**73.89**	37.75	92.86	17.39	33.46
TEMPS	11.34	3.38	5.36	84.50	36.49	59.34
BEAMS	71.49	68.09	**42.17**	79.80	37.39	54.97
ERIT(ours.)	**72.44**	72.79	28.43	**56.30**	**67.00**	**78.73**
w/o lv	70.28	69.53	24.30	57.96	64.39	75.31

Table 1: Automatic evaluation for diverse question generation on WikiSQL. **w/o lv** refers to our model without incorporating the latent variable.

Models	Quality			Diversity		
	Coverage ↑	ParseAcc ↑	maxBLEU ↑	Self-BLEU ↓	Self-WER ↑	Distinct-4 ↑
QGLV	13.76	3.97	14.45	96.90	11.73	38.16
TEMPS	6.58	3.87	4.00	**59.25**	7.59	19.73
BEAMS	13.68	4.16	**15.68**	89.39	21.12	50.17
ERI(ours.)	**15.89**	**18.09**	14.42	67.41	**53.23**	**66.96**
w/o lv	12.67	16.70	12.62	62.25	55.58	64.51

Table 2: Automatic evaluation for diverse question generation on Spider. **w/o lv** refers to our model without incorporating the latent variable.

Models	Fluency ↑	Consistency ↑	Diversity
QGLV	**4.56**	**4.64**	1.63
TEMPS	1.13	1.31	1.81
BEAMS	4.16	4.25	2.34
ERI	**4.56**	3.68	**4.31**

Table 3: Human evaluation results.

ity and diversity, as the temperature parameter α is a sensitive factor for the generation and needs tuning further. The expanded template set provides various valid sentence structures for expressing the same question which significantly contributes to the diversity expressions. But it also decreases the word-level overlapping, which leads to a relatively low maxBLEU score on WikiSQL. However, BLEU score may not be a suitable measurement for diversity-aware generation (Su et al., 2020; Shao et al., 2019). As the BLEU calculates the overlapping of n-grams, it does not necessarily reflect the quality of template-based generation. We illustrate that by an example of our model in Table 4. Although the BLEU scores are low, the generated questions are fluent and consistent with the SQL query with diversified structures.

To further measure the question quality and consistency in the semantic parsing task, we calculate the ParseAcc score. Our model performs competitively with QGLV on WikiSQL and shows a substantial improvement on Spider. The ParseAcc scores with ground-truth questions are 81.60% and 65.96% on WikiSQL and Spider, respectively. Our model also outperforms the baselines in the coverage of field names and values in the SQL query, indicating that essential terms from input are learnt and translated to questions. We also show the result of our model without the latent variable. In this setting, diversity of generated questions solely depends on selected templates. Without the latent variable, the proposed framework still outperforms the baselines in diversity metrics while maintains a good quality, which also supports that template contributes the most to the diversity of generation.

5.2 Manual Evaluation

To evaluate the quality of the generation, we run a manual evaluation to measure the quality and diversity for 800 SQL-question pairs from WikiSQL test set, produced by baseline models and our model (with $k = 5$). Each rater gives a 5-point scale for each SQL-question pair regarding the (1) **Fluency:** grammatical correctness, (2) **Consistency:** the semantic alignment with the corresponding SQL query, and (3) **Diversity:** the diverse expression of the generated questions.

We employ Fleiss' Kappa for inter-rater reliability measurement. The Kappa scores are 0.77, 0.60, 0.75 for fluency, consistency, and diversity, respectively, which indicates a good agreement on the scores. The results are presented in Table 3. Our model outperforms the baseline models in diversity and fluency. It can achieve the best trade-off for the three measurements. Although QGLV and BEAMS show the best performance in generating

SQL query	SELECT COUNT (rd #) WHERE pick # $<$ 5
Ground-truth	how many rounds exist for picks under 5 ?
Q1 (BLEU=2.79) :	**what is the number of** rd **when the** pick number **is less than** 5 **?**
Q2 (BLEU=11.73):	**how many** rounds **have a** pick # **less than** 5 **?**
Q3 (BLEU=2.79) :	**what is the total number of** rd **where the** pick **is less than** 5 **?**
Q4 (BLEU=2.02) :	**tell me the total number of** rd **for** pick **less than** 5
Q5 (BLEU=1.51) :	**for the** pick **less than** 5 **, what was the total number of** rd # **?**

Table 4: Examples of high-quality generation with low BLEU score. Template tokens are in bold font.

Models	BLEU↑	NIST↑	ROUGH↑	METEOR↑
QGLV	32.19	4.74	64.02	64.25
***Graph2Seq**	38.97	-	-	-
ERI	**48.12**	**5.24**	**76.52**	**75.76**
w/o A/D	43.30	4.85	72.41	73.76
w T	31.42	4.18	63.44	63.94
w/o T	29.76	4.05	62.37	63.17

Table 5: Performance on different sub-module combinations on WikiSQL. *: the value is cited from Xu et al. (2018)

Models	BLEU↑	NIST↑	ROUGH↑	METEOR↑
QGLV	12.60	2.39	44.37	38.74
ERI	**21.30**	**3.11**	**53.83**	**51.04**
w/o A/D	19.02	2.93	52.45	50.41
w T	12.40	2.41	45.8	39.99
w/o T	13.36	2.35	45.05	40.27

Table 6: Performance on different sub-module combinations on Spider.

Models	WikiSQL Dev	WikiSQL Test	Spider Dev
Random	0.1	0.07	0.80
Hard Filter	16.7	13.4	30.8
Ours.	**21.1**	**14.0**	**32.6**

Table 7: MAP results of template retrieval methods.

SQL query	SELECT attendance WHERE week $<$ 16 and date = bye
Ground-truth	what is the attendance for a week earlier than 16 , and a date of bye ?
ERI(ours.):	
Q1:	**what is** attendance , **when** week is **less than** 16 **, and when** date is bye ?
Q2:	**which** attendance **has a** week **smaller than** 16 **and a** date **of** bye ?
Q3:	**what is the** attendance **for the** week **earlier than** 16 **and is** dated bye ?
Q4:	attendance **before** week 16 **on what** bye **was the** date ?
Q5:	how many people attended the game before week 16 on bye ?
QGLV:	
Q1:	what was the attendance for the bye game before week 16 ?
Q2 (= **Q3**,**Q4**,**Q5:**)	what is the attendance for the bye game before week 16 ?
BEAMS:	
Q1:	what was the attendance for the bye week before week 16 ?
Q2:	which attendance has a week smaller than 16 and a date of bye ?
Q3:	which attendance has a week smaller than 16 , and an date of bye ?
Q4:	what is the attendance of the game with a week less than 16 and bye ?
Q5:	what is the attendance of the game with a week less than 16 and a bye date ?
TEMPS:	
Q1:	where week date
Q2:	where week date
Q3:	where week week week week bye and week
Q4:	where week and week date bye and attendance where week
Q5:	where week bye and week attendance

Table 8: Question generation example on WikiSQL.

high-quality sentences, they tend to create questions of fixed structures with only minor changes in expressions. With our method, the templates provide substantial changes in sentences' structures, which validates the benefit of the proposed template-extraction method. Examples from our model and the baselines are given in Table 8.

5.3 Ablation Study

For the ablation study, we present experimental results to verify performance in two aspects: (1) whether the generator in our model benefits from learning with the activation/deactivation mechanism; (2) whether our model maintains the consistency by selecting suitable templates in the joint semantic space.

Evaluation on Generator To analyze the ability of generating high-quality questions from the given templates, we extract the templates from the test set and use the corresponding template to guide the question generation. We measure the generation quality by **BLEU**, **NIST**, **ROUGE**, and **METEOR**. In order to analyze the impact of various modules in our generator, we evaluate the following versions of our framework: (1) **ERI w/o T** model that does not use templates for encoding and decoding. (2) **ERI w T** model where templates are encoded but not used as the input for decoding. (3) **ERI w/o A/D** model that applies the generator with the decoding strategy similar to a seq2seq model in Zhu et al. (2019). It treats each slot as a segment, and predicts each segment as an independent sentence. (4) **ERI** model that has all designed modules, including the A/D mechanism. The results are presented in Table 5 and 6. Our model outperforms existing methods in four metrics. Using template information improves our model on both datasets. The segmented-based infilling method gains further improvement, which shows the effectiveness of providing templates as hard constraints. By introducing the A/D mechanism to decoding, our model has seen further boosts, which demonstrates that the A/D mechanism enhances the learning of generation from the templates.

SQL-Template Consistency To validate the im-

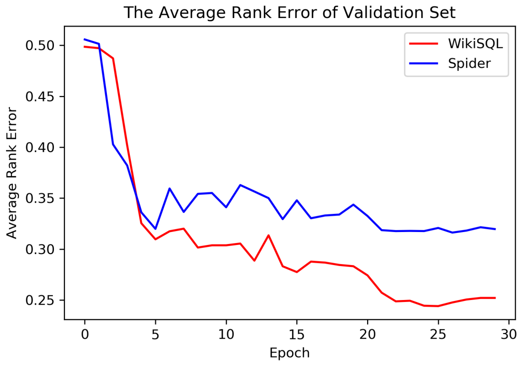


Figure 2: Average rank error in training phase.

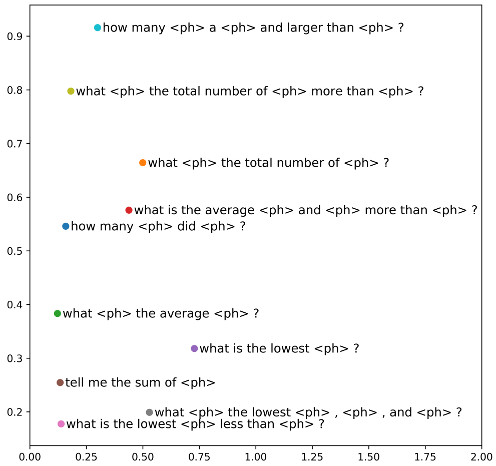


Figure 3: Visualization of the template features by tSNE.

pact of our template retrieval method, we show the average rank error of soft classifier during the training phase in Figure 2. For both datasets, the rank error decreases during training, which indicates the soft classifier can capture the semantic relation between SQL queries and templates. To observe if the soft classifier affects the performance by selecting the proper template, we also compare our 2-stage template retrieval to random strategies and hard filter in mean average precision (MAP). The result is presented in Table 7. Compared to hard filter, the soft classifier improves the MAP by 4.4%, which validates the effectiveness of the proposed template retrieval method.

Visualization of Joint Space To visualize the similarity of templates, we map feature samples to 2-dimensional space by t-Distributed Stochastic Neighbor Embedding(t-SNE) in Figure 3. The features from similar SQL-template pairs preserve closer distances, which shows the effectiveness of our instance-level classification in learning the semantic meaning in the joint feature space.

6 Conclusion

In this paper, we present a novel framework for question generation over SQL database to produce more diversified questions by manipulating the templates. We expand the template set from cross-domain SQL-to-text datasets, and retrieve proper templates from a template set by measuring the distance between the templates and the SQL query in a joint semantic space. We propose an activation/deactivation mechanism to make full use of templates to guide the question generation process. Experimental results have shown that the presented model can generate various questions while maintains their high quality. The model has also improved the matching between the templates and the content information of SQL queries.

References

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019. Global reasoning over database structures for text-to-sql parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3650–3655.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. *arXiv preprint arXiv:1811.02549*.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger

Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Kaustubh Dhole and Christopher D Manning. 2020. Syn-qg: Syntactic and shallow semantic rules for question generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 752–765.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618.

Le Fang, Chunyuan Li, Jianfeng Gao, Wen Dong, and Changyou Chen. 2019. Implicit deep latent variable models for text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3937–3947.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, Guodong Zhou, and Shuming Shi. 2019. A discrete cvae for response generation on short-text conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1898–1908.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

Daya Guo, Yibo Sun, Duyu Tang, Nan Duan, Jian Yin, Hong Chi, James Cao, Peng Chen, and Ming Zhou. 2018. Question generation from sql queries improves neural semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1597–1607.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885.

Kalpesh Krishna and Mohit Iyyer. 2019. Generating question-answer hierarchies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2321–2334.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016b. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949*.

Hao Peng, Ankur Parikh, Manaal Faruqui, Bhuwan Dhingra, and Dipanjan Das. 2019. Text generation with exemplar-based adaptive decoding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2555–2565.

Yevgeniy Puzikov and Iryna Gurevych. 2018. E2e nlg challenge: Neural models vs. templates. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 463–471.

Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. 2019. Exploring diverse expressions for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3164–3173.

Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, et al. 2019. Long and diverse text generation with planning-based hierarchical variational model. In *Proceedings of the 2019 Conference on Empirical*

Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3248–3259.

Xiaoyu Shen, Jun Suzuki, Kentaro Inui, Hui Su, Dietrich Klakow, and Satoshi Sekine. 2019. Select and attend: Towards controllable content selection in text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 579–590.

Linfeng Song and Lin Zhao. 2016. Domain-specific question generation from a knowledge base. *arXiv preprint arXiv*, 1610.

Hui Su, Xiaoyu Shen, Sanqiang Zhao, Zhou Xiao, Pengwei Hu, Cheng Niu, Jie Zhou, et al. 2020. Diversifying dialogue generation with non-conversational text. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7087–7097.

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1235–1246.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for qa. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5651–5656.

Yibo Sun, Duyu Tang, Nan Duan, Shujie Liu, Zhao Yan, Ming Zhou, Yuanhua Lv, Wenpeng Yin, Xiaocheng Feng, Bing Qin, et al. 2019. Joint learning of question answering and question generation. *IEEE Transactions on Knowledge and Data Engineering*.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187.

Kun Xu, Lingfei Wu, Zhiguo Wang, Yansong Feng, and Vadim Sheinin. 2018. Sql-to-text generation with graph-to-sequence model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 931–936.

Rong Ye, Wenxian Shi, Hao Zhou, Zhongyu Wei, and Lei Li. 2020. Variational template machine for data-to-text generation. *arXiv preprint arXiv:2002.01127*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Wanrong Zhu, Zhiting Hu, and Eric Xing. 2019. Text infilling. *arXiv preprint arXiv:1901.00158*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

Handling Out-Of-Vocabulary Problem in Hangeul Word Embeddings

Ohjoon Kwon[1†] **Dohyun Kim**[2†] **Soo-Ryeon Lee**[3] **Junyoung Choi**[3] **SangKeun Lee**[2,3]
[1]Samsung Research, Republic of Korea
[2]Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea
[3]Department of Artificial Intelligence, Korea University, Seoul, Republic of Korea
`ohjoon.kwon@samsung.com`
`{dhkim1028, bonny05616, dnduddl2015, yalphy}@korea.ac.kr`

Abstract

Word embedding is considered an essential factor in improving the performance of various Natural Language Processing (NLP) models. However, it is hardly applicable in real-world datasets as word embedding is generally studied with a well-refined corpus. Notably, in Hangeul (Korean writing system), which has a unique writing system, various kinds of Out-Of-Vocabulary (OOV) appear from typos. In this paper, we propose a robust Hangeul word embedding model against typos, while maintaining high performance. The proposed model utilizes a Convolutional Neural Network (CNN) architecture with a channel attention mechanism that learns to infer the original word embeddings. The model train with a dataset that consists of a mix of typos and correct words. To demonstrate the effectiveness of the proposed model, we conduct three kinds of intrinsic and extrinsic tasks. While the existing embedding models fail to maintain stable performance as the noise level increases, the proposed model shows stable performance.

1 Introduction

Word embedding refers to the process of generating vectors that contain semantic and syntactic information of languages. Machines can understand the meaning of words through the word embeddings. Several embedding methodologies have been proposed to improve various Natural Language Processing (NLP) models' performance, such as machine translation, sentence classification, and text generation (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017).

Word2vec (Mikolov et al., 2013) generates word vectors using the contextual data of words based on the distributional hypothesis. Word2vec assigns a unique vector to each word, and has proved its effectiveness by improving various systems' performance. Similarly, GloVe (Pennington et al., 2014) that uses co-occurrence information of each word found throughout the corpus, generates word vectors that contain semantic and syntactic characteristics.

However, Word2vec and GloVe suffer from being dependent on the training corpus. In particular, they do not include the word embeddings that did not appear in the training process. To tackle this OOV problem, replacing unseen words with a unique token "UNK" is widely used as a solution. This method can allow the model to deal with OOV words. However, this countermeasure reduces the performance of the system. This is because several unseen words are recognized as having the same meaning. Hence, the importance of each word is ignored and recognized as noise.

This OOV problem amplifies when dealing with real-world datasets that contain a large number of typos and intended slang that has not been included in the training phase. In particular, if words relevant to the core meaning of a sentence are replaced as OOV, the system's performance will drastically reduce. For example, suppose an evaluation set for a sentiment classification task contains typos such as "funnnny" and "coooool". When predicting the sentiment of the sentence, the original meanings of the two words, "funny" and "cool", serve as decisive information. However, the sentiment classification accuracy drastically reduces when these words are processed as "UNK" and encoded to have the same meaning. Therefore, to increase the effectiveness of NLP models, robust word embeddings are strongly needed in noisy text.

Meanwhile, FastText (Bojanowski et al., 2017) presents a method to address the OOV problem. FastText follows the training process of Word2vec,

---

[†] Equal contribution

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3213–3221
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

but understands a word as a set of n-gram subword information. For example, the meaning of the word 'funny' is learned by decomposing the characters into n-grams such as 'fun', 'unn', and 'nny.' In this way, FastText can generate embeddings for OOV words that did not appear during the training phase by utilizing this subword information.

FastText is suitable for languages that have characteristics of isolated words (e.g., English). This is because the meaning of the word can be inferred through the summation of subword information: prefix, root, and postfix. However, in the case of agglutinative languages such as Hangeul (Korean writing system), it is not easy to fully understand the meaning of a word by merely referring to the summation of the subword information. A word in Hangeul consists of at least one to ten or more different morphemes, which make extracting internal information complicated. Therefore, to decipher a Hangeul word, it is necessary to be sensitive when looking for the vital subword information within the word.

CNN captures locational and morphological information of words and characters, and many studies using CNN have achieved good performance in various NLP tasks (Zhang et al., 2015; Kim et al., 2016, 2018; Ma and Hovy, 2016). Inspired by this, in this paper, we propose a CNN-based channel attention word embedding model that generates robust Hangeul embeddings for noisy text. The proposed model utilizes CNN to extract n-gram pairs from Korean words. The model then applies an attention mechanism to features from each channel representing the inherent information in the n-gram pairs. An input word for training is replaced with intentionally generated typos with a certain probability of adapting to typos. This generated word vector enhances the semantics of a word by predicting the context words. In summary, the contributions of this paper are three-fold:

- We propose a robust word embedding model against typos by training it with intentionally generated typos.
- We introduce a channel attention mechanism to utilize the distinctive structure of Hangeul words.
- We demonstrate the effectiveness of the proposed model on the noisy text through word analogy, language modeling, and sentiment classification tasks.

The remainder of the paper is organized as follows. In Section 2, we investigate related work and discuss key differences from ours. In Section 3, we describe the proposed methodology in detail. Furthermore, we present the evaluation results and analyze the performance in Section 4 and 5, respectively. Finally, Section 6 contains the conclusion.

2 Related work

In this section, we review previous studies on 1) embedding generation methods that are robust to typos and 2) embedding generation methods that consider the unique characteristics of Hangeul.

2.1 Typos word embedding methods for English

There have been many studies for dealing with OOV problems (Bojanowski et al., 2017; Belinkov and Bisk, 2018; El Boukkouri et al., 2020). Unlike unseen words, typos require a different approach in that the originally intended word is likely to be known. RoVe (Malykh et al., 2018) and MOE (Piktus et al., 2019) explored the methodologies to deal with typos. RoVe (Malykh et al., 2018) utilized the English word's structural features as prefix, root, and postfix. The target word is encoded into Beginning, Middle, and End vectors. Embeddings generated in this way maintain robust model performance, although a noise level of the downstream task increases. However, RoVe performs poorly in generating agglutinative embeddings in languages such as Hangeul. As mentioned earlier, this is due to the feature of agglutinative languages where dozens of morphemes are combined to form a word. In other words, dividing morphemes into three parts – Beginning, Middle, and End – is not sufficient to comprehend Hangeul words.

MOE (Piktus et al., 2019) introduced supervised learning for misspelling patterns to FastText's training mechanism. The model was explicitly trained to infer the original meaning of the word from the irregular shape of typos by increasing the similarity between them. In this way, MOE was trained on typo data collected from real users on the web. Consequently, MOE showed stable performance for unseen words in intrinsic and extrinsic tasks. However, MOE generates the word embeddings with linear combinations of the constituent subwords. In other words, the word vectors do not reflect the differences in importance among the subword information. This is insufficient to fully

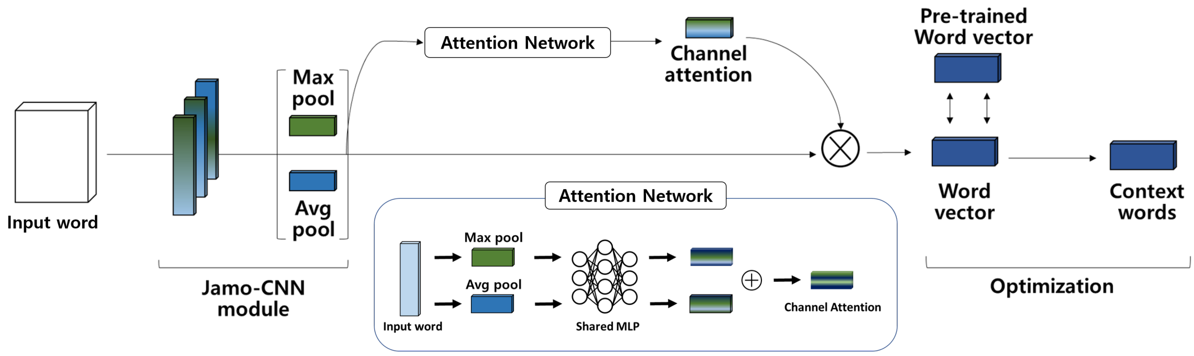


Figure 1: Overall framework for the word embedding generating system proposed in this paper.

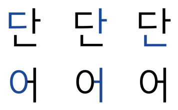


Figure 2: Example of the constituent of a Korean word ”단어 (word)”.

explain the meaning of words in complex agglutinative languages such as Hangeul. To generate effective Hangeul word embeddings, differentiating importance across n-gram pairs is necessary.

2.2 Word Embedding Methods for Korean

Ko-FastText (Park et al., 2018) and other morpheme-based studies (Lee et al., 2018; Nam and Kim, 2016) considered the characteristics of agglutinative languages during embedding. Ko-FastText performed better in intrinsic and extrinsic tasks with a Hangeul dataset by applying the FastText to each jamo (the smallest unit of a Hangeul word). However, this model did not fully utilize the sophisticated internal information of Hangeul words. Although the model refers to the morphological information revealed in the n-gram, lack of selectivity regarding the important morphological information of a word remains a problem.

Nam and Kim (2016) and Lee et al. (2018) explicitly considered the Korean words' morphological information. They decomposed words into character n-grams using morpheme analyzers. However, they encountered a fundamental problem that their models are dependent on the performance of the morpheme analyzer. In particular, the analyzer did not work appropriately for typos, so the performance of word embeddings cannot be guaranteed.

Unlike previous studies, we do not utilize a specific morpheme analyzer (or stemmer) or use typo data collected from the web. Instead, we propose a model that trains with our own generated typo set. Besides, we linearly combine the subword information by reinforcing the necessary information dynamically. Therefore, the model maintains robust performance against different grammatical errors as its embedding generation technique incorporates morphologically complicated Hangeul features.

3 Methodology

In this section, we describe the architecture of the proposed model. The model consists of three parts. The first part is the generation of Korean typos. We create rule-based typo datasets, and they reinforce the robustness of the embedding model for typos. Next, we create a jamo-level CNN with channel attention to represent the word vector. By applying channel attention to features extracted through convolution operation from each jamo, we selectively enhance the importance of the primary n-gram features. Based on a certain probability, the CNN learns to generate the original word vector from the typos, reducing the difference between the typos and the actual word embeddings. Finally, we perform the task of predicting the surrounding context words from the generated word embeddings. Through this process, we can proceed with the reinforcement of the semantics of words. Figure 1 shows the whole architecture of our model.

3.1 Generating Korean Typos

The smallest unit of Hangeul words that corresponds to the alphabet of English is jamo, consisting of 14 consonants and 10 vowels. In Hangeul, a syllable consists of three jamos: chosung, joongsung and jongsung. We divide each syllable into jamo units (Song, 2006). Figure 2 shows an example of dividing each syllable. The syllable “단” is divided into [ㄷ, ㅏ, ㄴ] and “어” is divided into

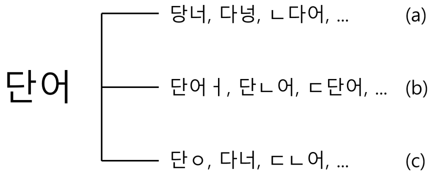


Figure 3: Examples of typos in Korean words. The typos generated in this paper are based on these cases.

[ㅇ, ㅓ]. In Hangeul, the chosung and joongsung jamos are always present while the jongsung jamo is only present or not (e.g., in "어").

After separating the Korean word into jamo, we make the following rules for creating a typo : (a) reversing the order of letters in the word, (b) adding arbitrary letters, or (c) dropping out a letter from the word. We generate our typo dataset following a well-known Korean typo distribution reported in the Korean information processing community (Jeon et al., 2010). Figure 3 shows examples of typos by the three rules. When following the rules, typos in Hangeul cause a breakdown of the word's form. This is because only consonants can be placed in the chosung position and only vowels can be placed in the joongsung position. Therefore, while typos in English can easily infer the original meaning (e.g., englihs, englis, ennglish), Hangeul typos are hard to understand even when they come from native speakers. Consequently, we generate at least 10 to 25 typo word pairs for each Korean word.

3.2 Jamo-level CNN with Channel Attention

After decomposing the input word into jamo units and generating typos, the convolution operation is used to extract the most prominent features of the word. Each one-hot encoded jamo representation is concatenated to set in a form suitable for convolution operations. We use zero padding to fix the input word vector's length to the size of the longest word. Then we obtain jamo representation $K \in \mathbb{R}^{m \times d}$, where m is the maximum length of the input word, and d is the embedding dimension. A convolution operation involves a filter $w \in \mathbb{R}^{h \times d}$, where $h \in \{1, 2, ..., 10\}$ is a filter size. We apply the convolution operation and activation function with these generated jamo representations. For example, a feature c is generated from a window of jamos $w_{i:i+h-1}$ by

$$c_i = ReLU(K^h \cdot w_{i:i+h-1} + b) \tag{1}$$

where $b \in \mathbb{R}$ is a bias. This convolution filter extracts the major feature of the jamos $\{w_{1:h}, w_{2:h+1}, ..., w_{m-h+1:m}\}$ to produce a feature map

$$c = [c_1, c_2, ..., c_{m-h+1}], \tag{2}$$

with $c \in \mathbb{R}^{m-h+1}$. We generate n feature maps in this way. Hence, we obtain concatenated feature map $C \in \mathbb{R}^{(m-h+1) \times n}$, where n is the number of the feature maps. To extract the features for the attention score from the convolution operation results, we employ average and max pooling in channel level as follows:

$$C_{avg} = average(C), \tag{3}$$

$$C_{max} = max(C), \tag{4}$$

where $C_{avg}, C_{max} \in \mathbb{R}^n$. Next, to extract the attention-applied feature map, we compute the attention scores between the channels.

$$C_{score} = W_2(W_1(C_{avg} + C_{max})), \tag{5}$$

where $W_1, W_2 \in \mathbb{R}^{n \times n}$ are the first and second weight matrix of the attention network for calculating the attention score, respectively. C_{score} is the result that comes from passing through the attention network. We apply a softmax function to create a standardized channel attention map. We then broadcast C_{score} into the same shape of C. The channel attention map is multiplied by the continuous jamo pair features to generate a feature map of the input word reflecting an attention score for each channel. Then max pooling is applied to the attention score as follows:

$$C_{att} = max(ReLU(C \times softmax(C_{score}))). \tag{6}$$

We obtain C_{att} for each filter size following this method, and consequently we obtain $C_{att}^h \in \mathbb{R}^{10n}$.

Then C_{att}^h is connected to the fully-connected layer, and we obtain final word representation w as follows:

$$w = C_{att}^h \cdot W + b, \tag{7}$$

where $W \in \mathbb{R}^{10n \times d}$, $b \in \mathbb{R}$. We have selectively enhanced the substructure of the word and created word vector $w \in \mathbb{R}^d$ that reflects contextual information. We then train the jamo-level CNN so that the word vector w resembles the representation of the vector $\hat{w}$ from the pre-trained word. In this step, we use the squared Euclidean distance with each normalized word vector for the objective function as follows:

$$Loss = \sum_{v \in V_w} \| w_v - \hat{w}_v \|^2 . \tag{8}$$

This equation approximates the generated word vector w to the pre-trained word vector $\hat{w}$. To create word embeddings using the proposed model, we split a word to jamo-level and insert it into the model as an input following a similar approach in the training phase. When the input tensor flows through the model and reaches the transformation layer, it is considered as the newly created embeddings of the word.

4 Experiments

In this section, we present our experimental results by comparing the performance of the proposed model with other embedding methods. We measure the performance of the models through word analogy, language modeling, and sentiment classification tasks.

4.1 Experimental Settings

We use an Adam optimizer with a learning rate of 0.001. The convolution filter has a size from one to nine, and each filter has 100 channels. The context window size for calculating semantic loss is set to an arbitrary number of less than five. The corpus used for learning is a Korean Wikipedia dump dataset released on June 1st, 2019. No preprocessing is performed other than removing special characters and replacing numbers with "N" tokens. We tokenize the corpus in word units for all models in the experiment. To ensure fairness, we set different random seeds in the training and evaluation phases.

4.2 Baselines

To measure the performance of the proposed model against widely used and state-of-the-art models, we used the following methods:

- *Word2vec* (Mikolov et al., 2013): *Word2vec* uses a representative embedding technique based on the distributional hypothesis. To find the meaning of a word in context, *Word2vec* learns the meaning of the target word from context words. Following this, if a word that is not registered at the training phase appeared at the evaluation phase, we replaced the word as "UNK" token. We used skip-gram architecture since it is generally proven to be better than other methods.

- *Ko-FastText (Ko-ft)* (Park et al., 2018): *Ko-ft* uses the jamo n-gram as the smallest unit. Therefore, it has an advantage that unseen words can be generated by a combination of jamo n-grams generated in the training phase. This model has proved to have the best performance in Hangeul embedding methods. $ch4$ means the model use one to four jamo-level n-grams, and $ch6$ means the model use one to six n-grams.

4.3 Word Analogy Task

We first conduct an embedding evaluation using the word analogy task as an intrinsic task. We measure how the word embedding process has learned semantic and syntactic meaning. Following (Park et al., 2018), we use 3COSADD based metric as an evaluation metric.

4.3.1 Dataset

We use the dataset introduced in (Park et al., 2018). The dataset is divided into two categories of semantic and syntactic meaning in the English version. Then they are translated into Korean and divided into ten subcategories. In this way, it is suitable to evaluate the performance of Hangeul embedding.

4.3.2 Results

Table 1 shows the analogy task results[1]. *Word2vec* has a big gap compared to *Ko-ft* and *misK*, which shows the limitations of the embedding technique from the viewpoint of words. In general, *misK* outperforms *Ko-ft*, indicating that the *misK* learned the semantic and syntactic meaning of each word better with the help of the well-designed attention mechanism in Figure 1. Also, *misK-ft* generally performs better than the model that learned the shape of words from *Word2vec* (*misK-w2v*). This demonstrates the importance of pre-trained embedding knowledge when creating new embeddings (Pinter et al., 2017).

4.4 Language Modeling Task

We conducted a language modeling experiment to check the embedding performance as an extrinsic

---

[1] Although our word embedding is designed to work in the noisy environment, we examine how it works in the clean (i.e. typo-free) environment.

Model	Semantics					Syntactics				
	Capt	Gend	Name	Lang	Misc	Case	Tense	Voice	Form	Horn
Word2vec	0.471	0.574	0.528	0.466	0.587	0.478	0.561	0.633	0.672	0.665
Ko-ft (ch6)	0.446	0.512	0.524	0.364	0.512	**0.194**	0.417	0.461	0.524	0.362
Ko-ft (ch4)	**0.441**	0.517	0.521	0.368	0.521	0.197	0.422	0.468	0.529	0.369
misK-w2v	0.463	**0.482**	0.513	0.331	**0.486**	0.328	**0.382**	0.436	0.389	0.326
misK-ft	0.460	0.488	**0.498**	**0.327**	0.491	0.311	0.388	**0.421**	**0.361**	**0.294**

Table 1: Result of a word analogy task experiment. *misK-w2v* and *misK-ft* are our proposed models trained from *Word2vec* (Mikolov et al., 2013) and *Ko-FastText* (Park et al., 2018), respectively. Lower score is better. Best results are highlighted in bold font.

Model	Noise level			
	0%	10%	20%	30%
Word2vec	248.3	259.7	269.1	281.4
Ko-ft (ch6)	**206.4**	**218.1**	229.5	241.7
Ko-ft (ch4)	210.2	223.1	231.9	246.3
misK-w2v	213.8	220.7	224.7	229.8
misK-ft	211.6	219.4	**224.6**	**228.4**

Table 2: Perplexity measured from the language modeling. The noise level is the probability that each word in the dataset will be replaced by typos. Best results are highlighted in bold font.

Model	Noise level				
	0%	10%	20%	30%	40%
Word2vec	64.1	62.9	60.4	59.3	58.4
Ko-ft (ch6)	**68.8**	67.3	66.3	64.1	62.3
Ko-ft (ch4)	68.4	66.4	65.2	63.4	61.7
misK-w2v	68.3	**67.6**	**66.7**	65.4	**64.2**
misK-ft	68.1	67.2	66.4	**65.5**	64.1

Table 3: Result of the sentiment classification experiment. The noise level is the probability that each word in the dataset will be mistyped. Best results are highlighted in bold font.

task. Also, we replaced the word embeddings of the basic LSTM network with dropout regularization (Kim et al., 2016). All embedding dimensions are set to 300 and the dropout probability is set to 0.5. Each embedding model is measured using perplexity, which is widely used to measure language modeling performance (Katz, 1987). The lower the perplexity value measured, the more likely the model predicts the next word correctly. Besides, in the downstream task, we used intentionally generated typo data in the experiment to verify the robustness of embedding for noisy text.

4.4.1 Datasets

The language modeling dataset was released by the Korean Wikipedia, published on June 1, 2019. Each embedding technique was trained based on this corpus in its way. The code to extract and refine the text from the Wikipedia database dump followed the open-source software[2].

4.4.2 Results

The results of the language modeling experiments are shown in Table 2. Regardless of the noise level, *Word2vec* has the lowest performance. In other words, it can be seen that the word-level embeddings separated by blank spaces are not suitable for Korean text irrespective of the response to misspelled words. *Ko-ft* performs best when the noise level is below 20%. Meanwhile, the performance of *misK* is better than the other models as the noise level rises to more than 10%. The fact that *misK* showed superior performance means it can stably respond to various typos in Hangeul text.

4.5 Sentiment Classification Task

The performance of each model is compared using the sentiment analysis experiment on movie reviews. We represent the reviews by averaging the word embeddings, constructing a review, and training through the logistic regression classifier for a fair comparison.

4.5.1 Datasets

For sentiment classification, we used the Naver Sentiment Movie Corpus[3] as our classification data. For the typos used in the real world, we did not proceed with any form of preprocessing. Therefore, it includes typos or slang used by Koreans. The experiment was also conducted by increasing noise

---

[2] https://github.com/attardi/wikiextractor

[3] https://github.com/e9t/nsmc/

	In-Vocabulary				Out-Of-Vocabulary			
	컴퓨터	대한민국	대학교	자동차	커뮤펕	댕나민국	댕가교	자도앚
Word2vec	개인용	한국	대학교의	승용차	-	-	-	-
	그래픽	국내	대학교에서	자동차의	-	-	-	-
	컴퓨터의	대한민국의	대학교의	자동차가	-	-	-	-
FastText	컴퓨터의	대한민국은	대학교는	자동차를	커뮤니티	**중화민국**	침례교	자
	컴퓨터에	대한민국을	대학교와	자동차도	객실	**대한민국**	외교	재차
	컴퓨터로	대한민국이	대학교의	자동차에	앱	**대한민국과**	페교	자작
misK-ft	컴퓨터는	한국	대학교의	자동차에	코뮌	**대한민국**	**대학원을**	**자동차**
	컴퓨팅	대한민국에서	대학의	자동차의	**소프트웨어**	**중화민국**	**대학원의**	**자동차의**
	소프트웨어	국내	대학교는	자동차가	**컴퓨터는**	웹	**학사**	광고

Table 4: In OOV experiments, words that are related to the original word are highlighted in bold font. 커뮤니티-community, 객실-room, 앱-app, 소프트웨어-software, 침례교-baptist church, 외교-diplomacy, 페교-closing schools, 학사-bachelor, 자-ruler, 재차-again, 자작-oneself.

level of the evaluation dataset and intentionally producing misspelled words to compare robustness to the typos.

4.5.2 Results

Table 3 shows that *Ko-ft* has the highest performance in the sentiment classification experiment results in the clean environment. Besides, *Word2vec* showed significantly reduced performance than *Ko-ft*. This is because several OOVs were included in the test set. However, as the noise level increases, the performance of *misK* overtakes that of the *Ko-ft*. It confirms that the performance of *misK* decreases relatively smoothly as the amount of noise in the data rises. In other words, *misK* is more robust to the noisy data than the others.

5 Analysis

In this section, we analyze the proposed methodology. We find the nearest neighbor words mapped close to the misspelled word to evaluate robustness of the typo embeddings qualitatively.

5.1 Nearest Neighbor of Words

We check the nearest neighbor of words to compare the performance of each word embedding. This evaluation is to identify which words are located around the represented target words (컴퓨터-computer, 대한민국-Korea, 대학교-university, 자동차-automobile) and typos transformed from these words.

Table 4 shows the experimental results. The results with In-Vocabulary show that all three models produce the appropriate embeddings. However, the results from the OOV confirm that *Word2vec* cannot deal with the typos. In contrast, *Ko-ft* and *misK* can generate embeddings for these unseen words to investigate these nearest words. In general, the typo embeddings generated by the *Ko-ft* are inaccurate (note the meaning of the words described in the caption). On the other hand, we deduce that words with appropriate meanings are mapped with our proposed model, *misK*.

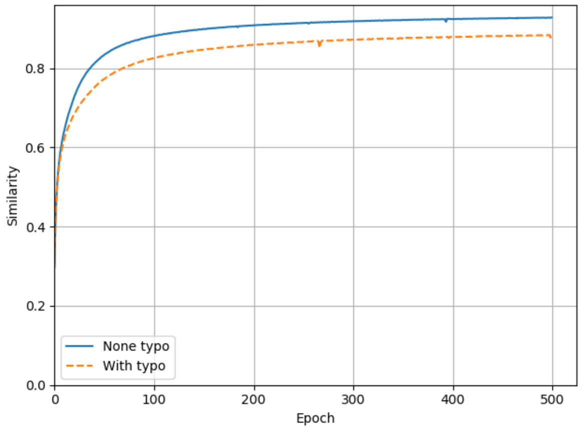


Figure 4: Cosine similarity between the generated word vector and pre-trained word vector.

5.2 Similarity between Generated and Pre-trained Word Vector

Experiments show superiority in the noisy text of our proposed embedding generation model. We measure the cosine similarity between the pre-trained word embeddings and those generated by our proposed model. As shown by the dotted curve in Figure 4, when training data includes intentional typos, the cosine similarity between the pre-trained word embeddings and those generated by the proposed model could not be increased beyond 0.86. On the other hand, in the absence of typos, the similarity increases to around 0.95. Besides, there is a tradeoff between the ability to map typos closely to the original word and the ability to mimic pre-trained word vectors.

6 Conclusion

In this paper, we proposed a robust Hangeul embedding model against typos. For this purpose, we intentionally generated typos to train word embeddings. A channel attention mechanism is used to utilize the structure information found in Korean words selectively. This reflects the distinctive characteristics of Hangeul, which combines several morphemes to form a single word, without depending on the performance of a morpheme analyzer. Based on this, the training progresses from the extracted jamo n-gram feature of the target words through the convolution layer to aim the pre-trained word embeddings. The proposed embedding model outperformed the current state-of-the-art in the intrinsic task. Also, we verify that the original research goal was achieved by functioning a stable embeddings even in the extrinsic tasks. We plan to apply the proposed methodology to various downstream tasks (e.g., POS tagging and machine translation) and other agglutinative languages, such as Japanese and Turkish.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This research was supported by the Basic Research Program through the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. 2020R1A4A1018309), the NRF grant funded by the Korea government (MSIT) (No. 2018R1A2A1A05078380), and Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2019-0-00079, Artificial Intelligence Graduate School Program (Korea University)).

References

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of International Conference on Learning Representations*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In *Proceedings of the International Conference on Computational Linguistics*, pages 6903–6915.

Hee-Won Jeon, Daniel Huang, and Hae-Chang Rim. 2010. Analyzing of hangul search query spelling error patterns and developing query spelling correction system based on user logs. In *Proceedings of Annual Conference on Human and Language Technology*, pages 15–21.

Slava Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.

Yeachan Kim, Kang-Min Kim, Ji-Min Lee, and SangKeun Lee. 2018. Learning to generate word representations using subword information. In *Proceedings of the International Conference on Computational Linguistics*, pages 2551–2561.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Proceedings of the American Association for Artificial Intelligence Conference on Artificial Intelligence*.

Dongjun Lee, Yubin Lim, and Taekyoung Kwon. 2018. Morpheme-based efficient korean word embedding. *Journal of Korean Institute of Information Scientists and Engineers*, 45(5):444–450.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1064–1074.

Valentin Malykh, Varvara Logacheva, and Taras Khakhulin. 2018. Robust word vectors: Context-informed embeddings for noisy texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 54–63.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3111–3119.

Kyeong-Min Nam and Yu-Seop Kim. 2016. A word embedding and a josa vector for korean unsupervised semantic role induction. In *Proceedings of the American Association for Artificial Intelligence Conference on Artificial Intelligence*, pages 4240–4241.

Sungjoon Park, Jeongmin Byun, Sion Baek, Yongseok Cho, and Alice Oh. 2018. Subword-level word vector representations for korean. In *Proceedings of* *the Annual Meeting of the Association for Computational Linguistics*, pages 2429–2438.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Aleksandra Piktus, Necati Bora Edizel, Piotr Bojanowski, Edouard Grave, Rui Ferreira, and Fabrizio Silvestri. 2019. Misspelling oblivious word embeddings. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3226–3234.

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2017. Mimicking word embeddings using subword rnns. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 102–112.

Jae Jung Song. 2006. *The Korean language: Structure, use and context*. Routledge.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 649–657.

Exploiting Multimodal Reinforcement Learning for Simultaneous Machine Translation

Julia Ive[1], **Andy Mingren Li**[1], **Yishu Miao**[1], **Ozan Caglayan**[1],
Pranava Madhyastha[1], **Lucia Specia**[1,2,3]
Imperial College London[1], University of Sheffield[2], ADAPT - Dublin City University[3]
`j.ive@ic.ac.uk`, `andy.li16@imperial.ac.uk`, `y.miao20@imperial.ac.uk`, `o.caglayan@ic.ac.uk`
`pranava@ic.ac.uk`, `l.specia@ic.ac.uk`

Abstract

This paper addresses the problem of simultaneous machine translation (SiMT) by exploring two main concepts: (a) adaptive policies to learn a good trade-off between high translation quality and low latency; and (b) visual information to support this process by providing additional (visual) contextual information which may be available before the textual input is produced. For that, we propose a multimodal approach to simultaneous machine translation using reinforcement learning, with strategies to integrate visual and textual information in both the agent and the environment. We provide an exploration on how different types of visual information and integration strategies affect the quality and latency of simultaneous translation models, and demonstrate that visual cues lead to higher quality while keeping the latency low.

1 Introduction

Research into automating real-time interpretation has explored deterministic and adaptive approaches to build policies that address the issue of translation delay (Ryu et al., 2006; Cho and Esipova, 2016; Gu et al., 2017). In another recent development, the availability of multimodal data (such as visual information) has driven the community towards multimodal approaches for machine translation (MMT) (Specia et al., 2016; Elliott et al., 2017; Barrault et al., 2018). Although deterministic policies have been recently explored for simultaneous MMT (Caglayan et al., 2020; Imankulova et al., 2020), there are no studies regarding how multimodal information can be exploited to build flexible and adaptive policies for simultaneous machine translation (SiMT).

Applications of reinforcement learning (RL) for unimodal SiMT have highlighted the challenges for the agent to maintain good translation quality while learning an optimal translation path (i.e. a sequence of `READ`/`WRITE` decisions at every time step) (Grissom II et al., 2016; Gu et al., 2017; Alinejad et al., 2018).

Incomplete source information will have detrimental effect especially in the cases where significant restructuring is needed while translating from one language to another.

In addition, the lack of information generally leads to high variance during the training in the RL setup. We posit that **multimodality** in adaptive SiMT could help the agent by providing extra signals, which would in turn improve training stability and thus the quality of the estimator and translation decoder.

In this paper, we present the first exploration on multimodal RL approaches for the task of SiMT.

As visual signals, we explore both image classification features as well as visual concepts, which provide global image information and explicit object representations, respectively. For RL, we employ the Policy Gradient method with a pre-trained neural machine translation model acting as the environment.

As the SiMT model is optimised for both translation quality and latency, we apply a combined reward function that consists of a decomposed smoothed BLEU score and a latency score. To integrate visual and textual information, we propose different strategies that operate both on the agent (as prior information or at each step) and the environment side.

In experiments on standard datasets for MMT, our models achieve the highest BLEU scores on most settings without significant loss on average latency, as compared to strong SiMT baselines. A qualitative analysis shows that the agent benefits from the multimodal information by grounding language signals on the images.

Our **main contributions** are as follows: (1) we

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3222–3233
April 19 - 23, 2021.

propose the first multimodal approach to simultaneous machine translation based on adaptive policies with RL, introducing different strategies to integrate visual and textual information (Sections 3 and 4); (2) we show how different types of visual information and integration strategies affect the quality and latency of the models (Section 5); (3) we demonstrate that providing visual cues to both agent and environment is beneficial: models achieve high quality while keeping the latency low (Section 5).

2 Related Work

In this section, we first present background and related work on SiMT, and then discuss recent work in MMT and multimodal RL.

2.1 Simultaneous Machine Translation

In the context of neural machine translation (NMT), Cho and Esipova (2016) introduce a greedy decoding framework where simple heuristic waiting criteria are used to decide whether the model should read more source words or instead write a target word. Gu et al. (2017) utilise a pre-trained NMT model in conjunction with an RL agent whose goal is to learn a `READ/WRITE` policy by maximising quality and minimising latency. Alinejad et al. (2018) further extend the latter approach by adding a `PREDICT` action with an aim to capture the anticipation of the next source word. Ma et al. (2019) propose an end-to-end, fixed-latency framework called 'wait-k' which allows *prefix-to-prefix* training using a deterministic policy: the agent starts by reading a specified number of source tokens (k), followed by alternating `WRITE` and `READ` actions.

Other approaches to SiMT include re-translation of previous outputs depending on new outputs (Arivazhagan et al., 2020; Niehues et al., 2018) or learning adaptive policies guided by a heuristic or alignment-based approaches (Zheng et al., 2019; Arthur et al., 2020). A general theme in these approaches is their reliance on *consecutive* NMT models pre-trained on full-sentences. However, Dalvi et al. (2018) discuss potential mismatches between the training and decoding regimens of these approaches and propose to perform fine-tuning of the models using chunked data or prefix pairs.

2.2 Multimodal Machine Translation

MMT aims at improving the quality of automatic translation using additional sources of information (Sulubacak et al., 2020). Different methods for fusing textual and visual information have been proposed. These include initialising the textual encoder or decoder with the visual information (Elliott and Kádár, 2017; Caglayan et al., 2017), combining the visual information through spatial feature maps using soft attention (Caglayan et al., 2016; Libovický and Helcl, 2017; Huang et al., 2016; Calixto et al., 2017), and projecting a summary of the visual representations to a common context space via a trained projection matrix (Calixto and Liu, 2017; Caglayan et al., 2017; Elliott and Kádár, 2017; Grönroos et al., 2018). Further, recent work has also focused on exploring Multimodal Pivots (Hitschler et al., 2016) and latent variable models (Calixto et al., 2019) in the context of multimodal machine translation. In this paper, we explore all these strategies, and also the use of *visual concepts*, similar to the approach by Ive et al. (2019).

2.3 Multimodal Reinforcement Learning

Previous work has explored RL with language inputs (Andreas et al., 2017; Bahdanau et al., 2018; Goyal et al., 2019) by making use of language to improve the policy or reward function: for example, the task of navigating in the world grid environment using language instructions (Andreas et al., 2016).

Alternatively, RL with language output can be shaped as sequential decision making for language generation, while conditioning on other modalities. This includes image captioning (Ren et al., 2017), video captioning (Wang et al., 2018), question answering (Das et al., 2018), and text-based games (Côté et al., 2018). Our study sits somewhere in between these different types of work. We have both the source language and respective images as input and the target language as output. Our agent is focused only on learning the `READ` and `WRITE` actions while the translation model is fixed for simplicity.

The central aim of the agent is learning to capture the relevant structures and relations of the modalities that can lead to a better SiMT system.

3 Methods

We first present the architectures for consecutive and baseline fixed policy simultaneous MT (Section 3.1). Then we introduce our RL approaches, both the baseline, the proposed multimodal extension (Section 3.2), and the visual features used by all multimodal approaches (Section 3.3).

3.1 Baselines

Unimodal MT. We implement a standard encoder-decoder baseline with attention (Bahdanau et al., 2014) which incorporates a two-layer encoder and a two-layer decoder with GRU (Cho et al., 2014) units. Given a source sequence of embeddings $X=\{x_1, \ldots, x_S\}$ and a target sequence of embeddings $Y=\{y_1, \ldots, y_T\}$, the encoder first computes the sequence of hidden states $H=\{h_1, \ldots, h_S\}$ unidirectionally.

The attention layer receives H as *key-values* whereas the hidden states of the first decoder GRU provide the *queries*. The context vector $c_t^{\mathbf{T}}$ produced by the attention layer is given as input to the second GRU. Finally, the output token (y_t) probabilities are obtained by applying a softmax layer on top of the concatenation of the previous word embedding, context vector and the second GRU's hidden state.

For **consecutive NMT**, all source tokens are observed before the decoder begins the process of generation.

Multimodal MT. We extend unimodal MT with multimodal attention (Calixto et al., 2016; Caglayan et al., 2016) in the decoder, in order to incorporate visual information into the baseline NMT. Let us denote the visual counterpart of textual hidden states H by V. Multimodal attention simply applies another attention layer on top of V, which yields a visual context vector $c_t^{\mathbf{V}}$ at each decoding timestep t. The final multimodal context vector that would be given as input to the second GRU is simply the sum of both context vectors.

Unimodal wait-k NMT. We explore deterministic wait-k (Ma et al., 2019) approach as a unimodal baseline[1] for simultaneous NMT. The wait-k model starts by reading k source tokens and writes the first target token. The model then reads and writes one token at a time to complete the translation process. This implies that the attention layer will now attend to a partial textual representation corresponding to k-words. We use the decoding-only variant which does not require re-training an NMT model i.e. it re-uses the already trained consecutive NMT baselines.

3.2 Policy Learning Framework

RL baseline. We closely follow Gu et al. (2017) and cast SiMT as a task of producing a sequence of `READ` or `WRITE` actions. We then devise an RL model that connects the MT system and these actions. The model is based on a reward function that takes into account both quality and latency. Following standard RL, the framework is composed of an environment and an agent. The agent takes the decision of either reading one more input token or writing a token into the output – hence two actions are possible: `READ` and `WRITE`. The environment is a pre-trained NMT system which is *frozen* during RL training.

The agent is a GRU that parameterises a stochastic policy which decides on the action a_t by receiving as input the observation o_t.[2] In our setup, o_t is defined as $[c_t^{\mathbf{T}}; y_t; a_{t-1}]$, i.e. the concatenation of vectors coming from the environment, as well as the previously produced action sequence. At each time step, the agent receives a reward $r_t = r_t^Q + r_t^D$ where r_t^Q is the quality reward (the difference of smoothed BLEU scores for partial hypotheses produced from one step to another) and r_t^D is the latency reward formulated as:

$$r_t^D = \alpha \left[\mathrm{sgn}(C_t - C^*) + 1\right] + \beta \lfloor D_t - D^* \rfloor_+$$

where C_t denotes the consecutive wait (CW) metric which is added to avoid long consecutive waits (Gu et al., 2017). CW measures how many source tokens are consecutively read between committing two translations. D_t refers to average proportion (AVP) (Cho and Esipova, 2016), which defines the average proportion of wait tokens when translating the words. D^* and C^* are hyper-parameters that determine the expected/target values. The optimal quality-latency trade-off is achieved by balancing the two reward terms. In our reward implementation we again closely follow Gu et al. (2017).

Multimodal extension. Here we focus on integrating the visual information with the agent (see Figure 1). The basic premise is that the addition of multimodal information, especially in the context of MMT, can result in the agent learning better and more flexible policies. We explore several ways to integrate visual information into this framework:

---

[1] These baselines are equivalent to the deterministic approaches used in Caglayan et al. (2020).

[2] We note that the use of GRU cells is not critical for the multimodal components. They were chosen as they led to the best performance in our implementation.

- **Multimodal initialisation** (`RL-init`) - the agent network is initialised with the image vector V as d_0. We expect this vector to give the agent some context w.r.t. the source sentence so it can potentially read fewer words before producing outputs.

- **Multimodal attention** (`RL-att`, Figure 1) applies another attention layer on top of V, which yields a visual context vector $c_t^{\mathbf{V}}$ at each agent time step t. This visual context vector is a dot product attention $c_t^{\mathbf{V}} = \text{Attention}(V, \text{query} \leftarrow y_t)$ that computes the similarity between V and the embedding of the target word produced by the decoder at the time step t. In this setting, we expect the agent to pay attention to the information in V that will help in defining whether y_t is good enough to be written to the output (potentially with closer relationship to some part of the image information) or we need to read more source words to produce a better y_t. We concatenate $c_t^{\mathbf{V}}$ to o_t, which now becomes $[c_t^{\mathbf{T}}; y_t; a_{t-1}; c_t^{\mathbf{V}}]$;

- As a **control**, we also study **multimodal environment** (`RL-env`, Figure 1) where we use the MMT baseline as environment. Here, we expect the initial translation quality of SiMT RL models be closer to the quality of the respective consecutive multimodal baseline as the image information is expected to compensate for partial source information. When combined with `RL-init` and `RL-att` settings, we expect the agent to exploit different kinds of image information than the environment.

Learning. To learn the multimodal agent, we introduce an additional neural network with the same structure as that of the agent GRU network to provide for control variates (baselines) that improve the Monte-Carlo policy gradient (REINFORCE (Williams, 1992)). Note that here we depart from the previous work where Gu et al. (2017) use a simple multilayer perceptron as the baseline.

Therefore, with the reward r_t at each time step, we obtain the estimation of the gradients by subtracting the baselines $b(o_t)$:

$$\nabla_\theta J(\theta) = \mathbb{E}[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t|o_t)(r_t - b(o_t))]$$

To further reduce the variance of the gradient estimator, we also introduce a temperature τ for controlling the interpolation between discrete action samples and continuous categorical densities, which yields to a Gumbel-Softmax reparameterisation (Jang et al., 2017) that smooths the learning. To be more precise, we use the Gumbel-Softmax distribution instead of argmax while sampling. So the probability of the `WRITE` action is given to the agent network instead of the index of the action.

3.3 Visual Features

In order to represent the visual information, we explore two settings that differ in the organisation of the spatial structure. Regardless of the setting, the image features are linearly projected into the hidden space of the decoder to yield the tensor V.

Image classification features (OC) are *global* image information represented by convolutional feature maps, which are believed to capture spatial cues. These features are extracted from the final convolution layer of a ResNet-50 convolutional neural network (CNN) (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009) for object classification. The size of the final feature tensor being 8x8x2048, the visual attention is applied on a grid of 64 equally-sized regions.

Visual Concepts (VC) are explicit object representations where *local* regions are detected as objects and subsequently encoded with 100-dimensional word representations. For a given image, the detector provides 36 object and 36 attribute region proposals which are abstract concepts associated with the image. We represent each of the detected region with its corresponding GloVe (Pennington et al., 2014) word vectors. An image is thus represented by a feature tensor of size 72x100 and the visual attention is now applied on these visual concepts, rather than the uniform grid of the first approach above. We hypothesise that this type of information can result in better referential grounding by using conceptually meaningful units rather than global features. The detector used here is a Faster R-CNN/ResNet-101 object detector (with 1600 object labels) (Anderson et al., 2018)[3] pre-trained on the Visual Genome dataset (Krishna et al., 2017).

---

[3] https://hub.docker.com/r/airsplay/bottom-up-attention

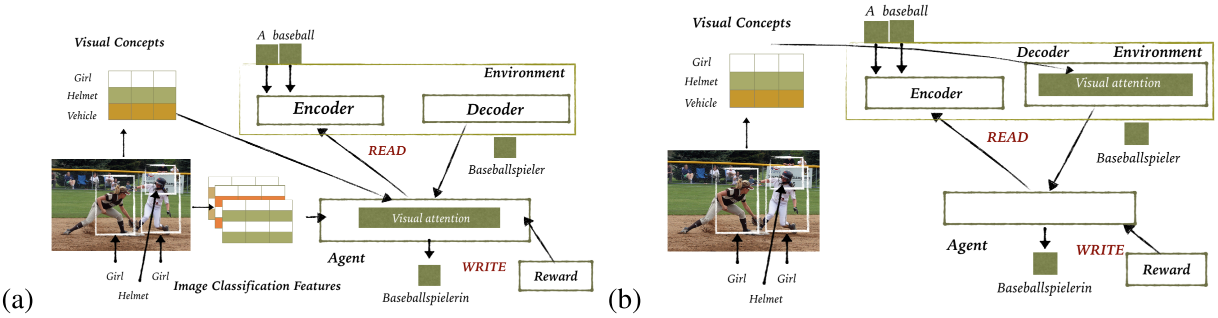


Figure 1: Our multimodal RL SiMT models: the agent interacts with the environment to receive new translation and at each time step produces the `READ/WRITE` action. For each action it receives a reward. The image information can be integrated into the agent by means of an attention mechanism (a, `RL-att`), or into the environment decoder (b, `RL-env`) producing the next translation.

4 Experimental Setup

4.1 Dataset

We perform experiments on the Multi30k dataset (Elliott et al., 2016)[4] which extends the Flickr30k image captioning dataset (Young et al., 2014) with caption translations in German and French (Elliott et al., 2017). Multi30k is a standard MMT dataset that contains parallel sentences in two languages that describe the images. The training set for each language direction comprises 29,000 image-source-target triplets whereas the development and the test sets have around 1,000 samples. We use the corresponding test sets from 2016, 2017 and 2018 for evaluation.

Pre-processing. We use Moses scripts (Koehn et al., 2007) to lowercase, normalise and tokenise the sentences. We then create word vocabularies on the *training* subset of the dataset. We did not use subword segmentation to avoid its potential side effects on fixed policy SiMT and to be able to better analyse the grounding capability of the models. The resulting English, French and German vocabularies contain 9.8K, 11K and 18K tokens, respectively.

4.2 Evaluation

We use **BLEU** (Papineni et al., 2002) for quality, and perform significance testing via bootstrap resampling using the `Multeval` tool (Clark et al., 2011). For latency, we measure **Average proportion** (AVP) (Cho and Esipova, 2016). AVP is the average number of source tokens required to commit a translation. This metric is sensitive to the difference in lengths between source and target. Hence, as our main latency metric we measure **Average Lagging** (AVL) (Ma et al., 2019) which estimates the number of tokens the "writer" is lagging behind the "reader", as a function of the number of input tokens read.

4.3 Training

Hyperparameters. We set the embeddings dimensionality and GRU hidden states to 200 and 320, respectively. We use the ADAM (Kingma and Ba, 2014) optimiser with the learning rate 0.0004 and the batch size of 64. We use `pysimt` (Caglayan et al., 2020) with PyTorch (Paszke et al., 2019) v1.4 for our experiments.[5] We early stop w.r.t. the validation BLEU with the patience of 10 epochs. On a single NVIDIA RTX2080-Ti GPU, the training takes around 35 minutes for the unimodal model and around 1 hour for the multimodal model. The number of learnable parameters is between 6.9M and 9.3M depending on the language pair and the type of multimodality.

For the **RL systems**, we follow (Gu et al., 2017).[6] The agent is implemented by a 320-dimensional GRU followed by a softmax layer and the baseline network is similar to the agent except with a scalar output layer.[7] We use ADAM as the optimiser and set the learning rate and mini-batch size to 0.0004 and 6, respectively. For each sentence pair in a batch, 5 trajectories are sampled. Following best practises in RL, the baseline network is trained to reduce the MSE loss between the predictions and the rewards using a second op-

---

[4] https://github.com/multi30k/dataset

[5] https://github.com/ImperialNLP/pysimt

[6] https://github.com/nyu-dl/dl4mt-simul-trans

[7] Note that that Gu et al. (2017) use a 2-hidden layer feed-forward network as the baseline network. In our implementation GRUs have demonstrated better performance.

timiser.

For inference, greedy sampling is used to pick action sequences. We set the hyperparameters C^{*}=2, D^{*}=0.3, α=0.025 and $\beta = -1$. To encourage exploration, the negative entropy policy term is weighed empirically with 0.001. Following (Gu et al., 2017), we choose the model that maximises the quality-to-latency ratio (BLEU/AVP) on the validation set with a patience of 5 epochs.[8] On a single NVIDIA RTX2080-Ti GPU, the training takes around 2 hours. The number of learnable parameters is around 6M.

Model configurations. We experiment with seven different configurations (below). We consider visual concepts (VC) as the main source of multimodal information. Visual concepts are more abstract forms of multimodal information. Unlike spatial image representation or region of interest-based object representations, where the representation for the same concept can vary significantly across images, visual concepts remain constant. For example, the visual concept "dog" is the same regardless of the breed, colour, size, position, etc. of the concept in different images. Image classification (OC) features are used as a contrastive setting.

- Unimodal RL baseline (`RL-base`): This baseline follows (Gu et al., 2017) where the environment is a text-only NMT model.
- Multimodal agent with VC initialisation (`RL-init` VC): We initialise the agent GRU using a projection of the flattened 72x100 matrix of visual concepts.
- Multimodal agent with attention over VC (`RL-att` VC): The agent attends over the set of visual concepts at each step.
- Multimodal agent with attention over OC (`RL-att` OC): The agent attends over the set of image classification-based spatial feature maps at each step.
- Visually initialised multimodal agent with attention over VC (`RL-init-att` VC): Similar to `RL-att` VC but the agent is also initialised with VC.
- Multimodal environment with unimodal RL agent (`RL-env` VC): The environment is an MMT model, however the agent is a standard RL agent akin to the baseline.
- Multimodal agent with multimodal environment (`RL-env-init-att` VC): This merges all the variants in that both the multimodal environment and the multimodal agent attend to visual concepts, the latter is also initialised with visual information.

5 Results

In this section, we first provide the results from our experiments (Section 5.1) and then analyse the behaviour of the (multimodal) agents (Section 5.2).

5.1 Quantitative Results

SiMT vs. Consecutive. We present the main results in Table 1. The top block for each language pair shows the textual Consecutive model and its multimodal counterpart (Consecutive+VC). These are our upperbounds since they have access to the entire source before translating. As expected, they have better BLEU but much larger AVL.

RL SiMT vs. Deterministic policy. The second block in Table 1 shows the deterministic policy Wait-2 and Wait-3 approaches. `RL-base` performs on par with the Wait-2 (English-French) and Wait-3 (English-German). We however emphasise the flexibility of the stochastic policies with RL models. These are particularly beneficial in the multimodal scenario and allow for exploitation of the image information more efficiently especially towards reducing the average lag. We further expand on this later in Section 5.2.

Unimodal RL vs. Multimodal RL. The third block in Table 1 compares all multimodal RL variants against the text-only SiMT RL (`RL-base`). In general, the multimodal RL models produce translations that are significantly better than `RL-base`.

Across Multimodal RL Setups. With regard to different configurations, we observe (1) an increase in quality for the `RL-att` models when compared to `RL-base` which is consistent in both types of visual inputs OC and VC, and (2) a decrease in the lag for the `RL-init` models at a small decrease in quality (for VC `RL-init` in comparison to `RL-base`).

This observation suggests that the RL model with the agent explicitly attending over image information leads to an increase in quality, as the

---

[8] We also attempted to choose the model that maximises BLEU or BLEU/AVL but those stopping criteria resulted in instability of convergence.

		test 2016			test 2017			test 2018		
		BLEU↑	AVL↓	AVP↓	BLEU↑	AVL↓	AVP↓	BLEU↑	AVL↓	AVP↓
English ⟶ French	Consecutive	58.0	13.1	1.0	50.6	11.1	1.0	36.0	13.8	1.0
	+VC	59.1	13.1	1.0	51.0	11.1	1.0	36.5	13.8	1.0
	Wait-2	48.1	2.6	0.7	42.9	2.6	0.7	32.1	2.7	0.7
	Wait-3	54.0	3.5	0.7	48.6	3.5	0.7	35.5	3.5	0.7
	`RL`	50.8	3.3	**0.7**	44.3	3.0	**0.7**	32.1	3.5	**0.7**
	+`att`-OC	53.0*	4.1	0.8	46.4*	3.9	0.8	33.3*	4.4	0.8
	+`att`-VC	53.0*	4.0	**0.7**	46.5*	3.7	0.8	33.3*	4.2	**0.7**
	+`init`-VC	49.6	**2.8**	**0.7**	43.3	**2.6**	**0.7**	31.5	**2.9**	**0.7**
	+`init-att`-VC	52.6*	3.8	**0.7**	46.3*	3.6	**0.7**	33.3*	4.1	**0.7**
	+`env`-VC	54.0*	3.3	**0.7**	47.2*	3.1	**0.7**	33.7*	3.4	**0.7**
	+`env-init-att`-VC	**54.0***	3.9	**0.7**	**47.7***	3.8	0.8	**34.4***	4.2	**0.7**
English ⟶ German	Consecutive	35.5	13.1	1.0	27.7	11.1	1.0	25.8	13.8	1.0
	+VC	35.9	13.1	1.0	27.0	11.1	1.0	25.4	13.8	1.0
	Wait-2	28.3	2.2	0.6	22.5	2.2	0.7	20.1	2.2	0.6
	Wait-3	32.6	3.0	0.7	25.4	3.0	0.7	24.1	3.0	0.7
	`RL`	31.0	2.7	0.7	23.0	2.6	0.7	22.0	2.7	0.7
	+`att`-OC	33.9*	3.7	0.7	**25.8***	3.4	0.7	**24.5***	3.8	0.7
	+`att`-VC	33.3*	3.3	0.7	24.7*	3.0	0.7	23.0*	3.2	0.7
	+`init`-VC	29.7	2.8	0.7	21.3	2.4	0.7	20.5	2.5	0.6
	+`init-att`-VC	**34.1***	3.3	0.7	25.3*	3.1	0.7	24.1*	3.4	0.7
	+`env`-VC	30.0	**2.5**	**0.6**	21.7	**2.2**	**0.6**	19.7	**2.2**	**0.6**
	+`env-init-att`-VC	31.4	3.0	0.7	24.0*	2.9	0.7	22.4	3.0	0.7

Table 1: Results for the test sets 2016, 2017 and 2018 (averaged over 3 runs): * marks statistically significant increases in BLEU w.r.t. `RL-base` (p-value ≤ 0.05). Bold highlights best scores across the RL approaches.

multimodal agent model is more selective towards the word choice. The `RL-init` configuration with prior image context on the other hand reduces the lag and seems to use `WRITE` actions more often than `READ` actions. It is interesting that OC and VC features result in similar quality translations, however we see that on average the average lag is lower with VC. We hypothesise that this could be due to the fact that the representations remain constant across images (see Section 4.3).

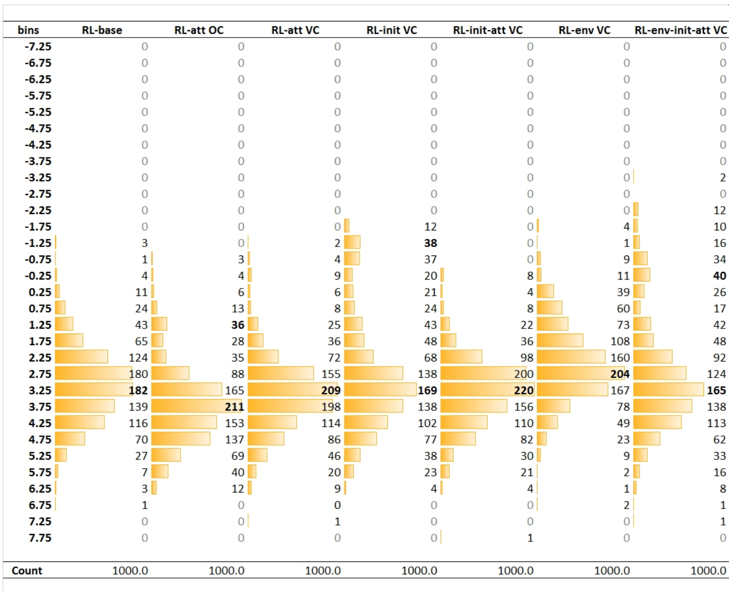
The `RL-init-att` configuration represents a middle ground and we see similar quality improvement to `RL-att` across setups (a gain of 2 BLEU points on average) but with a slightly lower latency. We however observe that `RL-env-init-att` has a slightly inferior performance with a a pronounced latency when compared to the `RL-env` model. We investigate this aspect in the next sections.

Investigating Average Lag. To further study the impact of our configurations on the sentence level lag, in Figure 2 we present the binned-histograms of sentence lags over the English→German test 2016 set. Generally, the models which are initialised with image information seem to have more mass towards the smaller delay bins. In terms of `RL-init` and `RL-env-init-att` setups, we also observe the presence of two modes around the lag value 3 as well as around two negative values (around -0.25 and -1.25 respectively). These negative lag values are due the difference in length between source and target sentences which is typical for the English→German. This also shows that the agent initialised with the image information tends to prefer `WRITE` actions with fewer `READ` actions. Further, on manual inspection of some samples, we observed that in the cases with negative lag the model begins with a `WRITE` action straight after reading the first token (See Table 2). As the agent is a GRU model, this behavior resembles that of an image captioning model. We also observe similar trends for English→French with `RL-init` models predominantly having more mass towards smaller delay bins (see Figure 3).

5.2 Agent Attention over Visual Inputs

In Figure 4 we visualize the agent's attention at each time step. On average, the agent actions correlate with the objects it attends to when producing the translation.

We now examine the general pattern of agent attention over the visual concepts across the four configurations using attention norm: a) `RL-att`-VC; b) `RL-att`-OC; c) `RL-init-att`; and d)



bins	RL-base	RL-att OC	RL-att VC	RL-init VC	RL-init-att VC	RL-env VC	RL-env-init-att VC
-7.25	0	0	0	0	0	0	0
-6.75	0	0	0	0	0	0	0
-6.25	0	0	0	0	0	0	0
-5.75	0	0	0	0	0	0	0
-5.25	0	0	0	0	0	0	0
-4.75	0	0	0	0	0	0	0
-4.25	0	0	0	0	0	0	0
-3.75	0	0	0	0	0	0	0
-3.25	0	0	0	0	0	0	2
-2.75	0	0	0	0	0	0	0
-2.25	0	0	0	0	0	0	12
-1.75	0	0	0	12	0	4	10
-1.25	3	0	2	**38**	0	1	16
-0.75	1	3	4	37	0	9	34
-0.25	4	4	9	20	8	11	**40**
0.25	11	6	6	21	4	39	26
0.75	24	13	8	24	8	60	17
1.25	43	**36**	25	43	22	73	42
1.75	65	28	36	48	36	108	48
2.25	124	35	72	68	98	160	92
2.75	180	88	155	138	200	**204**	124
3.25	**182**	165	**209**	**169**	**220**	167	**165**
3.75	139	**211**	198	138	156	78	138
4.25	116	153	114	102	110	49	113
4.75	70	137	86	77	82	23	62
5.25	27	69	46	38	30	9	33
5.75	7	40	20	23	21	2	16
6.25	3	12	9	4	4	1	8
6.75	1	0	0	0	0	2	1
7.25	0	0	1	0	0	0	1
7.75	0	0	0	0	1	0	0
Count	1000.0	1000.0	1000.0	1000.0	1000.0	1000.0	1000.0

Figure 2: Histogram of per sentence lag values in test 2016 English-German. Y axis shows mean values per bin. Bold highlights modes for each distribution.

`SRC`: the red car is ahead of the two cars in the background .
`REF`: das rote auto fährt vor den beiden autos im hintergrund .
'the red car goes before the both cars in the background'
`RL-init`: die person ist im begriff , die rote mannschaft auf dem roten auto versammelt .
'the person is in concept, that red manhood on the red car gathered'
`Actions`: 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 0 1
`BLEU`: 3.7
`LAG`: -1.875

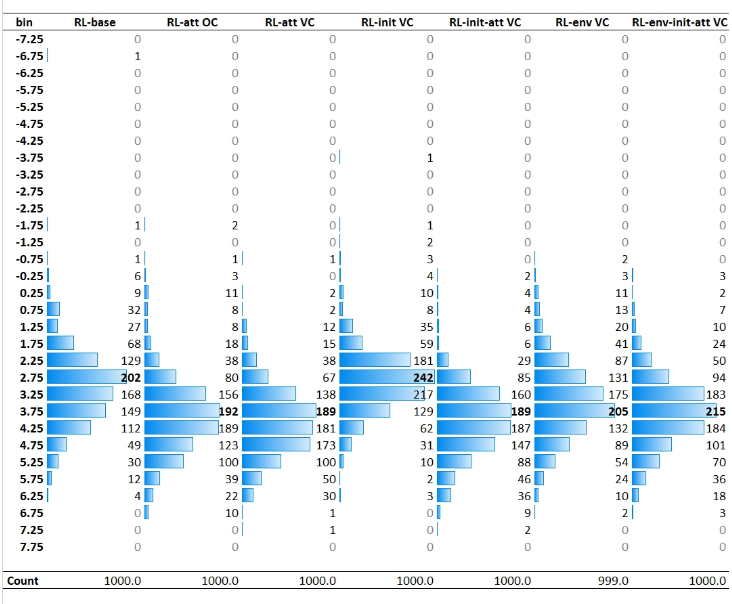
Table 2: Example of a German VC `RL-init` setup sentence with a negative lag, where the model tends to write more before reading new words.

`RL-env-init-att`. The attention norm is simply the average ℓ_2 norm between two consecutive attention time-steps. This can help in measuring the average visual attention per time step for a given sentence. We then compare the attention norm distributions over all the sentences in the English→German test 2016 set for the four different agent attention configurations. We present the result in Figure 5. Overall, `RL-init` and `RL-att` models are significantly more peaky than the `RL-env-init-att`. This suggests that `RL-env-init-att` model is generally spread across the 72 visual concepts more uniformly than the other two models. This perhaps is one of the causes for the slightly inferior performance of the model. We hypothesise that further regularisation of the attention distribution can ameliorate this behavior and leave it as future work.

6 Conclusion

In this paper we presented the first thorough exposition of multimodal reinforcement learning strategies for simultaneous machine translation. We demonstrate the efficacy of visual information and show that it leads to adaptive policies which substantially improve over the deterministic and unimodal RL baselines. Our empirical results indicate that both agent-side and environment-side visual information can be exploited to achieve higher quality translations with lower latency.

Throughout the experimental journey, we observed that the optimisation of simultaneous machine translation for dynamic policies is non-trivial, due to the two competing objectives: translation quality versus latency. For unimodal simultaneous machine translation, RL approaches tend to achieve translation quality on par with the quality of the deterministic policies within the same average lag. We believe that the fundamental issue is related



bin	RL-base	RL-att OC	RL-att VC	RL-init VC	RL-init-att VC	RL-env VC	RL-env-init-att VC
-7.25	0	0	0	0	0	0	0
-6.75	1	0	0	0	0	0	0
-6.25	0	0	0	0	0	0	0
-5.75	0	0	0	0	0	0	0
-5.25	0	0	0	0	0	0	0
-4.75	0	0	0	0	0	0	0
-4.25	0	0	0	0	0	0	0
-3.75	0	0	0	1	0	0	0
-3.25	0	0	0	0	0	0	0
-2.75	0	0	0	0	0	0	0
-2.25	0	0	0	0	0	0	0
-1.75	1	2	0	1	0	0	0
-1.25	0	0	0	2	0	0	0
-0.75	1	1	1	3	0	2	0
-0.25	6	3	0	4	2	3	3
0.25	9	11	2	10	4	11	2
0.75	32	8	2	8	4	13	7
1.25	27	8	12	35	6	20	10
1.75	68	18	15	59	6	41	24
2.25	129	38	38	181	29	87	50
2.75	**202**	80	67	**242**	85	131	94
3.25	168	156	138	217	160	175	183
3.75	149	**192**	**189**	129	**189**	**205**	**215**
4.25	112	189	181	62	187	132	184
4.75	49	123	173	31	147	89	101
5.25	30	100	100	10	88	54	70
5.75	12	39	50	2	46	24	36
6.25	4	22	30	3	36	10	18
6.75	0	10	1	0	9	2	3
7.25	0	0	1	0	2	0	0
7.75	0	0	0	0	0	0	0
Count	1000.0	1000.0	1000.0	1000.0	1000.0	999.0	1000.0

Figure 3: Histogram of per sentence lag values for test 2016 English-French. Y axis shows mean values per bin. Bold highlights modes for each distribution.

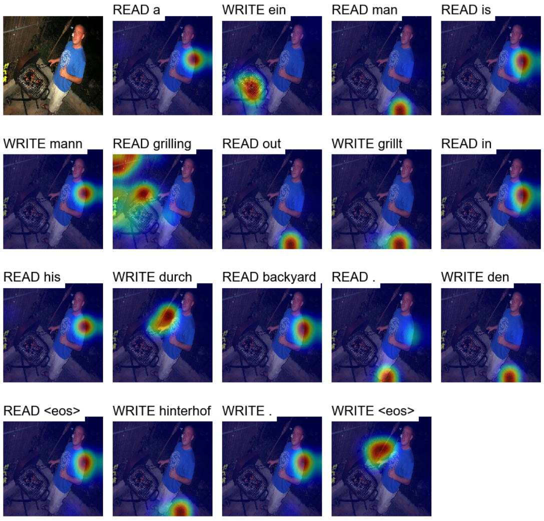


Figure 4: Visualisation of the agent attention and the corresponding actions over the source sentence from the test2016: 'A man is grilling out in his backyard.'

to the high variance of the estimator for sequence prediction, which increases sample complexity and impedes effective learning. On the other hand, the approaches with deterministic policies are simple and effective, as they are positively biased for language pairs that are close to each other. But the latter suffer from poor generalisation.

In the multimodal simultaneous machine translation setting, however, the variance of the estimator from RL models can be substantially reduced with to the presence of additional (visual) information.

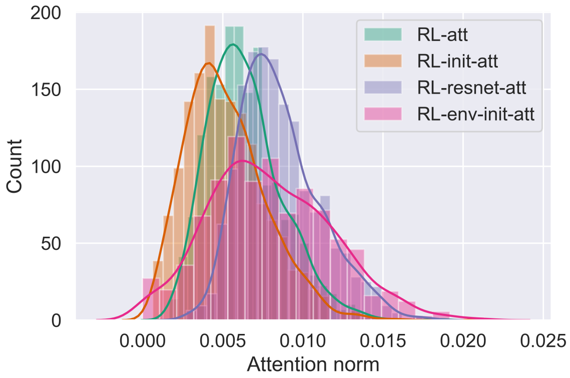


Figure 5: Distribution of attention norms for different agents with visual attention trained on the English→German dataset.

Acknowledgments

The authors thank the anonymous reviewers for their useful feedback. This work was supported by the MultiMT (H2020 ERC Starting Grant No. 678017) project. The work was also supported by the Air Force Office of Scientific Research (under award number FA8655-20-1-7006) project. Andy Mingren Li was supported by the Imperial College London UROP grant.

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Con-*

ference on Empirical Methods in Natural Language Processing, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

Jacob Andreas, Dan Klein, and Sergey Levine. 2017. Modular multitask reinforcement learning with policy sketches. In *International Conference on Machine Learning*, pages 166–175.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. Association for Computational Linguistics.

Philip Arthur, Trevor Cohn, and Gholamreza Haffari. 2020. Learning coupled policies for simultaneous machine translation. *arXiv preprint arXiv:2002.04306*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Computing Research Repository*, arXiv:1409.0473. Version 7.

Dzmitry Bahdanau, Felix Hill, Jan Leike, Edward Hughes, Arian Hosseini, Pushmeet Kohli, and Edward Grefenstette. 2018. Learning to understand goal specifications by modelling reward. *arXiv preprint arXiv:1806.01946*.

Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. 2018. Findings of the third shared task on multimodal machine translation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 304–323, Belgium, Brussels. Association for Computational Linguistics.

Ozan Caglayan, Walid Aransa, Adrien Bardet, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, Marc Masana, Luis Herranz, and Joost van de Weijer. 2017. LIUM-CVC submissions for WMT17 multimodal translation task. In *Proceedings of the Second Conference on Machine Translation*, pages 432–439, Copenhagen, Denmark. Association for Computational Linguistics.

Ozan Caglayan, Walid Aransa, Yaxing Wang, Marc Masana, Mercedes García-Martínez, Fethi Bougares, Loïc Barrault, and Joost van de Weijer. 2016. Does multimodality help human and machine for translation and image captioning? In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 627–633, Berlin, Germany. Association for Computational Linguistics.

Ozan Caglayan, Julia Ive, Veneta Haralampieva, Pranava Madhyastha, Loïc Barrault, and Lucia Specia. 2020. Simultaneous machine translation with visual context. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2350–2361, Online. Association for Computational Linguistics.

Iacer Calixto, Desmond Elliott, and Stella Frank. 2016. DCU-UvA multimodal MT system report. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 634–638, Berlin, Germany. Association for Computational Linguistics.

Iacer Calixto and Qun Liu. 2017. Incorporating global visual features into attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 992–1003, Copenhagen, Denmark. Association for Computational Linguistics.

Iacer Calixto, Qun Liu, and Nick Campbell. 2017. Doubly-attentive decoder for multi-modal neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1913–1924, Vancouver, Canada. Association for Computational Linguistics.

Iacer Calixto, Miguel Rios, and Wilker Aziz. 2019. Latent variable model for multi-modal translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6392–6405, Florence, Italy. Association for Computational Linguistics.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

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

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA. Association for Computational Linguistics.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, pages 41–75. Springer.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 493–499, New Orleans, Louisiana. Association for Computational Linguistics.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2054–2063.

J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the second shared task on multimodal machine translation and multilingual image description. In *Proceedings of the Second Conference on Machine Translation*, pages 215–233, Copenhagen, Denmark. Association for Computational Linguistics.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.

Desmond Elliott and Ákos Kádár. 2017. Imagination improves multimodal translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 130–141, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Prasoon Goyal, Scott Niekum, and Raymond J. Mooney. 2019. Using natural language for reward shaping in reinforcement learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 2385–2391. International Joint Conferences on Artificial Intelligence Organization.

Alvin Grissom II, Naho Orita, and Jordan Boyd-Graber. 2016. Incremental prediction of sentence-final verbs: Humans versus machines. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 95–104, Berlin, Germany. Association for Computational Linguistics.

Stig-Arne Grönroos, Benoit Huet, Mikko Kurimo, Jorma Laaksonen, Bernard Merialdo, Phu Pham, Mats Sjöberg, Umut Sulubacak, Jörg Tiedemann, Raphael Troncy, and Raúl Vázquez. 2018. The MeMAD submission to the WMT18 multimodal translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 603–611, Belgium, Brussels. Association for Computational Linguistics.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Julian Hitschler, Shigehiko Schamoni, and Stefan Riezler. 2016. Multimodal pivots for image caption translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2399–2409, Berlin, Germany. Association for Computational Linguistics.

Po-Yao Huang, Frederick Liu, Sz-Rung Shiang, Jean Oh, and Chris Dyer. 2016. Attention-based multimodal neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 639–645, Berlin, Germany. Association for Computational Linguistics.

Aizhan Imankulova, Masahiro Kaneko, Tosho Hirasawa, and Mamoru Komachi. 2020. Towards multimodal simultaneous neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 594–603, Online. Association for Computational Linguistics.

Julia Ive, Pranava Madhyastha, and Lucia Specia. 2019. Distilling translations with visual awareness. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6525–6538, Florence, Italy. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In

Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vision*, 123(1):32–73.

Jindřich Libovický and Jindřich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 196–202, Vancouver, Canada. Association for Computational Linguistics.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. In *Proc. Interspeech 2018*, pages 1293–1297.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. 2017. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 290–298.

Koichiro Ryu, Shigeki Matsubara, and Yasuyoshi Inagaki. 2006. Simultaneous English-Japanese spoken language translation based on incremental dependency parsing and transfer. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 683–690, Sydney, Australia. Association for Computational Linguistics.

Lucia Specia, Stella Frank, Khalil Sima'an, and Desmond Elliott. 2016. A shared task on multimodal machine translation and crosslingual image description. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 543–553, Berlin, Germany. Association for Computational Linguistics.

Umut Sulubacak, Ozan Caglayan, Stig-Arne Grönroos, Aku Rouhe, Desmond Elliott, Lucia Specia, and Jörg Tiedemann. 2020. Multimodal machine translation through visuals and speech. *Machine Translation*.

Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. 2018. Video captioning via hierarchical reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4213–4222.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simpler and faster learning of adaptive policies for simultaneous translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1349–1354, Hong Kong, China. Association for Computational Linguistics.

ement [R]epresentation for selecting relevant mathematical premises]

STAR: Cross-modal STAtement Representation for selecting relevant mathematical premises

Deborah Ferreira[1], **André Freitas**[1,2]
[1]The University of Manchester, Department of Computer Science, Manchester, UK
[2]Idiap Research Institute, Switzerland
{deborah.ferreira, andre.freitas}@manchester.ac.uk

Abstract

Mathematical statements written in natural language are usually composed of two different modalities: mathematical elements and natural language. These two modalities have several distinct linguistic and semantic properties. State-of-the-art representation techniques have demonstrated an inability in capturing such an entangled style of discourse. In this work, we propose STAR, a model that uses cross-modal attention to learn how to represent mathematical text for the task of Natural Language Premise Selection. This task uses conjectures written in both natural and mathematical language to recommend premises that most likely will be relevant to prove a particular statement. We found that STAR not only outperforms baselines that do not distinguish between natural language and mathematical elements, but it also achieves better performance than state-of-the-art models.

1 Introduction

Natural language understanding has been applied to several different tasks and areas, from question answering to visual grounding. Even though Mathematics is a well-established field with immense importance for most areas of science, applications of NLP in this field are still limited.

Natural language premise selection (NLPS) (Ferreira and Freitas, 2020a) is a task that requires the combination of natural language reasoning and mathematical reasoning. Given a certain conjecture (a mathematical statement written in natural language) that needs to be proven, we attempt to recommend useful premises that can be relevant for developing that mathematical argument.

Mathematical statements have a particular discourse structure that makes it challenging to use traditional NLP techniques. Some of its distinctive features are: (1) Entangled dual lexical spaces for the mathematical elements (ME) and natural language (NL); (2) Distinct syntactic phenomena between ME and NL.

Given this entangled nature of the discourse, where two very different linguistic modalities coexist in the same text, traditional information retrieval approaches are not able to capture the different semantics for each modality (Greiner-Petter et al., 2019). For example, in the mathematical domain, variables are represented using generic symbols; this lexical layer does not necessarily ground the semantics of the variables. The context surrounding the variables is more important than the symbol itself. When interpreting mathematical discourse, such particulars need to be taken into account.

In this work, we propose STAR, a cross-modal representation for mathematical statements for addressing the task of premise selection. In order to interpret the different modalities in the mathematical discourse (natural language and equational), STAR uses two different self-attention layers, one focused on the mathematical elements, such as expressions and variables, while the other attends to natural language features. STAR is taught to see these tokens as parts of different languages, the mathematical language and the English Language, similar to what our human brain does (Butterworth, 2002). Even though the brain interprets mathematics as a language, it requires different parts for processing it (Amalric and Dehaene, 2016). Using different attention layers, STAR can learn that understanding mathematics requires a different type of reasoning than natural language, approximating the behaviour of the brain when faced with mathematical tokens.

The approach presented in this work is based on the hypothesis that the use of cross-modal attention-based mechanisms provides a better encoding of the semantic content of mathematical statements

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3234–3243
April 19 - 23, 2021.

for the task of premise selection.

The contributions of this work can be summarised as follows:

- Proposal of a novel cross-modal embedding that captures the different modalities inside mathematical text: mathematical elements (expressions) and words.
- A systematic analysis of the transferability of this representation across different mathematical domains.
- An empirical evaluation, comparing our approach with state-of-the-art models and the performance of supporting ablation studies.
- We demonstrate an improvement of up to 70.34% in F1-Score, compared to a baseline that does not distinguish between mathematical elements and natural language. We also obtain competitive results with state-of-the-art approaches, using a smaller model and no pre-training.

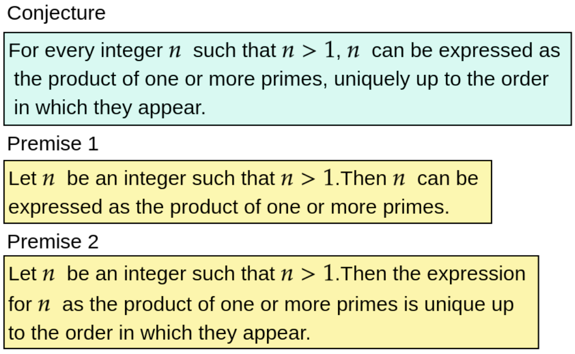
2 Background: Natural Language Premise Selection

In this work, we address the problem of *Natural Language Premise Selection* (Ferreira and Freitas, 2020a) (premise selection or NLPS). A mathematical statement can be a definition, an axiom, a theorem, a lemma, a corollary or a conjecture. Premises are composed of universal truths and accepted truths. Definitions and axioms are universal truths since the mathematical community accepts them without requiring proof.

On the other hand, accepted truths include statements that need proof before being adopted. Theorems, lemmas and corollaries are such types of statements. These statements were, at some point, framed as a conjecture, before they were proven. As such, they can be grounded on past mathematical discoveries, referencing their own supporting premises. This network structure of known premises can be used as a foundation in order to predict new ones.

Given a new conjecture c, that requires a mathematical proof, and a collection of premises $P = \{p_1, p_2, \ldots, p_{N_p}\}$, with size N_p, the NLPS task aims to retrieve the premises that are most likely to be useful for proving c. Premises of accepted truth statements can also have a subset of premises $\tilde{P} \subseteq P$.

Figure 1 presents an example of a conjecture containing two premises. Both *Premise 1* and *Premise 2* can be used as part of the proof for this conjecture.



Conjecture

For every integer n such that $n > 1$, n can be expressed as the product of one or more primes, uniquely up to the order in which they appear.

Premise 1

Let n be an integer such that $n > 1$.Then n can be expressed as the product of one or more primes.

Premise 2

Let n be an integer such that $n > 1$.Then the expression for n as the product of one or more primes is unique up to the order in which they appear.

Figure 1: Example of a conjecture and its premises.

Similar to previous approaches (Irving et al., 2016; Ferreira and Freitas, 2020a), we formulate this problem as a pairwise relevance classification problem. Given a pair (c, p_i), we classify if p_i can be used for proving c. Our approach is built on top of a cross-modal representation for mathematical statements, as the following section presents.

3 Our Approach: Cross-modal STAtement Representation (STAR)

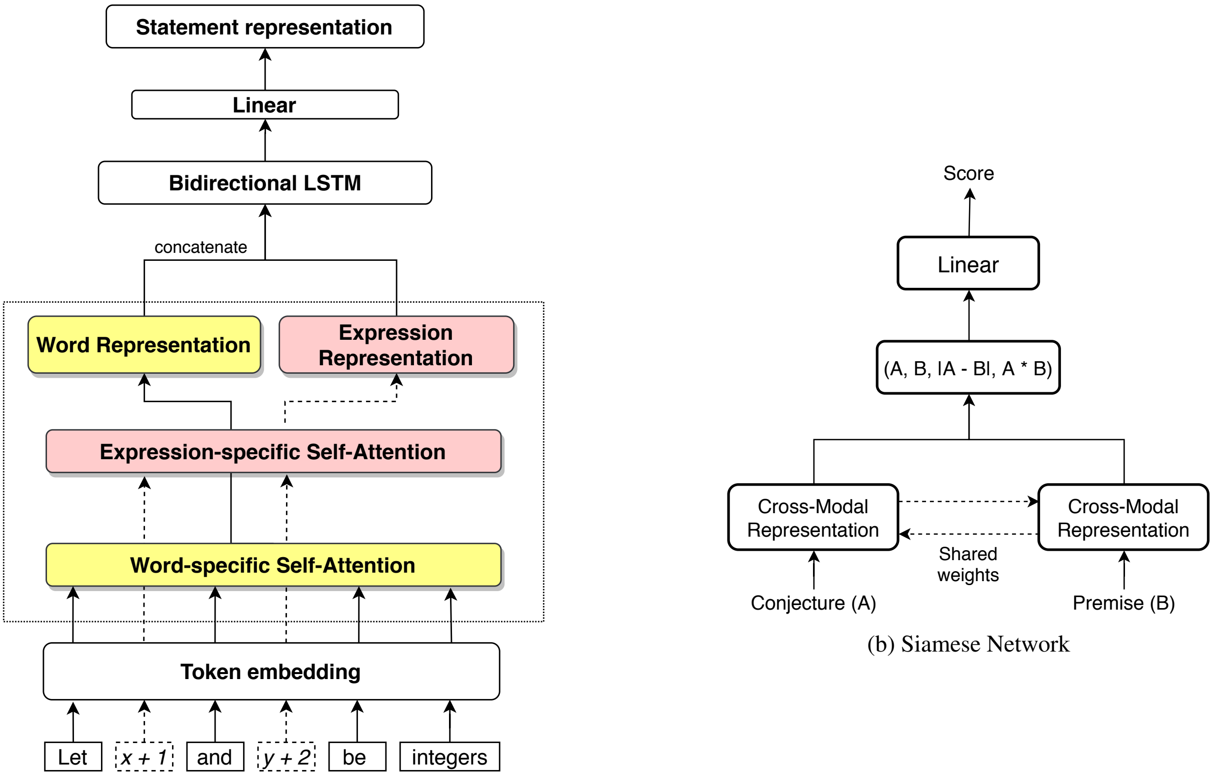
Mathematical language follows a regular pattern (in contrast to natural language) (Ganesalingam, 2013), regardless of representing a conjecture, universal truth or accepted truth. In this work, we consider mathematics written in natural language, instead of mathematics expressed in logical formal languages. The target corpus is composed of a combination of mathematical symbols and natural language words.

Given the set of mathematical statements $\mathcal{M}$ and a statement $m \in \mathcal{M}$, m is defined as a sequence of elements $m = \{s_1, s_2, \ldots, s_n\}$, where $s_i \in \mathcal{W}$, the set of words, or $s_i \in \mathcal{E}$, the set of mathematical elements present in $\mathcal{M}$. These components are situated in different lexical spaces; therefore, a function to generate a representation for m should take this into account.

We define an embedding model $\gamma : \mathcal{M} \mapsto \mathbb{R}^d$, where d is the dimension of the output vector. The complete architecture is presented in Figure 2a, where part of a statement is shown as an input example. Each layer is described in detail below.

3.1 Token embedding layer

The input to the embedding model is a mathematical statement. This embedding layer is a



(a) Embedding model.

(b) Siamese Network

Figure 2: Architecture for STAR. Figure (A) presents the model used to generate a representation for each statement, where we combine two self-attention layers, one for each modality of token. Figure (B) presents the Siamese Network used for classifying the conjecture-premise pair, based on the representations obtained.

$W_E \in \mathbb{R}^{k \times v}$ where k is the dimension of the word embeddings, and v is given by $|\mathcal{W}| + |\mathcal{E}|$.

3.2 Word/Expression-specific Self-Attention Layer

Research on the human brain has shown that there is no overlap between the parts of the brain that are activated in math-related tasks (both simple and complex) and sentence comprehension and general semantic knowledge tasks (Amalric and Dehaene, 2016). This behaviour hints at how we should map distinct representations to these different modalities of symbols and linguistic structures (maths and natural language).

Inspired by the behaviour of the human brain, we introduce two layers of self-attention (Vaswani et al., 2017), one for each modality, attempting to approximate human reasoning. One layer captures specific natural language linguistic features, while the other represents particular mathematical formalism features. Given a matrix of queries Q and matrices of keys and values K and V. The attention head is defined as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

where d_k is the dimension of the keys.

These attention heads compose a multi-head attention mechanism, defined as:

$$\begin{aligned} MultiHead(Q, K, V) = \\ Concat(head_1, \ldots, head_h)W^O \end{aligned} \quad (2)$$

where:
$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$
and W_i^Q W_i^K and W_i^V are parameter matrices. In order to apply self-attention, we consider Q, K and V as the same values, obtained using a linear layer on top of the output of the embedding layer. Words and expressions tokens have a very distinct nature, and we hypothesise that these two layers allow learning and representing these differences.

3.3 Long Short-Term Memory Layer

LSTM networks (Hochreiter and Schmidhuber, 1997) are a complex activation unit, based on a chain structure explicitly designed to capture long-term sequence dependencies. LSTM is an ideal candidate for treating sequential data such as mathematical statements. For the sake of brevity, we omit the description of this layer, as it is extensively described in the literature.

3.4 Training objective

Finally, in order to obtain the score between conjectures and premises, a siamese neural network setting is used (Figure 2b), where a pair of statements are simultaneously fed into two networks, with shared weights. This allows the model to learn the representation of each statement individually, while still being aware that the statements belong to the same semantic space.

The representation for each statement is obtained and combined, where the expected score is 1 if B is a premise to A, or 0 otherwise.

The used training objective function is the Cross Entropy Loss, defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N}\left[Y_n \log \hat{Y}_n + (1-Y_n)\log(1-\hat{Y}_n)\right] \tag{3}$$

where Y is the predicted classification and $\hat{Y}_i$ is the expected classification.

4 Experiments

This section presents the experiments performed to test our hypotheses. We use the dataset PS-ProofWiki (Ferreira and Freitas, 2020a) for these experiments. This dataset is composed of pairs of conjectures and premises, framing the problem as a pair classification task. Each statement is written using a combination of words and LaTeXnotation. For each positive pair, where the statement is a premise to the conjecture, there can be n number of negative pairs. For testing the robustness to noise in the proposed model, we use $n \in \{1, 2, 5, 10\}$. The number of entries for Train, Validation and Test for each value of n is shown in Table 1.

The negative pairs are obtained using two different methods. The first collects random examples of statements that are not premises to form a new pair (**negative examples**). In the second technique, we use BM-25 to retrieve statements that are lexically similar to the premises, but that are not part of positive pairs (**similar examples**). For these experiments, we used 512 as a the size of the hidden units layer in the LSTM, embedding size and output statement vector in the embedding architecture. We used 50 epochs for each training round. As shown in Figure 3, with this number of epochs we achieve convergence for all values of n. For each epoch, the validation set was evaluated, and the best model was chosen for testing.

n	**Train**	**Val**	**Test**
1	32,758	10,798	10,112
2	49,137	16,197	15,168
5	98,274	32,394	30,336
10	180,169	59,389	55,616

Table 1: Number of entries for Training, Validation and Test for different values of n.

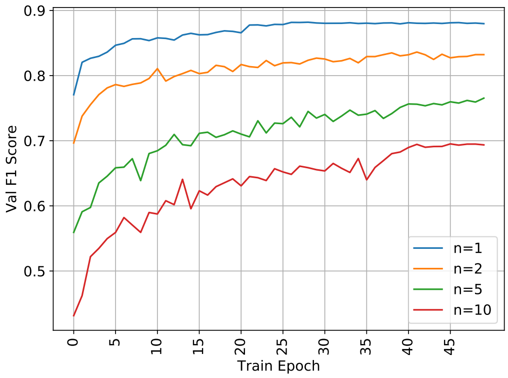


Figure 3: Number of training epochs and the obtained validation F1 score.

All experiments and data can be found in our Github repository[1].

4.1 Quantitative Analysis

In order to verify our hypothesis, we compare the proposed approach, i.e., using different self-attention layers for each modality (mathematical elements and natural language) with a modified model, using only one self-attention layer for all parts of the text. This modified model is obtained by replacing the layers inside the dotted rectangle from Figure 2a with a single self-attention layer. This modified model is referred here as **Self-attention + BiLSTM**.

---

[1] `http://github.com/ai-systems/crossmodal_embedding`

4.1.1 Results

Table 2 presents the results for the premise selection task using the random examples.

The aggregate scores obtained using STAR is consistently higher than the baseline. Even though there is an expected degradation in the score with the addition of more negative examples, STAR still outperforms the baseline in all cases, demonstrating robustness to noise. These results support our hypothesis that different modalities inside the mathematical text should be represented in different linguistic spaces.

Similarly, we re-run both models, but this time using the similar examples. The results can be found in Table 3.

We can notice that STAR precision decreases when compared with the results obtained using the random examples. However, once more, STAR outperforms the baseline for all values of n. The results of the baseline model do not change significantly from the previous result improving it in some cases. We hypothesise that this is due to the fact that the use of lexical similarity for the generation of similar examples does not provide reliable discriminators (due to the limited intrinsic semantics of variables). Variables can have the same lexical form across mathematical statements, without sharing the same meaning.

4.1.2 Transferring Knowledge across mathematical domains

Another targeted hypothesis is that STAR performs better than the baseline in the task of transferring knowledge between different mathematical domains. In order to verify this hypothesis, we train the baseline and our model using one topic and test it in a different one, the topics used are Abstract Algebra (AA), Topology (TP) and Set Theory (ST). Table 4 presents the number of statements for Train/Val/Test for each topic.

Table 5 shows the experimental results for the different mathematical topics. Initially, we expected that training using the largest dataset would allow both models to obtain the best performance. However, training using the Topology dataset topic did not achieve the highest results. This is likely because of the distinctive nature of its symbolic space, more focused on the properties of geometric objects. On the other hand, the best performing training and test dataset, Abstract Algebra, is heavily based on the algebraic notation that our model is capable of capture using cross-modal attention.

In terms of transferable knowledge, Set Theory is the tested dataset with the highest score, confirming the expectation that Set Theory is an important component of both Abstract Algebra and Topology, being an intrinsic part of the mathematical argumentation on these topics. Therefore, such knowledge is more natural to transport. Our proposed approach outperforms the baseline in all cases. However, both models see substantial performance degradation when trying to transfer the knowledge from one topic to another, indicating both the need for better abstractive mathematical models and an intrinsic domain-specificity mathematical inference.

4.1.3 Other baselines

In order to verify the model performance, we test our model against two state-of-the-art models. The first baseline is a Transformer-based model, BERT. We fine-tune BERT (Vaswani et al., 2017) using the same configuration as the one used for Natural Language Inference (Jiang and de Marneffe, 2019) since this task carries similarities with the premise selection task. The other baseline is MathSum (Yuan et al., 2019): an encoder-decoder model used to represent mathematical content found in online forums. We use only the encoder part of this model, together with the same siamese network as STAR and the same parameter configuration. The results can be found in Table 6.

Considering the F1-Score obtained, BERT was placed second in the test set evaluation. Even though BERT is not explicitly trained for the Mathematical domain, it presents an excellent performance for the premise selection task. BERT is a large-scale model that was also trained on sources containing mathematical notation, including latex notation, therefore it partially encodes mathematical notation. Our model outperforms BERT for the test set, even though it employs a significantly smaller set of parameters (5x less parameters) and is not pre-trained on a large corpus as BERT is.

4.2 Qualitative analysis

We present examples of predicted pairs in Table 7. When analysing the obtained classified pairs, we found that STAR not only can deal with heavily equational statements, such as the second pair from the table, but it can also handle statements that contain a high level of entanglement between mathematical and natural language terms, such as the first pair.

		Val			Test		
	n	**F1**	**P**	**R**	**F1**	**P**	**R**
STAR	1	**.885**	.854	.917	**.882**	.865	.899
	2	**.836**	.803	.871	**.829**	.793	.870
	5	**.765**	.693	.853	**.765**	.706	.835
	10	**.695**	.614	.799	**.684**	.603	.791
Self-att + LSTM	1	.651	.550	.796	.631	.573	.703
	2	.514	.406	.702	.514	.420	.663
	5	.493	.372	.728	.461	.344	.700
	10	.408	.283	.731	.406	.276	.766

Table 2: Comparison of STAR with a model containing a single self-attention layer. In this experiment, we test for a different number of random negative examples (n). The metrics used are F1-score (F1), precision (P) and recall (R).

		Val			Test		
	n	**F1**	**P**	**R**	**F1**	**P**	**R**
STAR	1	**.798**	.725	.886	**.793**	.723	.879
	2	**.716**	.624	.840	**.707**	.593	.875
	5	**.620**	.485	.857	**.626**	.493	.854
	10	**.546**	.412	.809	**.528**	.387	.834
Self-att + BiLSTM	1	.648	.561	.767	.538	.699	.437
	2	.537	.444	.679	.540	.448	.678
	5	.389	.261	.760	.379	.251	.773
	10	.289	.179	.759	.286	.174	.799

Table 3: Results for STAR and baseline for different number of negative examples (n) using similar examples.

Topic	**Train**	**Val**	**Test**
AA	2,246	633	580
ST	1,897	618	590
TP	2,539	810	788

Table 4: Distribution of dataset wih different topics.

However, we found that STAR can sometimes struggle with variable names. For example, in pair 3, the variable T appears several times. STAR infers that this implies that there is a relation between both statements. The relationship exists since both statements refer to the concept *spaces*; however, this does not define a dependency relationship. This result provides evidence for the need of an architecture which better captures variable semantics.

Figure 4 presents a comparison of our model with the single attention model. This graph shows the percentage of mathematical elements in the statement versus the percentage of the statements in the dataset that the model was able to predict correctly.

STAR has an consistent performance throughout different distributions of mathematical and natural language terms. Such results demonstrate a need of an attention layer for each term modality. On the other hand, we can observe that the baseline struggles to predict statements that are mostly mathematical (right-end of the graph), finding it easier to predict the statements which have the prevalence of natural language terms (left-end of the graph). The results show that our model is better suitable for dealing with this type of entangled text.

5 Related Work

Several areas of research apply Natural Language Processing for domain-specific tasks, Mathematics being one of these areas. One crucial task in this field is solving mathematical word problems, where the goal is to provide the answer to a mathe-

Topic		STAR			Self-att + BiLSTM		
Train	**Test**	**F1**	**P**	**R**	**F1**	**P**	**R**
AA	AA	**.862**	.823	.906	.629	.581	.684
TP	TP	**.752**	.692	.825	.722	.680	.769
ST	ST	**.787**	.763	.813	.613	.654	.578
AA	ST	**.662**	.595	.747	.627	.595	.664
AA	TP	**.595**	.520	.693	.570	.539	.605
TP	AA	**.654**	.536	.836	.649	.602	.704
TP	ST	**.673**	.588	.787	.628	.561	.714
ST	TP	**.535**	.539	.531	.578	.535	.627
ST	AA	**.644**	.598	.697	.625	.591	.663

Table 5: Testing how different mathematical areas are transportable to other areas. The areas considered here are Abstract Algebra(AA), Topology (TP) and Set Theory (ST). For these experiments, we use random examples with $n = 1$.

	Val			Test		
	F1	**P**	**R**	**F1**	**P**	**R**
BERT	**.886**	.871	.901	.877	.925	.834
MathSum	.644	.512	.869	.459	.562	.388
Self-attention + BiLSTM	.651	.550	.796	.631	.573	.703
STAR	.885	.854	.917	**.882**	.865	.899

Table 6: Comparison of our model with other baselines, using n=1 and random examples.

matical problem written in natural language (Zhang et al., 2020; Kushman et al., 2014; Ran et al., 2019). These problems are usually self-contained and are structured in a didactic and straightforward manner, not containing complex mathematical expressions.

Some contributions focus on the representation of mathematical text and mathematical elements. Zinn (2004) proposes a representation for mathematical proofs using Discourse Representation Theory. Similarly, Ganesalingam (2013) introduces a grammar for representing informal mathematical text, while Pease et al. (2017) presents this style of text using Argumentation Theory. Such explicit representations are relevant for representing the reasoning process behind mathematical thinking. However, it is still not possible to accurately extract these representations at scale.

Representations of mathematical elements are often used in the context of Mathematical Information Retrieval, used, for example, for obtaining a particular equation or expression, given a specific query. Tangent-CFT (Mansouri et al., 2019) is an embedding model that uses the subparts an expression or equation, to represent its meaning. This type of representation (Fraser et al., 2018; Zanibbi et al., 2016) often removes the expression for its original discourse, losing the textual context that can help to find a semantic representation. In this work, we focus on creating a representation that can integrate both of these aspects, natural language and mathematical elements. Similar to our work, Yuan et al. (2019) uses self-attention for mathematical elements in order to generate headlines for mathematical questions. Other relevant tasks for NLP applied to Mathematics include typing variables according to its surrounding text (Stathopoulos et al., 2018), obtaining the units of mathematical elements (Schubotz et al., 2016) and generating equations on a given topic (Yasunaga and Lafferty, 2019).

Premise selection is a well-defined task in the field of Automated Theorem Proving (ATP), where proofs are encoded using a formal logical representation. Given a set of premises P, and a new conjecture c, premise selection aims to predict those premises from P that will most likely lead to an

Conjecture	Premise	Predicted	Label
Let $T = (S, \tau)$ be a topological space. Let A, B be subsets of S. Then: $\partial(A \cap B) \subseteq \partial A \cup \partial B$ where ∂A denotes the boundary of A.	Let S, T_1, T_2 be sets such that T_1, T_2 are both subsets of S. Then, using the notation of the relative complement: $ST_1 \cap T_2 = ST_1 \cup ST_2$	1	1 ✓
$\int \frac{\dot{X}}{x(x^2-a^2)} = \frac{1}{2a^2}, \ln \frac{x^2-a^2}{x^2} + C$ for $x^2 > a^2$.	$\int \frac{dx}{x} = \ln x + C$ for $x \neq 0$.	1	1 ✓
Let $T = S, \tau$ be a compact space. Then T is countably compact.	Let $T = (S, \tau_{a,b})$ be a modified Fort space. Then T is not a T_3 space, T_4 space or T_5 space.	1	0 ✗

Table 7: Some of the premises existing in the dataset, together with the predictions from STAR.

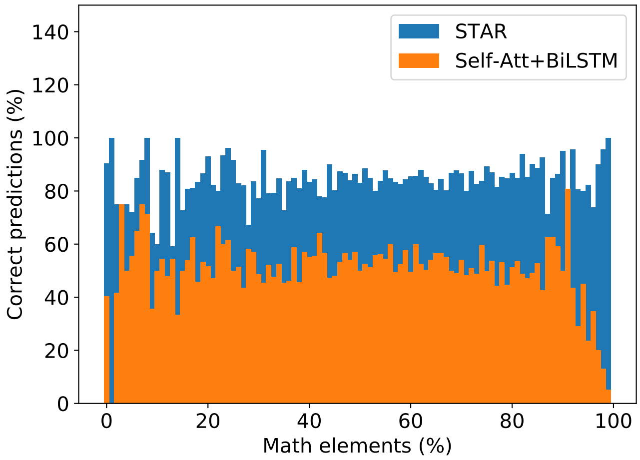


Figure 4: Comparison of our model and the baseline on the capability of predicting correctly statements with different levels of entanglement.

automatically constructed proof of c, where P and c are both written using a formal language. (Irving et al., 2016) is one of the first models to use Deep Learning for premise selection in ATPs.

Ferreira and Freitas (2020a) proposed an adaptation of this task, focusing on mathematical text written in natural language. A model based on Graph Neural Networks has been previously introduced for this task (Ferreira and Freitas, 2020b), however, the authors do not take into account the differences between mathematical and natural language terms, representing all statements homogeneously. The premise selection task can also be seen as an explanation reconstruction task, where premises are considered explanations for mathematical proofs. Approaches for dealing with such type of challenge in the science domain include unification retrieval (Valentino et al., 2020b,a) and abductive reasoning (Thayaparan et al., 2020).

In this work, we propose a new representation that distinctively captures both language modalities present in the mathematical discourse in order to solve the premise selection task.

6 Conclusion

In this work, we introduced STAR, a model to represent mathematical statements for the task Natural Language Premise Selection. In this model, we used two layers of self-attention, one for each language modality present in the mathematical text.

In order to test STAR's ability to capture the different aspects of each modality, verifying if it can interpret that expressions and words belong to different lexical spaces, we compared our performance with other baselines. We found that having one layer for each modality significantly increases the performance for premise selection. We also compared our approach with state-of-the-art models and found that STAR achieves the highest results for the Test set. STAR was also tested for transfer learning, revealing that cross-modal attention improves the transportability between different mathematical areas.

However, we discovered that STAR is still limited regarding variable modelling. There is still a gap in how to handle variable typing in latent models, considering its meaning instead of its lexical symbol. As future work, this issue will be addressed using latent representations trained specifically for variable modelling.

References

Marie Amalric and Stanislas Dehaene. 2016. Origins of the brain networks for advanced mathematics in expert mathematicians. *Proceedings of the National Academy of Sciences*, 113(18):4909–4917.

Brian Butterworth. 2002. Mathematics and the brain. *Opening address to the Mathematical Association, Reading*.

Deborah Ferreira and André Freitas. 2020a. Natural language premise selection: Finding supporting statements for mathematical text. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2175–2182.

Deborah Ferreira and André Freitas. 2020b. Premise selection in natural language mathematical texts. In *Proceedings of The 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Dallas Fraser, Andrew Kane, and Frank Wm Tompa. 2018. Choosing math features for bm25 ranking with tangent-l. In *Proceedings of the ACM Symposium on Document Engineering 2018*, pages 1–10.

Mohan Ganesalingam. 2013. The language of mathematics. In *The Language of Mathematics*, pages 17–38. Springer.

André Greiner-Petter, Terry Ruas, Moritz Schubotz, Akiko Aizawa, William Grosky, and Bela Gipp. 2019. Why machines cannot learn mathematics, yet. *arXiv preprint arXiv:1905.08359*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Geoffrey Irving, Christian Szegedy, Alexander A Alemi, Niklas Eén, François Chollet, and Josef Urban. 2016. Deepmath-deep sequence models for premise selection. In *Advances in Neural Information Processing Systems*, pages 2235–2243.

Nanjiang Jiang and Marie-Catherine de Marneffe. 2019. Evaluating bert for natural language inference: A case study on the commitmentbank. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6088–6093.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281.

Behrooz Mansouri, Shaurya Rohatgi, Douglas W Oard, Jian Wu, C Lee Giles, and Richard Zanibbi. 2019. Tangent-cft: An embedding model for mathematical formulas. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 11–18.

Alison Pease, John Lawrence, Katarzyna Budzynska, Joseph Corneli, and Chris Reed. 2017. Lakatos-style collaborative mathematics through dialectical, structured and abstract argumentation. *Artificial Intelligence*, 246:181–219.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. Numnet: Machine reading comprehension with numerical reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2474–2484.

Moritz Schubotz, David Veenhuis, and Howard S Cohl. 2016. Getting the units right. In *FM4M/MathUI/ThEdu/DP/WIP@ CIKM*, pages 146–156.

Yiannos Stathopoulos, Simon Baker, Marek Rei, and Simone Teufel. 2018. Variable typing: Assigning meaning to variables in mathematical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 303–312.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2020. Explanationlp: Abductive reasoning for explainable science question answering. *arXiv preprint arXiv:2010.13128*.

Marco Valentino, Mokanarangan Thayaparan, and André Freitas. 2020a. Explainable natural language reasoning via conceptual unification. *arXiv preprint arXiv:2009.14539*.

Marco Valentino, Mokanarangan Thayaparan, and André Freitas. 2020b. Unification-based reconstruction of explanations for science questions. *arXiv preprint arXiv:2004.00061*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Michihiro Yasunaga and John D Lafferty. 2019. Topiceq: A joint topic and mathematical equation model for scientific texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7394–7401.

Ke Yuan, Dafang He, Zhuoren Jiang, Liangcai Gao, Zhi Tang, and C Lee Giles. 2019. Automatic generation of headlines for online math questions. *arXiv preprint arXiv:1912.00839*.

Richard Zanibbi, Kenny Davila, Andrew Kane, and Frank Wm Tompa. 2016. Multi-stage math formula search: Using appearance-based similarity metrics at scale. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 145–154.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937.

Claus Zinn. 2004. Understanding informal mathematical discourse. *PhD thesis, Institut fur Informatik, Universitat Erlangen-Nurnberg*.

Do Multi-Hop Question Answering Systems Know How to Answer the Single-Hop Sub-Questions?

Yixuan Tang **Hwee Tou Ng** **Anthony K.H. Tung**

Department of Computer Science
National University of Singapore
{yixuan, nght, atung}@comp.nus.edu.sg

Abstract

Multi-hop question answering (QA) requires a model to retrieve and integrate information from multiple passages to answer a question. Rapid progress has been made on multi-hop QA systems with regard to standard evaluation metrics, including EM and F1. However, by simply evaluating the correctness of the answers, it is unclear to what extent these systems have learned the ability to perform multi-hop reasoning. In this paper, we propose an additional sub-question evaluation for the multi-hop QA dataset HotpotQA, in order to shed some light on explaining the reasoning process of QA systems in answering complex questions. We adopt a neural decomposition model to generate sub-questions for a multi-hop question, followed by extracting the corresponding sub-answers. Contrary to our expectation, multiple state-of-the-art multi-hop QA models fail to answer a large portion of sub-questions, although the corresponding multi-hop questions are correctly answered. Our work takes a step forward towards building a more explainable multi-hop QA system.

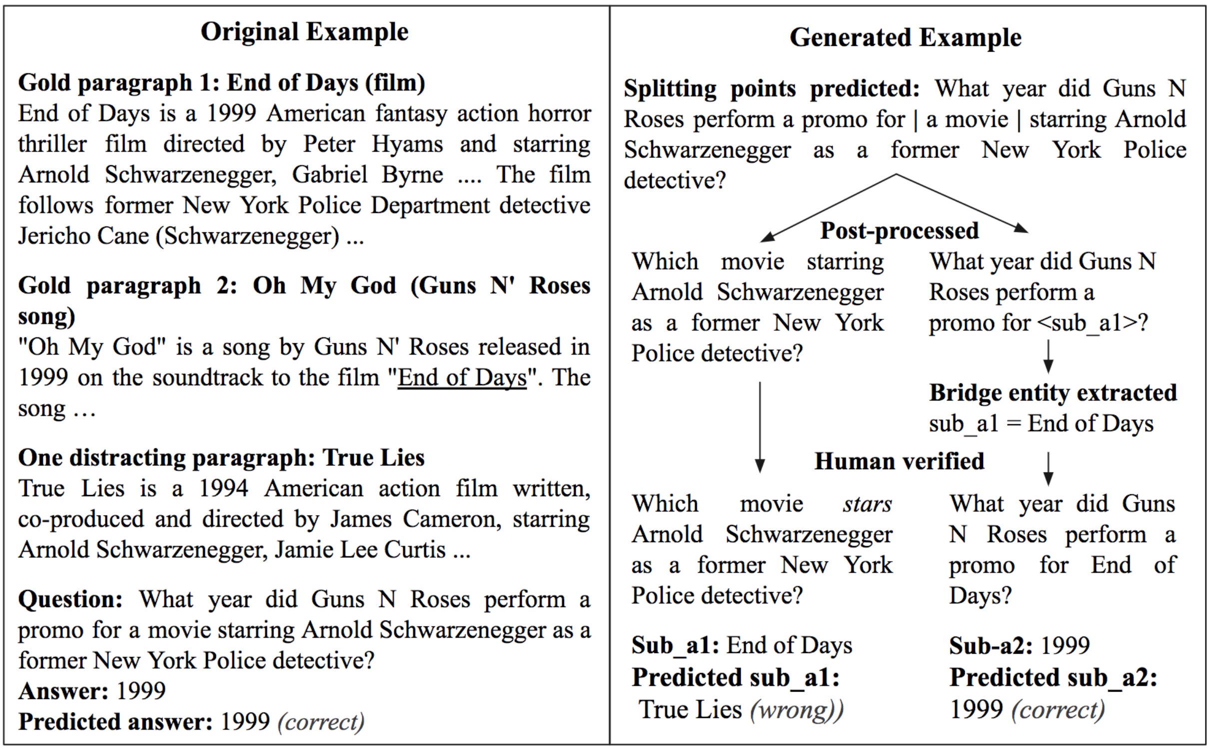
1 Introduction

Rapid progress has been made in the field of question answering (QA), thanks to the release of many large-scale, high-quality QA datasets. Early datasets (Hermann et al., 2015; Rajpurkar et al., 2016, 2018; Trischler et al., 2017; Joshi et al., 2017) mainly consist of single-hop questions, where an answer with supporting justification can be found within a short segment of text. These benchmarks focus on evaluating QA models' ability to perform local pattern matching between a passage and a question. Existing models (Lan et al., 2020; Zhang et al., 2020) have achieved super-human performance. Recently, multi-hop QA datasets (Khashabi et al., 2018; Welbl et al., 2018; Yang et al., 2018) have gained increasing attention. They require models to retrieve multiple pieces of supporting evidence from different documents and to reason over the evidence collected to answer a question. The standard evaluation metrics of QA datasets include exact match (EM) and F1 scores averaged over the test set. HotpotQA (Yang et al., 2018) also provides sentence-level supporting facts required for reasoning. However, providing supporting sentences is not sufficient for us to interpret the choice of an answer for end-to-end complex QA systems. It is unclear whether the systems have performed the desired multi-hop reasoning to reach the correct answer.

In this work, we propose an additional evaluation scheme to test multi-hop QA systems' performance on answering the single-hop sub-questions of a multi-hop question. When designing a multi-hop question, we expect it to require QA models to retrieve a chain of sentences as evidence and then reasoning over them to answer the question. Evaluating QA models on sub-questions helps us to understand their behavior on each hop of the reasoning process. In addition, it evaluates whether multi-hop QA models can generalize well on simpler questions. Figure 1 presents an illustrating example. A successful complex QA model should be able to answer the two sub-questions "*Which movie stars Arnold Schwarzenegger as a former New York Police detective*" and "*What year did Guns N Roses perform a promo for End of Days*" if it understands the underlying reasoning process for the original multi-hop question.

We focus on the HotpotQA (Yang et al., 2018) dataset under the distractor setting, in which multi-hop questions are asked over several Wikipedia paragraphs. We create the evaluation dataset by generating the sub-questions and then extracting their answers automatically. The candidate sub-questions and intermediate answers are then manually verified, which results in 1,000 sub-question

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3244–3249
April 19 - 23, 2021.



Original Example

Gold paragraph 1: End of Days (film)
End of Days is a 1999 American fantasy action horror thriller film directed by Peter Hyams and starring Arnold Schwarzenegger, Gabriel Byrne .... The film follows former New York Police Department detective Jericho Cane (Schwarzenegger) ...

Gold paragraph 2: Oh My God (Guns N' Roses song)
"Oh My God" is a song by Guns N' Roses released in 1999 on the soundtrack to the film "End of Days". The song …

One distracting paragraph: True Lies
True Lies is a 1994 American action film written, co-produced and directed by James Cameron, starring Arnold Schwarzenegger, Jamie Lee Curtis ...

Question: What year did Guns N Roses perform a promo for a movie starring Arnold Schwarzenegger as a former New York Police detective?
Answer: 1999
Predicted answer: 1999 *(correct)*

Generated Example

Splitting points predicted: What year did Guns N Roses perform a promo for | a movie | starring Arnold Schwarzenegger as a former New York Police detective?

Which movie starring Arnold Schwarzenegger as a former New York Police detective?

What year did Guns N Roses perform a promo for <sub_a1>?

Which movie *stars* Arnold Schwarzenegger as a former New York Police detective?

What year did Guns N Roses perform a promo for End of Days?

Sub_a1: End of Days
Predicted sub_a1: True Lies *(wrong))*

Sub-a2: 1999
Predicted sub_a2: 1999 *(correct)*

Figure 1: An illustrating example from the HotpotQA dataset in the distractor setting, with our construction procedure to generate an evaluation example. We only show one out of eight distracting paragraphs provided in the context due to paper length constraint.

evaluation examples. It is surprising to find that all three top-performing models which we experiment with fail to answer a large portion of sub-questions (49.8% to 60.4%), although their corresponding multi-hop questions are correctly answered.

Previous work has investigated the necessity of multi-hop reasoning on HotpotQA dataset. Jiang and Bansal (2019) construct distracting paragraphs adversarially to demonstrate that models learn to exploit reasoning shortcuts to locate the answer rather than performing multi-hop reasoning. Chen and Durrett (2019) show that a sentence-factored model can solve a large number of questions in HotpotQA, suggesting multi-hop reasoning is not really needed. Min et al. (2019a) also achieve similar result using a single-hop BERT-based model. Our sub-question evaluation is complementary to these approaches. While existing work shows the lack of multi-hop reasoning by limiting or adding text input to QA models, we provide sub-questions and intermediate answers explicitly to interpret model behavior on each hop of the reasoning process. It can be used as a complementary metric to ensure that models which can correctly answer both intermediate sub-questions and the final multi-hop question actually go through the reasoning steps as desired. Our work takes a step forward towards building a more explainable multi-hop QA system.

2 Construction of Evaluation Examples

In this section, we introduce our semi-automatic approach to generate two sub-questions and their corresponding answers for multi-hop questions from the HotpotQA dataset. As shown in Figure 1, the evaluation examples are generated in three steps. First, we decompose each source question into several sub-strings by predicting the breaking points and post-process them to generate two sub-questions. Then, the answers for the sub-questions are extracted from the paragraphs using some heuristics. Lastly, the candidate evaluation examples generated are sent for human verification. We first introduce the HotpotQA dataset and then elaborate on each step of the construction pipeline.

2.1 HotpotQA

HotpotQA contains 113K crowd-sourced multi-hop QA pairs on Wikipedia articles. We focus on bridge-type questions that actually require multiple steps of reasoning under the distractor setting. During the construction of such an example in HotpotQA, two related paragraphs p_{gold1}, p_{gold2}

from different Wikipedia articles titled t_{gold1}, t_{gold2} are presented to crowd-workers. The two paragraphs are related since the text content in one paragraph contains the title entity of the other paragraph. This shared title entity is referred to as the bridge entity. Using Figure 1 as an example, the second paragraph about *Oh My God* contains the title entity of the first paragraph, *End of Days* (underlined). Thus, *End of Days* is referred as the bridge entity. The crowd-workers are encouraged to ask a multi-hop question using both paragraphs and to annotate the supporting sentences which help to determine the answer. Then, eight other related distracting paragraphs are retrieved from Wikipedia and mixed with the two gold paragraphs to serve as the context for the question. Given an example $E = \{C, q, a\}$ from HotpotQA, we aim to generate an evaluation example $E' = \{C, q, a, sub\_q_1, sub\_a_1, sub\_q_2, sub\_a_2\}$, where sub_q_1 and sub_q_2 are the two sub-questions, and sub_a_1 and sub_a_2 are their corresponding answers.

2.2 Sub-Question Generation

Given a multi-hop question, the first step is to decompose it into sub-questions. We adopt the model introduced in DecompRC (Min et al., 2019b) to generate the sub-questions using a copying and editing mechanism. The multi-hop question is first converted into BERT word embeddings (Devlin et al., 2019), and then sent to a fully connected neural network to predict the splitting points. It is trained on 400 annotated examples. The separated text spans are post-processed to form the two sub-questions, following a set of handcrafted rules.

2.3 Intermediate Answer Extraction

One particular characteristic of bridge-type questions from HotpotQA is that the two gold paragraphs are linked by a bridge entity. Since the crowd-workers are required to form a multi-hop question which makes use of information from both paragraphs, there is a high probability that the bridge entity is the answer to the first sub-question. For the example shown in Figure 1, *End of Days* in gold paragraph 2 is the bridge entity. It is also the intermediate answer for the multi-hop question, i.e., the answer for the first sub-question.

Three different situations are considered in order to extract the bridge entity. First, if the title entity E_A of paragraph A occurs in the other paragraph B, while the title entity E_B of B does not occur in A, then E_A is recognized as the bridge entity. Second, if neither E_A nor E_B is contained in the other paragraph, then the title entity with more overlapping text with the other paragraph is chosen as the bridge entity (since sometimes the alias of the Wikipedia title is used in the paragraph). Lastly, if both E_A and E_B appear in the other paragraph, then the title entity which does not appear in both the question and the answer is chosen as the bridge entity, since an entity mentioned in the multi-hop question or included in the final answer is unlikely to be the bridge entity. The bridge entity is set to be unidentified if none or both of the title entities satisfy at least one of the requirements. As illustrated in Figure 1, once the bridge entity is retrieved, the blank in the second sub-question will be updated. The answer to the second sub-question should be the same as the original multi-hop question.

Case	Gold Answer	Predicted Answer
1	from 1986 to 2013	1986 to 2013
2	City of Angles (film)	City of Angles
3	Mondelez International, Inc.	the company Mondelez International

Table 1: Examples of partially matched answer string pairs.

Model	q		q_{sub1}		q_{sub2}	
	EM	F1	EM	F1	EM	F1
DFGN	58.1	71.96	54.6	68.54	49.3	60.83
DecompRC	63.1	77.61	61	75.21	56.8	70.77
CogQA	53.2	67.82	58.6	69.65	54	68.49

Table 2: EM and F1 scores of models on 1,000 human-verified sub-question evaluation examples.

2.4 Human Verification

Sub-question generation and intermediate answer extraction help to efficiently generate candidate sub-questions and their answers. To ensure the quality of the evaluation dataset, the examples generated are manually verified. For each example, we present to an annotator the original multi-hop question, the answer, two sub-questions generated and their answers, and two gold paragraphs. Questions that actually do not require multi-hop reasoning or with the wrong answer (due to wrong annotation by the HotpotQA crowd workers) are first filtered out. Then, the annotator is required to review whether sub_q_1 and sub_q_2 are two syntactically and semantically correct sub-questions of q and whether sub_a_1 and sub_a_2 are valid and to correct them if not. In total, a sample of 1,000 examples generated for the HotpotQA development set are manually verified for use in our evaluation[1].

3 Experiments and Results

In order to interpret the behavior of existing models on each hop of the reasoning process required for multi-hop questions and to determine their ability to answer simple questions, we perform sub-question evaluation on three published top-performing QA models with publicly available open-source code: DFGN (Qiu et al., 2019), DecompRC (Min et al., 2019b), and CogQA (Ding et al., 2019). For all experiments, we measure EM and F1 scores for q, sub_q_1, and sub_q_2 on 1,000 human-verified examples. To measure the correctness of a predicted answer, we first use exact string match as the only metric. However, during error analysis, we find that many predicted answers that partially match the gold answers should also be regarded as correct. Some representative examples are shown in Table 1. Although these predicted answers have zero EM scores, they are semantically equivalent to the correct answers given. Therefore, we define a more flexible metric named partial match (PM) as an additional evaluation of correctness. Given a gold answer text span a_g and a predicted answer text a_p, they partially match if either one of the following two requirements is satisfied:

$$\begin{gathered} f1 > 0.8 \\ f1 > 0.6 \wedge \{(a_g \; contains \; a_p) \vee (a_p \; contains \; a_g)\} \end{gathered} \tag{1}$$

Table 2 shows the performance of the three models on multi-hop questions and their single-hop sub-questions. Compared to multi-hop questions, the performance of DFGN and DecompRC drops on simpler sub-questions, especially on the second sub-questions (11.13 F1 reduction for DFGN and 6.84 F1 reduction for DecompRC). CogQA achieves slightly better performance on sub-questions, which shows that it is also able to answer single-hop questions. The EM and F1 scores are averaged over all examples. In order to understand whether models are able to answer the sub-questions of correctly answered multi-hop questions, we collect the correctness statistics with regard to each individual example. Table 3 and Table 4 present the results. The first four rows show the percentage of examples whose multi-hop question can be correctly answered. Among these examples, we notice that there is a high probability that the models fail to answer at least one of the sub-questions, as shown in rows 2 to 4. We refer to these examples as model failure cases. The percentage of model failure cases over all correctly answered multi-hop questions is defined as model failure rate. As shown in Figure 2, all three models evaluated have a high model failure rate, indicating that the models learn to answer the complex questions without exploring the multiple steps of reasoning process as desired. The same phenomenon appears when evaluated using exact match and the less strict partial match scores.

After analyzing the model failure cases, we observe a common phenomenon that there is a high

q	q_{sub1}	q_{sub2}	DFGN	DecompRC	CogQA
c	c	c	23.0	31.3	26.7
c	c	w	9.7	7.2	5.8
c	w	c	17.9	19.1	17.8
c	w	w	7.5	5.5	2.9
w	c	c	4.9	3.0	3.6
w	c	w	17.0	18.6	22.5
w	w	c	3.5	3.4	5.9
w	w	w	16.5	11.9	14.8

Table 3: Categorical EM statistics (%) of sub-question evaluation for the three models. Under the first three columns, c stands for *correct* and w stands for *wrong*. For example, the second row shows the percentage of questions where models correctly answer both multi-hop question and the first sub-question but wrongly answer the second sub-question.

q	q_{sub1}	q_{sub2}	DFGN	DecompRC	CogQA
c	c	c	36.3	47.4	40.9
c	c	w	11.9	8.5	6.1
c	w	c	16.4	17.2	16.5
c	w	w	6.5	3.9	3.4
w	c	c	4.2	4.0	4.5
w	c	w	12.1	11.1	15.2
w	w	c	3.1	1.9	5.6
w	w	w	9.5	6.0	7.8

Table 4: Categorical PM statistics (%) of sub-question evaluation for the three models.

---

[1] The verified dataset is available at https://github.com/yxxytang/subqa

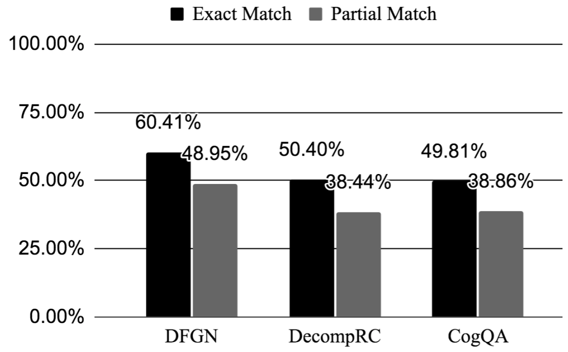


Figure 2: Model failure rates under EM and PM.

similarity between the words in the second sub-question and the words near the answer in the context. The model has learned to answer multi-hop question by local pattern matching, instead of going through the multiple reasoning steps. For the example presented in Figure 1, the model may locate the answer "*1999*" for the multi-hop question by matching the surrounding words " *Guns N Roses*" in the second sub-question. Despite answering the multi-hop question correctly, the model fails to identify the answer of the first sub-question which it is expected to retrieve as a multi-hop QA system.

4 Conclusion

We propose a new way to interpret whether multi-hop QA systems explore the multiple steps of reasoning over the evidence as desired by asking sub-questions. An automatic approach is designed to generate sub-questions for a multi-hop question. On a human-verified test set, our experiments demonstrate that top-performing multi-hop QA models fail to answer a large portion of sub-questions whose parent multi-hop questions can be correctly answered. We believe that progress on building complex QA systems that truly understand multi-hop reasoning is only possible if the evaluation metrics reward this kind of behavior. As an initial step towards a more explainable QA system, we hope our work would motivate the construction of multi-hop QA datasets with explicit reasoning paths annotated and the development of better multi-hop QA models.

References

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *NAACL-HLT*, pages 4026–4032.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *ACL*, pages 2694–2703.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *ACL*, pages 2726–2736.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, pages 1601–1611.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL-HLT*, pages 252–262.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *ACL*, pages 4249–4257.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*, pages 6097–6109.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *ACL*, pages 6140–6150.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *ACL*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Rep4NLP@ACL*, pages 191–200.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. In *TACL*, pages 287–302.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. SG-Net: syntax-guided machine reading comprehension. In *AAAI*, pages 9636–9643.

Multilingual LAMA: Investigating Knowledge in Multilingual Pretrained Language Models

Nora Kassner[*], **Philipp Dufter**[*], **Hinrich Schütze**
Center for Information and Language Processing (CIS), LMU Munich, Germany
`{kassner,philipp}@cis.lmu.de`

Abstract

Recently, it has been found that monolingual English language models can be used as knowledge bases. Instead of structural knowledge base queries, masked sentences such as "Paris is the capital of [MASK]" are used as probes. We translate the established benchmarks TREx and GoogleRE into 53 languages. Working with mBERT, we investigate three questions. (i) Can mBERT be used as a multilingual knowledge base? Most prior work only considers English. Extending research to multiple languages is important for diversity and accessibility. (ii) Is mBERT's performance as knowledge base language-independent or does it vary from language to language? (iii) A multilingual model is trained on more text, e.g., mBERT is trained on 104 Wikipedias. Can mBERT leverage this for better performance? We find that using mBERT as a knowledge base yields varying performance across languages and pooling predictions across languages improves performance. Conversely, mBERT exhibits a language bias; e.g., when queried in Italian, it tends to predict Italy as the country of origin.

1 Introduction

Pretrained language models (LMs) (Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019) can be finetuned to a variety of natural language processing (NLP) tasks and generally yield high performance. Increasingly, these models and their generative variants are used to solve tasks by simple text generation, without any finetuning (Brown et al., 2020). This motivated research on how much knowledge is contained in LMs: Petroni et al. (2019) used models pretrained with masked language to answer fill-in-the-blank templates such as "Paris is the capital of [MASK]."

	Query	Two most frequent predictions
en	X was created in MASK.	[Japan (170), Italy (56), . . .]
de	X wurde in MASK erstellt.	[Deutschland (217), Japan (70), . . .]
it	X è stato creato in MASK.	[Italia (167), Giappone (92), . . .]
nl	X is gemaakt in MASK.	[Nederland (172), Italië (50), . . .]
en	X has the position of MASK.	[bishop (468), God (68), ...]
de	X hat die Position MASK.	[WW (261), Ratsherr (108), ...]
it	X ha la posizione di MASK.	[pastore (289), papa (138), ...]
nl	X heeft de positie van MASK.	[burgemeester (400), bisschop (276) , ...]

Table 1: Language bias when querying (TyQ) mBERT. Top: For an Italian cloze question, Italy is favored as country of origin. Bottom: There is no overlap between the top-ranked predictions, demonstrating the influence of language – even though the facts are the same: the same set of triples is evaluated across languages. Table 3 shows that pooling predictions across languages addresses bias and improves performance. WW = "Wirtschaftswissenschaftler".

This research so far has been exclusively on English. In this paper, we focus on using *multilingual* pretrained LMs as knowledge bases. Working with mBERT, we investigate three questions. (i) Can mBERT be used as a multilingual knowledge base? Most prior work only considers English. Extending research to multiple languages is important for diversity and accessibility. (ii) Is mBERT's performance as knowledge base language-independent or does it vary from language to language? To answer these questions, we translate English datasets and analyze mBERT for 53 languages. (iii) A multilingual model is trained on more text, e.g., BERT's training data contains the English Wikipedia, but mBERT is trained on 104 Wikipedias. Can mBERT leverage this fact? Indeed, we show that pooling across languages helps performance.

In summary our contributions are: **i)** We automatically create a multilingual version of TREx and GoogleRE covering 53 languages. **ii)** We use an alternative to fill-in-the-blank querying – ranking entities of the type required by the template (e.g., cities) – and show that it is a better tool

---

[*] Equal contribution - random order.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3250–3258
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

to investigate knowledge captured by pretrained LMs. **iii)** We show that mBERT answers queries across languages with varying performance: it works reasonably for 21 and worse for 32 languages. **iv)** We give evidence that the query language affects results: a query formulated in Italian is more likely to produce Italian entities (see Table 1). **v)** Pooling predictions across languages improves performance by large margins and even outperforms monolingual English BERT. Code and data are available online (`https://github.com/norakassner/mlama`).

2 Data

2.1 LAMA

We follow the LAMA setup introduced by Petroni et al. (2019). More specifically, we use data from TREx (Elsahar et al., 2018) and GoogleRE.[1] Both consist of triples of the form (object, relation, subject). The underlying idea of LAMA is to query knowledge from pretrained LMs using templates without any finetuning: the triple (Paris, capital-of, France) is queried with the template "Paris is the capital of [MASK]." In LAMA, TREx has 34,039 triples across 41 relations, GoogleRE 5528 triples and 3 relations. Templates for each relation have been manually created by Petroni et al. (2019). We call all triples from TREx and GoogleRE together *LAMA*.

LAMA has been found to contain many "easy-to-guess" triples; e.g., it is easy to guess that a person with an Italian sounding name is born in Italy. *LAMA-UHN* is a subset of triples that are hard to guess introduced by Poerner et al. (2020).

2.2 Translation

We translate both entities and templates. We use Google Translate to translate templates in the form "[X] is the capital of [Y]". After translation, all templates were checked for validity (i.e., whether they contain "[X]", "[Y]" exactly once) and corrected if necessary. In addition, German, Hindi and Japanese templates were checked by native speakers to assess translation quality (see Table 2). To translate the entity names, we used Wikidata and Google knowledge graphs.

mBERT covers 104 languages. Google Translate covers 77 of these. Wikidata and Google Knowledge Graph do not provide entity translations for all languages and not all entities are contained in the knowledge graphs. For English we can find a total of 37,498 triples which we use from now on. On average, 34% of triples could be translated (macro average over languages). We only consider languages with a coverage above 20%, resulting in the final number of languages we include in our study: 53. The macro average of translated triples in these 53 languages is 43%. Figure 1 gives statistics. We call the translated dataset *mLAMA*.

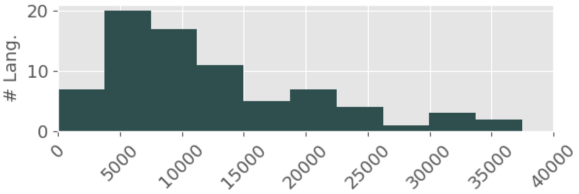


Figure 1: x-axis is the number of translated triples, y-axis the number of languages. There are 39,567 triples in the original LAMA (TREx and GoogleRE).

3 Experiments

3.1 Model

We work with mBERT (Devlin et al., 2019), a model pretrained on the 104 largest Wikipedias. We denote mBERT queried in language x as mBERT[x]. As comparison we use the English BERT-Base model and refer to it as BERT. In initial experiments with XLM-R (Conneau et al., 2020) we observed worse performance, similar to Jiang et al. (2020a). Thus, for simplicity we only report results on mBERT.

3.2 Typed and Untyped Querying

Petroni et al. (2019) use templates like "Paris is the capital of [MASK]" and give $\arg\max_{w \in \mathcal{V}} p(w|t)$ as answer where V is the vocabulary of the LM and $p(w|t)$ is the (log-)probability that word w gets predicted in the template t. Thus the object of a triple must be contained in the vocabulary of the language model. This has two drawbacks: it reduces the number of triples that can be considered drastically and hinders performance comparisons across LMs with different vocabularies. We refer to this procedure as *UnTyQ*.

We propose to use typed querying, *TyQ*: for each relation a candidate set $\mathcal{C}$ is created and the prediction becomes $\arg\max_{c \in \mathcal{C}} p(c|t)$. For templates like "[X] was born in [MASK]", we know which entity type to expect, in this case cities. We observed that (English-only) BERT-base predicts city

---

[1] `code.google.com/archive/p/relation-extraction-corpus/`

names for MASK whereas mBERT predicts years for the same template. TyQ prevents this.

We choose as $\mathcal{C}$ the set of objects across all triples for a single relation. The candidate set could also be obtained from an entity typing system (e.g., (Yaghoobzadeh and Schütze, 2016)), but this is beyond the scope of this paper. Variants of TyQ have been used before (Xiong et al., 2020).

3.3 Singletoken vs. Multitoken Objects

Assuming that objects are in the vocabulary (Petroni et al., 2019) is a restrictive assumption, even more in the multilingual case as e.g., “Hamburg” is in the mBERT vocabulary, but French “Hambourg” is tokenized to [“Ham”, “##bourg”]. We consider multitoken objects by including multiple [MASK] tokens in the templates. For both TyQ and UnTyQ we compute the score that a multitoken object is predicted by taking the average of the log probabilities for its individual tokens.

Given a template t (e.g., “[X] was born in [Y].”) let t_1 be the template with one mask token, (i.e., “[X] was born in [MASK].”) and t_k be the template with k mask tokens (i.e., “[X] was born in [MASK] [MASK] . . . [MASK].”). We denote the log probability that the token $w \in V$ is predicted at ith mask token as $p(m_i = w|t_k)$, where V is the vocabulary of the LM. To compute $p(e|t)$ for an entity e that is tokenized into l tokens $\epsilon_1, \epsilon_2, \ldots, \epsilon_l$ we simply average the log probabilities across tokens:

$$p(e|t) = \frac{1}{l} \sum_{i=1}^{l} p(m_i = \epsilon_i | t_l).$$

If k is the maximum number of tokens of any entity $e \in \mathcal{C}$ gets split into, we consider all templates $t_1, \ldots, t_k$, with C being the candidate set. The prediction is then the word with the highest average log probability across all templates $t_1, \ldots, t_k$.

Note that for UnTyQ the space of possible predictions is $V \times V \times \cdots \times V$ whereas for TyQ it is the candidate set $\mathcal{C}$.

3.4 Evaluation

We compute precision at one for each relation, i.e., $1/|T| \sum_{t \in T} \mathbb{1}\{\hat{t}_{object} = t_{object}\}$ where T is the set of all triples and $\hat{t}_{object}$ is the object predicted by TyQ or UnTyQ. Note that T is different for each language. Our final measure (p1) is then the precision at one averaged over relations (i.e., macro average). Results for multiple languages are the macro average p1 across languages.

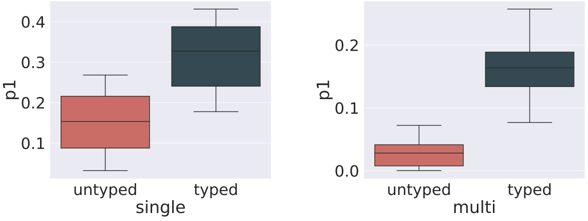


Figure 2: Distribution of p1 scores for 53 languages in UnTyQ vs. TyQ. Left: singletoken (object = 1 token). Right: multitoken (object > 1 token).

4 Results and Discussion

We first investigate TyQ and UnTyQ and find that TyQ is better suited for investigating knowledge in LMs. After exploring the translation quality, we use TyQ on mLAMA and observe rather stable performance for 21 and poor performance for 32 languages. When investigating the languages more closely, we find that prediction results highly depend on the language. Finally, we validate our initial hypothesis that mBERT can leverage its multilinguality by pooling predictions: pooling indeed performs better.

4.1 UnTyQ vs. TyQ

Figure 2 shows the distribution of p1 scores for single and multitoken objects. As expected, TyQ works better, both for single and multitoken objects. With UnTyQ, performance not only depends on the model’s knowledge, but on at least three extraneous factors: **(i)** Does the model understand the type constraints of the template (e.g., in “X is the capital of Y”, Y must be a country)? **(ii)** How “fluent” a substitution is an object under linguistic constraints (e.g., morphology) that can be viewed as orthogonal to knowledge? Many English templates cannot be translated into a single template in many languages, e.g., “in X” (with X a country) has different translations in French: “à Chypre”, “au Mexique”, “en Inde”. But the LAMA setup requires a single template. By enforcing the type, we reduce the number of errors that are due to surface fluency. **(iii)** The inadequacy of the original LAMA setup for multitoken answers. Figure 2 (right) shows that the original UnTyQ struggles with multitokens (mean p1 .03 vs. .17 for TyQ).

Overall, TyQ allows us to focus the evaluation on the core question: what knowledge is contained in LMs? From now on, we report numbers in the TyQ setting.

Manual template tuning or automatic template

	machine translated	manually corrected		manually paraphrased	
de	18.1	19.4	(6)	20.9	(18)
hi	5.4	6.2	(14)	6.2	(1)
ja	0.4	0.4	(14)	0.7	(5)

Table 2: Effect of manual template modification on UnTyQ. Shown is p1, number of templates modified (in brackets). Templates are modified to correct mistakes from machine translation and paraphrased to achieve the correct object type. Manual template correction has a small effect on UnTyQ.

mining (Jiang et al., 2020b) has been investigated in the literature to approach the typing problem. We had native speakers check templates for German, Hindi and Japanese, correct mistakes in the automatic translation and paraphrase the template to obtain predictions with the correct type. Table 2 shows that corrections do not yield strong improvements. We conclude that template modifications are not an effective solution for the typing problem.

4.2 Translation Quality

Contemporaneous work by Jiang et al. (2020a) provides manual translations of LAMA templates for 23 languages respecting grammatical gender and inflection constraints. We evaluate our machine translated templates by comparing performance on a common subset of 14 languages using TyQ querying on the TREx subset. Surprisingly, we find a performance difference of 1 percentage points (0.23 vs. 0.24, p1 averaged over languages) in favor of the machine translated templates. This indicates that the machine translated templates in combination with TyQ exhibit comparable performance but come with the benefit of larger language coverage (53 vs. 23 languages).

4.3 Multilingual Performance

In mLAMA, not all triples are available in all languages. Thus absolute numbers are not comparable across languages and we adopt a relative performance comparison: we report p1 of a model-language combination divided by p1 of mBERT's performance in English (mBERT[en]) on the exact same set of triples and call this *rel-p1*. A rel-p1 score of 0.5 for mBERT[fi] means that p1 of mBERT on Finnish is half of mBERT[en]'s performance on the same triples. rel-p1 of English BERT is usually greater than 1 as monolingual BERT tends to outperform mBERT[en].

Figure 3 shows that mBERT performs reasonably well for 21 languages, but for 32 languages rel-p1 is less than 0.6 (i.e., their p1 is 60% of English's p1). We conclude that mBERT does not exhibit a stable performance across languages. The variable performance (from 20% to almost 100% rel-p1) indicates that mBERT has no common representation for, say, "Paris" across languages, i.e., mBERT representations are language-dependent.

	LAMA	LAMA-UHN
BERT	38.5	29.0
mBERT[en]	35.0	25.7
mBERT[pooled]	**41.1**	**32.1**

Table 3: p1 for BERT, mBERT queried in English, mBERT pooled on LAMA and LAMA-UHN.

4.4 Bias

If mBERT captured knowledge independent of language, we should get similar answers across languages for the same relation. However, Table 1 shows that mBERT exhibits language-specific biases; e.g., when queried in Italian, it tends to predict Italy as the country of origin. This effect occurs for several relations: Table 4 in the supplementary presents data for ten relations and four languages.

4.5 Pooling

We investigate pooling of predictions across languages by picking the object predicted by the majority of languages. Table 3 shows that pooled mBERT outperforms mBERT[en] by 6 percentage points on LAMA, presumably in part because the language-specific bias is eliminated. mBERT[pooled] even outperforms BERT by 3 percentage points on LAMA-UHN. This indicates that mBERT can leverage the fact that it is trained on 104 Wikipedias vs. just one and even outperforms the much stronger model BERT.

5 Related Work

Petroni et al. (2019) first asked the question: can pretrained LMs function as knowledge bases? Subsequent analyses focused on different aspects, such as negation (Kassner and Schütze, 2020), easy to guess names (Poerner et al., 2020), integrating adapters (Wang et al., 2020) or finding alternatives to a "fill-in-the-blank" approach with single-token answers (Bouraoui et al., 2020; Heinzerling and Inui, 2020; Jiang et al., 2020b). Other work combines pretrained LM with information retrieval (Guu et al., 2020; Lewis et al., 2020a; Izacard and Grave, 2020; Kassner and Schütze, 2020; Petroni

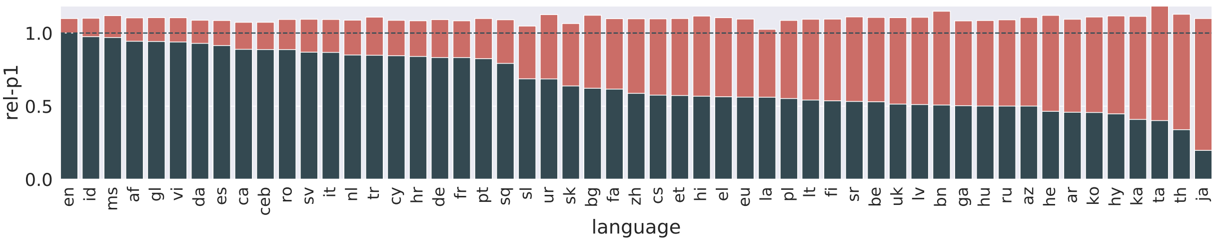


Figure 3: p1 of BERT (red) vs mBERT[x] (blue) divided by p1 of mBERT[en] on the same set of triples in each language x. mBERT captures less factual knowledge than monolingual English BERT. While performance is reasonable for 21 languages, it is below 60% for 32 languages. Dashed line is rel-p1 of mBERT[en] (by definition equal to 1.0). Performance of BERT varies slightly as the set of triples is different for each language. Note that the Wikipedia of Cebuano (ceb) consists mostly of machine translated articles.

et al., 2020). None of this work addresses languages other than English.

Multilingual models like mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) perform well for zero-shot crosslingual transfer (Hu et al., 2020). However, we are not aware of any prior work that analyzed to what degree pretrained multilingual models can be used as knowledge bases. There are many multilingual question answering datasets such as XQuAD (Artetxe et al., 2020), TiDy (Clark et al., 2020), MKQA (Longpre et al., 2020) and MLQA (Lewis et al., 2020b). Usually, multilingual models are finetuned to solve such tasks. Our goal is not to improve question answering or create an alternative multilingual question answering dataset, but instead to investigate which knowledge is contained in pretrained multilingual LMs without any kind of supervised finetuning.

There is a range of alternative multilingual knowledge bases that could be used for evaluation. Those include ConceptNet (Speer et al., 2017) or BabelNet (Navigli and Ponzetto, 2010). We decided to provide a translated versions of TREx and GoogleRE for the sake of comparability across languages. By translating manually created templates and entities we can ensure comparability across languages. This is not possible for crowd-sourced databases like ConceptNet.

In contemporaneous work, Jiang et al. (2020a) create and investigate a multilingual version of LAMA. They provide human template translations for 23 languages, propose several methods for multitoken decoding and code-switching, and experiment with a number of PLMs. In contrast to their work, we investigate typed querying, focus on comparabiliy and pooling across languages, and explore language biases.

6 Conclusion

We presented mLAMA, a dataset to investigate knowledge in language models (LMs) in a multilingual setting covering 53 languages. While our results suggest that correct entities can be retrieved for many languages, there is a clear performance gap between English and, e.g., Japanese and Thai. This suggests that mBERT is not storing entity knowledge in a language-independent way. Experiments investigating language bias confirm this finding. We hope that this paper and the dataset we publish will stimulate research on investigating knowledge in LMs *multilingually* rather than just in English.

Acknowledgements

This work was supported by the European Research Council (# 740516) and the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content. The second author was supported by the Bavarian research institute for digital transformation (bidt) through their fellowship program. We thank Yannick Couzinié and Karan Tiwana for correcting the Japanese and Hindi templates. We thank the anonymous reviewers for valuable comments.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Zied Bouraoui, José Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from BERT. In *The Thirty-Fourth AAAI Conference* *on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7456–7463. AAAI Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jonathan H. Clark, Jennimaria Palomaki, Vitaly Nikolaev, Eunsol Choi, Dan Garrette, Michael Collins, and Tom Kwiatkowski. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

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

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

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *Computing Research Repository*, arXiv:2002.08909.

Benjamin Heinzerling and Kentaro Inui. 2020. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. *Computing Research Repository*, arXiv:2008.09036.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Gautier Izacard and E. Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *ArXiv*, abs/2007.01282.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5943–5959, Online. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know. *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 3424–3430. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020b. MLQA: Evaluating cross-lingual extractive question answering. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7315–7330, Online. Association for Computational Linguistics.

Shayne Longpre, Yi Lu, and Joachim Daiber. 2020. MKQA: A linguistically diverse benchmark for multilingual open domain question answering. *CoRR*, abs/2007.15207.

Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 216–225. The Association for Computer Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 803–818, Online. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *Computing Research Repository*, arXiv:2002.01808.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2016. Intrinsic subspace evaluation of word embedding representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 236–246, Berlin, Germany. Association for Computational Linguistics.

A Language Bias

Table 4 shows the language bias for 10 relations. For each relation we aggregated the predictions across all triples and show the most common two predicted entities together with its count (in brackets). The querying language clearly affects results. The effect is drastic for relations that ask for a country (e.g., P495 or P1001). P39 yields very different results without exhibiting a clear pattern. Other relations such as P463 or P178 are rather stable.

B Data Samples

Table 4 and Table 5 show randomly sampled entries from the data.

C Pretraining Data

We investigate whether performance across languages is correlated with the amount of pretraining data for each language. To this end we investigate the number of articles per language as of January 2021 [2] and p1 for TyQ in Figure 6. We do not have access to the original pretraining data of mBERT. Thus, the number of articles we consider in the analysis might be different to the actual data used to train mBERT.

af	Microsoft Word word ontwikkel deur Microsoft.
	Aix-en-Provence is die hoofstad van Provence.
	Die hoofstad van Alpes-Maritimes is Nice.
ar	مات جون ديورانت بريفال في باريس.
	يتم إنتاج كيندل فاير بواسطة أمازون.
	اللغة الرسمية لـ كوسوفو هي الصربية.
az	Madrid və Sofiya əkiz şəhərlərdir.
	İbn ən-Nədim islam dinə bağlıdır.
	Baş qərargah Belavia Minsk -dədir.
be	Афіцыйнай мовай Фінляндская праваслаўная царква з'яўляецца фінская мова.
	Сталіцай Правінцыя Цзянсу з'яўляецца Нанкін.
	Артур Гардэн нарадзіўся ў Манчэстэр.
bg	Ник Филип е роден в Лондон.
	Наситена мазнина се състои от въглерод.
	Официалният език на Организация на обединените нации е руски език.
bn	মন্টানা আইডাহো এর সাথে সীমানা ভাগ করে দেয়।
	মৌরিতানিয়া মালি এর সাথে সীমানা ভাগ করে দেয়।
	খার্তুম এবং কায়রো হ'ল দুটি শহর।
ca	Rich Gannon juga en la posició quarterback.
	La llengua oficial de Nastola és finès.
	Nova York i Jerusalem són ciutats bessones.
ceb	Ang Mario Aldo Montano usa ka lungsuranon sa Italya.
	Ang opisyal nga sinultian sa Belhika mao ang Inolandes.
	Ang Praga ug Berlin mga kaluha nga lungsod.
cs	Mahárádža je právní termín v Indie.
	Felix Magath hraje v poloze záložník.
	Philipp Eduard Anton von Lenard pracuje v oblasti fyzika.
cy	Mae Maes Awyr Rhyngwladol Des Moines wedi'i leoli yn Iowa.
	Iaith swyddogol Aragón yw Sbaeneg.
	Enwir Flins-sur-Seine ar ôl Afon Seine.
da	Dylan Taite blev født i Liverpool.
	Johann Andreas Schmeller døde i München.
	Hovedstaden i Yolo County er Davis.
de	The Bill wurde in Englisch geschrieben.
	Buenaventura Sitjar starb in Kalifornien.
	Mai Jones funktioniert für British Broadcasting Corporation.
el	Ρεϊνάλντο Αν παίζει μουσική όπερα.
	Το Μπόσνα Σαράγεβο βρίσκεται στο Σαράγιεβο.
	Το Μεταδιθειωδες καλιο αποτελείται από θείο.
en	Vienna bread is a subclass of bread .
	Stevie Wonder is represented by music label Motown .
	Kinji Fukasaku is Japan citizen .
es	Luxemburgo comparte frontera con Francia.
	Charles Schreiber nació en Colchester.
	Polonia comparte frontera con Alemania.
et	Kohtumõistjate raamat on osa Piibel -st.
	Iisrael hoiab diplomaatilisi suhteid Jordaania -ga.
	Sambia jagab piiri Angola -ga.
eu	Ameriketako Estatu Batuak -ek Albania -ekin harreman diplomatikoak mantentzen ditu.
	Tajikistango presidente legezko terminoa da Tajikistan -n.
	Carla Bruni Paris -n lan egiteko erabiltzen zen.
fa	استان آنتورپ به آنتورپ نامگذاری شده است.
	سوزوکی کیزاشی توسط سوزوکی تولید می شود.
	رئیسجمهور زیر کلاس سیاستمدار است.
fi	Pääoman Genovan maakunta pääoma on Genova.
	Edward Joseph Kelly syntyi Chicago.
	The Home Depot perustettiin Atlanta.
fr	Mel Charles est pays de Galles citoyen.
	Audi A5 est produit par Audi.
	Honda XR est produit par Honda.
ga	Fuair Walter Gay bás i Páras.
	Tá Ollscoil Concordia suite i Montréal.
	Is cúpla cathair iad Vín agus An Bhratasláiv.
gl	Hannover e Bristol son cidades xemelgas.
	O idioma oficial de Mercia é Lingua latina.
	Afloramento de algas está composto por Alga.
he	אוניברסיטת וושינגטון נמצא ב- סיאטל.
	ויקיפדיה הצ׳כית נכתב ב- צ'כית.
	תיירות מין היא תת-סוג של תיירות.
hi	कीनिया युगाण्डा के साथ राजनयिक संबंध बनाए रखता है।
	कावेरी नदी एशिया में स्थित है।
	फ़िलीपीन्स का ध्वज फ़िलीपीन्स में एक कानूनी शब्द है।
hr	Istanbul i Bukurešt su gradovi blizanci.
	Gro Harlem Brundtland se nekada radila u Oslo.
	Izvorni jezik Ćelava pjevačica je francuski jezik.
hu	Gilles Grimandi Gap -ben született.
	Edo-kor elnevezése Edo.
	Joseph-Marie, comte Portalis anyanyelve francia.
hy	Իրան գտնվում է Ասիա –ում:
	Բարսելոնա և Դուբլին զույգ քաղաքներ են:
	Ադրբեջան -ի մայրաքաղաքն է Բաքու:

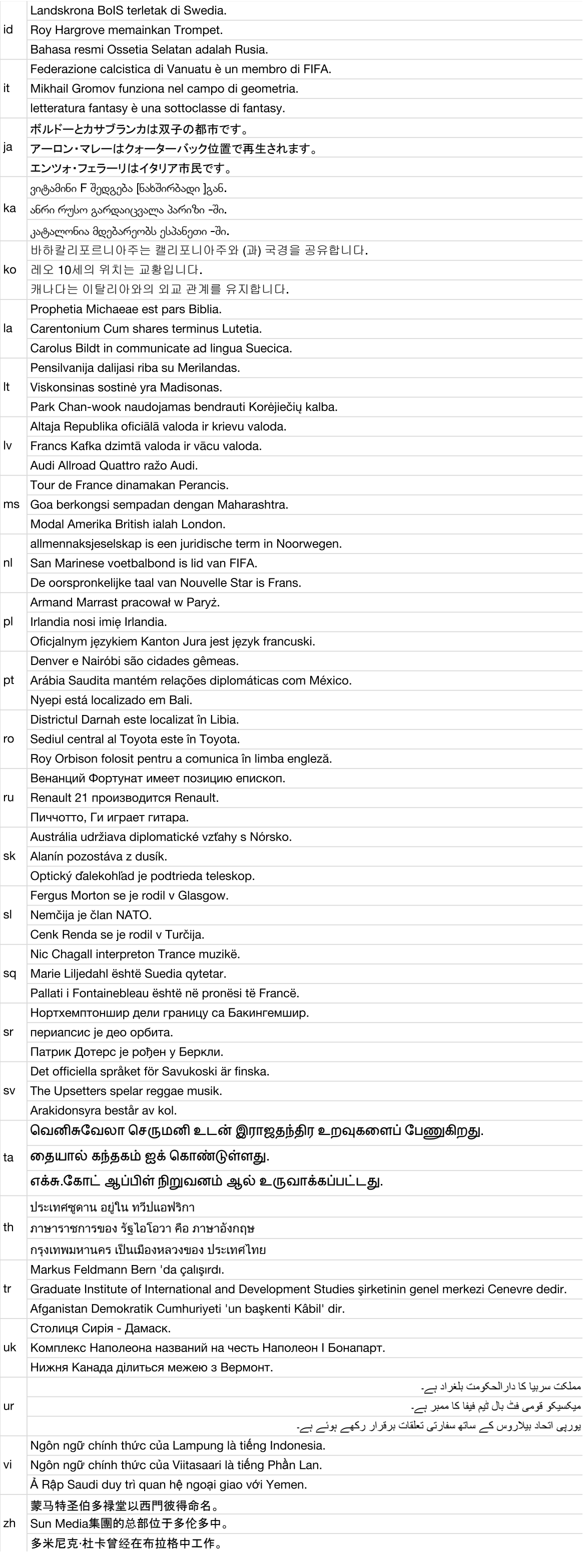
Figure 4: Three randomly sampled data entries from mLAMA per language. Due to the automatic generation of the dataset not all of them are fully correct.

---

[2] https://meta.wikimedia.org/wiki/List_of_Wikipedias

	en	de	nl	it
P495: “[X] was created in [Y]”	Japan (170), Italy (56)	Deutschland (217), Japan (70)	Nederland (172), Italië (50)	Italia (167), Giappone (92)
P101: “[X] works in the field of [Y]”	art (205), science (135)	Kunst (384), Film (64)	psychologie (263), kunst (120)	fisiologia (168), caccia (135)
P106: “[X] is [Y] by profession”	politician (423), composer (80)	Politiker (323), Journalist (128)	politicus (339), acteur (247)	giornalista (420), giurista (257)
P1001: “[X] is a legal term in [Y]”	India (12), Germany (11)	Deutschland (36), Russland (9)	Nederland (22), België (12)	Italia (31), Germania (16)
P39: “[X] has the position of [Y]”	bishop (468), God (68)	WW (261), Ratsherr (108)	burgemeester (400), bisschop (276)	pastore (289), papa (138)
P527 “[X] consists of [Y]”	sodium (125), carbon (88)	Wasserstof (398), C (49)	vet (216), aluminium (130)	calcio (165), atomo (96)
P1303 “[X] plays [Y]”	guitar (431), piano (165)	Gitarre (312), Klavier (204)	piano (581), harp (42)	arpa (188), pianoforte (139)
P178 “[X] is developed by [Y]”	Microsoft (177), IBM (55)	Microsoft (153), Apple (99)	Microsoft (200), Nintendo (69)	Microsoft (217), Apple (49)
P264 “[X] is represented by music label [Y]”	EMI (267), Swan (32)	EMI (202), Paramount Records (59)	EMI (225), Swan (50)	EMI (217), Swan (99)
P463 “[X] is a member of [Y]”	FIFA (126), NATO (33)	FIFA (118), NATO (38)	FIFA (157), WWE (16)	FIFA (121), NATO (36)

Table 4: Most frequent object predictions (TyQ) in different languages. Some relations exhibit language specific biases. WW = “Wirtschaftswissenschaftler”.



id	Landskrona BoIS terletak di Swedia.
	Roy Hargrove memainkan Trompet.
	Bahasa resmi Ossetia Selatan adalah Rusia.
it	Federazione calcistica di Vanuatu è un membro di FIFA.
	Mikhail Gromov funziona nel campo di geometria.
	letteratura fantasy è una sottoclasse di fantasy.
ja	ボルドーとカサブランカは双子の都市です。
	アーロン・マレーはクォーターバック位置で再生されます。
	エンツォ・フェラーリはイタリア市民です。
ka	ვიტამინი F შედგება [ნახშირბადი]გან.
	ანრი რუსო გარდაიცვალა პარიზი -ში.
	კატალონია მდებარეობს ესპანეთი -ში.
ko	바하칼리포르니아주는 캘리포니아주와 (과) 국경을 공유합니다.
	레오 10세의 위치는 교황입니다.
	캐나다는 이탈리아와의 외교 관계를 유지합니다.
la	Prophetia Michaeae est pars Biblia.
	Carentonium Cum shares terminus Lutetia.
	Carolus Bildt in communicate ad lingua Suecica.
lt	Pensilvanija dalijasi riba su Merilandas.
	Viskonsinas sostinė yra Madisonas.
	Park Chan-wook naudojamas bendrauti Korėjiečių kalba.
lv	Altaja Republika oficiālā valoda ir krievu valoda.
	Francs Kafka dzimtā valoda ir vācu valoda.
	Audi Allroad Quattro ražo Audi.
ms	Tour de France dinamakan Perancis.
	Goa berkongsi sempadan dengan Maharashtra.
	Modal Amerika British ialah London.
nl	allmennaksjeselskap is een juridische term in Noorwegen.
	San Marinese voetbalbond is lid van FIFA.
	De oorspronkelijke taal van Nouvelle Star is Frans.
pl	Armand Marrast pracował w Paryż.
	Irlandia nosi imię Irlandia.
	Oficjalnym językiem Kanton Jura jest język francuski.
pt	Denver e Nairóbi são cidades gêmeas.
	Arábia Saudita mantém relações diplomáticas com México.
	Nyepi está localizado em Bali.
ro	Districtul Darnah este localizat în Libia.
	Sediul central al Toyota este în Toyota.
	Roy Orbison folosit pentru a comunica în limba engleză.
ru	Венанций Фортунат имеет позицию епископ.
	Renault 21 производится Renault.
	Пиччотто, Ги играет гитара.
sk	Austrália udržiava diplomatické vzťahy s Nórsko.
	Alanín pozostáva z dusík.
	Optický ďalekohľad je podtrieda teleskop.
sl	Fergus Morton se je rodil v Glasgow.
	Nemčija je član NATO.
	Cenk Renda se je rodil v Turčija.
sq	Nic Chagall interpreton Trance muzikë.
	Marie Liljedahl është Suedia qytetar.
	Pallati i Fontainebleau është në pronësi të Francë.
sr	Нортхемптоншир дели границу са Бакингемшир.
	периапсис је део орбита.
	Патрик Дотерс је рођен у Беркли.
sv	Det officiella språket för Savukoski är finska.
	The Upsetters spelar reggae musik.
	Arakidonsyra består av kol.
ta	வெனிசுவேலா செருமனி உடன் இராஜதந்திர உறவுகளைப் பேணுகிறது.
	தையால் கந்தகம் ஐக் கொண்டுள்ளது.
	எக்சு.கோட் ஆப்பிள் நிறுவனம் ஆல் உருவாக்கப்பட்டது.
th	ประเทศซูดาน อยู่ใน ทวีปแอฟริกา
	ภาษาราชการของ รัฐไอโอวา คือ ภาษาอังกฤษ
	กรุงเทพมหานคร เป็นเมืองหลวงของ ประเทศไทย
tr	Markus Feldmann Bern 'da çalışırdı.
	Graduate Institute of International and Development Studies şirketinin genel merkezi Cenevre dedir.
	Afganistan Demokratik Cumhuriyeti 'un başkenti Kâbil' dir.
uk	Столиця Сирія - Дамаск.
	Комплекс Наполеона названий на честь Наполеон I Бонапарт.
	Нижня Канада ділиться межею з Вермонт.
ur	مملکت سربیا کا دارالحکومت بلغراد ہے۔
	میکسیکو قومی فٹ بال ٹیم فیفا کا ممبر ہے۔
	یورپی اتحاد بیلاروس کے ساتھ سفارتی تعلقات برقرار رکھے ہوئے ہے۔
vi	Ngôn ngữ chính thức của Lampung là tiếng Indonesia.
	Ngôn ngữ chính thức của Viitasaari là tiếng Phần Lan.
	Ả Rập Saudi duy trì quan hệ ngoại giao với Yemen.
zh	蒙马特圣伯多禄堂以西門彼得命名。
	Sun Media集團的总部位于多伦多中。
	多米尼克·杜卡曾经在布拉格中工作。

Figure 5: Data samples continued.

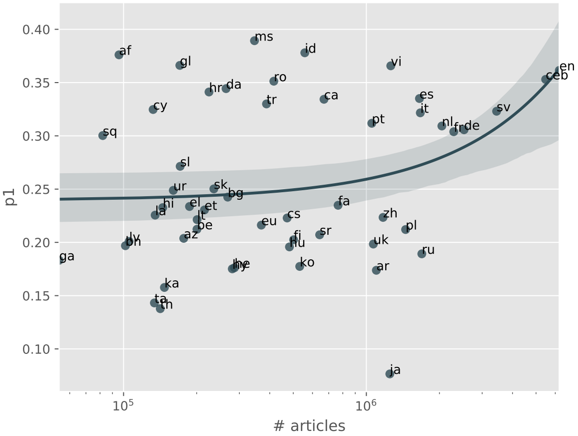


Figure 6: Scatter plot of p1 TyQ and number of articles in the corresponding Wikipedia. There is no clear trend visible.

Variational Weakly Supervised Sentiment Analysis with Posterior Regularization

Ziqian Zeng, Yangqiu Song
Department of CSE, HKUST, Hong Kong, China
`{zzengae, yqsong}@cse.ust.hk`

Abstract

Sentiment analysis is an important task in natural language processing (NLP). Most of existing state-of-the-art methods are under the supervised learning paradigm. However, human annotations can be scarce. Thus, we should leverage more weak supervision for sentiment analysis. In this paper, we propose a posterior regularization framework for the variational approach to the weakly supervised sentiment analysis to better control the posterior distribution of the label assignment. The intuition behind the posterior regularization is that if extracted opinion words from two documents are semantically similar, the posterior distributions of two documents should be similar. Our experimental results show that the posterior regularization can improve the original variational approach to the weakly supervised sentiment analysis and the performance is more stable with smaller prediction variance.

1 Introduction

Sentiment analysis is a task of identifying the sentiment polarity expressed in textual data (Liu, 2012). Most state-of-the-art sentiment analysis methods in the literature are supervised methods which require many labeled training data. However, human annotations in the real world are scarce. While we assume there is abundant annotated data to train more and more complex models, there is still a need to consider weakly supervised methods that require less human annotation.

One way to perform weakly supervised sentiment analysis is using a predefinited lexicon (Turney, 2002; Taboada et al., 2011). A lexicon consists of many opinion words. For each opinion word, its polarity (positive or negative) and strength (the degree to which the opinion word is positive or negative) are annotated by domain experts. lexicon-based weakly supervised methods perform a dictionary lookup and assign a polarity according to all opinion words extracted from a document. A good lexicon requires high precision and high coverage, which needs a lot of human effort.

Another way to do weakly supervised sentiment analysis is using limited keywords (Meng et al., 2018; Zeng et al., 2019). Compared with lexicon-based methods, user-provided keywords require less human effort. Among keyword-based methods, there are two directions. First, (Meng et al., 2018) leveraged limited keywords to expand more keywords and generate pseudo-labeled data, and then performed self-training on real unlabeled data for model refinement. Possible improvements of this direction include investigating more advanced keywords expansion techniques to generate better pseudo-labeled samples (Miller et al., 2012) and developing more advanced self-training algorithms(Coden et al., 2014).

Second, the Variational Weakly Supervised (VWS) sentiment analysis (Zeng et al., 2019) used target-opinion word pairs as supervision signal. Its objective function is to predict an opinion word given a target word. For example, in a sentence "the room is big," "room" is a target word and "big" is an opinion word. By introducing a latent variable (the sentiment polarity), they can learn a well-approximated posterior distribution via optimizing the evidence lower bound. The posterior probability here is the probability of a possible polarity (e.g., positive or negative) given a document, which is a typical sentiment classifier.

A potential issue with VWS is that optimizing the objective function may not guide the role of the latent variable to be sentiment polarity. For example, when half of reviews mention "big room" and half of reviews mention "small room," the latent variable is possibly related to the size of rooms, but the expected role of the latent variable is the sentiment polarity of rooms. Hence how to control and regularize the posterior distribution is very im-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3259–3268
April 19 - 23, 2021.

portant. One indirect way to control the posterior distribution is clever initialization (Ganchev et al., 2010). Originally, VWS aims to predict the sentiment polarity of each aspect for the multi-aspect sentiment analysis. So it uses the overall sentiment polarity to pretrain the model so that the posterior distribution has a good initialization. The overall polarity and polarity of each aspect are highly correlated. Thus, the initialization is highly likely to be similar to the true posterior distribution.

In this paper, we propose to use posterior regularization to regularize the VWS approach for sentiment analysis. There are two types of side information we can leverage to regularize the latent variable. First, calculating the similarity between keywords and extracted opinion words from a document can guide the model to decide which polarity that the document belongs to. Second, calculating the similarity between extracted opinion words from two documents can guide the model to decide whether two documents belong to the same polarity. The first type of side information is very easy to leverage, and it is reflected by our pretraining process. When a document is similar to keywords associated with a specific polarity, then we enforce that the posterior probability of a specific polarity should be large. In this case, in the pretraining process, we assign a pseudo label to the document. The second type of side information does not directly suggest which sentiment polarity that a document should be assigned to. It enforces pairwise constraints to the model. Our proposed posterior regularization leverage the second side information to ensure that when two documents are similar (dissimilar), the regularization enforces the posterior distribution of two documents to be similar (dissimilar).

Our contributions are summarized as follows,

- We develop a posterior regularization framework for the variational weakly supervised sentiment analysis.
- The experimental results show that the proposed regularization can improve the VWS model, make the results more stable, and outperform other weakly supervised baselines.

Our code is available at https://github.com/HKUST-KnowComp/VWS-PR.

2 Methodology

In this section, we first review the variational weakly supervised (VWS) sentiment analysis method in Section 2.1. Then we introduce our posterior regularization in Section 2.2.

2.1 VWS Sentiment Analysis

Before formally introducing the VWS framwork, we give a concrete example to illustrate how VWS works. Let $\mathbf{x}$ be the representation of a document x, e.g., bag of words or feature outputs of neural networks. Let C be a random variable, indicating the sentiment polarity of a document. The possible value assignment of C can be *positive* or *negative*, or rating from 1 to 10. Suppose there is a document x where we extract an opinion word "terrific." The objective function is to maximize the probability of opinion word "terrific." By introducing a latent variable C, the objective function is split into two probabilities, corresponding to two classifiers, namely, sentiment polarity classifier and opinion word classifier. The input of sentiment classifier is the document representation $\mathbf{x}$, and it produces a probability distribution of sentiment polarity, i.e., $p(C = positive|\mathbf{x})$ and $p(C = negative|\mathbf{x})$. The input of opinion word classifier is extracted opinion words and the estimated sentiment polarity distribution, and it produces a probability distribution of opinion word given estimated sentiment polarity distribution, i.e., $p(\text{"terrific"}|C = positive)$ and $p(\text{"terrific"}|C = negative)$.

2.1.1 Sentiment Polarity Classifier

The sentiment polarity classifier aims to estimate a distribution $q(C|\mathbf{x})$, where C is a discrete random variable representing the sentiment polarity of a document. Let c denote a possible value of the random variable C, representing a possible value of sentiment polarity, e.g., *positive* or *negative*. The sentiment classifier estimates the probability as

$$q(C = c|\mathbf{x}) = \frac{\exp\left(\mathbf{w}_c^T \mathbf{x}\right)}{\sum_{c'} \exp\left(\mathbf{w}_{c'}^T \mathbf{x}\right)} , \tag{1}$$

$$\mathbf{x} = CNN(x) , \tag{2}$$

where $\mathbf{w}_c$ is a trainable vector associated with a sentiment polarity c, x is a document, and $\mathbf{x}$ is the document representation. The representation of a document $\mathbf{x}$ can be various. We use Convolutional Neural Network (CNN) in the experiment.

2.1.2 Opinion Word Classifier

The opinion word classifier aims to estimate the probability of an opinion word w_o given a possible value of sentiment polarity c:

$$p(w_o|c) = \frac{\exp\left(\varphi(w_o, c)\right)}{\sum_{w'_o} \exp\left(\varphi(w'_o, c)\right)} , \tag{3}$$

where $\varphi(\cdot)$ is a scoring function taking opinion word w_o and a possible value of sentiment polarity c as inputs. The nature of the scoring function is about the frequency of occurrence. If an opinion word and a possible value of sentiment polarity co-occur frequently, the score will be high, otherwise, it will be low. Specifically, we define:

$$\varphi(w_o, c) = \mathbf{a}_c^T \mathbf{w}_o , \tag{4}$$

where $\mathbf{w}_o$ is the trainable word embedding of opinion word w_o, $\mathbf{a}_c$ is a trainable vector associated with c. The scoring function can be various, e.g., multilayer perceptron (MLP). Here we only introduce the simplest case.

Given a possible value of sentiment polarity c, VWS aims to maximize the probability of opinion words that frequently occurred with c. For example, the opinion word "good" is usually occurred with sentiment polarity *positive*, and the opinion word "terrible" is usually occurred with sentiment polarity *negative*.

2.1.3 Training Objective

The objective function of VWS is to maximize the log-likelihood of an opinion word w_o. After introducing a latent variable (i.e., the sentiment polarity of a document) to the objective function, we can derive a variational lower bound of the log-likelihood which can incorporate two classifiers. The first one corresponds to the sentiment classifier. The second one corresponds to the opinion word classifier. The variational lower bound of log-likelihood is shown as follows:

$$\begin{aligned}
\mathcal{L}_1 &= \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \log p(w_o) \\
&= \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \log \sum_c p(w_o, c) \\
&= \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \log \sum_c q(c|\mathbf{x}) \left[\frac{p(w_o, c)}{q(c|\mathbf{x})}\right] \\
&\geq \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \sum_c q(c|\mathbf{x}) \left[\log \frac{p(w_o, c)}{q(c|\mathbf{x})}\right] \\
&= \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \mathbb{E}_{q(C|\mathbf{x})} \left[\log p(w_o|c) p(c)\right] \\
&+ \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} H(q(C|\mathbf{x})),
\end{aligned} \tag{5}$$

where X is the training set containing all documents, and $\mathcal{P}_x$ is the set of all opinion words extracted from a document x, $H(\cdot)$ refers to the Shannon entropy, and $q(c|\mathbf{x})$ is short for $q(C = c|\mathbf{x})$. By applying Jensen's inequality, the log-likelihood is lower-bounded by Eq. (5). The equality holds if and only if the KL-divergence of two distributions, $q(C|\mathbf{x})$ and $p(C|w_o)$, equals to zero. Maximizing the evidence lower bound is equivalent to minimizing the KL-divergence. Hence, VWS can learn a sentiment classifier that can produce a similar distribution to the true posterior $p(C|w_o)$. We assume that the training set is perfectly balanced, which means the prior distribution of sentiment polarity, i.e., $p(C)$, is a uniform distribution. Hence, $p(c)$ is a constant, which can be ignored.

2.1.4 Approximation

The partition function in Eq. (3) requires the summation over all opinion words in the vocabulary. Since the size of the opinion word vocabulary is large, VWS uses the negative sampling technique (Mikolov et al., 2013) to approximate Eq. (3). Specifically, VWS approximates $p(w_o|c)$ in the objective (3) with the following objective function:

$$\log \sigma\big(\varphi(w_o, c)\big) + \sum_{w'_o \in \mathcal{N}} \log\left(1 - \sigma\big(\varphi(w'_o, c)\big)\right) , \tag{6}$$

where w'_o is a negative sample in opinion words vocabulary, $\mathcal{N}$ is the set of negative samples and $\sigma(\cdot)$ is the sigmoid function. In order to ensure that the approximation part and the entropy term are on the same scale (Marcheggiani and Titov, 2016), a hyper-parameter α is added to the entropy term. The objective function becomes:

$$\begin{aligned}
\mathcal{L}_2 = \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} & \mathbb{E}_{q(C|\mathbf{x})} \Big[\log \sigma\big(\varphi(w_o, c)\big) \\
&+ \sum_{w'_o \in \mathcal{N}} \log\left(1 - \sigma\big(\varphi(w'_o, c)\big)\right) + \log p(c)\Big] \\
&+ \sum_{x \in X} \sum_{w_o \in \mathcal{P}_x} \alpha H(q(C|\mathbf{x})).
\end{aligned} \tag{7}$$

2.2 Posterior Regularization

As pointed out by (Ganchev et al., 2010), controlling the posterior distribution is crucial for models that estimate posterior distribution by maximizing the likelihood of the observed data via marginalizing over the latent variables. We need side information to regularize the posterior distribution. The side information we leveraged is that if the opinion words extracted from two documents are similar semantically, then these two documents probably are in the same class, and if the opinion words are opposite semantically, then these two documents are probably not in the same class. For example, if one document x^i contains opinion words “great” and “awesome,” another document x^j contains opinion words “great” and “excellent,” and another document x^k contains opinion words “awful” and “terrible,” it is highly possible that x^i and x^j belong to the same class because their extracted opinion words are similar semantically, and x^i and x^k do not belong to the same class because their extracted opinion words are opposite semantically.

We formulate our posterior regularization as:

$$\mathcal{R}(x^i, x^j) = -d(x^i, x^j) \cdot \mathcal{S}(\mathcal{O}(x^i), \mathcal{O}(x^j)), \quad (8)$$

where $d(x^i, x^j)$ is short for $d(q(C|\mathbf{x}^i), q(C|\mathbf{x}^j))$, meaning the distance of two posterior distributions, and we use Euclidean distance metric; $\mathcal{S}(\cdot,\cdot)$ is a score function which measures the similarity or dissimilarity between two sets of opinion words; $\mathcal{O}(x^i)$ represents all opinion words extracted from a document x^i. We finally maximize Eq. (8) in the objective function. When $\mathcal{S}(\mathcal{O}(x^i), \mathcal{O}(x^j))$ is positive (suggesting similar), this regularization enforces the distance to be small, and when $\mathcal{S}(\mathcal{O}(x^i), \mathcal{O}(x^j))$ is negative (suggesting dissimilar), this regularization enforces the distance to be large. When $\mathcal{S}(\mathcal{O}(x^i), \mathcal{O}(x^j))$ is zero, it suggests comparison between opinion words cannot decide whether two documents are similar or not.

Next, we will introduce the scoring function $\mathcal{S}(\cdot,\cdot)$. Suppose a document x^i contains a set of opinion words $\mathcal{O}(x^i) = \{w^i_{o_1}, w^i_{o_2}, \cdots, w^i_{o_k}\}$ and a document x^j contains a set of opinion words $\mathcal{O}(x^j) = \{w^j_{o_1}, w^j_{o_2}, \cdots, w^j_{o_k}\}$. We define an operation $cos(\mathcal{O}(x^i), \mathcal{O}(x^j))$ over two sets of opinion words. It will return all cosine similarity values of all valid opinion word pairs where one word must come from $\mathcal{O}(x^i)$ and the other must come from $\mathcal{O}(x^j)$. We represent opinion words using embeddings, i.e., the embeddings in the opinion word classifier in Eq. (4). If there are k opinion words in each set, $cos(\mathcal{O}(x^i), \mathcal{O}(x^j))$ will return $\frac{k*(k-1)}{2}$ cosine similarity values. When we want to know whether two documents are similar in opinion words, we pay attention to the maximum value, i.e., $max\_cos = max\big(cos(\mathcal{O}(x^i), \mathcal{O}(x^j))\big)$. When we want to know whether two documents are dissimilar in opinion words, we pay attention to the minimum value, i.e., $min\_cos = min\big(cos(\mathcal{O}(x^i), \mathcal{O}(x^j))\big)$. So we define:

$$\mathcal{S}(\cdot,\cdot) = \begin{cases} max\_cos, & \begin{array}{l} max\_cos > \gamma_1 \quad \& \\ min\_cos \geq \gamma_2, \end{array} \\ min\_cos, & \begin{array}{l} max\_cos \leq \gamma_1 \quad \& \\ min\_cos < \gamma_2, \end{array} \\ \delta, & \begin{array}{l} max\_cos > \gamma_1 \quad \& \\ min\_cos < \gamma_2, \end{array} \\ 0, & otherwise, \end{cases} \quad (9)$$

where $\mathcal{S}(\cdot,\cdot)$ is short for $\mathcal{S}((\mathcal{O}(x^i), \mathcal{O}(x^j))$ due to space limit. The first condition means two documents have some semantically similar opinion words ($max\_cos > \gamma_1$) and have no semantically dissimilar opinion words ($min\_cos \geq \gamma_2$). The value returned by the function score is max_cos. It should be a positive value. The second condition means two documents have no semantically similar opinion words ($max\_cos \leq \gamma_1$) and have some semantically dissimilar opinion words ($min\_cos < \gamma_2$). The value returned by function score is min_cos. It should be a negative number. The third condition means two documents have some semantically similar opinion words ($max\_cos > \gamma_1$) and also have some semantically dissimilar opinion words ($min\_cos < \gamma_2$). This condition corresponds to a real-world situation that when some customers want to express some negative sentiment, they usually point out some positive aspects first, and then start with a “but”, and emphasize some negative aspects. The opinion words sets extracted from these type of documents have both negative and positive opinion words. When we compare two of them, they will have some similar opinion words and dissimilar opinion words. In this case, we tend to assume they are in the same class. If the third condition is satisfied, it will return an non-negative value $\delta \in [0, 1]$; The final condition means two documents have no semantically similar opinion words ($max\_cos \leq \gamma_1$) and have no semantically dissimilar opinion words ($min\_cos \geq \gamma_2$). It will return 0.

The mechanism of the regularization is that if the posterior distributions $q(C|\mathbf{x}^i)$ and $q(C|\mathbf{x}^j)$ are different from each other, i.e., $d\big(q(C|\mathbf{x}^i), q(C|\mathbf{x}^j)\big)$ is large, but opinion words suggest that these two documents should be in the same cluster i.e., $s(w^i_d, w^j_d)$

Datasets	Yelp	IMDB	Amazon
Size of training set	38,000	20,000	20,000
Size of development set	3,800	2,000	2,000
Average doc length	155.12	252.90	88.28
Opinion vocabulary size	1,097	688	394

Table 1: Statistics of Yelp, IMDB, and Amazon dataset.

is large, then $d\big(q(C|\mathbf{x}^i), q(C|\mathbf{x}^j)\big)$ will be encouraged to be small by applying the regularization. Oppositely, if the posterior distributions $q(C|\mathbf{x}^i)$ and $q(C|\mathbf{x}^j)$ are similar, i.e., $d\big(q(C|\mathbf{x}^i), q(C|\mathbf{x}^j)\big)$ is small, but opinion words suggest that these two documents should be in the different cluster i.e., $s(w_d^i, w_d^j)$ is small, then $d\big(q(C|\mathbf{x}^i), q(C|\mathbf{x}^j)\big)$ will be encouraged to be large by applying the regularization.

The final objective function with posterior regularization is as follows,

$$\mathcal{J} = \mathcal{L}_2 + \beta \sum_{x^i \in X} \sum_{x^j \in X} R(x^i, x^j) \ , \qquad (10)$$

where x^i and x^j are documents in the training set X. The constraints are defined in a $|X| \times |X|$ space. In practice, we train our model batch by batch. So we only apply the constraints within a mini-batch. There are at most $|X_b| \times |X_b|$ constraints in a mini-batch, where $|X_b|$ is the number of samples in a mini-batch.

3 Experiments

In this section, we evaluate the empirical performance of our method on binary sentiment classification tasks.

3.1 Datasets

We use three corpora to evaluate the performance of our proposed method. All corpora have two classes and perfectly balanced. For all methods, we use a development set for hyper-parameter tuning. For all methods, we use the training set as the test set since all methods do not use the ground truth in the training set.

(1) **Yelp** Review: We use the Yelp reviews polarity dataset from (Zhang et al., 2015) and take its test set containing 38,000 documents as the corpus for evaluation. For hyper-parameter tuning, we also extract 3,800 documents from the original training set of (Zhang et al., 2015) to serve as a development set.

(2) **IMDB** Review: We use the IMDB reviews polarity dataset from (Maas et al., 2011) and randomly extract $20,000$ reviews from its original test set as the corpus for evaluation. For hyper-parameter tuning, we also extract $2,000$ documents from the original training set of (Maas et al., 2011) to serve as a development set.

(3) **Amazon** Review: We use the Amazon reviews polarity dataset from (Zhang et al., 2015) and randomly extracted $20,000$ reviews from its original test set as the corpus for evaluation. For hyper-parameter tuning, we also extract $2,000$ documents from the original training set of (Zhang et al., 2015) to serve as a development set.

Table 1 provides the details of these datasets.

3.2 Compared Methods

Lexicon uses an opinion lexicon to assign sentiment polarity to a document (Read and Carroll, 2009; Pablos et al., 2015). We combine two popular opinion lexicons used by (Hu and Liu, 2004) and (Wilson et al., 2005) to get a larger lexicon. If an extracted opinion is in the positive (negative) lexicon, it votes for positive (negative). When the opinion word is with a negation word such as "no" and "not", its polarity will be the opposite. Then, the polarity of a document is determined by using majority voting among all extracted opinion words. When the number of positive and negative words is equal, the document will be randomly assigned a polarity.

WeSTClass (Meng et al., 2018) first generates pseudo labels for documents which contain user-provided keywords. Keywords are expanded to generate more pseudo samples. It pretrains a CNN/LSTM model using pseudo samples as the training set and then performs a self-training process. Here, we use CNN because it empirically outperforms LSTM. The CNN architecture we used here is the same as the one described in (Meng et al., 2018).

Keyword Pretrain generates pseudo labels for

Datasets	Positive			Negative		
Yelp	terrific	amazing	awesome	horrible	worst	bad
IMDB	great	fantastic	awesome	awful	worst	bad
Amazon	great	fantastic	awesome	poor	worst	bad

Table 2: Keywords of Yelp, IMDB, and Amazon datasets.

documents in training set which contain user-provided keywords. We pretrain a CNN model using pseudo samples as the training set. In the CNN model, four different filter sizes $\{2, 3, 4, 5\}$ are applied, and a max-pooling layer is applied to each convolutional layer, and each convolutional layer has 100 filters.

VWS (Zeng et al., 2019) uses target opinion word pairs as supervision signal. It trains a sentiment polarity classifier and opinion word classifier simultaneously via optimizing the variational lower bound. We use a CNN model as the sentiment polarity classifier. And we pretrain it by generating pseudo labels for documents which contain user-provided keywords in the training set. The CNN architecture is the same as the one in **Keyword Pretrain**.

VWS-PR is VWS method with proposed posterior regularization.

3.3 Keywords and Opinion Word Extraction

We manually select three keywords for each class. The details of keywords of three datasets are shown in table 2.

For opinion word extraction, we adopt four rules proposed by VWS (Zeng et al., 2019) in the implementation. All rules rely on dependency parser (Chen and Manning, 2014). When a target word and an opinion word satisfy a dependency relation, we will extract the opinion word. The details of dependency relation and examples are provided in Table 3. When a pair of words satisfy one rule, there are still some restrictions on head and tails words to be satisfied. There is no restriction for Rule 1. For Rule 2, the head word should be an adjective and the tail word should a noun. For Rule 3, the head word should be one of the following four words: “like,” “dislike,” “love,” and “hate.” For Rule 4, the head word should be one of the following word: “seem,” “look,” “feel,” “smell,” and “taste.”

3.4 Result Analysis

Table 4 shows that our method VWS-PR outperforms VWS by $\mathbf{4}\%$, $\mathbf{2}\%$, and $\mathbf{1}\%$ on Yelp, IMDB, and Amazon datasets respsectively. Compared with WeSTClass and VWS, our method is more stable, i.e., smaller standard deviation, which shows that the regularization confine the posterior distribution to a smaller space. The performance of lexicon method is bad across three datasets. The main reason is that it does not involve any learning process. Keyword pretraining method can outperform lexicon method. But pseudo labels are not ground truths, hence the pseudo training set contains noises. Also, user provided keywords are limited, so the training samples with pseudo labels are restricted to some samples which contain certain keywords. For example, reviews with an extreme polarity (only expressing positive polarity or only express negative polarity) are likely to be pseudo samples. But most of reviews express mixed polarities. This will hinder the generalization ability. WeSTClass outperforms the keyword pretraining method on Yelp and Amazon dataset due to keyword expansion and self-training process. But in IMDB dataset, WeSTClass is slightly worse than the keyword pretraining method. Possible reason would be keyword expansion involve some harmful keywords and self-training procedure amplifies errors. VWS outperforms WeSTClass on IMDB and Amazon datasets and is comparable to Yelp dataset.

3.5 Hyper-parameters Sensitivity Analysis

We first show F1 scores on three datasets with varied β in Figure 1(a). It shows that optimal β values of our model on three datasets are different. When they achieve optimal β value, the standard deviation is much smaller than others. The regularization makes models more stable. Our method on IMDB and Amazon is not very sensitive. The changes are within 2%. Our method on Yelp is more sensitive. But we could still find a range, e.g., 0.1 to 0.5, where the changes are within 2%. When β keeps growing, the performance in Yelp deteriorates a lot.

Rule	Dependency Relation	Example	Extracted Word
1	adjectival modifier	they have delicious food	delicious
2	nominal subject	the room is big	big
3	direct object	i like it	like
4	open clausal complement	i feel comfortable	comfortable

Table 3: Opinion words extraction rules.

Methods \ Datasets	Yelp		IMDB		Amazon	
	Mean	Std	Mean	Std	Mean	Std
Lexicon	0.5982	0.0006	0.5998	0.0013	0.5754	0.0014
Keyword Pretraining	0.7441	0.0060	0.7496	0.0050	0.6375	0.0171
WeSTClass (Meng et al., 2018)	0.8061	0.0105	0.7354	0.0096	0.7374	0.0082
VWS (Zeng et al., 2019)	0.8014	0.0179	0.7825	0.0045	0.7530	0.0025
VWS-PR	**0.8431**	0.0013	**0.8025**	0.0041	**0.7644**	0.0020

Table 4: F1 scores of weakly supervised sentiment analysis methods on Yelp, IMDB, and Amazon.

The main reason is probably that the opinion word vocabulary size in Yelp is much larger than other datasets, and hence it is likely that the vocabulary in Yelp contains more noisy opinion words. When β is large, the noises may harm the performance.

We then show F1 scores on three datasets with varied γ_1 in Figure 1(b). The optimal γ_1 values of three datasets are the same, i.e., 0.7. The trends are consistent on three datasets. F1 score first increases and then decreases. When γ_1 is small, the constraints are easier to satisfy and the performance is bad because it may involve more noisy constraints. It probably enforces two samples to be similar but in reality, they are not similar. When γ_1 is large, the performance is also bad because it has fewer constraints that enforce two samples to be similar. It could have made use of more constraints.

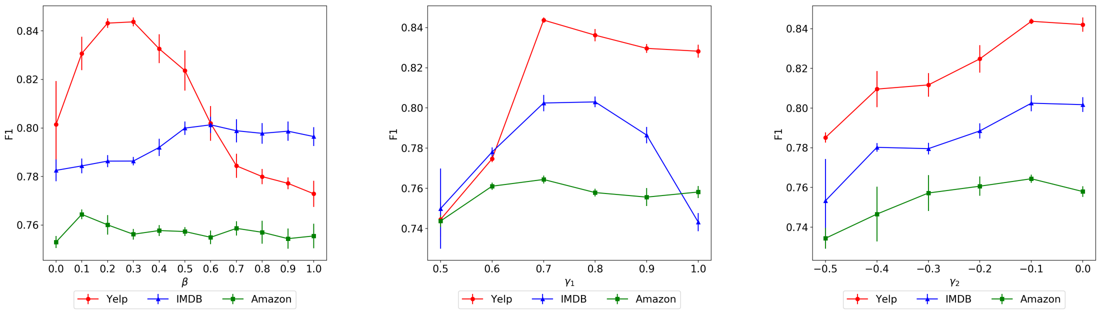
We show F1 scores on three datasets with varied γ_2 in Figure 1(c). The optimal γ_2 values of three datasets are the same, i.e., -0.1. The trends are consistent on three datasets. F1 score first increases and then decreases. When γ_2 is small, the performance is bad because it has fewer constraints that enforce two samples to be dissimilar. It could have made use of more constraints. When γ_2 is large, the performance is bad because it has more noisy constraints.

3.6 Error Analysis

We show some incorrectly predicted documents by VWS-PR on three datasets in Table 5. For the first document, the customer emphasizes price a lot. Our method cannot extract opinion words on snippets such as “not at that price” and “for half the price.” For the second document, the reviewer loves this movie because he/she loves the basketball player. The reviewer thinks that the movie itself does not deserve a high score. Our method detects both positive and negative polarities on this document, so it tends to predict as negative. Because most mixed polarities are likely to be negative polarity. The regularization enforces this pattern. This document obviously is different from other documents with mixed opinion words. For the last document, our method cannot extract opinion words on snippets such as “no wrist strap” and “without a place to attach a wrist strap.” Our method is good at extracting words on subjective expression such as “nice light,” but not on descriptive expression such as “no wrist strap.” Our method fails because no other knowledge source indicates that “no wrist strap” is negative.

3.7 Implementation Details

For WeSTClass and VWS, we used code released by (Meng et al., 2018) and (Zeng et al., 2019) respectively, and followed their preprocessing steps and optimal settings. For VWS and VWS-PR, we pretrain a CNN model using pseudo-labeled samples. After that, the embeddings are untrainable. The rest of parameters are trainable. For our method, the hyperparameter settings of VWS part



(a) Varying β with fixed γ_1 and γ_2. (b) Varying γ_1 with fixed β and γ_2. (c) Varying γ_2 with fixed β and γ_1.

Figure 1: Hyper-parameters sensitivity analysis.

Dataset	Document	Prediction	Ground Truth	Extracted Words
Yelp	It was good, but not at that price. There are so many other good italian places in the area for half the price.	good	bad	good
IMDB	I love gheorghe muresan, so i automatically loved this movie. Everything else about it was so so. Billy crystal is a good actor, even if he is annoying.	bad	good	love good annoying
Amazon	No wrist strap. Nice light, well made. But why would anyone design a tactical style flashlight without a place to attach a wrist strap.	good	bad	nice

Table 5: Documents that are predicted incorrectly by VWS-PR.

is the same as described in (Zeng et al., 2019). We implemented our models using TensorFlow (Abadi et al., 2016). When tuning hyper-parameters of regularization term, we perform grid search on $\gamma_1 \in [0.5, 0.6, \cdots, 1.0]$ and $\gamma_2 \in [-0.5, -0.4, \cdots, 0]$. After than, we fix γ_1 and $\gamma 2$, then tune $\beta \in [0.1, 0.2, \cdots, 1.0]$. δ is fixed to 1.

4 Related Work

In this section, we review the related work on weakly supervised sentiment analysis.

Using a lexicon is a typical way to perform weakly supervised sentiment analysis. One line of works perform simple assignment, i.e., majority voting, based on sentiment orientation scores of extracted opinion words. Some methods (Missen and Boughanem, 2009; Tsytsarau et al., 2010) used sentiment orientation scores in existing lexicons directly, and aggregated them within a document to determine polarity. Some methods developed their own semantic orientation estimation algorithm. For example, (Turney, 2002) first identified phrases in the review and then estimated the semantic orientation of each extracted phrases. The semantic orientation of a given phrase is calculated by comparing its similarity to a positive reference word ("excellent") with its similarity to a negative reference word ("poor"). This method determined the sentiment polarity based on the average semantic orientation of the phrases extracted from the review. (Kamps et al., 2004) used the minimum path distance between a phrase and pivot words ("good" and "bad") in WordNet to estimate the semantic orientation of extracted phrases.

Another line of works involve learning process when using a lexicon. (Li et al., 2009; Zhou et al., 2014) proposed a constrained non-negative matrix tri-factorization approach to sentiment analysis, and used a sentiment lexicon as prior knowledge. In these models, a term-document matrix is approximated by three factors that specify soft membership of terms and documents in one of k classes. All three factors are non-negative matrices. The first factor is a matrix representing knowledge in the word space, i.e., each row represents the posterior probability of a word belonging to the k classes. The second factor is a matrix providing a condensed view of the term-document matrix. The third factor is a matrix representing knowledge in document space, i.e., each row represents the posterior probability of a document belonging to the k classes. (Li et al., 2009) applied a regularization to encourage that the first factor is close to prior knowledge. This regularization is different from ours because it requires prior knowledge such as a predefined lexicon. A predefined lexicon needs a lot of human effort. (Zhou et al., 2014) applied a regularization based on an intuition that if two documents are sufficiently close to each other, they tend to share the same sentiment polarity. This intuition of this regularization is similar to ours. But when they compare document similarity, they use textual similarity (e.g., cosine similarity of bag of words) rather than similarity on opinion words. The regularization is applied under matrix factorization framework, it is not straightforward to fit in neural network based models.

Using keywords is another way to perform weakly supervised sentiment analysis. (Meng et al., 2018) leveraged keywords to generate pseudo-labeled samples for model pretraining, and then performed self-training on unlabeled data for model refinement. The possible improvement of this direction would be investigating more advanced keywords expansion techniques to generate better pseudo-labeled samples and developing a more advanced self-training algorithm. LOTClass (Meng et al., 2020), a parallel work to ours, fine-tuned a masked language model to generate relevant words that can replace label name such as “good” and “bad,” and performed self-training on unlabeled data for model refinement. Fine-tuning in LOTClass can be viewed as an advanced keyword expansion process using language models. VWS (Zeng et al., 2019) used target-opinion word pairs as supervision signal. Its objective function is to predict an opinion word given a target word. By introducing a latent variable (the sentiment polarity), they can learn a well-approximated posterior distribution via optimizing the evidence lower bound. The posterior probability here is the probability of a possible polarity (e.g., positive or negative) given text representation.

5 Conclusion

We propose a posterior regularization framework for the VWS sentiment analysis to better control the posterior distribution. The intuition behind the posterior regularization is that if extracted opinion words from two documents are semantically similar (dissimilar), the posterior distribution of two documents should be similar (dissimilar). Our experiments show that our posterior regularization can improve VWS and the performance is more stable.

Acknowledgments

We thank anonymous reviewers for their valuable comments. This paper was supported by the NSFC Grant U20B2053 from China, the Early Career Scheme (ECS, No. 26206717), the General Research Fund (GRF, No. 16211520), and the Research Impact Fund (RIF, No. R6020-19) from the Research Grants Council (RGC) of Hong Kong.

References

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: A system for large-scale machine learning. In *Proceedings of OSDI*, pages 265–283.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*, pages 740–750.

Anni Coden, Dan Gruhl, Neal Lewis, Pablo N Mendes, Meena Nagarajan, Cartic Ramakrishnan, and Steve Welch. 2014. Semantic lexicon expansion for concept-based aspect-aware sentiment analysis. In *Semantic Web Evaluation Challenge*, pages 34–40. Springer.

Kuzman Ganchev, Joao Gracca, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of SIGKDD*, pages 168–177.

Jaap Kamps, Maarten Marx, Robert J Mokken, Maarten De Rijke, et al. 2004. Using wordnet to measure semantic orientations of adjectives. In *Proceedings of LREC*, pages 1115–1118.

Tao Li, Yi Zhang, and Vikas Sindhwani. 2009. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of ACL*, pages 244–252.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL-HLT*, pages 142–150.

Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4:231–244.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *Proceedings of CIKM*, pages 983–992.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text classification using label names only: A language model self-training approach. In *Proceedings of EMNLP*, pages 9006–9017.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NeurIPS*, pages 3111–3119.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of COLING*, pages 1781–1796.

Malik Muhammad Saad Missen and Mohand Boughanem. 2009. Using wordnet's semantic relations for opinion detection in blogs. In *Proceedings of ECIR*, pages 729–733.

Aitor García Pablos, Montse Cuadros, and German Rigau. 2015. V3: Unsupervised aspect based sentiment analysis for semeval2015 task 12. In *Proceedings of SemEval*, pages 714–718.

Jonathon Read and John Carroll. 2009. Weakly supervised techniques for domain-independent sentiment classification. In *Proceedings of CIKM workshop on Topic-sentiment Analysis for Mass Opinion*, pages 45–52.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly D. Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.

Mikalai Tsytsarau, Themis Palpanas, and Kerstin Denecke. 2010. Scalable discovery of contradictions on the web. In *Proceedings of WWW*, pages 1195–1196.

Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL*, pages 417–424.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of EMNLP-HLT*, pages 347–354.

Ziqian Zeng, Wenxuan Zhou, Xin Liu, and Yangqiu Song. 2019. A variational approach to weakly supervised document-level multi-aspect sentiment classification. In *Proceedings of NAACL-HLT*, pages 386–396.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of NeurIP*, pages 649–657.

Guangyou Zhou, Jun Zhao, and Daojian Zeng. 2014. Sentiment classification with graph co-regularization. In *Proceedings of COLING*, pages 1331–1340.

Framing Word Sense Disambiguation as a Multi-Label Problem for Model-Agnostic Knowledge Integration

Simone Conia **Roberto Navigli**
Sapienza NLP Group
Department of Computer Science
Sapienza University of Rome
{conia,navigli}@di.uniroma1.it

Abstract

Recent studies treat Word Sense Disambiguation (WSD) as a single-label classification problem in which one is asked to choose only the best-fitting sense for a target word, given its context. However, gold data labelled by expert annotators suggest that maximizing the probability of a single sense may not be the most suitable training objective for WSD, especially if the sense inventory of choice is fine-grained. In this paper, we approach WSD as a multi-label classification problem in which multiple senses can be assigned to each target word. Not only does our simple method bear a closer resemblance to how human annotators disambiguate text, but it can also be extended seamlessly to exploit structured knowledge from semantic networks to achieve state-of-the-art results in English all-words WSD.

1 Introduction

Word Sense Disambiguation (WSD) is traditionally framed as the task of associating a word in context with its correct meaning from a finite set of possible choices (Navigli, 2009). Following this definition, recently proposed neural models were trained to maximize the probability of the most appropriate meaning while minimizing the probability of the other possible choices (Huang et al., 2019; Vial et al., 2019; Blevins and Zettlemoyer, 2020; Bevilacqua and Navigli, 2020). Although this training objective proved to be extremely effective and even led to Bevilacqua and Navigli (2020) reaching the estimated upper bound of inter-annotator agreement for WSD performance on the unified evaluation framework of Raganato et al. (2017b), adhering to it underplays a fundamental aspect of how human annotators disambiguate text. Indeed, past studies have observed that it is not uncommon for a word to have multiple appropriate meanings in a given context, meanings that can be used interchangeably under some circumstances because their boundaries are not clear cut (Tuggy, 1993; Kilgarriff, 1997; Hanks, 2000; Erk and McCarthy, 2009). This is especially evident if the underlying sense inventory is fine-grained, as the complexity, and therefore performance, of WSD is tightly coupled to sense granularity (Lacerra et al., 2020). The difficulty an annotator faces in choosing the most appropriate meaning from a fine-grained sense inventory becomes clear from an analysis of gold standard datasets: a non-negligible 5% of the target words are annotated with two or more sense labels in several gold standard datasets, including Senseval-2 (Edmonds and Cotton, 2001), Senseval-3 (Snyder and Palmer, 2004), SemEval-2007 (Pradhan et al., 2007), SemEval-2013 (Navigli et al., 2013), and SemEval-2015 (Moro and Navigli, 2015). Therefore, we follow Erk and McCarthy (2009), Jurgens (2012), and Erk et al. (2013), and argue that forcing a system to treat WSD as a single-label classification problem and learn that only one sense is correct for a word in a given context does not reflect how human beings disambiguate text.

In contrast to recent work, we approach WSD as a soft multi-label classification problem in which multiple senses can be assigned to each target word. We show that not only does this simple method bring significant improvements at low or no additional cost in terms of training and inference times and number of trainable parameters, but it can also be seamlessly extended to integrate senses from relational knowledge in structured form, e.g., similarity, hypernymy and hyponymy relations from semantic networks such as WordNet (Miller, 1995) and BabelNet (Navigli and Ponzetto, 2012). While structured knowledge has been naturally utilized by graph-based algorithms for WSD (Agirre and Soroa, 2009; Moro et al., 2014; Scozzafava et al., 2020), the incorporation of such information into

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3269–3275
April 19 - 23, 2021.

neural approaches has recently been garnering significant attention. However, currently available models can only take advantage of this knowledge with purposely-built layers (Bevilacqua and Navigli, 2020) that require additional complexities and/or trainable parameters. To the best of our knowledge, the work presented in this paper is the first to integrate structured knowledge into a neural architecture at negligible cost in terms of training time and number of parameters, while at the same time attaining state-of-the-art results in English all-words WSD.

2 Method

Single-label vs multi-label. WSD is the task of selecting the best-fitting sense s among the possible senses S_w of a target word w in a given context $c = \langle w_1, w_2, \dots, w_n \rangle$, where S_w is a subset of a predefined sense inventory S. Abstracting away from the intricacies of any particular supervised model for WSD, the output of a WSD system provides a probability y_i for each sense $s_i \in S_w$. Recently proposed machine learning models – Kumar et al., 2019; Barba et al., 2020; Blevins and Zettlemoyer, 2020; Bevilacqua and Navigli, 2020, *inter alia* – are trained to maximize the probability of the single most appropriate sense $\hat{s}$ by minimizing the cross-entropy loss $\mathcal{L}_{\text{CE}}$:

$$\mathcal{L}_{\text{CE}}(w, \hat{s}) = -\log(y_{\hat{s}}) \tag{1}$$

We observe that this loss function is only suitable for single-label classification problems. In the case of WSD, this is equivalent to assuming that there is just a single appropriate sense $\hat{s} \in S_w$ for the target word w in the given context c, that is, $\hat{s}$ is clearly dissimilar from any other sense in S_w. Indeed, minimizing the cross-entropy loss in order to maximize the probability of two or more senses generates conflicting training signals; at the same time, choosing to ignore one of the correct senses results in a loss of valuable information.

Since there is a not insignificant number of instances where multiple similar senses of the target word w fit the given context c (see Section 1), we frame WSD as a multi-label classification problem in which a machine learning model is trained to predict whether a sense $s \in S_w$ is appropriate for a word w in a given context c, independently of the other senses in S_w. This is simply equivalent to minimizing the binary cross-entropy loss $\mathcal{L}_{\text{BCE}}$ on the probabilities of the candidate senses S_w:

$$\begin{aligned}\mathcal{L}_{\text{BCE}}(w, \hat{S}_w) = &- \sum_{\hat{s} \in \hat{S}_w} \log(y_{\hat{s}}) \\ &- \sum_{s \in S_w \setminus \hat{S}_w} \log(1 - y_s)\end{aligned} \tag{2}$$

where $\hat{S}_w \subseteq S_w$ is the set of appropriate senses for the target word w in the given context c. We note that this simple yet fundamental change in paradigm does not come with an increased computational complexity as $|S_w|$ is usually small. Moreover, it is independent of the underlying model used to calculate the output probabilities and, therefore, it does not increase the number of trainable parameters.

Knowledge integration. If our model benefits from learning to assign multiple *similar* senses to a target word in a given context, then it makes sense that the very same model may also benefit from learning what *related* senses can be assigned to that word. For example, in the sentence *"the quick brown fox jumps over the lazy dog"*, our model may formulate a better representation of *fox* if it is also trained to learn that any fox is a canine (hypernymy relation) or that the fox species includes arctic foxes, red foxes, and kit foxes (hyponymy relations). In this way, not only would the model learn that canines, foxes and arctic foxes are closely related, but it would also learn that canines and arctic foxes may have the ability to jump, and this could act as a data augmentation strategy especially for those senses that do not appear in the training set.

There is a growing interest in injecting relational information from knowledge bases into neural networks but, so far, recent attempts have required purposely-designed strategies or layers. Among others, Kumar et al. (2019) aid their model with a gloss encoder that uses the WordNet graph structure; Vial et al. (2019) adopt a preprocessing strategy aimed at clustering related senses to decrease the number of output classes; Bevilacqua and Navigli (2020) introduce a logit aggregation layer that takes into account the neighboring meanings in the WordNet graph.

In contrast, our multi-labeling approach to WSD can be seamlessly extended to integrate relational knowledge from semantic networks such as WordNet without any increase in architectural complexity, training time, and number of trainable parameters. We simply relax the definition of the set of possible senses S_w for a word w to include all the senses related to a sense in S_w. More formally, let $G = (S, R)$ be a semantic network where S is a sense inventory and R is the set of semantic connections between any two senses. Then we define S_w^+ to also include every sense s_j that is connected to any sense $s_i \in S_w$ by an edge $(s_i, s_j) \in R$, that is, $S_w^+ = S_w \cup \{s_j : (s_i, s_j) \in R, s_i \in S_w\}$. The loss function is updated accordingly to maximize not only the probability of the correct senses, but also the probability of their related senses:

$$\begin{aligned} \mathcal{L}_{\text{BCE}}(w, \hat{S}_w^+) = &- \sum_{\hat{s} \in \hat{S}_w^+} \log(y_{\hat{s}}) \\ &- \sum_{s \in S_w^+ \setminus \hat{S}_w^+} \log(1 - y_s) \end{aligned} \tag{3}$$

where $\hat{S}_w^+ = \hat{S}_w \cup \{s_j : (\hat{s}_i, s_j) \in R, \hat{s}_i \in \hat{S}_w\}$. We note that the increase of the number of possible choices ($|S_w^+| \geq |S_w|$) and correct meanings ($|\hat{S}_w^+| \geq |\hat{S}_w|$) does not hinder the learning process since each probability is computed independently of the others. Finally, we stress that our approach to structured knowledge integration is completely model-agnostic, as it is independent of the architecture of the underlying supervised model.

Model description. In order to assess the benefits of our multi-labeling approach and avoid improvements that may not be related to the overall objective of this paper, we conduct our experiments with a simple WSD model. Similarly to Bevilacqua and Navigli (2020), this model is simply composed of BERT (large-cased, frozen), a non-linear layer, and a linear classifier. Thus, given a word w in context we build a contextualized representation $\mathbf{e}_w \in \mathbb{R}^{d_{\text{BERT}}}$ of the word w as the average of the corresponding hidden states of the last four layers of BERT, apply a non-linear transformation to obtain $\mathbf{h}_w \in \mathbb{R}^{d_\text{h}}$ with $d_\text{h} = 512$, and finally a linear projection to $\mathbf{o}_w \in \mathbb{R}^{|S|}$ to compute the sense scores. More formally:

$$\begin{aligned} \mathbf{e}_w &= \text{BatchNorm}\left(\frac{1}{4}\sum_{i=1}^{4} \mathbf{b}_w^{-i}\right) \\ \mathbf{h}_w &= \text{Swish}(\mathbf{W}_h \mathbf{e}_w + \mathbf{b}_h) \\ \mathbf{o}_w &= \mathbf{W}_o \mathbf{h}_w + \mathbf{b}_o \end{aligned}$$

where $\mathbf{b}_w^{-i}$ is the hidden state of the i-th layer of BERT from the topmost one, $\text{BatchNorm}(\cdot)$ is the batch normalization operation, and $\text{Swish}(x) = x \cdot \text{sigmoid}(x)$ is the Swish activation function (Ramachandran et al., 2017).

3 Experiments and Results

Experimental setup. We train our models in different configurations to assess the individual contribution of several factors. First, we compare our baseline model trained with a single-label objective (Equation 1) to the same model trained with a multi-label objective (Equation 2). Then, we gradually include structured knowledge in the form of WordNet relations using Equation 3, starting from similarity relations (similar-to, also-see, verb-group, and derivationally-related-form), and incrementally including generalization and specification relations (hypernymy, hyponymy, instance-hypernymy, instance-hyponymy). In order to keep a level playing field with single-label systems, we choose only the meaning with highest probability for our multi-label models.

Datasets. We evaluate the models on the Unified Evaluation Framework for English all-words WSD proposed by Raganato et al. (2017b). This evaluation includes five gold standard datasets, namely, Senseval-2, Senseval-3, SemEval-2007, SemEval-2013, and SemEval-2015. Following standard practice we use the smallest gold standard as our development set, SemEval-2007, and the remaining ones as test sets. We distinguish between two settings: closed and open. In the former setting, we include systems that only use SemCor (Miller et al., 1994) as the training corpus, while in the latter we also include those systems that use WordNet glosses and examples and/or Wikipedia.

Hyperparameters. We use the pretrained version of BERT-large-cased (Devlin et al., 2019) available on HuggingFace's Transformers library (Wolf et al., 2020) to build our contextualized embeddings (Section 2). BERT is left frozen, that is, its parameters are not updated during training. Each model is trained for 25 epochs using Adam (Kingma and Ba, 2015) with a learning rate of 10^{-4}. We avoid hyperparameter tuning and opt for values that are close to the ones reported in the literature so as to have a fairer comparison.

Comparison systems. In order to have a comprehensive comparison with the current state of the art in WSD, we include the work of:

							Concatenation of ALL datasets				
		SE2	**SE3**	**SE07**	**SE13**	**SE15**	**Nouns**	**Verbs**	**Adj**	**Adv**	**ALL**
SemCor only	Raganato et al. (2017a)	72.0	69.1	64.8	66.9	71.5	71.5	57.5	75.0	83.8	69.9
	BERT$_{\text{Large}}$	76.3	73.2	66.2	71.7	74.1	–	–	–	–	73.5
	Hadiwinoto et al. (2019)	75.5	73.6	68.1	71.1	76.2	–	–	–	–	73.7
	Peters et al. (2019)	–	–	–	–	–	–	–	–	–	75.1
	Vial et al. (2019)	–	–	–	–	–	–	–	–	–	75.6
	Vial et al. (2019) - *Ensemble*	77.5	77.4	69.5	76.0	**78.3**	79.6	65.9	79.5	85.5	76.7
	This work	**78.4**	**77.8**	**72.2**	**76.7**	78.2	**80.1**	**67.0**	**80.5**	**86.2**	**77.6**
SemCor + definitions / examples	Loureiro and Jorge (2019)	76.3	75.6	68.1	75.1	77.0	78.0	64.0	80.7	84.5	75.4
	Scarlini et al. (2020a)	–	–	–	78.7	–	80.4	–	–	–	–
	Conia and Navigli (2020)	77.1	76.4	70.3	76.2	77.2	78.7	65.6	81.1	84.7	76.4
	Bevilacqua et al. (2020)	78.0	75.4	71.9	77.0	77.6	79.9	64.8	79.2	86.4	76.7
	Huang et al. (2019)	77.7	75.2	72.5	76.1	80.4	–	–	–	–	77.0
	Scarlini et al. (2020b)	78.0	77.1	71.0	77.3	83.2	80.6	68.3	80.5	83.5	77.9
	Blevins and Zettlemoyer (2020)	79.4	77.4	74.5	79.7	81.7	81.4	68.5	83.0	**87.9**	79.0
	Bevilacqua and Navigli (2020)	**80.8**	**79.0**	75.2	80.7	81.8	**82.9**	69.4	82.9	87.6	80.1
	This work	80.4	77.8	**76.2**	**81.8**	**83.3**	**82.9**	**70.3**	**83.4**	85.5	**80.2**

Table 1: WSD results in F$_1$ scores on Senseval-2 (SE2), Senseval-3 (SE3), SemEval-2007 (SE07), SemEval-2013 (SE13), SemEval-2015 (SE15), and the concatenation of all the datasets (ALL). Top: closed setting (only SemCor allowed as the training corpus without definitions and/or examples). Bottom: open setting (WordNet glosses and examples are also used for training).

WSD	Sim	See	Rel	Vrb	Hpe	Hpo	Hpe$_I$	Hpo$_I$	**SE07**	**ALL**
SL	–	–	–	–	–	–	–	–	69.0	74.7
ML	–	–	–	–	–	–	–	–	69.2	75.7
ML	✔	✔	✔	✔	–	–	–	–	70.6	76.6
ML	✔	✔	✔	✔	✔	–	–	–	71.0	77.0
ML	✔	✔	✔	✔	–	✔	–	–	**72.5**	77.4
ML	✔	✔	✔	✔	✔	✔	–	–	72.2	**77.6**
ML	✔	✔	✔	✔	✔	✔	✔	✔	72.2	**77.6**

Table 2: WSD results in F$_1$ scores on SemEval-2007 (SE07) and the concatenation of all the datasets (ALL). SL/ML: single-label/multi-label. Sim: similar-to. See: also-see. Rel: derivationally-related-forms. Vrb: verb-groups. Hpe: hypernymy. Hpo: hyponymy. Hpe$_I$: instance-hypernyms. Hpo$_I$: instance-hyponyms.

- Raganato et al. (2017a) which was one of the first to propose a neural sequence model for WSD based on a stack of BiLSTM layers;
- BERT$_{\text{large}}$, a simple 1-neareast-neighbor approach based on the last hidden state of the BERT-large-cased model (Loureiro and Jorge, 2019);
- Hadiwinoto et al. (2019) which was among the first to exploit pretrained contextualized models for WSD;
- Peters et al. (2019) which incorporated WSD knowledge directly into the training process of BERT;
- Huang et al. (2019) which tasked the model to learn which gloss is the most appropriate for a word in context;
- Bevilacqua et al. (2020) which tackled WSD as a gloss generation problem;
- Loureiro and Jorge (2019) and Conia and Navigli (2020) which created and enhanced sense embeddings with relational knowledge from WordNet and BabelNet;
- Scarlini et al. (2020a) which proposed nominal sense embeddings built by exploiting BabelNet to automatically retrieve sense-specific context;
- Scarlini et al. (2020b) which extended the above approach to non-nominal senses and multiple languages;
- alongside the aforementioned work of Vial et al. (2019), Blevins and Zettlemoyer (2020), and Bevilacqua and Navigli (2020).

The systems are divided into two groups in Table 1: in the upper part we compare our approach against those systems that do not take advantage of information coming from WordNet glosses and/or examples, while in the lower part we also include those systems that make use of such knowledge.

Results. The first two rows of Table 2 show the results of switching from a single-label to a multi-label approach for WSD: this single change already brings a significant improvement in performance (+1.0% in F_1 score, significant with $p < 0.1$, χ^2 test). Not only that, increasing the number and variety of WordNet relations further increases the performance of the model, with hyponyms being particularly beneficial (+0.8% in F_1 score). Unfortunately, including instance hypernyms and instance hyponyms does not bring further improvements; this may be due to the relatively low number of instances that can take advantage of such relations in SemCor.

Nonetheless, the results obtained set a new state of the art among single and ensemble systems trained only on SemCor without the use of additional training data or resources external to WordNet such as Wikipedia, surpassing the previous state-of-the-art non-ensemble system of Vial et al. (2019) by 2.0% in F_1 score (significant with $p < 0.05$, χ^2 test), as shown in Table 1. When further trained on the WordNet glosses and examples, our model attains state-of-the-art results (+1.2% and +0.1% in F_1 score compared to the systems of Blevins and Zettlemoyer (2020) and Bevilacqua and Navigli (2020), respectively) despite being simpler than most of the techniques it is compared against.

4 Conclusion

WSD is a key task in Natural Language Understanding with several open challenges and with the granularity of sense inventories being undoubtedly the most pressing issue (Navigli, 2018). We departed from recent work on WSD and investigated the effect of tackling the task as a multi-label classification problem. Not only is our approach simple and model-agnostic, but it can also be seamlessly extended to integrate relational knowledge in structured form from semantic networks such as WordNet, and at no extra cost in terms of architectural complexity, training times, and number of parameters.

Our experiments show that our method, thanks to its more comprehensive notion of loss over equally valid and structurally-related senses, achieves state-of-the-art results in English all-words WSD, especially when there is a lower amount of annotated text available. These results open the path to further research in this direction, from exploring more complex models and richer knowledge bases to exploiting multiple labels in innovative disambiguation settings which can overcome the fine granularity of sense inventories. Not only that, our knowledge integration approach could potentially be applied to address the knowledge acquisition bottleneck in multilingual WSD (Pasini, 2020; Pasini et al., 2021). Finally, with the rise of ever more complex general and specialized pretrained models, we believe that our simple model-agnostic approach can be another step towards knowledge-based (self-)supervision.

We release our software and model checkpoints at `https://github.com/SapienzaNLP/multilabel-wsd`.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme.

This work was supported in part by the MIUR under grant "Dipartimenti di eccellenza 2018-2022" of the Department of Computer Science of Sapienza University.

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *EACL 2009, 12th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, Athens, Greece, March 30 - April 3, 2009*.

Edoardo Barba, Luigi Procopio, Niccolò Campolungo, Tommaso Pasini, and Roberto Navigli. 2020. MuLaN: Multilingual label propagation for Word Sense Disambiguation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or: "How we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in Word Sense Disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association* *for Computational Linguistics, ACL 2020, Online, July 5-10, 2020.*

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of Word Sense Disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020.*

Simone Conia and Roberto Navigli. 2020. Conception: Multilingually-enhanced, human-readable concept vector representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online).

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

Philip Edmonds and Scott Cotton. 2001. SENSEVAL-2: Overview. In *Proceedings of Second International Workshop on Evaluating Word Sense Disambiguation Systems, SENSEVAL@ACL 2001, Toulouse, France, July 5-6, 2001.*

Katrin Erk and Diana McCarthy. 2009. Graded word sense assignment. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL.*

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. 2013. Measuring word meaning in context. *Computational Linguistics*, 39(3):511–554.

Christian Hadiwinoto, Hwee Tou Ng, and Wee Chung Gan. 2019. Improved Word Sense Disambiguation using pre-trained contextualized word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019.*

Patrick Hanks. 2000. Do word meanings exist? *Comput. Humanit.*, 34(1-2):205–215.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for Word Sense Disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019.*

David Jurgens. 2012. An evaluation of graded sense disambiguation using word sense induction. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics, *SEM 2012, June 7-8, 2012, Montréal, Canada.*

Adam Kilgarriff. 1997. I don't believe in word senses. *Computers and the Humanities*, 31(2):91–113.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.*

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha P. Talukdar. 2019. Zero-shot Word Sense Disambiguation using sense definition embeddings. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers.*

Caterina Lacerra, Michele Bevilacqua, Tommaso Pasini, and Roberto Navigli. 2020. CSI: A coarse sense inventory for 85% word sense disambiguation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020.*

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through wordnet for full-coverage Word Sense Disambiguation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers.*

George A. Miller. 1995. WordNet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994.*

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 task 13: Multilingual all-words Sense Disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015.*

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: A Unified Approach. *Transactions of the Association for Computational Linguistics (TACL)*, 2.

Roberto Navigli. 2009. Word Sense Disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69.

Roberto Navigli. 2018. Natural language understanding: Instructions for (present and future) use. In *IJCAI*, pages 5697–5702.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 task 12: Multilingual Word Sense Disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2013, Atlanta, Georgia, USA, June 14-15, 2013*.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Tommaso Pasini. 2020. The knowledge acquisition bottleneck problem in multilingual Word Sense Disambiguation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4936–4942.

Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. XL-WSD: An extra-large and cross-lingual evaluation framework for word sense disambiguation. In *Proc. of AAAI*.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval@ACL 2007, Prague, Czech Republic, June 23-24, 2007*.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017a. Neural sequence learning models for Word Sense Disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*.

Alessandro Raganato, José Camacho-Collados, and Roberto Navigli. 2017b. Word Sense Disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2017. Searching for activation functions. *arXiv*, abs/1710.05941.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020a. SensEmBERT: Context-enhanced sense embeddings for multilingual Word Sense Disambiguation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020b. With more contexts comes better performance: Contextualized sense embeddings for all-round Word Sense Disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online.

Federico Scozzafava, Marco Maru, Fabrizio Brignone, Giovanni Torrisi, and Roberto Navigli. 2020. Personalized PageRank with syntagmatic information for multilingual Word Sense Disambiguation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), demos*, Online.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, SENSEVAL@ACL 2004, Barcelona, Spain, July 25-26, 2004*.

David Tuggy. 1993. Ambiguity, polysemy, and vagueness. *Cognitive Linguistics*, 4:273–290.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2019. Sense vocabulary compression through the semantic knowledge of WordNet for neural Word Sense Disambiguation.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Graph-based Fake News Detection using a Summarization Technique

Gihwan Kim and Youngjoong Ko
Department of Computer Science and Engineering
Sungkyunkwan University
Suwon, Gyeonggi-do, Korea
`kimgihwan3364@gmail.com, yjko@skku.edu`

Abstract

Nowadays, fake news is spreading in various ways, and this fake information is causing a lot of social damages. Thus the need to detect fake information is increasing to prevent the damages caused by fake news. In this paper, we propose a novel graph-based fake news detection method using a summarization technique that uses only the document internal information. Our proposed method represents the relationship between all sentences using a graph and the reflection rate of contextual information among sentences is computed by using an attention mechanism. In addition, we improve the performance of fake news detection by utilizing summary information as an important subject of the document.The experimental results demonstrate that our method achieves high accuracy, 91.04%, that is 8.85%p better than the previous method.

1 Introduction

Recently, people are easily exposed to large amount of information in various ways with the development of information propagation methods. But some of the large amount of information contains fake information generated for malicious purposes. This fake information is confusing people and causing a lot of social and economic damages. Therefore, the need to detect fake information is increasing to prevent the damages caused by fake news, and it is being researched industrially and academically (Yang et al., 2012; Castillo et al., 2011; Yan et al., 2015).

Prior fake news detection studies have methods of detecting fake news using external information as well as internal information. The method of using internal information detects fake news by analyzing linguistic features in news, such as news content (Levi et al., 2019), writing styles and consistency (Potthast et al., 2018), and relational structure between sentences (Karimi and Tang, 2019) within the news. On the other hand, that of using external information does by analyzing metadata such as aspects of news spreading (Monti et al., 2019) and user profiles of the people spreading the news (Lu and Li, 2020). However, a collection task for external information requires a lot of time and cost, and it is very difficult to identify external information in all documents. In addition, it is more basic and important to understand a document with internal information, and the construction of structural relationships among sentences within a document is one of effective methods for detecting fake news. Therefore, we propose an effective fake news detection method by structuring a context graph for representing the relationships among sentences based on summarization information.

Since all sentences in a document are strongly related to each other, the contextual information with other sentences should be reflected to generate sentence embeddings. In addition, the attention mechanism is exploited for constructing a context graph so that the different relation strength between sentences influences the contextual information of each sentence in the graph. In the context graph, nodes consist of initial sentences embeddings for all the sentences in a document and the weight of an edge is estimated by an attention score between two ended nodes of the edge. We assume that all the nodes are connected because all the sentence in a document is strong related each other. Then a contextualized sentence embedding on each node is generated by reflecting the neighbor's contextual information. The contextualized sentence embedding of a node is computed by the sum of the products of the attention score between the node and its neighbor node and the initial sentence embedding of the neighbor node.

Because the subject of a document is very im-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3276–3280
April 19 - 23, 2021.

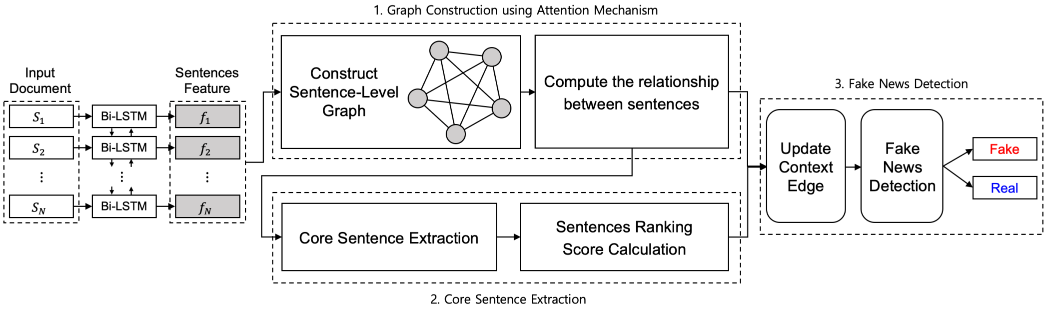


Figure 1: The architecture of our proposed model.

portant information in identifying the content of the document and fake information is commonly related to the subject, we apply a summarization technique (Jeong et al., 2016) that can effectively capture the subject information of the document to fake news detection. By using a summarization technique, sentences containing a lot of subject information are highly ranked and the ranked scores influence the attention scores for construction of the context graph.

For performance comparison with the proposed method, we implemented a baseline model that detects fake news with the sum of all sentence embeddings in a document. Our proposed model shows 91.04% accuracy, that is 11.19%p better performance than baseline model. It also shows 8.85%p better performance compared to other model (Karimi and Tang, 2019) that uses the same dataset and a dependency tree structure among sentences in a document.

2 Notations

We have a corpus D of fake and real news documents. Let a document $d \in D$ contain N sentences $s_1, s_2, ...s_N$ and each sentence $s_i \in d$ include words $word_1, word_2, ...word_l$ where l denotes the number of words in sentence s_i. We apply the Bi-LSTM network to all sentences of a document and obtain the initial sentence embedding representations $\mathcal{H} = h_1, h_2, ...h_N \in \mathbb{R}^{N \times dim}$.

3 Proposed Method

We propose a novel fake news detection method based on the context graph with a summarization technique (see Figure 1). Our method consists of three components. The first one is *graph construction using attention mechanism* for representing the relationship between all sentences using a graph. The second one is *core sentence extraction* for ranking sentences with subject information using a summarization technique. The third one is *fake news detection* for discriminating fake news from a lot of documents.

3.1 Graph Construction using Attention Mechanism

To model the relationship between sentences and their contextual information, we construct a graph $\mathcal{G} = (F, E)$. The graph $\mathcal{G}$ is composed of node $(i.e., F = f_1,\ f_2,\ \ldots f_N)$ and the edge $(i.e.,\ E = e_{1,2},\ e_{ij},\ \ldots e_{N-1,N})$. Each node is represented by sentence embedding H and the edge between the i-th node and the j-th node is represented by $e_{i,j}$ and its weight $w_{i,j}$ means the relation strength of the i-th sentence for the j-th sentence. Since all sentences in a document are strongly related to each other, we consider that $\mathcal{G}$ is a fully connected graph. To reflect the different relation strength between sentences, each edge $e_{i,j}$ is associated with a weight $w_{i,j}$, and the weight is derived by the attention mechanism between sentence embeddings h_i and h_j of two nodes f_i and f_j as follows (Eq. 1-3):

$$x_i = ReLU(Wf_i + bias) \tag{1}$$

$$x_j = ReLU(Wf_j + bias) \tag{2}$$

$$w_{i,j} = x_i U x_j \tag{3}$$

where $W \in \mathbb{R}^{m \times dim}$ and $x \in \mathbb{R}^m$. The dimension size of sentence embedding can be reduced to avoid overfitting without weakening the LSTM's capacity (Dozat and Manning, 2018) by Equations (1) and (2). U is a weight matrix to compute the relation strength between x_i and x_j. $w_{i,j}$ represent an edge weight between pair of nodes f_i and f_j in the graph $\mathcal{G}$.

3.2 Core Sentence Extraction

Based on the constructed graph, the subject information of a sentence is estimated by summing attention scores (edge scores) between the node of the sentence and its adjacent nodes, and a sentence with the most subject information is extracted as a core sentence. $Core_{sent}$ represents an extracted sentence.

$$Core_{sent} = \operatorname*{argmax}_{1 \leq i \leq N} \sum_{j=1, j \neq i}^{N} w_{i,j} \tag{4}$$

Based on the cosine similarity values between the core sentence and all other sentences, all sentences in the document are divided into a subject relevant sentence set and an irrelevant sentence set. The subject relevance score of each word is based on the frequency of appearance in the subject relevant and irrelevant sentence sets. The word relevance score RS_d is calculated by Equation (5) (Jeong et al., 2016).

$$\begin{aligned} RS_d\,(word) &= log\frac{p_d \times (1 - q_d)}{(1 - p_d) \times q_d} \\ &= log\frac{(r_d + 0.5)\,(S_d - s_d + 0.5)}{(R_d - r_d + 0.5)\,(s_d + 0.5)} \end{aligned} \tag{5}$$

where p_d and q_d are the probabilities that a word appears in subject relevant and irrelevant sentences set in a document d. Respectively, R_d and S_d are the number of subject relevant sentences and irrelevant sentences in document d. r_d and s_d are the number of subject relevant and irrelevant sentences that include the word in a document d, and 0.5 is a naïve smoothing factor to avoid zero-denominator or log-zero. On this paper, the top 30% of sentences with the subject relevance score were selected as the subject relevant sentence set, and the others were as the irrelevant sentence set. Afterward, the sentence score can be calculated with the sum of relevance score of words included in the sentence (Jeong et al., 2016).

$$Score(s_i) = \sum_{word \in s_i} RS_d(word) \tag{6}$$

3.3 Fake News Detection

After we calculate the sentence ranking using the sentence scores by Equation (6), we can construct a new updated graph by reflecting the sentence ranking into the weight of the edge. The weight of the edge represents the reflection rate of the subject information as well as the context information.

$$R_{Score(s_i)} = 1 - \frac{Sentence\_Rank(s_i) - 1}{N} \tag{7}$$

$$w'_{i,j} = w_{i,j} * R_{Score(s_i)} \tag{8}$$

$$(for\ i, j\ =\ 1,\ 2,\ ...,\ N\ and\ i \neq j)$$

The sentence ranking score is calculated by Equation (7) and the weight of all edges is updated by Equation (8). Subsequently, we can construct subject contextualized sentence embeddings using the updated graph. The subject contextualized sentence embedding (i.e., h'_i) is created by the weighted sum of the initial sentence embedding of the current sentence (i.e., h_i) and those of the other adjacent sentences (i.e., h_j) as follows:

$$h'_i = h_i + \sum_{j=1, j \neq i}^{N} w'_{i,j} * h_j \tag{9}$$

Finally, we create a document embedding by averaging the embedding vectors of all sentences in a document and then the document embedding is then fed into a multi-layer feedforward neural network to predict label, fake or real, by the binary classification, as a vector $\hat{y}$.

$$doc = \sum_{i=1}^{N} h'_i \tag{10}$$

$$\hat{y} = Softmax(tanh(Wdoc + bias)) \tag{11}$$

The cross-entropy loss function is used to optimize our neural network.

4 Experiments

4.1 Datasets

We used HDSF dataset (Karimi and Tang, 2019) for our fake news detection experiments. The dataset consists of 3,360 real documents and 3,360 fake documents. We follow the Karimi's data split: 6,452 documents for training data, 134 ones for validation set, and 134 ones for test set and each data split contains even number of documents from fake and real classes. In addition, we did experiments on 5-folds cross-validation because the original HDSF dataset contain too small size of test documents.

4.2 Experimental Settings

We used word2vec embedding (Mikolov et al., 2013) that is pre-trained from Google as an initial word embedding, and set the Bi-LSTM hidden unit size to 200. We set the epoch as 200, mini-batch size as 40, and dropout as 30% on each experiment. We used the Adam optimizer (Kingma and Ba, 2014) and set the initial learning rate as 0.001 and reduce by 10 times for every 50 epochs. We used accuracy as the metric of performance.

4.3 Models

The proposed model is compared to the baseline model and Karimi's model (Karimi and Tang, 2019) to prove a superiority of our proposed model in this subsection.

Karimi's Model (Karimi and Tang, 2019) A fake detection method to predicts whether a document is fake or real by constructing relationships of each sentence in a document using the hierarchical discourse-level dependency tree.

Baseline Model The baseline model uses the document embedding only using Bi-LSTM and predicts whether it is fake or real through the feed-forward neural network.

Graph Model This graph model connects all sentences in a document by constructing a graph structure. Fake news is detected by creating sentence embeddings that only reflect contextual information on a graph.

Graph + Summarization Model (Proposed) The graph + summarization model is our final proposed model. It classifies the fake or real documents after representing a document embedding that reflect the contextual and subject information on a graph structure by the summarization technique.

4.4 Experiment Results

Table 1 shows a comparison of the experiment results. The RST, LIWC, N-grams, and BiGRNN-CNN models implemented by (Karimi and Tang, 2019). Our proposed model showed best performance compared to other models and it achieved better performance than the baseline model about 11.19%p, and even better than the Karimi's model about 8.85%p. In addition, we obtained 3.73%p improvement when summarization technique is used in the graph model. Moreover, the final proposed model showed the best performance even in the 5-folds cross-validation experiments with 9.06%p improvement as well.

Model	**Data**	
	HDSF	Cross Validation
RST	67.68	-
LIWC	70.26	-
N-grams	72.37	-
BiGRNN-CNN	77.06	-
Karimi's	82.19	81.71
Baseline	79.85	79.27
Graph Model	87.31	84.52
Graph + Summarization (Proposed)	**91.04** **(+11.19%p∗)** **(+8.85%p⋆)**	**88.33** **(+9.06%p∗)** **(+6.62%p⋆)**

Table 1: Comparing results. The ∗ and ⋆ denote the differences between the proposed model and the Baseline or Karimi's.

5 Analysis

Herein, we introduce two analyses by an additional experiment for subject consistency detection in our model and dataset. The proposed method attempted to effectively detect subject information in fake news and the consistency of subject information is also important to detect fake news. In the first analysis, we verify our process of updating the graph in the proposed model, which uses a fully connected graph, by comparing other model using a 50% connected graph, which uses edges with only the top 50% with high attention scores. As a result, the proposed model with a fully connect graph showed 1.29%p higher performance than the model with a 50% connected graph. We think it means that a fully connect graph is more useful to detect fake news. Secondly, we observed the variances of the attention scores in the fake and real documents (see Table 2). The variance in the fake documents is higher than that of the real documents and it means that the fake documents has an inconsistent subject distribution.

	Variance	Standard Deviation
Fake	8.6	2.93
Real	5.41	2.33

Table 2: Variance/standard deviation of attention scores

6 Conclusions

We have proposed a novel graph based fake news detection method using summarization technique. Our model shows that the use of contextual and subject information is helpful in detecting fake news. Our final proposed model achieved better performance than the baseline model about 11.19%p.

Acknowledgements

This work was supported in part by Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques), in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1A2C2100362), and in part by the MSIT(Ministry of Science and ICT), Korea, under the ICT Creative Consilience program (IITP-2020-0-01821) supervised by the IITP.

References

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Hyoungil Jeong, Youngjoong Ko, and Jungyun Seo. 2016. How to improve text summarization and classification by mutual cooperation on an integrated framework. *Expert Systems with Applications*, 60:222–233.

Hamid Karimi and Jiliang Tang. 2019. Learning hierarchical discourse-level structure for fake news detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3432–3442, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Or Levi, Pedram Hosseini, Mona Diab, and David Broniatowski. 2019. Identifying nuances in fake news vs. satire: Using semantic and linguistic cues. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 31–35, Hong Kong, China. Association for Computational Linguistics.

Yi-Ju Lu and Cheng-Te Li. 2020. GCAN: Graph-aware co-attention networks for explainable fake news detection on social media. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 505–514, Online. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M Bronstein. 2019. Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 231–240, Melbourne, Australia. Association for Computational Linguistics.

Rui Yan, Ian EH Yen, Cheng-Te Li, Shiqi Zhao, and Xiaohua Hu. 2015. Tackling the achilles heel of social networks: Influence propagation based language model smoothing. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1318–1328.

Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD workshop on mining data semantics*, pages 1–7.

Cognition-aware Cognate Detection

Diptesh Kanojia$^{\dagger,\clubsuit,\star}$, **Prashant Sharma**$^{\diamond}$, **Sayali Ghodekar**‡,
Pushpak Bhattacharyya†, **Gholamreza Haffari***, and **Malhar Kulkarni**†
†IIT Bombay, India, $^{\diamond}$Hitachi CRL, Japan, ‡RingCentral, India
$^{\clubsuit}$IITB-Monash Research Academy, India
*Monash University, Australia
†{diptesh,pb,malhar}@iitb.ac.in, $^{\diamond}$prashaantsharmaa@gmail.com
‡sayalighodekar26@gmail.com , *gholamreza.haffari@monash.edu

Abstract

Automatic detection of cognates helps downstream NLP tasks of Machine Translation, Cross-lingual Information Retrieval, Computational Phylogenetics and Cross-lingual Named Entity Recognition. Previous approaches for the task of cognate detection use orthographic, phonetic and semantic similarity based features sets. In this paper, we propose a novel method for enriching the feature sets, with cognitive features extracted from human readers' gaze behaviour. We collect gaze behaviour data for a small sample of cognates and show that extracted cognitive features help the task of cognate detection. However, gaze data collection and annotation is a costly task. We use the collected gaze behaviour data to predict cognitive features for a larger sample and show that predicted cognitive features, also, significantly improve the task performance. We report improvements of 10% with the collected gaze features, and 12% using the predicted gaze features, over the previously proposed approaches. Furthermore, we release the collected gaze behaviour data along with our code and cross-lingual models.

1 Introduction

Cognates are word pairs, across languages, which have a common etymological origin. For example, the French and English word pair, *Liberté - Liberty*, reveals itself to be a cognate through orthographic similarity. The task of automatic cognate detection across languages[1] requires one to detect word pairs which are etymologically related, and carry the same meaning. Such word pairs share a formal and/or semantic affinity and facilitate the second language acquisition process, particularly between related languages. Although they can accelerate vocabulary acquisition, language learners also need to be aware of false friends and partial cognates, at times, leading to unrelated semantic coupling. For example, *"gift"* in German means *"poison"*, which is a known example of a False Friend pair. For an example of a partial cognate, the word *"police"* can translate to *"police"*, *"policy"* or *"font"*, in French, depending on the context in which it was used.

Manual detection of such cognate sets requires a human expert with a good linguistic background in multiple languages. Moreover, manual annotation of cognate sets is a costly task in terms of time and human effort. Automatic Cognate Detection (ACD) is a well-known task, which has been explored for a range of languages using different methods; and has shown to help Cross-lingual Information Retrieval (Meng et al., 2001), Machine Translation (MT) (Al-Onaizan et al., 1999), and Computational Phylogenetics (Rama et al., 2018). In the traditional approaches to automatic cognate detection, words with similar meanings or forms are used as probable cognates (Jäger et al., 2017; Rama, 2016; Kondrak, 2001). From such pairs, the ones that exhibit a high phonological, lexical and/or semantic similarity, are analyzed in order to find true cognates. Merlo and Andueza Rodriguez (2019) perform an investigation to evaluate the use of cross-lingual embeddings and show that these models inherit a lexical structure matching the bilingual lexicon. This study establishes that cross-lingual models can provide an effective similarity score compared to their monolingual counterparts, for both cognates and false friends. However, they do not evaluate machine learning (ML) approaches that distinguish between cognates and false friends. The absence of such an evaluation motivates us to use cross-lingual similarity scores with ML algorithms. Our work[2] reports whether these scores can provide an adequate distinction or not.

---

[1] Cognates can also exist in the same language. Such word pairs/sets are commonly referred to as *doublets*.

[2] Dataset and Code

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3281–3292
April 19 - 23, 2021.

	Hindi (Hi)	**Marathi (Mr)**	**Hindi Meaning**	**Marathi Meaning**
Cognate	ank	ank	Number	Number
False Friend	shikshA	shikshA	Education	Punishment

Table 1: An example each of a cognate and a false friend pair from Indian languages, Hindi and Marathi.

Inspired from their work, we investigate the use of cross-lingual word embeddings and cognitive features to distinguish between cognates and false friends, for the Indian language pair of Hindi and Marathi. Cognitively inspired features have shown to improve various NLP tasks (Mishra et al., 2018a). However, most of their work involves collecting the gaze behaviour data first on a large sample, and then splitting the data into training and testing data, before performing their experiments. While their work does show significant improvements over baseline approaches, across multiple NLP tasks, collecting gaze behaviour over a large cognate dataset can be costly, both in terms of time and money. Our approach tries to reduce annotation cost by predicting gaze behaviour data for a large sample based on the smaller sample of collected gaze data. Our investigations use three recently proposed cross-lingual word embeddings based approaches to generate features for the task of cognate detection. We also generate cognitive features from participants' gaze behaviour data and show that gaze behaviour helps the task of cognate detection. Additionally, we use the collected gaze behaviour data and predict gaze-based features for a much larger sample. We believe that using gaze features will be more applicable only if gaze features can be predicted for unseen samples. We believe that collection of gaze data cannot be performed in all the scenarios and hence hypothesize that predicting such data if it helps improve the task of cognate detection, should be a viable solution.

Motivation

Consider a scenario where an NLP task comes across the false friend pair in Table 1. Orthographic similarity or even phonetic similarity-based techniques will fail to detect the difference between the Hindi meaning of the word "shikhshA" and its Marathi counterpart. Here, semantic approaches should detect the distinction in meaning, but monolingual embeddings are trained using a large corpus **from the same language**. In such cases, it becomes imperative that a cross-lingual word embeddings model be utilized. However, Indian languages are known to be low-resource languages compared to English or even many European languages like French, Italian, German *etc.* Acquiring additional clean data for training cross-lingual models is, yet again, a painful task. In such a scenario then, we ask ourselves,

"Can cognitive features be used to help the task of Cognate Detection?"

furthermore,

"Using gaze features collected on a small set of data points, can we predict the same features on a larger set of data points to alleviate the need for collecting gaze data?"

With this work, we try to answer both the questions stated above and present the rest of the paper as follows. We discuss the current literature on the cognate detection task and cognitive NLP in Section 2. Section 3 discusses the dataset acquisition, including the description and analysis of our gaze behaviour data. We describe the feature sets used in section 4. Our approaches to the task of cognate detection are discussed in Section 5. The results of our work are discussed in Section 6. Finally, we conclude the study with possible future work in this direction in Section 7.

Terminology

An *interest area (IA)* is an area of the annotation screen to be processed by the human reader. In our experiments, it is an area where a word is shown to the reader. A *fixation* is an event where the reader focuses within an "interest area". A *saccade* is the movement of the eye from one fixation point to another. If the saccades move from an earlier IA to a later IA, such a saccade is called a *progression*. A *regression* is the saccade path when the reader moves back to a previous IA. We also use the terms *reader* and *participant* interchangeably.

2 Related Work

Current literature which uses gaze behaviour to solve downstream NLP tasks has been applied to the NLP tasks of sentiment analysis (Mishra et al., 2018a; Barrett et al., 2018; Long et al., 2019), sarcasm detection (Mishra et al., 2016), grammatical error detection (Barrett et al., 2018), hate speech detection (Barrett et al., 2018), named entity recognition (Hollenstein and Zhang, 2019), part-of-speech tagging (Barrett et al., 2016), sentence simplification (Klerke et al., 2016), and readability (González-Garduño and Søgaard, 2018; Singh et al., 2016). A comprehensive overview is provided by Mishra and Bhattacharyya (2018). The primary motivation of using cognitive features for NLP tasks is derived from the eye-mind hypothesis (Just and Carpenter, 1980), which establishes a direct correlation between a reader's comprehension of the text with the time taken to read the text. This hypothesis has initiated a large body of psycholinguistic research that shows a relationship between text processing and gaze behaviour. Yaneva et al. (2020) discuss the use of gaze features for the task of anaphora resolution. Their findings show that gaze data can substitute the classical text processing approaches along with the fact that human disambiguation process overlaps with the information carried in linguistic features. Rohanian (2017) use gaze data to automatically identify multiword expressions and observe that gaze features help improve the accuracy of the task when combined with traditional linguistic features used for the task.

Mathias et al. (2020) describe an approach to scoring essays in a multi-task learning framework automatically. Their approach relies on collecting gaze behaviour for essay reading for a small set of essays and then predicting the rest of the dataset's gaze behaviour in a multi-task learning setup. Similarly, Barrett et al. (2016) use token level averages of cognitive features at run time, to mitigate the need for these features at run time. Singh et al. (2016) and Long et al. (2019) predict gaze behaviour at the token-level as well. Mishra et al. (2018a), González-Garduño and Søgaard (2018), Barrett et al. (2018), and Klerke et al. (2016), use multi-task learning to solve the primary NLP task, where learning the gaze behaviour is an auxiliary task.

Orthographic/String similarity-based methods are often used as baseline methods for the cognate detection task, and the most commonly used method amongst them is the Edit distance-based similarity measure (Melamed, 1999; Mulloni and Pekar, 2006). Research in automatic cognate detection using various aspects involves the computation of similarity by decomposing phonetically transcribed words (Kondrak, 2000; Dellert, 2018), acoustic models (Mielke et al., 2012), clustering based on semantic equivalence (Hauer and Kondrak, 2011), and aligned segments of transcribed phonemes (List, 2012). Rama (2016) employs a Siamese convolutional neural network to learn the phonetic features jointly with language relatedness for cognate identification. Jäger et al. (2017) use SVM for phonetic alignment and perform cognate detection for various language families. Various works on orthographic cognate detection usually take alignment of substrings within classifiers like SVM (Ciobanu and Dinu, 2014, 2015) or HMM (Bhargava and Kondrak, 2009). Ciobanu and Dinu (2014) employ dynamic programming based methods for sequence alignment. Cognate facilitation in second language learners has previously been explored with the help of eye-tracking studies (Blumenfeld and Marian, 2005; Van Assche et al., 2009; Bosma and Nota, 2020) but the task of cognate detection has not been performed with gaze features obtained from the cognitive data in any of the previously available literature.

For the task of cognate detection, however, the use of cognitive features has not been established previously. The task of cognate detection is cross-lingual, and a reader's cognitive load should vary while trying to comprehend the meaning of concepts, from different languages. Our work tries to exploit the difference noted in terms of time taken and eye-movement patterns in cognates *vs* false friends, to generate additional features for the ACD task. Moreover, for Indian languages such as Marathi, where agglutination[3] varies the word length, the task becomes tougher, computationally.

3 Dataset Acquisition & Analysis

We pose the problem of cognate detection as a binary classification task in a supervised setting. We use a recently released challenging dataset (Kanojia et al., 2020) of cognates and false friends. We extract the Hindi-Marathi cognate and false friend pairs. The number of cognate and false friend pairs released by the paper cited above is 15726, and 5826. We select an equal number of cognates at random to reduce this skew, thus producing a balanced dataset for the classification task. For any further experiments in our paper, we use this artificially class-balanced dataset of 5826 (cognates) + 5826 (false friends) data points. We also augment the complete dataset with context information from the IndoWordnet (Bhattacharyya, 2017). The con-

---

[3] Agglutination is a linguistic process pertaining to derivational morphology in which complex words are formed by stringing together morphemes without changing them in spelling or phonetics.

	Cognates (1)	False Friends (0)
Kanojia et. al. (2020)	15726	5826
D1	**5826**	**5826**
D2	**100**	**100**

Table 2: Dataset Statistics for Cognate Detection Task

text information contains a *gloss* and an *example sentence* from the Wordnet data. The dataset released by Kanojia et al. (2020) contains the Synset IDs for each word pair, which helps us locate exact concept information from the dataset. We provide positive labels to cognates and negative labels to false friend pairs obtained from this data and construct what we call "D1".

We extract 100 pairs, at random, from each of the positive and negative labels for collecting gaze behaviour data, to construct what we call "D2". This data, extracted from D1, is used to collect gaze behaviour and annotation. Although we have gold labels for the data extracted, we ask the participants to annotate the data by asking them if the concepts shown on the screen mean the same. The annotation screen provides them with contextual clues obtained from Wordnet data on the screen, as shown in Figure 1. The complete dataset statistics are shown in Table 2.

3.1 Gaze Data Collection and Annotation

The task assigned to annotators was to read word pairs and the contextual clues provided on the screen, one pair at a time. The annotators were requested to label the pairs with a binary score indicating the similarity in meaning (*i.e.,* positive/negative). It should be noted that the participants were not instructed to annotate whether a pair is a cognate or a false friend, to rule out the Priming Effect, (*i.e.,* if the exact task (cognates *vs* false friends) is expected beforehand, processing cognate pairs will become relatively easier (Sáchez-Casas et al., 1992)). This ensures the ecological validity of our experiment in two ways: (1) The participant does not have any clue so that they can treat cognates with special attention (done by asking them to annotate based solely on meaning similarity) (2) Cognate pairs are mixed with false friend pairs and the participant does not have any prior knowledge about whether the next word pair would be a cognate or not. This also ensures that the participants pay attention to the task and do not just skim through the word-pair presented on the screen.

	μ_Pos	σ_Pos	μ_Neg	σ_Neg	p
P1	9.720	17.867	8.677	4.281	0.028
P2	8.596	10.526	7.619	13.794	0.049
P3	7.770	6.664	7.044	3.900	0.027
P4	9.686	17.729	8.664	4.306	0.031
P5	8.861	8.611	8.099	5.246	0.042
P6	7.854	6.286	7.184	3.442	0.033
P7	8.564	5.499	7.918	3.540	0.033
P8	8.018	5.955	7.340	3.742	0.031
P9	9.720	17.867	8.703	4.305	0.028

Table 3: T-test statistics for average fixation duration time per word for Positive labels (Cognates) and Negative labels (False Friends) for participants P1-P9.

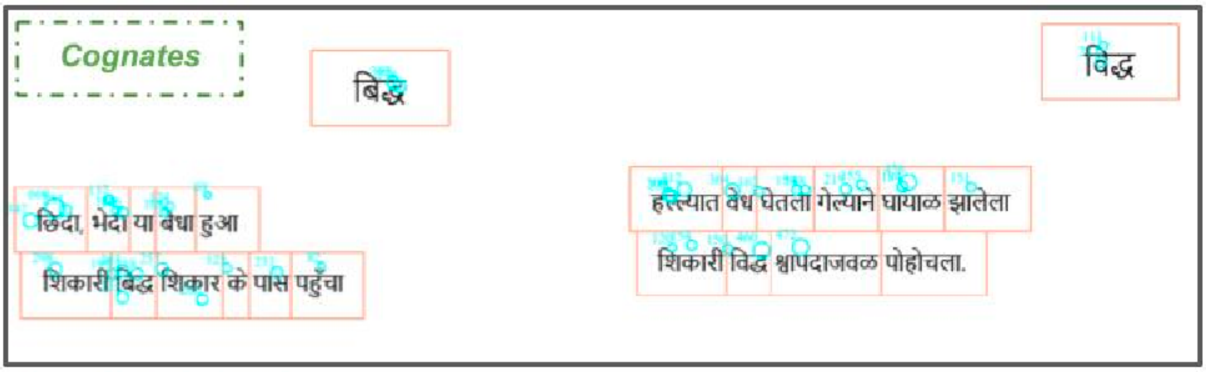
It should be noted that all our participants are primarily Marathi speakers and have learnt Hindi at the school level. Hindi and Marathi are considered to be relatively closer languages due to their shared vocabulary and the geographical location of the demographic. All the participants speak, understand and write - both Hindi and Marathi.

The collection of gaze data is conducted by following the standard norms in eye-movement research (Holmqvist et al., 2011). While reading, an SR-Research Eyelink-1000 eye-tracker (monocular remote mode, sampling rate 500Hz) records several eye-movement parameters like fixations (a long stay of gaze) and saccade (quick jumping of gaze between two positions of rest) and pupil size. For this experiment, the default value of 4ms was used for a gaze to be counted as fixation. We request a total of 15 participants to perform the annotation task, out of which only 11 participants could perform the data collection[4].

Out of the 11 completed annotations, we discarded the data from 2 participants as their gaze behaviour was erratic (the fixations were too far away from the IAs). The participants are graduates with science and engineering background. They are bilingual speakers who know both Hindi and Marathi. Our participants were given a set of instructions beforehand and were advised to seek clarifications before they proceed. The instructions mention the nature of the task as discussed above, annotation input method, and the necessity of head movement minimization during the experiment.

---

[4] We could not perform gaze data collection with the remaining 4 participants due to the COVID-19 pandemic.



छिदा, भेदा या बेधा हुआ
शिकारी बिद्ध शिकार के पास पहुँचा

हल्ल्यात वेध घेतला गेल्याने घायाळ झालेला
शिकारी विद्ध श्वापदाजवळ पोहोचला.

Figure 1: Screen capture showing collection of gaze features (via eye tracking) while displaying word pairs along with respective definitions and examples. The figure shows the cognate pair (*BiDha* - *BiDha*) where both mean "pierced" in the context of hunting. The figure also shown the glosses and example sentences provided to the annotator for this cognate pair in their respective languages (Hindi and Marathi)

3.2 Gaze Behaviour Data Analysis

The accuracy of similarity annotation by participants lies between 98% to 99.5% for individual annotators. Out of the 1800 annotations (9 annotators over 200 word-pairs), only 40 were predicted incorrectly. Annotation errors may be attributed to: (a) lack of patience/attention while reading, (b) issues related to word-pair comprehension, and (c) confusion/indecisiveness caused due to lack of contextual clues. In our analysis, we do not discard the data obtained from these incorrect annotations.

We observe distinct eye-movement patterns for cognate pairs in terms of fixation duration of the human readers. Our analysis shows that fixation duration normalized over word count is relatively larger for cognate pairs. For cognate pairs, we observe that average fixation duration amongst all participants is 1.3 times more than that of false friend pairs. To test the statistical significance, we conduct a two-tailed t-test (assuming unequal variance) to compare the average fixation duration per word for cognate and false friend pairs. The hypothesized mean difference is set to 0, and the error tolerance limit (α) is set to 0.05. The t-test analysis, presented in Table 3, shows that for all participants, a statistically significant difference exists between the average fixation duration per word for cognate pairs vs false friend pairs.

We believe this difference in average fixation duration is because the bilingual speakers who participated in the experiment can clearly distinguish between cognates and false friends and decide quickly in either case. The duration for which they fixate on either of the cases differs significantly for each participant. As per our observation, the participants take more time over cognate pairs to ensure similarity in meaning. Given their knowledge of both the languages and contextual clues, they were 'quickly' able to decide these word pairs did not mean the same. It should be noted that they were unaware of the 'cognate' or 'false friend' distinction concerning the experiment. They were simply asked to note the meaning of both the words given context, and annotate accordingly, as described in the paper.

3.3 Cross-lingual Word Embeddings

For this task, we use the cross-lingual word embedding models released by Kumar et al. (2020). The Hindi-Marathi cross-lingual models released with this paper are based on both MUSE (Conneau et al., 2017) and XLM[5] (Lample and Conneau, 2019). Additionally, we build the cross-lingual word embeddings model for Hindi-Marathi using VecMap (Artetxe et al., 2017). The model uses monolingual corpora released by Kunchukuttan et al. (2020) and a bilingual dictionary[6] required for the supervised method by Artetxe et al. (2017). These three cross-lingual models provide us with three feature sets for the task of cognate detection.

4 Feature Sets for Cognate Detection

In this section, we discuss the various features used for the task of cognate detection. It is to be noted that *false friends are spelt similarly across languages* but mean differently. Using false friends as data points with negative labels restricts us to the use of semantic similarity based features,

---

[5] Word representations extracted from the last layer of the contextual XLM model.

[6] Bilingual Lexicon

as orthographic or phonetic similarity-based measures would fail to detect sufficient distinction between them. Hence, we use the features proposed by Rama (2016) and Kanojia et al. (2019a) as baseline features for a comparative evaluation.

4.1 Phonetic Features

The IndicNLP Library provides phonetic features based vector for each character in various Indian language scripts. We utilize this library to compute a feature vector for each word by computing an average over character vectors. We compute vectors for both words in the candidate cognate pairs (PV_S and PV_T) and also compute contextual vectors (PCV_S and PCV_T) by averaging the vectors for all the contextual clues, generating a total of four vectors. We use these vectors as features for computing the baseline scores using the Siamese Convolutional Neural Network architecture proposed by Rama (2016).

4.2 Weighted Lexical Similarity (WLS)

The Normalized Edit Distance (NED) approach computes the edit distance (Nerbonne and Heeringa, 1997) for all word pairs in our dataset. A combination of NED with q-gram distance (Shannon, 1948) for a better similarity score. The q-grams ('n-grams') are simply substrings of length q. This distance measure has been applied previously for various spelling correction approaches (Owolabi and McGregor, 1988; Kohonen, 1978). Kanojia et al. (2019b) proposed Weighted Lexical Similarity (WLS) and we use it with the character-based Recurrent Neural Network architecture proposed by them to compute another set of baseline scores.

4.3 Cross-lingual Vectors & Similarity

As discussed above, we use the pre-trained cross-lingual embedding models for generating feature vectors for MUSE and XLM based approaches. These models are generated by aligning two disjoint monolingual vector spaces through linear transformations, using a small bilingual dictionary for supervision (Doval et al., 2018; Artetxe et al., 2017). Additionally, the cross-lingual embeddings model trained using Artetxe et al. (2017)'s approach provides us with the third set of feature vectors.

We use these models to obtain vectors for word-pairs (WV_S and WV_T) and averaged context vectors (CV_S and CV_T) from the contextual clues, to create three different feature sets. We obtain vectors for each candidate pair and their context using all the three cross-lingual methodologies. The use of cross-lingual models has been proposed for differentiating between cognates and false friends by Merlo and Andueza Rodriguez (2019), but evaluation with the cognate detection task had not been performed. We perform this evaluation on our datasets (D1, D2 and D1+D2) using various classification methods and discuss them later.

4.4 Cognitive Features from Gaze Data

Gaze behaviour of participants, characterized by fixations, forward saccades, skips and regressions, can be used as features for NLP tasks (Mishra et al., 2018b). Since these gaze features relate to the cognitive process in reading (Altmann, 1994), we consider these as features in our model. The current gaze data extraction adds up all the interest areas on the screen (word + context). The software used to analyze the gaze data, currently, provides us with collated results for all the gaze-based features.

From the gaze behaviour data collected, we extract a total of 18 features for each of the 1800 data points. Using supervised feature selection approach, we are able to select eight best features via grid search using Logistic Regression. We use the SelectKBest implementation along with hyperparameter tuning via GridSearchCV, present in the sklearn (Pedregosa et al., 2011) library. Here onwards, we refer to these eight features when discussing cognitive or gaze-based features in this paper. These eight best features, along with their description, are listed in Table 4.

4.5 Gaze Feature Prediction

Collection of gaze data for a large number of samples can be a costly task. We propose a neural model for cognitive features prediction. Our neural model is a feed-forward neural network to perform a regression task and predict gaze features. We collect gaze data for only 200 word-pairs (D2) with the help of 9 annotators which provides us with a total of 1800 data points for training and validation. As reported in Table 5, the initial results on D1 using different cross-lingual embeddings show that XLM based contextual features perform the best amongst all the cross-lingual models.

As an input to the network, we provide the feature vectors from the XLM model. This network's output is the predicted gaze features for the D1 dataset, using the gaze features as gold predictions

Gaze Feature	Description
Average Fixation Duration	The average of all fixation duration across all interest areas present on the screen.
Average Saccade Amplitude	Saccade amplitude is the amplitude of going back and forth measured in terms of duration.
Fixation Count	Counting the number of times user's eyes are fixated on the screen.
Fixation Duration Max	Maximum time for a single fixation on any Interest Area.
Fixation Duration Min	Minimum time for a single fixation on any Interest Area.
IA Count	Interest Area Count (no. of IAs on the screen)
Run Count	Consecutive counts for same Interest Area are ignored in Run Count
Saccade Count	Total counts of Saccades

Table 4: Gaze Features used for the task of Cognate Detection

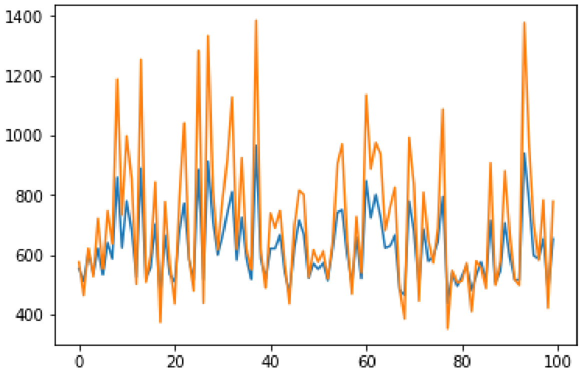


Figure 2: Predicted feature values (blue) vs. Gold feature values (orange) for the average fixation duration feature, on 100 samples.

from the D2 dataset. This network contains three linear hidden layers with 128, 64 and 32 dimensions. After each layer, we use the sigmoid activation function and dropout after each sigmoid with a dropout value of 0.2. We use 0.1 as the learning rate and use the Mean Squared Error (MSE) loss function. A graph comparing the values for the predictions vs the actual values for the average saccade amplitude for 100 samples, can be seen in Figure 2.

5 The Cognate Detection Task

We employ both classical machine learning-based models and a simple feed-forward neural network. To compare our work with the previously proposed approaches, we replicate the best-reported systems from Rama (2016) *i.e.,* Siamese Convolutional Neural Network with phonetic vectors as features and also replicate Kanojia et al. (2019b)'s approach which uses a Recurrent Neural Network architecture with a weighted lexical similarity (WLS) as the feature set. The input to our classifiers is the feature sets described above for each candidate pair.

Among the classical machine learning models, we use Support Vector Machines (SVM) and Logistic Regression (LR). We experiment with the use of both linear SVMs and kernel SVMs (Gaussian and Polynomial). We perform a grid-search to find the best hyper-parameter value for C over the range of 0.01 to 1000. We deploy the Feed Forward Neural Network (FFNN) with one hidden layer. We perform cross-validation with different activation function settings (tanh, hardtanh, sigmoid and relu) and the hidden layer dimension in the network (30, 50, 100, and 150). We use binary cross-entropy as the optimization algorithm. Finally, we choose the hyper-parameter configuration with the best validation accuracy. We train the model with the selected configuration with an initial learning rate of 0.4, and we halve the learning rate when the error on the validation split increases. We stop the training once the learning rate falls below 0.001. *We perform 5-fold stratified cross-validation, which divides the data into train and test folds, randomly.*

Initially, we perform our experiments with the feature sets from three different cross-lingual embeddings (MUSE, XLM, and VecMap) for the dataset D1, then with the smaller dataset D2 and later on the combined dataset D1+D2. We, then, perform the same task for the smaller dataset D2 by combining cognitive features with individual cross-lingual feature sets. We also observe the performance of standalone gaze features for the D2 dataset. Finally, we evaluate the predicted gaze features on the combined dataset by combining them with cross-lingual features and a standalone feature set, using the feed-forward neural network. We report the results of the cognate detection task in the next section and discuss them in detail.

6 Results and Dicussion

We report the results of the cognate detection task in Table 5. We use the original implementations of Rama (2016) and Kanojia et al. (2019b) on the combined (D1+D2) dataset. In our initial evaluation on the D1 dataset, cross-lingual model-based features (XLM, MUSE, and VecMap) can be seen to outperform the baseline systems which use phonetic and orthographic features. Using the XLM-based

	P	R	F	P	R	F	P	R	F	P	R	F
Feature Set →	Phonetic			WLS								
Rama et. al., 2016 (D1+D2)	0.71	0.69	0.70	-	-	-						
Kanojia et. al., 2019 (D1+D2)	-	-	-	0.76	0.72	0.74						
Feature Set →	XLM			MUSE			VecMap					
Linear SVM (D1+D2)	0.83	0.71	0.77	0.72	0.68	0.70	0.70	0.65	0.67			
LogisticRegression (D1+D2)	0.85	0.74	0.79	0.80	0.71	0.75	0.70	0.66	0.68			
FFNN (D1 + D2)	0.82	0.84	**0.83**	0.83	0.79	0.81	0.75	0.76	0.75			
Feature Set →	XLM+Gaze			MUSE+Gaze			VecMap+Gaze			Gaze		
Linear SVM (D2)	0.81	0.69	0.75	0.72	0.73	0.72	0.70	0.75	0.72	0.77	0.76	0.76
LogisticRegression (D2)	0.84	0.75	0.79	0.76	0.72	0.74	0.81	0.71	0.76	0.80	0.75	0.77
FFNN (D2)	0.83	0.85	**0.84**	0.83	0.78	0.80	0.86	0.83	0.84	0.81	0.71	0.76
Predicted Gaze Features On D1 (11652 samples) and Collected Gaze Features on D2 (200 samples)												
Feature Set →	XLM+Gaze			MUSE+Gaze			VecMap+Gaze			Gaze		
FFNN (D1 + D2)	0.84	0.88	**0.86**	0.85	0.78	0.81	0.83	0.85	0.84	0.77	0.76	0.76
FFNN (D1) [Only Predicted Gaze]	0.83	0.84	0.83	0.82	0.79	0.80	0.80	0.86	0.83	0.76	0.77	0.76

Table 5: Classification results in terms of weighted Precision (P), Recall (R), and F-scores (F) using 5-fold cross-validation using different feature sets as described above.

features, we observe an improvement of 9% over the stronger baseline (Kanojia et al., 2019b) and 13% over the system by Rama (2016). It can be seen that MUSE and VecMap based features also perform better on the combined dataset. In terms of both precision and recall, cross-lingual features are shown to outperform the baseline systems. The cross-lingual approach, with representations from VecMap-based model, fails to perform as well as MUSE and XLM-based models. The contextual XLM model achieves the best scores in almost all the settings. We believe its performance can be attributed to the linguistic closeness of the language pair and context from the contextual clues provided. For example, the false friend pair "*kaccHa*" (meaning inexperienced) - "*kaccHa*" [raw (food)] is classified correctly by XLM but incorrectly by both the baseline models, and VecMap-based classification. This signifies that fine-grained semantic difference between such false friend pairs can be captured via cognitive features. We report the additional results on individual datasets (D1 and D2), for all the baseline and cross-lingual approaches, in Table 6.

For all the classifiers, the gaze features are averaged across participants and augmented with cross-lingual features. The gaze fixation duration collects the total time spent, as fixations, on each interest area including the context clues. We were hopeful that the participants would focus only on important contextual clues and not the stop words with our experiment design. However, the sample points are not enough to concretely discuss this aspect of our study. These results are reported for all the classifiers with D2 dataset. Our feature combinations outperform the baselines with an F-score improvement of 10% points over the stronger baseline (WLS). We also report the precision, recall and F-score values when only gaze features are used to predict the labels for our candidate pairs. We observe that standalone gaze features are not as effective as when combined with cross-lingual feature vectors. When gaze features are predicted using the methodology described in Section 4.5, the model performance for FFNN on D1 remains the same with XLM+Gaze, decreases slightly for MUSE+Gaze and significantly improves for VecMap+Gaze. We observe that predicted gaze features do not significantly drop the performance, and hence, add the collected gaze data samples (D2) to D1.

On the combined dataset with collected gaze features (on D2) and predicted gaze features (on D1), we report our best system [FFNN (D1+D2)] which shown an improvement of 12% over the stronger baseline (WLS), and 16% over the weaker baseline (Phonetic). This system also outperforms the best reported cross-lingual features-based approach by 3%, as shown in Table 5.

For example, the cognate pair "*utPaNa*" (Hindi) - "*utpAaDit*" (Marathi) (both meaning manufactured) is classified correctly by this system, but incorrectly

	Phonetic			WLS					
	P	R	F	P	R	F	P	R	F
Rama et. al., 2016 (D1)	0.70	0.68	0.69	-	-	-			
Kanojia et. al., 2019 (D1)	-	-	-	0.74	0.70	0.72			
Rama et. al., 2016 (D2)	0.64	0.57	0.60	-	-	-			
Kanojia et. al., 2019 (D2)	-	-	-	0.61	0.66	0.63			
	XLM			MUSE			VecMap		
Linear SVM (D1)	0.81	0.71	0.76	0.70	0.68	0.69	0.70	0.65	0.67
LogisticRegression (D1)	0.80	0.75	0.77	0.72	0.74	0.73	0.70	0.73	0.71
FFNN (D1)	0.80	0.84	0.82	0.81	0.76	0.78	0.77	0.76	0.76
Linear SVM (D2)	0.72	0.65	0.68	0.65	0.60	0.62	0.62	0.57	0.59
LogisticRegression (D2)	0.78	0.69	0.73	0.67	0.67	0.67	0.63	0.61	0.62
FFNN (D2)	0.79	0.81	0.80	0.76	0.71	0.73	0.74	0.71	0.72

Table 6: Additional results in terms of weighted Precision (P), Recall (R), and F-scores (F) using 5-fold cross-validation using different feature sets as described above. These are additional results on the individual datasets D1 and D2 for which the combined results (D1 and D2) are already shown in Table 5 for a fair comparison.

by both the baselines, and all the cross-lingual systems. Furthermore, we also show that gaze features can be predicted based on a small sample data, and improved performance can be attained with the help of cross-lingual features, reported with our work.

7 Conclusion and Future Work

In this paper, we harness cross-lingual embeddings and gaze-based features to improve the task of cognate detection for the Indian language pair of Hindi-Marathi. We create a novel framework that derives insights from human cognition, that manifests over eye movement patterns. We hypothesize that by augmenting cross-lingual features with features obtained from the gaze data, the task of cognate detection can be improved. We use a linked knowledge graph (IndoWordnet) to augment a publicly released cognate dataset with contextual clues. We collect the gaze behaviour data from nine participants over 200 samples and perform the task of cognate detection for both our datasets (with gaze data and without gaze data). Then, we use a neural network to predict gaze features for unseen samples and perform the task of cognate detection to show improved performance, despite a small sample of collected gaze data.

We reproduce the previously proposed baseline approaches and perform experiments using additional features obtained via cross-lingual models for a comparative evaluation. The previously proposed approaches (Rama, 2016; Kanojia et al., 2019b) for this task are shown to be outperformed by cross-lingual features and the combination of these features with the obtained gaze data. Our experiments use three different approaches to generate feature representations for the cognate detection task, and all of them show improvements over previously proposed approaches. We observe consistent improvements in terms of precision, recall and F-scores. Over the stronger baseline, our best system shows an improvement of 12% points and 16% points over the weaker baseline. This system also outperforms the cross-lingual features based approaches by 3%, over the combined dataset. We release this augmented dataset, along with our code and cross-lingual models for further research.

In future, we aim to add more language pairs and leverage contextual information from knowledge graphs using sequence-based neural models. We also aim to collect gaze data and then model the gaze predictions in a multi-task setting. We plan to investigate other multilingual contextual embeddings' performance for this task (*e.g.,* M-BERT, IndicBERT, MuRIL). We also plan to look for a method to differentiate between different interest areas and see if a markup facility is present in the software used to analyze the gaze data. We also aim to investigate the task of cognate detection for the Indo-European language family, in the near future.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A Smith, and David Yarowsky. 1999. Statistical machine translation. In *Final Report, JHU Summer Workshop*, volume 30.

Gerry TM Altmann. 1994. Regression-contingent analyses of eye movements during sentence processing: Reply to rayner and sereno. *Memory & Cognition*, 22(3):286–290.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. 2018. Sequence classification with human attention. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, Brussels, Belgium. Association for Computational Linguistics.

Maria Barrett, Joachim Bingel, Frank Keller, and Anders Søgaard. 2016. Weakly supervised part-of-speech tagging using eye-tracking data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–584, Berlin, Germany. Association for Computational Linguistics.

Aditya Bhargava and Grzegorz Kondrak. 2009. Multiple word alignment with profile hidden markov models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 43–48. Association for Computational Linguistics.

Pushpak Bhattacharyya. 2017. Indowordnet. In *The WordNet in Indian Languages*, pages 1–18. Springer.

Henrike K Blumenfeld and Viorica Marian. 2005. Covert bilingual language activation through cognate word processing: An eye-tracking study. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 27.

Evelyn Bosma and Naomi Nota. 2020. Cognate facilitation in frisian–dutch bilingual children's sentence reading: An eye-tracking study. *Journal of experimental child psychology*, 189:104699.

Alina Maria Ciobanu and Liviu P Dinu. 2014. Automatic detection of cognates using orthographic alignment. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 99–105.

Alina Maria Ciobanu and Liviu P Dinu. 2015. Automatic discrimination between cognates and borrowings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 431–437.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Johannes Dellert. 2018. Combining information-weighted sequence alignment and sound correspondence models for improved cognate detection. In *Proceedings of the 27th international conference on computational linguistics*, pages 3123–3133.

Yerai Doval, Jose Camacho-Collados, Luis Espinosa-Anke, and Steven Schockaert. 2018. Improving cross-lingual word embeddings by meeting in the middle. *arXiv preprint arXiv:1808.08780*.

Ana V González-Garduño and Anders Søgaard. 2018. Learning to predict readability using eye-movement data from natives and learners. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Bradley Hauer and Grzegorz Kondrak. 2011. Clustering semantically equivalent words into cognate sets in multilingual lists. In *Proceedings of 5th international joint conference on natural language processing*, pages 865–873.

Nora Hollenstein and Ce Zhang. 2019. Entity recognition at first sight: Improving NER with eye movement information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1–10, Minneapolis, Minnesota. Association for Computational Linguistics.

Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford.

Gerhard Jäger, Johann-Mattis List, and Pavel Sofroniev. 2017. Using support vector machines and state-of-the-art algorithms for phonetic alignment to identify cognates in multi-lingual wordlists. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1205–1216.

Marcel A Just and Patricia A Carpenter. 1980. A theory of reading: From eye fixations to comprehension. *Psychological review*, 87(4):329.

Diptesh Kanojia, Malhar Kulkarni, Pushpak Bhattacharyya, and Gholamreza Haffari. 2020. Challenge dataset of cognates and false friend pairs from indian languages. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3096–3102.

Diptesh Kanojia, Malhar Kulkarni, Pushpak Bhattacharyya, and Gholemreza Haffari. 2019a. Cognate identification to improve phylogenetic trees for indian languages. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 297–300. ACM.

Diptesh Kanojia, Kevin Patel, Pushpak Bhattacharyya, Malhar Kulkarni, and Gholemreza Haffari. 2019b. Utilizing wordnets for cognate detection among indian languages. In *Global Wordnet Conference (2019)*.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1528–1533, San Diego, California. Association for Computational Linguistics.

Teuvo Kohonen. 1978. A very fast associative method for the recognition and correction of misspelt words, based on redundant hash addressing. In *Proceedings of the fourth International Joint Conference on Pattern Recognition, 1978*.

Grzegorz Kondrak. 2000. A new algorithm for the alignment of phonetic sequences. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 288–295. Association for Computational Linguistics.

Grzegorz Kondrak. 2001. Identifying cognates by phonetic and semantic similarity. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Saurav Kumar, Saunack Kumar, Diptesh Kanojia, and Pushpak Bhattacharyya. 2020. "a passage to India": Pre-trained word embeddings for Indian languages. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 352–357, Marseille, France. European Language Resources association.

Anoop Kunchukuttan, Divyanshu Kakwani, Satish Golla, Avik Bhattacharyya, Mitesh M Khapra, Pratyush Kumar, et al. 2020. Ai4bharat-indicnlp corpus: Monolingual corpora and word embeddings for indic languages. *arXiv preprint arXiv:2005.00085*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

Johann-Mattis List. 2012. Lexstat: Automatic detection of cognates in multilingual wordlists. In *Proceedings of the EACL 2012 Joint Workshop of LINGVIS & UNCLH*, pages 117–125. Association for Computational Linguistics.

Yunfei Long, Rong Xiang, Qin Lu, Chu-Ren Huang, and Minglei Li. 2019. Improving attention model based on cognition grounded data for sentiment analysis. *IEEE Transactions on Affective Computing*.

Sandeep Mathias, Rudra Murthy, Diptesh Kanojia, Abhijit Mishra, and Pushpak Bhattacharyya. 2020. Happy are those who grade without seeing: A multi-task learning approach to grade essays using gaze behaviour. *arXiv preprint arXiv:2005.12078*.

I Dan Melamed. 1999. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25(1):107–130.

Helen M Meng, Wai-Kit Lo, Berlin Chen, and Karen Tang. 2001. Generating phonetic cognates to handle named entities in english-chinese cross-language spoken document retrieval. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU'01.*, pages 311–314. IEEE.

Paola Merlo and Maria Andueza Rodriguez. 2019. Cross-lingual word embeddings and the structure of the human bilingual lexicon. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 110–120, Hong Kong, China. Association for Computational Linguistics.

Michelle M Mielke, Rosebud O Roberts, Rodolfo Savica, Ruth Cha, Dina I Drubach, Teresa Christianson, Vernon S Pankratz, Yonas E Geda, Mary M Machulda, Robert J Ivnik, et al. 2012. Assessing the temporal relationship between cognition and gait: slow gait predicts cognitive decline in the mayo clinic study of aging. *Journals of Gerontology Series A: Biomedical Sciences and Medical Sciences*, 68(8):929–937.

Abhijit Mishra and Pushpak Bhattacharyya. 2018. *Cognitively Inspired Natural Language Processing: An Investigation Based on Eye-tracking*. Springer.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2016. Harnessing cognitive features for sarcasm detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104, Berlin, Germany. Association for Computational Linguistics.

Abhijit Mishra, Srikanth Tamilselvam, Riddhiman Dasgupta, Seema Nagar, and Kuntal Dey. 2018a. Cognition-cognizant sentiment analysis with multi-task subjectivity summarization based on annotators' gaze behavior. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. 2018b. Author profiling for abuse detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1088–1098, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Andrea Mulloni and Viktor Pekar. 2006. Automatic detection of orthographics cues for cognate recognition. In *LREC*, pages 2387–2390.

John Nerbonne and Wilbert Heeringa. 1997. Measuring dialect distance phonetically. In *Computational Phonology: Third Meeting of the ACL Special Interest Group in Computational Phonology*.

Olumide Owolabi and DR McGregor. 1988. Fast approximate string matching. *Software: Practice and Experience*, 18(4):387–393.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Taraka Rama. 2016. Siamese convolutional networks for cognate identification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1018–1027.

Taraka Rama, Johann-Mattis List, Johannes Wahle, and Gerhard Jäger. 2018. Are automatic methods for cognate detection good enough for phylogenetic reconstruction in historical linguistics? *arXiv preprint arXiv:1804.05416*.

Morteza Rohanian. 2017. Multi-document summarization of Persian text using paragraph vectors. In *Proceedings of the Student Research Workshop Associated with RANLP 2017*, pages 35–40, Varna. INCOMA Ltd.

Rosa M Sáchez-Casas, José E García-Albea, and Christopher W Davis. 1992. Bilingual lexical processing: Exploring the cognate/non-cognate distinction. *European Journal of Cognitive Psychology*, 4(4):293–310.

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Abhinav Deep Singh, Poojan Mehta, Samar Husain, and Rajkumar Rajakrishnan. 2016. Quantifying sentence complexity based on eye-tracking measures. In *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*, pages 202–212, Osaka, Japan. The COLING 2016 Organizing Committee.

Eva Van Assche, Wouter Duyck, Robert J Hartsuiker, and Kevin Diependaele. 2009. Does bilingualism change native-language reading? cognate effects in a sentence context. *Psychological science*, 20(8):923–927.

Victoria Yaneva, Le An Ha, Richard Evans, and Ruslan Mitkov. 2020. Classifying referential and non-referential it using gaze. *arXiv preprint arXiv:2006.13327*.

A Simple Three-Step Approach for the Automatic Detection of Exaggerated Statements in Health Science News

Jasabanta Patro
ETRO, VUB
`jasabanta.patro@vub.be`

Sabyasachee Baruah
USC, California
`sbaruah@usc.edu`

Abstract

There is a huge difference between a scientific journal reporting 'wine consumption might be correlated to cancer', and a media outlet publishing 'wine causes cancer' citing the journal's results. The above example is a typical case of a scientific statement being exaggerated as an outcome of the rising problem of *media manipulation*. Given a pair of statements (say one from the source journal article and the other from the news article covering the results published in the journal), is it possible to ascertain with some confidence whether one is an exaggerated version of the other? This paper presents a surprisingly simple yet rational three-step approach that performs best for this task. We solve the task by breaking it into three sub-tasks as follows – (a) given a statement from a scientific paper or press release, we first extract relation phrases (e.g., 'causes' versus 'might be correlated to') connecting the dependent (e.g., 'cancer') and the independent ('wine') variable, (b) classify the strength of the relationship phrase extracted and (c) compare the strengths of the relation phrases extracted from the statements to identify whether one statement contains an exaggerated version of the other, and to what extent. Through rigorous experiments, we demonstrate that our simple approach by far outperforms baseline models that compare state-of-the-art embedding of the statement pairs through a binary classifier or recast the problem as a textual entailment task, which appears to be a very natural choice in this settings.

1 Introduction

Exaggerations in health news can have tremendous adverse effects on the lifestyle of the common masses who feed themselves mostly on such news instead of the source scientific publication. This problem is challenging as it involves many intrinsic complexities that need to be addressed. First, while encountering a scientific claim, we need to identify the true fact related to the claim from the knowledge base (in most cases the source journal article). For instance, if the press release/ news report states that 'chocolate causes acne' then this claim needs to be compared to the scientific study that actually recruits human subjects and does experiments to study connections between chocolate consumption and acne vulgaris. In fact, the study reported in Fulton-Jr. et al. (1969) proves that chocolate consumption is 'not related' to acne.

In this paper, we propose a very simple three-step method[1] that given a pair of statements, e.g., 'chocolate causes acne' taken from the press release and 'chocolate consumption is not related to acne' taken from the source journal, can identify if the former is an exaggerated version of the latter. Note that the problem that we aim to solve requires the pair of statements to be compared as inputs to produce the desired output (exaggerated or not). However, toward the end of the paper, we outline a simple heuristic that dissolves this constraint for the considered dataset that the pair of statements that we compare from the whole article (press release or source journal) needs to be known to us a priori.

We note that comparing state-of-the-art embedding of the two statements using binary classifiers does not give much advantage; even adapting the problem as a textual entailment task (a natural adaptation since one statement is an exaggerated version of the other) surprisingly does not bring much additional benefits.

This paper therefore puts forward *a simple three step approach, breaking the task down to three rational steps*: (1) *given a statement from a scientific paper or press release, extract relation phrases* (*e.g., 'causes' versus 'correlated to'*) *connecting*

---

[1] Code: `bit.ly/39crP39`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3293–3305
April 19 - 23, 2021.

the dependent (*e.g., 'acne'*) *and the independent* (*e.g., 'chocolate'*) *variable,* (2) *classify the strength of the relation phrase and* (3) *compare the strengths of the relation phrases extracted from the statements to identify whether one sentence contains an exaggerated version of the other, and to what extent.*

Our approach is operationalised on the data released by Sumner et al. (2014). The exact statements containing the relation phrases within an article or the source journal are annotated in the dataset allowing us to effectively train our models. The main results of our paper are,

- **Extraction of relation phrases**: We experiment with number of syntax driven and sequence labeling approaches to extract the relation phrases. A adaptive version of recently proposed and highly successful BERT-NER (Devlin et al., 2019) performs best with a F1-score of ~ 0.85 for this task.
- **Strength classification**: Next, given a statement with its relation phrase already labeled, we pass it through standard classifiers to learn the *strength* of phrase[2]. We achieve a micro-F1 and macro-F1 of 0.74 and 0.69 respectively for 6-class classification task.
- **Exaggeration detection**: Now given a pair of statements with their strength levels marked and one taken from the source journal while the other from the news article (or press release) we compute the difference in strengths to output whether the latter is an exaggerated version of the former. We obtain a perfect match of exaggeration levels for 0.62 fraction of cases.
- **Additional contribution**: We also identify a mechanism to spot the exact location of the main claim statement in the whole document so that the exact pair is not needed as inputs for our pipeline to work. Our mechanism seem to work for the considered dataset.

2 Related work

Media manipulation is a set of related techniques in which the manipulator attempts to create an image or argument that favors particular interests (Coxall, 2013). Media manipulation and fake news got huge attention from the research community in the current decade. Hundreds of studies got published in this domain which makes it impossible to cite them all. The surveys of different methods and datasets published in these domain can be found in Parikh and Atrey (2018); Zhou and Zafarani (2018); Sharma et al. (2019); Bondielli and Marcelloni (2019); Oshikawa et al. (2018); Haciyakupoglu et al. (2018); Zhou and Zafarani (2020); Van Eemeren et al. (2009). Previous study (Sumner et al., 2014) suggests that press releases are a major source of exaggeration. Identification of relationship between entities and associated strength in scientific articles and comparing them with that of news reports and press releases is the key to exaggeration detection. In literature, there are many techniques present separately to identify entities along with their relationships (Blake, 2010) and strength associated in a relationship (Light et al., 2004; Vlachos and Craven, 2010). The authors in Lim et al. (2016) propose a framework to identify claims from tweet corpus related to major events. In (Giasemidis et al., 2016) the authors build autonomous message classifier that filters relevant and trustworthy information from Twitter. In (Khoo et al., 2000) the authors develop a knowledge extraction and knowledge discovery system that extracts causal knowledge from textual databases.

Present study: None of the studies mentioned above tackle the exaggeration detection problem. In a recent study Li et al. (2017) analyze the same dataset as we use in this paper and attempt to identify exaggeration. They also assume that the pair of statements are available as inputs for the detection task. However, the biggest drawback of this work is that the authors additionally assume that the relationship phrase in the input statement is also known which makes the task significantly simpler. In fact, the authors also do not use these relationship phrases as important signals but treat each statement as a whole as bag-of-words. We, on the other hand, identify the relation phrases automatically in the first place and use it as an important signal for the next two sub-tasks (strength classification and exaggeration detection) which is the most important contribution of our work. We also compare this work with our method and show that we considerably outperform them.

---

[2] The strength levels are defined as in (Sumner et al., 2014) and certain coarse-grained revisions of the same.

3 Dataset and Preprocessing

3.1 Dataset

We use the publicly available dataset[3] released by Sumner et al. (2014) for our experiments. This dataset contains detailed annotations of 462 journals, corresponding 462 press releases and 668 news articles, issued in 2011 by the Russel Group of Universities (20 leading UK Research Universities) in health related topics. Every individual press release is a follow-up of a journal paper; we assume this journal paper to be the reference for our analysis as it is followed by Sumner et al. (2014). Every press release in turn, is discussed by some news reports. In the dataset, 230 out of 462 journals and press releases have at least one news article coverage.

For each of the three sources, the dataset has detailed annotations of different types. First for each journal/ news article/ press release, the statement containing the main claim is manually identified by the annotators within the running text. Now in this statement, the stretch of relationship phrase is marked in bold. Third, the strength of this relationship phrase is graduated on a Likert-like scale from '0' to '6'. (see section A in supplementary material).

Note that, as per the annotation guidelines, one article can have only one statement with the main claim. The relationship phrase connects the independent variable (*IV*) with the dependent variable (*DV*) in the statement with the main claim. Note that sometimes either the *IV* or the *DV* or both might not be part of the statement with the main claim (i.e., may be present in other parts of the text in the article and connected to statement with the main claim only implicitly). However, since for our work we need the statement with the main claim and the relationship phrases only the above limitation does not pose a hindrance.

While Sumner et al. (2014) provides the dataset already marked by one of the seven quantization levels for each journal, press release and news report, thus allowing for analysis of the exaggerated content, it is difficult to ascertain the robustness of the results obtained. This is primarily because some of the quantization levels seem to be too close (see the 'Description' of strength categories in supplementary material.) and the data set is unbalanced (see Table **??**). In order to test the robustness of the results that we present in the subsequent sections, we also club the above quantizations into more coarse-grained labels. Essentially, we consider a 4-class and a 2-class quantization in addition to the 6-class (omitting class '0'). For the 4-class we map the above seven quantizations as follows: $1 \rightarrow 1$, $(2,3) \rightarrow 2$, $(4,5) \rightarrow 3$ and $6 \rightarrow 4$. For the two class we map as follows: $(1,2,3,4,5) \rightarrow 1$ and $6 \rightarrow 2$.

Strength	#Statements	Fraction
1	69	0.04
2	321	0.2
3	132	0.08
4	159	0.1
5	108	0.07
6	812	0.51

Table 1: Distribution of statements across the different strength categories.

3.2 Data preprocessing

We discard statements with the main claim in which the relationship phrases are not marked by the annotators; these include the statements from category '0'[4] and a very few statements from the other categories. At the end of this process, we have a set of 1601 statements with annotated relation phrases. The distribution of statements across the different strength categories is shown in Table **??**. The table shows that the categories '6' and '2' are the dominant ones. We divide the 1601 statements (together from journals, press releases and news reports) into training, development and test sets. We keep 1000 statements in the training set, 300 statements in the development set and 301 statements in the test set. Note that from the 301 statements in the test set, we can construct a total of 316 distinct statement pairs each formed from the comparison of a source journal and a corresponding news paper article/press release. While for the first two sub-tasks (i.e., relation phrase labeling and strength classification) we need the individual statements as input, in the last sub-task (i.e., exaggeration identification) we need pairs of statements for final comparison and prediction.

4 Methodology

As discussed earlier, our approach has three steps. They are (i) relation phrase labeling, (ii) strength

---

[3] Dataset: `https://bit.ly/2qc86tk`

[4] Note that category '0' refers to cases where there is no relation between the IV and DV and is therefore not useful for further processing.

classification, and (iii) exaggeration identification. All of the three modules are connected in a sequence as shown in Figure 1. The relation phrase labeling module takes a selected statement and labels the relation phrase present in it. Next, the strength classification module takes a relation phrase as input and predicts its strength level. Finally, in the exaggeration identification module, the strengths of source statement and target statement are compared, and it calculates the exaggeration or underplay level of the target statement with respect to the source statement. The individual modules are described in details in the subsequent subsections.

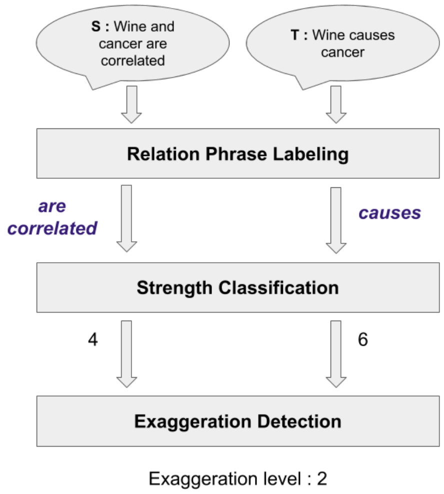


Figure 1: Flow diagram of the methodology. S: source statement, T: target statement.

4.1 Relation phrase labeling

As previously stated, the primary objective of this module is to identify the relation phrase describing the relation between the independent and the dependent variables. This problem is more difficult than the traditional entity-relation extraction problem as multiple entity pairs connected by relation phrases can be present in a statement. Therefore, finding relation phrase that denotes the main claim is a challenging task. We broadly employ two different types of approaches for relation labeling – (i) syntax driven approaches, and (ii) sequence labeling approaches. We describe each of them in next subsections.

4.1.1 Syntax driven approaches

The key idea: In syntax driven approaches, we rely on the fact that the position of the relation phrases in a statement is syntax driven. In particular, we mine the structural patterns in the dependency tree of the statements. In addition, we use heuristics over the state-of-the-art entity relation extraction tool (Angeli et al., 2015) to get relation phrases in the causal statements.

Dependency tree heuristics (DTH): The intuition behind this method is that relation phrases have characteristic syntactic (part-of-speech, lemmas, dependency edges etc.) and semantic features. We intend to leverage these in the dependency tree representation of statements, by collapsing unimportant relations and selecting the rightful node through various heuristics. The phrase corresponding to the selected node is identified as the relation phrase. The details of the steps we follow for the construction of the collapsed dependency tree and the extraction of the relation phrase from the causal statement are presented in supplementary material.

OpenIE heuristics: We use Stanford open information extraction (OpenIE) tool for the identification of entity-relation triplets. Let $t_1, t_2, ...t_n$ be the n triplets obtained for a statement s, where each t_i consists of two entities at its end connected by a relation phrase. We choose t_i having the largest and smallest phrase and consider these as the representative relation phrase for the statement s. The rationale behind choosing the largest phrase is to increase the probability of including the original relation phrase. The smallest phrase is chosen to show that smaller sized phrases always perform worse than the largest phrase.

4.1.2 Sequence labeling approaches

The key idea: The central idea these approaches put forward is use the training data to create a model to label the relation phrase in the input statement. The relation phrases are marked as per traditional BIO[5] encoding format. In addition, the BERT scheme that we shall use has three other labels – X for added morphological inflation, CLS for sentence beginning and SEP for sentence separations. Based on this annotated training data the sequence labeler is tasked to learn the beginning, the stretch and the end of the relationship phrase. Once this is marked for each statement the labeled phrases are passed on to the next phase of the pipeline for strength classification.

LSTM-CRF:We use the LSTM-CRF architecture

---

[5] https://en.wikipedia.org/wiki/Inside%E2%80%93outside%E2%80%93beginning_(tagging)

Features	Dimension
POS tag	40
POS bigram	40
Wordnet cluster	40

Table 2: Additional features at CRF layer.

similar to Lample et al. (2016) in addition to a set of novel features as described in Table ?? in the CRF layer[6] along with the hidden state vector of the BiLSTM layer. The combination is done by concatenating the feature vectors with the hidden state vector. The input layers to the model are vector representations of the individual words or characters.

LSTM-CNN-CRF: We use the architecture proposed by Ma and Hovy (2016) and adapt for our purpose[7].

LM-LSTM-CRF: Here we use the task-aware neural sequence labeling model proposed by (Liu et al., 2017) and adapt it for our purpose.

BERT-SL: We use the variant BERT (Devlin et al., 2018) technology originally proposed for solving the NER task[8], which is essentially modeled as a sequence labeling problem. We suitably adapt the BERT-NER framework to extract the relation phrases. BERT's model architecture is a multilayer bidirectional transformer encoder based on the original implementation described in (Vaswani et al., 2017)[9]. In particular, we use the BERT base architecture that has 12 transformer layers, hidden vector size of 768 and 12 self-attention heads one corresponding to each transformer module. For fine-tuning, the final hidden representation of each token is fed to a classification layer over the NER label set. For our purpose we train the BERT-NER model using our training data and the label set corresponds to the relationship phrase beginning and end markers instead of the NER labels.

4.1.3 Evaluation

We use the standard token level F1-score and accuracy to compare the different labeling approaches.

4.2 Strength classification

The primary objective of this module is to predict the strength level of the relationship phrase in the input statement, e.g., in the statement 'wine **causes** cancer' the strength of statement is 6. We feed the standard multi-class classifiers with the annotated relation phrases for each statement in the training set while the training label is the strength of the statement. Since BERT-SL performs best among all relation phrase labeling approaches, we present all our subsequent results for this case only. We obtain the best parameters for each of the classifiers using the validation set. Finally, we report our results on the test set for 6-class, 4-class as well as 2-class scenarios. Unlike prior work (Li et al., 2017) that uses bag of words feature drawn from the whole statement, our model uses the features drawn from the annotated relation phrase, i.e., a part of the statement only. Note that, the results mentioned in the prior work is not directly comparable as the authors assume in their model that the relation phrases are already known, but we extract these phrases in the first place. However, we re-implemented their model (Li et al., 2017) and report the results that we got for our train-test division of the dataset.

Evaluation: We compare the strength classification methods using the micro-F1, which in this case, is equal to the accuracy. This is, as usual, calculated as the fraction of statements that have a correctly classified strength level out of the total number of statements. We also report the macro-F1 obtained for each model for the better understanding of the performance of the models primarily since the classes are unbalanced. This is calculated as the average of F1 scores obtained for each class as predicted by each model for the test set.

4.3 Exaggeration identification

In this module we calculate the strength level difference between a source statement and a target statement. The source statement is the one with the main claim from the journal article, while the target statement is the one with the main claim from corresponding news reports or press release covering the source. We calculate exaggeration (or underplay) levels for each classifier result over each type of strength classification.

Baseline: We present two types of baseline here. In the first, we pass the BERT embedding of the two statements (using respective 'CLS' markers)

---

[6] In our experiments we have seen that increasing dimension size further improves the results further. POS tags are obtained using NLTK.

[7] All other parameters remaining same, we use LSTM state size of 100, dropout rate of 0.5 and input batch size of 5.

[8] BERT NER: `https://github.com/kyzhouhzau/BERT-NER`

[9] `http://nlp.seas.harvard.edu/2018/04/03/attention.htm`

to the standard classifiers and attempt to estimate if there is a component of exaggeration. In the second, we model the problem as a textual entailment task which is a natural choice. Given a pair of statements the goal is to predict whether the second statement is an exaggerated version of first. For this purpose, we have used the BERT text classifier module. The final hidden vector of BERT model is passed through a softmax layer for the classification.

Evaluation: We employ *fraction of perfect match* (PM), which in our case, is also equal to the accuracy, as the measure to compare the different exaggeration detection methods. This is calculated as the fraction of source-target statement pairs for which we get correct exaggeration or underplay level as we have in the ground-truth. We also report the mean square error (MSE) for each model which is calculated as the average of the squares of the differences between actual and predicted strength difference for every pair of statements in each model.

5 Results

In this section, we present the results from the three phases – relation phrase labeling, strength classification and exaggeration identification one after the other. Finally we outline a scheme to spot the exact location of main claims in the news articles and press releases that particularly works for this dataset. This enabled us to fully automate the process based on this scheme and measure its performance.

5.1 Relation phrase labeling

We note down the main results in Table ??. The table shows that BERT-SL outperforms all other approaches by a large margin. The F1-score and the accuracy for this method is 0.85 and 0.95, respectively. Among the others LSTM-CRF augmented with PoS tags and PoS bigrams is the most competitive with a F1-score of 0.73 and accuracy of 0.89.

Success of BERT-SL: Table ?? shows the number of perfect matches between the ground-truth relationship phrase and the relationship phrase extracted using BERT-SL and LSTM-CRF. For all n-gram relationship phrases BERT-SL achieves a much higher number of perfect matches with the ground-truth. The difference in the number of matches obtained from the two methods is particularly large for unigrams and bigrams which covers the bulk of the statements. This is the reason why BERT-SL is able to outperform the other methods by a large margin.

Model	F1-score	Accuracy
DTH	0.58	0.82
OpenIE (large)	0.48	0.76
OpenIE (small)	0.37	0.70
LSTM-CRF (RI (Li et al., 2017))	0.67	0.88
LSTM-CRF (RI) + PoS	0.67	0.87
LSTM-CRF (GloVe) + PoS	0.72	0.88
LSTM-CRF (GloVe) + PoS+ PoS BI	0.73	0.89
LSTM-CRF (GloVe) + PoS tag + PoS BI + WC	0.73	0.88
LSTM-CNN-CRF (GloVe)	0.56	0.83
LM-LSTM-CRF (GloVe)	0.52	0.80
BERT-SL	**0.85**	**0.95**

Table 3: Comparison of the baselines with our approach. RI: Random Initialization, PoS: Part of speech tags, PoS BI: PoS Bigrams, WC: Wordnet clusters

#words	#BERT-SL	#LSTM-CRF
1	**33**	12
2	**34**	28
3	**28**	24
4	**9**	8
5	**6**	2
6	**3**	0

Table 4: Number of perfect matches for the competing models as per the number of words present in the relation phrase.

5.2 Strength classification

We predict the strength of a statement based on its relation phrase. For this purpose we build classifiers that take bag of words vector of the relationship phrase as identified by BERT-SL and predict its strength. The bag of words vector is created from the BERT embeddings as follows. The BERT embeddings of each word in the relation phrase are concatenated to form a single vector which is fed to the classifier. However the number of words in each relation phrase could be different. We fix this length by imagining all relation phrases to have length equal to the length of the largest relation phrase across the 1601 data points. This makes the concatenated vector size same for all the relation phrases; the missing entries in each vector so created are replaced by zero.

We use various classification models such as multinomial naive-bays (MNB), random-forest (RF), stochastic gradient descent (SGD) and XG-

Test type	Classifier	$\mu F1_{6class}$	$MF1_{6class}$	$\mu F1_{4class}$	$MF1_{4class}$	$\mu F1_{2class}$	$MF1_{2class}$
True rel. phs. (Trn:1300, Tst: 301)	MNB	0.68	0.52	0.77	0.62	0.79	0.79
	RF	0.5	0.16	0.51	0.22	0.54	0.48
	SGD	**0.76**	**0.70**	**0.79**	**0.76**	**0.82**	**0.82**
	XGB	0.74	0.69	0.76	0.73	0.76	0.76
BERT-SL rel. phs. (Trn: 1300, Tst: 301)	MNB	0.64	0.45	0.77	0.62	0.76	0.76
	RF	0.51	0.16	0.51	0.22	0.55	0.50
	SGD	**0.74**	**0.69**	**0.79**	**0.76**	**0.79**	**0.79**
	XGB	0.71	0.64	0.76	0.73	0.74	0.74
Baseline (Li et al., 2017) (Trn: 1300, Tst: 301)	BOW (unigram+bigram)	–	–	0.64	0.62	–	–

Table 5: Strength classification results. $\mu F1_{xclass}$: micro-F1 for x (6/4/2) class, $MF1_{xclass}$: macro-F1 for x (6/4/2) class.

Boost (XGB) for this purpose. The training of these classifiers is done on actual relation phrase and strength class pairs as annotated in the dataset. Table **??** shows the micro-F1 and macro-F1 scores of various classifiers across different strength classes (6, 4 and 2). In the table, entries for the true relation phrases represent the obtained micro-F1 and macro-F1 scores corresponding to a test set where the ground-truth annotated relation phrase (instead of what is obtained from BERT-SL) is taken into account. It is interesting to note that for all the three strength classes strengths obtain from BERT labeled relation phrases reaches close to the F1 values as one would have obtained if the true test relationship phrases were supplied to the classifiers at the input. Finally, our model by far outperforms the only known baseline (Li et al., 2017) in both micro and macro-F1 scores. To be fair, we show results for the 4-class which is the only case reported in (Li et al., 2017).

5.3 Exaggeration level identification

We identify the exaggeration level by taking the strength difference between the source statement from the journal and the target statement from the news article/ press release. The number of such source-target statement pairs in our test data is 316. The strength for each statement is obtained from the classifiers reported in the previous section. We present the results of exaggeration detection in Table **??**. Our three step approach attains a perfect match score for 0.62, 0.68 and 0.69 fraction of cases in the 6, 4 and 2 classes respectively (best classifier outputs) which is again close to what one could have obtained if the ground-truth relation phrases were known to the strength classifier in the previous stage.

Baseline: For the baseline we take the entire training set and construct as many possible source-target statement pairs where, by definition, the source statement is from a journal and the target statement is from a corresponding news article/press release. From the 1000 training statements, we could construct a total of 1298 such source-target pairs. If we feed the direct BERT embedding of the source and the target statement to the set of classifiers (RF, SGD and XGB in Table **??**) the results are much worse than our approach. Further if we adapt the BERT text classifier (as used in textual entailment detection module) to solve our problem ('Neural' in Table **??**), there is no benefit obtained. This proves that our proposed concept, a simple yet more interpretable three step sequential approach performs better than all other approaches where the whole sentence is taken into account for the exaggeration detection.

6 Discussion

In this section we present the findings of our error analysis and results correspond to the special sub cases where simple heuristics is taken into account to automate the whole process.

6.1 Error Analysis:

The error cases that we believe could be the possible reasons affecting the overall performance of the system are presented below.

- **Ambiguous modifiers**: Some statements have certain modifiers like 'little', 'any' etc in their predicted relation phrases, which leads to an error in the strength class prediction.

- **Incorrect relation phrase**: In some cases, the predicted relation phrase is incorrect which leads to propagation of errors in next steps.

- **Incorrect relation interpretation**: The system is unable to differentiate between cases like "could be entirely independent" and

Test type	Classifier	PM_{6class}	MSE_{6class}	PM_{4class}	MSE_{4class}	PM_{2class}	MSE_{2class}
True rel. phs. (Tst: 301, Pairs: 316)	MNB	0.53	5.71	**0.68**	1.27	**0.69**	0.37
	RF	0.49	**3.93**	0.51	**1.18**	0.52	0.53
	SGD	0.62	5.01	0.67	1.30	0.68	**0.35**
	XGB	**0.63**	4.65	0.63	1.41	0.65	**0.35**
BERT-SL rel. phs. (Tst: 301, Pairs: 316)	MNB	0.52	4.99	**0.68**	1.27	0.67	0.40
	RF	0.49	**3.88**	0.51	**1.18**	0.53	0.58
	SGD	**0.62**	4.74	0.67	1.30	**0.69**	**0.37**
	XGB	0.6	4.73	0.63	1.41	0.64	0.39
Baseline (Train: 1298, Pairs: 316)	RF	**0.48**	**4.24**	0.49	1.37	**0.61**	0.47
	SGD	0.43	5.13	0.33	1.39	0.49	0.54
	XGB	0.47	4.51	**0.51**	**1.28**	0.59	**0.46**
	Neural	0.44	–	0.50	–	0.56	–

Table 6: Exaggeration level identification results. PM_{xclass}: Perfect match for x (6/4/2) class, MSE_{xclass}: Mean squared error for x (6/4/2) class.

“could be entirely dependent”, which leads to the prediction of wrong strength classes.

We present examples of above cases in the supplementary material.

6.2 Spotting the causal statement:

We manually inspect the position of main claims in the documents. We find that in 95% of cases the main claim is present either in the title or in any one of the first three sentences of the document. We repeat the relationship phrase identification experiment using the best model (i.e., BERT-SL) for a test set that is built using all the four sentences (i.e., the title, the first, the second and the third sentences) taken from each press release and news article present in our dataset. Now we select that sentence for which we get the maximum number of words tagged as a relation phrase. Using this heuristic selection criteria we obtain F1-score of **0.8** and an accuracy of **0.9** which is close to what we report in Table ??. More detail of this experiment are presented in supplementary material.

6.3 Inclusion of IV and DV

Our model can also be used to extract the IV and the DV as well. We also check if this brings additional benefits to the pipeline. The details of this experiment is given in the supplementary material. The F1-score (precision, recall) we obtained for the relation phrase labeling task in this case is 0.72 (0.75, 0.7). This experiment also gives token level accuracy of 0.87. As we observe, the inclusion of the IV and DV information does not improve the labeling performance and therefore we did not use it for the next stages of the pipeline. The use of these additional information actually seems to confuse the labeler more than benefiting it. However, the automatic extraction of the IV and DV can be useful for developing other applications in future.

7 Conclusion

In this work, we proposed a simple yet explainable three step approach that automatically identifies whether a given statement typically from a press release or a news article is exaggerated in comparison to the source statement present in the journal. Our first step adapts the recently proposed BERT technology and models the relationship extraction problem as a sequence labeling task. This beats other sequence labeling and syntax driven approaches by a large margin. The relationship phrase extracted is encoded as a bag of words and fed to standard classifiers to obtain the strength of the phrase. Ground-truth labels of relationship phrases and those obtained from our model achieve similar performance. Finally, once the strength of the relationship phrase is available, a pair of statements can be easily compared. This method of exaggeration identification beats standard baselines that directly feed the two individual BERT embedding of the statement pair into binary classifiers or use the BERT textual entailment framework. In future, one can check if this kind of approaches work for other tasks where whole task can be divided into explainable subtasks.

Acknowledgments

This work is carried out by the authors during their study period at IIT Kharagpur. We thank Dr. Animesh Mukherjee and Dr. Pawan Goyal for their valuable inputs during the development of the work.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 344–354.

Catherine Blake. 2010. Beyond genes, proteins, and abstracts: Identifying scientific claims from full-text biomedical articles. *Journal of biomedical informatics*, 43 2:173–89.

Alessandro Bondielli and Francesco Marcelloni. 2019. A survey on fake news and rumour detection techniques. *Information Sciences*, 497:38–55.

Malcolm Coxall. 2013. *Human Manipulation-A Handbook*. Malcolm Coxall-Cornelio Books.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL*. Association for Computational Linguistics.

James E. Fulton-Jr., Gerd Plewig, and Albert M. Kligman. 1969. Effect of chocolate on acne vulgaris. *JAMA*, 210(11):2071–2074.

Georgios Giasemidis, Colin Singleton, Ioannis Agrafiotis, Jason R. C. Nurse, Alan Pilgrim, Chris Willis, and Danica Vukadinovic Greetham. 2016. Determining the veracity of rumours on twitter. In *SocInfo*.

Gulizar Haciyakupoglu, Jennifer Yang Hui, VS Suguna, Dymples Leong, and Muhammad Faizal Bin Abdul Rahman. 2018. Countering fake news: A survey of recent global initiatives.

Christopher S. G. Khoo, Syin Chan, and Yun Niu. 2000. Extracting causal knowledge from a medical database using graphical patterns. In *ACL*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Yingya Li, Jieke Zhang, and Bei Yu. 2017. An nlp analysis of exaggerated claims in science news. In *NLPmJ@EMNLP*.

Marc Light, Xin Ying Qiu, and Padmini Srinivasan. 2004. The language of bioscience: Facts, speculations, and statements in between. In *HLT-NAACL*.

Wee-Yong Lim, Mong-Li Lee, and Wynne Hsu. 2016. Claimfinder: A framework for identifying claims in microblogs. In *#Microposts*.

Liyuan Liu, Jingbo Shang, Frank Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2017. Empower sequence labeling with task-aware neural language model. *arXiv preprint arXiv:1709.04109*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1064–1074.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*.

Shivam B Parikh and Pradeep K Atrey. 2018. Media-rich fake news detection: A survey. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 436–441. IEEE.

Karishma Sharma, Feng Qian, He Jiang, Natali Ruchansky, Ming Zhang, and Yan Liu. 2019. Combating fake news: A survey on identification and mitigation techniques. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(3):1–42.

Petroc Sumner, Solveiga Vivian-Griffiths, Jacky Boivin, Andy Williams, Christos A Venetis, Aimée Davies, Jack Ogden, Leanne Whelan, Bethan Hughes, Bethan Dalton, et al. 2014. The association between exaggeration in health related science news and academic press releases: retrospective observational study. *Bmj*, 349:g7015.

Frans H Van Eemeren, Bart Garssen, and Bert Meuffels. 2009. *Fallacies and judgments of reasonableness: Empirical research concerning the pragma-dialectical discussion rules*, volume 16. Springer Science & Business Media.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Andreas Vlachos and Mark Craven. 2010. Detecting speculative language using syntactic dependencies and logistic regression. In *CoNLL Shared Task*.

Xinyi Zhou and Reza Zafarani. 2018. Fake news: A survey of research, detection methods, and opportunities. *arXiv preprint arXiv:1812.00315*.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Computing Surveys (CSUR)*.

Strength category	Description	Example
0	No relationship is mentioned	...An international study of 220,000 people has challenged the idea that obese people who have an "apple shape" - fat around the middle section of the body - are at higher risk of heart attacks and strokes...
1	Explicitly stating there is no relationship	Clinical officers and doctors **did not differ significantly** in key outcomes for Caesarean section significantly..
2	Statement of correlation - IV and DV are associated, but causation cannot be explicitly asserted	the greatest excess risk **associated** with subsequent primary neoplasms at older than 40 years was for digestive and genitourinary neoplasms
3	Ambiguous statement of relationship - It is unclear what the strength of relationship of this statement	...**has linked** eight new DNA variants to the autoimmune disease...
4	Conditional statement of causation - Causal statements show that the IV directly changes the DV. Conditional causal statements carry an element of doubt in them	...one in four patients **may be wrongly** diagnosed with high blood pressure...
5	Statement of "can" - The word "can" is unique as a statement of relationship in that it implies that the IV has the potential to directly change the DV, Therefore it is stronger than any conditional statement of causation.	...An intensive diet intervention soon after diagnosis **can improve** glycaemic control. The addition of an activity intervention conferred no additional benefit...
6	Statement of causation - The strongest are the statements of causation. This statement says that the IV definitely and directly alters the DV.	...capsules containing concentrated phytonnutrients **improved** clinical outcomes...

IV: Independent Variable DV: Dependent Variable

Table 7: Description of strength categories with an example.

A Annotated strength levels

The dataset contains manually coded strength levels of main claims from three sources, based on which, authors found that the press release is the main source of exaggeration in health science reports. The main causal claims in journal article, press release and news reports are coded into seven categories with increase in strength of relationship (see Table **??**). Relation phrases in the claim representing the relationship are marked in bold. The relation phrase connects the independent variable (*IV*) with the dependent variable (*DV*).

B Detais of dependency tree heuristics

The steps followed for extraction of relation phrase from causal statement are given as follows:

- We create dependency trees for each statement, with each node corresponding to a word and edges representing the grammatical relation between them. Let D be the set of all possible dependencies. We work with the collapsed version of dependencies, e.g., prepositions are not represented as nodes but collapsed into edges etc. For the statement, 'gestures improved performance in spatial visualisation problems', the dependency tree is shown in Figure 2. Here 'in' in the statement has been collapsed to the edge *prep_in*.
- We identify a subset $A \subseteq D$ of dependencies that we collapse to merge the connected nodes. The intuition is to collect words that form a coherent phrase inside a single node. The set A has been formed by going through the definition of each grammatical relation from the Stanford dependency manual. We call the nodes of the compact dependency tree as compact nodes. Each compact node is a tree of simple nodes and thus represents a substring. The grammatical relation between two

Order	Heuristic	Definition
1	LONG_VB_NSUBJ_DOBJ	Select compact node with a *verb* root, and *nsubj* and *dobj* out-edges
2	LONG_VB_NSUBJPASS_AGENT	Select compact node with a *verb* root, and *nsubjpass* and *agent* out-edges
3	LONG_VB_NSUBJPASS_PREP_WITH	Select compact node with a *verb* root, and *nsubjpass* and *prep_with* out-edges
4	LONG_VB_NSUBJ_DOBJ_NSUBJPASS	Select compact node with a *verb* root, and having atleast one *nsubj*, *dobj* or *nsubjpass* out-edge
5	LONG_VB	Select compact node whose root is a *verb*
6	LONG_JJ_NSUBJ_XCOMP	Select compact node with an *adjective* root, and *nsubj* and *xcomp* out-edges
7	LONG_JJ	Select compact node with an *adjective* root, and *nsubjpass* and *prep_with* out-edges
8	LONG_NOUN	Select compact node with a *noun* root

Table 8: Dependency tree heuristics.

compact nodes is the grammatical relation between the roots of their corresponding tree.

- The set A consists of the following dependencies - *advmod, amod, appos, aux, auxpass, cop, det, expl, mwe, mark, neg, nn, npadvmod, num, number, pobj, poss, possessive, predet, prt, quantmod* and *vmod*. Note that most relations in A are modifiers. In the previous example (refer to Figure 2) the subtree with nodes 'problems'/NNS, 'spatial'/JJ and 'visualization'/NN are merged together to form a compact node 'spatial visualization problem'/NNS.
- The next step is to select an appropriate compact node. We use heuristic functions, that take a dependency tree as input and output a node (if its condition is met) or *nil*. We apply a sequence of heuristic functions $h1, h2, h3, ...$ to recover the relational phrase (see Table **??** for the list of heuristics). If $h1$ returns a node, we identify its constituent phrase as our relational phrase. Else if $h1$ returns *nil*, we select $h2$. If $h2$ returns a node, we use that else we try $h3$ and so on. The heuristic functions have been hand-coded by observing the statistics of the best compact node (highest normalized lexicalized edit similarity ($NLES$) with the annotated phrase) in the dependency trees of the training set.
  The $NLES$ between two phrases p_1 and p_2 is defined as

$$NLES(p_1, p_2) = 1 - \frac{dist(p_1, p_2)}{min(|p_1|, |p_2|)} \quad (1)$$

where, $|p_1|, |p_2|$ are the number of characters in p_1 and p_2 respectively. $dist$ refers to the standard edit distance where cost of the substitution, the insertion and the deletion are all taken as 1. We also tried jaccard similarity between lexicalized tokens in place of $NLES$ but we same set of heuristic sequence.

The list of heuristic functions is given in Table **??** along with their order of application, which has been found by iterating through all permutations and choosing the one that produces maximum average $NLES$ across all input training sentences. Each heuristic function proceeds according to its definition; if multiple nodes satisfy the heuristic condition, the node with the most number of words is chosen. If none of the nodes satisfy the heuristic condition, the function returns *nil*. For instance, in the previous example (refer to Figure 2), heuristics LONG_VB, LONG_VB_NSUBJ_DOBJ and LONG_NOUN match and return phrases, however, since LONG_VB has the highest priority so the respective phrase 'improved' is returned as the relation phrase. In another example 'Childhood cancer survivors at greater risk in middle age', no heuristics match to its compact tree and hence *nil* is returned as relation phrase.

C Error Analysis

In this section we discuss some of the error cases that we believe could be the possible reasons affecting the overall performance of the system. In future we plan to tackle some of these cases to further improve the system performance. We mention below some of these error cases. The examples of error cases are presented in Table **??**.

Ambiguous modifiers: Some statements have certain modifiers in their predicted relation phrases,

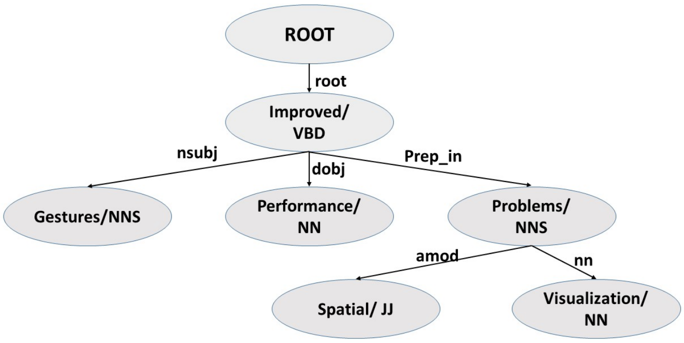


Figure 2: Collapsed dependency tree.

No.	Statement	Ann.Ph.	Pred. Ph.	A_S	P_S
1	a common treatment for a life-threatening heart condition has little significant impact on patient outcomes	has little significant impact on	has little significant impact on	1	6
2	We noted little variation between countries in the rate of maltreatment-related injury admission	little variation	little variation	1	6
3	this study excluded any large increase in the incidence of cases of or deaths from infective endocarditis	excluded any large increase	excluded any large increase	1	6
4	Falling in love sets brain circuits racing in the same way, regardless of sex or sexual orientation	regardless of	sets	1	6
5	breast cancer screening ... does women more harm than good	does	does women more harm than	6	2
6	that the progressive loss of lung function in asthma sufferers could be entirely independent of the effects of inflammation	could be entirely independent of	could be entirely independent	1	4

Table 9: Errors in strength classification. A_S: Actual strength, P_S: Predicted strength

Location	%
Title	26.10
First statement	46.12
Second statement	21.93
Third statement	2.14

Table 10: Location of the statement with the main claim in the press-releases/news articles.

which leads to an error in the strength class prediction. For instance, consider statements 1, 2 and 3 shown in Table ??. In all these cases the presence of the modifiers such as ‘little’, ‘any’ etc. are not separately tackled by the strength classifier.

Incorrect relation phrase: In some cases, the predicted relation phrase is incorrect. For example, consider the statement 4. Here the annotated phrase is ‘regardless of’ whereas the predicted phrase is ‘sets’. Another interesting case is statement 5. Here the predicted phrase fully contains the actual phrase and is also much larger than the actual phrase. The classifier therefore confuses the strength class.

Incorrect relation interpretation: In some cases, like statement 6, the system is unable to differentiate between “could be entirely independent” and “could be entirely dependent”, which leads to prediction of class 4 instead of the actual class 1.

D Spotting the causal statement

So far, we have assumed that statement with the main claim is already given to us for labeling and exaggeration identification. However, in order to completely automate the process we need to spot the exact location of these statements so that the previous pipeline of three sub-tasks could be smoothly executed. Manual inspection of the documents indicates that the statement with the main claim is present either in the title or in any one of the first three sentences of the document. In fact, we run an experiment on all the documents in our dataset and observe the location of the main statement. Table **??** shows the percentage of documents in the dataset, in which this statement is present in the title or in the first, second or the third sentence. Surprisingly, in 95% of the documents the causal statement is present in the title or the first three sentences of the document.

We repeat the relationship phrase identification experiment using the best model (i.e., BERT-SL) for a test set that is built using all the four sentences (i.e., the title, the first, the second and the third sentences) taken from each press release and news article present in our dataset. For each press release/news article, all the four sentences are labeled by the sequence labeler. Now we select that sentence for which we get the maximum number of words tagged as a relation phrase. Using this heuristic selection criteria we obtain F1-score of **0.8** and an accuracy of **0.9**. Given this encouraging result, we do not go for any additional algorithmic machinery for separately spotting the statement with the main claim for the dataset. However one can check in future if this works well for other problems and datasets.

E Inclusion of IV and DV

Our model can also be used to extract the IV and the DV as well. We also check if this brings additional benefits to the pipeline. Though the existing annotation of the dataset identifies the entities, i.e., the IVs and DVs associated with each document (journal/press-release/news article), in many cases, they are not explicitly part of the statement with the main claim (i.e., the entities are metaphorically mentioned); therefore, automatic labeling of these variables was impossible without re-annotation. To this purpose, we re-annotated the entire dataset for these two entities. Next we tasked the BERT-SL to identify the IV, DV and relation phrase altogether. The F1-score (precision, recall) we obtained for the labeling task in this case is 0.72 (0.75, 0.7). This experiment also gives token level accuracy of 0.87. As we observe, the inclusion of the IV and DV information does not improve the labeling performance and therefore we did not use it for the next stages of the pipeline. The use of these additional information actually seems to confuse the labeler more than benefiting it. However, the automatic extraction of the IV and DV can be useful for developing other applications in future.

Modeling Coreference Relations in Visual Dialog

Mingxiao Li
KU Leuven
mingxiao.li@kuleuven.be

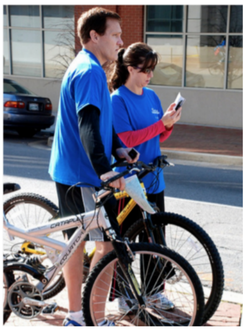
Marie-Francine Moens
KU Leuven
sien.moens@cs.kuleuven.be

Abstract

Visual dialog is a vision-language task where an agent needs to answer a series of questions grounded in an image based on the understanding of the dialog history and the image. The occurrences of coreference relations in the dialog makes it a more challenging task than visual question-answering. Most previous works have focused on learning better multi-modal representations or on exploring different ways of fusing visual and language features, while the coreferences in the dialog are mainly ignored. In this paper, based on linguistic knowledge and discourse features of human dialog we propose two soft constraints that can improve the model's ability of resolving coreferences in dialog in an unsupervised way. Experimental results on the VisDial v1.0 dataset shows that our model, which integrates two novel and linguistically inspired soft constraints in a deep transformer neural architecture, obtains new state-of-the-art performance in terms of recall at 1 and other evaluation metrics compared to current existing models and this without pretraining on other vision-language datasets. Our qualitative results also demonstrate the effectiveness of the method that we propose. [1]

1 Introduction

Recently, with the unprecedented advances in computer vision and natural language processing, we have seen a considerable effort in developing artificial intelligence (AI) agents that can jointly understand visual and language information. Visual-language tasks, such as image captioning (Xu et al., 2015) and visual question-answering (VQA) (Antol et al., 2015), have achieved inspiring progress over the past few years. However, the applications of these agents in real-life are still quite limited, since they cannot handle the situation when continuous information exchange with a human is necessary, such as in visual-language navigation (Anderson et al., 2018b) and visual dialog (Das et al., 2017). The visual dialog task can be seen as a generalization of VQA. Both tasks require the agent to answer a question expressed in natural language about a given image. A VQA agent needs to answer a single question, while a dialog agent has to answer a series of language questions based on its understanding of visual content and dialog history. Compared to VQA, the visual dialog task is more difficult because it demands the agent to resolve visual coreferences in the dialog. Considering the example in Figure 1. when the agent encounters question 6 "do they look old or new ?" and questions 9 "are they tall or short ?", it has to infer that the pronoun "they" in these two questions refers to different entities in the image or the dialog history.



Caption: A man and woman on bicycles are looking at a map
Person A (1) : where are they located
Person B (1) : in city
Person A (2) : are they on road
Person B (2) : sidewalk next to 1
Person A (3) : any vehicles
Person B (3) : 1 in background
Person A (4) : any other people
Person B (4) : no
Person A (5) : what color bikes
Person B (5) : 1 silver and 1 yellow
Person A (6) : do they look old or new
Person B (6) : new bikes
Person A (7) : any buildings
Person B (7) : yes
Person A (8) : what color
Person B (8) : brick
Person A (9) : are they tall or short
Person B (9) : I can't see enough of them to tell
Person A (10) : do they look like couple
Person B (10) : they are

Figure 1: An example taking from the VisDial v1.0 dataset. The questioner (Person A) sees the caption and tries to understand the whole scene of the image by asking questions to the answerer (Person B) who can see the whole image.

This paper studies how we can improve the results of a visual dialog task by better resolving the coreference relations in the dialog. In this work we restrict coreference resolution to pronouns. We

---

[1] Our code are released on: https://github.com/Mingxiao-Li/Modeling-Coreference-Relations-in-Visual-Dialog

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3306–3318
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

use a multi-layer transformer encoder as our baseline model. Based on the assumption that the output contextual embedding of a pronoun and its antecedent should be close in the semantic space, we propose several soft constraints that can improve the model's capability of resolving coreferences in the dialog in an unsupervised way (i.e., without ground truth coreference annotations in the training data). Our first soft constraint is based on the linguistic knowledge that the antecedent of a pronoun can only be a noun or noun phrase. To integrate this constraint in the baseline model, we introduce a learnable part-of-speech (POS) tag embedding and a part-of-speech tag prediction loss. Inspired by the observation that in human dialog the referents of pronouns often occur in nearby dialog utterances, we propose a second soft constraint using a sinusoidal sentence position embedding, which aims to enhance local interactions between nearby sentences.

Our contributions are as follows: First, as a baseline we adapt the multi-layer transformer encoder to the visual dialog task and obtain results comparable to the state of the art. Second, we propose two soft constraints to improve the model's ability of resolving coreference relations in an unsupervised way. We also perform an ablation study to demonstrate the effectiveness of the introduced soft constraints. Third, we conduct a qualitative analysis and show that the proposed model can resolve pronoun coreferents by making sure that in the neural architecture the pronoun mostly attends to its antecedent.

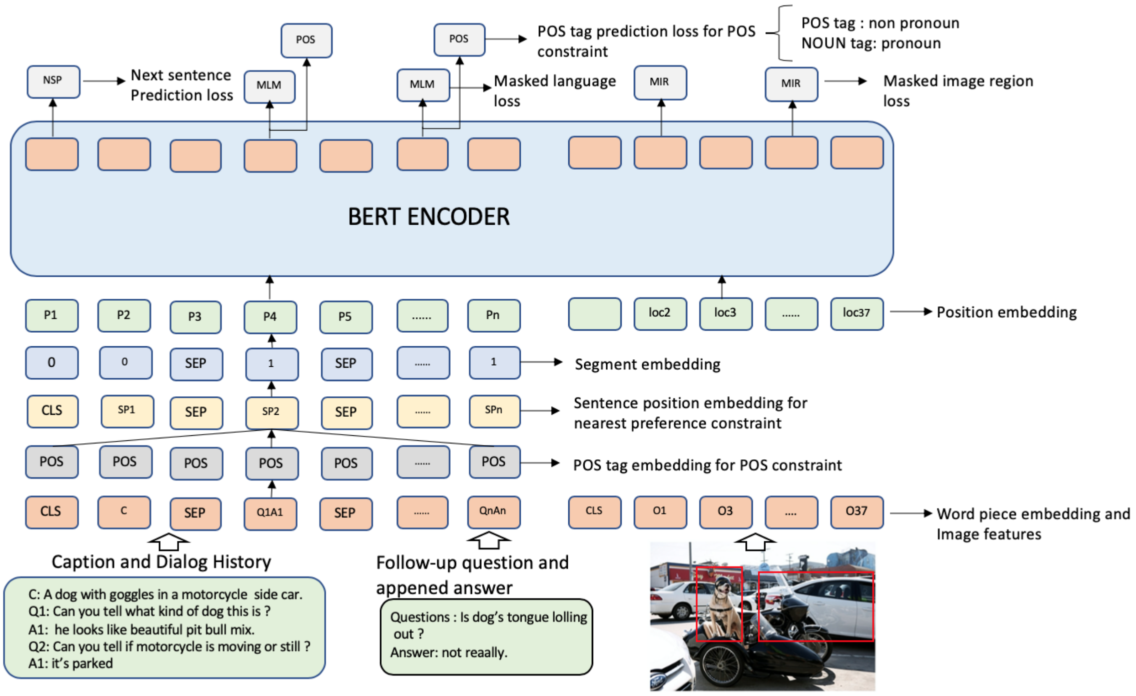
2 Related Work

Visual Dialog. The Visual Dialog task is proposed by Das et al. (2017), where a dialog agent has to answer questions grounded in an image based on its understanding of the dialog history and the image. Most of previous work focuses on using an attention mechanism to learn interactions between image, dialog history and question. Gan et al. (2019) use an attention network to conduct multi-step reasoning in order to answer a question. Niu et al. (2019) propose a recursive attention network, which selects relevant information from the dialog history recursively. Kang et al. (2019) apply a multi-head attention mechanism (Vaswani et al., 2017) to learn mutimodal representations. Schwartz et al. (2019) fuse information from all entities including question, answer, dialog history, caption and image using a factor graph. Murahari et al. (2019) propose two-stage training. They first pretrain their transformer based two-stream attention network on other visual-language datasets, then finetune it on the visual dialog dataset. Other approaches consider different learning methodologies to model the visual dialog task, for example, Lu et al. (2017) use adversarial learning and Yang et al. (2019) apply reinforcement learning.

Coreference Resolution. Coreference resolution aims at detecting linguistic expressions referring to the same entities in the context of the discourse. The task has been dominated by machine learning approaches since the first learning based coreference resolution system was proposed by Connolly et al. (1997). Before Lee et al. (2017) proposed the first end-to-end neural network based coreference resolution system, most of the learning-based systems have been built with hand engineered linguistic features. Durrett and Klein (2013) use surface linguistic features, such as mention type, the semantic head of a mention, etc., and their combinations to build a classifier to determine if two mentions refer to the same entity. Do et al. (2015) adopt integer linear programming (ILP) to introduce coreference constraints including centering theory constraints, direct speech constraints and definite noun phrase and exact match constraints in the inference step in order to adapt an existing coreference system trained on the newswire domain to short narrative stories without any retraining. Recently, Joshi et al. (2019) apply a BERT model to coreference resolution and achieve promising results on the OntoNotes corpus (Pradhan et al., 2012) and the GAP dataset (Webster et al., 2018). Different from all coreference systems mentioned above, which rely on supervised learning and on a dataset annotated with coreference links, our work focuses on applying soft linguistic constraints to improve the model's ability of resolving coreferents in an implicit and unsupervised way. Similar to the work of Venkitasubramanian et al. (2017) that operates on language and vision information, our model uses attention to jointly learn multi-modal representations.

3 Methodology

In this section, we formally describe the visual dialog task (Das et al., 2017) and the approaches we propose. In visual dialog, given an image I, the image caption C and the dialog history until round $t-$



C: A dog with goggles in a motorcycle side car.
Q1: Can you tell what kind of dog this is ?
A1: he looks like beautiful pit bull mix.
Q2: Can you tell if motorcycle is moving or still ?
A1: it's parked

Questions : Is dog's tongue lolling out ?
Answer: not reaally.

Figure 2: The model architecture with the two soft constraints that we propose. The baseline model takes the image feature I, image caption C, dialog history H_t, follow-up question Q_t and appended candidate answer A_t as input and is trained by using a masked language model (MLM), masked image region (MIR) and next sentence prediction (NSP) losses. Two soft constraints are integrated into the model by adding POS tag embedding and sentence position embedding to the input language sequence embedding and by introducing a new POS tag prediction objective during training.

1, $H = ((Q_1, A_1), (Q_2, A_2), \cdots, (Q_{t-1}, A_{t-1}))$, which is a sequence of question-answer pairs expressed in natural language and grounded in the image, a dialog agent is expected to correctly answer the question at round t by choosing the answer from 100 candidate answers $A_t = \{A_t^1, A_t^2, \cdots, A_t^{100}\}$.

We first introduce the baseline visual dialog model in Section 3.1, followed by the detailed explanation of our proposed soft coreference constraints and how we integrate these into the baseline model in Section 3.2. The coreference constraints are based on linguistic knowledge, so consequently our method is an example of how to integrate linguistic knowledge into a neural transformer architecture. Figure 2 shows the architecture of our proposed model.

3.1 Baseline Model

Transformer Encoder. We use a multi-layer transformer (Vaswani et al., 2017) encoder as our baseline model. The computation within a single layer transformer encoder is presented in appendix A.1, and the details of the input and training objective functions are illustrated in below subsections. The main idea of applying the transformer architecture to the visual dialog task is to use the multi-head self-attention mechanism to implicitly learn the intra and inter interactions within the single modality and between the different modalities (in this case language and vision), respectively.

Linguistic Representation. Following the monolingual BERT model (Devlin et al., 2019) and the multi-modal BERT model (Lu et al., 2019; Li et al., 2019; Su et al., 2020), we use the WordPiece (Wu et al., 2016) tokenization tool to tokenize each input sequence into word pieces sequence. Then the sum of the word piece embedding, position embedding and segment embedding, where the segment embedding is used to differentiate questions from answers and to delimit boundaries of question-answer pairs, are taken as the language sequence input of the model.

Image Representation. Following the multi-modal BERT model (Anderson et al., 2018a; Lu et al., 2019; Li et al., 2019; Su et al., 2020), we use Faster-RCNN (Ren et al., 2015) with ResNet (He et al., 2016) backbone to detect objects in the image and keep the top-36 detected objects and their corresponding bounding boxes. The representation of each detected object is obtained by applying a mean-pooled convolution on the region of that object. We also project a $5-d$ geometrical representation of each box, including the normalized top-left and bottom-right coordinates of the detected objects and the fraction of the area they cover, to the same dimensions as the image feature vector. In this way we obtain a vector with the same dimensions as the feature representation of the image. The final input image representation is the sum of its geometrical and feature representations. To avoid missing image information that is not captured by the top-36 bounding boxes, we also concatenate the mean-pooled feature vector of the whole image to the beginning of the image region sequence.

Multi-modal Input. As the transformer encoder receives a sequence of tokens as input, to feed both image and language into the model, we simply concatenate the language sequence embedding and image region sequence representation to form a whole input sequence. Like in the BERT model, a special token [CLS] is added to the beginning of the input sequence to perform the next sentence prediction task. We also use another special token [SEP] to separate each question-answer pair and the two modalities. Our input sequence can be formulated as follows: $Input = \{[CLS], C, [SEP], Q_1A_1, [SEP], Q_2A_2, \cdots, Q_t A_t, [SEP], O_0, O_1, O_2, \cdots, O_{36}\}$, where C is the image caption. Q_i,A_i are question and answer at round i, and $O_{0\sim 36}$ denote the input image region features.

Multitasks Training Objectives. To make the model learn a good alignment between different modalities, we utilize three losses: masked language model loss (MLM), masked image region loss (MIR), and next sentence prediction loss (NSP). Similar to the MLM in BERT, we randomly mask 15% word pieces in the language sequence by replacing the word piece with a special token [MASK], while in MIR, we randomly set 15% of the image region features to zero vectors. The model is trained to recover the masked words and predict the semantic category of the masked image regions:

$$L_{MLM} = -E_{(I,w)\sim D} log P(w_m | w_{\backslash m}, I) \quad (1)$$

$$L_{MIR} = \sum_{i}^{k} KL(P_m || P_g) \quad (2)$$

where $w_{\backslash m}$ and I denote the word sequence excluding the masked words w_m, and image regions, respectively. KL represents the KL divergence loss. P_m is the model output distribution and P_g is the ground truth classification distribution.

Recall that the visual dialog system aims to find the correct answer among the 100 candidate answers. We realize this in a discriminative manner by using the next sentence prediction loss (NSP). We randomly select 1 wrong answer from the candidate answers to generate negative samples, together with the ground-truth to form a balanced training dataset. During training, a candidate answer is appended to the dialog sequence, and the model is trained to predict whether or not the appended answer is the correct answer to the current question:

$$L_{NSP} = -E_{(I,w)\sim D} log P(\hat{y} | I, w) \quad (3)$$

where $\hat{y} \in [0, 1]$ is the output probability of the binary classifier at the last layer using the special [CLS] tag representation, which indicates the probability of the appended answer being correct. During inference, we rank the 100 candidate answers using their NSP score, which is the $\hat{y}$ in the above equation. During training, the total loss is the sum of MLM, MIR and NSP losses:

$$L_{total} = L_{MLM} + L_{MIR} + L_{NSP} \quad (4)$$

3.2 Soft Coreference Constraints

As discussed before, the existence of pronouns in language makes the visual dialog a more challenging task than VQA. A naive way to reduce the difficulty would be to use a loss to guide the model to jointly learn to resolve the coreferences in the dialog and to generate an answer to the question. However, the lack of coreference annotations in the visual dialog dataset prevents from using this supervised learning approach. Although it is impossible to resolve coreferences directly in the model, we propose to use linguistic knowledge to improve the model's ability to implicitly resolve the coreferences in an unsupervised way. We do so by exploiting the attention mechanisms of the transformer architecture where attention weights act as soft constraints to guide the training of the model. The intuition behind it is the following. As the baseline model will output a contextual representation for each input token in the last layer, if a pronoun refers to a noun or noun phrase in the input sequence, the output contextual embedding of this pronoun and its antecedent should be close in the semantic space, which also means that the pronoun should attend most to its antecedent.

Part-Of-Speech Constraint. Our first proposed soft constraint is based on the linguistic knowledge that if the antecedent of a pronoun exists, it can only be a noun or noun phrase. We use the POS tag information and introduce a POS tag prediction loss to help pronouns to find nouns in the dialog. The Stanford CoreNLP POS tagger (Manning et al., 2014) is used to obtain the POS tag of each word in the input dialog, and all sub-word splits from a word share the same POS tag. Similar to the word embedding, we use a learnable embedding for each POS tag, which is further summed with the word piece embedding, position embedding and segment embedding to form the input sequence embedding of the model. In POS prediction loss, similar to the MLM loss, we randomly mask 15% of POS tag of input tokens. Then, the model is trained to predict the ground truth POS tag[2] of non-pronoun masked words, while for masked pronouns we replace the ground truth PRP tag with NN tag forcing the model to learn the contextual pronoun embedding that is close to nouns in the semantic space. The POS prediction loss can be formulated as the equation below:

$$\begin{aligned}
&L_{POS} = -E_{(w,I)\sim D}(P_{non-pronoun} + P_{pronoun}) \\
&P_{non-pronoun} = -logP(POS(w)|w_{\setminus m}, I) \\
&P_{pronoun} = logP(NN|w_{\setminus m}, I)) \qquad (5)
\end{aligned}$$

where $w_{\setminus m}$ denote all unmasked words, and D is the dataset. This soft constraint will make pronouns focus more on nouns instead of other words such as verb, adverb or adjective, etc. As it does not violate any linguistic rules, it will not introduce a bias to the language model.

Nearest Preference Constraint. Our second soft constraint is inspired by the observation that in human dialog a pronoun is more likely to refer to the noun that is close to it. For example, in the visual dialog shown in Figure 1, in round 9 the pronoun “they” refers to the “buildings”, in round 6 “they” refers to “bikes”. However, it is not always the case that pronouns refer to the noun closest in the previous utterances hence our soft constraint. In visual dialog, some pronouns refer to noun phrases that occur much earlier in the discourse - skipping a few rounds - as utterances are very short. To integrate this preference into the model, we adapt the sinusoidal word position embedding proposed in (Vaswani et al., 2017) and introduce a sentence position embedding:

$$PE_{pos,2i} = \frac{1}{k} sin(pos/(M + 10000^{\frac{2i}{d}})) \qquad (6)$$

$$PE_{pos,2i+1} = \frac{1}{k} cos(pos/(M + 10000^{\frac{2i}{d}})) \qquad (7)$$

where pos is the sentence position in the dialog, d is the hidden state size and the M is the maximum number of sentences, which is 21 in this visual dialog task. k is a scaling factor to control the local interactions brought by sentence position embedding and we use $k = 100$. Compared to the original sinusoidal position embedding proposed in (Vaswani et al., 2017), our sentence position embedding has one more scaling factor and one more element M in the denominator, which aims at restricting the product of the sentence position embedding to be a monotonically decreasing function with respect to $|pos_1 - pos_2|$.

$$PE_{pos_1} \cdot PE_{pos_2} = \frac{1}{k^2} \sum_{i=0}^{\frac{d}{2}-1} cos(w_i \Delta pos) \qquad (8)$$

where $w_i = 1/(M + \epsilon^{\frac{2i}{d}})$, and Δpos denotes the distance between two positions. ϵ is a parameter, which makes the wavelength of the sinusoidal function in each dimension to form a geometrical progression.[3] Since $\Delta pos \in [-M, M]$, $w_i \Delta pos \in [-1, 1]$, it follows that $cos(w_i \Delta pos) == cos(w_i |\Delta pos|)$ which is monotonically decreasing in the region of $[0, 1]$.The details of the derivation of equation 8 are presented in appendix A.2. The closer two sentences are, the larger of the product of their sentence position embedding, resulting in stronger local interactions between nearby sentences in the dialog. This soft constraint can be easily integrated into the model by adding the sentence position embedding to the

---

[2] We use the POS tags used in Penn Treebank (Marcus et al., 1993).

[3] Following Vaswani et al. (2017), we set $\epsilon = 10000$.

input sequence embedding including word piece embedding, position embedding, segment embedding and POS tag embedding.

4 Experiments

4.1 Dataset

We use the real environment VisDial v1.0 dataset in this work. The VisDial v1.0 has 123k, 2k and 8k dialogs for training, validation and test, respectively. Each dialog contains one image with its caption from the MS-COCO dataset (Lin et al., 2014), and ten rounds of question-answer pairs, which were collected from the chatting log of one questioner and one answerer who both were discussing the image. For each question, except for the correct answer, the dataset also provides another 99 candidate answers to form an answer pool from which a model needs to select the relevant answer. Note that, although the VisDial v1.0 dataset also contains a small dense annotation set in which a relevance score is given to each candidate answer in the answer pool, we do not use this small dataset to finetune all our models, as we consider recall at 1 as main evaluation metric, and finetuning on this dense dataset could degrade the model's performance when measured by recall at 1 (Murahari et al., 2019).

4.2 Evaluation Metrics

We evaluate our proposed models using three evaluation metrics: (1) mean reciprocal rank (MRR) (Voorhees, 1999); (2) recall @ k, that is, the existence of the ground truth response in the top-k ranked items of the response list generated by the model with k = 1, 5, or 10; and (3) mean rank (Mean) of the ground truth response, that is, the average rank of the ground truth answer in the model's output ranked list (lower is better).

4.3 Training Details

Inspired by the open source code[4] of Murahari et al. (2019), we have implemented our model using the PyTorch framework (Paszke et al., 2019). Our model has the same configuration as the BERT$_{\text{BASE}}$ model, which contains 12 transformer layers and each layer has 12 attention heads with a hidden state size of 768. We set the maximum input length to be 256 including 37 image features. All models were trained using the Adam (Kingma and Ba, 2014) algorithm with a base learning rate of $5e^{-5}$. A linear learning rate decay schedule is employed to increase the leaning rate from $5e^{-6}$ to 5^{-5} over $30k$ iterations and decay to $5e^{-6}$ over $40k$ iterations. Together with negative samples, each image in the VisDial v1.0 dataset can generate 20 samples for the NSP task. Since these samples are fairly correlated and following the work of Murahari et al. (2019), we randomly sub-sample 8 out of these 20 during training. We use the validation set to decide when to stop training. The batch size is 32 in all our experiments, and different from the work of Murahari et al. (2019) and Lu et al. (2019), we do not pretrain our model on other vision-language datasets.

5 Results

5.1 Quantitative Results

We compare the results of our model with the results of the following previously published models obtained on the VisDial v1.0 dataset: LF (Das et al., 2017), HRE (Das et al., 2017), MN (Das et al., 2017), CorefNMN (Kottur et al., 2018), FGA (Schwartz et al., 2019), RVA (Niu et al., 2019), HACAN (Yang et al., 2019), Synergistic (Guo et al., 2019), DAN (Kang et al., 2019), Dual VD (Jiang et al., 2020), and CAG (Guo et al., 2020). To make a fair and transparent comparison, we do not compare our models with models which were pretrained on other vision-language datasets before finetuning them on the VisDial v1.0 dataset, all the more because the vision-language datasets used in the pretraining overlap with the testset of Visdial v1.0. Also for those models, such as FGA, for which the authors also provide results of ensemble models, we only consider the results of their single model.

Results on VisDial v1.0 testset. As presented in Table 1, our best model (baseline model with both soft constraints) significantly outperforms all the previous published models and reach new state-of-the-art performance on MRR, R@1, R@5 and R@10. Specifically, compared to the best performance of previous models, our best model improves around 2.3% on MRR, 1.78% on R@1, 2.13% on R@5 and 2.35% on R@10. The mean rank of our proposed model is also better than all previous models, although the difference is relative small around 0.71. We also tested all our models on the VisDial v1.0 development set, and the results are presented in appendix A.3.

Ablation study. To further study the effectiveness of the two soft constraints, we perform an abla-

---

[4] https://github.com/vmurahari3/visdial-bert

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
LF (Das et al., 2017)	55.42	40.95	72.45	82.83	5.95
HRE (Das et al., 2017)	54.16	39.93	70.45	81.50	6.41
MN (Das et al., 2017)	55.49	40.98	72.30	83.30	5.92
CorefNMN (Kottur et al., 2018)	61.50	47.55	78.10	88.80	4.51
FGA (Schwartz et al., 2019)	63.70	49.58	80.98	88.55	4.51
RVA (Niu et al., 2019)	63.03	49.03	80.40	89.83	4.18
HACAN (Yang et al., 2019)	64.22	50.88	80.63	89.45	4.20
Synergistic (Guo et al., 2019)	62.20	47.90	80.43	89.95	4.17
DAN (Kang et al., 2019)	63.20	49.63	79.75	89.35	4.30
Dual VD (Jiang et al., 2020)	63.23	49.25	80.23	89.70	4.11
CAG (Guo et al., 2020)	63.49	49.85	80.63	90.15	4.11
Baseline Model	62.13	47.38	80.40	90.17	4.09
Model + POS Embedding Only	64.20	48.10	81.22	90.20	3.98
Model + POS Loss only	64.78	49.88	82.40	90.85	3.86
Model + C1	65.44	51.20	83.38	92.03	3.64
Model + C2	66.14	51.97	83.63	91.55	3.60
Model + C1 + C2	**66.53**	**52.63**	**84.13**	**92.50**	**3.40**

Table 1: Results of the visual dialog models on the VisDial v1.0 test set. C1 and C2 refer to the POS constraint and nearest preference constraint, respectively.(↑: the higher the better; ↓: the lower the better)

tion study on the VisDial v1.0 dataset with four different models: (1) Baseline model; (2) Model with the POS soft constraint (Model + C1); (3) Model with the nearest preference constraint (Model + C2); (4) Model with both the POS and nearest preference constraints (Model + C1 + C2). Moreover, We conduct an ablation study for the two aspects (POS embedding and POS prediction loss) in POS constraint The results are presented in Table 1. The baseline model obtains the following results in terms of MRR (62.13%), R@1 (47.38%), R@5 (80.40%) and R@10 (90.17%). Models with only POS embedding and only POS prediction loss have better performance than the baseline model. Further combining both leads to our first POS constraint, which improve the performance across all evaluation metrics (3.31% for MRR, 3.82% for R@1, 2.98% for R@5, 1.86% for R@10 and 0.55 for MRR). Similarly, only considering the nearest preference constraint leads to better performance on all evaluation metrics. The last row of Table 1 illustrates that the proposed soft constraints jointly lead to better results. We also study the changes of attention distribution of the model with and without our proposed constraints. Figure 3 shows that the nearest constraint can enhance the local connection in dialog, and the POS constraint is able to make pronouns focus more on nouns. These results indicate the effectiveness of adding linguistic constraints to a neural network. Integrating linguistic knowledge in a transformer neural architecture effectively improves the model's performance in the visual dialog task.

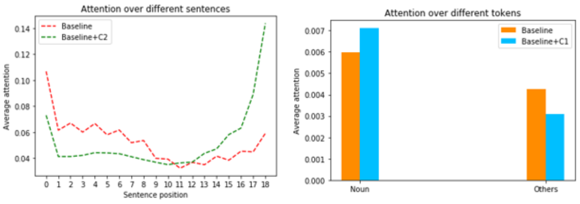


Figure 3: Left: Attention distribution of question over dialog history (Baseline Model+C2). Right: Attention distribution of pronoun over nouns and other words. (Baseline Model+C1

5.2 Coreference Analysis

As we do not have the access to the ground truth of the VisDial v1.0 test set, to further analyze the effectiveness of the proposed models, we create three small datasets, which each consists of samples with 2, 4, and 6 coreferences, respectively, from the Visdial v1.0 validation set. As there is no coreference annotation in the Visdial v1.0 dataset, we assume that each third-person pronoun (he, she, they, him, her, them) and possessive pronouns (its, his, her, their) has one coreference in the dialog.[5]

---

[5] Note that the dialog is about the objects in an image.

Results tested on validation set with 2 coreferences (237 samples)

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
Baseline Model	65.06	51.38	82.42	91.79	3.77
Model + C1	67.44	54.04	**84.54**	92.58	3.38
Model + C2	67.19	54.03	84.30	92.08	3.55
Model + C1 + C2	**67.61**	**54.45**	**84.54**	**92.87**	**3.34**

Results tested on validation set with 4 coreferences. (138 samples)

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
Baseline Model	62.94	48.93	80.78	90.28	3.99
Model + C1	65.77	51.99	83.49	**93.00**	3.48
Model + C2	65.94	52.07	83.36	92.00	3.52
Model + C1 + C2	**66.53**	**52.78**	**84.21**	92.07	**3.46**

Results tested on validation set with 6 coreferences. (58 samples)

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
Baseline Model	60.02	43.99	80.80	90.40	4.06
Model + C1	62.44	46.40	**83.86**	92.66	3.61
Model + C2	63.53	47.09	83.33	92.28	3.59
Model + C1 + C2	**63.66**	**48.14**	83.33	**92.67**	**3.42**

Results tested on validation set with coreferences. (1227 samples)

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
Baseline Model	64.62	50.73	82.40	91.42	3.73
Model + C1	**66.29**	52.41	84.17	92.08	3.62
Model + C2	**66.29**	52.54	83.46	91.25	3.59
Model + C1 + C2	65.97	**53.27**	**83.47**	**92.41**	**3.48**

Table 2: Results of the visual dialog models obtained on three small datasets each with a different number of pronoun coreferences that were collected from the VisDial v1.0 validation set. C1 and C2 refer to the POS constraint and nearest preference constraint, respectively.

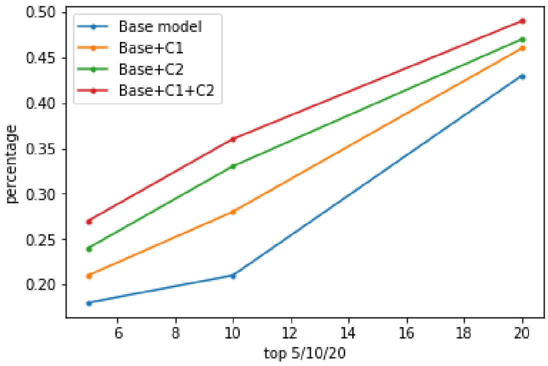
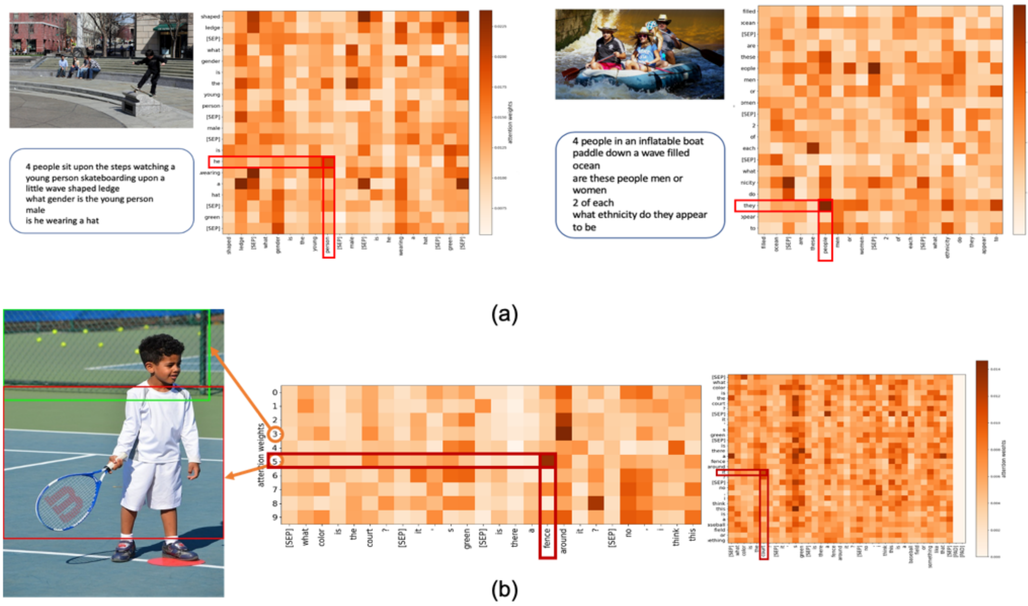


Figure 4: The percentage of the correct antecedent is within top 5/10/20 pronoun’s attention distribution.

To create these subsets, we do not take “it” into consideration, as many of the occurrences of “it”[6] do not have a corefering expression, for example, “is it daytime ?”. We test our models and the baseline model using these four small datasets, and the results are illustrated in Table 2. Comparing the results in Table 2 with that in Table 1, in some cases the performance in Table 2 is better, which means that the difficulty of these sampled data do not higher than that of the test set. One clear observation is that in almost all cases the model with two soft constraints has the best performance in terms of MRR, R@1, R@5, R@10 and Mean in all three datasets. Another observation is that integrating either the POS constraint or the nearest preference constraint improves the performance across all evaluation metrics in all four datasets, which again shows the effectiveness of our proposed linguistically inspired soft constraints. We can also see the trend that the models’ performance is worse when the dialog has more coreferences, which is reasonable and consistent with our previous assumption that the existence of coreferences makes this task more difficult. To further study

---

[6] “It” has many other functions apart from being a pronoun, such as “empty” subject or object, not referring to anything in particular, to introduce or anticipate the subject or object of a sentence, use in cleft or in passive voice sentences, etc.



4 people sit upon the steps watching a young person skateboarding upon a little wave shaped ledge
what gender is the young person
male
is he wearing a hat

4 people in an inflatable boat paddle down a wave filled ocean
are these people men or women
2 of each
what ethnicity do they appear to be

Figure 5: Attention score map of our best model (Model C1+C2). (a) Two correct cases: The attention map shows that the pronoun correctly attends to its antecedent. (b) An incorrect case: Although the coreference is correctly resolved, the word "fence" does not attend to the correct region in the image, which results in selecting the wrong answer.

the effectiveness of our proposed constraints, we manually annotated coreferences in a subset of the validation set and tested all our ablation models on this dataset. The results are presented in Figure 4, which shows that both the proposed constrains can improve the model's ability of resolving coreferences.

5.3 Qualitative Analysis

We visualize the attention scores of the model to investigate whether or not our best model can resolve pronoun coreferences in the dialog. We expect that the pronoun will attend the most to its antecedent, if the model correctly resolves the pronoun referent. Figure 5(a) presents the results of two samples taken from the VisDial v1.0 test set. Note that we only show the attention scores in the window of size 20 around the pronoun and its possible antecedent. The attention score maps illustrate that in these examples the attention weights between pronoun and its antecedent are significantly larger than those between the pronoun and other words, which indicates that the model can implicitly resolve the coreferences correctly.

Error Analysis Figure 5(b) presents a negative example. When facing the question "Is there a fence around it ?", the model answers "no, i think..." instead of the correct answer "yes". As shown in the attention map within language sequences, the pronoun "it" does attend to its antecedent "court", implying that the model successfully resolves the corefering noun. However, looking at the cross-modal attention, the word "fence" incorrectly refers to region 5 (court) in the image, which results in selecting a wrong answer. This negative example indicates that enhancing the model's ability of correct visual grounding is a meaningful future work.

6 Conclusion

In this paper, we have built a multi-layer transformer model for the visual dialog task. Based on linguistic knowledge and human dialog discourse patterns, we have proposed two soft constraints that effectively improve the model's performance by enhancing its ability of implicitly resolving pronoun coreferences. We have used the VisDial v1.0 dataset to evaluate our model. Our model obtains new state-of-the-art performance in correctly answering the dialog questions when compared to existing models without pretraining on other vision-language datasets. Our coreference and qualitative analysis further supports the proposed soft constraints.

Acknowledgement

This research received funding from the Flemish Government (AI Research Program).

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018a. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3674–3683.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433.

Dennis Connolly, John D Burger, and David S Day. 1997. A machine learning approach to anaphoric reference. In *New Methods in Language Processing*, pages 133–144.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2017. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 326–335.

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

Quynh Ngoc Thi Do, Steven Bethard, and Marie Francine Moens. 2015. Adapting coreference resolution for narrative processing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2262–2267.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1971–1982.

Zhe Gan, Yu Cheng, Ahmed El Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. 2019. Multi-step reasoning via recurrent dual attention for visual dialog. *"Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL).*

Dalu Guo, Chang Xu, and Dacheng Tao. 2019. Image-question-answer synergistic network for visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10434–10443.

Dan Guo, Hui Wang, Hanwang Zhang, Zheng-Jun Zha, and Meng Wang. 2020. Iterative context-aware graph inference for visual dialog. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10055–10064.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Xiaoze Jiang, Jing Yu, Zengchang Qin, Yingying Zhuang, Xingxing Zhang, Yue Hu, and Qi Wu. 2020. Dualvd: An adaptive dual encoding model for deep visual understanding in visual dialogue. In *The Association for the Advancement of Artificial Intelligence (AAAI)*, volume 1, page 5.

Mandar Joshi, Omer Levy, Daniel S Weld, and Luke Zettlemoyer. 2019. Bert for coreference resolution: Baselines and analysis. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).*

Gi-Cheon Kang, Jaeseo Lim, and Byoung-Tak Zhang. 2019. Dual attention networks for visual reference resolution in visual dialog. *The 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980.*

Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2018. Visual coreference resolution in visual dialog using neural module networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–169.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *ArXiv Preprint ArXiv:1908.03557.*

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 13–23.

Jiasen Lu, Anitha Kannan, Jianwei Yang, Devi Parikh, and Dhruv Batra. 2017. Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model. In *Advances in Neural Information Processing Systems (NIPS)*, pages 314–324.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Vishvak Murahari, Dhruv Batra, Devi Parikh, and Abhishek Das. 2019. Large-scale pretraining for visual dialog: A simple state-of-the-art baseline. *ArXiv Preprint ArXiv:1912.02379*.

Yulei Niu, Hanwang Zhang, Manli Zhang, Jianhong Zhang, Zhiwu Lu, and Ji-Rong Wen. 2019. Recursive visual attention in visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6679–6688.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NIPS)*, pages 8026–8037.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99.

Idan Schwartz, Seunghak Yu, Tamir Hazan, and Alexander G Schwing. 2019. Factor graph attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2039–2048.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. *The International Conference on Learning Representations (ICLR)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008.

Aparna Nurani Venkitasubramanian, Tinne Tuytelaars, and Marie-Francine Moens. 2017. Entity linking across vision and language. *Multimedia Tools and Applications*, 76(21):22599–22622.

Ellen M Voorhees. 1999. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the gap: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics (TACL)*, 6:605–617.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv Preprint ArXiv:1609.08144*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*, pages 2048–2057.

Tianhao Yang, Zheng-Jun Zha, and Hanwang Zhang. 2019. Making history matter: History-advantage sequence training for visual dialog. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2561–2569.

A Appendix

A 1

This Appendix shows the computation within a transformer encoder layer. Given an input sequence $H_0 = [e_0, e_1, \cdots, e_n]$, the transformer encodes it into different levels of contextual representations using a multi-head self-attention mechanism:

$$Q = H_{t-1}W_t^Q, K = H_{t-1}W_t^K, V = H_{t-1}W_t^V \tag{1}$$

$$AttentionHead_i(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d}})V \tag{2}$$

$$MultiHeadSelfAttention(H^t) = Concat(head_1, \cdots, head_k)W^O \tag{3}$$

where t is the layer index, $W_t^Q, W_t^K, W_t^V, W_t^O$ are learnable projection matrices, which map the hidden state h_t to the query q vector, key k vector, value v vector and output vector, and H_t denote the learned contextual representations at layer t. d is the size of hidden state.

A 2

The below formulations express the derivation of getting equation 8. Here, we use p to denote the pos in equation 11.

$$\begin{aligned} PE_1 \cdot PE_2 &= \frac{1}{k^2} \sum_{i=0}^{\frac{d}{2}-1} sin(w_i p_1) \cdot sin(w_i p_2) \\ &+ cos(w_i p_1) \cdot cos(w_i p_2) \\ &= \frac{1}{k^2} \sum_{i=0}^{\frac{d}{2}-1} cos(w_i(p_1 - p_2)) \\ &= \frac{1}{k^2} \sum_{i=0}^{\frac{d}{2}-1} cos(w_i \Delta p) \end{aligned}$$

A 3

Table 1 in this section shows the results of the four models on the VisDial v1.0 development set. Similar to the results obtained by testing on the test set, the model with both constraints (Model + C1 + C2) has the best performance across all evaluation metrics, and adding any one of the proposed soft constraint improves the performance of the baseline model, which indicates the effectiveness of our approach also during training.

Model	MRR ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mean ↓
Baseline Model	64.80	50.56	82.64	91.02	3.85
Baseline Model + C1	68.59	55.37	84.38	92.29	3.28
Baseline Model + C2	68.32	55.25	84.49	92.25	3.32
Baseline Model + C1 + C2	**69.49**	**56.46**	**85.33**	**93.37**	**3.19**

Table 1: Results of the visual dialog models on the VisDial v1.0 development set. C1 and C2 denote our proposed soft constraints: POS constraint and nearest preference constraint, respectively.

Increasing Robustness to Spurious Correlations using Forgettable Examples

Yadollah Yaghoobzadeh[1] Soroush Mehri[2] Remi Tachet des Combes[2]
Timothy J. Hazen[1] Alessandro Sordoni[2]
[1]Microsoft Turing, Montréal
[2]Microsoft Research, Montréal
{yayaghoo,alsordon}@microsoft.com

Abstract

Neural NLP models tend to rely on spurious correlations between labels and input features to perform their tasks. Minority examples, *i.e.*, examples that contradict the spurious correlations present in the majority of data points, have been shown to increase the out-of-distribution generalization of pre-trained language models. In this paper, we first propose using example forgetting to find minority examples without prior knowledge of the spurious correlations present in the dataset. Forgettable examples are instances either learned and then forgotten during training or never learned. We empirically show how these examples are related to minorities in our training sets. Then, we introduce a new approach to robustify models by fine-tuning our models twice, first on the full training data and second on the minorities only. We obtain substantial improvements in out-of-distribution generalization when applying our approach to the MNLI, QQP, and FEVER datasets.

1 Introduction

Despite the impressive performance of current NLP models, these models often exploit spurious correlations: they tend to capture prediction correlations that hold for most examples but do not hold in general. For instance, in natural language inference (NLI) datasets, word-overlap between hypothesis and premise is highly correlated with the *entailment* label (McCoy et al., 2019; Zhang et al., 2019). Therefore, these models are brittle when tested on examples that cannot be solved by recurring to these correlations, limiting their application in real-world scenarios. *Out-of-distribution* or *challenging* sets are benchmarks carefully designed to break systems that rely on such correlations.

The paradigm of fine-tuning pre-trained language models (PLM) has pushed the state-of-the-art in a large variety of tasks involving natural language understanding (NLU) (Devlin et al., 2019; Wang et al., 2019). This is achieved by self-supervised learning from an enormous amount of text. PLMs also show increased robustness on challenging datasets (Hendrycks et al., 2019). This increase is attributed to an empirical finding that PLMs perform better on *minority examples* present in the training data (Tu et al., 2020). These minority examples violate the spurious correlations and therefore likely support the examples in challenging datasets.

Tu et al. (2020) find minority examples by manually dividing the training data into two groups, according to the known spurious correlations (e.g., word-overlap in NLI). They present an analysis of the robustness of PLMs and its connection to minority examples. In this work, we first introduce a systematic way to find minority examples that does not need prior knowledge of spurious correlations, a big limitation of the earlier work. We then present a simple approach that increases the robustness of PLMs further by tuning models more on these examples.

To identify the set of minority examples, we adopt *example forgetting* (Toneva et al., 2019). This statistic has been shown to relate to the hardness of examples, so we assume it is useful to find minorities in the training data. Based on the definition presented in Toneva et al. (2019), we consider an example *forgettable* if during training it is either properly classified at some point and misclassified later, or if it is never properly classified. This method is model- and task-agnostic. We show in our datasets that minority examples w.r.t to spurious correlations, such as word-overlap in NLI, are well represented in forgettable examples.

After finding minorities through forgettable examples, we propose a simple method to increase the robustness of PLMs further. We perform an additional fine-tuning on the minorities ex-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3319–3332
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

clusively, after fine-tuning on the whole training data. We find this strategy effective, as it increases robust accuracy, i.e., performance on out-of-distribution data, while minimally impacting performance on in-distribution examples. We evaluate our proposed methods in three tasks, including NLI (MNLI, Williams et al., 2017), paraphrase identification (QQP, Iyer et al., 2017) and fact verification (FEVER, Thorne et al., 2018). For each task, recent work has introduced out-of-distribution test sets targetting specific spurious correlations.

Our contributions are the following:

- We propose using forgettable examples as a new approach for finding minority examples from training data without prior knowledge of spurious correlations.
- We show how to exploit minority examples and increase the robustness of deep neural models. This method outperforms other baselines in three challenging datasets: HANS (McCoy et al., 2019), PAWS (Zhang et al., 2019) and FEVER-Symmetric (Schuster et al., 2019). Our method performs effectively when applied to both base and large versions of PLMs (e.g., $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$).
- We observe that finding minorities using a network shallower than the PLM is more effective to robustify it via fine-tuning.
- We show that training models only on forgettable examples leads to poor performance in our datasets, which contrasts with the vision results from Toneva et al. (2019). Our code is available at `github.com/sordonia/hans-forgetting`

2 Datasets

We consider three sentence pair classification tasks, namely natural language inference, paraphrase identification, and fact verification. In the following, we describe the datasets we choose for each task following an introduction of the task.

2.1 Natural Language Inference

The first task we consider is MNLI (Williams et al., 2017), a common natural language inference dataset containing more than 400,000 premise and hypothesis pairs annotated with textual entailment information (*neutral*, *entailment* or *contradiction*). Models trained on this dataset have been shown to capture spurious correlations, such as word-overlap between hypothesis and premise as a strong signal for the *entailment* label (Naik et al., 2018; McCoy et al., 2019). A series of diagnostic out-of-distribution test sets have been devised to test robustness against such heuristics, e.g., HANS.

HANS (McCoy et al., 2019, Heuristic Analysis for NLI Systems) is composed of both *entailment* and *contradiction* examples that have high word-overlap between hypothesis and premise (e.g. "*The president advised the doctor*" $\not\rightarrow$ "*The doctor advised the president*"). A model relying exclusively on the word-overlap feature would not have a higher than chance classification accuracy on HANS. As a matter of fact, BERT (Devlin et al., 2019) performance on this dataset is only slightly better than chance (McCoy et al., 2019). We consider HANS (size: 30k examples) and the MNLI matched dev (Williams et al., 2017) (size: 9815 examples) as our out- and in-distribution test sets for MNLI.

2.2 Paraphrase Identification

QQP (Iyer et al., 2017) is a widely used dataset for paraphrase identification containing over 400,000 pairs of questions annotated as either paraphrase or non-paraphrase. As a consequence of the dataset design, pairs with high lexical overlap have a high probability of being paraphrases. Similarly to MNLI, models trained on QQP are thus prone to learning lexical overlap as a highly informative feature and do not capture the common sense underlying paraphrasing. PAWS dataset is designed to test that.

PAWS (Zhang et al., 2019, Paraphrase Adversaries from Word Scrambling) is a question paraphrase dataset, well-balanced with respect to the lexical overlap heuristic. The accuracy of BERT is around 91.3% on QQP and only 32.2% on PAWS (Table 5). This makes it an interesting test-bed for our method. We use PAWS-QQP as our out-of-distribution set, which contains 677 questions pairs. Training examples from PAWS were never used to update our models. Following Zhang et al. (2019) and Utama et al. (2020), our QQP training and testing splits are based on Wang et al. (2017).

2.3 Fact Verification

The task of fact verification aims to verify a claim given an evidence. The labels are *support*, *refutes*, and *not enough information*. This task is defined as part of the Fact Extraction and Verification (FEVER) challenge (Thorne et al., 2018). Schuster et al. (2019) show that models ignoring evidence can still achieve high accuracy on FEVER. They introduce an evaluation test set that challenges that bias. Following Utama et al. (2020), we use the **FEVER-Symmetric** datasets (Symm-v1 and Symm-v2 with 717 and 712 examples, respectively) for out-of-distribution evaluation[1].

3 Finding Minorities with Forgettables

We first define example forgetting and how to compute it. We then show that it can be used to find minority examples in the training data.

3.1 Forgettable examples

An example is forgotten if it goes from being correctly to incorrectly classified during training (each such occurrence is called a *forgetting event*). This happens due to the stochastic nature of gradient descent, in which gradient updates performed on certain examples can hurt performance on others. If an example is forgotten at least once or is never learned during training it is dubbed *forgettable*. Finding forgettable examples entails training the model on $\mathcal{D}$ and tracking the accuracy of each example at each presentation during training. The algorithm for computing forgettability is cheap (Toneva et al., 2019) and only requires storing the accuracy of each particular example at each epoch.

In Toneva et al. (2019), they extracted forgettable examples from a *shallower network* compared to their target model. This makes finding forgettables more efficient and also results in a more diverse set of examples, as the number of forgettable examples is usually higher for weaker models. Another factor is that the shallow models exhibit less memorization due to their fewer number of hyperparameters (Sagawa et al., 2020b) and therefore their forgettables are potentially more representative of the minorities.

We compute forgettable examples using two models with significantly lower capacity compared to PLMs. The first one is a "siamese" BoW classifier in which hypothesis and premise are independently encoded as a mean of word embeddings. This common model in NLP tasks has surprisingly good performance while relying only on the bag of lexical features. We also consider a siamese BiLSTM model. More details can be found in Appendix A. Finally, for comparison, we also experiment with the model used for HANS in SOTA baselines (Clark et al., 2019; Utama et al., 2020) (see also §4.2), as well as BERT$_{\text{BASE}}$ for in NLI.

Dataset	**Model**	$\lvert\mathcal{F}\rvert$	**Dev Acc.**
MNLI (392,703)	BoW	63,390	64.0
	BiLSTM	46,740	69.6
	BERT	17,748	84.5
QQP (384,348)	BoW	71,116	81.1
	BiLSTM	76,634	84.3
	BERT	20,498	91.3
FEVER (242,911)	BoW	76,368	53.3
	BiLSTM	68,406	56.7
	BERT	21,066	84.4

Table 1: Number of "forgettable" examples along with the accuracy on the MNLI matched, QQP, and FEVER development set. BERT's forgettables are used only in MNLI experiments. The full training size is shown in parenthesis for each dataset.

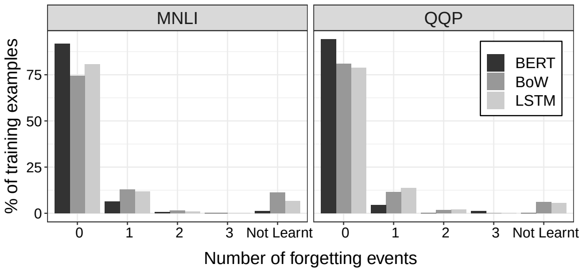


Figure 1: Forgetting events in MNLI and QQP training sets for the three models. A majority of examples are not forgotten during training.

We train the shallow models for five epochs and track forgetting statistics after each epoch. Table 1 shows the number of forgettable examples for BoW, BiLSTM and BERT$_{\text{BASE}}$ on the MNLI, QQP and FEVER training sets. The performance of the models on the dev set of MNLI is also included.

The distribution of forgetting events for each model can be found in Fig 1. We see that a majority of examples are not forgotten. Most of the forgettables are either those that are never learned or have only one forgetting event. In what follows, we denote the sets of forgettable examples from BERT$_{\text{BASE}}$, BiLSTM, and BoW as $\mathcal{F}_{\text{BERT}}$, $\mathcal{F}_{\text{BiLSTM}}$ and $\mathcal{F}_{\text{BoW}}$ respectively.

---

[1] https://github.com/TalSchuster/FeverSymmetric

	MNLI		QQP		FEVER	
	P	$\neg P$	P	$\neg P$	P	$\neg P$
all	**.33**	.22	**.51**	.34	**.19**	.15
$\mathcal{F}_{\text{BoW}}$	.26	**.30**	.48	.49	.18	.17
$\mathcal{F}_{\text{BiLSTM}}$	.25	.28	.48	**.50**	.18	17
$\mathcal{F}_{\text{BERT}}$	.26	.26	.50	.50	**.19**	**.18**

Table 2: Average Jaccard index as a measure of word-overlap between two sentences grouped by P (positive) and $\neg P$ (non-positive).

3.2 Forgettable and minority examples

We focus on two important spurious correlations: word-overlap and contradiction-word. These correlations or biases are addressed in related work for MNLI and QQP (Tu et al., 2020; Zhou and Bansal, 2020). For convenience, we use "positive" for either entailment, supports or paraphrase, and "negative" for contradiction, refutes or non-paraphrase

High word-overlap between two sentences is spuriously correlated to the positive label in all three datasets. In Table 2, we show that on average, positive examples have higher word-overlap compared to non-positive ones. In other words, minorities w.r.t. word-overlap correspond to non-positive examples with high word-overlap and positive examples with low word-overlap. For MNLI and QQP, the distribution in $\mathcal{F}_{\text{BoW}}$ and $\mathcal{F}_{\text{BiLSTM}}$ exhibit an interesting behavior: on average, non-positive examples have higher word-overlap. $\mathcal{F}_{\text{BERT}}$ has the same average for both labels. For FEVER, the difference is also clear as the gap in word-overlap between positive and non-positive examples is lower for forgettables. The table allows to conclude that forgettable examples contain more minority examples than a random subset of the same size.

In Table 3, we perform a similar analysis for the presence of contradiction words in the second sentence, which is shown to correlate with negative class in MNLI (Naik et al., 2018; Zhou and Bansal, 2020) and FEVER (Schuster et al., 2019). We choose these contradiction words: {"not", "no", "doesn't", "don't", "never", "any"}, and analyze all three datasets. We observe here as well that forgettables contain more minority examples, as their percentage of examples with a contradiction word is lower for negative examples, which is the opposite than in the overall dataset (with the exception of $\mathcal{F}_{\text{BERT}}$ and FEVER).

	MNLI		QQP		FEVER	
	N	$\neg N$	N	$\neg N$	N	$\neg N$
all	**31.5**	10.4	**6.2**	4.0	**16.4**	1.2
$\mathcal{F}_{\text{BoW}}$	9.9	11.5	4.1	**4.6**	1.6	**3.1**
$\mathcal{F}_{\text{BiLSTM}}$	11.5	11.2	4.2	4.4	2.9	2.7
$\mathcal{F}_{\text{BERT}}$	14.2	**12.3**	4.2	4.4	6.3	2.5

Table 3: Percentage of examples containing one of the negative keywords in the hypothesis / second question / claim in MNLI / QQP / FEVER. We group examples by binary labels (N: negative and $\neg N$: non-negative) to show the distribution difference between forgettable and overall training examples.

4 Robustifying by Fine-Tuning on Minority Examples

Prior work shows that PLMs generalize to out-of-distribution data because they generalize better on minority examples from the training set (Tu et al., 2020). Here, we introduce a simple approach that exploits minority examples to increase robustness. In this approach, we fine-tune a PLM in two successive phases, first on the full training set, and then on the minority examples only. Our method does not need changes to the training objectives. An illustration is shown in Fig 2.

4.1 PLMs

We are interested in the robustness of large PLMs. In this work, we focus on two such models, BERT and XLNet, and experiment with both their base and large versions. $\text{BERT}_{\text{BASE}}$ being the model of choice in previous work (Clark et al., 2019; Zhang et al., 2019; Utama et al., 2020), it will serve as our default architecture. We adopt the Transformers library (Wolf et al., 2019). Our robust models are obtained by fine-tuning PLMs on the full training set for 3 epochs (using the default hyperparameters for each task) and then on the forgettable examples only, for 3 more epochs with a smaller learning rate. See B in Appendix for more details.

4.2 Baselines

Recently, multiple methods have been proposed to learn more robust models through mitigating biases (Clark et al., 2019; He et al., 2019; Mahabadi and Henderson, 2019; Utama et al., 2020). In these works, PLMs are fine-tuned on a re-weighted version of the source dataset, in which examples are weighted based on their hardness. Hardness is measured by training biased models using prior knowledge of the biases or spurious correlations, e.g., a

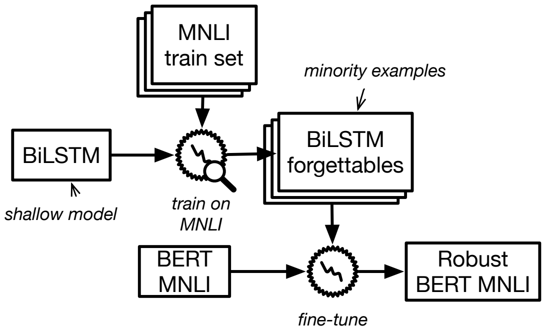


Figure 2: Our proposed framework to train more robust models for an example dataset (MNLI). We first detect minority examples through *forgettable* examples of a shallow model from the training set. We then fine-tune a PLM model (e.g. BERT) in two rounds: first on the full training set, and second on the forgettable subset exclusively. The final model is more robust.

linear model with word-overlap features for NLI. Compared to these works, our method does not need prior knowledge of spurious correlations and exploit the minority examples explicitly by further fine-tuning on them.

We consider the recent *confidence regularization* or "Reg-conf" technique from Utama et al. (2020), as our main baseline in all three tasks. This method is an improvement to the earlier related work in making more robust NLP models (He et al., 2019; Clark et al., 2019). Specifically, Reg-conf claims to maintain the in-distribution performance while improving out-of-distribution and is doing so without introducing new hyperparamters. For MNLI, we report the results of three other baselines of He et al. (2019), Clark et al. (2018) and Mahabadi and Henderson (2019).

Previous work generally design biased models assuming a priori knowledge of the specific dataset biases (with the exception of He et al. (2019), that use a BoW model for HANS). For HANS and PAWS, Clark et al. (2019) and Utama et al. (2020) employ a model with 7 input features, such as word-overlap between premise and hypothesis. To highlight the generality of our approach, we also add this biased model to the set of our shallow models for HANS and fine-tune on its forgettables ($\mathcal{F}_{\text{HANS}}$ with the size of around 200k). For FEVER-symmetric, Utama et al. (2020) consider an LSTM model that takes only the "claim" as input and ignores the "evidence". These baselines re-weight or confidence-regularize training examples using the biased models' performance.

Model	**MNLI**	**HANS**	**Avg.**
BERT	**84.4**$_{\pm0.1}$	62.9$_{\pm1.5}$	73.7$_{\pm0.8}$
BERT+ $\mathcal{F}_{\text{BERT}}$	83.0$_{\pm0.4}$	68.9$_{\pm1.4}$	75.9$_{\pm0.7}$
BERT+ $\mathcal{F}_{\text{BiLSTM}}$	82.9$_{\pm0.4}$	70.4$_{\pm0.9}$	76.7$_{\pm0.5}$
BERT+ $\mathcal{F}_{\text{BoW}}$	83.1$_{\pm0.3}$	**70.5**$_{\pm0.7}$	**76.8**$_{\pm0.4}$
BERT + Rand$_{63,390}$	84.3	63.6	73.9
BERT+ $\mathcal{F}_{\text{HANS}}$	83.9$_{\pm0.4}$	69.5$_{\pm0.9}$	76.7$_{\pm0.5}$
Clark et al. (2019)			
Reweight	83.5	69.2	76.4
Learned Mixin	84.3	64.0	74.2
Mahabadi and Henderson (2019)			
Product of Experts	84.0	66.5	75.3
He et al. (2019)			
DRiFt-HYPO	84.3	67.1	75.7
Utama et al. (2020)			
Reg-conf$_{\text{hans}}$	84.3$_{\pm0.1}$	69.1$_{\pm1.2}$	76.7$_{\pm0.6}$

Table 4: Results of our BERT models fine-tuned on different sources of forgettable examples. For each line, the accuracy on MNLI and HANS are shown, as well as their average.

4.3 Results

4.3.1 MNLI and HANS

In Table 4, we present the results of our models and four recent baselines. The first line reports the performance of BERT on MNLI and HANS. The following lines report the results obtained by fine-tuning BERT on the set of forgettable examples obtained using different shallow models. We also report the average performance between MNLI and HANS. The results confirm that tuning the model towards minority examples improves robustness with a slight drop in MNLI accuracy. Our best model is obtained by fine-tuning on $\mathcal{F}_{\text{BoW}}$, achieving a HANS mean accuracy of 70.5% (with a max of 71.3% over five seeds, which constitutes a +8.4% absolute improvement w.r.t to the initial BERT). To assess whether $\mathcal{F}_{\text{BoW}}$ is indeed responsible for the improvement, we also fine-tune BERT on the same number of randomly chosen examples (BERT + Rand$_{63,390}$), which leads to a negligible improvement.

Fine-tuning on $\mathcal{F}_{\text{BiLSTM}}$ is comparable to fine-tuning on $\mathcal{F}_{\text{BoW}}$, which demonstrates that both BoW and BiLSTM models learn similar spurious correlations. We also added results of fine-tuning BERT on its own forgettables for this task. Note

Model	QQP	PAWS	Avg.
BERT	**90.9**$_{\pm 0.4}$	34.5$_{\pm 1.5}$	62.7$_{\pm 0.6}$
BERT+ $\mathcal{F}_{\text{BoW}}$	89.0$_{\pm 0.9}$	**48.8**$_{\pm 5.2}$	**68.9**$_{\pm 2.2}$
BERT+ $\mathcal{F}_{\text{BiLSTM}}$	88.0$_{\pm 0.8}$	47.6$_{\pm 4.1}$	67.8$_{\pm 1.7}$
Utama et al. (2020)			
BERT	91.0	34.3	62.6
Reg-conf$_{\text{hans}}$	89.1	39.8	64.5

Table 5: Results of BERT$_{\text{BASE}}$ trained on different sets of training examples. Accuracy (%) is reported on the QQP test set (size: 10k) and the PAWS dev set (size: 677), alongside their average.

that while it provides less improvement in robustness than on $\mathcal{F}_{\text{BiLSTM}}$ or $\mathcal{F}_{\text{BoW}}$ [2], it does generate a significant 6.0% increase in performance. Finally, we also report fine-tuning results on $\mathcal{F}_{\text{HANS}}$, the biased model designed for HANS, and observe that it performs well with a smaller loss on MNLI and a smaller gain on HANS compared to $\mathcal{F}_{\text{BoW}}$ and $\mathcal{F}_{\text{BiLSTM}}$.

Compared to other baselines, our approach achieves a comparable or better average accuracy of MNLI and HANS, despite its simplicity. In Fig. 3, we breakdown the results of our best performing model for the three different heuristics HANS was built upon. Our method does not suffer as much as other baselines in the entailment class, and still provides a significant improvement for non-entailment. (More analysis is presented in Appendix.)

4.3.2 QQP and PAWS

Here we report the results of our method applied to QQP and PAWS as out-of-distribution dataset. Results can be found in Table 5. We observe that our method improves out-of-distribution accuracy substantially. It is worth noting that the ground-truth labels in QQP contain noisy annotations (Iyer et al., 2017); a portion of performance loss on QQP could be attributed to that.

Our method outperforms Reg-conf$_{\text{hans}}$, while being simpler in terms of both the biased model and the training regime. We notice that Reg-conf$_{\text{hans}}$ also loses in-distribution performance[3].

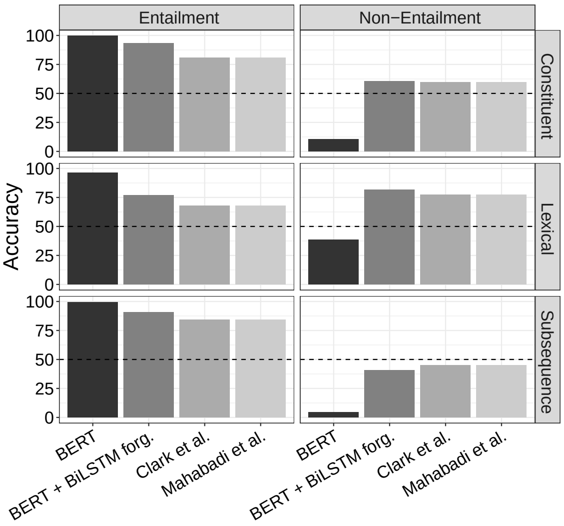


Figure 3: Performance of our BERT fine-tuned on the BiLSTM forgettables $\mathcal{F}_{\text{BiLSTM}}$, and baselines on the "entailment" and "non-entailment" categories for each heuristic HANS was designed to capture.

Model	FEVER	Sym-v1	Sym-v2
BERT	86.1$_{\pm 0.3}$	57.7$_{\pm 1.3}$	64.7$_{\pm 1.1}$
BERT+ $\mathcal{F}_{\text{BoW}}$	**87.1**$_{\pm 0.2}$	61.0$_{\pm 1.4}$	**67.0**$_{\pm 1.5}$
BERT+ $\mathcal{F}_{\text{BiLSTM}}$	86.5$_{\pm 0.4}$	**61.7**$_{\pm 1.2}$	66.6$_{\pm 1.0}$
Utama et al. (2020)			
BERT	85.8$_{\pm 0.1}$	57.9$_{\pm 1.1}$	64.4$_{\pm 0.6}$
Reweighting$_{\text{bigrams}}$	85.5$_{\pm 0.3}$	**61.7**$_{\pm 1.1}$	66.5$_{\pm 1.3}$
Reg-conf$_{\text{claim}}$	86.4$_{\pm 0.2}$	60.5$_{\pm 0.4}$	66.2$_{\pm 0.6}$

Table 6: Accuracy of different FEVER trained models on FEVER dev, and symmetric v1 and v2 datasets.

4.3.3 FEVER

In Table 6, we report the results of our method applied to the FEVER development and symmetric evaluation sets (see §2.3). Our approach again works well for both $\mathcal{F}_{\text{BoW}}$ and $\mathcal{F}_{\text{BiLSTM}}$, but here we also gain on the original dev set when compared to the initial BERT$_{\text{BASE}}$ results. The gains of our method are larger than those of the Reg-conf$_{\text{claim}}$ baseline, which uses a biased model tailored to FEVER-symmetric.

4.4 Analysis

Final loss to detect minority examples An alternative way to find examples from the minority is to simply rank training examples based on their final loss value. In Fig 4, we compare that with our method based on forgettables. The two are

---

[2] To eliminate the forgettables' size factor and focus on the type of model instead, we run an experiment where we sample from $\mathcal{F}_{\text{BoW}}$ the same numbers as $\mathcal{F}_{\text{BERT}}$. The result of our fine-tuning on that smaller $\mathcal{F}_{\text{BoW}}$ was still significantly better than $\mathcal{F}_{\text{BERT}}$.

[3] The authors report accuracy on each label individually and not the overall accuracy. We compute that based on their reported numbers.

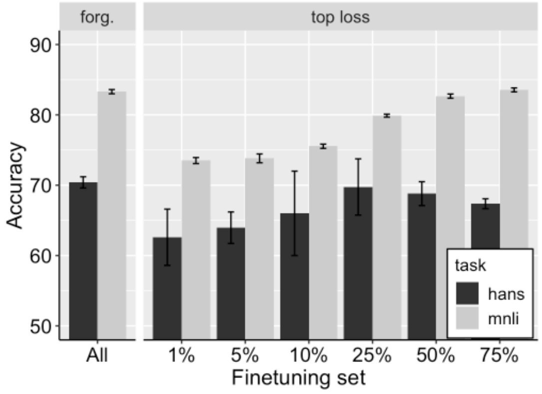


Figure 4: Accuracy on MNLI and HANS when the finetuning set is picked from examples in forgettables or in a range of percentages of examples with highest loss.

Train examples	MNLI	HANS
$\mathcal{F}_{\text{BERT}}$ (17,748)	$49.6_{\pm0.2}$	$37.9_{\pm1.3}$
Random (17,748)	$74.7_{\pm0.4}$	$50.8_{\pm0.2}$
$\mathcal{F}_{\text{BiLSTM}}$ (46,740)	$66.7_{\pm0.9}$	$54.0_{\pm0.6}$
Random (46,740)	$78.8_{\pm0.4}$	$51.3_{\pm0.6}$
$\mathcal{F}_{\text{BoW}}$ (63,390)	$68.2_{\pm0.8}$	$55.4_{\pm1.4}$
Random (63,390)	$79.9_{\pm0.4}$	$51.8_{\pm0.2}$
All	$84.5_{\pm0.1}$	$63.1_{\pm1.2}$

Table 7: Results of $\text{BERT}_{\text{BASE}}$ models fine-tuned on the set of forgettable examples only.

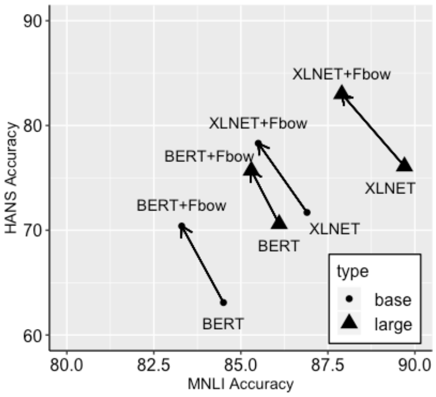


Figure 5: MNLI vs HANS accuracy for both base and large versions of BERT and XLNET.

Model	QQP	PAWS
$\text{BERT}_{\text{BASE}}$	90.9	34.5
$\text{BERT}_{\text{BASE}} + \mathcal{F}_{\text{BoW}}$	89.0	48.8
$\text{BERT}_{\text{LARGE}}$	**91.2**	36.0
$\text{BERT}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$	88.3	54.4
$\text{XLNET}_{\text{BASE}}$	90.9	37.1
$\text{XLNET}_{\text{BASE}} + \mathcal{F}_{\text{BoW}}$	88.2	55.2
$\text{XLNET}_{\text{LARGE}}$	89.2	48.4
$\text{XLNET}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$	87.8	**65.2**

Table 8: Average accuracy across random seeds on QQP and PAWS for BERT and XLNET base and large models, before and after fine-tuning on $\mathcal{F}_{\text{BoW}}$.

obviously related, as the examples that are never learned rank the highest w.r.t to the loss and are considered as forgettables. However, Fig 4 shows, for MNLI and HANS, that using forgettables produces better performance both in- and out-of-distribution. One additional issue with using the final loss to pick examples is the need to determine either a threshold value α on the loss (keep examples with a loss larger than α) or a number N of examples to retain. The optimal α or N might yield better performance but finding them implies using the out-of-distribution set.

Robustness of larger models We examine the performance of our method when applied to other PLMs and to larger networks by training BERT large and XLNET. Fig 5 shows the MNLI and HANS performance of those networks. Firstly, XLNet is noticeably more robust than BERT, compatible with its superior in-distribution performance (Yang et al., 2019). Secondly, we observe that the large versions generalize on HANS significantly better than their base counterparts (e.g., 76.1% vs 71.7% for XLNet, 70.6% vs 62.9% for BERT), confirming that larger models seem more robust. Lastly, $\text{XLNET}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$ shows a +7% increase in performance, reaching 83.1% on HANS with a maximum score of 86.8% over three seeds. We also show Table 8 and Table 9) similar findings on QQP and FEVER. For instance, $\text{XLNET}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$ achieves 65.2% on PAWS and 75.3% on FEVER-Sym-v1.

Training on forgettables only Toneva et al. (2019) showed that forgettable examples form the support of the training distribution. We follow their experimental setting and fine-tune BERT on the subset of forgettable examples only (i.e., without any fine-tuning on the whole dataset). Contrary to what was found in Toneva et al. (2019), we observe in Table 7 that the performance obtained by training only on forgettable examples is poor compared to random subsets of the same sizes; MNLI accuracy is only 37.9% for $\mathcal{F}_{\text{BERT}}$ compared to 74.7% for a random subset with the same size. The HANS accuracy is also poor. These results sug-

Model	**FEVER**	**Symm-v1**
$\text{BERT}_{\text{BASE}}$	86.1	57.7
$\text{BERT}_{\text{BASE}} + \mathcal{F}_{\text{BoW}}$	87.2	61.0
$\text{BERT}_{\text{LARGE}}$	86.9	59.7
$\text{BERT}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$	86.5	67.8
$\text{XLNET}_{\text{BASE}}$	86.4	63.9
$\text{XLNET}_{\text{BASE}} + \mathcal{F}_{\text{BoW}}$	87.5	67.8
$\text{XLNET}_{\text{LARGE}}$	88.2	68.8
$\text{XLNET}_{\text{LARGE}} + \mathcal{F}_{\text{BoW}}$	**88.7**	**75.3**

Table 9: Average accuracy over seeds on FEVER and Fever-symm-v1 for BERT and XLNET base and large models, before and after fine-tuning on $\mathcal{F}_{\text{BoW}}$

gest an intrinsic difficulty in $\mathcal{F}_{\text{BERT}}$ that makes it hard for $\text{BERT}_{\text{BASE}}$ to generalize from it. However, as we showed previously, when starting from an already trained model, forgettables increase the out-of-distribution performance.

Calibration of models We look into the confidence of entailment when $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{BASE}} + \mathcal{F}_{\text{BoW}}$ trained on MNLI are applied to HANS. In Fig 6, we show that $\text{BERT}_{\text{BASE}}$ can discriminate HANS entailments from non-entailments but with a very large classification threshold. Fine-tuning on forgettables recalibrates the classification threshold on HANS and makes 0.5 as the optimum value.

Other diagnostic evaluations Fine-tuning on the forgettable examples of simple biased models improves robustness in the three challenging benchmarks HANS, FEVER-Symmetric and PAWS. We additionally evaluate the trained models listed in Table 4 on Stress tests (Naik et al., 2018), adversarial NLI (Nie et al., 2019) and MNLI-matched-hard (Gururangan et al., 2018). For these test sets, we do not observe improvements when evaluating the robust model using $\mathcal{F}_{\text{BoW}}$. We posit that specific biased models might be needed in some of these cases. As a validation, for MNLI-matched-hard, we design a BiLSTM model that only takes the hypothesis as input, and apply our method using the forgettables of that model to fine-tune $\text{BERT}_{\text{BASE}}$. We observe an increase in performance from 76.5% to 78.0% (averaged across five seeds). These results suggest that the forgettable examples of simple biased models like BoW or BiLSTM capture the more informative heuristics like word-overlap well. However, for less informative heuristics like hypothesis-only features, a heuristic-designed biased model is a better choice since its forgettables likely violate the specific heuristic.

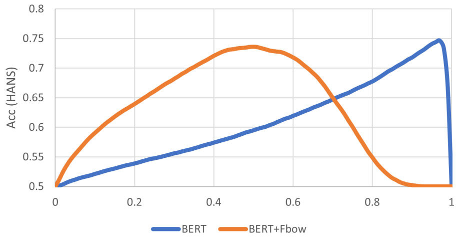


Figure 6: HANS accuracy vs classification threshold used to predict entailment/non-entailment. The base BERT model is overconfident in the entailment class while after fine-tuning on forgettables, we can improve model calibration.

5 Related Work

A growing body of literature recently focused on **out-of-distribution generalization**, showing that it is far from being attained, even in seemingly simple cases (Geirhos et al., 2019; Jia and Liang, 2017; Dasgupta et al., 2018). In particular, and in contrast with what Mitchell et al. (2018) recommend, NLP models do not seem to “embody the symmetries that allow the same meaning be expressed within multiple grammatical structures”. Supervised models seem to exhibit poor systematic generalization capabilities (Loula et al., 2018; Lake and Baroni, 2018; Baan et al., 2019; Hupkes et al., 2018) thus seemingly lacking *compositional* behavior (Montague, 1970). While this might seem at odds with the common belief that high-level semantic representations of the input data are formed (Bengio et al., 2009b), the reliance on highly predictive but brittle features is not confined to NLU tasks. It is also a perceived shortcoming of image classification models (Geirhos et al., 2019; Brendel and Bethge, 2019). To test systematically if machine learning models generalize beyond their training distribution, several challenging datasets have been introduced in NLP and other ML applications (Kalpathy-Cramer et al., 2015; Peng et al., 2019; Clark et al., 2019). Those test sets are made automatically from designed grammars (McCoy et al., 2019) and/or by human annotators (Zhang et al., 2019; Schuster et al., 2019).

Dataset re-sampling and weighting These techniques have been studied in order to solve class imbalance problem (Chawla et al., 2002) or covariate shift (Sugiyama et al., 2007), notably by importance weighted empirical risk minimization. In NLP, Clark et al. (2019); Mahabadi and Henderson (2019); He et al. (2019); Utama et al. (2020) give evidence of the effectiveness of re-weighting training examples to increase robustness. They generally assume *a priori* knowledge of the heuristics present in the dataset and up/down-weight examples concerning those heuristics. AFLITE (Sakaguchi et al., 2019) is an algorithmic method for bias removal in datasets without relying on prior knowledge about datasets. It filters out examples with a high average predictability score, relating them to points with biases or spurious correlations. Bras et al. (2020) adopt AFLITE to build more robust models. This method harms the in-distribution performance significantly. Comparatively, we aim to increase robustness while maintaining in-distribution performance, so we do not filter out easy examples in our approach. Our work also relates to distributionally robust optimization (Duchi and Namkoong, 2018; Hu et al., 2018) and the more recent group-DRO (Sagawa et al., 2020a), which does not assume access to target data and optimizes the worst-case performance under an unknown, bounded distribution shift. Recently, Swayamdipta et al. (2020) introduced a two-dimensional criterion to identify hard and easy examples. They consider both the confidence of an example (the average of its loss during training epochs) and the variability (those for which the loss has high variance) and show that ambiguous examples (high-variance and high-confidence) can enhance OOD accuracy. Easy examples (low-variance and high-confidence) can instead help model optimization. Although example forgetting is a coarser measure of variance of the loss, their results align with our findings: up-weighting hard/ambiguous examples enhance OOD generalization, but only training on those can harm optimization.

Curriculum Learning Dataset sampling is related to curriculum learning, where training proceeds along with a curriculum of samples with increasing difficulty (Bengio et al., 2009a). Kumar et al. (2010); Zhao and Zhang; Fan et al. (2017); Katharopoulos and Fleuret (2018); Kim and Choi (2018); Jiang et al. (2018) have shown the concept can be quite successful in a variety of areas. Our robustifying method is related to this concept. However, our models are first trained using i.i.d samples from the whole dataset and then fine-tuned on more difficult cases, i.e., the minorities.

Spurious correlations in NLU datasets like MNLI or FEVER are the subjects of many works. They include (i) the presence of specific words in the hypothesis or claim, for example, negation words like “not” are correlated with the contradiction label in entailment tasks (Naik et al., 2018; Gururangan et al., 2018), or bigrams like “did not” with the refute label, in fact, verification (Schuster et al., 2019); (ii) syntactic heuristics, like word-overlap between premise and hypothesis; and (iii) sentence length (Gururangan et al., 2018), and its correlation with labels. HANS (McCoy et al., 2019) and PAWS (Zhang et al., 2019) (which we evaluate on) generate plausible high word-overlap examples for both positive and negative classes. Glockner et al. (2018) build a new test example by simple lexical inference rules and show the brittleness of models on this out-of-distribution dataset. They also show that having supporting examples in training data is key to predict a test example correctly.

Feldman and Zhang (2020) show that when datasets are long-tailed, rare and atypical instances make up a significant fraction of the data distribution and **memorizing** them leads to better in-domain generalization. They find those rare and atypical examples using influence estimation. We instead study forgettable examples and their impact on out-of-distribution generalization. An interesting experiment would be to mine minority examples by influence estimation and compare with forgettable examples.

6 Conclusion

We introduced a novel approach, based on example forgetting, to extract minority examples and build more robust models systematically. Via example forgetting, we built a set of minority examples on which a pre-trained model is fine-tuned. We evaluated our method on large-scale models such as BERT and XLNet and showed a consistent improvement in robustness on three challenging test sets. We also showed that the larger versions obtain higher out-of-distribution performance than the base ones but still benefit from our method.

Acknowledgement

We thank Rabeeh Karimi Mahabadi, Yoshua Bengio and all the anonymous reviewers for their insightful comments.

References

Joris Baan, Jana Leible, Mitja Nikolaus, David Rau, Dennis Ulmer, Tim Baumgärtner, Dieuwke Hupkes, and Elia Bruni. 2019. On the realization of compositionality in neural networks. *arXiv preprint arXiv:1906.01634*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009a. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Yoshua Bengio et al. 2009b. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew E Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. *arXiv preprint arXiv:2002.04108*.

Wieland Brendel and Matthias Bethge. 2019. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1914–1925.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel Gershman, and Noah D. Goodman. 2018. Evaluating compositionality in sentence embeddings. In *Proceedings of the 40th Annual Meeting of the Cognitive Science Society, CogSci 2018, Madison, WI, USA, July 25-28, 2018*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

John C. Duchi and Hongseok Namkoong. 2018. Learning models with uniform performance via distributionally robust optimization. *CoRR*, abs/1810.08750.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint arXiv:1702.08635*.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. 2019. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *The 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142.

Dan Hendrycks, Kimin Lee, and Mantas Mazeika. 2019. Using pre-training can improve model robustness and uncertainty. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2712–2721. PMLR.

Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. 2018. Does distributionally robust supervised learning give robust classifiers? In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2034–2042.

Dieuwke Hupkes, Anand Singh, Kris Korrel, German Kruszewski, and Elia Bruni. 2018. Learning compositionally through attentive guidance. *arXiv preprint arXiv:1805.09657*.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. First quora dataset release: Question pairs. *data. quora. com.*

R. Jia and P. Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Empirical Methods in Natural Language Processing (EMNLP).*

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the 35th International Conference on Machine Learning.*

Jayashree Kalpathy-Cramer, Alba Garcia Seco de Herrera, Dina Demner-Fushman, Sameer K. Antani, Steven Bedrick, and Henning Mller. 2015. Evaluating performance of biomedical image retrieval systems - an overview of the medical image retrieval task at imageclef 2004-2013. *Comp. Med. Imag. and Graph.*, 39:55–61.

Angelos Katharopoulos and Franois Fleuret. 2018. Not all samples are created equal: Deep learning with importance sampling. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80, pages 2530–2539.

Tae-Hoon Kim and Jonghyun Choi. 2018. Screenernet: Learning curriculum for neural networks. *CoRR*, abs/1801.00904.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-Paced Learning for Latent Variable Models. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 1–9.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888.

João Loula, Marco Baroni, and Brenden M. Lake. 2018. Rearranging the familiar: Testing compositional generalization in recurrent networks. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 108–114.

Rabeeh Karimi Mahabadi and James Henderson. 2019. Simple but effective techniques to reduce biases. *CoRR*, abs/1909.06321.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.*

Jeff Mitchell, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Extrapolation in nlp. *arXiv preprint arXiv:1805.06648.*

Richard Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics.*

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599.*

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment matching for multi-source domain adaptation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1406–1415.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020a. Distributionally robust neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.*

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020b. An investigation of why overparameterization exacerbates spurious correlations. *arXiv preprint arXiv:2005.04345.*

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WINOGRANDE: an adversarial winograd schema challenge at scale. *CoRR*, abs/1907.10641.

Tal Schuster, Darsh J Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. *arXiv preprint arXiv:1908.05267.*

Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. 2007. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 9275–9293.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 809–819.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2019. An empirical study of example forgetting during deep neural network learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *CoRR*, abs/2007.06778.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Mind the trade-off: Debiasing nlu models without degrading the in-distribution performance. In *In ACL*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 3261–3275.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764.

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

Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37, pages 1–9.

Xiang Zhou and Mohit Bansal. 2020. Towards robustifying NLI models against lexical dataset biases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8759–8771. Association for Computational Linguistics.

A Details of biased models (BoW and BiLSTM)

Both models are Siamese networks, with similar input representations and classification layers. For the input layer, we lower case and tokenize the inputs into words and initialize their representations with Glove, a 300-dimensional pretrained embedding (Pennington et al., 2014). For the classification task, from the premise and hypothesis vectors p and h, we build the concatenated vector $s = [p, h, |p - h|, p \odot h]$ and pass it to a 2-layer feedforward network. To compute p or h, the BoW model max-pools the bag of word embeddings, while the BiLSTM model max-pools the top-layer hidden states of a 2-layer bidirectional LSTM. The hidden size of the LSTMs is set to 200. Overall, BoW and BiLSTM contain 560K and 2M parameters, respectively.

B Hyperparameters and training time

We use a learning rate of 5e-5 for MNLI and QQP when training the PLMs on the full training and the learning rate of 1e-5 when fine-tuning on forgettables. For FEVER, we use 2e-5 and 5e-6 for the full training and the fine-tuning on forgettables, respectively.

With a 4x Tesla P100 GPU machine and batch-size 256 per GPU, one epoch of training on the full train set takes around 4-6 minutes for BOW and BiLSTM models in all of the three training tasks.

For $\text{BERT}_{\text{BASE}}$, with batch-size 32 per GPU, one epoch of training on the full train set takes around 30 / 20 / 30 minutes (per task). The maximum input length after tokenization is set to 128 in all the experiments.

B.1 Forgettables and word-overlap in MNLI

Model	**Entailment**			**Non-Entailment**		
	All	*High*	*Low*	*All*	*High*	*Low*
BERT	**84.0**	**89.9**	**76.0**	84.9	85.5	84.6
BERT + $\mathcal{F}_{\text{BoW}}$	80.2	85.1	73.4	**85.6**	86.9	**85.0**
BERT + $\mathcal{F}_{\text{BiLSTM}}$	79.9	85.2	72.4	**85.6**	**87.4**	84.8

Table 10: Fine-grained accuracy results of $\text{BERT}_{\text{BASE}}$ on the MNLI dev set split by word-overlap between hypothesis and premise.

In Table 10, we show the performance of our method on the MNLI dev set as a function of word-overlap, the main heuristic HANS was designed against. We split the evaluation set into High (> mean) and Low (< mean) word-overlap examples, where word-overlap is measured using the Jaccard Index between hypothesis and premise. We see in particular that entailment pairs with high word-overlap suffer from the fine-tuning on forgettables, while non-entailment improves (we observe a similar trend for QQP; see App. C). This supports the observations in 3.2 that the initial model relied on the spurious correlation of word-overlap and entailment to classify pairs and that by fine-tuning on forgettable examples, the performance on minorities increased.

C Forgettables and word-overlap in QQP

In Table 11, we show the performance of our method on the QQ evaluation set as a function of word-overlap, the main heuristic PAWS was designed against. We see in particular that paraphrase pairs with high word-overlap suffered from the fine-tuning, while non-paraphrase improved. This supports the intuition that the initial model relied on word-overlap to classify pairs as paraphrase, while forgettables help mitigate that phenomenon to some extent.

	Paraphrase			**Non-Paraphrase**		
Model	*All*	*High*	*Low*	*All*	*High*	*Low*
BERT	**90.0**	**90.8**	**88.9**	92.2	85.6	95.0
BERT + $\mathcal{F}_{\text{BiLSTM}}$	85.2	84.9	85.8	**93.0**	**87.3**	**95.4**
BERT + $\mathcal{F}_{\text{BoW}}$	87.3	87.2	87.4	92.6	86.4	95.2

Table 11: Fine-grained accuracy results of BERT on QQP development set before and after fine-tuning on forgettables. We split the evaluation set into High (> mean) and Low (< mean) word-overlap examples, where word-overlap is measured under the Jaccard Index between two sentences. Similar observations hold true in the case of MNLI.

On Robustness of Neural Semantic Parsers

Shuo Huang[1]**, Zhuang Li**[1]**, Lizhen Qu**[1‡]**, Lei Pan**[2]
Faculty of Information and Technology, Monash University[1]
School of Info Technology, Deakin University[2]
`shua0043@student.monash.edu`
`Zhuang.Li@monash.edu`
`Lizhen.Qu@monash.edu`
`l.pan@deakin.edu.au`

Abstract

Semantic parsing maps natural language (NL) utterances into logical forms (LFs), which underpins many advanced NLP problems. Semantic parsers gain performance boosts with deep neural networks, but inherit vulnerabilities against adversarial examples. In this paper, we provide the empirical study on the robustness of semantic parsers in the presence of adversarial attacks. Formally, adversaries of semantic parsing are considered to be the perturbed utterance-LF pairs, whose utterances have exactly the same meanings as the original ones. A scalable methodology is proposed to construct robustness test sets based on existing benchmark corpora. Our results answered five research questions in measuring the sate-of-the-art parsers' performance on robustness test sets, and evaluating the effect of data augmentation.

1 Introduction

Semantic parsing aims to map natural language (NL) utterances into logical forms (LFs), which can be executed on a knowledge base (KB) to yield denotations (Kamath and Das, 2018). At the core of the state-of-the-art (SOTA) semantic parsers are deep learning models, which are widely known to be vulnerable to *adversarial samples* (Zhang et al., 2020). This kind of examples is created by adding tiny perturbations to inputs but can severely deteriorate model performance. To the best of our knowledge, despite the popularity of semantic parsing (Kamath and Das, 2018), there is still no published work on studying the robustness of neural semantic parsers against adversarial examples. Therefore, we conduct the *first* empirical study to evaluate the effect of adversarial examples on SOTA neural semantic parsers.

Unlike other disciplines, it is unclear what adversaries are for semantic parsers. For computer vision systems, adversaries are often generated by modifying inputs with *imperceptible* perturbations. In contrast, a flip of single word or a character in an utterance can significantly change its meaning so that the changes are perceptible by humans. To address this issue, (Michel et al., 2019) argue that adversaries for sequence-to-sequence models should maximally retain meanings after perturbing inputs. However, any meaning-changing utterances are supposed to have different meaning representations. A robust semantic parser should be invariant to meaning-preserving modifications. In light of this, given a semantic parser, we define its adversaries as the perturbed utterances satisfying two conditions: i) they have *exactly* the same meanings as the original ones according to human judgements; and ii) the parser consistently produces incorrect LFs on them.

Although new evaluation frameworks are proposed for NLP tasks (Xu et al., 2020; Michel et al., 2019), there is no framework designed for assessing robustness of semantic parsers against meaning-preserving adversaries. The current evaluation metrics focus only on standard accuracy, which measures to what degree predictions match gold standards. As pointed out by (Tsipras et al., 2018), it is challenging to achieve both high standard accuracy and high robust accuracy, which measures the accuracy on adversarially perturbed examples. In order to facilitate devising novel methods for robust semantic parsers, it is desirable to develop a semantic parsing evaluation framework considering both measures.

In this work, we propose an evaluation framework for robust semantic parsing. The framework consists of an evaluation corpus and a set of customized metrics. We construct the evaluation corpus by extending three existing semantic parsing

---

[‡] corresponding author

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3333–3342
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

benchmark corpora. In order to generate meaning-preserving examples, we apply automatic methods to modify the utterances in those benchmark corpora by paraphrasing and injecting grammatical errors. Among the perturbed examples generated from the test sets, we build meaning-preserving test sets by filtering out the meaning-changing ones using crowdsourcing. The robustness of the semantic parsers is measured by a set of custom metrics, with and without adversarial training methods.

We conduct the *first* empirical study on the robustness of semantic parsing by evaluating three SOTA neural semantic parsers using the proposed framework. The key findings from our experiments are three-folds:

- None of those SOTA semantic parsers can consistently outperform the others in terms of robustness against meaning-preserving adversarial examples;
- Those neural semantic parsers are more robust to word-level perturbations than sentence-level ones;
- Adversarial training through data augmentation indeed significantly improve the robustness accuracy but can only slightly influence standard accuracy.

The generated corpus and source code are available at https://github.com/shuo956/On-Robustness-of-nerual-smentic-parsing.git

2 Related Work

Semantic Parsing The SOTA neural semantic parsers formulate this task as a machine translation problem. They extend SEQ2SEQ with attention (Luong et al., 2015) to map NL utterances into LFs in target languages (e.g., lambda calculus, SQL, Python, etc.). One type of such parsers directly generates sequences of predicates as LFs (Dong and Lapata, 2016, 2018; Huang et al., 2018). The other type of parsers utilizes grammar rules to constrain the search space of LFs during decoding (Yin and Neubig, 2018; Chen et al., 2018; Guo et al., 2019b; Wang et al., 2020). However, neither of the two types of parsers are evaluated against adversarial examples.

Adversarial Examples The adversarial examples are firstly defined and investigated in computer vision. Adversarial examples in that field are generated by adding imperceptible noise to input images, which lead to false predictions of machine learning models (Madry et al., 2018; Goodfellow et al., 2014). However, it is non-trivial to add such noise to text in natural language processing (NLP) tasks due to the discrete nature of languages. Minor changes in characters or words may be perceptible to humans and may lead to change of meanings. To date, it is still difficult to reach an agreement on the definition of adversarial examples across tasks.

Jia and Liang (2017); Belinkov and Bisk (2017); Ebrahimi et al. (2018); Miyato et al. (2016) add distracting sentences and sequences of random words into text or flip randomly characters or words in input text, which can confuse the models but do not affect the labels judged by humans. In those works, such perturbations are not required to keep the semantics of original text. (Michel et al., 2019) argue that perturbations for SEQ2SEQ tasks should minimize the change of semantic in input text but dramatically alter the meaning of outputs. (Cheng et al., 2020) uses a sentiment classifier to verify whether sentiments of the original utterances is preserved after perturbation while we use crowdsourcing to ensure the meaning remains. Adversarial examples in semantic parsing cannot simply borrow from prior work because a parser does not make errors if it generates a different and correct LF when there is any subtle change in input text leading to change of semantics. In contrast, adversarial examples are supposed to cause parsing errors. Thus, adversarial examples w.r.t. meaning-changing perturbations are not well defined.

There are two types of methods in generating adversarial examples. The white-box methods (Papernot et al., 2016; Ebrahimi et al., 2017; Athalye and Carlini, 2018) assume that the attacker have direct access to model details including their parameters, but the black-box methods assume that attackers have no access to model details except feeding input data and getting outputs from models (Gao et al., 2018; Guo et al., 2019a; Chan et al., 2018; Blohm et al., 2018).

Adversarial Training Adversarial training aims at improving the robustness of machine learning models against adversarial examples (Goodfellow et al., 2014; Miyato et al., 2016; Li et al., 2018). One line of research is to augment the training data with the adversarial examples. However, Ebrahimi et al. (2018) points out that adversarial training may cause the model oversensitive to the adversarial examples. The other approach is to increase model capacity that may improve model's robustness (Madry et al., 2018). More techniques regarding adversarial defense can be found in the recent survey (Wang et al., 2019). In semantic parsing, data augmentation methods (Jia and Liang, 2016; Guo et al., 2018) are proposed only to improve performance of models on examples without perturbation but not to improve their robustness.

3 Evaluation Framework for Robust Semantic Parsing

A robust semantic parser aims to transduce all utterances with or without meaning-preserving perturbations into correct LFs. Formally, let $\mathbf{x} = x_1, \dots, x_{|\mathbf{x}|}$ denote a natural language utterance, and $\mathbf{y} = y_1, \dots, y_{|\mathbf{y}|}$ be its LF, a semantic parser estimates the conditional probability (denoted by $p(\mathbf{y}|\mathbf{x})$) of an LF $\mathbf{y}$ given an input utterance $\mathbf{x}$. A robust parser's predictions are invariant to all $\mathbf{x}'$ that are generated from $\mathbf{x}$ by meaning-preserving perturbations.

For any $(\mathbf{x}, \mathbf{y})$, let $S_{\text{perturb}}(\mathbf{x}, \mathbf{y})$ denote the set of *all* meaning-preserving perturbations of $\mathbf{x}$ that is parsed to $\mathbf{y}$:

$$S_{\text{perturb}}(\mathbf{x}, \mathbf{y}) = \{(\hat{\mathbf{x}}, \mathbf{y}) : \hat{\mathbf{x}} \in B(\mathbf{x}) \land o(\hat{\mathbf{x}}) = \mathbf{y}\} \tag{1}$$

where $B(\mathbf{x})$ denotes the set of *all* allowed perturbations of $\mathbf{x}$, and $o(\hat{\mathbf{x}})$ is an ideal parser that maps an utterance to its LF.

A set of meaning-preserving perburbed examples $S_{\text{perturb}}(D)$ w.r.t. a corpus D is the union of all $S_{\text{perturb}}(\mathbf{x}, \mathbf{y})$ created from D.

An adversarial example w.r.t. a semantic parser is an utterance-LF pair, which is in $S_{\text{perturb}}(\mathbf{x}, \mathbf{y})$ and is parsed into an incorrect LF by that parser. A set of adversarial examples w.r.t. a $S_{\text{perturb}}(\mathbf{x}, \mathbf{y})$ and a parser $f(\mathbf{x})$ is obtained by:

$$S_{\text{adv}}(\mathbf{x}, \mathbf{y}) = \{(\hat{\mathbf{x}}, \mathbf{y}) : f(\hat{\mathbf{x}}) \neq \mathbf{y}, \forall (\hat{\mathbf{x}}, \mathbf{y}) \in S_{\text{perturb}}(\mathbf{x}, \mathbf{y})\} \tag{2}$$

Subsequently, an adversary set w.r.t. a semantic parsing corpus D is created by taking the union of all adversary sets created from each example in D.

In the following, we present an evaluation framework for robust semantic parsing, which consists of an evaluation corpus, a set of evaluation metrics, and the corresponding toolkit for evaluating any new parsers.

3.1 Construction of the Evaluation Corpus

We construct the evaluation corpus in a scalable manner by combining the existing semantic parsing benchmark corpora. Each of such corpora will be referred to as a *domain* in the whole corpus. There are a train set, a validation set, and a standard test set in each domain. We perturb the examples in each test set to build a *meaning-preserving test set* for each domain. More specifically, each example in a meaning-preserving test set is a perturbed utterance paired with its LF before perturbation. In the following, we detail each perturbation method and how we apply the crowdsourcing to remove meaning-changing ones.

3.1.1 Meaning-Preserving Perturbations

Perturbations	**Meaning-preserving Examples**
Insertion	in what state is the largest in population
Deletion	what state is ~~the~~ largest in population
Substitution (f)	what state is the largest among population
Substitution (nf)	what state is the most in population
Back Translation	state with the largest population
Reordering	the largest population is in what state

Table 1: The meaning-preserving examples of the original utterance "what state is the largest in population".

Given an utterance, we perturb it by performing four different word-level operations and two sentence-level operations, *respectively*. Table 1 lists the examples of generated meaning-preserving examples categorized according to the respective generation methods. More details are given below.

Insertion Given an utterance $\mathbf{x}$, we randomly select a word position $t' \in \{1, \dots, |\mathbf{x}|\}$. A meaning-preserving example $\mathbf{x}'$ is created by inserting a function word at position t'.

Deletion We randomly remove a function word x_t from $\mathbf{x}$.

Substitution (f) Every function word in $\mathbf{x}$ is replaced with a random but different function word to generate a perturbed example.

Substitution (nf) For every non-function word in $\mathbf{x}$, we apply the pretrained language model ELECTRA (Clark et al., 2020) to select top-k candidate words and exclude the original word. We generate a perturbed utterance for each valid candidate word. Since this method may generate utterances far from their original meaning, we subsequently filter those utterances by measuring their semantic similarity with the original ones.

Specifically, we apply the SOTA sentence similarity model Sentence-Bert (Reimers and Gurevych, 2019) to compute similarity scores for each generated utterance resulting in only the n highest scored utterances.

Back Translation Inspired by (Lichtarge et al., 2019), we revise utterances using back translation. We apply the Google translation API to translate utterances into a bridge language and then translate them back to English. Russian, French, and Japanese are used as the bridge languages to diversify and maximize the coverage of meaning-preserving perturbations. We select the best translation among all three translations for each original utterance according to Sentence-Bert's scores.

Reordering Similar to back translation, we reorder utterances by the SOTA reordering model SOW-REAP (Goyal and Durrett, 2020). To increase the coverage, we follow the same strategy as *Substitution (nf)* to generate an extended set of reordered utterances with multiple instances per input utterance. We use Sentence-Bert to encode sentences and select the top-k best ones according to their cosine similarity scores with the original input. The reordered sentences share the same vocabulary as the original one except for the order of words.

3.1.2 Filter Examples by Crowdsourcing

As perturbation operations related to function words rarely alter meanings, we apply three crowdsourcing operations to the utterances perturbed, including *Substitution (nf)*, *Reordering*, and *Back Translation*. For each perturbed utterance paired with the original utterance, three turkers at Amazon Mechanical Turk discern any semantic changes by choosing an option out of three — *the same*, *different*, or *not sure*. By default, any sentences uncomprehended by a human are regarded as *not sure*. Finally, we keep only the ones that have the same meaning agreed by at least two turkers. After crowdsourcing, we keep 83%, 82% and 61% of the generated utterances for *Substitution (nf)*, *Back Translation* and *Reordering*, respectively.

3.2 Evaluation Metrics

Our framework assesses the performance of a semantic parser w.r.t. standard accuracy and robustness metrics. Those robustness metrics indicate how well a semantic parser resists to meaning-preserving perturbations and adversarial attacks. As adversarial training is widely used for adversarial defense and mostly applicable to any neural semantic parsers, this framework supports comparing a wide range of adversarial training methods w.r.t. standard accuracy and robustness metrics.

A training set, a validation set, a standard test set, and a meaning-preserving test set are established in each domain. The first three sets are obtained from the original benchmark corpus, and the meaning-preserving test set is created using the methods described in Sec. 3.1. We will examine the meaning-preserving test set (denoted by $S_{\text{perturb}}(D)$) and its two subsets. We refer to the first subset as the robustness evaluation set (denoted by $R_{\text{eval}}(D)$), where the counterparts before perturbation are parsed correctly. We refer to the second subset as the black-box test set (denoted by $B_{\text{attack}}(D)$), where the loss of a parser to the examples is higher than their counterparts before perturbation. For each target parser, we consider four metrics, including *standard accuracy*, *perturbation accuracy*, *robust accuracy*, and *success rate of black-box attack*.

Standard accuracy The most widely used metric on semantic parsing (Dong and Lapata, 2018; Yin and Neubig, 2018) to measure the percentage of the predicted LFs that exactly match their gold LFs in a standard test set.

Perturbation accuracy Perturbation accuracy is formally defined as $n/|S_{\text{perturb}}(D)|$, where n denotes the number of correctly parsed examples to their gold LFs in a meaning-preserving test set $S_{\text{perturb}}(D)$.

Robust accuracy Robust accuracy is calculated as $n/|R_{\text{eval}}(D)|$, where n denotes the number of examples that are parsed correctly by a parser in a robustness evaluation set $R_{\text{eval}}(D)$. Compared to perturbation accuracy, robust accuracy measures the number of examples that a parser can parse correctly before perturbation but fails to get them right after perturbation.

Success rate of black-box attack A black-box attack example is regarded as the one that increases the loss of a model after perturbation (Zhang et al., 2020). Here, the success rate of black-box attack is calculated as $n/|B_{\text{attack}}(D)|$, where n denotes the number of examples that are parsed *incorrectly* by a parser in the black-box test set $B_{\text{attack}}(D)$. White-box attacks require model specific implementation to generate adversarial examples, thus we leave the corresponding evaluation to the developers of semantic parsers.

The four metrics are computed to evaluate the efficacy of an adversarial training method. We inspect whether the metrics increase or decrease post-training and to what degree. An effective adversarial training method is expected to find a good trade-off between standard accuracy and robust accuracy.

Last but not least, all evaluation metrics are implemented with easy-to-use APIs in our toolkit. Our toolkit supports easy evaluation of a semantic parser and provides source code to facilitate integrating additional semantic parsing corpora.

4 Experiments

In this section, we present the first empirical study on robust semantic parsing.

4.1 Experimental Setup

Parsers We consider three SOTA neural semantic parsers — SEQ2SEQ with attention (Luong et al., 2015), COARSE2FINE (Dong and Lapata, 2018), and TRANX (Yin and Neubig, 2018). COARSE2FINE is the best performing semantic parser on the standard splits of GEOQUERY and ATIS. TRANX reports standard accuracy on par with COARSE2FINE and employs grammar rules to ensure validity of outputs.

Datasets Our evaluation corpus is constructed by extending three benchmark corpora — GEOQUERY (Zelle and Mooney, 1996), ATIS (Dahl et al., 1994), and JOBS. GEOQUERY contains 600 and 280 utterance-LF pairs to express the geography information in the training and test set, respectively. ATIS consists of 4434, 491, and 448 examples about flight booking in the training, validation, and test sets, respectively. And JOBS includes 500 and 140 pairs about job listing in the training and test set, respectively.

Adversarial Training Methods We apply three adversarial training methods to the parsers. We evaluate whether the three adversarial training methods could improve semantic parsers' robustness against the meaning-preserving examples generated by the word-level and sentence-level operations. The corresponding three adversarial training methods are as follows:

> **Fast Gradient Method (Miyato et al., 2016)** Fast Gradient Method (FGM) adds small perturbations to the word embeddings and train the semantic parsers with the perturbed embeddings. The perturbations are scaled gradients w.r.t. the input word embeddings.
>
> **Projected Gradient Descent (Madry et al., 2017)** Projected Gradient Descent (PGD) adds small perturbations to the word embeddings as well. Instead of calculating a single step of gradients, PGD accumulates the scaled gradients for multiple iterations to generate the perturbations.
>
> **Meaning-preserving Data Augmentation** Meaning-preserving Data Augmentation (MDA) augments the original training data with the meaning-preserving examples generated by the word-level and sentence-level operations. We randomly select 20% of the original instances for each dataset and generate their corresponding meaning-preserving instances. Since there are six different operations, each original training set is augmented with six different meaning-preserving sets independently.

Training Details For both supervised training and adversarial training, we set batch size to 20 for all parsers with 100 epochs. We follow the best settings of three parsers reported in (Dong and Lapata, 2018; Yin and Neubig, 2018) to set the remaining hyperparameters. The best performing implementation of SEQ2SEQ is included in Yin and Neubig (2018).

4.2 Results and Analysis

We discuss experimental results by addressing the following research questions:

RQ1: How do the SOTA parsers perform on meaning-preserving test sets? All three SOTA semantic parsers are trained on each training set before they are tested on the corresponding standard test sets and meaning-preserving test sets. Besides the overall results on whole test sets, Table 2 reports accuracy on each example subset perturbed by the respective perturbation operation. The results on those subsets are further compared

Generation Methods	Jobs			GeoQuery			Atis		
	Seq2Seq	Coarse2Fine	TranX	Seq2Seq	Coarse2Fine	TranX	Seq2Seq	Coarse2Fine	TranX
Insertion	87.09/84.98	84.40/77.77	81.89/82.97	85.59/80.50	87.57/85.21	85.96/81.73	67.36/68.33	84.16/82.73	77.45/75.63
Substitution (f)	86.58/78.89	84.22/71.59	82.23/86.10	85.27/78.38	86.97/82.30	84.23/79.04	70.93/59.19	86.06/68.22	76.70/67.69
Deletion	86.48/82.23	84.16/81.46	83.80/75.79	85.31/82.30	85.69/72.56	85.69/72.56	67.82/63.06	83.14/73.23	79.65/60.83
Substitution (nf)	80.34/78.63	79.05/76.92	82.90/83.33	85.71/64.95	87.37/70.93	86.37/66.11	75.28/56.36	85.57/68.94	78.55/60.57
Back Translation	87.66/44.15	84.41/46.10	87.01/58.44	85.92/48.74	86.93/56.28	85.42/41.13	75.91/56.38	86.55/63.30	81.23/58.36
Reordering	89.53/39.53	84.88/48.83	90.69/67.44	90.07/29.78	90.78/48.93	93.61/41.13	72.28/40.66	87.04/52.10	79.51/47.89
Overall(micro)	86.15/76.08	83.64/72.35	82.70/78.81	85.48/72.55	87.15/78.04	86.01/72.44	70.72/60.55	85.13/71.79	78.52/64.89
Overall(macro)	86.27/68.06	83.52/67.11	84.75/75.67	86.31/64.10	87.55/69.36	86.88/63.61	71.59/57.33	85.42/68.08	78.84/61.82

Table 2: Standard/Perturbation accuracy of SOTA parsers, trained on the original training sets.

Generation Methods	Jobs			GeoQuery			Atis		
	Seq2Seq	Coarse2Fine	TranX	Seq2Seq	Coarse2Fine	TranX	Seq2Seq	Coarse2Fine	TranX
Insertion	96.06	91.51	94.96	82.32	96.45	95.37	93.42	95.04	93.91
Substitution (f)	90.66	84.77	90.51	87.96	94.51	83.41	78.02	77.05	75.22
Deletion	94.64	96.33	97.18	88.03	93.67	92.04	88.04	86.06	8 7.03
Substitution (nf)	97.87	97.29	97.93	78.71	80.79	76.64	76.50	79.88	76.56
Back Translation	48.88	54.07	64.92	61.07	64.16	57.22	73.06	79.88	76.56
Reordering	42.85	57.53	73.06	38.21	53.90	44.96	53.75	58.82	57.63
Overall(micro)	87.79	86.03	92.21	83.08	89.06	85.05	81.29	82.27	83.06
Overall(macro)	78.50	80.08	86.43	72.31	80.58	74.94	77.71	78.12	76.82

Table 3: Robust accuracy for three SOTA parsers in each domain.

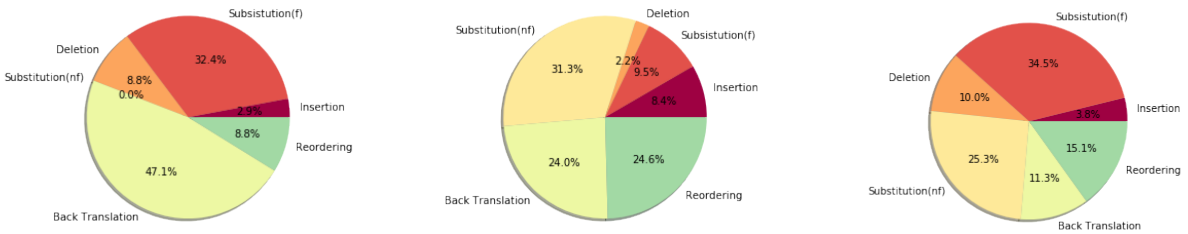
with the standard accuracy on the corresponding examples before perturbation.

As shown in Table 2, SOTA semantic parsers suffer from significant performance drop in almost all meaning-preserving test sets compared to the results on standard test sets. The performance ranking among the three SOTA semantic parsers varies across different datasets. Coarse2Fine achieves the best performance on GeoQuery and Atis, while Seq2Seq beats Coarse2Fine and TranX on Jobs. Although Coarse2Fine achieves better accuracy than TranX on the standard test set of Jobs, it falls short of TranX on the meaning-preserving test set of Jobs. A parser achieving higher standard accuracy does not necessarily obtain better perturbation accuracy against meaning-preserving perturbations than its competitors.

Our evaluation framework supports also in-depth analysis on the impact of different perturbation method on semantic parsers. All parsers are more vulnerable to sentence-level perturbations than word-level ones. Although reordering changes only word order in utterances, it leads to the lowest perturbation accuracy of all parsers on GeoQuery and Atis. Among word-level perturbation operations, substitution of non-function words is more challenging than deletion and insertion of function words on GeoQuery and Atis. On Jobs and Atis, even deletion or substitution of function words can impose significant challenges for the parsers. When we further investigate the deleted or replaced function words, such as in Table 1, semantic parsers are not expected to rely on such information. As all perburbations in our meaning-preserving test sets do not change meanings of original utterances, it imposes new research challenges on how to make semantic parsers resist meaning-preserving perturbations as well as how to avoid overfitting on semantically insignificant words.

RQ2: What kind of perturbed examples particularly degrade parser performance? Although perturbation accuracy allows comparing parsers on the same test sets, it includes the examples that a parser fails to parse correctly both before and after the meaning-preserving perturbation. Robust accuracy focuses on the examples a parser parses successfully before perturbation but fails after that. We investigate all parsers trained without adversarial training in terms of this measure. As shown in Table 3, TranX is superior to the other two parsers on Jobs, and Coarse2Fine is the clear winner on GeoQuery. This ranking of parsers is consistent with perturbation accuracy in the two domains. However, the differences among three parsers on Atis are marginal, while Coarse2Fine achieves significantly superior perturbation accuracy in the same domain. Atis is the domain with the most diverse paraphrases in natural language, Coarse2Fine cannot significantly outperform the other two parsers against meaning-preserving perturbations.

We further investigate adversary examples, which are defined in Eq. 2, for each parser in the meaning-preserving test sets. The shared adversarial examples among the parsers vary significantly across domains. More than 50% of the ad-



(a) JOBS (b) GEOQUERY (c) ATIS

Figure 1: Contributions of perturbation operations on the shared adversaries.

versarial examples are shared among the parsers on ATIS, but only less than 25% of adversaries are shared on JOBS. Fig. 1 illustrates which perturbation operation contributes to the intersection of the adversary set from different parsers. Back translation contributes almost half of the shared perturbed examples on JOBS, while the proportion of word-level perturbation operations reaches nearly 70% on ATIS. Although reordering and back translation impose the most difficult challenges for parsers, they cannot generate a significant number of valid adversarial examples. In contrast, it is relatively easy to resist a word substitution attack, but this operation can be easily applied to generate a large number of adversaries.

Insertion	
Utterance Adv. Utterance	show me the nonstop flight and fare from toronto to st. petersburg show me the nonstop flight and fare from from toronto to st. petersburg
Ground Truth	(lambda $0 e (and (flight $0) (nonstop $0) (from $0 toronto:ci) (to $0 st_petersburg:ci)))
SEQ2SEQ COARSE2FINE TRANX	(lambda $0 e (exists $1 (and (flight $0) (nonstop $0) (from $0 toronto:ci) (to $0 st_petersburg:ci) (= (fare $0) $1)))) (lambda $0 (exists $1 (and (flight $0) (nonstop $0) (from $0 toronto:ci) (to $0 st_petersburg:ci) (= (fare $0) $1)))) (lambda $0 e (exists $1 (and (flight $0) (nonstop $0) (from $0 toronto:ci) (to $0 st_petersburg:ci) (= (fare $0) $1))))
Substitution(f)	
Utterance Adv. Utterance	what is the capital of the state with the largest population density what are the capital of the state with the largest population density
Ground Truth	(capital:c (argmax $1 (state:t $1) (density:i $1)))
SEQ2SEQ COARSE2FINE TRANX	(lambda $0 e (and (capital:t $0) (loc:t $0 (argmax $1 (state:t $1) (density:i $1)))) (density:i $0)) (argmax $0 (state:t $0) (density:i $0)) (lambda $0 e (and (capital:t $0) (loc:t $0 (argmax $1 (state:t $1) (density:i $1)))))
Delection(f)	
Utterance Adv. Utterance	what is the capital of the state with the largest population density what is the capital ~~of~~ the state with the largest population density
Ground Truth	(capital:c (argmax $1 (state:t $1) (density:i $1)))
SEQ2SEQ COARSE2FINE TRANX	(argmin $0 (capital:t $0) (density:i $0)) (argmax $0 (capital:t $0) (population:i $0)) (argmax $0 (capital:t $0) (density:i $0))
Substitution(nf)	
Utterance Adv. Utterance	what are the major river in ohio what are the main river in ohio
Ground Truth	(lambda $0 e (and (major:t $0) (river:t $0) (loc:t $0 ohio)))
SEQ2SEQ COARSE2FINE TRANX	(lambda $0 e (and (river:t $0) (loc:t $0 ohio))) (lambda $0 (and (river:t $0) (loc:t $0 ohio))) (lambda $0 e (and (river:t $0) (loc:t $0 ohio)))
Back Translation	
Utterance Adv. Utterance	list job using sql what kind of work do you have with sql
Ground Truth	job (ANS) , language (ANS , ’sql’)
SEQ2SEQ COARSE2FINE TRANX	job (ANS) , application (ANS , ’sql’) job (ANS) , language (ANS ’sql’) , language (ANS ’sql’) job (ANS) , language (ANS ’sql’) , language (ANS ’sql’)
Reordering	
Utterance Adv. Utterance	what is the lowest point in mississippi in s0 , the lowest point in mississippi
Ground Truth	(lambda $0 e (loc:t (argmin $1 (and (place:t $1) (loc:t $1 mississippi:s)) (elevation:i $1)) $0))
SEQ2SEQ COARSE2FINE TRANX	(argmin $0 (and (place:t $0) (loc:t $0 mississippi:s)) (elevation:i $0)) (lambda $0 (loc:t mississippi:s $0) (loc:t (argmin $1 (and (place:t $1) (loc:t $1 mississippi:s)) (elevation:i $1)) $0)) (argmin $0 (and (place:t $0) (loc:t $0 mississippi:s)) (elevation:i $0))

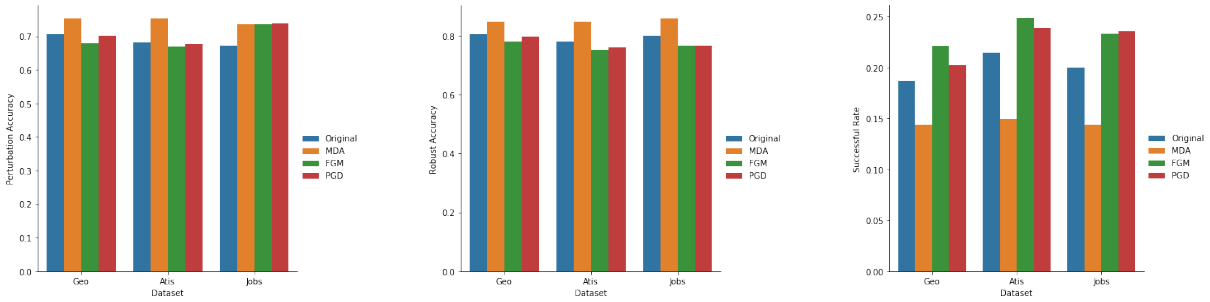
Table 4: Adversarial examples shared by all parsers.

We hand picked representative adversarial examples shared by all parsers and list them in Table 4. The errors made by substitution (f) and insertion often show a sign of overfitting due to the fact that the parsers learn dependencies between non-essential features and predicates. Substitution (nf) sometimes causes predicates to be missing after replacing a word with its synonym. The errors caused by deletion, back translation, and reordering are more diverse, including adding wrong predicates or missing predicates.

RQ3: How effective are those widely used adversarial training methods for semantic parsers? We applied FGM, PGD, and MDA to train the three parsers on the respective training sets. In the case of MDA, we consider all automatic perturbation operations to generate training examples, and each operation adds the same amount of training examples to the original train sets. Fig. 2 illustrates the results of COARSE2FINE w.r.t. different evaluation metrics. MDA always improves the three evaluation metrics, despite that the generated training examples are not checked by humans, resulting in a significant number of meaning-changing examples. FGM and PGD hardly work except on JOBS in terms of perturbation accuracy. Since both methods are white-box methods, the results indicate the gradient-based adversaries cannot effectively capture the meaning-preserving perturbations.

The biggest improvement made by MDA is the success rate of a black-box attack. The black-box attack examples are selected from the meaning-preserving examples that increase a parser's loss after perturbation. When we inspect the examples in the meaning-preserving test sets generated by different perturbation operation, the biggest improvement is from the ones perturbed by sentence-level operations and word substitution. All three parsers benefit from MDA significantly. The more vulnerable to black-box attack a parser is, the more improvement it can achieve after applying MDA.

RQ4: How does each adversarial training method influence standard accuracy? Tsipras et al. (2018) point out that there is a trade-off be-



(a) Perturbation accuracy.

(b) Robust accuracy.

(c) Success rate of black-box attack.

Figure 2: Robustness metrics of COARSE2FINE before and after adversarial training.

Generation Methods	JOBS			GEOQUERY			ATIS		
	SEQ2SEQ	COARSE2FINE	TRANX	SEQ2SEQ	COARSE2FINE	TRANX	SEQ2SEQ	COARSE2FINE	TRANX
No adv. train	87.14	83.57	87.14	83.92	86.78	86.07	73.88	86.16	79.91
FGM	87.31	86.12	91.34	83.89	86.35	84.01	81.00	79.81	81.72
PGD	82.84	86.18	92.93	84.56	86.35	84.01	79.96	79.94	81.52
MDA(all)	87.74	86.16	88.57	82.89	86.73	87.11	74.75	80.18	80.60
MDA(Insertion)	89.14	86.16	88.85	81.78	87.14	87.64	67.16	85.27	78.75
MDA(Substitution (f))	87.71	85.25	88.85	83.21	86.79	87.35	76.51	78.08	85.67
MDA(Deletion)	87.28	90.14	85.33	83.21	86.79	87.35	76.87	85.62	78.52
MDA(Substitution (nf))	87	80.68	90.14	83.85	86.79	86.85	76.11	62.58	85.58
MDA(Back Translation)	88.42	87.53	88.57	82.14	86.71	86.64	76.20	84.73	77.36
MDA(Reordering)	86.91	87.21	89.71	83.21	86.21	86.85	75.66	84.82	77.72

Table 5: Standard accuracy of adversarial training, which is trained with adversarial training methods and evaluated on the standard test sets. MDA(all) stands for MDA with all six perturbation operations.

tween standard accuracy and robust accuracy in most occasions. This finding is attributed to the presence of *non-robust* features, which are highly predictive but incomprehensible for humans (Ilyas et al., 2019). To verify the theory, after applying each adversarial training method to each parser, we compare the standard accuracy before and after adversarial training.

As shown in Table 5, none of the three adversarial training methods consistently improve standard accuracy across different domains and parsers. TRANX does not have a significant performance drop regardless which adversarial training method is applied. It may be due to the fact that TRANX uses grammar to filter out invalid outputs.

MDA cannot consistently improve parsers but also does not hurt parsers' performance in terms of standard accuracy. We conducted t-tests on the standard test sets to assess if MDA with different perturbation operations significantly improves accuracy. The results are negative so that the training examples generated by MDA at least do not hurt parsers' performance while increasing their robustness.

RQ5: How does our meaning-preserving data augmentation method compare with the data augmentation method proposed by (Jia and Liang, 2016)? (Jia and Liang, 2016) is one of the SOTA data augmentation methods for semantic parsing. In their work, they show that the augmented examples improve accuracy of predicting denotations. Although there are three methods proposed in (Jia and Liang, 2016), only the method of *concatenating* two random examples as a new example can be applied in our case. We evaluated this augmentation method with all three parsers. No significant improvement of standard accuracy and robust accuracy is found in all three domains. We conjecture that this is due to the fact that in (Jia and Liang, 2016) they report only improvement of accuracy of denotation matching, not the matching of LFs. In contrast, MDA can effectively reduce the harm of meaning-preserving perturbations.

4.3 Conclusion

We conduct the empirical study on robustness of neural semantic parsers. In order to evaluate robustness accuracy, we define first what are adversarial examples for semantic parsing, followed by constructing test sets to measure robustness using a scalable method. The outcome of this work is supposed to facilitate semantic parsing research by providing a benchmark for evaluating both standard accuracy and robustness metrics.

References

Anish Athalye and Nicholas Carlini. 2018. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*.

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and natural noise both break neural machine translation. *CoRR*, abs/1711.02173.

Matthias Blohm, Glorianna Jagfeld, Ekta Sood, Xiang Yu, and Ngoc Thang Vu. 2018. Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension. *CoRR*, abs/1808.08744.

Alvin Chan, Lei Ma, Felix Juefei-Xu, Xiaofei Xie, Yang Liu, and Yew-Soon Ong. 2018. Metamorphic relation based adversarial attacks on differentiable neural computer. *CoRR*, abs/1809.02444.

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. *arXiv preprint arXiv:1809.00773*.

Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, and Cho-Jui Hsieh. 2020. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. In *AAAI*, pages 3601–3608.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, page 43–48, USA. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. *CoRR*, abs/1806.09030.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. Hotflip: White-box adversarial examples for NLP. *CoRR*, abs/1712.06751.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. *CoRR*, abs/1801.04354.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. *arXiv preprint arXiv:2005.02013*.

Chuan Guo, Jacob R Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Q Weinberger. 2019a. Simple black-box adversarial attacks. *arXiv preprint arXiv:1905.07121*.

Daya Guo, Yibo Sun, Duyu Tang, Nan Duan, Jian Yin, Hong Chi, James Cao, Peng Chen, and Ming Zhou. 2018. Question generation from sql queries improves neural semantic parsing. *arXiv preprint arXiv:1808.06304*.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019b. Towards complex text-to-sql in cross-domain database with intermediate representation. *CoRR*, abs/1905.08205.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. *arXiv preprint arXiv:1803.02400*.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *CoRR*, abs/1707.07328.

Aishwarya Kamath and Rajarshi Das. 2018. A survey on semantic parsing. *CoRR*, abs/1812.00978.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Generating reasonable and diversified story ending using sequence to sequence model with adversarial training. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1033–1043, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. *arXiv preprint arXiv:1904.05780*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

A. Madry, Aleksandar Makelov, L. Schmidt, D. Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Paul Michel, Xian Li, Graham Neubig, and Juan Miguel Pino. 2019. On evaluation of adversarial perturbations for sequence-to-sequence models. *arXiv preprint arXiv:1903.06620*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Nicolas Papernot, Patrick D. McDaniel, Ananthram Swami, and Richard E. Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. *CoRR*, abs/1604.08275.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2018. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Wenqi Wang, Benxiao Tang, Run Wang, Lina Wang, and Aoshuang Ye. 2019. A survey on adversarial attacks and defenses in text. *CoRR*, abs/1902.07285.

Ying Xu, Xu Zhong, Antonio Jose Jimeno Yepes, and Jey Han Lau. 2020. Elephant in the room: An evaluation framework for assessing adversarial examples in nlp. *arXiv preprint arXiv:2001.07820*.

Pengcheng Yin and Graham Neubig. 2018. Tranx: A transition-based neural abstract syntax parser for semantic parsing and code generation.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, page 1050–1055. AAAI Press.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transaction on Intelligent System and Technology*, 11(3).

Benchmarking a transformer-FREE model for ad-hoc retrieval

Tiago Almeida
IEETA
Universidade de Aveiro
Aveiro, Portugal
`tiagomeloalmeida@ua.pt`

Sérgio Matos
DETI/IEETA
Universidade de Aveiro
Aveiro, Portugal
`aleixomatos@ua.pt`

Abstract

Transformer-based "behemoths" have grown in popularity, as well as structurally, shattering multiple NLP benchmarks along the way. However, their real-world usability remains a question. In this work, we empirically assess the feasibility of applying transformer-based models in real-world ad-hoc retrieval applications by comparison to a "greener and more sustainable" alternative, comprising only 620 trainable parameters. We present an analysis of their efficacy and efficiency and show that considering limited computational resources, the lighter model running on the CPU achieves a 3 to 20 times speedup in training and 7 to 47 times in inference while maintaining a comparable retrieval performance. Code to reproduce the efficiency experiments is available on `https://github.com/bioinformatics-ua/EACL2021-reproducibility/`.

1 Introduction

The Natural Language Processing (NLP) field has been revolutionised by the simple, yet extremely successful idea of transfer learning allied to the transformer neural architecture capability of exploring long sequences, while being computationally more efficient than the recurrent counterparts. More precisely, Dai and Le (2015) and later Howard and Ruder (2018) introduced the transfer learning approach to the NLP field by pre-training a language model and then fine-tuning it to multiple NLP tasks. However, it was Radford (2018) who applied the same technique to a large transformer encoder, shattering the SOTA in 9 of 12 NLP tasks.

Since then, the NLP field became dominated by works that use, explore or improve, in some way, the transformer-based architecture. Although their performance is undeniable, these are large models that comprise millions of parameters and require large computational resources to use and maintain, making them inoperable for the majority of smaller institutions. Furthermore, it also gives the impression that the NLP field was reduced to throwing a lot of computation power (money) to continually achieve SOTA results in multiple benchmarks, leaving behind the careful process of designing a neural solution that follows the human intuition to solve some specific task. Moreover, given their running costs and the consequent impact on the environment, their real-world virtues remain questionable, i.e., "is it really feasible to use these models in real-world applications, such as search engines or question answering systems to aid a broader range of people in their day-to-day tasks?". As an example, biomedical experts need to routinely search an unprecedented amount of scientific literature to keep updated with their research, which could be facilitated by an intelligent system.

In contradiction with this trend, this work examines a lightweight interaction-based model, with only 620 trainable parameters, carefully designed by considering years of research in ad-hoc retrieval systems and interaction-based architectures. This system was evaluated in two ad-hoc retrieval competitions where it was able to compete with the state-of-the-art transformer-based models. In this paper, we demonstrate a "greener and more sustainable" alternative to the transformer-based architecture, by thoroughly testing such a system against the most popular transformer behemoths in a series of performance experiments.

Following this section, we frame our work in the current literature, we then present the lightweight model, and finally detail and discuss the evaluation performed.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3343–3353
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2 Related Work

The Information Retrieval (IR) field is nowadays considered as being divided into traditional IR (Baeza-Yates and Ribeiro-Neto, 2011) and neural IR (Mitra and Craswell, 2018). The former explores exact match signals, i.e., the co-occurrence of query terms in the document, to derive handcrafted rules and formulas to directly compute the query-document importance, with the BM25 (Robertson and Zaragoza, 2009) ranking function being the most popular example. On the other end, neural IR explores the increasing success of neural networks to approximate a (sub)optimal ranking function by exploiting labelled examples. In the literature, the most successful neural IR architectures are Interaction Based, which measure multiple matching signals to create a joint representation of the query and the documents. Moreover, considering that the transformer architecture is now widely adopted in interaction-based architectures, we further propose to subdivide the works into shallow interaction-based and transformer interaction-based models.

2.1 Shallow Interaction-Based Models

In this subsection, we address some relevant interaction-based neural models proposed before the transformer revolution, and briefly present their intuition to tackle the ad-hoc retrieval challenge. Guo et al. (2016) proposed the DRMM model, which was one of the first neural models to achieve improvements over strong traditional IR solutions, and showed the importance of individually weighing each query term's contributions. Likewise, Pang et al. (2016); Hui et al. (2017); Dai et al. (2018) showed that 2D convolutions directly applied over the interaction matrix are capable of extracting strong hierarchical n-gram matching patterns. Pang et al. (2017) built an end-to-end neural solution, inspired by the human process of selecting relevant documents. Allied to the semantic matching signals, Fan et al. (2018) added, in parallel, exact matching signals and discussed ways of combining them.

2.2 Transformer Interaction-Based Models

Despite representing a recent trend, there is already an extensive literature on transformer architectures. Here we briefly introduce this architecture and its most adopted variants.

Models that follow a transformer-based architecture are composed of a fixed set of stacked transformer blocks (Vaswani et al., 2017), hence their name, and learn useful word representations from large text corpora. Since the first appearance (Radford, 2018), multiple other variants emerged, arguably the most notable and widely adopted being BERT (Devlin et al., 2019), which uses bidirectional self-attention and requires being trained as a masked language model. However, this model has some limitations. One of them is its large size, which is addressed by distillation models, such as distilBERT (Sanh et al., 2020), or by factorisation tricks to reduce the number of trainable parameters, as in Lan et al. (2020). Another limitation is their short input length (512 tokens), which derives from the quadratic complexity and associated memory requirements imposed by the self-attention mechanism. Beltagy et al. (2020); Wang et al. (2020b); Katharopoulos et al. (2020); Zaheer et al. (2020) address this issue by reducing this complexity order, hence increasing the input length.

Regarding the information retrieval field, some works tried to apply these models to the ad-hoc retrieval task with varying approaches. Yang et al. (2019) proposed to perform "best" relevance sentence inference and then linearly interpolate with the original document score (BM25). Dai and Callan (2019b) discuss different strategies to approximate the document relevance by aggregating the relevance across sentences. Nogueira et al. (2019) proposed a three-stage retrieval pipeline and introduced monoBERT and duoBERT, two BERT models aimed at addressing pointwise and pairwise ranking problems, respectively. Finally, MacAvaney et al. (2019) proposed to use the context-aware embedding produced by these models as the input embedding to the well studied shallow interaction-based models. Furthermore, and also acknowledged by MacAvaney et al. (2019), these models are still not suitable to be used or implemented as real-world solutions given their slow inference times and expensive costs.

Some notable works address the issue of computational costs by introducing precomputation, i.e., compute and store for later reuse. Dai and Callan (2019a) propose to precompute term weights that are then used by efficient traditional retrieval systems such as BM25. MacAvaney et al. (2020), on the other hand, precompute the term representation of the document at index time to be later merged with the query representation at query time.

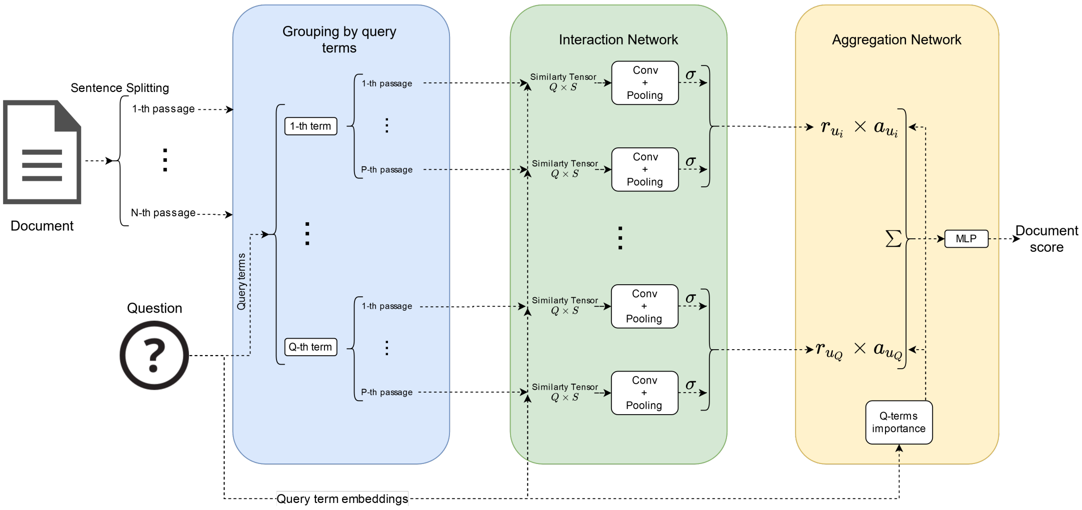


Figure 1: Lightweight neural model data and operation flow.

3 Proposed neural interaction based model

Our neural interaction-based model is an enhancement of Almeida and Matos (2020a) and was already used in the two international competitions, namely BioASQ (Almeida and Matos, 2020b; Tsatsaronis et al., 2015) and TREC-Covid (Almeida and Matos, 2020c; Roberts et al., 2020). However, in order to keep this paper self-contained, we will now introduce its insight and architecture.

From a general perspective, our model follows the successful ideas presented over years of research in the neural IR field, taking into its implementation key concepts of previous works. More precisely, it follows the DeepRank (Pang et al., 2017) intuition to mimic the human judgment process with neural networks; adopts the extraction capabilities, based on 2D convolution, of the PACRR (Hui et al., 2017) and MatchPyramid (Pang et al., 2016) models; explores the same assumption of independently scoring each sentence as addressed by McDonald et al. (2018); and finally, uses the DRMM (Guo et al., 2016) query term gating mechanism to consider the importance of different query terms. Overall, the model was designed with the intuition of weighting the importance of the document sentences in terms of the query by considering the context where the exact match occurs, i.e., this model produces a more refined judgment of the initial exact match signal evaluated by the BM25.

Regarding the architecture, the neural model is divided into three major blocks that can be observed in Figure 1. To clearly describe the model, let us first define a **query** as a sequence of tokens $q = \{u_0, u_1, ..., u_Q\}$, where u_i is the i-th token of the query and Q the size of the query; a **document passage** as $p = \{v_0, v_1, ..., v_S\}$, where v_j is the j-th token of the passage and S the size of the passage; and a document as a sequence of passages $D = \{p_0, p_1, ..., p_N\}$, where N is the total number of passages in the document. In the current implementation, a passage corresponds to a document sentence, as described below.

The first block aims to rearrange the input, query and document, to a suitable format for the downstream interaction model. More precisely, it splits the document into passages using the Punkt algorithm (Kiss and Strunk, 2006) and rearranges the passages by query-term, so that each query-term is associated with a set of passages where that term occurs. At this point, the document is represented by $D(u_i) = \{p_{i0}, p_{i1}, ..., p_{iP}\}$, where p_{ik} corresponds to the k-th passage with respect to the query term u_i, and P is the maximum number of passages that can be associated with each query-term. Note that it is possible to have repeated passages for different query-terms at this point since one passage can contain more than one query-term.

The second block, named Interaction Network, has the objective of scoring each passage according to its relevance for a given query. To accomplish this, we employed a shallow interaction-based model to capture the most relevant matching signals that are present in each query-passage combination. In more detail, for each query-term-passage pair, we construct an interaction matrix with dimension $Q \times S$, also designated similarity tensor, by computing the cosine similarity between the query and the passage embedding. Then 3 by 3 convolutions are applied to learn n-gram relevance signals that are further extracted by multiple pooling layers (max, average, and k-max-average) applied to the filter dimension, creating a feature vector r per each query-term-passage pair. Consequently, $D(u_i)$ at this point is given by $D(u_i) = \{r_{i0}, r_{i1}, ..., r_{iP}\}$, where r_{ik} corresponds to the k-th passage feature vector with respect to the query term u_i. Finally, each resulting feature vector is linearly combined with a trainable vector, and a sigmoid function, σ, is applied to produce a sentence level relevance signal (1=relevant, 0=irrelevant).

The third block, the Aggregation Network, is responsible for producing the final document relevance score for the given query. This is accomplished by weighting the importance of each document passage score with respect to the importance of each query term. More precisely, this importance is given by a probabilistic distribution computed (softmax) over a linear combination of each query term embedding and a trainable vector and is defined as $A = \{a_{u_0}, a_{u_1}, ..., a_{u_Q}\}$, where a_{u_i} represents the probabilistic importance of the query-term u_i. Each passage score is then weighted by the corresponding query-term importance, a_{u_i}, and the resulting set of weighted scores is fed to a multi-layer perceptron to produce the final document score.

Our model is trained in a pairwise fashion and uses word-level pre-trained embeddings, specifically word2vec embeddings (Mikolov et al., 2013) in these experiments. More details on hyperparameters, configuration, and training can be found in previous work Almeida and Matos (2020b).

4 Evaluation

In this section, we aim to empirically demonstrate the efficacy of the proposed model and compare its efficiency against the state-of-the-art transformer-based models.

4.1 Efficacy

To compare the retrieval efficacy of the lightweight system and show that despite having only 620 trainable parameters, it is capable of competing with state-of-the-art models, namely transformer-based models, we reuse the results obtained in two ad-hoc retrieval competitions. Concretely, the results in both challenges derive from a two-stage retrieval pipeline, where our proposed neural model reranks the top-N documents previously retrieved by the BM25 (Robertson and Zaragoza, 2009) ranking function.

4.1.1 BioASQ - Challenge

The BioASQ challenge (Tsatsaronis et al., 2015) is an annual competition on document classification, retrieval, and question-answering, currently in the eighth edition. For the document retrieval task, the objective was to retrieve the most relevant articles from the PubMed/MEDLINE annual database.

Concerning the system, the BM25 filter was fine-tuned with the 2700 biomedical questions provided by the organisers as training data. Furthermore, the neural model was trained on the same data using a pairwise cross-entropy loss with cyclic learning rates. We also trained the word embeddings using word2vec (Mikolov et al., 2013) skip-gram algorithm directly on the PubMed/MEDLINE articles. For more details about our participation, we direct the readers to the system description paper (Almeida and Matos, 2020b).

System	**Document Retrieval**		
	Rank	MAP@10	GMAP@10
Batch 1 (total of 21 systems)			
Proposed Model	1	**38.23**	**1.63**
Top Competitor (Pappas et al., 2020)	4	36.48	1.14
Batch 2 (total of 26 systems)			
Proposed Model	3	37.19	**6.75**
Top Competitor (Kazaryan et al., 2020)	1	**39.45**	6.00
Batch 3 (total of 28 systems)			
Proposed Model	5	52.30	**7.63**
Top Competitor (Pappas et al., 2020)	1	**53.29**	6.25
Batch 4 (total of 26 systems)			
Proposed Model	3	48.10	**7.96**
Top Competitor (Pappas et al., 2020)	1	**49.92**	7.00
Batch 5 (total of 25 systems)			
Proposed Model	2	50.98	**6.52**
Top Competitor (Kazaryan et al., 2020)	1	**52.02**	6.34

Table 1: Summary of the results obtained in BioASQ. Values in bold represent the top score achieved during the competition.

The results summarised in Table 1 show that the system provided the best ranking overall in terms of GMAP@10 for all five batches and achieved top 3 results in all but one batch, in terms of MAP@10. Additionally, this challenge received an average of 25 submissions for each batch and, according to Nentidis et al. (2020), all the other top-performing systems used either BERT or a BERT variant.

4.1.2 TREC-Covid - Challenge

TREC-Covid (Roberts et al., 2020) was an initiative to rapidly promote the development of an automatic system capable of searching the growing literature about the 2019 novel coronavirus. The challenge followed a TREC style format and relied on the CORD-19 dataset (Wang et al., 2020a) as the collection of scientific articles about the novel coronavirus. The objective was to retrieve the most relevant articles from this collection for each topic given by the organisers. The system results were evaluated in a residual manner since, with the exception of the first round, the rounds shared topics that had already been evaluated. The metrics adopted were P@5, NDCG@10, Brepf, and MAP. The organisers also allowed the use of the evaluation feedback of previous rounds as training data to tune/train the submitted systems. An important note is that for the first round no training data was available since no previous evaluation had been performed.

Concerning the system, we utilised the BioASQ system in the first round, which means this was a zero-shot approach, exploiting the proximity of the domains. The only change was on the embeddings used, which were trained on the PubMed/MEDLINE articles plus the CORD-19 dataset. For the remaining rounds, we kept a similar approach, but we also fine-tuned (trained) the model with the feedback data from previous rounds. A complete description of our participation is available in Almeida and Matos (2020c).

System	Rank	P@5	NCDG@10
Round 1 (total of 100 automatic runs)			
Proposed Model	9	63.33	52.98
Top Competitor*	1	78.00	60.80
Round 3 (total of 79 runs)			
Proposed Model	2	**86.50**	77.15
Top Competitor** (Zhang et al., 2020)	1	86.00	77.40

Runs description:
*`https://ir.nist.gov/covidSubmit/archive/round1/sab20.1.meta.docs.pdf`
**`https://ir.nist.gov/covidSubmit/archive/round3/covidex.r3.t5_lr.pdf`

Table 2: Summary of the two best results achieved on TREC-Covid.

Comparatively to BioASQ, TREC-Covid was more challenging given the high number of participating teams. As an example, the first round had a total of 53 teams and approximately 100 submissions in the same category as our submission and hence comparable. This large number also covers a variety of IR solutions, ranging from simple traditional IR to transformer interaction-based models. Table 2 shows our best results, a ninth-place out of 100 in the first round and a second-place out of 73 in the third round. In both rounds, our system was able to beat traditional IR techniques and more recent transformer-based architectures, such as BERT and its variants, T5 (Raffel et al., 2020), among others. Additionally, our round 3 submission differed from the overall system since, motivated by the residual nature of the evaluation, we ended up utilising a (pseudo)-relevance feedback baseline instead of our BM25 baseline.

4.2 Performance

In this section, we address the performance side of the proposed neural model by empirically comparing it to the transformer-based behemoths. We propose three dimensions of comparison, namely the number of parameters, time to infer document relevance, and time to train. The first dimension has the objective of giving an idea of the model's size. The second dimension is especially concerned with proving the feasibility of using these models in real-world applications. Finally, the third dimension is concerned with the impact and resources required to build systems with these models.

All the measures were performed on a machine with the characteristics presented in Table 3, which we consider a good representative of a generally accessible setup. Additionally, all the experiments were performed on TensorFlow 2.2.0 with CUDA 10.1, and for the transformer-based models, we adopted the HuggingFaces (Wolf et al., 2020) library with the TensorFlow version of the models. To keep the comparison fair we utilize the "tf.function"[1] decorator to convert the model to a static computation graph, significantly speeding up the transformer-based models.

CPU	2x Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz
GPU	Nvidia Tesla K80 with 12 GB
RAM	128 GB

Table 3: Specification of the hardware used during the experiments.

An important aspect is that we do not have access to the systems listed in Section 4.1. To keep a fair comparison, we implemented a simple minimal working example that uses a transformer-based model for ad-hoc retrieval. Pursuing the ideas of (Yang et al., 2019; Nogueira et al., 2019; Dai and Callan, 2019b), we fed to the model the following input "[CLS] query tokens [SEP] document tokens [SEP]", where "[CLS]" and "[SEP]" are special boundary tokens, with the first one being a classifier token that is further fed to a multi-layer perceptron to compute the final document relevance. We

---

[1] https://www.tensorflow.org/versions/r2.2/api_docs/python/tf/function

set the input size to the maximum that the model could handle and applied padding or truncation. This implementation gives us a best-case scenario in terms of computational performance for ad-hoc retrieval.

We compared the following transformer-based models against our proposed solution:

- BERT (x12): Corresponds to the original checkpoint “bert-base-uncased” of the BERT model and has 12 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 110 million trainable parameters.
- BERT (x24): Corresponds to the original checkpoint “bert-large-uncased” of the BERT model and has 24 layers, 1024 hidden dimension, 16 multi attention heads, totaling approximately 340 million trainable parameters.
- distilBert: Corresponds to the original checkpoint “distilbert-base-uncased” of the distilBERT model and has 6 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 66 million trainable parameters.
- ALBERT: Corresponds to the original checkpoint “albert-base-v2” of the ALBERT model and has 12 repeating layers, 128 size embeddings, 768 hidden dimension, 12 multi attention heads, totaling approximately 11 million trainable parameters.
- Longformer: Corresponds to the original checkpoint “allenai/longformer-base-4096” of the Longformer model and has 12 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 149 million trainable parameters.

We mostly use BERT models since these are widely adopted in the literature. Moreover, we also include distilBERT since it is currently one of the fastest transformer-based models available, and ALBERT, given its comparatively small number of trainable parameters. Furthermore, all the models had a maximum input size of 512, with the exception of Longformer, which had an input size of 4096.

Some final notes regarding our neural model implementation are also relevant. In the first place, this model was constructed to run on the CPU, and so we do not consider operations and data flow that would be easily accelerated on the GPU. Furthermore, our implementation is in a prototype phase, which means that we have not performed any optimisations. For example, in the current state, the model performs some redundant computation, and the amount of padding data reaches approximately 90%.

4.2.1 On Inference

Regarding the inference tests, these were performed both in CPU and GPU since it is well known that the transformer architecture can achieve a drastic speed up on the GPU. However, on the other side, this brings more operational costs and offers a new single point of failure that real-world systems would need to deal with.

With respect to the experiment, it consists of computing the document relevance score for a BioASQ test set that contains 100 queries and a pool of 250 documents for each question, making a total of 25000 query document pairs. We fed this to the models in a systematic way, only varying the batch size and measured the elapsed time in seconds. We discarded the time taken on the first and last batch since the first batch may perform some initialization operations and the last batch has fewer documents than the others. Moreover, in the column “Tokens seen”, we show the number of tokens fed to the model per sample.

Model	**Tokens seen**	**Time elapsed** (on a batch of **16**)	
		CPU(s) **Mean ± std**	**GPU(s)** **Mean ± std**
Proposed model	4500	0.074 ± 0.006	0.824 ± 0.093
distilBERT	512	2.421 ± 0.038	0.551 ± 0.007
BERT (x12)	512	4.967 ± 0.072	1.133 ± 0.011
ALBERT	512	5.144 ± 0.081	1.198 ± 0.027
BERT (x24)	512	15.729 ± 0.185	3.511 ± 0.052
LongFormer	4096	102.051 ± 1.009	*11.188 ± 0.035

*Measured on a batch size of 8

Table 4: Inference times in seconds measured over 25000 query document pairs using a batch size of 16 on a CPU and GPU.

Table 4 presents the measured times using a batch size fixed at 16. As illustrated, our simple model can run 32 times faster on the CPU compared to the fastest transformer-based solution while processing almost 9 times the amount of tokens. Compared with a model that processes almost the same amount of data, in our experiments the LongFormer, the time difference becomes more evident, with our proposed model being 1379 times faster.

Regarding the behaviour when running on the

GPU, our proposed model is not able to achieve a speedup when compared to the CPU time and to the transformer-based models. However, we believe the main reason for this is related to the time needed to transfer the data to the GPU since the proposed model needs to send 9 times more data, which also gives us an indication that it may not be worth to run on the GPU due to the small number of operations that the model performs compared to the amount of data that it needs to send. Furthermore, the results that we show for the LongFormer were obtained with a batch size of 8, since we could not fit the model with a batch of 16 on the GPU, due to memory constraints.

Figure 2 presents the models inference time for four different batch sizes, $\{16, 32, 64, 258\}$. On the left, which corresponds to the experiments on the CPU, the models seem to scale with the increase of the batch size linearly. Additionally, only our proposed model is capable of keeping an inference time under one second, which can be viewed as an acceptable query latency time. Similarly, on the right side, corresponding to the experiments performed on the GPU, we observe the same linear behaviour for the transformer models. However, the proposed model shows an almost constant time, which reinforces the idea that when executed in the GPU, the most time-consuming operation is data transfer.

4.2.2 On Training

The training experiments were performed on both CPU and GPU, similarly to the inference. Regarding this experiment, we follow a pairwise training approach, which may be a computationally heavier option when combined with the transformer architecture, since these are usually trained as a binary classification problem, i.e., a document is relevant to a question or not. However, we decided to keep the pairwise approach since the proposed model was designed to use it, this way keeping a more fair comparison and also demonstrating the burden of training transformer-based models in a pairwise setting. Moreover, the transformer-based model can be entirely finetuned, i.e., training the classification layer and all layers, or just training the classification layer, which we also evaluate, since it should speed up the training.

Concerning the training data, we used 100 questions from the BioASQ training dataset each associated with a list of relevant documents and a list of negative documents randomly sampled from the BM25 ranking order, producing a total of 1104 training samples.

After the sampling process, we store this data to ensure that every model processes exactly the same set of triplets (query, positive document and negative document). To further ensure homogeneity among the experiments, we fixed every random seed that we were aware of.

Model	**Tokens seen**	**Time elapsed**	
		(on a batch of **16**) **CPU(s)** **Mean ± std**	(on a batch of **1**) **GPU(s)** **Mean ± std**
Proposed model	4500	0.350 ± 0.011	1.204 ± 0.234
distilBERT	512	9.700 ± 0.055	1.116 ± 0.007
BERT (x12)	512	19.727 ± 0.280	0.484 ± 0.005
ALBERT	512	19.835 ± 0.414	0.451 ± 0.003
BERT (x24)	512	66.536 ± 0.724	1.582 ± 0.040

Table 5: Complete training times, i.e., training all the layers, in seconds measured over 1104 samples on a CPU and GPU.

Table 5 shows the time required to fully train the transformer-based models, i.e., training all the layers plus the classifier layer, in a pairwise setting. Similarly to the inference, we also adopted a batch of 16 for the CPU and a batch of 1 for the GPU, since we could not perform the training on the GPU due to memory issues for some models. Additionally, we did not measure results for the LongFormer, neither in CPU or GPU, since we were unable to store the model plus all the gradients on memory (128GB) for the given batch size.

Regarding the results, all the models seem to be four times slower at training time when compared with the inference times. As expected, the proposed model presented the fastest training time, being 10 times faster than the fastest transformer model. Also note that, for all the models, training with a batch of 1 on GPU does not bring any advantages over training on the CPU, which is approximately 3 times slower (per batch) but for a batch size that is 16 times larger. Finally, regarding the performance of the proposed model on the GPU, it also seems to support the previously enunciated problem related to data transfer to the GPU.

Table 6, presents the measured time, in seconds, required only to train the classification layer in the transformer-based models. In this case, both the CPU and GPU experiments could be done on a batch size of 16, since it is only necessary to store gradients for the classification layer.

Looking at the results, if we only update the classifier layers weights, the transformer-based models

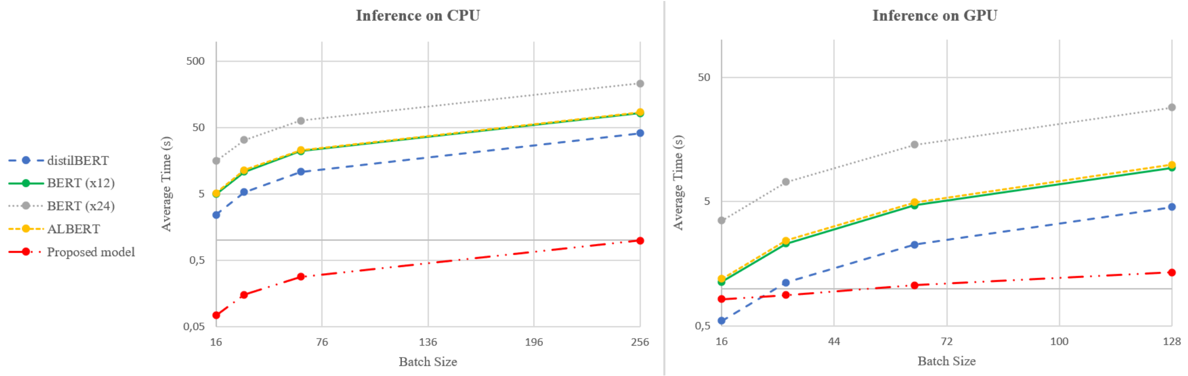


Figure 2: Inference times on the CPU, left side, and on the GPU, right side, as a function of the batch size.

Model	**Tokens seen**	**Time elapsed** (on a batch of **16**)	
		CPU(s) **Mean ± std**	**GPU(s)** **Mean ± std**
Proposed model	4500	0.350 ± 0.011	1.008 ± 0.094
distilBERT	512	3.696 ± 0.098	1.108 ± 0.011
BERT (x12)	512	7.513 ± 0.098	2.266 ± 0.012
ALBERT	512	7.867 ± 0.202	2.400 ± 0.011
BERT (x24)	512	24.699 ± 0.377	7.042 ± 0.033

Table 6: Training times, in seconds, only for the classification layer measured over 1104 samples on a CPU and GPU.

training becomes 2.5 times faster. Additionally, using the GPU further improves this speed up to 8.7 times. However, even using the GPU, the fastest transformer based model, distilBERT, is still 3.1 times slower than our proposed model.

5 Discussion

In this section, we take into consideration the presented results, both in terms of efficacy and performance, and present an overall overview of the comparison of our lightweight model and the transformer-based approaches.

First of all, we acknowledge some limitations of the proposed model, which are all related to the current implementation that processes a large portion of padding data, increasing the number of tokens that the model sees and thus injuring the performance of the current solution. However, this is something that can be addressed in future work by rearranging the model data flow and better understanding the required maximum input size of the query and document tokens. Moreover, to get a sense of speed up that we could achieve, we made a comparison with a transformer-based model that is able to encode up to 4096 tokens, which is close to the 4500 tokens of the proposed model. In this case, the lightweight model ends up being 1379 times faster, which gives an idea that the expected speedup may be in the order of a thousand.

Recalling the three proposed dimensions of comparison, the proposed model excels in all of them. First of all, it is a model with 10 thousand times fewer trainable parameters when compared to distilBERT. In the efficacy evaluation, the results show that the proposed model could achieve close to state of the art performance in both biomedical ad-hoc retrieval tasks. Finally, according to the presented inference times, more precisely, with the times presented in Figure 2, it is observable that the proposed model was the only one to perform inference under one second even with a batch size of 256, which is more than enough to adopt in real-world ad-hoc retrieval applications. Furthermore, this observation enables us to conclude that the transformer-based model solution is currently not suitable for a real-world ad-hoc retrieval application unless large scale computational resources are available.

Moreover, our results also show that the proposed model was always the fastest model by a significant margin, even though using only the CPU. This, in our view, ends up to be a preferable characteristic, since it facilitates the deployability and scalability of systems that implement this model.

6 Conclusion

In this paper, we discuss the feasibility of using transformer-based models in real-world ad-hoc retrieval applications and show an extreme lightweight solution, with only 620 trainable parameters.

We evaluate the solution against the transformer-based models, in terms of efficacy and performance, and show that this model is capable of matching the efficacy offered by the transformer-based models in two biomedical ad-hoc retrieval challenges while being considerably faster by just using the CPU compared to the best GPU runs of the transformer-based models.

Acknowledgments

This work received support from EU/EFPIA IMI 2 JU under grant No 806968, and from National Funds through FCT - Foundation for Science and Technology, in the context of project UIDB/00127/2020 and grant 2020.05784.BD.

References

Tiago Almeida and Sérgio Matos. 2020a. Calling attention to passages for biomedical question answering. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part II*, volume 12036 of *Lecture Notes in Computer Science*, pages 69–77. Springer.

Tiago Almeida and Sérgio Matos. 2020b. BIT.UA at bioasq 8: Lightweight neural document ranking with zero-shot snippet retrieval. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Tiago Almeida and Sérgio Matos. 2020c. Frugal neural reranking: evaluation on the covid-19 literature. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. 2011. *Modern Information Retrieval: The Concepts and Technology behind Search*, 2nd edition. Addison-Wesley Publishing Company, USA.

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

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3079–3087. Curran Associates, Inc.

Zhuyun Dai and Jamie Callan. 2019a. Context-aware sentence/passage term importance estimation for first stage retrieval.

Zhuyun Dai and Jamie Callan. 2019b. Deeper text understanding for ir with contextual neural language modeling. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. pages 126–134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Chengxiang Zhai, and Xueqi Cheng. 2018. Modeling diverse relevance patterns in ad-hoc retrieval. *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval*.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Croft. 2016. A deep relevance matching model for ad-hoc retrieval. pages 55–64.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. pages 328–339.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1049–1058, Copenhagen, Denmark. Association for Computational Linguistics.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Ashot Kazaryan, Uladzislau Sazanovich, and Vladislav Belyaev. 2020. Transformer-based open domain biomedical question answering at bioasq8 challenge. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Efficient document re-ranking for transformers by precomputing term representations. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ryan McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. pages 1849–1860.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval t. *Foundations and Trends® in Information Retrieval*, 13:1–126.

Anastasios Nentidis, Anastasia Krithara, Konstantinos Bougiatiotis, Martin Krallinger, Carlos Rodriguez-Penagos, Marta Villegas, and Georgios Paliouras. 2020. Overview of bioasq 2020: The eighth bioasq challenge on large-scale biomedical semantic indexing and question answering. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 194–214, Cham. Springer International Publishing.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with bert.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 257–266, New York, NY, USA. Association for Computing Machinery.

Dimitris Pappas, Petros Stavropoulos, and Ion Androutsopoulos. 2020. AUEB-NLP at bioasq 8: Biomedical document and snippet retrieval. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

A. Radford. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Kirk Roberts, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, Kyle Lo, Ian Soboroff, Ellen M. Voorhees, Lucy Lu Wang, and William R. Hersh. 2020. TREC-COVID: rationale and structure of an information retrieval shared task for COVID-19. *J. Am. Medical Informatics Assoc.*, 27(9):1431–1436.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael Alvers, Dirk Weißenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel-Cyrille Ngonga Ngomo, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, and Georgios Paliouras. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16:138.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020a. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020b. Linformer: Self-attention with linear complexity.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple applications of bert for ad hoc document retrieval.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Edwin Zhang, Nikhil Gupta, Raphael Tang, Xiao Han, Ronak Pradeep, Kuang Lu, Yue Zhang, Rodrigo Nogueira, Kyunghyun Cho, Hui Fang, and Jimmy Lin. 2020. Covidex: Neural ranking models and keyword search infrastructure for the covid-19 open research dataset.

Reanalyzing the Most Probable Sentence Problem: A Case Study in Explicating the Role of Entropy in Algorithmic Complexity

Eric Corlett **Gerald Penn**
Department of Computer Science
University of Toronto
{ecorlett,gpenn}@cs.toronto.edu

Abstract

When working with problems in natural language processing, we can find ourselves in situations where the traditional measurements of descriptive complexity are ineffective at describing the behaviour of our algorithms. It is easy to see why — the models we use are often general frameworks into which difficult-to-define tasks can be embedded. These frameworks can have more power than we typically use, and so complexity measures such as worst-case running time can drastically overestimate the cost of running our algorithms. In particular, they can make an apparently tractable problem seem NP-complete. Using empirical studies to evaluate performance is a necessary but incomplete method of dealing with this mismatch, since these studies no longer act as a guarantee of good performance. In this paper we use statistical measures such as entropy to give an updated analysis of the complexity of the NP-complete Most Probable Sentence problem for pCFGs, which can then be applied to word sense disambiguation and inference tasks. We can bound both the running time and the error in a simple search algorithm, allowing for a much faster search than the NP-completeness of this problem would suggest.

1 Introduction

Natural Language Processing uses many algorithms that are theoretically intractable, but work well in practice. The k-means clustering algorithm, for example, has an exponential worst-case time, but is generally polynomial in practical applications (Arthur et al., 2009). If we look at problems, as opposed to algorithms, the task of training neural networks (Blum and Rivest, 1993), sentence disambiguation in pCFGs and HMMs (Sima'an, 2002), and solving cryptograms (Nuhn and Ney, 2013) have even been shown to be NP-complete. These results would seem to be at odds with the observation that we are able to actually perform these tasks. Clearly, while NP-completeness results are a necessary part of our understanding, they do not tell the whole story about our algorithms' behaviour.

Faced with the inadequacy of existing descriptive complexity measures to represent program operation, some researchers have resorted to empirical measures of performance (e.g. Carroll, 1994). Unfortunately, empirical studies have their own pitfalls, and these are also difficult to address. Specifically, empirical studies of program performance have an implicit dependence on the distribution of inputs used in the study. Changing this distribution of inputs can change the overall performance drastically. This, after all, is why we separate training and evaluation test sets.

Ultimately our analytic tools should work together with our empirical studies, in the sense that we should be able to give quantify how much program performance changes as we change our input distributions. This would allow us to circumscribe our guarantees of program performance; if those bounds are not met (e.g., if a program takes much longer than expected to run), we have evidence that the set of inputs did not comply with the distribution in the first place.

The situation of NLP is a little more complicated, however. Many of our algorithms include, implicitly or explicitly, probability distributions as part of their input. Take for example a natural language parser. Thirty years ago, these relied upon lexicalized pCFGs that explicitly assigned probabilities to different rules and word n-grams, whereas today, the distributions used by neural parsers avail themselves of distributions that are rather more implicit. But in both cases, the distributions are acquired by sampling training data.

The aim of our research programme is to incorporate these distributions into our theoretical anal-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3354–3362
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

yses of program performance. If we were to follow existing analyses of a program's dependence on textual input, we may choose to characterize that dependency as a function of the length of the distribution's description. Uniform distributions, arguably the shortest to write, often ensure the worst program performance. Indeed, there is plenty of evidence from statistics, information theory and other AI applications that a particular derivative characteristic of distributions which is systematically related to the length of their descriptions captures many aspects of this dependency: *entropy*.

In this paper we take steps towards this goal by demonstrating how to link the run time of a specific inference problem, the Most Probable Sentence (MPS) problem for pCFGs (Sima'an, 2002), to the entropy of its input. While this is somewhat of a niche problem, it does provide a good demonstration of our approach. We will describe below in Section 8 some other graphical models that this approach can be readily applied to.

We start our analysis of the MPS problem in Section 3 by outlining the problem definition, the NP-hardness proof of the problem, and the main bound we are going to prove about it. In Sections 4 and 5, we develop an analysis of the MPS problem by showing how entropy, specifically, the conditional entropy of a parse tree given the words of a sentence, changes the running time of a simple search algorithm for the most probable tag sequence over that sentence. In Section 6, we show that there is an inherent trade-off between accuracy and run time whose worst case is dictated by the pCFG sentence-to-tree entropy. Finally, in Section 7, we characterize the run time of our search when we hold our grammar constant and draw many sentences from it.

2 Previous Work

Historically, most complexity results in computer science have dealt with worst-case complexity. A major NLP result in this vein has been the study by Sima'an (2002), which shows that the most probable parse problem (MPP) for pSTSGs and the most probable sentence problem (MPS) for both pSTSGs and pCFGs are all NP-complete. These problems can be thought of as renormalization over a graphical model, and have been studied further in several publications, including De la Higuera and Oncina (2013) and Goodman (1998). Similar NP-completeness results exist for the problem of finding the optimal word order for phrases in machine translation (Germann et al., 2001) and for solving letter-substitution problems such as cryptograms (Nuhn and Ney, 2013).

An argument could be made for average-case complexity (Levin, 1986; Impagliazzo, 1995) as an alternative to what we are attempting here. Average-case complexity is a step towards the type of results that we want, but it still has its shortcomings. In particular, an average case complexity analysis relies on the underlying distribution of inputs being known (and, usually, easy to work with) in advance. The distribution of choice is often naïvely uniform. In situations where the input distribution is defined by the problem task, or where it is actually defined as part of the input, the problem of how program behaviour changes with the distribution is not completely addressed. Average-case complexity, moreover, does not give any sort of guarantee on program performance.

Within artificial intelligence, approximation results are perhaps a more natural approach to analysis. Nearly all of these endeavour to establish a worst-case result, however, and in some cases, such as with Markov Logic Networks and Bayesian networks (Roth, 1996; Cooper, 1990), it is hard even to approximate for probabilistic inference. What we want is the worst case, but not the worst distribution. We want the worst case for a typical distribution.

A general framework for "probabilistic complexity" has been explored in Ackerman et al. (2011), in which it is found that the calculation of values such as conditional probability can be intractable, even on an otherwise tractable distribution. On the other hand, attempts to give decompositions for large families of distributions into tractable bases, as described in Erdélyi et al. (2009), can be used to characterize input probabilities for which algorithms will often be efficient. What these papers do not do is explore how the task-specific aspects of a problem in a powerful model can shift an algorithm's complexity into a hard or easy part of the input space.

A promising alternative approach to analyzing program complexity is to look at the smoothed complexity of an algorithm. In this paradigm, the complexity of an algorithm is not based on the worst input that an arbitrary adversary could choose for the problem. Instead, the adversary is constrained by having a small error added to their inputs. The expected complexity of the resulting problem is then reported, and can often be much better than the usual worst case. This approach to complexity analysis is very effective for describing the behaviour of many otherwise exponential algorithms, such as the simplex algorithm (Spielman and Teng, 2004) or the k-means clustering algorithm (Arthur et al., 2009). The major limitation of this approach for us, however, is that the improvement in performance is assumed to be due to measurement error in the input. Such an assumption is very appropriate when talking about, for example, measurements taken of continuous real-world quantities like length. It is not likely to be appropriate for situations where the program improvement is due to observable regularities in the input data.

What we really want to do is to exploit the fact that we see only a fragment of the theoretically possible input space in the real world. Unfortunately, the task of identifying good problem fragments for our models is itself very hard, and so we will instead attempt to use statistical measures to build tools for narrowing down the space in which our inputs will lie. A start towards this goal is found for the letter-substitution problem in (Corlett and Penn, 2013), but this approach does not give a tight enough bound to describe when we should expect better running times, and it does not immediately generalize to other tasks. We would like to strengthen the approach used in that paper to make it more applicable to different areas. With this in mind, we turn to another NP-complete problem, the MPS problem for pCFGs.

3 Tagging and NP-Completeness

To start, we recall the definition of the problem:

Most Probable Sentence Problem (MPS)

Instance: A pCFG model G and a string s.
Question: Find a sequence of part-of-speech tags, σ, that maximizes the sum of probabilities of trees with yield s and pre-terminals σ.

As an example, suppose we were looking at the sentence "Time flies like an arrow". There are several assignable tag sequences, such as

$$N\ V\ Adv\ Det\ N$$

or

$$Adj\ N\ V\ Det\ N.$$

Each tag sequence might be obtainable from more than one parse tree. The probability of any such tag sequence is the sum of the probabilities of the trees yielding that sequence.

The difficulty of this problem lies in the fact that we are looking for the POS tags that maximize the sum of the probability of all trees using those tags. If we were looking for the single most likely tree, or if we were looking for the sum of all trees with yield s, without fixing the POS tags, the usual $\mathcal{O}(n^3)$ parsers would work.

A reduction, as described in Sima'an (2002) is made from 3-SAT as follows: given a 3-SAT formula ϕ with η clauses and κ variables, we build a pCFG grammar G_ϕ that generates strings that contain 3η copies of a single terminal, such as x. The tags for these terminals are either T or F – that is, they are the truth assignments for the literals of ϕ (e.g., if $\eta = 2$, we would generate six-tag sequences such as *TTFTTF*). The grammar G_ϕ is capable of generating two types of trees for every input string: one that generates true assignments that may or may not be consistent across variable instances, and one that generates consistent assignments for one designated variable. A string of tag assignments that has one tree of the first type and κ trees (one for each variable) of the second type will have a total probability that is higher than trees that lack this many, and so there is a threshold Q that separates input strings with true, consistent assignments from input strings without true, consistent assignments.

In the following sections, however, we will prove the following:

Theorem 3.1 *Suppose that we draw a sentence s from a pCFG grammar G. Then, the MPS problem can be solved in $\mathcal{O}(n^4[p_G(s)/p_G(\tau(s))])$ time, where:*

- *n is the length of s,*
- *$\tau(s)$ is the tag sequence assigned within the most likely tree, $t(s)$, that has yield s, and*
- *$p_G(s)$ is the probability of s in G, and $p_G(\tau(s))$ is its probability given $\tau(s)$.*

This is a major improvement over the bound of Sima'an (2002) because it shows that the exponential run times that we expect from an NP-complete problem are in fact bounded by the ratio $p_G(s)/p_G(\tau(s))$. For the pathological grammars that are used in the NP-completeness proof, this ratio grows exponentially with the size n of the sentence. On the other hand, we conjecture that it is quite small for the natural language grammars that we see in practice. Empirically computing the average such ratio on a corpus can itself be very trying,[1] but there are good indications that it is relatively constrained. In the Penn Treebank, for example, 46.5% of the word tokens in the corpus have only one POS tag assignable, and these words are fairly well dispersed throughout the corpus: the median intrasentential distance between them (not including the single-tag words themselves) is 1, with an average of 4.4. In any case, a parser that avails itself of this theorem does not need to explicitly calculate $p_G(s)/p_G(\tau(s))$. $p_G(s)/p_G(t(s))$, which directly relates to the probability of the best tree for s, is likely to be a much looser bound. We will attempt to bound it theoretically in Section 7.

Proof of Theorem 3.1: This is an immediate consequence of Lemmata 5.5 and 5.6. The proofs are presented below. Section 4 describes a simple search algorithm that will be the focus of the discussion. Section 5 will analyze the running time of this algorithm in a way that provides the basis for these lemmata. In Section 6, we will extend this result to describe the error incurred if we allow early stopping. ■

4 Analysis of Problem and Exemplary Algorithm

We will use a similar approach to that used in (Corlett and Penn, 2013), in that we will use an A^* search to solve the MPS problem for pCFGs. The idea behind the search is as follows: given a pCFG G and a string s of n words, we can find the overall probability $p_G(s)$ in G through a cubic time algorithm that algebraically mirrors a conventional all-paths parsing algorithm. But for our purposes, we can do it differently: given any instance w of a word in s, we can make a guess as to the POS tag r that w will take, and then run the algorithm, but with the restriction that only the tag r will be counted for w. In our earlier example of "Time flies like an arrow," perhaps this would mean running the parser given the restriction that "Time" is an adjective, while letting the other words take any tag that they can. Clearly, this run of the parser will still take $\mathcal{O}(n^3)$ time, as it would if we simultaneously constrained any subsequence of the words of s rather than just one.

Recall that $p_G(s)$ is the sum of all $p_G(t)$, where t ranges over all trees with the yield of s. Furthermore, if, for a subsequence of word instances $(w_{u_1}, w_{u_2}, \ldots, w_{u_J})$, we restrict the parser so that only the POS tags $(r_1, r_2, \ldots, r_J)$ will be considered, the probability calculated will be the sum of $p_G(t')$, where t' ranges over all trees with both have the yield of s and which have the desired POS tag restrictions. We will use this probability as an admissible heuristic in our A^* priority queue, and consider possible sequences of tag assignments as nodes in the search space. We will refer to sequences of tag assignments with the character a, and we will index them by the letter i. Sequences of tag assignments will not, in general, fix a tag for every word in s.

To run the search, we fix an order for the words in s. We assume a last-to-first ordering here, but any ordering will suffice. We push an empty assignment $\{\}$ with the score $p_G(s)$ onto a probability queue Q. While Q is nonempty, we pop its maximum element a_i and look for the first word w_u in our ordering of s which is not fixed by a_i. For every possible POS tag r_j for w_u, we add the POS tag assignment $\{w_u : r_j\}$ to a_i to get a new sequence of tag assignments $a_i^{[j]}$, compute $p_G(s)$ restricted to $a_i^{[j]}$, and insert $a_i^{[j]}$ into Q with the resulting probability. We return the first tag assignment a popped from Q that fixes every word in s.

To see that this A^* search is correct, we first note that it must terminate: every time a tag assignment A is popped from the queue, either a fixes every word in s and the program terminates, or a series of strictly longer tag assignments a' is added to the queue.

Furthermore, since our heuristic for a tag assignment sequence a is the sum of the probabilities of all suitably restricted trees t in G with yields of s, we can see that when we extend a tag assignment sequence a, we are simply adding new constraints to the contributing trees t, and so the set of trees counted is non-increasing. This means that the heuristic is also non-increasing. The same argument tells us that any extension of a tag assignment sequence a that fixes every word in s must have a probability that is at most that of s given a. These arguments, taken together, indicate that the first tag sequence found that fixes every word in s will be the one that gives the maximum probability, and so this assignment will give the solution to the MPS problem.

---

[1] The reader may wish to refer to our work on this subject at `https://doi.org/10.5683/SP2/CM9QY1`.

5 Initial Analysis

In order to find the time complexity of this algorithm, recall the fact that the admissible heuristic is non-increasing. If we apply this to the first and last tag assignment sequences popped from the stack, we can see:

Lemma 5.1 *If $M_G(s)$ is the probability of s given its most likely tag sequence, and if a is any tag sequence expanded in the search, then*

$$p_G(s) \geq p_G(s|a) \geq M_G(s).$$

Similarly, if a is a tag sequence that is seen but not expanded in the search,

$$M_G(s) \geq p_G(s|a).$$

So understanding the algorithm complexity becomes an issue of determining how many a are in this range. Furthermore:

Lemma 5.2 *If we extend a specific tag sequence a_i in our search by one tag assignment into the new sequences $a_i^{[1]}, a_i^{[2]}, \ldots, a_i^{[m]}$, then*

$$p_G(s|a_i) = \sum_j^m p_G(s|a_i^{[j]}).$$

This just means that in our example sentence $s =$“Time flies like an arrow,” if we were to argue that “flies” can only be an *N* or a *V*, then the probability of s is the sum of the probability of s given that “flies” is a *N* plus the probability of s given that “flies” is a *V*. Any of the parse trees that are compatible with one restriction are incompatible with the other, and so the outcomes in their probabilities are disjoint.

In fact, since Lemma 5.2 applies to every tag sequence we expand, we can go so far as to apply it iteratively to cover the whole search space:

Corollary 5.2.1 *Let QO be the selected optimal solution together with the items left stranded on Q at the end of the search. Then:*

$$\sum_{a \in QO} p_G(s|a) = p_G(s).$$

Let D^* be the set of all a_i such that every expansion of a_i, $a_i^{[1]}, a_i^{[2]}, \ldots, a_i^{[m]}$ is a member of QO. Some of the members of QO will be complete sequences of tag assignments — every word in s has received a tag in these. Others may not be complete, because the search terminated before they could be expanded. Because of this, D^* is not merely the set of QO parent nodes in the search space — some parents will be left out because they were extended both to assignment sequences that were further extended, as well as to assignment sequences that were not further extended.

Let D' be the subset of QO consisting of every node with its parent in D^*. Every element of QO only has one parent node (no joins in the search space) because we always marshal out the tags of s in the order that we defined over its words. By Corollary 5.2.1:

$$\begin{aligned} p_G(s) &= \sum_{a \in QO} p_G(s|a) \\ &= \sum_{a \in D'} p_G(s|a) + \sum_{a \in QO \setminus D'} p_G(s|a) \\ &\geq \sum_{a \in D'} p_G(s|a). \end{aligned}$$

Then, by repeated applications of Lemma 5.2 to the elements of D^*:

Lemma 5.3

$$p_G(s) \geq \sum_{a \in D'} p_G(s|a) = \sum_{a \in D^*} p_G(s|a).$$

Interestingly, the nodes of D^* are the nodes that are the most complete assignment sequences that still satisfy $p_G(s|a) \geq M_G(s)$, from the first clause of Lemma 5.1 — every node expanded in the search is either in D^* or is an ancestor of a node in D^*. It takes at most $n-1$ steps of expansion to arrive at this frontier, because every step assigns a tag to a new word of s, and there are at most $|D^*|$ such paths through the search space. So the total number of nodes expanded must be at most $\mathcal{O}(n|D^*|)$.

Lemma 5.4 *Let $R_G(s) = p_G(s)/M_G(s)$. The total number of nodes expanded must be at most $\mathcal{O}(nR_G(s))$.*

Proof: We have just learned that for every $a \in D^*$, $p_G(s|a) \geq M_G(s)$. Therefore, by Lemma 5.3, $p_G(s) \geq \sum_{a \in D^*} M_G(s)$, and so $R_G(s) \geq \sum_{a \in D^*} 1 = |D^*|$. ■

Finally, note that every one-word expansion in our search procedure comes with a decoding step that is bounded by $\mathcal{O}(n^3)$ time. Thus we have:

Lemma 5.5 *The running time for the entire search is $\mathcal{O}(R_G(s)n^4)$.*

The proof by Sima'an (2002) of overall NP-completeness implies that $R_G(s)$ can be exponentially large in n. His reduction exhibits a grammar which generates a single s with probability one, while the value of $M_G(s)$ is proportional to a threshold value, which is exponentially small in n. So in this degenerate case, the ratio $R_G(s)$ will always be exponentially large in the size of the grammar. In any case, determining $R_G(s)$ is the key to understanding the running time of our search.

Finding $R_G(s)$ is as difficult as finding $M_G(s)$, since we can use one to calculate the other. So we cannot directly use $R_G(s)$ in our calculations — we need a bound for it instead. Clearly, given a positive lower bound for $M_G(s)$, L, then $R_G(s) \leq p_G(s)/L$ also.

A very simple lower bound L is the most probable single parse tree $t(s)$ for s, which can be calculated in $\mathcal{O}(n^3)$ time. Clearly, the probability of $t(s)$ is at most the probability of all trees with the same tag assignment as $t(s)$, and this sum is itself at most the probability of the largest full tag assignment. Thus:

Lemma 5.6 *For any sentence s,*

$$R_G(s) \leq p_G(s)/p_G(\tau(s)) \leq p_G(s)/p_G(t(s)).$$

In the case of pure pCFG parsing, finding $p_G(s)$ also admits an $\mathcal{O}(n^3)$ algorithm, although the implicit sum over all possible trees includes a potentially infinite series over cycles of unary phrase structure rules. When this series converges, finding the probability mass added by these extra trees involves inverting a potentially large matrix that can be prohibitively expensive, although this cost is independent of the input length, and can be precomputed offline.

6 A Trade-off Between Accuracy and Time

Practically, we may not want to run a full A^* search, especially if the overall search takes a long time. The above analysis is a convenient starting point for investigating the consequences of stopping our search early.

If we do decide to stop the A^* search early, we will want to output a tag sequence. Let this tag sequence be $\hat{a}(s)$. We will assume here that it is $\tau(s)$, the one associated with the most likely tree, $t(s)$, which means that it can be calculated easily, and can easily be combined with any other estimate (just check both tag sequence estimates and take the better).

The heuristic score of $\hat{a}(s)$ provides a lower bound on the probability of the most likely tag sequence, which may in fact not be $\hat{a}(s)$. As the search progresses, those scores will decrease, and so the range of values that the maximum probability can take on will also decrease. In particular, the error that we incur by simply choosing the tag sequence $\hat{a}(s)$ as our output will always be at most $p_G(a_i)/p_G(\hat{a}(s))$. Given some $k \leq n$, let us run our A^* search for $n2^k$ iterations. We assert that:

Theorem 6.1 *If we run our search for $n2^k$ iterations, we will get a reduction in error of at least k bits, i.e., we achieve an overall reduction in error of at least 2^{-k}.*

Proof: Suppose otherwise. Then, since $p_G(s|a_i)$ is the score of the latest partial solution on the queue after $n2^k$ iterations, we have that $p_G(a^*) < p_G(s|a_i)$, where a^* is an optimal tag assignment. But, since the error that we incur by choosing $\hat{a}_t$ is not reduced by at least 2^{-k}, then $p_G(s|a_i)/p_G(\hat{a}(s)) > 2^{-k}(p_G(s)/p_G(\hat{a}(s)))$.

Let Q_i be the partial tag sequence a_i together with the items left stranded on Q when we terminate the search. Then, as we saw in Corollary 5.2.1:

$$\sum_{a \in Q_i} p_G(s|a) = p_G(s).$$

Furthermore, let D_i and D'_i be defined analogously to D^* and D' from Section 5. The arguments of Lemma 5.3 still apply, so that

$$\sum_{a \in D_i} p_G(s|a) \leq p_G(s).$$

Since $p_G(s|a_i) \leq p_G(s|a)$ for every $a \in D_i$, it follows that:

$$\begin{aligned} &\sum_{a \in D_i} p_G(a) = p_G(s) \\ \Rightarrow &\sum_{a \in D_i} (p_G(s|a_i)) \leq p_G(s) \\ \Rightarrow &\sum_{a \in D_i} 1 \leq p_G(s)/(p_G(s|a_i)), \end{aligned}$$

in which case:

$$
\begin{aligned}
|D_i| &\leq p_G(s)/(p_G(s|a_i)) \\
&= \Big(\frac{p_G(s)}{p_G(\hat{a}(s))}\Big)/\Big(\frac{(p_G(s|a_i))}{p_G(\hat{a}(s))}\Big) \\
&< \Big(\frac{p_G(s)}{p_G(\hat{a}(s))}\Big)/\Big(2^{-k}\big(\frac{p_G(s)}{p_G(\hat{a}(s))}\big)\Big) \\
&= 2^k.
\end{aligned}
$$

At this point, the number of nodes expanded by the algorithm is at most the number of unexpanded nodes in the search space times the number of their ancestors. where the number of ancestors of any node is at most n (since this is the maximum depth in the tree). On analogy to Section 5, the total number of nodes that have been expanded in the search is less than $n2^k$. But we have stated that the algorithm has been run at least $n2^k$ times. Therefore, we have an error reduction of at least 2^{-k}, as desired. ■

If the algorithm halts in this time, we have an exact solution, and so our error rate is zero. So if we allow early stopping in our search we can guarantee that the error times the number of times the search iterates is $\mathcal{O}(n(p_G(s)/p_G(\hat{a}(s))))$. Due to the cubic run time of the parsing algorithm, we find:

Lemma 6.2 *If we allow early stopping as described above, the total run time multiplied by the algorithm error will be* $\mathcal{O}((p_G(s)/p_G(\hat{a}(s))n^4))$.

As in Section 5, we can see that what we need to find is a bound on $p_G(s)/p_G(t(s))$.

7 Bounding $p_G(s)/p_G(t(s))$

Here we will find a bound on $p_G(s)/p_G(t(s))$ given only the grammar G. We can consider G as a model that probabilistically generates trees t, which in turn generate the yield s.

We know $p_G(t) \leq p_G(t(s))$, where $t(s)$ is the most likely tree that has the yield s. So, by the previous arguments, $p_G(t)$ is a lower bound for $M_G(s)$, and so $p_G(s)/p_G(t(s)) \leq p_G(s)/p_G(t)$. We will relate these values to the per-word sentence and tree entropies $H_{G,n}(s)$ and $H_{G,n}(t)$, where n is fixed.

If the the generative process for trees in G is ergodic and stationary, then as the size of the tree t increases, the density of the nonterminals in t approaches some distribution π_t. Furthermore, as the density of the words in s approach some distribution π_s with high probability, then for any $\delta, \varepsilon > 0$, there is an $N > 0$ such that if $n > N$, then, with probability at least $1 - \delta$, $\log p_G(s) < -n(H_{G,n}(s) - \varepsilon/2)$ and $\log p_G(t) > -n(H_{G,n}(t) + \varepsilon/2)$. In this case:

$$
\begin{aligned}
p_G(s)/p_G(t(s)) &\leq p_G(s)/p_G(t) \\
&= 2^{\log(p_G(s)) - \log(p_G(t))} \\
&< 2^{-n(H_{G,n}(s)-\varepsilon/2)+n(H_{G,n}(t)+\varepsilon/2)} \\
&= 2^{n(H_{G,n}(t)-H_{G,n}(s)+\varepsilon)}.
\end{aligned}
$$

Since s is completely determined by the tree t, we have that $H_G(s) = H_G(s,t)$, and so:

$$
\begin{aligned}
2^{n(H_{G,n}(t)-H_{G,n}(s)+\varepsilon)} &= 2^{n(H_{G,n}(t)-H_{G,n}(s,t)+\varepsilon)} \\
&= 2^{n(H_{G,n}(t|s)+\varepsilon)}
\end{aligned}
$$

Thus:

Theorem 7.1 *If the grammar G, taken as a generative process, is ergodic and stationary, then for any* $\varepsilon, \delta > 0$ *there is a number* n *such that, for any tree with at least* n *leaves:*

$$\log\left(p_G(s)/p_G(t(s))\right) \leq n(H_{G,n}(t|s) + \varepsilon)$$

with probability greater than $1 - \delta$.

Even if the grammar is not stationary and ergodic, we can ask what happens when we hold the grammar fixed and draw sentences from it, as is the case in typical NLP applications. In this case, the expected value of $log(p_G(s)/p_G(t))$ is still the conditional entropy $H_G(t|s)$ – we just cannot fix the sentence size. This value is not guaranteed to be an upper bound for the running time of the algorithm, or even for the logarithm of the expected value of $(p_G(s)/p_G(t))$.

Instead, we can look at the second moment $\nu_G(t|s)$ of $log(p_G(s)/p_G(t))$ in order to bound the overall likelihood of encountering a high-running-time input: For any $\delta > 0$, we can find a c such that the probability of drawing an s such that $log(p_G(s)/p_G(t)) > H_G(t|s) + c\nu_G(t|s)$ is less than δ. Thus:

Theorem 7.2 *Suppose we hold G constant and draw trees t from it, and that s is the yield of t. Then, for any* $\delta > 0$ *there is a c such that:*

$$p_G(s)/p_G(t(s)) < 2^{H_G(t|s)+c\nu_G(t|s)}$$

with probability greater than $1 - \delta$.

This bound unfortunately does not tell us how the problem difficulty scales with the sentence length n, but it does give us a sense of how difficult the MPS problem will be for a given grammar. While it is true that finding the exact values for either $H_G(t|s)$ or $\nu_G(t|s)$ is undecidable for general pCFGs, these values are can be given rough bounds – for example $H_G(t|s)$ is bounded above by the tree entropy $H_{G,n}(t)$. Finding better bounds is clearly an area for future research. Further discussion of the variance of the tree entropy can be found in (Chi, 1999).

8 Other Applications

In this paper, we have only considered the MPS problem, but the same approach can be applied to other inference problems, such as those found in Bayesian networks or CRFs. In particular, suppose that we are performing inference on a graphical model in which:

- we are trying to assign values to a finite set S of hidden variables in order to maximize the probability p of an event E, based on a finite set of hidden variables,
- we can build partial solutions to our inference problem by assigning values to one or more of these variables,
- we can use the probability of E over the partial solutions, or an approximation thereof, as an A^* heuristic on these partial solutions,
- extending nodes can always proceed according to an order defined as a function of the input (but independently of the node being expanded or its score), and
- extending nodes in the A^* search will partition its score among the child nodes.

If all of these conditions are met, then the A^* search above will still solve the problem, and will admit the same analysis. If the probability of E with no constraints is $p_0(E)$ and the probability of E under the optimal constraints is $p^*(E)$, the ratio $p_0(E)/P^*(E)$ will fill the same role as the ratio $p_G(s)/p_G(t)$ in our above analysis, and can be used to create a similar entropic bound. The main difficulty lies in the time we take to find the probability $p(E)$ given a set of constraints. This time will be application-dependent: in our case it took $\mathcal{O}(n^3)$ time. Our current method will generalize to any task in which the $p(E)$ value can be efficiently found.

9 Conclusions

Our analysis tells us something about how a particular NP-complete problem – the MPS problem for pCFGs – relates to a specific property of grammars. Our worst case running time is $\mathcal{O}(n^4 2^{n(H_{G,n}(t|s)+\varepsilon)})$ with high probability. Slightly more promising is that, when we draw a sentence s, it will take at most $2^{H_G(t|s)+c\nu_G(t|s)}n^4$ steps with probability $1-\delta$, where ν is the second moment of the entropy and c depends on δ. If we perform early stopping on the search, the product of the running time with the error will have the same bound. More practically, if we can determine that the ratio $p_G(s)/p_G(\tau(s))$ is small, our algorithm will have a much quicker run time.

Even in the worst-case scenario, the problem complexity collapses if we can break a sentence into several smaller instances. While we cannot do so in the grammar given by Sima'an (2002), it is possible in many NLP grammars. Roark and Hollingshead (2008) showed, for example, that we can find clause boundaries in sentences generated by the Penn Treebank grammar both quickly and reliably. Future work should focus on formalizing this for the MPS problem.

References

Nathanael L Ackerman, Cameron E Freer, and Daniel M Roy. 2011. Noncomputable conditional distributions. In *Logic in Computer Science (LICS), 2011 26th Annual IEEE Symposium on*, pages 107–116. IEEE.

David Arthur, Bodo Manthey, and H Roglin. 2009. k-means has polynomial smoothed complexity. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 405–414. IEEE.

Avrim L Blum and Ronald L Rivest. 1993. Training a 3-node neural network is np-complete. In *Machine learning: From theory to applications*, pages 9–28. Springer.

John Carroll. 1994. Relating complexity to practical performance in parsing with wide-coverage unification grammars. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 287–294. Association for Computational Linguistics.

Zhiyi Chi. 1999. Statistical properties of probabilistic context-free grammars. *Computational Linguistics*, 25(1):131–160.

Gregory F Cooper. 1990. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence*, 42(2):393–405.

Eric Corlett and Gerald Penn. 2013. Why letter substitution puzzles are not hard to solve: A case study in entropy and probabilistic search-complexity. In *The 13th Meeting on the Mathematics of Language*, page 83.

Gábor Erdélyi, Lane A Hemaspaandra, Jorg Rothe, and Holger Spakowski. 2009. Generalized juntas and np-hard sets. *Theoretical Computer Science*, 410(38):3995–4000.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 228–235. Association for Computational Linguistics.

Joshua Goodman. 1998. Parsing inside-out. *arXiv preprint cmp-lg/9805007*.

Colin De la Higuera and Jose Oncina. 2013. The most probable string: an algorithmic study. *Journal of Logic and Computation*, 24(2):311–330.

Russell Impagliazzo. 1995. A personal view of average-case complexity. In *Proceedings of Tenth Annual IEEE*, pages 134–147.

Leonid A Levin. 1986. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286.

Malte Nuhn and Hermann Ney. 2013. Decipherment complexity in 1: 1 substitution ciphers. In *ACL (1)*, pages 615–621.

Brian Roark and Kristy Hollingshead. 2008. Classifying chart cells for quadratic complexity context-free inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 745–752.

Dan Roth. 1996. On the hardness of approximate reasoning. *Artificial Intelligence*, 82:273–302.

Khalil Sima'an. 2002. Computational complexity of probabilistic disambiguation. *Grammars*, 5(2):125–151.

Daniel A Spielman and Shang-Hua Teng. 2004. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463.

Probing the Probing Paradigm:
Does Probing Accuracy Entail Task Relevance?

Abhilasha Ravichander[1] **Yonatan Belinkov**[2*] **Eduard Hovy**[1]
[1]Language Technologies Institute, Carnegie Mellon University
[2]Technion – Israel Institute of Technology
`aravicha@cs.cmu.edu`
`belinkov@technion.ac.il`, `hovy@cmu.edu`

Abstract

Although neural models have achieved impressive results on several NLP benchmarks, little is understood about the mechanisms they use to perform language tasks. Thus, much recent attention has been devoted to analyzing the sentence representations learned by neural encoders, through the lens of 'probing' tasks. However, to what extent was the information encoded in sentence representations, as discovered through a probe, actually used by the model to perform its task? In this work, we examine this probing paradigm through a case study in Natural Language Inference, showing that models can learn to encode linguistic properties even if they are not needed for the task on which the model was trained. We further identify that pretrained word embeddings play a considerable role in encoding these properties rather than the training task itself, highlighting the importance of careful controls when designing probing experiments. Finally, through a set of controlled synthetic tasks, we demonstrate models can encode these properties considerably above chance-level *even when distributed in the data as random noise*, calling into question the interpretation of absolute claims on probing tasks.[1]

1 Introduction

Neural models have established state-of-the-art performance on several NLP benchmarks (Kim, 2014; Seo et al., 2017; Chen et al., 2017; Devlin et al., 2019). However, these models can be opaque and difficult to interpret, posing barriers to widespread adoption and deployment in safety-critical or user-facing settings (Belinkov and Glass, 2019). How can we know what information, if any, neural models learn and leverage to perform a task? This question has spurred considerable community effort to develop methods to analyze neural models, motivated by interest not just to have models perform tasks well, but also to understand the mechanisms by which they operate.

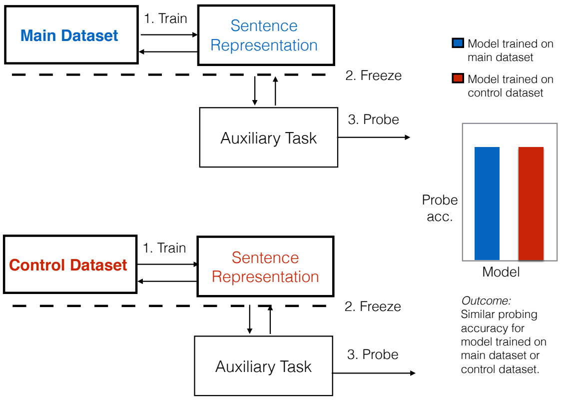


Figure 1: Illustration of our control dataset methodology for evaluating probing classifiers. Control datasets are constructed such that a linguistic feature is not discriminative with respect to the task. Representations from models trained on the main dataset and control dataset are probed for the linguistic feature, and demonstrate similiar probing performance.

A popular approach to model introspection is to associate the representations learned by the neural network with linguistic properties of interest, and examine the extent to which these properties can be recovered from the representation (Adi et al., 2017). This paradigm has alternatively been called probing (Conneau et al., 2018), auxiliary prediction tasks (Adi et al., 2017) and diagnostic classification (Veldhoen et al., 2016; Hupkes et al., 2018). As an example of this approach, let us walk through an application to analyze information about tense stored in a Natural Language Inference (NLI) model. In Conneau et al. (2018), three sentence-encoder models are trained on a

---

[*] Supported by the Viterbi Fellowship in the Center for Computer Engineering at the Technion.

[1] Code and data available at `https://github.com/AbhilashaRavichander/probing-probing`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3363–3377
April 19 - 23, 2021.

NLI dataset (MultiNLI; Williams et al., 2018). The encoder weights are frozen, and the encoders are then used to form sentence representations for the auxiliary task— predicting the tense of the verb in the main clause of the sentence. A separate classifier, henceforth called the probing classifier, is trained to predict this property based on the constructed representation. The probing task itself is typically selected to be relevant to the training task, and high probing performance is considered as evidence that the property is encoded in the learned representation. Due to its simplicity, a growing body of work uses this approach to pinpoint the information models rely on to do a task (Alt et al., 2020; Giulianelli et al., 2018; Saleh et al., 2020).

In this work, we examine the connection between the information encoded in a representation and the information a model relies on. Through a set of carefully designed experiments on the benchmark SentEval probing framework (Conneau et al., 2018), we shed light on information use in neural models. Our story unfolds in four parts:

1. First, we establish careful control versions of the training task such that task performance is invariant to a chosen linguistic property (Figure 1). We show that even when models cannot use a linguistic property to perform the task, the property can be reliably recovered from the neural representations through probing (§4.1).

2. Word embeddings could be a natural suspect for this discrepancy. We demonstrate that initializing models with pretrained word embeddings does play a role in encoding some linguistic properties in sentence representations. We speculate that probing experiments with pretrained word embeddings conflate two tasks — training word embeddings and the main task under consideration (§4.2).

3. What happens if we neutralize the effect of pre-trained word embeddings? Even when word embeddings are trained from scratch, we demonstrate that models still encode linguistic properties when they are not actually required for a task (§4.3).

4. Finally, through a carefully controlled synthetic scenario we demonstrate that neural models can encode information incidentally, even if it is distributed as random noise with respect to the training task (§5). We discuss several considerations when interpreting the results of probing experiments and highlight avenues for future research needed in this important area of understanding models, tasks and datasets (§6).

2 Background and Related Work

Progress in Natural Language Understanding (NLU) has been driven by a history of defining tasks and corresponding benchmarks for the community (Marcus et al., 1993; Dagan et al., 2006; Rajpurkar et al., 2016). These tasks are often tied to specific practical applications, or to developing models demonstrating competencies that transfer across applications. The corresponding benchmark datasets are utilized as proxies for the tasks themselves. How can we estimate their quality as proxies? While annotation artifacts are one facet that affects proxy-quality (Gururangan et al., 2018; Poliak et al., 2018; Kaushik and Lipton, 2018; Naik et al., 2018; Glockner et al., 2018), a dataset might simply not have coverage across competencies required for a task. Additionally, it might consist of alternate "explanations", features correlated with the task label in the dataset while not being task-relevant, which models can exploit to give the impression of good performance at the task itself.

Two analysis methods have emerged to address this limitation: 1) **Diagnostic examples**, where a small number of samples in a test set are annotated with linguistic phenomena of interest, and task accuracy is reported on these samples (Williams et al., 2018; Joshi et al., 2020). However, it is difficult to determine if models perform well on diagnostic examples because they actually learn the linguistic competency, or if they exploit spurious correlations in the data (McCoy et al., 2019; Gururangan et al., 2018; Poliak et al., 2018). 2) **External challenge tests** (Naik et al., 2018; Isabelle et al., 2017; Glockner et al., 2018; Ravichander et al., 2019; McCoy et al., 2019), where examples are constructed, either through automatic methods or by experts, exercising a specific phenomenon in isolation. However, it is challenging and expensive to build these evaluations, and non-trivial to isolate phenomena (Liu et al., 2019).

Thus, **probing** or *diagnostic classification* presents a compelling alternative, wherein learned representations can directly be probed for linguistic properties of interest (Ettinger et al., 2016; Belinkov et al., 2017; Adi et al., 2017; Tenney et al., 2019; Zhang and Bowman, 2018; Warstadt et al., 2019). There has been a variety of research that employs probing to test hypotheses about the mechanisms models used to perform tasks. Shi et al. (2016) examine learned representations in machine translation for syntactic knowledge. Vanmassenhove et al. (2017) investigate aspect in neural machine translation systems, finding that tense information could be extracted from the encoder, but that part of this information may be lost when decoding. Conneau et al. (2018) use probing to examine the correlation between linguistic properties and downstream tasks (including MT and NLI). Hupkes et al. (2018) train a 'diagnostic classifier' to extract information from a sequence of hidden representations in a neural network. If the classifier achieves high accuracy, it is concluded that the network is keeping track of the hypothesized information. Giulianelli et al. (2018) use diagnostic classifiers to predict number from the internal states of a language model. Kim et al. (2019) study what different NLP tasks teach models about function word comprehension. Alt et al. (2020) analyze learned representations in relation extraction, through a set of fourteen probing tasks for relevant linguistic properties. Saleh et al. (2020) examine the representations learned by neural dialog models for insights into what the model learns about engaging in dialog. See the survey by Belinkov and Glass (2019) for many more examples.

Closely related to our work is that of Hewitt and Liang (2019), which studies the role of lexical memorization in probing, and recently the work of Pimentel et al. (2020) and Voita and Titov (2020) who analyze probing from an information-theoretic perspective. These works join an ongoing debate on the correct way to characterize the expressivity of the probing classifier, with the latter proposing ease of extractability as a criterion for selecting appropriate probes. Our work pursues an orthogonal line of inquiry, demonstrating that relying on diagnostic classifiers to interpret model reasoning for a task suffers from a fundamental limitation: properties may be incidentally encoded even when not required for a task. Thus, our work is also related to a broader investigation of how neural models encode information (Tishby and Zaslavsky, 2015; Voita et al., 2019), studying to what extent information encoded in neural representations is indicative of information needed to perform tasks.

3 Methodology

In this section we describe our modified probing pipeline (Figure 1), where we construct control datasets, such that a particular linguistic feature is not required in making task judgements.[2] Control datasets are based on the intuition that a linguistic feature is not informative for a model to discriminate between classes if the linguistic feature remains constant across classes. For a task label T and linguistic property L, when every example in the control dataset has the same value for L, the linguistic property L in isolation is not discriminative of the task label .

To construct control datasets we hold constant the relevant property value across the whole dataset. In practice, the control datasets are constructed from existing large-scale datasets by partitioning them on the value of a linguistic property, such that every example in the sampled dataset has the same value of linguistic property.[3] They are designed with the following considerations:

1. The linguistic property of interest is auxiliary to the main task and a function of the input, but not of the task decision.
2. Every sample in the training and test sets has the same fixed value of the linguistic property.
3. The training set is large in order to train parameter-rich neural classifiers for the task.

We next describe our main training task, our three auxiliary prediction tasks, and procedures to construct control datasets corresponding to each auxiliary property. Models are trained either on datasets constructed for the main task, or on control datasets, and then probed for the auxiliary property using data from a probing dataset. In this work, we use the experimental settings of Conneau et al. (2018) for both the training task and probing task, due to its popularity as a probing benchmark. However, the conclusions we draw are meant to illustrate the limits and generality of probing as a diagnostic method, rather than discuss the specific experimental settings of Conneau et al. (2018).

Main Task: We study the Natural Language Inference (NLI) training task from Conneau et al. (2018) as the main task for training sentence en-

---

[2] While our motivating example of a task is natural language inference, we expect control datasets can be constructed for most text classification tasks with a small finite label space.

[3] All probing tasks in this work take a sentence representation as input and perform mappings to binary labels $\{0, 1\}$.

Linguistic Control	Property	# Train	# Test
MultiNLI	-	392,702	20,000
Tense	Past	69,652	1678
Subject Number	Singular	102,452	2584
Object Number	Singular	43,178	1060

Table 1: Statistics of control datasets partitioned by linguistic property.

coders. NLI is a benchmark task for research on natural language understanding (Cooper et al., 1996; Haghighi et al., 2005; Harabagiu and Hickl, 2006; Dagan et al., 2006; Giampiccolo et al., 2007; Zanzotto et al., 2006; MacCartney, 2009; Dagan et al., 2010; Marelli et al., 2014). Broadly, the goal of the task is to decide if a given hypothesis can be inferred from a premise in a justifiable manner. Typically, this is framed as the 3-way decision of whether a hypothesis is true given the premise (entailment), false given the premise (contradiction), or whether the truth value cannot be determined (neutral). We use MultiNLI (Williams et al., 2018), a broad-coverage NLI dataset, to train sentence encoders.

Auxiliary Tasks: We consider three tasks that probe sentence representations for semantic information from Conneau et al. (2018), all of which "require some understanding of what the sentence denotes". We construct the probing datasets such that lexical items that are associated with the probing task do not occur across the train/dev/test split for the target. This design controls for the effect of memorizing word types associated with target categories (Hewitt and Liang, 2019). The tasks considered in this study are:

1. TENSE: Categorize sentences based on the tense of the main verb.
2. SUBJECT NUMBER: Categorize sentences based on the number of the subject of the main clause.
3. OBJECT NUMBER: Categorize sentences based on number of the direct object of the main clause.

Control: For each auxiliary task, we partition MultiNLI such that premises and hypotheses agree on a single value of the linguistic property. For example, for the auxiliary task TENSE, sentences with VBP/VBZ/VBG forms are labeled as present and VBD/VBN as past tense.[4] Subsequently, premise-hypothesis pairs where the main verbs in both premise and hypothesis are in past tense are extracted from train/dev sets to form the control datasets for tense. Thus, every sentence in the dataset (both premises and hypotheses), has the same value of the auxiliary property.[5]

This procedure results in three control datasets/tasks: MultiNLI-PastTense, MultiNLI-SingularSubject, and MultiNLI-SingularObject. For all three, we fix the value of the linguistic property to the one that results in the maximum number of training instances on partitioning, namely fixing past tense, singular subject number, and singular object number. Descriptive statistics for each dataset appears in Table 1.

Models: We use CBOW and BiLSTM-based sentence-encoder architectures. The choice of these models is motivated by their demonstrated utility as NLI architectures (Williams et al., 2018), and because their learned representations have been extensively studied for the three linguistic properties used in this work (Conneau et al., 2017).[6]

1. Majority: The hypothetical performance of a classifier that always predicts the most frequent label in the test set.

2. CBOW: A simple Continuous Bag-Of-Words Model (CBOW). The sentence representation is the sum of word embeddings of constituent words. Word embeddings are finetuned during training.

3. BiLSTM-Last/Avg/Max: For a sequence of N words in a sentence $s = w_1 ... w_n$, the bidirectional LSTM (BiLSTM; Hochreiter and Schmidhuber (1997)) computes N vectors extracted from its hidden states $\vec{h}_1, ..., \vec{h}_n$. We produce fixed-length vector representations in three ways: by selecting the last hidden state h_n (BiLSTM-Last), by averaging the produced hidden states (BiLSTM-Avg) or by selecting the maximum value for each dimension in the hidden units (BiLSTM-Max).

---

[4] These heuristics are specific to English, as is MultiNLI. We use the Stanford Parser for constituency, POS and dependency parsing (Manning et al., 2014).

[5] This procedure replicates the original SentEval probing labels (Conneau et al., 2018) with 89.37% accuracy on tense, 87.77% accuracy on subject number and 88.19% accuracy on object number.

[6] We leave an exploration of recent transformer-based architectures to future work, noting however that this study stands alone as evidence that probing performance does not correlate to task importance.

	Tense		Subject Number		Object Number	
	Dev-ST	Probing	Dev-SS	Probing	Dev-SO	Probing
Majority	37.90	50.00	36.88	50.0	39.52	50.0
CBOW-DS	57.57	82.36	58.4	76.55	55.85	75.49
CBOW-PT	60.31	82.2	58.2	75.69	59.15	74.38
BiLSTM-Av-DS	63.53	82.93	64.24	79.53	66.23	76.11
BiLSTM-Av-PT	65.08	82.79	66.76	78.81	67.08	75.48
BiLSTM-Max-DS	63.35	81.14	65.91	78.56	65.94	74.79
BiLSTM-Max-PT	64.6	81.04	66.87	79.51	66.98	72.44
BiLSTM-Last-DS	61.08	80.43	64.2	81.52	62.26	72.65
BiLSTM-Last-PT	63.89	78.44	66.18	78.9	66.04	72.82

Table 2: Performance comparisons of task-controlled (PT) and downsampled models (DS). Dev-ST, Dev-SS and Dev-SO is the MultiNLI development set controlled for tense, subject number and object number, respectively. PT is a model trained on data partitioned by linguistic property—these models should not be able to leverage the linguistic property to perform their training task. DS is models trained on downsampled MNLI data to match the number of instances in partitioned. Majority baseline reflects distribution of main task classes for controlled development sets (Dev-ST, Dev-SS and Dev-SO), or class distribution of auxiliary property for probing datasets. We can observe that models consistently display similar probing accuracies whether the property was needed for the training task or not (Probing). Competitive performance of PT model variants to DS model variants on controlled MNLI development sets (Dev-ST, Dev-SS, Dev-SO) validates the controlled linguistic property is not useful to solve the controlled version of the task.

All models produce separate sentence vectors for the premise and hypothesis. They are concatenated with their element-wise product and difference (Mou et al., 2016), passed to a `tanh` layer and then to a 3-way softmax classifier. Models are initialized with 300D GloVe embeddings (Pennington et al., 2014) unless specified otherwise, and implemented in Dynet (Neubig et al., 2017). After the model is trained for the NLI task, the learned sentence vectors for the premise and hypothesis are probed. The probing classifier is a 1-layer multilayered perceptron (MLP) with 200 hidden units.

4 Probing the Probing Paradigm

4.1 Probing with Linguistic Controls

As a first step, we ask the question: to what extent is the information encoded in learned representations, as reflected in probing accuracies, driven by information that is useful for the training task? We construct multiple versions of the task (both training and development sets) where the entailment decision is independent of the given linguistic property, through careful partitioning as described in §3. To control for the effect of training data size, we downsample MultiNLI training data to match the number of samples in each partitioned version of the task. These results are in Table 2.

Strikingly, we observe that even when models are trained on tasks that *do not require the linguistic property at all* for the main task (rows with PT in Table 2), probing classifiers still exhibit high accuracy (sometimes up to ∼80%). Probing data is split lexically by target across partitions, and thus lexical memorization (Hewitt and Liang, 2019) cannot explain why these properties are encoded in the sentence representations. Across models, on the version of the task where a particular linguistic property is not needed, classifiers trained on data that does not require that property perform comparably to classifiers trained on MultiNLI training data (DS vs PT models, on Dev-ST, Dev-SS, and Dev-SO).

4.2 Effect of Word Embeddings

A potential explanation lies in our definition of a "task". Previous work directly probes models trained for a target task such as NLI. However, when models are initialized with pre-trained word embeddings, the conflated results of two tasks are being probed – the main training task of interest, and the task that was used to train the word embeddings. Both tasks may contribute to the encoding of information in the learned representation, and it is unclear to what extent they interact. Previous work has noted the considerable amount of information

	Tense		Subject Number		Object Number	
	Dev	Probing	Dev	Probing	Dev	Probing
Majority	36.50	50.0	36.50	50.0	36.50	50.0
CBOW-Word	62.21	83.74	62.1	76.91	61.93	75.4
CBOW-Rand	56.98	60.14	56.27	67.01	56.82	64.71
BiLSTM-Avg-Word	70.05	82.48	70.67	76.53	69.82	72.29
BiLSTM-Avg-Rand	63.33	61.4	64.0	67.68	63.71	63.87
BiLSTM-Max-Word	68.67	78.34	69.19	73.96	69.12	68.53
BiLSTM-Max-Rand	62.78	62.89	63.29	69.51	63.28	62.84
BiLSTM-Last-Word	68.32	74.61	69.04	71.82	68.82	69.27
BiLSTM-Last-Rand	62.14	62.96	61.88	67.45	62.29	61.32

Table 3: Performance comparisons of models initialized with pretrained word embeddings (Word) and models with randomly initialized embeddings (Rand) on MNLI Development Set (Dev) and on the probing task (Probing). Embeddings are updated during task-specific training. We can observe that probing performance decreases sharply for all models when word embeddings are randomly initialized, suggesting a considerable component of probing performance comes from pretraining word embeddings rather than what a model learns during the task.

present in word embeddings, and proposed methods to measure this effect, such as comparing with bag-of-word baselines or random encoders (Wieting and Kiela, 2018). However, these methods fail to isolate the contribution of the training task.

To study this, we compare models initialized with pre-trained word embeddings (Pennington et al., 2014) and then trained for the main task, to models initialized with random word embeddings and then updated during the main task. These results are presented in Table 3. We observe that probing accuracies drop across linguistic properties in this setting (compare rows with Word and Rand in the table), indicating that models with randomly initialized embeddings generate representations that contain less linguistic information than the models with pretrained embeddings. This result calls into question how to interpret the contribution of the main task to the encoding of a linguistic property, when the representation has already been initialized with pre-trained word embeddings. The word embeddings could themselves encode a significant amount of linguistic information, or the main task might contribute to encoding information in a way already largely captured by word embeddings.

4.3 How do models encode linguistic properties?

When we isolate the effect of the main task with randomly initialized word embeddings, are properties not predictive of the main task judgement *still* being encoded? To study this, we revisit our linguistic control tasks but train all models with randomly initialized word embeddings. We also train comparable models on downsampled MultiNLI training data. These results can be found in Table 4. We observe that even in the setting with randomly initialized word embeddings, these properties are still encoded to a similar extent (and above the majority baseline) in the downsampled and control versions of their task.

5 A Synthetic Experiment: Analyzing Encoding Dynamics

We have demonstrated that models encode properties even when they are not required for the main task. Thus, probing accuracy cannot be considered indicative of competencies any given model relies on. What circumstances could lead to models encoding properties incidentally? Can we determine when a linguistic property is not needed by a model for a task? To study this, we build carefully controlled synthetic tests, each capturing a kind of noise that could arise in datasets.

5.1 Synthetic Task

We consider a task where the premise P and hypothesis H are strings from $S = \{(a|b)(a|b|c)^*\}$ of maximum length 30, and the hypothesis H is said to be entailed by the premise P if it begins with the same letter a or b,[7] for example:

---

[7] A task with a similar objective was used by Belinkov et al. (2019a) to demonstrate unlearning bias in datasets. The task is equivalent to XOR, which is learnable by an MLP.

	Tense		Subject Number		Object Number	
	Dev-ST	Probing	Dev-SS	Probing	Dev-SO	Probing
Majority	37.90	50.0	36.88	50.0	39.52	50.0
CBOW-Rand-DS	49.88	61.33	51.04	67.32	49.25	63.63
CBOW-Rand-PT	53.28	61.37	50.97	67.02	52.45	63.84
BiLSTM-Avg-Rand-DS	57.21	63.75	60.76	68.5	59.53	63.89
BiLSTM-Avg-Rand-PT	60.91	63.07	61.18	69.12	60.57	63.77
BiLSTM-Max-Rand-DS	59.18	61.05	61.8	70.32	60.57	64.68
BiLSTM-Max-Rand-PT	60.55	61.53	63.78	70.6	63.49	64.26
BiLSTM-Last-Rand-DS	56.73	63.88	58.82	69.09	56.79	63.86
BiLSTM-Last-Rand-PT	57.39	62.88	61.88	68.8	60.75	61.96

Table 4: Performance of task-controlled (PT) and downsampled models (DS), when word embeddings are trained from scratch. (Rand) indicates the model is initialized with random embeddings, rather than pretrained embeddings. Dev-ST, Dev-SS and Dev-SO is the MultiNLI development set controlled for tense, subject number and object number, respectively. We observe that when the training task is isolated in this way, for all models probing performance is similar whether a linguistic property is necessary for the task or not (Probing).

(a, ab) → Entailed
(b, ba) → Entailed
(b, bc) → Entailed

(a, ba) → Not Entailed
(b, ab) → Not Entailed
(b, acb) → Not Entailed

Consider the auxiliary task of predicting whether a sentence contains the character c from a representation, analogous to probing for a task-irrelevant property. We sample premises/hypotheses from a set of strings $S' = (a|b)^*$ of maximum length 30, and simulate four kinds of correlations that could occur in a dataset by inserting c at a random position in the string after the first character:[8]

1. NOISE : The property could be distributed as noise in the training data. To simulate this, we insert c into 50% of randomly sampled premise and hypothesis strings.
2. UNCORRELATED : The property could be unrelated to the task decision, but correlated to some other property in the data. To simulate this, we insert c to premises beginning with a.
3. PARTIAL: The property could provide a partial explanation for the main task decision. To simulate this, we insert c to premise and hypothesis strings beginning with a.[9]

Dataset	# Train	# Dev	# Test
NOISE	20000	5000	5000
UNCORRELATED	20000	5000	5000
PARTIAL	20000	5000	5000
FULL	20000	5000	5000
PROBE	23732	5000	5000

Table 6: Descriptive statistics for NOISE, UNCORRELATED, PARTIAL and FULL synthetic datasets, as well as the dataset used to train the probing classifier(PROBE). We ensure that datasets do not have any data leakage in the form of strings appearing across train/dev/test splits, or across task and probing splits in either the main task or the probing dataset.

4. FULL: The property provides a complete alternate explanation for the main task decision. We insert c to premise and hypothesis strings whenever the hypothesis is entailed.

Descriptive statistics of all datasets are in Table 6.

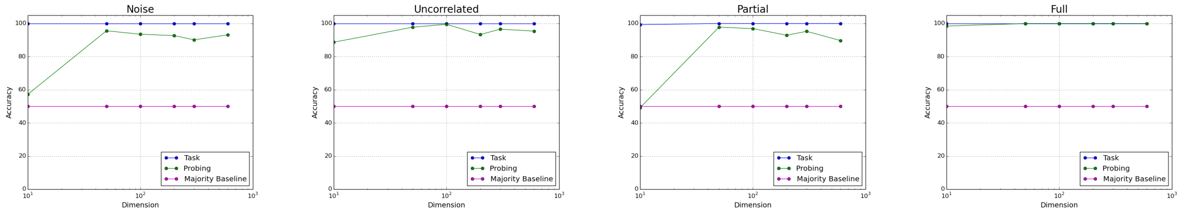
5.2 Results

Figure 2a presents the performance of the model and the probe on the four test sets. We observe that we are able to train a classifier to predict the presence of c considerably above chance-level in all four cases. This is notable, considering that even when the property is distributed as random noise (NOISE) uncorrelated with the actual task, the

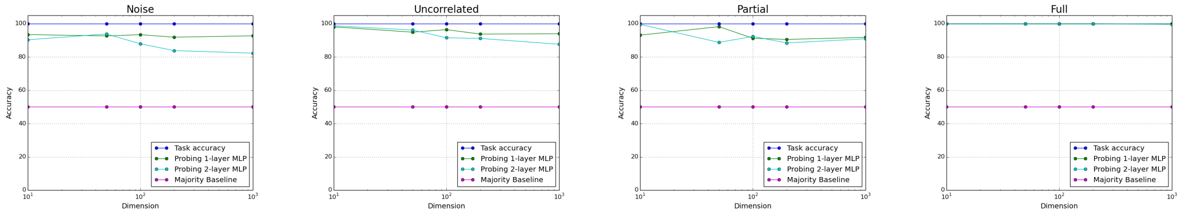
---

[8] We additionally explore the utility of adversarial learning, as a potential approach to identifying properties required by a model to perform a task, by suppressing a property and measuring task performance (Appendix. A). We find in our exploration that adversarial approaches are not completely successful at suppressing the linguistic property under consideration, though capacity of the adversary could play a role.

[9] Models can use either the presence of c, or the first character of the strings being a to make their prediction, but they must use whether the first character of the strings is b.



(a) Main Task and Probing Accuracy as a function of capacity of sentence representation (# units).



(b) Main Task and Probing Accuracy as a function of capacity of probing classifier (# units).

Figure 2: Task and probing accuracy of (BiLSTM, last) on Noise, Uncorrelated, Partial and Full synthetic datasets.

model encodes it. This simple synthetic task suggests that models learn to encode linguistic properties *incidentally*, implying it is a mistake to rely on the accuracy of probes to measure what information the model relies upon to solve a task. We further discuss the role of representation capacity and probing classifier expressivity:

Representation size: Lower-capacity models may encode task-specific information at the expense of irrelevant properties. To examine this, we train the BiLSTM architecture with hidden size 10, 50, 100, 200, 300 and 600 units, and train the probing classifier on the auxiliary task. These results are reported in Figure 2a. We observe that while the main task accuracy remains consistent across choice of dimension, probing accuracy decreases for models with lower capacity across categories. This suggests that the capacity of the representation may play a role in which information it encodes, with lower capacity models being less prone to incidentally encoding irrelevant information.

Probing classifier capacity: We examine whether probing classifier capacity is a factor in the incidental encoding of linguistic properties. A more complex probing classifier may be more effective at extracting linguistic properties from representations. We experiment with probing classifiers utilizing 1-layer and 2-layer MLP's of dimensions $\{10, 50, 100, 200, 1000\}$. The results are shown in Figure 2b. We find that a higher-capacity probing classifier does not necessarily imply higher probing accuracy. Further, in all the settings of probing classifier capacity we study, we are able to perform the auxiliary task considerably above chance accuracy, even when the property is distributed as random noise.

6 Discussion

We briefly discuss our findings, with the goal of providing considerations for deciding which inferences can be drawn from a probing study, and highlighting avenues for future research.

Linguistic properties can be incidentally encoded: Probing only indicates that some property correlated with a linguistic property of interest is encoded in the sentence representation — but we speculate that it cannot isolate what that property might be, whether the correlation is meaningful, or how many such properties exist. As shown in the controlled synthetic tests, even if a particular property is not needed for a task, the information can be extracted from the representation with high accuracy. Thus, probing cannot determine if the property is actually needed to do a task, and should not be used to pinpoint the information a model is relying upon. A negative result here can be more meaningful than a positive one. Adversarially suppressing the property may help determine if an alternate explanation is readily available to the model, with an appropriate choice of probing classifier. In this case, if the model maintains task accuracy while suppressing the information, one can conclude the property is not needed by the model for the task, but its failure to do so is not indicative of property importance. Causal alternatives to probing classifiers that intervene in model representations to examine effects on prediction also present another promising direction for future work (Giulianelli et al., 2018; Bau et al., 2018; Vig et al., 2020).

Careful controls and baselines: We emphasize the need for probing work to establish careful controls and baselines when reporting experimental results. When probing accuracy for a linguistic competence is high, it may not be directly attributable to the training task. In this work we identify two confounds: incidental encoding and interaction between training tasks. Perhaps future work will determine causes of incidental encoding and identify further baselines and controls that allow reliable conclusions to be drawn from probing studies.

Lack of gold-standard data of task requirements: While prior work has discussed the different linguistic competencies that might be needed for a task based on the results of probing studies, these claims are inherently hard to reliably quantify given that the exact linguistic competencies, as well as the extent to which they are required, are difficult to isolate for most real-world datasets. Controlled test cases (such as those in §5.1) are effective as basic sanity checks for claims based on diagnostic classification, and provide insight into encoding dynamics in sentence representations.

Datasets are proxies for tasks, and proxies are imperfect reflections: Finally, we speculate that while datasets are used as proxies for tasks, they might not reflect the full complexity of the task. Aside from having dataset-specific idiosyncrasies in the form of unwanted biases and correlations, they might also not require the full range of competencies that we expect models to need to succeed on the task. Future work should refine or move beyond the probing paradigm to carefully identify what the competencies reflected in any dataset are, and how representative they are of overall task requirements.

What probes are good for: This work explores only the implications of probing as a diagnostic tool for pinpointing the information models use to do a task. However, when sentence representations are used subsequently downstream (after being trained on the main task), probing can give insight into what information is encoded in the model (irrespective of how that encoding came to be). Future work could include exploring the connection between information encoded in the representation and whether models successfully learn to use them in downstream tasks.

7 Conclusion

The probing paradigm has evoked considerable interest as a useful tool for model interpretability. In this work, we examine the utility of probing for providing insights into what information models rely on to do tasks, and requirements for tasks themselves. We identify several considerations when probing sentence representations, most strikingly that linguistic properties can be incidentally encoded even when not needed for a main task. This line of questioning highlights several fruitful areas for future research: how to successfully identify the set of linguistic competencies necessary for a dataset, and consequently how well any dataset meets task requirements, how to reliably identify the exact information models rely upon to make predictions, and how to draw connections between information encoded by a model and used by a model downstream.

Acknowledgements

This research was supported in part by grants from the National Science Foundation Secure and Trustworthy Computing program (CNS-1330596, CNS15-13957, CNS-1801316, CNS-1914486) and a DARPA Brandeis grant (FA8750-15-2-0277). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the NSF, DARPA, or the US Government. This research was also supported by the ISRAEL SCIENCE FOUNDATION (grant No. 448/20). Y.B. was also supported by the Harvard Mind, Brain, and Behavior Initiative. The authors would like to extend special gratitude to Carolyn Rose and Aakanksha Naik, for insightful discussions related to this work. The authors are also grateful to Yanai Elazar, Lucio Dery, Paul Michel, Shruti Rijhwani and Siddharth Dalmia for reviews while drafting this paper, and to Marco Baroni for answering questions about the SentEval probing tasks.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *International Conference on Learning Representations*.

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. Probing linguistic features of sentence-level representations in neural relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1534–1545, Online. Association for Computational Linguistics.

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Bolei Zhou, Joshua B Tenenbaum, William T Freeman, and Antonio Torralba. 2018. Gan dissection: Visualizing and understanding generative adversarial networks. In *International Conference on Learning Representations*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019a. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 877–891, Florence, Italy. Association for Computational Linguistics.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019b. On adversarial removal of hypothesis-only bias in natural language inference. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 256–262, Minneapolis, Minnesota. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Robin Cooper, Dick Crouch, Jan Van Eijck, Chris Fox, Johan Van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, et al. 1996. Using the framework. Technical report.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. The fourth pascal recognizing textual entailment challenge. *Journal of Natural Language Engineering*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

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

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 134–139.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1180–1189. JMLR. org.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Aria Haghighi, Andrew Ng, and Christopher Manning. 2005. Robust textual inference via graph matching. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 905–912. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Pierre Isabelle, Colin Cherry, and George Foster. 2017. A challenge set approach to evaluating machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2486–2496, Copenhagen, Denmark. Association for Computational Linguistics.

Pratik Joshi, Somak Aditya, Aalok Sathe, and Monojit Choudhury. 2020. TaxiNLI: Taking a ride up the NLU hill. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 41–55, Online. Association for Computational Linguistics.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Najoung Kim, Roma Patel, Adam Poliak, Patrick Xia, Alex Wang, Tom McCoy, Ian Tenney, Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, and Ellie Pavlick. 2019. Probing what different NLP tasks teach machines about function word comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Nelson F Liu, Roy Schwartz, and Noah A Smith. 2019. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179.

Bill MacCartney. 2009. *Natural language inference*. Stanford University.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 130–136, Berlin, Germany. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, et al. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

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

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. 2019. EQUATE: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361, Hong Kong, China. Association for Computational Linguistics.

Abdelrhman Saleh, Tovly Deutsch, Stephen Casper, Yonatan Belinkov, and Stuart Shieber. 2020. Probing neural dialog models for conversational understanding. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 132–143, Online. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

Eva Vanmassenhove, Jinhua Du, and Andy Way. 2017. Investigating ‘aspect’in nmt and smt: Translating the english simple past and present perfect. *Computational Linguistics in the Netherlands Journal*, 7:109–128.

Sara Veldhoen, Dieuwke Hupkes, and Willem H Zuidema. 2016. Diagnostic classifiers revealing how neural networks process hierarchical structure. In *CoCo@ NIPS*.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Causal mediation analysis for interpreting neural nlp: The case of gender bias. *arXiv preprint arXiv:2004.12265*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International*

Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4387–4397.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2870–2880, Hong Kong, China. Association for Computational Linguistics.

John Wieting and Douwe Kiela. 2018. No training required: Exploring random encoders for sentence classification. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

F Zanzotto, Alessandro Moschitti, Marco Pennacchiotti, and M Pazienza. 2006. Learning textual entailment from examples. In *Second PASCAL recognizing textual entailment challenge*, page 50. PASCAL.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361, Brussels, Belgium. Association for Computational Linguistics.

Dataset	# Train	# Dev	# Test
Noise	20000	5000	5000
Uncorrelated	20000	5000	5000
Partial	20000	5000	5000
Full	20000	5000	5000
Attacker	23732	5000	5000

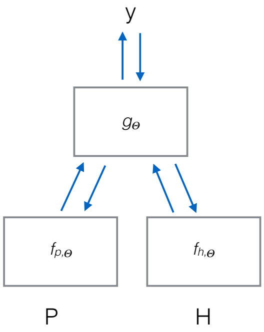
Table 7: Number of train/dev/test examples in constructed synthetic datasets.

A Adversarial Learning Framework

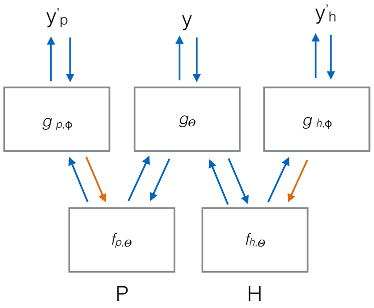
We explore an adversarial framework, as a potential approach to identifying incidentally-encoded properties. We study the utility of this framework within the controlled setting of the synthetic task described in Section 5, where a hypothesis H is entailed by a premise P, if they both begin with the same letter 'a' or 'b'.

We train an adversarial classifier to suppress task-irrelevant information, in this case the presence of 'c'. The goal is to analyze whether adversarial learning can help a model ignore this information while maintaining task performance. If the model succeeds, it indicates the model does not need the linguistic property for the task. Table **??** provides descriptive statistics for Noise, Uncorrelated, Partial and Full synthetic datasets, as well as the probing dataset used to train the external attack classifier. We ensure that datasets do not have any data leakage in the form of strings appearing across train/dev/test splits, or across task and probing splits in either the main task or the external held-out attacker dataset.

We follow the adversarial learning framework illustrated in Figure 3. In this setup, we have premise-hypothesis pairs $\langle p_1, h_1 \rangle ... \langle p_n, h_n \rangle$ and entailment labels $y_1 ... y_n$, as well as labels for linguistic properties in each premise–hypothesis pair $\langle z_{p,1}, z_{h,1} \rangle ... \langle z_{p,n}, z_{h,n} \rangle$. We would like to train sentence encoders f(p_i, θ) and f(h_i, θ) and a classification layer g_θ such that $y_i = g_\theta$ (f(p_i, θ), f(h_i, θ)), in a way that does not use $\langle z_{p,i}, z_{h,i} \rangle$. We do this by incorporating an *adversarial* classification layer g_ϕ such that $\langle z_{p,i}, z_{h,i} \rangle$ = $\langle g_\phi$(f(p_i, θ)), g_ϕ(f(h_i, θ)$\rangle$ (Goodfellow et al., 2014; Ganin and Lempitsky, 2015). Following Elazar and Goldberg (2018), we also have an external 'attacker' classifier ϕ' to predict $z_{p,i}$ and $z_{h,i}$ from the learned sentence representation.[10] A similar setup has been used by Belinkov et al. (2019b) to remove hypothesis-only biases from NLI models.



(a) Baseline



(b) Adversarial removal.

Figure 3: Illustration of (a) The baseline NLI task architecture, and (b) Adversarial removal of linguistic properties from the representations. Arrows represent direction of propagation of inputs in the forward pass and gradients in backpropagation. Blue and orange arrows correspond to the gradient being preserved and reversed respectively.

In training the adversarial classifier is trained to predict z from the sentence representations $f_\theta(p_i, h_i)$, and the sentence encoder f is trained to make the adversarial classifier unsuccessful at doing so. This is operationalized through the following training objectives optimized jointly:

$$\begin{aligned}\arg\min_{\phi} & \, L(g_\phi(f(p_i, \theta), z_{p,i})) \\ & + L(g_\phi(f(h_i, \theta), z_{h,i})) \quad (1)\end{aligned}$$

$$\begin{aligned}\arg\min_{f,\theta} & \, L(g_\theta(f_\theta(p_i, h_i)), y_i) - (L \\ & (g_\phi(f(p_i, \theta), z_{p,i}) + L(g_\phi(f(h_i, \theta), z_{h,i}))) \quad (2)\end{aligned}$$

where L is cross-entropy loss. The optimization is implemented through a Gradient Reversal Layer (Ganin and Lempitsky, 2015) g_λ which is placed between the sentence encoder and the adversarial classifier. It acts as an identity function in the forward pass, but during backpropogation scales the

---

[10] We train the attacker on a held-out dataset with the linguistic property distributed as random noise (Table **??**). We also ensure all examples in the attacker data are unseen in the main task, to prevent data leakage.

	Noise			Uncorrelated			Partial			Full		
	Dev	Adv.	Attack.	Dev	Adv.	Attack.	Dev	Adv.	Attack.	Dev	Adv.	Attack.
Majority	50.4	51.2	50.2	50.94	74.31	50.2	50.62	99.82	50.2	55.34	55.34	50.2
λ=0.0	100.0	-	90.3	100.0	-	93.6	100.0	-	91.08	100.0	-	100.0
λ=0.5	100.0	47.81	95.3	100.0	70.36	62.26	100.0	99.31	80.48	100.0	51.23	93.42
λ=1.0	100.0	49.43	94.5	100.0	71.28	74.1	100.0	99.79	68.8	100.0	52.37	92.58
λ=1.5	100.0	42.7	100.0	100.0	71.54	99.1	97.98	99.79	82.32	100.0	49.8	97.58
λ=2.0	100.0	46.19	99.36	100.0	70.62	99.98	100.0	94.83	91.12	100.0	40.94	94.64
λ=3.0	100.0	46.98	94.64	100.0	70.92	99.8	99.26	99.19	79.66	100.0	53.08	87.0
λ=5.0	99.98	38.87	96.92	99.94	71.0	86.6	100.0	98.73	100.0	100.0	51.32	98.74

Table 8: Adversarial performance on synthetic tasks: noise, uncorrelated, partial, full. Dev is accuracy of model on task, Adv. is accuracy of the adversarial classifier, Atttack. is accuracy of attacker classifier on held-out data.

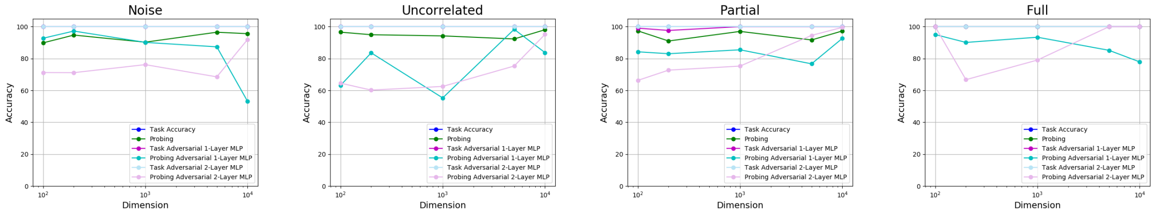


Figure 4: Main Task and Attacker Accuracy as a function of capacity of adversarial classifier for $\lambda = 0.0$ and $\lambda = 1.0$.

gradients by a factor $-\lambda$ [11], resulting in the objective:

$$\arg\min_{f,\theta} L(g_\theta(f_\theta(p_i,h_i)),y_i) + L(g_\phi(g_\lambda(f(p_i,\theta))),z_{p,i}) + L(g_\phi(g_\lambda(f(h_i,\theta))),z_{h,i}) \quad (3)$$

Implementation details : We implemented the adversarial model using the Dynet framework (Neubig et al., 2017), with a BiLSTM architecture of hidden dimension 200 units. Fixed length vector representations are constructed using the last hidden state and the model is trained for 10 epochs using early stopping. The attacker classifier is a 1-layer MLP with hidden size of 200 dimensions.

Results Table 8 reports the performance of the adversarial and attacker classifiers on the four test sets. We observe that information about irrelevant properties can be extracted by attackers even under adversarial suppression, consistent with the findings of Elazar and Goldberg (2018) and Belinkov et al. (2019b). In the case of random noise, we do not find any setting of adversary weight λ that suppresses the attribute. We further explore issues of adversarial classifier strength:

Adversarial classifier capacity: A more powerful adversarial classifier may be more effective at suppressing task-irrelevant information. To examine this, we fix the attacker classifier and experiment with adversarial classifiers with 1-layer and 2-layer MLP probes and dimensions 100, 200, 1000, 5000 and 10000 units, as reported in Figure 4. We find that varying the capacity of the adversarial classifier can decrease the attacker accuracy, though the choice of capacity depends on the setup used.

Considerations: 1) In synthetic tests, the main task function is learnable by a neural network. However, in practice for most NLP datasets this might not be true, making it difficult for models to reach comparable task performance while suppressing correlated linguistic properties. 2) Information might be encoded, but may still not be recoverable by the choice of probing classifier[12]. 3) Adversarial learning does not remove all information from the representation (Elazar and Goldberg, 2018). 4) If comparable task accuracy can't be reached, one cannot conclude a property is not relevant.[13]

---

[11] λ controls the extent to which we try to suppress the property.

[12] All claims related to probing task accuracy, as in most prior work, are with respect to the probing classifier used.

[13] This could be because the main task might be more complex to learn or unlearnable, or multiple alternate confounds could be present in data which are not representative of the decision-making needed for the main task.

One-class Text Classification
with Multi-modal Deep Support Vector Data Description

Chenlong Hu[1], **Yukun Feng**[1], **Hidetaka Kamigaito**[1], **Hiroya Takamura**[1,2]
and **Manabu Okumura**[1]
[1]Tokyo Institute of Technology
[2]National Institute of Advanced Industrial Science and Technology (AIST)
{huchenlong,yukun,kamigaito,oku}@lr.pi.titech.ac.jp
takamura@pi.titech.ac.jp

Abstract

This work presents *multi-modal deep SVDD (mSVDD)* for one-class text classification. By extending the uni-modal SVDD to a multiple modal one, we build *mSVDD* with multiple hyperspheres, that enable us to build a much better description for target one-class data. Additionally, the end-to-end architecture of *mSVDD* can jointly handle neural feature learning and one-class text learning. We also introduce a mechanism for incorporating negative supervision in the absence of real negative data, which can be beneficial to the *mSVDD* model. We conduct experiments on *Reuters* and *20 Newsgroup* datasets, and the experimental results demonstrate that *mSVDD* outperforms uni-modal SVDD and *mSVDD* can get further improvements when negative supervision is incorporated.

1 Introduction

One-Class Classification (OCC), a special classification problem, aims to learn a model on the basis of training samples only from *one* class. The learned model is expected to make an accurate description of the class (so called *target* or *normal*) and then to distinguish the *target* from samples for negative classes during testing (Moya et al., 1993; Tax, 2002). The one-class classification problem has arisen in many real-world applications, including anomaly or novelty detection (Roberts, 1999; Chandola et al., 2010; Gupta et al., 2013), bioinformatics (Alashwal et al., 2006), and especially computer vision (Rodner et al., 2011; Ruff et al., 2018).

One-class text classification would be beneficial to the scenario where anomalous text contents (e.g., web pages, spam emails) (Yu et al., 2004) need to be detected, and only a positive training corpus is available. One of the early work on one-class text classification is Manevitz and Yousef (2001), who implemented versions of one-class support vector machines (OC-SVM) (Schölkopf et al., 2001) and showed good performances over the *Reuters* dataset (Dumais et al., 1998). OC-SVM and support vector data description (SVDD) (Tax and Duin, 2004) are *boundary*-based methods (Tax, 2002). Both try to describe the target data using a boundary. SVDD learns an optimal hypersphere with the minimum radius to include the most target data, while OC-SVM builds a hyperplane to maximally separate the data points from the origin where outlier examples lie around.

Reconstruction-based approaches, including AutoEncoder (Jacobs, 1995) and principal component analysis (PCA) (Bishop, 1995), which aim to learn a more compact representation for the description of target data. The compact representation could be a set of prototypes or subspaces obtained by optimizing a reconstruction error on the target training data.

Regarding the features for representing text in OCC, document-to-word co-occurrence matrices or hand-crafted features have been commonly used in most of the previous work (Manevitz and Yousef, 2001, 2007; Kumaraswamy et al., 2015). Pretrained vectors have been popular for many NLP tasks (Mikolov et al., 2013; Bengio et al., 2003). The recent *context vector data description (CVDD)*, proposed by Ruff et al. (2019), fully uses word embedding knowledge and a neural network structure to process one-class classification problems.

Ruff et al. (2018) introduced *deep support vector data description (deep SVDD)*, a fully unsupervised method for deep one-class classification for image data. *Deep SVDD* learns to extract the common factors of target training samples with a neural network to minimize the radius of a hypersphere that encloses the network representations of the data. The learned hypersphere, with a center $\boldsymbol{c}$ and a neural feature transformer $\phi(\boldsymbol{x})$, can be an end-to-end

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3378–3390
April 19 - 23, 2021.

feature learning and one-class classification model.

Target data samples may have distinctive distributions that are located in different regions. Therefore, uni-modal *deep SVDD* with one hypersphere may not be enough to describe the target samples. In this work, we extend *deep SVDD* to multiple modes, where each mode describes the target samples from a distinctive aspect. Given our *multi-modal deep SVDD*, *mSVDD* in short, we can create an ensemble set of hyperspheres with different centers to build a better one-class model. Ghafoori and Leckie (2020) proposed *deep multi-sphere SVDD* (*DMSVDD*), a similar but different work from ours. We will also discuss the relationship between the two and compare them in the experiments.

In one-class classification, only samples from the target class are available for training, while the model needs to discriminate between the target class and other classes in testing. Due to the unavailability of training samples from negative classes, it is hard for the one-class models to learn effective discrimination information, especially for *mSVDD* with a multi-layer neural structure. In this study, we also propose an architecture for improving the discrimination ability of *mSVDD* by incorporating *negative supervision*. Specifically, we define two kinds of losses, *contrastive* and *triplet*, for joint training with the objective function of *mSVDD*, which is expected to enhance the discriminative power of *mSVDD*.

In summary, the main contributions of this work are as follows. 1) We propose a general one-class neural learning framework, called *mSVDD*, to extend the uni-modal *deep SVDD* to end-to-end multi-modal. 2) We also prove that three one-class models, *deep SVDD, DMSVDD*, and *CVDD*, are all special cases of the *mSVDD* model. 3) We propose two approaches for effectively incorporating negative supervision information to improve the performance of the proposed *mSVDD*.

2 Preliminaries

Before describing our *mSVDD*, we first introduce SVDD (Tax and Duin, 2004) and its extension, *deep SVDD* (Ruff et al., 2018).

2.1 SVDD

SVDD is a support vector learning method for one-class classification. It aims at constructing an optimal boundary in a feature space that includes almost all normal target data, given only the target training samples, $T = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$, $\boldsymbol{x}_i \in \mathcal{X}$, where $n \in \mathbb{N}$ is the size of the training data, and $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$. The main idea of SVDD is to optimize a hypersphere with a center $\boldsymbol{c}$ and radius R, that encloses the majority of the data. SVDD solves the following quadratic problem:

$$\min_{R,\boldsymbol{c},\boldsymbol{\xi}} \quad R^2 + C\sum_i \xi_i \tag{1}$$

$$s.t.\ \|\boldsymbol{x}_i - \boldsymbol{c}\|_2^2 \leq R^2 + \xi_i,\ \xi_i \geq 0, \quad \forall i = 1, ..., n,$$

where ξ_i is a slack variable for allowing a flexible boundary. C is a regularization parameter, that is usually represented by $\frac{1}{\nu n}$, where $\nu \in (0, 1]$ is a parameter that controls the tradeoff between the radius of the hypersphere and the penalties ξ_i.

Several efforts have been proposed to extend SVDD with multi-spheres. Hao and Lin (2007) was early work to use multi-sphere SVDD, which was used for multi-class tasks. For one-class tasks, (Xiao et al., 2009) used multi-sphere SVDD to encode multi-distribution target data. Two more efforts have been proposed by (Le et al., 2010, 2013), which found the optimal solution by an iterative algorithm consisting of the following two steps: 1) calculate radii and centers, and 2) calculate the assignments of data to centers.

While one limitation of SVDD, along with its extensions, would be that it has to perform hand-crafted feature engineering (Pal and Foody, 2010), the limitation could be solved by incorporating neural models into SVDD.

2.2 Deep SVDD

Deep SVDD (Ruff et al., 2018) is an end-to-end deep neural model that not only optimizes the SVDD objective loss but also learns a neural feature transformation. Given target training samples $T = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$, *deep SVDD* first transforms instance $\boldsymbol{x}$ into a data point of the output feature space with ϕ, which is a multi-layer neural network of $L \in \mathbb{N}$ layers with parameters $\mathcal{W} = \{\boldsymbol{W}^1, ..., \boldsymbol{W}^L\}$. *Deep SVDD* defines two kinds of loss functions:

Soft-boundary deep SVDD :

$$\frac{1}{\nu n}\sum_i max(0, \|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}\|_2^2 - R^2) \\ + R^2 + \frac{\lambda}{2}\sum_l \|\boldsymbol{W}^l\|_F^2 \tag{2}$$

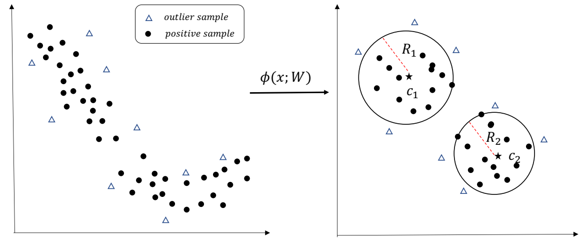


Figure 1: mSVDD with two modes. ϕ is a neural network. Fivestars denote a center, and black points denote positive target samples, while triangles denote negative outliers that need to be rejected by hyperspheres.

The first penalty term is for samples lying outside the sphere, i.e., when the distance of $\boldsymbol{x}_i$ to the center, $\|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}\|_2^2$, is greater than radius R after the transformation by network ϕ. The above loss also regularizes the radius and neural weight parameters in the second term. As with SVDD, parameter $\nu \in (0, 1]$ adjusts the tradeoff between the radius of the hypersphere and the points outside the hypersphere.

Schölkopf et al. (2001) proved that, in single-class classification, ν is the upper bound of the fraction of anomalies, and the lower bound of the fraction of training samples being anomalies or on the optimal boundary. Ruff et al. (2018) proved that this ν-property still holds for uni-modal soft-boundary *deep SVDD*.

Another simplified objective that minimizes the mean distance of all positive training samples to the center, the *one-class* form, can be defined as follows:

One-class deep SVDD (simplified form):

$$\frac{1}{n}\sum_i \|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}\|_2^2 + \frac{\lambda}{2}\sum_l \|\boldsymbol{W}^l\|_F^2 \quad (3)$$

Here, we can rewrite both the above in a unified form:

$$\mathcal{L}_{DSVDD} = C\sum_i [\|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}\|_2^2 - \beta]_+ + \beta + \frac{\lambda}{2}\sum_l \|\boldsymbol{W}^l\|_F^2, \quad (4)$$

where $[\cdot]_+ = max\{0, \cdot\}$, $\beta \in \{0, R^2\}$ and regularization parameter $C \in \{\frac{1}{n}, \frac{1}{\nu n}\}$ correspond to the two types of forms.

3 Multi-Modal Deep SVDD

In this section, we present our *mSVDD*, a method for deep one-class classification. Unlike a uni-modal model with a hypersphere, *mSVDD* uses a set of hyperspheres to describe target class data and to reject samples from negative classes. Figure 1 shows the general idea of *mSVDD* with two modes. Consider that we have m modes, each of which is described by a hypersphere $\boldsymbol{M}_j$ with center $\boldsymbol{c}_j$ and radius R_j; *mSVDD* uses each $\boldsymbol{M}_j$ to describe a distinctive aspect of the target class and then ensemble them. This ensembled deep *mSVDD* model could provide better descriptions for the target data.

As with *deep SVDD*, given target training samples $T = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$, *mSVDD* first transforms instance $\boldsymbol{x}$ into a data point of the output feature space with ϕ, where ϕ is a deep neural network of $L \in \mathbb{N}$ layers with parameters $\mathcal{W} = \{\boldsymbol{W}^1, ..., \boldsymbol{W}^L\}$. In contrast to *deep SVDD*, *mSVDD* uses m hyperspheres to include almost all of the target data with the minimum radii, i.e., $\frac{1}{m}\sum_{j=1}^m R_j^2$. As in kernel SVDD and *deep SVDD*, it should also punish points lying outside the sphere, i.e., if the distance of $\boldsymbol{x}$ to the center $\boldsymbol{c}$, $\|\phi(\boldsymbol{x}; \mathcal{W}) - \boldsymbol{c}\|_2^2$, is greater than radius R. Since we have a set of hyperspheres $M = \{\boldsymbol{M}_1, ..., \boldsymbol{M}_m\}$, one choice would be to punish $\boldsymbol{x}$ with respect to each hypersphere by adding $\sum_j \max\left(0, \|\phi(\boldsymbol{x}; \mathcal{W}) - \boldsymbol{c}_j\|_2^2 - R_j^2\right)$ to the loss function. However, the above penalty term is very hard, where one sample should satisfy each j-th constraint corresponding to $\boldsymbol{M}_j$. Therefore, we loosen the constraint. Given non-negative *attention weight* α_{ij} for $\boldsymbol{x}_i$ to each $\boldsymbol{M}_j$, the penalty term can be computed as the weighted average over m constraints. Now, only one ensembled constraint is required, i.e., *the sum of radii is greater than the sum distance to the center.* Formally, we can define our *mSVDD* objective as follows:

$$\frac{1}{\nu n}\sum_i \max(0, \sum_j \alpha_{ij}(\|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}_j\|_2^2 - R_j^2)) + \frac{1}{m}\sum_j R_j^2 + \frac{\lambda}{2}\sum_l \|\boldsymbol{W}^l\|_F^2, \quad (5)$$

where $\frac{1}{m}\sum_{j=1}^m R_j^2$ is the regularization term for radii from all m hyperspheres to get a closer boundary around the target data. This form can be seen as *mSVDD* with weighted soft-boundary constraints, which we call *soft-boundary mSVDD*.

Although the ν-property, mentioned in Section 2.2, does not hold true for our multi-modal case as it is in general, it is still true when the attention weight α_{ij} is constant for different hyperspheres. This will give us an intuition on the role of ν.

Proposition 1. [1] *The ν-property holds if we set equal attention weight to each hypersphere: i.) ν is an upper bound for the fraction of outlier samples and ii.) ν is a lower bound for the fraction of training samples being rejected or on the optimal boundary.*

3.1 One-class mSVDD (simplified from)

As in *deep SVDD*, we also have the simplified from and called: *one-class*[2] *mSVDD*. If we assume that the majority of the training data is not anomalous, then the radius can be ignored and we can define the simplified *mSVDD* as follows:

$$\frac{1}{n}\sum_{i}\sum_{j}\alpha_{ij}\|\phi\left(\boldsymbol{x}_i;\mathcal{W}\right)-\boldsymbol{c}_j\|_2^2+\frac{\lambda}{2}\sum_{l}\|\boldsymbol{W}^l\|_F^2, \tag{6}$$

where the attention weight α_{ij} will be kept, while the penalty of radius R is deleted.

3.2 Unified Form of mSVDD

We can write the two variants of *mSVDD* (i.e., *soft-boundary mSVDD* and simplified *one-class mSVDD*) in a unified form:

$$\begin{aligned}\mathcal{L}_{\text{mSVDD}} = C\sum_{i}[\sum_{j}\alpha_{ij}\left(\|\phi\left(\boldsymbol{x}_i;\mathcal{W}\right)-\boldsymbol{c}_j\|_2^2-\beta_j\right)]_+\\ +\frac{1}{m}\sum_{j}\beta_j+\frac{\lambda}{2}\sum_{l}\|\boldsymbol{W}^l\|_F^2,\end{aligned} \tag{7}$$

where $\beta_j \in \{0, R_j^2\}$. $\beta_j = 0$ corresponds to simplified *one-class mSVDD*, and $\beta_j = R_j^2$ corresponds to *soft-boundary mSVDD*. For $\boldsymbol{x}_i$ to the j-th hypersphere, attention weight α_{ij} should be inversely proportional to its distance to center $\boldsymbol{c}_j$. Thus, we define: $\alpha_{ij} = \exp(d(\boldsymbol{x}_i,\boldsymbol{c}_j)/\delta))/\sum_{k=1}^{m}\exp(d(\boldsymbol{x}_i,\boldsymbol{c}_k)/\delta)$, where $\delta < 0$ is a temperature hyperparameter.

3.3 Discussions on relationships between mSVDD and other models

Relationship with Uni-modal Deep SVDD Their relationship is obvious and can be summarized by the following proposition.

Proposition 2. *Deep SVDD is a special case of the unified form of mSVDD with one hypersphere used.*

Proof. Obviously, mSVDD becomes uni-modal (Ruff et al., 2018) if we use only one hypersphere, i.e., $m = 1$. ☐

Relationship between mSVDD and CVDD *CVDD* (Ruff et al., 2019) is a one-class model for text data. In *CVDD*, each training sample x_i (i.e., a text) is represented by r self-attention feature vectors $S_i = (\boldsymbol{s}_{i1}, ..., \boldsymbol{s}_{ir})$ (Lin et al., 2017). *CVDD* uses a group of r *context vectors* $C = (\boldsymbol{c}_1, ..., \boldsymbol{c}_r)$ to describe the target one-class data, where $\boldsymbol{c}_k \in \mathbb{R}^p$. *CVDD* tries to reduce the one-to-one *reconstruction* distance between feature vectors S_i and context vectors C. The loss can be defined as:

$$\mathcal{L}_{CVDD} = \frac{1}{n}\sum_{i}\sum_{k}\sigma_{ik}d(\boldsymbol{c}_k,\boldsymbol{s}_{ik}), \tag{8}$$

where $d(\boldsymbol{c}_k,\boldsymbol{s}_{ik})$ computes the distance, and σ_{ik} denotes the attention weight. The following proposition implies the close connection between two to learn one-class text problem.

Proposition 3. *CVDD is a very special case of one-class mSVDD when mSVDD is applied to text-based tasks under certain conditions.*

Proof. W.L.O.G., rewrite the loss function of *one-class mSVDD* in a simplified form for each sample as follows:

$$\begin{aligned}\mathcal{L}_{mSVDD} &= \frac{1}{n}\sum_{i}\sum_{j}\alpha_{ij}\|\phi_j\left(\boldsymbol{x}_i;\mathcal{W}\right)-\boldsymbol{c}_j\|^2\\ &= \frac{1}{n}\sum_{i}\sum_{j}\sigma_{ij}d(\boldsymbol{x}_i,\boldsymbol{c}_j)\approx\mathcal{L}_{CVDD},\end{aligned}$$

where we drop the regularization terms for weights of ϕ and radii, and set $m = r$, $\sigma_{ij} = \alpha_{ij}$, $d(\boldsymbol{x}_i,\boldsymbol{c}_j) = \|\phi_j\left(\boldsymbol{x}_i;\mathcal{W}\right)-\boldsymbol{c}_j\|^2$. $\phi\left(\boldsymbol{x}_i;\mathcal{W}\right)$ has to be a self-attention neural model, $\phi_j\left(\boldsymbol{x}_i;\mathcal{W}\right)$ is the $j-$th feature vector of sample $\boldsymbol{x}_i$, and $\boldsymbol{c}_j$ is the $j-$th context vector of target samples. Now the loss functions of *CVDD* and *one-class mSVDD* are almost the same. ☐

Relationship between mSVDD and DMSVDD *DMSVDD* (Ghafoori and Leckie, 2020) also uses multi-hyperspheres to extend SVDD. The loss function of *DMSVDD* is as follows:

$$\begin{aligned}\mathcal{L}_{\text{DMSVDD}} = \frac{1}{\nu n}\sum_{i}[\|\phi(\boldsymbol{x}_i;\mathcal{W})-\boldsymbol{c}_{i^*}\|_2^2-R_{i^*}^2]_+\\ +\frac{1}{K}\sum_{k}R_k^2+\frac{\lambda}{2}\sum_{l}\|\boldsymbol{W}^l\|_F^2,\end{aligned} \tag{9}$$

---

[1] The proofs of propositions can be found in the appendix.

[2] The term *"one-class"* is used following Ruff et al. (2018). Note that all the models discussed in this paper are one-class models.

where K is the number of hyperspheres[3], $\boldsymbol{c}_{i*}$ is the *nearest center* of sample $\boldsymbol{x}_i$ and R_{i*} is its radius.

Proposition 4. *DMSVDD can be seen as a hard-version of soft-boundary mSVDD if we set the attention weight in some way.*

Proof. In the calculation of the attention weight for *mSVDD* with $\alpha_{ij} = \frac{\exp(d(\boldsymbol{x}_i, \boldsymbol{c}_j/\delta))}{\sum_{k=1}^{m} \exp(d(\boldsymbol{x}_i, \boldsymbol{c}_k)/\delta)}$, the temperature parameter δ could influence the assignment of center c_k. If we set $\delta \to 0^-$, the above formula acts as the *argmin* operation.[4] In this case, $\alpha_{ii*} = 1$ if $i* = \arg\min_{k=1,...,K} d(\boldsymbol{x}_i, \boldsymbol{c}_k)$, and 0 otherwise. Now, we can get the form of *DMSVDD* from *soft-boundary mSVDD* (Eq. 5) through the adjustment of the attention weight. Therefore, we can prove that *DMSVDD* is also a special case of *mSVDD*. □

The above relation illustrates key difference between them: *DMSVDD* puts value on one hypersphere with the largest weight.

3.4 Summarizing mSVDD

We summarize the proposed *mSVDD* in accordance with the discussions presented above. The proposed *multi-modal deep SVDD (mSVDD)* learns a compact description of one-class data with multiple hyperspheres. *mSVDD* is also a generic framework that includes *deep SVDD, CVDD, and DMSVDD* if the corresponding conditions are met.

4 Multi-Modal Deep SVDD with Negative Supervision

In this section, we incorporate negative supervision into the training of *mSVDD*. The SVDD-related models are usually trained with only positive samples from the target one-class, while, if negative samples (samples which should be *rejected*) are available, the models can be extended to train with them to improve the description (Tax, 2002). Note that these samples are *not* necessarily required to be from "*real*" negative class. In our experiment, we use some external data as the *pseudo*-negative samples.

Given a set of extended training samples $T' = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_{n'}, y_{n'})\}$, where the first n samples are labeled $y_i = 1$, denoting positive, whereas the others are labeled $y_i = 0$, which denotes negative samples that should be rejected by *mSVDD*. Our *mSVDD* is represented with m hyperspheres and is formulated as $M = \{\boldsymbol{M}_1(\boldsymbol{c}_1, R_1), ..., \boldsymbol{M}_m(\boldsymbol{c}_m, R_m)\}$. It is required that the positive samples should be inside the m hyperspheres, while the negative samples should lie outside. Given training samples composed of positive and a negative samples, we can first get their corresponding distances to each center $\boldsymbol{c}_j$. The goal of optimization should be to pull the positive samples closer the center and to push the negative ones away. Formally, we define the distance between one sample $\boldsymbol{x}_i$ and one center $\boldsymbol{c}_j$ as $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{c}_j) = \|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}_j\|_2^2$. There are usually two types of losses to obtain the discriminative loss.

Contrastive type: The contrastive-type loss directly optimizes the distance by encouraging the distance between a positive sample and a center to be smaller, while it forces the larger distance to a negative sample:

$$\mathcal{L}_{Con\_d}^{(ij)} = y_i[d_{ij} - R_j^2]_+ + (1 - y_i)\,[R_j^2 - d_{ij}]_+, \tag{10}$$

where R_j^2 can be seen as a *margin* (or *threshold*) with a function that prevents too much effort from being wasted in enlarging/reducing distances (Hadsell et al., 2006).

Triplet type: The triplet-type loss is defined for a pair of positive sample $\boldsymbol{x}_i$ and negative sample $\boldsymbol{x}_{i'}$. If we consider center $\boldsymbol{c}_j$ as an *anchor* representative of target data, the triplet loss punishes only when d_{ij}, the distance from $\boldsymbol{x}_i$ to $\boldsymbol{c}_j$, is greater than $d_{i'j}$, the distance from $\boldsymbol{x}_{i'}$ to $\boldsymbol{c}_j$, with a margin $\tau > 0$:

$$\mathcal{L}_{Tri\_d}^{(ii'j)} = \left[d_{ij} - d_{i'j} + \tau\right]_+. \tag{11}$$

For clarity, Eqs. 10 and 11 show only the two types of losses for one hypersphere. Multi-modal version can be obtained by sum operation over $j \in \{1, ..., m\}$.

The triplet loss forces only positive samples to be closer to the center than negative samples, and the contrastive loss requires only keeping the distances for negative samples above the radius. These two types of objectives are easy to achieve, especially when we assume that negative samples are "*not real*." This can result in failing to make a full use of negative supervision. Therefore, we will reformulate both $\mathcal{L}_{Tri\_d}$ and $\mathcal{L}_{Con\_d}$.

---

[3] In DMSVDD, K changes dynamically. However, we ignore this difference and focus on the comparison of the models.

[4] δ approaches 0 from the negative side.

4.1 Reformulating Contrastive and Triplet Losses

Normalization layer In neural models with the contrastive or triplet loss, it is a common strategy to normalize the feature representations of samples for training stability (Schroff et al., 2015; Wang et al., 2017). Therefore, we apply the normalization to the input vectors: $\hat{\boldsymbol{x}} = \boldsymbol{x}/\sqrt{\sum |x_i|^2 + \epsilon}$, where $\epsilon > 0$ is a value avoiding division by zero.

Reformulation Given a center $\boldsymbol{c}_j$ and positive and negative samples, we can use the probability form in the optimization objective, rather than the two *non-probabilistic* ones: $\mathcal{L}_{Con\_d}$ and $\mathcal{L}_{Tri\_d}$. We introduce $p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j)$, which is the probability that a hypersphere with center $\boldsymbol{c}_j$ *accepts* the sample $\boldsymbol{x}_i$, and define it as follows:

$$p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j) = \sigma(s\hat{\boldsymbol{f}}_i^{\,T}\hat{\boldsymbol{c}}_j), \qquad (12)$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$, $\boldsymbol{f}_i = \phi(\boldsymbol{x}_i; \mathcal{W})$ denotes the feature output vector of $\boldsymbol{x}_i$, and s is a scale hyper-parameter for preventing failed convergence (Wu et al., 2018) after the normalization. For each sample $\boldsymbol{x}_i$, $\boldsymbol{c}_j$ acts as a *pseudo*-weight vector for the classification of the j-th hypersphere of *mSVDD*. Thus, given $p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j)$, the probability of a sample being accepted by hypersphere $\boldsymbol{M}_j$, we can reformulate the two discriminative losses with the probability.

***Contrastive type loss*:**

$$\begin{aligned}\mathcal{L}_{Con}^{(ij)} &= -y_i \log p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j) \\ &- (1 - y_i)\log p(y_i = 0|\boldsymbol{x}_i, \boldsymbol{c}_j) \qquad (13) \\ &= -y_i \log \sigma(s\hat{\boldsymbol{f}}_i^{\,T}\hat{\boldsymbol{c}}_j) - (1-y_i)\log \sigma(-s\hat{\boldsymbol{f}}_i^{\,T}\hat{\boldsymbol{c}}_j)\end{aligned}$$

This loss maximizes the likelihood of training positive samples being accepted or negative rejected.

***Triplet type loss*:**

$$\begin{aligned}\mathcal{L}_{Tri}^{(ii'j)} &= \left[\log \sigma(s\hat{\boldsymbol{f}}_{i'}^{\,T}\hat{\boldsymbol{c}}_j) - \log \sigma(s\hat{\boldsymbol{f}}_i^{\,T}\hat{\boldsymbol{c}}_j) + \tau\right]_+ \\ &= [\log p(y_{i'} = 1|\boldsymbol{x}_{i'}, \boldsymbol{c}_j) - \log p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j) + \tau]_+\end{aligned} \qquad (14)$$

The loss will punish when the log probability of a negative sample is greater than a positive sample with a margin τ.

Algorithm 1: mSVDD with Negative Supervision

```
Input: data_load load training batch from
       $T' = \{(\boldsymbol{x}_1, y_1), \ldots, \boldsymbol{x}_{n'}, y_{n'})\}$
Models: $model$ includes all modules for
        training, $f = \phi(\boldsymbol{x}_i; \mathcal{W})$ is a neural
        feature encoder
Parameters: Hyperparameters $\gamma$, $\nu$, $s$, $\tau$, $\epsilon$
for batch in batch_loader do
    /* load positive and negative
       samples                          */
    d_p, d_n = batch   // size: $n_p == n_n$
    p = $f$.forward(d_p)  // shape: $n_p \times d$
    n = $f$.forward(d_n )  // shape: $n_n \times d$
    Calculate $\mathcal{L}_{mSVDD}$ given (p) (Eq. 7)
    Calculate $\mathcal{L}_{Con|Tri}$ given (p,n) (Eqs. 13
     or 14).
    /* get final loss (Eq. 16)      */
    $\mathcal{L}oss = \mathcal{L}_{mSVDD} + \gamma * \mathcal{L}_{Con|Tri}$
    $\mathcal{L}oss$.backwards()
    $model$.update()
```

4.2 Reformulating Contrastive and Triplet Losses for Multiple Modes

While Eqs. (12), (13), and (14) show the uni-modal case, for the multi-modal one, we have to consider m different centers $\{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m\}$ in the calculation of the two reformulated discriminative losses. Therefore, we propose two strategies as follows:

$$p(y_i = 1|\boldsymbol{x}_i) = \begin{cases} \max\limits_{1 \leqslant j \leqslant m} p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j) & \textit{Max} \\ \frac{1}{m}\sum_j p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{c}_j) & \textit{Mean}, \end{cases} \qquad (15)$$

where *Max* references only $\boldsymbol{M}_j$ with the max logit output, while *Mean* takes account of all hyperspheres equally. Then, we can obtain the corresponding *Contrastive* and *Triplet* losses by substituting Eqs. (13) and (14) with the probability term (Eq. (15)).

4.3 Training Loss

The final training loss for the *mSVDD with negative supervision* can be formulated as:

$$\mathcal{L} = \mathcal{L}_{mSVDD} + \gamma\mathcal{L}_{Con|Tri}, \qquad (16)$$

where γ adjusts between the *mSVDD* loss and the discrimination with negative supervision. In the training process, $\mathcal{L}_{Con|Tri}$ will sum the loss from one batch samples with Eqs. (13) and (14). Algorithm 1 provides the training process for *mSVDD* with negative supervision in one epoch. Please see Section B in the appendix for more discussions on the relationship between *mSVDD* and the use of negative supervision.

5 Experiments

5.1 Datasets and Implementation Details

Datasets Experiments were conducted on two datasets: *20 Newsgroups*[5] and *Reuters*[6], which have been commonly used in other one-class text classification work (Manevitz and Yousef, 2001; Ruff et al., 2019). We used the same pre-processing steps as the ones used in earlier work (Ruff et al., 2019), including lowercasing, removing stopwords, and tokenization. We used the external data for negative supervision in the absence of "*real*" labeled negative instances. We followed the similar logic for choosing our external data as the one in the field of pretrained word vectors, in which one general corpus, such as Wikipedia articles, is often adopted as the training dataset (Mikolov et al., 2013). So we also chose one publicly available corpus WikiText-2 (Merity et al., 2016), extracted from Wikipedia articles, as our external data. As shown in Algorithm 1, data loader loads one batch of negative samples, i.e., sentences from WikiText-2, which are labeled with 0.

Encoder For encoding the text input, i.e., $\phi(\boldsymbol{x}, W)$, we used a Bidirectional LSTM with attention (Hochreiter and Schmidhuber, 1997; Xu et al., 2015), with the number of hidden units being 150. For the pre-trained word embeddings, we experimented with GloVe Vectors (Pennington et al., 2014) and set the dimension to 300. In our experiments, we did not adopt the widely used BERT model (Devlin et al., 2019), as Ruff et al. (2019) showed that BERT model did not improve the performance.

Settings As for the optimization of parameters, Adam (Kingma and Ba, 2014) with a base learning rate of 0.001 was used for 50 epochs. The batch sizes were set to 32 and 64 for *Reuters* and *Newsgroups*, respectively. For the initialization of *mSVDD* model, we employed two operation steps. In the absence of negative samples, *mSVDD* was first pre-trained on target samples by using an AutoEncoder with two objectives: 1) warm-up and 2) reducing the *reconstruction* error for the target samples, such that the model could be more robust to noise or anomalous inputs (Jacobs, 1995; Hinton and Salakhutdinov, 2006). An AutoEncoder feed-forward network with a 0.5 compression rate, which consists of an encoder and a decoder, was put on the back of the BiLSTM feature network. Then, the weights of the m hyperspheres in *mSVDD* were initialized by running k-means clustering on the features learned before (Lloyd, 1982). As for the regularization term of *mSVDD*, $\boldsymbol{c}_j$ was regularized (Ng, 2004), and a weight decay with 0.95 was applied for the parameters. As for the number of hyperspheres, different settings, $1, 3, 5, 10$, were tested. For the hyperparameters, we set parameter $s = 1.2$ for scale, $\nu = 0.1$, $\delta = -0.9$ for the attention weight, $\tau = 0.1$ for the triplet loss, $\epsilon = 1\mathrm{e}{-6}$ for norm, and $\gamma = 1$ for the training loss. The results were averaged over 10 runs with different random seeds.

Evaluation metrics The performance was measured by the *area under the receiver operating characteristics* (ROC) curve (AUCs), a commonly used metric for one-class text classification (Manevitz and Yousef, 2001; Ruff et al., 2019).

5.2 Results

5.2.1 Results of mSVDD

Table 1 shows the performance of *mSVDD* with different choices of m, i.e., the number of hyperspheres. Here, mSVDD(1) represents unimodal *deep SVDD* ((Ruff et al., 2018)). The results show that: 1) As for the *one-class* version, *mSVDD* could provide better performances than the uni-modal one, especially when more hyperspheres were used. We can see that mSVDD(10), which uses the largest number of m, outperforms mSVDD(1) in more times than mSVDD(5) and mSVDD(3), that performs comparable with unimodal *mSVDD*. Similar results can also be observed in *soft-boundary*, *mSVDD* with more hyperspheres (10 or 5) won more times, nine out of thirteen cases in two datasets, than other settings. This proves the necessity of incorporating more hyperspheres to better describe the target data.

2) While the performance of *mSVDD* did not improve linearly along with m, we can explain this from the following aspects. As for the model, *mSVDD* with more centers means that it has more

---

[5] http://qwone.com/json/20Newsgroups

[6] http://daviddlewis.com/resources/testcollections/reuters21578/

Target Class	mSVDD (1)		mSVDD (3)		mSVDD (5)		mSVDD (10)	
	One-v	*Soft-v*	*One-v*	*Soft-v*	*One-v*	*Soft-v*	*One-v*	*Soft-v*
Reuters								
earn	95.6	95.9	95.5	95.9	96.0	**96.2**	95.9	96.1
acq	89.4	89.0	90.0	89.1	89.3	89.1	**90.1**	89.2
crude	92.7	**92.8**	92.5	91.5	92.5	92.5	92.4	92.4
trade	98.4	98.3	98.3	**98.9**	98.8	98.7	98.6	98.8
money	86.3	86.2	85.1	86.2	86.4	86.4	**87.1**	86.8
interest	97.2	**97.3**	97.2	96.9	97.2	96.6	**97.3**	96.8
ship	92.5	91.7	**93.8**	91.6	**93.8**	92.3	92.6	91.7
20 News								
comp	85.3	84.9	86.2	86.0	86.1	85.9	**86.7**	86.5
rec	77.1	76.2	**77.7**	77.0	77.6	76.9	77.6	76.8
sci	66.5	66.3	**67.3**	**67.3**	67.1	66.7	66.9	67.0
misc	75.2	75.0	76.0	**76.2**	75.5	**76.2**	75.5	**76.2**
pol	**79.2**	79.1	78.5	78.7	78.7	78.4	78.5	78.4
rel	**83.6**	82.5	83.1	82.5	83.1	82.3	83.1	82.2

Table 1: mSVDD with different settings of m. Numbers in brackets denote the number of hyperspheres in mSVDD. *One-v* and *Soft-v* denote the two versions of mSVDD, *One-class* and *Soft-boundary*, respectively. AUCs in % on the *Reuters* (upper part) and *20 Newsgroup* (lower part) datasets. Best scores in each *row* are presented in **bold**, while the second best are underlined.

parameters and a complex model structure, which is hard to be optimized, especially on the data with a small training size (e.g., *pol* or *rel*.) As for the data, some data might have simple data distributions without the need for more modes. Another aspect would be the attention weights of multiple hyperspheres. (Ghafoori and Leckie, 2020) showed that focusing on some "*good*" hyperspheres would be beneficial rather than over all hyperspheres. In the calculation of attentions, we did not adjust δ so as to have a large weight for one specific hypersphere. This may cause limited improvements. We will compare mSVDD with DMSVDD later.

5.2.2 Results of mSVDD with negative supervision

Table 2 shows the performance of *mSVDD* trained with negative supervision and compares the results with the other methods. From the discussion in the last subsection, we used $m = 3$ in this subsection. To perform negative supervision for *mSVDD*, we evaluated four approaches where different losses and their reformulated probability forms were selected. For the method of *DMSVDD*, we report the results in the setting of the initial number of spheres $K_{init} = 10$. As for the comparison between *DMSVDD* and *mSVDD*, *DMSVDD* puts value on one hypersphere and performs slightly better over mSVDD(3) in some cases (e.g., *earn, acq* and *comp*). This indicates one inspiration that discarding "*bad*" hyperspheres is sometimes necessary.

For *Reuters*, the results indicate that *mSVDD* could benefit from the joint training of the discrimination losses, except for *acq* and *ship*. *mSVDD with negative supervision* also achieved the best scores in the four cases compared with other methods including *DMSVDD*.

We have more obvious comparisons for *20 Newsgroup*. All four negative supervision methods could improve *mSVDD* markedly and perform the best over all baselines for all target classes of *20 Newsgroup*. For example, mSVDD with negative supervision could increase 2-3 points for *comp*. For different losses for negative supervision, the *contrastive* type loss, which has larger punishment over negative data, performs better than the *triplet* type loss, which uses a relatively small margin. Much more distinct improvements can be seen in the comparison with CVDD for *rec* or with OC-SVM for *misc*, while we obtained their best scores from Ruff et al. (2019). Further, the *contrastive* loss consistently outperformed other models including the baselines. In addition, the performance of *Con+Max* was greater than the *Con+Mean* strategy to reformulate the probability. We hypothesize that focusing on one of the hyperspheres is effective when we used *mSVDD* with the *contrastive* loss.

5.2.3 Results of CVDD with negative supervision

Table 3 shows the results of *CVDD* with the proposed negative supervision for mSVDD. As mentioned in Section 3.3, *CVDD* can be seen as a special case of *mSVDD*. Therefore, the proposed negative supervision approaches to *mSVDD* can be also applied to *CVDD* theoretically. To highlight the usefulness of the negative supervision, we conducted the experiments to use the *triplet* loss with *Max* probability for CVDD. As for the implementation, since CVDD uses a different multi-head structure, we also used a different form to incorporate *Triplet+Max* to CVDD (See Section C in the appendix for the details of the implementation.).

Overall, we can see that the proposed negative supervision could enhance *CVDD* in most cases on the two datasets. The overall performance mainly shows the following: 1) The improvement by the negative supervision to CVDD is consistent with mSVDD due to the similarity between the two. 2) The generality of the negative supervision can be shown, as *Triple+Max* was successfully applied to the different multi-head structure.

Regarding different target-classes, *ship* with the smaller training data size may cause worse perfor-

	Reuters target class							*20 Newsgroup* target class					
Model	*earn*	*acq*	*crude*	*trade*	*money*	*interest*	*ship*	*comp*	*rec*	*sci*	*misc*	*pol*	*rel*
OC-SVM	91.1	**93.1**	92.4	99.0	88.6	97.4	93.1	82.0	75.6	64.1	63.1	75.5	79.2
CVDD	94.0	91.5	**95.5**	99.2	82.8	97.7	**97.6**	70.9	53.3	56.8	75.1	65.3	76.3
DSVDD	95.9	89.4	92.8	98.4	86.3	97.3	92.5	85.3	77.1	66.5	75.2	79.2	83.6
DMSVDD	96.0	89.8	92.1	98.8	87.1	97.2	93.0	86.3	77.1	66.8	75.3	78.5	82.0
mSVDD_*One*	95.5+	90.0	92.5+	98.3+	85.1+	97.2+	93.8	86.2+	77.7+	67.3+	76.0+	78.5+	83.1+
+Triple+Max	96.9	89.4	93.8	**99.6**	89.0	**98.4**	92.7	88.3	78.6	67.5	77.6	79.9	83.8
+Triple+Mean	97.1	89.9	93.9	99.5	89.3	98.3	92.3	87.9	77.5	67.7	75.5	79.1	83.9
+Con+Max	**97.2**	90.8	92.9	98.8	**91.3**	97.8	92.3	**89.4**	**79.1**	68.3	76.4	80.7	84.2
+Con+Mean	96.6	91.0	92.8	98.6	90.2	98.0	91.9	89.2	78.9	68.3	76.6	79.9	**84.4**
mSVDD_*Soft*	95.9+	89.1	91.5+	98.9+	86.2+	96.9+	91.6	86.0+	77.0+	67.3+	76.2+	78.7+	82.5+
+Triple+Max	97.1	89.6	92.8	99.4	89.3	97.4	92.6	87.8	78.5	68.8	76.1	79.9	82.9
+Triple+Mean	**97.2**	90.1	92.4	99.3	91.0	98.0	92.7	87.5	78.7	68.5	76.6	79.7	83.0
+Con+Max	97.0	88.2	93.1	99.2	91.2	**98.4**	91.6	88.6	78.3	**69.0**	**78.3**	80.5	83.2
+Con+Mean	**97.2**	88.1	93.0	98.9	**91.3**	**98.4**	90.4	88.3	78.7	68.4	78.1	**80.8**	83.4

Table 2: mSVDD with negative supervision. AUCs in % on the *Reuters* (left part) and *20 Newsgroup* (right part) datasets. For OC-SVM and CVDD, two baselines, we adopted their *best* scores from (Ruff et al., 2019). DSVDD and DMSVDD were our implementations. _*One* and _*Soft* mean *One-class* and *Soft-boundary* forms, respectively. +Triple+Max, which denotes mSVDD with *Triplet* loss with *Max* probability strategy, followed by three other negative supervision methods. In the rows of mSVDD_*Soft* and mSVDD_*One*, '+' following numbers means that there were improvements with negative supervision (three of four methods.) The best scores in each *column* are presented in **bold**, while the second best are underlined.

	Reuters target class						
Model (r)	*earn*	*acq*	*crude*	*trade*	*money*	*interest*	*ship*
CVDD (3)	94.0	90.2	89.6	98.3	82.5	92.3	97.6
+Triple+Max	**96.1**	90.2	**97.3**	98.3	**84.2**	**92.4**	91.8
CVDD (5)	92.8	88.7	92.5	98.2	76.7	91.7	96.9
+Triple+Max	**94.0**	**94.4**	**96.7**	**98.7**	**84.0**	**97.3**	92.5
CVDD (10)	91.8	91.5	95.5	99.2	82.8	97.7	95.6
+Triple+Max	**93.0**	91.2	**97.4**	**99.6**	**85.7**	**98.7**	94.2

	20 Newsgroup target class					
Model (r)	*comp*	*rec*	*sci*	*misc*	*pol*	*rel*
CVDD (3)	70.9	50.8	56.7	75.1	62.9	76.3
+Triple+Max	**74.5**	**64.2**	**61.0**	75.1	62.2	72.5
CVDD (5)	66.4	52.8	56.8	70.2	65.3	72.9
+Triple+Max	**73.2**	**64.5**	**58.4**	**76.2**	63.6	**76.1**
CVDD (10)	63.3	53.3	55.7	68.6	65.1	70.7
+Triple+Max	**78.3**	**69.7**	**60.5**	**73.3**	**67.5**	**79.1**

Table 3: CVDD with the proposed negative supervision. AUCs in % on the *Reuters* (upper part) and *20 Newsgroup* (lower part) datasets. Number r in brackets denotes the number of heads in CVDD. **Bold** means the better AUCs score.

mance, so does *real* with CVDD(3), which are similar phenomena with mSVDD. In addition, the negative supervision could also prevent over-fitting for CVDD. For example, CVDD(3) with the minimal parameters achieved the best score for *comp* when varying "r" among 3, 5 and 10. In contrast, when the negative supervision was used, CVDD(10) with the maximal parameters attained the best and also performed better for all six target classes of the *20 Newsgroup* dataset.

6 Conclusion

In this work, we proposed *mSVDD*, a new generic one-class text classification framework that uses multi-modal *deep SVDD*. Rather than the uni-modal *deep SVDD*, *mSVDD* can enhance the description ability to the target one-class data with multiple hyperspheres. We also proved that this generic framework can include three variants, *deep SVDD, DMSVDD*, and *CVDD* under certain conditions. In addition, in the absence of "*real*" negative training data, we also proposed approaches for effectively adding negative supervision to further improve the performance of *mSVDD*. The experiments validated that the proposed *mSVDD* provides better performance compared to uni-modal SVDD. The experiments also showed the further improvements in most cases when negative supervision was used for *mSVDD* and *CVDD*. For future work of this study, we will use some sampling strategies to improve the current work.

Acknowledgments

The authors would like to gratefully acknowledge the anonymous reviewers for their helpful comments and suggestions. Chenlong Hu acknowledges the support from China Scholarship Council(CSC).

References

Hany Alashwal, Safaai Deris, and Razib M Othman. 2006. One-class support vector machines for protein-protein interactions prediction. *International Journal of Biological and Medical Sciences*, 1(2).

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Christopher M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2010. Anomaly detection for discrete sequences: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):823–839.

Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Schölkopf. 2005. A tutorial on ν-support vector machines. *Applied Stochastic Models in Business and Industry*, 21(2):111–136.

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

ST Dumais, J Platt, D Heckerman, and M Sahami. 1998. Inductive learning algorithms and representations for text categorization. 1998. In *Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management (Bethesda, MD, 1998)*, pages 148–155.

Zahra Ghafoori and Christopher Leckie. 2020. Deep multi-sphere support vector data description. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 109–117. SIAM.

Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. 2013. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9):2250–2267.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Pei-Yi Hao and Yen-Hsiu Lin. 2007. A new multiclass support vector machine with multi-sphere in the feature space. In *New Trends in Applied Artificial Intelligence*, pages 756–765, Berlin, Heidelberg. Springer Berlin Heidelberg.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Robert A Jacobs. 1995. Methods for combining experts' probability assessments. *Neural computation*, 7(5):867–888.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Raksha Kumaraswamy, Anurag Wazalwar, Tushar Khot, Jude Shavlik, and Sriraam Natarajan. 2015. Anomaly detection in text: The value of domain knowledge. In *The Twenty-Eighth International Flairs Conference*.

Trung Le, Dat Tran, and Wanli Ma. 2013. Fuzzy multisphere support vector data description. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 570–581. Springer.

Trung Le, Dat Tran, Wanli Ma, and Dharmendra Sharma. 2010. A theoretical framework for multisphere support vector data description. In *International Conference on Neural Information Processing*, pages 132–142. Springer.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Larry Manevitz and Malik Yousef. 2007. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481.

Larry M Manevitz and Malik Yousef. 2001. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mary M Moya, Mark W Koch, and Larry D Hostetler. 1993. One-class classifier networks for target recognition applications. *NASA STI/Recon Technical Report N*, 93.

Andrew Y Ng. 2004. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78.

Mahesh Pal and Giles M Foody. 2010. Feature selection for classification of hyperspectral data by svm. *IEEE Transactions on Geoscience and Remote Sensing*, 48(5):2297–2307.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Stephen J Roberts. 1999. Novelty detection using extreme value statistics. *IEE Proceedings-Vision, Image and Signal Processing*, 146(3):124–129.

Erik Rodner, Esther-Sabrina Wacker, Michael Kemmler, and Joachim Denzler. 2011. One-class classification for anomaly detection in wire ropes with gaussian processes in a few lines of code. *training*, 1:1–5.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *International Conference on Machine Learning*, pages 4393–4402.

Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. 2019. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. 2001. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471.

Bernhard Schölkopf, Alex J Smola, Robert C Williamson, and Peter L Bartlett. 2000. New support vector algorithms. *Neural computation*, 12(5):1207–1245.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

David Martinus Johannes Tax. 2002. One-class classification: Concept learning in the absence of counter-examples.

David MJ Tax and Robert PW Duin. 2004. Support vector data description. *Machine learning*, 54(1):45–66.

Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. 2017. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742.

Yanshan Xiao, Bo Liu, Longbing Cao, Xindong Wu, Chengqi Zhang, Zhifeng Hao, Fengzhao Yang, and Jie Cao. 2009. Multi-sphere support vector data description for outliers detection on multi-distribution data. In *2009 IEEE international conference on data mining workshops*, pages 82–87. IEEE.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Hwanjo Yu, Jiawei Han, and Kevin Chen-Chuan Chang. 2004. Pebl: Web page classification without negative examples. *IEEE Transactions on Knowledge & Data Engineering*, (1):70–81.

A Proofs of Proposition 1

Schölkopf et al. (2001) proved that, in single-class classification, ν is the upper bound of the fraction of anomalies, and the lower bound of the fraction of training samples being anomalies or on the optimal boundary. Ruff et al. (2018) proved that this ν-property still holds for uni-modal soft-boundary *deep SVDD*. Although the same proposition does not hold true for our multi-modal case as it is in general, it is still true when the attention weight α_{ij} is constant for different hyperspheres. This will give us an intuition on the role of ν.

Proposition 1. *(ν-property)*[7] *The hyperparamter $\nu \in (0, 1]$ in soft-boundary deep mSVDD holds if we set an equal attention weight to each hypersphere:*

- *i. ν is an upper bound on the fraction of outlier samples.*
- *ii. ν is a lower bound on the fraction of training samples being rejected or on the optimal boundary.*

Proof. **Ad (i).** For each training instance $\boldsymbol{x}_i$, its loss function is defined as *hinge-loss*: $l(f(\boldsymbol{x}_i)) = \max\{0, \sum_j \alpha_{ij}\,(\|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}_j\|^2 - R_j^2)\}$, where f is the model with parameters. Let us define $d_i = \sum_j \alpha_{ij} d(\boldsymbol{x}_i, \boldsymbol{c}_j)$. Assume $\alpha_{ij} = 1/m$, we have $d_i = \frac{1}{m}\sum_j d(\boldsymbol{x}_i, \boldsymbol{c}_j)$. We also define $R_s = \frac{1}{m}\sum_j R_j^2$. And W.L.O.G, we also assume $d_1 \leqslant \ldots \leqslant d_n$ which means d_n is n-th farthest sum distance. The number of outliers is given by $n_{out} = |\{i | d_i > R_s\}|$. Rewrite the objective of *soft-boundary* deep mSVDD (Eq. 5) as:

$$J_{softm} = R_s - \frac{n_{out}}{\nu n} R_s = (1 - \frac{n_{out}}{\nu n}) R_s$$

Since the objective of *mSVDD* is to get a minimum R_s, therefore $1 - \frac{n_{out}}{\nu n}$ should be positive, Thus, $n_{out} \leqslant \nu n$ must hold in the training. It implies that at most νn outliers should be rejected.

Ad (ii). The optimal R_s^* has to hold the inequality $n_{out} \leqslant \nu n$. If $R_s^* >= d_n$, then n_{out} takes the minimum value of 0 which means the boundary includes all the samples. Since n_{out} is increased as long as R_s decreased. If n_{out} take the maximum value of νn under condition **(i)**, we can have the minimal $R_s^* = d_{i*}$, where $i* = n - n_{out}$ means d_{i*} is $(n - n_{out})$-th farthest distance. We define $\{\boldsymbol{x}_i | d_i \geqslant R_s^*\}$ is the set of training samples being rejected ($d_i > R_s^*$) or on the optimal boundary ($d_i = R_s^*$). Then we have inequality: $|\{\boldsymbol{x}_i | d_i \geqslant R_s^*\}| = |\{\boldsymbol{x}_i | d_i > R_s^*\} \cup \{\boldsymbol{x}_i | d_i = R_s^*\}| \geqslant n_{out} + 1 \geqslant \nu n$. This implies that at least νn samples being rejected or just on the optimal boundary. Figure 2 shows an example with 10 training samples. ☐

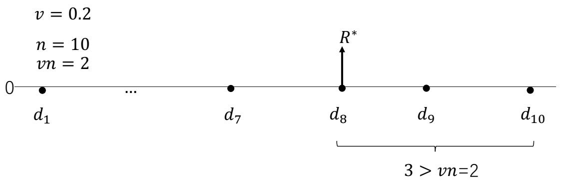


Figure 2: Example of **ii** of ν-property. Up arrow means $R^* = d_8$, where $n = 10$ means 10 samples, $\nu = 0.2$.

Proposition 1 and its proof refer to works (Ruff et al., 2018; Chen et al., 2005; Schölkopf et al., 2000).

B Discussions on mSVDD with Negative Supervision

B.1 Relationship between mSVDD and the use of negative supervision

mSVDD and negative supervision are not two independent sub-architectures. Negative supervision, including *contrastive* and *triplet* losses, are specially equipped to mSVDD. Specifically, these two components are closely connected by the center of the hypersphere, $\boldsymbol{c}_j$. Both mSVDD (Eq. 7) and negative supervision (Eq. 13 or 14) contain $\boldsymbol{c}_j$. Since there is no *real* negative data, external data are used as *pseudo* negative samples to complete negative supervision. The use of negative supervision could improve the discrimination ability of mSVDD. In training, negative supervision loss forces mSVDD to reject *unseen* samples since real negative data in testing are also unseen in training. This improves inter-class discrepancy, compared with intra-class loss mSVDD optimized. However, in testing, the decision function will be the same as mSVDD trained with only positive samples.

B.2 Necessity of joint loss

In training loss of mSVDD with *negative supervision*(Eq. 16), $\mathcal{L}_{mSVDD}$ aims to minimize the intra-class variations while $\mathcal{L}_{Con|Tri}$ tries to maximize the inter-class discrimination. If γ in Eq. 16 is set to 0, it will train mSVDD only with target

---

[7] Rewrite this proposition in the main body

positive samples, where discriminative information could not be learned. On the other hand, if we use only $\mathcal{L}_{Con|Tri}$ loss for training, it may result in large intra-target variations, especially when the *triplet* type loss is chosen, since it requires that only positive samples to be closer than *pseudo* negative samples. Additionally, because of the absence of *real* negative samples, it is another problem to sample the "appropriate" *pseudo* negative samples, such that the *contrastive* or *triplet* losses could fit our original objective, that is, learn a compact description boundary for the target one-class data. Therefore, it is necessary to jointly train with the loss of negative supervision.

C Implementation of CVDD with negative supervision

The proposed negative supervision methods can also be applied to CVDD. Now, we introduce our implementation of CVDD with *triplet* type loss and the *Max* probability strategy. CVDD uses a group of r *context vectors* $C = (\boldsymbol{c}_1, ..., \boldsymbol{c}_r)$ to describe the target one-class data, where $\boldsymbol{c}_k \in \mathbb{R}^p$. Given one context vector $\boldsymbol{c}_k, \forall k \in \{1, ..., r\}$ and a pair of training positive and negative samples, we can get the reformulated probability form. First, CVDD maps a training sample x_i to r heads of feature vectors $S_i = (\boldsymbol{s}_{i1}, ..., \boldsymbol{s}_{ir})$. Then, we denote $p(y_i = 1|\boldsymbol{s}_{ik}, \boldsymbol{c}_k)$ as the probability that k-th $\boldsymbol{s}_{ik}$ reconstructs k-th context vector $\boldsymbol{c}_k$ well.

$$p(y_i = 1|\boldsymbol{s}_{ik}, \boldsymbol{c}_k) = \sigma(\hat{\boldsymbol{s}_{ik}}^T \hat{\boldsymbol{c}_k}) \tag{17}$$

And with *triplet* and *Max* probability strategy, we can define the negative supervision loss as:

$$\begin{aligned}
\mathcal{L}_{Tri}^{(ii')} &= [\log p(y_{i'} = 1|\boldsymbol{x}_{i'}) - \log p(y_i = 1|\boldsymbol{x}_i) + \tau]_+ \\
&= [\log \max_{k=1,\dots,r} p(y_{i'} = 1|\boldsymbol{s}_{i'k}, \boldsymbol{c}_k) - \\
&\quad - \log \max_{k=1,\dots,r} p(y_i = 1|\boldsymbol{s}_{ik}, \boldsymbol{c}_k) + \tau]_+
\end{aligned} \tag{18}$$

where τ is a margin. Then, $\mathcal{L}_{Tri}^{(ii')}$ can then be added to Eq. 8 to obtain the training loss.

Unsupervised Word Polysemy Quantification with Multiresolution Grids of Contextual Embeddings

Christos Xypolopoulos*
École Polytechnique

Antoine J.-P. Tixier*
École Polytechnique

Michalis Vazirgiannis
École Polytechnique & AUEB

Abstract

The number of senses of a given word, or polysemy, is a very subjective notion, which varies widely across annotators and resources. We propose a novel method to estimate polysemy based on simple geometry in the contextual embedding space. Our approach is fully unsupervised and purely data-driven. Through rigorous experiments, we show that our rankings are well correlated, with strong statistical significance, with 6 different rankings derived from famous human-constructed resources such as WordNet, OntoNotes, Oxford, Wikipedia, etc., for 6 different standard metrics. We also visualize and analyze the correlation between the human rankings and make interesting observations. A valuable by-product of our method is the ability to sample, at no extra cost, sentences containing different senses of a given word. Finally, the fully unsupervised nature of our approach makes it applicable to any language. Code and data are publicly available[1].

1 Introduction

Polysemy, the number of senses that a word has, is a very subjective notion, subject to individual biases. Word sense annotation has always been one of the tasks with the lowest values of inter-annotator agreement (Artstein and Poesio, 2008). Yet, creating high-quality, consistent word sense inventories is a critical pre-requisite to successful word sense disambiguation.

Towards creating word sense inventories, it can be helpful to have some reliable information about polysemy. That is, knowing which words have many senses and which words have only a few senses. Such information can help in creating new inventories but also in validating and interpreting existing ones. It can also help select which words to include in a study (e.g., only highly polysemous words).

We propose a novel, fully unsupervised, and data-driven approach to quantify polysemy, based on basic geometry in the contextual embedding space.

Contextual word embeddings have emerged in the last few years as part of the NLP transfer learning revolution. Now, entire deep models are pre-trained on huge amounts of unannotated data and fine-tuned on much smaller annotated datasets. Some of the most famous examples include ULMFiT (Howard and Ruder, 2018) and ELMo (Peters et al., 2018), both based on recurrent neural networks; and GPT (Radford et al., 2018) and BERT (Devlin et al., 2018), based on transformers (Vaswani et al., 2017). These models all are very deep *language models*. During pre-training on large-scale corpora, they learn to generate powerful internal representations, including fine-grained contextual word embeddings. For instance, in a well pre-trained model, the word *python* will have two very different embeddings depending on whether it occurs in a programming context (as in, e.g., "I love to write code in python") or in an ecological context ("while hiking in the rainforest, I saw a python").

Our approach capitalizes on the contextual embeddings previously described. It does not involve any tool and does not rely on any human input or judgment. Also, thanks to its unsupervised nature, it can be applied to any language (even those with limited resources), provided that contextual embeddings are available.

The remainder of this paper is organized as follows. We detail our approach in section 2. Then,

---

[*] Equal contribution. CX handled the data, generated the rankings, and sampled the examples of section 7. AJPT computed the results, plots, and wrote the paper. Both authors participated in the design of the study.

[1] https://github.com/ksipos/polysemy-assessment

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3391–3401
April 19 - 23, 2021.

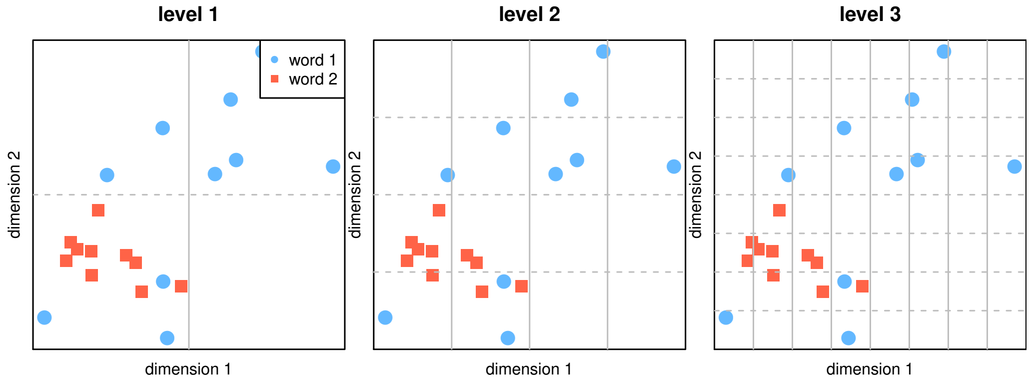


Figure 1: Illustration of the proposed approach with $D = 2$ and $L = 3$.

we present our experimental setup (sec. 3), evaluation metrics (sec. 4), and report and interpret our results (sec. 6). In section 7, we briefly touch on two other interesting applications of our method. One that allows the user to sample sentences containing different senses of a given word and one that goes towards word sense induction. Finally, related work is presented in section 8.

2 Proposed approach

2.1 Basic assumption

First, by passing diverse sentences containing a given word to a pre-trained language model, we construct a representative set of vectors for that word (one vector for each occurrence of the word). The basic and intuitive assumption we make is that *the volume covered by the cloud of points in the contextual embedding space is representative of the polysemy of the associated word.*

2.2 Main idea: multiresolution grids

As a proxy for the volume covered, we adopt a simple geometrical approach. As shown in Fig. 1, we construct a hierarchical discretization of the space, where, at each level, the same number of bins are drawn along each dimension. Each level corresponds to a different resolution. Our polysemy score is based on the proportion of bins covered by the vectors of a given word at each level.

Grid vs. clustering. Using a binning strategy makes more sense than a clustering-based approach. Indeed, clusters do not partition the space equally and regularly. This is especially problematic since word representations are not uniformly distributed in the embedding space (Ethayarajh, 2019). Therefore, the vectors lying in the same dense area of the space will always belong to one single large cluster, while outliers lying in the same, but sparser, area of the space, will be assigned to many different small clusters. Therefore, counting the number of clusters a given word belongs to is not a reliable indicator of how much of the space this word covers.

2.3 Scoring scheme

We quantify the polysemy degree of a word w as:

$$\text{score}(w) = \sum_{l=1}^{L} \frac{\text{coverage}_w^l}{2^{L-l}} \tag{1}$$

where coverage_w^l designates the proportion of bins covered by word w at level l, between 0 and 1. At each level, 2^l bins are drawn along each dimension (see the vertical and horizontal lines in Fig. 1). The hierarchy starts at $l = 1$ since there is only one bin covering all the space at $l = 0$ (so all words have equal coverage at this level). The total number of bins in the entire space, at a given level l, is equal to $(2^l)^D$.

Consider again the example of Fig. 1. In this example, each word is associated with a set of 10 contextualized embeddings in a space of dimension $D = 2$, and the hierarchy has $L = 3$ levels. First, we can clearly see that word 1 (blue circles) covers a large area of the space while all the vectors of word 2 (orange squares) are grouped in the same region. Intuitively, this can be interpreted as "word 1 occurs in more different contexts than word 2", which per our assumption, is equivalent to saying that "word 1 is more polysemous than word 2".

Let us now see how this is reflected by our scoring scheme. First, the penalization terms (denominators) for levels 1 to 3 are $\left[\frac{1}{2^2}, \frac{1}{2^1}, \frac{1}{2^0}\right] = \left[\frac{1}{4}, \frac{1}{2}, 1\right]$. Note that the higher the level, the exponentially more bins, and so the less penalized (or the more rewarded) coverage is, because getting good coverage becomes more and more difficult. Now, per

Eq. 1, the score of word 1 is computed as the dot product of its coverage vector $\left[\frac{3}{4}, \frac{7}{16}, \frac{10}{64}\right]$ (coverage at each level) with the penalization vector, which gives a score of 0.5625. Likewise, the score of word 2 is computed as $\left[\frac{1}{4}, \frac{1}{2}, 1\right] \cdot \left[\frac{1}{4}, \frac{4}{16}, \frac{7}{64}\right] = 0.297$. We can thus see that our scores reflect what can be observed in Fig. 1: word 1 covers a larger area of the space than word 2.

Note that the score of a given word is only meaningful compared to the scores of other words, i.e., in rankings, as will be seen in the next section.

3 Experiments

In this section, we describe the protocol we followed to test the extent to which our rankings match human rankings.

3.1 Word selection

The first step was to select words to include in our analysis. For this purpose, we downloaded and extracted all the text from the latest available English Wikipedia dump[2]. We then performed tokenization, stopword, punctuation, and number removal and counted the occurrence of each token of at least 3 characters in size. Out of these tokens, we kept the 2000 most frequent.

3.2 Generating vector sets

For each word in the shortlist, we randomly selected 3000 sentences such that the corresponding word appeared exactly once within each sentence. The words that did not appear in at least 3000 sentences were removed from the analysis, reducing the shortlist's size from 2000 to 1822. Then, for each word, the associated sentences were passed through a pre-trained ELMo model[3] (Peters et al., 2018) in test mode, and the top layer representations corresponding to the word were harvested. The advantage of using ELMo's top layer embeddings is that they are the most contextual, as shown by Ethayarajh (2019). We ended up with a set of exactly 3000 1024-dimensional contextual embeddings for each word.

3.3 Dimensionality reduction

Remember that the total number of bins in the entire space is equal to $(2^l)^D$ at a given level l, which would have given us an infinite number of bins even at the first level, since the ELMo representations have dimensionality $D = 1024$. To reduce the dimensionality of the contextual embedding space, we applied PCA, trying 19 different output dimensionalities, from 2 to 20 with steps of 1. Due to the quantity and high initial dimensionality of the vectors, we used the distributed[4] version of PCA provided by the PySpark's ML Library (Meng et al., 2016).

3.4 Score computation

We computed our scores for each PCA output dimensionality, trying with 18 different hierarchies whose numbers of levels L ranged from 2 to 19. So in total, we obtained $19 \times 18 = 342$ rankings.

3.5 Ground truth rankings and baselines

We evaluated the rankings generated by our approach against several ground truth rankings that we derived from human-constructed resources.

Since the number of senses of a word is a subjective, debatable notion, and thus may vary from source to source, we included 6 ground truth rankings in our analysis, in order to minimize source-specific bias as much as possible. For sanity checking purposes, we also added two basic baseline rankings (frequency and random). We provide more details about all rankings in what follows.

3.5.1 WordNet

We used WordNet (Miller, 1998) version 3.0 and counted the number of synonym sets or "synsets" of each word.

3.5.2 WordNet-Reduced

There are very subtle differences among the WordNet senses ("synsets"), making distinguishing between them difficult and even irrelevant in some applications (Palmer et al., 2004, 2007; Brown et al., 2010; Rumshisky, 2011; Jurgens, 2013). For instance, *call* has 41 senses in the original WordNet (28 as verb and 13 as noun). Even for other words with fewer senses, like *eating* (7 senses in total), the difference between senses can be very tiny. For instance, "take in solid food" and "eat a meal; take a meal" are really close in meaning. This very fine granularity of WordNet may somewhat artificially increase the polysemy of some words.

To reduce the granularity of the WordNet

---

[2] https://dumps.wikimedia.org/

[3] We used the implementation and pre-trained weights publicly released by the authors https://allennlp.org/elmo.

[4] 15 executors with 10 GB of RAM each.

synsets, we used their sense keys[5]. They follow the format `lemma%ss_type:lex_filenum:lex_id:head_word:head_id`, where `ss_type` represents the synset type (part-of-speech tag such as noun, verb, adjective) and `lex_filenum` represents the name of the lexicographer file containing the synset for the sense (`noun.animal`, `noun.event`, `verb.emotion`, etc.). We truncated the sense keys after `lex_filenum`.

For instance, "take in solid food" and "eat a meal; take a meal" initially correspond to two different senses with keys `eat%2:34:00::` and `eat%2:34:01::`, but after truncation, they both are mapped to the same sense: `eat%2:34`. However, coarse differences in senses are still captured. For instance, *bank* "sloping land" (`bank%1:17:01::`) and *bank* "financial institution" (`bank%1:14:00::`) are still mapped to two different senses after truncation, respectively `bank%1:17` and `bank%1:14`.

3.5.3 WordNet-Domains

WordNet Domains (Bentivogli et al., 2004; Magnini and Cavaglia, 2000) is a lexical resource created in a semi-automatic way to augment WordNet with domain labels. Instead of synsets, each word is associated with a number of semantic domains. The domains are areas of human knowledge (politics, economy, sports, etc.) exhibiting specific terminology and lexical coherence. As for the two previous WordNet ground truth rankings, we simply counted the number of domains associated with each word.

3.5.4 OntoNotes

OntoNotes (Hovy et al., 2006; Weischedel et al., 2011) is a large annotated corpus comprising various text genres (news, conversational telephone speech, weblogs, newsgroups, broadcast, talk shows) with structural information and shallow semantics.

We counted the senses in the sense inventory of each word. The senses in OntoNotes are groupings of the WordNet synsets, constructed by human annotators. As a result, the sense granularity of OntoNotes is coarser than that of WordNet (Brown et al., 2010).

3.5.5 Oxford

We counted the number of senses returned by the Oxford dictionary[6], which was, at the time of this study, the resource underlying the Google dictionary functionality.

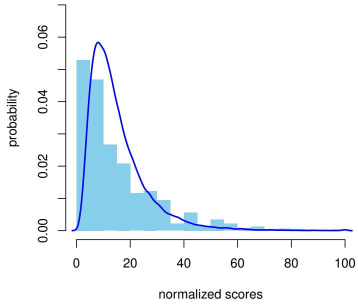


Figure 2: Average score distribution of the 5 ground truth rankings and frequency baseline (histogram) vs. average score distribution of the random baseline (**blue** curve).

3.5.6 Wikipedia

We capitalized on the Wikipedia disambiguation pages[7]. Such pages contain a list of the different categories under which one or more articles about the query word can be found. For example, the disambiguation page of the word *bank* includes categories such as geography, finance, computing (data bank), and science (blood bank). We counted the number of categories on the disambiguation page of each word to generate the ranking.

3.5.7 Frequency and random baselines

In the frequency baseline, we ranked words in decreasing order of their frequency in the entire Wikipedia dump (see subsection 3.1). The naive assumption made here is that words occurring the most have the most senses.

With the random baseline, on the other hand, we produced rankings by shuffling words. Further, we assigned them random scores by sampling from the Log Normal distribution[8], to imitate the long-tail behavior of the other score distributions, as can be seen in Fig. 2. All distributions can be seen in Fig. 6. Note that to account for randomness, all results for the random baseline are averages over 30 runs. Not every of the 1822 words included in our analysis had an entry in each of the resources described above. The lengths of each ground truth ranking are shown in Table 1.

4 Evaluation metrics

We used 6 similarity, correlation and information retrieval standard metrics to compare among methods: cosine similarity, Spearman's rho (Spearman,

---

[5] See 'Sense Key Encoding' here: `https://wordnet.princeton.edu/documentation/senseidx5wn`

[6] `www.lexico.com`

[7] www.wikipedia.org/wiki/word_(disambiguation)

[8] with mean and standard deviation 0 and 0.6 (resp.)

Ranking	# words
WN	1535
WN-reduced	1535
WN-Domains	1420
Oxford	1536
Wikipedia	1042
OntoNotes	723
Frequency & random	1822

Table 1: Length of the ground truth rankings.

1904), Kendall's tau (Kendall, 1938), precision at k, Normalized Discounted Cumulative Gain (Järvelin and Kekäläinen, 2002), and Rank Biased Overlap (Webber et al., 2010).

To ensure a fair comparison, the scores in the rankings of all methods were normalized to be in the $[0, 100]$ range before proceeding.

Also, each method played in turn the role of candidate and ground truth. This allowed us to compute not only the similarity between our rankings and the ground truth rankings, but also the similarity among the ground truth rankings themselves, which was interesting for exploration purposes.

For each pair of evaluated and ground truth method, only the parts of the rankings corresponding to the words in common (intersection) were compared. Thus, the rankings in each (candidate,ground truth) pair had equal length.

5 Implementation details

To compute our scores, we built on the code of the pyramid match kernel from the `GraKeL` Python library (Siglidis et al., 2018). We used the base R (R Core Team, 2018) `cor()` function[9] to compute the τ and ρ statistics. For RBO, we relied on a publicly available Python implementation[10]. For all other metrics, we wrote our own implementations. Full details about design choices, tokenizers, stopword lists etc., can also be found in our publicly available code repository: `https://github.com/ksipos/polysemy-assessment`.

6 Results and observations

Our rankings correlate well with human rankings. Results are shown in Fig. 3, as pairwise similarity matrices, for all six metrics. For readability, all scores are shown as percentages. For a given metric, our configuration that best matches, on average, all other methods (except random and frequency) is always shown as the first column. Since all metrics except NDCG are symmetric, we only show the lower triangles of the other matrices. For NDCG, candidate methods are shown as columns and ground truths as rows.

For each of the six evaluation metrics, it can be seen that the ranking generated by our unsupervised, data-driven method is well correlated with all human-derived ground truth rankings. This means that our method is robust to how one defines and measures correlation or similarity.

In some cases, we even very closely reproduce the human rankings. For instance, our best configurations for cosine and NDCG get almost perfect scores of 86.5 and 99.72 when compared against Wikipedia. In terms of Kendall's tau, Spearman's rho, p@k, and RBO, we are also very close to OntoNotes (scores of 49.43, 35.23, 39.53, and 33.47, resp.).

Finally, the correlation between our rankings and the human rankings can also be observed to be, everywhere, much stronger than that between the baseline rankings (random and frequency) and the human rankings.

Statistical significance. We computed statistical significance for the Spearman's rho and Kendall's tau metrics. As shown in Fig. 3, the null hypothesis that there is no correlation between our rankings and the human-derived ground truth rankings was systematically rejected everywhere, with very high significance ($p \leq 0.0001$).

However, against the random baseline, the same null hypothesis (no correlation) was accepted everywhere. Against frequency, the null was rejected, but very weakly (only at the $p \leq 0.01$ level), and with very low correlation coefficients (6.53 for Spearman and 4.44 for Kendall).

Finally, the correlation between the random and frequency rankings and the ground truth rankings is never statistically significant, except for the pair frequency/OntoNotes, but again, at a weak level ($p \leq 0.01$).

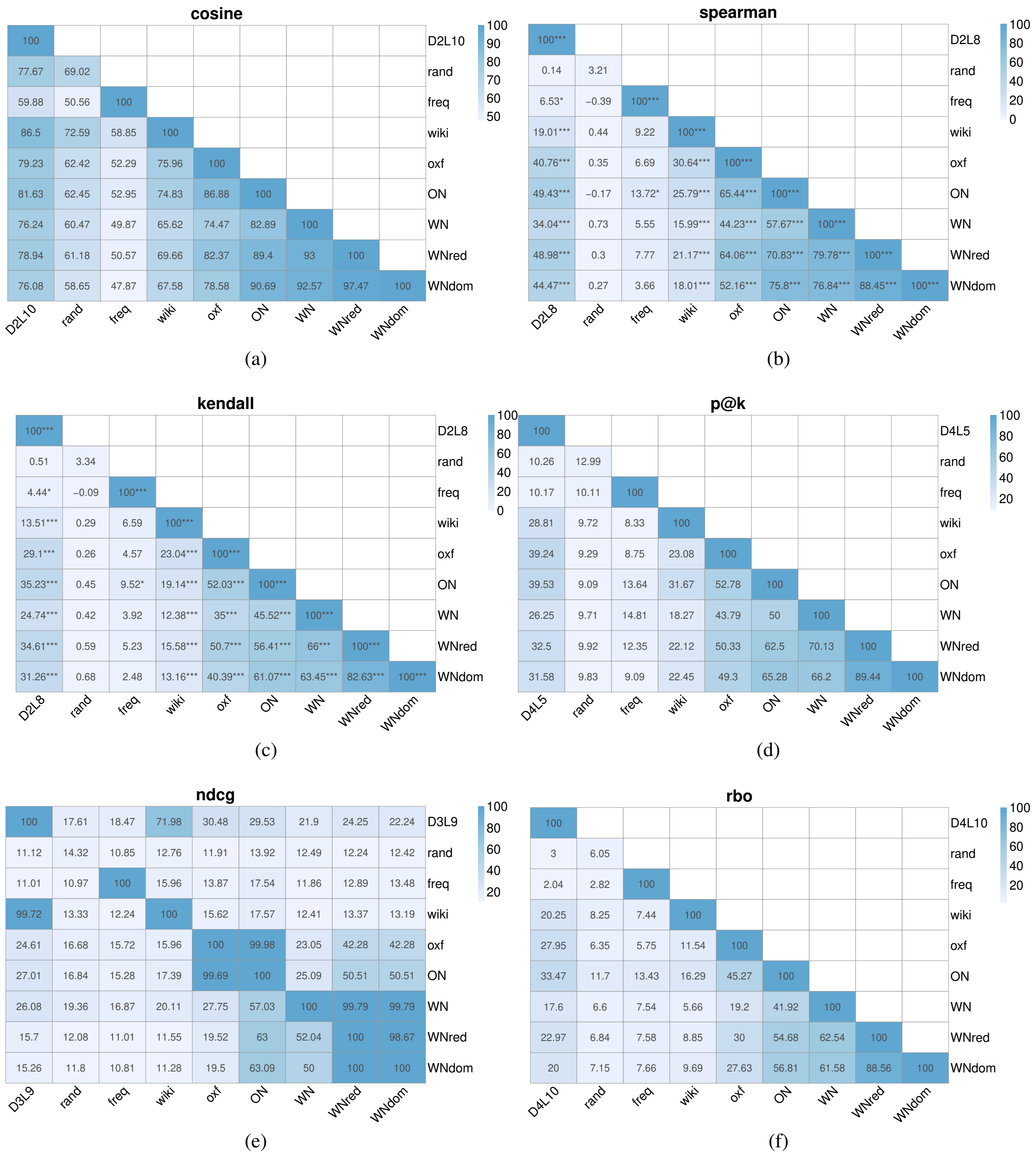
Hyperparameters have a significant impact on performance, but optimal values are consistent across metrics. First, as can be observed from Fig. 4 and Fig. 5, there is a large variability in performance when D (number of PCA dimensions) and L (number of levels in the hierarchy) vary.

However, for all six evaluation metrics, the best configurations are very similar: $D2L10$, $D2L8$,

---

[9] `https://stat.ethz.ch/R-manual/R-patched/library/stats/html/cor.html`

[10] `https://github.com/changyaochen/rbo`



cosine

	D2L10	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D2L10	100								
rand	77.67	69.02							
freq	59.88	50.56	100						
wiki	86.5	72.59	58.85	100					
oxf	79.23	62.42	52.29	75.96	100				
ON	81.63	62.45	52.95	74.83	86.88	100			
WN	76.24	60.47	49.87	65.62	74.47	82.89	100		
WNred	78.94	61.18	50.57	69.66	82.37	89.4	93	100	
WNdom	76.08	58.65	47.87	67.58	78.58	90.69	92.57	97.47	100

(a)

spearman

	D2L8	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D2L8	100***								
rand	0.14	3.21							
freq	6.53*	–0.39	100***						
wiki	19.01***	0.44	9.22	100***					
oxf	40.76***	0.35	6.69	30.64***	100***				
ON	49.43***	–0.17	13.72*	25.79***	65.44***	100***			
WN	34.04***	0.73	5.55	15.99***	44.23***	57.67***	100***		
WNred	48.98***	0.3	7.77	21.17***	64.06***	70.83***	79.78***	100***	
WNdom	44.47***	0.27	3.66	18.01***	52.16***	75.8***	76.84***	88.45***	100***

(b)

kendall

	D2L8	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D2L8	100***								
rand	0.51	3.34							
freq	4.44*	–0.09	100***						
wiki	13.51***	0.29	6.59	100***					
oxf	29.1***	0.26	4.57	23.04***	100***				
ON	35.23***	0.45	9.52*	19.14***	52.03***	100***			
WN	24.74***	0.42	3.92	12.38***	35***	45.52***	100***		
WNred	34.61***	0.59	5.23	15.58***	50.7***	56.41***	66***	100***	
WNdom	31.26***	0.68	2.48	13.16***	40.39***	61.07***	63.45***	82.63***	100***

(c)

p@k

	D4L5	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D4L5	100								
rand	10.26	12.99							
freq	10.17	10.11	100						
wiki	28.81	9.72	8.33	100					
oxf	39.24	9.29	8.75	23.08	100				
ON	39.53	9.09	13.64	31.67	52.78	100			
WN	26.25	9.71	14.81	18.27	43.79	50	100		
WNred	32.5	9.92	12.35	22.12	50.33	62.5	70.13	100	
WNdom	31.58	9.83	9.09	22.45	49.3	65.28	66.2	89.44	100

(d)

ndcg

	D3L9	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D3L9	100	17.61	18.47	71.98	30.48	29.53	21.9	24.25	22.24
rand	11.12	14.32	10.85	12.76	11.91	13.92	12.49	12.24	12.42
freq	11.01	10.97	100	15.96	13.87	17.54	11.86	12.89	13.48
wiki	99.72	13.33	12.24	100	15.62	17.57	12.41	13.37	13.19
oxf	24.61	16.68	15.72	15.96	100	99.98	23.05	42.28	42.28
ON	27.01	16.84	15.28	17.39	99.69	100	25.09	50.51	50.51
WN	26.08	19.36	16.87	20.11	27.75	57.03	100	99.79	99.79
WNred	15.7	12.08	11.01	11.55	19.52	63	52.04	100	98.67
WNdom	15.26	11.8	10.81	11.28	19.5	63.09	50	100	100

(e)

rbo

	D4L10	rand	freq	wiki	oxf	ON	WN	WNred	WNdom
D4L10	100								
rand	3	6.05							
freq	2.04	2.82	100						
wiki	20.25	8.25	7.44	100					
oxf	27.95	6.35	5.75	11.54	100				
ON	33.47	11.7	13.43	16.29	45.27	100			
WN	17.6	6.6	7.54	5.66	19.2	41.92	100		
WNred	22.97	6.84	7.58	8.85	30	54.68	62.54	100	
WNdom	20	7.15	7.66	9.69	27.63	56.81	61.58	88.56	100

(f)

Figure 3: Pairwise similarity matrices between methods. For readability, all scores are shown as percentages. For Kendal and Spearman, * and *** mean statistical significance at $p \leq 0.01$ and $p \leq 0.0001$, respectively. For a given metric, our configuration that best matches (on average) all other methods (except random and frequency) is always shown first. $DDLL$ means that the compressed contextual embedding space has D dimensions and that the hierarchy has L levels. Rand, freq, wiki, oxf, ON, WN, WNred, and WNdom are short for random, frequency, Wikipedia, Oxford, OntoNotes, WordNet, WordNet reduced, and WordNet domains. All metrics except NDCG are symmetric, hence we only show one triangle for them. For NDCG, candidate methods are shown as columns and ground truths as rows.

Sentences	Bin coordinates
it stars christopher lee as **count** dracula along with dennis waterman	$(3, 5, 1)$
the **count** of the new group is the sum of the separate counts of the two original groups	$(4, 1, 3)$
the first fight did not **count** towards the official record	$(4, 5, 1)$
five year old horatia came to **live** at merton in may 1805	$(2, 5, 2)$
it features various amounts of **live** and backstage footage while touring	$(4, 2, 4)$
first tax bills were used to pay taxes and to register **bank** deposits and bank credits	$(4, 2, 4)$
the ball nest is built on a **bank** tree stump or cavity	$(5, 2, 3)$

Table 2: Sentences containing different senses of the same word can be sampled by selecting from different bins.

Keywords	Bin coordinates
also, gas, used, system, protein, blood, new, steel, food, made	$(20, 11, 16, 9)$
first, new, one, second, later, world, national, olympic, team, games	$(19, 13, 15, 13)$
album music rock one labour number chart songs metal single	$(21, 14, 14, 16)$

Table 3: Towards automatic word sense induction: top 10 most frequent words for different bins containing the word *metal*. The bins correspond to 3 senses of metal: chemical element, Olympics (medals), and music.

$D2L8$, $D4L5$, $D3L9$, and $D4L10$[11]. Given the rather large grid we explored ($[2, 20] \times [2, 19]$ for D and L, resp.), with 342 combinations in total, we can say that all these optimal values belong to the same small neighborhood. This interpretation is confirmed by inspecting Fig. 4, where it can clearly be seen that the optimal area of the hyperparameter space is robust to metric selection and consistently corresponds to small values of D (around 3), and values of L at least above 3 or 4, ideally around 8. For larger values of L, performance plateaus (keeping D fixed). In other words, it is necessary to have some levels in the hierarchy, but having very deep hierarchies is not required for our method to work well. A benefit of having such small optimal values of D and L is their affordability, from a computational standpoint.

All rankings derived from WordNet-based resources are highly correlated. It is interesting to note that the rankings generated from OntoNotes, WordNet, WordNet reduced, and WordNet domains, all are highly similar. And this, despite the very different sense granularities they have. This means that despite the apparent differences in these resources, they all tend to produce similar polysemy rankings. The Oxford rankings tend to be part of this high-similarity cluster as well, to a lesser extent.

Frequent words are not the most polysemous. Finally, one last interesting observation is that while the frequency ranking is much better than the random ones, it still is far away from the human rankings. In other words, the frequency of appearance of a word (excluding stopwords, of course) is not as good an indicator of its polysemy as one could expect. Some words that follow this observation are ”number”, ”population”, and ”war”.

A note on ties. To assess the impact of ties on the reported results, we repeated all of our experiments multiple times with different tie-breaking strategies (e.g., random, alphabetical...). Results do not change: we find the same best parameter combinations, and the differences in the similarity matrices are minimal.

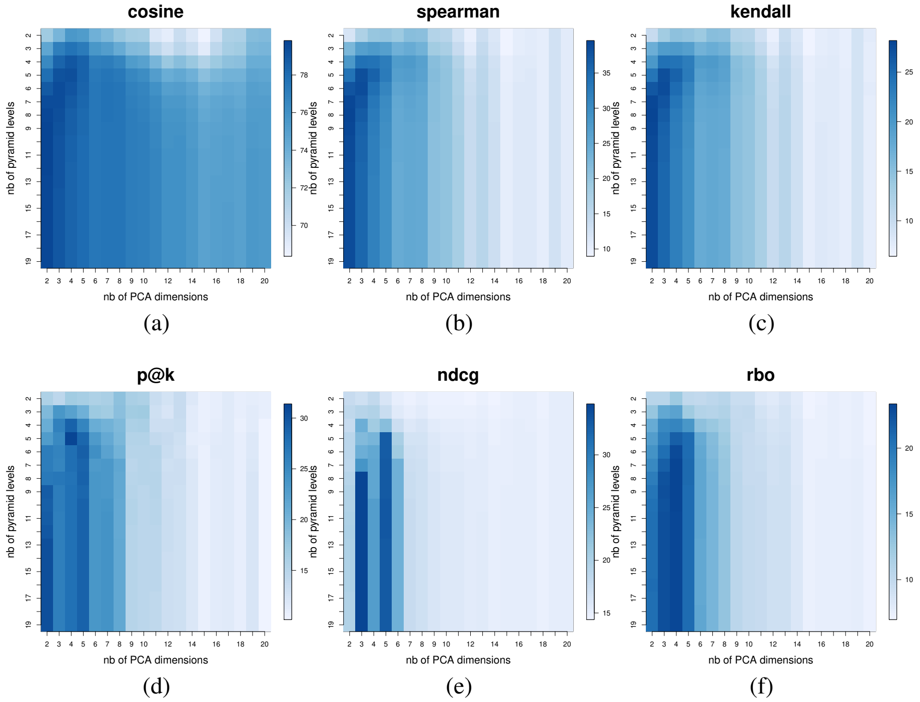
7 Other applications

Sampling diverse examples. An interesting by-product of our discretization strategy is that it can be used to select sentences containing different senses of the same word, as illustrated in Table 2. Provided a mapping, for a given word, between the sentences that were passed to the pre-trained language model and the vectors, we can sample vectors from different bins and retrieve the associated sentences. If the bins are distant enough, the sentences will contain different senses of the word. For instance, in Table 2, we can see that we are able to sample sentences containing three senses of the word *count*: (1) noble title, (2) determining the total number of, and (3) taking into account. This has many useful applications in practice, e.g., in information retrieval, NLG and conversational systems, dataset creation, etc.

Automatic word sense induction. A simple way of capitalizing on our binning strategy to create word sense inventories would consist in (1) selecting distant bins for a given word, and (2) labeling the selected bins with senses. Both steps can be performed automatically. While this will be investigated in future work, we still give, as a proof of concept, an example in Table 3. In this example,

---

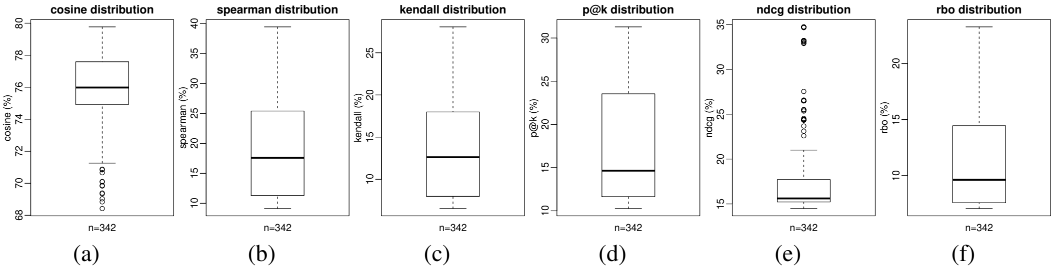
[11] for RBO, $D4L10$ and $D4L8$ had the same score.



(a) (b) (c)

(d) (e) (f)

Figure 4: Performance (color scale) vs. number of PCA dimensions (x axis) vs. number of levels in the hierarchy (y axis).



(a) (b) (c) (d) (e) (f)

Figure 5: Performance distributions over the 342 values in the discrete hyperparameter space (grids of Fig. 4).

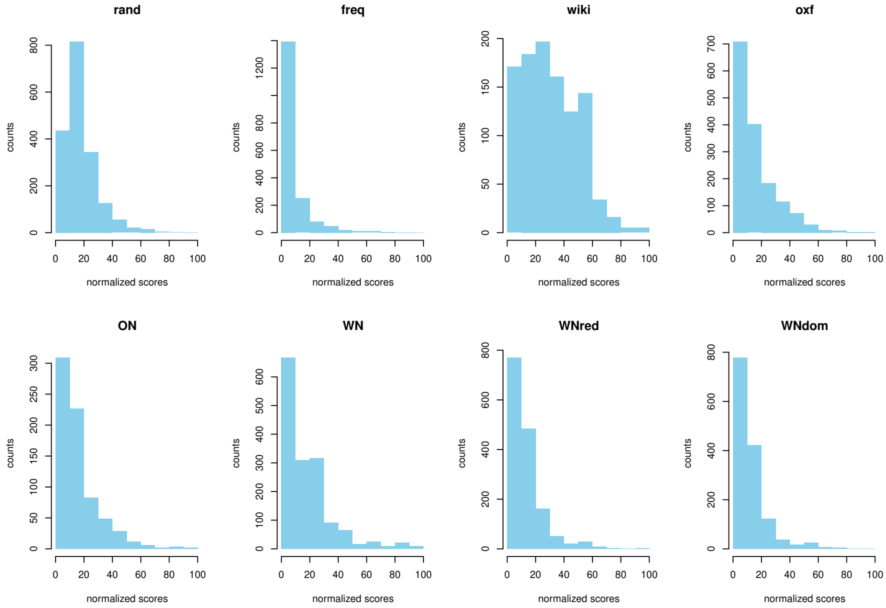


Figure 6: Normalized ranking score distributions for the random and frequency rankings and the human-derived ground truth rankings.

keywords are extracted from distant bins containing the word *metal*, and different senses are retrieved.

8 Related work

Task. Several previous efforts have interested themselves in creating sense inventories without human experts. As an example, in Rumshisky (2011); Rumshisky et al. (2012) [12], Amazon Mechanical Turk (AMT) workers are given a set of sentences containing the target word and one sentence that is randomly selected from this set as a target sentence. Workers are then asked to judge, for each sentence, whether the target word is used in the same way as in the target sentence. This creates an undirected graph of sentences where clustering can be applied to find senses. To label clusters with senses, one has to inspect the sentences in each cluster manually.

More recently, Jurgens (2013)[13] compared three annotation methodologies for gathering word sense labels on AMT. The methods compared are Likert scales, two-stage select and rate, and the difference between counts of when senses were rated best/worst. Regardless of the strategy, inter-annotator agreement remains low (around 0.3).

Methodology. In the original ELMo paper, Peters et al. (2018) have shown that using contextual word representations (through nearest-neighbor matching) improves word sense disambiguation. Hadiwinoto et al. (2019); Coenen et al. (2019) showed that this technique works well with BERT too. Pasini et al. (2020) uses a combination of BERT embeddings and a knowledge-based WSD model to generate word sense distributions, while Giulianelli et al. (2020) uses clustering over the embeddings to detect semantic shifts.

Our approach is also related in spirit to pyramid matching (Nikolentzos et al., 2017; Grauman and Darrell, 2007; Lazebnik et al., 2006). This kernel-based method originated in computer vision. It computes the similarity between objects by placing a sequence of increasingly coarser grids over the feature space and taking a weighted sum of the number of matches occurring at each level. Matches found at finer resolutions are weighted more than matches found at coarser resolutions.

9 Conclusion

We proposed a novel unsupervised, fully data-driven geometrical approach to estimate word polysemy. Our approach builds multiresolution grids in the contextual embedding space. Through rigorous experiments, we showed that our rankings are well correlated (with strong statistical significance) to 6 different human rankings, for 6 different metrics. Such fully data-driven rankings of words according to polysemy can help in creating new sense inventories, but also in validating and interpreting existing ones. Increasing the quality and consistency of sense inventories is a key first step of the word sense disambiguation pipeline. We also showed that our discretization could be used, at no extra cost, to sample contexts containing different senses of a given word, which has useful applications in practice. Finally, the unsupervised nature of our method makes it applicable to any language.

While our scores are a good proxy for polysemy, they are not equal to word sense counts. Moreover, we do not label each sense. Future work should address these challenges by, e.g., automatically selecting bins of interest and generating labels for them (see section 7).

Future work should also perform some sort of extrinsic evaluation. For instance, the Word-in-Context task (Pilehvar and Camacho-Collados, 2018) could be used, where two occurrences would be classified as having the same meaning if their two vectors fall in the same bin.

Another direction is investigating how different contextual embeddings (e.g., BERT, BART) impact our rankings, including in languages other than English (Eddine et al., 2020; Cañete et al., 2020), and low-resource languages.

Acknowledgments

We thank the three anonymous reviewers for their helpful feedback. We also thank Giannis Nikolentzos for helpful discussions about pyramid matching. The GPU used in this study was donated by the Nvidia corporation as part of their GPU grant program. This work was supported by the LinTo project.

References

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

---

[12] We asked the authors to share annotations with us to use as ground truth, but they were unable to do so.

[13] same as footnote 12.

Luisa Bentivogli, Pamela Forner, Bernardo Magnini, and Emanuele Pianta. 2004. Revising the wordnet domains hierarchy: semantics, coverage and balancing. In *Proceedings of the workshop on multilingual linguistic resources*, pages 94–101.

Susan Windisch Brown, Travis Rood, and Martha Palmer. 2010. Number or nuance: Which factors restrict reliable word sense annotation? In *LREC*.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda Viégas, and Martin Wattenberg. 2019. Visualizing and measuring the geometry of bert. *arXiv preprint arXiv:1906.02715*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Moussa Kamal Eddine, Antoine J-P Tixier, and Michalis Vazirgiannis. 2020. Barthez: a skilled pre-trained french sequence-to-sequence model. *arXiv preprint arXiv:2010.12321*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*.

Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2020. Analysing lexical semantic change with contextualised word representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3960–3973, Online. Association for Computational Linguistics.

Kristen Grauman and Trevor Darrell. 2007. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8(Apr):725–760.

Christian Hadiwinoto, Hwee Tou Ng, and Wee Chung Gan. 2019. Improved word sense disambiguation using pre-trained contextualized word representations. *arXiv preprint arXiv:1910.00194*.

Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

David Jurgens. 2013. Embracing ambiguity: A comparison of annotation methodologies for crowdsourcing word sense labels. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–562.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE.

Bernardo Magnini and Gabriela Cavaglia. 2000. Integrating subject field codes into wordnet. In *LREC*, pages 1413–1418.

Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, et al. 2016. Mllib: Machine learning in apache spark. *The Journal of Machine Learning Research*, 17(1):1235–1241.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Giannis Nikolentzos, Polykarpos Meladianos, and Michalis Vazirgiannis. 2017. Matching node embeddings for graph similarity. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Martha Palmer, Olga Babko-Malaya, and Hoa Trang Dang. 2004. Different sense granularities for different applications. In *Proceedings of the 2nd International Workshop on Scalable Natural Language Understanding (ScaNaLU 2004) at HLT-NAACL 2004*, pages 49–56.

Martha Palmer, Hoa Trang Dang, and Christiane Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13(2):137–163.

Tommaso Pasini, Federico Scozzafava, and Bianca Scarlini. 2020. Clubert: A cluster-based approach for learning sense distributions in multiple languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4008–4018.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Anna Rumshisky. 2011. Crowdsourcing word sense definition. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 74–81. Association for Computational Linguistics.

Anna Rumshisky, Nick Botchan, Sophie Kushkuley, and James Pustejovsky. 2012. Word sense inventories by non-experts. In *LREC*, pages 4055–4059.

Giannis Siglidis, Giannis Nikolentzos, Stratis Limnios, Christos Giatsidis, Konstantinos Skianis, and Michalis Vazirgiannis. 2018. Grakel: A graph kernel library in python. *arXiv preprint arXiv:1806.02193*.

Charles Spearman. 1904. The proof and measurement of association between two things.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

William Webber, Alistair Moffat, and Justin Zobel. 2010. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems (TOIS)*, 28(4):1–38.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Mega-COV: A Billion-Scale Dataset of 100+ Languages for COVID-19

Muhammad Abdul-Mageed, AbdelRahim Elmadany, El Moatez Billah Nagoudi,
Dinesh Pabbi, Kunal Verma, Rannie Lin
Natural Language Processing Lab
University of British Columbia
{muhammad.mageed,a.elmadany,moatez.nagoudi}@ubc.ca,
{dinesh09,vkunal96}@ece.ubc.ca, krieyalam@gmail.com

Abstract

We describe `Mega-COV`, a billion-scale dataset from Twitter for studying COVID-19. The dataset is diverse (covers 268 countries), longitudinal (goes as back as 2007), multilingual (comes in 100+ languages), and has a significant number of location-tagged tweets (~ 169M tweets). We release tweet IDs from the dataset. We also develop two powerful models, one for identifying whether or not a tweet is related to the pandemic (best F_1=97%) and another for detecting misinformation about COVID-19 (best F_1=92%). A human annotation study reveals the utility of our models on a subset of `Mega-COV`. Our data and models can be useful for studying a wide host of phenomena related to the pandemic. `Mega-COV` and our models are publicly available.

1 Introduction

The seeds of the coronavirus disease 2019 (COVID-19) pandemic are reported to have started as a local outbreak in Wuhan (Hubei, China) in December, 2019, but soon spread around the world (WHO, 2020). As of January 24, 2021, the number of confirmed cases around the world exceeded 99.14M and the number of confirmed deaths exceeded 2.13M.[1] In response to this ongoing public health emergency, researchers are mobilizing to track the pandemic and study its impact on all types of life in the planet. Clearly, the different ways the pandemic has its footprint on human life is a question that will be studied for years to come. Enabling scholarship on the topic by providing relevant data is an important endeavor. Toward this goal, we collect and release `Mega-Cov`, a billion-scale multilingual Twitter dataset with geo-location information.

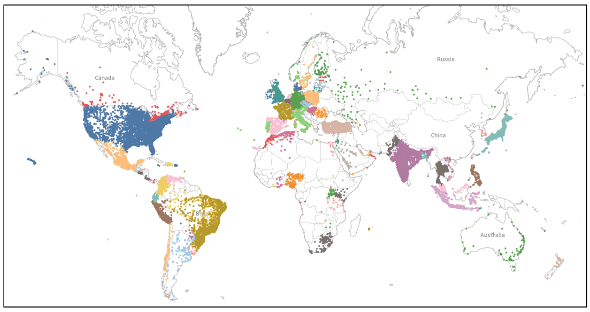


Figure 1: Global coverage of `Mega-COV` based on our geo-located data. Each dot is a city. Contiguous cities of the same color belong to the same country.

As a result of the pandemic, most countries around the world went into lockdown and the public health emergency has restricted physical aspects of human communication considerably. As hundreds of millions of people spend more time sheltering in place, communication over social media became more important than ever. In particular, the content of social media communication promises to capture significant details about the lives of tens of millions of people. `Mega-Cov` is intended as a repository of such a content.

There are several ongoing efforts to collect Twitter data, and our goal is to complement these. More specifically, we designed our methods to harvest a dataset that is unique in multiple ways, as follows: **Massive Scale**: Very large datasets lend themselves to analyses that are not possible with smaller data. Given the global nature of COVID-19, we realize that a large-scale dataset will be most useful as the scale allows for slicing and dicing the data across different times, communities, languages, and regions that are not possible otherwise. For this reason, we dedicated significant resources to harvesting and preparing the dataset. `Mega-COV` has solid international coverage and brings data from 1M users from 268 countries (see Section 3.1). Overall, our dataset has ~ 1.5B tweets (Section 2). This is one order of magnitude larger than #COVID-19 (Chen et al., 2020), the largest dataset we know

---

[1] Source: The Center for Systems Science and Engineering, Johns Hopkins University. Dashboard: `https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3402–3420
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

of ($\sim$ 144M tweets as of June 1, 2020).[2]

Topic Diversity: We do not restrict our collection to tweets carrying certain hashtags. This makes the data general enough to involve content and topics directly related to COVID-19, regardless of existence of accompanying hashtags. This also allows for investigating themes that may not be directly linked to the pandemic but where the pandemic may have some bearings which should be taken into account when investigating such themes. This is important because users can, and indeed do, post about activities impacted by the health crisis without using any hashtags. In fact, users may not mention COVID-19 at all, even though what they are posting about could be affected by the pandemic one way or another (e.g., "eating habits", "shopping behavior"). Section B and Section C in the Appendix provide a general overview of issues discussed in the dataset.

Longitudinal Coverage: We collect multiple data points (up to 3,200) from each user, with a goal to allow for comparisons between the present *and* the past across the same users, communities, and geographical regions (Section 3.2). Again, this is desirable since without data from pre-COVID-19 time it will be challenging to hold any such comparisons. For example, some users may have stopped posting about "exercising" during the pandemic but we cannot definitely identify this without access to these users' previous data where they may have been posting about their physical activities.

Language Diversity: Since our collection method targets users, rather than hashtag-based content, `Mega-COV` is *linguistically diverse*. In theory, any language posted to Twitter by a user whose data we have collected should be represented. Based on Twitter-assigned language codes, we identify a total of 65 languages. However, applying two different language detection tools to the whole dataset, we identify more than 100 languages. (Section 3.3).

No Distribution Shift: Related to the two previous points, but from a machine learning perspective, by collecting the data without conditioning on existence of specific (or any) hashtags we avoid introducing distribution bias. In other words, the data can be used to study various phenomena in-the-wild. This warrants more *generalizable* findings and models.

A dataset as large as `Mega-COV` can be hard to navigate. In particular, an informative description of the dataset is necessary for navigating it. In this paper, we provide an explanation of a number of global aspects of the dataset, including its geographic, temporal, and linguistic coverage. We also provide a high-level content analysis of the data, and explore user sharing of content from particular web domains with a focus on news media. In the context of our investigation of `Mega-COV`, we make an array of important discoveries. For example, we strikingly discover that, perhaps for the first time in Twitter history, *users address one another and retweet more than they post tweets*. We also find a noticeable rise in ranks for news sites (based on how frequent their URLs are shared) during 2020 as compared to 2019, with *a shift toward global (rather than local) news media*. A third finding is how *use of the Twitter platform surged in March*, perhaps making it the busiest time in the history of the network.

Furthermore, we develop two groups of effective neural models: **(1) COVID-relevance models** (for detecting whether a tweet is related to COVID-19 or not). **(2) COVID-misinformation models** (for detecting whether a text carries fake information or not). In addition to releasing our best models, we also apply them to a total of 30M tweets from `Mega-COV` and release our tags to accelerate further research on the topic.

The rest of the paper is organized as follows: In Section 2, we describe our data collection methods. Section 3 is where we investigate geographic, linguistic, and temporal dimensions of our data. We describe our models for detecting COVID-19 tweets and COVID-misinformation in Section 4. Section 5 is where we apply our relevance and misinformation models to a large sample of `Mega-COV`. Section 6 is about data release and ethics. We provide a literature review in Section 7, and conclude in Section 8.

2 Data Collection

To collect a sufficiently large dataset, we put crawlers using the Twitter streaming API[3] on Africa, Asia, Australia, Europe, North America, and South America starting in early January, 2020. This allows us to acquire a diverse set of tweets

---

[2] Both our own dataset and that of Chen et al. (2020) are growing over time. All our statistics in the current paper are based on our collection as of May 15, 2020. As of October 6, 2020, authors of #COVID-19 report 649.9M tweets on their GitHub (`https://github.com/echen102/COVID-19-TweetIDs`), and *our own dataset has exceeded 5B tweets*.

[3] API link: `https://github.com/tweepy/`

Data	Tweets	Retweets	Replies	All
2007-2020	612M	507M	369M	1.5B
2020	122M	174M	129M	425M
Users	1M	976K	994K	1M

Table 1: Distribution of tweets, retweets, and replies in `Mega-COV` (numbers rounded).

from which we can extract a random set of user IDs whose timelines (up to 3,200 tweets) we then iteratively crawl every two weeks. This gave us data from July 30^{th}, 2020 backwards, depending on how prolific of a poster a user is (see Table 4a for a breakdown.). In this paper, we describe and analyze the version of `Mega-COV` collected up to May 15, 2020 and use the term `Mega-COV` to refer it. `Mega-COV` comprises a total of $1,023,972$ users who contribute $1,487,328,805$ tweets. For each tweet, we collect the whole `json` object. This gives us access to various types of information, such as user location and the language tag (including "undefined") Twitter assigns to each tweet. We then use the data streaming and processing engine, Spark, to merge all user files and run our analyses. To capture a wide range of behaviors, we keep *tweets*, *retweets*, and *responses* (i.e., direct user-to-user interactions) as independent categories. Table 1 offers a breakdown of the distribution of the different types of posts in `Mega-COV`. Tweet IDs of the dataset are publicly available at our GitHub[4] and can be downloaded for research. To the extent it is possible, we intend to provide semi-regular updates to the dataset repository.

3 Exploring Mega-COV

3.1 Geographic Diversity

A region from which a tweet is posted can be associated with a specific 'point' location or a Twitter place with a 'bounding box' that describes a larger area such as city, town, or country. We refer to tweets in this category as *geo-located*. A smaller fraction of tweets are also *geo-tagged* with longitude and latitude. As Table 2 shows, `Mega-COV` has $\sim$ 187M geo-located tweets from $\sim$ 740K users and $\sim$ 31M geo-tagged tweets from $\sim$ 267K users. Table 2 also shows the distribution of tweets and users over the top two countries represented in the dataset, the U.S. and Canada (North America), and other locations (summed up as one category, but see also Table 3 for countries in the data by continent).

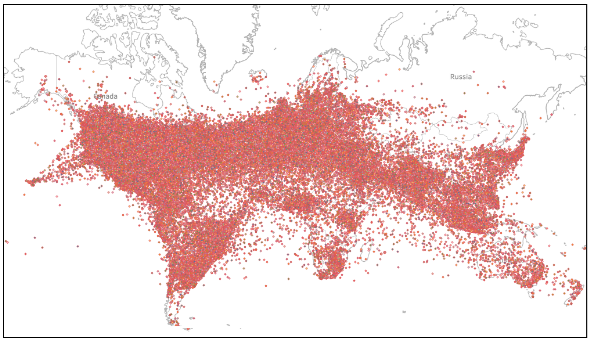


Figure 2: World map coverage of `Mega-COV`. Each dot is a point co-ordinate (longitude and latitude) from which at least one tweet was posted. Clearly, users tweet while traveling, whether by air or sea.

As explained, to allow comparisons over time (including behavioral changes during COVID-19), we include pre-2020 data in `Mega-COV`. For the year 2020, `Mega-COV` has $\sim$ 66M geo-located tweets from $\sim$ 670K users and $\sim$ 3M geo-tagged tweets from $\sim$ 109K users.[5] We note that significant parts from the data could still belong to the different countries but just not geo-located in the original `json` files retrieved from Twitter. Figure 2 shows actual point co-ordinates of locations from which the data were posted. Figure 3 shows the geographical diversity in `Mega-COV` based on *geo-located* data. We show the distribution in terms of the number of cities over *the 20 countries from which we retrieved the highest number of locations* in the dataset, broken by `all-time` and the year `2020`. Overall, `Mega-COV` has data posted from a total of $167,202$ cities that represent 268 countries. Figure A.2 in Appendix A shows the distribution of data over countries. The top 5 countries in the data are *the U.S., Canada, Brazil, the U.K.*, and *Japan*. As we mention earlier, other top countries in the data across the various continents are shown in Table 3.

3.2 Temporal Coverage

Our goal is to make it possible to exploit `Mega-COV` for comparing user social content over time. Since we crawl user timelines, the dataset comprises content going back as early as 2007. Figure 4a shows the distribution of data over the period 2007-2020. Simple frequency of user posting shows a surge in Twitter use in the period of Jan-April 2020 compared to the same months in 2019 (see Figure A.1 in Appendix A). Indeed, we identify 40.53% more posting during the first 4 months

---

[4] Accessible at: `https://github.com/UBC-NLP/megacov`.

[5] The dataset has $\sim$ 134K "locations" which we could not resolve to a particular country using only the json information.

	Geolocated	Tweeted From			Geotagged	Tweeted From		
		Canada	U.S.	Other		Canada	U.S.	Other
All-Time	186,939,854	16,459,655	70,756,282	99,723,917	31,392,563	3,600,952	11,449,400	16,342,211
All-Users	739,645	102,388	327,213	463,673	266,916	47,622	117,096	165,860
2020	65,584,908	3,331,720	24,259,973	37,993,215	2,942,675	246,185	1,187,131	1,509,359
2020-Users	670,314	61,205	254,067	392,627	109,348	14,525	43,486	62,837

Table 2: `Mega-COV` geolocated and geotagged users and their tweets from `North America` vs. `Other` locations. See also Table 3 for statistics from top countries by continent.

Continent	Country	All		2020	
		Geo-Located	Users	Geo-Located	Users
Africa	Nigeria	1,876,879	16,220	1,057,742	14,872
	South Africa	1,503,181	9,751	692,367	6,373
	Egypt	873,079	8,840	452,738	5,900
	Ghana	373,996	3,942	202,470	3,089
	Kenya	373,667	4,480	172,796	3,026
Asia	Japan	7,646,901	32,038	2,752,890	23,773
	Indonesia	4,540,286	22,893	1,871,154	18,056
	Spain	4,327,475	43,236	1,431,567	20,902
	Philippines	4,078,410	15,477	1,636,265	11,011
	India	3,107,917	33,931	1,576,549	27,940
	Saudi Arabia	2,158,584	18,402	833,634	15,087
Australia	Australia	1,179,205	12,090	352,215	5,454
Europe	UK	11,714,012	70,787	2,970,848	44,420
	Turkey	5,067,118	32,589	1,463,550	25,477
	France	2,030,523	36,017	729,500	12,497
	Italy	1,829,369	27,071	527,648	8,308
	Germany	1,272,339	24,215	385,306	7,412
North America	US	69,515,949	327,213	23,578,430	254,067
	Canada	16,066,337	102,388	3,200,804	61,205
	Mexico	3,665,791	36,190	1,106,352	17,406
South America	Brazil	15,879,664	48,339	8,060,537	41,277
	Argentina	3,142,778	14,576	1,298,381	10,901
	Colombia	1,612,765	10,319	629,426	6,884
	Chile	1,003,459	6,212	378,770	3,674
	Ecuador	447,250	3,435	170,098	2,221

Table 3: Distribution of data over top countries per continent in `Mega-COV` (*all data* vs. *2020*).

of 2020 compared to the same period in 2019. This is expected, both due to physical distancing and a wide range of human activity (e.g., "work", "shopping") moving online. More precisely, moving activities online causes users to be on their machines for longer times and hence have easier access to social media. The clear spike in the month of March 2020 is striking. It is particularly so given a *shifted pattern of use: retweeting and replying (to others) are both observably more frequent than tweeting itself*. This especially takes place during the month of March, and somewhat continues in April, as shown in Figure 4b. Figure 4a and Figure 4b also show a breakdown of tweets, retweets, and replies. ***A striking discovery is that, for 2020, users are engaged in conversations with one another more than tweeting directly to the platform.*** This may be the first time this pattern exists, perhaps in the history of the network. At least based on our massive dataset, this conclusion can be made. In addition, for 2020, we also see users retweeting more than tweeting. Based on `Mega-COV`, this is also happening for the first time.

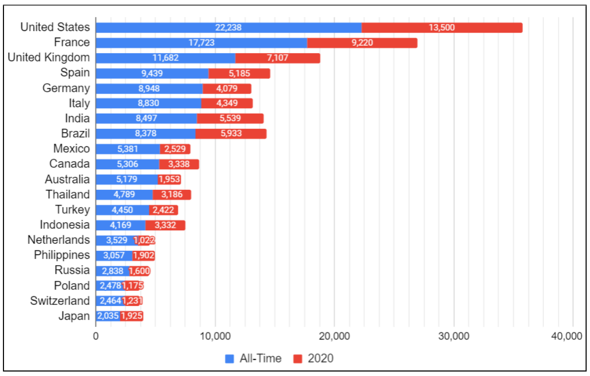


Figure 3: Geographical diversity in `Mega-COV` based on *geo-located* data.

3.3 Linguistic Diversity

We perform the language analysis based on tweets (n=$\sim$ 1.5B), including retweets and replies. Twitter assigns 65 language ids to $\sim$ 1.4B tweets, while the rest are tagged as "und" (for "undefined"). `Mega-COV` has $\sim$ 104M ($\sim$ 7%) tweets tagged as "und". We run two language identification tools, langid (Lui and Baldwin, 2012) and Compact Language Detector (Ooms and Sites, 2018)[6] langid (Lui and Baldwin, 2012),[7] on the whole dataset (including tweets tagged "und" by Twitter).[8] After merging language tags from Twitter and the 2 tools, we acquire a total of 104 labels. This makes `Mega-COV` very linguistically rich. Table 4 shows the top 20 languages identified by Twitter (left) and the top 20 languages tagged by one of the two tools, langid (Lui and Baldwin, 2012), *after* removing the 65 Twitter languages (right).

4 Models

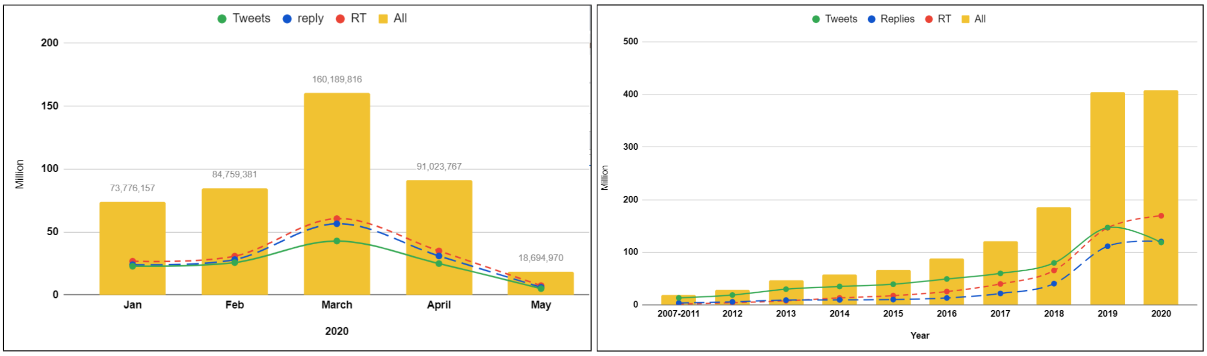
We develop two groups of models suited for answering important questions related to COVID-19, including making use of `Mega-COV`. These are **(1) COVID-relevance**, where a classifier will

---

[6] `https://code.google.com/p/cld2`.

[7] `https://github.com/saffsd/langid.py`

[8] As (Lui and Baldwin, 2014) point out, langid makes error on Twitter data. For this reason, we opted for adding predictions from CLD2.



(a) Twitter user activity for Jan-May, 2020.

(b) Distribution of `Mega-COV` (2007-2020)

Figure 4: Data distribution and user activity.

Lang	Freq	Lang	Freq
English (en)	900M	Hebrew (he)	1.2M
Spanish (es)	122M	Croatian (hr)	685K
Portuguese (pt)	79.5M	Maltese (mt)	325K
Japanese (ja)	46.6M	Slovak (sk)	246K
Arabic (ar)	45M	BI (id)	208K
Indonesion (in)	37M	Latin (la)	183K
French (fr)	29.5M	Bosnian (bs)	143K
Turkish (tr)	28.5M	Dzongkha (dz)	137.8K
Tagalog (tl)	19M	Swahili (sw)	92K
Italian (it)	8.8M	Azerbaijani (az)	68.9K
Thai (th)	7.7M	Quechua (qu)	61K
Hindi (hi)	7M	Albanian (sq)	61K
Dutch (nl)	6.9M	Malay (ms)	59K
Russuian (ru)	6.2M	Kinyarwanda (rw)	56.8K
German (de)	6M	Esperanto (eo)	55K
Catalan (ca)	3.5M	Javanese (jv)	53K
Korean (ko)	2.9M	Xhosa (xh)	47.7K
Haitian Creole (ht)	2.8M	Irish (ga)	44.6K
Polish (pl)	2.4M	Kurdish (ku)	43K
Estonain (et)	2.1M	Volapük (vo)	41K

Table 4: Top 20 languages assigned by Twitter (left) and top 20 languages assigned by langid (right) in `Mega-COV`. **BI:** Bahasa Indonesia.

label a tweet as *relevant* to COVID-19 or *not* and **(2) COVID-misinformation**, where a model predicts text veracity pertaining COVID-19 (i.e., whether a text carries *true* or *fake* information related to the pandemic). We now describe our methods.

4.1 Methods

For *all* our models, we fine-tune 3 popular pre-trained language models: (1) Multilingual cased BERT (mBERT) (Devlin et al., 2018) and (2-3) XLM Roberta base and large (XLM-R_{Base}, XLM-R_{Large}) (Conneau et al., 2020). The mBERT and XLM-R_{Base} models have similar architectures, with 12 layers each with 12 attention heads, and 768 hidden units. XLM-R_{Large} has 24 layers each with 16 attention heads, and $1,024$ hidden units. While all the 3 models use a masking objective, the XLM-R models do not include the next sentence prediction objective used in BERT.

4.2 Hyper-Parameters and Optimization

For each model, we use the same pre-processing in the respective code released by the authors. For *all* models, we typically use a sequence length of 50 tokens. We use a learning rate of $5e{-}6$ and a batch size of 32. We train each model for 20 epochs and identify the best epoch on a development set. We report performance on both development and test sets. We describe our baseline for each of the relevance and misinformation models in the respective sections below. We now introduce each of these two model groups.

4.3 COVID-Relevance Models

Data. Our COVID-relevance models predict whether a tweet is *related* to COVID-19 or *not* (i.e., not related). To train the models, we sample $\sim$ 2.3M multilingual tweets (65 languages) collected with COVID-19 hashtags from (Chen et al., 2020) and use them as our positive class (i.e., related to COVID-19). Examples of hashtags include *#Coronavirus*, *#covid-19*, and *#pandemic*. That is, we use the hashtags as a proxy for labels. This type of *distant supervision* has been validated and widely used in many NLP models (Go et al., 2009; Mohammad and Kiritchenko, 2015; Abdul-Mageed and Ungar, 2017). For the negative class (i.e., not related to COVID-19), we use a random sample of $\sim$ 2.3M from the 2019 part (Jan-Nov) of `Mega-COV`. More description of the dataset we created for training the relevance models and the distribution of the data over the various languages is in Table E.1 (Appendix E).

Splits and Training. We split the data into 80%

Model	DEV		TEST	
	Acc	F_1	Acc	F_1
Baseline I	55.10	71.05	54.99	70.96
Baseline II	79.88	88.81	75.33	85.93
mBERT	97.35	97.33	97.39	97.37
XLM-R_{Base}	97.72	97.70	97.71	97.69
XLM-R_{Large}	**97.92**	**97.90**	**97.95**	**97.93**

Table 5: Performance of COVID-relevance models. **Baseline I:** Majority class in TRAIN. **Baseline II:** A model that chooses the majority class (related class) 75% of the time.

TRAIN (n=3,146,334), 10% DEV (n=393,567), and 10% TEST (n=392,918). We then remove all hashtags which were used by (Chen et al., 2020) for collecting the data and fine-tune each of the 3 language models on TRAIN.

Results. As shown in Table 5, XLM-R_{Large} acquires best results with 97.95 *acc* and 97.93 macro F_1 on TEST. These results are significantly better than a majority class baseline (based on TRAIN) and another arbitrarily-chosen (yet quite competitive) baseline model that chooses the related class (majority class in TRAIN) 75% of the time.

Model Generalization. Our COVID-relevance models are trained with distant supervision (hashtags as surrogate labels). It is conceivable that content related to COVID-19 would still occur in real world without accompanying hashtags. To test the extent to which our best model would perform on external data, we evaluate it on two external Twitter datasets, CoAID (Cui and Lee, 2020) and ReCOVery (Zhou et al., 2020), both of which are claimed by the authors to be completely (100%) related to COVID-19.[9]

As Tabel 6 shows, We do observe a drop in model performance as compared to our best model on our own TEST set in Table 5 (acc drops on average by 15.5% and 7.6% F_1). However, the best model is still highly effective. It acquires an average acc of 82.46% and F_1 of 90.38% on the CoAID and ReCOVery datasets. We now introduce our misinformation models.

Data	Acc	F_1
COAID	76.25	86.52
ReCOVery	89.46	94.44
Average	82.46	90.38

Table 6: Performance of our COVID-relevance models on the Twitter data in CoAID, ReCOVery, and CoAID+ReCOVery.

4.4 COVID-Misinformation Models

To train models for detecting the veracity of news related to COVID-19, we exploit two recent and publicly available fake news datasets (in English): CoAID (Cui and Lee, 2020), and ReCOVery (Zhou et al., 2020). We now describe each of these datasets:

	Fake			True		
	Claims	News	Tweets	Claims	News	Tweets
CoAID	839	837	10,900	376	2716	149,343
ReCOVery	-	665	26,418	-	1,364	114,402
Total	839	1,502	37,318	376	4,080	263,745

Table 7: COVID-19 Misinformation Datasets.

	Tweets					
	Fake			True		
	TRAIN	DEV	TEST	TRAIN	DEV	TEST
CoAID	8,072	1,009	1,009	110,076	13,759	13,759
ReCOVery	18,272	2,284	2,284	86,437	10,805	10,805
CoAID*	8,072	163	171	110,076	6,314	6,388
ReCOVery*	18,272	154	139	86,437	1,218	1,263

Table 8: Statistics of CoAID and ReCOVery datasets across the data splits. CoAID* and ReCOVery* are de-duplicated versions.

CoAID. Cui and Lee (2020) present a **C**ovid-19 he**A**thcare m**I**sinformation **D**ataset (CoAID), with diverse COVID-19 healthcare misinformation, including fake news on websites and social platforms, along with related user engagements (i.e., tweets and replies) about such news. CoAID includes $3,235$ news articles and claims, $294,692$ user engagement, and 851 social platform posts about COVID-19. The dataset is collected from December 1, 2019 to July 1, 2020. Table 7 shows class distribution of news articles and tweets in CoAID. More information about CoAID is in Appendix F.

ReCOVery. Zhou et al. (2020) choose 60 news publishers with 'extreme' levels of credibility (i.e., *true* vs. *fake* classes) from an original list of $\sim 2,000$ to collect a total of $2,029$ news articles on COVID-19, published between January and May

---

[9] Each of the two datasets are also labeled for fake news (*true* vs. *fake*) focused on COVID-19, but our focus here is exclusively on using the two datasets as gold-labeled TEST sets for evaluating our COVID-relevance model. Note that we will use these two datasets again as explained in Section 4.4 as well.

2020. They also collect 140, 820 tweets related to the news articles, considering those tweets related to true articles to be true and vice versa. Table 7 shows class distribution of news articles and tweets in ReCOVery.

Splits and Cleaning. Table 8 shows the distribution of tweets in CoAID and Recovery *before* and *after* the de-duplication process. As Table 8 shows, de-duplication results in significantly reducing the sizes of DEV and TEST sets in the two resources. The distribution of news article is shown in Table F.1 (Appendix F).

Training. We use both CoAID and ReCOVery after de-duplication for training neural models to detect fake news related to Covid-19. Using the same hyper-parameters and training setup as the COVID-relevance models, we fine-tune the pre-trained language models on the Twitter dataset and the news dataset, independently.[10] Since `Mega-COV` is a social media dataset, we only focus on training Twitter models here and provide the news models in Appendix F. For the *Twitter models*, we develop one model on CoAID, another on ReCOVery, independently, and a third model for CoAID+ReCOVery (concatenated). Again, for each of these 3 datasets, we fine-tune on TRAIN and identify the best model on DEV. We then report the best model on both DEV and TEST.

Results. Since our focus is on detecting *fake* texts, we show results on the positive class only in Table 9. We report results in terms of *precision*, *recall* and F_1. Our baseline is a small LSTM with 2 hidden layers, each of which has 50 nodes. We add a dropout of 0.2 after the first layer and arbitrarily train the LSMT for 3 epochs. As Table 9 shows, our best results for fake tweet detection on TEST for CoAID is at 90% F_1 (mBERT/XLM-R$_{\text{Large}}$), for ReCOV 68% (mBERT), and for these two combined is 92%. All results are above the LSTM baseline. We show results of the COVID-misinformation *news* models in Table F.2 (Appendix F).

Data	Model	DEV			TEST		
		Precision	**Recall**	**F$_1$**	**Precision**	**Recall**	**F$_1$**
CoAID	LSTM	81.00	91.00	86.00	95.00	78.00	86.00
	mBERT	91.00	84.00	87.00	94.00	87.00	**90.00**
	XLM-R$_{\text{Base}}$	93.00	**87.00**	90.00	87.00	88.00	88.00
	XLM-R$_{\text{Large}}$	**98.00**	86.00	**92.00**	**97.00**	**93.00**	**90.00**
ReCOV	LSTM	60.00	56.00	58.00	54.00	57.00	55.00
	mBERT	81.00	**59.00**	**68.00**	87.00	**55.00**	**68.00**
	XLM-R$_{\text{Base}}$	72.00	58.00	64.00	75.00	**55.00**	64.00
	XLM-R$_{\text{Large}}$	**89.00**	52.00	66.00	**89.00**	51.00	65.00
CoAID+ReCOV	LSTM	79.00	58.00	67.00	66.00	70.00	68.00
	mBERT	**94.00**	89.00	**91.00**	**94.00**	89.00	**92.00**
	XLM-R$_{\text{Base}}$	88.00	88.00	88.00	88.00	88.00	88.00
	XLM-R$_{\text{Large}}$	86.00	**94.00**	90.00	85.00	**93.00**	89.00

Table 9: Performance of our COVID-misinformation *Twitter* models on the fake class only across the 3 settings CoAID, ReCOVery, and CoAID+ReCOVery. LSTM is our baseline.

5 Applications on Mega-COV

Now that we have developed two highly effective models, one for COVID-relevance and another for COVID-misinformation, we can employ these models to make discoveries using `Mega-COV`. Since our misinformation models are focused only on English (due to the external gold data we used for training being English only), we will restrict this analysis to the English language.[11] We were curious whether model predictions will have different distributions on the different types of Twitter posts (i.e., tweets, retweets, and replies). Hence, to enable such comparisons, we extract a random sample of 10M samples from each of these post types (for a total of 30M) from the year 2020 in `Mega-COV`. We then apply the XLM-R$_{\text{Large}}$ *relevance* and *misinformation* models on the extracted samples. Table 10 shows the distribution of predicted labels from each of the two models across the 3 posting types (tweets, retweets, and replies). Strikingly, as the top half of the table shows, while only 7.77% of tweets are predicted as *COVID-related*, almost all retweets (99.84%) are predicted as *related*. This shows that users' retweets were focused almost exclusively on COVID-19. The table (bottom half) also shows that retweets are highest carriers of content predicted as *fake* (3.67%), followed by tweets (2.3%). From the table, we can also deduce that only 2.45% of all English language Twitter content (average across the 3 posting types) are predicted as fake. ***Given the global use of English, and the large volume of English posts Twitter receives daily, this percentage of fake content is still problematically high.***

5.1 Annotation Study

We perform a human annotation study on a small sample of 150 random posts from those the model predicted as both COVID-related *and* fake. Two annotators labeled the 150 samples for two types of

---

[10] Even though we could have used the monolingual versions of the transformer-based language models (i.e., BERT and RoBERTa), we stick to the multilingual versions for consistency.

[11] But we emphasize the multilingual capacity of our COVID-relevance model.

tags, relevance and veracity. For relevance, all the 150 posts were found relevant by the two annotators (perfect agreement). For veracity, since some posts can be very challenging to identify, we asked annotators to assign one of the 3 tags in the set {*true, fake, unknown*}. We did not ask annotators to consult any outside sources (e.g., Wikipedia or independent fact-checking sites) to identify veracity of the samples. Inter-annotator agreement is at *Kappa (K)*=77.81%, thus indicating almost perfect agreement. On average, annotators assigned the *fake* class 39.39% of the time, the *true* class 3.02%, and the *unknown* class 57.05%. While these findings show that it is hard for humans to identify data veracity without resorting to external sources, it also demonstrates the utility of the model in detecting actual fake stories in the wild. We provide a number of samples from the posts that were automatically tagged as *COVID-related* and either *true* or *false* by our misinformation/veracity model in Table 11.

Model	Data	Prediction	Percentage
COVID Relevance	Tweets	Related	7.77
		Unrelated	92.23
	Retweets	Related	99.84
		Unrelated	0.16
	Replies	Related	12.94
		Unrelated	87.06
COVID Misinfo.	Tweets	Fake	2.3
		True	97.10
	Retweets	Fake	1.38
		True	98.33
	Replies	Fake	3.67
		True	96.62

Table 10: Distribution of predicted labels from our COVID-relevance and COVID-misinformation models on randomly selected 30M English samples from `Mega-COV` data.

6 Data Release and Ethics

Data Distribution. The size of the data makes it an attractive object of study. Collection and exploration of the data required significant computing infrastructure and use of powerful data streaming and processing tools. To facilitate use of the dataset, we organize the tweet IDs we release by time (month and year) and language. This should enable interested researchers to work with the exact parts of the data related to their research questions even if they do not have large computing infrastructure.

Ethical Considerations. We collect `Mega-COV` from the public domain (Twitter). In compliance with Twitter policy, we do not publish hydrated tweet content. Rather, we only publish publicly available tweet IDs. All Twitter policies, including respect and protection of user privacy, apply. We decided not to assign geographic region tags to the tweet IDs we distribute, but these already exist on the `json` object retrievable from Twitter. Still, location information should be used with caution. Twitter does not allow deriving or inferring, or storing derived or inferred, potentially sensitive characteristics about users. Sensitive user attributes identified by Twitter include health (e.g., pregnancy), negative financial status or condition, political affiliation or beliefs, religious or philosophical affiliation or beliefs, sex life or sexual orientation, trade union membership, and alleged or actual commission of a crime. If they decide to use `Mega-COV`, we expect researchers to review Twitter policy[12] and applicable laws, including the European Union's General Data Protection Regulation (GDPR)[13], beforehand. ***We encourage use of*** **`Mega-COV`** ***for social good, including applications that can improve health and well-being and enhance online safety.***

7 Related Works

Twitter Datasets for COVID-19. Several works have focused on creating datasets for enabling COVID-19 research. To the best of our knowledge, all these works depend on a list of hashtags related to COVID-19 and focus on a given period of time. For example, Chen et al. (2020) started collecting tweets on Jan. 22^{nd} and continued updating by actively tracking a list of 22 popular keywords such as *#Coronavirus, #Corona*, and *#Wuhancoronavirus*. As of May 30, 2020 (Chen et al., 2020) report 144M tweets. Singh et al. (2020) collect a dataset covering January 16 2020-March 15 2020 using a list of hashtags such as *#2019nCoV, #ChinaPneumonia* and *#ChinesePneumonia*, for a total of 2.8M tweets, $\sim$ 18M re-tweets, and $\sim$ 457K direct conversations. Using location information on the data, authors report that tweets strongly correlated with newly identified cases in these locations. Similarly, Alqurashi et al. (2020) use a list of keywords and hashtags related to Covid-19 with Twit-

---

[12] `https://developer.twitter.com/en/developer-terms/policy`

[13] `https://gdpr-info.eu`.

Post	Prediction
Vatican confirms Pope Francis and two aides test positive for Coronavirus - MCM Whoaa 😂 URL	Fake
😭∼CDC recommends men shave their beards to protect against coronavirus – USER URL	Fake
COVID - 19 : Chinese health authorities confirm patient zero ’ had sex with bats ’ URL	Fake
Royal Palace confirms Queen Elizabeth tests positive for coronavirus URL	Fake
Is COVID - 19 airborne contagious ? New study shows that coronavirus may be caught from the air * 3 - hours * after it has been exposed .	True
A close relative of SARS-CoV - 2 found in bats offers more evidence it evolved naturally URL	True
Antiviral remdesivir prevents disease progression in monkeys with COVID - 19 — National Institutes of Health (NIH) URL	True
COVID Surges Among Young Adults URL	True

Table 11: Sample `Mega-COV` posts predicted as *COVID-related*, and either *true* or *fake* by our models.

ter's streaming API to collect a dataset of Arabic tweets. The dataset covers the period of March 1 2020-March 30 2020 and is at 4M tweets. The authors' goal is to help researchers and policy makers study the various societal issues prevailing due to the pandemic. In the same vein, Lopez et al. (2020) collect a dataset of ∼ 6.5M in multiple languages, with English accounting for ∼ 63.4% of the data. The dataset covers January 22 2020-March 2020. Analyzing the data, authors observe the level of retweets to rise abruptly as the crisis ramped up in Europe in late February and early March.

Twitter in emergency and crisis. Social media can play a useful role in disaster and emergency since they provide a mechanism for wide information dissemination (Simon et al., 2015). Examples include use of Twitter information for the Typhoon Haiyan in the Philippines (Takahashi et al., 2015), Tsunami in Padang Indonesia (Carley et al., 2016), the Nepal 2015 earthquakes (Verma et al., 2019), Harvey Hurricane (Marx et al., 2020). A number of works have focused on developing systems for emergency response. An example is McCreadie et al. (2019). Other works focused on developing systems for detecting misuse of social media (Alshehri et al., 2018, 2020; Nagoudi et al., 2020; Elmadany et al., 2020).

Misinformation About COVID-19. Misinformation can spread fast during disaster. Social data have been used to study rumors and various types of fake information related to the Zika (Ghenai and Mejova, 2017) and Ebola (Kalyanam et al., 2015) viruses. In the context of COVID-19, a number of works have focused on investigating the effect of misinformation on mental health (Rosenberg et al., 2020), the types, sources, claims, and responses of a number of pieces of misinformation about COVID-19 (Brennen et al., 2020), the propagation pattern of rumours about COVID-19 on Twitter and Weibo (Do et al., 2019), check-worthiness (Wright and Augenstein, 2020), modeling the spread of misinformation and related networks about the pandemic (Cinelli et al., 2020; Osho et al., 2020; Pierri et al., 2020; Koubaa, 2020), estimating the rate of misinformation in COVID-19 associated tweets (Kouzy et al., 2020), the use of bots (Ferrara, 2020), predicting whether a user is COVID19 positive or negative (Karisani and Karisani, 2020), and the quality of shared links Singh et al. (2020). Other works have focused on detecting racism and hate speech (Devakumar et al., 2020; Schild et al., 2020; Shimizu, 2020; Lyu et al., 2020) and emotional response (Kleinberg et al., 2020).

8 Conclusion

We presented `Mega-COV`, a billion-scale dataset of 104 languages for studying COVID-19 pandemic. In addition to being large and highly multilingual, our dataset comprises data pre-dating the pandemic. This allows for comparative and longitudinal investigations. We provided a global description of `Mega-COV` in terms of its geographic and temporal coverage, over-viewed its linguistic diversity, and provided analysis of its content based on hashtags and top domains. We also provided a case study of how the data can be used to track global human mobility. The scale of the Mega-COV has also allowed us to make a number of striking discoveries, including **(1)** the shift toward retweeting and replying to other users rather than tweeting in 2020 and **(2)** the role of international news sites as key sources of information during the pandemic. In addition, we developed effective models for detecting COVID relevance and COVID misinformation and applied them to a large sample of our dataset. Our dataset and models are publicly available.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. EmoNet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 718–728.

Sarah Alqurashi, Ahmad Alhindi, and Eisa Alanazi. 2020. Large arabic twitter dataset on covid-19. *arXiv preprint arXiv:2004.04315*.

Ali Alshehri, El Moatez Billah Nagoudi, and Muhammad Abdul-Mageed. 2020. Understanding and detecting dangerous speech in social media. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 40–47, Marseille, France. European Language Resource Association.

Ali Alshehri, El Moatez Billah Nagoudi, Alhuzali Hassan, and Muhammad Abdul-Mageed. 2018. Think before your click: Data and models for adult content in Arabic twitter. In *The 2nd Text Analytics for Cybersecurity and Online Safety (TA-COS-2018), LREC*.

J Scott Brennen, Felix M Simon, Philip N Howard, and Rasmus Kleis Nielsen. 2020. Types, sources, and claims of covid-19 misinformation. *Reuters Institute*.

Kathleen M Carley, Momin Malik, Peter M Landwehr, Jürgen Pfeffer, and Michael Kowalchuck. 2016. Crowd sourcing disaster management: The complex nature of twitter usage in padang indonesia. *Safety science*, 90:48–61.

Emily Chen, Kristina Lerman, and Emilio Ferrara. 2020. Covid-19: The first public coronavirus twitter dataset. *arXiv preprint arXiv:2003.07372*.

Matteo Cinelli, Walter Quattrociocchi, Alessandro Galeazzi, Carlo Michele Valensise, Emanuele Brugnoli, Ana Lucia Schmidt, Paola Zola, Fabiana Zollo, and Antonio Scala. 2020. The covid-19 social media infodemic. *arXiv preprint arXiv:2003.05004*.

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

Limeng Cui and Dongwon Lee. 2020. Coaid: Covid-19 healthcare misinformation dataset. *arXiv preprint arXiv:2006.00885*.

Delan Devakumar, Geordan Shannon, Sunil S Bhopal, and Ibrahim Abubakar. 2020. Racism and discrimination in covid-19 responses. *Lancet (London, England)*, 395(10231):1194.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tien Huu Do, Xiao Luo, Duc Minh Nguyen, and Nikos Deligiannis. 2019. Rumour detection via news propagation dynamics and user representation learning. *arXiv preprint arXiv:1905.03042*.

Tien Huu Do, Duc Minh Nguyen, Evaggelia Tsiligianni, Bruno Cornelis, and Nikos Deligiannis. 2018. Twitter user geolocation using deep multiview learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6304–6308. IEEE.

AbdelRahim Elmadany, Chiyu Zhang, Muhammad Abdul-Mageed, and Azadeh Hashemi. 2020. Leveraging affective bidirectional transformers for offensive language detection. In *The 4th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT4), LREC*, pages 102–108.

Emilio Ferrara. 2020. # covid-19 on twitter: Bots, conspiracies, and social media activism. *arXiv preprint arXiv:2004.09531*.

Amira Ghenai and Yelena Mejova. 2017. Catching zika fever: Application of crowdsourcing and machine learning for tracking health misinformation on twitter. *arXiv preprint arXiv:1707.03778*.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Mark Graham, Scott A Hale, and Devin Gaffney. 2014. Where in the world are you? geolocation and language identification in twitter. *The Professional Geographer*, 66(4):568–578.

Bo Han, Afshin Rahimi, Leon Derczynski, and Timothy Baldwin. 2016. Twitter geolocation prediction shared task of the 2016 workshop on noisy user-generated text. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 213–217.

Janani Kalyanam, Sumithra Velupillai, Son Doan, Mike Conway, and Gert Lanckriet. 2015. Facts and fabrications about ebola: A twitter based study. *arXiv preprint arXiv:1508.02079*.

Negin Karisani and Payam Karisani. 2020. Mining coronavirus (covid-19) posts in social media. *arXiv preprint arXiv:2004.06778*.

Bennett Kleinberg, Isabelle van der Vegt, and Maximilian Mozes. 2020. Measuring emotions in the covid-19 real world worry dataset. *arXiv preprint arXiv:2004.04225*.

Anis Koubaa. 2020. Understanding the covid19 outbreak: A comparative data analytics and study. *arXiv preprint arXiv:2003.14150*.

Ramez Kouzy, Joseph Abi Jaoude, Afif Kraitem, Molly B El Alam, Basil Karam, Elio Adib, Jabra Zarka, Cindy Traboulsi, Elie W Akl, and Khalil Baddour. 2020. Coronavirus goes viral: Quantifying the covid-19 misinformation epidemic on twitter. *Cureus*, 12(3).

Christian E Lopez, Malolan Vasu, and Caleb Gallemore. 2020. Understanding the perception of covid-19 policies by mining a multilanguage twitter dataset. *arXiv preprint arXiv:2003.10359*.

Marco Lui and Timothy Baldwin. 2012. langid. py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 system demonstrations*, pages 25–30. Association for Computational Linguistics.

Marco Lui and Timothy Baldwin. 2014. Accurate language identification of twitter messages. In *Proceedings of the 5th workshop on language analysis for social media (LASM)*, pages 17–25.

Hanjia Lyu, Long Chen, Yu Wang, and Jiebo Luo. 2020. Sense and sensibility: Characterizing social media users regarding the use of controversial terms for covid-19. *arXiv preprint arXiv:2004.06307*.

Julian Marx, Milad Mirbabaie, and Christian Ehnis. 2020. Sense-giving strategies of media organisations in social media disaster communication: Findings from hurricane harvey. *arXiv preprint arXiv:2004.08567*.

Richard McCreadie, Cody Buntain, and Ian Soboroff. 2019. Trec incident streams: Finding actionable information on social media. *Proceedings of the 16th International Conferenc e on Information Systems for Crisis Response and Management, Valencia, Spain*.

Saif M Mohammad and Svetlana Kiritchenko. 2015. Using hashtags to capture fine emotion categories from tweets. *Computational Intelligence*, 31(2):301–326.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, Muhammad Abdul-Mageed, Tariq Alhindi, and Hasan Cavusoglu. 2020. Machine Generation and Detection of Arabic Manipulated and Fake News. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 69–84.

J Ooms and D Sites. 2018. cld2: Google's Compact Language Detector 2. *Retrieved Feburary*, 7:2019.

Abiola Osho, Caden Waters, and George Amariucai. 2020. An information diffusion approach to rumor propagation and identification on twitter. *arXiv preprint arXiv:2002.11104*.

Francesco Pierri, Carlo Piccardi, and Stefano Ceri. 2020. Topology comparison of twitter diffusion networks effectively reveals misleading information. *Scientific reports*, 10(1):1–9.

Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1500–1510. Association for Computational Linguistics.

Hans Rosenberg, Shahbaz Syed, and Salim Rezaie. 2020. The twitter pandemic: The critical role of twitter in the dissemination of medical information and misinformation during the covid-19 pandemic. *Canadian Journal of Emergency Medicine*, pages 1–7.

Leonard Schild, Chen Ling, Jeremy Blackburn, Gianluca Stringhini, Yang Zhang, and Savvas Zannettou. 2020. " go eat a bat, chang!": An early look on the emergence of sinophobic behavior on web communities in the face of covid-19. *arXiv preprint arXiv:2004.04046*.

Kazuki Shimizu. 2020. 2019-ncov, fake news, and racism. *The Lancet*, 395(10225):685–686.

Tomer Simon, Avishay Goldberg, and Bruria Adini. 2015. Socializing in emergencies—a review of the use of social media in emergency situations. *International Journal of Information Management*, 35(5):609–619.

Lisa Singh, Shweta Bansal, Leticia Bode, Ceren Budak, Guangqing Chi, Kornraphop Kawintiranon, Colton Padden, Rebecca Vanarsdall, Emily Vraga, and Yanchen Wang. 2020. A first look at covid-19 information and misinformation sharing on twitter. *arXiv preprint arXiv:2003.13907*.

Bruno Takahashi, Edson C Tandoc Jr, and Christine Carmichael. 2015. Communicating on twitter during a disaster: An analysis of tweets during typhoon haiyan in the philippines. *Computers in Human Behavior*, 50:392–398.

Rakesh Verma, Samaneh Karimi, Daniel Lee, Omprakash Gnawali, and Azadeh Shakery. 2019. Newswire versus social media for disaster response and recovery. *arXiv preprint arXiv:1906.10607*.

WHO. 2020. Who statement regarding cluster of pneumonia cases in wuhan, china. *Beijing: WHO*, 9.

Dustin Wright and Isabelle Augenstein. 2020. Fact check-worthiness detection as positive unlabelled learning. *arXiv preprint arXiv:2003.02736*.

Xinyi Zhou, Apurva Mulay, Emilio Ferrara, and Reza Zafarani. 2020. Recovery: A multimodal repository for covid-19 news credibility research. *arXiv preprint arXiv:2006.05557*.

Appendices

A Data Distribution

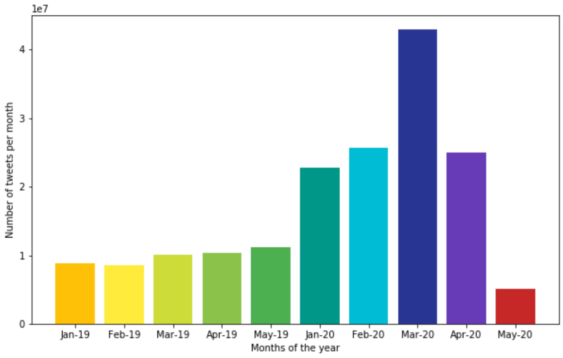


Figure A.1: Frequency of tweeting during Jan-May (10^{th}) 2020 vs. Jan-May 2019.

B Hashtag Content Analysis

Hashtags usually correlate with the topics users post about. We provide the top 30 hashtags in the data in Table B.1. As the table shows, users tweet heavily about the pandemic using hashtags such as *COVID19, coronavirus, Coronavirus, COVID19, Covid19, covid19* and *StayAtHome*. Simple word clouds of hashtags from the various languages (Figure B.1 provides clouds from the top 10 languages) also show COVID-19 topics trending. We also observe hashtags related to **gaming** (e.g., *NowPlaydo, PSshare*, and *NintendoSwitch*). This reflects how users may be spending part of their newly-found home time. We also note frequent occurrence of **political** hashtags in languages such Arabic, Farsi, Indian, and Urdu. This is in contrast to discussions in European languages where politics are not as visible. For example, in Urdu, discussions involving the army and border issues show up. This may be partly due to different political environments, but also due to certain European countries such as Italy, Sweden, Spain, and the U.K. being hit harder (and earlier) than many countries in the Middle East and Asia. In Indian languages such as Tamil and Hindi, posts also focused on movies (e.g., *Valimai*), TV shows (e.g., *Big Boss*), doctors, and even fake news along with the pandemic-related hashtags.

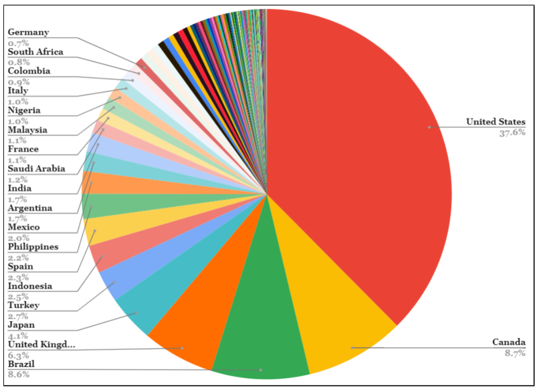


Figure A.2: Geographical diversity in `Mega-COV`. We show the distribution of our *geo-located* data over the top 20 countries with most tweets and responses. Overall, 268 countries are represented in the data.

2019		**2020**	
Hashtag	**Freq**	**Hashtag**	**Freq**
NewProfilePic	64,922	COVID19	260,024
love	41,964	coronavirus	219,615
Repost	39,128	NewProfilePic	102,724
art	35,825	BBB20	91,775
music	28,335	Covid-19	70,106
travel	28,236	COVID-19	67,737
GameofThrones	21,484	Coronavirus	53,251
nature	18,563	covid19	47,940
instagood	18,491	StayHome	44,165
photooftheday	18,032	NintendoSwitch	42,812
tbt	17,332	love	42,497
realestate	17,255	bbb20	39,974
shopmycloset	16,760	NowPlaying	39,069
GameOfThrones	16,127	Repost	37,036
peing	15,930	AnimalCrossing	36,369
fitness	15,623	ACNH	35,528
food	15,358	photography	35,209
BellLetsTalk	14,853	COVID2019	33,512
NowPlaying	14,849	shopmycloset	31,428
family	14,060	music	30,537
style	14,041	StayAtHome	30,313
SoundCloud	13,904	QuedateEnCasa	30,194
WeTheNorth	13,579	stayhome	27,540
GOT	13,458	PS4share	27,487
np	13,335	SocialDistancing	27,376
MyTwitterAnniv.	12,965	lockdown	27,344
Toronto	12,964	TikTok	27,287

Table B.1: Top 30 hashtags in `Mega-COV` for 2019 vs. 2020.

An interesting observation from the **Chinese language** word cloud is the use of hashtags such as *ChinaPneumonia* and *WuhanPneumonia* to refer to the pandemic. We did not observe these same hash-

(a) English (en) (b) Spanish (es) (c) Portugese (pt) (d) Japanese (ja) (e) Arabic (ar)

(f) Indonesian (in) (g) Turkish (tr) (h) French (fr) (i) Tagalog (tl) (j) Italian (it)

Figure B.1: Word clouds for hashtags in tweets from the top 10 languages in the data. We note that tweets in non-English can still carry English hashtags or employ Latin script.

tags in any of the other languages. Additionally, for some reason, **Apple** seems to be trending during the first 4 months of 2020 in China owing to hashtags such as *appledaily* and *appledailytw*. Some languages, such as Romanian and Vietnamese, involve discussions of **bitcoin** and crypto-currency. This was also seen in the Chinese language word cloud, but not as prominently.

C Domain Sharing Analysis

Domains in URLs shared by users also provide a window on what is share-worthy. We perform an analysis of the top 200 domains shared in each of 2019 and 2020. The major observation we reach is the *surge in tweets involving news websites, and the rise in ranks for the majority of these websites compared to 2019*. Table C.1 shows the top 40 news domains in the 2020 data and their change in rank compared to 2019. Such a heavy sharing of news domains reflects users' needs: Intuitively, at times of global disruption, people need more frequent updates on ongoing events. Of particular importance, especially relative to other ongoing political polarization in the U.S., is the striking rise of the conservative news network *Fox News*, which has moved from a rank of 118 in 2019 to 67 in 2020 with a swooping 51 positions jump. We also note the rank of some news sites (e.g., *The Globe and Mail* and *The Star* going down. This is perhaps due to people resorting to international (and more diverse) sources of information to remain informed about countries other than their own.

Other domains: Other noteworthy domain activities include those related to gaming, video and music, and social media tools. Ranks of these domains have not necessarily shifted higher than 2019 but remain prominent. This shows these themes still being relevant in 2020. In spite of the economic impact of the pandemic, shopping domains such as *etsy.me* and *poshmark.com* have markedly risen in rank as people moved to shopping online in more significant ways. We now introduce a case study as to how our data can be used for mobility tracking.

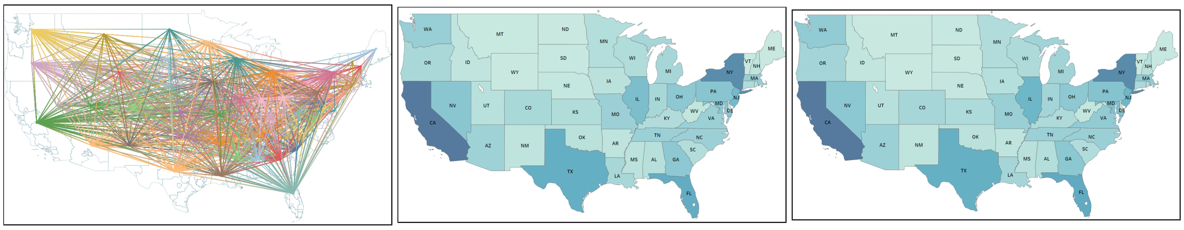
Domain	Rank	Domain	Rank
theguardian.com	↑ 3	thehill.com	↑ 51
nytimes.com	↑ 10	globeandmail.com	↓ -38
cnn.com	↑ 18	businessinsdr.com	↑ 31
apple.news	↑ 4	theatlantic.com	↑ 27
washingtonpost.com	↑ 16	newsbreakapp.com	↑ 472
cbc.ca	↓ -13	eldiario.es	↑ 62
bbc.co.uk	↓ -4	apnews.com	↑ 48
bbc.com	↑ 3	abc.es	↑ 89
nyti.ms	↓ -11	reuters.com	↑ 59
foxnews.com	↑ 51	thestar.com	↓ -64
forbes.com	↓ -14	francebleu.fr	↑ 424
nbcnews.com	↑ 39	globalnews.ca	↓ -78
wsj.com	↑ 11	independent.co.uk	↓ -10
bloomberg.com	↑ 13	elmundo.es	↑ 21
ctvnews.ca	↑ 2	indiatimes.com	↕ 0
nypost.com	↑ 100	radio-canada.ca	↓ -66
cnbc.com	↑ 43	lavanguardia.com	↑ 96
usatoday.com	↑ 6	dailymail.co.uk	↑ 23
latimes.com	↑ 23	politico.com	↑ 403
huffpost.com	↑ 66	sky.com	↑ 114

Table C.1: Top 40 domains in 2020 data and their rank change relative to their rank in 2019.

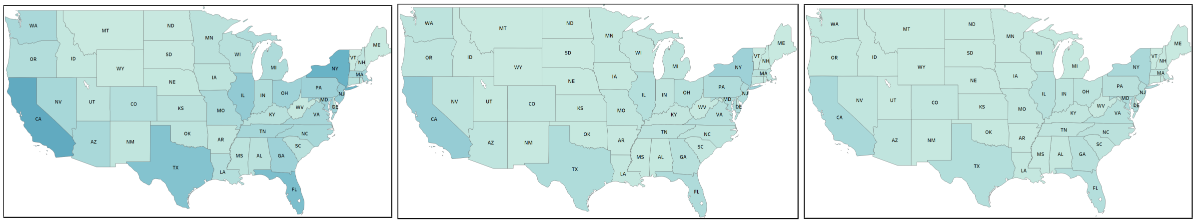
D Case Study: Mapping Human Mobility with Mega-COV

Geolocation information in `Mega-COV` can be used to characterize and track human mobility in various ways. We investigate some of these next.

Inter-Region Mobility. `Mega-COV` can be exploited to generate *responsive maps* where end users can check mobility patterns between different regions over time. In particular, geolocation information can show mobility patterns between regions. As an illustration of this use case, albeit



(a) Overall inter-state mobility (b) January (c) February



(d) March (e) April (f) May

Figure D.1: Inter-state user mobility in the U.S. for Jan-May, 2020.

in a static form, we provide Figure D.1a where we show how users move between U.S. states. We can also exploit `Mega-COV` to show inter-state mobility during a given window of time.[14] Figures D.1b- D.1e present user mobility between U.S. states. The figure shows a clear change from higher mobility in January and February to much less activity in March, April, and May. Clear differences can be seen in key states where the pandemic has hit hard such as New York (NY), California (CA), and Washington State (WA). We provide visualizations of mobility patterns for a number of countries where the pandemic has hit (sometimes hard), as follows: Brazil, Canada, Italy, Saudi Arabia, and the United Kingdom.

Intra-Region Mobility. We also use information in `Mega-COV` to map each user to a single home region (i.e., city, state/province, and country). We follow Geolocation literature (Roller et al., 2012; Graham et al., 2014; Han et al., 2016; Do et al., 2018) in setting a condition that a user must have posted at least 10 tweets from a given region. However, we also condition that at least 60% of all user tweets must have been posted from the same region. We use the resulting set of users whose home location we can verify to map user weekly mobility *within* their own city, state, and country exclusively for both Canada and the U.S. as illustrating examples. We provide the related visualization in supplementary material under "**User Weekly *Intra*-Region Mobility**".

D.1 User Weekly Intra-Region Mobility

We can also visualize user mobility as a distance from an average mobility score on a weekly basis. Namely, we calculate an *average weekly mobility score* for the year 2019 using geo-tag information (longitude and latitude) and use it as a *baseline* against which we plot user mobility for each week of 2019 and 2020 up until April. In general, we observe a drop in user mobility in Canada starting from mid-March. For U.S. users, we notice a very high mobility surge starting around end of February and early March, only waning down the last week of March and continuing in April as shown in Figure D.8. For both the U.S. and Canada, we hypothesize the surge in early March (much more noticeable in the U.S.) is a result of people moving back to their hometowns, returning from travels, moving for basic needs stocking, etc.

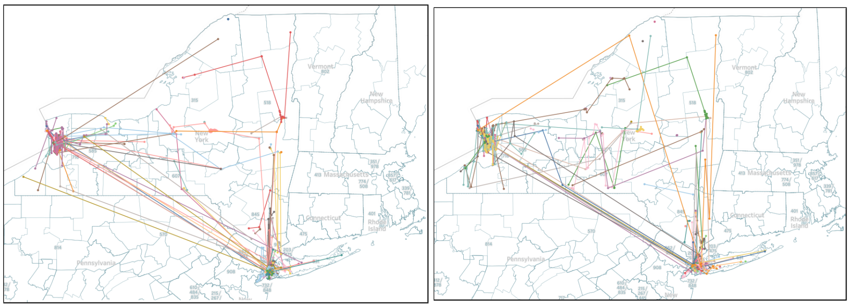
E COVID-Relevance Model

E.1 Dataset

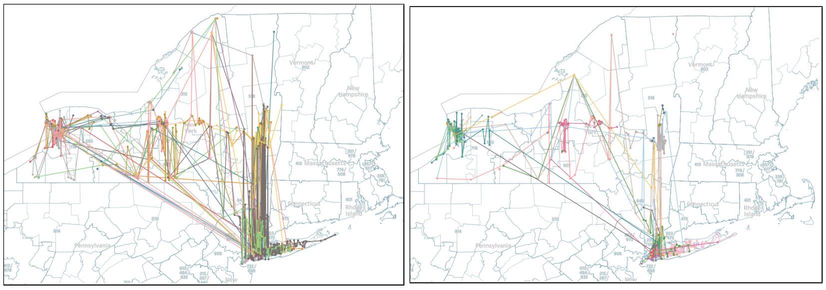
We randomly sample 200K tweets from the English data in Chen et al. (2020) and a maximum of 100K from each of the rest of languages. For languages where there is $<$ 100K tweets, we take all data. For the negative class, we extract data from Jan-Nov, 2019 from `Mega-COV`. For each language, we take roughly the same number of tweets we sampled for the positive class. Table E.1 shows the distribution of the positive class data from Chen et al. (2020).

---

[14] Here, due to increased posting in 2020, we normalize the number of visits between states by the total number of all tweets posted during a given month.



(a) January (3,949 users)　(b) February (4,500 users)



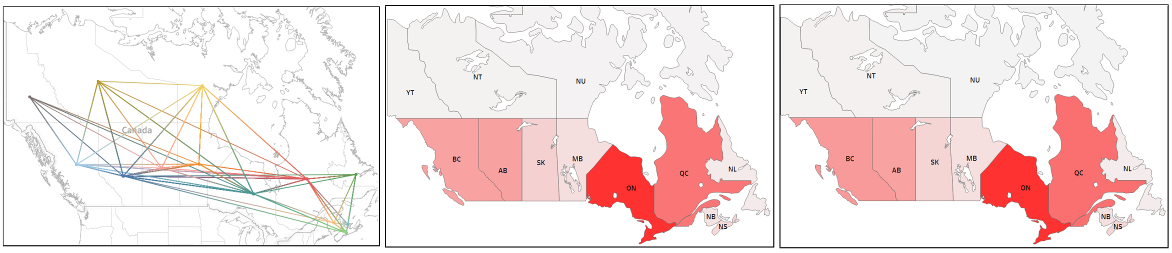
(c) March (5,145 users)　(d) April (1,870 users)

Figure D.2: User monthly mobility within New York State.

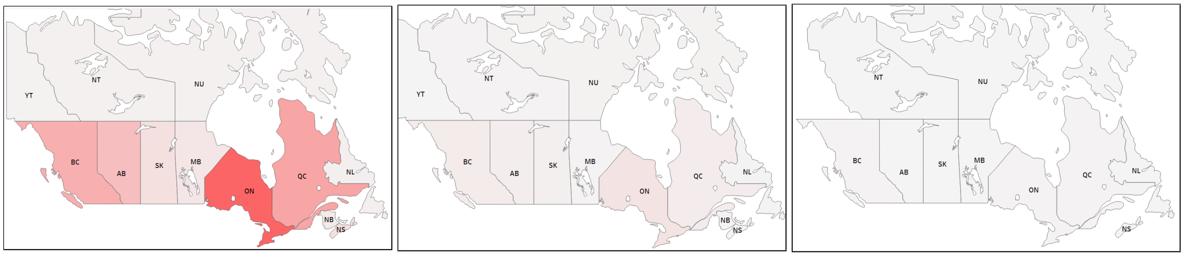
(a) Inter-States　(b) January　(c) February

(d) March　(e) April　(f) May

Figure D.3: User mobility between Brazil states (estados) during Jan-May 2020.

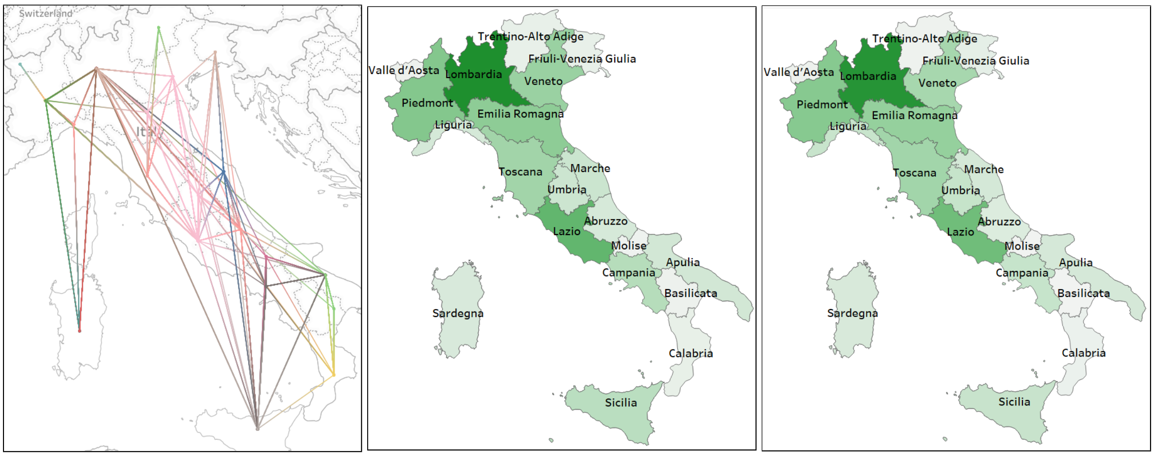


(a) Inter-provinces (b) January (c) February

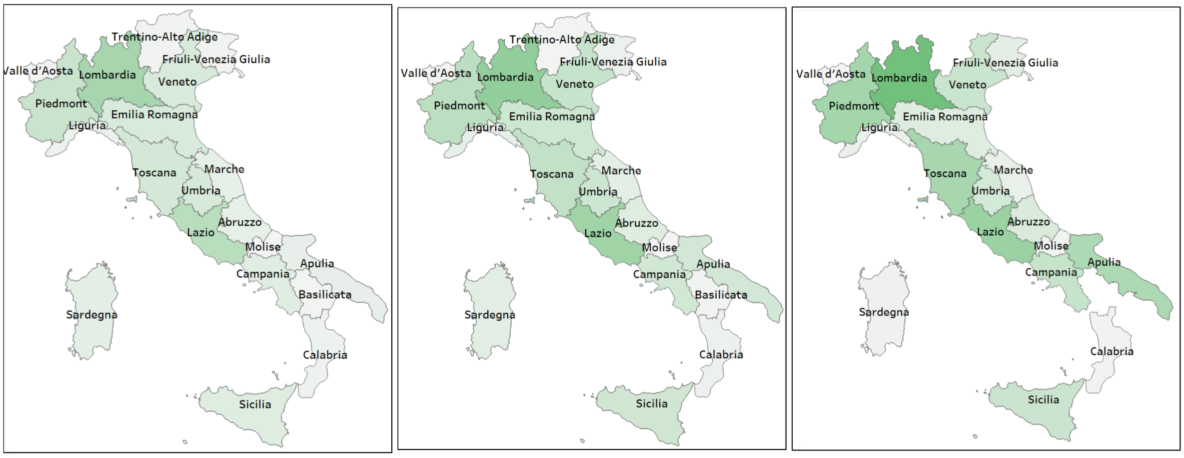


(d) March (e) April (f) May

Figure D.4: User mobility between Canada Provinces during Jan-May 2020



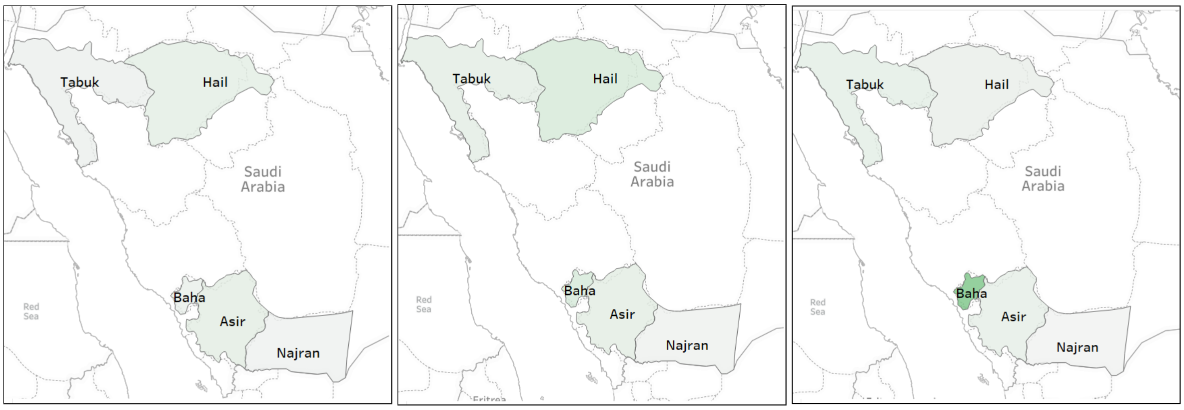
(a) Inter-Regions (b) January (c) February



(d) March (e) April (f) May

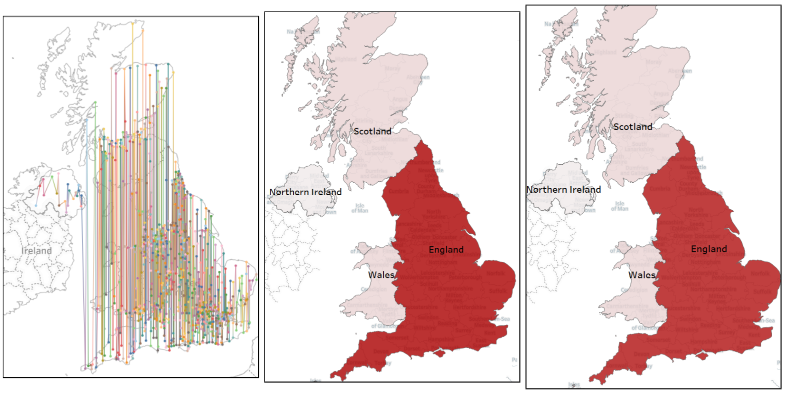
Figure D.5: User mobility between Italy regions (regioni) during Jan-May 2020.

(a) Inter-Regions mobility (b) January (c) February

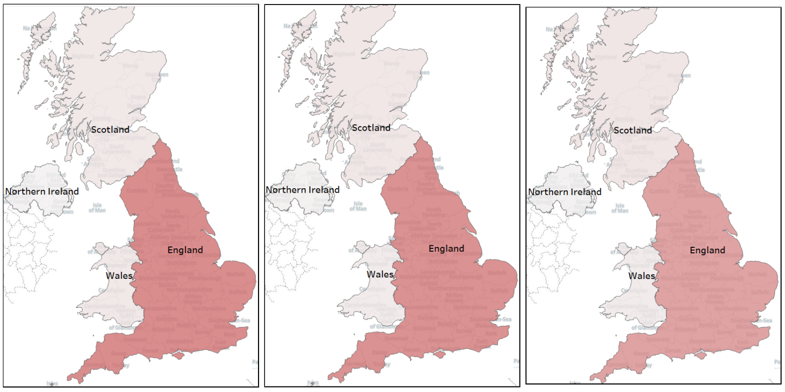


(d) March (e) April (f) May

Figure D.6: User mobility between Saudi Arabia regions during Jan-May 2020.



(a) Inter-cities (b) January (c) February

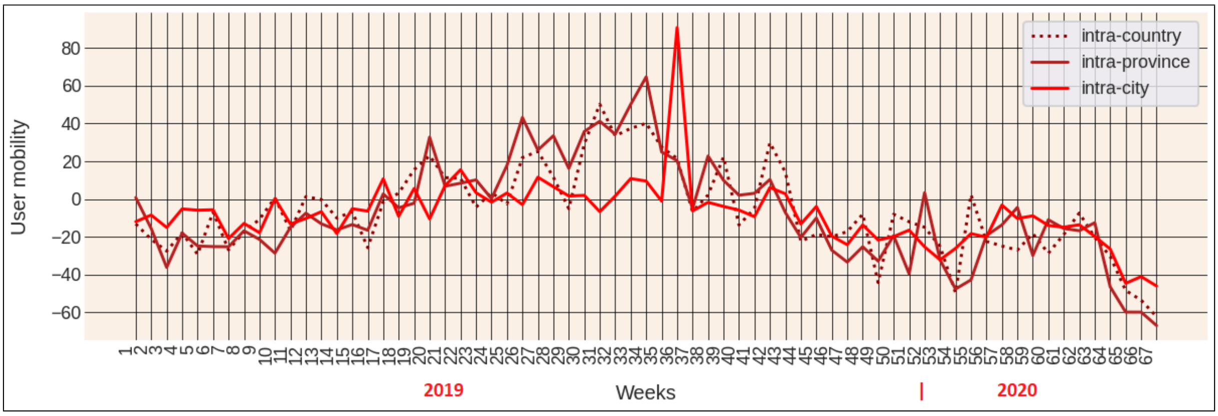


(d) March (e) April (f) May

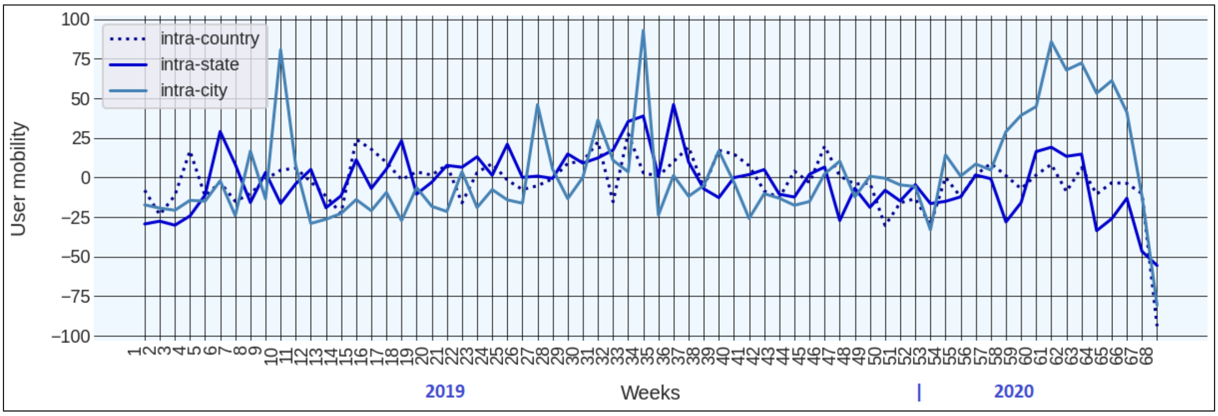
Figure D.7: User mobility between United Kingdom counties during Jan-May 2020.

Replies		RT		Tweets	
Lang	**Frequency**	**Lang**	**Frequency**	**Lang**	**Frequency**
Hebrew (he)	337,237	Hebrew (he)	322,351	Hebrew (he)	504,327
Croatian (hr)	242,772	Croatian (hr)	198,764	Croatian (hr)	243,601
Maltese (mt)	94,695	Maltese (mt)	85,054	Maltese (mt)	145,395
Dzongkha (dz)	64,063	Slovak(sk)	77,846	Slovak(sk)	131,544
Bahasa Indonesia (id)	46,463	Latin (la)	66,061	Latin (la)	104,488
Bosnian (bs)	43,066	Bahasa Indonesia (id)	61,295	Bahasa Indonesia (id)	100,561
Slovak(sk)	36,662	Bosnian (bs)	45,276	Bosnian (bs)	54,200
Swahili (sw)	21,803	Swahili (sw)	28,122	Dzongkha (dz)	46,950
Azerbaijani (az)	20,242	Dzongkha (dz)	26,853	Swahili (sw)	42,076
Latin (la)	13,030	Quechua (qu)	22,559	Malay (ms)	32,967
Albanian (sq)	12,878	Malay (ms)	19,511	Quechua (qu)	31,175
Xhosa (xh)	11,936	Esperanto (eo)	19,397	Albanian (sq)	30,361
Irish (ga)	8,607	Kinyarwanda (rw)	19,371	Kinyarwanda (rw)	30,080
Malagasy (mg)	7,727	Azerbaijani (az)	19,182	Azerbaijani (az)	29,507
Quechua (qu)	7,449	Javanese (jv)	18,180	Javanese (jv)	29,121
Kinyarwanda (rw)	7,427	Albanian (sq)	17,904	Esperanto (eo)	29,019
Esperanto (eo)	6,755	Xhosa (xh)	14,886	Kurdish (ku)	24,259
Malay (ms)	6,683	Irish (ga)	14,807	Afrikaans (af)	22,871
Assamese (as)	6,442	Kurdish (ku)	14,475	Volapük (vo)	21,840
Volapük (vo)	6,245	Galician (gl)	13,337	Irish (ga)	21,151

Table D.1: Top 20 languages detected by langid in `Mega-COV V0.1` which were not detected by twitter, broken by tweets, retweets, and replies.



(a) Canada users



(b) U.S. users

Figure D.8: Canadian and American user *weekly* mobility during 2019-2020. Each point (a week) is modeled as a mobility distance from weekly average mobility in 2019.

lang	**#tweets**	**lang**	**#tweets**	**lang**	**#tweets**	**lang**	**#tweets**
en	200K	ar	76K	uk	6.6K	sr	838
es	100K	ru	50K	no	5.5K	bg	739
th	100K	lt	44.7K	eu	4.8K	dv	634
fr	100K	pl	40.6K	cy	4.3K	pa	450
in	100K	fa	32.6K	ne	4K	my	277
ja	100K	ro	32.5K	lv	3.2K	ps	244
pt	100K	sv	24.2K	mr	3K	am	229
it	100K	fi	24K	iw	2.6K	ckb	190
und	100K	vi	22.7K	ml	2.4K	sd	144
tr	100K	et	21.3K	hu	2.2K	km	128
tl	100K	ur	20.3K	te	2.1K	lo	47
de	100K	ht	16.2K	gu	1.8K	hy	34
zh	100K	da	16.2K	bn	1.5K	ka	23
ca	100K	sl	13.5K	kn	1.4K	bo	15
nl	100K	cs	13.3K	or	1.2K	ug	5
ko	100K	ta	13.1K	is	1.2K		
hi	97.4K	el	10.7K	si	1.2K		

Table E.1: Distribution of language in the COVID-Relevance training data for the positive (i.e., *related*) classe.

F COVID-Misinformation Detection

Cui and Lee (2020) present a **C**ovid-19 he**A**thcare m**I**sinformation **D**ataset (CoAID), with diverse COVID-19 healthcare misinformation, including fake news on websites and social platforms, along with related user engagement (i.e., tweets and replies) about such news. CoAID includes $3,235$ news articles and claims, $294,692$ user engagement, and 851 social platform posts about COVID-19. The topics of CoAID include: $\{$*COVID-19, coronavirus, pneumonia, flu9, lock down, stay home, quarantine and ventilator*$\}$. The dataset is collected from December 1, 2019 to July 1, 2020 and is organized as follows:

- **News Articles.** To collect the *true* news (not fake), 9 reliable media outlets were identified. These include World Health Organization[15] and the U.S. National Institute of Health[16], for example. To collect *fake* news, 6 fact-checking websites were used (e.g. LeadStories[17], PolitiFact[18]).

- **Claims.** The true and fake claims (i.e., news with one or two sentences) were collected using: (1) the official WHO website,[19] (2) WHO official Twitter account,[20] and (3) the medical news today website[21].

	News					
	Fake			**True**		
	TRAIN	**DEV**	**TEST**	**TRAIN**	**DEV**	**TEST**
CoAID*	669	84	84	2,172	272	272
ReCOVery	532	66	66	1,091	136	136
	Tweets					
	Fake			**True**		
	TRAIN	**DEV**	**TEST**	**TRAIN**	**DEV**	**TEST**
CoAID	8,072	1,009	1,009	110,076	13,759	13,759
ReCOVery	18,272	2,284	2,284	86,437	10,805	10,805

Table F.1: Statistics of CoAID, ReCOVery, and FakeCovid datasets across the data splits. For **C**oAID*, we merge the claim and news.

- **User Engagement.** Queries based on the true and fake articles and claims were used to build a dataset of user engagement from Twitter where the goal was to acquire the tweets discussing the news in question and related Twitter replies.

Data	**Model**	**DEV**		**TEST**	
		Acc.	**F1**	**Acc.**	**F1**
CoAID	mBERT	98.88	98.45	**97.47**	**96.48**
	XLM-R$_{\text{Base}}$	98.31	97.64	96.35	94.74
	XLM-R$_{\text{Large}}$	**99.16**	**98.84**	96.91	95.66
ReCOVery	mBERT	86.76	84.14	85.64	82.73
	XLM-R$_{\text{Base}}$	85.78	83.56	87.13	85.01
	XLM-R$_{\text{Large}}$	**88.73**	**86.36**	**88.12**	**85.91**
CoAID+ReCOV.	mBERT	**93.39**	**91.41**	92.11	89.66
	XLM-R$_{\text{Base}}$	92.50	89.90	91.04	87.79
	XLM-R$_{\text{Large}}$	93.21	90.86	**92.83**	**90.37**

Table F.2: Results of our fake news detector models on the DEV and TEST splits of CoAID and ReCOVery news articles datasets.

---

[15] `https://www.who.int/`

[16] `https://www.nih.gov/`

[17] `https://leadstories.com/hoax-alert/`

[18] `https://www.politifact.com/coronavirus/`

[19] `https://www.who.int/`

[20] `https://twitter.com/who`

[21] `https://www.medicalnewstoday.com`

Disfluency Correction using Unsupervised and Semi-supervised Learning

Nikhil Saini[*1], **Drumil Trivedi**[*1], **Shreya Khare**[2], **Tejas I. Dhamecha**[2], **Preethi Jyothi**[1],
Samarth Bharadwaj[2], **Pushpak Bhattacharyya**[1]

[1]Indian Institute of Technology Bombay
`{nikhilra,drumilt,pjyothi,pb}@cse.iitb.ac.in`
[2]IBM Research - India
`{skhare34,tidhamecha,samarth.b}@in.ibm.com`

Abstract

Spoken language is different from the written language in its style and structure. Disfluencies that appear in transcriptions from speech recognition systems generally hamper the performance of downstream NLP tasks. Thus, a disfluency correction system that converts disfluent to fluent text is of great value. This paper introduces a disfluency correction model that translates disfluent to fluent text by drawing inspiration from recent encoder-decoder unsupervised style-transfer models for text. We also show considerable benefits in performance when utilizing a small sample of 500 parallel disfluent-fluent sentences in a semi-supervised way. Our unsupervised approach achieves a BLEU score of 79.39 on the Switchboard corpus test set, with further improvement to a BLEU score of 85.28 with semi-supervision. Both are comparable to two competitive fully-supervised models.

1 Introduction

Disfluencies are disruptions to the regular flow of speech, typically occurring in conversational speech. They include filler pauses such as *uh* and *um*, word repetitions, irregular elongations, discourse markers, conjunctions, and restarts. For example, the disfluent sentence "well we're actually uh we're getting ready" has its fluent form as, "we're getting ready". Here, the words highlighted in green, blue and red refer to discourse, filler and restart disfluencies, respectively.

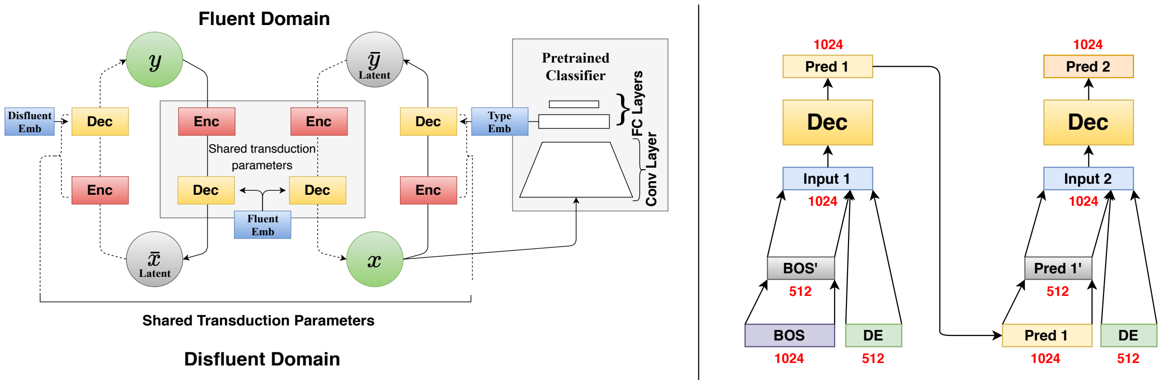
Disfluencies in the text can alter its syntactic and semantic structure, thereby adversely affecting the performance of downstream NLP tasks such as information extraction, summarization, translation, and parsing (Charniak and Johnson, 2001; Johnson and Charniak, 2004). These tasks also employ pretrained language models that are typically trained to expect fluent text. This motivates the need for disfluency correction systems that convert disfluent to fluent text. Prior work has predominantly focused on the problem of disfluency detection (Zayats et al., 2016; Wang et al., 2018; Dong et al., 2019). Inspired by recent work on unsupervised machine translation and style-transfer models for text, we propose an unsupervised encoder-decoder based model to tackle the problem of disfluency correction. Our model does not require access to a parallel corpus of disfluent and fluent sentences. We also show a semi-supervised variant of our model that uses a small amount of parallel disfluent-fluent text and significantly improves performance. To our knowledge, this is the first work to use state-of-the-art unsupervised models for the task of disfluency correction. Our main contributions are as follows:

- We cast the problem of disfluency correction as one of translation from disfluent to fluent text and we propose an unsupervised transformer-based encoder-decoder model for disfluency correction.
- We compare and contrast an unsupervised and semi-supervised approach for disfluency correction, where the latter has access to a small amount of parallel text. We also implement fully-supervised methods as a skyline and show how our models come very close in performance to these approaches, which are very resource-intensive and require large amounts of parallel text.
- We show detailed ablation analyses across disfluency types and present a qualitative study of disfluency corrections that our model can achieve.

---

[*] Joint first authors

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3421–3427
April 19 - 23, 2021.



(a) Style-transfer

(b) Use of domain embedding

Figure 1: Illustration of (a) Style transfer model modified to use type embedding drawn from a pretrained CNN classifier. (b) Conditioning on domain embeddings in the transformers' decoder. Pred(i) and Input(i) are the decoder's prediction and input to the decoder at the i_{th} time-step respectively.

2 Related work

Current literature has primarily focused on disfluency detection in both speech and text in fully supervised settings (Wang et al., 2016; Georgila et al., 2010; Zayats et al., 2014; Tran et al., 2019; Wang et al., 2018; Bach and Huang, 2019; Zayats et al., 2016; Lou and Johnson, 2020a). The grammatical error correction (Omelianchuk et al., 2020) approach does not perform well on the disfluency correction tasks. In most cases, simply removing disfluencies from an utterance can render the sentence ill-formed. More meaningful and syntactically well-formed utterances are generated by performing automatic disfluency removal from speech (Kaushik et al., 2010; Lou and Johnson, 2020b) and text (Wang et al., 2010; Honal and Schultz, 2005; Hassan et al., 2014). With the popularity of end-to-end spoken translation systems, several works translate fluent utterances from disfluent speech (Salesky et al., 2018; Ansari et al., 2020; Fukuda et al., 2020) or disfluent text (Cho et al., 2013; Saini et al., 2020; Cho et al., 2016). Most of these approaches work in a supervised setting or mitigate the lack of parallel disfluent-fluent text via data augmentation, model design, incorporating domain knowledge of the language, or using multi-lingual NMT. (Salesky et al., 2019) proposes a system for conversational speech translation with the joint removal of disfluencies.

3 Our Approach

We draw inspiration from unsupervised neural machine translation models (Lample et al., 2017) and style transfer models (He et al., 2020) to design the disfluency correction model illustrated in Figure 1a. It consists of a single encoder and a single decoder, used to translate in both directions, i.e., from disfluent to fluent text and vice-versa. The decoder is additionally conditioned using a *domain embedding* to convey the direction of translation, signifying whether the input to the encoder is a fluent or disfluent sentence. More details about our framework are described below.

3.1 Unsupervised Disfluency Correction

Figure 1a shows the two directions of translation. The model obtains latent disfluent and latent fluent utterances from the non-parallel fluent and disfluent sentences, respectively, which are further reconstructed back into fluent and disfluent sentences. We employ a backtranslation-based objective, followed by reconstruction for both domains, i.e., disfluent and fluent text. For every mini-batch of training, soft translations for a domain are first generated (denoted by $\bar{\mathrm{x}}$ and $\bar{\mathrm{y}}$ in Figure 1a), and are subsequently translated back into their original domains to reconstruct the mini-batch of input sentences. The sum of token-level cross-entropy losses between the input and the reconstructed output serves as the reconstruction loss.

Borrowing from prior work on unsupervised style transfer model (He et al., 2020), the decoder is conditioned on a domain embedding that specifies the direction of translation. In this work, we employ two types of embeddings: A vanilla *binary domain embedding* that takes a bit as input to indicate whether the input text is fluent or disfluent and a *classifier-based domain embedding*. The latter is obtained from a trained standalone CNN-based classifier (Kim, 2014) that predicts the disfluency type of a disfluent input sentence. (Here, we assume that disfluency type labels are available for the disfluent sentences in our training data.) The classifier's penultimate layer acts as our classifier embedding, which is further used to condition the decoder. We hypothesize that additional information about disfluency types via the classifier-based embedding might help guide the process of disfluency correction better.

Furthermore, similar to the noise models adopted by (He et al., 2020; Lample et al., 2017), a randomly sampled noisy version of every sentence in the input mini-batch is fed to the model, forcing it to behave like a denoising auto-encoder. We use noise perturbations (Lample et al., 2017) in the form of word-shuffle(α), word-blank(β) and word-dropout(γ) operations.

We explore two choices to implement our encoder-decoder modules: 1) BiLSTM-based (Bahdanau et al., 2015) and 2) Transformer-based (Vaswani et al., 2017). For the BiLSTM model, as proposed by (He et al., 2020), the BOS vector, i.e., the input to the decoder at the first time-step, is replaced by the domain embeddings. In the Transformer model, this conditioning needs to be carefully done. Figure 1b illustrates how we conditioned the transformer-based decoder. Word embeddings (with their dimensionality reduced) are concatenated with the domain embedding (denoted by DE) at every time-step to form the input for the decoder.

3.2 Semi-Supervised Disfluency Correction

Our unsupervised disfluency correction model can be easily fine-tuned using small amounts of parallel text, when available, lending itself to semi-supervised learning. The encoder-decoder modules are initialized using the unsupervised training described in the previous section and further fine-tuned with a supervised cross-entropy loss using small amounts of parallel disfluent-fluent text. We do not use domain embeddings during semi-supervised training; the inference is done as in the unsupervised model, i.e., with domain embeddings.

4 Experiments and Results

In this work, we use the Switchboard corpus (Godfrey et al., 1992) that includes telephonic conversations and their disfluency annotations (Schriberg, 1994; Zayats et al., 2014). We create a 70:15:15 train, test, and validation split. The train set contains 110,964 sentences, whereas validation and test sets have 11,889 disfluent-fluent sentence pairs each.

Model	BLEU		METEOR	
	Dev	Test	Dev	Test
Disfluent	70.98	71.53	57.02	57.19
US (BiLSTM)	61.26	62.64	48.31	49.13
US (Transformer)	78.72	79.39	56.59	57.25
SS (Transformer)	83.85	85.28	57.77	58.35
Seq2Seq	87.23	88.08	56.65	59.36
BART	89.27	90.08	62.17	63.01

Table 1: BLEU and METEOR scores on the Switchboard dev and test sets. US and SS represent our unsupervised and semi-supervised approaches, respectively.

4.1 Implementation Details

Our BiLSTM model uses a single layer of recurrent units of hidden size 750 with max-pooling over a window size of 5. The noise perturbation parameters, α, β, γ were tuned on the validation set and set to 0. The model was trained for 15 epochs with 10 for annealing, using mini-batches of size 32, with Adam optimizer (Kingma and Ba, 2015) and a learning rate 0.01 linearly scheduled with the rate of decrements of 0.5. Empirically, we also found it essential to allow gradients to pass through the backtranslations to generate meaningful sentences.

The transformer model uses 8 attention heads, word embedding and domain embedding dimensionalities of 1024 and 512. The noise perturbation parameters, α, β, γ are set as 3, 0.2, 0.1. Adam optimizer is used with an initial learning rate of 0.00001, with a linear scheduler and 10 warm-up steps. We used mini-batches of size 32. Dropout (Gal and Ghahramani, 2016) and label-smoothing (Szegedy et al., 2016) values were 0.3 and 0.1, respectively.

4.2 Results

Table 1 shows BLEU and METEOR scores between the gold fluent and the disfluency corrected output from five different models. We train two fully supervised skylines, based on Seq2Seq (Sutskever et al., 2014) and BART (Lewis et al., 2019), to compare against our approaches. The BLEU score using original disfluent text as the hypothesis is 71.53. The two supervised skylines use 55K pairs of parallel disfluent-fluent sentences during training and yield up to 90 BLEU score. In comparison, the unsupervised approach yields up to 80 BLEU scores without any parallel data. Fine-

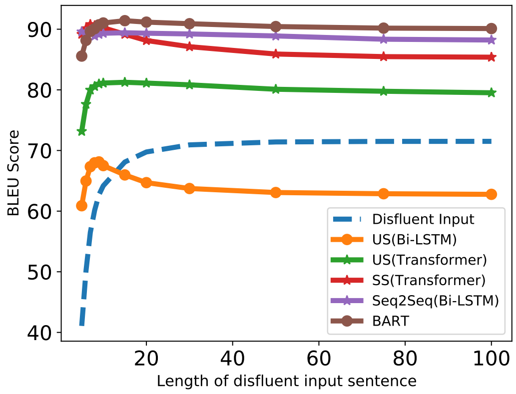


Figure 2: BLEU scores vs. Input lengths.

tuning the unsupervised model with a small parallel corpus containing only 554 pairs (i.e. two orders of magnitude smaller than the complete set of 55K pairs) significantly bridges this gap and yields up to 85 BLEU score. In terms of METEOR, the score using original disfluent text as the hypothesis is 57.19. The difference between unsupervised and supervised approaches is much smaller, indicating that with respect to adequacy or content preservation, these approaches perform at par. These results also show that the last few additional BLEU points (i.e., the difference between BART and SS) come at a high cost with having to create a large parallel corpus.

We obtain 77.34 and 77.97 BLEU on the dev and test sets using binary embeddings, respectively, whereas the disfluency-type classifier embedding yields 78.72 and 76.90 on the dev and test sets. The classifier embeddings do marginally improve performance. However, the BLEU scores obtained using the binary embeddings are almost comparable, which shows that our proposed model can effectively use non-parallel text without any disfluency type labels.

Sentence Length: Figure 2 shows BLEU scores as a function of maximum sentence length on the test set. The BLEU score is highest for the utterances smaller than ten tokens; on longer sentences, the BLEU scores drop. This trend is uniform across all models. Our transformer-based model significantly outperforms the BiLSTM-based model on utterances of all lengths. Interestingly, our semi-supervised approach is very similar in performance to the fully supervised approach for smaller (<10 token) utterances.

#Sentences	Percentage(%)	Dev	Test
0	(Unsupervised) 0	78.72	79.39
554	1	83.85	85.28
2774	5	84.67	86.03
5548	10	84.98	86.12
13870	25	85.88	87.04
27741	50	86.10	87.90
55482	100	87.16	88.22

Table 2: Effect of fine-tuning with a varying amount of supervised parallel corpus to fine-tune our model trained in unsupervised manner; in effect, results of semi-supervised training.

	Set	US BiLSTM	US Trans.	SS Trans.	Seq2Seq	BART
all	Dev	61.26	78.72	84.10	87.23	89.27
	Test	62.64	79.39	85.28	88.08	90.08
conj	Dev	62.68	80.17	84.65	87.63	88.98
	Test	63.60	80.18	86.24	89.13	89.79
filler	Dev	56.96	76.86	81.01	85.01	87.41
	Test	58.45	77.47	82.16	85.92	88.59
restart	Dev	53.76	72.39	78.13	82.24	84.99
	Test	54.92	73.06	79.52	82.84	85.60
disc	Dev	53.84	71.91	81.05	84.64	87.52
	Test	55.39	73.19	82.30	85.93	88.57
edit	Dev	49.06	63.20	78.28	82.86	85.45
	Test	52.51	64.35	80.60	85.82	87.13
aside	Dev	37.07	48.56	41.25	53.98	53.88
	Test	37.25	45.71	51.65	56.37	55.68

Table 3: Disfluency type specific BLEU scores. (Trans.: Transformer, conj: conjunctions and disc: discourse disfluencies).

Semi-supervised Learning: Table 2 shows the performance when our unsupervised model is fine-tuned with varying amounts of parallel text. By having access to only 554 parallel pairs (i.e., 1% total pairs), the performance improves by an impressive 5.89 BLEU on the test set. While BLEU improvements are a monotonically increasing function of the amount of parallel text, we see a trend of diminishing returns soon after the 1% mark.

Performance Across Disfluency Types: Intuitively, certain types of disfluencies (e.g., fillers) are easier to correct than others (e.g., edits). Table 3 reports the BLEU scores from all our models across disfluency types. Conjunctions and discourse disfluencies mark the easy end of the disfluency correction spectrum, while edits and asides mark the challenging end. (Edits are also hard to correct because of the lack of training data.)

Qualitative Analysis: Table 4 shows examples

	Disfluent	**BART**	**Seq-to-Seq**	**US(Bi-LSTM)**	**US(Trans.)**	**SS**	**Fluent**
disc., filler	so uh been a different turn	been a different turn	been a different turn	been a different turn	been a different turn	been a different turn	been a different turn
conj., rep.	but i i i find this whole	i find this whole	i find this whole	anyway i find it all	i find this whole	i find this whole	i find this whole
restart	it’s you’re you’re taking words and developing a picture in your mind	you’re taking words and developing a picture in your mind	you’re taking words and developing a picture in your mind	it’s you’re taking chicken and tobacco words in a mind	it’s taking words and developing and a picture in your mind	it’s taking words and developing a picture in your mind	you’re taking words and developing a picture in your mind
conj., disc., restart	and then you you know you had i think you had to pick it by by by the end of the second you had to pick some sort of major	then you think you had to pick it by the end of the second you had to pick some sort of major	you had to pick it by by the end of the second you had to pick some sort of major	then you think you had i think you had to pick it by by the end of the second you had to pick some sort of major	then you had to pick some sort of major	you had to pick of it by the end of you had to pick some major	by the end of the second you had to pick some sort of major
aside	i forgot sally’s last name anyway it’s a couple	i forgot sally’s last name anyway it’s a couple	i forgot seen last name anyway it’s a couple	gosh i forgot last name it’s a couple of years	i forgot wordstart last name anyway it’s a couple	i forgot harry name anyway it’s a couple	it’s a couple

Table 4: Analysis of generated text across all models. (disc.: Discourse; conj.: Conjunction; rep.:repetition US: Unsupervised; SS: Semi-supervised; Trans.: Transformer.)

using five different models along with the corresponding disfluent and fluent sentences. All five models can remove simple disfluencies (e.g., fillers and discourse) in shorter sentences. Conjunctions and repetitions are removed by all models except the unsupervised BiLSTM model. The third example shows how the transformer model is much better than the BiLSTM model in terms of content retention and adequacy. It also highlights better fluency of the semi-supervised model compared to the unsupervised model. The fourth example illustrates increased complexity due to the presence of multiple disfluency types(*conjunction, discourse, restart*) within a single utterance. The fifth example illustrates a case of an aside, which is difficult for all models. It shows how even the supervised BART model fails to detect the disfluent phrase *”i forgot sally’s last name anyway”*. (Additional contextual information is required for the disfluent phrase to be correctly identified.)

5 Conclusion

We propose an unsupervised disfluency correction model drawing motivation from prior work on unsupervised machine translation and style transfer. We investigate two kinds of domain embeddings for our model. We also present a semi-supervised disfluency correction approach. We finetune our model using only about 500 parallel sentences, which comes very close in performance (based on BLEU scores) to a state-of-the-art, fully supervised system. In future work, we intend to explore how these techniques can be integrated more closely with spoken translation.

Acknowledgements

The authors thank the anonymous reviewers for their helpful feedback and comments. The authors from IIT Bombay gratefully acknowledge support from IBM Research, specifically the IBM AI Horizon Networks-IIT Bombay initiative.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondřej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander Waibel, and Changhan Wang. 2020. FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 1–34, Online. Association for Computational Linguistics.

Nguyen Bach and Fei Huang. 2019. Noisy bilstm-based models for disfluency detection. *Proceedings of Interspeech*, pages 4230–4234.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Eunah Cho, Thanh-Le Ha, and Alex H. Waibel. 2013. Crf-based disfluency detection using semantic features for german to english spoken language translation.

Eunah Cho, J. Niehues, Thanh-Le Ha, and A. Waibel. 2016. Multilingual disfluency removal using nmt.

Qianqian Dong, Feng Wang, Zhen Yang, Wei Chen, Shuang Xu, and Bo Xu. 2019. Adapting translation models for transcript disfluency detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6351–6358.

Ryo Fukuda, Katsuhito Sudoh, and Satoshi Nakamura. 2020. NAIST's machine translation systems for IWSLT 2020 conversational speech translation task. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 172–177, Online. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS, pages 1027–1035, Red Hook, NY, USA. Curran Associates Inc.

Kallirroi Georgila, Ning Wang, and Jonathan Gratch. 2010. Cross-domain speech disfluency detection. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 237–240. Association for Computational Linguistics.

J. J. Godfrey, E. C. Holliman, and J. McDaniel. 1992. Switchboard: telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1.

Hany Hassan, L. Schwartz, Dilek Z. Hakkani-Tür, and G. Tür. 2014. Segmentation and disfluency removal for conversational speech translation. In *INTERSPEECH*.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. In *International Conference on Learning Representations*.

Matthias Honal and Tanja Schultz. 2005. Automatic disfluency removal on recognized spontaneous speech-rapid adaptation to speaker-dependent disfluencies. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I–969. IEEE.

Mark Johnson and Eugene Charniak. 2004. A tag-based noisy channel model of speech repairs. pages 33–39.

Mayank Kaushik, Matthew Trinkle, and Ahmad Hashemi-Sakhtsari. 2010. Automatic detection and removal of disfluencies from spontaneous speech. In *Proceedings of the Australasian International Conference on Speech Science and Technology*, volume 70.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Paria Lou and Mark Johnson. 2020a. Improving disfluency detection by self-training a self-attentive model. pages 3754–3763.

Paria Jamshid Lou and Mark Johnson. 2020b. End-to-end speech recognition and disfluency removal.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA â†' Online. Association for Computational Linguistics.

Nikhil Saini, Jyotsana Khatri, Preethi Jyothi, and Pushpak Bhattacharyya. 2020. Generating fluent translations from disfluent text without access to fluent references: IIT Bombay@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 178–186, Online. Association for Computational Linguistics.

Elizabeth Salesky, Susanne Burger, Jan Niehues, and Alex Waibel. 2018. Towards fluent translations from disfluent speech. In *IEEE Spoken Language Technology Workshop*, pages 921–926.

Elizabeth Salesky, Matthias Sperber, and Alex Waibel. 2019. Fluent translations from disfluent speech in end-to-end speech translation. *CoRR*, abs/1906.00556.

Elisabeth Schriberg. 1994. Preliminaries to a theory of speech disfluencies.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. Cambridge, MA, USA. MIT Press.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and ZB Wojna. 2016. Rethinking the inception architecture for computer vision.

Trang Tran, Jiahong Yuan, Yang Liu, and Mari Ostendorf. 2019. On the Role of Style in Parsing Speech with Neural Models. In *Proc. Interspeech 2019*, pages 4190–4194.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3529–3538, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaolei Wang, W. Che, and T. Liu. 2016. A neural attention model for disfluency detection. In *COLING*.

W. Wang, G. Tur, J. Zheng, and N. F. Ayan. 2010. Automatic disfluency removal for improving spoken language translation. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5214–5217.

V. Zayats, M. Ostendorf, and H. Hajishirzi. 2014. Multi-domain disfluency and repair detection. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pages 2907–2911.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional LSTM. *CoRR*, abs/1604.03209.

Complex Question Answering on knowledge graphs using machine translation and multi-task learning

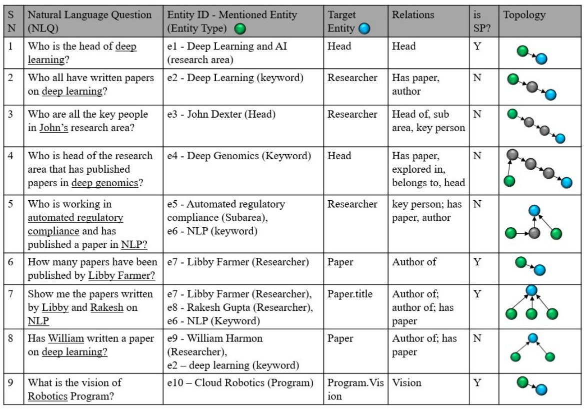
Saurabh Srivastava, Mayur Patidar, Sudip Chowdhury,
Puneet Agarwal, Indrajit Bhattacharya, Gautam Shroff
TCS Research, New Delhi, India
`{sriv.saurabh, patidar.mayur, sudip.chowdhury2`
`puneet.a, gautam.shroff}@tcs.com`

Abstract

Question answering (QA) over a knowledge graph (KG) is a task of answering a natural language (NL) query using the information stored in KG. In a real-world industrial setting, this involves addressing multiple challenges including entity linking, multi-hop reasoning over KG, etc. Traditional approaches handle these challenges in a modularized sequential manner where errors in one module lead to the accumulation of errors in downstream modules. Often these challenges are inter-related and the solutions to them can reinforce each other when handled simultaneously in an end-to-end learning setup. To this end, we propose a multi-task BERT based Neural Machine Translation (NMT) model to address these challenges. Through experimental analysis, we demonstrate the efficacy of our proposed approach on one publicly available and one proprietary dataset.

1 Introduction

Question answering on knowledge graphs (KGQA) has mainly been attempted on publicly available KGs such as Freebase Bollacker et al. (2008), DBPedia Lehmann et al. (2015), Yago Suchanek et al. (2007), etc. There is also a demand for questions answering on proprietary KGs created by large enterprises. For example, KGQA, on a) a KG that contains information related to retail products, can help the customers choose the right product for their needs, or b) a KG containing document catalogs (best practices, white papers, research papers) can help a knowledge worker find a specific piece of information, or c) a KG that stores profiles of various companies can be used to do preliminary analysis before giving them a loan, etc. Our motivating use-case comes from an enterprise system (referred to as *LOCA*) that is expected to answer users' questions about the R&D division of an enterprise.



S N	Natural Language Question (NLQ)	Entity ID - Mentioned Entity (Entity Type) ●	Target Entity ●	Relations	is SP?	Topology
1	Who is the head of deep learning?	e1 - Deep Learning and AI (research area)	Head	Head	Y	
2	Who all have written papers on deep learning?	e2 - Deep Learning (keyword)	Researcher	Has paper, author	N	
3	Who are all the key people in John's research area?	e3 - John Dexter (Head)	Researcher	Head of, sub area, key person	N	
4	Who is head of the research area that has published papers in deep genomics?	e4 - Deep Genomics (Keyword)	Head	Has paper, explored in, belongs to, head	N	
5	Who is working in automated regulatory compliance and has published a paper in NLP?	e5 - Automated regulatory compliance (Subarea), e6 - NLP (keyword)	Researcher	key person; has paper, author	N	
6	How many papers have been published by Libby Farmer?	e7 - Libby Farmer (Researcher)	Paper	Author of	Y	
7	Show me the papers written by Libby and Rakesh on NLP	e7 - Libby Farmer (Researcher), e8 - Rakesh Gupta (Researcher), e6 - NLP (Keyword)	Paper.title	Author of; author of; has paper	Y	
8	Has William written a paper on deep learning?	e9 - William Harmon (Researcher), e2 – deep learning (keyword)	Paper	Author of; has paper	N	
9	What is the vision of Robotics Program?	e10 – Cloud Robotics (Program)	Program.Vision	Vision	Y	

Figure 1: Example queries from a real-world dataset LOCA. Column 6 (is SP?) represents whether the queries can be answered via shortest path or not?, all the other columns are self-explanatory.

Sample questions from LOCA dataset are shown in Figure 1. The schema of the corresponding KG is shown in Figure 2. Answering such questions often requires a traversal of KG along multiple relations which may not form a directed chain graph, and may follow more complex topology as shown for question 5, 7 and 8 in Figure 1. It can also be observed that most often words of the natural language question (NLQ) and corresponding relations have a weak correlation. Most of the proposed approaches on the KGQA task Bollacker et al. (2008) parse the NLQ and convert it into a structured query and then execute the structured query on the KG to retrieve the factoid answers. Such conversion involves multiple sub-tasks: a) linking the mentioned entity with corresponding entity-node in the KG Blanco et al. (2015); Pappu et al. (2017), b) identification of the type of the answer entity Ziegler et al. (2017), c) identification of relations Dubey et al. (2018); Weston et al. (2013); Hakkani-Tür et al. (2014). These tasks are most often performed in a sequence Both et al. (2016); Dubey et al. (2016); Singh et al. (2018), or in parallel Veyseh (2016); Xu et al. (2014); Park et al. (2015), which results

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3428–3439
April 19 - 23, 2021.

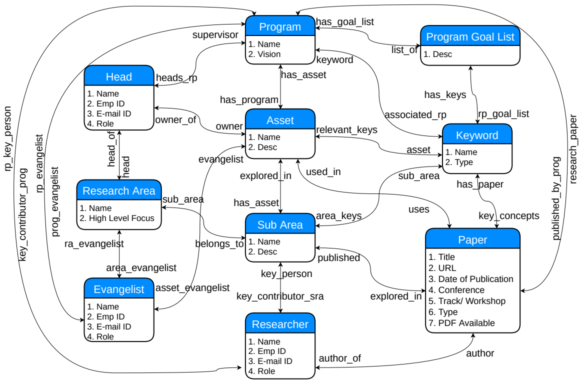


Figure 2: Schema of our propreitary KG LOCA.

in accumulation of errors Dubey et al. (2018). Further, most of the KGQA tasks are not as complex as LOCA. For example, a) All questions of SimpleQA Bordes et al. (2015) can be answered using single triple, b) NLQs for most of the datasets (e.g., SimpleQA, Meta QA) contain only one metioned entity, and c) Even if multiple relations are required for answer entity retrieval, they are organized in a sequence, i.e., chain.

Our motivating example contains specific types of questions that pose many challenges with respect to each of the aforementioned tasks. Moreover, some of the questions can only be answered via a model that attempts more than one sub-tasks together. For example, the first two questions of Figure 1 mention the same words, i.e., "deep learning" but they get associated with two different entity nodes of the KG. Additionally, the prior work could detect the set of relations when the schema sub-graph follows a specific topology, however, in our example, most of the questions follow a different topology. We demonstrate in Section 5 that most of the prior art approaches fail to solve such challenges. We provide a summary of such challenges in Section 2.

In this paper, we propose CQA-NMT, a novel transformer-based, NMT (neural machine translation) model to solve the aforementioned challenges by performing four tasks jointly using a single model, i.e., i) Detection of mentioned entities, ii) prediction of entity types of answer nodes, iii) prediction of topology and relations involved, and iv) question type classification such as 'Factoid', 'Count', etc. CQA-NMT not only performs the four sub-tasks but also helps downstream tasks of mentioned entity disambiguation and subsequent answer retrieval from the KG. The key contributions of this paper are:

(i) We propose a multi-task model that performs all tasks for parsing of natural language question together, rather than the traditional approach of performing these tasks in a sequential manner, which also involves candidate generation based on upstream task and then short-listing them to make the final prediction. We also demonstrate that using such an approach newer types of challenges of the KGQA task can be solved, which have not been attempted by prior work so far.

(ii) We propose the use of neural machine translation based approach to retrieve the variable number of relations involved in answering a complex NLQ against a KG.

(iii) We also demonstrate that every sub-task of parsing an NLQ is complementary to other tasks and helps the model in performing better towards the final goal of KGQA. In Table 3, we have demonstrated that via joint training on more than one task, the accuracy of individual tasks improves as compared to training them separately. For example, when trained separately, the best F1-score for detecting mentioned entity(s) was 83.3, and the best accuracy for the prediction of entity types of answer nodes was 75.7. When trained jointly, we get the corresponding metrics as 87.1 and 76.3. When trained jointly for all tasks, the results improve even further.

(iv) CQA-NMT predicts the relations involved in a sub-graph of KG and also helps to predict the topology of the sub-graph, resulting in compositional reasoning via a neural network on the KG. However, the prior work predicts the relations for a specific topology only[1].

(v) We also demonstrate that our approach outperforms the state-of-the-art approaches on the MetaQA dataset, and therefore we present a new baseline on this dataset. Our approach also performs better than standard approaches

---

[1] Topology is a specific arrangement of how the mentioned entities, and answer entities are connected to each other via the predicted relations. Sample topologies are given in Figure 1. Our approach can be used to answer questions of any topology, if adequate number of samples are included in the training data. The prior works have not attempted a dataset such as LOCA which contains many different topologies. To the best of our efforts we could not find another such dataset, which has led to our aforementioned belief.

as applicable to our dataset and helps us solve most of the real-world industrial challenges.

2 KGQA Problem and Challenges

For answering natural language questions (NLQ), we assume that the background knowledge is stored in a knowledge graph G, comprising of a set of nodes $V(G)$, and edges $E(G)$. Here, nodes represent entities, and edges represent the relationship between a pair of entities or connect an entity to one of its properties. An NLQ (q) is a sequence of words w_i of a natural language (e.g., English), i.e., $q = \{w_1, w_2, ..., w_N\}$. We also assume that the NLQs can mention zero, one, or more entities present in G and enquire about another entity of G, which is connected with the mentioned entity(s). We pose the KGQA problem as a supervised learning problem and next, describe the labels assumed to be available for every question in the training data and that need to be predicted for every question in the test data.

Entity Linking Annotation Some of the n-grams (η_i) in an NLQ refer to entity(s) of KG. Such n-grams have been underlined in Figure 1. The entity-id (as shown in the third column of Figure 1) of the mentioned entity is also assumed to be available as part of label annotation for every question.

Answer Entity Type Annotation (AET), τ: We assume that every NLQ has an entity type (t_i) for the answer entities. These are shown in the middle column of Figure 1. We refer τ as a set of all entity types in the knowledge graph G.

Relation Sequence and Topology Annotation **(path)** Sequence of relations connecting the linked entities to the answer entities can be considered paths $(path_i)$, each of which can contain one or more relations. These paths are connected to form a topology, as shown in Figure 1. This topology of the paths and relations are also assumed to be available for an NLQ in training data. These paths need not be the shortest paths between the linked entities and the answer entities. For example, the last three columns of Figure 1 indicate a) the set of paths separated by a semicolon (;), b) whether this is the shortest path, and c) topology of the paths connecting the linked entities to the answer entities.

Question Type Annotation (q_{type}) Some NLQs can be answered by a single triple of the knowledge graph('Simple'), while some of them require traversal along with more complex topology as indicated earlier('Factoid'), some questions require an aggregate operation such as count ('Count', see question 6, in Figure 1), and finally, some questions perform existence check ('Boolean', see question 8, in Figure 1). Such information is also assumed to be available for every NLQ in training data.

We now describe the challenges that need to be addressed while performing the KGQA task. To the best of our efforts we could not find any prior work that covers all these challenges together.

1. ***Incomplete Entity Mention***: In the NLQ users often do not mention the complete name of the intended entity Huang et al. (2019), e.g., only the first name of a person, short name of a group, etc., e.g., question 8 in Figure 1.

2. ***Co-occurrence disambiguation***: For situations when a mentioned entity should be linked to KG entity with help of another mentioned entity in the question, e.g., in question 7 of Figure 1, there can be many people who have the same first name ('Libby') but there is only one of them who works on NLP, the models needs to use this information to conclusively resolve the mentioned entities Mohammed et al. (2017).

3. ***Avoid un-intended match***: Some of the words in a sentence coincidently match with an entity name but are not an intended mention of an entity, e.g., the word 'vision' may get matched with 'Computer Vision' which is not intended in question 9 of Figure 1.

4. ***Duplicate KG Entity*** The intended entity names may be different from the words used in the NLQ, and there can be more than one entity in the KG that has the same name Shen et al. (2019), for example, "Life Sciences" is the name of a research area, as well as a keyword (see KG schema given in Figure 2). The model needs to link the entity using other words, similar to how it is shown in questions 1 and 2 of Figure 1.

5. ***Relation names mismatch***: Often the words of the KG relations and the words of the NLQ do not match, Huang et al. (2019), e.g., questions 2, 4, 6, etc. in Figure 1.

6. ***Implicit Relations Indication***: Sometimes words of the NLQ do not even make any mention of the relations involved, however, they

need to be inferred Zhang et al. (2018). For example, in question 4 of Figure 1, some of the relations are not mentioned in the question.

Problem Definition: The objective of the proposed approach is to output 1) the mentioned entity(s) (s_i) in the query, 2) the answer entity type , 3) the path or set of predicates, $P_q = \{p_q^1, p_q^2, p_q^3, ..., p_q^N\}$ where, each $p_i \in E(G)$ and, 4) the question type. The set P_q is a sequence of predicates, such that if traversed along these edges from the mentioned entity node(s), we can arrive at the answer entity(s) node(s). The final answer is then retrieved from KG and is post-processed as per the outputs of question type and the 'answer entity type' modules. We assume that we have N training samples $D_{Train} = \{(q_i, \eta_i, t_i, qt_i, path_i)\}_{i=1}^{N}$ available to us where, $q_i \in Q, s_i \subseteq V(G), t_i \in \tau, qt_i \in q_{type}$, and, $path_i \subseteq 2^{E(G)}$.

3 Related Work

In this section, we present a view of prior work, on the KGQA problem as an NLP task, and then on set of techniques used for this task.

3.1 KGQA Task

Berant et al. (2013); Berant and Liang (2014); Reddy et al. (2014); Luo et al. (2018) proposed an approach to perform KGQA by mapping a query to its logical form and then converting it to a formal query to extract answers. However, these are not joint learning tasks as proposed in our work.

Multi-Task based approaches Similar to us, many works like Lukovnikov et al. (2019); Huang et al. (2019); Shen et al. (2019) rely on the jointly learning multiple sub-tasks tasks of KGQA problem. However, all these approaches focus on single-hop relations only, and therefore we cannot take such approaches as a baseline for our model. In a more complex setting, Shen et al. (2019) proposed a joint learning task for entity linking, path prediction (chains topology only), and question type. However, their model does not predict answer entity type. We do not compare our approach with Shen et al. (2019) because they focus on the implicit mention of the entities in previous sentences of dialogue, and also because they do not attempt to predict non-chain topology or the answer entity type.

Non-Chain Multi-Hop Relations Agarwal et al. (2019) proposed an embedding based approach to predict non-chain multi-hop relation prediction (for a fixed and small set of topologies). They perform only one task of relationship prediction.

3.2 Techniques used for KGQA

Transformers and Machine Translation: Transformer Vaswani et al. (2017) has proved to be one of the most exciting approaches for NLP research. They have shown dominating results in Vaswani et al. (2017); Devlin et al. (2018), etc. The paper Lukovnikov et al. (2019) closely resembles our approach as they proposed a joint-learning based multi-task model using Transformer. However, they handle only 1-hop questions and consider relation prediction as a classification task. In its current form it cannot be used to solve the variable length path prediction form, as required in our motivating example. In an extension to the work of using logical forms for KGQA Dong and Lapata (2016) proposed the usage of attention-based seq2seq model to generate the logical form of an input utterance. However, they use an LSTM model and not Transformer.

Graph-Based Approaches GraftNet Sun et al. (2018) and PullNet Sun et al. (2019) approached the problem of KGQA using graph-based solutions. The approach extracts a subgraph related to query and then performs reasoning over it to extract the final answer(s). KV-MEM Bordes et al. (2015) proposed a memory network-based approach for a single relationship prediction.

Embedding Based Approaches Approaches for KGQA using KG embeddings (such as TransE Bordes et al. (2013) and TransR Lin et al. (2015)) were used by Huang et al. (2019) when only one relation (i.e., one RDF triple) is involved. In Bordes et al. (2013); Socher et al. (2013); Dettmers et al. (2018) also one-hop KG modeling approaches were proposed. Recently, Saxena et al. (2020) presented an approach, EmbedKGQA, for joint learning, again using KG Embeddings, in the context of multi-hop relations. However, their approach is not truly a joint model as they perform answer candidate selection via the model, i.e., they arrive at the candidates before executing the model.

Our proposed approach has outperformed PullNet and EmbedKGQA on the MetaQA dataset, as shown in Section 5.

4 Proposed Architecture

In this section, we describe our proposed joint model (CQA-NMT) which is an encoder-decoder

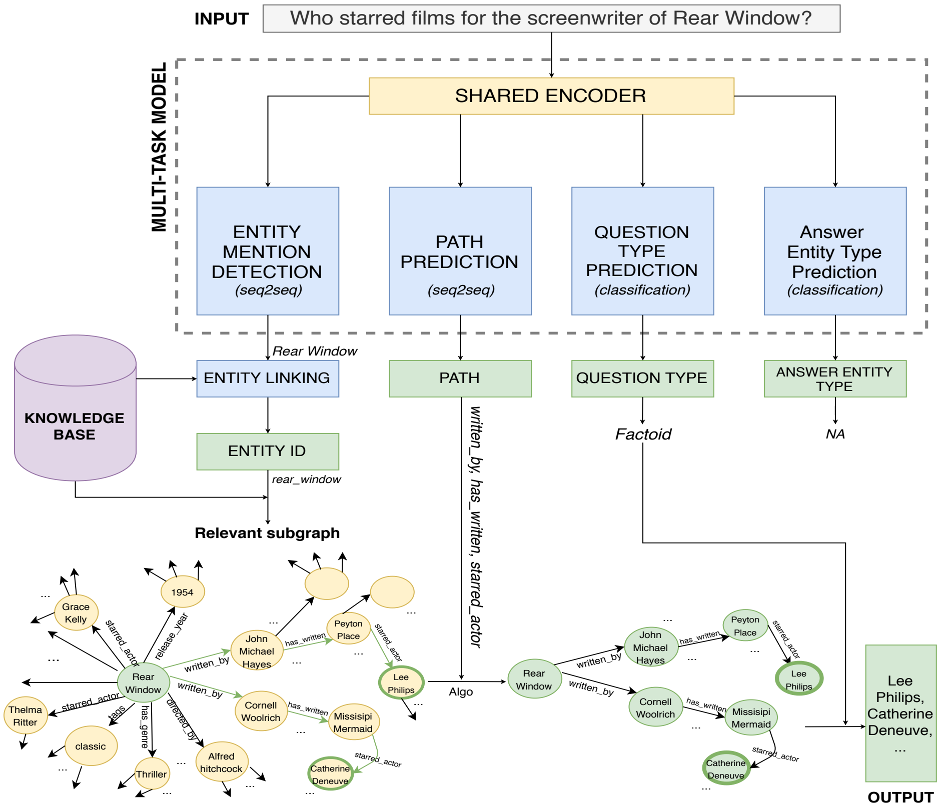


Figure 3: Proposed Multi-task model CQA-NMT. The query is passed through all the modules to (1) Extract the mentioned entity and link it to a KG node. (2) Generate the sequence of predicates required for traversal. (3) Predict the Question type and, (4) Predict AET. After extracting all the information from query, a corresponding formal query such as, SPARQL, is generated to retrieve the final answer.

based model where, BERT Devlin et al. (2018) is used as an Encoder and a Transformer Vaswani et al. (2017) as a decoder. Figure 3 illustrates a high-level view of the proposed model.

Joint Model for KGQA

In this paper, we extend BERT to generate path (or *inference chains*), perform sequence labeling and classification jointly. Details of each module are described next.

1. Entity Mention Detection Module: To extract the mentioned entity(s) from NL query, we performed a sequence labeling task using BERT's hidden states (Figure 4). Sequence labeling is a seq2seq task that tags the input word sequence $x = (w_1, w_2, ..., w_T)$ with the output label sequence $y_{seq} = (y_1, y_2, ..., y_T)$. In this paper, we augmented CQA-NMT to jointly infer the type of the mentioned entity(s) along with its(their) '*span*'. We feed the final hidden states of the tokens $h_2, h_3, ..., h_{T-1}$, into a softmax layer to generate output sequence. Also, we ignore the h_1 and h_T i.e., [CLS] and [SEP] tokens as they can never be a part of an entity(s) and are only required as a preprocessing step of BERT. Since BERT uses WordPiece tokenization, we assigned the same label to the other tokenized input corresponding to their first sub-token. For e.g., the output of BERT's Wordpiece tokenizer is 'Jim Hen ##son' for the input 'Jim Henson'. We assigned the labels for the tokenized output as 'B-Per I-Per I-per', i.e., the second sub-word '##son' was given the same label as the first sub-word 'Hen'. The output of the softmax layer is:

$$y^i_{etype} = softmax(\mathbf{W_{etype}}.h_i + b_{etype}) \quad (1)$$

where, h_i is the hidden state corresponding to the i^{th} token.

2. Entity Linking: The output of the *Entity Mention Detection Module* is a sequence of tokens along with its type (t_i) for a candidate entity. These mentioned entities still need to be linked to a KG node for traversal. In our work, we do not use any neural network for the linking process. Instead, we rely on an ensemble of string matching algorithms[2] and PageRank Page et al. (1999) to break the ties between candidate entities.

The Entity Mention Detection Module outputs as

---

[2] We used Levenshtein Distance and SequenceMatcher packages available in Python

many entities as provided in a query and their associated type (t_i). To link the mentioned entity in the NL query, we extract the candidates from V(G) of type t_i. We then apply 3 string-matching algorithms, similar to (Mohammed et al., 2017) , and take a majority voting to further break the ties. Finally, we apply the PageRank algorithm to link the mentioned entity with a KG entity. One way to understand the usability of the PageRank algorithm is to consider the notion of *popularity*. For e.g., if a user queries 'Where was Obama born?', the user here is more likely referring to the *famous Barack Obama*, compared to any other. A detailed description of the entity mention detection and the entity linking procedure is shown in Figure 4.

3. Path prediction Module: To generate the sequence of predicates for an input query, we augmented our architecture with a Transformer-based Vaswani et al. (2017) decoder which is often used in Neural Machine Translation (NMT) tasks.

We define y_{path}={$p_1, p_2, ..., p_N$} where each $p_i \in E(G)$. In our work, we do not constraint the number of predicates (multiple-hops) that are required to extract the final answer. Hence, an obvious choice was to use a decoder module which can stop generating the predicates once it has predicted the *end-of-sentence* ([EOS]) token (Figure 4).

4. Question Type and Answer Entity Type prediction module: In our work, we formulate the task of determining the question type and the AET as a classification task since we have a discrete set for both q_{type} and *Answer Entity Types*. Using the hidden states of the first special token from BERT, i.e., [CLS], we predict:

$$y_{q_{type}} = softmax(\mathbf{W}_{qt}.h_1 + b_{qt}) \qquad (2)$$

$$y_{\tau} = softmax(\mathbf{W}_{tgt}.h_1 + b_{tgt}) \qquad (3)$$

To jointly model all the task using a single architecture, we define our training objective as:

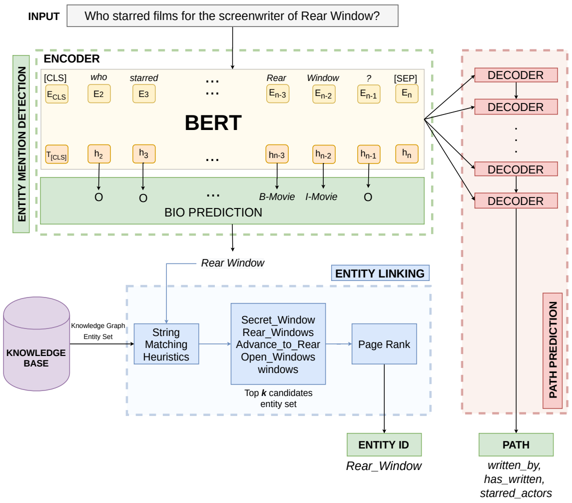
$$p(y|\mathbf{x}) = p(y_{e_{type}}, y_{path}, y_{q_{type}}, y_{\tau}|\mathbf{x}) \qquad (4)$$

$$p(y|\mathbf{x}) = p(y_{q_{type}}|\mathbf{x}).p(y_{\tau}|\mathbf{x}).p(y_{e_{type}}|\mathbf{x}).p(y_{path}|\mathbf{x}) \qquad (5)$$

The path and AET components of CQA-NMT are defined as,

$$p(y_{e_{type}}|\mathbf{x}) = \prod_{n=1}^{N} p(y^i_{e_{type}}|\mathbf{x}), \qquad (6)$$

$$p(y_{path}|\mathbf{x}) = \prod_{t=1}^{T} p(p_t|p_1, p_2, ..., p_{t-1}|\mathbf{x}). \qquad (7)$$



.

Figure 4: Entity Mention and Entity Linking Modules.

where,

$$y_{q_{type}} \in \{\text{factoid, count, boolean, simple}\} \qquad (8)$$

$$y_{\tau} \in \{\text{entity types in KG}\} \qquad (9)$$

$$y^i_{e_{type}} \in \{B, I\} \times \{\text{entity\_types}\} \cup \{O\} \qquad (10)$$

$$p_i \in E(G). \qquad (11)$$

For training we maximize the conditional probability $p(y_{e_{type}}, y_{path}, y_{q_{type}}, y_{\tau}|\mathbf{x})$. The model is fine-tuned end-to-end via minimizing the cross-entropy loss.

5 Experiments and System details

In this section, we first introduce the datasets used for our experiments. We pre-process all NLQs (of all datasets) by downcasing and tokenizing.

5.1 Datasets, Metrices, and Baselines

LOCA Dataset: We introduce a new challenging dataset *'LOCA'*, which consists of 5010 entities, 42 unique predicates, and a total of 45,869 facts. The dataset has 3,275 one or multi-hop questions that have 0, 1, or more entities mentioned in the questions. It contains multiple question types like *count, factoid, and boolean.*

For the questions with multiple entities, we used an operator ";" as a delimiter to separate paths corresponding to each entity (in Figure 1, query 5, 7, and 8). For the scope of this paper, we considered queries involving the only *intersection* which can be replaced with other operators like *union, set-difference, etc.* without loss of any generality. The operator ";" help us detect and predict the different topologies involved in an NL

	Train	Dev	Test
MetaQA 1-hop	96,106	9,992	9,947
MetaQA 2-hop	118,980	14,872	14,872
MetaQA 3-hop	114,196	14,274	14,274
LOCA 1-hop	2,000	250	250
LOCA 2-hop	400	50	50
LOCA 3[or more]-hop	220	28	28

Table 1: Statistics of MetaQA and LOCA dataset

query.

MetaQA: The dataset proposed in Zhang et al. (2018) consists of 3 different datasets namely, *Vanilla*, *NTM*, and, *Audio Data*. All the datasets contain single and multi-hop (maximum 3-hop) questions from the *movie* domain. For our experiments, we used the Vanilla and the NTM version of the datasets and the KB as provided in Zhang et al. (2018). Since, both versions of MetaQA do not consider the AET and question type, we assigned a default label to both the tasks.

Metrics: We used different metrics for different subtasks. Since a query can contain partially mentioned entities, we used F-score to evaluate mention and its type detection module. For Inference Chain (or Path prediction), question type, and, answer entity type prediction we use the accuracy measure. In Table 2, similar to prior works, we have used the Hits@1 to evaluate the query-answer accuracy.

Baselines: We have used, KV-mem Bordes et al. (2015), GraftNet Sun et al. (2018), PullNet Sun et al. (2019), VRN Zhang et al. (2018), and EmbedKGQA Saxena et al. (2020) as the baselines.

5.2 Training Details

All the baselines and the proposed approach were trained on DGX 32GB NVIDIA GPU using TensorFlow Abadi et al. (2015) and Texar Hu et al. (2018) libraries. For CQA-NMT, we used the small uncased version of pre-trained BERT Devlin et al. (2018) model. Adam Kingma and Ba (2014) optimizer was employed with a learning rate of $2e$-5 for BERT and default for others. The training objective of each model was maximized using the cross-entropy loss and the best models were selected using the validation loss. Dropout values were set to .5 and were optimized as described in Srivastava et al. (2014). For BERT we used 10% of total training data for the warmup phase of BERT Vaswani et al. (2017). Finally, for the division of dataset into train, test, and, dev, we used the same split as provided by Zhang et al. (2018) for the MetaQA dataset and a ratio of 80-10-10 for LOCA dataset.

5.3 Main Results

In this section, we report the results of the experiments on the MetaQA and the LOCA dataset. Next, we provide insights into the model outputs and results of error-analysis performed on LOCA dataset.

5.3.1 LOCA

The experimental results for LOCA dataset are shown in the last row of table 2. The results affirm that the proposed approach outperforms the baselines. We observed that the baselines' inability to handle ***Duplicate KG Entity*** (Section 2 challenge 4) limits their performance. Additionally, the ability of the NMT Bahdanau et al. (2014) model to effectively handle complex and un-known topologies helped us retrieve answers with better accuracy for variable-hop (v-hop) queries.

5.3.2 MetaQA

The experimental results for MetaQA are shown in table 2. For *Vanilla* MetaQA, we achieved better answer accuracy on 1-hop and 3-hop settings. However, in a 2-hop setting, we were able to achieve comparable results to the state-of-the-art. An increment of about 2% and 4.9% Hits@1 can be seen in the 1-hop and 3-hop settings.

To obtain the performance of each baseline on v-hop (variable-hop) dataset, we re-use the existing models for 1-hop, 2-hop, and 3-hop and assume that there is an oracle which can redirect query to the correct model. Thus estimated accuracy of various approaches is shown in the 4^{th} row of Table 2, while the actual results on v-hop dataset are shown in the 5^{th} row. It is evident that CQA-NMT outperforms all the baselines on MetaQA dataset in variable hop setting.

To gauge the effectiveness and robustness of our model, we used the same models trained on vanilla MetaQA dataset and evaluated its performance on *NTM* MetaQA, i.e., in zero-shot setting. For this, we achieved better results on 1 and 3-hop. The worse performance of CQA-NMT on MetaQA-NMT(2-hop) can be because of zero shot setting. Because, as compared to VRN, we have not trained CQA-NMT on MetaQA-NTM dataset, we trained it on MetaQA vanilla dataset only.

	KV-Mem	GraftNet	PullNet	EmbedKGQA	VRN	CQA-NMT
MetaQA(1-hop)	96.2	97.0	97.0	97.5	97.5	**99.96**
MetaQA(2-hop)	82.7	94.8	**99.9**	98.8	89.9	99.45
MetaQA(3-hop)	48.9	77.7	91.4	94.8	62.5	**99.78**
MetaQA(V-hop) Estimated	73.79	89.11	96.05	-	-	**99.69**
MetaQA(V-hop)	-	83.67*	-	-	-	**95.85**
MetaQA-NTM(1-hop)	-	-	-	-	81.3	**83.3**
MetaQA-NTM(2-hop)	-	-	-	-	**69.7**	62.1
MetaQA-NTM(3-hop)	-	-	-	-	38.0	**81.3**
LOCA	51.07	72.27	-	-	-	**78.29**

Table 2: Results of the baselines and CQA-NMT on MetaQA (vanilla and NTM) and LOCA dataset. Results for MetaQA-NTM were obtained in zero-shot setting for CQA-NMT. All other baseline results are taken from Sun et al. (2019), Zhang et al. (2018), and Saxena et al. (2020), except for * marked numbers. Note: Source code for PullNet, VRN is not available. We were not able to replicate the same results of KV-Mem.

	Mention Detection (F1-score)	AET (Accuracy)	Inference Chains (Accuracy)	Question Type (Accuracy)	Answer (Accuracy)
Unsupervised	*67.12*	*53.3*	*51.0*	-	42.9
BERT (mention only)	83.3	*53.22*	*51.9*	-	47.22
BERT (AET only)	*67.38*	75.7	*55.6*	-	45.2
BERT (IC only)	*67.8*	*50.9*	80.1	-	40.1
BERT (mention and AET)	87.1	76.3	*71.6*	-	65.33
BERT (AET and IC)	*66.3*	74.1	77.8	-	60.39
BERT (Mention and IC)	87.3	*53.1*	79.6	-	51.3
CQA-NMT	**93.66**	**79.89**	**81.95**	**97.65**	**71.01**

Table 3: Effects for reducing the supervision from our approach. The numbers in *italics* are obtained without any supervision.

5.4 Further Results and Analysis

Advantage of Transformers: In the LSTM based implementation of mentioned entity detection, it could not detect different entity types for the same phrase "deep learning" in query 1 and 2 of Figure 1. However, in BERT-based approach it was able to. We therefore infer that such phenomenon could occur due to key features of BERT such as multi head attention, WordPiece embeddings, Positional embeddings, and/ or Segment embeddings. Moreover, in a different context, it was able to assign different types to the entities with the same mentions (Query 1 and 2 from Figure 1).

Effects of using less annotations: To study the importance of annotation in our approach, we removed several components from our proposed approach and studied the effects (Table 3). We first studied CQA-NMT after removing all the supervision and used heuristics-based-approaches for AET and Mention Detection (both the approaches were taken from Mohammed et al. (2017)). The shortest path, similar to Sun et al. (2018, 2019), between the linked KG entity and AET, was then taken to retrieve the answers. This setting (row 1) results in the worst performance. In row 2, 3, and 4 of Table 3, we kept only one component of CQA-NMT as supervised and applied heuristics for others as mentioned above. As evident from these rows, mention detection plays a crucial role in extracting the correct answer (a jump in range of 2%-5% in answer accuracy). A similar analysis can be found in Dong and Lapata (2016); Guo et al. (2018); Srivastava et al. (2020), where authors found that entity linking error is one of the major errors leading to wrong predictions in KGQA.

In summary, while testing each components of CQA-NMT, we tried supervising different components at a time and used heuristics based approaches for the remaining components. The heuristics are:

a) Shortest path algorithm, similar to Sun et al. (2018, 2019), for Path Prediction Task.

b) Candidate and Relation Pairing Score Mohammed et al. (2017) for Entity Linking.

c) LSTM based Classifier Mohammed et al. (2017) for Answer Entity Type Prediction. [however in Row 1 "Unsupervised", we identify the AET if name of the entity-type is present in the NLQ].

Benefits of joint training: From row 5-7 of Table 3, we infer that joint training not only improves the scores of individual components (in range 15%-20%) but also, the overall answer accuracy. We observed that the challenges 5 and 6 from section 2 were handled significantly better after jointly training CQA-NMT for AET and mention detection (row 5). We found that the context used by a mentioned entity for AET was different in different queries. For e.g., in queries 'Who heads Deep Learning?' and 'papers in Deep Learning', *Deep Learning*, is a *research area* with AET 'head' in the former, and in the later, is a *keyword* with AET 'paper.title'. After jointly training the mention detection model with AET or IC, a jump of ∼4% can be observed for the individual components and ∼11-20% in answer accuracy.

Errors with Joint Training: From table 3, it seems that the joint-training helps. However, we also found that the error made by a single module produces a cascading of errors. For example, if for a **single** mention span 'deep learning and ai' with the gold label as 'B-research_area I-research_area I-research_area I-research_area', the entity detection predicts 'B-research_area I-research_area O B-keyword', the path prediction module generates two different paths, one for each entity, leading to wrong answer.

Similarly, if AET prediction module makes an 'error', it gets propagated to all the other modules simultaneously. The frequency of such error, however, was considerably small and in range of 1-2% of total training or test data.

Robustness against variable hops: We combined all the 3 vanilla MetaQA datasets into a single dataset. The results on this new dataset, *MetaQA v-hop (variable-hop)* is shown in table 2. Since the other approaches in the literature did not perform such analysis, MetaQA v-hop is a new challenge that we propose.

Motivation for PageRank: When we have more than one candidate entity for a mentioned entity, we want to choose the one with higher popularity (Sec 4). One of the most well established measure of popularity of nodes in graphs is PageRank. Therefore we have used it. Further, when more than one entity are mentioned in an NLQ, there can be more than one candidate entity for each of them. The graph-based approach also helps us choose the candidates that are well connected.

We also experimented using other measures such as in-degree and out-degree of nodes. However, for LOCA dataset, we achieved an increment of 22% using PageRank on Entity Linking task, as compared to the in-degree and out-degree measures. PageRank also helped in reducing the challenges 1-2 from Sec. 2.

5.4.1 Retrieval of answer(s) from KG

The final objective of a KG-QA system is to retrieve the correct answer from KG against a query q. To this end, we use the outputs of the different components of CQA-NMT and feed them to complete the pre-written SPARQL sketchs. We defined a bunch of rules for different question-types and used a simple-mapping rules to map the queries to the sketches. For e.g., consider the query, q = "Who is working in automated regulatory compliance and has published a paper in NLP?". The output of CQA-NMT contains all the information that is required to form a structured query such as SPARQL. The outputs of CQA-NMT are:

1. Linked Entities: {e5: automated regulatory compliance (sub-area), e6: NLP (keyword)}
2. Inference Chain: key_person; has_paper, author
3. Answer Entity Type (AET): researcher.name
4. Question Type (q_{type}): Factoid

After using the q_{type} information, we fill a sketch using other outputs. The generated SPARQL query is:

SELECT DISTINCT ?uri WHERE {<e5> <key_person> <?uri> . <e6> <has_paper> <?x> . <?x> <author> <?uri>}.

Where, e5 and e6 are unique identities assigned to 'automated regulatory compliance' (of type *sub-area*) and NLP (of type *keyword*).

6 Conclusion

We presented a complex version of the KGQA problem, which involves mention of multiple entities in the question. Multiple sequence of relationships combined in complex topologies, are required to answer such questions. It is evident that such questions, while required to be answered in real world industrial setting, cannot be answered using prior approaches. We propose a novel CQA-NMT model to answer such questions and have performed a detailed comparison of our approach with prior art on MetaQA and Loca datasets. We have shown that CQA-NMT not only solves more complex task, but also performs better on MetaQA dataset as compared to baseline approaches.

Acknowledgements

We would like to thank the reviewers of this paper for their valuable comments and all the people invlolved in this work. We would also like to thank Vaibhav Varshney (varshney.v@tcs.com) for the diagrams and his valuable comments toward this paper.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Puneet Agarwal, Maya Ramanath, and Gautam Shroff. 2019. Retrieving relationships from a knowledge graph for question answering. In *European Conference on Information Retrieval*, pages 35–50. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 179–188.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Andreas Both, Dennis Diefenbach, Kuldeep Singh, Saedeeh Shekarpour, Didier Cherix, and Christoph Lange. 2016. Qanary–a methodology for vocabulary-driven open question answering systems. In *European Semantic Web Conference*, pages 625–641. Springer.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. Earl: joint entity and relation linking for question answering over knowledge graphs. In *International Semantic Web Conference*, pages 108–126. Springer.

Mohnish Dubey, Sourish Dasgupta, Ankit Sharma, Konrad Höffner, and Jens Lehmann. 2016. Asknow: A framework for natural language query formalization in sparql. In *European Semantic Web Conference*, pages 300–316. Springer.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2018. Dialog-to-action: Conversational question answering over a large-scale knowledge base. In *Advances in Neural Information Processing Systems*, pages 2942–2951.

Dilek Hakkani-Tür, Asli Celikyilmaz, Larry Heck, Gokhan Tur, and Geoff Zweig. 2014. Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Zhiting Hu, Haoran Shi, Bowen Tan, Wentao Wang, Zichao Yang, Tiancheng Zhao, Junxian He, Lianhui Qin, Di Wang, Xuezhe Ma, et al. 2018. Texar: A modularized, versatile, and extensible toolkit for text generation. *arXiv preprint arXiv:1809.00794*.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 105–113.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2019. Pretrained transformers for simple question answering over knowledge graphs. In *International Semantic Web Conference*, pages 470–486. Springer.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2017. Strong baselines for simple question answering over knowledge graphs with and without neural networks. *arXiv preprint arXiv:1712.01969*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Aasish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. 2017. Lightweight multilingual entity extraction and linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 365–374.

Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, and Gary Geunbae Lee. 2015. Isoft at qald-5: Hybrid question answering system over linked data and text data. In *CLEF (Working Notes)*.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. *Proceedings of the 2020 conference on Association for Computational Linguistics*.

Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. *arXiv preprint arXiv:1910.05069*.

Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, et al. 2018. Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference*, pages 1247–1256.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.

Himani Srivastava, Prerna Khurana, Saurabh Srivastava, Vaibhav Varshney, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2020. Improved question answering using domain prediction.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Amir Pouran Ben Veyseh. 2016. Cross-lingual question answering using common semantic space. In *Proceedings of TextGraphs-10: the workshop on graph-based methods for natural language processing*, pages 15–19.

Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction. *arXiv preprint arXiv:1307.7973*.

Kun Xu, Sheng Zhang, Yansong Feng, and Dongyan Zhao. 2014. Answering natural language questions via phrasal semantic parsing. In *Natural Language Processing and Chinese Computing*, pages 333–344. Springer.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

David Ziegler, Abdalghani Abujabal, Rishiraj Saha Roy, and Gerhard Weikum. 2017. Efficiency-aware answering of compositional questions using answer type prediction. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 222–227.

Recipes for Adapting Pre-trained Monolingual and Multilingual Models to Machine Translation

Asa Cooper Stickland[♣] **Xian Li**[♠] **Marjan Ghazvininejad**[♠]

[♣] University of Edinburgh, [♠] Facebook AI
a.cooper.stickland@ed.ac.uk, {xianl,ghazvini}@fb.com

Abstract

There has been recent success in pre-training on monolingual data and fine-tuning on Machine Translation (MT), but it remains unclear how to best leverage a pre-trained model for a given MT task. This paper investigates the benefits and drawbacks of freezing parameters, and adding new ones, when fine-tuning a pre-trained model on MT. We focus on 1) Fine-tuning a model trained only on English monolingual data, BART. 2) Fine-tuning a model trained on monolingual data from 25 languages, mBART. For BART we get the best performance by freezing most of the model parameters, and adding extra positional embeddings. For mBART we match or outperform the performance of naive fine-tuning for most language pairs with the encoder, and most of the decoder, frozen. The encoder-decoder attention parameters are most important to fine-tune. When constraining ourselves to an out-of-domain training set for Vietnamese to English we see the largest improvements over the fine-tuning baseline.

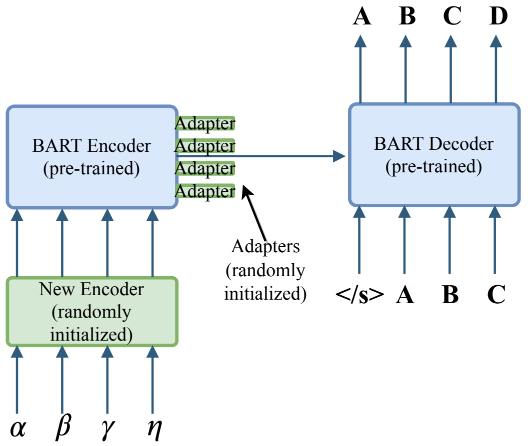


Figure 1: Schematic diagram showing the components of our system for adapting BART to MT. We learn a new encoder that takes as input the source language, with a potentially different vocabulary to the original BART system. We freeze most BART parameters (frozen model components are shown in blue).

1 Introduction

Machine Translation (MT) has recently seen significant advances, with improvements in modeling, especially since the advent of neural models (Sutskever et al., 2014; Bahdanau et al., 2015), and the availability of large parallel corpora for training such systems (Smith et al., 2013; Kocmi and Bojar, 2017; Tiedemann, 2012). However, often standard neural systems do not perform well on *low-resource* language pairs (Koehn and Knowles, 2017), especially when the language pairs are only distantly related. Since these languages are spoken by a large fraction of the world's population, reducing the gap in performance between high and low-resource MT could have a large impact.

An explosion of interest in large-scale pre-training in Natural Language Processing has led to increased performance on smaller datasets, by simple *fine-tuning* of large pre-trained models on downstream tasks. The typical approach is to train a large model on text from the web (for example English Wikipedia), with a common objective predicting masked out tokens using the unmasked context. For Natural Language Generation (for example summarization of text), performance can be improved by pre-training a sequence-to-sequence model (Song et al., 2019; Lewis et al., 2019).

However previous work has shown that on NLP tasks such as Natural Language Inference, the relative performance of fine-tuning vs. keeping the pre-trained model frozen depends on the similarity of the pre-training and downstream tasks (Peters et al., 2019). We observe empirically that simple fine-tuning of a monolingual model for MT can result in worse performance than training from scratch (e.g. Table 1). For MT the more common mono-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3440–3453
April 19 - 23, 2021.

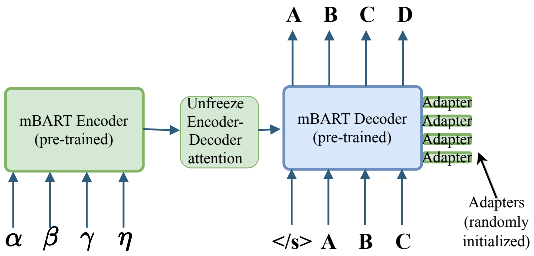


Figure 2: Schematic diagram showing one method of adapting mBART to MT, unfreezing the encoder and encoder-decoder attention, and adding adapters in the decoder. Model components colored blue are not updated during fine-tuning.

lingual (usually only English) pre-training (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Yang et al., 2019b; Liu et al., 2019) may be inadequate since the input or output domain for the downstream task will be a non-English language.

Multilingual pre-training offers a solution, by modifying the pre-training objective to include many languages. Using a multilingual pre-trained model for MT gives good performance, especially on lower-resource language directions (Liu et al., 2020). However it is challenging to balance the training data so that higher-resource languages do not overwhelm lower-resource ones (Arivazhagan et al., 2019; Conneau et al., 2019). For a particular language it may be hard to source monolingual data, or it may be simply not included in training.

We also consider multilingual MT (training on many language pairs and sharing all or most model parameters) as a downstream task. Sharing 'knowledge' across language directions can improve performance on low-resource language pairs by transfer from other pairs included in training. Previous work observed problems of performance degradation, often on high-resource languages, due to interference and constrained capacity (Johnson et al., 2017; Tan et al., 2019). And when initialising from a pre-trained model, we want to avoid 'catastrophic forgetting', where by fine-tuning on a particular language pair we lose the knowledge about another language pair that is stored in the model weights.

Previous work has explored how to improve on simple fine-tuning, by freezing pre-trained model parameters (Peters et al., 2019; Houlsby et al., 2019) and using lightweight 'adapter modules' (Houlsby et al., 2019; Stickland and Murray, 2019) which are inserted between the layers of the pre-trained network. We aim to explore and improve on these approaches for both bilingual and multilingual MT (in contrast to previous work largely focusing on text classification). We explore freezing different subsections of the pre-trained model.We expect freezing to be particularly useful when the parallel data is of low quality, in which case naive fine-tuning may, for example, over-specify the pre-trained model to a particular domain.

Our main contributions are:

- A novel fine-tuning approach, similiar to Lewis et al. (2019) but with adapter modules in the encoder of the pre-trained sequence-to-sequence model and combining both learnable, and fixed sinusoidal, positional embeddings in the input module (see sections 3.1 and 3.2) that feeds into the pre-trained encoder.

- Extensive experiments with fine-tuning a multilingual pre-trained model for MT, showing the benefits and drawbacks of freezing various parameters. We find we should freeze the decoder but unfreeze the encoder-decoder attention when fine-tuning on $Xx \rightarrow En$ data, and in the other direction we should freeze the encoder but unfreeze the entire decoder (section 5.3). We find monolingual models benefit more from freezing parameters than multilingual models (section 5.2).

- Results on fine-tuning a multilingual pre-trained model for multilingual MT showing that freezing parameters improves performance on some, mostly distantly related, language directions (section 5.5).

2 Background and Related Work

BART and mBART We briefly describe the pre-trained models we focus on in this work. In order to perform machine translation with the minimum of modifications to the pre-trained model, we prefer models that can perform conditional sequence generation. We concentrate on the BART (Bidirectional and Auto-Regressive Transformer) model (Lewis et al., 2019) and the multilingual BART (mBART; Liu et al., 2020) model. BART and mBART are sequence-to-sequence models with the standard transformer-based neural machine translation architecture, i.e. an encoder and autoregressive decoder. The pre-training task they are trained on is reconstructing a document from a noisy version of that document (so called 'denoising autoencoder'). Examples of noise added to the training data include randomly shuffling the order of the original sentences, randomly changing the start position of the document, and using a masking scheme where arbitrary length spans of text are replaced with a single mask token. BART and mBART are trained entirely on monolingual data from the web, with English data for BART and data from 25 different languages for mBART.

BART and mBART have almost identical architectures, with 12 encoder layers and 12 decoder layers with model dimension of 1024 and 16 attention heads. BART has a vocabulary of approximately 40k and $\sim$ 406M parameters, whereas mBART has a larger vocabulary of size 250k and $\sim$ 610M parameters.

Pre-trained Models for MT There has been much recent progress in pre-training for NLP applications (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Yang et al., 2019b; Liu et al., 2019), with the most relevant for our work focusing on text generation (Radford et al., 2019; Song et al., 2019; Dong et al., 2019; Raffel et al., 2019; Lewis et al., 2019) Specifically for MT, Ramachandran et al. (2017) proposed pre-training the encoder-decoder modules as two separate language models, and Yang et al. (2019a); Zhu et al. (2020) explored approaches incorporating BERT model weights into the usual seq-to-seq architecture.

Multilingual MT *Multilingual translation* (Firat et al., 2016; Viégas et al., 2016; Aharoni et al., 2019; Arivazhagan et al., 2019) aims to jointly train one translation model that translates multiple language directions, and shares representations to improve the translation performance on low-resource languages (Gu et al., 2018). Our freezing approach is similar in spirit to Sachan and Neubig (2018) who investigate which parameters are most useful to share for multilingual MT with transformer models. We start from a multilingual pre-trained model, and decide between sharing or freezing parameters.

Transfer Learning for MT *Transfer learning* hopes to leverage a related task to perform well on a target task, for example by initialising the model weights from those resulting from training on a related task. For MT various approaches have been explored, with a common method training on high-resource language(s) and fine-tuning on a low-resource language (Neubig and Hu, 2018).

Closely related to our work is that of Bapna and Firat (2019), who introduce freezing and adapters (extra parameters inserted within the transformer) for domain adaption in MT. They take an MT model trained on a large parallel corpus, and fine-tune in a different domain (e.g. legal text). We differ in that we start from a pre-trained model that has not been trained on parallel text, and study adapting it to MT. Approaches based on freezing various model components have also been proposed (Thompson et al., 2018; Zoph et al., 2016), but have focused on RNN models pre-trained with parallel data, not transformer models pre-trained on monolingual data.

3 Methods

Because BART has been trained on only English input, we need to use different techniques when fine-tuning BART and mBART for MT, with a schematic overview shown in Figure 1 and Figure 2. BART and mBART are standard sequence-to-sequence models, where an *encoder* consumes a sequence of source-side tokens, and a *decoder* acts as a conditional language model, generating target tokens given a source sequence. Intuitively, we want the encoder and decoder to be performing roughly the same tasks during fine-tuning as they were during pre-training. For BART this means the input to the encoder should be similar to (embedding vectors of) noisy English text. Therefore when training on say, Vietnamese to English, we first transform the Vietnamese source sentence into a representation useful for BART. We introduce new parameters (the 'Input Module') that consume the source sentence and produce hidden vectors we can feed into the BART encoder. We describe the Input Module architecture in section 3.1.

mBART can be fine-tuned without modification since during pre-training it saw the languages it will be fine-tuned on. To increase flexibility when freezing parts of the network, we optionally add extra parameters to both BART and mBART, described in section 3.3.

3.1 Input Module Architecture

We refer to the network that takes in the source language text and outputs hidden vectors useful for BART as an 'Input Module' or $\mathrm{IM}(\cdot)$. To improve performance on low-resource MT, we use smaller token embedding vectors on the source side of size $d_{\mathrm{s}} = 512$, whereas BART uses hidden vectors of size $d_{\text{BART}} = 1024$. The full network is as follows, with $\{\mathbf{e}_t\}_{t=0}^{l}$ token embeddings for a source sentence with l tokens,

$$\text{BART}(\text{IM}(\{\mathbf{e}_t\}_{t=0}^{l})), \tag{1}$$

where $\text{BART}(\cdot)$ is the full BART encoder-decoder model. Where we would normally input token embeddings to the BART model we use the outputs of the Input Module. The t-th element of $\text{IM}(\{\mathbf{e}_t\}_{t=0}^{l})$ as follows:

$$\alpha\text{LN}(\mathbf{W}\text{Transformer}(\{\mathbf{e}_t\}_{t=0}^{l})_t) \tag{2}$$

and where $\text{LN}(\cdot)$ is layer-norm, $\mathbf{W}$ is a matrix projecting up from d_{s} to d_{BART}, and $\text{Transformer}(\cdot)$ is the application of a series of Transformer layers. α is a scalar, in our case equal to $\sqrt{d_{\text{BART}}}$, which is required to insure the input to BART is on the same scale as the embedding vectors BART was trained on. If we remove $\text{LN}(\cdot)$, $\mathbf{W}$ and α, and set $d_{\text{s}} = d_{\text{BART}}$, we recover the method introduced by Lewis et al. (2019) for fine-tuning BART on MT.

3.2 Extra Positional Embeddings

We found empirically that the details of positional embedding vectors are important for good performance (see Table 1), perhaps because of the need for the BART model to deal with different word order to that it was trained on. Transformer models normally have either learnable positional embedding vectors, or fixed sinusoidal positional embedding (Vaswani et al., 2017) vectors $\mathbf{p}^t$, with $\mathbf{p}_i^t = \sin(t/10000^{i/(d_s/2-1)})$, if $0 \leq i < d_s/2$, and $\mathbf{p}_i^t = \cos(t/10000^{(i-(d_s/2-1))/(d_s/2-1)})$ if $d_s/2 \leq i < d_s$, where t indexes position and i indexes dimension.

Note that positional embedding are typically only added to the token embeddings. We use learnable positional embeddings at the embedding layer. But to get extra positional information, we optionally add fixed sinusoidal positional embedding to the input of each transformer layer in $\text{IM}(\cdot)$, i.e. the input to layer i, $\mathbf{h}_t^i = \mathbf{o}_t^{i-1} + \mathbf{p}^t$, with $\mathbf{o}_t^{i-1}$ the previous layer output. This means the network has access to both *learned positional embeddings* (only at the embedding layer), and *fixed sinusoidal* ones at the input to each layer.

3.3 Within-Network Adapter Architecture

When freezing parts of a pre-trained model (either BART or mBART in our case), we may want to add flexibility by modifying the pre-trained model architecture. One approach is to use 'adapters', introduced by Houlsby et al. (2019); Stickland and Murray (2019) which are newly-initialised neural network layers that can be 'slotted in' to the layers of the pre-trained model.

We only considered simple adapter architectures, essentially feed-forward networks, with one hidden layer, and a residual connection to the output. The dimension of the hidden layer can be much smaller than the model dimension to reduce computational cost and parameter count. We use one adapter per transformer layer, inserting them at the end of the layer (Stickland and Murray, 2019; Bapna and Firat, 2019). We use the following architectures, with $\mathbf{h}$ the hidden state of a particular token after the usual transformer layer, and $\mathbf{h}_{\text{out}}$ the hidden state of the token after the adapter layer:

$$\begin{aligned} \mathbf{z} &= \text{gelu}(\mathbf{W}_d\mathbf{h}) \\ \mathbf{h}_{\text{out}} &= \tanh(\mathbf{W}_u\mathbf{z}) + \mathbf{h} \end{aligned} \tag{3}$$

The $\tanh$ non-linearity helped with stability in early experiments, probably because it prevents the adapter output exploding by constraining it between -1 and 1.

We also considered a version of the adapter based on the 'gated linear unit' (GLU; Dauphin et al., 2016) architecture:

$$\begin{aligned} \mathbf{z} &= 2\sigma(\mathbf{W}_g\mathbf{h}) \odot \text{gelu}(\mathbf{W}_d\mathbf{h}) \\ \mathbf{h}_{\text{out}} &= \tanh(\mathbf{W}_u\mathbf{z}) + \mathbf{h}. \end{aligned} \tag{4}$$

We found the network was sensitive to changes in the magnitude of the hidden states the adapter produced, and therefore multiply the sigmoid gate by 2 so that it approximately leaves the magnitude of the hidden states unchanged.

3.4 Freezing Details

BART We freeze all parameters of BART except the weights and biases of the layer-norm modules (following Houlsby et al. (2019)), and additionally unfreeze the self-attention module of the first layer in the BART encoder, which is a small fraction of total BART parameters ($24 \cdot 2d_{\text{BART}}$ from layer-norm parameters and $4d_{\text{BART}}^2$ from the self-attention module). We freeze BART token embeddings (used in the softmax layer).

mBART In most of our experiments we unfreeze layer-norm parameters, positional and token embeddings, and either the entire encoder or decoder

Languages	Vi-En	Tuned Params (m)
(1): BART + InputModule (unfreeze all)	9.5	374
(2): BART (frozen) + InputModule	27.9	26
(3): (2) + unfreeze layer-norm	28.4	26
(3) + sinusoidal positional embeddings	18.3	26
(1) + extra positional embeddings	22.0	26
(4): (3) + extra positional embeddings	29.0	26
(5): (3) + encoder adapters	28.9	29
(3) + decoder adapters	28.3	29
(6): (5) + extra positional embeddings	30.0	29
(7): (6) + GLU adapters	30.5	29

Table 1: Ablation study for various choices in the frozen BART method, with validation set BLEU score. We organise model settings by a number in brackets, (n), and define a new model configuration in bold as **(n)**:. We use '+' to indicate the addition of new model settings on top of the previous ones. Method **(2)** is similar to the method introduced by Lewis et al. (2019). '+ sinusoidal positional embeddings' refers to adding sinusoidal positional embeddings to token embeddings, while '+ extra positional embeddings' refers to adding them within each transformer layer (see section 3.2). 'Tuned Params (m)' refers to the number of tunable parameters for each method in millions. Test set results are listed in Table 3 (as 'Frozen BART').

Languages	It-En	Si-En
(1): BART + InputModule + LN	34.1	5.1
(2): (1) + encoder adapters	35.0	7.3
(1) + decoder adapters	35.5	6.8
(3): (2) + extra pos. embeddings	36.3	8.7
(4): (3) + GLU adapters	35.7	9.2

Table 2: Further Ablation study for key settings of the frozen BART method, with validation set BLEU score. Test set results are listed in Table 3 (as 'Frozen BART').

module (or the encoder and subsections of the decoder). We unfreeze the self-attention module of the first layer in the mBART encoder and decoder.

4 Experimental Settings

We use the fairseq (Ott et al., 2019) library for all experiments. The final models are selected based on validation likelihood, except for multilingual fine-tuning where we evaluate the models after 10000 training steps. We use beam-search with beam size 5 for decoding, and evaluate all BLEU scores using SacreBLEU (Post, 2018) [1]. We use ISO 693-2 language codes in this work for convenience, and use the same parallel data as Liu et al. (2020), both listed in listed in Table 11 of the Appendix.

We fine-tune frozen BART and an Input Module on bilingual parallel text, feeding the source language into the Input Module. For mBART we feed the source language into the encoder, and use the same hyper-parameters as Liu et al. (2020). When using adapters we use 0.1 dropout in the adapter bottleneck layer (**z** in section 3.3), and a hidden dimension of either 128, or $\lfloor 2/3 \cdot 128 \rceil$ when using a gated linear unit adapter. We use the Adam (Kingma and Ba, 2015) optimizer. Hyper-parameters are listed in Appendix B, and we use the same hyper-parameter search space for frozen and non-frozen models.

4.1 Multilingual MT

We train with a very large effective batch size, training on 32 GPUs with a per-GPU batch size of 4096 tokens, meaning our total batch size is $N \cdot 32 \cdot 4096$ tokens, where N is the number of language pairs. We evaluate our model after 10000 training steps (amounting to $N \cdot 10000$ forwards-backwards passes through the model).

4.2 Vocabulary

BART uses the GPT-2 tokenizer, which uses the BPE (Sennrich et al., 2016) approach (on the level of bytes, not characters). BART could technically take any Unicode string as input, however the BPE is learned on English text. When fine-tuning BART on machine translation we therefore learn a

---

[1] SacreBLEU signature: `BLEU+case.lc+lang.[src-lang]-[tgt-lang]+numrefs.1+smooth.exp+tok.13a+version.1.3.6`

Languages	**Vi-En**†	**Vi-En**	**It-En**	**My-En**	**Ne-En**	**Si-En**	**Cs-En**	**Es-En**	**Pars (m)**
Size	110k	133k	250k	259k	564k	647k	11M	15M	
(1): Freeze decoder	12.1	30.0	36.5	27.4	11.0	13.6	26.6	34.1	407
Freeze encoder	12.0	29.7	36.6	25.2	8.8	12.3	25.6	33.8	457
(2): (1) + adapters	12.2	30.0	36.7	27.7	10.8	14.2	27.4	34.4	410
(2) + ft enc-attn	12.3	30.6	37.0	29.0	11.4	14.9	27.0	35.1	461
(2) + ft self-attn	11.7	30.4	36.1	28.3	10.6	14.3	27.4	34.7	461
(2) + ft last 3 lyrs	12.1	30.6	36.6	28.1	11.5	14.7	27.6	34.9	461
Test (random init)	8.1	23.6	31.7	23.3	7.6	7.2	22.0	29.0	N/A
Test (frozen BART)	-	35.2	38.5	21.0	0.5	7.8	-	-	29
Test (ft all)	14.1	**36.7**	**39.8**	**27.6**	14.1	**14.0**	29.2	**34.5**	610
Test (ft enc-attn)	**14.9**	**36.4**	39.4	**27.9**	**14.6**	**14.1**	**29.8**	**34.4**	461

Table 3: Validation BLEU score (unless stated otherwise) obtained by freezing various parts of the mBART and of adding adapters for $Xx \rightarrow En$. 'ft' refers to fine-tuning, i.e. unfreezing. Vi-En† refers to a new parallel, 'out-of-domain' dataset constructed similarly to the Flores (Guzmán et al., 2019) train sets (see section 5.2). 'Test (frozen BART)' indicates results from English-only BART with the best performing method from Table 2 or Table 1. 'Test (random init)' refers to training models (of various sizes) from scratch on the bitext for that language pair. 'Pars (m)' refers to the number of tunable parameters for each method in millions (note token embeddings are tuned in every method and account for 256m parameters). Bold indicates the best test set score and all scores whose difference from the best is not statistically significant (with p-value less than 0.05). (Statistical significance is computed via bootstrapping (Koehn, 2004).)

	Vi-En	**It-En**	**My-En**	**Ne-En**	**Si-En**
Freeze decoder (don't ft layer-norm)	26.6	35.1	26.6	10.3	13.1
Freeze encoder (don't ft layer-norm)	29.4	36.1	24.1	8.7	12.1

Table 4: Ablation study on improvement from fine-tuning layer-norm. Compare to the 'Freeze decoder' and 'Freeze encoder' methods in the first two rows of Table 3.

new subword vocabulary (using the sentencepiece (Kudo and Richardson, 2018) library) on the source data from the fine-tuning dataset, and use a smaller vocabulary size of 5000, which empirically performs better for low-resource MT (Guzmán et al., 2019; Sennrich and Zhang, 2019). We don't change the mBART tokenizer or vocabulary.

5 Results and Discussion

5.1 Frozen BART

Table 1 shows the effects of various choices we made in fine-tuning BART for MT. *Freezing* is important: we see an 18.4 BLEU point improvement from fine-tuning a frozen BART model compared to fine-tuning an unfrozen BART (both with an Input Module; see section 3.1).

Adding extra flexibility with within-network adapters helps performance, especially when added to the BART *encoder*. It is important to use *learned positional embeddings* at the embedding layer in the Input Module, with an 10.1 BLEU score drop if we use fixed positional embeddings (at the embedding layer). We see consistent gains in Table 1 and Table 2 by adding additional, fixed sinusoidal positional embeddings to the input of every transformer layer of the Input Module (see section 3.2), even when using an unfrozen BART. The BART encoder 'expects' English input, and it may be the Input Module with extra fixed embeddings can better account for the different word order in the input language. In the next section we compare to mBART and baselines.

5.2 Frozen mBART

In Table 3 and Table 5 we list results from freezing various parts of mBART. We get better performance than fine-tuning ('ft all' in Table 3) with our freeze decoder + fine-tune encoder-decoder attention method ('ft enc-attn' in Table 3) on Ne-En and Cs-En for $Xx \rightarrow En$, and mostly similar results to

Languages	**En-Vi**	**En-It**	**En-My**	**En-Ne**	**En-Si**	**En-Cs**	**En-Es**	**Pars (m)**
Freeze decoder	29.7	32.2	35.0	5.8	2.1	17.7	35.4	407
(1): Freeze encoder	30.1	31.5	36.0	5.3	3.7	16.5	35.0	457
(2): (1) + encoder adapters	30.3	32.3	36.9	5.4	4.2	16.6	35.3	461
Test (ft all)	**35.4**	34.0	**36.9**	**7.4**	**3.3**	**18.0**	**34.0**	610
Test (freeze enc. + adapters)	35.0	**34.3**	35.9	6.9	**3.3**	16.7	32.5	461

Table 5: Validation BLEU score (unless stated otherwise) obtained by freezing various parts of the mBART and of adding adapters for for *En* → *Xx*. 'Pars (m)' refers to the number of tunable parameters for each method in millions.

the baseline otherwise.

We believe a benefit to freezing, when fine-tuning on training data from a different domain to test data, will be avoiding specialising the pre-trained model to the fine-tuning train data domain. To test this we constructed a new Vi-En parallel dataset (Vi-En† in Table 3) using the some of the same sources as the Flores (Guzmán et al., 2019) training data (the Si-En and Ne-En training sets used in this work), specifically GNOME/KDE/Ubuntu domain from the OPUS repository[2] and Bible translations from the bible-corpus[3], and use the same test and validation sets as the IWSLT15 Vi-En dataset. By constraining ourselves to this out-of-domain training set we see the largest gains out of the language pairs we considered over the fine-tuning baseline (0.9 BLEU).

We also consider the effect of the size of the fine-tuning dataset. If we constrain the training data to a random subset of 200k training examples from Ro-En (Table 6), the 'ft enc-attn' method outperforms simple fine-tuning. This effect generalises to an mBART variant that was pre-trained on only Ro and En monolingual data (using the same data as Liu et al. (2020)). Further results on Ro-En data are available in the Appendix, Table 10, and show similar trends to Table 3, with fine-tuning encoder-decoder attention the most important.

Table 3 shows the relative performance of frozen BART, frozen mBART and baselines. Fine-tuning mBART gave consistently better results than frozen BART especially for distantly related languages. For Si, Ne and My the performance of frozen BART is roughly on par with a randomly initialised model (or much worse in the case of Ne-En). The parallel data for these languages is often lower quality, and the BART system has to learn about the non-English language from noisy or out-of-domain text (e.g. text from the Ubuntu manual for the En-Ne pair). For Vi and It, we have high quality parallel data, and the frozen BART method is only approximately 1.5 BLEU points behind the best mBART results. We note mBART was trained on more English data than BART, and with different noising function hyper-parameters.

5.3 What Should be Unfrozen?

Layer-Norm We find large benefits to simply fine-tuning the weights and biases of the pre-trained layer-norm weights (recall that after normalisation, the layer-norm module multiplies each hidden dimension by a weight and adds a bias); this was observed in the setting of BERT by Houlsby et al. (2019). This gains e.g. 0.5 BLEU for frozen BART (see Table 1) and an average of 0.8 BLEU across five languages for mBART (see Table 4 compared to Table 3). Since these weights and biases are only $2d$ parameters per layer-norm, where d is the model dimension. This is parameter-efficient, with adding more parameters with 'Adapters' on top of unfrozen layer-norm providing a smaller improvement.

Encoder vs Decoder For the *Xx* → *En* direction (Table 3) we can see that freezing the decoder always performs better than freezing the encoder (except for It-En where they perform roughly the same.) For the *En* → *Xx* direction (Table 5) we see slightly weaker evidence for the opposite trend, with the decoder more useful to fine-tune; but for the high resource languages Es and Cs freezing the decoder works better. There is more English data in mBART pre-training than data in other languages, which may account for better results with a frozen encoder (when English is the source language) or decoder (when English is the target language). Adding flexibility with adapters in the frozen layers

---

[2] http://opus.nlpl.eu/

[3] https://github.com/christos-c/bible-corpus/

Model	**mBART**		**En-Ro mBART**	
Languages (Size)	**Ro-En (608k)**	**Ro-En (200k)**	**Ro-En (608k)**	**Ro-En (200k)**
Test (ft all)	37.8	36.4	38.5	37.7
Test (ft enc-attn)	37.8	36.8	38.1	37.9

Table 6: Validation set BLEU (unless stated otherwise) comparing freezing various parts of mBART and En-Ro mBART (pre-trained only on En and Ro data), fine-tuned on *Ro* → *En* parallel data. 'ft' referes to fine-tuning, i.e. unfreezing. 'Ro-En (200k)' refers to a random subset of the Ro-En training data of size 200k.

Src. Lang.	**Ru**	**Fr**	**De**	**Zh**	**Es**	**Cs**	**Lv**	**Fi**	**Lt**	**Et**	**Hi**	**Si**
Size	32M	29M	28M	25M	15M	11M	4.5M	2.7M	2.1M	1.9M	788k	647k
Finetune all	**33.6**	**39.0**	**33.1**	**20.2**	33.7	**29.9**	**21.1**	**29.0**	**22.8**	**28.6**	**25.4**	**16.9**
Ft enc-attn	**33.4**	38.2	32.6	**20.2**	**34.0**	29.7	**20.8**	**29.1**	**22.7**	**28.3**	25.1	**16.7**

Src. Lang.	**Ro**	**Ne**	**My**	**Ar**	**It**	**Nl**	**Ko**	**Ja**	**Tr**	**Vi**	**Kk**	**Gu**
Size	612k	563k	259k	251k	251k	237k	230k	223k	207k	133k	91k	12k
Finetune all	**37.8**	**20.7**	**31.0**	**37.0**	**39.6**	**43.3**	**25.0**	**18.7**	**24.0**	**37.4**	14.6	**18.7**
Ft enc-attn	**37.9**	**20.8**	30.5	**36.9**	**39.3**	**43.0**	24.2	**18.8**	23.7	**37.5**	**15.0**	**18.3**

Table 7: Test set BLEU score on many-to-one (*Xx* → *En*) multilingual MT with a simple round-robin training schedule. 'Ft enc-attn' refers to fine-tuning the encoder, and fine-tuning the encoder-decoder attention module in every decoder layer, leaving the other decoder sub-modules frozen. The 'Ft enc-attn' model setting uses adapter modules in the decoder to increase flexibility after freezing parameters. Bold indicates the best score and all scores whose difference from the best is not statistically significant (with p-value less than 0.05). For clarity we underline language pairs where the 'Ft enc-attn' method matches or outperforms naive fine-tuning.

improves performance in all languages and directions, except for Ne→En.

We explore more fine-grained unfreezing for the *Xx* → *En* direction (Table 3). We fine-tuned three equally sized subsets of the decoder: the encoder-decoder attention layers (approx. $12 \cdot 4d^2_{\text{BART}}$ parameters), the self-attention layers in the decoder (approx. $12 \cdot 4d^2_{\text{BART}}$ parameters), or the entire last three layers of the decoder (approx. $3 \cdot 16d^2_{\text{BART}}$ parameters). We observe that fine-tuning the encoder-decoder attention performed well (note the last three layers include three encoder-decoder attention layers), with fine-tuning self-attention the least useful. We hypothesize that the pre-training task of mBART (reconstructing noisy monolingual sentences) does not help with teaching the encoder-decoder attention to align source and target text of different languages.

	Tokens per GPU
Finetune all	2304
(1): Freeze decoder	4096
Freeze encoder	3584
(2): (1) + decoder adapters	4096
(2) + ft enc-attn	3328

Table 8: Maximum number of tokens that would fit on one NVIDIA Volta GPU when fine-tuning mBART on the En-Vi training set. We evaluated batch sizes in increments of 256 tokens.

5.4 Memory Cost

Freezing parameters means we no longer need to allocate memory to storing their gradients. We will obtain additional memory savings when using an optimizer that stores various other quantities (i.e. the Adam optimizer stores running averages of the first and second moments of gradients.). The memory savings allow for roughly 45-75% larger batches for the methods we consider in this work (see Table 8 for our mBART methods), but for larger pre-trained models the proportion of GPU memory freed up by freezing will increase. At inference time we no longer require gradients and we have the same memory cost.

5.5 Multilingual Fine-tuning of mBART

We explore freezing parts of the mBART model when fine-tuning on a challenging multilingual MT task. Table 7 lists results from a naive fine-tuning baseline, and results from freezing most of the decoder but unfreezing the encoder-decoder attention (when freezing we use GLU adapters in the decoder, see section 3.3). Freezing parameters hurts performance on some language pairs, and since freezing removes flexibility from the model and we have to adapt to 25 different directions this is perhaps not surprising. The language pairs where we match or improve on the baseline are Zh, Es, Fi, Ne, Ja, Vi and Kk. These are mostly (five out of seven) non-European languages, and distantly related to En. However since most of these results are not statistically significant further study is needed to verify this. Note we see a clear benefit over bilingual fine-tuning for some language pairs (e.g. compare our best Ne result from Table 3, 14.6 BLEU vs. 20.8 BLEU for multilingual fine-tuning). We leave to future work a more thorough investigation of the multilingual MT setting.

6 Conclusion

We recommend: For a language with high quality parallel data but without a pre-trained model trained on monolingual data from that language, using a frozen (English-only) BART model with additional parameters at the source side (the 'input module') improves performance over a randomly initialised baseline. For this approach it is important to freeze the pre-trained model. We also give the model both learned positional embeddings at the embedding layer, and fixed sinusoidal positional embeddings at each layer of the input module.

For a multilingual pre-trained model, we found performance improvements on some (mostly distantly related) languages for multilingual many-to-one fine-tuning. For bilingual $En \rightarrow Xx$ fine-tuning we did not see any improvement, although the performance drops are small, and by freezing parameters we need less memory at training time compared to fine-tuning. For $Xx \rightarrow En$ bilingual fine-tuning it is important to unfreeze the encoder-decoder attention, and keep the rest of the decoder frozen. This can improve on simple fine-tuning, especially for distantly-related language pairs or those with out-of-domain training data.

We recommend fine-tuning layer-norm parameters as a parameter-efficient complement to adapter layers. For our mBART experiments we found it was necessary to fine-tune the token embeddings, which correspond to a large number of parameters, and future work could remove this cost by working out a subset of the vocabulary to fine-tune, or another method.

Acknowledgments

We'd like to thank James Cross, Mike Lewis, Naman Goyal, Jiatao Gu, Iain Murray, Yuqing Tang and Luke Zettlemoyer for useful discussion. We also thank our colleagues at FAIR and FAIAR for valuable feedback.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *CoRR*, abs/1612.08083.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *North American Association for Computational Linguistics (NAACL)*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *NAACL*.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. 2018. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 344–354, New Orleans, Louisiana. Association for Computational Linguistics.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6097–6110, Hong Kong, China. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799, Long Beach, California, USA. PMLR.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 379–386, Varna, Bulgaria. INCOMA Ltd.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 875–880, Brussels, Belgium. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. FAIRSEQ: A fast, extensible toolkit for sequence modeling. In *North American Association for Computational Linguistics (NAACL): System Demonstrations*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *North American Association for Computational Linguistics (NAACL)*.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *CoRR*, abs/1903.05987.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Prajit Ramachandran, Peter J Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391.

Devendra Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 261–271, Belgium, Brussels. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Association for Computational Linguistics (ACL)*, pages 1715–1725.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. 2013. Dirt cheap web-scale parallel text from the common crawl. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1374–1383, Sofia, Bulgaria. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning (ICML)*.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995, Long Beach, California, USA. PMLR.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Xu Tan, Jiale Chen, Di He, Yingce Xia, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 963–973, Hong Kong, China. Association for Computational Linguistics.

Brian Thompson, Huda Khayrallah, Antonios Anastasopoulos, Arya D. McCarthy, Kevin Duh, Rebecca Marvin, Paul McNamee, Jeremy Gwinnup, Tim Anderson, and Philipp Koehn. 2018. Freezing subnetworks to analyze domain adaptation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 124–132, Belgium, Brussels. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2214–2218, Istanbul, Turkey. European Languages Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Fernanda Viégas, Greg Corrado, Jeffrey Dean, Macduff Hughes, Martin Wattenberg, Maxim Krikun, Melvin Johnson, Mike Schuster, Nikhil Thorat, Quoc V Le, et al. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation.

Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Yong Yu, Weinan Zhang, and Lei Li. 2019a. Towards making the most of bert in neural machine translation. *arXiv preprint arXiv:1908.05672*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823*.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

A Additional Ablation Study

In Table 9 we reproduce Table 4 of the main paper with more context to study the effect of unfreezing layer-norm parameters when fine-tuning mBART. Across all language pairs we see improvements from fine-tuning layer norm parameters over not fine-tuning them, and additional, smaller, improvements from adding adapters, indicating both forms of adding flexibility are useful. In Table 10 we present additional results on the Ro-En pre-trained model (see section 3.2 of the main body).

B Fine-tuning Hyper-parameters

For all experiments with bilingual datasets we use a batch size of 2048×16 tokens, i.e. 2048 tokens per GPU and 16 GPUs (we investigate larger batch sizes for frozen models only to test GPU memory usage, and do not evaluate models trained with larger batch sizes). Ranking of hyper-parameters was done by validation set BLEU score.

Frozen BART We train with 0.3 dropout for the frozen BART parameters, and 0.2 dropout for the Input Module parameters, 0.1 label smoothing, 0.2 dropout for the self-attention scores in the Input Module, 5000 warm-up steps, and $7\text{e}{-4}$ maximum learning rate. We performed a grid search over learning rates in $\{7\text{e}{-4}, 5\text{e}{-4}, 3\text{e}{-4}\}$, dropout for Input Module parameters in $\{0.2, 0.1\}$, and dropout for self-attention scores in $\{0.2, 0.1\}$. We train for a maximum of 50K training updates for all low and medium resource pairs and 100K for high resource pairs (which takes roughly 8 hours and 16 hours respectively).

Frozen mBART We train with 0.3 dropout, 0.2 label smoothing, 2500 warm-up steps, and $3\text{e}{-5}$ maximum learning rate. We did not search over hyper-parameters, simply re-using those of Liu et al. (2020). Despite the adapter parameters being randomly initialised, the small learning rate did not affect performance (we performed a small sweep of larger learning rates and found only marginal gains, and so kept the same settings for simplicity). We use a maximum of 40K training updates for all low and medium resource pairs and 100K for high resource pairs (Es and Cs in our case), this takes roughly 12 hours and 30 hours respectively.

Multi-lingual MT We train with 0.3 dropout, 0.1 dropout for self-attention scores, 4000 warm-up steps, and $1\text{e}{-4}$ maximum learning rate.

Out-of-domain Vi-En Baseline To train a randomly initialised baseline for the out-of-domain Vi-En data (Vi-En† in Table 3 of the main body) we used the same model architecture and training settings as those of Guzmán et al. (2019) use for training MT systems on similar data (but with Si or Ne source language). Specifically a seq2seq transformer with 5 encoder and decoder layers, hidden dimension 512. shared embeddings between the input and softmax layers, and strong regularisation (e.g. 0.4 dropout on hidden states, 0.2 dropout on attention scores, 0.2 label smoothing). We learn a BPE vocabulary (joint across source and target data) of size 5000 on the training data. For full details of hyper-parameters we refer the reader to Guzmán et al. (2019) and the associated GitHub repository[4].

C Pre-training Languages

We reproduce in Table 11 the details from Liu et al. (2020) of the size of each pre-training language corpus for mBART.

---

[4] https://github.com/facebookresearch/flores

	Vi-En	**It-En**	**My-En**	**Ne-En**	**Si-En**
Freeze decoder	26.6	35.1	26.6	10.3	13.1
Freeze encoder	29.4	36.1	24.1	8.7	12.1
(1): Freeze decoder + ft layer norm	30.0	36.5	27.4	11.0	13.6
Freeze encoder + ft layer norm	29.7	36.6	25.2	8.8	12.3
(1) + decoder adapters	30.0	36.7	27.2	10.8	14.2

Table 9: Validation BLEU score (unless stated otherwise) obtained by fine-tuning layer-norm parameters and of adding adapters for mBART, for *Xx* → *En*. 'ft' refers to fine-tuning, i.e. unfreezing. Note we are simply reproducing rows from Table 3 and Table 4 of the main paper for ease of comparison.

	mBART		**En-Ro mBART**	
	Ro-En (608k)	**Ro-En (200k)**	**Ro-En (608k)**	**Ro-En (200k)**
(1): Freeze decoder	38.8	37.9	40.4	39.9
Freeze encoder	39.1	38.3	40.0	39.2
(2): (1) + decoder adapters	39.3	38.0	40.6	40.0
(1) + ft enc-attn	39.8	39.0	40.5	40.5
(1) + ft self-attn	39.6	38.3	40.4	40.1
(1) + ft last 3 lyrs	39.6	38.6	40.5	40.3
Test (ft enc-dec)	37.8	36.8	38.1	37.9
Test (ft all)	37.8	36.4	38.5	37.7

Table 10: Validation set BLEU (unless stated otherwise) comparing freezing various parts of mBART and En-Ro mBART (pre-trained only on En and Ro data rather than 25 languages), fine-tuned on *Ro* → *En* parallel data. 'ft' refers to fine-tuning, i.e. unfreezing. 'Ro-En (200k)' refers to a random subset of the Ro-En training data of size 200k.

Code	**Language**	**Tokens(M)**	**Size(GB)**	**Parallel data source**
En	English	55608	300.8	
Ru	Russian	23408	278.0	WMT19
Vi	Vietnamese	24757	137.3	IWSLT15
Ja	Japanese	530 (*)	69.3	IWSLT17
De	German	10297	66.6	WMT19
Ro	Romanian	10354	61.4	WMT16
Fr	French	9780	56.8	WMT19
Fi	Finnish	6730	54.3	WMT17
Ko	Korean	5644	54.2	IWSLT17
Es	Spanish	9374	53.3	WMT19
Zh	Chinese (Sim)	259 (*)	46.9	WMT19
It	Italian	4983	30.2	IWSLT17
Nl	Dutch	5025	29.3	IWSLT17
Ar	Arabic	2869	28.0	IWSLT17
Tr	Turkish	2736	20.9	IWSLT17
Hi	Hindi	1715	20.2	ITTB
Cs	Czech	2498	16.3	WMT19
Lt	Lithuanian	1835	13.7	WMT19
Lv	Latvian	1198	8.8	WMT17
Kk	Kazakh	476	6.4	WMT19
Et	Estonian	843	6.1	WMT18
Ne	Nepali	237	3.8	FLoRes
Si	Sinhala	243	3.6	FLoRes
Gu	Gujarati	140	1.9	WMT19
My	Burmese	56	1.6	WAT19

Table 11: **Languages and Statistics of the CC25 Corpus.** A list of the 25 languages used in mBART pre-training ranked with monolingual corpus size. (*) The Chinese and Japanese corpora are not segmented, so the token counts here are sentence counts.

From characters to words: the turning point of BPE merges

Ximena Gutierrez-Vasques[1] **Christian Bentz**[2] **Olga Sozinova**[1] **Tanja Samardžić**[1]
[1]URPP Language and Space, University of Zürich
[2]Department of General Linguistics, University of Tübingen
`{ximena.gutierrezvasques, olga.sozinova, tanja.samardzic}@uzh.ch`
`chris@christianbentz.de`

Abstract

The distributions of orthographic word types are very different across languages due to typological characteristics, different writing traditions, and other factors. The wide range of cross-linguistic diversity is still a major challenge for NLP, and for the study of language more generally. We use BPE and information-theoretic measures to investigate if distributions become more similar under specific levels of subword tokenization. We perform a cross-linguistic comparison, following incremental BPE merges (we go from characters to words) for 47 diverse languages. We show that text entropy values (a feature of probability distributions) converge at specific subword levels: relatively few BPE merges (around 200 for our corpus) lead to the most similar distributions across languages. Additionally, we analyze the interaction between subword and word-level distributions and show that our findings can be interpreted in light of the ongoing discussion about different morphological complexity types.[1]

1 Introduction

In NLP, one of the predominant methods for obtaining subword units is Byte-Pair Encoding (BPE). These subwords have proven to be useful for improving several NLP tasks, most likely because they capture morphological patterns to some extent (and also phonological and orthographic ones).

BPE is based on a compression algorithm which finds frequently ocurring patterns in a text by means of incrementally merging adjacent symbols into longer strings (Gage, 1994; Sennrich et al., 2015). The granularity of the subword units is controlled by the number of merge operations applied to the text (few merges lead to a text tokenization closer to the character level, while more merges lead to a tokenization closer to the word level). Usually the number of BPE merges is chosen arbitrarily depending on the application.

It is rarely analyzed how the distribution of these subwords changes across different merge operations. Our goal is to investigate if languages get 'closer' in terms of their subword distributions under specific levels of tokenization. We quantify this cross-linguistic variation using information-theoretic measures.

Information theory provides a useful tool for exploring variation, and for quantifying the predictability/organization of patterns, e.g., in morphological systems (Ackerman and Malouf, 2013). We measure Shannon entropy and redundancy over varied subword tokenizatons of texts obtained with BPE. At each incremental merge, we compare the values across 47 typologically diverse languages.

Cross-linguistic corpora are widely used as a means of quantifying linguistic diversity. For instance, the range of entropy values measured over word-level types varies greatly across languages. This is a reflection of the diversity of morphological systems. However, we show that this cross-linguistic variation is not so pronounced at the subword level. Namely, a convergence of entropy values across languages is achieved at a relatively low number of merge operations. The entropy of subword distributions grows quickly before this turning point, while the growth is considerably slower after it.

Furthermore, in this turning point, the subword distributions start to correlate with the ones observed at the word-level. We interpret this change of trend in light of previous findings regarding the difference between subword and word-level complexity: a language that is complex at the word level (rich inflectional morphology), is not necessarily complex at a more atomic subword level (predictable subword patterns).

---

[1] Data and code available at `https://github.com/ximenina/theturningpoint`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3454–3468
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2 Background

2.1 Byte-pair Encoding (BPE)

Originally, BPE is a data compression technique based on replacing the most common pair of consecutive bytes with a new symbol (Gage, 1994). This is currently one of the predominant approaches for subword tokenization (or morphological segmentation). It is widely used to improve tasks like machine translation or language modeling (Sennrich et al., 2015; Provilkov et al., 2020). Another popular method is provided by Morfessor (Smit et al., 2014).

When BPE is applied to text, each iteration merges two adjacent symbols. The main hyperparameter of BPE is the number of merge operations applied to the data, which controls the granularity of the subword units. In NLP, this hyperparameter is usually chosen empirically, e.g., based on the dataset size or on the task, regardless of the typological features of a specific language.

2.2 Text and information theory

Bentz et al., (2016) distinguish between corpus-based and paradigm-based approaches for quantifying morphological complexity. While the former approaches measure morphological productivity directly on raw text corpora, the latter make use of higher level language descriptions, i.e., grammars, and inflectional paradigms.

In corpus-based approaches, a text is usually regarded as a sequence of symbols. Each symbol is generated with a certain probability, and hence carries a certain information content (Juola, 1998; Ehret and Szmrecsanyi, 2016a; Ehret, 2016b; Koplenig et al., 2017; Bentz et al., 2017). The higher the probability of a symbol, the lower its information content. Against this backdrop, the average information content of a text can be estimated by Shannon entropy, and approximated with type-token-ratios (TTR). For instance, if we consider orthographic words as symbols, languages with a greater diversity of word types will have higher entropy (word types are less predictable, due to, e.g., richer morphology). In fact, such corpus-based measures have been shown to be correlated also with paradigm-based approaches that quantify morphosyntactic distinctions based on grammars (Bentz et al., 2016; Kirov et al., 2017).

The complexity of the morphological system of a language is not only related to the diversity of word types that can be produced, but also to the way in which subwords are organized within them.

On the corpus-based side, there are some studies which have focused on the predictability of internal word structure. For instance, there are cross-linguistic accounts illustrating the trade-off between the size of syllables and the size of words: languages with structurally simple and short syllables need more syllables for encoding the same content (Fenk-Oczlon and Fenk, 1999; Coupé et al., 2019). Another line of research proposes to quantify the amount of word-internal information by comparing the (character level) entropy of the original text with a version where the regularities within orthographic words have been masked (Juola, 1998; Ehret and Szmrecsanyi, 2016a; Ehret, 2016b; Koplenig et al., 2017).

Most recently, morphological complexity has been approached through the lense of neural language models, and their learning of subword structure (Vania and Lopez, 2017; Mielke et al., 2019). Gutierrez-Vasques and Mijangos (2020) propose a measure reflecting the predictability of the internal structure of words. It relies on the entropy rate of a neural language model that is trained to predict sequences of character n-grams within a word. We here compare the results of this latest neural network approach with the entropy of subword units based on BPE.

3 Data and methods

Our general proposal comprises calculating several measures over varied subword tokenizatons of texts obtained with BPE. In each consecutive tokenization, we hence regard a different set of strings of characters as symbols of our "alphabet". In merge 0, a text is a sequence of single UTF-8 characters; in the last merges, a text is a sequence closer to orthographic word types (i.e. original tokenization given by white spaces and punctuation). At each incremental step, we compare the values across parallel corpora in 47 typologically diverse languages.

3.1 Parallel corpus

Using parallel corpora facilitates meaningful comparisons across languages, as seen in cross-linguistic studies on morphological typology, lexical typology, and word order typology (Cysouw and Wälchli, 2007; Wälchli and Cysouw, 2012; Östling, 2015; Kelih, 2010; Mayer et al., 2014). In fact, the idea to compare language complexity through parallel corpora can be traced back to

Greenberg (1960).

In this work, we use a publicly available parallel corpus for 47 languages that was extracted from the Parallel Bible Corpus (PBC) (Mayer and Cysouw, 2014). This specific dataset[2] contains 115 preprocessed parallel verses per language consistently coded in UTF-8. The set of 47 languages is a subset of the WALS 100-language sample, which aims to maximize both genealogical and areal diversity. See the list of languages, their ISO639-3 code and linguistic families in Appendix A.

3.2 Scripts and Writing Systems

The respective texts are written in different scripts (Arabic, Cyrillic, Devanagari, Georgian (Mkhedruli), Korean Hangul, Latin, Modern Greek, Myanmar (Burmese), Thai), and reflect different writing systems (abugida, abjad, alphabet, syllabary). Since BPE starts to operate at the level of UTF-8 characters, the ideosyncrasies of encodings are relevant for our analyses. For instance, the word *beginning* in English consists of 9 UTF-8 character tokens and 5 types ('b','e','g','i','n'), while the corresponding word written in Korean Hangul 시작 (transliterated as 'sijak') consists of two syllable blocks, namely 시 ('si') and 작 ('jak'). It is these syllable blocks – rather than individual letters of the Korean alphabet – which are represented as UTF-8 characters. Thus, while English texts typically contain 26 UTF-8 character types (bare punctuation), Korean texts might display hundreds and thousands. A similar proliferation of UTF-8 characters is found in texts written with Abugidas (e.g. Hindi, Thai, Burmese), or in latinized scripts with many special characters and diacritics (e.g. Vietnamese).

3.3 BPE merge operations

The BPE algorithm starts by splitting words into a sequence of characters. We can think of this as characters separated by white spaces. In the first operation, the algorithm merges the most frequent pair of consecutive characters within the corpus, e.g., ('e','d') $\rightarrow$ ('ed'), thus creating a new symbol that is added to the vocabulary. In each of the following operations, the algorithm calculates the co-occurrence frequency of pairs of all the current consecutive symbols and it merges again the most frequent pair.

When the algorithm merges a frequent pair of symbols, it automatically removes many of the white spaces in the text (this is one aspect of how BPE achieves text compression). As more merges are applied, longer symbols (in terms of number of characters) are obtained – we are getting closer to the word level. The algorithm stops when a pre-specified number of merge operations has been reached, or when it cannot find a pair of consecutive symbols with frequency greater than 1.

A worked out toy example can be found in Table 1. Note that symbols occurring at the end of a word are considered different from the ones that occur at any other position. The symbols that are merged in BPE are hence character sequences of variable sizes. These can be interpreted as subword units. This is why BPE is usually seen as a morphological segmentation technique in NLP.

We applied an existing BPE implementation[3] to the texts of the parallel corpus. For each language, we obtain many different segmented versions of the text depending on the number of merges applied. In specific, we go from merge 0 to merge 10K.

We traverse the range of merge operations by using different step sizes. We simply do this to ease the computational load. Moreover, the different trends that we observe are already stable by merge 350.

- Fine-grained merges:
  0 to 350 (step size: 1)
- Coarse-grained merges:
  350 to 5K merges (step size: 50)
  5k to 10K merges (step size: 1K)

3.4 Information-theoretic measures

Once the texts are segmented, we apply two different information-theoretic measures. Both take as input a text T with a vocabulary of types $V = \{t_1, t_2, ..., t_V\}$ of size $|V|$. At the word level, these types correspond to words (strings separated by spaces). Analogously, at the subword level, the types are the subword units obtained at a specific number of merge operations. We distinguish between the subword units that are at the end of a word and the rest of them, e.g. *-ed* and *-ed-* are considered different types in the vocabulary.[4]

We use **entropy** as a measure of the average information content of types in a text. We can

---

[2] Dataset from the Interactive Workshop on Measuring Language Complexity (IWMLC 2019)

[3] `https://github.com/rsennrich/subword-nmt`

[4] We follow this distinction since it is made by the BPE implementation that we use.

Merge	Text Version	Alphabet (Vocabulary of Symbols)
0	g-o-d c-r-e-a-t-e-d t-h-e h-e-a-v-e-n a-n-d g-o-d d-i-v-i-d-e-d t-h-e l-i-g-h-t	a-, c-, d, d-, e, e-, g-, h-, i-, l-, n, n-, o-, r-, t, t-, v-
1	g-o-d c-r-e-a-t-e-d **th**-e h-e-a-v-e-n a-n-d g-o-d d-i-v-i-d-e-d **th**-e l-i-g-h-t	a-, c-, d, d-, e, e-, g-, h-, i-, l-, n, n-, o-, r-, t, t-, **th-**, v-
2	g-o-d c-r-e-a-t-e-d **the** h-e-a-v-e-n a-n-d g-o-d d-i-v-i-d-e-d **the** l-i-g-h-t	a-, c-, d, d-, e-, g-, h-, i-, l-, n, n-, o-, r-, t, t-, **the**, v-
3	g-**od** c-r-e-a-t-e-d the h-e-a-v-e-n a-n-d g-**od** d-i-v-i-d-e-d the l-i-g-h-t	a-, c-, d, d-, e-, g-, h-, i-, l-, n, n-, **od**, r-, t, t-, the, v-
4	**god** c-r-e-a-t-e-d the h-e-a-v-e-n a-n-d **god** d-i-v-i-d-e-d the l-i-g-h-t	a-, c-, d, d-, e-, g-, **god**, h-, i-, l-, n, n-, r-, t, t-, the, v-
5	god c-r-e-a-t-**ed** the h-e-a-v-e-n a-n-d god d-i-v-i-d-**ed** the l-i-g-h-t	a-, c-, d, d-, e-, **ed**, g-, god, h-, i-, l-, n, n-, r-, t, t-, the, v-
6	god c-r-**ea**-t-ed the h-**ea**-v-e-n a-n-d god d-i-v-i-d-ed the l-i-g-h-t	a-, c-, d, d-, e-, **ea-**, ed, g-, god, h-, i-, l-, n, n-, r-, t, t-, the, v-

Table 1: Example of BPE merge operations. Original text: *God created the heaven [...] and God divided the light [...]*

calculate the entropy as follows (Shannon, 1948):

$$H(T) = -\sum_{i=1}^{V} p(t_i) \log_2 p(t_i) \tag{1}$$

Where the probability of a type $p(t)$ is estimated using the so-called maximum likelihood method (i.e. its relative frequency in the text). Higher values of entropy indicate higher complexity (less predictability). We take this as our main measure of text-based morphological complexity through merges.

We also use **redundancy,** a measure that is related to entropy. The entropy of a source of data is maximum when the symbols comprising a message can be chosen freely and they are equiprobable (maximum uncertainty). The redundancy, as defined here, quantifies how close the empirically estimated entropy $H(T)$ is to the maximum value it can take, assuming that we utilize the same alphabet (or types in our case). It can be defined as follows (Partridge, 1981; Karmeshu, 2003):

$$R(T) = 1 - \frac{H(T)}{max\{H(T)\}} = 1 - \frac{H(T)}{\log_2 |V|} \tag{2}$$

Where $H(T)$ is the entropy of a text, calculated as in (1). The maximum entropy can be calculated as $max\{H(T)\} = log_2|V|$, i.e, the entropy when the probability distribution of types is uniform $p(t_i) = \frac{1}{|V|}$. The values of R range from 0 to 1. Values closer to 1 indicate higher redundancy.

3.5 Spearman's rank correlation

We use correlations for exploring the connection between values yielded by the complexity measures described above. We rank languages according to these measures. In particular, we use Spearman's rank correlation, which tests for a correlation between the rankings of two variables (monotonic relationships, not necessarily linear).

We apply Spearman's rank correlation to the following variables:

1. Final merge (the number of merge operations needed to reach the final step of the BPE algorithm);
2. Average word length (at the word level);
3. Size of the vocabulary of characters (of the original texts);
4. Entropy and redundancy measured over the texts at different merges;
5. Two external morphological complexity measures (based on unigrams and trigrams of characters).

Since we have many variables, and hence pairwise correlations, we apply the Bonferroni correction on p-values. We select the correlations that are still significant after correcting for multiple testing, see Appendix C.

4 Results

4.1 From characters to words: the turning point

Figure 1 shows the entropy of languages at different merge operation stages: merge 0, merge 30, merge 200, and the word level. We choose these specific merges for the sake of illustration. In particular, merge 200 is representative of the turning point of several trends.

At the very first merges, texts are closer to character level tokenizations, i.e., small subword units. The initial point is merge 0 (roughly corresponding to the character level). Here, the texts' entropies range between $4.01 - 7.77$ bits. We notice that languages with a larger inventories of UTF-8 characters start with higher entropy values. For instance,

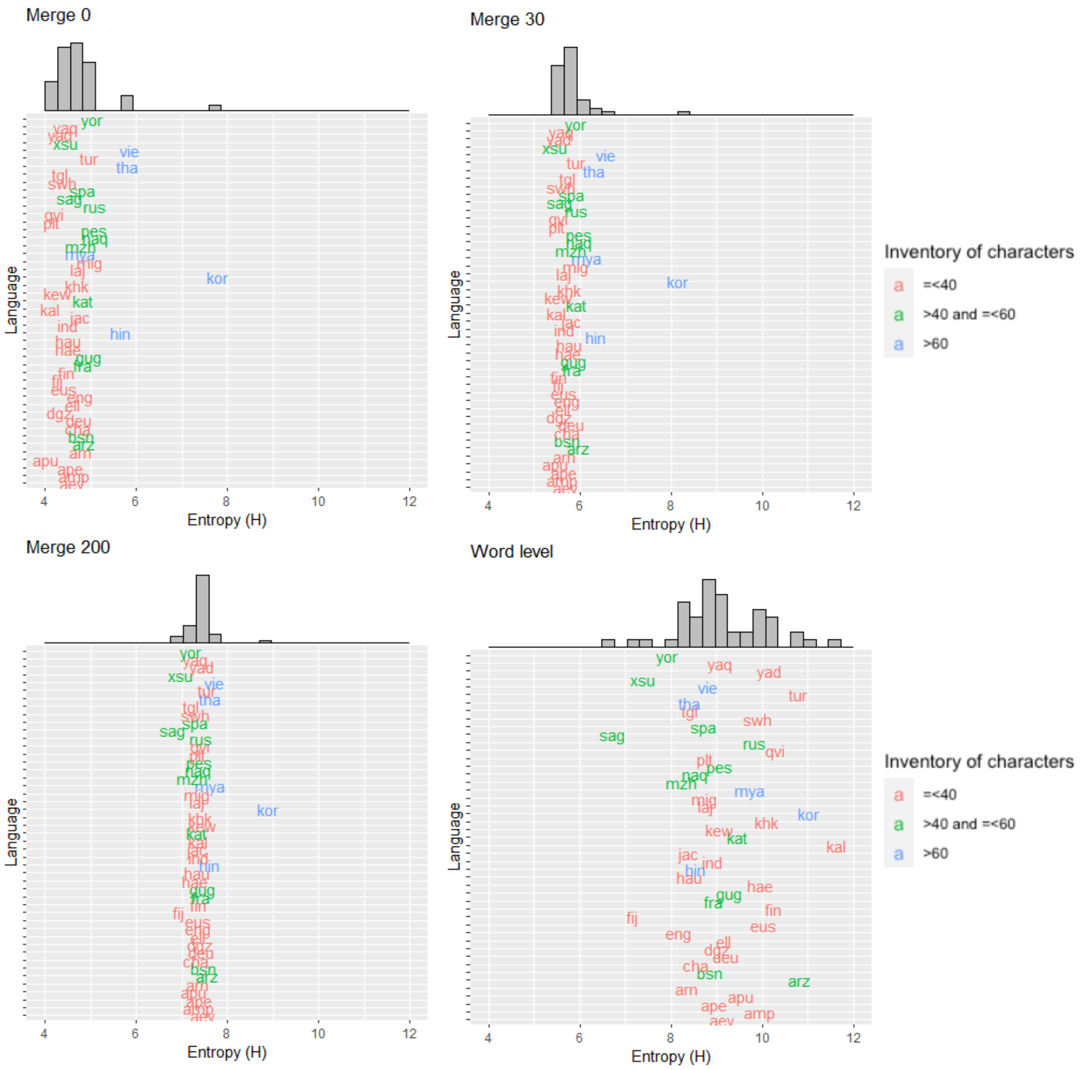


Figure 1: Entropies of languages at different merge operations. Histograms of the distributions are shown above the panels. The sizes of the original UTF-8 character sets are indicated by colors.

Korean *(kor)* is an outlier due to its alpha-syllabary writing system (Section 3.2).

In the subsequent merges, all languages start increasing their entropy. However, the values also start to become less dispersed across languages. In fact, we can see that, in merge 200, the majority of languages are centered around 7.3 bits. This means that, at this merge, the frequency distributions of subwords are similar across languages. Moreover, the size of the initial inventory of characters seems not to affect the texts' entropy at these later merges.

As BPE approaches tokenizations which are closer to orthographic words, entropy values start to disperse again. If we measure entropies over the original texts (without any subword tokenization), we can see that the cross-linguistic variation is considerably wider than the one obtained when the texts are represented by subwords (fewer merges).

This trend is also observable in Fig. 2, where the standard deviation (σ) of entropy across languages is shown as a function of the number of merges. A minimum is reached at merge 200; in fact, between merges 190-240 there is practically no variation of σ. This means that, around this number of merges, the entropies of subword distributions' across languages are closest to one another. After 240 merges, the values start to slowly disperse again and keep dispersing up to the final merges.

Moreover, if we rank languages by their entropy, the rankings obtained before merge 200 are not correlated with the ones observed in later merges (see Section 4.3). From this point onward, the rankings start to gradually correlate with the one observed at the word-level. For instance, at merge 350 (a relatively low number of merges) the vocabularies still contain many short subword units, and the variance

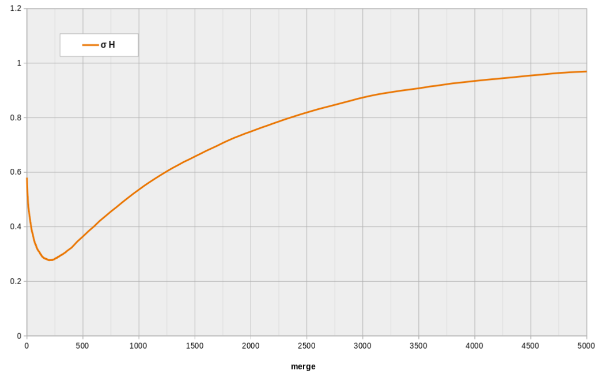


Figure 2: Standard deviations (σ) of entropies across languages per each BPE merge operation.

of entropies across languages is still close to the minimum. Despite of this, the language rankings obtained at this merge are already similar to the rankings observed at the word level.

For some languages, the entropy at the subword level (fewer merges) is systematically different from that at the word level (higher number of merges). For instance, Kalaallisut (*kal*), typically seen as polysynthetic, starts with low H in the first merges – it ranks almost last. However, at merge 200, it is rather in the middle range, and it further increases in entropy with subsequent merges, to the point where it ranks highest of all languages, namely at the word level (Fig. 1).

As we discussed earlier, the entropy (H) has its minimum at merge 0. After each merge operation, H increases. The first merges cause the most drastic changes in the entropy. After the first hundreds of merges, the entropy increases more slowly, i.e., each merge does not cause a big increment of the text entropy anymore. In contrast, redundancy starts decreasing since the first merges. R reaches a minimum after a certain number of merges for all languages (297 merges on average), and then it starts increasing again as the word level is approached. Figure 3 shows an example of the entropy (H) and redundancy (R) across merges for the French text. Appendix B contains the entropy and redundancy curves for all languages.

The first operations merge very frequently adjacent symbols, which impacts the subword distributions of the texts. This is the reason why redundancy and entropy are changing quickly for the early merges. These first merges find the most frequent and productive patterns, e.g., inflectional markers (‘-ed’ and ‘-ing’), and orthographic practices for representing sounds (e.g. ‘th’) in English. When highly recurrent patterns get merged, the redundancy of the texts is reduced. We can think of this in terms of skewed distributions. At merge 0, the vocabulary’s initial distribution of elements is skewed (few UTF-8 characters, high frequencies). When BPE starts merging the most salient patterns, the distribution of subwords gets closer to a uniform distribution (more symbols, lower frequencies), and a minimum of redundancy is reached. After this, the merge operations lead again to skewed distributions (redundancy grows again). However, these latter distributions across languages are correlated to the ones observed at the word level and not to the first merges’ skewed distributions.

Interestingly, the number of merges at which entropies start to grow slower, and redundancies reach a minimum, are in the same range of merges in which languages start to change their trends between subword and word level, ca. 200-300 merges. This is also the turning point in which the cross-linguistic standard deviation of entropies reaches a minimum.

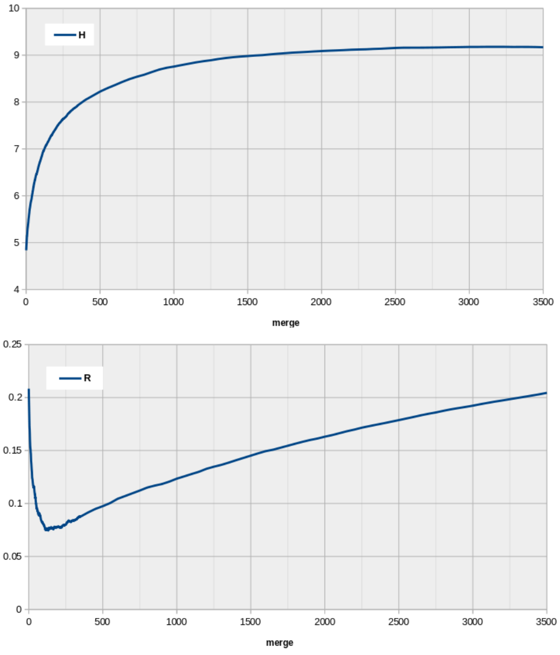


Figure 3: H and R across BPE merges for French (fra).

4.2 Max. number of merges per language

Not all languages require the same number of operations to reach the point where no pair of subwords can be merged anymore. In our corpus, languages needed between 1.1K and 7K operations for reaching this final merge (Figure 4).

We can see that languages with lower values

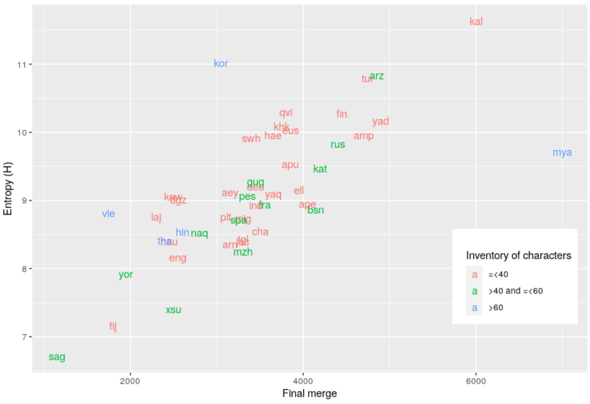


Figure 4: Maximum number of BPE merges for each language (x-axis) against word level entropy (y-axis).

of word entropy require fewer merges. This is the case for languages with isolating tendencies: Sango (*sag*), Vietnamese (*vie*), Fijian (*fij*), Yoruba (*yor*). The Indo-European language that requires the least merges is English (*eng*). These are also the languages that reach the entropy plateau faster, i.e., the first merges capture very productive recurrences, but after a relatively small number of operations, the text entropy does not change that much anymore.

On the other hand, languages with richer morphological processes (polysynthetic, agglutinative, or template morphology) require more merges to reach this final point. Namely, Kalaallisut (*kal*), Burmese (*mya*), Yagua (*yad*), Egyptian Arabic (*arz*), Turkish (*tur*), Alamblak (*amp*), Finnish (*fin*), require the most merges. Note that Burmese (*mya*) is not generally considered a morphologically complex language; it has very long orthographic words on average. This is related to the Burmese script, which uses white spaces differently from other scripts (therefore, it has very long character string sequences).

Notice that despite this outlier, the figure illustrates that characters' inventory size is not strongly influencing the final number of merge operations (Fig. 4). A language that has a small character inventory to start with can still require many merge operations to reach the final merge, as exemplified by Kalaallisut (kal), Turkish (tur), Yagua (yad), and Finnish (fin).

4.3 Correlations between measures

In order to investigate relationships between measures, a correlation matrix is shown in Figure 5 with the variables explained in Section 3.5. We only include correlations for the entropy and redundancy taken at merges 0, 30, 200, 350, and the word level in this matrix. See Appendix C for the complete correlation matrix, which includes a wider range of merges.

The final merge is strongly correlated with the entropy at the word level and the average word length of a language. This is expected since languages with higher entropy/TTR (at the word level) tend to have longer words because they encode more morphosyntactic distinctions within a word. Therefore, under this conceptualization of complexity, complex languages will require more merges during BPE encoding. We did not find any strong correlations between word length and any of the early merges' measures.

The entropy of the texts on the first operations (from merge 0 to merge 100) showed no strong correlation with the entropy at the word level, either positive or negative ($\rho < \pm 0.14$). Therefore, there is not a general trade-off between subword and word level complexities. Some languages display such a trade-off, e.g., being very complex at the word level, while having low complexity on the first merges. However, others are more stable across merges.

Interestingly, it seems that after the 'turning point', the correlation between subword tokenizations and the word level starts to be more prominent. At merge 200 the correlation is $\rho = 0.47$, this gradually increases, e.g., by merge 350, there is already a strong correlation with the word level ($\rho = 0.72$). As we saw in Section 4.1, the rankings obtained at the first merges (somewhere below 200) differ from the trend observed at the subsequent merges, .i.e., after merge 200 the complexity rankings of languages start to be more similar to the one observed at the word level.

Regarding redundancy, there is a strong trade-off between entropy and redundancy at the word level ($\rho = 0.72$). This is understandable since a language with high entropy at the word level will have a wide diversity of word forms, few repetitions, hence, less possibility of compression (low redundancy). However, entropy and redundancy are not always correlated. As shown in Fig. 3 (and Appendix B), the entropy tends to grow through merges, while redundancy first decreases and then grows again. In fact, cross-linguistically, we did not find a strong correlation between H and R during the early merges. By merge 200, H and R

already start to be negatively correlated. This trade-off is maintained in further operations and at the word level.

Even though R and H are not correlating on the first merges, they show similar behavior, i.e., both of them seem to follow a trend on the first merges, not correlated with their respective values at the word level, but then this trend changes (around the turning point).

We can also see that the size of a language's character inventory is correlated with the entropy on the first merges (around 0.6). While for redundancy, this correlation is not significant.

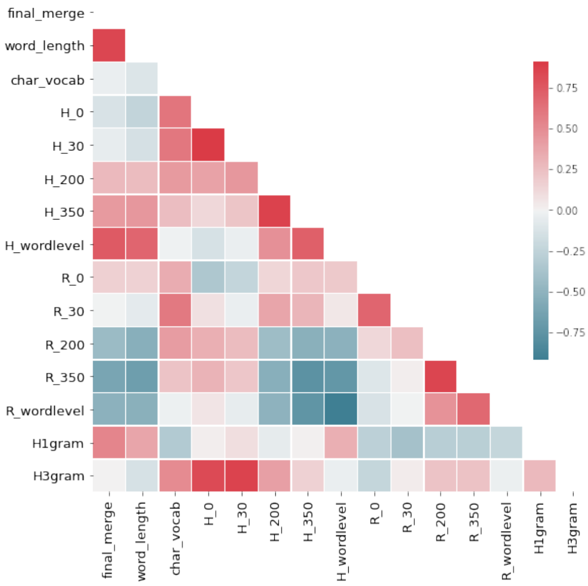


Figure 5: Spearman's rank correlation (subset of variables)

4.3.1 Predictability of sequences

Entropy is reflecting the degree of organization or predictability of subword types across several merges. This provides a glimpse of the morphological complexity of languages. However, one might argue that to really approach morphological complexity, our measures would have to consider the restrictions given by the allowed sequences of subword units within a word.

To address this concern, we compare our results with a corpus-based morphological complexity measure that aims to quantify the predictability of subword sequences within a word. This external approach uses a neural language model for estimating the predictability of sequences of n-grams within a word (Gutierrez-Vasques and Mijangos, 2020). H_{1gram} is the entropy rate obtained at the character level, H_{3gram} is the entropy rate using sequences of character trigrams.

The entropies of texts at merge 0 are already strongly correlated with H_{3gram} ($\rho = 0.81$). A higher correlation is obtained at merge 30 ($\rho = 0.86$). This means that at this point, the subword distribution is reflecting a complexity that is related to the predictability of its morphs – or at least of its character trigrams. This correlation starts to vanish in later merges, especially after the turning point (since we are probably starting to capture predictability more related to the word level).

Interestingly, H_{1gram} does not correlate with BPE entropies, not even at merge 0. This could be related to the fact that, even at merge 0, we distinguish between the characters at the end of an orthographic word versus any of the remaining positions. This already captures some degree of sequentiality that seems more related to H_{3gram}.

The measure H_{3gram} is restricted to predicting fixed-size overlapping sequences of characters within a word, while the tokenizations that we obtain with BPE contain subwords of variable lengths across merges. Despite this, it is interesting that these two measures strongly correlate at the first merge operations, suggesting that they reflect a similar phenomenon at the subword level.

5 Discussion

The "turning point" we discussed here reflects several phenomena. Firstly, languages become more similar in terms of their subword entropies around this point. Secondly, the trends of entropy and redundancy start to change (redundancy starts to grow, while entropy growth slows down). Thirdly, there is a shift in several cross-linguistic correlations around the same region of merges. It seems that two different subword distributions emerge, one before the turning point, and one afterward.

The entropy measured over word-level types reflect one dimension of diversity, i.e., some languages have rich inflectional morphology while others do not mark grammatical information word-internally. However, in another dimension, at the subword level, this variation is reduced.

If we rank languages by their entropy, the first merges' rankings are not correlated to the ones closer to the word level. In some of the languages, we see a clear trade-off across merges. For instance, some languages with high word entropy (low predictability, long words with the potential of combining many different morphological distinctions), have highly predictable subword distributions.

Languages where this trade-off is observed include: Basque (*eus*), Imbabura Highland Quichua (*qvi)*, Finnish (*fin*), Yagua (*yad*), Kalaallisut (*kal*). There are also examples of languages that start with high complexity on the first merges but they are not the most complex ones at the word level: Vietnamese (*vie*), Thai (*tha*), Hindi (*hin*), Yoruba (*yor*), Nama (*naq*). English (*eng*) is also an example of a language that has comparatively low entropy at the word level, probably due to a relative lack of productive inflections, but is not one of the least complex at the subword level (first merges).

However, this trade-off was not general, as we did not find a significant negative correlation between the entropies at first merges with the entropies at last merges.

Taking a linguistic perspective, several accounts have focused on the structure of the morphological paradigm and the predictability between the inflected forms (Blevins, 2006, 2016; Ackerman and Malouf, 2013; Cotterell et al., 2018). For instance, Ackerman and Malouf (2013) distinguish between two types of complexity: a) enumerative complexity (E-complexity), reflecting the diversity of morphological distinctions, word forms, paradigm size; and b) integrative complexity (I-complexity), related to systematic paradigmatic organization underlying the morphological surface patterns.

The entropy values of I-complexity tend to be lower and less disperse than the ones exhibited by E-complexity (Ackerman and Malouf, 2013). According to these observations, a morphological paradigm could grow (many different word forms, many morphosyntactic distinctions) as long as it maintains its predictive structure. This is argued to be the reason why languages vary more widely in the dimension of E-complexity, while being more constrained in the I-dimension. Even though our work is not based on paradigms, and it did not require the use of linguistically annotated data, our findings seem to point in a similar direction, at least in the sense that the internal predictability of words is more similar across languages than than their word-level predictability.

However, note that there is some evidence from language learning experiments – with neural networks and human participants – which suggests that both are more sensitive to E-complexity than I-complexity (Johnson et al., 2020). It is an open question how our word-internal predictability measures relate to language learning.

On a practical note, the fact that certain numbers of BPE merges lead to more similar entropies across languages could be beneficial for NLP multilingual tasks. To our knowledge, the entropy and redundancy of tokenized texts have not been used as a criterion for choosing an appropriate number of BPE merge operations. There is recent work that investigates how the number of merges can lead to more balanced distributions of subwords, improving tasks like NMT (Gowda and May, 2020).

Another important question is to what extent our findings can be generalized to other corpora. The corpus size, type of register, etc., are likely to influence the turning point. As a general trend we expect this convergence to arise in a relatively low number of BPE operations.

6 Conclusions

In this paper, we went from single characters to orthographic words through incremental merges of BPE. We observed that text entropy values across 47 typologically diverse languages are less dispersed at the subword level than at the word-level. Our findings revealed a curious turning point, around the merge 200, where the values are least dispersed. Around this point, subword token distributions gradually start to look like word-level distributions (subword- and word-level entropy rankings are correlated only after this point). Additionally, this is approximately the point where text redundancy starts to grow after an initial drop and also where entropy growth slows down considerably after initial fast growth.

At the early merges, the entropy of texts is strongly correlated with an independent measure based on modeling character trigrams sequences. This provides new evidence that contributes to the ongoing discussion regarding different types of linguistic complexity.

Finally, our analysis could provide a useful insight for NLP processing. Choosing the number of merges that result in more similar distributions across languages could lead to more suitable subword representations for multilingual settings.

Acknowledgments

We thank the EACL reviewers. This work has been partially supported by the SNSF grant no. 176305 and CONACYT.

References

Farrell Ackerman and Robert Malouf. 2013. Morphological organization: The low conditional entropy conjecture. *Language*, pages 429–464.

Christian Bentz, Dimitrios Alikaniotis, Michael Cysouw, and Ramon Ferrer-i Cancho. 2017. The entropy of words – learnability and expressivity across more than 1000 languages. *Entropy*, 19(6):275.

Christian Bentz, Tatyana Ruzsics, Alexander Koplenig, and Tanja Samardzic. 2016. A comparison between morphological complexity measures: typological data vs. language corpora. In *Proceedings of the workshop on computational linguistics for linguistic complexity (cl4lc)*, pages 142–153.

James P Blevins. 2006. Word-based morphology. *Journal of Linguistics*, pages 531–573.

James P Blevins. 2016. *Word and paradigm morphology*. Oxford University Press.

Ryan Cotterell, Sebastian J Mielke, Jason Eisner, and Brian Roark. 2018. Are all languages equally hard to language-model? *arXiv preprint arXiv:1806.03743*.

Christophe Coupé, Yoon Mi Oh, Dan Dediu, and François Pellegrino. 2019. Different languages, similar encoding efficiency: Comparable information rates across the human communicative niche. *Science Advances*, 5(9):eaaw2594.

Michael Cysouw and Bernhard Wälchli. 2007. Parallel texts: using translational equivalents in linguistic typology. *STUF-Sprachtypologie und Universalienforschung*, 60(2):95–99.

Katharina Ehret. 2016b. *An information-theoretic approach to language complexity: variation in naturalistic corpora*. Ph.D. thesis, Albert-Ludwigs-Universität Freiburg.

Katharina Ehret and Benedikt Szmrecsanyi. 2016a. An information-theoretic approach to assess linguistic complexity. In Raffaela Baechler and Guido Seiler, editors, *Complexity, isolation and variation*. de Gruyter, Berlin.

Gertraud Fenk-Oczlon and August Fenk. 1999. Cognition, quantitative linguistics, and systemic typology. *Linguistic Typology*, 3(2):151–177.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

Thamme Gowda and Jonathan May. 2020. Finding the optimal vocabulary size for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3955–3964, Online. Association for Computational Linguistics.

Joseph H Greenberg. 1960. A quantitative approach to the morphological typology of language. *International journal of American linguistics*, 26(3):178–194.

Ximena Gutierrez-Vasques and Victor Mijangos. 2020. Productivity and predictability for measuring morphological complexity. *Entropy*, 22(1):48.

Tamar Johnson, Kexin Gao, Kenny Smith, Hugh Rabagliati, and Jennifer Culbertson. 2020. Predictive structure or paradigm size? Investigating the effects of i-complexity and e-complexity on the learnability of morphological systems.

Patrick Juola. 1998. Measuring linguistic complexity: The morphological tier. *Journal of Quantitative Linguistics*, 5(3):206–213.

Jawaharlal Karmeshu. 2003. *Entropy measures, maximum entropy principle and emerging applications*, volume 119. Springer Science & Business Media.

Emmerich Kelih. 2010. The type-token relationship in slavic parallel texts. *Glottometrics*, 20:1–11.

Christo Kirov, John Sylak-Glassman, Rebecca Knowles, Ryan Cotterell, and Matt Post. 2017. A rich morphological tagger for English: Exploring the cross-linguistic tradeoff between morphology and syntax. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 112–117, Valencia, Spain. Association for Computational Linguistics.

Alexander Koplenig, Peter Meyer, Sascha Wolfer, and Carolin Mueller-Spitzer. 2017. The statistical trade-off between word order and word structure–large-scale evidence for the principle of least effort. *PloS one*, 12(3):e0173614.

Thomas Mayer and Michael Cysouw. 2014. Creating a massively parallel bible corpus. *Oceania*, 135(273):40.

Thomas Mayer, Bernhard Wälchli, Christian Rohrdantz, and Michael Hund. 2014. From the extraction of continuous features in parallel texts to visual analytics of heterogeneous areal-typological datasets. *Language Processing and Grammars. The role of functionally oriented computational models*, pages 13–38.

Sabrina J Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? *arXiv preprint arXiv:1906.04726*.

Robert Östling. 2015. Word order typology through multilingual word alignment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 205–211.

Derek Partridge. 1981. Information theory and redundancy. *Philosophy of Science*, 48(2):308–316.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? *arXiv preprint arXiv:1704.08352*.

Bernhard Wälchli and Michael Cysouw. 2012. Lexical typology through similarity semantics: Toward a semantic map of motion verbs.

A Languages

iso639_3	language	family
aey	Amele	Trans-New Guinea
amp	Alamblak	Sepik
ape	Bukiyip	Torricelli
apu	Apurinã	Arawakan
arn	Mapudungun	Araucanian
arz	Egyptian Arabic	Afro-Asiatic
bsn	Barasana-Eduria	Tucanoan
cha	Chamorro	Austronesian
deu	German	Indo-European
dgz	Daga	Dagan
ell	Modern Greek	Indo-European
eng	English	Indo-European
eus	Basque	Basque
fij	Fijian	Austronesian
fin	Finnish	Uralic
fra	French	Indo-European
hae	Eastern Oromo	Pama-Nyungan
gug	Paraguayan Guaraní	Afro-Asiatic
hau	Hausa	Afro-Asiatic
hin	Hindi	Indo-European
ind	Indonesian	Austronesian
jac	Popti'	Mayan
kal	Kalaallisut	Eskimo-Aleut
kat	Georgian	Kartvelian
kew	West Kewa	Trans-New Guinea
khk	Halh Mongolian	Altaic
kor	Korean	Korean
laj	Lango (Uganda)	Eastern Sudanic
mig	San Miguel El Grande Mixtec	Oto-Manguean
mya	Burmese	Sino-Tibetan
mzh	Wichí Lhamtés Güisnay	Matacoan
naq	Nama (Namibia)	Khoe-Kwadi
pes	Western Farsi	Indo-European
plt	Plateau Malagasy	Austronesian
qvi	Imbabura Highland Quichua	Quechuan
rus	Russian	Indo-European
sag	Sango	Niger-Congo
spa	Spanish	Indo-European
swh	Swahili	Niger-Congo
tgl	Tagalog	Austronesian
tha	Thai	Tai-Kadai
tur	Turkish	Altaic
vie	Vietnamese	Austro-Asiatic
xsu	Sanumá	Yanomam
yad	Yagua	Peba-Yaguan
yaq	Yaqui	Uto-Aztecan
yor	Yoruba	Niger-Congo

B Entropy and redundancy graphs

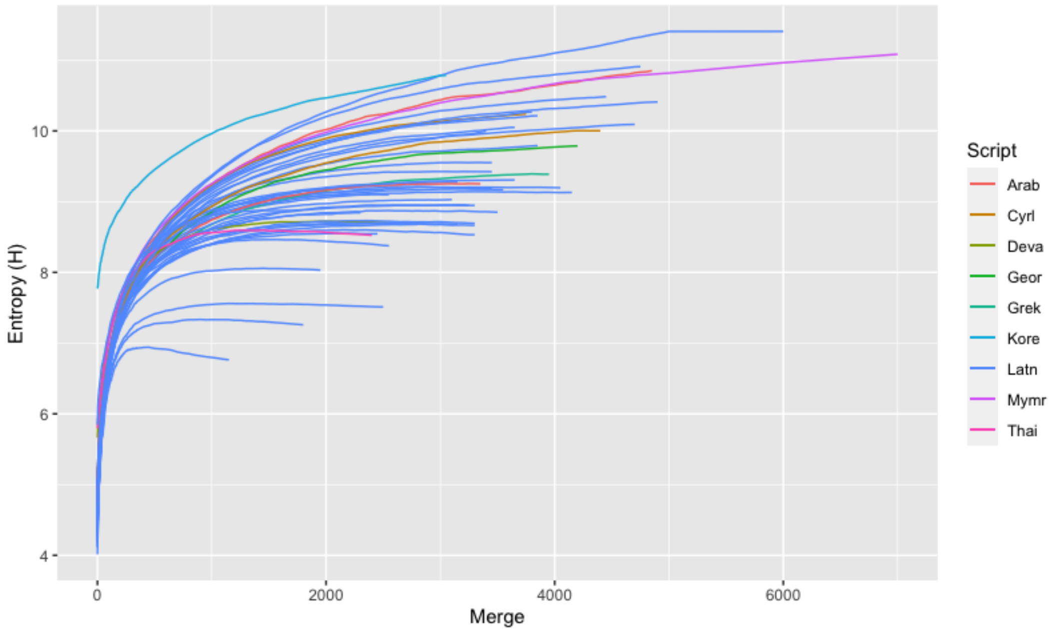


Figure 6: Entropy of all languages.

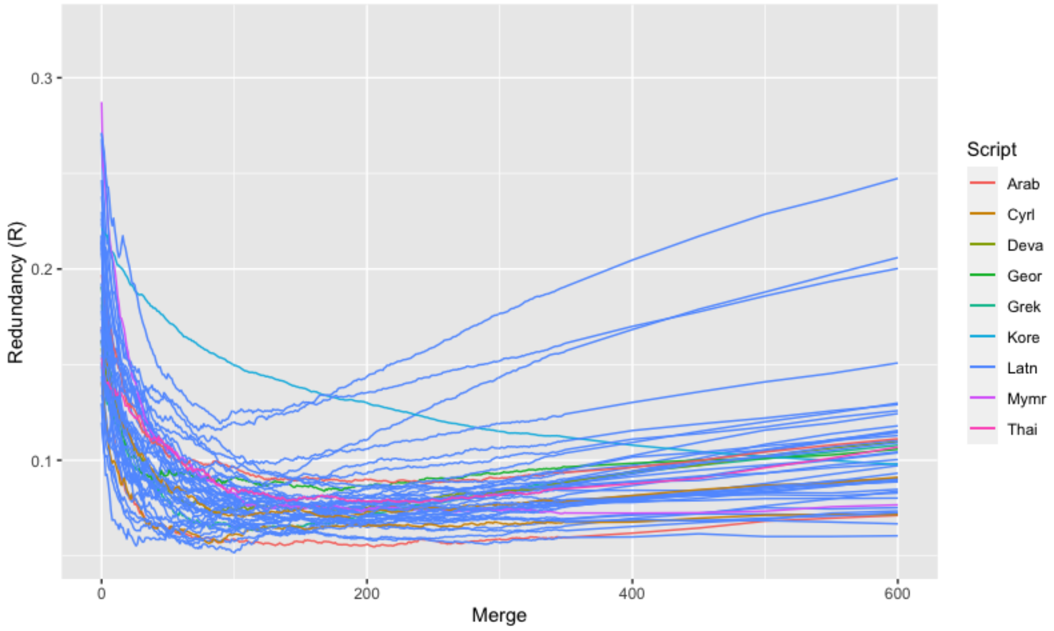
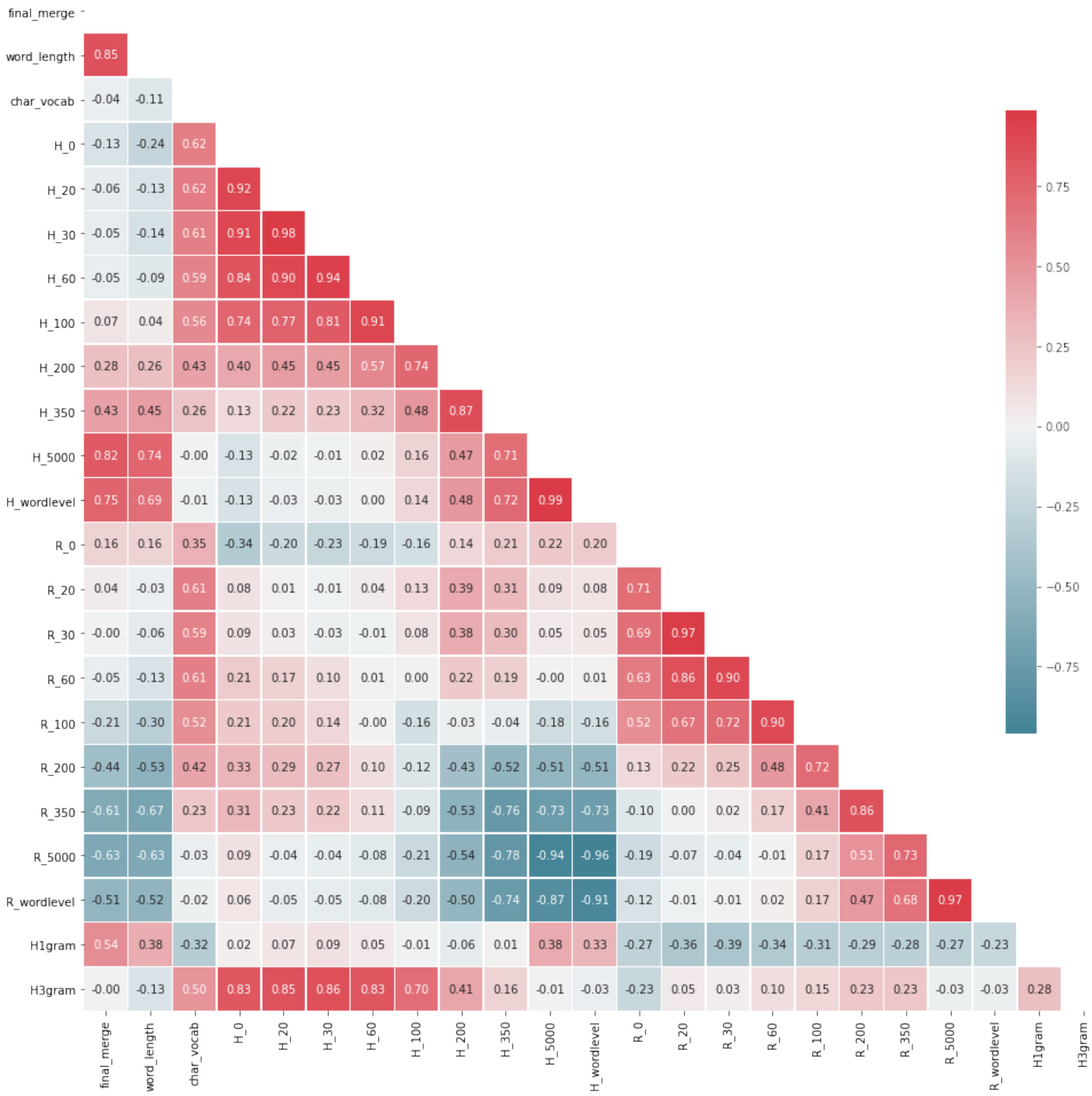


Figure 7: Redundancy of all languages (only the first 600 merges are shown in order to illustrate the merges in which the minimums are reached for most languages)

C Correlations



	final_merge	word_length	char_vocab	H_0	H_20	H_30	H_60	H_100	H_200	H_350	H_5000	H_wordlevel	R_0	R_20	R_30	R_60	R_100	R_200	R_350	R_5000	R_wordlevel	H1gram	H3gram
final_merge																							
word_length	0.85																						
char_vocab	-0.04	-0.11																					
H_0	-0.13	-0.24	0.62																				
H_20	-0.06	-0.13	0.62	0.92																			
H_30	-0.05	-0.14	0.61	0.91	0.98																		
H_60	-0.05	-0.09	0.59	0.84	0.90	0.94																	
H_100	0.07	0.04	0.56	0.74	0.77	0.81	0.91																
H_200	0.28	0.26	0.43	0.40	0.45	0.45	0.57	0.74															
H_350	0.43	0.45	0.26	0.13	0.22	0.23	0.32	0.48	0.87														
H_5000	0.82	0.74	-0.00	-0.13	-0.02	-0.01	0.02	0.16	0.47	0.71													
H_wordlevel	0.75	0.69	-0.01	-0.13	-0.03	-0.03	0.00	0.14	0.48	0.72	0.99												
R_0	0.16	0.16	0.35	-0.34	-0.20	-0.23	-0.19	-0.16	0.14	0.21	0.22	0.20											
R_20	0.04	-0.03	0.61	0.08	0.01	-0.01	0.04	0.13	0.39	0.31	0.09	0.08	0.71										
R_30	-0.00	-0.06	0.59	0.09	0.03	-0.03	-0.01	0.08	0.38	0.30	0.05	0.05	0.69	0.97									
R_60	-0.05	-0.13	0.61	0.21	0.17	0.10	0.01	0.00	0.22	0.19	-0.00	0.01	0.63	0.86	0.90								
R_100	-0.21	-0.30	0.52	0.21	0.20	0.14	-0.00	-0.16	-0.03	-0.04	-0.18	-0.16	0.52	0.67	0.72	0.90							
R_200	-0.44	-0.53	0.42	0.33	0.29	0.27	0.10	-0.12	-0.43	-0.52	-0.51	-0.51	0.13	0.22	0.25	0.48	0.72						
R_350	-0.61	-0.67	0.23	0.31	0.23	0.22	0.11	-0.09	-0.53	-0.76	-0.73	-0.73	-0.10	0.00	0.02	0.17	0.41	0.86					
R_5000	-0.63	-0.63	-0.03	0.09	-0.04	-0.04	-0.08	-0.21	-0.54	-0.78	-0.94	-0.96	-0.19	-0.07	-0.04	-0.01	0.17	0.51	0.73				
R_wordlevel	-0.51	-0.52	-0.02	0.06	-0.05	-0.05	-0.08	-0.20	-0.50	-0.74	-0.87	-0.91	-0.12	-0.01	-0.01	0.02	0.17	0.47	0.68	0.97			
H1gram	0.54	0.38	-0.32	0.02	0.07	0.09	0.05	-0.01	-0.06	0.01	0.38	0.33	-0.27	-0.36	-0.39	-0.34	-0.31	-0.29	-0.28	-0.27	-0.23		
H3gram	-0.00	-0.13	0.50	0.83	0.85	0.86	0.83	0.70	0.41	0.16	-0.01	-0.03	-0.23	0.05	0.03	0.10	0.15	0.23	0.23	-0.03	-0.03	0.28	

Figure 8: Correlation matrix

C.1 Significant correlations

Correlations still significant after Bonferroni correction (ordered from the highest to lowest coefficient).

var 1	**var 2**	**pvalue**	**corr**	**num**	**pvalue.correct**
H_5000	H_wordlevel	0	0.99	47	0
H_20	H_30	0	0.98	47	0
R_5000	R_wordlevel	0	0.97	47	0
R_20	R_30	0	0.97	47	0
H_30	H_60	0	0.94	47	0
H_0	H_20	0	0.92	47	0
H_60	H_100	0	0.91	47	0
H_0	H_30	0	0.91	47	0
R_30	R_60	0	0.9	47	0
H_20	H_60	0	0.9	47	0
R_60	R_100	0	0.9	47	0
H_200	H_350	1.55E-15	0.87	47	7.86E-13
H_30	H3gram	6.66E-15	0.86	47	3.37E-12
R_20	R_60	1.02E-14	0.86	47	5.17E-12
R_200	R_350	1.91E-14	0.86	47	9.66E-12
final_merge	word_length	3.46E-14	0.85	47	1.75E-11
H_20	H3gram	5.20E-14	0.85	47	2.63E-11
H_0	H_60	2.29E-13	0.84	47	1.16E-10
H_60	H3gram	6.39E-13	0.83	47	3.23E-10
H_0	H3gram	9.40E-13	0.83	47	4.75E-10
final_merge	H_5000	2.13E-12	0.82	47	1.08E-09
H_30	H_100	5.99E-12	0.81	47	3.03E-09
H_20	H_100	2.99E-10	0.77	47	1.51E-07
final_merge	H_wordlevel	1.10E-09	0.75	47	5.57E-07
word_length	H_5000	2.03E-09	0.74	47	1.03E-06
H_0	H_100	2.20E-09	0.74	47	1.11E-06
H_100	H_200	2.75E-09	0.74	47	1.39E-06
R_350	R_5000	6.46E-09	0.73	47	3.27E-06
R_30	R_100	1.03E-08	0.72	47	5.23E-06
H_350	H_wordlevel	1.07E-08	0.72	47	5.40E-06
R_100	R_200	1.63E-08	0.72	47	8.26E-06
H_350	H_5000	2.66E-08	0.71	47	1.35E-05
R_0	R_20	2.85E-08	0.71	47	1.44E-05
H_100	H3gram	4.84E-08	0.7	47	2.45E-05
word_length	H_wordlevel	7.18E-08	0.69	47	3.63E-05
R_0	R_30	7.28E-08	0.69	47	3.68E-05
R_350	R_wordlevel	1.52E-07	0.68	47	7.69E-05
R_20	R_100	1.97E-07	0.67	47	9.98E-05
R_0	R_60	2.05E-06	0.63	47	0.00104
char_vocab	H_20	2.96E-06	0.62	47	0.00150
char_vocab	H_0	3.96E-06	0.62	47	0.00200
char_vocab	R_60	4.81E-06	0.61	47	0.00243
char_vocab	H_30	5.67E-06	0.61	47	0.00287
char_vocab	R_20	6.21E-06	0.61	47	0.00314
char_vocab	H_60	1.05E-05	0.59	47	0.00529
char_vocab	R_30	1.12E-05	0.59	47	0.00569
H_60	H_200	3.24E-05	0.57	47	0.01639
char_vocab	H_100	4.30E-05	0.56	47	0.02175
H_200	R_5000	8.54E-05	-0.54	47	0.04322
final_merge	R_350	5.12E-06	-0.61	47	0.00259
final_merge	R_5000	2.45E-06	-0.63	47	0.00124
word_length	R_5000	2.28E-06	-0.63	47	0.00116
word_length	R_350	2.75E-07	-0.67	47	0.00014
H_wordlevel	R_350	6.79E-09	-0.73	47	3.43E-06
H_5000	R_350	5.52E-09	-0.73	47	2.79E-06
H_350	R_wordlevel	2.85E-09	-0.74	47	1.44E-06
H_350	R_350	4.38E-10	-0.76	47	2.22E-07
H_350	R_5000	8.86E-11	-0.78	47	4.48E-08
H_5000	R_wordlevel	4.44E-15	-0.87	47	2.25E-12
H_wordlevel	R_wordlevel	0	-0.91	47	0
H_5000	R_5000	0	-0.94	47	0
H_wordlevel	R_5000	0	-0.96	47	0

A Large-scale Evaluation of Neural Machine Transliteration for Indic Languages

Anoop Kunchukuttan, Siddharth Jain, Rahul Kejriwal
Microsoft India, Hyderabad
`{ankunchu,sija,rakejriw}@microsoft.com`

Abstract

We take up the task of large-scale evaluation of neural machine transliteration between English and Indian languages, with a focus on multilingual transliteration to utilize *orthographic similarity* between Indian languages. We create a corpus of 600K word pairs mined from parallel translation corpora and monolingual corpora, which is the largest transliteration corpora for Indian languages mined from public sources. We perform a detailed analysis of multilingual transliteration and propose an improved multilingual training pipeline for Indic languages. We analyse various factors affecting transliteration quality like language family, transliteration direction and word origin.

1 Introduction

Transliteration is an essential technology for multilingual and cross-lingual capabilities in NLP applications to handle named entities, support cross-script input methods. Transliteration between English and Indic languages is important since English is widely used in the Indian subcontinent. Indic languages are written in different scripts from various writing systems. We focus on languages using scripts derived from the ancient Brahmi script. Their character sets are very different from the Latin script - making transliteration non-trivial.

These scripts are *abugida* scripts, where the basic unit is the *akshar* which consists of one or more consonants along with a vowel diacritic (Daniels and Bright, 1996). They exhibit a high degree of grapheme-to-phoneme correspondence. There is a large overlap in the logical character sets of these scripts, though the visual appearance of the characters varies. The languages utilizing these scripts are said to exhibit *orthographic similarity* on account of various shared characteristics (Kunchukuttan et al., 2018a).

We undertake a systematic, large-scale evaluation of neural machine transliteration for 10 major Indic languages from 2 major language families (Indo-Aryan and Dravidian languages) spoken by more than a billion speakers. Other than *BrahmiNet* (Kunchukuttan et al., 2015) and *Dakshina* (Roark et al., 2020), no other previous work has explored a wide range of Indic languages; Dakshina only explores transliteration into Indic languages. Our major contributions are:

- For a large-scale evaluation, we mine 600K transliteration pairs across 10 languages from publicly available parallel and monolingual sources. This is much larger than existing corpora like MSR-NEWS (Banchs et al., 2015), Brahminet (Kunchukuttan et al., 2015), Dakshina (Roark et al., 2020) and other small datasets (Banchs et al., 2015; Kunchukuttan et al., 2018b; Gupta et al., 2012; Khapra et al., 2014). The BrahmiNet and Dakshina datasets span multiple languages; BrahmiNet is small and Dakshina by design consists mostly of Indian origin words.
- From the mined corpus, we create a high-quality, manually validated testset annotated with foreign and Indian origin words.
- We propose various improvements to the multilingual transliteration system proposed by Kunchukuttan et al. (2018a) for Indian languages, and suggest a *recipe* for building multilingual transliteration systems for Indic languages.
- We present an evaluation of transliteration systems according to various factors like language family, word origin and transliteration direction.

2 Mining transliteration corpus

This section explains our transliteration mining methods (from parallel and monolingual corpora) and presents an analysis of the mined corpus. We mine transliteration corpora from English to 10 In-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3469–3475
April 19 - 23, 2021.

Language	pa	hi	bn	or	gu	mr	kn	te	ml	ta
Word pair count (×1000)	55.3	157.7	65.4	34.7	65.5	38.0	24.7	77.4	31.1	57.1
Mining Accuracy	81.2	NA	76.7	NA	93.0	89.0	87.1	86.2	82.3	77.9

Table 1: Statistics on mined transliteration corpora

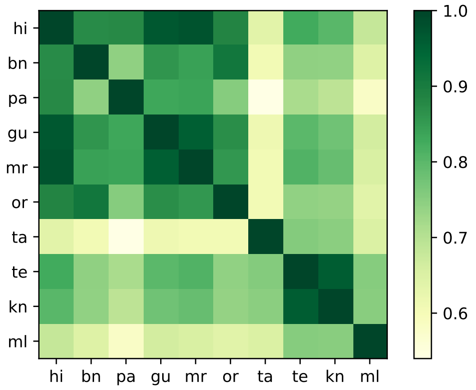


Figure 1: Orthographic Similarity: Indic languages

dian languages from 2 major language families: (a) Indo-Aryan branch of Indo-European family (Hindi, Marathi, Gujarati, Bengali, Odia, Punjabi), (b) Dravidian family (Kannada, Telugu, Malayalam, Tamil).

2.1 Mining from Parallel Translation Corpus

While very little transliteration corpora exists, a reasonable amount of parallel translation corpora between English and Indian languages are available in the public domain.

Method. Alignments between words in parallel sentences from two languages can be discovered from parallel translation corpora. The aligned words can either be translations or transliterations. We use the unsupervised method suggested by Sajjad et al. (2012) to mine transliteration pairs from these word alignments by distinguishing translation and transliterations. For morphologically rich languages, the approach can discover partial transliterations also. For instance, the English-Marathi pair word (*station*, स्टेशनावर [sTeshanaavara]). The Marathi word includes the locative case marker. To remove such transliteration pairs, we identify morphological variants by clustering together Marathi words corresponding to the same English word in the candidate transliteration pairs. We only retain the pair with the root word.

Mining Details. We mined transliteration pairs from English to Indian language parallel translation corpora from different sources (7.4 million sentence pairs across all languages, see Appendix A for details). We use the Moses transliteration mining module (Durrani et al., 2014) implementation of Sajjad et al. (2012) to mine transliteration pairs using the default settings.

2.2 Mining from Monolingual Corpora

Monolingual text corpora often have borrowed words from other languages (particularly English). We mine such transliteration pairs using only the vocabularies in the source and target languages.

Method. We first train initial transliteration models using available data in both directions ($L_e \rightarrow L_x$,$L_x \rightarrow L_e$) and build vocabularies for both languages (L_e, L_x). Given words in L_e, we identify the most promising transliteration candidates from L_x and then re-score these candidates. The scoring is based on edit-distance between Double Metaphone[1] representations of the words, which we found works well in practice. We consider scores in L_e as well as L_x. We use ITRANS[2] conversion from Indic scripts to Latin in order to be able to compute Double Metaphone representations on the Indic language side. Note that the phonetic nature of Indic scripts enables conversion of Indic scripts to Double Metaphone that is sufficient for transliteration mining. Thus, the score for a candidates pair $s(e, x)$ is $E(e, T_{XE}(x)) + E(x, T_{EX}(e))$, where E is the edit-distance function and T_{xy} denotes transliteration from x to y. As mentioned above, the strings are converted to Double Metaphone representation prior to edit-distance computation. Finally, we prune the generated pairs based on a chosen threshold of scores.

Mining Details. We use monolingual vocabulary from the AI4Bharat IndicNLP dataset (Kunchukuttan et al., 2020) and the OSCAR corpus (Ortiz Suarez et al., 2019) for Indic languages. For English, we use the AI4Bharat IndicCorp dataset (Kakwani et al., 2020) which contains crawls from English newspapers from India - this helps mining

---

[1] https://en.wikipedia.org/wiki/Metaphone#Double_Metaphone

[2] https://en.wikipedia.org/wiki/ITRANS

Indian named entities.

2.3 Characteristics of the Mined corpora

Corpora Statistics. Across 10 languages, we mined ~373k and ~339k transliteration pairs from the parallel translation and monolingual corpora respectively. The final train set of 606k word pairs was created after deduplicating and creating train, test and dev splits (See Table 1 for a summary of the mined corpus). We estimate that the training set has 55% non-Indian origin words and 45% Indian origin words.

Quality of the mined corpus. We evaluated the quality of mined transliterations via crowdsourcing. We used an internal, managed crowd-sourcing platform to validate the testsets and retained the transliteration pairs judged as correct in the final testset. The testset for every language had transliterations for 1500 English words and all their mined transliterations. This manual evaluation also gave us an estimation of the transliteration mining quality. We asked native-speaker judges for each language to report whether the pair is a transliteration or not. Our guidelines specified that pairs should be marked as valid if the pair is phonetically equivalent and are canonical spellings. In case no canonical spelling exists, the judges may mark the pairs solely on only phonetic equivalence. To control for quality, we used 3 judges per pair and used majority-voting for establishing correctness of a transliteration pairs. We added honey-pot pairs to tasks to filter out judges spamming our task.

Table 1 also shows the transliteration mining accuracies (average accuracy of 84.18%). An analysis of the errors revealed that an overwhelming majority involved wrong/missing/extra inflections (plurals and/or Indic casemarkers). These word pairs are also partial transliterations which are useful for learning transliteration models.

Test Set Creation. The test and dev sets were created by selecting 1500 English words each that are common across all language corpora along with their transliterations. We ensure that the test and dev set do not have any overlap with training set across languages. The testset were verified via crowdsourcing. The test set contains 928 foreign origin words and 572 Indian origin words.

Study of orthographic similarity. Following Kunchukuttan and Bhattacharyya (2020), we estimate the orthographic similarity between languages using the n-way parallel testset. For every language pair, it is the average Longest Common Subsequence Ratio (LCSR) (Melamed, 1995) between word pairs in the test set (See Figure 1) and follows linguistic genealogy. Tamil and Malayalam are most divergent to other languages. Punjabi is also divergent to other languages, possibly on account of: (a) some of its special characters like *tippi* and *addak*, (b) little use of conjunct consonants unlike other Indian languages.

3 Analysis: Multilingual Transliteration

We study multilingual transliteration models with the intent of identifying factors that improve multilingual models. First, we describe our baseline multilingual model and then introduce different variants to improve the baseline model.

Baseline Multilingual model (Kunchukuttan et al., 2018a). It is a character-level, attention-based, encoder-decoder model with all the model components shared amongst all the languages. We train joint EX (multi-target, English to Indian languages) and XE models (multi-source, Indian languages to English) separately. For EX models, we append a special target language token to the input sequence (Johnson et al., 2017).

Language Partitioning. To understand the role of orthographic similarity, we investigate two language groupings: (a) all the Indic languages are jointly trained, (b) Indo-Aryan and Dravidian languages are separately trained.

Vocabulary. Indic languages use a variety of scripts with a high overlap in the logical character set, but assigned unique characters in the Unicode character set. We investigate if transfer learning works better with a combined vocabulary by mapping logically equivalent characters across scripts for better transfer learning. We use the IndicNLP Library (Kunchukuttan, 2020) for mapping all Indic scripts to the Devanagari script, thus combining the vocabularies of all languages. We experiment with two configurations: (a) disjoint vocabularies (*i.e.,* different scripts), (b) combined vocabularies (*i.e.,* same script). Combining the vocabularies reduces the vocabulary significantly as the number of scripts reduces from 9 to 1.

Source language tag. In spite of the high degree of orthographic similarity between Indian languages, there are few cases of language-specific variations. For instance, the Malayalam script overloads a few characters with multiple sounds, Bengali pronunciation of the *aa* vowel differs, *etc.* To make the model sensitive to these language-

Experiment	Indo-Aryan (IA)						Dravidian (DR)				Average		
	pa	hi	bn	or	gu	mr	kn	te	ml	ta	IA	DR	IND
X to E TRANSLITERATION													
FOREIGN WORDS													
bilingual	48.53	52.29	50.85	48.84	43.81	52.93	56.95	50.58	54.9	38.28	49.54	50.18	49.80
all Indic	57.45	65.23	55.78	60.81	56.74	65.52	64.46	59.18	60.61	42.54	60.26	56.7	58.83
by family													
different scripts	61.27	65.23	59.64	60.58	59.18	66.19	63.23	58.28	61.02	44.29	62.02	56.7	59.89
same script	58.82	66.79	57.58	62.21	59.26	67.18	60.54	58.28	58.98	44.04	61.97	55.46	59.37
+source tag	60	64.50	60.36	60.93	59.10	68.07	62.56	57.48	63.88	45.04	**62.16**	**57.24**	**60.19**
INDIAN WORDS													
bilingual	68.01	73.18	68.02	74.79	71.14	81.29	77.63	74.29	71.9	45.99	72.74	67.45	70.62
all Indic	68.91	77.65	73.13	77.60	74.95	81.88	78	74.19	70.37	44.89	75.69	66.86	72.16
by family													
different scripts	71.49	78.32	73.85	79.07	75.16	82.96	79.5	76.11	75.18	49	76.81	69.95	74.06
same script	69.58	77.32	73.54	80.29	74.52	82	74.88	73.19	74.47	50.6	76.21	68.28	73.04
+source tag	71.16	78.44	72.92	79.56	75.05	84.03	78.88	76.21	75.18	50.5	**76.86**	**70.19**	**74.19**
E to X TRANSLITERATION													
FOREIGN WORDS													
bilingual	74.24	68.36	80.14	72.02	75.21	73.77	73.45	74.23	65.09	67.22	73.96	70	72.37
all Indic	75.83	76.92	81.29	75.43	80.68	79.97	77.4	78.55	68.53	73.22	**78.35**	74.43	76.78
by family													
different script	76.55	73.44	81.14	74.29	79.45	77.15	78.67	80.36	71.39	75.73	77.00	**76.54**	76.82
same script	77.99	74.17	82.29	74.43	80.27	77.57	77.68	79.11	71.24	73.5	77.79	75.38	**76.83**
INDIAN WORDS													
bilingual	78.61	71.69	75.34	79.65	76.47	80.33	75	78.9	72.45	76.26	77.01	75.65	76.47
all Indic	82.83	79.22	85.04	83.42	85.16	87.11	78.51	81.85	77.96	80.9	**83.80**	79.81	**82.20**
by family													
different script	81.34	77.96	81.54	80.19	82.49	84.94	79.86	81.18	76.58	80.22	81.41	79.46	80.63
same script	83.11	79.64	84.77	81.27	84.22	85.62	79.86	83.33	77.55	81.72	83.10	**80.62**	82.11

Table 2: Multilingual Transliteration results (%accuracy). The *all Indic* experiment trains with a common script.

specific variations in XE models, we add an special source language token in the input sequence.

Addressing divergence between Tamil and other Indic scripts. The Tamil script is highly under-specified and has fewer characters than sounds in the English language (unlike other Indic scripts). When training a multilingual model, there is an inconsistency in learnt mappings between Tamil and other Indic scripts. We address this issue by training a Tamil-specific multilingual model for Dravidian languages where all characters from other scripts are mapped to the closest character in the Tamil script via deterministic rules using the IndicNLP library.

3.1 Experimental Setup

We use *Marian* (Junczys-Dowmunt et al., 2018) to train our transliteration models. We use 128 LSTM units for encoder and decoder (1 layer for bilingual models and 2 layers for multilingual models). The encoder uses a bidirectional LSTM. The input embeddings are also 128 units in size. These hyperparameters were decided based on a parameter sweep on the dev set. We use a batch size of 100 sequences and early stopping with patience=100. We use beam-search for decoding (beam size=4).

3.2 Results and Discussion

Table 2 shows the top-1 accuracy of the different models. For translation into Indian languages, multiple references are available.

Bilingual models. Bilingual results show some trends about Indic transliteration. Transliteration is more difficult for non-Indian origin words than Indian origin words. EX direction accuracies are higher than XE direction. Tamil transliteration accuracy is the least in the XE direction (due to deficient orthography), while Malayalam has the least accuracy in the EX direction (possibly due to overloading of some characters).

Impact of multilingual training. In the XE direction, multilingual systems provide significant gains over bilingual systems (~20%). Most gains come from improved accuracy on non-Indian words. The multilingual system is better at generating canonical spelling in contrast to phonetically-equivalent incorrect spellings. More gains are observed for Indo-Aryan languages compared to Dravidian languages. Tamil benefits the most among Dravidian languages, but it still lags behind other languages. In the EX direction, accuracy improves by 6-7% for both Indian and non-Indian words. Both Indo-Aryan and Dravidian languages benefit from multilingual training.

Effect of language family. We do not observe any major advantage in training the two language families together. Hence, in subsequent experiments we train separate models. There are some differences in the spelling conventions between these languages. Thus, it seems a reasonable conservative choice to train separate model for the two families in the face of data not bringing any clarity.

Effect of source language tag. Adding source language tag improves XE transliteration accuracy 6-7% for some languages with divergent spelling conventions (Bengali, Malayalam and Tamil).

Effect of script conversion. It significantly reduces the vocab size (reducing number of scripts from 9 to 1), but results in just a small drop in accuracy. For the XE direction, the drop in accuracy is recovered by using the source language tag.

Mapping to Tamil Script. The table below show that this simple approach improves the Tamil transliteration accuracy by 2-5 points on non-Indian as well as Indian words in both directions.

Experiment	ta-en		en-ta	
	foreign	indic	foreign	indic
hiscript	44.04	50.6	73.5	81.72
tascript	**47.37**	**53.3**	**78.8**	**83.9**

4 Conclusion

We present a study of transliteration between English and Indic languages. We mine a 600k parallel transliteration corpus having a good coverage of Indian and non-Indian origin words as well as create a manually validated testset. We recommend the following recipe for Indic multilingual transliteration: (a) all training data in the same script, (b) separate models for IA and DR languages, (c) source language tags for XE transliteration. Multilingual training significantly improves non-Indian word transliteration. Our results validate previous results on benefits of multilingual transliteration on a wider set of languages and larger datasets. We also improve Tamil to English transliteration by representing multilingual data in Tamil script. More details about the corpus is available at `https://github.com/anoopkunchukuttan/indic_transiteration_analysis`.

References

Željko Agić and Ivan Vulić. 2019. JW300: A wide-coverage parallel corpus for low-resource languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy. Association for Computational Linguistics.

Rafael E. Banchs, Min Zhang, Xiangyu Duan, Haizhou Li, and A. Kumaran. 2015. Report of NEWS 2015 Machine Transliteration Shared Task. In *Proceedings of the Fifth Named Entities Workshop*.

Peter T Daniels and William Bright. 1996. *The world's writing systems*. Oxford University Press on Demand.

Nadir Durrani, Barry Haddow, Philipp Koehn, and Kenneth Heafield. 2014. Edinburgh's phrase-based machine translation systems for WMT-14. In *Proceedings of the ACL 2014 Ninth Workshop on Statistical Machine Translation*.

Kanika Gupta, Monojit Choudhury, and Kalika Bali. 2012. Mining Hindi-English transliteration pairs from online Hindi lyrics. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2459–2465, Istanbul, Turkey. European Language Resources Association (ELRA).

Barry Haddow and Faheem Kirefu. 2020. PMIndia – A Collection of Parallel Corpora of Languages of India. *arxiv 2001.09907*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of EMNLP*.

Mitesh M Khapra, Ananthakrishnan Ramanathan, Anoop Kunchukuttan, Karthik Visweswariah, and Pushpak Bhattacharyya. 2014. When transliteration met crowdsourcing: An empirical study of transliteration via crowdsourcing using efficient, non-redundant and fair quality control. In *LREC*, pages 196–202. Citeseer.

Anoop Kunchukuttan. 2020. The IndicNLP Library. `https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf`.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2020. Utilizing language relatedness to improve machine translation: A case study on languages of the Indian Subcontinent. *arXiv preprint arXiv:2003.08925*.

Anoop Kunchukuttan, Divyanshu Kakwani, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. AI4Bharat-IndicNLP Corpus: Monolingual Corpora and Word Embeddings for Indic Languages. *arXiv preprint arXiv:2005.00085*.

Anoop Kunchukuttan, Mitesh Khapra, Gurneet Singh, and Pushpak Bhattacharyya. 2018a. Leveraging orthographic similarity for multilingual neural transliteration. *Transactions of the Association for Computational Linguistics*, 6:303–316.

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2018b. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Anoop Kunchukuttan, Ratish Puduppully, and Pushpak Bhattacharyya. 2015. Brahmi-Net: A transliteration and script conversion system for languages of the Indian subcontinent. In *Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies: System Demonstrations*.

I Dan Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. In *Third Workshop on Very Large Corpora*.

Pedro Javier Ortiz Suarez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Shantipriya Parida, Ondrej Bojar, and Satya Ranjan Dash. 2018. Odiencorp: Odia-english and odia-only corpus for machine translation. In *Proceedings of the Third International Conference on Smart Computing and Informatics*.

Matt Post, Chris Callison-Burch, and Miles Osborne. 2012. Constructing parallel corpora for six Indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada. Association for Computational Linguistics.

Loganathan Ramasamy, Ondřej Bojar, and Zdeněk Žabokrtský. 2012. Morphological Processing for English-Tamil Statistical Machine Translation. In *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages*.

Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J. Mielke, Cibu Johny, Işin Demirşahin, and Keith Hall. 2020. Processing South Asian languages written in the Latin script: the Dakshina dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference (LREC)*, pages 2413–2423.

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2012. A statistical model for unsupervised and semi-supervised transliteration mining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *arxiv 1907.05791*.

Parth Shah and Vishvajit Bakrola. 2019. Neural Machine Translation System of Indic Languages - An Attention based Approach. In *2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP)*. IEEE.

Shashank Siripragrada, Jerin Philip, Vinay P. Namboodiri, and C V Jawahar. 2020. A multilingual parallel corpora collection effort for Indian languages. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3743–3751, Marseille, France. European Language Resources Association.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

A Corpora used for Transliteration mining

Table 3 shows the language-wise parallel corpora statistics and Table 4 lists the various parallel corpora used for transliteration mining. Table 3 also shows the vocab sizes of monolingual corpora used for each language for the monolingual approach.

Language		ParInfo	MonoInfo
Punjabi	(pa)	535,796	476K
Hindi	(hi)	1,586,775	511K
Bengali	(bn)	444,593	501K
Oriya	(or)	116,492	490K
Gujarati	(gu)	557,342	524K
Marathi	(mr)	732,093	539K
Kannada	(kn)	450,139	479K
Telugu	(te)	664,670	596K
Malayalam	(ml)	708,266	596K
Tamil	(ta)	1,640,920	599K
English	(en)	-	781K
Total		7,437,086	6092K

Table 3: Information on copora used for transliteration mining. **ParInfo**: Parallel Translation Corpora Size per English-Indian language pair. WikiMatrix contributes around 1.7 millions pairs across languages. **MonoInfo**: Vocabulary size of monolingual corpora per language.

Source	Citation
CVIT-Mann ki Baat	(Siripragrada et al., 2020)
CVIT-PIB	(Siripragrada et al., 2020)
IITB en-hi v2.0	(Kunchukuttan et al., 2018b)
MTurk Corpora	(Post et al., 2012)
JW300	(Agić and Vulić, 2019)
MTEnglish2Odia	
NLPC-Uom Corpus	
OdiEnCorp 1.0	(Parida et al., 2018)
OPUS	(Tiedemann, 2012)
PMIndia	(Haddow and Kirefu, 2020)
UFAL-en-ta-v2	(Ramasamy et al., 2012)
Urs Tarsadia Corpus	(Shah and Bakrola, 2019)
Wikimatrix	(Schwenk et al., 2019)
Wikititles	

Table 4: Parallel Translation Corpora used for mining transliterations. All download URLs can be obtained from `https://github.com/AI4Bharat/indicnlp_catalog`

Communicative-Function-Based Sentence Classification for Construction of an Academic Formulaic Expression Database

Kenichi Iwatsuki[1]
[1]The University of Tokyo
`iwatsuki@nii.ac.jp`

Akiko Aizawa[21]
[2]National Institute of Informatics
`aizawa@nii.ac.jp`

Abstract

Formulaic expressions (FEs), such as '*in this paper, we propose*' are frequently used in scientific papers. FEs convey a communicative function (CF), i.e. 'showing the aim of the paper' in the above-mentioned example. Although CF-labelled FEs are helpful in assisting academic writing, the construction of FE databases requires manual labour for assigning CF labels. In this study, we considered a fully automated construction of a CF-labelled FE database using the top–down approach, in which the CF labels are first assigned to sentences, and then the FEs are extracted. For the CF-label assignment, we created a CF-labelled sentence dataset, on which we trained a SciBERT classifier. We show that the classifier and dataset can be used to construct FE databases of disciplines that are different from the training data. The accuracy of in-disciplinary classification was more than 80%, while cross-disciplinary classification also worked well. We also propose an FE extraction method, which was applied to the CF-labelled sentences. Finally, we constructed and published a new, large CF-labelled FE database. The evaluation of the final CF-labelled FE database showed that approximately 65% of the FEs are correct and useful, which is sufficiently high considering practical use.

1 Introduction

Formulaic expressions (FEs), such as '*in this paper we propose*', are a type of multi-word expressions and are repeatedly used in scientific papers. Some FEs convey a communicative function (CF) of a sentence, which represents intentions of authors. For example, '*in this paper, we propose*' conveys the CF of 'showing the aim of the paper'.

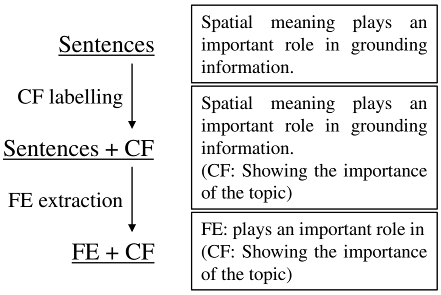


Figure 1: Process of creating FE database.

Databases comprising CF-labelled FEs are required from a pedagogical perspective (Martinez and Schmitt, 2012), and a computer-based academic writing assistance system[1] that uses such CF-labelled FEs has been proposed (Mizumoto et al., 2017). Several attempts have been made to extract FEs from scientific corpora and categorise them based on CFs (Cortes, 2013; Ädel, 2014; Mizumoto et al., 2017; Morley, n. d.; Simpson-Vlach and Ellis, 2010; Lu et al., 2018). A CF-labelled FE database can be constructed using two main approaches: top–down and bottom–up approaches (Biber et al., 2007; Durrant and Mathews-Aydınlı, 2011). By using the top–down approach, sentences are first assigned CF labels, and then FEs are extracted, while in the case of the bottom–up approach, FEs are first extracted and then assigned CF labels. To date, both the approaches have been adopted because CF assignment is performed manually (Table 1). In this paper, we propose a fully automated construction of the CF-labelled FE database, where we consider the top–down approach to be more beneficial (Figure 1). This is because the bottom–up approach requires FEs to be classified, which is difficult because a perfect FE-extraction technique is yet to be realised, and FE embeddings have not been investigated intensively. The top–down approach requires sentence classification, which has highly improved with the recent advancements on pre-trained models.

---

[1] `http://langtest.jp/awsum/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3476–3497
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

	Method for creating DB			DB statistics			
	Approach	CF	FE	Discipl.	#CFs	#Docs	#FEs
Simpson-Vlach and Ellis (2010)	bottom–up	manual	corpus	mixed	15	-	200
Morley (n. d.)	-	manual	manual	mixed	146	100	$\simeq 2,000$
Mizumoto et al. (2017)	top–down	manual	corpus	specific	52	1,000	-
Lu et al. (2018)	bottom–up	manual	corpus	mixed	12	600	454
Ours	top–down	automated	sentence	specific	32	61,728	86,931

Table 1: Properties of the existing and proposed methods for the construction of CF-labelled FE databases and the statistics of the databases. The approach of Morley (n. d.) is unknown. For the CF assignment (CF), we adopted supervised machine-learning. The FE extraction (FE) was conducted manually using a corpus- or sentence-level method. Either FEs specific to one discipline were extracted or FEs used in a corpus in which several disciplines were mixed were extracted. The number of documents used for extraction and the extracted FEs of the existing and presented database were shown. Some studies did not disclose the number of documents or FEs. Morley (n. d.) constantly revises the database, and therefore the number of FEs is not fixed.

For CF-based sentence classification, we created a dataset for supervised learning. The dataset consists of a small number of sentences that were assigned CF labels. We collected the sentences from scientific papers of multiple disciplines. By using this dataset, we fine-tuned SciBERT (Beltagy et al., 2019). Additionally, because there are preferences for CF usage depending on disciplines and as the preparation and coverage of all CFs of every discipline are difficult, sentences to which any prepared CF label should not be assigned may appear in a corpus (no-CF sentences). These no-CF sentences will have a negative effect on the classification performance. Based on the recent work on out-of-distribution detection in natural language processing (Hendrycks and Gimpel, 2017; Hendrycks et al., 2020), we used the maximum value of the softmax layer as the threshold to filter no-CF sentences in order to improve the final precision. The experimental results show that the maximum value of the softmax layer works well as the threshold to filter out undesirable sentences.

We carefully considered multidisciplinary problems in the classification. Although the development of a training dataset for every discipline in the world is obviously impossible, demonstrating a successful classification using a single disciplinary dataset is not sufficient for practical use. In this study, we determined whether a model trained on a corpus of one discipline can be applied to that of another discipline. Moreover, the effects of a pre-training dataset were examined by comparing SciBERT and BERT (Devlin et al., 2019). The experimental results show that the classifiers performed fairly well in terms of both in-discipline and cross-discipline data, and the performance was only slightly affected when scientific papers were not used as pre-training data.

For the FE-extraction process, one FE should be extracted from one sentence because CF labels are assigned to each sentence; this is termed as sentence-level approach (see Section 2.2). Therefore, we propose a sentence-level FE extraction method that is based on an existing method (Iwatsuki et al., 2020b). The method consists of three steps: named and scientific entity removal, dependency-structure-based word removal, and word-association-measure-based word removal.

Finally, we created a new, large, multidisciplinary CF-labelled FE database and evaluated it by asking human evaluators whether each instance was assigned a correct CF label and whether an FE was useful for writing a paper. The results show that approximately 65% of the collected FEs are appropriate.

The contributions of our study are as follows:

- we created and published the CF-labelled sentence dataset, which is the first dataset for training and evaluation of CF-based classification;
- we showed that a simple SciBERT-based neural classifier performed reasonably well for the CF labelling problem;
- we showed that the SciBERT classifier can be used even though the discipline of the training data is different from the inferred one;
- we proposed an FE extraction method; and

- we constructed a CF-labelled FE database with the top–down approach, which is larger than the existing databases but still maintains high quality.

2 Related Work

2.1 CFs in Scientific Papers

The CFs of scientific papers were first introduced by Swales (1990), who focused on the CFs in the introduction section. The author proposed a hierarchical structure of CFs, in which *move* was considered a larger unit of CF and *step* was a smaller unit belonging to *move*. He found that the introduction section consists of three *moves*: 'establishing a territory', 'establishing a niche', and 'occupying the niche'. Each *move* has several *steps*, such as 'claiming centrality' and 'presenting research questions or hypotheses' (Swales, 2004). Following his work, a host of studies extended the concept to all parts of a scientific paper. Most studies focused on very limited part of scientific papers; only the introduction (Ozturk, 2007), methods (Lim, 2006; Cotos et al., 2017), results (Basturkmen, 2009; Lim, 2010), discussion sections (Peacock, 2002; Basturkmen, 2012), or the abstracts (Lorés, 2004; Darabad, 2016; Rashidi and Meihami, 2018; Saboori and Hashemi, 2013).

The concept was extended to all parts of a scientific paper. For example, Kanoksilapatham (2005) proposed the CF structure of all the sections in biochemistry papers. Cotos et al. (2015) proposed a CF set for all four sections, i.e. introduction, methods, results, and discussion sections. Maswana et al. (2015) compared the usage of the CFs in five engineering fields and found that certain CFs are preferred depending on the discipline.

Argumentative Zoning is a similar concept based on the rhetorical moves (Teufel, 1999). It had seven categories, which were later extended to 15 categories by Teufel et al. (2009)

Previous studies on CF-based classification used conditional random fields (Hirohata et al., 2008), a classifier chain with sequential minimum optimisation, Rakel with the J48 algorithm (Dayrell et al., 2012), a Bayes classifier, and a decision tree (Soonklang, 2016). However, these studies only focused on abstracts of scientific papers. Therefore, existing CF-labelled FE lists were created by manually assigning CF labels (Table 1), complicating the construction of a large CF-labelled FE database. Recently, Fiacco et al. (2019) used a hierarchical Bi-LSTM+CRF to classify sentences. However, CF-labelled sentence corpora are yet to be made available to the public.

2.2 FE-Extraction Methods

Two approaches are used for extracting FEs: corpus- and sentence-level approaches. Based on the intuition that FEs appear frequently or words composing FE are strongly associated, most studies use the corpus-level approach, in which statistical metrics, such as frequency or mutual information, are applied to a whole corpus. To extract FEs, word n-grams were collected from a whole corpus by using the metrics (Biber et al., 2004; Simpson-Vlach and Ellis, 2010; Kermes, 2012; Mizumoto et al., 2017). However, this approach results in the extraction of an explosive number of overlapping n-grams, thus causing a serious problem in the CF-labelled FE database construction. For instance, suppose '*in this paper we propose*', '*this paper we propose a*', and '*in this paper we propose a new method*' are extracted, a criterion is needed to determine which of these are regarded as FEs; however, determining such a criterion is difficult.

The n-gram lattice method (Brooke et al., 2017) is one approach to address this problem; here, scores of various aspects of *formulaicity* are first calculated for all word n-grams. Next, an objective function that contains all scores of the n-grams is maximised to determine which n-grams should be disregarded and which should remain. However, this method is still not focused on FEs conveying CFs but on general phrasal expressions; thus, it is thus not suitable for our setting.

The sentence-level approach assumes that one FE occurs in one sentence. Thus, '*in this paper we propose a new method*' can be extracted, but '*this paper we propose a*' cannot be extracted from a sentence. This approach is also useful for extracting FEs with a slot (Vincent, 2013), into which some words can be inserted, such as '*however, * have not been reported*'. This setting is regarded as a sequence-labelling problem, in which each word of a sentence is labelled as either formulaic or non-formulaic. Liu et al. (2016) proposed removing topic-specific words as non-formulaic words, using latent Dirichlet allocation. They used a corpus consisting of papers from various disciplines, and tried to remove discipline-specific vocabulary. Thus, this is not suitable for extracting discipline-specific FEs. Iwatsuki et al. (2020b) proposed removing scientific and named entities in addition to dependency-based word removal.

The evaluation of the FE extraction model is another problem. Brooke et al. (2015) pointed out that the comparison of newly extracted FEs with existing reference is unreasonable because if a reference is on point, a new lexicon need not be created. Thus, Iwatsuki et al. (2020b) proposed evaluating FE extraction methods by a CF-based sentence retrieval task as an extrinsic task based on the idea that FEs convey a CF of a sentence.

2.3 CF-Labelled FE Databases

Table 1 describes the existing CF-labelled FE databases. Previous studies have shown that FEs are discipline-specific, and the resource of academic vocabulary should be presented for each discipline (Hyland and Tse, 2007; Liu, 2012). Thus, the development of CF-labelled FE databases for each discipline is important; however, many studies have focused on *general* FEs, which were extracted from a mixed corpus consisting of scientific papers on multiple disciplines. Some studies adopted the discipline-specific approach; Mizumoto et al. (2017) considered only the journals on applied linguistics, while Lu et al. (2018) used only the introductions of social-science papers. Moreover, only a small number of documents were used because the existing resources require manual labour for assigning CF labels.

Hence, we contend that the automated CF-based classification is helpful for constructing a large, comprehensive CF-labelled FE database. In this study, we developed a discipline-specific database based on large corpora of scientific papers from four disciplines.

3 Methods

3.1 Corpora and Datasets

3.1.1 Corpora of Scientific Papers

In this study, we considered the corpora which satisfy the following conditions. First, because we use full text of scientific papers and have made all the data public, papers must be open access. Second, to construct a comprehensive database, the corpora size is important. Third, for cross-discipline analyses, a discipline-specific journal is preferred to a multidisciplinary journal. We selected a corpus containing at least 10,000 papers.

Under these three conditions and based on the diversity of the disciplines, we selected four corpora: ACL Anthology Sentence Corpus[2] for computational linguistics (CL), Molecules[3] for chemistry (Chem), Oncotarget[4] for oncology (Onc), and Frontiers in Psychology[5] for psychology (Psy). Each corpus comprises more than 10,000 papers and is open access to full text (creative commons licence).

For pre-processing, we performed sentence splitting using ScispaCy (Neumann et al., 2019) and replaced citations and mathematical formulae with a special token. By using a simple rule-based method, section labels were normalised into five classes: introduction, methods, results, discussion, and other. Each sentence was assigned a section label; we did not use sentences belonging to the 'other' class. The numbers of sentences and documents are listed in Table 2.

Corpus	#Doc.	#Sent.	#Words
CL	13,921	1,612,921	32,698,072
Chem	15,949	1,703,902	39,303,460
Onc	19,541	3,029,285	68,719,634
Psy	12,317	1,948,082	49,329,526

Table 2: Number of documents (doc), sentences (sent), and words in each corpus.

3.1.2 CF Set and CoreFEs

Till date, there is no established CF set, and some CFs are not used or are frequently used in a specific discipline. Proposing a new CF set is beyond the scope of this study; however, we must select a CF set. We adopted the CF set proposed by Iwatsuki et al. (2020a), which was based on CFs used in Academic Phrasebank (Morley, n. d.). Table 3 describes the numbers of CFs in each section. (All the CFs are listed in Table 13 in the appendix.) CoreFE is an FE that is shortened so that it can be used as a query for sentence retrieval (Generally, longer phrases result in few or no results in sentence retrieval). We used CoreFEs to create the CF-labelled sentence dataset.

3.1.3 CF-Labelled Sentence Dataset

For the CF-based classification, we created a sentence dataset by using the aforementioned corpora. To effectively collect labelled sentences, we used the following procedures. First, the CoreFEs were

---

[2] `https://github.com/KMCS-NII/AASC`

[3] `https://www.mdpi.com/journal/molecules`

[4] `https://www.oncotarget.com/`

[5] `https://www.frontiersin.org/journals/psychology`

Section	#CFs
Introduction	11
Methods	6
Results	6
Discussion	9

Table 3: Numbers of CFs for each section.

used as queries to retrieve sentences from the corpora. Although the CoreFEs have CF labels, the retrieved sentences may not always have the same CFs.

Next, we used Amazon Mechanical Turk (AMT) to check if each sentence was assigned correct labels; this process was three-fold. First, a *correct* set of sentences was prepared. Two experts were asked whether the sentences in the correct set were correctly labelled, and the sentences whose labels were judged incorrect by at least one expert were removed. Another set of sentences, called the *incorrect* set, was prepared, in which the same sentences were randomly assigned incorrect labels. Second, by using these sets, a pilot test was conducted on AMT. Five annotators were recruited and asked to check whether the labels were correct or not. Based on this pilot test, we determined the threshold to cut off sentences. Finally, a larger set of sentences was prepared, which was different from the set used in the pilot test. Another five annotators were asked to perform the same task on the set. The final dataset comprises the sentences satisfying the threshold.

3.2 Sentence Classification

3.2.1 Classifier

We assigned each sentence a CF label, and this task can be regarded as a CF-based sentence-classification problem. In addition, we used SciBERT (Beltagy et al., 2019) with an additional linear layer for classification. We split the CF-labelled sentence dataset into training/development and evaluation sets so that four sentences for each CF were in the evaluation set. Then, we conducted five-fold cross validation using the training/development set for parameter tuning. Subsequently, we fine-tuned the classifier and evaluated the classification accuracy.

Because CF sets in scientific papers have not been established, the CF set we used cannot satisfactorily cover all sentences written in papers. Additionally, pre-processing errors, such as sentence splitting, sometimes result in no-CF sentences. Thus, in some scenarios, no CF should be assigned to a sentence and no-CF sentences must be removed. The no-CF class is not contained in the training dataset; this problem is regarded as the out-of-distribution detection problem. Although the maximum value of the softmax layer is not a perfect metrics for out-of-distribution detection, pre-trained transformers, such as BERT and RoBERTa, with a softmax layer are good detectors of out-of-distribution data (Hendrycks and Gimpel, 2017; Hendrycks et al., 2020).

To manage the no-CF sentences, we used the maximum softmax value of the classifier, and verified its performance. The verification was performed in the same manner as the creation of the CF-labelled sentence dataset. That is, we asked five AMT annotators whether the output label was correct. The threshold was also the same: 5/5.

3.2.2 Multidisciplinary Perspectives

To create a multidisciplinary database, the classification must be applied to various disciplinary texts. As it is costly to create a training dataset manually for each discipline, we tested whether the classifier trained on a dataset of one discipline can be immediately applied to the datasets of other disciplines.

SciBERT was trained on scientific papers from Semantic Scholar[6] (Beltagy et al., 2019). The corpora used in this study are open access and were also included in Semantic Scholar. Thus, we hypothesise that the cross-disciplinary adaptation is successful because the sentences are (partly) contained in the pre-training dataset. Therefore, the method cannot be applied to disciplines that are not covered by the pre-training dataset. To verify this hypothesis, we compared SciBERT to BERT, which was pre-trained on the book corpus and Wikipedia and not on scientific papers (Devlin et al., 2019), for cross-discipline sentence classification.

3.3 FE Extraction

To extract FEs, we propose a method based on Iwatsuki et al. (2020b), which is a sentence-level method; one FE was extracted from one sentence. We applied this method, which comprises three steps, to the classified sentences.

In the first step, the named and scientific entities are removed from a sentence. The entity recognition was performed using SpERT (Eberts and

---

[6] `https://www.semanticscholar.org/`

Ulges, 2020), which sits atop the leader-board of NER tasks for scientific entities[7]. For training, we used CoNLL04 (Roth and Yih, 2004), a corpus labelled with general-purpose named entities, and SciERC (Luan et al., 2018), a corpus of scientific papers labelled with scientific entities. The CoNLL04 labels are *location*, *organisation*, *people*, and *other*; SciERC labels are *task*, *method*, *evaluation metric*, *material*, *other scientific terms*, and *generic*. By removing the named entities, a sentence was split into several spans.

In the second step, we used the dependency structure of a sentence analysed by Stanford CoreNLP (Qi et al., 2018). Words that were neither in the span containing a sentence's root nor organised by the root were then removed. The assumption here was that FEs representing CFs of sentences appeared in the structural centres in the sentence dependency structures (Iwatsuki et al., 2020b).

Steps 1 and 2 work well if several named entities are contained in a sentence; otherwise, an almost full sentence is produced, which is too long to be an FE. Thus, we propose an additional filtering step that further removes non-relevant generic terms from the candidate FE spans. This is based on the assumption that each word of an FE is strongly associated with each other. Thus, the association between fragments of an FE should be strong. For instance, '*in this paper we*' and '*propose*' are strongly associated, while '*in this paper we*' and '*talk*' are not.

On the basis of this observation, we first extracted all pairs of an n-gram and its neighbour word from each candidate span obtained after Step 2. For example, pairs such as ('*in this*', '*paper*') or ('*paper we*', '*propose*') are obtained when $n = 2$. Next, for each pair, we calculated the association measures between an n-gram and a neighbour word. We used the local mutual information (LMI), which is formalised as follows:

$$\mathrm{LMI}(a, b) = f(a, b) \cdot \log \frac{p(a, b)}{p(a)p(b)}, \qquad (1)$$

where a and b denote a word, a, b denotes the co-occurrence of the words, $p(a)$ is a probability of occurrence of a, and $f(a)$ is a frequency of a in a corpus (Evert, 2005). Finally, the pairs with the top k scores were labelled as an FE. To avoid generating FEs that are too short, this third process was applied only when the length of the resulting word sequence of Step 2 was more than k words. From our preliminary experiments, we determined to use $(n, k) = (2, 7)$.

CF: Suggestion of future work
Sentence:
In the future, we plan to explore how to combine more features such as part-of-speech tags into our model.

Figure 2: Example of the database evaluation. An FE is underlined in the sentence, which has been retrieved from Cao et al. (2014).

Because FEs are assumed to be used as they are, we did not lemmatise them. Formulaicity sometimes does not allow the replacement of a word in an FE with another word or flection. For example, tenses can be section-specific (present or past): '*in this paper we proposed*' rarely occurs in the introduction sections. Formulaicity also avoids grammatical errors such as '*little researches have been done*'. Many previous studies did not lemmatise FEs (Simpson-Vlach and Ellis, 2010; Mizumoto et al., 2017; Pan et al., 2016; Esfandiari and Barbary, 2017).

3.4 Constructing CF-Labelled FE Database

We created the CF-labelled FE database using the following steps. Step 1: CF labels were assigned to each sentence in a corpus and no-CF sentences were removed. Step 2: FEs were extracted from each sentence. Step 3: Noisy FEs were filtered out. If an FE was assigned multiple CF labels, only one CF was selected by majority voting. If none of the CFs took the majority, the FE was removed. Any CF-labelled FE occurring less than three times was also removed.

We evaluated the final database from two perspectives: whether a sentence was assigned a correct label and whether an FE was useful for writing a scientific paper.

The evaluation was conducted on the AMT. A sentence and its CF label were shown to evaluators, and an FE was highlighted in the sentence (see Figure 2). The evaluators were asked whether the sentence conveyed the CF and whether the FE was useful. Each FE was annotated by five evaluators, and if it was not evaluated by all as correct or useful, it was regarded as incorrect or useless.

---

[7] `https://paperswithcode.com/sota/named-entity-recognition-ner-on-scierc`

Threshold	Precision	Recall
5/5	0.94	0.80
4/5	0.79	0.98
3/5	0.62	1.00
2/5	0.54	1.00
1/5	0.50	1.00

Table 4: Threshold indicates the number of annotators (out of five) who judged pairs of the sentence and CF label as correct.

Discipline	#Sentence
CL	612
Chem	644
Onc	600
Psy	687

Table 5: Numbers of sentences in the final dataset.

4 Results

4.1 CF-Labelled Sentence Dataset

The correct and incorrect sets consist of 55 sentences. The results of the pilot test are shown in Table 4. Accordingly, we set the threshold to 5/5 because high precision was important for creating the FE database rather than recall, and the strictest threshold did not significantly reduce the sentences. Table 5 lists the total number of sentences.

4.2 CF-Based Sentence Classification

The classification results are shown in Table 6. SciBERT worked well, which implies that this BERT-based classifier has the ability to capture CFs of sentences.

We also verified with SciBERT whether the maximum value of the softmax layer can be used as the threshold to filter out no-CF sentences. We first classified all the sentences in the corpora, and then split the classified sentences into six categories based on the maximum softmax score: [0.00, 0.60], (0.60, 0.70], (0.70, 0.80], (0.80, 0.90], (0.90, 0.99], (0.99, 1.00]. Next, we randomly sampled 100 sentences from each range, and the sentences were evaluated by five annotators on AMT. The evaluation method was the same as that used for collecting the CF-labelled sentences. The accuracy of each range is shown in Table 7. For database construction, we removed the sentences with a score of 0.80 or lower.

Discipline	I	M	R	D	Avg.
CL	0.83	0.83	1.00	0.91	0.90
Chem	0.95	0.79	0.88	0.89	0.89
Onc	0.92	0.63	0.92	0.92	0.88
Psy	0.93	0.88	0.96	0.81	0.84
ALL	0.97	0.92	0.98	0.94	0.95

Table 6: Accuracy scores of each section (**I**ntroduction, **M**ethods, **R**esults, **D**iscussion) in each discipline. The average (Avg.) indicates the macro average.

Range	Accuracy	Proportion
(0.99, 1.00]	0.69	76.1%
(0.90, 0.99]	0.67	12.4%
(0.80, 0.90]	0.74	3.7%
(0.70, 0.80]	0.51	2.4%
(0.60, 0.70]	0.51	2.1%
(0.00, 0.60]	0.43	3.3%

Table 7: Accuracy scores of each range of the maximum value of the softmax layer, and the proportion of sentences in the corpora.

4.3 Multidisciplinary Perspective

We tested whether SciBERT trained on one discipline can be applied to different disciplines. The results are shown in Table 8.

We also tested the effects of the pre-trained dataset by comparing the results of SciBERT and BERT. Table 9 and 10 show the BERT results; compared with the results shown in Table 6 and 8, the two models did not show a considerable difference.

4.4 Constructing CF-Labelled FE Database

The CF-labelled FE database was evaluated by sampling 200 FEs. The results are shown in Table 11.

Incorrect sentence–CF pairs were obtained because the classifier made errors and some sentences were not a complete sentence. An example of an incomplete sentence is ‘*of three independent experiments.*’; this was produced because of the error of sentence splitting. Examples of useful FEs are

		Evaluation			
		CL	Chem	Onc	Psy
Training	CL	0.90	0.88	0.86	0.84
	Chem	0.84	0.89	0.91	0.84
	Onc	0.75	0.89	0.88	0.82
	Psy	0.88	0.89	0.88	0.84

Table 8: Average accuracy scores. The training and evaluation datasets comprise different discipline.

Discipline	I	M	R	D	Avg.
CL	0.90	0.84	0.96	0.93	0.88
Chem	0.93	0.87	0.93	0.93	0.89
Onc	0.92	0.66	0.94	0.95	0.86
Psy	0.92	0.88	0.95	0.89	0.92

Table 9: Accuracy scores acquire by BERT classifier.

		Evaluation			
		CL	Chem	Onc	Psy
Training	CL	0.88	0.87	0.82	0.85
	Chem	0.85	0.89	0.91	0.86
	Onc	0.74	0.91	0.86	0.82
	Psy	0.87	0.92	0.88	0.92

Table 10: Average accuracy scores by BERT.

'*plays a crucial role in*' (CF: Showing the importance of the topic) and '*no significant differences were detected in*' (CF: Description of the results), while '*et al demonstrated that*' (CF: Showing background provided by past work) and '*is to use a*' (CF: Showing brief introduction to the methodology) were judged useless.

The statistics of the database are shown in Table 2. To show discipline-specific FEs, we calculated odds ratio for each CF of each discipline. Table 12 illustrates the top 5 high odds ratio FEs in the 'description of the process' CF in the introduction section. These FEs are not considered rare, as some of them occur more than a thousand times in a corpus. The differences between disciplines are relative, and these results may change if another corpus of a different discipline is added; however, preference for FEs still exists across disciplines. This reinforces the previous claim that FEs are discipline-specific (Hyland and Tse, 2007; Hyland, 2008; Durrant, 2015; Jalilifar et al., 2016). All the discipline-specific FEs are listed in Table 15 in the appendix.

		Sentence		
		Correct	Incorrect	Total
FE	Useful	130	12	142
	Useless	34	24	58
	Total	164	36	200

Table 11: Results of the evaluation of the constructed CF-labelled FE dataset.

Section: Methods		
CF: Description of the process		#
CL	we assume that the	19
	we calculate the	17
	we also use	15
	we then use the	11
	are trained using	10
Chem	was stirred at room temperature for	104
	hrms mz m h calcd	90
	were recorded on an	89
	were purchased from	642
	the mixture was stirred for	74
Onc	were purchased from	2,972
	was used for	1,129
	was purchased from	2,129
	were maintained in	527
	were used for	548
Psy	study was carried out in accordance with the	360
	gave written informed consent in accordance with the	250
	was approved by the	165
	study was approved by the ethics committee of	156
	gave written informed consent in accordance with the declaration of helsinki	111

Table 12: Examples of discipline-specific FEs. The complete list is provided in the appendix. All the FEs are lower-cased. The number of occurrences of each FE in the corpus is also shown.

5 Discussion

5.1 CF-Based Sentence Classification

The classification accuracy was quite high, and thus the results can be a good baseline for a CF-based sentence classification task. We published the dataset so that other researchers can tackle the classification task.

The no-CF detection worked fairly. From Table 7 it can be said that the maximum value is often too high; 30% of the CF labels assigned scores higher than 0.99 were incorrect. However, much lower ($\leq$ 0.80) scores tended to cause lower accuracy. Thus, this approach is useful to improve overall precision, which is more important to construct a FE-CF database than recall.

5.2 Problems in Multidisciplinary Data

We raised two questions: Can the classifier trained on one discipline be applied to other disciplines? Does the pre-training data affect the classification performance?

The results of the sentence classification imply that the SciBERT classifier trained on a dataset of one discipline can be applied to datasets of other disciplines. This mitigates the labour of creating a training dataset for all other disciplines. Therefore, we argue that to create another FE-CF database of another discipline, the CF-labelled sentence dataset we created can be used as a training dataset.

The comparison of SciBERT (Table 6) and BERT (Table 9) denied our hypothesis that the cross-discipline adaptation worked as long as the discipline was included in pre-training data. Therefore, the ability of discipline adaptation does not come from the pre-training dataset, which implies that the classifier could be used irrespective of whether a discipline is covered by the pre-training dataset.

5.3 Quality of the FE-CF Database

The results of the evaluation (Table 11) imply that if five CF-labelled FEs are retrieved, approximately three (130/200) are good FEs. Considering scenarios where users search for FEs to write a scientific paper, the selection of one FE from five candidates containing two incorrect FEs can be considered realistic.

Consider another case in which users use an FE as a query to obtain some example sentences that play the role of a specific CF. In this case, the evaluation results imply that approximately 90% (130/142) of the retrieved sentences are satisfying results. In some cases, the same FEs appear in different CF categories. For example, '*play critical roles in*' is used in 'Showing the importance of the topic (introduction)' and 'Showing background provided by past work (discussion)'. Thus, compared to the mere collection of FEs, the addition of CF labels to FEs is proved to be more helpful.

6 Conclusion

In this paper, we proposed the fully-automated construction of a CF-labelled FE database, by solving the problem of CF-based sentence classification. We carefully considered a practical case of creating a FE database of other disciplines. The experimental results showed that the proposed classification method and dataset can be utilised to construct FE databases for disciplines different from those that we used. We proposed the FE extraction method that utilised the named and scientific entity removal, dependency-structure-based word removal, and word-association-measure-based word removal. Combining the proposed methods, we finally constructed the new CF-labelled FE database. The CF-labelled sentence database and the CF-labelled FE database are available on our website[8]. We expect that the proposed database could be used by pedagogical practitioners and for computer-aided academic-writing assistance such as sentence retrieval and automated proofreading.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers 19J12466 and 18H03297.

References

Annelie Ädel. 2014. Selecting quantitative data for qualitative analysis: A case study connecting a lexicogrammatical pattern to rhetorical moves. *Journal of English for Academic Purposes*, 16:68–80.

Helen Basturkmen. 2009. Commenting on results in published research articles and masters dissertations in language teaching. *Journal of English for Academic Purposes*, 8:241–251.

Helen Basturkmen. 2012. A genre-based investigation of discussion sections of research articles in dentistry and disciplinary variation. *Journal of English for Academic Purposes*, 11:134–144.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Douglas Biber, Ulla Connor, and Thomas A. Upton. 2007. *Discourse on the Move: Using Corpus Analysis to Describe Discourse Structures*. John Benjamins Publishing.

Douglas Biber, Susan Conrad, and Viviana Cortes. 2004. If you look at ...: Lexical Bundles in University Teaching and Textbooks. *Applied Linguistics*, 25(3):371–405.

Julian Brooke, Adam Hammond, David Jacob, Vivian Tsang, Graeme Hirst, and Fraser Shein. 2015. Building a lexicon of formulaic language for language

---

[8] `https://iwa2ki.com/FE/`

learners. In *Proceedings of the 11th Workshop on Multiword Expressions*, pages 96–104, Denver, Colorado, USA. Association for Computational Linguistics.

Julian Brooke, Jan Šnajder, and Timothy Baldwin. 2017. Unsupervised acquisition of comprehensive multiword lexicons using competition in an n-gram lattice. *Transactions of the Association for Computational Linguistics*, 5:455–470.

Ziqiang Cao, Sujian Li, and Heng Ji. 2014. Joint learning of Chinese words, terms and keywords. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1774–1778, Doha, Qatar. Association for Computational Linguistics.

Viviana Cortes. 2013. The purpose of this study is to: Connecting lexical bundles and moves in research article introductions. *Journal of English for Academic Purposes*, 12(1):33–43.

Elena Cotos, Sarah Huffman, and Stephanie Link. 2015. Furthering and applying move/step constructs: Technology-driven marshalling of Swalesian genre theory for EAP pedagogy. *Journal of English for Academic Purposes*, 19:52–72.

Elena Cotos, Sarah Huffman, and Stephanie Link. 2017. A move/step model for methods sections: Demonstrating rigour and credibility. *English for Specific Purposes*, 46:90–106.

Ali Mohammadi Darabad. 2016. Move analysis of research article abstracts: A cross-disciplinary study. *International Journal of Linguistics*, 8(2):125–140.

Carmen Dayrell, Arnaldo Candido Jr., Gabriel Lima, Danilo Machado Jr., Ann Copestake, Valéria Feltrim, Stella Tagnin, and Sandra Aluisio. 2012. Rhetorical move detection in english abstracts: Multi-label sentence classifiers and their annotated corpora. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 1604–1609, Istanbul, Turkey. European Language Resources Association.

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

Philip Durrant. 2015. Lexical Bundles and Disciplinary Variation in University Students' Writing: Mapping the Territories. *Applied Linguistics*, 38(2):165–193.

Phillip Durrant and Julie Mathews-Aydınlı. 2011. A function-first approach to identifying formulaic language in academic writing. *English for Specific Purposes*, 30(1):58–72.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *the 24th European Conference on Artificial Intelligence*, pages 2006–2013, Santiago de Compostela, Spain. IOS Press.

Rajab Esfandiari and Fatima Barbary. 2017. A contrastive corpus-driven study of lexical bundles between english writers and persian writers in psychology research articles. *Journal of English for Academic Purposes*, 29:21–42.

Stefan Evert. 2005. *The statistics of word occurrences*. Ph.D. thesis, Universität Stuttgart.

James Fiacco, Elena Cotos, and Carolyn Rosé. 2019. Towards enabling feedback on rhetorical structure with neural sequence models. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 310–319, Tempe, Arizona, USA. Association for Computing Machinery.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *The Fifth International Conference on Learning Representations*, Toulon, France.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Kenji Hirohata, Naoaki Okazaki, Sophia Ananiadou, and Mitsuru Ishizuka. 2008. Identifying sections in scientific abstracts using conditional random fields. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*, pages 381–388, Hyderabad, India. Asian Federation of Natural Language Processing.

Ken Hyland. 2008. As can be seen: Lexical bundles and disciplinary variation. *English for Specific Purposes*, 27(1):4–21.

Ken Hyland and Polly Tse. 2007. Is there an "academic vocabulary?". *TESOL Quarterly*, 41(2):235–253.

Kenichi Iwatsuki, Florian Boudin, and Akiko Aizawa. 2020a. An evaluation dataset for identifying communicative functions of sentences in English scholarly papers. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1705–1713, Marseille, France. European Language Resources Association.

Kenichi Iwatsuki, Florian Boudin, and Akiko Aizawa. 2020b. Extraction and evaluation of formulaic expressions used in scholarly papers. arXiv:2006.10334.

Alireza Jalilifar, Seyed Mohammad Ghoreishi, and Seyed Abbas Emam Roodband. 2016. Developing an inventory of core lexical bundles in english research articles: a cross-disciplinary corpus-based study. *Journal of World Languages*, 3(3):184–203.

Budsaba Kanoksilapatham. 2005. Rhetorical structure of biochemistry research articles. *English for Specific Purposes*, 24:269–292.

Hannah Kermes. 2012. A methodology for the extraction of information about the usage of formulaic expressions in scientific texts. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2064–2068, Istanbul, Turkey. European Language Resources Association.

Jason Miin Hwa Lim. 2006. Method sections of management research articles: A pedagogically motivated qualitative study. *English for Specific Purposes*, 25:282–309.

Jason Miin Hwa Lim. 2010. Commenting on research results in applied linguistics and education: A comparative genre-based investigation. *Journal of English for Academic Purposes*, 9:280–294.

Dilin Liu. 2012. The most frequently-used multi-word constructions in academic written english: A multi-corpus study. *English for Specific Purposes*, 31:25–35.

Yuanchao Liu, Xin Wang, Ming Liu, and Xiaolong Wang. 2016. Write-righter: An academic writing assistant system. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 4373–4374, Phoenix, Arizona, USA. Association for the Advancement of Artificial Intelligence.

Rosa Lorés. 2004. On RA abstracts: from rhetorical structure to thematic organisation. *English for Specific Purposes*, 23(3):280–302.

Xiaofei Lu, Jungwan Yoon, and Olesya Kisselev. 2018. A phrase-frame list for social science research article introductions. *Journal of English for Academic Purposes*, 36:76–85.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Ron Martinez and Norbert Schmitt. 2012. A phrasal expressions list. *Applied Linguistics*, 33(3):299–320.

Sayako Maswana, Toshiyuki Kanamaru, and Akira Tajino. 2015. Move analysis of research articles across five engineering fields: What they share and what they do not. *Ampersand*, 2:1–11.

Atsushi Mizumoto, Sawako Hamatani, and Yasuhiro Imao. 2017. Applying the bundle–move connection approach to the development of an online writing support tool for research articles. *Language Learning*, 67(4):885–921.

John Morley. n. d. Academic phrasebank.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Ismet Ozturk. 2007. The textual organisation of research article introductions in applied linguistics: Variability within a single discipline. *English for Specific Purposes*, 26(1):25–38.

Fan Pan, Randi Reppen, and Douglas Biber. 2016. Comparing patterns of l1 versus l2 english academic professionals: Lexical bundles in telecommunications research journals. *Journal of English for Academic Purposes*, 21:60–71.

Matthew Peacock. 2002. Communicative moves in the discussion section of research articles. *System*, 30:479–497.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Naser Rashidi and Hussein Meihami. 2018. Informetrics of scientometrics abstracts: a rhetorical move analysis of the research abstracts published in scientometrics journal. *Scientometrics*, 116:1975–1994.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Fahimeh Saboori and Mohammad Reza Hashemi. 2013. A cross-disciplinary move analysis of research article abstracts. *International Journal of Language Learning and Applied Linguistics World*, 4(4):483–496.

Rita Simpson-Vlach and Nick C. Ellis. 2010. An Academic Formulas List: New Methods in Phraseology Research. *Applied Linguistics*, 31(4):487–512.

Tasanawan Soonklang. 2016. Move classification in scientific abstracts using linguistic features. In *The Eleventh International Symposium on Natural Language Processing*, Phranakhon Si Ayutthaya, Thailand.

John Malcolm Swales. 1990. *Genre Analysis: English in academic and research settings*. Cambridge University Press.

John Malcolm Swales. 2004. *Research Genres: Explorations and Applications*. Cambridge University Press.

Simone Teufel. 1999. *Argumentative Zoning: Information Extraction from Scientific Text*. Ph.D. thesis, University of Edinburgh.

Simone Teufel, Advaith Siddharthan, and Colin Batchelor. 2009. Towards domain-independent argumentative zoning: Evidence from chemistry and computational linguistics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1493–1502, Singapore. Association for Computational Linguistics.

Benet Vincent. 2013. Investigating academic phraseology through combinations of very frequent words: A methodological exploration. *Journal of English for Academic Purposes*, 12(1):44–56.

A Dataset and Databases

On our website[9], we published the following dataset and databases:

1. The CF-labelled sentence dataset for training and evaluation,
2. The CF-labelled sentence database, which was constructed by applying SciBERT classifier to every sentence in the corpora we used, and
3. The CF-labelled FE database, which was constructed by applying the proposed FE extraction method to the CF-labelled sentence database.

These data were formatted in tab-separated text. In the CF-labelled sentence dataset, a line consists of an ID and a sentence. In the CF-labelled sentence database, a line consists of a sentence ID (from the corpora), an ID, the maximum softmax value, and a sentence. In the CF-labelled FE database, a line consists of a CF, an FE, and the number of appearance in the corpus.

B CF Set

Table 13 lists the CF we used. The ID in the table corresponds to the ID used in the sentence dataset and database.

C General and Discipline-Specific FEs

General FEs are FEs that appear commonly in multiple disciplines. We calculated the average rank of each FE and Table 14 lists the top-5 general FEs for each CF. For most of the CFs, general FEs were not found. We also calculated the odds ratio and Table 15 lists the top-5 discipline-specific FEs for each CF. Some CFs did not happen in a corpus.

---

[9] https://iwa2ki.com/FE/

Section	ID	CF
introduction	0	Showing the importance of the topic
introduction	1	Showing the main problem in the field
introduction	2	Showing what is already done in the past work
introduction	3	Showing controversy within the field
introduction	4	Showing limitation or lack of past work
introduction	5	Showing the aim of the paper
introduction	6	Showing brief introduction to the methodology
introduction	7	Showing the importance of the research
introduction	8	Showing the limitation of the research
introduction	9	Showing the outline of the paper
introduction	10	Showing explanation or definition of terms or notations
method	0	Showing methodology used in past work
method	1	Showing reasons why a method was adopted or rejected
method	2	Using methods used in past work
method	3	Showing the characteristics of samples or data
method	4	Showing criteria for selection
method	5	Description of the process
result	0	Restatement of the aim or method
result	1	Reference to tables or figures
result	2	Description of the results
result	3	Describing interesting or surprising results
result	4	Comparison of the results
result	5	Summary of the results
discussion	0	Showing background provided by past work
discussion	1	Restatement of the results
discussion	2	Unexpected outcome
discussion	3	Comparison of the results and past work
discussion	4	Explanation for findings
discussion	5	Suggestion of hypothesis
discussion	6	Implications of the findings
discussion	7	Comments on the findings
discussion	8	Suggestion of future work

Table 13: CF list.

CF	FE
Section: Introduction	
Showing the importance of the topic	plays an important role in
Showing the importance of the topic	play an important role in
Showing the importance of the topic	also plays an important role in
Showing the importance of the topic	is related to
Showing the importance of the topic	plays a crucial role in
Showing limitation or lack of past work	to the best of our knowledge there is no
Showing limitation or lack of past work	to the best of our knowledge no
Showing the importance of the research	to the best of our knowledge this is the first
Showing brief introduction to the methodology	is based on
Showing brief introduction to the methodology	is based on the
Showing brief introduction to the methodology	are based on
Showing brief introduction to the methodology	is to use
Showing the outline of the paper	the paper is organized as follows

CF	FE
Showing the outline of the paper	this paper is organized as follows
Showing the outline of the paper	the paper is structured as follows
Showing the outline of the paper	this paper is structured as follows
Showing the outline of the paper	the rest of the paper is structured as follows
Section: Results	
Description of the results	there was no significant difference in
Description of the results	we found that the
Description of the results	there was no significant difference between the
Description of the results	there was no significant difference between
Description of the results	we found that
Restatement of the aim or method	we used the
Restatement of the aim or method	we compared the
Restatement of the aim or method	we used a
Restatement of the aim or method	we performed a
Restatement of the aim or method	was performed using
Reference to tables or figures	as shown in
Describing interesting or surprising results	it is interesting to note that
Describing interesting or surprising results	it is interesting to note that the
Summary of the results	these results suggest that
Summary of the results	this suggests that
Summary of the results	this suggests that the
Summary of the results	this indicates that the
Section: Discussion	
Restatement of the results	we found that the
Restatement of the results	we found that
Restatement of the results	it is interesting to note that
Restatement of the results	it is worth noting that the
Restatement of the results	it is important to note that the
Suggestion of hypothesis	our results suggest that
Explanation for findings	can be explained by the fact that
Explanation for findings	this is due to the fact that
Unexpected outcome	it is not surprising that the
Implications of the findings	this raises the possibility that

Table 14: General FEs.

CL	Chem	Onc	Psy
Section: Introduction			
CF: Showing limitation or lack of past work			
to the best of our knowledge there	to the best of our knowledge there	however * role * remains unclear	to our knowledge only one study
few attempts have been made to	to the best of our knowledge there have been	however * role * remains unknown	best of our knowledge no study has
there has been little work on	to the best of our knowledge there are few	remains to be elucidated	only a few studies have investigated
there is no	has not been reported	however * mechanism * remains unclear	little attention has been paid to

CL	Chem	Onc	Psy
it is not clear how to	to the best of our knowledge the	has not been reported	studies * are scarce
CF: Showing the importance of the topic			
is an important task in	has been used in	is one of the most common	it is important to note that
there has been a growing interest in	is one of the most popular	et al reported that	is that the
in this work we are interested in	belongs to the family	et al found that	there is a growing body of research
has received a lot of attention	is used as a	et al showed that	is defined as a * cite-
the main contribution of this paper is	is widely distributed in	in this study we found that	refers to the extent to which
CF: Showing controversy within the field			
the state of the art in	it should be noted that the	role * is controversial	is still a matter of debate
this research was partially supported by	it should be noted that	is still a matter of debate	is an open question
this work was supported by the	it is important to mention that	this has led to the suggestion that	the question arises as to whether
the current state of the art	it is worth mentioning that the	is still under debate	are still a matter of debate
this work was done while the	it should be noted however that		it remains an open question whether
CF: Showing the aim of the paper			
in this paper we focus on	the aim of this study was to	the aim of this study was to	the aim of the present study was to
in this paper we propose a novel	objective of this study was to	purpose of this study was to	the aim of the present study was to investigate
in this paper we propose a	the aim of this work was to	aim of this study was to investigate the	the aim of this study was to
in this paper we present an	in this paper we report the	the aim of the present study was to	the aim of the present study was to examine
in this paper we propose an	the aim of the present study was to	purpose of this study was to investigate the	the purpose of the present study was to
CF: Showing the importance of the research			
we propose a novel	to our knowledge this is the first report on	we show for the first time that	to our knowledge the present study is the first
we propose a new	to the best of our knowledge this is the first time	in this study we * for the first time that	to our knowledge this is the first study to
to the best of our knowledge we are the first	to our knowledge this is the first time that	we demonstrate for the first time that	best of our knowledge the present study is
we present a novel	was reported in * cite-	we report for the first time that	should be able to discriminate between
we present a new	was reported in	here we show for the first time that	as far as we are aware * is the first
CF: Showing the limitation of the research			

CL	Chem	Onc	Psy
is not limited to	beyond the scope of this review		beyond the scope of this paper
			beyond the scope of this article
			in the absence of any * that could be
			is the focus of this study
CF: Showing brief introduction to the methodology			
page numbers and proceedings footer are added	were characterized by	in this study we investigated the	in the present study we focus on
we evaluate our	were also investigated	in the present study we investigated the	in the present study we investigated
cite- proposed a	was used as the	in this study we aimed to	cite- cite- cite- cite-
section 3 describes the	et al cite- used	in this study we evaluated the	et al cite- used
we propose a	was used as a	in this study we investigated the role of	the present study was designed to
CF: Showing the main problem in the field			
is that the	is one of the most common	is the leading cause of	is one of the most common
is a fundamental problem in	is the most common form of	is the second leading cause of	there are two reasons for this
is the lack of	is one of the most serious	is the third leading cause of	however is that
there are two main	therefore it is necessary to develop	is the leading cause of death	this is one of the reasons why the
the main contribution of this work is	is one of the most frequent	there is an urgent need to identify	for this reason it is necessary to
CF: Showing explanation or definition of terms or notations			
we call this	is defined as a * cite-	are defined as * cite-	refers to * cite-
we call such	are defined as * cite-	is also called	refers to the * cite-
is called a	is defined as	is defined as a * cite-	this is referred to as the
is called the		hereafter referred to as	is often referred to as
we denote by		is defined as * cite-	refer to * cite-
CF: Showing what is already done in the past work			
have been applied to	have been used as	cite- cite- cite-	et al cite-
have been proposed for	have been isolated from	accumulating evidence suggests that	and * cite-
have been developed for	et al reported that	increasing evidence suggests that	et al cite- found that
have been proposed	have been used in	several lines of evidence suggest that	for example it has been shown that
there have been a number of	have been reported cite- cite- * cite- cite- cite-	a growing body of evidence suggests that	it has been argued that the
CF: Showing the outline of the paper			

CL	Chem	Onc	Psy
of this paper is organized as follows			the rest of the paper is organized as follows
the rest of * paper is organized as follows			the remainder of the paper is organized as follows
remainder of this paper is structured as follows			the rest of this paper is organized as follows
paper is organized as follows section 2			can be divided into
the contributions of this paper are as follows			remainder of this paper is organized as follows
Section: Methods			
CF: Using methods used in past work			
we propose a	the title compound was prepared from	was performed as previously described cite-	was calculated using the
is based on	characterization data is in accordance with that reported in cite-	were performed as previously described cite-	was based on the
is based on the	was calculated according to the following equation	was performed as described previously cite-	was adapted from cite-
in this section we describe our	was calculated using the following equation	were performed as described previously cite-	is based on the
is based on a	was prepared according to the general procedure	conducted in accordance with the * according to the	was developed by cite-
CF: Showing reasons why a method was adopted or rejected			
is used for	was used for	was used to analyze the relationship between	was used to assess
are used for	was used as a positive control	was used to evaluate the association between	was used to measure
can be used for	was used for the	was used to analyze the correlation between	et al cite- was used to assess
is used as the	was used as the	was used to assess the association between	version * was used
is that the	were used for	were used to estimate	was used as a measure of
CF: Showing the characteristics of samples or data			
are shown in table 1	are listed in cite-	experiments were repeated at least three times	all participants had
are added to the	are shown in cite-	all experiments were performed in triplicate	had normal or corrected to normal vision
submission are marked with an asterisk	used in this study are listed in cite-	of at least three independent experiments	ranged in age from 18 to
are listed in table 1	were used as positive controls	each experiment was performed in triplicate	participants * were excluded

CL	Chem	Onc	Psy
are shown in table 2	s singlet d doublet t triplet	experiment was repeated at least three times	all participants were native speakers of
CF: Showing methodology used in past work			
we use two	include * cite-	is how to formulate a	et al cite-
we adopt a	has been routinely and widely used in	has been widely recognized and increasingly * to examine the quality of * see eg cite-	and * cite-
we use the following	cite- cite- cite-	here the * ratio	has been shown to have good
we consider two	cite- lists the	described in cite- * there is no need to repeat	it should be noted that the
there are two	is one of the most widely used	are needed as elaborated in cite-	has been shown to be a
CF: Showing criteria for selection			
figure 1 the	were maintained in	p 005 was considered statistically significant	was defined as
is shown in figure 1	was defined as the amount of enzyme	p values 005 were considered statistically significant	was defined as the
figure 2 the	was defined as the lowest concentration of	005 was considered to be statistically significant	were selected from the * cite-
figure 1 a	in accordance with the * care and use of laboratory animals	p 005 was considered significant	is defined as
is a set of	cells were cultured in	value * was considered statistically significant	were defined as
CF: Description of the process			
we assume that the	was stirred at room temperature for	were purchased from	study was carried out in accordance with the
we calculate the	hrms mz m h calcd	was used for	gave written informed consent in accordance with the
we also use	were recorded on an	was purchased from	was approved by the
we then use the	were purchased from	were maintained in	study was approved by the ethics committee of
are trained using	the mixture was stirred for	were used for	gave written informed consent in accordance with the declaration of helsinki
Section: Results			
CF: Comparison of the results			
we compare our	cite- compares the	analysis of the	at each measurement point showed that
we compare the	comparison * is shown in cite-	comparison of the	in this section we present the

CL	Chem	Onc	Psy
table 3 comparison of	a comparison of the	comparison of * revealed	conditions there was an effect of condition with
table 2 comparison of	summary * is presented in cite-	comparison of * cite-	it can be seen that
table 1 comparison of	cite- shows the comparison of	analysis of the * cite-	inspection * indicated a significant influence of
CF: Reference to tables or figures			
table 2 shows the	cite- shows the	as shown in figure cite-	figure cite- shows the
table 1 shows the	are shown in cite-	figure cite- shows the	are presented in table cite-
table 3 shows the	it can be seen that the	are shown in table cite-	table cite- shows the
results are shown in table 2	the results are shown in cite-	are summarized in table cite-	are shown in table cite-
figure 2 shows the	is shown in cite-	are shown in figure cite-	table cite- presents the
CF: Description of the results			
achieves the highest	et al cite- reported that the	it has been reported that	revealed a significant main effect of
performs better than	indicated the presence of	our results showed that	there was a significant main effect of
significantly outperform the	was determined to be	was observed in	showed a significant main effect of
is significantly better than the	was confirmed by	we have previously shown that	there was a significant interaction between
outperforms all other	was assigned to the	showed * figure cite-	there was a main effect of
CF: Describing interesting or surprising results			
this is due to the fact that	it is worth noting that the	interestingly we found that	et al cite-
is due to the fact that the	it is important to mention that	interestingly we observed that	it should be noted that the
it should be noted that the	it is worth mentioning that the	indeed we found that	and * cite-
we call this	best of our knowledge this is the first report	interestingly we found that the	however it is important to note that
this can be explained by the fact that	it is important to note that	moreover * figure cite-	et al cite- found
CF: Summary of the results			
this shows that our	the results indicated that	taken together these data demonstrate that	this indicates that * likely
this result shows that the proposed	these results are in agreement with those	taken together these results demonstrated that	this pattern is consistent with the
from these results we can conclude that	these results are in accordance with	taken together these findings indicate that	therefore hypothesis 3 is supported
this suggests that for	this indicated that the	taken together our data suggest that	this suggests that during both meditation conditions saline
these results demonstrate that the proposed	the results show that the	these results suggest that * promotes	this suggests that the * had

CL	Chem	Onc	Psy
CF: Restatement of the aim or method			
we use the	was used as a positive control	were treated with	was conducted on
we use a	was reacted with	was confirmed by	was conducted on the
we evaluate our	was used as the positive control	investigate * we performed	were submitted to a
we evaluate the	were used as positive controls	determine * we performed	we conducted a
we use the same	were evaluated for their	we next examined the effect of	was conducted with
Section: Discussion			
CF: Comparison of the results and past work			
material is based upon work supported in part by	et al showed that	et al cite- also reported that	is in line with previous research
this material is based in part on research sponsored by the	et al found that	our results also showed that	is in line with previous studies
material is based upon work supported by the	et al demonstrated that	these findings are consistent with previous reports	these findings are in line with
this is also the case for	et al cite-	our results are consistent with previous	is in line with previous findings
this paper is * upon work supported in part by	et al indicated that	our results are consistent with those	this finding is in line with
CF: Implications of the findings			
		these findings raise the possibility that	these findings have important implications for
		this suggests the possibility that	the present study contributes to the
		these results raise the possibility that	it is assumed that
		highlights the importance of	limit the generalizability of our findings
		there are several important implications in this	our findings also have implications for
CF: Restatement of the results			
we would like to thank	was found to be	to the best of our knowledge this is the first	it is important to note that
the experimental results show that our	were characterized by	et al found that	the results showed that
experimental results show that the proposed	was found to be the most potent	to our knowledge this is the first	cite- found that
in this paper we have shown that	were tested for their ability to inhibit	in this study we found that	we did not find any significant
i would like to thank	were evaluated for their	in this study we demonstrated that	it is important to note * the current study

CL	Chem	Onc	Psy
CF: Showing background provided by past work			
in this paper we have presented a	include * cite-	et al reported that	et al cite-
in this paper we presented a	in the present study	it has been reported that	cite- cite- cite-
in this paper we propose a	it has been reported that	et al demonstrated that	it has been argued that
in this paper we proposed a	will be reported in due course	it was reported that	it has been argued that the
we presented a	et al reported that	it has been shown that	and * cite-
CF: Suggestion of hypothesis			
we have presented a	best of our knowledge this is the first report	in conclusion our study demonstrates that	this finding supports the notion that
we have presented a novel	it can be concluded that the	these data suggest that	suggests that the
we have proposed a	in summary we have developed a	in conclusion our data suggest that	the present findings suggest that the
we have presented a new	in conclusion we have developed a	our results showed that	this finding supports the idea that
we have shown that it is possible	it is known that	in summary our results indicate that	the present study provides the first
CF: Comments on the findings			
we have presented a simple and effective	was successfully applied to the	is a promising	we used a
we achieved an	in summary we have successfully developed	may be a promising strategy for	is that the
we expect the	was successfully applied for the	is a promising strategy for	on the one hand * on the other hand
acknowledgements we are grateful to	has been successfully applied to the	might be a promising strategy for	declares that despite being affiliated to * same institution as
has several advantages	has been successfully applied to a	in the present study we successfully	the aim of the present study was to
CF: Explanation for findings			
this can be explained by the fact that the	can be attributed to the	may be involved in	it should be noted that the
this is due to the fact that the	can be explained by the presence of	therefore it is possible that the	however it should be noted that
we believe that this is due to the	can be attributed to the presence of	however * mechanism * is unclear	it should be noted however that
one reason for this is that	this could be explained by the fact that	this may explain why	it is also possible that the
this can be explained by the fact that	may be due to the presence of	mechanism * is unknown	could be due to the fact that the
CF: Suggestion of future work			
in the future we plan to	studies * are in progress	this study has several limitations	it would be interesting to compare
in the future we would like to	are currently underway in our laboratory	our study has several limitations	further research is needed to clarify

CL	Chem	Onc	Psy
in future work we plan to	studies * are currently underway	this study has some limitations	further research is needed to determine
in future work we would like to	are in progress in our laboratory	our study has some limitations	beyond the scope of this paper
as future work we plan to	are in progress and will be reported in due course	it remains to be determined whether	it would be interesting to examine whether
CF: Unexpected outcome			
government is authorized to reproduce and distribute reprints for		therefore it is not surprising that	this was not the case
this is not surprising given that		it is not surprising that	however this was not the case
what are kinds of		unexpectedly we found that	this was not observed
it ports easily to new language pairs the		thus it is not surprising that	this was not the case for
is slightly different from * official one because * this figure if		interestingly we observed that	this was not the case in the present study

Table 15: Discipline-specific FEs.

Regulatory Compliance through Doc2Doc Information Retrieval: A case study in EU/UK legislation where text similarity has limitations

Ilias Chalkidis [†][‡]   **Manos Fergadiotis** [†][‡]   **Nikolaos Manginas** [†]

Eva Katakalou [♭][*]   **Prodromos Malakasiotis** [†][‡]

[†] EY AI Centre of Excellence in Document Intelligence, NCSR “Demokritos”
[‡] Department of Informatics, Athens University of Economics and Business
[♭] Department of International, European and Area Studies, Panteion University

`[Ilias.Chalkidis, Fergadiotis.Manos]@ey.com`
`[Nikolaos.Manginas, Prodromos.Malakasiotis]@ey.com`
`[ichalkidis, mfergadiotis]@iit.demokritos.gr`
`[nmanginas, pmalakasiotis]@iit.demokritos.gr`
`e.katakalou@panteion.gr`

Abstract

Major scandals in corporate history have urged the need for *regulatory compliance*, where organizations need to ensure that their controls (processes) comply with relevant laws, regulations, and policies. However, keeping track of the constantly changing legislation is difficult, thus organizations are increasingly adopting Regulatory Technology (RegTech) to facilitate the process. To this end, we introduce *regulatory information retrieval* (REG-IR), an application of *document-to-document information retrieval* (DOC2DOC IR), where the query is an entire document making the task more challenging than traditional IR where the queries are short. Furthermore, we compile and release two datasets based on the relationships between EU directives and UK legislation. We experiment on these datasets using a typical two-step pipeline approach comprising a pre-fetcher and a neural re-ranker. Experimenting with various pre-fetchers from BM$_{25}$ to k nearest neighbors over representations from several BERT models, we show that fine-tuning a BERT model on an in-domain classification task produces the best representations for IR. We also show that neural re-rankers underperform due to *contradicting* supervision, i.e., similar query-document pairs with opposite labels. Thus, they are biased towards the pre-fetcher's score. Interestingly, applying a date filter further improves the performance, showcasing the importance of the time dimension.

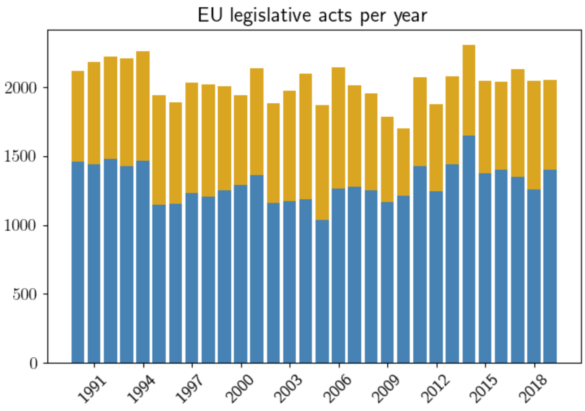


Figure 1: Number of legislative acts issued by the EU per year. The gold color of the bars indicates how many of the published acts are amendments to older ones.

1 Introduction

Major scandals in corporate history, from Enron to Tyco International, Olympus, and Tesco,[1] have led to the emergence of stricter regulatory mandates and highlighted the need for *regulatory compliance* where organizations need to ensure that they comply with relevant laws, regulations, and policies (Lin, 2016). However, keeping track of the constantly changing legislation (Figure 1) is hard, thus organizations are increasingly adopting Regulatory Technology (RegTech) to facilitate the process.

Typically, a compliance regimen includes three distinct but related types of measures, *corrective*, *detective*, and *preventive* (Sadiq and Governatori,

---

[*] The contribution of Ms. Eva Katakalou was restricted to the creation and the validation of the datasets as well as to the authoring of the corresponding parts of the manuscript.

[1] `www.theguardian.com/business/2015/jul/21/the-worlds-biggest-accounting-scandals-toshiba-enron-olympus`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3498–3511
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2015). Corrective measures are usually undertaken when new regulations are introduced to update existing controls. Detective measures, ensure "after-the-fact" compliance, i.e., following a procedure, a manual or automated check is carried out, to ensure that every step of the procedure complied with the corresponding regulations. Finally, preventive measures ensure compliance "by design", i.e., during the creation of new controls. All types of measures include an underlying information retrieval (IR) task, where laws need to be retrieved given a control or vice versa. We identify two use cases:

1. *Given a new law retrieve all the controls of the organization affected by this law*. The organization can then apply corrective measures to ensure compliance for these controls.
2. *Given a control retrieve all relevant laws the control should comply with.* This is useful for ensuring compliance after a procedure has been carried out (detective measures) or when creating new controls (preventive measures).

Regulatory information retrieval (REG-IR), similarly to other applications of *document-to-document* (DOC2DOC) IR, is much more challenging than traditional IR where the query typically contains a few informative words and the documents are relatively small (Table 1). In DOC2DOC IR the query is a long document (e.g., a regulation) containing thousands of words, most of which are uninformative. Consequently, matching the query with other long documents where the informative words are also sparse, becomes extremely difficult.

Although legislation is available, organizations' controls are strictly private and very hard to obtain. Fortunately, the European Union (EU) has a legislation scheme analogous to regulatory compliance for organizations. According to the Treaty on the Functioning of the European Union (TFEU),[2] all published EU *directives* must take effect at the national level. Thus, all EU member states must adopt a law to transpose a newly issued directive within the period set by the directive (typically 2 years). Notably, the United Kingdom (UK) having a high compliance level with the EU (Figure 2),[3] is a good test-bed for REG-IR. Thus we compile and release two datasets for REG-IR, EU2UK and UK2EU, containing EU directives and UK regulations, which can serve both as queries and documents under the ground truth assumption that a UK law is relevant to the EU directives it transposes and vice versa.

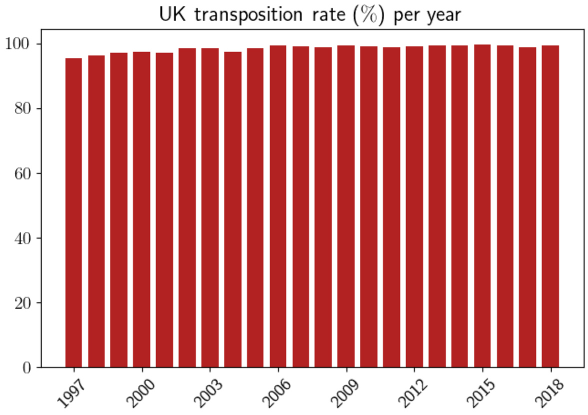


Figure 2: The percentage of EU directives transposed by UK legislation per year. Over 98% of the published EU directives have been transposed.

Dataset	Domain	$\tilde{q}$	$\tilde{d}$
IR datasets in the literature			
TREC ROBUST (Voorhees, 2005)	News	3 / 14	254
BIOASQ (Tsatsaronis et al., 2015)	Biomedical	9	197
IR datasets with verbose queries			
GOV2 (Clarke et al., 2004)	Web	11 / 57	682
WT10G (Chiang et al., 2005)	Web	11 / 35	457
Regulatory Compliance datasets			
EU2UK (ours)	Law	2,642	1,849
UK2EU (ours)	Law	1,849	2,642

Table 1: Statistics for query and document length for IR datasets used in literature.

Since REG-IR is a new task, our starting point is the two-step pipeline approach followed by most modern neural information retrieval systems (Guo et al., 2016; Hui et al., 2017; McDonald et al., 2018). First, a conventional IR system (*pre-fetcher*) retrieves the k most prominent documents. Then a neural model attempts to rank relevant documents higher than irrelevant ones. In most approaches, the pre-fetcher is based on Okapi BM$_{25}$ (Robertson et al., 1995), a bag-of-words scoring function that does not consider possible synonyms or contextual information. To overcome the first limitation, we follow Brokos et al. (2016) who employed k nearest neighbors over tf-idf weighted centroids of word embeddings, without however improving the results, probably because the centroids are noisy considering many uninformative words. Furthermore, we employ BERT (Devlin et al., 2019) to extract contextualized representations for queries and documents but again the results are worse than BM$_{25}$. We also experiment with S-BERT (Reimers

---

[2] Articles 291 (1) and 288 paragraph 3.

[3] Data for Figures 1 and 2 obtained from `ec.europa.eu/internal_market/scoreboard/performance_by_governance_tool/eu_pilot`.

Query: DIRECTIVE 2006/66/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 6 September 2006 on batteries and accumulators and waste batteries and accumulators and repealing Directive 91/157/EEC

BM$_{25}$ rank	Relevant	Document title
1	No	The Batteries and Accumulators (Placing on the Market) (Amendment) Regulations 2012
2	No	The Batteries and Accumulators (Containing Dangerous Substances) (Amendment) Regulations 2000
3	No	The Batteries and Accumulators (Placing on the Market) (Amendment) Regulations 2015
4	No	The Batteries and Accumulators (Containing Dangerous Substances) Regulations 1994
5	No	The Waste Batteries and Accumulators (Amendment) Regulations 2015
6	Yes	The Waste Batteries and Accumulators Regulations 2009
12	Yes	The Batteries and Accumulators (Placing on the Market) Regulations 2008

Table 2: Example from the EU2UK dataset where the retrieved UK laws are ranked by BM$_{25}$. The top-5 documents seem similar to the query but are not relevant. Documents ranked 1st, 3rd, and 5th are amendments of the relevant documents, i.e., UK laws that transpose the query.

and Gurevych, 2019) and LEGAL-BERT (Chalkidis et al., 2020), a model specialized in the legal domain. Both models perform better than BERT but are still worse than or comparable to BM$_{25}$. The inability of BERT-based models motivated us to find an auxiliary task that will result in better representations for REG-IR. Following Chalkidis et al. (2019), we fine-tune BERT to predict EUROVOC concepts that describe the core subjects of each text. As expected this model (C-BERT) is the best pre-fetcher by a large margin in EU2UK, while being comparable to BM$_{25}$ in UK2EU. To summarize, our contributions are:

(a) We introduce REG-IR, an application of DOC2DOC IR, which is a new family of IR tasks, where both queries and documents are long typically containing thousands of words.

(b) We compile and release the two first publicly available datasets, EU2UK and UK2EU, suitable for REG-IR and DOC2DOC IR in general.[4]

(c) We show that fine-tuning BERT on an in-domain classification task produces the best document representations with respect to IR and improves pre-fetching results.

2 Datasets curation

2.1 Data sources

EU/UK Legislation: We have downloaded approx. 56K pieces of EU legislation (approx. 3.9K directives), from the EURLEX portal.[5] EU laws are 2,642 words long on average and are structured in three major parts: the *title* (Table 2, query), the *recitals* consisting of references in the legal background of the act, and the *main body*. We have also downloaded approx. 52K UK laws, publicly available from the official UK legislation portal.[6] UK laws are 1,849 words long on average and contain the *title* (Table 2, document title) and the *main body*.

Transpositions: We have retrieved all transposition relations (approx. 3.7K) between EU directives and UK laws from the CELLAR database. CELLAR only provides the mapping between the CELLAR ids of EU directives and the title of each UK law. Therefore we aligned the CELLAR ids with the official UK ids based on the law title.[7] One or more UK laws may transpose one or more EU directives.

2.2 Datasets compilation

Let $\mathcal{E}, \mathcal{U}$ be the sets of EU directives and UK laws, respectively. We define REG-IR as the task where the query q is a document, e.g, an EU directive, and the objective is to retrieve a set of relevant documents, $\mathcal{R}_q$, from the pool of all available documents, e.g., all UK laws. We create two datasets:

EU2UK: $q \in \mathcal{E}, \mathcal{R}_q = \{r_i \colon r_i \in \mathcal{U}, r_i \xrightarrow{\text{transposes}} q\}$.

UK2EU: $q \in \mathcal{U}, \mathcal{R}_q = \{r_i \colon r_i \in \mathcal{E}, q \xrightarrow{\text{transposes}} r_i\}$.

Table 3 shows the statistics for the two datasets, which are split in three parts, *train*, *development*, and *test*, retaining a chronological order for the queries. EU2UK has a much larger pool of available documents than UK2EU (52.5K vs. 3.9K) which may impose an extra difficulty during retrieval. More importantly, the average number of relevant documents per query is small (at most 2) for both datasets, as our ground truth assumption is strict, i.e., relevant documents are those linked to the query with a transposition relation. Also, EU legislation is frequently amended (Figure 1) which also

---

[4] The datasets are available at `https://archive.org/details/eacl2021_regir_datasets`.

[5] `eur-lex.europa.eu`

[6] `legislation.gov.uk`

[7] See Appendix A for details on the dataset curation.

Dataset	Documents in pool	Train		Development		Test	
		Queries	Avg. relevant	Queries	Avg. relevant	Queries	Avg. relevant
EU2UK	52,515	1,400	1.79	300	2.09	300	1.74
UK2EU	3,930	1,500	1.90	300	1.46	300	1.29

Table 3: Detailed statistics for EU2UK and UK2EU. Both datasets have relatively small number of relevant documents while EU2UK has also large pool which may impose extra difficulties in the retrieval.

imposes difficulty in the retrieval task. Let $d_1 \in \mathcal{E}$ be a directive transposed by $u_1 \in \mathcal{U}$ and $d_2 \in \mathcal{E}$ be a directive amending d_1. The UK must adopt a law, u_2, to transpose d_2. Both d_2 and u_2 cover similar concepts to those of d_1 (d_2 is an amendment and u_2 must comply with d_2), but, strictly speaking u_2 is relevant only to d_2. Table 2 shows an example from EU2UK, where the top-5 documents seem very similar to the query but are not considered relevant. Note that the documents ranked 1st, 3rd and 5th, are amendments of the relevant documents.

3 IR pipelines

Modern neural IR systems usually follow a two-step pipeline approach. First, a conventional IR system (*pre-fetcher*) retrieves the top-k most prominent documents aiming to maximize its recall. Then a neural model attempts to re-rank the documents by scoring relevant higher than irrelevant ones. While this configuration is widely adopted in literature, the re-ranking step could be omitted provided an effective pre-fetching mechanism, i.e., the pre-fetcher will act as an end-to-end IR system.

3.1 Document pre-fetching

Okapi BM$_{25}$ (Robertson et al., 1995) is a bag-of-words scoring function estimating the relevance of a document d to a query q, based on the query terms appearing in d, regardless their proximity within d:

$$\sum_{i=1}^{n} \mathrm{idf}(q_i) \cdot \frac{\mathrm{tf}(q_i,d)\cdot(k_1+1)}{\mathrm{tf}(q_i,d)+k_1\cdot\left(1-b+b\cdot\frac{L}{\bar{L}}\right)} \quad (1)$$

where q_i is the i-th query term, with $\mathrm{idf}(q_i)$ inverse document frequency and $\mathrm{tf}(q_i,d)$ term frequency. L is the length of d in words, $\bar{L}$ is the average length of the documents in the collection, k_1 is a parameter that favors high tf scores and b is a parameter penalizing long documents.[8]

W2V-CENT: Following Brokos et al. (2016), we represent query/document terms with pre-trained embeddings. For each query/document we calculate the tf-idf weighted centroid of its embeddings:

$$\mathrm{cent}(t) = \frac{\sum_{i=1}^{l} \boldsymbol{x}_i \cdot \mathrm{tf}(x_i,t)\cdot \mathrm{idf}(x_i)}{\sum_{i=1}^{l} \mathrm{tf}(x_i,t)\cdot \mathrm{idf}(x_i)} \quad (2)$$

where t is a text (query or document) and x_i is the i-th text term with embedding $\boldsymbol{x}_i$. The documents are ranked, with respect to the query, by a k nearest neighbours (kNN) algorithm with cosine distance:

$$\cos_{\mathrm{d}}(q,d) = 1 - \frac{\mathrm{cent}(q)\cdot\mathrm{cent}(d)}{\|\mathrm{cent}(q)\|\cdot\|\mathrm{cent}(d)\|} \quad (3)$$

BERT, similarly to W2V-CENT, relies in pre-trained representations which now are extracted from BERT, thus being context-aware. A text can be represented by its `[cls]` token or by the centroid of its token embeddings. In the latter case the embeddings can be extracted from any of the 12 layers of BERT.[9] Note that the texts in our datasets do not entirely fit in BERT. We thus split them into c chunks (2 to 3 per text) and pass each chunk through BERT to obtain a list of token embeddings per layer (i.e, the concatenation of c token embeddings lists) or c `[cls]` tokens. The final representation is either the centroid of the token embeddings or the centroid of the `[cls]` tokens.

S-BERT (Reimers and Gurevych, 2019) is a BERT model fine-tuned for NLI. According to the authors, training S-BERT for NLI results in better representations than BERT for tasks involving text comparison, like IR. We use the same setting as in BERT.

LEGAL-BERT: Our datasets come from the legal domain which has distinct characteristics compared to generic corpora, such as specialized vocabulary, particularly formal syntax, semantics based on extensive domain-specific knowledge, etc., to the extent that legal language is often classified as a 'sub-language' (Tiersma, 1999; Williams, 2007; Haigh, 2018). BERT and S-BERT were trained on generic corpora and may fail to capture the nuances of legal language. Thus we used a BERT model further pre-trained on EU legislation (Chalkidis et al., 2020), dubbed here LEGAL-BERT, in a similar fashion.

---

[8] We use *elastic*, a widely used IR engine with the BM$_{25}$ scoring function. See `www.elastic.co/`.

[9] BERT is not fine-tuned during this process.

C-BERT: EU laws are annotated with EUROVOC concepts covering the core subjects of EU legislation (e.g., environment, trade, etc.). Our intuition is that a UK law transposing an EU directive will most probably cover the same subjects. Thus we expect that a BERT model, fine-tuned to predict EUROVOC concepts, will learn rich representations describing these concepts which may be useful for pre-fetching. We fine-tune BERT following Chalkidis et al. (2019)[10] and use the resulting model to extract query and document representations similarly to the previous BERT-based methods.

ENSEMBLE is simply a combination of our best two pre-fetchers, C-BERT and BM25:

$$\text{ENS}(q,d) = \alpha \cdot \text{CB}(q,d) + (1-\alpha) \cdot \text{BM}_{25}(q,d) \quad (4)$$

where CB is the score of C-BERT and α is tuned on development data and the scores of the pre-fetchers are normalized in $[0,1]$.

3.2 Document re-ranking

Modern neural re-rankers operate on pairs of the form (q,d) to produce a relevance score, $\text{rel}(q,d)$, for a document d with respect to a query q. Note, however, that the main objective is to rank relevant documents higher than irrelevant. Thus, during training the loss is calculated as:

$$\mathcal{L} = \max(0, 1 - \text{rel}(q,d^+) + \text{rel}(q,d^-)) \quad (5)$$

where d^+ is a relevant document and d^- is an irrelevant document. We have experimented with several neural re-ranking methods each having a function that produces a relevance score s_r for each of the top-k documents returned by the best pre-fetcher. The final relevance score of a document is calculated as: $\text{rel}(q,d) = w_r \cdot s_r + w_p \cdot s_p$, where s_p is the normalized score of the pre-fetcher and w_s, w_p are learned during training.

Given the concerns on the strictness of the ground truth assumption raised in Section 2.2, we hypothesize that re-rankers will eventually over-utilize the pre-fetcher score, s_p, when calculating document relevance, $\text{rel}(q,d)$. As shown in Table 2, in many cases both relevant and irrelevant documents may have high similarity with the query. This in turn may confuse and therefore degenerate the re-ranker's term matching mechanism, i.e., MLPs or CNNs over term similarity matrices.

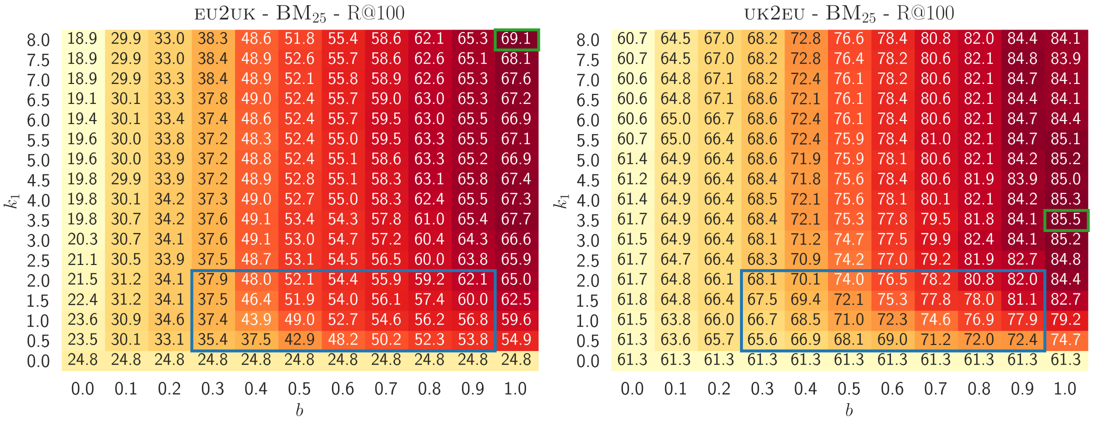
DRMM (Guo et al., 2016) uses pre-trained word embeddings to represent query and document terms. A histogram captures the cosine similarities of a query term, q_i, with all the terms of a particular document. Then an MLP consumes the histograms to produce a document-aware score for each q_i, which is weighted by a gating mechanism assessing the importance of q_i. The sum of the weighted scores is the relevance score of the document. A caveat of DRMM is that it completely ignores the context of the terms which could be of particular importance in our datasets where texts are long.

PACRR (Hui et al., 2017) represents query and document terms with pre-trained embeddings and calculates a matrix S containing the cosine similarities of all query-document term pairs. A row-wise k-max pooling operation on S keeps the highest similarities per query term (matrix S_k). Then, wide convolutions of different kernel (filter) sizes ($n \times n$) with multiple filters per size are applied on S. Each filter of size $n \times n$ attempts to capture n-gram similarities between queries and documents. A $\max$-pooling operation keeps the strongest signals across filters and a row-wise k-$\max$ pooling keeps the strongest signals per query n-gram, resulting in the matrix $S_{n,k}$. Subsequently, a row-wise concatenation of S_k with all $S_{n,k}$ matrices (for different values of n) is performed and a column containing the softmax-normalized idf scores of the query terms is concatenated to the resulting matrix (S_{sim}). In effect, each row of the matrix contains different n-gram based similarity views of the corresponding query term, q_i, along with an idf-based importance score. The relevance score is produced as the last hidden state of an LSTM with one hidden unit, which consumes the rows of S_{sim}. PACRR tries to take into account the context of the query and document terms using n-grams but this context sensitivity is weak and we do not expect much benefits in our datasets which contain long texts.

BERT-based re-rankers: Recent work tries to exploit BERT to improve re-ranking. Following MacAvaney et al. (2019), we use DRMM and PACRR on top of contextualized BERT embeddings derived from BERT. Based on the results of Figure 4, we use C-BERT as the most promising BERT model. We call these two models C-BERT-DRMM and C-BERT-PACRR. We also experiment with two settings depending on whether C-BERT weights are updated (*tuned*) or not (*frozen*) during training.

---

[10] We use all EU laws excluding EU directives that exist in our development and test sets.



EU2UK - BM_{25} - R@100

k_1 \ b	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
8.0	18.9	29.9	33.0	38.3	48.6	51.8	55.4	58.6	62.1	65.3	69.1
7.5	18.9	29.9	33.0	38.4	48.9	52.6	55.7	58.6	62.6	65.1	68.1
7.0	18.9	29.9	33.3	38.4	48.9	52.1	55.8	58.9	62.6	65.3	67.6
6.5	19.1	30.1	33.3	37.8	49.0	52.4	55.7	59.0	63.0	65.3	67.2
6.0	19.4	30.1	33.4	37.4	48.6	52.4	55.7	59.5	63.0	65.5	66.9
5.5	19.6	30.0	33.8	37.2	48.3	52.4	55.0	59.5	63.3	65.5	67.1
5.0	19.6	30.0	33.9	37.2	48.8	52.4	55.1	58.6	63.3	65.2	66.9
4.5	19.8	29.9	33.9	37.2	48.9	52.8	55.1	58.3	63.1	65.8	67.4
4.0	19.8	30.1	34.2	37.3	49.0	52.7	55.0	58.3	62.4	65.5	67.3
3.5	19.8	30.7	34.2	37.6	49.1	53.4	54.3	57.8	61.0	65.4	67.7
3.0	20.3	30.7	34.1	37.6	49.1	53.0	54.7	57.2	60.4	64.3	66.6
2.5	21.1	30.5	33.9	37.5	48.7	53.1	54.5	56.5	60.0	63.8	65.9
2.0	21.5	31.2	34.1	37.9	48.0	52.1	54.4	55.9	59.2	62.1	65.0
1.5	22.4	31.2	34.1	37.5	46.4	51.9	54.0	56.1	57.4	60.0	62.5
1.0	23.6	30.9	34.6	37.4	43.9	49.0	52.7	54.6	56.2	56.8	59.6
0.5	23.5	30.1	33.1	35.4	37.5	42.9	48.2	50.2	52.3	53.8	54.9
0.0	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8

UK2EU - BM_{25} - R@100

k_1 \ b	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
8.0	60.7	64.5	67.0	68.2	72.8	76.6	78.4	80.8	82.0	84.4	84.1
7.5	60.7	64.5	67.0	68.2	72.8	76.4	78.2	80.6	82.1	84.8	83.9
7.0	60.6	64.8	67.1	68.2	72.4	76.1	78.2	80.6	82.1	84.7	84.1
6.5	60.6	64.8	67.1	68.6	72.1	76.1	78.4	80.6	82.1	84.4	84.1
6.0	60.6	65.0	66.7	68.6	72.4	76.1	78.4	80.6	82.1	84.7	84.4
5.5	60.7	65.0	66.4	68.6	72.4	75.9	78.4	81.0	82.1	84.7	85.1
5.0	61.4	64.9	66.4	68.6	71.9	75.9	78.1	80.6	82.1	84.2	85.2
4.5	61.2	64.9	66.4	68.4	71.8	75.6	78.4	80.6	81.9	83.9	85.0
4.0	61.4	64.9	66.4	68.5	72.1	75.6	78.1	80.1	82.1	84.2	85.3
3.5	61.7	64.9	66.4	68.4	72.1	75.3	77.8	79.5	81.8	84.1	85.5
3.0	61.5	64.9	66.4	68.1	71.2	74.7	77.5	79.9	82.4	84.1	85.2
2.5	61.7	64.7	66.4	68.3	70.9	74.2	77.0	79.2	81.9	82.7	84.8
2.0	61.7	64.8	66.1	68.1	70.1	74.0	76.5	78.2	80.8	82.0	84.4
1.5	61.8	64.8	66.4	67.5	69.4	72.1	75.3	77.8	78.0	81.1	82.7
1.0	61.5	63.8	66.0	66.7	68.5	71.0	72.3	74.6	76.9	77.9	79.2
0.5	61.3	63.6	65.7	65.6	66.9	68.1	69.0	71.2	72.0	72.4	74.7
0.0	61.3	61.3	61.3	61.3	61.3	61.3	61.3	61.3	61.3	61.3	61.3

Figure 3: Heatmaps showing R@100 for different values of k_1 and b on EU2UK (left) and UK2EU (right). The selected optimal values (green boxes) are outside the proposed ranges in the literature (blue boxes).

4 Experimental setup

4.1 Pre-trained resources

As several methods rely on word embeddings, we trained a new WORD2VEC model (Mikolov et al., 2013) in both corpora (EU and UK legislation) to better accommodate legal language. Preliminary experiments showed that domain-specific embeddings perform better than generic 200-dimensional GloVe embeddings (Pennington et al., 2014) in development data (EU2UK: 66.5 vs. 59.3 at R@100 and UK2EU: 72.6 vs. 69.8 at R@100).[11]

All BERT (pre-fetching) encoders and BERT-based re-rankers use the -BASE version, i.e., 12 layers, 768 hidden units and 12 attention heads, similar to the one of Devlin et al. (2019).[12]

4.2 Pre-processing - document denoising

One of the major challenges in DOC2DOC IR, as opposed to traditional IR, is the length of the queries and the documents which may induce noise (many uninformative words) during retrieval. Thus we applied several filters (stop-word, punctuation and digits elimination) on both queries and documents and reduced their length by approx. 55% (778 words for UK laws and 1,222 words for EU directives on average). Further on, we filtered both queries and documents by eliminating words with idf score less than the average idf score of the stopwords. Our intuition is that words (e.g., regulation, EU, law, etc.) with such a small idf score are uninformative. Still, the texts are much longer (387 words for UK laws and 631 words for EU directives on average) than the queries used in traditional IR (Table 1). As an alternative to drastically decrease the query size, we experimented with using only the title of a legislative act as a query but the results were worse, i.e., approx. 5-20% lower R@100 on average across datasets, indicating that the full-text is more informative, although the information is sparse. Hence, we only consider the full-text, including the title, for the rest of the experiments.

4.3 Evaluation measures

Pre-fetching aims to bring all the relevant documents in the top-k, thus we report R@k. We observe that for $\mathrm{k} > 100$ the best pre-fetchers have not significant gains in performance in development data, thus we select $\mathrm{k} = 100$, as a reasonable threshold.[13] For re-ranking we report R@20, nDCG@20 and R-Precision (RP) following the literature (Manning et al., 2009). We report the average and standard deviation across three runs considering the best set of hyper-parameters on development data for neural re-rankers.

4.4 Tuning BM_{25}: The case of DOC2DOC IR

The effectiveness of BM_{25} is highly dependant on properly selecting the values of k_1 and b. In traditional (ad-hoc) IR, k_1 is typically evaluated in the range$[0, 3]$ (usually $k_1 \in [0.5, 2.0]$); b needs to be in $[0, 1]$ (usually $b \in [0.3, 0.9]$) (Taylor et al., 2006; Trotman et al., 2014; Lipani et al., 2015). As a general rule of thumb BM_{25} with k_1=1.2 and b=0.75 seems to give good results in most cases (Trotman et al., 2014). We observe that in the case of DOC2DOC IR where the queries are much longer, the optimal values are outside the proposed ranges

---

[11] See also the discussion for legal language in Section 3.1.

[12] See Appendix B for more details.

[13] See Appendix A.3 for an extended ($\mathrm{k} \in [0, 2000]$) performance evaluation on pre-fetching.

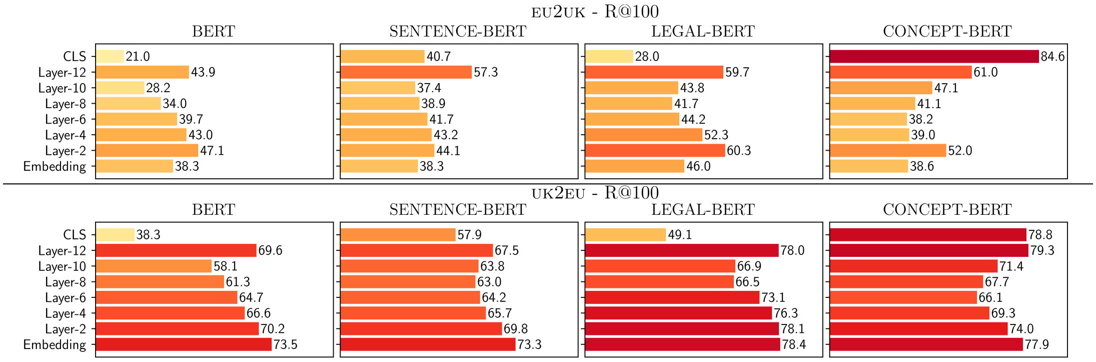


Figure 4: Heatbars showing R@100 (on development data) for text representations extracted from different layers of the various BERT-based pre-fetchers we experimented with.

(Figure 3). In both datasets the optimal values for k_1 and b are relatively high, favoring terms with high tf, while penalizing long documents. In effect BM_{25} uses k_1 and b as a denoising regularizer to over-utilize highly frequent query terms normalized by document length.

4.5 Extracting representations from BERT

Recently there has been a lot of research on understanding the effectiveness of BERT's different layers (Liu et al., 2019; Hewitt and Manning, 2019; Jawahar et al., 2019; Goldberg, 2019; Kovaleva et al., 2019; Lin et al., 2019). Figure 4 shows heatbars comparing representations extracted from different layers of the various BERT-based pre-fetchers we experimented with.[14] LEGAL-BERT and C-BERT which have been adapted in the legal domain perform much better than BERT and S-BERT which were trained on generic corpora. An interesting observation is that the `[cls]` token is a powerful representation only in C-BERT where it was trained to predict EUROVOC concepts. Also, in UK2EU the embedding layer produces the best representations in all BERT variants except C-BERT, where the embedding layer achieves comparable results to the top-2 representations (`[cls]`, Layer-12). This is an indication that the context in this dataset is not as important as in EU2UK.

4.6 Implementation details

All neural models were implemented using the Tensorflow 2 framework. Hyper-parameters were tuned on development data, using early stopping and the Adam optimizer (Kingma and Ba, 2015).

Method	EU2UK	UK2EU
	R@100	R@100
BM_{25} (Robertson et al., 1995)	57.5	93.7
W2V-CENT (Brokos et al., 2016)	50.6	88.2
BERT (Devlin et al., 2019)	54.0	85.1
S-BERT (Reimers and Gurevych, 2019)	57.7	84.8
LEGAL-BERT (Chalkidis et al., 2020)	57.6	90.1
C-BERT (ours)	83.8	92.9
ENSEMBLE (BM_{25} + C-BERT)	**86.5**	**95.0**

Table 4: Pre-fetching results across test datasets.

5 Experimental results

Pre-fetching: Table 4 shows R@100 on the test datasets for the various pre-fetchers considered. On EU2UK, C-BERT is the best method by a large margin, followed by S-BERT and LEGAL-BERT, verifying our assumption that the concept classification task is a good proxy for obtaining rich representations with respect to IR. Both S-BERT and LEGAL-BERT are better than BERT for different reasons. LEGAL-BERT was adapted to the legal domain and is, therefore, able to capture the nuances of the legal language. S-BERT was trained to produce representations suitable for comparing texts with cosine similarity, a task highly related to IR. Nonetheless, having been trained on generic corpora with small texts, it performs much worse than C-BERT. Interestingly, BM_{25} is comparable to both S-BERT and LEGAL-BERT despite its simplicity. As expected, combining C-BERT with BM_{25} further improves the results. In UK2EU R@100 is much higher compared to EU2UK probably because of the shortest queries. Also, as discussed in Section 4.5, the contextual information is not so critical in this dataset, thus we expect the context unaware BM_{25} and W2V-CENT to perform well. Indeed, BM_{25} achieves the best results followed closely by C-BERT and LEGAL-BERT, while W2V-

---

[14] Recall that a text can be represented by its `[cls]` token or by the centroid of its token embeddings which can be extracted from any of the 12 layers of BERT.

Method	EU2UK					UK2EU				
	w_p	w_s	R@20	nDCG@20	RP	w_p	w_s	R@20	nDCG@20	RP
BM_{25}	-	-	45.8	34.4	25.5	-	-	87.5	66.8	**49.4**
C-BERT (ours)	-	-	55.7	37.9	21.8	-	-	79.7	53.0	33.1
ENSEMBLE (BM_{25} + C-BERT)	-	-	54.1	43.1	29.6	-	-	88.0	**67.7**	49.3
+ DRMM	+1.1	-0.8	**59.9** (± 3.2)	41.7 (± 2.4)	24.3 (± 2.9)	+1.3	-0.8	86.3 (± 1.1)	61.6 (± 1.1)	40.1 (± 1.5)
+ PACRR	+4.2	+0.6	54.3 (± 0.2)	**43.3** (± 0.2)	**30.1** (± 0.4)	+4.0	+0.1	88.0 (± 0.0)	**67.7** (± 0.0)	49.3 (± 0.0)
+ C-BERT-DRMM *(frozen)*	+3.3	-1.6	57.9 (± 3.4)	43.1 (± 0.3)	27.3 (± 2.2)	+3.5	-1.0	88.3 (± 0.4)	67.3 (± 0.6)	48.5 (± 1.3)
+ C-BERT-PACRR *(frozen)*	+4.6	+0.9	54.1 (± 0.0)	43.1 (± 0.0)	29.6 (± 0.0)	+2.9	-0.9	**89.6** (± 0.4)	66.5 (± 0.5)	46.0 (± 0.9)
+ C-BERT-DRMM *(tuned)*	+1.9	-0.5	54.1 (± 0.0)	43.1 (± 0.0)	29.6 (± 0.0)	+1.2	+0.5	88.0 (± 0.0)	**67.7** (± 0.0)	49.3 (± 0.0)
+ C-BERT-PACRR *(tuned)*	+1.8	-0.6	54.1 (± 0.0)	43.1 (± 0.0)	29.6 (± 0.0)	+2.0	+2.1	88.0 (± 0.0)	**67.7** (± 0.0)	49.3 (± 0.0)
+ ORACLE	-	-	86.5	87.7	86.5	-	-	95.0	95.3	95.0
Applying date filtering on top of predictions										
Year range	±5 years					±15 years				
ENSEMBLE (BM_{25} + C-BERT)	-	-	76.6	54.6	37.1	-	-	**86.2**	**68.2**	**50.0**
+ DRMM (*pre-filtering*)	+1.1	-0.8	**81.4**	**56.5**	35.4	+1.3	-0.8	85.3	62.6	42.3
+ DRMM (*post-filtering*)	+1.1	-0.8	75.7	49.2	31.1	+1.3	-0.8	83.6	63.5	44.2
+ PACRR (*pre-filtering*)	+4.2	+0.6	76.6	54.8	**37.6**	+4.0	+0.1	**86.2**	**68.2**	**50.0**
+ PACRR (*post-filtering*)	+4.2	+0.6	74.2	52.9	36.5	+4.0	+0.1	85.5	67.6	49.6

Table 5: Re-ranking results across test datasets. The upper zone shows the results of neural re-rankers on top of the best pre-fetchers with respect to (w_s, w_p). It also reports re-ranking results of the best pre-fetchers. The lower zone reports the re-ranking results after applying temporal filtering.

CENT outperforms S-BERT and BERT. Again the ENSEMBLE improves the results.

Re-ranking: Table 5 shows the ranking results on test data for EU2UK and UK2EU. We also report results for BM_{25}, C-BERT, ENSEMBLE and an ORACLE, which re-ranks the top-k documents returned by the pre-fetcher placing all relevant documents at the top. On EU2UK ENSEMBLE performs better than the other two pre-fetchers. Interestingly, neural re-rankers fall short on improving performance and are comparable (or even identical) with ENSEMBLE in most cases, possibly because very similar documents may be relevant or not (Section 2.2, Table 2), leading to *contradicting* supervision.[15] As we hypothesized (Section 3.2), re-rankers overutilize the pre-fetcher score when calculating document relevance, as a defense mechanism (bias) against contradicting supervision, which eventually leads to the degeneration of the re-ranker's term matching mechanism. Inspecting the corresponding weights of the models, we observe that indeed $w_p >> w_s$ across all methods. This effect seems more intense in BERT-based re-rankers (C-BERT + DRMM or PACRR), especially those that fine-tune C-BERT, possibly because these models perform term matching considering sub-word units, instead of full words. In other words, relying on the neural relevance score (s_r) is catastrophic. Similar observations can be made for UK2EU. In both datasets all methods have a large performance gap compared to the ORACLE, indicating that there is still large room for improvement, possibly utilizing information beyond text.

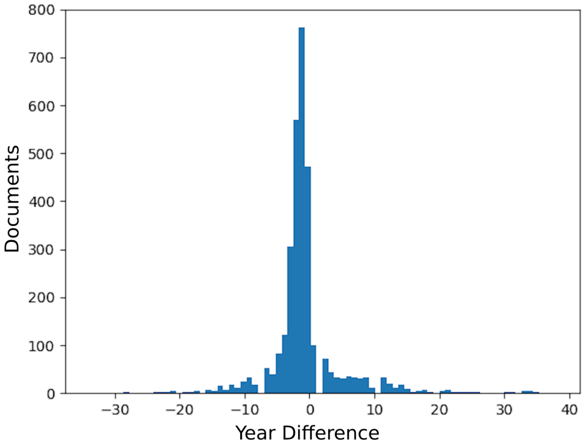


Figure 5: Relevant documents according to their chronological difference with the query on EU2UK development data.

Filtering by year: We have already highlighted the difficulties imposed to our datasets by the frequently amended EU directives (Section 2.2, Table 2). Also, recall that each EU directive defines a deadline (typically 2 years) for the transposition to take place. On the other hand, as we observe in Figure 5, EU directives may already be transposed by earlier legislative acts of member states (the member states act in a proactive manner), or they may delay the transposition for political reasons. In effect, the relevance of a document to a query depends both on the textual content and the time the laws were published. Thus, we filter out documents that are outside a predefined distance

---

[15] By *contradicting* supervision we mean similar training query-document pairs with opposite labels.

(in years) from the query in two ways, *pre-filtering* and *post-filtering*. Pre-filtering is applied to the pre-fetcher, i.e., prior to re-ranking, while post-filtering is applied after the re-ranking. Note that our main goal is to improve re-ranking. We thus apply the filtering scheme to the ENSEMBLE, DRMM and PACRR. The lower zone of Table 5 shows the results of the whole process. In EU2UK, the hardest out of the two datasets, the time filtering has a positive impact, improving the results by a large margin. On the other hand, filtering seems to have a minor effect in UK2EU.

5.1 EU2UK $\neq$ UK2EU

Across experiments, we observe that best practices vary between the EU2UK and UK2EU datasets. EU2UK benefits from C-BERT representations, while in UK2EU context-unaware and domain-agnostic BM_{25} has comparable or better performance than C-BERT. Similarly, we observe that time filtering further improves the performance in EU2UK, while we have a contradicting effect in UK2EU. Given the overall results, we conclude the two datasets have quite different characteristics. Thus, it is important to consider both EU2UK and UK2EU independently, although one may initially consider them to be symmetric.

6 Related work

IR in the legal domain is widely connected with the Competition on Legal Information Extraction/Entailment (COLIEE). From 2015 to 2017 (Kim et al., 2015, 2016; Kano et al., 2017), the task was to retrieve Japanese Civil Code articles given a question, while in COLIEE 2018 and 2019 (Kano et al., 2018; Rabelo et al., 2019), the task was to retrieve supporting cases given a short description of an unseen case. However, the texts of these competitions are small compared to our datasets. Also, most submitted systems do not consider recent advances in IR, i.e, neural ranking models (Guo et al., 2016; Hui et al., 2017; McDonald et al., 2018; MacAvaney et al., 2019), which have recently managed to improve rankings of conventional IR, or end-to-end neural models which have recently been proposed (Fan et al., 2018; Khattab and Zaharia, 2020). Again, these end-to-end methods were applied on small texts. On the other hand, there has been some work trying to cope with larger queries, i.e., *verbose* or expanded queries, (Paik and Oard, 2014; Gupta and Bendersky, 2015; Cummins, 2016). Nonetheless, the considered queries are at most 60 tokens long, contrary to our datasets where, depending on the setting, the average query length is 1.8K or 2.6K tokens (Table 1). Neural methods greatly rely on text representations, thus Reimers and Gurevych (2019) proposed S-BERT which is trained to compare texts for an NLI task and could thus be used to extract representations suitable for IR. Towards the same direction, Chang et al. (2020) experimented with several auxiliary tasks to extract better representations. However, the latter two methods have been evaluated on datasets with much smaller texts than the ones we consider.

7 Conclusions and future work

We proposed DOC2DOC IR, a new family of IR tasks, where the query is an entire document, thus being more challenging than traditional IR. This family of tasks is particularly useful in regulatory compliance, where organizations need to ensure that their controls comply with the existing legislation. In the absence of publicly available DOC2DOC datasets, we compile and release two datasets, containing EU directives and UK laws transposing these directives. Experimenting with conventional (BM_{25}) and neural pre-fetchers we showed that a BERT model fine-tuned on an in-domain classification task, i.e., predict EUROVOC concepts, is by far the best pre-fetcher in our datasets. We also showed that neural re-rankers fail to improve the performance, as their term matching mechanisms degenerates, and over-utilize the pre-fetcher score. In the future, we would like to investigate alternatives in exploiting additional information that may be critical in the newly introduced tasks (EU2UK, UK2EU). In this direction naively utilizing chronological information leads to vast performance improvement in EU2UK dataset. One possible direction is to model the cross-document relations (e.g., amendments) using Graph Convolutional Networks (Kipf and Welling, 2016), while better modeling the dimension of time (i.e., chronological difference between a query and a document) is also crucial. Further on, to better deal with long documents, we plan to investigate text summarization by employing a state-of-the-art neural summarizer, e.g., BART of Lewis et al. (2020), or sentence selection techniques, e.g., rationale extraction (Lei et al., 2016; Chang et al., 2019), to find the most important sections or sentences and create shorter and more informative versions of queries/documents.

References

Georgios-Ioannis Brokos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2016. Using centroids of word embeddings and word mover's distance for biomedical document retrieval in question answering. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing (BioNLP 2016), at the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 114–118, Berlin, Germany.

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Large-Scale Multi-Label Text Classification on EU Legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi S. Jaakkola. 2019. A Game Theoretic Approach to Class-wise Selective Rationalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training Tasks for Embedding-based Large-scale Retrieval. In *International Conference on Learning Representations*.

Wei-Tsen Milly Chiang, Markus Hagenbuchner, and Ah Chung Tsoi. 2005. The wt10g dataset and the evolution of the web. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, WWW '05, page 938–939, New York, NY, USA. Association for Computing Machinery.

Charles Clarke, Nick Craswell, and Ian Soboroff. 2004. Overview of the trec 2004 terabyte track. In *TREC*.

Ronan Cummins. 2016. A study of retrieval models for long documents and queries in information retrieval. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 795–805, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, abs/1810.04805.

Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Chengxiang Zhai, and Xueqi Cheng. 2018. Modeling Diverse Relevance Patterns in Ad-Hoc Retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 375–384, New York, NY, USA. Association for Computing Machinery.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *CoRR*, abs/1901.05287.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 55–64, New York, NY, USA. Association for Computing Machinery.

Manish Gupta and Michael Bendersky. 2015. Information retrieval with verbose queries. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 1121–1124, New York, NY, USA. Association for Computing Machinery.

Rupert Haigh. 2018. *Legal English*. Routledge.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1049–1058, Copenhagen, Denmark. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Yoshinobu Kano, Mi-Young Kim, Randy Goebel, and Ken Satoh. 2017. Overview of coliee 2017. In *COLIEE@ ICAIL*, pages 1–8.

Yoshinobu Kano, Mi-Young Kim, Masaharu Yoshioka, Yao Lu, Juliano Rabelo, Naoki Kiyota, Randy Goebel, and Ken Satoh. 2018. Coliee-2018: Evaluation of the competition on legal information extraction and entailment. In *JSAI International Symposium on Artificial Intelligence*, pages 177–192. Springer.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT.

Mi-Young Kim, Randy Goebel, Yoshinobu Kano, and Ken Satoh. 2016. Coliee-2016: evaluation of the competition on legal information extraction and entailment. In *International Workshop on Juris-informatics (JURISIN 2016)*.

Mi-Young Kim, Randy Goebel, and S Ken. 2015. Coliee-2015: evaluation of legal question answering. In *Ninth International Workshop on Juris-informatics (JURISIN 2015)*.

Diederik P. Kingma and Jim Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 5th International Conference on Learning Representations*.

Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR*, abs/1609.02907.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing Neural Predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas.

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

Tom CW Lin. 2016. Compliance, technology, and modern finance. *Brook. J. Corp. Fin. & Com. L.*, 11:159.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Aldo Lipani, Mihai Lupu, Allan Hanbury, and Akiko Aizawa. 2015. Verboseness fission for bm25 document length normalization. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, ICTIR '15, page 385–388, New York, NY, USA. Association for Computing Machinery.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic Knowledge and Transferability of Contextual Representation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1101–1104, New York, NY, USA. Association for Computing Machinery.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2009. *Introduction to Information Retrieval*. Cambridge University Press.

Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. *CoRR*, abs/1809.01682.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations*, Scottsdale, AZ.

Jiaul H. Paik and Douglas W. Oard. 2014. A fixed-point method for weighting terms in verbose informational queries. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, page 131–140, New York, NY, USA. Association for Computing Machinery.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2019. A summary of the coliee 2019 competition.

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

S. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at trec3. *In Overview of the Third Text Retrieval Conference*, pages 109—-126.

Shazia Sadiq and Guido Governatori. 2015. *Managing Regulatory Compliance in Business Processes*, pages 265–288. Springer Berlin Heidelberg, Berlin, Heidelberg.

Michael Taylor, Hugo Zaragoza, Nick Craswell, Stephen Robertson, and Chris Burges. 2006. Optimisation methods for ranking functions with multiple parameters. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, page 585–593, New York, NY, USA. Association for Computing Machinery.

Peter M Tiersma. 1999. *Legal language*. University of Chicago Press.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 2014 Australasian Document Computing Symposium*, ADCS '14, page 58–65, New York, NY, USA. Association for Computing Machinery.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel-Cyrille Ngonga Ngomo, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16(138).

Ellen M. Voorhees. 2005. The TREC Robust Retrieval Track. *SIGIR Forum*, 39(1):11–20.

Christopher Williams. 2007. *Tradition and change in legal English: Verbal constructions in prescriptive texts*, volume 20. Peter Lang.

A Dataset Compilation: Technical Details

In this section, we present the technical details associated with the compilation of both datasets described in the main paper. More specifically we present the procedure of creating both corpora as well as modelling the transposition relations between EU and UK entries.

A.1 EU corpus

The compilation of the EU corpus is more straightforward than its UK counterpart but involves some in-domain knowledge to filter unwanted legislation.

- We initially download the core metadata associated with each document in the EU corpus by utilizing the SPARQL endpoint of the EU Publications Office (`http://publications.europa.eu/webapi/rdf/sparql`) and the EURLEX platform (`https://eur-lex.europa.eu`), as a REST-ful API.
- Following the metadata collection, we proceed to filter out documents based on their type in order to retain only EU directives and regulations. This involves excluding corrigendums. Corrigendums introduce corrections to prior EU legislation. Usually these corrections are minimal and change single phrases such as ("In Regulation X, for: '… 4 July 2019 …', read: '… 4 July 2015 …'."). Thus these documents lack the context to be both classified and correlated with other documents. [16] and decisions, both of which are irrelevant to our use case. The final EU corpus contains approximately 60k entries.

A.2 UK corpus

Compiling the UK corpus is not as trivial, since the `legislation.gov.uk` API is not as evolved and we therefore have to manually crawl large parts of the database to build our corpus.

- The collected UK laws from the `legislation.gov.uk` portal form the initial corpus which includes approximately 100k documents.
- Similarly to our processing of the EU corpus, we only retain documents in specific legislation types (UK Public General Acts, UK Local Acts, UK Statutory Instruments and UK Ministerial Acts). We then eliminate laws that aim to align English legislation with the rest of the United Kingdom's, more specifically Scotland, Northern Ireland and Wales. The final UK corpus includes 52K UK entries.

A.3 EU2UK Transpositions

Transpositions are relations between entries in the EU and UK corpora which we use to define relevance for our retrieval tasks. Processing these relations is the most challenging aspect of compiling our datasets and involves several steps.

- We use the aforementioned SPARQL endpoint, to retrieve the transpositions between EU directives and the corresponding UK regulations

---

[16] See `https://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1593684165879&uri=CELEX:32004L0038R(02)` as an example.

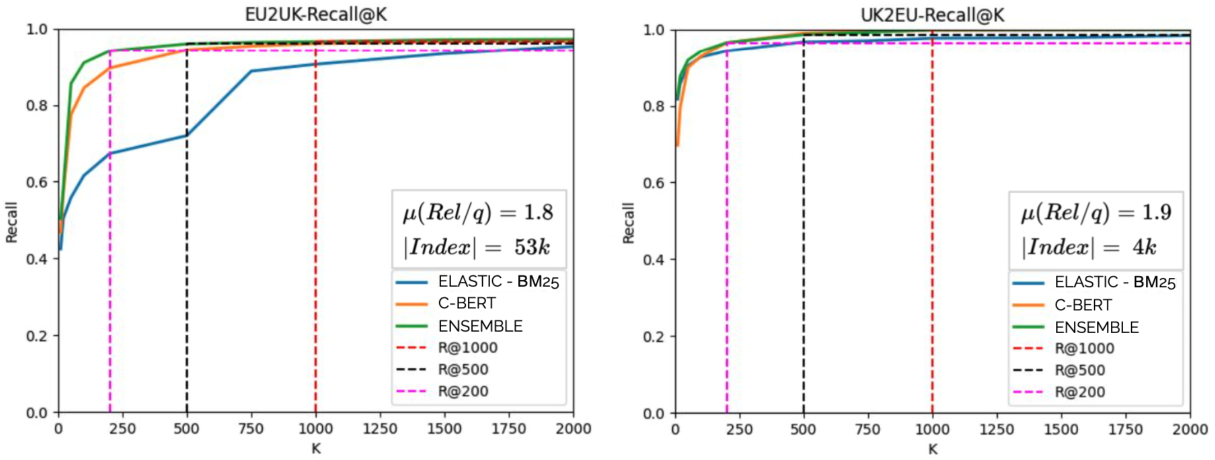


Figure 6: Recall@k, where $k \in [0, 2000]$, across the three best pre-fetchers (i.e., BM$_{25}$, C-BERT and ENSEMBLE) on the development dataset.

that implement them. We initially collect approximately 10k EU2UK pairs. In these pairs the transposed EU law is referred to by its unique portal ID but the transposing UK law is referred to by its title. This is the primary challenge in modelling the transposition relations, since mapping legislation titles to unique entries in our UK corpus is not trivial. We hypothesize that these relations are manually inserted in the database and therefore human errors make performing exact matches often impossible. Apart from the matching difficulties, some of the pairs in the pool are inserted mistakenly and hence need to be filtered.

- We first filter the noisy pairs. Pairs are considered noisy either because they are duplicates or because the do not meet some manually set criteria. In turn, duplication can occur either because identical pairs are inserted more than once or because pairs in which the UK title is mildly paraphrased are erroneously considered different. Our pool is reduced to 8k pairs after resolving the former and to 7k pairs after also resolving the latter. We further reduce the pool size by filtering pairs in which the UK title refers to non-English legislation (Scotland, Northern Ireland, Wales or Gibraltar) Non-English legislation usually has an almost identical counterpart within the pure english corpus. [17]. or in which the title does not contain certain keywords (e.g., Act, Regulation, Order, Rule). Documents that do not contain any of these keywords are not officially published in the `legislation.gov.uk` portal. Most of these are official releases from national governmental bodies, e.g. Ministries. For instance the *First Annual Report of the Inter-Departmental Ministerial Group on Human Trafficking* is not part of the UK's national legislation..
- To resolve the matching challenge, we employ a complex matching scheme where for each pair we gradually normalize the UK title until we find either a singular match or multiple ones. In the latter case, we resolve the matches with heuristics. Our normalizations include lower-casing, leading and trailing phrase removal, punctuation elimination, date removal and manually inserted substitutions.
- After reducing our pair pool and then implementing our matching scheme we can with high confidence present 4k transposition pairs which we use in our datasets.

B BERT models

All BERT variants (BERT, S-BERT, LEGAL-BERT) are publicly available from Hugging Face:

- **BERT**: The original BERT pre-trained for Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) in English Wikipedia and Books corpus. Available at `https://huggingface.co/nlpaueb/bert-base-uncased-eurlex`.
- **S-BERT**: This is the original BERT fine-tuned in STS-B NLI dataset. Available at `https://`

---

[17] See `https://www.legislation.gov.uk/uksi/2017/407/contents` and `https://www.legislation.gov.uk/nisr/2017/81/contents`

`huggingface.co/deepset/sentence_bert`.

- **LEGAL-BERT (EURLEX)**: This is the original BERT further pre-trained in EU legislaiton. Available at `https://huggingface.co/nlpaueb/bert-base-uncased-eurlex`.

C Selecting k for pre-fetching

In Section 4.1, we stated that we report $\mathrm{R@k}$ with $\mathrm{k} = 100$ in order to evaluate and compare pre-fetching methods. In Figure 6, we present the performance of the best pre-fetching methods (i.e., BM25, C-BERT and ENSEMBLE) for different values of $k \in [0, 2000]$ on the development set. We observe that after $k = 100$, the ENSEMBLE pre-fetcher has not significant gains in performance, thus we select $k = 100$, as a reasonable threshold.

The Chinese Remainder Theorem for Compact, Task-Precise, Efficient and Secure Word Embeddings

Patricia Thaine
Department of Computer Science
University of Toronto
`pthaine@cs.toronto.edu`

Gerald Penn
Department of Computer Science
University of Toronto
`gpenn@cs.toronto.edu`

Abstract

The growing availability of powerful mobile devices and other edge devices, together with increasing regulatory and security concerns about the exchange of personal information across networks of these devices has challenged the Computational Linguistics community to develop methods that are at once fast, space-efficient, accurate and amenable to secure encoding schemes such as homomorphic encryption. Inspired by recent work that restricts floating point precision to speed up neural network training in hardware-based SIMD, we have developed a method for compressing word vector embeddings into integers using the Chinese Reminder Theorem that speeds up addition by up to 48.27% and at the same time compresses GloVe word embedding libraries by up to 25.86%. We explore the practicality of this simple approach by investigating the trade-off between precision and performance in two NLP tasks: compositional semantic relatedness and opinion target sentiment classification. We find that in both tasks, lowering floating point number precision results in negligible changes to performance.

1 Introduction

In recent years, NLP models, particularly language models, have come under increasing scrutiny for their potential privacy leaks (e.g., Carlini et al. (2018)). One answer has been to push NLP models onto edge devices, such as mobile phones and browsers, for on-device inference using differentially private federated learning (e.g., Yang et al. (2018)). Edge computing, in turn, requires smaller and faster models, such as DistillBERT (Sanh et al., 2019) or mobileBERT (Sun et al., 2020), which theoretically improves the viability of using BERT embeddings on these devices.

What has yet to take place, however, is careful numerical analysis of word embeddings at varying precisions for different NLP tasks. Numerical analysis can inform potential alternatives to defining the representations of word embeddings themselves. We must carefully orchestrate a balance among the size of these representations, their security, the algebra of operations that they enable, and, because many edge devices are resource-restricted relative to the GPU clusters that we do research on, the efficiency and power consumption of those operations. In particular, feature/basic phones are growing in popularity in developing nations[1][2], often with the possibility of expanding their memory to about 16GB with a MicroSD. Many privacy-enabled language models, in particular, are simply out of reach for these low-resource devices.

No work to date has proposed using the Chinese Remainder Theorem (CRT) to create software-based compressed word embeddings, adapted to Single Instruction Multiple Data (SIMD). We propose here a CRT-based method which speeds up addition by up to 48.27% and compresses GloVe (Pennington et al., 2014) word embedding libraries by up to 25.86% (up to 68.68% of the original full-precision library), depending on the precision selected. We also explore different levels of numerical precisions for a representative task, as well as more abstractly by analysing the absolute error resulting from adding and multiplying truncated and rounded values.

Related Work. A number of recent papers have explored limiting input precision in order to speed up neural network training and inference. Zhang et al. (2018) and Ling et al. (2016) specifically

---

[1] https://qz.com/africa/1206462/smartphones-lost-market-share-to-feature-phones-in-africa-last-year/

[2] https://techpoint.africa/2019/03/18/drive-for-feature-phone-penetration-in-africa/

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3512–3521
April 19 - 23, 2021.

looked at the effects of limiting word embedding precision. Zhang et al. (2018) evaluate embeddings of different sizes combined with hardware-implemented SIMD for an adaptation of stochastic gradient descent that achieves a memory improvement of 2X and 1.2X faster training speed. Ling et al. (2016) demonstrate that using word embeddings made up of 8-bit fixed-point values performs just as well as other embeddings on word similarity, phrase similarity, and dependency parsing tasks. Tissier et al. (2019), on the other hand, has shown how to generate binary word embeddings that fit within a CPU's cache to increase computing speed. With limited effect on task accuracy, they manage to generate 256-bit embeddings which are 37.5× smaller than traditional 300-dimensional word embeddings.

While we have demonstrated that CRT can be used to increase the efficiency of word vector addition by transforming a vector into a single large number, it is usually used for the opposite purpose: to transform larger numbers into many smaller numbers to improve multiplication efficiency. For this purpose, CRT is integrated in certain homomorphic encryption libraries.

CRT has also been proposed as a method of encrypting multiple entries in a database (e.g., a_i) as one number, each decryptable by a secret modulus (e.g., m_i) (Yan, 2002). The security of such schemes are shaky and more complexity is usually added to guarantee security (Liu et al., 2014; Lin et al., 1992).

2 Background

2.1 Single Instruction Multiple Data

Single Instruction Multiple Data is a computer architecture classification described in Flynn (1966) as a single instruction acting simultaneously on multiple operands. Flynn (1966) points out that performance increases with the number of units. Prior to Flynn (1966), SIMD was discussed in Unger (1958); Slotnick et al. (1963); Crane and Githens (1965); Hellerman (1966). More recently, the Chinese Remainder Theorem was proposed as a method for attaining SIMD in order to optimize arithmetic for homomorphically encrypted values (Gentry et al., 2012; Smart and Vercauteren, 2014). We do not know of work that takes advantage of CRT-based SIMD for NLP, despite the repetitive tasks performed within certain algorithms.

2.2 Chinese Remainder Theorem

Theorem 2.1 (The Chinese Remainder Theorem)**.** *Let $m_1, m_2, ..., m_r$ be pairwise relatively prime positive integers. Then the system of congruences*

$$X \equiv a_1 (\text{mod } m_1)$$

$$X \equiv a_2 (\text{mod } m_2)$$

$$\dots$$

$$X \equiv a_r (\text{mod } m_r)$$

has a unique solution modulo $M = m_1 m_2 ... m_3$. (Rosen, 2000)

We use $\mathbf{a}$ to denote the vector of values $a_1, a_2, ..., a_r$ and $\mathbf{m}$ to denote the vector $m_1, m_2, ..., m_r$.

The Chinese Remainder Theorem can also be used to separate very large numbers into smaller chunks on which, in certain cases, arithmetic operations can be performed more efficiently (Rosen, 2000).

3 Algorithm: Vec2int

First, we must determine a minimum floating point precision (ϕ) the values within our word embeddings must have. Since the CRT only works with positive integers (and polynomials), we will multiply each value in the word embeddings (a_i) by 10^ϕ and truncate (or round) the result, then add the lowest possible integer value that can be found within our word embeddings (s). Before running the CRT algorithm, we must pre-compute a vector $\mathbf{m}$ of moduli, each of which are coprime and strictly greater than any possible value of a_i. $\mathbf{a}$ and $\mathbf{m}$ can then be input into the CRT algorithm in order to produce an integer X. Since we know the values of $\mathbf{m}$, we can easily perform a "reverse CRT" by calculating $a_i \leftarrow X \pmod{m_i}$.

Algorithm 1 Vector To Integer

```
procedure VEC2INT($\mathbf{a}, \phi, \mathbf{m}, s$)
    for $i \leftarrow 1$ to $r$ do
        $a_i \leftarrow \text{truncate}(a_i * 10^\phi) + s$
    end for
    $X \leftarrow \text{CRT}(\mathbf{a}, \mathbf{m})$
    return $X$
end procedure
```

When two CRT-encoded vectors (say X_1 and X_2) are added together, their sum at index i can still be decoded by calculating $(X_1 + X_2)$ (mod

m_i), provided that either $X_1 + X_2 < m_i$, or $(X_1 + X_2) \pmod{2m_i}$, when $X_1 + X_2 \geq m_i$. The same property holds for multiplication; i.e., $(X_1 * X_2) \pmod{(m_i)^2}$, when $X_1 * X_2 \geq m_i$. In other words, this representation supports vector addition, multiplication and, in a more restricted range, subtraction, through executing those operations directly on integers.

Addition and multiplication are fundamental to most compositional semantics approaches, beginning with Mitchell and Lapata (2008, 2010). Addition is also central to measuring the relationship between word pairs, particularly when using the L1-norm as a metric, which is less sensitive to outliers than the L2-norm (Stratos, 2017). Word analogy is an example of such a task. Of course, the these operations can also be applied to measuring the relationship between sentence pairs and document pairs.

4 Reasoning about Precision using Numerical Analysis

One might assume that we have to perform extensive empirical analysis to determine the minimum floating point precision necessary for the inputs of a particular NLP algorithm. However, there are generalizations we can make based on concepts as simple as the absolute error and relative error of components of the operations, described in Heath (2018) as:

$$\text{Absolute error} = \text{approx. value} - \text{true value} \quad (1)$$

$$\text{Relative error} = \frac{\text{absolute error}}{\text{true value}} \quad (2)$$

For our operations, the vectors truncated or rounded at precision Φ within a dataset $|D_\Phi|$ are denoted $\mathbf{v_{\Phi i}}$, while the vectors of the original dataset are denoted $\mathbf{v_i}$. If we want to refer to index k of vector $\mathbf{v_i}$, we write it as $\mathbf{v_i}[k]$, where $|\mathbf{v_1}|$ is the size of the embeddings we are working with. The average absolute error between the vectors of the two datasets of size $|D|$ is computed as follows:

$$\frac{\sum_i \sum_k |\mathbf{v_{\Phi i}}[k] - \mathbf{v_i}[k]|}{|D| * |\mathbf{v_1}|} \quad (3)$$

We calculate the average absolute error of addition as follows:

$$\frac{\sum_i \sum_j \sum_k |(\mathbf{v_{\Phi i}}[k] + \mathbf{v_{\Phi j}}[k]) - (\mathbf{v_i}[k] + \mathbf{v_j}[k])|}{\binom{|D|}{2} * |\mathbf{v_1}|} \quad (4)$$

5 Experimentation & Analysis

5.1 Arithmetic

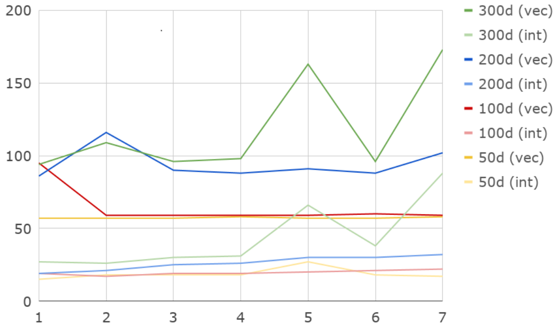


Figure 1: Milliseconds (y-axis) it takes to run 100000 element-wise additions over vectors of (*d) dimensions at ϕ precision (x-axis).

We have tested the efficiency of adding and multiplying integer representations of word embeddings compared to adding and multiplying their original vectors. On a 1.9GHz Intel i7-8650U with a 2.11 GHz burst rate, 1024MB L2 cache and 8192MB L3 cache, 100K additions (Figure 1) generally take 3x longer to perform on vectors than their integer encodings, regardless of precision and dimensionality, while the results for multiplication (Figure 2) are profound and negative at larger precisions and dimensionalities.

We may then theoretically expect the CRT-encoding of vectors as integers to result in a significant performance gain in the case of addition, but for the most part a significant performance loss in the case of multiplication. Nevertheless, in comparing the various pairs of rows in Tables 1 and 2, we notice that the computational gains made from adding CRT representations rather than word vec-

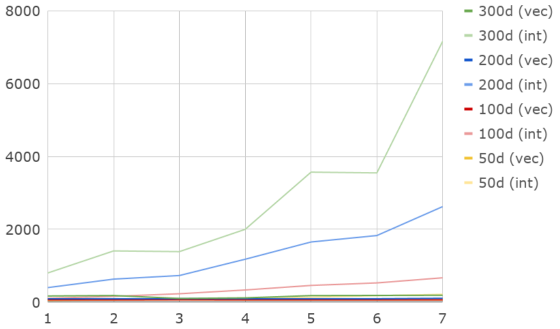


Figure 2: Milliseconds (y-axis) it takes to run 100000 element-wise multiplications over vectors of (*d) dimensions as ϕ precision (x-axis).

	100000	**1000000**	**10000000**
50d1(int)	**15**	**156**	**1548**
50d1(vec)	57	572	5764
50d2(int)	**18**	**172**	**1716**
50d2(vec)	57	572	5711
50d3(int)	**18**	**174**	**1746**
50d3(vec)	57	570	5712
50d4(int)	**18**	**184**	**1835**
50d4(vec)	58	570	5718
50d5(int)	**27**	**174**	**1719**
50d5(vec)	57	572	5712

Table 1: Milliseconds it takes to run 100000, 1000000, and 10000000 many additions over vectors of (*d) dimensions at (d*) precision.

	100000	**1000000**	**10000000**
50d1(int)	**37**	**371**	**3711**
50d1(vec)	57	574	5701
50d2(int)	**54**	**539**	**5385**
50d2(vec)	57	571	5690
50d3(int)	72	719	7203
50d3(vec)	**60**	**571**	**5698**
50d4(int)	102	1026	10261
50d4(vec)	**57**	**570**	**5698**
50d5(int)	131	1310	13107
50d5(vec)	**57**	**571**	**5700**

Table 2: Milliseconds it takes to run 100000, 1000000, and 10000000 many element-wise multiplication over vectors of (*d) dimensions at (d*) precision.

tors can considerably outweigh the losses from multiplying integer representations when the number of additions is greater or equal to the number of multiplications. In practice, as illustrated in several use cases below, the number of additions can be far greater in many important tasks. Recall also the findings of Ling et al. (2016), who did not observe a significant decrease in performance when using an 8-bit fixed-point value for word embeddings in word and phrase similarity and dependency parsing tasks. This implies that there is some room for compromise on precision practical NLP tasks as well.

5.2 Use Case 1: Compositional Semantics

A prime example of the significant performance improvements brought on by integer addition of word embeddings in the area of compositional semantics can be found in Salton and McGill (1986). It was demonstrated that vector addition is more effective than other proposed unsupervised compositional models (multiplication (Mitchell and Lapata, 2010), tensor product with convolution (Widdows and Ferraro, 2008), and dilation (Mitchell and Lapata, 2010)) for determining semantic relatedness between bigrams and other bigrams or unigrams (Asaadi et al., 2019).

Dataset and previous results. In Asaadi et al. (2019), the authors introduce BIRD, a bigram relatedness dataset created using the Best-Worst Scaling annotation method. To accomplish this task, annotators are provided with n sample, where n is often 4, and are asked which of the samples best represent the a given property and which one represents it the worst (Kiritchenko and Mohammad, 2016, 2017). Asaadi et al. (2019) compute bigram semantic relatedness using three different kinds of word embeddings; namely, pre-trained GloVe vectors[3], pre-trained fastText word embeddings[4], and word-context co-occurrence vectors extracted from a corpus of university websites (Turney et al., 2011)[5]. Specifically, the authors compute the relatedness score for the vectors representing the term pair AB-X, where AB is a bigram and X can be either a bigram or a unigram. The results of four unsupervised compositional models were compared:

- Weighted addition (Salton and McGill, 1986);
- Multiplication (Mitchell and Lapata, 2010);
- Tensor product with convolution (Widdows and Ferraro, 2008);
- Dilation (Mitchell and Lapata, 2010).

They then used Pearson correlation to compare the semantic relatedness scores, computed using these unsupervised compositional models, with the gold-standard in the BiRD. Addition turns out to be the composition method that results in the highest Pearson correlation scores.

Integrating vec2int The task introduced by Asaadi et al. (2019) is a prime candidate for using vec2int to speed up computations and reduce space: vast numbers of additions need to be performed in order to compute bigram relatedness in large datasets. These datasets often occupy a large amount of space and would also benefit from a compression method that can support addition in the compressed domain. The catch is that, in

---

[3] Originally obtained at: `https://nlp.stanford.edu/projects/glove`

[4] Originally obtained at: `https://fasttext.cc/docs/en/crawl-vectors.html`

[5] Code and data can be found at: `https://github.com/sasaadi/BiRD`

order to convert from a vector to an integer, we must limit the precision of each of the entries of the vector. Clearly, this can lead to concerns regarding the levels of accuracy that can be guaranteed by functions taking these precision-limited vectors as inputs. For our calculations, we forego any comparisons with floating point precision below 8 digits (other than the original number of digits[6]), since we want to maximize computational efficiency and minimize storage size. Recall that ϕ is the number of digits we keep after the decimal point.

Figure 3 and Figure 4 display the effects of truncating and rounding floating point numbers at various precisions then converting them to integers for determining bigram relatedness through the composition of fastText and GloVe vectors, respectively[7]. Interestingly, even sticking to a floating point precision of 1 or 2 can lead to results which are just as good as with the original precision.

Figure 5 shows the average absolute error (see Equation 3) and Figure 6, the average relative error (see Equation 4), for each possible sum of two vectors within the term context, fastText, and GloVe datasets. We can see that an increase in average absolute and relative errors lines up with the decrease in correlation score, and so we can determine at which point bigram relatedness would start to falter without computing Pearson or Spearman correlations. In fact, it appears to suffice only to calculate the average (Figure 9) and relative (Figure 10) errors of the truncated and rounded word vectors themselves (see Equations 1 and 2, respectively).

For interest, we show the average absolute error (Figure 7) and average relative error (Figure 8) for composition through multiplying fastText vectors at various precisions. fastText exhibits the most dramatic difference in performance between composition by addition and composition by multiplication at lower precisions.

5.3 Compression

We test the utility of the CRT as a method of compressing word embeddings. The results are shown in Table 3, reaching up to a 25.85% space reduction for GloVe vectors. We also calculate an upper bound of how large a dataset of word embeddings can get when compressed using CRT representations using the same relative primes $\mathbf{m}$ as the ones used to calculate the results displayed on Tables 3. Calculating the upper bound essentially comes down to associating each word in the dataset to the largest possible number N given $\mathbf{m}$:

$$N = \prod_i m_i \tag{5}$$

To demonstrate that, no matter what the vectors were the compression would work, we show the upper bounds compared to the size of the original dataset of 50-dimensional GloVe embeddings and the true CRT-compressed dataset of those embeddings on Table 4.

5.4 Use Case 2: Data Preprocessing for Private and Secure Computing

Another potential use case is if one wants to convert a text to its corresponding word embeddings directly on device, before being sent for cloud processing through homomorphic encryption or secure multiparty computation algorithms, which compute on precision-limited obfuscated data.

Arora et al. (2020) shows that pre-trained embeddings perform within 5-10% accuracy of benchmark tasks (NER, Sentiment analysis, GLUE) compared to contextual embeddings (e.g., BERT) and can even regularly match their performance when using industrial-level data.

Directly converting the words to their respective word embeddings on device is particularly useful for those algorithms, where table lookup over large amounts of data can be a fairly expensive operation. Not only can CRT representations compress the word embeddings to save valuable space on edge devices, but they can be selectively decompressed, unlike zip files, as needed. If you want to only convert the words in a user query to embeddings, there is no need to decompress the entire dataset!

We compare the results of CNN-based sentence-level sentiment analysis using GloVe vectors as input (Kim, 2014)[8] with those using

---

[6] The largest precision within the term context vector is $\phi = 20$, within fastText is $\phi = 12$, and within GloVe is $\phi = 8$.

[7] We would like to thank Shima Asaadi, Saif M. Mohammad, and Svetlana Kiritchenko for providing the code and data they used for the their Big BiRD paper. The correlation scores for term context vectors are nearly the same as those for GloVe at untruncated precision, but remain stable down to $\phi = 2$.

[8] `https://github.com/yoonkim/CNN_`

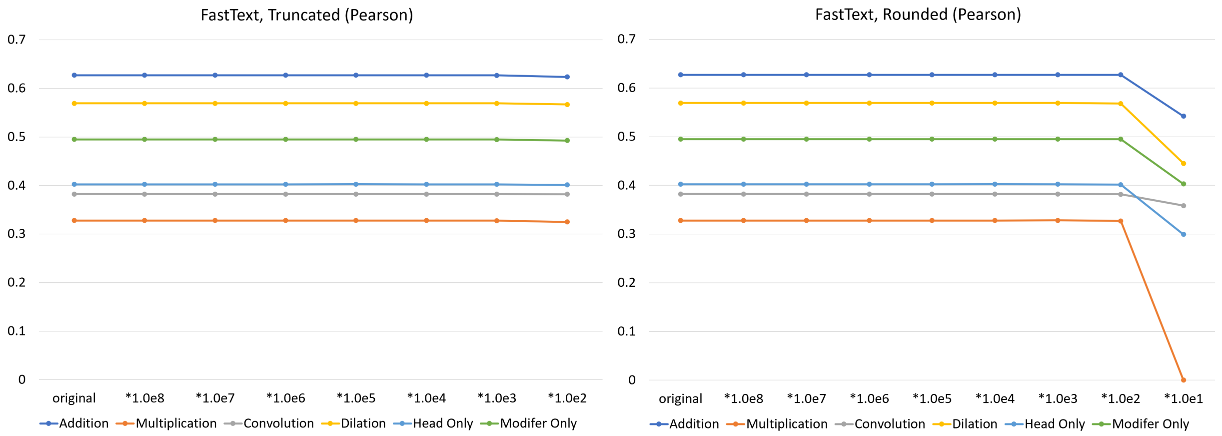


Figure 3: Pearson correlation results for determining bigram relatedness using addition of FastText vectors. $*1.0\mathrm{e}\phi$ means we are approximating at precision ϕ. The trends are analogus with Spearman correlations.

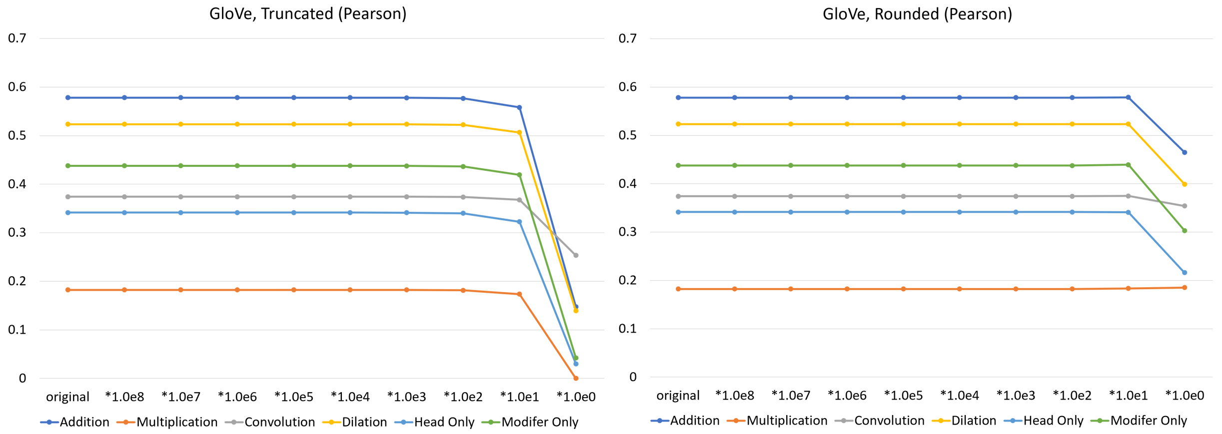


Figure 4: Pearson correlation results for determining bigram relatedness using addition of GloVe vectors. $*1.0\mathrm{e}\phi$ means we are approximating at precision ϕ. The trends are analogus with Spearman correlations.

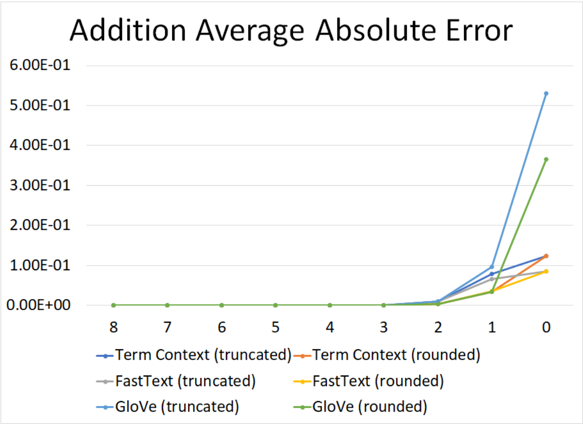


Figure 5: Average Absolute Error (y-axis) of Addition Composition at various precisions ϕ (x-axis).

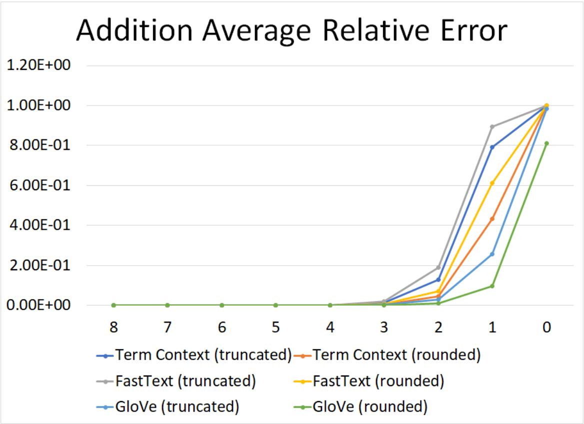


Figure 6: Average Relative Error (y-axis) of Addition Composition at various precisions ϕ (x-axis).

Φ	50d (vec)	50d (int)	% comp.	300d (vec)	300d (int)	% comp.
1	72370294	53660860	25.85	409761382	362331318	11.58
2	93553333	70375007	24.78	541234251	421515395	22.12
3	113672109	89863047	20.95	662404899	522909838	21.06
4	133684216	109772868	17.89	782523349	640203015	18.19
5	153685462	129763285	15.57	902535343	759975428	15.80
6	173685641	149762224	13.77	1022536755	879944768	13.94
7	193685703	169762728	12.35	1142537169	999941457	12.48
original	171350515	-	-	1037965801	-	-
zipped	69182687	-	-	394362421	-	-

Table 3: Size (in bytes) of 50-dimensional (50d) and 300-dimensional (300d) GloVe dataset (400k terms) at various precisions, stored as integer vectors (vec) and their CRT representations (int). % comp. shows the level of compression obtained.

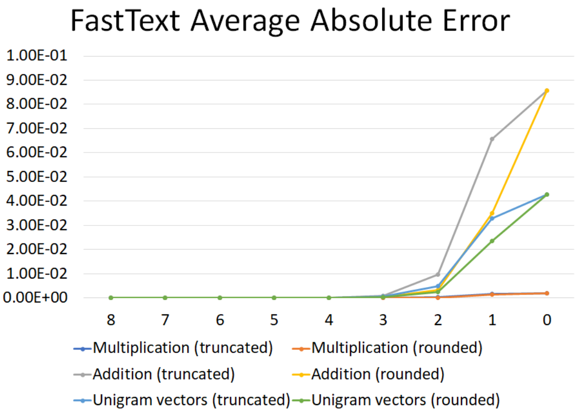


Figure 7: Average Absolute Error (y-axis) for fastText at various precisions ϕ (x-axis).

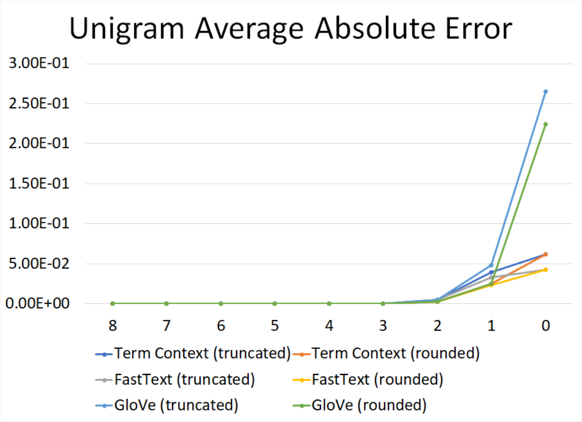


Figure 9: Average Absolute Error (y-axis) for unigram vectors at various precisions ϕ (x-axis).

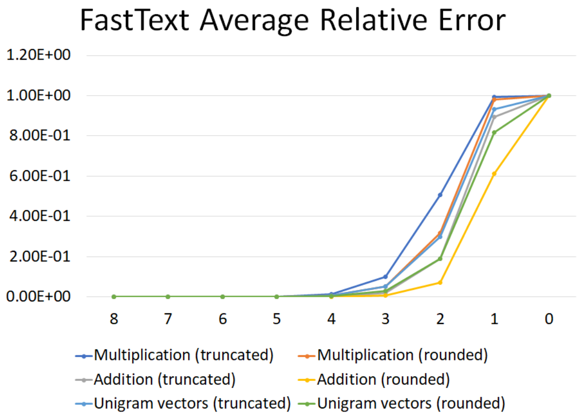


Figure 8: Average Relative Error (y-axis) for fastText at various precisions ϕ (x-axis).

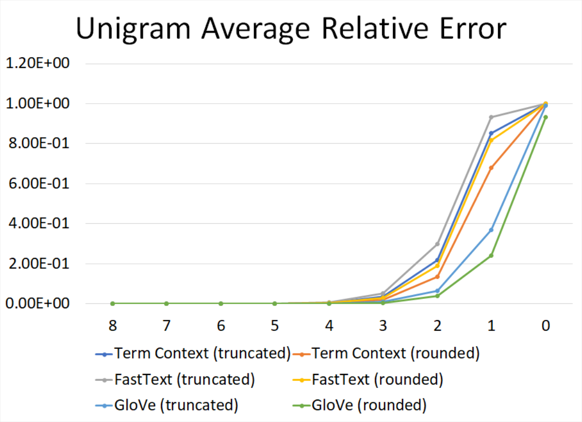


Figure 10: Average Relative Error (y-axis) for unigram vectors at various precisions ϕ (x-axis).

Φ	**50d (vec)**	**50d (int)**	**50d Up. Bd.**
1	72370294	53660860	53756606
2	93553333	70375007	138556818
3	113672109	89863047	90156697
4	133684216	109772868	110156747
5	153685462	129763285	130156797
6	173685641	149762224	150156847
7	193685703	169762728	170156897

Table 4: Size in bytes of 50-dimensional GloVe embeddings dataset, their CRT-compression and the upper bound size of that CRT-compression for any possible 50-dimensional vectors, at various precisions Φ.

Φ	**50d**	**300d**
-1	0.4908	0.5001
0	0.7571	0.7320
1	0.7730	0.7902
2	0.7745	0.7900
5	0.7730	-
full	0.769	0.7942

Table 5: Performance when varying the precision Φ of input GloVe embeddings to sentence-level sentiment analysis using a CNN.

BERT. Kim (2014) is the same work that Arora et al. (2020) used to compare with BERT over 6 different datasets. Of those datasets, we choose to analyze the variation in performance on the MR dataset (movie reviews – one sentence per review) Pang and Lee (2005) and on the opinion polarity detection subtask of the MPQA dataset (Wiebe et al., 2005).

In addition to comparing BERT-based sentiment analysis results with CNNs that take full-precision 300-dimensional GloVe vectors as inputs (which Arora et al. (2020) does), we verify the feasibility of this use case, by conducting experiments with varying precisions of 50-dimensional and 300-dimensional GloVe embeddings. Table 5 shows that sentence-based sentiment analysis with GloVe embeddings remains almost unchanged even at $\Phi = 1$ on the MR dataset, on which Arora et al. (2020) shows that BERT outperforms CNNs by 6.9. BERT therefore outperforms the CNN with 50d GloVe embeddings at precision $\phi = 1$ (which achieves a performance of 77.3) by 8.9. However, GloVe often matches BERT in this task when the BERT embeddings are trained on 16 times less data. Arora et al. (2020) also show that BERT outperforms GloVe on the MPQA dataset by a measly 0.9. We determine that BERT outperforms the CNN with 50d GloVe embeddings at precision $\phi = 1$ (which achieves a performance of 88.46) by only 1.14.

As is mentioned in Arora et al. (2020), it takes "440MB to store BERT$_{\text{BASE}}$ parameters, and on the order of 5-10 GB to store activations[, while p]retrained non-contextual embeddings (e.g., GloVe) require $O(nd)$ to store a n-by-d e-bedding matrix (e.g., 480 MB to store a 400k by 300 GloVe embedding matrix)." Our method manages to reduce this embedding matrix size by 25%. MobileBERT uncased, which is made up of 23.21% of the number of parameters of BERT$_{\text{BASE}}$, ends up with a storage size of 139 MB (compare with 421 MB for BERT$_{\text{BASE}}$ uncased). That resource-constrained model is still 2.62x larger than our 53.66MB compressed GloVe embedding matrix, and without the benefit of our significant performance gain. Indeed it is difficult to imagine how a method that manipulates 512-bit vectors could compete with adding integers. MobileBERT's performance scores are within a couple of percentage points from those of BERT$_{\text{BASE}}$.

6 Discussion and Future Work

These preliminary results suggest that the vec2int algorithm would be an efficient way of encoding word vectors for specific NLP tasks, namely those which would benefit from arithmetic-supporting vector compression (which `tar` and `zip` are not). They also suggest that analysis of average relative and absolute error can be used to tune these representations. We have yet to test the effectiveness of using integer representations of word2vec (Mikolov et al., 2013) and BERT (Devlin et al., 2018), but expect the outcomes to be similar to the results for term context vectors, fastText, and GloVe. It would also be particularly interesting to see what kind of performance improvements can be obtained by applying the CRT to sentence- as well as document-level embeddings.

Relevant to privacy concerns, some constraints on homomorphic encryption schemes include limiting the number of possible multiplications and also limiting the precision of the values being computed upon. The same sort of numerical precision analysis and computational limitations that we have done in this paper can inform how we as

---

`sentence`

NLP scientists think of how our methods are integrable into certain privacy technology. Average relative and absolute error might be useful for creating better encrypted NLP algorithms using homomorphic encryption, since many homomorphic encryption schemes tend to require integer inputs or limited-precision inputs.

Acknowledgements

We are grateful to Professors Christina Christara and Sergey Gorbunov for their feedback, as well as to Shima Asaadi, Dr. Saif Mohammad, and Dr. Svetlana Kiritchenko for making their code available and for their responsiveness to our questions. This work was supported by an RBC Graduate Fellowship, the BRAIN Ontario Research Fund, and the Vector Institute.

References

Simran Arora, Avner May, Jian Zhang, and Christopher Ré. 2020. Contextual embeddings: When are they worth it? *arXiv preprint arXiv:2005.09117*.

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

Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. 2018. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *arXiv preprint arXiv:1802.08232*.

Bently A Crane and JA Githens. 1965. Bulk processing in distributed logic memory. *IEEE Transactions on Electronic Computers*, (2):186–196.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Michael J Flynn. 1966. Very high-speed computing systems. *Proceedings of the IEEE*, 54(12):1901–1909.

Craig Gentry, Shai Halevi, and Nigel P Smart. 2012. Fully homomorphic encryption with polylog overhead. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 465–482. Springer.

Michael T Heath. 2018. *Scientific computing: an introductory survey*, volume 80. SIAM.

Herbert Hellerman. 1966. Parallel processing of algebraic expressions. *IEEE Transactions on Electronic Computers*, (1):82–91.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Svetlana Kiritchenko and Saif M. Mohammad. 2016. Capturing reliable fine-grained sentiment associations by crowdsourcing and best–worst scaling. In *Proceedings of The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, San Diego, California.

Svetlana Kiritchenko and Saif M Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. *arXiv preprint arXiv:1712.01765*.

Chu-Hsing Lin, Chin-Chen Chang, and Richard C. T. Lee. 1992. A record-oriented cryptosystem for database sharing. *The Computer Journal*, 35(6):658–660.

Shaoshi Ling, Yangqiu Song, and Dan Roth. 2016. Word embeddings with limited memory. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 387–392.

Yanjun Liu, Chin-Chen Chang, and Shih-Chang Chang. 2014. An access control mechanism based on the generalized aryabhata remainder theorem. *IJ Network Security*, 16(1):58–64.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. *proceedings of ACL-08: HLT*, pages 236–244.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Kenneth H Rosen. 2000. *Elementary number theory and its applications*. Addison Wesley Longman, Inc.

Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

DL Slotnick, WC Borck, and RC McReynolds. 1963. The solomon computer–a preliminary report. In *Proc. 1962 Workshop on Computer Organization*, page 66. Washington, DC, Spartan.

Nigel P Smart and Frederik Vercauteren. 2014. Fully homomorphic simd operations. *Designs, codes and cryptography*, 71(1):57–81.

Karl Stratos. 2017. Reconstruction of word embeddings from sub-word parameters. *arXiv preprint arXiv:1707.06957*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2019. Near-lossless binarization of word embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7104–7111.

Peter D Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 680–690. Association for Computational Linguistics.

Stephen H Unger. 1958. A computer oriented toward spatial problems. *Proceedings of the IRE*, 46(10):1744–1750.

Dominic Widdows and Kathleen Ferraro. 2008. Semantic vectors: a scalable open source package and online technology management application. In *LREC*.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Song Y Yan. 2002. *Number theory for computing*. Springer Science & Business Media.

Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*.

Jian Zhang, Jiyan Yang, and Hector Yuen. 2018. Training with low-precision embedding tables.

Don't Change Me! User-Controllable Selective Paraphrase Generation

Mohan Zhang[1,2], **Luchen Tan**[1], **Zhengkai Tu**[1], **Zihang Fu**[1], **Kun Xiong**[1], **Ming Li**[1,3], **Jimmy Lin**[1,3]
[1]RSVP.ai, [2] University of Toronto,
[3]David R. Cheriton School of Computer Science, University of Waterloo
`zhangmo4@cs.toronto.edu, lctan@rsvp.ai, siriuself1103@gmail.com,`
`{zhfu,kun}@rsvp.ai, {mli,jimmylin}@uwaterloo.ca`

Abstract

In the paraphrase generation task, source sentences often contain phrases that should not be altered. Which phrases, however, can be context dependent and can vary by application. Our solution to this challenge is to provide the user with explicit tags that can be placed around any arbitrary segment of text to mean "don't change me!" when generating a paraphrase; the model learns to explicitly copy these phrases to the output. The contribution of this work is a novel data generation technique using distant supervision that allows us to start with a pretrained sequence-to-sequence model and fine-tune a paraphrase generator that exhibits this behavior, allowing user-controllable paraphrase generation. Additionally, we modify the loss during fine-tuning to explicitly encourage diversity in model output. Our technique is language agnostic, and we report experiments in English and Chinese.

1 Introduction

Notions of semantic similarity and paraphrase are highly context dependent. Consider "I'm looking for cheap hotels in New York." vs. "What are cheap lodging options in Beijing?": from the perspective of intent classification, both express similar intents, but from the perspective of paraphrasing in a community QA application, a user looking for the answer to one question would not find the other response helpful. This is because the location "New York" anchors the information need, and any changes to it would be unacceptable to the user.

It is not always the case that named entities are "immutable" in this respect: consider a user looking for vacation destinations in the South of France. From the perspective of an advertiser, there might be good reason to tempt the user with alternative locations such as the Italian Riviera; the user may even welcome these suggestions. Also, it is not always the case that these immutable anchors are named entities: For example, some metrics assign high similarity to antonyms, and so "cheap hotels" and "expensive hotels" might be considered semantically close, but obviously not from the perspective of an end user looking for inexpensive lodging.

Although whether or not certain words can be changed without affecting the meaning of a sentence is highly dependent on context, the user of a paraphrase generation system usually would know. Consider the application of paraphrase generation in a community QA application, where a developer wishes to automatically generate question variants to increase the chances of a semantic match: A naïve system will indeed generate "What are cheap lodging options in Beijing?" as a paraphrase to "I'm looking for cheap hotels in New York."

What if we provide the user with a way to explicitly tag portions of the input so that a paraphrase generator knows what parts of the input to repeat verbatim? For example, a simple annotation scheme like "What are cheap lodging options in $\langle$tag$\rangle$ Beijing $\langle$/tag$\rangle$?", where words between $\langle$tag$\rangle$ and $\langle$/tag$\rangle$ should *not* be paraphrased. We present a paraphrase generator that implements such tags, allowing user-controllable paraphrase generation.

On a standard sequence-to-sequence model (mBART), our contribution is a novel data generation technique via distant supervision to fine-tune a paraphrase generator that supports this tagging behavior. To our knowledge, we are the first to describe such a capability—this solves a practical problem that hinders deployment of models in real-world applications.[1] Our technique is entirely language agnostic, and we report experiments in both English and Chinese. Additionally, we modify the loss during fine-tuning to explicitly encourage diversity in the paraphrase generation process.

---

[1] This is a feature requested by many of our customers.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3522–3527
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

2 Paraphrase Generator

We treat paraphrase generation as a standard supervised sequence-to-sequence task by fine-tuning mBART-large (Liu et al., 2020). Although our technique can be applied to any sequence-to-sequence framework, we selected mBART because a multilingual model is widely available. Following standard practice, custom language tags are used to denote the desired behavior of the model, but otherwise everything in our model is language agnostic.

Building on our running example, a training pair might be ("What are cheap lodging options in ⟨tag⟩ Beijing ⟨/tag⟩?", "I'm looking for cheap hotels in ⟨tag⟩ Beijing ⟨/tag⟩?"). Multiple such corpora exists (without these tags), and this is a straightforward use of mBART, but the challenge is this: Starting with an existing paraphrase dataset, how do we know where the tags should go? At inference time, the user supplies the tags, but during training, such knowledge is not available. Obviously, we could go through and manually insert tags to existing paraphrase datasets, but such large annotation efforts are impractical. Instead, we adopt a solution based on distant supervision by building three different taggers, discussed below.

2.1 Taggers

The **Oracle Tagger** represents an upper bound: tags are assigned to surround consecutive word sequences that appear in both the source and multiple references. Our intuition is that tags should go around exactly the portions of text that do not change across the paraphrases. Here, we heuristically filter out stopwords and other common n-grams. However, the Oracle Tagger won't work on datasets like QQP, where each sentence only has one reference. The two similar sentences are going to largely overlap with each other, so the Oracle Tagger will almost tag the entire sentence, which cannot be regarded as the anchors. For datasets like MSCOCO, since there are multiple references, the overlapping substrings are only a few words long therefore we can treat them as anchors.

With the **NER Tagger**, we simply tag all NERs. Since we aim to generate paraphrases in multiple languages, we use the ID-CNN language-independent named entity recognizer (Strubell et al., 2017). With the **Auto Tagger**, we use the output of the oracle tags to train a standard BERT-based (Devlin et al., 2019) token classifier.

2.2 Encouraging Diversity

During fine-tuning, our paraphrase generator learns to keep the content between ⟨tag⟩ and ⟨/tag⟩ tokens—provided by one of the three taggers above—since the content inside are not changed from a source to its reference output. We want our model to be able to keep the anchors (that are tagged) but paraphrase the other parts as much as possible.

To accomplish this, in addition to the original mBART architecture, we add another loss term to encourage diversity in generated paraphrases. During fine-tuning, we also minimize the mutual information between our paraphrase distribution and the source sentence. The mutual information term is controlled by a hyperparameter weight, indicating how "different and diverse" we want our paraphrases to be compared to the source sentence. By default, this weight is set to 0.3.

To be more specific, let $|\mathcal{D}|$ be the number of tokens in our dictionary, ϵ be the weight of label smoothing, w be the weight of the entropy term in our mutual information evaluated with source sentences. Note that a larger w means the more diversity we are encouraging our paraphrases to be. Let B be the batch size, $p(t|s_i)$ be the ground truth one-hot token probability distribution of reference sentences given a source sentence s_i where $i \in \{1, 2, ..., B\}$, and let $q_\theta(t|s_i)$ be the model predicted token probability distribution given a source sentence s_i for a fixed set of parameters θ. Our loss function can be written as:

$$\begin{aligned}\mathcal{L} =&(1-\epsilon)[-\sum_i p(\mathbf{t}|\mathbf{s_i}) \log \mathbf{q}_\theta(\mathbf{t}|\mathbf{s_i})]\\ &+\sum_i [-\frac{\epsilon}{|\mathcal{D}|} \log q_\theta(\mathbf{t}|\mathbf{s_i})]\\ &-w[-\sum_i p(\mathbf{t} == \mathbf{s_i}) \log \mathbf{q}_\theta(\mathbf{t}|\mathbf{s_i})]\end{aligned} \qquad (1)$$

where $p(\mathbf{t} == \mathbf{s_i})$ can also be viewed as a vector of results of an indicator function; the value of such a vector is one at the jth location if and only if for a source sentence s_i with length J, t_j and $s_{i,j}$ are of the same token for $j \in 1, 2, ..., J$.

3 Experimental Setup

We describe the data we used for experiments and the basic setup. At a high level, we have two types of datasets, with or without paraphrase clusters (sentences with the same meaning) to evaluate our tagging behavior and model output diversity improvement. Three English paraphrase cluster datasets were utilized, but no Chinese cluster datasets since we did not find such a dataset to be available. One English and one Chinese paraphrase pair dataset were used. Some of the datasets are widely used in paraphrase generation tasks, the others are adopted from other natural language tasks. Below we briefly describe each dataset:

MS COCO Captions 2017 (MSCOCO) (Chen et al., 2015) is an English dataset designed for automatic image caption generation. Five human-written caption descriptions were collected for each image. The total number of images is about 118k with 590k captions.

Parabank (Hu et al., 2019) is an English sentence re-writing dataset. We took its eval dataset (ParabankEval), which contains 400 semantic paraphrase with an average of 14 paraphrases in each cluster.

ComQA is an English dataset of real user questions from the WikiAnswers community QA platform. The questions are grouped into paraphrase clusters, of which 1,809 clusters have more than one question.

Quora Question Pairs (QQP) contain human-annotated duplicate English questions, with 50k training paraphrase pairs and 20k testing instances.

ATEC is a Chinese dataset that comes from Ant Technology Exploration Conference Developer competition. It contains 14,946 financial question pairs that are semantically similar.

For MSCOCO, ParabankEval, and ComQA, we randomly picked one sentence from each cluster as the paraphrase source sentence and the rest as the ground-truth references. For the other datasets, we pick one sentence as the source and the other one as the reference. We divided all datasets with 80% in the training set and 20% in the test set.

4 Results

As a preface to our results, we emphasize that, to our knowledge, user-controllable paraphrasing in the manner that we have described is a novel capability. That is, no previous work has addressed this problem—and thus, points of comparison are limited. Further note that the point of our technique is *not* to establish state-of-the-art performance on these various datasets, but rather to illustrate our tagging feature. Nevertheless, it is worth noticing that our model outperforms the previous state-of-the-art paraphrase generation model (Fu et al., 2019) on MSCOCO and QQP datasets in terms of ROUGE scores. We only present the results of ROUGE 2-gram scores for brevity, but the results of other ROUGE scores are consistent.

We choose ROUGE (Lin, 2004) instead of BLEU (Papineni et al., 2002) as our evaluation metrics for both tagger performance and diversity evaluation because of the diversity loss that we introduced. In an ideal dataset, the references of a source sentence should be able to cover all possible paraphrases. In this case, precision-based metrics like BLEU should be an even more suitable evaluation metric of generated paraphrases. However, the datasets we experimented on only contain a handful of references, far from covering all possibilities. The substituted words and diversified descriptions encouraged by our minimized mutual information will all be false negatives since they are not covered by the reference, thus leading to low precision.

4.1 Tagger Performance

Experimental results and descriptions are shown in Table 1. Compared to 118k semantic clusters in MSCOCO, ComQA and ParabankEval contain only 400 and 1809 clusters, respectively. Therefore, only MSCOCO has enough data to train a BERT-based token classifier as its Auto Tagger, while ComQA and ParabankEval's Auto Tagger results are gibberish.

How should we read the results in Table 1? We need to consider the "R" row and the "R vs. S" row together. The values in the "R" row are 2-gram ROUGE scores; the higher the score, the more similar our generated paraphrase is to the reference sentences, and therefore the "R" score means how good our paraphrases are—in other words, *quality*. The values in the "R vs. S" row are 2-gram ROUGE scores of the generated paraphrase with respect to the source; the higher the score, the more similar our generated paraphrase is to the source sentence—in other words, *diversity*. We desire both high quality and high diversity, in other words, a high "R" score and a low "R vs. S" score, but this is dataset dependent: if the references are similar to the sources (like in ParabankEval), both "R" scores and "R vs. S" scores will inevitably be high; if the references are very different from the sources (like in the MSCOCO caption dataset), both "R" scores and "R vs. S" scores will tend to be low.

Dataset	MSCOCO				ComQA			ParabankEval			QQP		ATEC	
Tagger	No Tag	Auto	NER	Oracle	No Tag	NER	Oracle	No Tag	NER	Oracle	No Tag	NER	No Tag	NER
R	20.6	30.7	28.8	22.8	38.0	42.2	47.1	30.4	77.4	79.9	35.2	38.3	35.2	36.3
R vs. S	39.5	34.6	30.5	27.9	69.5	49.4	52.4	41.5	81.5	85.8	57.5	51.3	85.4	83.3
T%		99.7	99.3	99.9		91.3	92.7		99.1	100		80.2		96.1

Table 1: Experiment results. Reading horizontally, the Tagger columns show the tagger (Auto Tagger, NER Tagger, Oracle Tagger, or no tagger) used during training for each dataset. Note that the Oracle Tagger is a realistic condition in this context because in a deployed application, the correct tags would be supplied by a human. The first row ("R") refers to the result of 2-gram ROUGE scores of the generated paraphrases against references; this measures the *quality* of the paraphrases. The second row ("R vs. S") refers to the 2-gram ROUGE scores of generated paraphrases against the source sentence; this measures the *diversity* of the paraphrases. In other words, the lower this value, the more diverse the paraphrases are. It is important to present both figures because paraphrase generation requires a balance between these two factors. The final row ("T%") shows the percentage of tagged substrings that remain unchanged during paraphrase generation; this demonstrates how well our model learns to preserve the content surrounded by tags. We can see that our tagging technique achieves the intended effect after fine-tuning.

Why don't we use BLEU (Papineni et al., 2002) score instead? Consider again generating paraphrases for our running example: "What are cheap lodging options in ⟨tag⟩ Beijing ⟨/tag⟩?" We want our generated paraphrases to retain "Beijing" and change other parts of the sentence as much as possible (but preserving the semantic meaning). Let's suppose we have ground-truth references that contain all possible paraphrases. If so, the paraphrase "I'm visiting Beijing and trying to find cheap hotels." will have a perfect "R" score and also a 0 "R vs. S" score (since it does not have any 2-gram overlaps with the source sentence); this will be the case with either ROUGE or BLEU. For paraphrases like "I'm visiting New York and trying to find cheap hotels." or "I'm visiting Beijing and trying to find the Great Wall.", the "R" score will be lower. A paraphrase like "What are cheap hotels in Beijing?" will have a higher "R vs. S" score (lower diversity). These are exactly the behaviors we desire.

More realistically, though, our references only contain a handful of paraphrases. For example, what if the term "cheap hotels" never appears in our references? Then, even though we know that "cheap hotels" is a valid paraphrase of "cheap lodging options", the related 2-grams like "cheap hotels" and "hotels in" will all be false negatives. In this case, recall-based metrics like ROUGE will be more robust than precision-based metrics like BLEU. Moreover, since we introduced the mutual information loss term to force our model to change the untagged parts of a sentence, it is likely that our model will generate correct paraphrases containing terms that are not covered by the references, and those false negatives will result in artificially (and unfair) lower BLEU scores. For this reason, we argue that ROUGE is the more appropriate metric in our study.

For three out of four English datasets, our proposed tagging approach yields not only better paraphrases (in terms of matching the reference) but more diverse paraphrases as well. ATEC also yields similar behavior with the NER tagger. However, ParabankEval seems to be an outlier here: the quality of the paraphrases increases dramatically (over double the score), although the generated output is far less diverse. This is understandable since ParabankEval only has 400 semantic clusters in total and the sentences are usually two to three times longer than MSCOCO; the paraphrase generator did not see enough examples to generate diversified long paraphrases.

4.2 Cross-Lingual Transfer

Ideally, we desire a model with strong cross-lingual capabilities—for example, along the lines of previous work in tagging tasks (Wu and Dredze, 2019) and information retrieval (Shi et al., 2020). From a practical perspective, such capabilities can reduce the need for language-specific paraphrase training data. From a scientific perspective, such explorations might help reveal language-agnostic "universals" for semantics. In this section, we present two experiments that anecdotally provide some interesting observations.

In our first experiment, we fine-tuned the model only with Chinese sentence pairs. During evaluation, we feed it English sentences and ask it to generate English paraphrases. We do not provide a formal evaluation, but it appears that our model is able to generate English paraphrases fluently, including the ability to preserve tagged substrings. For example, paraphrases of the input "How do you get deleted ⟨tag⟩ Instagram ⟨/tag⟩ chats?" include the following:

> How do I get deleted ⟨tag⟩ Instagram ⟨/tag⟩ messages?
> How do I recover deleted ⟨tag⟩ Instagram ⟨/tag⟩ messages?
> How do you get deleted ⟨tag⟩ Instagram ⟨/tag⟩ messages?
> How do I get deleted ⟨tag⟩ Instagram ⟨/tag⟩ posts?

In the second experiment, we fine-tuned our model with English sentence pairs, and then feed it Chinese sentences and ask for Chinese paraphrases. In this case, the model generates a mix of English and Chinese tokens, but, interestingly, the code switching occurs in a semantically coherent manner. For example, paraphrases of sentence "吃什么东西能⟨tag⟩补肾⟨/tag⟩呀?" (translation, "What foods can fortify the kidneys?"), where "补肾" means "fortify the kidneys"):

> What is the best food to ⟨tag⟩补肾⟨/tag⟩?
> What are some foods that ⟨tag⟩补肾⟨/tag⟩?
> Which is the best food to ⟨tag⟩补肾⟨/tag⟩?
> What is the best thing to eat to ⟨tag⟩补肾⟨/tag⟩?"

When we paraphrase sentences without any tags in our second experiment, for example, "手机如何快速散热?" (translation, "How to quickly dissipate the heat of a phone?", where "散热" means "dissipate heat" and "手机" means "phone"), the paraphrases are:

> What is the best way to散热your phone?
> What is the best way to散热your 手机?
> What is the best way to散热my phone?
> What is the quickest way to散热your phone?
> What is the quickest way to散热your 手机?

Based on the second experiment, it appears that tagged Chinese tokens are in general preserved. In the absence of tags, the model is mostly performing translation, as most of the generated tokens are in English (as well as overall word order and grammar). However, as the above examples show, some Chinese tokens are idiosyncratically preserved. Most interestingly, the code switches are semantically coherent.

Although these results are at best anecdotal, it rules out obvious and low-hanging fruit in cross-lingual transfer capabilities. We suspect these observations point to the dominance of English in the pretraining of mBART—it seems like the case that multi-lingual capabilities are pivoting through English. Even when trained on Chinese paraphrase pairs, results suggest that they are likely mapped to English latent semantic space, and that Chinese is easily "forgotten".

5 Conclusions and Future Work

This paper tackles a practical, real-world problem in paraphrase generation that to our knowledge has not been previously addressed: there are tokens that a user might wish to preserve verbatim for a variety of reasons. We further assume, that in many scenarios, the user knows exactly what those tokens are. This leads to our relatively straightforward solution—we provide the user with tags whose semantics are "don't change me".

The contribution of our work is a language-agnostic implementation of this capability using a pretrained sequence-to-sequence model (mBART), coupled with an objective that encourages diversity in the parts of the input that are not surrounded by tags. Evaluations on both English and Chinese paraphrase datasets demonstrate the empirical success of our proposed model, and additional experiments reveal interesting observations about cross-lingual transfer effects, potentially paving the way for future studies.

References

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

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

Yao Fu, Yansong Feng, and John P. Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems*.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019. ParaBank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6521–6528.

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

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318.

Peng Shi, He Bai, and Jimmy Lin. 2020. Cross-lingual training of neural models for document ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2768–2773.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. *arXiv preprint arXiv:1702.02098*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China.

Rethinking Coherence Modeling: Synthetic vs. Downstream Tasks

Tasnim Mohiuddin*¶, **Prathyusha Jwalapuram***¶, **Xiang Lin***¶, and **Shafiq Joty** *¶†
¶Nanyang Technological University, Singapore
†Salesforce Research Asia, Singapore
`{mohi0004, jwal0001, linx0057, srjoty}@ntu.edu.sg`

Abstract

Although coherence modeling has come a long way in developing novel models, their evaluation on downstream applications for which they are purportedly developed has largely been neglected. With the advancements made by neural approaches in applications such as machine translation (MT), summarization and dialog systems, the need for coherence evaluation of these tasks is now more crucial than ever. However, coherence models are typically evaluated only on synthetic tasks, which may not be representative of their performance in downstream applications. To investigate how representative the synthetic tasks are of downstream use cases, we conduct experiments on benchmarking well-known traditional and neural coherence models on synthetic sentence ordering tasks, and contrast this with their performance on three downstream applications: coherence evaluation for MT and summarization, and next utterance prediction in retrieval-based dialog. Our results demonstrate a weak correlation between the model performances in the synthetic tasks and the downstream applications, motivating alternate training and evaluation methods for coherence models.[1]

1 Introduction and Related Work

Coherence is an important aspect of discourse that distinguishes a well-written text from a poorly-written one that is difficult to comprehend (Halliday and Hasan, 1976). Computational models that can assess coherence have applications in text generation and ranking, such as summarization, machine translation, essay scoring and dialog systems.

Researchers have proposed a number of formal theories of discourse coherence, which have inspired the development of many coherence models – both traditional and neural ones. Inspired by the Centering Theory (Grosz et al., 1995), the *entity* based local models (Barzilay and Lapata, 2008; Elsner and Charniak, 2011b) formulate coherence in terms of syntactic roles (*e.g.,* subject, object) of entities in nearby sentences. Another branch of models (Pitler and Nenkova, 2008; Lin et al., 2011; Feng et al., 2014) use *coherence relations* between adjacent sentences to model local coherence, inspired by the discourse structure theories of Mann and Thompson (1988) and Webber (2004). Other traditional methods include *word co-occurrence* based local models (Soricut and Marcu, 2006), *topic* based global models (Barzilay and Lee, 2004; Elsner et al., 2007), and *syntax* based local and global models (Louis and Nenkova, 2012).

Despite continuous research efforts in developing novel coherence models, their usefulness in downstream applications has largely been ignored. They have been evaluated in mainly two ways. The most common approach has been to evaluate them on *synthetic* discrimination tasks that involve identifying the right order of the sentences at the local and global levels (Barzilay and Lapata, 2008; Elsner and Charniak, 2011b; Moon et al., 2019). The other (rather infrequent) way has been to assess the impact of coherence score as an additional feature in downstream tasks like *readability assessment* and *essay scoring* (Barzilay and Lapata, 2008; Mesgar and Strube, 2018). But since the concept of coherence goes beyond these constrained tasks and domains, so should the models.

Given the recent advances in neural NLP methods, with claims of reaching human parity in machine translation (Hassan et al., 2018), fluency in summarization (Liu et al., 2017; Celikyilmaz et al., 2018), or context-consistent response generation (Zhang et al., 2020; Hosseini-Asl et al., 2020), coherence modeling of machine-generated texts, par-

---

*Equal contribution

[1] Code and data used for evaluation available at `https://ntunlpsg.github.io/project/coherence/coh-eval/`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3528–3539
April 19 - 23, 2021.

ticularly at a document-level, is now more crucial than ever (Läubli et al., 2018; Sharma et al., 2019). Traditional task-specific evaluation methods (*e.g.,* BLEU, ROUGE) may not be an accurate reflection of their real-world performance in terms of readability (Paulus et al., 2017; Reiter, 2018). However, it is unclear if existing coherence models are capable of this task, since their performance on downstream applications is rarely studied, even though that is one of the main motivations for their development.

Our main goal in this work is to assess the performance of the existing coherence models not only on standard, challenging synthetic tasks like global and local discrimination, but more importantly on real downstream text generation problems. Specifically, we investigate the performance of coherence models in three different settings:

- Traditional synthetic tasks involving discrimination of real documents from their permutations.
- Coherence evaluation for machine translations and system-generated extractive and abstractive summaries, which are more representative of real-world use cases for coherence models.
- Next utterance ranking for dialogs, which is a downstream application similar to the synthetic task of insertion, but uses conversational data from DSTC 8 (Kim et al., 2019).

We show through experiments that there is only a slight correlation between model performances on synthetic tasks and the real-world use cases. Although models perform strongly in the synthetic tasks, they show poor performance and low correlations with human judgments on distinguishing coherent machine translations and system-generated summaries from incoherent ones. They also fail to perform well on the next utterance ranking task, which is similar to the synthetic task of insertion (Elsner and Charniak, 2011b), even if re-trained with task-specific data.

However, we show that re-training the coherence models with task-specific data for machine translation evaluation leads to improved results and agreements with human judgments. This leads us to conclude that there is a possible mismatch in the task setting that is used to train coherence models. Models trained on traditional synthetic tasks do not seem to be learning features that are useful for downstream applications. We hope that our results will motivate the broadening of the standard of coherence model evaluations to include more downstream tasks, and also motivate the redesigning of the training paradigm for coherence models.

2 Coherence Models

Advancements in deep learning have inspired researchers to neuralize many of the traditional models. Li and Hovy (2014) model syntax and inter-sentence relations using a recurrent sentence encoder followed by a fully-connected layer. In a follow-up work, Li and Jurafsky (2017) use generative models to incorporate global topic information with an encoder-decoder architecture. Mohiuddin et al. (2018) propose a neural entity grid model using convolutions over distributed representations of entity transitions. Mesgar and Strube (2018) model change patterns of salient semantic information between sentences. Xu et al. (2019) propose a local discriminative model that retains the advantages of generative models and uses a smaller negative sampling space that can learn against incorrect orderings. Moon et al. (2019) propose a unified model that incorporates sentence syntax, inter-sentence coherence relations, and global topic structures in a single Siamese framework.

We benchmark the performance of five representative coherence models on the tasks discussed above. Our selected models comprise of both traditional and neural models. Moreover, two models are currently the state-of-the-art at the time of submission (Transferable and Unified Neural Model).

Entity Grid (EGRID). Barzilay and Lapata (2005, 2008) introduced the popular entity-based model for representing and assessing text coherence motivated by the Centering Theory (Grosz et al., 1995). This model represents a text with a two-dimensional array called an *entity grid*, that captures transitions of discourse entities across sentences. These local entity transitions are used as deciding patterns for text coherence; a local entity transition of length k is a sequence of $\{S,O,X,–\}^k$ representing grammatical roles (*Subject*, *Object*, *Other*, and *Absent*, respectively) played by an entity in k consecutive sentences. The *salience* of the entities, quantified by the occurrence frequency, is also incorporated to identify transitions of important entities. Elsner and Charniak (2011b) improve the basic entity grid by including non-head nouns as entities (with the grammatical role X). Instead of using a coreference resolver, they match the nouns to detect coreferent entities. In our work, we consider this version of the entity grid model.

Neural Entity Grid (NEURALEGRID). A neural version of the entity grid model was proposed by Nguyen and Joty (2017). The grammatical roles in the grid are converted into their distributed representations, and the entity transitions are modeled in the distributed space by performing convolutions over it. The final coherence scores are computed from convolved features that have gone through a spatial max-pooling operation. A global, document-level pairwise loss is used to train the model.

Lexicalized Neural Entity Grid. Mohiuddin et al. (2018) propose an improvement of the neural entity grid (LEXNEUEGRID) by *lexicalizing* the entity transitions using off-the-shelf word embeddings to achieve better generalization.

Transferable Neural Model (TRANSMODEL). In order to generalize the coherence model across domains, Xu et al. (2019) propose a transferable neural model that considers coherence at a local level, taking only adjoining sentences as input. Coupled with pre-training of the sentence encoders in a generative fashion, their model demonstrates significant improvements in performance, despite being a local coherence model.

Unified Neural Model (UNIFIEDMODEL). Moon et al. (2019) propose a unified model that captures syntax (as a proxy of intention), discourse relations, entity attention and global topic structures. The syntax is captured by incorporating an explicit language model loss. A bi-linear layer is used to capture the inter-sentential discourse relations, while light-weight convolution is used to capture the attention and topic structures.

3 Evaluation Tasks and Experiments

In this section, we present the performance of the coherence models on standard synthetic tasks (*i.e.,* Global/Local Discrimination), followed by the experiments where we apply the coherence models trained on the global discrimination task to three downstream tasks (*i.e.,*, abstractive summarization, extractive summarization, and machine translation). We then present the results of the coherence models re-trained on the next utterance ranking task.

For each of the coherence models, we conducted experiments with publicly available codes from the respective authors. The three recent methods use word embeddings: LEXNEUEGRID, TRANSMODEL and UNIFIEDMODEL use Word2vec (Mikolov et al., 2013), average GloVe (Pennington et al., 2014), and ELMo (Peters et al., 2018) embeddings respectively. We use the default settings and hyperparameters suggested by the authors.

	Sections	# Doc.	# Pairs
Train	00-13	1,378	26,422
Test	14-24	1,053	20,411

Table 1: Statistics of the WSJ news dataset used for the **Global discrimination** task.

3.1 Synthetic Tasks

Traditionally coherence models have been evaluated mostly on synthetic tasks. For comparison with previous work, we use two representative synthetic tasks to compare the coherence models.

3.1.1 Global Discrimination.

Introduced by Barzilay and Lapata (2008), in this task coherence models are asked to distinguish an original (coherent) document from its incoherent renderings generated by random permutations of its sentences. We follow the same experimental setting of the Wall Street Journal (WSJ) news dataset as used in previous studies (Elsner and Charniak, 2011b; Moon et al., 2019; Xu et al., 2019). Similar to them, we use 20 random permutations of each document for both training and testing. Additionally, we evaluate on *inverse discrimination* (Mohiuddin et al., 2018), where the sentence order is reversed to create the incoherent version.

Setup. We follow the same experimental settings of the WSJ news dataset as used in previous works (Xu et al., 2019; Mohiuddin et al., 2018; Elsner and Charniak, 2011b; Feng et al., 2014). We use 20 random permutations of each document for both training and testing, excluding the permutations that match the original one. Table 1 summarizes the data sets used in the global discrimination task. We randomly select 10% of the training set for development purposes.

Results. Table 2 presents the results in terms of accuracy on the two *global discrimination* tasks – the *standard* and the *inverse* order discrimination. We see that UNIFIEDMODEL achieves the highest accuracy on the standard order discrimination task and TRANSMODEL performs the best on the Inverse order discrimination task. The other three

Model	Emb.	Standard	Inverse
EGrid	–	81.60	75.78
NeuralEGrid	–	84.36	83.94
LexNeuEGrid	word2vec	88.51	88.13
TransModel	Avg. Glove	91.77	**99.62**
UnifiedModel	ELMo	**93.19**	96.78

Table 2: Results: Accuracies of the coherence models in the **Global Discrimination** task.

	Sections	# Doc.	# Pairs			
			$\mathcal{D}_{w=1}$	$\mathcal{D}_{w=2}$	$\mathcal{D}_{w=3}$	$\mathcal{D}_{w=1,2,3}$
Train	00-13	748	7,890	12,280	12,440	32,610
Test	14-24	618	6,568	9,936	9,906	26,410

Table 3: Statistics on the WSJ news dataset used for the **Local discrimination** task. The w denotes the number of permuted local windows in a document.

Model	$\mathcal{D}_{w=1,2,3}$	$\mathcal{D}_{w=1}$	$\mathcal{D}_{w=2}$	$\mathcal{D}_{w=3}$
EGrid	59.78	53.89	60.43	63.04
NeuralEGrid	57.49	56.74	57.11	60.0
LexNeuEGrid	56.65	58.21	58.95	58.42
TransModel	66.87	66.25	67.95	65.52
UnifiedModel	**77.07**	**67.29**	**76.12**	**81.23**

Table 4: Results: Accuracies of the models in the **Local Discrimination task**.

models use entity grids, hence they may lose the sentence-level syntactic and semantic information.

3.1.2 Local Discrimination.

Local discrimination was proposed by Moon et al. (2019). In this task, two documents differ only in a local context (windows of 3 sentences). In this case, the models need to be sensitive to local changes. We use the same WSJ dataset as used by Moon et al. (2019).

Setup. We use the same WSJ articles used in the global discrimination task (Table 1) to create our local discrimination datasets. We use the code released by Moon et al. (2019) to generate these datasets.[2] Sentences within a local **window of size 3** are re-ordered to form a locally incoherent text. Only articles with **more than 10 sentences** are included in the dataset. Table 3 summarizes the datasets. We randomly select 10% of the training set for development purposes.

Following Moon et al. (2019), we create four datasets for our local discrimination task: $\mathcal{D}_{w=1}$, $\mathcal{D}_{w=2}$, $\mathcal{D}_{w=3}$ and $\mathcal{D}_{w=1,2,3}$. $\mathcal{D}_{w=1}$ contains the documents where only one randomly selected window is permuted, $\mathcal{D}_{w=2}$ contains the documents where two randomly selected windows are permuted; $\mathcal{D}_{w=3}$ is similarly created for 3 windows. $\mathcal{D}_{w=1,2,3}$ denotes the concatenated datasets.

Results. From Table 4, we see that the UnifiedModel achieves the highest accuracy on all four datasets. A possible reason could be the loss function it uses to train the model. Unlike other models, UnifiedModel uses an adaptive pairwise ranking loss which does not penalize the locally coherent sentences. In the local discrimination task, the difference between positive and negative examples is small; they differ only in 1-3 windows, while the other parts are locally coherent. UnifiedModel's loss function can model this better.

3.2 Coherence Evaluation Tasks

We evaluate the coherence models trained on the global discrimination task on two downstream tasks: machine translation (MT) and summarization coherence evaluation. Note that both the MT and summarization data are from the same domain (news) as the original WSJ training data.

3.2.1 Machine Translation Evaluation

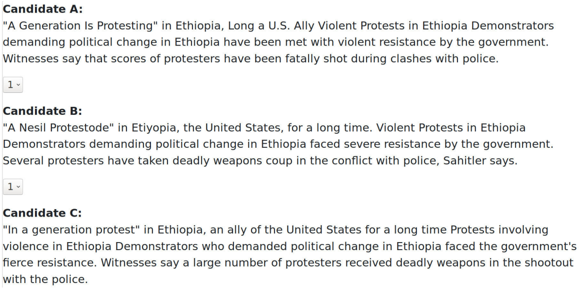
The outputs of neural machine translation (NMT) systems have been shown to be more fluent than their phrase-based predecessors (Castilho et al., 2017). However, recent studies have shown that there is a statistically strong preference for human translations in terms of both adequacy and fluency at a document level (Läubli et al., 2018; Popel et al., 2020).

Smith et al. (2016) evaluated traditional (non-neural) coherence models to see if they can distinguish a reference from a system translated document, and reported very low accuracy. However, the situation has changed with the advancements of neural models; today's coherence models are claimed to be much more accurate.

Our goal therefore is to evaluate the coherence models on how well they can judge the coherence of MT outputs at the document level. To do this, we use the system translations released by the annual Workshop (now Conference) on Machine Translation (WMT) through the years 2017 and 2018. At a document level, reference (human) translations have been shown to be more coherent than MT outputs (Smith et al., 2015, 2016; Läubli et al., 2018). Therefore, we evaluate the performance of the coherence models based on their accuracy of scoring

---

[2] https://github.com/taasnim/unified-coherence-model



Candidate A:
"A Generation Is Protesting" in Ethiopia, Long a U.S. Ally Violent Protests in Ethiopia Demonstrators demanding political change in Ethiopia have been met with violent resistance by the government. Witnesses say that scores of protesters have been fatally shot during clashes with police.

Candidate B:
"A Nesil Protestode" in Etiyopia, the United States, for a long time. Violent Protests in Ethiopia Demonstrators demanding political change in Ethiopia faced severe resistance by the government. Several protesters have taken deadly weapons coup in the conflict with police, Sahitler says.

Candidate C:
"In a generation protest" in Ethiopia, an ally of the United States for a long time Protests involving violence in Ethiopia Demonstrators who demanded political change in Ethiopia faced the government's fierce resistance. Witnesses say a large number of protesters received deadly weapons in the shootout with the police.

Figure 1: User study interface for coherence ranking.

Model	Acc. (%)	AC1 Agr.
EGrid	51.75	0.80
NeuralEGrid	54.75	0.77
LexNeuEGrid	49.34	0.76
TransModel	48.67	0.77
UnifiedModel	43.36	0.78

Table 5: **Machine Translation** setting results on WMT2017-2018 data. **Acc**uracies: % of times reference scored higher and **AC1 agr**eements for system translation rankings between annotators and models.

the reference (document) higher than the system translation (document).

We also obtain rankings given by humans in a user study. Fig. 1 shows the layout of the study, where participants were shown four sentences from three candidate translations of the same source text and asked to rank them against each other. One of the given translations is the reference, used as a control, and to validate our assumption that the reference is more coherent than the system translations. 3 participants annotated 100 such samples.

Participants chose the reference as more coherent with an agreement of **0.84**, confirming our assumption.[3] We evaluate the system translations by producing a ranking between the different translations of the same source text. To do this, we first obtain scores from the coherence models for the reference and each of the corresponding system translations. Then, we *normalize* the scores of the system translations by subtracting them from score of the reference. These normalized coherence scores are used to rank the system translations, which are then used to calculate agreements.

Setup. We use the reference and the system translations provided by WMT2017-2018 as our test data, under the assumption that the reference translations are more coherent than the system translations. This results in a testset of 20,680 reference-system translation document-pairs.

Results. We report the accuracy of the coherence models trained on the global discrimination task in distinguishing the more coherent reference text from the less coherent system translations in Table 5. We can see that most models perform worse than a random baseline of 50%, showing that their training on the global discrimination task is not helpful in detecting coherence quality in MT text. The difference in performance is particularly glaring for the TransModel and the UnifiedModel, both of which have over 90% accuracy on the global discrimination tasks, but only manage 48.67% and 43.36% on this task respectively.

We also report the agreement with human rankings on the study data in Table 5. Overall, only EGrid has good agreement with human rankings, with all other models doing similarly poorly.[4]

3.2.2 Abstractive Summarization

Generating coherent summaries has always been a goal in summarization (Nenkova and McKeown, 2011). The widely used automatic evaluation metric ROUGE (Lin, 2004) measures the n-gram overlap between the generated summaries and the reference summaries at a sentence level, and thus is not sufficient for measuring coherence. Kryściński et al. (2019) also recently found almost negligible correlation between ROUGE scores and human judgments on summary coherence, especially for abstractive summaries generated by recent neural summarization models. We therefore propose to evaluate the coherence of summaries using different coherence models and measure their effectiveness on this task.

For abstractive summarization, we use summaries from popular neural abstractive summarization systems for CNN/DM dataset (Hermann et al., 2015; Nallapati et al., 2016). Since abstractive systems vary in their architectures and loss functions, they may produce very different summaries. We run a human study to validate the rankings given by the coherence models.

---

[3] Traditional correlation measures such as Cohen's Kappa are not robust to skewed distributions of annotations, which was an issue here since the annotators were always more likely to choose the reference as better. Thus, we report the more appropriate Gwet's AC1/gamma coefficient (Gwet, 2008), which controls for this.

[4] Note that the study data is different from the test data, so the accuracies and agreements may not correlate.

Models	Abs. Agr.	Ext. Agr.
EGRID	**0.71**	0.52
NEURALEGRID	0.68	**0.70**
LEXNEUEGRID	**0.71**	0.57
TRANSMODEL	0.55	0.38
UNIFIEDMODEL	0.68	0.35

Table 6: **Abs**tractive **Agr**eement and **Ext**ractive **Agr**eement shows the AC1 agreements for the pairwise ranking of the generated abstractive summaries and extractive summaries between two annotators and the models, respectively.

Setup. We use the CNN/DM (Hermann et al., 2015; Nallapati et al., 2016) for this task. We collect the reference summaries from the CNN/DM testset as well as the summaries generated by the following four representative abstractive summarization systems: (*a*) Pointer-Generator (PG) (See et al., 2017), (*b*) BERTSUMEXTABS (BSEA) (Liu and Lapata, 2019), (*c*) UniLM (Dong et al., 2019), and (*d*) SENECA (Sharma et al., 2019).

As discussed, we directly use the coherence models trained on the WSJ dataset for the global discrimination task. The coherence models predict the scores for each system-generated summary in the testset. The scores produced by the models are then used to rank the system-generated summaries of the same original article.

We conducted a user study to validate the effectiveness of the rankings produced by the coherence models. We randomly sampled 10 sets of summaries from the dataset with each set containing four generated summaries of the same article, thus resulting in $\binom{4}{2} \times 10 = 60$ pairs of system summaries. Two annotators were asked to rank each pair of the summaries in terms of coherence; see Appendix for the human study interface.

Results. For the user study, the agreement between the two annotators was **0.78**, which indicates fairly reliable data. After we obtain the rankings based on the coherence scores produced by the models, we compute the agreements between the systems and the two annotators. From the results in Table 6, we see that EGRID and LEXNEUEGRID show the highest agreement with human judgements. However, despite strong performance in synthetic tasks, models like UNIFIEDMODEL and TRANSMODEL are unable to convert the high accuracy into high human agreement, which demonstrates the inefficiency of current synthetic tasks.

3.2.3 Extractive Summarization

For evaluating the coherence of extractive summaries, we use the dataset prepared by Barzilay and Lapata (2008) for their coherence model evaluation. The dataset comes with human ratings of the summaries from the Document Understanding Conference (DUC), 2003.

Setup. The dataset from Barzilay and Lapata (2008) provides 16 sets of summaries where each set corresponds to a multi-document cluster and contains summaries generated by 5 systems and 1 human. The human ratings for these summaries based on coherence are also available.[5]

We follow the same experimental setup as in abstractive summarization. We use the coherence models trained on the WSJ dataset to produce scores that can be used to obtain the pairwise ranking of generated summaries. Based on the ratings provided by Barzilay and Lapata (2008), we can generate the human pairwise rankings.

Results. We present the agreements between the generated human ranking and the systems in Table 6. We observe the same problem as in abstractive summarization that high accuracy in synthetic tasks does not lead to high human agreement in evaluating downstream summarization systems.

3.3 Task-specific Training for Dialog

The global and local discrimination tasks are synthetic, while the MT and summarization coherence evaluation performance may be affected by the difference between the testing and training setup. To control for this, we re-train and test the coherence models on a task-specific setup for next utterance ranking. This task has the advantage of being non-synthetic while providing task specific training data, but also being similar to the synthetic task of insertion, helping us evaluate the generalizability of the coherence model performance.

3.3.1 Next Utterance Ranking

The quality of a dialog depends on various conversational aspects such as engagement, coherence, coverage, conversational depth, and topical diversity (See et al., 2019). Liu et al. (2016) show that commonly used metrics such as BLEU and ROUGE show very weak or no correlation with human judgements. They also suggest using metrics

---

[5] See Appendix for details. Rankings are available at `http://homepages.inf.ed.ac.uk/mlap/coherence/`

	Train	Dev	Test
Advising dataset			
# of conv.	50,535	500	269
# of coh.-incoh. pairs/conv.	20	99	99
# of total example pairs	10,10,700	49,500	26,631
Ubuntu dataset			
# of conv.	49,387	500	1078
# of coh.-incoh. pairs/conv.	20	99	99
# of total example pairs	9,87,740	49,500	1,06,722

Table 7: Statistics of the refined Advising and Ubuntu datasets for the **utterance ranking** task.

that take dialog context into account. This is particularly important as Sankar et al. (2019) empirically show that current neural dialog systems rarely use conversational history. We therefore propose to evaluate the usefulness of coherence models in dialog systems.

We evaluate the models on the Noetic End-to-End Response Selection Challenge II (NOESIS II), a track in the Dialog System Technology Challenges 8 (DSTC 8) (Kim et al., 2019). In this problem, each example consists of a conversational context $U = (u_1, \ldots, u_{|U|})$ and a set of potential utterances (candidates) $C = \{c_1, \ldots, c_{|C|}\}$ that may occur next in the dialog; the task is to select the correct next-utterance $r \in C$.

This task is a nice fit for evaluating coherence models, as a good model should rank a coherent dialog higher than an incoherent one. The correct utterance along with the conversational context forms the coherent example $P = (u_1, \ldots, u_{|U|}, r)$, while other candidate utterances $c_j \in C$ with the conversational context form the incoherent examples $N = (u_1, \ldots, u_{|U|}, c_j)$. This is a considerably harder task as the difference between coherent and incoherent dialog is only the last utterance. We train the coherence models with these coherent (P) and incoherent (N) examples. The trained models give a score for each example based on its coherence. We then use our aforementioned assumption (coherence models should score P higher than N) for the evaluation. This task resembles the (synthetic) *insertion* task (Elsner and Charniak, 2011b) in that the goal here is to find the next correct utterance for the last position.

Setup. We evaluated the coherence models on both datasets of the DSTC8 response selection track, *i.e.,* the Advising and Ubuntu datasets.[6] The former contains two-party dialogs that simulate a discussion between a student and an academic advisor, while the latter consists of multi-party conversations extracted from the Ubuntu IRC channel (Kummerfeld et al., 2019).

	R@1	**R@5**	**R@10**	**MRR**	**Acc.**
Advising dataset					
Official Evaluation					
Best	0.564	0.81	0.88	0.68	X
Median	0.14	0.37	0.51	0.26	X
Worst	0.01	0.05	0.09	0.05	X
Coherence Model					
EGrid	0.004	0.03	0.07	0.04	47.16
NeuralEGrid	0.057	0.17	0.23	0.13	56.15
LexNeuEGrid	0.046	0.17	0.26	0.13	57.66
TransModel	0.067	0.20	0.30	0.14	**66.62**
UnifiedModel	0.022	0.06	0.19	0.11	54.33
Ubuntu dataset					
Official Evaluation					
Best	0.761	0.96	0.98	0.85	X
Median	0.55	0.86	0.93	0.68	X
Worst	0.24	0.38	0.46	0.32	X
Coherence Model					
EGrid	0.007	0.05	0.09	0.05	47.48
NeuralEGrid	0.18	0.39	0.49	0.29	73.18
LexNeuEGrid	0.15	0.31	0.39	0.24	74.39
TransModel	0.045	0.14	0.26	0.12	70.94
UnifiedModel	0.035	0.17	0.33	0.13	**74.49**

Table 8: **Utterance ranking** results for different coherence models on Advising and Ubuntu datasets. **R@k** indicates Recall@k, X indicates result not shared.

For a given conversational context, the goal is to select the next utterance from a candidate pool of 100 utterances, which may or may not contain the correct next utterance. We filter the datasets to suit the settings for coherence models. In our refined datasets, we exclude the conversations that have less than 7 or more than 50 utterances in the context. To ensure that we have pairwise coherent and incoherent examples, we only include the conversations that contain the correct next utterance in the candidate pool. Table 7 shows the statistics of our refined datasets for the utterance ranking task.

Results. Table 8 summarizes the results on the refined datasets for the utterance ranking task. In the last column, we report the accuracy for the number of samples in which the coherence models score the positive sample higher than the negative one. All model performances are better than a random baseline, with UnifiedModel reaching 74.49% on the Ubuntu dataset. Note, however, that because there are 100 negative samples for every positive sample, the accuracies are skewed and not representative of actual task difficulty.

---

[6] https://github.com/dstc8-track2/NOESIS-II/

The DSTC8 challenge ranking considers the average of Recall@1, Recall@5, Recall@10 and Mean Reciprocal Rank (MRR). We report both the official evaluation results and the coherence models' performance even though the latter is tested on the refined datasets. From the results, we see that the overall performance of all the coherence models is quite poor. Despite being re-trained on task specific data, we find that coherence model performance in this task is sub-par.

4 Task-specific Training for MT

As a special use case, we report the results of re-training the coherence models using machine translation data for coherence evaluation. The aim is to investigate whether changing the usual training setup, that uses negative documents which are only small variations of the positive documents, might help coherence models learn more useful task-specific features.

Setup. Under the assumption that the reference translations are more coherent at the document level than the system translations, we train the coherence models with the reference text as the positive and the system translation as the negative document, forming a positive-negative document pair. We use the data from WMT-2011 to WMT-2015 for training (28,985 document-pairs), WMT-2016 for development (7,647 document-pairs) and the same test data (WMT-2017 to WMT-2018; 20,680 document-pairs) and study data as used for the previous experiment (§3.2.1).

Results. Table 9 reports the accuracy of the re-trained models and the results of the model ranking comparison against human rankings. Many of the models show improved performance, with the agreements increasing correspondingly. The UNIFIEDMODEL has the highest accuracy improvement by far of 34%, improving from 43.36% to 77.35%. It also has the highest agreement with human rankings at 0.82. We surmise that the model's adaptive pairwise ranking loss along with its additional language model loss boosts its performance on in-domain test data.

Model	**Acc. (%)**	**AC1 Agr.**
EGRID	48.74	0.797
NEURALEGRID	52.58	0.760
LEXNEUEGRID	56.84	0.795
TRANSMODEL	57.65	0.751
UNIFIEDMODEL	**77.35**	**0.828**

Table 9: **Re-trained MT** setting results on WMT2017-2018 data. **Acc**uracies: % of times reference scored higher and **AC1 agr**eements for system translation rankings between annotators and models.

5 Discussion

Compared to the downstream tasks of coherence evaluation in MT and extractive and abstractive summarization, the traditional global discrimination task can be considered to be a simpler task (Elsner and Charniak, 2011b), since the difference between the positive and the negative document is a permutation/re-ordering of the sentences. This may be rendering the models unable to learn features that are useful for downstream applications, which are likely to have other, different kinds of errors.

On the next utterance ranking task, the models fail to generalize and perform quite poorly despite task-specific re-training. The best model performance for the synthetic task of insertion, which is similar, also barely reaches 26% (Elsner and Charniak, 2011b; Nguyen and Joty, 2017). This indicates that the training procedures may not be providing the right setting to learn features that are generic enough to apply to tasks in a harder setup.

In the synthetic tasks, the models' self-supervision comes from distinguishing an original coherent document from its incoherent renderings generated by random permutations of its sentences. This permutation-based self-supervision tries to capture document-level language properties. However, it is quite likely that this is simply a poor approximation of real-world coherence problems. Consider for example that MT systems mostly translate at the sentence-level. Consecutive sentences may lack coherence, but if two system translations of a text are compared, the translations themselves will be in the same order for both. The coherence models are not trained for such (real-world) settings.

Another possibility is that outputs from downstream tasks have different error distributions that are captured to varying degrees by different models, since they are originally designed based on synthetic tasks. That is, models that perform very well on the permutation task might be overfitting on this task, and therefore failing to find coherence issues that are more subtle than shuffled text. Thus, we conclude that the current self-supervision for coherence modeling is not suitable for downstream coherence problems.

When re-trained on machine translation data, most of the model performances improve, implying that a different training setting may be required to make the models applicable to actual downstream tasks. This is not apparent from the evaluation results that are usually reported, which show performances crossing the 90% mark.

Elsner and Charniak (2011a) show a similar lack of generalizability and applicability of coherence models to the downstream task of chat disentanglement. Our results suggest that despite nearly a decade of research since, the standard training and testing paradigm for coherence modeling continues to be inadequate in its capability to generalize to real-world use-cases and even to similar task settings, and also fails in being indicative of real-world task performance.

6 Conclusions

We benchmark the performance of representative traditional and neural coherence models on standard synthetic discrimination tasks, and contrast this with their performance on various downstream application tasks in NLP. We show that higher accuracies on synthetic tasks do not translate into better performance on downstream tasks. We demonstrate this for real-world tasks like MT and summarization coherence evaluation, and next utterance ranking. Our results signal a need for change in the way coherence models are typically trained and evaluated.

Other downstream applications like coherence evaluation of language model generated text and tasks such as chat disentanglement are also good candidates for testing coherence models. It would be worthwhile to build a coherence testset that is independent of the training tasks and similar to downstream applications, which could be used by the community to test the generalization ability of their models. In future work, we also hope to investigate the possible training scenarios that will result in more generalizable coherence models which can be used for evaluating downstream tasks.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Main Proceedings*, pages 113–120, Boston, Massachusetts, USA. Association for Computational Linguistics.

Sheila Castilho, Joss Moorkens, Federico Gaspari, Iacer Calixto, John Tinsley, and Andy Way. 2017. Is neural machine translation the new state of the art? *Prague Bull. Math. Linguistics*, 108:109–120.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 436–443, Rochester, New York. Association for Computational Linguistics.

Micha Elsner and Eugene Charniak. 2011a. Disentangling chat with local coherence models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1179–1189, Stroudsburg, PA, USA. Association for Computational Linguistics.

Micha Elsner and Eugene Charniak. 2011b. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 125–129, Portland, Oregon. Association for Computational Linguistics.

Vanessa Wei Feng, Ziheng Lin, and Graeme Hirst. 2014. The impact of deep hierarchical discourse structures in the evaluation of text coherence. In *COLING*.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Comput. Linguist.*, 21(2):203–225.

Kilem Li Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48.

Michael Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*, chapter xx. Longman, London.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan R. Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, T. M. Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *ArXiv*, abs/1803.05567.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue.

Seokhwan Kim, Michel Galley, Chulaka Gunasekara, Sungjin Lee, Adam Atkinson, Baolin Peng, Hannes Schulz, Jianfeng Gao, Jinchao Li, Mahmoud Adada, Minlie Huang, Luis Lastras, Jonathan K. Kummerfeld, Walter S. Lasecki, Chiori Hori, Anoop Cherian, Tim K. Marks, Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, and Raghav Gupta. 2019. The eighth dialog system technology challenge.

Wojciech Kryściński, Nitish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China. Association for Computational Linguistics.

Jonathan K. Kummerfeld, Sai R. Gouravajhala, Joseph Peper, Vignesh Athreya, Chulaka Gunasekara, Jatin Ganhotra, Siva Sankalp Patel, Lazaros Polymenakos, and Walter S. Lasecki. 2019. A large-scale corpus for conversation disentanglement. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *EMNLP*.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2039–2048, Doha, Qatar. Association for Computational Linguistics.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 198–209, Copenhagen, Denmark. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, page 10.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 997–1006, Portland, Oregon. Association for Computational Linguistics.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative adversarial network for abstractive text summarization. *ArXiv*, abs/1711.09357.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China. Association for Computational Linguistics.

Annie Louis and Ani Nenkova. 2012. A coherence model based on syntactic patterns. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1157–1168, Stroudsburg, PA, USA. Association for Computational Linguistics.

W. Mann and S. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.

Mohsen Mesgar and Michael Strube. 2018. A neural local coherence model for text quality assessment. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4328–4339, Brussels, Belgium. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Muhammad Tasnim Mohiuddin, Shafiq Joty, and Dat Tien Nguyen. 2018. Coherence modeling of asynchronous conversations: A neural entity grid approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 558–568, Melbourne, Australia. Association for Computational Linguistics.

Han-Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 2019. A unified neural coherence model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, EMNLP'19, pages xx—–-xx, Hong Kong. ACL.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Ani Nenkova and Kathleen McKeown. 2011. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2-3):103–233.

Dat Nguyen and Shafiq Joty. 2017. A neural local coherence model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1320–1330. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *ArXiv*, abs/1705.04304.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT'18, pages 2227–2237, New Orleans, LA, USA.

Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 186–195, Honolulu, Hawaii. Association for Computational Linguistics.

M. Popel, M. Tomková, J. Tomek, Łukasz Kaiser, Jakob Uszkoreit, Ondrej Bojar, and Z. Žabokrtský. 2020. Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals. *Nature Communications*, 11.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 32–37, Florence, Italy. Association for Computational Linguistics.

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

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1702–1723, Minneapolis, Minnesota. Association for Computational Linguistics.

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An entity-driven framework for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3271–3282.

Karin Sim Smith, Wilker Aziz, and Lucia Specia. 2015. A proposal for a coherence corpus in machine translation. In *DiscoMT@EMNLP*.

Karin Sim Smith, Wilker Aziz, and Lucia Specia. 2016. The trouble with machine translation coherence. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 178–189.

Radu Soricut and Daniel Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, COLING-ACL '06, pages 803–810, Sydney, Australia. Association for Computational Linguistics.

B. Webber. 2004. D-LTAG: Extending Lexicalized TAG to Discourse. *Cognitive Science*, 28(5):751–779.

Peng Xu, Hamidreza Saghir, Jin Sung Kang, Teng Long, Avishek Joey Bose, Yanshuai Cao, and Jackie Chi Kit Cheung. 2019. A cross-domain transferable neural coherence model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 678–687, Florence, Italy. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*.

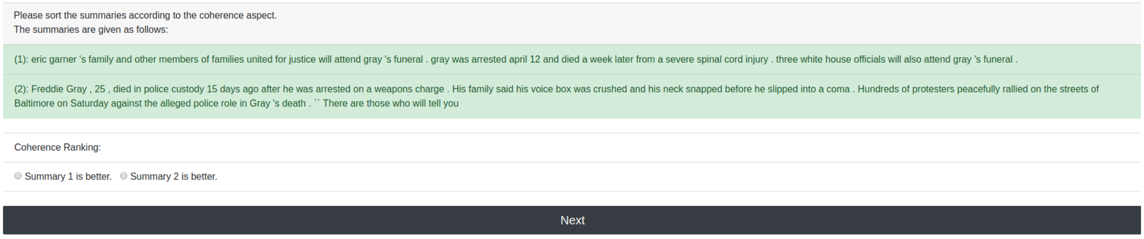
A Appendix

A.1 Human Study Interface for Abstractive Summarization

We show the interface of human study for abstractive summarization in Figure 2.

A.2 Human Study for Extractive Summarization

We briefly describe the human study for extractive summarization. The human study was conducted by Barzilay and Lapata (2008). Coherence ratings for summaries were collected during an elicitation study by 177 unpaid native speakers of English. The annotators were asked to use a seven point-scale to rate each summary based on how coherent the summaries were without having seen the source texts. The ratings (approximately 23 per summary) given by the subjects were averaged to provide a final rating score between 1 and 7 for each summary.



Please sort the summaries according to the coherence aspect.
The summaries are given as follows:

(1): eric garner 's family and other members of families united for justice will attend gray 's funeral . gray was arrested april 12 and died a week later from a severe spinal cord injury . three white house officials will also attend gray 's funeral .

(2): Freddie Gray , 25 , died in police custody 15 days ago after he was arrested on a weapons charge . His family said his voice box was crushed and his neck snapped before he slipped into a coma . Hundreds of protesters peacefully rallied on the streets of Baltimore on Saturday against the alleged police role in Gray 's death . `` There are those who will tell you

Coherence Ranking:

○ Summary 1 is better. ○ Summary 2 is better.

Figure 2: Human Study Interface for Abstractive Summarization

From the Stage to the Audience: Propaganda on Reddit

Oana Balalau
Inria, Institut Polytechnique de Paris
France
oana.balalau@inria.fr

Roxana Horincar
Thales Research & Technology
France
roxana.horincar@thalesgroup.com

Abstract

Political discussions revolve around ideological conflicts that often split the audience into two opposing parties. Both parties try to win the argument by bringing forward information. However, often this information is misleading, and its dissemination employs propaganda techniques. In this work, we analyze the impact of propaganda on six major political forums on Reddit that target a diverse audience in two countries, the US and the UK. We focus on three research questions: *i*) who is posting propaganda? *ii*) how does propaganda differ across the political spectrum? and *iii*) how is propaganda received on political forums?

1 Introduction

Propaganda, translated from Latin as "things that must be disseminated", represents information intended to persuade an audience to accept a particular idea or cause by using specific strategies or stirring up emotions. Our work is the first study that leverages a high quality annotated dataset of propaganda techniques (Da San Martino et al., 2019) to understand the impact of propaganda on online conversations.

In this paper, we perform an in-depth and long-term analysis of propaganda on online forums. We focus on six subreddits from two English speaking countries, the US and the UK, for one year. We select a popular subreddit for political news with no party affiliation and two subreddits dedicated to each country's dominant parties. In the US, the two main parties are the Democrat and the Republican party. The Democrat party is center-left; however, it contains several factions with ideologies varying from the center to the left. The Republican Party is a center-right party and has shifted in recent years towards national conservatism. In the UK, the most popular parties are the Labour Party and the Conservative Party. Similarly to the US, these parties represent the center-left and the center-right. The Labour Party has social democratic and socialist factions, while the Conservative Party has many factions, such as one-nation conservatism, liberal conservatism, or social conservatism. In recent years, both countries passed through significant political turmoil, such as Donald Trump's election in the US and the referendum on leaving the EU in the UK. However, a recent opinion piece in the Washington Post highlights an essential difference between the political discourse in the two countries. The journalist believes that the division between the left and the right in America is driven by the different interpretations the two parties give to the words "rights", "liberty" or "freedom", which have a strong moral imperative. This difference is not present in the UK, hence political parties there might find it easier to reach common grounds.

Our contribution to the study of propaganda in online discussions is in investigating the following research questions: *i*) Who is posting propaganda? *ii*) How does propaganda differ across the political spectrum or different countries? and *iii*) How is propaganda received on political forums? We believe we are the first to investigate these important questions in forums with different political leaning. For the first question, we find that media sources' political bias is a strong indicator of the tendency of using propaganda and that a smaller community of users is disproportionately spreading propagandistic articles. Regarding the second question, we find that forums dedicated to less popular parties in a country are more likely to post biased news and that cultural differences might dictate which propaganda techniques are employed. Finally, we find that if a submission or comment has more propaganda content, it might receive more user engagement, measured either as the number of comments or as upvotes and downvotes.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3540–3550
April 19 - 23, 2021.

2 Related Work

Analysis of political discussions. (Roozenbeek and Salvador Palau, 2017) explore the role of online communities in elections and how different types of new events impact their dynamics. In (Soliman et al., 2019), the authors analyze political communities (subreddits on Reddit), comparing them to the content posted, their relationships to other subreddits, and the distribution of attention received in these subcommunities. They compare left-leaning with right-leaning communities, with significant differences emerging, such as higher use of derogatory language in the right-leaning communities, stronger connectivity between the US and the European right-leaning communities, and more substantial focus on media sources reflecting their political leaning in the left-leaning subreddits. In (Guimaraes et al., 2019), the authors identify different conversation patterns that refine the notion of controversy into disputes, disruptions, and discrepancies and perform a systematic analysis of discussion threads based on essential facets of a conversation like users, sentiments, and topics. (An et al., 2019) proposes an analytical template to explore the nature of political discussions by studying the interaction and linguistic patterns within and between politically homogeneous and heterogeneous communication spaces on Reddit. (Carman et al., 2018) analyzes the effects of vote manipulation on article visibility and user engagement by comparing political threads on Reddit whose visibility is artificially increased.

Propaganda detection. Previous works on propaganda have focused on proposing datasets to foster further research, including document-level annotations (Rashkin et al., 2017; Barrón-Cedeño et al., 2019) and fragment level annotations (Da San Martino et al., 2019). Efforts for constructing annotated datasets have also been made in other European languages different from English (Kmetty et al., 2020; Baisa et al., 2019). Automatic propaganda detection approaches are almost always proposed alongside new corpora. (Rashkin et al., 2017) defines a four-class text classification task that detects propaganda, satire, hoaxes, and real news, while (Barrón-Cedeño et al., 2019) uses a binary classification to detect propaganda and non-propaganda articles. (Da San Martino et al., 2019, 2020) perform fine-grained analysis of texts by detecting all fragments that contain propaganda techniques, as well as their type. In (Kellner et al., 2020), the authors quantify the influence of trolls on Twitter that contribute to the propaganda spread during political elections in online communities. Studies on the use of propaganda have also helped understand how terrorist organizations share their ideology and attract new members (Al-Rawi and Groshek, 2020; Bisgin et al., 2019). (Martino et al., 2020) reviews the state of the art of computational propaganda detection from both an NLP and a network analysis perspective, arguing on the need to combine these communities' efforts.

Bot detection in political discussions. Research on political discussions has mostly focused on specialized topics such as adversarial debates between two parties, like election campaigns and referendums. (Rizoiu et al., 2018; Davis et al., 2016) use machine learning approaches to study social bots' influence in the diffusion of tweets containing partisan hashtags surrounding a political debate. (Hurtado et al., 2019) studies political discussions on Reddit and uses graph-based methods to reveal a fully connected community of users who exhibit a bot-like behavior. (Costa et al., 2015) introduces a generative model based on users' temporal activity patterns to study abnormal posting behavior both on Twitter and Reddit data.

Journalistic efforts in studying online content. There have been some relevant initiatives by communities of expert journalists or volunteers to raise awareness of different online news issues by evaluating the content published by news outlets and social media. For instance, Media Bias/Fact Check (MBFC) is an independent organization that analyzes media in terms of their factual reporting, bias, and propagandist content, among other aspects. Full Fact, an independent fact-checking organization in the UK, provides free tools, information, and advice for checking claims by politicians and the media. Similar initiatives have been taken by US News and World Report and the European Union.

3 Propaganda Techniques

Propaganda is a communication technique primarily used to influence public opinion towards an a-priory established agenda.

According to the Institute for Propaganda Analysis, propaganda had its definition pinned in 1938 as being "the expression of an opinion or an action by individuals or groups deliberately designed to influence the opinions or the actions of other individuals or groups with reference to predetermined ends" (for Propaganda Analysis, 1938).

In the past century, spreading propaganda required controlling traditional journalism media, such as newsprint, TV, and radio stations. It represented a form of communication that only large institutions and governments could afford. With the recent rise of the Internet and its use as online mass media, "computational propaganda" appeared (Bolsover and Howard, 2017) as a social and technical phenomenon that made propaganda campaigns easily accessible to a wide variety of small organizations and individuals that targeted audiences of unprecedented size. Recent striking examples include the propaganda allegedly set to influence the 2016 US presidential elections (Mueller, 2018) and the 2016 Brexit referendum (Howard and Kollanyi, 2016).

While the definition of propaganda has reached consensus in the literature, the complete list of techniques considered propagandist are still under discussion, Wikipedia[1] mentioning 68 of them. We adhere to the hypothesis previously made by (Barrón-Cedeño et al., 2019; Da San Martino et al., 2019) that argues that *propaganda is a communication technique that does not depend on the document topic and its topic-specific vocabulary and for which representations based on writing style, readability, and stylistic features generalize better than word-level based representations.* (Da San Martino et al., 2019) chooses to investigate a curated list of eighteen propaganda techniques found in journalistic articles that can be judged intrinsically, without the need to retrieve supporting information from external resources. Many of these techniques are also fallacies since propagandists use arguments that are sometimes convincing and not necessarily valid. A fallacy is an argument where the evidence does not support the claim that is put forward. The other techniques employ emotional language or use rhetorical, psychological, and disinformation strategies to present an idea.

We leverage the list of eighteen propaganda techniques proposed by (Da San Martino et al., 2019).

- *Appeal to authority (fallacy)* cites an expert's opinion to support an argument, without any other supporting evidence.
- *Appeal to fear or prejudice (fallacy)* supports a claim by increasing fear towards an alternative, possibly based on preconceived judgments.
- *Bandwagon (argumentum ad populum fallacy)* persuades the audience that a claim is true because many people believe so.
- *Black and white fallacy* presents only two choices out of many available, with the choice on the agenda as being the better one.
- *Causal oversimplification (fallacy of the single cause)* assumes only one cause for a complex issue out of many possible ones.
- *Flag waving (fallacy)* exploits strong patriotic feelings for a group or idea to justify an action or a claim.
- *Name calling or labeling* uses names, labels, or euphemisms to construct a good/bad image of a group or idea that is to be supported/denounced.
- *Red herring (fallacy)* presents an irrelevant, although possible convincing argument to divert the attention from the matter at hand.
- *Reductio ad Hitlerum (fallacy)* persuades the target audience to disapprove of a claim by associating it with a group widely held in contempt.
- *Straw man (fallacy)* addresses and refutes a superficially similar claim instead of the real one.
- *Whataboutism (fallacy)* discredits the opponent's claim by accusing them of hypocrisy without directly addressing the original argument.
- *Doubt* questions the credibility of an idea by disseminating negative information about it.
- *Exaggeration or minimization* makes the reality look more meaningful or more insignificant than it is.
- *Loaded language* uses words and phrases with substantial emotional implications.
- *Obfuscation, intentional vagueness, confusion (ambiguity fallacy)* deliberately employs

---

[1] https://en.wikipedia.org/wiki/Propaganda_techniques, visited October 2020

vague generalities leaving the audience to draw its interpretations.

- *Repetition* repeatedly uses the same symbol or idea to make it unforgettable.
- *Slogans* make use of brief and striking phrases to deliver the intended message.
- *Thought terminating cliches* take advantage of short, generic phrases that divert the attention or seem to offer simple answers to complex problems to stop an argument from proceeding further.

4 Reddit Dataset

We select six subreddits: Politics, Democrats, Republican, UKPolitics, LabourUK, and Tories. Politics is a subreddit for "current and explicitly political U.S. news.". The subreddit does not claim any political affiliation. The Democrats subreddit description contains "We are here to get Democrats elected up and down the ballot.", and it is a partisan subreddit. Republican is "a partisan subreddit" and the place where "Republicans discuss issues with other Republicans", hence it is a subreddit for people supporting the US Republican party. UKPolitics is a forum for "political news and debate concerning the United Kingdom" and does not claim any political affiliation. LabourUK is a subreddit that discusses breaking news concerning the British Labour Party. Finally, Tories is a subreddit for news concerning the Conservative Party in the UK, also known as the Tories. When there are several subreddits on the same topic (for example, BritishPolitics is also a subreddit for politics in the United Kingdom), we select the subreddit with the largest number of members. We note that Reddit does not ask for or encourages users to share personal data, such as their locations. Statistics on Reddit users are available only through data gathered from independent polls and surveys. For example, we know that the US and UK are the best-represented countries among the Reddit users. In the light of the surveys, we hypothesize that there are many users from the US and UK that engage in political subreddits.

We take all content posted for a period of one year, January 2019 to December 2019, from the PushShift dataset (Baumgartner et al., 2020). On Reddit, a discussion is started by a **submission**, e.g. a news article or a piece of text, and users will engage by writing **comments**. A comment is described by *author*, *body* (the content of the comment), and *score* (computed as upvotes minus downvotes) among others. We remove comments tagged as "[deleted]" or "[removed]", which are comments removed by the moderators or the users themselves. A submission has several properties, including *content* (often linked via a URL), *number of comments*, *score* (upvotes minus downvotes), and *author*. For simplicity, we refer to the submission and the article linked in the submission using the term *submission*. We retrieve the external articles by following the link in the submission. We filter out the submissions whose corresponding articles were not found by the crawler, either because cookie permissions cannot be given automatically or because the link is no longer valid. We also filter out the submissions linking to articles with less than 200 words. We want to focus on journalistic like content, a piece of text large enough to develop well an idea. Overall, we keep around $43-71\%$ of the original submissions, depending on the subreddit.

Subreddit	Submissions	Comments
Politics	317K	20M
Democrats	9.8K	54K
Republican	8.2K	41K
UKPolitics	42K	1.8M
LabourUK	7K	58K
Tories	1.1K	12K

Table 1: Reddit dataset

An overview of our dataset is given in Table 1.

To further understand the subreddits' dynamics, we report the overlap between the users commenting or posting a submission in the forums over the period we study. In the US related forums, there are $736K$ unique users, out of which $730K$ unique users in Politics, $8.5K$ in Democrats, and $7.7K$ in Republican. We have that 75% of users in Democrats and 57% of users in Republican also post in Politics, while only 5% of users posting in Republican also post in Democrats. In the UK forums, we have $46K$ unique users, out of which $44K$ post in UKPolitics, $3.3K$ in LabourUK, and $1K$ in Tories. The overlap between the forums shows a more balanced dynamics, with 61% of the LabourUK users and 63% of the Tories users also posting in UKPolitics, and 23% of the Tories users posting in LabourUK.

5 Methodology

The dataset introduced in (Da San Martino et al., 2019) consists of news articles manually annotated with propaganda techniques. The propaganda techniques are in order of frequence of instances: loaded language (2547), name-calling (1294), repetition (767), exaggeration or minimization (571), doubt (562), appeal to fear and prejudice (367), flag-waving (330), causal oversimplification (233), slogans (172), appeal to authority (169), black and white fallacy (134), thought-terminating cliches (95), whataboutism (76), reductio ad hitlerum (66), red herring (48), bandwagon (17), labeling, obfuscation or intentional vagueness (17), straw men (15). The annotations are fined-grained, with each propaganda instance being labeled at the token level. One technique might span more than one sentence.

We define two classification tasks based on the propaganda dataset described in Section 4: *i*) **propaganda identification**, which predicts if a sentence contains any propaganda techniques and *ii*) **propaganda technique identification**, which given a sentence containing propaganda, predicts the type of technique.

For each task, we test the following classifiers: a random classifier which predicts a class uniformly at random, a suite of transformer classifier BERT (Devlin et al., 2019), ROBERTA (Liu et al., 2019) and XLNet (Yang et al., 2019), and an ensemble classifier that makes a prediction based on the most confident label given by one of the three classifiers (BERT, ROBERTA or XLNet). Finally, we add the multi granularity model proposed in (Da San Martino et al., 2019), MGN ReLU. To fine-tune the transformer models, we add a final linear layer. We use a sequence length of 210, a learning rate of 0.01, a mini-batch of size 16, anneal factor of 0.5, patience of 2, and the maximum number of epochs to 20. To deal with dataset imbalance in both tasks, we weight the loss function samples according to the class weight.

The first task, propaganda identification, is a binary classification task, with classes propaganda and nonpropaganda. We present the results in Table 2. We note that propaganda identification is a difficult task, and all the classifiers obtain moderately good results, however much better than random selection.

Classifier	Precision	Recall	F1
Random	24.14	25.65	28.87
BERT	58.52	52.02	55.08
ROBERTA	**63.96**	41.41	50.28
XLNet	53.27	59.29	56.12
Ensemble	62.72	48.57	54.74
MGN ReLU	60.41	**61.58**	**60.98**

Table 2: Precision, recall and $F1$ score for propaganda identification.

The second task allowed us to understand if we have enough instances of each propaganda technique to classify them. We ran an experimental study, and we observed that bandwagon, obfuscation, red herring, straw men, and thought-terminating cliches were never recognized in the test set by our classifiers. Given this, we removed them from the annotations, and we kept the remaining techniques for the first and second tasks. We present the results in Table 3.

Classifier	F1-micro	F-macro
Random	17.07	15.76
BERT	29.75	22.17
ROBERTA	**26.96**	22.00
XLNet	29.07	**23.95**
Ensemble	28.17	22.71

Table 3: $F1$ score for technique identification.

Topical confounds. Finally, we study the effect of topical confounds in propaganda and technique classification. This analysis aims to understand if there are topical biases in the annotated dataset, which might bias our analysis. For example, if the data contains many articles on Trump, we might tend to label as propaganda any article referring to him. To identify topical biases, we use the approach presented in (Kumar et al., 2019). We first identify statistically overrepresented words in each propaganda technique in the training set and then replace them with a special token in the test set. The overrepresented words are computed using log-odds ratio with Dirichlet prior (Monroe et al., 2008), and we present the results in Table 4. We recall that we removed the techniques bandwagon, obfuscation, red herring, straw men, and thought-terminating cliches from our labeled dataset. As we can observe, for certain categories, the words are very intuitive. For example in reductio ad hitlerum we have many words related to totalitarian regimes, or in flag-waving we have many words around the notion of country. However, for most techniques, the words do not form cohesive topics, which is

Appeal to authority	thousands, regard, voter, altering, bea, notes, schema, muhammad, homosexual, viganò
Appeal to fear or prejudice	student, eucharist, jewish, easiest, bds, cliff, lew, eucharistic, campus, mcgill
Black-and-White Fallacy	easiest, uk, die, focus, burn, throw, dear, blessing, bless, chop
Causal oversimplification	anderson, backlash, cia, hillary, continued, god, alleging, knows, ruined, shahada
Doubt	guardian, story, wills, harding, assange, fake, claims, luke, failed, evidence
Exaggeration,Minimization	absolutely, history, worse, impossible, biggest, world, extraordinary, greatest, ukrainians, ingenious
Flag-Waving	american, america, europe, country, people, hungary, orban, nation, americans, citizen
Loaded Language	advantage, neo, devastating, democrats, shocking, lies, church, voice, grave, jews
Name calling, labeling	partisan, bergoglio, witch, hunt, dems, spy, assange, google, guardian, righteous
Reductio ad hitlerum	hitler, german, nazis, vichy, labour, communists, farrakhan, nazi, soviet, occupation
Repetition	muslim, san, entry, invaders, port, hungary, hat, inconvenient, orban, tijuana
Slogans	hat, character, jimenez, duke, school, america, sadikov, whataburger, foot, home
Whataboutism	admitting, guys, prosecute, focus, jihad, interpreted, ship, prosecuting, west, happened

Table 4: Top 10 words statistically overrepresented in each propaganda technique in the training set.

expected as propaganda represents a communication technique, and it is not restricted to a topic. To further verify that our classifiers learn style and not the topic, in the test set, we replace with a special token the top k words strongly associated with each technique, computed from the training set. For both $k = 10$ and $k = 20$ we report a very small decrease in $F1$ score for the BERT classifier in the propaganda classification task, from 55.08 to 52.47 and from 55.08 to 53.08. For the technique classification task, for $k = 10$ we do not observe a drop in performance, while for $k = 20$ we pass from 29.75 $F1$-micro score to 27.26, and from 22.17 $F1$-macro to 19.85. Besides, we note that the decrease in performance for this task is distributed among techniques. For flag waving and reductio ad hitlerum, for which certain words were important with respect to their definition, we do not observe a large decrease in $F1$ score. For example, the F1 scores for flag waving for $k = 10$ and $k = 20$ decrease from 43.98 to 39.57 and to 39.36, respectively. Given the small decrease in performance, we can conclude that our classifier does not learn topical confounds but the language patterns of propaganda techniques.

We leverage the propaganda identification classifier to define a **propaganda score**. The propaganda score of a document is the percentage of sentences that were labelled as containing propaganda. We compute the propaganda score of each submission, and based on the distribution of the score values in a subreddit, we define two groups: the **least propaganda**, which represents the 25% submissions with the lowest propaganda score, and the **most propaganda**, which represents the 25% submissions with the highest propaganda score. Our aim in defining the two groups is to mitigate part of the classifier's imprecision and make our analysis more robust.

6 Propaganda on Reddit

In this section, we focus on several research questions around propaganda on online forums.

RQ1. Who is posting propaganda? In the context of political forum discussions, this question targets two different groups: *media outlets and social media users*. The initial publishers are the media outlets, but users handpick what news to share on political forums. To study what media outlets are present and in which measure they are responsible for the propaganda content, we look at the groups defined in Section 5, least propaganda submissions, and most propaganda submissions. We compute the top-level domain for each submission in the two groups, which corresponds to the media outlet. We give each media outlet a label measuring its political leaning, according to MediaBiasFactCheck: center, left-center, right-center, left, right, questionable, and others. The center label is interpreted as having no or little political bias, left-center and right-center have a slight bias, left and right have a moderate bias, while the questionable label has a strong bias. The others label is given to sites that are not found in our dataset. MediaBiasFactCheck computes a media source's political bias taking into account bias by story selection, bias by omission, or bias by labeling, among others. In Table 5, we observe a strong relationship between the political bias of the media sources and the groups we computed using our propaganda score. Hence, we can infer that political bias often translates into the use of propaganda techniques.

Concerning users posting propaganda content on Reddit, we cannot link them to real entities; however, we can observe them as a community. We find

	Politics
Least Propaganda	(LeftCenter, 34.49%), (Center, 24.81%), (Left, 22.4%), (RightCenter, 3.08%), (Right, 2.91%), (Questionable, 1.44%), (Others, 10.88%)
Most Propaganda	(Left, 39.1%), (LeftCenter, 25.41%), (Center, 17.09%), (Right, 6.92%), (RightCenter, 3.3%), (Questionable, 3.07%), (Others, 5.11%)
	Democrats
Least Propaganda	(LeftCenter, 33.82%), (Left, 27.98%), (Center, 21.74%), (RightCenter, 2.76%), (Right, 0.53%), (Questionable, 0.41%), (Others, 12.78%)
Most Propaganda	(Left, 41.74%), (LeftCenter, 24.44%), (Center, 15.72%), (Right, 1.13%), (RightCenter, 1.05%), (Questionable, 0.4%), (Others, 15.51%)
	Republican
Least Propaganda	(Right, 35.94%), (Questionable, 23.69%), (LeftCenter, 7.67%), (RightCenter, 7.28%), (Center, 6.27%), (Left, 1.98%), (Others, 17.14%)
Most Propaganda	(Right, 41.58%), (Questionable, 29.28%), (RightCenter, 6.58%), (Left, 2.81%), (LeftCenter, 2.58%), (Center, 2.24%), (Others, 14.93%)
	UKPolitics
Least Propaganda	(LeftCenter, 47.66%), (Center, 10.42%), (Right, 3.84%), (Questionable, 2.22%), (RightCenter, 2.18%), (Left, 1.04%), (Others, 32.64%)
Most Propaganda	(LeftCenter, 40.65%), (Right, 11.31%), (Questionable, 6.11%), (Left, 5.33%), (RightCenter, 4.48%), (Center, 3.76%), (Others, 28.37%)
	LabourUK
Least Propaganda	(LeftCenter, 48.87%), (Left, 3.94%), (Center, 3.1%), (RightCenter, 1.69%), (Right, 0.96%), (Questionable, 0.45%), (Others, 40.99%)
Most Propaganda	(LeftCenter, 49.63%), (Left, 10.46%), (RightCenter, 2.7%), (Right, 1.74%), (Center, 1.24%), (Questionable, 0.73%), (Others, 33.5%)
	Tories
Least Propaganda	(LeftCenter, 47.18%), (Right, 9.86%), (Center, 4.93%), (Questionable, 2.11%), (RightCenter, 1.76%), (Left, 1.06%), (Others, 33.1%)
Most Propaganda	(LeftCenter, 28.87%), (Right, 27.11%), (Questionable, 5.63%), (RightCenter, 3.87%), (Center, 2.82%), (Left, 1.06%), (Others, 30.63%)

Table 5: There is a strong relation between the political bias of the media sources and the groups we computed using our propaganda score. For example, on Politics, the majority of media sources are *left center* in the least propaganda group, while the majority of sources are *left* in the most propaganda group.

that the most propaganda group's submissions are created by a smaller number of unique users than the submissions in the least propaganda group, on all subreddits except LabourUK, where we observe the opposite trend. On Politics, the least propaganda group has 9% more unique submitters, on Democrats 5%, on Republican 32%, on UKPolitics 9%, on Tories 10% while on LabourUK the most propaganda group has 28% more unique users that created a submission. This trend might indicate that certain users are more active in publishing propaganda content.

One follow-up question that we ask is how many of these users are bots. The presence of bots could explain why in one group there are fewer users posting articles. While there are several lists of Reddit bots, none of them is complete. Given this, we employ Rest-Sleep-and-Comment (RSC) (Ferraz Costa et al., 2015), a generative method that can distinguish human from bot posting activity. The method receives in input the intervals between two consecutive posts of a user, and these intervals are then compared with the aggregated distributions of intervals of all the users. The authors provide an initial training set of normal users and bots consisting of 37 bots and 999 users, to which we add 94 extra bots to make the model more robust. RSC has an average $F1$-score of 77.3 in cross-validation. The model requires at least 800 consecutive timestamps at which a user has written a comment. We retrieve from our subreddits all the users that posted a submission, and we keep the users for which we could retrieve the required number of timestamps. We note that the timestamps for a user are retrieved such that they represent consecutive chronological posts. Hence we do not restrict the subreddits in which the user might have posted. We find 748 possible bots on Politics, 91 on Democrats, 21 on Republican, 135 on UKPolitics, 23 on LabourUK and 9 on Tories. We investigate if these suspicious users posted a larger percentage of most propaganda articles in comparison with least propaganda articles. We find that this is the case on all subreddits, except Republican. However, the results are not statistically significant $(p > 0.05)$ and the differences are close, as seen in Table 6. Hence, we can conclude that the bots' automatic activity in our dataset is not necessarily linked to posting propaganda content. Also, the small percentage of content in most propaganda group published by the bots shows that the majority of the propaganda content is published by real users.

Subreddit	% articles in LP	% articles in MP
Politics	2.86	3.17
Democrats	7.70	9.13
Republican	2.36	1.95
UKPolitics	3.46	4.19
LabourUK	0.78	1.23
Tories	2.81	3.16

Table 6: The percentage of submissions posted by suspicious users in the least propaganda group (LP) and the most propaganda group (MP)

RQ2. Does propaganda differ across the political spectrum? For this analysis, we will distinguish between US-based subreddits and UK subreddits. We compare these subreddits using our propaganda score. We find that there is a statistically significant difference between the median propaganda score of articles on all subreddits in US ($p < 0.001$), with the most propagandistic content being shared on the subreddit Republican ($median = 0.307$), followed by Democrats ($median = 0.250$), and finally Politics ($median = 0.222$). In the UK subreddits, UKPolitics ($median = 0.214$) and Tories ($median = 0.217$) contains less propaganda than LabourUK ($median = 0.257$). There is no statistical difference between UKPolitics and Tories, tested using Kruskal–Wallis one-way analysis of variance, followed by Conover posthoc tests. These results indicate that *right leaning forums are not more likely to post propaganda than left leaning*. However, the tendance of using propaganda could result from the popularity of the respective party in the country. The Conservative Party in the UK has been in government since 2010, and a 2019 survey showed the party 15 points ahead of the Labour party. Even if the Republican party in the US won the White House in 2016, it didn't win the popular vote, and according to surveys more Americans identify as democrats. We also note that the subreddits that don't claim any political affiliation, Politics and UKPolitics, have less propagandistic content, which is consistent with the results in Table 5.

A second question is if the propaganda techniques employed differ according to the subreddits' political leaning or according to the country. We annotate using our propaganda technique identification classifier the sentences we previously labeled as propaganda to test this. We restrict ourselves to articles in the group most propaganda, using the intuition that if many sentences in the same article raise flags in the classifier, it is more likely that the article contains propaganda. For each subreddit, we rank the propaganda techniques by their frequency. We find that the relative ranking of techniques does not differ much between subreddits from the same country. The top 5 most frequent techniques are in the US *loaded language, name-calling, exaggeration or minimization, flag waving, doubt*, while in the UK based subreddits we have *loaded language, name calling, doubt, appeal to fear or prejudice, exaggeration or minimization*. Given the low accuracy of our technique classifier, we cannot make any definitive claims. However, such differences between the subreddits discussing politics in the two countries are plausible when considering the cultural differences. For example, Americans might be more susceptible to flag-waving, the technique of using patriotic feelings to justify an action. In 2017, 67% of Americans believed that the US is the leader of the free world according to a survey by the Public Broadcasting Service.

RQ3. How is propaganda received on political forums? To answer this last question, we aim to understand if more propaganda content will create more engagement. On Reddit, engagement is measured in the number of comments or the number of votes.

Firstly, we investigate if users comment more on submissions with higher propaganda score. We compare the median number of comments between the least propaganda group and the most propaganda group for each subreddit using the one sided Mann–Whitney U test. We find that on Politics, Democrats, Republican, UKPolitics and LabourUK, submissions in the most propaganda group receive more comments, while on Tories we observe the opposite effect.

We usually associate propaganda with media outlets. However, people can employ the same techniques to persuade the audience. We investigate how comments with propagandistic undertones are received on Reddit. For this we look at the comment's score, which is the difference between the upvotes and downvotes that a comment received. We construct two groups of comments: *positively received* comments that have the $score \geq t_{pos} \geq 10$, and *negatively received* comments with the score $score \leq t_{neg} \leq -5$. We compute the average propaganda score in the positively and negatively received groups while increasing the absolute value of the thresholds (t_{pos} from 10 to 50 and t_{neg} from -5 to -50), as shown in Figure 1. We observe that the average propaganda score of a comment increases with the engagement it generates, measured as the number of upvotes or downvotes it received. However, the trend is not observed on Republican and on Tories, one of the smaller subreddits for which we have very few data points in the plot.

7 Conclusion

In this work, we perform an extensive analysis of propaganda on online forums. We study for one year six subreddits from two English-speaking

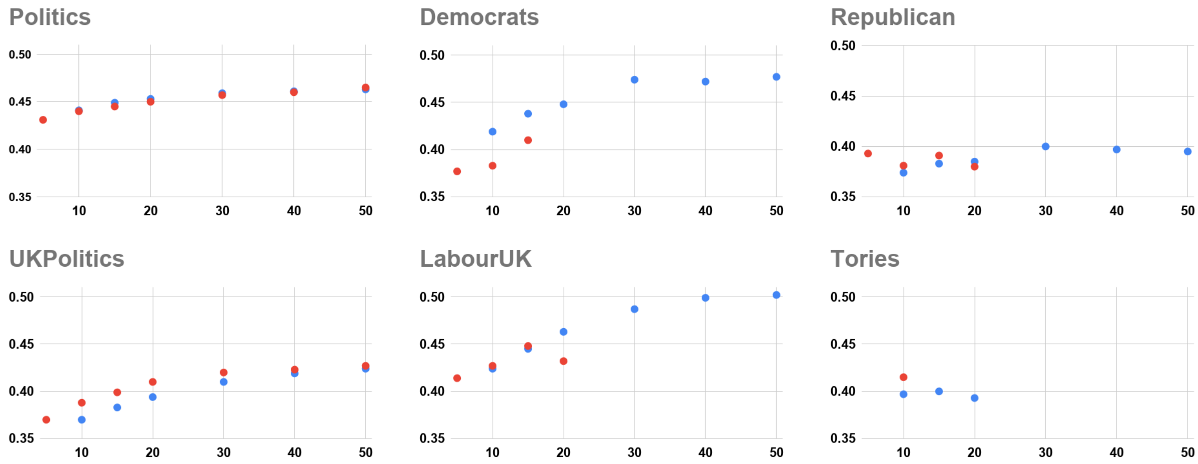


Figure 1: The average propaganda score in the positively received (blue) and negatively received (red) groups, while increasing the absolute value of the threshold. A data point represents a group with more than 100 comments.

countries, the US and the UK. We find several interesting patterns that can be leveraged by Reddit users and moderators to create better online discussions. We have found trends that we believe were not observed before in the literature. For example: *i*) the parties which represent a minority in a country might tend to use more propaganda; *ii*) political bias (either towards the right or the left) might be an indication of propaganda; *iii*) users that post more biased content form smaller communities; *iv*) differences in the use of propaganda techniques across countries might be rooted in cultural differences; *v*) submissions and comments having more propaganda content tend to receive more engagement in the form of number of comments, upvotes or downvotes. We note that while we have thoroughly tested all our hypotheses, our work is based on the automatic labelling of submissions and comments, with all the imprecision of such a method. We believe that understanding how propaganda affects us is of utmost importance for ensuring we live in democratic societies.

Acknowledgements

We thank Ioana Manolescu for our initial discussions on propaganda identification. We also thank the anonymous reviewers for their time and insightful comments.

Code. Our code is available online.

References

Ahmed Al-Rawi and Jacob Groshek. 2020. Jihadist propaganda on social media: An examination of isis related content on twitter. In *Cyber Warfare and Terrorism: Concepts, Methodologies, Tools, and Applications*, pages 1442–1457. IGI Global.

Jisun An, Haewoon Kwak, Oliver Posegga, and Andreas Jungherr. 2019. Political discussions in homogeneous and cross-cutting communication spaces. In *Proceedings of the Thirteenth International Conference on Web and Social Media, ICWSM 2019, Munich, Germany, June 11-14, 2019*, pages 68–79. AAAI Press.

Vít Baisa, Ondrej Herman, and Ales Horák. 2019. Benchmark dataset for propaganda detection in czech newspaper texts. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2019, Varna, Bulgaria, September 2-4, 2019*, pages 77–83. INCOMA Ltd.

Alberto Barrón-Cedeño, Israa Jaradat, Giovanni Da San Martino, and Preslav Nakov. 2019. Proppy: Organizing the news based on their propagandistic content. *Information Processing & Management*, 56(5):1849 – 1864.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset.

Halil Bisgin, Hasan Arslan, and Yusuf Korkmaz. 2019. Analyzing the dabiq magazine: The language and the propaganda structure of isis. In *Social, Cultural, and Behavioral Modeling*, pages 1–11, Cham. Springer International Publishing.

Gillian Bolsover and Philip Howard. 2017. Computational propaganda and political big data: Moving toward a more critical research agenda. *Big Data*, 5:273–276.

Mark J. Carman, Mark Koerber, Jiuyong Li, Kim-Kwang Raymond Choo, and Helen Ashman. 2018. Manipulating visibility of political and apolitical threads on reddit via score boosting. In *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 12th IEEE International Conference On Big Data Science And Engineering, TrustCom/BigDataSE 2018, New York, NY, USA, August 1-3, 2018*, pages 184–190. IEEE.

Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina Jr., and Christos Faloutsos. 2015. RSC: mining and modeling temporal activity in social media. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 269–278. ACM.

Giovanni Da San Martino, Shaden Shaar, Yifan Zhang, Seunghak Yu, Alberto Barrón-Cedeño, and Preslav Nakov. 2020. Prta: A system to support the analysis of propaganda techniques in the news. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 287–293, Online. Association for Computational Linguistics.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5636–5646, Hong Kong, China. Association for Computational Linguistics.

Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*, pages 273–274. ACM.

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

Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina, and Christos Faloutsos. 2015. Rsc: Mining and modeling temporal activity in social media. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 269–278, New York, NY, USA. Association for Computing Machinery.

Anna Guimaraes, Oana Balalau, Erisa Terolli, and Gerhard Weikum. 2019. Analyzing the traits and anomalies of political discussions on reddit. *Proceedings of the International AAAI Conference on Web and Social Media*, 13(01):205–213.

Philip Howard and Bence Kollanyi. 2016. Bots, #strongerin, and #brexit: Computational propaganda during the uk-eu referendum. *SSRN Electronic Journal*.

Sofia Hurtado, Poushali Ray, and Radu Marculescu. 2019. Bot detection in reddit political discussion. In *Proceedings of the Fourth International Workshop on Social Sensing*, SocialSense'19, page 30–35, New York, NY, USA. Association for Computing Machinery.

Ansgar Kellner, Christian Wressnegger, and Konrad Rieck. 2020. What's all that noise: Analysis and detection of propaganda on twitter. In *Proceedings of the 13th European Workshop on Systems Security*, EuroSec '20, page 25–30, New York, NY, USA. Association for Computing Machinery.

Zoltán Kmetty, Veronika Vincze, Dorottya Demszky, Orsolya Ring, Balázs Nagy, and Martina Katalin Szabó. 2020. Pártélet: A Hungarian corpus of propaganda texts from the Hungarian socialist era. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2381–2388, Marseille, France. European Language Resources Association.

Sachin Kumar, Shuly Wintner, Noah A. Smith, and Yulia Tsvetkov. 2019. Topics to avoid: Demoting latent confounds in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4153–4163, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeño, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4826–4832. ijcai.org.

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

Robert Mueller. 2018. *Indictment of Internet Research Agency*.

Institute for Propaganda Analysis. 1938. *Propaganda Analysis: Volume I of the Publications of the Institute for Propaganda Analysis*. Institute for Propaganda Analysis, Inc., 130 Morningside Drive, New York, N.Y.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2931–2937. Association for Computational Linguistics.

Marian-Andrei Rizoiu, Timothy Graham, Rui Zhang, Yifei Zhang, Robert Ackland, and Lexing Xie. 2018. #debatenight: The role and influence of socialbots on twitter during the 1st U.S. presidential debate. *CoRR*, abs/1802.09808.

Jon Roozenbeek and Adrià Salvador Palau. 2017. I read it on reddit: Exploring the role of online communities in the 2016 us elections news cycle. In *Social Informatics*, pages 192–220, Cham. Springer International Publishing.

Ahmed Soliman, Jan Hafer, and Florian Lemmerich. 2019. A characterization of political communities on reddit. In *Proceedings of the 30th ACM Conference on Hypertext and Social Media*, HT '19, page 259–263, New York, NY, USA. Association for Computing Machinery.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763. Curran Associates, Inc.

Probing for idiomaticity in vector space models

Marcos Garcia
CiTIUS Research Centre
Universidade de Santiago de Compostela
Galiza, Spain

Tiago Kramer Vieira
Federal University
of Rio Grande do Sul, Brazil

Carolina Scarton
University of Sheffield, UK

Marco Idiart
Federal University
of Rio Grande do Sul, Brazil

Aline Villavicencio
University of Sheffield, UK
Federal University
of Rio Grande do Sul, Brazil

marcos.garcia.gonzalez@udc.gal, tiagokv@hotmail.com,
c.scarton@sheffield.ac.uk, marco.idiart@gmail.com,
a.villavicencio@sheffield.ac.uk

Abstract

Contextualised word representation models have been successfully used for capturing different word usages, and they may be an attractive alternative for representing idiomaticity in language. In this paper, we propose probing measures to assess if some of the expected linguistic properties of noun compounds, especially those related to idiomatic meanings, and their dependence on context and sensitivity to lexical choice, are readily available in some standard and widely used representations. For that, we constructed the Noun Compound Senses Dataset, which contains noun compounds and their paraphrases, in context neutral and context informative naturalistic sentences, in two languages: English and Portuguese. Results obtained using four types of probing measures with models like ELMo, BERT and some of its variants, indicate that idiomaticity is not yet accurately represented by contextualised models.

1 Introduction

Contextualised word representation models, like BERT (Devlin et al., 2019) and ELMo (Peters et al., 2018), seem to represent words more accurately than static word embeddings like GloVe (Pennington et al., 2014), as they can encode different usages of a word. In fact, representations of a word in several contexts can be grouped in different clusters, which seem to be related to the various senses of the word (Schuster et al., 2019), and they can be used to match polysemous words in context to specific sense definitions (Chang and Chen, 2019). However, multiword expressions (MWEs) fall into a continuum of idiomaticity[1] (Sag et al., 2002; Fazly et al., 2009; King and Cook, 2017) and their meanings may not be directly related to the meanings of their individual words (e.g., *graduate student* vs. *eager beaver* as a hardworking person). Therefore, one question is whether and to what extent idiomaticity in MWEs is accurately incorporated by word representation models.

In this paper, we propose a set of probing measures to examine how accurately idiomaticity in MWEs, particularly in noun compounds (NCs), is captured in vector space models, focusing on some widely used representations. Inspired by the semantic priming paradigm (Neely et al., 1989), we have designed four probing tasks to analyse how these models deal with some of the properties of NCs, including non-compositionality (*big fish* as an important person), non-substitutability (*panda car* vs. *bear automobile*), or ambiguity (*bad apple* as either a rotten fruit or a troublemaker), as well as the influence of context in their representation. To do so, we have created the new Noun Compound Senses (NCS) dataset, containing a total of 9,220 sentences in English and Portuguese. This dataset includes sentence variants with (i) synonyms of the original NCs; (ii) artificial NCs built with synonyms of each component; or (iii) either the head or the modifier of the NC. Moreover, it is composed of naturalistic and controlled sense-neutral sentences, to minimise the possible effect of context words.

We compare five models (one static, GloVe, and four contextualised, ELMo and three BERT-based models) in English and Portuguese. The probing measures suggest that the standard and widely adopted composition operations display a limited ability to capture NC idiomaticity.

Our main contributions are: (i) the design of novel probes to assess the representation of id-

---

[1] We understand idiomaticity as *semantic opacity* and its continuum as different *degrees of opacity* (Cruse, 1986).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3551–3564
April 19 - 23, 2021.

iomaticity in vector models, (ii) a new dataset of NCs in two languages, and (iii) their application in a systematic evaluation of vector space models examining their ability to display behaviors linked to idiomaticity.

The remainder of this paper is organized as follows: First, Section 2 presents related work. Then, we describe the data and present the probing measures in Section 3. In Section 4, we discuss the results of our experiments. Finally, the conclusions of our study are drawn in Section 5.

2 Related Work

Priming paradigms have been traditionally used in psycholinguistics to examine how humans process language. For compounds, some findings suggest that idiomatic expressions are processed more slowly than semantically transparent ones, as processing the former may involve a conflict between the non-compositional and the compositional meanings (Gagné and Spalding, 2009; Ji et al., 2011). However, studies using event-related potential (ERP) data showed that idiomatic expressions, especially those with a salient meaning (Giora, 1999), have processing advantages (Laurent et al., 2006; Rommers et al., 2013). In NLP, probing tasks have been useful in revealing to what extent contextualised models are capable of learning different linguistic properties (Conneau et al., 2018). They allow for more controlled settings, removing obvious biases and potentially confounding factors from evaluations, and allowing both the use of artificially constructed but controlled sentences and naturally occurring sentences (Linzen et al., 2016; Gulordava et al., 2018). In priming tasks, related stimuli are easier to process than unrelated ones. One assumption is that, for models, related stimuli would achieve greater similarity than unrelated stimuli. These tasks have been used, for instance, to evaluate how neural language models represent syntax (van Schijndel and Linzen, 2018; Prasad et al., 2019), and the preferences that they may display, such as the use of mainly lexical information in a lexical substitution task even if contextual information is available (Aina et al., 2019).

Concerning pre-trained neural language models, which produce contextualised word representations, analyses about their abilities have shown, for instance, that they can encode syntactic information (Liu et al., 2019) including long-distance subject–verb agreement (Goldberg, 2019). Regarding semantic knowledge, the results of various experiments suggest that BERT can somewhat represent semantic roles (Ettinger, 2020). However, its improvements appear mainly in core roles that may be predicted from syntactic representations (Tenney et al., 2019). Moreover, from the representations generated by BERT, ELMo and Flair (Akbik et al., 2018) for word sense disambiguation, only the clusters of BERT vectors seem to be related to word senses (Wiedemann et al., 2019), although in cross-lingual alignment of ELMo embeddings, clusters of polysemous words related to different senses have also been observed (Schuster et al., 2019).

The use of contextualised models for representing MWEs has been reported with mixed results. Shwartz and Dagan (2019) evaluated different classifiers initialised with contextualised and non-contextualised embeddings in five tasks related to lexical composition (including the literality of NCs) and found that contextualised models, especially BERT, obtained better performance across all tasks. However, for capturing idiomaticity in MWEs, static models like *word2vec* (Mikolov et al., 2013) seem to have better performance than contextualised models (Nandakumar et al., 2019; King and Cook, 2018). These mixed results suggest that a controlled evaluation setup is needed to obtain comparable results across models and languages.

Therefore, we have carefully designed probing tasks to assess the representation of NCs in vector space models. As the same word can have different representations even in related paraphrased contexts (Shi et al., 2019), we adopt paraphrases with minimal modifications to compare the idiomatic and literal representations of a given NC.

3 Materials and Methods

3.1 Noun Compound Senses Dataset

The Noun Compound Senses (NCS) dataset is based on the NC Compositionality dataset, which contains NCs in English (Reddy et al., 2011), Portuguese and French (Cordeiro et al., 2019). Using the protocol by Reddy et al. (2011), human judgments were collected about the interpretation of each NC in 3 naturalistic corpus sentences. The task was to judge, for each NC, how literal the contributions of its component were for its meaning (e.g., "Is *climate change* truly/literally a *change* in *climate*?"). Each NC got a score, which was the average of the human judgments with a *Likert* scale from 0 (non-literal/idiomatic) to 5 (literal/compositional).[2]

For the NCS dataset, a set of probing sentences for the 280 NCs in English and the 180 NCs in Portuguese was added. For each NC, the sentences exemplify two conditions: (i) the naturalistic context provided by the original sentences (NAT), and (ii) a neutral context where the NCs appear in uninformative sentences (NEU). For the latter we use the pattern *This is a/an <NC>* (e.g., *This is an eager beaver*) and its Portuguese equivalent *Este/a é um(a) <NC>*. As some NCs may have both compositional and idiomatic meanings (e.g., *fish story* as either *an aquatic tale* or *a big lie*), these neutral contexts will be used to examine the representations that are generated for the NCs (and the sentences) in the absence of any contextual clues about the meaning of the NC. Moreover, they enable examining possible biases in the NC representation especially when compared to the representation generated for the NAT condition.

For each NC and condition, we created new sentence variants with lexical replacements, using synonyms of the NC as a whole or of each of its components. The synonyms of the NCs are the most frequent synonyms provided by the annotators of the original NC Compositionality dataset (e.g., *brain* for *grey matter*). The synonyms of each component were extracted from WordNet (Miller, 1995, for English) and from English and Portuguese dictionaries of synonyms (e.g., *alligator* for *crocodile* and *sobs* for *tears*). In cases of ambiguity (due to polysemy or homonymy), the most common meaning of each component was used. Experts (native or near-native speakers with linguistics background) reviewed these new utterances, keeping them as faithful as possible to the original ones, but with small modifications for preserving grammaticality after the substitution (e.g., modifications in determiners and adjectives related to gender, number and definiteness agreement).

NCS contains a total of 5,620 test items for English and 3,600 for Portuguese among neutral and naturalistic sentences, and it is freely available.[3]

3.2 Probing Measures

This section presents the probing measures defined to assess how accurately idiomaticity is captured in vector space models. For these measures we consider comparisons between three types of embeddings: (i) the embedding for an NC out of context (i.e. the embedding calculated from the NC words alone), represented by ϵ_{NC}; (ii) the embedding for an NC in the context of a sentence S, represented by $\epsilon_{\text{NC} \subset \text{S}}$[4] (iii) finally, the sentence embedding that contains an NC, which is represented by $\epsilon_{\text{S} \supset \text{NC}}$. Here we use the standard output of some widely used models with no fine-tuning to avoid possible interference. However, in principle, these measures could apply to any embedding even after fine-tuning.

The similarities between embeddings are calculated in terms of cosine similarity: $\cos(\epsilon, \epsilon')$ where ϵ and ϵ' are embeddings from the same model with the same number of dimensions. In NAT cases, the similarity scores for each of the three available sentences for a given NC are averaged to generate a single score. We use Spearman ρ correlation between similarities and the NC idiomaticity scores (280 for English and 180 for Portuguese) to check for any effects of idiomaticity in the probing measures. We also calculate Spearman ρ correlation between different embedding models to determine how much the models agree, and between the NAT and NEU conditions to see how much the context affects the distribution of similarities. We also analyse the distribution of cosine similarities produced by different models for each of the probing measures. All probing measures are calculated for both NAT and NEU conditions.

P1: Probing the similarity between an NC and its synonym. If a contextualised model captures idiomaticity accurately, the embedding for a sentence containing an NC should be similar to the embedding for the same sentence containing a synonym of the NC (NC_{syn}, e.g., for *grey matter*, NC_{syn} = *brain*). Thus, $\text{sim}^{(\text{P1})}_{\text{Sent}} \simeq 1$, where $\text{sim}^{(\text{P1})}_{\text{Sent}} = \cos(\epsilon_{\text{S} \supset \text{NC}}, \epsilon_{\text{S} \supset \text{NC}_{syn}})$. This should occur regardless of how idiomatic the NC is, that is, similarity scores are not expected to correlate with NC idiomaticity scores ($\rho^{(\text{P1})}_{\text{Sent}} \simeq 0$). Moreover, this should also hold for the NC and NC_{syn} embeddings generated in the context of this sentence, which means that $\rho^{(\text{P1})}_{\text{NC}} \simeq 0$ and $\text{sim}^{(\text{P1})}_{\text{NC}} \simeq 1$

---

[2] We averaged the Likert judgments for comparability with previous work, even though the median may reflect better the cases where there is more disagreement among the annotators. However, both mean and median are strongly correlated in our data: $\rho = 0.98$ (English) and $\rho = 0.96$ (Portuguese), $p < 0.001$.

[3] `https://github.com/marcospln/noun_compound_senses`

[4] For non-contextualised embeddings $\epsilon_{\text{NC} \subset \text{S}} = \epsilon_{\text{NC}}$.

Naturalistic sentence	NC	NC$_{syn}$	NC$_{synW}$
***Field work** and practical archaeology are a particular focus.*	field work	research	area activity
*The town centre is now deserted - it's almost like a **ghost town**!*	ghost town	abandoned town	spectre city
*How does it feel to experience a **close call** only to come out alive and kicking?*	close call	scary situation	near claim
*Eric was being an **eager beaver** and left work late.*	eager beaver	hard worker	restless rodent
*No wonder Tom couldn't work with him; he is a **wet blanket**.*	wet blanket	loser	damp cloak

Table 1: Naturalistic examples with their NC$_{syn}$ and NC$_{synW}$ counterparts.

where $\text{sim}^{(\text{P1})}_{\text{NC}} = \cos(\epsilon_{\text{NC} \subset \text{S}}, \epsilon_{\text{NC}_{syn} \subset \text{S}})$. The baseline similarity scores can be approximated using the out-of-context embeddings for NC and NC$_{syn}$.

P2: Probing single component meaning preservation. As the meaning of a more compositional compound can be inferred from the meanings of its individual components, we evaluate to what extent an NC can be replaced by one of its component words and still be considered as representing a similar usage in a sentence. We measure $\text{sim}^{(\text{P2})}_{\text{Sent}} = \cos(\epsilon_{\text{S} \supset \text{NC}}, \epsilon_{\text{S} \supset \text{w}_i})$ and $\text{sim}^{(\text{P2})}_{\text{NC}} = \cos(\epsilon_{\text{NC} \subset \text{S}}, \epsilon_{\text{w}_i \subset \text{S}})$, where w_i is the component word (head or modifier) with the highest similarity, as for some NCs the main meaning may be represented by either its head or modifier. Similarity scores for idiomatic NCs should be low as they usually cannot be replaced by any of its components. In contrast, for more compositional NCs, the similarity is expected to be higher. For example, while for a more compositional NC like *white wine*, the head *wine* would provide a reasonable approximation as w_i, the same would not be the case for *grey matter*, a more idiomatic NC. Therefore, we expect significant correlations between the similarity values and the NC idiomaticity scores, that is $\rho^{(\text{P2})}_{\text{Sent}} > 0$ and $\rho^{(\text{P2})}_{\text{NC}} > 0$.

P3: Probing model sensitivity to disturbances caused by replacing individual component words by their synonyms. We examine whether vector representations are sensitive to the lack of individual substitutability of the component words displayed by idiomatic NCs (Farahmand and Henderson, 2016). To compare an NC with an expression made from synonyms of its component words (NC$_{synW}$, e.g., for *grey matter*, NC$_{synW}$ = *silvery material*), we measure $\text{sim}^{(\text{P3})}_{\text{Sent}} = \cos(\epsilon_{\text{S} \supset \text{NC}}, \epsilon_{\text{S} \supset \text{NC}_{synW}})$ and $\text{sim}^{(\text{P3})}_{\text{NC}} = \cos(\epsilon_{\text{NC} \subset \text{S}}, \epsilon_{\text{NC}_{synW} \subset \text{S}})$. These substitutions should provide more similar variants for compositional than for idiomatic cases, and the similarity scores should correlate to the NC idiomaticity scores, that is $\rho^{(\text{P3})}_{\text{Sent}} > 0$ and $\rho^{(\text{P3})}_{\text{NC}} > 0$.

P4: Probing the similarity between the NC in the context of a sentence and out of context. To determine how much for a given model an NC in context differs from the same NC out of context we measure $\text{sim}^{(\text{P4})}_{\text{in-out}} = \cos(\epsilon_{\text{NC} \subset \text{S}}, \epsilon_{\text{NC}})$. We expect similarity scores to be higher in the NEU condition, given their semantically vague context, than for the NAT condition.

3.3 Calculating Embeddings

We use as a baseline the static non-contextualised GloVe model (Pennington et al., 2014) and, for contextualised embeddings, four widely adopted models: ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), and two BERT variants, DistilBERT (DistilB) (Sanh et al., 2019) and Sentence-BERT (SBERT) (Reimers and Gurevych, 2019b). For all the contextualised models, we use their pre-trained weights publicly available through the Flair implementation[5]. For GloVe, the English and Portuguese models described in Pennington et al. (2014) and Hartmann et al. (2017). For ELMo, we use the small model provided by Peters et al. (2018), and for Portuguese we adopt the weights provided by Quinta de Castro et al. (2018). For all BERT-based models, we used the multilingual models for both English and Portuguese.[6]

To have a single embedding for the whole sentence or its parts, e.g., the NC representation, we use the standard procedure of averaging the vectors of the involved tokens.[7] In GloVe and ELMo, we average the output embeddings of each word, while in BERT-based models we obtain the final vector by averaging those of the sub-tokens (e.g., 'wet', 'blank' and '##et' for *wet blanket*).

Different combinations of the last five layers were probed in BERT-based models. However, they led to qualitatively similar results, and for reasons of presentation clarity, have been omitted

---

[5] `https://github.com/flairNLP/flair`

[6] We also investigated dedicated models for English, however, for allowing a more direct comparison between the languages, we report results only for the multilingual models.

[7] We discuss other operations in section 4.6.

from the discussion. We focus on embeddings calculated from a combination of the last four layers as they have been found to be representative of the other combinations. For ELMo, as it is intended to serve as a contextualised baseline, we represent the word embeddings using the concatenation of its three layers, albeit it is known that separate layers and weighting schemes generate better results in downstream tasks (Reimers and Gurevych, 2019a).

4 Results

This section discusses our results for each probing measure, using cosine similarities and Spearman ρ correlations. A qualitative analyses is also presented where we compare BERT and GloVe results of the five NCs in Table 1 (which shows the naturalistic sentences for each NC, together with their respective NC_{syn} and NC_{synW})[8]. We also discuss the average results of other NCs in both conditions and these results and other examples can be found in the Appendix.

4.1 Can contextualised models capture the similarity between an NC and its synonym?

If a contextualised model successfully captures idiomaticity, we would expect (i) high cosine similarity between a sentence containing an NC and its variant using a synonym of the NC (P1), and (ii) little or no correlation with the NC idiomaticity score. The results confirm high similarity values for all models, as shown in Figure 1a. However, this is not the case if we consider only the embeddings in context for NC and NC_{syn}, which display a larger spread of similarity values (see Figure 1b). Moreover, contrary to what was expected, a moderate correlation was found between most models and the idiomaticity scores (P1 in Table 2), indicating lower similarity scores for idiomatic than for compositional cases, for both NAT and NEU conditions.

Even though the high $\text{sim}^{(\text{P1})}_{\text{Sent}}$ values seem to suggest idiomaticity is captured, lower $\text{sim}^{(\text{P1})}_{\text{NC}}$ and moderate correlations with idiomaticity scores contradict it. Therefore a possible explanation for high similarities for Sent may be the effect of the overlap in words between a sentence and its variant (i.e., the context in Sent). This is also compatible with the larger similarities observed for NAT than for NEU condition since the average sentence length for the naturalistic sentences is 23.39 for English and 13.03 for Portuguese, while for the neutral it is five words for both languages. Moreover, a similar performance was also obtained with GloVe.[9] It is also worth noting that, in contrast to static embeddings, contextualised word representations are anisotropic, occupying a narrow cone in the vector space and therefore tending to produce higher cosine similarities (Ethayarajh, 2019).

The results with the first probing measure show that even though the similarities can be relatively high, they are consistently lower for idiomatic than for compositional cases, suggesting that idiomaticity may not be fully incorporated in the models.

Qualitative analysis: In Table 3, in P1, the similarity scores between NC in Table 1 and their respective NC_{syn} for BERT and GloVe models are shown. As expected, BERT shows higher scores than GloVe for all cases, and even if the values for P1 differ, both models follow the same tendency. There is a larger spread for GloVe (e.g., $\text{sim}^{(\text{P1})}_{\text{wet blanket}} = 0.21$ vs. $\text{sim}^{(\text{P1})}_{\text{ghost town}} = 0.80$) which could be explained by the choices of NC_{syn}. For *wet blanket* NC_{syn} = *loser*, which has probably a very dissimilar representation from both *wet* and *blanket*. On the other hand, *ghost town* with NC_{syn} = *abandoned town* not only shares a word with the original NC, but we can also argue that *ghost* and *abandoned* are likely to have similar embeddings. Finally, the average results of P1 show that BERT-based models tend to intensify lexical overlap, resulting in high cosine similarities when both the NC and NC_{syn} share (sub-)words. For instance, 47 (in English) and 49 (in Portuguese) out of the 50 compounds with highest $\text{sim}^{(\text{P1})}_{\text{NC-NAT}}$ share surface tokens, whether the NCs are more compositional (e.g., *music journalist* vs. *music reporter*) or more idiomatic (e.g., *ghost town* vs. *abandoned town*).

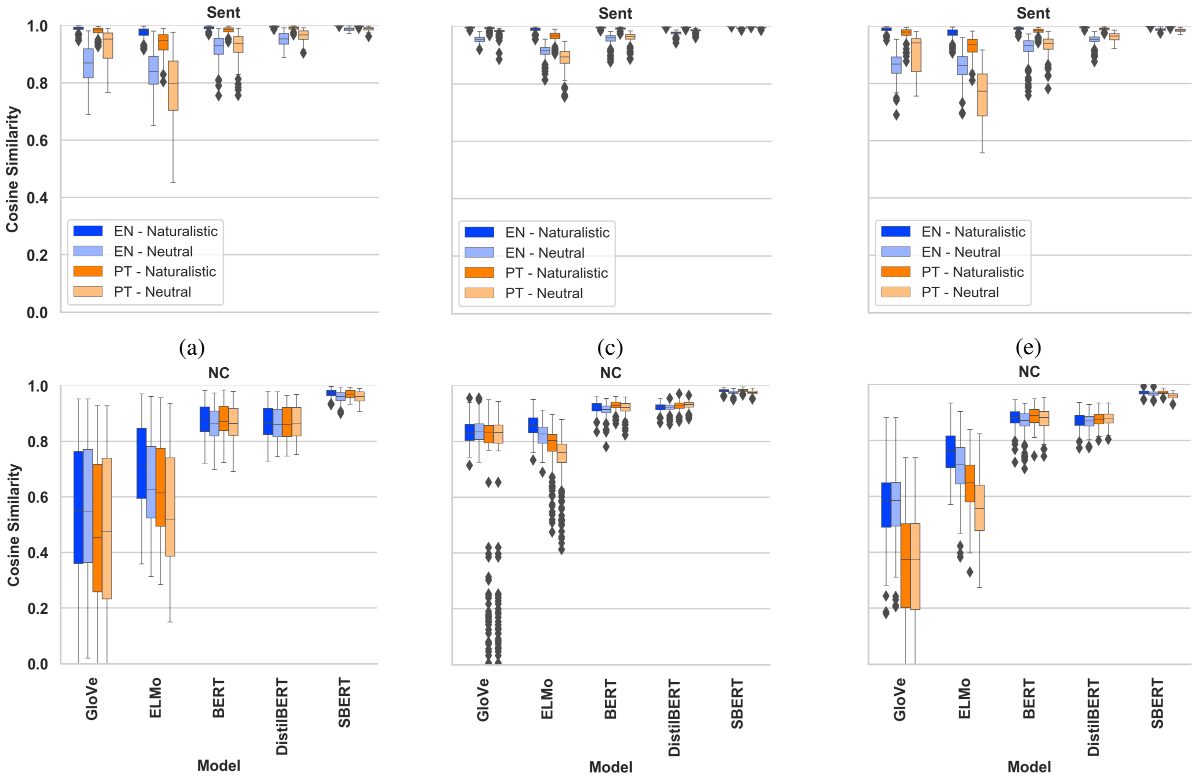
4.2 Can the lower semantic overlap between idiomatic NCs and their individual components be captured?

We would expect idiomatic NCs not to be similar to either of their individual components, which would be reflected by a larger spread of cosine similarity values for P2 than for P1. However, all models produced high similarities across the idiomaticity spectrum, see Figures 1c for Sent and 1d for NC.

---

[8] Neutral sentences are omitted since they all follow the same pattern *This is a/an* <NC>.

[9] GloVe-NC can be viewed as the baseline for the lack of contextual information.



(a) (c) (e)

(b) (d) (f)

Figure 1: Cosine similarities in English (blue) and Portuguese (orange). First column for PI (a and b), second for P2 (c and d) and third for P3 (e and f). Sentence condition at the top and NC at the bottom.

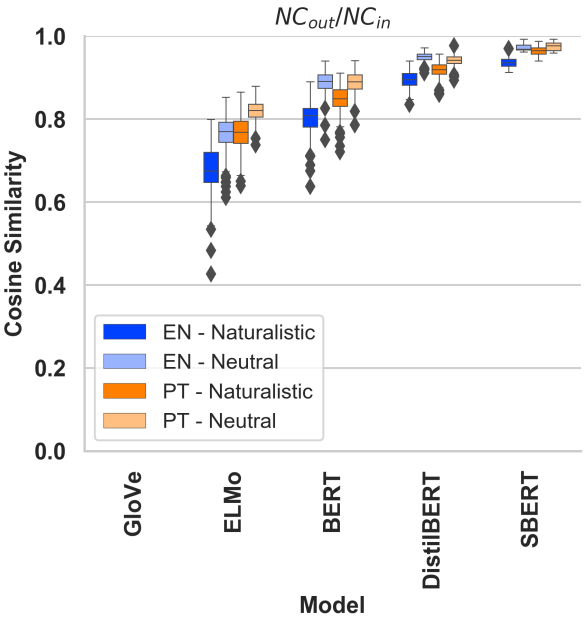


Figure 2: P4 ($\cos(\epsilon_{\text{NC} \subset \text{S}}, \epsilon_{\text{NC}})$).

The higher average similarities for P2 than for P1, compare Figures 1a and 1b with Figures 1c and 1d, reinforces the hypothesis that the models prioritise lexical overlap with one of the NC components rather than semantic overlap with a true NC synonym, even for idiomatic cases. Although there is some correlation with idiomaticity when it exists, it is lower than for P1, contrary to what would be expected (see P1 and P2 in Table 2). All of these indicate that these models cannot distinguish the partial semantic overlap between more compositional NCs and their components and the absence of overlap for idiomatic NCs.

Qualitative analysis: The P2 results in Table 3 show the highest similarity scores between each example in Table 1 and one of its components. These high similarity scores highlight the prioritisation of lexical over semantic overlap mentioned above. Furthermore, some idiomatic NCs also show strong similarities with their components, suggesting that the idiomatic meaning is not correctly represented. For instance, *poison pill* (meaning an emergency exit) has an average similarity of $\text{sim}^{(P2)}_{\text{poison pill-NAT}} = 0.94$ with its head (*pill*).

4.3 Can they capture the lack of substitutability of individual components for idiomatic NCs?

We do not expect an idiomatic NC to keep the idiomatic meaning when each of its components is

	GloVe		ELMo		BERT		DistilB		SBERT		BERTRAM	
EN_{NAT}	ρ_{Sent}	ρ_{NC}	ρ Sent	ρ NC	ρ Sent	ρ NC	ρ Sent	ρ NC	ρ Sent	ρ NC	ρ Sent	ρ NC
P1	0.31	0.62	0.43	0.60	0.51	0.67	0.38	0.58	0.30	0.43	0.14	0.30
P2	-	0.45	-	0.15	-	0.32	-	0.25	-	0.19	0.21	0.45
P3	-	0.18	-	-	-	0.21	-	0.15	-	0.20	0.18	0.39
EN_{NEU}												
P1	0.58	0.61	0.55	0.60	0.53	0.59	0.56	0.54	0.47	0.42	0.24	0.23
P2	0.29	0.44	-	0.22	-	-	-0.12	-	0.12	0.17	0.26	0.31
P3	-	0.18	-	-	-	-	-	-	0.17	0.19	0.32	0.26
PT_{NAT}												
P1	-	0.40	0.32	0.47	0.29	0.44	0.20	0.39	0.18	0.37	-	0.22
P2	-	0.20	-	0.28	-	-	-0.17	-	-	-	-	0.21
P3	-0.19	-	-	-	-	-	-	-	-	-	-	0.22
PT_{NEU}												
P1	0.22	0.41	0.37	0.47	0.30	0.35	0.31	0.37	0.30	0.36	-	0.18
P2	-	0.18	0.17	0.20	-	-	-	-	-	-	0.22	-
P3	-	-	-	-	-	-	-	-	-	-	0.22	0.18

Table 2: Spearman ρ correlation with human judgments, $p \leq 0.05$. Non-significant results omitted from the table.

Noun Compound	P1			P2			P3		
	GloVe	BERT		GloVe	BERT		GloVe	BERT	
	NAT/NEU	NAT	NEU	NAT/NEU	NAT	NEU	NAT/NEU	NAT	NEU
field work	0.58	0.92	0.92	0.86(2)	0.94(2)	0.90(2)	0.54	0.90	0.88
ghost town	0.80	0.95	0.91	0.85(2)	0.93(2)	0.91(2)	0.66	0.90	0.84
close call	0.52	0.83	0.84	0.86(2)	0.94(2)	0.91(2)	0.61	0.86	0.84
eager beaver	0.43	0.82	0.83	0.84(2)	0.94(2)	0.92(2)	0.49	0.87	0.86
wet blanket	0.21	0.77	0.79	0.84(1)	0.94(2)	0.94(2)	0.69	0.91	0.90

Table 3: Similarities results from P1 to P3 at NC level of the examples in Table 1. In P2, number in parenthesis corresponds to the position of the w_i with highest similarity score in the NC.

individually replaced by synonyms, and this would be reflected in lower similarity values for P3 than for P1. However, high similarity values are found across the idiomaticity spectrum, and for all models and all conditions the average similarities are higher than those for P1 (see Figures 1e and 1f). Contrary to what would be expected, the correlations with idiomaticity scores are mostly nonexistent, and when they do exist they are much lower than for P1, (see P1 and P3 in Table 2).

The overall picture painted by P3 points towards contextualised models not being able to detect when a change in meaning takes place by the substitution of individual components by their synonyms.

Qualitative analysis: For P3, Table 3 shows the similarities scores at NC level between each NC and their NC_{synW} counterpart. Again, similarity scores for GloVe are considerably lower than for BERT. As expected for GloVe, $\text{sim}^{(\text{P3})}_{\text{wet blanket}} = 0.69$ is noticeably higher than $\text{sim}^{(\text{P1})}_{\text{wet blanket}} = 0.21$, since individually the words *damp* and *cloak* are closer in meaning to *wet* and *blanket*, respectively, than *loser* is. Another evidence that contextualised models are not modelling idiomaticity well is, for NAT cases, the considerably higher $\text{sim}^{(\text{P3})}_{\text{wet blanket}} = 0.91$ in comparison to $\text{sim}^{(\text{P1})}_{\text{wet blanket}} = 0.77$, for BERT. Although for the other NCs, $\text{sim}^{(\text{P3})}_{\text{NC}}$ and $\text{sim}^{(\text{P1})}_{\text{NC}}$ are comparable, the special case of the more idiomatic *wet blanket* highlights the issues of idiomaticity representation.

4.4 Is there a difference between an NC in and out of context?

For contextualised models, the greater the influence of the context, the lower we would expect the similarity to be between an NC in and out of context. However, especially for BERT models the results (Figure 2) show a high similarity between the NC in and out of context ($\text{sim}^{(\text{P4})}_{\text{in-out}} > 0.8$). Moreover, a comparison with the similarities for the synonyms in P1 resulted in $\text{sim}^{(\text{P4})}_{\text{in-out-NEU}} > \text{sim}^{(\text{P1})}_{\text{NC-NEU}}$ and $\text{sim}^{(\text{P4})}_{\text{in-out-NAT}} \simeq \text{sim}^{(\text{P1})}_{\text{NC-NAT}}$, which indicates that these models consider the NC out of context to be a better approximation for the NC in context than its synonym. In addition, for BERT models $\text{sim}^{(\text{P4})}_{\text{in-out}}$ is only weakly correlated with the idiomaticity score (Table 4), which suggests that the context may not play a bigger role for idiomatic than it does for more compositional NCs.

Qualitative analysis: The $\text{sim}^{(\text{P4})}_{\text{in-out}}$ of the examples in Table 1 ranged from 0.78 (for *ghost town*) to 0.87 (*field work*) in the NAT condition, and from 0.84

	ELMo	BERT	DistilB	SBERT	BTRAM
EN_{NAT}	-	-	0.14	-0.16	0.14
EN_{NEU}	-	-	0.24	-0.24	-0.14
PT_{NAT}	0.25	0.17	0.18	-	0.21
PT_{NEU}	-	-	0.15	-	-

Table 4: Spearman ρ correlation with human judgments for P4, $p \leq 0.05$. Non-significant results are omitted.

(also for *ghost town*) to 0.90 (*eager beaver* and *wet blanket*) in the neutral sentences for BERT.[10] Together with these examples, the general results of P4 show large differences not explained by the semantic compositionality of the NCs, as suggested by the weak correlation with the idiomaticity scores. In this respect, both the largest and smallest differences between $\text{sim}^{(P4)}_{\text{in-out}}$ in NAT and NEU conditions appear in compositional NCs (*engine room* with $\text{sim}^{(P4)}_{\text{in-out-NAT}} = 0.68$, $\text{sim}^{(P4)}_{\text{in-out-NEU}} = 0.89$, and *rice paper* with $\text{sim}^{(P4)}_{\text{in-out-NAT}} = 0.84$, $\text{sim}^{(P4)}_{\text{in-out-NEU}} = 0.86$).

Besides, we expected ambiguous compounds such as *bad apple* or *bad hat* to have large $\text{sim}^{(P4)}_{\text{in-out}}$ differences between both conditions, as they occur with an idiomatic meaning in the NAT sentences. However, the differences were of just 0.06 in both cases, while other less ambiguous idiomatic NCs showed higher variations (e.g., *melting pot*, with 0.16). In sum, the results of P4 suggest that contextualised models do not properly represent some NCs.

4.5 But how informative are the contexts?

As the neutral sentences do not provide informative contextual clues, if the NCs in NAT and NEU conditions are similar, this would provide an additional indication that for these models contexts are not playing an important role in distinguishing usages (in this case between a neutral and uninformative usage and a naturalistic one). Indeed, the two conditions follow the same trends in the two languages, see Figure 1. Furthermore, there are significant correlations between NAT and NEU conditions, and some are very strong correlations. For example, for SBERT the correlations between the NC in context in naturalistic and neutral conditions are $\rho^{(P1,P2,P3)}_{\text{NC(Nat/Neu)}} > 0.85$ for English and > 0.76 for Portuguese, for probes P1, P2 and P3. This indicates that to evaluate the effect of the variants in each of these probes, a neutral sentence is as good as a naturalistic one. This reinforces the possibility that these models do not adequately incorporate the context in a way that captures idiomaticity.

In terms of the similarity between a sentence and its variants, as we assumed that the representation of a sentence corresponds to the average of the individual components, sentence length may have a strong impact on cosine similarity. This would explain the high values obtained for sentence similarities throughout the probes, as they could be more the effect of the number of words in a sentence than of their semantic similarity. Indeed, the correlation between naturalistic sentence length and the cosine similarities for the first three probes is moderate to strong for all models (Table 5), and higher for some of the contextualised models than for the baseline (e.g., DistilB in English and P2).

EN	GloVe	ELMo	BERT	DistilB	SBERT
P1	0.71	0.47	0.52	0.66	0.67
P2	0.87	0.79	0.78	0.89	0.84
P3	0.88	0.71	0.80	0.87	0.77
PT					
P1	0.60	0.46	0.61	0.68	0.62
P2	0.80	0.68	0.72	0.84	0.75
P3	0.69	0.58	0.64	0.76	0.75

Table 5: Spearman ρ correlation between naturalistic sentence length and cosine similarity, $p \leq 0.001$.

4.6 Other Operations

As referred in section 3.3 we have used vector averaging to obtain the NC embedding, as it is the standard procedure to represent not only MWEs but also out-of-vocabulary words, which are split into sub-tokens in contextualised models (Nandakumar et al., 2019; Wiedemann et al., 2019). However, we have also explored other methods to represent NCs in a single vector.

First, we have incorporated type-level vectors of the NCs into a BERT model, inspired by compositionality prediction methods (Baldwin et al., 2003; Cordeiro et al., 2019). To do so, we annotated the target NCs in large English and Portuguese corpora (Baroni et al., 2009; Wagner Filho et al., 2018) and used attentive mimicking with one-token-approximation (Schick and Schütze, 2019, 2020b) to learn up to 500 contexts for each NC. These new vectors encode each NC in a single representation, therefore avoiding possible biases produced by the compositional operations. Then, we used BERTRAM (Schick and Schütze, 2020a) to inject these type-level vectors in the BERT multilingual model. As expected, learning the vectors

---

[10] For Glove, $\text{sim}^{(P4)}_{\text{in-out}} = 1$.

of the NCs as single tokens improved the representation of idiomatic expressions (see BERTRAM in Tables 2 and 4), decreasing the correlation with idiomaticity in P1 (e.g., $\rho^{(P1)}_{\text{NC-NAT}} = 0.30$ in English), and increasing it in P2 ($\rho^{(P2)}_{\text{NC-NAT}} = 0.45$) and P3 ($\rho^{(P3)}_{\text{NC-NAT}} = 0.39 > \rho^{(P1)}_{\text{NC-NAT}}$). For P4, the correlation also increased in NAT contexts. In sum, these results were in general better and more statistically significant (at the expense of re-training a model).

Second, we compared the performance of averaging vs. concatenating the vectors of the NC subwords. In this case, we selected those utterances in English including NCs with the same number of sub-words of their synonyms (273 sentences), thus allowing for vector concatenation. Using this operation instead of average slightly improved the results of the BERT-based models (e.g., ≈0.06 higher correlations on average for P3 NAT) and obtained more significant values.

As the latter approach does not involve retraining a model, in further work we plan to probe other concatenation and pooling methods able to compare MWEs with different number of input vectors (e.g., *grey matter* vs. *brain*) which have achieved good results in sentence embeddings (Rücklé et al., 2018).

5 Conclusions

This paper presented probing tasks for assessing the ability of vector space models to retain the idiomatic meaning of NCs in the presence of lexical substitutions and different contexts. For these evaluations, we constructed the NCS dataset, with a total of 9,220 sentences in English and Portuguese, including variants with synonyms of the NC and of each of its components, in neutral and naturalistic sentences. The probing tasks revealed that contextualised models may not detect that idiomatic NCs have a lower degree of substitutability of the individual components when compared to more compositional NCs. This behaviour is similar in the controlled neutral and naturalistic conditions both in English and Portuguese.

The next steps are to extend the probing strategy with additional measures that go beyond similarities and correlations. Moreover, for ambiguous NCs, we intend to add probes for the different senses. Finally, we also plan to apply them to more languages, examining how multilingual information can be used to refine the representation of noun compounds and other MWEs.

Acknowledgments

Aline Villavicencio and Carolina Scarton are funded by the EPSRC project MIA: Modeling Idiomaticity in Human and Artificial Language Processing (EP/T02450X/1). Marcos Garcia is funded by the *Consellería de Cultura, Educación e Ordenación Universitaria* of the Galician Government (ERDF 2014-2020: Call ED431G 2019/04), and by a *Ramón y Cajal* grant (RYC2019-028473-I).

References

Laura Aina, Kristina Gulordava, and Gemma Boleda. 2019. Putting words in context: LSTM language models and lexical ambiguity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3342–3348, Florence, Italy. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. 2003. An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 89–96, Sapporo, Japan. Association for Computational Linguistics.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

Pedro Vitor Quinta de Castro, Nádia Félix Felipe da Silva, and Anderson da Silva Soares. 2018. Portuguese Named Entity Recognition Using LSTM-CRF. In *Proceedings of the 13th International Conference on the Computational Processing of the Portuguese Language (PROPOR 2018)*, pages 83–92, Canela–RS, Brazil. Springer, Cham.

Ting-Yun Chang and Yun-Nung Chen. 2019. What does this word mean? explaining contextualized embeddings with natural language definition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6064–6070, Hong Kong, China. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised compositionality prediction of nominal compounds. *Computational Linguistics*, 45(1):1–57.

David Alan Cruse. 1986. *Lexical semantics*. Cambridge university press.

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

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Meghdad Farahmand and James Henderson. 2016. Modeling the non-substitutability of multiword expressions with distributional semantics and a log-linear model. In *Proceedings of the 12th Workshop on Multiword Expressions*, pages 61–66, Berlin, Germany. Association for Computational Linguistics.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

Christina L Gagné and Thomas L Spalding. 2009. Constituent integration during the processing of compound words: Does it involve the use of relational structures? *Journal of Memory and Language*, 60(1):20–35.

Rachel Giora. 1999. On the priority of salient meanings: Studies of literal and figurative language. *Journal of pragmatics*, 31(7):919–929.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *arXiv preprint arXiv:1901.05287*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Nathan Hartmann, Erick Fonseca, Christopher Shulby, Marcos Treviso, Jéssica Silva, and Sandra Aluísio. 2017. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. In *Proceedings of the 11th Brazilian Symposium in Information and Human Language Technology*, pages 122–131, Uberlândia, Brazil. Sociedade Brasileira de Computação.

Hongbo Ji, Christina L Gagné, and Thomas L Spalding. 2011. Benefits and costs of lexical decomposition and semantic integration during the processing of transparent and opaque English compounds. *Journal of Memory and Language*, 65(4):406–430.

Milton King and Paul Cook. 2017. Supervised and unsupervised approaches to measuring usage similarity. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 47–52, Valencia, Spain. Association for Computational Linguistics.

Milton King and Paul Cook. 2018. Leveraging distributed representations and lexico-syntactic fixedness for token-level prediction of the idiomaticity of english verb-noun combinations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 345–350. Association for Computational Linguistics.

Jean-Paul Laurent, Guy Denhières, Christine Passerieux, Galina Iakimova, and Marie-Christine Hardy-Baylé. 2006. On understanding idiomatic language: The salience hypothesis assessed by ERPs. *Brain Research*, 1068(1):151–160.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Navnita Nandakumar, Timothy Baldwin, and Bahar Salehi. 2019. How Well Do Embedding Models Capture Non-compositionality? A View from Multiword Expressions. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 27–34, Minneapolis, USA. Association for Computational Linguistics.

James H. Neely, Dennis E. Keefe, and Kent L. Ross. 1989. Semantic priming in the lexical decision task: Roles of prospective prime-generated expectancies and retrospective semantic matching. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(6):1003–1019.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Grusha Prasad, Marten van Schijndel, and Tal Linzen. 2019. Using priming to uncover the organization of syntactic representations in neural language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8-13, 2011*, pages 210–218. The Association for Computer Linguistics.

Nils Reimers and Iryna Gurevych. 2019a. Alternative Weighting Schemes for ELMo Embeddings. *CoRR*, abs/1904.02954.

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

Joost Rommers, Ton Dijkstra, and Marcel Bastiaansen. 2013. Context-dependent semantic processing in the human brain: Evidence from idiom comprehension. *Journal of Cognitive Neuroscience*, 25(5):762–776.

Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. 2018. Concatenated power mean word embeddings as universal cross-lingual sentence representations. *arXiv preprint arXiv:1803.01400*.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2002)*, pages 1–15, Mexico City, Mexico. Springer, Berlin, Heidelberg.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing at NeurIPS 2019*, Vancouver, Canada.

Timo Schick and Hinrich Schütze. 2019. Attentive mimicking: Better word embeddings by attending to informative contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 489–494, Minneapolis, Minnesota. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020a. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3996–4007, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020b. Rare Words: A Major Problem for Contextualized Embeddings and How to Fix it by Attentive Mimicking. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8766–8774.

Marten van Schijndel and Tal Linzen. 2018. A neural model of adaptation in reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4704–4710, Brussels, Belgium. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Weijia Shi, Muhao Chen, Pei Zhou, and Kai-Wei Chang. 2019. Retrofitting contextualized word embeddings with paraphrases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1198–1203, Hong Kong, China. Association for Computational Linguistics.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. In *Proceedings of the Seventh International Conference on Learning Representations (ICLR 2019)*, New Orleans, Louisiana.

Jorge A. Wagner Filho, Rodrigo Wilkens, Marco Idiart, and Aline Villavicencio. 2018. The brWaC corpus: A new open resource for Brazilian Portuguese. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. 2019. Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 161–170, Erlangen, Germany. German Society for Computational Linguistics & Language Technology.

Appendices

A Naturalistic examples in English

Table 6 includes naturalistic examples in English. We include the compositionality scores provided by the annotators and the BERT and GloVe results at NC level.

B Naturalistic examples in Portuguese

Table 7 includes naturalistic examples in Portuguese. We include the compositionality scores provided by the annotators and the BERT and GloVe results at NC level.

Table 6: Naturalistic examples in English, including human compositionality (HC). Results for BERT model at NC level ($\mathbf{sim}^{(\mathbf{P})}_{\mathbf{NC}}$) together with GloVe results for the same measure. Average (Avg) values were calculated with BERT using the three NAT/NEU sentences for the same NC.

Probe	**Sentence (original compound in bold)**	**HC**	**Probe-specific results**		
			NC$_{syn}$	$\mathbf{sim}^{(\mathbf{P1})}_{\mathbf{NC}}$	**GloVe**
P1	He later became a **music journalist** covering the new wave and punk rock explosion [...]	4.54	music reporter	0.98	0.89
	The UN also held a **world conference** on Human Rights in Vienna.	3.96	global meeting	0.97	0.77
	The 3 month limit though is not a **brick wall**, if circumstances demand an extension of time [...]	3.79	obstacle	0.64	0.31
P2			**W**$_i$	$\mathbf{sim}^{(\mathbf{P2})}_{\mathbf{NC}}$	**GloVe**
	Allowing young people to opt out of the basic state pension is giving them a **poison pill**.	0.96	pill	0.92	0.85
	Arguably the king of comedy for the last ten years, Jim Carrey is **box office** gold.	0.88	box	0.81	0.81
P3			**NC**$_{synW}$	$\mathbf{sim}^{(\mathbf{P3})}_{\mathbf{NC}}$	**GloVe**
	It is not right that criminal enterprises try to use **dirty money** with a clean face.	2.21	smotty cash	0.93	0.63
	Formal evenings require a suit or dinner jacket for men and a **cocktail dress** for ladies.	3.04	appetizer costume	0.92	0.65
	If you burn coal without any kind of pollution control you get large amounts of ash and sulphur (and **radioactive waste** from natural Uranium decay in the coal).	4.58	dangerous rubbish	0.84	0.54
P4			$\mathbf{sim}^{(\mathbf{P4})}_{\mathbf{in\text{-}out}}$	**Avg** $\mathbf{sim}^{(\mathbf{P4})}_{\mathbf{in\text{-}out\text{-}NAT}}$	**Avg** $\mathbf{sim}^{(\mathbf{P4})}_{\mathbf{in\text{-}out\text{-}NEU}}$
	The roll-on/roll-off nuclear cargo ferry Atlantic Osprey suffered an **engine room** fire on Monday.	4.93	0.66	0.68	0.89
	And we had to explain to her the difference between **rice paper** and ordinary paper.	4	0.83	0.84	0.86
	However, it will not work unless every single person does it, because one **bad apple** ruins the whole barrel.	1.13	0.82	0.83	0.89
	The jury heard the evidence presented, that he was general **bad hat**.	0.62	0.76	0.76	0.83
	Yet its heyday was down with the epochal **melting pot** of punk/funk/art/jazz/dub [...]	0.54	0.73	0.73	0.89

Table 7: Naturalistic examples in Portuguese, including human compositionality (HC). Results for BERT model at NC level ($\mathbf{sim}^{(\mathbf{P})}_{\mathbf{NC}}$) together with GloVe results for the same measure. Average (Avg) values were calculated with BERT using the three NAT/NEU sentences for the same NC. English translations are in italic, together with the literal translation of the compounds where they are not word-to-word equivalents.

Probe	**Sentence (original compound in bold)**	**HC**	**Probe-specific results**		
			$\mathbf{NC}_{syn}$	$\mathbf{sim}^{(\mathbf{P1})}_{\mathbf{NC}}$	**GloVe**
P1	Normalmente, os restaurantes encontram-se dentro de **centros comerciais**. *Restaurants are usually located inside **shopping malls** (lit. comercial centres).*	3.68	shoppings	0.94	0.45
	Foi um dia pesaroso, um **sexto sentido** me alertava que uma coisa ruim puxa outra. *It was a sorrowful day, a **sixth sense** alerted me that one bad thing pulls another.*	1.4	intuição *intuition*	0.79	0.11
	Existe mesmo no serviço secreto inglês um **agente secreto** com licença para matar! *There really is in the English secret service a **secret agent** licensed to kill!*	4.58	espião *spy*	0.81	0.56
P2			$\mathbf{W}_i$	$\mathbf{sim}^{(\mathbf{P2})}_{\mathbf{NC}}$	**GloVe**
	Alguns dos estádios novos foram criticados por se tornarem “**elefantes brancos**” após a Copa. *Some of the new stadiums were criticized for becoming “**boondoggles**” (lit. white elephants) after the World Cup.*	0.16	elefantes *elephants*	0.96	0.81
	As espécies de **mar aberto** têm por princípio a natação contínua. *The **open sea** species have as a principle the continuous swimming.*	4.03	aberto *open*	0.9	0.79
P3			$\mathbf{NC}_{synW}$	$\mathbf{sim}^{(\mathbf{P3})}_{\mathbf{NC}}$	**GloVe**
	Foices e facões são **armas brancas** de uso corriqueiro. *Scythes and machetes are commonplace **white weapons**.*	0.65	pistolas alvas *untanned guns*	0.92	0.50
	Não deu quase ninguém, só alguns **gatos-pingados**! *There's hardly anybody, just a **few people** (lit. dripping cats)!*	0	felinos chuviscados *drizzled felines*	0.92	0.01
P4			$\mathbf{sim}^{(\mathbf{P4})}_{\textbf{in-out}}$	**Avg** $\mathbf{sim}^{(\mathbf{P4})}_{\textbf{in-out-NAT}}$	**Avg** $\mathbf{sim}^{(\mathbf{P4})}_{\textbf{in-out-NEU}}$
	Troque o **leite integral** pelo desnatado e economize nas calorias. *Replace **whole milk** (lit. integral milk) with skimmed milk and save on calories.*	4.67	0.86	0.84	0.79
	Ganhou até uma fama de “**pé-frio**”, por ter alguns rebaixamentos em seu currículo [...] *He even gained a reputation as an “**unlucky person**” (lit. cold foot), for having some downgrades in his resume [...]*	0.09	0.84	0.89	0.87
	Para muitos povos antigos, um novo mês era anunciado na passagem da **lua nova** para a lua crescente. *For many ancient peoples, a new month was announced as the **new moon** passed into the crescent moon.*	1.4	0.77	0.74	0.84
	Esse diagnostico é realizado através de **exame clínico** e radiográfico. *This diagnosis is made through **clinical** and radiographic **examination**.*	4.75	0.84	0.84	0.92

Is the Understanding of Explicit Discourse Relations Required in Machine Reading Comprehension?

Yulong Wu, Viktor Schlegel and Riza Batista-Navarro
Department of Computer Science, University of Manchester
Manchester, United Kingdom
`yulong-wu@outlook.com`
`{viktor.schlegel, riza.batista}@manchester.ac.uk`

Abstract

An in-depth analysis of the level of language understanding required by existing Machine Reading Comprehension (MRC) benchmarks can provide insight into the reading capabilities of machines. In this paper, we propose an ablation-based methodology to assess the extent to which MRC datasets evaluate the understanding of explicit discourse relations. We define seven MRC skills which require the understanding of different discourse relations. We then introduce ablation methods that verify whether these skills are required to succeed on a dataset. By observing the drop in performance of neural MRC models evaluated on the original and the modified dataset, we can measure to what degree the dataset requires these skills, in order to be understood correctly. Experiments on three large-scale datasets with the BERT-base and ALBERT-xxlarge model show that the relative changes for all skills are small (less than 6%). These results imply that most of the answered questions in the examined datasets do not require understanding the discourse structure of the text. To specifically probe for natural language understanding, there is a need to design more challenging benchmarks that can correctly evaluate the intended skills[1].

1 Introduction

Machine Reading Comprehension (MRC) is concerned with the automatic extraction and generation of answers over unstructured textual data. Due to its complexity, the task is seen as suitable for evaluating Natural Language Understanding (NLU) (Chen, 2018). While neural MRC systems achieve impressive performance (Devlin et al., 2019; Lan et al., 2020), it has been revealed by some research efforts that existing MRC benchmarks might be insufficient to establish model performance, i.e., that the models are not being assessed for their capabilities to read and comprehend (Jia and Liang, 2017; Mudrakarta et al., 2018; Min et al., 2018; Sugawara et al., 2018; Feng et al., 2018; Jiang and Bansal, 2019; Min et al., 2019; Chen and Durrett, 2019; Schlegel et al., 2020; Sugawara et al., 2020). These analyses provide insights into the weaknesses of modern MRC gold standards. Nonetheless, to stimulate the development of robust MRC systems with generalisable NLU capabilities, it is necessary to investigate the strengths and weaknesses of MRC datasets on a deeper level.

In the task of MRC, it is assumed that questions test a cognitive process which involves various skills, such as retrieving stored information and performing inferences (Sutcliffe et al., 2013). Therefore, considering metrics that reflect skills required to answer questions is useful for analysing the capabilities of MRC datasets to benchmark NLU (Sugawara et al., 2020). This leads to the following intuition: if a question is solvable even after removing features (e.g., specific words) associated with an MRC skill, the question does not require the skill. Sugawara et al. (2020) examined 10 datasets with regard to multiple requisite skills for answering questions. One of the identified 12 skills is the *understanding of adjacent discourse relations*, which relies on information given by the sentence order in a passage. By randomly shuffling the order of the sentences in the context and comparing model performance on the original and the modified dataset, they concluded that most existing MRC datasets might be inadequate for benchmarking adjacent discourse relations understanding.

Discourse relations describe how two segments of discourse are logically connected to one another. Understanding them is key to answering reading comprehension questions correctly. Though the findings in Sugawara et al. (2020) are useful to understand MRC datasets with respect to discourse

---

[1] Our code is available at `https://github.com/Yulong-W/mrcdr`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3565–3579
April 19 - 23, 2021.

relations understanding, we argue that it is not enough to only consider inter-sentential relations as discourse relations also widely exist within sentences[2]. Furthermore, there also exist various types of relations and senses. Hence, to comprehensively assess the capacity of MRC datasets to benchmark discourse relations understanding, we assert that further research is needed.

In this paper, our aim is to provide a fine-grained analysis of the level of discourse relations understanding that is needed to answer questions in existing MRC datasets. Specifically, we focus on explicit discourse relations, which are expressed using explicit connectives. This allows us to perform analysis that goes beyond shuffling sentence order. In our work, we identify seven MRC skills that represent different aspects of understanding explicit discourse relations. With these, we examine three datasets using two strong MRC models. Our results show that these datasets might be insufficient for evaluating the understanding of explicit discourse relations. This work can potentially encourage the development of more challenging benchmarks that evaluate MRC models with respect to NLU capabilities that require discourse relations understanding.

2 Requisite Skills

As mentioned above, we identified a set of seven reasoning-related skills that require the understanding of explicit discourse relations, as shown in Table 1.

Skill s_1 is inspired by Sugawara et al. (2020), which aims to evaluate whether the understanding of adjacent explicit discourse relations is required in answering questions. Different from their proposed method (i.e., randomly shuffling the order of the sentences in a passage), we only shuffle those containing explicit connectives.

The selection of skills s_2 to s_7 is informed by the annotation scheme of the PDTB 3.0 corpus, which is annotated with information on discourse relations (Webber et al., 2019). The scheme defines 36 different senses of discourse relations. In the corpus, more than 24, 000 explicit connectives were annotated and categorised according to these senses. Based on this, we obtained a distribution of explicit connectives over the 36 senses (see Appendix A). Afterwards, we selected a subset of them (6 senses) based on the number of unique explicit connectives, total number of explicit connectives for which each sense was recorded, and the exclusiveness of these explicit connectives. The identification process is detailed in Appendix B. In the following, we provide an overview of skills s_2 to s_7.

Skills s_2 and s_3 are for the understanding of asynchronous temporal relations. Specifically, s_2 focuses on precedence while s_3 tests succession. Skill s_4 evaluates the understanding of causal relations, which are explicitly marked in the passage by connectives such as *because* and *due to*. Meanwhile, our motivation for selecting skill s_5 is to reveal whether explicit conditional reasoning is required to answer questions. Different from s_4, s_6 is for the understanding of negative causality, in which a causal relation expected on the basis of the first argument is negated by the situation described in the other. Finally, s_7 assesses expansions which provide further detail to an argument.

3 Methodology

For each of the seven identified skills, we defined an ablation method, as shown in Table 1. The

Skill	**NLU Tested**	**Ablation Method**
	shuffling method	
s_1	Explicit discourse relations between adjacent sentences	Shuffle the order of the sentences that contain explicit connectives in the context
	masking methods (Drop all occurrences of corresponding explicit connectives)	
s_2	Temporal reasoning (precedence)	Drop e.g. *afterward, later, …*
s_3	Temporal reasoning (succession)	Drop e.g. *earlier, since before, …*
s_4	Explicit causality reasoning	Drop e.g. *because of, due to …*
s_5	Explicit conditional reasoning	Drop e.g. *only if, depending on, …*
s_6	Negative causality reasoning	Drop e.g. *albeit, but then again, …*
s_7	Expansion of explicit discourse relations	Drop e.g. *additionally, moreover, …*

Table 1: Requisite skills and ablation methods.

---

[2] In the Penn Discourse Treebank (PDTB) 3.0 corpus (Webber et al., 2019), 24,369 and 29,818 tokens were annotated as connectives for intra-sentential and inter-sentential discourse relations, respectively.

design of these methods is based on the fact that explicit discourse relations are expressed using explicit discourse connectives (Webber et al., 2019). The scope of the proposed methodology hence captures only relations represented by explicit connectives, rather than all discourse relations-related features of the datasets. We assume that through shuffling the order of the sentences with connectives in the context, as well as through dropping these connectives, the corresponding relations will be broken. After applying the ablation method on the development set of an MRC dataset, if the performance of the model did not change significantly, we can say that most of the questions in the dataset are solvable even without the given skill; hence, the dataset does not sufficiently evaluate models with respect to the said skill. On the contrary, if the performance gap between the original and the modified dataset is large, we might infer that a substantial proportion of the questions require that skill. Nonetheless, should the model perform badly on the ablated dataset, we cannot take this as evidence that the model in fact acquired the investigated reasoning capabilities as the bad performance can stem from many different factors (e.g., distribution shift induced by dropping numerous words).

4 Experiments

In this section, we describe our experimental settings, present the results of our experiments and provide insights drawn from experimentation under an extreme setting whereby all explicit connectives were dropped.

4.1 Experimental Settings

Datasets. We examined three datasets with two answering styles. For span prediction datasets in which the goal is to identify a span in the passage as the answer, we used SQuAD 1.1 (Rajpurkar et al., 2016) and SQuAD 2.0 (Rajpurkar et al., 2018). For multiple choice datasets in which the correct answer is chosen from a candidate set of answers, we used SWAG (Zellers et al., 2018). We applied the ablation methods on the development set of each dataset. Sentence segmentation and tokenisation are performed as part of the pre-processing step.

Models. In the main experiment, we used the BERT-base (uncased) model (Devlin et al., 2019). Our goal is to analyse whether there exists at least one model architecture that can solve the MRC task without the understanding of explicit discourse relations; hence, it is enough to use a single model (Sugawara et al., 2020). Then, from the perspective of testing the effectiveness of the proposed MRC skills, we employed a stronger model, ALBERT-xxlarge (Lan et al., 2020). We fine-tuned the pre-trained BERT-base (uncased) and ALBERT-xxlarge model on the training set of each dataset and evaluated them on the original and the modified development sets by making use of the HuggingFace's *Transformers* library (Wolf et al., 2020). The hyperparameters of the models are reported in Appendix C.

Ablation methods. Method m_1: For the choice of explicit connectives, we used the 173 explicit connectives from the PDTB 3.0 corpus (Webber et al., 2019) (see Appendix D). We averaged the scores over five runs and report the mean and variance values in Appendix E. Methods m_2 to m_7: we list explicit connectives dropped for each sense in Appendix F. When a token is dropped, it is replaced with an [UNK] token to preserve the correct answer span. More in-depth results are reported in Appendix G.

4.2 Results and Discussion

In this section, we report the results for the skills in Table 2. In this table, for each of the ablation method used for skills s_2 to s_7, there are two versions of experimental results, shown in the white and shaded areas, respectively. Results written in the white areas were obtained by applying the ablation methods detailed in Section 4.1, i.e., by masking explicit connectives selected using the threshold-based method (see Appendix B) for which each sense was annotated. However, it can be seen in the table that except for s_7, the relative differences for s_2 to s_6 were extremely small (less than 1%) across all datasets. To further investigate whether these skills are truly not required to answer questions in the three datasets, we performed additional experiments as follows.

For the senses that represent a skill under evaluation, we dropped every explicit connective associated with those senses according to the PTDB 3.0 annotations (Webber et al., 2019). By applying these modified ablation methods, we obtained additional experimental results, shown in the shaded areas of Table 2. In the following, we discuss the observations for all the defined skills.

s_1: adjacent explicit discourse relations understanding. On all datasets, the relative changes

Skill	**SQuAD 1.1** *(F1)*	**SQuAD 2.0** *(F1)*	**SWAG** *(Acc.)*
Orig.	88.6	76.1	79.0
s_1	$86.2_{-2.7}$	$74.8_{-1.7}$	-
s_2	$88.3_{-0.3}$	$76.0_{-0.1}$	$79.0_{-0.0}$
	$88.0_{-0.7}$	$75.8_{-0.4}$	$79.0_{-0.0}$
s_3	$88.5_{-0.1}$	$76.1_{-0.0}$	$79.0_{-0.0}$
	$87.4_{-1.4}$	$75.4_{-0.9}$	$79.0_{-0.0}$
s_4	$87.9_{-0.8}$	$75.6_{-0.7}$	$79.0_{-0.0}$
	$78.6_{-11.3}$	$71.0_{-6.7}$	$78.6_{-0.5}$
s_5	$88.4_{-0.2}$	$76.1_{-0.0}$	$79.0_{-0.0}$
	$83.4_{-5.9}$	$73.8_{-3.0}$	$78.6_{-0.5}$
s_6	$88.4_{-0.2}$	$76.1_{-0.0}$	$79.0_{-0.0}$
	$87.5_{-1.2}$	$75.6_{-0.7}$	$78.6_{-0.5}$
s_7	$84.9_{-4.2}$	$74.1_{-2.6}$	$79.1_{+0.1}$
	$83.7_{-5.5}$	$73.7_{-3.2}$	$78.6_{-0.5}$

Table 2: The performance (%) of the BERT-base model with the ablation tests on the development set. Values in smaller font are changes (%) relative to the original performance of the model. For mask-related methods (m_2 to m_7), the results shown in the white areas are obtained from the initial test while the results shown in the shaded areas represent the further test, i.e., dropping all explicit connectives for which each identified sense was annotated. Acc.: accuracy as a percentage.

for s_1 were small. We do not apply m_1 to SWAG because its contexts are only one sentence long. On SQuAD 1.1 and SQuAD 2.0, the difference was hardly noticeable (less than 3% and 2%, respectively). These results indicate that most of the questions already solved in these datasets do not necessarily require the understanding of adjacent explicit discourse relations and are solvable even if the sentences appear unnaturally. This confirms the findings of Min et al. (2018), which reported that 92% of questions in SQuAD 1.1 are solvable by only looking at the sentence containing the answer.

s_2 and s_3: performing asynchronous temporal reasoning. We found that for the three examined datasets, the relative changes for s_2 and s_3 were extremely small (the biggest drop was even less than 1.5%), regardless of whether only a part or all associated explicit connectives were dropped. This indicates that these datasets might not adequately benchmark the understanding of asynchronous temporal relations.

s_4: explicit causality reasoning. In the initial experiment, the relative changes for s_4 on the three datasets were extremely small (less than 1%). However, surprisingly, after masking all explicit connectives cueing causality, except for SWAG which still featured a low drop (0.5%), the relative drops on SQuAD 1.1 and SQuAD 2.0 increased noticeably (from less than 1% to 11.3% and 6.6%, respectively). Particularly, for SQuAD 1.1, the decrease was the largest in all our experiments. Nevertheless, we cannot simply conclude that s_4 is needed to answer questions in the two datasets as the additionally dropped explicit connectives were also recorded as many other senses in the PDTB 3.0 corpus and not associated with this sense for the majority of occurrences. As we do not know exactly whether the decrease in model performance is due to this sense or any other senses, further analyses are necessary. Based on the PDTB 3.0 Annotation Manual (Webber et al., 2019), we calculated the percentage of each additionally dropped connective for this sense among the multiple senses for which it was annotated, and removed those which are rarely used for this sense from the candidate set of the dropped explicit connectives. The experiments demonstrated that the model achieved 85.1 and 74.3 (4.0% and 2.3% relative drop) F1 score on SQuAD 1.1 and SQuAD 2.0, respectively. This implies that the examined datasets might not correctly benchmark the understanding of causal relations and the reason why the relative drops were large after dropping all explicit connectives is that the other senses might be important.

s_5: explicit conditional reasoning. In the initial test, on all datasets, the relative changes were extremely small (less than 0.3%). Nonetheless, after dropping all explicit connectives describing conditional relations, except for SWAG which still showed a low drop (0.5%), the performance on SQuAD 1.1 and SQuAD 2.0 decreased by more than 3%. However, similarly to s_4, we cannot conclude whether such a decrease is due to sense representing s_5 or other senses that the explicit connectives are also associated with. As a result, we removed explicit connectives which are rarely used for this sense from the candidate set of explicit connectives and conducted further analyses. The experiments demonstrated that the model achieved 88.4 and 76.1 F1 on SQuAD 1.1 and SQuAD 2.0, respectively, both less than 0.5% relative difference. This indicates that s_5 might not necessarily

Dataset	**SQuAD 1.1** *(F1)*	**SQuAD 2.0** *(F1)*	**SWAG** *(Acc.)*
Original	94.4	87.6	89.0
Ablated	$92.4_{-2.1}$	$84.8_{-3.2}$	$86.3_{-3.0}$

Table 3: Performance of ALBERT-xxlarge on the original development set and on a version with all explicit connectives dropped. Acc.: accuracy as a percentage.

be required to answer questions in these datasets either.

s_6**: reasoning about negative causality.** On all datasets, the relative drops for s_6 were extremely small (less than 1.3%), whether with part of or all explicit connectives dropped. This demonstrates that most of the solved questions in the three MRC datasets do not necessarily require negative causal reasoning.

s_7**: recognising the expansion of explicit discourse relations.** In the initial experiment, the relative changes for s_7 on SWAG and SQuAD 2.0 were small, while that on SQuAD 1.1 was slightly larger (more than 4%). After dropping all explicit connectives for which sense *Expansion.Conjunction* was annotated, the performance of the model further decreased moderately – up to 5.5% for SQuAD 1.1, implying that compared to the other two datasets, SQuAD 1.1 might have more potential for benchmarking the understanding of the expansion of explicit discourse relations.

4.3 Further Analyses

Surprised by the moderate performance changes, we investigated the extent to which understanding of any explicit discourse relations is required by the datasets. Therefore, we dropped all explicit connectives and employed a stronger model, ALBERT-xxlarge (Lan et al., 2020) to generalise our assumption from the six specific senses to all senses. To mitigate the effect of distribution shift between training and evaluation data introduced by removing large parts of the context, we applied the ablation methods on the training set as well. The results are shown in Table 3. The performance drops no more than 3.2% for all three datasets, contributing further evidence towards the hypothesis that understanding the discourse structure of the text is hardly required to perform well on the investigated benchmarks.

5 Conclusion

In this paper, we proposed a methodology to assess the capabilities of MRC datasets to benchmark the understanding of explicit discourse relations. With seven fine-grained skills and corresponding ablation methods, we examined three large-scale datasets. The experimental results demonstrated that explicit discourse relations are not sufficiently evaluated by them, and thus there is a need to develop more challenging datasets so that their questions can correctly benchmark our defined skills. As for future work, we will develop a machine learning-based system that can recognise various senses of implicit discourse relations in the passage and further reveal whether the awareness of implicit discourse relations is required to do well on contemporary MRC benchmarks.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable suggestions. We also acknowledge the use of the Computational Shared Facility at The University of Manchester.

References

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems.

In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2726–2736, Florence, Italy. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1725–1735, Melbourne, Australia. Association for Computational Linguistics.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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

Viktor Schlegel, Marco Valentino, Andre Freitas, Goran Nenadic, and Riza Batista-Navarro. 2020. A framework for evaluation of machine reading comprehension gold standards. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5359–5369, Marseille, France. European Language Resources Association.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Brussels, Belgium. Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8918–8927. AAAI Press.

Richard FE Sutcliffe, Anselmo Penas, Eduard H Hovy, Pamela Forner, Alvaro Rodrigo, Corina Forascu, Yassine Benajiba, and Petya Osenova. 2013. Overview of QA4MRE Main Task at CLEF 2013. In *CLEF (Working Notes)*.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The Penn Discourse Treebank 3.0 Annotation Manual.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

A Senses and Their Associated Explicit Connectives

This Appendix provides a distribution of the 36 distinct senses annotated for explicit connectives (Table 4), which are calculated by referring to Appendix A of the PDTB 3.0 annotation scheme (Webber et al., 2019). For each sense, the second column lists the explicit connectives for which the sense was annotated, with counts given for each connective (in parentheses). The third column lists the total number of explicit connectives for which each sense was annotated. Discontinuous connectives are indicated with a "+" symbol between their parts.

Sense	Explicit Connectives	Total
Temporal.Synchronous	as (383), as long as (4), as soon as (9), at the same time (65), by (1), in (18), in the meantime (11), in the meanwhile (1), meantime (2), meanwhile (120), now that (2), simultaneously (6), still (1), then (4), upon (2), when (509), while (163), with (5)	1306
Temporal.Asynchronous.Precedence	afterward (6), afterwards (5), before (309), finally (13), in the end (3), later (92), later on (2), next (4), since (10), still (2), subsequently (3), then (310), thereafter (11), till (4), ultimately (15), until (143), when (4)	936
Temporal.Asynchronous.Succession	after (533), as (3), as soon as (11), before (2), by then (6), earlier (15), in the meantime (2), once (70), previously (53), since (83), since before (1), until (7), when (160)	946
Contingency.Cause.Reason	about (2), and (4), as (180), because (833), because of (12), by (10), due to (1), for (34), from (2), given (6), in (1), indeed (1), insofar as (1), not only because of (1), now that (10), on (1), since (96), ultimately (1), when (21), with (109), without (1)	1327
Contingency.Cause.Result	accordingly (5), and (5), as a result (78), consequently (10), for (1), hence (5), in the end (2), so (222), so that (10), then (7), thereby (9), therefore (26), thus (111), without (1)	492
Contingency.Cause.NegResult	—	0
Contingency.Cause+Belief.Reason+Belief	as (3), because (2), from (2), given (3), in (1), indeed (4), with (5)	20
Contingency.Cause+Belief.Result+Belief	so (1), thus (1)	2
Contingency.Cause+SpeechAct.Reason+SpeechAct	but (1)	1
Contingency.Cause+SpeechAct.Result+SpeechAct	and (1)	1
Contingency.Condition.Arg1-as-cond	and (22), then (1)	23
Contingency.Condition.Arg2-as-cond	as long as (13), by (2), depending on (3), depending upon (1), for (7), if (1084), if and when (2), if only (4), if+then (37), in (8), in case (6), in order (4), once (4), only if (13), so long as (4), until (17), when (116), whenever (9), where (2), with (2)	1338
Contingency.Condition+SpeechAct	because (2), if (56), if+then (1), or (2), when (12)	73

Continued on next page

Table 4 – Continued from previous page

Sense	**Explicit Connectives**	**Total**
Contingency.Negative-condition.Arg1-as-negCond	either+or (2), else (1), lest (2), or (7), otherwise (4)	16
Contingency.Negative-condition.Arg2-as-negCond	till (1), unless (98), without (9)	108
Contingency.Negative-condition+SpeechAct	—	0
Contingency.Purpose.Arg1-as-goal	—	0
Contingency.Purpose.Arg2-as-goal	and (128), for (16), if only (1), in (2), in order (51), so (44), so as (3), so that (21)	266
Comparison.Concession.Arg1-as-denier	although (206), as (7), as much as (2), by (1), despite (9), even as (2), even if (87), even though (69), even when (8), even with (2), for (1), however (5), if (6), no matter (8), regardless of (6), though (91), whatever (4), when (3), whether (7), while (203), with (2)	729
Comparison.Concession.Arg2-as-denier	albeit (1), although (105), as if (4), but (3063), but then (3), but then again (1), even so (9), even though (26), however (390), if (3), if only (1), in any case (3), in fact (4), in the end (1), indeed (1), meanwhile (2), nevertheless (32), nonetheless (25), nor (1), not only+but (1), on the one hand+on the other hand (1), on the other hand (4), only (2), or (1), regardless (2), still (115), though (128), when (1), while (2), without (19), yet (96)	4047
Comparison.Concession+SpeechAct.Arg2-as-denier+SpeechAct	and (2), but (2), if (1), or (11)	16
Comparison.Contrast	although (14), and (16), as (5), but (618), by comparison (11), by contrast (28), conversely (2), however (95), if (2), in contrast (12), in fact (7), in the end (1), like (1), meanwhile (7), neither+nor (1), nevertheless (12), nonetheless (2), not only+but also (1), on the contrary (4), on the one hand+on the other (1), on the one hand+on the other hand (1), on the other hand (32), only (1), still (75), though (16), when (1), whereas (5), while (140), with (1), yet (4)	1116
Comparison.Similarity	as (65), as though (1), as well (6), like (3), similarly (18), while (1)	94

Continued on next page

Table 4 – Continued from previous page

Sense	Explicit Connectives	Total
Expansion.Conjunction	additionally (7), along with (2), also (1736), and (6189), as much as (1), as well (12), as well as (7), besides (19), beyond (1), both+and (6), but (42), but also (1), finally (18), further (7), furthermore (12), in addition (165), in fact (36), in the end (1), indeed (54), likewise (8), meanwhile (27), moreover (103), much less (3), neither+nor (2), nor (31), not just+but (1), not just+but+also (1), not only (5), not only+also (1), not only+but (18), not only+but also (9), or (71), plus (1), separately (72), then (11), ultimately (1), while (43), with (41), yet (2)	8767
Expansion.Disjunction	alternatively (4), and then (1), as an alternative (2), either+or (36), nor (1), or (258), or otherwise (2)	304
Expansion.Equivalence	in other words (17), indeed (2), or (6), that is (2)	27
Expansion.Exception.Arg1-as-excpt	otherwise (15)	15
Expansion.Exception.Arg2-as-excpt	although (2), but (3), except (12), only (3)	20
Expansion.Instantiation.Arg1-as-instance	as if (1), in (1)	2
Expansion.Instantiation.Arg2-as-instance	as (4), for example (200), for instance (98), in fact (3), in particular (6), indeed (2), like (1), such as (2), with (7)	323
Expansion.Level-of-detail.Arg1-as-detail	as (8), in (17), in fact (1), in short (4), in sum (2), in the end (2), indeed (1)	35
Expansion.Level-of-detail.Arg2-as-detail	and (4), as though (2), by (2), for (1), in (2), in fact (34), in particular (9), in that (1), in the end (1), indeed (37), insofar as (1), only (1), specifically (10), that is (2), with (111), without (2)	220
Expansion.Manner.Arg1-as-manner	thereby (3)	3
Expansion.Manner.Arg2-as-manner	and (19), as (3), as if (1), by (174), in (13), when (2), with (6), without (62)	280
Expansion.Substitution.Arg1-as-subst	from (1), instead of (43), rather than (40)	84
Expansion.Substitution.Arg2-as-subst	alternatively (2), as much as (1), instead (112), more accurately (1), not so much as (1), rather (17), so much as (1)	135

Table 4: Senses and their associated explicit connectives annotated in the PDTB 3.0 corpus (Webber et al., 2019).

B Identification of the Senses of Explicit Discourse Relations

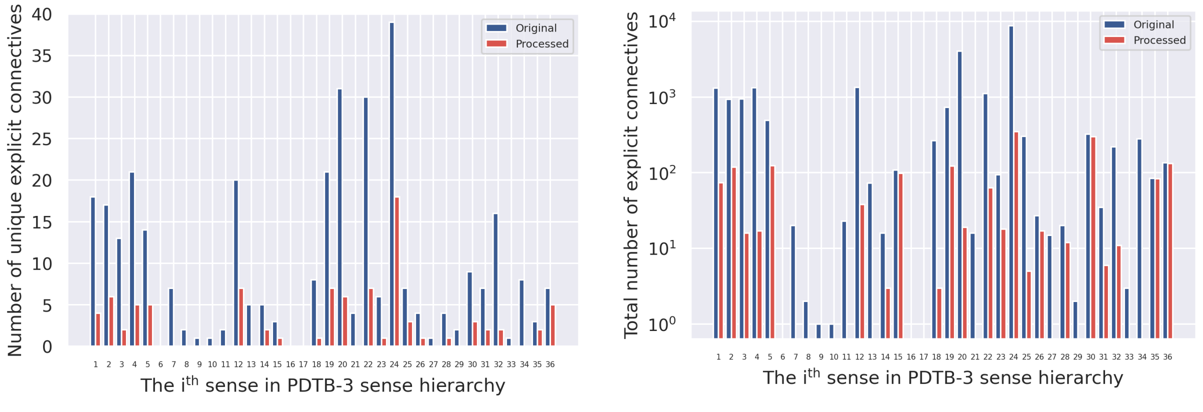
Table 4 provides the distribution of the 36 senses and their associated explicit connectives. To determine which kind of senses to focus on, we first defined two metrics: *"uniqueness"* and *"instances"* to measure the 36 senses. The first metric "uniqueness" measures the number of unique explicit connectives in each sense. For instance, as can be seen from Table 4, the sense "Contingency.Condition.Arg1-as-cond" was annotated for two unique explicit connectives: "and (22), then (1)". Therefore, its uniqueness is equal to 2. The second metric "instances" measures the total number of connectives for which each sense was annotated. Take the same example, we can see that the connective "and" was annotated 22 times as the sense "Contingency.Condition.Arg1-as-cond" and the connective "then" was annotated once as the same sense. Under this circumstance, there is a total of 23 (22+1) explicit connectives for which the sense "Contingency.Condition.Arg1-as-cond" was annotated, and thus its instances is equal to 23.

We propose that the two metrics can reflect the breadth and importance of these senses in a passage of text as Table 4 was developed from the large-scale PDTB 3.0 corpus (Webber et al., 2019), which provides a certain degree of representation. In this context, the higher "uniqueness" and "instances" a sense features, the more widely it might spreads in the context. Consequently, choosing such a sense to focus on is more likely to reveal whether the existing MRC benchmarks test the model's understanding of it.

Besides the two defined metrics, we also noticed that in the PDTB 3.0 corpus (Webber et al., 2019), many different senses were recorded for the same connective. For example, the connective "in the end" was annotated as seven types of senses. In this case, we cannot exactly examine which kind of senses the MRC datasets assessed by dropping their associated non-exclusive explicit connectives. This indicates that there is a need to consider the issue of managing explicit connectives for which multiple senses were annotated. To this end, we introduced the third metric: *"exclusiveness"*, which measures the degree of semantic overlap of explicit connectives in each sense. Ideally, to ensure that there are no overlapping explicit connectives among these senses, we can just remove all of the explicit connectives for which multiple senses were annotated and keep those that represent only one type of sense. However, after doing this, the "uniqueness" and "instances" of most senses are greatly decreased (see Figure 1a and Figure 1b). Based on this, we posit that the cost, i.e., most senses losing a considerable number of explicit connectives, is too high when attempting to retain their exclusiveness. Though the senses with only exclusive explicit connectives could meet the three metrics, they might not be enough for our data ablation purposes, as most of the explicit connectives were eliminated. Considering this, we need to find a balance between preserving the number and types of explicit connectives in each sense and maintaining its exclusiveness.

To minimise the loss in terms of "uniqueness" and "instances" of each sense while preserving "exclusiveness", we propose that if a connective C was annotated with multiple senses and it is used for sense X majority of the time, then we could include it in sense X. To identify the exact value of the "majority", we calculated the percentage of the distinct senses annotated for each non-exclusive connective and selected the sense with the highest percentage. Subsequently, we averaged these highest values and obtained the threshold, which is about 69%. Finally, we chose explicit connectives where the highest proportion of the sense for which they were annotated exceeds 69% and eliminated those below the threshold. From Figure 2a and Figure 2b, one can see that both the "uniqueness" and "instances" of the most senses with some retained non-exclusive explicit connectives increased, compared with those that only contain the exclusive connectives. This demonstrates that our method has effectively increased the number and types of explicit connectives in the most senses while maintaining their exclusiveness.

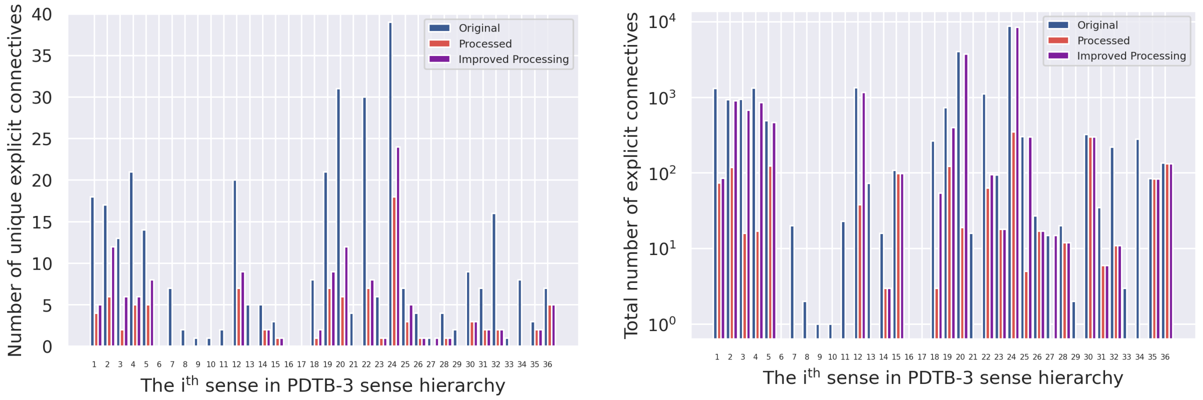
Finally, to select the candidate senses from the 36 senses, we visualised them in terms of their "uniqueness" and "instances", as shown in Figure 3a and Figure 3b, respectively. As can be seen in Figure 3a, there are a total of 12 senses with the number of unique explicit connectives above the mean value (sense 24, 2, 20, 12, 19, 5, 22, 3, 4, 1, 25, 36). Furthermore, it can be seen from Figure 3b that there are a total of 6 senses with the total number of explicit connectives larger than the average (sense 24, 20, 12, 2, 4, 3). Then, we took the intersection of these two sets of senses and obtained a sense set whereby "uniqueness" and "instances" of each sense is above the mean, and its



(a)

(b)

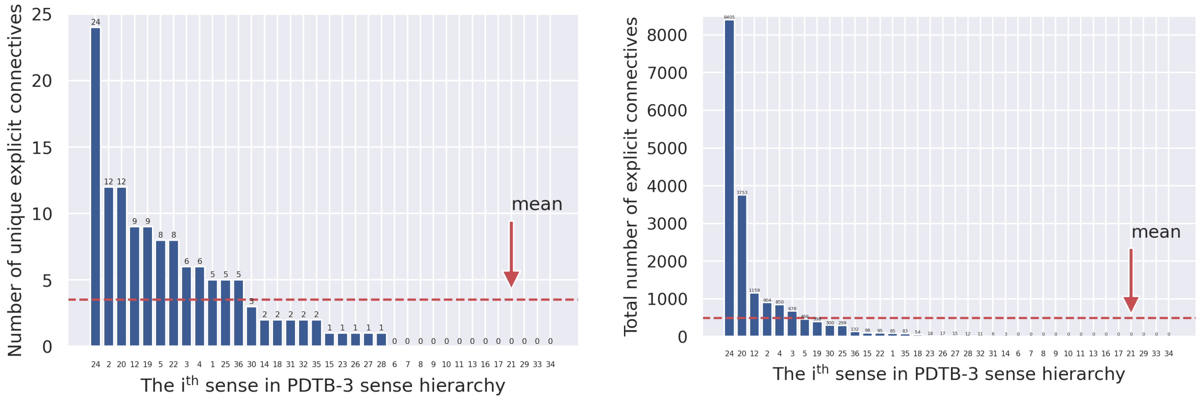
Figure 1: Visualisation of the number of unique explicit connectives and total number of explicit connectives for each sense in the PDTB 3.0 sense hierarchy (Webber et al., 2019). The blue bar represents the original senses, while the red one represents the senses after removing the non-exclusive explicit connectives.



(a)

(b)

Figure 2: Visualisation of the number of unique explicit connectives and total number of explicit connectives for each sense in the PDTB 3.0 sense hierarchy (Webber et al., 2019). The purple bar represents the senses with some retained non-exclusive connectives.



(a)

(b)

Figure 3: Number of unique explicit connectives and total number of explicit connectives for which each sense (processed) was annotated (sorted from largest to smallest).

"exclusiveness" is retained to a certain extent:[3]

- Sense 2: Temporal.Asynchronous.Precedence
- Sense 3: Temporal.Asynchronous.Succession
- Sense 4: Contingency.Cause.Reason
- Sense 12: Contingency.Condition.Arg2-as-cond
- Sense 20: Comparison.Concession.Arg2-as-denier
- Sense 24: Expansion.Conjunction

C Hyperparameters of the BERT-base and ALBERT-xxlarge Model

Hyperparameters used in the BERT-base and ALBERT-xxlarge model are shown in Table 5.

Dataset	d	b	lr	ep
BERT-base				
SQuAD 1.1	384	12	3e-5	2.0
SQuAD 2.0	384	12	3e-5	2.0
SWAG	80	8	5e-5	3.0
ALBERT-xxlarge				
SQuAD 1.1	384	12	3e-5	2.0
SQuAD 2.0	384	12	3e-5	4.0
SWAG	80	128	5e-5	3.0

Table 5: The hyperparameters used to fine-tune the BERT-base and ALBERT-xxlarge model on each dataset. *d* is the size of the token sequence fed into the model, *b* is the training batch size, *lr* is the learning rate, and *ep* is the number of training epochs. We used stride = 128 for documents longer than *d* tokens.

D A Set of Explicit Connectives

We list the set of explicit connectives used in this work in Figure 4.

E Performance Means and Variances in Shuffle-Based Method

We report the means and variances for the shuffling ablation method for skill s_1 in Table 6.

F The Six Identified Senses and Their Associated Explicit Connectives

Table 7 shows the six identified senses and their associated explicit connectives. For each sense, the associated explicit connectives were selected using the threshold-based method detailed in Appendix B.

G Detailed Results of SQuAD 2.0

We report the ablation results for has-answer and no-answer questions in SQuAD 2.0 in Table 8.

---

[3] A detailed introduction of these senses is available at https://catalog.ldc.upenn.edu/docs/LDC2019T05/PDTB3-Annotation-Manual.pdf.

Continuous explicit connectives (One token):
about, accordingly, additionally, after, afterward, afterwards, albeit, also, alternatively, although, and, and/or, as, because, before, besides, beyond, but, by, consequently, conversely, despite, earlier, else, except, finally, for, from, further, furthermore, given, hence, however, if, in, indeed, instead, later, lest, like, likewise, meantime, meanwhile, moreover, nevertheless, next, nonetheless, nor, on, once, only, or, otherwise, plus, previously, rather, regardless, separately, similarly, simultaneously, since, so, specifically, still, subsequently, then, thereafter, thereby, therefore, though, thus, till, ultimately, unless, until, upon, whatever, when, whenever, where, whereas, whether, while, with, without, yet

Continuous explicit connectives (Two tokens):
along with, and then, as if, as though, as well, because of, but also, but then, by comparison, by contrast, by then, depending on, depending upon, due to, even after, even as, even before, even if, even so, even then, even though, even when, even while, even with, for example, for instance, if only, in addition, in case, in contrast, in fact, in order, in particular, in short, in sum, in that, insofar as, instead of, later on, more accurately, much less, no matter, not only, now that, only if, or otherwise, rather than, regardless of, since before, so as, so that, such as, that is

Continuous explicit connectives (Three tokens):
as a result, as an alternative, as long as, as much as, as soon as, as well as, before and after, but then again, even before then, if and when, in any case, in other words, in the end, in the meantime, in the meanwhile, on the contrary, so long as, so much as, when and if

Continuous explicit connectives (Four tokens):
at the same time, not only because of, not so much as, on the other hand

Discontinuous explicit connectives:
both+and, either+or, if+then, neither+nor, not just+but, not just+but+also, not only+also, not only+but, not only+but also, on the one hand+on the other, on the one hand+on the other hand

Figure 4: A set of 173 explicit connectives from the annotation scheme of the PDTB 3.0 corpus (Webber et al., 2019).

Ablation Method	The i^{th} Run	SQuAD 1.1	SQuAD 2.0		
			Has-Ans	No-Ans	Total
1. Randomly shuffle the order of the sentences with explicit connectives in the context	1	86.1	74.5	75.6	75.0
	2	86.0	74.4	75.3	74.9
	3	86.3	74.4	75.3	74.8
	4	86.2	73.9	75.7	74.8
	5	86.2	74.4	75.1	74.7
mean (variance)	—	86.2 (0.0)	74.3 (0.0)	75.4 (0.0)	74.8 (0.0)

Table 6: Ablation results with means and variances (in parentheses) for the shuffling-based method for skill s_1 over five different runs.

Sense	Explicit Connectives	Retention Rate (Uniqueness)	Retention Rate (Instances)
Temporal. Asynchronous. Precedence	afterward (6), afterwards (5), before (309), later (92), later on (2), next (4), subsequently (3), then (310), thereafter (11), till (4), ultimately (15), until (143)	70.59%	96.58%
Temporal. Asynchronous. Succession	after (533), by then (6), earlier (15), once (70), previously (53), since before (1)	46.15%	71.67%
Contingency. Cause. Reason	about (2), because (833), because of (12), due to (1), not only because of (1), on (1)	28.57%	64.05%
Contingency. Condition. Arg2-as-cond	depending on (3), depending upon (1), if (1084), if+then (37), in case (6), only if (13), so long as (4), whenever (9), where (2)	45%	86.62%
Comparison. Concession. Arg2-as-denier	albeit (1), but (3063), but then (3), but then again (1), even so (9), however (390), in any case (3), nevertheless (32), nonetheless (25), regardless (2), though (128), yet (96)	38.71%	92.74%
Expansion. Conjunction	additionally (7), along with (2), also (1736), and (6189), as well as (7), besides (19), beyond (1), both+and (6), but also (1), further (7), furthermore (12), in addition (165), likewise (8), moreover (103), much less (3), nor (31), not just+but (1), not just+but+also (1), not only (5), not only+also (1), not only+but (18), not only+but also (9), plus (1), separately (72)	61.54%	95.87%

Table 7: The six identified senses and their associated explicit connectives. Exclusive connectives are underlined. The fourth and last column provides the retention rate with respect to the "uniqueness" and "instances" of each sense, respectively.

Subset Ablation Method	Has-Ans 5928	No-Ans 5945	Total 11873
Original dataset	78.4	73.9	76.1
1. Randomly shuffle the order of the sentences with explicit connectives	$74.3_{-5.2}$	$75.4_{+2.0}$	$74.8_{-1.7}$
2. Drop explicit connectives associated with asynchronous temporal reasoning (precedence)	$77.8_{-0.8}$	$74.1_{+0.3}$	$76.0_{-0.1}$
	$77.5_{-1.1}$	$74.2_{+0.4}$	$75.8_{-0.4}$
3. Drop explicit connectives associated with asynchronous temporal reasoning (succession)	$78.3_{-0.1}$	$73.9_{-0.0}$	$76.1_{-0.0}$
	$76.8_{-2.0}$	$74.0_{+0.1}$	$75.4_{-0.9}$
4. Drop explicit connectives associated with causality reasoning	$77.3_{-1.4}$	$73.9_{-0.0}$	$75.6_{-0.7}$
	$64.7_{-17.5}$	$77.4_{+4.7}$	$71.0_{-6.7}$
5. Drop explicit connectives associated with conditional reasoning	$78.1_{-0.4}$	$74.0_{+0.1}$	$76.1_{-0.0}$
	$71.3_{-9.1}$	$76.4_{+3.4}$	$73.8_{-3.0}$
6. Drop explicit connectives associated with negative causality reasoning	$78.1_{-0.4}$	$74.0_{+0.1}$	$76.1_{-0.0}$
	$77.0_{-1.8}$	$74.2_{+0.4}$	$75.6_{-0.7}$
7. Drop explicit connectives associated with the expansion of explicit discourse relations	$73.6_{-6.1}$	$74.5_{+0.8}$	$74.1_{-2.6}$
	$72.2_{-7.9}$	$75.1_{+1.6}$	$73.7_{-3.2}$

Table 8: Results (%) on the development set of SQuAD 2.0 for subsets with normal (Has-Ans) and no-answer (No-Ans) questions. Values in smaller font are changes (%) relative to the original performance of the model. For mask-related methods (m_2 to m_7), the results shown in the white areas are obtained from the initial test while the results shown in the shaded areas represent the further test, i.e., dropping all explicit connectives for which each identified sense was annotated.

Why Is MBTI Personality Detection from Texts a Difficult Task?

Sanja Štajner, Seren Yenikent
Symanto Research
Nuremberg, Germany
`{sanja.stajner, seren.yenikent}@symanto.com`

Abstract

Automatic detection of the four MBTI personality dimensions from texts has recently attracted noticeable attention from the natural language processing and computational linguistic communities. Despite the large collections of Twitter data for training, the best systems rarely even outperform the majority-class baseline. In this paper, we discuss the theoretical reasons for such low results and present the insights from an annotation study that further shed the light on this issue.

1 Introduction

Apart from being long and requiring to be administrated by a skilled human assessor in artificial circumstances (laboratory conditions), the traditional questionnaire-based personality tests often introduce *social desirability bias* (Krumpal, 2011) and *the reference-group effect* (Heine et al., 2002), which can be introduced either from the subject's or assessor's side. To avoid those biases, it was suggested that analysing person's writing provides more objective assessment of one's personality than the traditional questionnaires (Stachl et al., 2019).

1.1 The MBTI

The original MBTI model was based on the comprehensive theoretical work of Carl Jung (1921), and was further developed by Myers and Briggs by adding the fourth dimension (judgment/perception) and several decades of extensive practical use within the industrial and educational settings (Briggs-Myers and Myers, 1995). Today, the MBTI is one of the most widely used non-clinical psychometric assessments, regularly used in understanding team building processes in work environments (Kuipers et al., 2009), for career suggestions (Garden, 1997), and in marketing and consumer behavior (Gountas and Gountas, 2001).

The MBTI lays out a binary classification based on four distinct functions, and draws the typology of the person according to the combination of those four values (e.g. INFP, ESTJ):

- **E**xtraversion/**I**ntroversion (EI) - preference for how people direct and receive their energy, based on the outer or inner world
- **S**ensing/**IN**tuition (SN) - preference for how people take information in, by five senses or by interpretation and meanings
- **T**hinking/**F**eeling (TF) - preference for how people make decisions, by relying on logic or emotions towards people and special circumstances
- **J**udgment /**P**erception (JP) - how people deal with the world, by organizing it or staying open for new information

While many studies investigated linguistic characteristics of the Big 5 personality traits (Mairesse et al., 2007; Furnham, 1990; Pennebaker and King, 1999; Gill and Oberlander, 2002; Scherer, 2003; Pennebaker and King, 1999; Gill and Oberlander, 2003), to the best of our knowledge, there have been no studies reporting on linguistic characteristics of different MBTI types. The most probable reason for this is a different nature of the MBTI framework. Unlike the Big 5 model that originated from lexical analyses (Cattell, 1946; Tupes and Christal, 1961; Goldberg, 1982; Costa and McCrae, 1992), the MBTI fundamentally makes use of the behavioral implications in theoretical and professional contexts. Hence, the available data rarely refers to any linguistic contexts, but more to practical results of the questionnaires. However, as linguistic data has been shown as one of the best indicators of personality-related characteristics such as behaviors and motivations (Pennebaker

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3580–3589
April 19 - 23, 2021.

and King, 1999; Tausczik and Pennebaker, 2010) and given that the MBTI makes use of behavioral implications to study personality, there is a room for linguistic representation of the concept with proper datasets and methods.

1.2 Automatic Detection of MBTI from Texts

Unlike automatic detection of the Big 5 personality traits that in the last 15 years has been attempted at from various types of texts, e.g. essays (Argamon et al., 2005), personal weblogs (Oberlander and Nowson, 2006), and Facebook posts (Kosinski et al., 2013), the automatic detection of MBTI gained popularity only recently and exclusively using Twitter data. Attempts were made for various languages: English (Plank and Hovy, 2015), six Western European languages (Verhoeven et al., 2016), and Japanese (Yamada et al., 2019). All those studies, despite using large training datasets (over 1M instances) and various features (word and character n-grams, or count-based meta-features such as number of followers, favourites, etc.), barely managed to outperform the majority-class baseline, and even that only in some of the four MBTI dimensions. The best English models were trained on over 1M Twitter instances using logistic regression classifier and combining linguistic and count-based meta-features. Nevertheless, they outperformed the majority-class baseline only on the IE and TF dimensions, achieving the accuracy of 72.5% and 61.5% on those binary tasks (Plank and Hovy, 2015).

Comparison of performances of the Big 5 and MBTI computational models trained on Twitter data showed that type of architecture and settings practically have no influence on the MBTI detection from such data (Celli and Lepri, 2018) indicating thus that Twitter data might not contain sufficient amounts of lexical signals.

1.3 Goals and Contributions

The main goal of our study is to investigate why the automatic detection of MBTI personality traits from texts does not outperform even the simple majority-class baseline and why its performance on Twitter data is not influenced by the architecture type and settings. Furthermore, we shed some light on the natural complexity of the task and discuss the theoretical constraints of the task. We pose two hypotheses:

- H1: *Twitter data does not contain enough signals for MBTI personality detection.*
- H2: *Textual data does not resonate well with MBTI personality scores from questionnaires.*

To test those hypotheses, we collect a new dataset, write the guidelines for human annotation, conduct an extensive human annotation task, and provide both a quantitative and qualitative analysis of the results.

The main contributions of our study are the following:

1. Proposing the guidelines that translate behavioural characteristics of the MBTI dimensions into linguistic cues in texts;
2. Releasing a new dataset for MBTI analysis from textual data;
3. Testing the two above-mentioned hypotheses (H1 and H2);
4. Revealing the specificities of the MBTI constructs that make them difficult for automatic detection from texts.

2 Datasets

To test our hypotheses, we used two datasets, a subset of the MBTI-Twitter dataset (Plank and Hovy, 2015), and a dataset that we collected especially for this purpose, the MBTI-MTurk dataset.[1]

2.1 MBTI-Twitter Dataset Selection

To test our first hypothesis (H1), we randomly selected 96 user data from the MBTI-Twitter dataset (Plank and Hovy, 2015), six for each of the 16 MBTI types, using the version with 50 concatenated tweets for each user/instance. Out of those 50 concatenated tweets, we only retained the first 10 tweets for two reasons: (1) to maintain the task managable for human annotation; and (2) to have the length of posts from each user roughly comparable to those we collected (MBTI-MTurk).

2.2 MBTI-MTurk Dataset Compilation

Our goal was to compile an 'ideal' dataset of short text snippets that would maximize the potential of finding linguistic signals of the four MBTI personality dimensions.

---

[1] Available upon request for research purposes.

We posted a human intelligence task (HIT) using the Amazon Mechanical Turk (MTurk) crowdsourcing platform. The task consisted of three open-ended questions:

1. You might have done an MBTI personality test in the past. If you did, and you know the MBTI personality type you obtained, please write it here:
2. What is your favourite type of vacation and why?
3. Which are your favourite hobbies and why?

The first question was optional, while the other two questions were required for completing the task and getting monetary compensation. For the last two questions, we enforced the answers to be at least 300 characters long, to allow for a sufficient length to capture some linguistic signals. The last two questions were selected following the assumption that how people spends their free time would be the most natural version of their personality. Research suggests that activities engaged during leisure time are the outcomes of what the person wants to be doing by choice with high levels of intrinsic motivation (Kuykendall et al., 2015). Therefore, we considered that the types of vacation and hobbies are representative realms of the participants' true personalities.

Data Collection. We posted our HIT in 20 batches spanning a two-months period. From all collected data, we first filtered out those that did not contain the answer to the MBTI personality type question.[2]

Quality Control. All collected answers were manually checked to filter out those answers that were copied from the internet (about one third of all participants tried to trick the system by copying texts from internet about favourite types of vacations and hobbies).

Consistency Check. Out of all HITs that contained the answer to the MBTI question, we only retained those for which we had two HITs from the same user completed with at least one month in between and where the asnwer to the MBTI question contained the same personality type both times. The assumption was that if a user provided two different MBTI results in those two instances it was for one of the following reasons: (1) the user gives random MBTI types to intentionally harm the study (Ipeirotis et al., 2010), or (2) in at least one of the MBTI dimensions, the user is somewhere in the middle range, and might have genuinely obtained two different scores on an MBTI questionnaire.

Final Dataset. The final dataset for this study consisted of 96 HITs, all completed by different users (MTurk IDs)[3]. Similar as in the case of the MBTI-Twitter dataset, the selected dataset comprises of equal number of HITs (six) per each of the 16 MBTI personality types. For the two pilot rounds, the additional 30 HITs were used, ensuring that each of the eight polarities (extravert, introvert, sensing, intuitive, etc.) is present at least twice.

3 Annotation

As there are no studies laying out linguistic characteristics of the MBTI model, we created the annotation guidelines starting from the behavioral characteristics. The MBTI's theoretical and practical framework provides detailed profiles for each type ranging from general and typical characteristics to real-time depictions of those characteristics, in the contexts of relationships, marriage, work, and learning settings (Briggs-Myers and Myers, 1995). We translated this behavioral information into linguistic and textual signals in general, as well as some specific signals relevant for the contexts we asked the participants to write about.

To define the characteristics that would linguistically distinguish the two polarities of the EI dimension, we focused on the processes of socialization depicted in the text. As the dimension of JP is mainly related to organizational preferences, we chose the linguistic signals referring to arrangement of schedules and plans. The fact that people high on J tend to be more exact and pay more attention to details, whereas people high on P show more flexible and accidental characteristics (Briggs-Myers and Myers, 1995) we translate into guidelines considering sentence structures and grammar. The SN dimension identifies preferences for the characteristics of tasks and information people would like to process, either using facts and five senses (S), or using imagination and abstraction

---

[2] We opted for making the MBTI personality type question optional to avoid people writing random MBTI types, as those are widely known and popular on the internet. By paying equally those who provided the answer to the MBTI question and those who did not, we tried to ensure that provided MBTI types are not just randomly chosen but rather represent the real results of the user's MBTI testing.

[3] From each user ID, that passed all above-mentioned checks, we randomly chose one answer about the hobbies and one answer about the vacations.

Extravert	Introvert
Mention of new people (e.g. *crowd*, *strangers*)	Mention of closer people rather than any group of people (e.g. *husband*, *family*)
Mention of social activities and events that contain interaction with other groups of people (e.g. *party*, *dancing*, *couchsurfing*)	Mention of individual activities or activities that can be done without interaction with other people (e.g. *by myself*, *spending time at home*)
Mention of outside world and vibrant places (e.g. *bars*, *restaurants*)	Mention of inner world, and calm and quiet places (e.g. *home*, *museum*)
We references	*I* references
Use of intensifiers and exclamation marks	Hedging
More assertive, positive, enthusiastic arguments	Less assertive arguments

Table 1: Linguistic signals of extraversion and introversion.

Sensing	Intuitive
Technical, object-based and hands-on hobbies	Inspirational and imaginative hobbies (e.g. *creating*, *exploring*)
Facts and real cases (e.g. *documentary*, *diary*)	Abstraction rather than facts (e.g. *sci-fi*, *cartoons*)
Details and examples (more adjectives and adverbs to provide details, use of the words *example*, *for instance*)	Main ideas rather than details
Needs to use the 5 senses	Needs to focus on the bigger picture
Puzzles, model planes, crafts, carving, rowing, sailing, diving, rock climbing, etc.	Painting, music, dancing, poetry, chess, literature, arts, martial arts, yoga, meditation, etc.
Simplified and straightforward writing style (short sentences)	Complex writing style (long sentences)
Clear and concise writing style	Artistic, longer, more words

Table 2: Linguistic signals of sensing and intuition.

Thinking	Feeling
Logical reasoning for their actions and choices (e.g. *reading books for learning*)	Emotional reasoning for their actions and choices (e.g. *reading books for gateway feeling*)
Mention of opinions, ideas, comparisons	Mention of people, values, feelings
Direct (e.g. *reading is nice*)	Tactful, indirect (e.g. *reading feels nice*)

Table 3: Linguistic signals of thinking and feeling.

Judging	Perceiving
Holidays that include planning such as ski holidays, city tours etc. (e.g. *tour*, *pass*, *ticket*, *reservation*)	Spontaneous holidays such as going to the beach, a new city etc. (e.g. *flexible*, *spontaneous*)
Decisive, planful, organized (e.g. *plan*, *schedule*, *followed by*)	Curiosity, anticipation of change, and spontaneity
Organizers of the plans (e.g. *invite*, *organize*)	Followers of the plans (e.g. *join*, *tag along*)
Warranty (e.g. *insurance*, *make sure*)	Autonomy and impulsiveness (e.g. *suddenly*, *out of the blue*, *last minute*)
Past tense or present perfect tense	Present simple tense
Formal and structured writing style with grammatical rules followed as much as possible (e.g. *I like ski holidays and sometimes prefer city tours.*)	Informal writing style with grammar mistakes (e.g. *I like going to the beach. Also, do art sometimes.*)

Table 4: Linguistic signals of judging and perceiving.

(N). The TF dimension is related to preferences for decision-making processes. Hence, we focus on the reasoning aspect of the linguistic characteristics, whether it is logical or emotional.

Tables 1– 4 provide linguistic signals for each label. They were provided to the hired annotators as the main annotation guidelines.

3.1 Annotators

Two annotators, one with a PhD degree in psychology and the other in computational linguistics, were hired to annotate all instances in both datasets. Both annotators underwent a six-months (paid) training which covered reading the extensive literature on personality assessment (e.g. traditional questionnaire approaches and dictionary-based methods), the MBTI framework and its use cases.

3.2 Annotation Procedure

We asked the annotators to assign to each instance (either coming from the MBTI-Twitter dataset or from Hobbies and Vacation questions in the MBTI-MTurk dataset)[4], for each of the MBTI dimensions separately, one of the following four labels: either of the two polarities (e.g. E or I for the EI dimension) etc.), *unsure* (in cases where they saw signals from both polarities and are not confident to make a binary decision), or *not enough signal* (in cases where they did not find any signals for any of the two polarities).

Both annotators were first asked to complete two pilot rounds of annotation so that they can be fully familiarised with the annotation guidelines and the procedure. In each pilot round, the dataset they were ask to annotate consisted of 15 MBTI-Twitter and 15 MBTI-MTurk user-instances which were not used for the final round. After each pilot round, a question and answering session was organized to address all potential issues with the guidelines or the procedure. Furthermore, after each pilot round, the annotators showed their annotations and commented their decisions to the other annotator to calibrate their annotations. During the pilot rounds, the annotators were asked to mark the parts of the instances which guided their decisions for assigning certain polarity.

The annotations and text mark-ups obtained during the pilot rounds are used in Section 4 to show how the proposed guidelines were used in practice, as well as to point out the most challenging aspects of the annotation process.

After finishing both pilot rounds, the annotators were given the final dataset which consisted of 96 user-instances from the MBTI-Twitter dataset and 96 user-instances from the MBTI-MTurk dataset. In total, each annotator annotated 192 user-instances for each of the four MBTI personality dimensions. The annotators were instructed to have enough breaks to avoid the fatigue effect.

4 Findings from the Pilot Rounds

The annotation and mark-up obtained during the pilot rounds revealed several important characteristics of the task.

4.1 Middle Cases

One of the recurring issues found during the pilot rounds was the case of people whose answers truly belong to the middle of the spectrum, as in the following case (the signals for introversion are shown in italics, those for extraversion are shown in bold, and those for the JP dimension are shown underlined):

(1) “I like travelling to some new places. *Mostly* I like travelling to a city I have never been before, but from time to time I *can* also enjoy just going to a nice seaside place and relax for a week without any fixed plans and sightseeing schedules. The only type of holidays **I really don’t like** is **just staying at home**. I find that OK if just for a day or two maximum, but longer than that **I get bored**.”

The words *mostly* and *can* represent hedging and are thus signals for introversion (Table 1), whereas the word *really* is an intensifier and thus a signal for extraversion (Table 1). The negation of the fixed plans and schedules signalizes a truly mid-range personality along the JP dimension, as it shows that the person does not like fixed plans and schedules, but is still aware of their existence and thus mentions them.

The ‘middle cases’ were frequent also for the other two dimensions. The following example was annotated as *unsure* for the SN dimension by both annotators (the signals for sensing are shown in bold, and the signals for intuition are shown in italics):

---

[4] The answers to the Hobbies question were used for annotating only SN and TF dimensions, and the answers to the Vacations question were used for annotating only EI and JP dimensions.

(2) “I enjoy doing sports, although its been a while since I have been active. Such like **volleyball**, **gym fitness**, and I am trying to get into *yoga*. On the complete opposite side, I also really love *baking and exploring new recipes to cook*. In my spare time I am trying to learn to appreciate being outdoors more, as I usually spend too much time binging netflix.”

The noticeable amount of such ‘middle cases’ is, however, not caused by the flaw of the guidelines, but rather reflects the drawbacks of the binary nature of the MBTI framework. Those people who have characteristics of both polarities, which is a common case (Pittenger, 1993), are by the traditional questionnaire-based assessment placed in one of the two groups (e.g. extravert or introvert). Analysis of the short posts can reveal the presence of signals for both polarities but their ratio in the short posts might be different from that obtained by the questionnaire-based assessment.

4.2 Content vs. Style

As can be seen in Tables 1– 4, the annotation guidelines contain pointers regarding both content (i.e. lexical choices and argumentation) and style (i.e. grammatical and stylistic preferences). The pilot phases discovered that in approximately 10% of the cases, a given instance shows the content-based signals of one polarity and the stylistic-based signals of the opposite polarity. One such example is the following, where the content cues are clearly of the judging polarity (shown in italics), while the stylistic cues (sentence structure, grammatical errors, and typos) are of the perceiving polarity (shown in bold):

(3) “I *would like my vacation to be well organised* so **i wont** have to deal with **anything just** enjoy the flow. I like visiting new places, that is **me** perfect vacation. **Its** not interesting for me going several times in one place just because **its quite**. **Don’t** like big buildings, crowded places and cemented environments**, i** like nature and historical places”

In all such cases, the polarity that was signalized by the stylistic preferences was in line with the reported official MBTI label. It is known that people can consciously change the content of their answers, but not the style (Chung and Pennebaker, 2007). Therefore, the content of the answers to the open-end questions can also suffer from social desirability bias, similar as the answers to the questionnaires.

4.3 Writing Style in Twitter

Another issue that was revealed during the pilot rounds was that twitter posts, by their nature, often have incorrect grammar, punctuation, and ill-formed sentences. That makes it difficult for the annotators to assign J or P labels unless there are lexical cues (which are extremely rare in Twitter), because otherwise, focusing on writing style (Table 4) the great majority of posts would be annotated as P. Similarly, the nature of the Twitter posts to overuse intensifiers and exclamation marks makes many posts stylistically extraverted, and thus, the last two points in the annotation guidelines (Table 1) may lead to false (extravert) positives.

Two examples of instances from the MBTI-Twitter dataset are given later, in Table 6.

5 Final Annotation Results

The annotation of the final dataset that consists of 96 instances from the MBTI-Twitter dataset and 96 instances from the MBTI-MTurk dataset revealed further particularities of each dataset, and shed some light on how feasible the task is, even for trained human annotators.

5.1 The Presence of the MBTI Signals

As expected, in the final datasets, the instances from the MBTI-MTurk dataset were reported to have enough signals across all four MBTI dimensions, as opposed to the instances in the MBTI-Twitter dataset (Table 5). In the MBTI-Twitter dataset, many instances were reported to have insufficient signals across the JP and SN dimensions.

These results support our first hypothesis (H1) that Twitter posts (even when grouped per user) do not always contain linguistic signals to allow for personality detection, even for the trained human annotators. Furthermore, these results give a potential explanation to the question why previously proposed binary classification models for the JP and SN dimensions did not manage to outperform even the majority-class baselines.

In Table 6, we present examples of two Twitter users, one for whom both annotators reported insufficient signals for the JP and SN dimensions, and another one whose personality was correctly annotated across all four dimensions by both annotators.

Annotator	Statistic	MBTI-Twitter				MBTI-MTurk			
		EI	SN	TF	JP	EI	SN	TF	JP
A (CL)	Not enough signal	2	31	8	59	0	0	0	0
	Unsure	14	14	22	7	31	23	38	33
	Confident (class assigned)	84	55	71	33	69	77	62	67
B (Psychologist)	Not enough signal	10	17	10	33	0	0	0	0
	Unsure	29	17	19	25	23	39	54	7
	Confident (class assigned)	61	67	71	42	77	61	46	93

Table 5: Human annotation statistics (in percentage of cases, out of the total number of 96 instances) on the MBTI-MTurk dataset (Holidays and Hobbies) and the subset of the MBTI-Twitter dataset.

Insufficient signals for any dimension	Correctly annotated (ESFP) by both annotators
“what happened in the fandom why are the makorra shippers so angry ? @URL / 5mfxado0lp photoset : calmorrison : aer-dna : sweetlikepoison 528 : aer-dna : ” just the four of us . ” remember when team ... @URL / hdcrf 7ljsc omg remind me to not go into the tags i just saw a post that essentially said makorra was baited like how ... @URL / jxdtfcmlld dylanftsw : shut the fuck up about “ shitty writing ” or the end of the show not making sense or being rushed .... @URL / r62qnjt7di photo : hellkatespangled : sleepy girlfriends and lazy art @URL / 5oq9n8P2Kd wall-maria-around-ba-sing-se : so i saw this gif : and all i could think was how it made her look like an ... @URL / gom 5newfum photo : insomniadiesdown : some kuvira sketches @URL / ahrgfwwqsb photo : joeldosreisviegas : dariucdraws :...”	“omg i just met doug adams from season 12 of top chef , here in portland , the dude was super mellow and friendly ! :D @USER craaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaling in my skiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii-iiiiiiiiiiiiiiiiiiiiii @USER he was super chill . i shook his hand and told him i loved his attitude and was rooting for him , and he was just like , ” cool ! ” @USER hey cat , you guys get home okay ? @USER i sure hope so ! i’ve loved doug since the beginning of this season his attitude is great , he’s talented , he never whines @USER conversion ” therapy ” is indeed the greatest sham it’d be an amazing step to see it banned , and lgbt’s treated normally @USER it’s essentially the last bastion of clinical homophobia the day it dies is the day attitudes towards us improve drunk tonight and just can’t get over how awesome bjork is <3 ...”

Table 6: An MBTI-Twitter instance which did not contain sufficient signals and could not be annotated (left) as opposed to another MBTI-Twitter instance which was correctly annotated by both annotators across all four dimensions (right). In both cases, we do not present the full instance for the space constraints.

5.2 Assigned Classes

We found that in the MBTI-MTurk dataset, a relatively high number of instances (ranging between 7% and 54% depending on the dimension and annotator, with most of them being in the range between 23% and 39%) was reported to have mixed signals (labelled as *unsure*).

As expected, the number of instances for which the annotators felt confident enough to assign one of the polarities was higher in the MBTI-MTurk than in the MBTI-Twitter dataset, in most of the cases. The difference was most noticeable in the JP dimension, where the number of instances with an assigned polarity doubled.

A closer look at the instances for which both annotators were confident enough to assign either of the polarities revealed even more prominent differences between the two datasets (Table 7). In the MBTI-Twitter dataset, only in 46% of the instances the annotators agreed on the EI dimension, and only in 50% cases they agreed on the JP dimension. In the MBTI-MTurk dataset, in contrast, the annotators agreed across those two dimensions (EI and JP) in 100% and 78% of the cases, respec-

Statistic	MBTI-Twitter				MBTI-MTurk			
	EI	SN	TF	JP	EI	SN	TF	JP
Both annotators confident	53	**30**	43	**15**	62	**54**	38	**69**
Annotators agree	**46**	*81*	61	**50**	**100**	*69*	62	**78**
Annotator A agrees with the gold label	77	64	64	53	78	54	77	44
Annotator B agrees with the gold label	47	44	54	47	60	54	85	42
Both annotators agree with the gold label	77	54	57	50	75	62	54	43

Table 7: Inter-annotator agreement statistics on the MBTI-MTurk dataset (Holidays and Hobbies) and the subset of the MBTI-Twitter dataset. The percentage of instances, out of those for which both annotators were confident to assign a polarity, for which both annotators assigned the same polarity is presented in the row "Annotators agree". Similarly, the next two rows present the percentage of cases in which each annotators label was equal to the gold label (the provided MBTI label) out of all cases for which the annotator was confident to assign a polarity (as opposed to the other two labels: 'unsure' and 'not enough signal'). The percentage of cases in which both annotators agree with the gold label is calculated taking into account only those instances on which both annotators agreed.

tively. Nevertheless, if we calculate the percentage of cases in which the 'gold' label was the same as the shared label of the two annotators (the row 'both annotators agree with the gold label' in Table 7), we find that, surprisingly, it only happens in up to 50% of the cases for the JP dimension. This indicates that the linguistic signals for the JP dimension captured from text, even with high confidences of the annotators and their inter-annotator agreement, are not correctly associated with the results of the traditional questionnaire-based MBTI personality assessment. The only MBTI dimension for which the percentage of cases in which both annotators agreed with the gold label is noticeably above the majority-class baseline (50%) in both datasets is the EI dimension. These results indicate that the EI dimension is the only dimension for which a noticeable association between the results of the traditional questionnaire-based MBTI personality assessment and the textual-analysis-based MBTI personality assessment was found. This evidence supports our second hypothesis (H2) for three out of four MBTI dimensions.

6 Discussion and Conclusions

The results of presented analyses shed some light on possible causes of the poor performances of the automatic MBTI personality detection systems proposed so far. Furthermore, they indicate that the linguistic cues found in short texts do not seem to directly correspond to the results of the questionnaire-based results, which are commonly used as the 'gold labels' in classification experiments. This is in line with academic studies that showed the psychometric inadequacy of the questionnaire (Pittenger, 1993; Boyle, 1995).[5]

The high number of instances without enough signal and with mixed signals in the MBTI-Twitter dataset across all MBTI dimensions, and especially the JP dimension, leads to high amounts of noise in the training datasets. Therefore, it is not a surprise that the best systems trained on those datasets rarely outperform even the majority-class baseline.

Even in the cases where user-instances contain sufficient signals and the agreement between the human annotators is high, the agreement between the annotators and the gold label is still very low for three out of four dimensions (Table 7). These results indicate that the constructs set up by the traditional questionnaire-based personality assessment might not have the exact equivalent translation into the linguistic cues. The only exception for this is the EI dimension where the agreement between the annotators and the gold label reaches 75%-77%. This is somewhat expected as the EI is the highest correlated dimension between the MBTI and Big 5 models amongst all dimensions (Furnham, 1996), and in the Big 5 model, the EI has a good linguistic correspondence.

Finally, the results of the in-depth analyses performed during the two pilot rounds (Section 4) revealed three phenomena that need to be taken into

---

[5] Due to the methodology followed to develop and improve the questionnaire (i.e. qualitative methods such as observations and introspection), the MBTI has received considerable criticism for not relying on a scientifically proven background (i.e. data-driven approaches).

account when using the linguistically-based MBTI personality analysis from texts:

(1) The content and the style of the text sometimes exhibit signals of the opposite polarities;

(2) Many people naturally express signals of the opposite polarities, as they probably belong to the middle ranges of those personality dimensions;

(3) The language used in Twitter shows specific stylistic characteristics in terms of tonality, use of exclamation marks, sentence structure and grammar, thus making everyone seem more extraverted than they are.

References

Shlomo Argamon, Sushant Dhawle, Moshe. Koppel, and James W. Pennebaker. 2005. Lexical predictors of personality type. In *Proceedings of the Joint Annual Meeting of the Interface and the Classification Society of North America*.

Gregory J. Boyle. 1995. Myers-briggs type indicator (mbti): Some psychometric limitations. *Australian Psychologist*, 30(1).

Isabel Briggs-Myers and Peter B. Myers. 1995. *Gifts differing: Understanding personality type*. Davies-Black Publishing.

Raymond B. Cattell. 1946. *The description and measurement of personality*. Yonkers-on-Hudson.

Fabio Celli and Bruno Lepri. 2018. Is Big Five Better than MBTI? A Personality Computing Challenge Using Twitter Data. In *CLiC-it*.

Cindy Chung and James W. Pennebaker. 2007. The psychological functions of function words. In *Social communication*, pages 343–359. New York: Psychology Press.

Paul T. Costa, Jr and Robert R. McCrae. 1992. *Revised NEO Personality Inventory (Neo-PI-R) and NEO Five-Factor Inventory (NEO-FFI): Professional manual*. Psychological Assessment Resources.

Adrian Furnham. 1990. *Handbook of Language and Social Psychology*, chapter Language and personality. Winley.

Adrian Furnham. 1996. The big five versus the big four: the relationship between the myers-briggs type indicator (mbti) and neo-pi five factor model of personality. *Personality and Individual Differences*, 21(2).

Anna Garden. 1997. Relationships between mbti profiles, motivation profiles, and career paths. *Journal of Psychological Type*, 41.

Alastair J. Gill and Jon Oberlander. 2002. Taking care of the linguistic features of extraversion. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, pages 363—-368.

Alastair J. Gill and Jon Oberlander. 2003. Perception of e-mail personality at zero-acquaintance: Extraversion takes care of itself; neuroticism is a worry. In *Proceedings of the 25th Annual Conference of the Cognitive Science Society*, pages 456—-461.

Lewis R. Goldberg. 1982. From Ace to Zombie: Some explorations in the language of personality. *Advances in personality assessment*, 1:203–234.

John Gountas and Sandra Gountas. 2001. A new psychographic segmentation method using jungian mbti variables in the tourism industry. *Consumer psychology of tourism, hospitality and leisure*, 2.

Steven J. Heine, Darrin. R. Lehman, Kaiping Peng, and Joe Greenholtz. 2002. What's wrong with cross-cultural comparisons of subjective likert scales?: The reference-group effect. *Journal of Personality and Social Psychology*, 82(6):903—-918.

Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*.

Carl G. Jung. 1921. *Psychological Types: Volume 6*. Routledge.

Michal Kosinski, David Stillwell, and Thore Graepell. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 15:5802–5805.

Ivar Krumpal. 2011. Determinants of social desirability bias in sensitive surveys: a literature review. *Quality Quantity*, 47(4).

Ben S. Kuipers, Malcolm J. Higgs, Natalia V. Tolkacheva, and Marco C. de Witte. 2009. The influence of myers-briggs type indicator profiles on team development processes: An empirical study in the manufacturing industry. *Small Group Research*, 40(4).

Lauren Kuykendall, Louis Tay, and Vincent Ng. 2015. Leisure engagement and subjective well-being: A meta-analysis. *Psychological Bulletin*, 141(2).

François Mairesse, Marilyn A. Walker, Matthias R. Mehl, and Roger K. Moore. 2007. Using linguistic cues for the automatic recognition of personality in conversation and text. *J. Artif. Int. Res.*, 30(1):457–500.

Jon Oberlander and Scott Nowson. 2006. Whose thumb is it anyway? classifying author personality from weblog text. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

James W. Pennebaker and Laura A. King. 1999. Linguistic styles: Language use as an individual difference. *Journal of personality and social psychology*, 77(6):1296–1312.

David J. Pittenger. 1993. Measuring the MBTI… and coming up short. *Journal of Career Planning and Employment*, 54:48–52.

Barbara Plank and Dirk Hovy. 2015. Personality traits on twitter—or—how to get 1,500 personality tests in a week. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 92–98, Lisbon, Portugal. Association for Computational Linguistics.

Klaus R. Scherer. 2003. Vocal communication of emotion: A review of research paradigms. *Speech Communication*, 40 (1-2):227—-256.

Clemens Stachl, Florian Pargent, Sven Hilbert, Gabriella M. Harari, Ramona Schoedel, Sumer Vaid, Sam Gosling, and Bühner Markus. 2019. Personality Research and Assessment in the Era of Machine Learning.

Yla R. Tausczik and James W. Pennebaker. 2010. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Ernest C. Tupes and Raymond E. Christal. 1961. Recurrent personality factors based on trait ratings. *USAF ASD Technical Report*, pages 61–97.

Ben Verhoeven, Walter Daelemans, and Barbara Plank. 2016. Twisty: a multilingual twitter stylometry corpus for gender and personality profiling. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1632–1637, Portoroz, Slovenia. European Language Resources Association (ELRA).

Kosuke Yamada, Ryohei Sasano, and Koichi Takeda. 2019. Incorporating textual information on user behavior for personality prediction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 177–182, Florence, Italy. Association for Computational Linguistics.

ENCONTER: Entity Constrained Progressive Sequence Generation via Insertion-based Transformer

Lee-Hsun Hsieh **Yang-Yin Lee** **Ee-Peng Lim**
Singapore Management University
Singapore
jesus255221@gmail.com yylee@smu.edu.sg eplim@smu.edu.sg

Abstract

Pretrained using large amount of data, autoregressive language models are able to generate high quality sequences. However, these models do not perform well under hard lexical constraints as they lack fine control of content generation process. Progressive insertion-based transformers can overcome the above limitation and efficiently generate a sequence in parallel given some input tokens as constraint. These transformers however may fail to support hard lexical constraints as their generation process is more likely to terminate prematurely. The paper analyses such early termination problems and proposes the ENtity-CONstrained insertion TransformER (ENCONTER), a new insertion transformer that addresses the above pitfall without compromising much generation efficiency. We introduce a new training strategy that considers predefined hard lexical constraints (e.g., entities to be included in the generated sequence). Our experiments show that ENCONTER outperforms other baseline models in several performance metrics rendering it more suitable in practical applications. [1]

1 Introduction

The field of Natural Language Generation (NLG) (Gatt and Krahmer, 2018) has seen significant improvements in recent years across many applications such as neural machine translation (Bahdanau et al., 2015), text summarization (Chopra et al., 2016), poem generation (Zugarini et al., 2019) and recipe generation (H. Lee et al., 2020). Constrained text generation (CTG) is one of the challenging problems in NLG that is important to many real world applications but has not been well addressed. CTG imposes input constraints which may be in the form of objects expected to exist in the generated text or rules over objects in the generated text (Hokamp and Liu, 2017). The objects here can be entities, phrases, predefined nouns, verbs, or sentence fragments. The constraints can be categorized into two types: (1) *Hard-constraints* which require mandatory inclusion of certain objects and complete compliance of given rules (Post and Vilar, 2018; Hu et al., 2019; Miao et al., 2019; Welleck et al., 2019; Zhang et al., 2020); and (2) *Soft-constraints* which allow the some constraint objects or rules to be not strictly enforced in the generated text (Qin et al., 2019; Tang et al., 2019). As autoregressive models generate tokens from left to right, they cannot easily support constraints involving multiple input objects, hard-constrained text generation therefore often requires non-autoregressive models.

Recently, Zhang et al. (2020) proposed a non-autoregressive hard-constrained text generation model (POINTER) that generates a text sequence in a progressive manner using an insertion-transformer (Stern et al., 2019). To train an insertion transformer to generate a missing token between every two tokens in an input sequence, the training data is prepared by masking "less important" tokens in the original text sequence in an alternating manner. The process is then repeated using the masked input sequence as the new original sequence, and further masking alternate tokens in it. The process ends when the masked sequence meets some length criteria.

While POINTER shows promising results, it does not consider hard constraints which involve entities that must be included in the generated sequence. Such entity constraint requirements are unfortunately prevalent in many applications. For example, we may want to generate a job description with some given skills, or a food recipe with some given ingredients.

A naive approach to the problem is to apply con-

---

[1] Our code is available at `https://github.com/LARC-CMU-SMU/Enconter`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3590–3599
April 19 - 23, 2021.

straints on the POINTER's masking strategy forcing it to keep entity tokens. We call this modified model POINTER-E. Although this allow entity information entering POINTER-E, another problem rises. POINTER-E suffers from *cold start problem* which refers to the inability to generate meaningful tokens at the early stages of inference forcing the generation to end prematurely. This issue can be attributed to the POINTER-E's top-down masking strategy for training the insertion transformer and the tokens of input entities not evenly spread out across the sequence.

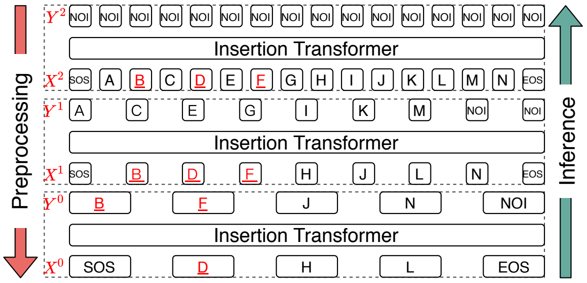
To solve the cold start generation problem, we propose ENCONTER that incorporates bottom-up masking strategy. ENCONTER supports hard entity constraints, and encourages more meaningful tokens to be generated in the early stages of generation thus reducing cold start. On top of that, we further introduce the balanced binary tree scheme (Stern et al., 2019) to reduce the number of stages in generation and to improve the efficiency of generation.

2 Entity Constrained Sequence Generation

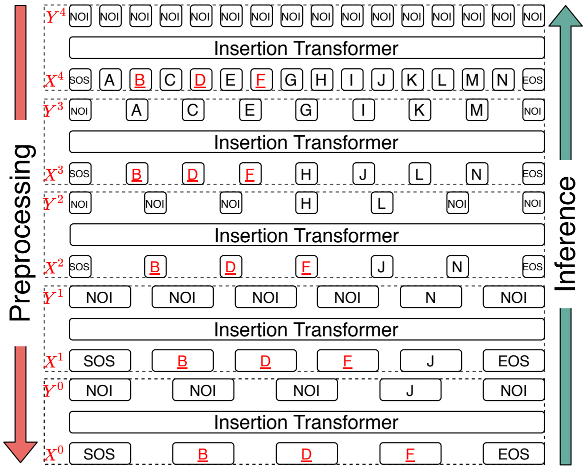
In this section, we first describe the state-of-the-art POINTER model, its preprocessing of training data and inference process. We highlight the pitfalls of the entity constrained variant of POINTER, POINTER-E. We then present our proposed entity constrained insertion transformer called ENCONTER.

2.1 POINTER

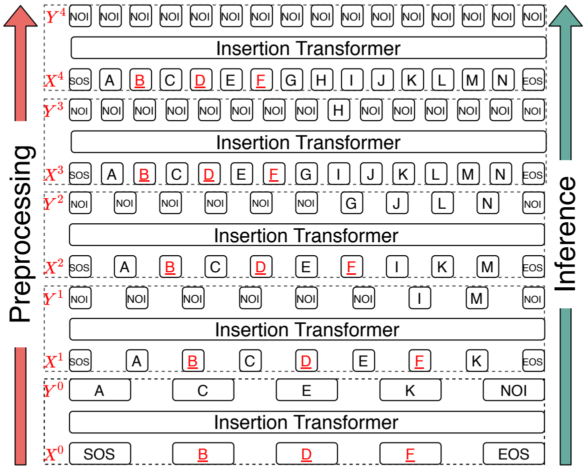
POINTER adopts a progressive masking approach to train an insertion transformer. Let $X = \{x_1, x_2, \ldots, x_T\}$ denote a sequence where $x_t \in V$, where T is the sequence length and V is a finite vocabulary set. Suppose X is a training sequence, POINTER preprocesses it to obtain the training pairs $S = \{(X^k, Y^k) | k \in \{K, \ldots, 0\}\}$ using a progressive masking strategy. As shown in Figure 1a, in each stage X^k represents the input sequence for stage k, and Y^k represents the sequence of masked tokens to be inferred. X^K is identical to the final training sequence $X^K = X$, and there should not be any additional tokens to infer. X^0 on the other hand represents the initial lexical constraints. In stage k, Y^k are the tokens to be predicted between adjacent tokens of X^k. A special no-insertion token $[NOI]$ is added to the vocabulary V and used in Y^k to indicate that no token is to be generated between adjacent tokens. Y^K is thus a sequence of all $[NOI]$'s indicating the end of generation. WordPiece (Wu et al., 2016) tokenization is applied in POINTER, and tokens split from the same word share the same score.



(a) POINTER masking.



(b) POINTER-E masking.



(c) ENCONTER insertion.

Figure 1: POINTER, POINTER-E, and ENCONTER with original sequence $X = \{A, B, C, D, E, F, G, H, I, J, K, L, M, N\}$ where B,D, and F are the tokens forming the entity constraints. The stopping criteria for POINTER is set to $n = 3$.

Token importance scoring POINTER assigns each token $x_t \in X$ an importance score α_t:

$$\alpha_t = \alpha_t^{TF-IDF} + \alpha_t^{POS} + \alpha_t^{YAKE}, \tag{1}$$

where α_t^{TF-IDF}, α_t^{POS}, and α_t^{YAKE} denote term frequency-inverse document frequency (TF-IDF), POS tag scores and YAKE (Campos et al., 2020) keyword scores, respectively. These scores are normalized to [0,1]. α_t^{POS} is defined such that the scores of nouns and verbs are higher than those of other POS tags. The token importance scores are used to derive the masking pattern Y^{k-1} of stage $k-1$ from X^k.

POINTER adopts four criteria to derive Y^{k-1} from X^k: (1) Y^{k-1} can only include non-adjacent tokens in X^k; (2) the number of tokens to be masked are maximized in each stage to make the model more efficient; (3) less important tokens are masked before more important ones and (4) A stopping criteria n is defined. The algorithm stops when $|X^k| = n$. Kadane's algorithm (Gries, 1982) has been use in POINTER to fulfill the criteria. Specifically, the algorithm selects as many unimportant tokens as possible to be masked while not masking two adjacent tokens. X^0 is automatically determined when $|X^k| = n$, it does not necessarily match the way the initial input sequence is provided by real world applications or users, including the entity constraints.

Inference Given X^0 as input sequence, POINTER starts to infer $\hat{Y}^0$ and combines the two sequences to get $\hat{X}^1 = \left\{\hat{x}_1^0, \hat{y}_1^0, \hat{x}_2^0, \hat{y}_2^0, \ldots, \hat{x}_{|\hat{X}^0|}^0, \hat{y}_{|\hat{X}^0|}^0\right\}$. If $\hat{y}_t^0$ happens to be $[NOI]$, it will be deleted and leaving only non-$[NOI]$ tokens in $\hat{X}^1$. The process repeats until all the generated tokens in $\hat{Y}^k$ are $[NOI]$s.

As shown in Figure 1a, entities may not be preserved during the preprocessing steps and the lexical constraint X^0 is not guaranteed to cover entity constraint X^e even entity tokens are assigned high importance scores. The trained POINTER therefore may not be able generate a sequence successfully when given entity constraints during the inference. We therefore propose some changes to POINTER to make it entity-aware.

2.2 Entity Aware POINTER (POINTER-E)

The entity-aware POINTER model, POINTER-E, adopts a different preprocessing approach. Let $X^e \subset X$ be an ordered sequence of entity tokens (e.g., the person names in a news document). As X^e is likely to be used as the initial generation input (i.e., $X^0 = X^e$), POINTER-E's preprocessing does not mask these entity tokens over the different preprocessing stages. This way, the model is trained to focus on generating tokens around the entities. Such tokens form the context around the entities and context relating one entity to others. We achieve such goal by ignoring the importance scores applied on entity tokens. That is, we only compute α_t for $x_t \notin X^e$.

We then apply the POINTER's masking strategy on the sub-sequence between every two entity tokens in X. Suppose $(x_i, x_j) \subset X$ is a subsequence spanned by two entity tokens $\{x_l^e = x_i, x_{l+1}^e = x_j\} \in X^e$ where $l \in \{0, \ldots, |X^e| - 1\}$. Masking is applied on this subsequence iteratively until only $\{x_l^e, x_{l+1}^e\}$ are left:

$$\begin{aligned} S = &\{(X^K = X, Y^K), \ldots, \\ &(X^0 = X^e, Y^0)\}. \end{aligned} \tag{2}$$

As shown in Figure 1b, POINTER-E always picks the optimal masking patterns while preserving the entities.

Cold Start Problem While POINTER-E is aware of entities, entities in X^e may appear very close or very far from one another in the full sequence X, i.e., the gap between entities in the sequence X can vary a lot. Consider two sub-sequences $(x_i = x_l^e, x_j = x_{l+1}^e), (x_u = x_w^e, x_v = x_{w+1}^e) \subset X$ where $w, l \in (0, T_e - 1)$ and $w \neq l$. Suppose $j - i \gg v - u$. The tokens between (x_u, x_v) will then be masked out long before tokens in (x_i, x_j) during preprocessing and training. This results in POINTER-E trained to generate a lot of $[NOI]$s in Y^k for small k's. Figure 1b depicts this cold start problem as entity tokens B, D and E are near one another in X. As tokens between them are masked in early stages, the masked sequences in stages 0 and 1, Y^0 and Y^1, contain many $[NOI]$ tokens. POINTER-E trained with such data will therefore lack the ability to generate meaningful tokens in-between these entity tokens. In the worst case, POINTER-E simply generates all $[NOI]$ tokens and ends the generation prematurely which is known as the *cold start problem.*

To better show the problem, we define:

$$NOI\ ratio = \frac{\#[NOI] \text{ in } Y^k}{\#tokens \text{ in } Y^k}. \tag{3}$$

A clear problem of high $NOI\ ratio$ is that Y^k is very similar to Y^{k+1}. When $NOI\ ratio = 1$, the generation will end, In cases where $NOI\ ratio$ is very high for masked sequences in early stages, say Y^0, the trained POINTER-E will more likely infer from X^0 all $[NOI]$'s for $\hat{Y}^0$ and end the generation process. To address this, we need to re-examine the top-down masking stratey used in POINTER and POINTER-E.

2.3 ENCONTER

In this section, we propose ENCONTER which adopts a bottom-up masking strategy to overcome the cold start problem. There are two variants: GREEDY ENCONTER and BBT-ENCONTER.

GREEDY ENCONTER Different from POINTER-E, we now construct training pairs S from X by setting X^0 to be X^e:

$$\begin{aligned} S =\{&(X^0 = X^e, Y^0), (X^1, Y^1), \ldots, \\ &(X^K = X, Y^K)\}, \end{aligned} \tag{4}$$

where Y^k represents the sequence of masked tokens to be inserted into X^k to form X^{k+1}. Similar to POINTER, Y^K contains $[NOI]$'s only. For every two adjacent tokens $\{x_t^k, x_{t+1}^k\} \in X^k$ where $t \in \{0, \ldots, |X^k| - 1\}$, we insert a mask token. Let $\{x_t^k = x_i, x_{t+1}^k = x_j\}$ and (x_i, x_j) be the span of (x_i, x_j) in X. If $i + 1 = j$, the mask token is $[NOI]$. Otherwise, we select a token $x_{t'}$ from (x_i, x_j) with maximum importance score $\alpha_{t'}$ within (x_i, x_j) as the mask token. The sequence Y^k is formed after we go through all the t's. By inserting Y^k into X^k, we obtain the next sequence X^{k+1}. The iterative process stops when all the tokens to be inserted are $[NOI]$s. This method GREEDY ENCONTER greedily selects the token with maximum importance score in the span to be generated in a bottom up insertion (or unmasking) process. By forcing more non-$[NOI]$ tokens to be included in Y^0 and Y^k of small k's, *Greedy Enconter* achieves lower $NOI\ ratio$ in the early stages of inference. Experimentally, we find that the cold start problem is eliminated.

Balanced binary tree ENCONTER (BBT-ENCONTER) To further improve the efficiency of GREEDY ENCONTER, we incorporate balanced binary tree (Stern et al., 2019) into ENCONTER to bias the masking of tokens to be those near the center of the unobserved subsequence of tokens. BBT reward is added to the importance score function as follows. Suppose x_i and x_j are two adjacent tokens in X^k, and (x_i, x_j) represents the corresponding subsequence in X. We define the distance d_p for token $x_p \in (x_i, x_j)$ as:

$$d_p = min(p - i, j - p). \tag{5}$$

We use a softmax function to compute the reward for weighted score based on d_p:

$$w_p = \frac{exp(d_p/\tau)}{\sum_{k=i}^{j} exp(d_k/\tau)}. \tag{6}$$

The weights in the span are then normalized to $[0, 1]$. Then the importance score is defined as:

$$\alpha_p = w_p \cdot (\alpha_p^{TF-IDF} + \alpha_p^{POS} + \alpha_p^{YAKE}). \tag{7}$$

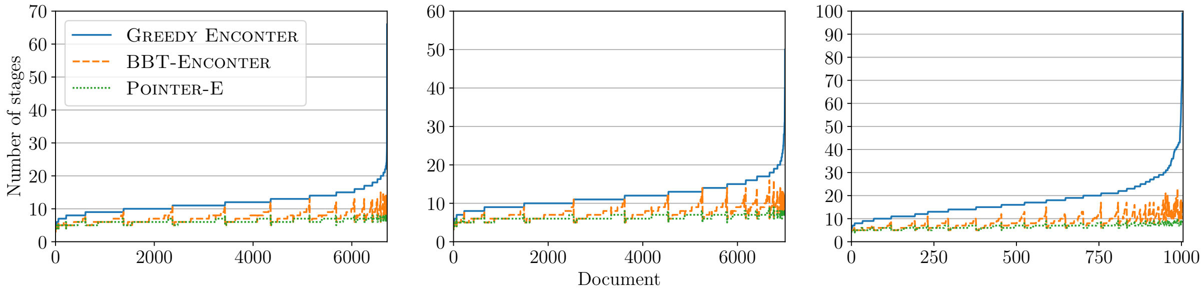
The construction of S is almost the same as GREEDY ENCONTER. The only difference is the new importance score function defined by Eq. 7. This proposed model, known as BBT-ENCONTER, will predict the center and semantically important token in X between two adjacent tokens of X^k.

2.4 Models with Entity Span Aware Inference Option (ESAI)

So far, all above-mentioned models assume that each entity consists of one single token. In real world use cases, an entity may contain more than one token. Without any control during the inference process, it is possible for other tokens to be generated in-between tokens of the same entity. For example in Table 5, "Group Consolidation" may be split into "handling Group s project / Consolidation". To avoid inserting any tokens in between any multi-token entity, we introduce the *entity span aware inference* option to the inference process of POINTER-E and ENCONTER to force the inference of $\hat{Y}^k$ to always generate $[NOI]$ in between the tokens of the multi-token entities. After applying ESAI, the multi-token entities will remain unbroken duing the generation process.

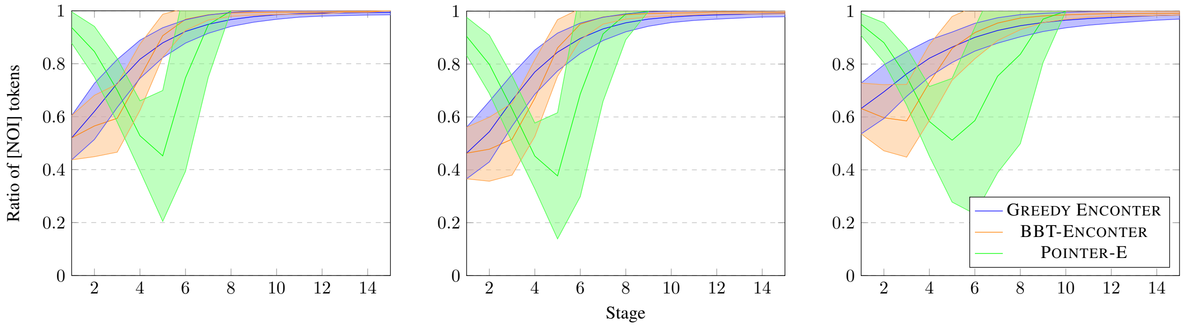
3 Empirical Analysis of POINTER-E and ENCONTER

In this section, we conduct an analysis of the data preprocessing step in POINTER-E, GREEDY ENCONTER and BBT-ENCONTER. Our objective is to empirically evaluate the characteristics of training data generated for the two models. We have left out POINTER as it is inherently not entity-aware and POINTER-E is its entity-aware variant. We first present the two datasets used in this study.



(a) SD (b) SM (c) CoNLL-2003

Figure 2: Number of stages of each document in SD, SM, and CoNLL-2003 (sorted).



(a) SD (b) SM (c) CoNLL-2003

Figure 3: Mean and standard deviation of the ratio of inserted/masked [NOI] tokens in each stage. All x axis are capped to 15 stages. The original maximum number of stages of (a), (b), and (c) are 67, 51, and 100, respectively.

3.1 Datasets

CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003): We select the English version which contains 1,393 news articles labeled with four named entity types: persons, locations, organizations and names of miscellaneous. Training and development sets are used to train the model. Documents having more than 512 tokens by wordpiece tokenizer used in BERT (Devlin et al., 2019) are discarded to ensure that the whole document can fit into the models.

Jobs: This is a job post dataset collected from Singapore's Jobsbank [2]. The dataset consists of 7,474 job posts under the software developer occupation (SD) and 7,768 job posts under the sales and marketing manager occupation (SM). We extract the requirement section of these job posts as the text sequences to be generated. For each requirement text sequence (or document), we use a dictionary of skills to annotate the skill and job related entities in the sequence.

The detailed information of the datasets can be found in Table 1. Table 1 reveals that Pointer-E has much higher $NOI\ ratio$ than Enconter in all the datasets.

	CoNLL	**SM**	**SD**
#training docs	1,004	6,715	7,006
#testing docs	231	754	761
Avg length	220.7	99.4	121.1
Avg entities	24.6	24.4	27.7
#training pairs			
Pointer-E	6,557	43,913	41,343
Greedy Enconter	17,694	83,587	79,467
BBT-Enconter	8,492	52,609	48,625
$NOI\ ratio$ of Y^0			
Pointer-E	0.820	0.904	0.936
Greedy Enconter	0.546	0.463	0.519
BBT-Enconter	0.546	0.463	0.519

Table 1: Summary of the datasets. #training pairs refers to the total number of training pairs derive from each dataset

3.2 Analysis of NOI ratio and Stage Counts

We first analyse the ratio of $[NOI]$ tokens inserted or masked in every stage of the *training data*. Figure 3 shows the mean together with one standard deviation of the Pointer-E, Greedy Enconter and BBT-Enconter for each dataset. X-axis is in log scale. Note we add 1 to the stage number for showing log scale (e.g., the 10^0 in the figure indicates the ratio of $[NOI]$ tokens in Y^0). From Figure 3, we find all datasets share a few similar characteristics, namely: (1) for Pointer-E, the $[NOI]$ ratio is quite high in the first few stages, and drops when the stage is higher. A sudden increase of the ratio to 1 is due to the ending sequence consists all $[NOI]$'s; (2) for Enconter the $[NOI]$

---

[2] https://www.mycareersfuture.sg/

ratio is low in the first few stages, and slowly increase to 1. The result shows ENCONTER can learn to generate balance proportion of $[NOI]$ and non-$[NOI]$ tokens in the first few stages, and also learn not to generate to many non-$[NOI]$ tokens when approaching the end of the generation process.

Figure 2 shows the number of stages each training document requires under different models. The numbers are sorted according to the following priority: GREEDY ENCONTER, POINTER-E, then BBT-ENCONTER. Since BBT-ENCONTER incorporates the binary tree reward scheme, it is able to perform insertion in the middle stages more efficiently compare to GREEDY ENCONTER. This helps to lower the total number of stages required to derive training pairs.

4 Experiment

4.1 Models for Comparison

GPT-2 (Radford et al., 2019) GPT-2 can be used to conduct conditional generation as well (soft-constraints). For a training sequence X together with its entities X^e, we concatenate X^e with X to form a training sequence $\{X^e, X\}$. X^e is then served as a control code sequence to guide GPT-2 in the generation of X. We fine-tune the *GPT-2 small* pretrained by huggingface [3] with 10^{-5} learning rate. Warmup and weight decay are applied. 10 epochs are used for fine-tuning.

POINTER-E, GREEDY ENCONTER, and BBT-ENCONTER: We use BERT (Devlin et al., 2019) as the underlying insertion transformer for all these models similar to that of POINTER. Specifically, we use the *bert-based-cased* pretrained by huggingface. BERT with language model head is fine-tuned on all the training pairs to obtain the models. Learning rate is set to 10^{-5} with warmup and weight decay. 10 epochs are used for fine-tuning.

For POINTER-E, GREEDY ENCONTER, and BBT-ENCONTER, top-k (top-20) sampling method is used to derive $\hat{Y}^k$. For GPT-2, we feed in the $\hat{X}^e$ and let GPT-2 generate the following tokens until reaching the end-of-generation token.

4.2 Evaluation Metrics

We evaluate the models using a few criteria, namely: recall of entities, quality with respect to human crafted text, diversity, fluency, cold start, and generation efficiency. We measure recall of entity constraints by the proportion of entity tokens found in the generated text. Even without ESAI, the recall metric will allow to compare the recall ability of models. Besides recall, we also consider BLEU (Papineni et al., 2002), METEOR (MTR) (Lavie and Agarwal, 2007) and NIST (Doddington, 2002), which are common metrics for evaluating the quality of generated text against human craft text. We compute the BLEU-2 (B-2) and BLEU-4 (B-4) which are n-gram precision-based metrics. For the BLUE based evaluation metric NIST, we compute the NIST-2 (N-2) and NIST-4 (N-4). To measure the diversity of generation, Entropy (Zhang et al., 2018) and Distinction (Li et al., 2016) are used. Entropy-4 (E-4) is defined as the frequency distribution of unique 4-gram terms. Dist-1 (D-1) and Dist-2 (D-2) are used to derive distinct n-grams in the generated text. We also utilize pretrained language model to measure fluency. Perplexity (PPL) is calculated using pretrained GPT-2 (Radford et al., 2019) without fine-tuning. The lower the perplexity is, the more fluent the generation is (based on GPT-2). “AvgLen” is the averaged word counts of the generated sequence. “failure” indicates the proportion of test sequences that fail to be generated at the first step (i.e., $\hat{Y}^0$ are all $[NOI]$’s). Finally, “AvgSteps” shows the average number of steps for the model to complete the generation. Note for GPT-2, the AvgSteps is based on tokens, while the AvgLen is based on words.

4.3 Experiment Results

Tables 2, 3, and 4 show the results of different models on the different datasets. On recall, GPT-2, due to its inability to enforce hard lexical constraints, yields the worst recall. For non-autoregressive models without ESAI, they still achieve high recall. Nevertheless, the high recall of POINTER-E is “contributed by” relatively high failure ratio ("failure") as recall is 1 even when the model fails to generate anything in the first stage. In other words, POINTER-E suffers from cold start problem. GREEDY ENCONTER and BBT-ENCONTER, in contrast, enjoy both good recall and zero failure ratio. With ESAI option, all non-autoregressive models can achieve perfect recall without much additional generation steps. However, this option does not reduce the high failure ratio of POINTER-E. On generation quality compared with human crafted text, GREEDY ENCONTER and BBT-ENCONTER outperform all other models by NIST, BLEU, and

---

[3] https://huggingface.co/

Method	Recall	NIST N-2	NIST N-4	BLEU B-2	BLEU B-4	MTR	Entropy E-4	DIST D-1	DIST D-2	PPL	AvgLen	failure	AvgSteps
Baselines													
GPT-2	0.70	1.38	1.39	0.13	0.08	0.19	4.91	0.16	0.57	**35.7**	201.4	**0.00**	256.84
POINTER-E	0.98	0.72	0.72	0.08	0.04	0.19	3.63	0.22	0.65	285.7	88.9	0.35	4.18
POINTER-E (+ESAI)	**1.00**	0.63	0.64	0.08	0.04	0.18	3.54	**0.23**	**0.67**	337.1	81.4	0.34	3.84
ENCONTER													
Greedy	0.96	1.95	1.96	0.19	0.09	**0.25**	**4.99**	0.16	0.58	112.1	192.5	**0.00**	17.56
Greedy (+ESAI)	**1.00**	**1.99**	**2.00**	**0.20**	**0.10**	**0.25**	4.95	0.16	0.59	111.7	181.9	**0.00**	16.48
BBT	0.94	1.83	1.84	0.19	**0.10**	0.23	4.87	0.18	0.62	154.9	161.1	**0.00**	8.19
BBT (+ESAI)	**1.00**	1.87	1.87	**0.20**	**0.10**	0.24	4.84	0.18	0.62	150.0	156.7	**0.00**	8.26
Human	-	-	-	-	-	-	4.99	0.20	0.66	53.0	202.1	-	-

Table 2: CoNLL-2003 result

Method	Recall	NIST N-2	NIST N-4	BLEU B-2	BLEU B-4	MTR	Entropy E-4	DIST D-1	DIST D-2	PPL	AvgLen	failure	AvgSteps
Baselines													
GPT-2	0.63	1.42	1.42	0.13	0.09	0.20	4.63	0.05	0.30	**66.3**	123.2	**0.00**	168.44
POINTER-E	0.99	1.69	1.70	0.20	0.12	0.28	3.84	**0.08**	**0.46**	901.2	73.4	0.38	5.01
POINTER-E (+ESAI)	**1.00**	1.71	1.72	0.21	0.12	0.28	3.85	**0.08**	**0.46**	966.4	74.1	0.34	4.91
ENCONTER													
Greedy	0.99	3.31	3.33	0.40	0.28	0.41	4.53	0.07	0.37	124.3	111.1	**0.00**	9.80
Greedy (+ESAI)	**1.00**	3.31	3.33	0.40	0.28	0.40	4.52	0.07	0.37	125.2	109.5	**0.00**	9.79
BBT	0.99	**3.59**	**3.62**	**0.45**	**0.34**	**0.44**	4.53	0.07	0.38	135.6	110.6	**0.00**	5.86
BBT (+ESAI)	**1.00**	3.55	3.57	0.44	**0.34**	**0.44**	**4.54**	0.07	0.38	137.7	111.5	**0.00**	5.92
Human	-	-	-	-	-	-	4.66	0.08	0.41	90.7	125.1	-	-

Table 3: SD result

Method	Recall	NIST N-2	NIST N-4	BLEU B-2	BLEU B-4	MTR	Entropy E-4	DIST D-1	DIST D-2	PPL	AvgLen	failure	AvgSteps
Baselines													
GPT-2	0.72	1.50	1.51	0.15	0.10	0.23	**4.40**	0.05	0.32	**101.0**	96.4	**0.00**	127.48
POINTER-E	0.98	1.32	1.32	0.17	0.10	0.26	3.46	**0.09**	**0.48**	2447.7	52.7	0.34	4.89
POINTER-E (+ESAI)	**1.00**	1.26	1.26	0.16	0.09	0.25	3.42	**0.09**	**0.48**	2535.7	53.3	0.38	5.07
ENCONTER													
Greedy	0.99	2.48	2.49	0.31	0.20	0.36	4.21	0.07	0.40	153.9	82.2	**0.00**	9.75
Greedy (+ESAI)	**1.00**	2.44	2.45	0.31	0.20	0.36	4.19	0.07	0.40	147.4	80.2	**0.00**	9.62
BBT	0.98	**2.73**	**2.74**	**0.34**	**0.24**	**0.38**	4.26	0.07	0.41	161.1	83.8	**0.00**	6.04
BBT (+ESAI)	**1.00**	2.69	2.70	**0.34**	0.23	**0.38**	4.25	0.07	0.41	157.5	83.6	**0.00**	6.05
Human	-	-	-	-	-	-	4.45	0.08	0.43	104.3	101.6	-	-

Table 4: SM result

MTR. This suggests that ENCONTER models learn the context of entities better compared to other models. On generation diversity, POINTER-E again has the highest diversity largely due to its high failure ratio. Finally, we discuss the efficiency of models measured by AvgSteps. The autoregressive nature of GPT-2 makes it the least efficient model among all. POINTER-E's ability to optimize masking patterns makes it the most efficient model. With balance binary tree reward, BBT-ENCONTER is able to finish its generation in fewer iterations than GREEDY ENCONTER.

Case example Table 5 shows a case example from Jobs SM dataset. The entities of the given constraint are underlined. Invalid entities generated are colored in red, while the remaining ones are colored in blue. There are three types of invalid cases. First, the case of entity is not the same as specified. Second, the entity is not recalled in the generation. Third, the entity has its tokens separated by some other token(s). In this example, POINTER-E and POINTER-E ESAI terminate their generations prematurely. They fail to perform generation at the very first stage.

5 Related Work

Recent years have witnessed significant success using autoregressive (Dai and Le, 2015; Peters et al., 2018; Radford, 2018) generative models to conduct conditional generation on various tasks. CTRL (Keskar et al., 2019) uses control codes trained together with large amount of data to control the content to be generated. RecipeGPT (H. Lee et al., 2020) takes ingredients as a series of control and trains the generation of recipe text. PPLM (Dathathri et al., 2020)

Greedy Enconter:
Degree / ACCA / CIMA / CA / CFA / F & B / Group Consolidation experience
Degree in General Accounting
Good track record in IFRSRSM, Fixed Assets, IFRS

Greedy Enconter ESAI:
* * Degree / ACCA / CIMA / CA / CFA / CA / CAPA / Singapore Group Consolidation / Management / General Accounting experience
* Experience in IFRS or preferred
* Experience in Fixed Assets Management
* Extensive experience in IFRS

BBT-Enconter:
Degree or ACCA / CIMA or CFA qualifications
YEARSPAN'experience in handling Group s project / Consolidation / Good Inteconor / General Accounting
Knowledge of IFRSPAN, Fixed Assets (IFRS, etc)

BBT-Enconter ESAI:
Minimum Degree / ACCA / CIMA, CFA or equivalent
Minimum of YEARSPAN of experience in Marketing and Group Consolidation and General Accounting
Knowledge of IFRS) and Fixed Assets (IFRS)

GPT-2: (missing: IFRS, CFA, Group Consolidation, Fixed Assets)
Job Requirements :
- Degree in General Accounting / ACCA Qualification
- At least YEARSPAN of applicable working experience in similar capacity
- Must be able to multi-task and handle different priorities simultaneously
- General accounting knowledge will be advantageous

Interested applicant, kindly send in your CPA or CIMA reference number to EMAIL
EA Licence number : LICENSENUM
Registration number : REGNUM

Human:
Professional Qualifications : Bachelors Degree
Qualified with a professional financial body (ICAEW / ICPA / ACCA / CIMA / CFA etc)
Specialist Knowledge / Skills : Group Consolidation General Accounting IFRS Fixed Assets Industry Experience
Experience : YEARSPAN post qualified with extensive IFRS experience and industry experience

Table 5: A generated example from SM dataset. Pointer-E and Pointer-E ESAI are not shown since they failed to generate at first step.

directly steers the pretrained language by a bag-of-words model or simple linear discriminator. The above models in their own ways gain certain level of control over the content generation process. However, they do not provide a mechanism to directly enforce some lexical constraints in the final generation. Non-monotonic sequence generation (Welleck et al., 2019) is designed to perform hard lexical constrains generation based on binary tree structure. By leveraging level-order and in-order traversal of binary tree, the model allows text to be generated non-monotonically. Although the results from non-monotonic generation models seem promising, they do not perform token generation in parallel and the tree structure governing the generation process may produce many unused tokens during the generation.

The emergence of non-autoregressive language model provides another approach to support hard lexical constraints. Insertion transformer (Stern et al., 2019) uses transformer architecture with balanced binary tree loss to perform insertion-based generation. KERMIT (Chan et al., 2019) is proposed as a structure to unify insertion transformers. Levenshtein transformer (Gu et al., 2019) further introduces deletion as an action to take during generation. Our Enconter models differ from these previous models as they are not designed to support any lexical constraints, including entity constrains.

6 Conclusions

Constrained text generation is an important task for many real world applications. In this paper, we focus on hard entity constraints and the challenges associated with enforcing them in text generation. Our analysis of the state-of-the-art insertion transformers reveals issues, namely, cold start problems and inefficient generation. We therefore propose two insertion transformer models, Greedy Enconter and BBT Enconter, that use a bottom-up preprocessing strategy to prepare training data so as to eliminate the cold start problem caused by top-down preprocessing strategy. BBT Enconter further incorporates a balanced tree reward scheme to make the generation process more efficient. Through experiments on real world datasets, we show that the two models outperform the strong baselines, Pointer-E and GPT2, in recall, quality and failure rate while not compromising much generation efficiency. For future research, it will be interesting to consider more diverse constraints (e.g., soft constraint, rules, etc.) and user interaction in the generation process to expand the scope of applications that can benefit from this research.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. Any opinions, findings conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. Kermit: Generative insertion-based modeling for sequences. *arXiv preprint arXiv:1906.01604*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

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

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

David Gries. 1982. A note on a standard strategy for developing loop invariants and loops. *Science of Computer Programming*, 2(3):207–214.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, pages 11181–11191.

Helena H. Lee, Ke Shu, Palakorn Achananuparp, Philips Kokoh Prasetyo, Yue Liu, Ee-Peng Lim, and Lav R Varshney. 2020. Recipegpt: Generative pretraining based cooking recipe generation and evaluation system. In *Companion Proceedings of the Web Conference 2020*, pages 181–184.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

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

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324.

Lianhui Qin, Michel Galley, Chris Brockett, Xiaodong Liu, Xiang Gao, Bill Dolan, Yejin Choi, and Jianfeng Gao. 2019. Conversing by reading: Contentful neural conversation with on-demand machine reading. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5427–5436.

A. Radford. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Sean Welleck, Kianté Brantley, Hal Daumé, and Kyunghyun Cho. 2019. Non-monotonic sequential text generation. In *36th International Conference on Machine Learning, ICML 2019*, pages 11656–11676. International Machine Learning Society (IMLS).

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. Generating informative and diverse conversational responses via adversarial information maximization. In *Advances in Neural Information Processing Systems*, pages 1810–1820.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020. Pointer: Constrained text generation via insertion-based generative pre-training. *arXiv preprint arXiv:2005.00558*.

Andrea Zugarini, Stefano Melacci, and Marco Maggini. 2019. Neural poetry: Learning to generate poems using syllables. In *International Conference on Artificial Neural Networks*, pages 313–325. Springer.

Meta-Learning for Effective Multi-task and Multilingual Modelling

Ishan Tarunesh[1] **Sushil Khyalia**[1] **Vishwajeet Kumar**[2]
Ganesh Ramakrishnan[1] **Preethi Jyothi**[1]
[1] Indian Institute of Technology Bombay
[2] IBM India Research Lab
`{ishan, sushil, ganesh, pjyothi}@cse.iitb.ac.in`
`vishk024@in.ibm.com`

Abstract

Natural language processing (NLP) tasks (e.g. question-answering in English) benefit from knowledge of other tasks (*e.g.*, named entity recognition in English) and knowledge of other languages (*e.g.*, question-answering in Spanish). Such shared representations are typically learned in isolation, either across tasks or across languages. In this work, we propose a meta-learning approach to learn the interactions between both tasks and languages. We also investigate the role of different sampling strategies used during meta-learning. We present experiments on five different tasks and six different languages from the XTREME multilingual benchmark dataset (Hu et al., 2020). Our meta-learned model clearly improves in performance compared to competitive baseline models that also include multi-task baselines. We also present zero-shot evaluations on unseen target languages to demonstrate the utility of our proposed model.

1 Introduction

Multi-task and multilingual learning are both problems of long standing interest in natural language processing. Leveraging data from multiple tasks and/or additional languages to benefit a target task is of great appeal, especially when the target task has limited resources. When it comes to multiple tasks, it is well-known from prior work on multi-task learning (Liu et al., 2019b; Kendall et al., 2018; Liu et al., 2019a; Yang and Hospedales, 2017) that jointly learning a model across tasks can benefit the tasks mutually. For multiple languages, the ability of deep learning models to learn effective embeddings has led to their use for joint learning of models across languages (Conneau et al., 2020; Conneau and Lample, 2019; Artetxe and Schwenk, 2019); learning cross-lingual embeddings to aid languages in limited resource settings is of growing interest (Kumar et al., 2019; Wang et al., 2017; Adams et al., 2017). Let us say we had access to M tasks across N different languages - *c.f.* Table 1 that outlines such a matrix of tasks and languages from the XTREME benchmark (Hu et al., 2020). How do we perform effective joint learning across tasks and languages? Are there specific tasks or languages that need to be sampled more frequently for effective joint training? Can such sampling strategies be learned from the data?

In this work, we adopt a meta-learning approach for efficiently learning parameters in a shared parameter space across multiple tasks and multiple languages. Our chosen tasks are question answering (QA), natural language inference (NLI), paraphrase identification (PA), part-of-speech tagging (POS) and named entity recognition (NER). The tasks were chosen to enable us to employ a gamut of different types of language representations needed to tackle problems in NLP. In Figure 1, we illustrate the different types of representations by drawing inspiration from the Vauquois Triangle (Vauquois, 1968), well-known for machine translation, and situating our chosen tasks within such a triangle. Here we see that POS and NER are relatively 'shallower' analysis tasks that are token-centric, while QA, NLI and PA are 'deeper' analysis tasks that would require deeper semantic representations. This representation suggests a strategy for effective parameter sharing. For the deeper tasks, the same task in different languages could have representations that are closer and hence benefit each other, while for the shallower tasks, keeping the language unchanged and exploring different tasks might be more beneficial. Interestingly, this is exactly what we find with our meta-learned model and is borne out in our experimental results. We also find that as the model progressively learns, the meta-learning based models for the tasks requiring deeper semantic analysis benefit more from joint learning compared to the shallower tasks.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3600–3612
April 19 - 23, 2021.

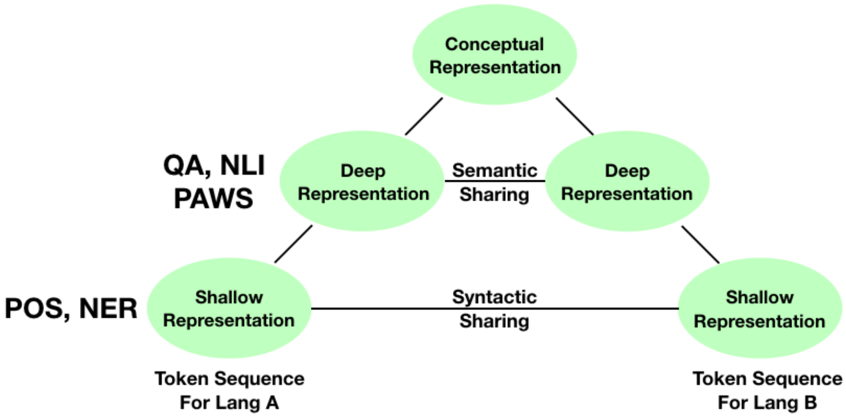


Figure 1: Illustration derived from Vauquois Triangle to linguistically motivate our setting. POS and NER being lower down in the representations (and are thus 'shallower') are further away from the same task in another language. QA, XNLI and PAWS being higher up in the representations (and are thus 'deeper') are closer to the same task in another language.

With access to multiple tasks and languages during training, the question of how to sample effectively from different tasks and languages also becomes important to consider. We investigate different sampling strategies, including a parameterized sampling strategy, to assess the influence of sampling across tasks and languages on our meta-learned model.

Our main contributions in this work are threefold:

- We present a meta-learning approach that enables effective sharing of parameters across multiple tasks and multiple languages. This is the first work, to our knowledge, to explore the interplay between multiple tasks at different levels of abstraction and multiple languages using meta-learning. We show results on the recently-released XTREME benchmark and observe consistent improvements across different tasks and languages using our model. We also offer rules of thumb for effective meta-learning that could hold in larger settings involving additional tasks and languages.

- We investigate different sampling strategies that can be incorporated within our meta-learning approach and examine their benefits.

- We evaluate our meta-learned model in zero-shot settings for every task on target languages that never appear during training and show its superiority compared to competitive zero-shot baselines.

2 Related Work

We summarize three threads of related research that look at the transferability in models across different tasks and different languages: multi-task learning, meta-learning and data sampling strategies for both multi-task learning and meta-learning. Multi-task learning (Caruana, 1993) has proven to be highly effective for transfer learning in a variety of NLP applications such as question answering, neural machine translation, *etc.* (McCann et al., 2018; Hashimoto et al., 2017; Chen et al., 2018; Kiperwasser and Ballesteros, 2018). Some multi-task learning approaches (Jawanpuria et al., 2015) have attempted to identify clusters (or groups) of related tasks based on end-to-end convex optimization formulations. Meta-learning algorithms (Nichol et al., 2018) are highly effective for fast adaptation and have recently been shown to be beneficial for several machine learning tasks (Santoro et al., 2016; Finn et al., 2017). Gu et al. (2018) use a meta-learning algorithm for machine translation to leverage information from high-resource languages. Dou et al. (2019) investigate multiple model agnostic meta-learning algorithms for low-resource natural language understanding on the GLUE (Wang et al., 2018) benchmark.

Data sampling strategies for multi-task learning and meta-learning form the third thread of related work. A good sampling strategy has to account for the imbalance in dataset sizes across tasks/languages and the similarity between tasks/languages. A simple heuristic-based solution to address the issue of data imbalance is to assign more weight to low-resource tasks or languages (Aharoni et al., 2019). Arivazhagan et al. (2019) define a temperature parameter which controls how often one samples from low-resource tasks/languages. The MultiDDS algorithm, proposed by Wang et al. (2020b), actively learns a different set of parameters for sampling batches given a set of tasks such that the performance on a held-out set is maximized. We use a variant of MultiDDS as a sampling strategy in our meta-learned model. Nooralahzadeh et al. (2020) is most similar in spirit to our work in that they study a cross-lingual and cross-task meta-learning architecture but only focus on zero-shot and few-shot transfer for two natural language understanding tasks, NLI and QA. In contrast, we study many tasks in many languages, in conjunction with sampling strategies, and offer concrete insights on how best to guide the meta-learning process when multiple tasks are in the picture.

3 Methodology

Our setting is pivoted on a **grid** of tasks and languages (with some missing entries as shown in Table 1). Each row of the grid corresponds to a single **task**. A cell of the grid corresponds to a Task-Language pair which we refer to as a TL pair (**TLP**). We denote by $q_i = |\mathcal{D}^i_{train}| / \left(\sum_{k=1}^{n} |\mathcal{D}^k_{train}|\right)$, the fraction of the dataset size for the i^{th} TLP and by $P_{\mathcal{D}}(i)$, the probability of sampling a batch from the i^{th} TLP during meta training. The distribution over all TLPs, *viz.*, is a Multinomial (say $\mathcal{M}$) over $P_{\mathcal{D}}(i)$s.

3.1 Our Meta-learning Approach

The goal in the standard meta learning setting is to obtain a model that generalizes well to new test/target tasks given some distribution over training tasks. This can be achieved using optimization-based meta-learning algorithms that modify the learning procedure in order to learn a good initialization of the parameters. This can serve as a useful starting point that can be further fine-tuned on various tasks. Finn et al. (2017) proposed a general optimization algorithm called Model Agnostic Meta Learning (MAML) that can be trained using gradient descent. MAML aims to minimize the following objective

$$\min_{\theta} \sum_{T_i \sim \mathcal{M}} \mathcal{L}_i \left(U_i^k(\theta) \right) \tag{1}$$

where $\mathcal{M}$ is the Multinomial distribution over TLPs, $\mathcal{L}_i$ is the loss and U_i^k a function that returns θ after k gradient updates both calculated on batches sampled from T_i. Minimizing this objective using first order methods involves computing gradients of the form $\frac{\partial}{\partial\theta} U_i^k(\theta)$, leading to the expensive computation of second order derivatives. Nichol et al. (2018) proposed an alternative first-order meta-learning algorithm named "Reptile" with simple update rule:

$$\theta \leftarrow \theta + \beta \frac{1}{|\{T_i\}|} \sum_{T_i \sim \mathcal{M}} (\theta_i^{(k)} - \theta) \tag{2}$$

where $\theta_i^{(k)}$ is $U_i^k(\theta)$. Despite its simplicity, a recent study by Dou et al. (2019) showed that Reptile is atleast as effective as MAML in terms of performance. We therefore employed Reptile for meta learning in all our experiments.

Algorithm 1 Our Meta-learning Approach

```
Input: $\mathcal{D}_{train}$ set of TLPs for meta training
(Also $\mathcal{D}_{dev}$ for parametrised sampling)
        Sampling Strategy (Temperature / MultiDDS)
Output: The converged multi-task multilingual model parameters $\theta^*$
Initialize $P_D(i)$ depending on the sampling strategy
while not converged do
    ▷ Perform Reptile Updates
    Sample $m$ TLPs $T_1, T_2, \ldots, T_m$ from $\mathcal{M}$
    for i = 1,2,...,m do
        $\theta_i^{(k)} \leftarrow U_i^k(\theta)$, denoting $k$ gradient updates from $\theta$ on batches of TLP $T_i$
    end for
    $\theta \leftarrow \theta + \frac{\beta}{m} \sum_{i=1}^{m} (\theta_i^{(k)} - \theta)$
    if Sampling Strategy ← MultiDDS then
        for $\mathcal{D}^i_{train} \in \mathcal{D}_{train}$ do
            $R(i;\theta) \leftarrow cos(g_{dev}, g_{train})$, $g_{dev}$ is gradient on $\{\mathcal{D}_{dev}\}$ and $g_{train}$ is gradient on $\mathcal{D}^i_{train}$
        end for
        ▷ Update Sampling Probabilities
        $d_\psi \leftarrow \sum_{i=1}^{n} R(i;\theta) \cdot \nabla_\psi log(P_{\mathcal{D}}(i;\psi))$
        $\psi \leftarrow$ GradientUpdate$(\psi, d_\psi)$
    end if
end while
```

3.2 Selection and Sampling Strategies

3.2.1 Selection

The choice of TLPs in meta-learning plays a vital role in influencing the model performance, as we will see in more detail in Section 5. Apart from the use of all TLPs across both tasks and languages during training, selecting all languages for a given task (Gu et al., 2018) and selecting all tasks for a given language (Dou et al., 2019) are two other logical choices. We refer to the last two settings as being `Task-Limited` and `Lang-Limited`, respectively.

3.2.2 Heuristic Sampling

Once the TLPs for meta training (denoted by $\mathcal{D}$) have been selected, we need to sample TLPs from $\mathcal{M}$. We investigate temperature-based heuristic sampling (Arivazhagan et al., 2019) which defines the probability of any dataset as a function of its size. $P_{\mathcal{D}}(i) = q_i^{1/\tau} / \left(\sum_{k=1}^{n} q_k^{1/\tau} \right)$ where $P_{\mathcal{D}}(i)$ is the probability of the i^{th} TLP to be sampled and τ is the temperature parameter. $\tau = 1$ reduces to sampling TLPs proportional to their dataset sizes and $\tau \rightarrow \infty$ reduces to sampling TLPs uniformly.

3.2.3 Parameterized Sampling

The sampling strategy defined in Section 3.2.2 remains constant throughout meta training and only depends on dataset sizes. Wang et al. (2020b) proposed a parameterized sampling technique called MultiDDS that builds on Differential Data Selection (DDS) (Wang et al., 2020a) for weighing multiple datasets. The $P_{\mathcal{D}}(i)$ are parameterized using ψ_i as $P_{\mathcal{D}}(i) = e^{\psi_i} / \sum_j e^{\psi_j}$ with the initial value of ψ satisfying $P_{\mathcal{D}}(i) = q_i$. The optimization for ψ and θ is performed in an alternating manner (Colson et al., 2007)

$$\psi^* = \underset{\psi}{\text{argmin}} \; J(\theta^*(\psi), \mathcal{D}_{dev}) \tag{3}$$

$$\theta^*(\psi) = \underset{\theta}{\text{argmin}} \; E_{x,y \sim P(T;\psi)}[l(x,y;\theta)] \tag{4}$$

$J(\theta, \mathcal{D}_{dev})$ is the objective function which we want to minimize over development set(s). The reward function, $R(x,y;\theta_t)$, is defined as:

$$R(x,y;\theta_t) \approx \underbrace{\nabla J(\theta_t,\mathcal{D}_{dev})^T}_{g_{dev}} \cdot \underbrace{\nabla_\theta l(x,y;\theta_{t-1})}_{g_{train}} \tag{5}$$

$$\approx cos(g_{dev}, g_{train}) \tag{6}$$

ψ's are updated using the REINFORCE (Williams, 1992) algorithm.

$$\psi_{t+1} \leftarrow \psi_t + R(x,y;\theta_t) \cdot \nabla_\psi log(P(x,y;\psi)) \tag{7}$$

The Reptile meta-learning algorithm (along with details of the parameterized sampling strategy) is outlined in Algorithm 1.

4 Experimental Setup

4.1 Evaluation Benchmark

The recently released XTREME dataset (Hu et al., 2020) is a multilingual multi-task benchmark consisting of classification, structured prediction, QA and retrieval tasks. Each constituent task has associated datasets in multiple languages. The sources of POS and NER datasets are Universal Dependency v2.5 treebank (Nivre et al., 2020) and WikiAnn (Pan et al., 2017) respectively, with ground-truth labels available for each language. Large-scale datasets for QA, NLI and PA were originally available only for English. The PAWS-X (Yang et al., 2019) dataset contains machine-translated training pairs and human-translated evaluation pairs for PA. The authors of XTREME train a custom-built translation system to obtain translated datasets for QA and NLI. For the NLI task, we train using MultiNLI (Williams et al., 2018) and evaluate on XNLI (Conneau et al., 2018). For the QA task, SQuAD 1.1 (Rajpurkar et al., 2016) was used for training and MLQA (Lewis et al., 2019) for evaluation.

Regarding evaluation metrics, for QA we report F1 scores and for the other four tasks (PA, NLI, POS, NER) we report accuracy scores.

4.2 Implementation Details

BERT (Devlin et al., 2019) models yield state-of-the-art performance for many NLP tasks. Since we are dealing with datasets in multiple languages, we build our meta learning models on mBERT (Pires et al., 2019; Wu and Dredze, 2019) base architecture, implemented by Wolf et al. (2020), with output layers specific to each task. In our experiments, we use the AdamW (Loshchilov and Hutter, 2017) optimizer to make gradient-based updates to the model's parameters using batches from a particular TLP (Alg. 1, Line 6). This optimizer is shared across all the TLPs. When performing the meta-step (Alg. 1, Line 8), we use vanilla stochastic gradient descent (SGD) (Robbins and Monro, 1951) updates. Similarly, in the case of parameterized sampling the weights are updated (Alg. 1, Line 15) using vanilla SGD.

Meta training involves sampling a set of m tasks, taking k gradient update steps from the initial parameter to arrive at $\theta_i^{(k)}$ for task T_i and finally updating θ using the Reptile update rule. For meta-

Task	en	hi	es	de	fr	zh
Natural Language Inference (NLI)	392K	🟥	392K	392K	392K	🟥
Question Answering (QA)	88.0K	82.4K	81.8K	80.0K	🟥	🟥
Part Of Speech (POS)	21.2K	13.3K	28.4K	166K	🟥	7.9K
Named Entity Recognition (NER)	20K	5K	20K	20K	20K	20K
Paraphrase Identification (PA)	49.4K	🟥	49.4K	49.4K	49.4K	49.4K

Table 1: Dataset matrix showing datasets that are available (green) from the XTREME Benchmark. The number of training instances are also mentioned for each available dataset.

models we fix learning rate = 3e-5 and dropout probability = 0.1 (provided by XTREME for reproduction of baselines). Grid search was performed on $m \in \{4, 8, 16\}$, $k \in \{2, 3, 4, 5\}$ and $\beta \in \{0.1, 0.5, 1.0\}$ for `All TLPs` model ($\tau = 1$). The best setting ($m = 8$, $k = 3$, $\beta = 1.0$) was selected based on validation score (accuracy or F1) averaged over all TLPs. These hyperparameters were kept constant for all further experiments. Each meta-learning model is trained for 5 epochs. We then finetune the meta model individually on each TLP and evaluate the results. Finetuning parameters vary for different task and are mentioned in Appendix B.

4.3 Data Selection and Sampling Strategies

We experiment with three different configurations for the set of TLPs to be considered during meta-learning: (a) using all tasks for a given language (`Lang-Limited`) (b) using all languages for a given task (`Task-Limited`) and (c) using all tasks and all languages (`All TLPs`). Since the dataset size varies across tasks (as also across languages), we use temperature sampling within each setting for $\tau = 1, 2, 5$ and ∞. (In Table 4 of the Appendix C in the supplementary material, we report results for different choices of TLP selection and different values of the temperature.)

With respect to the **Input** in Algorithm 1, there are two sets of TLPs that need to be selected for parameterized sampling: $\mathcal{D}_{train}$ and $\mathcal{D}_{dev}$. In order to analyse the effect of the choice of task and language, we experiment with the following 4 settings -

(a) $\mathcal{D}_{train}$ = `Lang-Limited`, $\mathcal{D}_{dev}$ = `Target TLP`

(b) $\mathcal{D}_{train}$ = `Task-Limited`, $\mathcal{D}_{dev}$ = `Target TLP`

(c) $\mathcal{D}_{train}$ = `All TLPs`, $\mathcal{D}_{dev}$ = `Lang-Limited`

(d) $\mathcal{D}_{train}$ = `All TLPs`, $\mathcal{D}_{dev}$ = `Task-Limited`.

The models (a), (b) are referred to as `mDDS` and (c), (d) are called `mDDS-Lang` and `mDDS-Task` respectively. Results for these 4 models are reported in Table 2 alongside temperature sampling for comparison.

4.4 Baselines

Our first baseline system for each TLP uses mBERT-based models trained on data specific to each TLP, which is either available as ground-truth or in a translated form. We follow the same hyperparameter settings as reported in XTREME. We also present three multi-task learning (MTL) baseline systems: task limited (`Task-Limited`), language limited (`Lang-Limited`), and the use of all TLPs during training (`All TLPs MTL`). During MTL training, we concatenate and shuffle the selected datasets. The model is trained for 5 epochs with a learning rate of 5e-5. We refer the reader to Appendix A for more training details.

5 Results and Analysis

Table 2 presents all our main results comparing different data selection and sampling strategies used for meta-learning. Each column corresponds to a target TLP; the best-performing meta-learned models for each target TLP within each data selection setting have been highlighted in colour. (Light-to-dark gradation reflects improvements in performance.) From Table 2, we see that our meta-learned models outperform the baseline systems across all the TLPs corresponding to QA, NLI and PA. (POS and NER also mostly benefit from meta-learning, but the margins of improvement are much smaller compared to the other tasks given the already high baseline scores).

`Task-Limited` vs `Lang-Limited` models. For QA and NLI, we observe that the `Task-Limited` models are *always better* than

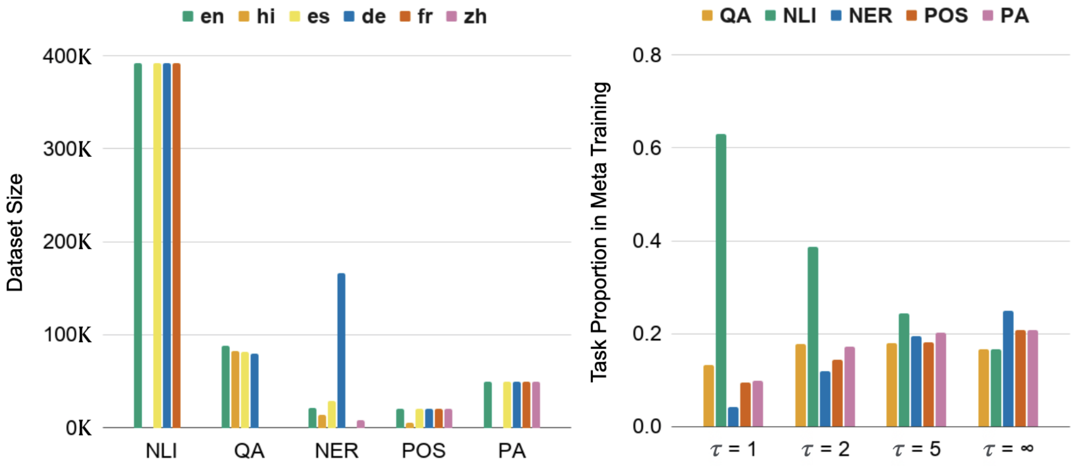


Figure 2: (a) Size of train dataset by language for each task (b) Proportion of dataset in meta training for different value of τ.

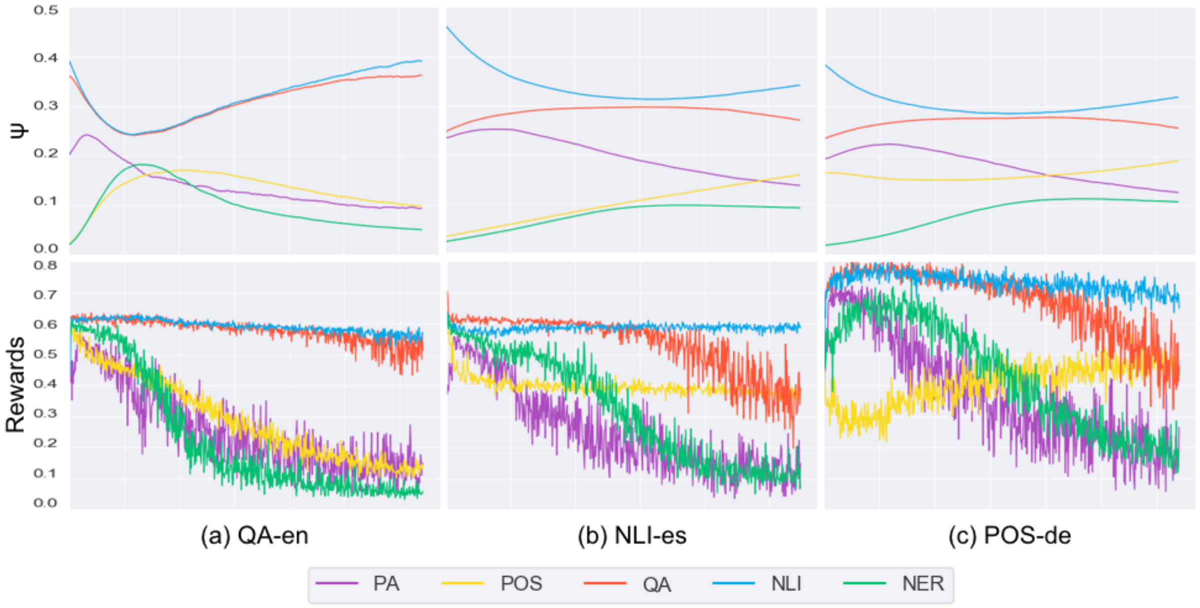
Model	SS	QA (F1)				NLI (Acc.)				PA (Acc.)				
		en	hi	es	de	en	es	de	fr	en	es	de	fr	zh
`Baselines`		79.94	59.94	65.83	63.17	81.39	78.37	76.82	77.30	92.35	89.75	87.45	89.61	83.32
`Lang-Limited MTL`		69.80	53.24	62.29	58.91	80.49	76.10	75.18	74.94	93.75	87.75	85.35	88.55	80.49
`Task-Limited MTL`		74.04	57.77	64.28	61.47	80.95	78.15	75.90	77.14	93.65	86.65	86.25	86.82	81.24
`All TLPs MTL`		63.22	42.94	54.05	51.61	80.05	76.48	74.86	76.18	93.50	90.30	88.45	89.71	82.66
`Lang-Limited`	`Temp`	-0.04	-0.24	-0.27	+0.07	+0.06	+0.39	+0.03	-0.70	+0.45	+0.05	+0.35	+0.40	-0.06
	`mDDS`	+0.07	-0.12	+0.06	+0.14	+0.02	-0.61	-0.80	-0.60	-0.25	-0.05	0.00	-0.30	-1.41
`Task-Limited`	`Temp`	+0.55	+0.43	+0.50	+0.40	+1.65	+1.12	+1.25	+0.79	+0.20	-0.15	-0.55	+0.85	-0.15
	`mDDS`	+0.21	+0.62	-0.67	+1.06	+1.32	+1.10	+1.39	+0.48	+0.50	-0.65	-0.35	+1.45	+1.06
`All TLPs`	`Temp`	+0.53	+0.47	+0.32	+0.47	+1.90	+1.22	+1.45	+0.95	+0.35	+0.45	+1.20	+1.05	+0.85
	`mDDS-Lang`	+0.08	+0.50	-1.57	+0.08	+0.76	+0.26	-0.10	+0.32	+0.25	+0.85	+0.75	+0.75	+1.11
	`mDDS-Task`	+0.18	+0.60	+0.11	+0.54	+1.50	+0.90	+0.72	+0.72	+0.10	+0.80	+1.27	+1.10	+1.16

Model	SS	NER (Acc.)						POS (Acc.)				
		en	hi	es	de	fr	zh	en	hi	es	de	zh
`Baselines`		93.23	95.72	95.84	97.32	95.48	94.34	96.15	93.57	96.02	97.37	92.60
`Lang-Limited MTL`		92.54	92.67	95.14	96.40	94.38	92.97	95.08	92.43	95.19	97.19	89.71
`Task-Limited MTL`		93.51	93.94	95.77	97.09	95.27	93.72	95.70	93.34	95.73	97.35	92.52
`All TLPs MTL`		92.28	91.95	94.90	96.18	94.38	92.53	94.70	91.89	95.10	97.03	89.92
`Lang-Limited`	`Temp`	+0.60	+0.06	+0.09	+0.24	-0.09	-0.47	-0.06	-0.01	+0.10	+0.04	-0.17
	`mDDS`	-0.21	-0.85	-0.20	-0.10	-0.57	-0.55	-0.27	-0.02	-0.19	-0.06	-0.37
`Task-Limited`	`Temp`	+0.79	-0.46	0.00	-0.07	-0.18	-0.51	-0.22	-0.05	-0.21	+0.02	-0.09
	`mDDS`	-0.10	-1.61	0.00	-0.16	-0.33	-0.69	-0.38	-0.02	-0.22	+0.05	-0.12
`All TLPs`	`Temp`	-0.15	-0.70	+0.13	0.00	-0.16	-0.39	-0.22	-0.09	-0.21	+0.03	-0.16
	`mDDS-Lang`	-0.16	-0.09	+0.11	-0.08	-0.14	-0.65	-0.21	-0.10	-0.11	+0.03	-0.17
	`mDDS-Task`	-0.27	-0.42	+0.08	-0.14	-0.07	-0.58	-0.22	-0.14	-0.19	+0.02	-0.09

Table 2: Main results comparing different data selection and sampling strategies. Sampling strategy, SS=`Temp` refers to the temperature-based sampling strategy and SS=`mDDS` refers to the multiDDS-based sampling strategy. `mDDS-Task` and `mDDS-Lang` refer to the use of a development set for multiDDS that contains all languages for a task and all tasks for a language, respectively. The best result among Baseline and three MTL models is highlighted using orange. For each column we present the difference (positive or negative) of the meta models from the best baseline (highlighted in orange) of that column

the `Lang-Limited` models. This is in line with our intuition that tasks like QA and NLI (which require deeper semantic representations) will benefit more by using data from different languages for the same task. We see the opposite seems to hold for POS and NER where the `Lang-Limited` models are *almost always better* than the `Task-Limited` models. With POS and NER being relatively shallower tasks, it makes sense that they benefit more from language-specific training that relies on token embeddings shared across tasks.

Investigating Sampling Strategies. In Table 2, all the scores shown for the `Temp` sampling strategy are the best scores across four different values of T, $T = 1, 2, 5, \infty$. (The complete table is available in Appendix C in the supplementary material.) We also present comparisons with the



(a) QA-en (b) NLI-es (c) POS-de

Figure 3: Evolution of ψs and rewards as a function of training time for three `Lang-Limited` tasks evaluated on (a) QA-en (b) NLI-es and (c) POS-de.

`mDDS`, `mDDS-Lang` and `mDDS-Task` sampling strategies enforced within the `Lang-Limited`, `Task-Limited` and `All TLPs` models, respectively. For POS and NER, our best meta-learned models are mostly `Lang-Limited` with `Temp` sampling. It is intuitive that for these shallower tasks, `mDDS` does not offer any benefits from allowing to sample instances from other tasks.

To better understand the effects of `mDDS` sampling, Figure 3 shows plots of the rewards and sampling probabilities ψ's computed as a function of training time for two deeper tasks - QA-en and NLI-es along with a shallower task - POS-de. We note that initially all the TLPs in any `mDDS` setting would start with similar rewards, thus lending ψ's to converge towards the $T = \infty$ state. We highlight the following three observations:

- We find that the `mDDS` strategy does not help NLI at all. This is because the NLI task occupies the largest proportion across tasks at the start, as shown in Figure 2, and the proportion of NLI decreases substantially over time (since all tasks start with similar rewards at the beginning of meta training). Thus, for tasks that are over-represented in the meta-learning phase, temperature-based sampling is likely to be sufficient.

- We observe that the rewards for both QA and NLI are consistently high, irrespective of the target TLP. This suggests that both QA and NLI are information-rich tasks and could benefit other tasks in meta-learning. This is also apparent from the accuracies for PA in Table 2, where all the best meta-learned models employ `mDDS` sampling.

- From the sampling probabilities for QA-en, we see that both QA and NLI are given almost equal weightage. However, from the F1 scores in Table 2, the best numbers for QA are in the `Task-Limited` setting which suggests that QA does not benefit from any other task. One explanation for this could be that the sequence length of inputs for NLI is 128 while the inputs for QA are of length 384, thus allowing lesser room for QA to be benefited by NLI.

Zero-shot Evaluations. Zero-shot evaluation is performed on languages that were not part of the training (henceforth, we refer to them as external languages). In the case of QA, NLI and PA we select all external language for which datasets were available in XTREME. For NER and POS, the number of external languages is close to 35 so we choose a subset of these to report the results. For evaluation, we compare models that are agnostic to the target language during meta training (`Task-Limited`, `All TLPs` and `All TLPs mDDS-Task`). Since `Lang-Limited MTL` is language specific and does not offer a competitive

Model	NER (Acc.)								POS (Acc.)						
	bn	et	fi	ja	mr	ta	te	ur	et	fi	ja	mr	ta	te	ur
`Task-Limited MTL`	81.80	93.98	94.47	81.03	90.63	83.46	87.67	69.25	85.21	83.98	58.42	72.56	73.88	79.15	86.08
`All TLPs MTL`	77.49	90.35	92.65	77.80	81.19	81.21	86.17	64.27	69.63	73.50	57.24	68.80	70.52	72.41	81.59
`Task-Limited`	+1.91	+0.63	+0.16	+0.35	-0.67	+1.34	+0.63	+2.14	+2.94	+2.15	+0.83	+8.64	+2.34	+2.82	-0.30
`All TLPs`	+0.62	+0.35	-0.11	+0.19	-0.92	+1.25	+0.43	+9.10	+2.56	+2.01	-1.42	+8.27	+1.24	+2.51	-0.16
`All TLPs mDDS-Task`	-0.83	+0.09	-0.20	-1.34	-1.87	+0.49	+0.05	+3.62	+1.91	+1.08	-1.74	+8.64	+1.24	+1.88	-0.72

Model	QA (F1)		NLI (Acc.)									PA (Acc.)	
	ar	vi	ar	bg	el	ru	sw	th	tr	ur	vi	ja	ko
`Task-Limited MTL`	32.25	44.35	62.88	67.47	66.09	67.85	43.61	43.16	57.79	57.03	69.45	78.23	74.85
`All TLPs MTL`	40.14	54.08	64.54	67.99	66.25	70.05	43.89	45.72	56.73	56.93	72.02	77.61	73.49
`Task-Limited`	+8.14	+6.63	+4.35	+5.15	+4.62	+2.72	+8.51	+14.42	+6.79	+5.27	+1.3	+0.21	+1.81
`All TLPs`	+5.24	+3.62	+4.41	+4.73	+4.79	+2.94	+11.44	+13.04	+7.05	+5.67	+1.24	+3.07	+4.57
`All TLPs mDDS-Task`	+6.89	+6.29	+3.19	+4.33	+4.09	+2.38	+8.71	+13.16	+7.09	+4.41	+1.04	+2.81	+4.92

Table 3: Results comparing Zero-shot evaluations for several external languages with competitive MTL baselines. The best MTL model is highlighted using orange. Rows for meta models show the difference (positive or negative) of the meta model result from the best MTL setting (orange) for that column

baseline when applied to an external language, we compare against `Task-Limited MTL` and `All TLPs MTL` that are more competitive.

An interesting observation from the zero shot results in Table 3 is that for every external language, on the 'shallower' NER and POS tasks, the `Task-Limited` variant of meta-learning performs better than both the variants of MTL, *viz.*, `Task-Limited MTL` and `All TLPs MTL`. In contrast, the 'deeper' tasks, *viz.*, QA, NLI and PA benefit more from the use of meta-learning using `All TLPs` setting, presumably because, as argued earlier, the deeper tasks tend to help each other more.

6 Conclusion

We present effective use of meta-learning for capturing task and language interactions in multi-task, multi-lingual settings. The effective use involves appropriate strategies for sampling tasks and languages as well as rough knowledge of the level of abstraction (deep vs. shallow representation) of that task. We present experiments on the XTREME multilingual benchmark dataset using five tasks and six languages. Our meta-learned model shows clear performance improvements over competitive baseline models. We observe that deeper tasks consistently benefit from meta-learning. Furthermore, shallower tasks benefit from deeper tasks when meta-learning is restricted to a single language. Finally, zero-shot evaluations for several external languages demonstrate the benefit of using meta-learning over two multi-task baselines while also reinforcing the linguistic insight that tasks requiring deeper representations tend to collaborate better.

Acknowledgements

We thank anonymous reviewers for providing constructive feedback. We are grateful to IBM Research, India (specifically the IBM AI Horizon Networks - IIT Bombay initiative) for their support and sponsorship.

References

Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. 2017. Cross-lingual word embeddings for low-resource language modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 937–947.

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, M. Gatu Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *ArXiv*, abs/1907.05019.

Mikel Artetxe and Holger Schwenk. 2019. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the ACL 2019*.

Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ICML*.

Junkun Chen, Xipeng Qiu, Pengfei Liu, and Xuanjing Huang. 2018. Meta multi-task learning for sequence modeling. In *AAAI*.

Benoît Colson, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization. *Annals of operations research*, 153(1):235–256.

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

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32, pages 7059–7069. Curran Associates, Inc.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia. PMLR.

Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Pratik Jawanpuria, Jagarlapudi Saketha Nath, and Ganesh Ramakrishnan. 2015. Generalized hierarchical kernel learning. *J. Mach. Learn. Res.*, 16:617–652.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Vishwajeet Kumar, Nitish Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019. Cross-lingual training for automatic question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4863–4872, Florence, Italy. Association for Computational Linguistics.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. MLQA: Evaluating Cross-lingual Extractive Question Answering. *arXiv preprint arXiv:1910.07475*.

Shikun Liu, Edward Johns, and Andrew J. Davison. 2019a. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *ArXiv*, abs/1803.02999.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4547–4562, Online. Association for Computational Linguistics.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

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

Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Adam Santoro, Sergey Bartunov, Matthew M Botvinick, Daan Wierstra, and Timothy P. Lillicrap. 2016. Meta-learning with memory-augmented neural networks. In *ICML*.

Bernard Vauquois. 1968. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In *IFIP Congress*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Dingquan Wang, Nanyun Peng, and Kevin Duh. 2017. A multi-task learning approach to adapting bilingual word embeddings for cross-lingual named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 383–388.

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Jaime Carbonell, and Graham Neubig. 2020a. Optimizing data usage via differentiable rewards. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9983–9995. PMLR.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020b. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8526–8537, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical*

Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Yongxin Yang and Timothy M. Hospedales. 2017. Deep multi-task representation learning: A tensor factorisation approach. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Appendices

Appendix A: Baseline Training Details

For QA learning rate is 3e-5 and sequence length is 384 and the model is trained for 2 epochs. For PA, NLI, POS and NER the learning rate is 2e-5 and sequence length is 128. NLI and PA models are trained for 5 epochs while POS and NER models are trained for 10 epochs. The choice of hyperparameters was kept constant across different languages for the same task.

Appendix B: Finetuning Details

For finetuning we kept the same number of epochs as the baseline of that task i.e 2 epochs for QA, 10 epochs for POS and NER, 5 epochs for NLI and PA. For QA we finetune with learning rate 3e-5 and 3e-6 and POS/NER we finetune with learning rate 2e-5 and 2e-6 and select the better of the two model. For PA and NLI the results for learning rate 2e-5 were consistently worse compared to 2e-6 so we just use lr = 2e-6 for PA and NLI.

Appendix C: Temperature Sampling

Model	T	QA (F1)				NLI (Acc.)				PA (Acc.)				
		en	hi	es	de	en	es	de	fr	en	es	de	fr	zh
`Baselines`		79.94	59.94	65.83	63.17	81.39	78.37	76.82	77.30	92.35	89.75	87.45	89.61	83.32
`Lang-Limited MTL`		69.80	53.24	62.29	58.91	80.49	76.10	75.18	74.94	93.75	87.75	85.35	88.55	80.49
`Task-Limited MTL`		74.04	57.77	64.28	61.47	80.95	78.15	75.90	77.14	93.65	86.65	86.25	86.82	81.24
`All TLPs MTL`		63.22	42.94	54.05	51.61	80.05	76.48	74.86	76.18	93.50	90.30	88.45	89.71	82.66
`Lang-Limited`	T = 1	79.49	59.42	64.67	63.04	81.13	78.76	76.23	76.51	93.85	89.15	87.83	89.63	82.56
	T = 2	78.81	59.68	65.10	63.24	80.87	77.56	76.85	76.60	93.85	90.15	87.70	89.41	83.10
	T = 5	79.90	58.74	65.56	62.12	81.19	78.17	76.10	76.56	93.65	90.35	88.60	90.11	83.20
	T = ∞	79.71	59.70	65.29	62.89	81.45	78.45	76.74	76.46	94.20	89.65	88.80	89.56	83.26
`Task-Limited`	T = 1	80.30	60.37	66.32	63.57	82.91	79.49	77.96	78.02	93.95	90.15	87.50	90.56	82.66
	T = 2	79.95	59.94	66.33	63.50	83.03	79.41	77.94	78.08	93.05	89.85	87.90	89.66	83.17
	T = 5	80.49	60.17	65.94	62.74	82.75	79.33	77.98	78.00	93.90	89.80	87.65	90.21	83.12
	T = ∞	79.77	59.86	66.01	62.96	83.03	79.39	78.07	78.09	93.60	89.75	87.75	89.61	82.42
`All TLPs`	T = 1	80.20	59.89	66.10	63.64	83.29	79.59	77.84	78.19	93.90	89.95	88.70	90.41	83.57
	T = 2	80.47	60.41	66.04	63.56	82.71	78.83	77.96	78.04	93.50	90.75	89.65	90.71	84.02
	T = 5	80.01	59.38	66.15	63.53	83.19	79.51	78.10	78.21	94.10	90.05	88.70	90.26	84.17
	T = ∞	80.27	59.82	64.41	63.08	83.27	79.43	78.27	78.25	94.05	90.75	88.70	90.76	83.42

Model	T	NER (Acc.)						POS (Acc.)				
		en	hi	es	de	fr	zh	en	hi	es	de	zh
`Baselines`		93.23	95.72	95.84	97.32	95.48	94.34	96.15	93.57	96.02	97.37	92.60
`Lang-Limited MTL`		92.54	92.67	95.14	96.40	94.38	92.97	95.08	92.43	95.19	97.19	89.71
`Task-Limited MTL`		93.51	93.94	95.77	97.09	95.27	93.72	95.70	93.34	95.73	97.35	92.52
`All TLPs MTL`		92.28	91.95	94.90	96.18	94.38	92.53	94.70	91.89	95.10	97.03	89.92
`Lang-Limited`	T = 1	93.14	95.36	95.40	97.21	95.39	93.63	95.96	93.33	95.81	97.32	92.32
	T = 2	93.24	94.76	95.80	97.56	95.07	93.53	95.87	93.53	95.93	97.39	92.40
	T = 5	94.03	95.78	95.93	97.24	94.99	93.60	96.09	93.56	95.85	97.33	92.43
	T = ∞	94.11	95.40	95.75	96.89	95.35	93.87	95.99	93.28	96.12	97.41	92.35
`Task-Limited`	T = 1	94.30	95.26	95.82	97.25	95.26	93.62	95.93	93.36	95.81	97.31	92.38
	T = 2	93.30	94.92	95.82	97.07	95.30	93.63	95.84	93.52	95.78	97.31	92.38
	T = 5	93.29	95.02	95.73	96.98	95.19	93.56	95.92	93.34	95.75	97.39	92.43
	T = ∞	93.37	94.70	95.84	96.95	95.20	93.83	95.77	93.33	95.76	97.33	92.51
`All TLPs`	T = 1	93.14	93.63	95.91	97.30	95.32	93.53	95.90	93.35	95.76	97.36	92.43
	T = 2	93.35	95.02	95.78	97.30	95.29	93.58	95.92	93.48	95.81	97.39	92.44
	T = 5	93.36	94.51	95.93	97.26	95.28	93.95	95.92	93.35	95.78	97.40	92.42
	T = ∞	93.35	94.95	95.97	97.32	95.28	93.63	95.93	93.31	95.80	97.30	92.43

Table 4: Detailed results of temperature based heuristic sampling for different selections settings. The best result among Baseline and three MTL models is highlighted using orange. For each column we present the difference (positive or negative) of the meta models from the best baseline (highlighted in orange) of that column

“Killing Me” Is Not a Spoiler: Spoiler Detection Model using Graph Neural Networks with Dependency Relation-Aware Attention Mechanism

Buru Chang† **Inggeol Lee**‡ **Hyunjae Kim**‡ **Jaewoo Kang**‡*
Hyperconnect† Korea University‡
`buru@hpcnt.com`
`{ingulbull,hyunjae-kim,kangj}@korea.ac.kr`

Abstract

Several machine learning-based spoiler detection models have been proposed recently to protect users from spoilers on review websites. Although dependency relations between context words are important for detecting spoilers, current attention-based spoiler detection models are insufficient for utilizing dependency relations. To address this problem, we propose a new spoiler detection model called SDGNN that is based on syntax-aware graph neural networks. In the experiments on two real-world benchmark datasets, we show that our SDGNN outperforms the existing spoiler detection models.

1 Introduction

Spoilers on review websites, which reveal critical details of the original works, can ruin an appreciation for the works. Review websites, such as Rotten Tomato, IMDb, and Metacritic, provide self-reporting systems that tag spoiler information to warn users of spoilers. However, since self-reporting systems depend solely on the active participation of users, they cannot handle the fast-growing volume of newly generated reviews. During the past decade, several machine learning-based spoiler detection (SD) models have been proposed to solve the inefficiency of self-reporting systems. Guo and Ramakrishnan (2010) proposed an automatic SD model that measures the similarity between reviews and synopses of movies. Support vector machine (SVM)-based SD models using handcrafted features have been proposed (Boyd-Graber et al., 2013; Jeon et al., 2016). Recently, attention-based SD models that utilize metadata of review documents achieve state-of-the-art performance on the SD task (Chang et al., 2018; Wan et al., 2019).

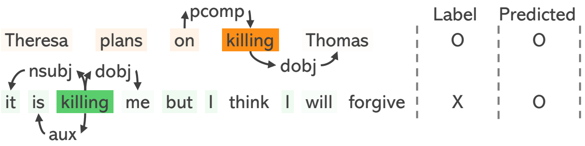


Figure 1: Attention-based models focused on the word “killing” because the word is frequently used in spoiler sentences, which results in incorrect predictions.

However, the attention-based SD models have a lack of using dependency relations between context words. Dependency relations are useful for capturing the semantics of given sentences and detecting spoilers. As shown in Figure 1, although the phrase “killing me” is not a spoiler because the phrase is a metaphor, the attention-based SD models often focus on the word “killing” and classify sentences that contain the phrase “killing me” as spoilers. By providing the information that the word “me” is used as the direct object of the verb “killing,” SD models can understand that the phrase is a metaphor.

In this paper, we propose SDGNN, which is a new **S**poiler **D**etection model based on syntax-aware **G**raph **N**eural **N**etworks (GNNs) (Marcheggiani and Titov, 2017) for leveraging dependency relations between context words in sentences to fully capture the semantics. With the success of GNNs We also propose a dependency relation-aware attention mechanism, which is a modification of the gating mechanism used by syntax-aware GNNs, to be suitable for the spoiler detection task. In SD, considering the relative importance of dependency relations. However, existing syntax-aware GNN-based models compute the importance of each dependency relation individually in sentences without considering the context of the given

---

[*] Corresponding author.

[†] This work was done while the author was affiliated with Korea University.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3613–3617
April 19 - 23, 2021.

sentence. Our proposed dependency relation-aware attention mechanism considers the relative importance of dependency relations. Also, we adopt a previously proposed genre-aware pooling method (Chang et al., 2018) to utilize the genre of works efficiently. In the experiments, we demonstrate the effectiveness of SDGNN on two real-world benchmark datasets in both quantitative and qualitative ways.

2 Our Approach

SDGNN classifies whether a given sentence $x = (w_1, w_2, \cdots, w_n)$ is a spoiler sentence. SDGNN consists of three stages: contextualized word representation, dependency relation-aware attention mechanism, and genre-aware pooling.

Contextualized Word Representation Each word w in the given sentence x is represented with the pretrained word embedding vector (Pennington et al., 2014). We then utilize bi-directional LSTMs (Hochreiter and Schmidhuber, 1997) to encode contextualized word representations $\mathbf{h}^{(0)} \in \mathbb{R}^d$.

Dependency Relation-Aware Attention Mechanism While the gating mechanism in syntax-aware GNNs (Marcheggiani and Titov, 2017; Nguyen and Grishman, 2018) computes the scalar weight of each dependency relation, it does not consider the relative importance of dependency relations, which varies depending on the context of the given sentence. We present a dependency relation-aware attention mechanism that considers the relative importance of dependency relations in the given sentence. The relation-aware attention weights are computed as follows:

$$a^{(k)}_{L(u,v)} = g\left(\mathbf{h}^{(k)}_u \overline{\mathbf{W}}^{(k)} \mathbf{e}^{(k)}_{L(u,v)} + \bar{b}^{(k)}_{L(u,v)}\right), \quad (1)$$

$$\hat{a}^{(k)}_{L(u,v)} = \frac{\exp\left(a^{(k)}_{L(u,v)}\right)}{\sum_{v' \in \mathcal{N}(u)} \exp\left(a^{(k)}_{L(u,v')}\right)}, \quad (2)$$

where $\hat{a}^{k}_{L(u,v)}$ is a scalar attention weight of the dependency relation label $L(u, v)$ of the edge between word nodes u, v. g is the non-linear function and $\exp(\cdot)$ is an exponential function. $\overline{\mathbf{W}}^{(k)} \in \mathbb{R}^{d \times d}$ is the attention weight matrix for the k-th layer. $\mathbf{e}^{(k)}_{L(u,v)} \in \mathbb{R}^d$ and $\bar{b}^{k}_{L(u,v)} \in \mathbb{R}$ are latent features of the dependency relation $L(u, v)$.

Finally, we aggregate the latent feature for each node u as follows:

$$\mathbf{h}^{(k)}_u = f\left(\sum_{v \in \mathcal{N}(u)} \hat{a}^{(k)}_{L(u,v)} \mathbf{W}^{(k)} \mathbf{h}^{(k-1)}_v + \mathbf{b}^{(k)}\right), \quad (3)$$

where $\mathbf{W}^{(k)} \in \mathbb{R}^{d \times d}$ and $b^{(k)} \in \mathbb{R}^d$ are the weight matrix and bias term, respectively, for the k-th layer. f is a non-linear function. $\mathbf{h}^{(0)}$ is the outputs of the bi-directional LSTMs in the previous stage.

There are two main differences between our proposed dependency-aware attention mechanism and the gating mechanism used by syntax-aware GNNS. First, the dependency-aware attention mechanism employs the softmax function to capture the relative importance of dependency relations, while the gating mechanism computes the scalar weights by the inner-product of latent features of words and dependency relations. Second, the gating mechanism utilizes only three dependency relations (*forward*, *backward*, and *self*) because of the over-parameterization issue. On the other hand, our proposed dependency relation-aware attention mechanism utilizes all the 82 types of dependency relations without suffering from the over-parameterization issue since the weight matrix in Equation 3 does not depend on the number of relations. The number of trainable parameters of SDGNN is proportionate to d^2 while that of syntax-aware GNNs is proportionate to $|L| \cdot d^2$, where $|L|$ is the number of relations.

Genre-Aware Pooling Genre information is useful for detecting spoilers. To leverage genre information, we employ a genre-aware pooling method following Chang et al. (2018). The genre-aware pooling computes the attention weights between the latent features of words and a genre feature captured from genre information of works. We then obtain a latent feature vector $\mathbf{x}$ for the given sentence x.

Optimization We compute the spoiler probability $\hat{y}$ of the given sentence x with the following the linear transformation:

$$\hat{y} = \sigma(\mathbf{w}\mathbf{x} + b), \quad (4)$$

where $\mathbf{w}$ and b are trainable parameters, and σ is a sigmoid function. We use the weighted binary cross entropy (Wan et al., 2019) as the loss function.

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{x_i \in \mathcal{D}} (y_i \texttt{log}(\hat{y}_i) + \eta \cdot (1 - y_i) \texttt{log}(1 - \hat{y}_i)), \quad (5)$$

where y id the ground truth of spoiler information and $\mathcal{D}$ indicates the dataset. η is a hyperpameter used to balance the number of spoiler and non-spoiler labels in the training data. All the trainable parameters of SDGNN are updated by minimizing the loss function with gradient descent.

Statistics	Goodreads	TVTropes
# of Training Sentences	14,007,593	11,970
# of Validation Sentences	128,718	2,808
# of Test Sentences	3,536,341	1,477
# of Edge Types	82	82
# of Genre	542	30
Avg. # of Nodes per Sentence	17.7	21.03
Avg. # of Edges per Sentence	33.4	40.06
Avg. # of Genre per Sentence	4.95	2.40

Table 1: Statistics of the datasets.

3 Experiments

3.1 Experimental Setup

Datasets We evaluated our proposed model on the two public spoiler datasets: **Goodreads** (Wan et al., 2019) and **TVTropes** (Boyd-Graber et al., 2013) [1]. The Goodreads dataset consists of spoiler sentences on book reviews, and only 3.22% of entire sentences are labeled as spoiler sentences. The TVTropes dataset consists of descriptions of 884 TV programs from the TVTropes site, and 52.7% of the descriptions are labeled as spoilers. The statistics of the datasets are summarized in Table 1.

Baseline Models We compared our proposed model with the following state-of-the-art SD models: **SVM** (Boyd-Graber et al., 2013; Jeon et al., 2016), **CNN** (Kim, 2014), **HAN** (Yang et al., 2016), **SpoilerNet** (Wan et al., 2019) and **DNSD** (Chang et al., 2018). Note that the implementation details about our experiments are described in the Appendix due to space limitations.

Metrics We use Area Under the Receiver Operating Characteristics curve (**AUROC**) used in Wan et al. (2019) as an evaluation metric. We also use an **F1** score following Chang et al. (2018).

Implemention Details We trained and evaluated the models on two TITAN X (Pascal) GPUs. We implemented SDGNN using PyTorch v1.1. We used Stanford CoreNLP (Manning et al., 2014) to generate dependency parse trees. We employed GloVe (Pennington et al., 2014) to represent word vectors in neural network-based models including SDGNN. Using the validation set and grid search, we searched optimal hyper-parameters for each SD model. All the neural network-based models were trained with the learning rate of 0.001 and the Adam optimizer (Kingma and Ba, 2014). A batch size of 1024 was used for training TextCNN, HAN, and DNSD, and a batch size of 512 was used for training SpoilerNet and SDGNN. To prevent overfitting, we applied L2-normalization with λ = 1e-5 and a dropout rate of 0.5. For TextCNN, we used 50 filters with kernel sizes of 3, 4, and 5. For efficient training on deep learning libraries, SDGNN set the maximum number of words to 50. For SDGNN, we used Leaky ReLU for the non-linear function g, and ReLU for f. We set $k = 2$ for SyntacticGCN, C-GCN, and SDGNN. We use $\eta = 0.05$ for the Goodreads dataset, which is unbalanced.

Models	Goodreads AUROC	Goodreads F1	TVTropes AUROC	TVTropes F1
SVM	0.880	0.162	0.735	0.698
TextCNN	0.904	0.188	0.779	0.738
HAN	0.915	0.190	0.785	0.750
SpoilerNet	0.924	0.194	0.808	0.768
DNSD	0.928	0.199	0.818	0.788
SDGNN	**0.938**	**0.210**	**0.828**	**0.801**

Table 2: Evaluation results on two benchmark datasets. The best results are highlighted in bold.

Models	Goodreads AUROC	Goodreads F1
SytacticGCN	0.933	0.204
C-GCN	0.923	0.193
SDGNN	**0.938**	**0.210**

Table 3: Evaluation results on the Goodreads dataset.

3.2 Results

The experimental results are summarized in Table 2. Evaluation results show that our proposed SDGNN outperforms all the baseline models including attention-based models. This result demonstrates that our proposed dependency relation-aware attention mechanism contributes to improving SD performance.

4 Analysis and Discussion

4.1 Analysis of Relative Importance

To further demonstrate the usefulness of the relative importance of dependency relations, we conducted quantitative and qualitative analysis.

Quantitative We compared SDGNN with the more syntax-aware GNN-based models, SyntacticGCN (Marcheggiani and Titov, 2017) and C-GCN (Zhang et al., 2018). We trained and evaluated the models on the Goodreads dataset. We

---

[1] We obtained the datasets from Wan et al. (2019) and Boyd-Graber et al. (2013), respectively.

Models	Genres	Sentences	Prediction	Label
DNSD	Romance	it 's killing me but i think i 'll forgive him no matter what he did or did n't do .	Positive	Negative
	Fantasy	the villains are decidedly vicious and in some cases insane .	Positive	Negative
SDGNN	Romance	it 's killing me but i think i 'll forgive him no matter what he did or did n't do .	Negative	Negative
	Fantasy	the villains are decidedly vicious and in some cases insane .	Negative	Negative

Table 4: Visualization of attention scores from DNSD, SpoilerNet, and SDGNN on test data.

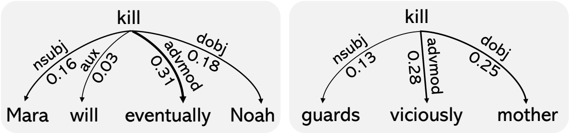


Figure 2: Partial graphs of dependency parse trees with dependency relation-aware attention weights.

utilize contextualized word representations and the genre-aware pooling method to SyntacticGCN and C-GCN. The evaluation results are summarized in Table 3. Our proposed SDGNN outperformed SyntacticGCN and C-GCN. This result demonstrates that our proposed attention mechanism is effective by considering the relative importance of dependency relations. Although SDGNN significantly reduced the number of parameters, SDGNN achieved better results compared to SyntacticGCN and C-GCN.

Qualitative In Figure 2, the attention weights of the adverbial modifier (advmod) linked to the words "eventually" and "viciously" are high, which indicates that adverbial modifiers frequently can be important hints for detecting spoilers. In the right partial graph, the attention weight of the (dobj) is relatively higher than that in the left partial graph. Since the word "mother" is not typically used as the object of the word "kill" in the original works, the phrase "kill mother" is a critical hint in detecting spoilers, and SDGNN effectively captures the phrase.

4.2 Case Study

We sample several sentences from the test set of the Goodreads dataset to explore how the models detect spoilers. Table 4 shows the visualization of attention scores in the pooling layer obtained by DNSD and SDGNN, respectively. The first sentence contains the verb "killing," but it is not a spoiler sentence because the phrase "killing me" is a metaphor. In this case, DNSD failed to correctly classify the sentence since DNSD cannot fully capture the semantics of the sentence. On the other hand, SDGNN focused on not only the word "killing" but also on the word "me" and classified the sentence correctly since SDGNN employs the dependency relation (dobj) between the word "me" and the word "killing".

The second sentence is a non-spoiler because it is obvious that villains are vicious in most original works. DNSD classified the sentence as a spoiler because the model solely focused on individual words such as "villains", "vicious", and "insane", rather than the understanding of the overall semantics of the sentence. On the other hand, SDGNN classified the sentence correctly as the word "are" is used to describe characters in many cases, and SDGNN understands the semantics of the sentence.

4.3 Discussion

Dependency Parsing on User-Generated Texts The spoiler datasets are user-generated texts, which are intrinsically noisy. To examine the influence of noises on dependency parsing results and the performance of SDGNN, we sampled 100 sentences from Goodreads. We manually classified whether the sentences are noise or not, and 28 of 100 sentences were classified as noisy sentences. Dependency parsing results on well-structured sentences seem good, but dependency parsing results on noisy sentences are poor. However, there is no significant gap in performance. SDGNN achieved 85.7% accuracy on noisy sentences and 87.5% accuracy on well-structured sentences. Since our proposed dependency relation-aware attention mechanism of SDGNN filters noisy information, SDGNN could detect spoilers even on noisy sentences.

Subjectivity in Judging Spoilers Since judging a sentence as a spoiler is a subjective task, label inconsistency occurs in spoiler datasets crawled from self-reporting systems. Guo and Ramakrishnan (2010) found that 23% of the labels of their manually labeled data is different from the original labels of IMDb reviews. One of the ways to mitigate label inconsistency is to solidify the definition of a spoiler. Although the TV Tropes site defines spoilers, efforts should be made for a more rigorous and linguistic definition in future studies. Another possible way is to employ reviewers' information in detecting spoilers. *Reviewer biases* of SpoilerNet can alleviate label inconsistency between users.

5 Conclusion

In this paper, we proposed a novel spoiler detection model called SDGNN which is based on syntax-aware GNNs that utilize dependency relations between context words. We also proposed a dependency relation-aware attention mechanism for considering the relative importance of dependency relations. In the experiments, our proposed SDGNN model achieved the state-of-the-art performance on two spoiler datasets. Our experimental results demonstrate the effectiveness of dependency relations in the spoiler detection task and our dependency relation-aware attention mechanism.

Acknowledgments

This research was supported by National Research Foundation of Korea (NRF-2020R1A2C3010638). This research was also supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Creative Consilience program (IITP-2020-0-01819) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation).

References

Jordan Boyd-Graber, Kimberly Glasgow, and Jackie Sauter Zajac. 2013. Spoiler alert: Machine learning approaches to detect social media posts with revelatory information. In *Proceedings of the 76th ASIS&T Annual Meeting: Beyond the Cloud: Rethinking Information Boundaries*, page 45. American Society for Information Science.

Buru Chang, Hyunjae Kim, Raehyun Kim, Deahan Kim, and Jaewoo Kang. 2018. A deep neural spoiler detection model using a genre-aware attention mechanism. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 183–195. Springer.

Sheng Guo and Naren Ramakrishnan. 2010. Finding the storyteller: automatic spoiler tagging using linguistic cues. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 412–420. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sungho Jeon, Sungchul Kim, and Hwanjo Yu. 2016. Spoiler detection in tv program tweets. *Information Sciences*, 329:220–235.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. 2019. Fine-grained spoiler detection from large-scale review corpora. *arXiv preprint arXiv:1905.13416*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Yuhao Zhang, Peng Qi, and Christopher D Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*.

BERTese: Learning to Speak to BERT

Adi Haviv[1] **Jonathan Berant**[1,2] **Amir Globerson**[1]

[1]School of Computer Science, Tel Aviv University
[2]Allen Institute for AI
{adi.haviv,joberant}@cs.tau.ac.il, gamir@post.tau.ac.il

Abstract

Large pre-trained language models have been shown to encode large amounts of world and commonsense knowledge in their parameters, leading to substantial interest in methods for extracting that knowledge. In past work, knowledge was extracted by taking manually-authored queries and gathering paraphrases for them using a separate pipeline. In this work, we propose a method for automatically rewriting queries into "BERTese", a paraphrase query that is directly optimized towards better knowledge extraction. To encourage meaningful rewrites, we add auxiliary loss functions that encourage the query to correspond to actual language tokens. We empirically show our approach outperforms competing baselines, obviating the need for complex pipelines. Moreover, BERTese provides some insight into the type of language that helps language models perform knowledge extraction.

1 Introduction

Recent work has shown that large pre-trained language models (LM), trained with a masked language modeling (MLM) objective (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2019; Sanh et al., 2019; Conneau et al., 2020), encode substantial amounts of world knowledge in their parameters. This has led to ample research on developing methods for extracting that knowledge (Petroni et al., 2019, 2020; Jiang et al., 2020; Bouraoui et al., 2020). The most straightforward approach is to present the model with a manually-crafted query such as *"Dante was born in [MASK]"* and check if the model predicts *"Florence"* in the *[MASK]* position. However, when this fails, it is difficult to determine if the knowledge is absent from the LM or if the model failed to understand the query itself. For example, the model might return the correct answer if the query is *"Dante was born in the city of [MASK]"*.

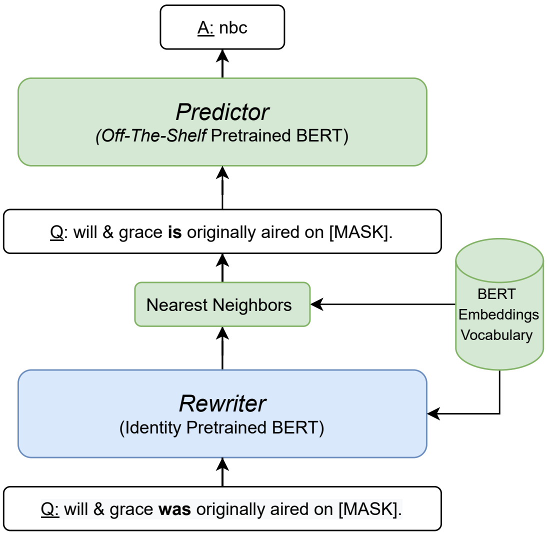


Figure 1: The BERTese Model. The model takes an input query, rewrites it, and feeds the output to a pre-trained BERT model. The untrained components are marked in green, and the blue component is trained.

Motivated by the above observation, we ask: can we automatically find the best way to "ask" an LM about its knowledge? We refer to this challenge as speaking "BERTese". In particular, we ask how to rewrite a knowledge-seeking query into one that MLMs understand better, where understanding is manifested by providing a correct answer to the query.

Prior work (Jiang et al., 2020) tackled this problem using a 2-step pipeline, where first a small list of paraphrase templates is collected using external resources, and then a model learns to extract knowledge by aggregating information from paraphrases of the input query. In this work, we propose a more general approach, where the model learns to rewrite queries, directly driven by the objective of knowledge-extraction.

Figure 1 provides an overview of our approach. Our model contains a BERT-based *rewriter*, which

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3618–3623
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

takes a query as input, and outputs for each input position a new token, which is its rewrite. This new query is fed into a different BERT *predictor* from which the answer is extracted. Importantly, the downstream predictor BERT is a fixed pre-trained model, and thus the goal is to train the rewriter to produce queries for which the predictor outputs the correct answer.

A technical challenge is that outputting discrete tokens leads to a non-differentiable model, which we tackle by adding a loss term that encourages the rewriter's output to be similar to BERT token embeddings. Moreover, we must guarantee that the BERTese query contains the *[MASK]* token from which the answer will be read. To achieve this, we first add an auxiliary loss term that encourages the model to output precisely one masked token in the query rewrite. We then add a layer that finds the token index that most closely resembles *[MASK]*, and this is where we expect the correct answer to be completed. Training of this selection process is done using the straight-through estimator (Hinton, 2012; Bengio et al., 2013).

We evaluate our approach on the LAMA dataset (Petroni et al., 2019), and show that our model significantly improves the accuracy of knowledge extraction. Furthermore, many of the rewrites correspond to consistent changes in query wording (e.g., changing tense), and thus provide information on the types of changes that are useful for extracting knowledge from BERT. While we experiment on BERT, our method is generic and can be applied to any MLM.

Taken together, our results demonstrate the potential of rewriting inputs to language models for both obtaining better predictions, and for potentially gaining insights into how knowledge is represented in these models. Our code can be downloaded from `https://github.com/adihaviv/bertese`.

2 Related Work

Choosing the right language for extracting world knowledge from LMs has attracted much interest recently. First, Petroni et al. (2019) observed that MLMs can complete simple queries with correct factual information. Jiang et al. (2020) and Heinzerling and Inui (2020) then showed that in the zero-shot setting, small variations to such queries can lead to a drop in fact recall. Orthogonally, another line of research focused on query reformulation for standard Question Answering (QA) tasks. Gan and Ng (2019) demonstrated that even minor query modifications can lead to a significant decrease in performance for multiple QA models and tasks. Buck et al. (2017) showed that it is possible to train a neural network to reformulate a question using Reinforcement Learning (RL), optimizing the accuracy of a black-box QA system. Similarly, Nogueira and Cho (2017) used RL to create a query reformulation system that maximizes the recall of a black-box information retrieval engine.

Jiang et al. (2020) proposed an ensemble method for query reformulation from LMs, that includes: (1) mining new queries, (2) using an off-the-shelf pre-trained translation model to collect additional paraphrased queries with back-translation, and (3) using a re-ranker to select one or more of the new queries. They then feed those queries to BERT to get the masked token prediction.

In this work, we take the idea of Jiang et al. (2020) a step forward and train a model in an end-to-end fashion to generate rephrased queries which are optimized to maximize knowledge extraction from the MLM.[1]

3 The BERTese Model

Recall that our goal is to build a model that takes as input a query in natural language, and re-writes it into a query that will be fed as input to an existing BERT model.

We refer to the above re-writing model as the *rewriter* and the existing BERT model as the *predictor*. We note that both input and output queries should include the token [MASK]. For example the input could be *"Obama was born in [MASK]"* and the output *"Obama was born in the state of [MASK]"*.

We first describe the behaviour of our model at inference time (see Figure 1). Given a query, which is a sequence of tokens, $S = (s_1, \ldots, s_n)$, we map S into a sequence of vectors $Q(S) \in \mathbb{R}^{d \times n}$ using BERT's embeddings of dimensionality d. This input is fed into a (BERT-based) stack of transformer layers that outputs a new sequence of vectors $\hat{Q}(S) \in \mathbb{R}^{d \times n}$.

To obtain vectors that can be used as input to the predictor, we need to map the vectors in each position to their nearest neighbor in the set of BERT

---

[1] Although knowledge retrieval has been investigated in autoregressive models as well, similar to Jiang et al. (2020), in this work we focus on MLMs only, as AR-LM only predict an answer if the masked token is at the end of the query.

embeddings. Specifically, let B_V be the set of BERT embeddings, and let $\hat{Q}_i \in \mathbb{R}^d$ be the re-written vector in position i. We map $\hat{Q}_i \in \mathbb{R}^d$ to $\arg\min_{\mathbf{v} \in B_V} \left(\left\| v - \hat{Q}_i \right\|_2^2 \right)$. We next pass the re-written query into the pre-trained predictor BERT model, and obtain an answer from the most probable token in the masked position.

Training this model involves two technical challenges. First, the nearest-neighbor operation is non-differentiable. Second, to obtain the prediction of the `[MASK]` token, we need to guarantee that the rewriter generates a `[MASK]` token, and know its position (because this is where the ground-truth answer should be predicted). We overcome these by adding two auxiliary loss functions. The first encourages the model to output vectors that are similar to BERT embeddings (thus reducing the loss in the nearest neighbor operation), and the second encourages the model to output one masked token.

Finally, we apply the straight-through estimator, which allows us to feed discrete word representations into the predictor and backpropagate the signal back to the rewriter. We next provide more details on the terms in our loss function used to train the rewriter.

Valid Token Loss: At training time we do not apply the non-differentiable nearest-neighbor operation. Thus, we would like the vectors $\hat{Q}(S)$ output by the rewriter to be as close as possible to valid BERT embeddings. This loss is the average over tokens of the distance between a re-written query token and its nearest neighbor:

$$f_1(S) = \frac{1}{|\hat{Q}(S)|} \sum_{\mathbf{q} \in \hat{Q}(S)} \min_{\mathbf{v} \in B_V} \left(\|\mathbf{v} - \mathbf{q}\|_2^2 \right). \quad (1)$$

Single [MASK] Loss: The output of the rewriter must contain the *[MASK]* token, so that the predictor can extract an answer from this token. To encourage the rewriter to output a [MASK] we add a loss as follows. We define the following "softmin" distribution over $i \in \{1, \ldots, |\hat{Q}(S)|\}$:

$$m_i(S) = \frac{e^{-\beta \left\| B_{[MASK]} - \hat{Q}_i(S) \right\|_2^2}}{\sum_j e^{-\beta \left\| B_{[MASK]} - \hat{Q}_j(S) \right\|_2^2}}, \quad (2)$$

where β is a trained parameter. The maximum value of this distribution will be highest when there is a single index i that is closest to the embedding of [MASK] (if there are two maxima, they will both have equal values). Thus the loss we consider is:

$$f_2(S) = -\max_i m_i(S). \quad (3)$$

Prediction Loss: The predictor should return the gold answer y when given $\hat{Q}$ as input. Without non-differentiability, we could find the index of the [MASK] token in $\hat{Q}$, and use cross-entropy loss between the output distribution of the predictor in that index and the gold answer y. To remedy this, we use a differentiable formulation, combined with the straight-through estimator (STE) (Bengio et al., 2013): Let $\mathbf{o}_i$ be the output distribution at the i^{th} position of the predictor, and let $\ell(y, \mathbf{p})$ be the cross-entropy between the one-hot distribution corresponding to y and a distribution $\mathbf{p}$. Then, we use the loss:

$$f_{CE}(S, y) = \sum_i m_i(S) \ell(y, \mathbf{o}_i). \quad (4)$$

Thus, if m is a one-hot on the index corresponding to [MASK], the loss will be the desired cross-entropy between the gold answer and the predicted distribution. We optimize this objective using the STE. Namely, in the forward pass, we convert m to a one-hot vector.

Our final training loss is the sum of the above three loss terms:

$$L(S, y) = f_{CE}(S, y) + \lambda_1 \cdot f_1(S) + \lambda_2 \cdot f_2(S). \quad (5)$$

The weights λ_1, λ_2 are tuned using cross-validation.

To summarize, the main challenge is that the rewriter output needs to be optimized to predict the correct label for the [MASK] token (Eq. 4). However, the [MASK] token needs to appear once in the rewriter output. In order to enforce the above, the "Single [MASK] Loss" (Eq. 3) is used. In addition, in order for the rewriter output to be a valid sentence, the "Valid Token Loss" (Eq. 1) is added. This encourages the model to output tokens that are close to BERT input embeddings. This is done by minimizing the distance between each rewriter vector to some vector in the BERT input embedding dictionary.

Rewriter pre-training We initialize the rewriter with a BERT-based model, additionally fine-tuned to output the exact word embeddings it received as input (i.e., fine-tuned to the identity mapping). Thus, when training for knowledge extraction, the rewriter is initialized to output exactly the query it received as input.

4 Experiments

Experimental setup We conduct our experiments on the LAMA dataset (Petroni et al., 2019; Jiang et al., 2020), a recently introduced unsupervised knowledge-extraction benchmark for pretrained LMs. LAMA is composed of a collection of cloze-style queries about relational facts with a single token answer. As in Jiang et al. (2020), we limit our main experiment to the T-REx (Elsahar et al., 2018) subset. The T-REx dataset is constructed out of 41 relations, each associated with at most 1000 queries, all extracted from Wikidata.

For training our model, we use a separate training set, created by Jiang et al. (2020), called T-REx-train. This dataset is constructed from Wikidata and has no overlap with the original T-REx dataset. We evaluate our model on the complete T-REx dataset.

Implementation Details Both the rewriter and the predictor are based on $\text{BERT}_{\text{base}}$ with the default settings from the Huggingface (Wolf et al., 2020) platform. We optimize BERTese using AdamW with an initial learning rate of 1e-5. We train the model on a single 32GB NVIDIA V100 for 5 epochs with a batch size of 64. For the loss coefficients (see Eq. (5)) we set $\lambda_1 = 0.3$ and $\lambda_2 = 0.5$.

Baselines We compare our method to three baselines: (a) BERT - A $\text{BERT}_{\text{base}}$ model without any fine-tuning, as evaluated in Petroni et al. (2019). (b) LPAQA - The model proposed by Jiang et al. (2020), based on mining additional paraphrase queries. We report results on a single paraphrase.[2] (c) FT-BERT: An end-to-end differentiable $\text{BERT}_{\text{base}}$ model, explicitly fine-tuned on T-REx-train to output the correct answer. This model, like ours, is trained for knowledge extraction, but does this internally, without exposing an interpretable intermediate textual rewrite.

Results We use the same evaluation metrics as Petroni et al. (2019) and report precision at one (P@1) macro-averaged over relations (we first average within relations and then across relations). As shown in Table 1, BERTese outperforms all three baselines. Compared to the zero-shot setting, where BERT is untrained on any additional data, we improve performance from $31.1 \rightarrow 38.3$. Our model also outperforms a BERT model fine-tuned for knowledge extraction on the same data as our model ($36 \rightarrow 38.3$). Last, we outperform the $\text{BERT}_{\text{base}}$ version of LPAQA by more than 4 points.

Table 2 presents example rewrites that are output by our model. It can be seen that rewrites are usually semantically plausible, and make small changes that are not meaningful to humans, but seem to help extract information from BERT, such as *was* → *is* and *a* → *the*. In some cases, rewrites can be interpreted, for example, replacing the word *airfield* with the more frequent word *airport*.

Ablation Study In Table 3 we present P@1 results on the T-REx test set after ablating different parts of the loss function. We keep the same label loss, same rewriter pretraining scheme, hyperparameters, and inference process. We show that removing all auxiliary losses hurts performance significantly on the T-REx dataset. Next, we evaluate the impact of removing the “Single [MASK] Loss”, and report a drop from 38.3 to 37.3. In addition, when further observing the rewrites the model produces, we find that those will have in some cases more than one [MASK] token. Overall, the results show that having just one of the loss terms substantially improves the performance (either “Valid Token Loss” or “Single [MASK] Loss”), but using both losses further improves accuracy.

Ablation	**P@1**
No auxilary losses	25.3
SML	36.6
VTL	37.5
SML + VTL (BERTese)	38.3

Table 3: Ablation experiments on T-REx. We abbreviate the ”Single [MASK] token” as SML and the ”Valid Token Loss” as VTL.

Part Of Speech Analysis To better understand what types of changes our rewriter performs, Table 4 shows the distribution over part-of-speech-tags replaced by the rewriter. We show all part-of-speech tags for which the frequency is higher than 1%. More than 70% of the replacements are nouns and verbs, which carry substantial semantic content. Interestingly, 15% of the replacements are determiners, which bear little semantic content.

---

[2] It is possible to improve results by aggregating over multiple rewrites, but our focus is on a single rewrite.

Corpus	**BERT**	**FT-BERT**	**LPAQA**	**BERTese**
T-REx	31.1	36	34.1	**38.3**

Table 1: Mean precision at one (P@1) for three baselines and our BERTese model on the T-REx dataset.

Modification	**Original Masked Query**	**Bertese Masked Query**
”!” removed	yahoo**!** tech is owned by [MASK].	yahoo tech is owned by [MASK].
verb patterns	**working** dog is a subclass of [MASK].	**work** dog is a subclass of [MASK].
was → is	will & grace **was** originally aired on [MASK].	will & grace **is** originally aired on [MASK].
a → the	tom terriss is **a** [MASK] by profession.	tom terriss is **the** [MASK] by profession.
rephrasing	istanbul hezarfen **airfield** is named after [MASK].	istanbul hezarfen **airport** is named after [MASK].
token → [SEP]	**lub**ka kolessa plays [MASK].	**[SEP]**ka kolessa plays [MASK].

Table 2: Examples of rewrites from the T-REx test-set, where the original query resulted in a wrong answer, and the BERTese rewrite resulted in correct one.

POS Tag	**Frequency**
NN	47.6%
VBN	23%
DT	15.3%
JJ	4.4%
CD	3%
NNP	1.7%
NNS	1.3%

Table 4: Part-of-speech analysis of rewrites from the T-REx test-set.

5 Conclusion

We presented an approach for modifying the input to a BERT model, such that factual information can be more accurately extracted. Our approach uses a trained rewrite model that is optimized to maximize the accuracy of its rewrites, when used as input to BERT. Our rewriting scheme indeed turns out to produce more accurate results than baselines. Interestingly, our rewrites are fairly small modifications, highlighting the fact that BERT models are not invariant to these edits.

Our approach is not limited to knowledge extraction. It can, in principle, be applied to BERT in general question answering datasets and even language modeling. In the former, we can change the predictor to a multiple-choice QA pretrained BERT and exclude the single [MASK] token loss. In the latter, we can for example envision a case where rewriting a sentence can make it easier to complete a masked word.

Our empirical setting focuses on the LAMA dataset, where a single mask token prediction is required. There are several possible extensions to multiple masks, and we leave these for future work. Finally, it will be interesting to test the approach on other masked language models such as RoBERTa (Liu et al., 2019) and ERNIE (Zhang et al., 2019), a MLM that is enhanced with external entity representations.

6 Acknowledgments

We thank Yuval Kirstain for his contribution in both the Rewriter Pretraining and overall valuable comments, and Edo Cohen-Karlik for his insightful suggestions. This project received funding from the European Research Council (ERC) under the European Union Horizon 2020 research and innovation programme (grant ERC HOLI 819080).

References

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7456–7463.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Andrea Gesmundo, Neil Houlsby, Wojciech Gajewski, and Wei Wang. 2017. Ask the right questions: Active question reformulation with reinforcement learning. *CoRR*, abs/1705.07830.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco

Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. *ACL*.

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

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Wee Chung Gan and Hwee Tou Ng. 2019. Improving the robustness of question answering systems to question paraphrasing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6065–6075, Florence, Italy. Association for Computational Linguistics.

Benjamin Heinzerling and Kentaro Inui. 2020. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. *ArXiv*, abs/2008.09036.

Geoffrey Hinton. 2012. Neural networks for machine learning. In *Coursera, video lectures*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Lifelong Knowledge-Enriched Social Event Representation Learning

Prashanth Vijayaraghavan
MIT Media Lab
Cambridge, MA, USA
pralav@mit.edu

Deb Roy
MIT Media Lab
Cambridge, MA, USA
dkroy@media.mit.edu

Abstract

The ability of humans to symbolically represent social events and situations is crucial for various interactions in everyday life. Several studies in cognitive psychology have established the role of mental state attributions in effectively representing variable aspects of these social events. In the past, NLP research on learning event representations often focuses on construing syntactic and semantic information from language. However, they fail to consider the importance of pragmatic aspects and the need to consistently update new social situational information without forgetting the accumulated experiences. In this work, we propose a representation learning framework to directly address these shortcomings by integrating social commonsense knowledge with recent advancements in the space of lifelong language learning. First, we investigate methods to incorporate pragmatic aspects into our social event embeddings by leveraging social commonsense knowledge. Next, we introduce continual learning strategies that allow for incremental consolidation of new knowledge while retaining and promoting efficient usage of prior knowledge. Experimental results on event similarity, reasoning, and paraphrase detection tasks prove the efficacy of our social event embeddings.

1 Introduction

Everyday life comprises the ways in which people typically act, think, and feel on a daily basis. Our life experiences unfold naturally into temporally extended daily events. The event descriptions can be packaged in various ways depending on several factors like speaker's perspective or the related domain. Interpretation of event descriptions will be incomplete without understanding multiple entities involved in the events and even more so when the focus is primarily on "social events", i.e., events explaining social situations and interactions. Therefore, a social event representation model must capture the semantic properties from the event text description and embed salient knowledge that encompasses the implicit pragmatic abilities. Early definitions of pragmatic aspects refer to the use of language in context; comprising the verbal, paralinguistic, and non-verbal elements of language (Adams et al., 2005). Contemporary definitions have expanded beyond just communicative functions to include behavior that includes social, emotional, and communicative aspects of language (Adams et al., 2005; Parsons et al., 2017).

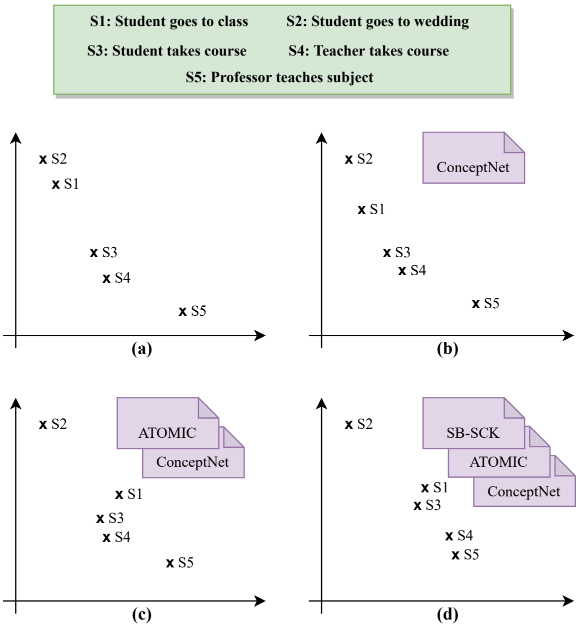


Figure 1: Illustration of functioning of our lifelong representation learning approach that produces incrementally richer social event representations. Event texts are given in the top green box. With more knowledge, social event embeddings move beyond high lexical overlap [shown in (a)] and learn to integrate semantic and pragmatic properties [shown in (b), (c)] of event texts along with social role information [shown in (d)].

Moving away from the extensively studied

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3624–3635
April 19 - 23, 2021.

speech acts, we analyze characteristics that reflect how a person behaves in social situations and how social contextual aspects influence linguistic meaning. In the context of event representations, the pragmatic properties can specifically refer to the human's inferred implicit understanding of event actors' intents, beliefs, and feelings or reactions (Wood, 1976; Hopper and Naremore, 1978).

Understanding the pragmatic implications of social events is non-trivial for machines as they are not explicitly found in the event texts. Prior studies (Ding et al., 2014, 2015; Granroth-Wilding and Clark, 2016; Weber et al., 2018) often extract the syntactic and semantic information from the event descriptions but ignore the pragmatic aspects of language. In this work, we address this shortcoming, and aim to (a) disentangle semantic and pragmatic attributes from social event descriptions and (b) encapsulate these attributes into an embedding that can move beyond simple linguistic structures and dispel apparent ambiguities in the real sense of their context and meaning.

Towards this goal, we propose to train our models with social commonsense knowledge about events focusing specifically on intents and emotional reactions of people. Such commonsense understanding can be obtained from existing knowledge bases like ConceptNet (Speer et al., 2017), Event2Mind/ATOMIC (Sap et al., 2019a; Rashkin et al., 2018) or by collecting more noisy commonsense knowledge using data mining techniques. As new domain sources emerge, each containing different knowledge assertions, it is essential that the representation models for social events keep evolving with this growing knowledge. Since it is generally infeasible to retrain models from scratch for every new knowledge source, we consider the need to employ prominent continual learning practices (Kirkpatrick et al., 2017; Lopez-Paz and Ranzato, 2017; Asghar et al., 2018; d'Autume et al., 2019) and enable semantic and pragmatic enrichment of social event representations. This problem can be addressed from the perspective of incremental domain adaptation (Asghar et al., 2018; Wulfmeier et al., 2018), which quickly adapts to new domain knowledge without interfering with existing ones. Figure 1 presents a sample functioning scenario producing incrementally richer social event embeddings. As the model gains more knowledge from different sources, it learns to discern events based on semantic and pragmatic properties, including social roles. For example, "Student takes course", and "Teacher takes course" has significant lexical and semantic relatedness. However, the social role information changes the meaning as depicted in Figure 1(d) with the introduction of our in-house dataset (SB-SCK).

In this paper, we develop a lifelong representation learning approach for embedding social events from their free-form textual descriptions. Our model augments a growing set of knowledge obtained from various domain sources to allow for positive knowledge transfer across these domains. Our contributions are as follows:

- We propose a continual representation learning approach-that integrates both text encoding and lifelong learning techniques to aid better representation of social events.

- We adopt a domain-representative episodic memory replay strategy with text encoding techniques to effectively consolidate the expanding knowledge from several domain sources and generate a semantically & pragmatically enriched social event embedding.

- We evaluate our models primarily on four different tasks: (a) intent-emotion prediction for event texts based on an in-house *Lifelong EventRep Corpus*, (b) event similarity task using hard similarity dataset (Ding et al., 2019; Weber et al., 2018), (c) paraphrase detection using Twitter URL corpus (Lan et al., 2017), and (d) social commonsense reasoning task using SocialIQA (Sap et al., 2019b) dataset.

2 Related Work

2.1 Social Events Representation Learning

Early work in the domain of events can be traced back to modeling narrative chains. Chambers and Jurafsky (Chambers and Jurafsky, 2008, 2009) introduced models for event sequences involving coreference resolution and inferring event schemas. Similar efforts (Balasubramanian et al., 2013; Cheung et al., 2013; Jans et al., 2012) have explored the use of open-domain relations to extract event schemas but suffer from reduced predictive capabilities and increased sparsity. Recent advancements, aimed at addressing the limitations of prior works, compute distributed embeddings of events involving word embeddings, recurrent sequence models, and tensor-based composition models (Modi and

Titov, 2013; Granroth-Wilding and Clark, 2016; Pichotta and Mooney, 2016; Hu et al., 2017). Specifically, tensor-based methods have demonstrated improved performance by representing events that predict implicit arguments with event knowledge (Cheng and Erk, 2018), combine (subject, predicate, object) triples information (Weber et al., 2018) and reflect thematic fit (Tilk et al., 2016).

2.2 Lifelong Learning

Lifelong learning or continual learning approaches can be grouped into regularization-based, data-based, and model-based approaches. Regularization based approaches (Kirkpatrick et al., 2017; Schwarz et al., 2018; Zenke et al., 2017) minimize significant changes to the previously learned representations as we update parameters for the current task. This is usually implemented as an additional constraint to the objective function based on the sensitivity of parameters. Recent studies (Kemker and Kanan, 2017; d'Autume et al., 2019; Lopez-Paz and Ranzato, 2017; Chaudhry et al., 2019) under data-based approaches store previous task data either using a replay memory buffer or a generative model. In NLP domain, lifelong language learning approaches have investigated the use of memory replay and local adaptation techniques (d'Autume et al., 2019).

Finally, model-based approaches allow models to allocate or grow capacity (layers or features) necessary for the tasks (Rusu et al., 2016; Lee et al., 2016). More recently, (Asghar et al., 2018) augmented RNNs with a progressive memory bank leading to increased model capacity. However, the challenges related to increased architectural complexity are tackled by hybrid models as in (Sodhani et al., 2020). In this paper, we build on ideas from the hybrid models and apply them for our learning task. While previous studies (Ding et al., 2019) have attempted to incorporate commonsense knowledge, this work is one of the first efforts to integrate multi-source knowledge and address it through the lens of incremental domain adaptation.

3 Problem Formalization

Formally, we assume that our learning framework has access to streams of social commonsense knowledge data obtained from n different domains, denoted by $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_n\}$. At a particular point in time, we extract knowledge from the current domain $\mathcal{D}_i$. We produce an embedding of social events by consolidating the accumulated knowledge across the modeled domains $\mathcal{D}_{\leq i}$. Data from each domain source contains source-specific textual descriptions of social situations and their intuitive commonsense information such as intents and emotions. Training samples, drawn from a domain dataset $\mathcal{D}_i$, could contain either a significant overlap or an entirely new set of knowledge when compared with the previously processed domains $\mathcal{D}_{1:i}$. Given such a setup, we aim to generate incrementally richer social event representations using our continual learning framework.

4 Datasets

For our representation learning task, we aggregate social commonsense knowledge data from various domain sources. This knowledge contains details about pragmatic aspects like intents and emotional reactions. We create a continual learning benchmark based on these commonsense data sources[1].

4.1 Lifelong Social Events Dataset

Different domain sources of social commonsense knowledge used for training our social event representation model are explained as follows.

ATOMIC dataset consists of inferential knowledge based on 24k short events covering a diverse range of everyday events and motivations. Though each event contains nine dimensions per event, the scope of this work will be limited to intent and emotions as our inferential pragmatic dimensions.

CONCEPTNET knowledge base contains several commonsense assertions. For our purpose, we choose ConceptNet's relevant relations: /r/MotivatedByGoal, /r/CausesDesire, /r/Entails, /r/Causes, /r/HasSubevent. We convert triples in the dataset into template form.

SB-SCK Since social roles (e.g., student, mother, teacher, worker, etc.) provide additional information about the motives and emotions behind actions specified in the events (as shown in Figure 1, 2, we adopt web-based knowledge mining techniques for capturing this aspect. This dataset was collected as a part of our recent work (Vijayaraghavan and Roy, 2021) using the following steps: (a) process texts from Reddit posts containing personal narratives as in (Vijayaraghavan and

---

[1] The project details about future data/code releases or any updates will be available at https://pralav.github.io/lifelong_eventrep?c=10

Search-based Social Commonsense Knowledge		
Social Roles	Event Phrases	Motives
Politicians	use social media	to woo voters
Activists		to create a movement
Police		to connect with residents and solve crime
Workers	gather around table	to solve business problems
Priests		to pray to god, share wine and bread
Friends		to share a meal, conversation

SB-SCK Dataset	
#events w/ motives	103,357
#events w/ emotions	69,584
#unique social roles	586

Figure 2: **Left**: Samples from Search-based Social Commonsense Knowledge (SB-SCK) dataset with highlighted motivations for social roles, **Right**: Statistics of SB-SCK dataset.

Roy, 2021), (b) extract propositions from text using OpenIE tools, (c) perform a web search for plausible intents and emotions by attaching purpose clauses (Palmer et al., 2005) and feelings lexical units from Framenet (Baker et al., 1998) and (d) finally, remove the poorly extracted facts using a simple classifier trained on some seed commonsense knowledge. Figure 2(Left) shows samples from this dataset indicating how the same action could have different social role related motivations. We refer to this as Search-based Social Commonsense Knowledge (SB-SCK) data. Figure 2(Right) presents the data statistics.

For data from each of the above domain sources, we sample free-form event text, its paraphrase, intent, emotional reactions, and negative samples of paraphrases, intents, and emotional reactions. Based on the annotated labels for motivation (Maslow's) and emotional reactions (Plutchik) in STORYCOMMONSENSE data, we run a simple K-Means clustering on the open text intent data. We identify five disjoint clusters on each of the three domains and map them to those categories. For the purpose of our lifelong learning problem, we divide each domain data into two sets (3 clusters and 2 clusters) and consider them as different sub-domains. Therefore, this results in 6 tasks in our continual learning setup. We refer to this dataset as *Lifelong EventRep Corpus*.

4.2 Paraphrase Datasets

We use random samples of parallel texts from paraphrase datasets like PARANMT-50M corpus(Wieting and Gimpel, 2017) and Quora Question Pair dataset [2]. These paraphrase datasets are primarily used for pretraining our model. We also produce paraphrases of free-form event texts in our dataset using a back-translation approach (Iyyer et al., 2018). We used pretrained English↔German translation models for this purpose.

5 Framework

Our goal is to learn distributed representations of social events by incorporating pragmatic aspects of the language beyond shallow event semantics. Moving away from conventional supervised multi-task classification based lifelong learning approaches, we focus on a lifelong representation learning approach that enables us to adapt and sequentially learn a social event embedding model. The motivation for a lifelong learning framework is that the growing knowledge obtained from various domain sources can effectively guide the modeling of complex social events. This involves systematically updating the model by consolidating this expanding knowledge to produce richer embeddings without forgetting previously accumulated knowledge. In this section, we will explain various components of our modeling framework.

5.1 Social Event Representation

Given an input event text description, the core idea is to first encode the free-form event text and decompose the ensuing representation into pragmatic (implied emotions and intents) and non-pragmatic (syntactic and semantic information) components. Eventually, we combine these decomposed representations to obtain an overall event representation and apply it in different downstream tasks.

5.1.1 Encoder

The input to our model is a free-form event text description from i^{th} domain, $x_j^{(i)} \in \mathcal{D}_i$. This free-form event text contains a sequence of tokens, $x_j^{(i)} = [w_1, w_2, ..., w_L]$, where each token $w_{(\cdot)}$ is obtained from an input vocabulary $\mathcal{V}$. The model encodes the input event text $x_j^{(i)} \in \mathcal{R}^{L \times d_X}$ in multiple steps. First, we construct a context-dependent token embedding using a context embedding function $\mathcal{G} : \mathcal{R}^{L \times d_X} \mapsto \mathcal{R}^{L \times d_H}$, where d_X and d_H refer to the embedding and hidden layer dimensions respectively. Following this encoding step, we incorporate pooling or projection function, $\mathcal{G}_{p\bar{p}}$: $\mathcal{R}^{L \times d_H} \mapsto \mathcal{R}^{3 \times d_H}$, that transform event text from context-dependent embedding space into pragmatic and semantic space. More specifically, we produce latent vectors for intents (h_I), reactions (h_R) and non-pragmatic (h_N) information. Finally, we com-

---

[2] https://www.kaggle.com/c/quora-question-pairs/data

bine the latent vectors h_N, h_I, h_R using a simple feed-forward layer, $\mathcal{G}_{\mathcal{C}}$: $\mathcal{R}^{3 \times d_H} \mapsto \mathcal{R}^{d_H}$, to produce a powerful social event representation, h_C, capable of dispelling apparent ambiguities in the true sense of their meaning. Given positive and negative examples of intents, emotional reactions and paraphrases associated with the input event text, we learn to effectively sharpen each of these embeddings h_I, h_R and using metric learning methods.

For the sake of brevity, we drop the domain index i and the sample index j in this section. These encoding steps are summarized as:

$$H_e = [h_1, h_2, ..., h_L] = \mathcal{G}([w_1, w_2, ..., w_L]) \quad (1)$$

$$h_I, h_R, h_N = \mathcal{G}_{p\bar{p}}(H_e) \quad (2)$$

$$h_C = \mathcal{G}_C(h_I, h_R, h_N) \quad (3)$$

We denote this multi-step encoding process resulting in h_I, h_R, h_C as a function $\mathcal{G}_{event}$. Now, we experiment with the following text embedding techniques as our context embedding function ($\mathcal{G}$):

BiGRU Using bidirectional GRUs (Chung et al., 2014), we compute the context embedding of the input event text by concatenating the forward ($\overrightarrow{h_t}$) and backward hidden states ($\overleftarrow{h_t}$), $\overleftrightarrow{h_t} = [\overrightarrow{h_t}; \overleftarrow{h_t}]$.

BERT We employ BERT (Devlin et al., 2018), a multi-layer bidirectional Transformer-based encoder, as our context embedding method $\mathcal{G}$. We fine-tune a BERT model that takes attribute-augmented event text $x = [CLS]\ m\ [SEP]\ w_1, ..., w_L\ [SEP]$ as input and outputs a powerful context-dependent event representation H_e. The attribute $m \in \{xIntent, xReact, xNprag\}$ refers to special tokens for intents, reactions and non-pragmatic aspects.

In our default case, our $\mathcal{G}_{p\bar{p}}$ function is the output embedding of $[CLS]$ token associated with their respective attribute-augmented input. In cases where input event text is not augmented with attribute special tokens, we apply pooling strategies such as attentive pooling (AP) and mean (MEAN) of all context vectors obtained from the previous encoding step $\mathcal{G}$. We obtain h_I, h_R, h_N based on these techniques. Depending on the type of context embedding function, we refer our multi-step event text encoder, $\mathcal{G}_{event}$, as EVENTGRU or EVENTBERT.

5.1.2 Objective Loss

Using positive $\{u_I^p, u_R^p, u_C^p\}$ and $N-1$ negative $\{u_I^n, u_R^n, u_C^n\}$ examples of intents, emotions and paraphrases associated with the event texts, we calculate N-pair loss, $\mathcal{L}_v(h, z^p, \{z_k^n\}_{k=1}^{N-1})$, to maximize the similarity between the representation of positive examples (z_v^p) and the computed embeddings (h_v). Here, z_v^e is computed using a transformation function f_v as: $z_v^e = f_v(u_v^e)$, where $v \in \{I, R, C\}$ and $e \in \{p, n\}$. Thus, our loss function is devised as:

$$\mathcal{L}_{\mathcal{T}} = \beta_D \cdot (\mathcal{L}_I + \mathcal{L}_R) + \beta_E \cdot \mathcal{L}_C \quad (4)$$

where $\mathcal{L}_I, \mathcal{L}_R$ are used to learn disentangled pragmatic embeddings (intent and emotion), $\mathcal{L}_C$ is intended to jointly embed semantic and pragmatic aspects to produce an overall social event representation. β_D, β_E are loss coefficients that weigh the importance of disentanglement loss and an overall joint embedding loss. These coefficients are non-negative and they sum to 1.

5.2 Continual Learning

Given a never-ending list of social events, once-and-for-all training on a fixed dataset limits the utility of such models in real-world applications. Therefore, we draw ideas from lifelong learning literature to adapt our models to new data yet retaining prior knowledge. First, we implement a data-based approach, which is a variant of episodic memory replay (EMR) (Lopez-Paz and Ranzato, 2017; Chaudhry et al., 2019; d'Autume et al., 2019) for mitigating catastrophic forgetting while allowing for beneficial backward knowledge transfer. Next, we combine it with simple vocabulary expansion for incremental domain adaptation.

5.2.1 Domain-Representative Episodic Memory Replay (DR-EMR)

We augment our model explained in Section 5.1 with an episodic memory module to perform sparse experience replay. As we train our lifelong representation learning model, we create mixed training mini-batches ($\mathcal{B}_{dom}, \mathcal{B}_{rep}$) by drawing samples from: (a) new domain dataset, $\mathcal{B}_{dom} \subset \mathcal{D}_i$ and (b) episodic memory containing old domain samples, $\mathcal{B}_{rep} \subset \mathcal{M}$. Training our model using mixed mini-batches introduces parameter changes that alter past event data embeddings, including those stored in our episodic memory, $\mathcal{M}$. The episodic memory module is implemented as a "Read-Write" memory store containing selective event samples from the original domain dataset and their respective embeddings. The key memory operations are stated as follows:

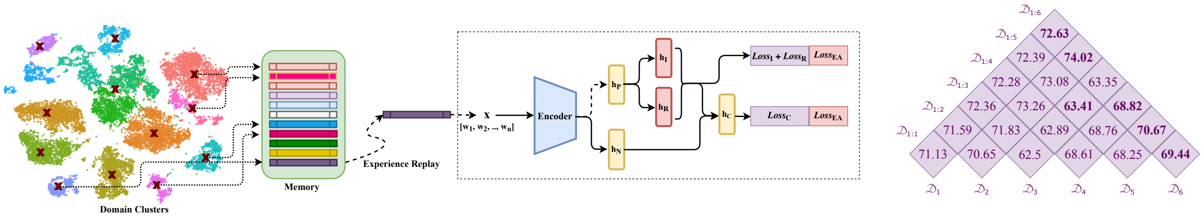


Figure 3: **Left:** Illustration of our Lifelong Social Event Representation Model, **Right:** Average accuracy score (%) of one specific permuted run across 6 domains. Table shows the dynamics of our continual learning model allowing positive backward transfer, i.e., performance increases in previous domains gradually with new knowledge.

Read Operation The read operation retrieves domain-specific random samples from our episodic memory for experience replay. These samples contain the original event training data and their representations. We choose samples from a previously trained domain at every read step assuming an overall uniform distribution over domains.

Write Operation In this work, we incorporate the following desirable characteristics of an episodic memory module: (a) $\mathcal{M}$ stores domain-representative samples that best approximate the domain's data distribution and (b) $\mathcal{M}$'s capacity is bounded or at most expands sub-linearly. To achieve this, we adopt the following strategies:

Domain-Representative Sample Selection: Past studies have explored using random writing strategy (d'Autume et al., 2019) and distribution approximation or K-Means to cluster the samples (Rebuffi et al., 2017). In our work, we perform sample selection using a CURE (Clustering Using REpresentatives) algorithm (Guha et al., 2001) to find C representative points. CURE employs hierarchical clustering that is computationally feasible by adopting random sampling and two-pass clustering. Using event representations obtained after embedding alignment transformation, we identify the domain-representative samples by computing euclidean metric-based nearest neighbors to the C representative points. We set the value of C based on the memory budget. These domain-representative samples are stored in our episodic memory with their corresponding representations.

Replacement Policy: When the memory becomes full, we follow a simple memory replacement policy that selects an existing memory entry to delete. Specifically, we replace the q^{th} memory entry which is determined by: $q = argmax_q(softmax(\phi(x_j^{(t)}) \cdot \mathcal{M}_q)$ similar to idea proposed in (Gulcehre et al., 2018).

Inspired by Wang et al. (Wang et al., 2019), we propose a variant of the alignment model that helps overcome catastrophic forgetting by ensuring minimal distortion to previously computed representational spaces. To accomplish this, we define simple linear transformations: $\mathcal{G}_{IA}, \mathcal{G}_{RA}, \mathcal{G}_{CA}$ on the top of the multi-step encoder function $\mathcal{G}_{event}$ outputs. For simplicity, we drop the subscripts (I, R, C) and denote these transformation functions as $\mathcal{G}_A$. Given a new domain i, we initialize our multi-step event encoder $\mathcal{G}_{event}^i$ and alignment function $\mathcal{G}_A^i$ with the last trained parameters as in $\mathcal{G}_{event}^{i-1}$ and $\mathcal{G}_A^{i-1}$ respectively. The optimization is implemented in the following two steps: (a) For samples from $(\mathcal{B}_{dom}, \mathcal{B}_{rep})$, we optimize the encoder output representations to be closer to their respective ground-truth examples in this linearly transformed space. Therefore, we modify the N-pair loss function to accommodate the alignment function as: $\mathcal{L}(\mathcal{G}_A^i(\mathcal{G}_{event}^i(x)),\ \mathcal{G}_A^i(z^p),\ \{\mathcal{G}_A^i(z^n)\}_{k=1}^{N-1})$;
(b) For samples from $\mathcal{B}_{rep}$, we add an extra constraint $(\mathcal{L} + \mathcal{L}_{EA})$ for each embedding component to align old and new domain embedding spaces:

$$\mathcal{L}_{EA} = ||\mathcal{G}_A^i(\mathcal{G}_{event}^i(x)) - \mathcal{G}_A^{i-1}(\mathcal{G}_{event}^{i-1}(x))||^2 \quad (5)$$

6 Training

Since our model involves metric learning, hard negative data mining is an essential step for faster convergence and improved discriminative capabilities. However, selecting too hard examples too often makes the training unstable. Therefore, we choose a hybrid negative mining technique where we choose few semi-hard negatives examples (Hermans et al., 2017) and combine it with random negative samples to effectively train our model.

In our work, we define a heuristic objective by weighing samples based on two factors: (i) word overlap or similarity in embedding space of the event text and (ii) intent and emotion free-form text or categories based on STORYCOMMONSENSE data. More specifically, given an event text as anchor and a positive intent text based on a ground

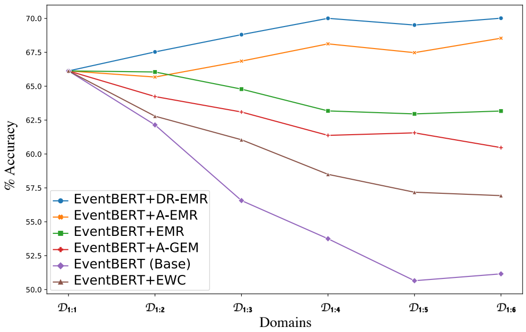


Figure 4: Average accuracy scores (%) of data after 6 permuted runs from domains that have been observed at that point during the continual learning process.

truth motivation category, we mine negative instances for intent as follows: (a) choose random text samples associated with a motivation category that is different from that of the positive example but closer in the embedding space or word overlap, (b) choose random text samples within the same motivation category but with different emotion category. We repeat this process for drawing negative instances related to emotions. For paraphrases, we consider few examples with significant word overlap while the rest are randomly chosen examples. N-pair loss helps alleviate the sensitivity of triplet loss function to the choice of hard triplets. Finally, we pre-train our model with paraphrase data and fine-tune it using the examples obtained from hard negative mining for intents, emotions and paraphrases. For our training, the learning rate is set to 0.0001, the number of training epoch is 20. By default, we use EVENTBERT as our multi-step encoder. We conduct a study by assigning different values for loss coefficients, β_D, β_E, and explain the results of the study in Section 7.1.2.

7 Experiments

We conduct several experiments to study the power of our learned embeddings. Our experiments are designed to answer the following questions:

RQ1: How well does our model perform in comparison to other continual learning approaches for intent-emotion prediction?

RQ2: To what extent do our modeling choices impact the results in predicting intents & emotions?

RQ3: Does our model outperform existing state-of-the-art methods in hard similarity task that evaluates the effectiveness of the learned embeddings?

RQ4: Do the learned embeddings demonstrate transfer capability to downstream tasks – Paraphrase detection & Social IQA reasoning?

7.1 Intent-Emotion Prediction (RQ1)

The continual learning methods evaluated using our *Lifelong EventRep Corpus* are given below.

- **Base**: We simply fine-tune our model on successive tasks from previously trained checkpoint.
- **A-GEM**: This method (Chaudhry et al., 2019) uses a constraint that enables the projected gradient to decrease the loss on older tasks. We randomly choose 2-3% samples from all the previous tasks to form a constraint.
- **EWC**: A regularization-based technique, Online-EWC (Schwarz et al., 2018), is used to address catastrophic forgetting.
- **EMR**: We use randomly stored examples for sparse experience replay.
- **A-EMR**: A variant of our model that uses random samples for experience replay with alignment constraints.
- **DR-EMR**: This is our complete model involving domain representative experience replay and alignment constraints.

7.1.1 Empirical Results

After running six permuted sequence of tasks, we calculate the mean performance on the test set of all observed task domains after time step k, given by $AvgAcc = \frac{1}{k}\sum_{i=1}^{k} Acc^{(i)}$, where $Acc^{(i)}$ is the model accuracy on the test set from the domain $\mathcal{D}_i$. Further, we also compute standard deviation to determine the importance of the order of the tasks during training. Table 1a contains the comparison of last-step $AvgAcc$ scores and standard deviation for predicting intents and emotions. Figure 4 plots the $AvgAcc$ at every step where $\mathcal{D}_{1:i}$ indicates that the model has seen data from i domains to evaluate our continual learning process.

In the absence of any lifelong learning strategies, the performance drops significantly for *Base* model while emphasizing the importance of task order as indicated by the high standard deviation value. Compared to the *Base* model, our results show some improvement in intent and emotion prediction as we introduce methods like *A-GEM* and *EWC*. However, we observe a significant performance gain with *EMR*-based techniques. Our complete model (*DR-EMR*) outperforms all the other methods, thereby demonstrating the importance of

Models	Intents		Emotions	
	AvgAcc	Std	AvgAcc	Std
EVENTBERT (Base)	51.16	10.41	62.27	9.74
EVENTBERT + A-GEM	60.47	6.75	69.44	6.12
EVENTBERT + EWC	56.93	8.86	66.89	7.55
EVENTBERT + EMR	63.17	4.20	72.04	4.66
EVENTBERT + A-EMR	68.54	2.85	73.42	3.10
EVENTBERT+ DR-EMR	**70.02**	**0.38**	**78.48**	**0.65**

(a) Lifelong EventRep Dataset

Models	% Acc.
KGEB (Ding et al., 2016)	50.09
NTN + Int+ (Ding et al., 2019)	58.83
NTN + Int + Senti (Ding et al., 2019)	64.31
EVENTBERT$_{\beta_E=0.3}$+ DR-EMR	66.19
EVENTBERT$_{\beta_E=0.5}$+ DR-EMR	**71.23**
EVENTBERT$_{\beta_E=0.7}$+ DR-EMR	69.79

(b) Hard Similarity Dataset

Table 1: Evaluation results on: (a) held-out Lifelong EventRep test set and (b) combined hard similarity dataset.

domain-representative sampling and alignment constraints towards learning representations that help effective prediction of intents and emotions. Moreover, we assess the domain sequences that cause a performance drop. For example, whenever SB-SCK is trained at the end, the model shows reduced accuracy. The reason can be ascribed to the effect of interference of noisy knowledge over the previously trained cleaner domains. Training this noisy source ahead leads to positive knowledge transfer and hence produces sharpened performance outcomes. Despite these odds, our *DM-EMR* model records the least standard deviation implying reduced sensitivity to training order. Figure 3 (Right) shows the dynamics of our continual learning mode. It contains accuracy scores of a single run and displays the lower-triangular values. We see that our approach allows positive backward transfer, i.e., model performance in previous domains gradually increases with new domain knowledge. Our model achieves the best performance (see the bold-faced accuracy scores in Figure 3(Right)) in most domains after observing all the data ($\mathcal{D}_{1:6}$).

7.1.2 Ablation Study (RQ2)

We analyze different model configurations related to: (a) encoding: EVENTGRU, EVENTBERT, (b) pooling: attribute-augmented input (CLS), Mean Pooling (MP) and Attentive Pooling (AP) and (c) sampling: K-Means, CURE. Results of the study are given in Figure 5(Left). We evaluated different combinations of these strategies but only report the average accuracy scores of configurations having the best strategy at each category combined with the variants in the following category, i.e., for pooling strategies, we only report scores with EVENTBERT encoding strategy and so on. From the results, we ascertain that the best configuration comprises an EVENTBERT encoder supported by attentive pooling and CURE-based sample selection.

Methods	Intents	Emotions
Encoding Strategy		
EVENTGRU	60.92	70.61
EVENTBERT	**68.56**	**78.48**
Pooling Strategy		
MP	65.50	76.13
AP	67.28	76.69
CLS	**68.56**	**78.48**
Sampling Strategy		
K-Means	66.45	75.88
CURE	**68.56**	**78.48**

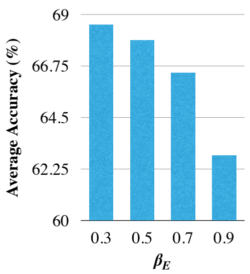


Figure 5: Results of our ablation study on a held-out validation set. **Left:** ***AvgAcc*** scores (%) on intent-emotion prediction task using different encoding, pooling & sampling strategies. **Right:** ***AvgAcc*** scores (%) to measure the effect of β_E in predicting intents.

Additionally, we measure the effect of β_E in the prediction of intents. As shown in Figure 5 (Right), the model performs significantly better for lower values β_E as more weight is assigned for the disentanglement of pragmatic aspects. However, we observe that a balanced loss function with $\beta_E = 0.5$ allows for consistently good performance in both intent-emotion prediction (Figure 5 (Right)) and hard similarity tasks (see Section 7.2). Despite other hyperparameters, changes to β_E determine the importance of incorporating semantic or pragmatic information in the ensuing event embedding.

7.2 Hard Similarity Task (RQ3)

By following the work of Ding et al. (Ding et al., 2019), we evaluate our social event representation on an extended dataset of event pairs containing: (a) similar event pair having minimum lexical overlap (e.g., people admired president / citizens loved leader) (b) dissimilar event pair with high lexical overlap (e.g., people admired president / people admired nature). A good performance in this task will ensure that similar events are pulled closer to each other than dissimilar events. Combining hard similarity datasets from (Ding et al., 2019) and (Weber et al., 2018), the total size of this balanced dataset is 2,230 event pairs. Using our joint embedding h_C for an event text and triplet loss setup, we compute a similarity score between similar and dissimilar pairs. The baselines include: Knowledge-graph based embedding model (KGEB) (Ding et al., 2016), Neural Tensor Network (NTN) and its variants augmented with ATOMIC dataset based embeddings (Int, Senti) (Ding et al., 2019). We report the model's accuracy in assigning a higher similarity score for similar pairs than dissimilar pairs. Table 1b shows that our model outperforms the state-of-the-art method for this task.

7.3 Paraphrase Detection (RQ4)

Given a sentence pair, the objective is to detect whether they are paraphrases or not. For each sentence pair (s_1, s_2), we pass them through our model and obtain their respective h_C, given by vectors (u, v). We concatenate these vectors (u, v) with the element-wise difference $|u - v|$ and feed to a feed-forward layer. We optimize binary cross-entropy loss. For evaluation purposes, we compare our model against baselines like BERT and ESIM (Chen et al., 2016). Trained on a subset of dataset explained in Section 4.2, we choose an out-of-domain test dataset where samples stem from a dissimilar input distribution. To this end, Twitter URL paraphrasing corpus (Lan et al., 2017), referred to as TwitterPPDB, is selected. Table 2b contains results of our evaluation. The results testify the efficacy of our embeddings.

7.4 Social IQA Reasoning (RQ4)

We determine the quality of our latent social event representations by evaluating on a social commonsense reasoning benchmark – SocialIQA dataset (Sap et al., 2019b). Given a context, a question and three candidate answers, the goal is to select the right answer among the candidates. Following Sap et al.(Sap et al., 2019b), the context, question, and candidate answer are concatenated using separator tokens and passed to the BERT model. Additionally, we feed the context to our EventBERT model to obtain three embeddings h_I, h_R, h_C. While the original work computed a score l using the hidden state of $[CLS]$ token, we introduce a minor modification to this step as: $l = W_5 tanh(W_1 h_{CLS} + W_2 h_I + W_3 h_R + W_4 h_C)$, where $W_{1:4} \in \mathcal{R}^{d_H \times d_H}$ and $W_5 \in \mathcal{R}^{1 \times d_H}$ are learnable parameters. Similar to (Sap et al., 2019b), triple with the highest normalized score is used as the model's prediction. We fine-tune BERT models using our new scoring function with social event embedding (denoted as "w/") and compare against baselines (like GPT (Radford et al., 2018)) without our event embeddings (denoted as "w/o"). Results in Table 2a indicate that a simple enhancement procedure at the penultimate step can offer significant performance gains. Our findings also suggest that our enhanced model performed well for question types like 'wants' and 'effects' that weren't explicitly modeled in our embedding model. This confirms that our pragmatics-enriched embeddings lead to improved reasoning capabilities.

Models	**Dev**	**Test**
w/o Social Event Embeddings		
GPT	63.3	63.0
BERT-base	63.3	63.1
BERT-large	**66.0**	**64.5**
w/ Social Event Embeddings		
BERT-base	65.1	64.0
BERT-large	**68.7**	**67.9**

(a) SocialIQA Dataset

Models	**% Acc**
ESIM	84.01
BERT	87.63
EventBERT$_{0.5}$	88.23
EventBERT$_{0.7}$	**90.16**

(b) TwitterPPDB dataset

Table 2: Accuracy scores (%) of different models on: (a) SocialIQA dev & test set, and (b) Twitter URL Paraphrasing corpus, TwitterPPDB. EventBERT models employ DR-EMR and subscript indicates value of β_E.

8 Conclusion

Humans rely upon commonsense knowledge about social contexts to ascribe meaning to everyday events. In this paper, we introduce a lifelong learning approach to effectively embed social events with the help of a growing set of social commonsense knowledge assertions acquired from different domains. First, we leverage social commonsense knowledge to sharpen social event embeddings with semantic and pragmatic attributes. Next, we employ domain-representative episodic memory replay (DR-EMR) to overcome catastrophic forgetting and enable positive knowledge transfer with the emergence of new domain knowledge. By evaluating on a corpus of social events aggregated from multiple sources, we establish that our model is able to outperform several baselines. Experimental results on downstream tasks like event similarity, reasoning, and paraphrase detection demonstrate the capabilities of our social event embeddings. We hope that our work will motivate further exploration into lifelong representation learning of social events and advance the research in inferring pragmatic dimensions from texts.

References

Catherine Adams, Janet Baxendale, Julian Lloyd, and Catherine Aldred. 2005. Pragmatic language impairment: case studies of social and pragmatic language therapy. *Child Language Teaching and Therapy*, 21(3):227–250.

Nabiha Asghar, Lili Mou, Kira A Selby, Kevin D Pantasdo, Pascal Poupart, and Xin Jiang. 2018. Progressive memory banks for incremental domain adaptation. *arXiv preprint arXiv:1811.00239*.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90.

Niranjan Balasubramanian, Stephen Soderland, Oren Etzioni, et al. 2013. Generating coherent event schemas at scale. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1721–1731.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 602–610. Association for Computational Linguistics.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. 2019. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Pengxiang Cheng and Katrin Erk. 2018. Implicit argument prediction with event knowledge. *arXiv preprint arXiv:1802.07226*.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. *arXiv preprint arXiv:1302.4813*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. *arXiv preprint arXiv:1906.01076*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. *arXiv preprint arXiv:1909.05190*.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2014. Using structured events to predict stock price movement: An empirical investigation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1415–1425.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2016. Knowledge-driven event embedding for stock prediction. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers*, pages 2133–2142.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. 2001. Cure: an efficient clustering algorithm for large databases. *Information systems*, 26(1):35–58.

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. 2018. Dynamic neural turing machine with continuous and discrete addressing schemes. *Neural computation*, 30(4):857–884.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In defense of the triplet loss for person reidentification. *arXiv preprint arXiv:1703.07737*.

Robert Hopper and Rita C Naremore. 1978. *Children's speech: A practical introduction to communication development*. HarperCollins Publishers.

Linmei Hu, Juanzi Li, Liqiang Nie, Xiao-Li Li, and Chao Shao. 2017. What happens next? future subevent prediction using contextual hierarchical lstm. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the*

European Chapter of the Association for Computational Linguistics, pages 336–344. Association for Computational Linguistics.

Ronald Kemker and Christopher Kanan. 2017. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. *arXiv preprint arXiv:1708.00391*.

Sang-Woo Lee, Chung-Yeon Lee, Dong-Hyun Kwak, Jiwon Kim, Jeonghee Kim, and Byoung-Tak Zhang. 2016. Dual-memory deep learning architectures for lifelong learning of everyday human behaviors. In *IJCAI*, pages 1669–1675.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476.

Ashutosh Modi and Ivan Titov. 2013. Learning semantic script knowledge with event embeddings. *arXiv preprint arXiv:1312.5198*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Lauren Parsons, Reinie Cordier, Natalie Munro, Annette Joosten, and Renee Speyer. 2017. A systematic review of pragmatic language interventions for children with autism spectrum disorder. *PloS one*, 12(4):e0172242.

Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A Smith, and Yejin Choi. 2018. Event2mind: Commonsense inference on events, intents, and reactions. *arXiv preprint arXiv:1805.06939*.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019b. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. 2018. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*.

Shagun Sodhani, Sarath Chandar, and Yoshua Bengio. 2020. Toward training recurrent neural networks for lifelong learning. *Neural computation*, 32(1):1–35.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ottokar Tilk, Vera Demberg, Asad Sayeed, Dietrich Klakow, and Stefan Thater. 2016. Event participant modelling with neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 171–182.

Prashanth Vijayaraghavan and Deb Roy. 2021. Modeling human motives and emotions from personal narratives using external knowledge and entity tracking. In *Proceedings of The Web Conference 2021*.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. *arXiv preprint arXiv:1903.02588*.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018. Event representations with tensor-based compositions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

John Wieting and Kevin Gimpel. 2017. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*.

Barbara S Wood. 1976. Children and communication: Verbal and nonverbal language development.

Markus Wulfmeier, Alex Bewley, and Ingmar Posner. 2018. Incremental adversarial domain adaptation for continually changing environments. In *2018*

IEEE International conference on robotics and automation (ICRA), pages 1–9. IEEE.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3987–3995. JMLR. org.

GLaRA: Graph-based Labeling Rule Augmentation for Weakly Supervised Named Entity Recognition

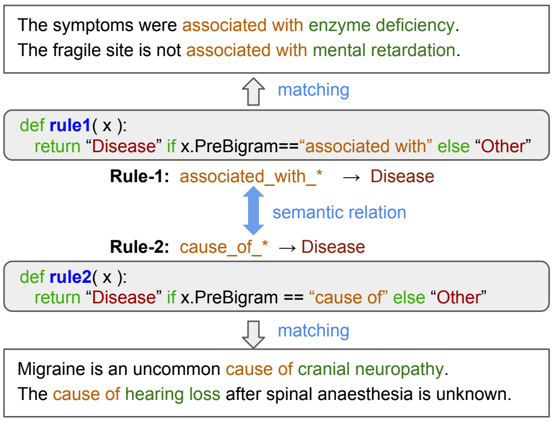
Xinyan Zhao[*] **Haibo Ding**[†] **Zhe Feng**[†]
[*]School of Information, University of Michigan, MI 48109, USA
[†] Bosch Research and Technology Center, Sunnyvale, CA 94085, USA
`zhaoxy@umich.edu, {haibo.ding,zhe.feng2}@us.bosch.com`

Abstract

Instead of using expensive manual annotations, researchers have proposed to train named entity recognition (NER) systems using heuristic labeling rules. However, devising labeling rules is challenging because it often requires a considerable amount of manual effort and domain expertise. To alleviate this problem, we propose GLaRA, a graph-based labeling rule augmentation framework, to learn new labeling rules from unlabeled data. We first create a graph with nodes representing candidate rules extracted from unlabeled data. Then, we design a new graph neural network to augment labeling rules by exploring the semantic relations between rules. We finally apply the augmented rules on unlabeled data to generate weak labels and train a NER model using the weakly labeled data. We evaluate our method on three NER datasets and find that we can achieve an average improvement of +20% F1 score over the best baseline when given a small set of seed rules.

1 Introduction

Named entity recognition (NER) models often need to be trained with many manual labels to perform well. Due to the high cost of manual annotations, collecting labeled data to train NER models is challenging for real-world applications. Recently, researchers have proposed to collect weak labels using heuristic rules, which are called labeling rules (Bach et al., 2017; Fries et al., 2017; Ratner et al., 2020; Safranchik et al., 2020). This kind of methods typically first ask domain experts to write labeling rules for a NER task, then use these manual rules to generate labeled data and train a NER model with the weakly labeled data. The advantage of these methods is that they do not require manual annotations. However, during our study, we find that writing labeling rules is also challenging for domain-specific tasks. Devising accurate rules often demands a significant amount of manual effort because it requires developers to have deep domain expertise and a thorough understanding of the target data.



The symptoms were associated with enzyme deficiency.
The fragile site is not associated with mental retardation.

```
def rule1( x ):
    return “Disease” if x.PreBigram==“associated with” else “Other”
```

Rule-1: associated_with_* → Disease

Rule-2: cause_of_* → Disease

```
def rule2( x ):
    return “Disease” if x.PreBigram == “cause of” else “Other”
```

Migraine is an uncommon cause of cranial neuropathy.
The cause of hearing loss after spinal anaesthesia is unknown.

Figure 1: Two examples of rules for recognizing `Diseases`. `PreBigram` means the 2 tokens on the left of a candidate entity.

To alleviate manual effort on writing labeling rules, we propose GLaRA, a **G**raph-based **La**beling **R**ule **A**ugmentation framework to automatically learn new rules from unlabeled data with a handful of seed rules. Our work is motivated by the intuition that if two rules can accurately label the same type of entities, then they are semantically related via the entities matched by them. Therefore, we can acquire new labeling rules based on their semantic relatedness with the rules that we have already known. For example, Figure 1 shows two example rules for labeling `Disease` entities. If we know that `rule1` “`associated with` $*\rightarrow$`Disease`” is an accurate rule for labeling diseases, and `rule1` is semantically related to `rule2` “`cause of` $*\rightarrow$`Disease`”, then we can derive that `rule2` can be another accurate rule for labeling diseases.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3636–3649
April 19 - 23, 2021.

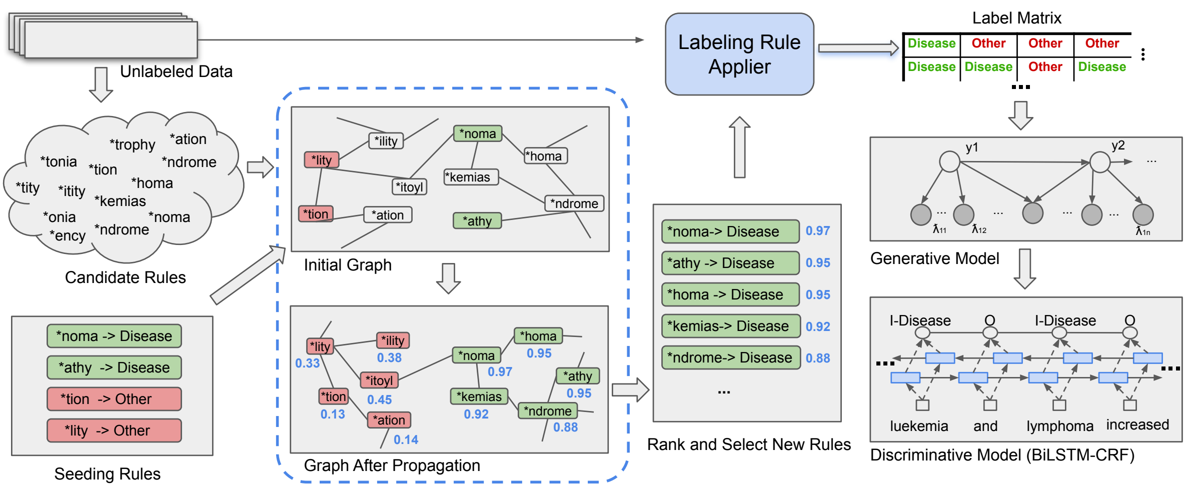


Figure 2: An example workflow of GLaRA framework that using `suffix rules` to recognize `Diseases`. Given unlabeled data and seeding rules, (1) we first extract candidate rules from unlabeled data, (2) then build a graph of rules and learn new rules by propagating the labeling information from seeding rules to other rules. (3) Next we apply selected rules on unlabeled data and obtain a label matrix. (4) Finally, we estimate the noisy labels using a generative model, and (5) train a final NER model with noisy labels. "$*$`noma` → `Disease`" denotes that if a word's suffix is "`noma`" then it will be labeled as `Disease`.

To augment labeling rules, we first define six types of rules and extract all possible rules from unlabeled data as candidate rules. Then, for each rule type, we build a graph by connecting rules of this type based on their semantic similarities. In our work, we compute a rule's embedding as the average of contextual embeddings of entities matched by the rule, and compute the similarities between rules using their embeddings. We learn new rules using a graph neural network model from a small set of seed rules. Next, we train a discriminative NER model using the weak labels generated by both the seeding and learned rules. To obtain weak training labels, we first obtain a label matrix by applying all the augmented rules on each token in unlabeled data. Then, we estimate the labels of unlabeled instances using the LinkedHMM model (Safranchik et al., 2020). We evaluate our framework on three datasets. In our experiments, we first show that our method can achieve better results than baselines when abundant rules are available. We also demonstrate that we can achieve an average improvement of +20% F1 when only a small set of rules are available.

We summarize our major contributions as:

- We propose a new Graph-based Rule Labeling Rule Augmentation (GLaRA)[1] framework, which can effectively learn new labeling rules from unlabeled data automatically.

- We propose a new graph neural network to estimate rules' labeling confidence with a new class distance-based loss function.

- We define six types of labeling rules, which have been proven to be effective on three named entity recognition tasks.

2 The GLaRA Framework

Our goal is to build a NER system with a small set of manually selected seeding rules and unlabeled data. Our key idea is to first augment labeling rules using graph neural networks, based on the hypothesis that semantically similar rules should have similar abilities to recognize entities. Then, we train a NER model using the weak training data labeled by the augmented rules.

Overview Figure 2 shows an example workflow of GLaRA framework using `suffix` rules to recognize `Disease` entities. Our framework consists of five major components. (1) *Rule extractor*: We define six types of rules and extract all possible rules from unlabeled data as candidate rules. (2) *Rule augmentation*: For each rule type, we first build a graph of rules by connecting rules based on their semantic similarities. Given a small set of

---

[1] The code is available at `https://github.com/zhaoxy92/GLaRA`.

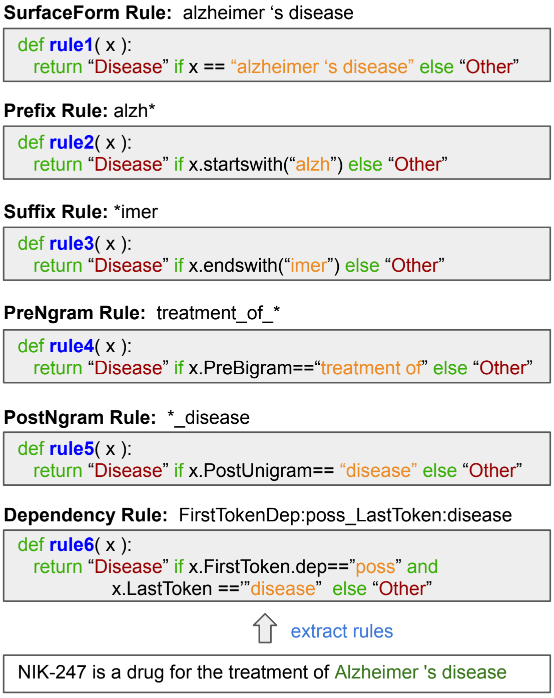
manual seeding rules, we learn new rules by propagating the labeling confidence from seeding rules to other rules. (3) *Rule Applier*: We obtain a label matrix by applying all the augmented rules on each token in the unlabeled data. (4) *Generative Model*: We estimate the labels of unlabeled instances using a generative model. (5) *Discriminative Model*: We train a final discriminative NER model using the weak labels produced by the generative model.

2.1 Candidate Rule Extraction

As demonstrated in previous work on NER (Zhou and Su, 2002), lexical and contextual clues are strong indicators for entity recognition. Therefore, we define and extract six types of rules: `SurfaceForm`, `Prefix`, `Suffix`, `PreNgram`, `PostNgram`, and `DependencyRule` to recognize entities by considering their lexical, contextual, and syntax information.

Given an unlabeled sentence, we first extract all noun phrases (NPs) using a set of Part-of-Speech (POS) patterns, as candidate entity mentions. The POS patterns include "JJ? NN+" (JJ denotes an adjective, and NN denotes a noun) and top 15 most frequent POS patterns of the entity mentions in the development sets. Then, we extract all six types of rules from the unlabeled data as candidate rules for each candidate entity mention. Specifically, we extract the surface form of each candidate entity mention as a `SufraceForm` rule. If the mention is a single token, we extract its first and last m characters as `Prefix` and `Suffix` rules, respectively. For `PreNgram` rule, we extract leading n words of a candidate entity as `inclusive PreNgram` rule; meanwhile, we also extract n words on the left of candidate as `exclusive PreNgram` rule. `PostNgram` rules are created similarly from the right context. Also, for each multi-token entity candidate, we first extract the dependency relations of the first token and the second last token, respectively, and then each dependency is combined with the last token as `Dependency` rules of this mention. We treat m and n as hyperparameters in our work.

Figure 3 show some example rules that extracted for the candidate entity mention "*Alzheimer 's disease*" and how they are used for labeling entities.



SurfaceForm Rule: alzheimer 's disease

```
def rule1( x ):
  return "Disease" if x == "alzheimer 's disease" else "Other"
```

Prefix Rule: alzh*

```
def rule2( x ):
  return "Disease" if x.startswith("alzh") else "Other"
```

Suffix Rule: *imer

```
def rule3( x ):
  return "Disease" if x.endswith("imer") else "Other"
```

PreNgram Rule: treatment_of_*

```
def rule4( x ):
  return "Disease" if x.PreBigram=="treatment of" else "Other"
```

PostNgram Rule: *_disease

```
def rule5( x ):
  return "Disease" if x.PostUnigram== "disease" else "Other"
```

Dependency Rule: FirstTokenDep:poss_LastToken:disease

```
def rule6( x ):
  return "Disease" if x.FirstToken.dep=="poss" and
            x.LastToken =="disease"  else "Other"
```

NIK-247 is a drug for the treatment of Alzheimer 's disease

Figure 3: Example rules extracted for an entity mention, and how these rules will be used to label entities.

2.2 Labeling Rule Augmentation

We aim to learn new labeling rules from seed rules by exploiting the semantic relations between rules. Specifically, we first build a graph with rules as nodes. Then, we initialize the graph with both manually selected positive and negative seeding rules. Positive seeding rules are those that can be used to predict a target entity type. Negative ones are used to predict instances of the "*Other*" class. Next, we estimate the representations of rules by optimizing a graph neural network model. Finally, we compute the distances of a rule to the centroids of positive and negative seeding rules, respectively, and then select rules close to positive centroid as new labeling rules.

Graph of Rules For each type rules, we create a graph $G = (\mathcal{V}_u, \mathcal{V}_s^{pos}, \mathcal{V}_s^{neg}, \mathcal{A})$ with this type of rules as nodes, where $\mathcal{V}_u$ denotes the candidate rules extracted in the previous step, $\mathcal{V}_s^{pos}$ and $\mathcal{V}_s^{neg}$ denotes the positive and negative seeding rules, respectively. $\mathcal{A}$ is the adjacency matrix of nodes. In our graph, each node (i.e., rule) is connected with the top 10 semantically similar nodes. The similarity between two rules is computed as the cosine similarity using their embeddings.

Rule Embeddings We estimate semantic relatedness between rules with their embeddings, which are computed using pre-trained contextual embedding models. Specifically, we first apply the pre-trained ELMo (Peters et al., 2018) model on all unlabeled sentences to obtain contextual embeddings of candidate entity mentions. Then, we compute the embedding of a rule as the average of the embeddings of all the candidate entities that can be matched by this rule. For example, the embedding of the prefix rule "*noma" is calculated as the average of the embeddings of all candidate mentions that ends with "noma".

Graph Propagation Model Given a graph of rules and seeding rules, we formulate the problem of learning new labeling rules (i.e., positive rules) as a graph-based semi-supervised node classification task that aims to classify candidate rules (which could be treated as unlabeled nodes in the graph) as positive or negative.

Based on the intuition that semantically similar rules should predict entity labels similarly. We propose a graph neural network model to propagate labeling information from seeding nodes to other nodes based on the recent work on Graph Attention Network (Veličković et al., 2017). Specifically, given the input embedding h_i of node i and its neighbors $\mathcal{N}_i$, we first compute an attention weight for each connected pair (i, j) as,

$$\alpha_{ij} = \frac{exp(f(A^T[Wh_i, Wh_j]))}{\sum_{k \in N_i} exp(f(A^T[Wh_i, Wh_k]))} \tag{1}$$

where W is a parameter matrix, and f is the LeakyReLU activation function. We then recompute the embedding of i:

$$h_i^* = \alpha_{i,i} Wh_i + \sum_{j \in N_i} Wh_j \tag{2}$$

To keep model stable, we apply multi-head attention mechanism to obtain K attentional states for each node, the average of which is used as final node representation, i.e., $h_i^{*T} = \frac{1}{K} \sum_k h_i^{kT}$.

The objective of our model is defined as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{sup} + \mathcal{L}_{reg} + \mathcal{L}_{dist} \tag{3}$$

where $\mathcal{L}_{sup} = -(y_i \log(p_i)) + (1 - y_i) \log(1 - p_i)$
$\mathcal{L}_{reg} = \sum_{i,j \in N_i} \|h_i - h_j\|_2$
$\mathcal{L}_{dist} = dist(h_{pos}, h_{neg})$

where $\mathcal{L}_{sup}$ is the supervision loss computed on both positive and negative seeding rule nodes, $\mathcal{L}_{reg}$ is the regularization that encourages connected nodes to share similar representations, and $\mathcal{L}_{dist}$ aims to maximize the distance between positive and negative seeding nodes. $dist(\cdot)$ computes the cosine similarity between the centroids of the positive and negative seeds. p_i is the probability of a node being classified as positive, and h_{pos} and h_{neg} are the average embeddings of positive and negative nodes, respectively.

When the learning process is finished, each rule is associated with a new embedding representation, i.e., h_i^{*T}. For each rule, we first compute its cosine distances to the centroids of positive and negative seeding nodes using their embeddings, respectively. Then, we rank all rules by the difference of two distances and select the top M rules closest to the centroid of the positive seeding rules as new labeling rules.

2.3 Generative Model for Label Estimation

After the rule learning process, we apply both the newly learned rules and the seeding rules on unlabeled data to produce a matrix of labels.[2] Since one token can be potentially matched by several different rules, the resulting labels can have conflicts. Therefore, we use the LinkedHMM (Safranchik et al., 2020) model to combine these labels into one label for each token. Briefly, the main idea of LinkHMM is to treat the true label of a token as a latent random variable and estimate its value by relating it to the label outputs from different labeling rules. The estimated labels can be used to train final discriminative NER models.

2.4 Discriminative NER Model

After the training of the generative model (i.e., the LinkedHMM in our work) is completed, each token in the unlabeled data is associated with a weak label. Each weak label is a probability distribution over all entity classes, which can be used to train a discriminative NER model. One advantage of training a discriminative NER model is that it can use other token features while the generative model can only use labeling rules' outputs as inputs. Therefore, even if a token is not matched by any labeling rules, it can still be predicted correctly by the discriminative model.

In our work, we use BiLSTM-CRF (Huang et al., 2015) as our discriminative model. The model first uses BiLSTM to generate a state representation for each token in a sequence. The CRF layer then predicts each token by maximizing the expected likelihood of the entire sequence based on the estimated labels.

---

[2] In our work, we also applied the linking rules developed by (Safranchik et al., 2020) on unlabeled data, so the label matrix also contains the results from linking rules.

3 Experimental Setup

In this section, we present the details of our experimental setup. First, we evaluate our method on three NER datasets, which we refer to `NCBI`, `BC5CDR`, and `LaptopReview`. Then, we describe the baseline methods compared in our experiments. We also give a detailed description of the seed rules used in our experiments.

3.1 Datasets

We evaluated our method on three datasets. Details of each dataset are described below.

`NCBI` (Doğan et al., 2014) contains 793 PubMed abstracts with 6,892 `Disease` mentions and is split into 593 train, 100 dev and 100 test data.

`BC5CDR` (Li et al., 2016) has 1,500 PubMed articles with 5,818 `Disease` and 3,116 `Chemical` mentions. It is split into train, dev, and test sets with 500 articles each.

`LaptopReview` (Pontiki et al., 2016) contains sentences regarding laptop `AspectTerms`, including 3,048 training and 800 test sentences. Following (Safranchik et al., 2020), we hold out 20% of training data as dev set.

Note that we use all training data as our unlabeled data by removing the manual annotations.

3.2 Compared Methods

We compare our method with state-of-the-art weakly supervised methods using dictionaries or heuristic rules as supervision. In this section, we briefly describe these baseline methods.

AutoNER (Shang et al., 2018) is a distantly supervised method, which automatically builds a neural named entity recognition model using dictionaries as weak supervision.

Snorkel (Ratner et al., 2020) is a general machine learning framework that can train classifiers using heuristic rules. By default, it uses a Naive Bayes generative model to denoise labeling rules by predicting each token's label independently.

SwellShark (Fries et al., 2017) is an extension of Snorkel that was developed for biomedical NER. Same as Snorkel, it uses a naive Bayes generative model to denoise manual labeling rules. It also requires a special entity candidate generator to detect entity spans accurately before predicting their entity labels. In our experiments, we report both results using simple noun phrases as candidates, and that generated using extra expert effort.

LinkedHMM (Safranchik et al., 2020) is a framework for training sequence tagging models using weak supervision from manual rules. Besides using labeling rules, it can also use linking rules that indicate whether two consecutive tokens have the same label.

RuleType	NCBI		BC5CDR (Disease)		BC5CDR (Chem)		Laptop	
	S	M	S	M	S	M	S	M
Positive seeding rules								
Surface	875	0	632K	0	1.5M	0	274	0
Suffix	12	5	14	2	7	9	0	5
Prefix	0	7	2	13	0	8	0	5
PreNgram†	5	9	0	9	1	8	6	5
PostNgram†	4	6	5	10	2	9	0	5
DepRule	5	8	5	7	0	6	0	4
Negative seeding rules								
Surface	34	0	34	6	34	0	34	3
Suffix	0	10	0	18	0	22	0	5
Prefix	0	10	0	12	0	8	0	8
PreNgram†	0	15	0	18	0	16	0	9
PostNgram†	0	10	0	19	0	10	0	7
DepRule	0	8	0	13	0	5	0	4

Table 1: Number of positive and negative seed rules for each dataset. `S` denotes the number of seed rules by (Safranchik et al., 2020) and `M` denotes the rules we manually selected. † denotes that corresponding rule type includes both `inclusive` and `exclusive` rules.

3.3 Seed Rules

In our experiments, we used all the labeling rules[3] developed by Safranchik et al. (2020) as part of positive seeding rules. Besides, we manually selected another small set of labeling rules as our input because: (1) we defined six types of rules as described in Section 2.1, but some types of rules were not used in Safranchik et al. (2020) (e.g., prefix rules are not used in NCBI), (2) our method requires negative seed rules, which are used to identify terms that are not entities, to initiate its learning process. To automatically learn new labeling rules, we use both labeling rules from (Safranchik et al., 2020) and our manually selected rules as seeding rules. Numbers of both positive and negative seeding rules used in our experiments are shown in Table 1.[4] Our manually selected seed rules are

---

[3] In our experiments, we used the linked rules from (Safranchik et al., 2020) to obtain weak labels, which are another type of rules that can be used to vote if the two consecutive tokens have the same label, to train our LinkedHMM model. However, our work only focused on augmenting labeling rules, and did not augment linking rules.

[4] Note that in previous work a whole lexicon or dictionary is counted as one rule. However, in our work we count each term in a lexicon or dictionary as one surface form rule.

Method	Human Effort	NCBI			BC5CDR			LaptopReview		
		P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Supervised	Full Annotations	85.2	89.2	87.2	87.2	88.0	87.5	83.5	82.2	82.9
Snorkel	Saf.'s Manual Rules	-	-	68.7	-	-	83.2	-	-	60.0
SwellShark	Fries' Manual Rules	64.7	69.7	67.1	85.0	83.5	84.2	-	-	-
+Special	Candidate Generator	81.6	80.1	**80.8**	86.1	82.4	84.2	-	-	-
AutoNER	Dictionaries	79.4	72.0	75.5	89.0	81.0	84.8	72.3	59.8	65.4
LinkedHMM	Saf.'s Manual Rules	-	-	78.7	-	-	85.9	-	-	68.2
	Our Seed Rules	89.4	70.7	79.0	88.0	84.5	86.2	82.7	59.8	69.4
GLARA	Our Seed Rules	89.9	73.2	80.2±.2	88.2	84.6	**86.3**±.3	82.4	64.2	**72.2**±1.2

Table 2: Micro F1 performance of baselines and our method on test sets. Our results are the mean and std across 5 random runs. `Saf.'s Manual Rules` are the rules developped by Safranchik et al. (2020), and `Fries' Manual Rules` are those used in (Fries et al., 2017).

included in Appendix C.

3.4 Implementation Details

In our experiments, we create a graph for each type of rules for each dataset and learn new rules independently with the same setup. `Prefix` and `Suffix` candidate rules are generated by considering the first and last 3 to 6 characters, and `PreNgram` and `PostNgram` candidate rules are extracted with the windows of 1 to 3 tokens.

We use a two-layer graph attention network to train our graph model. After training, we select M new rules for each type of rules, where the value of M is searched between 20 and 500 on dev sets.

For different datasets, our discriminative NER models used different pre-trained contextual models. Since NCBI and BC5CDR datasets are in the biomedical domain, we finetuned our NER model on the pretrained SciBERT embedding (Beltagy et al., 2019), while for LaptopReview data, the NER model is finetuned on the pretrained BERT_{base} (Devlin et al., 2018) embedding. More details are provided in Appendix A.

4 Experimental Results

In this section, we first compare our method with state of the art methods using all available manual rules. Then, we evaluate our method under scenarios when only limited seeding rules are available, which are very common in real-world applications. Next, we conduct an ablation study to investigate the effectiveness of different types of rules and our newly proposed loss function based on centroid distance. Finally, we also perform a quality analysis of the automatically learned labeling rules.

4.1 Results with Abundant Seeding Rules

In this subsection, we report both the result of our *generative* model with augmented rules in Table 2 and the result of our *discriminative* NER model that is trained using weak labels from the generative model in Table 3.

Table 2 shows the performance of our generative model with augmented rules and baseline generative models[5]. The *Supervised* line is the performance of the LinkedHMM model trained on fully labeled training data. The results show that our method with augmented labeling rules performed best on BC5CDR and LaptopReview datasets. We also achieved a comparative F1 score of 80.2 on NCBI with SwellShark (F1 80.8). Note that SwellShark used lots of manual effort from experts to carefully tune a particular candidate generator for a given dataset. We also notice that, with augmented rules, our method outperforms LinkedHMM by an average of 2.0 F1 points, which demonstrates the effectiveness of augmented rules.

WeakSupervision	NCBI	BC5CDR	Laptop
Snorkel	73.4±1.7	82.2±.5	63.5±1.7
LinkedHMM	79.0±.4	83.0±.2	69.0±1.1
GLARA	**80.8**±.3	**83.5**±.6	**72.3**±1.0

Table 3: Results of discriminative models using weak labels generated by different methods. Our results are the average of five runs.

Table 3 shows the performance of our discriminative NER model (BiLSTM-CRF) trained on weak labels from our GLARA with augmented rules and baseline discriminative models using weak labels from LinkedHMM and Snorkel without augmented rules (Safranchik et al., 2020). The results show

---

[5] Some scores were reported in (Safranchik et al., 2020)

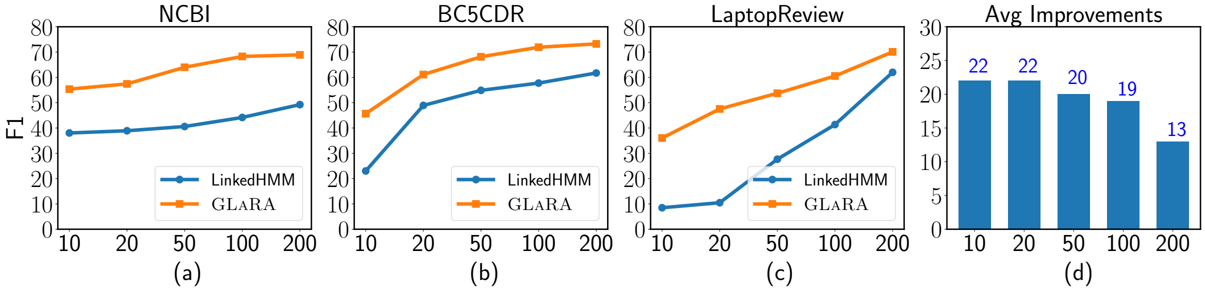


Figure 4: Results of LinkedHMM with only seeding rules and augmented rules by our GLaRA method. The bottom numbers are the numbers of seeding rules used.

that our method achieved a +1.9 F1 point improvement over the second-best system. We also notice that our discriminative model performs slightly better on `NCBI` and `Laptop`, but worse on `BC5CDR` than the generative model, which is consistent with that from (Safranchik et al., 2020).

4.2 Results with Limited Seeding Rules

Though weakly supervised state-of-the-art methods reported in Table 2 achieved relatively good performance, they require a significant amount of manual effort from domain experts for designing and tuning labeling rules. Well-performing methods that require less manual effort are often more desirable. Therefore, we also evaluate our method with little manual effort (i.e., limited seeding rules). We conducted experiments by randomly select at most k rules for each rule type from our seed rules (Section 3.3), where $k \in \{10, 20, 50, 100, 200\}$. When a rule type has less than k seeding rules, we use all of them. Figure 4 shows the performance of the LinkedHMM model using only seeding rules and our GLaRA method with augmented rules on three datasets. Figure 4(d) shows that, with automatically learned rules, our method can obtain average F1 gains of +22 points when using 10 seeding rules and +13 points when there are 200 seeding rules.

4.3 Impact of Different Types of Rules

To investigate the effectiveness of different types of labeling rules, we conducted ablation experiments to evaluate our generative model with augmented rules (i.e., GLaRA) by adding each type of rules cumulatively. Results are shown in Table 4. Empty cells denote that the corresponding type of rules are not used. Results show that `Prefix` rules are most effective on the Laptop dataset, producing +1.5 F1 point improvement. All other rules except `PostNgram` can improve the performance by +0.4 to +0.6 F1 points.

	NCBI	CDR	Laptop	Δ
LinkedHMM	78.7	85.9	68.2	
GLaRA				
+DepRule	79.3	-	-	+0.6
+PostNgram	-	86.0	-	+0.1
+PreNgram	79.6	86.2	69.3	+0.5
+Prefix	-	-	70.8	+1.5
+Suffix	79.8	86.3	71.7	+0.4
+Surface	80.2	-	72.2	+0.5

Table 4: Impact of each type of rules used in our generative model. Δ denotes the average improvement achieved by adding rules cumulatively.

4.4 Effectiveness of Distance Loss $\mathcal{L}_{dist}$

In this work, we proposed a new graph neural network model with a new loss function, i.e., $\mathcal{L}_{dist}$ in Eq. 3. This loss function measures the distance between the centroids of positive and negative seeding rules by computing their cosine similarity. The motivation is that the model should keep positive rules distant from negative ones during the learning process. Table 5 shows the performance of our generative model with and without the distance loss. We find that our method can obtain an average F1 gain of 1.2 points across three datasets with this new loss function.

GLaRA	NCBI	BC5CDR	LaptopReview
w/ $\mathcal{L}_{dist}$	$80.2_{\pm .2}$	$86.3_{\pm .3}$	$72.2_{\pm 1.2}$
w/o $\mathcal{L}_{dist}$	$79.3_{\pm 1.2}$	$86.1_{\pm .1}$	$69.7_{\pm .8}$

Table 5: Results of our generative model using augmented rules learned by our graph neural network model with and without distance loss.

RuleType	**Top 5 learned rules**
`Surface`	neurologic_disease, spherocytic_hemolytic_anemia, episodic_hemolytic_anemia, choroid_plexus_carcinoma, sporadic_human_renal_cell_carcinoma
`Suffix`	*ioma, *ophy, *ndrome, *umonia, *phoma
`Prefix`	mesot*, mesoth*, prost*, dyst*, scler*
`PreNgram`	stage_ii_*, breast_/_ovarian_*, heterogeneity_/_in_*, a_syndromal_*, pathology_of_*
`PostNgram`	*_hepatitis, *_cardiomyopathy, *_carcinoma, *_colitis, *_lymphoma
`DepRule`	FirstTokenDep:amod_LastToken:syndrome FirstTokenDep:compound_LastToken:sarcoma SecondLastTokenDep:amod_LastToken:dystrophy SecondLastTokenDep:conj_LastToken:dystrophy SecondLastTokenDep:nmod_LastToken:syndrome

Table 6: Examples of learned new rules for recognizing diseases on NCBI data. "_" denotes the space character.

4.5 Quality Analysis of Learned New Rules

In Table 6, we present top 5 learned rules for each rule type that we automatically learned on the NCBI data set. Each rule is formatted in the way described in Section 2.1.

Table 7 shows the analysis results of the number of new rules (#Rule) and their accuracy (ACC) on the dev sets, which were learned during the experiments presented in Section 4.1. Results show that all the learned rules have accuracy ≥64% with an average of 76%, which justifies the quality of these new rules.

RuleType	**Dataset**	**#Rule**	**ACC**
Surface	NCBI	301	78%
	Laptop	154	69%
Suffix	NCBI	64	100%
	CDR(Dis)	138	90%
	Laptop	25	66%
Prefix	Laptop	16	78%
PreNgram	NCBI	156	65%
	CDR(Dis)	19	64%
	Laptop	17	71%
PostNgram	CDR(Chem)	37	82%
DepRule	NCBI	47	70%

Table 7: Quality analysis of learned new rules.

Besides, we also manually analyzed why some learned rules are not helpful to improve recognition performance. First, some of the learned rules can be overlapping. For example, both "`∗demia→Disease`" and "`∗edemia→Disease`" can be used to recognize `Diseases`. However, if `∗demia` is learned first, then learning `∗edemia` rule will not help because the entities matched by these two rules have large overlaps. Second, some of the rules learned from unlabeled data may not be applied to testing data due to the mismatch between two datasets. For example, though the PostNgram rule "`∗_dystonia→Disease`" is an accurate rule to label diseases such as "oromandibular dystonia" and "responsive dystonia". However, we do not find any matches on test data.

5 Related Work

Training reliable NER systems usually requires large annotation efforts, which is often considered expensive and impractical in certain domains. Therefore, previous studies have been trying to reduce the manual efforts required for annotation while producing comparable performance in NER tasks by utilizing manually designed rules that are cheap and accurate.

Studies have shown that human-defined rules (i.e dictionaries) can greatly aid NER tasks, especially in the domains where the identification of entities needs domain knowledge (Cohen and Sarawagi, 2004; Savova et al., 2010; Xu et al., 2010; Eftimov et al., 2017), and they can also be used to distantly create labeled data set to benefit machine learning models (Mann and McCallum, 2010; Neelakantan and Collins, 2015; Giannakopoulos et al., 2017).

While distantly labeled data sets can be created at a low cost to boost NER tasks, the models still suffer from the noise introduced by the imperfect rules. Therefore, denoising models have been proposed to allow the model to better tolerate the imperfect annotations created by rules. Shang et al. (2018) proposed AutoNER that trains NER systems using only lexicons with a tie-or-break tagging denoising model. Similarly, some recent work (Liu et al., 2019; Yang et al., 2018; Cao et al., 2019) have used a partial matching module to denoise the noisily labeled data sets.

Recently, weak supervision is proposed as another form of denoising framework without using any labeled data. Weakly supervised systems use handcrafted labeling rules to create weak training instances from unlabeled data and then use a denoising generative model to approximate the posteriors of the rules. In this process, the unknown gold label is treated as latent variable by the generative model. The performance of the labeling rules could be further enhanced by training a neural network-based discriminative model by treating the posteriors as soft labels. Related studies such as Snorkel (Ratner et al., 2020), SwellShark (Fries et al., 2017), LinkedHMM (Safranchik et al., 2020), and (Lison et al., 2020) have demonstrated great success with carefully curated labeling rules.

However, manually designing those high-quality rules often require domain expertise and easy to have a low sensitivity on identifying entities. In (Lison et al., 2020), the weak training data is created by broadly collecting available labeling rules from multiple sources, which demonstrates the importance of being able to automatically find new heuristics missed by human efforts. To find new heuristic rules on the basis of a relatively limited number of manually designed rules, previous studies have tried bootstrapping by relying on the co-occurrence, context and pattern features (Thelen and Riloff, 2002; Riloff et al., 2003; Yangarber, 2003; Shen et al., 2017; Tao et al., 2015; Berger et al., 2018; Yan et al., 2019).

Recent studies on graph neural networks has opened up another possibility for learning new rules. By internally infusing the semantics of the neighboring nodes, the popular Graph Convolutional Network (GCN) (Kipf and Welling, 2016) and Graph Attention Network (GAT) (Veličković et al., 2017) have shown great success in semi-supervised node classification when the number of labeled nodes is limited. Graph neural networks have been applied for many NLP tasks such as text classification (Yao et al., 2019; Zhang et al., 2019a; Hu et al., 2019), semantic role labeling (Marcheggiani and Titov, 2017), machine translation (Beck et al., 2018), question answering (Song et al., 2018; Saxena et al., 2020), information extraction (Liu et al., 2018; Vashishth et al., 2018; Nguyen and Grishman, 2018; Sahu et al., 2019; Sun et al., 2019; Fu et al., 2019; Zhang et al., 2019b), etc. In our work, we proposed to use graph neural networks to learn new labeling rules. Based on Graph Attention Network (Veličković et al., 2017), we designed a new graph network model with a new loss function. Experimental results demonstrated that our model performed better than the original graph attention network on learning accurate labeling rules.

6 Conclusion

In this work, we proposed a weakly supervised NER framework that automatically learns high-quality new rules from only a handful of manually designed rules with a graph-based labeling rule augmentation method (GLARA). Experiments on three NER datasets demonstrate that our model outperforms baseline systems and achieves substantially better performance when the number of manual rules is limited. In addition, we also defined six types of rules that have been demonstrated useful for recognizing entities. In the future, we plan to improve GLARA by investigating more complex rule types and rule representation methods for weakly supervised NER.

References

Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. 2017. Learning the structure of generative models without labeled data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 273–282. JMLR. org.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. *arXiv preprint arXiv:1806.09835*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3606–3611.

Matthew Berger, Ajay Nagesh, Joshua Levine, Mihai Surdeanu, and Helen Zhang. 2018. Visual supervision in bootstrapped information extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2043–2053.

Yixin Cao, Zikun Hu, Tat-Seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-resource name tagging learned with weakly labeled data. *arXiv preprint arXiv:1908.09659*.

William W Cohen and Sunita Sarawagi. 2004. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Tome Eftimov, Barbara Koroušić Seljak, and Peter Korošec. 2017. A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations. *PloS one*, 12(6):e0179488.

Jason Fries, Sen Wu, Alex Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *arXiv preprint arXiv:1704.06360*.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1409–1418.

Athanasios Giannakopoulos, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. 2017. Unsupervised aspect term extraction with b-lstm & crf using automatically labelled datasets. *arXiv preprint arXiv:1709.05094*.

Linmei Hu, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,(EMNLP-IJCNLP)*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Pierre Lison, Aliaksandr Hubin, Jeremy Barnes, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. *arXiv preprint arXiv:2004.14723*.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5301–5307.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. *arXiv preprint arXiv:1809.09078*.

Gideon S Mann and Andrew McCallum. 2010. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of machine learning research*, 11(Feb):955–984.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Arvind Neelakantan and Michael Collins. 2015. Learning dictionaries for named entity recognition using minimal supervision. *arXiv preprint arXiv:1504.06650*.

Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *10th International Workshop on Semantic Evaluation (SemEval 2016)*.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2020. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, 29(2):709–730.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 25–32.

Esteban Safranchik, Shiying Luo, Stephen H Bach, Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, et al. 2020. Weakly supervised sequence tagging from noisy rules. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Inter-sentence relation extraction with document-level graph convolutional neural network. *arXiv preprint arXiv:1906.04684*.

Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. 2010. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507.

Jingbo Shang, Liyuan Liu, Xiang Ren, Xiaotao Gu, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. *arXiv preprint arXiv:1809.03599*.

Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 288–304. Springer.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv preprint arXiv:1809.02040*.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1361–1370.

Fangbo Tao, Bo Zhao, Ariel Fuxman, Yang Li, and Jiawei Han. 2015. Leveraging pattern semantics for extracting entities in enterprises. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1078–1088.

Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 conference on empirical methods in natural language processing (EMNLP 2002)*, pages 214–221.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. Reside: Improving distantly-supervised neural relation extraction using side information. *arXiv preprint arXiv:1812.04361*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Hua Xu, Shane P Stenner, Son Doan, Kevin B Johnson, Lemuel R Waitman, and Joshua C Denny. 2010. Medex: a medication information extraction system for clinical narratives. *Journal of the American Medical Informatics Association*, 17(1):19–24.

Lingyong Yan, Xianpei Han, Le Sun, and Ben He. 2019. Learning to bootstrap for entity set expansion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 292–301.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised ner with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169.

Roman Yangarber. 2003. Counter-training in discovery of semantic patterns. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 343–350.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019a. Aspect-based sentiment classification with aspect-specific graph convolutional networks. *arXiv preprint arXiv:1909.03477*.

Yan Zhang, Zhijiang Guo, and Wei Lu. 2019b. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480.

A Hyperparameter configuration

Graph Propagation Model We use the same graph architecture to train propagation for all types of rules. The model contains 2 graph attention layers each with 3 attention heads and dropout rate is set to be 0.5. The hidden size of each layer is set as 64. We use Adam optimizer with a learning rate of 0.0001. Other hyperparameters (training epoch and number of selected new rules) are presented in Table 8.

Hyperparam	NCBI		BC5CDR		Laptop	
	epoch	#rules	epoch	#rules	epoch	#rules
SurfaceForm	50	50	50	-	50	25
Prefix	50	-	50	-	50	15
Suffix	50	25	50	25(`D`)	50	25
PreNgram$_{\text{inclusive}}$	50	25	30	-	30	
PreNgram$_{\text{exclusive}}$	50	-	50	25(`D`)	30	
PostNgram$_{\text{inclusive}}$	50	-	30		50	
PostNgram$_{\text{exclusive}}$	50	-	50	30(`C`)	50	
Dependency	50	25	50	-	50	25

Table 8: Summary of hyperparameters for propagation. In BC5CDR data. "`D`" denotes the number of selected rules for Disease and "`C`" denotes the that for Chemical for the corresponding rule type. "-" means the corresponding type of propagated rules are not used on our final model.

Generative Model Table 9 presents the hyperparameters used for tuning our LinkedHMM generative model, including "Initial Accuracy (estimated initial accuracy of the rules)", "Accuracy Prior (regularization for initial accuracy)", and "Balance Prior (the entity class distribution)". We used grid search to find the best hyperparameters. The search ranges are created around the default settings of the LinkedHMM model on the three data sets. For training epoch, we use the default setting, 5, for all three datasets. For more details about the hyperparameters, please refer to (Safranchik et al., 2020).

Hyperparam		NCBI	CDR	Laptop
Init Acc	Search	[0.75-0.95]	[0.75-0.95]	[0.75-0.95]
	Best	0.85	0.85	0.9
Acc Prior	Search	[45-65]	[0-15]	[0-5]
	Best	55	5	1
Bal Prior	Search	[440, 460]	[440, 460]	[0, 20]
	Best	450	450	10

Table 9: Summary of hyperparameters for training the LinkedHMM generative model on each data set. The search steps for `InitialAccuracy`, `AccuracyPrior`, and `BalancePrior` are 0.05, 5, and 5, respectively.

Discriminative Model Table 10 presents the hyperparameter configuration for training discriminative model (BiLSTM-CRF). BERT$_{base}$ is used to extract word embeddings and fine-tuned with BiLSTM-CRF. All discriminative models are trained on a 11G 1080Ti GPU with training time being up to ∼20s/epoch. All models have ∼110M parameters.

B Performance on Development Data

In Table 11, we present the performance of the LinkedHMM model on developement sets, with the additional seeding rules manually selected by us, referred as LinkedHMM-M, the performance of the following discriminative model, referred as LinkedHMM-M-D. Also, we report the performance of our models, GLARA and GLARA-D (with discriminative model) on development sets.

Hyperparam		NCBI	BC5CDR	Laptop
BERT		SciBERT	SciBERT	BERT
BiLSTM	Hidden Dim	256	256	256
	Dropout	0.1	0.1	0.1
CRF		yes	yes	no
AdamW	Learning Rate	1e-4	1e-4	1e-4
	Epoch	30	30	30
Batch size		8	8	8
Max sent length		128	128	128

Table 10: Summary of hyperparameters for training discriminative model on each data set.

Model	**NCBI**	**BC5CDR**	**Laptop**
LinkedHMM-M	82.3	87.5	70.1
LinkedHMM-M-D	82.8$\pm$.3	84.5$\pm$.2	71.5$\pm$.8
GLARA	83.1$\pm$.2	87.4$\pm$.3	72.3$\pm$.8
GLARA-D	83.4$\pm$.3	84.5$\pm$.1	72.6$\pm$.6

Table 11: Micro F1 performance on each development data set. $\mathtt{LinkedHMM-M}$ denotes the baseline LinkedHMM model trained with the extra manually selected seeding rules. $\mathtt{LinkedHMM-M-D}$ denotes the discriminative mode (LSTM-CRF) trained based on $\mathtt{LinkedHMM-M}$. Similarly, GLARA$-\mathtt{D}$ denotes the LSTM-CRF model trained based on GLARA.

C Manually Selected Seeds for propagation

As mentioned in the paper, the baseline LinkedHMM does not have seeding rules for all types of rules. For example, negative seeding rules are missing from the baseline LinkedHMM model, except that `SurfaceForm` rules uses a list of stopwords as negative seeding rules. For some types of seeding rules, there are only a few available that are not good enough for training the propagation model. Therefore, for the rule types that do not have enough seeds, we manually select a small set of additional rules as seeds. Note that we keep the total number of seeding rules (including the ones from baseline system) less than 15. We report the manually selected seeding rules for NCBI, BC5CDR-Disease, BC5CDR-Chemical, and LaptopReview in Table 12, Table 13, Table 14, and Table 15, respectively.

Rules		NCBI
`Surface`	pos	-
	neg	-
`Suffix`	pos	*skott, *drich, *umour, *axia, *iridia
	neg	*ness, *nant, *tion, *ting, *enesis, *riant, *tein, *sion, *osis, *lity
`Prefix`	pos	carc*, myot*, tela*, ovari*,atax*, carcin*, dystro*
	neg	defi*, comp*, fami*, poly*, chro*, prot*, enzym*, sever*, develo*, varian*
`exclusive PreNgram`	pos	suffer_from_*, fraction_of_*, pathogenesis_of_*, cause_severe_*
	neg	-pron_*, suggest_that_*, -_cell_*, presence_of_*, expression_of_*, majority_of_* loss_of_*, associated_with_*,impair_in_*, cause_of_*, defect_in_*, family_with_*
`inclusive PreNgram`	pos	breast_and_ovarian_*, x_-_link_*, breast_and_*, stage_iii_*, myotonic_*
	neg	enzyme_*, primary_*, non_-_*,
`exclusive PostNgram`	pos	
	neg	*_and_the, *_cell_line, *_in_the
`inclusive PostNgram`	pos	*_-_t, *_cell_carcinoma, *_muscular_dystrophy, *_'s_disease, *_carcinoma, *_dystrophy
	neg	*_muscle, *_ataxia, *_'system, *_defect , *_other_cancer, *_i, *_ii
`DepRule`	pos	FirstTokenDep:amod_LastToken:dystrophy, FirstTokenDep:punct_LastToken:telangiectasia, FirstTokenDep:compound_HeadSurf:t, FirstTokenDep:amod_LastToken:dysplasia SecondLastTokenDep:compound_LastToken:syndrome,
	neg	FirstTokenDep:amod_LastToken:deficienc, FirstTokenDep:amod_LastToken:deficiency FirstTokenDep:amod_LastToken:defect, FirstTokenDep:pobj_LastToken:cancer SecondLastTokenDep:compound_LastToken:cancer, SecondLastTokenDep:compound_LastToken:disease SecondLastTokenDep:appos_LastToken:t, SecondLastTokenDep:compound_LastToken:t

Table 12: Manually selected seeding rules for NCBI dataset. "-" means no seeding rules selected, and we only use the rules provided in baseline.

Rules		BC5CDR (Disease)
`Surface`	pos	-
	neg	-
`Suffix`	pos	*epsy, *nson
	neg	*ing, *tion, *tive, *lity, *mone, *fect, *crease, *sion, *lion, *elet, *gical, *nosis, *sive, *ment, *tory, *sionetic, *ency, *ture,
`Prefix`	pos	anemi*, dyski*, heada*, hypok*, hypert*, ische*, arthr*, hypox*, toxic*, arrhyt*, ischem*, hypert*, dysfunc*
	neg	symp*, resp*, funct*, inter*, decre*, prote*, neuro*, cardi*, myoca*, ventr*, decre* syst*
`exclusive PreNgram`	pos	to_induce_*, w_-_*, and_severe_*, suspicion_of_*, die_of_*, have_severe_*, of_persistent_*, cyclophosphamide_associate_*
	neg	seizure_and_*, symptom_and_*, dysfunction_and_*, failure_with_*, sign_of_*, lead_to_*
`inclusive PreNgram`	pos	parkinson_'s_*, torsade_de_*, acute_liver_*, neuroleptic_*, malignant_*, alzheimer_'s_*, congestive_heart_*, migraine_with_*, sexual_side_*, renal_cell_*, tic_-_*
	neg	renal_function_*, decrease_in_*, increase_in_*, reduction_in_*, rise_in_*, loss_of_* chronic_liver_*, abnormality_in_*, human_immunodeficiency_*, optic_nerve_*, drug_-_* non_-_*
`exclusive PostNgram`	pos	-
	neg	*_and_the, *_cell_line, *_in_the
`inclusive PostNgram`	pos	*_'s_disease, *_infarction, *_'s_sarcoma, *_epilepticus , *_artery_disease, *_de_pointe *_insufficiency, *_with_aura, *_artery_spasm, *_'s_encephalopathy
	neg	*_toxicity, *_pain, *_fever, *_function, *_blood_pressure, *_effect, *_impairment, *_loss *_event, *_protein, *_pressure, *_impair, *_phenomenon, *_system, *_side_effect *_of_disease
`DepRule`	pos	FirstTokenDep:poss_LastToken:disease FirstTokenDep:compound_LastToken:cancer, FirstTokenDep:amod_LastToken:dysfunction FirstTokenDep:compound_LastToken:disease, FirstTokenDep:compound_LastToken:failure FirstTokenDep:compound_LastToken:anemia, FirstTokenDep:compound_LastToken:cancer SecondLastTokenDep:pobj_LastToken:disease
	neg	FirstTokenDep:amod_LastToken:toxicity FirstTokenDep:amod_LastToken:impairment, FirstTokenDep:amod_LastToken:syndrome FirstTokenDep:amod_LastToken:complication, FirstTokenDep:amod_LastToken:symptom FirstTokenDep:amod_LastToken:damage, FirstTokenDep:amod_LastToken:disease FirstTokenDep:amod_LastToken:function, FirstTokenDep:amod_LastToken:damage FirstTokenDep:compound_LastToken:loss, SecondLastTokenDep:nmod_LastToken:b SecondLastTokenDep:conj_LastToken:arrhythmia SecondLastTokenDep:pobj_LastToken:symptom

Table 13: Manually selected seeding rules for BC5CDR (Disease) dataset. "-" means no seeding rules selected, and we only use the rules provided in baseline.

Rules		BC5CDR (Chemical)
`Surface`	pos	-
	neg	-
`Suffix`	pos	*pine, *icin, *dine, *ridol, *athy, *zure, *mide, *fen, *phine
	neg	*ing, *tion, *tive, *tory, *inal, *ance, *duce, *atory, *mine, *line, *tin, *rate, *late, *ular, *etic, *onic, *ment, *nary, *lion, *ysis, *logue, *mone
`Prefix`	pos	chlor*, levo*, doxor*, lithi*, morphi*, hepari*, ketam*, potas*
	neg	meth*, hepa*, prop*, contr*, pheno*, contra*, acetyl*, dopami*
`exclusive PreNgram`	pos	dosage_of_*, sedation_with_*, mg_of_*, application_of_*, -_release_*, ingestion_of_* intake_of_*
	neg	of_to_*, to_the_*, be_the_*, with_the_*, in_the_*, on_the_*, for_the_*
`inclusive PreNgram`	pos	external_*, mk_*, mk_-_*, cis_*, cis_-_*, nik_*, nik_-_*, ly_*, ly_-_*, puromycin_*
	neg	reduce_*, all_*, a_*, the_*, of_*, alpha_*, alpha_-_*, beta_*, beta_-_*
`exclusive PostNgram`	pos	*_-_associate, *_-_induced
	neg	*_-_related, *_that_of, *_of_the
`inclusive PostNgram`	pos	*_aminocaproic_acid, *_-_aminocaproic *_acid, *_retinoic_acid, *_dopa, *_tc *_-_aminopyridine, *_aminopyridine, *_-_penicillamine, *_-_dopa, *_-_aspartate, *_fu *_hydrochloride
	neg	*_drug, *_cocaine, *_calcium, *_receptor_agonist, *_blockers, *_block_agent *_inflammatory_drug
`DepRule`	pos	FirstTokenDep:amod_LastToken:oxide, FirstTokenDep:compound_LastToken:chloride FirstTokenDep:amod_LastToken:acid, FirstTokenDep:compound_LastToken:acid FirstTokenDep:compound_LastToken:hydrochloride SecondLastTokenDep:amod_LastToken:aminonucleoside,
	neg	FirstTokenDep:compound_LastToken:a, SecondLastTokenDep:pobj_LastToken:acid SecondLastTokenDep:pobj_LastToken:a, SecondLastTokenDep:pobj_LastToken:the SecondLastTokenDep:pobj_LastToken:a

Table 14: Manually selected seeding rules for BC5CDR (Chemical) dataset. "-" means no seeding rules selected, and we only use the rules provided in baseline.

Rules		LatopReview
`Surface`	pos	-
	neg	-
`Suffix`	pos	*pad, *oto, *fox, *chpad, *rams
	neg	*ion, *ness, *nant, *lly, *ary
`Prefix`	pos	feat*, softw*, batt*, Win*, osx*
	neg	pro*, edit*, repa*, rep*, con*, dis*, appl*, equip*
`exclusive PreNgram`	pos	-
	neg	in_the_*, on_the_*, for_the_*, -pron_*
`inclusive PreNgram`	pos	windows_*, hard_*, extended_*, touch_*, boot_*
	neg	mac_*, apple_*, a_*, launch_*, software_*
`exclusive PostNgram`	pos	*_and_seal, *_that_come_with
	neg	*_shut_down, *_do_not_work,
`inclusive PostNgram`	pos	*_x, *_xp, *_vista, *_drive, *_processing
	neg	*_screen, *_software, *_quality, *_technical, *_cut
`DepRule`	pos	FirstTokenDep:compound_LastToken:port, FirstTokenDep:compound_LastToken:button FirstTokenDep:nummod_LastToken:ram, FirstTokenDep:amod_LastToken:drive
	neg	FirstTokenDep:compound_LastToken:option, SecondLastTokenDep:pobj_LastToken:plan SecondLastTokenDep:nsubj_LastToken:design, SecondLastTokenDep:pobj_LastToken:plan

Table 15: Manually selected seeding rules for LaptopReview dataset. "-" means no seeding rules selected, and we only use the rules provided in baseline.

An End-to-end Model for Entity-level Relation Extraction using Multi-instance Learning

Markus Eberts **Adrian Ulges**
RheinMain University of Applied Sciences
Wiesbaden, Germany
`{markus.eberts, adrian.ulges}@hs-rm.de`

Abstract

We present a joint model for entity-level relation extraction from documents. In contrast to other approaches – which focus on local intra-sentence mention pairs and thus require annotations on mention level – our model operates on entity level. To do so, a multi-task approach is followed that builds upon coreference resolution and gathers relevant signals via multi-instance learning with multi-level representations combining global entity and local mention information. We achieve state-of-the-art relation extraction results on the DocRED dataset and report the first entity-level end-to-end relation extraction results for future reference. Finally, our experimental results suggest that a joint approach is on par with task-specific learning, though more efficient due to shared parameters and training steps.

1 Introduction

Information extraction addresses the inference of formal knowledge (typically, entities and relations) from text. The field has recently experienced a significant boost due to the development of neural approaches (Zeng et al., 2014; Zhang and Wang, 2015; Kumar, 2017). This has led to two shifts in research: First, while earlier work has focused on sentence level relation extraction (Hendrickx et al., 2010; Han et al., 2018; Zhang et al., 2017), more recent models extract facts from longer text passages (document-level). This enables the detection of inter-sentence relations that may only be implicitly expressed and require reasoning across sentence boundaries. Current models in this area do not rely on mention-level annotations and aggregate signals from multiple mentions of the same entity.

The second shift has been towards multi-task learning: While earlier approaches tackle entity mention detection and relation extraction with separate models, recent joint models address these tasks at once (Bekoulis et al., 2018; Nguyen and Verspoor, 2019; Wadden et al., 2019). This does not only improve simplicity and efficiency, but is also commonly motivated by the fact that tasks can benefit from each other: For example, knowledge of two entities' types (such as *person+organization*) can boost certain relations between them (such as *ceo_of*).

The **Portland Golf Club** is a private golf club in the northwest **United States**, in suburban Portland, Oregon. The **PGC** is located in the unincorporated **Raleigh Hills** area of eastern Washington County, southwest of downtown Portland and east of Beaverton. **PGC** was established in the winter of **1914**, when a group of nine businessmen assembled to form a new club after leaving their respective clubs. The **golf club** hosted the Ryder Cup matches of 1947, the first renewal in a decade, due to World War II. The **U.S.** team defeated Great Britain 11 to 1 in wet conditions in early November.

Figure 1: Our goal is to perform end-to-end entity-level relation extraction on whole documents. We extract entity mentions ("PGC"), entity clusters ({Portland Golf Club, PGC, golf club}), their types (ORG) and relations to other entities in the document, such as ({Portland Golf Club, PGC, golf club}$_{ORG}$, *inception*, {1914}$_{TIME}$), with a single, joint model. Note that document-level relation extraction requires the aggregation of relevant information from multiple sentences, such as in ({Raleigh Hills}$_{LOC}$, *country*, {United States, U.S.})$_{LOC}$). Other entities in the example document are omitted for clarity.

We follow this line of research, and present JEREX[1] ("**J**oint **E**ntity-Level **R**elation **Ex**tractor"),

---

[1] The code for reproducing our results is available at https://github.com/lavis-nlp/jerex.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3650–3660
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

a novel approach for joint information extraction. JEREX is to our knowledge the first approach that combines a multi-task model with entity-level relation extraction: In contrast to previous work, our model jointly learns relations and entities without annotations on mention level, but extracts document-level entity clusters and predicts relations between those clusters using a *multi-instance learning* (MIL) (Dietterich et al., 1997; Riedel et al., 2010; Surdeanu et al., 2012) approach. The model is trained jointly on mention detection, coreference resolution, entity classification and relation extraction (Figure 1).

While we follow best practices for the first three tasks, we propose a novel representation for relation extraction, which combines global entity-level representations with localized mention-level ones. We present experiments on the DocRED (Yao et al., 2019) dataset for entity-level relation extraction. Though it is arguably simpler compared to recent graph propagation models (Nan et al., 2020) or special pre-training (Ye et al., 2020), our approach achieves state-of-the-art results.

We also report the first results for end-to-end relation extraction on DocRED as a reference for future work. In ablation studies we show that (1) combining a global and local representations is beneficial, and (2) that joint training appears to be on par with separate per-task models.

2 Related Work

Relation extraction is one of the most studied natural language processing (NLP) problems to date. Most approaches focus on classifying the relation between a given entity mention pair. Here various neural network based models, such as RNNs (Zhang and Wang, 2015), CNNs (Zeng et al., 2014), recursive neural networks (Socher et al., 2012) or Transformer-type architectures (Wu and He, 2019) have been investigated. However, these approaches are usually limited to local, intra-sentence, relations and are not suited for document-level, inter-sentence, classification. Since complex relations require the aggregation of information distributed over multiple sentences, document-level relation extraction has recently drawn attention (e.g. Quirk and Poon 2017; Verga et al. 2018; Gupta et al. 2019; Yao et al. 2019). Still, these models rely on specific entity mentions to be given. While progress in the joint detection of entity mentions and intra-sentence relations has been made (Gupta et al., 2016; Bekoulis et al., 2018; Luan et al., 2018), the combination of coreference resolution with relation extraction for entity-level reasoning in a single, jointly-trained, model is widely unexplored.

Document-level Relation Extraction Recent work on document-level relation extraction directly learns relations between entities (i.e. clusters of mentions referring to the same entity) within a document, requiring no relation annotations on mention level. To gather relevant information across sentence boundaries, multi-instance learning has successfully been applied to this task. In multi-instance learning, the goal is to assign labels to bags (here, entity pairs), each containing multiple instances (here, specific mention pairs). Verga et al. (2018) apply multi-instance learning to detect domain-specific relations in biological text. They compute relation scores for each mention pair of two entity clusters and aggregate these scores using a smooth max-pooling operation. Christopoulou et al. (2019) and Sahu et al. (2019) improve upon Verga et al. (2018) by constructing document-level graphs to model global interactions. While the aforementioned models tackle very specific domains with few relation types, the recently released DocRED dataset (Yao et al., 2019) enables general-domain research on a rich relation type set (96 types). Yao et al. (2019) provide several baseline architectures, such as CNN-, LSTM- or Transformer-based models, that operate on global, mention averaged, entity representations. Wang et al. (2019) use a two-step process by identifying related entities in a first step and classifying them in a second step. Tang et al. (2020) employ a hierarchical inference network, combining entity representations with attention over individual sentences to form the final decision. Nan et al. (2020) apply a graph neural network (Kipf and Welling, 2017) to construct a document-level graph of mention, entity and meta-dependency nodes. The current state-of-the-art constitutes the CorefRoBERTa model proposed by Ye et al. (2020), a RoBERTa (Liu et al., 2019) variant that is pre-trained on detecting co-referring phrases. They show that replacing RoBERTa with CorefRoBERTa improves performance on DocRED.

All these models have in common that entities and their mentions are both assumed to be given. In contrast, our approach extracts mentions, clusters them to entities, and classifies relations jointly.

Joint Entity Mention and Relation Extraction Prior joint models focus on the extraction of mention-level relations in sentences. Here, most approaches detect mentions by BIO (or BILOU) tagging and pair detected mentions for relation classification, e.g. (Gupta et al., 2016; Zhou et al., 2017; Zheng et al., 2017; Bekoulis et al., 2018; Nguyen and Verspoor, 2019; Miwa and Bansal, 2016). However, these models are not able to detect relations between overlapping entity mentions. Recently, so-called span-based approaches (Lee et al., 2017) were successfully applied to this task (Luan et al., 2018; Eberts and Ulges, 2019): By enumerating each token span of a sentence, these models handle overlapping mentions by design. Sanh et al. (2019) train a multi-task model on named entity recognition, coreference resolution and relation extraction. By adding coreference resolution as an auxilary task, Luan et al. (2019) propagate information through coreference chains. Still, these models rely on mention-level annotations and only detect intra-sentence relations between mentions, whereas our model explicitly constructs clusters of co-referring mentions and uses these clusters to detect complex entity-level relations in long documents using multi-instance reasoning.

3 Approach

JEREX processes documents containing multiple sentences and extracts entity mentions, clusters them to entities, and outputs types and relations on entity level. JEREX consists of four task-specific components, which are based on the same encoder and mention representations, and are trained in a joint manner. An input document is first tokenized, yielding a sequence of n byte-pair encoded (BPE) (Sennrich et al., 2016) tokens. We then use the pre-trained Transformer-type network BERT (Devlin et al., 2019) to obtain a contextualized embedding sequence $(\mathbf{e}_1, \mathbf{e}_2, ...\mathbf{e}_n)$ of the document. Since our goal is to perform end-to-end relation extraction, neither entities nor their corresponding mentions in the document are known in inference.

3.1 Model Architecture

We suggest a multi-level model: First, we localize all entity mentions in the document (a) by a *span-based* approach (Lee et al., 2017). After this, detected mentions are clustered into entities by *coreference resolution* (b). We then classify the type (such as *person* or *company*) of each entity cluster by a fusion over local mention representations (*entity classification*) (c). Finally, relations between entities are extracted by a reasoning over mention pairs (d). The full model architecture is illustrated in Figure 2.

(a) Entity Mention Localization Here our model performs a search over all document token subsequences (or *spans*). In contrast to BIO/BILOU-based approaches for entity mention localization, span-based approaches are able to detect overlapping mentions. Let $s := (\mathbf{e}_i, \mathbf{e}_{i+1}, ..., \mathbf{e}_{i+k})$ denote an arbitrary candidate span. Following Eberts and Ulges (2019), we first obtain a span representation by max-pooling the span's token embeddings:

$$\mathbf{e}(s) := \text{max-pool}(\mathbf{e}_i, \mathbf{e}_{i+1}, ..., \mathbf{e}_{i+k}) \qquad (1)$$

Our *mention classifier* takes the span representation $\mathbf{e}(s)$ as well as a span size embedding $\mathbf{w}^s_{k+1}$ (Lee et al., 2017) as meta information. We perform binary classification and use a sigmoid activation to obtain a probability for s to constitute an entity mention:

$$\hat{y}^s = \sigma\Big(\text{FFNN}^s(\mathbf{e}(s) \circ \mathbf{w}^s_{k+1})\Big) \qquad (2)$$

where $\circ$ denotes concatenation and FFNN^s is a two-layer feedforward network with an inner ReLu activation. Span classification is carried out on all token spans up to a fixed length L. We apply a filter threshold α^s on the confidence scores, retaining all spans with $\hat{y}^s \geq \alpha_s$ and leaving a set $\mathcal{S}$ of spans supposedly constituting entity mentions.

(b) Coreference Resolution Entity mentions referring to the same entity (e.g. "Elizabeth II." and "the Queen") can be scattered throughout the input document. To later extract relations on entity level, local mentions need to be grouped to document-level entity clusters by coreference resolution. We use a simple mention-pair (Soon et al., 2001) model: Our component classifies pairs $(s_1, s_2) \in \mathcal{S}\times\mathcal{S}$ of detected entity mentions as coreferent or not, by combining the span representations $\mathbf{e}(s_1)$ and $\mathbf{e}(s_2)$ with an edit distance embedding $\mathbf{w}^c_d$: We compute the Levenshtein distance (Levenshtein, 1966) between spans $d := D(s_1, s_2)$ and use a learned embedding $\mathbf{w}^c_d$. A mention pair representation $\mathbf{x}^c$ is constructed by concatenation:

$$\mathbf{x}^c := \mathbf{e}(s_1) \circ \mathbf{e}(s_2) \circ \mathbf{w}^c_d \qquad (3)$$

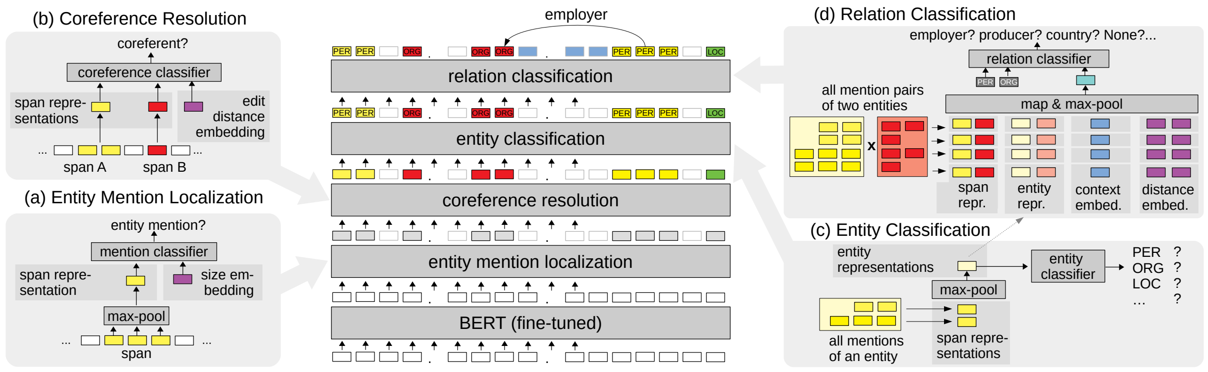


Figure 2: Our approach combines entity mention localization (a), coreference resolution (b), entity classification (c) and relation classification (d) within a joint multi-task model, which is trained jointly on entity-level relation extraction. The sub-components share a single BERT encoder for document encoding. Each input document is only encoded once (*single-pass*) to speed-up training/inference, with sub-components operating on the contextualized embeddings. Both entity classification and relation classification use multi-instance learning to synthesize relevant signals scattered throughout the input document.

Similar to span classification, we conduct binary classification using a sigmoid activation, obtaining a similarity score between the two mentions:

$$\hat{y}^c := \sigma\Big(\text{FFNN}^c(\mathbf{x}^c)\Big) \tag{4}$$

where FFNN^c follows the same architecture as FFNN^s. We construct a similarity matrix $C \in \mathbb{R}^{m \times m}$ (with m referring to the document's overall number of mentions) containing the similarity scores between every mention pair. By applying a filter threshold α^c, we cluster mentions using complete linkage (Müllner, 2011), yielding a set $\mathcal{E}$ containing clusters of entity mentions. We refer to these clusters as *entities* or *entity clusters* in the following.

(c) Entity Classification Next, we map each entity to a type such as $location$ or $person$: We first fuse the mention representations of an entity cluster $\{s_1, s_2, ..., s_t\} \in \mathcal{E}$ by max-pooling:

$$\mathbf{x}^e := \text{max-pool}(\mathbf{e}(s_1), \mathbf{e}(s_2), ..., \mathbf{e}(s_t)) \tag{5}$$

Entity classification is then carried out on the entity representation $\mathbf{x}^e$, allowing the model to draw information from mentions spread across different parts of the document. $\mathbf{x}^e$ is fed into a softmax classifier, yielding a probability distribution over the entity types:

$$\hat{y}^e := \text{softmax}\Big(\text{FFNN}^e(\mathbf{x}^e)\Big) \tag{6}$$

We assign the highest scored type to the entity.

(d) Relation Classification Our final component assigns relation types to pairs of entities. Note that the directionality, i.e. which entity constitutes the head/tail of the relation, needs to be inferred, and that the input document can express multiple relations between different mentions of the same entity pair. Let $\mathcal{R}$ denote a set of pre-defined relation types. The relation classifier processes each entity pair $(e_1, e_2) \in \mathcal{E} \times \mathcal{E}$, estimating which, if any, relations from $\mathcal{R}$ are expressed between these entities. To do so, we score every candidate triple (e_1, r_i, e_2), expressing that e_1 (as head) is in relation r_i with e_2 (as tail). We design two types of relation classifiers: A *global relation classifier*, serving as a baseline, which consumes the entity cluster representations $\mathbf{x}^e$, and a *multi-instance classifier*, which assumes that certain entity mention pairs support specific relations and synthesizes this information into an entity-pair level representation.

Global Relation Classifier (GRC) The global classifier builds upon the max-pooled entity cluster representations $\mathbf{x}_1^e$ and $\mathbf{x}_2^e$ of an entity pair (e_1, e_2). We further embed the corresponding entity types ($\mathbf{w}_1^e$ / $\mathbf{w}_2^e$), which was shown to be beneficial in prior work (Yao et al., 2019), and compute an entity-pair representation by concatenation:

$$\mathbf{x}^p := \Big(\mathbf{x}_1^e \circ \mathbf{w}_1^e\Big) \circ \Big(\mathbf{x}_2^e \circ \mathbf{w}_2^e\Big) \tag{7}$$

This representation is fed into a 2-layer FFNN (similar to FFNN^s), mapping it to the number of relation types $\#\mathcal{R}$. The final layer features sigmoid activations for multi-label classification and assigns any relation type exceeding a threshold α^r:

$$\hat{y}^r := \sigma\Big(\text{FFNN}^p(\mathbf{x}^p)\Big) \tag{8}$$

Multi-instance Relation Classifier (MRC) In contrast to the global classifier (GRC), the multi-instance relation classifier operates on mention level: Since only entity-level labels are available, we treat entity mention pairs as latent variables and estimate relations by a fusion over these mention pairs. For any pair of entity clusters $e_1 {=} \{s_1^1, s_2^1, ..., s_{t_1}^1\}$ and $e_2 {=} \{s_1^2, s_2^2, ..., s_{t_2}^2\}$, we compute a mention-pair representation for any $(s_1, s_2) {\in} e_1 {\times} e_2$. This representation is obtained by concatenating the global entity embeddings (Equation (5)) with the mentions' local span representations (Equation (1))

$$\mathbf{u}(s_1, s_2) := \Big(\mathbf{e}(s_1) \circ \mathbf{x}_1^e\Big) \circ \Big(\mathbf{e}(s_2) \circ \mathbf{x}_2^e\Big) \tag{9}$$

Further, as we expect close-by mentions to be stronger indicators of relations, we add meta embeddings for the *distances* d_s,d_t between the two mentions, both in sentences (d_s) and in tokens (d_t). In addition, following Eberts and Ulges (2019), the max-pooled context between the two mentions ($\mathbf{c}(s_1, s_2)$) is added. This *localized context* provides a more focused view on the document and was found to be especially beneficial for long, and therefore noisy, inputs:

$$\mathbf{u}'(s_1{,}s_2){:=}\mathbf{u}(s_1{,}s_2) \circ \mathbf{c}(s_1{,}s_2) \circ \mathbf{w}_{d_s}^r \circ \mathbf{w}_{d_t}^{r'} \tag{10}$$

This mention-pair representation is mapped by a single feed-forward layer to the original token embedding size (768):

$$\mathbf{u}''(s_1, s_2) := \text{FFNN}^p(\mathbf{u}'(s_1, s_2)) \tag{11}$$

These focused representations are then combined by max-pooling:

$$\mathbf{x}^r {=} \text{max-pool}(\{\mathbf{u}''(s_1, s_2) | s_1 {\in} e_1{,} s_2 {\in} e_2\}) \tag{12}$$

Akin to GRC, we concatenate $\mathbf{x}^r$ with entity type embeddings $\mathbf{w}_1^e/\mathbf{w}_2^e$ and apply a two-layer FFNN (again, similar to FFNN^s). Note that for both classifiers (GRC/MRC), we need to score both (s_1, r_i, s_2) and (s_2, r_i, s_1) to infer the direction of asymmetric relations.

3.2 Training

We perform a supervised multi-task training, whereas each training document features ground truth for all four subtasks (mention localization, coreference resolution, as well as entity and relation classification). We optimize the joint loss of all four components:

$$\mathcal{L} := \beta_s \cdot \mathcal{L}^s + \beta_c \cdot \mathcal{L}^c + \beta_e \cdot \mathcal{L}^e + \beta_r \cdot \mathcal{L}^r \tag{13}$$

$\mathcal{L}^s$, $\mathcal{L}^c$ and $\mathcal{L}^r$ denote the binary cross entropy losses of the span, coreference and relation classifiers. We use a cross entropy loss ($\mathcal{L}^e$) for the entity classifier. A batch is formed by drawing positive and negative samples from a single document for all components. We found such a *single-pass approach* to offer significant speed-ups both in learning and inference:

- Entity mention localization: We utilize all ground truth entity mentions $\mathcal{S}^{gt}$ of a document as positive training samples, and sample a fixed number N_s of random non-mention spans up to a pre-defined length L_s as negative samples. Note that we only train and evaluate on the full tokens according to the dataset's tokenization, i.e. not on byte-pair encoded tokens, to limit computational complexity. Also, we only sample intra-sentence spans as negative samples. Since we found intra-mention spans to be especially challenging ("New York" versus "New York City"), we sample up to $\frac{N_s}{2}$ intra-mention spans as negative samples.

- Coreference resolution: The coreference classifier is trained on all span pairs drawn from ground truth entity clusters $\mathcal{E}^{gt}$ as positive samples. We further sample a fixed number N_c of pairs of random ground truth entity mentions that do not belong to the same cluster as negative samples.

- Entity classification: Since the entity classifier only receives clusters that supposedly constitute an entity during inference, it is trained on all ground truth entity clusters of a document.

- Relation classification: Here we use ground truth relations between entity clusters as positive samples and N_r negative samples drawn from $\mathcal{E}^{gt} {\times} \mathcal{E}^{gt}$ that are unrelated according to the ground truth.

Each component's loss is obtained by averaging over all samples. We learn the weights and biases of sub-component specific layers as well as the

		Joint Model*			**Pipeline**		
Level	**Task**	Precision	Recall	F1	Precision	Recall	F1
(a)	Mention Localization	93.29	92.70	92.99	92.87	92.46	92.66
(b)	Coreference Resolution	82.52	83.06	82.79	82.11	82.66	82.39
(c)	Entity Classification	79.84	80.36	80.10	79.00	79.52	79.26
(d)	Relation Classification	42.76	38.25	40.38	43.61	37.50	40.32
	Relation Classification (GRC)	38.69	37.32	37.98	39.07	36.44	37.70

Table 1: Test set evaluation results of our multi-level end-to-end system JEREX on DocRED (using the end-to-end split). We either train the model jointly on all four sub-components (left) or arrange separately trained models in a pipeline (right) (* joint results are for MRC except for the last row).

meta embeddings during training. BERT is fine-tuned in the process.

4 Experiments

We evaluate JEREX on the DocRED dataset (Yao et al., 2019). DocRED ist the most diverse relation extraction dataset to date (6 entity and 96 relation types). It includes over 5,000 documents, each consisting of multiple sentences. According to Yao et al. (2019), DocRED requires multiple types of reasoning, such as logical or common-sense reasoning, to infer relations.

Note that previous work only uses DocRED for relation extraction (which equals our relation classifier component) and assumes entities to be given (e.g. Wang et al. 2019; Nan et al. 2020). On the other hand, DocRED is exhaustively annotated with mentions, entities and entity-level relations, making it suitable for end-to-end systems. Therefore, we evaluate JEREX both as a relation classifier (to compare it with the state-of-the-art) and as a joint model (as reference for future work on joint entity-level relation extraction).

While prior joint models focus on mention-level relations (e.g. Gupta et al. 2016; Bekoulis et al. 2018; Chi et al. 2019), we extend the strict evaluation setting to entity level: A mention is counted as correct if its span matches a ground truth mention span. An entity cluster is considered correct if it matches the ground truth cluster exactly and the corresponding mention spans are correct. Likewise, an entity is considered correct if the cluster as well as the entity type matches a ground truth entity. Lastly, we count a relation as correct if its argument entities as well as the relation type are correct. We measure precision, recall and micro-F1 for each sub-task and report micro-averaged scores.

Split	#Doc.	#Men.	#Ent.	#Rel.
Train	3,008	78,677	58,708	37,486
Dev	300	7,702	5,805	3,678
Test	700	17,988	13,594	8,787
Total	4,008	104,367	78,107	49,951

Table 2: DocRED dataset split used for end-to-end relation extraction.

Dataset split The original DocRED dataset is split into a train (3,053 documents), dev (1,000) and test (1,000) set. However, test relation labels are hidden and evaluation requires the submission of results via Codalab. To evaluate end-to-end systems, we form a new split by merging train and dev. We randomly sample a train (3,008 documents), dev (300 documents) and test set (700 documents). Note that we removed 45 documents since they contained wrongly annotated entities with mentions of different types. Table 2 contains statistics of our end-to-end split. We release the split as a reference for future work.

Hyperparameters We use BERT$_{\text{BASE}}$ (cased)[2] for document encoding, an attention-based language model pre-trained on English text (Devlin et al., 2019). Hyperparameters were tuned on the end-to-end dev set: We adopt several settings from (Devlin et al., 2019), including the usage of the Adam Optimizer with a linear warmup and linear decay learning rate schedule, a peak learning rate of 5e-5[3] and application of dropout with a rate of 0.1 throughout the model. We set the size of meta embeddings ($\mathbf{w}^s$, $\mathbf{w}^c$, $\mathbf{w}^e$, $\mathbf{w}^r_{d_s}$, $\mathbf{w}^{r'}_{d_t}$) to 25 and the number of epochs to

---

[2] We use the implementation from (Wolf et al., 2019).

[3] We performed a grid search over [5e-6, 1e-5, 5e-5, 1e-4, 5e-4].

Model	Ign F1	F1
CNN (Yao et al., 2019)	40.33	42.26
LSTM (Yao et al., 2019)	47.71	50.07
Ctx-Aware (Yao et al., 2019)*	48.40	50.70
BiLSTM (Yao et al., 2019)	48.78	51.06
Two-Step (Wang et al., 2019)*	-	53.92
HIN (Tang et al., 2020)*	53.70	55.60
JEREX (GRC)*	53.76	55.91
LSR (Nan et al., 2020)*	56.97	59.05
CorefRo (Ye et al., 2020)*	57.90	60.25
JEREX (MRC)*	**58.44**	**60.40**

Table 3: Comparison of our relation classification component (GRC/MRC) with the state-of-the-art on the DocRED relation extraction task. We report test set results on the original DocRED split. Ign F1 ignores relational facts also present in the train set. Models marked with $*$ use a Transformer-type model for document encoding.

	JM*	**SM**
Task	F1	F1
Mention Localization	92.99	92.66
Coreference Resolution	90.54	90.46
Entity Classification	95.66	95.29
Relation Classification	59.46	59.76
Relation Classification (GRC)	56.45	56.55

Table 4: Single-task performance of the joint model (left) and separate models (right) on the end-to-end split (* joint results are for MRC except for the last row).

20. Performance is measured once per epoch on the dev set, out of which the best performing model is used for the final evaluation on the test set. A grid search is performed for the mention, coreference and relation filter threshold (α^s=0.85, α^c=0.85, α^r(GRC)=0.55, α^r(MRC)=0.6) with a step size of 0.05. The number of negative samples (N_s=N_c=N_r=200) and sub-task loss weights (β_s=β_c=β_r=1, β_e=0.25) are manually tuned. Note that some documents in DocRED exceed the maximum context size of BERT (512 BPE tokens). In this case we train the remaining position embeddings from scratch.

4.1 End-to-End Relation Extraction

JEREX is trained and evaluated on the end-to-end dataset split (see Table 2). We perform 5 runs for each experiment and report the averaged results. To study the effects of joint training, we experiment with two approaches: (a) All four sub-components are trained jointly in a single model as described in Section 3.2 and (b) we construct a pipeline system by training each task separately and not sharing the document encoder.

Table 1 illustrates the results for the joint (left) and pipeline (right) approach. As described in Section 3, each sub-task builds on the results of the previous component during inference. We observe the biggest performance drop for the relation classification task, underlining the difficulty in detecting document-level relations. Furthermore, the multi-instance based relation classifier (MRC) outperforms the global relation classifier (GRC) by about 2.4% F1 score. We reason that the fusion of local evidences by multi-instance learning helps the model to focus on appropriate document sections and alleviates the impact of noise in long documents. Moreover, we found the multi-instance selection to offer good interpretability, usually selecting the most relevant instances (see Figure 3 for examples). Overall, we observe a comparable performance by joint training versus using the pipeline system.

This is also confirmed by the results reported in Table 4, where we evaluate the four components independently, i.e. each component receives ground truth samples from the previous step in the hierarchy (e.g. ground truth mentions for coreference resolution). Again, we observe the performance difference between the joint and pipeline model to be negligible. This shows that it is not necessary to build separate models for each task, which would result in training and inference overhead due to multiple expensive BERT passes. Instead, a single neural model is able to jointly learn all tasks necessary for document-level relation extraction, therefore easing training, inference and maintenance.

4.2 Relation Extraction

We also compare our model with the state-of-the-art on DocRED's relation extraction task. Here, entity clusters are assumed to be given. We train and test our relation classification component on the original DocRED dataset split. Since test set labels are hidden, we submit the best out of 5 runs on the development set via CodaLab to retrieve the test set results. Table 3 includes previously reported results from current state-of-the-art models. Note that our global classifier (GRC) is similar to

Queequeg is a fictional character in the 1851 novel Moby-Dick by American author **Herman Melville**. The son of a South Sea chieftain who left home to explore the world, **Queequeg** is the first principal character encountered by the narrator, Ishmael. The quick friendship and relationship of equality between the tattooed cannibal and the white sailor shows **Melville**'s basic theme of shipboard democracy and racial diversity...

Shadowrun:Hong Kong is a turn-based tactical role-playing video game set in the Shadowrun universe. It was developed and published by **Harebrained Schemes**, who previously developed **Shadowrun Returns** and its standalone expansion. It includes a new single - player campaign and also shipped with a level editor that lets players create their own Shadowrun campaigns and share them with other players. In January 2015, **Harebrained Schemes** launched a Kickstarter campaign in order to fund additional features and content they wanted to add to the game, but determined would not have been possible with their current budget. The initial funding goal of US $ 100,000 was met in only a few hours. The campaign ended the following month, receiving over $ 1.2 million. The game was developed with an improved version of the engine used with **Shadowrun Returns** and Dragonfall. **Harebrained Schemes** decided to develop the game only for Microsoft Windows, OS X, and Linux, ...

Figure 3: Two example documents of the DocRED dataset. Highlighted are relations "creator" between "Queequeg" and "Herman Melville" (top) and "developer" between "Shadowrun Returns" and "Harebrained Schemes" (bottom). Bordered pairs are the top selections of the multi-instance relation classifier.

the baseline by (Yao et al., 2019). However, we replace mention span averaging with max-pooling and also choose max-pooling to aggregate mentions into an entity representation, yielding considerable improvement over the baseline. Using the multi-instance classifier (MRC) instead further improves performance by about 4.5%. Here our model also outperforms complex methods based on graph attention networks (Nan et al., 2020) or specialized pre-training (Ye et al., 2020), achieving a new state-of-the-art result on DocRED's relation extraction task.

4.3 Ablation Studies

We perform several ablation studies to evaluate the contributions of our proposed multi-instance relation classifier enhancements: We remove either the global entity representations $\mathbf{x}_1^e, \mathbf{x}_2^e$ (Equation 5) (a) or the localized context representation $\mathbf{c}(s_1, s_2)$ (Equation 10) (b). The performance drops by about 0.66% F1 score when global entity representations are omitted, indicating that multi-instance reasoning benefits from the incorporation of entity-level context. When the localized context representation is omitted, performance is reduced by about 0.90%, confirming the importance of guiding the model to relevant input sections. Finally, we limit the model to fusing only intra-sentence mention pairs (c). In case no such instance exists for an entity pair, the closest (in token distance) mention pair is selected. Obviously, this modification reduces computational complexity and memory consumption, especially for large documents. Nevertheless, while we observe intra-sentence pairs to cover most relevant signals, exhaustively pairing all mentions of an entity pair yields an improvement of 0.67%.

Model	F1
Relation Classification (MRC)	59.76
- (a) Entity Representations	59.10
- (b) Localized Context	58.85
- (c) Exhaustive Pairing	59.09

Table 5: Ablation studies for the multi-level relation classifier (MRC) using the end-to-end split. We either remove global entity representations (a), the localized context (b) or only use intra-sentence mention pairs (c). The results are averaged over 5 runs.

5 Conclusions

We have introduced JEREX, a novel multi-task model for end-to-end relation extraction. In contrast to prior systems, JEREX combines entity mention localization with coreference resolution to extract entity types and relations on an entity level. We report first results for entity-level, end-to-end, relation extraction as a reference for future work. Furthermore, we achieve state-of-the-art results on the DocRED relation extraction task by enhancing multi-instance reasoning with global entity representations and a localized context, outperforming several more complex solutions. We showed that training a single model jointly on all subtasks instead of using a pipeline approach performs roughly on par, eliminating the need of training separate models and accelerating inference. One of the remaining shortcomings lies in the detection of false positive relations, which may be expressed according to the entities' types but are actually not expressed in the document. Exploring options to reduce these false positive predictions seems to be an interesting challenge for future work.

Acknowledgments

This work was funded by German Federal Ministry of Education and Research (Program FHprofUnt, Project DeepCA (13FH011PX6)).

References

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45.

Renjun Chi, Bin Wu, Linmei Hu, and Yunlei Zhang. 2019. Enhancing Joint Entity and Relation Extraction with Language Modeling and Hierarchical Attention. In *Proc. APWeb-WAIM, LNCS 11641*, pages 314–328.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Connecting the Dots: Document-level Neural Relation Extraction with Edge-oriented Graphs. In *Proc. of EMNLP and IJCNLP 2019*, pages 4925–4936, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NAACL-HLT 2019*, pages 4171–4186, Minneapolis, Minnesota. ACL.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1):31 – 71.

Markus Eberts and Adrian Ulges. 2019. Span-based Joint Entity and Relation Extraction with Transformer Pre-training. In *24th European Conference on Artificial Intelligence (ECAI 2020)*, pages 2006 – 2013.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Thomas A. Runkler. 2019. Neural Relation Extraction within and across Sentence Boundaries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6513–6520. AAAI Press.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction. In *Proc. of COLING 2016*, pages 2537–2547, Osaka, Japan. The COLING 2016 Organizing Committee.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In *Proc. of EMNLP 2018*, pages 4803–4809, Brussels, Belgium. ACL.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó. Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 Task 8: Multi-way Classification of Semantic Relations Between Pairs of Nominals, booktitle = Proc. of the 5th International Workshop on Semantic Evaluation. SemEval '10, pages 33–38, Stroudsburg, PA, USA. ACL.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Shantanu Kumar. 2017. A Survey of Deep Learning Methods for Relation Extraction. *CoRR*, abs/1705.03645.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proc. of EMNLP 2017*, pages 188–197, Copenhagen, Denmark. ACL.

V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet physics. Doklady*, 10:707–710.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In *Proc. of EMNLP 2018*, pages 3219–3232, Brussels, Belgium. ACL.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A General Framework for Information Extraction using Dynamic Span Graphs. In *Proc. of NAACL-HLT 2019*, volume 1, pages 3036–3046, Minneapolis, Minnesota. ACL.

Makoto Miwa and Mohit Bansal. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In *Proc. of ACL 2016*, pages 1105–1116, Berlin, Germany. ACL.

Daniel Müllner. 2011. Modern hierarchical, agglomerative clustering algorithms. *CoRR*, abs/1109.2378.

Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. 2020. Reasoning with Latent Structure Refinement for Document-Level Relation Extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1546–1557, Online. Association for Computational Linguistics.

Dat Quoc Nguyen and Karin Verspoor. 2019. End-to-end neural relation extraction using deep biaffine attention. In *Advances in Information Retrieval*, pages 729–738, Cham. Springer International Publishing.

Chris Quirk and Hoifung Poon. 2017. Distant Supervision for Relation Extraction beyond the Sentence Boundary. In *Proceedings of EACL: Volume 1, Long Papers*, pages 1171–1182, Valencia, Spain. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labeled Text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Inter-sentence Relation Extraction with Document-level Graph Convolutional Neural Network. In *Proc. of the 57th ACL*, pages 4309–4316, Florence, Italy. Association for Computational Linguistics.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A Hierarchical Multi-Task Approach for Learning Embeddings from Semantic Tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6949–6956.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-vector Spaces. In *Proc. of EMNLP-CoNLL 2012*, pages 1201–1211, Stroudsburg, PA, USA. ACL.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4):521–544.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance Multi-label Learning for Relation Extraction. In *Proceedings of EMNLP-CoNLL 2012*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Hengzhu Tang, Yanan Cao, Zhenyu Zhang, Jiangxia Cao, Fang Fang, Shi Wang, and Pengfei Yin. 2020. HIN: Hierarchical Inference Network for Document-Level Relation Extraction. In *Advances in Knowledge Discovery and Data Mining*, pages 197–209, Cham. Springer International Publishing.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously Self-Attending to All Mentions for Full-Abstract Biological Relation Extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884, New Orleans, Louisiana. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Hong Wang, Christfried Focke, Rob Sylvester, Nilesh Mishra, and William W. J. Wang. 2019. Fine-tune Bert for DocRED with Two-step Process. *ArXiv*, abs/1909.11898.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, abs/1910.03771.

Shanchan Wu and Yifan He. 2019. Enriching Pretrained Language Model with Entity Information for Relation Classification. *CoRR*, abs/1905.08284.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A Large-Scale Document-Level Relation Extraction Dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential Reasoning Learning for Language Representation. In *Proc. of EMNLP*, pages 7170–7186, Online. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation Classification via Convolutional Deep Neural Network. In *Proc. of COLING 2014*, pages 2335–2344, Dublin, Ireland. Dublin City University and ACL.

Dongxu Zhang and Dong Wang. 2015. Relation Classification via Recurrent Neural Network. *CoRR*, abs/1508.01006.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proc. of the EMNLP 2017*, pages 35–45. ACL.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. In *Proc. of ACL 2017*, pages 1227–1236, Vancouver, Canada. ACL.

Peng Zhou, Suncong Zheng, Jiaming Xu, Zhenyu Qi, Hongyun Bao, and Bo Xu. 2017. Joint Extraction of Multiple Relations and Entities by Using a Hybrid Neural Network. In *Proc. of CCL 2017*, pages 135–146.

WER–BERT: Automatic WER Estimation with BERT in a Balanced Ordinal Classification Paradigm

Akshay Krishna Sheshadri* **Anvesh Rao Vijjini*** **Sukhdeep Kharbanda**

Samsung R&D Institute India - Bangalore
`{a.sheshadri,a.vijjini,sukhdeep.sk}@samsung.com`

Abstract

Automatic Speech Recognition (ASR) systems are evaluated using Word Error Rate (WER), which is calculated by comparing the number of errors between the ground truth and the transcription of the ASR system. This calculation, however, requires manual transcription of the speech signal to obtain the ground truth. Since transcribing audio signals is a costly process, Automatic WER Evaluation (e-WER) methods have been developed to automatically predict the WER of a speech system by only relying on the transcription and the speech signal features. While WER is a continuous variable, previous works have shown that positing e-WER as a classification problem is more effective than regression. However, while converting to a classification setting, these approaches suffer from heavy class imbalance. In this paper, we propose a new balanced paradigm for e-WER in a classification setting. Within this paradigm, we also propose WER-BERT, a BERT based architecture with speech features for e-WER. Furthermore, we introduce a distance loss function to tackle the ordinal nature of e-WER classification. The proposed approach and paradigm are evaluated on the Librispeech dataset and a commercial (black box) ASR system, Google Cloud's Speech-to-Text API. The results and experiments demonstrate that WER-BERT establishes a new state-of-the-art in automatic WER estimation.

1 Introduction

ASR systems are ubiquitous now. They are available across applications such as Voice Assistants, Assisted Living or Hands free device usage. However, with the widespread usage of ASR systems, there comes a heavy need for ASR Evaluation as well - to select, compare or improve alternate ASR systems. WER is widely considered as the standard metric for ASR evaluation. A higher WER means a higher percentage of errors between the ground truth and the transcription from the system. WER is calculated by aligning the two text segments using string alignment in a dynamic programming setting. The formula is as follows:

$$WER = \frac{ERR}{N} \tag{1}$$

where ERR is the sum of errors (Insertions, Deletions or Substitutions) between the transcription and the ground truth. N is number of words in ground truth. As evident from this equation, the presence of ground truth is necessary for the calculation of errors, and hence, for WER. However, manual transcription of speech at word level is a expensive and dilatory process. Hence, the need for an automatic ASR evaluation is important but few attempts have addressed this. Furthermore, for effectively training, evaluating and judging the performance of an ASR system, WER calculation needs to be done on adequate hours of data. As this test set increases, a more accurate estimation of WER is possible. Since automatic WER evaluation does not have the bottleneck of manual transcription, it can be calculated over large test sets leading to more accurate estimates. Because of the immense popularity of attention based architectures for text classification (Madasu and Rao, 2019, 2020a; Rao et al., 2020; Madasu and Rao, 2020b), we propose the transformer (Vaswani et al., 2017) encoder architecture — Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) for . BERT is pretrained on huge amounts of open domain language and is extensively used for its effectiveness in natural language understanding tasks. By pretraining on such data, the model gains knowledge of general domain language structure which aids in predicting

---

*The authors contributed equally to the work.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3661–3672
April 19 - 23, 2021.

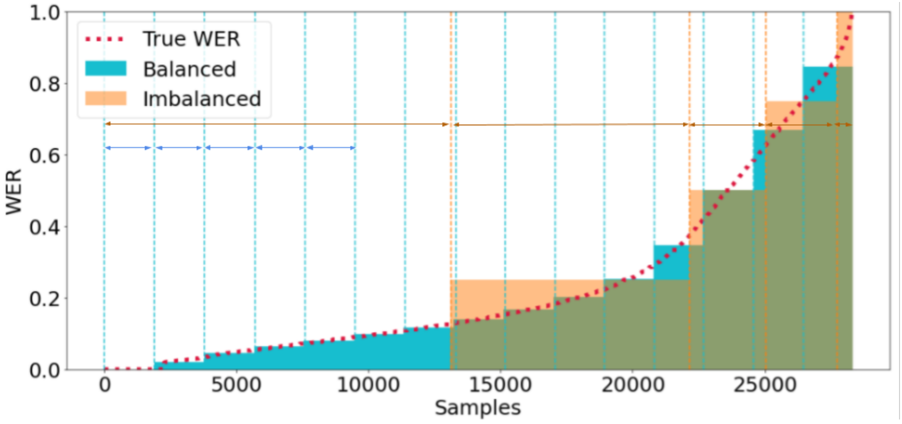


Figure 1: Distribution of WER classes of the 100 hr dataset, using proposed Balanced approach and using the Elloumi et al. (2018) imbalanced approach. The space between two consecutive vertical lines indicates the size of a respective class. The blue lines are evenly spaced, whereas the orange lines are spaced irregularly, indicating an imbalanced distribution.

speech errors which are typically observed as a deviation from the general syntax and semantics of a sentence. While previous approaches address e-WER (Ali and Renals, 2018; Elloumi et al., 2018, 2019) in classification settings, their models suffer with gross imbalance in the WER classes. To address this issue, we present a training framework which will always consist of training on equal sized classes no matter the true WER distribution. Additionally, these previous e-WER classification tasks assume that there is no inherent relative ordering to the classes. However, this is not the case with WER classification since a misclassification closer to the ground truth will lead to a lower mean absolute error (MAE) in the WER prediction. Such classification tasks are called Ordinal classification(Frank and Hall, 2001).

The overall contributions of our paper can be summarized follows:

(i) A new balanced paradigm for training WER as a classification problem which addresses the label imbalance issue faced by previous works.

(ii) WER-BERT - We find that language model information is helpful in predicting WER and hence propose a BERT based architecture e-WER.

(iii) Distance Loss to address the ordinal nature of WER classification which penalizes misclassifications differently based on how far off the predicted class is when compared to the real class.

2 Related Work

While the importance of an automatic WER prediction system is immense, there have not been many works directly addressing it. Related works such as exploring the word-level confidence in ASR prediction are abundant (Seigel and Woodland, 2011; Huang et al., 2013). There have also been works predicting the errors or error estimates as well in some form (Ogawa and Hori, 2015, 2017; Seigel and Woodland, 2014; Yoon et al., 2010). These approaches either predict some of the errors described in WER prediction or alternate metrics to rate ASR systems such as accuracy or error type classification. However, they lack calculation of the complete WER score. Transcrater (Jalalvand et al., 2016) was one of the first works which aim at predicting WER directly. They propose a neural network in a regression setting trained on various features such as parts of speech, language model, lexicon, and signal features. However, more recent approaches(Ali and Renals, 2018; Elloumi et al., 2018, 2019) phrase WER prediction as a classification approach. Ali and Renals (2018) propose two types of models based on the input available — the glassbox model which uses internal features of the target ASR system such as its confidence in transcribing the audio clip; and the black box model which only uses the transcripts and other features generated from the transcript such as the word and the grapheme count. They propose a bag-of-words model along with additional transcription features such as duration for e-WER.

The black box setting is a harder task since ASR model features such as the average log likelihood and the transcription confidence can give a good indication on how many errors may have occurred during the automatic transcription. However, the black box approach is not specific to the architectural design of an ASR system and can be used with any ASR system without access to its internal metrics such as the aforementioned confidence. Thus our proposed approach is trained in a black box setting.

Elloumi et al. (2018, 2019) build a CNN based model for WER classification. We built models based on them as baselines to evaluate WER-BERT's performance. They are further explained in the Sections 4 and 6.

ASR errors often make a transcription ungrammatical or semantically unsound. Identifying such constructs is also reflected in the dataset of Corpus of Linguistic Acceptability(CoLA) (Warstadt et al., 2019b,a). CoLA is a dataset intended to gauge at the linguistic competence of models by making them judge the grammatical acceptability of a sentence. CoLA is also part of the popular GLUE benchmark datasets for Natural Language Understanding (Wang et al., 2018). BERT (Devlin et al., 2018) is known for outperforming previous GLUE state-of-the-Art models, including the CoLA dataset.

3 Dataset

For our experiments, we have used the Librispeech dataset (Panayotov et al., 2015) which is a diverse collection of audio book data along with the ground text. It has around 1000 Hours of audio recordings with different levels of complexity. We pass these audio clips through an ASR system to get its transcripts and the WER is calculated by comparing it with the ground text. This paper reports findings in the experiments run with Google Cloud's Speech-to-Text API. We chose this commercial ASR system, rather than reporting results on an internal ASR system, since it's easily accessible through the google-api and the results are reproducible. For our experiments, we have used the 10 and 100 hour datasets and made a 60:20:20 split into train, dev and test sets for each dataset. As can be seen in Table 1 of Section B of the Appendix, the characteristics of the 100 and 10 hour datasets are quite different. However, within a dataset, the train, dev and test sets have similar distributions of WER and other characteristics. Table 1 lists a few examples with each row having the ground text, the transcript obtained from Google Cloud's Speech-to-Text API, the True WER and the WER predicted by our proposed model.

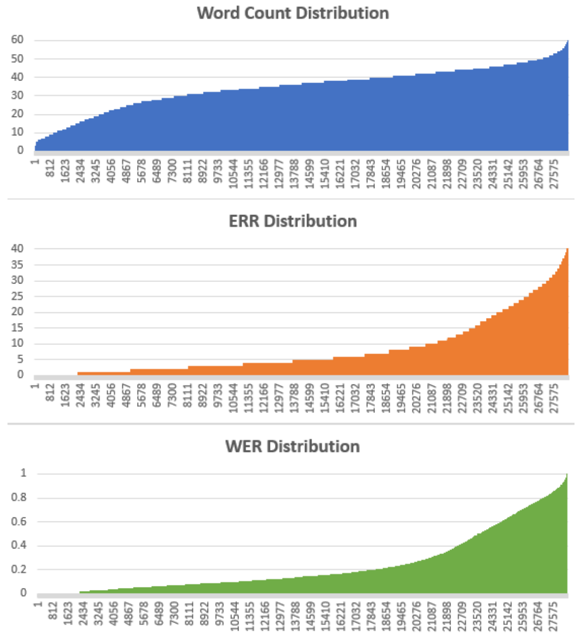


Figure 2: Distribution of WER, ERR and Word Count on our Training (Train 100hr) set.

4 Single Task and Double Task for WER Estimation

As shown in Equation 1, the WER of an utterance is the fraction obtained by the division of 2 integers — Errors per sentence (ERR), which is the total number of insertions, deletions and substitutions needed to convert an ASR's transcript to the ground text, and the word count of the ground text (N). Since the WER of a sentence is a continuous variable between 0 and 1 (mostly), a common way to model this is through a regression model. Elloumi et al. (2018, 2019) instead present a way to turn this into a classification problem for e-WER. They experiment with various combinations of text and audio signal inputs and show that a classification approach outperforms its corresponding regression approach trained on the same inputs. Elloumi et al. (2018)'s approach estimates WER directly with a 6 class classification model (with classes corresponding to $0\%, 25\%, 50\%, 75\%, 100\%$ and 150%). Once the model is trained, the predictions are calculated as follows:

$$WER_{Pred}(s) = P_{softmax}(s) \cdot WER_{fixed} \quad (2)$$

where s is a sample, WER_{Pred} is the predicted WER for s, $P_{softmax}(s)$ is the softmax probability distribution output by the classification model, $WER_{fixed} = [0, 0.25, 0.5, 0.75, 1.0, 1.5]$ is the fixed vector and '$\cdot$' is the dot product operator. We

---

https://cloud.google.com/speech-to-text.

call this approach as the Single Task method for e-WER. Alternatively, Ali and Renals (2018) present another classification approach for e-WER. They argue that the calculation of WER relies on two distinct calculations — ERR and N. Since both of them are discrete integers, they propose two independent classification problems to the estimate errors in the sentence $ERR_{est.}$ and to estimate the Word Count of the ground text $N_{est.}$. The predicted WER is then calculated as $(ERR_{est.}/N_{est.})$. We call this approach as the Double Task method for e-WER.

4.1 The Problem of Class Imbalance

While these approaches had good results, both of them suffer from the problem of imbalanced classes. As we can seen from Figure 2, the ERR and the WER are highly imbalanced. The nature of WER is such that it is very less likely for this task to be balanced for any data and any ASR system. This imbalance leads to poor performance due to certain WER or ERR classes having very few samples in them. With Elloumi et al. (2018), all the true WER's are 'cast' to the nearest multiple of 0.25. It can be seen from Figure 2 that the number of samples belonging to a class varies tremendously in this imbalanced setting. This leads to the model having poor performance, especially in the higher WER ranges. Moreover, different ASR systems will have their own distributions and some may be relatively well balanced in some ranges while other may be much worse. This approach is not scalable and it fails to generalize a method for creating balanced class distributions, irrespective of the ASR system.

4.2 Proposed Balanced Division

We propose an alternate paradigm for creating WER class distributions. We extend the single task setting for e-WER to a balanced WER class distribution irrespective of the dataset and ASR system. Instead of fixing a list of WER values based on a factor such as 0.25 to represent the classes, the total number of classes desired K is decided. Let $w_1, w_2, w_3, ...w_D$ be the WERs of individual samples ordered in an ascending manner where D is the total number of samples in the corpus. Then the number of samples in each of the K classes when divided uniformly will be $n_s = (D/K)$. A class C_i will be defined as the samples with WER in the range: $[w_{(i-1)*n_s}, w_{(i)*n_s}]$ where $i \in \{1, 2, ...K\}$. This is shown in Figure 1 where $K = 15$ classes are made with equal number of samples in each. The WER value WER_{Fixed_i} associated with each class C_i is defined as the mean WER of that class:

$$WER_{Fixed_i} = \sum_{j=(i-1)*n_s}^{(i)*n_s} \frac{w_j}{n_s} \qquad (3)$$

Once WER_{Fixed} is calculated, we use Equation 2 to compute the predicted WER of a sample using $P_{softmax}$ calculated from a neural network model. Apart from being balanced, this approach also has the benefit of generating classes which fits the true WER curve better than the previous approach as show in Figure 1. It's important to understand that while the ERR estimation in the Double Task setting of Ali and Renals (2018) is also imbalanced, it can't be mapped into arbitrary classes based on ordering. Since WER is a continuous variable, unlike ERR, it is possible to decide the boundary for a WER class arbitrarily to create balanced sets.

5 WER-BERT

In this section we explain our proposed architecture WER-BERT, which is primarily made of four sub-networks. Our architecture is shown with details in Figure 3.

Signal Sub-network: Elloumi et al. (2018) use the raw signal of the audio clip to generate features such as MFCC and Mel Spectrogram. They are features commonly used in the design of ASR systems, particularly systems which use an acoustic model and furthermore these features aid their model performance. These signal features are passed through the m18 architecture (Dai et al., 2017). m18 is a popular deep convolutional neural network (CNN) used for classification with audio signal features. This CNN model has 17 convolutional+max pooling layers which is followed by global average pooling. L2 Regularization of $1e-4$ and Batch Normalization are added after each of the convolutional layers.

Numerical Features Sub-network: Ali and Renals (2018) black box models had two major components — text input and numerical features. These numerical features are important to the model as they contain information regarding the number of errors. For instance, in ASR systems, there are errors if a user speaks too fast or too slow and this is directly reflected in the du-

---

Refer to Section A.2 of the appendix for tuning of the class hyperparameter K

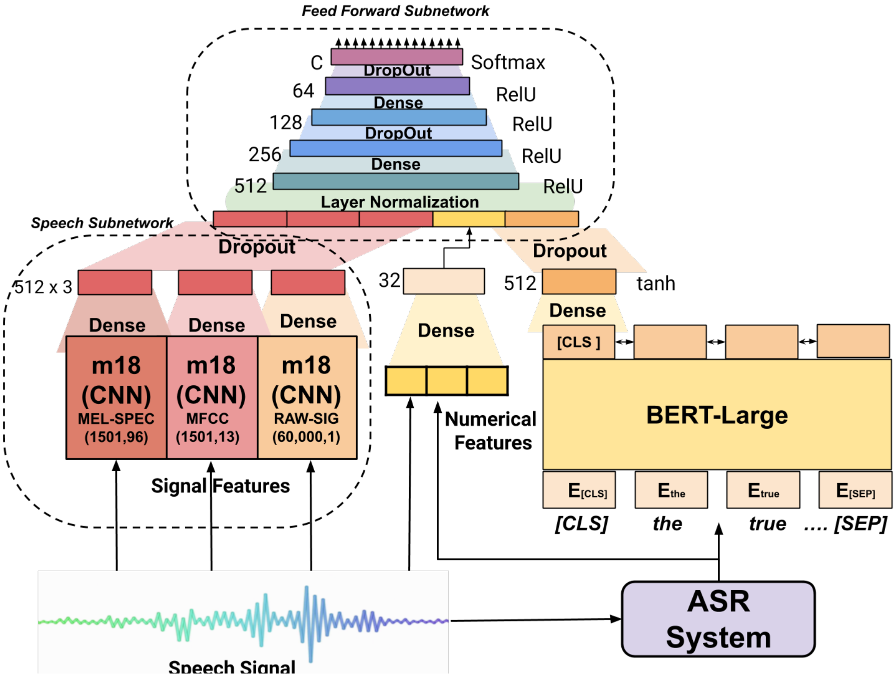


Figure 3: Proposed WER-BERT

ration and word count features. The numerical features we have used are Word Count, Grapheme Count and Duration. These features are concatenated and passed through a simple feed forward network which is used to upscale the numerical features fed into the model (from 3 to 32).

BERT: Bi-directional Encoder Representations (BERT) (Devlin et al., 2018) is a pre-trained unsupervised natural language processing model. It is a masked language model which has been trained on a large corpus including the entire Wikipedia corpus. The transcription samples from the ASR system, are passed through the associated tokenizer which gives a contextual representation for each word. It also adds 2 special tokens — the [CLS] token at the beginning and the [SEP] token at the end of the sentence. We have used the BERT-Large Uncased variant. The large variant has 24 stacked transformer Vaswani et al. (2017) encoders. It gives an output of the shape (Sequence Length X 1024) of which only the 1024 shaped output corresponding to the [CLS] token is used. In WER-BERT, BERT weights are fine tuned with rest of architecture during training.

Feed Forward Sub-Network: This sub-network is a deep fully connected network which is used to concatenate and process the features generated by the sub-networks predating it (BERT, numerical sub-network and the signal sub-network). It has 4 hidden layers (512, 256, 128 and 64 neurons) followed by the output softmax layer. Dropout regularization is added to prevent overfitting considering the large amount of parameters. To account for outputs from the eclectic sub-networks with disparate distributions, we further add Layer Normalization (Ba et al., 2016) before concatenation. Normalization is important to lessen the impact of any bias the network may learn towards one or the other representations.

Distance Loss for Ordinal Classification: Typical classification problems deal with classes which are mutually exclusive and independent such as sentiment prediction or whether an image is a cat or a dog. In such a setting, classification accuracy is the most important metric and there is no relation or relative ordering between the classes. However, e-WER in a classification setting is an ordinal classification problem (Frank and Hall, 2001). Previous approaches which propose WER estimation as classification tasks ignore this idea (Elloumi et al., 2018; Ali and Renals, 2018; Elloumi et al., 2019). While the classification accuracy is important, it is more important that given a sample is misclassified, the predicted label is close to the true label. For instance, if the true label corresponds to the WER class of 0.1, a prediction of 0.2 and a prediction of 0.7 are treated the same in the current classification scenario. Since we want the prediction to be as close as possible, if not exactly the same, we introduce a “distance” loss which is

Ground Truth	Google Cloud's Speech-to-Text Transcription	True WER	Predicted WER
one historian says that an event was produced by napoleon's power another that it was produced by alexander's	when is dorian says that an event was produced by napoleon's power another that it was produced by alexander's	16.7	16.5
rynch watched dispassionately before he caught the needler jerking it away from the prisoner the man eyed him steadily and his expression did not alter even when rynch swung the off world weapon to center its sights on the late owner	wrench watch dispassionately before he caught a kneeler jerking it away from the prisoner the man i can steadily and his expression did not alter even when wrench swampy off world weapon to center its sights on the late owner	21.9	22.1
of acting a father's part to augustine until he was fairly launched in life he had a child of his own	acting a father's part 2 augustine until he was fairly launched in life	42.8	42.7
supported by an honorable name how could she extricate herself from this labyrinth to whom would she apply to help her out of this painful situation debray to whom she had run with the first instinct of a woman towards the man she loves and who yet betrays her	supported by an honorable name how could you extricate herself in this labyrinth to whom would she apply to help her out of this painful situation dubray to whom should run the first instinct of a woman towards the man she loves and who yep betrays her	14.3	14.4
seventeen twenty four	1724	100.0	30.7
saint james's seven	st james 7	100.0	32.1
mamma says i am never within	mama says i am never with him	50.0	13.44

Table 1: Some examples of the proposed approach's WER Prediction

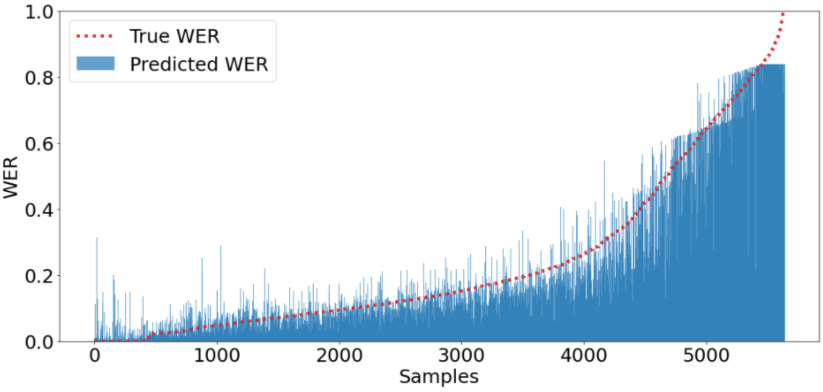


Figure 4: WER Predicted by WER-BERT SOTA model as compared to the True WER of all samples (Test 100hr)

$L_{custom}(s) = L_{\theta}(s) + \alpha * \gamma$ and γ is as follows

$$|y_{pred}(s) \cdot WER_{fixed} - y_{true}(s) \cdot WER_{fixed}| \tag{4}$$

where s is a sample, α is a hyperparameter (we have used $\alpha = 50$ in our experiments), $L_{\theta}(s)$ is the cross entropy loss, $y_{true}(s)$ is a one hot vector representing the true WER class of s and y_{pred} is the estimated probability distribution of s over all the classes output by the softmax of the classification model.

6 Experiments And Baselines

For each of the experiments below, the training is repeated for 10 runs and we report the average performance of all the runs on the test set. For all the experiments, we use Crossentropy as the loss function and MAE of WER as the evaluation metric.

6.1 BOW

Bag of Words + Num. Feat.(Black Box): Following (Ali and Renals, 2018), we build a black box model for the double task estimation of ERR and word count.The word count estimation task was treated as a 46 class classification model with class 1 corresponding to word count of 2, class 46 with word count of 47 (90^{th} percentile). Similarly, the ERR estimation task was modelled as a 20 class classification problem with class 0 corresponding to no errors and class 19 corresponding to 19 errors(90^{th} percentile). Both of these tasks are handled by logistic regression models which use Bag of Words features for both the words present in the sentence as well as the graphemes (monograms

---

Refer to Section A.1 of the appendix for tuning of the Distance Loss hyperparameter α

and bigrams) of the sentence. These features are concatenated with numerical features such as word count, grapheme count and the duration and then fed to a feedforward network. Dropouts are also added after each layer to prevent overfitting.

6.2 CNN

Following, Elloumi et al. (2018) for e-WER. These CNN models use ASR transcript and signal features (Raw signal, MFCC and Mel Spectrogram) as inputs and CNN for learning features from the textual transcription itself.

Text Input: The text input is padded to T words (where T was taken as 50, the 95^{th} percentile of the sentence length) and was transformed into a matrix $Embeddings$ of the size NXM where M is the embedding size (300). These embeddings were obtained from GloVe (Pennington et al., 2014). The CNN architecture is Kim (2014).

Signal Inputs:

RAWSIG: This input is obtained by sampling the audio clip at 8KHz and max duration was set to 15s; shorter audio clips were padded while the longer ones were clipped. While using this with the M18 architecture, it was further down-sampled to 4KHz due to memory constraints and its dimension is $60000X1$.

MELSPEC: This input was calculated with 96 dimensional vectors, each of which corresponds to a particular Mel frequency range. They were extracted every 10ms with an analysis window of 25ms and its dimension is $1501X96$.

MFCC: This input was calculated by computing 13 MFCCs every 10ms and its dimension is $1501X13$.

These signal features are used as an input to the M18 architecture Dai et al. (2017) (refer to Section 5: Signal sub-network). For joint use of both text and signal inputs, the outputs of the text and signal sub-networks are followed by a hidden layer (512 processing units) whose outputs are concatenated (with a dropout regularization of 0.1 being applied between the hidden layer and the concatenation layer). This is followed by 4 hidden layers (of 512, 256, 128 and 64 neurons) and the output layer with Dropout regularization added to prevent overfitting due to the large amount of parameters

CNN-text (balanced): While Elloumi et al. (2018) show that just CNN-text isn't enough to get good results. This experiment provides a baseline to compare against Bert-large since that does not have signal features as well. This model is trained in our balanced class setting with 15 classes explained in Section 4.2.

CNN-text + RAWSIG (Double Task): The best architecture in Elloumi et al. (2018) was trained in the double task setting proposed by (Ali and Renals, 2018). The ERR estimation was modelled as a 20 class task and the word count estimation was modelled as a 46 class task.

CNN-text + RAWSIG (Elloumi et al., 2018): This was the best architecture proposed by Elloumi et al. (2018). This is a single task approach which uses the imbalanced WER class distribution (classes corresponding to WER of 0, 0.25...1.5 WER).

CNN-text + RAWSIG (Balanced): Same model as CNN-text + RAWSIG but trained in our balanced class setting with 15 classes explained in Section 4.2.

CNN-text + MFCC + MELSPEC + RAWSIG (Balanced): Instead of using only RAWSIG input, we pass all 3 signal features into their respective m18 models. The outputs of the KIM CNN model (for the text input) and each of these m18 models are concatenated and processed in same fashion as WER-BERT explained in Section 5.

6.3 BERT architectures

Experiments are carried with the architecture described in Section 5. We carry out Ablation studies to identify important input features in isolation. We start with the full architecture shown in section 5 and subsequently remove sub-networks to compare performance. All the experiments except for the distance loss one are carried out with Crossentropy as the loss and MAE of WER as the evaluation metric and are trained with the balanced classes paradigm.

7 Results and Discussion

Table 2 captures the results of proposed approach with previous approaches along with ablation studies for the proposed approach. While the WER of the 10 hour dataset is low (≈10) and the 100 hour dataset is high (≈20), we see that the proposed approach models both effectively. Figure 4 shows that WER-BERT's estimation of WER closely follows the true WER curve. Furthermore, due to the proposed balanced paradigm, it is able to predict well in the mid-high WER classes even with less samples in this region. Figure 5 shows the com-

	Model	MAE WER 100 hour	RMSE WER 100 hour	MAE WER 10 hour	RMSE WER 10 hour
BoW	Bag of Words + Num. Feat.(Black Box) (Ali and Renals, 2018)	16.45	26.76	12.65	19.66
CNN	CNN-text (Balanced)	14.44	18.9	13.08	18.79
	CNN-text + RAWSIG (Double Task)	14.09	18.11	19.36	24.79
	CNN-text + RAWSIG (Elloumi et al., 2018)/(Imbalanced)	11.34	16.74	13.3	17.48
	CNN-text + RAWSIG (Balanced)	9.65	13.64	11.3	15.89
	CNN-text + MFCC + MELSPEC + RAWSIG / Signal Feat. (Balanced)	9.35	11.84	10.11	15.17
WER-BERT	BERT-*large*	12.04	17.98	11.19	16.38
	BERT-*large* + Num. Feat.	11.03	16.09	9.92	14.58
	BERT-*large* + MFCC + MELSPEC + RAWSIG	8.15	11.43	9.12	14.17
	BERT-*large* + Num. Feat. + Signal Feat.	6.91	10.01	8.89	13.52
	BERT-*large* + Num. Feat. + Signal Feat. + Distance Loss	5.98	8.82	7.37	12.67

Table 2: Performance Comparison.

parison of Balanced and Imbalanced class setting. Comparing Figure 4 and 5, we see that the WER-BERT models much better in the lower and mid regions compared to the CNN balanced model.

7.1 Effectiveness of WER-BERT and Ablation Studies

The best WER-BERT model, with all Num., Signal feats. and distance loss outperforms other models in both MAE and RMSE metrics. In particular, we see that a BERT model outperforms the corresponding CNN model with the same inputs - 14.44 v/s 12.04 & 13.08 v/s 11.19 for the CNN-text and BERT-large model in the 100 hour and 10 hour data respectively (and similar results for the CNN-text + signal feat and BERT-large + signal feat models). We credit effectiveness of BERT at its language model information and ability to identify improbable word sequences which often correlate with transcription errors. Models such as CNN, while being effective, lack the backing of a language model and hence fail to do the same. While this is important, it alone fails to beat the earlier approaches which utilise signal features i.e. Making a WER prediction from just the transcription is not enough. Signal features tells us how an utterance was spoken along with background noise. This contains valuable information such as signal noise which correlates with higher WER. The effectiveness of WER-BERT is particularly evident in Table 1 where we see that it is able to predict WER which is very close to the True WER. We hypothesize this is due to irregular word sequences such as "..when is dorian says.." or "..the man i can steadily." being identified by BERT's language model. Furthermore, even when our model predicts a far off WER, it does so in a justified manner. The transcription "1724" and "st james 7" give a high True WER score whereas WER-BERT identifies these sequences as correct and probable and gives low scores. Since WER matches strings exactly, it ends up giving a 100% error in this case due to library's text processing limitations. Note that while we use BERT, the language model itself can be any transformer, since they all have the common idea that LM backing helps identifying speech errors in ways CNN models can't.

7.2 Effectiveness of proposed balancing setting

We use the best architecture reported by Elloumi et al. (2018): CNN-text + RAWSIG for comparing the three paradigms. Balanced outperforms the Double Task approach of Ali and Renals (2018) and Single Task approach Elloumi et al. (2018) (9.65 v/s 11.34 v/s 14.09 for the 100 Hour set). The class imbalance in their approach causes bias towards the largest classes during training, where most of the predictions end up. This especially harms the WER prediction in higher ranges where number of samples are less. Figure 5 shows the performance difference between CNN + RAWSIG models in the new balanced paradigm and the imbalanced Elloumi et al. (2018) setting. Since the model encounters heavily imbalanced labels in the later setting, its predictions also reflect the same. There are only two kinds of predictions - one corresponding to the lowest WER class (also largest in number as seen in Figure 1) and another in the range of second largest class. The information of

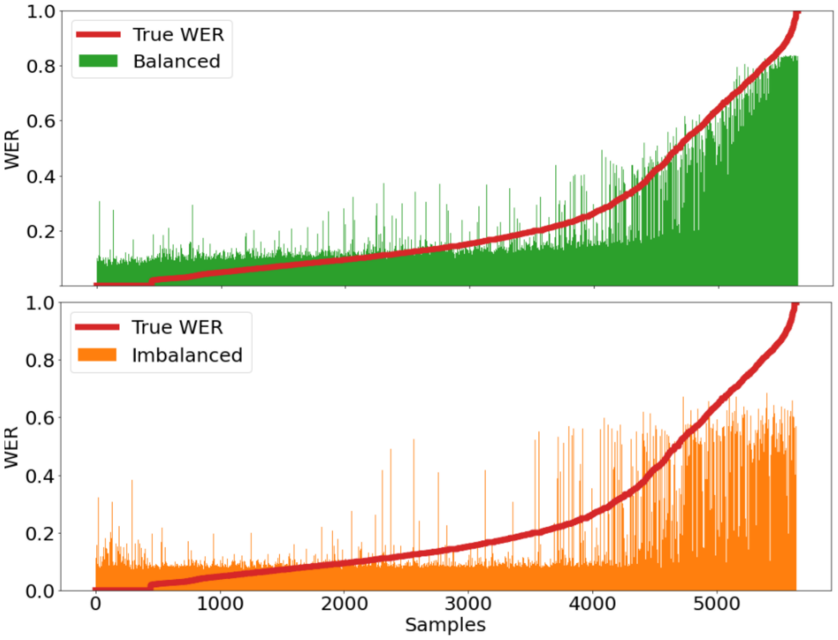
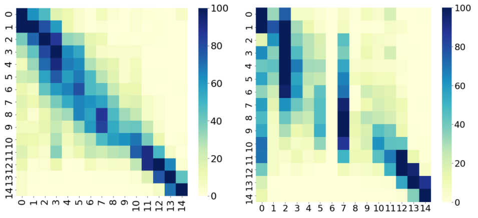


Figure 5: WER Predicted by the CNN + RAWSIG (Balanced) and CNN + RAWSIG (Elloumi et al., 2018) (Imbalanced) as compared to True WER for all samples (Test 100hr)



(a) with Distance Loss (b) w/o Distance Loss

Figure 6: Confusion Matrices of the BERT-*large* + All Features model

the other classes are mostly ignored due to being present in low numbers. Meanwhile, the balanced paradigm fits the slope of the curve after 40 WER well. The range of > 40 WER is tough for the model to predict as the number of samples available in this region is lesser than samples in < 40 WER (almost 70% of data). Despite this, balanced paradigm effectively divides this area into adequate number of classes for good performance. While the performance in the mid region (20-40 WER) is poor for this model, custom loss and BERT model in WER-BERT take care of this as seen in Figure 4.

7.3 Effectiveness of Distance loss

Addition of Distance loss to WER-BERT certainly improves the performance. In Figure 6 we see the confusion matrix of WER classification visualized with and without the custom loss. Due to the custom loss the models predicted class is much closer to the ground truth class represented by the diagonal of the matrix. This is due to distance loss' ability to penalize far off predictions which is lacking in typical classification loss functions such as crossentropy. We see an improvement of nearly 1 MAE with this.

8 Conclusion

We propose WER-BERT for Automatic WER Estimation. While BERT is an effective model, addition of speech signal features boosts the performance. Phrasing WER classification as a ordinal classification problem by training using a custom distance loss encodes the information regarding relative ordering of the WER classes into the training. Finally, we propose a balanced paradigm for training WER estimation systems. Training in a balanced setting allows proposed model to predict WER adequately even in regions where samples are scarce. Furthermore, this balanced paradigm is independent of WER prediction model, ASR system or the speech dataset, making it efficient and scalable.

References

Ahmed Ali and Steve Renals. 2018. Word error rate estimation for speech recognition: e-wer. In *Proceed-*

Number of WER classes $K = 15$

ings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 20–24.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das. 2017. Very deep convolutional neural networks for raw waveforms. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 421–425. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zied Elloumi, Laurent Besacier, Olivier Galibert, Juliette Kahn, and Benjamin Lecouteux. 2018. Asr performance prediction on unseen broadcast programs using convolutional neural networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5894–5898. IEEE.

Zied Elloumi, Olivier Galibert, Benjamin Lecouteux, and Laurent Besacier. 2019. Investigating robustness of a deep asr performance prediction system.

Eibe Frank and Mark Hall. 2001. A simple approach to ordinal classification. In *European Conference on Machine Learning*, pages 145–156. Springer.

Po-Sen Huang, Kshitiz Kumar, Chaojun Liu, Yifan Gong, and Li Deng. 2013. Predicting speech recognition confidence using deep learning with word identity and score features. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7413–7417. IEEE.

Shahab Jalalvand, Matteo Negri, Marco Turchi, José GC de Souza, Daniele Falavigna, and Mohammed RH Qwaider. 2016. Transcrater: a tool for automatic speech recognition quality estimation. In *Proceedings of ACL-2016 System Demonstrations*, pages 43–48.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Avinash Madasu and Vijjini Anvesh Rao. 2019. Sequential learning of convolutional features for effective text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5662–5671.

Avinash Madasu and Vijjini Anvesh Rao. 2020a. A position aware decay weighted network for aspect based sentiment analysis. *arXiv preprint arXiv:2005.01027*.

Avinash Madasu and Vijjini Anvesh Rao. 2020b. Sequential domain adaptation through elastic weight consolidation for sentiment analysis. *arXiv preprint arXiv:2007.01189*.

Atsunori Ogawa and Takaaki Hori. 2015. Asr error detection and recognition rate estimation using deep bidirectional recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4370–4374. IEEE.

Atsunori Ogawa and Takaaki Hori. 2017. Error detection and accuracy estimation in automatic speech recognition using deep bidirectional recurrent neural networks. *Speech Communication*, 89:70–83.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Vijjini Anvesh Rao, Kaveri Anuranjana, and Radhika Mamidi. 2020. A sentiwordnet strategy for curriculum learning in sentiment analysis. *arXiv preprint arXiv:2005.04749*.

Matthew Stephen Seigel and Philip C Woodland. 2011. Combining information sources for confidence estimation with crf models. In *Twelfth Annual Conference of the International Speech Communication Association*.

Matthew Stephen Seigel and Philip C Woodland. 2014. Detecting deletions in asr output. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2302–2306. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019a. Cola: The corpus of linguistic acceptability (with added annotations).

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019b. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Su-Youn Yoon, Lei Chen, and Klaus Zechner. 2010. Predicting word accuracy for the automatic speech recognition of non-native speech. In *Eleventh Annual Conference of the International Speech Communication Association*.

A Hyperparameter Tuning Details

In this section we explain the tuning and selection procedure of the proposed experiments' hyperparameters, namely the distance loss weight α and number of classes K. Furthermore, we also attempt to explain the behaviour of these variables on the performance in terms of MAE.

A.1 Tuning Distance Loss Hyperparameter α

Figure 7 shows the effect of varying α with effect on performance on the 100hr dataset with the WER-BERT model with and without custom loss. As evident, we get the lower MAE and RMSE for $\alpha = 50$. Lower or higher α leads to decrement in performance. Recalling Equation 4 from the main paper:

$$L_{custom}(s) = L_{\theta}(s) + \alpha * \gamma \tag{5}$$

where γ is:

$$|y_{pred}(s) \cdot WER_{fixed} - y_{true}(s) \cdot WER_{fixed}| \tag{6}$$

For low values of α (0.0001 to 0.001), in the above equations $L_{custom}(s)$ becomes equal to just $L_{\theta}(s)$ which the classification cross entropy loss. Hence, essentially low α performance tends to the line of No custom loss WER-BERT performance. On the other hand, higher α values weight the custom loss too much more than the cross entropy loss. While the decrement in performance isn't very high, moderate adverse effects still show that, regular classification cross entropy loss is required for WER-BERT training.

A.2 Tuning Class Hyperparameter K

Figure 8 shows the effect on performance of the CNN + RAWSIG model for varying MAE. We get the minimum MAE and RMSE at $K = 15$. Largely speaking, for lower K, RMSE and MAE suffer because the new classes no more accurately depicts the true WER. For example, in Figure 1 of main paper, we can see that if $K = 3$ or 4, WERs in a wide range from 40 to 100 will be clubbed into a single class despite their huge differences. This

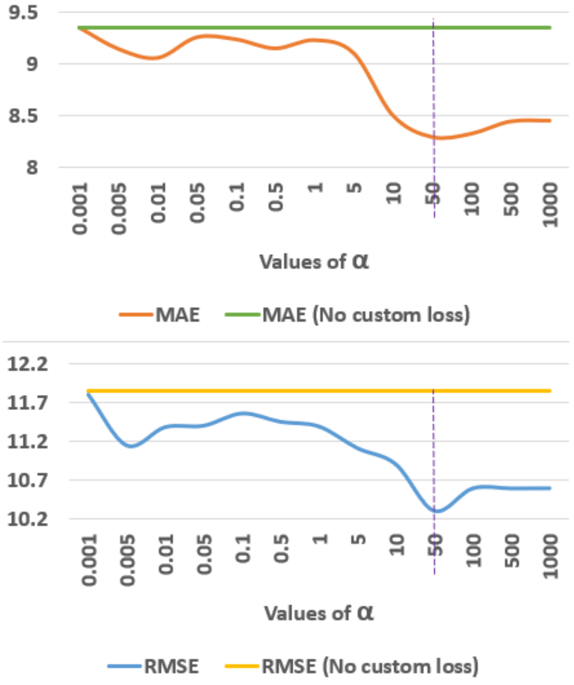


Figure 7: Distribution of MAE and RMSE for different values of α against no custom loss in WER-BERT. The values here are presented for the 100hr dataset.

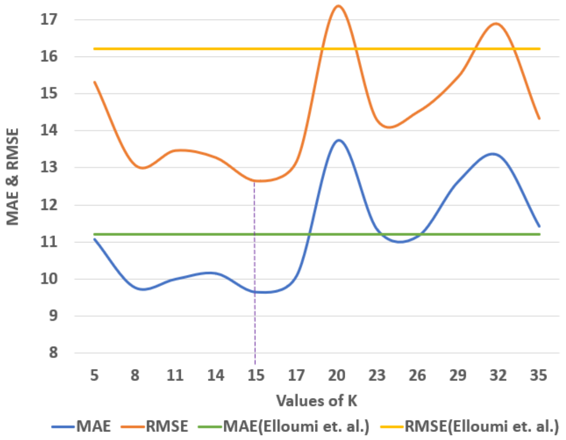


Figure 8: Distribution of MAE and RMSE for different values of K against Elloumi et al. (2018) approach. The values here are presented for the CNN + RAWSIG model on the 100hr dataset.

Datasets	ERR	Duration	Length	True WER
Train 100hr	8.16	12.74	34.84	23.42
Dev 100hr	8.13	12.70	34.64	23.47
Test 100hr	8.18	12.61	34.50	23.70
Train 10hr	2.65	7.29	20.44	12.97
Dev 10hr	2.28	7.15	19.96	11.40
Test 10hr	2.41	7.18	20.25	11.91

Table 3: Averaged Statistics regarding the Train 100hr and Test 100hr dataset.

will further make it hard for the model to predict WERs in this range.

On the other hand, higher K yields new classes accurate to true WER especially in the higher ranges(50 to 100), but at the cost of class size. For example at K=30, each class will have about 900 (28000/30) samples.Furthermore, since True WER is a highly imbalanced variable, there exists a long tail in the lower WER regions in Figure 1 of main paper. This WER region is divided into many classes, despite the WER range being largely small. For example for $K > 30$ the model will be forced to distinguish between nearly four classes with arbitrary samples of all 0 True WER. Just for $K = 15$, model encounters two classes which are actually both 0 WER. This number of redundant classes with arbitrary samples will only increase as K increases.

Furthermore, for a range of values of K, we see that the balanced CNN + RAWSIG model still outperforms the CNN + RAWSIG (Elloumi et al., 2018)'s performance. This further reinforces the efficacy of the balanced paradigm even with its senstivity to a hyperparameter.

B Dataset

Additional dataset details are present in Table 3.

Two Training Strategies for Improving Relation Extraction over Universal Graph

Qin Dai[1], **Naoya Inoue**[2], **Ryo Takahashi**[1,3], **Kentaro Inui**[1,3]
[1]Tohoku University, Japan
[2]Stony Brook University, USA
[3]RIKEN Center for Advanced Intelligence Project, Japan
`{daiqin, naoya-i, ryo.t, inui}@ecei.tohoku.ac.jp`

Abstract

This paper explores how the Distantly Supervised Relation Extraction (DS-RE) can benefit from the use of a Universal Graph (UG), the combination of a Knowledge Graph (KG) and a large-scale text collection. A straightforward extension of a current state-of-the-art neural model for DS-RE with a UG may lead to degradation in performance. We first report that this degradation is associated with the difficulty in learning a UG and then propose two training strategies: (1) Path Type Adaptive Pretraining, which sequentially trains the model with different types of UG paths so as to prevent the reliance on a single type of UG path; and (2) Complexity Ranking Guided Attention mechanism, which restricts the attention span according to the complexity of a UG path so as to force the model to extract features not only from simple UG paths but also from complex ones. Experimental results on both biomedical and NYT10 datasets prove the robustness of our methods and achieve a new state-of-the-art result on the NYT10 dataset. The code and datasets used in this paper are available at `https://github.com/baodaiqin/UGDSRE`.

1 Introduction

Relation Extraction (RE) is an important task in Natural Language Processing (NLP). RE aims to turn unstructured texts into structured Knowledge Graph (KG), which is typically stored as (e_1, r, e_2) triplets, where e_1 is a *head entity*, r is a relation and e_2 is a *tail entity*, such as (*aspirin*, *may_treat*, *pain*) and (*Guy Maddin*, *place_lived*, *Winnpeg*). RE can be formulated as a classification task to predict a predefined relation r from entity pair (e_1, e_2) annotated evidences.

One obstacle that is encountered when building a RE system is the generation of a large amount of manually annotated training instances, which is expensive and time-consuming. For coping with this difficulty, Mintz et al. (2009) propose Distant Supervision (DS) to automatically generate training samples via linking KGs to texts. They assume that if (e_1, r, e_2) is in a KG, then all sentences that contain (e_1, e_2) (hereafter, *sentence evidences*) express the relation r. It is well known that the DS assumption is too strong and inevitably accompanies the wrong labeling problem, such as the sentence evidences (1b and 2) below, which fail to express *may_treat* and *place_lived* relation respectively.

(1) a. ***Aspirin***$_{e_1}$ *is widely used for short-term treatment of* ***pain***$_{e_2}$*, fever or colds.*

b. *The tumor was remarkably large in size, and* ***pain***$_{e_2}$ *unrelieved by* ***aspirin***$_{e_1}$.

(2) *He is now finishing a documentary about* ***Winnipeg***$_{e_2}$*, the final installment of a personal trilogy that began with "Cowards Bend the Knee" (a 2003 film that also featured a hapless hero named* ***Guy Maddin***$_{e_1}$*).*

Recently, neural network models with attention mechanism have been proposed to alleviate the wrong labeling problem and attend informative sentence evidences such as (1a) (Lin et al., 2016; Ji et al., 2017; Du et al., 2018; Jat et al., 2018; Han et al., 2018a,b). However, there can be a large portion of entity pairs that lack such informative sentence evidences that explicitly express their relation. This makes Distantly Supervised Relation Extraction (DS-RE) further challenging (Sun et al., 2019).

For compensating the lack of informative sentence evidences, Dai et al. (2019) utilize multi-hop paths connecting a target entity pair (hereafter, *path*) over a KG as extra evidences for DS-RE. An example of such multi-hop KG path can be seen in Figure 1, where p_1 depicts a multi-hop KG path

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3673–3684
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

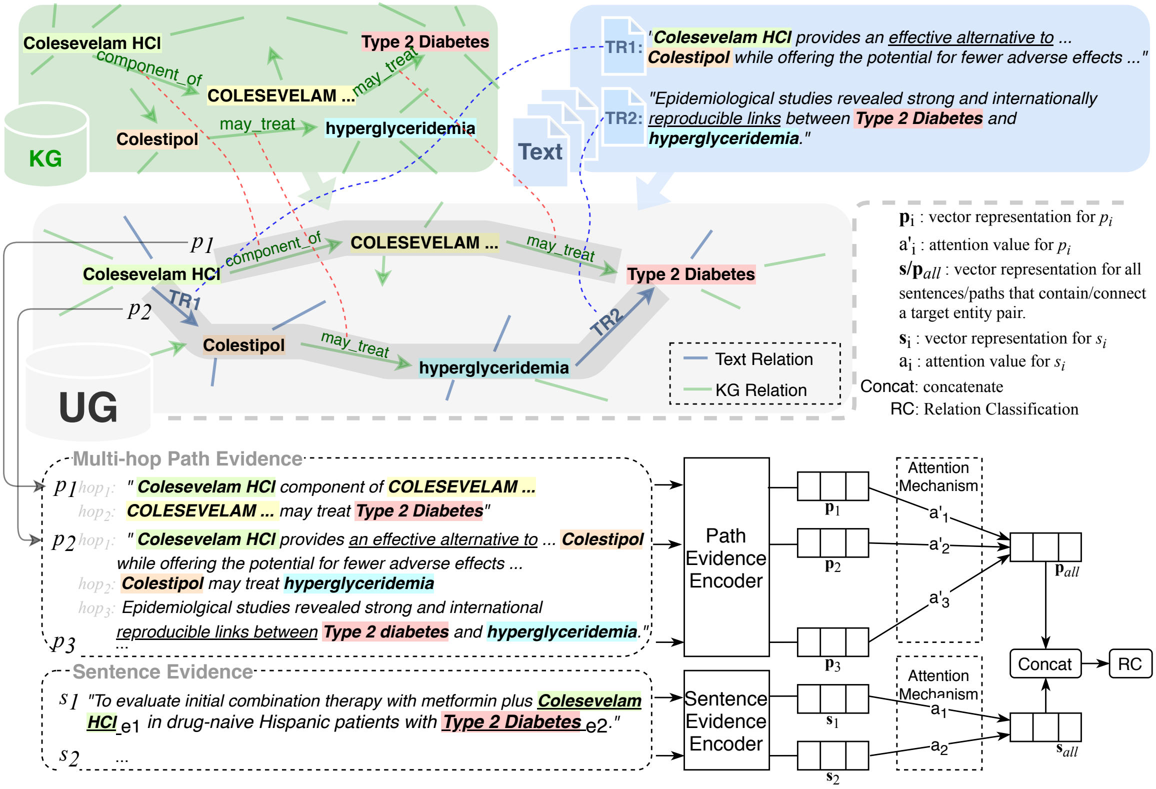


Figure 1: Overview of our UG-based framework, where **Colesevelam HCl** and **Type 2 Diabetes** are the target entities, **COLESEVELAM ...**, **Colestipol** and **hyperglyceridemia** are intermediate entities, each UG path consists of multiple hops and each hop represents a KG relation (such as "*Colestipol may treat hyperglyceridemia*") or Text (or Textual) relation (such as TR1 and TR2), which is the sentence containing two (target or intermediate) entities.

of the form of $e_1 \xrightarrow{component\_of} e_3 \xrightarrow{may\_treat} e_2$. The model of Dai et al. (2019) uses such multi-hop paths as additional features for predicting the relation between a given target entity pair (e_1, e_2), which is reported effective for performance improvement. However, KGs are often highly incomplete (Min et al., 2013) and may be too sparse to provide enough informative paths in practice, which may hamper the effectiveness of multi-hop paths.

Given this background, in this study, we take one step further, aiming for inducing maximal signals of distant supervision from both a KG and a large text collection (hereafter, *Text*). For this purpose, we consider using multi-hop paths over a Universal Graph (UG) as extra features for DS-RE. Here, we define a UG as a joint graph representation of both KG and Text, where each node represents an entity from KG or Text, and each edge indicates a KG relation or Textual relation, as shown in Figure 1. The path p_2 in the figure is an example of UG path, comprising a textual edge TR1, a KG edge *may_treat*, and another textual edge TR2. By augmenting the original KG with textual edges, one can expect far more chances to find informative path evidences between any given target entity pairs, because the number of such textual edges is likely to be much larger than the number of KG edges (Note that one can collect as many textual edges as needed from a raw text corpus with an entity linker). Extending a KG to a UG, therefore, may allow a DS-RE model to learn richer distant supervision signals.

The idea of using multi-hop paths over a UG is not necessarily new on its own. For example, Toutanova et al. (2015) propose to use a UG for knowledge graph completion, and Das et al. (2017b) propose a model trained to reason over a UG for question answering. However, there is no prior study that has explored the effective way to use a UG for the task of DS-RE from text. In fact, finding an effective way of using a UG for DS-RE is not as simple as it may seem. As we report in this paper, a straightforward extension of the Dai et al. (2019) model to the UG setting may result in performance degradation.

Motivated by this, in this paper, we address how one can make effective use of UG for DS-RE. We first report our observation that a straightforward extension of the Dai et al. (2019) model to the UG setting tends to allocate the majority of attention to only a limited set of UG paths such as short KG paths and miss out the learning from a wide range of UG paths (§4.1), which hinders performance gain. In order to alleviate the negative effect of the attention bias and realize the potential of UG paths, we propose two training (or debiasing) strategies: (1) Path Type Adaptive Pretraining (§4.2), which aims to improve the adaptability of the model to various UG paths; and (2) Complexity Ranking Guided Attention mechanism (§4.3), which enables the model to learn from both simple and complex UG paths. Experimental results on both biomedical and NYT10 (Riedel et al., 2010) datasets prove that: (1) UG paths have the potential to bring performance gain for DS-RE as compared with KG paths; (2) the proposed training methods are effective to fully exploit the potential of UG paths for DS-RE because the proposed methods significantly and consistently outperform several baselines on both datasets and especially achieve a new state-of-the-art result on the NYT10 dataset.

2 Related Work

To improve the performance of a DS-RE model, recently, researchers introduce various attention mechanisms. Lin et al. (2016) propose a relation vector based attention mechanism. Jat et al. (2018); Du et al. (2018) propose multi-level (e.g., word-level and sentence-level) structured attention mechanism. Ye and Ling (2019) apply both intra-bag and inter-bag attention for DS-RE. Han et al. (2018b) propose a relation hierarchy based attention mechanism. Han et al. (2018a) propose a joint model that adopts a KG embeddings based attention mechanism. Jia et al. (2019) propose an attention regularization framework to select informative sentence evidences for DS-RE. However, these models rely only on noisy sentence evidences from DS, neglecting the rich UG paths for DS-RE.

Besides the sentence evidences from DS, researchers also leverage external evidences for DS-RE. Ji et al. (2017) apply entity descriptions generated from Freebase and Wikipedia as extra evidences, Lin et al. (2017) utilize multilingual text as extra evidences and Vashishth et al. (2018) use multiple extra evidences including entity types, dependency and relation alias information for DS-RE. Alt et al. (2019) utilize pretrained language model as background information for DS-RE. Sun et al. (2019) apply relational table extracted from Web as supplementary evidences for DS-RE.

To apply DS-RE beyond sentence boundary, Quirk and Poon (2017) utilize syntactic information to extract relation from neighboring sentences. Zeng et al. (2017) apply two-hop KG paths identified from two-hop textual paths as extra evidences for DS-RE. Different from this work, we directly use the rich UG paths as extra evidences. Dai et al. (2019) extend the framework of Han et al. (2018a) by introducing multiple KG paths as extra evidences for DS-RE. Neelakantan et al. (2015); Das et al. (2017a) use multiple reasoning paths over Text and KG for relation prediction in the paradigm of Knowledge Graph Completion. Our work differs from the ones mentioned above in two ways: (i) We utilize the UG paths as extra evidences for the task of DS-RE from text, (ii) We take into account the factor of attention bias while encoding UG paths and propose two effective debiasing methods to exploit the potential of UG paths for DS-RE.

3 Base Model

We select the DS-RE model proposed by Dai et al. (2019) as our base model and extend it into our UG setting. Given a target entity pair (e_1, e_2), a bag of corresponding sentence evidences $S_r = \{s_1, ..., s_n\}$ and a bag of UG paths $P_r = \{p_1, ..., p_m\}$, the base model aims to measure the probability of (e_1, e_2) having a predefined relation r (including the empty relation NA). The base model consists of four main modules: KG Encoder, Sentence Evidence Encoder, Path Evidence Encoder and Relation Classification Layer, as shown in Figure 2.

3.1 KG Encoder

Suppose we have a KG containing a set of fact triplets $\mathcal{O} = \{(e_1, r, e_2), ...\}$, where each fact triplet consists of two entities $e_1, e_2 \in \mathcal{E}$ and their relation $r \in \mathcal{R}$. Here $\mathcal{E}$ and $\mathcal{R}$ stand for the set of entities and relations respectively.

The KG Encoder then encodes $e_1, e_2 \in \mathcal{E}$ and their relation $r \in \mathcal{R}$ into low-dimensional vectors $\mathbf{h}, \mathbf{t} \in R^d$ and $\mathbf{r} \in R^d$ respectively, where d is the dimensionality of the embedding space. The KG Encoder adopts TransE (Bordes et al., 2013) to score a given triplet. Specifically, given a triplet (e_1, r, e_2), TransE evaluates its plausibility

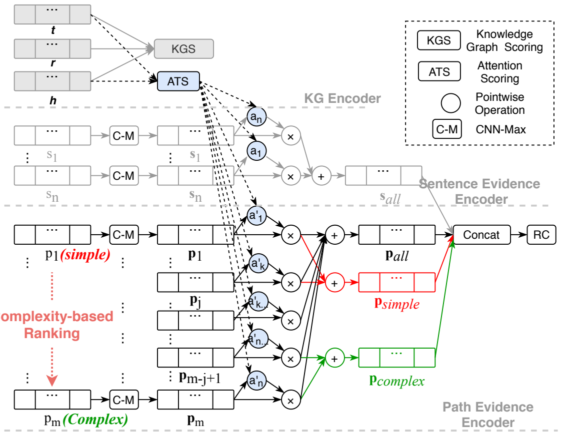


Figure 2: Overview of the base model and our proposed Complexity Ranking Guided Attention mechanism (§4.3). The base model takes the sentence evidences (e.g., s_1, ...) containing a target entity pair and the UG paths (e.g., p_1, ...) connecting the entity pair as input for predicting their relation. The KG embeddings of the entity pair (i.e., $\mathbf{h}$ and $\mathbf{t}$) are used for calculating the attention over these sentences and paths. The Complexity Ranking Guided Attention mechanism is proposed to force the model to attend both simple UG paths (e.g., $p_1 \sim p_j$) and complex ones (e.g., $p_{m-j+1} \sim p_m$).

via Equation 1:

$$f_r(e_1, e_2) = b - \|\mathbf{r}_{ht} - \mathbf{r}\|, \tag{1}$$

$$\mathbf{r}_{ht} = \mathbf{t} - \mathbf{h}, \tag{2}$$

where b is a bias constant and $\mathbf{r}_{ht}$ is a latent relation embedding for (e_1, e_2). The conditional probability can be formalized over all fact triplets $\mathcal{O}$ as follows:

$$P(e_1, r, e_2|\theta_{\mathcal{E}}, \theta_{\mathcal{R}}) = \frac{\exp(f_r(e_1, e_2))}{\sum_{r' \in \mathcal{R}} \exp(f_{r'}(e_1, e_2))} \tag{3}$$

where $\theta_{\mathcal{E}}$ and $\theta_{\mathcal{R}}$ are parameters for entities and relations respectively.

3.2 Sentence Evidence Encoder

Given a bag of sentence evidences $S_r = \{s_1, ..., s_n\}$, the Sentence Evidence Encoder applies CNN-Max (see Appendix §A.1) on each sentence, namely $\mathbf{s}_i = \text{CNN-Max}(s_i)$, to derive the sentence representations $\{\mathbf{s}_1, ..., \mathbf{s}_n\}$. The encoder then calculates the bag-level vector representation $\mathbf{s}_{\text{all}}$ via Equation 4:

$$\mathbf{s}_{\text{all}} = \sum_{i=1}^{n} a_i \mathbf{s}_i, \tag{4}$$

$$a_i = \frac{\exp(\langle \mathbf{r}_{ht}, \mathbf{x}_i \rangle)}{\sum_{k=1}^{n} \exp(\langle \mathbf{r}_{ht}, \mathbf{x}_k \rangle)},$$

$$\mathbf{x}_i = \tanh(\mathbf{W}\mathbf{s}_i + \mathbf{b})$$

where $\mathbf{r}_{ht}$ is from Equation 2, $\mathbf{W}$ is the weight matrix, $\mathbf{b}$ is the bias vector, a_i is the weight for the i-th sentence in S_r.

3.3 Path Evidence Encoder

Given a bag of UG paths $P_r = \{p_1, ..., p_m\}$ connecting an entity pair of interest (e_1, e_2), the Path Evidence Encoder encodes them into a bag-level vector representation $\mathbf{p}_{\text{all}}$. Since we represent a path as a sequence of words (or a long sentence), as shown in Figure 1, analogously to the Sentence Evidence Encoder, we apply a CNN-Max (see Appendix §A.1) to encode each path p_i, namely $\mathbf{p}_i = \text{CNN-Max}(p_i)$. The bag-level path representation $\mathbf{p}_{\text{all}}$ for P_r is then calculated via Equation 5:

$$\mathbf{p}_{\text{all}} = \sum_{i=1}^{m} a'_i \mathbf{p}_i, \tag{5}$$

$$a'_i = \frac{\exp(\langle \mathbf{r}_{ht}, \mathbf{x'}_i \rangle)}{\sum_{k=1}^{m} \exp(\langle \mathbf{r}_{ht}, \mathbf{x'}_k \rangle)},$$

$$\mathbf{x'}_i = \tanh(\mathbf{W}\mathbf{p}_i + \mathbf{b})$$

where a'_i is the weight for the i-th path in P_r.

3.4 Relation Classification Layer

The conditional probability of (e_1, e_2) having a relation r is formulated via Equation 6:

$$P(e_1, r, e_2|S_r, P_r, \theta_S, \theta_P) = \frac{\exp([\mathbf{o}]_r)}{\sum_{c=1}^{n_r} \exp([\mathbf{o}]_c)} \tag{6}$$

where $\mathbf{o} = \mathbf{M}[\mathbf{s}_{\text{all}}; \mathbf{p}_{\text{all}}] + \mathbf{d}$, θ_S, θ_P are the parameters in Sentence Evidence Encoder and Path Evidence Encoder, $\mathbf{M}$ is the representation matrix of relations, $\mathbf{d}$ is a bias vector, $\mathbf{o}$ is the output vector containing the prediction scores of all predefined relations, $[\mathbf{o}]_c$ is the prediction score for the relation c, and n_r is the total number of relations.

Given a training dataset consisting of triplets $\mathcal{O} = \{(e_1^1, r^1, e_2^1), (e_1^2, r^2, e_2^2), ...\}$, we minimize the objective function as follows:

$$J(\theta) = -\frac{1}{|\mathcal{O}|} \sum_{i=1}^{|\mathcal{O}|} \log P(e_1^i, r^i, e_2^i|\theta_{\mathcal{E}}, \theta_{\mathcal{R}}) + \log P(e_1^i, r^i, e_2^i|S_{r^i}, P_{r^i}, \theta_S, \theta_P) \tag{7}$$

The base model is optimized with Stochastic Gradient Descent (SGD). Following (Han et al., 2018a), we optimize $P(e_1^i, r^i, e_2^i|\theta_{\mathcal{E}}, \theta_{\mathcal{R}})$ and $P(e_1^i, r^i, e_2^i|S_i, P_i, \theta_S, \theta_P)$ in parallel.

4 Proposed Training Method

4.1 Problem of Attention Bias

While extending the base model into our UG setting, we observe that the base model tends to allocate more attention to KG or linguistically simple paths as compared to Textual paths (i.e., the path comes form Text), Hybrid paths (i.e., the path comes from both Text and KG), or linguistically complex ones, as shown in Figure 3a and Figure 3b. We consider that this would be because paths including Textual relations (i.e., Textual and Hybrid paths) or complex paths are comparatively noisier than KG or simple paths, but which does not necessarily mean the former is not useful. For instance, in Figure 1, the complex Hybrid path p_2 is useful for predicting (**Colesevelam_HCl**, *may_treat*, **Type_2_Diabetes**), because p_2 implies a plausible line of reasoning " ***Colesevelam_HCl*** $\xrightarrow{\textit{alternative\_to}}$ *Colestipol* $\xrightarrow{\textit{may\_treat}}$ *hyperglycemia* $\xrightarrow{\textit{strong\_link\_to}}$ ***Type_2_Diabetes***". However, due to the attention bias mentioned above, the base model allocates low attention ($a'_2 \approx 8.0 \times 10^{-36}$) on the informative path, and thus fails to learn from such complex but useful evidences.

To reduce the negative effect of the attention biases and make full use of the UG path, we propose the following two training (or debiasing) strategies: Path Type Adaptive Pretraining (§4.2) and Complexity Ranking Guided Attention (§4.3).

4.2 Path Type Adaptive Pretraining

As shown in Figure 3a, the base model tends to bias toward KG paths. This indicates that the base model mainly relies on KG paths so that it is incapable of capturing informative features from Textual and Hybrid paths. This bias will decrease the flexibility and adaptability of the base model to different types of paths.

To address this issue, we propose a debiasing strategy called Path Type Adaptive Pretraining. In this strategy, we pretrain the base model sequentially using Textual, Hybrid, and KG Paths as path evidences, and then finetune it with all types of paths as illustrated in Figure 4. We hypothesize that this strategy can prevent the reliance on a single type of UG path and improve the capacity of extracting features from the entire UG paths, and thereby increase the performance.

4.3 Complexity Ranking Guided Attention

Similar to the bias towards KG paths, the base model also focuses on linguistically simple paths, as shown in Figure 3b, even though complex ones are informative (e.g. p_2 in Figure 1). We hypothesize that restricting the attention span to the complex (simple) paths can force the model to pay attention to the complex (simple) paths, thereby effectively utilize them. Under this hypothesis, we propose a Complexity Ranking Guided Attention mechanism, as illustrated in Figure 2.

Specifically, given a bag of paths $P_r = \{p_1, ..., p_m\}$, we rank them according to their complexity scores (κ), which are calculated via $\kappa = \lambda_1\tau_1 + \lambda_2\tau_2 + ...$, where τ denotes the feature for capturing linguistic complexity (e.g., path length) and λ is a corresponding weight, which is a hyperparameter. Sentence length (i.e., the number of tokens in a sentence) and lexical richness (i.e., the number of token types) are commonly used features for evaluating sentence complexity (Brunato et al., 2018). Therefore, this work adopts them to calculate the complexity for a given path.

Then, we group top j most and least complex paths into a set of complex and simple paths respectively, where j is a hyperparameter[1]. The set level representation is calculated via the Equation 8.

$$\mathbf{p}_{\text{complex} \textit{ or } \text{simple}} = \sum_{i \in \textit{top\_j} \textit{ or } i \in \textit{last\_j}} a'_i \mathbf{p}_i \tag{8}$$

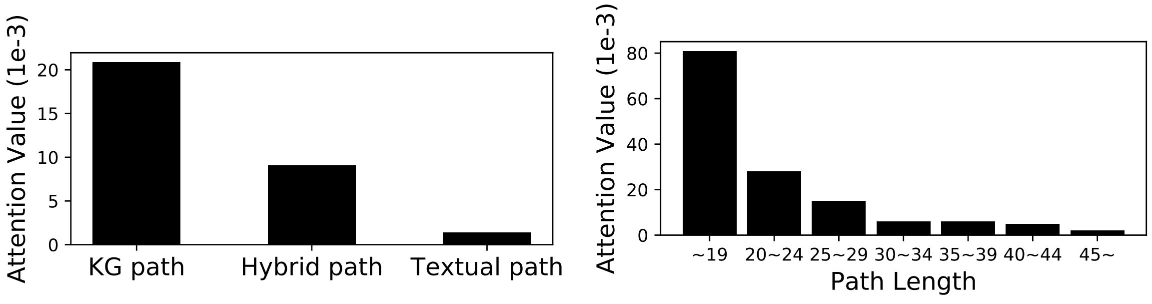
$$a'_i = \frac{\exp(\langle \mathbf{r}_{ht}, \mathbf{x'}_i \rangle)}{\sum_{k \in \textit{top\_j} \textit{ or } k \in \textit{last\_j}} \exp(\langle \mathbf{r}_{ht}, \mathbf{x'}_k \rangle)},$$
$$\mathbf{x'}_i = \tanh(\mathbf{W}\mathbf{p}_i + \mathbf{b})$$

Finally, we concatenate the resulting representation $\mathbf{s}_{\text{final}}$, $\mathbf{p}_{\text{final}}$, $\mathbf{p}_{\text{simple}}$ and $\mathbf{p}_{\text{complex}}$ as the input to the relation classification layer. The conditional probability $P(e_1, r, e_2|S_r, P_r, \theta_S, \theta_P)$ is formulated via Equation 9, where $\mathbf{o} = \mathbf{M}[\mathbf{s}_{\text{all}}; \mathbf{p}_{\text{all}}; \mathbf{p}_{\text{complex}}; \mathbf{p}_{\text{simple}}] + \mathbf{d}$.

$$P(e_1, r, e_2|S_r, P_r, \theta_S, \theta_P) = \frac{\exp([\mathbf{o}]_r)}{\sum_{c=1}^{n_r} \exp([\mathbf{o}]_c)} \tag{9}$$

---

[1] In our experiments, we set j as 30 for NYT10 dataset and 50 for Biomedical dataset.



(a) Path type and attention

(b) Path complexity and attention

Figure 3: (**Left**) Average attention weights across different path types. (**Right**) Average attention weights across different length range of paths. Here, we use path length as an indicator of path complexity, which denotes the number of words in a path.

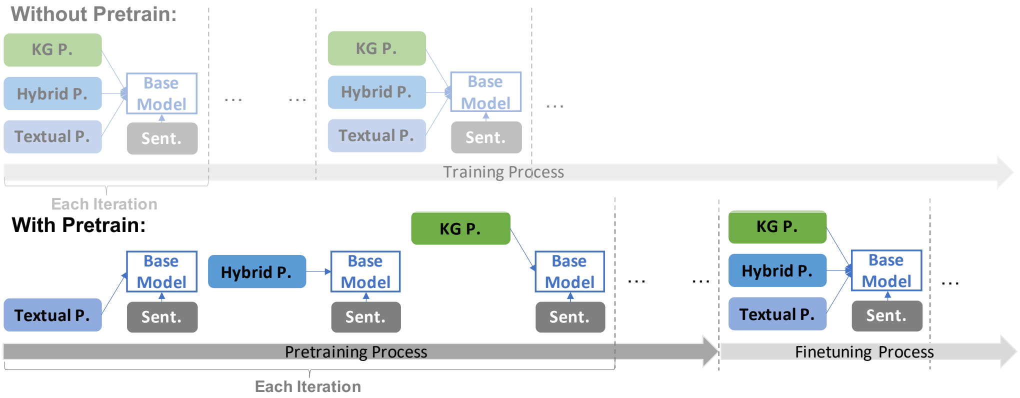


Figure 4: Path Type Adaptive Pretraining strategy, where "Textual/Hybrid/KG P." represent Textual, Hybrid, and KG paths respectively, and "Sent." represents the sentence evidences. In this strategy (**Pretrain**), instead of using all types of paths to train the base model in all iterations, we sequentially train the model with Textual, Hybrid and KG paths, and then finetune it with all types of paths.

5 Experiments

5.1 Data

We evaluate our proposed framework on a biomedical dataset and NYT10 dataset (Riedel et al., 2010). The statistics of both datasets is summarized in Table 1. We will detail both datasets as follows.

Biomedical Dataset. This datatset is created by linking biomedical KG with biomedical Text. We choose UMLS[2] and Medline corpus as the biomedical KG and Text respectively. UMLS is a frequently used biomedical knowledge base, while Medline corpus is a large collection of biomedical abstracts, both are developed and maintained by the U.S. National Library of Medicine[3]. For identifying UMLS entity mentions in the Medline corpus, we use a state-of-the-art UMLS Named Entity Recognizer (NER), ScispaCy (Neumann et al., 2019). The NER identifies UMLS concepts and annotates them by their corresponding UMLS Concept Unique Identifier (CUI) and entity types.

From the UMLS KG and the entity linked Medline corpus, we extract fact triplets (i.e., (e_1, r, e_2)) and corresponding sentence evidences containing (e_1, e_2) under the restriction that: (1) each entity pair should be connected by a RO (RO stands for "has Relationship Other than synonymous, narrower, or broader") relationship; (2) each entity should belong to the following entity types: Protein, Gene, Disease or Syndrome, Enzyme, Chemical, Sign or Symptom and Pharmacologic Substance. Then we divide the collected triplets and sentence evidences into training and testing set according to the year when the source abstract of sentence evidence was published. The former is aligned to the years until 2008 and the latter to the years 2009 $\sim$ 2018, ensuring the testing set only contains the

---

[2] `https://www.nlm.nih.gov/research/umls/`

[3] `https://www.nlm.nih.gov/`

	#R	**#EP**	**#Related EP**	**#Sentence**	**#UG Path**
Biomedical	40	100,549 / 21,081	10,936 / 1,804	165,692 / 28,912	12,854,696 / 2,346,007
NYT10	53	281,270 / 96,678	18,252 / 1,950	522,611 / 172,448	8,967,153 / 2,984,611

Table 1: Statistics of datasets in this work, where **R** and **EP** stand for the target Relation and Entity Pair, $\#_1/\#_2$ represent the number of training and testing data respectively.

unobserved triplets.

To simulate the noise in the real world, besides the "related" triplets, we also extract the "unrelated" triplets and sentence evidences based on a closed world assumption: pairs of entities not listed in a KG are regarded to have NA relation and sentences containing them are considered to be the NA sentence evidences. We divide the NA triplets and NA sentence evidences in the same way mentioned above. We use a subset of UMLS (see Appendix §A.3) and the Medline abstracts published until 2008 as the KG and Text respectively to create the UG for path retrieval. In addition, we use the same subset of UMLS triplets mentioned above to train the KG Encoder introduced in §3.

NYT10. The dataset is created by aligning Freebase relational facts with the New York Times Corpus. Sentence evidences from the year 2005 $\sim$ 2006 are used for training and the evidences from 2007 are used for testing. NYT10 dataset has been widely used by (Lin et al., 2016; Ji et al., 2017; Du et al., 2018; Jat et al., 2018; Du et al., 2018; Han et al., 2018a,b; Vashishth et al., 2018; Ye and Ling, 2019; Alt et al., 2019). We use Freebase[4] and ClueWeb12 with Freebase entity mention annotations (Gabrilovich et al., 2013) as the KG and Text to create the UG for path searching. In addition, following (Han et al., 2018a), we use FB60K for training the KG Encoder.

UG path search. Given an entity pair (e_1, e_2), the UG path set P_r is obtained by performing random walks over the UG from e_1 till e_2 with maximum step[5].

5.2 Settings

We follow (Lin et al., 2016) and conduct the held-out evaluation, in which the model for DS-RE is evaluated by comparing the fact triplets identified from evidences (i.e., the bag of sentence evidences S_r and the bag of UG path evidences P_r) with those in KG. Following the evaluation of previous works, we draw Precision-Recall curves and report the Area Under Curve (AUC) and Precision@N (P@N) metrics, which gives the percentage of correct triplets among top N ranked candidates. The parameter settings of our experiments are detailed in Appendix §A.2.

To demonstrate the effectiveness of our framework, we choose the model proposed by Dai et al. (2019) as the baseline model, because this is the closest model in terms of incorporating multiple paths for DS-RE. Henceforth, "Sent+KG" is the baseline model, which uses both sentence evidences and KG paths. "Sent+UG" represents the base model in §3 which takes UG paths instead of KG paths as path evidences. "Sent+UG+Pretrain" and "Sent+UG+Ranking" denote the base model trained with Path Type Adaptive Pretraining strategy and the base model with Complexity Ranking Guided Attention mechanism, respectively. "Sent+UG+Ranking+Pretrain" means the base model trained with both strategies.

5.3 Results and Discussion

Precision-Recall Curves. The Precision-Recall (PR) curves of each model on the biomedical and NYT10 datasets are shown in Figure 5 and Figure 6, respectively. The results show that: (1) "Sent+UG" does not have obvious advantages than "Sent+KG", illustrating that due to the biases discussed in §4.1, simply applying UG paths on the base model has limited effect on improving the performance of DS-RE. (2) "Sent+UG+Pretrain" and "Sent+UG+Ranking" achieve better overall performance than "Sent+KG" on both datasets, especially when the recall is greater than 0.3, demonstrating that UG has the potential to enhance the performance and the two proposed debiasing strategies are effective for exploiting the potential of UG for DS-RE. (3) "Sent+UG+Ranking+Pretrain" achieves the highest precision over the (almost) entire recall range on both datasets, proving that the two proposed strategies have a mutual complementary relationship on exploiting UG for DS-RE. This is understandable because the two proposed strategies deal with different types of biases, in addition, "Pretrain" helps the base model adapt to UG paths by effectively tuning its weights, while "Ranking" enhances the base model by adjusting its attention mechanism. (4) The consistent improvement on

---

[4] From the entire Freebase, we only collect the triplets with the relations that are mentioned in NYT10 dataset for UG creation, ensuring not to overlap with testing set.

[5] We manually set the maximum step as 3.

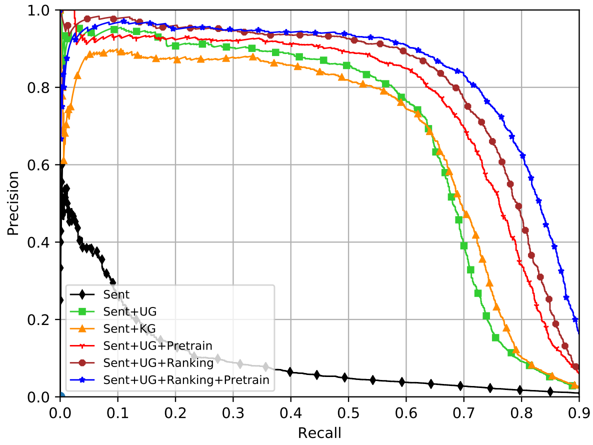


Figure 5: PR curves on Biomedical dataset.

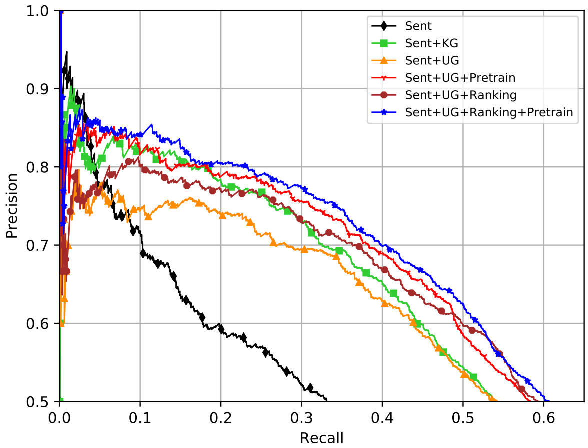


Figure 6: PR curves on NYT10 dataset.

	Biomedical dataset					**NYT10 dataset**						
Model	AUC	P@0.5k	P@1k	P@1.5k	P@2k	AUC	P@0.1k	P@0.2k	P@0.3k	P@0.5k	P@1k	P@2k
Sent	9.6	30.0	20.8	18.7	16.3	36.6	81.0	73.5	68.3	62.0	53.8	40.2
Sent+KG	62.6	91.4	86.1	74.2	58.5	50.2	80.0	82.0	81.3	77.2	67.9	50.3
Sent+UG	61.0	87.6	83.4	73.8	58.5	48.4	74.0	76.0	74.7	74.0	66.7	50.3
Sent+UG +Pretrain	70.1	95.4	89.7	76.3	60.4	52.7	83.0	82.0	80.3	78.6	70.4	52.6
Sent+UG +Ranking	74.2	95.2	92.2	81.1	62.2	52.1	77.0	80.0	79.3	77.4	70.3	54.4
Sent+UG +Ranking +Pretrain	**77.5**	**95.4**	**93.1**	**83.9**	**64.4**	**55.0**	**86.0**	**84.0**	**83.3**	**80.4**	**71.9**	**54.5**

Table 2: P@N and AUC on Biomedical and NYT10 dataset (k=1000).

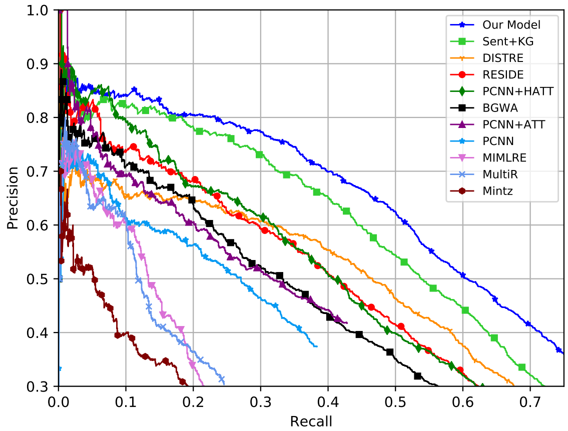


Figure 7: PR curves of previous state-of-the-art methods and our proposed model on NYT10 dataset.

two datasets from different domains further proves the validity of our proposed methods.

AUC and P@N Evaluation. Table 2 further presents the results in terms of AUC and P@N. From them, we have similar observation to the PR curves. We also observe that the effectiveness of UG paths is more pronounced on Biomedical dataset than on NYT10 dataset. We speculate that compared to the generic NYT10 dataset, further Background Knowledge (BK) is needed to identify relations from Biomedical dataset, and UG paths could be utilized as the BK to facilitate the scientific DS-RE.

Comparison with State-of-the-art Baselines on NYT10. To demonstrate the effectiveness of our proposed model, we also compare it against the following baselines on NYT10 dataset: Mintz (Mintz et al., 2009), MultiR (Hoffmann et al., 2011), MIMLRE (Surdeanu et al., 2012), PCNN (Zeng et al., 2015), PCNN+ATT (Lin et al., 2016), BGWA (Jat et al., 2018), PCNN+HATT (Han et al., 2018b), RESIDE (Vashishth et al., 2018), DISTRE (Alt et al., 2019) and Sent+KG (Dai et al., 2019). The results shown in Figure 7 and Table 3 indicate that: (1) our selected base model, "Sent+KG", is a strong baseline because it signif-

System	AUC	P@0.1k	P@0.2k	P@0.3k	P@0.5k	P@1k	P@2k
Mintz†	10.7	52.3	50.2	45.0	39.7	33.6	23.4
PCNN+ATT†	34.1	73.0	68.0	67.3	63.6	53.3	40.0
RESIDE†	41.5	81.8	75.4	74.3	69.7	59.3	45.0
PCNN+HATT‡	42.0	81.0	79.5	75.7	68.0	58.6	42.1
DISTRE†	42.2	68.0	67.0	65.3	65.0	60.2	47.9
Sent+KG	50.2	80.0	82.0	81.3	77.2	67.9	50.3
Our Model	**55.0**	**86.0**	**84.0**	**83.3**	**80.4**	**71.9**	**54.5**

Table 3: P@N and AUC from previous state-of-the-art DS-RE models and our proposed model on NYT10 dataset, where †represents that these results are quoted from (Alt et al., 2019) and ‡indicates the results using the pretrained model from (Han et al., 2018b).

icantly outperforms other state-of-the-art models; and (2) our model can effectively take advantage of the rich UG paths for DS-RE because it beats the strong baseline and achieves a new state-of-the-art result on the commonly used DS-RE dataset.

Base	Prop.	**Biomedical Triplet**
✗	✓	(Beta-2...Gene , gene_associated_with_disease, Asthma)
		Multi-hop Path
Low	**High**	*hop*$_1$: *"The human Beta-2...Gene is responsible for the binding of endogenous Catecholamine and their ..."* *hop*$_2$: *" Catecholamine chemical structure of Epinephrine "* *hop*$_3$: *" Epinephrine may treat Asthma ".*
Base	Prop.	**NYT10 Triplet**
✗	✓	(San_Francisco , /location/contains, Noe_Valley)
		Multi-hop Path
Low	**High**	*hop*$_1$: *" San_Francisco /location/contains Fort_Point "* *hop*$_2$: *"Surf spots and surfing regions include Northern CA, the Bay_Area , San Francisco, Ocean Beach and Fort Point "* *hop*$_3$: *" Bay_Area /location/contains Noe_Valley "*

Table 4: Some examples of attention distribution over paths from "Sent+UG" (Base) and "Sent+UG+Ranking+Pretrain" (Prop.), where ✓(or ✗) represents the correct (or incorrect) prediction of the target relation.

Case Study. Table 4 shows the UG path examples that are scored with highest ("High") or lowest (or lower than 1.0×10^{-3}) ("Low") attention by the base model and our proposed framework. The paths in the table generally mean "*Beta-2... Gene* $\xrightarrow{is\_responsible\_for}$ *Catecholamine* $\xrightarrow{is\_the\_chemical\_class\_of}$ *Epinephrine* $\xrightarrow{may\_treat}$ *Asthma*" and "*San_Francisco* $\xrightarrow{contains}$ *Fort_Point* $\xrightarrow{equal\_status}$ *Bay_Area* $\xrightarrow{contains}$ *Noe_Valley*", and thus can be seen as the useful path evidences for identifying *gene_associated_with_disease* and */location/contains* relation respectively. These examples indicate that our proposed training strategies could help the base model attend such informative UG paths so that it can correctly identify the target relation.

6 Conclusion and Future Work

We have introduced UG paths as extra evidences for the task of DS-RE from text. In order to fully take advantage of the rich UG paths, we have proposed two training (or debiasing) strategies: Path Type Adaptive Pretraining and Complexity Ranking Guided Attention mechanism. We have conducted experiments on both biomedical and NYT10 datasets. The results show that the two proposed methods are effective for exploiting the potential of UG paths for improving the performance of DS-RE.

In the future, we plan to carry out the following steps: (1) we further investigate how the proposed training methods influence the performance via manual analysis so as to better the efficiency; and (2) instead of random walk, we may collect UG paths by adopting more sophisticated mechanisms such as training a path searching agent via reinforcement learning to prevent redundant and noisy paths.

Acknowledgement

This work was supported by JST CREST Grant Number JPMJCR1513, Japan and KAKENHI Grant Number 16H06614. We would like to thank Benjamin Heinzerling, Shota Sasaki and other collaborators for their useful comments and suggestions.

References

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. *arXiv preprint arXiv:1906.08646*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Dominique Brunato, Lorenzo De Mattei, Felice Dell'Orletta, Benedetta Iavarone, and Giulia Venturi. 2018. Is this sentence difficult? do you agree? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2690–2699.

Qin Dai, Naoya Inoue, Paul Reisert, Takahashi Ryo, and Kentaro Inui. 2019. Incorporating chains of reasoning over knowledge graph for distantly supervised biomedical knowledge acquisition. In *Proceedings of the 33nd Pacific Asia Conference on Language, Information and Computation (PACLIC33)*, pages 19–28, Hakodate, Japan. Waseda Institute for the Study of Language and Information.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017a. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 132–141.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017b. Question answering on knowledge bases and text using universal schema and memory networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–365.

Jinhua Du, Jingguang Han, Andy Way, and Dadong Wan. 2018. Multi-level structured self-attentions for distantly supervised relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2216–2225.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0).

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018a. Neural knowledge acquisition via mutual attention between knowledge graph and text. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018b. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Sharmistha Jat, Siddhesh Khandelwal, and Partha Talukdar. 2018. Improving distantly supervised relation extraction using word and entity based attention. *arXiv preprint arXiv:1804.06987*.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2017. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. Arnor: attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Neural relation extraction with multi-lingual attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–43.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 777–782.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*

(Volume 1: Long Papers), pages 156–166, Beijing, China. Association for Computational Linguistics.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1171–1182.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Huan Sun et al. 2019. Leveraging 2-hop distant supervision from table entity pairs for relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 410–420.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 455–465. Association for Computational Linguistics.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1499–1509.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. Reside: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2810–2819.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344.

Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Incorporating relation paths in neural relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1768–1777.

A Appendix

A.1 CNN-Max

Convolutional Neural Network with Max pooling layer (CNN-Max) is adopted to derive the sentence representation $\mathbf{s}$ and path representation $\mathbf{p}$. Specifically, vector representation $\mathbf{v}_t$ for each word w_t is calculated via Equation 10, where $\mathbf{W}^{w}_{emb}$ is a word embedding projection matrix (Mikolov et al., 2013), $\mathbf{W}^{wp}_{emb}$ is a word position embedding projection matrix (Zeng et al., 2014), $\mathbf{x}_t^{w}$ is a one-hot word representation and $\mathbf{x}_t^{wp}$ is a one-hot word position representation, which indicates the relative distance between the current word and the target entity pair.

$$\begin{aligned} \mathbf{v}_t &= [\mathbf{v}_t^{w}; \mathbf{v}_t^{wp1}; \mathbf{v}_t^{wp2}], \\ \mathbf{v}_t^{w} &= \mathbf{W}^{w}_{emb}\mathbf{x}_t^{w}, \\ \mathbf{v}_t^{wp1} &= \mathbf{W}^{wp}_{emb}\mathbf{x}_t^{wp1}, \\ \mathbf{v}_t^{wp2} &= \mathbf{W}^{wp}_{emb}\mathbf{x}_t^{wp2} \end{aligned} \tag{10}$$

The sentence representation $\mathbf{s}$ and path representation $\mathbf{p}$ are formulated via the Equation 11, where $\mathbf{W}^{sent}$ (or $\mathbf{W}^{path}$) is the convolution kernal, $\mathbf{b}^{sent}$ (or $\mathbf{b}^{path}$) is the corresponding bias vector, $\mathbf{v}_t^{sent}$ (or $\mathbf{v}_t^{path}$) is the vector for each word w_t in a sentence (or path), $[vec]_i$ is the i-th value of vec, ν is the dimensionality of $\mathbf{s}$ and $\mathbf{p}$, and k is the convolutional window size.

$$\begin{aligned} [\mathbf{s}]_i &= \max_t\{[\mathbf{h}_t^{sent}]_i\},\ \forall i = 1, ..., \nu \\ [\mathbf{p}]_i &= \max_t\{[\mathbf{h}_t^{path}]_i\},\ \forall i = 1, ..., \nu \end{aligned} \tag{11}$$

$$\begin{aligned} \mathbf{h}_t^{sent} &= \tanh(\mathbf{W}^{sent}\mathbf{z}_t^{sent} + \mathbf{b}^{sent}), \\ \mathbf{h}_t^{path} &= \tanh(\mathbf{W}^{path}\mathbf{z}_t^{path} + \mathbf{b}^{path}), \\ \mathbf{z}_t^{sent} &= [\mathbf{v}^{sent}_{t-(k-1)/2}; ...; \mathbf{v}^{sent}_{t+(k-1)/2}], \\ \mathbf{z}_t^{path} &= [\mathbf{v}^{path}_{t-(k-1)/2}; ...; \mathbf{v}^{path}_{t+(k-1)/2}] \end{aligned}$$

A.2 Parameter Settings

All of the hyperparameters used in our experiments are listed in Table 5. Most of them follow the hyperparameter setting in (Dai et al., 2019) and (Han et al., 2018a). We use a Word2Vec model[6] to train the word embeddings on the UMLS entity linked corpus for the biomedical dataset, and adopt the word embeddings released by (Lin et al., 2016) for NYT10 dataset. We apply Stochastic Gradient Descent (SGD) to optimize the proposed DS-RE model.

Hyperparameter	**Biomedical**	**NYT10**
word embedding dimension	50	50
KG embedding dimension	50	50
position embedding dimension	5	5
CNN window size	3	3
CNN filter number	100	230
dropout rate	0.5	0.5
learning rate (for sentences and paths)	0.02	0.05
learning rate (for KG)	0.05	0.001
batch size	50	160

Table 5: Hyperparameters used in our experiments.

Selected Entity Types
Antibiotic, Biologically Active Substance, Bacterium, Organ, Cell Component, Cell Function, Cell, Clinical Drug, Ion, Eukaryote, Food, Genetic Function Hazardous or Poisonous Substance, Hormone, Immunologic Factor, Inorganic Chemical, Organic Chemical, Pathologic Function, Receptor, Steroid, Virus and Vitamin.

Table 6: List of selected UMLS entity types.

A.3 Subset of UMLS

Besides the 7 entity types mentioned above, we also use other 22 entity types, as listed in Table 6, to collect the UMLS triplets that are connected by RO relationship, ensuring all testing triplets are removed. The main reasons to manually restrict the entity type is because (1) we observe that most of the Medline abstracts discuss the relationship among these entity types; (2) these concrete entities could prevent semantic drift while searching UG paths.

---

[6] Gensim word2vec implementation: `https://radimrehurek.com/gensim/models/word2vec.html`

Adaptation of Back-translation to Automatic Post-Editing for Synthetic Data Generation

WonKee Lee[†*] **Baikjin Jung**[†*] **Jaehun Shin**[†] **Jong-Hyeok Lee**[†‡]
[†]Departmet of Computer Science and Engineering,
[‡]Graduate School of Artificial Intelligence,
Pohang University of Science and Technology (POSTECH), Republic of Korea
`{wklee, bjjung, jaehun.shin, jhlee}@postech.ac.kr`

Abstract

Automatic Post-Editing (APE) aims to correct errors in the output of a given machine translation (MT) system. Although data-driven approaches have become prevalent also in the APE task as in many other NLP tasks, there has been a lack of qualified training data due to the high cost of manual construction. eSCAPE, a synthetic APE corpus, has been widely used to alleviate the data scarcity, but it might not address genuine APE corpora's characteristic that the post-edited sentence should be a minimally edited revision of the given MT output. Therefore, we propose two new methods of synthesizing additional MT outputs by adapting back-translation to the APE task, obtaining robust enlargements of the existing synthetic APE training dataset[1]. Experimental results on the WMT English-German APE benchmarks demonstrate that our enlarged datasets are effective in improving APE performance.

1 Introduction

src	Manipulates the shape of an item .
mt	**Bearbeitet** die Form eines Elements **an** .
pe	**Verändert** die Form eines Elements .

Table 1: An example of APE triplets from the WMT dataset (Bojar et al., 2017). **Boldface** words are either incorrect words in *mt* or post-edited words in *pe*.

Automatic Post-Editing (APE) seeks to automatically correct errors included in the output of a black-box machine translation (MT) system to improve the final translation quality, thereby reducing the effort required for manual post-editing (Allen and Hogan, 2000; Chatterjee et al., 2015; Bojar et al., 2016; Chatterjee et al., 2018). In general, APE can be considered as a task of sequence-to-sequence supervised learning, which requires a considerable amount of human-annotated data. However, constructing an APE corpus—a set of triplets (Table 1), each of which includes a source text (*src*), a machine-translated text (*mt*), and a manually post-edited text (*pe*)—is labor-intensive work because post-editors should create *pe* in principle by **minimally editing** *mt* while preserving the meaning of *src*. In fact, the sizes of currently available 'genuine' APE corpora provided by WMT (Bojar et al., 2016, 2017; Chatterjee et al., 2018, 2019, 2020) are too small to train deep APE models effectively.

To overcome the lack of genuine APE corpora, several previous studies have proposed methods to construct synthetic training datasets (Junczys-Dowmunt and Grundkiewicz, 2016; Negri et al., 2018; Lee et al., 2020), and they appear to be partially helpful in mitigating the data scarcity problem. One such study is eSCAPE (Negri et al., 2018), which has been shown to be effective in training deep models and adopted in a number of APE works (do Carmo et al., 2020). Utilizing parallel corpora, which comprise pairs of a source sentence (*src*) and a reference sentence (*ref*), eSCAPE was constructed as a set of synthetic APE triplets in the form of (*src*, *mt*, *ref*) where *mt* is a machine translation of *src*, and *ref* serves as an alternative to *pe* of a genuine APE triplet.

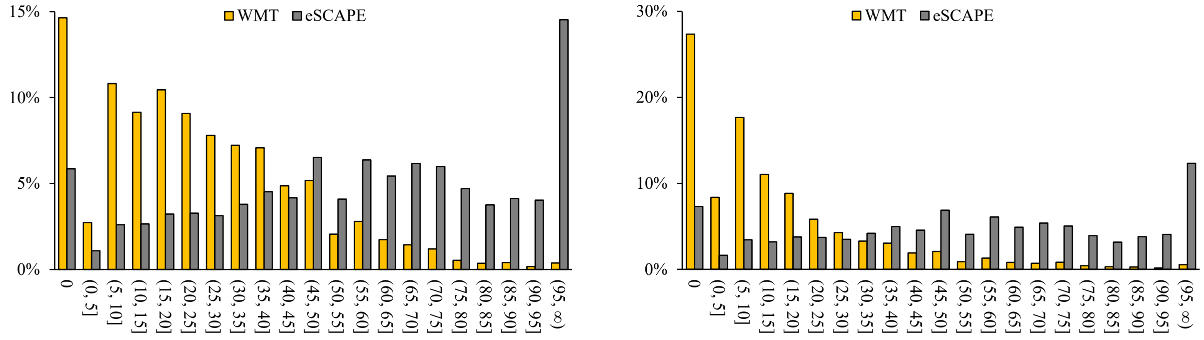
Despite the effectiveness of eSCAPE, we argue that it may have two major drawbacks: (1) eSCAPE's method relies heavily on parallel resources, so its scalability is restricted to the quantity of available parallel resources and can be even more limited in low-resource scenarios; (2) the relation between *mt* and *ref* may not thoroughly reflect the actual relation between *mt* and *pe* because *ref* is not guaranteed to be a minimally edited revision of *mt*, potentially leading to the discrepancy between

---

[*] Equal contribution to this work.

[1] Our synthetic APE data is available at `https://github.com/wonkeelee/APE-backtranslation.git`

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3685–3691
April 19 - 23, 2021.



(a) PBSMT dataset

(b) NMT dataset

Figure 1: (a) and (b) are categorical distributions that indicate the proportion y [%] of samples belonging to a specific TER interval x for each of the two APE data types classified according to the type of the target MT system. Here, TER is a metric that presents the distance (error ratio) between *mt* and its target (*pe* or *ref*).

the distribution of translation errors in genuine data and that of translation errors in the synthetic data (Figure 1).

In this paper, we propose two automatic methods that are inspired by back-translation from the MT task (Sennrich et al., 2016), employing the APE process in the forward direction and the backward direction with applying the eSCAPE resource to them to create additional synthetic *mt*. Our approach not only extends the existing resource, but also aims to better simulate the characteristics of real APE data by making our synthetic *mt* better approximate the error distribution of the WMT APE benchmark dataset.

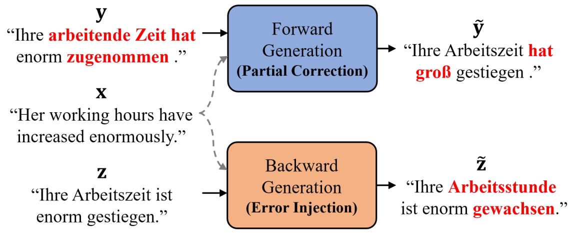


Figure 2: An illustration of our two synthetic data generation methods. $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ are examples of *src*, *mt*, and *ref*, respectively. $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{z}}$ are expected outputs that the proposed methods will produce. **Boldface** words indicate incorrect words.

2 Background and Related Work

Back-translation. Back-translation is a method to create synthetic source texts from clean target texts by using an MT system that is trained in the target–to–source direction. Back-translation has allowed many MT studies to use monolingual data to generate additional parallel data so that they alleviate data scarcity; moreover, it has also been successfully adopted by other NLP tasks such as summarization (Parida and Motlicek, 2019; Jernite, 2019) and grammatical error correction (Xie et al., 2018).

Learning Objective of APE. Given that APE aims to revise *mt* to *pe* while preserving the meaning of *src*, each one of the two sources (*src*, *mt*) plays a distinct and critical role: *src* is treated as an auxiliary source, not only offering intact semantic and contextual information but also being helpful in identifying mistranslation; *mt*, meanwhile, serves as the primary source, which needs to be corrected. In this perspective, the multi-source approach: (*src*, *mt*) $\mapsto$ *ref*, is commonly used to take both *src* and *mt* into account (Chatterjee et al., 2018, 2019). Specifically, considering *src*, *mt*, and *pe* as $\mathbf{x} = \{x_i\}_{i=1}^{T_x}$, $\mathbf{y} = \{y_j\}_{j=1}^{T_y}$, and $\mathbf{z} = \{z_k\}_{k=1}^{T_z}$ with the sequence lengths T_x, T_y, and T_z, respectively, the APE model learns to predict *pe* with the following conditional probability:

$$p(\mathbf{z}) = \prod_{k=1}^{T_z} p(z_k|\mathbf{x}, \mathbf{y}, z_{<k}; \theta), \tag{1}$$

where θ is a set of model parameters.

3 Method

Beyond the eSCAPE corpus, to yield a more convincing error distribution as well as to supply APE models with more APE resources made out of limited parallel resources, we propose synthetic-data generation methods that can be seen as adaptations of back-translation to the APE task in terms of creating synthetic *mt*, which is one of the two sources of APE.

We produce new synthetic *mt* so that *ref* can better act as its minimally post-edited text, whereas this *ref* may not do so for the original *mt*. Specifically, we suggest two strategies, both of which apply the APE process: '**forward generation**' and '**backward generation**'; each one of them performs APE in the forward direction and the backward direction, respectively. As described in Figure 2, the former partially corrects *mt* to reduce the distance between *mt* and *ref*, while the latter injects the right quantity of translation errors into *ref*.

3.1 Forward Generation

The '**F**orward **G**eneration' (FG) method lets an APE model take *src* and *mt* as input to produce mt_{FG} as output by partially correcting *mt* through the forward path of APE; the training objective of an FG model is identical to that of a normal APE model (Eq. 1). The output mt_{FG} then forms a new synthetic triplet (*src*, mt_{FG}, *ref*) together with *src* and *ref*. We use such triplets to construct a new set of synthetic triplets $\mathrm{eSCAPE}_{\mathrm{FG}}$.

Considering that *mt* generally requires a lot of excessive correction to match *ref*, this approach's motivation is that mt_{FG}, in itself a product of the APE process, will generally be closer to *ref* than the original *mt*. However, if the distance between mt_{FG} and *ref* is excessively small, indicating that the two texts are almost identical, APE models trained on $\mathrm{eSCAPE}_{\mathrm{FG}}$ may not learn error-correction patterns sufficiently. Thus, unlike the standard training procedure, we force the FG model's training process to stop earlier before convergence, making the remaining errors in its output mt_{FG} ample. We therefore use simple arrangements (§4) to find one optimal value for this stop point.

3.2 Backward Generation

Borrowing the idea of back-translation, the '**B**ackward **G**eneration' (BG) method reverses the APE process during training by moving *mt* to the position of *ref* and vice versa; hence, a BG model is trained on (*src*,*ref*) $\mapsto$ *mt* to maximize the following conditional probability:

$$p(\mathbf{y}) = \prod_{j=1}^{T_y} p(y_j|\mathbf{x}, \mathbf{z}, y_{<j}; \theta). \tag{2}$$

In other words, the model learns to generate mt_{BG} to contain translation errors that occur in *mt* conditioned on a pair of *src* and *ref*. The output mt_{BG} then composes a new synthetic triplet (*src*, mt_{BG}, *ref*) together with *src* and *ref*. We use such triplets to construct another set of synthetic triplets $\mathrm{eSCAPE}_{\mathrm{BG}}$.

Dataset		# Triplets
`WMT-PBSMT`	Train	23,000
	Dev	1,000
	Test2016	2,000
	Test2017	2,000
	Test2018	2,000
`WMT-NMT`	Train	13,442
	Dev	1,000
	Test2018	1,023
`eSCAPE-PBSMT`		7,258,533
`eSCAPE-NMT`		7,258,533

Table 2: Statistics of the WMT and eSCAPE datasets on the PBSMT and NMT subtasks.

In contrast to FG, the concept of BG is to corrupt a clean text (*ref*) by learning until the distance between the BG output and *ref* becomes similar to the edit distance of real APE data. However, if we let the BG model fully converge, the output mt_{BG} may not have a big difference from the original *mt*; on the other hand, if the model has been barely trained, mt_{BG} would be almost the same as *ref*. In both instances, APE models trained on $\mathrm{eSCAPE}_{\mathrm{BG}}$ may not learn error-correction patterns sufficiently. We use the same arrangements (§4) as in FG to find an optimal value for the BG model's stop point.

4 Experiments

Metric. Following the evaluation setting used in the WMT APE shared task, we adopt TER (Snover et al., 2006) as the primary metric to measure the distance between the model's prediction and the reference text; and BLEU (Koehn et al., 2007) as the secondary metric to measure the degree of n-gram match. In addition, all evaluations in our experiments are case-sensitive.

Dataset. We use two kinds of APE datasets: human-made APE datasets, which are provided by WMT, and eSCAPE. Both are English–German (EN–DE) APE corpora; they are further categorized according to their subtask depending on whether the target MT system is a phrase-based statistical MT (PBSMT) system or a neural MT (NMT) system. The WMT datasets are in the IT domain, whereas eSCAPE was made out of domain-general parallel corpora. Detailed data statistics are presented in Table 2. We tokenized all words in our datasets into sub-word units by using Senten-

MODELS	PBSMT								NMT	
	`Test16`		`Test17`		`Test18`		Avg.		`Test18`	
	TER(↓)	BLEU(↑)	TER(↓)	BLEU(↑)	TER(↓)	BLEU(↑)	TER(↓)	BLEU(↑)	TER(↓)	BLEU(↑)
WMT Baseline (No edit)	24.76	62.11	24.48	62.49	24.24	62.99	24.49	62.53	16.84	74.73
eSCAPE	16.97	73.94	17.35	72.93	17.74	72.34	17.35	73.07	16.39	75.70
eSCAPE$_{\text{FG}}$	17.06	73.96	17.40	72.81	18.00	72.19	17.48	72.98	16.30	75.77
eSCAPE$_{\text{BG}}$	17.25	73.58	17.85	72.30	17.93	72.12	17.66	72.66	16.50	75.40
eSCAPE + eSCAPE$_{\text{FG}}$	16.79	74.25	17.05*	73.30*	17.32*	72.95*	17.05*	73.50*	16.09*	76.11*
eSCAPE + eSCAPE$_{\text{BG}}$	16.73	74.32*	**16.96***	73.41*	**17.26***	**73.14***	16.98*	73.62*	**15.95***	**76.14***
eSCAPE + eSCAPE$_{\text{FG}}$ + eSCAPE$_{\text{BG}}$	**16.57***	**74.52***	16.99*	**73.50***	17.29*	73.11*	**16.95***	**73.71***	16.15*	76.00*
BERT-APE (Correia and Martins, 2019)	16.91	74.29	17.26	73.42	17.71	72.74	17.29	73.48	–	–
BERT-APE (Correia and Martins, 2019) (Ensemble)	16.49	74.98	16.83	73.94	17.15	73.60	16.82	74.17	–	–
BERT-APE (Lopes et al., 2019)	–	–	–	–	–	–	–	–	16.06	75.96

Table 3: Evaluation results of our APE models using different configurations on training datasets. '*' represents that our model's improvement is significant enough compared to the eSCAPE baseline in the second row with $p < 0.05$. The best result among our models in each column is in **bold** type. The three models at the bottom are current state-of-the-art models.

cePiece (Kudo and Richardson, 2018).

Model Configuration. We implemented a Transformer-based APE model, the "sequential" model proposed by Lee et al. (2019), which is one of the best performing models. We use this model both as generation models that create synthetic *mt* with our two proposed methods and also as the final APE models to examine the effectiveness of those synthesized data as additional training data. We follow the hyperparameter setting described in Lee et al. (2019), which again follows almost the same setting of the "base" Transformer described in the original paper Vaswani et al. (2017). However, we adjust the warm-up rate to 15,000 and the batch size to 25,000. We used OpenNMT-py[2] to implement and execute all models.

Synthetic Data Generation. To prevent our data generation model from generating what it has already seen during the training phase, we adopt the n-fold jack-knifing technique, which splits the whole dataset into $n-1$ folds for training and 1 left-out fold for generation and validation, into our data generation process. Specifically,

1. Split eSCAPE into $n = 8$ folds: $\{f_i\}_{i=1}^{8}$.
2. Construct a training set, $\mathcal{D}_i = \text{Append}\left(\{f_j\}_{j=1}^{8} \setminus \{f_i\}\right)$.
3. Train a data generation model (FG or BG) $\mathcal{M}_i$ on $\mathcal{D}_i$ and use 2,000 randomly extracted held-out samples from f_i for validation.
4. At a given model checkpoint, generate $\widetilde{mt}_i$ with $\mathcal{M}_i$ by supplying it with the pair of two sources in f_i.
5. Construct $mt_{FG/BG} = \text{Append}\left(\{\widetilde{mt}_i\}_{i=1}^{8}\right)$.

To examine the optimal stop point (§3.1, §3.2), we saved a model checkpoint every 25K training steps up to 150K steps, where the model converges with respect to its validation perplexity; thus, we obtained 6 sets of synthetic *mt* for each one of the two methods. Finally, for each method, we trained 6 APE models by using each new set of triplets including synthetic *mt*; and choose one set of synthetic *mt* that reports the best performance on the WMT validation dataset.

Evaluation With assistance from the FG and BG methods, we have a set of synthetic APE triplets $\mathcal{S} = \{\text{eSCAPE}, \text{eSCAPE}_{FG}, \text{eSCAPE}_{BG}\}$ available for training. In our experiments, we trained several APE models on various combinations of synthetic triplets in $\mathcal{S}$ together with the WMT training datasets and then compared the evaluation results to investigate how each data configuration affects the model's APE performance. Finally, after training the models until their perplexities on the WMT development dataset converge, we evaluated them on the WMT test datasets. We considered two baselines: (1) TER between *mt* and *pe* of the test datasets and (2) the performance of the APE model that is trained only on eSCAPE; the former

---

[2] `https://github.com/OpenNMT/OpenNMT-py.git`

Task	Gen. type	New samples (Ratio)	TER
PBSMT	FG	6,041,622 (83.23 %)	53.35
	BG	6,444,517 (88.79 %)	49.11
NMT	FG	4,969,521 (68.46 %)	52.53
	BG	6,304,471 (86.86 %)	45.89

Table 4: Statistics of synthetic *mt* produced by each proposed scheme. TERs are computed between *mt* and *ref*. 'New samples (Ratio)' indicates the number of synthetic *mt* that do not overlap with *mt* in eSCAPE.

implies that no post-editing has occurred yet, and it is used as the official baseline for the WMT APE shared task.

5 Results and Discussion

Table 3 shows the evaluation results. We observed that when eSCAPE$_{\text{FG}}$ or eSCAPE$_{\text{BG}}$ is used instead of eSCAPE, the APE model's performance does not make a big difference from the eSCAPE baseline. One possible reason that we expect is the gap between those synthetic *mt* and *mt* in the WMT dataset; in other words, synthetic *mt* is not produced by an existing MT system.

Nevertheless, we found that when we augment eSCAPE with eSCAPE$_{\text{FG}}$ and/or eSCAPE$_{\text{BG}}$, the trained APE model shows consistent improvements in its APE performance and most of the improvements upon the eSCAPE baseline are statistically significant. Moreover, the results also surpass current state-of-the-art (except the ensemble models) APE models (Correia and Martins, 2019; Lopes et al., 2019), which are built on top of BERT (Devlin et al., 2019), thus contain more model parameters, and exploit a huge amount of monolingual data. We expect that these results are because, in addition to an increase in the total quantity of training samples, the integration of multiple synthetic datasets, each of which focuses on different aspects of APE from the other—eSCAPE contains actual MT outputs; on the other hand, synthetic triplets better satisfy the minimal-edit criterion—appears to have an effect on the models' APE performance.

We found that our proposed methods derive a large number of new *mt* (Table 4) from eSCAPE and also yield a more similar TER distribution to that of WMT data than that of eSCAPE in terms of not only the mean TER (Table 4) but also the decrease in KL-divergence (Figure 3).

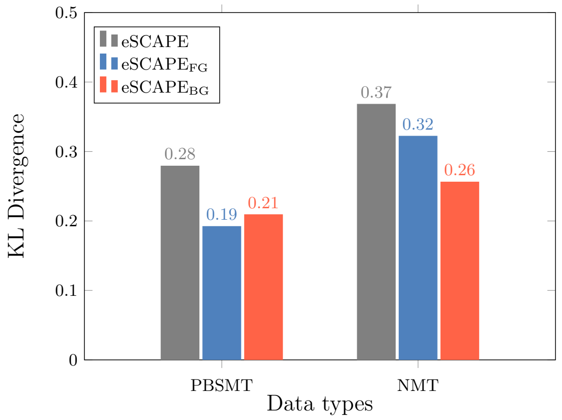


Figure 3: Presentation of $D_{KL}(P \parallel Q)$ (with base-10 logarithms) where P and Q are TER categorical distributions; P is for the WMT data, and Q is for each kind of synthetic triplets. The TER categorical distributions are plotted in the same manner as in Figure 1.

6 Conclusion and Future Work

In this paper, we tried to alleviate the drawbacks of eSCAPE by suggesting two new methods that adapt back-translation to the APE task, consequently increasing the data quantity and address the minimum editing characteristic. According to our experimental results, although APE models trained on each one of our two synthetic datasets show just comparable performances to the eSCAPE baseline, those trained on integrations of multiple synthetic datasets show consistent improvements over the baseline, implying that our new synthetic datasets are beneficial enlargements of eSCAPE. However, we manually selected the optimal stop points for both of our proposed generation schemes, so we will automate these selection processes in our future work.

Acknowledgments

We appreciate all of the reviewers giving their invaluable comments on this paper. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01906, Artificial Intelligence Graduate School Program (POSTECH)).

References

Jeffrey Allen and Christopher Hogan. 2000. Toward the development of a post editing module for raw machine translation output: A controlled language perspective. In *Proc. 3rd International Workshop on Controlled Language Applications (CLAW), 2000*, pages 62–71.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 Conference on Machine Translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics, Taberg Media Group AB.

Félix do Carmo, Dimitar Shterionov, Joss Moorkens, Joachim Wagner, Murhaf Hossari, Eric Paquin, Dag Schmidtke, Declan Groves, and Andy Way. 2020. A review of the state-of-the-art in automatic post-editing. *Machine Translation*, pages 1–43.

Rajen Chatterjee, Christian Federmann, Matteo Negri, and Marco Turchi. 2019. Findings of the WMT 2019 Shared Task on Automatic Post-Editing. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 11–28, Florence, Italy. Association for Computational Linguistics.

Rajen Chatterjee, Markus Freitag, Matteo Negri, and Marco Turchi. 2020. Findings of the WMT 2020 Shared Task on Automatic Post-Editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 646–659, Online. Association for Computational Linguistics.

Rajen Chatterjee, Matteo Negri, Raphael Rubino, and Marco Turchi. 2018. Findings of the WMT 2018 Shared Task on Automatic Post-Editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 710–725, Belgium, Brussels. Association for Computational Linguistics.

Rajen Chatterjee, Marion Weller, Matteo Negri, and Marco Turchi. 2015. Exploring the planet of the APEs: a comparative study of state-of-the-art methods for MT automatic post-editing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 156–161, Beijing, China. Association for Computational Linguistics.

Gonçalo M. Correia and André F. T. Martins. 2019. A simple and effective approach to automatic post-editing with transfer learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3050–3056, Florence, Italy. Association for Computational Linguistics.

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

Yacine Jernite. 2019. Unsupervised Text Summarization via Mixed Model Back-Translation. `arXiv:1908.08566 [cs.CL]`.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Log-linear Combinations of Monolingual and Bilingual Neural Machine Translation Models for Automatic Post-Editing. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 751–758, Berlin, Germany. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

WonKee Lee, Jaehun Shin, Baikjin Jung, Jihyung Lee, and Jong-Hyeok Lee. 2020. Noising Scheme for Data Augmentation in Automatic Post-Editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 783–788, Online. Association for Computational Linguistics.

WonKee Lee, Jaehun Shin, and Jong-Hyeok Lee. 2019. Transformer-based Automatic Post-Editing Model with Joint Encoder and Multi-source Attention of Decoder. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 112–117, Florence, Italy. Association for Computational Linguistics.

António V. Lopes, M. Amin Farajian, Gonçalo M. Correia, Jonay Trénous, and André F. T. Martins. 2019. Unbabel's Submission to the WMT2019 APE Shared Task: BERT-Based Encoder-Decoder for Automatic Post-Editing. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 118–123, Florence, Italy. Association for Computational Linguistics.

Matteo Negri, Marco Turchi, Rajen Chatterjee, and Nicola Bertoldi. 2018. ESCAPE: a Large-scale Synthetic Corpus for Automatic Post-Editing. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Shantipriya Parida and Petr Motlicek. 2019. Abstract Text Summarization: A Low Resource Challenge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5994–5998, Hong Kong, China. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000—-6010, Long Beach, California, USA. Curran Associates Inc.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and Denoising Natural Language: Diverse Backtranslation for Grammar Correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana. Association for Computational Linguistics.

Removing Word-Level Spurious Alignment between Images and Pseudo-Captions in Unsupervised Image Captioning

Ukyo Honda[1,3] **Yoshitaka Ushiku**[2] **Atsushi Hashimoto**[2]
Taro Watanabe[1] **Yuji Matsumoto**[3]
[1] Nara Institute of Science and Technology [2] OMRON SINIC X Corp.
[3] RIKEN Center for Advanced Intelligence Project
[1] `{honda.ukyo.hn6, taro}@is.naist.jp`
[2] `{yoshitaka.ushiku, atsushi.hashimoto}@sinicx.com`
[3] `yuji.matsumoto@riken.jp`

Abstract

Unsupervised image captioning is a challenging task that aims at generating captions without the supervision of image–sentence pairs, but only with images and sentences drawn from different sources and object labels detected from the images. In previous work, *pseudo-captions*, *i.e.*, sentences that contain the detected object labels, were assigned to a given image. The focus of the previous work was on the alignment of input images and pseudo-captions at the sentence level. However, pseudo-captions contain many words that are irrelevant to a given image. In this work, we investigate the effect of removing mismatched words from image–sentence alignment to determine how they make this task difficult. We propose a simple gating mechanism that is trained to align image features with only the most reliable words in pseudo-captions: the detected object labels. The experimental results show that our proposed method outperforms the previous methods without introducing complex sentence-level learning objectives. Combined with the sentence-level alignment method of previous work, our method further improves its performance. These results confirm the importance of careful alignment in word-level details.[1]

1 Introduction

Image captioning is a task to describe images in natural languages. This is a fundamental challenge with regard to automatically retrieving and summarizing the visual information in a human-readable form. Recently, considerable progress has been made (Vinyals et al., 2015; Xu et al., 2015; Anderson et al., 2018b) owing to the development of neural networks and a large number of annotated image–sentence pairs (Young et al., 2014; Lin et al., 2014; Krishna et al., 2017). However, these pairs are limited in their coverage of scenes[2], and scaling them is difficult owing to the cost of manual annotation.

Unsupervised image captioning (Feng et al., 2019) aims to describe scenes that have no corresponding image–sentence pairs, without requiring additional annotation of the pairs. The only available resources are images and sentences drawn from different sources and object labels detected from the images. Although it is highly challenging, unsupervised image captioning has the potential to cover a broad range of scenes by exploiting a large number of images and sentences that are not paired by expensive manual annotation.

To train a captioning model in this setting, previous work (Feng et al., 2019; Laina et al., 2019) employed sentences that contained the object labels detected from given images. We refer to these sentences as *pseudo-captions*. However, pseudo-captions are problematic in that they are likely to contain words that are irrelevant to the given images. Assume that an image contains two objects *cat* and *girl* (Figure 1). This situation could give rise to various possible pseudo-captions, *e.g.*, “a girl is holding a cat,” “a cat is sleeping with a girl,” “a girl is running with a cat.” When the first sentence is the correct caption of the image, the words *sleeping* and *running* of the other sentences are irrelevant to the image. As the detected object labels provide insufficient information to judge which sentence corresponds to the image, many pseudo-captions containing such mismatched words can be produced.

---

[1] Code will be available at `https://github.com/ukyh/RemovingSpuriousAlignment`

[2] For example, the standard captioning dataset MS COCO (Lin et al., 2014) covers only approximately 100 object categories out of 500 object categories defined in an object detection dataset (Agrawal et al., 2019). In addition to objects, attributes and relations are also not covered well owing to the small vocabulary size, 8791 (Karpathy and Fei-Fei, 2015).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3692–3702
April 19 - 23, 2021.

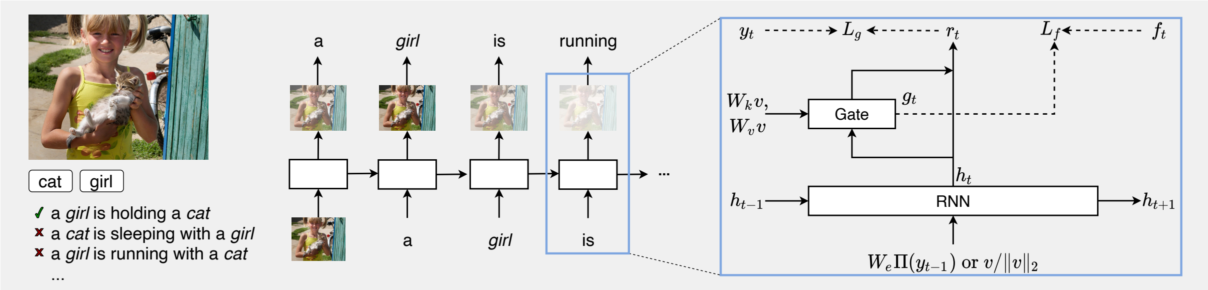


Figure 1: Overview of our model. The input is listed on the left-hand side: an image, its detected object labels, and its pseudo-captions. The model learns to generate the pseudo-captions while considering the correspondence between the image and each word being generated. The detailed process is shown in the blue box on the right-hand side. The base encoder–decoder model output $\boldsymbol{h}_t$, a gate value g_t, and a pseudo-label f_t on the gate are described in Sections 2.1, 2.2, and 2.3, respectively. The dashed arrows indicate the processes conducted only during training.

Regardless of the problem in pseudo-captions, previous work (Feng et al., 2019; Laina et al., 2019) did not explicitly remove word-level mismatches. They tried aligning the features of images and their pseudo-captions at the sentence level. Although this line of approach can potentially align the images and sentences correctly if there are sentences that exactly describe each image, it is not likely to hold for the images and sentences retrieved from different sources.

To shed light on the problem of word-level spurious alignment in the previous work, we focus on removing mismatched words from image–sentence alignment. To this end, we introduce a simple gating mechanism that is trained to exclude image features when generating words other than the most reliable words in pseudo-captions: the detected objects. The experimental results show that the proposed method outperforms previous methods without introducing complex sentence-level learning objectives. Combined with the sentence-level alignment method of previous work, our method further improves its performance. These results confirm the importance of careful alignment in word-level details.

2 Method

Our model comprises a sequential encoder–decoder model, a gating mechanism on the encoder–decoder model, a pseudo-label on the gating mechanism, and a decoding rule to avoid the repetition of object labels, as presented in Figure 1.

2.1 Base Encoder–Decoder Model

Typical supervised, encoder–decoder captioning models maximize the following objective function during training:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_{(\boldsymbol{I},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{I};\boldsymbol{\theta}), \tag{1}$$

where $\boldsymbol{\theta}$ are the parameters of the models, $\boldsymbol{I}$ is a given image, and $\boldsymbol{y} = y_1, ..., y_T$ is its corresponding caption, the last token y_T is a special end-of-sentence token.

However, in unsupervised image captioning, the corresponding caption $\boldsymbol{y}$ is not available. Instead, object labels in given images are provided by pretrained object detectors. Previous work utilized the detected object labels to assign a roughly corresponding caption $\hat{\boldsymbol{y}}$, *i.e.*, a pseudo-caption, to the given image. Following the previous work, we define pseudo-captions of an image as sentences containing the object labels detected from the image. Given the pseudo-caption $\hat{\boldsymbol{y}}$, our base encoder–decoder model maximizes the following objective function:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_{(\boldsymbol{I},\hat{\boldsymbol{y}})} \log p(\hat{\boldsymbol{y}}|\boldsymbol{I};\boldsymbol{\theta}). \tag{2}$$

In encoder–decoder captioning models, the probability $p(\boldsymbol{y}|\boldsymbol{I})$[3] is auto-regressively factorized as $p(\boldsymbol{y}|\boldsymbol{I}) = \prod_{t=1}^{T} p(y_t|y_{<t}, \boldsymbol{I})$ and each $p(y_t|y_{<t}, \boldsymbol{I})$ is computed by a single step of recurrent neural networks (RNNs). The encoder encodes the given image $\boldsymbol{I}$ to an image representation $\boldsymbol{v} \in \mathbb{R}^{d'}$ that is fed to the decoder as an initial input to generate a sequence of words auto-regressively. The detailed

---

[3] Hereafter, we omit the model parameter $\boldsymbol{\theta}$ for brevity.

computation of $p(\hat{y}_t|\hat{y}_{<t}, \boldsymbol{I})$ is as follows:

$$p(\hat{y}_t|\hat{y}_{<t}, \boldsymbol{I}) = \frac{\exp(\boldsymbol{h}_t^\top \boldsymbol{W}_o \Pi(\hat{y}_t))}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{h}_t^\top \boldsymbol{W}_o \Pi(y'))}, \quad (3)$$

$$\boldsymbol{h}_t = \begin{cases} \mathrm{Dec}\left(\frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_2}, \boldsymbol{h}_0\right), & \text{if } t = 1; \\ \mathrm{Dec}(\boldsymbol{e}_t, \boldsymbol{h}_{t-1}), & \text{otherwise}, \end{cases} \quad (4)$$

$$\boldsymbol{v} = \boldsymbol{W}_a \mathrm{Enc}(\boldsymbol{I}), \quad (5)$$

$$\boldsymbol{e}_t = \boldsymbol{W}_e \Pi(\hat{y}_{t-1}), \quad (6)$$

where Enc$(\cdot)$ is a pre-trained image encoder with a linear transformation matrix $\boldsymbol{W}_a \in \mathbb{R}^{d \times d'}$ on top of it, Dec$(\cdot)$ is an RNN decoder, $\Pi(\cdot)$ is the one-hot encoding function, $\boldsymbol{h}_0 \in \mathbb{R}^d$ is a zero vector, $\mathcal{Y}$ is the whole vocabulary to use, and $\boldsymbol{W}_e, \boldsymbol{W}_o \in \mathbb{R}^{d \times |\mathcal{Y}|}$ are the word embedding matrices. Details of the encoder and decoder are provided in Section 3.2.

2.2 Gating Mechanism to Consider Word-Level Correspondence

As indicated in Eq. 2, our base encoder–decoder model decodes all of the words in pseudo-captions from the images. However, pseudo-captions are highly likely to contain words that are irrelevant to the given images. Thus, forcing a model to decode the pseudo-captions in their entirety from the images might be more disadvantageous than beneficial for training precise captioning models.

To enable our model to handle word-level mismatches, we introduce a simple gating mechanism. Our model, which is equipped with this gating mechanism, takes an image representation at each t-th time step. The gate is designed to control the amount of image representation used to generate the t-th word. In other words, the gate is expected to determine the extent to which the given image corresponds to the t-th word. With a slight modification to Eq. 3, our model with the gating mechanism is defined as follows:

$$p(\hat{y}_t|\hat{y}_{<t}, \boldsymbol{I}) = \frac{\exp(\boldsymbol{r}_t^\top \boldsymbol{W}_o \Pi(\hat{y}_t))}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{r}_t^\top \boldsymbol{W}_o \Pi(y'))}, \quad (7)$$

$$\boldsymbol{r}_t = g_t \frac{\boldsymbol{W}_v \boldsymbol{v}}{\|\boldsymbol{W}_v \boldsymbol{v}\|_2} + (1 - g_t)\boldsymbol{h}_t, \quad (8)$$

$$g_t = \mathrm{sigmoid}(\tanh(\boldsymbol{W}_k \boldsymbol{v})^\top \boldsymbol{h}_t), \quad (9)$$

where $\boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{d \times d}$ are the linear transformation matrices for computing the gate value $g_t \in [0, 1]$ and the output of the gate $\boldsymbol{r}_t \in \mathbb{R}^d$. When g_t is close to one, it forces the model to use more information from the image ($\boldsymbol{v}$) to generate the t-th word; when g_t is close to zero, it forces the model to do the opposite.

The fed image representation $\boldsymbol{W}_v \boldsymbol{v}$ is kept constant at every time step t. Thus, even when the t-th word is correctly pictured in the image $\boldsymbol{I}$, $\boldsymbol{W}_v \boldsymbol{v}$ itself cannot determine which specific object in the image should be generated according to the current context in the output caption. Therefore, we apply L2 normalization to the image representation in Eq. 8 to ensure that a relatively greater amount of the contextual information ($\boldsymbol{h}_t$) is used.

To train our model with the gating mechanism, we minimize the following cross-entropy loss for each pair of images and their pseudo-captions:

$$\mathcal{L}_g = -\frac{1}{T} \sum_{t=1}^{T} \log p(\hat{y}_t|\hat{y}_{<t}, \boldsymbol{I}). \quad (10)$$

2.3 Pseudo-Labels on Gate to Remove Word-Level Spurious Alignment

The above gate is expected to reflect the correspondence between images and words in pseudo-captions. However, learning to reflect the correspondence correctly is difficult for the gate under the noisy and weak supervision of pseudo-captions.

In this work, our focus is to remove the spurious alignment between images and words in pseudo-captions. Consequently, we apply the following rule to the gate that largely suppresses image representations to use: g_t should be close to one if the t-th word to generate is a detected object label; otherwise, it should be close to zero. This is based on the assumption that, given an image and its pseudo-caption, the reliable words in the pseudo-caption are only the detected object labels, and the others are likely to be irrelevant to the image.

We assign a pseudo-label $f \in \{0, 1\}$ on the gate: $f_t = 1$ if the word $\hat{y}_t$ corresponds to any of the object labels detected from a given image; otherwise, $f_t = 0$. The gate then learns the correspondence by minimizing the following loss function:

$$\mathcal{L}_f = -\frac{1}{T} \sum_{t=1}^{T} \Big[\alpha f_t \log g_t + (1 - f_t) \log(1 - g_t)\Big], \quad (11)$$

where α is the weight to emphasize the loss when $f_t = 1$. A relatively large value is recommended for α to prevent g_t from always being zero because the number of detected object labels (where $f_t = 1$)

	Training Text	Object Detector	Image Encoder	Text Decoder
A (Feng et al., 2019)	SS	Faster-RCNN trained on OpneImages-v2	Inception-v4	1-layer LSTM of 512 dimensions
B (Laina et al., 2019)	GCC	Faster-RCNN trained on OpneImages-v4	ResNet-101	1-layer GRU of 200 dimensions

Table 1: Summary of the difference in the experimental settings.

in pseudo-captions is generally smaller than the number of the other words (where $f_t = 0$).

Combined with the loss function of Eq. 10, the final loss function is defined as follows:

$$\mathcal{L} = \mathcal{L}_g + \mathcal{L}_f. \tag{12}$$

2.4 Unique-Object Decoding

An evaluation of our model revealed that it tends to repeat words in object categories. Although repetition is common in encoder–decoder models, this repetition was generated owing to a different cause. As mentioned in Section 2.2, the image representation $\boldsymbol{v}$ cannot correctly predict the word $\hat{y}_t$ without the context representation $\boldsymbol{h}_t$; if the gate value g_t is exactly one, the model always outputs the most salient object label in the given image.

To avoid ignoring contextual information, we applied a simple decoding rule during the evaluation. Given that the model generates a word y_t at t-th time step, our decoding rule checks whether y_t is in predefined object categories, *i.e.*, object categories defined for object detectors. If y_t is found in the object categories, the rule forces the probability of generating y_t to be zero in the subsequent time steps.

3 Experiments

We ran the experiments under two different settings, Feng et al. (2019) and Laina et al. (2019), for a fair comparison with each. For brevity, we refer to the settings in Feng et al. (2019) and Laina et al. (2019) as setting A and B, respectively. The difference of the settings is clarified in Table 1.

3.1 Datasets

Evaluation Set. To evaluate our proposed method, we used the MS COCO dataset (Lin et al., 2014) with the validation/test split defined by Karpathy and Fei-Fei (2015). Each split has 5,000 images and five reference captions for each image.

Training Images. We used the images (without their captions) in the remaining training split of MS COCO (113,286 images), and a pre-traind object detector (Huang et al., 2017) to retrieve the object labels found in the images[4]. The detector is a publicly available Faster-RCNN model[5] (Ren et al., 2015). The training data of the object detector differs depending on the previous work; thus, we used the object detector trained on OpenImages-v2 (Krasin et al., 2017) to compare with Feng et al. (2019) and that trained on OpenImages-v4 (Kuznetsova et al., 2020) to compare with Laina et al. (2019). Note that these object detectors were not trained on MS COCO images. Following the previous work, we refrained from using the detected bounding boxes and their features.

Training Text. Following the previous work, we used the Shutterstock image description corpus (SS) (Feng et al., 2019) and the training split captions (without images) of Google's Conceptual Captions (GCC) (Sharma et al., 2018) for comparison with Feng et al. (2019) and Laina et al. (2019), respectively. SS consists of 2.3M image descriptions crawled from Shutterstock, an online stock photography website; GCC consists of 3.3M image descriptions crawled from the web. Note that these sentences are not the descriptions of the images in MS COCO.

3.2 Implementation Details

Image Encoder. For a fair comparison with the previous work, we employed different image encoders depending on the compared method: Inception-v4 (Szegedy et al., 2017) in the settings of Feng et al. (2019) and ResNet-101 (He et al., 2016a,b) in the settings of Laina et al. (2019). Both image encoders were pre-trained on ImageNet (Russakovsky et al., 2015) and are publicly available[6]. The parameters of the image encoder were fixed during training and prediction.

Text Decoder. Similar to the image encoder, we

---

[4] Although this pre-trained object detector requires bounding box and semantic label annotations, it can be replaced with any multi-label image classifier, which can be trained on image-tag pairs that are largely and freely available on the web. To ensure this compatibility, bounding box features are not used in unsupervised image captioning.

[5] `https://github.com/tensorflow/models/tree/master/research/object_detection`

[6] `https://github.com/tensorflow/models/tree/master/research/slim`

		BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE
A	Feng et al. (2019)	41.0	22.5	11.2	5.6	12.4	28.7	28.6	8.1
	Ours	**49.5 $\pm$ 0.7**	**27.3 $\pm$ 1.2**	**13.1 $\pm$ 0.8**	**6.3 $\pm$ 0.5**	**14.0 $\pm$ 0.1**	**34.5 $\pm$ 0.3**	**31.9 $\pm$ 1.0**	**8.6 $\pm$ 0.2**
B	Laina et al. (2019)				6.5	12.9	35.1	22.7	
	Ours	50.4 $\pm$ 1.5	29.5 $\pm$ 0.8	14.4 $\pm$ 0.5	**7.6 $\pm$ 0.4**	**13.5 $\pm$ 0.3**	**37.3 $\pm$ 0.2**	**31.8 $\pm$ 0.7**	8.4 $\pm$ 0.1

Table 2: Comparison with the state-of-the-art results on the experimental settings A and B. The scores of our model are the *mean* $\pm$ *standard deviation* of five runs. The scores obtained for BLEU-1 to 3 and SPICE are not provided in the original paper of Laina et al. (2019).

used a different RNN as our decoder: LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014) to enable us to compare our results with those of Feng et al. (2019) and Laina et al. (2019), respectively. Following the previous work, the number of hidden layers' dimensions was set to 512 for LSTM and 200 for GRU. The number of the RNN layer was set to one. Word embeddings were randomly initialized and had the same dimensions as the RNN hidden layer.

Pseudo-Captions. Captions tend to describe salient objects, not all detected objects. For example, the frequent object *person* often co-occurs with *face* and *clothing* in images, but these three are not always the salient objects to be described in a caption. To avoid collecting the pseudo-captions that only contain these frequent objects, we picked up each detected object and their pairs to retrieve pseudo-captions, rather than using all detected objects. In this retrieval, we converted object labels to their plural forms using a dictionary used in Feng et al. (2019) so that the pseudo-captions could also cover the plural forms of the objects.

Pseudo-Caption Preprocessing. For each pair of objects, we selected sentences where fewer than four words existed between the objects. This is to pick up the sentences likely to describe the relations of the target objects. We then removed the sentences wherein the target objects were adjacent to avoid collecting the objects' compound words. For each object, we selected sentences wherein fewer than two words were in between the object and its dependent adjective to pick up the sentences likely to describe the object in detail. We used spaCy[7] en_core_web_lg model for parsing.

Value of α. As described above, each pseudo-caption contains only one or two detected objects, which is very few compared with the average sentence lengths of the text corpora (12.0 in SS and 10.7 in GCC). To balance the label imbalance of f_t, we searched the value for α (Eq. 11) at a power of 2 and found that $\alpha = 16$, which roughly equals the quotient of $\frac{\text{Sentence Length}}{\text{Detected Objects}}$, worked well across the settings.

Training Iteration. After collecting the pseudo-captions, we created a set of the objects and pairs that were used to collect the pseudo-captions. The training is iterated over the pairs in this set, rather than over each image, to avoid overfitting for the most frequent object labels. On each iteration of the pairs of objects, we randomly sampled the image and pseudo-caption, wherein both of the objects were contained. Likewise, we did the same sampling on each object in the pairs. The number of the object pairs was 11,607 and 10,612 in the settings A and B, respectively. We set the batch size to eight and terminated the training when the best validation score (specifically, the CIDEr score) did not exceed for 20 epochs. For the optimizer, we used Adam with the recommended hyperparameters (Kingma and Ba, 2015).

Evaluation. In the evaluation, we set the maximum decoding length to 20. Our model decoded captions by using greedy search and unique-object decoding, described in Section 2.4. The evaluation metrics we used were BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Denkowski and Lavie, 2014), CIDEr (Vedantam et al., 2015) and SPICE (Anderson et al., 2016).

3.3 Comparison with the State-of-the-Art Results

Table 2 lists the results of our model compared with the previous state-of-the-art results. To avoid evaluating cherry-picked scores, we computed the mean and standard deviation of five results obtained with different seeds[8]. Our method outperforms the previous approaches in terms of all evaluation metrics. These results confirm the effectiveness of our simple method.

---

[7] `https://spacy.io`

[8] In all the experiments, we specified a seed of 0, 1, 2, 3, 4 for each run.

		gate	*pseudoL*	*unique*	*image*	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE
A	Ours (full)	✓	✓	✓	✓	**49.5**	**27.3**	**13.1**	6.3	14.0	34.5	**31.9**	**8.6**
	w/o *pseudoL*	✓		✓	✓	0.0	0.0	0.0	0.0	0.0	0.5	0.9	0.3
	w/o *gate*			✓	✓	40.9	21.5	10.1	4.8	12.7	32.1	17.6	6.0
	w/o *unique*	✓	✓		✓	47.2	26.2	13.0	**6.4**	**14.1**	**34.9**	28.3	8.5
	w/o *image*	✓	✓	✓		43.3	23.3	10.8	5.1	13.1	31.7	25.5	7.8
B	Ours (full)	✓	✓	✓	✓	**50.4**	**29.5**	**14.4**	**7.6**	**13.5**	**37.3**	**31.8**	**8.4**
	w/o *pseudoL*	✓		✓	✓	44.5	25.4	12.2	6.2	12.4	36.7	29.2	7.5
	w/o *gate*			✓	✓	44.5	24.2	12.0	6.2	11.6	34.2	19.4	5.8
	w/o *unique*	✓	✓		✓	47.9	27.1	13.0	6.4	12.6	36.3	26.9	7.4
	w/o *image*	✓	✓	✓		47.1	26.0	12.8	6.6	13.1	34.7	29.7	8.0

Table 3: Ablation studies on the experimental settings A and B. The scores of Ours (full) are the mean of five runs; those of the other ablated models are the results of a single run.

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr	SPICE
Feng et al. (2019)	41.0	22.5	11.2	5.6	12.4	28.7	28.6	8.1
Ours	49.5 ± 0.7	**27.3 ± 1.2**	**13.1 ± 0.8**	6.3 ± 0.5	**14.0 ± 0.1**	34.5 ± 0.3	31.9 ± 1.0	8.6 ± 0.2
Ours + Feng et al. (2019)	**50.9 ± 0.1**	**28.0 ± 0.1**	**14.0 ± 0.1**	**7.1 ± 0.0**	**14.1 ± 0.0**	**35.2 ± 0.1**	**35.7 ± 0.1**	**9.2 ± 0.0**

Table 4: Results of combining our method with previous methods (Feng et al., 2019). Scores of our model and the combined model are the *mean* ± *standard deviation* of five runs. We marked in bold the scores within the standard deviation of the best scores.

3.4 Ablation Study

Table 3 lists the results of our model obtained in the ablation studies. We tested the ablation of the gating mechanism (*gate*), pseudo-labels on the gating mechanism (*pseudoL*), unique-object decoding (*unique*), and image features (*image*). The pseudo-labels cannot be implemented without the base gating mechanism. Thus, the model "w/o *gate* w/ *pseudoL*" is not applicable. The model w/o *image* is the same as Ours (full) except that it only uses the word embeddings of detected object labels, rather than image features. It encodes detected object labels into word embeddings and then takes their mean[9] and replaces the image feature $\boldsymbol{v}$ with it. All models here were trained in the same manner as described in Section 3.2.

The results show that the pseudo-labels on the gating mechanism contribute a lot to the performance; the score degrades significantly from Ours (full) to w/o *pseudoL* in all metrics. On the other hand, the base gating mechanism does not function well by itself; not all scores of w/o *gate* are lower than those of w/o *pseudoL*. These results demonstrate that explicitly removing the word-level spurious alignment contributes the most to the relatively high performance of our model. Although it is a relatively low contribution compared with the pseudo-labels, unique-object decoding also enhances performance.

The degraded performance of w/o *image* suggests that object labels themselves are insufficient to describe images correctly. We observed that this model was vulnerable to errors propagated through object detectors. See Section 3.8 for the examples.

3.5 Combining with Previous Methods

Our method focuses on removing word-level spurious alignment between images and pseudo-captions, whereas the previous methods focus on aligning images and pseudo-captions at the sentence level. To utilize the strength of each, we combined our method with the previous methods as a model initialization method.

We first trained our model on the setting A and generated captions for the images in training data. We then paired the captions with the images as their pseudo-captions[10]. With the pairs, the caption generator of Feng et al. (2019) was initialized by learning to generate the pseudo-captions from the images. After the initialization, we trained the previous model using their publicly available code[11]. We used the same hyperparameters as Feng et al. (2019) except for the learning rate of 10^{-5} for the

---

[9] The number of detected objects was 3.0 in setting A and 4.0 in setting B on average. Thus, taking the mean does not break the detected information significantly.

[10] To avoid assigning obviously incorrect pseudo-captions, we omitted the captions that contained fewer than one detected object for the images with more than two detected objects. For the images with fewer than one detected object, we omitted the captions that contained no detected objects.

[11] `https://github.com/fengyang0317/unsupervised_captioning`

		Precision	Recall	F1
Detected	Feng et al. (2019)	**56.6**	57.4	**55.4**
	Ours	51.0	56.7	51.6
	Ours + Feng et al. (2019)	54.0	**61.8**	**55.4**
Others	Feng et al. (2019)	22.3	17.0	18.8
	Ours	27.8	**21.9**	23.4
	Ours + Feng et al. (2019)	**29.9**	**21.9**	**24.2**

Table 5: Bag-of-words matching scores with respect to detected object labels and the other words.

generator and 10^{-8} for the discriminator.

Table 4 shows the results. The combined model further improves the performance of our model and Feng et al. (2019). In particular, the improvement from Feng et al. (2019) is much larger than that from our model. These results suggest that removing the word-level spurious alignment is critical for the subsequent sentence-level alignment.

3.6 Negative Effect of Spurious Alignment

To further investigate the effect of removing the spurious alignment, we evaluated our model on *noisier words*: words other than the detected object labels. Our method discourages from aligning them with images because they are likely to be irrelevant to given images, while previous methods force the alignment. We tested the following bag-of-words matching on the MS COCO test set.

Let S be the bag of words of a caption generated from an image $\boldsymbol{I}$ and T^m be the m-th reference caption of $\boldsymbol{I}$. Given a set of detected object labels $\mathcal{O}$ of $\boldsymbol{I}$, we took the intersections $S_{det} = S \cap \mathcal{O}$ and $S_{other} = S \cap \overline{\mathcal{O}}$ for S, as well as for T^m. We define the precision (P), recall (R) and F1 score (F) of S against T^m as follows: $P = \frac{|S \cap T^m|}{|S|}$, $R = \frac{|S \cap T^m|}{|T^m|}$, $F = 2 \cdot \frac{P \cdot R}{P+R}$. Based on this, we define the precision, recall, and F1 score of S_{det} against T^m_{det} by replacing S with S_{det} and T^m with T^m_{det}, and likewise for those of S_{other} against T^m_{other}. We calculated the above scores for each pair of a generated captions and their reference captions and subsequently averaged it across the pairs. The pairs with empty T^m_* were excluded from the calculation.

Table 5 shows the results. Overall, the scores on detected object labels (Detected) are about two times higher than those on the other words (Others), indicating the difficulty of learning the alignment of the latter, noisier words. Our model performs better in predicting the noisier words, outperforming Feng et al. (2019) in all metrics. These results indicate that refraining from the alignment works better than forcing it for the noisier words.

		Word Type	Frequency
Object	Feng et al. (2019)	205	20013
	Ours	306	15052
	Ours + Feng et al. (2019)	239	18226
Others	Feng et al. (2019)	827	24865
	Ours	169	83693
	Ours + Feng et al. (2019)	121	110358

Table 6: Analysis of generated captions with respect to object labels and the other words. Word Type is the number of unique words, and Frequency is the mean of the frequency of the words in the training text corpus.

On the other hand, our model performs worse in predicting detected object labels. This is because our method trusts all detected object labels and aligns them with images without any constraints used in previous work. Combined with the previous method (Ours + Feng et al. (2019)), our model improves the prediction on detected object labels.

3.7 Positive Effect of Frequency

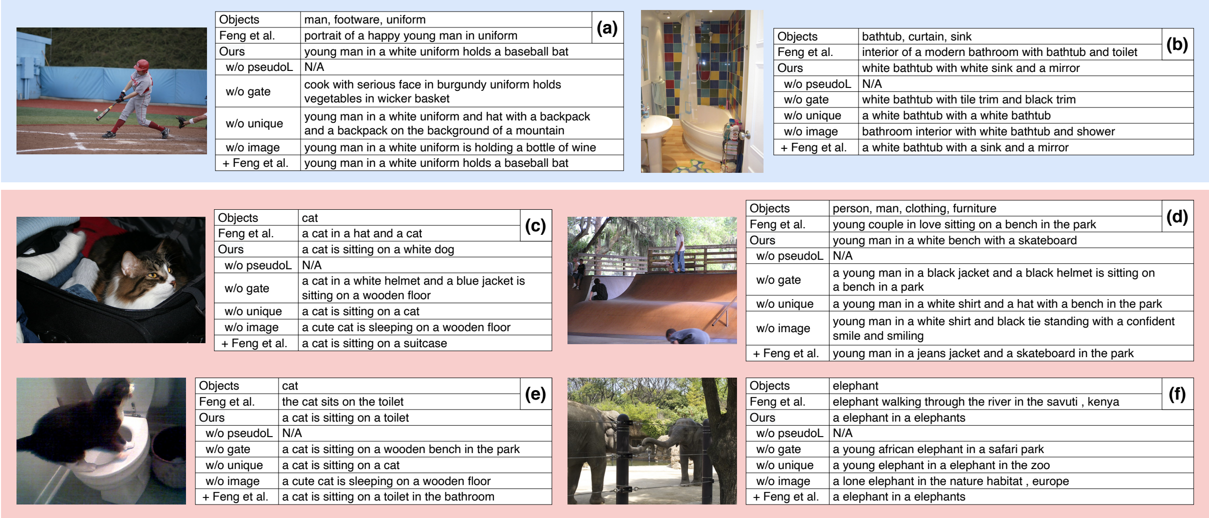
By assigning the pseudo-label f, our method encourages to align detected object labels with the image representation $\boldsymbol{v}$ and the other words with the contextual representation $\boldsymbol{h}$. Thus, our model is likely to predict the latter words mostly based on the previous output sequences, as language models do. If this is the case, then the latter words predicted tend to be the frequent words in the training text corpus.

To verify this tendency, we analyzed the frequency of output words in the training text corpus for object labels and the other words[12]. Table 6 presents the results. In contrast to object labels, our outputs' vocabulary is about five times smaller than that of Feng et al. (2019), and the words tend to be highly frequent in the training text corpus.

The results also show that a model performs better if it has the smaller and more frequent vocabulary of the words other than object labels. This correlation is convincing considering the coverage of frequent words. For example, a general caption such as "a man *with* a bike" can correctly cover various scenes in which a man is riding/sitting on/leaning on/standing near/... a bike. This positive effect of frequency suggests that firstly aligning the frequent words and gradually extending them can

---

[12] As we analyzed each unique word across all output captions in the MS COCO test set, we roughly divided the words into object labels and the others, not into detected object labels and the others.



(a)

Objects	man, footware, uniform
Feng et al.	portrait of a happy young man in uniform
Ours	young man in a white uniform holds a baseball bat
w/o pseudoL	N/A
w/o gate	cook with serious face in burgundy uniform holds vegetables in wicker basket
w/o unique	young man in a white uniform and hat with a backpack and a backpack on the background of a mountain
w/o image	young man in a white uniform is holding a bottle of wine
+ Feng et al.	young man in a white uniform holds a baseball bat

(b)

Objects	bathtub, curtain, sink
Feng et al.	interior of a modern bathroom with bathtub and toilet
Ours	white bathtub with white sink and a mirror
w/o pseudoL	N/A
w/o gate	white bathtub with tile trim and black trim
w/o unique	a white bathtub with a white bathtub
w/o image	bathroom interior with white bathtub and shower
+ Feng et al.	a white bathtub with a sink and a mirror

(c)

Objects	cat
Feng et al.	a cat in a hat and a cat
Ours	a cat is sitting on a white dog
w/o pseudoL	N/A
w/o gate	a cat in a white helmet and a blue jacket is sitting on a wooden floor
w/o unique	a cat is sitting on a cat
w/o image	a cute cat is sleeping on a wooden floor
+ Feng et al.	a cat is sitting on a suitcase

(d)

Objects	person, man, clothing, furniture
Feng et al.	young couple in love sitting on a bench in the park
Ours	young man in a white bench with a skateboard
w/o pseudoL	N/A
w/o gate	a young man in a black jacket and a black helmet is sitting on a bench in a park
w/o unique	a young man in a white shirt and a hat with a bench in the park
w/o image	young man in a white shirt and black tie standing with a confident smile and smiling
+ Feng et al.	young man in a jeans jacket and a skateboard in the park

(e)

Objects	cat
Feng et al.	the cat sits on the toilet
Ours	a cat is sitting on a toilet
w/o pseudoL	N/A
w/o gate	a cat is sitting on a wooden bench in the park
w/o unique	a cat is sitting on a cat
w/o image	a cute cat is sleeping on a wooden floor
+ Feng et al.	a cat is sitting on a toilet in the bathroom

(f)

Objects	elephant
Feng et al.	elephant walking through the river in the savuti , kenya
Ours	a elephant in a elephants
w/o pseudoL	N/A
w/o gate	a young african elephant in a safari park
w/o unique	a young elephant in a elephant in the zoo
w/o image	a lone elephant in the nature habitat , europe
+ Feng et al.	a elephant in a elephants

Figure 2: Sample captions for six input images taken from the MS COCO validation set. Our model generated correct captions for the images in the top row and wrong captions for the rest. Best viewed by zooming in.

be a promising approach.

3.8 Qualitative Analysis on Outputs

Figure 2 shows the captions generated by our model, its ablated models, Feng et al. (2019), and the combined model trained on the setting A. Our model generated correct captions for images (a) and (b). It successfully generated object labels that were not even detected by the object detector: *bat* in (a) and *mirror* in (b). On the other hand, errors of the object detector directly propagated to the output captions of w/o *image* model: the model generated an incorrect object *a bottle of wine*, owing to the missing object *bat* in (a).

Captions of the other images are negative results of our model. We observed that our model tended to repeat similar objects: *cat* and *dog* in (c), and *elephant* and *elephants* in (f). Without unique-object decoding, this tendency got worse: w/o *unique* model repeated *cat* in (c) and (e), and *elephant* in (f). Ours + Feng et al. (2019) model did not change much of the prediction of our model, as we set the learning rate low (see Section 3.5). However, it allowed the partial correction seen in (c): the combined model modified *dog* to *suitcase*.

In our outputs, words other than object labels tended to be frequent words and composed short phrases. On the contrary, Feng et al. (2019) tended to generate less frequent words (*savuti* and *kenya* in (f)) and longer phrases (*portrait of a happy young* in (a) and *young couple in love* in (d)), which were incorrect predictions in these examples.

Figure 3 shows output captions of our model and the gate values for each word. Overall, the gate values were high for object labels and low for the other words. Although our model was correct on the words other than object labels in these examples, these words were generated mostly by contextual features, thus heavily relied on contextual frequency. This heavy reliance on contexts resulted in generating the same word after an object label without considering images: *is sitting on* followed *cat* in both (c) and (e).

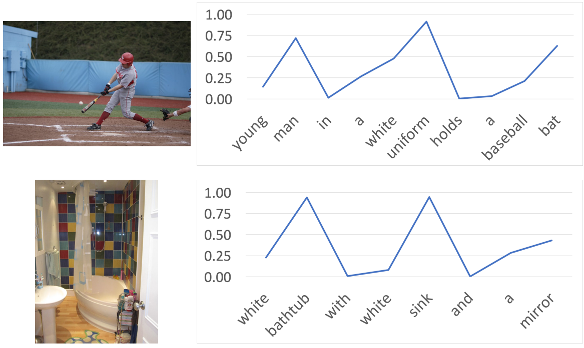


Figure 3: Sample captions with gate values. The plot represents the values of g_t for each predicted word. The value of g_t becomes high when the word is predicted using mainly image representation.

4 Related Work

There has been considerable research with different settings and approaches to describe scenes that have no image–sentence pairs. Novel object captioning (Hendricks et al., 2016; Venugopalan et al., 2017; Anderson et al., 2018a; Agrawal et al., 2019) attempted describing unseen objects in captions.

They incorporated an image classifier or object detector trained on objects not included in image–sentence pairs. Lu et al. (2018) tested captioning models on the generation of unseen combinations of objects, and Nikolaus et al. (2019) extended this to the unseen combinations of objects, attributes, and relations. In both settings, only the combinations were unseen, but each word in the combinations appeared in the training data. Semi-supervised approaches utilized caption retrieval models to automatically collect the corresponding captions for unannotated images to augment image–sentence pairs (Liu et al., 2018; Kim et al., 2019).

The above work was evaluated on the scenes where correct descriptions partially overlapped with those in the training image–sentence pairs. However, there can be scenes with no such overlap due to the limited coverage of the currently available image–sentence pairs. Taking a step further, unsupervised image captioning (Feng et al., 2019; Laina et al., 2019) aims to describe scenes that have no overlap with the image–sentence pairs, without the annotation of the pairs. To test in that situation, the task does not allow to use any image–sentence pairs. The only available resources are images and sentences drawn from different sources and object labels detected from the images.

Feng et al. (2019) first trained an encoder–decoder model that takes object labels in a sentence as its input and outputs the original sentence. After training, this model took the object labels detected from each image and outputted a sentence to pair with the image as its pseudo-caption. These pairs were then used to initialize a caption generator for the subsequent image–sentence alignment: bi-directional (image-to-sentence and sentence-to-image) feature reconstruction and GAN training (Goodfellow et al., 2014) to ensure fluency in generated captions. In the work of Laina et al. (2019), pseudo-captions were sentences that contained object labels detected from a given image. They employed metric learning and GAN training to minimize the difference between images and pseudo-captions in their latent space, as well as to maximize the difference between images and sentences wherein no detected object label was included.

Our approach is different from them in that it focuses on removing the mismatched words of pseudo-captions to take reliable supervision only, rather than forcing the use of the entire pseudo-captions for image–sentence alignment. Although the previous work additionally ensured to align detected object labels to images, they did not prevent the spurious alignment between images and words.

As an eased setting of unsupervised image captioning, unpaired image captioning has also been explored (Feng et al., 2019; Laina et al., 2019; Gu et al., 2019; Liu et al., 2019). The major difference from unsupervised image captioning is that images and sentences are drawn from image–sentence pairs, rather than from different sources. That is, every image has completely matched captions in pseudo-captions, which is not the case in unsupervised image captioning. As correct captions exist for each image, previous approaches focused on matching images and sentences at the sentence level. Contrary to these approaches, we focus on employing unsupervised image captioning and devising a method to remove word-level spurious alignment in the much noisier pseudo-captions.

Another variation of unpaired image captioning is the generation of captions in one language that has no image–sentence pairs, using paired images and captions in another language (Gu et al., 2018; Song et al., 2019). However, this line of research is beyond the scope of our work, as it requires image–sentence pairs to be at least in one language.

Our gating mechanism borrowed the idea of adaptive attention (Lu et al., 2017, 2018). Adaptive attention serves to control when generating words from image representations. Although these methods assume that the control is automatically learned from image–sentence pairs, this is not the case in an unsupervised setting. Our method is different from theirs in that we add heuristic pseudo-labels to train the gate when using image representations.

5 Conclusion

We investigated the importance of removing word-level spurious alignment between images and pseudo-captions in the task of unsupervised image captioning. For this purpose, we introduced a simple gating mechanism trained to align image features with only the most reliable words in pseudo-captions. The experimental results showed that our proposed method outperformed the previous methods without the sentence-level learning objectives used in the previous methods. Moreover, our method improved the performance further by combining with the previous methods. These results confirmed the importance of careful alignment in word-level details.

References

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. In *ICCV*.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *ECCV*.

Peter Anderson, Stephen Gould, and Mark Johnson. 2018a. Partially-supervised image captioning. In *NeurIPS*.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018b. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *the ninth workshop on statistical machine translation*.

Yang Feng, Lin Ma, Wei Liu, and Jiebo Luo. 2019. Unsupervised image captioning. In *CVPR*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NeurIPS*.

Jiuxiang Gu, Shafiq Joty, Jianfei Cai, and Gang Wang. 2018. Unpaired image captioning by language pivoting. In *ECCV*.

Jiuxiang Gu, Shafiq Joty, Jianfei Cai, Handong Zhao, Xu Yang, and Gang Wang. 2019. Unpaired image captioning via scene graph alignments. In *ICCV*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *CVPR*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Identity mappings in deep residual networks. In *ECCV*.

Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, and Trevor Darrell. 2016. Deep compositional captioning: Describing novel object categories without paired training data. In *CVPR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. 2017. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

Dong-Jin Kim, Jinsoo Choi, Tae-Hyun Oh, and In So Kweon. 2019. Image captioning with very scarce supervised data: Adversarial semi-supervised learning approach. In *EMNLP-IJCNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. 2017. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. 2020. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*.

Iro Laina, Christian Rupprecht, and Nassir Navab. 2019. Towards unsupervised image captioning with shared multimodal embeddings. In *ICCV*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Fenglin Liu, Meng Gao, Tianhao Zhang, and Yuexian Zou. 2019. Exploring semantic relationships for image captioning without parallel data. In *ICDM*.

Xihui Liu, Hongsheng Li, Jing Shao, Dapeng Chen, and Xiaogang Wang. 2018. Show, tell and discriminate: Image captioning by self-retrieval with partially labeled data. In *ECCV*.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018. Neural baby talk. In *CVPR*.

Mitja Nikolaus, Mostafa Abdou, Matthew Lamm, Rahul Aralikatte, and Desmond Elliott. 2019. Compositional generalization in image captioning. In *CoNLL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*.

Yuqing Song, Shizhe Chen, Yida Zhao, and Qin Jin. 2019. Unpaired cross-lingual image caption generation with self-supervised rewards. In *ACMMM*.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *CVPR*.

Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2017. Captioning images with diverse objects. In *CVPR*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *CVPR*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*.

Towards More Fine-grained and Reliable NLP Performance Prediction

Zihuiwen Ye, Pengfei Liu, Jinlan Fu†, Graham Neubig
Carnegie Mellon University, † Fudan University
`{zihuiwey,pliu3,gneubig}@andrew.cmu.edu,`
`fujl16@fudan.edu.cn`

Abstract

Performance prediction, the task of estimating a system's performance without performing experiments, allows us to reduce the experimental burden caused by the combinatorial explosion of different datasets, languages, tasks, and models. In this paper, we make two contributions to improving performance prediction for NLP tasks. First, we examine performance predictors not only for holistic measures of accuracy like F1 or BLEU, but also *fine-grained* performance measures such as accuracy over individual classes of examples. Second, we propose methods to understand the *reliability* of a performance prediction model from two angles: confidence intervals and calibration. We perform an analysis of four types of NLP tasks, and both demonstrate the feasibility of fine-grained performance prediction and the necessity to perform reliability analysis for performance prediction methods in the future. We make our code publicly available: `https://github.com/neulab/Reliable-NLPPP`

1 Introduction

Performance prediction (P^2) aims to predict a machine learning system's performance based on features of the underlying problem, dataset, or learning algorithm. While this topic is still relatively unexplored in the NLP context, there are a few examples of predicting performance as: (i) a function of training or model parameters for determining the number of training iterations (Kolachina et al., 2012) or value of hyperparameters (Rosenfeld et al., 2019) and identifying and terminating bad training runs (Domhan et al., 2015). (ii) a function of dataset characteristics to illustrate which factors are significant predictors of system performance (Birch et al., 2008; Turchi et al., 2008), or find a subset of representative experiments to run in order to obtain plausible predictions (Xia et al., 2020). In this paper, we ask two research questions with respect to performance prediction: can we predict performance on a more *fine-grained* level, and can we quantify the *reliability* of performance predictions?

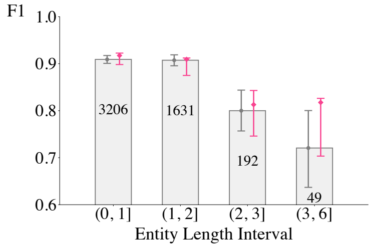


Figure 1: Breakdown of performance over different entity lengths of an NER system. *Actual F1* (gray point) is calculated from actual results while *predicted F1* (red point) is estimated by a performance prediction model. Gray and red lines represent corresponding confidence intervals. Numbers in each bar indicate the number of test samples in each length bucket.

With respect to the first contribution, previous P^2 methods have almost entirely focused on predicting *holistic* measures of accuracy such as entity F1 (Ratinov and Roth, 2009) or BLEU score (Papineni et al., 2002) over the entire dataset (§2.2). However, from a perspective of understanding the workings of our models, work on model analysis has demonstrated the need for more *fine-grained* analysis over a wide variety of tasks (Kummerfeld et al., 2012; Kummerfeld and Klein, 2013; Karpathy et al., 2015; Neubig et al., 2019; Fu et al., 2020a,b,c). These methods calculate separate accuracy scores for different types of examples (e.g. accuracies for entity recognition by entity length). Our first contribution is to examine experimental settings where we predict these fine-grained evaluation scores (§2.3), and also propose performance prediction methods particularly suited to this fine-grained evaluation setting (§3).

Our second contribution is the development of methods for estimating the reliability of performance predictions. While allowing estimation

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3703–3714
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

of experimental results without actually having to run the corresponding experiments may improve efficiency, if the performance predictor is wrong it may lead to missing results of a potentially important experiment. This particularly becomes an issue when developing methods for fine-grained performance prediction, as the number of data points which can be used to predict each performance number decreases as we subdivide datasets into finer-grained categories. Thus, we make methodological steps towards answering two specific questions: (i) how can we define and calculate a confidence interval over performance predictions? (ii) how well does the confidence interval of prediction performance calibrate with the true probability of an experimental result? Fig. 1 is an example of performance prediction and reliability analysis over fine-grained performance estimates (F1 scores over different entity length buckets) of an NER system are obtained in two ways: (i) calculated based on results from the NER system itself (in gray); (ii) estimated based on a performance prediction model, without running an actual experiment (in red). We can observe that: (1) with fewer test samples (e.g. `49`), confidence intervals of both actual and predicted F1 become much wider, suggesting larger uncertainty. (2) in the last bucket, the predicted F1 (red point) is far from the actual F1 (gray point), but with a confidence interval of predicted performance (red bar), the actual F1 still falls within it, indicating the importance of knowing the level of confidence.

In experiments, we investigate the efficacy of different performance prediction models on four typical NLP tasks under both holistic and fine-grained settings, then explore methods for the reliability analysis of these performance prediction models. Major experimental results show: 1) there is no one-size-fits-all model: best-scoring performance prediction systems in different scenarios are diverse. In particular, one of our proposed models achieved the best results on the Part-of-Speech task (§6.1). 2) a better performance prediction model doesn't imply better calibration (§6.2). 3) all four performance prediction models (including previous top-scoring ones) produce confidence intervals over-confidently (§6.2).

2 Performance Prediction: Formulation and Applicable Scenarios

In this section, we will mathematically define performance prediction and its application in the holistic and fine-grained evaluation.

2.1 Formulation

Given a machine learning model $\mathcal{M}$, which is trained over a training set $\mathcal{D}^{tr}$ based on a specific training strategy $\mathcal{S}$, we then test the dataset $\mathcal{D}^{ts}$ under evaluation setting $\mathcal{E}$ and the test result y can be formulated as a function of the following inputs:

$$y = f(\mathcal{M}, \mathcal{D}^{tr}, \mathcal{D}^{ts}, \mathcal{S}, \mathcal{E}), \qquad (1)$$

This we will refer to as the *actual* performance (e.g., F1 score), which requires us to run an actual experiment.

Alternatively, to calculate y, instead of performing a full training and evaluation cycle, one can directly estimate it by extracting features of $\mathcal{M}$, $\mathcal{D}^{tr}$, $\mathcal{D}^{ts}$, $\mathcal{S}$, and running them through a prediction function

$$\hat{y} = g(\Phi_{\mathcal{M}}, \Phi_{\mathcal{D}^{tr}}, \Phi_{\mathcal{D}^{ts}}, \Phi_{\mathcal{S}}, \mathcal{E}; \Theta), \qquad (2)$$

where $\Phi(\cdot)$ represents features of the input, and Θ denotes learnable parameters. We will refer to this as our *predicted* performance. As long as Eq. 2 is fast to calculate and a relatively accurate approximation of Eq. 1, it allows us to get a reasonable idea of expected experimental results much more efficiently than if we had to actually experiment.

In a real scenario, not all inputs in Eq. 2 need to be taken into account, and researchers can adopt different inputs for a particular use. For example, Domhan et al. (2015) define $\hat{y}$ as a function of training strategy $\mathcal{S}$ (e.g., different hyper-parameter settings) so that they can know which training setting can lead to bad performance without running. Dodge et al. (2020) estimate validation performance as a function of computation budget to conduct more robust model comparisons.

Why Performance Prediction matters for NLP tasks Firstly, for some NLP tasks with few resources, it is challenging to build and test systems for all languages or domains. For example, the task of Machine Translation (MT) for low resource languages is hard due to the lack of the large parallel corpora, preventing us from measuring system performance in these scenarios (Xia et al., 2019, 2020). Therefore, performance prediction is useful in that it can efficiently and comprehensively give insights about the workings of models over a wide variety of task settings. Secondly, performance prediction can be used to alleviate the data sparsity problem in fine-grained evaluation, which plays an important role in current NLP task evaluation (Fu et al., 2020a).

In this paper, we consider two performance prediction scenarios, a holistic evaluation setting that most previous works have explored, and a novel setting of predicting fine-grained evaluation metrics. Below, we briefly describe them.

2.2 Holistic Evaluation

Performance prediction in holistic evaluation aims to estimate an overall score (e.g., BLEU) based on dataset characteristics, specifically,

$$\hat{y} = g^{\text{holistic}}(\Phi_{\mathcal{D}^{tr}}, \Phi_{\mathcal{D}^{ts}}; \Theta), \tag{3}$$

where $\Phi(\cdot)$ represents features of input and Θ denotes learnable parameters.

Featurization In practice, we choose a machine translation (MT) task and a Part-of-Speech task (POS) task in this setting. We use the same set of dataset features as (Xia et al., 2020), including the language features and the source and the target, or transfer language.

2.3 Fine-grained Evaluation

In contrast, fine-grained evaluation aims to break down the overall score into different interpretable parts, allowing us to identify the strengths and weakness of learning systems. For example, the accuracy of an NER system with an overall F1 score 90 (%) can be partitioned into four buckets based on different entity lengths l (e.g., $[l = 1, 1 < l \leq 3, 3 < l \leq 5, l > 5]$) of test entities, thereby obtaining fine-grained F1 scores: $[93, 91, 89, 75]$, identifying that the model struggles on longer entities ($l > 5$).

Although fine-grained evaluation is advantageous in interpreting systems' performance, it frequently suffers from the data sparsity problem—a few or no test samples may be included within certain buckets. For example, in the above case it's difficult to calculate the F1 score for entities whose lengths satisfy $l > 7$ since few entities can be found in the whole test set.

With the above dilemma in mind, we define a performance prediction problem in fine-grained evaluation where the paucity of test samples in some buckets leads to an inability to compute performance accurately.

$$\hat{y} = g^{\text{fine}}(\Phi_M, \Phi_{\mathcal{D}^{tr}}, \Phi_{\mathcal{D}^{ts}}; \Theta), \tag{4}$$

where $\Phi(\cdot)$ represents features of input and Θ denotes learnable parameters.

Featurization Performing fine-grained evaluation involves two major steps: (i) partition the test set into different *buckets* based on a certain *aspect* (e.g., `entity length`), (ii) and calculate performance (e.g., `F1 score`) for each bucket. Therefore, data-wise ($\Phi_{\mathcal{D}^{ts}}$), the input of performance prediction function in Eq. 4 ($\mathrm{g}^{fine}(\cdot)$) can be featurized as different types of (i) buckets (ii) aspects (iii) datasets. Additionally, we take (iv) different types of models as input. We present brief descriptions of the above four types of features.

1. *Models*: We choose 12 models for the NER task and 8 models for the Chinese Word Segmentation (CWS) task. The models are built by choosing the different character encoder (e.g., ELMo (Peters et al., 2018) and Flair (Akbik et al., 2018; Akbik et al.)), word embedding (e.g., GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013b)), sentence-level encoder (e.g., LSTM (Hochreiter and Schmidhuber, 1997) and CNN (Kalchbrenner et al., 2014)), and decoder (e.g., MLP and CRF (Lample et al., 2016; Collobert et al., 2011)).
2. *Datasets*: We consider 6 (5) datasets for the NER (CWS) task, detailed in appendix.
3. *Attributes*: We consider the interpretable evaluation aspects proposed in works (Fu et al., 2020a). We consider 9 attributes for the NER task and 8 attributes for the CWS task in this paper (e.g, entity length and sentence length).
4. *Buckets*: The test entities (words) of the NER (CWS) task are partitioned into four buckets according to their attribute value. We compute the F1 score for the entities.

3 Parameterized Regression Functions

The performance prediction model takes in a set of features that characterize an experiment's peculiarities and predict performances based on different parameterized regressors $\mathrm{g}(\cdot)$ in Eq. 2. We first describe methods explored by previous works and then present a tensor regression-based approach that is particularly well-suited for fine-grained performance prediction.

3.1 Gradient Boosting Methods

Previous work on performance prediction has used gradient boosted decision tree models (Ganjisaf-

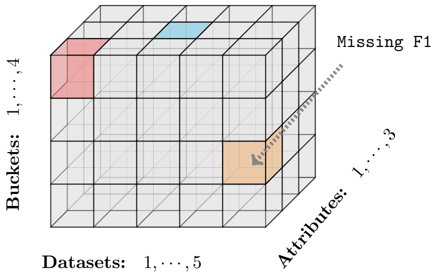


Figure 2: Illustration of performance tensor in the fine-grained evaluation scenario. Colored entries represent missing performances that would be predicted.

far et al., 2011; Chen and Guestrin, 2016), which demonstrate robust performance on the relatively low-data scenarios we often encounter in performance prediction tasks. We specifically explore the following two models:

XGBoost (Chen and Guestrin, 2016) is a tree boosting system widely used to solve problems such as ranking, classification, and regression. We use the same experimental setting as described in (Xia et al., 2020).

LightGBM (Ke et al., 2017) is a gradient boosting framework. Compared with XGBoost, which utilizes a level-wise tree growth in the decision tree, LightGBM uses a leaf-wise splitting method.

3.2 Tensor Regression

Besides gradient boosted trees, we also present tensor regression-based performance prediction models. Tensors are multidimensional arrays that can concisely depict the structure of the data. The order of a tensor is its number of dimensions. For example, in the NER task, the four feature dimensions of a tensor are model, dataset, attribute and bucket, with each slice representing one underlying relationship between the two dimensions. Applying tensor factorization algorithms in the performance prediction setting allows us to determine the interdependencies between multiple aspects of the tasks simultaneously.

Performance Prediction as Tensor Completion To formulate the performance prediction task as a tensor regression problem: (i) we first define a *performance tensor* that each entry stores a performance value under a specific setting determined by input features (described in §2.3); (ii) missing entries in performance tensor can be predicted based on different tensor completion techniques.

Specifically, taking fine-grained evaluation for example, we define a fine-grained performance tensor as $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$, where $\mathbf{Y}_{ijkt}$ denotes the performance (e.g. F1 score) of the i-th model (e.g. `BERT-based Tagger`) on the j-th bucket (e.g. 2_{nd}) that is obtained by partitioning the k-th dataset (e.g. `CoNLL03`) based on the t-th attribute (e.g. `entity length`). I_1, I_2, I_3, I_4 denote the number of models, buckets, datasets, attributes. Fig. 2 elaborates on this, in which three dimensions (buckets, datasets, and attributes) are considered for the sake of presentation.

CP Decomposition The CP decomposition (Hitchcock, 1927) expresses a tensor $\mathbf{Y}$ as a sum of lower rank tensors. For example, an order 4 tensor can be decomposed as the sum of R rank-1 tensors, each being the outer product of four vectors in each dimension.

Robust PCA Robust PCA is a modification of principal component analysis (PCA) (Candès et al., 2009). If a tensor can be conceived as a superposition of low-rank components and a sparse component, Robust PCA attempts to recover the low-rank and sparse components. The sparse components can be considered as the gross, but sparse noise in the dataset.

4 Statistical Preliminaries

Before going into our second contribution to establishing reliability of performance prediction, we describe two relevant concepts from statistics.

4.1 Confidence Interval (CI)

The confidence interval (CI) is a range of possible values for an unknown parameter associated with a *confidence level* of γ (Nakagawa and Cuthill, 2007; Dror et al., 2018) that the actual parameter can fall into the suggested range. Specifically, suppose that we are interested in estimating the underlying true parameter of ω. Given an observed parameter estimate of $\hat{\omega}$, obtained from the data, we aim to compute an interval with a confidence level γ that ω lies in an interval CI.

Commonly, there are two approaches to calculate confidence intervals, depending on our knowledge about the distribution of the statistics of interest. When an analytical form exists and we have reasonable assumptions on the distribution, we can employ the normal theory or use Student's t-distribution to construct a confidence interval.

Regarding data drawn from a completely unknown distribution, a CI can be calculated by a bootstrapping method (Efron, 1992; Johnson, 2001). The main idea behind the bootstrapping method is to simulate the real distribution by sampling with replacement from a distribution that approximates it, thereby allowing us to make inferences about the statistics of interest and construct confidence intervals. Common methods to construct the CI with bootstrap include the percentile method, where after specifying a confidence level γ, we take the range of points that cover the middle γ proportion of bootstrap sampling distribution $\hat{Y}$ as the desired confidence interval, represented by $(Q_{\hat{Y}((1-\gamma)/2)}, Q_{\hat{Y}((1+\gamma)/2)})$, where Q denotes the quantile. Works on establishing confidence for results in NLP tasks using this bootstrap method include Koehn (2004) and Li et al. (2017).

4.2 Model Calibration (MC)

Calibration (Gleser, 1996), also known as reliability, refers to the ability of a model to make good probabilistic predictions. For a discrete distribution over events, a model is said to be well-calibrated if for those events that the model assigns a probability of p, the long-run proportion that the event actually occurs turns out to be p. For example, if a weather forecast model predicts that there is a 0.1 probability of rain at 7 a.m., then when observed on a large number of random trials at 7 a.m., the model is well-calibrated if 0.1 of them actually do result in rain. Similarly, for a classification model matching the probability a model assigns to a predicted label (i.e., *confidence*) and the correctness measure of the prediction (i.e., *accuracy*) (Wang et al., 2020) is desired.

Nonetheless, it is common that a model could have a high predictive accuracy, but poor calibration if the model systematically over- or under-estimates its confidence in the predictions it makes. One way to quantify miscalibration is to use *Expected Calibration Error* (ECE; Nae (2015)), which aims to quantitatively characterize the difference in expectation between confidence and accuracy. To calculate ECE, the predictions should first be partitioned into M buckets based on the confidence of the predictions, where N represents the total number of prediction samples and $|B_m|$ is the number of samples in the m-th bucket. Given these buckets, ECE can be defined as,

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} |\mathrm{acc}(B_m) - \mathrm{conf}(B_m)|, \quad (5)$$

where $\mathrm{acc}(B_m)$ denotes the accuracy of B_m,

$$\mathrm{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}(\hat{y}_i = y_i), \quad (6)$$

where $\hat{y}$ and y represent predicted and ground truth labels respectively. $\mathrm{conf}(B_m)$ represents the average confidence of bucket B_m,

$$\mathrm{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i. \quad (7)$$

where $\hat{p}$ represents the prediction confidence of sample i.

5 On Reliability of P^2 Models

Now we discuss our methodology for predicting the reliability of performance prediction models through confidence intervals and the calibration of those confidence intervals.

5.1 CIs of Predicted Performance

We refer to $y \sim Y$ as an actual observed performance as in Eq. 1 for a specific task (e.g., NER). y is the output of an NLP system learned on a dataset $\mathcal{D} = (\mathcal{D}^{tr}, \mathcal{D}^{ts})$. We refer to $\hat{y} \sim \hat{Y}$ as a predicted performance estimated as Eq. 2. $\hat{y}$ is the output of a performance prediction model learned from a dataset $\Phi(\mathcal{D}) = (\Phi(\mathcal{D}^{tr}), \Phi(\mathcal{D}^{ts}))$, where $\Phi(\cdot)$ represents the input dataset features. Our goal is to compute a confidence interval w.r.t a predicted performance $\hat{y}$, to make inference about Y.

Bootstrap for CI of Predicted Performance One potential challenge is that we cannot make plausible assumptions about the distribution of predicted performances $\hat{Y}$, which prevents us from using popular parametric methods (as mentioned in § 4.1) to calculate the confidence interval. Instead, we resort to a bootstrap resampling method as adopted in (Efron, 1992), to simulate $\hat{Y}$.

To achieve this, we first (i) sample different training sets for the performance prediction model $\Phi(\mathcal{D})_1^{tr}, \Phi(\mathcal{D})_2^{tr}, \cdots, \Phi(\mathcal{D})_K^{tr} \sim \Phi(\mathcal{D})^{tr}$, and then (ii) train K performance prediction models using Eq. 2 on each of the K partitions, and (iii) evaluate K models on $\Phi(\mathcal{D})^{ts}$, thereby obtaining a prediction distribution $\hat{Y}$. From this resampling distribution, we use the percentile method, taking the top $(1-\gamma)/2$ and the bottom $(1+\gamma)/2$ of the distribution as higher and lower bounds for the confidence interval.

5.2 Calibration of CI

Because we calculate confidence intervals of the predicted performance $\hat{y}$, drawn from the distribution of $\hat{Y}$, rather than the actual y from Y, it's still unclear if our predicted CI is reliable enough to cover the actual performance. In other words, "from an infinite number of independent trials, does the true value *actually* lie within the intervals approximately 95% of the times?"

To answer this question, we establish a method to measure calibration for the confidence interval of predicted performance. To check (i) if y could be generally contained in the prediction intervals reasonably well, and (ii) if a prediction model produces predictions that are not over or under-confident, we empirically examine the prediction distributions and establish the reliability of the confidence intervals.

To this end, we extend the definition of calibration in classification setting to our regression problem. Specifically, we formulate confidence level γ as prediction confidence conf defined in Eq. 7, and then the original definition of different M buckets can be instantiated as different confidence levels here: $\gamma_1, \cdots, \gamma_M$. The *accuracy* at each confidence level γ_b defined as follows:

$$\mathrm{acc}(\gamma_b) = \frac{\sum_{i=1}^{N} \mathbb{1}(A < y_i < B)}{N}, \qquad (8)$$

where $i \in [1, N]$, $b \in [1, M]$. N represents the number of test samples. y_i denotes the actual performance for the test sample i. $A = (Q_{\hat{Y}((1-\gamma_b)/2)}$ and $B = Q_{\hat{Y}((1+\gamma_b)/2)})$.

Intuitively, $\mathrm{acc}(\gamma_b)$ represents the relative frequency of the actual value y falling into the predicted confidence interval w.r.t. $\hat{y}$. Fig. 3 illustrates how $\mathrm{acc}(\gamma_b)$ is calculated: given three samples whose performances are to be predicted, the denominator of $\mathrm{acc}(\gamma_b = 0.8)$ is 3 while the numerator tallies how many times (2 in this case) the actual performances (i.e., y_1, y_2, y_3) of three samples fall into the confidence interval (with $\gamma_b = 0.8$) of corresponding bootstrapped distributions.

Based on Eq. 8, we can re-write a calibration error CE as:

$$\begin{aligned} \mathrm{CE} &= \sum_{b=1}^{M} \mid \mathrm{acc}(\gamma_b) - \gamma_b \mid & (9) \\ &= \sum_{b=1}^{M} \mid \frac{\sum_{i=1}^{N} \mathbb{1}(A < y_i < B)}{N} - \gamma_b \mid & (10) \end{aligned}$$

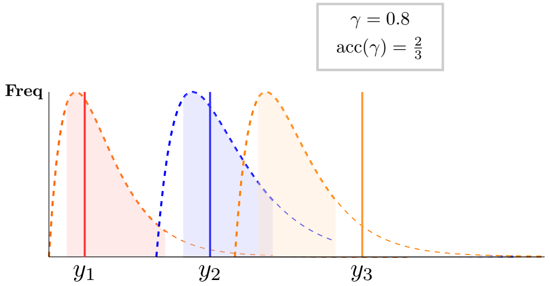


Figure 3: Illustration of calibration on confidence intervals ($\gamma_b = 0.8$) w.r.t. the predicted performance. Solid lines represent actual performances (y) while dashed lines denote resampled predicted performances using bootstrap. Boundaries of shaded areas indicate confidence interval $(Q_{\hat{Y}((1-\gamma_b)/2)}, Q_{\hat{Y}((1+\gamma_b)/2)})$. Intuitively, y_1 and y_2 fall into confidence intervals of corresponding bootstrapped distributions $\hat{Y}_1$ and $\hat{Y}_2$.

6 Experiments

In this section, we break down our experimental results into answering two research questions sections: (1) how well do our underlying performance predictors work, particularly the newly proposed tensor-based predictors and on the newly proposed task of fine-grained performance prediction? (2) how well can we estimate the reliability of our performance predictions?

Models Besides the four performance prediction models (CP, PCA, XGBoost, LGBM) that we have introduced in §3, following Xia et al. (2020), we additionally use a simple mean value baseline model which predicts an average of scores s from the training folds for all test entries in the left-out evaluation fold:

$$\hat{s}^{(i)}_{\text{mean}} = \frac{1}{|\mathcal{D} \setminus \mathcal{D}^{(i)}|} \sum_{s \in \mathcal{D} \setminus \mathcal{D}^{(i)}} s; i \in 1, ...k, \qquad (11)$$

where $\mathcal{D}^{(i)}$ is the left-out data used to evaluate the model performance.

Hyper-parameters Detailed information about the hyper-parameters used in training the performance prediction models in various tasks is provided in the appendix.

Tasks We explore performance prediction on four tasks: (1) Machine Translation (MT) (Schwenk et al., 2019), (2) Part-of-Speech tagging (POS), (3) Named Entity Recognition (NER), (4) Chinese Word Segmentation (CWS). To compare the performance of tensor-based models and gradient boosting models on the same dataset, we convert the datasets used in different prediction tasks to tensors. Statistics of the tensor data are shown in the appendix.

6.1 Evaluation of Performance Prediction

Setup To investigate the effectiveness of the performance prediction models across different tasks, we conduct k-fold cross-validation for evaluation. Specifically, we randomly partition the entire experimental data $\mathcal{D}$ into $k = 5$ folds, use 4 folds for training, and test the model's performance on the remaining fold. To evaluate the result, we calculate the average root mean square error (RMSE) between the predicted scores and the true scores.

Results The RMSE scores of different performance tasks are shown in Tab. 1. Notably, RMSE scores across different tasks should not be compared directly, because the scales of the evaluation metrics are different. We observed that:

(1) Overall, all four models we investigated outperform the baseline by a large margin, indicating their effectiveness on these four performance prediction tasks. (2) Comparing two tensor-based models, PCA consistently outperforms CP. Notably, our proposed tensor regression model (PCA) has surpassed the previous best-performing system (XGBoost (Xia et al., 2020)) on the `POS` dataset and achieved comparable result on the `MT` dataset despite the relatively high sparsity of the tensor (0.346). (3) We observe that CP achieves much worse performance on the `POS` dataset. One potential reason is that: CP is sensitive to datasets (like `POS`) that exhibit large variance along some feature dimensions, which can not be alleviated by feature scaling. (4) There is no one-size-fits-all model: on different datasets, the corresponding best-scoring performance prediction models are diverse, suggesting that we should take dataset's characteristics into account when selecting a model for a specific performance prediction scenario.

Prediction Error Analysis In §1 and Fig. 1, we reveal how entities with different lengths influence the performance prediction, a result of the underlying paucity of data. Here we perform a more detailed error analysis to understand the factors that influence the performance of performance prediction models. Specifically, we perform a case study on the `NER` task using XGBoost and look for feature combinations on which performance predictions show poor results. We use XGBoost to predict F1 scores on all possible combinations of four feature dimensions (models, datasets, attributes, and buckets) to obtain $\hat{y}_{ijkt}$ using the combined test sets from 5-fold cross-validation. For each prediction, we calculate a square residual $(\hat{y}-y)^2$. Then, we group the square residuals by 2 of the 4 dimensions[1] and take their mean value aggregated over the other 2 dimensions. Fig. 4 shows the aggregated mean square residual (MSR) fixed on the model and dataset dimensions, and Fig. 5 shows the result fixed on the attribute and bucket dimensions. In both figures, a high MSR (dark grid) means a poor performance prediction. In Fig. 4, we notice that (1) dataset-wise: `WB` and `WNUT`, and (2) model-wise: *CcnnWgloveLstmMlp* and *CnoneWrandLstmCrf* show poor results. We observe that (1) `WB` is generated from weblogs and `WNUT` is generated from Twitter, both of which are noisy. (2) *CcnnWgloveLstmMlp* does not use a CRF-decoder, and *CnoneWrandLstmCrf* does not encode character-level features, both of which are important characteristics in building an `NER` model. It is plausible that the systems have an unstable performance in those experimental settings and thus make them harder to predict. In Fig. 5, we notice that (1) a lower bucket value along the attributes *entity consistency*, *token consistency*, and *entity density*, (2) a higher bucket value along the attributes *token frequency* or *entity length* lead to poor performance prediction results. In other words, the performance prediction model finds it hard to predict when there is a low label consistency of token or entity, a low entity density, and when token frequency is high and entity is long.

Model	Fine-grained		Holistic	
	NER	**CWS**	**MT**	**POS**
Baseline	0.209	0.137	6.388	29.09
XGBoost	**0.055**	**0.021**	2.463	7.319
LGBM	0.059	0.041	**2.389**	7.673
CP	0.068	0.043	4.065	24.70
PCA	0.057	0.029	2.920	**5.860**

Table 1: Results (RMSE, lower scores indicate better performances) of different performance prediction models on four tasks. The lowest value of each column is bold.

6.2 Evaluation of Reliability

Setup As described in §5.1, we use non-parametric bootstrap to produce confidence intervals for $\hat{y}$. For the holistic evaluation setting, we

---

[1] Readers can refer to this work (Fu et al., 2020a) to get more details.

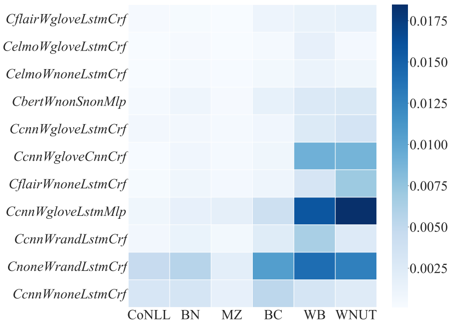


Figure 4: Each grid in the heatmap denotes the mean square residual fixed on the corresponding model (y-axis) and dataset (x-axis) aggregated over all attributes and buckets. The colorbar on the right denotes the value of the mean square residual. Readers can refer to this work (Fu et al., 2020a) to get more details about the information of models and attributes.

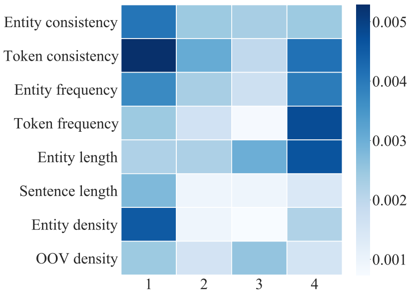


Figure 5: Each grid denotes the mean square residual fixed on the corresponding attribute (y-axis) and bucket (x-axis) aggregated over all models and datasets. Buckets 1 to 4 are ordered in increasing attribute value.

do not include tensor-based models since the property, "*with replacement*", of the bootstrap makes it difficult to construct resampled tensors in the holistic evaluation setting.

When calculating a calibration error as defined in Eq. 10, we set $M = 20$, choosing a range of 20 increasing confidence levels, (i.e. $\gamma_1 = 0.05, \gamma_2 = 0.10 \cdots, \gamma_{20} = 1.00$), to evaluate the correctness of the confidence intervals given by prediction models. Besides using a reliability diagram and a calibration error, to compare the calibration performances of different models more comprehensively, we additionally use the following quantitative metrics: (1) *average width* is the mean range of all the prediction distributions, formally, the difference between the maximum and minimum: $\frac{1}{N}\sum_{i\in[1,N]} \max(\hat{Y}_i) - \min(\hat{Y}_i)$. (2) *coverage* is the value of acc_b evaluated at $\gamma = 1$. (i.e. proportion of y $\hat{y}$ that fall into the distributions $\hat{Y}$, out of all the N prediction entries).

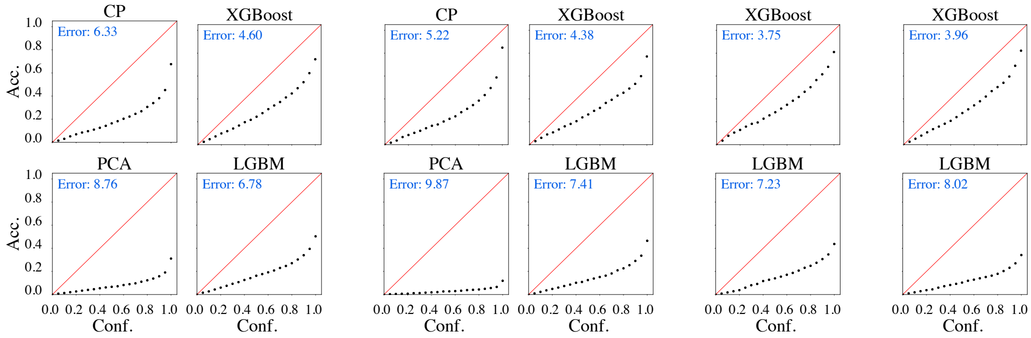
Results The reliability diagram of different models and their corresponding metrics on four tasks are illustrated in Fig. 6 and Tab. 2 respectively. Intuitively, the smaller the CE (calibration error) value is, the more closely the black dotted line becomes diagonal. Ideally, a perfectly calibrated model should have a CE of 0.

Model	CE	Wid.	Cov.	CE	Wid.	Cov.
	NER			CWS		
Baseline	10.47	0.006	0.01	10.50	0.003	0.01
XGBoost	4.60	0.093	0.75	4.38	0.045	0.77
LGBM	6.78	0.093	0.50	7.41	0.029	0.46
CP	6.33	0.110	0.68	5.22	0.099	0.85
PCA	8.76	0.051	0.31	9.87	0.015	0.12
Model	**MT**			**POS**		
Baseline	9.98	1.46	0.08	10.30	3.96	0.03
XGBoost	3.75	5.55	0.81	3.96	17.61	0.82
LGBM	7.23	3.01	0.44	8.02	12.90	0.34

Table 2: Calibration errors (CE), average width (Wid.), coverage (Cov.) of different models over four tasks. (`NER`, `CWS` for fine-grained evaluation setting and `MT`, `POS` for holistic evaluation setting)

From these two tables, we see that: (1) Overall, in both holistic (`MT` and `POS`) and fine-grained settings (`NER` and `CWS`), we see that XGBoost achieves the lowest calibration error together with a higher coverage, especially in the holistic setting. (2) We observe that all of the plots indicate that the intervals produced by the models are over-confident, as the dots lie under the identity function. In other words, given a confidence level γ, the actual accuracy is lower than γ. (3) In Tab. 1, we find that LGBM achieves the lowest RMSE (2.389) in task MT, but its calibration error (7.23) is worse than XGBoost (3.75), implying that a model that predicts accurately is not necessarily well calibrated. This could be explained by the observation that the predicted distribution $\hat{Y}$ of LGBM has a narrower width (3.01). Given a large number of trials predicted by LGBM, we cannot be confident that the true y is contained in the range of values predicted.

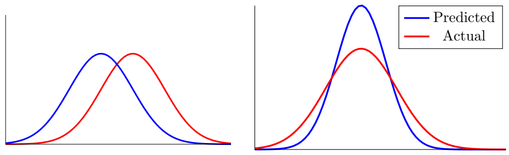
Case Analysis To get a better understanding of how calibration analysis is conducted on different performance models, we perform a case study on `NER` task. Fig. 7(a-b) illustrates two plots that artificially simulate two common relations following Diebold et al. (1997) between actual and predicted distribution: (i) *Bias* (ii) *Over-confidence*. From Tab. 2, we see that XGBoost is better calibrated than LGBM in NER task. To interpret this gap, we (i) first randomly select test samples from NER dataset and then (ii) use two performance prediction models XGBoost and LGBM to produce blue



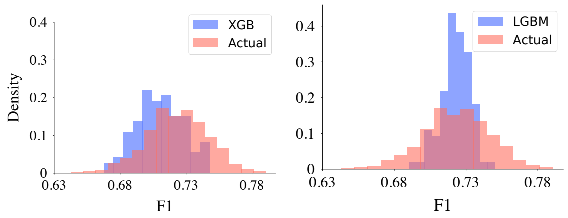
(a) NER (b) CWS (c) MT (d) POS

Figure 6: Calibration of different performance prediction models on four tasks (`NER`, `CWS` for fine-grained evaluation setting and `MT`, `POS` for holistic evaluation setting).

distributions in Fig. 7(c-d) using the bootstrap (as §5.1). A perfectly calibrated model will show a histogram shape that resembles the actual one. We can see that the histogram shape of (d) signifies an over-confidence problem, in which the predicted distribution (in blue) is covered by the actual distribution (in red). By contrast, in (c) the histogram of XGBoost in blue shifts to the left compared with the actual observed distribution, indicating that the prediction on this bucket is biased.



(a) Bias (b) Over-Confident



(c) Bias (d) Over-Confident

Figure 7: The first row of two plots (a,b) artificially simulate two typical relations between actual and predicted distributions. The second row of two plots (c,d) show two real-world distributions of predicted performance w.r.t one test sample from NER task against corresponding actual distributions.

7 Implications and Future Directions

In this work, we not only widen the applicability of performance prediction, extending it to fine-grained evaluation scenarios, but also establish a set of reliability analysis mechanisms to improve its practicality. In closing, we highlight some potential future directions:

Confidence over confidence: Our work provides an idea for reliability analysis of the predicted confidence interval, which could also be explored on other scenarios, e.g., density forecasting (Diebold et al., 1997). Another potentially valuable research topic is to build connections with the probability integral transform (Angus, 1994), which is a typical method of calibration evaluation in financial risks, and our proposed calibration method.

Calibration for automated evaluation metrics: From a broader point of view, the role of existing learnable automatic evaluation metrics for text generation, such as BLEURT (Sellam et al., 2020) and COMET (Rei et al., 2020), is similar to a performance prediction model (i.e., both take features of input data as input and then output an evaluation score). Reliability analysis of these metrics is also an important topic since they determine the direction of model optimization.

Acknowledgements

We sincerely thank all reviewers for their insightful comments and suggestions. We also thank Mengzhou Xia for discussions on details of performance prediction for different NLP tasks. This work was supported in part by a grant under the Northrop Grumman SOTERIA project and the Air Force Research Laboratory under agreement number FA8750-19-2-0200. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

2015. Obtaining well calibrated probabilities using Bayesian Binning. *Proceedings of the National Conference on Artificial Intelligence*, 4:2901–2907.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. Pooled contextualized embeddings for named entity recognition.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649.

John E Angus. 1994. The probability integral transform and related results. *SIAM review*, 36(4):652–654.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2008. Predicting success in machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 745–754, Honolulu, Hawaii. Association for Computational Linguistics.

Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2009. Robust principal component analysis? *CoRR*, abs/0912.3599.

Hui Chen, Zijia Lin, Guiguang Ding, Jianguang Lou, Yusen Zhang, and Borje Karlsson. 2019. Grn: Gated relation network to enhance convolutional neural network for named entity recognition. 33(01):6236–6243.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Francis X Diebold, Todd A Gunther, and Anthony S Tay. 1997. Evaluating Density Forecasts. *International Economic Review*, 39:863–883.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2020. Show your work: Improved reporting of experimental results. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, (2):2185–2194.

Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. 2015. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Bradley Efron. 1992. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer.

Jinlan Fu, Pengfei Liu, and Graham Neubig. 2020a. Interpretable multi-dataset evaluation for named entity recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jinlan Fu, Pengfei Liu, Qi Zhang, and Xuan-Jing Huang. 2020b. Is chinese word segmentation a solved task? rethinking neural chinese word segmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5676–5686.

Jinlan Fu, Pengfei Liu, Qi Zhang, and Xuanjing Huang. 2020c. Rethinking generalization of neural models: A named entity recognition case study. In *AAAI*, pages 7732–7739.

Yasser Ganjisaffar, Rich Caruana, and Cristina Videira Lopes. 2011. Bagging gradient-boosted trees for high precision, low variance ranking models. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 85–94.

Leon Jay Gleser. 1996. Measurement, regression, and calibration.

Frank L. Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Roger W Johnson. 2001. An introduction to the bootstrap. *Teaching statistics*, 23(2):49–54.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of ACL*.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 2017-December(Nips):3147–3155.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 4:388–395.

Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. 2012. Prediction of learning curves in machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22–30, Jeju Island, Korea. Association for Computational Linguistics.

Jonathan K. Kummerfeld, David Hall, James R. Curran, and Dan Klein. 2012. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1048–1059, Jeju Island, Korea. Association for Computational Linguistics.

Jonathan K. Kummerfeld and Dan Klein. 2013. Error-driven analysis of challenges in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 265–277, Seattle, Washington, USA. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Shinichi Nakagawa and Innes C Cuthill. 2007. Effect size, confidence interval and statistical significance: a practical guide for biologists. *Biological reviews*, 82(4):591–605.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. *arXiv preprint arXiv:1903.07926*.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of NAACL*, volume 1, pages 2227–2237.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. Comet: A neural framework for mt evaluation. *arXiv preprint arXiv:2009.09025*.

Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. 2019. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *CoRR*, abs/1907.05791.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Marco Turchi, Tijl De Bie, and Nello Cristianini. 2008. Learning performance of a machine translation system: a statistical and computational analysis. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 35–43.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079, Online. Association for Computational Linguistics.

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8625–8646, Online. Association for Computational Linguistics.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized data augmentation for low-resource translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5786–5796, Florence, Italy. Association for Computational Linguistics.

A Datasets

Named Entity Recognition (NER) We choose two well-established benchmark datasets: CoNLL-2003 [2] and OntoNotes 5.0. [3] CoNLL-2003 is drawn from Reuters news. OntoNotes 5.0 is collected from newsgroups (`NW`), broadcast news (`BN`), broadcast conversation (`BC`), weblogs (`WB`), magazine genre (`MZ`), and telephone speech (`TC`), in which the first five genres of text are used in this paper.

Chinese Word Segmentation (CWS) We consider five mainstream datasets: `CKIP`, `CTB`, `MSR`, `NCC`, and `SXU`, from SIGHAN2005 [4] and SIGHAN2008 [5]. The traditional Chinese characters in `CKIP` are mapped to simplified Chinese characters in our experiment.

B Models

Our NER (CWS) models can be decomposed into four aspects: 1) character/subword encoders; 2) word (bigram) embeddings; 3) sentence-level encoders; 4) decoders.

The four aspects of NER can be summarized as: 1) character/subword encoder: `ELMo` (Peters et al., 2018), `Flair` (Akbik et al., 2018; Akbik et al.), `BERT` [6] (Peters et al., 2018; Devlin et al., 2018); 2) additional word embeddings: `GloVe` (Pennington et al., 2014); 3) sentence-level encoders: `LSTM` (Hochreiter and Schmidhuber, 1997), `CNN` (Kalchbrenner et al., 2014; Chen et al., 2019); 4) decoders: `MLP` or `CRF` (Lample et al., 2016; Collobert et al., 2011).

The four aspects' setting of CWS: 1) character/subword encoder: `ELMo`, `BERT`; 2) bigram embeddings: `Word2Vec` (Mikolov et al., 2013a), averaging the embedding of two contiguous characters; the settings of 3) the sentence-level encoders and 4) decoders are equal to NER.

Feature	NER	CWS	MT	POS
Sparisity	0.0	0.0	0.346	0.019
Shape	(11,6,9,4)	(5,8,8,4)	(39,39,22)	(26,60,14)

Table 3: Statistics of performance tensors for four tasks. *Sparsity* denotes the percentage of missing values in the tensor

We can also do bootstrap for predictions using regression models. If we consider recovering missing data with CP decomposition as a prediction method, we can construct a CI on the predicted values too.

C Hyper-parameters

For XGBoost, we use squared error as the objective function for regression and set the learning rate as 0.1. We allow the maximum tree depth to be 10, the number of trees to be 100, and use the default regularization terms to prevent the model from overfitting. For LGBM, we set the objective as regression for LGBMRegressor, the number of boosted trees and maximum tree leaves to be 100, adopt a learning rate of 0.1, and use the default regularization terms. For the Robust PCA model, we scale all the datasets, adopt the default regularization parameter of 1 for both the low rank and the sparse tensor, and set the learning rate as 1.1. For CP Decomposition, we do not standardize the features in CWS and NER, but do so for WMT and POS. We adopt a rank $r = 5$ in training and performance prediction, expressing the recovered tensor used for prediction to be a sum of 5 rank-1 tensors.

Statistics of Tensor Tab. 3, where *sparsity* denotes the percentage of missing values in the tensor.

---

[2] https://www.clips.uantwerpen.be/conll2003/ner/

[3] https://catalog.ldc.upenn.edu/LDC2013T19

[4] http://sighan.cs.uchicago.edu/bakeoff2005/

[5] https://www.aclweb.org/mirror/ijcnlp08/sighan6/chinesebakeoff.htm

[6] BERT is grouped into the subword encoder because we use it to obtain the representation of subwords.

Metrical Tagging in the Wild:
Building and Annotating Poetry Corpora with Rhythmic Features

Thomas Haider
Department of Language and Literature,
Max Planck Institute for Empirical Aesthetics, Frankfurt am Main
Institute for Natural Language Processing (IMS),
University of Stuttgart
`thomas.haider@ae.mpg.de`

Abstract

A prerequisite for the computational study of literature is the availability of properly digitized texts, ideally with reliable meta-data and ground-truth annotation. Poetry corpora do exist for a number of languages, but larger collections lack consistency and are encoded in various standards, while annotated corpora are typically constrained to a particular genre and/or were designed for the analysis of certain linguistic features (like rhyme). In this work, we provide large poetry corpora for English and German, and annotate prosodic features in smaller corpora to train corpus driven neural models that enable robust large scale analysis.

We show that BiLSTM-CRF models with syllable embeddings outperform a CRF baseline and different BERT-based approaches. In a multi-task setup, particular beneficial task relations illustrate the inter-dependence of poetic features. A model learns foot boundaries better when jointly predicting syllable stress, aesthetic emotions and verse measures benefit from each other, and we find that caesuras are quite dependent on syntax and also integral to shaping the overall measure of the line.

1 Introduction

Metrical verse, lyric as well as epic, was already common in preliterate cultures (Beissinger, 2012), and to this day the majority of poetry across the world is drafted in verse (Fabb and Halle, 2008). In order to reconstruct such oral traditions, literary scholars mainly study textual resources (rather than audio). The rhythmical analysis of poetic verse is still widely carried out by example- and theory-driven manual annotation of experts, through so-called close reading (Carper and Attridge, 2020; Kiparsky, 2020; Attridge, 2014; Menninghaus et al., 2017). Fortunately, well-defined constraints and the regularity of metrically bound language aid the prosodic interpretation of poetry.

However, for projects that work with larger text corpora, close reading and extensive manual annotation are neither practical nor affordable. While the speech processing community explores end-to-end methods to detect and control the overall personal and emotional aspects of speech, including fine-grained features like pitch, tone, speech rate, cadence, and accent (Valle et al., 2020), applied linguists and digital humanists still rely on rule-based tools (Plecháč, 2020; Anttila and Heuser, 2016; Kraxenberger and Menninghaus, 2016), some with limited generality (Navarro-Colorado, 2018; Navarro et al., 2016), or without proper evaluation (Bobenhausen, 2011). Other approaches to computational prosody make use of lexical resources with stress annotation, such as the CMU dictionary (Hopkins and Kiela, 2017; Ghazvininejad et al., 2016), are based on words in prose rather than syllables in poetry (Talman et al., 2019; Nenkova et al., 2007), are in need of an aligned audio signal (Rosenberg, 2010; Rösiger and Riester, 2015), or only model narrow domains such as iambic pentameter (Greene et al., 2010; Hopkins and Kiela, 2017; Lau et al., 2018) or Middle High German (Estes and Hench, 2016).

To overcome the limitations of these approaches, we propose corpus driven neural models that model the prosodic features of syllables, and to evaluate against rhythmically diverse data, not only on syllable level, but also on line level. Additionally, even though practically every culture has a rich heritage of poetic writing, large comprehensive collections of poetry are rare. We present in this work datasets of annotated verse for a varied sample of around 7000 lines for German and English. Moreover, we collect and automatically annotate large poetry corpora for both languages to advance computational work on literature and rhythm. This may include the analysis and generation of poetry, but also more general work on prosody, or even speech synthesis.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3715–3725
April 19 - 23, 2021.

Our main contributions are:

1. The collection and standardization of heterogenous text sources that span writing of the last 400 years for both English and German, together comprising over 5 million lines of poetry.
2. The annotation of prosodic features in a diverse sample of smaller corpora, including metrical and rhythmical features and the development of regular expressions to determine verse measure labels.
3. The development of preprocessing tools and sequence tagging models to jointly learn our annotations in a multi-task setup, highlighting the relationships of poetic features.

2 Manual Annotation

We annotate prosodic features in two small poetry corpora that were previously collected and annotated for aesthetic emotions by Haider et al. (2020). Both corpora cover a time period from around 1600 to 1930 CE, thus encompassing public domain literature from the modern period. The English corpus contains 64 poems with 1212 lines. The German corpus, after removing poems that do not permit a metrical analysis, contains 153 poems with 3489 lines in total. Both corpora are annotated with some metadata such as the title of a poem and the name and dates of birth and death of its author. The German corpus further contains annotation on the year of publication and literary periods.

Figure 1 illustrates our annotation layers with three fairly common ways in which poetic lines can be arranged in modern English. A poetic line is also typically called *verse*, from Lat. *versus*, originally meaning to turn a plow at the ends of successive furrows, which, by analogy, suggests lines of writing (Steele, 2012).

In this work, we manually annotate the sequence of syllables for metrical (meter, met) prominence (+/−), including a grouping of recurring metrical patterns, i.e., foot boundaries (|). We also operationalize a more natural speech rhythm (rhy) by annotating pauses in speech, caesuras (:), that segment the verse into rhythmic groups, and in these groups we assign main accents (2), side accents (1) and null accents (0). In addition, we develop a set of regular expressions that derive the verse measure (msr) of a line from its raw metrical annotation.

In English or German, the rhythm of a linguistic utterance is basically determined by the sequence of syllable-related accent values (associated with pitch, duration and volume/loudness values) resulting from the 'natural' pronunciation of a line, sentence or text by a competent speaker who takes into account the learned inherent word accents as well as syntax- and discourse-driven accents. Thus, lexical material comes with n-ary degrees of stress, depending on morphological, syntactic, and information structural context. The prominence (or stress) of a syllable is thereby dependent on other syllables in its vicinity, such that a syllable is pronounced relatively louder, higher pitched, or longer than its adjacent syllable.

```
msr iambic.pentameter
met −  + | −   + | −   + |  −  +| −  + |
rhy 0  1   0   0   0   2 :  0  1  0  2  :
   My love is like to ice, and I to fire:

msr iambic.tetrameter
met  −   + |−   + | −   + | −   +  |
rhy  0   1  0   2   0   0   0   1  :
    The winter evening settles down

msr trochaic.tetrameter
met  +    − | +   − | +  −  | +
rhy  0    0   1   0   1  0    2  :
    Walk the deck my Captain lies,
```

Figure 1: Examples of rhythmically annotated poetic lines, with meter (+/−), feet (|), main accents (2,1,0), caesuras (:), and verse measures (msr). Authors: Edmund Spenser, T.S. Eliot, and Walt Whitman.

2.1 Annotation Workflow

Prosodic annotation allows for a certain amount of freedom of interpretation and (contextual) ambiguity, where several interpretations can be equally plausible. The eventual quality of annotated data can rest on a multitude of factors, such as the extent of training of annotators, the annotation environment, the choice of categories to annotate, and the personal preference of subjects (Mo et al., 2008; Kakouros et al., 2016).

Three university students of linguistics/literature were involved in the manual annotation process. They annotated by silent reading of the poetry, largely following an intuitive notion of speech rhythm, as was the mode of operation in related work (Estes and Hench, 2016). The annotators additionally incorporated philological knowledge to recognize instances of poetic license, i.e., knowing how the piece is supposed to be read. Especially the annotation accuracy of metrical syllable stress and foot boundaries benefited from recognizing the schematic consistency of repeated verse measures, license through rhyme, or particular stanza forms.

2.2 Annotation Layers

In this paper, we incorporate both a linguistic-systematic and a historically-intentional analysis (Mellmann, 2007), aiming at a systematic linguistic description of the prosodic features of poetic texts, but also using labels that are borrowed from historically grown traditions to describe certain forms or patterns (such as verse measure labels).

We evaluated our annotation by calculating Cohen's Kappa between annotators. To capture different granularities of correctness, we calculated agreement on syllable level (accent/stress), between syllables (for foot or caesura), and on full lines (whether the entire line sequence is correct given a certain feature).

Main Accents & Caesuras: Caesuras are pauses in speech. While a caesura at the end of a line is the norm (to pause at the line break) there are often natural pauses in the middle of a line. In few cases the line might also run on without a pause. As can be seen in Figure 1, punctuation is a good signal for caesuras. Caesuras (csr) are denoted with a colon. We operationalize rhythm by annotating three degrees of syllable stress, where the verse is first segmented into rhythmic groups by annotating caesuras, and in these groups we assign main accents (2), side accents (1) and null accents (0).

	Syllable		Whole Line	
	m.ac	caesura	m.ac	caesura
DE$_{blind}$	.84	.92	.59	.89
EN$_{blind}$	.80	.88	.66	.86

Table 1: Cohen Kappa Agreement for Main Accents and Caesura

Six German and ten English poems were annotated by two annotators to calculate the agreement for rhythm. Table 1 lists the agreement figures for main accents (m.ac) and caesuras. It shows that caesuras can be fairly reliably detected through silent reading in both languages. On the other hand, agreement on main accents is challenging. Figure 2 shows the confusion of main accents for German. While 0s are quite unambiguous, it is not always clear when to set a primary (2) or side accent (1).

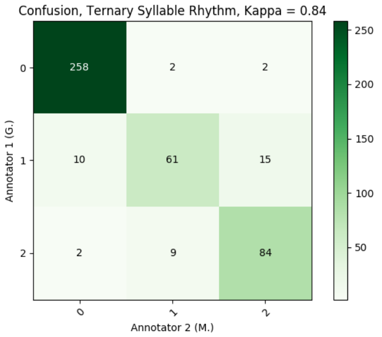


Figure 2: Confusion of German Main Accents

Meter and Foot: In poetry, meter is the basic prosodic structure of a verse. The underlying abstract, and often top-down prescribed, meter consists of a sequence of beat-bearing units (syllables) that are either prominent or non-prominent.Non-prominent beats are attached to prominent ones to build metrical feet (e.g. iambic or trochaic ones). This metrical structure is the scaffold, as it were, for the linguistic rhythm. Annotators first annotated the stress of syllables and in a subsequent step determined groupings of these syllables with foot boundaries, thus a foot is the grouping of metrical syllables. The meter (or measure) of a verse can be described as a regular sequence of feet, according to a specific sequence of syllable stress values.

	Syllable		Whole Line	
	meter	foot	meter	foot
DE$_{corr.}$	.98	.87	.94	.71
DE$_{blind}$	.98	.79	.92	.71
EN$_{blind}$	.94	.95	.87	.88

Table 2: Cohen Kappa Agreement for Metrical Stress and Foot Boundaries. Corr. is the agreement of the first version against the corrected version. Blind means that annotators did not see another annotation.

The meter annotation for the German data was first done in a full pass by a graduate student. A second student then started correcting this annotation with frequent discussions with the first author. While on average the agreement scores for all levels of annotation suggested reliable annotation after an initial batch of of 20 German poems, we found that agreement on particular poems was far lower than the average, especially for foot boundaries. Therefore, we corrected the whole set of 153 German poems, and the first author did a final pass. The agreement of this corrected version with the first version is shown in Table 2 in the row DE$_{corr.}$. To check whether annotators also agree when not exposed to pre-annotated data, a third annotator and the second annotator each annotated 10 diverse German poems from scratch. This is shown in DE$_{blind}$. For English, annotators 2 and 3 annotated 6 poems blind and then split the corpus.

Notably, agreement on syllables is acceptable, but feet were a bit problematic, especially for German. To investigate the sources of disagreement, we annotated and calculated agreement on all 153 poems. Close reading for disagreement of foot boundaries revealed that poems with κ around .8 had faulty guideline application (annotation error). 14 poems had an overall $\kappa < .6$, which stemmed from ambiguous rhythmical structure (multiple annotations are acceptable) and/or schema invariance, where a philological eye considers the whole structure of the poem and a naive annotation approach does not render the intended prosody correctly.

As an example for ambiguous foot boundaries, the following poem, Schiller's 'Bürgschaft', can be set in either *amphibrachic* feet, or as a mixture of *iambic* and *anapaestic* feet. Such conflicting annotations were discussed by Heyse (1827), who finds that in the Greek tradition the *anapaest* is preferable, but a 'weak amphibrachic gait' allows for a freer rhythmic composition. This suggests that Schiller was breaking with tradition.

```
(Foot Boundary Ambiguity) Schiller, 'Die Bürgschaft'

(1) met="-+-|-+-|-+-|"
    Ich lasse |    den Freund   dir | als Bürgen, |
(2) met="-+|--+|--+|-"
    Ich las   | se den Freund | dir   als Bürg | en,
Transl.: I leave this friend to you as guarantor

(1) met="-+-|-+-|-+-|"
    Ihn magst   du, | entrinn'   ich, | erwürgen. |
(2) met="-+|--+|--+|-"
    Ihn magst | du,   entrinn' | ich,   erwür | gen.
Transl.: Him you may strangle if I escape.

(1) (amphibrach)
(2) (iambus / anapaest)
```

Verse Measures: We develop a set of regular expressions to determine the measure of a line from its raw metrical annotation. We orient ourselves with the handbook of Knörrich (1971). The 'verse measure' (msr) is a label for the whole line according to recurring metrical feet. We label the verse according to its dominant foot, i.e., the repetition of patterns like `iambus` (−+), `trochee` (+−), `dactyl` (+−−), `anapaest` (−−+), or `amphibrach` (−+−). Also, the rules determine the number of stressed syllables in the line, where *di-*, *tri-*, *tetra-*, *penta-*, and *hexameter* signify 2, 3, 4, 5, and 6 stressed syllables accordingly. Thus, `+-+-+-` is an example for a trochaic.trimeter and `-+-+-+-+` is a iambic.tetrameter, since the foot boundaries should look like this: `-+|-+|-+|-+|`. Typically, female (unstressed) line endings are optional (cadence). Additionally, we annotate labels for (i) `inversion`, when the first foot is inverted, e.g., the first foot in a iambic line is trochaic: `+--+-+-+`, (ii) `relaxed`, if an unstressed syllable was inserted: `-+-+--+-+` (iambic.tetrameter.relaxed), (iii) and choliambic endings: `-+-+-+--+`. Besides these basic forms, we also implement historically important forms such as a strict `alexandrine`,[1] the `dactylic hexameter`,[2] conventionally known as 'hexameter', and some ode forms like the `asklepiadic` verse (`+-+--++--+-+`).

Table 3 lists the most frequent labels for each language without length, called short measure (smsr). The English data includes all datasets that are used in the experiments, as discussed in section 4.1.3.

English		German	
freq.	smsr	freq.	smsr
2096	iambic	1976	iambic
490	trochaic	793	trochaic
306	anapaest	258	amphibrach
255	amphibrach	206	alexandrine
248	daktylic	76	daktylic
152	hexameter	72	anapaest
91	prosodiakos	26	asklepiade
52	other	17	pherekrateus
35	alexandrine	14	glykoneus

Table 3: Most frequent verse measures in small English and German corpora, without length.

3 Large Poetry Corpora

In order to enable large scale experiments on poetry, we collect and standardize large poetry corpora for English and German. The English corpus contains around 3 million lines, while the German corpus contains around 2 million lines. The corpora and code can be found at `https://github.com/tnhaider/metrical-tagging-in-the-wild`

---

[1] Alexandrine: `-+-+-+-+-+-+-?`
The symbol before ? is optional

[2] Hexameter: `+--?+--?+--?+--?+--+-`

Our resources are designed in a standardized format to sustainably and interoperably archive poetry in both .json and TEI P5 XML. The .json format is intended for ease of use and speed of processing while retaining some expressiveness. Our XML format is built on top of a “Base Format”, the so-called DTA-Basisformat[3] (Haaf et al., 2014) that not only constrains the data to TEI P5 guidelines, but also regarding a stricter relaxNG schema that we modified for our annotation.[4]

3.1 A Large German Poetry Corpus

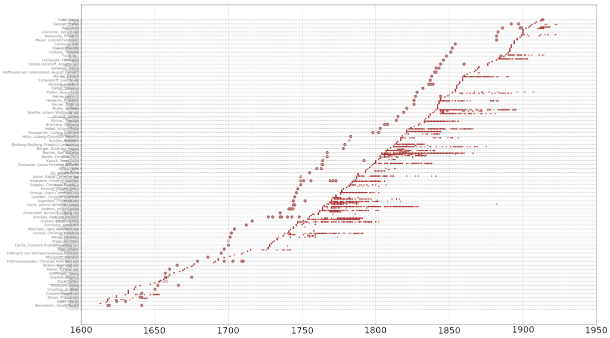


Figure 3: Each dot represents a poem of a German Author (y-axis) over Time (x-axis) from Textgrid (small dots bottom) and DTA (large dots top). 1600–1950. Authors are not aligned, and poems can be on top of each other.

We built a large, comprehensive, and easily searchable resource of New High German poetry by collecting and parsing the bulk of digitized corpora that contain public domain German literature. This includes the German Text Archive (DTA) (`http://deutschestextarchiv.de`) the Digital Library of Textgrid (`http://textgrid.de`), and also the German version of Project Gutenberg (which we omit from our experiments due to inconsistency).[5]

Each of these text collections is encoded with different conventions and varying degrees of consistency. Textgrid contains 51,264 poems with the genre label ‘Verse’, while DTA contains 23,877 poems with the genre label ‘Lyrik’. It should be noted that the whole DTA corpus contains in total 40,077 line groups that look like poems, but without the proper genre label, poems are likely embedded within other texts and might not come with proper meta-data. We implement XML parsers in python to extract each poem with its metadata and fix stanza and line boundaries. The metadata includes the author name, the title of the text, the year it was published, the title and genre of the volume it was published in, and finally, an identifier to retrieve the original source. We perform a cleaning procedure that removes extant XML information, obvious OCR mistakes, and normalize umlauts and special characters in various encodings,[6] particularly in DTA. We use langdetect[7] 1.0.8 to tag every poem with its language to filter out any poems that are not German (such as Latin or French). The corpus finally almost 2M lines in over 60k poems. In Figure 3 we plotted each poem in DTA and Textgrid over time, from 1600 to 1950. The x-axis shows the year of a poem, while the y-axis is populated by authors. One can see that DTA consists of full books that are organized by author (large dots) so that the datapoints for single poems get plotted on top of each other, while Textgrid has a time stamp for most single poems (after 1750), outlining the productive periods of authors.

3.2 A Large English Poetry Corpus

The English corpus contains the entirety of poetry that is available in the English Project Gutenberg (EPG) collection. We firstly collected all files with the metadatum ‘poetry’ in (temporal) batches with the GutenTag tool (Brooke et al., 2015), to then parse the entire collection in order to standardize the inconsistent XML annotation of GutenTag and remove duplicates, since EPG contains numerous different editions and issues containing the same material. We also filter out any lines (or tokens) that indicate illustrations, stage directions and the like. We use langdetect to filter any non-English material.

The github repository of Parrish (2018) previously provided the poetry in EPG by filtering single lines with a simple heuristic (anything that could look like a line), not only including prose with line breaks, but also without conserving the integrity of poems but providing a document identifier per line to find its origin. We offer our corpus in XML with intact document segmentation and metadata, still containing over 2.8 million lines.

---

[3] `http://www.deutschestextarchiv.de/doku/basisformat/`

[4] This schema defines a strict layout of poetic annotation. It allows us to validate XML files regarding their correctness. It is thus useful for manual annotation with the OxygenXML editor, avoiding parsing errors downstream.

[5] `https://www.projekt-gutenberg.org/`

[6] We fix the orthography both on string and bytecode level. We replace the rotunda (U+A75B) and the long s (U+017F), the latter of which is pervasive in DTA.

[7] `https://pypi.org/project/langdetect/`

4 Experiments

In the following, we carry out experiments to learn the previously annotated features and determine their degree of informativeness for each other with a multi-task setup. We include two additional datasets with English meter annotation, and evaluate pre-processing models for syllabification and part-of-speech tagging.

4.1 Preprocessing

Tokenization for both languages is performed with SoMaJo with a more conservative handling of apostrophes (to leave words with elided vowels intact) (Proisl and Uhrig, 2016). This tokenizer is more robust regarding special characters than NLTK. We also train models for hyphenation (syllabification) and part-of-speech (POS) tagging, since syllabification is a prerequisite to analyse prosody, and POS annotation allows us to gauge the role of syntax for prosodic analysis.

4.1.1 Hyphenation / Syllabification

For our purposes, proper syllable boundaries are paramount to determine the segmentation of lines regarding their rhythmic units. The test the following systems:[8] *Sonoripy*,[9] *Pyphen*,[10] *hypheNN*,[11] and a BiLSTM-CRF (Reimers and Gurevych, 2017)[12] with pretrained word2vec character embeddings. These embeddings were trained on the corpora in section 3.

To train and test our models, we use CELEX2 for English and extract hyphenation annotation from wiktionary for German.[13] We evaluate our models on 20,000 random held-out words for each language on word accuracy and syllable count. Word accuracy rejects any word with imperfect character boundaries, while syllable count is the more important figure to determine the proper length of a line. As seen in Table 4, the BiLSTM-CRF performs best for English and does not need any postprocessing. For German, the LSTM model is less useful as it tends to overfit, where over 10% of annotated lines were still rejected even though in-domain evaluation suggests good performance. We therefore use an ensemble with HypheNN, Pyphen and heuristic corrections for German, with only 3% error on the gold data, as seen in Table 5 (the datasets are discussed in section 4.1.3).

	German		English	
	w. acc.	sy. cnt	w. acc.	sy. cnt
SonoriPy	.476	.872	.270	.642
Pyphen	.839	.875	.475	.591
HypheNN	.909	.910	.822	.871
BiLSTM-CRF	**.939**	**.978**	**.936**	**.984**

Table 4: Evaluation of Syllabification Systems on Wiktionary (German) and CELEX (English).

	German	EPG64	FORB	PROS
# correct lines	3431	1098	1084	1564
# faulty lines	58	114	49	173

Table 5: Size of manually annotated corpora. Faulty lines denotes the number of lines where the automatic syllabification failed. Correct lines are used for experiments, since only there the gold annotation aligns.

4.1.2 POS tagging

Since we are dealing with historical data, POS taggers trained on current data might degrade in quality and it has been frequently noted that poetry makes use of non-canonical syntactic structures (Gopidi and Alam, 2019). For German, we evaluate the robustness of POS taggers across different text genres. We use the gold annotation of the TIGER corpus (modern newspaper), and pre-tagged sentences from DTA, including annotated poetry (Lyrik), fiction (Belletristik) and news (Zeitung).[14] The STTS tagset is used. We train and test Conditional Random Fields (CRF)[15] to determine a robust POS model.[16] See Table 7 for an overview of the cross-genre evaluation. We find that training on TIGER is not robust to tag across domains, falling to around .8 F1-score when tested

---

[8] Syllabipy determines boundaries based on the sonority principle, Pyphen uses the Hunspell dictionaries, and HypheNN is a simple feed forward network that is trained on character windows.

[9] `https://github.com/alexestes/SonoriPy` `https://github.com/henchc/syllabipy`

[10] `pyphen.org`

[11] `github.com/msiemens/HypheNN-de`

[12] `https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf`

[13] For German, wiktionary contains 398.482 hyphenated words, and 130.000 word forms in CELEX. Unfortunately, German CELEX does not have proper umlauts, and models trained on these were not suitable for poetry. For English, wiktionary only contains 5,142 hyphenated words, but 160,000 word forms in CELEX.

[14] DTA was tagged with TreeTagger and manually corrected afterwards. `http://www.deutschestextarchiv.de/doku/pos`

[15] From the sklearn crf-suite

[16] As features, we use the word form, the preceding and following two words and POS tags, orthographic information (capitalization), character prefixes and suffixes of length 1, 2, 3 and 4.

POS	Noun	Adj.	Full V.	Adverb	Modal V.	Interj.	Pron.	Prep.	Konj.	Art.
Abr. POS Tag	NN	ADJ	VV	ADV	VM	ITJ	P	AP	KO	AR
Accent Ratio	.97	.89	.84	.75	.73	.55	.4–.015	.27	.23	.06

Table 6: Accent ratio for part-of-speech of German monosyllabic words (ratio of metrical stress).

against poetry and news from DTA. These results suggest that this is mainly due to (historical) orthography, and to a lesser extent due to local syntactic inversions.

Test	Train				
	TIGER	DTA	DTA+TIG.	Belletr.	Lyrik
Lyrik	.795	.949	.948	.947	**.953**
Belletristik	.837	**.956**	.954	.955	.955
DTA Zeitung	.793	**.934**	.933	.911	.900
TIGER	**.971**	.928	.958	.929	.913

Table 7: Evaluation of German POS taggers across genres. F1-scores.

For English, we test the Stanford core-nlp tagger.[17] The tagset follows the convention in the Penn TreeBank. This tagger is not geared towards historical poetry and consequently fails in a number of cases. We manually correct 50 random lines and determine an accuracy of 72%, where particularly the 'NN' tag is overused. This renders the English POS annotation unreliable for our experiments.

4.1.3 Additional Data and Format

The annotated corpora for English include: (1) The for-better-for-verse (FORB) collection[18] with around 1200 lines which was used by Agirrezabal et al. (2016, 2019), and (2) the 1700 lines of poetry against which `prosodic`[19] (Anttila and Heuser, 2016; Algee-Hewitt et al., 2014) was evaluated (PROS). We merge these with our own (3) 1200 lines in 64 English poems (EPG64). The first two corpora were already annotated for metrical syllable stress. However, FORB does not contain readily available foot boundaries, and in PROS foot boundaries are occasionally set after each syllable.[20] Table 5 shows the number of lines in each of our datasets and the number of lines that were incorrectly segmented by our best syllabification systems.

Figure 4 shows an example line in the data layout that is used for the experiments, including the 'measure' that was derived with regular expressions from the meter line. 'Syll' is the position of the syllable in a word, 0 for monosyllaba, otherwise index starting at 1. We removed punctuation to properly render line measures, even through punctuation is a good signal for caesuras (see Figure 1).

```
# tok  met ft pos syll csr main smsr    measure    met_line

1  Look  +  .  VB   0    .  1  iambic  i.penta.inv +--+-+-+-+
2  on    −  .  IN   0    .  0  iambic  i.penta.inv +--+-+-+-+
3  my    −  .  PRP$ 0    .  0  iambic  i.penta.inv +--+-+-+-+
4  works +  :  NNS  0    :  2  iambic  i.penta.inv +--+-+-+-+
5  ye    −  .  PRP$ 0    .  0  iambic  i.penta.inv +--+-+-+-+
6  Might +  :  NNP  1    .  1  iambic  i.penta.inv +--+-+-+-+
7  y     −  .  NNP  2    :  0  iambic  i.penta.inv +--+-+-+-+
8  and   +  :  CC   0    .  0  iambic  i.penta.inv +--+-+-+-+
9  de    −  .  VB   1    .  0  iambic  i.penta.inv +--+-+-+-+
10 spair'+  :  VB   2    :  1  iambic  i.penta.inv +--+-+-+-+
```

Figure 4: Tabular data format for experiments. Author of this line: Percy Blythe Shelley.

4.2 Accent Ratio of Part-of-Speech

Previous research has noted that part-of-speech annotation provides a good signal for the stress of words (Nenkova et al., 2007; Greene et al., 2010). To test this, we calculate the pos-accent ratio of monosyllabic words in our German annotation by dividing how often a particular part-of-speech appears stressed (+) in the corpus by how often this part-of-speech occurs in the corpus. We restrict this to monosyllabic words, as polysyllabic words typically have a lexical stress contour. The result is a hierarchy of stress that we report in Table 6. At the ends of the spectrum, we see that nouns are usually stressed, while articles are seldom stressed.

4.3 Learning Meter

To learn the previously annotated metrical values for each syllable, the task is framed as sequence classification. Syllable tokens are at the input and

---

[17] `https://nlp.stanford.edu/software/tagger.shtml`

[18] `https://github.com/manexagirrezabal/for_better_for_verse/tree/master/poems`

[19] `https://github.com/quadrismegistus/prosodic`

[20] Additionally, FORB makes use of a `<seg>` tag to indicate syllable boundaries, so we do not derive the position of a syllable in a word. It also contains two competing annotations, `<met>` and `<real>`. The former is the supposedly proper metrical annotation, while the latter corresponds to a more natural rhythm (with a tendency to accept inversions and stress clashes). We only chose `<real>` when `<met>` doesn't match the syllable count (ca. 200 cases), likely deviating from the setup in (Agirrezabal et al., 2016, 2019).

the respective `met` labels at the output. We test a nominal CRF (see section 4.1.2) and a BERT model as baselines and implement a BiLSTM-CRF[21] with pre-trained syllable embeddings. These embeddings were trained by splitting all syllables in the corpora from section 3, and training word2vec embeddings over syllables. This system uses three layers of size 100 for the BiLSTM and does the final label prediction with a linear-Chain CRF. Variable dropout of .25 was applied at both input and output. No extra character encodings were used (as these hurt both speed and accuracy).

We do a three fold cross validation with 80/10/10 splits and average the results, reporting results on the test set in Table 8. We evaluate prediction accuracy on syllables and the accuracy of whether the whole line was tagged correctly (line acc.). Line accuracy is especially important if we want to classify poetic verse measures.

	English		German	
	syll. acc	line acc	syll. acc	line acc
CRF	.922	.478	.941	.553
BERT	.850	.371	.932	.498
BiLSTM-CRF	**.955**	**.831**	**.968**	**.877**
Agirrezabal (2019)	.930	.614	-	-
Antilla & Heuser (2016)	.894	.607	-	-

Table 8: Best Classifiers for Metrical Syllable Stress

Though not directly comparable (data composition differs), we include results as reported by Agirrezabal et al. (2019) for the English for-better-for-verse dataset. We also test the system '`prosodic`' of Anttila and Heuser (2016) against our gold data (EPG64), resulting in .85 accuracy for syllables and .44 for lines. When only evaluating on lines that were syllabified to the correct length (their syllabifier), 27% of lines are lost, but on this subset it achieves .89 syllable and .61 line accuracy.

Learning the sequence of metrical syllable stress with BERT cannot compete our other models, possibly resulting from an improper syllable representation, as the word-piece tokenizer segments word chunks other than syllables.

We also experiment with framing the task as document (line) classification, where BERT should learn the verse label (e.g., iambic.pentameter) for a given sequence of words. On the small English dataset, BERT only achieves around .22 F1-macro and .42 F1-micro. We then tagged 20,000 lines of the large English corpus with a BiLSTM-CRF model and trained BERT on this larger dataset, reaching .48 F1-macro and .62 F1-micro. In this setup, BERT detects frequent classes like iambic.pentameter or trochaic.tetrameter fairly well (.8), but it appears that this model mainly picks up on the length of lines and fails to learn measures other than iambus and trochee like dactyl or anapaest or irregular verse with inversions. This might limit experiments with transfer learning of verse measure knowledge.

4.4 Pairwise Joint Prosodic Task Learning

	met	feet	syllin	pos	csra	m.ac	emo
single	.964	.871	**.952**	.864	.912	.866	.328
+met	-	**.922**	.949	.856	.918	.869	.347
+feet	.961	-	.948	.853	.917	.863	.368
+syllin	.966	.900	-	.860	.919	.867	.330
+pos	.956	.879	**.953**	-	**.924**	**.879**	**.393**
+csra	.961	.886	.940	.855	-	.868	.364
+m.ac	.964	**.915**	.948	.865	.915	-	.354
+smsr	.965	.884	.942	.854	.918	.868	**.378**
+fmsr	**.968**	.899	.938	.858	**.926**	.868	**.395**
+m_line	.966	.882	.937	.853	.919	.868	**.398**
+all	.967	**.930**	.947	.790	.919	.870	.377

Table 9: Accuracy for Pairwise Joint Task Learning.

With the aim of learning the relationships between our different annotation layers, we performed experiments with a multi-task setup. We used the BiLSTM architecture from the previous experiment, where the sequence of syllable embedding vectors is at the input, and the respective sequence of labels at the output. We used the German dataset here, as the annotation is generally more reliable (e.g., POS). In this experiment we also try to learn the annotation of aesthetic emotions that was described for this dataset by Haider et al. (2020). Each line was annotated with one or two emotions from a set of nine emotions. Here, we only used the primary emotion label per line.

First, we trained a single task model for each annotation layer, then all tasks jointly (`+all`), and finally pair-wise combinations (`+<auxiliary task>`). In Table 9, we report the accuracy on syllable level for each main task with their respective auxiliary tasks.

Note that learning syllable-level POS does not benefit from any other task, not even the syllable position in the word, while several tasks like caesuras, main accents and emotions benefit from additional POS information. Predicting meter also degrades from an additional POS task, which possibly interferes with the syllable embeddings. Meter might be also more contextual than suggested in Table 6.

---

[21] `https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf`

However, meter tagging slightly benefits from fine-grained verse measure labels. Interestingly, learning foot boundaries heavily benefits from jointly learning syllable stress. In a single task setup, foot boundaries are learned with .871 accuracy, but in combination with metrical stress, feet are learned with .922 acc. and in combination with main accents at .915. This might be expected, as foot groupings are dependent on the regularity of repeating metrical syllable stresses (though less dependent on main accents). However, our annotators only achieved Kappa agreement of .87 for feet. It is curious then, how the model overcomes this ambiguity. When learning all tasks jointly (`+all`), foot prediction even reaches .930, suggesting that feet are related to all other prosodic annotations.

We observe that the exchange between caesuras and main accents is negligible. However, caesuras benefit from POS (despite the absence of punctuation), syllable position (syllin) and global measures (msr), indicating that caesuras are integral to poetic rhythm and fairly dependent on syntax.

For emotions we find, despite the hard task (line instead of stanza), and only using syllable embeddings rather than proper word embeddings, that the single task setup is already better than the majority baseline. More importantly, we can see that jointly learning POS or verse measure benefits the emotion prediction (slightly the meter prediction itself: .97). This suggests that there might be a systematic relationship between meter and emotion.

5 Related Work

5.1 Annotation of Prosodic Features

Earlier work (Nenkova et al., 2007) already found strong evidence that part-of-speech tags, accent-ratio[22] and local context provide good signals for the prediction of word stress. Subsequently, models like MLP (Agirrezabal et al., 2016), CRFs and LSTMs (Estes and Hench, 2016; Agirrezabal et al., 2019) and transformer models (Talman et al., 2019) have notably improved the performance to predict the prosodic stress of words and syllables. Unfortunately, most of this work only evaluates model accuracy on syllable or word level, with the exception of Agirrezabal et al. (2019).

A digital resource with annotation of poetic meter was missing for New High German. For Middle High German, Estes and Hench (2016) annotated a metrical scheme for hybrid meter. Anttila et al. (2018) annotated main accents in political speeches. Agirrezabal et al. (2016, 2019) used the English *for-better-for-verse* and the dataset of Navarro et al. (2016), who annotated hendecasyllabic verse (11 syllables) in Spanish Golden Age sonnets. Algee-Hewitt et al. (2014) annotated 1700 lines of English poetry to evaluate their system.

5.2 Poetry Corpora & Generation

Several poetry corpora have been used in the NLP community. Work on English has strongly focused on iambic pentameter, e.g., of Shakespeare (Greene et al., 2010) or with broader scope (Jhamtani et al., 2017; Lau et al., 2018; Hopkins and Kiela, 2017). Other work has focused on specific genres like Spanish sonnets (Ruiz Fabo et al., 2020), limericks (Jhamtani et al., 2019), or Chinese Tang poetry (Zhang and Lapata, 2014). There are further resources with rhyme patterns (Reddy and Knight, 2011; Haider and Kuhn, 2018) or emotion annotation (Haider et al., 2020). Truly large corpora are still hard to find, besides the Gutenberg project for English and Textgrid and DTA for German.

6 Conclusion

We created large poetry corpora for English and German to support computational literary studies and annotated prosodic features in smaller corpora. Our evaluation shows that a multitude of features can be annotated through silent reading, including meter, main accents and caesuras, even though foot annotation can be challenging. Finally, we performed first experiments with a multi-task setup to find beneficial relations between certain prosodic tasks. Learning metrical annotation, including feet and caesuras, largely benefits from a global verse measure label, while foot boundaries also benefit from any joint learning with syllable stress and all features alltogether, even surpassing the human upper bound.

Acknowledgments

We thank Gesine Fuhrmann and Debby Trzeciak for their annotations, and Steffen Eger, Winfried Menninghaus, Anne-Kathrin Schumann, and our reviewers for feedback on the manuscript. Thanks also goes to Ryan Heuser for help with his tool `prosodic`.

---

[22] The ratio of how often a word form appears stressed vs. unstressed in a corpus

References

Manex Agirrezabal, Iñaki Alegria, and Mans Hulden. 2016. Machine learning for metrical analysis of English poetry. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 772–781, Osaka, Japan. The COLING 2016 Organizing Committee.

Manex Agirrezabal, Iñaki Alegria, and Mans Hulden. 2019. A comparison of feature-based and neural scansion of poetry. *RANLP 2019, arXiv preprint arXiv:1711.00938.*

Mark Algee-Hewitt, Ryan Heuser, Maria Kraxenberger, JD Porter, Jonny Sensenbaugh, and Justin Tackett. 2014. The stanford literary lab transhistorical poetry project phase ii: Metrical form. In *DH*.

Arto Anttila, Timothy Dozat, Daniel Galbraith, and Naomi Shapiro. 2018. Sentence stress in presidential speeches. *Lingbuzz Preprints*.

Arto Anttila and Ryan Heuser. 2016. Phonological and metrical variation across genres. In *Proceedings of the Annual Meetings on Phonology*, volume 3.

Derek Attridge. 2014. *The rhythms of English poetry*. Routledge.

MH Beissinger. 2012. Oral poetry. *The Princeton encyclopedia of poetry and poetics*, pages 978–81.

Klemens Bobenhausen. 2011. The metricalizer2–automated metrical markup of german poetry. *Current Trends in Metrical Analysis, Bern: Peter Lang*, pages 119–131.

Julian Brooke, Adam Hammond, and Graeme Hirst. 2015. Gutentag: an nlp-driven tool for digital humanities research in the project gutenberg corpus. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 42–47.

Thomas Carper and Derek Attridge. 2020. *Meter and meaning: an introduction to rhythm in poetry*. Routledge.

Alex Estes and Christopher Hench. 2016. Supervised machine learning for hybrid meter. In *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*, pages 1–8.

Nigel Fabb and Morris Halle. 2008. *Meter in poetry: A new theory*. Cambridge University Press.

Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1191.

Amitha Gopidi and Aniket Alam. 2019. Computational analysis of the historical changes in poetry and prose. In *Proceedings of the 1st International Workshop on Computational Approaches to Historical Language Change*, pages 14–22.

Erica Greene, Tugba Bodrumlu, and Kevin Knight. 2010. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 524–533.

Susanne Haaf, Alexander Geyken, and Frank Wiegand. 2014. The dta "base format": A tei subset for the compilation of a large reference corpus of printed text from multiple sources. *Journal of the Text Encoding Initiative*, (8).

Thomas Haider, Steffen Eger, Evgeny Kim, Roman Klinger, and Winfried Menninghaus. 2020. Po-emo: Conceptualization, annotation, and modeling of aesthetic emotions in german and english poetry. In *Proceedings of The 12th Language Resources and Evaluation Conference & arXiv:2003.07723*, pages 1652–1663.

Thomas Haider and Jonas Kuhn. 2018. Supervised rhyme detection with siamese recurrent networks. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 81–86.

Johann Christian August Heyse. 1827. Theoretisch-praktische grammatik oder lehrbuch zum reinen und richtigen sprechen, lesen und schreiben der deutschen sprache.

Jack Hopkins and Douwe Kiela. 2017. Automatically generating rhythmic verse with neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 168–178.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence-to-sequence models. *arXiv preprint arXiv:1707.01161*.

Harsh Jhamtani, Sanket Vaibhav Mehta, Jaime Carbonell, and Taylor Berg-Kirkpatrick. 2019. Learning rhyming constraints using structured adversaries. *arXiv preprint arXiv:1909.06743*.

Sofoklis Kakouros, Joris Pelemans, Lyan Verwimp, Patrick Wambacq, and Okko Räsänen. 2016. Analyzing the contribution of top-down lexical and bottom-up acoustic cues in the detection of sentence prominence. *Proceedings Interspeech 2016*, 8:1074–1078.

Paul Kiparsky. 2020. Metered verse. *Annual Review of Linguistics*, 6:25–44.

Otto Knörrich. 1971. *Die deutsche Lyrik der Gegenwart*, volume 401. Kröner.

Maria Kraxenberger and Winfried Menninghaus. 2016. Mimological reveries? disconfirming the hypothesis of phono-emotional iconicity in poetry. *Frontiers in Psychology*, 7:1779.

Jey Han Lau, Trevor Cohn, Timothy Baldwin, Julian Brooke, and Adam Hammond. 2018. Deep-speare: A joint neural model of poetic language, meter and rhyme. *arXiv preprint arXiv:1807.03491*.

Katja Mellmann. 2007. Versanalyse. *Handbuch Literaturwissenschaft 2: Methoden und Theorien*, pages 81–97.

Winfried Menninghaus, Valentin Wagner, Eugen Wassiliwizky, Thomas Jacobsen, and Christine A Knoop. 2017. The emotional and aesthetic powers of parallelistic diction. *Poetics*, 63:47–59.

Yoonsook Mo, Jennifer Cole, and Eun-Kyung Lee. 2008. Naïve listeners' prominence and boundary perception. *Proc. Speech Prosody, Campinas, Brazil*, pages 735–738.

Borja Navarro, María Ribes Lafoz, and Noelia Sánchez. 2016. Metrical annotation of a large corpus of spanish sonnets: representation, scansion and evaluation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4360–4364.

Borja Navarro-Colorado. 2018. A metrical scansion system for fixed-metre spanish poetry. *Digital Scholarship in the Humanities*, 33(1):112–127.

Ani Nenkova, Jason Brenier, Anubha Kothari, Sasha Calhoun, Laura Whitton, David Beaver, and Dan Jurafsky. 2007. To memorize or to predict: Prominence labeling in conversational speech. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 9–16.

Allison Parrish. 2018. A Gutenberg Poetry Corpus https://github.com/aparrish/gutenberg-poetry-corpus.

Petr Plecháč. 2020. Relative contributions of shakespeare and fletcher in henry viii: An analysis based on most frequent words and most frequent rhythmic patterns. *Oxford Journal of Digital Humanities (DSH), arXiv preprint arXiv:1911.05652*.

Thomas Proisl and Peter Uhrig. 2016. SoMaJo: State-of-the-art tokenization for German web and social media texts. In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 57–62, Berlin. Association for Computational Linguistics (ACL).

Sravana Reddy and Kevin Knight. 2011. Unsupervised discovery of rhyme schemes. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 77–82.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 338–348, Copenhagen, Denmark.

Andrew Rosenberg. 2010. Autobi-a tool for automatic tobi annotation. In *Eleventh Annual Conference of the International Speech Communication Association*.

Ina Rösiger and Arndt Riester. 2015. Using prosodic annotations to improve coreference resolution of spoken text. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 83–88.

Pablo Ruiz Fabo, Helena Bermúdez Sabel, Clara Martínez Cantón, and Elena González-Blanco. 2020. The diachronic spanish sonnet corpus: Tei and linked open data encoding, data distribution, and metrical findings. *Digital Scholarship in the Humanities*.

T Steele. 2012. Verse and prose. *The Princeton encyclopedia of poetry and poetics*, pages 1507–1513.

Aarne Talman, Antti Suni, Hande Celikkanat, Sofoklis Kakouros, Jörg Tiedemann, and Martti Vainio. 2019. Predicting prosodic prominence from text with pretrained contextualized word representations. *arXiv preprint arXiv:1908.02262*.

Rafael Valle, Kevin Shih, Ryan Prenger, and Bryan Catanzaro. 2020. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. *arXiv preprint arXiv:2005.05957*.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680.

Enhancing Aspect-level Sentiment Analysis with Word Dependencies

Yuanhe Tian[♥][*], **Guimin Chen**[♡][*], **Yan Song**[♠][♡][†]
[♥]University of Washington [♡]Shenzhen Research Institute of Big Data
[♠]The Chinese University of Hong Kong (Shenzhen)
[♥]`yhtian@uw.edu` [♡]`chenguimin@sribd.cn` [♠]`songyan@cuhk.edu.cn`

Abstract

Aspect-level sentiment analysis (ASA) has received much attention in recent years. Most existing approaches tried to leverage syntactic information, such as the dependency parsing results of the input text, to improve sentiment analysis on different aspects. Although these approaches achieved satisfying results, their main focus is to leverage the dependency arcs among words where the dependency type information is omitted; and they model different dependencies equally where the noisy dependency results may hurt model performance. In this paper, we propose an approach to enhance aspect-level sentiment analysis with word dependencies, where the type information is modeled by key-value memory networks and different dependency results are selectively leveraged. Experimental results on five benchmark datasets demonstrate the effectiveness of our approach, where it outperforms baseline models on all datasets and achieves state-of-the-art performance on three of them.[1]

1 Introduction

Aspect-level sentiment analysis (ASA) determines the sentiment polarity of a given input text on the fine-grained level, where the sentiment towards a particular aspect in the text is predicted instead of the entire input. E.g., the sentiment of an aspect "*bar service*" in the sentence "*Total environment is fantastic although bar service is poor.*" is negative, although the text as a whole conveys a positive sentiment polarity. Due to its high practical value in many scenarios, e.g., product review analysis, social media tracking, etc., ASA attracts much attention in the natural language processing (NLP) community for years (Tang et al., 2016a,b; He et al., 2018a; Sun et al., 2019; Zhang et al., 2019; Song et al., 2019; Huang and Carley, 2019).

In recent studies, neural networks, especially recurrent models with attention mechanism, are widely applied in this task, where many of them (Wang et al., 2016; Tang et al., 2016a; Chen et al., 2017; Ma et al., 2017; Fan et al., 2018; Liang et al., 2019; Tang et al., 2020) model semantic relatedness between context and aspect words to facilitate sentiment analysis on aspects. There are other approaches using additional inputs such as word position (Gu et al., 2018), document information (He et al., 2018b; Li et al., 2018a), commonsense knowledge (Ma et al., 2018). Among all such inputs, dependency results of the input text are proved to be a kind of useful information (He et al., 2018a; Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019; Wang et al., 2020; Tang et al., 2020), because they can help the model locate important content that modifies the aspect words and thus further suggests the sentiment towards the aspect words. Previous approaches with attention mechanism (He et al., 2018a; Wang et al., 2020), graph neural networks (GNN) (Sun et al., 2019; Huang and Carley, 2019; Zhang et al., 2019; Wang et al., 2020) and transformer (Tang et al., 2020) are applied in leveraging such information. However, most of them mainly focus on using the dependencies among words and omit to leverage other information such as relation types, which could provide useful cues to predict the sentiment. Also, they model all dependency information instances equally without weighting them according to their contribution to the task, where noisy information from the auto-generated dependency tree may hurt model performance. Therefore, improved methods are expected to comprehensively and efficiently learn dependencies among words to enhance ASA.

To address the aforementioned limitations, in this paper we propose an effective and efficient neural approach to ASA with incorporating word dependencies, which is acquired from off-the-shelf toolkits and modeled by key-value memory net-

---

[*] Equal contribution.

[†] Corresponding author.

[1] The code and different models are released at `https://github.com/cuhksz-nlp/ASA-WD`.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3726–3739
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

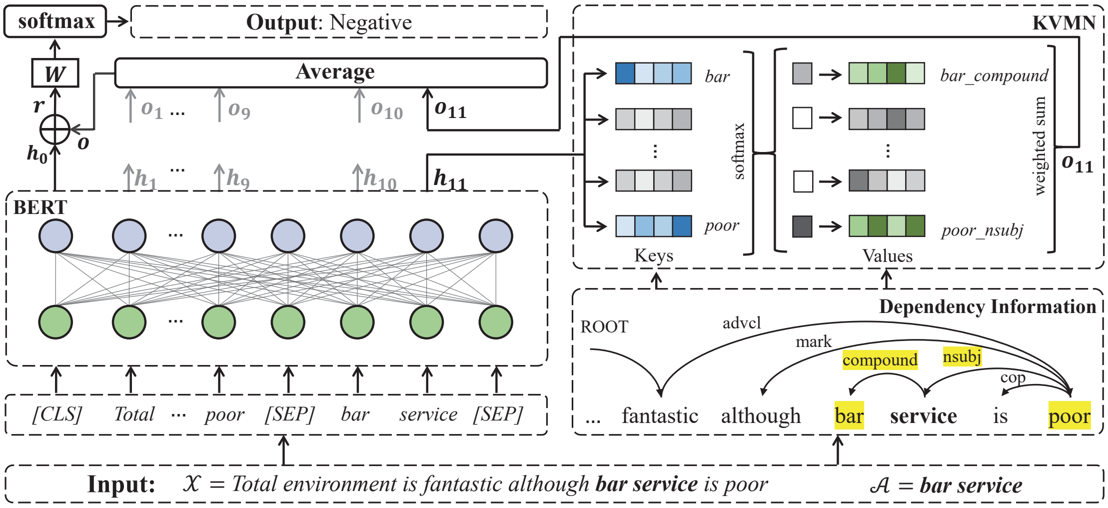


Figure 1: The overall architecture of the proposed model. The left part illustrates the backbone encoder (BERT) and decoder for ASA; the right part demonstrates the key-value memory networks (KVMN) for dependency information incorporation, where we use example word dependencies and their types (highlighted in yellow) of the aspect term "*service*" to show that how they are extracted, weighted and then fed into the left part for ASA.

works (KVMN) (Miller et al., 2016). In detail, for each input text parsed by a dependency parser, we extract its dependency relations and feed them into the KVMN, in which word-word associations and their corresponding dependency types are mapped to keys and values, respectively. Then the KVMN learns and weights different dependency knowledge according to the contribution of their corresponding keys to the ASA task, and provides the resulted representations to a regular ASA model, i.e., a BERT-based classifier, for final aspect-level sentiment predictions. In doing so, the proposed approach not only comprehensively leverages both word relations and their dependency types, but also effectively weights them through the memory mechanism according to their contributions to the ASA task. We evaluate the proposed approach on five benchmark datasets, where our approach outperforms the baselines on all datasets and achieves state-of-the-art on three of them.

2 The Approach

The task of ASA aims to analyze the sentiment of a text towards a specific aspect, which is formalized as a classification task performing on sentence-aspect pairs (Tang et al., 2016b; Ma et al., 2017; Xue and Li, 2018; Hazarika et al., 2018; Fan et al., 2018; Huang and Carley, 2018; Tang et al., 2019; Chen and Qian, 2019; Tan et al., 2019). In detail, each input sentence and the aspect terms in it are denoted by $\mathcal{X} = x_1, x_2, \cdots, x_n$ and $\mathcal{A} = a_1, a_2 \cdots, a_m$, respectively, where $\mathcal{A}$ is the sub-string of $\mathcal{X}$ ($\mathcal{A} \subset \mathcal{X}$), n and m refer to the word-based length of $\mathcal{X}$ and $\mathcal{A}$. Following this paradigm, we design the architecture of our approach in Figure 1, with a BERT-based (Devlin et al., 2019) encoder illustrated on the left to compute the sentence-aspect pair representation $\mathbf{r}$, and enhanced by the word dependency information obtained from the KVMN module on the right, then the result is fed into a softmax decoder to predict the text sentiment towards the aspect. Therefore, ASA through our approach can be formalized as

$$\hat{y} = \underset{y \in \mathcal{T}}{\arg\max}\, p(y|\mathcal{X}, \mathcal{A}, \mathrm{KVMN}(\mathcal{X}, \mathcal{A})) \quad (1)$$

where $\mathcal{T}$ denotes the set of sentiment polarities for y and p computes the probability of predicting $y \in \mathcal{T}$ given $\mathcal{X}$ and $\mathcal{A}$. $\hat{y}$ refers to the predicted sentiment polarity type for $\mathcal{A}$ in the context of $\mathcal{X}$. In the rest of this section, we firstly describe KVMN for leveraging word dependencies, then explain how the resulted representations are integrated into the backbone sentiment classifier.

2.1 KVMN for Word Dependencies

High quality text representations always play a crucial role to obtain good model performance for different NLP tasks (Song et al., 2017; Seyler et al., 2018; Song and Shi, 2018; Song et al., 2018; Babanejad et al., 2020), where contextual features, including n-grams and syntactic information, have been demonstrated to be effective in enhancing text representation and thus leads to improvements on different models (Song et al., 2006, 2009; Song and Xia, 2012; Song et al., 2012; Song and Xia, 2013; Dong et al., 2014; Miller et al., 2016; Seyler et al., 2018; Diao et al., 2019; Sun et al., 2019; Zhang et al., 2019; Huang and Carley, 2019; Tian et al., 2020b,c,d,e; Chen et al., 2020). Among all these features, dependency ones have been widely used, especially for ASA. To incorporate word dependencies into ASA task, there are many options, including attention mechanism (He et al., 2018a) where the information of dependency types among word pairs are omitted, and GNN and Transformer-based methods (Sun et al., 2019; Zhang et al., 2019; Wang et al., 2020; Tang et al., 2020) that require complicated architectures to model the entire dependency structure of an input text. Compared to these options, KVMN, whose variants have been demonstrated to be effective in incorporating contextual features (Miller et al., 2016; Guan et al., 2019; Song et al., 2020; Tian et al., 2020a,f; Nie et al., 2020), not only provides an appropriate way to leverage both word-word relations as well as their corresponding dependency types, but also weights different dependency information according to their contribution to the ASA task.

In detail, to build the KVMN, we firstly collect all word-word relations extracted from the parse results of a corpus via an off-the-shelf toolkit and use them to form the key set, and map their corresponding dependency types to the value set. Then, two embedding matrices, $\mathbf{K}$ and $\mathbf{V}$ are applied to the key and value sets with each vector representing a key or a value in the sets. At training or prediction stage, given an input text, our model obtains its dependency parsing result, i.e., for each w_i in a sentence-aspect pair, where w_i comes from $\mathcal{X}$, $\mathcal{A}$, or both $\mathcal{X}$ and $\mathcal{A}$, we extract words associated with w_i and their corresponding dependency types from the parse results. Note that, for each word, we use its inbound and outbound dependency types to represent its governor and dependent word, respectively. Therefore, for example, as illustrated in Figure 1, the words associated to the aspect word "*service*" are "*poor*" (governor) and "*bar*" (dependent); their corresponding dependency types are thus "*nsubj*" and "*compound*", respectively. Afterwards, we map the associated words and their corresponding dependency types to keys $\mathcal{K}_i = \{k_{i,1}, k_{i,2}, \cdots, k_{i,j}, \cdots, k_{i,q}\}$ and values $\mathcal{V}_i = \{v_{i,1}, v_{i,2}, \cdots, v_{i,j}, \cdots, v_{i,q}\}$ from $\mathbf{K}$ and $\mathbf{V}$ in the KVMN, where each item in $\mathcal{K}_i$ and $\mathcal{V}_i$ has its embedding denoted by $\mathbf{e}^k_{i,j}$ and $\mathbf{e}^v_{i,j}$, respectively. Once the keys and values are placed, we take the hidden vector $\mathbf{h}_i$ for w_i from the encoder (i.e., BERT), and compute the weight assigning to each value $v_{i,j}$ by

$$p_{i,j} = \frac{exp(\mathbf{h_i} \cdot \mathbf{e}^k_{i,j})}{\sum_{j=1}^{q} exp(\mathbf{h_i} \cdot \mathbf{e}^k_{i,j})} \tag{2}$$

We thus use $p_{i,j}$ to activate the corresponding values $v_{i,j}$ and compute the weighted sum by

$$\mathbf{o}_i = \sum_{j=1}^{q} p_{i,j} \mathbf{e}^v_{i,j} \tag{3}$$

where $\mathbf{o}_i$ refers to the output of the KVMN model for w_i and carries its word dependency information.

2.2 Word Dependency Integration for ASA

As shown in Figure 1, the entire model starts from encoding the input text. For the aforementioned sentence-aspect pair for ASA, it is normally organized by concatenating $\mathcal{X}$ and $\mathcal{A}$ to form a special sequence of $[[CLS], \mathcal{X}, [SEP], \mathcal{A}, [SEP]]$, and then feed it into an encoder, i.e., BERT, to obtain the hidden vectors by

$$[\mathbf{h}_0, \mathbf{H}^{\mathcal{X}}, \mathbf{H}^{\mathcal{A}}] = BERT(\mathcal{X}, \mathcal{A}) \tag{4}$$

where $\mathbf{h}_0$ denotes the hidden vector for the text-initial symbol $[CLS]$, and $\mathbf{H}^{\mathcal{X}}$, $\mathbf{H}^{\mathcal{A}}$ the embedding matrices of words in $\mathcal{X}$ and $\mathcal{A}$, respectively.

Upon the modeling of word dependencies for each w_i, different $\mathbf{o}_i$ are obtained and averaged, then concatenated with $\mathbf{h}_0$ by

$$\mathbf{r} = \mathbf{h}_0 \oplus \frac{1}{l} \cdot \sum_{i=1}^{l} \mathbf{o}_i \tag{5}$$

where $\mathbf{r}$ is the representation for the input sentence-aspect pair enhanced by word dependencies, and the value of l equals to n, m, or $n+m$ if all w_i come from $\mathcal{X}$ only, $\mathcal{A}$ only, or $\mathcal{X}+\mathcal{A}$, respectively.[2] Then, we use a dense layer with a trainable matrix $\mathbf{W}$ and vector $\mathbf{b}$ to align $\mathbf{r}$'s dimension to the output space by $\mathbf{u} = \mathbf{W} \cdot \mathbf{r} + \mathbf{b}$, with each dimension of $\mathbf{u}$ corresponding to a sentiment type. Finally, a softmax function is applied to $\mathbf{u}$ to predict the output sentiment $\hat{y}$ for the aspect $\mathcal{A}$ in $\mathcal{X}$:

$$\hat{y} = \arg\max \frac{exp(u^t)}{\sum_{t=1}^{|\mathcal{T}|} exp(u^t)} \tag{6}$$

where u^t is the value at dimension t in $\mathbf{u}$.

---

[2] Figure 1 illustrate the case that w_i comes from $\mathcal{X} + \mathcal{A}$, where $i \in [1, 11]$ for all $\mathbf{h}_i$.

[3] For all datasets, the sum of aspect samples under three sentiment polarities is larger than the total sentence numbers,

	LAP14		REST14		REST15		REST16		TWITTER	
	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST	TRAIN	TEST
POSITIVE #	994	341	2,164	728	907	326	1,229	469	1,561	173
NEUTRAL #	464	169	637	196	36	34	69	30	3,127	346
NEGATIVE #	870	128	807	182	254	207	437	114	1,560	173
SENTENCE #	1,572	430	2,054	625	863	408	1,271	432	6,242	692
DIFF. #	147	36	301	76	39	34	72	18	0	0
DIFF. %	9.35	8.37	14.65	12.16	4.52	8.33	5.66	4.17	0	0

Table 1: The statistics of the five benchmark datasets, where the number of aspects on three sentiment polarities and sentences are reported.[3] We also report the number and percentage of the contrastive cases (DIFF.) where in a sentence the sentiments on aspect(s) are different from the entire sentence.

3 Experimental Settings

3.1 Datasets

Five benchmark datasets, i.e., LAP14 and REST14 (Pontiki et al., 2014), REST15 (Pontiki et al., 2015), REST16 (Pontiki et al., 2016), TWITTER (Dong et al., 2014), are used in our experiments. Specifically, LAP14 is a dataset consists of laptop computer reviews; REST14, REST15, and REST16 consist of restaurant reviews from online users; TWITTER includes tweets collected by querying the Twitter API. For all datasets, we use their official train/test splits and follow Tang et al. (2016b) to clean them by filtering out the aspects with the conflict label[4] as well as the sentences without an aspect. The statistics of the processed five datasets are reported in Table 1, where the numbers of aspects with positive, neutral, and negative polarities are reported. Note that in some datasets, e.g., LAP14 and REST14, there are rather high percentages of sentences (e.g., the sentence in Figure 1) that contain different sentiments towards aspects, as shown in the DIFF. rows in Table 1, which indicates a bigger challenge on ASA comparing to sentiment analysis on an entire sentence.

3.2 Word Dependency Extraction

Similar to previous studies (Wang et al., 2020; Tang et al., 2020) that also require dependency information, we employ the English version of SAPar[5] (Tian et al., 2020e), which is the most effective constituency parser trained on English Pen Tree-Bank (PTB) (Marcus et al., 1993), to obtain the constituency trees of the input text and then convert them into dependency trees by Stanford converter[6]. Therefore, when a dependency tree is built on the entire input text, for each word in the text, one can find its dependent words and types according to the dependency paths on the tree. Consequently, the dependency relations of each word to others can be extended along with the dependency paths and it is not restricted that only one-hop (first-order) relations can be considered in our model. One could easily extend the coverage of word dependencies with two- or three-hop relations from a given word, which are known as second- and third-order dependencies, e.g., "*poor* $\rightarrow$ *service* $\rightarrow$ *bar*" in Figure 1 is a second-order dependency relation.

As described in §2.1, extracting first-order word dependencies is straightforward; to extend it with higher order ones, we follow the same principle to extract word dependencies and assign dependency types as follows: (1) for the governor w_g of the target word w, we collect all its governor and dependents (except for w) associated with w_g's inbound and outbound dependency types, respectively; (2) for each dependent w_d of w, we find all dependents of w_d and use outbound dependency types to represent w_d's dependent words; (3) we include all context words and their corresponding dependency types collected in (1) and (2) as the input to KVMN for w and repeat the process for further higher order word dependencies.

For example, in the input text in Figure 1, the second-order word dependencies and types for "*service*" are started from its governor "*poor*" and dependent "*bar*". Then for "*poor*", we collect its governor "*fantastic*" with an inbound dependency type of "*advcl*", and dependents "*although*" and "*is*" with the outbound dependency types of "*mark*" and "*cop*", respectively. For "*bar*", it is not able to expand because it has no dependent,

---

because that many sentences have more than one aspect and such aspects usually have contrastive sentiment polarities.

[4] The "*conflict*" label is used in LAP14, REST14/16 to identify aspects that have conflict sentiment polarities.

[5] `https://github.com/cuhksz-nlp/SAPar`

[6] We use the converter of version 3.3.0 from `https://stanfordnlp.github.io/CoreNLP/index.html`.

Models	Lap14		Rest14		Rest15		Rest16		Twitter	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
BERT-base	77.90	73.30	84.11	76.66	83.02	67.92	89.38	64.98	73.27	71.52
+ $\mathcal{X}$-KVMN	78.37	74.18	84.46	**78.44**	84.14	66.12	90.36	72.77	74.13	72.16
+ $\mathcal{A}$-KVMN	**79.78**	**76.14**	**85.98**	77.94	**84.14**	**68.49**	**90.52**	**73.15**	**75.14**	**73.68**
+ $\mathcal{XA}$-KVMN	78.53	75.00	85.09	78.32	83.77	66.57	90.36	72.20	74.13	73.11
BERT-large	78.68	73.75	85.17	77.94	83.21	70.55	90.52	72.88	74.13	73.04
+ $\mathcal{X}$-KVMN	79.31	75.58	86.34	79.63	84.14	70.93	92.13	77.15	74.56	73.07
+ $\mathcal{A}$-KVMN	**80.41**	**77.38**	**86.88**	**80.92**	**84.70**	**72.71**	**92.48**	**79.54**	**76.59**	**74.91**
+ $\mathcal{XA}$-KVMN	80.16	77.20	86.70	79.95	84.58	71.05	91.83	77.28	74.57	72.76

Table 2: Experimental results (accuracy and F1 scores) of using different encoders (BERT-base and BERT-large) with and without KVMN on five benchmark datasets, where $\mathcal{X}$, $\mathcal{A}$, and $\mathcal{XA}$ refer to that KVMN models word dependencies from $\mathcal{X}$ only, $\mathcal{A}$ only, and $\mathcal{X} + \mathcal{A}$, respectively.

the collection thus stops here. Therefore, the resulted words (keys) in second-order dependencies and their corresponding dependency types (values) for "*service*" are $\mathcal{K}_{11} = \{bar, poor, fantastic, although, mark\}$, and $\mathcal{V}_{11} = \{bar\_compound, poor\_nsubj, fantastic\_advcl, although\_mark, is\_cop\}$, respectively.

3.3 Implementation Details

We adopt BERT-base-uncased and BERT-large-uncased[7] as the encoders in our approach, which are demonstrated to be the most effective encoders for many NLP tasks (Straková et al., 2019; Baldini Soares et al., 2019; Xu et al., 2019). In our experiments, we use their default settings for the two BERT encoders (i.e., for BERT-base-uncased, we use 12 layers with 768 dimensional hidden vectors; and for BERT-large-uncased, we use 24 layers with 1024 dimensional hidden vectors). For all experiments, we use Adam optimizer (Kingma and Ba, 2014) and try different combinations of learning rates, dropout rates, and batch size.[8] In addition, we apply Xavier initialization (Glorot and Bengio, 2010) on all trainable parameters including the embeddings for keys and values in the KVMN. Moreover, we use the cross-entropy loss function to optimize our model and follow the convention to evaluate our models via accuracy and macro-averaged F1 scores over all sentiment polarities, i.e., positive, neutral and negative.

4 Experimental Results

4.1 Effect of Using Word Dependencies

In the main experiments, we test our model with and without integrating word dependencies by KVMN, where both the base and large BERT encoders are used. In detail, when leveraging word dependencies, we run experiments on our proposed model to explore the effect of learning from different parts of the input, i.e., we try word dependencies from three sources: $\mathcal{X}$ only, $\mathcal{A}$ only, and both $\mathcal{X}$ and $\mathcal{A}$ (see §2.2). Experimental results are reported in Table 2 with the prefixes of KVMN denoting which part is encoded from.

There are several observations. First, KVMN works well with both the base and large BERT. Although BERT baselines have already achieved good performance, improvements of our proposed model over the baselines are observed on all datasets with respect to both accuracy and F1 scores. Second, among the three settings of encoding from different parts of the input (i.e., $\mathcal{X}$, $\mathcal{A}$, $\mathcal{X}$+$\mathcal{A}$) in KVMN, in most datasets (except for Twitter), the highest performance is observed on "$\mathcal{A}$-KVMN". These results comply with the intuition where extracting and learning word dependencies from $\mathcal{A}$ ensures KVMN only incorporates the information from the content directly associated with the aspect words, thus focuses the model on the words that are most likely to be helpful on ASA for a particular aspect in a sentence. Third, although the overall performance of $\mathcal{X}$-KVMN and $\mathcal{XA}$-KVMN are not as good as that of $\mathcal{A}$-KVMN, they are still better than the baselines without using word dependencies. Especially for $\mathcal{X}$-KVMN, where word dependencies are extracted from the entire sentence, in this case,

---

[7] We obtain the BERT models from `https://github.com/huggingface/pytorch-pretrained-BERT`.

[8] We report the hyper-parameter settings of different models, as well as their size and running speed, in the Appendix.

Order	Lap14			Rest14			Rest15			Rest16			Twitter		
	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.
1st	77.59	73.00	26.89	84.28	75.90	27.05	83.76	67.06	32.26	90.03	70.39	32.50	74.28	73.31	22.26
2nd	**79.78**	**76.14**	52.47	**85.98**	**77.94**	53.83	**84.14**	**68.49**	60.95	**90.52**	**73.15**	61.19	**75.14**	**73.68**	44.70
3rd	78.99	74.60	72.55	85.35	77.78	75.16	82.83	62.81	80.44	89.54	66.56	80.66	74.27	72.31	67.27

(a) BERT-base

Order	Lap14			Rest14			Rest15			Rest16			Twitter		
	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.	Acc	F1	Cvge.
1st	80.25	76.74	26.89	86.43	79.55	27.05	84.33	69.47	32.26	92.12	79.09	32.50	75.43	73.45	22.26
2nd	**80.41**	**77.38**	52.47	**86.88**	80.92	53.83	**84.70**	**72.71**	60.95	**92.48**	**79.54**	61.19	**76.59**	**74.91**	44.70
3rd	80.09	76.84	72.55	86.52	**81.02**	75.16	84.14	68.05	80.44	92.27	79.20	80.66	76.16	74.85	67.27

(b) BERT-large

Table 3: Experimental results of our models with the best setting (i.e., using base and large BERT with $\mathcal{A}$-KVMN) of using dependency relations on different (i.e., 1st, 2nd and 3rd) orders. Average percentage of words in a sentence covered by word dependencies on different orders are also reported in the Cvge. column.

the dependency information also helps ASA even though it introduces some noise to the task when the entire sentence possesses a different sentiment polarity (as shown in the Diff. rows in Table 1), while such noise contributes to its inferior performance to the $\mathcal{A}$-KVMN setting. Therefore, for the case that the sentiment is agreed between the entire sentence and its aspect (e.g., Twitter dataset is in this case according to Table 1), $\mathcal{X}$-KVMN and $\mathcal{A}$-KVMN have similar performance.

4.2 Effect of Different Dependency Orders

Previous experiments showed the effectiveness of our model with KVMN on first-order word dependencies. In this experiment, we use the best setting (i.e., models using $\mathcal{A}$-KVMN) for base and large BERT and run them with encoding higher-order dependencies to further investigate the effectiveness of our model with more dependency information. Particularly, we try second- and third-order word dependencies and compare their results with the previous first-order ones. The results on all datasets, as well as average coverage (%) of words in each sentence with respect to different dependency orders,[9] are reported in Table 3, where (a) and (b) show the results of models with BERT-base and BERT-large encoders, respectively. From the results, it is found that in most cases (e.g., for both base and large BERT), models using second-order word dependencies achieve the overall highest performance, which can be explained by that first-order dependency for aspect words is not enough to cover enough salient information helping ASA. This is a common phenomenon when negation is included in a sentence. For example, in "*the pizza is not good*", for its aspect "*pizza*", whose first-order dependencies only link "*pizza*" with "*good*", the classifier is thus misled to predict a positive sentiment polarity. Compared to using second-order word dependencies, third-order dependencies in general do not provide further improvement to ASA, which owes to the reason that more irrelevant information is introduced to the encoder thus distract the model for final prediction. In fact, third-order dependencies lead to that around 75% words in each sentence are fed into KVMN, which could severely affect ASA by sentence-level sentiment polarities, and eventually harm model performance especially when an aspect-level sentiment differs from the sentence-level sentiment.

4.3 Comparison with Previous Studies

To further demonstrate the effectiveness of our approach, we compare our best-performing model, i.e., the BERT-large encoder with second-order word dependencies incorporated through $\mathcal{A}$-KVMN, with previous studies, where the comparisons on all datasets are reported in Table 4, where the results of BERT-large baseline, as well as the ones using BERT-base, are also reported for references. It is observed that, our model consistently outperforms the BERT-large baseline on all datasets and achieves state-of-the-art on three of them (i.e., Lap14, Rest15, Rest16) in terms of

---

[9] This metric is used to present how many words in each input sentence are involved when different orders are applied for extracting word dependencies, so as to illustrate how much information in a sentence is helpful for ASA.

MODELS	LAP14		REST14		REST15		REST16		TWITTER	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
ATAE-LSTM (WANG ET AL., 2016)	68.70	-	77.20	-	-	-	-	-	-	-
MEMNET (TANG ET AL., 2016B)	72.21	-	80.95	-	-	-	-	-	-	-
IAN (MA ET AL., 2017)	72.10	-	78.60	-	-	-	-	-	-	-
RAM (CHEN ET AL., 2017)	74.49	71.35	80.23	70.80	-	-	-	-	69.36	67.30
PBAN (GU ET AL., 2018)	74.12	-	81.16	-	-	-	-	-	-	-
TNET-AS (LI ET AL., 2018B)	76.54	71.75	80.69	71.27	-	-	-	-	74.97	73.60
PRET+MULT (HE ET AL., 2018B)	71.15	67.46	79.11	69.73	81.30	68.74	85.58	69.76	-	-
SYNATT (HE ET AL., 2018A)	72.57	69.13	80.63	71.32	81.67	66.05	64.61	67.45	-	-
PF-CNN (HUANG AND CARLEY, 2018)	70.06	-	79.20	-	-	-	-	-	-	-
MGAN (FAN ET AL., 2018)	75.39	72.47	81.25	71.94	-	-	-	-	72.54	70.81
CAN (HU ET AL., 2019)	-	-	84.28	74.45	78.58	54.72	-	-	-	-
TRANSCAP (CHEN AND QIAN, 2019)	73.87	70.10	79.55	71.41	-	-	-	-	-	-
IACAPSNET (DU ET AL., 2019)	76.80	73.29	81.79	73.40	-	-	-	-	75.01	73.81
ANTM (MAO ET AL., 2019)	75.84	72.49	82.49	72.10	-	-	-	-	72.35	69.45
CDT (SUN ET AL., 2019)	77.19	72.99	82.30	74.02	-	-	85.58	69.93	74.66	73.66
ASGCN (ZHANG ET AL., 2019)	75.55	71.05	81.22	72.94	79.89	61.89	88.99	67.48	72.69	70.59
†TD-GAT-BERT (HUANG AND CARLEY, 2019)	80.10	-	83.00	-	-	-	-	-	-	-
†AEN-BERT (SONG ET AL., 2019)	79.93	76.31	83.12	73.76	-	-	-	-	74.71	73.13
†BERT-PT (XU ET AL., 2019)	78.07	75.08	84.95	76.96	-	-	-	-	-	-
†R-GAT-BERT (WANG ET AL., 2020)	78.21	74.07	86.60	**81.35**	-	-	-	-	76.15	74.88
†DGEDT-BERT (TANG ET AL., 2020)	79.8	75.6	86.3	80.0	84.0	71.0	91.9	79.0	**77.9**	**75.4**
BERT-BASE	77.90	73.30	84.11	76.66	83.02	67.92	89.38	64.98	73.27	71.52
OUR BEST MODEL (BERT-BASE)	*79.78	*76.14	*85.98	*77.94	*84.14	*68.49	*90.52	*73.15	*75.14	*73.68
†BERT-LARGE	78.68	73.75	85.17	77.94	83.21	70.55	90.52	72.88	74.13	73.04
†OUR BEST MODEL (BERT-LARGE)	***80.41**	***77.38**	***86.88**	*80.92	***84.70**	***72.71**	***92.48**	***79.54**	*76.59	*74.91

Table 4: Performance Comparison (on accuracy and F1 scores) of our best model (BERT-LARGE + $\mathcal{A}$-KVMN with second-order word dependencies) with previous studies on all datasets. The results of BERT-large baseline are also reported for references. Models that use BERT-large as the encoder are marked by "†". The results marked by "*" indicate that our model is significantly better than the corresponding baseline model (t-test with $p < 0.05$).

both accuracy and F1 scores. Specifically, compared with previous studies that also leverage dependency information, our approach outperforms He et al. (2018a); Sun et al. (2019); Huang and Carley (2019); Zhang et al. (2019) on all dataset and outperforms Wang et al. (2020) and Tang et al. (2020) on most datasets. This observation is valid because, in previous models, they are weighting or averaging hidden vectors of the (aspect related) words rather than on the relations, and omitting dependency types which provide guidance to emphasize some useful relations, e.g., the "*amod*" (i.e., adjectival modifier) type identifies that an adjectival modifier could be the sentiment words of a corresponding aspect. Therefore, the superiority of our model comes from two aspects, weighting word-word relations and leveraging dependency types. KVMN highlights salient dependency relations and learns from them and their dependency types, which alleviates the influence of noisy dependency information. In addition, we note that our approach achieves inferior results on TWITTER dataset compared with Tang et al. (2020). One possible explanation is that a dependency parser trained in the general domain can get inferior parsing results on TWITTER texts from the social media domain, which makes it harder for our approach to improve the BERT-large baseline compared to other datasets. Nevertheless, the effectiveness of our approach is still valid given that our approach outperforms Tang et al. (2020) on all other datasets.

5 Analyses

5.1 Ablation Study

To confirm the validity of using both word relations (keys) and their corresponding dependency types (values) for ASA, we conduct an ablation study by learning from either part of the two types of dependency information. We choose the models using BERT-base and BERT-large with our best setting (i.e., models with second-order word dependencies and $\mathcal{A}$-KVMN) for this study and adapt the KVMN module to key-only or value-only input. The experimental results on all benchmark datasets are reported in Table 5, where keys or values are ablated. It is clearly indicated in the table that, for models with different encoders (i.e., base and large BERT), the model performance drops on all datasets if either keys or values are excluded. Specifically, in most cases, the drop of performance (especially on accuracy) is higher when keys are ablated ("– KEYS"), comparing with the ablation of values ("– VALUES"). This phenomenon indicates the context words, which attracts much attention from previous studies (He et al., 2018a; Sun et al., 2019; Zhang et al., 2019; Huang and Carley, 2019), play a more important role compared with their

Setting	Lap14		Rest14		Rest15		Rest16		Twitter	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Full Model	**79.78**	**76.14**	**85.98**	**77.94**	**84.14**	**68.49**	**90.52**	**73.15**	**75.14**	**73.68**
− Keys	79.47	75.29	84.91	77.33	82.65	67.95	89.54	72.21	74.41	73.32
Δ	−0.31	−0.85	−1.07	−0.61	−1.49	−0.54	−0.98	−0.94	−0.73	−0.36
− Values	79.62	75.45	85.18	77.71	83.21	68.20	89.71	72.57	74.70	73.45
Δ	−0.16	−0.69	−0.80	−0.23	−0.93	−0.29	−0.81	−0.58	−0.44	−0.23

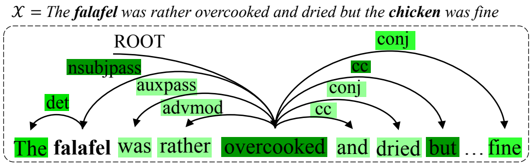
(a) BERT-base

Setting	Lap14		Rest14		Rest15		Rest16		Twitter	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Full Model	**80.41**	**77.38**	**86.88**	**80.92**	**84.70**	**72.71**	**92.48**	**79.54**	**76.59**	**74.91**
− Keys	79.94	77.06	86.43	79.60	83.84	70.89	91.50	77.99	75.58	72.94
Δ	−0.47	−0.32	−0.45	−1.32	−0.86	−1.82	−0.98	−1.55	−1.01	−1.97
− Values	80.26	77.34	86.63	80.82	84.40	70.31	91.82	78.53	75.87	73.69
Δ	−0.15	−0.04	−0.25	−0.10	−0.30	−2.04	−0.66	−1.01	−0.72	−1.22

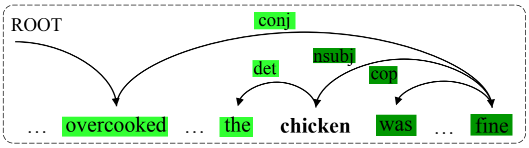
(b) BERT-large

Table 5: Results on five datasets from our full models (base and large BERT with $\mathcal{A}$-KVMN and second-order word dependencies) and its variants where keys ("− Keys") and values ("− Values") are ablated. Δ refers to the drop of accuracy and F1 score when keys or values are excluded from the full model.

dependency types. Still, one cannot deny the contribution of dependency types because the drop is still significant if values are excluded, where even on some datasets (e.g., Rest15 and Rest16) higher drops are observed on F1 than Key ablation. The results for this ablation study demonstrate that dependency type is of high importance to improve ASA if they are appropriately encoded.



(a): $\mathcal{A}$ = ***falafel*** y^* = *negative*



(b): $\mathcal{A}$ = ***chicken*** $y*$ = *positive*

Figure 2: Illustration of an example sentence with two aspects in different sentiment polarities. For each aspect, weights (from our best model) assigned to dependent words and dependency types are visualized with colors, where darker color refers to higher weights.

5.2 Case Study

To illustrate the effect of KVMN module on weighting salient word dependencies and thus improve ASA, we conduct a case study on the sentence "*The falafel was rather overcooked and dried but the chicken was fine*" shown in Figure 2, in which it contains two aspects with contrast sentiment polarities, i.e., *negative* towards "*falafel*" and *positive* towards "*chicken*". For each aspect, we run our best model (BERT-large + $\mathcal{A}$-KVMN with second-order word dependencies), and visualize the weights ($p_{i,j}$ in Eq. (2)) assigned to all associated dependency types and their corresponding words, where darker color refers to higher weights.

For the first aspect "*falafel*" (Figure 2(a)), although there are some adjectives carrying opposite sentiment polarities within its second-order relations, KVMN successfully distinguishes "*overcooked*" is more important to it and assigns a relatively higher weight. This is because that the corresponding type ("*nsubjpass*", passive nominal subject) to "*overcooked*" is intensively highlighted so that the model identifies it as the main sentiment carrier for the aspect word "*falafel*" where other adjectives (i.e., "*dried*" and "*fine*") share the "*conj*" (conjunction) type and are distantly related to the aspect words, making them less important.

For the other aspect "*chicken*" (Figure 2(b)), similar to the first one, both "*overcooked*" and "*fine*" are included in its associated context words. In this case, "*fine*" is more closely dependent on "*chicken*" than "*overcooked*", where it has a "*nsubj*" (noun subject) type showing a predicate role thus receives higher weight from KVMN, resulting in a positive sentiment polarity prediction towards "*chicken*". Overall, this case study perfectly explains the effectiveness of our model, where two aspects share the same context and the only change is the dependency information ($\mathbf{o_i}$) comes from KVMN. Therefore, the different prediction results for the two aspects suggest that KVMN appropriately learns from salient dependency relations and types for each aspect, where different types have their own capabilities to enhance ASA accordingly (e.g., "*nsubj*" may contribute more than "*conj*").

6 Related Work

Different from sentiment analysis for large granular texts, such as document and sentences, ASA focuses on processing sentiment polarities for a specific aspect (e.g., "*pizza*") or category (e.g., "*food*") in a piece of text. To address this task, early approaches (Jiang et al., 2011; Dong et al., 2014) followed the sentence classification paradigm and recent studies enhanced it as a mission of sentence-aspect pair classification with applying neural approaches (Wang et al., 2016; Tang et al., 2016a; Ma et al., 2017; Chen et al., 2017; Xue and Li, 2018; Li et al., 2018b; Hu et al., 2019; Xu et al., 2019) such as recurrent models (e.g., bi-LSTM) and pre-trained encoders (e.g., BERT) for effectively capturing contextual information. In addition to improving the input form, advanced models such as memory networks (Tang et al., 2016b; Chen et al., 2017; Wang et al., 2018; Zhu and Qian, 2018; Mao et al., 2019), attention mechanism (Wang et al., 2016; Ma et al., 2017; Hazarika et al., 2018), capsule networks (Du et al., 2019; Chen and Qian, 2019; Jiang et al., 2019), GNN (Huang and Carley, 2019; Sun et al., 2019; Zhang et al., 2019; Wang et al., 2020), and transformer (Tang et al., 2020) are applied to this task, with other studies leveraging external resources, including position information (Gu et al., 2018), document information (He et al., 2018b), commonsense knowledge (Ma et al., 2018), etc. Among all resources, syntactic information was proved to be the most effective one and successfully adopted in recent studies with GNN (Huang and Carley, 2019; Sun et al., 2019; Zhang et al., 2019). Compared with previous studies, our approach offers an alternative way to use KVMN and syntactic information for ASA. Consider those studies using memory networks where their memories are represented by contextual features of the aspect terms, dependency information was not leveraged in their work. In addition, compared with those approaches leveraging word dependencies (i.e., using attention mechanism or GNN), where they not only omitted useful dependency information such as relation types, but also demanded a complicated model structure in doing so, our approach ensures comprehensively encoding from both word-word relations and their dependency types, and models them in an efficient way by KVMN.

7 Conclusion

In this paper, we propose an effective neural approach to improve ASA with word dependencies by KVMN, where for each aspect term, we firstly extract the words associated to it according to the dependency parse of the input sentence and their corresponding dependency relation types, then use KVMN to encode and weight such information to enhance ASA accordingly. In our approach, not only word-word relations but also their dependency types are leveraged in a KVMN, which to our best knowledge are the first attempts in all related syntax-driven studies for ASA. Experimental results on five widely used benchmark datasets demonstrate the effectiveness of our approach, and shows that second-order word dependency is the best choice for ASA, where the new state-of-the-art results are achieved on three datasets. Moreover, further analyses illustrate the validity of applying KVMN on both dependency relation and type information, especially the effectiveness of dependency types, which are often omitted in previous studies.

Acknowledgements

This work is supported by The Chinese University of Hong Kong (Shenzhen) under University Development Fund UDF01001809. This work is also partially supported by NSFC under the project "The Essential Algorithms and Technologies for Standardized Analytics of Clinical Texts" (12026610).

References

Nastaran Babanejad, Ameeta Agrawal, Aijun An, and Manos Papagelis. 2020. A Comprehensive Analysis of Preprocessing for Word Representation Learning in Affective Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5799–5810, Online.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279, Barcelona, Spain (Online).

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 452–461.

Zhuang Chen and Tieyun Qian. 2019. Transfer capsule network for aspect level sentiment classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 547–556.

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

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2019. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. *ArXiv*, abs/1911.00720.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 49–54.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Tong Xu, and Ming Liu. 2019. Capsule network with interactive attention for aspect-level sentiment classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5492–5501.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Shuqin Gu, Lipeng Zhang, Yuexian Hou, and Yin Song. 2018. A position-aware bidirectional attention network for aspect-level sentiment analysis. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 774–784.

Chaoyu Guan, Yuhao Cheng, and Hai Zhao. 2019. Semantic Role Labeling with Associated Memory Network. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3361–3371, Minneapolis, Minnesota.

Devamanyu Hazarika, Soujanya Poria, Prateek Vij, Gangeshwar Krishnamurthy, Erik Cambria, and Roger Zimmermann. 2018. Modeling inter-aspect dependencies for aspect-based sentiment analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 266–270.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018a. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1121–1131.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018b. Exploiting document knowledge for aspect-level sentiment classification. *arXiv preprint arXiv:1806.04346*.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2019. CAN: Constrained Attention Networks for Multi-Aspect Sentiment Analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4593–4602.

Binxuan Huang and Kathleen M Carley. 2018. Parameterized Convolutional Neural Networks for Aspect Level Sentiment Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1091–1096.

Binxuan Huang and Kathleen M Carley. 2019. Syntax-Aware Aspect Level Sentiment Classification with Graph Attention Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5472–5480.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 151–160.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A Challenge Dataset and Effective Models for Aspect-Based Sentiment Analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6281–6286.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Junjie Li, Haitong Yang, and Chengqing Zong. 2018a. Document-level multi-aspect sentiment classification by jointly modeling users, aspects, and overall ratings. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 925–936.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018b. Transformation Networks for Target-Oriented Sentiment Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956.

Bin Liang, Jiachen Du, Ruifeng Xu, Binyang Li, and Hejiao Huang. 2019. Context-aware Embedding for Targeted Aspect-based Sentiment Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4678–4683, Florence, Italy.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4068–4074.

Yukun Ma, Haiyun Peng, and Erik Cambria. 2018. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Qianren Mao, Jianxin Li, Senzhang Wang, Yuanning Zhang, Hao Peng, Min He, and Lihong Wang. 2019. Aspect-based sentiment classification with attentive neural turing machines. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5139–5145.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245, Online.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35.

Dominic Seyler, Tatiana Dembelova, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. A Study of the Importance of External Knowledge in the Named Entity Recognition Task. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 241–246.

Yan Song, Jiaqing Guo, and Dongfeng Cai. 2006. Chinese Word Segmentation Based on an Approach of Maximum Entropy Modeling. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 201–204, Sydney, Australia.

Yan Song, Chunyu Kit, and Xiao Chen. 2009. Transliteration of Name Entity via Improved Statistical Translation on Character Sequences. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 57–60, Suntec, Singapore.

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based Training Data Selection for Domain Adaptation. In *Proceedings of COLING 2012: Posters*, pages 1191–1200, Mumbai, India.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 143–152.

Yan Song and Shuming Shi. 2018. Complementary Learning of Word Embeddings. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4368–4374.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 175–180, New Orleans, Louisiana.

Yan Song, Yuanhe Tian, Nan Wang, and Fei Xia. 2020. Summarizing Medical Conversations via Identifying Important Utterances. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 717–729, Barcelona, Spain (Online).

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese Word Segmentation. In *LREC*, pages 3853–3860.

Yan Song and Fei Xia. 2013. A Common Case of Jekyll and Hyde: The Synergistic Effect of Using Divided Source Training Data for Feature Augmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 623–631, Nagoya, Japan.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification. *arXiv preprint arXiv:1902.09314*.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-Level Sentiment Analysis Via Convolution over Dependency Tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5683–5692.

Xingwei Tan, Yi Cai, and Changxi Zhu. 2019. Recognizing Conflict Opinions in Aspect-level Sentiment Classification with Dual Attention Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3417–3422.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective LSTMs for Target-Dependent Sentiment Classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect Level Sentiment Classification with Deep Memory Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency Graph Enhanced Dual-transformer Structure for Aspect-based Sentiment Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online.

Jialong Tang, Ziyao Lu, Jinsong Su, Yubin Ge, Linfeng Song, Le Sun, and Jiebo Luo. 2019. Progressive Self-Supervised Attention Learning for Aspect-Level Sentiment Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 557–566.

Yuanhe Tian, Wang Shen, Yan Song, Fei Xia, Min He, and Kenli Li. 2020a. Improving Biomedical Named Entity Recognition with Syntactic Information. *BMC Bioinformatics*, 21:1471–2105.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 2020b. Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8286–8296, Online.

Yuanhe Tian, Yan Song, and Fei Xia. 2020c. Joint Chinese Word Segmentation and Part-of-speech Tagging via Multi-channel Attention of Character N-grams. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2073–2084, Barcelona, Spain (Online).

Yuanhe Tian, Yan Song, and Fei Xia. 2020d. Supertagging Combinatory Categorial Grammar with Attentive Graph Convolutional Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6037–6044, Online.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020e. Improving Constituency Parsing with Span Attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703, Online.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020f. Improving Chinese Word Segmentation with Wordhood Memory Networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8274–8285, Online.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational Graph Attention Network for Aspect-based Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online.

Shuai Wang, Sahisnu Mazumder, Bing Liu, Mianwei Zhou, and Yi Chang. 2018. Target-sensitive memory networks for aspect sentiment classification. In

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 957–967.

Yequan Wang, Minlie Huang, Li Zhao, et al. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335.

Wei Xue and Tao Li. 2018. Aspect Based Sentiment Analysis with Gated Convolutional Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based Sentiment Classification with Aspect-specific Graph Convolutional Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4560–4570.

Peisong Zhu and Tieyun Qian. 2018. Enhanced Aspect Level Sentiment Classification with Auxiliary Memory. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1077–1087.

Appendix

A. Model Size and Running Speed

Table 6 reports the number of trainable parameters and inference speed (sentences/second)[10] of baseline (i.e., the ones without using KVMN and the dependency information) and our best performing models (i.e., the ones with $\mathcal{A}$-KVMN and the second-order dependencies) on all datasets.

B. Hyper-parameter Settings

Table 7 reports the hyper-parameters we used for tuning our models. For each dataset, we try all combinations of the hyper-parameters and report the one with the highest accuracy score in our paper.

---

[10] The test is performed on a Quadro RTX 6000 GPU.

MODELS	LAP14		REST14		REST15		REST16		TWITTER	
	PARA.	SPEED	PARA.	SPEED	PARA.	SPEED	PARA.	SPEED	PARA.	SPEED
BERT-BASE	109.5M	37.1	109.5M	38.1	109.5M	37.3	109.5M	38.5	109.5M	38.2
FULL MODEL	125.2M	34.6	125.2M	30.6	125.2M	34.5	125.2M	34.6	147.2M	34.3
BERT-LARGE	335.1M	20.0	335.1M	20.1	335.1M	20.5	335.1M	20.5	335.1M	19.6
FULL MODEL	356.1M	19.6	356.1M	19.0	356.1M	19.2	356.1M	19.6	385.4M	19.8

Table 6: The number of trainable parameters (PARA.) and the running speed (sentences/second) on the test sets of the baseline models (the ones without using KVMN and the dependency information) and our best performing models (the ones with $\mathcal{A}$-KVMN and the second-order dependencies).

HYPER-PARAMETER TYPES	TRIED HYPER-PARAMETER VALUES
LEARNING RATE	$e^{-5}, 2e^{-5}, 3e^{-5}, 4e^{-5}, 5e^{-5}, 6e^{-5}, 7e^{-5}, 8e^{-5}, 9e^{-5}, e^{-4}$
DROPOUT RATE	$0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$
BATCH SIZE	$8, 16, 32$

Table 7: The hyper-parameters for tuning our models.

Author Index